**PROJECT DOCUMENTATION**

**INTRODUCTION**

**Project Title : A Cryptocurrency Dashboard**

|  |  |
| --- | --- |
| **TEAM MEMBERS** | **EMAIL ID** |
| NAFIZA BEGUM A | nafizaa185@gmail.com |
| MONIKA S | monikasmonikas2@gmail.com |
| MONISHA B | monishababu040@gmail.com |
| KETSIAL PRADEEPA | ketsialkiru@gmail.com |

**PROJECT OVERVIEW :**

The Crypto Currency Dashboard is a web-based application designed to provide real-time data, analytics, and insights on various cryptocurrencies. The platform enables users to track market trends, view price movements, manage portfolios, and access historical data.

### 2.1 Objectives

* Provide real-time cryptocurrency market data.
* Enable users to manage and track their cryptocurrency investments.
* Offer historical and analytical data visualization.
* Support user authentication for personalized experience.
* Ensure a responsive and user-friendly interface.

### 2.2 Target Audience

* Crypto traders and investors.
* Blockchain enthusiasts.
* Financial analysts.
* Developers and data scientists interested in cryptocurrency trends.

## PURPOSE :

The purpose of this project is to create a user-friendly and efficient dashboard that provides comprehensive cryptocurrency market insights. It aims to:

* Help investors and traders make informed decisions by providing real-time and historical data.
* Offer a portfolio management system to track investments and analyze performance.
* Enhance user experience with interactive charts, alerts, and notifications.
* Improve accessibility to cryptocurrency data through a well-structured and responsive web application.
* Support secure authentication and data protection for user accounts.

**FEATURES :**

Real-Time Price Tracking & Charts  
Display live cryptocurrency prices, price fluctuations, and historical data with graphical charts for various timeframes (hourly, daily, etc.).

1. Watchlist & Portfolio Tracking  
   Allow users to create a watchlist of preferred coins and track their personal portfolio, including profit/loss calculations and real-time updates.
2. Market Overview & Trending Coins  
   Provide an overview of top-performing cryptocurrencies, market cap, 24-hour volume, and display trending coins based on volume and social sentiment.
3. News & Alerts  
   Offer a news feed with the latest updates and allow users to set custom price alerts for specific cryptocurrencies.
4. User Authentication & Personalization  
   Implement user sign-up/sign-in functionality and allow users to customize their experience by saving settings, preferred coins, and theme options.

**ARCHITECTURE :**

### 1. Frontend (UI/UX)

* Framework: React.js (for dynamic and interactive user interface)
* State Management: Redux (for managing global app state, like user preferences, portfolio data)
* Charting Libraries: Chart.js or Recharts (to display price graphs and historical data)
* Design: Material UI or Tailwind CSS (for responsive and sleek design)
* API Integration: Axios or Fetch API (to get real-time data from cryptocurrency APIs)

### 2. Backend (Server-side)

* Framework: Node.js with Express.js (to handle API requests and serve data to the frontend)
* Authentication: Passport.js (for user authentication, OAuth with Google, Facebook, etc.)
* Database: PostgreSQL (to store user data, portfolios, and settings)
* API Integration: External APIs like CoinGecko, CoinMarketCap, CryptoCompare (for fetching live cryptocurrency data)
* Real-time Communication: WebSockets or Server-Sent Events (for real-time price updates)

### 3. API Layer

* Endpoints:
  + /api/cryptos: Fetch a list of cryptocurrencies with real-time price data
  + /api/portfolio: Handle user portfolio data (add/remove coins, track prices)
  + /api/alerts: Set user alerts for price changes
  + /api/user: Manage user authentication and settings

### 4. Data Flow

* Client Request: User interacts with the frontend (e.g., selecting coins or adding them to their portfolio).
* API Call: The frontend sends an API request to the backend (Express.js server).
* Backend Processing: The backend fetches data from external cryptocurrency APIs, processes it, and sends it to the frontend.
* Database Interaction: The backend stores user preferences, portfolios, and alert settings in PostgreSQL.
* Real-Time Updates: Use WebSockets to push real-time price changes or portfolio updates to the frontend.

### 5. Security

* Authentication: JWT (JSON Web Tokens) for managing secure user login sessions.
* Data Encryption: Encrypt sensitive data, like user credentials, in the PostgreSQL database.
* Validation: Input validation in backend to ensure that data submitted by the user is valid and secure.

## SETUP INSTRUCTIONS :

* Backend Setup:
  + Step-by-step instructions for setting up the Node.js backend (installation of required packages, database configuration, setting up environment variables).
* Frontend Setup:
  + Instructions for setting up the React app, installing dependencies, and connecting to the backend API

.

* Database Setup:
  + Instructions for creating the PostgreSQL database and tables, including necessary SQL commands.

### **INSTALLATION PROCESS :**

* Steps for Setting Up the Backend:
  + “Clone the repository, install dependencies using npm, configure PostgreSQL database, and run the Node.js server.”
* Steps for Setting Up the Frontend:
  + “Install frontend dependencies using npm and ensure API calls are properly configured.”
* How to Run the Application:
  + “Start the backend server with node server.js and the frontend with npm start.”

**FOLDER STRUCTURE :**

### 

### Backend (backend/)

* controllers/: Contains the logic for handling HTTP requests, including fetching data, processing, and returning it to the client.
* models/: Contains the models for interacting with the database (PostgreSQL in this case).
* routes/: Defines the routes that connect the backend to the frontend (e.g., /api/cryptos, /api/portfolio).
* server.js: The entry point for the Express server; this file sets up the backend app and listens on a port.
* .env: Environment variables (e.g., database credentials, API keys, JWT secrets) that should be kept secure.
* package.json: Manages backend dependencies (e.g., express, pg for PostgreSQL, axios for external API calls).

#### Frontend (frontend/)

* public/: Contains static files like the index.html page and assets (images, icons).
* src/: The source code for the frontend React application.
  + components/: React components that define the UI (e.g., charts, portfolio section, cryptocurrency list).
  + services/: Handles API calls to the backend, such as fetching cryptocurrency data or managing the portfolio.
  + context/: React Context API files for managing application-wide state (e.g., user authentication state).
  + App.js: The main React component where routes and UI components are assembled.
  + index.js: The entry point for React that renders the app into the index.html file.

#### Root-Level Files

* package.json: Manages the overall project dependencies and scripts.
* .gitignore: Specifies files and folders to be ignored by version control (e.g., node\_modules, .env).
* README.md: Provides documentation for setting up and using the project.
* docker-compose.yml: Optional file for Docker configuration, which is useful if you want to containerize both the frontend and backend services.

**COMPONENT DOCUMENTATION :**

### 1. Backend Components

#### 1.1 Server Setup (server.js)

* Purpose:
  + Initialize the Express.js server.
  + Set up middleware like CORS (for cross-origin requests) and bodyParser (to handle incoming JSON requests).
  + Serve as the entry point to handle API routes for cryptocurrency data, portfolio, and user authentication.

#### 1.2 Cryptocurrency Data Fetching (cryptoController.js)

* Purpose:
  + Handle logic to fetch real-time cryptocurrency data from an external API (e.g., CoinGecko).
  + Return the fetched data to the frontend.

#### 1.3 Portfolio Management (portfolioController.js)

* Purpose:
  + Handle the logic to manage user portfolios, such as adding coins to the portfolio.
  + Interact with the PostgreSQL database to store portfolio data.

#### 1.4 Routes (cryptoRoutes.js)

* Purpose:
  + Define routes that connect the frontend with backend functionality.
  + These routes handle API requests like fetching cryptocurrency prices, managing portfolios, and user login.

### 2. Frontend Components

#### 2.1 App Component (App.js)

* Purpose:
  + Acts as the main container for routing and rendering different views of the application (like crypto price list, portfolio, etc.).
  + Uses React Router to navigate between pages.

#### 2.2 Fetching Crypto Data (CryptoList.js)

* Purpose:
  + This component fetches real-time cryptocurrency data from the backend API and displays it.
  + It updates the UI dynamically with the latest price information.

#### 2.3 Portfolio Management (Portfolio.js)

* Purpose:
  + Allows users to manage their cryptocurrency portfolio by adding or removing coins.
  + Fetches and displays the portfolio from the backend, allowing users to track the value of their holdings.

#### 2.4 Watchlist and Alerts (Watchlist.js)

* Purpose:
  + Allows users to track cryptocurrencies they are interested in.
  + Users can add coins to a watchlist and set price alerts for those coins.

#### 2.5 API Service (apiService.js)

* Purpose:
  + A utility file for handling API requests to the backend.
  + Simplifies the process of making HTTP requests like fetching cryptocurrency data or managing user portfolios.

### USER INTERFACE :

* Crypto List: On the homepage, users see a list of cryptocurrencies and their prices.
* Landing Page: User sees a quick overview and logs in or creates an account.
* Portfolio Management: Users can add/remove coins to/from their portfolio and track their holdings.
* Watchlist & Alerts: Users can track coins they are interested in and set price alerts for notifications.
* User Profile: Users can view or update their account settings.

**TESTING :**

1. Functional Testing (Manual Testing)

This type of testing ensures that all features of the dashboard work as expected from a user's perspective.

#### Core Features Testing:

* Login and Authentication:
  + Test if the user can sign up, log in, and log out correctly.
  + Verify if the login error message appears for incorrect credentials.
* Cryptocurrency Price Tracking:
  + Test if real-time prices of cryptocurrencies (e.g., Bitcoin, Ethereum) are displayed correctly.
  + Verify the data is updating periodically or when manually refreshed.
* Portfolio Management:
  + Add a cryptocurrency to the portfolio and verify if it appears with the correct amount.
  + Remove a cryptocurrency from the portfolio and check if the portfolio is updated.
* Watchlist and Alerts:
  + Add a cryptocurrency to the watchlist and verify its presence.
  + Set a price alert and check if it triggers when the price reaches the target.
* User Profile:
  + Test if users can view and update their profile information (e.g., username, email).

### 2. User Interface (UI) Testing

UI testing ensures that the application is visually correct and intuitive to use.

#### Visual Consistency:

* Layout and Design:
  + Check if the elements (buttons, tables, charts) are properly aligned.
  + Ensure that the color scheme, font sizes, and styles are consistent.
* Responsive Design:
  + Verify if the dashboard is responsive across different screen sizes (desktop, tablet, mobile).
  + Ensure elements resize appropriately and the navigation is easy to use on all devices.
* Accessibility:
  + Check if all text has sufficient contrast against the background for readability.
  + Test if the app is navigable using keyboard shortcuts (e.g., tabbing through form fields).
  + Verify that the application works with screen readers.

### 3. Usability Testing

Usability testing focuses on how easy and user-friendly the application is.

#### Navigation and Workflow:

* Ease of Use:
  + Check if users can easily navigate between different sections (Crypto List, Portfolio, Watchlist).
  + Ensure the "Add Coin" and "Remove Coin" functionalities are intuitive.
* Help and Guidance:
  + Verify if there is a help section or tooltips to guide new users.
  + Test if the user can find important information such as "My Portfolio," "Add Coin," or "Set Alert" easily.

### 4. Performance Testing

Performance testing assesses how well the system performs under normal and high loads.

#### Load Time:

* Page Load Time:
  + Measure the time it takes for pages (e.g., cryptocurrency list, portfolio) to load completely. Ensure it's within acceptable limits (e.g., under 3 seconds).
* API Response Time:
  + Manually observe the time it takes for the dashboard to retrieve cryptocurrency data from the API.

#### Stress Testing:

* Handling Multiple Requests:
  + Test the app with multiple users simultaneously accessing the same features (e.g., fetching prices).
* Error Handling:
  + Simulate slow or broken internet connections and check if the application handles errors gracefully (e.g., show a retry button).

### 5. Security Testing

Security testing ensures that the application is safe from vulnerabilities and attacks.

#### Data Protection:

* Data Privacy:
  + Check if sensitive information (e.g., user credentials, portfolio details) is encrypted during transmission.
  + Verify that passwords are stored securely (e.g., hashed in the database).
* Authentication Security:
  + Test the login process with incorrect passwords to ensure proper error messages are shown without revealing too much information.

#### Session Management:

* Session Timeout:
  + Verify that the user is logged out automatically after a period of inactivity.

### 6. Compatibility Testing

This testing ensures that the application works across various browsers, devices, and operating systems.

#### Cross-Browser Testing:

* Verify that the application works as expected on popular browsers (Chrome, Firefox, Safari, Edge).

#### Cross-Device Testing:

* Ensure the app works smoothly on various devices, including desktops, tablets, and smartphones, with different operating systems (Windows, macOS, Android, iOS).

### 7. Regression Testing

Regression testing ensures that new changes or additions do not break existing functionality.

* Testing After Updates:
  + After adding new features or making changes to the codebase, retest all core functionalities (e.g., portfolio management, real-time price updates) to ensure they still work as expected.

### 8. User Acceptance Testing (UAT)

UAT involves testing the application with real users (or stakeholders) to ensure it meets the intended requirements and expectations.

#### Test Scenarios:

* Have real users test key features like portfolio management and setting price alerts.
* Collect feedback on ease of use, any missing features, or bugs they encounter.
* Ask for feedback on the app's design, performance, and any other usability concerns.

### 9. Smoke Testing

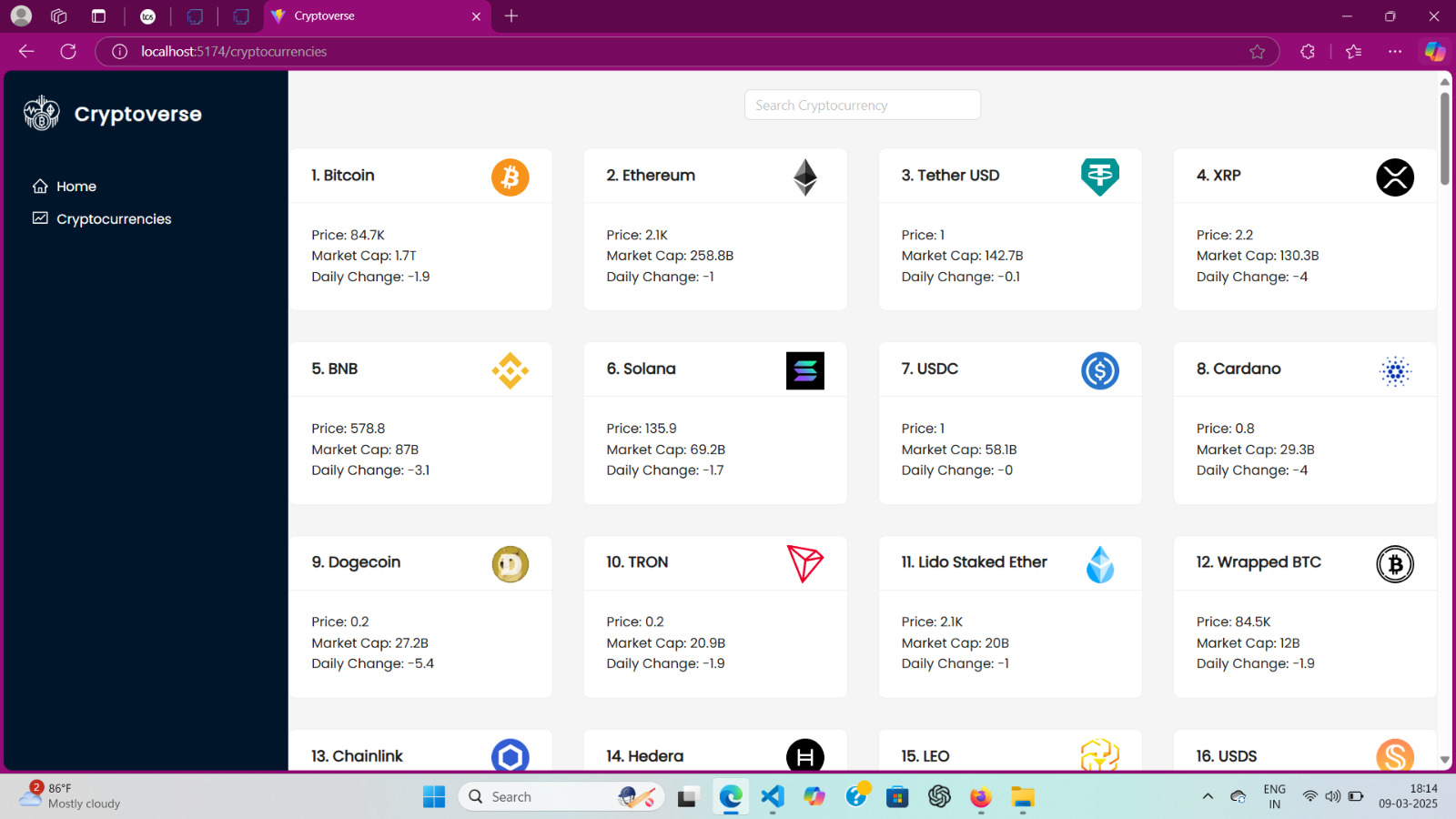
Smoke testing is a quick check to ensure the basic functionality of the application is working before more detailed testing.

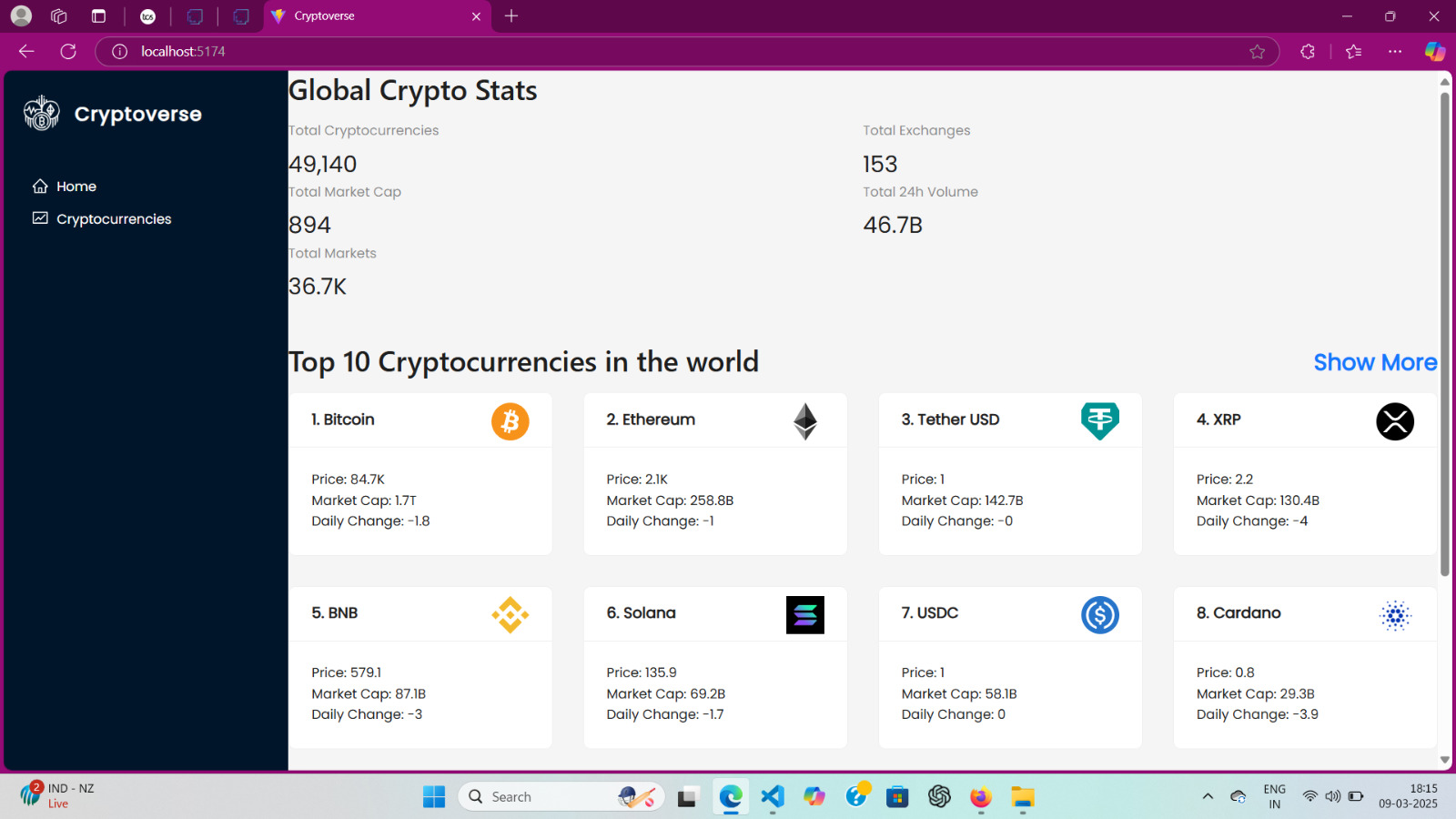
* Quick Check:
  + Ensure that the app loads without crashing.
  + Verify that users can log in, view cryptocurrency prices, and navigate between basic pages.

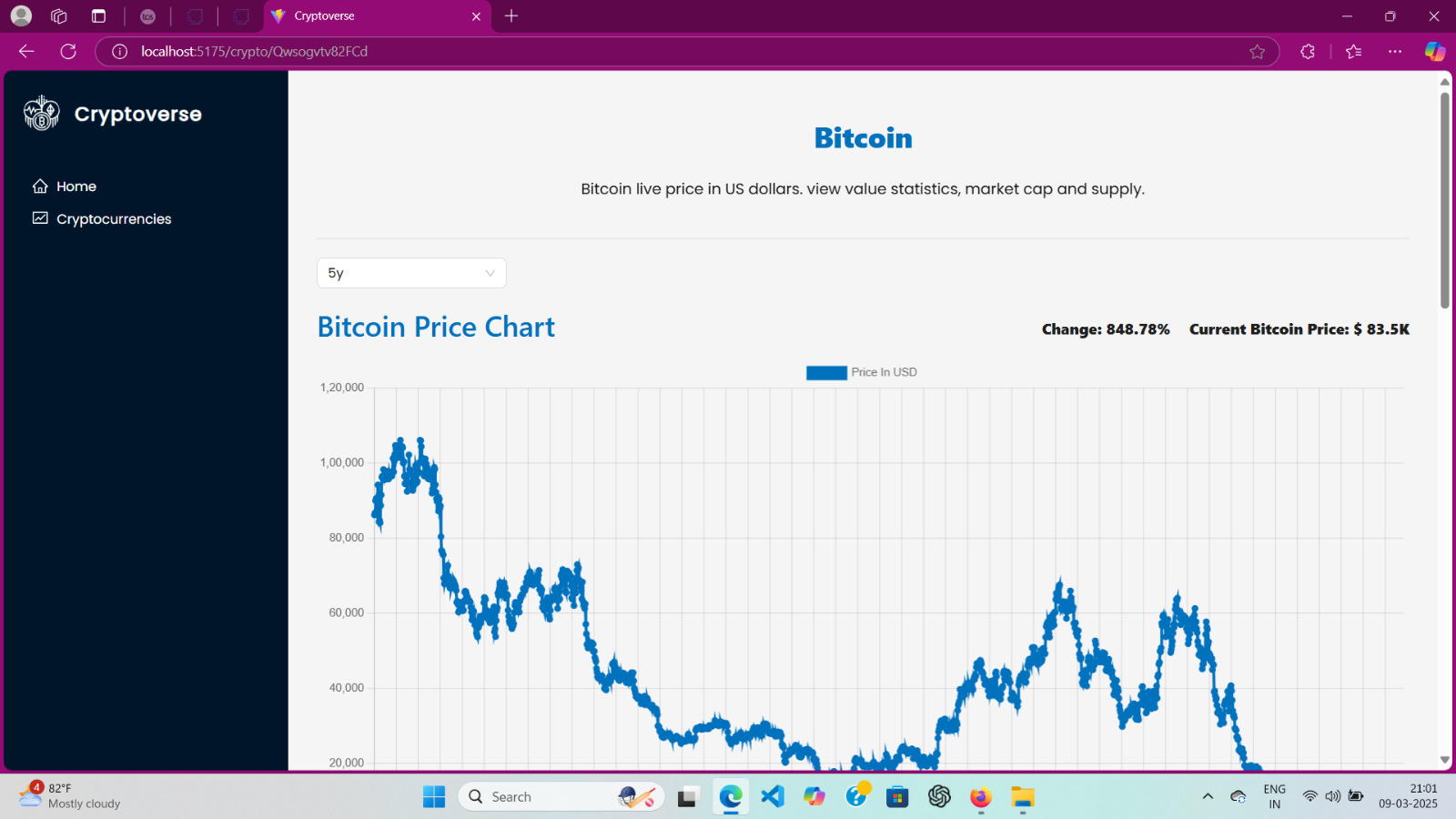
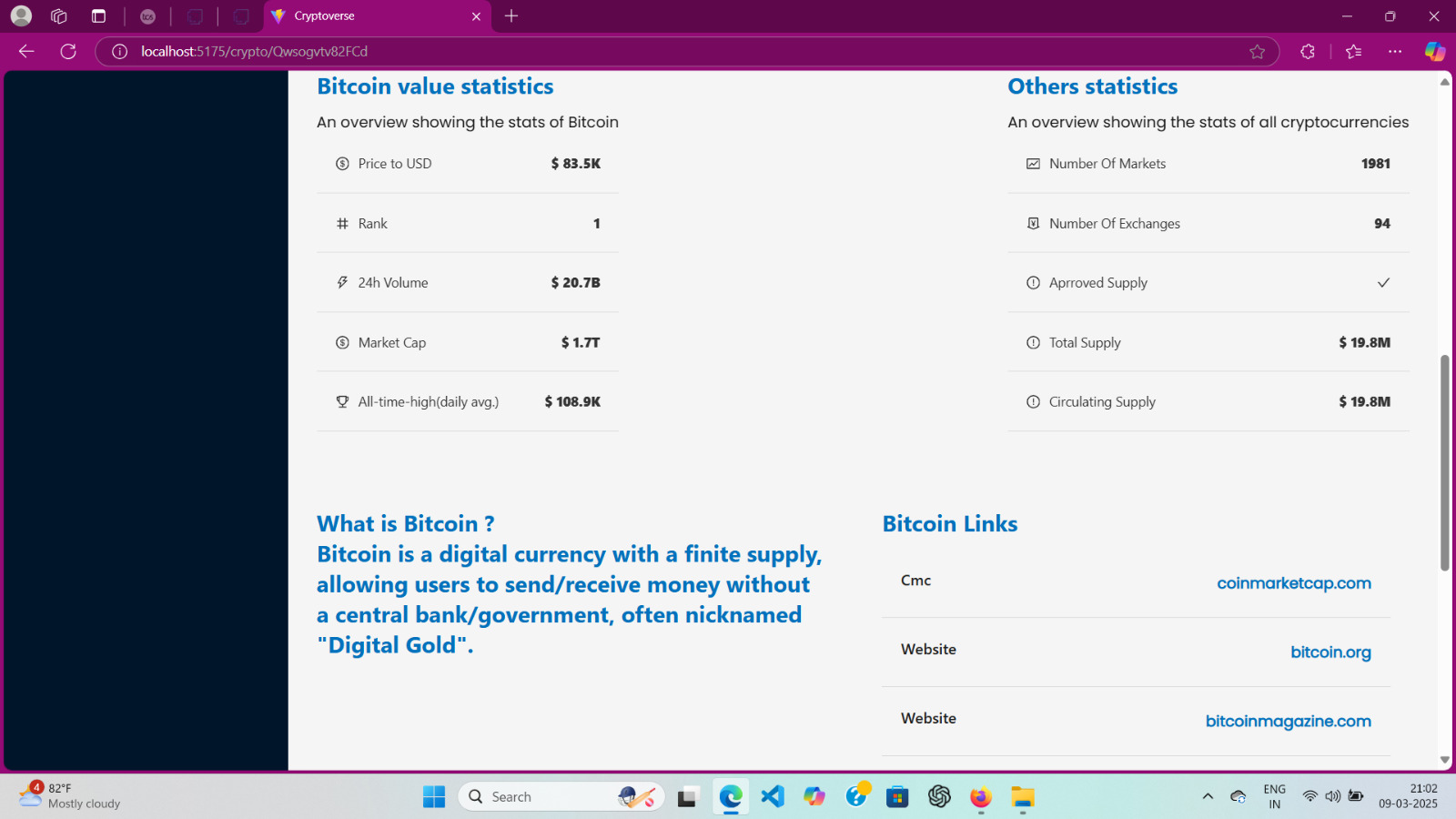
### 10. Beta Testing

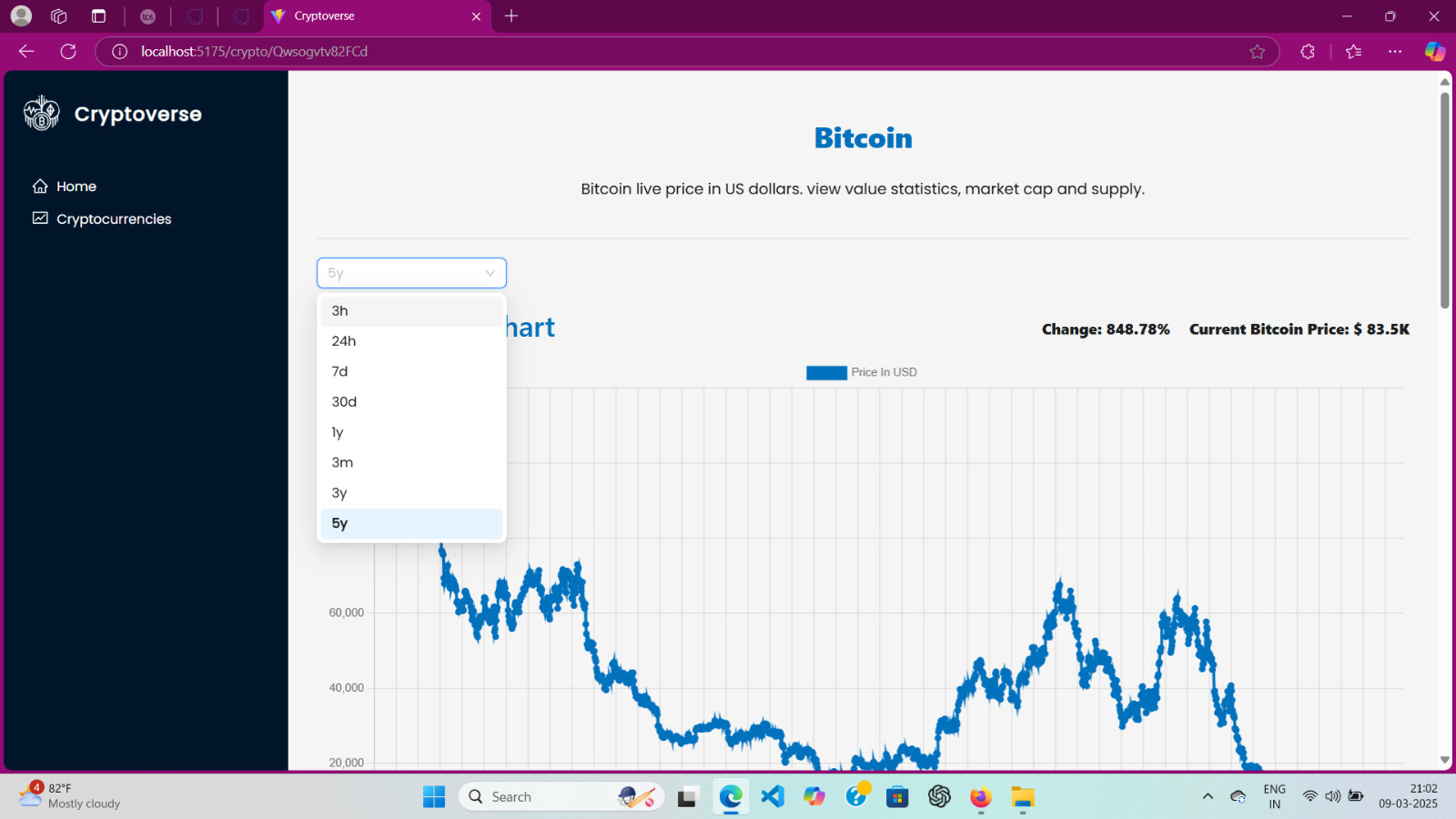
Before releasing the application to the public, beta testing involves inviting a limited group of users to test the app in a real-world scenario.

* Feedback:
  + Gather user feedback on any issues or improvements they suggest.
  + Monitor for any unforeseen bugs or usability issues.

**SCREENSHOTS OR DEMO :**







FUTURE SCOPE :

### 1. Advanced Charting and Technical Analysis

#### Enhanced Visualizations:

* Feature: Integrate advanced charting libraries like TradingView or Chart.js to display real-time price trends, historical data, and performance graphs.
* Benefit: Users can perform technical analysis on cryptocurrencies directly within the dashboard, making it a more comprehensive tool for traders.

#### Indicators:

* Feature: Add popular technical indicators such as RSI, MACD, Bollinger Bands, and Moving Averages.
* Benefit: Advanced users will be able to analyze market trends more effectively and make informed investment decisions.

### 2. Multi-Exchange Support

#### Cross-Exchange Integration:

* Feature: Integrate data from multiple cryptocurrency exchanges like Binance, Coinbase, Kraken, etc.
* Benefit: Users will be able to compare prices across different exchanges and find the best prices for trading.

#### Account Linking:

* Feature: Allow users to link their exchange accounts to track their holdings directly from the dashboard.
* Benefit: This will automate the portfolio management process and give users a centralized view of their cryptocurrency holdings across multiple platforms.

### 3. Mobile App Development

#### Mobile App:

* Feature: Develop a mobile version of the cryptocurrency dashboard for iOS and Android using frameworks like React Native or Flutter.
* Benefit: The app will allow users to track their portfolios, set alerts, and view real-time prices on the go, making it more convenient for users.

#### Push Notifications:

* Feature: Implement push notifications for price alerts and other important updates.
* Benefit: Users can get instant notifications directly on their mobile devices, making it easier to respond to market changes.

### 4. AI and Machine Learning Integration

#### Price Prediction:

* Feature: Integrate machine learning algorithms to provide price predictions and market trend forecasts.
* Benefit: Users could leverage AI to make more informed decisions about when to buy or sell cryptocurrencies.

#### Sentiment Analysis:

* Feature: Use sentiment analysis from social media platforms (like Twitter, Reddit, etc.) to gauge market sentiment around specific coins.
* Benefit: Users can get insights into the overall mood of the cryptocurrency market, potentially helping them make better trading decisions.

### 5. Social Features

#### Community and Social Trading:

* Feature: Introduce a community-based feature where users can share their portfolio, discuss trading strategies, and follow other traders.
* Benefit: Building a social component can enhance user engagement and provide valuable insights through peer learning.

#### Copy Trading:

* Feature: Allow users to copy the trades of experienced traders (also known as social trading or copy trading).
* Benefit: New users can learn from the strategies of more experienced traders and potentially improve their returns.

### 6. Cryptocurrency News and Insights

#### Integrated News Feed:

* Feature: Add a news section that aggregates cryptocurrency-related news from trusted sources such as CoinDesk, CoinTelegraph, and CryptoSlate.
* Benefit: Users can stay informed about market-moving news without leaving the app, helping them make timely decisions.

#### Market Insights and Alerts:

* Feature: Provide market insights based on news and events (e.g., regulatory updates, technological advancements, etc.) that could affect cryptocurrency prices.
* Benefit: Users can gain valuable insights into factors influencing the market, helping them stay ahead of the curve.

### 7. DeFi and Staking Integration

#### Staking Dashboard:

* Feature: Introduce staking options for supported cryptocurrencies. Allow users to stake their coins directly from the dashboard and track rewards.
* Benefit: Users can earn passive income by participating in staking, making the platform a one-stop shop for their cryptocurrency investment needs.

#### DeFi Integration:

* Feature: Allow users to interact with decentralized finance (DeFi) protocols directly from the dashboard for lending, borrowing, and yield farming.
* Benefit: This will enhance the platform’s functionality by providing more opportunities for users to earn returns on their holdings.

### 8. Portfolio Diversification and Risk Management Tools

#### Risk Analytics:

* Feature: Introduce tools that help users manage the risk of their portfolio, such as value-at-risk (VaR) calculations or portfolio diversification analysis.
* Benefit: These tools will help users better understand the risk levels of their investments and optimize their portfolios accordingly.

#### Auto-Diversification:

* Feature: Provide automatic portfolio diversification features, where the system recommends diversifying into different coins or sectors (e.g., DeFi, NFTs, etc.) based on the user’s risk tolerance.
* Benefit: This helps users optimize their portfolios for better risk-adjusted returns.

### 9. Cryptocurrency Tax Calculation

#### Tax Reporting:

* Feature: Integrate tax calculation tools that help users calculate the tax on their cryptocurrency gains or losses based on their local regulations.
* Benefit: This will simplify the tax filing process for users, ensuring they stay compliant with cryptocurrency tax laws.

#### Tax Reports and Export:

* Feature: Allow users to generate and download detailed tax reports, including transaction history, capital gains, and losses.
* Benefit: Tax reporting will become more automated and less time-consuming for users, adding value to the platform.

### 10. Multi-Language and Multi-Currency Support

#### Global User Base:

* Feature: Provide multi-language support for users from different regions.
* Benefit: The dashboard will be accessible to a wider audience, increasing its reach.

#### Currency Conversion:

* Feature: Add support for displaying cryptocurrency prices in different local currencies (USD, EUR, GBP, INR, etc.) or allow users to select their preferred currency.
* Benefit: This feature will make the dashboard more user-friendly for a global audience, improving accessibility and engagement.

### 11. Advanced Security Features

#### Two-Factor Authentication (2FA):

* Feature: Implement 2FA for an extra layer of security during login or when making sensitive changes (e.g., portfolio management, withdrawing funds).
* Benefit: This enhances the security of user accounts, making it harder for unauthorized users to access sensitive information.

#### Biometric Authentication:

* Feature: Support biometric login (e.g., Face ID or Fingerprint Authentication) for mobile apps.
* Benefit: Adds convenience and security for mobile users, making it faster and more secure to access their accounts.

### 12. API for Third-Party Integration

#### Public API:

* Feature: Develop and expose a public API for third-party developers to integrate the cryptocurrency data, portfolio management, or price alerts into their own applications.
* Benefit: This will increase the adoption and usage of the platform, opening up opportunities for collaborations with other financial apps or services.

**CONCLUSION :**

Comprehensive Solution: The Cryptocurrency Dashboard provides users with a centralized platform to track real-time cryptocurrency prices, manage portfolios, set price alerts, and monitor market trends.

* User-Friendly Interface: The dashboard’s design is intuitive, ensuring a seamless user experience. Even beginners can navigate through various features like portfolio management and watchlists without difficulty.
* Real-Time Data: By integrating APIs like CoinGecko and CoinMarketCap, the project ensures that users always have access to up-to-date cryptocurrency prices and market data.
* Custom Alerts: The project includes a feature that allows users to set personalized price alerts, enabling them to make informed investment decisions based on market changes.
* Portfolio Tracking: Users can manage their cryptocurrency holdings, track their performance, and calculate the overall value of their portfolios using real-time market data.
* Security and Authentication: The application uses secure login methods with encryption, ensuring that user data and sensitive information are protected.
* Scalability: The system is designed to handle a growing number of users and can be expanded in the future to integrate new cryptocurrencies and features.
* Performance: The dashboard operates efficiently, with fast load times and smooth navigation, ensuring a positive user experience.
* Future Potential: There are various opportunities for expanding the platform with features like advanced charting tools, multi-exchange integration, mobile app support, and machine learning-based price predictions.
* Educational Value: This project offers insights into building real-time web applications, handling cryptocurrency data, and implementing secure authentication, serving as a valuable learning experience.

REFERENCES :

* CoinMarketCap API Documentation: CoinMarketCap API
* React.js Documentation: React
* Node.js Documentation: Node.js
* Express.js Documentation: [Express.js](https://expressjs.com/)
* Chart.js Documentation: Chart.js
* JWT Authentication Overview: JWT.io
* OWASP Top 10 Security Risks: OWASP Top 10
* TradingView API (Charting): TradingView
* Material-UI Documentation: [Material-UI](https://mui.com/)