Criterion B

GUI design, and layout

Figure 1 – Initial GUI design for the login window, showing input fields for username and password. A developer key system is included, allowing clients to access the solver.

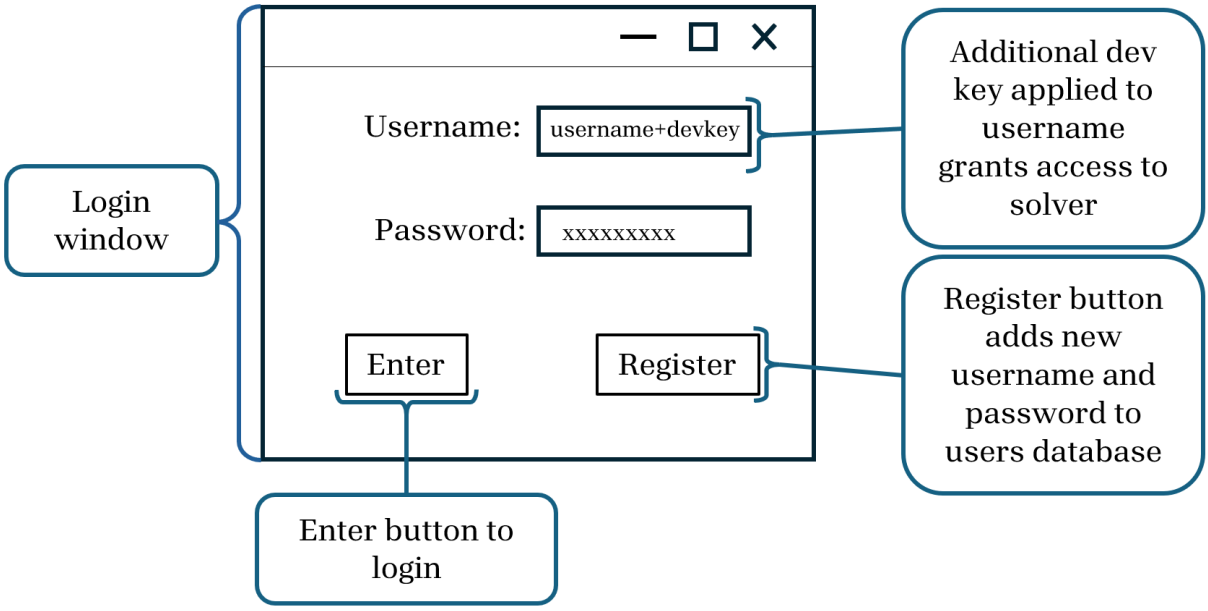


Figure 2 – First iteration of the game GUI. The interface displays 16 word grids, allowing users to enter five-letter words with feedback on correct and incorrect placements. (created in PowerPoint)

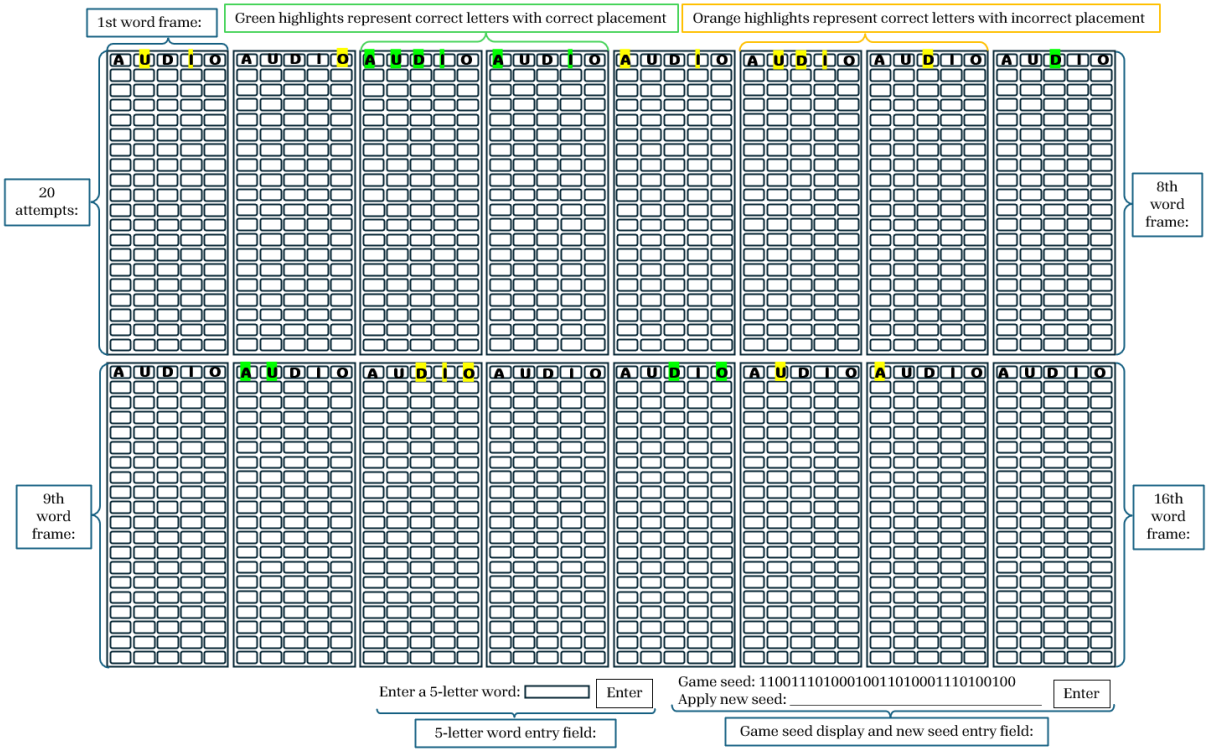
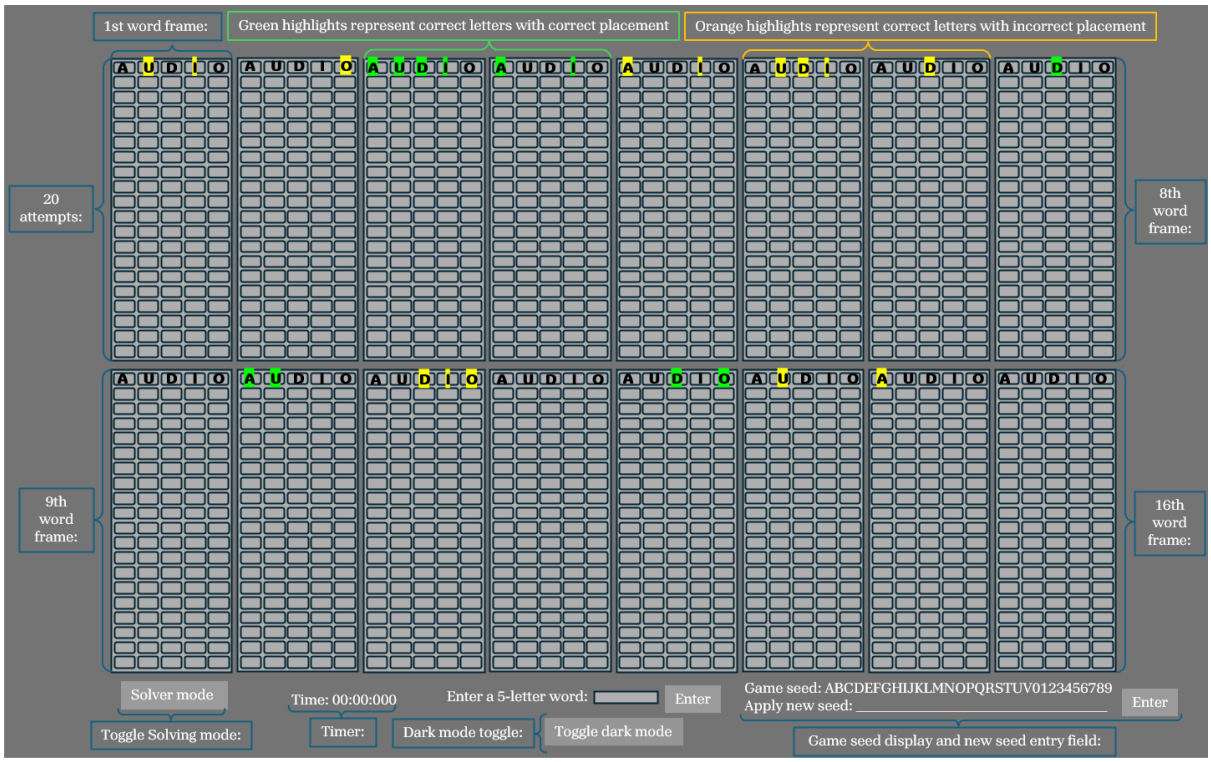


Figure 3 – Changes to the GUI after consultation with the client (Appendix A, Entry 2). A dark mode toggle, a timer, and a solver toggle switch were added to improve functionality. The game seed now follows a structured 32-bit format (A-V, 0-9). (created in PowerPoint)



Flowcharts

Figure 4 – Flowchart representing the main program loop. It outlines how the game initializes, loads word data, and processes inputs.

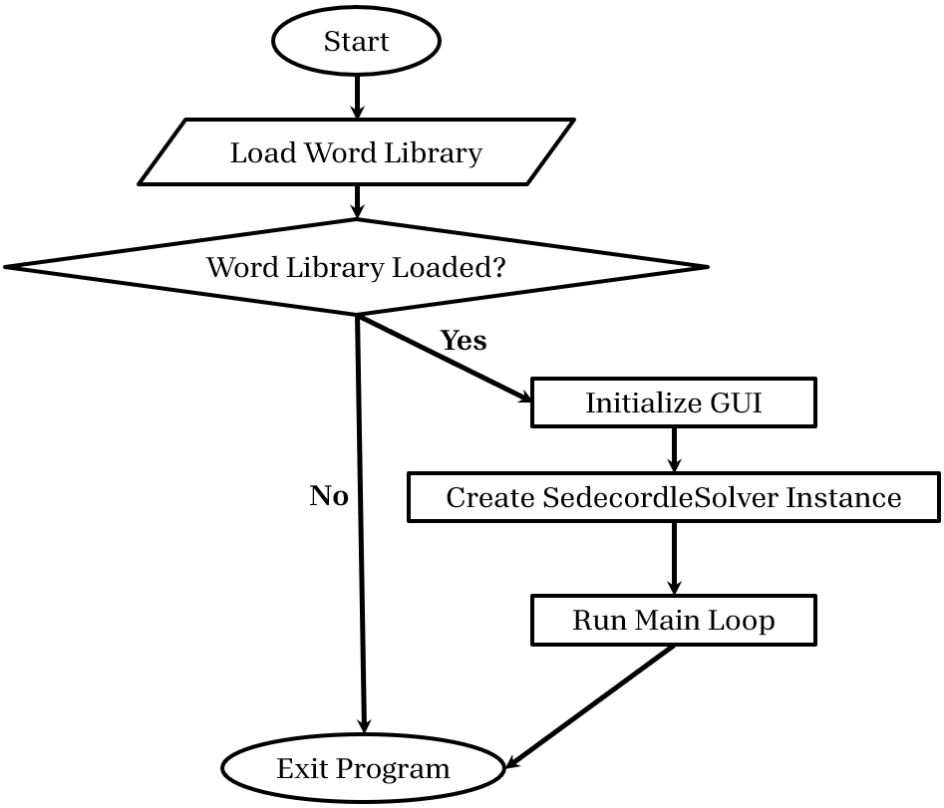


Figure 5 – Flowchart of the login and registration process, detailing user authentication, new user registration, and dev key validation for solver access.

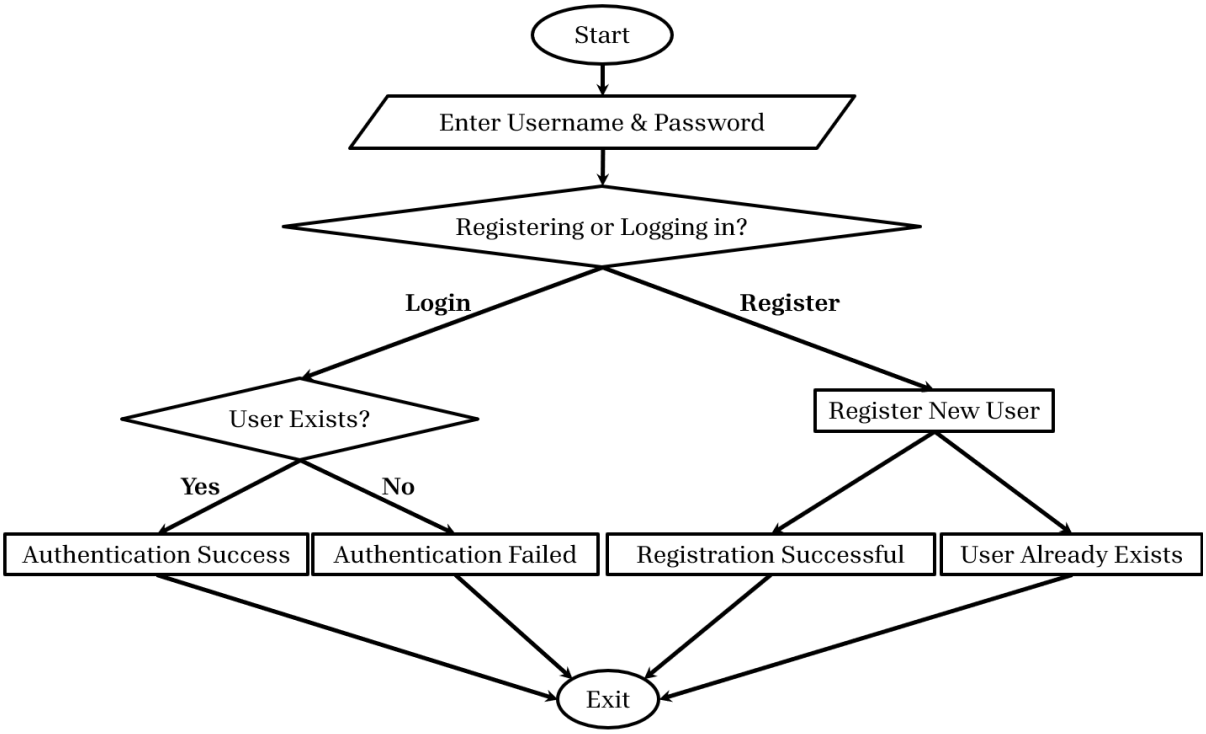


Figure 6 – Word input and checking logic flowchart. Ensures only valid five-letter words are entered, checks against the dictionary, and updates the grid accordingly.

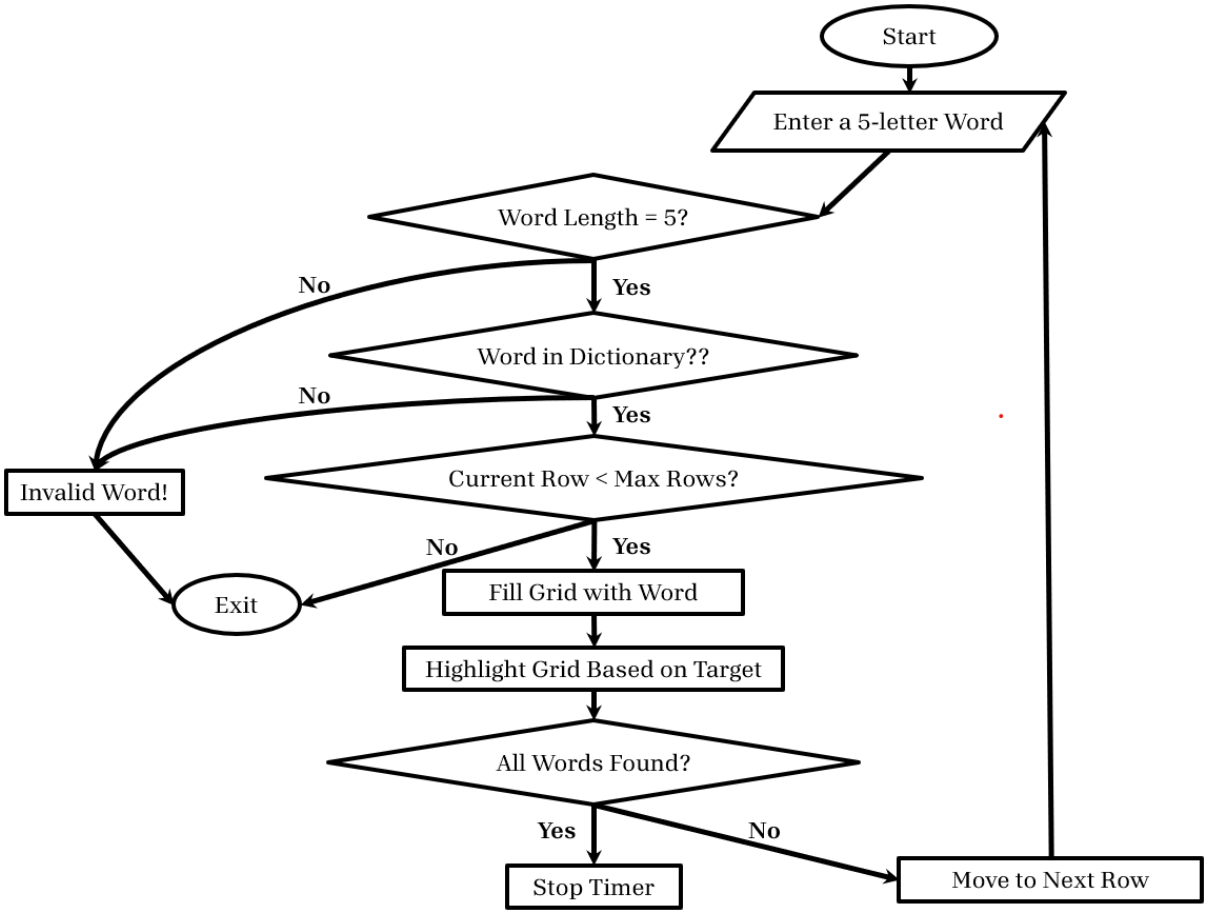
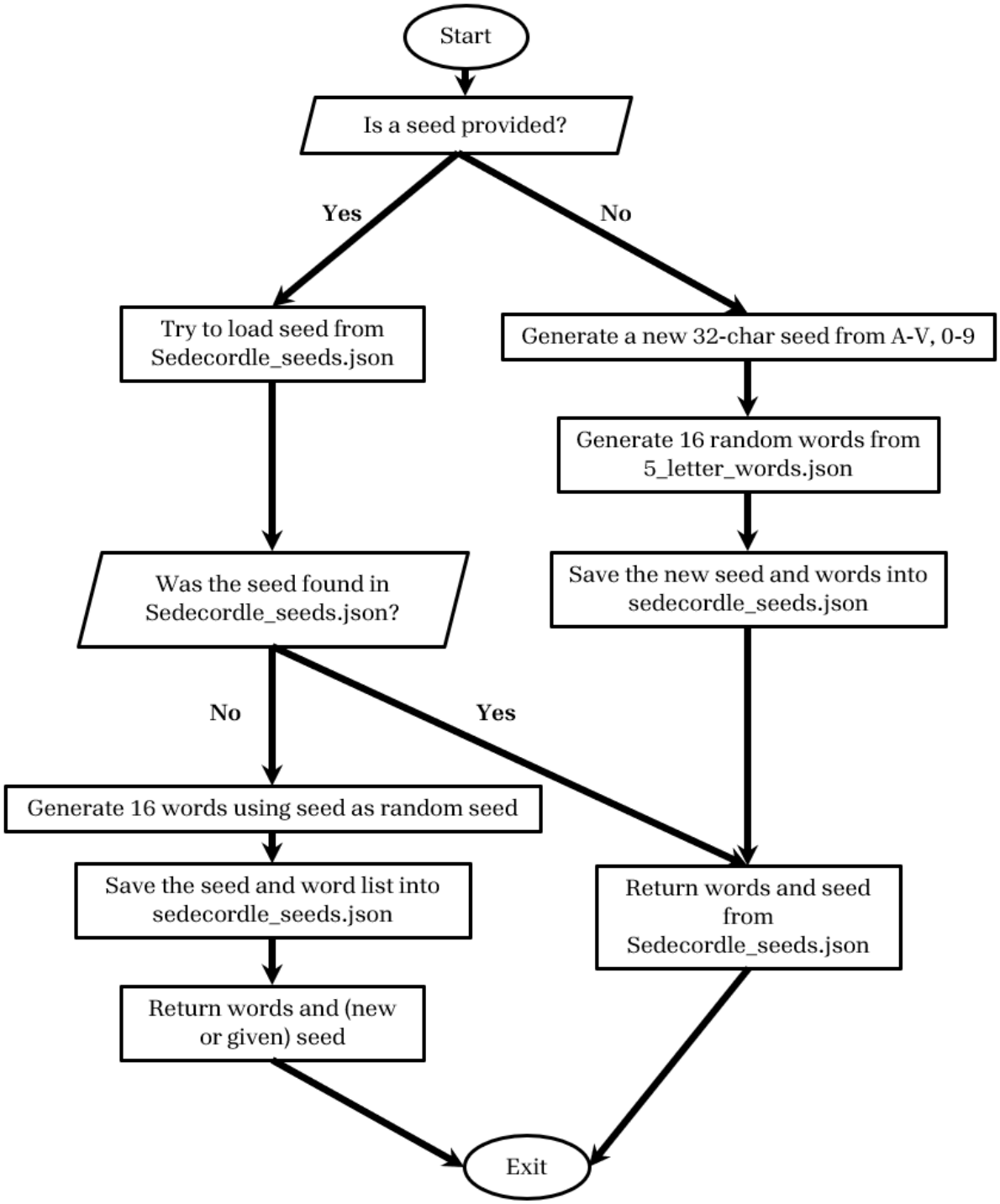
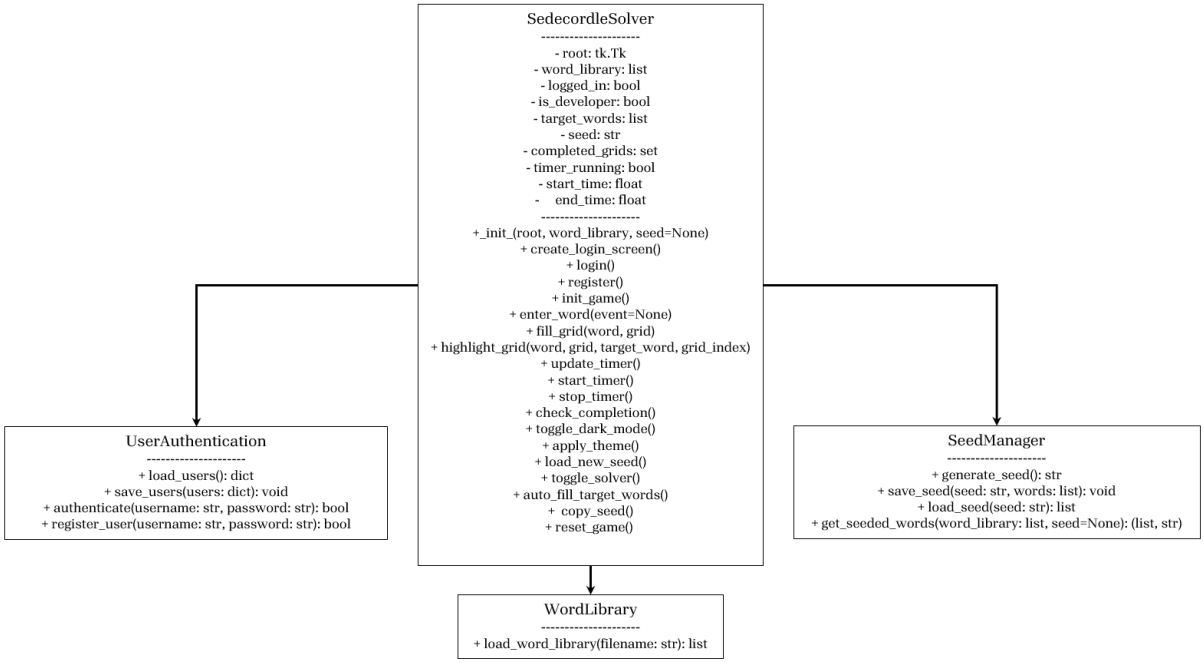


Figure 7 – Seeding logic Flowchart. If seed is provided, its associated words are loaded. If seed isn't provided or does not exist, a new seed is generated along with a random set of 16 words and saved into sedecordle_seeds.json for future use.



UML diagram

Figure 7 – UML diagram representing the class structure of the Sedecordle Solver. Displays key components, including UserAuthentication, SeedManager, and WordLibrary, showing how different parts of the program interact.



Test plan

Action to be tested	Test method	Expected result	Success criteria
Users can register and login	Attempt to register a new user and log in with valid credentials	User is successfully registered and can log in without errors	1
Clients with a dev key can register and login and be granted access to the solver	Register a user with a dev key and check if solver access is granted	Solver mode is enabled for dev key users	1
GUI is simple and easy to use	Ask multiple users to navigate the interface and provide feedback	Users can understand and use the GUI without confusion	2
Every sequence of 16 different words has a unique game seed	Generate multiple game seeds and compare word sequences	Each seed corresponds to a unique set of 16 words and will generate the same sequence of words every time for every user.	3
Timer functions as specified	Start a game and observe if the timer runs and records time correctly	Timer starts upon first word entry, updates, and stops properly upon solving all 16 words.	4
All 16 words can be solved in a few seconds	Run the solver on newly generated game seeds and time the response	Solver finds all words within seconds	5
Solver doesn't take more than 16-17 attempts to solve all 16 words	Run multiple test cases with different word sets	Solver consistently finds all words within the expected number of attempts	6

Game and solver function consistently without crashes or errors.	Run multiple sessions with different inputs and stress-test edge cases	The program runs without crashing or freezing	7