HW 3: Basic HTML/JavaScript Practice (100pt+)

Due Date: Nov 10, 2024, 11:59 PM

Objective

The goal of this assignment is to provide a straightforward practice with HTML and JavaScript by implementing a basic ticketing service interface. This is part of a exercise to support COSE432 - fall 2024 final project: AI-assisted ticketing service. While this assignment does not need to directly relate to your final project, you should keep in mind that features like these may need to be implemented in it (you can extend or modify it later). Advanced HCI design elements, such as usability or aesthetic design, **are not required** for this task. Additionally, **no** server or database implementation is necessary, although you may use one if you wish.

What to Implement

Initial Page Setup (20pt)

1. When index.html or ticket.html is first loaded, it should display an entry page with a basic menu. This page can include options or links to previous reservations.

2. You are encouraged to add more functionality if you feel it enhances the project.

Data Handling with Local CSV File (40pt)

1. Assume that reservation data is saved in a structured format, such as a .csv file, within your local directory. This assignment does not require you to use a database or server.

2. When index.html loads, it should read the .csv file and display relevant information or links for any existing reservations, as shown in **Figure 1.'s** Entry Page.

3. You will be given example code to help with reading .csv files using JavaScript. (refer to the Example.zip file provided with the assignment).

Creating and Displaying Reservation Details (40pt)

1. When creating a reservation, you will generate a new object with attributes such as date, time, and destination, similar to the format shown.

```
Reservation object 124 {
 Date: 2003 Feb 3
 Time: 13:00
 Train no: KTX 001
  ...
}
```

.csv file

```
...
2003, Feb, 3, 13:00, "KTX 001", ...
2024, Oct, 4, 17:00, "KTX 011", ...
...
```

2. The .csv file will store this information as a list of comma-separated values.

Example: "Create" Functionality

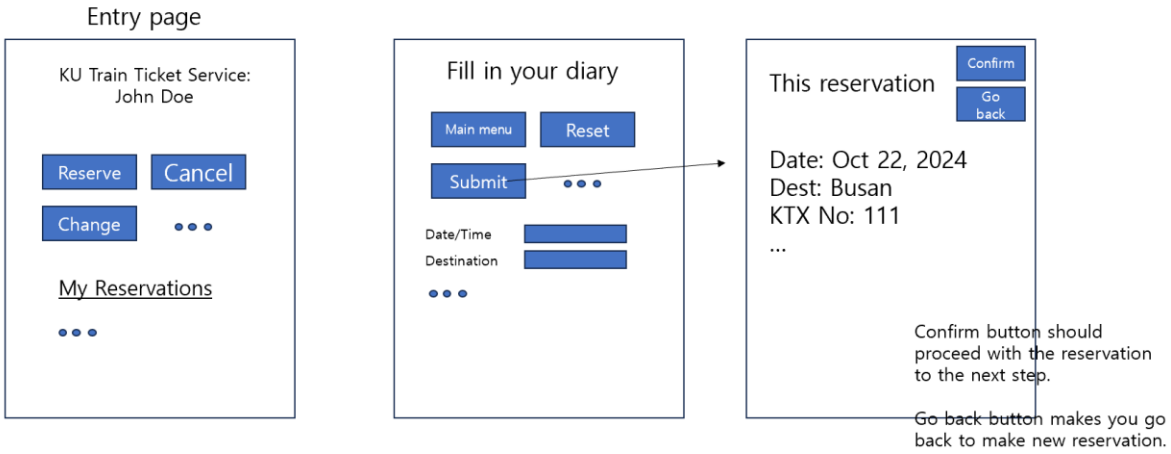

FIGURE 1.

- When the user selects the "Reserve" option, the diary system should display a form where they can enter reservation details (e.g., date, time, and destination). The format is up to you to design.
- After the user submits the form, display the reservation summary in a clear format.
- Additional functionality, such as "Go to main menu" or "Reset," can be added as needed.

Additional Features (Optional)

- **Cancel/Change Reservation**: Allow users to select a reservation to cancel or change and update the .csv file accordingly.
- **Other Enhancements**:
    - Additional attributes for the reservation, like style preferences (e.g., background color, font).
    - Adaptation for mobile view.
    - Simple security measures, such as hiding specific information.

NOTE. This assignment rewards good grades for fulfilling the basic requirements. However, additional clear and detailed implementations of optional features may earn extra points. If you wish to provide further explanation of your implementation, consider including a README.md file with your submission.

Submission Guidelines

Submit a **zip** file named as follows: **HW3-JohnDoe-2012345678.zip**.

NOTE. The submitted code must work as-is without additional setup. Ensure your code works correctly in the environment you used for testing. If there are specific requirements or configurations (e.g., the browser or version used), please include them in the README.md file.

The project directory structure can vary; however, to meet the minimum requirements, it could look like the following example:

```
HW3-YourName-YourStudentID.zip
│
├── index.html                  # Entry page with the menu
├── ticket.html                 # Page to display and manage reservations
├── reservations.csv            # CSV file storing reservation data
├── script.js                   # Any additional JavaScript code you implemented
└── README.md                   # Optional: explanations or notes about your submission
```

Minimum information to include in README.md (if provided):

- The browser and version used for testing.
- Any additional setup instructions or known issues.
- Optional explanations of implemented features or additional functions.