# Text-based Artificially Intelligent Gaming Adventure

## First Mentor Evaluation

**Submitted by:**

**(101916052) Harsh Thakur**

**(101916017) Shrey Bandlish**

**(101916030) Navneet Kaur**

**(101916069) Harshit Rai**

**BE Third Year- CSE**

**CPG No. 66**

Under the Mentorship of

Dr. Shivendra Shivani Assistant Professor

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

**Computer Science and Engineering Department**

**Thapar Institute of Engineering and Technology, Patiala**

**March 2022**

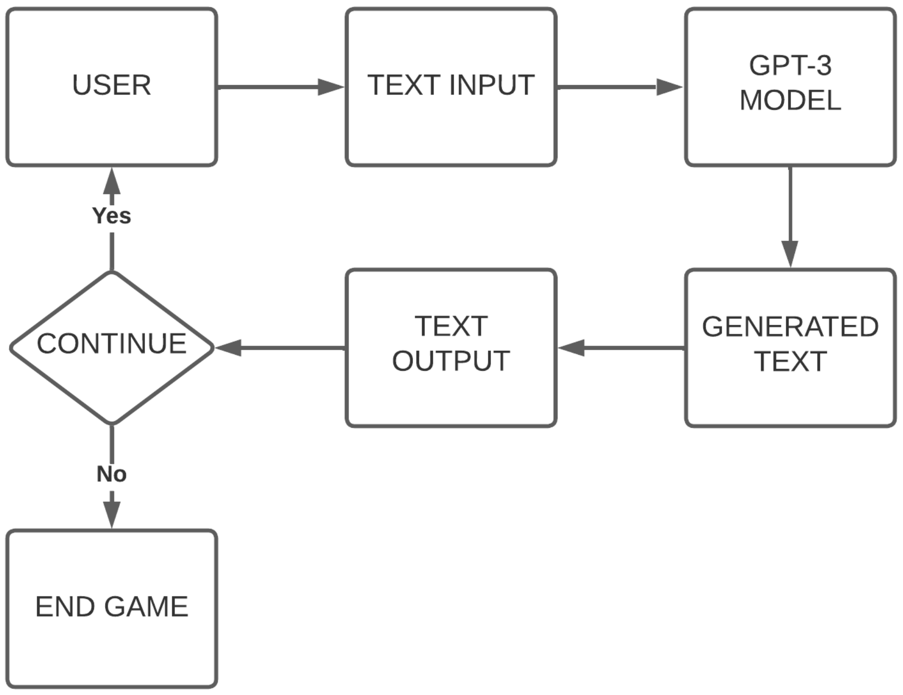
## TABLE OF CONTENTS

## Project Overview

TAIGA is a solution to the problem of linearity most stories/games suffer through, having an abundance of repetitive parts, leaving users with little to no motivation for replayability. Being an artificially intelligent text based game gives TAIGA a tree-like structure, where users' choices act as branches concluding to a different leaf every time one wishes to replay and thus proposes a much needed new perspective for story driven games. A random seed is generated, restricted only by the genre of the story the user chooses to play, giving roots to this tree structure and building branches upon it every time a new choice comes forth.

Users’ text input will be taken in the form of sentences and converted to appropriate actions using natural language processing, continuing the story by generating text using a specifically trained model and then asking for further user input, building a dynamic data exchange loop. This will create a Text-based Artificially Intelligent GAming Adventure aka TAIGA, where the only boundary will be one’s own imagination.

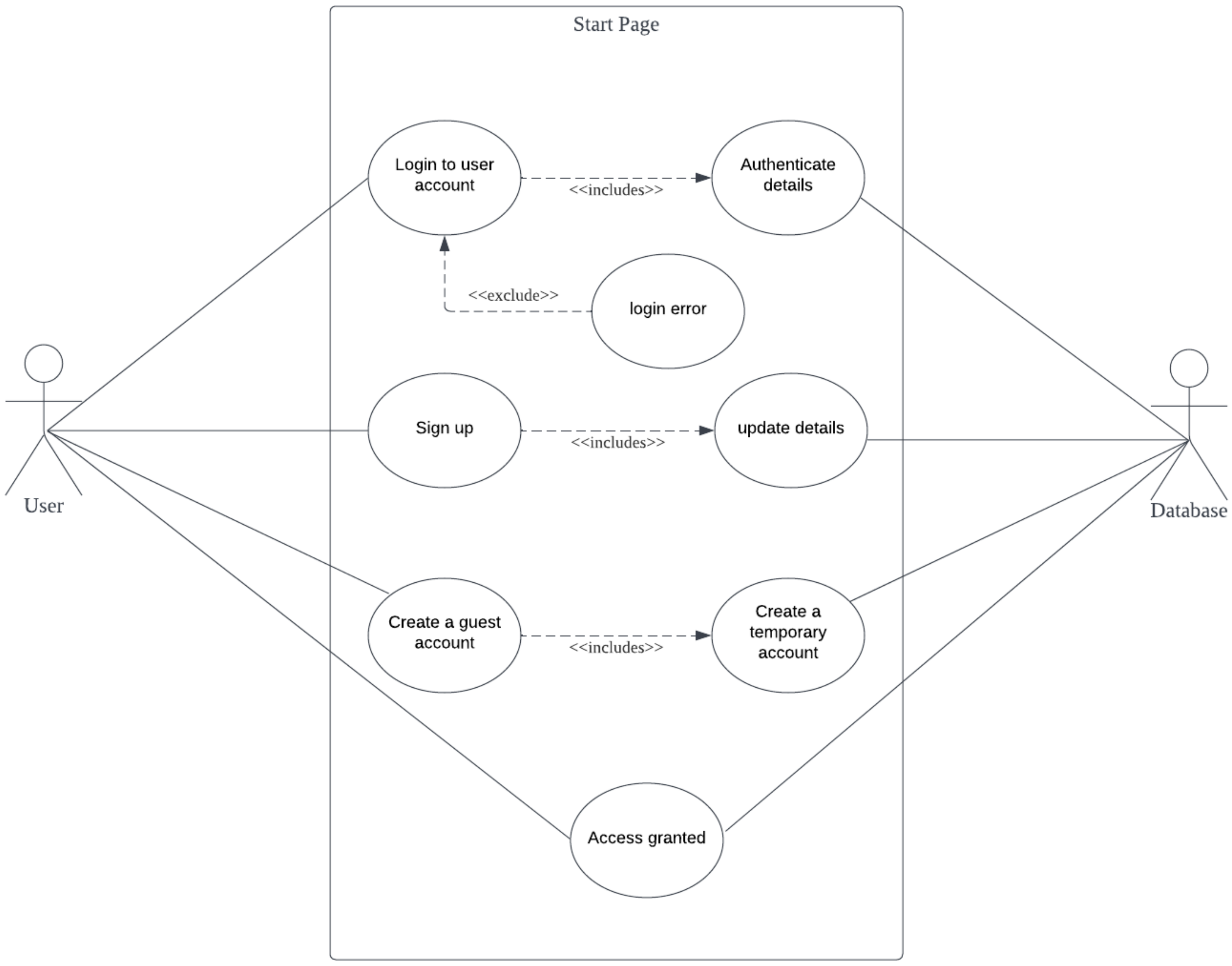
Input can also be given in the form of speech, which our application will process using models already available for speech to text conversion, this will help TAIGA cater to a bigger audience, focusing more on user interactivity. The text provided by the game can also be converted to a vocal format enabling appropriate options, which will give TAIGA the right of narration and make the whole experience theatrical, increasing accessibility and providing a hands free experience.



*BLOCK DIAGRAM*

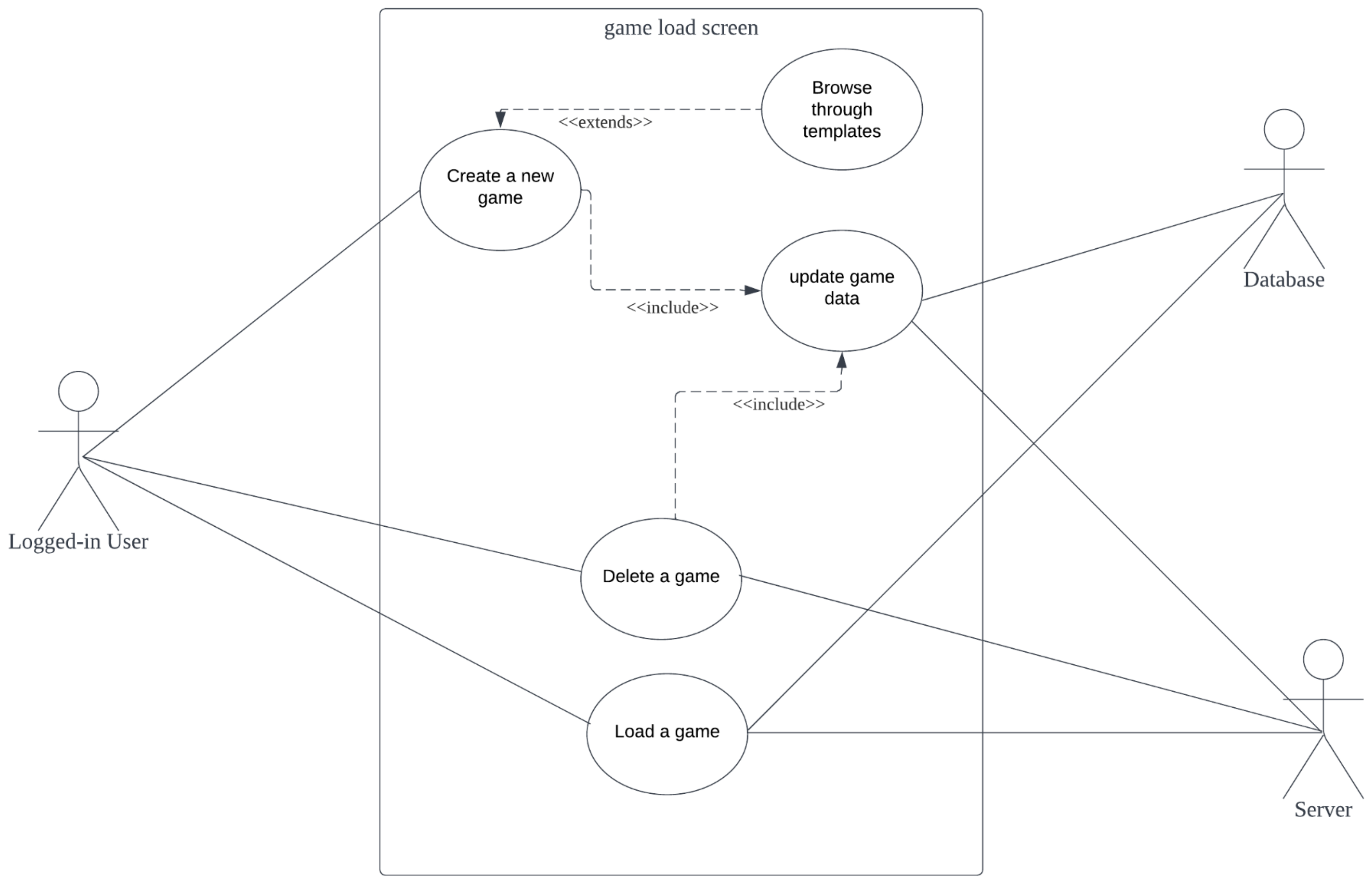
## Use Case Diagrams

| | |
|---|---|
| Use case ID | UC_01 |
| Use case name | Start page |
| Description | The Login page is a portal module that enables users to log in by typing their user name and password. |
| Basic Flow | This use case starts when a user wants to log into the System.<br>Below are the steps followed:<br>1) The system requests the user to enter their login credentials (the details of which are present in the database)<br>2) The user is logged into the system after the system validates their login credentials. |
| Alternate Flow | 1) If the login credentials are not found in the database then a login error is displayed. |
| Primary Actor | User |
| Secondary Actor | Database |
| Pre-Conditions | 1) New user needs to sign up with the website and create the account.<br>2) Otherwise, the user needs to have a valid account |
| Post-Conditions | 1) The actor is now logged into the system if the use case was successful - login page is displayed.<br>2) Otherwise, the system state remains intact and if the account is not valid a login error is displayed. |


Start Page
Login to user account
<<includes>>
Authenticate details
<<exclude>>
login error
Sign up
<<includes>>
update details
Create a guest account
<<includes>>
Create a temporary account
Access granted
User
Database

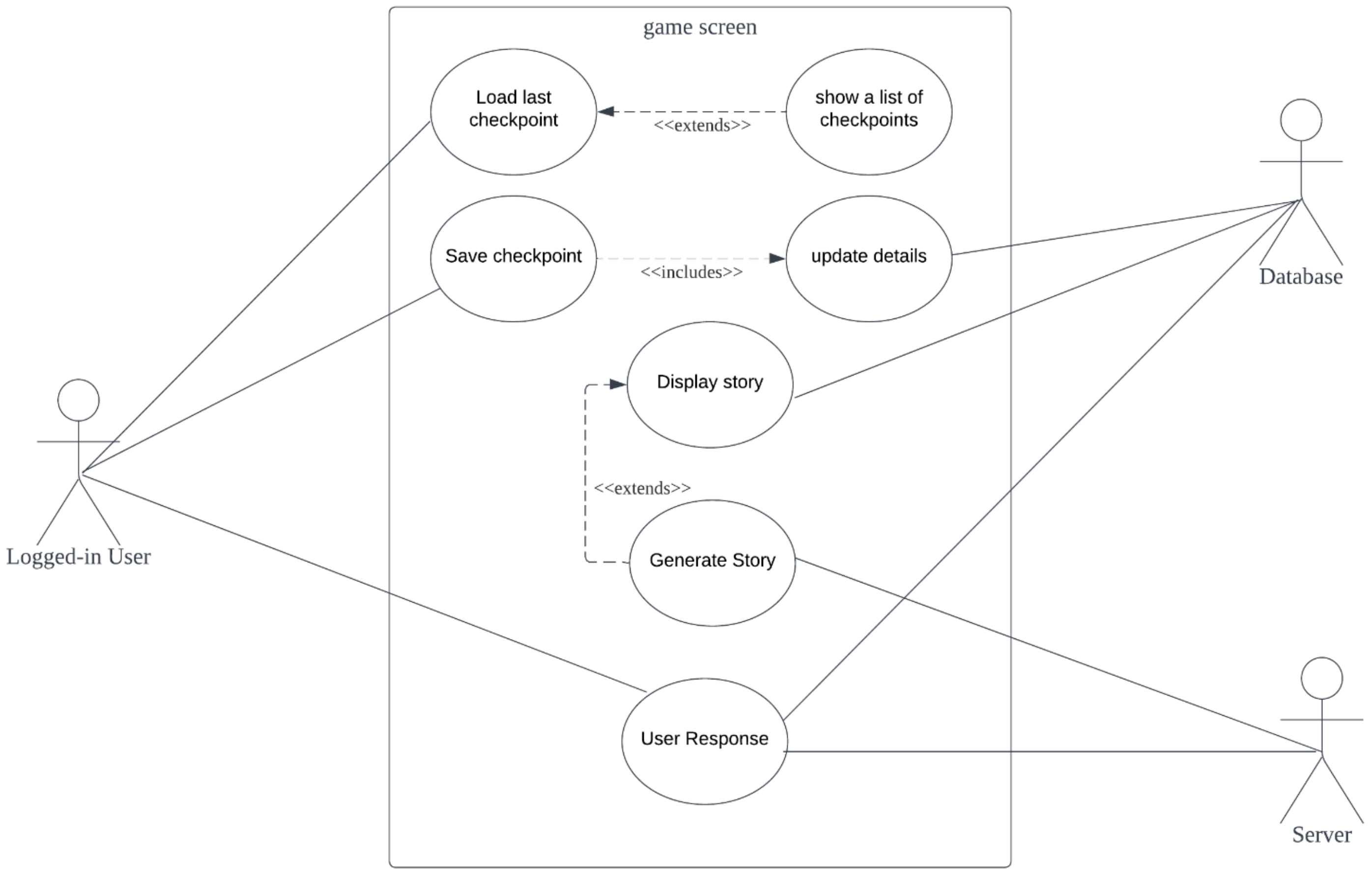
*UC_01*

| | |
|---|---|
| Use case ID | UC_02 |
| Use case name | Game Load Screen |
| Description | The game load screen is a module that enables the user to select the game file that they want to resume playing or delete the game file that they don't want to be saved in the system. |
| Basic Flow | This use case starts when the user has logged-in to the system.<br><br>Below are the steps followed:<br><br>1) The system shows the user their available game files.<br><br>2) The user selects and loads the game file that they want to resume. |
| Alternate Flow | 1) The user can start a new game from the given templates.<br>2) The user can delete the selected game file if they don't wish to continue the selected game file. |
| Primary Actor | Logged-in User |
| Secondary Actor | Database, Server |
| Pre-Conditions | 1) The user needs to be already signed in to the system.<br><br>2) The user should have a file game that he has saved on the system.<br><br>3) Otherwise the user needs to start a new game file from the given templates. |
| Post-Conditions | 1) The user can now access the game file and resume the game from the last checkpoint.<br><br>2) Otherwise, the system state remains intact and the user can start a new game file from the given templates. |


game load screen
Browse through templates
<<extends>>
Create a new game
update game data
<<include>>
<<include>>
Delete a game
Load a game
Logged-in User
Database
Server

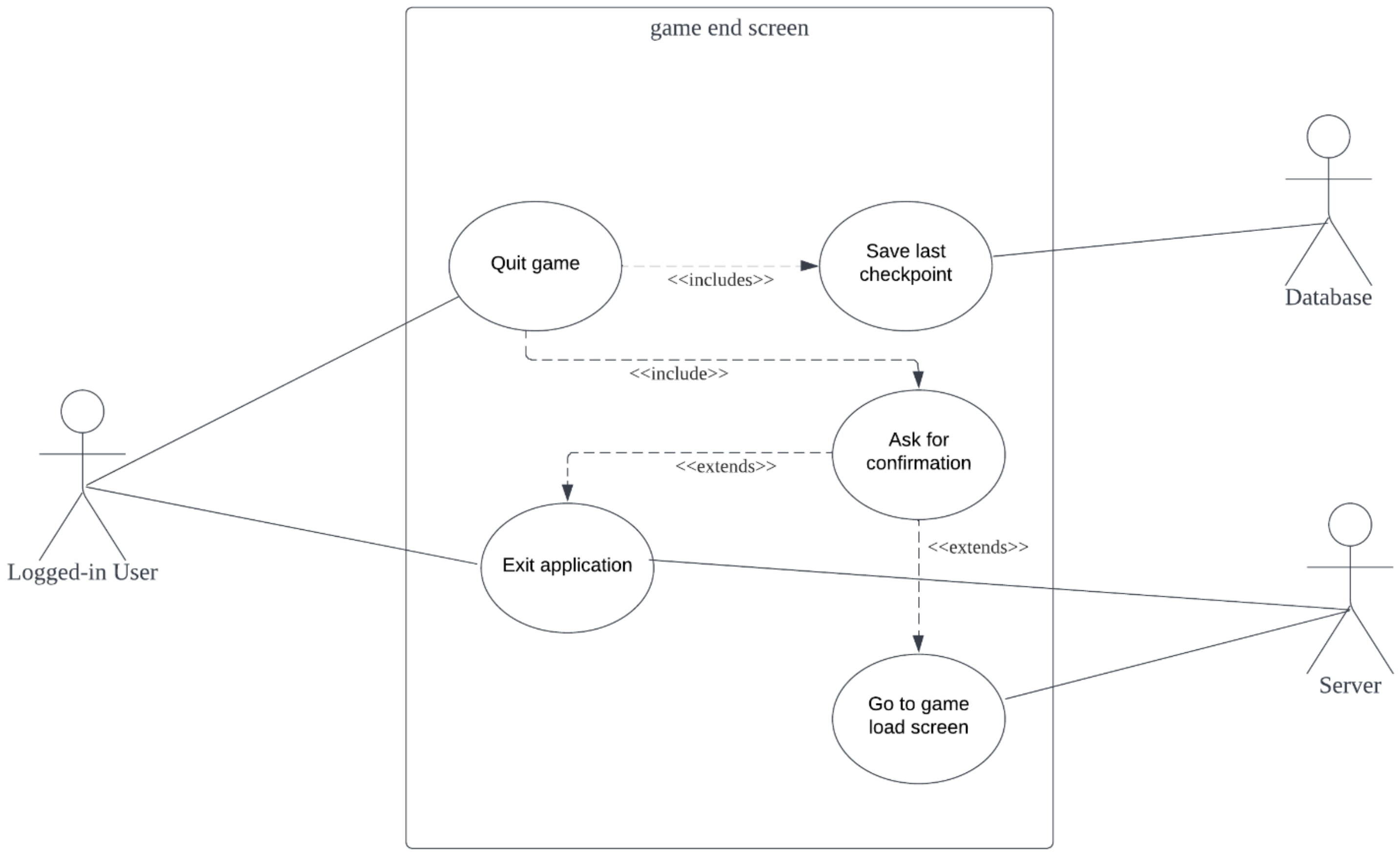
*UC_02*

| | |
|---|---|
| Use case ID | UC_03 |
| Use case name | Game screen |
| Description | The Game Screen is a module that enables the user to play the game file that they previously selected. |
| Basic Flow | This use case starts when the user has selected the game file that they want to play. Below are the steps followed:<br><br>1) The system displays the story from the last saved checkpoint and displays the story generated by the server.<br><br>2) The user responds to the generated story and the server generates the story with respect to the response given by the user.<br><br>3) The user can create a checkpoint if they want to. |
| Alternate Flow | 1) The user loads a previously saved checkpoint from the 5 last saved checkpoints. |
| Primary Actor | Logged-in User |
| Secondary Actor | Database, Server |
| Pre-Conditions | 1) The user needs to have a game file saved. |
| Post-Conditions | 1) The user can now play an interactive, procedurally generated game. |


game screen
Load last checkpoint
<<extends>>
show a list of checkpoints
Save checkpoint
<<includes>>
update details
Display story
<<extends>>
Generate Story
User Response
Logged-in User
Database
Server

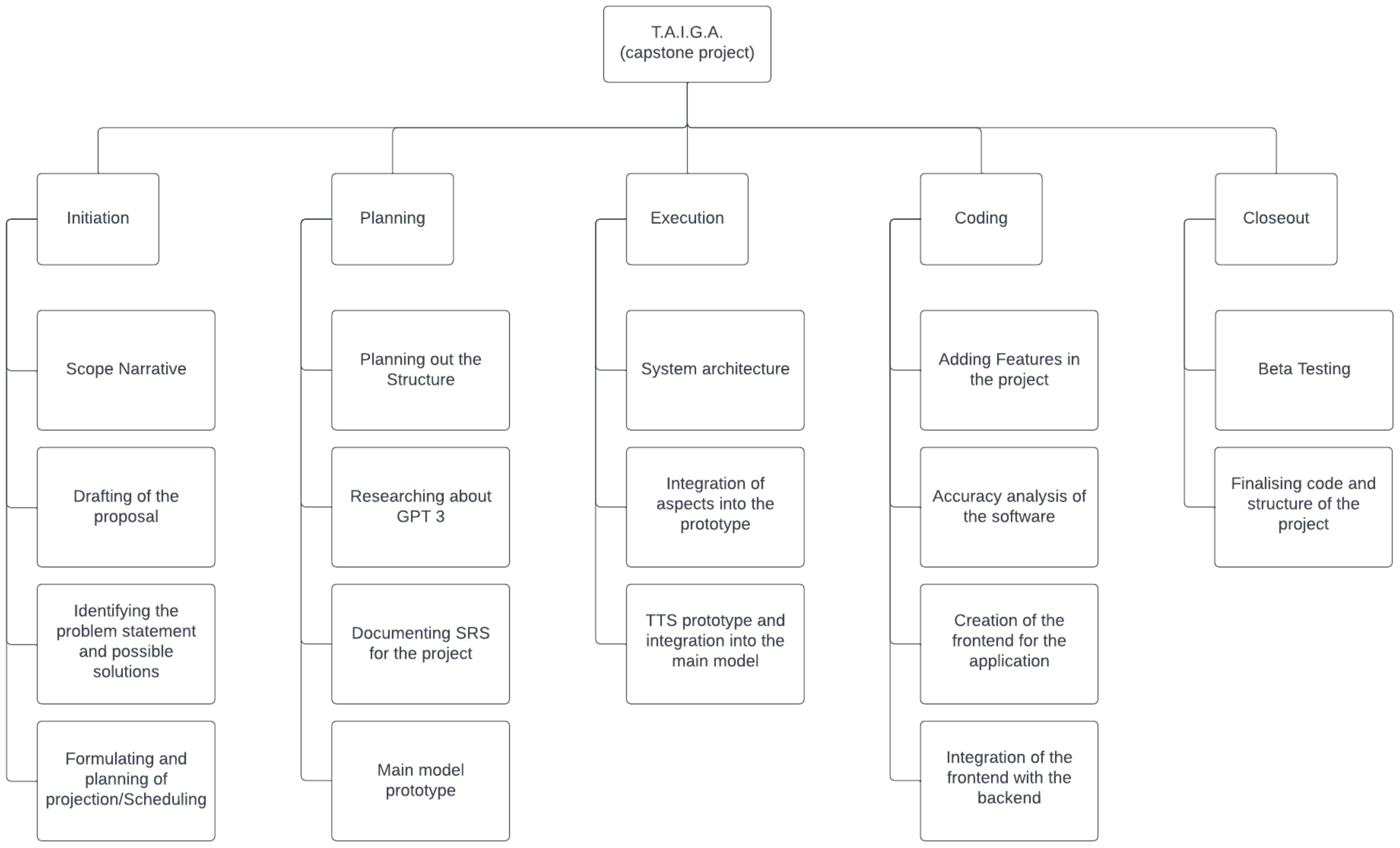
*UC_03*

| | |
|---|---|
| Use case ID | UC_04 |
| Use case name | Game End Screen |
| Description | The Game End Screen is a module that enables the user to exit the game file or the entire application |
| Basic Flow | This use case starts when the user wants to exit the application. Below are the steps followed:<br><br>1) The user clicks on the exit button that automatically saves the game file.<br><br>2) The system asks the user for confirmation.<br><br>3) The user confirms and exits the application. |
| Alternate Flow | 1) The user can cancel the action on the confirmation page and return back to the game screen.<br><br>2) The user can choose to return to the game load screen and can exit only the game file instead of the entire application. |
| Primary Actor | Logged-in User |
| Secondary Actor | Database, Server |
| Pre-Conditions | 1) The user needs to click on the exit button. |
| Post-Conditions | 1) The user has exited the entire application.<br><br>2) The user has exited the game file and returned to the game load screen. |


game end screen
Quit game
<<includes>>
Save last checkpoint
Database
<<include>>
Ask for confirmation
<<extends>>
Exit application
<<extends>>
Logged-in User
Server
Go to game load screen

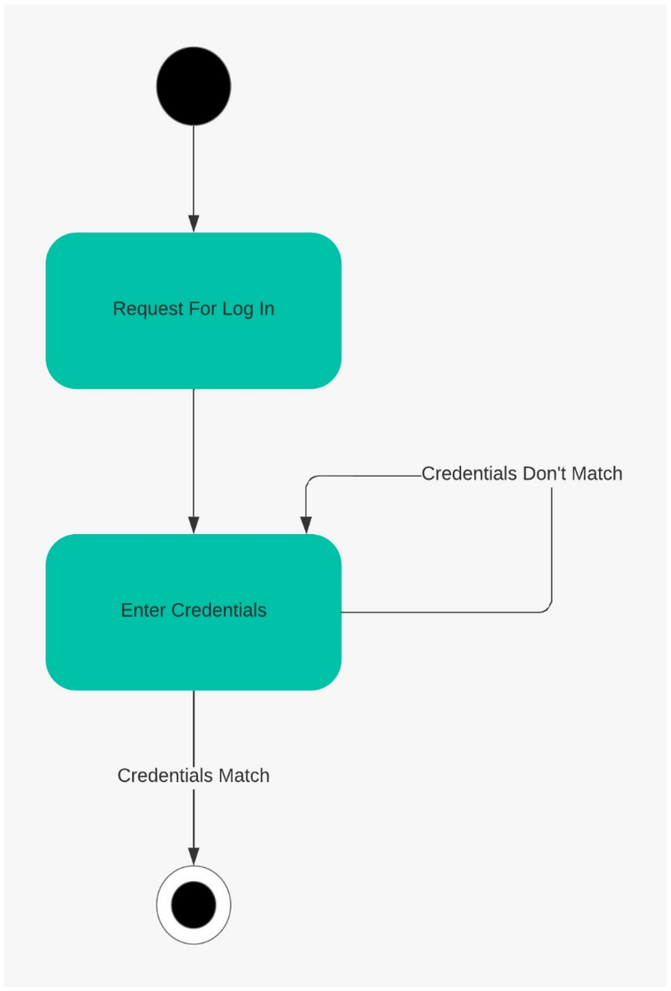
*UC_03*

## Work Breakdown Structure

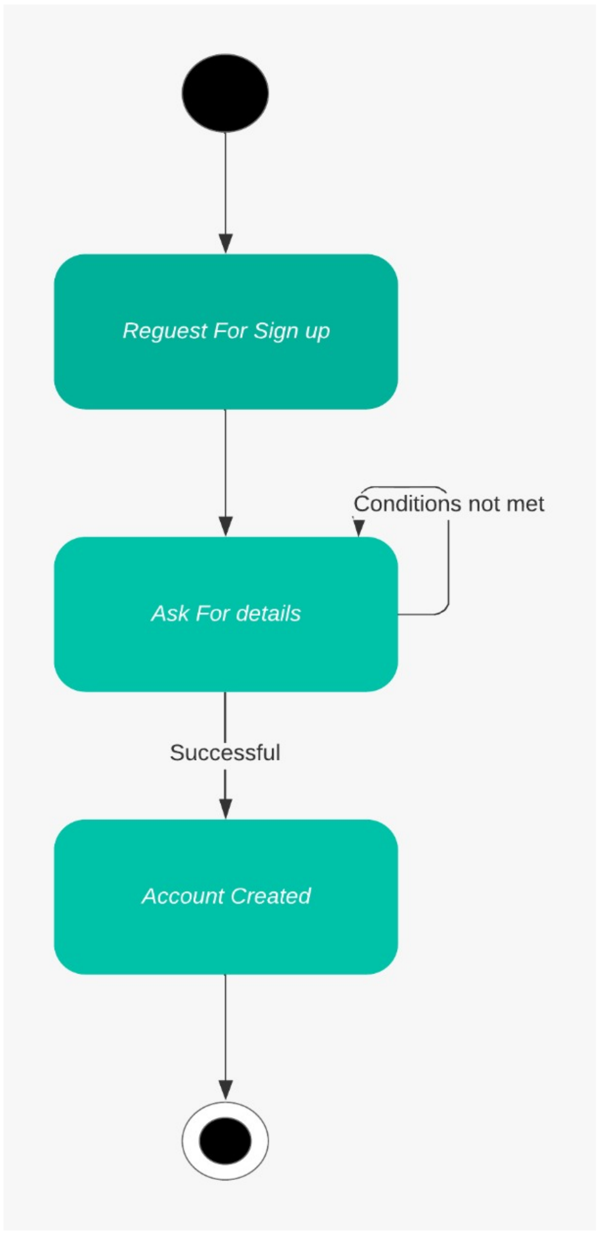

T.A.I.G.A. (capstone project)
Initiation
Scope Narrative
Drafting of the proposal
Identifying the problem statement and possible solutions
Formulating and planning of projection/Scheduling
Planning
Planning out the Structure
Researching about GPT 3
Documenting SRS for the project
Main model prototype
Execution
System architecture
Integration of aspects into the prototype
TTS prototype and integration into the main model
Coding
Adding Features in the project
Accuracy analysis of the software
Creation of the frontend for the application
Integration of the frontend with the backend
Closeout
Beta Testing
Finalising code and structure of the project

## Activity Diagrams

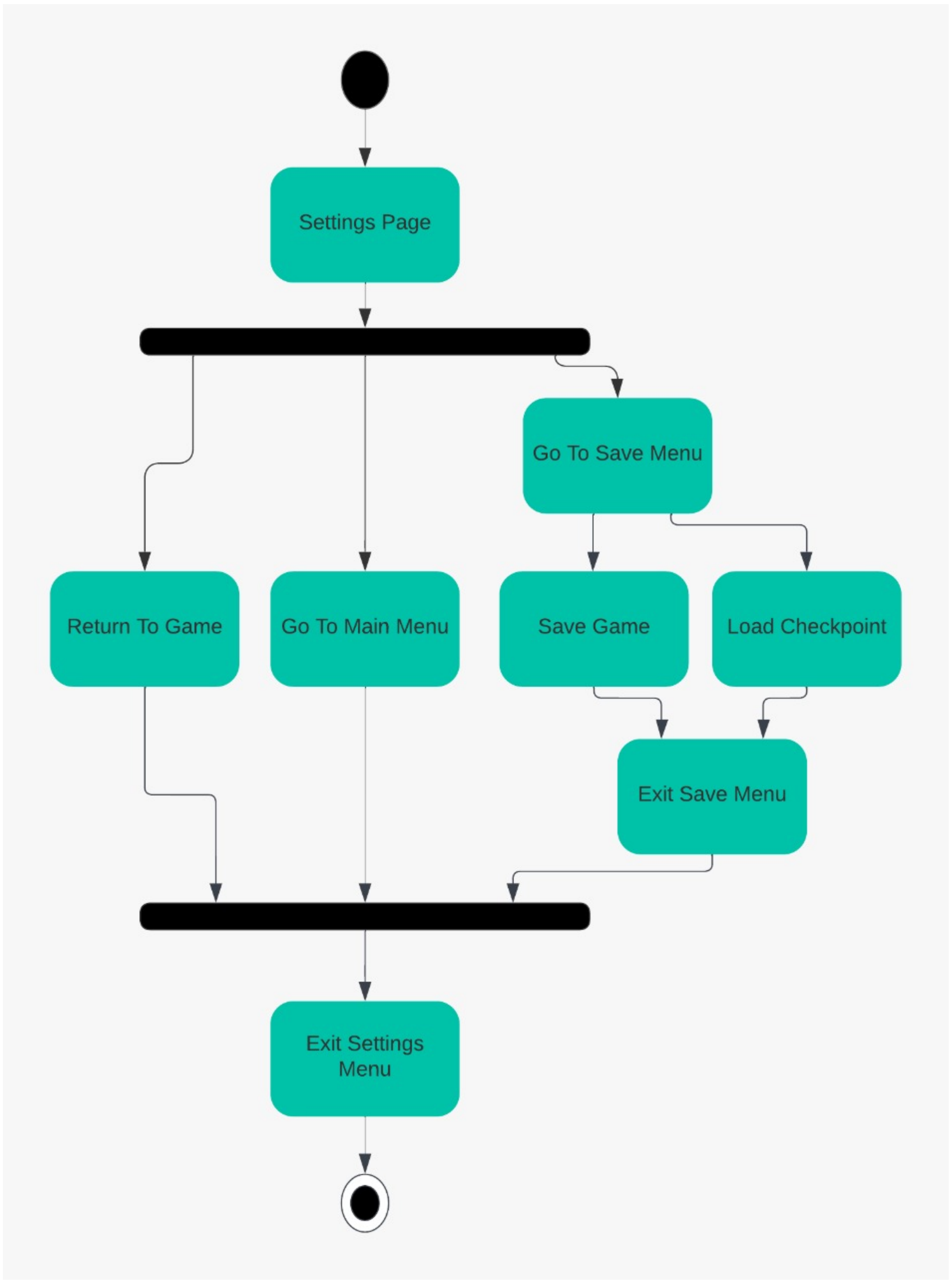
### Log In



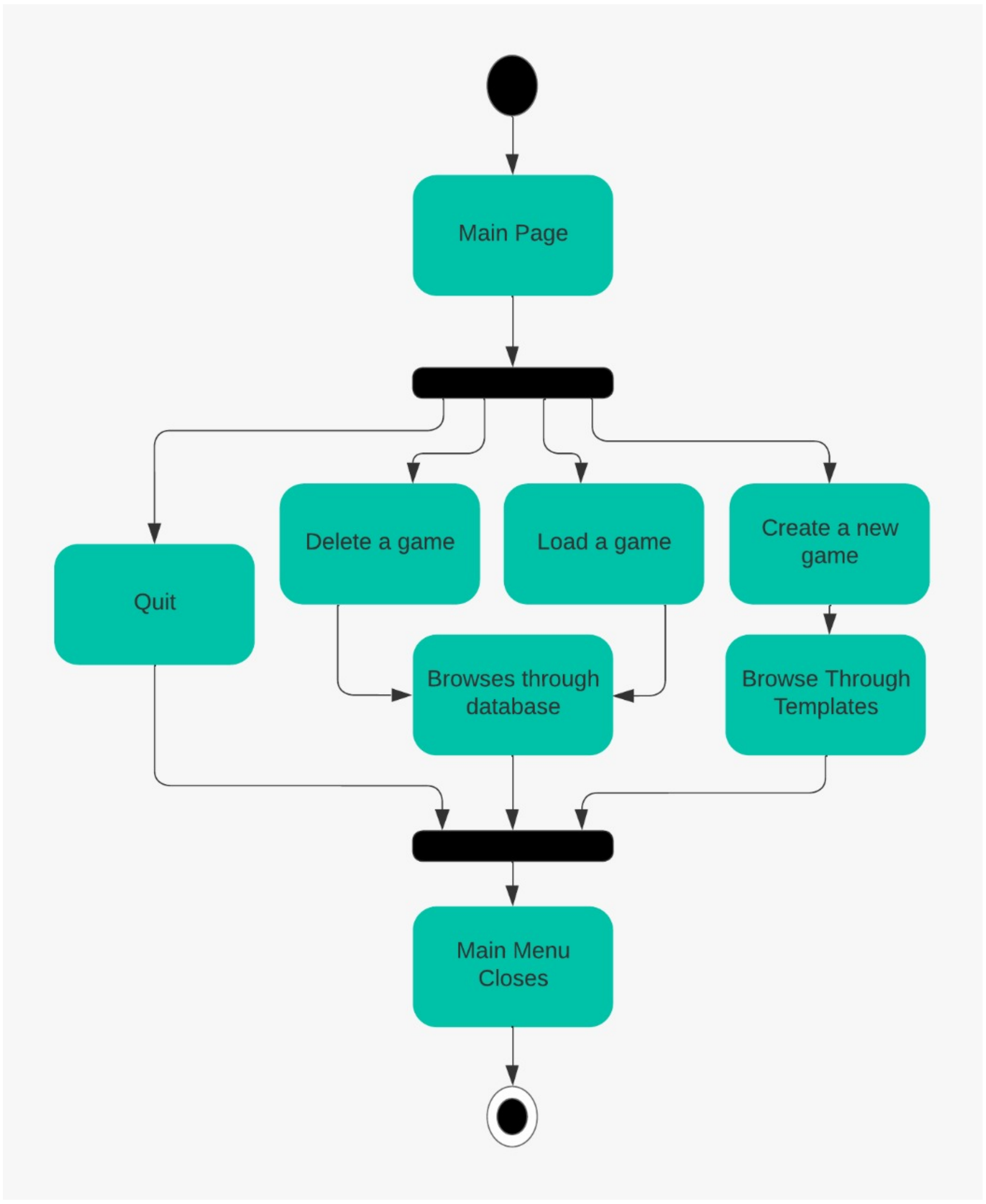
### Sign Up



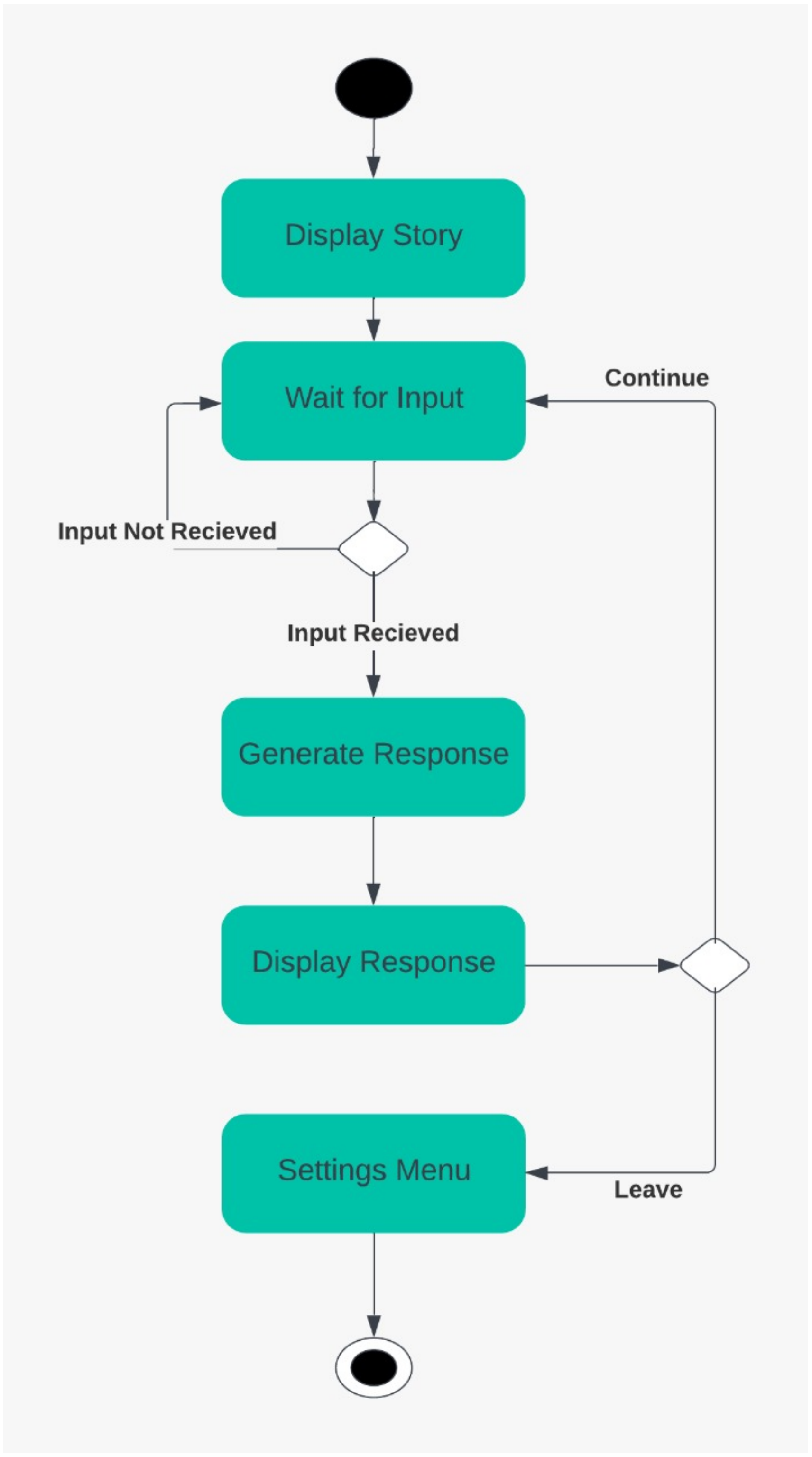
Settings Menu



Main Menu


Main Page
Quit
Delete a game
Load a game
Create a new game
Browses through database
Browse Through Templates
Main Menu Closes

Game Interface



## Objectives

- To create a fully fledged web app which is a Text Based Artificially Intelligent Gaming Adventure aka TAIGA.
- Detecting Hate speech and censoring it successfully in the application.
- Producing a story that converges to an endpoint and does not diverge too much from the main plot whilst also having random NPC encounters and world building.
- Text To Speech and Speech To Text capabilities for more user interaction using APIs' to call already available models.

## Individual Roles

- A web hosted text based adventure game with checkpoints
- Harsh: Development of the main model for the game and data cleaning
- Shrey: implementing Text To Speech capabilities and data cleaning
- Navneet: Implementing and tweaking various parameters to reduce discrimination
- Harshit: Implementing the hosting environment and creating the web app that will be used to play the game

# Work Plan

| Serial number | Activity | Month | March | | | | | April | | | | May | | | | | June | | | | July | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Week | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 1 | Formulation and Planning of projection | Plan | 🟦 | 🟦 | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 2 | Researching about GPT-3 and planning out the structure | Plan | | | 🟦 | 🟦 | 🟦 | 🟦 | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 3 | SRS | Plan | | | | | | | 🟦 | 🟦 | 🟦 | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 4 | Main model prototype | Plan | | | | | | | | | | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 5 | Integration of racism and sexism detection into the prototype | Plan | | | | | | | | | | | | | | | | | | | 🟦 | 🟦 | 🟦 |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 6 | TTS prototype and integration into main model | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 7 | Final Prototype | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 8 | System testing | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 9 | Creation of front end for the web app | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 10 | Integration of backend and front and hosting the final product | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 11 | Beta testing | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 12 | Finalising the project and tweaking | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |

| Serial number | Activity | Month | July | August | | | | | September | | | | October | | | | | November | | | | December | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Week | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 |
| 1 | Formulation and Planning of projection | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 2 | Researching about GPT-3 and planning out the structure | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 3 | SRS | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 4 | Main model prototype | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 5 | Integration of racism and sexism detection into the prototype | Plan | | | | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 6 | TTS prototype and integration into main model | Plan | 🟦 | 🟦 | 🟦 | | | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 7 | Final Prototype | Plan | | | | 🟦 | 🟦 | | | | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 8 | System testing | Plan | | | | | | 🟦 | 🟦 | 🟦 | | | | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 9 | Creation of front end for the web app | Plan | | | | | | | | | 🟦 | 🟦 | 🟦 | | | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 10 | Integration of backend and front and hosting the final product | Plan | | | | | | | | | | | | 🟦 | 🟦 | | | | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 11 | Beta testing | Plan | | | | | | | | | | | | | | 🟦 | 🟦 | 🟦 | | | | | |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |
| 12 | Finalising the project and tweaking | Plan | | | | | | | | | | | | | | | | | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 |
| | | Actual | | | | | | | | | | | | | | | | | | | | | |

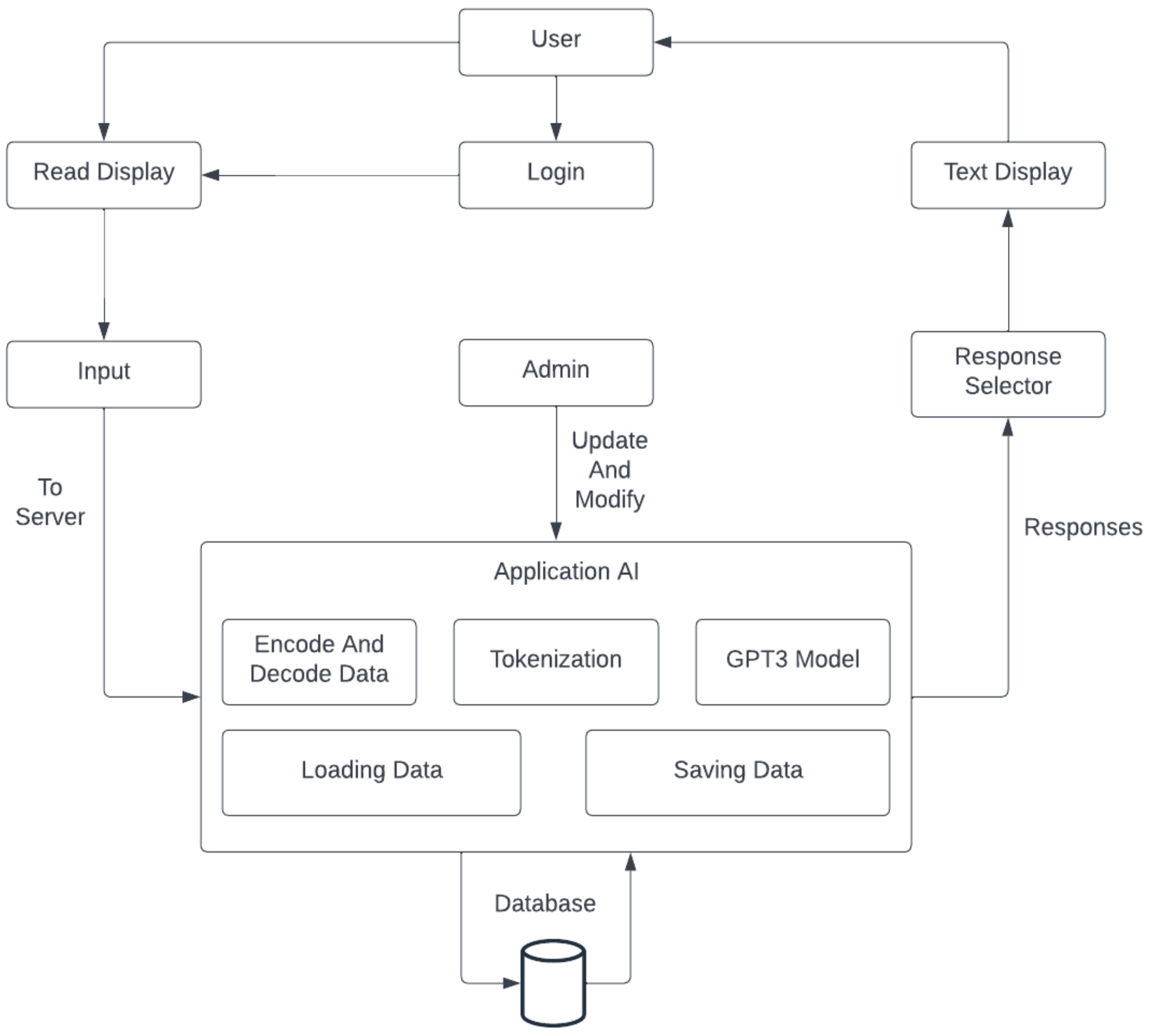
## Functional Requirements

- Input Validation - Only legitimate phrases should be entered. To get the intended result, the input provided should be specified. If a user types an invalid phrase, the system will prompt them to enter a valid phrase. The only language available for input is English.
- Database - The dataset is separated into categories in the database. The data is in json format and includes the genre, plot, choices and other information. The story that the user requested will be saved in the database.
- Model - The model will provide the summary of the story which is provided to it. The model will be fine-tuned to provide the best accuracy and user satisfaction.
- User Interface - The product summary will be presented alongside the story details on the user interface. The user will gain knowledge about the plot's main insight because of this. This summary can also be used by the user as a comparison tool with other genres.

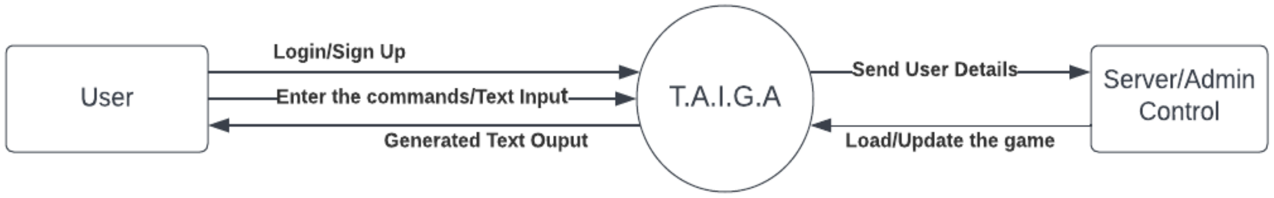
## Non Functional Requirements

- Maintainability - The development team will follow best practices for clean code and software modularity to make the application easily maintainable.
- Response Time - Users will be able to change between interfaces with minimum delay.
- Safety - A backup of the database must be taken from time to time ensuring timely recovery in case of any severe loss or a power cut. Also, information of the users must be stored in the database with proper authorization.
- Ease of Use - The system shall be able to be used by any user without requiring any training. This is achievable by adopting a user-friendly interface design. The user should only give input and get the result.
- Robustness - If there is any error in any window or module then it does not affect the remaining part of the software. It should be able to handle unexpected terminations and unexpected actions.
- Scalable - The system should be capable enough to handle all its users without affecting its performance.

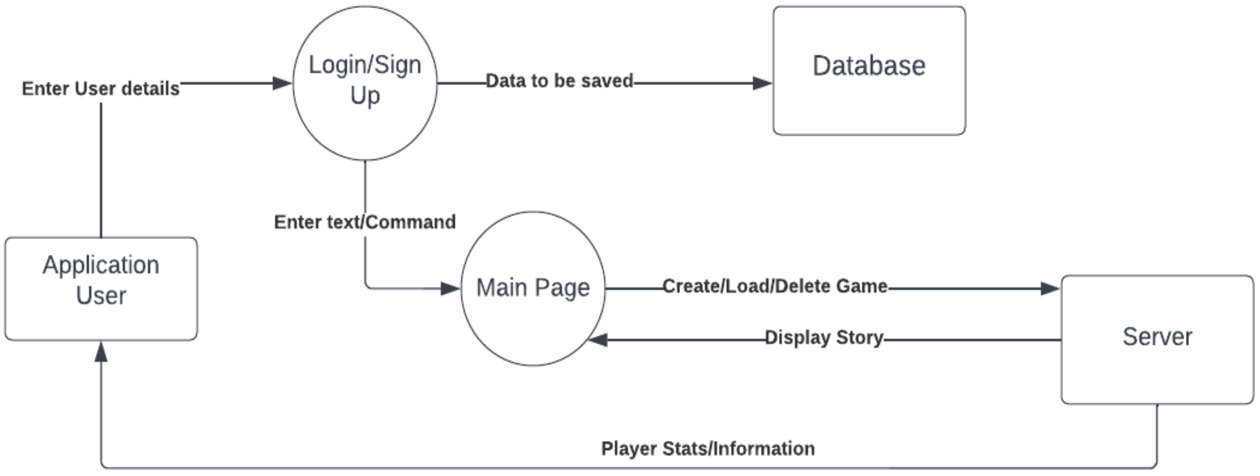
## Architecture Design



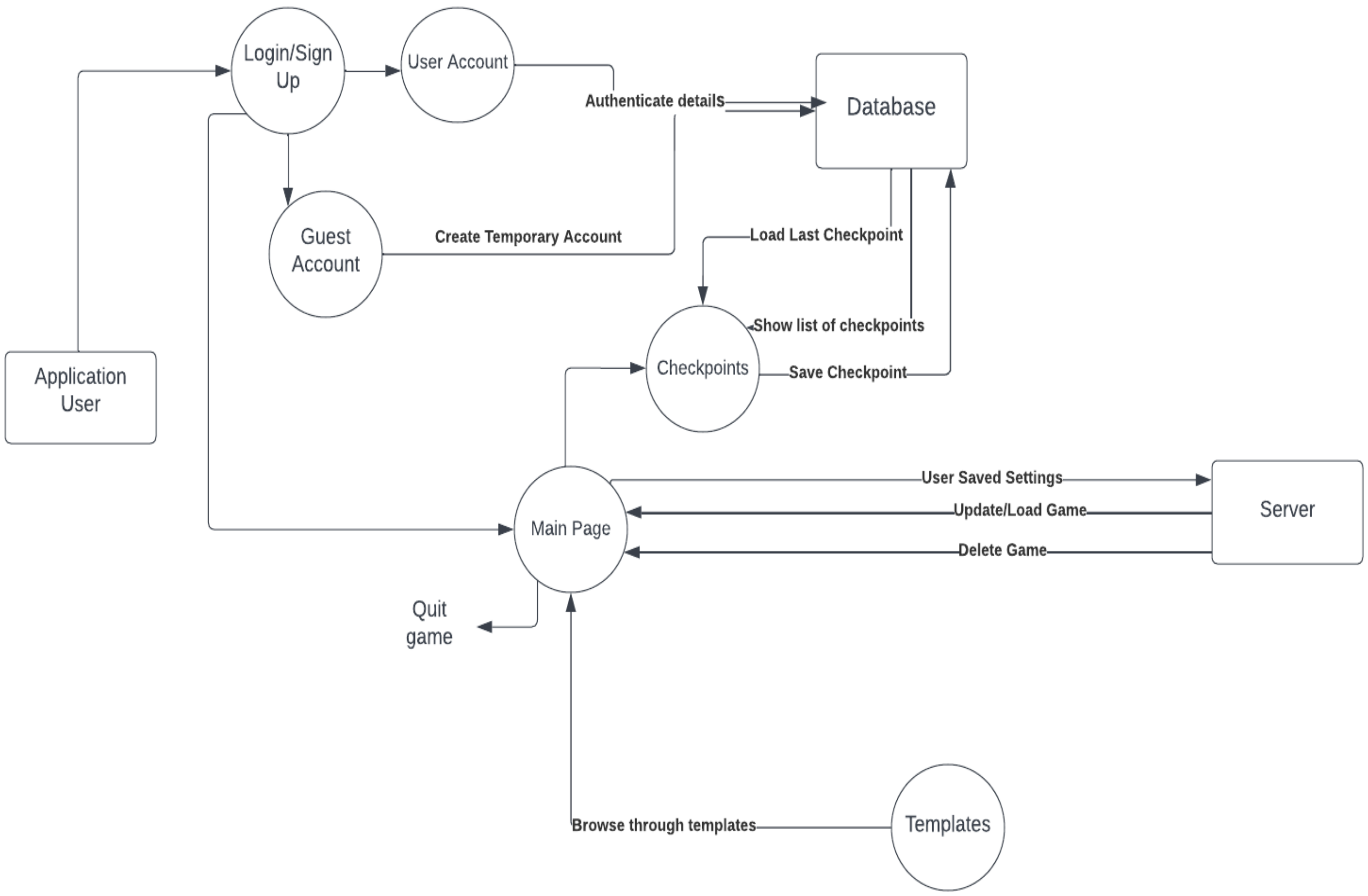
## Data Flow Diagrams



Level 0

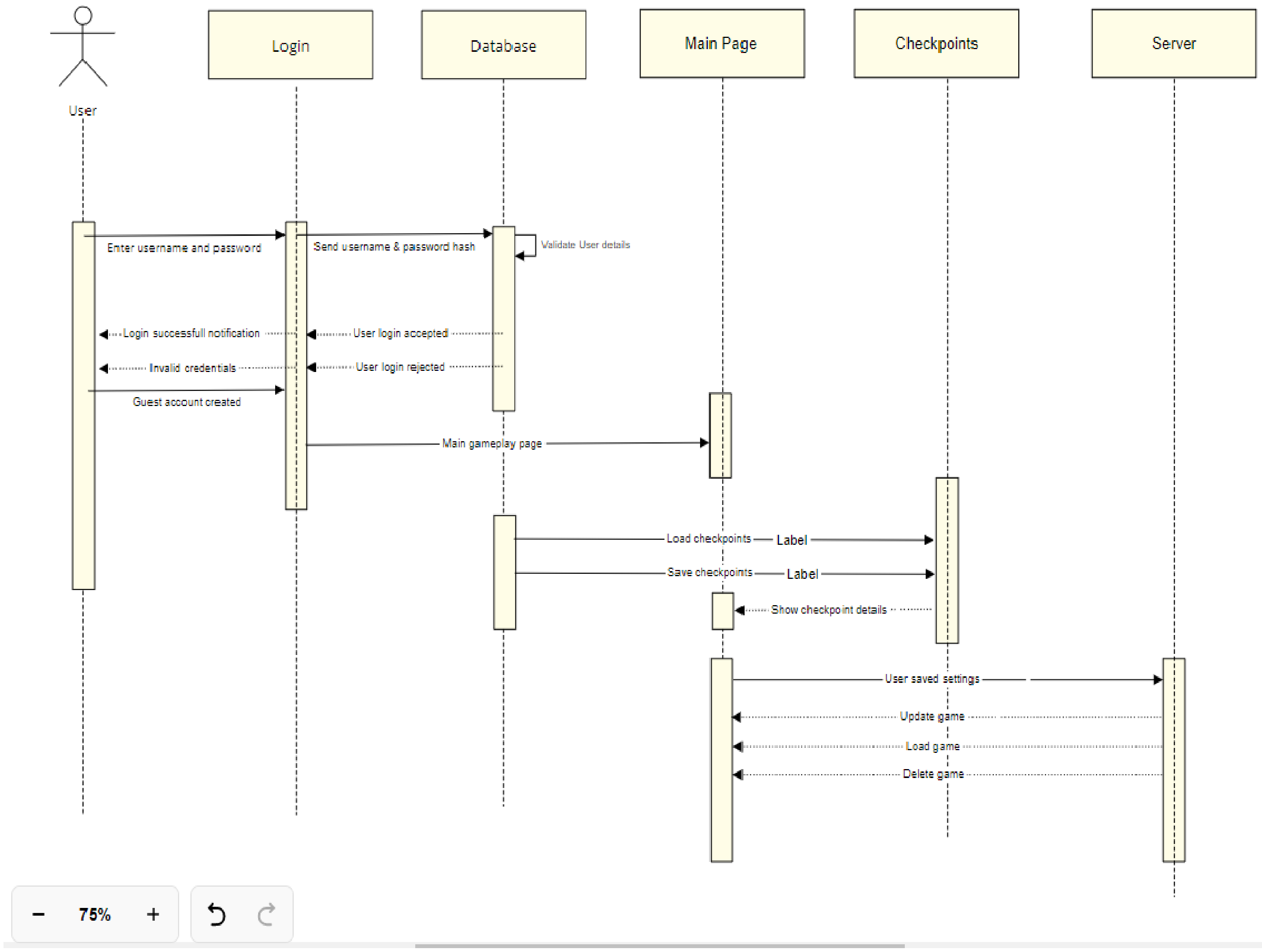


Level 1

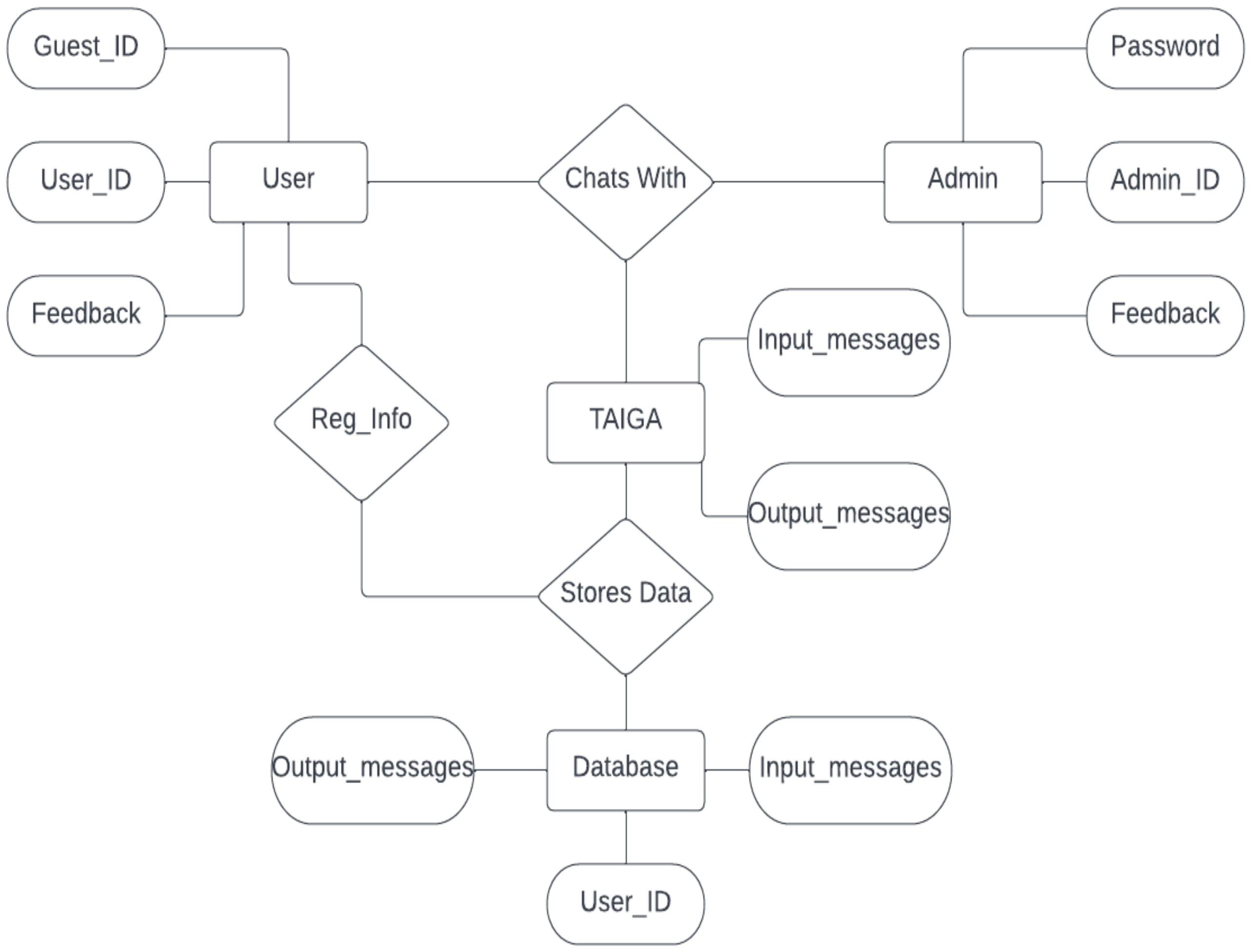

Login/Sign Up
User Account
Authenticate details
Database
Guest Account
Create Temporary Account
Load Last Checkpoint
Show list of checkpoints
Application User
Checkpoints
Save Checkpoint
User Saved Settings
Update/Load Game
Server
Main Page
Delete Game
Quit game
Browse through templates
Templates

Level 2

## Sequence Diagram



## Entity-Relationship Diagram



## Drive Link For All The Diagrams

https://drive.google.com/drive/folders/1LrF4FLbtOXrYWqROo27Mr2IhbTRmhX0C?usp=sharing