1. Fill out this form to help us evaluate you better: https://forms.gle/tETMxwUHHJak7TVK8
2. Go through the website: https://capx.live/
    a. There are three pages on the site. You will have to **sign up via your google account** to access the last one i.e. Profile Page. Pages are Home page, Join waitlist and Profile.
    b. Please fill this survey after that to give your feedback on the site and improvements needed for the same. **Please use the same email to fill the survey as the one you used for signing up on CapX site.**
        i. Survey link: https://forms.gle/1bEvZyFPYQAzAzz28
3. **Assignment:**

   Build a **Simple Portfolio Tracker** application that allows users to:

    1. Add, view, edit, and delete stock holdings.
    2. Track the total portfolio value based on real-time stock prices.
    3. View a dashboard displaying key portfolio metrics (e.g., total value, top-performing stock, portfolio distribution).

Requirements:

Frontend:

- Build a responsive web application using **React** (preferred) or any modern frontend framework.
- The interface should include:
    - A **dashboard** showing portfolio metrics.
    - A **form** to add/edit stock details (e.g., stock name, ticker, quantity, buy price).
    - A **list/table** displaying current stock holdings with options to edit or delete them.

Backend:

- Build the backend using **Java** with **Spring Boot (or Dropwizard)** as the framework.
- Requirements:
    - Expose RESTful APIs to:
        - Add a new stock.
        - Update existing stock details.
        - Delete a stock.
        - Fetch all stocks and calculate the portfolio value.
    - Use **JPA** and Hibernate for database interactions.
    - Properly handle exceptions and include meaningful HTTP status codes.

Database:

- Use **MySQL** (preferred) or any relational database.
- Design a schema to store stock details (e.g., stock name, ticker, quantity, buy price).
- Include relevant relations if needed (e.g., users and portfolios).

Real-Time Data:

- Integrate with a free stock price API (e.g., Alpha Vantage, Yahoo Finance, Finnhub).
- Use these prices to calculate the total portfolio value dynamically.
- Pick any 5 stocks randomly for each user and these 5 stocks will constitute the portfolio of the user.
- For assignment purposes, the quantity of each stock purchased is assumed to be 1.

Deployment:

- Deploy your application:
  - Backend: On platforms like Heroku, AWS, or Render.
  - Frontend: On platforms like Vercel or Netlify.
- Share the deployed link with your submission.

Evaluation Criteria:

1. **Functionality:**
   - Application meets the core requirements and performs CRUD operations effectively.
   - Live stock prices are integrated, and portfolio value updates dynamically.
2. **Code Quality:**
   - Clean, modular, and readable code.
   - Proper use of Java and Spring Boot design principles.
3. **UI/UX Design:**
   - Responsive and intuitive interface.
   - Proper use of styling frameworks (e.g., TailwindCSS, Bootstrap, or Material-UI).
4. **Technical Knowledge:**
   - Demonstrated understanding of frontend and backend technologies.
   - Efficient database schema design.
   - Proper API design and exception handling.

Submission Instructions:

1. Upload your code to a public repository (e.g., GitHub, GitLab).
2. Include a **README** file with:

    - Steps to run the project locally.
    - Any assumptions or limitations.
    - Links to deployed application and live API documentation (if any).
3. Put the repository link in a PDF and upload the PDF file on Unstop
4. Instructions on how to upload available on Unstop assignment listing

---