The Github link

The github link for this coursework is https://github.com/ChanKhaiShen/SoccerDB.git.

The API

The SportDB is a third-party web API that provides metadata of sports leagues worldwide and information about sports events and teams. The limit of the API is 100 requests per minute. The API allows free user to use '3' as the API key for testing or education purposes. The free tier does not include live result and player list. The documentation of the API can be accessed from https://www.thesportsdb.com/api.php.

In this coursework, only the API call for searching event by name and season (optional) is used. The base query string for this API call is https://www.thesportsdb.com/api/v1/json/3/searchevents.php. There are two parameters for this feature, "e" and "s". Parameter "e" is for event name, whereas parameter "s" is for season of the league. Parameter "e" is mandatory, whereas parameter "s" is optional. There are two situations for parameter "e". When "e" is "Team A vs Team B", the API will return events which Team A is the home team and Team B is the away team, whereas when "e" is "Team A", the API will return events which Team A is the home team. One limitation is when searching "Team A vs Team B", only the events which is Team A vs Team B and Team A is the home team are returned, but not the events which is Team A vs Team B and Team B is the home team.

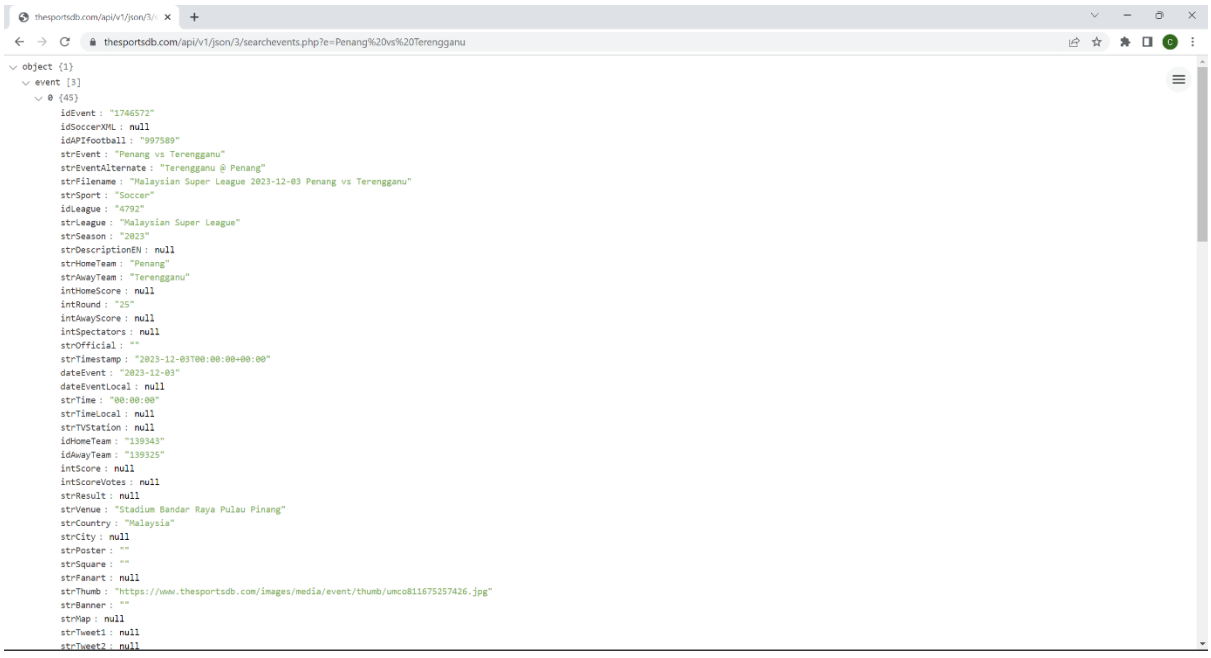

Figure 1 Sample result of API call for searching event by name

Figure 1. are the sample fields returned from the API call for searching event by name. The API call for searching event by name will return a list of events with multiple fields, except when there is no event found, it will return "null". The fields used in this coursework are "strEvent" (the event name), "strSport", (the type of sport, like soccer and tennis), "strSeason", "strLeague", "intRound", "date Event", "strVideo" (the video link), "strHomeTeam", "strAwayTeam", "intHomeScore" and "intAwayScore". In this coursework, the type of sport is limited to "soccer" because for different sports, the results are recorded in different fields. For example, for soccer, the results are recorded in "intHomeScore" and "intAwayScore", whereas for tennis, the results are recorded in "strResult".

The database

The database used in this project is MongoDB. The database is connected to Visual Studio Code using connection string. The network connection for accessing the database is configured as 0.0.0.0 because it allows access from any IP address. An admin user that has read and write privileges named "cks" is used for accessing the database of this coursework. The database is managed from MongoDB Atlas.

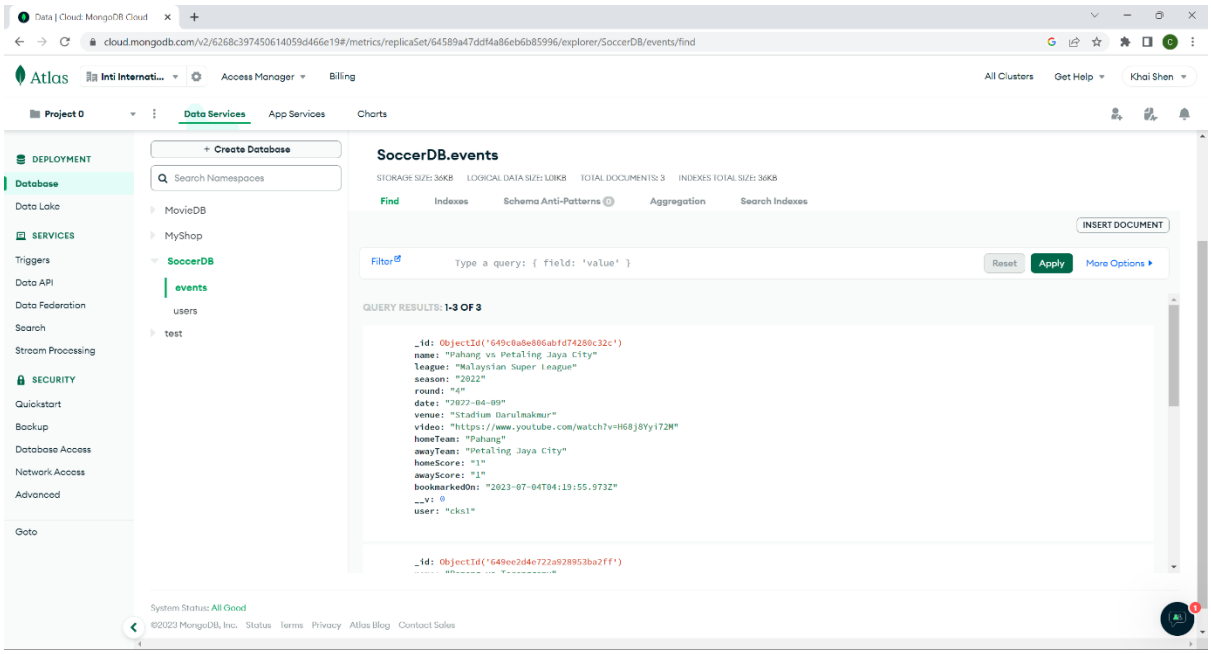

Figure 2 Collection "events"

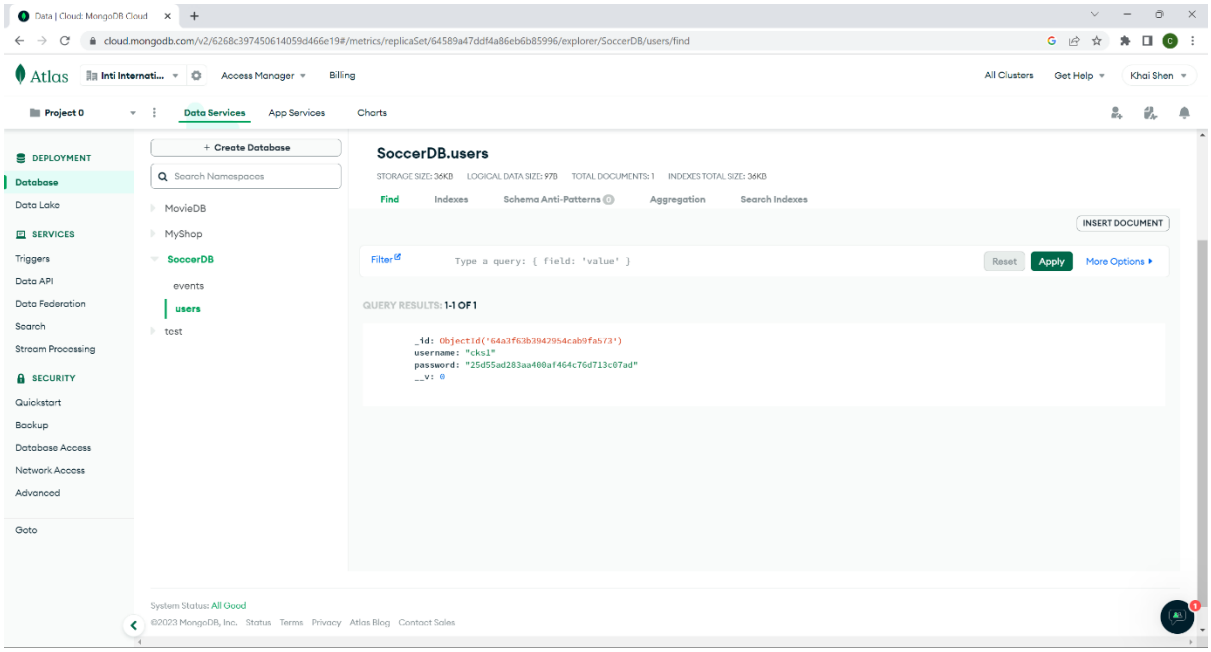

Figure 3 Collection "users"

A database named "SoccerDB" is used for this coursework. The database has twos collections, "events" and “users”. Figure 2. is the collection “events” viewed from MongoDB Atlas, and Figure 3. is collection “users”. The fields in the schema of collection "events" are "name" (the event name), "date" (the event date), "league", "season", "round", "venue", "video" (the video link), "homeTeam" "homeScore", "awayTeam", "awayscore", "bookmarkedOn" (datetime when the event is saved) and “user” (the “username” in collection “users”). The datatype of all fields in collection “events” is “String”. The fields in collection “users” are “username” and “password”. The datatype of both fields in collection “users” is “String”.

Packages

Packages used in this coursework are "express", "axios" and "cors", “path”, “md5”, “jsonwebtoken” and “cookie-parser”. The server in this coursework is built from "express". “express.json()” is used at the server to read the parameters sent in the request body through post method from the client. The server uses "axios" to execute http get operation to get data from the API, whereas the client uses "axios" to execute http get, post and delete operation send, delete and get data from the server. "cors" is used for allowing sending data across the client and the server hosted on the same machine. “path” is used together with res.sendFile() to specify the path of the html file (for web UI). “md5” is used to hash password at the client before it is sent to the server through http. “jsonwebtoken” is used at the server for generating a token to identify a user after login. The token is stored at the

cookie. “cookie-parser” is used at the server to read the token. The token needs to be verified before a user is allowed for adding, updating, getting and deleting bookmark. “eslint” is used in this coursework to ensure code quality. Figure 14. is the eslint file.

Code quality

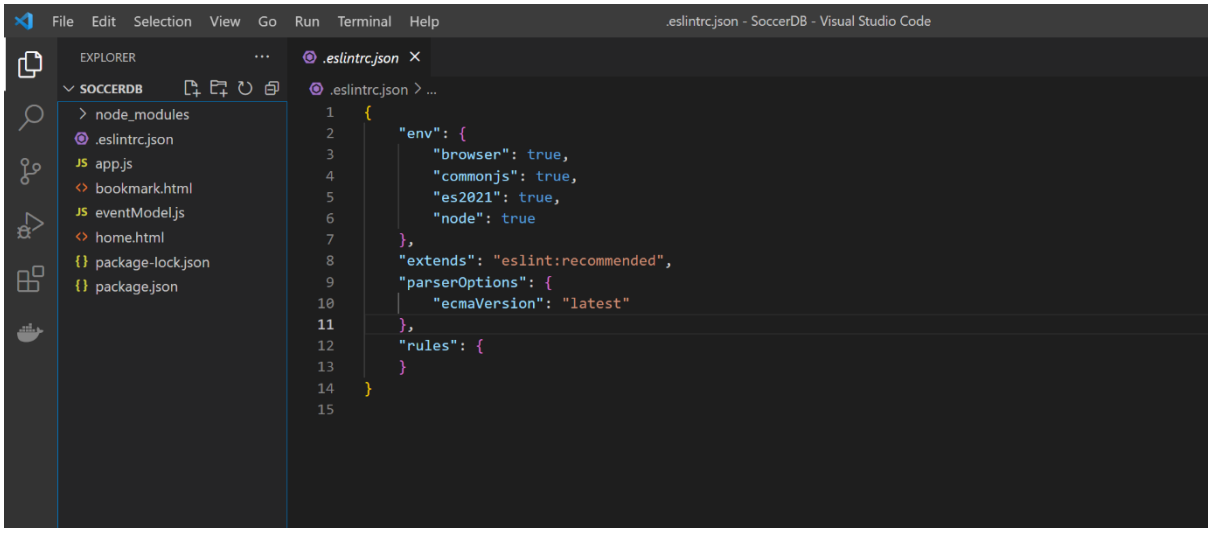

Figure 4 eslint file

“eslint” is used in this coursework to ensure code quality. Figure 4. is the eslint file.

The UI

Figure 5 Login page

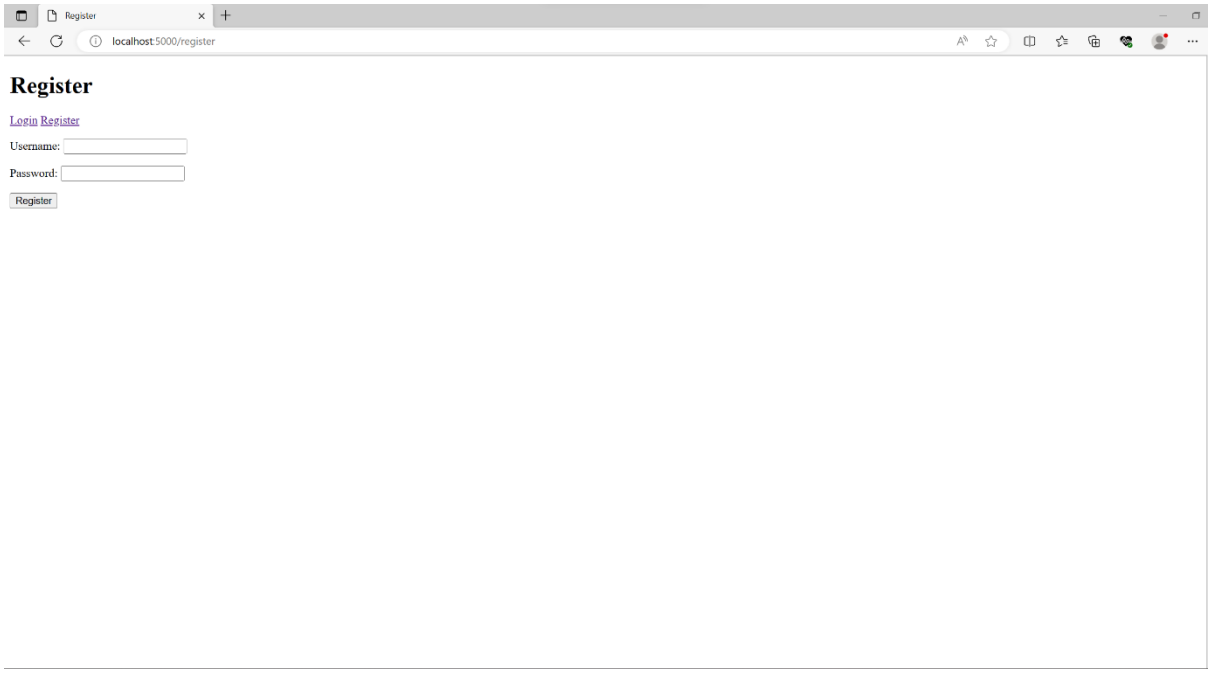

Figure 6 Register page

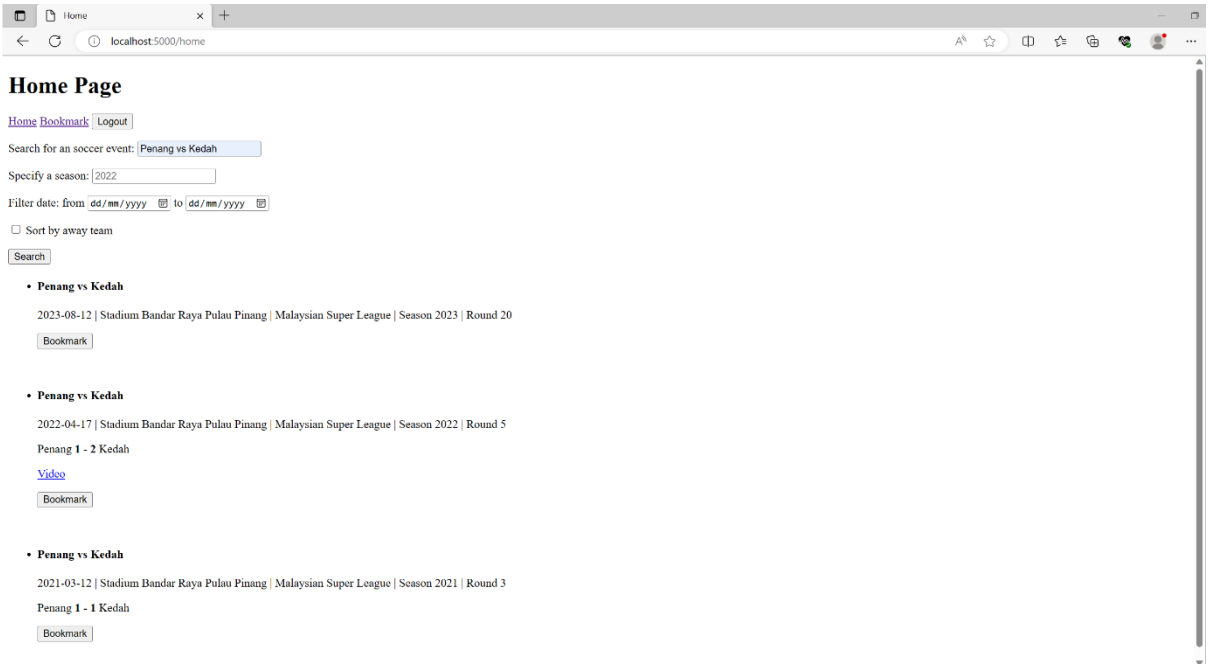

Figure 7 Home page (Search event 1)

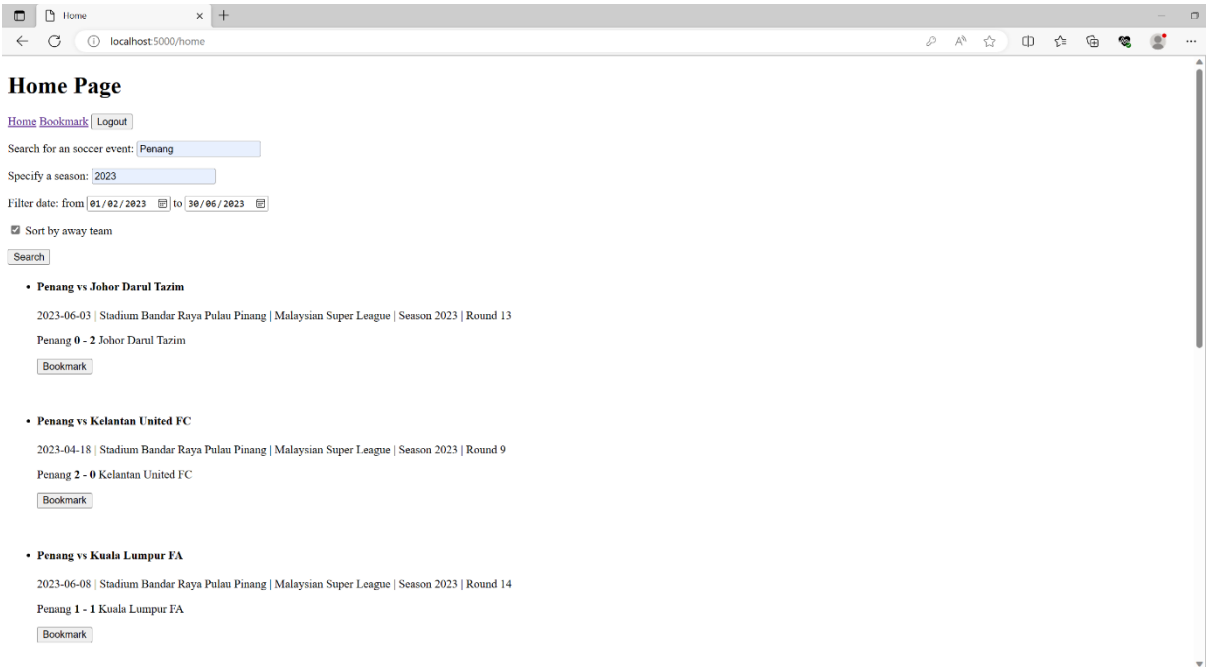

Figure 8 Home page (Search event 2)

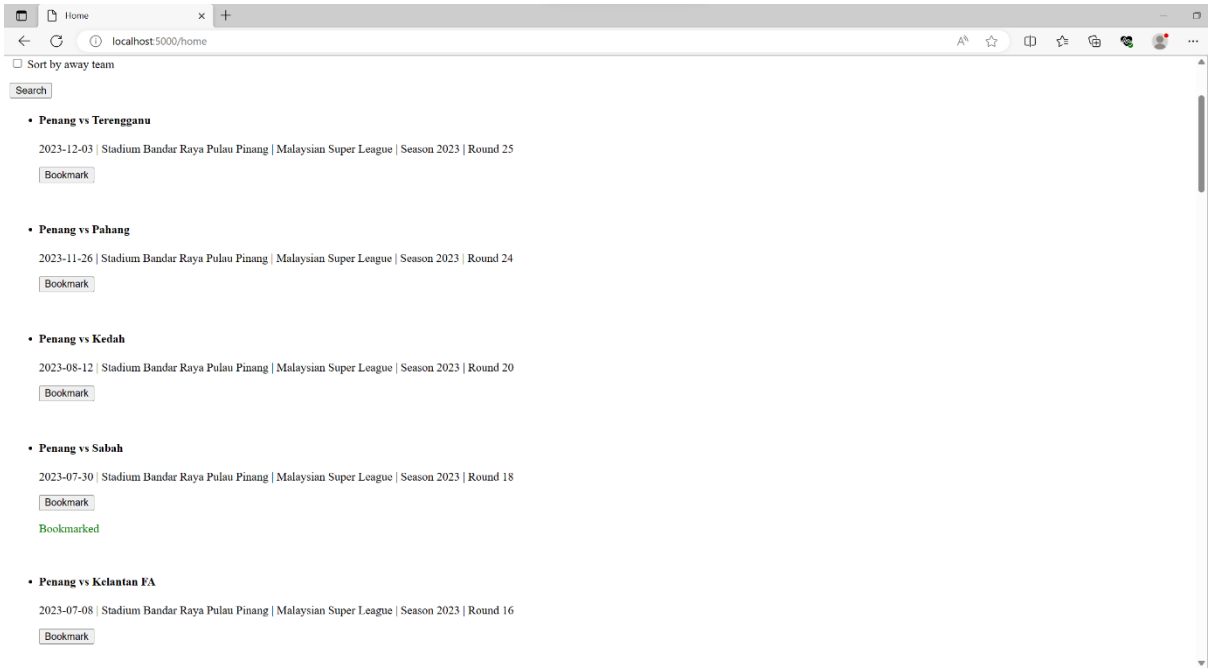

Figure 9 Home page (Bookmark event 1)

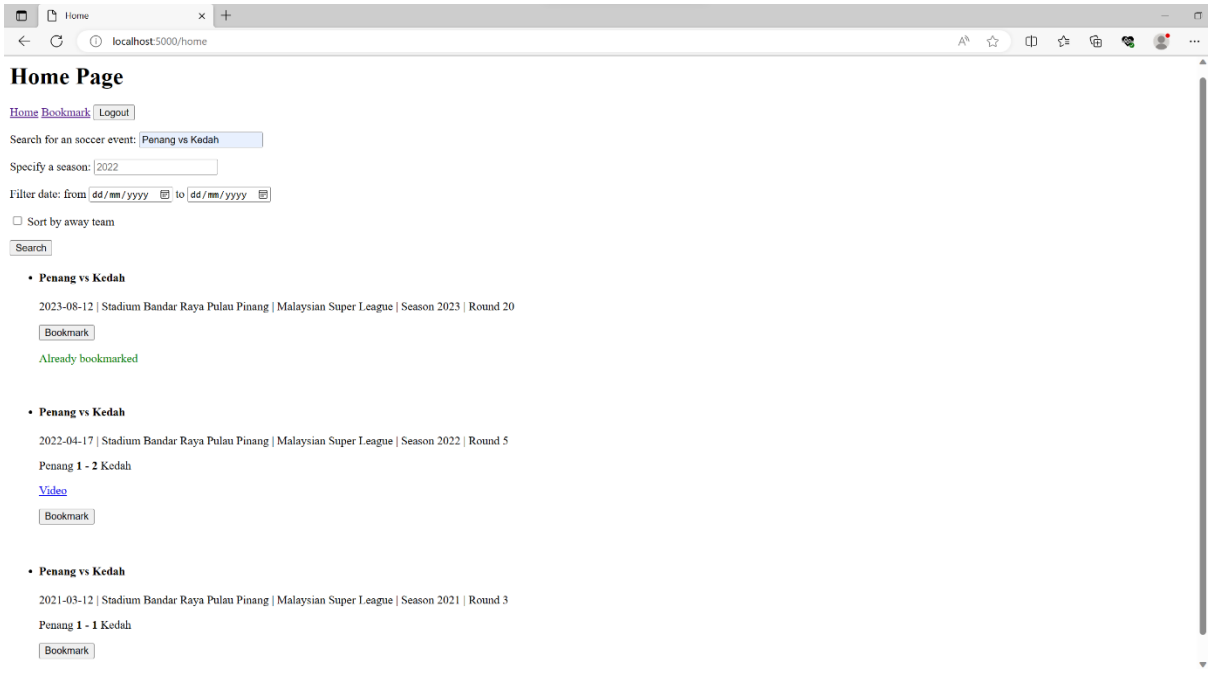

Figure 10 Home page (Bookmark event 2)

Figure 5. is the login page and Figure 6. is the register page. Excluding login and register, the system has two pages, “Home” and “Bookmark”. The features in “Home” page are search events and bookmark event. The search events feature allow user to specify the event (in the form of “Team A vs Team B” or “Team A” where Team A is the home team and Team B is the away team), specify the season (optional), filter the range of date (optional) and enforce sorting the events by away team name (optional). Figure 7. demonstrates the search feature with event name in the form of “Team A vs Team B”, whereas Figure 8.

demonstrates the search feature with event name in the form of “Team A”, season and date range specified and the sorting feature toggled. The bookmark event feature allow user to bookmark an event for future use. Figure 9. demonstrates the bookmark event feature when the bookmark operation is successful, whereas Figure 10. demonstrates the bookmark event feature when the requested event is already in the bookmark list.

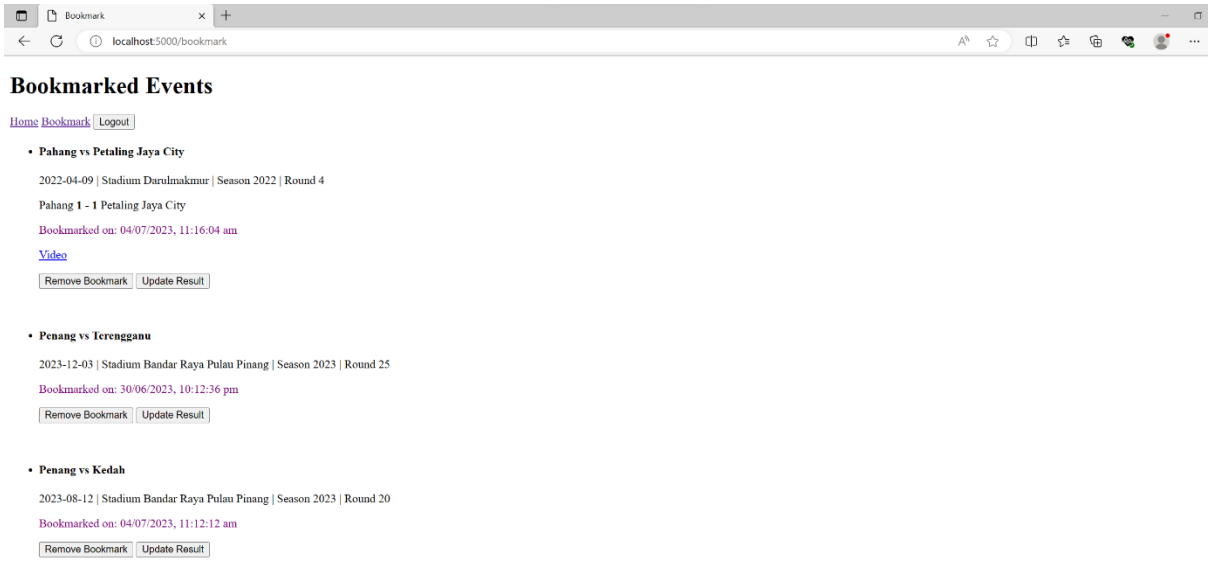

Figure 11 Bookmark page

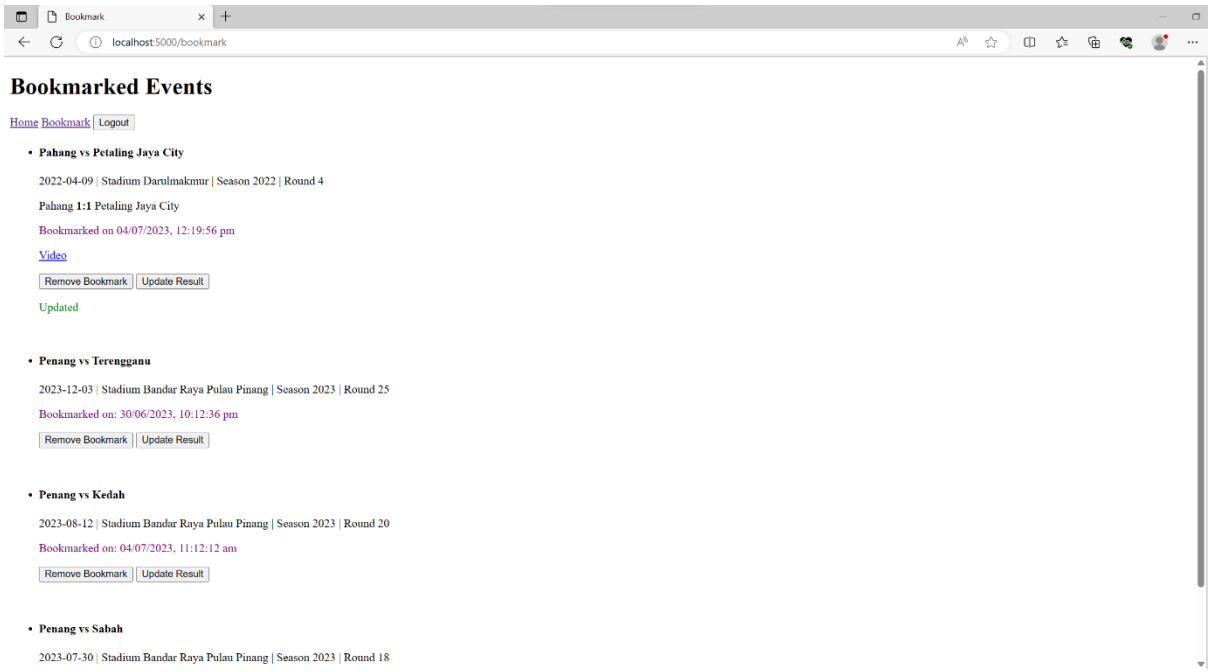

Figure 12 Bookmark page (Update bookmark 1)

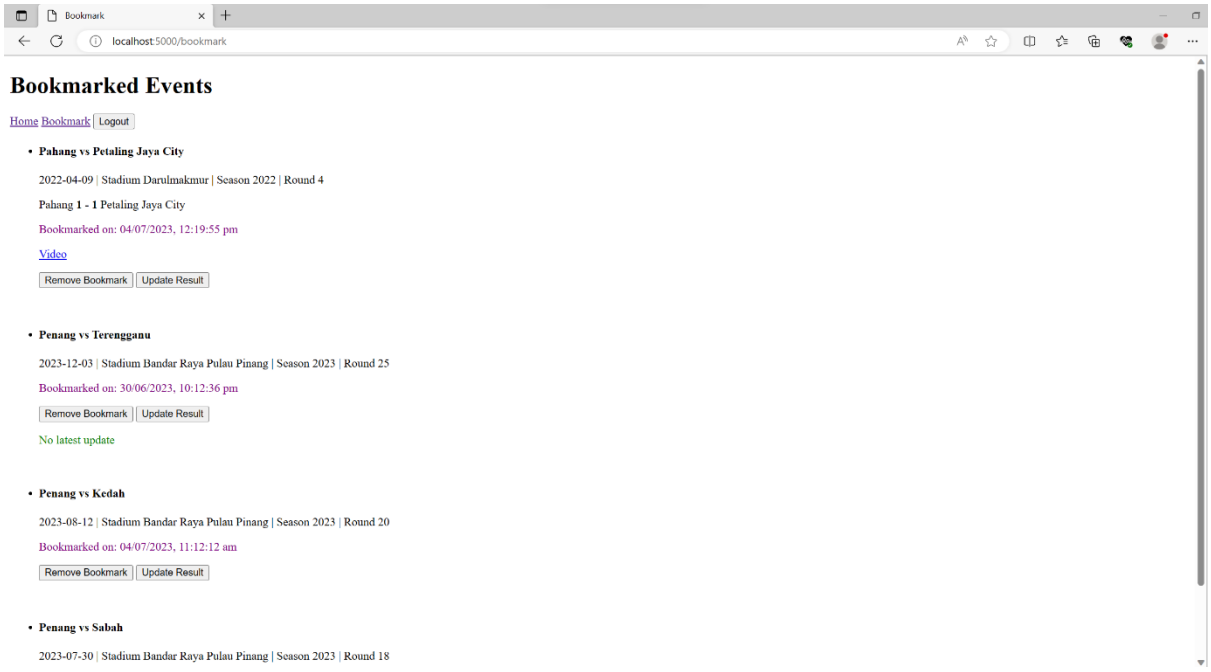

Figure 13 Bookmark page (Update bookmark 2)

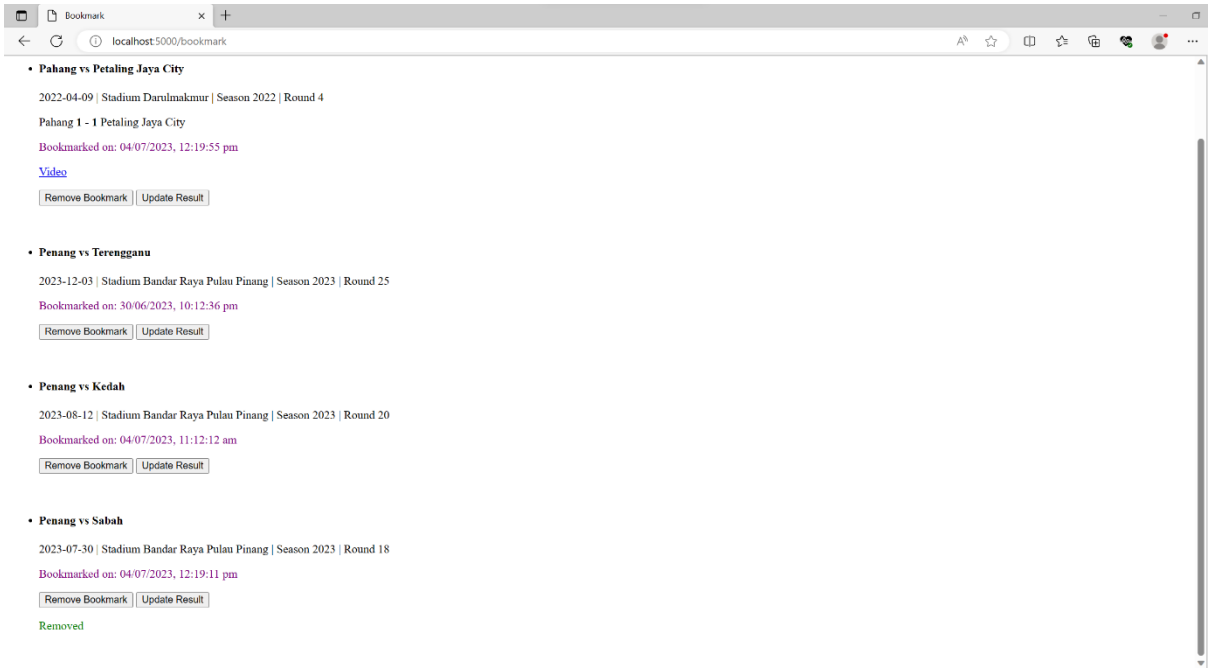

Figure 14 Bookmark page (Remove bookmark)

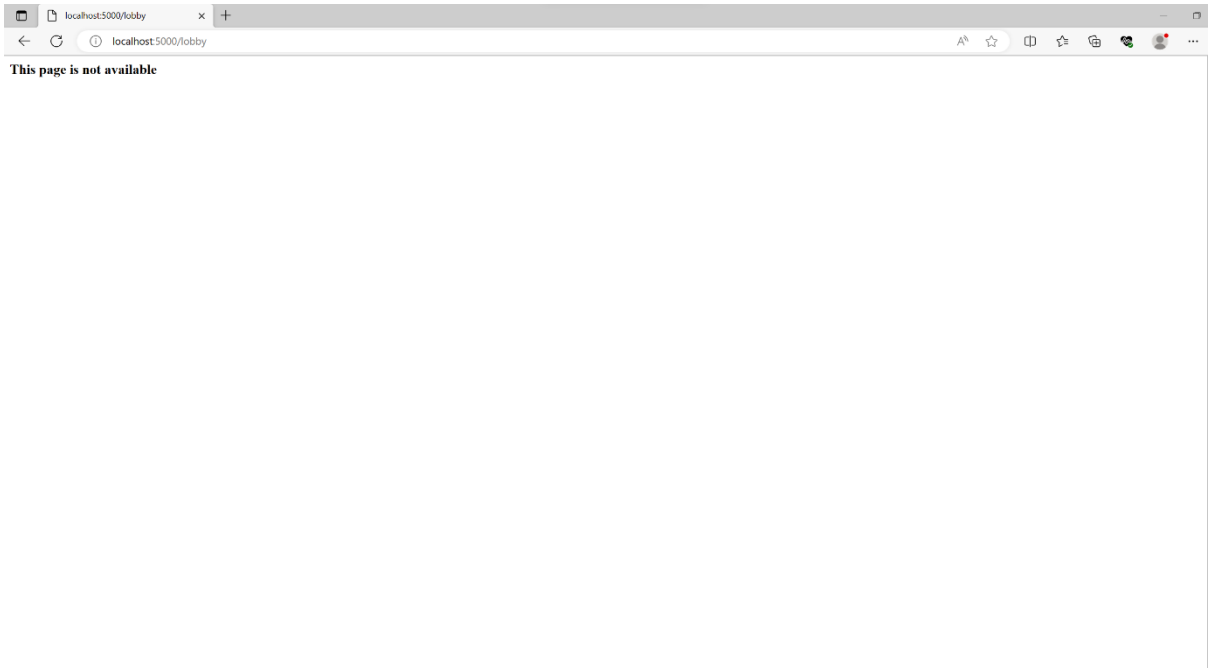

Figure 15 Page not available

When the “Bookmark” page is loaded, the list of bookmarked events will be displayed, as demonstrated in Figure 9. The features in “Bookmark” page are remove bookmark and update bookmark. The update bookmark feature updates the database with the data from the API. The fields that will be updated are “video” (the video link), “homeScore” and “awayScore”. Figure 10. demonstrates the update bookmark feature when the update operation is successful, whereas Figure 11. demonstrates the update bookmark feature when there is no latest update (no video link and results). The remove bookmark feature allows user to remove an event from bookmark list. Figure 12. demonstrates the remove bookmark feature when the remove operation is successful. When the requested route is not available, the server renders “This page is not available” message, as shown in Figure 13.

References

Farid, F. (n.d.). *What is Access-Control-Allow-Origin in Axios?* Retrieved from Educative: https://www.educative.io/answers/what-is-access-control-allow-origin-in-axios

Getting the request body in Express. (2019, October 22). Retrieved from Mastering js: https://masteringjs.io/tutorials/express/body

gouravhammad. (n.d.). *Password hashing with MD5 module in Node.js*. Retrieved from Geeks for geeks: https://www.geeksforgeeks.org/password-hashing-with-md5-module-in-node-js/

Shahid. (2021, April 21). *Render HTML file in Node.js and Express.js framework*. Retrieved from Code geek: https://codeforgeek.com/render-html-file-expressjs/

Tkacz, L. (2022, March 26). *How to securely store JWT tokens*. Retrieved from https://tkacz.pro/how-to-securely-store-jwt-tokens/

Using HTTP cookies. (n.d.). Retrieved from MDN web docs: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies