**Software Requirements Specification**

For

Mendeleev’s Marvel

25-10-2023

Prepared by

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll Number** | **Specialization** |
| Shahan Ali | R2142211305 | Gaming and Graphics |
| Mridul Pratap Singh | R2142211483 | Gaming and Graphics |
| Priyanshu Raj | R2142211450 | Gaming and Graphics |
| Ahzam Khan | R2142211273 | Gaming and Graphics |

Department of Informatics

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

Table of Contents

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page No** |
| Table of Content | | 1 |
| Revision History | | 2 |
| 1 | Introduction | 3 |
|  | 1.1 Purpose of the Project | 3 |
|  | 1.2 Target Beneficiary | 3 |
|  | 1.3 Project Scope | 3 |
|  | 1.4 References | 3 |
| 2 | Project Description | 4-10 |
|  | 2.1 Reference Algorithm | 4 |
|  | 2.2 Data/ Data structure | 5 |
|  | 2.3 SWOT Analysis | 5 |
|  | 2.4 Project Features | 5 |
|  | 2.5 User Classes and Characteristics | 6 |
|  | 2.6 Design and Implementation Constraints | 6 |
|  | 2.7 Design diagrams | 6-10 |
|  | 2.8 Assumption and Dependencies | 10 |
| 3 | System Requirements | 11 |
|  | 3.1 User Interface | 11 |
|  | 3.2 Software Interface | 11 |
|  | 3.3 Database Interface | 11 |
|  | 3.4 Protocols | 11 |
| 4 | Non-functional Requirements | 12 |
|  | 4.1 Performance requirements | 12 |
|  | 4.2 Security requirements | 12 |
|  | 4.3 Software Quality Attributes | 12 |
| 5 | Other Requirements | 13-14 |
| Appendix A: Glossary | | 15 |
| Appendix B: Analysis Model | | 15 |
| Appendix C: Issues List | | 15 |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Change** | **Reason for Changes** | **Mentor Signature** |
| 10-08-2023 | Initial project kick-off | Finalized documentation |  |
| 30-08-2023 | Added use case and main structure | Defined project structure |  |
| 06-09-2023 | Implemented data loading from a file | Incorporating external data |  |
| 10-09-2023 | Enhanced element consecutive checking | Better user feedback |  |
| 10-10-2023 | Conducted extensive testing | Ensuring program stability |  |
| 15-10-2023 | Finalized documentation | Preparing for project completion |  |

1. INTRODUCTION

1.1 Purpose of the Project:

The purpose of this project is to create "Mendeleev's Marvel," an educational game designed to help users learn and remember the periodic table's order. The project aims to address the challenge of making periodic table education engaging and interactive, motivating learners to understand and retain the order of elements. This project's motivation is to provide an enjoyable way for individuals, especially students and science enthusiasts, to enhance their knowledge of the periodic table.

1.2 Target Beneficiary:

The prime beneficiaries of this project are:

* Students of chemistry and related fields who want to improve their understanding of the periodic table.
* Educators who may use the game as a teaching aid in their classrooms.
* Anyone interested in science and chemistry looking for a fun and interactive way to learn about the periodic table.

1.3 Project Scope:

"Mendeleev's Marvel" is a game application aimed at making the periodic table more accessible and memorable. The software's area of application is primarily educational, focusing on periodic table learning and retention. Key objectives and goals include:

* Creating a game that encourages users to select elements in the periodic table order.
* Providing an engaging and fun user experience to motivate learning.
* Ensuring factual accuracy in the information presented.
* Developing a user-friendly and efficient game that runs on various platforms.
* Complying with open-source principles, allowing accessibility to the source code for potential contributions and modifications.

1.4 References:

No external references are included in this SRS. All information and requirements are based on internal project knowledge and objectives.

2. PROJECT DESCRIPTION

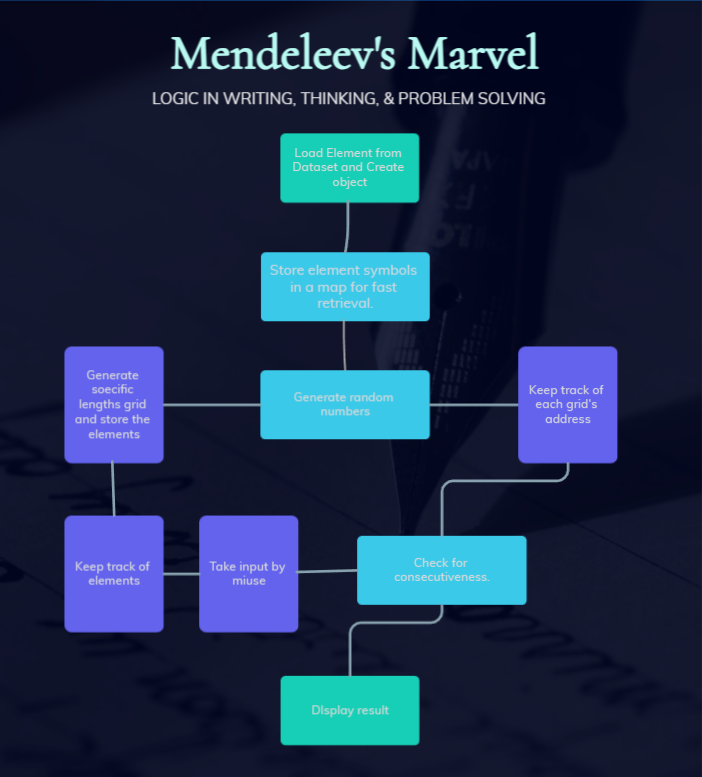
2.1 Reference Algorithm:

Algorithm: The reference algorithm for "Mendeleev's Marvel" is a game that tests the player's knowledge of the periodic table. Players select elements in the correct order to earn points. The game stores elements in a data structure, ensuring that the chosen element follows the periodic table's order.

Data structure used are Hash maps, vectors (one dimensional and two dimensional) and object classes.

The code is relatively unique and does not follow a typical algorithm or standard algorithmic pattern. It combines various elements of C++ programming, including file input/output, random number generation, and grid manipulation, to create a specific game.

The Defined algorithm will be like:



2.2 Characteristic of Data:

* Dataset: The dataset consists of elements from the periodic table with properties such as atomic number, name, and symbol.
* Data Sources: Primary data sources include the periodic table, while secondary sources may include additional data about the elements.
* Sampling: As the data is predefined, no specific sampling techniques are required.
* Data Processing: The project does not involve significant statistical data processing.

2.3 SWOT Analysis:



2.4 Project Features:

* Game Mechanics: Allows players to choose elements in the periodic table order.
* Scoring System: Awards points for correct selections.
* Game Over: Ends the game when the player makes an incorrect choice.
* Educational: A tool to reinforce knowledge of the periodic table.

2.5 User Classes and Characteristics:

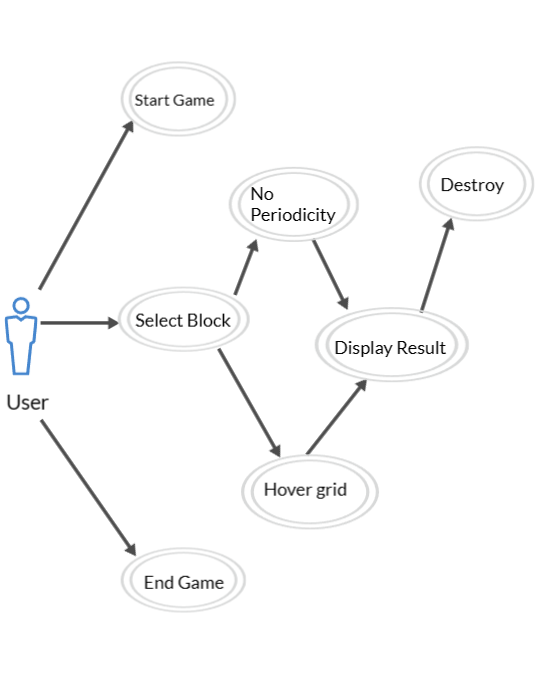
* Players: Individuals who want to learn or reinforce their knowledge of the periodic table.
* Educators: Teachers and educators who may use the game as a teaching aid.
* Developers: Those who may contribute to or modify the game.

2.6 Design and Implementation Constraints:

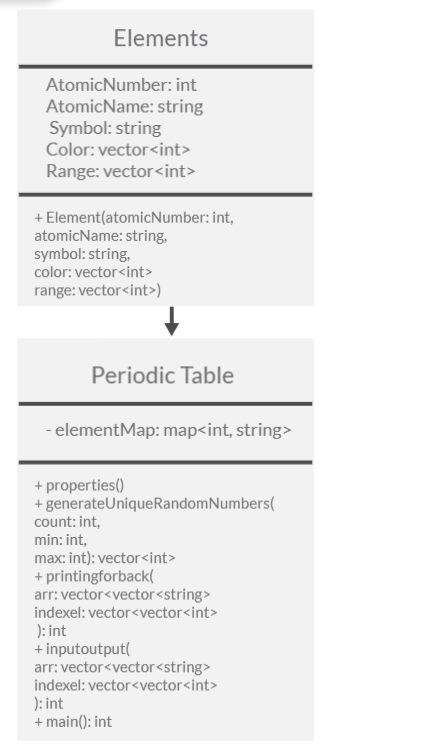
* Hardware: The game should run on standard PC hardware.
* Interfaces: No significant interfaces to other applications.
* Technologies: C++ for programming and OpenGL for graphics.
* Language: The game will primarily be in English and the code is in C++ only.
* Security: No specific security considerations for this game.
* Design Standards: Follow C++ and OpenGL coding standards and object oriented programing.

2.7 Design Diagrams:

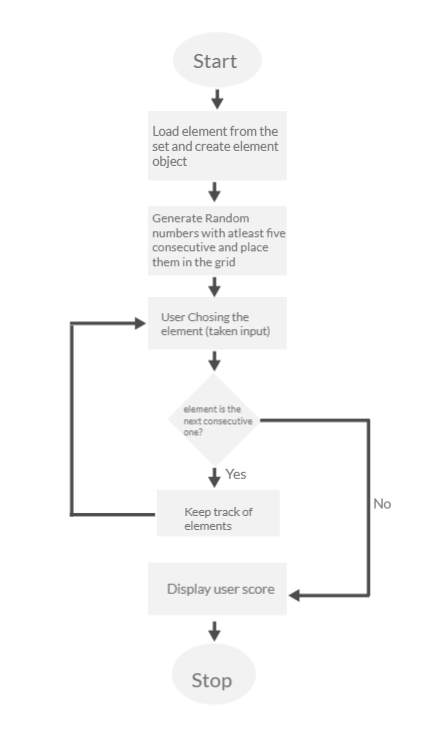
* USE-Case Diagram:



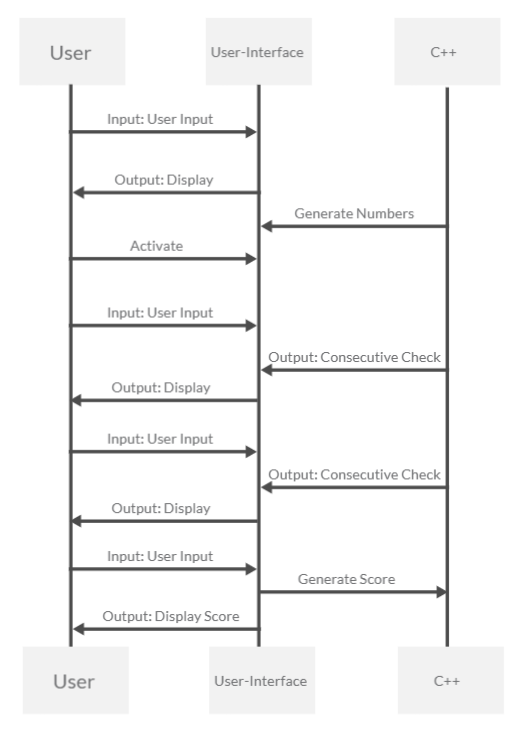
* Class Diagram:



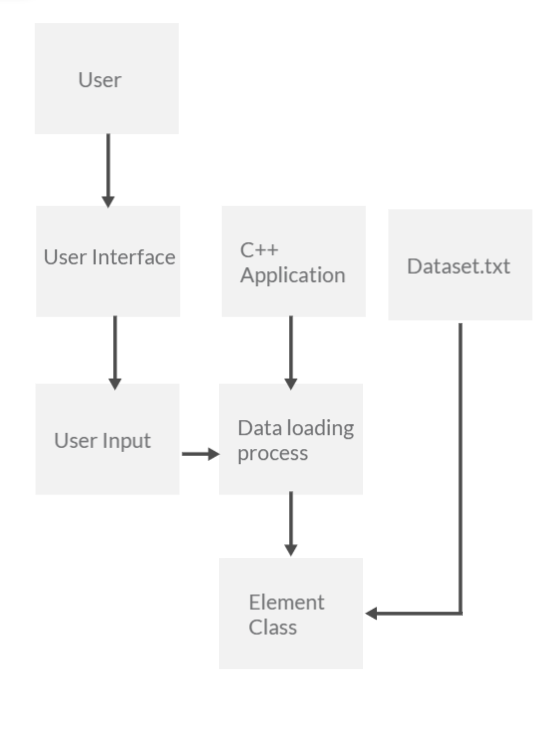
* Activity Diagram:



* Sequence Diagram:



* Data Flow Diagram:



2.8 Assumptions and Dependencies:

* Assumptions: Assumes users have basic computer skills and understand the periodic table's order.
* Dependencies: The project relies on OpenGL for graphics rendering and the availability of C++ development tools.

3. SYSTEM REQUIREMENTS

3.1 User Interface:

Graphical User Interface (GUI): The software requires a graphical user interface for players to interact with the game. The GUI should provide elements for selecting and confirming the order of elements in the periodic table. Used opengl to do so.

3.2 Software Interface:

* Internal Module Interaction: The software modules (e.g., data processing, random number generation, user input handling) should communicate seamlessly to ensure the game functions correctly.
* OpenGL Interface: The project requires an interface with the OpenGL library to handle graphics rendering.
* C++ Interface: The project relies on C++ for coding and communication between different program components.

3.3 Database Interface:

* No Database: This project does not involve a database management system as it primarily deals with static data from the periodic table created in text file, data is stored in memory rather than in a database.

3.4 Protocols:

* Communication Security: The project does not involve external communication, so communication security protocols are not applicable.
* Encryption: Since the project doesn't transmit data over networks, encryption is not required.
* Data Transfer Rates: Data transfer rates are not relevant as there is no external data transfer.
* Synchronization: The project does not have synchronization requirements, as it operates as a standalone game without external data sources.

4. NON-FUNCTIONAL REQUIREMENTS

4.1 Performance Requirements:

* Responsiveness: The game should respond to user input promptly, with no noticeable delay. The game's performance should not hinder the user experience.
* Smooth Gameplay: The game should run smoothly without stuttering, even on standard PC hardware. Achieving a stable frame rate is desirable.
* Resource Usage: The game should be resource-efficient, utilizing memory and CPU power effectively, to ensure it runs on various systems.

4.2 Security Requirements:

* Data Security: The project does not involve external data transfer, but it should ensure the safety of user data, such as high scores or game settings.
* No Authentication: Since it's a standalone game, user authentication is not required.
* User Data Privacy: The game should not collect or store any personal user data. Users' privacy should be respected.
* External Policies: Comply with any external policies or regulations regarding data protection, even though it doesn't collect personal information.

4.3 Software Quality Attributes:

* Usability: The game's user interface should be intuitive, making it easy for users to understand and play the game.
* Maintainability: Code should be well-documented and structured for easy maintenance and future enhancements.
* Portability: The game should be portable across various platforms, ensuring it can run on different operating systems.
* Robustness: The game should handle unexpected user inputs gracefully, avoiding crashes or errors.
* Reliability: The game should run reliably without frequent crashes or technical issues.
* Testability: Code should be structured to allow for thorough testing, ensuring the game's functionality is free of bugs.
* Adaptability: The game should be adaptable to potential updates or expansions, such as adding more elements or features.
* Availability: The game should be available to users without extended downtime.
* Correctness: The game should function as intended, adhering to the rules and logic of the periodic table.
* Flexibility: The project should be flexible enough to accommodate potential future changes or user preferences.
* Interoperability: While the game primarily operates standalone, it should be able to run smoothly on various system configurations.
* Reliability: The game should not crash frequently and should provide a stable gaming experience.
* Reusability: Code components should be reusable in future projects or game expansions.

5. OTHER REQUIREMENTS

5.1 Educational Content:

* Accuracy: The information related to the elements in the periodic table should be factually accurate and aligned with scientific knowledge.
* Learning Element: The game should have an educational focus, encouraging users to learn and remember the elements' order in the periodic table.
* Engagement: The game should be engaging and fun, encouraging repeated play to reinforce knowledge.

5.2 Game Mechanics:

* Game Over Condition: Clearly define the conditions for ending the game, such as selecting an element out of order.
* Scoring System: Define the scoring mechanism, which should reward players for selecting elements in the correct order.
* User Feedback: Provide feedback to the user after each move, indicating whether the selection was correct and the player's current score.

5.3 User Experience:

* User-Friendly Interface: The game's interface should be user-friendly and visually appealing to enhance the gaming experience.
* Sound and Graphics: The game may include sound effects and graphics to make the gameplay more engaging.

5.4 Platform Compatibility:

* Operating Systems: Ensure that the game is compatible with common operating systems like Windows, macOS, and Linux.
* OpenGL Version: Specify the minimum required version of OpenGL for graphics rendering.

5.5 Licensing and Distribution:

* Open Source: Declare the project's open-source nature, including the licensing terms and where the source code can be accessed.

5.6 Documentation:

* User Manual: Provide a user manual or in-game instructions explaining how to play the game.
* Code Documentation: Document the code comprehensively to assist future developers or contributors.

5.7 Testing:

* Quality Assurance: Define a testing plan that includes unit testing, integration testing, and user testing to ensure the game is error-free.
* Bug Reporting: Specify a mechanism for users to report any bugs or issues encountered during gameplay.

Appendix A: Glossary

* SRS: Software Requirement Specification
* GUI: Graphical User Interface
* OpenGL: Open Graphics Library
* C++: A programming language used for the development of the project.
* Dataset: A collection of elements from the periodic table.
* Element: Each entry in the periodic table, with properties like atomic number, name, and symbol.

Appendix B: Analysis Model

The analysis model for this project involves the creation and maintenance of a data structure for storing periodic table elements and their properties. This model guides the game's functionality and logic.

Appendix C: Issues List

This is a dynamic list of open requirements issues. It may include items related to coding, testing, user feedback, or future enhancements. The issues list is maintained and updated throughout the project's development cycle to track and address any pending tasks, challenges, or improvements.