Assignment 3

Submission Checklist

For your submission to be graded, you must provide a **zip** file of your project, and a **screen recording** demonstrating the functionality you implemented.

1. Creating Your Project

- When creating your project, name the project: **Rentals-firstname,** example: **Rentals-Bobby**
- Ensure your project is configured as a NodeJS project. You can do this by running "**npm init -y**" in your project folder
- After completing your project, create a zip file containing all project code.
- Name the project **Rentals-firstname.zip,** example: **Rentals-Bobby.zip**.
    - Do NOT submit 7zip or rar files

2. Creating Your Screen Recording

- In the screen recording, demonstrate the app functionalities you implemented
- During the screen recording, you must verbally narrate/explain what you are doing while you are doing it (do not assume the instructor will understand what you are doing simply by watching you click things on the screen).
- You are not required to explain your code
- Max 5 min

3. In the assignment dropbox:

- Submit your zip file and screen recording to the dropbox
- If your screen recording is too large for the dropbox, then upload your screen recording to **Microsoft OneDrive** and ensure that the link is set to: "Anyone with the link can view". Paste a link to the recording in the **submission comments**.

Academic Integrity

- This is an individual assessment.
- Permitted activities: Usage of Internet to search for syntax only; usage of course materials. If you are using the Internet, then ensure the resources align with the materials presented in class.
- Not permitted:
    - Communication with others (both inside and outside the class)
    - Discussion of solution or approaches with others; sharing/using a "reference" from someone
    - Searching the internet for full or partial solutions
    - Using homework help sites with Chegg, Studoc, or CourseHero
    - Sharing of resources, including links, computers, accounts

Problem Description

Using Node, Express, and Handlebars, create a server-side application that enables a user to rent items from a store.

The webpage consists of three pages:

1. **Home Page**: Displays the home page for the electronics store
2. **Rental Catalogue Page**: Displays a list of items available for rent
3. **Error page**: Displays error messages.

HTML and CSS Styling

1. The website's pages must be implemented using Handlebars templates.

2. You must make appropriate usage of Handlebars **templates, layouts,** and **partials**

3. Use the screenshots as guidance on the webpage layout.

4. You should customize colors, typography, and white spacing to match your own design aesthetic. Use reasonably pretty colors and non-default fonts (example: no Times New Roman)

Data Modeling

1. The items for rent must be modelled as an **array of Javascript object literals**.

2. Each item has properties for:
   - Item id. The id must consist of a 4 letter string. You may choose any value for the ids, but each id must be unique.
   - The name of the item
   - The minimum rental period (the minimal number of days that the item must be rented)
   - Is the item available? If yes, then it can be borrowed. If no, then you must have already rented the item.

3. The application must contain a minimum of 6 items. You must use your own meaningful values for these items - do not copy the values shown in the screenshots!

Website Pages

Page Layout

Every page of the website contains two sections.

1. Header section: Displays the store name and search box.

   a. Store Name:
      - **Use your first name** as the store name (example: Bobby's Store)
      - Clicking on the store name will navigate the user to the **Home Page**

   b. Search Box:
      - User enters keyword in box. When the search button is pressed, search for **items** that contain the specified keyword in the item name.
      - If results are found, navigate the user to the Rental Catalog page.
        - On the Rental Catalog Page, display the search results.
      - If no results are found, navigate the user to the Error page.
        - You must display an appropriate error message.

2. Main section: Displays the primary content for the page

Home Page

The main content of the Home Page shows:

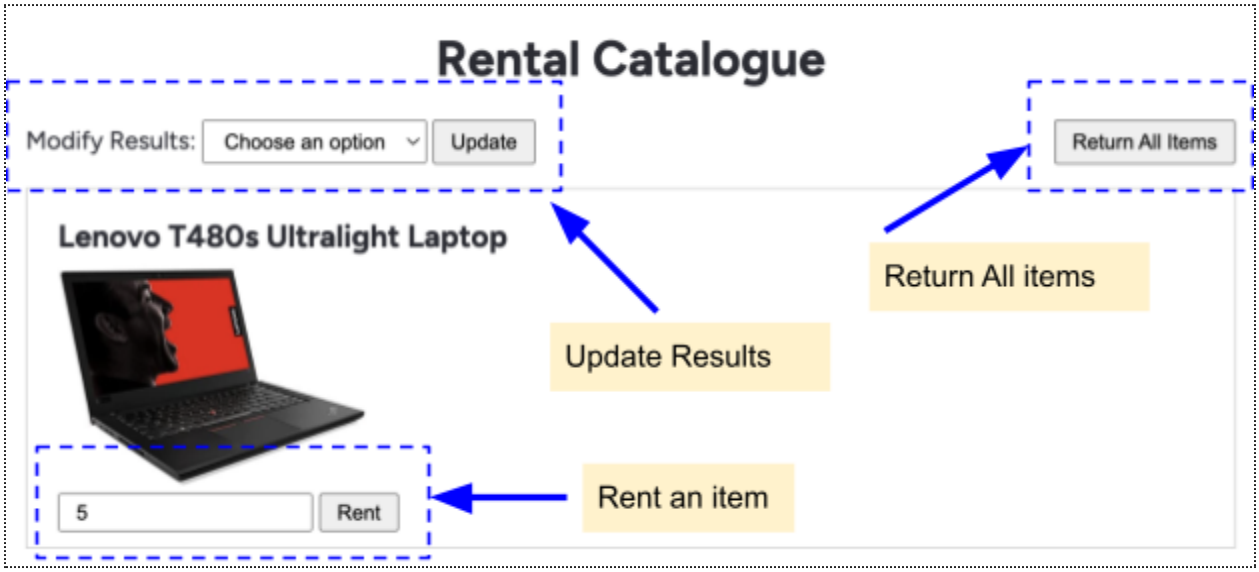
1. Information about store
2. Example of items that can be rented
   - Display three **random** items from the items list
   - The items must be programmatically selected
   - It is okay if your random items contain duplicates.
3. Link to the Rental Catalogue page
   - Clicking on the link navigates user to Rental Catalogue Page

Rental Catalog Page

1. By default, this page displays a list of *all rental items*.

2. For each item, you must display
- Name of item
- If the item is available, then display a form for renting the item
- If the item is not available, then display an appropriate message.

3. The page also contains forms to enable the user to perform operations on the list of items. Each form must communicate with an individual server endpoint.
- **Rent an Item Form**: Used to rent one item in the store. Create one form per item.
- **Return all Items Form**: Used to return the user's rentals back to the store
- **Update results Form**: Used to update the displayed results



2. **Rent an item Form**: To rent an item, the user must enter the length of time they want to borrow the item for.
- By default, populate the field's **hint text** and **value** to the item's minimum loan period.
- The user can update the displayed form field value. If the user attempts to rent a item for *less* than the minimum loan period, display appropriate error message on the Error Page
- Otherwise, update the status of the item to indicate that it is rented by the current website user (you)

3. **Update Results Form:** The list of displayed items can be updated in one of two ways
    a. **Available Items**: Reloads the page to show *only* items that are available for rent
    b. **Your rentals:** Reloads the page to show items rented by the current user (you)**.** If the user does not have any items, display appropriate message in the Error Page
5. **Return All Items**
    - Clicking on the button will return the user's items to the store.
    - If the user does not have any items, display appropriate message in the Error Page

Error Page

If the user performs an operation that results in an error, an error page should be displayed.

The error message must be programmatically set by the server.

Pressing the GO BACK button returns the user to the **Rental Catalogue** page.

END OF ASSESSMENT