Distributed Programming I
Web Programming Test Assignment

Build a simplified version of a website to manage seat reservations for an airplane. For simplicity, consider reservations for one plane and one journey only. Consider a cabin with seats arranged according to a rectangular layout of known size (6 seats in width x 10 places in length). These values must be easily adjustable by setting two variables in a single position in the PHP code. The seats in a row are indicated with a letter starting from A, the rows with a number starting from 1. The website must have the following characteristics:

1. On the homepage of the site anyone can view, even without any registration or authentication, the seats map, which shows, using differentiated colors, the purchased seats (red), the free ones (green) and those reserved for purchase by any user (orange). The page must also display the total number of seats, the total number of those purchased, those reserved, and the free ones.

2. The reservation and purchase of seats is possible only after registration and authentication on the site. Each user can freely register on the site by providing only a username, which must be a valid email address, and a password, which must contain at least one lower-case alphabetic character, and at least one other character that is either alphabetical uppercase or numeric. Otherwise the user must be warned before sending the password to the server, and in any case the registration must be prevented. If registration succeeds, the user results authenticated to the system.

3. An authenticated user sees the seats map on his personal page. In addition, the authenticated user can reserve seats. In order to do it, the user has to click on the seat to reserve, which can be a free seat (green) or even one reserved by others (orange). Without reloading the page, upon a click, the server must be queried to verify that the seat was not purchased in the meantime. If verification succeeds, the seat is displayed as reserved (in yellow) and this reservation is stored on the server. If the seat has been purchased in the meantime, it must be displayed in red, and obviously it must not be selectable by the user anymore. If the seat had been previously reserved (but not purchased) by another user, the old reservation by that user is removed, and a new one made by the last user is inserted instead. In any case, with a click on a seat, a message indicating what happened must be shown, and the seat color must be updated according with its status. A click on a reserved seat (in yellow) must free it, without reloading the page, but communicating it to the server.

4. The seats map is completely updated only if an "Update" button is pressed (which can be used by any authenticated user), or via the purchase operation. Seats reserved by another user, but not purchased, must be shown in orange.

5. After a user has reserved at least one seat, a "Buy" button should allow the user to send the server a request to purchase the (yellow) seats currently displayed on the map. All seats must be purchased in the way they are displayed. If this operation is not possible in its entirety, the user must be informed, and all reserved seats must be released, and they will be displayed on the map based on their new status. If the purchase is successful, all seats must be marked as purchased and displayed in red on the map. In any case, after a purchase request the page is reloaded, showing the new current status of all the seats as known by the server, and a message must inform the user about what happened, including the reason for a failure of the operation.

6. Example:

Initially all seats are free (green).

User u1@p.it clicks 3 times, to reserve seats A1, A2, B2, which become yellow.

User u2@p.it, having entered the site when the seats map showed them all still free, clicks to reserve seat B2. Seat B2 is now reserved for user u2@p.it and it appears in yellow. User u1@p.it is unaware of what happened.

User u1@p.it presses the purchase button, thus sending the purchase request to the server. The server recognizes that seat B2, requested for purchase by u1@p.it, is not reserved for this user, and therefore it refuses the request, freeing up all the seats reserved by u1@p.it. Now the only seat reserved is B2, by u2@p.it. User u1@p.it sees the map of seats with B2 in orange and all the others in green.

User u2@p.it reserves seats B3 and B4, and proceeds with their purchase request. The server confirms the operation and seats B2 B3 B4 become purchased (therefore red).

User u1@p.it clicks to reserve seat B4: without reloading the page, user u1@p.it is notified of the impossibility of reserving seat B4, since it has been purchased. The map displayed to user u1@p.it is all green, except for B4 in red.

User u1@p.it reserves seats A4, D4, F4, which then will appear all in yellow in the user's view.

User u2@p.it reserves seat F4, which then will appear in yellow in the user's view.

User u1@p.it presses the update button and sees a map in which seat F4 has turned orange.

Final state of the map displayed on the front page: B2 B3 B4 red, A4 D4 F4 orange.

7. In the delivered project there must already be two users *u1@p.it, u2@p.it,* with passwords *p1, p2,* respectively, who are in the situation at the end of the previous example.

8. Authentication (through username and password) remains valid until the user has periods of inactivity exceeding 2 minutes. If a user attempts to perform an operation that requires registration or authentication after the inactivity has exceeded 2 minutes, the operation has no effect and the user is forced to re-authenticate with a username and password. The use of the HTTPS protocol for registration and authentication must be enforced, as well as in every part of the site that shows information relating to an authenticated user.

9. The general layout of the web pages must contain: a header at the top, a navigation bar on the left side with links or buttons to perform the various possible operations, and a central part which is used for the main operation.

10. Cookies and Javascript must be enabled, otherwise the website must not work (in that case, the user has to be notified in the main page, and the application web pages must not be displayed). Forms should be provided with small informational messages in order to explain the meaning of the different fields. These messages may be put within the fields themselves or may appear when the mouse pointer is over them.

11. The more uniform the views and the layouts are by varying the adopted browser, the better.