Software Requirements Specification

for

Trivia Maze

Version 1.0 approved

Prepared by Ji Lu, Jesse Flores, Xiying Long

TCSS 504 GROUP 3

February 3, 2023

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Project Scope 1

1.3 References 1

2. Overall Description 1

2.1 Product Perspective and Product Features 1

2.2 User Classes and Characteristics 1

2.3 Operating Environment 2

2.4 Design and Implementation Constraints 2

2.5 User Documentation 2

2.6 Assumptions and Dependencies 2

3. System Features 2

3.1 System Feature 1 2

3.2 System Feature 2 (and so on) 3

4. External Interface Requirements 3

4.1 User Interfaces 3

4.2 Hardware Interfaces 3

4.3 Software Interfaces 4

4.4 Communications Interfaces 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 4

5.3 Security Requirements 4

5.4 Software Quality Attributes 4

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: Issues List 5

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Ji Lu, Jesse Flores, Xiying Long | February 3, 2023 | Initial Draft | 1.0 |
|  |  |  |  |

# Introduction

## Purpose

The Software Requirement Specification (SRS) for Trivia Maze has been developed and served as a guideline to all users as well as team members. It delineates product’s functionality, features, external requirements and Non-functional requirements. Besides that, it also reflects design philosophy.

## Project Scope

The project is aimed to delivering a graphical user interface game named Trivia Maze to the user. It will help user navigate through the maze which is consists of different size of rooms from entrance to exit. To maximize entertainments of the game, one question at each room is prepared for the user. User has to answer the question correctly to pass the room and eventually win the game; otherwise, game will be terminated. In order to achieve the goal, project team has to implement but are not limited to developing a Unified Modeling language (UML) diagram, employing Model View Controller (MVC) design pattern, and deploying SQLite database.

## References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

# Overall Description

## Product Perspective and Product Features

As mentioned above, the maze is composed of different size of rooms at a minimum 4-room by 4-room, however, it will not have an upper limitation. The difficulty level is depending on user’s choice. Each room will have one door and one question. In order to successfully pass the room, the user has to correctly answer the question. If the user could not answer the question correctly, door will be locked permanently. Questions will be presented in different ways that includes multiple choice questions, True/False questions, and even fill the blank questions. In addition, questions are well developed by project team and it will cover different categories.

## User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy.>

## Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

## Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

## User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

3.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

3.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

3.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# External Interface Requirements

## User Interfaces

1. The TriviaMaze GUI interface shall have one menu system. The menu system will contain at least two tabs which are File and Help. Under File tab, it shall have choices which include Start New Game, Save Current Game, Load Last Game and Exit Game. Under Help tab, it shall display About and Game Instruction choice.
2. The GUI interface shall have a legend section to display what all key symbols stand for to the user
3. The GUI interface shall include a map section which permit the user navigating through from entrance to exit.
4. The GUI interface shall also have a section which display the information of the current room.
5. The GUI interface shall include a question section which displays the question provided to the user.
6. Depending on the user’s answer, if the answer is wrong, the door associated with the answer shall not be displayed anymore. If the answer is correct, the user shall automatically move to that room.
7. The GUI interface shall prompt user game is over if all doors are closed, and message of “Do you want to play it again?” shall be displayed.

## Hardware Interfaces

1. All questions used for this project shall be stored in a SQLite database.
2. The TriviaMaze GUI shall be able to save the current game and load the game save from last time.

## Software Interfaces

1. The TriviaMaze\_GUI\_Interface shall pass all necessary information to Class Maze to generate appropriate size of maze.
2. Class TriviaMaze will be called if the user wants to initiate game or get game instruction.
3. Class Player shall get and set information such as points the user has and transmit it back to TriviaMaze\_GUI\_Interface to display the user wins or not.
4. Class Question shall be used to collect question and correct answer associated with it from SQLite database. User’s answer shall also be checked here.
5. Once the size of maze is determined by the user, the information will be passed to Class Room and it shall generate all necessary rooms and set doors in each room.

## Communications Interfaces

1. The TriviaMaze GUI shall display a dialog box to the user to display the information the user wants to know.
2. The TriviaMaze GUI shall send feedback confirmation message regarding the choice made by the user.

# Other Nonfunctional Requirements

## Performance Requirements

1. The maze shall be consist of four by four rooms.
2. Questions shall be loaded from SQLite database.

## Safety Requirements

For project wise purpose, safety requirements does not apply to this project.

## Security Requirements

1. The program shall only notify the user the answer is correct or wrong. It shall not display the correct answer.

## Software Quality Attributes

1. The program shall allow users to exit the game whenever the user decides to do so.
2. The program shall allow users to exit the game with saving their current status.
3. The program shall allow users to load the game from last time they saved.
4. The program shall allow users to restart a game whenever they decide to do so.

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: Issues List

< This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.>