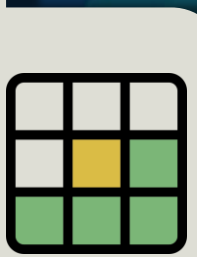


Temu Wordle

Developed By:

Ken Linzmeier, Melanie Porter, Oscar Rodriguez,
Emmett Macias, and Olivia Newton

T

O

O

L

S

What is Temu Wordle?

Temu Wordle is our attempt to create a harder version of Wordle where you can compete with your friends!

With our site, you can:

- Play our six letter singleplayer to try to be on top of the leaderboard!
- Create a profile and add a friend!
- Compete against a friend to see who can solve a word in less guesses!
- View your statistics to see how good you are!

What Tools Were Used?

Project Tracker and VCS Repository

We used GitHub as our VCS repository to store our files and used the GitHub Projects feature to track the progress of our project. It received five stars because everything we needed GitHub to do was easy to utilize and documented well!

What Tools Were Used?

Database

We used PostgreSQL as our database for the users and match data, and because it was easy to access and well documented we gave it five stars!

What Tools Were Used?

IDE

We used Visual Studio Code as our IDE (Live Server Extension) to modify our code in real time and see changes to the site! The addition of highlights to different areas of code made it easy to see, so VSCode gets five stars!

What Tools Were Used?

UI Tools

For our UI tools, we utilized HTML and Handlebars to display our site to the user. Both worked exceptionally and minimized our time working on visuals; therefore, they get five stars!

What Tools Were Used?

Application Server

When coding our wordle page, NodeJS made it easy to implement JavaScript code onto our site and helped organize our code, so it gets five stars!

What Tools Were Used?

Deployment Environment

render

Our deployment environment is Render, and we used it to host our site on the cloud. It receives only three stars because we felt it wasn't well documented, constantly pushes for payment, and some of its features did not work (updating with new commits). However, our site is hosted well after set-up!

What Tools Were Used?

External APIs

When coding our wordle page, we used “Random Word API” to make generating a word easier. It receives four stars only because it goes down sometimes and could make our site not functional.

https://random-word-api.herokuapp.com/home

Random Word API

What Tools Were Used?

Testing Tools

To test some test cases such as whether a page loaded or if an invalid/valid “POST” could be made, we used Mocha and Chai. It only receives two stars due to us using the “render” function often, making it difficult to test certain capabilities and most tests ended up needing to be manual.

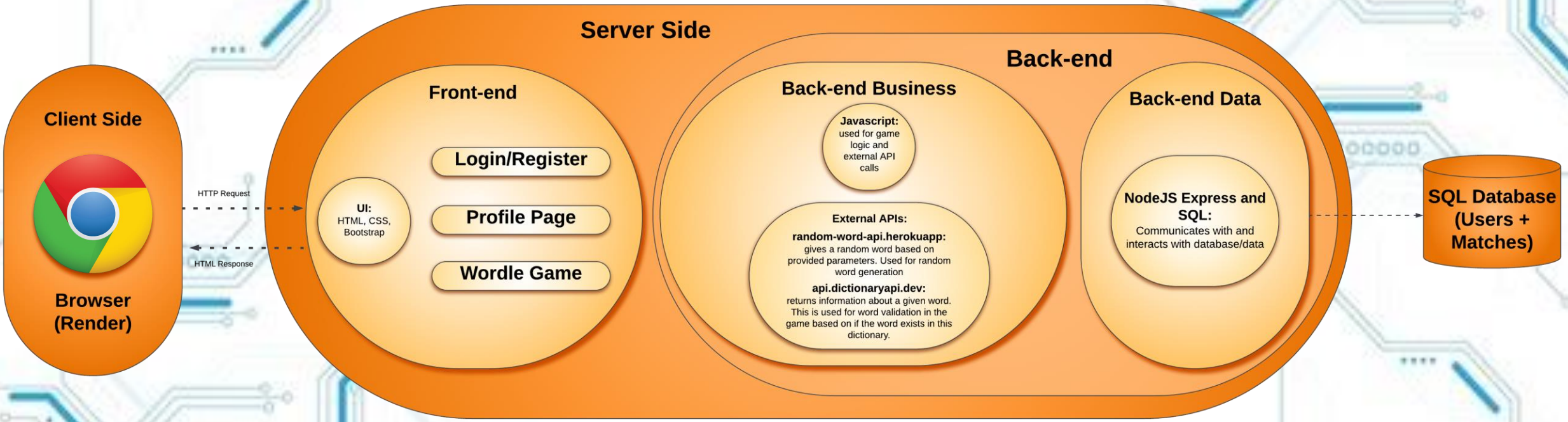
What Tools Were Used?

Framework

We used NodeJS Express for our framework, and it was easily one of the most helpful tools of our project. Being able to GET and POST data was especially helpful, and as such receives five stars!

What Tools Were Used?

Architecture Diagram



Challenges

Bugs/Errors

When coding the site, we found that there were many errors we forgot to account for, like what if someone presses guess after an end screen? What about multiple friend requests? We ended up having to go back often, planning what the site would account for beforehand would have prevented some of these problems in the future.

Match Data

When accounting for match data, the problem of storage came up. We didn't know how to store that data when a match was pending or finalized, but we ended up creating a table of finalized match data that is initialized on friend acceptance and another table where the pending matches were stored with who challenged, who received, guesses of the challenger, and the word.

Live Gameplay

When we finished single player and the profile page, the question of how a live action game could be implemented was brought up. We felt as though we didn't have the resources or understanding yet regarding the cloud, so we adjusted our head-to-head to be a "send and receive" methodology with the same functionality!

Challenges

Bugs/Errors

When coding the site, we found that there were many errors we forgot to account for, like what if someone presses guess after an end screen? What about multiple friend requests? We ended up having to go back often, planning what the site would account for beforehand would have prevented some of these problems in the future.

Match Data

When accounting for match data, the problem of storage came up. We didn't know how to store that data when a match was pending or finalized, but we ended up creating a table of finalized match data that is initialized on friend acceptance and another table where the pending matches were stored with who challenged, who received, guesses of the challenger, and the word.

Live Gameplay

When we finished single player and the profile page, the question of how a live action game could be implemented was brought up. We felt as though we didn't have the resources or understanding yet regarding the cloud, so we adjusted our head-to-head to be a "send and receive" methodology with the same functionality!

Future Scope

Bio/Profile

When viewing your profile, there is currently a default profile picture and a biography, however these features are not displayed to other Users or influence the site. In the future, we foresee adding and implementing a way to view another person's biography or viewing their profile picture on the leaderboard.

User ID

During creation, we didn't implement User IDs as we felt the username was enough, however adding User IDs to our database would allow us to access Users easier alongside new features, such as changing your username.

Live Gameplay

Although the idea was scrapped initially, revisiting the idea and adding both the live option and match option for offline and online play would be a cool feature!

Demo Time!

https://csci-3308-group-project-015-4-fall-2024-w5o1.onrender.com/login