Intro to Computer Science, Wahluke High School

Bonus 4 – Wordle Solver Frontend

Bonus projects are opportunities to independently develop your computer science skills by creating interesting, advanced projects that are not part of the regular syllabus. Note that bonus projects are optional, do not need to be submitted, and are not eligible for a grade.

Overview: Wordle Solver Frontend

During our March 4 culture day, we got to explore software engineering specializations including frontend and backend:

- "Frontend" is the code that users see and interact with directly. This includes the content users see, how the content appears, and how users can interact with it. For example, the user interfaces of websites and apps are considered frontend.

- "Backend" is the code that users do not interact with directly. For example, if you send an email, it goes through multiple backend systems: one system might store the email in a database, another system would scan the email for spam, another system would actually send your email to the recipient, etc.

In working on your Project 3, you have implemented an impressive tool. The code you wrote can function as the backend of a system: you wrote all the logic that controls Wordle Solver.

All that's missing is a frontend. A frontend for Wordle Solver could be a website that utilizes your existing code to drive a visual user interface. We could provide textboxes instead of requiring users to type in commands, we could add a formatted output section for users to see their results, and we could include an attractive layout to please our users.

(continued...)

Side Note: Utility of Functions

Throughout this unit, we have been emphasizing how useful functions are! This bonus content is only possible because of how functions work:

- Functions allow different pieces of code to interact with each other. In this case, you have written a set of functions that implement the logic of Wordle Solver. The instruction team has written code that implements the frontend of Wordle Solver. Because you implemented Wordle Solver as a set of functions, the frontend code can use your functions.

- Computers rely on having well-defined "interfaces" to communicate with each other. In the assignment prompt, we described exactly how the Wordle Solver function interfaces should be implemented: the function names, inputs, and outputs. The frontend code relies on these functions being implemented with those exact names, inputs, and outputs. If the names, inputs, or outputs are slightly mismatched, the code will not work as expected. This is how all complex computer systems work: there are standard "interfaces" for everything including music (mp3 files), video (mp4 files), websites (http protocol), wireless internet (wifi), and anything else you can imagine.

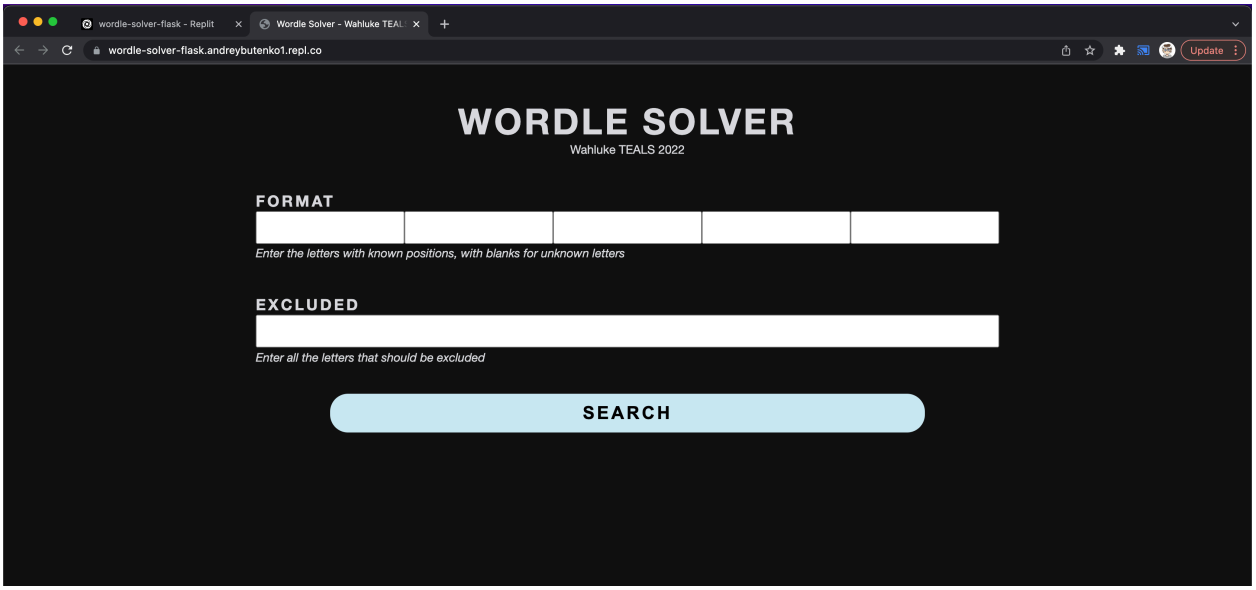
(continued...)

Instructions: Wordle Solver Frontend

1. Start the “Bonus 4 – Wordle Solver Frontend” assignment on Repl. You’ll see that your assignment has a lot of code already provided:
    a. main.py – This is the web server that “glues” together the frontend (user interface) and backend (your code). The frontend accesses the web server, and the web server accesses the backend.
    b. templates/index.html – This is the frontend structure that users see. If you open it, you’ll see an element called header, an element called main, multiple elements called button, and multiple elements called input. Although this programming language may not be familiar with you, it is likely not too difficult to understand the website structure just from the code.
    c. static/styles.css – These are the frontend styles that format the webpage. If you open it, you’ll see keywords like “font-family,” “background-color,” “margin,” and others that define how different elements should appear.
    d. static/scripts.js – This is the frontend code that makes the webpage interactive. If you open it, you’ll see that it implemented with functions having descriptive names and comments like “submitSearch,” “showResults,” and more. Again, although this programming language may not be familiar with you, you will find that it is not too difficult to follow, especially with your Python knowledge.
2. Open the “assignment.py” file. You will modify this file.
3. Copy and paste your Wordle Solver assignment into the bottom of the “assignment.py” file.
4. Make these modifications to the code you just pasted into the file:
    a. Remove any while loops. We used while loops to repeatedly prompt the user and make the project work in the Python console. The web server will call your functions directly, so there is no need for while loops.

(continued...)

5. Press “run!” It will take longer than usual for Repl to start because running a web server is a relatively intensive process.
    a. Once the web server starts, there will be a new pane in your Repl window that shows a webpage.
    b. Click the icon in a corner to open the website directly in a new tab. It should look like this:



(continued...)

Troubleshooting

The web server may not work as expected the first time. This is okay! We're trying to integrate multiple separate, complex pieces of code. Try these troubleshooting steps:

- Verify that all your functions are implemented exactly as described in the assignment prompt. In particular, make sure the names, parameters, and return values are correct.

- Check the Repl console for error messages. Read the error messages and address them to the best of your ability.

- Ask an instructor if you need any help!

Suggested Activities

Great job! In addition to completing the very ambitious Wordle Solver project, you now have a frontend that provides a user-friendly way to run your project. Here are some suggested activities to try now:

- Your Repl web server has a unique URL. You can share this URL with friends and family to show off your cool project!

- Look at the provided index.html, styles.css, and scripts.js files to how they work. Try making a small change to a file, reload the web page, and see what happens! Some examples of small changes:
    - Change some text in index.html
    - Change a font, background color, width, or margin in styles.css
    - Change a function in the scripts.js