Meeting: Tuesday, Jan 21, 11am - 12pm & Friday, Jan 24, 5pm-5:45pm

Languages

React JS (Avi, Jerome, fatima)

* Pro
  + *Promotes reusable components, supports fast rendering with a virtual DOM, and integrates well with other libraries and frameworks*
  + *Components are encapsulated, which means they manage their own state and render independently*
* Con
  + Has a steep learning curve due to JSX and state management, and requires significant repetitive code for project setup
  + view layer limitation
    - <https://ellow.io/advantages-and-disadvantages-of-react-js/#:~:text=The%20view%20layer%20limitation%20of,often%20need%20to%20integrate%20React>.
  + Tailwind CSS
    - Pro
      * Enables fast and efficient styling with utility classes, ensures design consistency, and allows extensive customization through configuration files
    - Con
      * Can lead to cluttered HTML with long class names and requires initial setup
  + Express framework of Node.js (Russell)
    - Pro: Quicker and more intuitive than using base Node for everything back-end (<https://www.geeksforgeeks.org/node-js-vs-express-js/>)
    - Con: Multiple middleware solutions for one problem, so everyone has to be on the same page to prevent spaghetti (this link may be useful <https://sandydev.medium.com/node-js-express-js-5-common-bad-practices-and-their-optimized-solutions-805e22597531>)

Next JS(Avi, Ashu)

**Pros**

* + Uses SSR (react uses client side, all this means is that everything will be loaded on the initial page visit, so when users switch between pages, which I assume they will have to frequently on our site, it will be really fast, where as react renders each page separately, so when users go from their dashboard, to their settings page, or etc, it will re render it and take time.
  + Managing routing is easy in next since it is built in, you just create a new folder for each route (a route is like [coursemix.com/about](http://coursemix.com/about) or [coursemix.com/dashboard](http://coursemix.com/dashboard)) the route is the part after the /, aka different pages
  + NextJS is better for large scale apps, whereas react is better for a single page app, since our app is going to be more than a single page, nextjs is better (mostly because of the SSR vs CSR) but next is simply built as a framework on top of react, so they aren’t really all that different, and the same results can be achieved in react as well, it just takes more configuration and a bit more work on our end
  + Can use Tailwind CSS as well here
  + Faster
  + More structured to build an app
  + Uses typescript or js

**Cons**

* Python (web scraping)[Ola]
  + **Permission check**: added /robots.txt to the URL to verify web scraping permissions.
  + Pro
    - **Feasibility**: It is highly doable with the right tools and strategies. Successfully implementing it could significantly enhance web integration and data persistence within the project.
    - **Scalability**: If managed well, the solution could be extended to automate data retrieval processes for other tasks, saving time and effort.
  + Con
    - **Complexity**: While using Python and Beautiful Soup, I successfully designed a script to scrape and retrieve all the HTML from the target webpage. However, I quickly realized that the process involves more intricacies than initially expected.
      * **Data Parsing**: Extracting specific data (e.g., course details) requires deeper logic and advanced parsing techniques. This might include the use of ‘regexes’.
      * **Structure Variability**: Websites often have inconsistent or dynamic HTML structures, making the design of a universal scraping logic challenging.
    - **Time Constraints**: Additional time—possibly another week—will be needed to develop a workaround for parsing the course information and designing robust logic to handle variations in the data structure.
* Django (Russell)
  + Pro
    - As a communicator, works seamlessly with SQLite: creating a Django project **automatically creates** an SQLite database (<https://www.w3schools.com/django/django_models.php>)
    - A Python framework, so Python experience applies
  + Con
    - Python’s structure and intricacies can be unintuitive

Database:

Supabase (Ashu)

**Pros**

* + Handles user authentication for us securely so we don’t need to learn additional auth libraries or frameworks
  + Supabase is a relational sql db hosted for us on servers, so we do not have to host our own db somewhere and worry about it, and their free tier is fairly generous, other alternatives will likely cost us more
  + Has many features built in for us, such as requiring a captcha, don’t need to seek out other libraries to implement these features, and it's all handled through one service

**Cons**

* + **The con is that there is no con!**

SQLite3

* + Pros
    - SQLite3 is a serverless database, meaning it requires minimal setup and is easy to integrate into applications. This makes it ideal for small projects or embedded applications.
    - The entire database is stored in a single file, which simplifies backup and transfer. You can easily copy or move the database file without complicated processes.
    - SQLite3 works on various operating systems (Windows, macOS, Linux), making it versatile for different development environments.
    - Since it operates without a server, there’s no need for separate installation and configuration, which saves time for developers.
  + Cons
    - SQLite is used to handle low to medium traffic HTTP requests.
    - Database size is restricted to 2GB in most cases.
    - SQLite does not support plugins like [TimescaleDB](https://github.com/timescale/timescaledb) for Postgres.
    - SQLite does not support enums which means you're forced to use strings.

Sources: <https://www.javatpoint.com/sqlite-advantages-and-disadvantages>

<https://www.epicweb.dev/why-you-should-probably-be-using-sqlite>

Authentication Technique: JWT or OAuth, google recaptcha 4.0

**Tuesday Task: Research the options, add pros and cons you find, we will discuss it on Friday, and select the architecture to use. Add at least 1 unique pro and 1 unique con for chosen language.**

**Packages**

React + Supabse (next)

Python + DJango + SQLite

Python + DJango + Supabase

**Friday Task:**

**Download the required software for NEXTJS.**

**If you want: Play with it, and try linking it to Supabase. To Guide you through the learning process, try creating a signup/login page with NEXTJS, linked to usernames and passwords from Supabase database.**

**Due: Tuesday 11:00 am**