Project Plan

UMGC CMSC 495

Python Game Hub

Lee, Victoria (Member 1)

Mutry, James (Member 2)

Foster, Todasha (Member 3)

Chung, Dajin (Member 4)

Payne, Javon (Member 5)

Ipaye, Oluwatumininu (Member 6)

Table of Contents

[1. Project Overview 2](#_Toc193139081)

[2. Group Organizational Structure 3](#_Toc193139082)

[3. Statement of Work & Project Management 5](#_Toc193139083)

[4. Schedule 8](#_Toc193139084)

[5. References 9](#_Toc193139085)

[6. Table: CMSC 495 Project Plan 11](#_Toc193139086)

# Project Overview

**Project Purpose:** The goal of this project is to develop a Python-based Game Hub, a centralized platform that hosts multiple simple and engaging games. The hub will provide an interactive user interface, allowing users to select and play different games within a single application. The games included will be Break-Out, Tic-tac Toe, and Trivia.

**Project Deliverables:** The deliverables for the Python Game Hub project include a fully functional and interactive game hub application that serves as a centralized platform for multiple mini games. The application will feature a main menu for game selection, allowing users to navigate between different games. Each game will be developed as a separate modular component, creating easy maintenance and potential expansion in the future. The project will also include a graphical user interface for an easy-to-navigate user experience, built mainly using Pygame. If applicable, a lightweight SQLite database will be integrated to store user profiles, high scores, and game progress. The project’s source code repository will be hosted on GitHub or another version control platform for version tracking, collaboration, and future enhancements.

# Group Organizational Structure

**Overview:** This project’s organizational structure is split into several different units. The overall structure is led by the Project Manager, and each unit consists of a leader responsible for the management of that section. However, it is important to note that all team members are flexible and responsible for multiple duties throughout all units. This means that a person can have multiple roles, so if there is a name that is mentioned more than once, it demonstrates multiple roles.

**Meetings:** All team members will have flexible roles and may be responsible for multiple areas throughout the project. Team meet-ups will be on Wednesdays and Saturdays from 5:30 PM to 11:59 PM, with team members able to join or leave as needed. Below is our team’s sample of the Gantt Chart below for what the communication, period and goals are (UMGC, n.d). We will be using this schedule to make sure we stay focused, however; plans are not set completely and therefore might be changed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Communication | Application | Time Period | Goal | Owner |
| **Project Team** | | | | |
| Project Status/ Routine Meeting | Discord | Weekly (Wednesday & Saturday) | Review status of project modules and discuss details of deliverables for the week | Project Manager |
| Project Draft and Consolidation | Discord/Google Drive/Microsoft Teams | 1-2 days prior to deadline | Discuss Project Draft consolidation details for any issues, delays, and suggestions | Everyone |
| Task/Project Review | Discord | During routine meeting | Discuss feedback from Project Draft and Consolidation submitted the prior week | Testers (Incase: All) |
| Final Product Demonstration | Discord | At end of project | Demonstrate and explain all functionality to the Project Manager and Testing Director Roles. | Project Manager (Incase: All) |
| Issue Resolution | Call/Text/Discord | As needed | Resolve issues that could delay or prevent on-time deliverable submission | All |
| **Project Approval for Modifications** | | | | |
| Deliverable Submission | UMGC Submission Folder | Weekly | Provide project status update and present tentative deliverable | Project Manager |
| Final Product Demonstration | UMGC Final Submission | At end of project | Demonstrate and explain all functionality to project sponsor | All |
| Feedback | UMGC Discussion Board | Weekly | Receive feedback on weekly deliverables | All |

**Roles/Responsibilities:**

**Project Lead - Victoria Lee**

The Project Manager is responsible for acting as a liaison between the team and upper management (professor), including communication between teammates and unit leads. Additionally, the Project Manager is responsible for scheduling and ensuring that all deliverables are completed and submitted on time, as well as organizing and recording all necessary forms of contact and conversation among the team.

**Documentation — James Mutry, Victoria Lee**

Those assigned to Documentation are responsible for the organization of all documentation and work with the Project Manager, the client, and all team members to ensure that all deliverables meet the requirement in the statement of work. The Requirements Manager for this team also takes the role of the Technical Writer and is responsible for creating and maintaining documentation required in the process of developing the software solution.

**Testing — James Mutry, Oluwatumininu Ipaye, Victoria Lee**

The Testing team is responsible for creating testing scenarios to ensure that all requirements for the software are met. Additionally, Testing is responsible for carrying out these scenarios and identifying any defaults in the program, as well as maintaining documentation of these tests and directing developers towards the issue that requires attention.

**Developers - Javon Payne, Dajin Chung, Todasha Foster, Oluwatumininu Ipaye**

Developers are responsible for creating the design of the program in accordance with mathematical, PyGame logic, and game rules that correlate with the game type. Additionally, the Developers are responsible for working with the Project Manager, and all developers to create a functional final product. The Work Breakdown chart below shows the distribution of the roles and their goals. It will also show any other exceptions to the project (UMGC, n.d.).

|  |  |  |
| --- | --- | --- |
| Component/Game | Developer | Responsibility |
| Tic-Tac-Toe | Dajin Chung | Develop and implement Tic-Tac-Toe with working game logic and UI. |
| Breakout | Oluwatumininu Ipaye & Todasha Foster | Design and develop the Breakout game with proper physics and scoring |
| Menu/UI System | Javon Payne & Todasha Foster | Create the main game hub menu, navigation system, and overall user interface. |
| Trivia Game | Javon Payne | Develop a trivia-based game with a database of questions and user input handling. |
| General Support | Victoria Lee & James Mutry | Assist in debugging, UI improvements, and additional game development as needed. |

# Statement of Work & Project Management

As a team and demonstrated in unit 1 group discussion, every member has agreed to work on this project (Python Game Hub) together. The finalized project will be a Python Game Hub featuring the developed games using the Pygame library such as Breakout, Tic-Tac-Toe, and Trivia. These games will be designed to help beginners to intermediate Python programmers practice coding, user interface design, and event-driven programming. Each student will complete their assigned roles and goals while also helping other students along. The final submission should be run on a compatible desktop where the user can run the game hub and will allow users to select and play the games. This is a statement agreed upon by the team to work on this project to learn basic concepts of game development in Python.

**Work Location & Timeline:** All work will be completed virtually, as this is a desktop-based Python application, and have weekly follow up meetings during Wednesday and Saturday from 5:30pm to 11:59pm ET where they join then leave any time after the meeting. The project will follow the timeline outlined below, with the final product to be delivered on May 3, 2025. The project schedules are detailed in the **5. Schedule** and **6. CMSC 495 Project Plan**. The following is an overview of the timeline:

* Project Start Date: March 16, 2025
* Final Submission Date: May 4, 2025
* Official Date CMSC 495 Ends: May 6, 2025

**Assumptions, Constraints, Quality, and Costs:** The Python Game Hub Development project operates under several assumptions and constraints that influence its scope and execution. It is assumed that users will have a basic understanding of navigating a desktop application, and the system requirements will be minimal, ensuring compatibility with most operating systems. Additionally, the project assumes that Pygame and other smaller libraries will be sufficient for building the graphical user interface, and that an SQLite database (if implemented) will handle user profiles and high scores efficiently without requiring a complex backend. Constraints include time and resource limitations, meaning the project will focus on a predefined set of mini games rather than extensive customization or highly advanced game mechanics. To ensure the quality of the project is completed well, every individual team member is responsible for their parts. To help maintain quality, the Tester role will thoroughly test the games to ensure they are bug-free and meet the project requirements. Since this is a project on the Python Game Hub, we will manage costs throughout SDLC and since this is using only open-source application and python, there is no budget required for purchasing software or hardware (GeeksforGeeks, n.d). Since there are no major financial resources required, the completion of the project will be monitored based on time and effort spent by each team member.

**Project Risks:** The Python Game Hub Development project carries several potential risks that could impact its timeline, functionality, and overall success. Technical risks include possible bugs, crashes, or performance issues, particularly when integrating multiple games into a single platform. Ensuring smooth transitions between games, proper resource management, and avoiding memory leaks will be crucial. Development risks arise if any developer encounters difficulties in implementing PyGame logic or game mechanics, potentially leading to delays or inconsistencies in gameplay. Team collaboration risks include miscommunication or lack of coordination, which could result in integration challenges between different game modules and the UI. Additionally, scope creep is a potential risk if new features or games are added without proper planning, leading to extended development time and resource constraints. User experience risks include unintuitive navigation or poorly optimized controls, which could affect engagement and usability. To mitigate these risks, the team must maintain regular communication, follow a structured development plan, conduct thorough testing, and adhere to project scope and deadlines to ensure a smooth and functional final product.

**Process:** This team group will be following the SDLC process model to help develop and deploy the Python Game Hub. This allows the team to plan in the early stages to prevent major design flaws from developing and below is the SDLC (GeeksforGeeks, n.d.)

Requirements

Design

Implementation

Verification

Maintenance

**Application Requirements:**  These requirements define the core functionalities of the Python Game Hub application.

**Main Menu/UI:**

* Select a game to play
* View Game Instructions
* Exit the application

**Tic-Tac Toe:**

* 2-player mode
* Display Win, Loss, or Draw Conditions
* Clear game board for a new game

**Breakout:**

* Ball movement with physics-based collision
* Player-controlled paddle movement
* Brick destruction mechanics and score tracking

**Trivia:**

* Display random trivia questions with multiple-choice answers
* Keep track of correct and incorrect answers
* Provide feedback for user responses

**Technical Requirements:** For the Python Game Hub project, the development team will utilize the following tools and techniques:

* Language – Python
* Library – PyGame (for game development)
* IDE – Any Python IDE (PyCharm, Visual Studio Code, etc)
* Code repository/ Version Control – GitHub for Git (sync files)
* Documentation – Microsoft Word
* Communication – Discord (verbal/text communication), Google/Microsoft Teams or Drive (Sometimes for file sharing as well)

The development team will use a Python IDE (e.g., PyCharm or Visual Studio Code) to write and test the code. These IDEs provide excellent debugging tools and support for Python, making them ideal for development. Git will be used for version control, allowing the team to manage changes, track progress, and collaborate effectively. The code will be shared via a Git repository (such as GitHub or GitLab), ensuring that all team members have access to the latest updates and can work on different parts of the game concurrently. All project documentation will be maintained in Microsoft Word. This ensures the documentation is easily accessible and compatible for team collaboration and future reference. The development team will use Discord or other alternatives for team communication, discussions, and updates. By utilizing Python, Pygame, Git, and other tools listed above, the project will be developed efficiently.

# Schedule

Below is our team’s sample of the Gantt Chart below of our plans and we will be using this schedule to make sure we stay focused, however; plans are not set completely and therefore might be changed (UMGC, n.d.).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Week | Dates | Lead | Topic | Description | Due Date | Assignments Due |
| 1 | 3/10-3/14 | Everyone | Team Formation | Get to know team members & pick a project | 3/14 | N/A |
| 2 | 3/15-3/18 | Project Manager | Project Plan | -Outline Milestones  -Delegate responsibility  -Describe projects purpose | 3/18 | N/A |
| 3 | 3/18-3/24 | Documentation  And Requirements | Project Plan | -Create formal Project Plan | 3/25 | Project Plan |
| 4 | 3/25-3/31 | Documentation  And Developers | Design | -Create application structure  - Develop user interface/functionality  -Create UML diagrams | 4/1 | Project Design |
| 5 | 4/1-4/8 | Everyone | Phase 1 Source | * Software Development | 4/8 | Phase 1 Source  Peer Review 1 |
| 6 | 4/8-4/14 | Everyone | Testing | * Software Development | 4/15 | Test Plan |
| 7 | 4/15-4/22 | Everyone | Phase 2 Source | * Software Development | 4/22 | Phase 2 Source  User Guide |
| 7 | 4/22-4/25 | Everyone | Phase 2 Source (Extended) | * Software Development * (Last week to finalize project!) | 4/27 | Phase 2 Source  User Guide |
| 8 | 4/27-5/4 | Everyone | Final Report | * Compile all Topics into a single document | 5/4 | Final  Peer Review 1 |

**Note:** Last day of CMSC 495 is on May 6, 2025, at 11:59pm ET, but we will turn it in on May 4, 2025 at 11:59pm ET in case we want to make some minor changes. (ex: reupload files, etc.)

# References

GeeksforGeeks. (n.d.). Software Development Life Cycle (SDLC). <https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/>

Stellman, A., & Greene, J. (2005). Software project planning. In *Applied software project management* (Part I, Chapter 2, sec. 2.2-2.3). O'Reilly Media. <https://learning.oreilly.com/library/view/applied-software-project/0596009488/ch02s02.html?sso_link=yes&sso_link_from=umgc>

University of Maryland Global Campus. (n.d.). *Sample Project Plan 1*. UMGC. <https://leocontent.umgc.edu/content/dam/course-content/tus/cmsc/cmsc-495/document/Sample%20project%20plan%20UMGC%20CMSC%20495.pdf?ou=1248245>

University of Maryland Global Campus. (n.d.). *Sample Project Plan 2*. UMGC. <https://leocontent.umgc.edu/content/dam/course-content/tus/cmsc/cmsc-495/document/Sample%20Project%20Plan%202.pdf?ou=1248245>

# Table: CMSC 495 Project Plan

**A screenshot of a computer

AI-generated content may be incorrect.**