Project Challenge

Basic Idea

Come up with a small app that you can build across the week (today through Thursday). It can be a rebuild of something that you know and like already (goodreads.com, twitter.com, a todo app), or even a project that you have in mind, although I would keep it simple for this week.

This project is not graded, and just for you to get your hands on the tools, and to practice in longer form what you have been doing over the past six weeks. You can expect things to go wrong, and for it to be hard, and we will be here to guide you along the way. If you get stuck or don't understand something, that's no worries - come for help. If there is something that several people are getting stuck on, then we can run review classes. This is a project, but it's time for learning.

Also, please help each other out. Cooperation and teamwork are a huge part of coding. Explaining things to others is often a good way to check your own understanding (believe me). Use each other as resources.

Deliverables

The idea will be to build up a small app over the course of Monday to Thursday with the following stipulations:

- Javascript on the front-end.
- An Express API as the backend.
- At least one Axios GET request from the front-end to the API.
- At least one Axios POST request from the front end to the API.
- It must have at least one form.
- It should build up some data on the backend, either as an in-memory structure, or written to a file.
- It can be structured as one page only.
- Some basic styling. These are the minimums.

Project suggestion

Build an app to store a record of your favourite music. Start very simple and build from there. Plan out your app. What will the data look like? How will you add an artist or an album? Even getting to the point of being able to add an album, and see a series of albums rendered to screen is good starting point. You can build from there. The next step would be to build in all the CRUD operations.

Possible steps (if a bit stuck)

1. We are going to start this as a new project to get used to putting it all together (unless you have started a project you like with Mike last week).
2. Set up your project folders for front- and back-ends.
3. Create a basic HTML structure with a heading, and a div for the data to be attached to. You can keep this simple for now, and it can be rearranged as you get deeper into the project.
4. Create your backend server in Express.
5. Hit the server from the front end, and check that you are receiving data. This can be as simple as a 'hello world' response. You may need to solve some CORS issues.

6. Think through your data, and then create a basic data store using a JS structure (an array or an object). Add a few items of data to the structure (hard coding them in) so that you have something to work with.
7. Create a route to serve that structure.
8. Have the front-end make a call to that route and check that the you are able to get a response with the data.
9. Now that you are getting data on the front-end, you might need to start thinking about structuring that within your `index.js`. You can have a variable in the global scope to contain this data, and this can be your 'single source of truth' for the front end, meaning that this is the place that all other data eminates from in the front end of your app.
10. As a basic starting point, you can pull all the data in one hit from the database when the app starts up, and store it into this front-end data storage point.
11. When that data comes in, you want to render it to screen in some way. You might start by doing this in the `.then` of the Axios request to get all the data. You will need to create some elements and cycle through the data appending these elements to a parent (probably a div in your HTML that is set up to be ready to have these elements attached).
12. If you manage to get this working, the next step would be to look at cleaning up your code. I would suggest that you put all the logic you have just made in the `.then` into a function/s in the global scope. The function should take in the data as an argument. Then you will call this function in the `.then`.
13. Add a form to your HTML. We can abstract this later, but for now it might be easier to just add it straight into the HTML, probably below your list of albums.
14. You will need to add a route to deal with the incoming data from the form in your Express app.