Project Specification Document

CS 491 Senior Project

2025-2026 Fall Semester

VeriFact (Group T2503)

Orhun Ege Çelik - 22202321
Egehan Yıldız - 22203014
Alhassan Raad Jassim Al-Badri - 22201170
İrem Damla Karagöz - 22203691
Eray İşçi - 22201686

Supervisor: Sinem Sav
Innovation Expert: Mustafa Sakalsız

Table of Contents

1. Introduction

1.1 Description

VeriFact is a real-time claim verification platform for online meetings. It helps presenters and participants make sure every claim has evidence. As a native Zoom application, it transcribes speech, detects factual claims, and checks them against uploaded relevant documents using a Retrieval-Augmented Generation (RAG) pipeline. Verified statements show color-coded indicators and clickable citations. Additionally, a wakeable Q&A agent provides permitted attendees with instant, source-based answers, serving as a secondary workflow that complements claim verification. By bringing together claim detection, secure document management and real-time fact checking, VeriFact reduces misinformation risks and helps organizations hold reliable, data-driven meetings.

1.2 High Level System Architecture & Components of Proposed Solution

1.2.1. Architecture Diagram

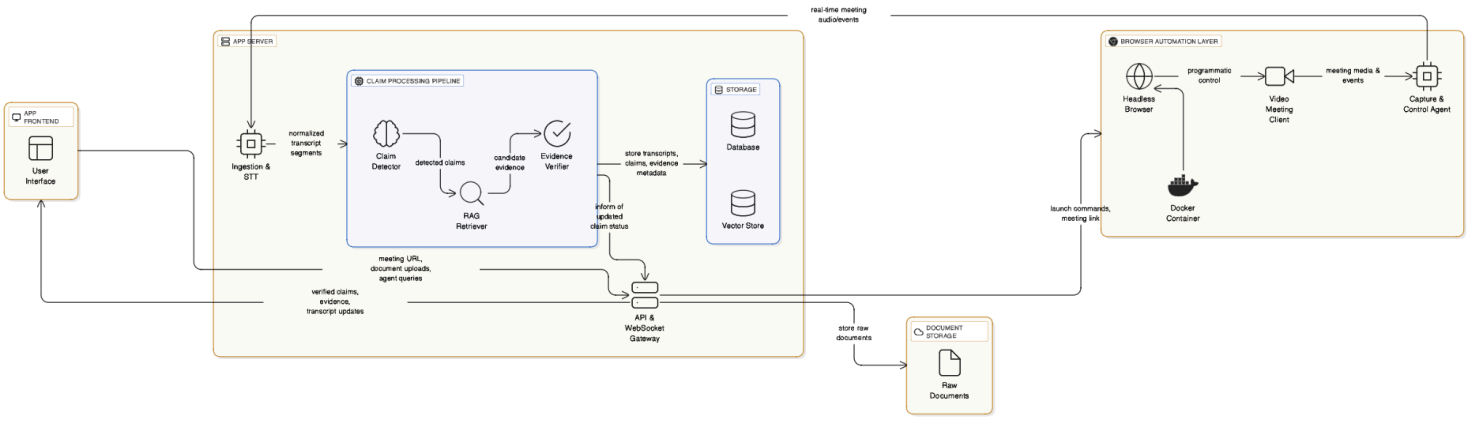

Figure 1: High Level System Architecture Diagram of Verifact.

1.2.1. App (Frontend) Layer

Purpose: This layer is the interface will be in an iFrame on Zoom's app interface, allowing participants to perform operations both in and out of meetings. Examples of actions include viewing previous meeting records, including verification sessions, uploading documents, and even watching evidence appear in real time during a meeting.

Modules:

- User Interface: A React app that allows for:
    - Initiating and stopping verification settings in a meeting
    - Uploading documents (either temporary or permanent)
    - Renders live transcripts, detected claims and linked evidence

Data Flow:

- Frontend to App Server:
    - Sends the Meeting Details (including url) and session configuration
    - Uploads documents and its metadata (linked meeting, uploaded by)
- App Server to Frontend:
    - Using a websocket connection, streams transcript segments
    - Streams claim detection verdicts and other verification details
    - Provides answers to Q&A agent queries in the transcript log

1.2.2. Application (Server) Layer

Purpose: The server layer turns raw audio into transcripts using STT model, detects and verifies claims, coordinates RAG requests and sends verification results to the UI via WebSockets. Sequentially, it is responsible for transcribing, detecting claims, retrieving evidence through the RAG pipeline, verifying claims and returning results.

Modules:

- Ingestion & STT:
    - Receives audio streams from the zoom bot and converts into a structured text, normalizing it into time-stamped segments
- Claim Detection Module
    - Identifies which sentences may contain factual claims.
- RAG Retrieval Module
    - Fetches the top-k relevant evidence for a detected claim, queried from the Vector Store
- Evidence Verification Module
    - Classifies claims using retrieved evidence, with a verdict and optional evidence according to verdict status
- Storage Module
    - Stores transcripts, detected claims, verification results in a metadata store submodule and stores both temporary and permanent document embeddings in a vector database. In addition, stores Zoom user entries, meeting entries, and other required relevant data, in a generic SQL database.
- API & WebSocket Gateway Module
    - Handles all API calls from UI for document uploads, meeting URLs or agent queries.
    - Pushes updates to the UI for verification and transcription results.
    - Issues instructions for the browser automation layer to follow

Data Flow:

- Browser Automation Layer to Ingestion & STT:
    - Propagate real-time audio and meeting events.
- Ingestion & STT to Claim Processing Pipeline:
    - Normalized transcript segments with speaker identification present.
- Claim Processing Pipeline to Storage:

    - Documents, transcripts, claims, classifications, and evidence links for claim.
- Claim Processing Pipeline & Ingestion & STT to API Gateway:
    - Securely push transcript updates and claim-status updates to clients.
- API Gateway to Browser Automation Layer:
    - Sends join/leave commands and meeting links to initiate/kill bot instances.
- API Gateway to Document Storage:
    - Stores raw uploaded documents and registers their indexes to be used by the RAG component

1.2.3. Document Storage Layer

Purpose: This layer simply exists as an abstraction for the service to be used to store all user-uploaded documents that have been vectorized (indexed), to be used strictly when an authorized user requests to download.

Modules:

- Raw Documents: S3-like storage solution where original files are kept at rest, encrypted using built in methodologies and grouped by tenant (organizations; collection of users) and by session scope (permanent or temporary).

Data Flow:

- App Server to Document Storage:
    - Encrypts uploaded files and stores them to the corresponding organization and proper session scope
- Document Storage to App Server:
    - Background jobs to index documents as we receive them; computing embeddings, storing to vector stores.
    - Related metadata and IDs written to the database

1.2.4 Browser Automation Layer

Purpose: This layer formulates the VeriFact meeting bot, which is at the core of this system. It joins the meeting as if it were a normal participant, captures audio and join/leave events for bookkeeping purposes. This is then relayed to the server in the form of a normalized audio stream to be transcripted. The implementation is discussed in detail in the subsequent sections.

Modules:

- Docker Container *(Docker Compose)*:
    - Contains a fully bootstrapped environment to run the headless browser, Zoom Meeting SDK, and the *Capture & Control bot*.
    - Every instance of the Zoom Meeting SDK allows for one meeting to be live, so this allows for horizontal scaling.
- Headless Browser:
    - Chromium based and controlled browser instance that loads the meeting url when spun up
    - Executes commands from the orchestrator (being the Application Server) to leave/join the meeting, and other possible functionalities.
- Video Meeting Client:
    - Zoom Meeting SDK which is ran through the headless browser
    - Joins a meeting as if it is a normal user, exposing an audio steam and join/leave events
- Capture & Control Agent*:*
    - Forwards utterances in shape of an audio stream to the Ingestion & STT module
    - Receives high-level commands from the Application Server, to perform operations such as: join a certain meeting, leave the current meeting, output this audio stream.

Data Flow:

- App Server to Headless Browser *(via API Gateway)*
    - Meeting link with initiation and command issuance
- Headless Browser to Video Meeting Client:
    - Programmatic control to launch the invite url, join the meeting, initiate recording to access audio stream (as required by Zoom Meeting SDK).
- Video Meeting Client to Capture & Control Agent:
    - Media streaming and meeting event propagation
- Capture & Control Agent to Ingestion & STT:
    - Live audio stream and events for transcription and downstream processing for visual representation

1.3 Constraints

1.3.1. Implementation Constraints

The most crucial implementation constraint for our app is the real-time nature of the system. VeriFact must listen to ongoing conversations, generate accurate transcriptions, detect factual claims, and retrieve evidence within a few seconds; hence each stage of the pipeline must operate under strict latency requirements. This restricts our choice of models and algorithms. Although large language models offer higher accuracy, their inference times are not suitable for our primary use case: live meeting environments. For this reason, we use ClaimBuster, web-based automated, live fact-checking tool developed by UTA [1] for claim detection and verify the claims FEVER verification [2].

We would like to experiment with the open-source ClaimBuster model for claim detection, but running this model efficiently requires access to a somewhat-powerful GPU. This limits us to the hardware that team members already possess; any additional fine-tuning or large-scale experimentation would require cloud GPUs, which may quickly become costly.

Our storage and retrieval architecture also faces privacy-driven restrictions: all documents and logs must be encrypted and stored securely, and temporary meeting documents cannot persist after the session ends.

Finally, the entire system must operate within the boundaries of the Zoom App Framework, which defines which UI elements, event triggers, and data streams we are allowed to use during a meeting [3].

1.3.2. Economic Constraints

We aim to keep the project financially lightweight, especially during the early development phase. Most components are built using open-source tools such as Whisper for transcription and ClaimBuster model for claim detection, which helps us avoid licensing fees [1]. However, the cost of GPU training and cloud hosting will be unavoidable. If we need to fine-tune models beyond what our local hardware supports, we will rely on pay-as-you-go cloud GPUs (e.g. Google Colab subscription), which must be used effectively to stay within budget.

Running a live system also requires backend servers for the automatic speech recognition (ASR) pipeline, claim detection, RAG retrieval, and Zoom bot orchestration. In early stages, these services will be used from existing sources (such as the available ClaimBuster API), or deployed on minimal cloud instances [1]. However the scalability of these servers remains a long-term concern.

For the scope of this course project, we will only implement the minimal infrastructure needed to demonstrate feasibility. We assume that future commercialization could justify the cost of scaling to multiple clients, but for now, resource usage must remain deliberately limited.

1.3.3. Ethical Constraints

Due to our core features that rely on processing live speech and user-uploaded documents, ethical constraints shape almost every design decision.

First, all meeting content, including audio streams, transcripts, and derived embeddings, must be handled with the highest standards to preserve our clients' privacy and confidentiality. We're committed to keeping the storage of your raw audio to a minimum. Although we acknowledge that in the early development stage. We might temporarily store some data in order to analyze performance and refine the accuracy of our models, only with explicit consent from users. This ensures that only necessary meeting data is stored on persistent storage, protecting the privacy of the meeting. Temporary documents will automatically be removed after the meeting from our database, while permanent documents will be stored according to privacy standards. This will remain under full user control, and can be removed whenever they wish. Additionally, participants will always be informed when the Zoom bot is active and what parts of the meeting it processes.

A second ethical consideration involves model bias and misclassification. Claim detection and verification are probabilistic processes. There is always a risk that the system may flag benign statements as claims, misclassify truthful statements as refuted, or fail to find evidence due to incomplete documents. Incorrect flags can create confusion or undermine user trust, especially in professional settings. To address this, VeriFact presents evidence excerpts alongside verdict labels and clearly communicates when confidence is low. The system is designed to support human decision-making, not override it.

Finally, VeriFact must not be used as a surveillance tool. The system is intended to support transparency, accuracy, and collaboration, not to monitor employees or penalize individuals for mistakes. The architecture therefore focuses exclusively on assisting users during the meeting itself. The app will include options for the visibility of alerts. Clients may choose for the alerts to only show when they are supporting, or visible only to the speaker or the meeting host, to avoid creating social pressure or harming the conversational dynamic.

1.4 Professional and Ethical Issues

1.4.1. Privacy and Consent

The most fundamental responsibility of the system is to respect the privacy of all meeting participants. As mentioned in 1.3.3, only necessary meeting data (such as transcripts) will be stored and all other data will **not be** stored, logged, or forwarded to external parties without explicit approval. Participants will also be made aware when the Zoom bot is active, what information it processes, and how their speech is being handled. These measures ensure that users remain in control of their data at all times.

1.4.2. Protection of User-Uploaded Documents

User-provided documents (policies, regulations, financial files, academic papers) often contain sensitive or proprietary information. The measures below, ensure that the factual verification process does not compromise confidentiality or expose sensitive information.

All uploaded documents must be encrypted using industry-standard protocols (e.g., AES-256 at rest, TLS in transit) [4]. Only authorized meeting participants and the verification pipeline will be allowed to access document embeddings or text.

Permanent documents remain encrypted and tied to the user's workspace. Temporary documents (uploaded only for the current meeting) will be automatically deleted after the session ends.No document should be shared across organizations without explicit consent.

Documents will be processed in secure cloud environments with no external data sharing, model training, or storage beyond the user's consent. Uploaded documents may never be used in secondary ways, such as to train models, improve datasets, or generate analytics.

1.4.3. Bias and Misclassification

Claim detection and verification models may reflect biases present in their training datasets (e.g., FEVER, Wikipedia-based evidence) [2]. Incorrectly flagging benign statements as

“check-worthy” or incorrectly classifying a claim as Supported/Refuted may mislead users. For this reason, our UI will emphasize that outputs are probabilistic, not authoritative, and evidence snippets will always be shown for transparency. The app will include options to control the visibility of alerts.

1.4.4. Transparency

Users will be informed about the system’s overall pipeline (transcription → claim detection → retrieval → verification). The models used in the pipeline and their limitations (e.g. probability bias, misclassification) should be communicated clearly. Users must understand that the system does not replace domain experts or formal audits.

1.4.5. No Manipulation or Surveillance Use-Cases

VeriFact must not be used for covert monitoring, employee surveillance, or disciplinary evaluation. The system must remain strictly within its intended use which is supporting accuracy, improving information quality, and enhancing transparency in meetings. Any deployment in sensitive contexts (legal, healthcare, government) also requires additional ethical review.

1.5 Standards

1.5.1 Software Engineering & Documentation Standards

- **IEEE 830 / ISO/IECIEEE 29148 (Requirements Engineering):** These standards guide how we structure our functional and non-functional requirements, ensuring they are complete, testable, and traceable throughout the system [5].
- **IEEE 1016 (Software Design Descriptions):** We use this standard when documenting the system architecture, ensuring that module descriptions, interfaces, and data flows follow a consistent and industry-recognized format [6].

- **UML 2.5.1 (Modeling):** All of our system diagrams, including use case, component, and sequence diagrams, follow UML notation to maintain clarity and standardization across design artifacts [7, 8].
- **IEEE 829 / ISO/IEC/IEEE 29119 (Software Test Documentation):** We are planning to structure our test plans , and reports using these standards so that each requirement is linked to repeatable and verifiable test cases [8].

1.5.2 Security and Privacy Standards

- **GDPR (Data Protection Regulation):** GDPR principles determine how we handle user consent, data retention, temporary document deletion, and encrypted processing of meeting content [9].
- **ISO/IEC 27001 (Information Security Management):** We apply the core controls of this standard to manage risks around document storage, user data, and backend access, creating a structured security environment [10].
- **ISO/IEC 27018 (Protection of PII in Clouds):** This standard guides our policies on data minimization, consent, and safe cloud storage of user documents and embeddings [11].
- **OWASP ASVS (Web Application Security Verification):** We reference ASVS for securing our APIs through proper authentication, input validation, and protection against common web vulnerabilities [12].

1.5.3 AI/ML Standards

- **ISO/IEC 23894 (AI Risk Management):** We follow this standard to identify and mitigate AI-related risks such as bias, misclassification, and incorrect evidence retrieval [13].
- **NIST AI Risk Management Framework:** This framework guides our transparency and reliability practices, including providing evidence snippets and confidence indicators to support human oversight [14].
- **ML Reproducibility Best Practices:** We ensure reproducible model training by versioning datasets, embedding pipelines, and model checkpoints [15].

- **Datasheets for Datasets:** We document dataset sources, limitations, and intended uses to maintain transparency about the evidence base our models rely on [16].

1.5.4 Cloud & Data Standards

- **TLS 1.2+ (Encrypted Communication):** All communication between the Zoom bot, backend services, and storage layers will use TLS to prevent interception of transcripts and documents.
- **AES-256 (Encryption at Rest):** Stored documents, embeddings, and metadata will be encrypted using AES-256 to ensure confidentiality [4].
- **S3-Compatible Object Storage Conventions:** We use S3 lifecycle rules and bucket policies to enforce document expiration for temporary files and controlled retention for permanent ones.

1.5.5 UI/UX and Accessibility Standards

- **ISO 9241-210 (Human-Centered Design):** We follow this human-centered design process to reduce cognitive load and ensure that private fact-checking alerts improve, rather than disrupt, meeting interactions [17].
- **Zoom App Framework UI Consistency Standards:** Our UI follows Zoom's design guidelines so that VeriFact integrates seamlessly with native Zoom controls and user expectations.

2. Design Requirements

2.1. Functional Requirements

2.1.1 Document Management (DM)

REQ-DM-1:

The system shall support two document storage modes: Permanent Documents and Temporary Documents.

REQ-DM-2:

The system shall maintain separate logical storage namespaces for permanent and temporary documents to ensure correct retrieval context.

REQ-DM-3:

The system shall allow users to upload one or more documents before a meeting and during an active meeting session.

REQ-DM-4:

The system shall validate file formats and reject unsupported formats with an appropriate error message.

REQ-DM-5:

Upon upload, the system shall extract text, generate vector embeddings, and store the document in an S3-like bucket and vector database.

REQ-DM-6:

The system shall restrict upload operations to users who have upload permissions. If a user without permission attempts to upload, the system shall display an “Insufficient Permissions” error.

REQ-DM-7:

The system shall allow importing of documents from external storage providers (e.g., Google Drive) using OAuth-based authentication.

REQ-DM-8:

The system shall index and retrieve external documents only if the user has explicit access rights.

REQ-DM-9:

When a user deletes a document, the system shall remove the document and all associated vector embeddings from all storage layers.

REQ-DM-10:

The system shall update document metadata when a stored file is replaced or modified by an authorized user.

2.1.2 Permission and Access Control (PAC)

REQ-PAC-1:

The system shall allow the meeting host to modify user permissions during the meeting.

REQ-PAC-2:

The system shall support the following permission types: Upload Documents, Delete Documents, View Evidence, Access Confidential Files, and to-be-determined additional permissions.

REQ-PAC-3:

The system shall notify participants when their permissions change.

REQ-PAC-4:

The system shall restrict evidence visibility based on user permissions.

REQ-PAC-5:

The system shall enforce organization-level access controls when retrieving from external storage repositories.

REQ-PAC-6:

The system shall hide refuting-evidence content (red claims) from users who do not have permission to access the underlying document.

2.1.3 In-Meeting Features (IMF)

2.1.3.1 Meeting Initialization

REQ-IMF-1:

When the user opens the VeriFact panel, the system shall display the real-time transcript, claim verification panel, document list, evidence window, and Q&A interface.

REQ-IMF-2:

The system shall activate audio capture, transcription, claim detection, and retrieval pipelines when the meeting begins.

REQ-IMF-3:

The system shall load both permanent and temporary documents selected for the meeting at initialization time.

2.1.3.2 Participant Management

REQ-IMF-4:

The system shall register new participants and enable audio capture with speaker identification.

REQ-IMF-5:

The system shall update participant roles dynamically in response to host permission changes.

2.1.3.3 Real-Time Transcription

REQ-IMF-6:

The system shall transcribe audio into text in real time.

REQ-IMF-7:

The system shall perform speaker attribution for all transcribed segments.

REQ-IMF-8:

The system shall display transcriptions immediately.

2.1.3.4 Real-Time Claim Detection

REQ-IMF-9:

The system shall analyze each sentence using a ClaimBuster-derived BERT classifier to detect factual claims.

REQ-IMF-10:

The system shall flag claims that exceed a configurable confidence threshold.

2.1.3.5 Real-Time Claim Verification

Evidence Retrieval

REQ-IMF-11:

The system shall retrieve evidence from permanent documents, temporary documents, and authorized external documents.

REQ-IMF-12:

The system shall perform vector similarity search and return the top-k relevant passages.

Claim Classification

REQ-IMF-13:

The system shall classify each claim into one of the following: SUPPORTS, REFUTES, or NOT-ENOUGH-INFO (NEI).

REQ-IMF-14:

The system shall compute a confidence score for each classification.

Visualization

REQ-IMF-15:

Supported claims shall be highlighted in green and shown to all authorized users.

REQ-IMF-16:

Refuted claims shall be highlighted in red, with evidence visible only to users who have the required access level.

REQ-IMF-17:

NEI claims shall be shown in gray and added to the manual review queue.

2.1.4 Q&A Agent (QA)

REQ-QA-1:

The system shall activate Q&A mode when the user says “Hey Agent” or when text is submitted via the input panel.

REQ-QA-2:

The system shall retrieve evidence from all authorized sources (permanent, temporary, external).

REQ-QA-3:

The system shall present answers with citations including document names and page or section numbers.

2.1.5 Post-Meeting Review (PMR)

REQ-PMR-1:

The system shall present post-meeting transcripts including speaker identities, detected claims, classifications, and associated evidence.

REQ-PMR-2:

The system shall allow exporting the meeting summary in PDF, Markdown, or plain text formats.

2.2 Non-Functional Requirements

2.2.1 Usability Requirements (USE)

REQ-USE-1:

The system shall display evidence cards containing the source document name, page number, and a relevant excerpt (maximum 150 characters) without requiring additional user interaction.

REQ-USE-2:

The system shall auto-save document upload progress and allow users to resume interrupted uploads.

REQ-USE-3:

The system shall provide access to permission management for document uploads within no more than two user interactions from the main panel.

REQ-USE-4:

If a query requires more than 8 seconds to process, the system shall provide a temporary acknowledgment message (e.g., “Searching for information…”).

2.2.2 Reliability Requirements (REL)

REQ-REL-1:

The system shall enforce document retention policies by automatically archiving documents older than 365 days to cold storage with a retrieval time of no more than 24 hours. Permanent documents are held to this restriction, whereas temporary documents will be deleted after the meeting ends.

REQ-REL-2:

If the claim classification service becomes unavailable, the system shall buffer up to 50 unprocessed claims and process them when service is restored.

REQ-REL-3:

If the Q&A bot fails to respond within 10 seconds, the system shall notify the user with "Agent is temporarily unavailable" and log the event for diagnostic purposes.

REQ-REL-4:

The speaker identification subsystem shall attribute speech to the correct participant ID with at least 95% accuracy when multiple speakers are present.

REQ-REL-5:

The RAG retrieval subsystem shall return at least one evidence snippet with cosine similarity > 0.6 for at least 90% of detected claims.

2.2.3 Performance Requirements (PERF)

REQ-PERF-1:

The system shall ensure that end-to-end latency from speech utterance to claim verification output does not exceed 5 seconds for 95% of claims, distributed as:

- Speech-to-text: ≤ 2 seconds
- Claim classification: ≤ 0.5 seconds
- RAG retrieval: ≤ 1.5 seconds
- Evidence classification: ≤ 1 second

REQ-PERF-2:

Document upload and vectorization shall support parallel processing and achieve the following completion times:

- Small documents (≤10 pages): ≤ 15 seconds
  Medium documents (11-50 pages): ≤ 45 seconds
- Large documents (51-100 pages): ≤ 90 seconds
  Processing throughput shall average at least 1.1 pages per second.

REQ-PERF-3:

The system shall support concurrent document uploads by up to 3 users within the same meeting without degradation in processing time.

REQ-PERF-4:

The Q&A bot shall acknowledge user queries within 1 second of activation phrase detection.

REQ-PERF-5:

The claim classification service shall process at least 20 claims per second under load.

2.2.4 Supportability Requirements (SUP)

REQ-SUP-1:

The RAG retrieval configuration parameters (e.g., top-k, similarity threshold) shall be adjustable through environment variables or configuration files.

REQ-SUP-2:

Database schema migrations shall be backward-compatible for at least one version to support safe rollbacks.

2.2.5 Scalability Requirements (SCA)

REQ-SCA-1:

The system shall support up to 50 concurrent meetings without measurable degradation in per-meeting performance metrics.

REQ-SCA-2:

Each meeting shall support up to 50 participants with claim verification services active for the host and designated presenters.

REQ-SCA-3:

The system shall support organizations with up to 1,000 registered users accessing the shared document repository.

REQ-SCA-4:

The vector database shall scale to 1 million embedded document chunks while maintaining query response time below 150 ms.

REQ-SCA-5:

The RAG pipeline shall support parallel processing of up to 10 concurrent claims within a single meeting with latency overhead not exceeding 20%.

3. Feasibility Discussion

3.1. Market & Competitive Analysis

3.1.1. Current Market Landscape

Market conditions are very optimal right now for VeriFact, because there is a strong demand and the technology is now ready to support it. Today, remote and hybrid work are completely integrated in organizational operations. The result of this is that hundreds of million people are using Zoom, Microsoft Teams and Google Meet every day for real-time communication. Every single day, millions or billions of critical decisions are made within these platforms and this means the accuracy and reliability of the information shared in meetings has a direct effect on how well those organizations perform.

At the same time, there is a growing concern related to the risk of misinformation and factual errors which is especially serious in high-risk industries like finance, healthcare, law and other regulated fields. In these areas, even one wrong claim during a meeting can lead to serious financial, legal or reputational damage.

As a result, more and more enterprises are choosing to invest in tools that reduce information risk. They are actively looking for solutions where they can verify facts and conversations in real time. This shift creates a well-designed opportunity for VeriFact since unlike basic note-taking tools, VeriFact is designed to act as a real-time truth layer inside the meeting itself.

3.1.2. Competitive Landscape

Other parts of the AI meeting assistant market are also very crowded. However, most of those tools are still focusing on recording and summarizing what people say, rather than checking whether those statements are actually correct. For example, tools like Otter.ai, Fireflies.ai, MeetGeek, Granola AI and Webex AI Companion mainly support live transcription, post-meeting summaries, and action item extraction. But, they don't verify claims against trusted documents in real time.

Within this space, two competitors are especially threatful for VeriFact.

- First one of them is Cluely, which presents itself as a real-time AI assistant that quietly supports a single person during a meeting. It listens the call, reads what is on the screen and then provides suggested real-time answers and talking points. Cluely's main strength is real-time coaching which is particularly useful in stressful situations such as sales calls, interviews or negotiations. Cluely also has a very different brand perception among its users due to its unethical behavior.

- Second one of them is Glean, which positions itself as an enterprise "Work AI" platform. Glean is among the top AI agent firms in the U.S. which has the highest valuation, and they are very huge. They have many things, not just Zoom bot integration. However, since we're only concerned about that, it can be said that our secondary workflow is fully covered by Glean, which gives them an opportunity to react quickly in the case of exponential growth of VeriFact. But what is missing in Glean is the real-time verification workflow because Glean does not have any features to do real-time verification among given documents.

Surely, beyond these two, there are several more competitors however they can be categorized as indirect; such as Granola AI, Fireflies.ai, Otter.ai and Webex AI Companion. This categorization is suitable since these tools are concentrated on transcription, summarization and meeting analytics. And none of them focused on either a triggerable Q&A agent or real-time fact verification.

3.1.3. Differentiation and Unique Value Proposition

Our big strength is that we're turning the AI meeting assistant into an active fact-checking partner. Whenever a participant makes a factual claim, for example, "Our Q2 revenue grew by 35% in Europe," our bot detects the claim in real time and compares the claim against relevant possible evidence, after classifying it as Supported, Refuted, or Not Enough Information. Then we display the result in the UI according to the given permissions.

Second differentiator of VeriFact is the dual document model, which means our tool can work with both permanent documents such as policies, regulations, core financial reports, etc., and temporary documents such as a specific proposal, quarterly slide deck, etc., that are uploaded during a single session. This design lets users keep a persistent knowledge base while still adding meeting-specific context.

Third differentiator is the fact that VeriFact includes an in-meeting Q&A agent that everyone in the call can use (but they may also not be able to call based on given permissions). So, participants can ask focused questions such as "What is the latest version of our retention policy?" or "What was the approved budget for this project last quarter?" and receive answers that are grounded in the uploaded documents with given clear citations.

Finally, VeriFact is intentionally designed as domain-agnostic. So, independent of the domain, we can detect claims and verify them against the documents uploaded to the system. This means everybody can bring their own domain-specific documents, and VeriFact supplies the verification layer on top.

3.1.4. Market Opportunity and Adoption Readiness

According to our research, there are several clear signals suggesting that the market is ready for VeriFact's approach. First, the AI workplace tools market is already large and still growing. At the same time, remote and hybrid work have become a lasting norm rather than a short term fix. Since the perception around meetings are changing toward physical to online the volume of factual claims inside those meetings will remain high.

Second, there is strong cultural and technological readiness. Many users are already comfortable with AI note-takers; they are even capable of using copilots, etc. On the technical side, modern real-time transcription APIs have emerged, and retrieval-augmented generation pipelines are now more robust than ever which enables the possibility of low-latency claim verification.

Moreover, emerging regulations are increasingly favoring tools like VeriFact, especially due to the growing concerns about misinformation. All domains are affected by this, whether it be finance, healthcare, law, or public administration. They are now under growing pressure to prove that the information used in decisions is well-grounded and traceable. A system that can clearly show "this claim was supported or refuted by this paragraph in this document" inside the meeting can shorten audit trails and reduce compliance risk.

Given these, VeriFact can realistically start with early adopters such as mid-sized financial teams, compliance and regulatory groups, consulting and advisory firms, research labs, and university research groups. These users share a strong common need for accuracy and evidence in recurring meetings, and they may also view large enterprise platforms like Glean as too heavy or too costly, and "stealth coaching" tools like Cluely as ethically misaligned with their open and collaborative ways of working.

3.2. Academic Analysis

3.2.1 System Architecture and Zoom Integration Feasibility

When we consider VeriFact architecturally, it is thoughtfully designed using technologies that are already reliable and well-tested for real-world use. At the heart of the system we have a Zoom bot. This bot can either join meetings when scheduled or enter on demand. Once it's in the meeting, it captures real-time audio from participants, shows a side panel in Zoom for live feedback, and enables further features.

For example, through this bot, users can upload and manage documents, get live transcriptions, and see which claims are being made during the meeting. Even more importantly, they can view

whether their claims are supported, refuted, or not evaluated based on given permissions. They can also interact with a Q&A agent without ever leaving the Zoom UI.

What makes this approach great is that Zoom already allows bots like this and supports side panes within in-meeting apps. In fact, Zoom's ability to provide a different audio track for each speaker is a big plus. Because of this facility, it’s so much easier to figure out who is saying what. That way, any detected claims can be clearly linked to the correct speaker.

Once the audio is captured, it is sent to a STT system such as Whisper. This system rapidly turns the words that are spoken into text in real time. That live transcript is then passed into VeriFact’s claim detection process.

There are three main parts to the interface:

- A special screen for uploading and managing files, which users can access from the Zoom app entry point;
- A real-time verification pane which displays the live transcript, emphasizes important claims, and shows whether each one is supported, refuted, or NEI;
- A Q&A area where users can ask questions using text or voice triggered by saying something like “Hey Agent”.

In summary, the architecture of VeriFact is already proven by Zoom’s SDK that previous/already-existing are built upon. The components include the Zoom bot, a speech-to-text service, backend tools for information retrieval and claim verification, and a web-based interface at its core. Which means, there’s no need to rely on untested or experimental technologies.

3.2.2 Machine Learning Feasibility

In the machine learning side we have two components which are claim detection and claim verification. Both of those aspects are supported by research and public datasets.

When it comes to claim verification, VeriFact uses a fine-tuned BERT-based model trained on the FEVER dataset, which includes over 185,000 labeled claims. Also, thanks to the research by

The University of Texas Arlington, we have an open source model called ClaimBuster which is fine-tuned for live-fact detection with approximately 110M parameters and modest hardware requirements.

The claim verification part follows a Retrieval-Augmented Generation (RAG) pipeline:

- Retriever: Converts text chunks into embeddings stored in a vector database; optionally enhanced with a graph DB like Neo4j (at the first stage we're planning to only work with vector databases to see if that's enough).

- Classifier: Evaluates claims against retrieved evidence, returning a verdict (Supported, Refuted or Not Enough Information [NEI]) along with citations, all happening in real time.

Our research shows that this architecture is feasible because RAG is now widely used in production systems, and FEVER-trained models adapt well to user-provided documents. The most important thing our team has to focus on is the low-latency target that VeriFact promises within this architecture, because the meeting happens nearly in real time. Which means even 3-4 seconds of latency might be cumbersome. So, there is ongoing research within our team about this issue.

Our model's performance is measured using well-used, common metrics: precision, recall, retrieval accuracy, and classification scores. These metrics guide our team's continuous improvement efforts. In summary, VeriFact's ML approach uses well-tested techniques to a novel context which makes it both technically ambitious and practically doable.

3.2.3 Cost, Performance, and Scalability

If we consider the resources needed to get this project going, we have designed VeriFact to be both affordable for small teams and scalable as adoption grows. For ML training, a single GPU with 12 GB VRAM (e.g., RTX 4080) is sufficient to fine-tune a BERT model on the FEVER dataset, which keeps early costs minimal since two of our teammates have this VRAM capability

and can host the training locally for PoC purposes. Surely, when the application is deployed, we will need to acquire our own robust servers that can effectively serve our customers.

Speaking of deployment, it will be done in a modular sense: Zoom bot, STT, retriever, classifier, and Q&A agent all run as containerized microservices that can scale independently. We expect our major cost drivers to include ASR inference, embedding/retrieval, and claim verification. However, these can be optimized by caching frequent document embeddings, filtering for high "check-worthiness," and using lightweight models for live use.

3.2.4 Privacy, Security, and Compliance Feasibility

Since VeriFact handles sensitive meeting content and internal documents, privacy and compliance issues are very important to us and it should be embedded everywhere in the architecture. Meaning, all data incoming and outgoing to our system will be encrypted, and users will have full control over their documents and transcripts at any time. Access will be governed by scoped permissions which will ensure only authorized users to be able to view or query specific materials.

We have to meet the standards like **GDPR** and **SOC-2** by:

- Minimizing data retention,
- Logging all access and actions for auditability,
- Ensuring user data is never used for training without explicit consent.

VeriFact is planned to run on cloud infrastructure with built-in compliance features like encrypted storage, role-based access, which improves reliability and trust factors.

4. Glossary

API (Application Programming Interface)

Interfaces that allow different system components to communicate. In this project, APIs connect the Zoom bot, the transcription service, and the claim-detection pipeline.

ASR (Automatic Speech Recognition)

The technology used to convert spoken audio into text. All later stages of the system rely on this transcription.

Claim Detection Model

A machine learning model that determines whether a sentence should be considered a claim. Our system uses a lightweight local model to perform this analysis in real time.

Confidence Score

A probability value that reflects the model's certainty in its prediction. This score is used to filter out low-confidence outputs.

Client Application

The interface presented to the user. It displays transcripts, identified claims, and retrieved evidence.

Data Retention Policy

Rules that specify how long audio, transcripts, and embeddings remain available in memory.

Embedding

A numerical vector that represents the meaning of text. These vectors help the system perform semantic search during evidence retrieval.

Encryption (AES-256)

An encryption algorithm used to encrypt stored documents, embeddings, and metadata at rest, where stored (e.g., in S3-compatible object storage).

Evidence Retrieval Module

The part of the system that searches a curated dataset to locate information relevant to a detected claim. It relies on embeddings and similarity search.

GDPR (General Data Protection Regulation)

A data-protection framework that influences how the system handles consent, privacy, and deletion of temporary data.

In-Memory Buffer

A temporary storage area used for audio chunks and intermediate model outputs. These buffers are removed immediately after processing.

Latency Budget

The maximum allowed delay between the moment someone speaks in Zoom and the moment the system displays a result. This limit informs decisions about model size and streaming approaches.

Non-Functional Requirements (NFRs)

Requirements that describe the quality attributes of the system, including privacy, responsiveness, usability, and reliability.

Pipeline

The full sequence of operations that data passes through. This includes audio streaming, transcription, segmentation, claim detection, retrieval, scoring, and presentation in the user interface.

RAG (Retrieval-Augmented Generation)

A method that enriches model outputs with information retrieved from a knowledge source. In this project, retrieval is used to find relevant evidence for each detected claim.

Precision and Recall

Metrics used to evaluate how effectively the claim-detection model identifies true claims and avoids false detections.

Streaming Transcription

The practice of processing audio continuously without waiting for full sentences to finish. This is necessary for real-time feedback.

Thresholding

A rule that marks a sentence as a claim only when its confidence score exceeds a specified value.

Tokenization

A preprocessing step that divides text into smaller units before it is sent to the model.

Zoom Bot

An automated participant in a Zoom meeting that receives audio, processes it, and sends results to the user interface.

Zoom SDK (Software Development Kit)

The toolkit that enables integration with Zoom and access to audio streams and meeting events [3].

5. References

[1] “Automated live fact-checking,” ClaimBuster, https://idir.uta.edu/claimbuster/ (accessed Nov. 27, 2025).

[2] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, “Fever dataset,” Fact Extraction and VERification, https://fever.ai/dataset/fever.html (accessed Nov. 27, 2025).

[3]“Class ZoomSdk,” ZoomSdk | @zoom/appssdk - v0.16.36, https://appssdk.zoom.us/classes/ZoomSdk.ZoomSdk.html (accessed Nov. 27, 2025).

[4] National Institute of Standards and Technology (NIST), “Advanced Encryption Standard (AES),” FIPS PUB 197, Nov. 2001.

[5] IEEE Standard 830-1998, *IEEE Recommended Practice for Software Requirements Specifications*, IEEE, 1998.

[6] ISO/IEC/IEEE 29148:2018, *Systems and Software Engineering — Life Cycle Processes — Requirements Engineering*, 2018.

[7] IEEE Standard 1016-2009, *IEEE Standard for Information Technology — Systems Design — Software Design Descriptions*, IEEE, 2009.

[8] Object Management Group (OMG), *Unified Modeling Language (UML) Specification, Version 2.5.1*, Dec. 2017. [Online]. Available: https://www.omg.org/spec/UML/2.5.1/

[9] IEEE Standard 829-2008, *IEEE Standard for Software and System Test Documentation*, IEEE, 2008.

[10] ISO/IEC/IEEE 29119-1:2013, *Software and Systems Engineering — Software Testing — Part 1: Concepts and Definitions*, 2013.

[11] ISO/IEC 27018:2019, *Information Technology — Security Techniques — Code of Practice for Protection of Personally Identifiable Information (PII) in Public Clouds Acting as PII Processors*, 2019.

[12] OWASP Foundation, *OWASP Application Security Verification Standard (ASVS) Version 4.0.3*, OWASP, 2021. [Online]. Available: https://owasp.org/ASVS/

[13] ISO/IEC 23894:2023, *Information Technology — Artificial Intelligence — Guidance on Risk Management*, 2023.

[14] National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, NIST, Jan. 2023.

[15] P. Pineau *et al.*, "Improving reproducibility in machine learning research," *arXiv preprint arXiv:2003.12206*, 2020.

[16] T. Gebru *et al.*, "Datasheets for datasets," *Communications of the ACM*, vol. 64, no. 12, pp. 86–92, 2021.

[17] ISO, *ISO 9241-210:2019 — Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems*. International Organization for Standardization, Geneva, Switzerland, 2019.