# Software Requirements Specification

**for**

# CADUCEUS

**Version 1.0 pending approval**

**Prepared by Harsh Avinash**

**Vellore Institute of Technology, Vellore**

**31st August 2023**

# Table of Contents



# Revision History

| Name | Date | Reason For Changes | Version |
|---|---|---|---|
| Init | 31-8-23 | Initi | 1.0 |
| | | | |

# 1. Introduction

## 1.1 Purpose

*This document meticulously articulates the precise software requirements and expectations for the Caduceus offline LLM solution. Its central aim is to bridge the understanding between stakeholders and the development team, facilitating clear comprehension, and reducing ambiguity throughout the development cycle.*

## 1.2 Document Conventions

*This document utilizes certain terminologies and acronyms that are pivotal to the understanding of the software solution:*

- ***LLM (Large Language Model):** Advanced machine learning models designed for understanding and generating human-like text based on provided input.*
- ***GUI (Graphical User Interface):** Visual elements such as windows, icons, or buttons used for interaction with a device.*

## 1.3 Intended Audience and Reading Suggestions

*This documentation is curated for a diverse set of professionals including but not limited to software developers, project managers, quality assurance testers, and primary stakeholders associated with the Caduceus project. For individuals new to the project, it's recommended to commence with the "Overall Description" segment, providing a holistic view of the Caduceus system's architecture and capabilities.*

## 1.4 Product Scope

*Caduceus emerges as a groundbreaking initiative to harness the power of AI without the incessant need for internet connectivity. It's not just a software; it's a conduit allowing users to interact with LLMs through a pendrive, ensuring swift, secure, and offline interactions. Beyond its fundamental capabilities, Caduceus vows to uphold the sanctity of user privacy, ensuring all interactions remain confined to the pendrive, minimizing potential external vulnerabilities.*

## 1.5 References

- *IEEE SRS Documentation Standards*
- *OpenAI's GPT Documentation*

- *Electron Framework Official Documentation*
- *Flask & Django Server Documentation*

# 2. Overall Description

## 2.1 Product Perspective

*Caduceus, though a standalone entity, functions as an ecosystem, harmoniously interweaving various components — from the writer application to the dedicated server. Each of these elements, though capable of independent operations, amalgamate to manifest the core vision of offline, secure AI interactions.*

## 2.2 Product Functions

*Caduceus is imbued with several salient features:*

- ***Pendrive Interactivity:*** *Format and prepare pendrives for LLM interactions.*
- ***Offline LLM Facilitation:*** *Engage with LLMs without reliance on internet connectivity.*
- ***Secure Updates:*** *Fetch necessary updates from a dedicated server while maintaining a fortress of security.*

## 2.3 User Classes and Characteristics

*Caduceus doesn't discriminate; it's designed for everyone. However, certain cohorts might find it especially beneficial:*

- ***Educators:*** *For those desiring enhanced, offline interactive learning.*
- ***Researchers:*** *For professionals necessitating dependable offline AI computations.*
- ***Business Professionals:*** *For those who require data analytics without the strings of online dependencies.*

## 2.4 Operating Environment

*Caduceus is envisaged to function across a multitude of platforms including Windows, MacOS, and Linux, thanks to the Electron framework's adaptability.*

## 2.5 Design and Implementation Constraints

- ***Hardware Dependency:*** *The software is contingent on specific pendrive hardware specifications.*
- ***Model Size:*** *Given the quantum of data, there might be constraints related to the LLM model size.*
- ***Data Transfer Rates:*** *While fetching updates, data transfer speeds might vary based on network connectivity.*

## 2.6 User Documentation

*Alongside this SRS, user manuals, FAQs, and video tutorials will be made available to ensure smooth onboarding and troubleshooting for users.*

## 2.7 Assumptions and Dependencies

*It's assumed that users possess basic computer literacy. Dependencies include but aren't restricted to the Electron framework for the application interface and Flask + Django for server-side operations.*

# 3. External Interface Requirements

## 3.1 User Interfaces

*The Caduceus software exhibits a polished, user-centric Graphical User Interface (GUI) crafted within the Electron framework. Users are welcomed by a seamless interface that intuitively guides them through:*

- ***Writer Application Interface:*** *It furnishes options for selecting desired LLMs, initializing pendrive formatting, and initiating server-triggered updates. Integrated progress bars and notifications ensure user awareness at every stage.*

## 3.2 Hardware Interfaces

*At the very heart of Caduceus lies its ability to effectively interact with USB pendrives:*

- ***Pendrive Interaction Protocol:*** *Caduceus, upon invocation, discerns the presence of a connected pendrive, establishes a secure connection, and then undertakes necessary formatting or LLM data transfers.*

### 3.3 Software Interfaces

*Caduceus leverages multiple software platforms to deliver a holistic experience:*

- ***Electron Framework:*** *Used for crafting the principal application GUI ensuring cross-platform functionality.*
- ***Flask + Django Backend:*** *These technologies empower the server-side operations, managing data transfers, updates, and ensuring security protocols.*

### 3.4 Communications Interfaces

*Secure HTTP/HTTPS protocols facilitate data updates and transfers. These encrypted channels ensure that data moving between Caduceus and the dedicated server remains secure, warding off potential vulnerabilities.*

## 4. System Features

### 4.1 Pendrive Formatting and LLM Integration

One of Caduceus' crowning features is its ability to metamorphose standard pendrives into repositories of LLM knowledge.

- **Functional Description:** Upon a pendrive's connection and user consent, Caduceus initialises it, making it compatible for LLM data storage and interactions. Post formatting, users can select and transfer desired LLM models onto the pendrive.

### 4.2 Secure Server-triggered Updates

Maintaining the LLM's up-to-date knowledge base is paramount. Caduceus ensures this through its server-triggered update mechanism.

- **Functional Description:** Users, upon accessing internet connectivity, can opt to update their LLM datasets. Caduceus interacts with its dedicated Flask + Django server, fetches updates, and securely transfers them onto the pendrive.

# 5. Other Nonfunctional Requirements

## 5.1 Performance Requirements

*Caduceus, despite its offline functionality, promises rapid response times, ensuring LLM interactions remain smooth. Update latency is minimized, with a keen focus on maximizing data transfer speeds during server interactions.*

## 5.2 Safety Requirements

*Any potential harm to the user's primary data storage, like their computer's hard drive, is stringently avoided. Caduceus confines its operations strictly to the connected pendrive.*

## 5.3 Security Requirements

*Every byte of data within Caduceus undergoes encryption, ensuring user interactions and data updates are shielded from potential threats. The sanctity of user privacy remains an unwavering commitment.*

## 5.4 Software Quality Attributes

*Caduceus is built on the pillars of reliability, modularity, and maintainability. Rigorous testing ensures minimal bugs, and the software's modular design facilitates future enhancements.*

## 5.5 Business Rules

*Caduceus remains compliant with all prevalent data protection regulations, including GDPR. User data is never shared, sold, or exploited.*

# 6. Other Requirements

*Caduceus is envisioned to eventually accommodate multiple languages, serving a global user base. Continuous improvements in terms of LLM model enhancements and GUI refinements are on the horizon.*

# Appendix A: Glossary

***LLM (Large Language Model):*** *These are state-of-the-art machine learning models designed for comprehending, generating, and predicting human-like text based on provided input. LLMs have found extensive use in Natural Language Processing tasks.*

***GUI (Graphical User Interface):*** *This refers to the interface through which users interact with electronic devices, such as computers, hand-held devices, office equipment, and other appliances.*

***Electron Framework:*** *A framework to build native applications with web technologies like JavaScript, HTML, and CSS.*

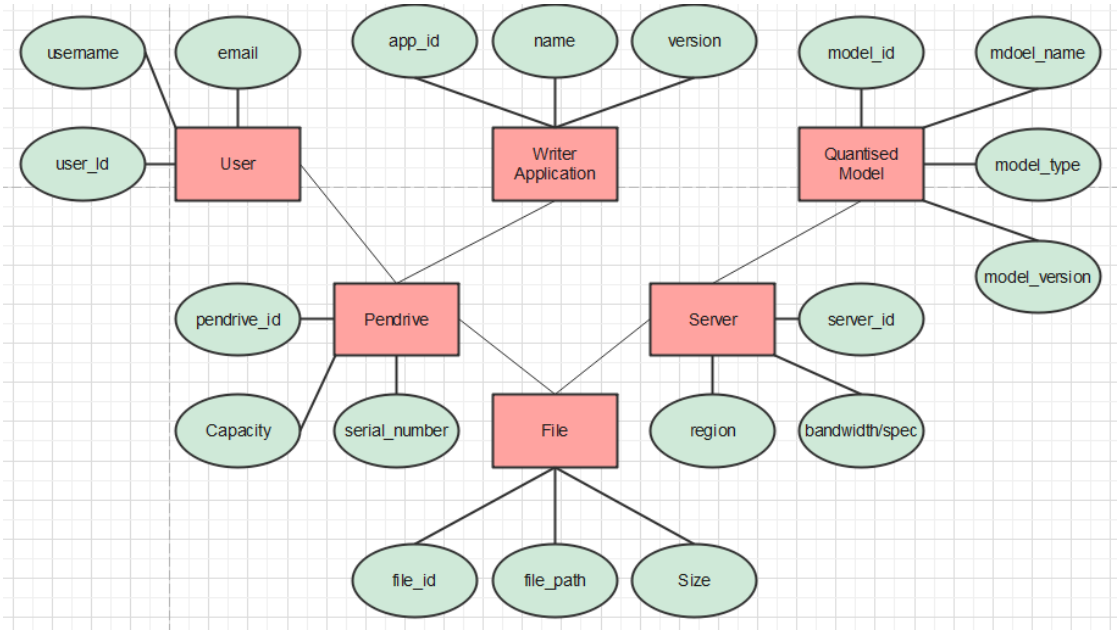
***Flask + Django:*** *Flask is a micro web framework written in Python. It does not include database abstraction, form validation, or any other components where pre-existing third-party libraries provide common functions. Django, on the other hand, is a high-level Python Web framework that encourages rapid design and a clean, pragmatic design.*

***HTTPS (HyperText Transfer Protocol Secure):*** *An extension of the HTTP. It is used for secure communication over a computer network, widely used on the Internet.*

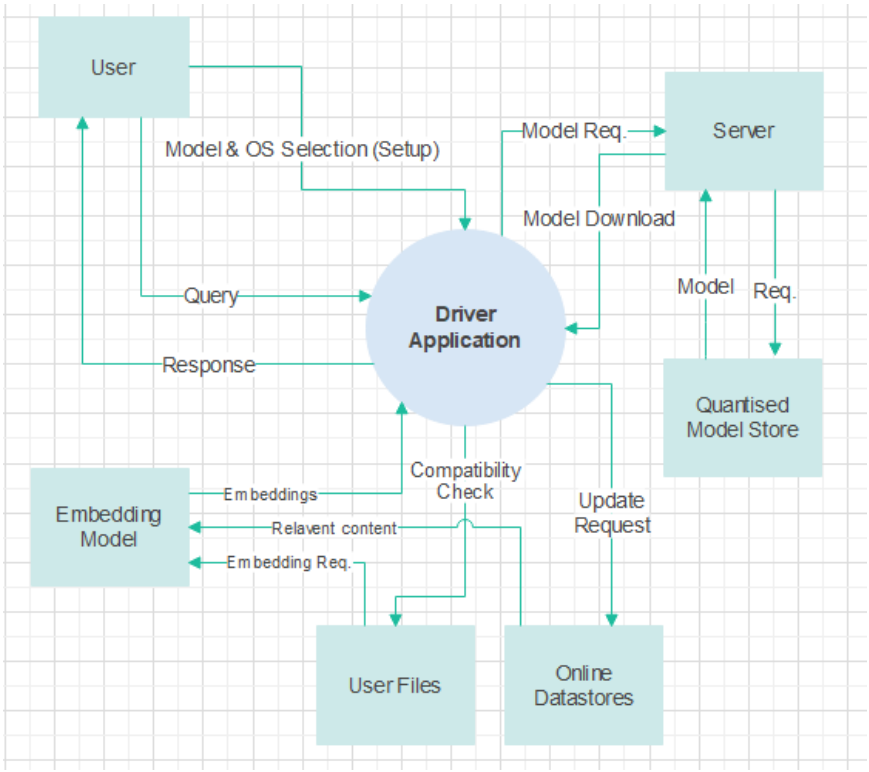
***GDPR (General Data Protection Regulation):*** *A regulation in EU law on data protection and privacy for all individuals within the European Union.*

# Appendix B: Analysis Models

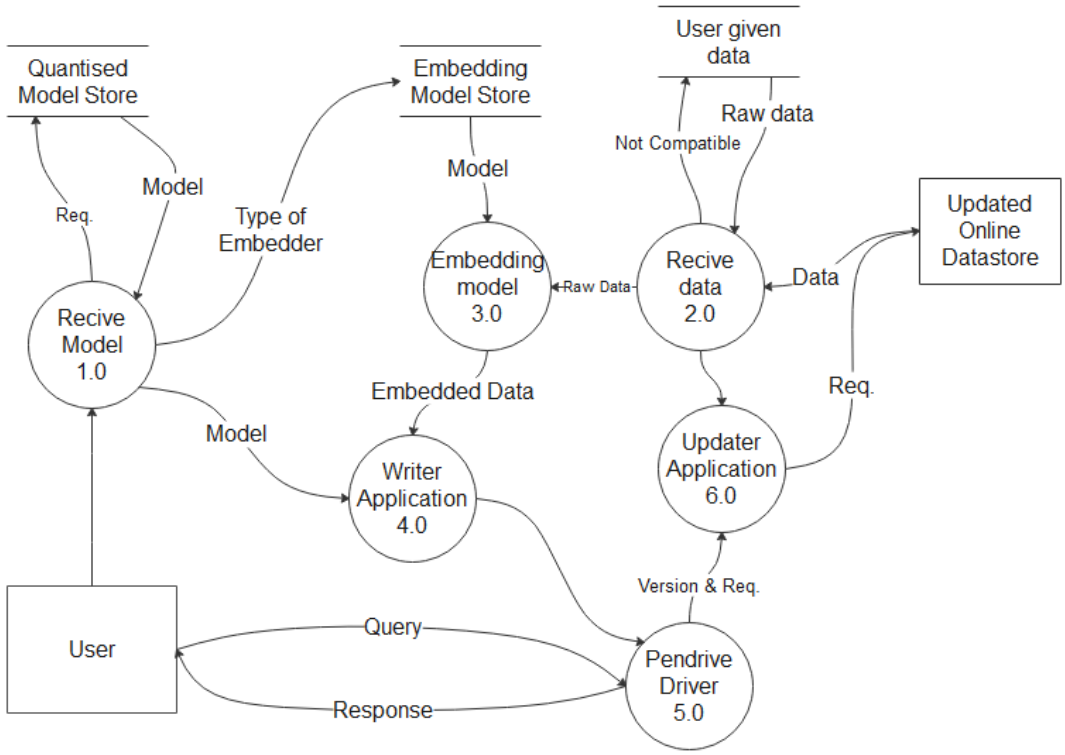
### *ER Diagram of the Project*



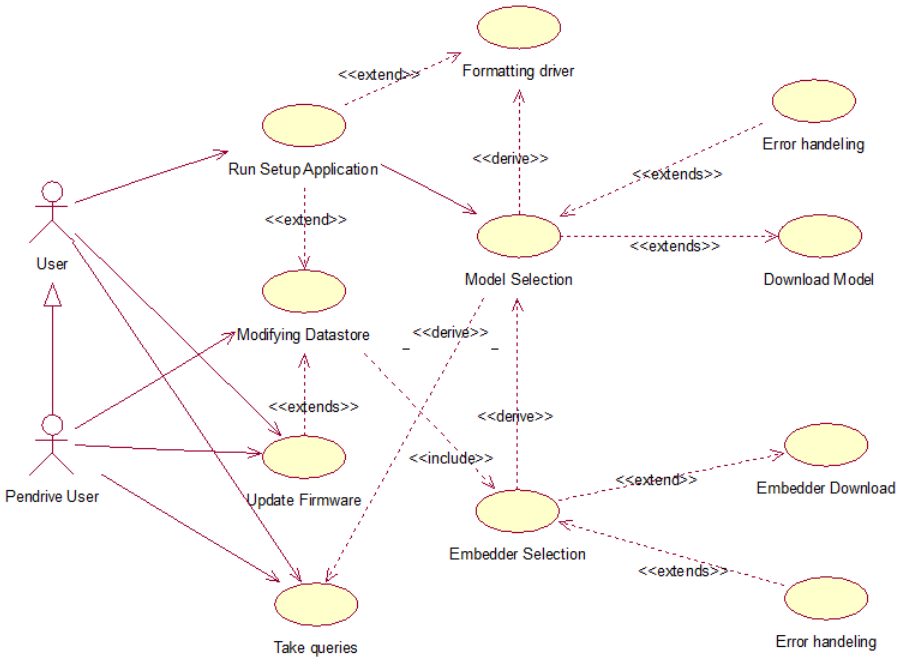

### *Context Flow Diagram*



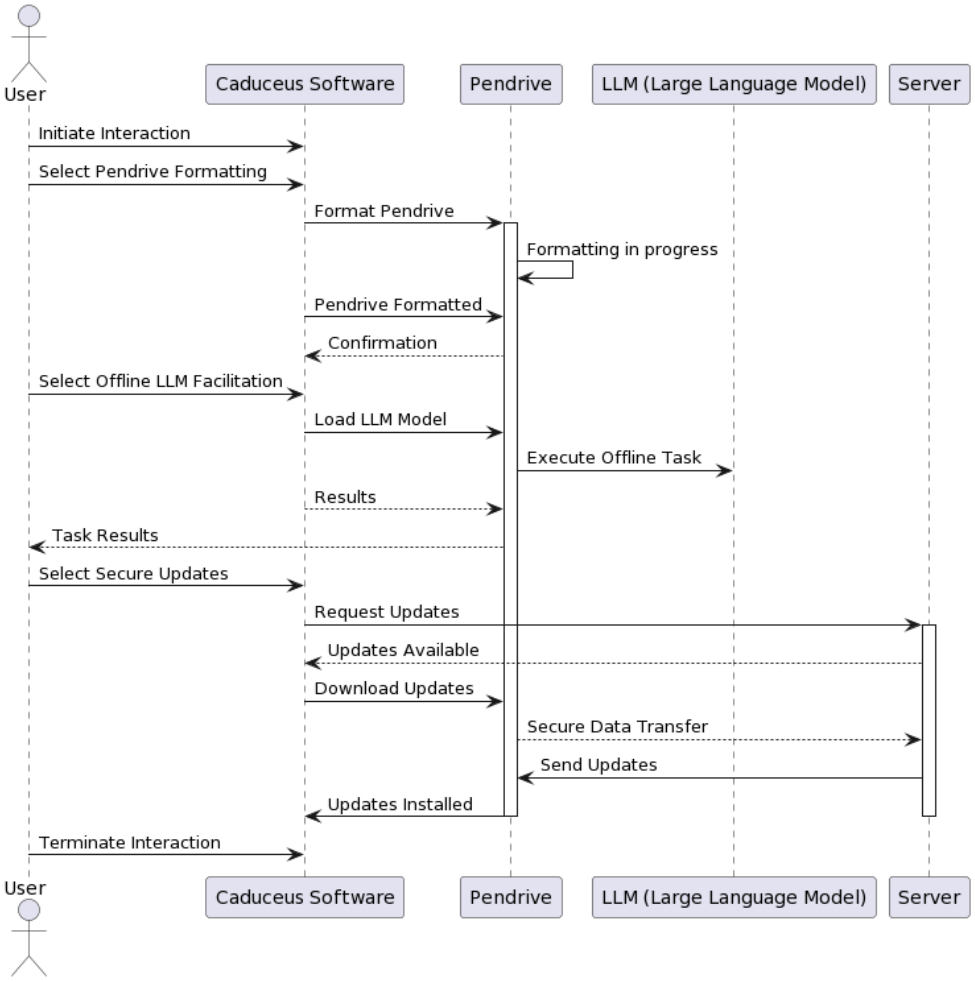
## *Data flow diagram*



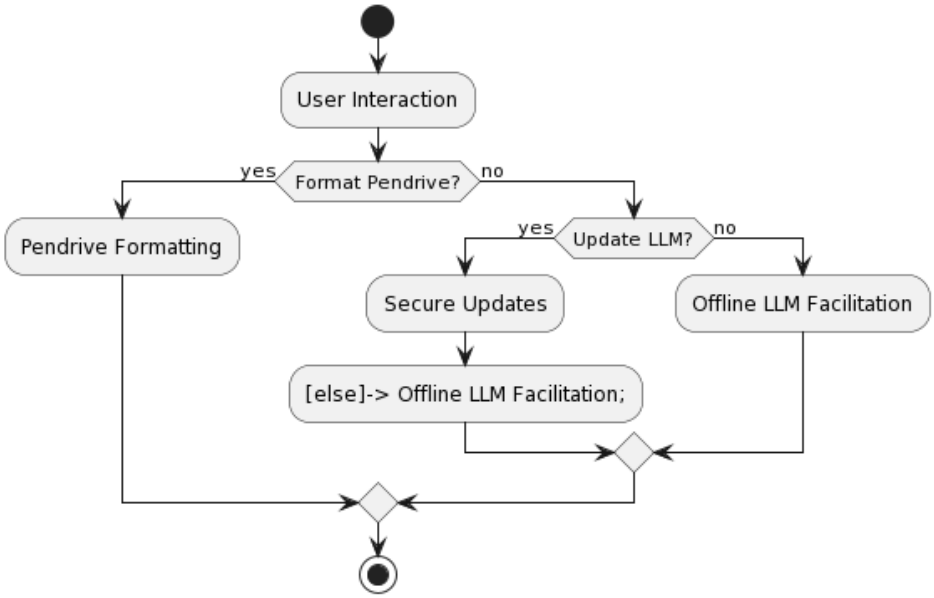

## *Use Case Diagram*



### *Sequence Diagram*




### *Activity Diagram*



### *Class Diagram*