Escape From Hellspawn - Architecture and Class Diagrams

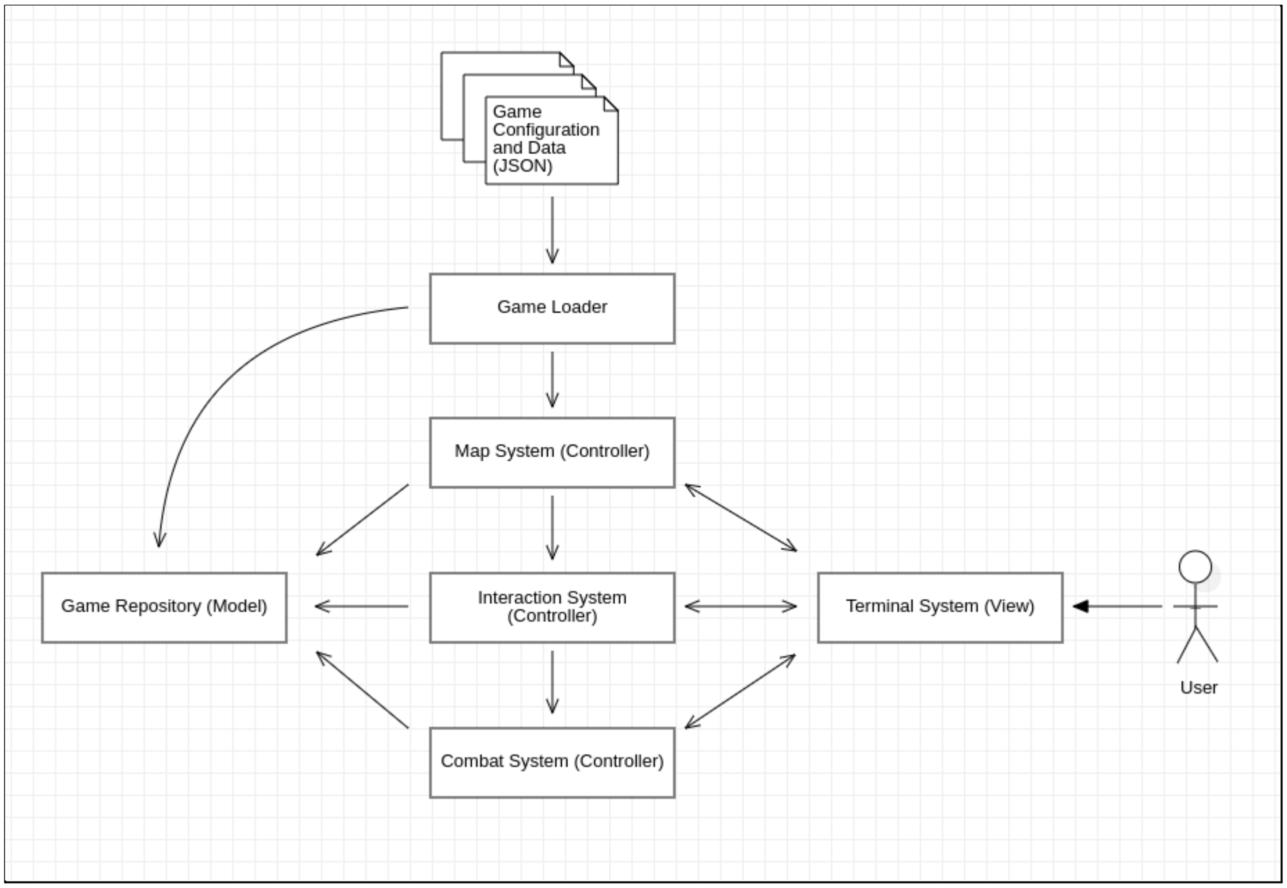
Toronto Metropolitan University

Course: CCPS406 - Intro to Software Engineering - P2022

Student Names: Jay Sharma, Dryden Zarate, Clement Zhang

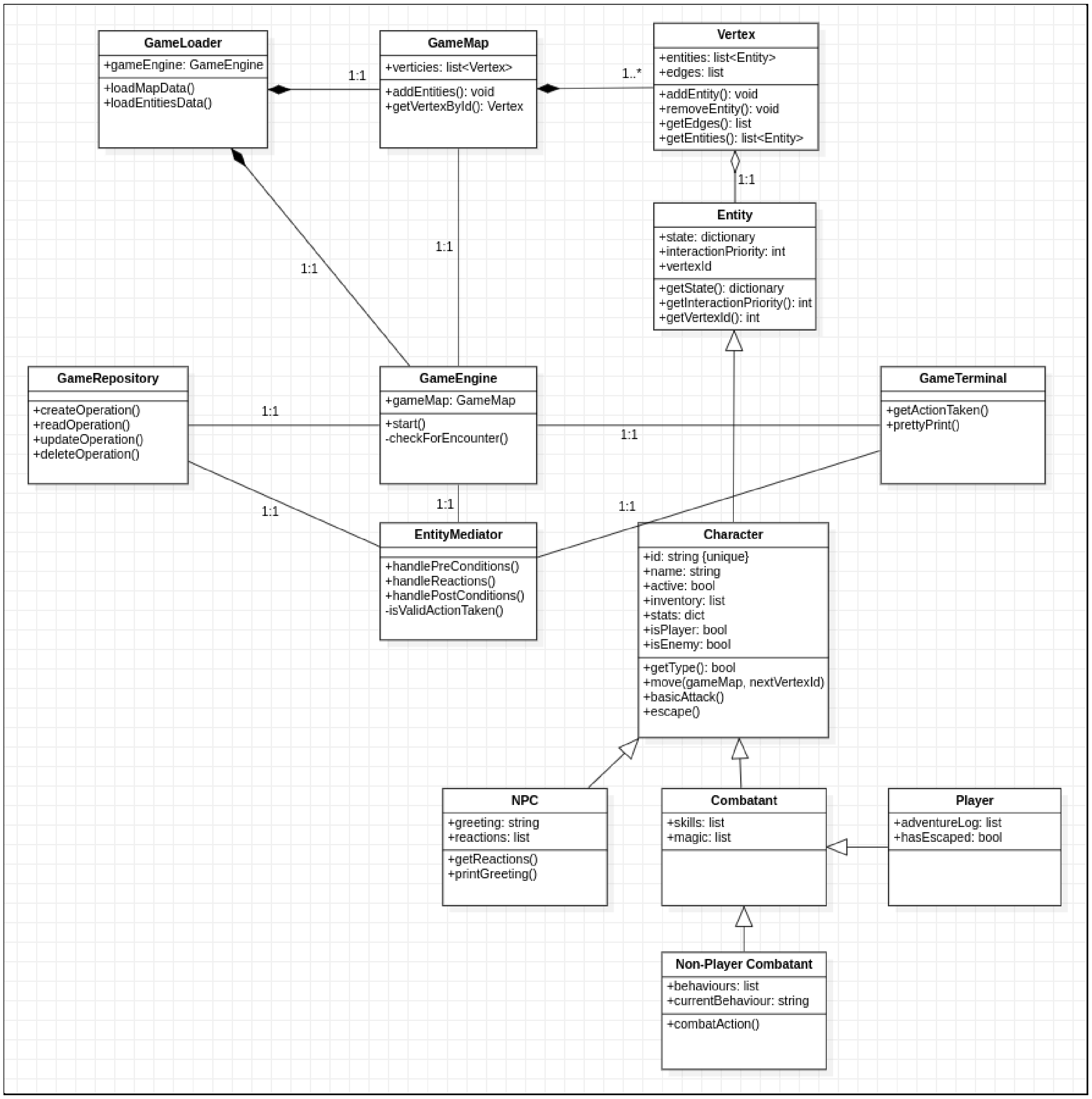
Date: May 22, 2022

Architecture Diagram


Game Configuration and Data (JSON)
Game Loader
Map System (Controller)
Game Repository (Model)
Interaction System (Controller)
Terminal System (View)
User
Combat System (Controller)

Our game engine follows the Model-View-Controller (MVC) architectural design pattern. From the diagram above, the starting point of the game engine is the **Game Loader**. It loads the game's map and entity data as program objects, thereby initializing the **Game Repository**, which is responsible for the Model aspect of MVC**.** The Game Loader also starts the **Map System** which contains business logic to facilitate game events which could trigger the **Interaction System** or the **Combat System**. Together these three systems are the Controller aspects of MVC. The Interaction System is responsible for interactions with NPCs whereas the Combat System orchestrates a battle between player and enemy. Lastly, there is the **Terminal System** which is the View aspect of MVC; it is responsible for formatting and displaying messages to the user and acquiring input from them.

Class Diagram



From the diagram above, GameMap, GameEngine, and EntityMediator form the **Controller** aspect of this program's MVC architecture. GameRepository is the **Model** and GameTerminal is the **View**. The GameMap is based on a graph data structure with edges stored by the relevant vertices. Each vertex knows what entities are in it and the allowed movements in each vertex is determined by the vertex's edges. Each entity has a local state which the GameRepository (Model) can perform CRUD operations on. These entity states and behaviour are all configured via JSON loaded at the start of the program by the GameLoader. The GameLoader also initializes the GameMap and starts the GameEngine. The user interacts with the program via the GameTerminal.