**<Team D>**

**Software Requirements Specification For LLM-Based Cooperative Editor**

**Version 1.0**

| LLM-Based Cooperative Editor | Version: 1.0 |
| --- | --- |
| Software Requirements Specification | Date: 17/Mar/25 |
| <document identifier> | |

**Revision History**

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 17/Mar/25 | 1.0 | First Version of SRS for LLM-Based Cooperative Editor | Diallo, Alhassana  Mamun, Md  Moussa, Ezzeldin A  Ojilere, Lesley  Wu, Cuiwen |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Confidential © Page 2

| LLM-Based Cooperative Editor | Version: 1.0 |
| --- | --- |
| Software Requirements Specification | Date: 17/Mar/25 |
| <document identifier> | |

**Table of Contents**

1. Introduction 4

| 1.1 | Purpose | 4 |
| --- | --- | --- |
| 1.2 | Scope | 4 |
| 1.3 | Definitions, Acronyms, and Abbreviations | 4 |
| 1.4 | References | 4 |
| 1.5 | Overview | 4 |

2. Overall Description 4

2.1 Use-Case Model Survey 5

2.2 Assumptions and Dependencies 5

3. Specific Requirements 5

3.1 Use-Case Reports 5

3.2 Supplementary Requirements 5

4. Supporting Information 5

| LLM-Based Cooperative Editor | Version: 1.0 |
| --- | --- |
| Software Requirements Specification | Date: 17/Mar/25 |
| <document identifier> | |

**Software Requirements Specification**

**1. Introduction**

This SRS document provides a complete description of the LLM-Based Cooperative Editor System. The system leverages a large language model (LLM) to assist users in editing their texts interactively. The document details the system’s functionalities, user roles, interface requirements, and constraints.

**1.1 Purpose**

The purpose of this SRS is to fully describe the external behavior of the LLM-Based Cooperative Editor. It details both functional and non-functional requirements, design constraints, and user interactions. Key objectives include:

* Enabling users to submit texts for correction via an LLM or self-editing.
* Managing different user roles (free, paid, and super users) with varying privileges.
* Implementing a token-based system to manage usage and enforce limitations.
* Supporting collaboration and complaint management among paid users.

**1.2 Scope**

The LLM-based Cooperative Editor is an application featuring a graphical user interface (GUI) that supports multiple user types:

* **Free Users:** Limited text submissions (max 20 words per submission, with a 3-minute login cooldown).
* **Paid Users:** Extended text submissions governed by token availability, file saving, collaboration, and usage statistics.
* **Super Users:** Administrative users who handle sign-up approval, complaint resolution, and user moderation (suspending, fining, or terminating accounts).

**1.3 Definitions, Acronyms, and Abbreviations**

* **LLM:** Large Language Model.
* **GUI:** Graphical User Interface.
* **Token:** A unit of currency used for text submissions, corrections, and other system operations.
* **Free/Paid/Super Users:** The different classes of users with defined privileges.
* **Blacklist:** A list of words that are disallowed in user submissions.

**1.4 References**

The following documents and resources were referenced during the development of this Software Requirements Specification (SRS):

* **Project Requirements Document:**
  + Title: LLM-Based Cooperative Editor Project Requirements
  + Date: 17/Mar/25
  + Author: Provided by course instructor
  + Description: Outlines the functional and non-functional requirements, user roles, and system capabilities.
* **Sample Specification Document:**
  + Title: Software Requirements Specification Template
  + Date: 17/Mar/25
  + Author: Provided by course instructor
  + Description: A sample SRS document used as a reference for formatting and content structure.

**1.5 Overview**

*This SRS is organized as follows:*

* **Section 1: Introduction** – Provides an overview and outlines the purpose, scope, and definitions.
* **Section 2: Overall Description** – Covers the general factors affecting the system, including user characteristics and operational constraints.
* **Section 3: Specific Requirements** – Details of functional requirements (use-case reports) and supplementary specifications.
* **Section 4: Supporting Information** – Includes additional documentation such as the table of contents, indexes, and any relevant appendices.

| LLM-Based Cooperative Editor | Version: 1.0 |
| --- | --- |
| Software Requirements Specification | Date: 17/Mar/25 |
| <document identifier> | |

**2. Overall Description**

*Product Perspective:*

The LLM-based cooperative editor is an application that provides self-correction and LLM correction features designed to help users refine text through AI-driven corrections. The editor utilizes a local LLM from Hugging Face under Ollama. The system has hierarchical user access permissions and a token-based payment pattern. Free users, paid users, and super users all have different features and access levels.

**2.1 Use-Case Model Survey**

*Product Functions / Use Cases:*

* **Text Submission**: The user can input text either by typing in a text box or uploading a text file. The system will validate the text length and charge tokens.
* **Blacklist Management:** Content filtering through a blacklist of words. Words flagged by users and reviewed by super users.
* **Self-Correction and LLM Correction:** Users can choose between self-correction and LLM correction. The LLM will highlight corrections.
* **Collaboration:** Paid users can share and jointly edit their text files with other paid users.
* **Bonus System:** Users who submit an error-free text of more than 10 words will receive 3 bonus tokens.
* **Complaint Handling:** Paid users can submit complaints about collaborators, which will be reviewed by super users.

*User Characteristics / Actors:*

* **Free Users:** can submit text with up to 20 words and words for the blacklist.
* **Paid Users:** have access to token-based text submissions, self-correction, LLM correction, text sharing, and statistics viewing.
* **Super Users:** manage user accounts, handle complaints, maintain the blacklist, and have the authority to suspend/fine/terminate paid users.

*Constraints:*

* Free user access is limited to texts of up to 20 words with a 3-minute lockout period if this limit is exceeded.
* The system uses a local free LLM, which may have limitations in terms of accuracy for corrections compared to larger paid models.

**2.2 Assumptions and Dependencies**

*Main:*

* System assumes that there is a suitable free LLM that can perform accurate text correction.
* System assumes that users will follow the guidelines for submissions, blacklists, and complaints.
* System assumes that super users will have the expertise and time to properly handle blacklists and complaints.
* User trust depends on a secure and functional token management system.
* User satisfaction depends on the reliability and accuracy of LLM correction.

*Requirements Subsets:*

* **Functional Requirements:** text input, file upload, blacklist control, self-correction, LLM correction, collaboration for paid users, complaint handling, token management, and token-based payments processing.
* **Non-Functional Requirements:** a user-friendly GUI, fast response times of the LLM, and security for user data and transactions

| LLM-Based Cooperative Editor | Version: 1.0 |
| --- | --- |
| Software Requirements Specification | Date: 17/Mar/25 |
| <document identifier> | |

**3. Specific Requirements**

This section details all software requirements necessary for design and testing, broken down by use-case reports and supplementary specifications.

**3.1 Use-Case Reports**

*Use Case 1: Text Submission*

* **Actor:** Free User, Paid User
* **Description:** The user submits text for correction.
* **Preconditions:** User must be logged in.
* **Flow of Events:**
  + User inputs text manually or uploads a file.
  + System validates the text (length, blacklist compliance).
  + System deducts tokens (if applicable) and processes the text.
  + System returns the corrected text.
* **Exceptions:**
  + If a free user exceeds 20 words, the system rejects the submission and initiates a 3-minute cooldown.
  + If a paid user lacks tokens, the system prompts to purchase more.

*Use Case 2: Blacklist Management*

* **Actor:** Super User
* **Description:** Super Users manage the blacklist.
* **Preconditions:** Super User must be logged in.
* **Flow of Events:**
  + Super User reviews flagged words.
  + Super User adds, removes, or modifies blacklist entries.
  + Changes are saved to the database.
* **Exceptions:**
  + If system storage is down, changes are queued.

*Use Case 3: Self-Correction and LLM Correction*

* **Actor:** Free User, Paid User
* **Description:** Users choose between self-correction and LLM correction.
* **Preconditions:** User must submit valid text.
* **Flow of Events:**
  + User selects self-correction or LLM correction.
  + System highlights corrections.
  + The user applies changes.
* **Exceptions:**
  + If the LLM service is unavailable, users default to self-correction.

*Use Case 4: Collaboration*

* **Actor:** Paid User
* **Description:** Paid users share and jointly edit text.
* **Preconditions:** Both users must be paid users.
* **Flow of Events:**
  + User invites another paid user.
  + System verifies the invited user.
  + Both users edit the document in real time.
* **Exceptions:**
  + If one user disconnects, the remaining user retains control.

*Use Case 5: Complaint Handling*

* **Actor:** Paid User, Super User
* **Description:** Paid users submit complaints, reviewed by Super Users.
* **Preconditions:** User must submit a valid complaint.
* **Flow of Events:**
  + User submits a complaint.
  + System logs the complaint.
  + Super User reviews and resolves the complaint.
* **Exceptions:**
  + If a complaint is invalid, the system notifies the user.

**3.2 Supplementary Requirements**

*3.2.1 Functional Requirements*

* The system must validate text length and check for blacklisted words.
* The system must support text correction via self-correction and LLM.
* The system must handle token deduction, bonuses, and top-ups.
* The system must allow collaboration between paid users.
* The system must enable complaint submission and resolution.

*3.2.2 Non-Functional Requirements*

* The system must respond to user submissions within 2 seconds.
* The system must ensure data persistence for text submissions and corrections.
* The system must encrypt user data and transactions.
* The system must handle up to 10,000 simultaneous users.
* The system’s uptime must be 99.9% or higher.

**4. Supporting Information**

This section contains additional documentation and appendices to assist in the understanding and use of this SRS.

* **Table of Contents:** A detailed index of the SRS sections and subsections.
* **Index:** Keywords and terms used throughout the document.
* **Appendices:** May include:
  + User-case storyboards
  + User-interface prototypes