Final Report

Unit 8: Deployment

Lee, Victoria (Member 1)

Mutry, James (Member 2)

Foster, Todasha (Member 3)

Chung, Dajin (Member 4)

Payne, Javon (Member 5)

Ipaye, Oluwatumininu (Member 6)

Python Game Hub

UMGC: CMSC 495 – Section 2252

5/3/2025

Table of Contents

[**1. Introduction 3**](#_heading=h.qunsrjijnxrw)

[**2. Process Overview 3**](#_heading=h.j487iinmyk59)

[**3. Project Plans 5**](#_heading=h.wtnmxqz17ow7)

[**4. Requirements and System Specification 6**](#_heading=h.vux1s6qg3zc2)

[**5. Project Design and User Guide 10**](#_heading=h.14n06vza01rz)

[**6. Project Evaluation, Test Plan, and Results 13**](#_heading=h.yv026ksmtgvt)

[**7. Design and Alternate Designs 16**](#_heading=h.le653m3ahtf)

[**8. Development History and Deployment 18**](#_heading=h.32xbchbzi2eq)

[**9. Discussion and Target Audience 21**](#_heading=h.kpt74er49p79)

[**10. Conclusion 22**](#_heading=h.abvr81hishqe)

[**11. Acknowledgements 23**](#_heading=h.7cwg3ijsf63o)

[**12. References 25**](#_heading=h.lsd1ip7nog55)

[**13. APPENDIX A: 29**](#_heading=h.5t5m2awz9gsb)

[**14. APPENDIX B: 34**](#_heading=h.ut9r51uz962g)

[**15. APPENDIX C: 43**](#_heading=h.r24si1m2qbsq)

[**16. APPENDIX D: 44**](#_heading=h.7nmtktb0rp1y)

[**17. APPENDIX E: 53**](#_heading=h.7ypckt9a2en3)

[**18. APPENDIX F: 65**](#_heading=h.iiyex3bmrcdq)

[**19. APPENDIX G: 65**](#_heading=h.so2vaco0npof)

# **Introduction**

The Python Game Hub project was developed as the capstone for CMSC 495 at the University of Maryland Global Campus. This project aimed to create a centralized game application written in Python that allows users to navigate and play multiple mini-games from a single main menu. The primary goal was to apply real-world software development principles—such as requirements analysis, design, testing, deployment, and documentation—using the structured Waterfall methodology.

Throughout the project, the team designed and implemented three fully playable games: Tic Tac Toe, Trivia, and Breakout, each accessible through an interactive graphical interface built with the Pygame library. An emphasis was placed on making the Game Hub user-friendly, visually appealing, and modular to allow for potential future expansion. The design process incorporated clear user interaction paths, dynamic background effects, animated buttons, audio integration, and consistent cross-platform support (Windows, macOS, and Linux).

Every phase of the software development lifecycle (SDLC) was carefully documented and executed: from the initial project plan to system specification gathering, design creation, coding, extensive unit and manual testing, and finally deployment preparation. Testing strategies included both automated unit testing via Python’s unittest library and GitHub Actions for CI/CD validation, as well as structured manual testing to simulate real user interactions.

This final report summarizes the entire project lifecycle, including the planning and design stages, the functional and non-functional requirements met, individual and team contributions, evaluation through rigorous testing, lessons learned, and suggestions for future improvements. Ultimately, the Python Game Hub project served not only as a platform to demonstrate technical skills but also as a comprehensive exercise in professional team collaboration, project management, and software engineering best practices.

# **Process Overview**

**Python Game Hub Approach:** For our Python Game Hub project, our team followed a structured and collaborative approach centered around modular development, continuous integration, and regular testing. Our team will be following the SDLC process model to help develop and deploy the Python Game Hub. We worked iteratively to design and build a multi-game application composed of a central Main Menu and three mini-games: Tic Tac Toe, Trivia, and Breakout. We divided work based on each member’s strengths, frequently collaborated through GitHub, and used a combination of manual and automated testing to maintain quality. Peer reviews and debugging sessions helped us address issues early and improve the overall stability of the system.

**Python Game Hub Tools/Artifacts Used:** We used several tools and artifacts to support the development and documentation of the project which are briefly explained below.

* **Python 3.9+**: Core programming language for the project
* **Pygame**: Library for handling graphics, user input, and game rendering
* **GitHub**: Version control system for code collaboration, pull requests, and reviews
* **PyCharm / VSCode**: Integrated Development Environments (IDEs) used by team members
* **GitHub Actions**: Implemented basic CI/CD workflows to automate unit testing and file existence checks
* **Excel**: Used to create and maintain the Manual Test Plan covering ~20 test cases
* **README.md**: Documented project setup instructions, installation steps, gameplay details, and troubleshooting
* **Unit Tests**: Built using Python’s unittest framework to cover critical functionalities across modules

**Python Game Hub Summary of each individual contributions (all members):**

**Chung, Dajin**:

* Created the Tic Tac Toe mini-game including full game logic, structure, and AI opponent using the minimax algorithm.
* Assisted in debugging and troubleshooting complex logic issues in both the Trivia and Breakout games when other developers encountered difficulties.

**Lee, Victoria**:

* Focused on project documentation, updated and improved the UI across games, assisted with debugging, implemented GitHub Actions for testing, and contributed to enhancing the Main Menu and mini-games when needed.

**Mutry, James**:

* Led the manual testing about (20) and unit testing efforts by developing the test.py script, maintaining around 24 unit tests across the main modules, and collaborated with Victoria to verify code correctness through GitHub Actions.

**Foster, Todasha** and **Payne, Javon**:

* Designed and implemented the Main Menu system, managing the navigation between mini-games, and ensuring a unified user interface for the Game Hub application.

**Ipaye, Oluwatumininu**:

* Developed the Breakout mini-game from scratch, incorporating physics, collision detection, block-breaking mechanics, and scoring.

# **Project Plans**

The Project Plan and Design Plan were essential in our Python Game Hub project as they provided a clear roadmap and structure, helping the team stay organized, meet deadlines, and ensure all features were well thought out before coding began. The development phases (Phase 1 and Phase 2) ensured all core and advanced features were implemented and refined. The Testing plan confirmed functionality and reliability across all games, while the User Guide made the project accessible for users. Finally, the Deployment Report and Presentation ensured everything was ready, reviewed, and clearly communicated for final submission.

**Project Plans:**

* **Unit 2: Project Plan**
  + The Project Plan is a foundational document that outlines the overall strategy for the project. It includes key tasks, milestones, deadlines, and team roles. It serves as a roadmap to guide the development process, ensuring that the team stays organized and on schedule throughout the duration of the project. The updated changes to the timeline and tasks begin around Unit 4 and Unit 5 to adjust to the feedback and delays.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1F2LR0j118Oje_9Gw5iaabmx6HMe-YYtb/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit2)
* **Unit 3: Project Design Plan**
  + The Project Design document includes all design-related materials and planning. This may involve UI/UX sketches, architecture diagrams, data flow charts, and detailed descriptions of how different components of the game hub (like menus and individual games) interact. It acts as a blueprint for development and ensures that everyone on the team shares a common understanding of the system’s structure.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1hjNWfBtiacfR40g9wAcIkf-xypBD3PlF/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit3)

**Other Plans:**

* Unit 4: Phase 1 Source Report Plan
  + This Phase 1 document captures the early development work where core features and basic game functionality were implemented. It typically includes source code implementation details for key components like Mainmenu.py, TicTacToe.py, Breakout.py, and the initial version of Trivia.py. The document reflects the progress made during the foundational stage of building the application.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1oHjphF6c_7OVMOcpK2WWTJcwo7Pe4vee/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit4)
* Unit 5: Testing Plan
  + The Testing document contains records of all the testing activities, including manual tests, unit tests, and validation processes. It demonstrates how each feature was tested to ensure functionality and stability. For this project, it also included completed tests for Trivia.py and validation of the Questions.json file, confirming data accuracy and bug-free implementation.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1-Gexgrs11wsRRzTdWwDQkw0dHNirzS0a/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit5)
* Unit 6: Phase 2 Source Report Plan
  + The Phase 2 document details the final stage of feature development. It covers the implementation of advanced functionalities, final code refinements, and all tests applied. It also includes documentation practices such as adding inline code comments, maintaining a detailed README.md, and organizing supporting files. This phase finalizes both the technical and documentation aspects of the project.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit6)
* Unit 7: User Guide
  + The User Guide is a user-facing document designed to help end-users understand how to install, launch, and navigate the game hub. It includes setup instructions, game rules, descriptions of each game feature, and links to resources (like the GitHub repo). It’s written in a clear and simple format for non-developers and is a key part of the project’s deliverables.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)
* Unit 8: Deployment Report
  + The Deployment Report outlines how the project was packaged and prepared for final submission. It details everything included in the final release—source code, user guide, tests, documentation, and any relevant project files. It ensures that everything has been verified and meets the submission requirements. This report typically includes final reflections, validation checks, and readiness for peer review. Meanwhile, the Google Slide (Video) presentation of this is a visual summary of the entire project, suitable for stakeholders, instructors, or peer reviewers. It highlights the objectives, key phases (Plan, Design, Development, Testing, Deployment), challenges faced, major accomplishments, and a walkthrough of the game hub’s features. It complements the deployment report by offering a high-level overview with visuals and talking points.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1CYzgpDp1lFNmwkbjhaIwnEbEDStKLXPn/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [GitHub (doc) Link] (Check it at the GitHub repository doc folder!)
    - [[Google Slides Link]](https://docs.google.com/presentation/d/10nZwOQwZNqL12yTPeKOJZYSwu5xBDgPzDS-rd9FtDY8/edit?usp=sharing) (Check it at the GitHub repository doc folder or the Unit 8 discussion!)  [[PythonGameHub GitHub Repo Link]](https://github.com/javonpayne100/CMSC495Capstone)

# **Requirements and System Specification**

**Project Plan (Unit 6):** The Phase 2 document details the final stage of feature development of the functionalities, final code refinements, and all tests applied. It also includes documentation practices such as adding inline code comments, maintaining a detailed README.md, and organizing supporting files.

All Requirement and System Specifications are covered in the finalized project plan in Unit 6.

Access it by clicking on the link here: [Unit 6 document](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101848940126344817524&rtpof=true&sd=true)

Extra Link (GitHub Link): [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit6)

**README.md:** The link below renders the code to README.md. This readme file should show how to install, run the application, and install any imports. It will also explain how to play the games and how the unit tests are done. It also teaches the user how the unit test is processed.

<https://gist.github.com/34c40d49bcef96c70ed35a5a80c33b92.git>

**requirements.txt:** The code below renders the code of the requirements.txt. This file should allow the user to import and install dependencies into the PyCharm environment in order to run the application. If it does not work, the user has other options to install them following the directions on the README.md file. If it doesn't work the user must also install these: random; math; time; json; unittest.

**Code of the requirements.txt file:**

pygame

pyautogui

numpy

**Project Purpose and Objectives:** The purpose of the GameHub is to create a public amenity that consists of multiple games for schools, libraries, hotels, and similar environments. It is built with inclusivity in mind, ensuring that all age groups can seamlessly enjoy engaging games without the hassle of downloading, signing up, or connecting to the internet.

* Main Menu: Serves as the central navigation hub, allowing users to select from available games. All game files are connected to this interface for ease of access.
* Tic-Tac-Toe: A single-player game where the user competes against the CPU using simple Tic-Tac-Toe strategies..
* Trivia: This is a 1-2 player game in which the user plays against themselves or another player. The players then get to select a category and must answer questions within 10 seconds. The player with the highest score wins.
* Breakout: A classic arcade-style game where players control a paddle to break color-coded bricks, each with varying levels of durability.

GameHub will serve as a valuable addition to public spaces by offering offline, inclusive entertainment that encourages cognitive development and community engagement. It will provide a smooth, no-barrier gaming experience tailored for diverse users. GameHub reflects a thoughtful blend of fun, accessibility, and usability, aiming to become a staple interactive solution in shared environments.

**Python Game Hub’s Core Functionalities and Features:**

**Game Playability**

* Each game must be fully functional and interactive, with clear, easy-to-understand rules displayed or explained to the user.
* Users should be able to complete a full game cycle (start, play, end) without errors or confusion.

**Progress and Score Tracking**

* Implement tracking for winners, losers, ties, high scores, and remaining lives
* The scoreboard should dynamically display current results and update in real time based on game outcomes.

**User Interface (UI)**

* The interface must be simple, intuitive, and responsive across different screen sizes.
* Navigation between games and menus should be seamless and require minimal user input.

**Performance**

* All games should load within 3 seconds.
* Gameplay is smooth and free of noticeable lag, ensuring a positive user experience on standard hardware.

**Version Control**

* All development will be tracked using Git, allowing team members to monitor changes, resolve conflicts, and revert issues efficiently.
* Feature branches and commit messages should follow consistent naming conventions for clarity.

**Testing and Debugging**

* Unit testing (including Github Actions) and Manual testing will be applied to individual components of both the games and UI to verify functionality during development.
* Functional and non-functional requirements will be tested regularly during debugging to ensure quality and reliability.
* Coders are responsible for debugging any feature that fails to meet expected behavior, particularly scoreboard and life tracking.

**Project Deliverables:**

The final deliverable will be a fully interactive GameHub featuring three games: Tic-Tac-Toe, Trivia, and Breakout. A central main menu will allow users to browse and launch each game.

Each game will be built as a separate module for easy maintenance and debugging. Audio effects will enhance user experience, and scores will be tracked during gameplay.

Users can restart, exit, or return to the main menu at the end of each session.

The complete source code will be hosted on GitHub with version control via Git. Unit testing will be used to verify functionality across all components.

**Functional and Non-Functional Requirements:**

#### **Functional Requirements:**

1. **Game Accessibility:** Each game must be fully playable with clear, easy-to-understand rules.
2. **Main Menu Navigation:** Users should be able to easily navigate the main menu and select any game to play.
3. **Outcome Tracking:** The system must track and display game outcomes, including winners, losers, ties, scores, and remaining lives.
4. **Scoreboard:** A real-time scoreboard must show current scores and update at the end of each round.
5. **Session Control:** Users should be able to restart the game, return to the main menu, or exit after each session.

#### **Non-Functional Requirements:**

1. **Performance:** Games must load within 3 seconds and run smoothly without noticeable lag.
2. **Usability:** The interface should be intuitive and user-friendly, accessible to users of all ages.
3. **Maintainability:** The game design should be modular, allowing for isolated updates and bug fixes without disrupting other components.
4. **Reliability:** The system must handle all expected inputs and edge cases reliably, with no crashes.
5. **Testability:** Unit testing should be conducted to ensure each component performs as expected and meets functional requirements.

**Note: Refer to Unit 6 documentation link for more details!**

# **Project Design and User Guide**

**User Guide:** The Python Game Hub User Guide is a comprehensive document that has been vetted by the team and the professor through multiple revisions. The current version provides clear and concise step-by-step instructions on how to access each section of the Game Hub, detailing their rules and controls to allow effective usage. Instructions given are laid out with images and links when necessary to guide inexperienced users through the Game Hub. The final version of the User Guide is embedded in this document and can also be accessed by clicking the link below.

**User Guide Link (from Unit 7):**

[User Guide CMSC495 Capstone.docx](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)

Alternative link (GitHub version): [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)

**Python Game Hub:** The Python based Game Hub is a desktop-based application featuring multiple games. It was developed as a capstone project for the UMGC course CMSC 495. Since the Game Hub is based on the Python language, it utilizes the Pygame library and features three simple games: Tic-Tac-Toe, Trivia, and Breakout. The idea behind the Game Hub is that it is designed to be pre-installed by default on shared desktops in places such as schools, libraries, hospitals, etc. In short, it should serve diverse user bases and as such, is targeted towards all age groups. As there is no need for installation or account setup, the Game Hub simplifies access to digital entertainment. Its offline functionality, smooth UI, and ready-to-play games, also bolster this fact. With its low maintenance design, it gives a trouble free means of addressing bugs and errors without ruining the user’s experience. One should note that the only need for an internet connection is to access further information about the Game Hub.

**Project Setup:** This portion is in the code section, scroll to find the source code and find the README.md file and click on it to read the code. That code explains all the steps to install, run, and make the application work. It also includes the documentation and how to play the games, the unit tests, manual tests, GitHub Actions and many more.

**README.md:** The link below renders the code to README.md. This readme file should show how to install, run the application, and install any imports. It will also explain how to play the games and how the unit tests are done. It also teaches the user how the unit test is processed.

<https://gist.github.com/34c40d49bcef96c70ed35a5a80c33b92.git>

**requirements.txt:** The code below renders the code of the requirements.txt. This file should allow the user to import and install dependencies into the PyCharm environment in order to run the application. If it does not work, the user has other options to install them following the directions on the README.md file. If it doesn't work the user must also install these: random; math; time; json; unittest.

**Code of the requirements.txt file:**

pygame

pyautogui

numpy

**Python Game Hub’s Development Environment:**

The Python Game Hub was developed in a cross-platform environment supporting Windows, macOS, and Linux. The project was implemented using Python 3.9 or later and developed in IDEs like VSCode and PyCharm, as well as directly through the command line, terminal, or PowerShell. The primary game interface and functionality were built using Pygame for rendering graphics, animations, and audio. The development environment ensured compatibility across operating systems, with a display resolution of at least 600x400 to maintain visual consistency. Continuous integration was handled via GitHub Actions, running automated unit tests on different environments (Windows, Mac, Ubuntu).

**Python Game Hub’s Completion and Verification of All Project Steps Setup:**

All major development and setup phases have been completed and verified. This includes:

* Final implementation of all three mini-games (Tic Tac Toe, Trivia, and Breakout).
* Development of a functional and interactive Main Menu (MainMenu.py) to navigate between games.
* Execution of unit tests using unittest with 13 tests in the main branch and 24 tests in the James branch.
* Successful integration and validation of core functionalities including AI mechanics, question validation, and collision detection.
* Manual testing verified via a structured 20-point checklist in the CMSC 495 Project Test Plan.xlsx. This accounts for integration and user-interaction testing.
* Ensured that all necessary assets (e.g., questions.json, sound files like brick.wav, wall.wav) were properly linked and functioning.
* Refer to APPENDIX A to see the Tasks, milestones, and overall completion progress comparing it for Units 1-8.
* Refer to APPENDIX B to see the expansion of the job contribution, tasks, and milestones completed.
* Refer to APPENDIX D to see other Testing expansions completed.
* Refer to APPENDIX E to see Git (version control) issues, tasks, milestones completed.

**Python Game Hub’s Coding Tools:**

* **Python 3.9+**: Primary programming language.
* **Pygame**: Core library used for game logic, graphics, and audio.
* **PyAutoGUI**: Used for handling GUI prompts (e.g., replay options).
* **Numpy:** Used for handling timer, or mathematics code.
* **Unittest**: Python’s built-in framework for unit testing.
* **Git**: For version control and branch management.
* **GitHub**: For collaboration and CI/CD integration via GitHub Actions.
* **IDE Tools**: PyCharm, or similar environments like Visual Studio Code, were used for writing and debugging code.

**Python Game Hub’s Source Libraries:**

* python 3.x – this is the basic library for python including pip to install imports and ve.
* pygame – For all game-related graphics, event handling, and sound.
* json – To parse and load question data for the Trivia game.
* math – Used for calculations in graphics (e.g., countdown timer arcs).
* random – For shuffling trivia questions and randomized game behavior.
* sys, time – General system-level operations and timers.
* unittest – Automated testing framework.
* numpy – Used where numerical array management is required.
* pyautogui – For GUI prompts like replay confirmation screens.

P**ython Game Hub’s Dependencies:**

**README.md:** The link below renders the code to README.md. This readme file should show how to install, run the application, and install any imports. It will also explain how to play the games and how the unit tests are done. It also teaches the user how the unit test / manual tests are processed.

<https://gist.github.com/34c40d49bcef96c70ed35a5a80c33b92.git>

**requirements.txt:** The code below renders the code of the requirements.txt. This file should allow the user to import and install dependencies into the PyCharm environment in order to run the application. If it does not work, the user has other options to install them following the directions on the README.md file. If it doesn't work the user must also install these: random; math; time; json; unittest.

**Code of the requirements.txt file of the essential dependencies include:**

pygame

pyautogui

numpy

Note: All required dependencies were either manually installed using pip or listed in the requirements.txt file for easy setup. The essential dependencies include:

To install, the following command can be used:

*pip install -r requirements.txt*

These dependencies were essential for enabling graphical rendering, UI interaction, audio support, and automation tasks throughout the game experience.

**Git, Unit Testing, and Debugging:** The project will use Git for version control to manage code changes and facilitate collaboration among developers. Each developer will create feature branches for specific tasks, making incremental changes and committing them with descriptive messages. After completing their work, they will push their changes to the remote repository and create a pull request (PR) for code review. The project lead will review and approve the PRs, merging them into the main branch once the code meets the required standards. Unit testing, Manual testing, and GitHub actions will ensure the core game mechanics, user interface, and features like high scores function correctly, while debugging will address any issues in gameplay, UI, performance, and cross-platform compatibility. Regular updates will be made to keep the main branch stable and up-to-date, and any merge conflicts will be resolved promptly to maintain consistency. Note: The testing and Git version control processes are further detailed in APPENDIX D and APPENDIX E. D shows more breakdown on the Testing section while E shows the breakdown of the Git issues/milestones completed for that week. Meanwhile APPENDIX C is referring to the delays and emergencies so far to push back the code.

**Note: Refer to Unit 6 documentation link for more details!**

# **Project Evaluation, Test Plan, and Results**

The test plan documents and results have been key to ensuring that the release of our Python Game Hub project was successful. Carefully stepping through the GUI and associated code with our detailed test plans allowed us to identify and resolve issues that could have made it into the final production. All identified problems were fixed and retested until every test case could be executed with no failures. The finalized test plans, including both manual and automated testing (unit tests and github actions), have been embedded in this document and can be opened by clicking on their links below.

* **Unit 4: Phase 1 Source Report Plan**
  + Unit 4 has 13 unit tests on the old test.py file (mina branch) but with no GitHub actions.
  + This Phase 1 document captures the early development work where core features and basic game functionality were implemented. It typically includes source code implementation details for key components like Mainmenu.py, TicTacToe.py, Breakout.py, and the initial version of Trivia.py. The document reflects the progress made during the foundational stage of building the application.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1oHjphF6c_7OVMOcpK2WWTJcwo7Pe4vee/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit4)
* **Unit 5: Testing Plan**
  + Unit 5 has 20 manual tests (excel), 24 unit tests on the new test.py file (james branch), and 6 (.yml) GitHub Actions.
  + The Testing document contains records of all the testing activities, including manual tests, unit tests, github actions, and validation processes. It demonstrates how each feature was tested to ensure functionality and stability. For this project, it also included completed tests for Trivia.py and validation of the Questions.json file, confirming data accuracy and bug-free implementation.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1-Gexgrs11wsRRzTdWwDQkw0dHNirzS0a/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit5)
* **Unit 6: Phase 2 Source Report Plan**
  + Unit 6 uses the same tests as in Unit 5 where it has 20 manual tests (excel), 24 unit tests on the new test.py file (james branch), and 6 (.yml) GitHub Actions.
  + Unit 6 has 2 new Github Actions testing to validate that the files are on main for deployment preparation.
  + The Phase 2 document details the final stage of feature development. It covers the implementation of advanced functionalities, final code refinements, and rerun of all tests applied. It has new tests like the Github actions and the other tests are from unit 5. It also includes documentation practices such as adding inline code comments, maintaining a detailed README.md, and organizing supporting files. This phase finalizes both the technical and documentation aspects of the project.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit6)
* **Unit 7: User Guide**
  + The User Guide is a user-facing document designed to help end-users understand how to install, launch, and navigate the game hub. It includes setup instructions, game rules, descriptions of each game feature, and links to resources (like the GitHub repo). It’s written in a clear and simple format for non-developers and is a key part of the project’s deliverables.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)
* **Unit 8: Deployment Report**
  + The Deployment Report outlines how the project was packaged and prepared for final submission. It details everything included in the final release—source code, user guide, tests, documentation, and any relevant project files. It ensures that everything has been verified and meets the submission requirements. This report typically includes final reflections, validation checks, and readiness for peer review. Meanwhile, the Google Slide (Video) presentation of this is a visual summary of the entire project, suitable for stakeholders, instructors, or peer reviewers. It highlights the objectives, key phases (Plan, Design, Development, Testing, Deployment), challenges faced, major accomplishments, and a walkthrough of the game hub’s features. It complements the deployment report by offering a high-level overview with visuals and talking points.
    - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1CYzgpDp1lFNmwkbjhaIwnEbEDStKLXPn/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [GitHub (doc) Link] (Check it at the GitHub repository doc folder!)
    - [[Google Slides Link]](https://docs.google.com/presentation/d/10nZwOQwZNqL12yTPeKOJZYSwu5xBDgPzDS-rd9FtDY8/edit?usp=sharing) (Check it at the GitHub repository doc folder or the Unit 8 discussion!)  [[PythonGameHub GitHub Repo Link]](https://github.com/javonpayne100/CMSC495Capstone)

**A brief summary of the overall Testing is below:**

**Manual Testing (CMSC 496 Project Test Plan.xlsx) (Excel) (Unit 5 and 6):** This document contains a total of 20 manual test cases, distributed evenly across the application modules: 5 for the Main Menu, 5 for Trivia, 5 for Tic-Tac-Toe, and 5 for Breakout. Each test entry includes test steps, expected outcomes, actual results, and pass/fail status, ensuring comprehensive validation of user interactions and application behavior. The manual testing plan for the Python Game Hub covers the core components of the application: the main menu interface and three mini-games—Tic Tac Toe, Trivia, and Breakout. Each section is carefully designed to ensure the functionality, responsiveness, and usability of the software from a user perspective. This also accounts for the integration testing and the user-interaction testing. The main menu serves as the central navigation point, so tests focus on verifying that each menu option leads to the appropriate game or action. Visual feedback through hover effects and background music playback are also tested to ensure they enhance user experience. We decided to implement these ones as well as adding more unit tests would be over engineering it since our testers already implemented all core parts.All testing artifacts will be documented within this file [CMSC 495 Project Test Plan.xlsx](https://docs.google.com/spreadsheets/d/1MhQiBjRPqWWcELhFAi9Y0fcVixZCgKGo/edit?usp=sharing&ouid=104007192992096171932&rtpof=true&sd=true) containing detailed specifications for each test case. Each entry will include the expected outcome, actual result with pass/fail status, actions performed during testing, and the specific software module or component under evaluation.

**Unittest (Python file):** During Unit 5 and 6, we have implemented all the possible unit tests and have not implemented any more due to overengineering and repetition. We have imported the Unit 5 Testing to Unit 6. To make up for this, there is now new GitHub Actions and Manual Testing to make up for it.The test.py script for the Python Game Hub is structured to validate a wide range of scenarios across the application’s three core games—Tic Tac Toe, Trivia, and Breakout—as well as the main menu interface. Functionality tests are designed to ensure that each component behaves as expected under normal use. These include verifying correct navigation between menu options, detecting win and draw conditions in Tic Tac Toe, registering block collisions and scoring in Breakout, and managing answer selection and timeouts in the Trivia game. The tests confirm that users are able to interact with the games fluidly, and that appropriate responses (such as displaying results or progressing to the next question) occur in real time. Performance testing ensures that the system remains stable and responsive even under stress or edge cases. These tests also check whether repeated use of sound effects—such as for ball bounces or correct answers—maintains consistent performance without causing lag, audio distortion, or crashes. Additionally, the use of pop-up alerts in Breakout’s game over state is validated for timing and correctness. Usability tests focus on the overall user experience. These tests examine whether visual elements like menu highlights or countdown timers provide clear feedback to the user. “Play Again” options at the end of games are tested to make sure they correctly restart the game or exit without unexpected behavior. Altogether, the tests in test.py provide a thorough quality assurance framework for the Game Hub. They not only validate that the individual games work as intended but also that the overall software provides a consistent, user-friendly experience with reliable performance. These tests are crucial for identifying bugs early and ensuring the software is polished and production-ready.

**GitHub Actions Testing with .yml files (Units 5 and 6) (6 .yml files):** This includes the code to the updated **.yml** files to automate the unit tests for our application. It is highly recommended that you run the test.py following the instructions of the IDE, but this method will also confirm it as a double checker to make sure it works through Git’s (CI/CD) pipeline. GitHub Actions Automated testing was tested and created by Victoria to confirm and validate the unittest of James’s test.py files. There are three GitHub Actions for the old test.py file in the main branch and three GitHub Actions for the new test.py in the james branch (test.yml, test2.yml, test3.yml, testJames.yml, testJames2.yml, testJames3.yml). The first three without James label should test the old test.py on the main branch with 13 unittest passed. The next three with the James label should test the new test.py on the James branch with 24 unittest passed. This should allow our team to double check with Jame’s manual IDE python runs for the test.py files and make sure it is working. The test.py (old), test.py (new), and CMSC 495 Project Test Plan.xlsx were created and tested by James and the GitHub Actions was made by Victoria while integrating it to the branches and test.py files.

**Git (Version Control) (main) branch only (Unit 6):** This was created by Victoria and the test should be in the .github/workflows of files: testFiles.yml and testFiles2.yml. The first one tests if the main branch has all the core game files (detects it’s not missing/exists) for Mac and Ubuntu. The second one tests if the main branch has all the game files (detects it’s not missing/exists) for Windows. It uses Node.js and Git Actions commands(Markdown) to check if the files exists.These two tests ensure that our members when merging did not change the file names or delete/remove it from the main. This ensures that there are no core files missing when prepping for pull/push/merge, and deployment for Unit 8.

**Note 1:** Refer to Unit 4, Unit 5, Unit 6, and README.md documents to see all the testing requirements, process, and debugging! Refer to Unit 7, and 8 if you need help to navigate the applications or need an overview.

**Note 2:** Refer to APPENDIX D and APPENDIX E for Tests and Git. The Milestones and Gnatt chart updates with the Contribution Report are in APPENDIX A and APPENDIX B and APPENDIX E (issues related to tasks/milestones). APPENDIX C is the agreement for delays.

# **Design and Alternate Designs**

**Our Finalized Design Plan (Python Game Hub):**

**Phase 1: Source Report Summary -** Primary designs for the Python Game Hub were developed in a modular and object-oriented approach using the Waterfall methodology. The original system design included a unified Main Menu and three mini-games (Tic-Tac-Toe, Trivia, Breakout), each treated as a separate module. Core diagrams like Use Case, Activity, and Class Diagrams illustrated early program logic and flow. This foundational phase focused on ensuring individual game functionality and graphical interface through Pygame. Unit testing was performed manually via Python’s unittest framework, and version control was managed using GitHub with structured pull request workflows.

**[Link to Phase 1 Source Report]**:

[Phase 1 Source Report](https://gist.github.com/VictoriaRaven/cde9d58550020450900cd8742dbe9b1e)

**Alternative Link (Google Drive):**

[Phase 2 Source Report](https://docs.google.com/document/d/1oHjphF6c_7OVMOcpK2WWTJcwo7Pe4vee/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)

**Phase 2: Source Report Summary -** Building on Phase 1, Phase 2 finalized and optimized all aspects of the Game Hub. The project integrated GitHub Actions for automated unit testing (CI/CD pipelines), expanded unit testing coverage, and completed manual testing with a comprehensive Excel-based test plan. The finalized version included improved error handling, a polished Main Menu with hover and click functionality, and strong game modules with features like AI behavior, score tracking, and timed gameplay. Minor changes were made to fix usability and flow issues, ensuring a production-ready, user-friendly experience.

**[Link to Phase 2 Source Report]**:

[Phase 2 Source Report](https://gist.github.com/34c40d49bcef96c70ed35a5a80c33b92.git)

**Alternative Link (Google Drive):**

[Phase 2 Source Report](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)

**Note: These parts below is extra incase you need some more design outlook for the “overall application”**

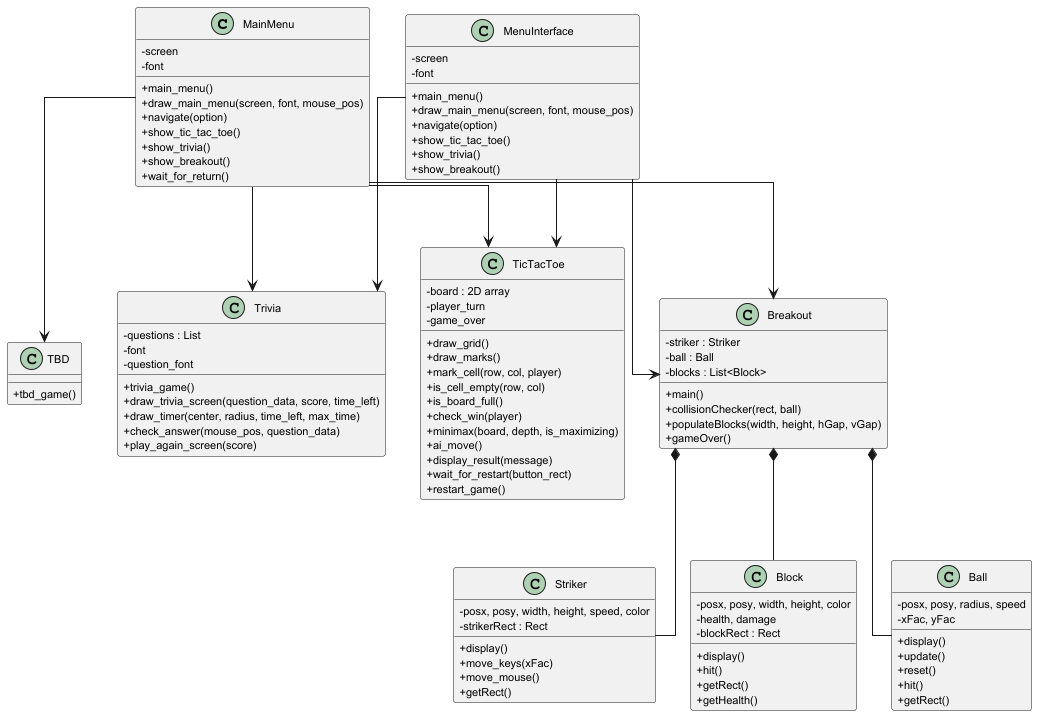
**Below is the Project Design Plan** that demonstrates the overall UML diagrams for the applications so far in Phase 1.

**[Link to Project Design]:**

[Project Design Plan](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit3)

**Alternative Link (Google Drive):**

[Project Design Plan](https://docs.google.com/document/d/1hjNWfBtiacfR40g9wAcIkf-xypBD3PlF/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)



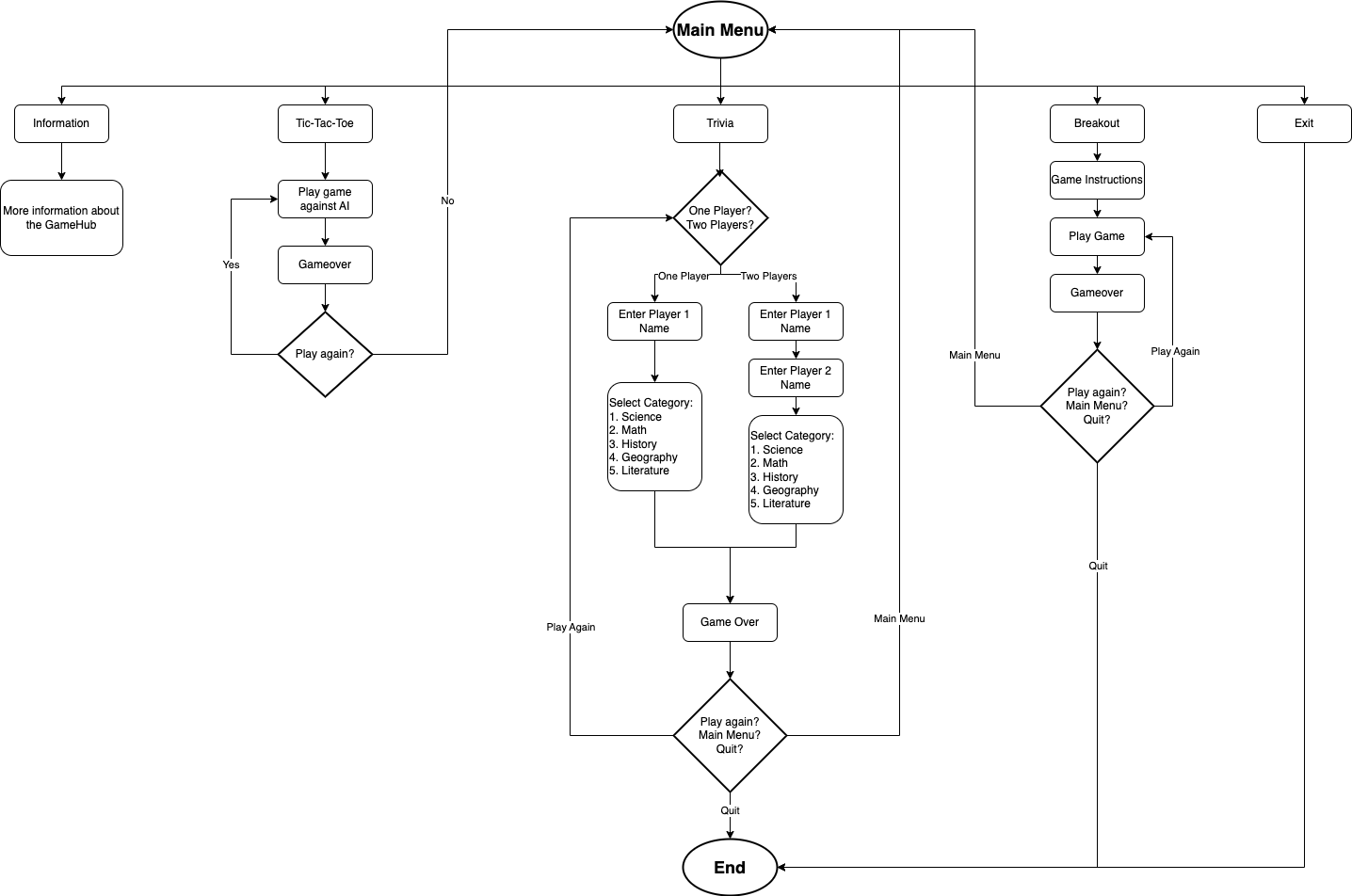
**Below is the User Guide** that demonstrates these changes from Phase 1, phase 2, and in preparation for the Deployment (Unit 8). This should include a UI Flow Chart Diagram.

**[Link to User Guide]:**

[User Guide](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)

**Alternative Link (Google Drive):**

[User Guide](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)



**Information About Our Team’s Updates:** In **Unit 6**, the team refactored the Python Game Hub code to improve organization, readability, and maintainability. Large blocks of logic were broken into smaller, reusable components, repeated code was removed, and UI was separated from game logic to simplify testing. Each game—TicTacToe, Breakout, and Trivia—was cleaned up, with improvements like a separate question manager for Trivia and better collision detection in Breakout.

For **deployment**, the team ensured easy setup by creating a clear README.md, adding a requirements.txt, and allowing the game to launch from MainMenu.py. The project is hosted on GitHub with documentation, test plans, and diagrams organized in folders like /docs/ and .github/workflows/.

In the **maintenance** phase, final bug fixes, UI enhancements, and feature upgrades were completed. Code was cleaned, commented, and standardized. All games were polished—Trivia got better scoring and timers, TicTacToe’s AI was refined, and Breakout was made more responsive. The GitHub repo is now ready for future contributions or updates.

**Documentation Summary:** The project meets all documentation requirements with clear inline comments, docstrings, and a detailed README.md covering setup, gameplay, and development info. Additional resources like UML diagrams, test plans, and version logs are stored in /docs/ and .github/workflows/ for easy access and maintenance.

**Outlook:** The team may submit early by May 3, 2025, allowing time for any final fixes before the May 6 deadline. The GitHub repo remains publicly active and well-documented for future contributions. With modular code, clear setup via requirements.txt, and CI testing through unittest and GitHub Actions, the project is ready for long-term growth and use in future development or portfolios.

**Note:** Refer to Unit 4: Phase 1 Source Report and Unit 6: Phase 2 Source Report, which discusses the changes before deployment for more details

# **Development History and Deployment**

Based on the timeline (Units 1-8), the Project Plan and Design Plan were essential in our Python Game Hub project as they provided a clear roadmap and structure, helping the team stay organized, meet deadlines, and ensure all features were well thought out before coding began. Meanwhile, the Development Phases (Phase 1 and Phase 2) ensured all core and advanced features were implemented and refined. The Testing plan confirmed functionality and reliability across all games, while the User Guide made the project accessible for users. Finally, the Deployment Report and Presentation ensured everything was ready, reviewed, and clearly communicated for final submission including all links to each section. Based on our development history

**Process Model:** This team group will be following the SDLC process model to help develop and deploy the Python Game Hub. This is a waterfall methodology that allows the team to plan in the early stages to prevent major design flaws from developing and below is the SDLC (GeeksforGeeks, n.d.) According to GeeksforGeeks (n.d.) and UMGC (n.d.), the Waterfall model is a linear and sequential approach, where each phase must be completed before the next one begins. In the case of the Python Game Hub, the Waterfall methodology allows for a structured, step-by-step approach to deliver each component of the project on time because the features and games are clearly defined from the outset. **As of now for Unit 8 our team will use all parts of the Waterfall method, making sure all parts are completed to submit the documentation and code.** The testing and Git version control processes are further detailed in APPENDIX D and APPENDIX E. The Milestones and Gnatt chart updates with the Contribution Report are in APPENDIX A, APPENDIX B, and APPENDIX E. A shows the overall timeline due dates and how our progress compares with units 1-8. B shows the breakdown of each task/milestones completed and costs. APPENDIX C is referring to the delays and emergencies so far to push back the code. APPENDIX E shows the breakdown of the Git issues/milestones completed for that week (screenshots).



**Project Plan Timeline (Graphs and dates, Gnatt chart):**

A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

**Project Timeline (based on updates[Phases 1 & 2]):**

* **Unit 2: Project Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1F2LR0j118Oje_9Gw5iaabmx6HMe-YYtb/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit2)
* **Unit 3: Project Design Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1hjNWfBtiacfR40g9wAcIkf-xypBD3PlF/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit3)
* **Unit 4: Phase 1 Source Report Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1oHjphF6c_7OVMOcpK2WWTJcwo7Pe4vee/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit4)
* **Unit 5: Testing Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1-Gexgrs11wsRRzTdWwDQkw0dHNirzS0a/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit5)
* **Unit 6: Phase 2 Source Report Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit6)
* **Unit 7: User Guide**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)
* **Unit 8: Deployment Report**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1CYzgpDp1lFNmwkbjhaIwnEbEDStKLXPn/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [GitHub (doc) Link] (Check it at the GitHub repository doc folder!)
    - [[Google Slides Link]](https://docs.google.com/presentation/d/10nZwOQwZNqL12yTPeKOJZYSwu5xBDgPzDS-rd9FtDY8/edit?usp=sharing) (Check it at the GitHub repository doc folder or the Unit 8 discussion!)  [[PythonGameHub GitHub Repo Link]](https://github.com/javonpayne100/CMSC495Capstone)

**Note:** Refer to Unit 4, Unit 6, and Unit 8 to compare the project timelines, job contributions, Git (Version Control), Testing, and Changes to the Project Management sections. It is crucial to view the APPENDIX sections:

* The testing and Git version control processes are further detailed in APPENDIX D and APPENDIX E. The Milestones and Gantt chart updates with the Contribution Report are in APPENDIX A, APPENDIX B, and APPENDIX E.
* A shows the overall timeline due dates and how our progress compares with units 1-8. B shows the breakdown of each task/milestones completed and costs.
* APPENDIX C is referring to the delays and emergencies so far to push back the code.
* APPENDIX E shows the breakdown of the Git issues/milestones completed for that week (screenshots).

### **Source Code Deployment (GitHub):** The deployment of the **Python Game Hub** marks the final stage of the development process. The goal is to make the software easily accessible to users across different platforms while ensuring stability, performance, and ease of use. Our team decided to deploy and host the application online through a public GitHub repository link to our Capstone Project.

**Python Game Hub must be Publicly Accessed on GitHub:**

* **GitHub Repository Link:** [[CMSC495Capstone-Python Game Hub]](https://github.com/javonpayne100/CMSC495Capstone)
* Read the [README.md](https://github.com/javonpayne100/CMSC495Capstone#cmsc495capstone-python-game-hub-group-1) OR [User Guide](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
* Follow the instructions on the User guide / README.md so that users can clone the repository or download the ZIP and follow the instructions to install dependencies and launch the game.
* **Note 1:** Our team decided to not do the executable packaging to any (.exe) files or use a “release package” as it was complicated and not in the user guide instructions.
* **Note 2:** Refer to Unit 6 Document for more details about the deployment and verification.

### **Hosting Options:**

* **Local/Offline Distribution:**
  + After the user downloads all the source files from the repository:
  + The user can share or use it via USB drive, Dropbox, Google Drive, or GitHub release ZIP files.

**Post-Deployment Considerations:**

* **Bug Reporting:** Users are encouraged to report bugs or issues via the GitHub Issues tab.
* **User Feedback Loop:** Collect feedback through surveys or GitHub discussions for continuous improvement.

**Note:** Refer to Unit 6 Document for more details about the deployment and verification.

# **Discussion and Target Audience**

**Lessons Learned:**

Through the development of the Python Game Hub, our team learned the importance of early planning, clear communication, and modular coding practices. One major takeaway was that breaking the project into smaller, manageable parts made it easier to identify issues early and allowed multiple people to work independently yet still integrate their work smoothly. We also realized the value of detailed documentation and thorough testing. These steps helped catch bugs early and made the final polishing process much faster.

**Design Strengths:**

One of the major strengths of our project was its modular structure. Each mini-game (Tic Tac Toe, Trivia, and Breakout) was developed as an independent module, which made debugging and updating specific features more efficient. Our use of a Main Menu to unify the experience also provided a smooth and consistent user flow. Additionally, we maintained strong version control practices with GitHub, supported by unit testing and manual testing, which helped ensure the reliability and stability of the codebase across different operating systems.

**Project Limitations:**

While the project achieved its main goals, there were still some limitations. First, more extensive automated testing could have been added, especially for UI interactions, although we avoided over-engineering the testing phase. Second, the design did not prioritize mobile responsiveness or very high-resolution displays, limiting the application primarily to standard desktop environments. Finally, integrating sound and visuals consistently across all operating systems posed challenges, particularly during GitHub Actions workflows where display and audio drivers were not available.

**Suggestions for Future Improvement:**

For future iterations, we recommend expanding the Game Hub by adding more games and creating a more dynamic and responsive UI. Incorporating database support could allow for features like saving high scores or user progress. We also suggest improving cross-platform compatibility further and expanding the automated testing suite to include integration tests that simulate user interactions. Finally, more frequent code reviews and smaller, quicker sprints would help catch integration bugs even earlier in the process although, we as a team were doing this quite well already.

# **Conclusion**

The Python Game Hub project successfully demonstrated the practical application of the Software Development Life Cycle (SDLC) through a structured, collaborative team effort. By following the Waterfall methodology, we were able to systematically plan, design, implement, test, and deploy a fully functional gaming platform that integrates three mini-games—Tic Tac Toe, Trivia, and Breakout—into a single cohesive application.

Throughout the course of this project, the team overcame challenges related to cross-platform compatibility, game logic refinement, user interface design, and automated testing. Each member contributed to coding, documentation, testing, and debugging, ensuring that the final product met both functional and non-functional requirements. Rigorous unit testing, manual testing, and GitHub Actions integration reinforced the stability and reliability of the application across various operating systems.

The Game Hub’s modular architecture, detailed user guide, and extensive testing infrastructure provide a strong foundation for future enhancements, such as the addition of new games, online multiplayer functionality, or more advanced graphical features. Additionally, the project documentation—including the project plan, design documents, testing artifacts, and deployment strategy—ensures maintainability and scalability for future developers.

Ultimately, the Python Game Hub project not only resulted in the successful delivery of a fully functional software product but also strengthened the team’s experience with professional software development practices, collaborative workflows, and project management techniques. The lessons learned from this project will serve as valuable assets as team members advance into their future academic, professional, and technical endeavors.

# **Acknowledgements**

**As a Team**, we would like to take this moment to express our heartfelt **thanks to our teammates** who helped bring this project to life. The **Python Game Hub** wouldn’t have been possible without the dedication, collaboration, and expertise of each team member. All of the gratitudes are below:

**Victoria Lee** ([GitHub: VictoriaRaven](https://github.com/VictoriaRaven)): As a Project Manager / Technical Writer / Tester (GitHub Actions / Repo), I want to express my deepest gratitude to my amazing team for their unwavering dedication and hard work throughout this project. The way we came together, blending our skills, strengths, commitment,and teamwork like in the *Avengers*, a Marvel movie, made this experience truly special (Whedon, 2012). I also want to extend a heartfelt thank you to my peers and family, whose suggestions helped shape the final product. Your support, both on and off the project, was invaluable and truly appreciated. Thank you and as always, “Not all treasure is silver and gold, mate.” - Captain Jack Sparrow (Pirates of the Caribbean) (Verbinski, 2003).

**James Mutry** ([GitHub: jamesmutry](https://github.com/jamesmutry)): As a Technical Writer / Tester (Unit Tests & Manual Tests), I would like to take this opportunity to sincerely thank my entire team for their dedication, hard work, and collaboration throughout the Python Game Hub project. Each member brought their own unique skills and strengths, contributing to the success of this challenging and rewarding experience. From initial planning to final deployment, our team consistently supported one another, tackled obstacles together, and maintained a strong commitment to producing a polished and functional final product. I am especially grateful for the positive attitude, adaptability, and professionalism that everyone demonstrated during each phase of the project. Working alongside such a talented and motivated group made this capstone experience not only educational but truly enjoyable. I am proud of what we accomplished together and thankful to have had the opportunity to collaborate with such an outstanding team.

**Javon Payne**  ([GitHub: javonpayne100](https://github.com/javonpayne100)): As a Git Lead (.py files) / Developer (MainMenu & Trivia) / Sub Tester), I want to give a huge thanks to my team for bringing me on board with open arms. It’s a blessing to have a team with diverse skills, restless drive, and great character. This experience has definitely taught me a lot and this type of camaraderie should be the standard of many development teams. I want to especially thank Victoria; her organization skills and leadership is unmatched. Congratulations everyone, graduation is here! “May the force be with you.” - Yoda

**Todasha Foster** ([GitHub: DayFoster123](https://github.com/DayFoster123)): As a Developer (Breakout) / Sub Tester, I would like to express my sincere appreciation to my team members for their energy, creativity, and collaboration throughout this project. Contributing as a developer was a rewarding experience that allowed me to strengthen my skills and grow professionally. I am proud of the challenges we overcame together and the final products we created. This achievement reflects the passion, effort, and dedication of every member of the team. "I never dreamed about success. I worked for it." – Estée Lauder

**Dajin Chung** ([GitHub: dajinchung](https://github.com/dajinchung)): As a Developer (TicTacToe) / Sub Tester, I’d like to thank my entire team for their hard work and collaboration throughout this project. Each member contributed their strengths, from development and testing to design and documentation, to bring this project to life. A special thanks to Victoria, who truly stepped up as project manager. Her organization, communication, and consistent support helped keep our team focused and on track. We would not have reached this level of polish and completion without her leadership. Everybody, thank you for all your hard work. As a wise ogre once said, “That’ll do Donkey. That’ll do.” - Shrek

**Oluwatumininu Ipaye** ([GitHub: tumiwiththewave](https://github.com/tumiwiththewave)): As a Developer (Breakout) / Sub Tester, I want to express my deepest gratitude to the entire team for their hard work and resilience to get this project done. I have been able to learn a lot from each single member and effectively learnt how to work in a group setting such as this. My skills have been boosted and challenged at the same time, which shows that being a coder/programmer is a never ending lesson. I would like to specially thank Victoria Lee, as she was the backbone of this entire project and ensured everything went smoothly and without any hitches. I am proud of what we were able to accomplish in a short period, and wish you all the best ahead. *The sky’s the limit.*

# **References**

Amazon Web Services. (n.d.). *What is unit testing?* Amazon Web Services, Inc.<https://aws.amazon.com/what-is/unit-testing/>

Deitel, P. J., & Deitel, H. (2015). *Java: How to program (Early objects)* (10th ed.). Pearson. Online textbook available. (n.d.)

Devaraj, K. (2024, December 19). 6 key phases of testing in software testing. *Testsigma*.<https://testsigma.com/blog/phases-of-testing/>

Doe, J., & Lee, A. (2020). Creating effective contribution reports in software projects: Guidelines and strategies. *IEEE Software Engineering Journal*, 45(6), 102–115.<https://doi.org/10.1109/SEJ.2020.3082548>

Erickson, J. (n.d.). *Algorithms*. University of Illinois at Urbana-Champaign. Online textbook available.

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

GeeksforGeeks. (2024). *Control Flow Software Testing*.<https://www.geeksforgeeks.org/control-flow-software-testing/>

GeeksforGeeks. (2024). *Software engineering: Differences between coupling and cohesion*.<https://www.geeksforgeeks.org/software-engineering-differences-between-coupling-and-cohesion/>

GeeksforGeeks. (2024). *Difference between Function-Oriented Design and Object-Oriented Design*.<https://www.geeksforgeeks.org/difference-between-function-oriented-design-and-object-oriented-design/>

GeeksforGeeks. (2024). *Object-Oriented Design (OOP)*.<https://www.geeksforgeeks.org/oops-object-oriented-design/>

GeeksforGeeks. (2024). *Functions decomposition in software engineering*.<https://www.geeksforgeeks.org/functions-decomposition-in-software-engineering/>

GeeksforGeeks. (2024). *What is a Project Plan*.<https://www.geeksforgeeks.org/project-plan/>

Gehry, F. (Producer). (2019, April 10). *How to make your design stand out* [Video]. YouTube.<https://www.youtube.com/watch?v=Va64slVDpI4>

Git. (n.d.). *Git user manual*.<https://git-scm.com/docs/user-manual>

Jones, M., & Davis, L. (2019). Effective project design planning in software engineering: Best practices and strategies. *IEEE Software Engineering Conference Proceedings*, 31(2), 95–110.<https://doi.org/10.1109/SEConf.2019.2787093>

Khan, S., & Patel, A. (2018). Designing and implementing a centralized game hub for multiplayer platforms. *IEEE Transactions on Game Development*, 39(1), 77–89.<https://doi.org/10.1109/TGD.2018.2876062>

Kleinberg, J., & Tardos, É. (2006). *Algorithm design*. Pearson Education.

LeetCode. (n.d.). *LeetCode - The world's leading online programming learning platform*. LeetCode. Retrieved April 12, 2025, from<https://leetcode.com/>

Lehman, E., Leighton, F. T., & Meyer, A. R. (2021). *Mathematics for computer science*. LibreTexts, MIT, & UC Davis.

Liang, Y. D. (2023, May 23). *Introduction to Java programming and data structures* (13th ed.). Pearson. Online textbook available. (n.d.)

Mount, D. (n.d.). *Algorithms [Lectures CMSC 251]*. University of Maryland Global Campus.

Nievergelt, J. (n.d.). *Algorithms and data structures*. University of Maryland Global Campus.

Sierra, K., & Bates, B. (2005). *Head First Java* (1st ed.). O'Reilly Media.

Smith, J., & Taylor, R. (2021). Using Git for version control in Python-based game development. *IEEE Transactions on Software Engineering*, 47(3), 215–230.<https://doi.org/10.1109/TSE.2021.3056789>

Solomon, J. (2015). *Numerical algorithms*. MIT.

Stellman, A., & Greene, J. (2005). Software project planning. In *Applied software project management* (Part I, Chapters 5, 6, 8, 9, 10, 11). O'Reilly Media.<https://go.oreilly.com/umgc/https://learning.oreilly.com/library/view/applied-software-project/0596009488/ch08.html>

Sweigart, A. (n.d.). *Making games with Python & Pygame*.<https://inventwithpython.com/makinggames.pdf>

Tsui, F., Karam, O., & Bernal, B. (2014). *Essentials of software engineering* (3rd ed.). Jones and Bartlett Learning.<https://library-books24x7-com.ezproxy.umgc.edu/toc.aspx?site=VGX8U&bookID=51648>

University of Maryland Global Campus. (n.d.). *Sample Project Plan*.<https://leocontent.umgc.edu/content/dam/course-content/tus/cmsc/cmsc-495/document/2020%20Sample%20Project%20Design%2004-2022.pdf?ou=1248245>

Unity Technologies. (2020, September 3). *Testing and quality assurance tips for Unity projects*.<https://unity.com/how-to/testing-and-quality-assurance-tips-unity-projects>

Verbinski, G. (Director). (2003). *Pirates of the Caribbean: The Curse of the Black Pearl*. Buena Vista Pictures.

Whedon, J. (Director). (2012). *The Avengers*. Walt Disney Studios Motion Pictures.

Winters, T., Manshreck, T., & Wright, H. (2020). Software engineering at Google (Chs. 18–19). O'Reilly Media.<https://learning.oreilly.com/library/view/software-engineering-at/9781492082781/ch18.html>

# **APPENDIX A:**

Notes: This includes Overall tasks, milestones, hours, and cost (for unit 1-8) and progression.

Refer to APPENDIX B for a contribution report for specific Tasks, Milestones, Issues Completed.

Refer to APPENDIX E for Git Tasks, Milestones, Issues Completed.

A Large Table Gantt Chart repeat of Project Plan in Unit 4:

| Week | Dates | Lead | Topic | Description | Due Date | Assignments Due |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 3/10-3/14 | Everyone  And Project Manager | Team Formation  And staring on the Project Plan | Get to know team members & pick a project  -Outline Milestones  -Delegate responsibility  -Describe project's purpose | 3/14 | N/A |
| 2 | 3/15-3/18 | Documentation  And Requirements | Project Plan | -Create a formal Project Plan   * Testing Units   Updating Git | 3/25 | Project Plan |
| 3 | 3/18-3/28 | Documentation  And Developers | Design | -Create application structure  - Develop user interface/functionality  -Create UML diagrams   * Testing Units   Updating Git | 4/1 | Project Design |
| 4 | 3/28-4/4 | Everyone | Phase 1 Source | Software Development   * Testing Units   Updating Git | 4/8 | Phase 1 Source  Peer Review 1 |
| 5 | 4/4-4/11 | Everyone | Testing | * Software Development | 4/15 | Test Plan |
| 6 | 4/11-4/18 | Everyone | Phase 2 Source | * Software Development * Testing Units * Updating Git | 4/27 | Phase 2 Source  (Prepare for Peer Review 2 on week 8) |
| 7 | 4/18-4/25 | Everyone | User Guide and Phase 2 | * Software Development * Testing Units * Updating Git * User Guide documentation and submission | 4/27 | User Guide  and Phase 2 Source (Prepare for Peer Review 2 on week 8) |
| 8 | 4/25-5/2 | Everyone | Final Report | * Compile all Topics into a single document * Testing Units * Updating Git | 5/4 | Final  Peer Review 2 |

Below is a bigger picture version of this:

A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

### **Overall Project Status and Progress Update:**

This report provides an updated analysis of our project’s progress, highlighting the completion status of key project phases, including the project plan, design, testing, and upcoming milestones. Based on the current progress, we estimate the project is **80% complete**, with testing being a key focus in the current week.

We have made substantial progress so far. Here’s an overview of what has been completed, the current phase, and what is left to finish:

#### **Completed Phases (80% complete):**

* **Project Plan**: Fully completed, outlining key tasks, milestones, and deadlines.
* **Project Design**: All design elements and documentation are complete.
* **Phase 1**: The development phase, where all initial features and functionalities were implemented, including core game functionality.
* **Testing:** Fully completed manual and unit testing of the application and the implemented features. The tests are being executed on the project. WE also completed **Trivia.py and Questions.json**, which is part of the game suite for Unit 5.
* **Phase 2:** The last development phase, where all new features and functionalities were implemented, including core game functionality. And all the tests are implemented. And all the documentation best practices are fulfilled. (Code comments, Readme.md file, documentations, etc.)
* **User Guide:** Team will create a Simplistic User Guide for the Python Game Hub project and will work together to prepare for the final stage Unit 8 next week and peer review. This will provide comprehensive instructions for using the software.
  + During Week 7, our team will work together since there is no more code to be implemented and complete the user guide. This document will cover how to set up and play the games, as well as detailed descriptions of each feature. There is no more code developing or testing this week, but documentation as a whole team.

#### **Upcoming Tasks (Remaining 10%):**

1. **Deployment**: The project will be deployed in the final phase. This will include submitting the **User Guide** and the final source code (all implemented game files, tests, etc.). This includes finishing and completing the peer review 2 for Unit 8.

### **Feature Completion**

Here is a breakdown of the current status of the project features and the remaining work needed:

Note: We will not implement a score saver as doing it in real time, updating it through the leaderboard was complicated, as most of the servers that needed to store it online require premium access, which we preferred to be free only (no costs). We also did not have enough time to implement due to the (2 weeks left), and the additional features were more significant than storing real-time-updated data scores on a hosted server. Thus, that scoring idea on a hosted real-time-updated server is removed due to the time constraints for the last two weeks, and would rather be spent on implementing/completing the additional features.

| **Feature** | **Current Status** | **Remaining Tasks** | **Progress** |
| --- | --- | --- | --- |
| **Mainmenu.py** | Completed and fully implemented. | No remaining tasks.  -Updated buttons, UI, updated look, and transitions.  - Updated information button to allow user a hyperlink to Github repository for the README.md page for information/history. | 100% |
| **TicTacToe.py** | Completed and fully implemented. | No remaining tasks.  -Updated AI/CPU/transition, music | 100% |
| **Breakout.py** | Completed and fully implemented. | No remaining tasks.  -Updated scores/lives, including transition, music | 100% |
| **Trivia.py** | Completed and fully implemented. | Finish the implementation for **Unit 5 and 6** and include tests.  -Updates 1-2 player mode and also scores, music, and subject questions.  -Wrap the text around the screen and fix questions.json for symbols issues. | 100% |
| **Unit Testing (all games)** | Comprehensive testing of all games with both manual and unit tests.  Completed and fully implemented | Complete tests for **Trivia.py** and implement all tests (unittest and manual tests based on the Excel sample) | 100% |
| **Phase 2 (all parts from unit 1-6)** | Completed and fully implemented | Complete Phase 2  Completed and fully implemented for Trivia.py Questions.json and other minor issues. | 100% |
| **User Guide (Unit 7)** | Completed and fully implemented. | Completed and fully implemented. Team worked together on the User Guide and Preparation for Deployment Phase Unit 8. | 100% |

### **Current Deployment Week and Plan:**

As we are currently in **Deployment Week**, the main focus is on ensuring the correctness and reliability of the implemented features, testing, documented, installation, dependencies, and all three systems work to be submitted in a finalized document and zip file codes:

* **Project Deployment**: During Week 8, once the documentation is complete from Week 7, the final source code, along with the user guide and other documentation, and the peer review 2 will be submitted for deployment. This week, our team will work together to ensure that the documentation and coding files are working and complete each rubric. There is no more code developing or testing this week, but documentation and submitting the final project as a whole team.

### **Project Completion Timeline**

| **Week** | **Task** | **Team Completion Percentage** |
| --- | --- | --- |
| **Week 1** | Create and Form Teams (Brainstorm) | 100% |
| **Week 2** | Project Plan completed (Start Base Code for Python) | 100% |
| **Week 3** | Project Design completed (Continue Developing Mainmenu.py, TicTacToe.py, Breakout.py, Trivia.py) (Brainstorm a README.md and implement Git Version Control and Brainstorm Unit Tests) | 100% |
| **Week 4** | Phase 1 (Implementing features of Mainmenu.py, TicTacToe.py, Breakout.py, Trivia.py; Implement Git Version Control, and start out Unit Tests; implement README.md to prep install and run) | 100% |
| **Week 5** | Testing Week (Finish Unit testing and Manual Testing based on the Excel sample and also finish implemented features (Breakout.py; TicTacToe.py; and MainMenu.py features are already finished, waiting on Trivia.py)) | 100% |
| **Week 6** | Phase 2 (Testing and working on Documentation: Making sure installation works, etc) (Mainmenu.py, TicTacToe.py, Breakout.py, Trivia.py) | 100 % |
| **Week 7** | User Guide Documentation and Start Deployment | 100 % |
| **Week 8** | Deployment: Finish Docs, Code, Finalize Application | 100 % |
| **Overall Progress** |  | **100 % Complete** |

### **Total Progression**

The project is currently **100% complete**, with significant milestones already reached. Most of the core functionality, including the main games and their associated files, has been implemented and is undergoing testing. The remaining tasks involve finalizing and ensuring all **documents from past Units (1-7) are inside the final deployment documentation (Unit 8) submission including source codes.** By working together during the last week 8, our team will complete the remaining documentation and prepare the project for submission. All of the codes will be reviewed, and the project will be thoroughly tested to ensure that it meets all requirements.

# **APPENDIX B:**

CONTRIBUTION REPORT

===============================================================

Project: Python Game Hub

Notes: This includes tasks, milestones, hours, and cost.

Refer to APPENDIX E for Git Tasks, Milestones, Issues Completed.

| TASKS COMPLETED / CONTRIBUTION LOG | | | |  |
| --- | --- | --- | --- | --- |
| DATE | TASKS/MILESTONES | NAME OF VOLUNTEER | HOURS | COST |
| 3/10/2025 to 3/14/2025 | Created at team for project | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~5 hour | $0 |
| 3/15-3/17 and 3/18/2025 | Worked and completed Unit 1 and 2 Discussion and Replies | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3 hours per person | $0 |
| 3/15-3/17 and 3/18/2025 | Worked on Project Plan (3/11-3/17);  Turned in Project Plan (3/18) | Lee, Victoria;  Mutry, James  Chung, Dajin | ~6-8 hour | $0 |
| 3/18/2025 to 3/26/2025 | Worked on Project Design Plan (3/18-3/26) (delayed by 2-4 days for not rushing and lots of time);  Turned in Project Design Plan (3/27/2025) | Lee, Victoria;  Mutry, James | ~6-8 hour | $0 |
| 3/16/2025 to 3/22/2025 | **Contribution (Dajin Chung):** Created the foundational code for a playable 1v1 Tic Tac Toe, Assisted with Breakout Game background display **Milestones:** Base grid display, turn based logic **Challenges:** Grid refresh bugs on re-click; addressed with a redraw method, Breakout game background flickering **Resolution:** Bug fixed and committed to Git.  Contribution (Todasha Foster): Spent 1 hour coding click sounds for player interactions with Xs and Os | Chung, Dajin  Foster, Todasha | ~3 hours  1 hour |  |
| 3/16/2025 to 3/22/2025 | Worked on the Breakout portion of the game hub. Contribution (Todasha Foster): Spent one hour coding game sound effects for Breakout. Implemented audio for the ball hitting blocks, the ball hitting the paddle, and the player losing the game. Implemented a condition to loop the game back to the main menu.  **Contribution (Dajin Chung):** Implemented player vs AI turn structure, Added basic win condition checks, used pygame.draw methods to render player marks of X and O dynamically **Milestones:** Turn management and player input handling, Drawing X/O marks in correct grid square, board matrix integration for logic evaluation, initial win/draw detection functionality  **Challenges/Issues:**Clicks on board did not register correctly, early bug where could overwrite an existing move **Resolution:** added bound checks to prevent overwrites, tested and verified board accuracy after each turn | Ipaye, Oluwatumininu;  Foster, Todasha.  Chung, Dajin | ~7 hour ~1 hour  ~2 hour | $0 |
| 3/16/2025 to 3/26/2025 | Worked on the Main Menu portion and Contribution (Todasha Foster): Spent 1 hour coding the main menu sound to play when the screen loads and stops once the player navigates away.  Worked on base game, MenuGame Hub, and combine the games into the main through Git pull/push (~ 7 hour) | Foster, Todasha;  Payne, Javon; | ~1 hour  ~ 7 hours | $0 |
| 3/16/2025 to 3/22/2025 | Worked on the Trivia portion. Contribution (Todasha Foster): Expanded the background graphics to enhance the visual appeal of the gameplay environment. Spent 1 hour coding background music to automatically play during gameplay and stop when the game ends. Also, added crowd reaction sound effects, cheering sound to trigger upon a win and an "aww" sound to play after a loss.  Worked on tic tac toe functions/logic 1hr | Foster, Todasha;  Chung, Dajin;  Javon | ~1 hour | $0 |
| 3/20/2025 to 3/26/2025 | Debugged other mini games to make game hub work when combining to the menu (~1 hour) | Payne, Javon;  Lee, Victoria; | ~1 hour | $0 |
| 3/16/2025 to 3/22/2025 | Testing Games work/debugging for help (~1 hour) | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 1 hour | $0 |
| 3/26/2025 to 3/31/2025 | Worked and completed Unit 3 Discussion and Replies  **Contribution(Dajin Chung):** Assisted with Trivia game logic  **Milestones:** Trivia menu buttons, return to menu logic  **Challenges:** Trivia game ends abruptly; addressed with refactoring method logic  **Resolution:** Bug fixed | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3 hours per person | $0 |
| 3/29/2025 to 4/5/2025 | Worked on Phase 1 Source Reporting and submitting it on 4/4/2025 | Lee, Victoria;  Mutry, James; | ~8 hours | $0 |
| 3/29/2025 to 4/5/2025 | Worked on Peer Review Evaluations, completed, and turned it in before or on Tues. | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3-4 hours per person | $0 |
| 3/29/2025 to 4/5/2025 | Worked on Unit 4 discussions, due Tuesday | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3-4 hours per person | $0 |
| 3/29/2025 to 4/5/2025 | Worked on Additional Features to the Three Games and testing them out  Worked on Making the games transition back to the main menu hub (Not finished for some and will continue this for unit 5). [If the game ends, make it go back to the main menu instead of exiting.] | Payne, Javon;  Ipaye, Oluwatumininu  Foster, Todasha;  Chung, Dajin; | ~ 7 hours | $0 |
| 3/29/2025 to 4/5/2025 | Working on creating and tested out the Unit Test in the test.py file with comments. It has the code for the unit testing. (~ 7 hours)  Tested out the Unit Testing. (~1 hour) | Mutry, James;  Lee, Victoria; | ~ 7 hours  ~ 1 hour | $0 |
| 3/29/2025 to 4/5/2025 | Worked on creating and testing out the README.md file based on the requirements and meeting the professor's feedback (email).  [Explains how to set up; how to run; how to play; how to do the unit test; project’s progress; git;]  Worked on creating the media folder and adjusting the other files to move the sounds/music. | Lee, Victoria; | ~ 7 hours | $0 |
| 3/29/2025 to 4/5/2025 | Worked on adding the GitHub Code Links to the document with a short description. Worked on fixing the Git Version control on the documentation (~2 hours)  Tested Links, updated it, and checked documentation updates (~ 1 hour) | Payne, Javon;  Lee, Victoria; | ~ 3 hours  ~ 1 hour | $0 |
| 4/5/2025 to 4/11/2025 | Worked on Test Plan and submitting it on 4/12/2025 | Lee, Victoria;  Mutry, James; | ~8 hours | $0 |
| 4/5/2025 to 4/11/2025 | Worked on the Unit Tests on new test.py, making changes to the files on james branch, testing the new unit tests, creating manual tests, and updating the excel sheet.  Write on parts 5, 6, 9 of the Test Plan | Mutry, James; | ~ 16 hours | $0 |
| 4/5/2025 to 4/11/2025 | Worked on the GitHub Actions for all three systems (Mac, Win, Linux) with the old test.py and new test.py  Worked on writing / completing all Test Plan sections, except 5, 6, 9.  Worked on updating the README.md file | Lee, Victoria; | ~ 16 hours | $0 |
|  | Assisted Victoria to test out the GitHub Actions on their ends even though Victoria tested it out on her side and it works. This is for all 6 yml files (old test.py and new test.py) | Ipaye, Oluwatumininu  Mutry, James; | ~ 1 hour | $0 |
| 4/5/2025 to 4/11/2025 | Worked on Unit 5 discussions, due Tuesday | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3-4 hours per person | $0 |
| 4/5/2025 to 4/11/2025 | Worked on completing Trivia.py, and Questions.json and including testing the features. All games already completed 100%.  Merged Questions.json into main through upload. | Payne, Javon; | ~8 hours | $0 |
| 4/5/2025 to 4/11/2025 | Worked on helping expand the Tasks, Milestones, and Job Contribution on the Test Plan  Worked on helping Javon and Jame’s fix Mac issues with breakout.py | Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 8 hours | $0 |
| 4/11/2025 to 4/18/2025 | Worked on Phase 2 Source Code Report for Unit 6 and submitting it on 4/19/2025 | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~8 hours per person | $0 |
| 4/11/2025 to 4/18/2025 | Worked on Unit 6 discussions, due Tuesday | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3-4 hours per person | $0 |
| 4/11/2025 to 4/18/2025 | Worked on debugging the MAC os issue on Breakout.py  Javon and James helped detect the issue which was the pyautogui.alert issue before breakout.py starts.  Todasha helped rewrite the code for MAC | Mutry, James;  Payne, Javon;    Foster, Todasha; | ~8 hours per person | $0 |
| 4/11/2025 to 4/18/2025 | Worked on fixing (changing minor code for) the text wrap issue for Triva.py and the math symbols not showing up through the Questions.json file. (UTF-8 or some symbols unsupported through pygame.)  This peron also tested it out so it works. | Payne, Javon; | ~8 hours | $0 |
| 4/11/2025 to 4/18/2025 | Worked on fixing README.md file and making sure it is easy to intall and run. Makes sure it breaks down both old and new test.py unittests and manual tests onto the readme.md file.  Created the Information hyperlink button on the MainMenu.py and fixed screen sizes and captions on the other .py games. | Lee, Victoria | ~16 hours | $0 |
| 4/18/2025 to 4/25/2025 | Worked on User Guide for Unit 7 and submitting it on 4/26/2025 | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~16 hours per person | $0 |
| 4/18/2025 to 4/25/2025 | Worked on Unit 6 discussions, due Tuesday | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 3-4 hours per person | $0 |
| 4/25/2025 to 5/2/2025 | Worked on Deployment for Unit 8 and submitting it on 5/3/2025 | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~16 hours per person | $0 |
| 4/25/2025 to 5/2/2025 | Worked on Unit 8 discussions, due Tuesday  Worked on Peer Review 2, due Tuesday | Lee, Victoria;  Mutry, James;  Foster, Todasha;  Chung, Dajin;  Payne, Javon;  Ipaye, Oluwatumininu | ~ 6 hours per person | $0 |

**Expansion on the Contribution Report for the tasks, milestones, not completed, completed, hours, cost, etc:**

**Refer to Git for more proof of milestones, tasks, and issues on APPENDIX E.**

3/10/25 – 3/14/25: Team created for final project. Project ideas shared and discussed, decisions made for language to be used and meeting times, and tasks delegated to each member. Time spent totaled roughly five hours, costing zero dollars.

3/15/25 – 3/18/25: Unit 1 and 2 discussions completed. Replies made to other classmates to collaborate and encourage success on their respective projects. Time spent totaled roughly three hours per member, costing zero dollars.

3/15/25 – 3/18/25: Project Plan created and edited to meet design rubrics. The Project Plan gives an overview of the chosen project, and goes into detail on the goals, objectives, and deliverables. Time spent totaled roughly six to eight hours, costing zero dollars.

3/18/25 – 3/26/25: Project Design Plan created and edited to meet design rubrics. The Project Design Plan is similar to the Project Plan and also includes risks associated with the project, as well as an evaluation plan and the schedule showing the timeline and/or milestones. Time spent totaled roughly six to eight hours, costing zero dollars.

3/16/25 – 3/22/25: Dajin created base code to be used for the Tic-Tac-Toe game. Base code to be expanded on with functions and refinements. Time spent totaled roughly two hours, costing zero dollars.

3/16/25 – 3/22/25: Oluwatumininu created the base code for Breakout and added functionality to the code. Todasha created and implemented sound effects for various actions to be performed when running the code. Time spent totaled roughly eight hours, costing zero dollars.

3/16/25 – 3/22/25: Dajin worked further on the Tic-Tac-Toe code functionality and logic. Javon created base code for the Trivia game. Todasha created and implemented sound effects for actions performed when running the code. Time spent totaled roughly one hour, costing zero dollars.

3/16/25 – 3/26/25: Javon created base code for the main menu hub that ties in all three games to be created. Games were combined into the main menu through Git pull/push. Todasha created and implemented sound effects for various actions performed when running the code. Time spent totaled roughly eight hours, costing zero dollars.

3/20/25 – 3/26/25: Victoria and Javon worked on debugging the mini games to ensure proper functionality when they are combined with the main menu. Time spent totaled roughly one hour, costing zero dollars.

3/16/25 – 3/22/25: The entire team worked on testing out the games and their features, as well as debugging issues. Time spent totaled roughly one hour, costing zero dollars.

3/26/25 -3/31/25: Unit 3 discussion and replies to other classmates completed. Time spent totaled roughly three hours per person, costing zero dollars.

3/29/25 – 4/5/25: Victoria and James worked on the Phase 1 Source Code and edited the file to meet the design rubrics. The Source Code entails the project’s development environment, functionalities and algorithms used, unit tests to verify correctness of the code modules, and documentation to enhance readability. Time spent totaled roughly eight hours, costing zero dollars.

3/29/25 – 4/5/25: Peer Review Evaluation forms filled out and turned in by due date. Time spent totaled roughly three to four hours per person, costing zero dollars.

3/29/25 – 4/5/25: Unit 4 discussion and replies to other classmates completed. Time spent totaled roughly three to four hours per person, costing zero dollars.

3/29/25 – 4/5/25: Additional features (if feasible) are created for the mini games and tested out. Function to return back to the main menu from any game added to enhance user experience. Time spent totaled roughly seven hours, costing zero dollars.

3/29-25 – 4/5/25: Victoria worked on the README.md file to meet design rubrics and feedback received from the professor. The README.md file is crucial as it documents the contents and structure of our project in the scenario that certain information needs to be located by a researcher. Media folder was also created to easily locate files for sounds or music for in-game actions. Time spent totaled roughly seven hours, costing zero dollars.

3/29/25 – 4/5/25: Javon and Victoria worked on Git version control to be added to the Phase 1 Source Code documentation. GitHub code links were also added to the document with short descriptions for easier access to the repository if needed. Links were tested and updated to ensure correctness. Documentation updates checked out for accuracy. Time spent totaled roughly four hours, costing zero dollars.

4/5/2025 – 4/11/2025: Victoria Lee worked on the Test Plan and submitted it on 4/11/2025.

4/5/2025 – 4/11/2025: James Mutry worked on the Unit Tests in the new test.py, made changes to files on James' branch, tested new unit tests, created manual tests, updated the Excel sheet, and wrote parts 5, 6, and 9 of the Test Plan (~8 hours, $0).

4/5/2025 – 4/11/2025: James Mutry worked on GitHub Actions for Mac, Windows, and Linux using both the old and new test.py, completed all sections of the Test Plan except parts 5, 6, and 9, and updated the README.md (~16 hours, $0).

4/5/2025 – 4/11/2025: Victoria Lee assisted in testing GitHub Actions across all six .yml files, even though her local tests passed (~16 hours, $0).

4/5/2025 – 4/11/2025: Oluwatumininu Ipaye and James Mutry assisted Victoria with testing GitHub Actions across all systems (~1 hour, $0).

4/5/2025 – 4/11/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye worked on Unit 5 discussions, due Tuesday (~3–4 hours per person, $0).

4/5/2025 – 4/11/2025: Javon Payne completed work on Trivia.py and Questions.json, tested all features, and confirmed all games were completed 100%; he also merged Questions.json into the main branch (~8 hours, $0).

4/5/2025 – 4/11/2025: James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye contributed to expanding the Tasks, Milestones, and Job Contributions in the Test Plan, and helped resolve Mac issues with breakout.py (~8 hours, $0).

4/11/2025 – 4/18/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye Worked on Phase 2 Source Code Report for Unit 6 and submitted it on 4/19/2025. (~8 hours per person, $0)

4/11/2025 – 4/18/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye Worked on Unit 6 discussions, due Tuesday. (~3–4 hours per person, $0)

4/11/2025 – 4/18/2025 James Mutry, Javon Payne, and Todasha Foster Worked on debugging the Mac OS issue on breakout.py. Javon and James detected the issue, which was related to pyautogui.alert before breakout.py starts. Todasha helped rewrite the code for Mac compatibility. (~8 hours per person, $0)

4/11/2025 – 4/18/2025: Javon Payne Worked on fixing the text wrap issue in Trivia.py and resolving math symbols not displaying correctly from Questions.json (UTF-8 or Pygame symbol support issue). The fix was tested successfully. (~8 hours, $0)

4/11/2025 – 4/18/2025: Victoria Lee Updated README.md to ensure it is clear and easy to follow. Added breakdowns for both old and new test.py unit tests and manual test instructions. Created the Information hyperlink button on the MainMenu.py and fixed screen sizes and captions on the other .py games. (~16 hours, $0)

4/18/2025 – 4/25/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye Worked on User Guide for Unit 7 and submitted it on 4/26/2025. (~16 hours per person, $0)

4/18/2025 – 4/25/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye Worked on Unit 7 discussions, due Tuesday. (~3–4 hours per person, $0)

4/25/2025 – 5/2/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, Oluwatumininu Ipaye Worked on Deployment for Unit 8 and submitted it on 5/3/2025. (~16 hours per person, $0)

4/25/2025 – 5/2/2025: Victoria Lee, James Mutry, Todasha Foster, Dajin Chung, Javon Payne, and Oluwatumininu Ipaye Worked on Unit 8 discussions (due Tuesday) and Peer Review 2 (due Tuesday). (~6 hours per person, $0)

**Signed: Lee, Victoria; Mutry, James; Foster, Todasha; Chung, Dajin; Payne, Javon; Ipaye, Oluwatumininu**

**Date: 3/10/2025 - 5/3/2025**

# **APPENDIX C:**

DELAY AGREEMENT / EMERGENCIES (PUSH BACK MILESTONES):

===============================================================

Project: Python Game Hub

Agreement:

All team members agree that if any project or weekly milestones/tasks are not completed as scheduled, they will be pushed back by a period of 1-2 days. This delay will apply unless otherwise agreed upon by the team, with consideration for any unforeseen circumstances. During this period of delay, all members can also swap roles to complete the tasks within the delay time period. After the extra period, the tasks must be completed with no excuses.

Emergencies and Exceptions:

In cases of emergencies or other excusable events, the delay in completion of milestones/tasks may be longer. Each team member is expected to communicate any such issues in advance to ensure that an appropriate delay period is decided upon. Examples of acceptable delays may include, but are not limited to: personal emergencies, technical difficulties, illness, or unexpected external factors that impede progress.

Weekly Meetings:

To mitigate delays and issues, weekly meetings will be held to address any ongoing challenges, ensure clear communication, and provide an opportunity for team members to discuss progress and potential risks. These meetings will serve as a platform for verbal communication, enabling the team to stay aligned and address any emergent issues quickly, preventing further delays. Discord and Git Control will also help mitigate these issues as well.

Acknowledgment:

As of **5/3/2025**, our team has not made any delays and completed each unit within a week.

The following team members acknowledge and agree to the delay terms as outlined above:

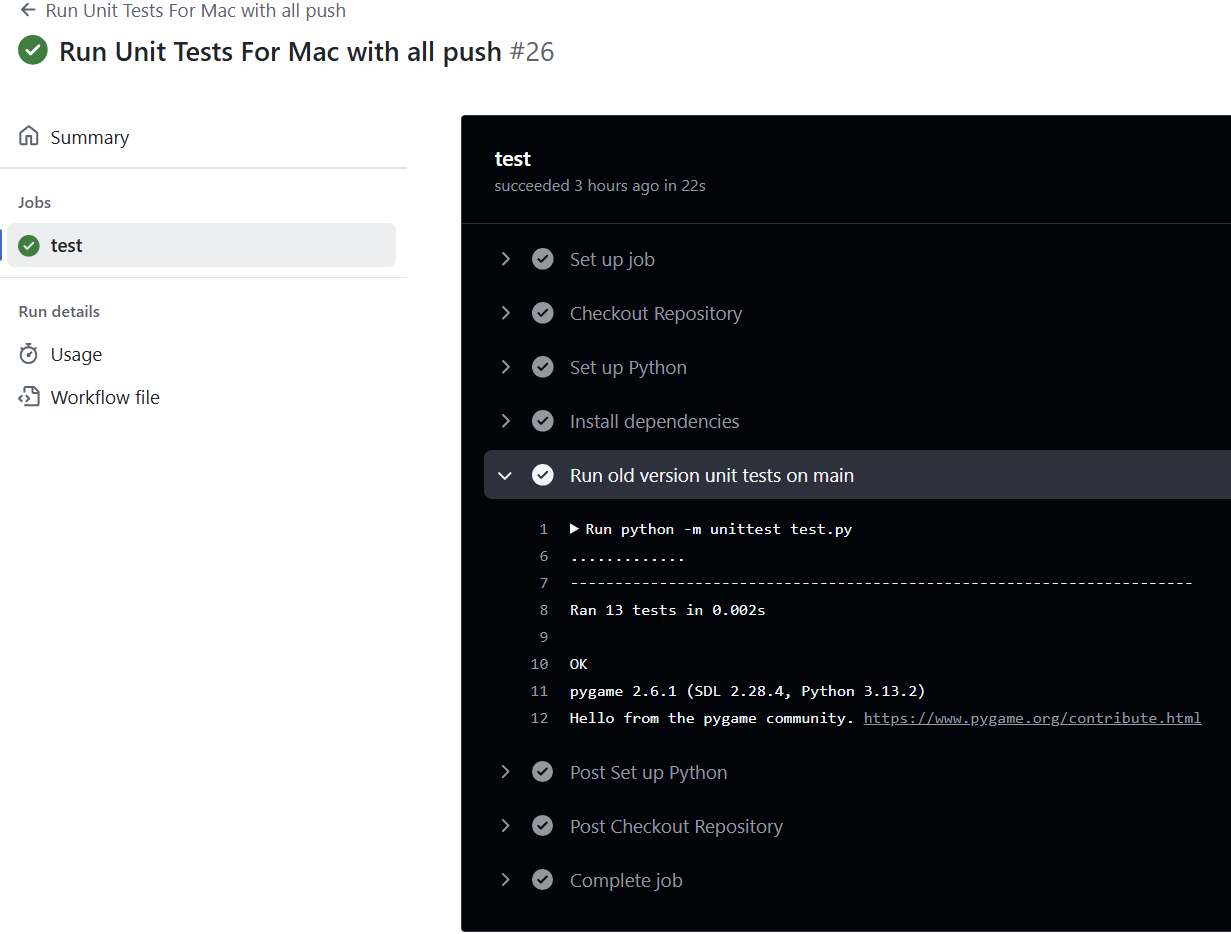
**Signed:  
Lee, Victoria; Mutry, James; Foster, Todasha; Chung, Dajin; Payne, Javon; Ipaye, Oluwatumininu**

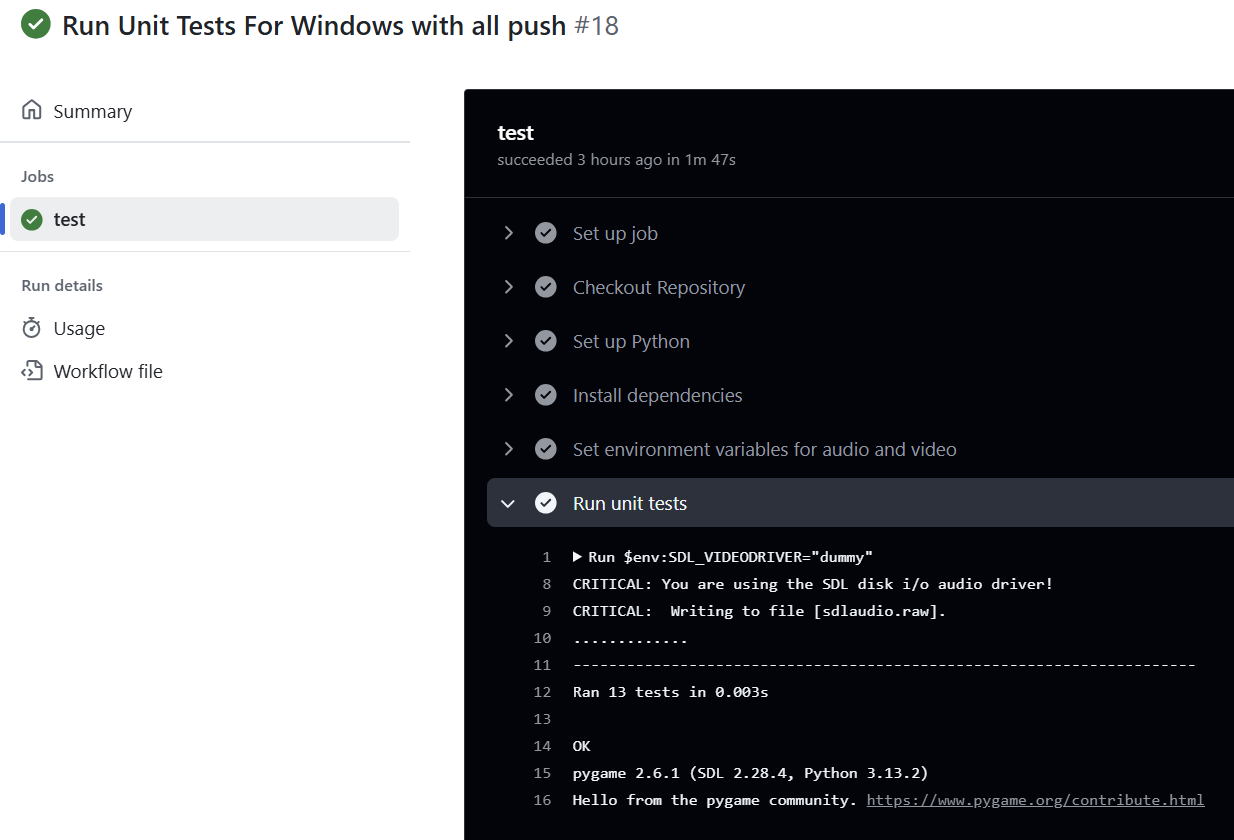
**Date: 3/18/2025 - 5/3/2025**

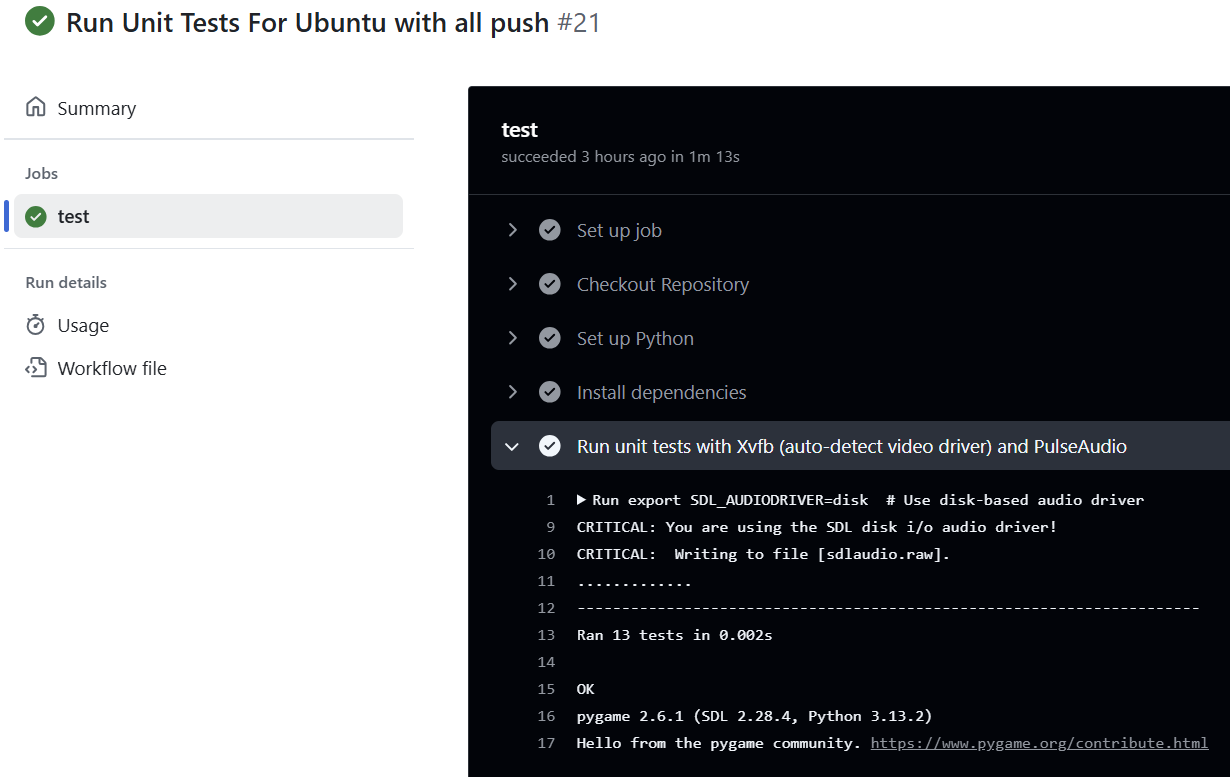
# **APPENDIX D:**

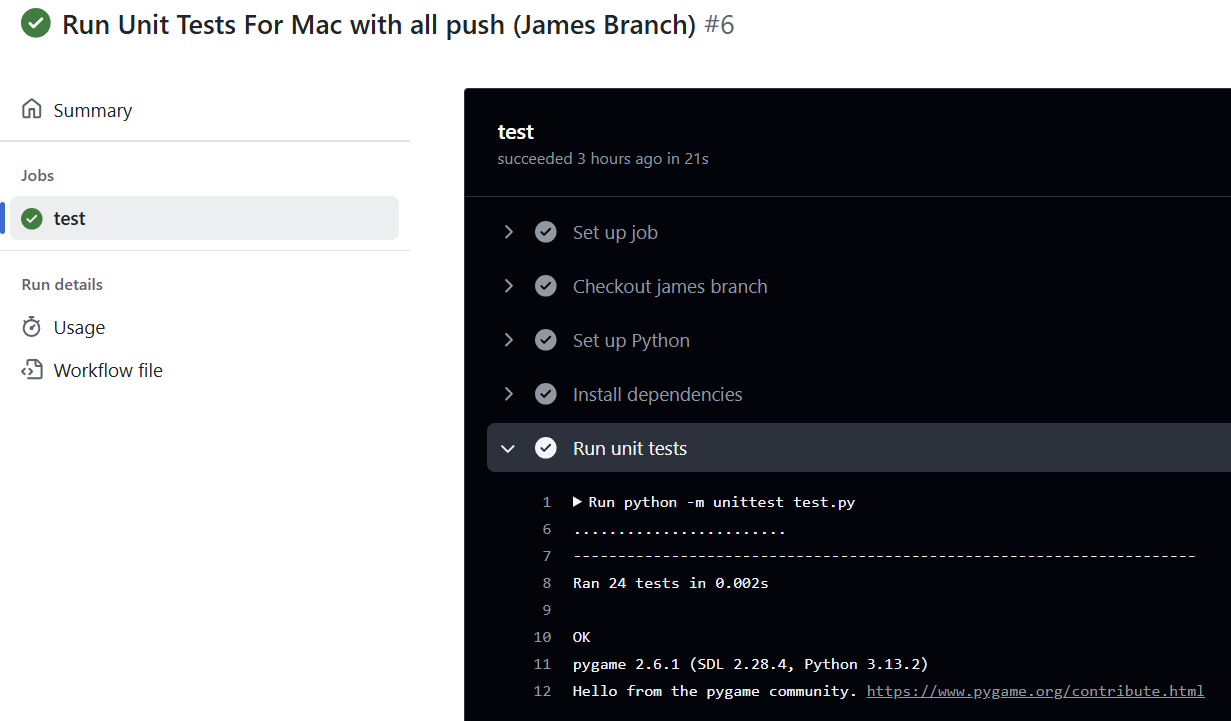
**Continued Screenshots for GitHub Actions working for all 6 unit tests:**

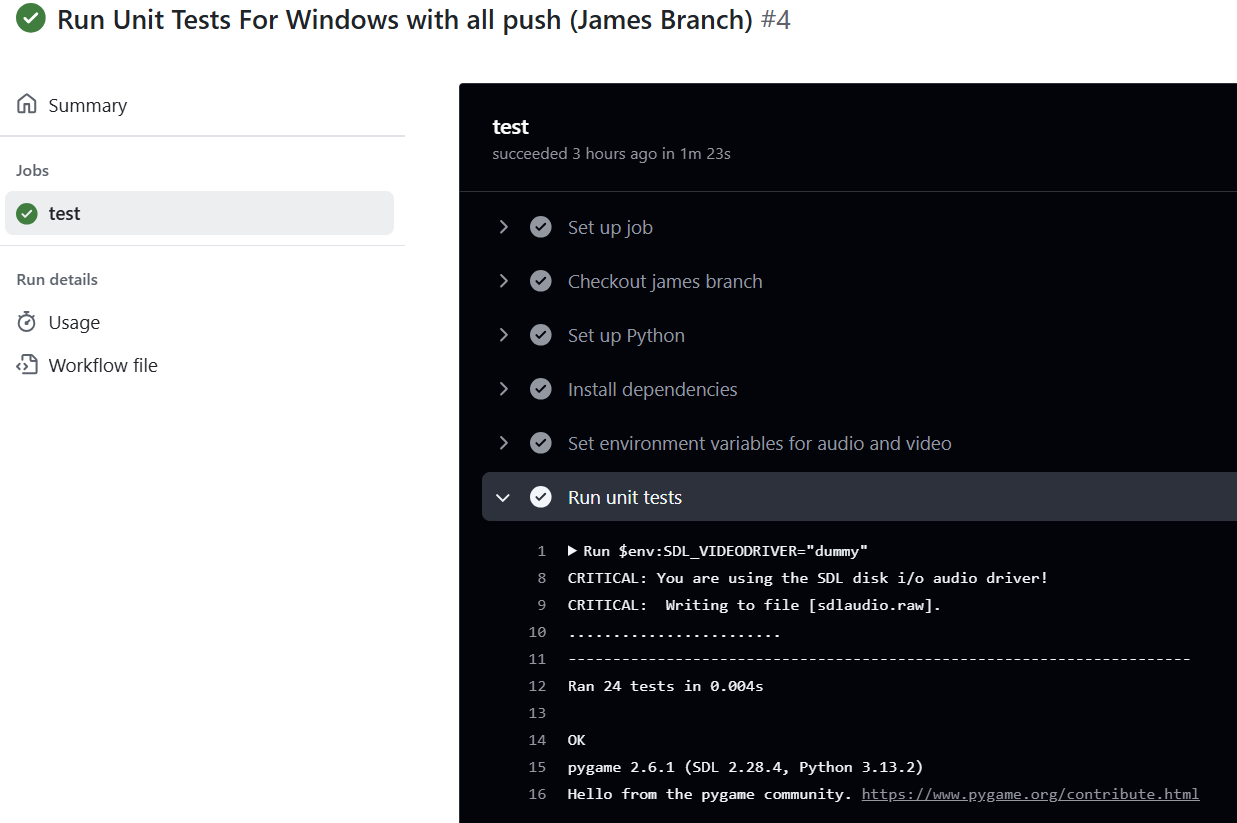
**(I placed it here as the document was becoming long if I didn’t put it in the APPENDIX)**

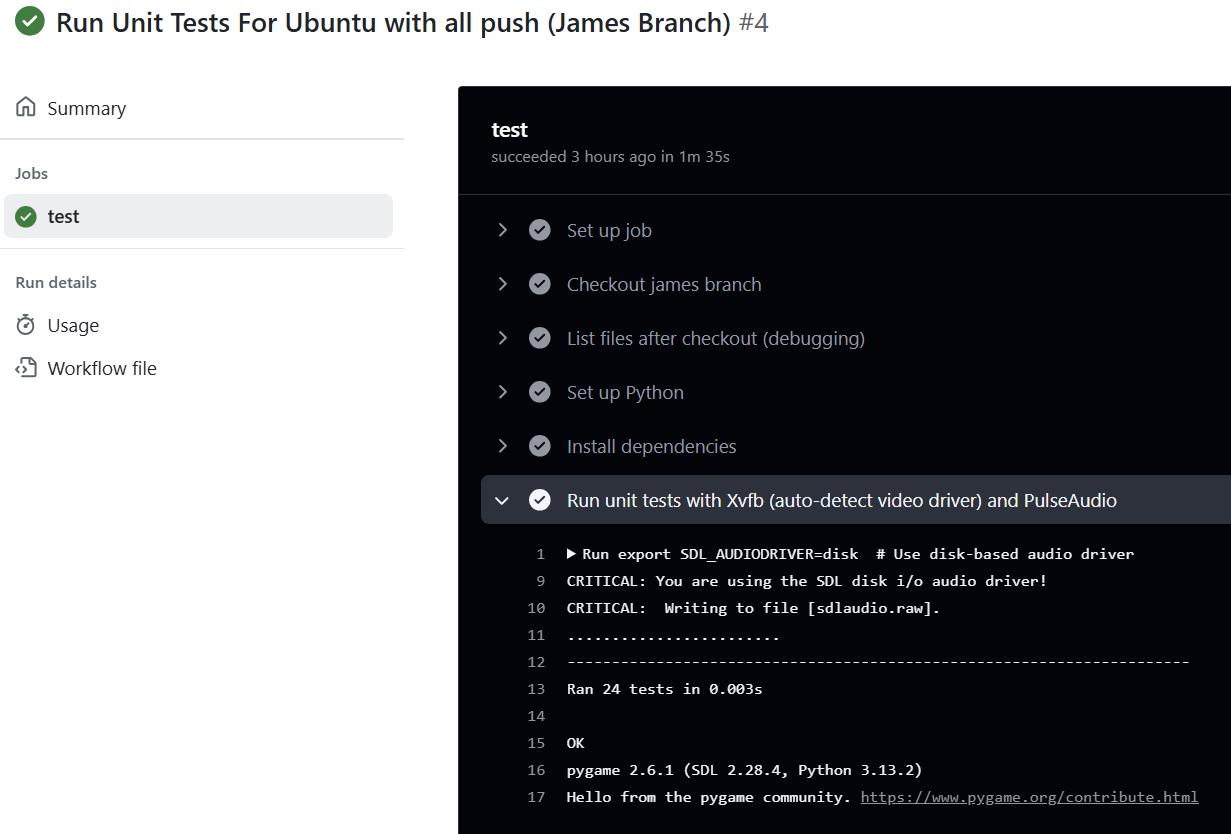
****

****

****

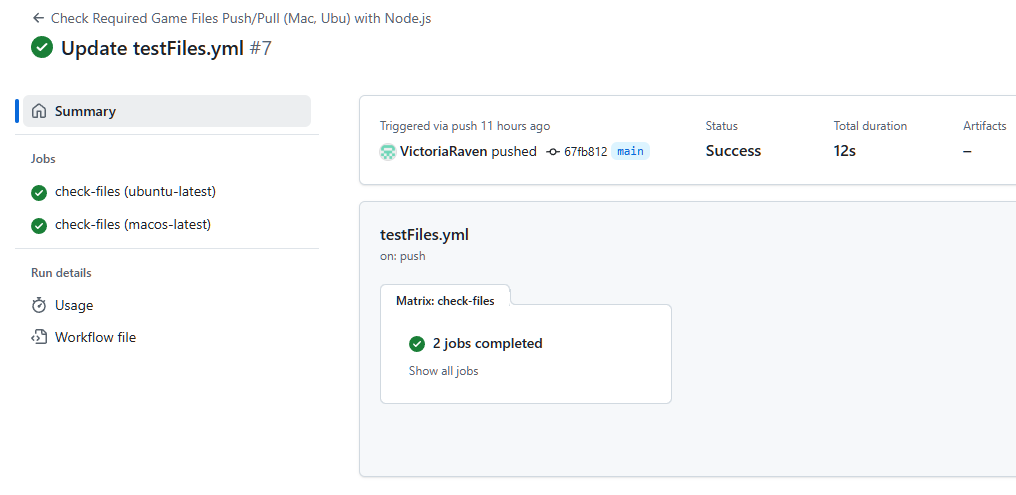
****

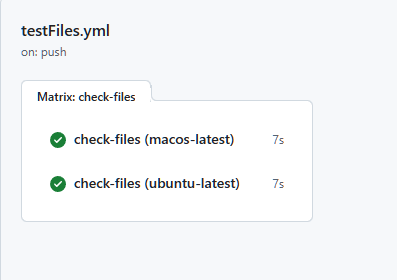
****

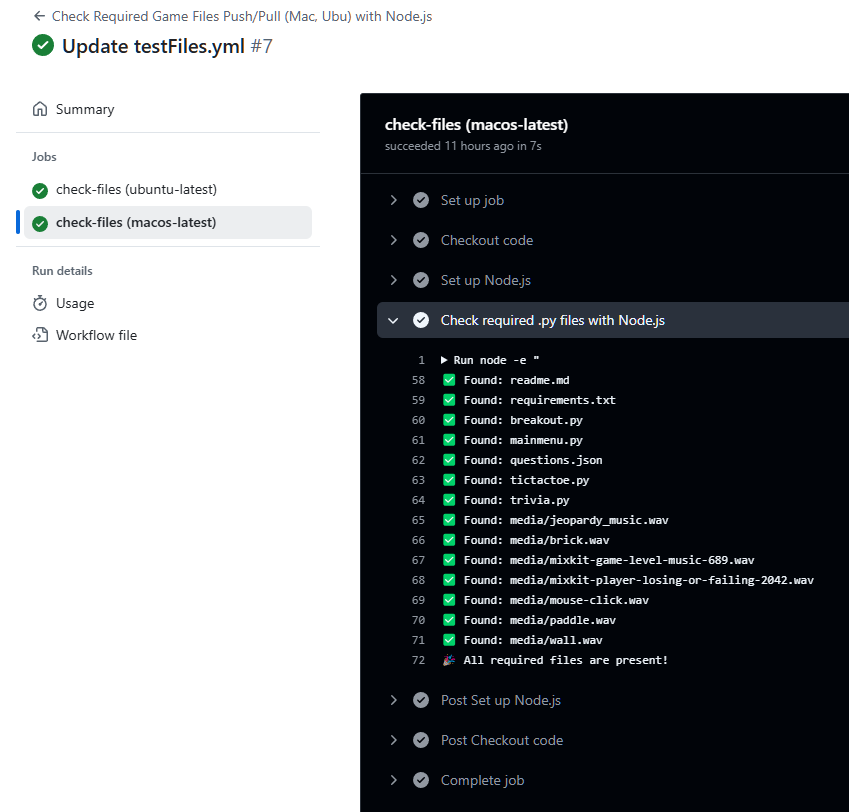
****

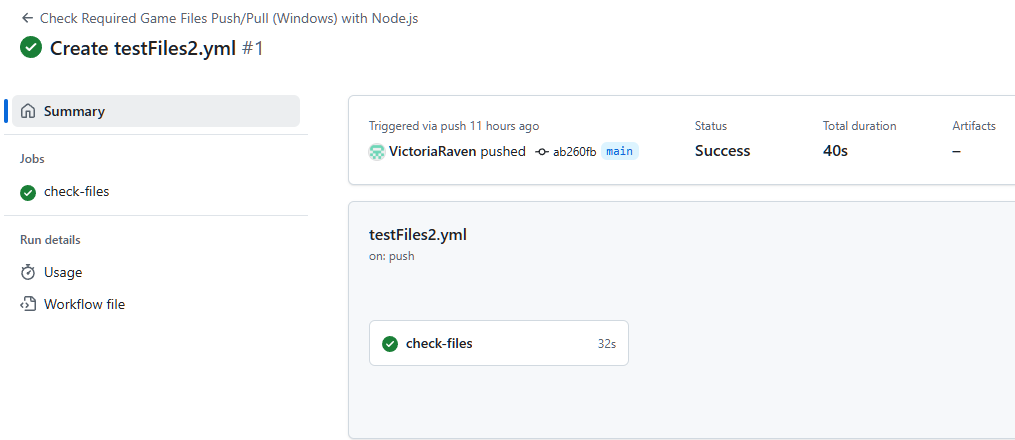
**Continued Screenshots for GitHub Actions working for all 2 .github/workflows testFiles:**

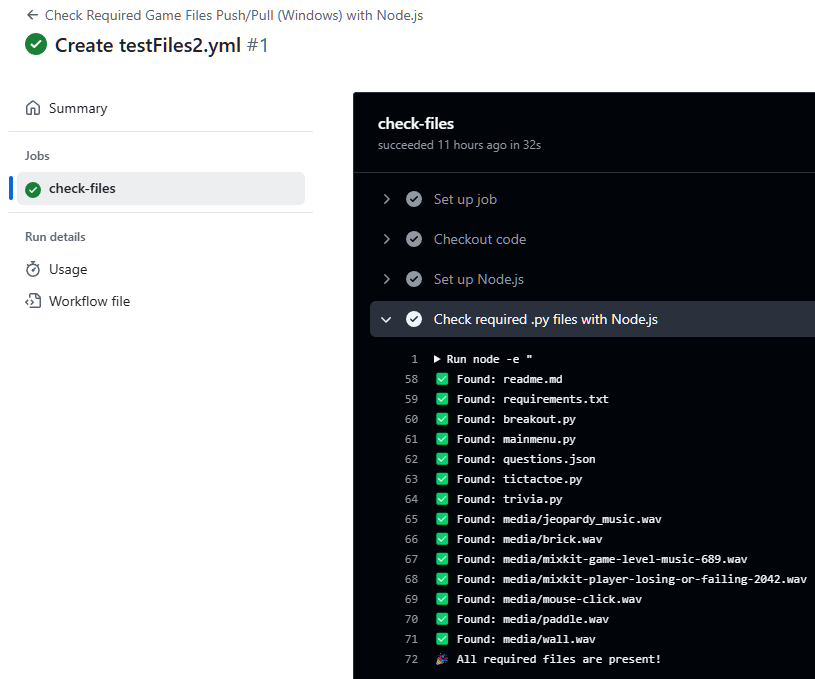
**(I placed it here as the document was becoming long if I didn’t put it in the APPENDIX)**

****

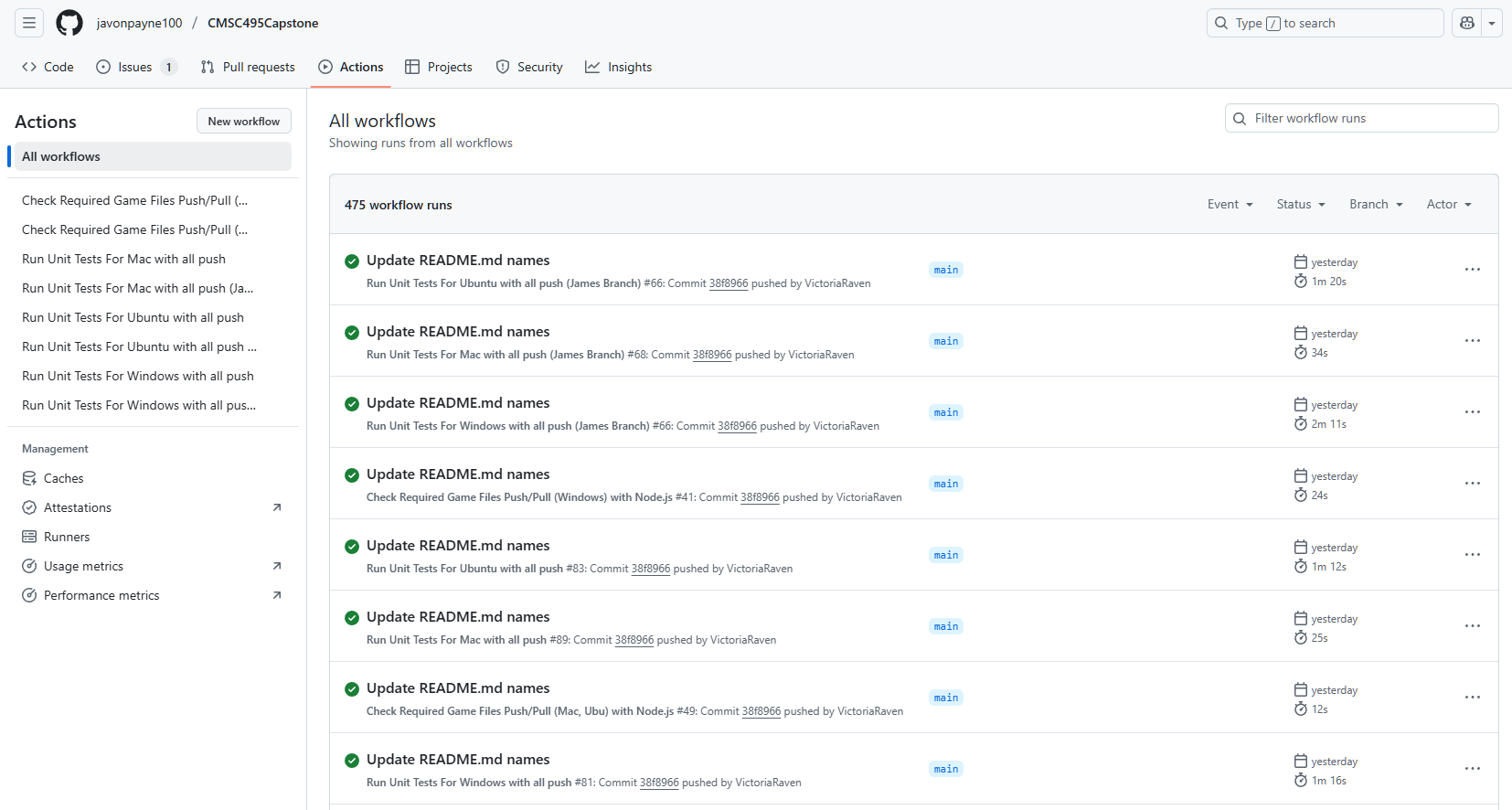
****

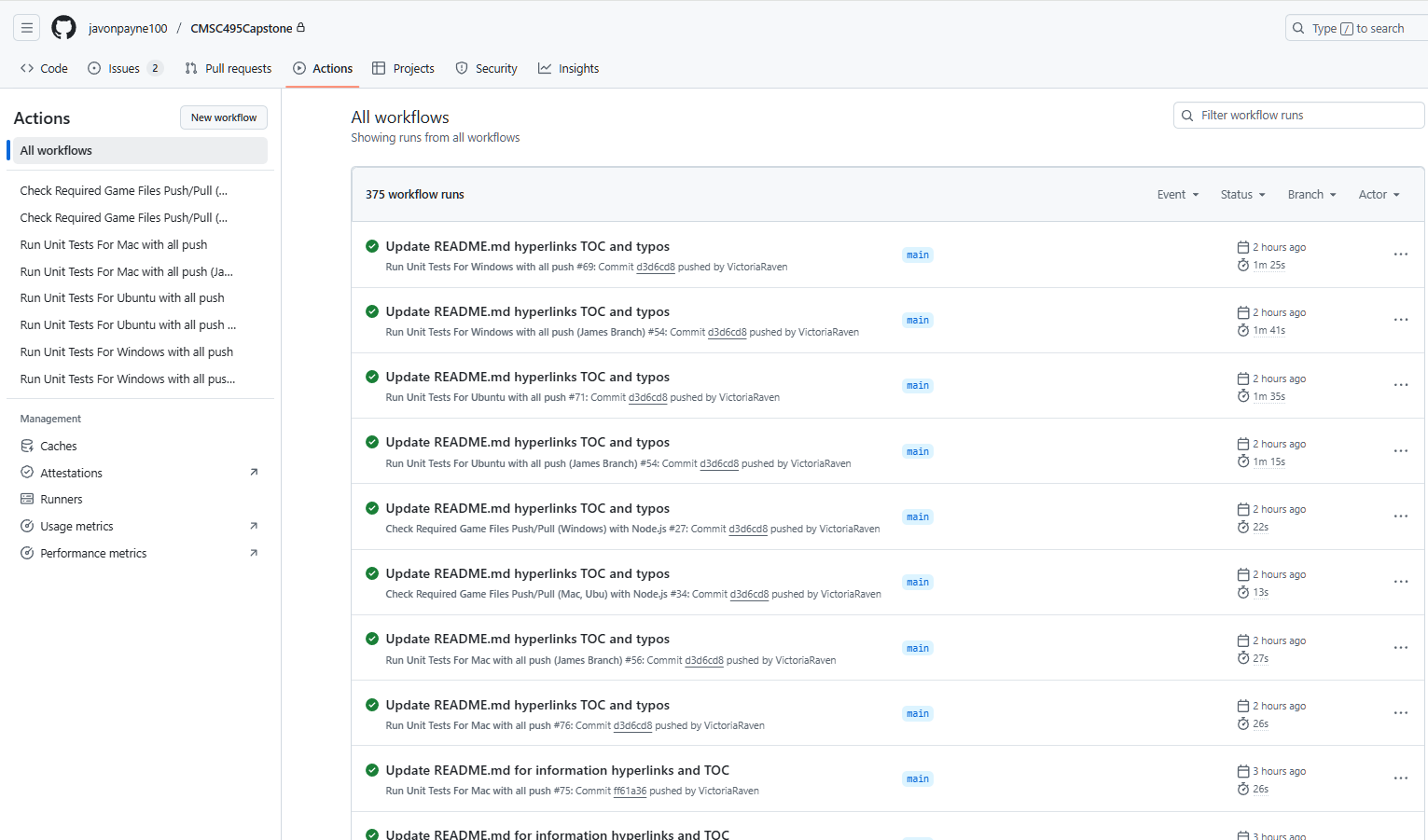
****

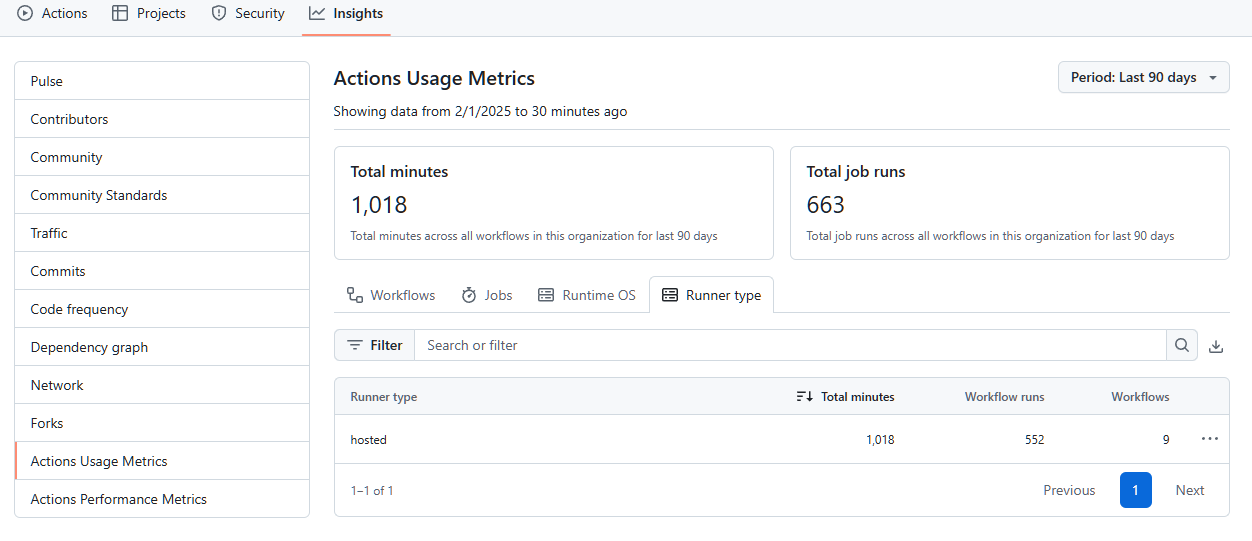
****

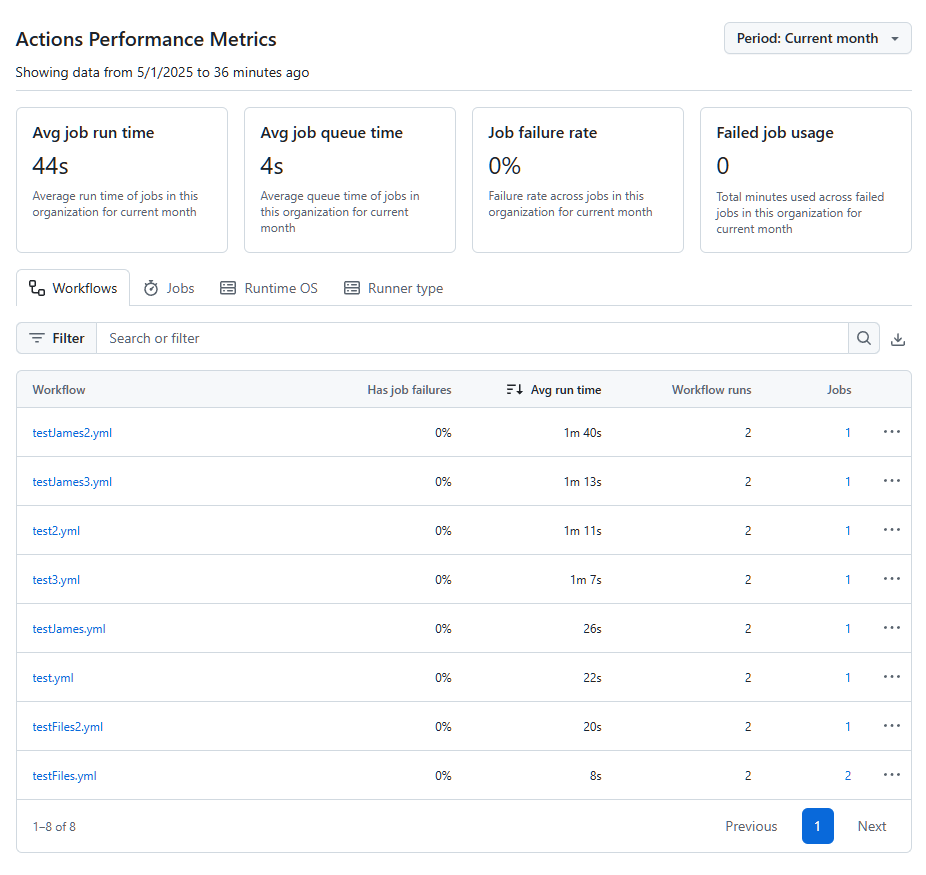
****

**Our GitHub Actions Workflow Overall Screenshots:**

****

****

****

****

**Unit 4’s Unit Testing Process Screenshots and basis for Unit 5 and Unit 6:**

This covers all parts of the Unit Testing, Debugging, and the relevant Software Engineering Techniques for each Break-Out, Tic-Tac-Toe, and Trivia. This table outlines the steps, questions to ask, and techniques that align with the software engineering process. I am placing this here so that you know the transition from Unit 4 to Unit 5 to Unit 6. This is based on GeeksforGeeks (n.d.) and Tsui, F., Karam, O., & Bernal, B. (2014). Below is the Chart diagram of this (Simplified):

A close-up of a grid

AI-generated content may be incorrect.

| **Category** | **Game/Menu** | **Meets Requirements** |
| --- | --- | --- |
| **Unit Testing/ Manual Testing:** | Break-Out | Yes; Completed; Closed |
|  | Tic-Tac-Toe | Yes; Completed; Closed |
| **Unit Testing/Manual Testing (Cont.)** | Trivia  Main Menu | Yes; Completed; Closed  Yes; Completed; Closed |
| **Unit Testing** | Break-Out | Yes; Completed; Closed |
|  | Tic-Tac-Toe | Yes; Completed; Closed |
|  | Trivia  Main Menu | Yes; Completed; Closed  Yes; Completed; Closed |
| **Manual Testing** | Break-Out | Yes; Completed; Closed |
|  | Tic-Tac-Toe | Yes; Completed; Closed |
|  | Trivia  Main Menu | Yes; Completed; Closed  Yes; Completed; Closed |
| **Debugging Process** | Break-Out | Yes; Completed; Closed |
|  | Tic-Tac-Toe | Yes; Completed; Closed |
|  | Trivia  Main Menu | Yes; Completed; Closed  Yes; Completed; Closed |
| **Performance Testing** | Break-Out | Yes; Completed; Closed |
|  | Tic-Tac-Toe | Yes; Completed; Closed |
|  | Trivia  Main Menu | Yes; Completed; Closed  Yes; Completed; Closed |
| **Cross-Platform Testing** | Break-Out | Yes; Completed; Closed |
| (Windows, Mac, Linux, IDE, etc) | Tic-Tac-Toe | Yes; Completed; Closed |
|  | Trivia  Main Menu | Yes; Completed; Closed  Yes; Completed; Closed |
| **Additional Features & Fixes** | Break-Out | Yes; Completed; Closed |
|  | Tic-Tac-Toe | Yes; Completed; Closed |
|  | Trivia | Yes; Completed; Closed |
|  | Main Menu | Yes; Completed; Closed |
| **Fixing 1 Issue in Unit 6** | Trivia and Questions | Yes; Completed; Closed |

**Note: Refer to APPENDIX E for the Git Version Control Process**

# **APPENDIX E:**

**Git (GitHub Version Control):**

**Testing/Debugging Requirements:**

* To track progress on each coding section, we will use Git for Version Control.
  + Each process will include a Git commit section
  + Each readme file will ensure that the user knows how to set the requirements of the software and run the application to play the games.
  + Refer to APPENDIX D for the Testing Unit Process and APPENDIX E for the Git Version Control Process

**Version Control with Git:**

* **Git as Version Control:**
  + Git will be used to track the progress of the project, manage code changes, and ensure collaboration between developers. Every change made to the codebase will be tracked with commit messages, providing a clear history of the project's development.
  + Refer to APPENDIX D for the Testing Unit Process and APPENDIX E for the Git Version Control Process
* **Branching and Commit Process:**
  + **Create a Separate Branch for Each Feature or Update:**
    - Each developer will create a separate feature branch to work on a specific task or update. This ensures that changes are isolated and don’t interfere with the main codebase (usually the main or develop branch).
    - Example: A developer working on the Tic-Tac-Toe game logic would create a branch named feature/tic-tac-toe-game-logic.
  + **Make Changes and Commit (Push/Pull/Directly through Main):**
    - Developers will work on their assigned feature in their own branches, making incremental changes as needed. For each change, they will commit with clear, descriptive commit messages explaining what was added or modified in the code.
    - After pushing the changes, the developer will create a Pull Request (PR) in GitHub to merge their feature branch into the main branch.
    - Or the developer can just upload it or change it directly through the main after asking permission form the other team members.
    - This ensures that the main branch always contains the most up-to-date, stable version of the code.
* **Tracking Code Changes and Progress:**
  + Each commit and PR will serve as a record of progress in the project. This enables easy tracking of features and bug fixes over time.
  + Developers can refer back to specific commits and PRs to understand when and why certain changes were made, ensuring transparency and accountability throughout the development process.
* **Collaboration and Coordination:**
  + Developers will regularly pull the latest changes from the main branch to keep their feature branches up to date and avoid conflicts.
  + If any merge conflicts arise, the developer will resolve them before pushing their changes, ensuring that the codebase remains functional and consistent.

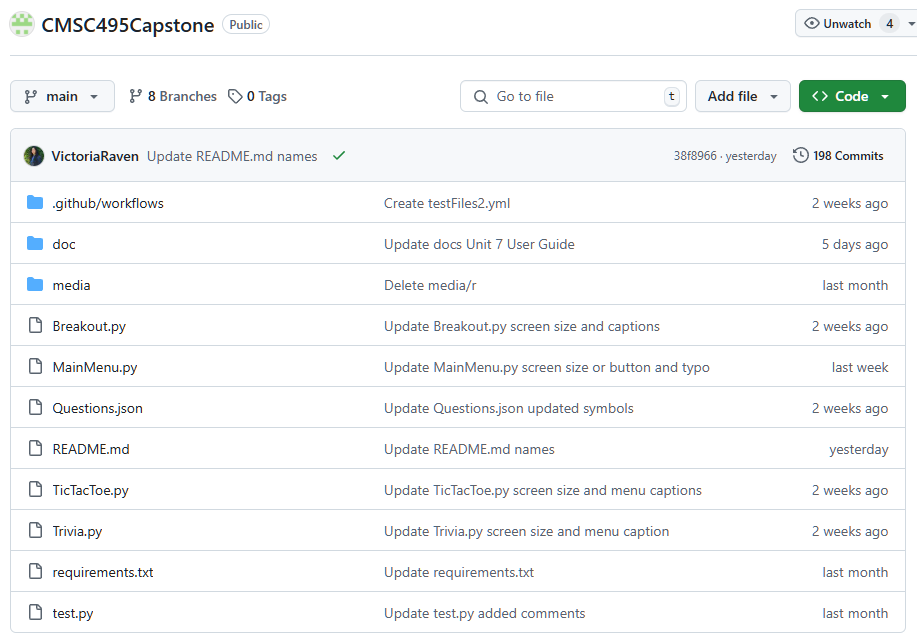
**Explanation:**

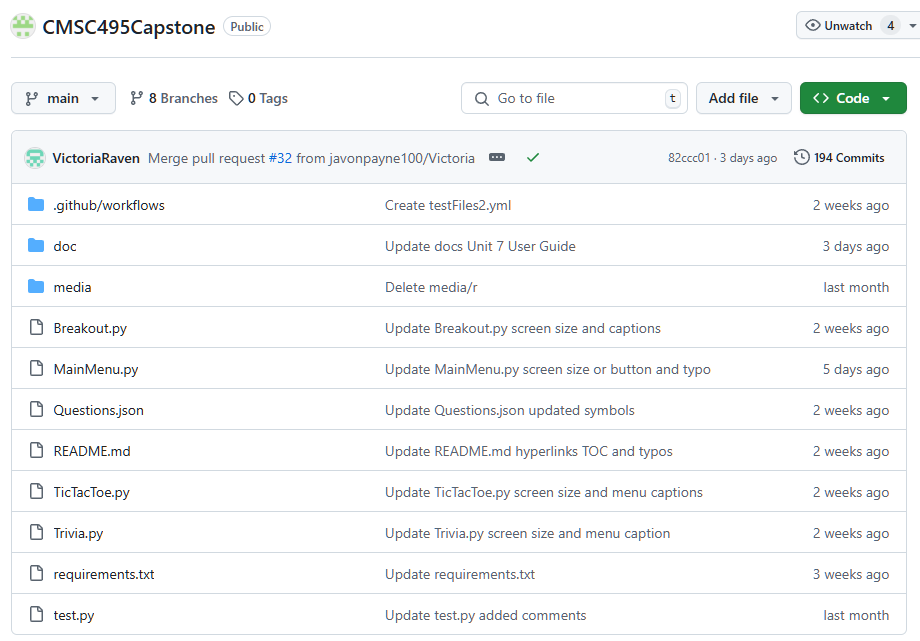
Our team uses GitHub to collaborate and track progress throughout the duration of the project. We maintained a shared repository on GitHub where each team member cloned the main branch into their own branch. Before each member was able to clone from the main branch, there had to be an established workflow. This consisted of four separate .py files in which they all connected to the MainMenu.py file, where the program starts. These steps assisted the team tremendously in the long run by making it easy to merge code seamlessly.

**To ensure a smooth collaboration, we followed a structured process:**

1. Feature Development: Each team member worked on a separate feature within their branches
2. Review and Commit: Before commits are made, each team member uploads their changes to their branch. The Git Lead then pulls each member's changes and tests them on the local environment. Once all code is confirmed to operate as expected it is uploaded to the main branch on GitHub. After it has been uploaded, each member pulls the code from the main and confirms that it is working. If the commit ever causes complications, we will simply revert the main branch back to its previous version.
3. Conflict resolution: We occasionally encounter merge conflicts, in which we discuss and collaborate using Git’s merging tools. If there are any bugs or glitches found, we simply record it in the issues section and begin repairing.
4. Tracking progress: To track changes we use GitHub commit history and issue tracking to monitor progress and assign tasks.

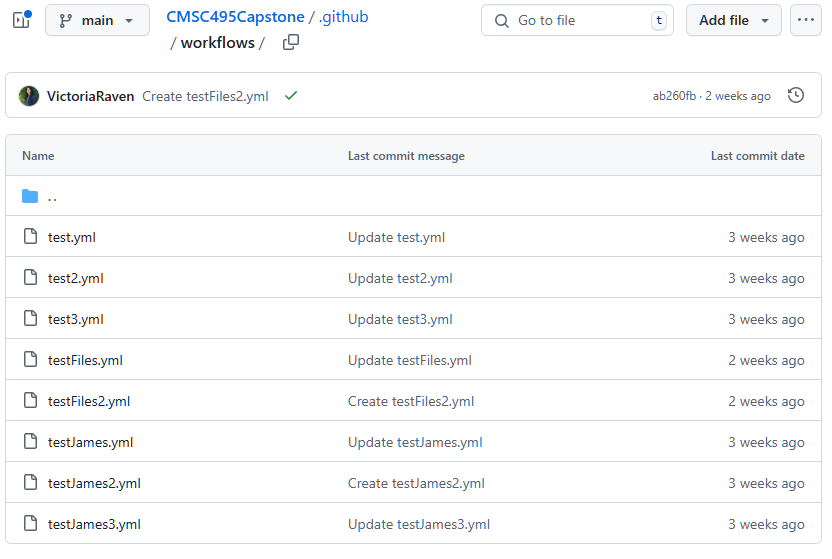
**Main branch:**

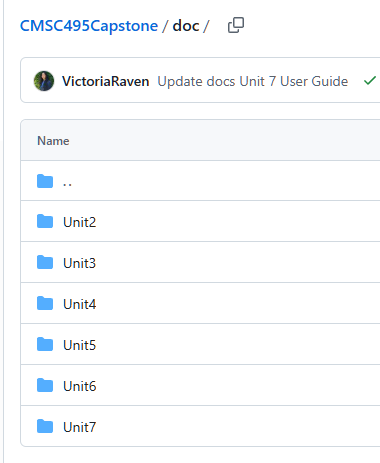
****



Media files and doc files and .github/workflow files:

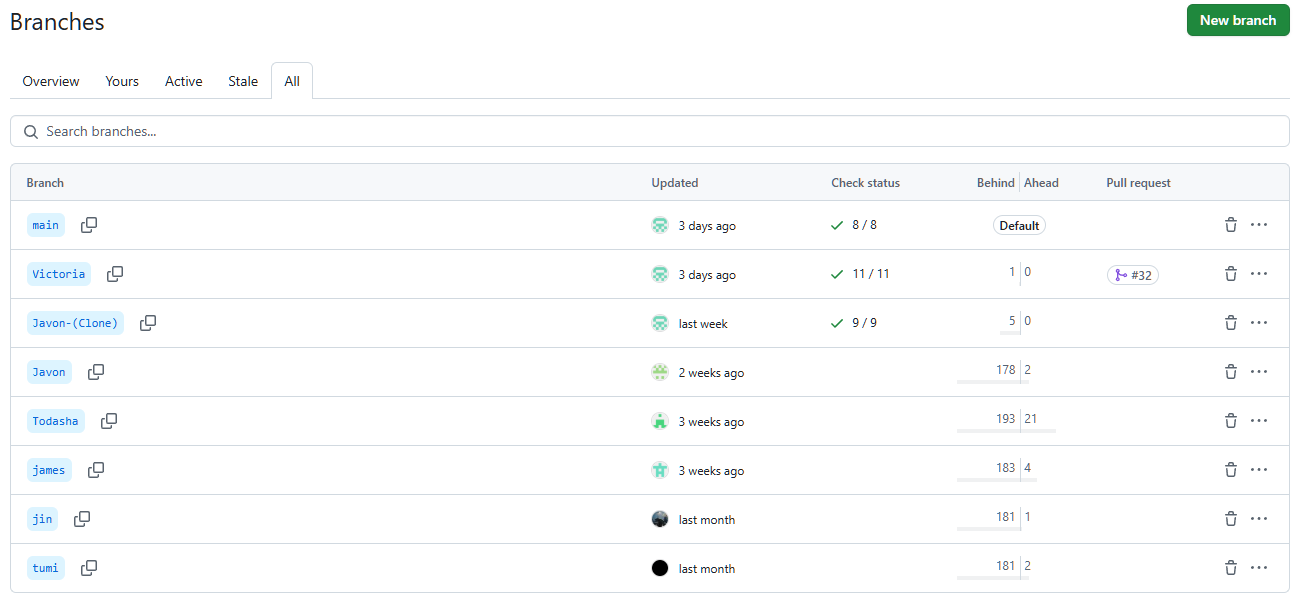
A screenshot of a computer

AI-generated content may be incorrect. 



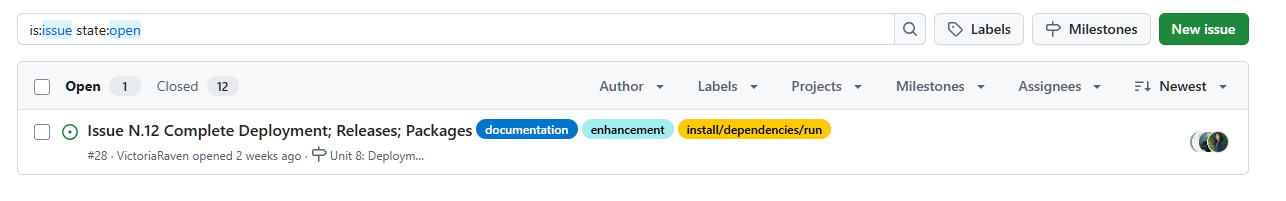
Note: Unit 8 is there, but I left it like that as I didn't want to re-screenshot it again.

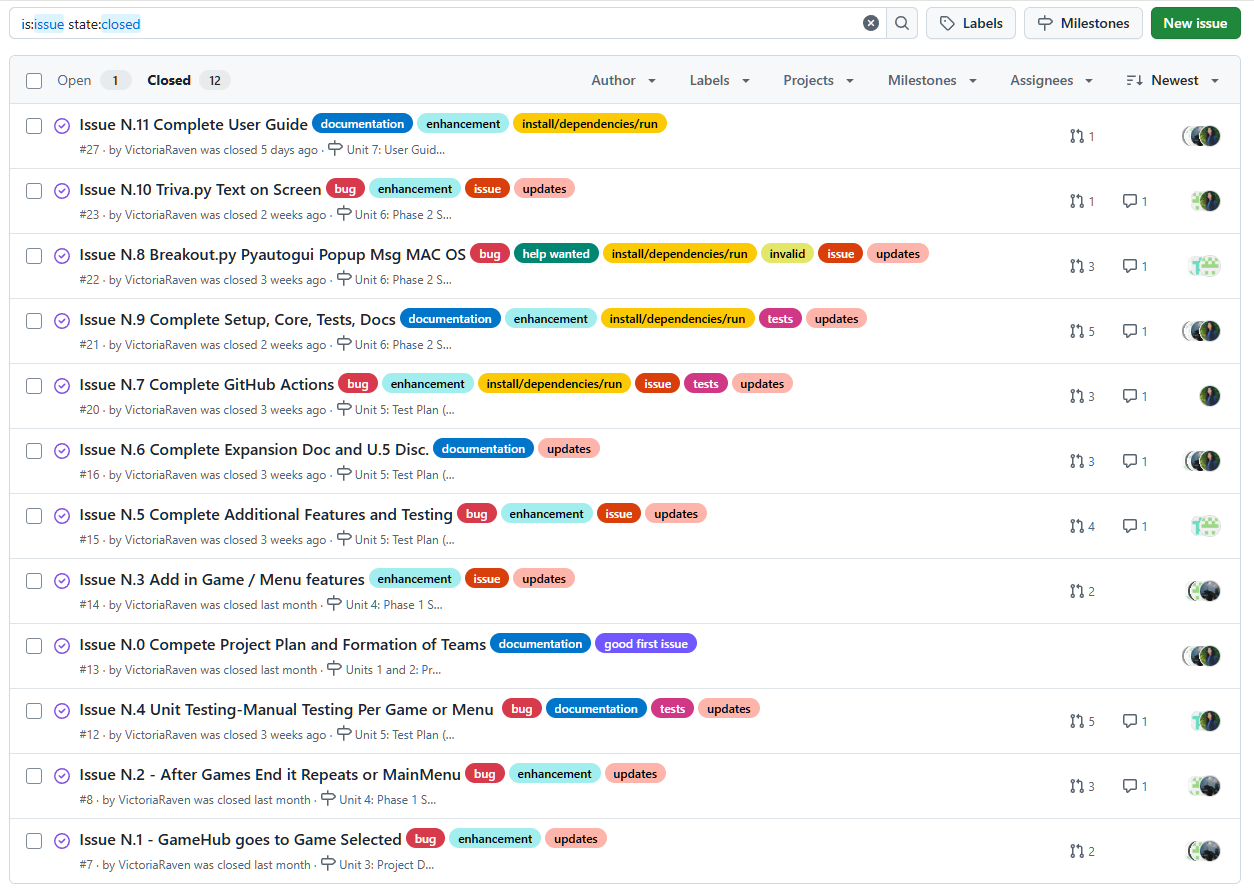
We have a total of 8 branches:



These are All our issues both Open and Closed:

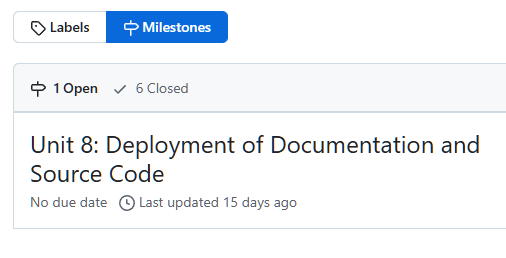
Note: Unit 8 parts are also closed and completed, but I left it like that as I didn't want to re-screenshot it again.

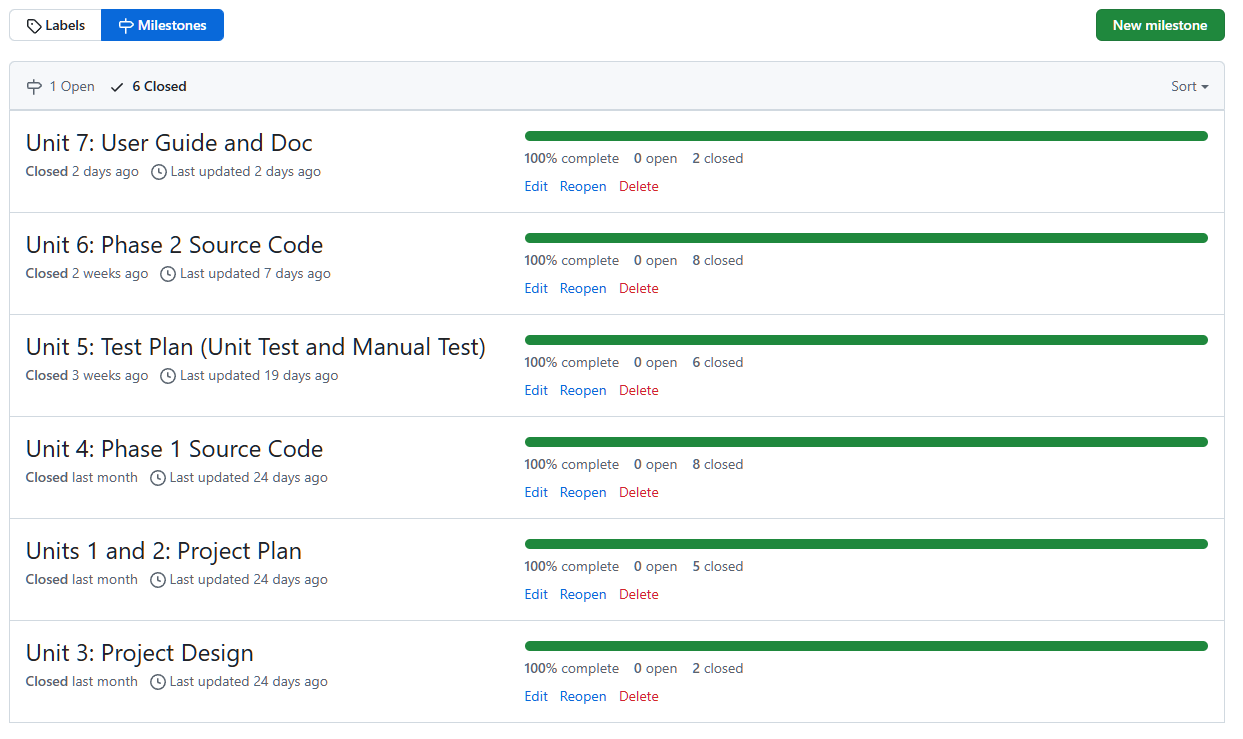




These are our milestones (open and closed):

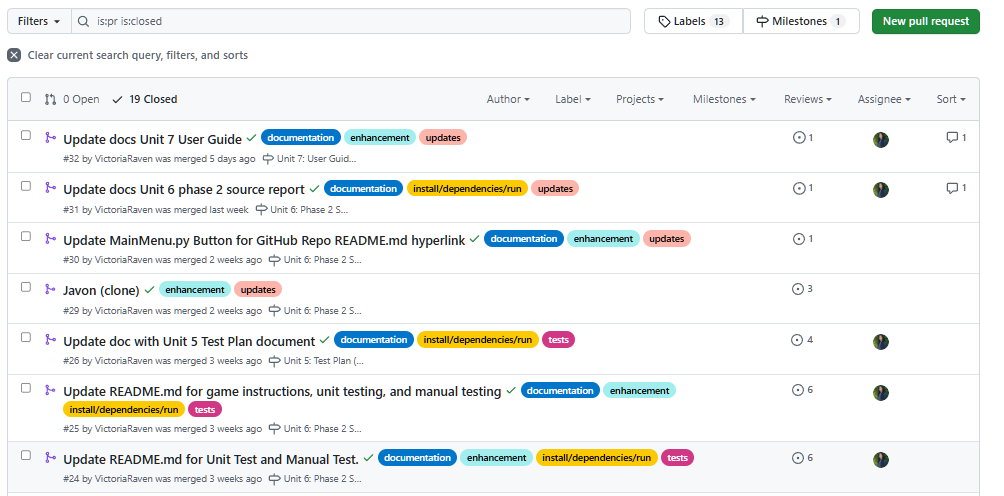
Note: Unit 8 parts are also closed and completed, but I left it like that as I didn't want to re-screenshot it again.

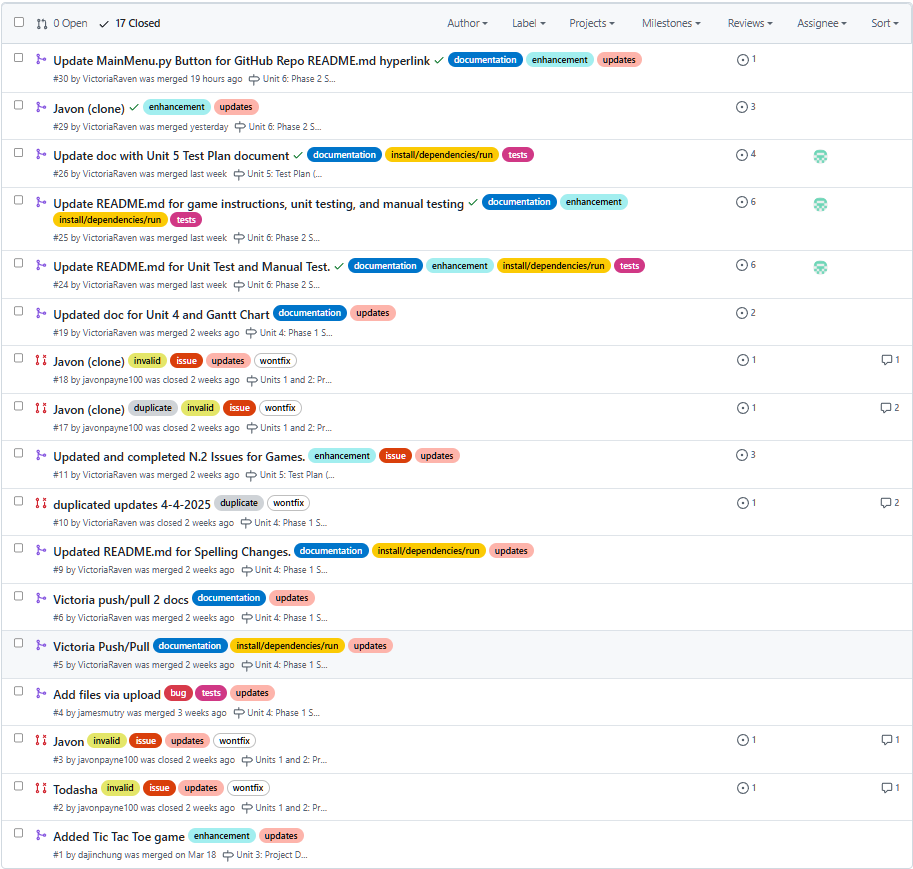




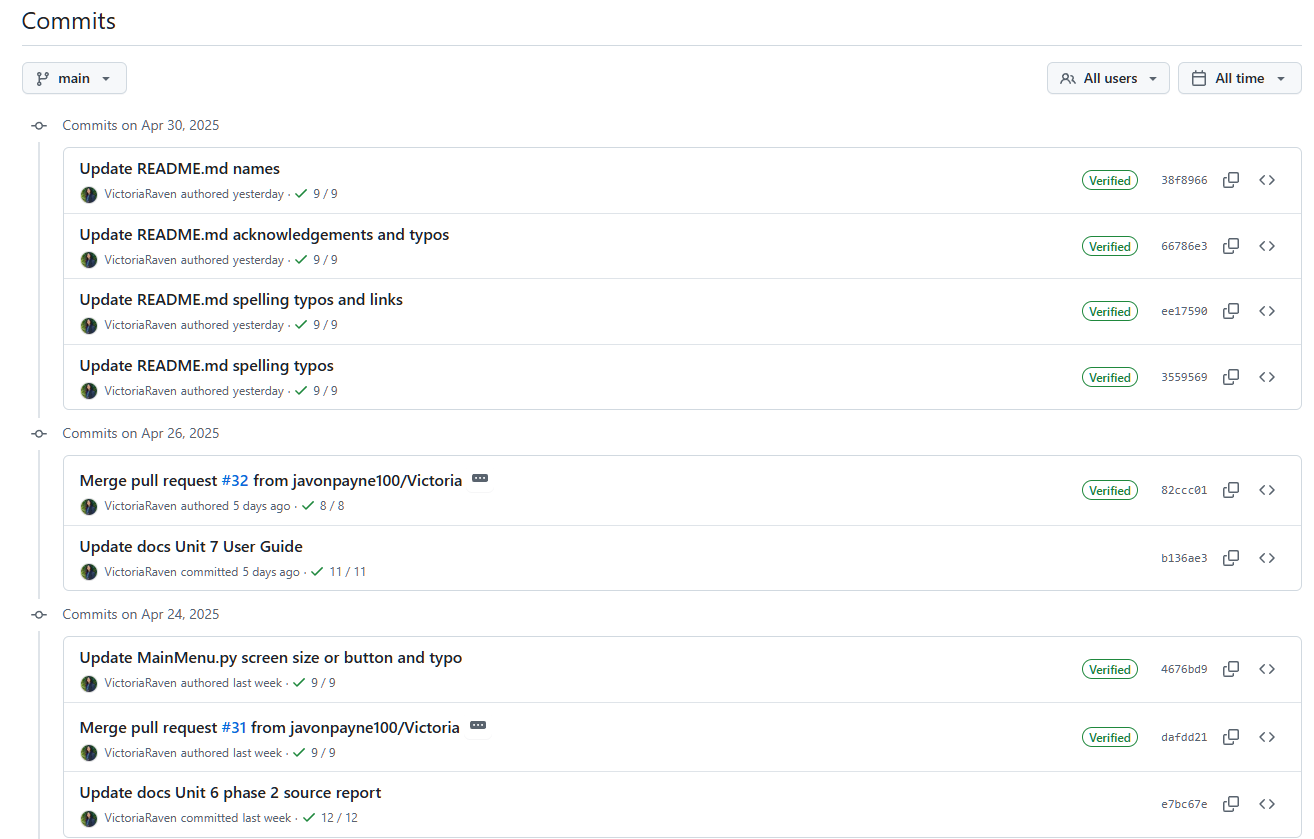
Here is our pull requests to merge onto the main using the branches:

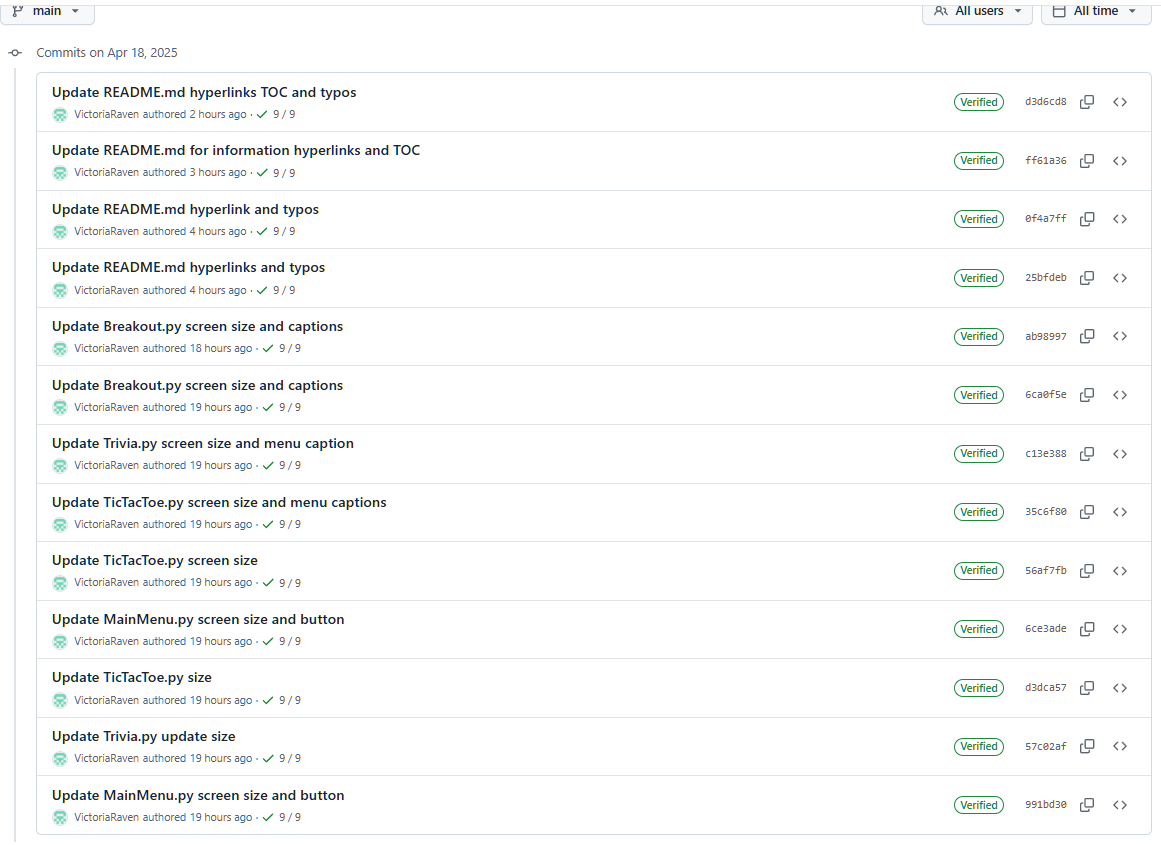
Note: Unit 8 parts are also closed and completed, but I left it like that as I didn't want to re-screenshot it again.

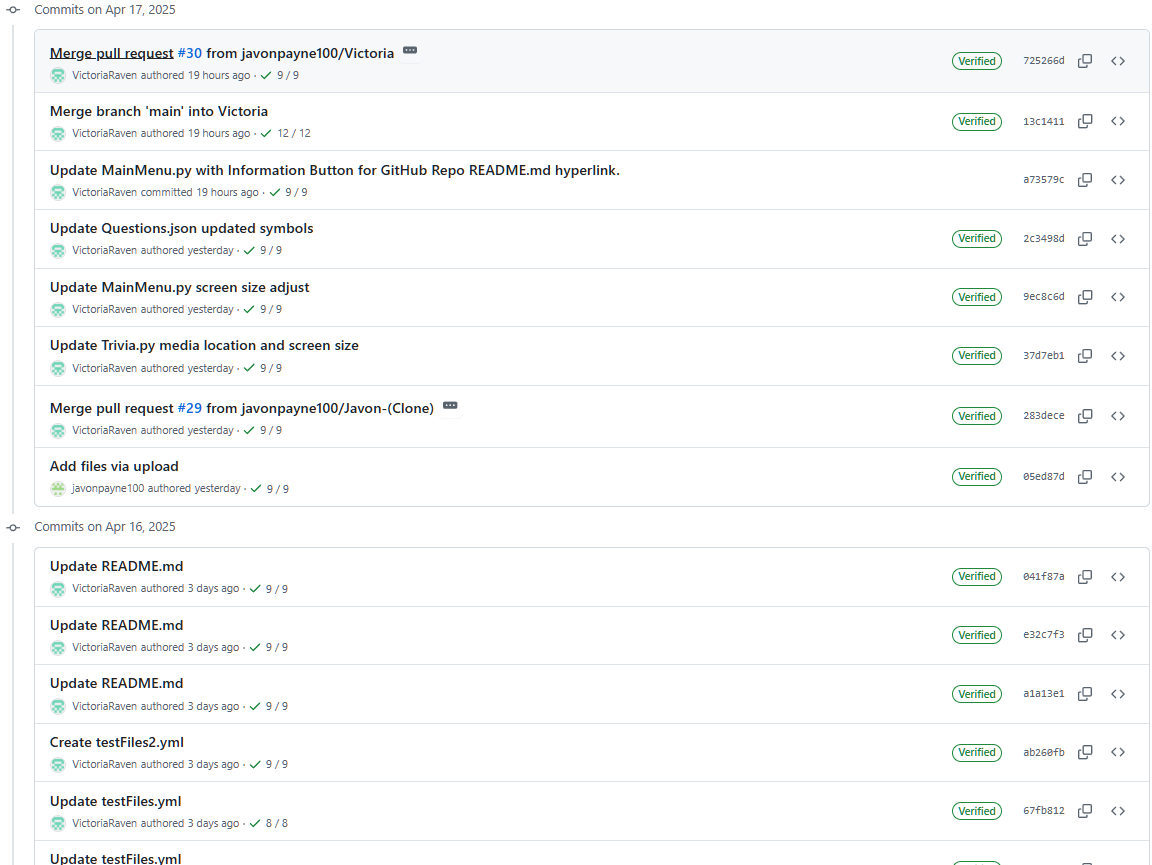


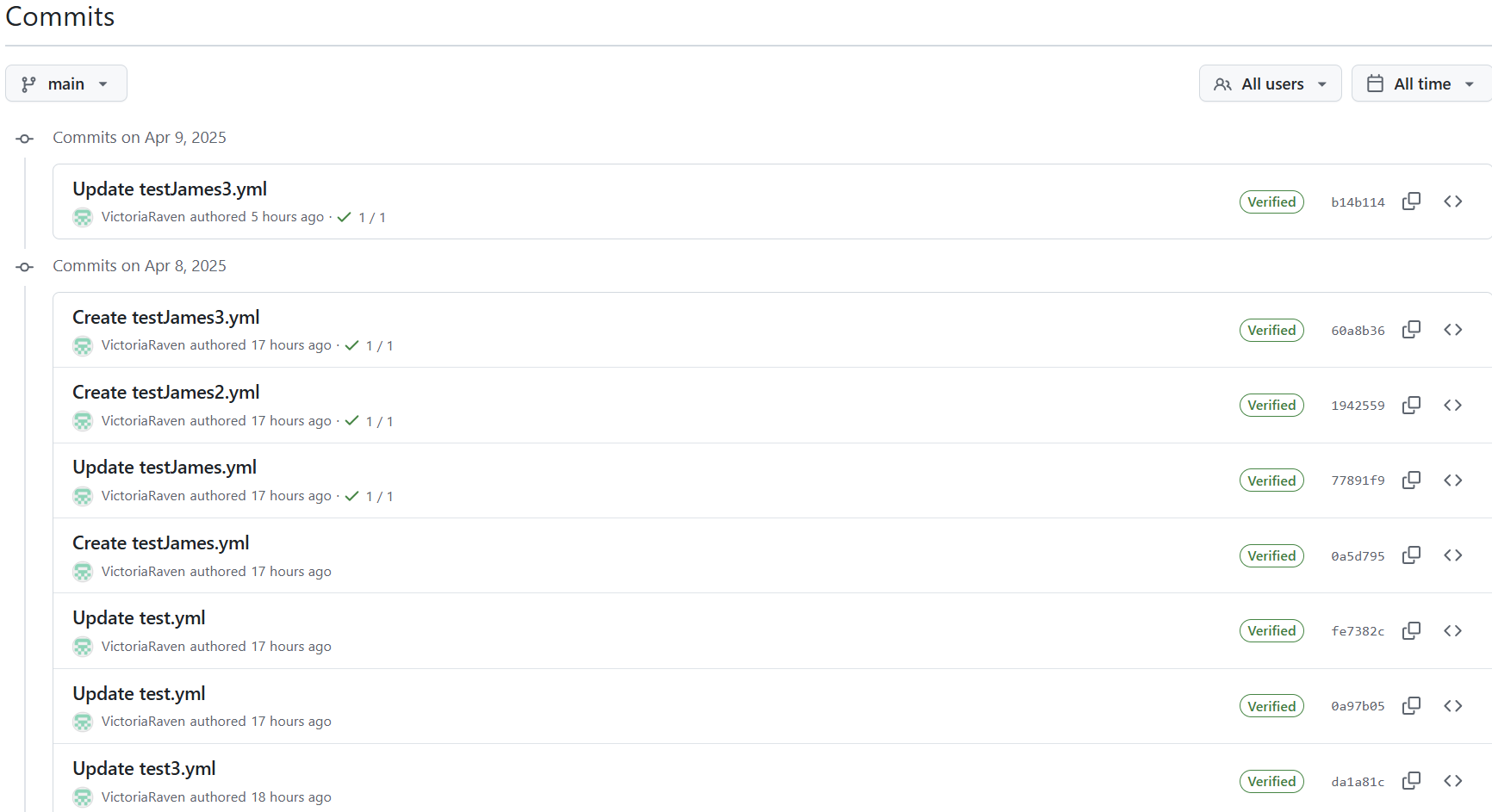


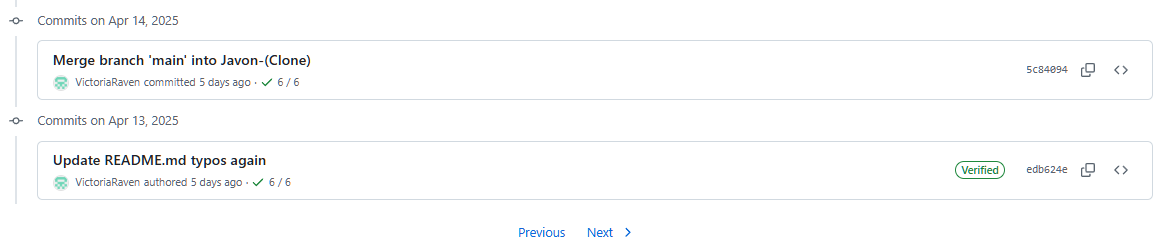
We have a total of 198 Commits (I will not show all screenshots as it is quite long; only main parts):

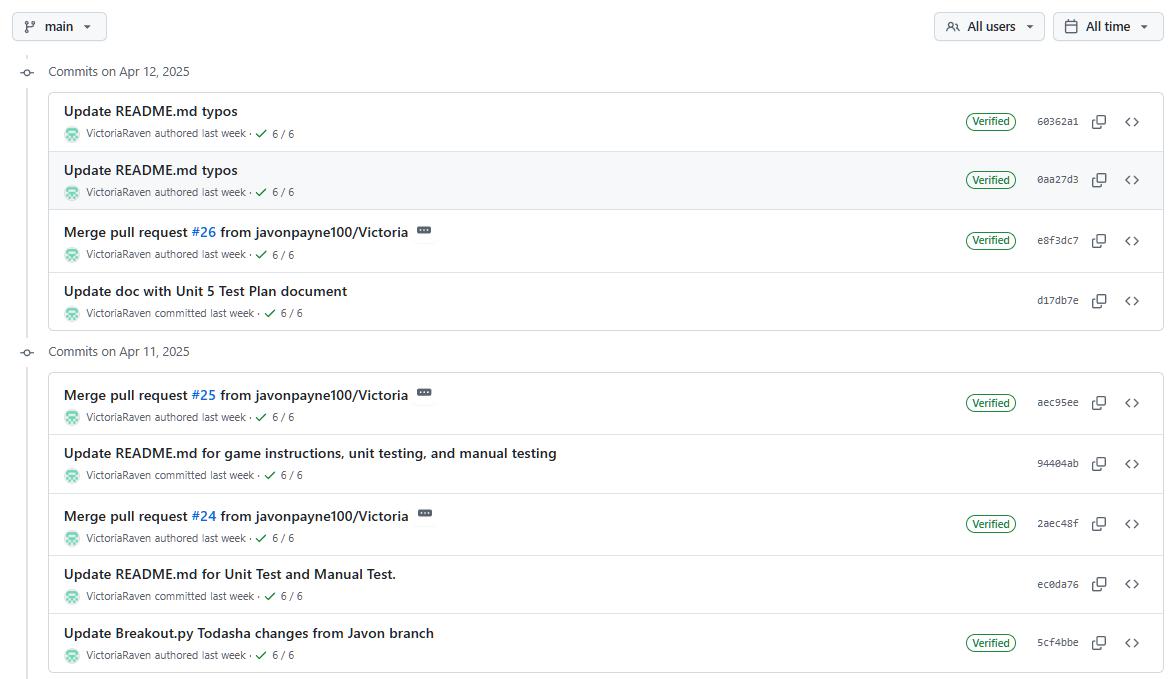


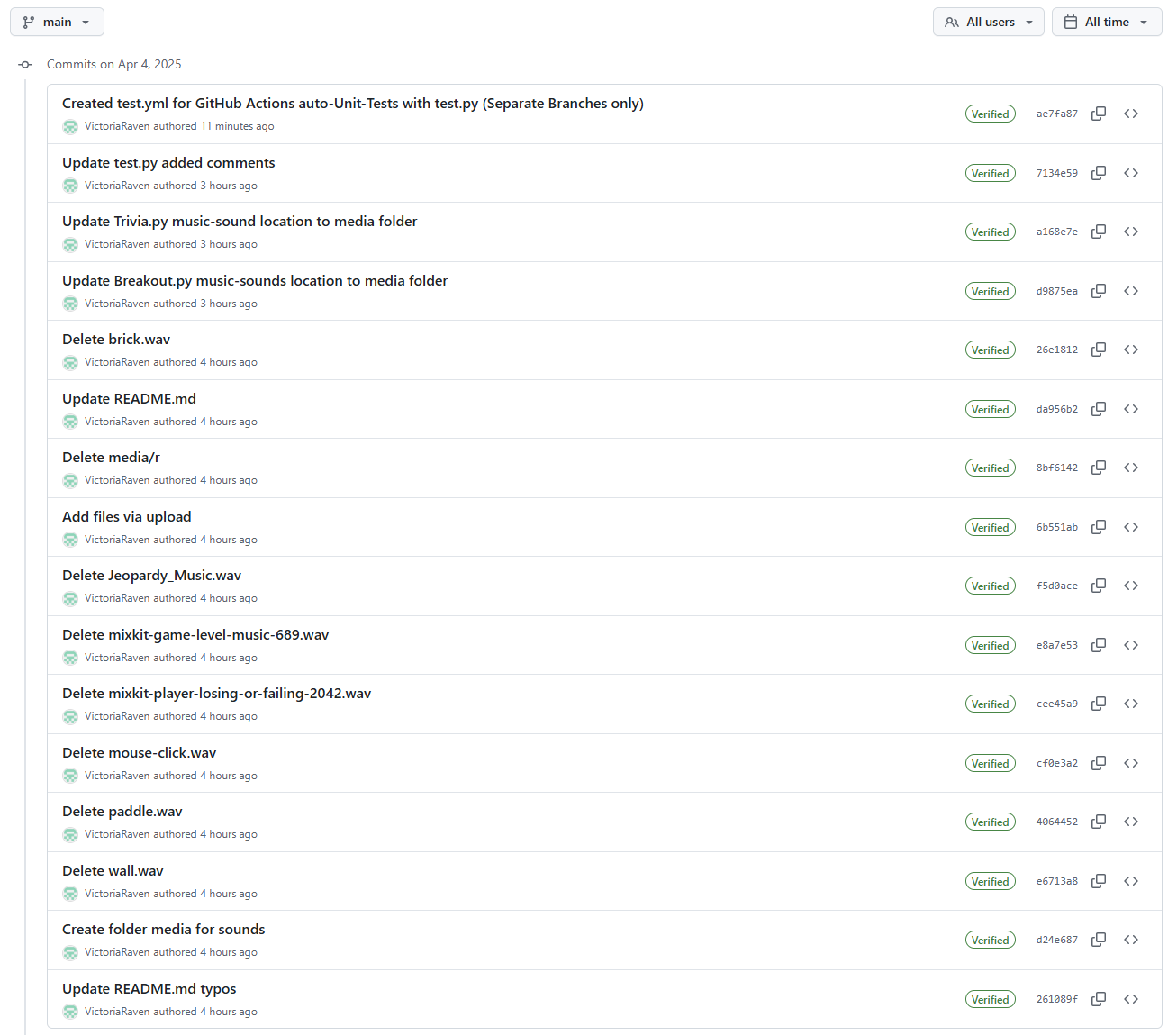


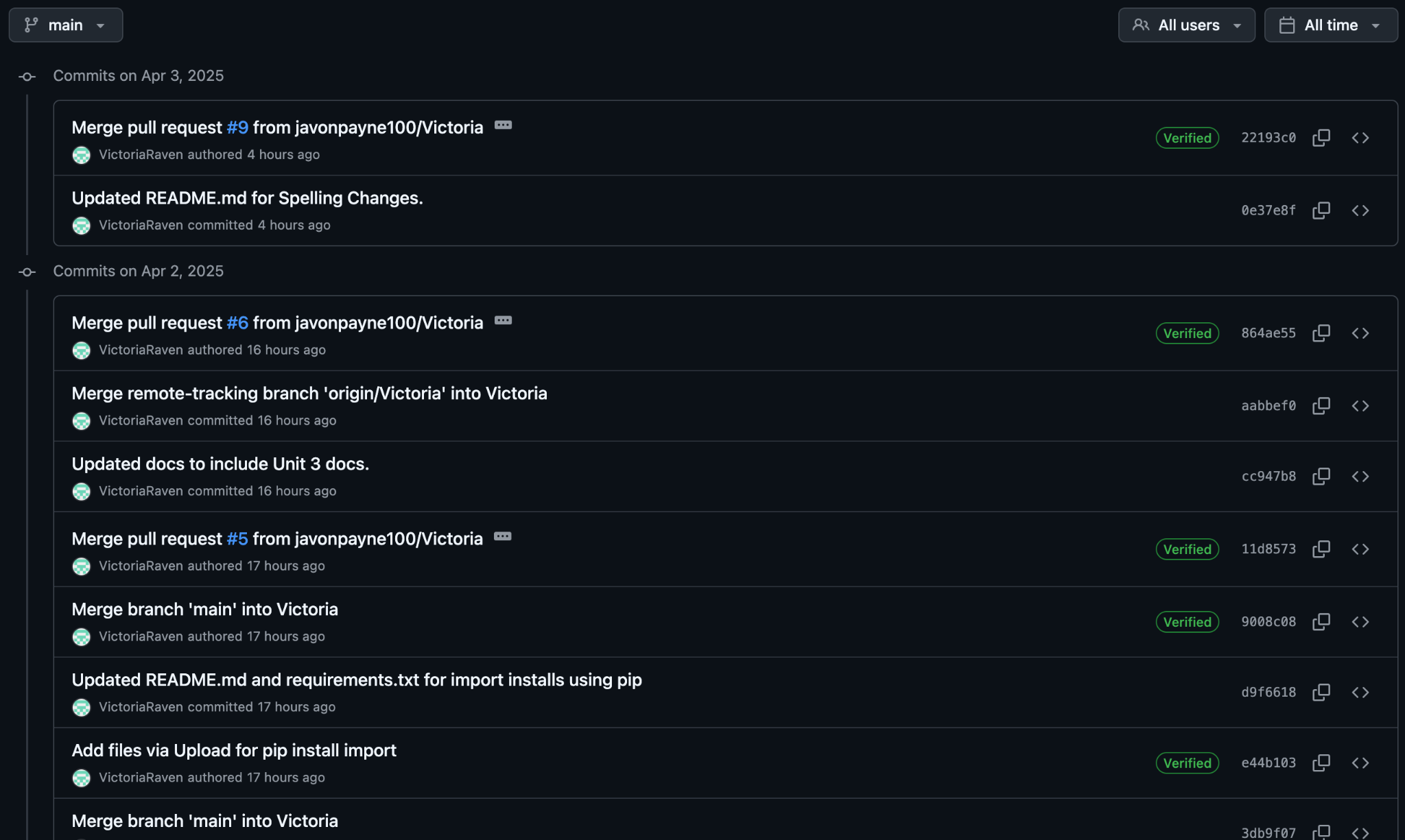


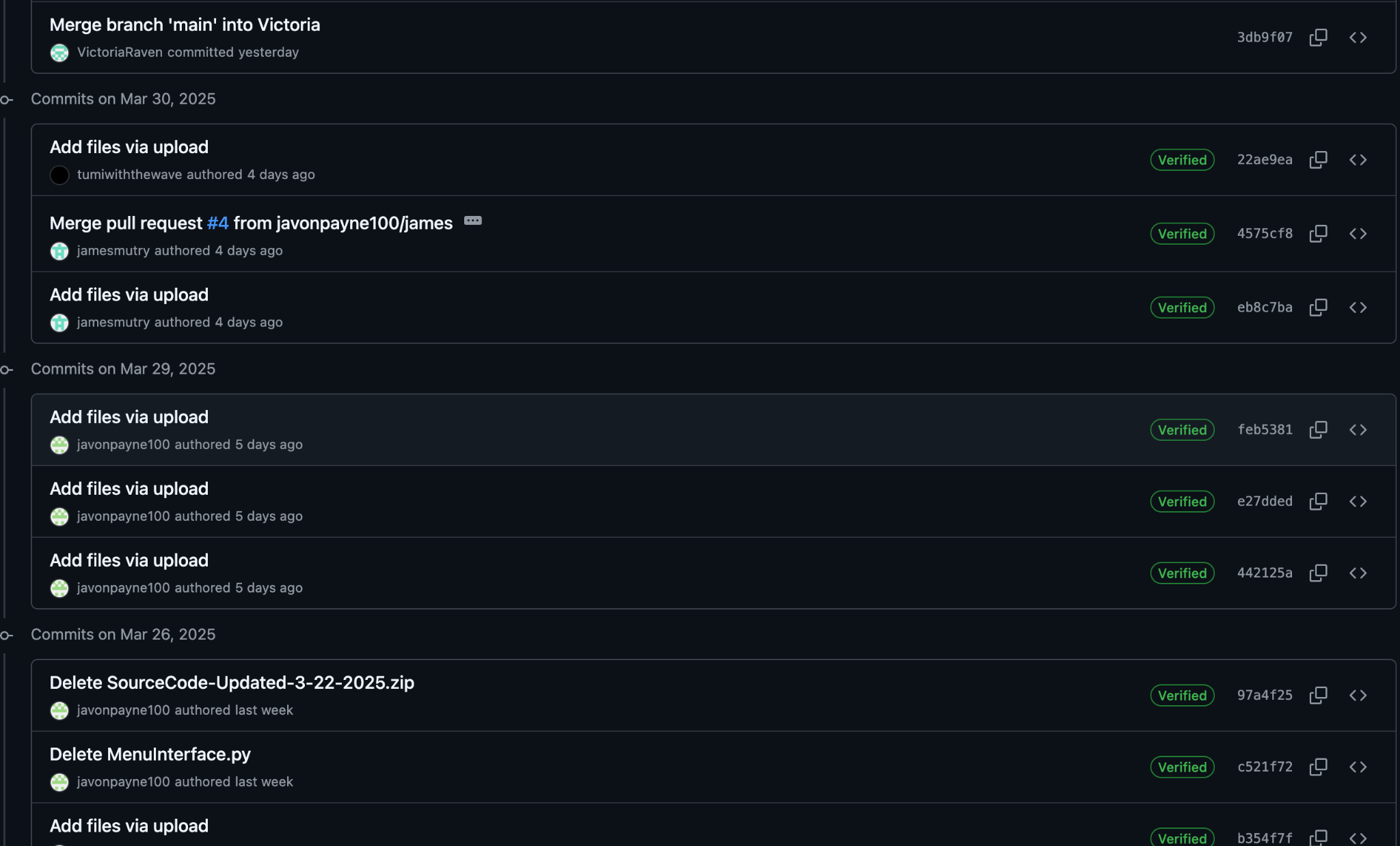














# **APPENDIX F:**

Links to specific details of system specification, technical specifications, minimum hardware requirements of the project implemented:

* **Unit 6: Phase 2 Source Report Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit6)
* **Unit 7: User Guide**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)
* **Unit 8: Deployment Report**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1CYzgpDp1lFNmwkbjhaIwnEbEDStKLXPn/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [GitHub (doc) Link] (Check it at the GitHub repository doc folder!)
    - [Google Slides Link] (Check it at the GitHub repository doc folder or the Unit 8 discussion!)  [[PythonGameHub GitHub Repo Link]](https://github.com/javonpayne100/CMSC495Capstone)
* **README.md file:**
  + [[GitHub (readme.md) Link]](https://github.com/javonpayne100/CMSC495Capstone#cmsc495capstone-python-game-hub-group-1)

# **APPENDIX G:**

Here is all the links Units (1-8) if you had a hard time navigating around this document:

* **README.md file:**
  + [[GitHub (readme.md) Link]](https://github.com/javonpayne100/CMSC495Capstone#cmsc495capstone-python-game-hub-group-1)
* **Unit 2: Project Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1F2LR0j118Oje_9Gw5iaabmx6HMe-YYtb/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit2)
* **Unit 3: Project Design Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1hjNWfBtiacfR40g9wAcIkf-xypBD3PlF/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit3)
* **Unit 4: Phase 1 Source Report Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1oHjphF6c_7OVMOcpK2WWTJcwo7Pe4vee/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit4)
* **Unit 5: Testing Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1-Gexgrs11wsRRzTdWwDQkw0dHNirzS0a/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit5)
* **Unit 6: Phase 2 Source Report Plan**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1HAwOUjxfqhFCv_1LqyUl1gVJ1GLbih6D/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit6)
* **Unit 7: User Guide**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1GQypNUDMwe6NIpPggC_DLJccZz0VAXHf/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [[GitHub (doc) Link]](https://github.com/javonpayne100/CMSC495Capstone/tree/main/doc/Unit7)
* **Unit 8: Deployment Report**
  + - [[Google Drive (Docx) Link]](https://docs.google.com/document/d/1CYzgpDp1lFNmwkbjhaIwnEbEDStKLXPn/edit?usp=sharing&ouid=101579491048440949067&rtpof=true&sd=true)
    - [GitHub (doc) Link] (Check it at the GitHub repository doc folder!)
    - [[Google Slides Link]](https://docs.google.com/presentation/d/10nZwOQwZNqL12yTPeKOJZYSwu5xBDgPzDS-rd9FtDY8/edit?usp=sharing) (Check it at the GitHub repository doc folder or the Unit 8 discussion!)