**TEAM MEMBERS NAME: Vishnu M (PES2UG24CS593),** **Vishal (PES2UG24CS591), Yaseen Mulla (PES2UG24CS607), Yajat Mathur (PES2UG24CS606)**

***WORD SEARCH GAME***

**Software Requirement Specification (SRS) Document**

**Word Search Game**

**1. Brief Description of Project**

The Word Search Game is a console-based mini-project developed in C that allows users to play a grid-based word puzzle game. Players must find hidden words in a character grid based on hints, with increasing difficulty across levels and a special emoji-based bonus round.

**2. Purpose / Goal**

The primary goal of this game is to provide users with a fun and educational word puzzle experience. It tests vocabulary, memory, and pattern recognition skills, while offering levels of increasing complexity and interactivity through hints and a bonus emoji round.

**3. Usefulness / Benefit**

* **For Users:**
  + Improves vocabulary and concentration.
  + Offers a stimulating recreational activity.
  + Provides a progressive challenge with hints and bonuses.
* **For Developers:**
  + Reinforces skills in C programming and file handling.
  + Demonstrates modular design and real-time grid manipulation.

**4. Hardware / Software Involved**

**Hardware Requirements:**

* Standard PC/Laptop
* Minimum 2GB RAM
* 500MB disk space

**Software Requirements:**

* Programming Language: C
* Compiler: GCC
* Operating System: Windows (uses windows.h for colours)

**5. Detailed Feature List**

**Player Module:**

* Enter name
* View and interact with word grid
* Input words to check if correct
* Request hints (up to 2 per level)
* Progress through Easy, Medium, and Hard levels

**Game Logic Module:**

* Grid setup and random word placement
* Hint display mechanism
* Score and penalty calculation
* Dynamic grid highlighting of found words
* Bonus round based on emoji clues

**File Handling:**

* Load word-hint pairs from external .txt files (Easy.txt, Medium.txt, Hard.txt) according to the difficulty.

**Interface:**

* Console-based interface
* Coloured text (green for correct answers)
* Uses ASCII approximation for emojis in bonus round

**6. Test / Demonstration Plan**

**Unit Testing:**

* Test word loading, grid filling, and placement logic
* Test hint retrieval and validation

**Integration Testing:**

* Ensure interaction between user input, grid display, scoring, and hint mechanisms

**System Testing:**

* Verify full game flow from start to final score and bonus round

**User Acceptance Testing:**

* Test with different users to verify usability, fun factor, and clarity of interface

**7. Expected Interaction Interface and Sample Use Cases**

**Interaction Interface:**

* Console-based UI
* Menu-driven, input-driven interactions
* Color-coded grid updates

**Sample Use Cases:**

**Use Case 1: Start Game and Play Easy Level**

* User enters name → Game loads Easy level → User enters words → Game gives feedback → User progresses

**Use Case 2: Request Hint**

* User types HINT → Game provides related hint from file

**Use Case 3: Bonus Round**

* After Hard level, game displays emoji clue → User guesses word → Game awards bonus if correct

**8. Individual Member Contribution**

* **Yajat Mathur:** Designed game logic, implemented grid placement algorithm,
* Developed user interaction flow and hint system
* Integrated file handling for loading word-hint data
* Worked on scoring system, emoji round, and final polishing

**Conclusion:**

*Thank you for providing me this opportunity. Had a lot of fun finding the bugs and correcting them!!*