CS 300: Advanced Programming Spring 2022
Module Exam 3
Wordle

Programming isn't about what you know, it's about what you can figure out.

Submission Instructions and guidelines:
Submit your entire application as a zip file with the naming convention **<rollnumber>.zip**. **Please make sure that you remove the node_modules before zipping your application.** Apart from the zip file you must submit a PDF file with the naming convention **<rollnumber>.pdf.** In this pdf file, you will write a detailed report on how you developed this application.

Please read the entire document thoroughly before starting the exam. The html and css file for the Wordle board is provided to you. However, it is your choice if you want to use it as reference or write your own html/css.

Plagiarism Policy
The plagiarism policy is as follows:

1. Students must not share the actual program code with other students.
2. Students must be prepared to explain any program code they submit.
3. All submissions are subject to automated plagiarism detection. Students are strongly advised that any act of plagiarism will be reported to the Disciplinary Committee.

Deadline:
The exam is due on LMS at **3PM, 23rd April, 2022.** No late submissions or dropbox submissions will be accommodated.

You have to develop a full stack (MERN) application consisting of the game, Wordle. You can read more about the rules of the game here. However, your application will consist of some extra features along with the functionality of the game. The details of the application are given below.

Your application will consist of the following pages:

1. Homepage
2. Login page
3. Sign up page
4. Leaderboard page
5. Game page (where the actual game will be played)
6. History page

Sign-up page [10 marks]:
This page consists of a simple form, which will take in the **username** and **password,** and a signup button. Upon signing up, you need to check if the username already exists in the database or not, if it does then signup should be unsuccessful since usernames must always be unique. However, if the username is not present then an account is made, the credentials are stored in the database and the user is redirected to the login page where they can then login using the credentials they signed up with.

Login page [20 marks]:
This will be the default page in your application meaning that this page should open when your application starts. It will consist of a simple form, which will take in the **username** and **password,** along with a login button. The username has to be **unique** for all users. A user can login only if their credentials are already present in the database.

After successful login, the user is redirected to the **homepage**. If the login is unsuccessful then you need to show an alert. This alert can be a simple javascript alert or a custom-made alert (this is totally up to you!).

At this page, a sign-up link will also be present. By clicking this link, the user will be redirected to the sign-up page.

Please note that you will have to manage the state of the user to display the username at some of the parts ahead, and use it to store information related to the user in the database.

Note: The user should not be able to visit any of the following pages without logging in:

- Homepage
- Leaderboard page
- Game page (where the actual game will be played)
- History page

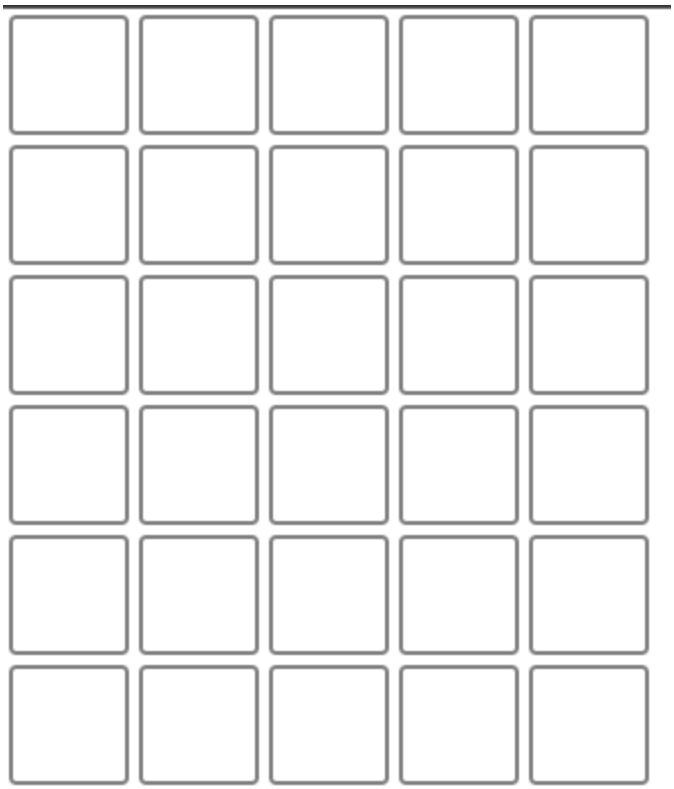
Homepage [5 marks]:
This page will display the username of the user logged in and will have 4 buttons. These buttons will be:

1. Logout
2. Leaderboard
3. Game
4. History

The logout button will log the user out by removing the state that you previously stored and will redirect the user to the login page.

The leaderboard button will redirect the user to the leaderboard page. Similarly, the game and history button will redirect the user to the game and history page respectively.

Game page [30 marks]:
On this page the user will be able to play the game, Wordle. It will contain a 6x5 empty board as shown below:



Firstly, there needs to be a 5 letter word that the user will be guessing. For that, you will be using the following api:
https://random-word-api.herokuapp.com/word?length=5

This api returns a word of length 5 which the user will be guessing. For every game, a new word must be used by calling this api. You can open this api on your browser to see what it returns.

After getting the word, the users will be able to make guesses. For that there will be a form below along with a button. This form will take in a word as input from the user. Upon submission, you need to check if the word is of the correct length (must always be of the length 5). If it is correct then place the word into the board, with all the correct colours according to the rules of the game. At this point, if the word entered contains no matching letter then this word must not be accepted. For example, if the word that the user is guessing is “FAVOR” and the user enters “TENTH” then this word should be discarded. Similarly, if the user enters “EARLY” then the board will be as following:

Then, if the user enters the following words “RAPID”, “MARKS” and then is able to guess the word at his 4th turn, then the board will be as the following:

If the user is able to guess the word within their 6 tries, then they win otherwise they lose. In both cases, as the game ends:

1. The form needs to be disabled so that the user is not able to enter anything else.
2. The board, result of the game (won or lost) and the word which the user had to guess needs to be stored in the database.
3. A button needs to be shown, which refreshes all the states on click and allows the user to play the game again.

History page [25 marks]:
The history page will show the game history of the logged in user. For each of the games played by the user, you need to show the 6x5 board of that game which was filled by the user during the game. You also need to show if the user won or not and what was the word which the user had to guess. You also need to display the username of the logged in user.

Leaderboard page [10 marks]:
The leaderboard page will show the rankings of all the users along with their usernames in the correct format (highest to lowest). These rankings will be based on the number of successful games for each of the users.

The leaderboard page, game page and history page will have a back button as well. Using this button the user can go back to their homepage.

It is up to you on how you want to design your database. Similarly, there will be no restrictions or deductions for frontend design as long as it is readable. For simplicity and your ease, you can simply use bootstrap v5. You are allowed to use any of the react libraries as well.