Team: 105-1

Current List of Features:

-The current list of features has not been significantly modified since milestone 2.

Player/Character class

- A set of different character types with varying features
- The ability to interact with the in-game environment

Storyline

- Will implement a designed and predetermined map
- A description of each place and potential interaction between the player and the environment
- Each "room" on the map will have a description and contain several collectibles, NPCs, or clues/information to advance the plot.

User Interface

- A display and a set of buttons or other input structures to allow the user to interact with all capabilities of the program

SQL Database integration

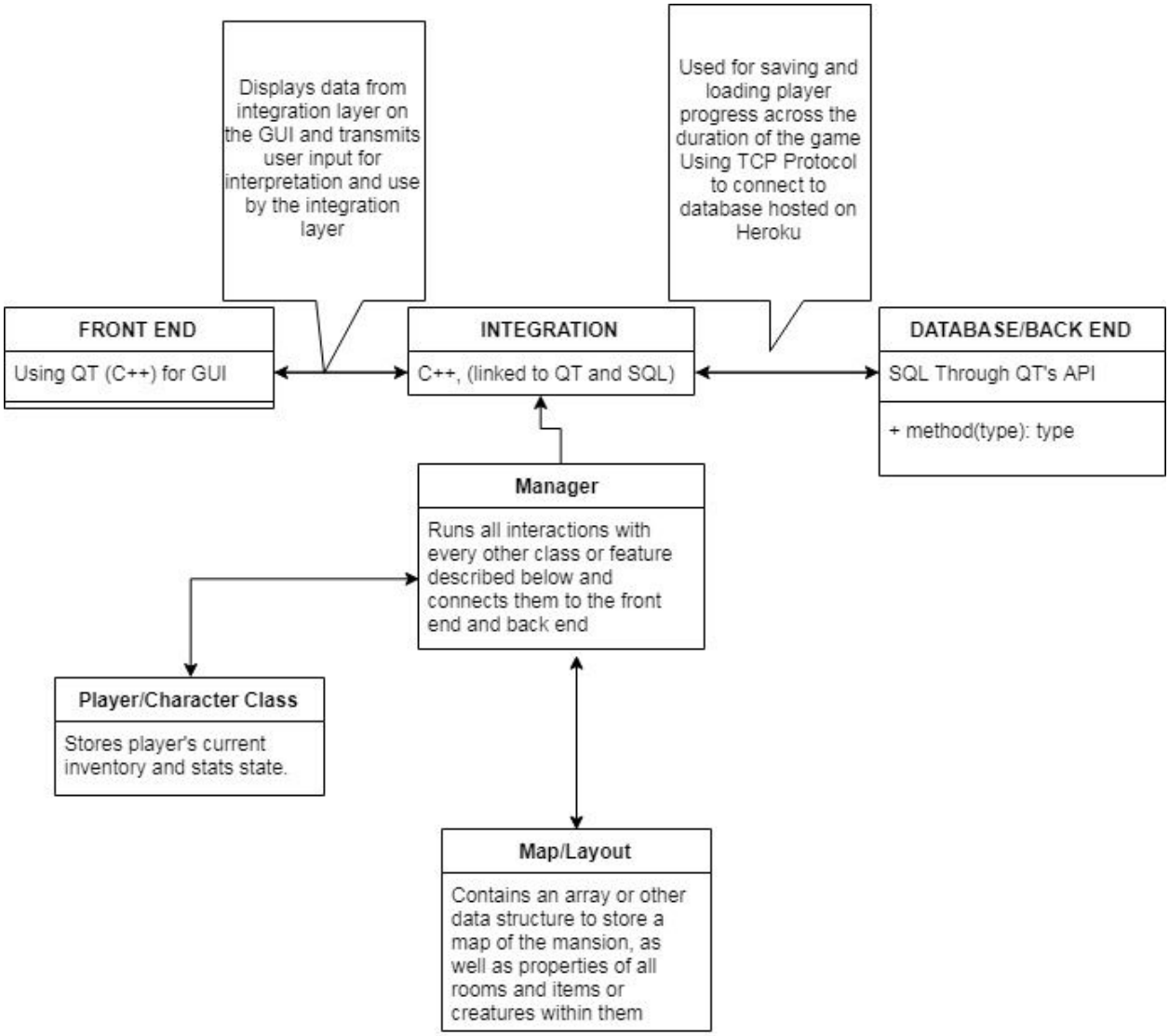
- Allows user to save a nearly instantaneous state of the game
- Database will be hosted on a server through Heroku--the game will only require an active internet connection to load or save the game state.

-

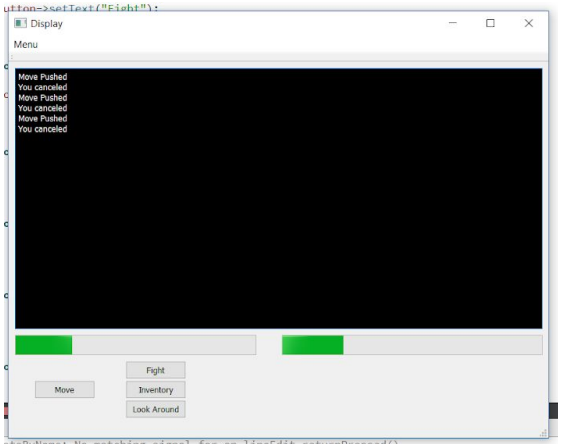
Feature Priority:

-A part of the map and a basic storyline would be completed first, to then have
-The player class(es) designed to fully interact with all objects and features of the map.
-The user interface will be completed after the player class(es) have completely been designed, to cover all functionalities contained therein.
-Since the functionality of most other components does not depend heavily on the workings of the back end and database, that will be built and deployed last.

Architecture Diagram:



Front End Design:



-The UI will display information about features within the current room on a terminal-like display

-There will be a series of buttons (depending on current game state), allowing the player to execute different tasks, pick up or inspect items, interact with NPCs, and a menu to save the state of the game (or reference documentation)
-a green status bar (like displayed above) will be used to visually represent a player's health.
-a visual implementation of the player's inventory may be added

Web Service Design:
Using QT PSQL API/Driver to link c++ integration layer to our database server hosted on Heroku
(To be updated when we cover more about web services in class)

Back End Design:
-Back end will be connected through the QT SQL API, running PostgreSQL
The database will run on a server over Heroku
We'll update this as soon as we learn about Heroku and relevant implementations
Our current design plan gives 3 tables, joined by a User ID field (through foreign keys)..
Since the database is later in our design, we will not be able to definitively describe each field the database will contain until we design most of the classes in the integration layer.
User ID / Username / Hashed Password (for authentication)
UserID / (Player Class information)
UserID/ (Map State Information)