COMP2116: Software Engineering

Final Project Guideline

Class:221

Name: KEONG MAN HEI

Student ID: P2001258

Catalog

[1 Overview of the project 1](#_Toc13427)

[2 Masterplanningand Development Environment 1](#_Toc11200)

3 Development Process Options 2

4 Time Planning ..........................................................................3

5 Reference..................................................................................3

**1 Overview of the project**

**1.1Game Description**

This project will develop a game called ‘Push Sagittarius’, which is similar to the traditional Push the Box game, but with unique elements in terms of screen style, level design or special mechanisms. Players need to control their characters to push objects (e.g. Sagittarius or other elements) in a limited space to achieve the level objectives. This not only hones the player's logical thinking and strategic planning skills, but also brings the nostalgic fun of classic puzzle games.

**1.2 Game Objectives and Market Positioning**

Target Audience: Players who love puzzle and solving games; people who are interested in traditional and innovative gameplay.

**1.3 Project Objective**

To demonstrate the team's ability to collaborate on requirements analysis, design, development, testing and deployment while implementing the complete development process of game design and software engineering, and to deepen the understanding of Agile development process.

**2Masterplanningand Development Environment**

**2.1 Development Languages and Tools**

Python is used as the main programming language, and the game logic is implemented with Pygame, drawing reference. Choose VS Code as the development environment, and make use of its rich plug-ins (e.g. Python linting,) to improve development efficiency.

**2.2 Individually developed considerations**

As the project will be completed independently, it requires a rigorous time planning and self-feedback process. You can develop iteratively and at your own pace.

**3 Development Process Options**

**3.1 Demand analysis**

Define the basic gameplay, level design, main flow, and required features of the game.

Describe how the user interacts with the game and identify the minimum usable product (MVP) of the software.

**3.2 System Design**

To complete the initial system architecture design, it is recommended to adopt the MVC model:

Model:Responsible for managing the game data (e.g. level configurations, character status).

View:Responsible for presenting the game screen and animation effects.

Controller: Handle user input and game logic.

**3.3 Practical**

Use VS Code's real-time debugging capabilities to continuously test and iterate on game features.

**3.4 Testing and Bug Fixing**

Write simple unit tests or simulate the game flow through manual testing to check for functionality, but there may be some bugs.

**4 Time Planning**

|  |  |  |
| --- | --- | --- |
| **Stage** | **Tasks** | **Estimated time** |
| **Requirements analysis** | **Identify game functionality, scale, and MVP** | **1 week** |
| **System design** | **Writing program** | **2-4week** |
| **Implementation** | **Implement core game logic, interface, etc** | **2-4 week** |
| **Testing** | **Gaming Test** | **1 week** |
| **Documentation and Demo production** | **Write README.md, record and edit demo video** | **1 week** |

**5 Reference**

Characters, chests and walls of image:<https://kenney.nl/assets/sokoban>