Eras

GP-05

CS 309

Jacob Petsche
Andrew Jones
Alec Lucas

Actors

- Players
- Database Managers
- Server Managers

Use Cases

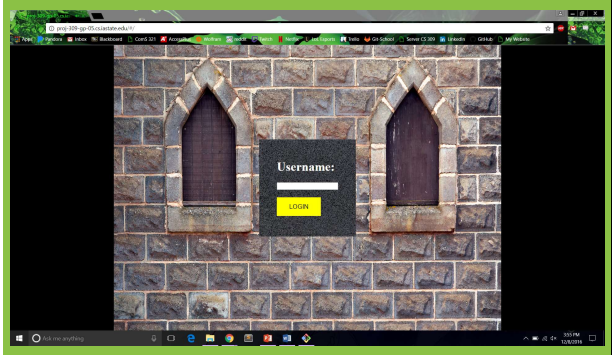
- Create Account
- Play Game
- Chat With Others

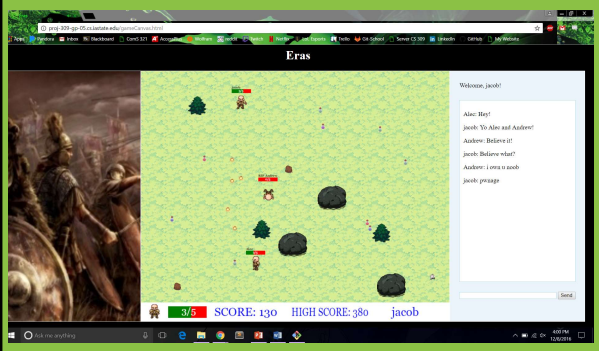
What Went Wrong

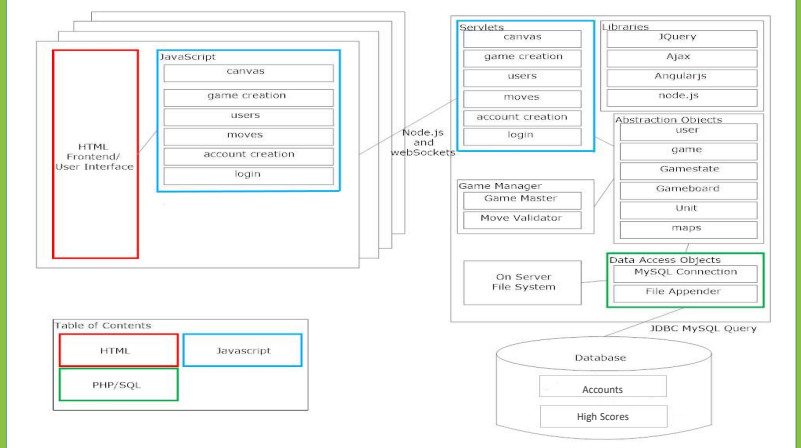
Lack of planning and previous experience.

What Went Right

Running, working project.
Fun for the whole family!





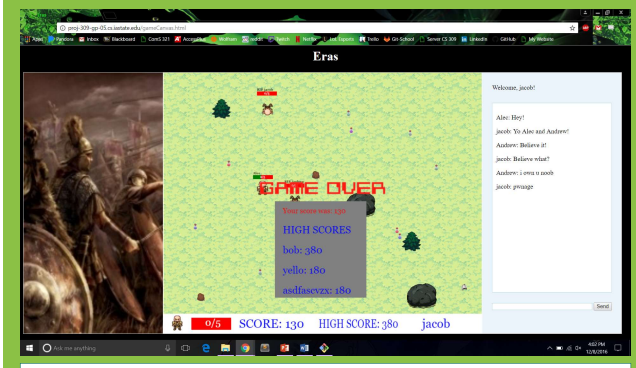


Project Description

Eras is a multiplayer, top-down, real-time brawler game. Once the player enters their desired username in the login screen they are put into the game. The game is a free-for-all where 2-4 players fight and score points based on damage done to, and kills of, other players. Each username has a high score attached to it, and the top high scores are displayed on the death menu.

Key Design Decisions

- Node.js to run Server
- Websockets for connections
- JSCanvas for game visuals



Lessons Learnt

- JavaScript
- How To Implement Scrum
- Server-Client-Database Interactions

Module Interfaces

ws.addEventListener("message", function(e){})
Listens for messages from the server or client, because websocket messages are being sent from both sides, and does action according to contents of the message.

Client

sendMove()
Send the movement commands from the client to the server through websocket connections.
updateHighscores()
Calls php file that finds top 3 scores from the database and sends those to the server.
drawFromServer()
Uses updated information from server and redraws entire canvas.
sendClose(id)
Send to server that a player has close browser.
Window.onload
When game file is loaded username is sent to the server and all the player's current scores are sent to the database.
window.addEventListener("beforeunload", function(e){})
When the player closes their window the server is notified that the player has left and the scores are updated in the database.

Server

connection.on('message', function(message)
An event listener that handles all of the commands sent from any client to the server. Handles movements, number of client connection, and messages sent to server.
wsServer.on('request', function(r){})
Runtime handler of basically entire game. Creating new player, movements, health, usernames, etc.
playerRespawn(object)
Respawns a player that had died.
setInterval(function(), 100)
Timed method call that updates projectiles every 1/10th of a second.
function wallCollision(object, direction, obstObj)
Conducts collision detection between the player and walls and the player and obstructions. This function is called when movement is attempted by the player.
function playerCollision(object, direction, playerObj)
Conducts collision detection between just players. This function is called when movement is attempted by the player.
function projCol(bullet, playerObj, obstObj)
Conducts collision detection between projectiles and walls, projectiles and obstructions, and projectiles and players. This function is called every 1/10th of a second when the projectile array is updated.
function spawnItems(itemObj){}
The function takes an item array as an argument and fills the array with randomized item objects.
function itemCollision(object, itemObj){}
The function takes an object and an item array as arguments. The object's (ie. player) position is tested against the position of the items in the array and returns if they are colliding.