Software Project Management Plan

### Project Horseman

### October 08, 2023

### RoguePixel Games

### Computer Science Dept/ CSUN

### COMP490, Edmund Dantes

# Revisions

| Version | Primary Author | Description of Version | Date completed |
| --- | --- | --- | --- |
| 1.0 | All Team Members | Original Document | 10/08/23 |
| 1.1 |  |  |  |
| 1.2 |  |  |  |
| 1.3 |  |  |  |
| 1.4 |  |  |  |
| 1.5 |  |  |  |
|  |  |  |  |

# Table of Contents

[**Revisions** 2](#_heading=h.gjdgxs)

[**Table of Contents** 3](#_heading=h.30j0zll)

1. [**Introduction** 4](#_heading=h.1fob9te)
   1. [Project Overview 4](#_heading=h.3znysh7)
   2. [Literature Review](#_heading=h.2et92p0) 7
2. [**Project Organization**](#_heading=h.tyjcwt) 9
   1. [Roles and Responsibilities](#_heading=h.3dy6vkm) 9
   2. [Tools and Techniques](#_heading=h.1t3h5sf) 10
3. [**Project Management Plan**](#_heading=h.4d34og8) 12
   1. [Tasks](#_heading=h.2s8eyo1) 12
   2. [Assignments](#_heading=h.17dp8vu) 14
   3. [Timetable](#_heading=h.3rdcrjn) 18

[**Additional Material** 1](#_heading=h.26in1rg)9

[Definitions, Acronyms and Abbreviations 1](#_heading=h.lnxbz9)9

[References](#_heading=h.1ksv4uv) 20

# Introduction

## Project Overview

##### **Purpose**

##### Our project is going to be a pixelated top-down role-playing video game. It is inspired by Nintendo games like Pokemon and Legend of Zelda. The purpose is to bring feelings of nostalgia and pay homage to the great games many of our generation started off playing. The theme of this game will be Halloween. It will have an immersive storyline that follows our character throughout the game with a variety of non-playable characters and enemies. Our main goal is to make a fun game that appeals to fans of classic Nintendo games.

##### **Scope**

##### Our project will be available on Steam. Anyone with a computer can play and we might expand it as LibGDX has Android and IOS capabilities. The intended audience are fans of Halloween or fans of pixelated top-down games (such as Legend of Zelda or Pokemon).

##### **Art Style**

##### The graphics are inspired by Nintendo games like Pokemon and Legend of Zelda. It will be a 32-bit pixelated game. The purpose is to bring feelings of nostalgia and pay homage to the great games many generations started off playing. Many of us do not have access to a working DS or GameCube like we once had and some of these online simulators are hard to set up or take up a lot of space on our devices. So we want to implement an easy-to-access game with similar graphics to these Nintendo games. We will be able to create as well as incorporate online sources to create a good-looking game.

##### **Story Synopsis**

##### Our main character Todd is tasked with saving his village from an evil curse plaguing his people. He must follow the trail of the legendary Headless Horseman. As he searches for clues and interacts with villagers he will piece together the story behind the curse that haunts the town every Halloween and save his people.

##### **Gameplay**

##### Combat

##### Our main character will be able to do common movements of left, right, up, and down. Limited combat will be used to solve puzzles and complete quests. Enemies will follow a path and have coded attack sequences similar to Legend of Zelda enemies.

##### Role-playing

##### Project Horseman will have many locations across the three acts. These three acts are Act 1: The Town, Act 2: Hollow Woods, and Act 3: Horseman Layer. The player will need to solve puzzles by finding keys and uncovering the secrets of the Horseman's past to advance throughout the game.

##### Exploration

##### In this dark and mysterious world, players will encounter friendly NPCs who are not just static characters but integral to the narrative. Engage in conversations with these NPCs to glean valuable information, unravel lore, and even receive assistance on your journey. Some of these NPCs may entrust you with side questlines, adding depth to the narrative and challenging you to perform tasks or uncover secrets that unlock hidden facets of the game's world. Your inventory becomes a vital companion as you collect important information, keys, and puzzle pieces.

##### Character Select

##### Our main character Todd has two Halloween costumes you can choose from Chicken Boy or Hooded Boy.

##### **Assumptions and Constraints**

##### Assumptions:

##### **Operating System**: Our game is designed to run on mac, windows & linux. We assume players will be using one of these operating systems.

##### **Sound**: We assume players will have speakers or headphones for the best gaming experience, as audio plays a crucial role in the atmosphere of our game.

##### **Gameplay**: The game is designed for a single player, the player will have their own set of controls on the keyboard or using external input devices.

##### **Stable Environment**: We assume that the game will not be interrupted by other software applications during gameplay.

##### Constraints:

**Resolution**: The game is optimized for 1920x1080 resolution, though it will scale to fit other standard resolutions.

**Controls**: Our game will support keyboard input and, in some cases, mouse input, but will not natively support game controllers.

**Offline Mode**: Our game is a desktop application that doesn’t require an internet connection for gameplay after initial download and installation.

**Multiplayer**: The game will not support multiplayer features. Not until the initial release, we may plan to add that in the future.

##### **Hardware Requirements**

**Windows**

**Minimum**

**OS:** Windows 7

**Processor (CPU):** Intel I5 or AMD Ryzen 5

**Graphics (GPU):** Nvidia GTX 650 or AMD Radeon HD 7850

**Memory (RAM):** 4 GB

**Storage (ROM):** 2 GB

**Recommended**

**OS:** Windows 10

**Processor (CPU):** Intel I7 or AMD Ryzen 7

**Graphics (GPU):** Nvidia GTX 1050 TI or AMD Radeon RX 570

**Memory (RAM):** 8 GB

**Storage (ROM):** 4 GB

**MacOS**

**Minimum**

**OS:** MacOS 10.12 (Sierra)

**Processor (CPU):** Intel I5

**Graphics (GPU):** Dedicated graphics with Open GL 3.2+ support

**Memory (RAM):** 4 GB

**Storage (ROM):** 2 GB

**Recommended**

**OS:** MacOS 11 (Big Sur)

**Processor (CPU):** Intel I7 or Apple M1

**Graphics (GPU):** Dedicated graphics with Open GL 3.2+ support

**Memory (RAM):** 8 GB

**Storage (ROM):** 4 GB

**SteamOS + Linux**

**Minimum**

**OS:** Steam OS 2.0 or Arch Linux prior to version 6

**Processor (CPU):** Intel I5 or AMD Ryzen 5

**Graphics (GPU):** Nvidia GTX 650 or AMD Radeon HD 7850

**Memory (RAM):** 4 GB

**Storage (ROM):** 2 GB

**Recommended**

**OS:** Steam OS 3.0 or Arch Linux version 6

**Processor (CPU):** Intel I7 or AMD Ryzen 7

**Graphics (GPU):** Nvidia GTX 1050 TI or AMD Radeon RX 570

**Memory (RAM):** 8 GB

**Storage (ROM):** 4 GB

## Literature Review

##### **Ideas / History / Audience**

Embracing the pixelated style reminiscent of the 8-bit and 16-bit eras, Project Horseman takes players on a journey that intertwines nostalgia, innovation, and a spooky story. Those who fondly remember the pixelated classics of the past will be drawn to the familiar art style and gameplay. "Hollow's Curse: Pursuit of the Headless Horseman" offers them a chance to relive the glory days of retro gaming with a modern twist. Younger gamers who may not have experienced the pixelated era firsthand can discover the magic of classic gaming aesthetics. The game provides an accessible entry point to a style that has stood the test of time. Fans of the horror genre, regardless of their gaming background, will find themselves immersed in a chilling narrative that pays homage to timeless horror tropes while introducing fresh, spine-tingling elements. Our game accommodates a wide range of players with its approachable pixel art style and straightforward gameplay. It invites those who may not typically engage with complex games to experience its haunting tale. By embracing the essence of classic gaming while utilizing contemporary tools, the game strikes a balance between nostalgia and innovation. Many of us do not have access to a working DS or GameCube like we once had and some of these online simulators are hard to set up or take up a lot of space on our devices. So we want to implement an easy-to-access game that will give off the same feelings Nintendo games once did.

##### 

##### 

##### 

##### 

During our research phase, we found several pertinent points that drove our decision-making for the project:

##### **Version Control**

GitHub version control can be incredibly beneficial for college group projects, especially when working on a project like LibGDX, which is a popular game development framework for Java. Here's how GitHub version control can help:

1. **Collaborative Work**: In group projects, multiple students are usually working on the same codebase simultaneously. GitHub allows for concurrent collaborative work by enabling multiple contributors to make changes to the codebase without interfering with each other. Each contributor can work on their own branch and merge changes when they are ready, reducing conflicts and making it easier to work together.
2. **Code Backup and Recovery**: GitHub serves as a centralized repository for your project's code. This means that even if a team member loses their local copy of the code or makes a mistake, they can easily retrieve the latest code from GitHub. This minimizes the risk of data loss and code corruption.
3. **Version History**: GitHub records a complete history of all changes made to the project. This is invaluable when you need to track who made what changes and when. You can review the commit history to understand the evolution of the project and pinpoint where issues might have arisen.
4. **Branching and Merging**: With GitHub, you can create branches for different features, bug fixes, or experimental changes. This allows you to work on new features or bug fixes without affecting the main codebase. When your changes are ready, you can merge them back into the main branch (typically master or main), ensuring that only well-tested code gets integrated.
5. **Issue Tracking**: GitHub offers an issue-tracking system that allows you to create, assign, and track tasks, bugs, and feature requests. This is beneficial for project management, as you can prioritize work and keep a record of what needs to be done.
6. **Code Review**: GitHub facilitates code review through pull requests. Team members can review each other's code changes, provide feedback, and suggest improvements before changes are merged into the main branch. This helps maintain code quality and consistency.
7. **Continuous Integration**: You can integrate GitHub with continuous integration (CI) tools like Travis CI or CircleCI. This automates the process of building, testing, and deploying your project whenever changes are pushed to the repository. This ensures that your codebase remains stable and error-free.
8. **Documentation and Wiki**: GitHub allows you to create and maintain project documentation and a wiki. This is useful for documenting the project's architecture, setup instructions, and other essential information that helps team members understand and contribute to the project.
9. **Access Control**: GitHub allows you to control who has access to your project. You can manage permissions for team members, granting read-only or read-write access as needed.

In summary, GitHub version control is an invaluable tool for college group projects, including those involving LibGDX or any other software development framework. It promotes collaboration, code quality, and project organization, making it easier for students to work together effectively on complex assignments.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Everhour, “What is GitHub? Everything You Need to Know”. [Online]. Available: <https://everhour.com/blog/what-is-github/> [Accessed: 3rd October 2023]
2. libgdx.com, “LibGDX”. [Online]. Available <https://libgdx.com/> [Accessed: 3rd October 2023]
3. spiceworks, “Why Is Java’s Use in the Gaming Industry Limited?”. [Online]. Available: <https://www.spiceworks.com/tech/devops/guest-article/why-is-javas-use-in-the-gaming-industry-limited/> [Accessed: 3rd OCtober 2023]
4. Github, “gson”. [Online]. Available: <https://github.com/google/gson>. [Accessed: 3rd October 2023]
5. Medium, “What is player behavior?”. [Online]. Available: <https://medium.com/@davengdesign/what-is-player-behavior-cdaa07271f86>. [Accessed: 3rd October 2023]
6. Jet Brains, “IntelliJ IDEA”. [Online]. Available: <https://www.jetbrains.com/idea/> [Accessed: 3rd October 2023]

##### 

# Project Organization

## Roles and Responsibilities

| Team Member | Roles | Email |
| --- | --- | --- |
| Jean Hanna | Team Lead, Head Java Developer, Gameplay tester | jean.hanna.885@my.csun.edu |
| Cailin Jefferson | Map & Level Designer, Gameplay tester | cailin.jefferson.567@my.csun.edu |
| Mikel Nuila | Functionality Programmer, Gameplay tester | mikel.nuila.618@my.csun.edu |
| Mohammed Hussain | Graphics Programmer, Gameplay tester | mohammed.hussain.403@my.csun.edu |
| Shawn Takhirov | Software Design Engineer, Gameplay tester | shawn.takhirov.647@my.csun.edu |

| Roles | Responsibility |
| --- | --- |
| Team Leader | In charge of leading and organizing the project and the team. responsible for gathering all inquiries and concerns, communicating those to the instructor, and turning in materials and assignments. responsible for directing team meetings and acquiring project-related data. Additionally accountable for delegating work and assignments to team members. |
| Java Developer | Responsible for writing Java code. Their main purpose is to program, design, analyze, and debug Java code for the game. |
| Functionality Programmer | Responsible for implementing and optimizing core gameplay systems, mechanics, and functionalities |
| Level Designer | Responsible for crafting the gameplay experience by designing and creating engaging and immersive game levels. |
| Map Designer | Responsible for creating detailed and visually appealing tile maps. |
| Graphics Programmer | Responsible for optimizing and implementing cutting-edge graphics technologies to create visually stunning and immersive gaming experiences. |
| Software Design Engineer | Plays a critical role in designing, developing, and maintaining software systems and tools that enable the creation and management of our game. |
| Narrative designer (need to assign people to this) | Responsible for shaping the storytelling and narrative elements of our games. |
| Gameplay tester (need to assign people to this) | In charge of testing the quality, functionality, and playability of our game by thoroughly testing and providing feedback on gameplay mechanics and systems |

## Tools and Techniques

#### **Tiled** map editor to create our maps, game world, and environment.

#### **Gradle** for building our project.

#### **YAML** for project configuration.

#### **Trello** for our project management system. Tasks are published by our Project Manager as cards. Team members select the cards with which they are comfortable with and start working on them.

#### **GitHub** which is an online repository service. Allows us to store and manage resources and code, in addition to tracking and controlling changes to code.

#### **Discord** is an online social messaging platform. Allows us to communicate with each other through text and voice chat.

#### **Waterfall method** is a project development with the breakdown of project activities into linear sequential phases and each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks.

#### **itch.io** game asset packs created by users that include artwork for our backgrounds, characters, and enemies.

#### **Adobe Photoshop** which is an image editing software.

#### **Steam** is a video game digital distribution service and storefront developed by Valve Corporation.

#### **Gson** which is a Java library that is commonly used to convert Java objects to JSON

#### **SnakeYAML** which is a parser and emitter library for Java

**LibGDX & Ashley Synergy**: LibGDX has shown strong performance metrics in the realm of 2D game development, making it a robust choice for desktop application games. This framework integrates seamlessly with Ashley, providing an efficient entity component system for better game logic separation and management.

**Java & Cross-Platform Compatibility**: Java remains one of the most versatile programming languages for cross-platform development. Leveraging its capabilities ensures our game will cater to a broader audience without the need for extensive modifications for different platforms.

**GSON’s Role in Data Handling**: Gson, a renowned Java library from Google, emerged as a top choice for our game's data serialization needs. Whether saving game states, loading configuration files, or facilitating potential network communications, Gson provides efficient conversion between Java objects and their JSON representations. This standardization streamlines development and enhances gameplay experience, especially in scenarios requiring quick data processing and storage.

**IntelliJ IDEA for Java Development**: Using IntelliJ IDEA, one of the premier Integrated Development Environments (IDEs) for Java, has accelerated our development process. With its powerful code analysis, debugging tools, and integrated support for LibGDX and Gson, IntelliJ IDEA enhances productivity, reduces development overhead, and allows for efficient code collaboration among the team.

# Project Management Plan

## Tasks

| Milestone | Task Description | Due Date |
| --- | --- | --- |
| Progress Report #1 | Report outlining our project's progress | 9/10/23 |
| Project Presentation | Presentation showcasing our project concept. | 9/17/23 |
| Progress Report #2 | Report outlining our project's progress | 9/24/23 |
| Software Project Management  Plan (SPMP) | Creating our plan for our project. | 10/08/23 |
| Progress Report #3 | Report outlining our project's progress | 10/08/23 |
| Progress Report #4 | Report outlining our project's progress | 10/22/23 |
| Project progress | Starting Act 1, all baseline for project done | Before 10/23/23 |
| Software Requirement Specification (SRS) | Assignment where we write project requirements | 10/23/23 |
| Progress Report #5 | Report outlining our project's progress | 11/05/23 |
| Code Review #1 Artifacts | Peer review assignment. | 11/12/23 |
| Project progress | Second act maybe? | Before 11/12/23 |
| Software Design Document (SDD) | Assignment where we design our project. | 11/12/23 |
| Progress Report #6 | Report outlining our project's progress | 11/19/23 |
| Code Review #2 Artifact | Peer review assignment. | 12/03/23 |
| Progress Report #7 | Report outlining our project's progress | 12/03/23 |
| Progress Report #8 | Report outlining our project's progress | 12/11/22 |
| Project progress | Demo mostly complete: multiple enemies and NPCs as well as player functionality done. Level design and environment for the first act done. | Before 12/11/22 |
| Software Test Plan (STP) | Assignment where we plan the testing for the project. | 12/17/23 |
| Project Presentation Demo | Full Presentation/Demo of code with documentation | 12/18/23 |
| Progress Report #9 | Report outlining our project's progress in 491 | 02/04/24 |
| Progress Report #10 | Report outlining our project's progress in 491 | 02/18/24 |
| Progress Report #11 | Report outlining our project's progress in 491 | 03/03/24 |
| Progress Report #12 | Report outlining our project's progress in 491 | 03/17/24 |
| Progress Report #13 | Report outlining our project's progress in 491 | 04/14/24 |
| Progress Report #14 | Report outlining our project's progress in 491 | 04/28/24 |
| Progress Report #15 | Report outlining our project's progress in 491 | 05/12/24 |
| Progress Report #16 | Report outlining our project's progress in 491 | 05/14/24 |
| Final Presentation | Present our finished game. | 05/14/24 |

## Assignments

| Task | Deliverables | Description | Due | Assigned |
| --- | --- | --- | --- | --- |
| Project planning | Planning | Get story premise with details on the main character, settings, and enemies. | September | All |
| Learn basic fundamentals of IntelliJ | Knowledge | Know basic knowledge of pushing and pulling as well as branches | September | All |
| Learn basic fundamentals of Tiled | Knowledge | Learn tiled software like how to upload files to IntelliJ, how to set up camera and collision | September | Cailin |
| Write Software Project Management Plan | SPMP | Class assignment where we write a plan for our project | 10/08/23 | All |
| Write Software Requirements Specification | SRS | Class assignment where we write our requirements for our project. | 10/22/23 | All |
| Start designing the first area | Environment design, level design | Early level design of the area our character first starts off in | October | All |
| Dialogue System | Programming | Set up dialogue system to get ready for starting in game narratives | November | John |
| Tiled Maps for Act 1 completed | Programming | Have all tile maps done for this area of the map | November | Cailin |
| Player Input/Output Control | Programming | Finish player input/output | Before October | Mikel and John |
| Title Screen | Programming | Have game title screen completed | Before October | Mohammed |
| Animation System | Programming | Finish code to change texture region | Before October | John |
| Weather | Programming | Finish code to add different weather conditions to code | End of October | Mikel |
| Act 1: Quest 1 | Programming | Finish code for quest | Before December | Shawn |
| Act 1: Quest 2 | Programming | Finish code for quest | Before December | Cailin |
| Act 1: Quest 3 | Programming | Finish code for quest | Before December | Mohammed |
| Act 1: Quest 4 | Programming | Finish code for quest | Before December | John |
| Act 1: Quest 5 | Programming | Finish code for quest | Before December | Mikel |
| Narratives | Programming | Complete dialogue between the main character and NPCs | December | All |
| Write Software  Test Plan | STP | Class  assignment  where we write a plan to test  the project. | 12/17/23 | All |
| Test Quests | Testing | Test the questlines | December | All |
| Finish game demo | Programming | Finish programming the demo of act one | December | All |
| Prepare game demo | Testing | Prepare the demo for our presentation | Before 12/18/23 | All |
| Present game demo | Presentation | Present a demo  of our game | 12/18/23 | All |
| Start Second Act | Programming | Start programming second act of game | February | All |
| Act 2: Quest 1 | Programming | Finish code for quest | February | Cailin |
| Act 2: Quest 2 | Programming | Finish code for quest | February | Mikel |
| Act 2: Quest 3 | Programming | Finish code for quest | February | Mohammed |
| Tiled Maps for Act 2 completed | Programming | Have all tile maps done for this area of the map | February | Cailin |
| Finish Second Act | Programming | Finish programming second act of game | End of February | All |
| Testing Second Act | Testing | Test second act of game | End of February | All |
| Start Third Act | Programming | Start programming third act of game | Beginning of March | All |
| Act 3: Quest 1 | Programming | Finish code for quest | April | John |
| Act 3: Quest 2 | Programming | Finish code for quest | April | Shawn |
| Tiled Maps for Act 3 completed | Programming | Have all tile maps done for this area of the map | April | Cailin |
| Finish Third Act | Programming | Finish programming third act of game | April | All |
| Testing Third Act | Testing | Test third act of game | End of April | All |
| Final Testing | Testing | Test entire game | Before 05/14/24 | All |
| Finish End Credits | Programming | Give credit to our team and any assist packs we used in our game | Before 05/14/24 | Mohammed |
| Final Presentation | Presentation | Present our finished game | 05/14/24 | All |

## Timetable

Blue- Coding

Red- Planning

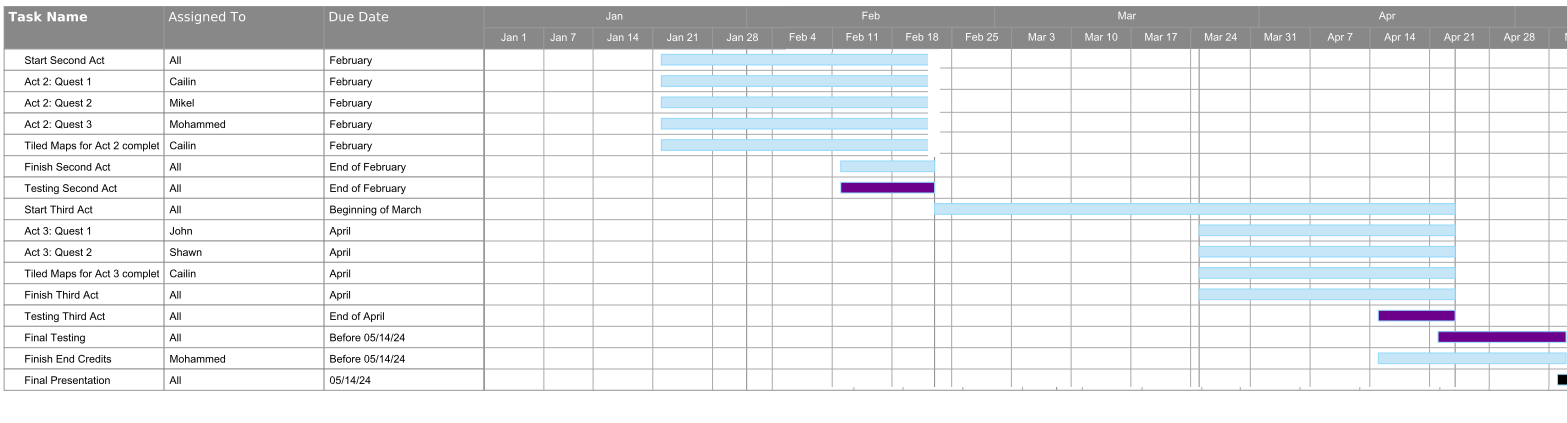
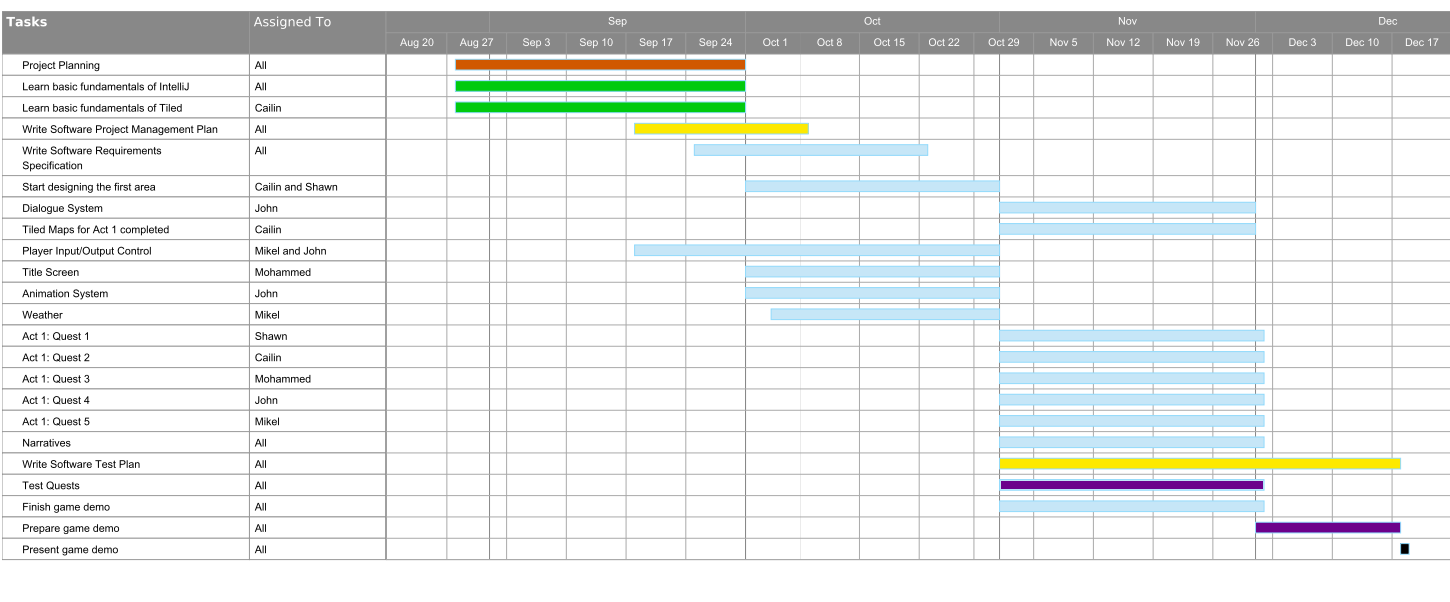
Yellow- Documentation

Green- learning

Purple- testing

Black- Presenting

Link to better picture: [Here](https://drive.google.com/file/d/1sWlFb7qz572P03J0qFobjMCqK9Rdm-tT/view?usp=drive_link)



# Additional Material

## Definitions, Acronyms and Abbreviations

| Acronyms and Abbreviations | Definition |
| --- | --- |
| Top-down Perspective | Video game where the player's perspective is from above and looking down. Also known as the overhead perspective. |
| Nintendo | Japanese video game company founded in 1889. |
| Legend of Zelda | An action-adventure game franchise created by Nintendo in 1986. We will be referring to earlier games that have a top-down perspective such as Legend of Zelda: Four Swords Adventures. |
| Pokemon | Franchise was created by Nintendo in 1996 and consists of video games, animated series and films, trading card games, and other related media. We will be referring to the earlier games that have a top-down perspective such as Pokémon Green Version. |
| Role-playing game (RPG) | Game in which players assume the roles of characters in a fictional setting by acting out roles within the storyline. |
| Non-player character (NPC) | Any character in a game that is not controlled by a player. |
| Steam | Popular game client service. Facilitates purchasing, storing, and playing of games. |
| Single-player | A game designed to be played by one person at a time |
| IDE | Integrated Development Environment |

## References

## IntelliJ Overview: <https://www.jetbrains.com/help/idea/discover-intellij-idea.html>