

AI-CODE

AI services for COntinuous trust in emerging Digital Enviroments

D3.2

Functional specifications and technical requirements

Grant Agreement no.	101135437
----------------------------	-----------

Project Title	AI-CODE - AI services for COntinuous trust in emerging Digital Environments
----------------------	---

Contractual Due date	31/03/2025
-----------------------------	------------

Actual submission date:	30/05/2025
--------------------------------	------------

Lead beneficiary:	DS TECH
--------------------------	---------

Deliverable number	D3.2
Deliverable Title:	Functional specifications and technical requirements
Description:	Description of the necessary functionalities and technical requirements of the AI-CODE services based on results from tasks T3.1 und T3.2.
Type/Nature	X R - Document, report ☐ DEM - <i>Demonstrator, pilot, prototype</i> ☐ DEC - <i>Websites, patent filings, videos, etc.</i> ☐ OTHER ☐ ETHICS – <i>Ethics requirement</i> ☐ ORDP – <i>Open Research Data Pilot</i>
Dissemination Level	☒ Public – fully open ☐ Sensitive — limited under the conditions of the Grant Agreement ☐ EU classified — RESTREINT-UE/EU-RESTRICTED, CONFIDENTIEL-UE/EU-CONFIDENTIAL, SECRET-UE/EU-SECRET under Decision 2015/444. For items classified under other rules (e.g. national or international organization), please select the equivalent EU classification level.
Work Package number	WP3
Work Package leader	EIPCM
Author(s) <i>in alphabetical order</i>	Miguel Antonio Barbero-Álvarez ,R iccardo Gallotti, Panagiotis Galopoulos, Beatriz Arregui Garcia, Lucio La Cava, Róbert Móro, Giovanni Natale, Manos Schinas, Giulio Spinozzi, Ivan Srba, Andrea Tagarelli
Internal Reviewers <i>in alphabetical order</i>	Manos Schinas (CERTH)

AUTHORS

Name	Partner	E-Mail
Giovanni Natale	DSTECH	g.natale@dstech.it
Giulio Spinozzi	DSTECH	g.spinozzi@dstech.it
Lucio La Cava	UNICAL	lucio.lacava@dimes.unical.it
Andrea Tagarelli	UNICAL	andrea.tagarelli@unical.it
Riccardo Gallotti	FBK	rgallotti@fbk.eu
Beatriz Arregui Garcia	FBK	barreguigarcia@fbk.eu
Manos Schinas	CERTH	manosetro@iti.gr
Panagiotis Galopoulos	CERTH	gpan@iti.gr
Róbert Móro	KInIT	robert.moro@kinit.sk
Ivan Srba	KInIT	ivan.srba@kinit.sk
Miguel Antonio Barbero-Álvarez	UPM	m.balvarez@upm.es

HISTORY OF CHANGES

Version	Date	Changes
0.1	10/03/2025	First version
0.2	25/03/2025	All service requirements added
0.3	20/05/2025	First complete version

TABLE OF CONTENTS

1. Executive summary.....	6
2. Introduction.....	7
2.1 Role of the Requirements Analysis in the Project	7
2.2 The Survey.....	8
2.3 Outline of this Deliverable	9
3. Service Specification	9
3.1 Disinformation Detection in the Next-Generation Social Media.....	9
3.1.1 Scope of the service	9
3.1.2 Functional Specification	10
3.1.3 Design considerations for specific user cohorts	13
3.1.4 Non-Functional Specification	13
3.1.5 Technical Requirement	13
3.2 Trustability and Credibility Assessment.....	14
3.2.1 Scope of the service	14
3.2.2 Functional Specification	14
3.2.3 Design considerations for specific user cohorts	16
3.2.4 Non-Functional Specification	16
3.2.5 Technical Requirement	17
3.3 Media Asset Annotation and Management (MAAM).....	17
3.3.1 Scope of the service	17
3.3.2 Functional Specification	18
3.3.3 <i>Design considerations for specific user cohorts</i>	20
3.3.4 Non-Functional Specification	20
3.3.5 <i>Technical Requirement</i>	21
3.4 Generative AI Interactive Coaching Service & Dynamic Simulator	21
3.4.1 Scope of the service	21
3.4.2 <i>Functional Specification</i>	22
3.4.3 Design considerations for specific user cohorts	22
3.4.4 <i>Non-Functional Specification</i>	22
3.4.5 Technical Requirement	22

3.5	Transparency Service for AI Model Cards	23
3.5.1	Scope of the service	23
3.5.2	Functional Specification	24
3.5.3	Design considerations for specific user cohorts	25
3.6	Personal Companion for Understanding Disinformation	26
3.6.1	Scope of the service	26
3.6.2	Functional Specification	27
3.6.3	Design considerations for specific user cohorts	29
3.6.4	Non-Functional Specification	29
3.6.5	Technical Requirement	30
4.	Results and Overall Architecture	33
5.	Conclusions and Next Steps	32

Table of figures

Figure 1	Single-node architecture	31
Figure 2	Multi-node architecture	32

Table of tables

Table 1	Overview of AI-CODE services	7
Table 2	Functional Specification	10

1. Executive summary

The primary objective of the AI-CODE project is to develop an innovative ecosystem of services designed to assist media professionals in detecting and countering disinformation while supporting the production of trustworthy content. Building on the groundwork laid in previous tasks and deliverables, this document provides a detailed description of the functional specifications and technical requirements for the AI-CODE services.

These services, which include tools for disinformation detection, credibility assessment, interactive coaching, and transparency mechanisms, are designed to empower journalists, fact-checkers, and media professionals to operate effectively in dynamic and increasingly complex digital environments.

This deliverable describes the functional and technical requirements of the following AI-CODE services, based on the insights gathered in T3.1 (User-centred requirements analysis) and T3.2 (Functional and non-functional requirements definition):

- Disinformation Detection in the Next-Generation Social Media
- Trustability and Credibility Assessment
- Media Asset Annotation and Management (MAAM)
- Generative AI Interactive Coaching Service & Dynamic Simulator
- Transparency Service for AI Model Cards
- Personal Companion for Understanding Disinformation

The process for defining these requirements was grounded in a collaborative approach, leveraging input from surveys, workshops, and iterative discussions with project partners. The analysis included both functional needs, such as the specific features and capabilities required by the services, and technical requirements, such as infrastructure, deployment, scalability, and security considerations.

This document builds upon the outcomes of Deliverable D3.1, where user-centred requirements and scenarios were identified, to translate those insights into actionable technical and functional specifications. It serves as a critical bridge between the conceptual design of the services and their practical implementation in subsequent phases of the project.

Finally, this deliverable highlights the alignment of the AI-CODE services with project objectives, ensuring that the developed solutions will address real-world needs while incorporating cutting-edge technologies for combating disinformation and promoting the credible use of Generative AI.

2. Introduction

2.1 Role of the Requirements Analysis in the Project

The AI-CODE project is centered on the development of six innovative services (table 1) that aim to proactively assist media professionals in combating disinformation and producing trustworthy content. To ensure these services are effective, interoperable, and aligned with user needs, the user-centered requirements analysis and the definition of functional specifications and technical requirements play an important role.

Table 1 Overview of AI-CODE services

Service Name	Short Description
1. Disinformation Detection in the Next-Generation Social Media	Identifies disinformation on Fediverse social media platforms.
2. Trustability and Credibility Assessment	Evaluates the trustability and credibility of content and sources found in posts io Fediverse social media platforms
3. Media Asset Annotation and Management (MAAM)	Automatically annotates media content to facilitate organization and retrieval and detects synthetic and manipulated content, including deepfakes
4. Generative AI Interactive Coaching Service & Dynamic Simulator	Helps media professionals better understand how to appropriately use and critically evaluate outputs from LLMs and associated risks, including AI-supported verification of claims and counter-narratives for fighting disinformation.
5. Transparency Service for AI Model Cards	Reports the capabilities, limits, potential risks, and repercussions of different AI models.
6. Personal Companion for Understanding Disinformation	Helps to detect disinformation, hate speech and logical fallacies in articles.

The functional specifications definition ensures that the AI-CODE services are designed to address user needs and use cases identified in D3.1. To this end, for each service it refines the user-centered requirements identified in D3.1 into specific functional features to be implemented .

On the other hand, the technical requirements analysis defines the technological foundation that will enable the seamless deployment and operation of these services. It involves detailing infrastructure needs, deployment environments, interoperability standards, and scalability requirements. Together, these specifications form the backbone of the project, ensuring that the services are not only user-centered but also technically robust and future-proof.

In this way, this deliverable (D3.2) represents a continuation of the work initiated in D3.1, refining the user requirements into a functional specification, while introducing a more detailed focus on technical requirements. The document serves as a critical step in transitioning from the requirements analysis phase to the design and implementation phases of the AI-CODE services, ensuring that every component is aligned with both user and technical requirements.

To ensure a comprehensive and standardized gathering of technical requirements, a structured survey was developed and shared with all project partners. This survey, which is included in the next section, was specifically designed to capture the key technical aspects of each service.

The survey covered a range of topics, aiming to define all relevant technical parameters and specifications for each service, including infrastructure requirements, integration needs, and dependencies. All partners

actively participated in this process by completing the survey, providing valuable input that reflects the technical needs and constraints of their respective services.

The data collected through this methodology served as a foundation for defining a common structure applicable to all services, ensuring consistency and facilitating a streamlined implementation process across the project.

2.2 The Survey

This section lists the questions that make up the questionnaire that was shared among all service managers. The questionnaire was then collected and unified into the following sections.

- Service Name
- Preferred service namespace or group name if any
- Service preferred deployment name
- Brief description of the service and main functionalities
- Programming language used
- if any, framework/s used
- Type of service
- Is the service containerized?
- Do you need support for containerization?
- If yes, describe specific issues or requirements
- Preferred base image (if known)
- Should the service be publicly exposed?
- Does the service need to interact with other services?
- If yes, list the services it needs to interact with
- Does your service needs eBPF?
- Does your service need specific linux kernel variables?
- Required communication protocol
- Is there an OpenAPI schema or other documentation?
- Minimum resource requirements (CPU)
- Minimum resource requirements (RAM)
- Minimum resource requirements (Disk space)
- Should the service scale automatically (autoscaling)?
- If yes, specify scaling parameters (CPU, RAM, requests, etc.)
- Does the service need multiple instances?
- Needs to be load balanced.
- If yes specify why
- Does the service require persistent storage?
- If yes, describe the filesystem or outline its specific needs
- If yes, describe the type of data and approximate size
- Does it need redundancy?
- Does it need data sync?
- Does the service require authentication?
- If yes, specify the type of authentication
- Do you need SSL/TLS certificates?
- Does the service need access to sensitive environmental variables or secrets?
- If yes, list the required environment variables or secrets
- Is there a logging system already configured?
- If yes, specify the log format | technology
- Do you need to integrate the service with monitoring systems (e.g., Prometheus, Grafana)?
- Does your service need CI/CD?
- Does your service need benchmarking?

- Does your service need architecture-level testing and validation?
- Provide a list of partners involved in the development of the same service
- Are there any specific requirements not covered by the previous questions?

2.3 Outline of this Deliverable

The deliverable is structured into several chapters that guide the reader through the methodologies, analyses, and outcomes of the requirement gathering process for the project.

Chapter 3 provides a comprehensive specification of the six key services that constitute the core outputs of the project. For each service, the functional requirements, capturing the expected capabilities from the user perspective, are outlined, followed by the technical requirements that define the specifications necessary to support their implementation. Functional requirements that are not directly linked to a specific user story have been marked with the label “A/F” (Advanced Functionality). The services analyzed include the Disinformation Detection in the Next-Generation Social Media (Section 3.1), the Trustability and Credibility Assessment (Section 3.2), Media Asset Annotation and Management (Section 3.3), Generative AI Interactive Coaching Service & Dynamic Simulator (Section 3.4), Transparency services for AI Model Cards (Section 3.5), and the Personal Companion for Understanding Disinformation (Section 3.6). This structure ensures clarity and consistency in presenting the requirements across all services. Some of the services comprise of multiple tools (components) and they may also integrate other tools (modules) through their APIs. In those cases, all relevant components and integrated modules are described.

Chapter 4 summarizes the results of the requirement analysis and their implications for the subsequent phases of the project. Finally, Chapter 5 concludes the deliverable by outlining the next steps, focusing on how the defined requirements will inform the design and development of the services moving forward.

3. Service Specification

3.1 Disinformation Detection in the Next-Generation Social Media

3.1.1 Scope of the service

The scope and objective of the service have been described in D3.1. For easier understanding of the subsequent text, we provide here a brief summary of the service scope.

The growth of decentralized social networks, such as Mastodon and other platforms within the Fediverse, has created both opportunities and challenges. These networks offer users more control over their data and communities but also introduce complexities in moderation, content management, and transparency. To address these challenges, this service combines two essential tools: the **Fediverse Network Moderation Tool** and the **Mastodon Instances Inspector**.

The **Fediverse Network Moderation Tool** supports administrators in maintaining safe, inclusive spaces by streamlining moderation tasks and fostering cross-instance collaboration. Meanwhile, the **Mastodon Instances Inspector** provides valuable insights into the structure, activity, and health of Mastodon instances, enabling informed decision-making and better network management.

Together, these tools empower administrators, moderators, and users to navigate the decentralized social web effectively, ensuring a balance between freedom and accountability.

3.1.2 Functional Specification

As defined in D3.1, the service combines two tools: the Fediverse Network Moderation Tool (the main aim of which is to support administrators in maintaining safe, inclusive online spaces in the next-generation (federated) social media by streamlining moderation tasks and fostering cross-instance collaboration) and the Mastodon Instances Inspector (which aims to provide valuable insights into the structure, activity, and health of Mastodon instances, enabling informed decision-making and better network management). For clarity, we prefix the user stories (introduced in the deliverable D3.1) of the former with 1 (e.g., 1.1, 1.2) and with 2 in case of the latter (e.g., 2.1, 2.2, etc.)

Table 2 Functional Specification

Feature Specifications (based on Revised Functional Requirements)	Corresponding User Stories
1. The system shall allow users to enter keywords to find and manage a list of servers associated with harmful interactions (e.g., racism, harassment). The system shall allow users to filter and customize the curated list of servers based on selected criteria and keyword matches. 1.1. Allow users to input keywords related to harmful behaviour. 1.2. Allow filtering and customization of the curated list.	User Story 1.1: Network Moderation User Story 1.2: Server Discovery
2. The system should be able to regularly (e.g., daily) crawl the Fediverse for getting the most up-to-date network moderation information from existing servers, that will be used to build the lists. 2.1. Schedule regular (daily) crawling of Fediverse instances. 2.2. Extract and update moderation-related metadata automatically.	User Story 1.3: Community Safety
3. The system shall include a database for storing and updating the metadata about network moderation publicly available from each instance. 3.1 Provide querying functionalities for fact-checkers and analysts.	User Story 2.3: Access to the database of bans
4. The system shall generate and provide Mastodon-compatible files for enforcing moderation on identified servers. Export blocklists and moderation rules in Mastodon-compatible formats.	User Story 1.4: Policy Analysis

5. The system shall display historical data on server flags, including the number of times servers have been flagged for specific reasons. Allow trend analysis on flagging reasons over time.	User Story 1.1: Network Moderation
6. The system shall implement AI-driven content analysis to assess and report on recent posts from instances flagged for misinformation or hate speech 6.1. Hate-speech detection: 6.1.1. Show ratio of posts of the selected instance considered to spread hate-speech. 6.1.2. Show ratio of users of the instance whose posts spread hate-speech 6.2. Detection of machine-generated text 6.2.1. Allow users to set parameters of detection (classification threshold or accepted level of error) 6.2.2. Show ratio of posts of the selected instance classified as machine generated 6.2.3. Show ratio of users of the instance whose posts are classified as machine generated 6.2.4. Show evolution of the number of the machine-generated texts (posts) in time (if data spanning a large time range is available) 6.3. Prevalence of fact-checked content: 6.3.1. Show ratio of posts of the selected instance that contain a previously fact-checked claim 6.3.2. Show a list of most prevalent fact-checked claims together with links to the fact-checking organization 6.4. Check-worthiness detection: 6.4.1. Show ratio of posts of the selected instance considered to be check-worthy. 6.4.2. Show ratio of users of the instance whose posts are highlighted as check-worthy. 6.5: Synthetic Image detection: 6.5.1. Show ratio of images from posts	User Story 2.1: Instance Inspector User Story 2.6: Assessing most common topics User Story 2.7: Safe Mastodon adherence

considered to be synthetically generated	
7. The system shall enable users to search for specific instances and retrieve their credibility status within the banning network. 7.1. An instance can be found by introducing into the system the specific name 7.2. Other searches can be done by filtering instance by feature: most banned, most followed...	User Story 2.1: Instance Inspector
8. The system shall provide tools for tracking the spread of content across the Fediverse, identifying key sources involved in misinformation dissemination. 8.1. Identification of key instances initiating misinformation within the network 8.2. Highlight frequent misinformation clusters or hubs. 8.3. Computation of an Infodemic Risk Index (IRI) for an instance.	User Story 2.4: Tracking questionable sources User Story 2.5: Tracing information origin and dissemination
9. The system shall generate visual reports that highlight trends and patterns in information spread and moderation dynamics between instances. 9.1. Network visualizations showing the safety degree of the instances. 9.2. Show cluster maps of instance relationships based on moderation behaviour.	User Story 2.2: Network analysis for dis/misinformation insights
10. The system shall display a summary of bans issued and received for each instance, including the reasons for bans along with their frequency and timestamps. 10.1. Show summary of bans issued and received per instance. 10.2. Display reasons for bans with frequency and timestamps.	User Story 2.4: Tracking questionable sources
11. The system shall implement network analysis features to quantify moderation participation and polarization among instances based on their banning history. 11.1. Show banning tendency of each instance. 11.2. Highlight clusters of instances with similar moderation behaviour provided their role in the network (central instance, outlier...).	User Story 2.5: Tracing information origin and dissemination

3.1.3 Design considerations for specific user cohorts

The system is designed to support both novice and expert users through a user-friendly interface with progressive disclosure of advanced features. Novice users benefit from guided keyword-based server searches (Feature 1) and easy downloads of moderation files and inspector output (Feature 4), supported by tooltips and simple workflows.

Expert users, such as fact-checkers and journalists, can access advanced options like direct database queries (Feature 3) and historical flagging data (Feature 5). Customizable crawling schedules (Feature 2) and detailed metadata access ensure flexibility for deeper analysis. This layered approach ensures usability and efficiency for all user types.

3.1.4 Non-Functional Specification

#	Revised Non-functional Requirements
1	The application shall be accessible and useable for non-technical audiences
2	The application shall be well documented and described
3	The application shall be responsive, with low computation time
4	The system shall provide a user-friendly interface that facilitates easy navigation and interpretation of data for non-technical audiences.
5	The system shall ensure scalability to accommodate a growing number of instances and users without performance degradation.

3.1.5 Technical Requirement

The service comprises of two separate modules: Fediverse Network Moderation tool and Mastodon Instances Inspector. The Mastodon Instances Inspector further uses other existing modules as defined in functional requirements and also in the technical requirements below.

3.1.5.1 Fediverse Network Moderation Tool

Survey Fields	Service Requirement
Service Name	Fediverse Network Moderation Tool
Programming language used	Python
Required communication protocol	HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	CPU, 16 RAM, >10GB Disk Space (10/20GB)
Type of service	Web application + Rest API
Interaction with other services	Mastodon Instances Inspector Tool, Machine Generated Text Detection, Fact-Checked Claim Detection, Hate-Speech Detection, Check-worthiness Detection, Synthetic Image Detection
Required authentication	Yes
Needing SSL/TLS certificate	Yes (Depending on deployment)
Needing to integrate with monitoring systems	No
Specific requirements	REST API

3.1.5.2 Mastodon Instances Inspector

Survey Fields	Service Requirement
Service Name	Mastodon Instances Inspector
Programming language used	Python
Required communication protocol	HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	CPU, 16 RAM, >10GB Disk Space (10/20GB)
Type of service	Web application + Rest API
Interaction with other services	Fediverse Network Moderation Tool, Machine Generated Text Detection, Fact-Checked Claim Detection, Hate-Speech Detection, Check-worthiness Detection, Synthetic Image Detection
Required authentication	Yes
Needing SSL/TLS certificate	Yes (Depending on deployment)
Needing to integrate with monitoring systems	No
Specific requirements	REST API

3.2 Trustability and Credibility Assessment

3.2.1 Scope of the service

The scope and objective of the service have been described in D3.1. For easier understanding of the subsequent text, we provide here a brief summary of the service scope.

This service will consist of a set of tools (AI-based methods and models utilising deep learning, NLP, image matching, etc.) able to automatically detect potential credibility signals that can be later combined by various assessment algorithms into credibility indicators to decide on the trustability and credibility of content and sources. As input, media professionals will be able to provide either: 1) an existing news article/a social media post (e.g., to assess the trustability/credibility of a content/source they aim to write); or 2) a custom piece of content (e.g., to assess the trustability/credibility of their own work). As output, various forms of explainable trustability and credibility assessments will be provided. The service will cover credibility signals related to both content as well its source (either at a level of individual creators or at a higher level of social network groups/pages/channels or web domains).

3.2.2 Functional Specification

Feature Specifications (based on Revised Functional Requirements)	Corresponding User Stories
- The credibility assessment service will allow the users to identify low credibility content in posts belonging to Fediverse social media. - It will return an aggregated score within a scale of labelled levels of credibility of the analyzed post. - It will have the option for the user of check the individual scores that compose the main output, based on individual functionalities within the service - Optional explanations about the meaning of the	Use Case 1: Professional investigation of online sources of information Use Case 2: Checking social media

results and how they have been calculated will be provided to enhance its understanding from the user's perspective 1.4. One of the functionalities analyzes the metadata of the account that shared the post under analysis. The user of the service will be able to check the most relevant related metadata, as in date of usage, followings and followers, etc. 1.5. One of the functionalities checks the artificiality of the account that shared the post under analysis. The user of the service will be able to assess whether the account has been self-described as a bot, or judging by the score, infer if the account behaves sharing artificial data. 1.6. One of the functionalities checks interactions with the post under analysis. The user of the service will be able to assess how many interactions with the post have been done and infer its influence or virality with it. 1.7. One of the functionalities checks the propagation graph analysis of the post under analysis. Relating to point 1.6, the user of the service will be able to assess the existing degrees of interaction with this post and check how many accounts it has reached. 1.8. One of the functionalities checks the URL linked in the post under analysis, if it exists. The user of the service will be able to check valuable data of the URL, about its domain and ranking. 1.9. One of the functionalities checks the URL linked in the post under analysis, if it exists. The user of the service will be able to check valuable data of the URL, about its domain and ranking. 1.10. One of the functionalities checks the text of the post under analysis. The user of the service will be able assess different analyses over it, such as checking the probabilities of the text being synthetic, or hate speech detection. 1.11. One of the functionalities checks the image found in the post under analysis. The user of the service will be able to check image forensics using different services through REST APIs, so valuable conclusions about the credibility of the image will be drawn.	
2. The service should allow the user to customise the dashboard to display only the results of the output modules they are interested in. 2.1 The service should have intuitive functionalities that will display or hide results, helping the user to customise the content they see with ease	Use Case 1: Professional investigation of online sources of information Use Case 2: Checking social media

3. The platform should have an history of the searches that can be updated with new ones 3.1. The history of the searches should display the previous searches of the logged user 3.2. The history of the searches will allow users to compare between the metrics previously retrieved and the most recent ones	Use Case 3: Consulting history of searches
4. The service must be accessible by UI and REST API 4.1. The user will be able to select which access is more adequate for them in any moment 4.2. The UI will be accessible and navigated like a webpage, thus providing the user a familiar environment to navigate 4.3. For the users with technical background, endpoints will be provided for REST API access	Use Case 1: Professional investigation of online sources of information Use Case 2: Checking social media

3.2.3 Design considerations for specific user cohorts

The system is designed to be intuitive and straightforward. By simply entering the link to a post, users can extract all the information that was gathered to generate the credibility score. Each section includes a brief description explaining how the corresponding module works, along with an option to expand the view for more detailed, technical insights.

When using this service, regular social media users can quickly get a sense of whether a post is credible, while also learning about the key parameters commonly associated with trustworthy content versus those linked to misinformation. The plots, graphs, and tables are presented in a clear and accessible format. For more advanced users, such as journalists, data scientists, or marketing managers, the platform also allows for deeper exploration of the underlying data and methods, enabling more comprehensive analysis and interpretation.

3.2.4 Non-Functional Specification

#	Revised Non-functional Requirements
1	The service must work in English language
2	The service must be available on Mastodon and Bluesky
3	The service showing should show the trustability assessment for each component and then show transparently how the overall score is calculated
4	The service could have a “Help me” section
5	The “Help me” section could include definitions on what each tool does for a better understanding for the user
6	The application should be well documented and described
7	The application should be accessible and useable for non-technical audiences
8	The application should be easy to use
9	The application should be easy to learn
10	The application should be used in a browser

3.2.5 Technical Requirement

Survey Fields	Service Requirement
Service Name	Trust and Credibility Assessment
Programming language used	Javascript + HTML (Vue.js), Python
Required communication protocol	HTTP/HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	CPU, 16 RAM, >10GB Disk Space (10/20GB)
Type of service	API REST, Web App, Celery “task broker”
Interaction with other services	Deepfake detection, Machine generated text detector
Required authentication	Not required for now
Needing SSL/TLS certificate	Yes (Depending on deployment)
Needing to integrate with monitoring systems	No
Specific requirements	• GPU desirable • REST API to hook endpoints from other services

The Trustability and Credibility Assessment Service can incorporate additional functionalities. The Machine-generated text detector is currently linked to this service, too.

3.3 Media Asset Annotation and Management (MAAM)

3.3.1 Scope of the service

The scope and objective of the service have been described in D3.1. For easier understanding of the subsequent text, we provide here a brief summary of the service scope.

The Media Asset Annotation and Management platform (MAAM) is an advanced service, initially developed in the context of the H2020 MediaVerse project, which forms the initial technological basis for advanced and trusted multimedia asset management and retrieval in AI-CODE. It is designed to annotate and manage media content using AI models. It applies labels to uploaded media based on various categories such as objects, actions, persons, news-related tags, and moderation tags. The platform supports powerful retrieval capabilities through text-based and visual similarity searches, enabling users to efficiently filter and find relevant media content, e.g. via its Near-Duplicate Detection (NDD) functionality, which identifies variations of the same multimedia asset that may be visually similar but not identical. The system also integrates several on-demand AI models for synthetic content detection, manipulation detection, deepfakes identification, and other forms of media integrity analysis.

Planned developments in AI-CODE aim to further enhance the platform’s effectiveness in combating disinformation by incorporating more advanced forgery detection, media manipulation analysis tools, and improved versions of existing detectors. In addition, explainability will be strengthened through the integration of model cards (from T6.2), offering greater transparency around how AI models make decisions. User profiles will also be enriched to capture users' expertise in disinformation analysis, making the platform more relevant for experts in the field. Tailored to serve journalists, fact-checkers, social scientists, researchers, and political science researchers, the enhanced MAAM platform will provide these professionals with robust tools to manage, analyze, and verify collections of media assets—particularly in the context of fighting disinformation and media manipulation.

3.3.2 Functional Specification

Feature Specifications (based on Revised Functional Requirements)	Corresponding User Stories
- The system shall support uploads from external sources through a list of URLs. - The system shall provide feedback on the upload progress and notify users upon completion. - The system shall provide feedback on the upload progress and notify users upon completion.	User Story 5: Media Asset Management
- The system shall support state-of-the-art video analysis techniques to detect deepfake content, including manipulated faces and other forms of visual alterations. - The system should provide results from multiple detectors to support identification of different manipulation techniques. - The system should provide localization and tracking of detected deepfakes across the video duration.	User Story 1: Identifying and Analysing Manipulated Video Content Using Near Duplicate Detection and Deepfake Analysis User Story 2: Detecting and Reporting Manipulated Media During Elections
- The system should analyze audio tracks in videos to detect inconsistencies or manipulations that indicate deepfake content, particularly in cases where the voice does not match the detected face. - The system shall provide localization of audio manipulations across the video duration.	User Story 1: Identifying and Analysing Manipulated Video Content Using Near Duplicate Detection and Deepfake Analysis
- The system shall support advanced video analysis techniques to identify fully synthetic videos generated by AI models. - The system shall support the analysis of large videos with long durations, ensuring efficient processing.	
The system should support near duplicate detection at the segment level, identifying and providing matching segments in the UI.	User Story 1: Identifying and Analysing Manipulated Video Content Using Near Duplicate Detection and Deepfake Analysis
- The system shall allow users to run synthetic image detection on demand using state-of-the-art detectors. - The system shall support synthetic image detection for batches of images to minimize user effort and enhance efficiency in analyzing multiple media files. - The system shall provide an overall assessment of media authenticity based on the results of multiple synthetic content detectors. - The system shall allow users to provide their own assessments by submitting textual evaluations and supporting claims regarding the authenticity of the analyzed media.	User Story 2: Detecting and Reporting Manipulated Media During Elections
The system shall provide user profiles that reflect their	User Story 4: Collaborating to

expertise in disinformation analysis. 7.1. The system shall display a user's past activity, including assessments of public images, videos, and audio files for manipulation. 7.2. The system shall allow users to receive endorsements and acknowledgments from other users based on their contributions and expertise in disinformation analysis.	Verify Potential Deepfake Content
8. The system shall allow users to share an asset with another user and request their contribution in the form of analysis, assessment, or additional insights. 8.1. The system shall notify users within the platform when they receive an asset for contribution.	User Story 4: Collaborating to Verify Potential Deepfake Content
9. The system shall provide model cards for the detection models, particularly those related to disinformation (e.g., synthetic content detection, deepfake detection, and manipulation detection). 9.1. The system shall provide insights into the intended use of each model to help prevent misuse or misinterpretation. 9.2. The system shall generate clear, textual explanations of automatic detection results to improve accessibility for non-expert users. 9.3. The system shall display confidence levels for all detections to help users assess the reliability of the results.	New User Story: Non-Expert User Understanding AI Detection Results
10. The system shall generate detailed reports based on detection results and user assessments formatted for easy sharing with stakeholders. 10.1. The system shall allow users to select which annotations to include in the report before generating it. 10.2. The system shall allow users to generate reports as either a downloadable PDF for offline access and official documentation or a sharable MAAM webpage for real-time collaboration and external stakeholder access.	User Story 1: Identifying and Analysing Manipulated Video Content Using Near Duplicate Detection and Deepfake Analysis User Story 2: Detecting and Reporting Manipulated Media During Elections User Story 3: Investigating and Debunking Out-of-Context Media Claims User Story 4: Collaborating to Verify Potential Deepfake Content
11. The system shall extend workspace access control by introducing user roles with different permissions, allowing for more flexible collaboration and management.	All user stories require workspaces to enable content sharing between users.
12. The system shall provide a dedicated Disinformation Analysis page to streamline and accelerate the detection of synthetic	User Story 1: Identifying and Analysing Manipulated Video

and manipulated content without requiring users to go through the standard upload and analyze workflow. 12.1. The system shall automatically apply relevant synthetic content and manipulation detectors upon media submission, as selected by the user, to reduce manual steps and streamline the analysis process. 12.2. Users can save analyzed media for later review or share findings with collaborators directly from the page.	Content Using Near Duplicate Detection and Deepfake Analysis
13. The system shall provide detection and tracking capabilities across all video analysis models, including manipulation detection.	User Story 5: Media Asset Management
14. The system shall generate detailed textual descriptions of videos to facilitate retrieval using free-text search queries, integrating scene-based captions	

3.3.3 Design considerations for specific user cohorts

The MAAM platform is being designed with different user cohorts in mind, particularly addressing the needs of both non-expert and expert users. The platform will include layered information visualizations: for example, novice users are supported with simplified outputs such as overall detection scores for deepfake videos or synthetic images. This high-level prediction score summarizes whether an asset is likely manipulated, without requiring technical background knowledge. These summaries will be paired with “Learn More” tooltips, offering additional help and guiding users toward deeper understanding of how these predictions have been produced. In line with a scaffolding approach, individual detector scores, originally designed for expert analysis, will also be available to non-experts, but only upon user interaction (e.g., expanding a results section), allowing gradual exposure to complex information (e.g., detectors specialized in Diffusion or GAN-based synthesis).

In addition, both user categories will benefit from access to documentation of the integrated AI models, by integrating the transparency service developed in T6.2. This service will provide standardized model cards that describe models’ functionalities, training data, evaluation results, known limitations, and potential risks. Importantly, the documentation will be designed to serve both expert and non-expert users through a multi-layered structure: model cards will be available at different levels of detail, ranging from simplified summaries with visual explanations and key metrics for non-experts, to comprehensive technical descriptions and performance breakdowns for expert users. The MAAM platform will follow this layered structure in its user interface, ensuring that users can explore AI model information progressively, starting with high-level summaries and optionally drilling down into more technical details.

For expert users, the interface will provide features that support more advanced tasks and workflows. The functional specifications include expert-mode functionalities such as a structured feedback mechanism that allow experts to annotate model outputs, validate predictions, and raise concerns about misclassifications or performance inconsistencies. These interactions are intended to improve transparency and model refinement over time.

3.3.4 Non-Functional Specification

#	Revised Non-functional Requirements
1	The system shall be able to scale to handle large volumes of media files, supporting simultaneous analysis for multiple users without significant degradation in performance.

2	The system shall be built with a modular architecture to facilitate easy updates, maintenance, and the integration of new features or models.
3	The system shall provide a user-friendly interface that is easy to navigate, allowing users with varying levels of technical expertise to effectively use the tool.
4	The system shall have an uptime of at least 99.9% over any given month, ensuring that it is reliably available for users during critical periods such as election monitoring.
5	The application should be well documented and described.
6	The application should be accessible and usable for non-technical audiences.
7	The application should be easy to use and learn.

3.3.5 Technical Requirement

Survey Fields	Service Requirement
Service Name	Media Asset Annotation and Management
Programming language used	Javascript + HTML (Nuxt, Vue.js), Java (Spring)
Required communication protocol	HTTP/HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	4 cores CPU, 16 GB RAM, > 50GB Disk Space, 16 GB VRAM
Type of service	Web App, REST API
Interaction with other services	Deepfake Detection Service, Media Annotation Service
Required authentication	Not required for now
Needing SSL/TLS certificate	Yes (Depending on deployment)
Needing to integrate with monitoring systems	No
Specific requirements	GPU is needed for the integrated services

3.4 Generative AI Interactive Coaching Service & Dynamic Simulator

3.4.1 Scope of the service

The scope and objective of the service have been described in D3.1. For easier understanding of the subsequent text, we provide here a brief summary of the service scope.

The Generative AI Interactive Coaching Service & Dynamic Simulator for Media Professionals helps media professionals to understand how generative AI can be used in trusted media production and countering disinformation. The two main components of this service, the **Interactive Coaching and Simulation Module (ICSM)** and the **Counter-narrative Recommendation Module (CNRM)**, are designed to work in tandem, providing media professionals with a holistic learning and continuous usage experience. Users can begin with the Interactive Coaching and Simulation Module to gain a foundational understanding of main principles of generative AI systems (LLMs) and prompt engineering. This knowledge can then be directly applied in the Counter-narrative Recommendation Module, where users can see how these principles inform the generation of effective responses to disinformation. Conversely, users of the Counter-narrative Recommendation Module can access the Interactive Coaching and Simulation Module to deepen their understanding of the underlying mechanisms that drive the Counter-narrative Recommendation Module. The Interactive Coaching and Simulation Module also integrates selected functionalities of text veracity verification from the Personal Companion for Understanding Disinformation service. This interconnected approach ensures that media professionals can develop both theoretical knowledge and practical skills in using generative AI responsibly and effectively in their work.

3.4.2 Functional Specification

3.4.2.1 Interactive Coaching and Simulation Module

3.4.3 Design considerations for specific user cohorts

The Interactive Coaching and Simulation Module addresses diverse user needs through a multi-tiered approach. For novices, it offers guided exploration with structured tutorials, visual demonstrations of LLM principles, and reliability checklists (“Guided Simulator”). Advanced users benefit from self-exploration modes (“Prompt Playground”) to experiment with complex adjustments (prompt or system parameters), and RAG implementations. The module interface progresses from foundational concepts to sophisticated techniques, allowing users to engage at their preferred complexity level. Integration with complementary services provides a scalable learning path, ensuring both beginners and experts can develop appropriate mental models of generative AI applications in journalism (“AI Claim Debunker”).

For the Counter-narrative Recommendation Module, the interface allows to address novice vs expert users in two main functionalities

1. The amount of evidence to be displayed (i.e. “number of relevant sentences”). For advanced users, a higher number of sentences can be required to have a deeper analysis of the background knowledge used.
2. Advanced users can play with the prompt field and test more advanced behavior of the LLM, while novice users can rely on the pre-defined (default optimized) prompt.
3. Depending on the context of use, the user can adapt the prompt to specific needs (e.g. tuning the verdict wording to better match social media posts style).

3.4.4 Non-Functional Specification

3.4.4.1 Interactive Coaching and Simulation Module

#	Revised Non-functional Requirements
1	The application shall be accessible and useable for non-technical audiences
2	The application shall be easy to use
3	The application shall be easy to learn
4	The application shall be used in a browser
5	The application shall provide differentiated support for beginner and expert users as two main types of user cohorts within the target user group of the application.

3.4.4.2 Counter-narrative Recommendation Module

#	Revised Non-functional Requirements
1	The application shall be accessible and usable for non-technical audiences.
2	The application shall be easy to use.
3	The application shall be used in a web browser.

3.4.5 Technical Requirement

3.4.5.1 Interactive Coaching and Simulation Module

Survey Fields	Service Requirement
Service Name	Interactive Coaching and Simulation Module
Programming language used	Python , Bash

Required communication protocol	HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	2 Cores CPU, 20GB RAM, 8GB VRAM, >6GB Disk Space
Type of service	Web App
Interaction with other services	Counter-narrative Recommendation Module, Personal Companion for Understanding Disinformation for Media Professionals, Transparency Service for AI Model Cards, Trustability and Credibility Assessment
Required authentication	Not for now
Needing SSL/TLS certificate	Yes
Needing to integrate with monitoring systems	No
Specific requirements	

3.4.5.2 Counter-narrative Recommendation Module

Survey Fields	Service Requirement
Service Name	Counter-Narrative Generation API (CNG API)
Programming language used	Python
Required communication protocol	HTTP/HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	24GB VRAM, 30GB Disk Space
Type of service	API REST
Interaction with other services	Counter-Narrative Generation API demo interface
Required authentication	Yes, Basic Auth, OAuth2, JWT
Needing SSL/TLS certificate	Yes
Needing to integrate with monitoring systems	No
Specific requirements	

3.5 Transparency Service for AI Model Cards

3.5.1 Scope of the service

The scope and objective of the service have been described in D3.1. For easier understanding of the subsequent text, we provide here a brief summary of the service scope.

The transparency service for AI model cards will be developed to define, create, and manage model cards for AI models. It will establish a standardized structure for model cards, specifying necessary fields such as performance metrics, algorithm descriptions, and ethical considerations, ensuring consistency across models and third-party applications that integrate these cards.

The service will offer an intuitive user interface (UI) to make model card creation accessible to users with varying technical backgrounds. It will also provide a software development kit (SDK) for developers to annotate their code, automating the generation of performance metrics and relevant information directly from the AI models.

Beyond creation, the service will include management functionalities like version control, updates, and model card storage, enabling developers and teams to maintain a history of model improvements, compare

versions, and ensure that the latest information is readily available. Additionally, the service will feature varying levels of detail within model cards to cater to both technical experts and non-technical stakeholders. It will also integrate advanced chatbot assistants to guide engineers through filling out necessary fields, offering suggestions and explanations, especially for complex model aspects.

3.5.2 Functional Specification

Feature Specifications (based on Revised Functional Requirements)	Corresponding User Stories
1. The service should offer solutions to create and communicate the performance, limitations and implications of the model. 1.1. Users must be able to input key metadata such as model name, version, type (e.g., classification, generation), and domain. 1.2. The service should utilize AI technology to suggest ethical risks based on model type, domain, use case and project description. 1.3. The system must provide a live preview of the model card as the user fills in the information.	User Story 1: Documenting an AI Model Using the Transparency Service
2. The service should offer tools to fill detailed information while providing a tool to construct multiple levels of details. 2.1. The structure of the model card should be fixed to avoid confusion when comparing models. 2.2. The model card should have a dedicated field for limitations and indented use. 2.3. Users must be able to describe datasets used for training, including source, size, labels, and known issues. 2.4. The system must guide users to describe possible biases or limitations in the datasets. 2.5. Users must be able to switch between different level of details view of each model card.	User Story 2: Interpreting AI Results Through Detailed Model Cards
3. The service should provide tools to simplify technical text and explain technical terminology. It should also give the freedom to edit every field including free text and structured entries. 3.1. The service should provide simplified non-expert summaries of technical text using utilizing AI technology. 3.2. The service should highlight and provide an explanation for technical terms using AI technology.	User Story 3: Generating Simplified Model Cards for Effective Communication of AI Insights
4. The service must be able to evaluate the performance of a model on different data sets and slices specified by the creator. It will also be able to present the results in an explanatory manner. 4.1. The metric calculation tool should be flexible and offer the creator the option to use different data sets and slices. 4.2. The service should provide a textual description of the metrics table by utilizing AI technology. 4.3. The system must support different metrics based on the task type (e.g., F1 score for classification, BLEU for NLP)	Title: Evaluating AI Model Performance Using Model Cards

tasks, etc). 4.4. Users must be able to upload a JSONL formatted file with the evaluation results for automatic metric extraction.	
5. The service should offer machine-readable and human-readable Model Cards in structured formats such as JSON and HTML.	User Story 5: Integrating Machine-Readable Model Cards for Enhanced AI Output Transparency
6. The service should offer storing management and version control for the Model Cards to update its content along with the model updates. 6.1. The service should store previous versions of model cards. The user should be able to access and revert to previous versions. 6.2. The service will keep track of changes between model card versions. 6.3. Users must be able to save work-in-progress and resume later.	User Story 6: Collaborative Management and Versioning of Model Cards for Deep Learning Model Development

3.5.3 Design considerations for specific user cohorts

We aim to provide a service for different user groups both for the creators and consumers of model cards. For the creators we aim for expert and novice AI engineers and researchers by providing several tools to guide and help through the process. The service offers suggested fields such as model name, version, license, ethical risks, limitations, training description, testing description, quantitative analysis, and more to guide the creator through necessary information about the model. Furthermore, the AI assistant is a tool that helps the users fill some of the fields by summarizing big text, metrics table, and dataset information to provide textual description for the respected fields. Through its chatbot the AI assistant can also suggest ethical risks when provided by a description of an AI project.

For the consumers we aim to provide model cards that can offer valuable information for experts and non-experts in AI. For this the creator can produce three levels of detail for the same model. The first one being the detailed model card aimed at AI experts which will provide technical information about the model. The second is almost identical to the first one with the addition of pop-up explanatory tips of technical words when the user hovers over them. This level of detail is aimed at consumers familiar with AI but not experts that might lack knowledge about specific technologies and algorithms. The third and final level is the more simplistic one by providing necessary descriptions to inform consumers that are not familiar with AI technologies. Several technical information such as model architecture, training algorithms, and more are omitted for the favor of more important fields such as ethical risks, limitations and performance. The creator can create those levels of detail by hand or by using the AI assistant which is equipped to create explanations for technical words for the second level and simplify text for the third level.

3.5.3.1 Non-Functional Specification

#	Revised Non-functional Requirements
1	Support for various machine learning frameworks (e.g., Python, TensorFlow, PyTorch).
2	The UI must be user-friendly and intuitive, with clear instructions and tooltips to guide

	users through creating and managing model cards.
3	Support standard file formats for export (e.g., JSON, PDF, HTML) to allow model cards to be used across various environments.
4	The system should be capable to recognize the content of a dataset to minimize user workload (e.g., “labels” column contains the target, “bbox” contains boundary boxes for object detection, etc).
5	The system must include several different metrics. The system will automatically choose and calculate the proper metrics for the task.
6	The AI Assistant responses should be provided to the user in a reasonable time.
7	The AI Assistant responses should be reliable and provide relevant information.
8	All model card information should be stored in a JSON file with a fixed structure while allowing field deletions and additions.
9	The service should be accessible from a browser.
10	Saving and loading model cards and version control management should be reliable. The service should provide backup storage in case of loss or corrupted data.

3.5.3.2 Technical Requirement

Survey Fields	Service Requirement
Service Name	Transparency Service for AI Model Cards
Programming language used	Python, Javascript + HTML
Required communication protocol	HTTP
Minimum resource requirements (CPU, RAM, Disk Space)	4 cores CPU, 6GB RAM, 16 GB Disk Space
Type of service	SDK/WEB APP
Interaction with other services	No
Required authentication	Not for now
Needing SSL/TLS certificate	Not for now
Needing to integrate with monitoring systems	No
Specific requirements	12GB VRAM

3.6 Personal Companion for Understanding Disinformation

3.6.1 Scope of the service

The scope and objective of the service have been described in D3.1. For easier understanding of the subsequent text, we provide here a brief summary of the service scope.

The Personal Companion for Understanding Disinformation is an LLM-based service that aims to assist junior, experienced journalists and fact-checkers to combat the spread of disinformation and misinformation, and, hence, improve the quality of media content. Specifically, it aims to: i) assist journalist students/in training to write articles without any logical fallacies, hate speech, and to not spread disinformation, ii) assist young journalists and media professionals to check the validity and credibility of the articles they consume, iii) produce reports on specific claims based on the websites/debunking reports they appear, iv) obtain a rough estimation of the validity of claims that have not been debunked based on the type of the media outlets they appear in, v) assist fact-checkers to search, maintain, and share their local debunking database, vi) understand the spread level of disinformation.

3.6.2 Functional Specification

Feature Specifications (based on Revised Functional Requirements)	Corresponding User Stories
- The system shall provide a logical fallacy detection functionality, which will detect logical fallacies in the text and provide the type of fallacy along with an informative link. - The system shall provide a textbox where the users can copy-paste the text they want to process. - The system shall highlight all logical fallacies that it can identify in the text. - The system shall provide the type of fallacy for each identified fallacy. - The system shall provide the option for the user to check the definition of the fallacy.	User Story 1: Fallacy detection for own articles User Story 2: Fallacy detection for website articles
- The system shall provide a hate speech detection functionality, which will detect the hate speech sentences in the text and provide the type of hate speech along with an informative link. - The system shall provide a textbox where the users can copy-paste the text they want to process. - The system shall highlight all hate speech sentences that it can identify in the text. - The system shall provide the type of hate speech for each detected sentence.	User Story 3: Hate speech detection for own articles User Story 4: Hate speech detection for website articles
- The system shall provide a claim detection and matching functionality that matches claims appearing in text with debunked claims stored in the database “Debunked Claims” (functional specification #10) - The system shall provide a textbox where the users can copy-paste the text they want to process. - The system shall highlight the detected claims in the text - The system shall provide the URL of the debunking report for each detected claim. - The system shall provide the summary of the debunking report (if exists) upon request.	User Story 5: Credibility assessment of claims included in own article. User Story 6: Credibility assessment of claims included in website article
- The system shall provide a claim detection and matching functionality that matches claims appearing in text with claims appearing on external webpages - The system shall provide a textbox where the users can copy-paste the text they want to process. - The system shall highlight the detected claims in the text - The system shall provide the URL of the matching web resource for each detected claim.	User Story 6: Spread assessment of claims included in own article User Story 7: Spreading assessment of claims User Story 9: Spreading assessment of false claims included in external articles

5. Based on user role, the system shall provide an automated generation of report functionality. 5.1. The system shall provide a textbox for insertion of multiple URLs. 5.2. The system shall provide a textbox for the insertion of a claim. 5.3. The system shall generate a report for the input claim based on the provided URLs. 5.4. The report shall be editable.	User Story 10: Automated generation of report on claims appearing on own claims. User Story 11: Automated generation of report on claims appearing in external website articles.
6. The service shall allow different functionalities per role: 6.1. Users must be authenticated to access the service. Role-based access control (RBAC) must be implemented. The types of users will be: 6.1.1. Journalist Students / Journalists in Training 6.1.2. Journalists 6.1.3. Editors 6.1.4. Fact-checkers 6.1.5. Social Media Managers/Community Editors 6.1.6. Trainers/Curriculum Developers. 6.2. Users with elevated rights must be associated with a specific organization upon registration or account setup.	A/F
7. The system shall allow the user to provide specific external webpages for similarity-checking 7.1. For users with elevated rights: Under functionality #4 the user shall be able to provide a set of URLs for searching.	A/F
8. The system shall allow the user to navigate and process the database with the debunked claims 8.1. For users with standard rights: The system must display a list of debunked claims. For each debunked claim must displayed the claim text, the debunking report and optionally a summary of the debunking report, date of debunking, topic. The user should be able to search for specific claims. 8.2. For users with elevated rights: -Must be able to add debunking reports by uploading spreadsheets. -They must see an option to mark any existing debunking report as invalid.	User Story 12: Processing of database with debunked articles
9. The service should support also non-english languages. 9.1. The system must provide a language selector in a prominent location . 9.2. When the user selects a language all UI text must appear in that language.	A/F
10. The database with the debunked claims ("Debunked Claims" database) should be optionally shared between people of the same organization 10.1. All debunking reports created by users within an organization should be optionally visible to all other users in the same organization 10.2. The organizational database should not be accessible by externals	A/F

11. The system shall provide a toolbox of all allowed functionalities (based on user type) on the initial page, allowing users to select the ones they need.	A/F
11.1. The initial page must display clearly in separate section all available functionalities with clickable boxes for selection.	

3.6.3 Design considerations for specific user cohorts

The system interface is thoughtfully designed to support the needs of its primary user cohorts—**journalism students, early-career journalists, journalists, editors, and fact-checkers**—by presenting information in a clear, modular format. Claim detection and matching both debunked claims and related media articles are presented in an intuitive, side-by-side layout, allowing users to quickly assess agreement or contradiction and verify factual accuracy (functional specification #3). The user can click on the various detected claims to be further informed about their veracity based on the debunking reports. For journalism students and those in training, detected claims are additionally checked against a curated set of web resources selected by experienced journalists, providing a trusted foundation for learning fact-checking methods (functional specification #4). This will guide them toward reputable materials and avoid overwhelming them with choices. Highlighting logical fallacies (functional specification #1) and hate speech (functional specification #2) within the article text further aids in training critical reading skills and supports editorial decision-making, particularly useful for users still developing media literacy or journalistic rigor, such as **journalism students** and **early-career journalists**. For the same reason, the database with debunked claims should also be available to this group in read-only mode. At the same time, hate speech and fallacy detection will allow experienced **journalists** and **fact-checkers** to quickly assess the credibility of the articles that they read. Further, the automated generation of reports functionality will be visible to journalists and fact-checkers to help them streamline their workflow.

For more advanced users, such as **experienced editors** and **professional fact-checkers**, the system will offer customizable browsing and editing functionality within the debunked claims database (functional specification #8). These users will be able to filter and tag content, update claim statuses, or cross-reference sources. This will empower them to verify and contribute to the evolving integrity of the database, fostering professional accountability and transparency. These users will be able to determine the external web sources that they wish the detected claims to be checked against.

Potential user cohorts such as **media asset managers** and **publication editors** will benefit from the system's backend coherence and structured metadata, which supports integration into larger editorial pipelines. The design will allow these users to quickly assess content risk (e.g., presence of hate speech or logical inconsistencies) before publication or archival. Overall, the interface will balance accessibility for learners and sophistication for professionals, ensuring broad usability across different levels of expertise.

3.6.4 Non-Functional Specification

#	Revised Non-functional Requirements
1	The application shall be well documented and described.
2	The application shall be easy to use.
3	The application shall be easy to learn.
4	The application shall be used in a browser.
5	The application shall provide differentiated support for beginner and expert users as two main types of user cohorts within the target user group of the application
6	The system shall be built with a modular architecture to facilitate easy updates, maintenance, and the integration of new features or models.
7	The system shall enable the automatic, timely, and secure delivery of updates and changes.

8	The system shall provide a user-friendly interface that is easy to navigate, allowing users with varying levels of technical expertise to effectively use the tool.
9	Where possible, the system shall be built by taking into account energy considerations.
10	The system shall provide the possibility for the user to select as many functionalities as they need.

3.6.5 Technical Requirement

Survey Fields	Service Requirement
Service Name	Personal Companion for Understanding Disinformation
Programming language used	Python
Required communication protocol	HTTP/HTTPS
Minimum resource requirements (CPU, RAM, Disk Space)	4 cores CPU, 28GB RAM, 128GB Disk Space
Type of service	API REST/WEB APP
Interaction with other services	Currently uses external services, e.g. Tavily, Groq
Required authentication	Yes, Basic Auth / OAuth2 / JWT / Other
Needing SSL/TLS certificate	
Needing to integrate with monitoring systems	No
Specific requirements	GPU with at least 28GB

4. Results and Overall Architecture

The completion of this deliverable marks a significant milestone in defining the functional specifications and technical requirements of the AI-CODE services. Through a structured information-gathering process, we have successfully collected and analysed the necessary details to outline a comprehensive framework for the development and integration of these services.

A key achievement of this phase has been survey-based data collection, which allowed all project partners to provide detailed insights into their respective services. The survey covered both functional aspects, such as service purpose and user interactions, and technical aspects, including programming languages, frameworks, communication protocols, and infrastructure requirements. This structured approach has ensured a clear and consistent understanding of how each service will operate within the AI-CODE ecosystem.

Following the survey, we have defined the functional specifications for each service, identifying their primary use cases, dependencies, and expected behaviors. This process has helped establish a coherent architectural vision, ensuring interoperability and alignment with the overarching project goals. Additionally, we have outlined the technical requirements, covering containerization needs, authentication mechanisms, security constraints, and scalability considerations. These specifications provide a solid foundation for the subsequent implementation and deployment phases.

The functional and non-functional requirements gathered throughout Task 3.3 have laid the foundation for defining a flexible and scalable deployment architecture for the AI CODE platform. This architecture concerns all services in general. Services that are already in an advanced stage, such as MAAM, will be integrated into the platform. The proposed architectural approach is designed to accommodate the diverse needs of the project's services, ensuring adaptability from early development stages to production-level deployments.

Two architectural scenarios have been outlined:

- **Single-Node Architecture:** Shown in Figure 1, it is designed for development and prototyping purposes, this setup uses Docker Compose to enable fast and cost-effective deployment of microservices. It offers an efficient entry point for partners to begin integrating and testing their services.
- **Multi-Node Architecture:** Shown Figure 2, it is targeted at production environments, this setup leverages Kubernetes for orchestrating containers, ensuring high availability, scalability, and automated service management. A suite of modern technologies—such as Cilium for networking, Rook Ceph for storage, Traefik for load balancing, and Prometheus/Grafana for observability—complements the architecture to create a robust ecosystem.

To further support interoperability and ease of development, a collection of essential tools (databases, caching systems, message queues, etc.) will be pre-deployed across both environments. This shared baseline allows each partner to work within a standardized, ready-to-use infrastructure while retaining the flexibility to customize it according to individual needs.

Overall, the defined architecture ensures that all services can be integrated in a coherent, scalable, and future-proof manner, aligning with the project’s strategic vision and technical requirements.

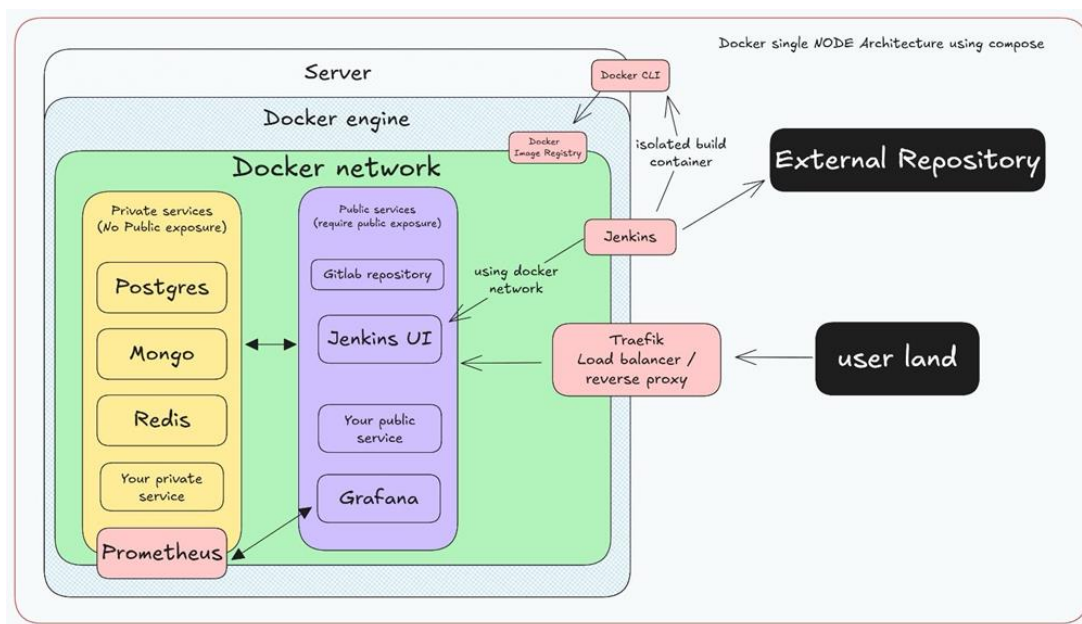


Figure 1 Single-node architecture

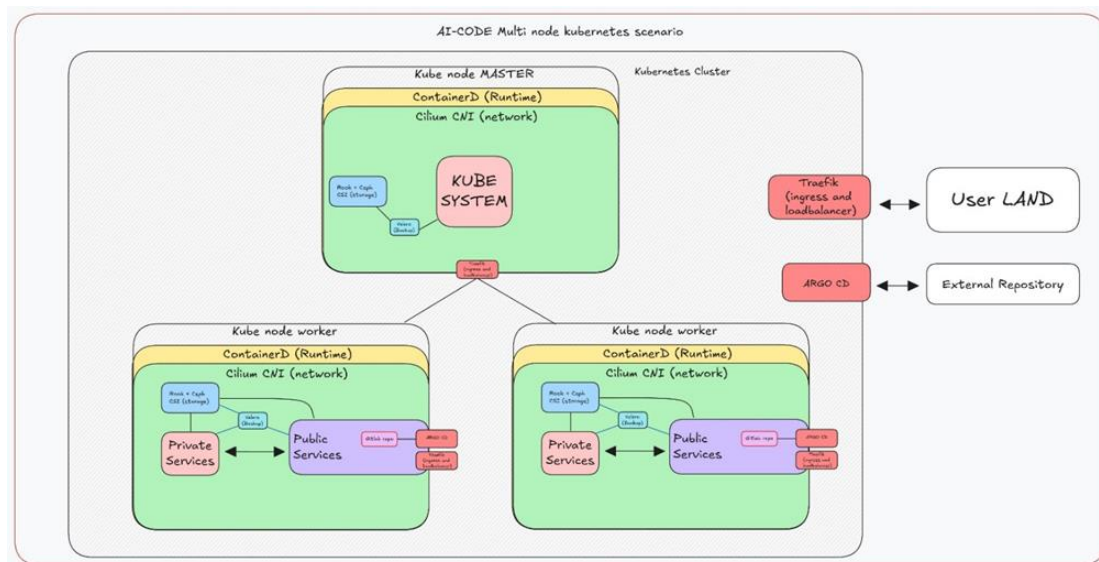


Figure 2 Multi-node architecture

5. Conclusions and Next Steps

In this deliverable, we provided detailed functional, non-functional, and technical specifications of all AI-CODE services, as well as a first proposal of the overall architecture.

Looking ahead, the findings documented in this deliverable will serve as the basis for the upcoming development phases. The specified requirements will guide the technical implementation and help ensure that all AI-CODE services are developed in a way that meets the needs of media professionals while maintaining a high standard of reliability, security, and interoperability.