Architecture:

The components of our system are Input, Engine, Display. The player enters input through the keyboard that gets processed by our game engine and then displays what’s happening on the screen. The connectors are function calls that pass data to each component.

System Design:

The important modules that we will implement are Character, GUI, AI, Engine, and Input.

The Character module contains the information modeling each character. It will contain a record for each character with mutable fields including the character’s weights, speed, attacks, position, etc. Functions in the Character module (attack, run, change\_velocity) will change the fields of the character.

The GUI module will use the Graphics module of OCaml to open a graphics window and draw the stage, background, and the characters. It will redraw the updated characters on every tick of the game clock. The GUI module will have a list of Character records to draw each of their locations and animations and a draw function that will use the Graphics module functions to draw to the screen.

The AI module will implement the behavior of the Artificial Intelligence. Given the state of the game, the AI module will decide what the character should do.

The Engine module will handle most of the processing of the data. It will process inputs, execute inputs, handle collision detection, and take care of the physics of the game. It will handle gravity, calculate launch velocities, determine if characters can move or not based on hit stun, and calculate anything to do with the characters and change their positions.

The Input module will recursively call wait\_on\_event [key\_pressed] to process inputs from the user. Based on the key pressed, the Input module will figure out what needs to be done, then tell the engine to do it.

Module Design:

Contained in interface zip file.

Data:

Our system will maintain the data of the characters. The data will be kept in records with mutable fields and will be communicated using functions. The data structure Add stuff here