AVALANCHE

COMBATING TRANSNATIONAL ILLEGAL ACTIVITY THROUGH INTERNATIONAL COOPERATION

Grant agreement	101168393
Project acronym	AVALANCHE
Full title	Countering Crime and Terrorism and their Links to Transnational Illegal Activities by Fostering International Cooperation
Start date	1 Oct 2024
End date	30 Sep 2026
Call & topic	HORIZON-CL3-2023-SSRI-01-02 Accelerating uptake through open proposals for advanced SME innovation

D4.1
AVALANCHE Reference Architecture

This work is part of the AVALANCHE project. AVALANCHE has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement no. 101168393

Document Information

Work package	WP4
Deliverable number and title	D4.1 AVALANCHE Reference Architecture
Version	1.0
Submission date	03/07/2025
Leading author(s)	Elena Valari (ITML)
Contributor(s)	Christos Kotronis (UBI), Giorgos Pantelis (UBI), Tayyar Louai (NET), Niklas Palaghias (QAD), Leoni Chondromatidou (QAD), Alexandru VLADUTA (SPP), Ovidiu Dumitrache (SPP)
Reviewing peer(s)	Niklas Palaghias (QAD), Evangelos Agorogiannis (NET),
Dissemination level	PU-Public

Revision history

Version	Issue Date	Status	Notes	Distribution
0.1	21/03/2025	Draft	2nd version of ToC	Elena Valari (ITML)
0.2	05/05/2025	Draft	2nd version of ToC	Elena Valari (ITML)
0.3	02/06/2025	Draft	1st Integrated version	Christos Kotronis (UBI) Giorgos Pantelis (UBI) Tayyar Louai (NET) Alex Vladuta (SPP) Elena Valari (ITML) Niklas Palagias (QAD) Leoni Chondromatidou (QAD)
0.4	11/06/2025	Draft	2nd Integrated version	Elena Valari (ITML)
0.5	17/06/2025	Draft	KPIs, References	Christos Kotronis (UBI) Giorgos Pantelis (UBI) Niklas Palagias (QAD) Elena Valari (ITML)
0.6	19/06/2025	Draft	3rd Integrated version	Christos Kotronis (UBI) Elena Valari (ITML)
0.7	24/06/2025	Draft	4th Integrated version	Alex Vladuta (SPP) Ovidiu Dumitrache (SPP) Christos Kotronis (UBI) Giorgos Pantelis (UBI) Elena Valari (ITML)
0.8	02/07/2025	Draft	5th Integrated Version	Elena Valari (ITML) Nikos Evangeliou (ITML)
1.0	03/07/2025	Final	Final Version to be submitted	Mariza Konidi (UBI)

This work is part of the AVALANCHE project. AVALANCHE has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement no. 101168393

Table of Contents

List of Figures

List of Tables

This work is part of the AVALANCHE project. AVALANCHE has received funding from the European Union’s Horizon Europe research and innovation programme under Grant Agreement no. 101168393

Abbreviations

Abbreviation	Definition
API	Application Programming Interface
BD	Big Data
BERT	Bidirectional Encoder Representations from Transformers
CI/CD	Continuous Integration and Continuous Delivery
CKAN	Comprehensive Knowledge Archive Network
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
DevOps	Development and Operations
DFB	Data Fusion Bus
FR	Functional Requirements
GA	Grant Agreement
GUI	Graphical User Interface
HTML	Hyper Text Markup Language
IAM	Identification Management System
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LEA	Law Enforcement Agency
LLM	Large Language Model
MoSCoW	Must have, Should have, Could have, Won't have (a prioritization method)
NLP	Natural Language Processing
NLU	Natural Language Understanding
NFR	Non-Functional Requirements
OSINT	Open-Source Intelligence
REQ-DFA	Deepfake Analysis Requirements
REQ-DFD	Deepfake Detection Requirements
REQ-DISA	Disinformation Analysis Requirements
REQ-DISD	Disinformation Detection Requirements
REQ-HSA	Hate Speech Analysis
REQ-HSD	Hate Speech Detection
SoTA	State-of-the-art
SQL	Structured Query Language
SysML	Systems Modeling Language
TOR	The Onion Router
UC	Use Case
UI	User Interface
URL	Uniform Resource Locator
WP	Work Package

This work is part of the AVALANCHE project. AVALANCHE has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement no. 101168393

Executive Summary

This deliverable presents the architectural design of the AVALANCHE platform, detailing its system structure, key components, and supporting technologies aimed at detecting, analysing, a harmful online content focusing on 3 use cases (disinformation, hate speech and deepfake).

The platform is designed using a 3-tier architecture that promotes scalability, maintainability, and flexibility. Through its secure, user-friendly web interface, the Presentation Layer offers robust functionalities. These include advanced visualization tools, personalized analytics, and custom reporting capabilities. To enhance understanding, accompanying mock-ups and user journey scenarios are provided. User access and authentication are managed through an integrated Identity and Access Management (IAM) system, ensuring secure interaction with the platform.

At the core of the architecture lies the Application Layer, which hosts the orchestration engine responsible for coordinating complex workflows across the platform's services. This includes managing detection, analysis, and data collection processes to efficiently address user requests. The Crawling Service collects relevant data from the web, feeding it into the system for further analysis. The Detection and Analysis Services provide processing for tasks such as disinformation detection, hate speech analysis, deepfake identification, and sentiment and behavioural insights.

The Data Layer supports diverse data storage needs through multiple technologies, including relational databases, search engines, and object storage. Data discoverability and sharing are enhanced via integrated data management tools, while asynchronous communication between services is facilitated by a message broker, improving system resilience and modularity.

An additional feature of the AVALANCHE architecture is its Security Layer, which ensures data protection, integrity, and regulatory compliance. This layer includes strong identity management, encryption, anonymization techniques, and secure communication protocols to maintain user trust and secure sensitive information.

Beyond serving as technical documentation, this architecture provides a flexible integration framework that enables modular development, testing, and deployment of individual components. This approach supports rapid iteration and simplifies maintenance throughout the project lifecycle. Moving forward, focus will shift to detailed design, implementation, and integration of each module, culminating in a secure, intelligent, and scalable platform ready to meet the evolving challenges posed by digital misinformation and harmful online content.

1. Introduction

1.1 Purpose of the Document

This deliverable, D4.1 – AVALANCHE Reference Architecture, provides a comprehensive and structured blueprint of the AVALANCHE system architecture, developed in direct response to user-driven requirements. Building upon the findings of WP3 – ANALYSE: Gap Analysis, End User Requirements & Use Cases Specification, this document bridges the transition from high-level user needs to a concrete architectural framework. It outlines both the functional (FRs) and non-functional requirements (NFRs) that guide system design and defines the core components of the AVALANCHE solution. Furthermore, it details the interactions, data flows, and communication pathways between these components, ensuring that the architecture is aligned with its intended operational context and performance goals.

The document structure is the following:

- Chapter 1 serves an introduction to the document, outlining its purpose and how it connects with other project deliverables.
- Chapter 2 describes the overall system requirements including both functional and non-functional requirements, as derived from the user needs.
- Chapter 3 presents the AVALANCHE solution architecture, describing the system's actors, user journeys, and a comprehensive overview of its main components.
- Chapter 4 addresses the SOTA analysis outcomes based on the SOTA finding from D3.1.
- Chapter 5 presents the deployment and infrastructure planning, including the deployment design and the initial CI/CD and DevOps strategy.
- Chapter 5 presents technical KPIs for the key components of the AVALANCHE solution at this design stage.

1.2 Brief overview and deliverable interlink with other project deliverables

D4.1, which provides a detailed overview of the overall architecture and features of the AVALANCHE platform, is informed by the insights and requirements established in D3.1: "State-of-Play, End User Requirements and KPIs" [1]. Furthermore, this deliverable will serve as the basis for the development-focused work packages (WPs) WP5 and WP6, directly providing input to the following deliverables:

- D5.1 Solutions for Data Collection, Behavioural Analysis & Reporting
- D5.2 Solutions for Decision Support, Data Interconnection & Sharing
- D6.1 AVALANCHE Integration and Unified UI

2. Overall System Requirements

This section defines the overall system requirements for the AVALANCHE solution, providing a clear understanding of both what the system must achieve (Functional Requirements) and the conditions it must operate under (Non-Functional Requirements). These requirements are derived from user needs, expressed as goals, and are systematically transformed into technical specifications that guide the system's design and development.

2.1 Functional requirements

This subsection outlines the functional requirements of the AVALANCHE system, which defines the specific functionalities and operations that the system must perform to meet the user's needs. Each requirement is described in a structured format, ensuring clear understanding and traceability. These requirements originate from user goals [1] and are translated into system functionalities, addressing areas like data collection, analysis, and detection for UC1, secure information exchange for UC2, and user interaction. The tables below provide detailed lists of these functional requirements, along with their unique identifiers, descriptions, and associated priorities and feasibility. The functional requirements are organized in separate tables for each use case, providing a clearer view. Each table provides the requirement's unique identifier, name, description, priority, and feasibility status. These modules collectively form the architecture of the overall solution, which will be detailed in the following section. Feasibility is assessed using the MoSCoW methodology, where: (i) M (Must have) and most S (Should have) requirements are targeted for the first release of the AVALANCHE solution. (ii) The remaining S (Should have) and C (Could have) requirements are planned for the final release at the end of the project.

As mentioned above, the functional requirements are outlined in the following tables, grouped by the main modules/building blocks of the AVALANCHE solution. Beginning with the crawler services stated as REQ-CR (Requirement Crawler), the table below (*Table 1*) lists the functional requirements derived from the corresponding user needs.

Table 1: Crawling Service: Functional Requirements

ID	Name	Priority	Feasibility	Description
REQ-CR-001	Data collection from Surface and Dark Web	High	M	The system must extract data from Surface (and forums), Dark Web.
REQ-CR-002	Blacklist exclusion rules	High	S	Rules can be defined in the blacklist to exclude specific websites and digital formats (e.g., .exe, .pdf, etc.) from the crawling process, ensuring they are blocked from being crawled.
REQ-CR-003	Detect Account/User /Group of Interest for Analysis	High	C	Generate a list of all users within the target group and extract both textual and image content posted by the specified user from forums such as Reddit.
REQ-CR-004	Keyword/Key phrase Filtering in Crawling	High	M	The system enables manual input or file upload of unlimited keywords/key phrases, provides slang/alternative phrases/synonym suggestions, supports special characters (including Romanian), and allows exact word searches.
REQ-CR-005	Search/Query/Task Scheduling and Customization	High	S	The system allows tasks to run multiple times, retrieving only new information, and enables manual execution, pausing, and resuming. Tasks can be scheduled for specific times, days, or periods, and multimedia content will be displayed inline without losing context.

The functional requirements derived from user requirements for disinformation detection and analysis are presented in *Table 2* . At this stage, it is important to note that requirement IDs are coded as REQ-DISD for Disinformation Detection and REQ-DISA for Disinformation Analysis.

Table 2: Disinformation Detection & Analysis Services: Functional Requirements

ID	Name	Priority	Feasibility	Description
REQ-DISD-001	Content and/or Author Authenticity Verification, Source Analysis and Fact-checking Integration	High	S	The system verifies the origins of articles to check alterations, manipulation, or context misrepresentation, with a web sources (dark and surface web) list that can be edited and updated by the analyst.
REQ-DISD-002	Real-Time Monitoring disinformation detection (Near – real time)	Medium	C	The system continuously scans open web sources for potential disinformation in near real time, ensuring timely detection and alerts to maintain the relevance of the information from an operational standpoint.
REQ-DISD-003	Source and Author Reliability Scoring	Medium	C	The system assigns a reliability score to authors of open web sources, dark web sources, and entities from open web sources (such as news outlets) to evaluate their trustworthiness and credibility.
REQ-DISD-004	Misinformation Trend Detection	Medium	C	The system identifies recurring themes, narratives, or trending topics linked to disinformation campaigns, such as false political claims or misleading information.
REQ-DISD-005	Support of multilingual and regional content	High	S	The system detects disinformation in multiple languages and adapts to regional slang and cultural nuances.
REQ-DISD-006	Identify and Flag Disinformation and False Positive	Medium	C	The system automatically flags suspicious content, notifies users, and provides explanations, such as "misleading statistics" or "altered image." It also collects large volumes of near real-time data from both the surface and dark web as content is posted.
REQ-DISA-001	Mapping and Analysis of Disinformation Network	Medium	C	The system maps networks of accounts, groups, and pages that amplify disinformation to

				detect coordinated campaigns or bot activity. It provides clear explainability for the relationships within these networks.
REQ-DISA-002	Sentiment Analysis	High	M	The platform must implement a sentiment analysis algorithm to analyse text and classify it into one of three sentiment categories: positive, negative, or neutral.

Table 3 provides an overview of the functional requirements related to hate speech detection and analysis. At this point, it's worth highlighting that the requirement identifiers follow the convention REQ-HSD for Hate Speech Detection and REQ-HSA for Hate Speech Analysis.

Table 3: Hate speech detection services: Functional Requirements

ID	Name	Priority	Feasibility	Description
REQ-HSD-001	Detect Hate Speech Across Contexts	High	S	The platform detects hate speech across different contexts, including explicit slurs, subtle derogatory language, and coded or indirect expressions. The platform must implement a binary classification algorithm to detect hate speech from text, categorizing content as either "hate" or "non-hate."
REQ-HSD-002	Real-Time Hate Speech Monitoring	High	W	The system continuously opens web sources to detect hate speech in real time.
REQ-HSD-003	Monitor and Detect Hate Speech in Real Time	High	W	The system continuously opens web sources to detect hate speech as it occurs in real time.
REQ-HSD-004	Differentiate Hate Speech from Contextual Usage	High	S	The system distinguishes between genuine hate speech and benign use of potentially offensive terms, considering context such as academic discussions, satire, or non-malicious intent.
REQ-HSD-005	Detect Hate Speech in Romanian and Other Languages with Cultural Nuance	High Low	S - (Romanian) C (any other language)	The system prioritizes detecting hate speech in Romanian, accounting for regional and cultural nuances to ensure accurate identification. Detection for other languages will be explored later.

REQ-HSD-006	Maintain Dynamic Hate Speech Database	Medium	C	The system maintains an up-to-date database of hate terms, slurs, and evolving coded language, reflecting the language used by various groups or movements.
REQ-HSD-007	Analyse User Behaviour for Hate Speech Patterns	Medium	C	The system monitors and analyses user behaviour to identify accounts that consistently engage in hate speech or target specific individuals or groups.
REQ-HSD-008	Perform Sentiment Analysis for Hate Speech Detection	High	M	The system evaluates the tone, sentiment, and intent of posts or comments to determine whether they constitute hate speech.

In the following *Table 4* the functional requirements for hate speech analysis services are presented.

Table 4: Hate speech analysis services: Functional Requirements

ID	Name	Priority	Feasibility	Description
REQ-HSA-001	Flag and Report Hate Speech	Medium	C	The system automatically flags hate speech with clear explanations and allows users to report suspected hate speech for further review and analysis.
REQ-HSA-002	Classify Hate Speech by Type (Ethnic/Cultural/Religious, Incitement to Violence Instigation/Mockery/Banter)	Medium	C	The system categorizes hate speech based on type (e.g., racial, religious, gender-based) to offer deeper insights and enhance explainability of detection decisions.
REQ-HSA-003	Map Hate Speech Propagation Networks	Medium	C	The system identifies and maps networks of users, accounts, or pages that spread hate speech, including detecting potential coordinated campaigns.
REQ-HSA-004	Retrain Model for New Trends and Slang	N/A	W	The system retrains the model to adapt to new trends, patterns, and evolving slang, ensuring continuous accuracy in detecting hate speech.

The functional requirements extracted from user requirements for deepfake detection and analysis are detailed in the *Table 5*, *Table 6*. Same as the other cases, the requirement identifiers follow the convention REQ-DFD for Deepfake Detection and REQ-DFA for Deepfake Analysis.

Table 5: Deepfake detection services: Functional Requirements

ID	Name	Priority	Feasibility	Description
REQ-DFD-001	Detect and Classify (and Retrain AI Models (if needed) Deepfakes in Multimedia Formats	High	C	The system detects and classifies deepfakes across various multimedia formats (audio, video, and images) in near real time, ensuring accurate identification of manipulated or synthetic media. It also supports the retraining of AI models if necessary to improve detection capabilities, covering content from both the surface and dark web.
REQ-DFD-002	Verify Deepfake Media Sources and Analyse Deepfake Contextual Integrity	Medium	C	The system verifies the authenticity of media files suspected to be deepfakes by cross-referencing metadata and timestamps to detect inconsistencies or potential manipulation. The system evaluates whether deepfake media is presented in its original context or manipulated to mislead, such as repurposing old or unrelated content.
REQ-DFD-003	Score Deepfake Media Authenticity and Analyse Metadata for Deepfake Indicators	Medium	C	The system assigns an authenticity score to media content suspected of being deepfakes, providing detailed reasoning and highlighting signs of potential manipulation or editing. The system examines media metadata—such as timestamps, geotags, and file history—to detect anomalies that may indicate deepfake generation or manipulation.

REQ-DFD-004	Verify Multilingual Audio for Deepfakes	N/A	W	The system detects manipulated or synthesized voices across various languages and dialects, ensuring accurate deepfake detection in multilingual audio content.
REQ-DFD-005	Maintain Verified Media Repository for Deepfake Detection	Medium	C	The system builds and maintains a database of verified, original media files to support cross-checking and authentication of potentially manipulated or deepfake content.
REQ-DFD-006	Eliminate Deepfake Content Duplication	N/A	W	The system integrates deepfake content from various sources, including surface web and dark web, while ensuring there is no duplication of previously detected or verified deepfake content.

Table 6: Deepfake analysis services: Functional Requirements

ID	Name	Priority	Feasibility	Description
REQ-DFA-001	Deepfake Indicators Dashboard	High	C	The system provides a dashboard that highlights potential deepfake indicators, such as face or voice mismatches, unrealistic lighting or shadows, and sudden audio artifacts, to assist in identifying manipulated content.
REQ-DFA-002	Map Deepfake Source Networks	Medium	C	The system identifies and tracks networks or accounts responsible for creating or propagating deepfakes, including both networks and coordinated disinformation campaigns.

REQ-DFA-003	Speech-to-Text Algorithm	Medium	C	The platform must implement a speech-to-text algorithm to convert video content into text for further analysis.

The functional requirements derived from user requirements for secure evidence exchange are listed in the *Table 7*.

Table 7: Secure Evidence - Functional Requirements

ID	**Name**	**Priority**	**Feasibility**	**Description**
REQ-SE -001	Data Security & Integrity Enforcement	High	S	Ensure confidentiality, integrity, and authenticity of data using hashing, encryption (symmetric & asymmetric), and signing hashes for secure transmission
REQ-SE -002	Secure and Tamper-Proof Logging Mechanism	High	S	The system should hash data before sending and verify it on receipt, using signatures on the hash to ensure security without impacting performance.
REQ-SE -003	Cryptographic Key Management & Compliance	High	S	Manage cryptographic materials securely (no plaintext storage), enforce NIST-compliant algorithms, and prepare for post-quantum cryptography.
REQ-SE -004	Scalable and Standardized Evidence Exchange	Medium	C	Enable secure, efficient, and scalable digital evidence exchange between LEAs using selective data transfer and reusable transmission flows.
REQ-SE -005	Automated and Auditable Security Workflows	Medium	C	Automate security operations without user interaction, maintain auditability, and support independent audits to verify compliance.
REQ-SE -006	Time Synchronization for Forensic Accuracy	High	S	Use synchronized time servers considering time zones to ensure consistent timestamps across systems.

The AVALANCHE web platform and user interface represent the final core module of the AVALANCHE solution. The associated functional requirements are detailed in the *Table 8*.

Table 8: AVALANCHE UI & DSE platform functional requirements

ID	Name	Priority	Feasibility	Description
REQ-G-001	Login	High	M	This function checks the user's credentials to decide whether he is allowed to access.
REQ-G-002	Register	High	M	This function allows users to create an account afterwards to log in securely to the platform.
REQ-G-003	Logout	High	M	This function is usually the last action done when the platform is used. It disconnects the user from the application.
REQ-G-004	Role Based Access Control	High	M	This function defines user roles and implements corresponding access permissions.
REQ-G-005	User Identity and Access Management	High	M	The system shall provide a comprehensive user identity and access management function, allowing administrators to manage user accounts, enforce secure authentication and password policies, and configure role-, group-, and user-based access permissions directly through the user interface.
REQ-G-006	Data Crawling Setup	High	M	The platform must provide a user interface form to configure and initiate data crawling, including input parameters such as sources, frequency, keywords, and time range.
REQ-G-007	View and Manage Crawled Data	High	M	The platform must allow users to view, filter, and manage crawled data, supporting multiple data types including text, images, videos, and audio. Users should be able to interact with the content (e.g., preview media), apply filters, and perform actions like deleting etc.

REQ-G-008	Perform Crawler Data Analytics	High	M	The platform must allow users to request analytics on crawled data, including statistics.
REQ-G-009	Perform Sentiment Analysis and view analysis results	High	M	The platform should enable users to perform sentiment analysis on crawled text data and present the results through visual analytics, including graphs, pie charts, sentiment scores, and a breakdown of sentiment classifications (positive, negative, neutral).
REQ-G-010	Perform Disinformation Detection and view results	High	M	The platform should allow users to run disinformation detection on selected or newly added content via a UI form, with optional parameter adjustments. Results should be clearly presented (using tools such as xAI solutions), including confidence scores, affected elements, manipulation types, and visual or textual explanations.
REQ-G-011	Perform Hate speech detection and view results	High	M	The platform supports hate speech detection on selected or newly ingested content via a UI form, with optional parameter adjustments. Results are displayed clearly, including confidence scores, affected content, and explanatory visuals or text (using tools such as xAI solutions).
REQ-G-012	Perform Deepfake detection and view results	Medium	C	The platform enables deepfake detection on video content submitted through its UI. Detection results are presented in a clear format, including confidence scores, identified manipulations, affected frames, and visual or textual explanations (using tools such as xAI solutions).
REQ-G-013	Monitor Disinformation, Deepfake and hate speech detection request status	High	M	The platform must allow users to view the status of their disinformation, deepfake and hate speech detection requests (e.g., queued, processing, completed, error), including timestamps and request history.

REQ-G-014	Task Definition and Information Retrieval Access Control	High	S	The platform allows the analyst who created the task to define which users can edit or view information related to the crawling task. The analyst can modify user access or revoke access at any time. Logs will be maintained to track who accessed the task and when.
REQ-G-015	Export information within the platform	High	S	The system supports users in selecting specific columns or data to export via the UI, with options to export in CSV and JSON formats.
REQ-G-016	Enable User Reporting of Suspicious Media	Medium	C	The system allows users to report suspicious media for further analysis and verification by the platform, aiding in the identification and validation of potential deepfakes.
REQ-G-017	Multilingual Text Translation	Medium	C	The platform must support multilingual text translation (e.g., English to Romanian, Romanian to Arabic, etc.) for various language pairs.
REQ-G-018	Speech-to-Text Algorithm	Medium	C	The platform must implement a speech-to-text algorithm to convert audio or video content into text for further analysis.

2.2 Non-functional requirements

Non-functional requirements (NFRs) define the qualities and characteristics that govern how the system should perform, rather than specifying its functional behaviour. These requirements are essential for ensuring that the AVALANCHE solution meets user expectations in terms of operational performance and delivers satisfactory overall user experience. To support a robust and traceable development process, these NFRs are based on the Quality Goals thoughtfully outlined in [1]. This deliberate strategy is crucial for ensuring continuity and establishing a clear logical link. Beyond high-level qualities such as performance, security, and scalability, the system's technical specifications will outline the specific technologies and configurations required to meet these requirements. Table 9 presents the most critical NFRs for the AVALANCHE platform.

Table 9: Non-functional Requirements

ID	Name	Priority	Feasibility	Description
NFR-001	Performance (High Availability & Redundancy)	High	M	Ensure high availability with minimal downtime. Implement redundancy through load balancing to guarantee business continuity.
NFR-002	Usability, UI Design, and Navigation	High	M	The platform should have an integrated, intuitive UI with a consistent design, allowing easy navigation through hierarchical menus and minimizing clicks to complete tasks.
NFR-003	Deployment Support & Guidelines	High	M	Provide clear deployment guidelines and support for on-premises installations, including licenses for all required software prerequisites.
NFR-004	Hardware and Platform Compatibility & Optimization	High	M	The platform must seamlessly integrate with hardware provided by LEAs during on-premises deployment. It should ensure consistent performance across environments, browsers, and client configurations, without requiring third-party apps or client-side setup. Optimization strategies must be implemented for high efficiency and scalability.
NFR-005	Security, APIs, and Data Protection	High	M	The platform must ensure secure communication, encryption, and access control, complying with the latest security standards. It should support strong password policies, two-factor authentication, and audit logging, while ensuring data privacy, integrity, and secure backup and recovery.

NFR-006	Resilient and Independent Modules	High	M	Each feature/module should operate independently, ensuring that the platform remains functional even if one module fails.
NFR-007	Scalability and Load Handling	High	M	The platform must be scalable and capable of handling increasing loads without requiring additional hardware or software upgrades. Performance must remain consistent under varying loads, and the system must be able to manage increased data volumes efficiently.

Regarding the associated technical specifications, these will outline the specific technologies, infrastructure elements, and system configurations necessary to meet the non-functional requirements, ensuring that the system performs reliably under real-world conditions. The following sections provide a detailed description of the core components of the AVALANCHE platform, along with the corresponding technology stacks and implementation details.

In conclusion to this section on the overall system requirements, it is essential to emphasize that the system requirements were systematically derived from multiple sources. These include user requirements identified through the SCORE methodology [2], as well as direct inputs from the SPP (end-users). Notably, the functional requirements concerning the crawler services, the AVALANCHE web platform, and the user interface were primarily informed by feedback from the SPP, with further details available in Appendix A. Moreover, the technical requirements submitted by the SPP; also documented in Appendix A, served as a valuable input for defining both the non-functional requirements and the initial set of technical specifications. These elements will play a key role in shaping and guiding the forthcoming development tasks within WP5.

3. AVALANCHE solution overview & architecture design

This section provides a high-level overview of the AVALANCHE solution and its underlying architecture. It describes the core system components, their interactions using sequence diagrams, and the overall structural design that supports scalability, modularity, and interoperability. The architecture is built on a microservices-based approach, enabling flexible deployment and independent service updates. Key design principles such as service orchestration, containerization, and API-driven communication are discussed, along with the roles of major actors and data flows within the system. This overview serves as the core for understanding how AVALANCHE meets its functional and non-functional requirements through versatile architecture.

3.1 High-Level System Description

The initial version of the AVALANCHE platform architecture was completed as of M09, following the GA. However, given the iterative nature of the implementation phase, the system architecture may be modified and optimized throughout the development process. Figure 1 depicts the high-level final version of the AVALANCHE platform architecture. It provides an overview of the system's main components and illustrates the main data flow between the AVALANCHE modules.

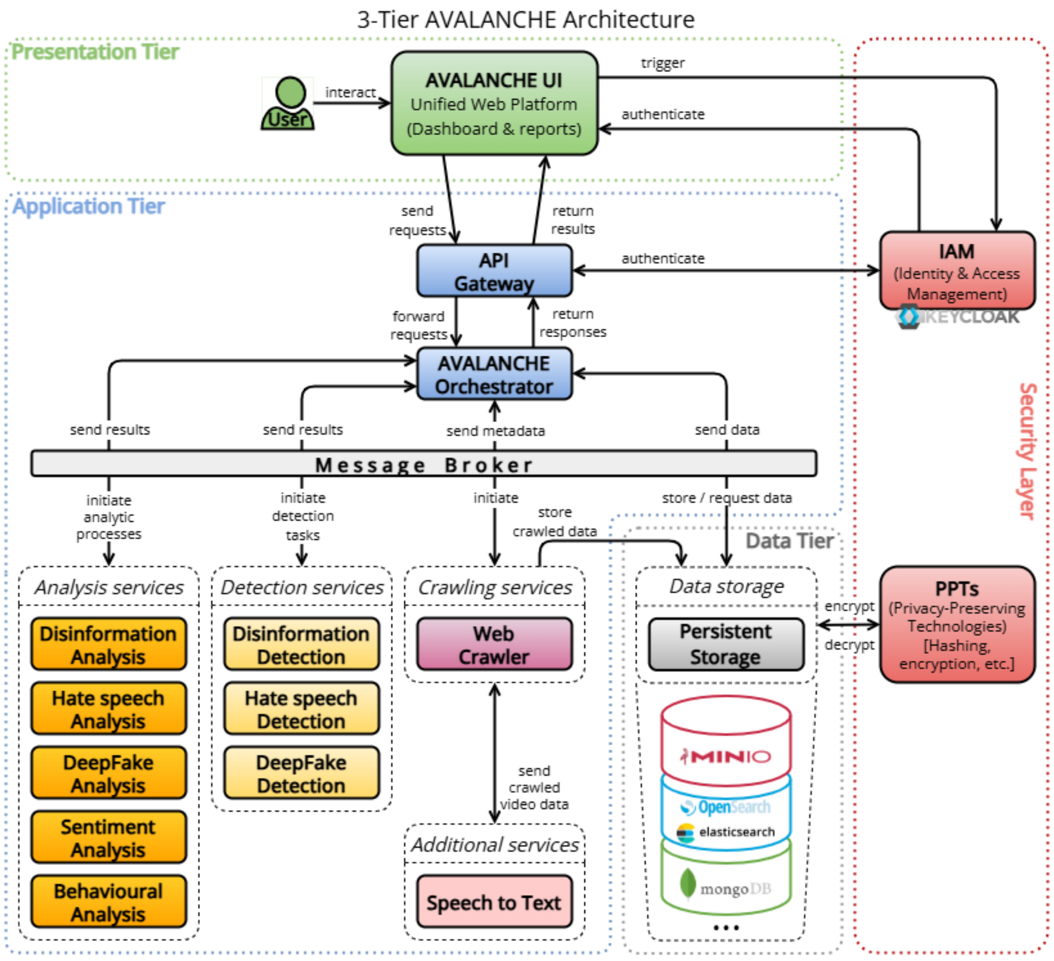


Figure 1:AVALANCHE High level architecture

Before delving into the detailed architecture design, it is important to first identify the intended users of the platform. Specifically, the AVALANCHE platform is designed for use by Law Enforcement Agencies (LEAs), enabling them to analyse, detect, and respond to emerging digital threats. While SPP is the primary end-user of the AVALANCHE platform, the design aims to address the operational needs of a broad range of Law Enforcement Agencies (LEAs). A detailed overview of the AVALANCHE actors, outlining the various roles within LEA organizations and across collaborating agencies, is provided in Section 3.2.

AVALANCHE will adopt a 3-tier architecture, which is essential for promoting separation of concerns, ultimately making the system more scalable, maintainable, and flexible. By organizing the application into distinct layers; presentation, application, security and data; each layer is responsible for specific functions. This structure simplifies updates and improvements, reduces complexity, and enhances the system's ability to adapt to changes or growth in an organized manner. More specifically,

1. Presentation Layer: This is where users interact with the system via the AVALANCHE Web platform - UI. The AVALANCHE web platform will offer a user-friendly dashboard that brings together all core components of AVALANCHE. The platform will include interactive visuals and decision support features to help users as well as allowing them for personalized analysis and custom reporting. Furthermore, the requests made by users are routed through the API Gateway, which authenticates the user via the Identification management (IAM) system and directs the requests to the appropriate service in the application layer.
2. Application Layer: This layer consists of the following primary modules.
    a. The Orchestrator who is responsible for coordinating and managing the overall flow of tasks across multiple services. It ensures that complex user requests, such as those requiring data from more than one service, are processed correctly. For example, a user's request for results could involve data from detection services, analysis layers, and crawling services, which need to be combined or processed in a specific sequence. The Orchestrator manages this sequence and ensures all involved services work in harmony to fulfil the request.
    b. API Gateway: This is also a key component in AVALANCHE system design which will serve as a centralized entry point for managing and routing requests from clients to the appropriate backend services. There are many technologies that could be used during the development process, with a strong preference for Spring WebFlux, which will integrate seamlessly with the orchestrator development based on Spring Boot.
        i. Message broker: To be able to publish and consume events a Message broker is needed. So as, the Message broker acting as the communication hub in the data layer, ensuring messages or events are delivered between services asynchronously. For this module's development, an in-house Data Fusion Bus (DFB) solution which is provided ITML can be used, leveraging robust, production-grade, and fully scalable open-source technologies such as Apache Kafka, OpenSearch, Grafana, and Keycloak.

It is important to note that the Message Broker facilitates asynchronous communication between services, effectively decoupling them and allowing independent operation. It ensures reliable data flow, message transmission, and event handling without direct service communication. In contrast, the Orchestrator manages complex workflows, coordinating the execution of services in a specific order to fulfil user requests. It ensures that services like detection, crawling, and analysis work together in sequence to produce the desired results.

c. Detection Services: These services perform detection tasks based on user requests, such as:
    i. Disinformation Detection: Identifying false or misleading information.
    ii. Hate Speech Detection: Identifying harmful language or hate speech.
    iii. Deepfake Detection: Identifying manipulated or synthetic media.

These detection services are invoked based on the user's needs and help to identify relevant content for further analysis or actions.

d. Analysis Services: These services process and analyse data based on the user's preferences, including:
    i. Disinformation Analysis: Analysing detected disinformation for patterns, sources, or impact.
    ii. Hate Speech Analysis: Analysing hate speech to understand context, sentiment, and potential harm.
    iii. Deepfake Analysis: Examining deepfake content in greater detail to identify its origins or implications.
    iv. Behaviour and Sentiment Analysis: Analysing the behaviour and sentiment behind data (e.g., text, social media posts, images) to derive insights.

These analysis services take the output from the detection services and perform deeper analysis or generate reports.

e. Crawling Services: This service crawls the web or other designated sources to retrieve data based on user-defined preferences. It collects information that may be relevant for detection or analysis, operating independently of the orchestrator. Once crawled and collected, the data is stored in the Data Layer for further processing or analysis.
f. Additional Services: This category includes any extra services that enhance the overall solution, such as the speech-to-text module. This module converts audio from videos into text, which serves as the primary input for many services, including disinformation detection, hate speech analysis, sentiment analysis, and others.

3. Data Layer: The data layer stores and retrieves information from databases or other data storage systems, providing necessary data to the application layer. Thus, in the data layer there is the Persistence Storage block which is responsible for storing data persistently in databases, ensuring data is available for future use or retrieval. The databases that may potentially be used include relational databases (e.g. PostgreSQL), non-relational

databases (e.g. MongoDB, Elasticsearch), object storage (e.g. MinIO). The CKAN[1] data platform is selected for data storage, due to its interoperability, support for multiple database types, user-friendly interface, extensibility, and robustness. It will support uploading data through a user interface or API, as well as viewing data for file types such as images and CSVs.

In the proposed AVALANCHE architecture, an additional Security Layer is introduced to address end-user requirements for secure communication and reliable exchange of evidence. This layer is designed to ensure robust protection and privacy, comprising two main sub-modules:

- Identity and Access Management (IAM): Manages user authentication, authorization, and access control, ensuring that only authorized users can access sensitive data and services. The primary technology that will be used is Keycloak.
- Privacy Protection and Security Technologies: Implements advanced privacy-preserving mechanisms, including encryption, hashing, anonymization, and other techniques to protect data integrity and confidentiality throughout communication and storage.

3.2 AVALANCHE Actors

As mentioned in the previous section, the primary users of the AVALANCHE platform are Law Enforcement Agencies (LEAs), with SPP being the focus within this project. However, the defined user roles are not limited to SPP's needs, they are designed to address the broader requirements of LEAs in general. To support this, the platform outlines specific roles within a LEA organization to ensure tailored access to different parts of the AVALANCHE solution. In addition to these internal roles, the platform also considers other actors, including internal system actors and external users such as third-party LEAs. The main actors of the AVALANCHE platform are outlined below (*Table 10*):

Table 10: AVALANCHE Actors

Type	Name	Role	Description	Key responsibilities
Internal actor	Platform Administrator	System configuration and management	Manages system integrity, user roles, and access control.	• Configures user roles and permissions (IAM setup). • Monitors system performance and resolves technical issues. • Manages data storage, backup, and disaster recovery.
Internal actor	AI/ML Engineer	Model Development and Maintenance	Designs, trains, and maintains AI models for data analysis.	• Develops, deploys, and fine-tunes machine learning models (disinformation detection,

[1] https://ckan.org/

				deepfake detection, sentiment analysis). • Monitors model performance and ensures accuracy. • Updates models with new data and improvements.
Internal actor	Security Engineer	System Monitoring and Incident Response	Monitors system for suspicious activities and investigates incidents.	• Reviews user access logs for anomalies. • Investigates and responds to security incidents. • Ensures system compliance with security policies.
Authorized Internal Users – SPP Users	LEAs Officer – Decision Maker	Strategic Oversight & Targeted Analysis	High-level decision-maker responsible for defining investigation objectives, reviewing analysis outcomes, and coordinating strategic actions based on system insights.	• Define investigation priorities (disinformation, hate speech, deepfakes). • Request general and case-specific data analysis. • Review outputs to derive actionable insights. • Lead inter-agency collaboration for cross-analysis. • Make high-level decisions based on risk and intelligence reports.
Authorized Internal Users – LEAs Users	LEAs Operator – OSINT	System Use, Data Collection & Crawling	Technical user who configures and initiates data crawling, manages system workflows, and ensures effective data acquisition for analysis tasks.	• Initiate and manage data collection tasks (e.g., URLs, keywords). • Conduct OSINT crawling for relevant content (social media, websites). • Ensure proper configuration of detection workflows (hate speech, disinformation, etc.). • Coordinate with analysts to prioritize targets and sources. • Monitor data pipeline and system resource use.
Authorized Internal Users – LEAs Users	LEAs Analyst (Intelligence Analyst)	Risk & Intelligence Assessment	Expert responsible for interpreting detection results, assessing risks,	• Analyse output from detection modules and crawled data. • Perform case-specific risk assessments.

			and producing analytical reports to support operational decision-making.	• Validate and contextualize insights for operational use. • Prepare analytical summaries and intelligence briefings. • Support the decision-making process through structured reporting.
External Actors (3rd Party LEAs - Law Enforcement Agencies)	LEA User (External Law Enforcement Access)	Access to Pre-Produced Analysis Results	Has limited access to review already produced analysis of results relevant to ongoing investigations or public safety matters. They cannot initiate new searches or data collection tasks.	• Accesses existing analysis results (e.g., disinformation, deepfake detection) for specific cases or general security insights. • Reviews relevant insights without initiating any new data collection or search tasks. • Requests further clarification or additional details on the analysis of results (through SPP, if authorized).

At this stage, it is worth noting that the identified AVALANCHE actors were influenced by input from the SPP, provided during the user requirements collection process, as outlined in Appendix B, where the use cases and other key elements are described. Furthermore, these use cases also shape the user journey design of the AVALANCHE platform, as discussed in Section 3.2.2.

3.2.1 Sequence Diagrams

Sequence diagrams played a key role in the design of the AVALANCHE platform, providing a structured visualization of the interaction flow between system components over time and illustrating how these components communicate. Based on the defined use cases, four sequence diagrams were developed, each corresponding to one of the platform's core modules: (i) Crawling Services (ii) Disinformation Detection and Analysis Services (iii) Hate Speech Detection and Analysis Services (iv) Deepfake Detection and Analysis Services.

It is also important to note that additional services described in the architecture walkthrough such as Sentiment Analysis and other Analytics Services are integrated within these four diagrams.

Figure 2 provides an overview of the crawling service, highlighting the crawler component and its interactions with the other modules within the AVALANCHE platform.

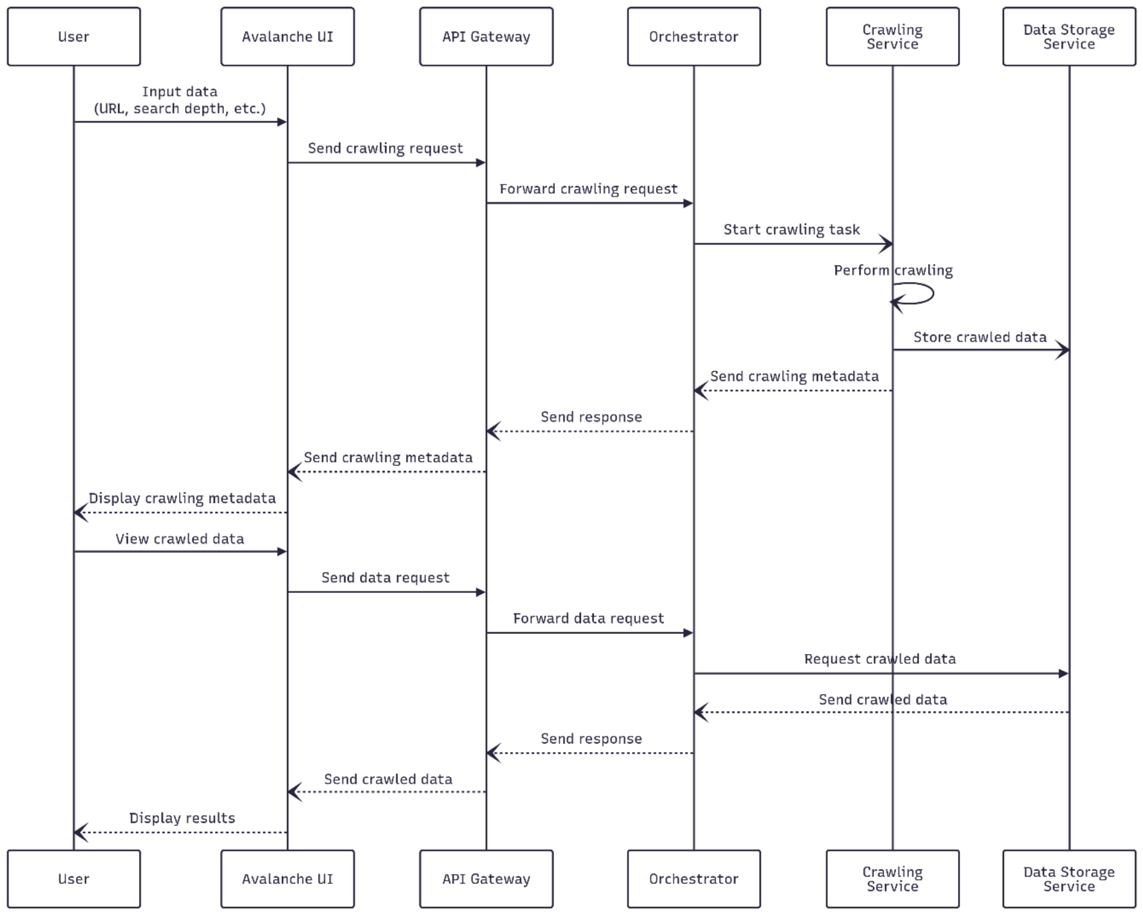


Figure 2: Crawling Services Crawling Services operation/data flow

Crawling Service Interaction Flow

1. User Interaction Initiation:
    - The interaction begins when the User provides input through the AVALANCHE UI. This input typically includes details such as the target URL, desired depth search, search depth, and potentially other configuration parameters.
2. Request Dispatch:
    - The Avalanche UI sends a crawling request to the API Gateway.
    - The API Gateway, acting as a conduit, forwards this request to the Orchestrator for further processing.
3. Crawling Task Execution:
    - The Orchestrator processes the request and initiates a crawling task by sending instructions to the Crawling Service.
    - The Crawling Service then executes the operation, fetching the necessary data from the specified sources.
4. Data Storage and Metadata Response:
    - Upon completion, the Crawling Service stores the collected data in the Data Storage Service.

- It then sends metadata (such as crawl status, duration, or number of items crawled) back to the Orchestrator.
- The Orchestrator relays this metadata to the API Gateway, which then returns it to the AVALANCHE UI for display to the User.

5. User Requests Crawled Data: After reviewing the metadata, the User may choose to view the crawled data.
    - The Avalanche UI sends a data retrieval request to the API Gateway, which forwards it to the Orchestrator.
6. Data Retrieval and Display:
    - The Orchestrator requests the relevant data from the Data Storage Service.
    - This data is sent back to the Orchestrator and finally delivered to the AVALANCHE UI via the API Gateway.
    - The Avalanche UI then displays the results to the User.

Continuing with the disinformation detection and analysis service, Figure 3 illustrates the crawler component and its integration with the other modules of the AVALANCHE platform.

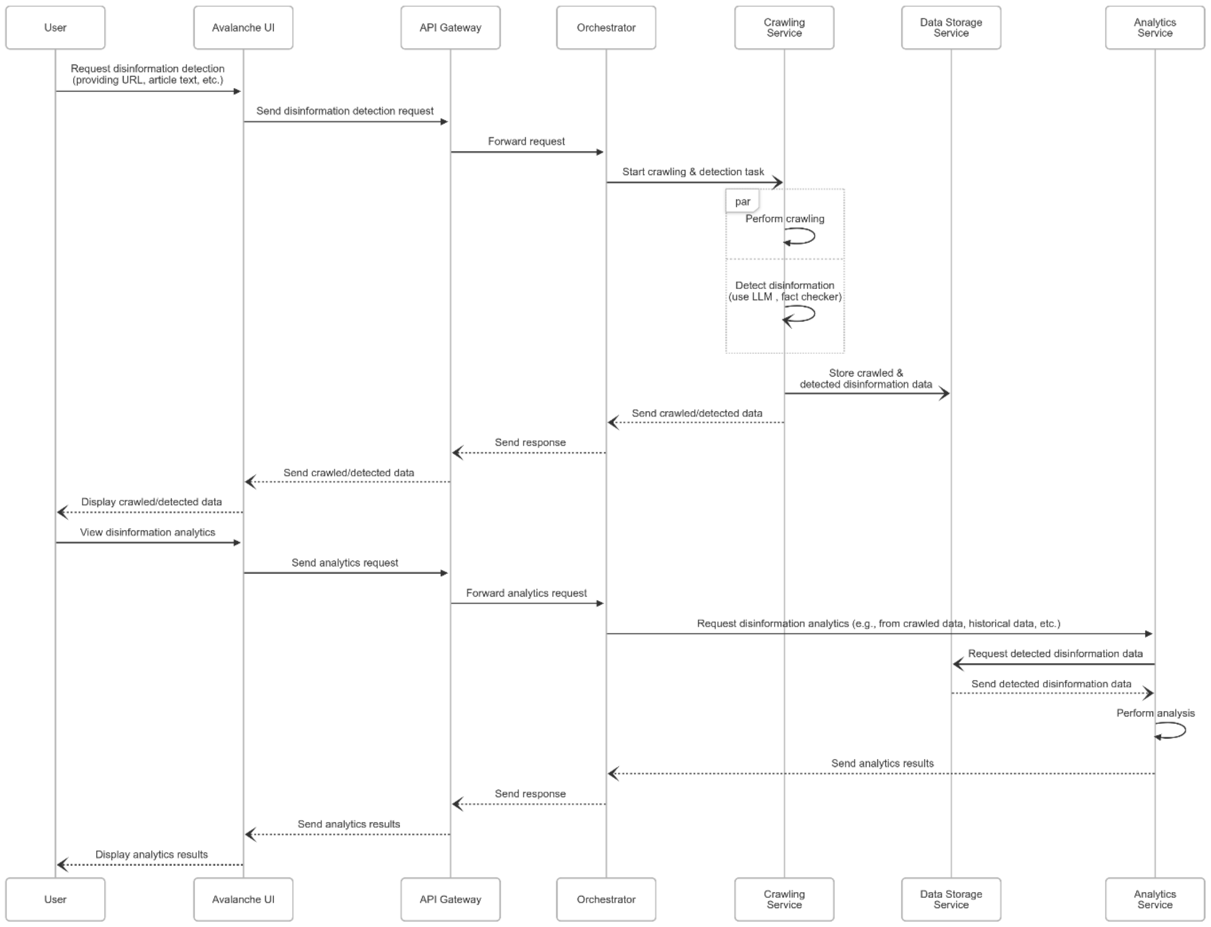

User
Avalanche UI
API Gateway
Orchestrator
Crawling Service
Data Storage Service
Analytics Service
Request disinformation detection (providing URL, article text, etc.)
Send disinformation detection request
Forward request
Start crawling & detection task
par
Perform crawling
Detect disinformation (use LLM , fact checker)
Store crawled & detected disinformation data
Send crawled/detected data
Send response
Send crawled/detected data
Display crawled/detected data
View disinformation analytics
Send analytics request
Forward analytics request
Request disinformation analytics (e.g., from crawled data, historical data, etc.)
Request detected disinformation data
Send detected disinformation data
Perform analysis
Send analytics results
Send response
Send analytics results
Display analytics results

Figure 3: Disinformation Detection and Analysis Service operation/data flow

Disinformation Detection and Analysis Service Interaction Flow

1. User Request Submission:
   The process begins when the User submits a request via the AVALANCHE UI to detect disinformation. This request typically includes a URL, article text, or similar content to be evaluated.
2. Request Routing:
   - The AVALANCHE UI sends the disinformation detection request to the API Gateway.
   - The API Gateway then forwards the request to the Orchestrator for processing and task coordination.
3. Crawling & Detection Task Execution:
   - The Orchestrator initiates the process by starting a crawling and detection task via the Crawling Service.
   - Within the Crawling Service, two operations are performed in parallel (par block):
     - Crawling: Data is fetched from the specified sources.
     - Disinformation Detection: The content is analysed using a Large Language Model (LLM) or a fact-checking component to identify disinformation.
   - Once both steps are completed, the crawled and detected disinformation data is stored in the Data Storage Service.
4. Response to User:
   - The Crawling Service sends the crawled and detected data back to the Orchestrator.
   - The Orchestrator returns this response via the API Gateway, and the AVALANCHE UI displays the data to the User.
5. Requesting Analytics:
6. After reviewing the detection results, the User may want to view disinformation analytics
   - The AVALANCHE UI sends analytics request to the API Gateway, which is then forwarded to the Orchestrator.
7. Analytics Computation
   - The Orchestrator sends a request to the Crawling Service to retrieve analytics-related information, such as data from previous crawls or disinformation trends.
   - The Crawling Service requests detected disinformation data from the Data Storage Service, which sends the data back.
   - Using this data, the Analytics Service performs the required analysis.
8. Returning Analytics Results:
   - The Analytics Service sends the analysis results back through the Crawling Service and Orchestrator to the API Gateway.
   - The AVALANCHE UI receives the results and displays the analytics to the User.

With respect to the hate speech detection and analysis service, Figure 4 presents a sequence diagram illustrating the interactions between this service and the other components of the AVALANCHE platform.

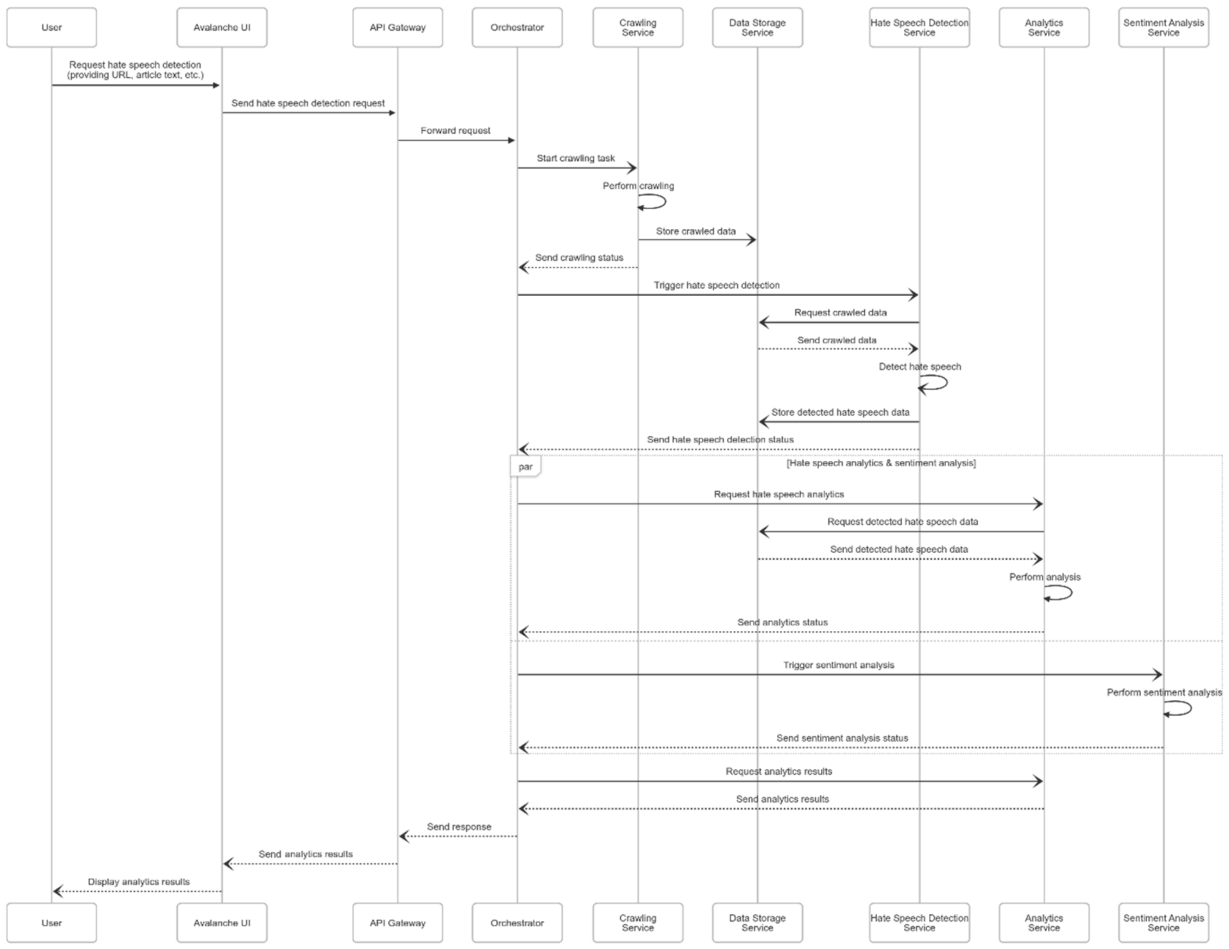

User
Avalanche UI
API Gateway
Orchestrator
Crawling Service
Data Storage Service
Hate Speech Detection Service
Analytics Service
Sentiment Analysis Service
Request hate speech detection (providing URL, article text, etc.)
Send hate speech detection request
Forward request
Start crawling task
Perform crawling
Store crawled data
Send crawling status
Trigger hate speech detection
Request crawled data
Send crawled data
Detect hate speech
Store detected hate speech data
Send hate speech detection status
par
[Hate speech analytics & sentiment analysis]
Request hate speech analytics
Request detected hate speech data
Send detected hate speech data
Perform analysis
Send analytics status
Trigger sentiment analysis
Perform sentiment analysis
Send sentiment analysis status
Request analytics results
Send analytics results
Send response
Send analytics results
Display analytics results

Figure 4: Hate speech Detection and Analysis Service operation/data flow

Hate Speech Detection and Analysis Service Interaction Flow

1. User Interaction Initiation:
2. The process begins when the User requests hate speech detection through the AVALANCHE [UI, providing input such as a URL or plain text.
3. Detection Request Routing:
    - The AVALANCHE UI sends the hate speech detection request to the API Gateway.
    - The API Gateway forwards the request to the Orchestrator, which coordinates the execution of the detection pipeline.
4. Crawling Phase:
    - The Orchestrator instructs the Crawling Service to begin the task.
    - The Crawling Service performs crawling to collect relevant content and stores the crawled data in the Data Storage Service.
    - A crawling status message is sent back to the Orchestrator.
5. Hate Speech Detection:
    - Upon receiving confirmation that crawling is complete, the Orchestrator triggers the Hate Speech Detection Service.

   - This service requests crawled data from the Data Storage Service, performs hate speech detection, and then stores the detected hate speech data.
   - A detection status update is returned to the Orchestrator.
6. Parallel Analytics Tasks:
7. Once hate speech detection is complete, the Orchestrator concurrently initiates two analytics processes (if requested by the user):

   a. Hate Speech Analytics:
      - The Orchestrator sends a hate speech analytics request to the Analytics Service.
      - The Analytics Service retrieves the detected hate speech data from the Data Storage Service and performs analysis (e.g., frequency, source categorization, intensity, etc.).
      - A status update is sent back once analytics are complete.

   b. Sentiment Analysis:
      - Simultaneously, the Orchestrator triggers a sentiment analysis task via the Sentiment Analysis Service.
      - This service performs sentiment analysis on the same or related data and sends back a status update upon completion.

8. Analytics Results Compilation:
   - Once both analytics processes are complete, the Orchestrator requests the final analytics results.
   - These results are returned by the Analytics Service, passed through the Orchestrator, and sent via the API Gateway back to the AVALANCHE UI.
9. Results Display to User: Finally, the AVALANCHE UI displays the analytics results to the User, including insights from both hate speech and sentiment analyses.

The final sequence diagram, presented in Figure 5, describes the interaction between the deepfake detection and analysis services and the AVALANCHE platform.

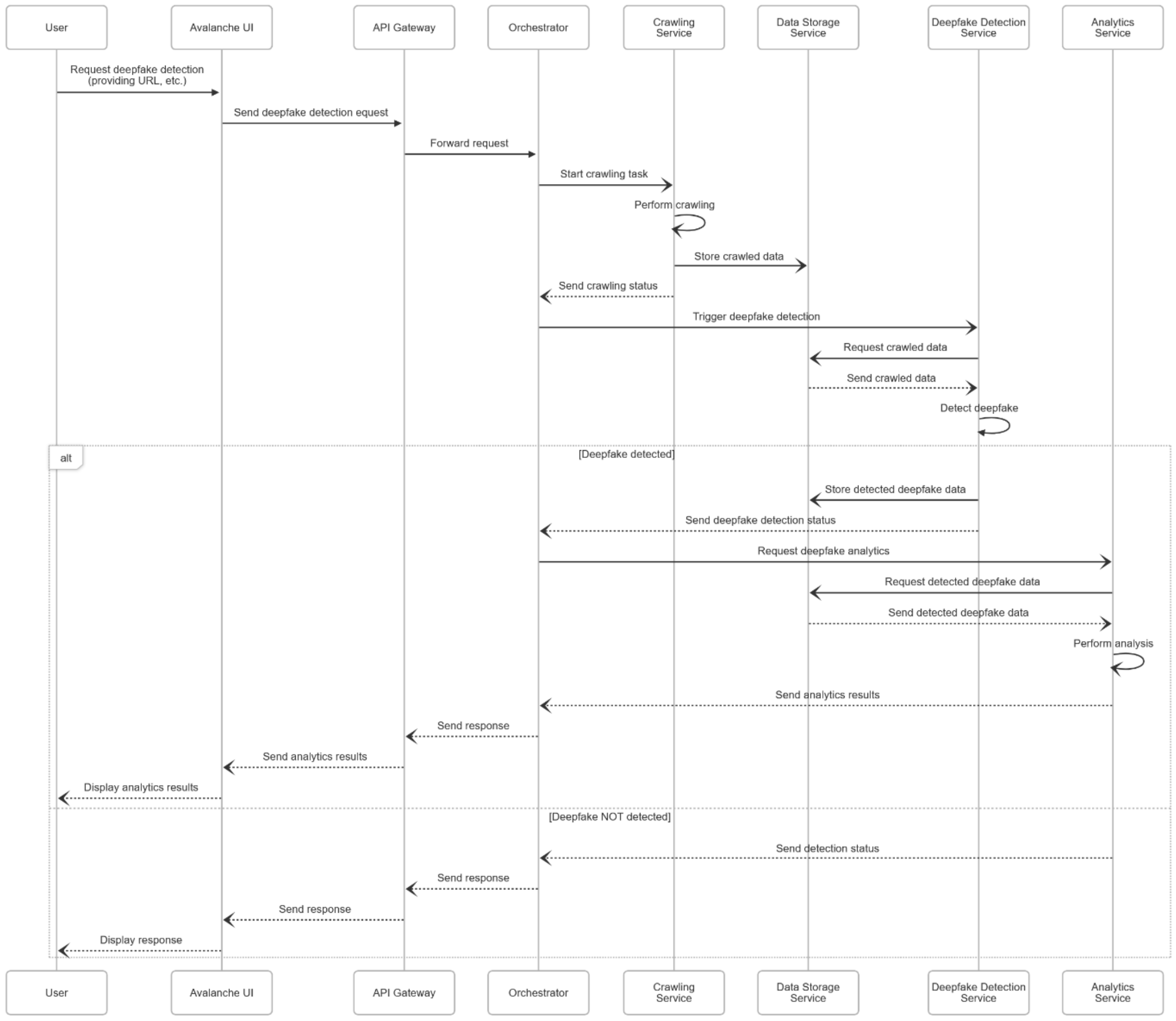


Figure 5: DeepFake Detection and Analysis Service operation/data flow

DeepFake Detection and Analysis Service Interaction Flow

The sequence diagram illustrates the end-to-end flow of the Deepfake Detection and Analysis process within the AVALANCHE platform, detailing the interactions between the components involved.

1. User Request Initiation:
2. The process begins when the User initiates a deepfake detection request through the AVALANCHE UI, providing a URL or relevant media source.
3. Request Propagation:
4. The Avalanche UI sends this request to the API Gateway, which forwards it to the Orchestrator for further processing.
5. Crawling Task Execution:
6. The Orchestrator triggers the Crawling Service to perform the crawling task. Once crawling is completed, the gathered media data is stored in the Data Storage Service, and a crawling status update is sent back to the Orchestrator.

7. Trigger Deepfake Detection:
8. After successful crawling, the Orchestrator invokes the Deepfake Detection Service. This service requests the necessary data from the Data Storage Service, performs deepfake analysis, and determines whether deepfake content is present.
9. Conditional Handling (alternative):
10. The sequence includes an alternative flow (alternative; alt) block that handles two outcomes:
    - If Deepfake is Detected:
        - The detection results are stored in the Data Storage Service.
        - A status update is sent back to the Orchestrator.
        - The Orchestrator triggers the Analytics Service to perform deeper analysis on the detected deepfake content.
        - The Analytics Service requests the relevant data from the Data Storage Service, performs the analysis, and sends the results back to the Orchestrator.
        - The results are then sent through the API Gateway to AVALANCHE UI, which presents them to the User.
    - If No Deepfake is Detected:
        - A corresponding status update is sent directly back from the Detection Service to the Orchestrator, which forwards it through the API Gateway to the Avalanche UI, informing the User that no deepfake content was found.

3.2.2 User journey scenarios

This section provides a high-level overview of the main user flows and journeys within the AVALANCHE solution, outlining the typical steps and interactions users follow across key scenarios. The flows are presented in a text-based format alongside supporting wireframes, offering a clear and intuitive depiction of the user experience. This section provides a high-level overview of the main user journey scenarios within the AVALANCHE solution, highlighting how the users interact with the platform in various operational scenarios. These user flows represent the main steps, actions that users, such as investigators, analysts, and administrators, follow when performing key tasks like disinformation detection, analysis, reporting etc.

Each flow is described in a text-based format to ensure clarity, supported by wireframes that visually represent the interface at each stage. The goal is to present not only the functionality available, but also how it aligns with operational workflows, ensuring the AVALANCHE platform effectively supports its users (LEAs).

Registration of a new account and login

Login page: User selects "Register" (Figure 6)

- **Registration form:** User inputs credentials
- **Admin moderation** (workflow optional; Figure 8,Figure 9)
    - Admin accesses a moderation panel.
    - Admin reviews the user request, verifies organizational details, and approves/rejects the new account
- **Notification:** Upon approval, the user receives an email confirmation with a one-time link and can log in.

- **Login Page:** User logs in and is directed to the “AVALANCHE Dashboard**”** (Figure 7)

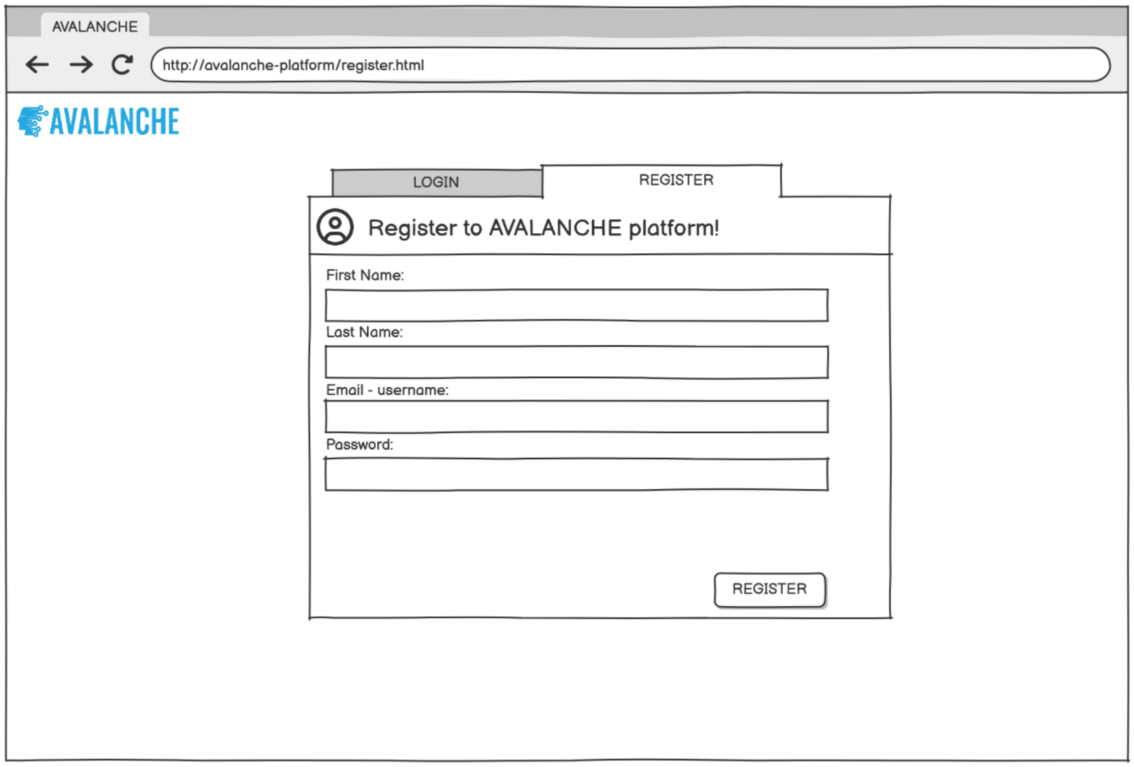


Figure 6: AVALANCHE Register Screen

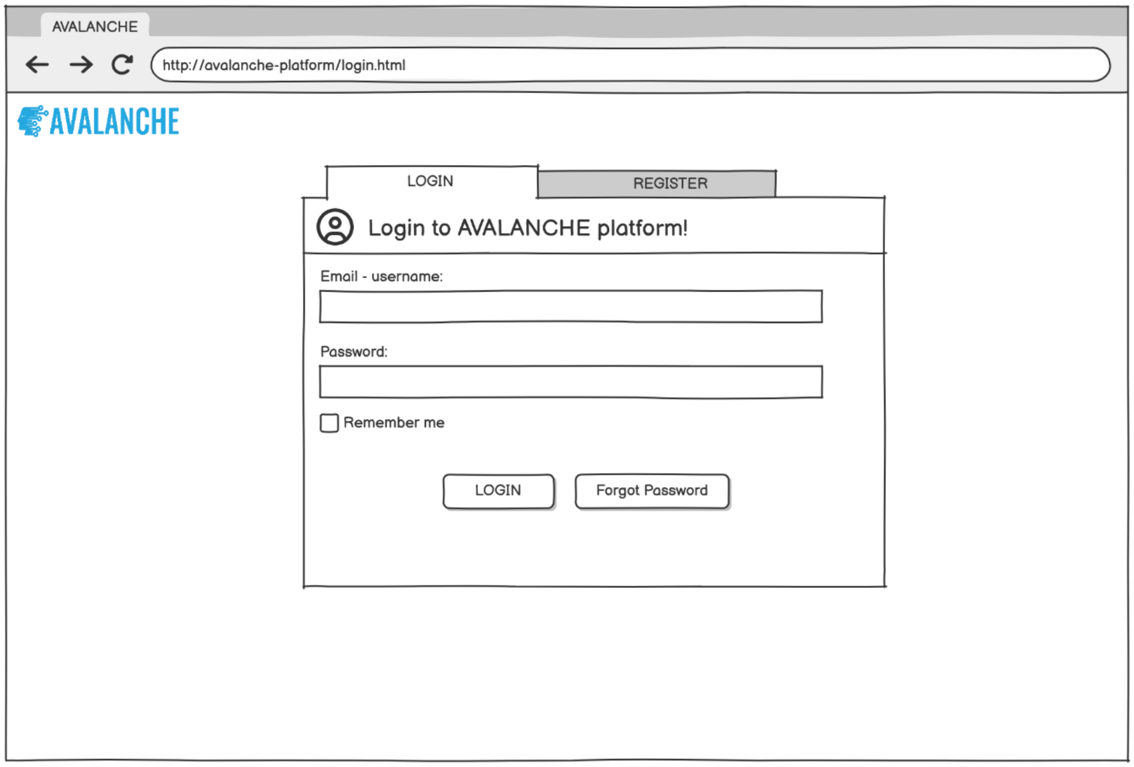
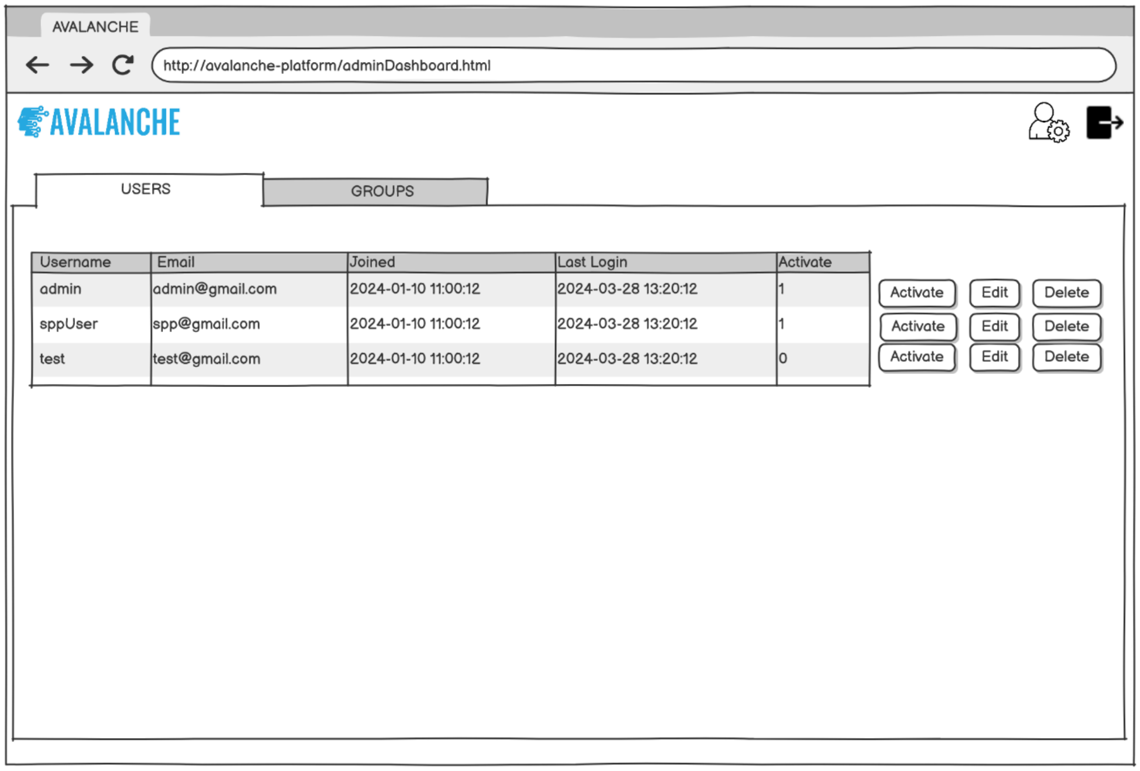
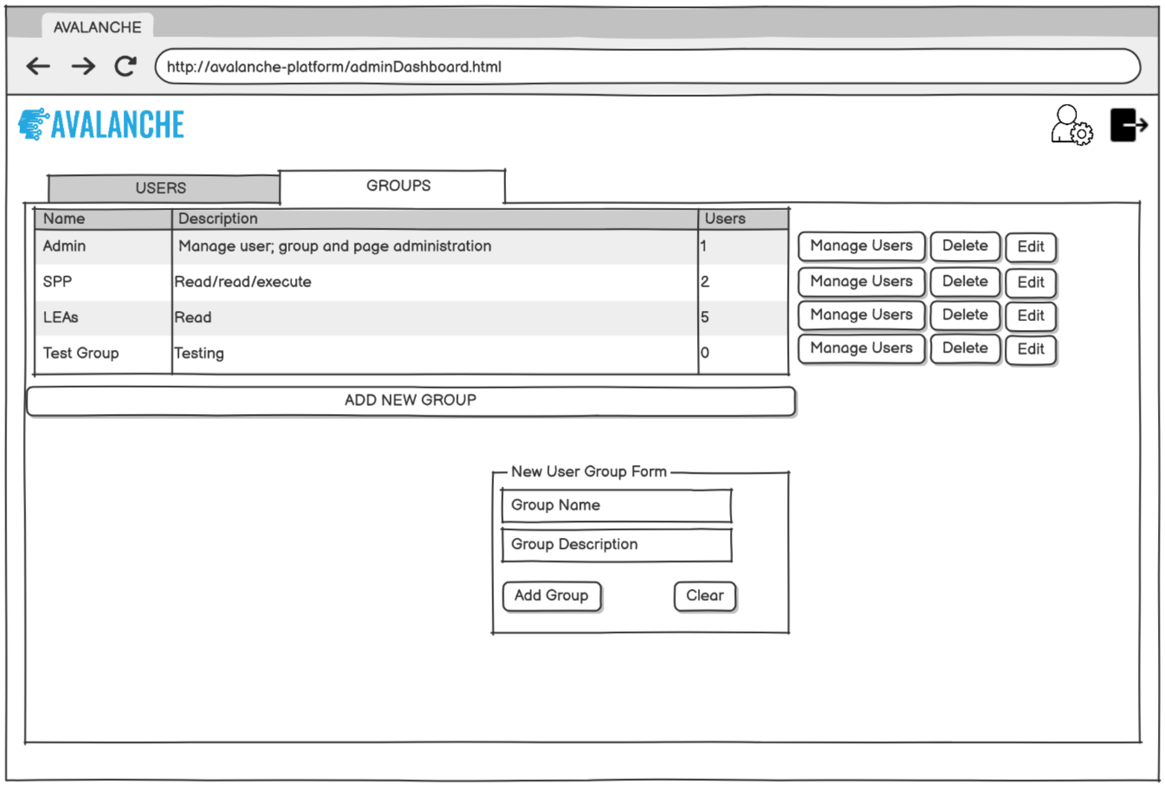


Figure 7: AVALANCHE Login Screen



Username	Email	Joined	Last Login	Activate
admin	admin@gmail.com	2024-01-10 11:00:12	2024-03-28 13:20:12	1
sppUser	spp@gmail.com	2024-01-10 11:00:12	2024-03-28 13:20:12	1
test	test@gmail.com	2024-01-10 11:00:12	2024-03-28 13:20:12	0

Figure 8: AVALANHCE Administrator Dashboard screen

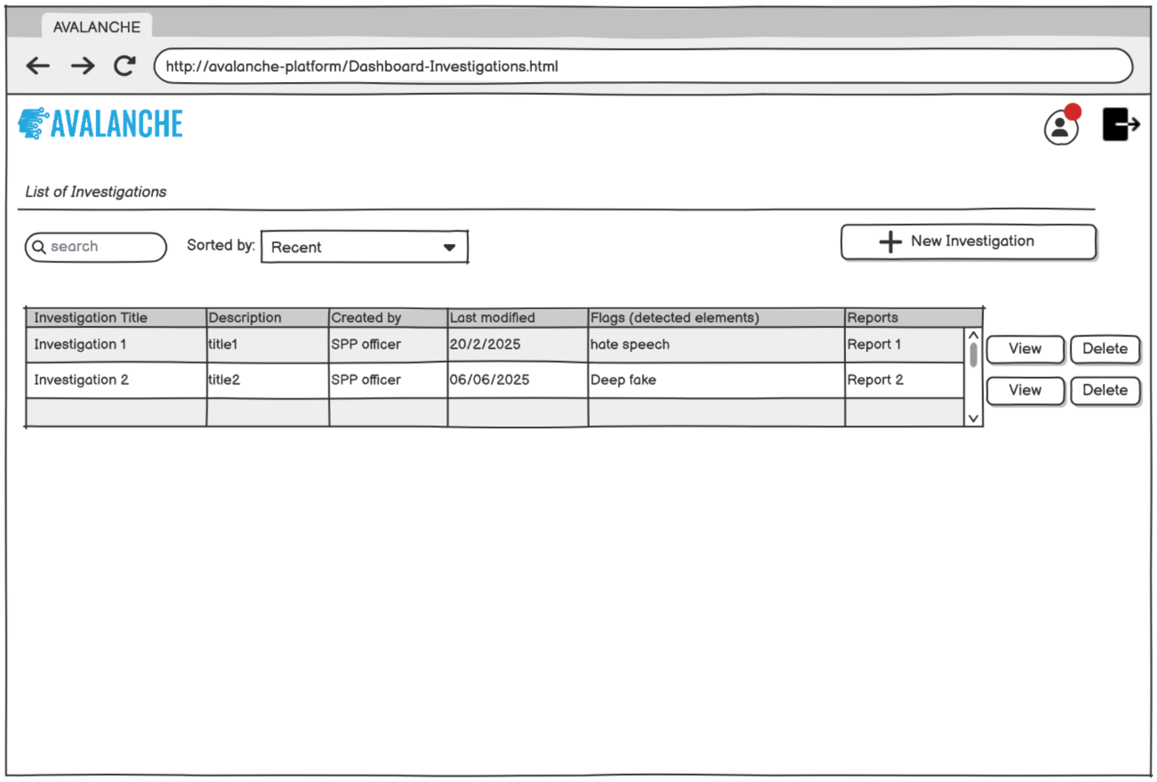


Name	Description	Users
Admin	Manage user; group and page administration	1
SPP	Read/read/execute	2
LEAs	Read	5
Test Group	Testing	0

Figure 9: AVALANHCE Administrator Dashboard screen

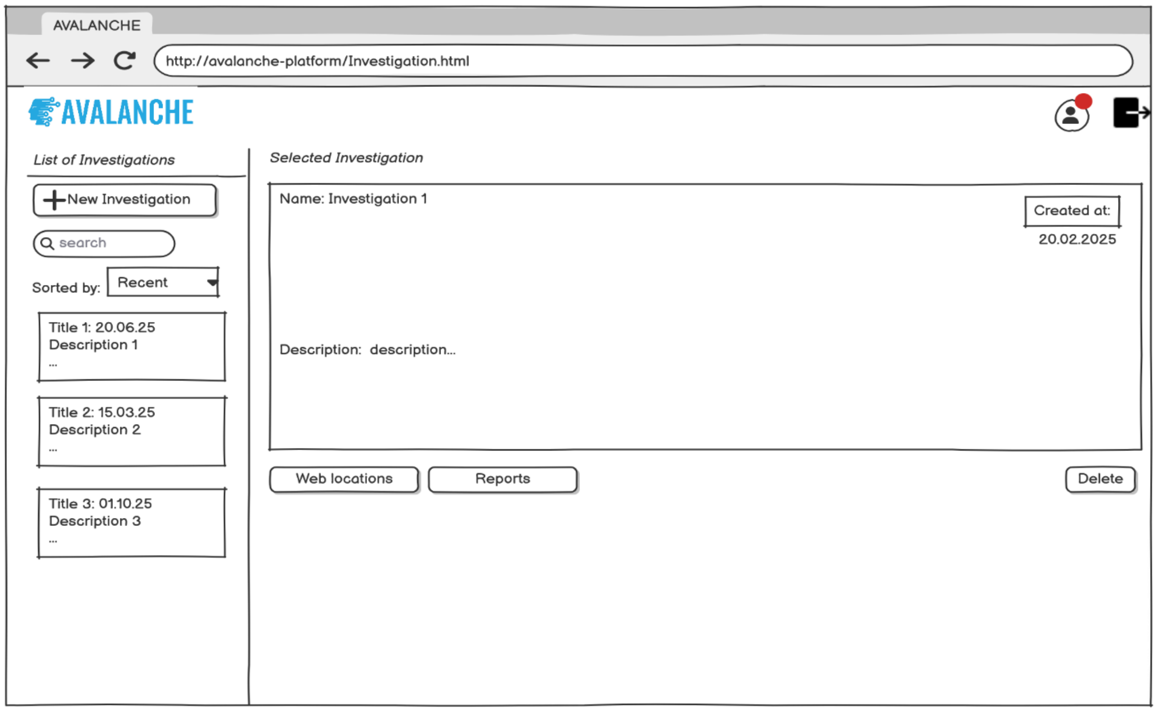
Creation of a new investigation and crawling a new web location

- **AVALANCHEs Maybe add 1 screen / figure Dashboard** (Figure 10):
    - User clicks "New Investigation".
    - A modal prompts for basic metadata: name, description, initial goals.
- **Investigation Detail Page** (Figure 11):
    - Investigation is created; user is directed here.
    - User clicks “Web locations”
- **Web location Detail Page** (Figure 12):
    - User sees an empty “Web Locations table” and clicks “+ Add new web location”.
    - A modal allows URL input and crawl settings (simple or advanced) (Figure 13)
- **System triggers crawl** (backend process).



Investigation Title	Description	Created by	Last modified	Flags (detected elements)	Reports
Investigation 1	title1	SPP officer	20/2/2025	hate speech	Report 1
Investigation 2	title2	SPP officer	06/06/2025	Deep fake	Report 2

Figure 10: AVALANCHE Dashboard



Title 1: 20.06.25
Description 1
...

Title 2: 15.03.25
Description 2
...

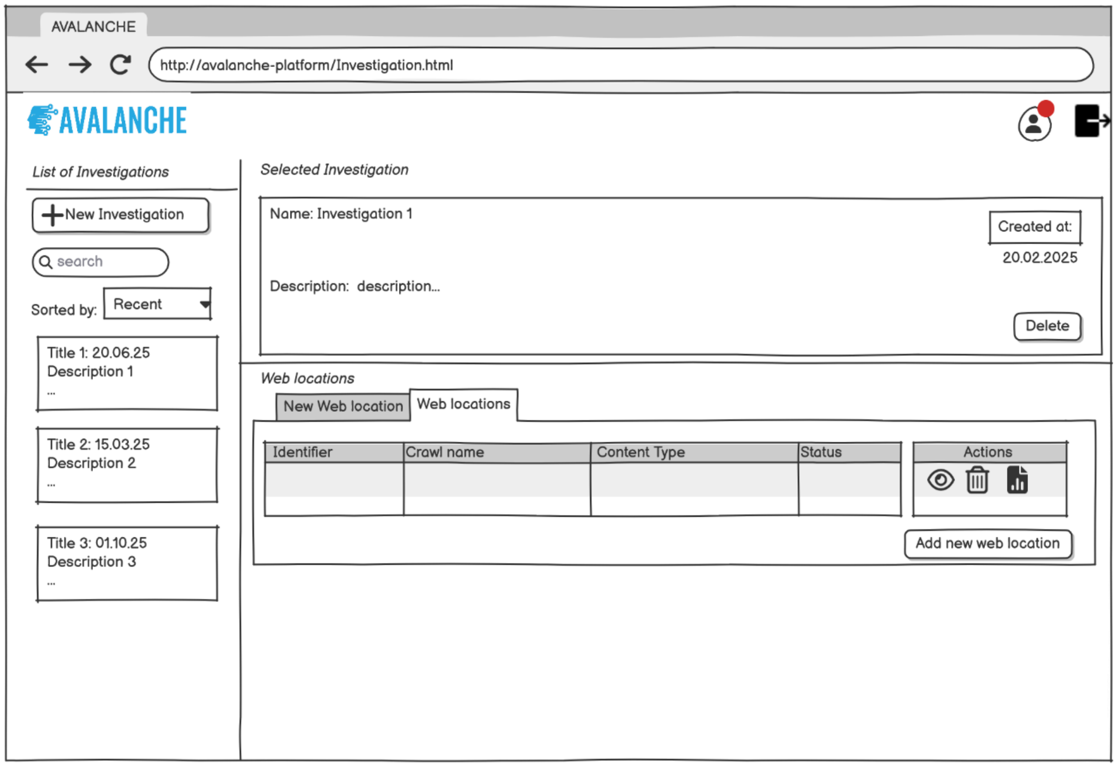
Title 3: 01.10.25
Description 3
...

Name: Investigation 1

Created at:
20.02.2025

Description: description...

Figure 11: AVALANCHE Selected Investigation Overview screen



AVALANCHE

http://avalanche-platform/Investigation.html

List of Investigations

New Investigation

search

Sorted by: Recent

Title 1: 20.06.25
Description 1
...

Title 2: 15.03.25
Description 2
...

Title 3: 01.10.25
Description 3
...

Selected Investigation

Name: Investigation 1

Created at: 20.02.2025

Description: description...

Delete

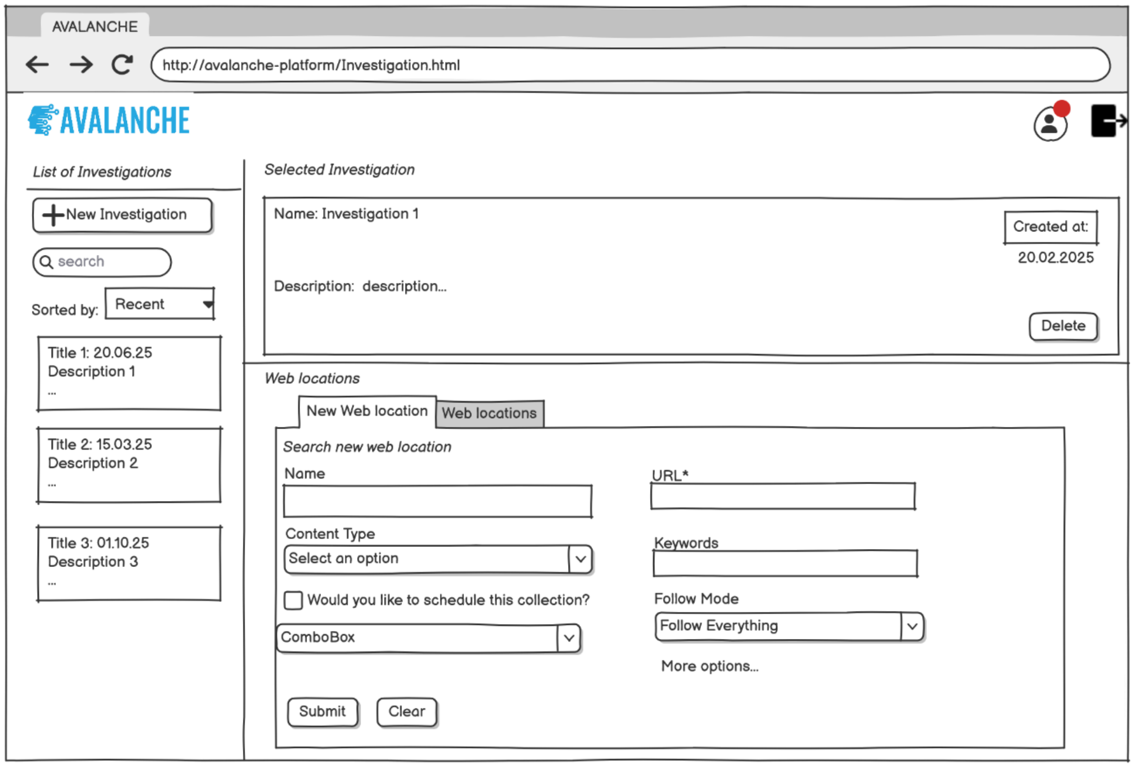
Web locations

New Web location | Web locations

Identifier	Crawl name	Content Type	Status	Actions

Add new web location

Figure 12: AVALANCHE; Selected investigation: Web locations



AVALANCHE

http://avalanche-platform/Investigation.html

List of Investigations

New Investigation

search

Sorted by: Recent

Title 1: 20.06.25
Description 1
...

Title 2: 15.03.25
Description 2
...

Title 3: 01.10.25
Description 3
...

Selected Investigation

Name: Investigation 1

Created at: 20.02.2025

Description: description...

Delete

Web locations

New Web location | Web locations

Search new web location

Name

URL*

Content Type: Select an option

Keywords

Would you like to schedule this collection?

Follow Mode: Follow Everything

ComboBox

More options...

Submit | Clear

Figure 13: AVALANCHE; Selected investigation: New Web location screens

Deeper analysis of initial crawl results

- Web location (initial crawl) is added to the investigation, with status and basic results populated. (Figure 14)
  - **Pane 1 (overview):** User views basic statistics and high-level summaries.
    - User can trigger further actions: e.g., “Timeline reconstruction”, “Actor relationships”, “Sentiment Analysis”, “Deepfake detection / analysis”, etc. These will likely be modal-based interactions tied to internal services, requested and fetched asynchronously.
  - **Pane 2 (crawl analysis)** (Figure 15)
    - User initiates **new deeper analysis** (modal) with configurable fields:
      - Keyword search
      - hate speech detection
      - disinformation detection
      - sentiment scoring,
      - etc.
    - User reviews results; including visualizations (Charts) and textual summaries
    - Interesting items can be flagged and “Added to Report”.
  - **Pane 3 (details):** For item-level drilldown and annotations.

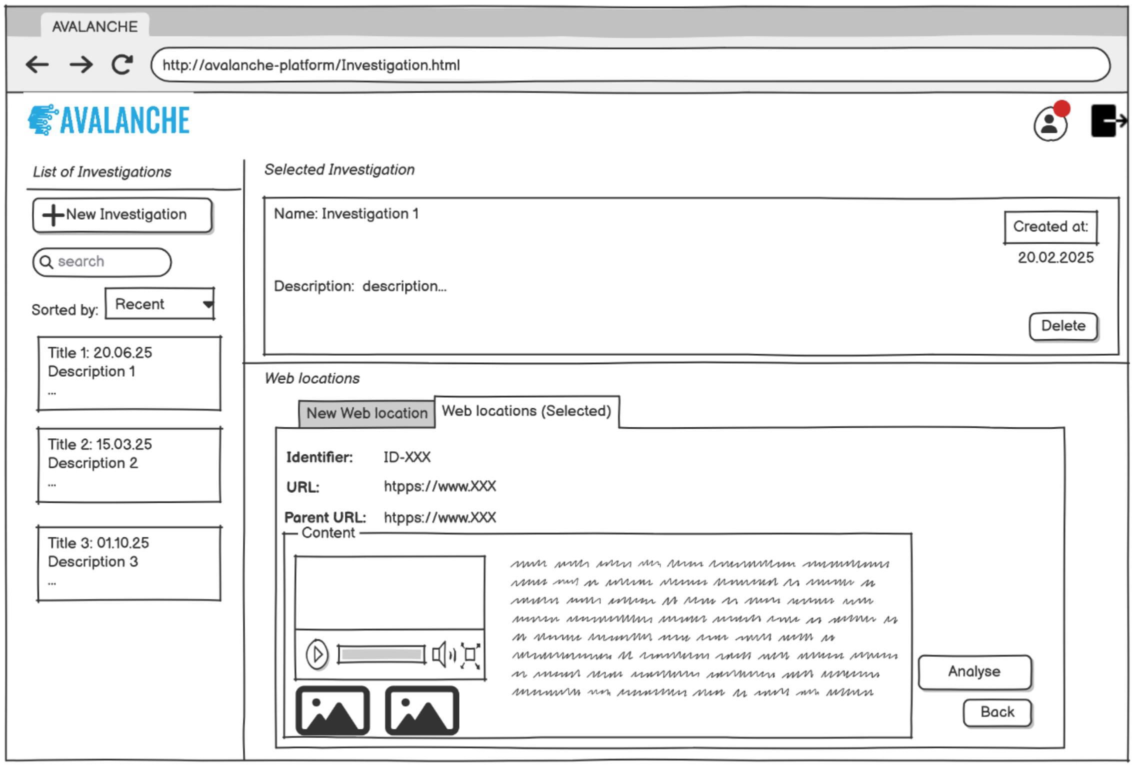
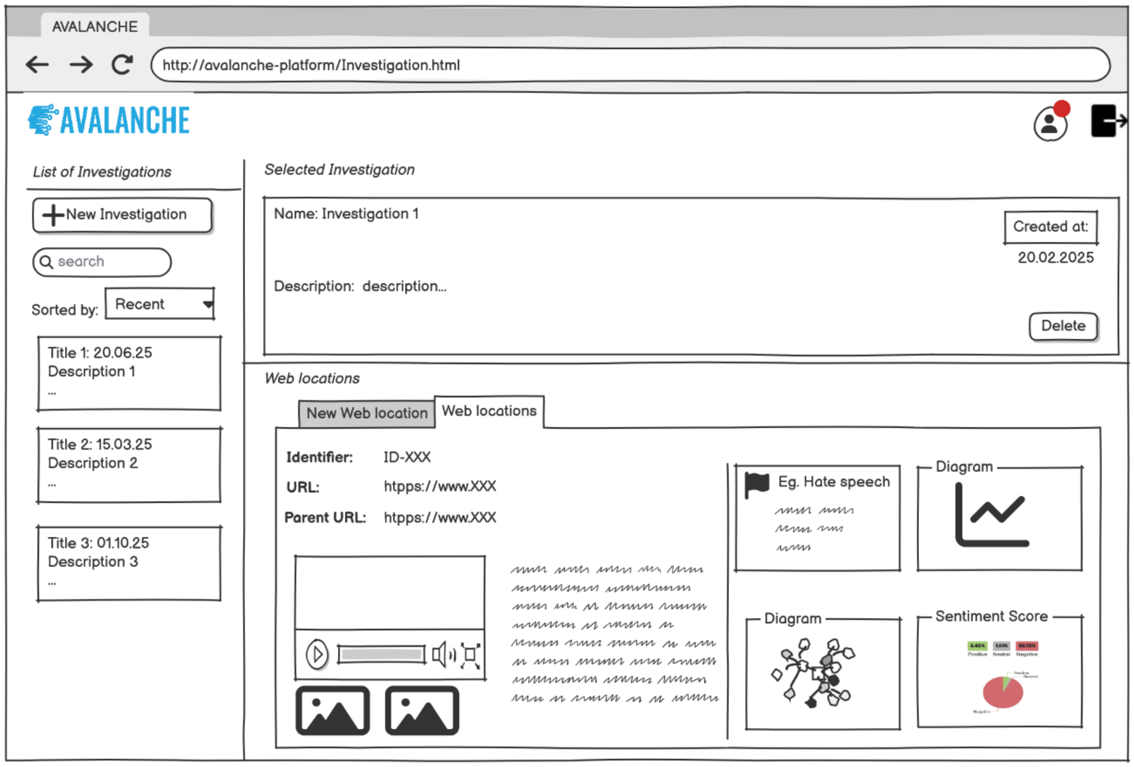


Figure 14: AVALANCHE; Selected investigation - Web location (selected) screen



Name: Investigation 1

Created at: 20.02.2025

Description: description...

Title 1: 20.06.25
Description 1
...

Title 2: 15.03.25
Description 2
...

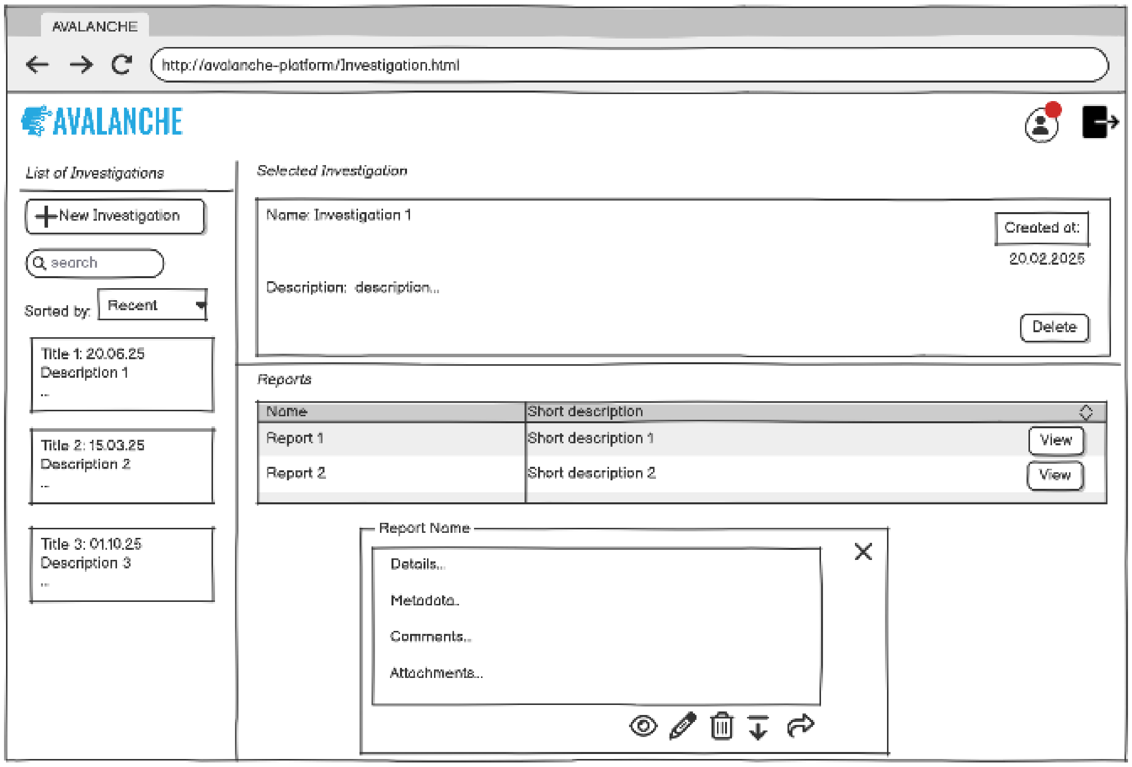
Title 3: 01.10.25
Description 3
...

Identifier: ID-XXX
URL: htpps://www.XXX
Parent URL: htpps://www.XXX

Figure 15: AVALANCHE; Selected investigation - Web location analysis screen

Creation and Viewing of a Report in PDF Format

- **From Investigation page:**
  - Scenario 1: Create report with all available data on the investigation
- **From each Web Location page:**
  - Scenario 2: User clicks on **"Reports"** pane or accesses "Add to my report" for individual items. User assembles content from flagged elements across web locations
- **Report Detail Page** (Figure 16)
  - A report will be structured into section-by-section, including findings, evidence, visualizations, and metadata.
  - Attachments and comments can be added.
  - User clicks "Download" for PDF export.
  - Report metadata is saved in the investigation context.



AVALANCHE

http://avalanche-platform/Investigation.html

AVALANCHE

List of Investigations

New Investigation

search

Sorted by: Recent

Title 1: 20.06.25
Description 1
..

Title 2: 15.03.25
Description 2
..

Title 3: 01.10.25
Description 3
..

Selected Investigation

Name: Investigation 1

Created at: 20.02.2025

Description: description...

Delete

Reports

Name	Short description	
Report 1	Short description 1	View
Report 2	Short description 2	View

Report Name

Details..

Metadata..

Comments..

Attachments..

Figure 16: AVALANCHE; Selected investigation: Reports – New Report screen

Secure sharing of a report with an external LEA

- **Report detail page**
  - User clicks "Share**"** from the list of report actions.
- **Sharing page**
  - User inputs recipient LEA's contact or selects from pre-approved organizations (based on roles/domains).
  - Secure sharing method is selected. The methodology for securing and encrypting data at rest and in transit is clarified here.

It is important to note that this is a generic, non-technical overview intended to convey the overall logic and structure of interactions. More detailed, technical representations of system behaviour are provided separately in the sequence diagrams section.

3.3 Component and Module design

This section provides a more detailed overview of the main components of the AVALANCHE architecture. For each module, it outlines the underlying business logic, along with relevant technical details, particularly those related to the functional requirements previously described in Section 2. Where applicable, a brief description of the internal architecture will also be included, along with the envisioned technology stack and tools to be used during the development phase of the AVALANCHE project.

3.3.1 Crawler

3.3.1.1 Business Logic

The Crawler is a key component of the AVALANCHE platform, designed as a flexible, powerful, and scalable backend service. At its core, it helps Law Enforcement Agencies (LEAs) gather vital online intelligence by using Big Data (BD) technologies. Built to meet the diverse needs of LEAs, it plays an essential role in identifying and analysing online content related to hate speech, disinformation, and deepfakes.

The main responsibility of the Crawler is to carry out automated, targeted web searches using specific URLs or keywords. It collects relevant data from the web and turns it into useful insights. For every webpage it visits, it creates a structured text version that can be further analysed to reveal patterns and trends. This helps investigators understand how harmful content spreads online and where it comes from. In addition to data collection, the Crawler classifies web pages based on their content and builds visual maps that show how it navigates between different URLs. These conceptual graphs give the connections between online sources. By continuously gathering high-quality data and offering initial analysis, the Crawler strengthens the overall ability of the AVALANCHE platform to monitor, detect, and respond to digital threats.

3.3.1.2 Functional description

Operating as a dedicated backend service within the AVALANCHE platform, the Crawler performs a robust set of integrated functions for data acquisition and preliminary analysis. It focuses on collecting content that is related to specific keywords provided by the user, which may in turn be associated with use cases such as hate speech, disinformation, and deepfakes.

The crawling process is initiated through service invocation by other AVALANCHE components, beginning with the UI, passing through the API Gateway, and then the Orchestrator, which provides essential parameters like seed URLs and a list of keywords. While the system can also commence crawling autonomously with predefined URLs related to known sources of content, this is outside the scope of this architectural description. A crucial initial step involves determining the search depth, which defines how many levels of hyperlinks the crawler will follow starting from the initial page. To ensure the search is highly relevant and efficient, the system is provisioned with necessary service configurations, proxies, and a specialized keyword list that narrows the scope of exploration to concepts pertinent to the user's search. Note that the Crawler archives and analyses content based on these keywords; the identification of hate speech, disinformation, or deepfakes itself is handled by the AI components within the AVALANCHE platform, not solely by the Crawler's keyword-based collection.

For data fetching, the Crawler utilizes a TOR proxy, enabling access to the dark web and anonymous communication on the surface. The system systematically gathers HTML content and associated metadata like fetch timestamps. To optimize efficiency and prevent redundant efforts, it maintains an internal record of visited URLs, ensuring the crawler avoids revisiting the same pages within the same crawling session. New URLs discovered during a crawl are added to an internal queue, which adheres to a Priority Queue strategy. Crucially, only URLs extracted from pages containing keywords pertinent to the focused search are enqueued, each accompanied by an importance score. In subsequent processing, URLs with higher scores are prioritized. The system also actively detects image/video(s) URLs (e.g., .jpg, .png, .mp4) during crawling and acquires the image data as a stream of bytes rather than HTML.

Upon retrieving HTML content, the Crawler undertakes data curation and harmonization. This involves converting raw HTML into clean, readable text by removing extraneous tags, JavaScript, and CSS code. Hyperlinks are extracted to identify subsequent URLs for crawling, while image tags are retained for later image processing. Regular expressions are employed to eliminate unnecessary characters, ensuring textual cleanliness.

All collected data is stored. Static metadata about the crawling processes, such as the total number of crawls and their status, is maintained in MongoDB. Images acquired are stored as objects in MinIO for subsequent analysis. All gathered textual data, including cleaned HTML and preliminary analytical results, is indexed and stored in Elasticsearch, enabling fast querying and retrieval by other AVALANCHE services.

Natural Language Understanding (NLU) is employed internally to identify and extract entities, a process that continuously enriches the keyword list used for focused crawling. The Crawler makes this curated entity data available for post-extraction by other AVALANCHE modules, which are responsible for deeper analysis and identification of specific issues like hate speech, disinformation, or deepfake indicators.

Note that all processed data and analytical results are stored in Crawler's designated data stores, ensuring they are readily available and consumable by the AVALANCHE platform's centralized GUI, Orchestrator, and other specialized services for comprehensive analysis and reporting.

3.3.1.3 Internal architecture & Technology stack

The internal architecture of the Crawler is designed for modularity, scalability, and robust data handling. The provided figure (Figure 17) illustrates the architectural components and data flows of it primarily as a standalone system.

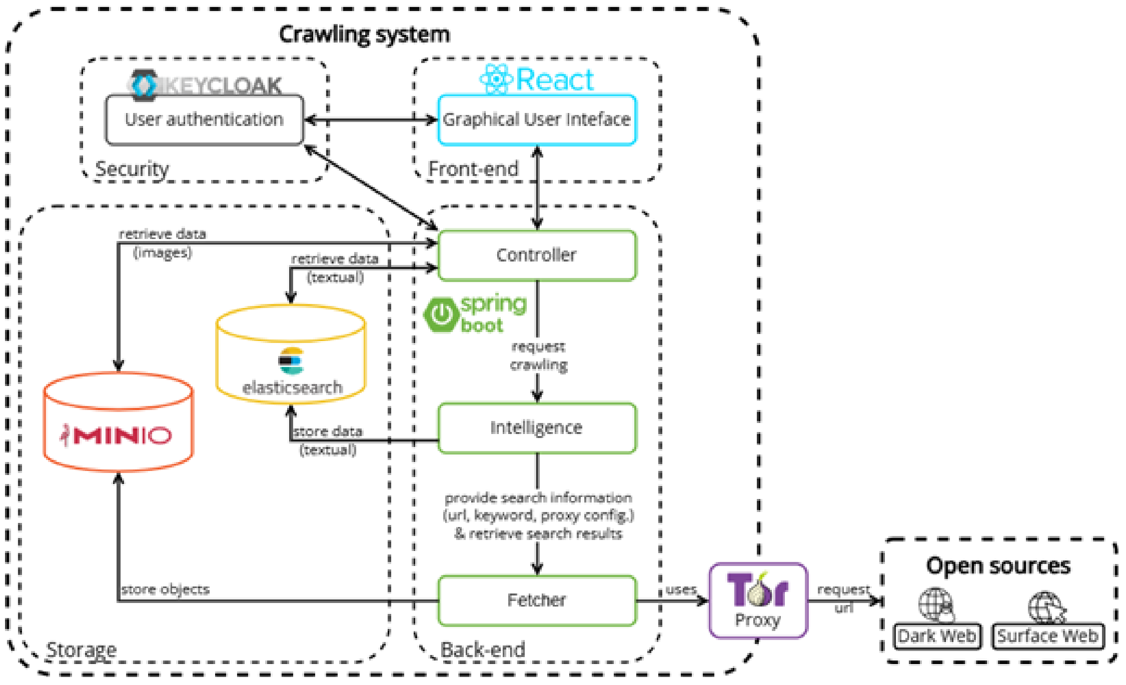


Figure 17: Crawler High level Architecture

It is crucial to understand that while this diagram depicts the crawler with its own GUI and Security layer, its deployment within the AVALANCHE platform fundamentally changes its operational context. In the AVALANCHE architecture, it functions exclusively as a core backend service, providing specialized crawling capabilities for hate speech, disinformation, and deepfakes to the larger platform. It does not possess its own direct user interface or manage user authentication independently. Instead, it is invoked by other AVALANCHE components, and its outputs are consumed by them.

Below, we detail the components shown in the diagram, explaining their role in the standalone context and clarifying their function and position when the Crawler operates as a service within AVALANCHE:

- **Security Layer (Centralized within AVALANCHE Platform):** In the depicted standalone crawling system, the User Authentication component, utilizing Keycloak, is responsible for managing user identities and granting access. However, within the AVALANCHE ecosystem, Keycloak serves as the central identity and access management service for the entire platform. This means user authentication is handled centrally by AVALANCHE. Consequently, the Crawler, operating as a service, does not perform its own user authentication; it receives pre-authenticated and authorized requests from other AVALANCHE services, relying on the platform's overarching security framework.
- **Front-end Layer (Managed by AVALANCHE Platform):** As illustrated in the standalone architecture, the Graphical User Interface (GUI), developed with React, would serve as Crawler's direct user interface. However, within the AVALANCHE platform, it will not feature its own dedicated GUI. User interactions for initiating crawls and reviewing results from Crawler's operations will occur through a distinct, centralized GUI provided by the AVALANCHE platform itself. The Crawler Controller will expose APIs for other AVALANCHE components to interact with it, which will then facilitate data presentation via the unified AVALANCHE interface.
- **Back-end Layer (Core Service Components):** This layer constitutes the essential processing and operational core of the crawling service. It performs the fundamental tasks of requesting, fetching, and initially processing web content for hate speech, disinformation, and deepfakes.
- **Controller** (Spring Boot): In the standalone system, the Controller orchestrates internal interactions. As an AVALANCHE service, the Controller functions as the primary API endpoint for the crawler. It receives authenticated requests from other AVALANCHE services (e.g., to initiate a crawl with specific parameters or to retrieve collected data). The Controller then directs these requests to the Intelligence module and manages data retrieval from the dedicated storage components.
- **Intelligence**: This critical component processes incoming crawling requests from the Controller. It is responsible for translating these requests into detailed search parameters (URLs, keywords, and proxy configurations). The intelligence module maintains and manages the entire crawling process, including the internal URL queueing logic and coordinating the initial processing and analytical tasks once data is retrieved. It also receives raw search results from the Fetcher.

- **Fetcher**: This is the dedicated data acquisition component. It utilizes the search information received from the Intelligence module to send requests for URLs to external web sources. The Fetcher communicates directly with the Tor Proxy to securely access both the Dark Web and Surface Web. Upon retrieving data (either raw HTML code or binary image data), the Fetcher directs it to the appropriate component: textual data are sent to intelligence for further analysis and consequently to be stored in Elasticsearch, while image objects are stored to MinIO.
- **Tor Proxy**: This specialized software component is integral for enabling anonymous communication with the Dark Web and supporting communication with the Surface Web. It acts as a secure gateway that the Fetcher uses to request and retrieve URLs from Open Sources (both Dark Web and Surface Web).
- **Storage Layer (Dedicated Data Stores):** This layer comprises Crawler's dedicated data stores, designed to efficiently handle different types of collected content and analytical results. These stores are internal to the crawling service but are accessible via the Controller's APIs for consumption by other AVALANCHE components.
- **Elasticsearch**: This distributed search and analytics engine is the primary repository for all gathered textual data (e.g., cleaned HTML code, fetch timestamps) from the surface and dark web. It indexes this data for fast querying and retrieval by Crawler's internal components and by the Controller for AVALANCHE consumption. It also stores the results of Crawler's preliminary textual and graph analytics.
- **MinIO**: Represented by the orange cylinder in the diagram, MinIO is an object storage solution specifically designated for housing all multimedia objects (images, videos, etc.) retrieved from the surface and dark web in their raw binary format. The Fetcher stores image objects directly into MinIO.
- **MongoDB**: While not explicitly shown with a distinct icon for content storage, MongoDB is an integral part of the Crawler's storage layer. It functions as the in-memory database for holding static metadata about the crawling processes, such as the total number of crawls, the URLs associated with specific crawls, and their status. This metadata is crucial for internal management and for providing context to the data consumed by AVALANCHE.

The way the AVALANCHE platform interacts with the crawling services is explained in detail in Section 3.2.1, where you'll also find a sequence diagram that illustrates and walks through the entire interaction flow.

Disclaimer
The technology described herein must be contextualized within the specific deployment scenario. Where used for criminal prevention, investigation, or detection, processing of personal data will fall under the Law Enforcement Directive, which may impose additional safeguards on the platform's operation that will be analysed in detail under D1.3.

3.3.2 Disinformation Detection & Analysis

3.3.2.1 Business Logic

The Disinformation Detection and Analysis module is a key part of the AVALANCHE system, built to tackle the rising problem of misleading and harmful content online. It uses advanced analysis tools to monitor and assess digital information, helping users better understand the accuracy and risks of what they're seeing. By offering clear, context-aware insights, the module supports smarter, faster decision-making and helps strengthen the system's ability to respond to disinformation threats.

3.3.2.2 Functional description

The Disinformation Detection and Analysis module offers a robust suite of capabilities designed to counter disinformation across various information sources. In accordance with REQ-DISD-001, the system enables content and author authenticity verification, source analysis, and fact-checking integration. It examines the origins of content to detect manipulation or context distortion and allows analysts to manage the list of monitored sources, both from the surface and dark web. In line with REQ-DISD-002, the system supports near real-time monitoring of open web sources to promptly detect potential disinformation, ensuring information remains timely and operationally relevant.

As specified in REQ-DISD-003, the platform also includes a reliability scoring mechanism that assesses the trustworthiness of content authors and sources across both the surface and dark web. Further, REQ-DISD-004 highlights the tool's ability to identify disinformation trends by detecting recurring narratives and coordinated messaging strategies.

Multilingual and culturally adaptive capabilities are addressed in REQ-DISD-005, enabling the tool to detect disinformation in multiple languages and recognize regional slang or cultural expressions. Moreover, REQ-DISD-006 ensures the system can flag suspicious content and provide users with contextual explanations, such as highlighting altered visuals or misleading statistics, while processing large-scale data in near real time.

On the analysis front, REQ-DISA-001 focuses on mapping disinformation networks, including users, groups, and bots, to reveal coordinated amplification efforts. It also offers explainability in understanding the connections within these networks. Finally, REQ-DISA-002 mandates the inclusion of a sentiment analysis feature, which categorizes content into positive, negative, or neutral sentiments, enriching the contextual analysis of flagged material.

Disinformation Detection Module and Crawling Service in AVALANCHE:

The AVALANCHE platform's crawling service is designed to identify and collect online content potentially linked to disinformation campaigns. When a user initiates a crawl with specific keywords of interest, such as words/phrases associated with conspiracy theories or misinformation trends, the system focuses on acquiring relevant data and content across the surface and dark web pertinent to those terms. In the following, we describe how disinformation detection is integrated into the crawling process:

- The user, through the AVALANCHE UI, provides seed URLs, onions for Dark Web and keyword parameters that guide the crawling process. These keywords are crucial for narrowing the scope of the search for relative content. For instance, a user might input

keywords related to a specific conspiracy theory about the president of a country or a known disinformation campaign.
- The crawling service, after receiving the user's keywords and seed URLs, systematically navigates the web (including the Dark Web via a Tor proxy). It prioritizes URLs containing the specified keywords and collects raw HTML content and associated metadata. Image URLs are also detected, and image data is acquired.
- As the HTML content is retrieved, the crawling service performs preliminary processing. This involves converting raw HTML into clean, readable text, extracting hyperlinks, and identifying image tags. Within this stage, the crawler leverages NLU internally to identify and extract entities from the text. This entity extraction can enrich the keyword list, further focusing the crawl on relevant concepts. Complementary NLP techniques are also applied to extract events from the text that is crawled. Those events will then be verified via our fact checkers, based on trusted sources.
- All collected textual data, including the clean HTML and preliminary analytical results (like content classifications and extracted entities), are indexed and stored in Elasticsearch for fast retrieval. Multimedia objects (images, videos) are stored in MinIO. This ensures that the data is readily available for other AVALANCHE components.

It's crucial to note that while the crawling service performs preliminary detection and categorizes content relevant to user-defined keywords, the analysis of disinformation is handled by other specialized components within the AVALANCHE platform. The crawler's role is to acquire high-quality, relevant data and provide initial insights. It makes this curated data available to those other AVALANCHE modules, which then takes on the responsibility of performing more sophisticated analysis to confirm and understand the nature of the disinformation.

In essence, the crawling service acts as an intelligent data acquisition and preliminary processing engine, meticulously gathering and organizing web content based on user-defined disinformation-related keywords and preparing it for deeper analytical scrutiny by the broader AVALANCHE platform.

3.3.2.3 Technology stack and Tools

The Disinformation Module and Crawling Service are built using the following technologies, enabling the implementation of the features previously described:

- Elasticsearch:
  - Used for indexing and storing all collected textual data, including clean HTML and analytical results such as content classifications and extracted entities.
  - Enables efficient search and retrieval across the dataset for other AVALANCHE components.
- MinIO:
  - Object storage system used to store multimedia content such as images and videos.
  - Ensures scalable and reliable storage of large, unstructured data files.
- Tor Proxy:
  - Utilized to access and crawl content from the dark web securely and anonymously.
  - Facilitates the expansion of the crawling process beyond the surface of the web.
- Natural Language Understanding (NLU):

  - Applied during the preliminary processing of crawled text to perform entity extraction.
  - Helps enrich the keyword list and guide more focused crawling.
- Natural Language Processing (NLP):
  - Used for extracting events from crawled text.
  - Supports subsequent fact-checking and analysis steps in the disinformation detection pipeline.

The core technologies that enable mapping and analysis of disinformation networks (REQ-DISA-001) are presented in the Behaviour Analysis Module (Section 3.3.6), which is responsible for managing and delivering this functionality.

The core technologies enabling sentiment analysis (REQ-DISA-002) are outlined in detail in Section 3.3.5. Two principal approaches will be adopted: BERT-based models and Large Language Models (LLMs). BERT-based architectures, particularly monolingual Romanian variants, exhibit high accuracy when fine-tuned on domain-specific corpora, making them well-suited for tasks requiring fine-grained sentiment classification. In parallel, LLMs such as GPT and Claude offer strong performance in zero-shot and multilingual settings, capable of handling sentiment dimensions like polarity, intensity, and affective state classification without additional training. However, LLMs present limitations in terms of output determinism, prompt sensitivity, and computational overhead. Consequently, for Romanian-language sentiment analysis with strict accuracy requirements, fine-tuned BERT models are the preferred solution.

3.3.3 Hate speech Detection & Analysis

3.3.3.1 Business Logic

Hate speech detection is a critical task in Natural Language Processing (NLP) that involves identifying language expressing hostility, discrimination, or incitement against individuals or groups based on characteristics such as race, religion, gender, sexual orientation, or nationality. This process analyses written or spoken content to determine whether it contains harmful or offensive language that qualifies as hate speech. Beyond identification, the goal is to facilitate responsible moderation, enforce policies, and support social interventions.

Within the AVALANCHE project, hate speech detection faces additional complexities due to its focus on Romanian, a language with limited NLP resources. The scarcity of large, high-quality annotated datasets hinders the development of models capable of nuanced and accurate detection. Moreover, hate speech often incorporates cultural references, local slang, idioms, and subtle sarcasm, which are challenging for standard multilingual models to interpret. Although Romanian monolingual models offer better linguistic alignment, they frequently lack coverage of informal and rapidly evolving language commonly found on online platforms.

To tackle these challenges, the approach is structured into two main stages: detection and analysis. The detection phase identifies potentially hateful content and provides transparent explanations for flagging decisions, supporting effective automation and human moderation. In the analysis phase, the system conducts a deeper investigation into the flagged content, tracking the spread of hate speech across networks. These insights inform targeted interventions, aid platform governance, and contribute to broader social research.

3.3.3.2 Functional description

The Hate Speech Detection Module is designed to comprehensively identify and analyse harmful content across the web. According to REQ-HSD-001, the module employs a binary classification model to distinguish hate speech from non-hate content. This includes the ability to detect not only explicit language but also subtle and coded expressions that may constitute hate speech in various contexts. To enhance interpretability and reduce false positives, the system also adheres to REQ-HSD-004, which ensures the distinction between genuine hate speech and the benign use of potentially offensive language, such as that found in academic discourse or satire, by leveraging contextual analysis. A core functionality outlined in REQ-HSD-005 focuses on detecting hate speech in Romanian, with careful attention to cultural and regional nuances, ensuring reliable identification within national linguistic contexts. While Romanian detection is a strategic requirement (S), support for other languages is considered conditional (C) and will be explored incrementally.

Beyond detection, the module offers advanced analytical capabilities. In compliance with REQ-HSA-002, the system classifies hate speech by providing deeper insight and enhanced explainability for decision-makers. Additionally, as described in REQ-HSA-003, the system is capable of mapping networks involved in the propagation of hate speech. This includes identifying coordinated campaigns and recurrent actors, which is critical for understanding patterns, supporting both the Hate Speech and Behavioural Analysis services. Per REQ-HSD-006, a continuously updated database of hate speech terms is maintained, incorporating evolving slang and coded language used by specific groups or movements. Furthermore, REQ-HSD-007 emphasizes the analysis of user behaviour to identify accounts that consistently engage in hate speech or exhibit patterns of targeting individuals or groups. This functionality supports both investigational workflows and longer-term strategic assessments (G1.1.6). It is important to note that the feasibility assessment of the outlined requirements is provided in the corresponding Section 2.

3.3.3.3 Technology stack and Tools

In the following section, the outcomes of the research which was made to address hate speech detection task is presented. BERT models offer a strong foundation for hate speech detection, especially when fine-tuned with task-specific data. Romanian-specific BERT variants such as BERT Base Romanian [2] [3], RoBERT[3], and Bertweetro[4] [4] outperform their multilingual counterparts, largely due to their focused linguistic training. However, they require significant time investment for customization and programming, which may be a bottleneck under tight project deadlines. A major advantage is their customizability, which allows the integration of rules and domain-specific knowledge into the pipeline.

Large Language Models, on the other hand, provide flexibility via zero-shot and few-shot learning capabilities. These enable LLMs to handle hate speech tasks without needing extensive labelled data. However, their effectiveness in this area is inconsistent. Performance varies across models, and reliance on prompt engineering means that results may differ widely with slight input changes.

[2] Bert Base Romanian

[3] RoBERT

[4] Bertweetro options

Moreover, LLMs' inherent ethical constraints and hallucination risk present reliability challenges for production use [5].

Table 11 summarizes the pros and cons of the previously presented methods.

Table 11: BERT Models vs. LLMs in Hate speech detection

Approach	Pros	Cons
BERT Models	- Performs well when fine-tuned - Available monolingual Romanian models (higher efficacy) - Customizable for specific tasks	- Dataset limitations in slang and idioms - Language-specific training doesn't always generalize well - Customization takes time
LLMs	- Works with low-resource languages - Zero-/few-shot for quicker prototyping	- Highly variable performance - Hate speech rules can conflict with model behaviour - Inconsistent and sometimes unexpected outputs

3.3.4 Deepfake Detection and Analysis

3.3.4.1 Business Logic

Following the use case definition carried out in Work Package 3 during the project's first six months, it was evident that the detection and analysis of deepfakes was an essential element requested by the end-user of the project (SPP). The focus of SPP as an organization is the protection and defence of dignitaries including their families and relatives. Deep fakes are a common issue that affects dignitaries across the globe and thus the problem is a core focus of the organization.

Deepfakes refer to the cases where audio, video or images that appear real and are digitally altered through simple editing software as well as Artificial Intelligence to deceive someone with regarding to what they did or said. Thus, considering the popularity and exposure of the dignitaries, deep fakes are commonly affecting them. Deep fakes constitute a wide category of cases considering that there are different element types that can be altered such as audio, video or images. Furthermore, there are different cases of alternation that may take place such as face swap, expression manipulation and many more depending on the media type that is being manipulated.

Overview and Challenges in AVALANCHE solution
In AVALANCHE, deepfake detection and analysis were not foreseen in the initial description of work. However, considering the alternation of the use cases and the user requirements extracted in WP3, the consortium focused on conducting research and development of a tool that will enable the detection and analysis of deep fakes.

Currently, the approach followed by the LEAs to detect if a media file is deep fake consists of empirical evaluation of the content to understand if it is a deep fake. In some cases, the media file is obviously a deep fake considering that there are clear indications of the alternation of the media file. However, in cases where the alternation has been performed through Artificial Intelligence, there are cases where it is very difficult to understand if the media file has been altered digitally. In such cases, the LEAs may need to cross-check the media file across multiple sources including various credible websites to ensure that the content of the media file is reliable or has been digitally altered.

In AVALANCHE, the focus will be to essentially detect if a media file has been digitally altered to perform disinformation. The component will receive as input a media file which may have been altered or not and will provide an output result indicating to the LEAs if the media file is a deepfake or not.

3.3.4.2 Functional description

Deepfakes detection constitutes in essence a binary classification problem, given a media storage file, with the sole focus on estimating if the media file is legit or has been digitally modified. Depending on the type of the media file, different types of digital modification may take place. Since the media files could be audio, video or images, and each of these file types may be affected from various digital modifications, there is no solution that fits all these different cases, creating a multidimensional problem. For this reason, within the context of AVALANCHE the focus of the research will be narrowed down to a specific media file type as well as to certain types of digital modifications, considering the limited timeframe. The tool will be interoperable with other components and systems, by exposing APIs using standardised authentication mechanisms. To ensure the security of the tool, only authenticated requests will be allowed, while the component is envisioned to operate without requirements for Internet connectivity considering that most of the systems leveraged by LEAs operate offline and at restricted intranets. The tool is envisioned to operate as a stateless processor, meaning that the tool will receive as input a media file and in near real-time will provide a response regarding if the media file has been digitally modified or not. Thus, the tool will not be required to store the result, as in common architectures the storage of the result of the processing is conducted by the system/tool that performs the request to the deepfake detection tool. Further details on the identified system requirements are provided in *Table 5* and *Table 6*, Section 2.

3.3.4.3 Internal architecture &Technology stack and Tools

The detection and analysis of deep fakes requires primarily technologies such as Artificial Intelligence and Machine Learning that will enable the avoidance of complex mathematical modelling required in such multi-dimensional problems. Artificial Intelligence and machine learning models are predominately built through the programming language Python. Python is a programming language that is widely used by data analysts and machine learning engineers in order to quickly build models, while having a wide support community and a large ecosystem of libraries that speed up the development of the appropriate preprocessing steps. It has a clear and intuitive syntax that enables ML engineers to quickly create a prototype and focus on solving machine learning problems. Python incorporates a large variety of libraries and frameworks that enable the processing of media files (e.g. audio, image, and video), filtering and conversion into the appropriate format.

For the development of machine learning and artificial intelligence models, Python incorporates libraries such as sci-kit as well as Keras/Pytorch which provide the appropriate abstraction required. Technologies such as Jupyter Notebooks allow ML engineers to quickly create the process of reading the data, conduct the appropriate preprocessing, train the model and then validate the trained model based on standardized performance metrics such as accuracy, precision and recall. Evaluating the trained models is important to understand the performance of the model, especially on previously unseen data. Another important aspect of the Jupyter Notebooks is that the code can be executed in sections Meaning that the data can be loaded only once and the step by step the ML engineer can test and validate the various preprocessing and post processing mechanisms in order to understand their importance, as well as easily debug potential issues or errors.

Having developed machine learning and artificial intelligence models, the component will expose a specific API that enables communication with other components. Communication will enable serving a specific request incorporating the input and providing the response to the request. A commonly used framework for serving such requests is FastAPI. It constitutes the equivalent of Node.js in Python, thus enables event-based processing without locking the processing for simultaneous requests. FastAPI constitutes a fast-growing framework incorporating various security mechanisms while enabling fast prototyping.

3.3.5 Sentiment Analysis

3.3.5.1 Business Logic

The sentiment analysis service within the context of the AVALANCHE solution involves examining text data, sourced from both the open and dark web, to identify the emotional tone as positive, negative, or neutral. The following sections provide an overview of sentiment analysis and outline the key challenges specific to its application in AVALANCHE. Additionally, they highlight its alignment with previously defined functional requirements and include extensive research of existing models that could be utilized during the implementation phase (WP5).

Overview and Challenges in AVALANCHE solution
Sentiment analysis, while more mature as a task in natural language processing, faces similar obstacles in low-resource languages. Romanian models suffer from limited dataset sizes, often capping at around 5,000 records, which restricts their effectiveness in capturing subtle sentiment gradations. Additionally, the lack of slang and idiom inclusion in training data further limits generalization in informal contexts.

Despite these limitations, sentiment analysis benefits from a wealth of research and well-established evaluation metrics. The task lends itself well to benchmarking and is often used to validate the baseline performance of language models. Section 3.3.5.3 presents the results of the conducted research of existing sentiment analysis methodologies, with a particular focus on addressing the specific needs of the SPP, including support for the Romanian language.

3.3.5.2 Functional description

The AVALANCHE platform integrates a sentiment analysis module as an essential analytical component, designed to assess the emotional tone and communicative intent of textual data. In accordance with REQ-DISA-002, the system must implement a sentiment analysis algorithm capable of classifying content into one of three categories: positive, negative, or neutral. This foundational capability supports the platform’s broader analytical functions by offering nuanced interpretation of user-generated content. Building upon this, REQ-HSD-008 outlines that sentiment analysis functionality should also evaluate tone and intent to determine the nature of communication. While this requirement is originally linked to hate speech detection, it reinforces the system’s ability to perform sentiment-based interpretation as a standalone analytical layer.

Both requirements are fulfilled through the Sentiment Module, which is aligned with user requirements, ensuring consistency and integration across multiple services within the platform's internal architecture.

3.3.5.3 Technology stack and Tools

The following section presents the findings of the research conducted to address the task of sentiment analysis. BERT-based[5] [6] models demonstrate excellent benchmark performance in sentiment analysis, particularly when fine-tuned with relevant data. Monolingual Romanian models again offer better efficacy compared to their multilingual versions. Their customizability makes them ideal for domain-specific applications where nuanced sentiment categories are necessary [5], [7].

LLMs perform particularly well in sentiment analysis, especially in multilingual settings. Even without fine-tuning, models like GPT and Claude can achieve competitive results through zero-shot inference. Their inherent multitasking ability allows them to manage various aspects of sentiment classification, including polarity, intensity, and emotion recognition. However, the lack of result standardization, high computational demands, and sensitivity to prompt structure are notable drawbacks [8].

Sentiment analysis pipelines are easier to implement making LLMs more viable for rapid prototyping and deployment. However, for applications requiring high accuracy, especially in Romanian, BERT models with dedicated tuning remain the most promising option.

Table 12 below summarizes the pros and cons of the previously presented methods.

Table 12: BERT Models vs LLMs in Sentiment Analysis

Approach	Pros	Cons
BERT Models	- Strong performance in benchmarks - Romanian monolingual models available and effective - Great customizability	- Requires fine-tuning for optimal results - Limited coverage for slang and idioms - Limited datasets for low-resource languages
LLMs	- Zero-shot/few-shot capability (less task-specific data needed) - Strong multilingual performance - Minimal customization required - Good sentiment benchmarks	- Heavily reliant on prompt engineering - Resource intensive - Potential for hallucinations and result variability

3.3.6 Behavioural Analysis

3.3.6.1 Business Logic

Behavioural Analysis constitutes a systematic examination of online and digital content to understand intent, influence patterns, and potential human behaviours behind its creation. Understanding patterns that could indicate the existence of a specific type of content e.g.

[5] Romanian NLP datasets

fraudulent, manipulative etc. also fall under the scope of the behavioural analysis. Furthermore, information that could indicate how the content could change over time, across different occurrences (e.g. disinformation spread across different websites), following certain patterns. For this reason, apart from directly focusing on detecting specific categories of content, research has shown that it is possible to explore the behaviours and patterns related to the content in order to strengthen the detection of such content, as well as to monitor and predict the evolvement as well as to derive additional intelligence related to the content.

Towards addressing this growing need, behavioural analysis has evolved into a multi-disciplinary and multi-faceted problem that requires a combination of various emerging technologies such as natural language processing, pattern recognition, machine learning and potentially clustering of the respective content. The common approach constitutes direct classification of the content into the targeted classes or categories of information. However, to acquire a more spherical understanding of the actual content, it is important to reveal how malicious or manipulative content evolves, emerges, spreads and adapts across the time as well as across different online websites, platforms and potentially languages. For example, the same narrative, such as a conspiracy theory or a hate-inciting phrase, may appear in different variations across different dark web forums, videos, or mirrored pages. Thus, it is important to be able to cluster the content based on the narrative and quickly assist the LEAs to understand the similarities among the different pages, facilitating and speeding up the investigation process. Clustering similar content, identifying potential origin of spread and understanding the evolution over time of the content, is important for the LEAs to understand the actionable insights required as well as the risks associated with the content.

Overview and Challenges in AVALANCHE solution

Following the revision of the targeted use cases, the primary focus of the AVALANCHE project is to detect and analyse cases of disinformation, hate speech and deep fakes, as well as interconnection and secure data exchange. Apart from the various detection mechanisms developed within the project, with respect to disinformation, hate speech and deep fakes, it is essential to assist LEAs by enabling the analysis of the detected cases and providing additional intelligence. This additional intelligence may assist LEAs to perform further correlations with related content across the dark web and potentially the surface web. The goal is to enable and facilitate the investigation process and the internal procedures following currently by LEAs in order to minimise manual work and analysis.

In the content of AVALANCHE, the behavioural analysis tool will build on top a technology that has been developed for behaviour analysis on transactions, in order to derive additional intelligence with respect to the targeted categories of content, i.e. disinformation, hate speech and deep fakes. The behavioural analysis tool will leverage information received from the Crawler module, a crawler focused on collecting and preprocessing content from the dark web, (in the context of AVALANCHE) targeting disinformation, hate speech and deep fakes. The crawled content will include primarily information such as the extracted text from the dark web page, as well as the timestamp when the page was crawled and the parent URL from which the crawler arrived at the page. The purpose of the behavioural analysis tool is to leverage the information provided by the Crawler module to extract patterns (e.g. potential origin of spread) towards assisting the LEAs in deriving higher level of information as well as classifying more accurately the content with respect to the targeted categories i.e. disinformation, hate speech and deep fakes.

3.3.6.2 Functional description

The Behavioural Analysis tool is focused on automatically process, classify and analyse textual content that has been collected from the dark web, through potential crawlers such as Crawler module (in the context of AVALANCHE). The tool incorporates various technologies to assist the investigation process such as classification of the content of the dark web page into disinformation, hate speech and deep fakes based on certain preselected keywords. The input received from the crawler includes various types of information extracted during the crawling process such as the timestamp of the visit of the web page, the actual URL of the page as well as the parent URL which the crawler arrived at the particular page, the number of links that may direct the user to other pages, some additional fields about the structure of the page and the extracted text which constitutes the body of the page. The Behavioural Analysis tool is focused on leveraging the aforementioned information to derive patterns and higher level of information towards assisting the investigation process. A language detection process is conducted as a pre-processing step in order to assist the tool in optimising the behaviour analysis in the specific language. Through natural language processing and language specific keywords, the tool estimates the top-k labels meaning the most likely content categories (i.e. disinformation, hate speech and deep fakes) along with their confidence scores to enable the LEA to understand the content. Following, once each of the crawled content items has been classified, they are grouped through clustering into topic specific content and accompanied by dominant keywords along each cluster. Additionally, the system extracts salient keyword phrases from each item—primarily using sentence-level scoring; to summarise the core behavioural signal of each crawled item. In this way, LEAs are able to identify cases with coordinated campaigns of disinformation and deep fakes. Furthermore, the tool monitors the first occurrence of such specific campaign in order to estimate a potential origin of spread.

3.3.6.3 Internal architecture & Technology stack and Tools

The internal architecture of the Behavioural Analysis tool is built in a modular manner in order to enable the extension of the tool with additional functionalities. The tool has been designed to enable the processing of batches of crawled items received from the crawler. Currently the tool supports the input from the Crawler, thus in order to support additional crawlers there is a need to modify the expected input format. A sequence of components is currently incorporated in the tool such as detection, zero-shot classification, semantic embedding generation, clustering, keyword extraction, and deduplication. Each of these components has been integrated into a service-like functionality, conducting sequential analysis, while enabling concurrent requests in an event-based manner.

The technology stack is primarily based on Python, which is dominant and widely used in data science and data engineering. For language detection, it uses Lang detect, while zero-shot classification is powered by Hugging Face’s transformers library and the joeddav/xlm-roberta-large-xnli multilingual model. Semantic embeddings are generated via Sentence Transformers using the paraphrase-multilingual-MiniLM-L12-v2 model, ensuring language-agnostic clustering. Clustering is performed with HDBSCAN, a density-based clustering algorithm suitable for discovering variable-sized groups of related content. For storage, the component temporarily incorporates SQLite during the development process. However, more scalable storage is envisioned to be incorporated such as Postgres.

The tool is deployed as a RESTful API using the FastAPI framework, which offers high performance, validation, and ease of integration. FastAPI is the equivalent of Node.js in Python, following the event-based processing without enforcing a synchronous approach, creating bottlenecks in high processing and traffic.

3.3.7 AVALANCHE Web Platform & UI

3.3.7.1 *Business Logic*

The AVALANCHE Web Platform serves as the primary interface where users, particularly Law Enforcement Agency (LEA) officers, interact with the system. This module provides a unified, user-friendly dashboard that integrates all core functionalities of the AVALANCHE platform. It offers a suite of front-end applications, each designed to support various operational tasks such as data analysis, incident detection, and secure evidence exchange.

Leveraging the Decision Support Enabler, the platform delivers insights through interactive interfaces and advanced visualizations, enabling users to understand complex data in an explainable manner. These visual tools are specifically engineered to assist LEA officers in their investigations, directly supporting the use cases defined in the previous section (e.g., disinformation detection, hate speech analysis, deepfake identification).

The AVALANCHE UI is not just a static dashboard; it is an adaptive environment that empowers users by:

- Allowing customized analysis and reporting tailored to the specific needs of ongoing investigations.
- Supporting decision-making through analytics and user-defined countermeasures.
- Providing streamlined user experience, enabling the management of multiple rules and response actions with ease.

In essence, the AVALANCHE Web Platform UI is designed to be an intuitive, scalable, and secure interface, ensuring that LEA officers can efficiently perform their tasks and make informed decisions.

3.3.7.2 *Functional description*

The AVALANCHE Web Platform is designed with a range of functional capabilities that ensure it meets user needs while maintaining secure and efficient operations.

At the core of user interaction is the authentication process. Users begin their journey by securely logging into the platform (REQ-G-001), where their credentials are verified, ensuring that only authorized personnel gain access. For new users, the platform offers a straightforward registration process (REQ-G-002), allowing them to create an account with secure authentication methods. Once their session is complete, users can securely log out (REQ-G-003), terminating their access to the system. To maintain a secure operational environment, the platform implements Role-Based Access Control (REQ-G-004), where user roles are clearly defined, and each role is granted specific permissions. This is reinforced by a comprehensive User Identity and Access

Management system (REQ-G-005), which allows administrators to manage user accounts, enforce password policies, and configure access permissions directly through the user interface.

The platform's core functionality revolves around data crawling. Users can configure and initiate data crawling tasks (REQ-G-006) by specifying the sources (surface or dark web), frequency, keywords, and time range. Once data is collected, it can be viewed, filtered, and managed (REQ-G-007). Users can interact with various data types, such as text, images, videos, and audio, applying filters to refine their analysis.

Beyond mere data collection, the AVALANCHE platform empowers users with advanced analytics. Users can initiate crawler data analytics (REQ-G-008), gaining insights into the collected data. They can also perform sentiment analysis (REQ-G-009), visually interpreting the data's emotional context through graphs and sentiment scores. For more targeted insights, users can trigger disinformation detection (REQ-G-010), hate speech detection (REQ-G-011), and deepfake detection (REQ-G-012), with each analysis type presenting results in a clear, explainable format, including confidence scores and detected manipulations.

Throughout the analysis process, users can monitor the status of their detection requests (REQ-G-013), ensuring they are informed of whether tasks are queued, processed, completed, or encountered errors. Moreover, the platform offers granular control over task access (REQ-CR-014), allowing the creator of a task to manage who can view or modify it, ensuring data security and traceability.

For greater flexibility, the system supports data export (REQ-CR-015), allowing users to extract analysis results in CSV or JSON formats. Additionally, users can report suspicious media (REQ-CR-016) for further analysis, aiding in identifying potential disinformation or deepfakes.

Finally, recognizing the multilingual nature of modern information sources, the platform is equipped with a multilingual text translation feature (REQ-G-017), allowing text content to be translated between multiple language pairs, including English, Romanian, Arabic, Russian, Ukrainian, and Hungarian.

Together, these functionalities make the AVALANCHE Web Platform a powerful tool for secure data collection, analysis, and user interaction, tailored to the needs of Law Enforcement Agencies (LEAs).

A more consolidated description of the functional requirements of this module is given in Section 2 in *Table 8*.

3.3.7.3 Technology stack and Tools

The web platform utilizes the following technologies to deliver a highly interactive, responsive, and scalable user interface:

- React: The core of the front-end development, React allows for the creation of dynamic, component-based architecture. This enables fast, efficient rendering and seamless user interactions, with the ability to easily update the UI in response to user actions or data changes.

- Material-UI: This popular React component library adheres to Google's Material Design principles, providing a wide range of pre-designed, customizable UI components. These include elements like buttons, cards, grids, and modals, ensuring that the interface is visually appealing, consistent, and responsive across different screen sizes. Material-UI's theming capabilities also allow for easy customization, aligning the platform's look and feel with brand requirements.
- Apache ECharts: Integrated into the platform for advanced data visualization, Apache ECharts is a flexible charting library that enables the display of interactive, dynamic charts. It supports various chart types, such as bar, line, pie, and heatmaps, offering an intuitive way to represent complex datasets. Users can engage with these charts, exploring data interactively through zoom, tooltip, and drill-down features.

This stack ensures a modern, efficient, and user-friendly platform, with a focus on performance, interactivity, and responsive design.

3.4 Interconnection & Information Exchange among LEAs

3.4.1 Overview

AVALANCHE needs to provide secure, interoperable, and efficient information exchange between LEAs and/or other related and authorized organisations, also considering cross-border scenario. This is the goal of Task 5.4 of the project, which addresses the challenge of connecting heterogeneous, distributed, and institutionally siloed data. Such data might be spanning structured databases, RESTful APIs, and open- or closed-source intelligence services. T5.4 will design and implement a federated architecture which will ensure semantic coherence, integrity, and compliant access control.

3.4.2 AVALANCHE's federated schema

To support semantic interoperability across international databases and services, AVALANCHE will adopt and extend a data schema, based on the Schengen Information System (SIS II) data exchange model, as specified by the European Commission. The core SIS schema provides a solid foundation for handling entities such as persons, objects, alerts, and administrative metadata across agencies.

This schema will be adapted and extended to accommodate the additional data types and operational nuances specific to AVALANCHE's scope, for items specifically selected by LEA operators for sharing, such as:

- Specific annotated OSINT items from the surface or dark web (e.g., hate speech indicators, disinformation artifacts, flagged deepfakes)
- AVALANCHE reports or structured investigation findings / case summaries
- Evidence-supporting and/or chain-of-custody data or metadata, including verification hashes for integrity and non-repudiation

To facilitate interoperability and extensibility, the extended schema will support both JSON and RDF-based serializations. Schema stitching will be applied as necessary to merge and harmonise ambiguous or unstructured data and align disparate naming conventions.

3.4.3 Secure AVALANCHE API endpoints for data exchange

AVALANCHE will expose the federated data layer via a unified GraphQL API, orchestrated through an Apollo Federation gateway. This architecture will enable secure aggregation and invocation of data endpoints while respecting schema boundaries, access permissions, and provenance.

Key technical features will include

(i) integration of data structures representing specific LEA or judicial domains into the AVALANCHE domain model (Section 3.4.2),
(ii) integration with UBI's ubi:fedquery framework for asynchronous multi-source lookups,
(iii) a centralized mechanism to combine schemas and gateway to serve data consumers securely, and
(iv) logging mechanisms for API access traceability and accountability.

From a security perspective, the above will be supported by AVALANCHE's security layer for role-based access control and enforcement, leveraging Keycloak and several PETs, with interactions additionally subject to strict encryption (TLS 1.3+), digitally signed request tokens, contextual authorization checks, and an optional layer of encryption with pre-shared keys.

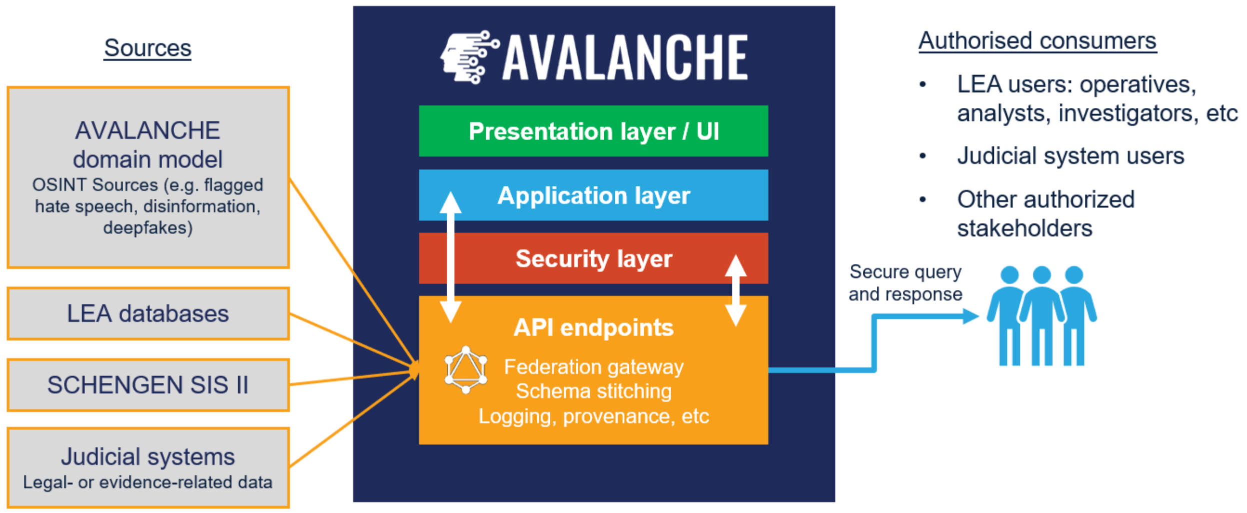


Figure 18: Conceptual view of AVALANCHE's Information Exchange architecture

This approach enables approved third parties (e.g., participating LEAs, prosecutors, courts, investigators, etc) to query and retrieve cross-border data (those that have been deemed shareable by AVALANCHE analysts and operators) under applicable privacy and legal constraints, such as the items mentioned in Section 3.4.2.

By combining schema federation, semantic enrichment, and secure APIs deployment, AVALANCHE will provide a robust backbone for trusted LEA collaboration, balancing operational needs with legal and technical safeguards.

3.5 System Requirements and Module Mapping

This section provides a mapping between the system's functional and non-functional requirements (FRs and NFRs) and the corresponding modules and infrastructure components of the AVALANCHE platform, as presented in Section 3.1. It is important to emphasize that the comprehensive specification of system requirements, together with their corresponding feasibility evaluations, is presented in Section 2.1 and Section 2.2 of this document. By establishing these connections, the section ensures clear traceability from requirements to implementation, reinforcing alignment between stakeholder expectations and the system's architecture. This approach promotes transparency in the system's design, facilitates maintainability, and helps guide future enhancements based on requirement coverage and modular responsibility; as shown in Table 13.

Table 13: System Requirements and module mapping

Mapped Components	**FRs & NFRs Names**
AVALANCHE web platform	REQ-DISD-002 REQ-DISD-006 REQ-HSA-001 REQ-G-001 REQ-G-002 REQ-G-003 REQ-G-004 REQ-G-005 REQ-G-006 REQ-G-007 REQ-G-008 REQ-G-009 REQ-G-010 REQ-G-011 REQ-G-012 REQ-G-013 REQ-G-014 REQ-G-015 REQ-G-016 REQ-G-017 REQ-G-018
Data crawler	REQ-CR-001 REQ-CR-002 REQ-CR-003 REQ-CR-004 REQ-CR-005
Disinformation detection	REQ-DISD-001 REQ-DISD-002 REQ-DISD-003 REQ-DISD-004

	REQ-DISD-005 REQ-DISD-006
Hate speech detection	REQ-HSD-001 REQ-HSD-004 REQ-HSD-004 REQ-HSD-006 REQ-HSD-007
Deepfake detection	REQ-DFD-001 REQ-DFD-002 REQ-DFD-003 REQ-DFD-005
Disinformation analysis	REQ-DISA-001 REQ-DISA-002
Hate speech analysis	REQ-HSA-001 REQ-HSA-002 REQ-HSA-003
Deepfake analysis	REQ-DFA-001 REQ-DFA-002 REQ-DFA-003
Sentiment analysis	REQ-DISA-002 REQ-HSD-008
Behaviour analysis	REQ-DISA-001
Secure databases interconnection and information exchange	REQ-SE -001 REQ-SE -002 REQ-SE -003 REQ-SE -004 REQ-SE -005 REQ-SE -006
API Gateway	NFR-001 NFR-005 NFR-007
IAM	NFR-005
Persistent storage	NFR-003 NFR-004 NFR-005 NFR-007
Message broker	NFR-001 NFR-006 NFR-007
Orchestrator	NFR-001 NFR-003 NFR-004 NFR-006 NFR-007

4. SOTA Analysis Outcomes: Next Steps for Feature Development

The initial state-of-the-art (SOTA) analysis was conducted and documented in deliverable [1]. As we continue designing the AVALANCHE solution, we are evaluating selected SOTA tools and platforms, particularly those that align with its 3-tier architecture and could effectively support its implementation. For the detection and analysis services, we could integrate several AI-driven solutions. The AI monitoring systems from Full Fact AI, known for identifying false or misleading claims across diverse media, are one such example we could adapt for our own purposes. We might also consider Meedan's Sydny and ClassyCat for their methods of synthetic dataset generation and automated content labelling, which could be beneficial for training our classifiers for hate speech and misinformation. Logically's AI-powered content analysis and urgency scoring are approaches we could adopt for real-time misinformation scanning and prioritization, including their deepfake detection capabilities. The Factual's method of AI-driven credibility scoring, offering nuanced content assessments rather than simple true/false labels, is another concept we might explore for social media posts and deepfakes. To enhance our deepfake detection, we could integrate solutions from Sensity AI, focusing on identifying visual inconsistencies and manipulation techniques, or Sentinel AI's multi-layered deepfake detection, which provides confidence scoring.

For our analysis and crawling services, we are exploring advanced platforms that could be incorporated. Alto Analytics' use of AI and real-time data for risk management is informing our thinking about developing AI-powered analysis for detecting disinformation campaign patterns. We might consider Factiverse's live fact-checking for real-time verification of video and audio content. Identrics' WASPer and Information Disorder Intelligence, which detect AI-generated content and manipulative narratives, are capabilities we could integrate to enhance our content moderation and real-time monitoring. The methods employed by the Tilt Hate Speech Detection Model and Perspective API for identifying and scoring toxic content are also approaches we might consider for our hate speech detection services. To build a multi-layered defence against harmful content, we could utilize Hive's content tagging APIs, including their visual and text moderation, OCR, and audio moderation. Truly Media's collaborative verification dashboard and structured workflow capabilities are inspiring features that we might choose to implement for our own collaborative verification. Furthermore, the modular verification suite of InVID/WeVerify, with its multimedia forensic capabilities, text and sentiment tools, geolocation estimation, credibility scoring, and smart assistant, represents a comprehensive set of tools that we could integrate for content assessment. Finally, to ensure our system is adaptable and explainable, we could potentially adopt the principles of Vera.AI, particularly their focus on fairness, transparency, and explainability in AI models, multimodal and multilingual analysis, and the use of verified real-world data for training classifiers. We're also examining insights from AI4TRUST concerning sentence-level disinformation scoring, logical fallacy detection, cross-modal validation, and social network mapping to potentially enhance the analytical depth of the AVALANCHE solution.

5. Deployment & Infrastructure Planning

5.1 Preliminary Deployment Architecture Design

The deployment architecture has been designed to support the release of software using the CI/CD methodology.CI/CD, which stands for continuous integration and continuous deployment/delivery, is a software release approach that ensures reliability, speed, and high quality. The deployment of AVALANCHE components will be handled by the CI/CD stack which consists of the following tools:

- **NGINX**: A high-performance web server and reverse proxy used to serve static content, balance load, and secure traffic.
- **Keycloak**: An open-source identity and access management solution that provides single sign-on (SSO).
- **Let's Encrypt certbot**: A tool that automates the process of obtaining and renewing free SSL/TLS certificates from Let's Encrypt
- **Jenkins**: An open-source automation tool used for continuous integration and continuous delivery (CI/CD) of software.
- **SonarQube**: A platform for continuous inspection of code quality that detects bugs, vulnerabilities, and code smells in your codebase
- **Harbor**: A container image registry that secures, stores, and manages Docker images with role-based access control.
- **Portainer**: A lightweight management UI that allows you to easily manage Docker environments, containers, and clusters.

All infrastructure tools will be containerized using Docker technology for consistency, portability, and efficient resource utilization. The deployment servers will be ready to host the AVALANCHE components. The exact location of the infrastructure where the components will be deployed is not yet defined as it will depend on the limitations and preferences of partners.

To facilitate automation and monitoring of deployment servers, Jenkins and Portainer will have dedicated agents on each deployment server. These agents serve two functions:

- Enable Jenkins to execute deployment pipelines directly on target servers
- Allow Portainer to maintain visibility and management capabilities over all deployed containers and their host environments

The following diagram (Figure 19) helps visualize the design of the deployment approach and preliminary architecture:

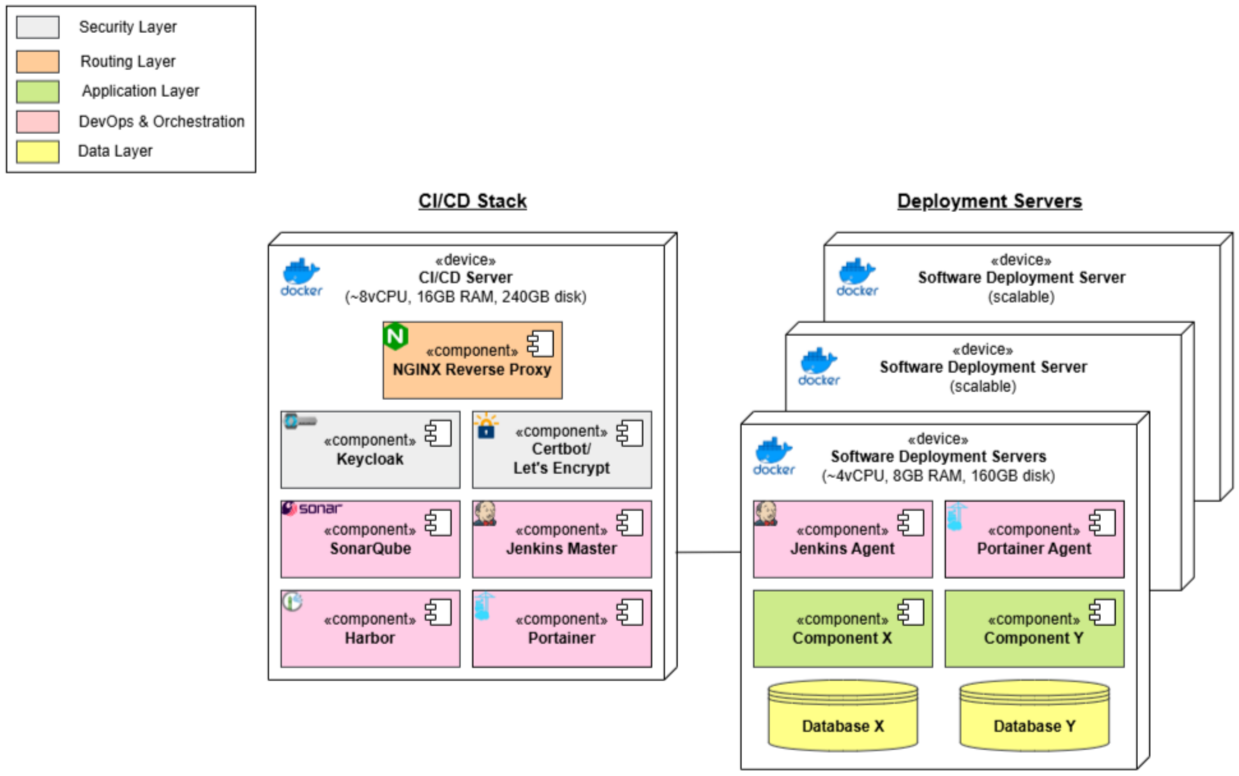

Security Layer
Routing Layer
Application Layer
DevOps & Orchestration
Data Layer
CI/CD Stack
Deployment Servers
«device» CI/CD Server (~8vCPU, 16GB RAM, 240GB disk)
«component» NGINX Reverse Proxy
«component» Keycloak
«component» Certbot/ Let's Encrypt
«component» SonarQube
«component» Jenkins Master
«component» Harbor
«component» Portainer
«device» Software Deployment Server (scalable)
«device» Software Deployment Server (scalable)
«device» Software Deployment Servers (~4vCPU, 8GB RAM, 160GB disk)
«component» Jenkins Agent
«component» Portainer Agent
«component» Component X
«component» Component Y
Database X
Database Y

Figure 19: High level deployment architecture of AVALANCHE

The diagram gives an initial view of how deploying software will be handled in AVALANCHE. Hardware requirements for servers will be different depending on the software being deployed, with the ability to upscale in the future whenever needed. As the AVALANCHE project evolves and more information is gathered, the deployment architecture design will be updated to show exactly which servers will be hosting each AVALANCHE component. All components are categorized into different layers using different colours to better help visualize and understand the purpose they serve and how they fit into the architecture design.

5.2 Initial CI/CD & DevOps Strategy

The CI/CD stack will be used to create pipelines that will build, test, and deploy AVALANCHE components. The stack will be accessible through Keycloak, which will function as the Identity Access Management and SSO (Single Sign On) provider that the partners can use to log in to all the other tools in the CI/CD stack. Keycloak will have GitHub as an Identity provider, so partners will log in using their GitHub credentials. Jenkins will be the main component in the CI/CD stack. Partners will use Jenkins to create pipelines for deploying their components. The pipelines created can integrate with other tools in the stack such as Harbor or SonarQube. Stages in pipelines can include a stage that pushes docker images to Harbor, for example, or a stage that scans source code with SonarQube, which the partners can access to see the result of their scanned code. Portainer can then be used to monitor the state of the deployed containers. This will be the DevOps strategy used in AVALANCHE.

The strategy includes two main workflows (Figure 20), one for pushing code, and one for pushing docker images. Either one of the scenarios will trigger a Jenkins pipeline that starts the entire CI/CD cycle and ends with a fully tested and deployed component.

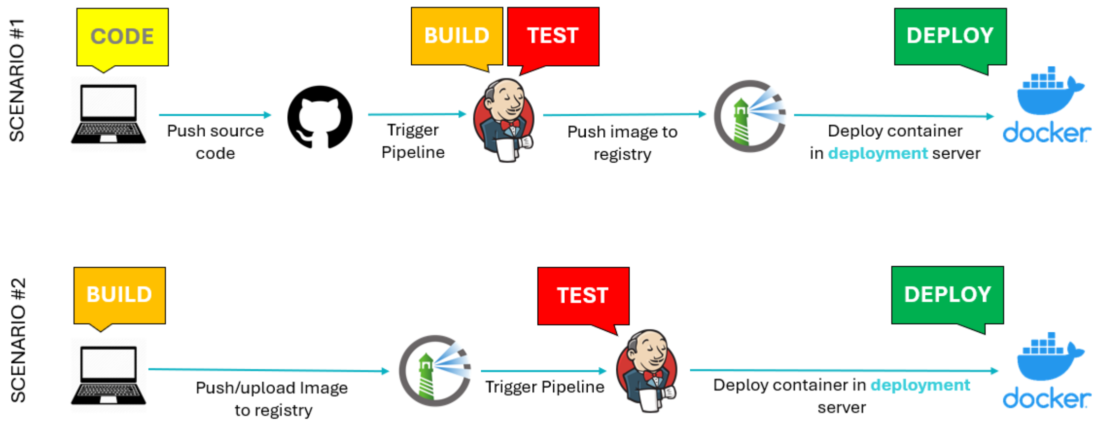


Figure 20: CI/CD deployment workflows

In the first scenario, code gets pushed to the GitHub repository that is in the AVALANCHE GitHub Organization. This will trigger the Jenkins pipeline that will check out the code, build the software, run any tests and source code scans that are set up, then proceed to push the built docker image to the Harbor registry before using it to deploy the containerized component.

In the second scenario, the built docker image gets tagged and pushed directly to the Harbor registry, where there is a webhook that will trigger a Jenkins pipeline that will test and deploy it in the same way as in the first scenario.

Other scenarios/pipelines can be created to meet the needs of deployments where other stages need to be included by a partner.

6. Technical KPIs

This section lays the basis for a robust monitoring framework by defining the critical Technical Key Performance Indicators (KPIs) relevant to the system's core technology blocks. By providing insights, stakeholders gain a clear understanding of the underlying technical performance, ensuring that the system operates optimally and in alignment with its strategic objectives. Tracking these KPIs is critical for staying ahead of problems. They act as an early warning system, helping teams find bottlenecks and slowdowns and fix them before they cause major disruptions. By setting clear performance goals, the team can focus on making targeted improvements.

It is important to note that these initial metrics represent the insights formulated based on the current understanding of the system's architecture and requirements. As the system progresses through the development and testing phases, the development teams will adjust and adopt the technical KPIs accordingly. This iterative process will allow for the refinement of these metrics and the establishment of more ambitious targets, ensuring the monitoring framework remains relevant and effective throughout the project lifecycle.

Web crawling & disinformation detection

- Number (#) of distinct information sources to collect data from: **>5000**
- Documents (#) collected from online sources per day: **>9000**
- Key semantic entities (including slang and special terms) correctly identified in textual sources: **>95% of true positives**
- Percentage (%) of targeted domains successfully crawled: **>90%** if the targeted domain is fully functional, i.e. “up”
- Percentage (%) of correct disinformation classifications: **>85%**
- Percentage (%) of legitimate articles incorrectly flagged as disinformation: **<10%**
- Fact-checking verdict: within **15 seconds** in **90% of requests**

Hate speech detection & sentiment analysis services

For hate detection and sentiment analysis services, a preliminary set of technical KPIs has been identified based on the current stage of system design. These KPIs are expected to be refined and updated in future deliverables as the development of these services progresses.

Provisional Technical Targets:

- Accuracy, Precision, Recall, and F1-score: approximately 70%

These metrics were chosen to provide a more balanced evaluation of model performance, particularly in scenarios involving class imbalance. The combination of these indicators offers a more reliable assessment of the model’s ability to generalize effectively.

Important Considerations:

- Most existing benchmarks are based on English-language datasets.
- For low-resource languages, the expected performance will vary depending on the specific approach adopted and the quality and availability of training data.

- More accurate technical targets will be established once the modelling approach and implementation strategy are fully defined.

Deepfake & behaviour analysis services

During the initial stage of AVALANCHE, there was a need to refactor the use cases based on the needs of the end-user of the project. For that reason, new technological components were introduced based on the needs of the end-user, such as the Deep Fake detection. Thus, certain components are currently at a lower technological readiness than expected and are currently being developed. Hence the provisional targets of the introduced components are:

- Accuracy of the deep fake detection module > 70%
- Number of topics across which behaviour analysis will take place >= 3
- Identification of potential origin of spread based on crawled timestamp
- Identification of potential spread tree/graph based on crawled timestamp and connecting URLs

7. Conclusions

This deliverable has outlined the architectural foundation of the AVALANCHE platform, detailing its system structure, core components, and the enabling technologies that collectively support its mission to detect, analyse, and mitigate harmful online content. Developed as of M09, this initial version of the system architecture establishes a robust baseline for the iterative development phases ahead, while maintaining flexibility for further refinement as technical requirements evolve.

The AVALANCHE architecture is structured following a three-tier approach, which promotes separation of concerns and enables scalability, maintainability, and adaptability. The Presentation Layer is dedicated to user interaction through the AVALANCHE Web UI. This layer facilitates user interaction and offers a set of advanced features, including interactive visualization tools, personalized analysis, and custom reporting. Furthermore, Secure access is ensured through an integrated Identity and Access Management (IAM) system. All user requests are processed via an API Gateway, which is responsible for authentication and appropriate routing.

At the core of the architecture lies the Application Layer, which hosts the platform's intelligence and coordination logic. Central to this is the Orchestrator, responsible for managing complex workflows across services, ensuring that detection, analysis, and crawling tasks are seamlessly coordinated to fulfil user queries. This orchestration is further enhanced by the API Gateway, which not only manages traffic but integrates tightly with backend services.

The Detection Services and Analysis Services represent the functional heart of the system. They enable real-time processing of user-submitted content, supporting use cases such as disinformation detection, hate speech analysis, deepfake identification, sentiment interpretation, and behavioural insight generation. Complementary Crawling Services autonomously retrieve relevant data from online sources, storing it in the Data Layer for downstream processing.

The Data Layer itself is designed for both persistence and flexibility, supporting a variety of data storage technologies including PostgreSQL, MongoDB, Elasticsearch, and MinIO. The integration of CKAN enhances data discoverability and accessibility, enabling efficient uploading, viewing, and sharing of structured and unstructured data. A dedicated Message Broker (based on Apache Kafka and the in-house Data Fusion Bus) ensures asynchronous communication across services, enhancing decoupling and resilience.

A critical and distinguishing feature of the AVALANCHE architecture is the Security Layer, introduced to uphold the platform’s commitment to data protection, integrity, and trust. This includes comprehensive Identity and Access Management through Keycloak, as well as Privacy and Security Technologies that implement encryption, anonymization, and secure communication protocols to safeguard user data and maintain regulatory compliance.

This architecture is not only a blueprint for development but also serves as the integration framework for assembling and testing the full system. It allows for modular development, where services can be independently built, tested, and deployed, ensuring rapid iteration and easier maintenance. As the project progresses, the architecture will guide both the technical implementation and the functional integration of all AVALANCHE modules into a coherent, operational platform.

Moving forward, the focus will shift to the detailed design and implementation of each module, followed by their integration into the broader ecosystem. This will culminate in the realization of a secure, intelligent, and scalable platform capable of responding to the complex challenges of digital misinformation, harmful content, and the evolving landscape of online threats.

References

[1] D3.1, "State-of-Play, End User Requirements and KPIs".

[2] K. Bittner and I. Spence., Use Case Modeling, 2002.

[3] S. Dumitrescu, A.-M. Avram and S. Pyysalo, "The birth of Romanian BERT," in *MNLP 2020*, 2020.

[4] D. C. Neagu, "Bertweetro: Pre-Trained Language Models for Romanian Social Media Content," Studia Universitatis Babes-Bolyai Oeconomica, 2025.

[5] F. M. Plaza-del-Arco, S. Halat, S. Padó and R. Klinger, "Multi-Task Learning with Sentiment, Emotion, and Target Detection to Recognize Hate Speech and Offensive Language," in *CEUR Workshop*, 2022.

[6] N. Melania and M. Dascalu, "Natural Language Processing Tools for Romanian – Going Beyond a Low-Resource," in *Interaction Design and Architecture(s)*, 2024.

[7] P. Röttger, D. Nozza, F. Bianchi and D. Hovy, "Data-Efficient Strategies for Expanding Hate Speech Detection into Under-Resourced Languages," 2023.

[8] X. Sun, X. Li, S. Zhang, S. Wang, F. Wu, J. Li, T. Zhang and G. Wang, "Sentiment Analysis through LLM Negotiations," 2023.

This work is part of the AVALANCHE project. AVALANCHE has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement no. 101168393

Appendix A

This Appendix presents, in tabular form (Table 14), the user requirements provided by the SPP end-user concerning the crawling services, which were not included in D3.1. That deliverable focused solely on user requirements related to the defined use cases, disinformation, hate speech, deepfake—and the evidence and secure exchange scenario. Additionally, Table 15 summarizes further input from the SPP partner regarding user needs and requests to be addressed in the technical specifications of the AVALANCHE solution.

Table 14: Crawling / Collecting data: User requirements provided by SPP.

ID	General requirement	Detailed requirement
1	Must be able to collect data from Social Media Platforms	integrate with Facebook
2		integrate with Twitter
3		integrate with YouTube
4		integrate with Telegram/X
5		integrate with Instagram
6		integrate with WhatsApp
7		integrate with Reddit
8		integrate with TikTok
9		integrate with Discord
10		may manually insert targeted accounts when defining a task
11		may upload a file with targeted accounts when defining a task
12		will make suggestions of accounts with the aid of Machine Learning based on the ones inputted by the analyst and the ones from previous tasks
13		will identify typos/errors from account format and raise the awareness of the analysts
14		rules for excluding specific users from the crawling process can be defined within the blacklist
15		rules for excluding specific groups/communities from the crawling process can be defined within the blacklist
16		rules for excluding specific formats from the crawling process can be defined within the blacklist
17	Must be able to collect data from the Web	can extract data from Surface Web
18		can extract data from Deep Web
19		can extract data from Dark Web
20		may manually insert URLs when defining a task
21		may upload a file with URLs when defining a task
22		will make suggestions of URLs with the aid of Machine Learning based on the ones inputted by the analyst and the ones from previous tasks
23		will identify typos/errors from URLs format and raise the awareness of the analysts
24		rules for excluding specific websites from the crawling process can be defined within the blacklist

This work is part of the AVALANCHE project. AVALANCHE has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement no. 101168393

25		rules for excluding specific formats from the crawling process can be defined within the blacklist
26	Must mark an account of interest for the analyst and extract relevant information	compile a list will all the users present in the targeted group
27		extract information about the friends of the targeted user
28		extract information about the followers of the targeted user
29		extract text information made by the targeted user
30		extract images posted by the targeted user
31		extract videos posted by the targeted user
32		extract live transmissions made by the targeted user
33		extract audio posted by the targeted user
34	Must mark an account of group/community for the analyst and extract relevant information	compile a list will all the users present in the targeted group
35		compile a list with the most active users within the targeted group
36		compile a list with the most shared/liked posts within the group
37		extract text information made within the group
38		extract images posted within the group
39		extract videos posted within the group
40		extract audio posted within the group
41	Will define and use keywords/key phrases during the crawling process	can manually input unlimited number of keywords/key phrases
42		can upload a txt file with keywords/key phrases
43		can receive suggestions of keywords based on the input ones (synonyms generated automatically by the platform and some of the most used keywords in previous queries)
44		can receive suggestions on typos for input keywords (spelling errors)
45		will support special characters (support for Romanian)
46		will support logical operators in order to create complex queries (AND, OR, NOT)
47		will support partial words queries
48		will support exact words queries
49	Will support scheduling and customization of searches/queries/tasks	a task can be run multiple times (without recreating the search from scratch)
50		a task ran multiple times will retrieve only new information and prevent text duplication
51		a previous task can be edited and created as a new task (do not overwrites the previous task, but implies the creation of the new task)
52		multimedia content will be fetched and displayed in line and formatted in a similar manner as in the original location (prevent loss of context)
53		the analyst can select what type of information to retrieve (text, image, audio, video)
54		a task can be configured to run on specific days of the week/month in a repetitive manner for a given period of time
55		a task can be configured to run at specific hours a day (once or multiple times a day) for a given period of time
56		A task can be configured to run at specific dates/hours, but not in a recurrent method
57		A task can be configured to run for a specific period of time (e.g. X minutes or Y hours or Z days or until a condition is met)

58		a task can be run manually by the analyst, independent from the planning, and without interfering with the initial schedule
59		a task can be paused by the analyst and results analysed and also the analyst can resume the task, and the crawler will continue from where it remained
60	Tasks definition and information retrieve will respect the need-to-know principle	a specific task can be created, with a unique name to ensure that duplicates are prevented
61		A list of users will be defined to ensure that the need-to-know principle is ensured for the specific task
62		task specific information (e.g. description, period of time, coordinating officer, etc.) can be defined as metadata associated to it
63		the platforms enable the analyst that created the task to select what users are allowed to edit/view information related to the crawling task
64		the platforms enable the analyst that created the task to modify users that access or revoke access at any time
65		logs will be maintained to keep track of who and when they had access
66	Information export within the platform	The users can pick from the UI what columns/data is of interest to be exported
67		Data can be exported in CSV format
68		Data can be exported in JSON format

Table 15: Technical requirements provided by SPP

ID	General requirement	Detailed requirement
1	Use state of the art architecture	Use a three-tier client-server application (web)
2		Ensure high availability, with under 0.01% downtime
3		Ensure redundancy through load balancing to ensure business continuity
4		Ensure an integrated architecture with a single Graphical User Interface to access all features from a single point
5		The platform will use a time server or equivalent to provide consistent timestamps within all the tools/VMs/dockers
6		Remote/physical support will be offered for the on-premises deployment by technical partners
7		A guideline for on-premises deployment will be provided by technical partners
8		License for all software pre-requisites will be provided by the technical partners
9		The platform will work adequately with the HW resources provided by LEAs for on-premises deployments
10		All algorithms, mechanisms or standards used for ensuring security for the platform must not be deprecated
11		Each feature/module needs to work also standalone to ensure that in case of failure, all remaining functionalities continue to work, and the platform does not crash

12	Platform characteristics	Ensure the solution is accessible and fully operational with no third-party application installed on the client, with no settings to be configured on the OS/browser/application
13		The solution needs to have the same performance regardless the number of concomitant users, regardless of the data stored and without the need to upgrade HW or SW specifications
14		All functionalities are accessible through the browser
15		The platform is optimized to work on Google Chrome and Mozilla Firefox
16		The platform must offer API support for integration with external applications
17		APIs must be well documented to simplify a possible integration with other systems at LEA premises
18		The platform must be functional and provide the same performance regardless of the operating system on the client and/or on the server
19		The platform must optimize performance by caching data in forms and prevent data loss in case of errors
20		The platform must optimize performance by minimizing response payload sizes with gzip compression.
21		The platform must optimize performance by using query optimizations and indexing in the database.
22		The platform must optimize performance by minimizing render blocking and optimizing images
23	Platform security	To ensure compatibility with LEA procedures, the platform must ensure security by design approach and use secure communications between the Server and Client like SSL/TLS, last stable version available
24		The platform must use HTTPS communications for all operations
25		To ensure compatibility with LEA procedures, the platform must ensure security by design approach and prevent SQL injections by design that could be performed through XSS and other types of attacks
26		To ensure compatibility with LEA procedures, the platform must ensure security by design approach and prevent data inconsistencies by using a transactional approach like everything or nothing through rollback operations in case of errors/crashes
27		The solution will enable the administrator to configure from the web browser through the interface the rights for each group of users to different resources (read, edit, delete, no access is by default)
28		The solution will enable administrator to configure directly from the interface an unlimited number of groups without the need to modify the back end (coding or database)
29		The solution will enable the administrator to configure from the web browser through the interface the rights for each user to different resources (read, edit, delete, no access is by default)
30		The solution will enable the administrator to configure directly from the interface an unlimited number of users without the need to modify the back end (coding or database)

31		The platform will ensure data privacy by encrypting sensitive data in the database to prevent data disclosure to ensure the need-to-know principle
32		The platform will ensure comprehensive logging capabilities (all actions are audited, all edits are audited to save the old and new values, along to the user that made the change, the timestamp (not the query, but individual data)
33		Use only the latest stable version for each dependency and download it locally (avoid using online resources)
34		Use only reliable/trusted/well-known/signed sources for external dependencies (libraries, dll, nugget, etc.)
35		Avoid Exposing Sensitive Data: Do not include secrets or API keys in your front-end code. Use environment variables and keep sensitive logic on the backend.
36		Content Security Policy (CSP): Implement a strict CSP to block unauthorized scripts or content from running on your site.
37		Avoid Open Redirects: Validate URLs before redirecting users to prevent phishing attacks.
38		Information is never deleted from the database, but only hidden in the UI
39		Data is manipulated only from the platform UI
40		No operations directly on the database are permitted to any category of users
41		All content from the database is encrypted to ensure data privacy
42		Data can be back-up directly from the UI
43		Periodical back-ups can be programmed by the administrators to limit the possibility to lose data
44		Data can be restored from the UI at any point of time using an existing back-up
45		Two factor authentication will be integrated and can be activated/deactivated from the UI
46		Usernames are unique within the platform in order to keep traceability
47		Users are logged off automatically after a predefined period of inactivity
48		Administrator role has the objective to manage users and has no access to data or tools in the UI
49		Administrators can create roles and assign permission dynamically in a granular manner (tool level/data access level)
50		Security features must not affect the performance of the system (e.g. slower processing, higher delay, lagging, etc.)
51		All algorithms, mechanisms or standards used for ensuring security for the platform must follow current NIST guidelines or equivalent
52		All activities within the platform will be audited (e.g. login, read, write, etc.)
53		Filters will be made to narrow down to a limited dataset of interest for the administrator directly in the UI

54		Audit data cannot be manipulated under no circumstance by any category of users
55		Access to the audit will be made to a limited number of users, with specific rights
56		Audit data format must follow international standards
57		The platform is not susceptible to top 10 OWASP vulnerabilities
58		Password complexity can be set by the administrator (e.g. include Uppercase, lowercase, digit, special character)
59		Password length can be set (e.g. minimum of 9 characters)
60		Passwords can be reset/changed by the user without the intervention of an administrator
61		Password expiration after N days/enforce password change can be set by the administrator
62		Users are blocked automatically if the password has been entered repeatedly incorrect
63		Low complexity passwords (e.g. password, qwerty, includes the name of the person, 12345, etc.) /password reusage (last 10 passwords) are prohibited
64	Platform UI/UX	Organise information hierarchically, through menus, submenus, contextual menus and nomenclatures/classes
65		Use the same design, theme, characteristics (colour, shape, font, etc.) for all modules to ensure consistency
66		Ensure a friendly and easy approach to navigate through the UI, without no technical knowledge and with a minimum number of operations/clicks
67		Ensure contextual help to find relevant information on a feature (prevent the use of classical long manuals and use hover features to give hints)
68		Optimize the UI by hiding useless inputs or fields that a specific user does not have access to (clean the UI and display minimal information based on the need-to-know principle)
69		Optimize the UI and prevent loading the entire page for each operation and perform targeted submission/fetch through JavaScript or similar
70		The platform must have a Responsive Design and use CSS media queries to adapt the layout to different screen sizes.
71		Implement Fast Loading Times (Target a Time to Interactive (TTI) of under 3 seconds)
72		The platform will natively open images directly from the UI, without downloading the file locally
73		The platform will natively open documents (word, pdf) directly from the UI, without downloading the file locally
74		The platform will natively open video directly from the UI, without downloading the file locally
75		The platform will natively open audio directly from the UI, without downloading the file locally

76		Implement Feedback and Loading States: • Add loading spinners or skeleton screens to show progress for API calls or dynamic content. • Provide clear feedback for user actions (e.g., buttons with loading states).
77		Implement Intuitive Navigation: • Use breadcrumbs, clear headers, and sticky navigation for better orientation. • Ensure all clickable elements are easy to identify and tap on
78		Manage Error Handling: Gracefully handle errors with friendly messages (e.g., "Something went wrong. Please try again."). Use fallback UIs when the application cannot load (e.g., offline mode or network errors).
79		Write Maintainable and Scalable Code: Organize components logically and avoid duplicating code.
80	API best practices	Follow REST Principles Statelessness: Each API request should contain all the necessary information for the server to process it (e.g., authentication tokens, query parameters) Resource-based URLs: Structure URLs around nouns, not verbs. For example (Good: /users/123/posts, Bad: /getUserPosts) HTTP Methods: • GET: Retrieve resources. • POST: Create new resources • PUT/PATCH: Update existing resources. • DELETE: Remove resources.
81		Use Consistent Naming Conventions: Use lowercase and hyphens in URLs (/users/{user-id} instead of /Users/{userId}). Use plural nouns for collections (e.g., /products instead of /product).
82		Use appropriate Versioning mechanism Include a version number in the URL or header. URL versioning: /v1/users Header versioning: Accept: application/vnd.company.v1+json Clearly deprecate old versions with sufficient notice.
83		Use the appropriate HTTP status codes for every response: 200 OK: Success. 201 Created: Resource created successfully. 204 No Content: Successful operation with no response body. 400 Bad Request: Client error (e.g., invalid parameters). 401 Unauthorized: Authentication is required. 403 Forbidden: Access is denied. 404 Not Found: Resource does not exist. 500 Internal Server Error: Unexpected server-side error.
84		Use specific Design for Pagination and Filtering For large datasets, provide pagination parameters such as: • GET /users?page=2&limit=20 • Use headers for metadata like total count: X-Total-Count.

		• Allow filtering, sorting, and searching: /users?role=admin&sort=created_at&search=John
85		Provide meaningful error messages in a consistent structure
86		Use Rate Limiting and Throttling Prevent abuse with rate limiting: Example: Allow 1000 requests per hour per user. Send rate limit headers: X-RateLimit-Limit: Total request limit. X-RateLimit-Remaining: Remaining requests. X-RateLimit-Reset: Time until reset.
87		Allow cross-origin requests when necessary but restrict domains to improve security. Use proper headers: • Access-Control-Allow-Origin • Access-Control-Allow-Methods • Access-Control-Allow-Headers
88	AI Algorithms	Entity extraction algorithm
89		Binary classification from text algorithm for hate speech (hate or non-hate)
90		Speech to text algorithm for audio/video
91		Multilingual text translation (en - ro, ro - en, arabic - ro, ro - arabic, rus - ro, ro - rus, ukr - ro, ro - ukr, hun - ro, ro - hun, etc.)
92		RSS feed data processing algorithm (includes extraction, filtering, parsing, organising, cleaning, standardized storage)
93		TV/radio data processing algorithm (includes extraction, filtering, parsing, organising, cleaning, standardized storage)
94		Object detection algorithm (classes to be agreed upon in a later stage)
95		Sentiment analysis algorithm with 3 scales (positive, negative, neutral)
96		Image classification algorithm
97		Text identification and transcription through OCR from image or video
98		Person identification algorithm
99		Symbol identification algorithm
100	AI - text processing and standardization	Develop a dictionary for hate speech with relevant terms and slangs from different languages (especially Romanian, English, Russian, Arabic) - automatic population and manual edit
101		Develop a dictionary for aliases for dignitaries with relevant terms from different languages (especially Romanian, English, Russian, Arabic) - automatic population and manual edit
102		Develop a dictionary for hate symbols for dignitaries with relevant images from different contexts (to be agreed upon on a later stage)
103		Classify hate speech in the minimum classes: - ethnic/cultural/religious - incite to violence - instigation / mockery / banter

Appendix B

Appendix B presents the input provided by SPP regarding operational steps and use case tables related to the defined scenarios. This input played a valuable role in shaping the definition of AVALANCHE actors and the development of user journey scenarios, particularly when aligned with the feasibility rankings assigned to each functional requirement (FR) of the AVALANCHE solution. **It is important to note that the input presented here does not consider feasibility constraints but rather reflects the full scope of user needs as identified and documented during the user requirements collection process (as part of the D3.1 activities).**

Disinformation Case

Avalanche platform features**:**

- Search open web and social platforms using advanced keyword detection;
- Analyse sentiment (positive, negative, neutral) and severity score (mild, moderate and high) from the comments section and timeline visualization;
- Propagation map and amplifier identification;
- Identify „patient" zero (the users` Telegram account);
- Automatic transcription and threat classification of the original video content;
- View count, like count, and share tracking;
- Export of structured reports in PDF or JSON formats, labelled „*Incident Report*", that can further on be used by the specific stakeholders (the protected dignitary; the security team; other advisors);
- Alert system for high-risk or rapidly spreading content.

Table 16: Disinformation use case presentation

Use Case ID:	**Disinformation_01**
Name:	Detecting disinformation using Avalanche, concerning one of our protected dignitaries
Timeframe:	Continuous monitoring activity within a predefined timeframe
Primary Actor(s):	• Administrators (Avalanche technical maintenance team); • Intelligence Operational Team (OSINT extraction and integration); • Intelligence Analyst Team (risk assessment); • High-ranking officers (decision makers); • Avalanche Platform (connected to the Internet; does the analysis and gives alerts).
Main Goal:	• Identify disinformation targeting public figures; • Perform sentiment analysis; • Identify patient zero (source/origin of spread); • Generate propagation map; • Perform text transcript from video (speech to text capability); • Perform OCR from images / video frames (map text from images); • Perform text translation from foreign languages in Romanian; • View count, like count, and share tracking; • Export content in report format (pdf and JSON) – for further analysis; • Generate alerts for specified thresholds.

Overview:	After starting the monitoring of the suspicious source, the Avalanche platform will allow: • Collection in real time and analyses of the social media, blogs, forums, onions and websites; • Integration of static analysis like the prediction of links between suspicious users and early detection of groups that perform disinformation; • Cross-checks with other sources to detect disinformation; • Create alerts when disinformation is being used; • Proactive online threat assessment (Surface/Deep/ Dark Web and social media).
Precondition(s):	• Datasets (including text posts, posts with pictures, posts with video sequences, etc.) – data populated online in the platforms / social media accounts to simulate real operation (no static data from files or from database will be considered); • Equipment needed for demonstration during the used case – Computers / Avalanche Platform / Access to the internet; • Access to the platform (users created, roles defined, need-to-know access to specific activities).
Main Input(s):	• Social media networks (Facebook, Twitter, YouTube, TikTok, etc.); • Surface/deep/dark web; • Keyword list (manual input or file); • URL list (manual input or file); • Credentials.
Main Output(s):	• Data regarding disinformation messages posted by individuals / groups of people on social networks / web addressed to the high officials; • List of suspicious persons/groups/communities discovered and links between entities involved; • Multimedia content with analysis results (bounding boxes, transcripts, OCR, etc.); • Security gathering report / threat assessment report.
Success Post Condition(s):	• Disinformation content is accurately detected and flagged from web; • Disinformation content is accurately detected and flagged from social media; • Patient zero is detected; • Propagation map is constructed; • Statistical data is collected to offer insight on dissemination level of information; • Reports are generated.
Failed Post Condition(s):	• Disinformation content is not accurately detected; • Disinformation content is false positively detected; • Information is not collected from social media; • Information is not collected from surface web; • Information is not collected from dark web; • Patient zero is not detected or falsely detected; • Propagation map is not constructed or is not correctly generated; • Statistical data is not collected or is inaccurately reported; • Reports are not generated with accurate data.

Operational steps:

Step 1: Access the platform

Firstly, the analyst will launch the specialized tool designed to detect disinformation targeting the prime minister, namely the Avalanche Platform, by providing the credentials necessary for authentication. Each analyst has its own credentials, provided by the administrator in advance. Otherwise, the administrator generates the account and provides credentials for access (one time password that is changed at the first logon process by the user for privacy concerns).

Step 2: Define the subject

We will create a new operation for the specific task that we are performing at that given time (e.g., “Pavel Andrei - faliment”) in the platform’s dashboard, to centralize all monitoring activities related to our protected prime minister, at that exact time. Each operation is uniquely identified and named (no duplicates are permitted, and access to the operation is based on the need-to-know principle – by default nobody has access until explicit access is provided to the users). Granular access is relevant for sensible tasks like OSINT operations.

Step 3: Set up keyword tracking

To capture relevant content, we will configure a set of **specific, high-risk keywords and phrases** that may be used in hate speech or threats. Based on the case we are investigating, we include generic and context-specific terms but also misspelled or disguised hate content, like:

- "Pavel Andrei";
- "Prim ministru" (prime minister);
- "fură țara" ("steals the country");
- "concedieri masive" ("massive dismissal");
- "faliment" ("bankruptcy");
- "economia în picaj” ("economy is dropping");
- “șomer” ("unemployed").

Step 4: Search across open web sources

We begin data gathering from Romania located open web sources such as https://hotnews.ro/; https://www.g4media.ro/; https://www.cotidianul.ro/; https://adevarul.ro/; https://defapt.ro/. The list can be manually provided by the analyst by inputting it manually or by uploading a file (txt or csv) with a list of websites (100-10.000 URLs).

Step 5: Detect and flag content

The platform returned a hit. An article published on March 15 by the platform https://factual.ro, detected false alarming news about bankruptcy of Romania, the desire of the Prime Minister to fire people and shut down the economy. Moreover, on Telegram different users started sharing text, video and audio messages claiming that people will get fired and the economy will fail. The messages in question — which, despite breaching community standards, have not been taken down — have accumulated nearly 200,000 views and approximately 50,000 likes. Even if the crawl began in the open web, it correlated with social media to determine the spread (viralisation).

In the same time, some foreign posts were detected from some refugees from Ukraine were detected within the relevant content and with the translation module, it was translated from Ukrainian to Romanian to determine if results are relevant according to the given keywords. The information is automatically flagged as **Level 2 Priority (Disinformation** and **Incitement to Protest)**, an alert is being issued, and the information is further analysed by the analyst in charge of assessing threats.

Step 6: Multimedia analysis

Based on the textual hit, a number of multimedia content was also discovered in the same sources. Multimedia has different formats: video, images, and audio. We start performing multimedia analysis like:

- **Visual analysis through AI**: detection of protests in images that incite to violence and overthrow the government;
- **OCR extraction**: certain images contain text captions that incite protests, that will be further distributed to the text analysis for flagging. A similar approach is done also to video (that is being split into frames);
- **Speech to text analysis**: for video and audio files detected, transcripts are generated through dedicated modules, to determine topics, speakers and analyse the presence of disinformation and incitement to protest.

Step 7: Analyse distribution and reach

Using the platform's analytics tools, we identify the initial comments posted on Telegram and start examining:

- **Engagement metrics**: shares, comments, reposts;
- **Amplifiers**: Who shared it (e.g., known political agitators or public figures);
- **Audience demographics;**
- **Virality curve**: How fast it spreads over time.

The platform identifies that the video was **reshared 4000 times** on Telegram, **cross-posted** to Facebook groups, and also on Reedit. The origin of spread is detected (***patient zero***), with clear explanation of the results through xAI to particular user **agitatorul01**.

Step 8: Generate an incident report

You export a detailed report that includes: The original content (video, captions, timestamp); Screenshots and transcript of the disinformation message; Spread analysis (where, who, how many); Sentiment and severity score; Historical context (if the user is a repeat offender); Recommended action: flag to Telegram, notify law enforcement liaison, etc. The format of the report is based on a template that can be edited later without coding in order to adapt it to the needs required by high-ranking officers.

Step 9: Continue monitoring

You enable continue **tracking** on open web sources for other materials posted by the specific user; imitation accounts, threats in the comment section (both on open web sources and social media platforms).

Step 10: Platform monitoring

The administrator monitors that the platform is working adequately, that it performed a back-up of the collected data in order to prevent data loss in case of corruption, and that logs are being stored with all user activity within the platform. In the event of an audit, all of these are relevant to ensure that all procedures are taken accordingly. Logs will contain detailed information about who (user), what (resources accessed/requests), when (timestamp), and where (server/machine/location).

Hate speech case

Avalanche platform features**:**

- Search open web and social platforms using advanced keyword detection;
- Analyse sentiment (positive, negative, neutral) and severity score (mild, moderate and high) from the comments section and timeline visualization;
- Propagation map and amplifier identification;
- Identify „patient” zero (the users` TikTok account);
- Automatic transcription and threat classification of the original video content;
- View count, like count, and share tracking;
- Export of structured reports in PDF or JSON formats, labelled „*Incident Report*”, that can further on be used by the specific stakeholders (the protected dignitary; the security team; other advisors);
- Alert system for high-risk or rapidly spreading content.

Table 17: Hate speech use case presentation

Use Case ID:	**Hatespeech_01**
Name:	Detecting hate speech using Avalanche, concerning one of our protected dignitaries
Timeframe:	Continuous monitoring activity within a predefined timeframe
Primary Actor(s):	• Administrators (Avalanche technical maintenance team); • Intelligence Operational Team (OSINT extraction and integration); • Intelligence Analyst Team (risk assessment); • High-ranking officers (decision makers); • Avalanche Platform (connected to the Internet; does the analysis and gives alerts).

Main Goal:	• Identify hate speech and incitement content targeting public figures; • Perform sentiment analysis; • Identify patient zero (source/origin of spread); • Generate propagation map; • Perform text transcript from video (speech to text capability); • Perform OCR from images / video frames (map text from images); • Perform text translation from foreign languages in Romanian; • View count, like count, and share tracking; • Export content in report format (pdf and JSON) – for further analysis; • Generate alerts for specified thresholds.
Overview:	After starting the monitoring of the suspicious source, the Avalanche platform will allow: • Collection in real time and analyses of the social media, blogs, forums, onion and websites; • Integration of static analysis like the prediction of links between suspicious users and early detection of groups that perform hate speech; • Create alerts when hate speech is being used; • Proactive online threat assessment (Surface/Deep/ Dark Web and Social Media).
Precondition(s):	• Datasets (including text posts, posts with pictures, posts with video sequences, etc.) – data populated online in the platforms / social media accounts to simulate real operation (no static data from files or from database will be considered); • Equipment needed for demonstration during the used case – Computers / Avalanche Platform / Access to the internet; • Access to the platform (users created, roles defined, need-to-know access to specific activities).
Main Input(s):	• Social media networks (Facebook, Twitter, YouTube, TikTok, etc.); • Surface/deep/dark web; • Keyword list (manual input or file); • URL list (manual input or file); • Credentials.
Main Output(s):	• Data regarding threat/hate speech messages posted by individuals / groups of people on social networks / web addressed to the high officials; • List of suspicious persons/groups/communities discovered and links between entities involved; • Multimedia content with analysis results (bounding boxes, transcripts, OCR, etc.); • Security gathering report / threat assessment report.
Success Post Condition(s):	• Hate speech content is accurately detected and flagged from web; • Hate speech content is accurately detected and flagged from social media; • Patient zero is detected; • Propagation map is constructed; • Statistical data is collected to offer insight on dissemination level of information; • Reports are generated.

Failed Post Condition(s):	• Hate speech content is not accurately detected; • Hate speech content is false positively detected; • Information is not collected from social media; • Information is not collected from surface web; • Information is not collected from dark web; • Patient zero is not detected or falsely detected; • Propagation map is not constructed or is not correctly generated; • Statistical data is not collected or is inaccurately reported; • Reports are not generated with accurate data.

Operational steps:

Step 1: Access the platform

Firstly, the analyst will launch the specialized tool designed to detect threats or hate speech directed at our protected president, namely the Avalanche Platform, by providing the credentials necessary for the authentication. Each analyst has its own credentials, provided by the administrator in advance. Otherwise, the administrator generates the account and provides credentials for access (one time password that is changed at the first logon process by the user for privacy concerns).

Step 2: Define the subject

We will create a new operation for the specific task that we are performing at that given time (e.g., “Andrei Toma - amenințări”) in the platform’s dashboard, in order to centralize all monitoring activities related to our protected president, at that exact time. Each operation is uniquely identified and named (no duplicates are permitted and access to the operation is based on the need-to-know principle – by default nobody has access until explicit access is provided to the users). Granular access is relevant for sensible tasks like OSINT operations.

Step 3: Set up keyword tracking

To capture relevant content, we will configure a set of **specific, high-risk keywords and phrases** that may be used in hate speech or threats. Based on the case we are investigating, we include generic and context-specific terms but also misspelled or disguised hate content, like:

- "Andrei Toma";
- "Președinte" (president)
- "îți dau cu capul de bordură" ("smash your head against the curb");
- "vă iau familia" ("I'll take your family");
- "moarte " ("death");
- "vă înjur" (a phrase used to mask hate speech as free speech – I curse you).

Step 4: Search across open web sources

We begin data gathering from Romania located open web sources such as https://hotnews.ro/; https://www.g4media.ro/; https://www.cotidianul.ro/; https://adevarul.ro/; https://defapt.ro/. The list can be manually provided by the analyst by inputting them manually or by uploading a file (txt or csv) with a list of websites (100-10.000 URLs).

Step 5: Detect and flag content

The platform returned a hit. An article published on February 14 by the platform *defapt.ro* reported on the case of an individual who issued threats of violence against the President and incited acts of violence targeting both him and his family, via the social media platform TikTok. The message in question — which, despite breaching community standards, has not been taken down — has accumulated nearly 100,000 views and approximately 20,000 likes. Even if the crawl began in the open web, it correlated with social media to determine the spread (virilisation). In the same time, some foreign text was detected within the relevant content and with the translation module, it was translated from English, Arabic, Russian, Ukrainian and Hungarian to Romanian to determine if results are relevant according to the given keywords. The information is automatically flagged as **Level 1 Priority (Threatening Language** and **Incitement to Violence)**, an alert is being issued, and the information is further analysed by the analyst in charge of assessing threats.

Step 6: Multimedia analysis

Based on the textual hit, a number of multimedia content was also discovered in the same sources. Multimedia has different formats: video, images and audio. We start performing multimedia analysis like:

- **Visual analysis through AI**: detection of hate symbols in images that incite to violence and hate;
- **OCR extraction**: certain images contain text captions that incite violence and hate speech, that will be further distributed to the text analysis for flagging. A similar approach is done also to video (that is being split into frames) and a similar technical approach is performed;
- **Speech to text analysis**: for video and audio files detected, transcripts are generated through dedicated modules, to determine topic, speakers and analyse the presence of hate speech, incitement to violence and so on.

Step 7: Analyse distribution and reach

Using the platform's analytics tools, we identify the initial video posted on TikTok and start examining:

- **Engagement metrics**: likes, shares, comments, reposts;
- **Amplifiers**: Who shared it (e.g., known political agitators or public figures);
- **Audience demographics;**
- **Virality curve**: How fast it spreads over time.

The platform identifies that the video was **reshared 215 times** on TikTok, **cross-posted** to Telegram channels, and quoted in **3 Facebook public groups**. The origin of spread is detected (***patient zero***), with clear explanation of the results through xAI to user **ucideipetoti**.

Step 8: Generate an incident report

You export a detailed report that includes: The original content (video, captions, timestamp); Screenshots and transcript of the threatening message; Spread analysis (where, who, how many); Sentiment and severity score; Historical context (if the user is a repeat offender); Recommended action: flag to TikTok, notify law enforcement liaison, etc. The format of the report is based on a template that can be later on edited without coding in order to adapt it to the needs required by high-ranking officers.

Step 9: Continue monitoring

You enable continue **tracking** on open web sources for other materials posted by the specific user; imitation accounts, threats in the comment section (both on open web sources and social media platforms).

Step 10: Platform monitoring

The administrator monitors that the platform is working adequately, that it performed a back-up of the collected data in order to prevent data loss in case of corruption, and that logs are being stored with all user activity within the platform. In the event of an audit, all of these are relevant to ensure that all procedures were taken accordingly. Logs will contain detailed information about who (user), what (resources accessed/requests), when (timestamp), where (server/machine/location).

DeepFake Case

Avalanche platform features**:**

- Search open web and social platforms using advanced keyword detection;
- Analyse sentiment (positive, negative, neutral) and severity score (mild, moderate and high) from the comments section and timeline visualization;
- Propagation map and amplifier identification;
- Detect and correctly classify deepfake content (AI or manually generated multimedia: images, videos or audio)
- Identify „patient" zero (the users` account);
- Automatic transcription and threat classification of the original video content;
- View count, like count, and share tracking;
- Export of structured reports in PDF or JSON formats, labelled „*Incident Report*", that can further on be used by the specific stakeholders (the protected dignitary; the security team; other advisors);
- Alert system for high-risk or rapidly spreading content.

Table 18: Deepfake use case presentation

Use Case ID:	**Deepfake_01**
Name:	Detecting deepfake using Avalanche, concerning one of our protected dignitaries
Timeframe:	Continuous monitoring activity within a predefined timeframe

Primary Actor(s):	• Administrators (Avalanche technical maintenance team); • Intelligence Operational Team (OSINT extraction and integration); • Intelligence Analyst Team (risk assessment); • High-ranking officers (decision makers); • Avalanche Platform (connected to the Internet; does the analysis and gives alerts).
Main Goal:	• Detect deepfake content in posted images files; • Detect deepfake content in posted videos files; • Detect deepfake content in posted audio files; • Identify patient zero (source/origin of spread); • Generate propagation map; • Perform text transcript from video (speech to text capability); • Perform OCR from images / video frames (map text from images); • Perform sentiment analysis on text; • View count, like count, and share tracking; • Export content in report format (pdf and JSON) – for further analysis; • Generate alerts for specified thresholds.
Overview:	After starting the monitoring of the suspicious source, the Avalanche platform will allow: • Collection in real time and analyses of the social media, blogs, forums, onions and websites; • Integration of static analysis like the prediction of links between suspicious users and early detection of groups that perform deepfake; • Create alerts when deepfake is being used; • Proactive online threat assessment (Surface/Deep/ Dark Web and social media).
Precondition(s):	• Datasets (including text posts, posts with pictures, posts with video sequences, etc.) – data populated online in the platforms / social media accounts to simulate real operation (no static data from files or from database will be considered); • Equipment needed for demonstration during the used case – Computers / Avalanche Platform / Access to the internet; • Access to the platform (users created, roles defined, need-to-know access to specific activities) .
Main Input(s):	• Social media networks (Facebook, Twitter, YouTube, TikTok, etc.); • Surface/deep/dark web; • Keyword list (manual input or file); • URL list (manual input or file); • Credentials.
Main Output(s):	• Data regarding deepfake messages posted by individuals / groups of people on social networks / web addressed to the high officials; • List of suspicious persons/groups/communities discovered and links between entities involved; • Multimedia content with analysis results (bounding boxes, transcripts, OCR, etc.); • Security gathering report / threat assessment report.

Success Post Condition(s):	• Deepfake content is accurately detected and flagged from web; • Deepfake content is accurately detected and flagged from social media; • Patient zero is detected; • Propagation map is constructed; • Statistical data is collected to offer insight on dissemination level of information; • Reports are generated.
Failed Post Condition(s):	• Deepfake content is not accurately detected; • Deepfake content is false positively detected; • Information is not collected from social media; • Information is not collected from surface web; • Information is not collected from dark web; • Patient zero is not detected or falsely detected; • Propagation map is not constructed or is not correctly generated; • Statistical data is not collected or is inaccurately reported; • Reports are not generated with accurate data.

Operational steps:

Step 1: Access the platform

Firstly, the analyst will launch the specialized tool designed to detect deepfakes directed at our protected president, namely the Avalanche Platform, by providing the credentials necessary for the authentication. Each analyst has its own credentials, provided by the administrator in advance. Otherwise, the administrator generates the account and provides credentials for access (one time password that is changed at the first logon process by the user for privacy concerns).

Step 2: Define the subject

We will create a new operation for the specific task that we are performing at that given time (e.g., “Ion Marian – amenințări”) in the platform’s dashboard, in order to centralize all monitoring activities related to our protected president, at that exact time. Each operation is uniquely identified and named (no duplicates are permitted and access to the operation is based on the need-to-know principle – by default nobody has access until explicit access is provided to the users). Granular access is relevant for sensible tasks like OSINT operations.

Step 3: Set up keyword tracking

To capture relevant content, we will configure a set of **specific, high-risk keywords and phrases** that may be used in deepfake or threats. Based on the case we are investigating, we include generic and context-specific terms but also misspelled, like:

- "Andrei Toma";
- "Președinte" (president);
- "bancă" ("bank");
- "vindeți tot" ("sell everything");
- "faliment" ("bankruptcy");
- "vă jur”; ("I swear");

- “Euro”;
- “economie” ("economy").

Step 4: Search across open web sources

We begin data gathering from Romania located open web sources such as https://hotnews.ro/; https://www.g4media.ro/; https://www.cotidianul.ro/; https://adevarul.ro/; https://defapt.ro/. The list can be manually provided by the analyst by inputting them manually or by uploading a file (txt or csv) with a list of websites (100-10.000 URLs).

Step 5: Detect and flag content

The platform returned a hit. An article published on February 14 by the platform https://trueRomania.ro reported on the case of an individual who distributed on TikTok videos with former president Andrei Toma inciting people to protest against banks and the national bank, via the social media platform TikTok. The messages in question — which, despite breaching community standards, has not been taken down — has accumulated nearly 1,000,000 views and approximately 200,000 likes. Even if the crawl began in the open web, it correlated with social media to determine the spread (viralisation). While analysing the content, it was discovered that on Telegram some deepfake images are circulating with same topic, inciting to protests and bankruptcy. Some deepfake audio files and video were also discovered on the dark web, with the former president speaking in a foreign language (Hungarian), to invite also local communities to follow the lead as well.

Step 6: Multimedia analysis

Based on the textual hit, several multimedia contents was also discovered in the same sources. Multimedia has different format: video, images and audio. We start performing multimedia analysis like:

- **OCR extraction**: certain images contain text captions that incite protests and disorder, that will be further distributed to the text analysis for flagging. A similar approach is done also to video (that is being split into frames) and a similar technical approach is performed;
- **Speech to text analysis**: for video and audio files detected, transcripts are generated through dedicated modules, to determine topic, speakers and analyse the presence of incitement to violence, protests, disorder and so on.

Step 7: Analyse distribution and reach

Using the platform's analytics tools, we identify the initial video posted on TikTok and start examining:

- **Engagement metrics**: likes, shares, comments, reposts;
- **Amplifiers**: Who shared it (e.g., known political agitators or public figures);
- **Audience demographics;**
- **Virality curve**: How fast it spread over time.

The platform identifies that the video was **reshared 1000 times** on TikTok, **cross-posted** to Telegram channels, and quoted in **10 Facebook public groups**. The origin of spread is detected (***patient zero***), with clear explanation of the results through xAI to user **bancilesuntcancer**.

Step 8: Generate an incident report

You export a detailed report that includes: The original content (video, captions, timestamp); Screenshots and transcript of the message; Spread analysis (where, who, how many); Sentiment and severity score; Historical context (if the user is a repeat offender); Recommended action: flag to TikTok, notify law enforcement liaison, etc. The format of the report is based on a template that can be later on edited without coding in order to adapt it to the needs required by high-ranking officers.

Step 9: Continue monitoring

You enable continue **tracking** for on open web sources for other materials posted by the specific user; imitation accounts, threats in the comment section (both on open web sources and social media platforms).

Step 10: Platform monitoring

The administrator monitors that the platform is working adequately, that it performed a back-up of the collected data to prevent data loss in case of corruption and that logs are being stored with all user activity within the platform. In case of an audit, all of these are relevant to ensure all procedures were taken accordingly. Logs will contain detailed information about who (user), what (resources accessed/requests), when (timestamp), where (server/machine/location).

Evidence-based & Secure Information Exchange

Avalanche platform features**:**

- Capability to select granularly relevant information that can be sent securely from one end to another;
- Capability to encrypt information securely, ensuring that information cannot be decrypted by unauthorized parties;
- Capability to hash information in order to verify data integrity and prevent manipulation;
- Capability to log all activities performed in order to ensure that non-repudiation principle is ensured;
- Capability to decrypt information securely, ensuring that information can be validated at the recipient;
- Capability to compare hashes at the recipient to ensure it was not altered during transit;
- Capability to ensure logs cannot be altered by any type of user (including root).

Table 19: Evidence-based & Secure Information Exchange use case presentation

Use Case ID:	**Secure_exchange_01**

Name:	Rapid Response Through Secure and Timely Evidence Collaboration in Law Enforcement
Timeframe:	On-demand action when joint operations are performed between LEAs
Primary Actor(s):	• Administrators (Avalanche technical maintenance team); • High-ranking officers from LEA1 (decision makers); • High-ranking officers from LEA2 (decision makers); • Avalanche Platform (connected to the Internet; to perform transfer).
Main Goal:	• Select relevant information; • Encrypt information to ensure a high level of security; • Hash information to ensure integrity checks; • Log information to ensure non-repudiation principle; • Transfer any type of data and any amount of data (with no restrictions).
Overview:	After starting the monitoring of the suspicious source, the Avalanche platform will allow: • Collection in real time and analyses of the social media, blogs, forums and websites; • Integration of static analysis like the prediction of links between suspicious users and early detection of groups that perform hate speech; • Create alerts when hate speech is being used; • Proactive online threat assessment (Surface/Deep/ Dark Web and social media).
Precondition(s):	• Keys are installed and pre-configured within the systems; • Data is available within the platform, can be selected and sent securely; • Equipment needed for demonstration during the used case – Computers / Avalanche Platform / Access to the internet; • Access to the platform (users created, roles defined, need-to-know access to specific activities) .
Main Input(s):	• Keys for encryption; • Clear Data / evidence to be sent at the sender; • Encrypted Data at the receiver; • Credentials for access.
Main Output(s):	• Encrypted data at the sender; • Clear data at the receiver; • Activity logging.
Success Post Condition(s):	• Information is encrypted successfully; • Information is hashed successfully; • Information is signed successfully; • Information is sent successfully; • Information is received successfully; • Information is verified successfully; • Information is decrypted successfully; • Activity is logged accurately; • Key exchange is performed accurately.

Failed Post Condition(s):	• Information cannot be encrypted; • Information cannot be hashed; • Information cannot be signed; • Information cannot be sent; • Information is not received; • Information cannot be verified; • Information cannot be decrypted; • Not all activity is logged; • Key exchange/refresh is not performed.

Operational steps:

Step 1: Access the platform

Firstly, the high-ranking officer will log within the Avalanche platform, by providing the credentials necessary for the authentication. Each high-ranking officer has its own credentials, provided by the administrator in advance. Otherwise, the administrator generates the account and provides credentials for access (one time password that is changed at the first logon process by the user for privacy concerns).

Step 2: Define the subject

The high-ranking officer selects the existing case where information is being stored and selects granularly the relevant content (text, image, video, audio) that will be considered to be sent to their counterparts. Besides actual information, also the result of the analysis is attached to the transfer. As a result, a total of 10 text posts, 3 videos, 20 images and 1 audio file are selected and prepared for sending from SPP to SPPS.

Step 3: Set up the security pre-requisites

To ensure proper security measures, keys are pre-loaded within the system to ensure symmetric encryption with the secret key and signing of the content with the private key, while the public key is sent to the destination for verification/validation. This can be done automatically by the system, but also with the action of the user, with certain prompts and input. All actions are audited, step by step to ensure a proper logging mechanism, that all steps are accounted for and security measure are met.

Step 4: Data transfer from the source (SPP)

All content defined in step 2 are prepared (encrypted, hashed, signed, etc.) and is sent to the destination one by one (or bulk), in real-time, with minimum delay between the sender and the recipient (selected from a list of know organisations that have Avalanche installed), without sacrificing the security robustness. Moreover, a notification is sent to the destination to inform about the sending process (e-mail or notification within the application).

Step 5: Data receive at the destination (SPP)

SPPS high ranking officers are informed about the transfer, and they approve it. Encrypted data is received at the destination (SPPS), a total of 10 text posts, 3 videos, 20 images and 1 audio file and the decryption and verification process are started. After successful validations of hashing, signature and decryption, information is stored locally at SPP headquarters.

Step 6: Content analysis

The high-ranking officer from SPPS analyses the content and assigns analysts to also review content and perform new searches on the topic. The received content contains actual information but also analysis results from the SPP Avalanche platform (there is no need for local platform to rerun analysis on the information received). As a result, information is validated and extended searches are performed to understand the wider perspective of the situation.

Step 8: Generate an incident report

You export a detailed report that includes: The original content (video, captions, timestamp); Screenshots and transcript of the threatening message; Spread analysis (where, who, how many); Sentiment and severity score; Historical context (if the user is a repeat offender). The format of the report is based on a template that can be later on edited without coding in order to adapt it to the needs required by high-ranking officers.

Step 9: Continue monitoring

You enable continue **tracking** for on open web sources for other materials posted by the specific user; imitation accounts, threats in the comment section (both on open web sources and social media platforms).

Step 10: Platform monitoring

The administrator monitors that the platform is working adequately, that it performed a back-up of the collected data to prevent data loss in case of corruption and that logs are being stored with all user activity within the platform. In case of an audit, all of these are relevant to ensure all procedures were taken accordingly. Logs will contain detailed information about who (user), what (resources accessed/requests), when (timestamp), where (server/machine/location).