elev8.ai

elevate your finance game.

GitHub Repository: https://github.com/Samash1129/Final-Year-Project

System Requirement Specification

(Semester 7)

Syed Ali Mashoud – 23068
AmanUllah Faisal – 22944
Hassan Yahya – 22965
Ibrahim Abdul Rehman – 22748

Supervisior: Mr. Behraj Khan
Mentor: Mr. Raheel Siddiqui (VP Product Sales - Systems Limited)

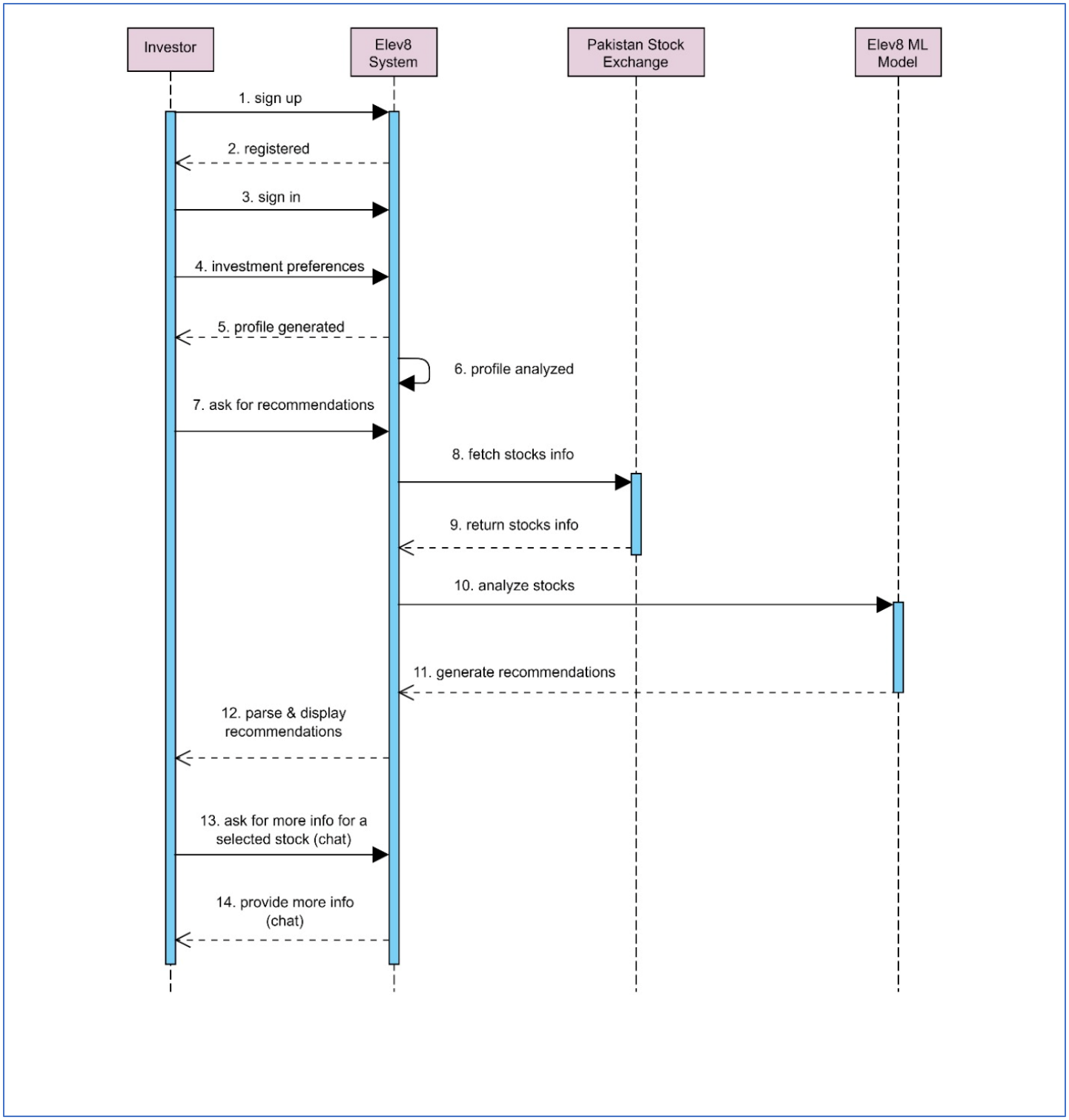
Submitted to Projects Manager- FYP

IBA SMCS
School of Mathematics and Computer Science

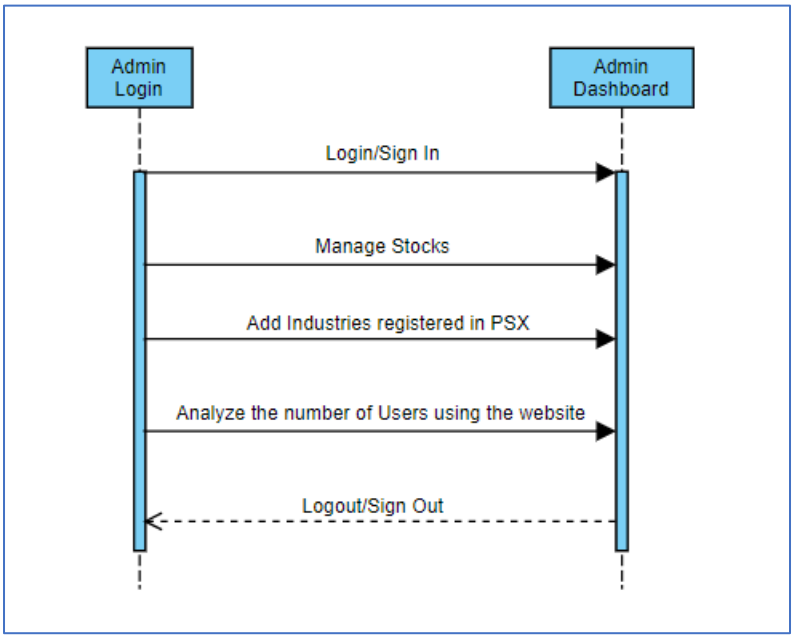
INTERACTION DESIGN

1. Sequence Diagrams

I. User Sequence Diagram

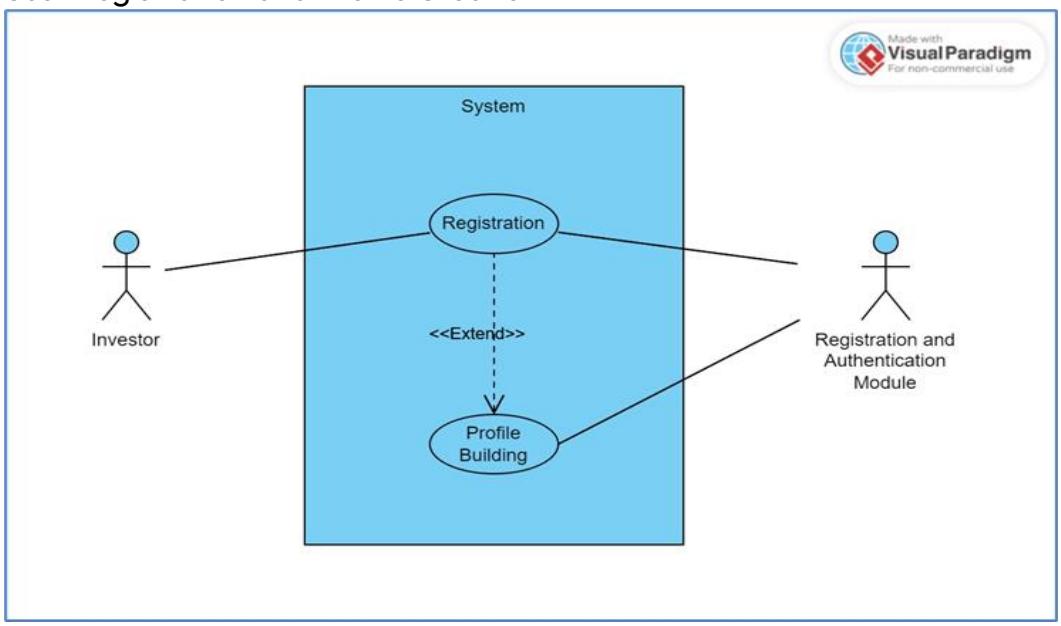


II. Admin Sequence Diagram

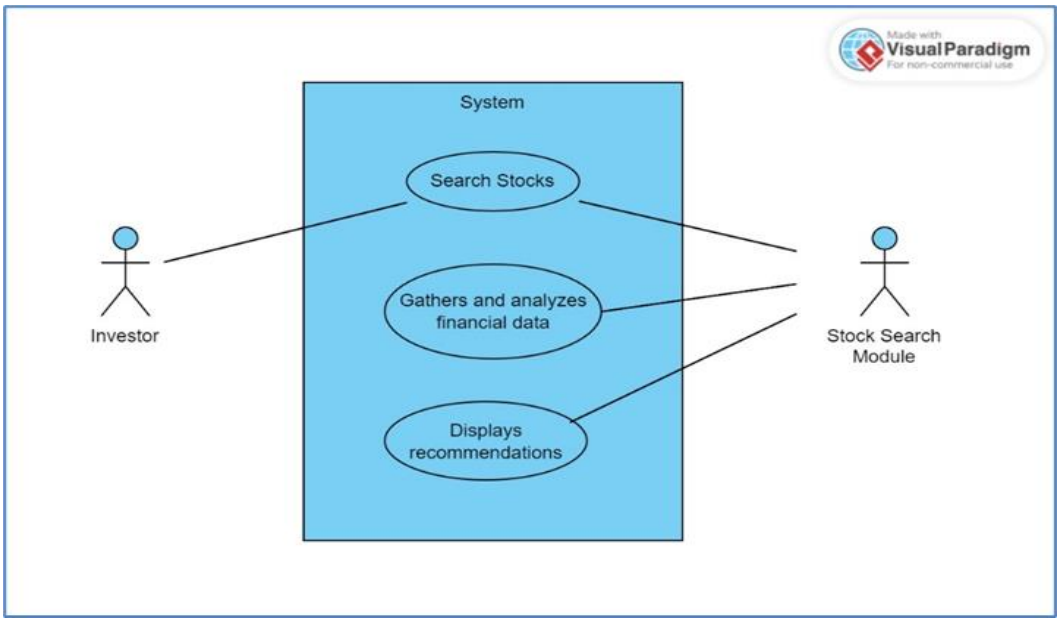


2. Use Case Diagrams

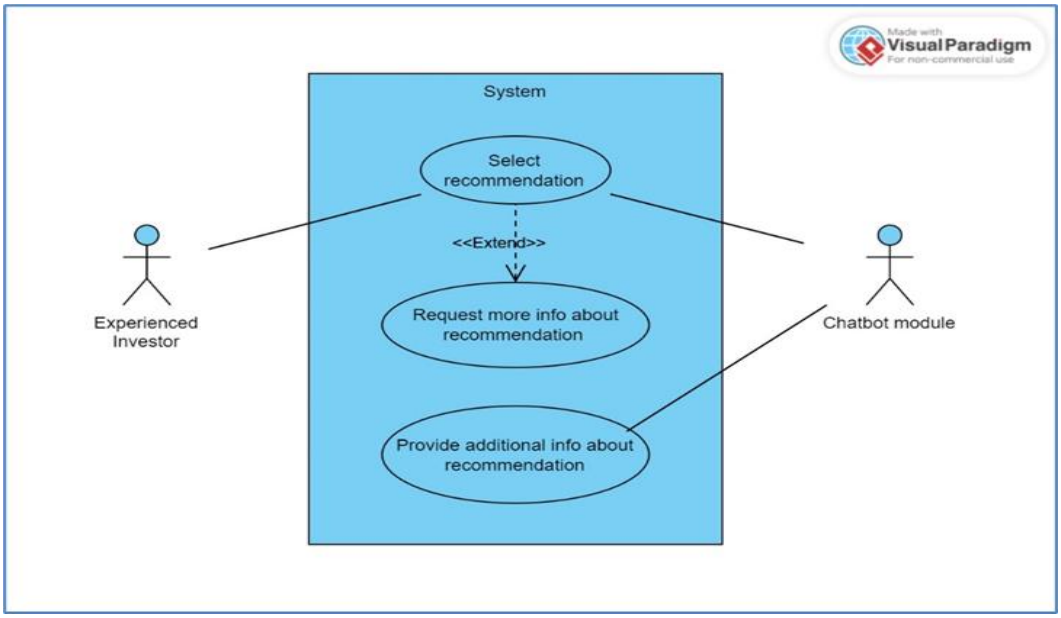
I. User Registration and Profile Creation



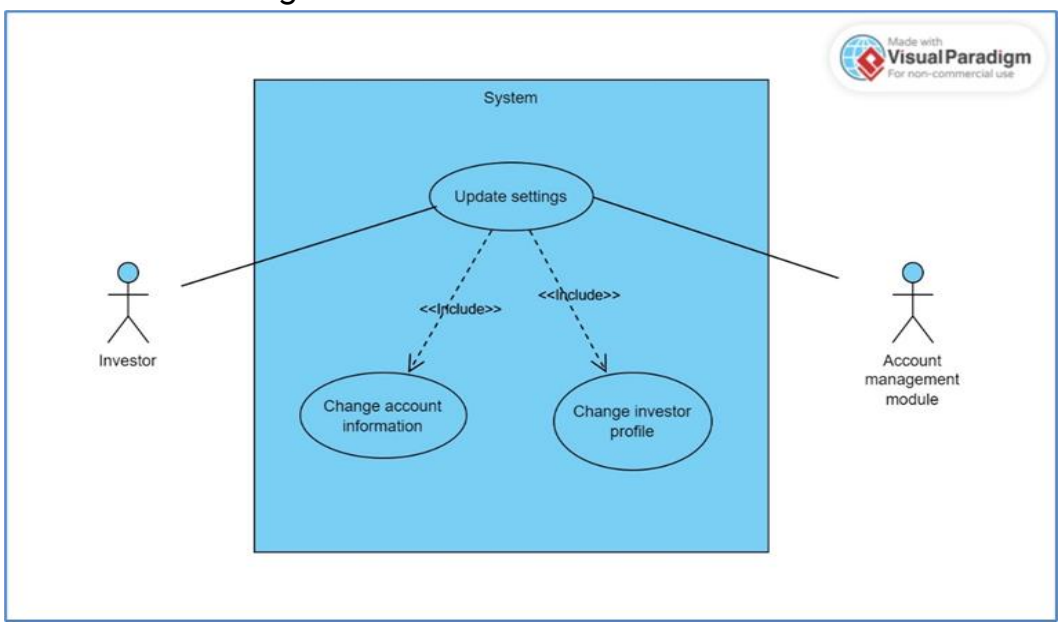
II. Stock Selection and Generated Recommendations



III. Chatting with chatbot – Elevy (premium users)



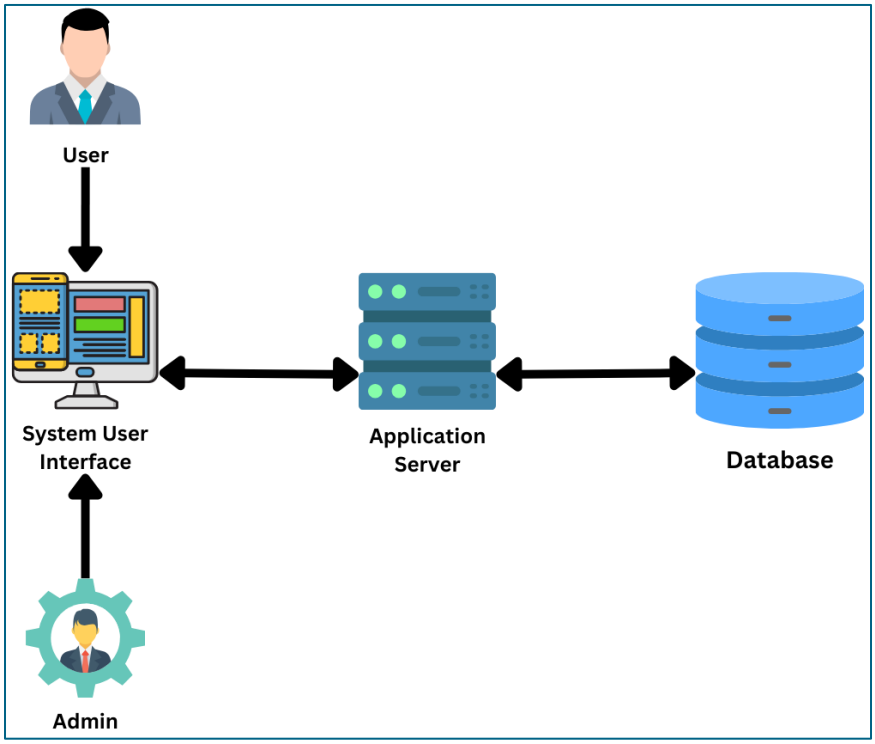
IV. User Account Management



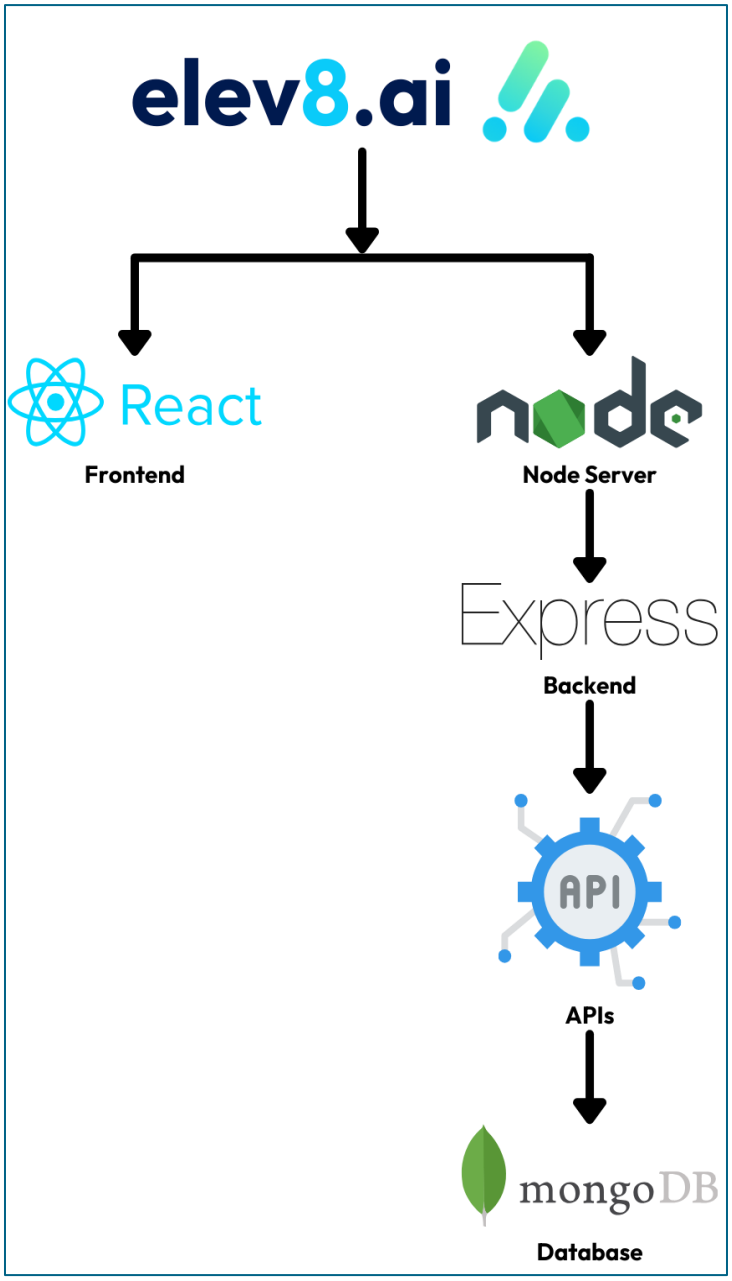
SYSTEM ARCHITECTURE & SYSTEM DESIGN

1. **Architectural Style**

elev8.ai follows a client-server-based architecture.



2. Identifying Subsystem



3. Mapping Subsystems to Hardware

In a MERN-based webapp,

I. ReactJS will be running on the client's machine – Frontend.
II. NodeJS will be running on the server along with ExpressJS – Backend
    i. NodeJS – JavaScript runtime environment
    ii. ExpressJS – Runtime framework.
III. Database will be cloud-based – MongoDB Atlas

4. **Persistent Data Storage**

I. User Schema

```
{
        username: String,
        email: String,
        password: String,
        isAdmin: boolean
}
```

II. Preference Schema

```
{
        investmentGoals: String,
        riskTolerence: String,
        amountToInvest: number,
        industries: [
                {
                        industryName: String
                }
        ],
        stockType: String,
        users: [
                {
                        username: String,
                        userID: String
                }
        ]
}
```

III. Industry Schema

```
{
        industryName: String,
        companies: [
                {
                        companyName: String
                }
        ]
}
```

5. **Network Protocol**

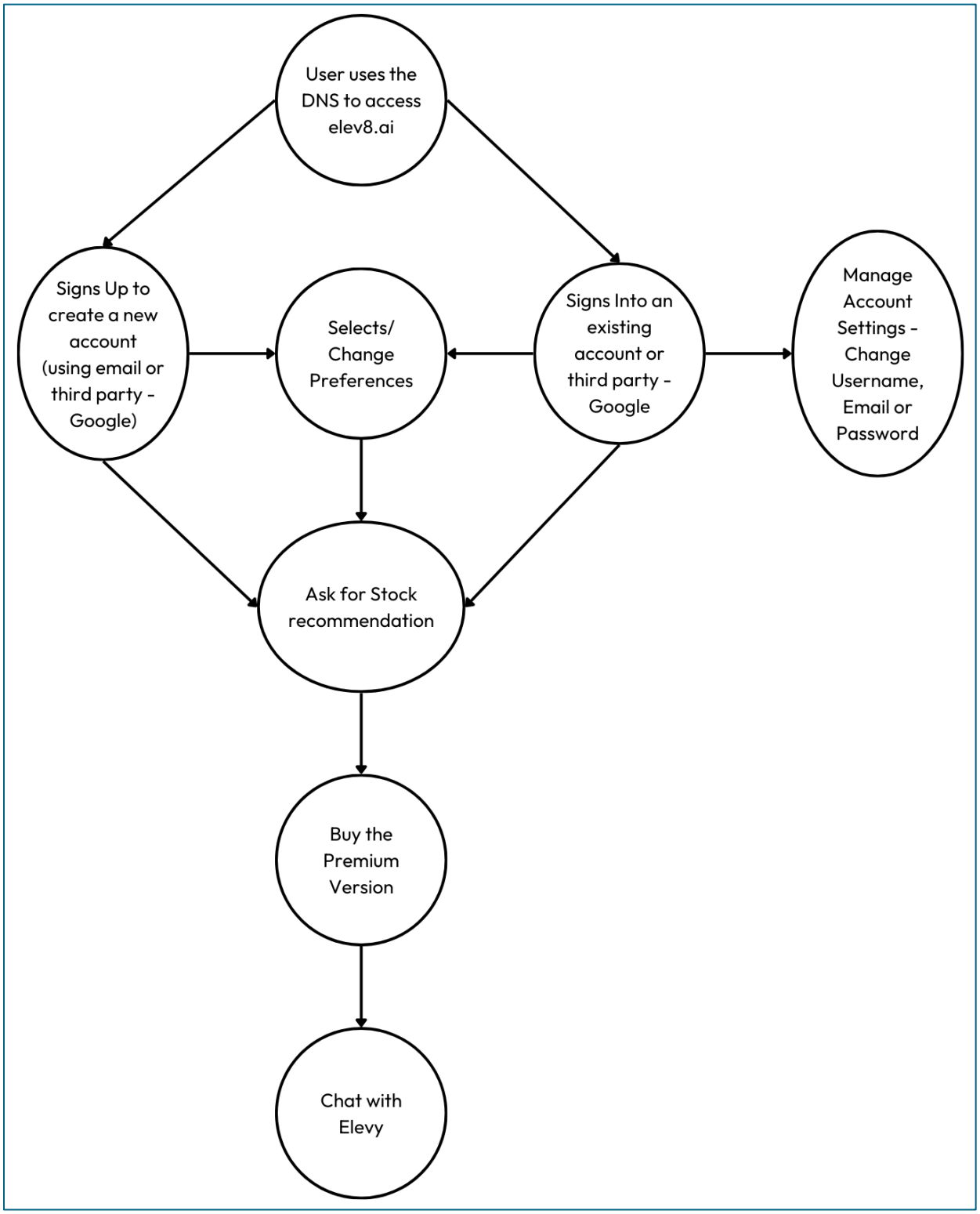
I. Following Network Protocols are to be used in elev8.ai

i. HTTPS (Hypertext Transfer Protocol Secure) OR HTTP (Hypertext Transfer Protocol) – For API communication and routing

ii. WebSockets – For maintaining the chat feature with Elevy

iii. DNS (Domain Name Server) – This will be used by the user to access elev8.ai

6. Global Control Flow



7. Hardware/Software Requirements

I. Minimum Hardware Requirements

i. Laptop/Desktop

- RAM: 4 GB
- Processor: Dual-core 1.8 GHz
- Storage: Around 5 GB (for caching etc)
- Display: 1280x800 resolution

ii. Server

- RAM: 12-16 GB
- Processor: Quad-core 2.4 GHz
- Storage: 50 GB
- Network: 1 Gbps Ethernet

iii. Mobile

- RAM: 2 GB
- Processor: Dual-core 1.4 GHz
- Storage: 3-4 GB (for caching etc)
- Display: 720x1280 resolution

II. Minimum Software Requirements

i. Laptop/Desktop

- OS: Windows 10, Windows 11, macOS, or Linux
- Web Browser: Google Chrome, Mozilla Firefox, or Microsoft Edge
- NodeJS and npm: NodeJS version 14.x, npm version 6.x
- Database: MongoDB Atlas

ii. Mobile

- OS: Android 7.0+, iOS 12.0+
- Mobile Browser: Chrome, Safari

iii. Required Protocols

- HTTP/HTTPS: For communication between client and server
- WebSockets: For managing chats with our chatbot
- DNS: For finding our website on google

iv. Drivers

- Database Drivers: MongoDB, NodeJS drivers for server communication
- Network Drivers: Mobile Network Drivers for connectivity

v. Mobile-Specific Considerations

- Responsive Design
- Touch Interaction

8. **Technical Stack**
    I. Frontend
        i. Framework: ReactJS
        ii. Programming Language: JavaScript (ES6+)
        iii. State Management: Redux
        iv. UI-Design: Material UI or creating our own components
    II. Backend
        i. Framework: ExpressJS (NodeJS)
        ii. Database: MongoDB
        iii. Authentication: JSON Web Tokens (JWT)
        iv. API Documentation: Swagger/apiary
    III. Version Control
        i. Git: Github Repository
    IV. Development Tools
        i. Code Editor: Visual Studio Code
        ii. Package Management: npm (Node Package Manager)
    V. Machine Learning Model
        i. Programming Language: Python
        ii. Libraries: PyTorch, TensorFlow, or Scikit-Learn
        iii. Model: Regression Models (*subject to change*)
9. **API Documentation**

URL to the API Documentation: (To be added later)

    I. /login → Post API
        i. The user shall login to his dashboard using a valid username/email and password
    II. /generateOTP → Post API
        i. This will be used to generate a one-time passcode that will enable the user to change his/her password
        ii. The OTP will be received on the email of the user
    III. /changePassword → Put/Patch API
        i. The user shall enter his new password
    IV. /signup → Post API
        i. The user shall create a new account given that the account that is to be created does not exists
        ii. User's password will be hashed then stored using any of the libraries of JavaScript
    V. /logout → Post API
        i. Logs out the user based on a valid token
    VI. /dashboard/selectPreference → Post API
        i. After signing into the account, the user will be prompted to select his/her preference on which the user shall receive the suggestions on stocks that clearly assigns with his selected preference
    VII. /dashboard/searchStock → Get API
        i. This will filter out the stocks. Some features will work from the client side

VIII. /dashboard/recommededStock → Get API
    i. This api call will call all the recommended stocks after performing the necessary machine learning algorithm
IX. /dashboard/stockHistory → Get API
    i. The api call will get all the stocks a particular user has search or has asked for previously
X. /dashboard/report/stockID → Get API
    i. Clicking on a particular stock will call this api, it will simply generate a brief report on selected stock
XI. /dashboard/report/stockID/detail → Get API
    i. If a user wants to access a detailed report on the selected stock, this api will be responsible for the task
    ii. In simple words, this api call is a continuation of the previous api call
XII. /changeProfileDetails → Put/Patch API
    i. Performs necessary changes to the data based on the user's choice
XIII. /changePreference → Put/Patch API
    i. Allows the user to change his/her stock preference.

ALGORITHMS AND DATA STRUCTURES

Our system will be advising the user based on the following metrices,

I. Historical Share Price Growth:
    i. A detailed exploration of literature on historical share price growth and the application of regression models for predicting and analyzing trends. The section emphasizes the significance of numerical analysis in capturing long-term market dynamics.
II. Recent Price Performance:
    i. Examining recent price performance within a some time frame, this section delves into literature on the application of regression models for short-term/long-term analysis. It evaluates the trade-offs between capturing recent market dynamics and the challenges posed by data volatility.
III. Financial Ratios from the Last Financial Year:
    i. Building on the foundation of financial ratios, this section reviews literature on their significance in predicting stock performance. It explores how the integration of financial ratios enhances the predictive capabilities of regression models.
IV. Industry Average Growth vs. Particular Stock Growth:
    i. Industry benchmarking is scrutinized in this section, focusing on literature discussing the comparison of industry average growth with the growth of specific

stocks. It explores the nuances of benchmarking and the implications for informed decision-making.

V. Dividend-to-Price Ratio:
   i. The literature review delves into the importance of dividend-to-price ratio as a financial metric. It explores how this ratio influences investment decisions and how regression models can be leveraged to predict and analyze dividend trends.

VI. Real-time Stock News:
   i. Examining the role of NLP in processing real-time stock news, this section reviews literature on sentiment analysis and information extraction from news articles. It discusses the potential of NLP in transforming textual data into actionable insights.

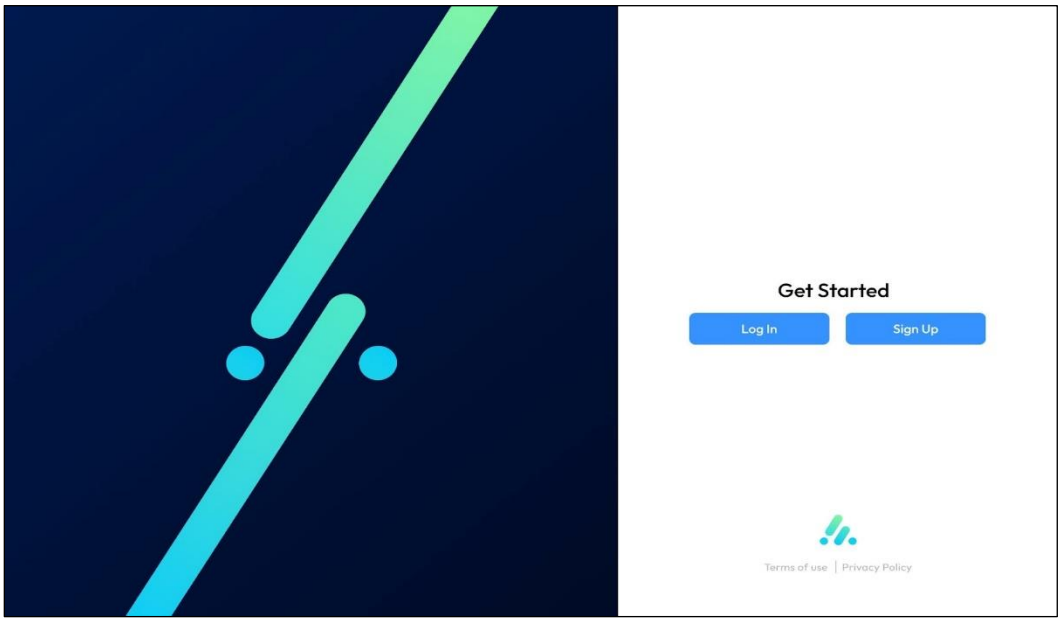
USER INTERFACE DSIGN AND IMPLEMENTATION

*Project will be built using the MERN stack and will need a scorn job tool. Figma URL can be find **here***

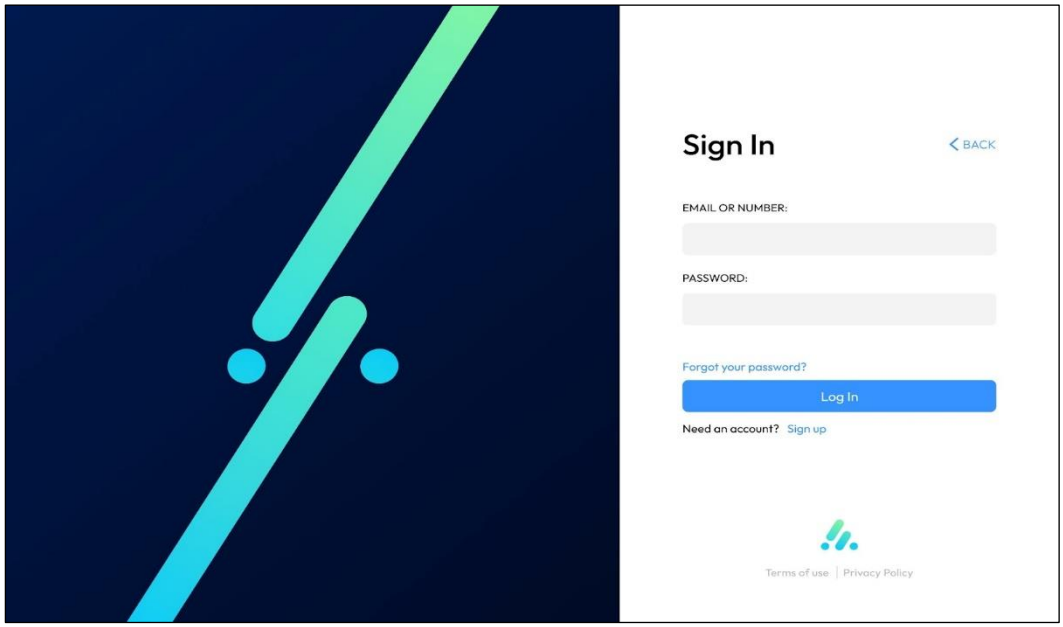
1. **Getting Started:**

   This portion of our UI involves Sign In with your existing account or Sign Up for a new account and login to our Main Dashboard. You can make a new account by simply entering your name, email address, and a secure password which further leads to the preferences you want to set for report generation.

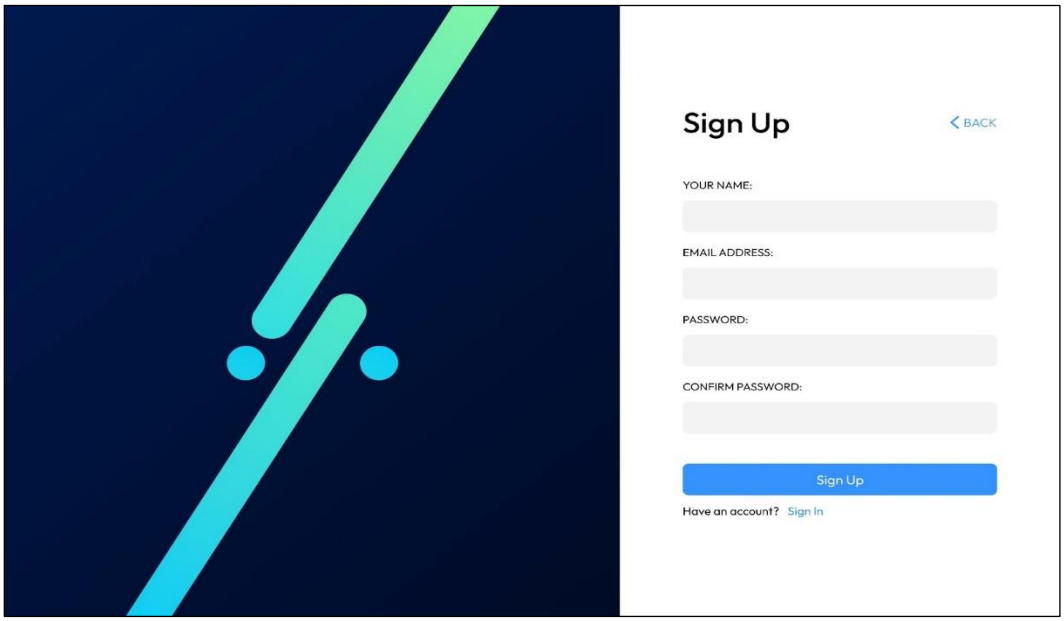
   I. Landing Page



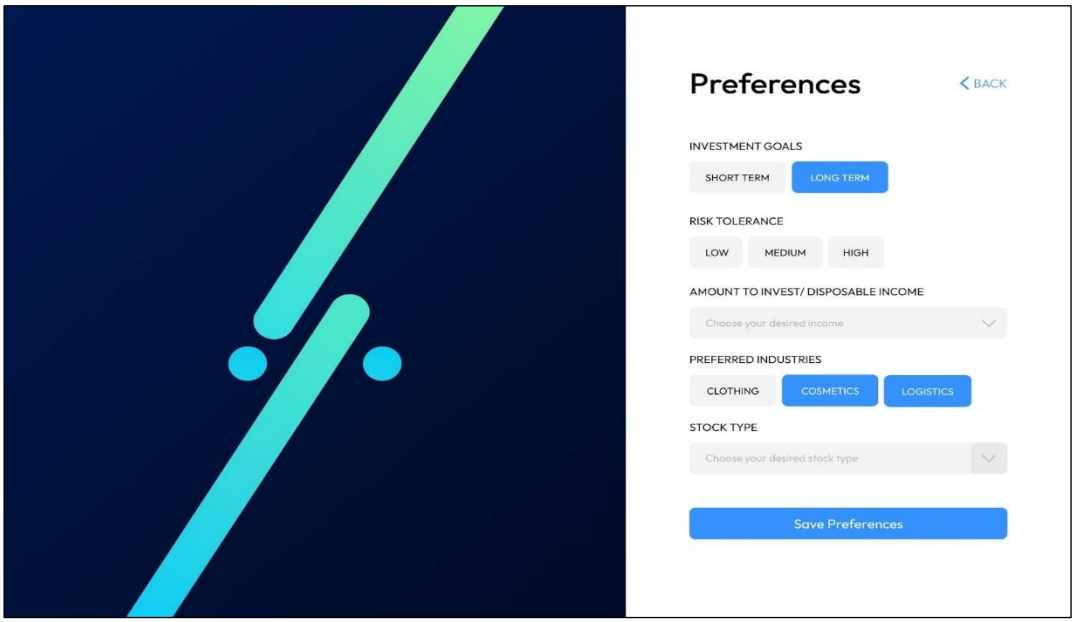
II. Sign In Page



III. Sign Up Page



IV. Preference Page



2. Main Dashboard:

This portion of our UI revolves around how you can generate stock reports within a few clicks.

I. Main Dashboard Page

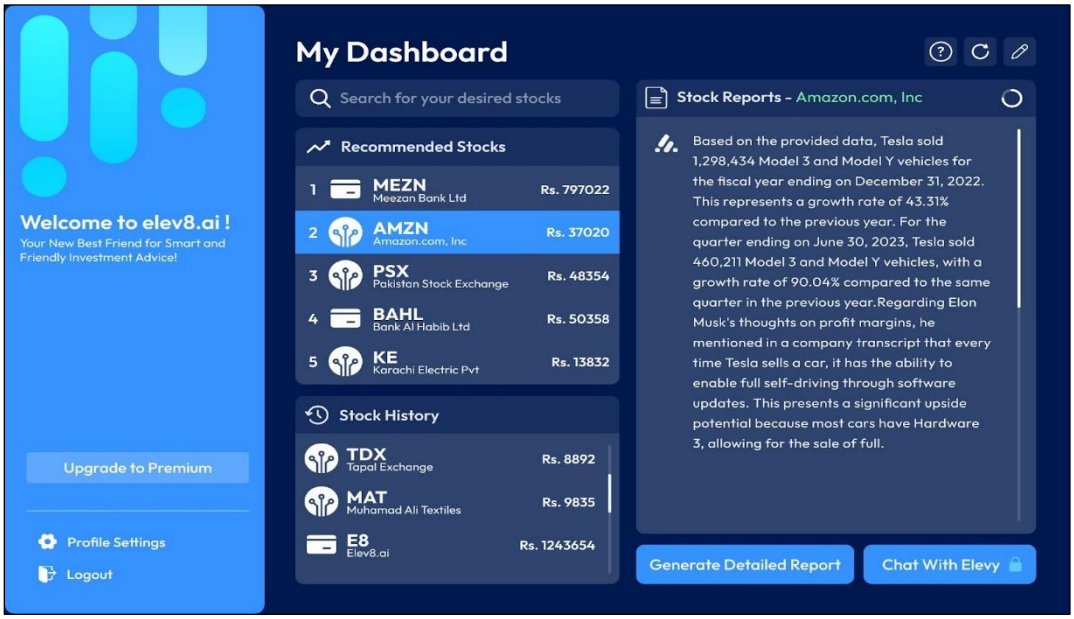
i. On the left side of your dashboard, you see a welcoming message and 3 buttons. Upgrade to premium opens a prompt where it guides with the benefits of our premium plan and purchasing cost.

ii. Profile Settings takes you where you can edit your account details and preferences.

iii. Logout simply logs you out of the site.

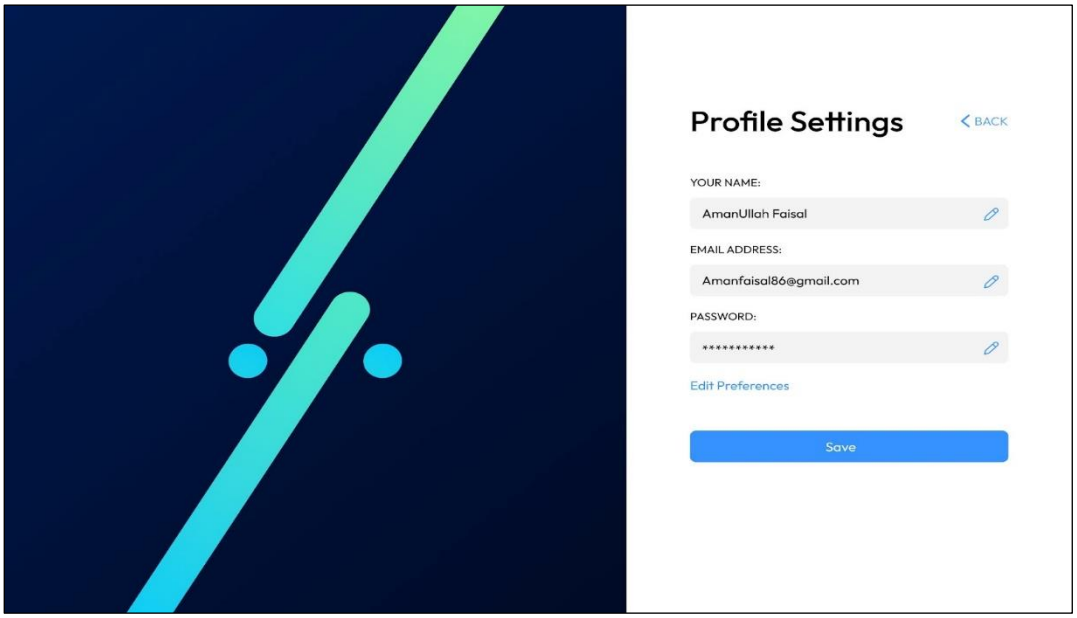
iv. In the center of the screen, you can find a search bar where you can find all the stocks according to user preferences and the history of previous stocks searched by the user.

v. Amongst them you choose one which takes you to the third and right portion of the screen, where the report of your chosen stock starts to generate. The report you see on your screen gives you a

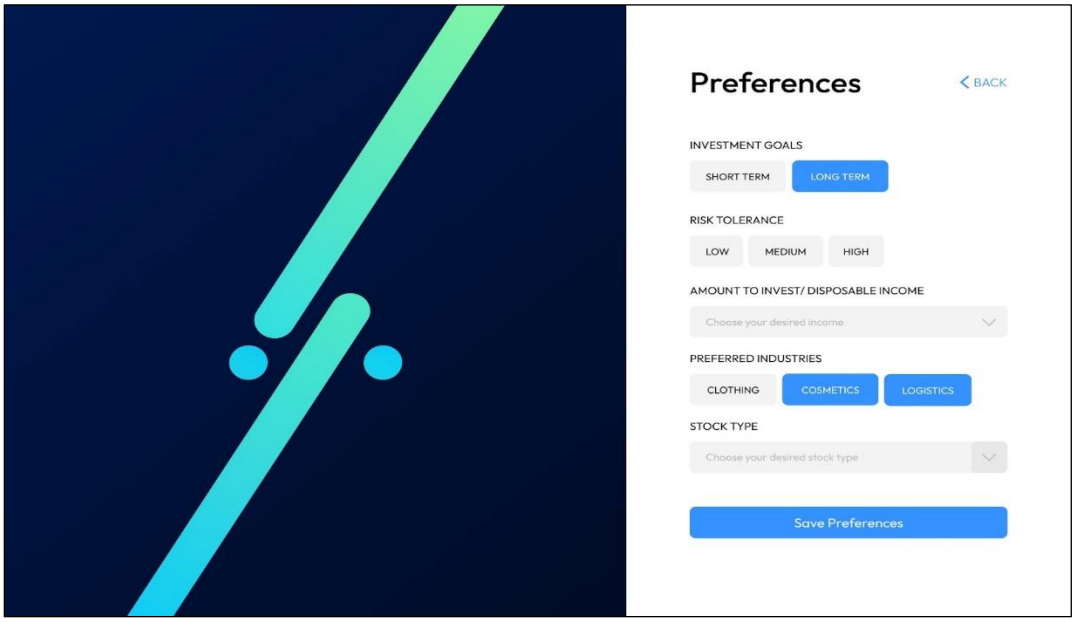


basic overview whereas if you click on the generate detailed report, it downloads a PDF file where you get detailed insights of the stock with a recommendation of buying it or not. Or you can even talk with our AI bot → 'Chat with Elevy' which is an integration of your reports with GPT-4, which transforms your reports into a powerful decision (Only available for premium user).

II. Profile Settings Page

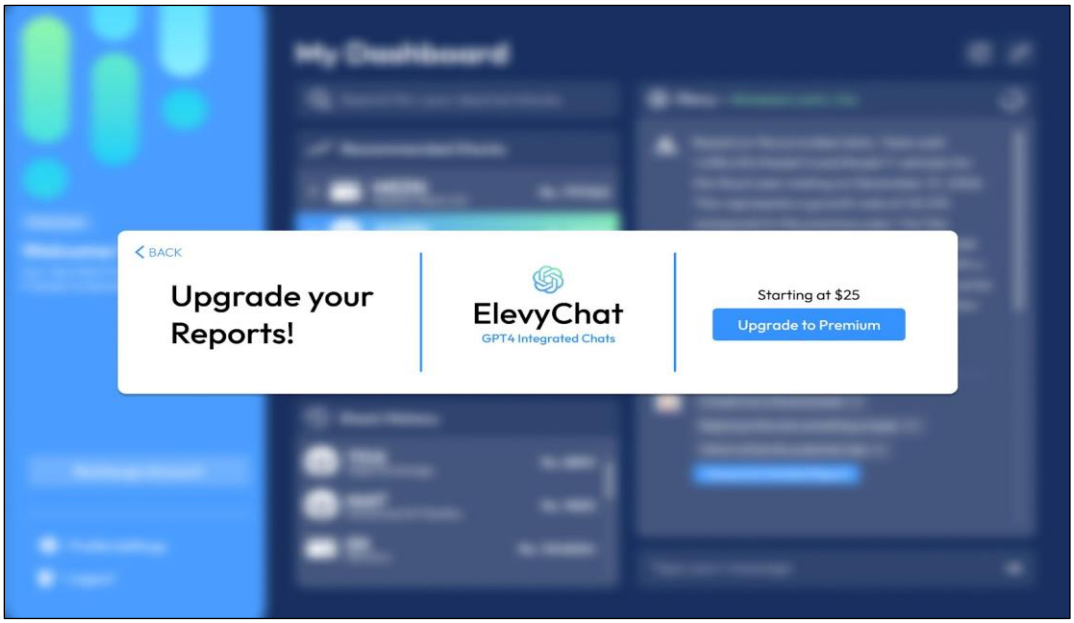


III. Update Preference Page



3. Purchasing Premium:

This portion of the UI involves viewing our premium plan and purchasing. (one time purchase)



4. Premium Main Account:

I. Dashboard

i. A few clicks and get more powerful insights with our very own chatbot → 'Elevy'. On the left side of your dashboard, you see a welcoming message and 3 buttons. Recharge Account opens a prompt where it guides you with the monthly plan the user has and remaining days of the plan and if he wants to extend.

ii. Profile Settings takes you where you can edit your account details and preferences.

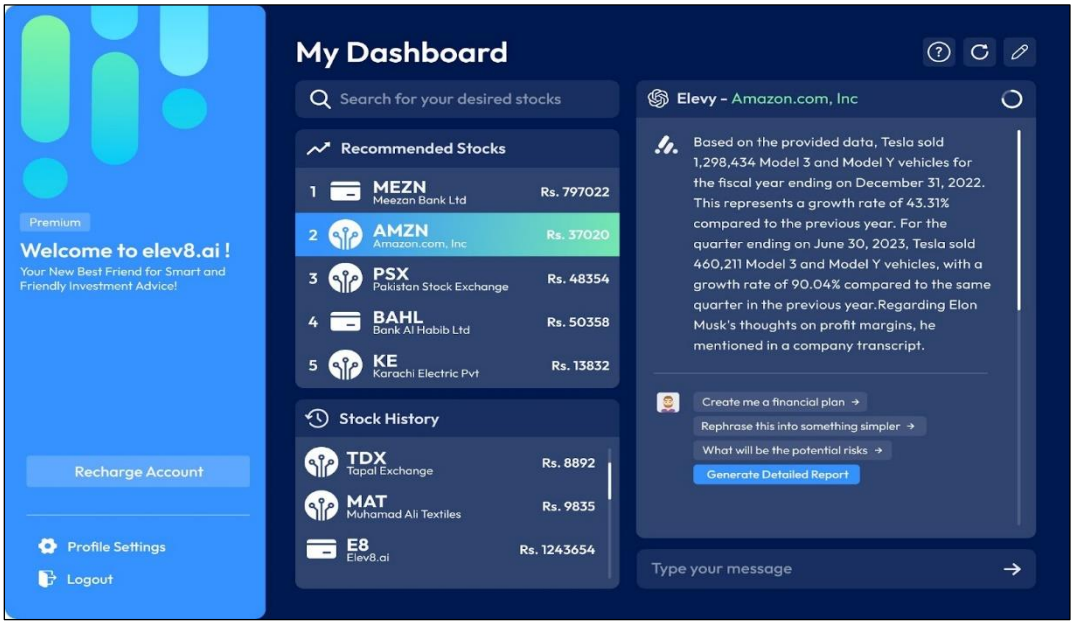
iii. Logout simply logs you out of the site.

iv. Then in the center of the screen you can find our search bar where you can find all the stocks registered to PSX. And once you leave that, it also shows the recommended stocks according to user preferences and the history of previous stocks he once searched.

v. Amongst them you choose one which takes you to the third & right portion of the screen, where the report of your chosen stock starts to generate. The report you see on your screen gives you a basic overview whereas if you click on the generate a detailed report, it downloads a PDF file where you get detailed insights of the stock with a recommendation of buying it or not.

vi. Or you can even talk with our AI bot -> 'Chat With Envy' which is an integration of your reports with GPT-4, which transforms your reports into powerful decisions. Now you can actually talk with your reports and ask about GPT's own recommendation and deep insights.

vii. This UI has a recharged look, like if you see some colors pop which makes the user feel more confident and powerful.



DESIGN OF TESTS

Not Decided as of yet.