IT5007: Software Engineering on Application Architecture

Tutorial-3: Ticket To Ride Enhanced Version

Deadline: 25th Sep. 23:59 PM

Points: 20

Your efforts in designing the reservation system for the high-speed railway network has not gone unnoticed by the Singapore Government. However, the front desk feels that your reservation system could use some improvements, with which you stand a great chance of winning the best employee award. Your task is to improve the offline system to an online web application with the long-term goal of supporting better responsiveness. In the IT5007 module, you have learned that React is the best way to go about doing this. In this MVP stage, styling (CSS) is not a requirement although you could experiment with it if you like. To win this award, you must provide the following improvements:

(1) Code the variable for storing traveller data. This should be local to your application (i.e., do not use a database). You can do this using a JS/ES6/React variable. This step also involves choosing the right set of fields for storing user information (e.g., Name, phone number etc.). {Difficulty: Easy, Marks: 1 point}

(2) displayHomepage: Write a navigation bar with which you can navigate to each of the component. Note that, when the Nav button corresponding to addTraveller react component is clicked, the other components (e.g., displayTravellers) should be hidden from user view. Another sub-component of this component is the displayFreeSeats that displays the total number of empty seats. {Difficulty: Hard, Marks: 4 point}

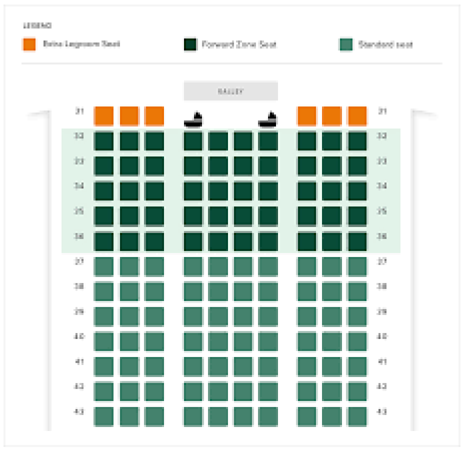
(3) displayTraveller: Code the logic for displaying the details of travellers in displayTraveller component. Fetch the details from the aforementioned traveller data variable, format and display them as a table. {Difficulty: Medium, Marks: 6 point}

(4) addTraveller: Code a form for adding a Traveller. On clicking the submit button, the internal data structure/JS variable in which you store the reservation list must be updated. {Difficulty: Medium, Marks: 4 point}

(5) deleteTraveller: Likewise, code a form for deleting a Traveller. On submit, the internal data structure must be updated. You can also choose to provide a delete button for each entry on the table. But this is not necessary. {Difficulty: Medium, Marks: 3 point}

(6) <Advanced>: Visual Representation of reserved/unreserved tickets. Display an illustration (similar to air/movie theatre seating arrangement as shown below) where the empty seats are highlighted in grey and the occupied seats are highlighted in grey. Note that the seats do not have a number and the figure is just illustrative as to what percentage of seats are free. {Difficulty: hard, Marks: 2 points}

(7) Testing: You need to add/remove Travellers and see if the table and free slots are updated correctly. Corner cases such as underflow and overflow must also be tested assuming that the train has 10 seats in total.



Submission details:

1) Submission is through Luminus. The folder will be available soon.
2) Use port 3000 for your express server; same as textbook examples. Do not change this. Tutors will be executing your code on a docker container with port 3000 open.
3) You will be given a code skeleton (git) to start working on your project. You can clone this repository and start working. You may also code from scratch if you like. [Will be available by Monday 12th September 23:59]
4) The skeleton code will have markers/pointers telling you where to write your code.
5) Submission Instructions:
    a. Will be made available ASAP.