MYORPG

Design Document

Team 24
Advisor: Mohammed Selim
Nadine Quibell: Design Lead
Henry Williams: Security Manager
Clayton Surfus: Server Developer
Jonathan Morris: Meeting Scribe

Executive Summary

Development Standards & Practices Used
MYORPG is developed using the Agile software development cycle.

Summary of Requirements
- Players can connect to the MYORPG website through a web browser
- Players can log in or create new accounts
- Gameplay allows players to explore the game world using the WASD keys
- Players can interact live with each other
- Players may chat with each other or the world as a whole using a semi-persistent chat
- Players may attack enemies using equipable weapons
- Players can collect and use items using a persistent inventory
- Players can upload and, with moderator approval, use their own player sprites and weaponry, given the use of an in-game item for creation
- Moderators can approve and suspend player-uploaded items, mute players, and request the banning of a player
- Admins can approve or deny a ban request and delete player-uploaded items, as well as use moderator powers
- Players can combine and upgrade weapons using in-game materials and the Weapons Forge location

Applicable Courses from Iowa State University Curriculum
COMS 309, COMS 319, COMS 363

New Skills/Knowledge acquired that was not taught in courses
- Much of our node.js knowledge was learned for this project or gained independently during other project classes (where it was not required)
- Where/how to host a website and database.

Table of Contents

List of figures/tables/symbols/definitions

[1]*MMORPG* - Massively Multiplayer Online Role-Playing Game. A genre of gaming in which players participate in an adventure together on a large scale.
[2]https://mmos.com/news/black-desert-announces-10000-costume-design-contest

1 Introduction

1.1 Acknowledgement

MYORPG has had significant time and effort put into it by all of its members.

1.2 Problem and Project Statement

Problem Statement: This project is driven by the lack of online role playing games that integrate their players into gameplay, specifically, allow for user-created content, short of modifications to existing games.

Solution Approach: MYORPG is a browser-based multiplayer online role playing game, also known as an MMORPG[1], which is built around the concept of allowing players to essentially build the game, allowing the upload of everything from player sprites to weapons to dungeons, using their own art and stat distributions. Our hope is that by the end of the project, users will be able to submit custom items or content for moderators or administrators to approve for the game proper, a unique experience that has gone unutilized in the gaming world. The output is this project is a fun, playable multiplayer online role playing game.

1.3 Operational Environment

The MYORPG environment will involve multiple users with a web based client connecting to a server. The users will be able to access the web page as an HTML file as with any webpage. The web page will display what the users need to play the game, as well as connect to multiplayer. The server itself will receive connections from multiple players, facilitate multiplayer, and store data via a database. While the project is currently being tested on a local network with direct IP connections and use of HTML and javascript files stored on a computer, the end product will be accessing the HTML through an online webpage and the server being connected over the internet. The webpage will be optimized for desktop computers and the game's interface will require a mouse and keyboard. There will not be any special runtimes required. The server itself can be a normal computer that runs the server with Node.

1.4 Requirements

As a video game MYORPG will contain several key components to make the game playable and a manageable experience. MYORPG will contain an account system accessible by all via a signup up and login page that can be accessed repeatedly. Using the login will connect the user to the server and be attached to a multiplayer instance. The game itself will consolidate several elements in one page. There will be a main game window from which graphics are displayed. There will be a chat window which allows a user to chat with others in the same instance of the game. There will be an inventory window for managing ingame items and tying into a crafting and shop menu which allows a user to create, purchase, or sell existing items. The game world will be controlled via the keyboard, and include a detailed overworld with grass and buildings, along with dungeons with a more constrained environment. Finally there will also be a combat system with different, more detailed, graphics. Combat will effectively be another menu that replaces that of the game world's when prompted. The game will need to support upwards of

eighty players in one instance, multiple user cases, uploading and rendering of graphics while maintaining an acceptable performance on all connected machines (30 frames per second) and be controllable by users with desktop computers.

1.5 Intended Users and Uses

The RPG has three main users, players, moderators, and admins. Players can login, register, play the game, and use items to upload new weapons made with custom images. They would have access to the screens associated with these activities. Moderators would have all the same privileges that players would. In addition they would be able to approve the custom images that players upload, mute players use the chat if they act inappropriately, and request a ban for players that routinely break the rules. These would be implemented via additional screens and buttons added to the chat window. Admins would have all of the privileges of both players and moderators. In addition to this they would be able to ban people and view requested bans through another window. They would have the most privileges out of all user cases as a result. The intended users of the game would be anyone with a computer and an interest in playing video games. Moderators and admins would be community volunteers who would spend extra time to ensure that the game's community remains fun for all.

1.6 Assumptions and Limitations

Assumptions:

- Our end product will be accessed through an internet browser worldwide, where the website is permitted
- Users in the sub-category of 'players' will be able to submit items, dungeons, or other custom content for users in the sub-category of 'mods' or 'admins' to be added to the game at a later date.
- Up to 32 players will be able to interact on one world at once, although more than 32 will be able to play the game at once (just not interact directly)

Limitations:

- Users will require an internet browser and internet connection to make use of our end product.
- User will need an internet connection to access the product website
- Project schedule will determine which features will be available in the initial release of the game

1.7 Expected End Project and Deliverables

A minimum-viable product version of MYORPG will be delivered for beta testing over the summer, in May 2020, with the end of S E 491. This version of the game will be a bare-bones but playable version of the game, allowing players to create accounts, log in, interact, and fight monsters. Item upload will be implemented and regulated, but without in-game balancing.

The end product of this project is MYORPG, a browser-based multiplayer RPG. The product will be available for use free of charge. It will require an internet connection and internet browser to

access. Access to the product will also be controlled on a black-list basis by administrators. The game will be completed in December of 2020, with the end of S E 492.

2 Specifications and Analysis

2.1 Proposed Approach

The product will be developed with a frontend acting as the client and a backend acting as the server. The focus of the frontend is to display the game. The server will handle the game logic. Both the frontend and backend will use the same language to keep code consistent and for ease of use. Our server will be written in JavaScript using node.js with the database being MySQL. The frontend will make use of HTML, JavaScript, and the canvas. The game will be browser based with anyone being able to play by having the website url. Making the product a browser based game has a benefit of running on any browser and computer that supports the HTML canvas. No specific code will be needed for a certain operating system.

The frontend will talk to the server to relay information such as players, player movements, player interactions, messages, and more. The server will then talk to all of the clients to keep them all consistent. The communication will be done through sockets. When a user logs in and starts the game the frontend will let the server know they connected. The server will then notify all of the clients connected to the game.

We have identified several node libraries to make use of in our project. The libraries allow us to rapidly develop and prototype certain features of the product such as multiplayer with ease. This allows us to focus on the task at hand instead of spending time reinventing the wheel. Several of the libraries such as socket.io are heavily documented and are the industry standard when working with node.js.

2.2 Design Analysis

Currently, MYORPG has a working backend and frontend communicating through sockets. Objects are movable on a canvas using the WASD keyboard keys. All movements are sent to the server and update on each player's canvas. Images are uploadable through the game's HTML page. Each image uploaded gets validated for file type and size. The images are stored on the server in a folder with a unique identifier. Users can create accounts with the information being stored into a MySQL database.

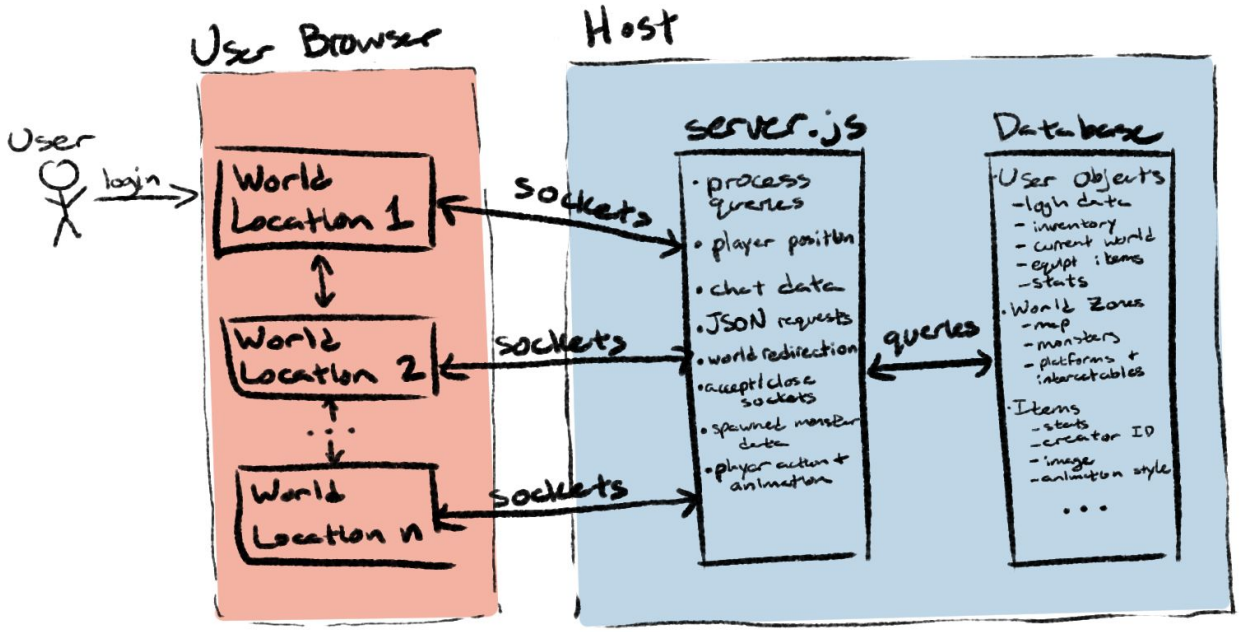
Thus far, our development on the project has been successful with significant progress being made each week.

2.3 Development Process

Our project will be following an agile development process. Everyone will be assigned a task for that week and move it through the Trello board until the task is completed. Weekly meetings will be set up to share current progress on their assigned task. Every two weeks the team will meet with the project advisor to share status updates on the current project. This will keep everyone involved within the project informed of where the project is currently at and where it will be

heading. Any project blocks that the team comes across will be maneuvered around early. By following the agile development process we hope it will help guide the project to success.

2.4 Conceptual Sketch



There are three main modules of MYORPG: the website, the node.js server server.js, and the mySQL database.

The website, which is what the user connects to and loads via a web browser, consists of a main menu page for login and a series of world zone pages for each of the areas the player can explore. Aside from the main menu page, the rest of these pages consist of generally the same CSS block and in-game menu blocks, with the map and player connections changing. There is no dependency between world locations, and they are entirely modular.

The server.js is the active server that runs the game from our remote host. It calls basic functionality from module scripts, such as logging in and uploading an object. The main functions of server.js are to connect to the players in various world locations and host their sockets, updating player data to the users at 30 FPS, and to interact with the mySQL database using queries.

The database is the static storage for MYORPG, storing user, item, monster, zone, etc. data. It interacts with server.js using query responses.

3 Statement of Work

3.1 Previous Work and Literature

MYORPG is loosely based on old-school MMORPG games, such as RuneScape, WoW, Guild Wars, Everquest… The basic mechanics of those games and the aesthetic of those early days of online gaming are the foundations of MYORPG. Creating a fantasy fighter, equipping ridiculous weapons, fighting monsters, and dungeon-crawling are not new concepts to the MMORPG genre and are being re-used in this game to incite the nostalgia those things bring with them.

New to the MMORPG genre and much of gaming, however, is MYORPG's concept of adding player-created content to the game as a core feature. Player engagement like this is a step up from forums and the occasional user-create content contests many of these games have [2]. With this, MYORPG hopes to foster a unique, creative community among the MMORPG community.

3.2 Technology Considerations

AWS is scalable enough that serverside burdens are not much of a consideration right now. Client-side, the game should run at 30 frames per second, and is only designed in mind for personal computers rather than for mobile devices. The controls will keep in mind that the user has a full keyboard and a mouse or trackpad available.

3.3 Task Decomposition

Fundamentally, there are three major tasks to decompose for our project: Player Accounts, Multiplayer Functionality, and User Interface. Between the three of them, there are a number of smaller tasks that show up in their decomposition. These tasks are building the player inventory, incorporating submitted cosmetics, the combat system, dungeons, and the separation between different player zones. One primary dependency underlying all of this is the multiplayer server itself. Without a multiplayer server, players will have no way to getting game data for their inventories, cosmetics, combat, dungeons, or even what zone they are in.

3.4 Possible Risks and Risk Management

Right now the main risks would be instability or downtime. AWS is robust enough to have little downtime. However if the client performs poorly or has major technical flaws the game will need to be left online until these issues are addressed. With continuous integration even the flawed version will be kept online, which will require evaluating whether to keep the game online even in the face of major security flaws. To manage this and keep downtime at a low rate only upon major vulnerabilities being discovered should the game go down. Otherwise even in the face of poor performance changes can be made while the game itself stays up.

3.5 Project Proposed Milestones and Evaluation Criteria

Each of the milestones below represent a part of the game system. The system will interact with each other to provide an overall good gaming experience. Only once the client approves of the work done will the milestone be considered completed.

Milestone	**Description/Satisfiability**
Accounts	Users are able to enter their information such as username, email, and password to create an account. They may use the information entered to sign into the game.
Multiplayer	Players can interact with other players and monsters currently connected to the game in real time. Each interaction by a player or monster shows up on each client's browser.
User Interface	Players are able to view and use interfaces to gain knowledge about their player such as viewing items currently in their inventory.
Zones	Zones split up the players into different areas of the game. Players are only able to see actions of the other players in their zone.
Items	Items enhance the player's character in the game. Player's are able to store items in their inventory.
Combat System	Players can fight monsters around the game. The system can use the player's stats and the monsters stats to calculate the damage done and resulting effects.
Dungeons	The game will create dungeons instanced to a player. Each dungeon will generate their own monsters, items, and more. The dungeon can only be used by the player(s) assigned to the dungeon.

3.6 Project Tracking Procedures

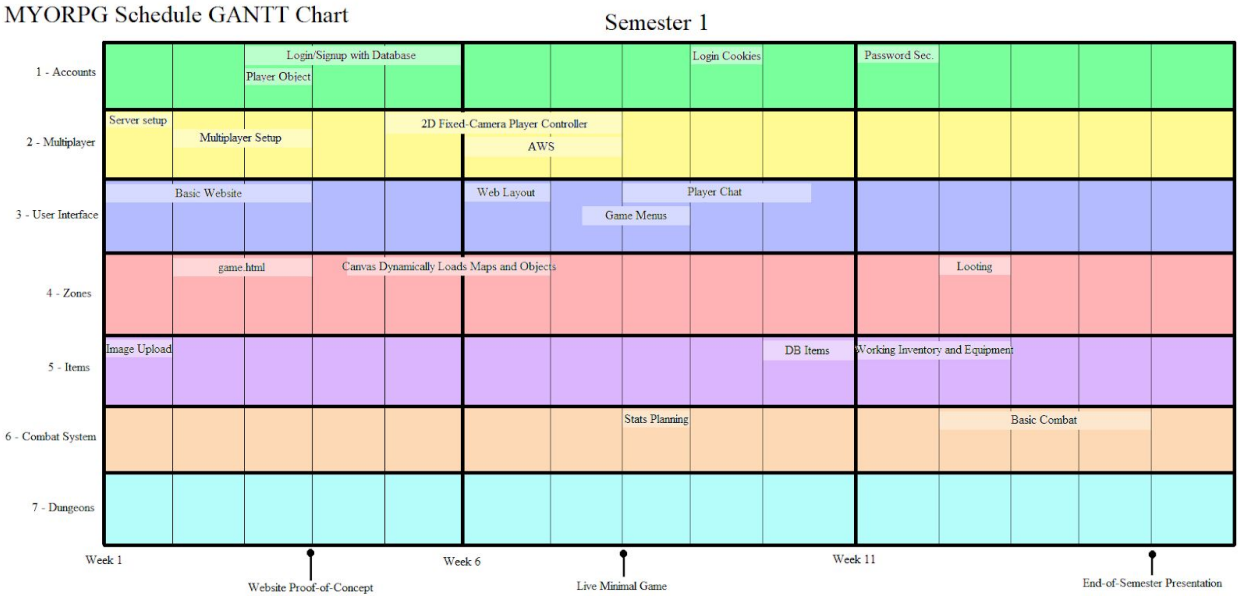
Trello will be used to store a backlog of tasks, current task progress, and completed tasks. Each team member will be assigned a task and update the task card as they make progress. Once the work for the task has been completed, the card should be marked as done.

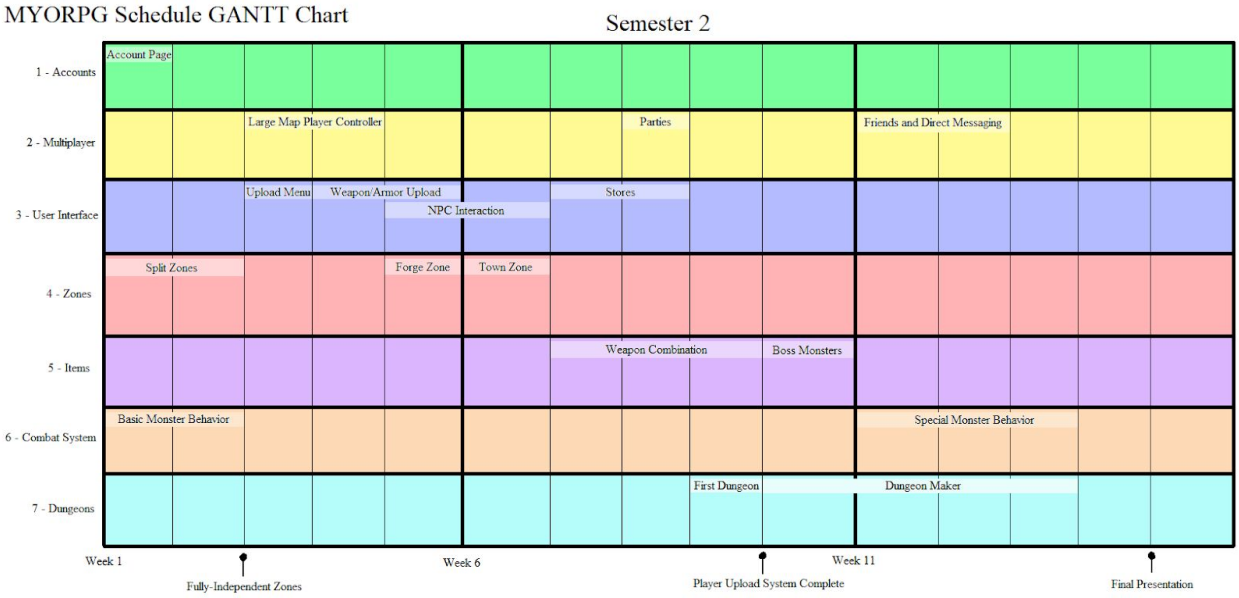
3.7 Expected Results and Validation

Each end product is expected to work alongside the rest of the project and depending on the product, independently. This is validated by running the result with the rest of the master branch on the AWS server and seeing if it works. When this is verified and any changes that need to be made are made, the result is marked as done.

4 Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline





Weeks 1-11 of the semester 1 timeline are tasks that have already been completed or are on the current sprint for the team. For the rest of the first semester, we plan to complete a minimum viable product for the game that showcases all it can do, if not all to its fullest extent. Players can make accounts, join the game, chat, fight monsters, loot items, and, of course, upload images. While still firmly in its developmental stages, this product could be released for its first live testing.

This allows the team, over the course of the summer, to gather valuable feedback on the combat system and UI experience, which will be used to make adjustments in the second semester.

While much of semester 1 is spent building the game up from the ground and making it a workable product, semester 2 is focused on adding functionality and really making MYORPG into what it's meant to be. The ability to make custom uploads into usable items, new zones, dungeons, and the forge are the main focus on this semester. Although efforts will be more focused in the first weeks on making zones more plug-in-able, after this, the team can optimize scheduling by being able to work on these separate pieces concurrently without worrying about them affecting other pieces. The last 2 weeks of semester 2 are reserved for schedule changes, testing, and finishing touches. Semester 2's schedule maintains a similar momentum as semester 1, and we plan to hold each other accountable to it, and allow for creative freedom in how it's made, by continuing our twice-weekly team meetings.

4.2 Feasibility Assessment

Our goal is to develop a MMORPG that involves customization allowing for the player to upload their own images to create their own items and characters. Players will then be able to take the items and use them to fight monsters in the game. Developing the game for the browser allows users to quickly create an account and jump right into the game. This makes the game easily shareable via a link. No downloads will be required by the user in order to play the game. The timeframe of the project is two semesters. The first semester focuses on prototyping and building the foundation while the second semester takes the foundation and builds on it. Getting the challenging tasks out of the way in the first semester allows us more time to come up with solutions to challenging tasks.

Potential Project Challenges	
Challenge	**Description**
Concurrent Player Count	The game needs to handle the max amount of players per server with no noticeable drop in gameplay experience.
Custom Items and Characters	Players are able to upload custom images for items. Additional support will be needed to make sure the items fit the game and don't degrade the experience of the game such as framerate drops or connection issues.
Combat System	Developing the combat system involves coming up with a well thought out experience for the player. The system needs to take the

	player's and monsters stats into consideration when calculating damage done. Additionally, it will display what is happening on the backend to the currently in combat player and other players in the current zone.
Monster Artificial Intelligence	Developing the intelligence for monsters will be a large undertaking as it will need to interact with the players and combat system. Monsters need to know where it can move, what it can attack, and more. Different monster types will have different intelligence as not all monsters move the same way. Additionally, the monsters will need to interact with the combat system as their main purpose is to attack players.

4.3 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be based on the projected effort required to perform the task correctly and not just "X" hours per week for the number of weeks that the task is active

Task	**Personnel**	**Projected Effort**
Database Table Creation	Backend Developer	The table will need to be designed carefully but overall is quick to implement.
User Interface Creation	Game Developer	Depends on the complexity of the component needing the interface.
Sending Data Over Socket	Game Developer Backend Developer	Quick to add data that needs to be sent over a socket.
Creating Unit Test	Game Developer Backend Developer	Depends on the complexity of the component needing the test.
Backend Feature	Backend Developer	Depending on the complexity of the task. The feature may be a small feature that could be implemented quickly. While the feature could be

		large and requiring weeks of work to be completed.
Canvas Game Feature	Game Developer	Depending on the complexity of the task. The feature may be a small feature that could be implemented quickly. While the feature could be large and requiring weeks of work to be completed.

4.4 Other Resource Requirements

Due to the solely software nature of MYORPG, no additional outside resources have been needed for constructing the project.

4.5 Financial Requirements

No financial resources were required in the making of MYORPG, as the team produced everything in-home using free software. In order to be hosted to a worldwide audience, however, a web host is needed to host the server and database for the game. MYROG is using Amazon Web Services to host a server. Currently MYORPG is using free credits for AWS for the time being. However, in the future, AWS will cost money to be used. In the table below are the costs and services to be utilized. AWS charges hourly based on type and size of instance being used. Our service will use an EC2 instance to run our code, S3 for storing custom images uploaded by the users, and RDS for hosting our database. In the future, if MYORPG needed to be scaled, the price would be increased by deploying more instances.

AWS Instance	**Cost (Monthly)**
EC2 Micro (Server)	$0.0116 * 750 Hours = $8.70
S3 (Storage)	$0.023 * X GB ($0.023 * 50 GB = $1.15)
RDS (Databases)	$0.017 * 750 Hours = $12.75
Total	$22.60

5 Testing and Implementation

5.1 Interface Specifications

There are no notable hardware/software interfaces involved in our project.

5.2 Hardware and Software

The MySQL database, Jasmine on Node.js, and an internet browser are being used to test the effectiveness of our product.

5.3 Functional Testing

Unit testing will be used to test critical parts of our application. As features get added, tests are needed to keep the older set of features working and running as expected. Our project is utilizing a javascript test runner called Jasmine. This framework will allow us to do both client side and server side unit testing. The runner can run in a web browser or via command line. Additionally, the runner can be configured to run tests in a random order. Running tests in random order can help catch uncommon failures.

Functional Requirement	**Test Case**
Register	A user can create an account
Sign In	A user can sign in to the game.
Move	A user can move their character in the game.
Items	Items are spawned into the game. Players are able to pick up items and the items show up in their inventory.
Collision	Players, platforms, and monsters have accurate collision detection. They will not go through each other.

Ultimately, the client will try out the changes once a feature has been implemented. If the developed feature meets their satisfaction then the feature is completed. Furthermore, our application is continuously deployed on AWS from the master branch of our git repository. In the future, unit tests will run before deploying. If the tests fail, the application will not be deployed.

5.4 Non-Functional Testing

Keeping performance, security, and compatibility in mind during development is key. Developers will test for performance, security, and compatibility during development of their feature. If the requirements are impacted harshly, the feature will need to be reworked.

Non-Functional Requirement	Outcome
Frame Rate	A steady 30 FPS is maintained while playing the game.
Security	User information is kept safe. Unintended actions are not possible in the game.
Communication	Players are able to communicate with each other.
Zones	Communication and actions that happen in zones are kept to their specific zone.
Compatibility	The game works on web browsers. Each client is identical to other clients.

5.5 Process

As each branch was made to implement specific features, a trello card was moved from the backlog to being in progress. This was verified to ensure that we were following the agile methodology. After commits were pushed to the gitlab page we could then check the final result via CI/CD on our AWS server. After features were handled by this automated system they manually verified to ensure that everything was working properly. If not then the feature could receive more work and the process repeated until the AWS server showed that everything was working as intended.

5.6 Results

In the future, our tests will ideally run when a branch is merged with the master branch. This will check to make sure the merged code does not break our functional features. This will be updated second semester to provide the results of our testing.

6 Closing Material

6.1 Conclusion

During the first semester of senior design, the MYORPG development team got decently ahead of the original schedule. Midway through the semester, a modified schedule (appendix 4.1.a) was put into place and a schedule for next semester (appendix 4.1.b) was created. Secure player login, the first game zone, multiplayer functionality including a player controller and chat, live web and database hosting, inventory, equipment, and minimal monsters are the major goals that have been completed so far.

Much of the groundwork for the rest of the game has already been laid. The main focus for next semester, then, is to tweak some base mechanics and implement features that will really define MYORPG as its own unique game. These mechanics are primarily the split of zone control (all zone information is currently processed centrally, and therefore not separated by zone), improved monster behavior, and the secondary player controller. The main feature implementations are other zones, the dungeon maker, and the weapons forge.

To complete these goals, we've set up the next semester's schedule beginning with the zone split. By completing the zone split first and foremost, this not only limits the amount of work that will be needed to sort out the code, but also makes the zones independently workable. Since the main features to implement are each their own zone, having the split allows team members to work on each section without the worry of their code affecting other completed code.

6.2 References

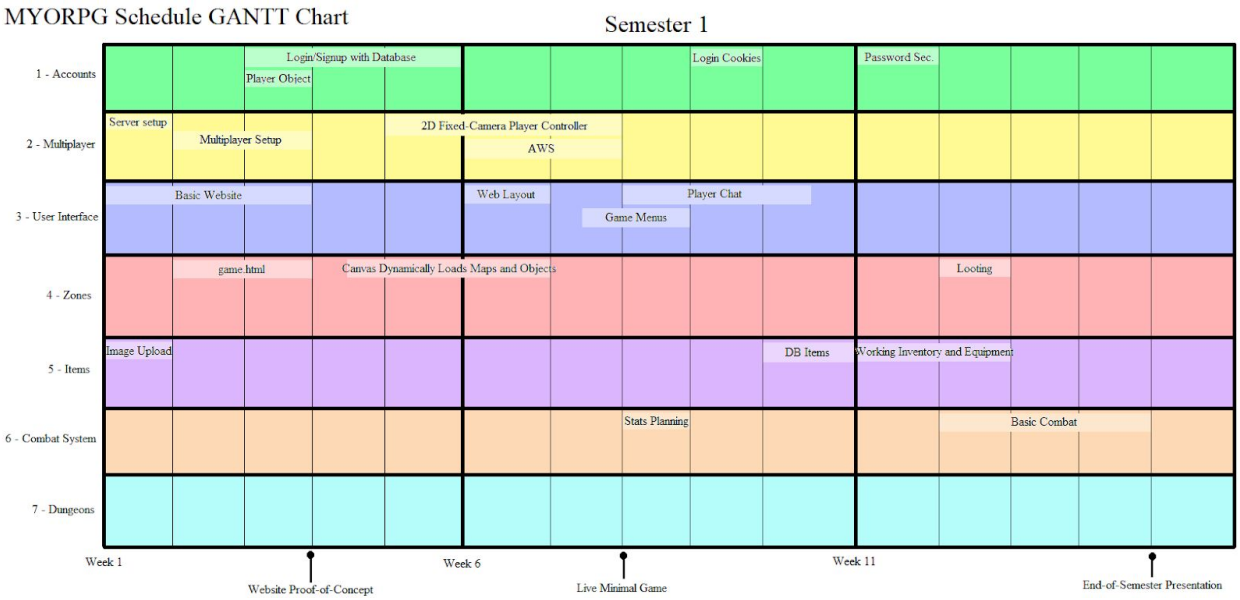
"Economics 2: EC2," *Amazon*. [Online]. Available: https://aws.amazon.com/ec2/pricing/on-demand/. [Accessed: 26-Apr-2020].

"Amazon S3 pricing," *Amazon*. [Online]. Available: https://aws.amazon.com/s3/pricing/. [Accessed: 26-Apr-2020].

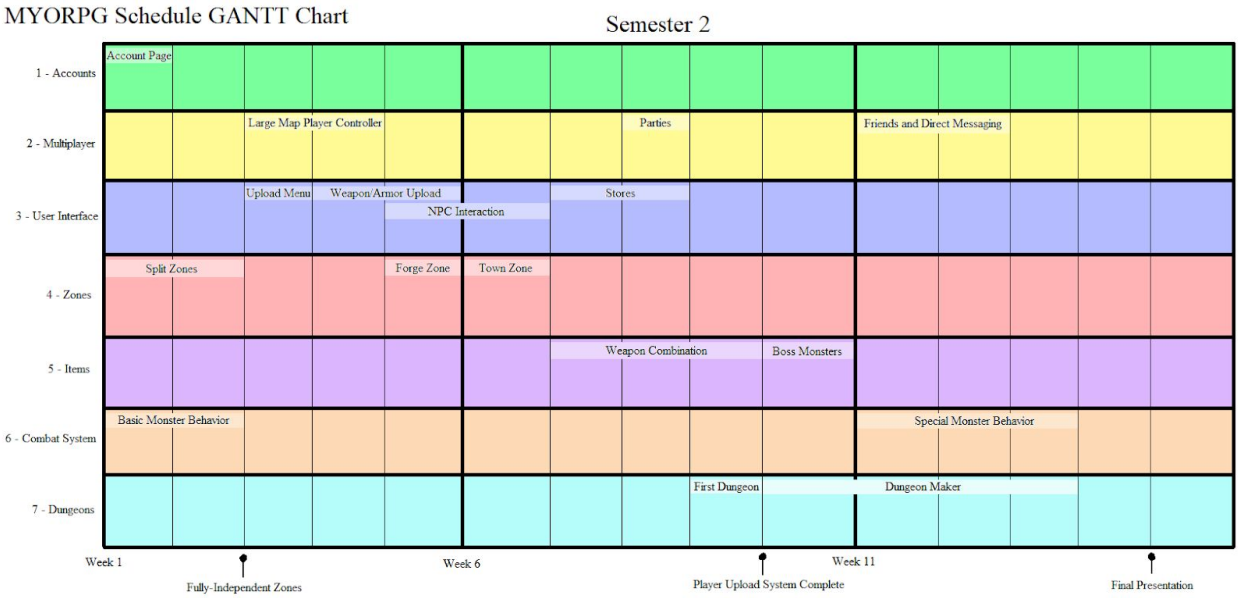
"Amazon RDS for MySQL Pricing," *Amazon*. [Online]. Available: https://aws.amazon.com/rds/mysql/pricing/. [Accessed: 26-Apr-2020].

6.3 Appendices

GANTT charts for semesters 1 and 2 schedules (from section 4.1):


MYORPG Schedule GANTT Chart
Semester 1
1 - Accounts
2 - Multiplayer
3 - User Interface
4 - Zones
5 - Items
6 - Combat System
7 - Dungeons
Login/Signup with Database
Player Object
Login Cookies
Password Sec.
Server setup
Multiplayer Setup
2D Fixed-Camera Player Controller
AWS
Basic Website
Web Layout
Player Chat
Game Menus
game.html
Canvas Dynamically Loads Maps and Objects
Looting
Image Upload
DB Items
Working Inventory and Equipment
Stats Planning
Basic Combat
Week 1
Week 6
Week 11
Website Proof-of-Concept
Live Minimal Game
End-of-Semester Presentation

4.1.a


MYORPG Schedule GANTT Chart
Semester 2
1 - Accounts
2 - Multiplayer
3 - User Interface
4 - Zones
5 - Items
6 - Combat System
7 - Dungeons
Account Page
Large Map Player Controller
Parties
Friends and Direct Messaging
Upload Menu
Weapon/Armor Upload
NPC Interaction
Stores
Split Zones
Forge Zone
Town Zone
Weapon Combination
Boss Monsters
Basic Monster Behavior
Special Monster Behavior
First Dungeon
Dungeon Maker
Week 1
Week 6
Week 11
Fully-Independent Zones
Player Upload System Complete
Final Presentation

4.1.b