**Ruchami Bamberger & Naomi Fierstone Confident Drivers Quiz**

We are creating a practice theory online driving test. The user will have to login, and then have options to view previous scores or take a quiz. New users will have to sign up first. The quiz will be multiple choice questions on the rules of the road. As soon as an answer is submitted, they will see if it’s correct. The score will appear at the end.

The project has a login, welcome, questions and final scores page, as well as a page to view previously saved scores. A view of each page is shown on the slides. Below is a description of the elements on each page, frontend functions, and API calls and functions.

ALL PAGES:

There is a menu bar with a logo that takes user to the home page, and a logout button that navigates to the login page. (On the login page, it is slightly different – the logo has no functionality, and there is a sign-up button instead of a log-out button).

There is also a footer with website info.

LOGIN:

Two inputs for user to login with email and password. Button to login.

When user clicks login, the user\_email and user\_password are sent with an API PUT call to backend with url ‘api/validateuser/’. The function checks if this email exists in the users table. If it does, it then checks if the password matches. The user ID is returned if correct, ‘wrong password’ if email exists but doesn’t match password, and ‘please sign up’ if there is no such email in the database.

If login was successful, ID is saved to the session storage, and user is navigated to the welcome page. If password incorrect, an alert pops up. Otherwise, user is told to sign up. The login button changes to a sign-up button, which when clicked, the user\_email and user\_password are sent with an API POST call to backend with url ‘api/adduser/’. The details are used to save a new user in the database. User can then login using the new details.

There is also an option on the top right corner for the user to sign up.

WELCOME PAGE:

A description of the quiz is shown, with a video of a car driving. There are two buttons – ‘start quiz’ and ‘view scores’.

The start quiz button renders the questions component so user will see question 1. The view scores button renders the view scores component where user will see all previous scores.

When the welcome page is loaded, the questions are pulled from the backend with an API GET call with url ‘/api/getquestions/’ and saved as an array.

QUESTION PAGE:

This component receives the array of questions as a prop. The user sees one question, image, which number the question is, and a choice of four answers with a submit button.

There is a state for the question number. By default, it is set to one. The question with an id that matches the question number is displayed. Therefore, when the question page is first loaded, the first question is displayed.

When a user clicks an answer, it gets highlighted. Only one can be clicked so by clicking another, the first one ‘un-highlights’.

When a user clicks submit, the correct answer goes green, rest red. On the selected answer, if it is correct, there is a thumbs up by it, otherwise a thumbs down. There is also an alert saying correct or incorrect answer. If the user got the correct answer, one is added to the score variable. The submit button changes to a ‘next’ button. When clicked, 1 is added to the question number, and the next question is therefore loaded.

When question number is more than 11, the final score page is rendered.

FINAL SCORE PAGE :

This component receives the array of questions as a prop, as well as the score that the user received on the quiz. User sees a comment, score, video saying ‘success’ or ‘try again’, and three buttons – save my score, view my score, and take another test.

If user scored above 6, the success video is displayed. Otherwise, the try again video is displayed.

The take another test button renders the questions component so user will see question 1.

The view scores button renders the view scores component where user will see all previous scores.

If save score is clicked, the current\_score that the user received is sent to the backend, together with the current\_user\_id from the storage, and saved to the scores table on the database. This is done with POST with url 'api/savescore/'.

VEIW SCORES PAGE:

This component receives the array of questions as a prop. The user sees a list of all previous saved scores, as well as a button to take another quiz.

The user’s scores are retrieved from the database, by sending the user id from the storage as a parameter in a get call with url `/api/getscores/${id}`.

If the user has no saved scores, a message is displayed instead of the scores.

If the button is clicked, the questions component is rendered.

DATABASE:

Below is a list of our tables with the fields, data types and foreign keys:

QUIZ:

* question - Text
* answer\_1 - Text
* answer\_2 - Text
* answer\_3 - Text
* answer\_4 - Text
* correct - char
* image - char

USER\_DETAILS:

* email\_address - email
* password - char

USER\_SCORES:

* user\_id – foreign key from user\_details table
* score – integar

**Enjoy our quiz and looking forward to meeting you on the road!!!**