L3M (Local Large Language Models)

by

Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, Carl Guillaume

Stevens.edu

June 3, 2025

L3M (Local Large Language Models)

Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, Carl Guillaume
Stevens.edu

This document provides the requirements and design details of the PROJECT. The following table (Table 1) is updated whenever the document is updated by any of the authors. Updates to this page are noted next to the update title.

Table 1: Document Update History

Date	**Updates**
05/12/2025	NK • Added new figures to zoom into 12: Figure 12.2, 12.3, 12.4, 12.5,12.5. • Added project download link in Chapter 16
04/25/2025	NK • Added Weekly Report 26 in Chapter 17, Section 17.1 • Added Project poster in 16, Figure: 16.3 • Added Bibliography [1]
04/18/2025	NK, BP • Added Weekly Report 25 in Chapter 17, Section **??**
04/11/2025	NK, RH • Added Weekly Report 24 in Chapter 17, Section 17.3
04/4/2025	NK, RH • Added Weekly Report 23 in Chapter 17, Section 17.4
03/28/2025	NK, BP • Added Weekly Report 22 in Chapter 17, Section 17.5
03/26/2025	BP • Rewrite/Update of Team Declaration 1

Table 1: Document Update History

Date	Updates
03/14/2025	NK, BP • Added Weekly Report 21 in Chapter 17, Section 17.6
03/07/2025	NK, RH, BP, CG • Added Weekly Report 20 in Chapter 17, Section 17.7 • Added Deployment diagram in Chapter 14, Figure 14.2 • Updated Class Diagram in Figure 12.1
02/28/2025	NK, RH, BP • Added Weekly Report 19 in Chapter 17, Section 17.8 • Updated Requirements in Chapter 8 (Requirments), Section 8.3; wording and use of shall statements. • Added 4 new requirements to Chapter 8 (Requirements) – **Update after adding new requirments • Updated Chapter 8 requirements tables. – Section 8.3; User Requirements Table – Section 8.4 System Requirements Table – Section 8.5; Non-Functional Requirements Table – Section 8.6 Domain Requirements Table
02/21/2025	NK, RH, BP • Added Weekly Report 17 and 18 in Chapter 17, Section 17.9 • Added other details to UC_7 in Chapter 10, in 10.1 • Updated Diagrams 12.1
02/07/2025	NK, RH • Added Weekly Report 16 in Chapter 17, Section 17.11 • Added UC_7 to Chapter 10, Use Case Card for main GUI • Updated Diagrams 13.1 and 13.7
01/31/2025	NK, RH • Added Weekly Report 15 in Chapter 17, Section 17.12 • Updated Use Case diagram in Chapter 10 (Use Cases), Figure 10.1 • Created Activity Diagram for Main GUI in Chapter 13 (Process View), Figure 13.7 • Improved document consistency

Table 1: Document Update History

Date	Updates
01/24/2025	NK • Added Weekly Report 14 in Chapter 17, Section 17.13
12/19/2024	NK, BP, RH, CG • Finished writing Demo Discussion in Chapter 16 • Added 3 new requirements to Chapter 8 (Requirements) – $reqInterface_{??}$ – $reqFunctional_{??}$ – $reqConstraint_4$
12/12/2024	NK, RH • Wrote Weekly Report 13 Chapter 17, Section 17.14 • Created Chapter 16 (Demo Discussion), highlighting details from our prototype demonstration. • Added a new requirement to Chapter 8 (Requirements) – $reqFunctional_8$
12/02/2024	NK, BP • Wrote Weekly Report 12 in Chapter 17, Section 17.15
12/02/2024	NK, BP, CG, RH • Created necessary chapters for IDE 401 requirements: 3, 4, 5, & 6
11/21/2024	NK, BP, CG, RH • Wrote Weekly Report 11 in Chapter 17, Section 17.16
11/17/2024	NK • Updated Descriptions in Chapters: 11, 12, 13, 14, & 15
11/12/2024	NK, BP, RH, CG • Wrote Weekly Report 10 in Chapter 17, Section 17.17 • Created Package Diagram in Chapter 14 (Development View), Figure 14.1 • Created Class Diagram in Chapter 12 (Logical View), Figure 12.1 12.1 • Updated Activity Diagrams in Chapter 13 (Process View), Figures 13.1 & 13.6 • Updated Use Case Diagram in Chapter 10 (Use Cases), Figure 10.1 • Created User Personas in Chapter 11

Table 1: Document Update History

Date	Updates
11/07/2024	NK, BP, RH, CG • Wrote Weekly Report 9 in Chapter 17, Section 17.18 • Updated Requirement Reference names in Chapter 10 (Use Cases) & Chapter 8 (Requirements) again, changes seen in table 10.1 • Updated Activity Diagrams in Chapter 13 to be more accurate with Use Case Activity Flow • Updated Use Case Diagram in Chapter 10 (Use Cases) to be more concise: Figure 10.1 • Updated Use Case Activity Flow in UC_1
11/04/2024	NK, BP, RH, CG • Updated Requirement Reference names in Chapter 10 (Use Cases) & Chapter 8 (Reqirements)
10/31/2024	NK, BP, RH • Wrote Weekly Report 8 in Chapter 17, Section 17.19 • Updated Activity Diagrams in Chapter 13 (Process View) • Added UI Concept Designs to Chapter 11 (User Interface Design): Figures 11.1, 11.2, & 11.3
10/24/2024	NK, BP, CG • Wrote Weekly Report 7 in Chapter 17, Section 17.20 • Updated Chapter 10 (Use Cases) with improved descriptions for use case cards. • Updated figure 10.1 (Use Case Diagram) making it more accurate with system description.
10/22/2024	NK, BP, RH, CG • Updated Chapter 8 (Requirements) with references to the Use Cases chapter. • Updated Chapter 10 (Use Cases) with descriptions for use case cards.
10/19/2024	NK, • Updated Chapter 8 (Requirements) with references to Use Case chapter. • Updated Chapter 10 (Use Cases) with references to Requirements Chapter

Table 1: Document Update History

Date	Updates
10/17/2024	NK, BP, RH • Wrote Weekly Report 6 in Chapter 17, Section 17.21 • Updated references in Chapter 8 (Requirements) • Added use cases and content in Chapter 10 (Use Cases) • Created Chapter 9 (User Stories) where the ideal user experience is detailed from the user's perspective
10/10/2024	NK, BP • Wrote Weekly Report 5 in Chapter 17, Section 17.22 • Updated Chapter 8 (Requirements): Sections 8.1 (Stakeholders) & 8.2 (Key Concepts)
10/09/2024	CG • Updated Chapter 8 (Requirements): Sections 8.1, 8.4, 8.5, & 8.6
10/08/2024	NK, BP, RH • Updated Chapter 8 (Requirements): Sections 8.1 & 8.2
10/03/2024	NK, BP • Wrote Weekly Report 4 in Chapter 17, Section 17.23
10/02/2024	NK, BP • Updated Chapter 7 (Development Plan): Section 7.1 (Introduction) and added terms to Glossary
10/01/2024	NK, BP, RH: • Updated Chapter 7 (Developement Plan): Sections 7.6 & 7.7, Subsections 7.13.1 (Hosting) & 7.13.2 (Platforms)
09/26/2024	NK, BP, CG: • Updated Chapter 7 (Development Plan): Sections 7.6 & 7.13, Subsection 7.3.5
09/24/2024	NK, BP, CG, RH: • Updated Chapter 7 (Development Plan): Sections 7.6, 7.7, 7.8, 7.10, & 7.12
09/23/2024	NK: • Updated Chapter 7 (Development Plan): Sections 7.4 & 7.5
09/19/2024	NK: • Updated Chapter 7 (Development Plan): Sections up to Section 7.3.6

Table 1: Document Update History

Date	Updates
09/17/2024	NK: • Wrote Weekly Report 2 Chapter 17, Section 17.25 • Updated Chapter 7 (Development Plan): Section 7.1 (Introduction)
09/12/2024	RH: • Wrote Weekly Report 1 in Chapter 17, Section 17.26 • Created Chapter 2 (Introduction), introducing each of the team members. • Created Chapter 17 (Weekly Reports), where the team will add weekly updates on the project.
09/05/2024	RH: • Created Chapter 1 (Team Declaration), describing project mission, description, driver, and constraints.

Table of Contents

List of Tables

List of Figures

Chapter 1

Team Declaration

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

Team Name: L3M

1.1 Project Description:

We want to create a new generative AI model platform that can be stored as an application on personal computers. The application would be mainly a GUI that allows users to install open source models to their local machine, issue prompts, and receive responses. This would increase security for LLM users since there would be no broadband connection between a company server hosting the AI program and the computer the prompt was entered from except for the initial installation. Current AI models consume too much energy and are reliable on an internet connection, which is something we hope to eliminate with our project. Our priorities for this project are to enable open source AI to help users with everyday office tasks and promote the democratization of the technology. Businesses could employ large language models in their workflows without relying on services like Open AI to be active 24/7.[2]

1.2 Mission Statement

To create an easy installation package for locally hosted large language models for personal and business use, to promote the security and democratization of LLM AI.

1.3 Key Drivers

Almost all large language models are only accessible online with extended features only accessible by subscription. Our project aims to provide a service for installing open source AI models for personal or business use. The models will run on local machines and will be either a one-time payment for the current version of the installer or a subscription service that allows users to download updated versions as new functionality is developed and added to the GUI.

At base, the installer will perform all of the necessary prerequisites to get a large language model up and running on a local machine. The installed model will then have the capability for the user

to expose folders to the model so the model can answer questions about the contents of those folders and files within, (i.e. asking the selected model to find a certain phrase in several pdf files). Expanding upon this, the user will be able to provide files to the model to further train it so that it can become an expert on everything the user requires of it. Finally, by default, the selected model will not have internet access for security reasons, but this option will be able to be turned on after installation. All of these services will be accessible through the unified GUI given that the installed model supports it.

Running a pre-trained large language model locally contributes greatly to the security of the user's information. One of the biggest concerns with current LLMs is the privacy of the user's prompts to the model, as most modern LLMs are hosted in the cloud on servers owned by large companies. This project aims to democratize the technology that is otherwise in the hands of only a few tech giants. While hosting LLMs in the cloud on dedicated servers has its advantages like lightning-fast response times and huge compute capacity, most users of these services do not fully utilize these resources in their day-to-day use of the technology. As such, it can be said that a locally-running LLM, on at least a somewhat recent mid-range computer, should be able to satisfy the needs of a user for personal use, and if the installer should be used for business, it will be up to the business's discretion as to how many resources will be available to the model.

The most appealing part of generative AI is the ability to customize the outputs users receive from the models they choose. As a team, we want to expand on that customization by allowing users to be able to train their installed models through a wide swath of file types. Although this may be complicated, we feel that users should be able to use our services in the most creative ways possible. This would ideally enable users to have more accurate results for the tasks they are attempting with faster load times.

1.4 Key Constraints

Using and training LLMs requires advanced hardware to perform demanding and intensive algorithms. Utilizing lightweight LLMs that are pre-trained for the user to install could mitigate this issue. This would allow users to download and install the system to any household local machine. The most intensive part of AI in terms of hardware is training the model itself. This requires extreme amounts of processing power, energy, and storage. A task no household machine would be able to do efficiently and effectively. The best way around this issue would be to have the models largely pre-trained for the user to install. LLMs, once trained, can be downloaded without the training material. The user can still add their own training to the model for customization, however, with the bulk of the task being completed, the training would be slow and could operate in the background.

Running an LLM on a local machine: even being pre-trained is still a demanding task for many household computers. One issue we could face with this is having an adequate response time. Using lightweight LLMs that help users with everyday tasks, and improving their efficiency, is important for ensuring sufficient response times. Other factors to consider when running these models are how to most efficiently utilize the system's hardware for maximum performance and minimal buffering/load.

Chapter 2

Team Introduction

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

2.1 Team Members

- Nick Katzenberger: 4/4 Software Engineering major, minoring in Quantitative Finance. From Oceanport New Jersey. I have been learning computer science and software development skills since 2018.
- Brandon Penman: 4/4 Software Engineering, from Bucks County PA, practicing development for 6 years, likes to skate.
- Ryan Hajtovik: 4/4 Software engineering major. I am from Verona, New Jersey. I've been learning about computer science and software engineering since 2018. Some of my hobbies include reading and power lifting.
- Carl Guillaume: 4/4 Software Engineering major, I am from Elizabeth, New Jersey and started learning how to code since middle school. A few of my hobbies are to play video games and watching anime.

Chapter 3

Business Objectives and Success Criteria

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

3.1 Business Objectives and Success Criteria:

- Promote Security: The system will keep a user's information safe and will not use it for any purpose other than that is required for the essential functioning of the application. This will create value for the consumer by securing the well-being of their information. It will increase our revenue by allowing the application to reach a wider audience including those who are concerned about information security or those who would like to use generative AI for applications that require said security.
  - Success Criteria: The application will be able to operate in almost its entirety without an internet connection.
- Make the technology of large-language models more accessible: by providing users with a way of installing generative AI models that are open-source, we will be able to expand our customer base to users beyond those who are familiar with programming languages. This will increase revenue to our solution. Additionally, as the technology's barrier to entry will be lowered, in the long run, the technology will be more widely adopted leading to even more increased demand.
  - Success Criteria: The application is as close to a one-touch installation and use as possible.

The greatest limiting factor in achieving these successes is hardware limitations. Large-language models are typically demanding to run and, as such, a user will need a sufficiently modern computer to be able to run our application. The main worry is that the non-tech-savvy demographic may not have the modern computers that are necessary for our application, leading to reduced customer adoption.

Chapter 4

Stakeholders

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

4.1 Project Stakeholders:

Table 4.1: Stakeholders

Stakeholder	**Major Benefits**	**Attitudes**	**Win Conditions**	**Constraints**
Project Manager	Has a new tool to access LLMs	Champion: Active advocate for the project	Students can successfully present a business pitch for their project	The students must memorize their project pitches
Project Sponsor: Professor	Has a new example of a project to show to future students	Champion: Active advocate for the project/initiative	Students attempt with valid evidence that we tried	The project must be within a specified budget
University Students	Access to tools that otherwise require extensive technical knowledge to install	Supporter: Supports the project/initiative	The system works with ease The system is easy to install The system is educational	The project does not collect company data and is not invasive on student hardware
IT Businesses	Companies can use LLMs without the risk of sensitive data being collected	Champion: Active advocate for the project/initiative	The system is easy to install	The project does not collect company data and is not invasive on company hardware

Stakeholder	**Major Benefits**	**Attitudes**	**Win Conditions**	**Constraints**
Systems with low internet connection	Access to LLMs in offline use	Champion: Active advocate for the project/initiative	The system can be locally installed	The system must work without an internet connection The project shall meet my system's hardware requirements
Data Engineers	Can directly compare output between models, potentially use differently trained models to perform different tasks	Supporter: Supports the project/initiative	Models generate differing outputs and are fully functional	The system can handle large-sized models The system can handle multiple models running at once The project shall meet my system's hardware requirements The project is compatible with my system's OS
Software Developers	Can directly compare and use multiple models to perform minor programming tasks	Neutral - No views for or against the initiative	The system provides relevant output to the user's prompt Output is mostly accurate The system has comprehensive documentation	The system must be compatible with generating code in different programming languages

Chapter 5

Project Scope

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

5.1 Project Scope:

5.1.1 In-Scope Items

1. Standalone Software Solution
    - Develop a user-friendly, installable application capable of running large language models locally.
    - Features to include are offline functionality, data privacy controls, and installation of multiple large language models.
2. Pre-Trained Model Installer
    - Allow users to install pre-trained models from Hugging-Face that address varying use cases, such as writing emails, data analysis, and other tasks.
3. Single-Platform Support
    - Version 1.0 of the project is currently planned to only be supported by Windows, however, support for Mac and Linux is planned for future development.
4. Security Measures
    - Built-in data privacy features to ensure no external data transmission occurs.
5. Offline Capability
    - Once downloaded and installed, the user can utilize the LLMs within the application without the need for an internet connection.
    - Users can switch between models, and prompt them without internet connection.

5.1.2 Planned Growth Strategies for Subsequent Releases

1. Performance Optimization
    - Efficient resource utilization to enable the system to operate on a variety of hardware configurations, including low-spec systems.
2. Enable feature for updating models
    - Enable incremental model updates through secure, optional download packages.
3. Support for Additional Operating Systems
    - Support for other operating systems including Linux and MacOS.
4. Model Customization
    - Add user-defined fine-tuning capabilities, allowing personalized model tuning.

5.1.3 Out-of-Scope Items

1. Real-Time Online Model Updates
    - No real-time updates or dependency on cloud services for the initial version (1.0).
2. Custom Hardware Development
    - The project will not include the development of specialized hardware for running the LLM.
3. Extensive Multilingual Support
    - Full support for all languages is excluded from the scope of version 1.0.
4. Third-Party Data Integration
    - Integration with external APIs or services requiring an internet connection is not planned.
5. Training for Enterprises
    - The ability for enterprises to fine-tune models through model training with proprietary datasets for specific tasks is currently out of the scope of this project.
6. Building LLMs from Scratch
    - Direct training of the LLMs from scratch is not within the project's scope; it will rely on open-source pre-trained models for foundational capabilities.

Chapter 6

Approvals

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

6.1 Approvals:

Role or Title	**Name and Signature**	**Date**
Project Sponsor	Darian Muresan	
Project Manager	Maryam Daryabegi	

Chapter 7

Development Plan

Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

7.1 Introduction

In today's digital landscape, the reliance on internet connectivity for AI models poses significant security and energy consumption challenges. Our project aims to revolutionize this by developing a new generative AI model installer that can be store and operate LLMs directly on personal computers. This approach improves security by eliminating the need for a broadband connection between a company server and the user's device, thus reducing potential vulnerabilities. By enabling businesses to have their own dedicated generative large language models, we eliminate the dependency on external services like OpenAI, ensuring 24/7 availability and control.

To facilitate this, we will develop an installer that allows users to download pre-trained models tailored to their specific needs. The installer will handle all necessary prerequisites to get a large language model up and running on a local machine. The installer will be downloadable from a website hosted in AWS containing other LLMs to download. [3]

Once installed, users will be able to prompt the model and recieve responses from the model that are relevant to the prompt. Users will also be able to provide additional files to further train the model, making it an expert on their specific requirements. For training with large datasets, this feature will only be accessible through an online connection to the website due to storage constraints of modern hardware. All these features will be accessible through a unified, user-friendly GUI.

7.2 Roles and Responsibilities

1. Development Lead : Carl
2. Buildmeister : Brandon
3. Architect : Nick
4. Developers : Ryan, Brandon Nick, Carl
5. Test Lead : Ryan
6. Testers : Brandon, Ryan, Nick

7. Documentation : Brandon, Ryan, Nick, Carl
8. Documentation Editor : Nick
9. Designer : Brandon
10. User advocate : Ryan
11. Risk Management : Carl
12. System Administrator : Brandon
13. Modification Request Board : Leader: Nick, Members: Ryan, Brandon, Carl
14. Requirements Resource : Carl
15. Customer Representatives : Ryan, Carl
16. Customer responsible for acceptance testing : Nick

7.3 Methods (Resources)

7.3.1 Software

Coding Languages

- Utilize XML for creating an installer using VS code as the compiler and the Wix Toolset [3]
- Using Python Flask 3.0.3 to create website backend and HTML/CSS for the frontend
- Give instructions and train installed models with various versions of Python depending on the LLM

Operating Systems

Our goal is to first create a system operable in Windows, then move to Mac and Linux if Windows 11 deployment is successful and without fatal errors.
Possible Operating Systems to Consider:

- Windows 11
- Windows 10
- Windows 8
- MAC OS
- Linux

Software Packages

- Github Api v 2022-11-28 [4]
- Various LLMs from GitHub
- AWS api CLI version 2 [1]
- Wix Toolset v5.0.1

Documentation Refefences

- IEEE Conops documentation
- UML: Visual Paradigm

Code conventions – this should preferably be a pointer to a document agreed to and followed by everyone

7.3.2 Hardware

1. Development Hardware: Windows 11/10 desktops and laptops
2. Test Hardware: Windows 11/10 desktops and laptops
3. Target/Deployment Hardware: 'Bare metal' and virtualized Windows 11/10/8* OS to start, then proceed to Linux-based systems, then MacOS

7.3.3 Backup plan

Our plan for an alternative to this project is that we would rely more heavily on the website aspect of the project. This would rely on AWS servers to host our website as well as the LLMs that users can work with and train different models on the website. Our original plan of having the LLMs work with Microsoft Office tools could still be feasible but require an internet connection, like Copilot. If systematically more parts of the project cannot work, the group will search for alternative ideas that enable our core idea to still be developed.

Errors with the project will be reported on Github's support ticket system. Then the team will evaluate the error reports and act accordingly to fix the error, if possible. If an individual working on the project cannot achieve the specific task, then they should notify the team immediately so the rest of the group can discuss potential alternatives or solutions to the task they are attempting to achieve. [4]

7.3.4 Review Process

1. We will be doing code reviews for each part of the code. We will then review the usability of that code for our final project.

2. The way we will approach the reviews is by reviewing the work done in each phase and how well it works with the phase before it. To make sure that everything is done and functional. Once we have something more complete to test we will do formal reviews having outside people attempt using the product.

3. Every team member will take part in reviewing. Each member will of course review the work they have done and its viability towards our project. If issues are found they are expected to try and find it or ask for help. Then together we will review the final project and host the more formal outside participant review.

4. code readings?

7.3.5 Build Plan

We plan to use Github Actions for our builds to start, this will include steps like linting and unit tests. After the project becomes sufficiently complex, we may move to a solution such as Circle CI as it is free to use and aids in the build and testing process. [5] [6]

1. GitHub/Git will be used for the code revision control and as the remote repository

2. GitHub actions will be utilized for CI/CD [4]

3. Builds will be done every time there is a merge to the main branch

4. Deadlines for the builds – deadline for source updates

5. Multiplicity of builds

6. Regression test process – see test plan

7.3.6 Modification Request Process

1. Github Pull requests will be used for modification requests

2. Decision process: Pull requests to the main will be reviewed by at least one other developer other than the requester before being approved

3. the same process stream will be used during and after development

7.4 Virtual and Real Workspace

For working together in person, we were planning mainly to meet in the library where we can choose any open room to meet and discuss the progress of the week. In the past, we mainly used the Software Engineering Lab in the Kenneth J. Altorfer Academic Complex Building at Stevens Institute of Technology.

Our virtual workspace is hosted on a GitHub repository where we will make branches to view and commit changes to the project. This website works most efficiently because it organizes where

files are stored and its broad integration into many software compilers. GitHub's notoriety for file security and computer safety also makes it a great choice for being reliable. The documentation for our progress and methodology for the project will be stored in this document under different sections. We update our weekly progress in Chapter 17, Weekly Reports; and we record our processes in several other chapters throughout the document.

7.5 Communication Plan

7.5.1 Heartbeat Meetings

These meetings have occurred twice a week after class time on Tuesdays and Thursdays. These meetings are roughly 30 minutes long and in person, since every group member is expected to attend the class periods. During these meetings, we update the weekly progress reports with a plan outlined for the rest of the week and reflect on the progress of the past week. The plan is universally accepted by the group members and followed throughout the week. We record all the essential notions of the meeting in the weekly report. In addition, we log issues, risks, and blockers at the end of the report to ensure that we are taking these issues into account.

7.5.2 Status Meetings

Status Meetings occur biweekly and include the project manager. The manager for the project is Professor David Darian Muresan. They are about 1 hour long and mainly focus on the progress made in the two weeks of development. We mainly take feedback in these meetings based on our progress and attempt to implement said feedback in the next two weeks.

7.5.3 Issues Meetings

If issues or blockers arise in our project that the team cannot immediately handle, we schedule issues meetings with the project manager. Issues meetings range between 30 minutes to an hour, depending on the issue. In these meetings, we outline the problem and the potential plans the team has to fix these problems. The issues discussed are mainly logged after a heartbeat meeting and are ordered in terms of importance. At the end of the meeting, the team aims to have a solid solution to the problem at hand.

7.6 Timeline and Milestones

Table 7.1: Timeline and Milestones

Start Date	Event or Deliverable	Target Date	Responsibility
September 16	Create Items in the Research Chapter	December 2	Record all research in that chapter (not included in this submission)
September 16	Begin documentation on the project.	April 20, 2025	Record all documentation in the Overleaf document

Table 7.1: Timeline and Milestones

Start Date	**Event or Deliverable**	**Target Date**	**Responsibility**
September 16	Add all terms to Glossary	April 30th	Provide definitions to specific terms throughout the document
September 16	Finish Development Plan Chapter	October 3rd	Present the plan
October 3	Start making other UML Diagrams.	October 31st.	To be put on Logical View Chapter, Process View Chapter, Development View Chapter, and Physical View Chapter
October 4	Create User Requirements	October 11th	Write User Requirements
October 11	Create Use Case diagrams	October 18th	Draw Use Case Diagrams in Use Case Chapter
October 18	Create User Stories Chapter	October 25th	Write User Stories
November 1st	Begin development on the website	March 1st	• Hosts installer files • Hosts application UI files • Provides information about the project and models available
November 23	Develop installer for simple Large Language model	December 19th	LLMs must Work on the local system with and without internet connection
December 12	Present project progress	December 12	Show progress on project to the class

Table 7.1: Timeline and Milestones

Start Date	**Event or Deliverable**	**Target Date**	**Responsibility**
December 18	Begin Development of a user-friendly UI app	February 19th	• Uses and prompts AI models. • Works with the installer. • Can view different installed models • can switch model in use • can view available models to install • can install models from Hugging Face [7]
January 15	Implement a diverse list of Large Language Models	February 3	List on the website/installer and test functions on a local machine.
January 21	Test installing and prompting Large Language Models	February 10	Ensure functionality and accuracy
February 3	Make the front end of the website more appealing	February 28	Appeal to general users. Add more functionality (Accounts, Accessibility, etc)
March 3rd	Begin/Continue development of methods for training models	March 31	Models should be able to receive data that will affect their response depending on the prompt.
April 1	Implement More diverse models	April 18	Models include (image, sound, video, etc.) and Create UI methods to access these different models. Test their deployment
April 7	Begin final testing of product	April 24	and ensure all tests pass
April 25	Present final project poster	Innovation Expo day	Show audiences what we worked on over the semester and show final product

7.7 Testing Plan

In this section, we describe the testing methods and plans for each test used to develop project. For our project, will use CircleCI and Github Actions to control the testing and maintenance of our project.

1. Test Driven Development
   Test-driven will be used to help build functions and ensure that the functions work as intended.

2. Unit Testing
   The first tests to be run. For each feature, all functions and classes should be individually tested using pytest, unittest, or other testing libraries to determine if there are potential failures or functionality issues in the function or class. Testing will be monitored on CircleCI

3. Integration testing
   The second set of tests is to be run. This testing ensures all classes and functions integrate well with one another.

4. System testing
   System testing is the last test to run before any feature release. This test checks the entire system for any fatal issues.

5. Regression testing
   Regression Testing should be done before every feature release after the first few features are implemented. This will be done by creating a test for each existing function and then testing them after new functions are introduced to ensure they still work as intended.

6. Acceptance testing
   Acceptance testing will be utilized later in the project development. This tests to ensure that the business requirements are met and the system performs its intended use cases.

7. Testing for critical quality attributes
   This testing will be done towards the end of the project when there is more of an established product. This testing helps with security, usability, reliability, and scalability. It is not necessary early in the project development but is crucial later on.

7.8 Risks

The largest and main risk that we may face is the large language models being inaccessible ranging from causes like the files being too big or the models being behind too large of a paywall. Another risk we face is that the hardware we are using to test the local hosting of the language models is not powerful enough to run the models efficiently. This would limit the use of our project to only be able to work naively on high-end systems not widely available to the public.

7.9 Assumptions

- Assumptions About Customer
  - Target hardware will be at least somewhat modern (within 5 years or less of age)
  - User will have basic proficiency in using a keyboard and navigating software like web browsers, Microsoft Windows, etc...
  - User's item possesses either an integrated or dedicated graphics unit
- Assumptions About Project
  - Developers will be using their own hardware to contribute to the project
  - This project is unpaid, as it is a "start-up" for a college class
  - Tasks will, for the most part, be completed on time and with minimal technical debt
  - technical debt will not be repaid during the duration of the class, only afterward

7.10 Distribution List

- Nick Katzenberger
- Brandon Penman
- Carl Guillaume
- Ryan Hajtovik
- Professor David Darian Muresan

7.11 IRB Protocol (required)

This project does not require an IRB application.

7.12 Required Resource and Budget

- Resources:
  - Development Hardware: laptops, desktops, Integrated Dev Environment
  - Testing Hardware: machines with different operating systems, virtual machines for rapid testing
  - Data Storage: models need to be stored on local hard drives during testing
- Budget:

- Cloud Storage: if AWS or another similar service is utilized for pre-trained model hosting, a rate $/GB will be charged, if not, the ”cloud” will be one of our machines running as a web server which will cost watt-hours.
- Domain Name
- Web Hosting
- WE DO NOT EXPECT TO NEED TO BE REIMBURSED FOR THESE COSTS

7.13 Documentation Plan

We will use the IEEE standards for recording our research and progress on the project. The team will record everything related to this project will be stored on these documents throughout various chapters. Each member of the group will equally contribute to writing different chapters of the document. Additionally, each member will also be responsible for editing and correcting errors made on the document prior to each version update. For recording our progress on the project, we will use the Weekly Updates (Chapter 17).

7.13.1 Hosting

We plan to use Hugging Face to source pre-trained large language models in early development. These models would be trained and developed by other companies and are open-source for our use. Hugging Face will allow us to prototype and test downloading and locally hosting large language models. In future development, the plan is to utilize Amazon Web Services (AWS) and build proprietary large language models for users to download and customize to their specific needs.

For the website portion, we plan to build and test the design and functionality with local machine hosting. For deployment, we would utilize AWS to host the website. AWS is a great tool for this project as it is highly scalable, requires minimal investment, and doesn't require any physical hardware on our end.

7.13.2 Platforms

We plan to start our project building for Windows 10 and 11 usages. By minimizing the platforms initially we test the business viability and gather user feedback. Later in the project development, we would expand to incorporate other operating systems, such as Apple and Linux, and later versions of operating systems.

Chapter 8

Requirements

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

8.1 Stakeholders

8.1.1 Customers

Clients and users. Who are they, why do they use your system

- Businesses Focusing on IT
- University Students
- People who want to use AI but aren't comfortable with data collection and/or using complex methods to use LLMs
  - Artists
  - Accountants
- Laptop users without connection to the internet

8.1.2 Sponsors

These are possible entities or individuals funding and supporting the development and distribution of the project.:

- Private Donors/Investors: Could be individuals or businesses that see value in creating an easy-to-install, secure local AI model that does not rely on data harvesting.
- Open-Source Advocacy Groups: Organizations that promote open-source technologies and see this project as a means to provide secure AI access.

8.1.3 Engineering and Technical Persons

- Businesses Focusing on IT

- Data Engineers
- Software Developers
- Contractors

8.1.4 Regulators

- Universities - May have rules against students using AI on assignments
- Copyright Law - international laws prevent the resale of products protected by brands. In this case, the models of AIs/LLMs we plan to use

8.1.5 Third Parties

- **Hugging Face**: We will use various LLMs on Huggingface to provide open source AI models to our project
- **Github**: Github also hosts open source LLMs and will be the primary website for hosting our project's code.

8.1.6 Competitors

- Nvidia Chat RTX - Provides a similar service utilizing Nvidia's RTX line of graphics units, it will be hard to match the performance of the specialized hardware, but our system will be compatible with systems with other types of graphics processing units besides Nvidia's own.
- Google Gemini: Google's AI model
- OpenAI's ChatGPT: The most popular generative AI model on the market that many companies like Microsoft, Snapchat, Morgan Stanley, and others use to elevate their businesses.
- Amazon AI: Various AI models provided by Amazon
  - Bedrock
  - Sagemaker
  - Codewhisperer
  - Jumpstart
  - Q
  - Polly
- LM Studio - Provides software similar to our project idea. This software allows users to download and run LLM's on local machines.

8.2 Key Concepts

- Large Language Models (LLMs)
- Cloud
- Data
- Local Hosting

8.3 User Requirements

Table 8.1: User Requirements Table

Requirement	Priority	Use Case(s)
Interface Requirement 1 (reqiGUI) *The system shall have a GUI built using pyQT for locally using installed LLM models.*	MustHave	UC_7
Interface Requirement 2 (reqiWebsite) *The system shall have a website where users can download and install the GUI application.*	MustHave	UC_5 & UC_6
Functional Requirement 1 (reqfDownloadModel) *The system shall allow users to download at least one LLM at a time from the GUI or Website using Hugging Face's API.*	MustHave	UC_1 & UC_4 & UC_6
Functional Requirement 2 (reqfUninstallModel) *The system shall allow users to uninstall the locally installed LLMs from the GUI in 3 or fewer steps.*	ShouldHave	UC_1
Functional Requirement 3 (reqfMultipleModels) *The system shall allow the user to have multiple LLMs installed on their device.*	MustHave	UC_1 & UC_6
Functional Requirement 4 (reqfSwitch) *The system shall allow the user to switch between various installed models within the GUI.*	MustHave	UC_3

Table 8.1: User Requirements Table

Requirement	Priority	Use Case(s)
Functional Requirement 5 (reqfPrompt) *The system shall allow the user to enter text as a prompt for the LLM to generate a response using the GUI.*	MustHave	UC_2
Functional Requirement 6 (reqfResponse) *The system shall output a text-generated response to the GUI that correlates with the text prompt using the selected LLM.*	MustHave	UC_2
Functional Requirement 7 (reqfTrain) *The system shall provide a method for imposing custom weights or constraints on the LLM(s) within the GUI, to allow for user customizability of responses.*	WouldHave	UC_1
Functional Requirement 8 (reqfSpecs) *The system shall identify the computer hardware specifications of the user's device and recommend LLMs to be installed based on the computer's hardware capabilities.*	CouldHave	UC_1 & UC_6 & UC_7
Functional Requirement 9 (reqfOtherModelsDownload) *The system shall allow users to download models that generate photos, audio, and video.*	CouldHave	UC_1 & UC_6
Functional Requirement 10 (reqfOtherModelsResponse) *The system shall provide a photo, video, or audio response within the GUI when a correlated model is selected.*	CouldHave	UC_2

8.4 System (Constraints) Requirements

Table 8.2: System Requirements Table

Requirement	Priority	Use Case(s)
Constraint Requirement 1 (reqcAnyHardware) *The system shall have a GUI that is capable of running on machines with varying hardware configurations, including those without high-end GPUs.* 1. *2GB GPU Ram* 2. *AMD xyz processor* 3. *Mac 2020 M1 processor* 4. *8GB of RAM* 5. *500 GB of free HDD space.*	MustHave	UC_5 & UC_7
Constraint Requirement 2 (reqcEZInstall) *The installation process shall be streamlined to avoid requiring extensive technical knowledge from the user.*	MustHave	UC_1 & UC_5 & UC_6 & UC_7
Constraint Requirement 3 (reqcSecurity) *The system shall ensure that users have full control over their data, with no external servers involved unless explicitly chosen by the user.*	ShouldHave	UC_3 & UC_7
Constraint Requirement 4 (reqcStorage) *The system shall check to see if the user's system has enough storage space available before installing any new models.*	WouldHave	UC_1 & UC_4

8.5 Non-functional (Quality) Requirements

Table 8.3: Non-Functional Requirements Table

Requirement	Priority	Use Case(s)
Quality Requirement 1 (reqqOptimize) *Performance: The system shall be able to perform LLM tasks under 1 minute on machines better or equal to the specifications listed in* $reqConstraint_1$	MustHave	UC_2

Table 8.3: Non-Functional Requirements Table

Requirement	Priority	Use Case(s)
Quality Requirement 2 (reqqEZSetup) *Usability: The system shall be easy to install and configure, with minimal setup required.*	MustHave	UC_5 & UC_7
Quality Requirement 3 (reqqExternalData) *Security: The system shall ensure user data remains secure, with no external data transmission unless the user opts in.*	ShouldHave	UC_2 & UC_6
Quality Requirement 4 (reqqScalability) *Scalability: The system shall be able to handle models of varying sizes, from smaller models (less than 1 Gigabyte) for personal use to larger ones (up to 500 GB) for business applications.*	MustHave	UC_1 & UC_2 & UC_4
Quality Requirement 5 (reqqDisplayInstalled) *The system shall show the user which models they have installed to their device on the website.*	ShouldHave	UC_1 & UC_6

8.6 Domain (Business) Requirements

Table 8.4: Domain Requirements Table

Requirement	Priority	Use Case(s)
Business Requirement 1 (reqbLicense) *The system shall not violate licensing terms for open-source LLMs.*	MustHave	UC_1 & UC_4
Business Requirement 2 (reqbAccredit) *The system shall credit various models used in its Terms of Service and operation.*	MustHave	UC_1 & UC_4

Table 8.4: Domain Requirements Table

Requirement	Priority	Use Case(s)
Business Requirement 3 (reqbDemocracy) *The system shall promote the democratization of AI by making it accessible to a broad audience, including non-technical users and to both individual users and businesses.*	MustHave	UC_4 & UC_7
Business Requirement 4 (reqbWindows) *The system shall operate without crashing on Windows operating systems.*	MustHave	UC_2 & UC_6
Business Requirement 5 (reqbMacLinux) *The system shall operate without crashing on Mac and Linux operating systems.*	WouldHave	UC_2 & UC_6

Chapter 9

User Stories

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

9.1 User Stories

1. As a user, I should be able to easily install a large language model onto my local machine without any extensive technical knowledge background.
    (a) As a user, I should not need extensive technical knowledge.
    (b) As a user, I need a tool that installs the software for me.
    (c) As a user, I want the model to be able to run without an internet connection.
2. As a user, I want a graphical user interface to operate the large language model with.
3. As a developer, I want to source the large language models from Hugging Face.
4. As a LLM developer, I want my model's use to be credited.
5. As a developer, I want the product to run on all major operating systems.
    (a) As a developer, I want the product to run on Windows.
    (b) As a developer, I want the product to run on Mac.
    (c) As a developer, I want the product to run on Linux.
6. As a stakeholder, I want large language models to be more broadly accessible and in the hands of the user, to promote the democratization of the technology.
7. As a user, I want the product to be capable of running on a wide range of low to high-end systems
    (a) As a user, I want different models with varying levels of computational complexity to be available for use.
8. As a host for a large language model, I want my software to be credited appropriately in this system.
9. As a user, the performance of the system should be comparable to other generative AI software available to me.
10. As a user I want to be able to easily change between LLMs I am using to answer prompts.

11. As a user I want to be able to train the LLM online so my system's performance is not impacted severely. system

12. As a user, I want to tailor and constrain the model to complete my specific tasks better.

Chapter 10

Use Cases

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

10.1 Table of Use Cases

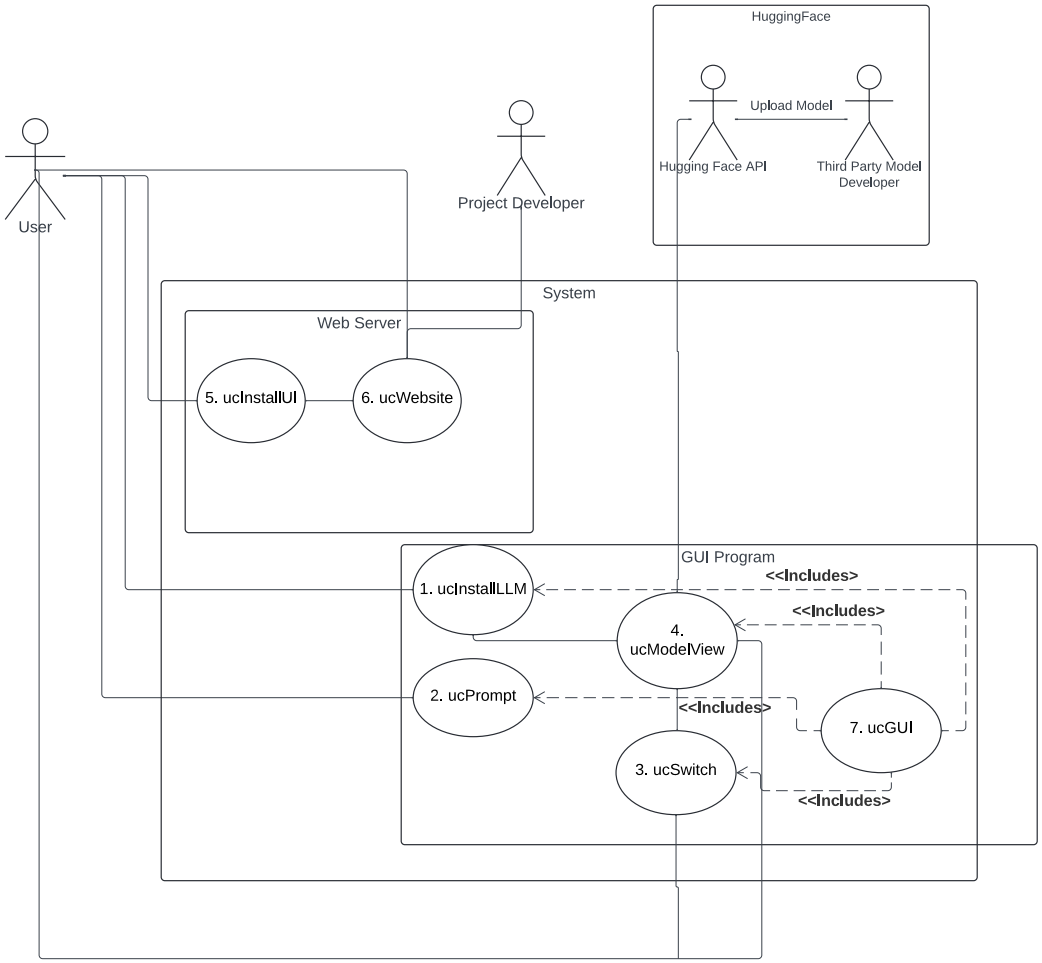
Table 10.1: Table of Use Cases

Use Case	Requirements	User Stories	Name and Description
UC_1	$reqFunctional_1$ $reqFunctional_2$ $reqFunctional_3$ $reqFunctional_7$ $reqFunctional_8$ $reqFunctional_9$ $reqBusiness_1$ $reqBusiness_2$ $reqQuality_4$ $reqQuality_5$ $reqConstraint_2$ $reqConstraint_4$	7a, 1a, 6	ucInstallLLM - User installs a large language model onto their local system from Hugging Face and other sources using our application.
UC_2	$reqFunctional_5$ $reqFunctional_6$ $reqFunctional_{10}$ $reqQuality_1$ $reqQuality_3$ $reqQuality_4$ $reqBusiness_4$ $reqBusiness_5$	7, 5, 9, 2	ucPrompt - User is able to enter prompts into the model to generate an output from the model
UC_3	$reqFunctional_4$ $reqBusiness_2$ $reqConstraint_3$	10, 2, 9	ucSwitch - User can change between the LLM they are using using built in options. It is assumed that the user already has 1 LLM installed

Table 10.1: Table of Use Cases

Use Case	Requirements	User Stories	Name and Description
UC_4	$reqFunctional_1$ $reqFunctional_9$ $reqBusiness_1$ $reqBusiness_2$ $reqBusiness_3$ $reqQuality_4$ $reqConstraint_4$	8, 6 9	ucModelView - System updates the available models to download from Hugging Face in the GUI and/or website and credits its creators.
UC_5	$reqInterface_2$, $reqConstraint_1$, $reqConstraint_2$, $reqQuality_2$	1b, 2, 5a, 5c, 5b, 7a, 5	ucInstallUI User can download and install the application on to their local system from the website, and the installer downloads all software dependencies
UC_6	$reqInterface_2$ $reqFunctional_8$ $reqFunctional_1$ $reqFunctional_9$ $reqFunctional_3$ $reqQuality_3$ $reqQuality_5$ $reqConstraint_2$	7a 5a, 11, 7a	ucWebsite Users can access a website that hosts the installer and other models found on Hugging Face or Github.
UC_7	$reqInterface_1$ $reqFunctional_8$ $reqQuality_2$ $reqBusiness_3$ $reqBusiness_4$ $reqBusiness_5$ $reqConstraint_1$ $reqConstraint_2$ $reqConstraint_3$	2, 5a, 5c, 5b, 11	ucGUI Users access a GUI application that allows them to operate the application and access other use cases.

10.2 Use Case Diagrams



1. ucInstallLLM: install LLM Model to target OS
2. ucPrompt: user prompts model and recieves output
3. ucSwitch: User swtiches installed model
4. ucModelView: User selects a model, with creators accredited to each model
5. ucInstall UI: Application is installed on target OS
6. ucWebsite: Models and application stored on a website
7. ucGUI: Application that the user interacts with

10.3 Use Case Cards

Table 10.2: Installer Use Case Card

Use Case 1 (ucInstallLLM) *Install all of the necessary dependencies for GUI program operation.*
Diagrams: Figure 10.1, 13.1, 13.7
Brief description: An installer is provided in order for a user to easily install LLMs onto their system through the application as well as any dependencies which will ultimately allow the user to run these models on their local machine.
Primary actors: User Project Developer / Owner
Secondary actors: Copyright Laws Model Developer / Owner
Preconditions: 1. Installer and necessary files are hosted on a web-server for the user to download from
Main flow: 1. The User accesses the web server hosted by the Product Owner through the application. 2. Depending on the User's operating system, different installers will deploy (Mac, Linux, Windows) 3. The installer is retrieves files from web server 4. installer begins installation 5. During installation, the user may change installation instructions such as target destination.
Post conditions: The GUI Program is installed and set up correctly.
Alternative flows: The installer does not install the model correctly due to an error.

[3]

Table 10.3: Prompt Use Case Card

Use Case 2 (ucPrompt) *Prompt installed models to generate an output.*
Diagrams: Figure 10.1, 13.2, 13.9
Brief description: Once a LLM/Gen AI is installed to the user's system, the user will be able to enter text, image, or audio prompts (depending on the model) to receive an output that correlates with the user's prompt.
Primary actors: User LLM/Gen AI Developer / Owner
Secondary actors: Project Developers Copyright Law Privacy Law Internet Resources/Connection
Preconditions: 1. User has installed a LLM model through our application 2. User has access to running our system 3. Model installed correctly
Main flow: 1. User enters the prompt into the application via UI indication 2. LLM and/or Gen AI model receives prompt 3. Model generates output 4. UI displays model output
Postconditions: 1. Prompt correlates with user's input 2. GUI allows user to enter next prompt
Alternative flows: 1. User enters prompt 2. Model outputs an error 3. User is requested to enter a different prompt and/or fix any issues with their prompt. Issues may be: • prompt size is too large • prompt contains invalid files • prompt contains illegal characters • prompt contains profanity (depends on model)

Table 10.4: Switch Use Case Card

Use Case 3 (ucSwitch) *User Switches between installed AI Models*
Diagrams: Figure 10.1, 13.3, 13.7
Brief description: The user will be able to switch between the various AI models installed on the user's machine. This allows the user to control which model provides the output response to the user's prompt.
Primary actors: User
Secondary actors: Hugging Face API Application UI interface
Preconditions: 1. The user has installed multiple LLM's on their machine 2. Available models are ones that are completely installed
Main flow: 1. User clicks on the option to switch models 2. User selects desired LLM from UI interface. 3. User prompts model
Postconditions: The user can use different models depending on the task desired by the user.
Alternative flows: 1. User selects desired LLM from UI interface 2. System prevents User from selecting a model that is not fully installed 3. User's model in use does not change

Table 10.5: Model Viewer Use Case Card

Use Case 4 (ucModelView) *Download Models directly from the application GUI.*
Diagrams: Figure 10.1, 13.4, 13.1 13.8
Brief description: A part of the GUI that displays relevant AI models from Hugging Face to the user and allows them to select and download them.
Primary actors: User Project Developer / Owner Hugging Face API
Secondary actors: None.
Preconditions: 1. GUI application is installed.
Main flow: 1. User will open the menu to select a new AI model 2. User will select an AI model 3. application will communicate with Hugging Face API to retrieve the new model 4. model will be loaded onto the user's local machine
Postconditions: The user will be able to prompt and receive output from the newly downloaded model.
Alternative flows: 1. User opens the model viewer to download a new model 2. User decides to do nothing and closes panel / goes back to the main screen of GUI.

Table 10.6: Download GUI Use Case Card

Use Case 5 (ucInstallUI) *User downloads the application from the website onto their local system*
Diagrams: Figure 10.1, 13.5, 14.2
Brief description: The user will be able to download an application that includes a UI that allows users to enter prompts, switch between models, and install new models with ease. The application files are hosted on the website where the user can download the files and install the application onto their system.
Primary actors: User Project Developer / Owner System website User's System
Secondary actors: AWS server
Preconditions: 1. User has an internet connection 2. User's device meets minimum requirements for downloading and running applications/models
Main flow: 1. User accesses website 2. User selects Application & Installer version depending on their system. 2.1. Download Windows installer. 2.2. Download Mac installer. 2.3. Download Linux installer. 3. User clicks the download button 4. User's system downloads files 5. User executes application installer 6. Application is installed onto the user's system 7. User executes application
Postconditions: The application installs successfully.
Alternative flows: 1. User accesses website 2. User clicks to download files 3. System downloads files from website 4. Application is unable to be installed and/or downloaded because of error or system security preventing changes to user's system.

Table 10.7: Website Use Case Card

Use Case 6 (ucWebsite) *Location of the application and models*
Diagrams: Figure 10.1, 13.6, 14.2, 14.1, 15.1
Brief description: This is where the users can find and download the application. It is also where users will go to browse and download different models for the application.
Primary actors: User Project Developer / Owner Hugging Face API
Secondary actors: AWS server
Preconditions: 1. User has internet access
Main flow: 1. User browses to the website 2. User uses navigation features to find the application installer 3. User downloads the application from the website 4. User uses navigation features to browse various LLM's 5. User reads the description of LLM and decides to install or not. 6. User downloads desired LLM's to use with the application.
Postconditions: User can now download and install the application and different LLM's on their local machine from the website.
Alternative flows: 1. User browses to the website 2. User cannot access website due to regional restrictions, system security, or other issues
1. User browses to the website 2. User uses navigation features to find the application installer 3. User attempts downloads the application from the website 4. User's system prevents downloading the application due to their system's security settings.

Table 10.8: GUI Use Case Card

Use Case 7 (ucGUI) *Location of the application and models*
Diagrams: Figure 10.1, 13.7, 15.1, 14.2, 14.1
Brief description: This use case demonstrates how users access the application and what they see on the front end. The use case includes UC_1 UC_3, UC_2, and UC_4. Users can click buttons allow users to access the different use cases.
Primary actors: User Project Developer / Owner Hugging Face API
Secondary actors: Hugging Face Developers
Preconditions: 1. User clicks to open .exe file.
Main flow: 1. User opens the application 2. User chooses a model to prompt 3. User receives response from model 4. User browses available models • requires internet connection 5. User chooses to download LLM from list of available models 6. User switches to installed model 7. User can exit application at any time
Postconditions: Prompt history is stored in a JSON file, changed model names are stored in separate JSON files.
Alternative flows: 1. User opens application 2. User tries to browse available models 3. Hugging Face API refuses to connect or User does not have internet connection 4. User is redirected to GUI front page 1. User opens application 2. User browses available models 3. User attempts to install model 4. Installation is prevented because file size of LLM is too large 5. GUI returns error message 6. GUI returns user to home page.

Chapter 11

User Interface Design

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

11.1 User Persona

11.1.1 Sarah Thompson

Sarah Thompson, a 42-year-old freelance writer, values privacy, simplicity, and offline functionality in her tools. While she's comfortable with basic tech, she finds complex setups intimidating and prefers quick, straightforward installations without the need for technical skills. Sarah often works in locations with unreliable internet and prioritizes solutions that work entirely offline. Her work frequently involves sensitive topics, so data privacy is a must.

For Sarah, the ideal language model installer would offer a seamless experience: download, click 'Install', and go. She appreciates tools that are designed for nontechnical users, with a clear, user-friendly interface that ensures her data remain private on her local machine. Offline capability is essential for your productivity, as it allows you to brainstorm, edit and write regardless of connectivity. This ease of use, combined with a focus on privacy, would make her feel comfortable using the software in any setting.

11.1.2 James Carter

James Carter, a 38-year-old Chief Information Officer for a mid-size financial services firm, oversees company technology and data management with a strong focus on security and compliance. The firm handles highly sensitive client data and proprietary analytics, which means that any software they adopt must guarantee full data privacy and control. Although James sees the potential for AI to improve productivity, he requires solutions that operate entirely within the company's secure environment, without the risk that proprietary or client information will be used for external model training.

For James, the ideal AI installer would offer a quick, straightforward setup and operate fully offline, enabling his team to use the AI capabilities without any data leaving the company's network. He values a clear, user-friendly interface that allows easy configuration to meet strict compliance and data protection standards. This combination of security, ease of use, and operational efficiency would give James confidence that the AI tool is both safe for business use and scalable as the firm grows.

11.2 User Interface Design

Below we have included a mock design of what we intend our user interface to look like for our project's prototype. We have included designs for what prompting the model, viewing different/switching models, and accessing the website looks like.

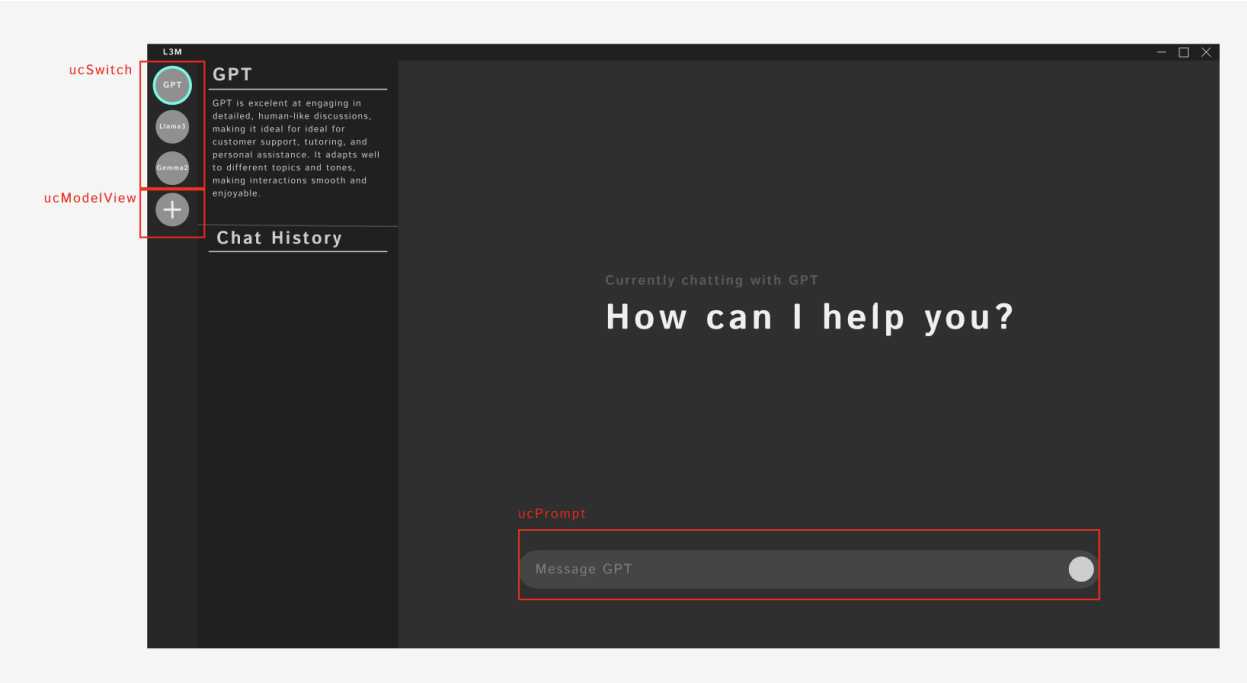


Figure 11.1: Main GUI prototype

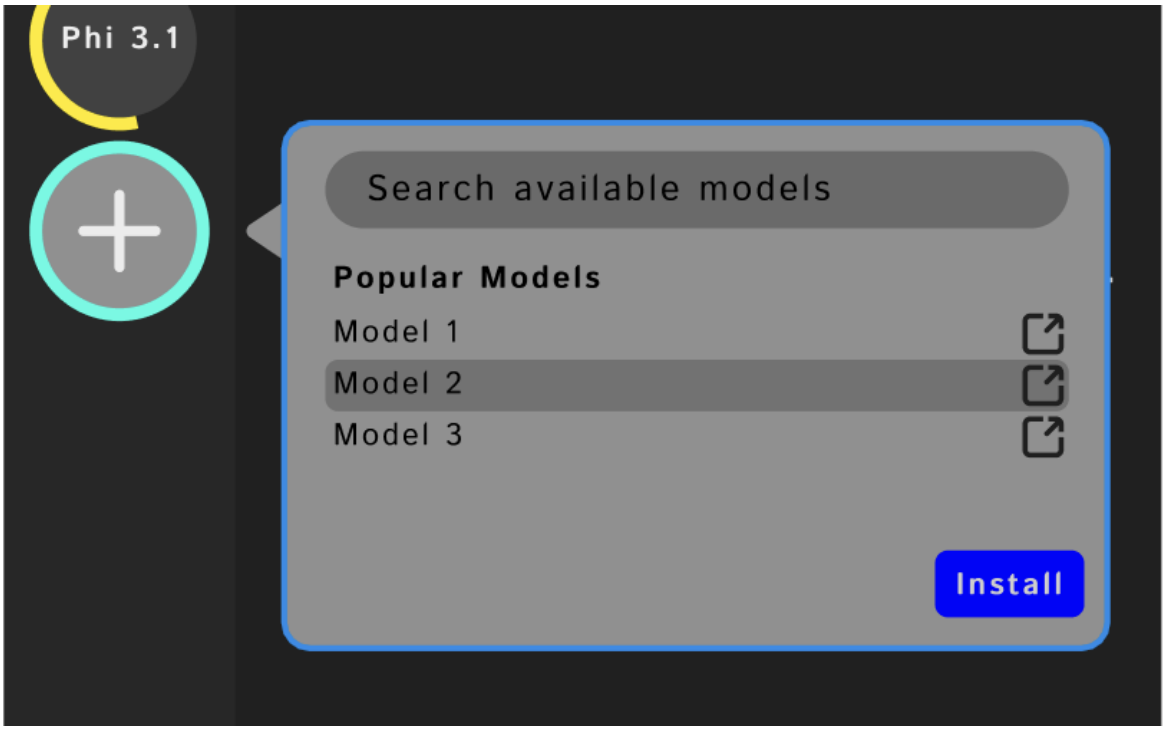


Figure 11.2: ucModelView prototype

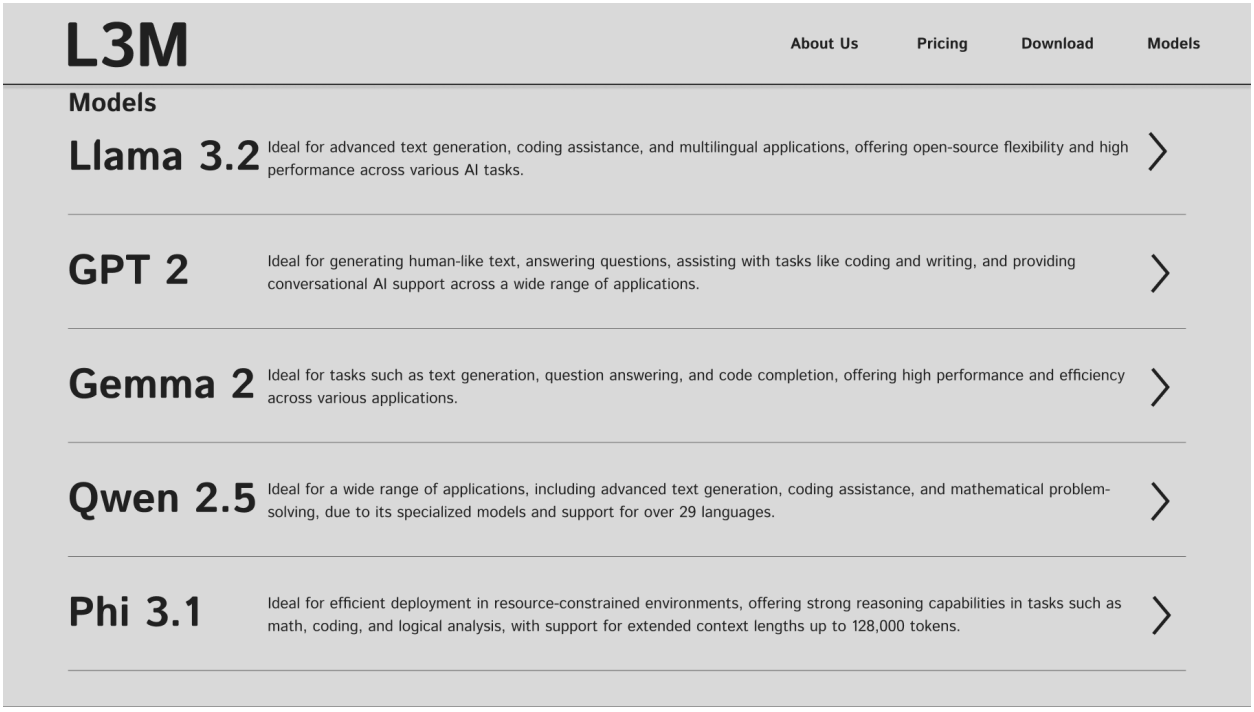


Figure 11.3: Website Model View prototype

11.3 Updated Prototype Images

11.3.1 GUI

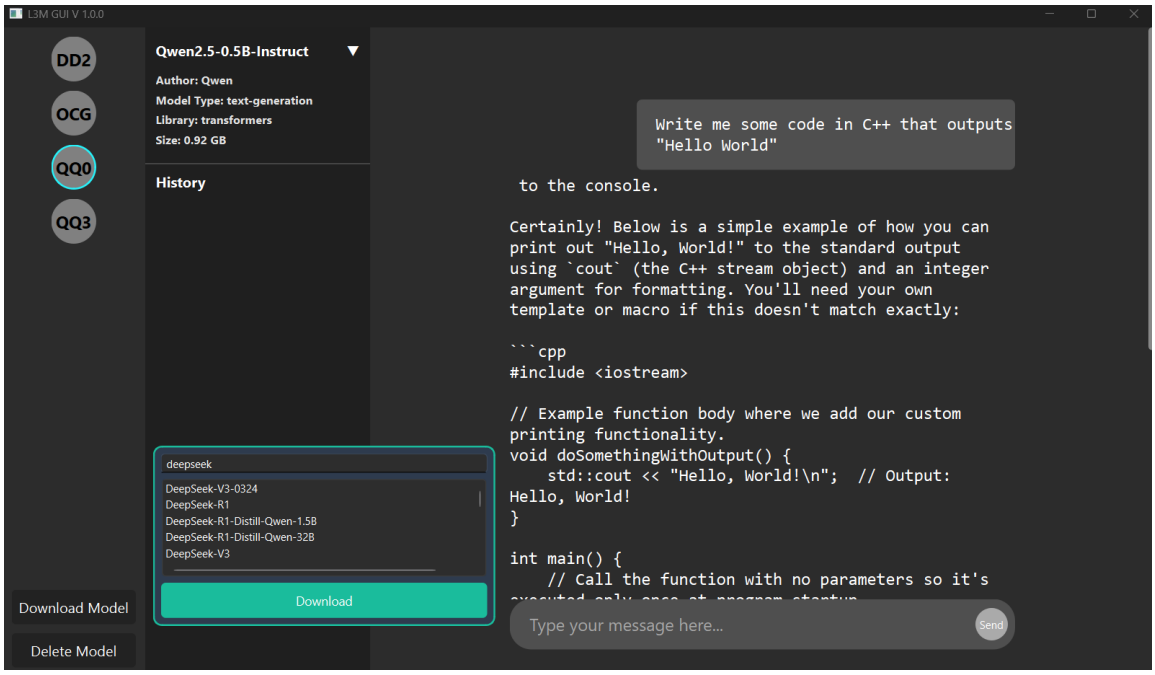


Figure 11.4: GUI Prototype 4/17/2025

11.3.2 Website

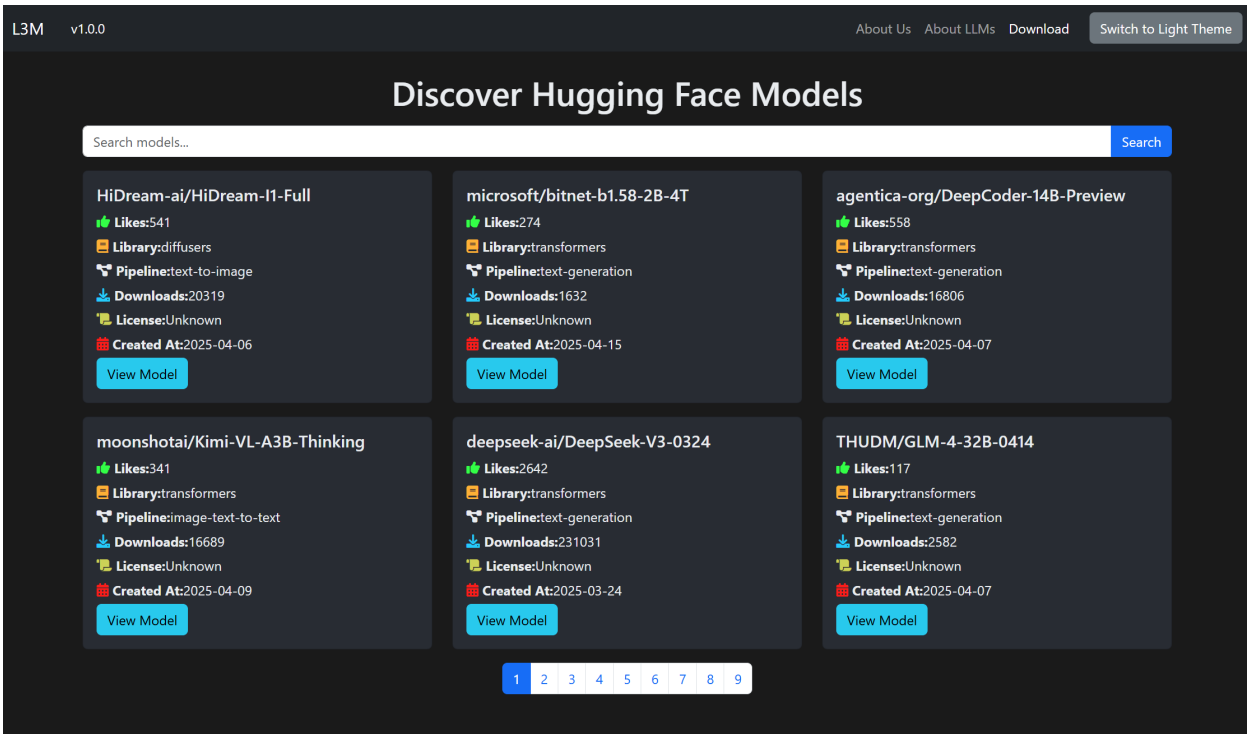


Figure 11.5: Website Home Page 4/17/2025

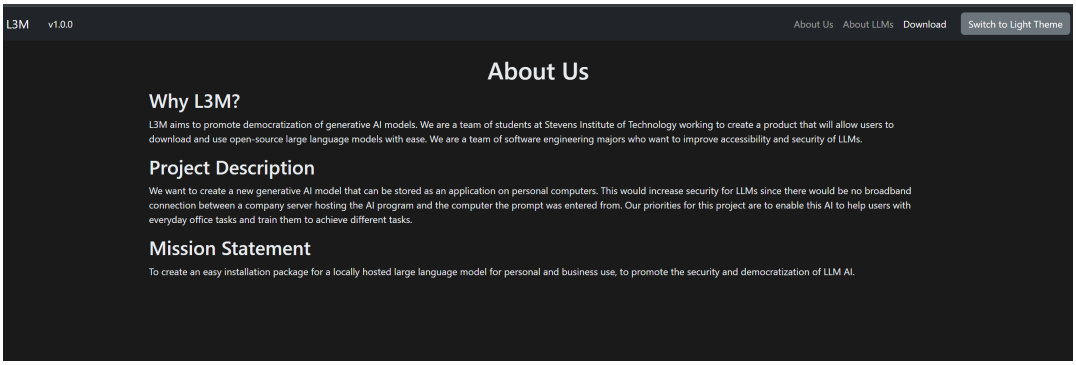


Figure 11.6: Website About Us Page 4/17/2025

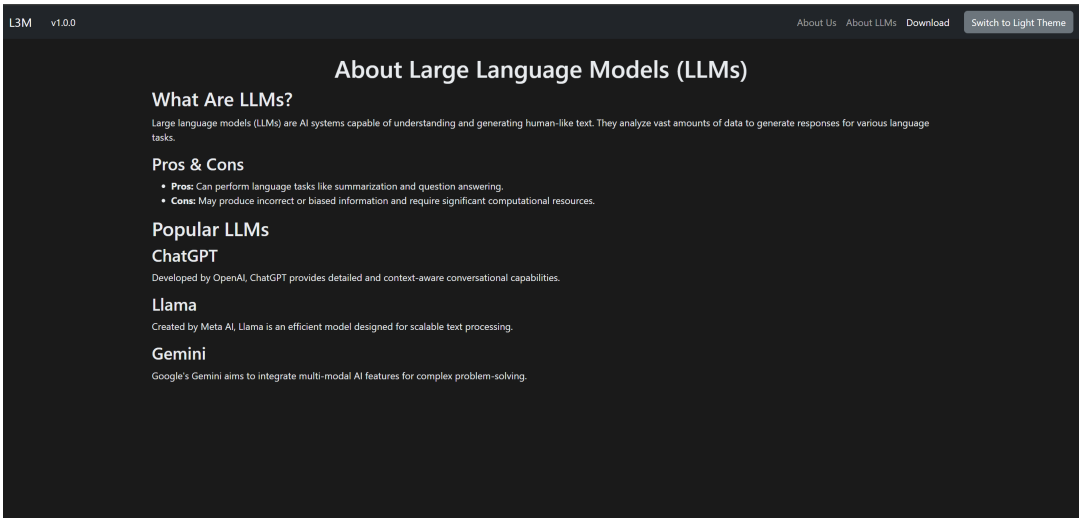


Figure 11.7: Website About LLMs Page 4/17/2025

Chapter 12

Logical View

–Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

12.1 Class Diagrams

In this chapter, we created the class diagram for our system that describes how the different classes in our system interact. It also displays some functions and variables that our classes use to function. This diagram may be updated as we create more classes for our systems and variables or functions for each class.

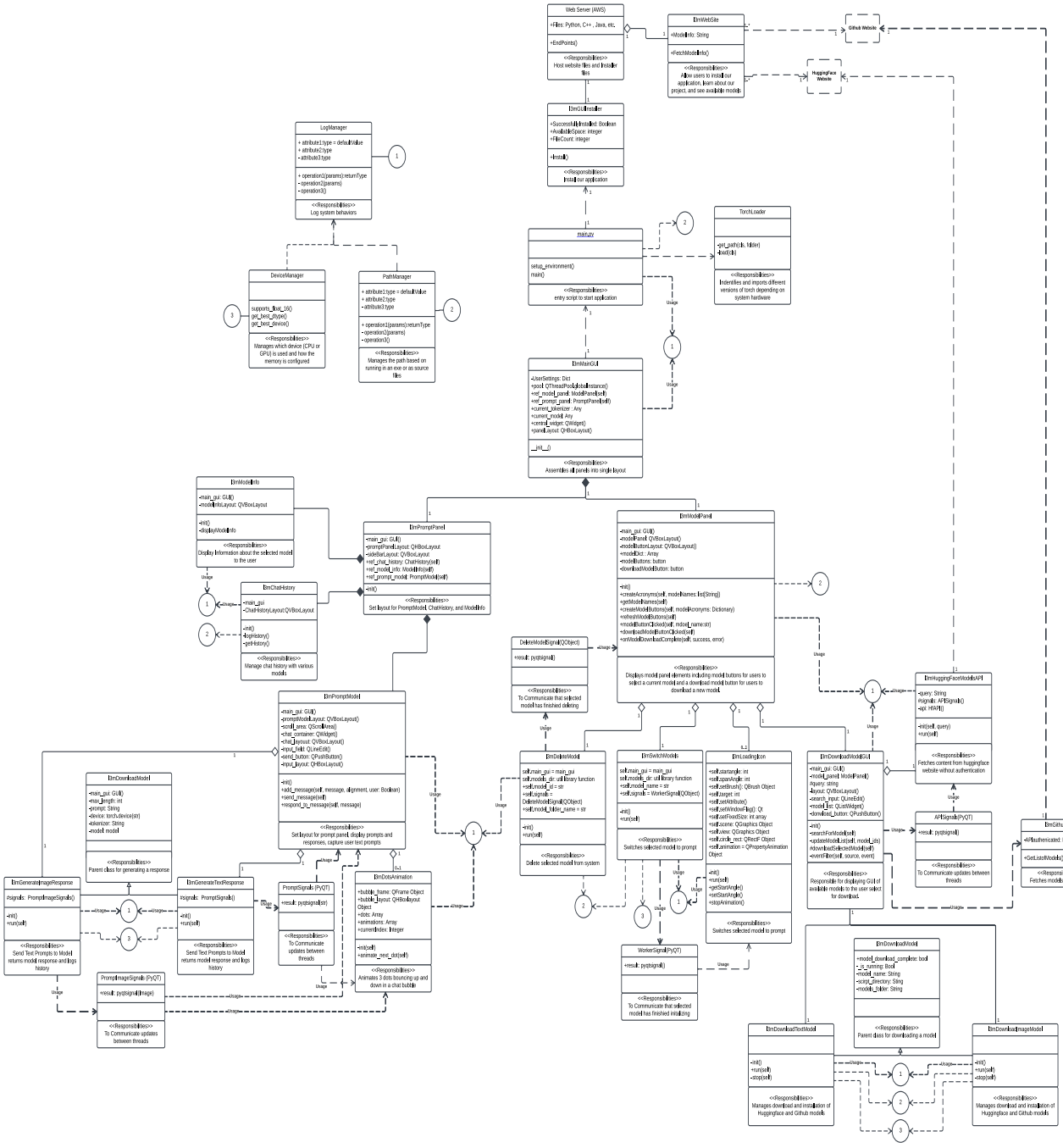


Figure 12.1: Class Diagram

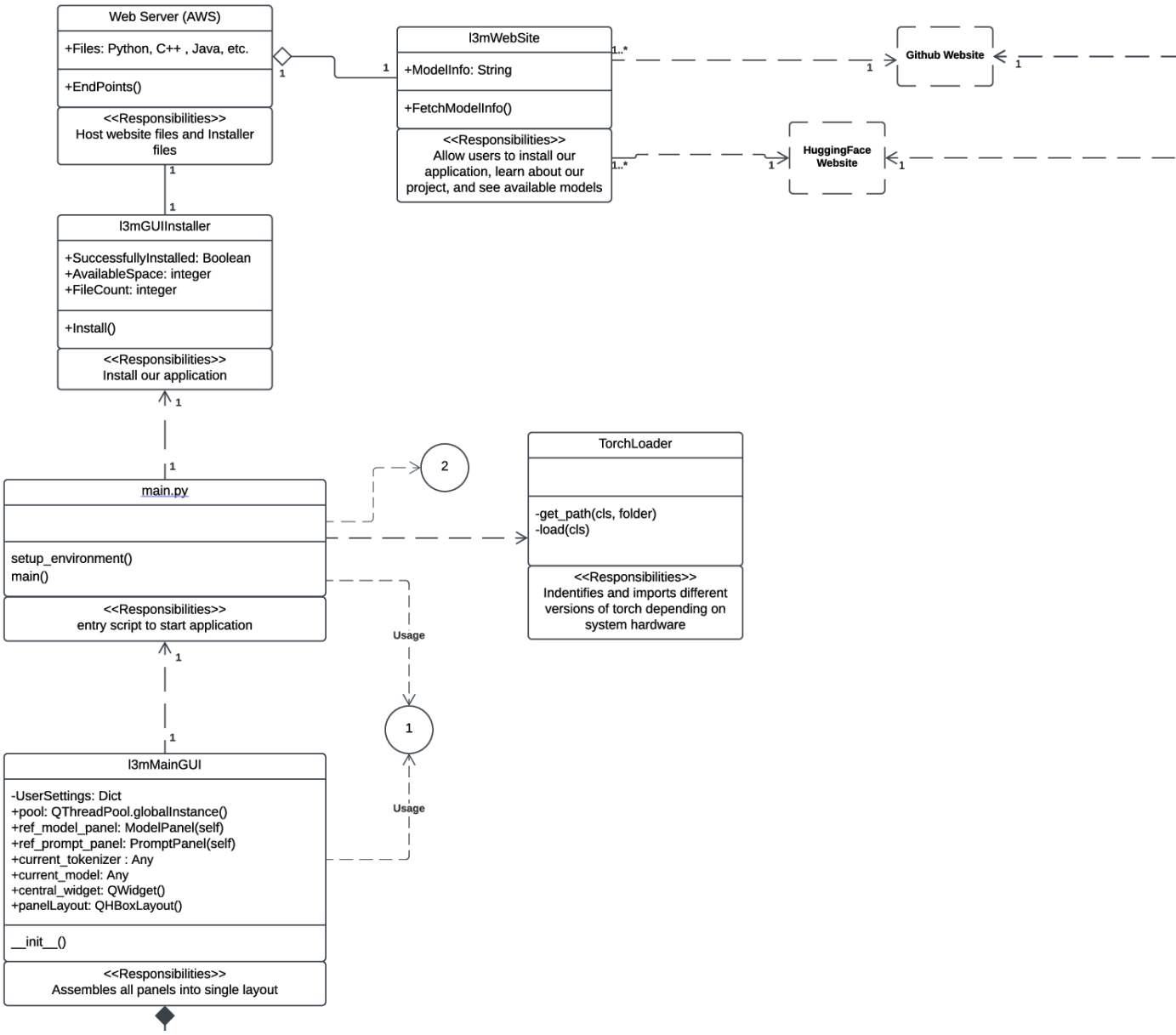


Figure 12.2: Class DiagramWeb Server and GUI Classes

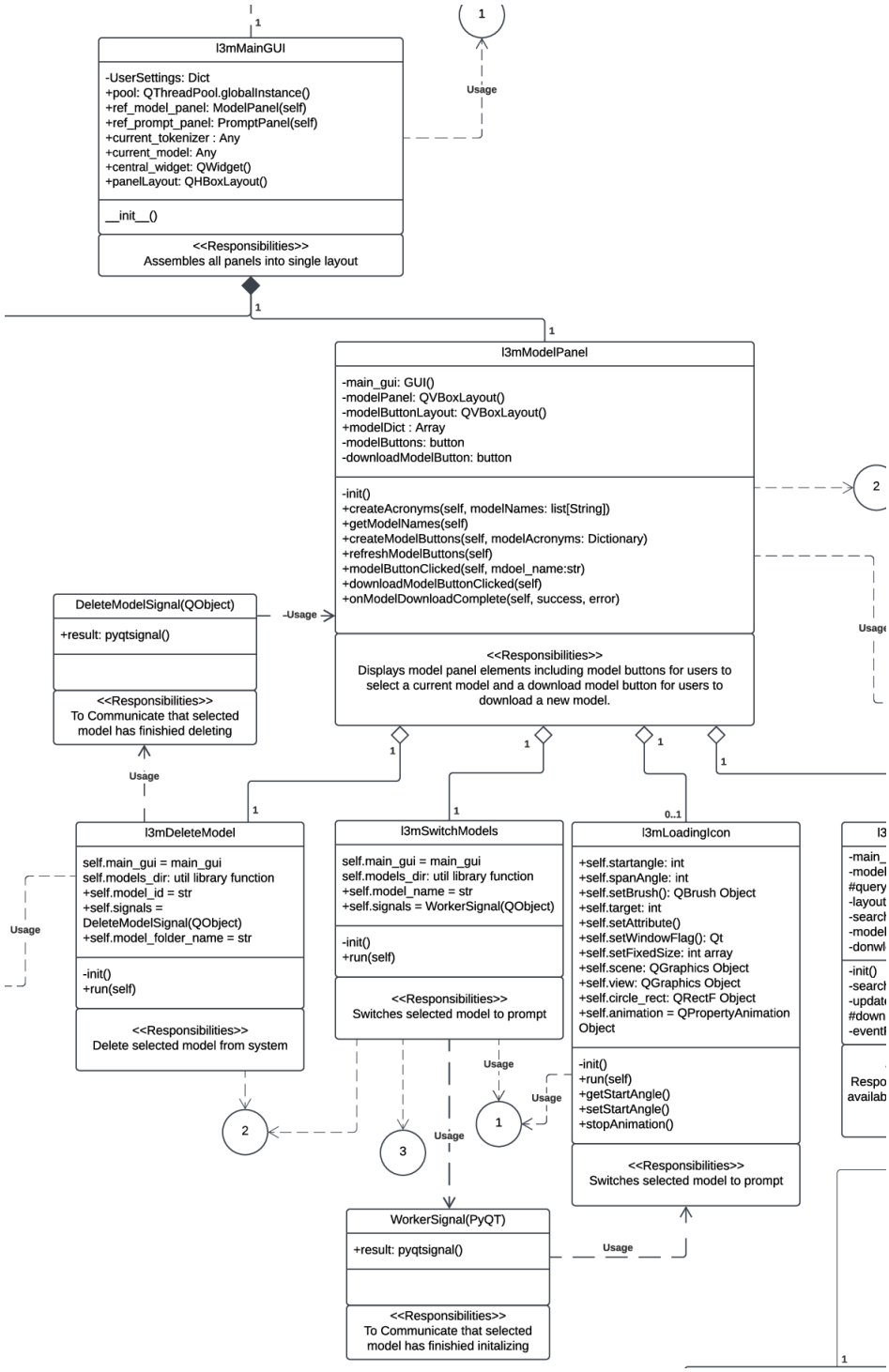


Figure 12.3: Class Diagram Main GUI Classes

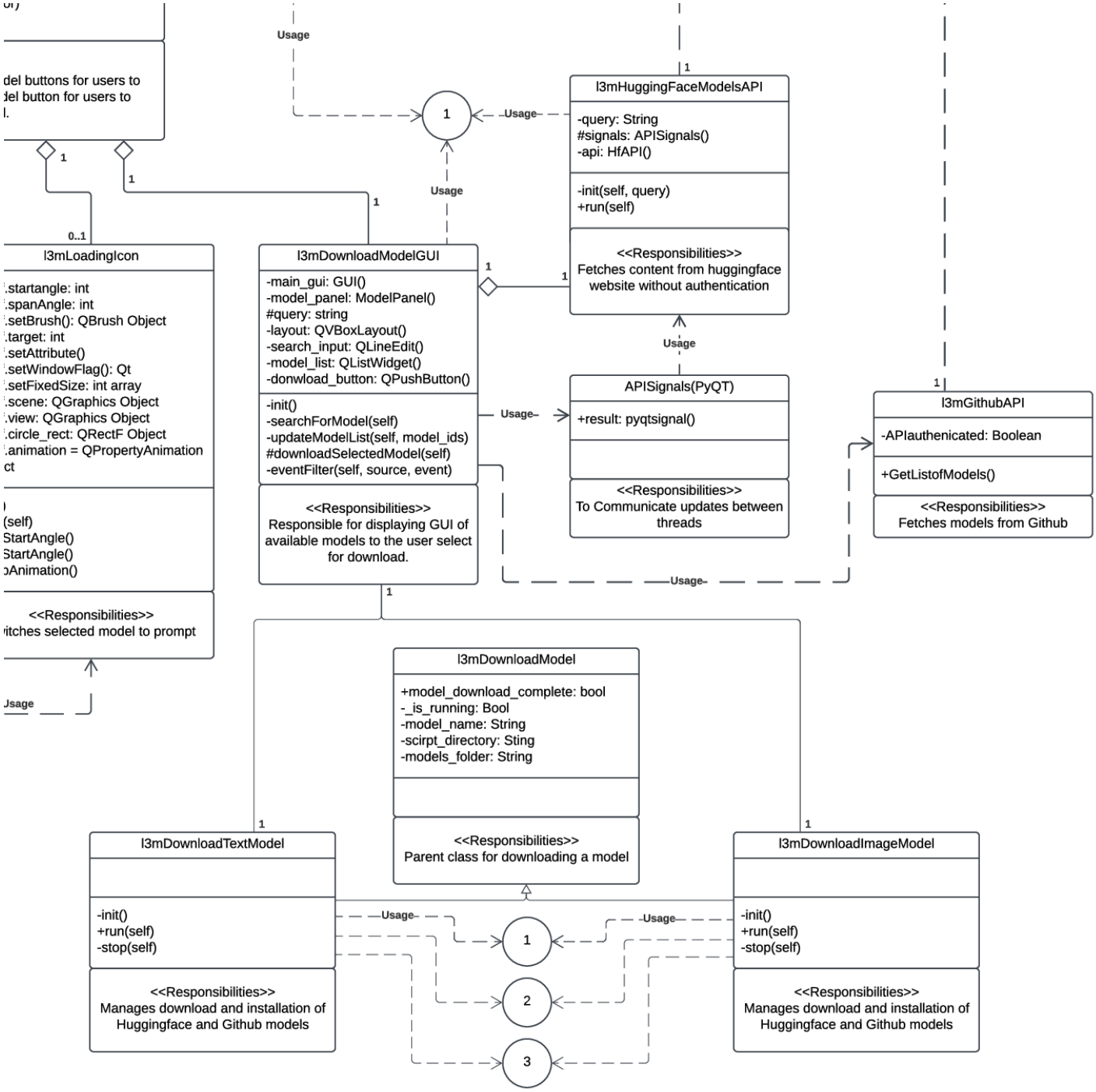


Figure 12.4: Class Diagram Download Model Classes

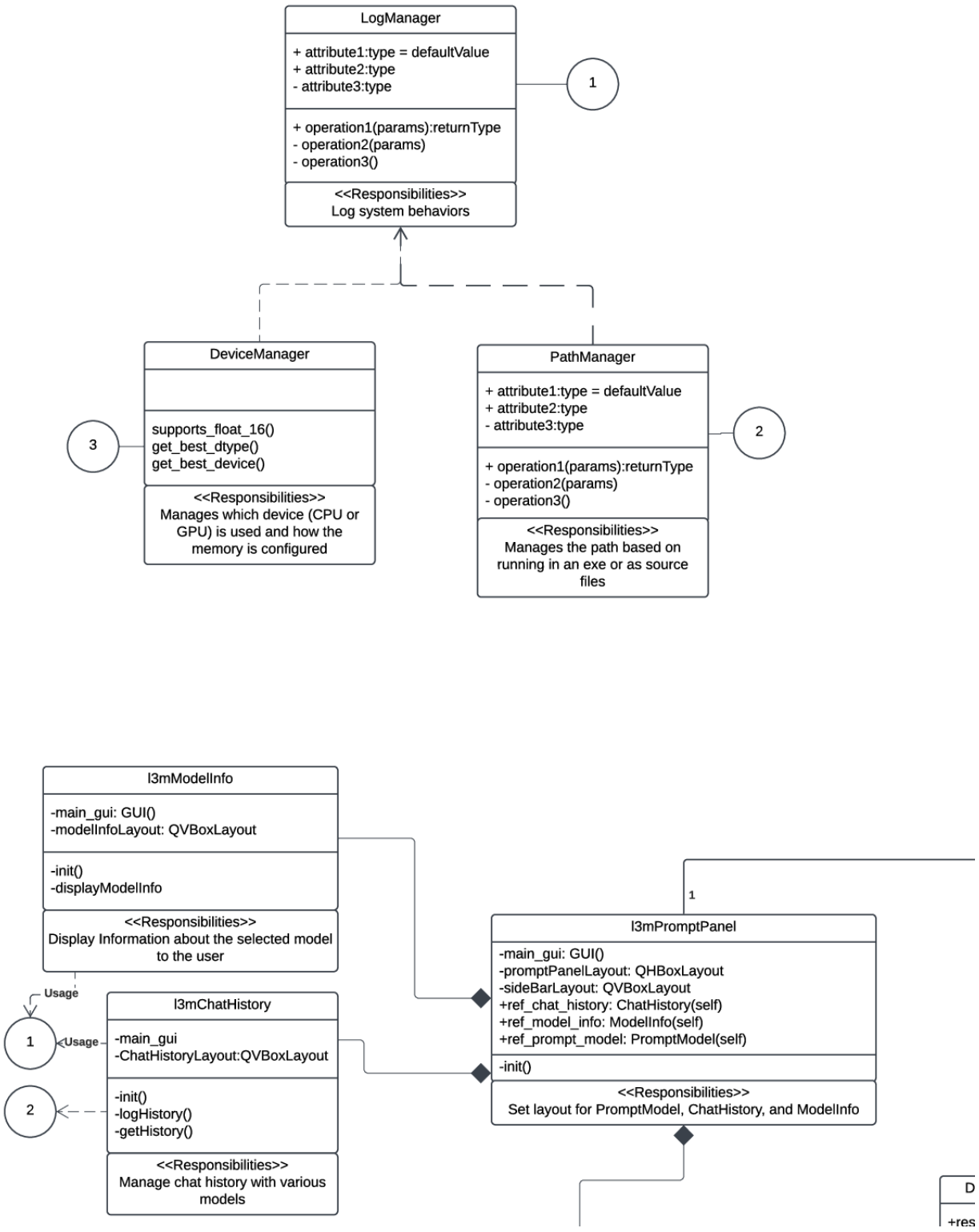


Figure 12.5: Class Diagram Utilities and Model Info

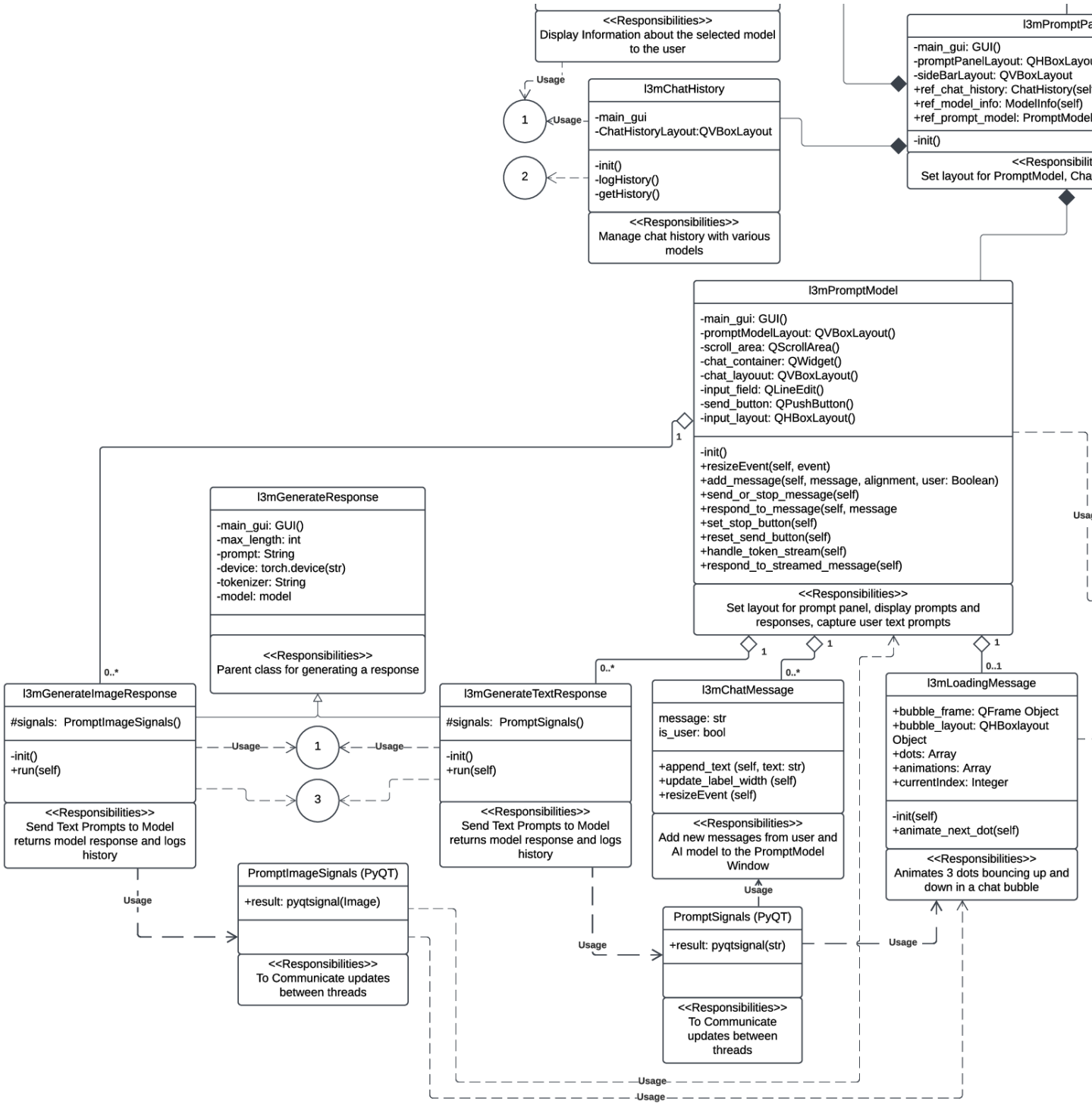


Figure 12.6: Class DiagramPrompt Model Classes

Chapter 13

Process View

Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

Below visualizes the activity diagrams for each use case of our system. Each use case depicted is labeled and referenced below each diagram. Zoom in to see details for each diagram.

13.1 Activity Diagrams

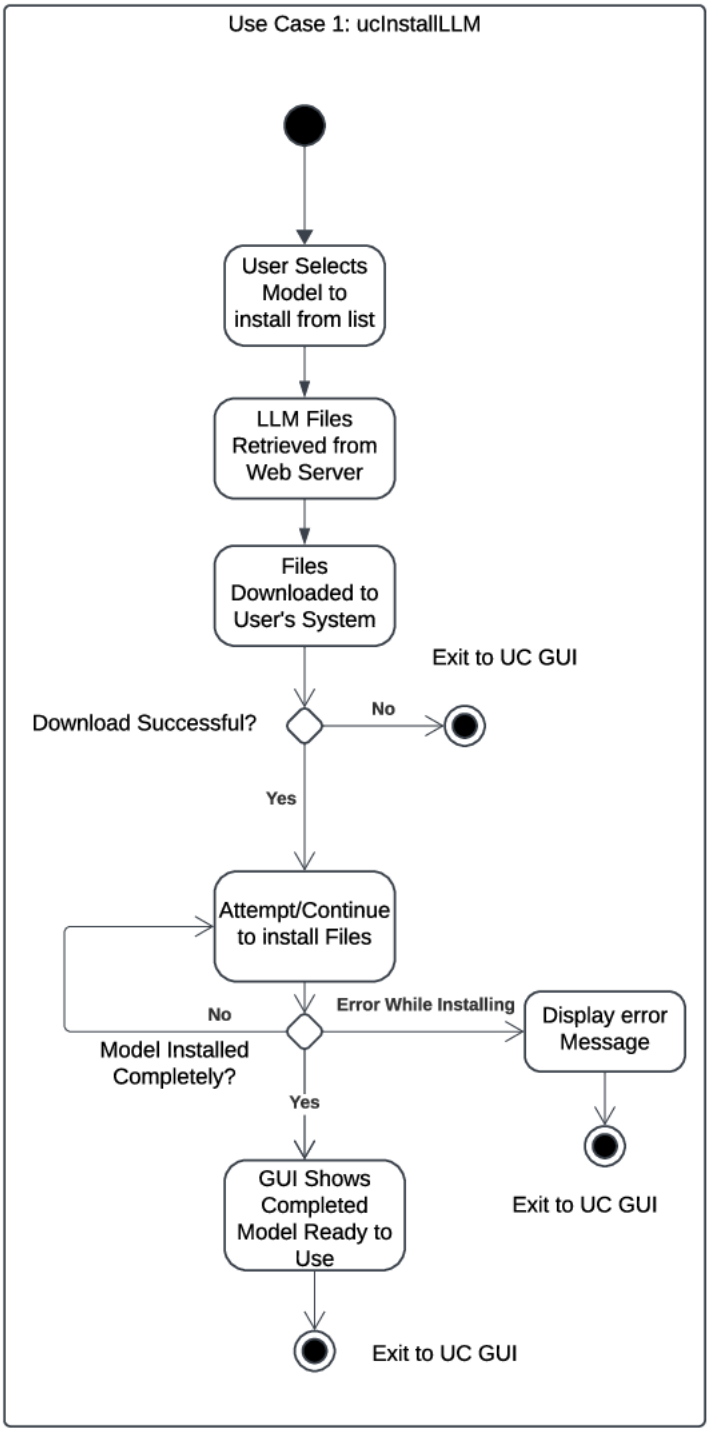


Figure 13.1: Activity Diagram for the Installer use case UC_1 exits to UC_7

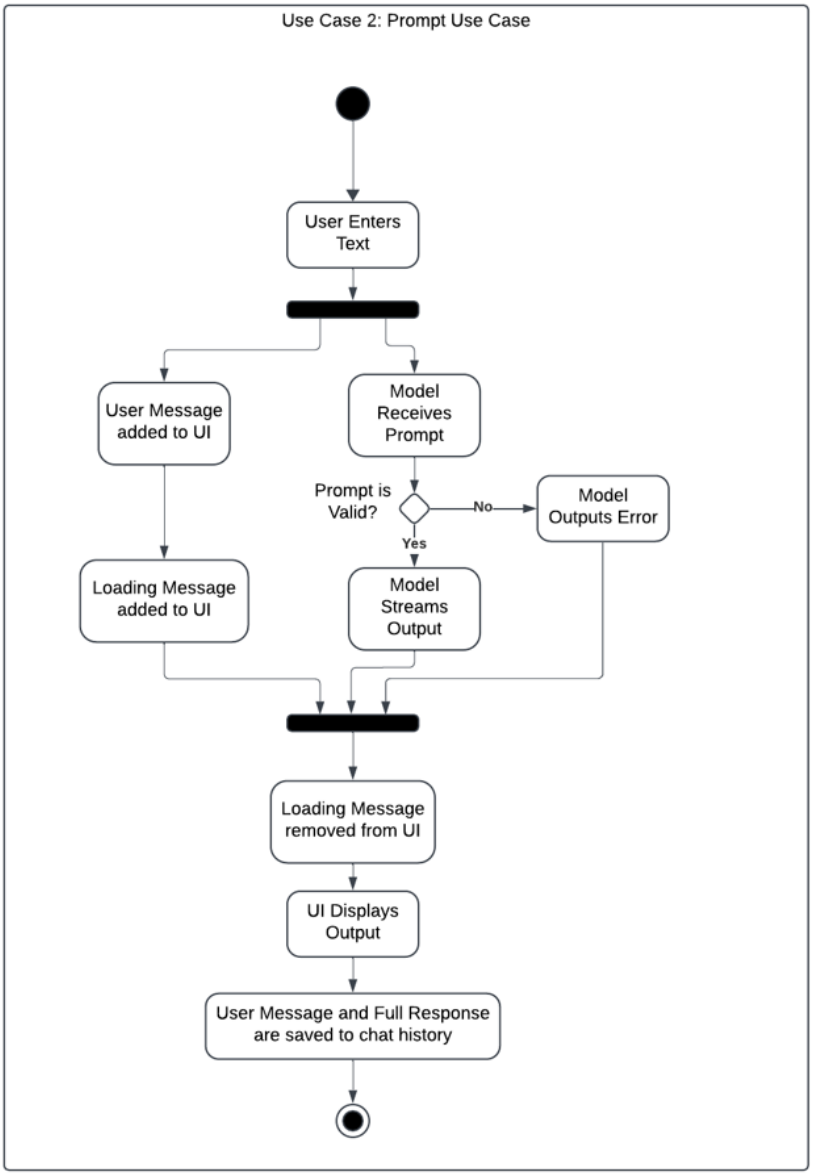


Figure 13.2: Activity Diagram for the Prompt use case UC_2

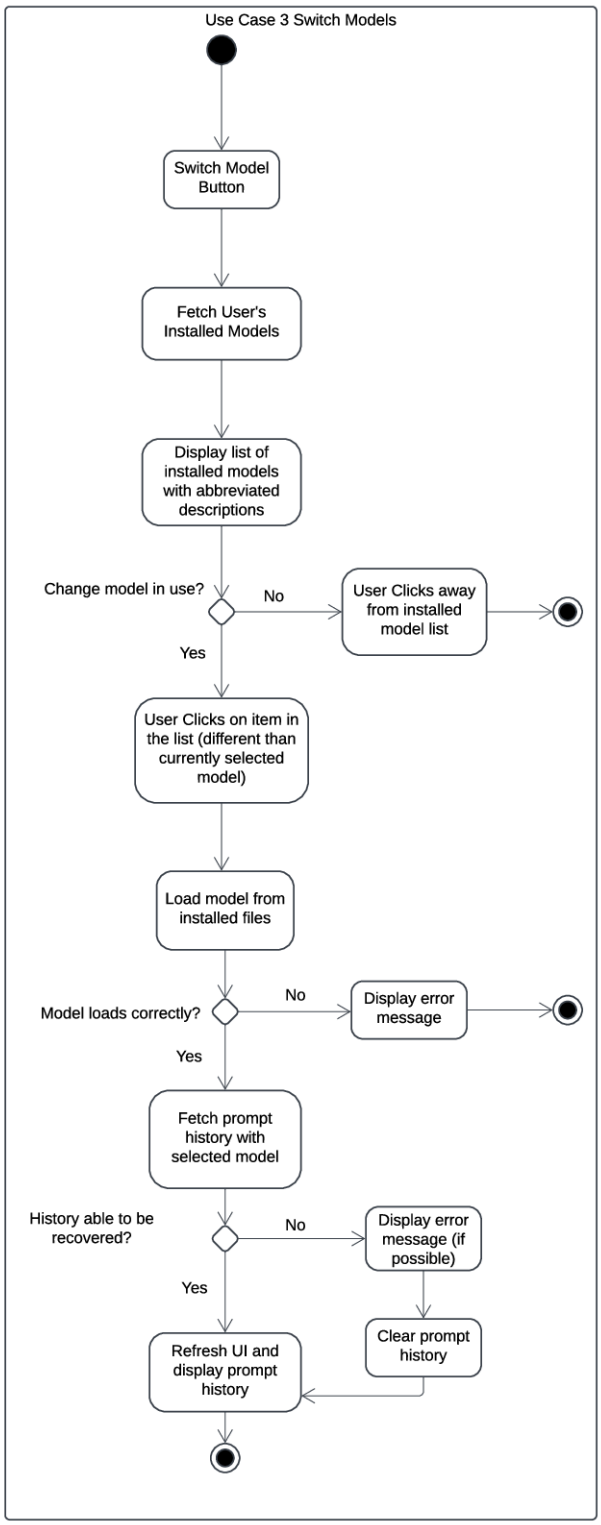


Figure 13.3: Activity Diagram for the Switch use case UC_3

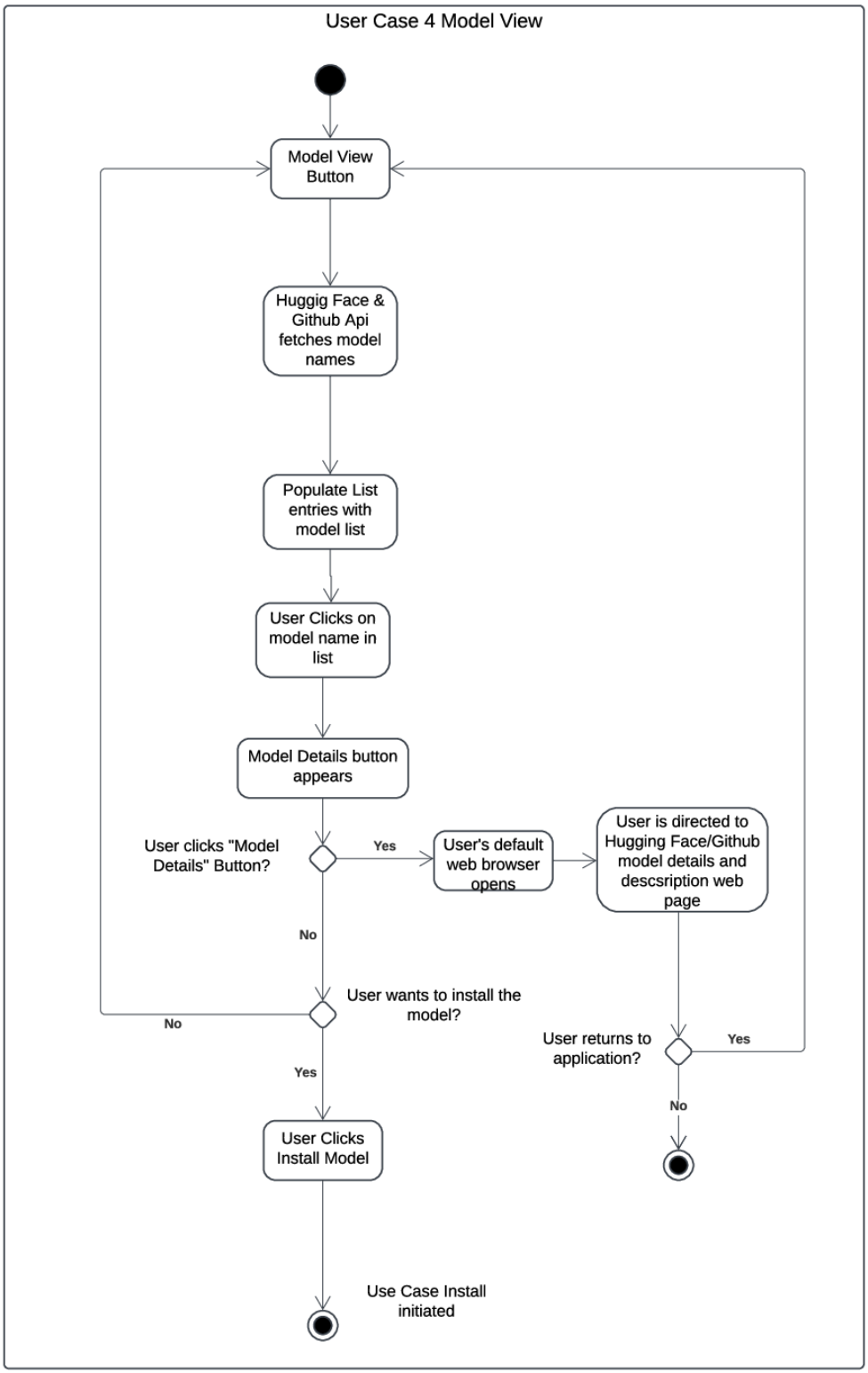


Figure 13.4: Activity Diagram for the Model View use case UC_4

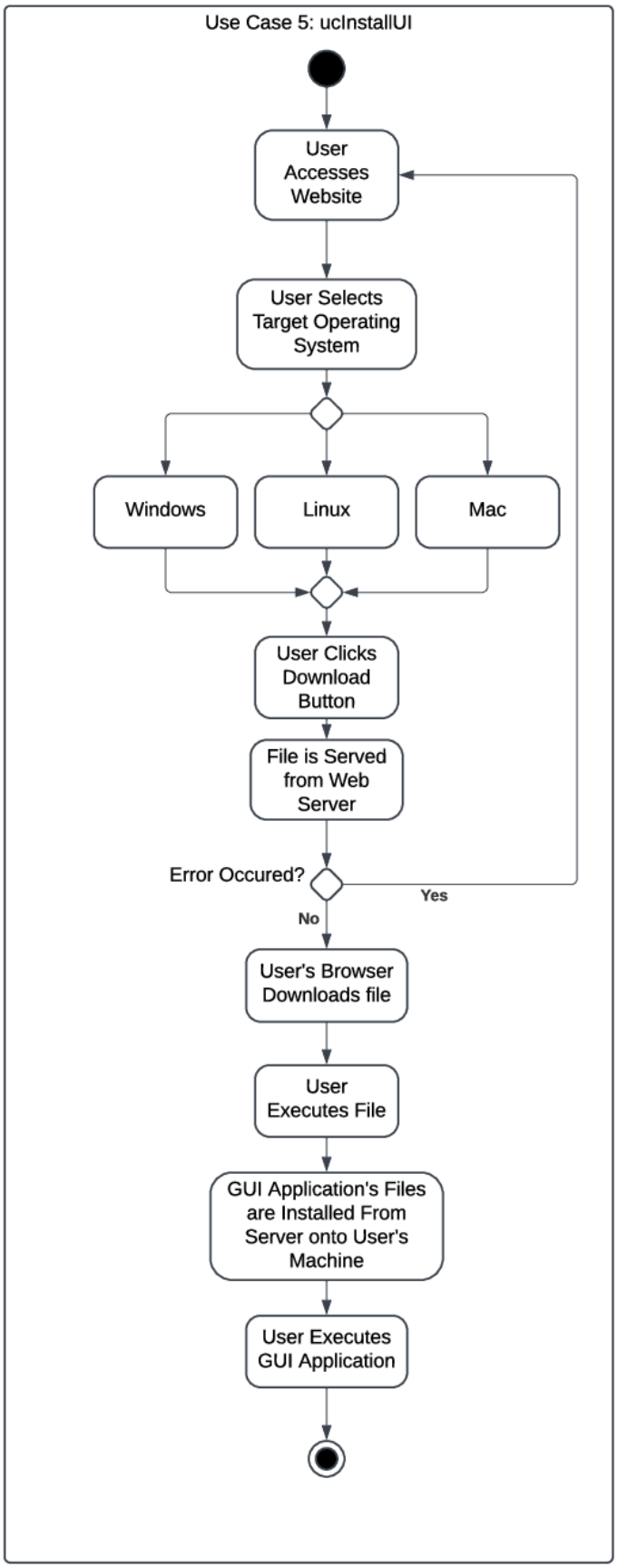


Figure 13.5: Activity Diagram for the InstallUI use case UC_5

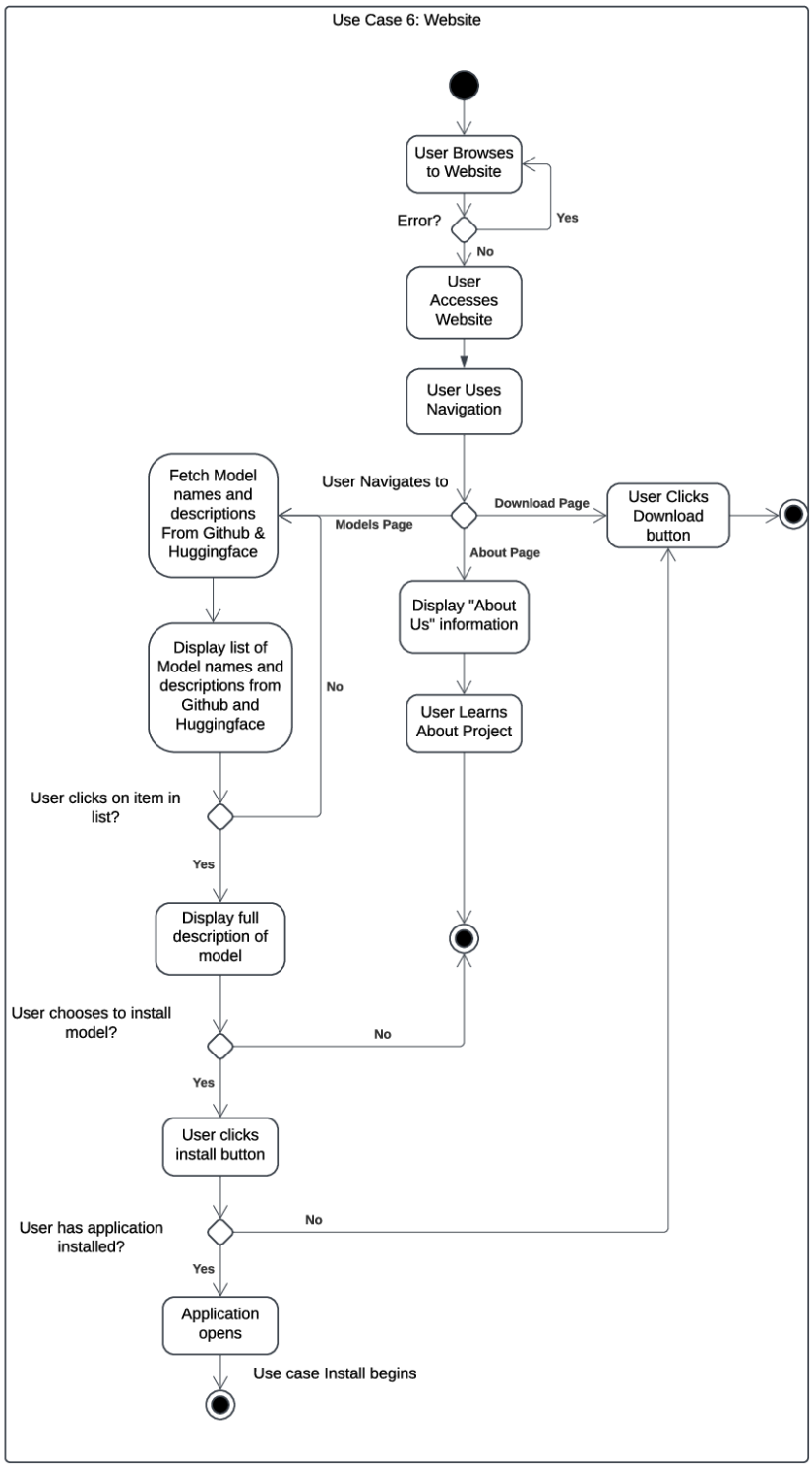


Figure 13.6: Activity Diagram for the Website use case UC_6

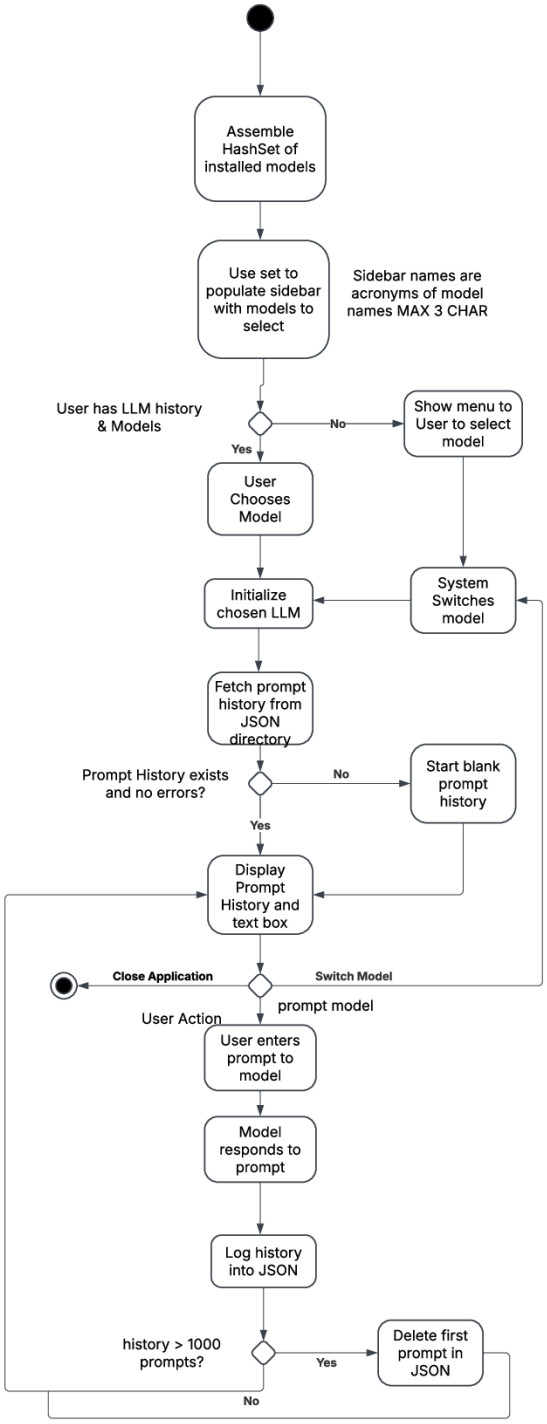


Figure 13.7: Activity Diagram for the main GUI use case UC_7

13.2 Sequence Diagrams

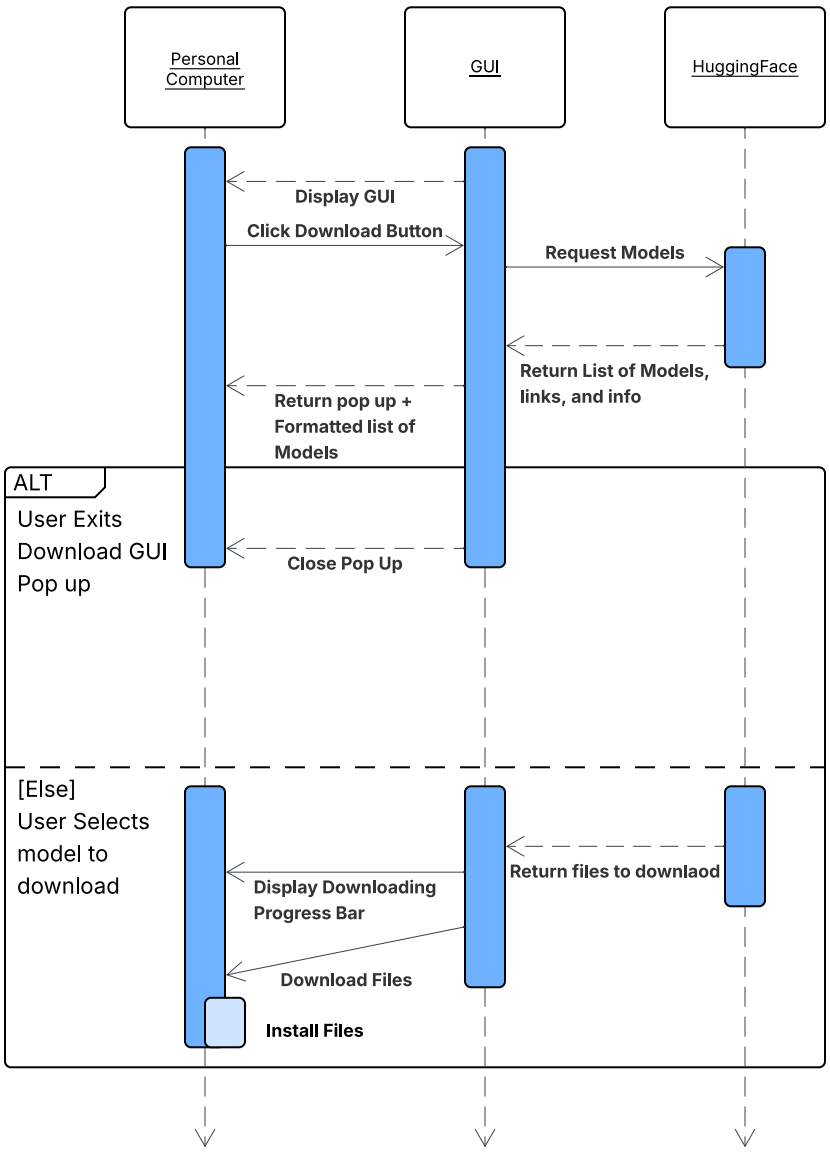


Figure 13.8: Install LLM Sequence Diagram from UC_1

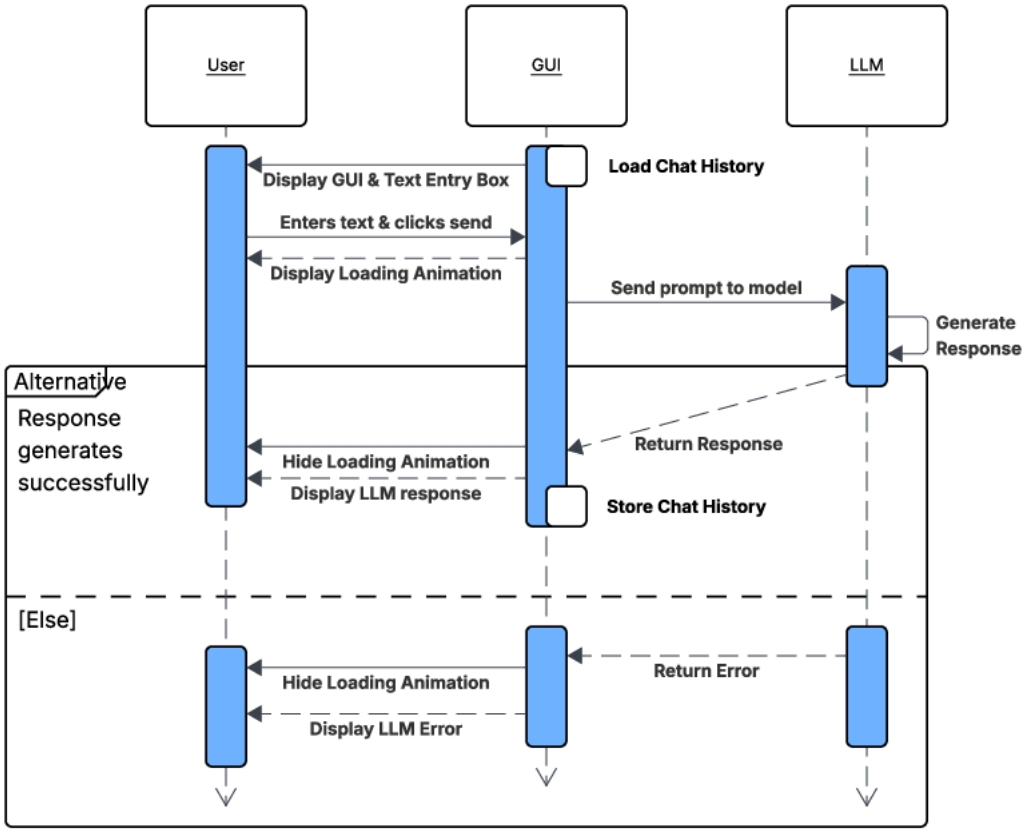


Figure 13.9: Prompt Model Sequence Diagram from UC_2

Chapter 14

Development View

Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

14.1 Package Diagram

Below we included the package diagram for our system. This describes the connections between packages that our Components use. We may alter the diagram in the future if more packages are needed.

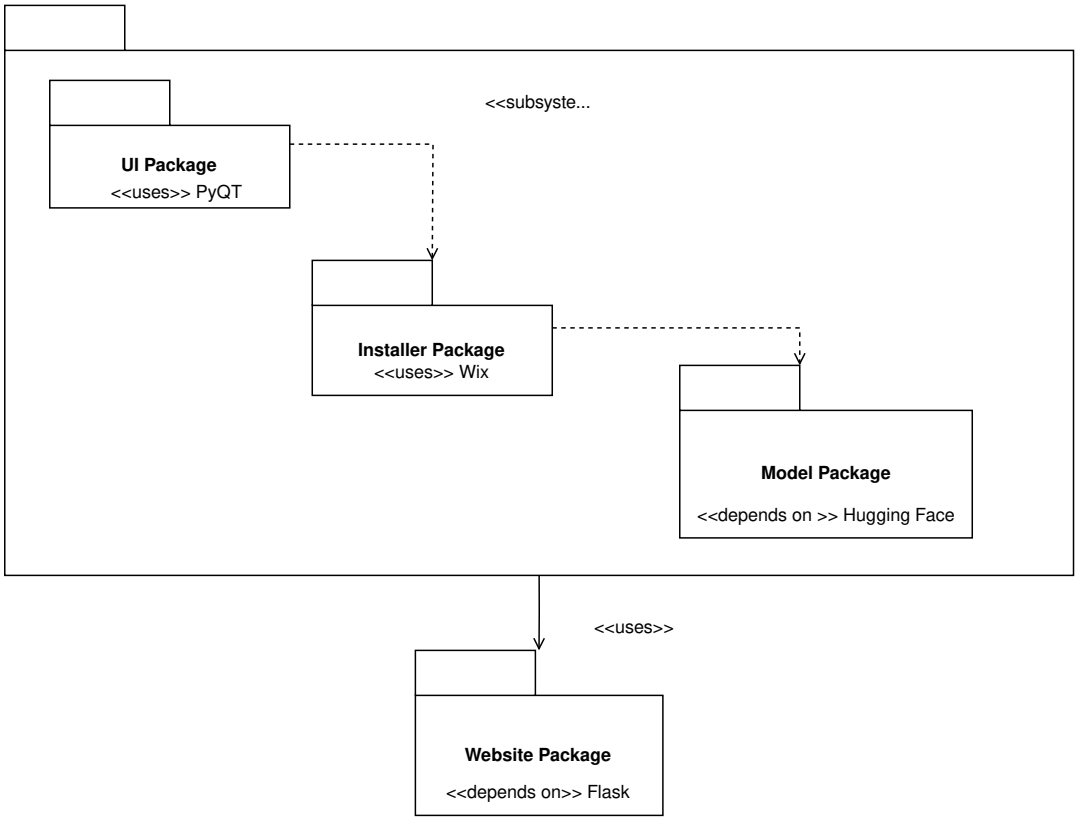


Figure 14.1: Package Diagram

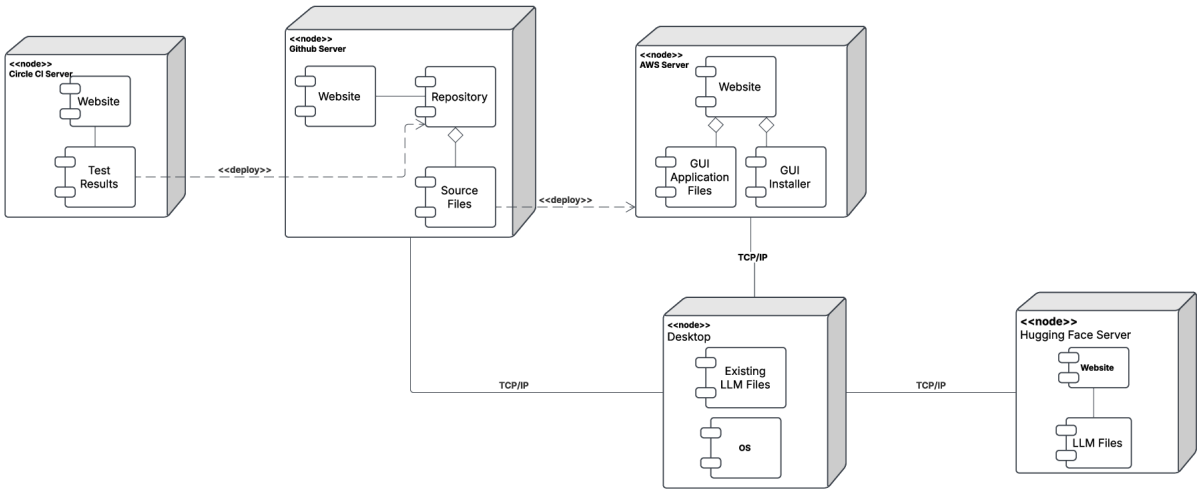


Figure 14.2: Deployment Diagram

Chapter 15

Physical View

–Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

15.1 Component Diagram

Below, our Component diagram is depicted for our system. As we work on the programming for the project, this may be updated to more accurately reflect how we designed the system . We expect the main components to remain consistent, but with possible different interface corrections.

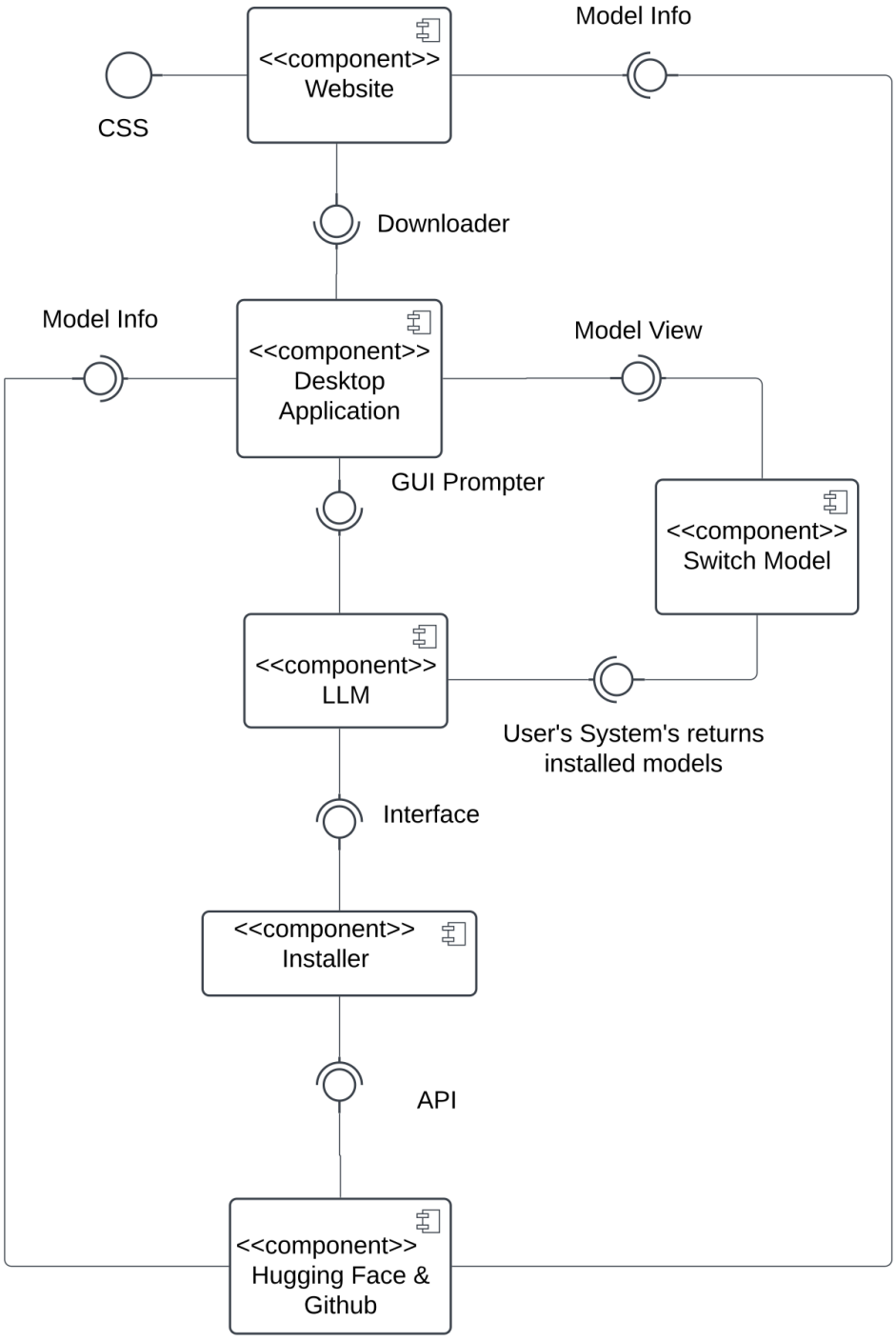


Figure 15.1: Component Diagram

Chapter 16

Prototype Demo Discussion

Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

16.1 Overview

In this presentation, we showcased our current progress in development. Prior to our demonstration, we built a prototype installer for installing a single LLM. The model used for this was GPT2. We were also able to demonstrate in the command prompt, prompting the model and successfully generating an output.

Following this, we presented our current progress on the website. Currently, on our website, we have Hugging Face Hub API running. With this, we can list and show various models available for download in the future. We also have the preliminary steps for installing the GUI application from the website. We have set up a function that allows the users to download files from our web server successfully. Lastly, on the website, we have an about us section highlighting details of the team, project description, and our mission statement.

16.2 Images of Website Prototype

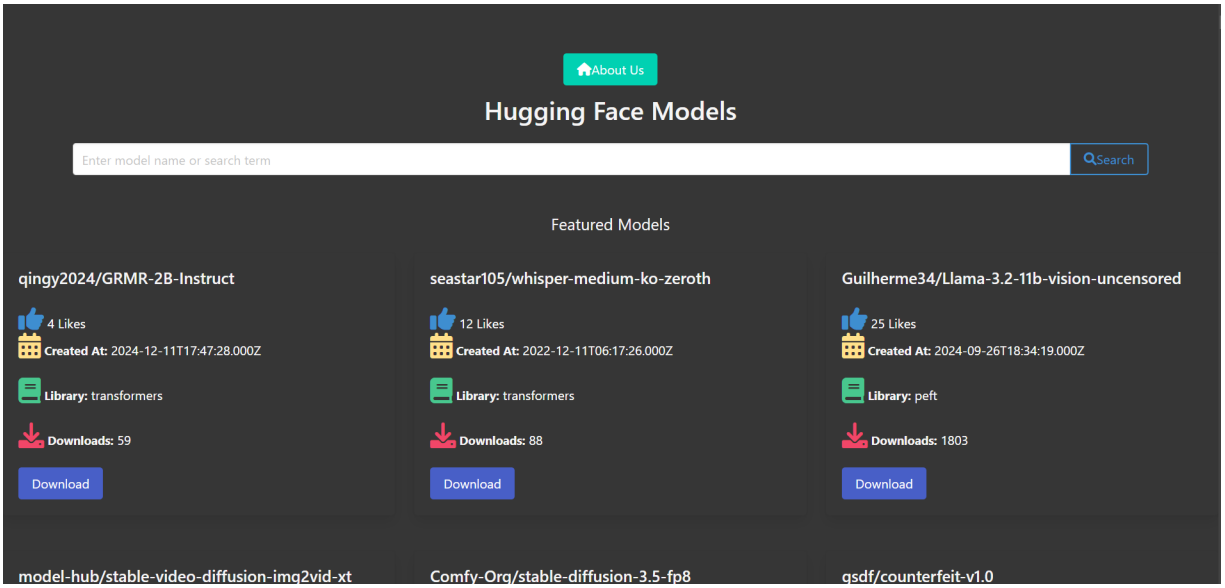


Figure 16.1: Model Search Page

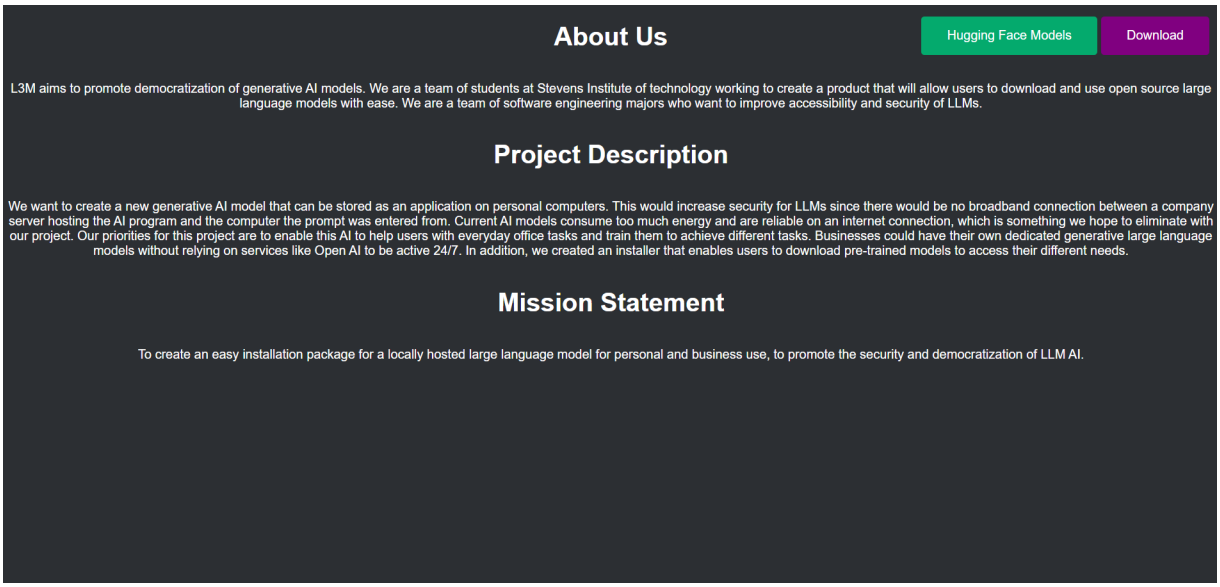


Figure 16.2: About Us Page

16.3 Video Demo

Demo link to video

16.4 Feedback and Discussion

Overall we are delighted with our current progress and demonstration. We have some of the primary functions of our project in place and running properly, which we successfully demonstrated to the class. With this, we have a solid foundation for the continuation of development for this project. From the class we received some useful feedback and have implemented it into our project plan. One student recommended that we implement a function that identifies the user's system specs and recommends models based on the system's capabilities. We have included this as a functional requirement that the system could have and we will focus on implementing it after other higher-priority functions are complete.

16.5 Updated Prototype Images

GUI: 11.4
Website Page 1: 11.5
Website Page 2: 11.6
Website Page 3: 11.7

16.5.1 Project Poster

Figure 16.3: Poster to be used in Innovation Expo

[6]

Link to zip Application Installer: https://drive.google.com/file/d/1va6nJQVo4ADoUVhpF44dJbKJNCav9FNY/view?usp=sharing

Chapter 17

Weekly Reports

– Nicholas Katzenberger, Ryan Hajtovik, Brandon Penman, and Carl Guillaume

17.1 Week Report 26 (4/25/2025)

17.1.1 What We Did

This week we focused on resdesigning our project poster to make it similar to a movie poster. This way it will draw the most attention for the judges and guests viewing our project. In addition, we made an elevator pitch for the annual Ansary Elevator Pitch Competition, and have a video of it posted privately on Youtube. Also, we did some further bug testing with our application. Elevator Pitch: https://youtu.be/5_kp65PdJzg
Poster:16.3

17.1.2 What We Will Do

This week we will focus on implementing some new features to show off at the Innovation Expo and the Project Beta Release Presentation. The features to be implemented are listed below. Also, we will release a beta version of our software to present to the class.

17.1.3 Action Items

- Allow users pc specs to help adjust the search for models they want to install
- Add image model functionality
- Improve Download Progress Bar to be more detailed
- Publish User manual on the github
- publish version of this pdf on the github
- Storing and Diplaying Prompt History
- Log debug files

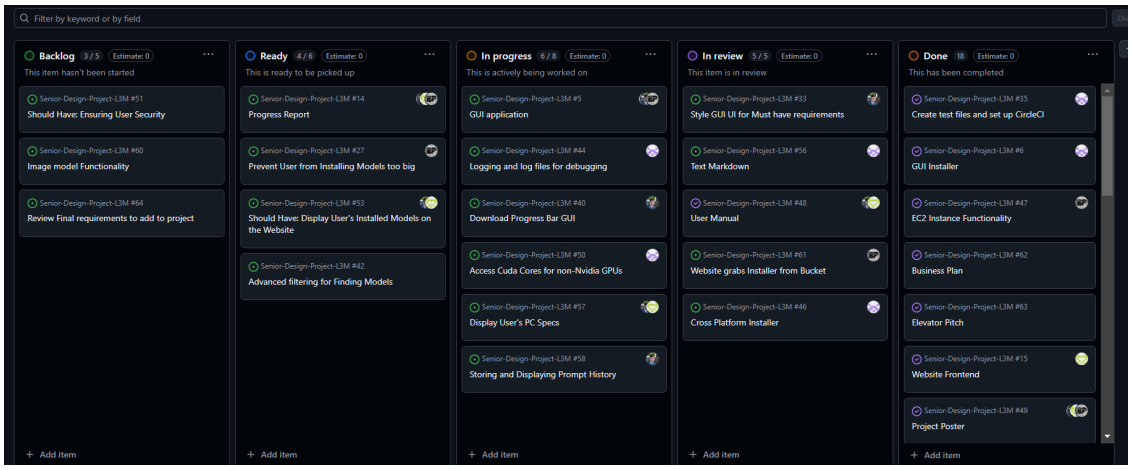


Figure 17.1: Kanban Board Week 26

17.1.4 Blockers, Issues, Risks

Blockers

Some of our new features reqiure other features to be completed first. For exapmle, implementing image models will need UI and download improvements before it can be considered complete.

Issues

There are no issues as of this date.

Risks

As time is ticking down until the expo, the team runs the risk of not getting all the desired features working in the application on time. This is concerning since we want to show our combined effort with our product, and don't want to fall short of our initial goals.

17.1.5 Sprint Screenshot

17.1.6 UML Diagrams

No new UML Diagrams this week, but sequence diagrams are being worked on for the other use cases.

17.2 Week Report 25 (4/18/2025)

17.2.1 What We Did

This week we focused on revising and editing our project poster, user manual, business plan, and improving this document. Plus, we also made a elevator pitch for our project that we intend to use in an entreprenuership competition. In addition, this chapter has been moved to the end of the document, and will serve as an appendix item in future updates.

17.2.2 What We Will Do

Next week we will likely focus on finalizing deliverables for our project and making adjustments to ensure that our project can have a seamless first official version release. This first release would likely come out near the time of the Innovation Expo, which we are going to demonstrate our product. In the below action items, we have some repeating items that were carried over from the previous week.

17.2.3 Action Items

- Review Poster, User Manual, Business Plan, and Elevator Pitch
- Finish Download Progress Bar
- Get to review stage of User Manual
- display user's pc specs on the gui and website
- Establish requirements to be done by May 9th (Innovation Expo)
- Submit new Elevator Pitch for competition

17.2.4 Blockers, Issues, Risks

Blockers

Some of the other things we wanted to do with the product cannot be started until some of the above features are implemented, as well as those not listed.

Issues

There are a few minor bugs that impact the performance of our product, but will be ironed out by the time of the final delivery.

Risks

We run the risk of not getting all the requirements that we have lised in this document for the project done by the day of the expo. This is expected, but we were able to get all of our MustHave and ShouldHave requirements and a decent amount of WouldHave requirements completed over the course of the semester.

17.2.5 Sprint Screenshot

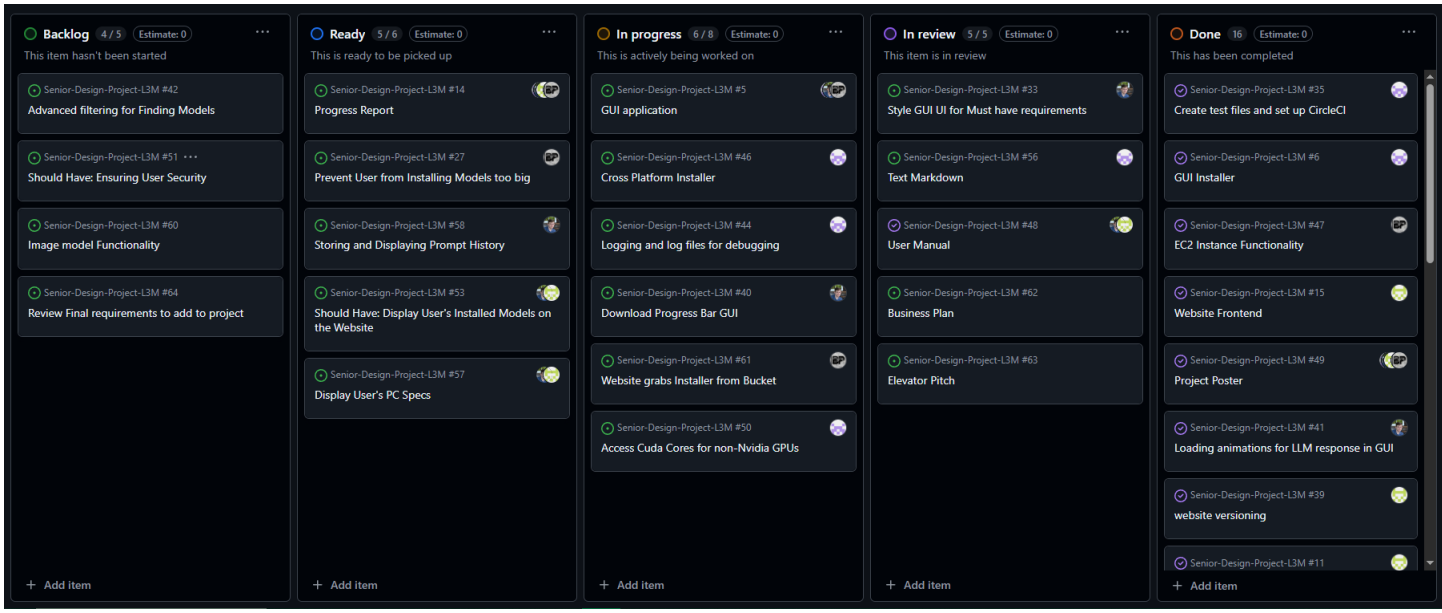


Figure 17.2: Kanban Week 25

17.2.6 UML Diagrams

Updated Prompt Model Activity Diagram 13.2

17.3 Week Report 24 (4/11/2025)

17.3.1 What We Did

This week we focused on presenting our Alpha test build to the class and professors and our Circle CI build. Also the text markdown feature is mostly completed and functioning. The user manual is still a work in progress because the team wants to have an established UI that will be the same in the 1.1 release as in the manual. We want to make 1 or two changes to the UI before we can complete the entirety of the manual. However, we did create sequence diagrams, and update our activity diagrams for this week. We also presented an elevator pitch for our project for another class this week.

17.3.2 What We Will Do

This week we will implement the necessary UI changes to finish the user manual. We will also start implementing the prompt history log for users to reference on future initializations of the application. We still need to implement the download progress bar, fix the mac OS version, and display the user's pc specs.

17.3.3 Action Items

- Finish Download Progress Bar

- Get to review stage of User Manual
- log User's pc specs
- fix errors with Mac OS version
- initialize bucket for website to pull application installer from

17.3.4 Blockers, Issues, Risks

Blockers

We need to get the Mac OS version working before we can release our 1.1 version of the application. The UI must be complete in order to take screenshots for the User manual

Issues

There are no issues currently.

Risks

We run the risk of not being able to complete all of our requirements listed in the Requirements Chapter 8. Additionally, the website bucket for our application installer may cause more issues as we test its implementation to the website.

17.3.5 Sprint Screenshot

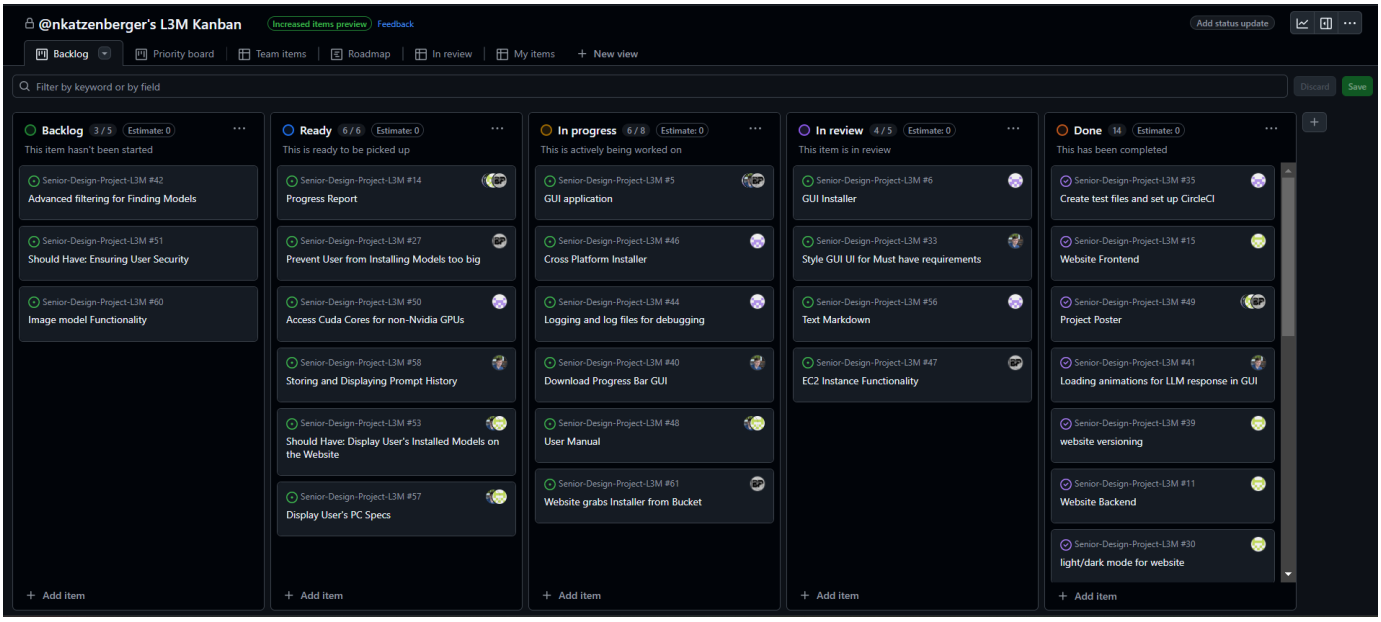


Figure 17.3: Kanban Week 24

17.3.6 UML Diagrams

Created 2 Sequence Diagrams 13.8 and 13.9

17.4 Week Report 23 (4/3/2025)

17.4.1 What We Did

This week we created a poster for the Innovation Expo, improved the UI, worked on improving the text markdown of the LLM output, and improved the performance on models by using computing power on the user's gpu. We also improved the website by re-including the icons shown in 16.1 that had been lost in a past commit. Also, we reworked the class diagram again to include the new classes we intend to build in the next couple of weeks, such as preparing to implement image generation models.

17.4.2 What We Will Do

This week we will finish up the text markdown implementation, add indication for the download progress, and begin working on logging the prompt history. Text markdown allows the LLM to have a nicely formatted response when it outputs code or multiple paragraphs of a long response. Also, the user manual needs to be finished

17.4.3 Action Items

- Finish up text markdown
- Finish User Manual
- Implement method to log user's PC specs
- implement download progress bar
- log prompt history
- ensure that Mac OS version works properly

17.4.4 Blockers, Issues, Risks

Blockers

Since we have moved on to make a version that works with Mac OS, we need to ensure that new changes work on both versions before moving on to the next issue. Also, the user manual needs to have a complete version before we can start improving it.

Issues

The text markdown feature seems to have some issues at the moment with it's display, therefore we need to fix these issues before they cause more problems with the LLM's output.

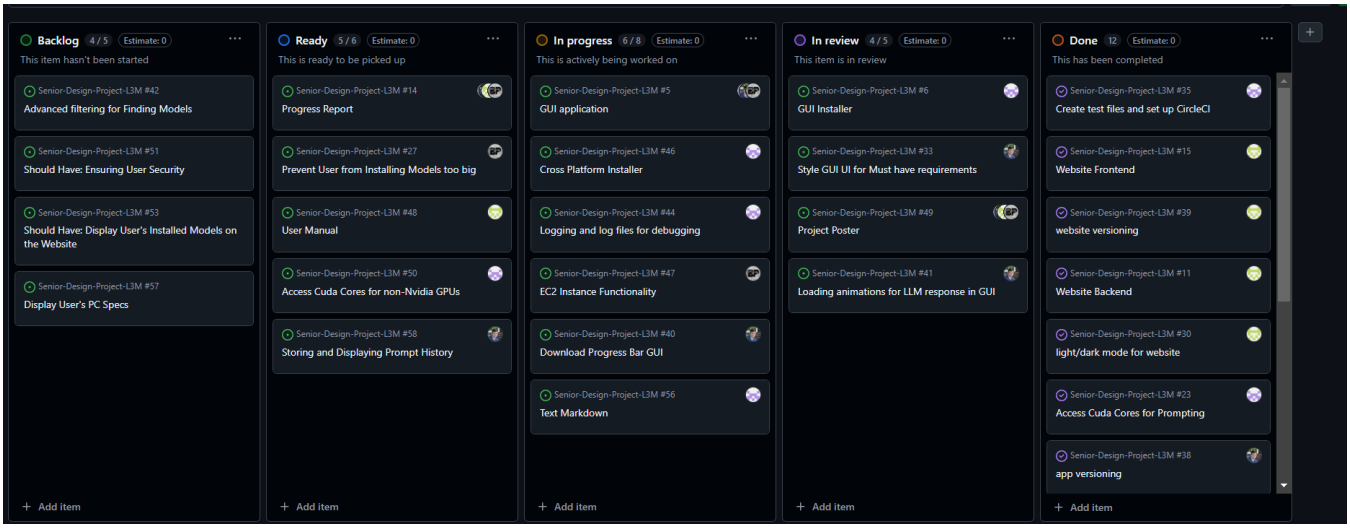


Figure 17.4: Kanban Week 23

Risks

Time until the Innovation Expo is becoming short, and we have a lot of ideas to implement into the application. However, we run the risk of not being able to develop everything on time.

17.4.5 Sprint Screenshot

17.4.6 UML Diagrams

Updated Class Diagram 12.1

17.5 Week Report 22 (03/28/2025)

17.5.1 What We Did

This week we focused on initializing our website server, refining the project document, creating a delete function for the models, improving the UI, and optimizing the performance of the application. Ryan found a way to dedicate the system's resources to have the GPU handle all computations regarding the LLM's response to a prompt. This currently works best on NVIDIA GPUs, but we are looking into other methods to make this feature more universal. This optimization feature significantly improved the response time of LLMs. The EC2 web serveron AWS is working for our application but is not live at the moment. We need to conduct testing before we can make the domain public. Loading animations are being implemented into the system with a loading spinner being added when the user switches models, and Nick is currently working on implementing a different loading animation for the LLM's response loading animation. A delete model function has been added, which allows users to easily uninstall a selected model.

17.5.2 What We Will Do

This week we will focus on creating our poster for the Innovation Expo, continue improving optimization, improve UI interactivity, create a user manual and test the website performance. This will enable us to move on to ShouldHave requirements and further improve our product.

17.5.3 Action Items

- Finalize Poster
- improve optimization for hardware other than Nvidia GPUs
- Create User manual
- Test Website Performance
- Continue improving UI

17.5.4 Blockers, Issues, Risks

Blockers

In order to move to our should have requirements, all of our must have requirements need to be fully complete.

Issues

We have limited access to hardware that does not use NVIDIA GPUs, and we may need to borrow a computer with that hardware.

Risks

Currently no risks, other than possibly not being able to complete all features planned before the Innovation Expo.

17.5.5 Sprint Screenshot

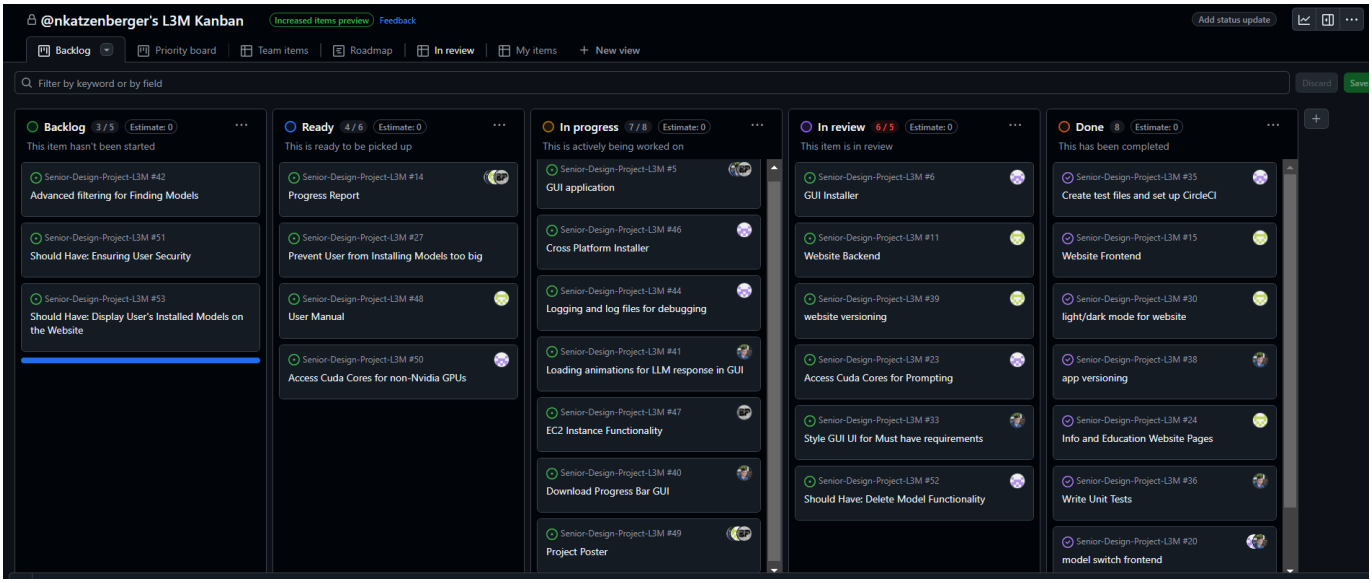


Figure 17.5: Kanban Board Week 22

17.5.6 UML Diagrams

Updated Class Diagram 12.1

17.6 Week Report 21 (3/13/2025)

17.6.1 What We Did

This week we presented our progress to the class including an updated class diagram, new deployment diagram, new and improved version of our website, and test results from Circle CI. Additionally, we released a 0.1.0 version on our Github page. This week we focused on improving the UI elements and establishing a method for committing releases.

17.6.2 What We Will Do

The team plans to improve the user experience by adding a feature to uninstall models, creating settings to customize the accuracy of the output for the models. Setting up AWS to host our application and figuring out how to reduce the size of our installer so users can download it from our website. Also the installer's size is too large to be hosted on Github at the moment, so we need to figure out a different place online where it can be hosted.

17.6.3 Action Items

- "Uninstall" model functionality

- Host web server on AWS EC2 with attached Elastic Block Store volume
- research installer size reduction
- improve UI features

17.6.4 Blockers, Issues, Risks

Blockers

Creating new functions for our application or website may take time and delay the release of our next version.

Issues

The installer is larger than anticipated which makes keeping track of versions on Github difficult, we need to find an alternative method of sharing these files or otherwise find a way to reduce the size of the installation file

Risks

Setting up the EC2 instance may change the way we need to work our application if the Hugging Face API refuses to connect to our Linux instance. However, we would be able to mitigate this risk by redeploying a new version with a different structure through git to fix the issues.

17.6.5 Sprint Screenshot

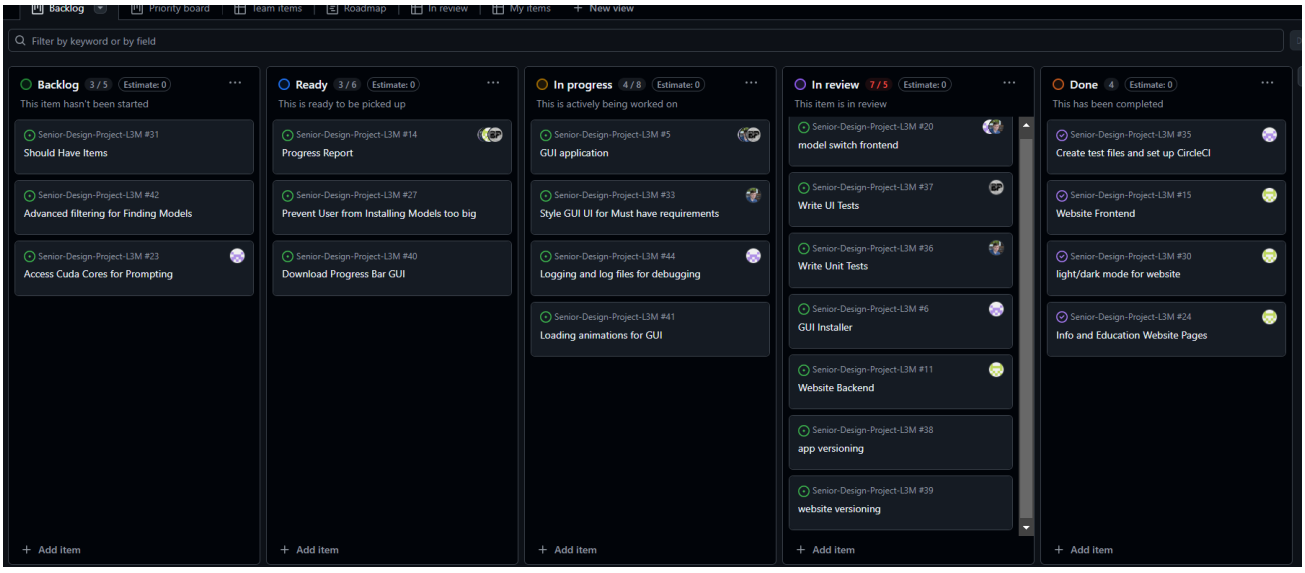


Figure 17.6: Kanban Week 21

17.6.6 UML Diagrams

No new UML diagrams

17.7 Week Report 20 (3/7/2025)

17.7.1 What We Did

This week we focused on writing tests, working on the slides for presenting next week, creating the deployment diagram; and working on creating the installation package.

17.7.2 What We Will Do

Next week we will primarily be finishing the slides and presenting. We will also finish writing tests and hopefully achieve release 1.0 for our versioning. We have moved adding the uninstall feature until after our 1.0 release so that we can really focus on having a minimal viable product before we leave for break while we are all still on the same page. In addition, we need to implement and test the GUI installer for local PCs.

17.7.3 Action Items

- Present slides
- Add 'delete' functionality to model list
- Finish tests
- Finish GUI installer
- Release 1.0 for desktop application

17.7.4 Blockers, Issues, Risks

Blockers

There are no blockers currently

Issues

There are no issues currently

Risks

If we do not finish version 1.0 by the end of next week, this goal will be pushed into spring break.

17.7.5 Sprint Screenshot

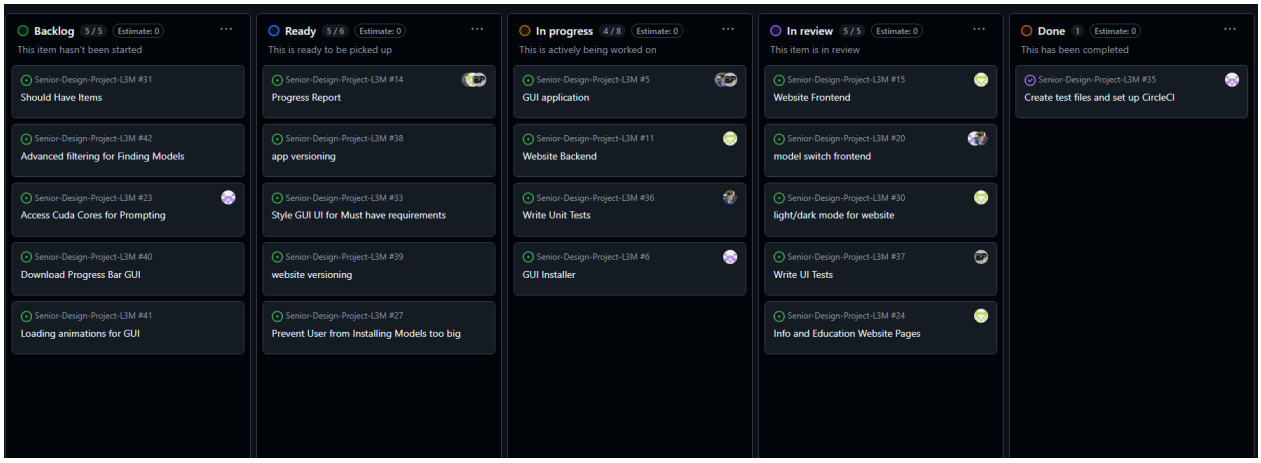


Figure 17.7: Kanban Week 20

17.7.6 UML Diagrams

Updated Class Diagram 12.1
Created Deployment Diagram 14.2

17.8 Week Report 19 (02/26/2025)

17.8.1 What We Did

This week we focused on improving the user interface to make it more user friendly as well as fixing bugs that prevented models from installing using an updated api class for Hugging Face. The front end of the website was also improved to look more appealing. Also, we presented our progress, versioning, and updated diagrams to the other groups. After the presentation, we showed a working demo for our project that displayed most of the must have requirements for our project working as intended.

17.8.2 What We Will Do

This week we will focus on creating a uninstaller for the models that the user installs to enable the user to easily free up space on their drives. We also need to implement some small UI elements that show download progress and/or errors. Once this is done, we can release our first version of the product.

17.8.3 Action Items

- Add 'delete' functionality to model list
- Improve UI for downloading models
- Fix any remaining functional errors

- Optimize hardware usage
- Create test cases for all elements of the application
- Release first version with installer

17.8.4 Blockers, Issues, Risks

Blockers

Any new errors that are found may delay our progress in releasing a first version of the application. The first release of our application will be an Alpha build and does not represent our final product.

Issues

During testing of our application, we found that different open source models perform worse than expected on users hardware. Therefore, we may need to optimize what hardware resources are used and fine tune the universal settings we have implemented for models.

Risks

There are currently no risks.

17.8.5 Sprint Screenshot

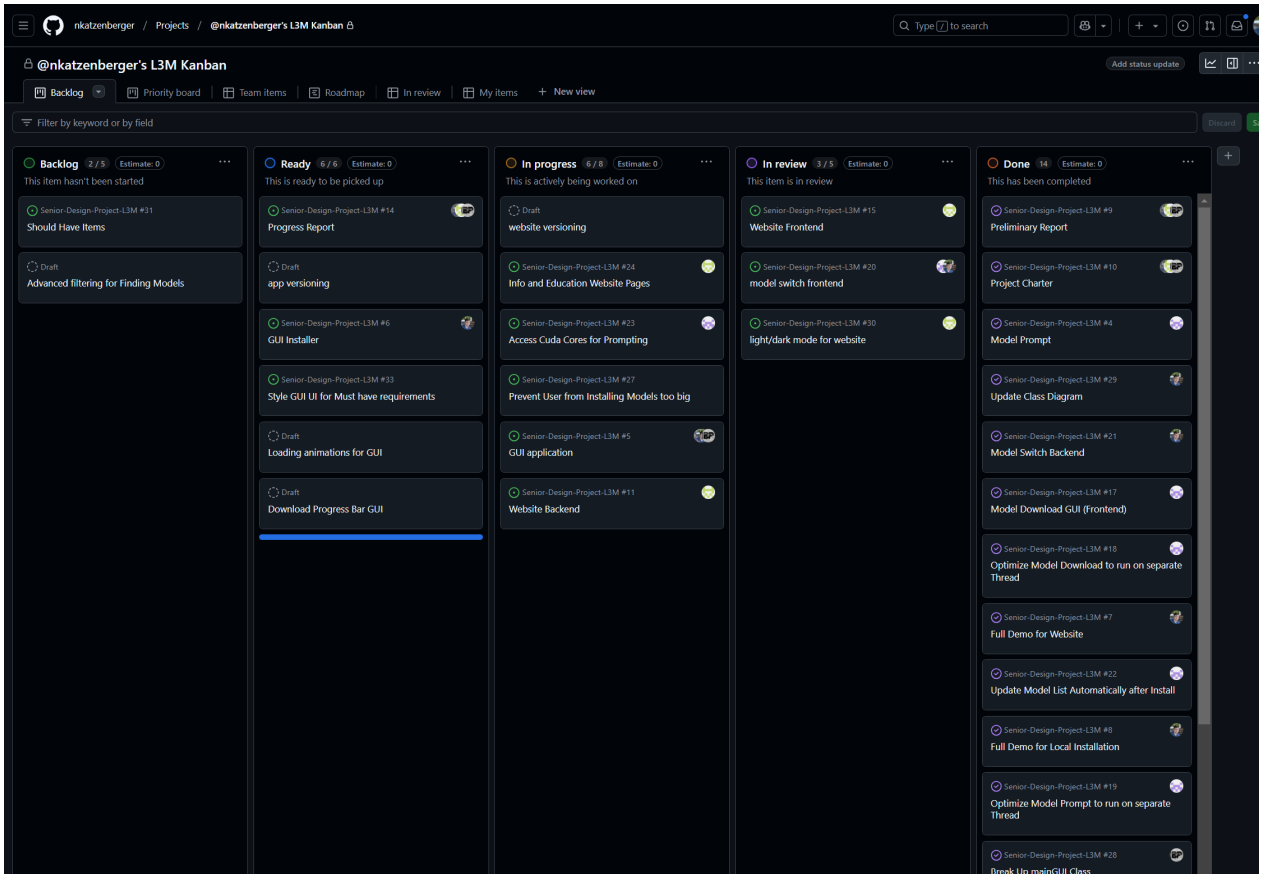


Figure 17.8: Kanban Week 19

17.8.6 UML Diagrams

Updated Class Diagram 12.1
Created Deployment Diagram 14.2

17.9 Week Report 18 (2/21/2025)

17.9.1 What We Did

This week we focused on fixing the errors that came with splitting up the GUI class as well as adding some optimization features to our application. We were able to replicate the original functionality of the application this week and figure out what versioning method we will implement once all of our must-have requirements are finished.

17.9.2 What We Will Do

This week we will focus on getting the front end to look more appealing to the users as the buttons are not as interactive as the team originally intended. In addition, we will present our progress with the other groups in the class next week. Therefore, we need to have a working demo by next Tuesday.

17.9.3 Action Items

- Present to class on progress
- Work on styling of different UI elements
- Add 'delete' functionality to model list
- Polish website front end

17.9.4 Blockers, Issues, Risks

Blockers

There are no blockers currently.

Issues

Model switching functionality still needs to be connected to the front end buttons.

Risks

There are no risks currently.

17.9.5 Sprint Screenshot

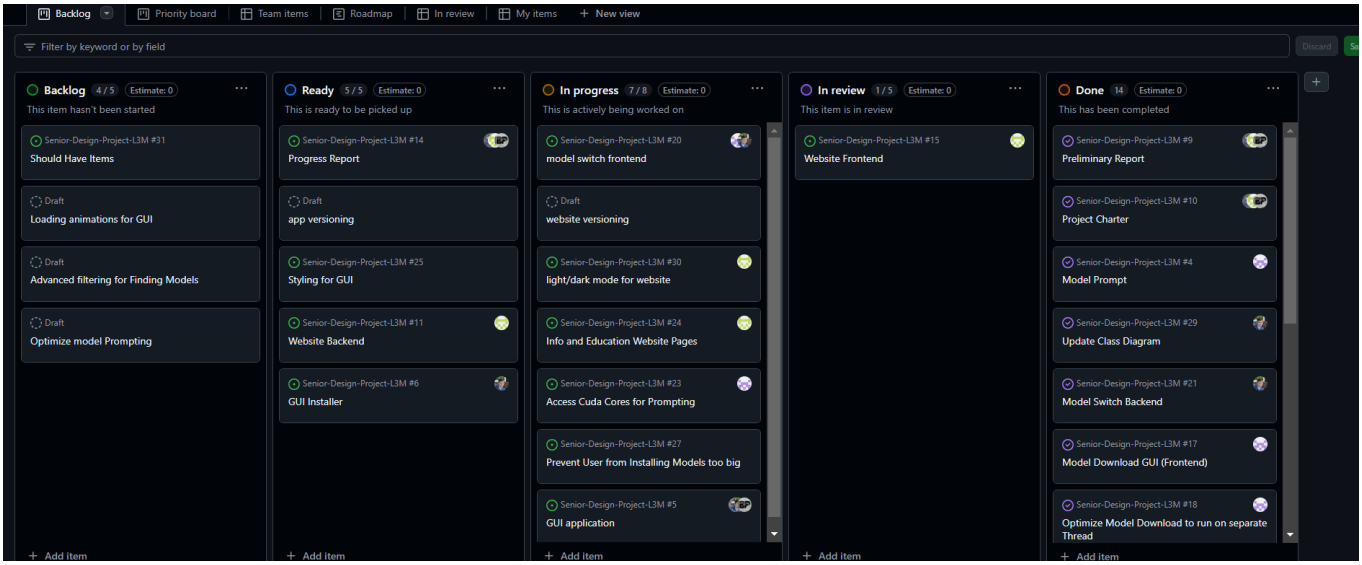


Figure 17.9: Kanban Board Week 18

17.9.6 UML Diagrams

Updated Class Diagram 12.1

17.10 Week Report 17 (2/14/2025)

17.10.1 What We Did

This week we focused on splitting up our GUI class into separate classes to make our system more comprehensive and less volatile in the event that this one class fails. We updated our class diagram below to reflect how our system works currently. In addition, we found a potential method that would prevent users from installing models that are too large for their drives to handle. This method also allows us to access models from Hugging Face without needing an API key. Using the api without the key means that users will not be required to make their own Hugging Face account when using our application.

17.10.2 What We Will Do

This week we will ensure that our new class structure works with our system thus we will test our functions on the system before developing new ones. In addition, we need to start developing the installer for our GUI application to start versioning our software. More front-end polish needs to be applied to our GUI to make it seem user friendly before we can begin working on our ShouldHave requirements

17.10.3 Action Items

- Test functionality of existing features built into the system

- Develop installer for our python application
- Release a version of our software
- develop interactive front end functionality

17.10.4 Blockers, Issues, Risks

Blockers

Need to test our application before we can start working on other features.

Issues

There are currently no issues.

Risks

Our new class diagram may need to be reworked if bug fixes cause major code changes.

17.10.5 Sprint Screenshot

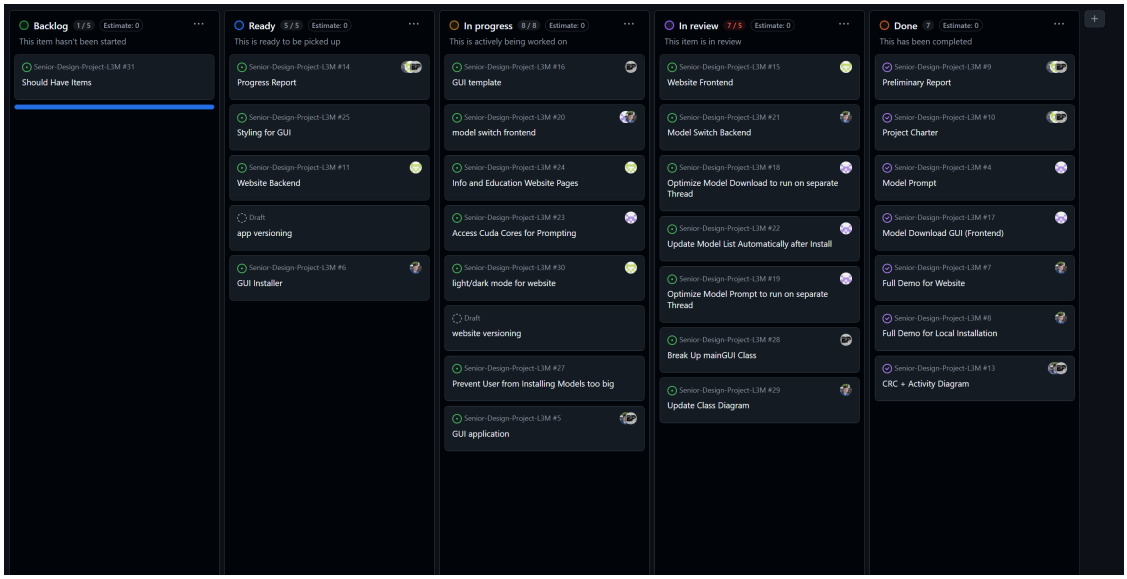


Figure 17.10: Kanban Board as of Week 17

17.10.6 UML Diagrams

Updated Class Diagram 12.1

17.11 Week Report 16 (2/7/2025)

17.11.1 What We Did

This week we implemented the use of thread management to handle behind-the-scenes tasks of prompting the model, downloading the model, and making API calls. This improves the performance of the application and makes it more user-friendly because now the GUI is less likely to freeze when performing these tasks. We also implemented some bug handling for when the user runs out of space while installing a model. Lastly, we added UC_3 (Use Case for main GUI), to our use case cards. Additionally, we updated 13.1 and 13.7 to be more coherent.

17.11.2 What We Will Do

Find a solution to prevent users from attempting to install a model too large for their system. Break up the main GUI class into smaller, more manageable classes. Update the Class Diagram. We did get the model prompting to run on a separate thread, however, it still runs on the CPU. We plan to implement the usage of the system's GPU if the system has one available.

17.11.3 Action Items

- Prevent users from installing models that their system cannot support
- Break up main GUI class
- Update Class diagram
- Enable use of Cuda cores if available for LLM Prompting

17.11.4 Blockers, Issues, Risks

Blockers

We face the same blocker as last week. we did implement some error handling when the user installs a model too big and the download fails, however, we would like to prevent the user from starting the download to begin with.

Issues

We don't face any issues with development or design currently.

Risks

There are no risks currently.

17.11.5 Sprint Screenshot

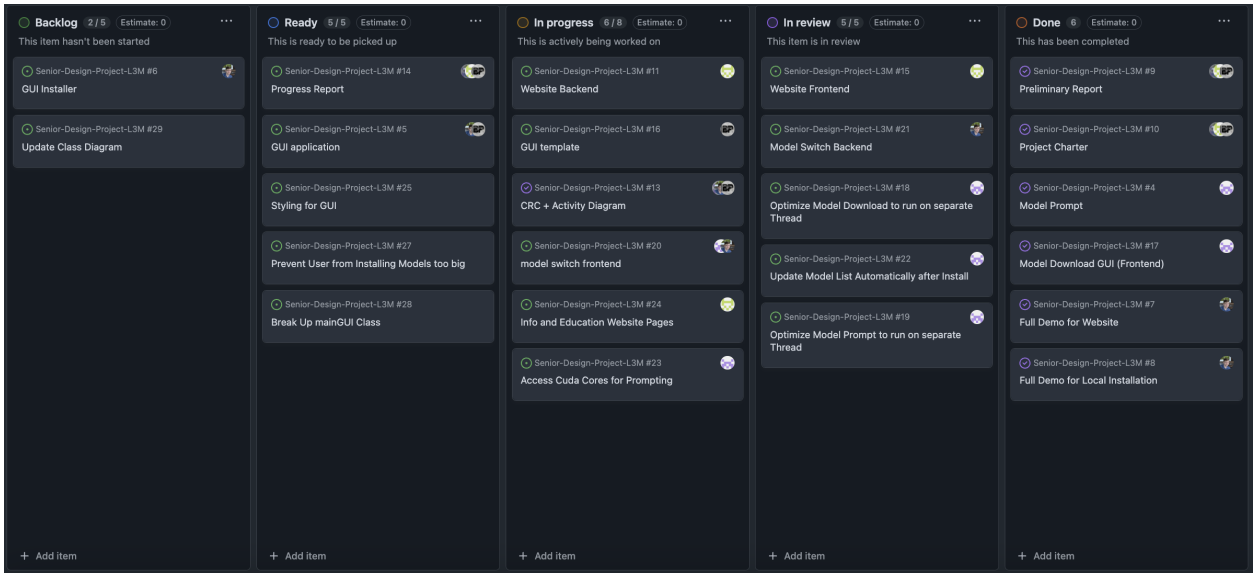


Figure 17.11: Week 16 Sprint Screenshot

17.11.6 UML Diagrams

Updated Activity Diagrams: 13.1 & 13.7

17.12 Week Report 15 (1/31/2025)

17.12.1 What We Did

This week we discussed how the GUI should function concerning the other use cases. We ultimately decided that the GUI needs its own use case to make sense of our development plan. Therefore, we updated this in our Use Case Diagram. We created an activity diagram for the main GUI. We also implemented some of the bigger features. This week we implemented the backend functionality to allow users to switch between selected models. We also built the API to allow users to find new models directly in the GUI and install them. Lastly, we built the front end to display all the models that the user has installed.

17.12.2 What We Will Do

Over the next week, we plan to optimize some of our functions. We plan to use thread management to address some minor issues with the application freezing while making API calls or installing LLM's. We plan to begin shifting some focus on our website as development for the GUI is going very smoothly. Some bugs and errors are handling that we need to address if a user accidentally selects a model too large to install.

17.12.3 Action Items

- Error handling for a user attempting to install a model too large

- Optimization of features including thread management
- Enable use of Cuda cores if available for LLM Prompting
- Add informational/educational pages to our website

17.12.4 Blockers, Issues, Risks

Blockers

We face one blocker, the API call we are currently using to list models for installation does not contain any information on model size or recommended hardware. For this, we will have to find another way to reliably warn or prevent users from attempting to use a model too large.

Issues

The only issue we currently face is that users can crash the app if they try to install a model the system cannot handle

Risks

We currently do not face any risks.

17.12.5 Sprint Screenshot

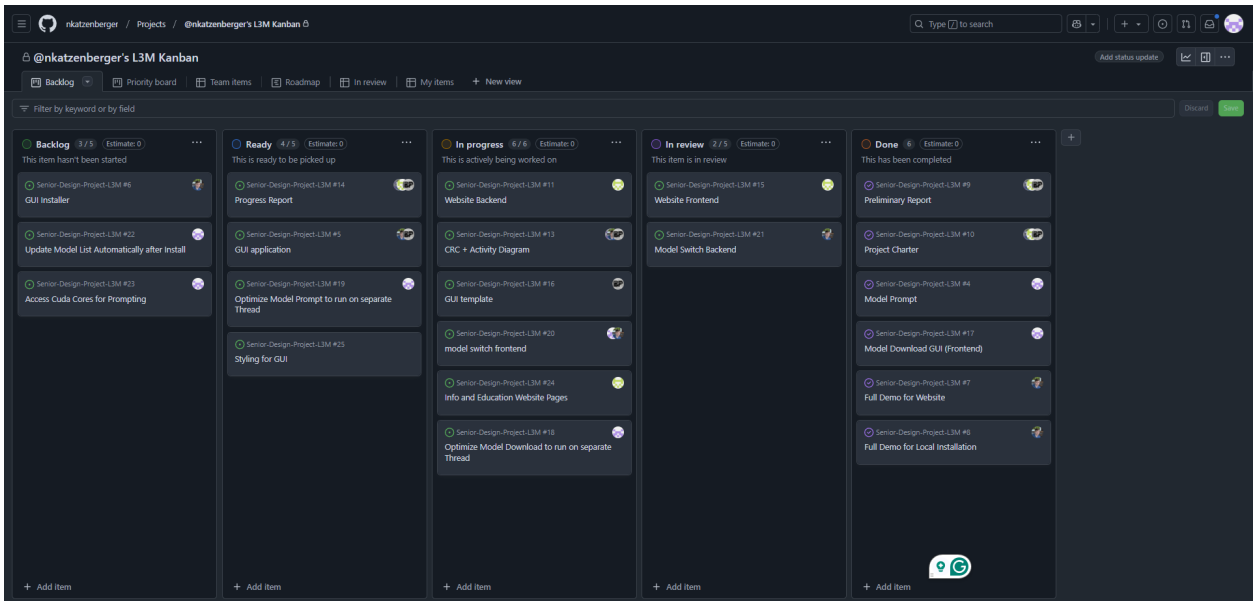


Figure 17.12: Week 15 Sprint Screenshot

17.12.6 UML Diagrams

Updated Diagram 10.1 and Created Diagram 13.7

17.13 Week Report 14 (1/23/2025)

17.13.1 What We Did

This week we refined our Kanban board and assigned each member different features of the project to work on. We mainly split our Kanban board up regarding the different use cases outlined in this document, and made each use case 1 or 2 issues in the board. Under these issues, we detailed each user requirement in the description and features that need to be created that match these user requirements. Each member was assigned at least 2 issues to work on over the next couple of weeks. So far, a rudimentary chat UI has been implemented and integrated with the existing installed-model prompting logic to display the model's response to the user through the UI

17.13.2 What We Will Do

Starting this week and next week, each member of the team will work on the different use cases of the project. Some notable use cases we are currently working on are the UC_2 and UC_5, where the team is working on tuning the LLM's response to be formatted in a PyQT GUI. The GUI will be the main feature of our application that the user interacts with to operate the application. After having a usable GUI, that is feature complete, the team will work on creating an executable file that will install the application to a local computer.

17.13.3 Action Items

- create CRC cards for developing GUI and installers
- Establish format standard for prompting model and length of model output.
- Test that multiple models can be applied to this format
- Build basic framework for GUI
- clean up and add more features to the website
- Research how Wix Toolset can be utilized to install our application
- Start pipeline for creating LLM installer

17.13.4 Blockers, Issues, Risks

Blockers

The team needs to create some executable code for the GUI part of our project before creating an installer for the GUI. Otherwise, the installer would not install anything. We should also review our activity diagrams in the Process View chapter 13 before coding to ensure that we create the features of the system using an optimized flow of operations. This way the application will run with the shortest amount of time and use the least amount of resources possible.

Issues

Some models on Hugging Face, like Llamas, require additional security clearances and steps to use instead of just installing. This requires the team to find a solution that automates these additional steps, if possible.

Risks

The team needs to figure out how to store our Hugging Face API key between the four of us without storing it on GitHub. This poses a security risk to our access to Hugging Face, and may be denied access in the future if we do not handle the key securely.

17.13.5 Sprint Screenshot

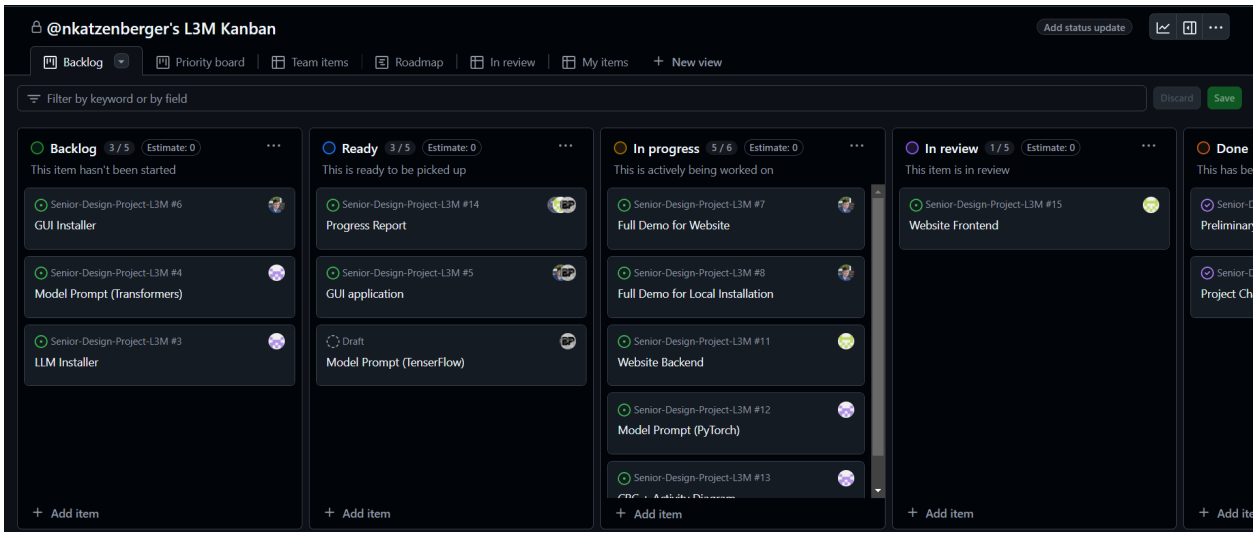


Figure 17.13: Week 14 Sprint Screenshot

17.13.6 UML Diagrams

No new UML Diagrams yet.

17.14 Week Report 13 (12/12/2024)

17.14.1 What We Did

This week we presented a demo of our project's progress for this semester. We were able to build a website for our project as well as download and prompt an open-source AI model using Python. We presented this demo to the class in a 10-minute presentation.

17.14.2 What We Will Do

Over the next few weeks we will stop working on the project for the holiday season. After that, we will begin development on the GUI program for our project. We obtained feedback from our presentation to allow users of our website to have a method to enter their computer specifications to see what models are compatible with their hardware. The goal for this addition is to cater more demanding LLMs to more powerful machines. Another piece of feedback is that the website would be more useful if the descriptions were more elaborate. This would help distinguish different models to users of our website. Some final feedback is that the website should be able to display what models the user already has installed so they don't install the same model twice, and can find similar LLMs.

17.14.3 Action Items

- Build GUI for interacting with installed models
- Allow users to install multiple models on their machine
- Allow users to select which model to prompt within the GUI
- Build the installer for installing GUI from the website.
- Allow users to enter their machine specs into the website and generate recommendations
- Allow the website to view which LLMs the user has installed, assuming they have the GUI application installed.

17.14.4 Blockers, Issues, Risks

Blockers

The main blocker is that the next few weeks will be a winter break for all team members. After the break, we will be working on the project again.

Issues

There are no current issues.

Risks

One risk we face right now is the use of packages. We need to make sure that the installer installs the correct version of Python and the correct packages to run the GUI and the installed models.

17.15 Week Report 12 (12/06/2024)

17.15.1 What We Did

This week we started to prototype our solution. So far we have set up the GitHub repo with CircleCI integration. Moreover, we have laid out the basic folder structure of the three main components of our solution, the web server, the application installer, and the GUI application. Work has started on both the web server and the GUI application, but progress is still early.

17.15.2 What We Will Do

This week we will continue to build our prototype and mainly focus on the website and LLM installer portions of our project. Our goal is to have some basic version of these two features of our project completed to present by next week. This way, we can demonstrate that our basic ideas are feasible for a full implementation by the spring of next year.

17.15.3 Action Items

- continue prototype development
- update diagrams when needed
- submit meeting reports in IDE 401
- submit updated project charter to IDE 401

17.15.4 Blockers, Issues, Risks

Blockers

There are no blockers currently.

Issues

There are no current issues.

Risks

The risk we face is during development our implementation of our prototype may be different than how we diagrammed our UML diagrams. In the event that this happens, we will have to update the diagrams we made to reflect the actual interaction.

17.16 Week Report 11 (11/21/2024)

17.16.1 What We Did

This week we presented the preliminary design, ie diagrams of our system to the professors of the class. We also worked on the prototype of our project.

17.16.2 What We Will Do

Next week we will begin to create a command prompt version of our application. We will test various models to test for this version of our system.

17.16.3 Action Items

- Document research for our project's construction in the document.
- Find LLM to use
- Install specific LLM
- Set up environment in command prompt to prompt the LLM
- Test performance on available systems

17.16.4 Blockers, Issues, Risks

Blockers

Thanksgiving break will likely stop us from working on the majority of the action items this week.

Issues

No issues currently.

Risks

Programming this version of our system may take longer than expected if we are not efficient with how we structure the work flow. In addition, working with large language models on the backend is a relatively new thing to most members of our group. This means we will have to spend extra time learning how to use the right functions in the LLM to get it to take prompts correctly.

17.17 Week Report 10 (11/15/2024)

17.17.1 What We Did

This week we finished up the remaining diagrams that we plan on presenting on next week. These include the package diagram, component diagram, and class diagram. We also worked on and are putting the finishing touches on the slides for the Project Preliminary Design.

17.17.2 What We Will Do

Next week we will present our diagrams and the work done this week. Also, we will continue developing minimal viable prototypes for the system to ensure that the models we have created are feasible.

17.17.3 Action Items

- Present
- make basic components of system

17.17.4 Blockers, Issues, Risks

Blockers

There are no blockers currently.

Issues

There are no issues currently.

Risks

There are no new risks currently.

17.18 Week Report 9 (11/8/2024)

17.18.1 What We Did

Completed Activity diagram in Chapter 13 and package diagram, started class and component diagrams. In addition, we renamed requirement labels in 10 to make them more concise with the requirements table. We also updated the 10.1 to include descriptions of the use cases.

17.18.2 What We Will Do

We will finish the class and component diagrams. Begin working on the presentation slides. Start prototyping and initialize Github repository.

17.18.3 Action Items

- Finish Class Diagram
- Finish Component Diagram
- Presentation Slides
- Github Repository

17.18.4 Blockers, Issues, Risks

Blockers

No blockers currently.

Issues

No issues currently.

Risks

When we begin prototyping, our actual architecture may change from what we had envisioned the project being. These changes can range in size but the final architecture should still incorporate all of our use cases.

17.19 Week Report 8 (10/31/2024)

17.19.1 What We Did

This week we worked on creating diagrams for the use cases of our system. We were able to complete all the activity diagrams, and make user interface templates. We have started other diagrams such as package and component diagrams, but need to finish the final designs of those. In addition, we are re-naming our requirements labels to make them more concise and easier to identify.

17.19.2 What We Will Do

Next week we will focus on finishing the other diagrams and possibly start development on a prototype of the system.

17.19.3 Action Items

- Package Diagram
- Component Diagram
- Class Diagrams
- Other necessary diagrams
- Begin development on project implementation

17.19.4 Blockers, Issues, Risks

Blockers

There are no blockers currently

Issues

One issue may be is that we may not have enough time to completely develop a working prototype for our complex system by the end of the semester. We should have enough time to develop the website front end and application installation.

Risks

A risk we may face during the creation of our diagrams is that they may change the way we develop our project. This is because the way the diagram comes out may be different than what the group envisioned for the project.

17.20 Week Report 7 (10/25/2024)

17.20.1 What We Did

This week we put the finishing touches on the requirements specification, creating use cases, our main use case diagram, and elaborating on the different kinds of requirements. Additionally, we presented on our progress and received feedback which was then used to iterate upon what we had at that point.

17.20.2 What We Will Do

Next week, we plan to create activity diagrams for each use case. Also we will start development on the package diagrams, class diagrams, component diagrams, and outline the user interface.

17.20.3 Action Items

- Create Activity Diagrams
- Create Package Diagrams
- Create Component Diagrams
- Create Class Diagrams
- Create User Interface Mock Up

17.20.4 Blockers, Issues, Risks

Blockers

There are no blockers currently.

Issues

There are no issues currently.

Risks

When creating these diagrams for our system, the diagrams could be more complex than we intend. This may require revisions of the diagrams which takes up more time in planning out our system. In addition, in realizing our diagrams, we may have to rethink how our system will work if the diagrams highlight some complications with our design.

17.21 Week Report 6 (10/17/2024)

17.21.1 What We Did

This week we focused on creating user, system, domain, and other requirements (Chapter 8) and user stories for our project (Chapter 9). We were able to create user requirements that satisfies our intentions for the project. We also created numerous user stories that relate to our requirements, but may add on to this chapter as we identify more user requirements throughout the project.

17.21.2 What We Will Do

This will we will focus on creating use cases for our project, as well as diagramming how these use cases interact with each other when the system is complete. It is important to establish clearly defined use cases for the system since it is the beginning framework of how our software will function. These use cases will also be seen in the requirements chapter. Additionally, we will start work on creating other diagrams to get ahead with the UML diagramming of the project.

17.21.3 Action Items

- Create Use Cases
- Link references to use cases on the requirements chapter
- Create Use Case Diagrams
- Begin development of other UML diagrams

17.21.4 Blockers, Issues, Risks

Blockers

There are no blockers currently.

Issues

There are no issues currently

Risks

When creating use cases, we need to carefully consider the requirements we have laid out for our project. We need to ensure that the use cases we make will include all of the user requirements in our use cases. Otherwise, it would not satisfy our project specifications

17.22 Week Report 5 (10/10/2024)

17.22.1 What We Did

This week we focused on creating the Requirements (Chapter 8) of our project. We met twice this week to discuss the different kinds of requirements that our project should fulfil. We have decided on a set of System Requirements 8.4, Non-functional Requirements 8.5, and Domain Requirements 8.6. Additionally we discovered an application, LM Studio, that functions similarly to what we have envisioned, we discussed whether we should course-correct to account for the existence of this application. We ultimately decided that the best course of action was to see what we could learn from LM Studio and continue the project on the prior trajectory.

17.22.2 What We Will Do

Next week we will focus primarily on diagramming the requirements that we made this week. Additionally we will further explore the implementation that the team behind LM Studio came up with and seeing what strategies and requirements we can derive from it.

17.22.3 Action Items

- Complete list of requirements
- create user stories
- create use cases
- create use case diagrams

17.22.4 Blockers, Issues, Risks

Blockers

There are no blockers currently.

Issues

LM Studio discovery, this hindered progress as we had to think about how to proceed and whether or not this product invalidated our proposed implementation. We ultimately decided that it would be better if we used their application as a basis for developing user requirements and stakeholders for the rest of the project. The app showed the group that our project is possible and can function efficiently.

Risks

Potential patents that LM studio may hold could prevent us from developing a similar solution for profit. In this case, we would shift the project to an open source, not-for-profit model as this would still be in line with our mission statement of democratizing LLMs

17.23 Week Report 4 (10/03/2024)

17.23.1 What We Did

This week we completed the first iteration of the development plan and presented our progress to Professor Muresan and the class. We also discussed amongst ourselves about the overall direction of the project to ensure that each team member was on the same page about intended features and the loose road-map. Topics like which project components lie within the scope of the time and funding that we currently possess were refined and certain use-cases / components were decided to be mocked in our minimal viable product.

17.23.2 What We Will Do

Next week we will put special focus on narrowing down the feature set of the minimal viable product. We will do this through requirements analysis and use cases.

Diagrams and other artifacts that are generated from these activities will be added to this file for documentation purposes.

Next week we also plan to conduct some interviews with potential users of the application to help with determining user requirements.

17.23.3 Action Items

- Begin logging user requirements for our system
- Create use cases that include these requirements
- Create use case diagrams
- Begin interviews with potential users
- Continue research on available resources and open source LLMs

17.23.4 Blockers, Issues, Risks

Blockers

There are no blockers currently

Issues

During the presentation, we were made aware of a potential alternative solution to the problem we are trying to solve. NVIDIA's Chat RTX is a system that claims to allow users to customize GPT models, but it only works on NVIDIA graphics cards. When asked about their experience with building a new LLM on this service, users said it was difficult and time consuming. Thus, our advantage will be that our installer will work on any graphics cards, will provide a more diverse range of models to train from, and be easy to use.

Risks

Some risks with our progress this week may involve problems with diagramming our solution, and it disagreeing with our original idea for the product. This risk is not too much of an issue since it allows for the group to understand how different use cases will realistically interact with each other. The team can always adapt to changes in how the system will function. Additionally, the user requirements we collect may be different than what the team originally anticipated with the project. This may alter the development process and even the features available in the final product. However, we can always add extra features outside of the user requirements if there is time allotted to this.

17.24 Week Report 3 (09/26/2024)

17.24.1 What We Did

This week we primarily worked on the Development Plan (Chapter 7) for our project. Along with this, we created some slides to summarize everything in our development plan. We also did some further research on how hosting and developing our project will plan out. We found that Python Flask will likely work best for our project. Through creating the Dev plan will likely need to store datasets to train the LLMs on a server we use to host the other downloadable models.

17.24.2 What We Will Do

Next week we will focus primarily on delivering slides to present on our development plan. Additionally, we will polish the development plan before moving to the next step in the planning phase. We will also begin creating UML diagrams for the structure and flow of our project after developing use case diagrams.

17.24.3 Action Items

- Continue working on slides presentation for development plan
- Present slides to the other groups
- Begin work on the Development View (Chapter not added yet) and create UML diagrams
- Create Use Cases
- Add more terms to the glossary

17.24.4 Blockers, Issues, Risks

Blockers

There are no blockers currently

Issues

Training a model will not be able to be done from the ground up on a local machine due to storage limitations, this feature is now considered outside of our scope.

Risks

Same risks as last week apply. As we do research we may find that our original plan is not feasible and the project's conception will have to be restructured. The main things we realized is that the training of the models will likely have to be a feature not attainable on local machines due to storage capacity issues. LLMs (especially Gen AIs) require large datasets consisting of multiple terabytes of data to be trained effectively.

17.25 Week Report 2 (09/20/2024)

17.25.1 What We Did

- Carl joined the team
- Created Kanban board for our project
- Initialized Github project and repository for code
- Began research on project costs, feasibility, and server hosting platforms

17.25.2 What We Will Do

This week we will be focusing on research on the logistics for the project and documenting the different methods we will use to create this project. In the past week, we found that Hugging-face, Pytorch, and TensorFlow are viable libraries for open-source AI and LLMs. In addition, we plan to make UML diagrams for the development of the website and installer we plan to create for the project.

17.25.3 Action Items

- Create Class, Use Case, Object, Sequence, Deployment, and Timing diagrams for our installer and website
- Research best servers for hosting websites particularly ones that enable downloads of large file sizes
- Fill in Development Plan (Chapter 7) with relevant information
- Create tasks on the Kanban board for the week.
- Create branches on GitHub to begin the development process
- Create Mockups for the project development, IE starting code, uploading format, etc.

17.25.4 Blockers, Issues, Risks

Blockers

Software development cannot start until we decide the best course of action with hosting our website and installer so we don't waste time in development. Also, we need to ensure that we have a solid understanding of the server we want to host our project on to create an accurate Deployment diagram. This applies to the structure of the rest of development such as the installer and the libraries we want to access.

Issues

Getting Carl up to speed is the only issue we are experiencing at this point.

Risks

'Potential Risks we can encounter are our original plan failing and inability to implement nor construct a phase one. Another Risk is the models being too big in size making it hard to work around and run on laptops.

17.26 Week Report 1 (09/12/2024)

17.26.1 What We Did

The first thing we did was create the project idea and create use cases that apply to our project idea. Plus, we created the Milestone 1 report that affirmed our team name, mission statement, key drivers, a description of our project, and some key constraints. Also added a bibliography.

17.26.2 What We Will Do

Refine our project idea and layout preliminary resources to begin development on the software on our project. We want to plan out how the classes and use cases of the program interact carefully since AI development can get complicated quickly. Creating UML diagrams for our project is essential for documentation and explanation of how our project is intended to work.

17.26.3 Action Items

- Create a Kanban Board to define steps to developing the project
- Create a GitHub repository for the project
- Begin development on URL diagrams
- Research best methods for using Open AI sources and program installers [2]

17.26.4 Blockers, Issues, Risks

Blockers

Right now we aren't experiencing any blockers, since we are primarily in the planning phase of the project.

Issues

An issue with our project idea may be that the development of our project may be very time-consuming if we are not careful with how we approach development. Therefore, we are going to create diverse plans for how we approach the project.

An issue we face is that we will only be able to pre-trained AI models. Training AI is very resource-intensive and requires vast hardware resources and energy. In addition, we may be limited to the amount of data that we have access to train the model with, since using data from the internet may cause legal issues.

Risks

Installing software from the internet can be risky, opening up the potential to install malware on your personal computer. This is especially the case when looking at code-sharing websites like GitHub. The team will have to evaluate the types of downloadable content on the AI selection store we wish to create with this project. Even after the completion of our project, adding new open-source LLMs would have to be reviewed by the team before considering adding them to our installer.

Another factor to consider is that our AI models may not be 100% accurate and may produce false

information when asked certain questions. This is reliant on the quality and type of data the user trains the AI on. Therefore the team must warn users of this potential misinformation.

Glossary

Activity Diagram A type of UML diagram that visually represents the flow of activities within a business process or system. xvii, 54, 55, 56, 57, 58, 59, 60

AI the theory and development of computer systems able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.. 1, 2, 10, 16, 21, 34, 35, 36

AWS Amazon Web Services. A collection of tools and hosting services aimed at reducing upfront costs like hardware in web development by providing virtualized environments and serverless code execution.. 19, 78

Class Diagram A type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. xvii, 47, 48, 49, 50, 51, 52

Cloud Computer system resources and compute capacity that are provided over the internet-of-things to an end user. Applications hosted on the 'Cloud' are really just hosted on a server in a secure facility owned by the hosting company.. 22

code The set of instructions, or source code, written in a programming language that a computer understands and executes to perform specific tasks. 12, 13

Component A software component is a unit of composition that has well-defined interfaces and context dependencies. It can be deployed independently and composed by third parties. 63

Component Diagram Visually represents the structural relationships between components within a system. xvii, 66

CouldHave This defines the third highest priority requirement.The system could implement all of the tasks, requirements, or anything that is marked this way, but if resources are limited, it can be left out of the current and next version. Build in two versions from now. 23

Data In the context of this document, personal data. All of the pieces of information that a service collects from its users through use of said service. This information is often sold to third parties or used to train artificial intelligence without the user's direct knowledge or notification.. 22

Github a web-based platform that allows developers to store, share, and collaborate on code. It's built on Git, an open-source version control system that tracks changes to files and allows multiple people to work on the same files simultaneously.. 12, 13, 17, 21, 30, 80

GUI short for graphical user interface. A visual way of interacting with a computer using items such as windows, icons, and menus, used by most modern operating systems. 10, 22, 23, 24, 34, 36, 67, 85, 87, 92

Hugging Face A platform where developers upload their machine learning models and datasets to share with the broader developer community.. 19, 21, 27, 29, 30, 36, 67, 83, 87

IEEE Acronym of Institute of Electrical and Electronics Engineers; A widely accepted format for writing research papers, commonly used in technical fields, particularly in computer science. . 12, 19

Linux Linux is an open-source operating system (OS) for computers, servers, and other devices that is based on the Unix operating system.. 11, 12, 19, 37

LLM LLM stands for large language model, which is a type of artificial intelligence (AI) program that can process and generate human-like text. LLMs are a subset of machine learning called deep learning, which uses algorithms to analyze large data sets to recognize patterns. 1, 2, 10, 11, 12, 20, 22, 23, 27, 33, 38, 67, 78, 94, 95, 101

Local Hosting Local hosting, or local networking is the act of providing application services on a single machine or a network of connected systems that are not connected to the broader internet.. 22

Mac macOS is the operating system (OS) that powers Apple's Mac computers and laptops.. 11, 12, 37

MustHave This defines the first highest priority requirement. All of the tasks, requirements, or anything that is marked this way are build in the current version. 22, 23, 24, 25, 26, 74

OpenAI OpenAI is an American artificial intelligence research organization founded in December 2015 and headquartered in San Francisco, California.. 10

Regression Testing A type of software testing performed after making changes (like bug fixes or new features) to ensure that the changes haven't introduced any new issues or broken existing functionality. 17

Sequence Diagram A type of interaction diagram that visually illustrates the interactions between objects or participants in a system over time. xvii, 61, 62

ShouldHave This defines the second highest priority requirement. The system should implement all of the tasks, requirements, or anything that is marked this way, but if resources are limited, it can be left out of the current version. Build in next version. 22, 24, 25, 74, 79, 87

UML Acronym of Unified Modeling Language. The unified modeling language is a general-purpose visual modeling language that is intended to provide a standard way to visualize the design of a system. 12, 15, 100

Windows Windows is a computer operating system (OS) developed by Microsoft that manages a computer's resources and allows users to interact with their device. 11, 12, 18, 19, 37

WouldHave This defines the lowest priority requirement. The system would like to implement all of the tasks, requirements, or anything that is marked this way, but only if resources are available. It can be left out of all future versions. 23, 24, 26, 74

Bibliography

[1] (2006) Amazon web services. [Online]. Available: https://aws.amazon.com/

[2] (2015) Open ai. [Online]. Available: https://openai.com/

[3] (2024) Application installer. [Online]. Available: https://drive.google.com/file/d/1va6nJQVo4ADoUVhpF44dJbKJNCav9FNY/view?usp=sharing

[4] (2008) Github. [Online]. Available: https://github.com/

[5] (2011) Circle ci. [Online]. Available: https://circleci.com/

[6] (2024) Github repository. [Online]. Available: https://github.com/nkatzenberger/Senior-Design-Project-L3M

[7] (2016) Hugging face. [Online]. Available: https://huggingface.co/

Index