Wordingo

CS157A - Team 37
Instructor: Mike Wu

Xiaohan Sun
Andrew Wilson
Alexis Huerta

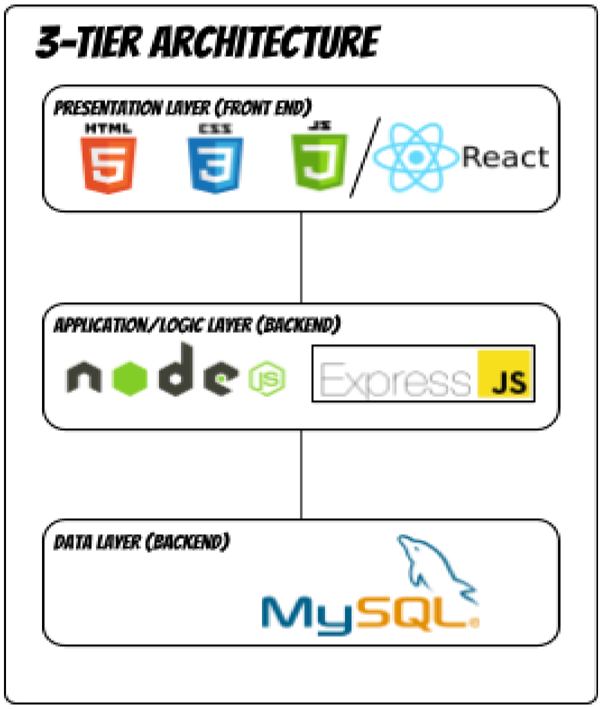
Project Description

Wordingo is a user-defined dictionary web-application, which aims to help users quickly search a MySQL database for word definitions, and add or edit their own words in the database. Our application is free for all users and far more legible with a clean GUI. With multiple functions provided, our web application allows users to find the word they want within only a few seconds as all they need to do is to type the word in the search box and press search.

As for our stakeholders, we will focus on people who work in a field that has a rapidly and dynamically evolving jargon such as medical professionals and software engineers. This is application is important and relevant because someone who works in such a field will often be confronted with terminology that they are not familiar with, and maybe in the position to coin a new word for a process or procedure that they originate. By allowing users to create their own words, our application will, therefore, have a high degree of applicability for these professions. Specifically, this application is intended to be deployed within an organization (such as a large tech company for example) to keep track of internal terminology.

System Environment

Structure of the System



Hardware and Software used

- Git
  - A version-control system for tracking changes in source code during software development
- Github
  - Used to host our remote repository for version control
- Bootstrap 4
  - CSS framework directed to more easily creating responsive, mobile-first front-end web applications
- React.js
  - A JavaScript library/framework for building user interfaces
- Node.js
  - JavaScript run-time environment that executes JavaScript code outside of a browser
- Express.js
  - Minimal and flexible Node.js web application framework that gives a robust set of features for web application and provides HTTP utility methods and middleware to create a robust API

RDBMS

- MySQL 8
  - An open-source relational database management system base on Structured Query Language (SQL)

Application Languages

- Javascript
  - Hypertext Markup Language is the standard markup language for documents designed to be displayed in a web browser
- HTML
  - A standardized system for tagging text files to achieve font, color, graphic, and hyperlink effects on World Wide Web pages

- CSS
    - A style sheet language used for describing the presentation of a document written in a markup language like HTML
- SQL
    - A programming language designed for managing data held in a relational database management system

Functional Requirements

Describe each individual function, functional process and I/O

Functions:

Create Account

- The user shall be able to enter a username, email and password with at least 8 characters to create an account.
- The system should provide a form which allows a user to create an account.

Login

- The user shall be able to log in with valid email and password entered.
- The system should provide a "Sign in" button redirecting the user to the home page after receiving user's correct login information. If email or password is invalid, the system will notify the user with an error message : "Invalid Credentials" and prompt the user to re-enter their information.

Search by Word

- The user shall be able to search words in the search bar.
- The system will show all definitions to the user if the word exists. If the word does not exist, the system will notify the user with an error message: "Word does not exist." and prompt user to re-enter another word.

Search by Alphabetical Browsing

- The user shall be able to click a letter inside the alphabetical list the system provides.
- The system will show all words starting with the letter alphabetically.

Add a Description to an Existing Word

- The user shall be able to add a description to an existing word found in the dictionary.
- The system will store the description in the database and prompt user that “Description is saved successfully!”

Edit Word Definition

- The user shall be able to choose a word with a definition he/she has written and edits the definition.
- The system will catch the change, save it to the database and prompt user that “Change is made!” If the change is not caught, the system will notify the user that “Definition edited fails, please try again.” and direct the user back to the previous page.

Remove Words

- The user shall be able to choose a word to delete.
- The system will provide a “Delete” button for the user. After the user clicks the button, the system will notify the user with the message: “Do you want to delete this word?”. If the user clicks the “Delete” button, the word is deleted from the database; if the user clicks the “Cancel” button, the system will redirect the user to return to the previous page.

Upvote/Downvote Words

- The user shall be able to upvote or downvote a certain definition for a word.
- The system will provide buttons standing for upvote and downvote and store the vote in the database.

Non-functional Requirements

- **Graphical User Interface**

  The graphical user be oriented to be very simple, incorporating a focused set of essential elements. The very first element a user will be confronted with is the login or create an account page. They will be prompted to select if they are an existing user or if they want to create an account.

  Following the conclusion of their login, they are directed to the main application.
  The two navigation elements are the sidebar and search bar. The sidebar will be used to navigate word definitions alphabetically, as well as logout. The sidebar will occupy the left side of the screen.

  The search bar will be used to look up words. It will be placed at the top of the screen. Both navigation elements will be descriptively labeled. By incorporating the two navigation elements on one screen, the user is able to quickly understand the two primary ways to leverage the application, (i.e usability).

  Content will be displayed in the center of the screen. When browsing words, the number of upvotes a particular definition has will be clearly displayed.

  Color will not be utilized in order to define any groupings or functionalities, thus allowing the colorblind to use the service with no difficulty. Additionally, because all elements will be labeled textually, the blind will be able to navigate using audio description software.

  Design principles incorporated will include the Gestalt principle (by grouping the navigation options), Hick's law (by limiting the number of choices presented to the user at one time).

- **Performance Requirements**

  Performance: All user performed actions (clicking a hyperlink and clicking a button) the system will receive a system response within 5 seconds.
  Storage: The database must be of a sufficient size to hold up to 5000 word definitions. This is approximately 25% of an average person's vocabulary, and therefore a more than adequate size to accommodate all the words in a tradespeak or lingo.

- **Security Requirements**

  Security: User accounts will be isolated and password protected. A user account can not be accessed by another user. User passwords will be required to be secure character strings (I.E of not less than 8 characters in length and using a combination of alphabetical and numeric characters).

- **Software Quality Attributes**

  Availability: The system will have a high index of availability. The application should be available 99.9% of the time when operational (i.e, in one week's time, the application should be down for a maximum of ten minutes).

  Usability: A naive (literate) user will be able to understand how to use the primary features of the system (search, add, edit and delete words) by examining the GUI and accompanying text.