Eastern Oregon University

Wordle Design Specification Document

Green Team

Project Title: Wordle Game
Project Status: ~~Release~~Release
Version: 1.1
Date: 06/15/2024
Institution: Eastern Oregon University, Computer Science Department

Table of Contents

Design Overview

Green Team's Wordle Game entails the creation of an interactive web-based version of the popular Wordle game. Users guess a five-letter word and receive immediate feedback on the accuracy of their guesses. This feedback is visually represented, helping players understand how close their guess is to the target word. The project employs a robust three-tier architecture, consisting of a web front end, a REST API server, and a database server. When a user accesses the Wordle Game, they are presented with a web interface where they can start a new game without needing to log in. The user types a five-letter word guess using either their keyboard or the on-screen buttons. After entering their guess, the user presses the enter key to submit it. The game then provides immediate visual feedback by changing the colors of the letters in their guess: green for correct letters in the right position, yellow for correct letters in the wrong position, and white for incorrect letters. The user has six attempts to guess the correct word, with each guess updating the on-screen feedback. If the user guesses the word correctly within six tries, a "You Win" message is displayed; if not, a "You Lose" message appears. The user can then choose to restart the game and play again. This document provides a comprehensive overview of each system component, detailing their functions and interactions.

System Components

The Web Front End, REST API server, and Database should all be running on the same machine

- Web Front End
    - The web front end is the user-facing part of the application. It is designed using HTML, CSS, and JavaScript to create a responsive and interactive user interface. This interface is crucial as it directly impacts user experience
    - Functionality:
        - User Input:
            - Users can input their guesses using either the physical keyboard or on-screen buttons. All guesses and words are 5 letters and the user gets a maximum of six guesses
        - Feedback:
            - After each guess the system provides visual feedback by changing the colors of the letters
                - Green- The letter is in the word and in the correct position

        - Yellow - The letter is in the words but in the incorrect position
        - White - Indicates that the letter is not in the word at all
    - The digital keyboard is also updated in correspondence with the accuracy of the guesses
        - Green - the letter is in the word
        - Grey - the letter has been used
  - Game Control:
    - The front end includes controls to start a new game, reset the current game, and handle end-of-game scenarios
- REST API Server
  - The REST API Server acts as a bridge between the web front end and the database server. It is implemented using Python and Flask Restful
  - Functionality:
    - Request Handling:
      - The server processes requests from the front end, such as validating a word guess or fetching the solution word for the current game session
    - Endpoints:
      - '/generate_user_id': Generates a unique user ID for each game session, ensuring that multiple players can play simultaneously without interference.
      - '/check_word': Validates whether the guessed word is correct and returns the appropriate feedback to the front end
      - '/get_solution': Retrieves the current session's solution word front the database
    - JSON Communication:
      - Data exchange between the front end and REST API server is done using JSON
- Database Server
  - The database server is a MySQL server that stores all game-related data and is running its own process
  - Functionality:
    - Data Storage:
      - A table containing the list of all acceptable words for the game
      - Each game session's solution word, randomly selected from the valid words list
      - Records player performance data, including wins, losses, and streaks
    - Data Retrieval:

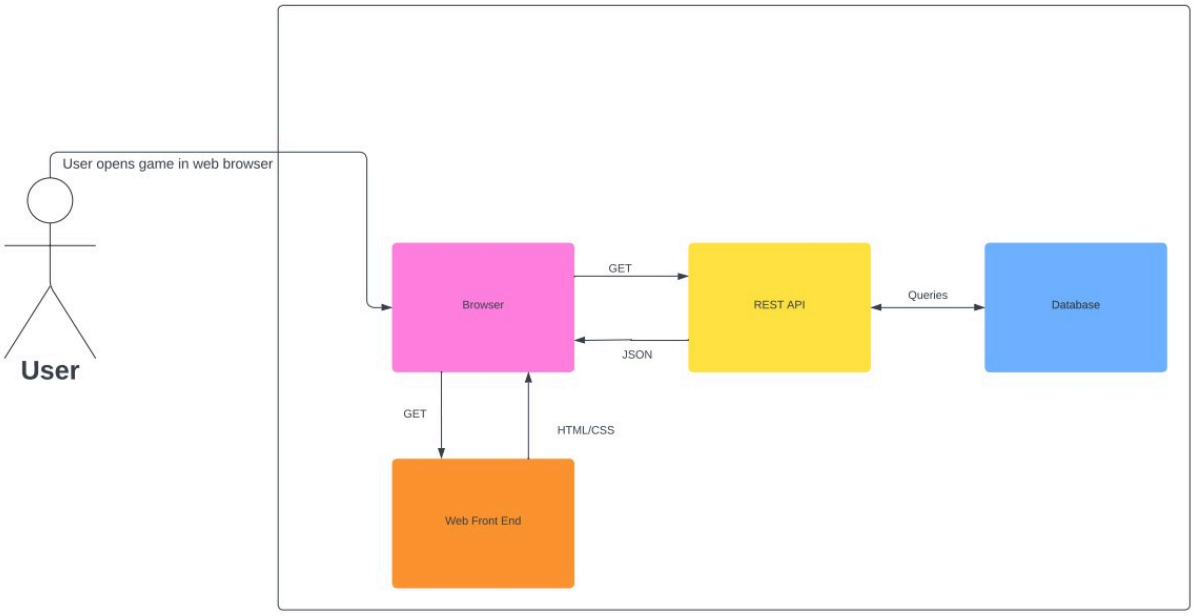
- Queries the database to fetch solution words and validate guesses

Data Flow

The interaction between the system components follows a structured flow:

- Game Start:
    - The user opens the game in a web browser. The front end sends a request to the /generate_user_id endpoint of the REST API server
    - The REST API server generates a unique user ID and returns it to the front end
    - The front end then sends a request to the /get_solution endpoint, passing the user ID to retrieve a solution word
    - The REST API server queries the database for a random valid word, stores it as the solution word for the user ID, and returns it to the front end
- Gameplay:
    - The user inputs a guess and submits it. The front end sends the guessed word and user ID to the /check_word endpoint
    - The REST API server checks if the guessed word is valid. If valid, it compares the guessed word with the solution word
    - The server constructs a response indicating which letters are correct, which are in the wrong position, and which are incorrect
    - The front end receives this response and updates the UI to reflect the feedback, changing the colors of the guessed letters
- Game End:
    - After 6 attempts or a correct guess, the game concludes
    - The front end displays an appropriate message (“You Win” or “You Lose”)
        - The popup message should contain information about the stats accumulated by that particular computer
    - The user can choose to start a new game, which resets the session and repeats the process

System Architecture



The above diagram visually represents the interactions and data flow between the system components. The web front end captures user interactions and sends requests to the REST API server. The REST API server processes these requests, interacts with the database server, and returns the necessary data to the front end.

Conclusion

This high-level design overview provides a detailed look at the architecture and components of the Wordle project. By employing a three-tier structure, the design ensures clear separation of concerns, facilitating easier development, testing, and maintenance. The web front end offers a user-friendly interface, the REST API server handles business logic and data processing, and the database server ensures data persistence and integrity. Together, these components create a cohesive system that delivers an engaging and seamless gaming experience.