Group 6

Quick Sands

September 24, 2020

"A game is an opportunity to focus our energy, with relentless optimism, at something we're good at (or getting better at) and enjoy. In other words, gameplay is the direct emotional opposite of depression."

— Jane McGonigal, Reality is Broken: Why Games Make Us Better and How They Can Change the World

(https://www.goodreads.com/quotes/tag/video-games)

Table of Contents

Introductions

Group 6:

- Robert Parker-Lak
- Andriy Ostapovych
- Faramarz Hosseini

Version: 1.00, updated December 09, 2020

Contract

We in Group 6 agree to the following conditions:

- We will meet every weekly deliverable as outlined in the Addendum for this course.
- The violence in this game will not be excessive and will not cross the line to areas of hate or discrimination of any kind. No bullying or reference to this topic will be present in the game.

We all understand the terms and conditions of this approval by Seneca.

Industry Analysis

Over the past two decades the video games industry has grown significantly. The revenue from video games before Covid-19 rivaled that of all the movies, tv shows and other media combined.

When Covid-19 reached Canada, it forced the country into lockdown, and this naturally caused a massive hit to many sectors of the economy. The video game industry was not an exception here, as many workers were forced to stay home. Despite this, it proved more resilient than many other sectors, and continues to make profits. People forced into isolation have more time available and many choose video games to pass the time. As such there is evidently demand for games.

Based on our vision, and work on this game so far, we believe that this game would be rated E. Taking this into consideration, we would like to point you to games with a Rating of "E for Everyone" ages 10+:

- The world-famous Pokémon, whose battle system mirrors our project (https://www.esrb.org/ratings/36456/Pok%C3%A9mon+Shield/)
- Super Smash Bros, a game with the only goal being defeating the other fighters (https://www.esrb.org/ratings/35839/Super+Smash+Bros+Ultimate/)
- The Zelda Series, an adventure game of fighting monsters across the world to save a princess (https://www.esrb.org/ratings/9947/The+Legend+of+Zelda/)
- Decap Attack, a game with a mummy that throws its Decapitated Head at enemies to defeat them (https://www.esrb.org/ratings/30035/Decap+Attack/)

Quick Sands is an Indie game based around the building and management of an adventure party that travels, trades, and engages in combat on transports that expand the party size and cargo capacity. Boost party stats with a loot drop based crafting system, gathering scales and chitin to craft class specific armor and weapons. Traverse an unforgiving barren world, conquer the beasts and rival factions controlling it, and unite the scattered remnants of humanity.

This game would appeal to many people's interests, and sales would be boosted by showing up on the PC marketplace such as Steam.

Mission Statement

Our project's objective is to build an MVP (Minimum Viable Product) of a PC Game Application that will prosper in the ever-growing demand for entertainment influenced by the current Covid-19 Pandemic Isolation.

When finished, the Application will serve as the Demo Product of an official video game company, for a Kickstarter which will aim at:

- Raising Funding
- Updating the artwork
- Adding a Campaign
- Adding a Multiplayer Arena
- Adding a Cosmetic-Micro Transaction Store

And many other interesting features.

This project aims to use the skills developed in the team members by Seneca to create a marketable piece of software, simulating realistic software time management and feature development.

Feasibility

Unity is a powerful game design software that is being used to precisely implement our game and compile it for Personal Computers. Most of the coding will be done in C#. The character design is handled by Photoshop for the art and Spine for bone rigging and animation, an incredible tool set to create 2D renders and animation for character models.

These tools are popular and widely used in the industry, providing a solid safety net of online tutorials for both feature development and error handling that might occur along the way.

Having a viable plan is one of the important steps in this project, keeping the project interesting with many solid core features, and interesting stretch goals. These goals would be alright to leave out but would enrichen the final product if there is additional time available.

We will enforce feature limitations to make sure our ideas do not exceed our abilities and time allocated. The result is not going to be an overly ambitious incomplete game, but an engaging RPG title that meets the expectations of the players and developers based on the set boundaries.

Combining powerful industry tools with comprehensive training material, and project guidelines placed on ourselves, we believe this project's completion to be feasible.

Schedule

The schedule is included in a separate document

Stakeholder Analysis

Stakeholder	**Stakeholder Interest in the Project**	**Assessment of Impact**	**Potential Strategies for Gaining Support/Reducing Obstacles**
Developers	Creating the Project	Maximum	Time Managed Plan, Communication Plan, Project Layout
Consumers	Using the Result	Maximum	Contacting the Customer Service
CRA	Making sure the revenue is taxed	Minimal	Inspecting the money cycle
Seneca	Sponsoring the developers	Moderate	Signing a contract with the developers
Instructor	Responsible for developers' final product	Maximum	Inspecting the whole process, Signing a contract with the developers
Distributor	Provides a marketplace which allows for online transactions, charging a portion of each.	Moderate	Abide by the terms of agreement, and the rating of the ESRB.
ESRB	Rating the game for a certain age.	Moderate	Abide by the rating of the ESRB.

The Stakeholders are:

- Consumers
- ESRB
- CRA
- Seneca
- Creative Designer
- Distributor
- Design Director
- Technology Manager

This list may be subject to change.

System Request

Quick Sands

Project Sponsor: Professor Ben Torres

Business Needs: Unlike many other sectors of the economy in the current Covid-19 Crisis, the video game industry is booming. This is due in large part to the fact that video games are a medium that you can enjoy on your own/in socially distanced groups. This video game is designed to appeal to a sizable part of the market in that industry, providing enjoyment and profit in equal measure.

Business Requirements:

- User must be able to launch and play game
- User must be able to select a character
- User must be able to explore game world
- User must be able to engage in game systems: travel, combat, and trading

Business Value: This game will provide entertainment for consumers and profit for the creators and investors. The current pricing outlook for this game is that the game will likely be free, but with an in-game store to sell additional content to consumers. This is a common pricing method for mobile games and can make these game a great deal of money.

Special Issues or Constraints:

- This project needs to be finished in four months.
- The full project must be finished in eight months.

Business Rules

1. BR 01: Users must have a computer to use Application
2. BR 02: Users must be above or at least 10 years of age
3. BR 03: User must have access to Internet to download Application
4. BR 04: User must have enough storage available for Application on device

Constraints

1. Regulatory Constraint - limited by Seneca's Academic Standards and our Contract
2. Academic Constraint - limited by the time we have available to complete the project
3. Technological Constraint - limited by the software available and their features
4. Funding – limited by the money that the project has available for assets, and software
5. Talent - limited by the skills that we have as developers/programmer

Functional Requirements

1. Provide ability to: Select Character (Warrior, Archer, Mage)
2. Provide ability to: Travel to Foreign Locations (Towns, Nests)
3. Provide ability to: Engage in Encounters (Positive or Negative)
4. Provide ability to: Load and Control Battles (Win or Lose)
5. Provide ability to: Manage Quests (Accept or Abandon)
6. Provide ability to: Collect Rewards (Quest or Battle)
7. Provide ability to: Buy or Sell (Trade Goods, Armor, or Weapons)

Non-Mandatory Goals

- A reputation system
- A faction system
- Special Abilities for each hero
- Additional touch-screen implementation
- A crafting system

Non-Functional Requirements

- Security: This game will be an offline game, not requiring online database security.
- Reliability: Reliability will be ensured through stringent testing before the release of the final product. The game will be stress-tested on all its potential platforms to ensure a smooth gameplay experience.
- Performance: With the advances in game engine technologies and the never-ending increase of graphical standards, it is challenging to maintain the appearance of the game alongside optimizing its performance on a wide range of devices, each with different hardware strength.
- Maintainability: We will be gathering feedback from users before and after the release of the game to learn what new features are wanted. Continued support after release will add new features, balance patch - microtransactions, patches
- Scalability: We will use the object-oriented model hierarchies to scale our classes, weapons, armor, enemies and transports, which increases and maintains performance as the project grows.
- Usability: Understanding game mechanics can be confusing for users with less experience. It is required for the game to be self-explanatory and use easy to remember controls while keeping the gameplay challenging and fun.

SWOT Analysis

Strengths	Weaknesses
Accessibility Indie Appeal Free to Play Expandable	Hardware Limitations Data Plan Limitations
Opportunities	**Threats**
Covid Pandemic Multi-platform	Distributor Issues Numerous Competitors ESRB Rating limitations Scope Creep

SWOT Justification

Strengths

- Indie Appeal: Independent games have a very large fanbase. This is a consumer base that this game will tap.
- Free to Play: This game will be free to download and play, giving it a broader appeal.
- Expandable: A game lives on through updates and new features. Having a story makes it possible and more convenient to bring content updates to our game, resulting in more relevancy.

Weaknesses

- Hardware Limitations: The game requires space on the device to install, and many users are limited by their storage.
- Data Plan Limitations: Users are often discouraged from downloading sizable games when limited by their Data Plan, they might hold off until they are connected to Wi-Fi or skip the product entirely.

Opportunities

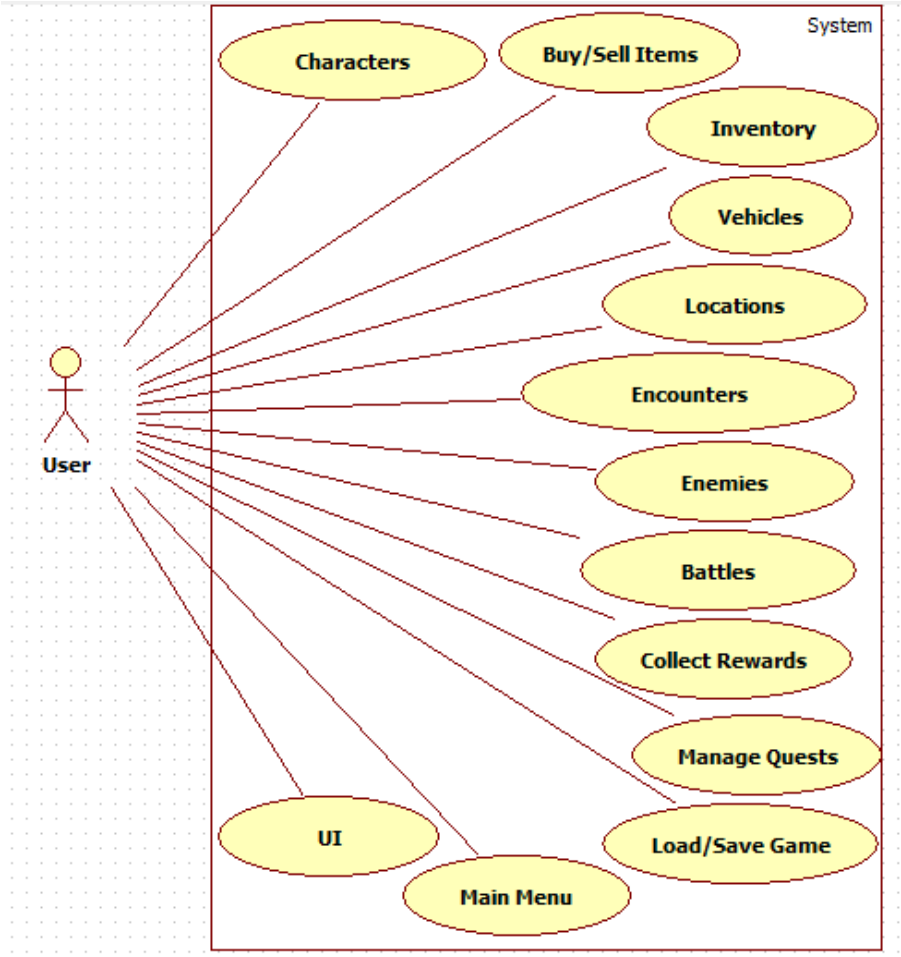
- Covid Pandemic: The Covid-19 Pandemic is causing intermittent lockdowns around the world. These lockdowns keep people inside, and as such, limit their options in day-to-day life. With less to do, more people are looking at new games on distributor sites. This gives our game a possible boost.
- Multi-platform: This game is multi-platform; it will be hosted on Google Play and on Steam, which will allow this game to be played on PC. This gives the game a wider market, and so more opportunities for sales.
- Marketing Campaign - Modern day Indie Devs are starting marketing early in the project development with the use of Development Update Videos (Dev Blogs) posted to YouTube. This generates a following, revenue, free alpha and beta testers, massive amounts of feedback, and even produces free game assets in the forms of artistic contributions from fans.

Threats

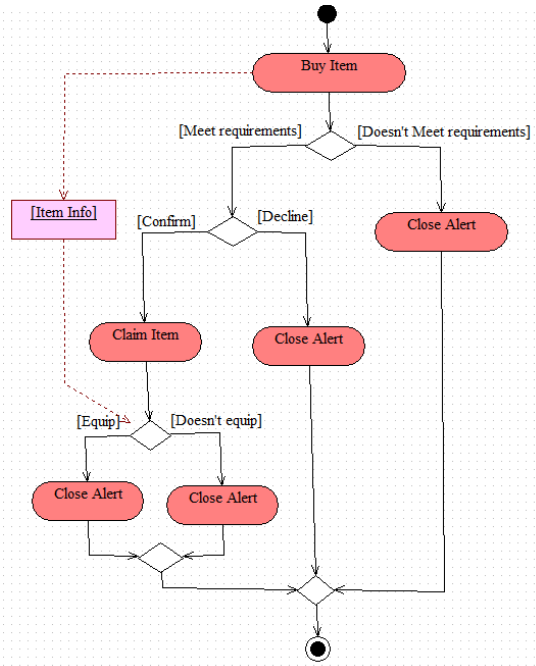
- Distributor Issues: The game will be available on two distributor stores: Steam and Google Play. If one of these distributors is having issues (e.g., a DDOS attack), then this game will be unable to download.
- Numerous Competitors: There are thousands of Indie games on the Steam store, so there is a lot of competition for this game. This could cause customers to choose another game over this one.

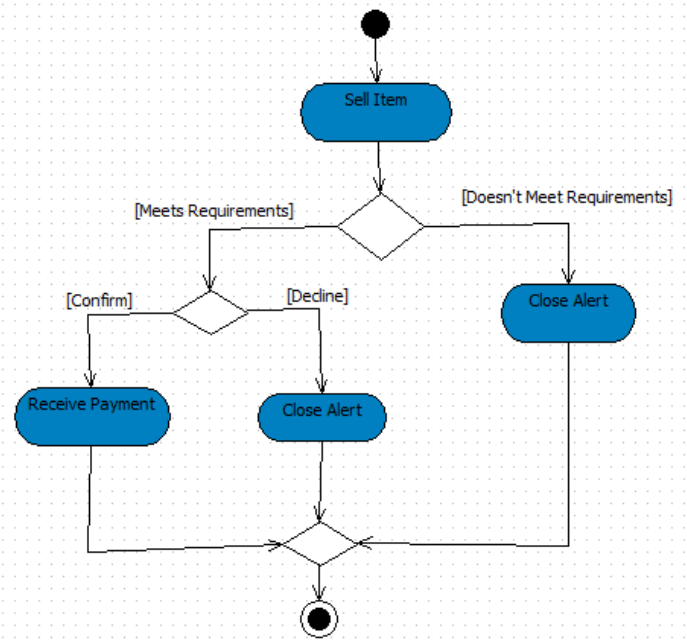
- ESRB Rating limitations: Anything above an E rating will limit the potential consumer base for this game.
- Scope Creep: Always a threat, the more we try to add to the development of this game the more work it will take to do it all.

Business Use Case Diagram



Activity Diagrams





Use Case Description

Launch Game

Use Case Name: Launch Game	ID: 1	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User - wants to play the game Distributor – Hosts the game		
Brief Description: This use case describes how a user starts the game		
Trigger: User wants to play the game Type: External		
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User taps the game icon on mobile device 2. User presses 'Play' to start game 3. System boots up, after a period of loading user is presented with the main menu		
SubFlows:		
Alternate/Exceptional Flows:		

Buy Items

Use Case Name: Buy Item(s)	ID: 2	Importance Level: Medium
Primary Actor: User	Use Case Type: Detail	
Stakeholders and Interests: User, wants to purchase in-game item(s)		
Brief Description: User purchases in-game item(s)		
Trigger: User decides to buy item(s) Type: Internal		
Relationships: Association: User Include: Extend: Generalization:		
Normal Flow of Events: 1. User opens dialogue with an in-game vendor 2. System displays list of items available for purchase 3. User selects desired item and presses 'Buy' 4. System checks if user has enough money 5. If user has met requirements for purchase, system pops-up request for confirmation		

6. User selects ‘yes’ 7. System removes money from user and places item(s) in inventory
SubFlows:
Alternate/Exceptional Flows: 5a. User does not meet requirements 6a. System displays message “Not enough money for purchase.” 7b. User closes trade dialogue

Sell Items

Use Case Name: Sell Item(s)	ID: 3	Importance Level: Medium
Primary Actor: User	Use Case Type: Detail	
Stakeholders and Interests: User, wants to sell item(s)		
Brief Description: User wants to sell item(s)		
Trigger: User decides to sell item(s) Type: Internal		
Relationships: Association: User Include: Extend: Generalization:		
Normal Flow of Events: 1. User chooses to sell item(s) 2. System checks if the user meets the requirements 3. An alert comes up showing the amount of gold the item is worth 4. User confirms selling the item 5. The item is removed from the user’s inventory 6. The user is given the amount of gold the item was worth		
SubFlows:		
Alternate/Exceptional Flows: 2a. User does not meet the requirements 3a. System displays message “Not eligible to sell item” 4a. User closes the alert		

Inventory

Use Case Name: Inventory Management	ID: 4	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User - wants to access, equip and manage items in their inventory		

Brief Description: This use case describes how a user uses the database that contains all of the items they have acquired, including equipping or dropping an item.
Trigger: User wants to interact with an item Type: Internal
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:
Normal Flow of Events: 1. User taps the inventory icon on their UI 2. System displays inventory screen, with items delineated by icons 3. User selects a basic item icon 4. System displays information on item 5. User selects drop icon 6. System displays message “Drop Item?” 7. User selects “Yes” 8. System removes item from inventory 9. User drags item icon to equip slot 10. System puts item into character’s equip slot 11. User drags item from equip slot 12. System removes item from character’s equip slot 13. User selects “X” on top-right of Inventory screen 14. System closes inventory screen
SubFlows:
Alternate/Exceptional Flows: 7a. User selects “No”

Vehicles

Use Case Name: Vehicles	ID: 5	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to travel from location to location, have a larger inventory and fun items		
Brief Description: This use case describes how a user interacts with in-game vehicles		
Trigger: User wants to use a vehicle Type: Internal		

Relationships: Association: User, Host Application, Game Include: Extend: Generalization:
Normal Flow of Events: 1. User interacts with vehicle merchant 2. System displays list of vehicles available for purchase 3. User selects vehicle to purchase 4. System removes vehicle from merchant list and adds vehicle to user vehicle list 5. User clicks on “X” in top-right corner 6. System closes vehicle merchant screen 7. User clicks on Vehicle drop-down list and selects new vehicle 8. System sets vehicle to user current vehicle, removes all party members from user
SubFlows:
Alternate/Exceptional Flows: 3a. User clicks on “X” in top-right corner 4a. System closes vehicle merchant screen

Locations

Use Case Name: Locations	ID: 6	Importance Level: Medium
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User, wants to access another part of the in-game map		
Brief Description: This use case describes how a user changes their in-game location		
Trigger: User is ready to leave area Type: Internal		
Relationships: Association: User, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User presses ‘World Map’ button 2. System game map opens, highlights accessible areas with individual buttons 3. User selects a location button to travel to that area 4. System displays pop-up confirming desire to travel to new area 5. User selects ‘Yes’ 6. System displays loading screen as it loads Travel/Battle Scene 7. User is placed in starting area of Travel/Battle Scene		
SubFlows:		

Alternate/Exceptional Flows: 3a. User presses ‘Close Map’ 4a. System closes map, displays player characters again 5b. User selects ‘No’ 6b. System closes pop-up, displaying map again

Encounters

Use Case Name: Encounters	ID: 7	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to be able to engage in random encounters when travelling		
Brief Description: This use case describes how a user interacts with encounters		
Trigger: user is travelling and an encounter randomly spawns Type: Internal		
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User is travelling between locations 2. System generates a number of encounters when the user starts travelling based on length of trip with random types 3. System spawns an encounter of a battle type 4. System displays battle event 5. User engages in battle event and wins 6. System displays reward screen 7. User selects rewards and closes rewards screen 8. System puts user back into travelling scene 9. System spawns an encounter of trade type 10. System displays trade screen for user 11. User trades as desired and closes the screen when done 12. System puts user back into travelling scene 13. User reaches destination 14. System pulls user out of travelling scene, puts them in location		
SubFlows:		
Alternate/Exceptional Flows:		

Battles

Use Case Name: Battles	ID: 8	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to be able to fight enemies		
Brief Description: This use case describes how a user fights a battle		
Trigger: user is travelling and engages in a battle encounter Type: Internal		
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User encounters a battle 2. System spawns' enemies and displays the Battles screen 3. User chooses action: attack, item, run away 4. System enacts user action 5. Enemies do action 6. System enacts enemies' action 7. Repeat steps 3-6 until User wins, loses or successfully flees 8. User Wins 9. System displays rewards screen 10. User chooses rewards and closes screen 11. System closes battle screen		
SubFlows:		
Alternate/Exceptional Flows: 8a. User loses 9a. System closes battle screen 10a. System moves player back to previous location 11a. System removes all rewards player had earned on that travel 12a. User resumes play 8b. User successfully runs away 9b. System closes battle screen and puts player back into travel scene they were in 8c User Unsuccessfully runs away 9c User loses turn		

Rewards

Use Case Name: Rewards	ID: 9	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to be able to acquire items		
Brief Description: This use case describes how a user gets rewarded after completing a quest or a battle		
Trigger: user completes a quest or wins a battle Type: Internal		
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User wins a battle or completes a quest 2. System calculates rewards based on numerous factors 3. System displays rewards screen with items it calculated should be in there 4. User selects items they want from rewards screen 5. System moves selected items into player inventory 6. User closes reward screen by clicking “x” on top right of screen		
SubFlows:		
Alternate/Exceptional Flows:		

Quests

Use Case Name: Quest	ID: 10	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to be able to complete quests for rewards and entertainment		
Brief Description: This use case describes how a user begins and finishes a quest		
Trigger: user interacts with a quest-giver Type: Internal		
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User interacts with a quest-giver		

2. System displays list of available and active quests 3. User selects a quest from the list 4. System displays quest message, explaining details about the available quest 5. User selects “Accept” 6. System adds a quest to the user and moves quest from available list to active list 7. System displays available quest list 8. User selects “Active Quests” 9. System displays active quests list 10. User selects “x” on top right corner of quests screen 11. System closes quests screen 12. User completes quest objectives (defeat enemies, trade goods, etc) 13. System removes quest from user, displays reward screen 14. User selects reward 15. System adds rewards to user inventory and closes reward screen 16. System displays quest screen
SubFlows:
Alternate/Exceptional Flows: 5a User selects “Decline” 6a System displays available quest list

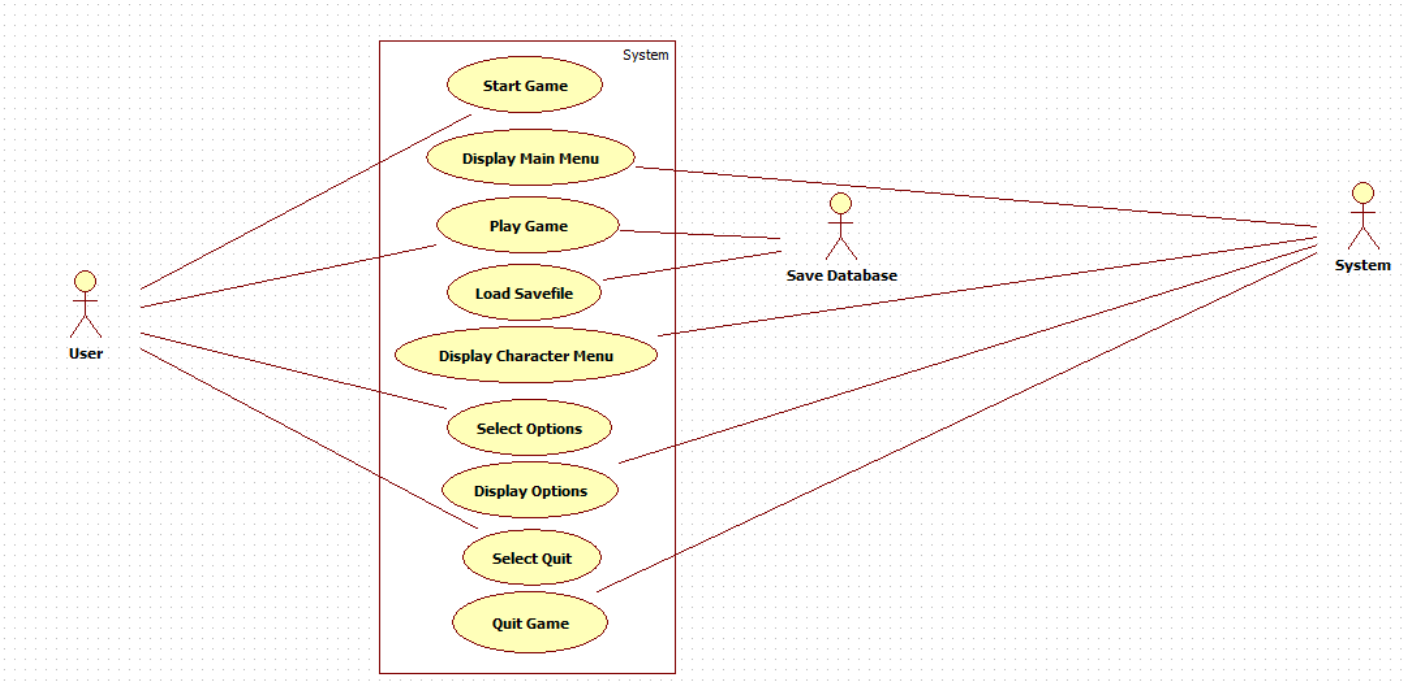
Save/Load Game

Use Case Name: Save/Load	ID: 11	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to be able to save their progress and continue it later		
Brief Description: This use case describes how a user saves and loads a game		
Trigger: user interacts with a quest-giver Type: Internal		
Relationships: Association: User, Host Application, Game Include: Extend: Generalization:		
Normal Flow of Events: 1. User selects “start game” from the main menu 2. System loads the current save file 3. User changes something in their inventory 4. System saves over the current save file 5. User Starts a travel scene 6. System saves over the current save file 7. User ends a travel scene		

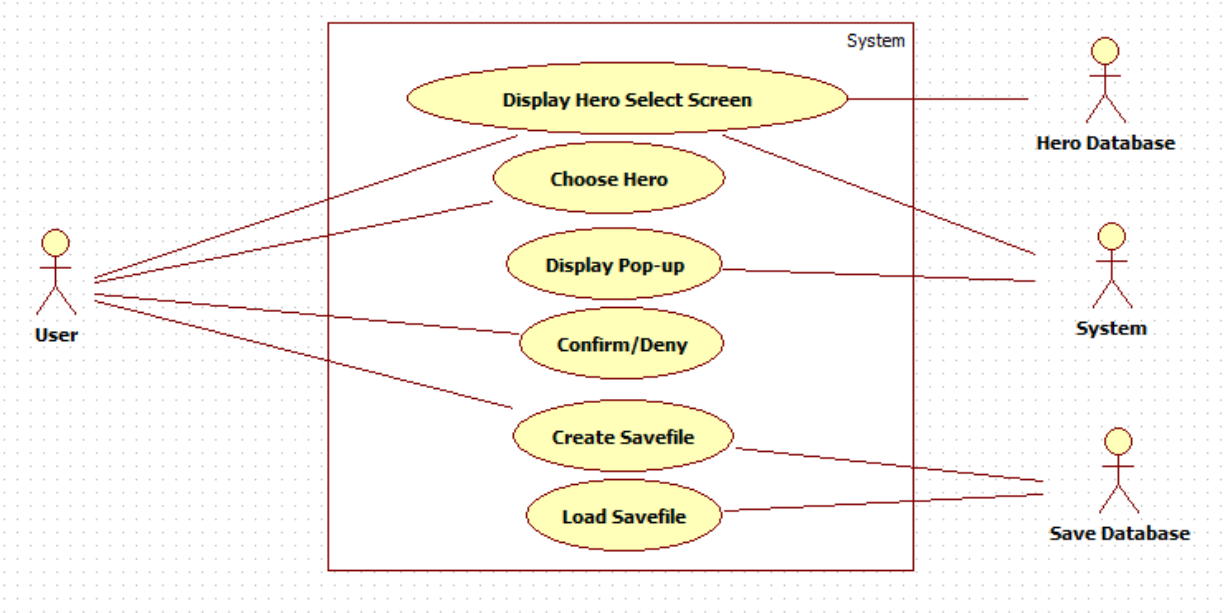
8. System saves over the current save file 9. User changes their party composition 10. System saves over the current save file 11. User begins or completes a quest 12. System saves over the current save file
SubFlows:
Alternate/Exceptional Flows:

System Use Case Diagrams

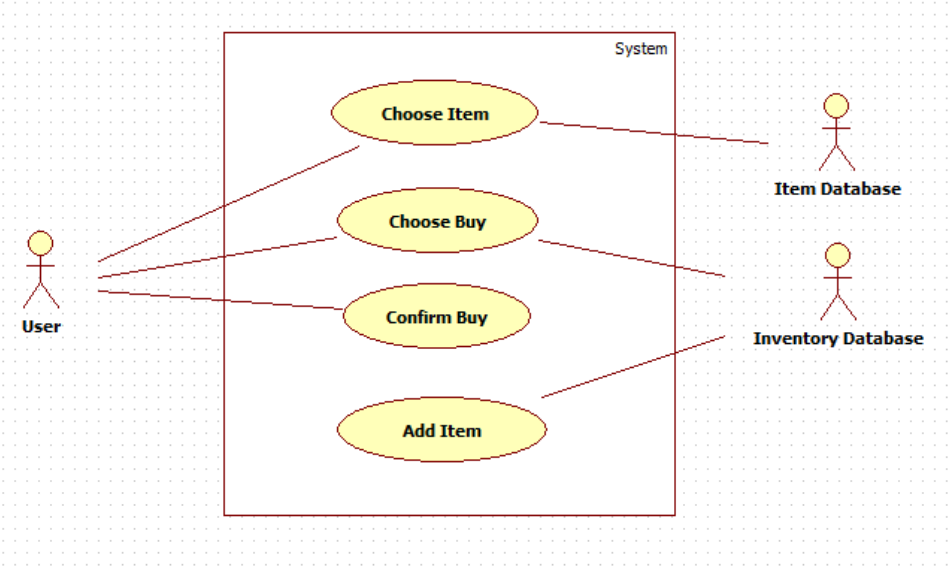
Start Menu



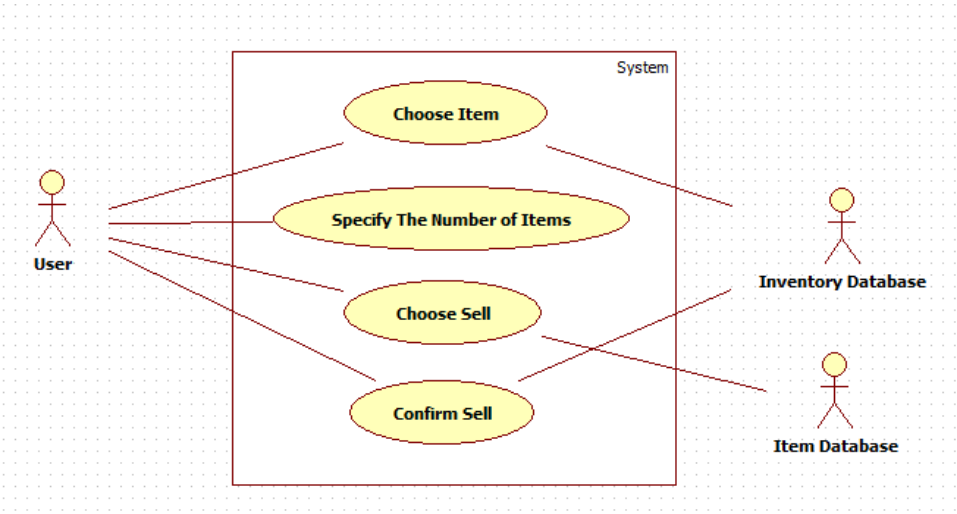
Select Hero



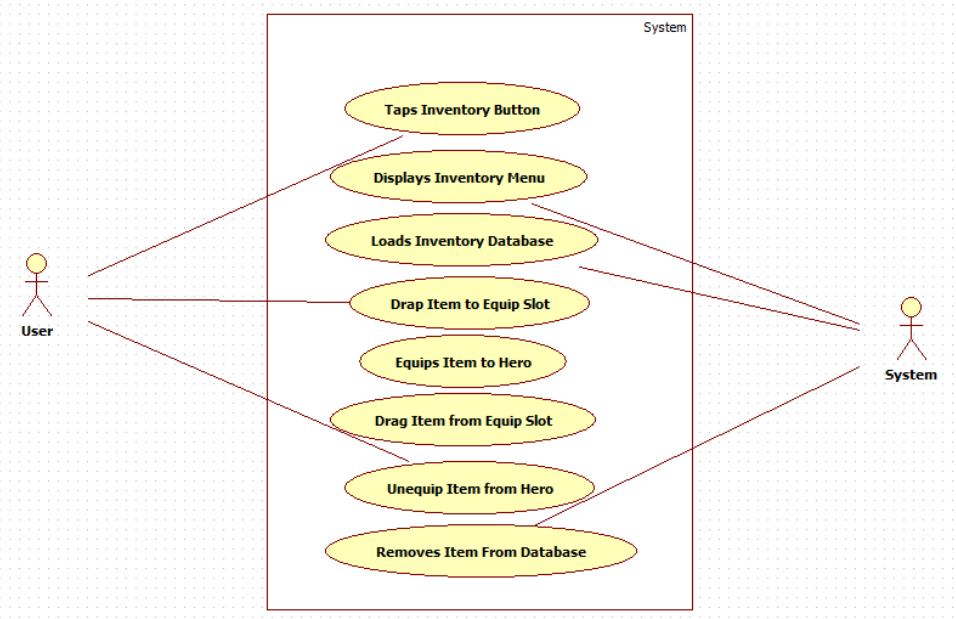
Buy Item



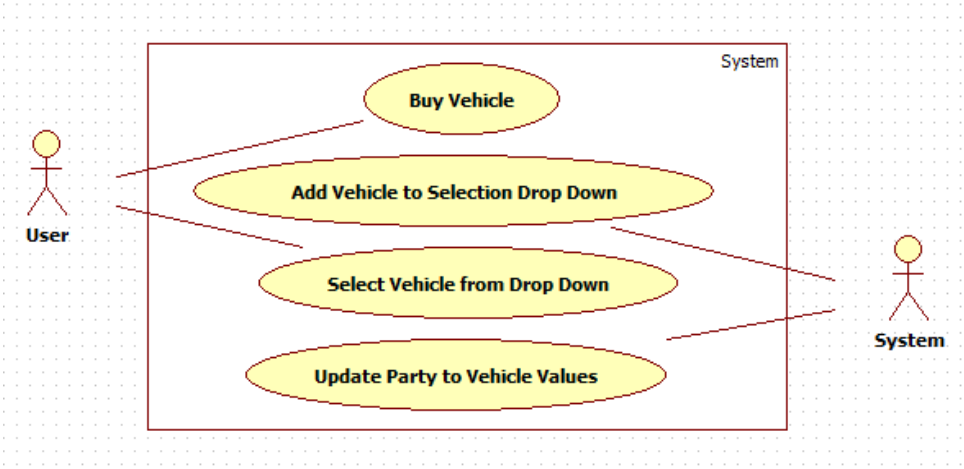
Sell Item



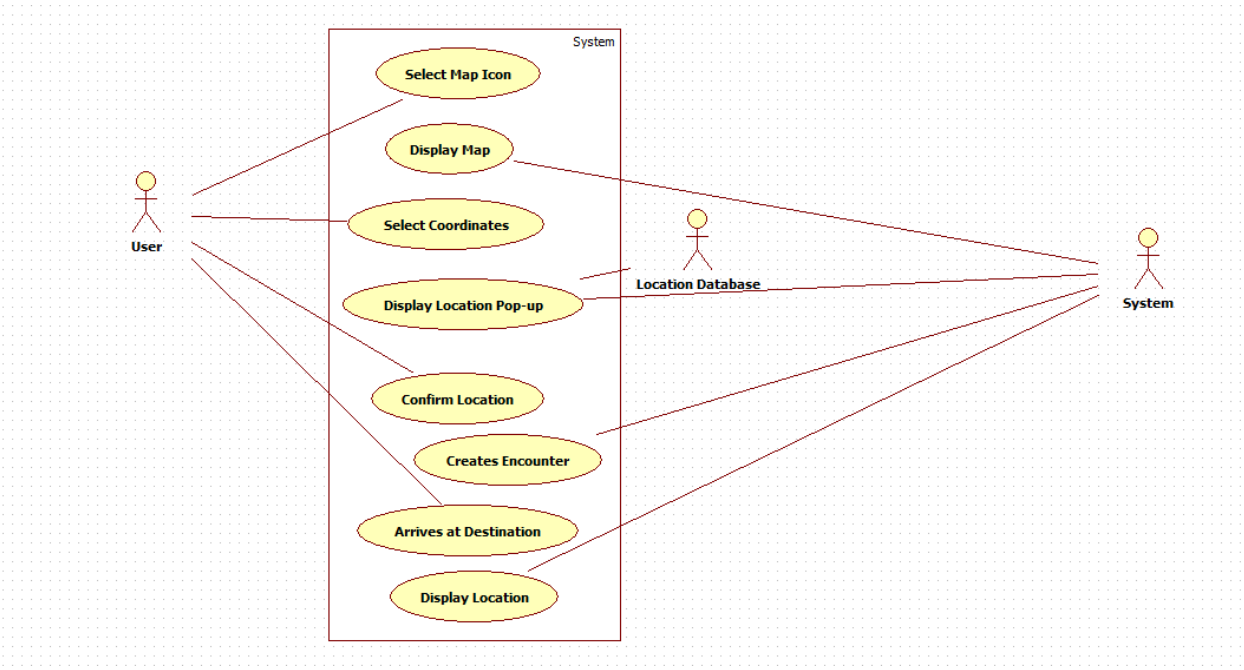
Inventory



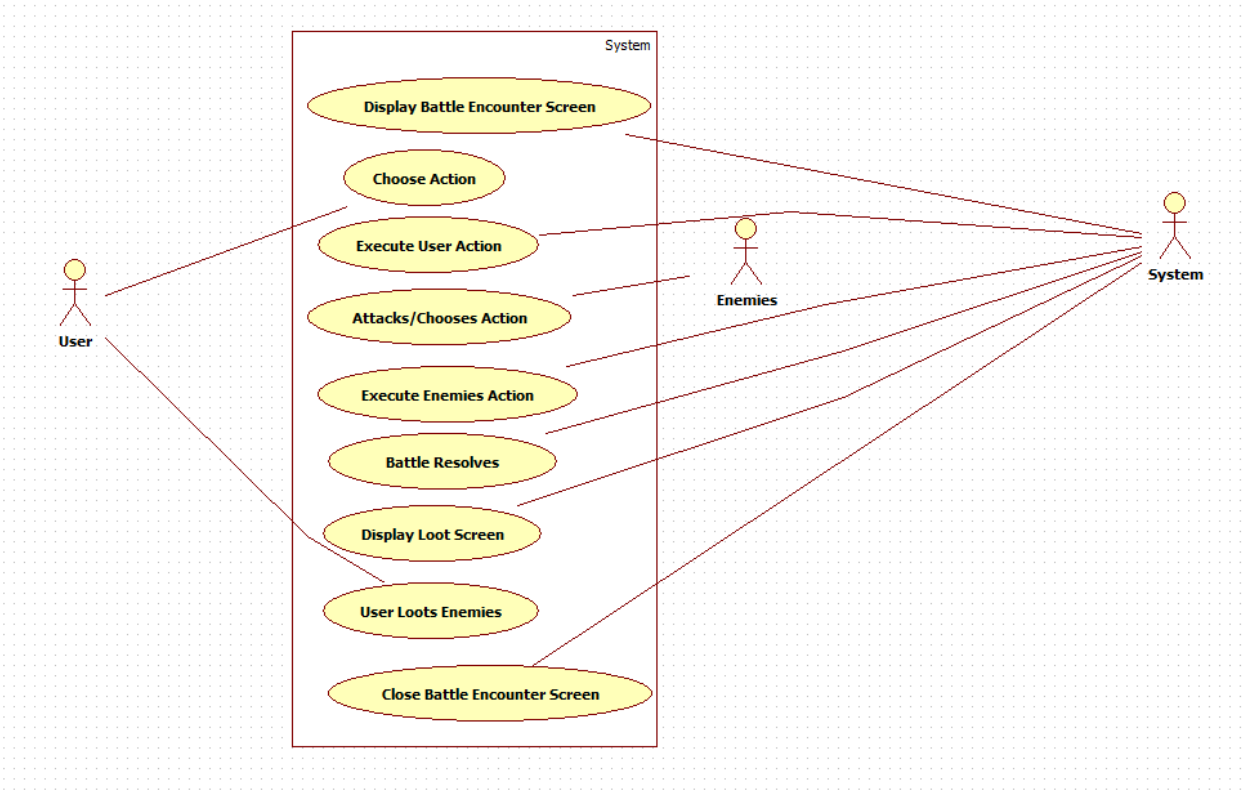
Vehicles



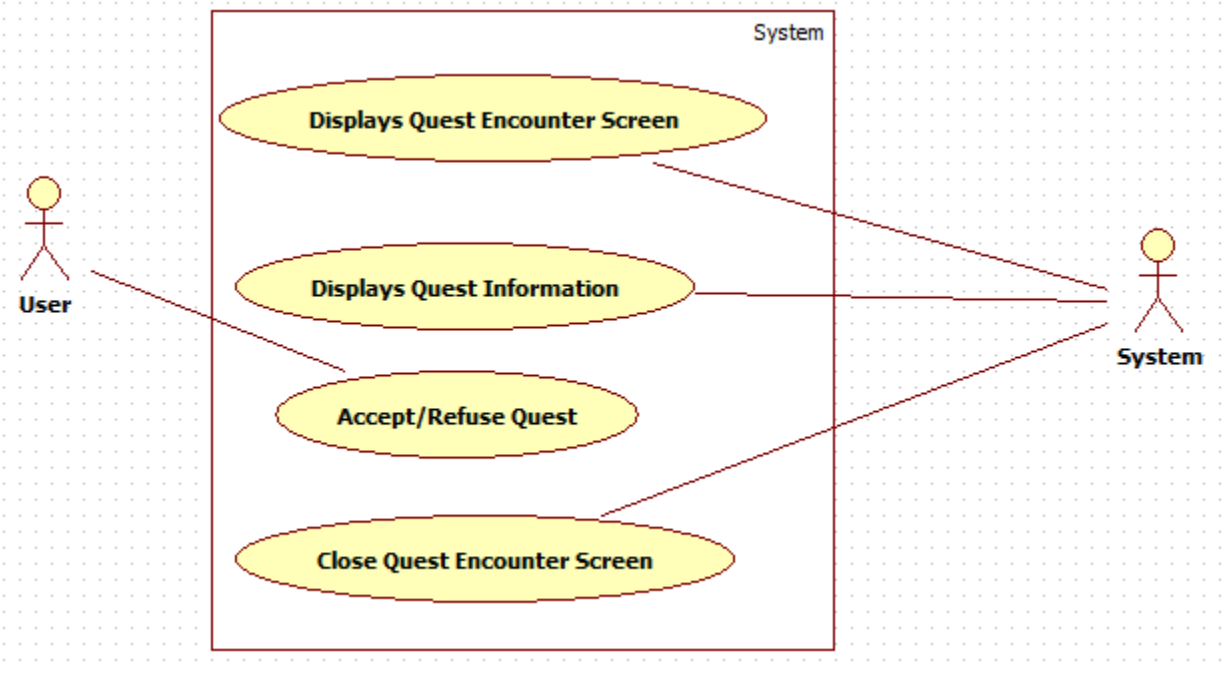
Location



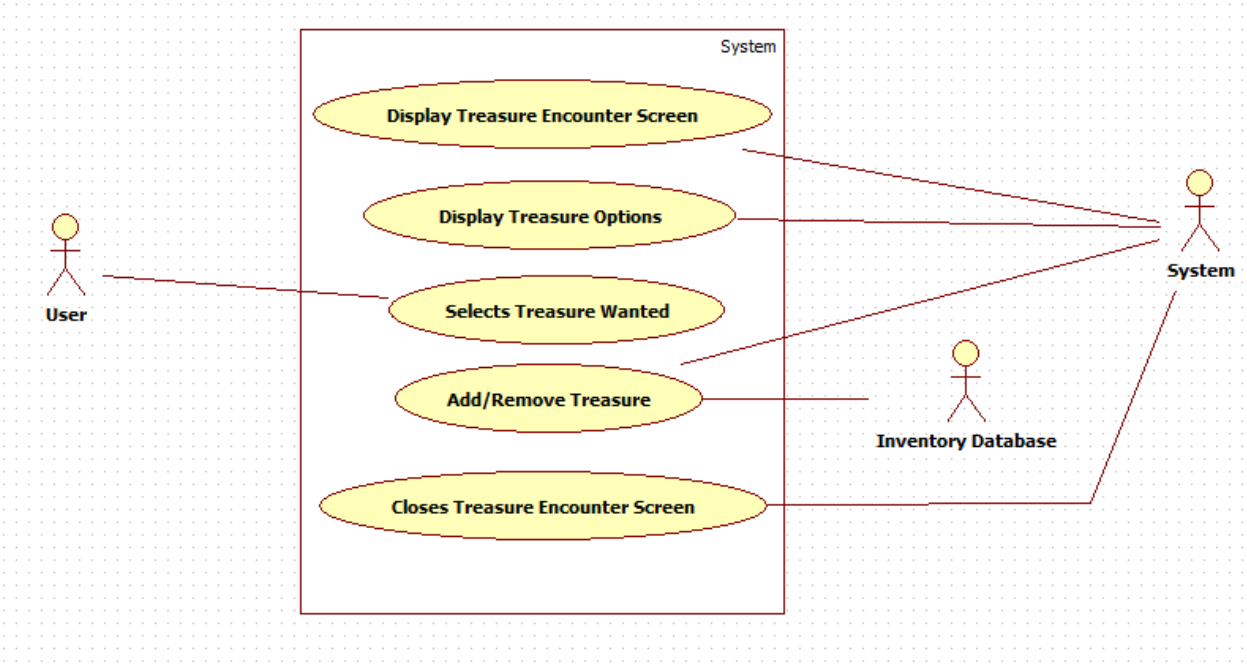
Battle Encounter



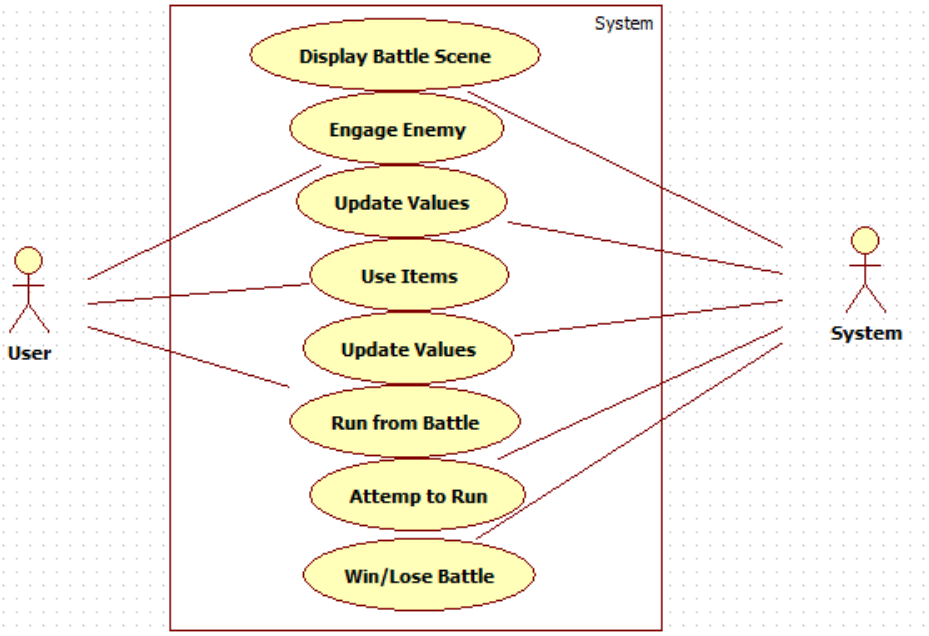
Quest Encounter



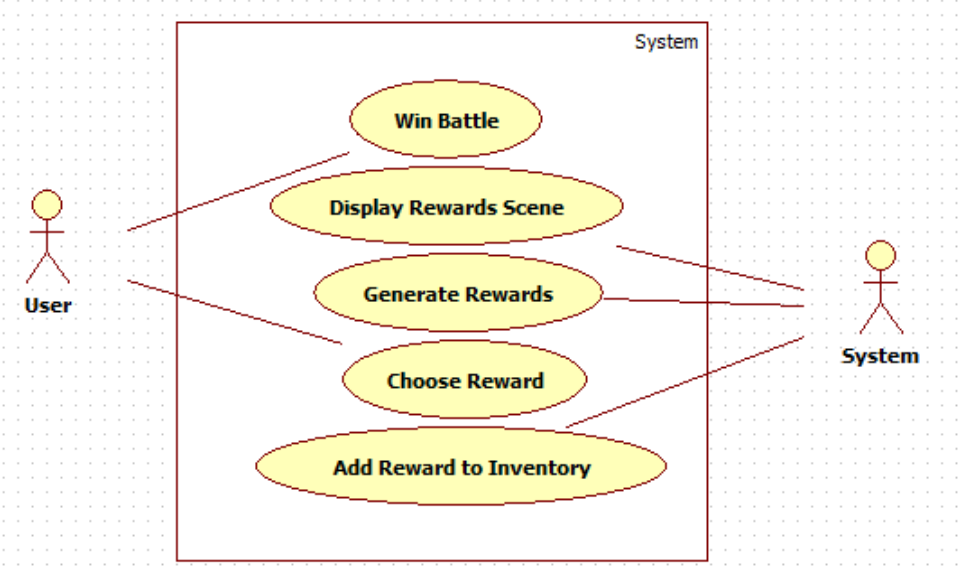
Treasure Encounter



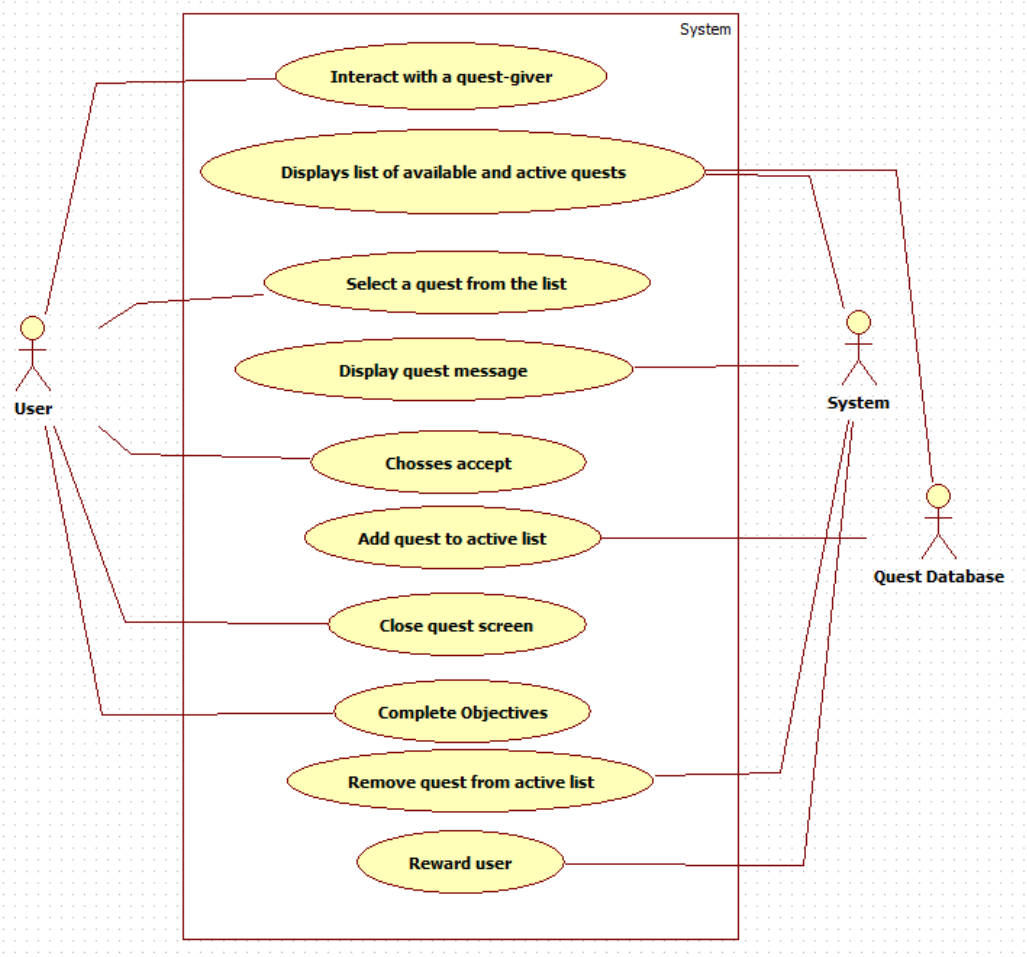
Battles



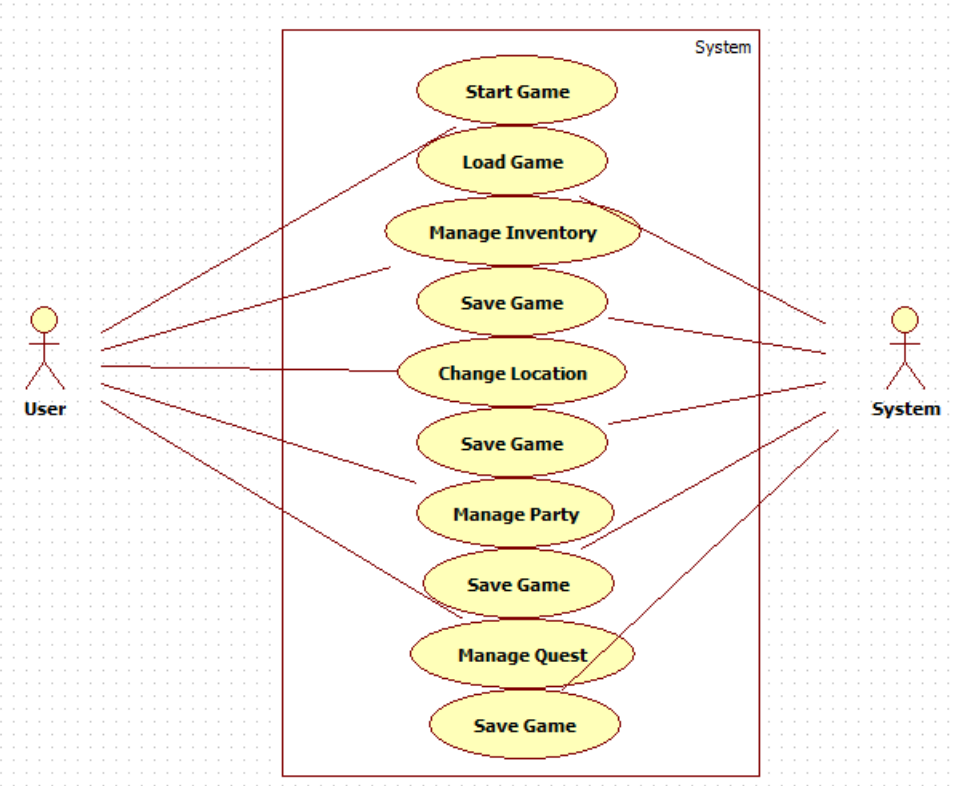
Rewards



Quests



Load/Save Game



Preliminary Conceptual Model of Use Case

CRD Cards

Launch Game Card

Class Name: User	**ID:** 1	**Type:** Concrete, Domain
Description: The user can choose to launch the game, select from the main menu to play a game or to quit	**Associated Use Cases:** 11	
Responsibilities	**Collaborators**	
Attributes Level (int) Inventory (UserInventory) Currency (double)		
Relationships Other Associations:		

Select Character

Class Name: Hero	**ID:** 2	**Type:** Concrete, Domain
Description: Heros are the characters the user can play as or recruit later	**Associated Use Cases:** 1	
Responsibilities Fight enemies Save the world	**Collaborators**	
Attributes Name (text) Capacity (double) Damage (double) Health (int) CritDamage (double) CritChance (double) Icon (Icon) Ability (Ability)		
Relationships		

Buy/Sell Cards

Class Name: User	**ID:** 3	**Type:** Concrete, Domain
Description: The user can choose to interact with the vendors to buy and sell items from their inventory	**Associated Use Cases:** 11	
Responsibilities Open Inventory Buy from vendors Sell to vendors	**Collaborators** UserInventory VendorInventory UserInventory	
Attributes Level (int) Inventory (UserInventory) Currency (double)		
Relationships Other Associations: UserInventory		

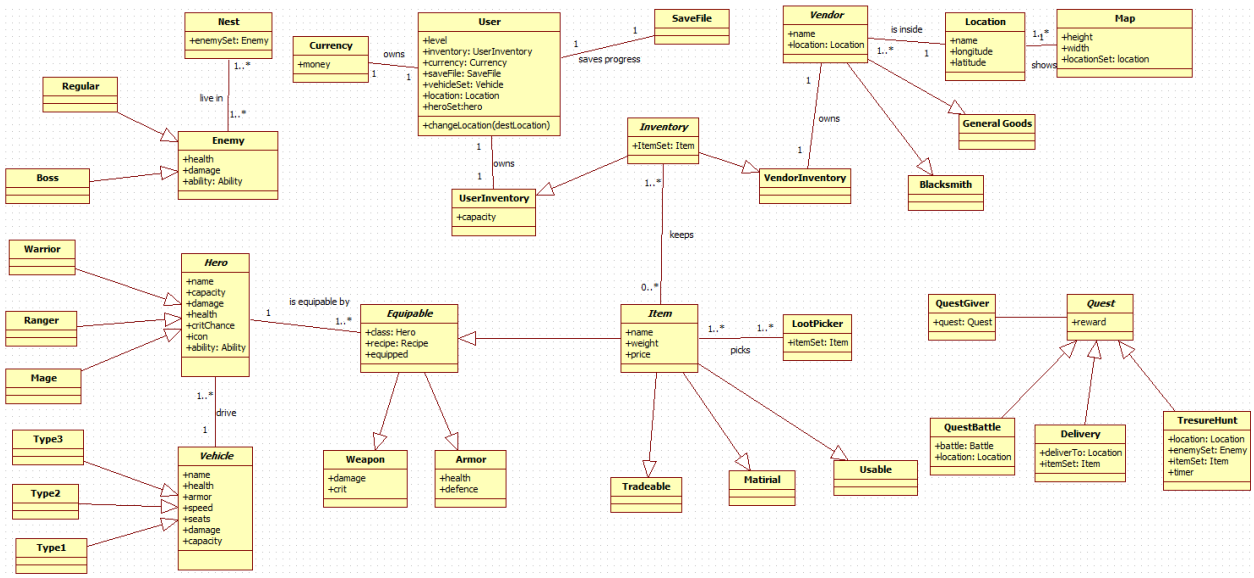
Class Name: UserInventory	**ID:** 4	**Type:** Concrete, Domain
Description: This holds all the items the user has.	**Associated Use Cases:** 2	
Responsibilities Increases in size when new items bought Decreases in size when items are sold Items in inventory can be accessed by user	**Collaborators** VendorInventory VendorInventory User	

Attributes Capacity (double)
Relationships Generalization (is a part-of) - Inventory Other Associations: User

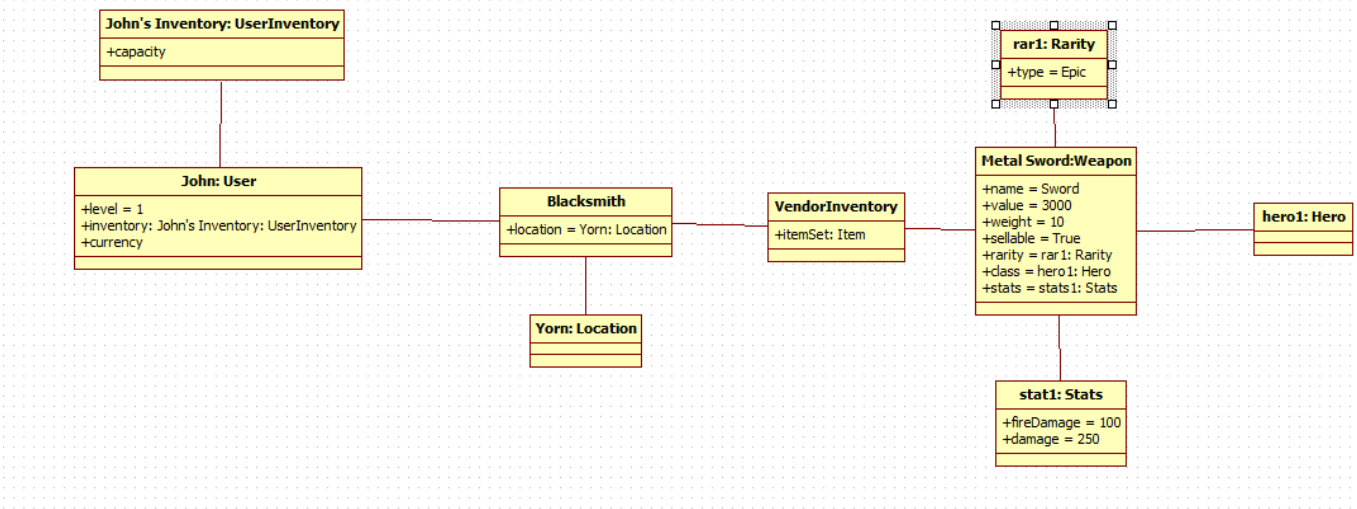
Class Name: Item	**ID:** 5	**Type:** Concrete, Domain
Description: An object that can go in the user's inventory	**Associated Use Cases:** 2	
Responsibilities Buy from vendors Sell to vendors Loot from battles	**Collaborators** VendorInventory VendorInventory Battle	
Attributes Name(text) Type (ItemType) Value (double) Capacity (double) Sellable (boolean) Rarity (Rarity)		
Relationships Generalization (has classes that are a kind-of relationship): Equipable, Tradeable, Cartable Composition: Rarity		

Class Name: Vendor	**ID:** 6	**Type:** Concrete, Domain
Description: Merchants who buy and sell items from the user	**Associated Use Cases:** 2	
Responsibilities Buy Items from players Sell Items to players	**Collaborators** UserInventory VendorInventory	
Attributes Name (text) Location (Location)		
Relationships Other association: Location		

Class Diagram



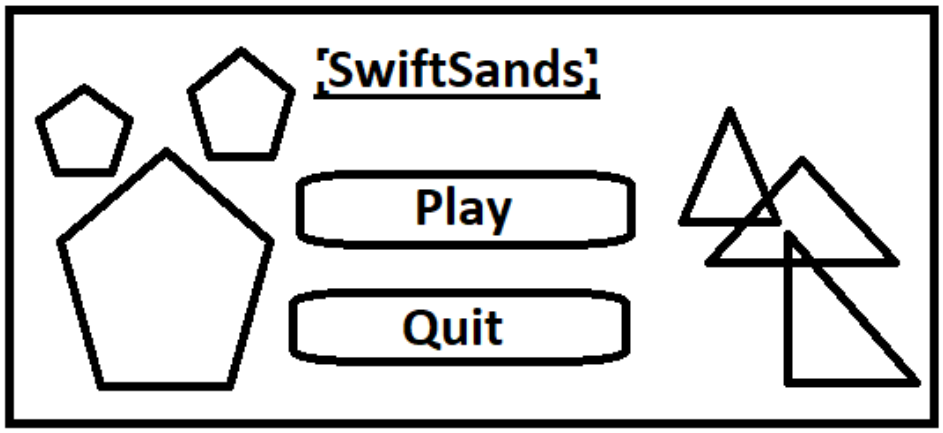
Buy/Sell Object Diagram



Use Case Specifications + Interface Mock-ups

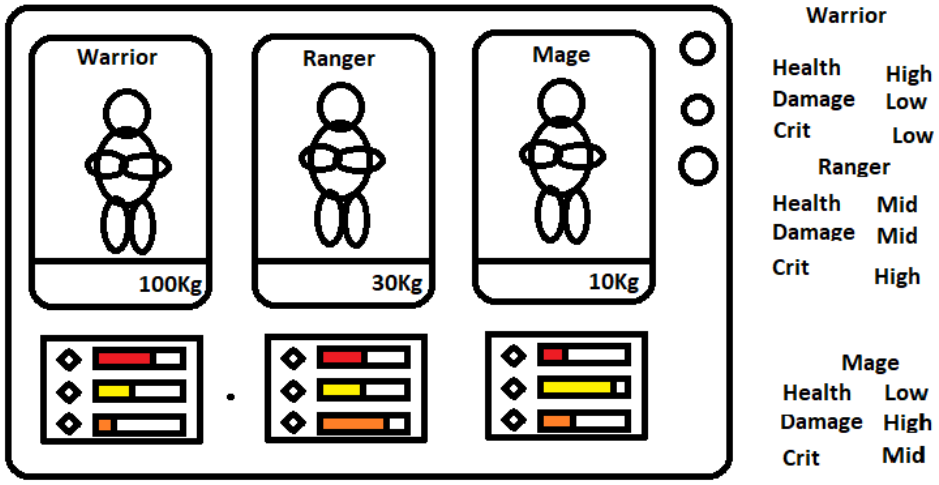
Use Case: Main Menu

The user can start the game by pressing the Play button or Quit the game pressing the Quit button.



Use Case: Character Selection

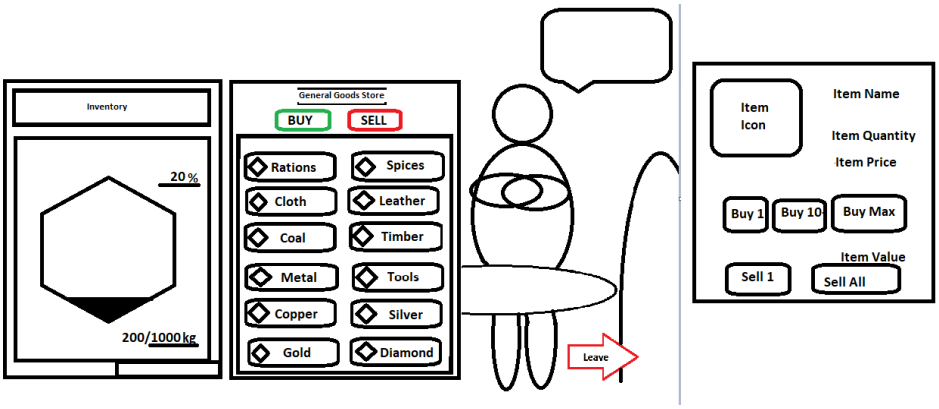
User get to choose the first Hero that will be joining his/her party pressing on a hero's icon pops up a window that confirms selection which the user can accept or reject.



Use Case: Buy Item

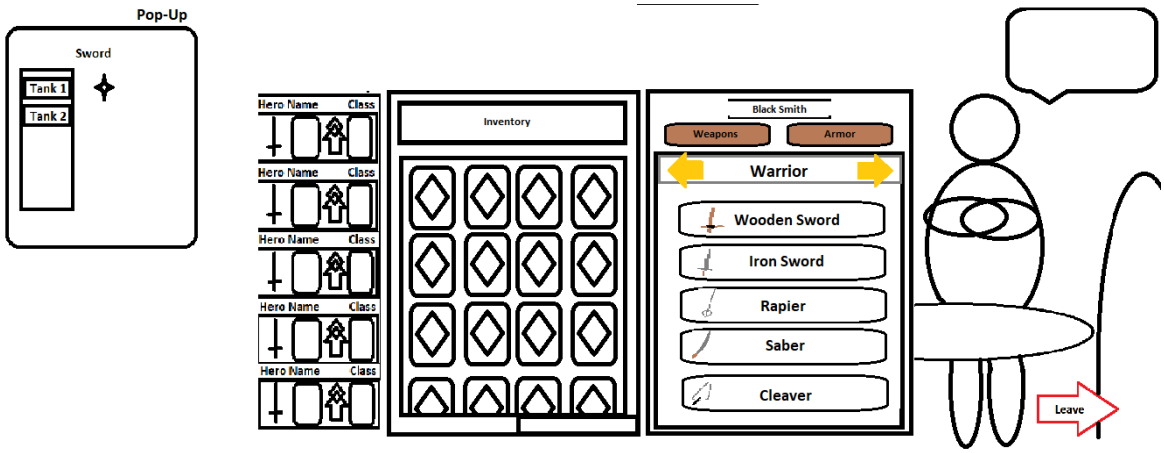
The user taps on one of the several buildings in town, that he or she wants to visit. The user than gets moved inside the store of choice with a merchant saying something. There is a Shop Menu already displaying, with everything the vendor sells. (Default to Buy) There is a Green

“Buy” Button and a Red “Sell” Button to let you switch between your inventory and the vendor’s inventory.



The user can select an item to bring up a small screen in the center of the screen that will display the details of the item. The items available have the following stats: Name, Type, Class, Rarity, and Price. This screen will show the icon of the item on the left side, display a detailed list of the item’s stats on the right, and the bottom will have a button that says, “Buy”. If the user wants to buy the item, they can press the button, if they want to check another item, they can close the small box by clicking on the “X” at the top-right corner or maybe swipe the pop up away.

When the user is done, there is a RED “Leave” Arrow Button beside the merchant that will let you leave the store.



Use Case: Sell Item

1. The user taps on one of the several buildings in town, that he or she wants to visit. The user than gets moved inside the store of choice with a merchant saying something. There is a Shop Menu already displaying.

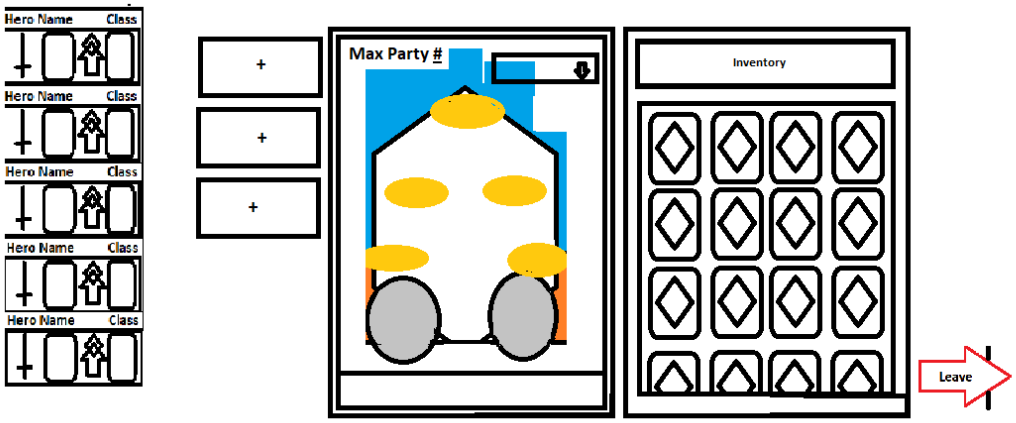
The user clicks on “Sell”, which makes the system close the Buy Item Screen and opens the Sell Item Screen. The Sell Item screen is an image of the player’s inventory. This screen looks

like a rectangle that holds smaller black squares, each of which is a card that represents something stored in the user's inventory. When the user clicks on an item a small pop-up appears next to the card showing its properties and how much money the user would get in return for selling the item. If the user wants to sell the item, they can click on the items "sell" button. The item will disappear from their inventory and their money will go up. If the user wants to close the cards details screen, they can press the "X" on the top-right corner of the card or click away or on a different card.

Use Case: Inventory/Party

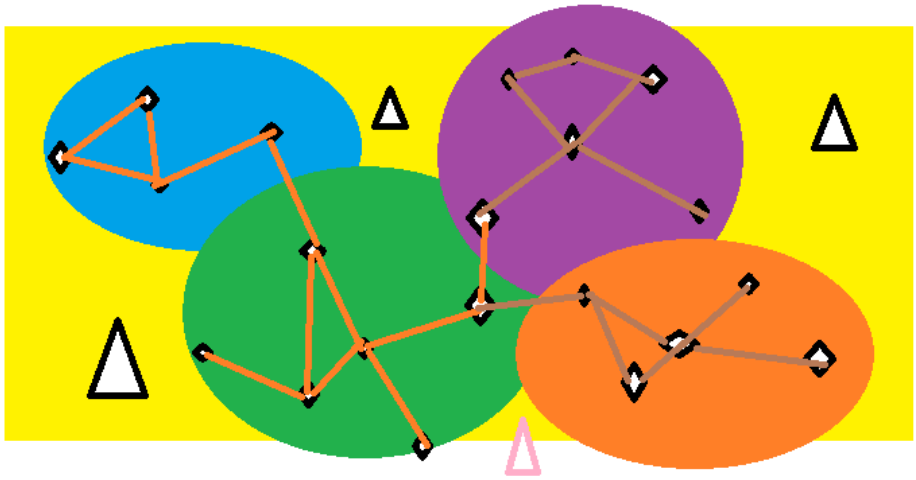
In the Inventory the player can manage their cargo, choose their vehicle which sets the party size and control the amount of hero slots available which you must reselect each time you change a vehicle. The game doesn't let you start unless you have ATLEAST 1 hero selected.

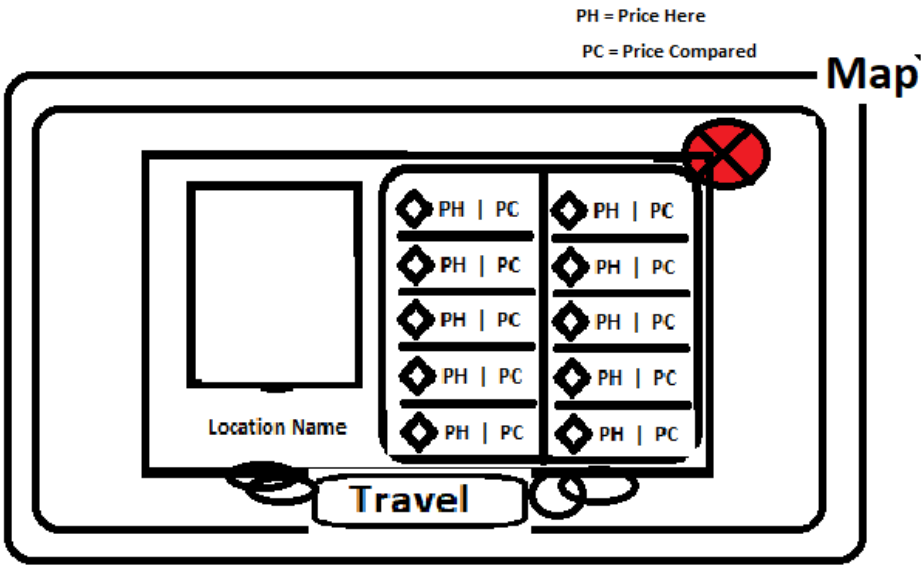
Here you can drop cargo, equip items, unequip items, and use items.



Use Case: Locations

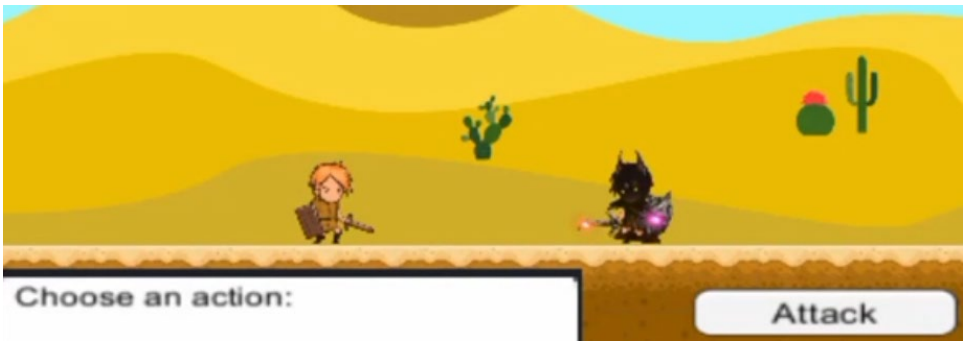
When selecting a new location to travel to, you press on a location points in the map, which displays a window with that locations name and the price comparison of goods in the location compared to your own. This is a stretch goal that might end up the name and icon of the location. Pressing the "Travel" button confirms the desire to change locations.





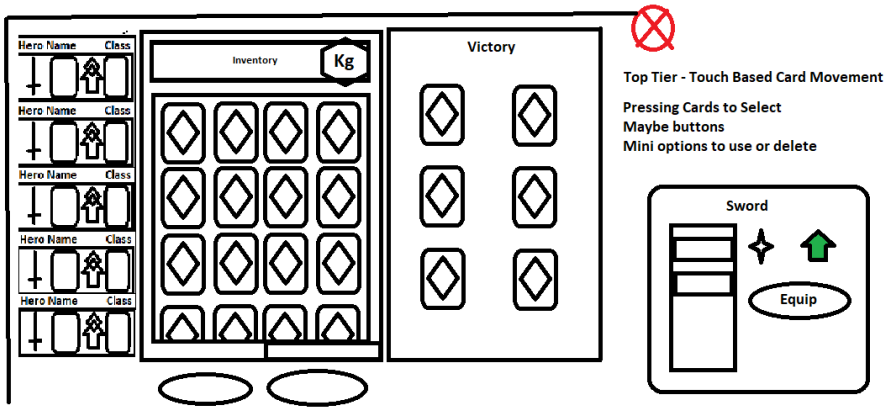
Use Case: Battles

When a user is put into a battle encounter or a quest battle, the system will display the battle screen. The battle screen will show the user's party members and all the enemies they are fighting arrayed for battle. From this screen the user can choose to have their party-members attack the enemy, use an item or run away. After the user has made their choice of action, their turn ends. The system enacts their choice and the enemy's attack. After the enemies do their action it is the user's turn again. The user may attempt to run away, if they succeed the battle ends and the battle screen closes, if they fail, they waste their turn and their enemy's attack. The battle lasts so long as the heroes and enemy's health remain above zero. Once either the heroes or enemy's health reaches zero, or the heroes successfully run away. the battle is over, and the battle screen closes.



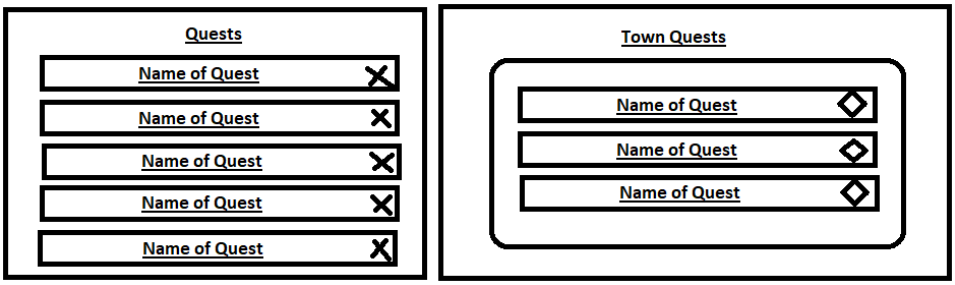
Use Case: Rewards

When the user wins a battle or completes a quest the system will present the user with the rewards screen. This screen will show the user the items that their victory has won them and allow them to take items from the screen and put them in their inventory.



Use Case: Quests

User can have up to five quest at any time and every town carries up to three randomly generated quests from a pool of quantities, trade goods and destinations/locations. User can Drop a quest at any time by clicking on the Icon to the right, a confirmation window pops up. Accepting Quests is as easy as pressing the icon on the right of the Town Quests board quests, a confirmation window pops up.



Data Dictionary

Weapon

Attribute Name	Required	Type	Field Length	Default Values	Notes
WeaponName	Yes	Text	20	n/a	Primary Key
WeaponWeight	Yes	Double	3	n/a	
WeaponValue	Yes	Double	6	n/a	
WeaponClass	No	Text	15	n/a	Warrior, Mage, or Ranger
WeaponDamage	Yes	Integer	6		
WeaponCrit	No	Double	5		
RecipeID	Yes	Integer	3		Foreign Key

Armour

Attribute Name	Required	Type	Field Length	Default Value:	Notes
ArmorName	Yes	Text	20	n/a	Primary Key
ArmorWeight	Yes	Double	3	n/a	
ArmorValue	Yes	Double	6	n/a	
ArmorClass	No	Text	15	n/a	Warrior, Mage, or Ranger
ArmorHealth	Yes	Integer	6		
ArmorDefence	No	Integer	6		
RecipeID	Yes	Integer	3		Foreign Key

Hero

Attribute Name	Required	Type	Field Length	Default Value:	Notes
HeroClass	Yes	Text	10	n/a	
HeroName	Yes	Text	20	n/a	Primary Key
HeroBaseHealth	Yes	Int	6	100	Hero Class can add to it
HeroBaseDamage	Yes	Int	4	20	Hero Class can add to it
HeroBaseCrit	Yes	Double	3	15	Hero Class can add to it

Enemy

Attribute Name	Required	Type	Field Length	Default Values	Notes
EnemyName	Yes	Text	20	N/A	Primary Key
EnemyType	Yes	Text	20	N/A	
EnemyBaseHealth	Yes	Int	6	100	
EnemyBaseDamage	Yes	Int	4	20	

Tradeable

Attribute Name	Required	Type	Field Length	Default Values	Notes
TradeableName	Yes	Text	15	n/a	Primary Key
TradeableWeight	Yes	Double	3	n/a	
TradeableValue	Yes	Double	6	n/a	

Usable

Attribute Name	Required	Type	Field Length	Default Values	Notes
ItemName	Yes	Text	15	n/a	Primary Key
ItemWeight	Yes	Double	3	n/a	
ItemValue	Yes	Double	6	n/a	
EffectAmount	Yes	Double	3	n/a	

Quest

Attribute Name	Required	Type	Field Leng	Default Values	Notes
QuestID	Yes	Int	5	n/a	PrimaryKey
QuestName	Yes	Text	20	n/a	
QuestType	Yes	Text	20	Delivery	Treasure Hunt, Delivery, Battle
QuestStartLocation	Yes	Text	20	n/a	ForeignKey
QuestEndLocation	Yes	Text	20	n/a	ForeignKey
QuestReward	No	Text	20	Gold	ForeignKey

Location

Attribute Name	Required	Type	Field Length	Default Values	Notes
LocationName	Yes	Text	20	n/a	
LocationCoordinates	Yes	Double	4	n/a	Primary Key

Party

Attribute Name	Required	Type	Field Length	Default Values	Notes
HeroID	Yes	Int	2	n/a	Primary Key

Vehicle

Attribute Name	Required	Type	Field Length	Default Values	Notes
VehicleName	Yes	Text	25	n/a	Primary Key
VehicleHealth	Yes	Int	6	0	
VehicleArmor	Yes	Int	6	0	
VehicleSpeed	Yes	Double	4	0	
VehicleSeats	Yes	Int	1	0	
VehicleDamage	Yes	Int	6	0	
VehicleCapacity	Yes	Double	5	0	

DATABASE Choice and Justification

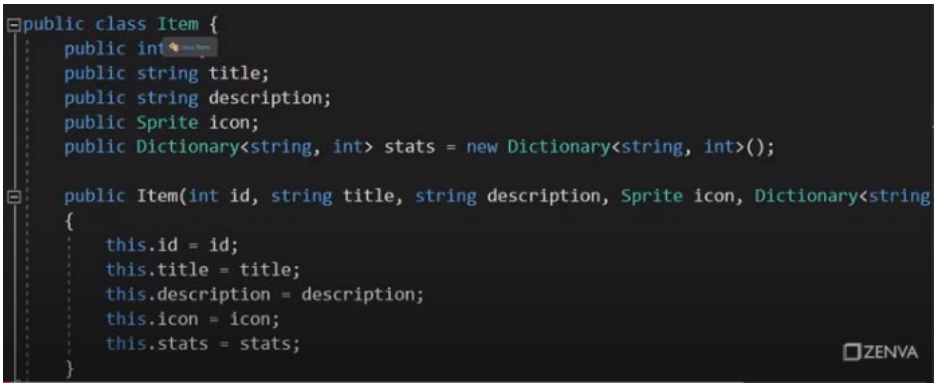
For our game app we chose to use Lists of Databases as no new Data is added by the users, so there is no use for a large-scale online database.

Lists in Unity Games are used by many programmers and so there are a lot of tutorials available on building item class database structures like the ones that we will be using.

Additionally, Recipes for crafting new items can be stored in such a manner as well, which is a late game feature of ours.

The following are examples of a structure used by a game tutorial building an example database:

https://www.youtube.com/watch?v=S-XR37KM7_o&list=PLMp2peNEblP1IchkRi-26k1HwdNyz2mdp&index=37&ab_channel=GameDevHQ



```
void BuildItemDatabase()
{
    items = new List<Item>()
    {
        new Item(1, "Diamond Sword", "A sword made of diamond.",
        new Dictionary<string, int> {
            { "Power", 15 },
            { "Defence", 7 }
        }),
        new Item(2, "Diamond Ore", "A shiny diamond.",
        new Dictionary<string, int> {
            { "Power", 15 },
            { "Defence", 7 }
        })
    };
}
```

Based on this choice we have built this preliminary DB design.

Database Tables

WeaponID	WeaponName	WeaponWeight	WeaponValue	WeaponClass	WeaponDamage	WeaponCrit
1	Iron Sword	10	20	Warrior	10	10
2	Steel Sword	10	40	Warrior	20	10
3	Damascus Steel Sword	10	80	Warrior	30	10
4	Sky Steel Sword	10	160	Warrior	40	10
5	Ashen Remeberance	25	500	Warrior	80	20
6	Hunting Bow	5	20	Ranger	5	20
7	Longbow	5	40	Ranger	10	20
8	CrossBow	5	80	Ranger	15	20
9	Compound Bow	5	160	Ranger	20	20
10	Desert's Call	10	500	Ranger	50	33
11	Mage's Orb	1	10	Wizard	15	0
12	Elm Wand	1	20	Wizard	10	5
13	Elm Staff	5	40	Wizard	20	15
14	Oak Wand	1	80	Wizard	15	5
15	Obsidian Staff	5	160	Wizard	25	15
16	Magnum Opus	10	500	Wizard	60	25

ArmourID	ArmourName	ArmourWeight	ArmourValue	ArmourClass	ArmourHealth	ArmourDefense
1	Chainmail Vest	15	30	Warrior	10	10
2	Iron Gamberson	20	50	Warrior	20	15
3	Steel Gamberson	25	70	Warrior	30	20
4	Full Plate Armour	40	200	Warrior	40	25
5	Mobile Fortress	50	500	Warrior	80	33
6	Leather Armour	10	20	Ranger	5	1
7	Hardened Leather Armour	10	40	Ranger	10	5
8	Leather Cuirass	10	50	Ranger	20	10
9	Chitin Armour	15	80	Ranger	30	15
10	Vestments of the Unseen	20	500	Ranger	40	20
11	Torn Doublet	1	10	Wizard	1	1
12	Hermit's Old Robes	5	20	Wizard	5	1
13	Cloth Robes	5	50	Wizard	10	1
14	Voluminous Robes	5	80	Wizard	15	5
15	Drapes of the Profligate Seer	10	500	Wizard	20	10

Project Quality Management Plan

Planning Quality Management

- Game should keep system load times short
- Game is single-player, system will handle one user at time
- Game should allow users to access all Functional Requirements (p. 8)
- Game may allow users to access non-mandatory goals (p. 9) if there is additional time
- Game must run on computer

Performing Quality Assurance

- Team will perform many tests on the game
- Glitches that are found in testing will be isolated and fixed
- All team members will be developers and Quality Assurance
- Team will update schedule as required

Control Quality Assurance

- Team will create an online forum for user feedback
- User feedback will be gathered to identify strengths and weaknesses of game
- Feedback will help team develop features users might be interested in
- Feedback will help identify glitches that team will work on fixing
-

Project Communications Management Plan

Plan Communications Management

- Provide Deliverables and Work Division on time to our Professor Stakeholder
- Seneca's Administration will receive the final submission as a Stakeholder
- Keep the Team up to date on everything that goes on with the project as the Main Stakeholders

Manage Communications

- All Deliverables will be submitted by the Team either by email or by submission link as specified by the Professor Stakeholder
- Seneca's Administration will receive the final submission by submission link at the end of the final term as a Stakeholder
- The team will communicate via: Discord, WhatsApp, Email, and Telegram.

Monitoring and Controlling Communications

- Keeping our weekly Deliverables done and handed in on time for the Professor Stakeholder
- Having a finished working final submission on time for the Seneca Administration Stakeholder
- The Teams Communication will keep updating on all needs of the project for Data Transfer, work and home communication, and several project tracking software such as MS Teams, Click Up, and MS Projects. Meetings required for project communication.

Cost Estimates

Our costs will be relatively low. This is because, as a student project for Seneca College, we are not being paid for the work. In addition to that, we are using our own hardware so the cost for computers does not factor in. Most of the Software we are using is free, but we will have to use Spine, which costs money. Our main expenses come from Spine and from paying graphics designers to create the visuals.

	# Units/Hrs.	Cost/Unit/Hr.	Subtotals	WBS Level 2 Totals	% of Total
WBS Items					
1. Project Management				**$600**	**74.72%**
Project Manager	440	$0	$0		
Project Team Members	440	$0	$0		
Constractors(Artists)	20	$30	$600		
2. Hardware				**$0**	**0%**
3. Software				**$69**	**8.59%**
3.1 Licensed Software	1	$69	$69		
3.2 Software Development			$0		
4. Testing			$0	**$0**	**0%**
Project Team Members			$0		
5. Training and Support				**$0**	**0%**
Project Team Members	440	$0	$0		
Subtotal			$669		
6. Reserves (20% of total estimate)			$133.8	**$133.8**	**16.66%**
Total Project Cost Estimate				$803	

Risk Management Plan

Project	Video Game	**Project #**	000-x-001
Project Manager	Andriy Ostapovych	**Sponsor**	Seneca College
Project Artifacts		**Updated**	

ID	Risk Description	Probability	Impact	Detectability	Importance	Category	Trigger Event/Indicator	Risk Response and Description	Contingency Plan	Owner	Status	Date Entered	Date to Review
1	We have never done a project like this before and may run out of time before we finish all of the features before the project is due.					Time	When multiple tasks go over the slotted time for completion in succession.	We will create and use a Gantt chart to keep our work on schedule	We will realign the schedule and work overtime to get overdue tasks done	Developers		2020-11-24	
2	With the current pandemic, one or more of us can get a bad case of Covid-19. This would be bad for many reasons, and slow the project down considerably					Plague	When one or more of us start to feel badly sick.	We will obey masks and social distancing rules.	Developers who are not sick will have to work extra hard.	Developers		2020-11-24	
3	We could hit a wall in programming this game where there is something that we just do not know how to do a language we are not used to.					Time	When we all encounter an issue that even when working together we cannot solve.	We shall try and learn as much of each language we are working with as possible.	We will look up issues we encounter online, discuss with peers and if that does not work find someone at Seneca.	Developers		2020-11-24	
4	Data corruption could cost us all of our work or all of our incremental work.					Data Corruption	When we are unable to access our previous work.	We will each backup our individual work on our local devices and use an online server to keep the work on.	If we lose all our work we will simply have to do crunch time to make it all again.	Developers		2020-11-24	

Cost Benefit Analysis

Development Cost from Cost Estimate

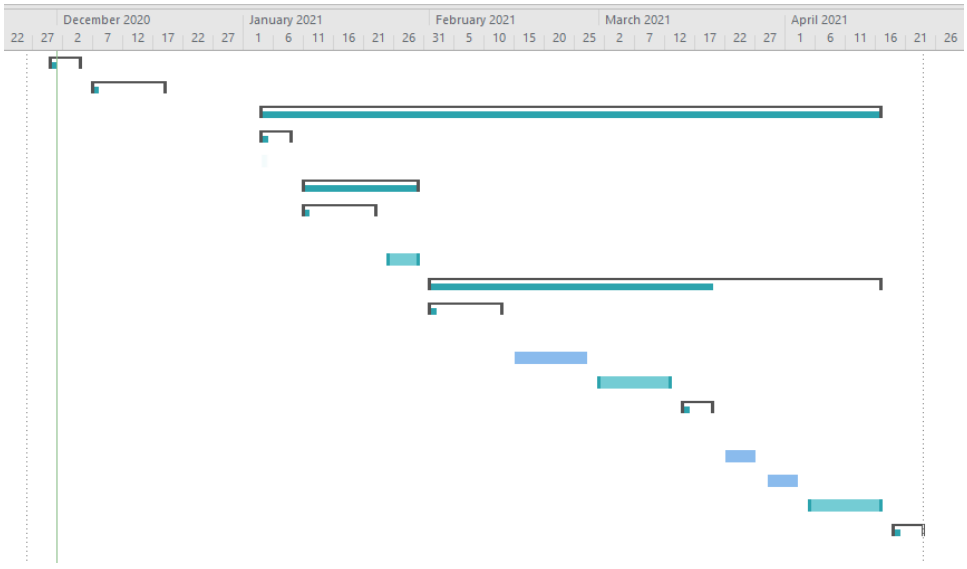
Operating Costs Include: Advertising, and Supplies with no further expenses post development

The Tangible Benefits equal to the sales of the app which in Game Application history Games rise in sales and if no further updates are introduced the sales level out and then drop significantly.

Costs	Year						
	0	1	2	3	4	5	6
Development costs	-803						
Operating Costs		-200	-200	-200	-200	-200	
Total Costs	-803	-200	-200	-200	-200	-200	
Discount Factor (Discount rate = 15% p.a.)	1	1	1	1	1	1	
Present Value of Costs	-803	-200	-200	-200	-200	-200	
Cumulative PV Costs	-803	-1003	-1283	-1483	-1683	-1883	-1883
Benefits							
Tangible Benefits from new System		1200	3300	5500	4000	2300	16300
Intangible Benfits from new System							
Total Benefits		1200	3300	5500	4000	2300	
Discount Factor (Discount rate = 15% p.a.)	1	1	1	1	1	1	1
Present Value of Benefits		1200	3300	5500	4000	2300	1500
Cumulative PV Benefits		1200	4500	10000	14000	16300	17800
Cumulative PV Benefits + Costs	-803	197	3217	8517	12317	14417	15917

PRJ666 Schedule

Task Mode	Task Name	Duration	Start	Finish
	Creating Classes	**5 days**	**Mon 11/30/2**	**Fri 12/4/20**
	Creating Databases	**10 days**	**Mon 12/7/2(**	**Fri 12/18/20**
	Creating UI	**75 days**	**Mon 1/4/21**	**Fri 4/16/21**
	Menus	**5 days**	**Mon 1/4/21**	**Fri 1/8/21**
	Main Menu	1 day		
	Locations on Map	**15 days**	**Mon 1/11/21**	**Fri 1/29/21**
	Villages	**10 days**	**Mon 1/11/21**	**Fri 1/22/21**
	Shops			
	Nests	5 days	Mon 1/25/21	Fri 1/29/21
	Interactions	**55 days**	**Mon 2/1/21**	**Fri 4/16/21**
	Shops	**10 days**	**Mon 2/1/21**	**Fri 2/12/21**
	Trading			
	Encounters	10 days	Mon 2/15/21	Fri 2/26/21
	Quests	10 days	Mon 3/1/21	Fri 3/12/21
	Battles	**5 days**	**Mon 3/15/21**	**Fri 3/19/21**
	Battle result Screen			
	Reward Screen	5 days	Mon 3/22/21	Fri 3/26/21
	Player Inventory	5 days	Mon 3/29/21	Fri 4/2/21
	Vehicles	10 days	Mon 4/5/21	Fri 4/16/21
	Load/Save Game	**5 days**	**Mon 4/19/21**	**Fri 4/23/21**
	Save File			
	Load Game			



Work Division

- **Robert Parker-Lak**

  Industry Analysis

  The Stakeholders

  Stakeholder Analysis

  System Request

  Functional & Non - Functional Requirements

  SWOT

  Use Case Descriptions

  Preliminary Conceptual Model of the System Use Case

  Use Case Specifications and corresponding interface mock-ups

  Interface Mock-ups

  Data Dictionary

  ERD or JSON model

  Project Quality Management Plan

  Project Communications Management Plan

  Risk Management Plan

  Cost Benefit Analysis

- **Andriy Ostapovych**

  Problem Statement

  Constraints/Business Rules

  Functional & Non - Functional Requirements

  SWOT

  Business Use Case Diagrams

  DFDs

  System Use Case Diagrams

Domain Class Diagram

Data Dictionary

ERD or JSON model

Project Quality Management Plan

Project Communications Management Plan

Project Cost Estimate

Cost Benefit Analysis

- **Faramarz Hosseini**

  Feasibility Report

  PRJ566 Schedule

  Stakeholder Analysis

  Functional & Non - Functional Requirements

  SWOT

  Activity Diagram

  DFDs

  System Use Case Diagrams

  Preliminary Conceptual Model of the System Use Case

  Domain Class Diagram

  Data Dictionary

  ERD or JSON model

  Project Quality Management Plan

  Project Communications Management Plan

  Implementation Schedule for PRJ666

  Work Breakdown Structure

PID Revision

- **Robert Parker-Lak**

  Added 10 Use Case Descriptions

  Revised the text from Introductions to SWOT Justification

- **Andriy Ostapovych**

  Added 10 Use Case Diagrams

  Revised the text from Introductions to SWOT Justification

  Added 6 Interface Mock-ups

- **Faramarz Hosseini**
  Added 1 Use Case Diagram
  Added 1 Use Case Description
  Added 3 Data Dictionaries
  Added Screen Shots
  Added PRJ666 Schedule

Research Sources

- Mobile Vs. Desktop Internet Usage - https://www.broadbandsearch.net/blog/mobile-desktop-internet-usage-statistics
- https://www.reddit.com/r/gamedev/comments/28td8c/how_are_databases_used_in_games/
- StackOverflow – research on databases vs arrays
- https://www.projectengineer.net/tutorials/pmp-exam-tutorial/project-communications-management/ - Communication management template research
- Game Database Design - https://vertabelo.com/blog/mmo-games-and-database-design/