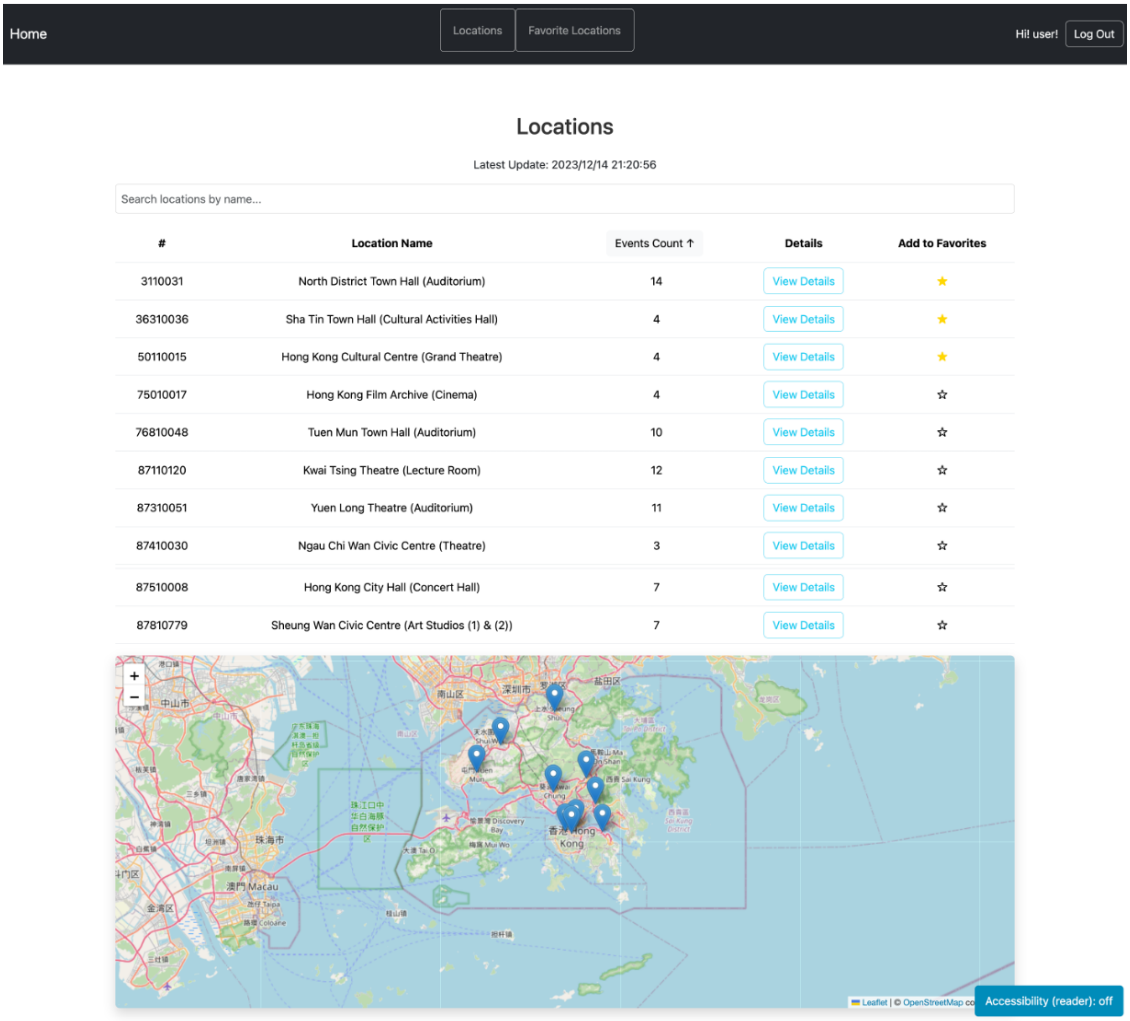
CSCI2720 Group 17 Project Report

PAN Yuchen 1155173729, LI Heming 115515726,
YUAN Lin 1155141399, XU Xinyi 1155173854, KONG Deyuan 1155141454

1. Abstract:

This project is a JavaScript-based web application built with React and supported by Node.js and MongoDB. It offers a single-page application design, allowing users to log in as regular users or admins. Users can browse locations, view details, add favorites, post comments, and manage their favorite locations. Admins have additional capabilities to manage events and users. The application incorporates accessibility features with speech synthesis which could be controled by button or keyboard shortcuts. It utilizes libraries like React Leaflet for map integration. Although there are limitations, such as handling data relationships in MongoDB and lack of session persistence, the project demonstrates a feature-rich web application emphasizing user experience and accessibility.



2. Files Submitted:

1) README:
    a) *README.md*: An instruction file teaching you how to set up and use our websites.
2) HTML:
    a) *index.html*: A HTML file to be further rendered.
3) Route Control Components:
    a) *index.js*: Handle the navigation bar and basic route control for all pages.
4) User Page Components:
    a) *UserPage-Favorites.jsx*: Display user's favorite locations..
    b) *UserPage-Location.jsx*: Display the location table and the map with their functions.
    c) *UserPage-Specific-Location.jsx*: Display one specific location with its events and comments.
5) Admin Page Components:
    a) *HomePage.jsx*: The main landing page of the admin page.
    b) *AdminPage.jsx*: Display the event list and the user list with their corresponding functions.
    c) *CreateUserForm.jsx* & *UpdateUserForm.jsx*: Create and update new users. *CreateEventForm.jsx* & *UpdateEventForm.jsx*: Create and update new events.
6) Server Component:
    a) *server.js*: Provide backend service for the frontend pages.
7) Authentication Component:
    a) *AuthContext.jsx*: A React context to manage authentication states and logic.
    b) *PrivateRoute.jsx*: Redirect unauthenticated users from protected pages to the login page.
8) Accessibility Component:
    a) *Accessibility.jsx*: An additional accessibility feature which is available on “location”, “location details” and “favorite locations” pages.
9) Styling:
    a) Stylesheets including *index.css, CreateEventForm.css, UpdateEventForm.css*, and *AdminPage.css*.
10) JSON files:
    a) Four json files including *comment.json, events.json, user.json,* and *venues.json*, which are the stored data.
    b) *package-lock.json* and *package.json*: The dependency resolution and version locking file and the configuration file.

3. Data Pre-processing

We used the recommended data and selected the locations with the following venueIds: "36310036", "50110015", "75010017", "76810048", "87510008", "3110031", "87110120", "87810779", "87410030", "87310051" as well as the events belonging to these locations.

1) Some events have multiple prices (eg. event id=155709 has price="HK$200, 100, 80, 60"); in our app, we kept the lowest price as its price (eg. for event id=155709, price=60). We also removed the non-number character (eg. "$", "HK").
2) Some events have price="Free Admission", "admission by enrolment" or other strings expressing similar meanings. We converted it as price=0 in the database.
3) Some events at these locations include only Chinese information and some even input Chinese characters in their title-e field on the website. We will exclude these events but will guarantee after exclusion there are still >=3 events at a certain location.

4. Application Functionalities

Login:

1) Our application can be logged in as either a user or the admin. If the username and password match an existing pair in the database, users or admin can continue with their actions. Otherwise, a login failed message will be displayed without logging in. After logging in, the user name will be displayed on the top-right corner as “Hi! Username!”.

User Page:

1) Location: All the locations are listed in a table with the following information: location ID, location name, events count of the location, “View Details” button leading to Location Details, and “Add to Favorites” star icon. Below the table, all the locations are shown on a map. Users can search for specific locations by name using the input bar located above the table. The location pins on the map will be updated showing only the filtered locations. By clicking the “Events Count” button, the locations can be sorted by the number of events in ascending or descending order. Filled star icons indicate a location is already in a specific user's favorite location, based on the existing record in the database, while outlined stars suggest otherwise. By clicking the icon, users can add or remove locations from their favorites list, which is displayed on the “Favorite Locations” page and stored in the database.
2) Location Details: The location name and location ID are listed at the top of the page, followed by a map to show the specific location. All the events of the location are also listed in a table, along with the event title, date, and price. By entering a number in the price filter bar above the table, only events with the price under the specified value will be shown. Below the event table, there is a section for

making comments. It will display existing comments made by all users under this specific location. By typing in the text area and clicking the “Post Comment” button, the comment and the user name will be displayed on the screen and stored in the database with venueId.

3) Favorite Locations: Every user’s favorite locations are retrieved from the database in real time and shown on this page. All the updates of the favorite list will also be preserved. The Location Details page is accessible here as well.
4) Accessibility: We enhance the application’s accessibility by implementing a speech synthesis feature using the Web Speech API for reading content out. *Please turn on your audio when testing this feature.* Available on the location/favorite/detail page, this component first identifies and tells the user which page it is currently (eg. “You are on the location detail page”) and then speaks out the displayed text content such as location/event/comment information on the page. It also responds immediately to re-rendering. Users can toggle the speech functionality on and off by: 1) clicking the button, or 2) keyboard shortcuts provided (shift + a to activate and shift + q to deactivate). The status (on/off) of this function is displayed as the text content of the button with the default status being off.

Admin Page:

1) Event Database: Admin can access a form page for creating new events by clicking the “Create New Event” button. After entering relevant information, new events can be created by clicking the “Create Event” button. All the events in the database and their detailed information are listed below, with an “Update” button to update the information and a “Delete” button to delete the event.
2) User Database: Same as the Event Database page, the application allows creating new users with custom names and passwords. The updated form can be accessed by clicking the “Create New User” button. Each user record can also be updated or deleted using the respective buttons below the specific information.

Log out:

1) After logging in, the user can log out by clicking the top-right “Log Out” button and the login page will be displayed again.

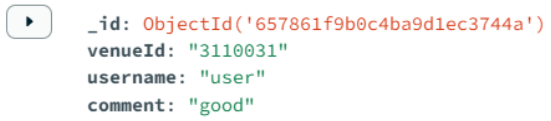
5. Programming Languages

The major programming language we use is JavaScript with React framework and the following packages: Express, Path, Cors, Mongoose. The backend is supported by Node.js and the database is supported by MongoDB.

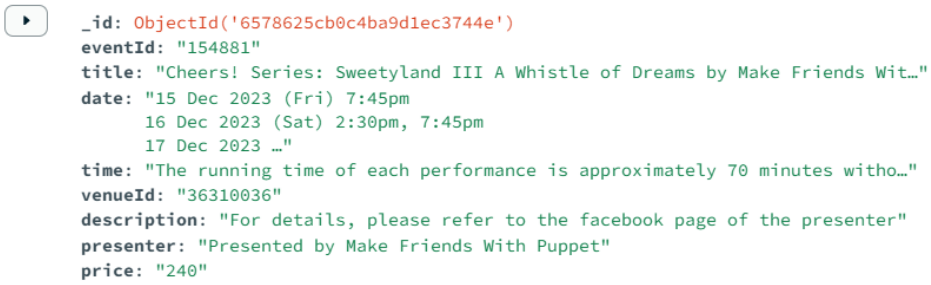
6. Database Schemas Design

Our dataset has four schemas: comments, events, users and venues.

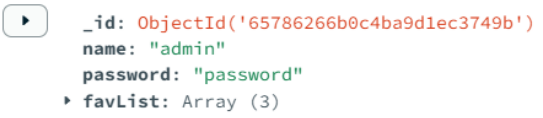
1) Comments schema constitutions: venuaId, username, comment.



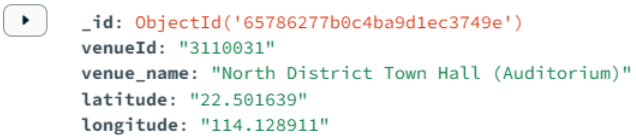
2) Events schema constitutions: eventId, title, date, time, venueId, description, presenter, price



3) Users schema constitutions: name, password, favList



4) Venues schema constitutions: venueId, venue_name, latitude, longitude



7. Use of Libraries and Frameworks

react-leaflet: A React library that integrates Leaflet maps, offering components for map interaction and display like markers, pop ups, and layers in a declarative and React-friendly way.

createContext & useContext: *createContext* initializes a new context for data sharing. *useContext* enables access to context data in functional components without prop drilling, simplifying state management across the component hierarchy.

Web Speech API: *SpeechSynthesisUtterance* is used for initiating speech requests. The language is set to be en-US. Text contents in the displayed tables/elements are concatenated with pauses and passed to the API as strings.

8. Comparison Table of Chosen Platform/Technologies Advantages and Disadvantages

Advantages	Disadvantages
We use single-page applications (SPA) to enhance user experience and reduce server loads while simplifying development. SPA provides a **seamless user experience** akin to desktop applications, with instantaneous content updates and no page reloads, unlike traditional multi-page websites that incur loading delays each time a new page is fetched, creating a more disjointed and slower user experience.	We use the non-relational database MongoDB. As the comment and user information are stored in different datasets in MongoDB, when changing the username / deleting the user, the corresponding comments will still be displayed with the original user name. However, in relational databases like SQL, it will be easier to **handle foreign key constraints** automatically.
We use Leaflet to show our locations instead of the suggested APIs: Google Maps or MapBox. Leaflet is a lightweight JavaScript library for embedding interactive maps, renowned for its simplicity, extensibility, and mobile responsiveness. At approximately 39 KB gzipped, it ensures **quick loading times and optimal performance**. Unlike proprietary counterparts like Google Maps and MapBox, Leaflet is free and offers greater flexibility, enabling customizations and avoiding vendor lock-in with the ability to switch tile providers.	We **do not use cookies for session management**. So when refreshing the page, the user will be logged out. This happens because, without cookies, the session state - which is not inherently preserved across HTTP requests - is lost when the browser reloads a page. To improve the user experience, enabling a session in Express will be very helpful.

9. Appendix

Workload Distribution:

PAN Yuchen: Admin Page, User Page, and Login Page programming and database CRUD.

LI Heming: Server-side construction, Back-end and Front-end interaction in User Page and environment configuration.

YUAN Lin: Location Details page, Accessibility Component, and Data Pre-processing.

XU Xinyi: Favorite Location Page, Location Details Page, and Events and Users database update.

KONG Deyuan: Location Page, Location Detailed page, Several Functions in server.js.