

A roleplaying fighting game told through text

*“Serpents: Petetra’s Journey”*

Project Portfolio

*Due date: 11/13/2023*

[**Introduction 2**](#_heading=h.30j0zll)

[**The Serpents Team 3**](#_heading=h.1fob9te)

[**System Requirements 4**](#_heading=h.3znysh7)

[Requirements 4](#_heading=h.2et92p0)

[Epics 4](#_heading=h.tyjcwt)

[*Epic #1 4*](#_heading=h.3dy6vkm)

[User Stories 4](#_heading=h.1t3h5sf)

[*User Story #1 4*](#_heading=h.4d34og8)

[**Project Management 5**](#_heading=h.2s8eyo1)

[Continuity of Operations Plan (COOP) 5](#_heading=h.17dp8vu)

[Project Plan 0](#_heading=h.3rdcrjn)

[*System Architecture Design and Development 0*](#_heading=h.26in1rg)

[*System Implementation <Milestone 2: Architecture & Milestone 3: System Implementation> 0*](#_heading=h.lnxbz9)

[Project Postmortem <Postmortem> 0](#_heading=h.35nkun2)

[*Project Wins 0*](#_heading=h.1ksv4uv)

[*Root Cause Analysis 0*](#_heading=h.44sinio)

[*Lessons Learned 0*](#_heading=h.2jxsxqh)

[**System Design 1**](#_heading=h.z337ya)

[System Architecture 1](#_heading=h.3j2qqm3)

[*Component Design 1*](#_heading=h.1y810tw)

[*Data Flow 1*](#_heading=h.4i7ojhp)

[System Components <Milestone 3: System Implementation> 1](#_heading=h.2xcytpi)

[*Component [Component Name 1] 1*](#_heading=h.1ci93xb)

[*Component [Component Name 2] 1*](#_heading=h.3whwml4)

[*Component [Component Name n] 1*](#_heading=h.2bn6wsx)

[Design Pattern <Milestone 3: System Implementation> 1](#_heading=h.qsh70q)

[Design Pattern <Milestone 3: System Implementation> 2](#_heading=h.3as4poj)

[**System Implementation <Milestone 3: System Implementation> 3**](#_heading=h.1pxezwc)

[Project portfolio template directives and placeholders (delineated by “[ ]” or “< >” and/or highlighted or optional sections not included) should be removed from the document prior to submission. Empty sections for inclusion in later submissions may remain in the document for early submissions.]

[IMPORTANT: All diagrams developed using Enterprise Architectures must include the following acknowledgement: “Thanks to SPARX Systems for LSU student and faculty use of Enterprise Architect for academic purposes”.]

# Introduction

The project involves the player navigating a dungeon with a story driven narrative. In the dungeon prompts will be given in order to explain the situation and give options to the player. This can be achieved in C++ using several exchanges of inputs and consequential outputs. This is mostly going to be the intersection of various classes that reference each other to have the desired output of the player. CMake would be required to have our code compiled.



Core Features:

* Interface always gives the player options on what to do next, the player can type in a simple response and they will be told the consequences of their actions.
* Combat system that takes player input while engaging with an enemy
* Map traversal that affects scenarios and outcomes

Viable Features:

* Updating in-depth story
* Status effects during combat that alter the flow of the game
* Random map generation

Stretch Features

* Difficult boss battle with special item interactions
* Multiple endings
* Expanding map showing discovered areas

# The Serpents Team

The team is structured in a loose fashion.

Paul will handle meeting scheduling and C++ coding and will try to take the initiative when opening new software and running it. He should be the first to share his screen when demonstrating things on Discord, and he manages the Google DOCX that the team uses to work in parts of the project template collaboratively. He has experience in Java, Python, Linux Terminal, and Overleaf, as well as experience using most of the apps in Google Drive and Microsoft Office, including Colaboratory. He’ll try to coordinate what he can in the first milestone and pick up with C++ coding after. In particular, he intends to design the dialogue system and write the story as it pertains to the monsters and items the others create.

Cory: He handles JavaScript/TypeScript, React, Postgresql, Node.js. He will take a role enemy types and their abilities

Colby Blank: Front-end focused development. In this project of mostly C++ he will work on the combat system’s fundamentals. He will also work on the submission and proofreading of documents submitted.

Ethan has prior experience in C++ via personal projects and can handle some of the C++ programming. In the first milestone he can offer functionality ideas and help plan the overall structure of the program. Ethan has programmed something similar in Java and plans to handle and implement map design and possibly random map generation.

Joshua Pugh has experience in Java and some basic knowledge of C and C++. He will handle the items and armor class.

# System Requirements

## Requirements

[*A list of system requirements. This should include, at a minimum, the requirements imposed by the class project*.]

The system requires an ability to run C++ code, and presumably at least some storage.

## Epics

### Epic #1

*As a player of the game, I want to have an enjoyable experience in which I can decide on an adventure for the fictional protagonist, so I can have fun.*

## User Stories

### User Story #1

*As a player interested in combat, I want to have an engaging combat system, so I can relish the achievement of overcoming a difficult challenge.*

### User Story #2

*As a player interested in story, I want to see an interesting story, so I can have fun and experience the headspace of the protagonist.*

# Project Management

## Continuity of Operations Plan (COOP)

Communication for the group will be done through multiple discord servers/groupme. Coordination involves set meetings either in person or over a voice call in which the group will conduct work on the project. This would not be a set time, but would incorporate a set of goals to be accomplished in that time. In the event of anyone becoming unavailable the group would pick up the slack and redistribute the main workload. In the case of extended absence or outright removal of the group, a new evaluation will be done on the current roles and redistributed to meet the teams needs.

The focus of these procedures would allow for weekly meetings in which set goals can be discussed and strategized. If work can be done independently it would be handed to any participant deemed appropriate to take care of said work. This, along with the absence procedure, can allow for every group member to make adequate contributions no matter the level of their programming/organization.

## Project Plan

### System Architecture Design and Development

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Activity** | **Pre #** | **Estimated**  **Effort** | **Actual**  **Effort** | **Estimated**  **Start Date** | **Estimated**  **Finish Date** | **Actual**  **Start Date** | **Actual**  **Finish Date** |
| 1 | Create a ‘backbone’ of classes from a template Ethan once made, converting it all | N/A | A week of work | A week of work | 9/25/23 | 10/9/23 | 10/1 | 10/9/23 |
| 2 | Implementing a prototype of a simple dialogue between the player and interface allowing them to do simple tasks, not including fighting. | N/A | Half a week | Three days | 9/25/23 | 10/9/2023 | 10/6/23 | 10/9/23 |

### System Implementation <Milestone 2: Architecture & Milestone 3: System Implementation>

[Milestone 2 (Architecture): The Project Plan WBS provides a list of activities/tasks to be undertaken to complete Milestone 3 (System Implementation). The WBS activity chart should include task dependencies, estimated level of effort, and expected start and completion dates.

Milestone 3 (System Implementation): The WBS activity chart for the milestone should be updated to include actual level of effort and start and completion dates.]

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Activity** | **Pre #** | **Estimated**  **Effort** | **Actual**  **Effort** | **Estimated**  **Start Date** | **Estimated**  **Finish Date** | **Actual**  **Start Date** | **Actual**  **Finish Date** |
| 1 | Creating a full dialogue tree and story | 4 | 1-2 week |  | 10/12 | Milestone3 |  |  |
| 2 | Finishing all the items | 1 | 1 week |  | 10/15 | Milestone3 |  |  |
| 3 | Creating all the monsters and combat options | N/A | 1 week |  | 10/10 | Milestone3 |  |  |
| 4 | Creating the map | N/A | 1-2 week |  | 10/10 | 10/15 |  |  |

## Project Postmortem <Postmortem>

### Project Wins

[Provide a bulleted list of at least 3 positive aspects of the project.]

### Root Cause Analysis

[Provide a bulleted list of at least 3 negative aspects of the project. For each negative, provide the answer to the three successive “Why” questions. ]

### Lessons Learned

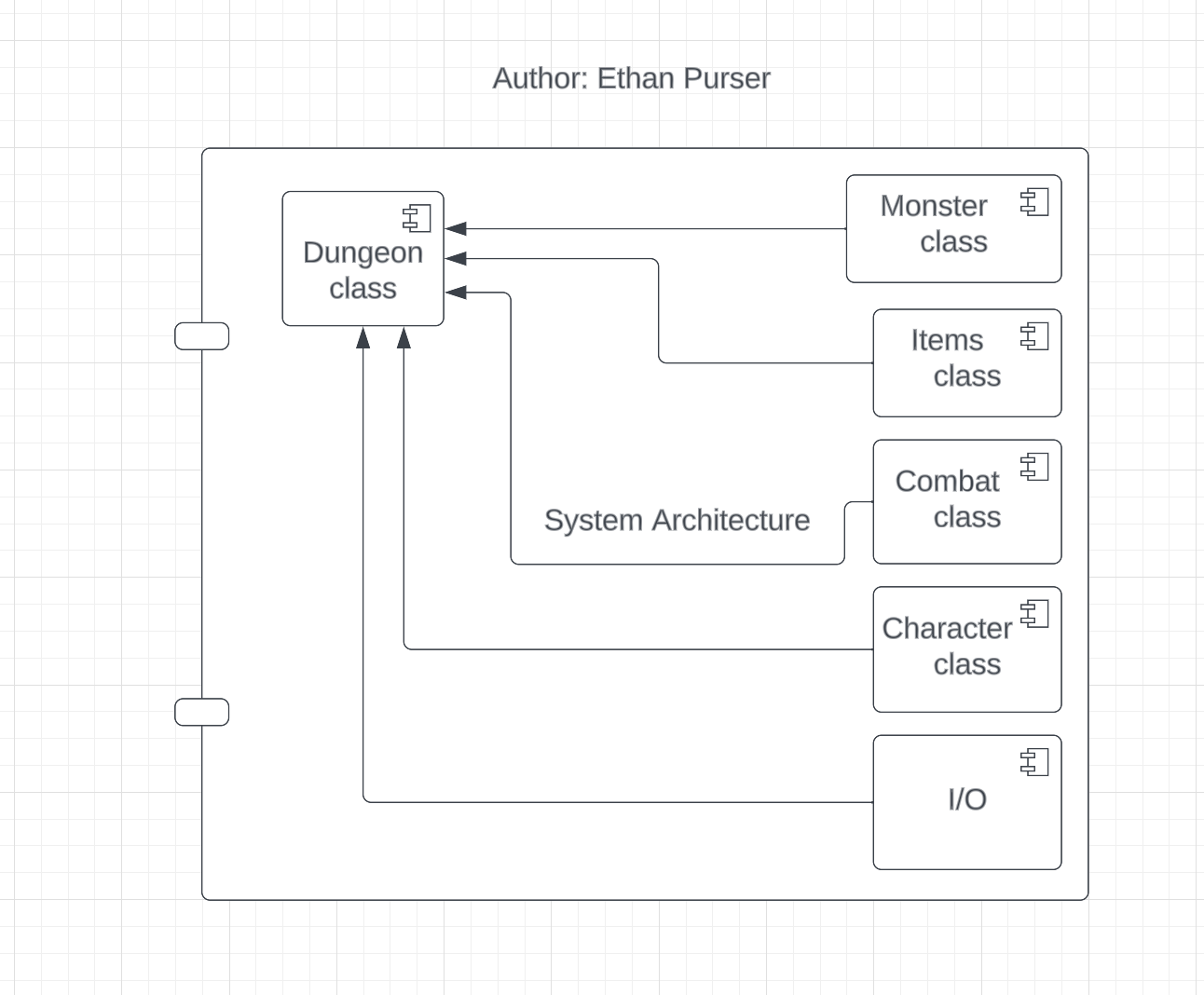
[For each negative aspect identified in the Root Cause Analysis, provide a mitigation strategy (i.e., what process should be introduced) to ensure that the problem is not repeated in subsequent projects.]

# System Design

## System Architecture

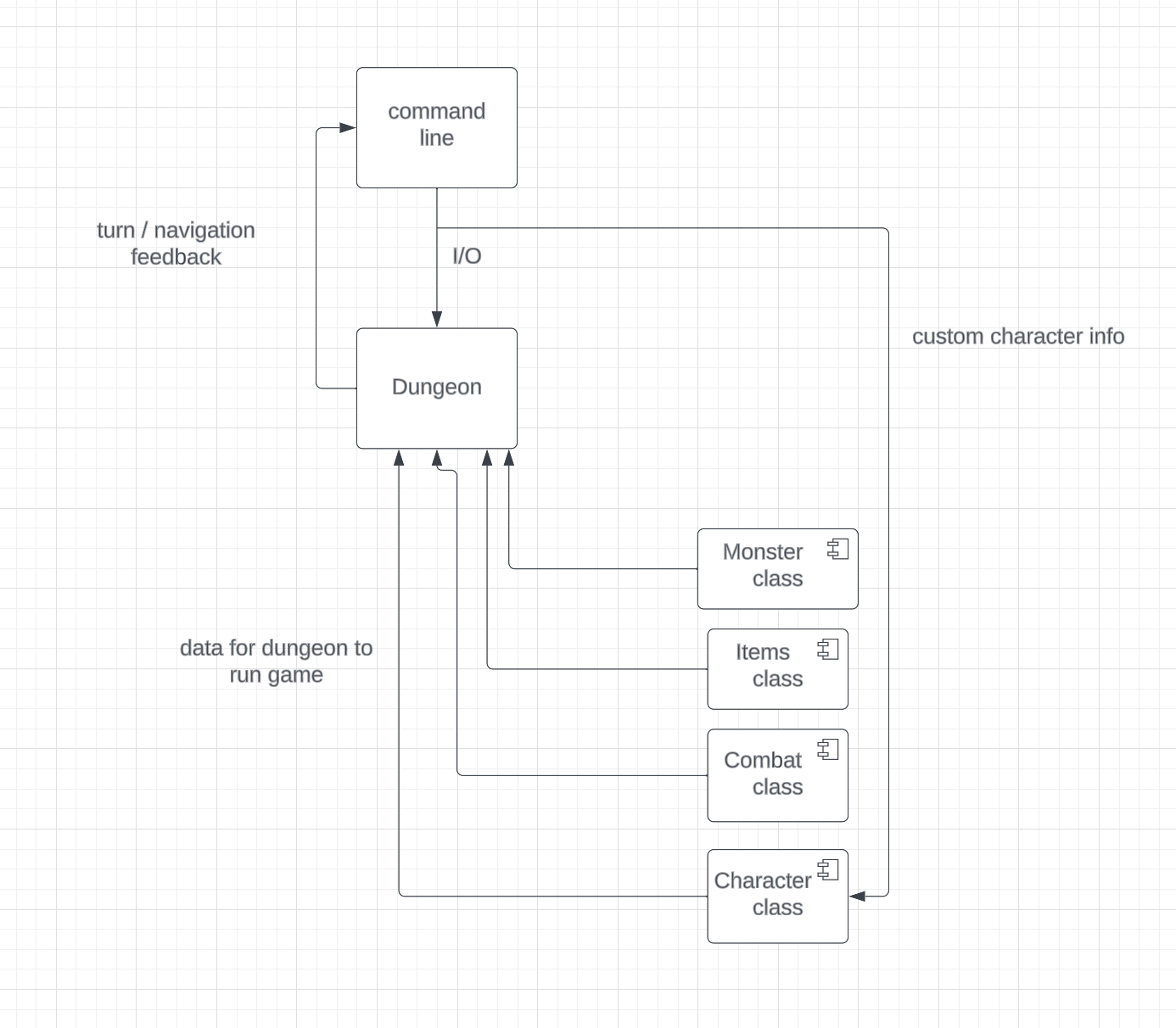
Interactivity will be done with the command prompt, based on readouts given by the map and notifications during events. The game will primarily be a map with sub-events on spots in the map, with class-defined objects interacting within these sub-events.

### Component Design



The Dungeon class contains the bulk of logic, pulling from objects of the other 5 classes. I/O controls player actions.

### Data Flow



The user will create his character based on I/O data, and interact with the game via I/O to the dungeon class. The dungeon will then speak back to the player about combat state, as well as map information.

## System Components <Milestone 3: System Implementation>

[*Include a component sub-section for each component in the architecture diagram. Each component subsection will include a class diagram*]

### Component [Component Name 1]

[*A short description of the component*.]

[*An EA class diagram of the component that includes method parameters. Include the name of the team member that created the diagram in EA.*]

### Component [Component Name 2]

[*A short description of the component*.]

[*An EA class diagram of the component that includes method parameters. Include the name of the team member that created the diagram in EA.*]

### Component [Component Name n]

[*A short description of the component*.]

[*An EA class diagram of the component that includes method parameters. Include the name of the team member that created the diagram in EA.*]

## Design Pattern <Milestone 3: System Implementation>

[*Class diagram of design pattern incorporated into the project. Pattern must be specific to the project and not a general design pattern class diagram. The project must include at least design patterns covered in class. Include the name of the team member that created the diagram in EA.*]

## Design Pattern <Milestone 3: System Implementation>

[*Class diagram of design pattern incorporated into the project. Pattern must be specific to the project and not a general design pattern class diagram. Include the name of the team member that created the diagram in EA. A second design pattern may be included for bonus points.*]

# System Implementation <Milestone 3: System Implementation>

[*In the table below, include a row for each component in your System Architecture diagram. In the second column, list the programming language(s) used to implement the component and the what % of that programming language is used in the implementation. In the third column, list the team member(s) that implement the component and what % of that implementation was completed by that team member. IMPORTANT NOTE: All architectural components must be implemented by an object-oriented programming language: Java, C++, or C#.*]

|  |  |  |
| --- | --- | --- |
| **Architectural Component** | **Programming Language(s) %** | **Team Member(s) %** |
| *[Data Manager]* | *[C++ (45%)*  *Java (55%)]* | *[Mickey Mouse (15%)*  *Donald Duck (20%)*  *Daisy Duck (40%*  *Pluto (25%)]* |