*Dunes of the Far Lands: A text-based adventure game written in C++.*

*Group 1*

Project Portfolio

*November 13th, 2023*

[Introduction 2](#_Toc125988975)

[The GROUP ONE Team 3](#_Toc125988976)

[System Requirements 4](#_Toc125988977)

[Requirements 4](#_Toc125988978)

[User Stories 4](#_Toc125988981)

[User Story #1 4](#_Toc125988982)

[User Story #2 4](#_Toc125988982)

[Project Management 5](#_Toc125988983)

[Continuity of Operations Plan (COOP) 5](#_Toc125988984)

[Project Plan 0](#_Toc125988985)

[System Architecture Design and Development < Milestone 2: Architecture> 0](#_Toc125988986)

[System Implementation <Milestone 2: Architecture & Milestone 3: System Implementation> 0](#_Toc125988987)

[Project Postmortem <Postmortem> 0](#_Toc125988988)

[Project Wins 0](#_Toc125988989)

[Root Cause Analysis 0](#_Toc125988990)

[Lessons Learned 0](#_Toc125988991)

[System Design <Milestone 2: System Architecture> 1](#_Toc125988992)

[System Architecture <Milestone 2: System Architecture> 1](#_Toc125988993)

[Component Design 1](#_Toc125988994)

[Data Flow 1](#_Toc125988995)

[System Components <Milestone 3: System Implementation> 1](#_Toc125988996)

[Component [Component Name 1] 1](#_Toc125988997)

[Component [Component Name 2] 1](#_Toc125988998)

[Component [Component Name n] 1](#_Toc125988999)

[Design Pattern <Milestone 3: System Implementation> 1](#_Toc125989000)

[Design Pattern <Milestone 3: System Implementation> 2](#_Toc125989001)

[System Implementation <Milestone 3: System Implementation> 3](#_Toc125989002)

# Introduction

The main goal for our project is to create a terminal-based interactive game with the C++ programming language. Our plan is to make an adventure game in the same vein as classic text-based adventure games, such as Zork I: The Great Underground Empire. This will require the program to take in user input, parse text strings, output story events from a file, as well as save and update the current game state so that the game can respond to the player’s actions. In order to implement these systems, we will be utilizing the C++ programming language and the C++ standard template library (STD).



Core Features:

* Read & Parse User Input
* Read & Parse Story Elements from a File
* Output Game Events
* Update Game State

Viable Features:

* Inventory System
* Save Game System

Stretch Features:

* Combat System
* Basic Item Combination System
* Audio Output for Events

# The Group One Team

Project GitHub link: <https://github.com/hkaiserteaching/csc3380-fall-2023-project-group1>

Group One is made up of six people:

* Connor Morris
  + Role for Milestone 2 and 3 is TBD.
* Lane Durst
  + Role for Milestone 2 and 3 is TBD.
* Shawn Russell
  + Role for Milestone 2 and 3 is TBD.
* Maureen Sanchez
  + Role for Milestone 2 and 3 is TBD.
* Logan Remondet
  + Role for Milestone 2 and 3 is TBD.
* Yu Joo
  + Role for Milestone 2 and 3 is TBD.

The leader for Milestone 1 is Connor Morris; the leader for Milestone 2 and Milestone 3 is TBD.

# System Requirements

## Requirements

Any computer running Windows 7+ or any recent Unix-based OS on which C++ can compile.

## User Stories

### User Story #1

*As a Gamer, I want to make decisions in the game, so I can feel a sense of agency.*

### User Story #2

*As a Developer, I want to access different story files, so I can easily update and make changes to the game’s story.*

# Project Management

## Continuity of Operations Plan (COOP)

Group One plans on communicating and coordinating via Discord, with meet-ups in person if required. If someone suddenly becomes unavailable for a period of time, the workload they were assigned will be evenly divided among the rest of the group members. Said person will also be kept informed about the project’s status while they are unavailable so that they can rejoin the group quickly once they’re available again. If someone becomes permanently unavailable, via illness or dropping the class, the same steps of spreading their workload out among the rest of the team will be taken.

## Project Plan

### System Architecture Design and Development < Milestone 2: Architecture>

Milestone 2 (Architecture): 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 | Working File System |  | Medium |  | Sep. 19th | Sep. 20th |  |  |
| 2 | Visual Output |  | Low |  | Sep. 22nd | Sep. 24th |  |  |
| 3 | User Input | 2 | High |  | Sep. 24th | Sep. 30th |  |  |
| 3.1 | Take and Store User Input |  |  |  |  |  |  |  |
| 3.2 | Parse User Input | 3.1 |  |  |  |  |  |  |
| 3.3 | Update Game State | 3.2 |  |  |  |  |  |  |

### 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** |
|  |  |  |  |  |  |  |  |  |

## 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 <Milestone 2: System Architecture>

[*Include a short (1-2 sentences) statement about the system design*.]

## System Architecture <Milestone 2: System Architecture>

[*A short description of the system architecture.*]

### Component Design

[*Insert image of system architecture component diagram. Include the name of the team member that created the diagram in EA.*]

[*Architecture overview, to include user I/O, external data sources, and major system components.* ]

### Data Flow

[*Insert image of system architecture data flow diagram. Include the name of the team member that created the diagram in EA.*]

[*Architecture data flow discussion: a high-level description of the data between both internal major components and external data sources.*]

## 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%)]* |