TEAM-6
FOREXNOW

Team members

Pragya Mittal
Parth Kumbhar
Piyush Sonigara
Parth Pawar
Navdeep Bhargava

• Approach to the Problem Statement

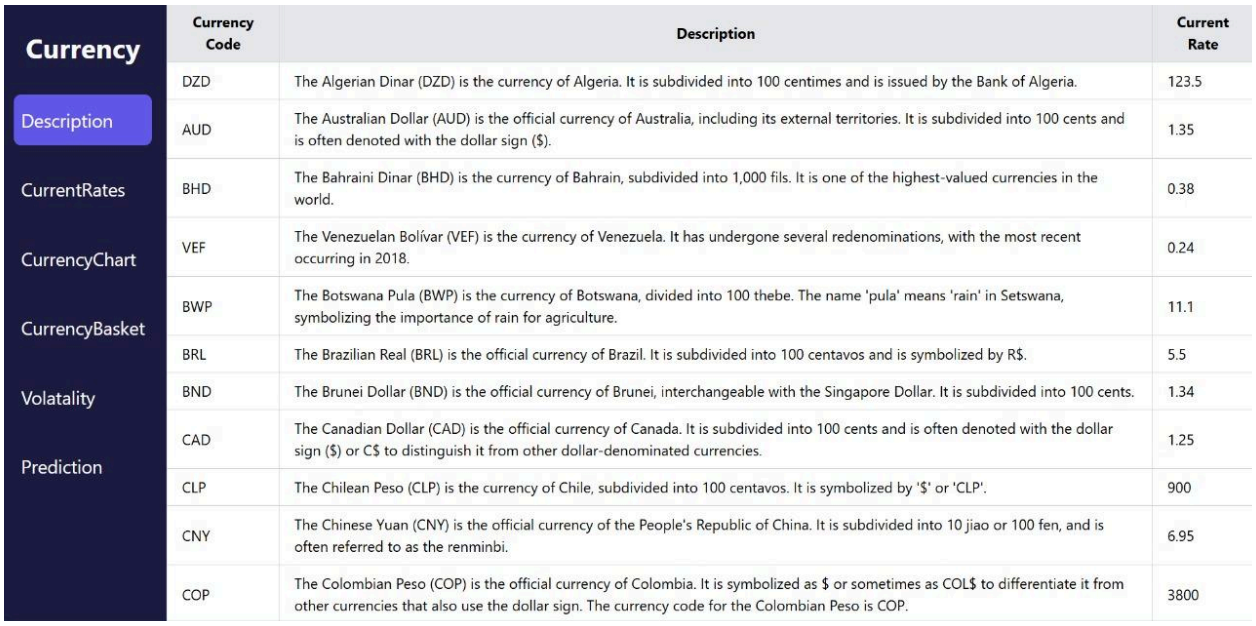
The technical approach to building ForexNow focused on creating a dynamic platform for tracking and analyzing currency exchange rates. The frontend was built using React to provide an interactive user interface, while Flask handled the backend, managing data retrieval and API communication. Historical currency data was stored in SQLite and processed using Pandas to ensure efficient handling of large datasets.

Key features included customizable chart visualizations, currency selection, and period filtering. Python-based machine learning models were integrated to predict future currency trends, with predictions delivered to the frontend via API calls. The modular design ensured scalability, allowing seamless interaction between the frontend and backend for real-time analysis and visualization.

• Introduction to ForexNow

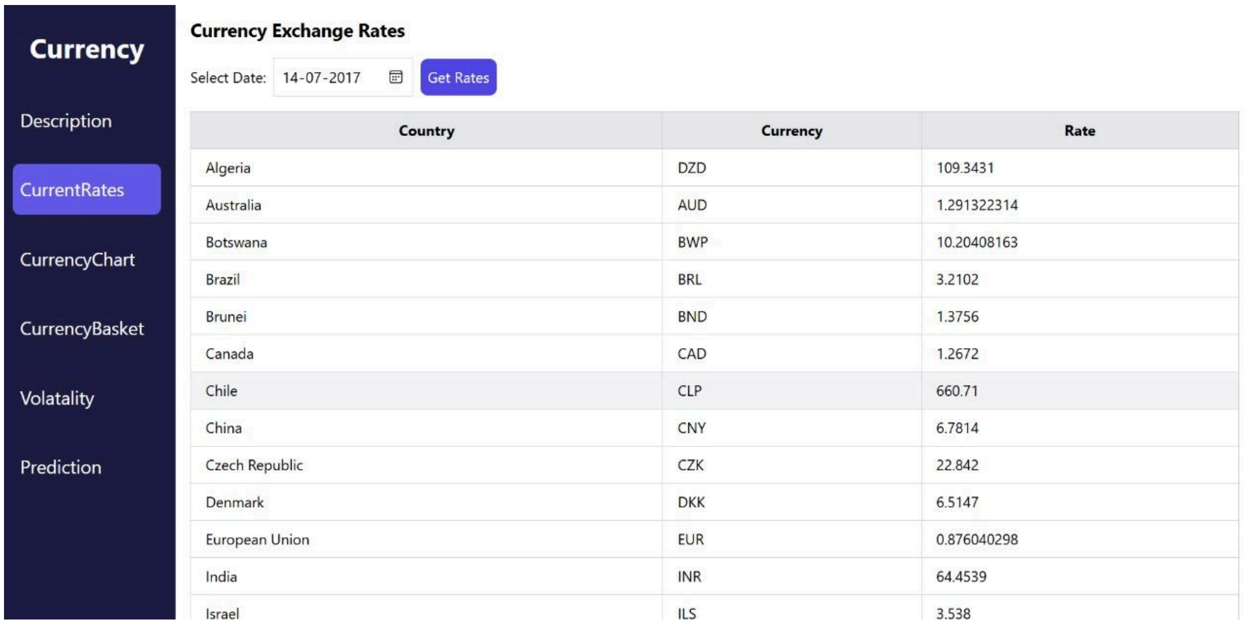
Modules within ForexNow

1. **Description:** Provides users with detailed information about various currencies and their current exchange rates.



Currency Code	Description	Current Rate
DZD	The Algerian Dinar (DZD) is the currency of Algeria. It is subdivided into 100 centimes and is issued by the Bank of Algeria.	123.5
AUD	The Australian Dollar (AUD) is the official currency of Australia, including its external territories. It is subdivided into 100 cents and is often denoted with the dollar sign ($).	1.35
BHD	The Bahraini Dinar (BHD) is the currency of Bahrain, subdivided into 1,000 fils. It is one of the highest-valued currencies in the world.	0.38
VEF	The Venezuelan Bolívar (VEF) is the currency of Venezuela. It has undergone several redenominations, with the most recent occurring in 2018.	0.24
BWP	The Botswana Pula (BWP) is the currency of Botswana, divided into 100 thebe. The name 'pula' means 'rain' in Setswana, symbolizing the importance of rain for agriculture.	11.1
BRL	The Brazilian Real (BRL) is the official currency of Brazil. It is subdivided into 100 centavos and is symbolized by R$.	5.5
BND	The Brunei Dollar (BND) is the official currency of Brunei, interchangeable with the Singapore Dollar. It is subdivided into 100 cents.	1.34
CAD	The Canadian Dollar (CAD) is the official currency of Canada. It is subdivided into 100 cents and is often denoted with the dollar sign ($) or C$ to distinguish it from other dollar-denominated currencies.	1.25
CLP	The Chilean Peso (CLP) is the currency of Chile, subdivided into 100 centavos. It is symbolized by '$' or 'CLP'.	900
CNY	The Chinese Yuan (CNY) is the official currency of the People's Republic of China. It is subdivided into 10 jiao or 100 fen, and is often referred to as the renminbi.	6.95
COP	The Colombian Peso (COP) is the official currency of Colombia. It is symbolized as $ or sometimes as COL$ to differentiate it from other currencies that also use the dollar sign. The currency code for the Colombian Peso is COP.	3800

2. CurrentRates: Users can check the exchange rate of any selected currency on a specified date.

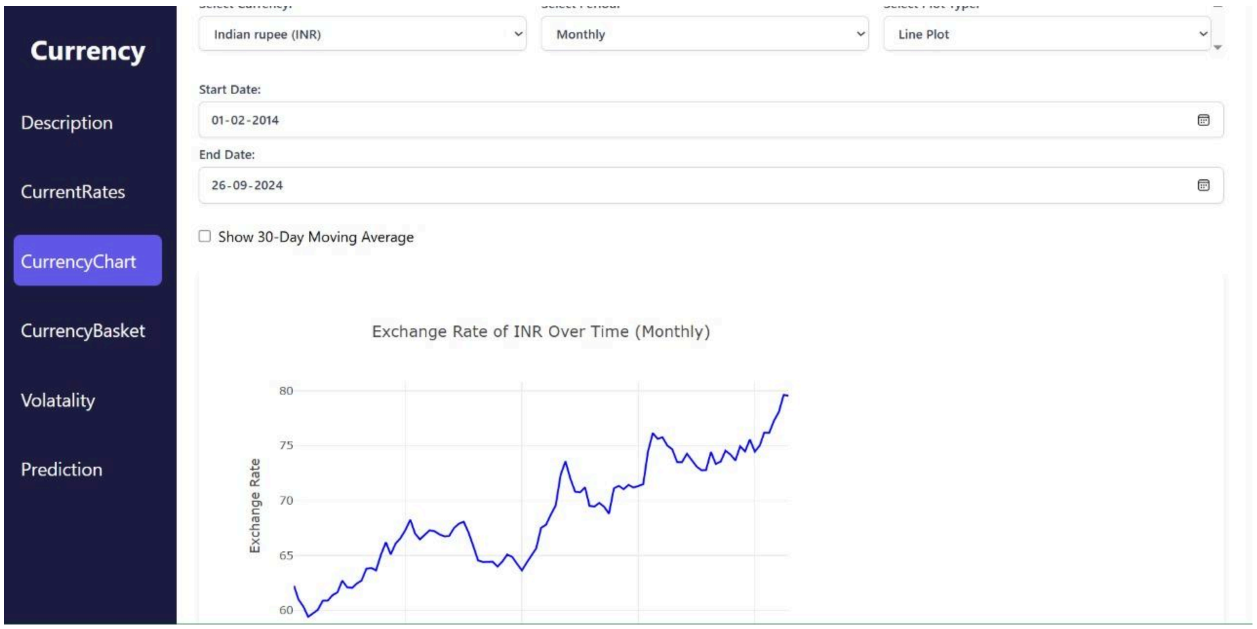


Currency Exchange Rates

Select Date: 14-07-2017 Get Rates

Country	Currency	Rate
Algeria	DZD	109.3431
Australia	AUD	1.291322314
Botswana	BWP	10.20408163
Brazil	BRL	3.2102
Brunei	BND	1.3756
Canada	CAD	1.2672
Chile	CLP	660.71
China	CNY	6.7814
Czech Republic	CZK	22.842
Denmark	DKK	6.5147
European Union	EUR	0.876040298
India	INR	64.4539
Israel	ILS	3.538

3. CurrencyChart: Displays charts showing the progression of exchange rates over different periods (yearly, monthly, quarterly) or between custom start and end dates.



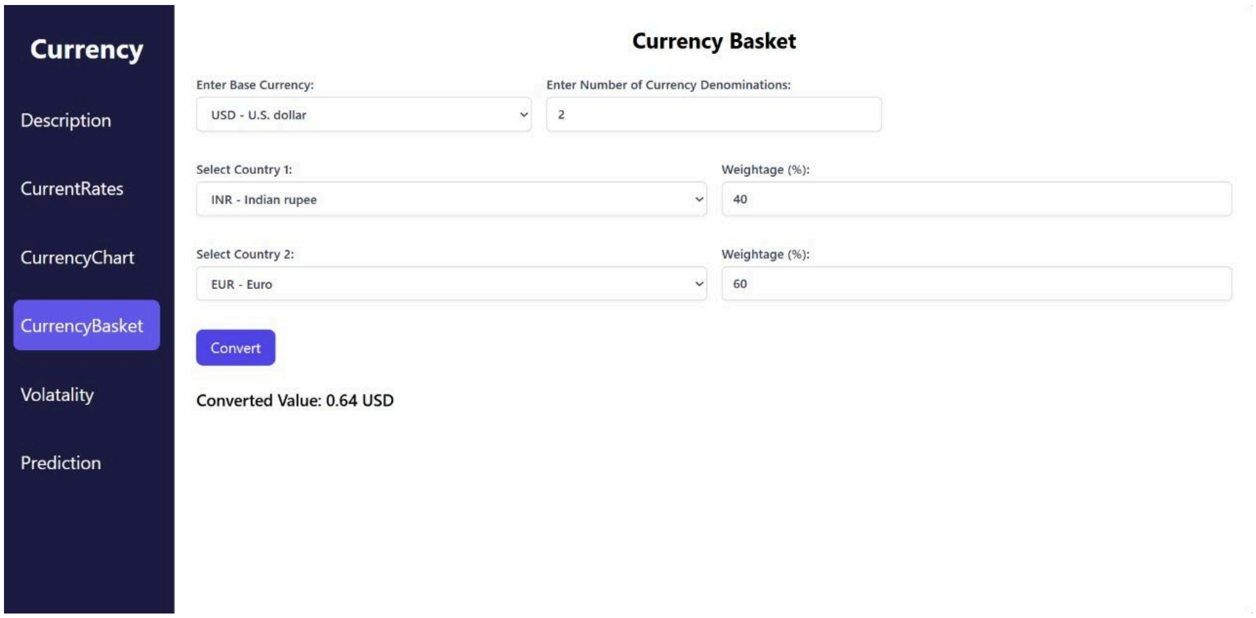
Indian rupee (INR) | Monthly | Line Plot

Start Date: 01-02-2014

End Date: 26-09-2024

☐ Show 30-Day Moving Average

4. Currency Basket: Allows users to calculate the value of a basket of multiple currencies based on their exchange rates.



Currency Basket

Enter Base Currency: USD - U.S. dollar

Enter Number of Currency Denominations: 2

Select Country 1: INR - Indian rupee

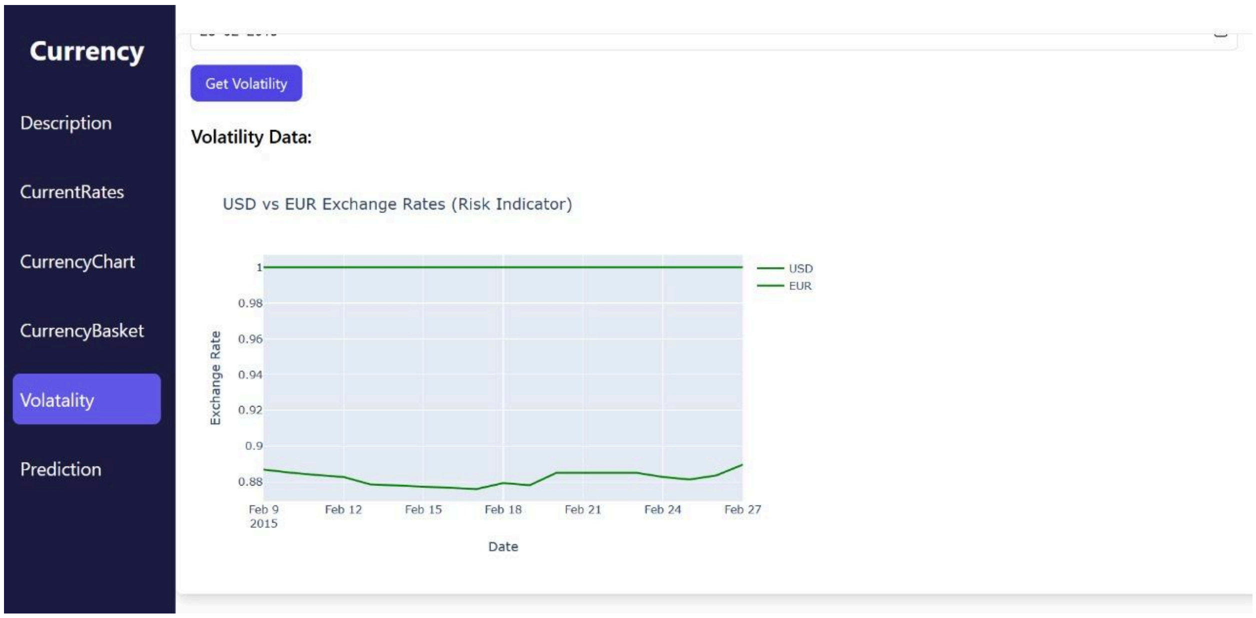
Weightage (%): 40

Select Country 2: EUR - Euro

Weightage (%): 60

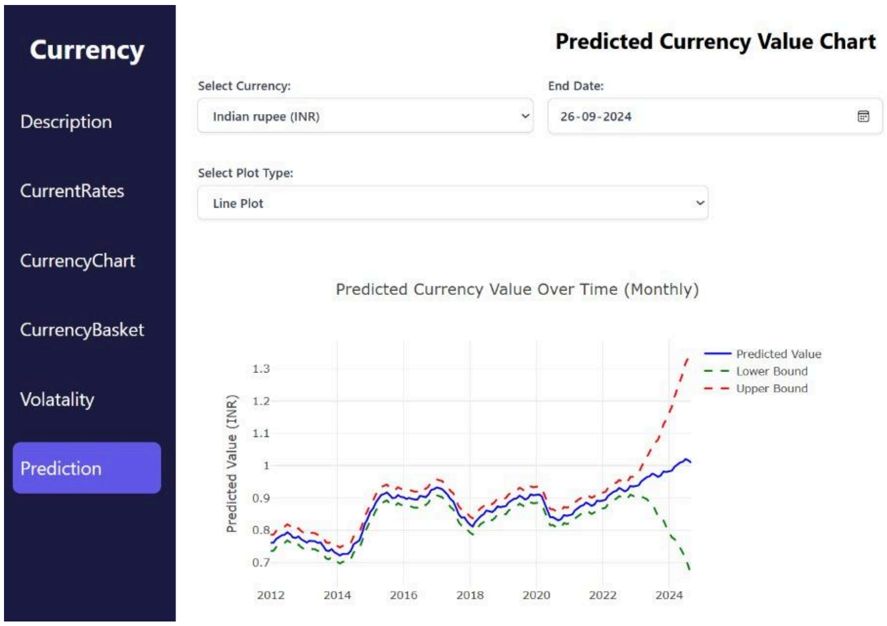
Converted Value: 0.64 USD

5. Volatility: Analyzes exchange rate trends to categorize currencies as highly or low volatile.



Volatility Data:

6. Prediction: Predicts future exchange rates based on previous years' data and displays them on a chart for a user-defined time period.



• Tech Stack Usage

Backend: Flask

The backend is powered by Flask, a lightweight web framework written in Python, which serves data and handles logic for the frontend via RESTful APIs.

1. Flask Framework:
- Flask is used to create the API endpoints that serve data to the frontend. The main entry point of the backend is `app.py`, which defines several routes for different functionalities of the application such as fetching exchange rates, calculating currency volatility, handling currency basket calculations, and predicting future currency trends.
- CORS (Cross-Origin Resource Sharing) is enabled to allow the frontend, hosted on a different domain or port, to interact with the Flask backend. This ensures smooth communication between the frontend React app and the backend APIs.

2. SQLite Database:
- SQLite is used as the database for storing historical exchange rate data. It is accessed using the `sqlite3` module to fetch currency rates and perform calculations based on historical data.
- The database `ntx.db` contains a `currency` table where each currency's exchange rate against other currencies is stored along with the date.
- The backend queries data from SQLite and converts it into Pandas DataFrames for easy manipulation and analysis.

3. Pandas:
- Pandas is used extensively for data manipulation, especially for extracting, filtering, and transforming the currency exchange data retrieved from SQLite.
- It helps manage large datasets, making it easier to calculate changes in exchange rates, filter data by date, and prepare time-series data for analysis.

4. Plotly:
- Plotly is used to generate visualizations of currency exchange rate trends. The Flask app creates line charts that show exchange rate trends, volatility, and risks associated with different currencies. The visualizations are converted to JSON format and sent to the frontend to be rendered interactively.

5. Prophet:
- Prophet is a time-series forecasting library from Facebook. It is used in the `/predict` route to model and predict future currency exchange rates based on historical data.
- The currency rates are predicted for a future date range, and the forecasted data (along with confidence intervals) is sent to the frontend for visualization.

6. APIs:
- Several API endpoints are provided by the Flask backend for different purposes:
  - `/api/exchange-rates`: Retrieves exchange rate data for all currencies.
  - `/api/volatility`: Computes the volatility and risk level of selected currencies over a specified time range.
  - `/api/currency-basket`: Allows the user to calculate the value of a custom currency basket based on selected currencies and weights.
  - `/predict`: Predicts future exchange rates based on historical data using the Prophet library.
  - `/api/current_rates`: Returns the exchange rates for a specific date selected by the user.

Frontend: React

The frontend is built using React, a JavaScript library used for building interactive user interfaces. The frontend makes HTTP requests to the Flask backend to retrieve data and display it dynamically in the UI.

1. React Components:
- The frontend is divided into multiple JSX components, each responsible for a specific module or functionality. These components communicate with the backend by making API calls and rendering the data received.

Some key components are:
- CurrencyChart.jsx:
  - This component fetches currency exchange rate data from the backend `/api/exchange-rates` and displays it as a chart using Plotly.js. Users can select a time range and currencies to visualize their exchange rates over time.
- Volatility.jsx:
  - This component calls the `/api/volatility` endpoint, allowing the user to select two currencies and compare their volatility over a specific time period. The risk levels and volatility are visualized in a chart with risk indicators (colors representing low, medium, or high risk).
- CurrencyBasket.jsx:
  - This component is responsible for handling user inputs for creating a custom basket of currencies. It sends a request to `/api/currency-basket` with selected currencies and their respective weights, and displays the calculated value of the basket.
- Prediction.jsx:
  - This component handles the prediction feature, which calls the `/predict` API. It allows the user to input a currency and an end date to forecast future exchange rates based on historical data. The forecast is visualized in a chart that includes predicted values and confidence intervals.

2. Axios for API Calls:
- Axios, a popular HTTP client for JavaScript, is used in the React components to make asynchronous API calls to the Flask backend. The API responses are then processed and the results are displayed to the user.
- For example, the `useEffect` hook is used in each component to make the API requests when the component loads or when the user inputs change.

3. State Management:
  - React's state management is used to store user-selected currencies, time periods, and the data fetched from the backend. Hooks like `useState` and `useEffect` manage the asynchronous data flow, ensuring the UI updates whenever new data is received from the backend.

4. Plotly.js for Charting:
  - Plotly.js is used on the frontend to render interactive charts. The JSON data received from the Flask backend (generated using Plotly in Python) is parsed and rendered as line charts in the React components. This allows users to view historical trends, volatility, and predicted values in an intuitive, graphical format.

5. CSS for Styling:
  - CSS is used to style the application, ensuring that it provides a clean and responsive user interface. Each component has its own layout, with consistent styling to make the application user-friendly.

Frontend-Backend Communication

- The frontend and backend communicate through HTTP API calls. For instance, when a user selects a date range and currencies in the CurrencyChart component, an API request is sent to the Flask backend, which processes the request, fetches the necessary data from the SQLite database, and returns it to the frontend in JSON format. The frontend then visualizes the data using Plotly.js.
- Similarly, when a user creates a custom currency basket, the CurrencyBasket component sends a POST request with the selected currencies and weights, and the backend responds with the calculated value of the basket.

Tech Stack Summary

- Frontend:
  - React (JavaScript library for building user interfaces)
  - Axios (for API calls)
  - Plotly.js (for rendering charts)
  - CSS (for styling)

- Backend:
  - Flask (Python web framework for building APIs)
  - SQLite (for data storage)

- Pandas (for data manipulation)
- Prophet (for time-series forecasting)
- Plotly (for chart generation)
- Flask-CORS (for handling cross-origin requests)

This tech stack enables a smooth flow of data from the backend to the frontend, providing users with an interactive platform to explore, visualize, and predict currency exchange rates.

• Backend Architecture for modules of ForexNow

1. Description Module
- Route: `/api/exchange-rates`
- Functionality: Fetches currency exchange rates from the SQLite database.
- Backend Code: Queries the SQLite database for historical exchange rates, processes the data using Pandas, and returns it as JSON for the frontend to display currency details.

2. CurrentRates Module
- Route: `/api/current_rates`
- Functionality: Returns exchange rates for a specific date selected by the user.
- Backend Code: Accepts a date as a query parameter, fetches the corresponding exchange rates from the database, and returns them as JSON.

3. CurrencyChart Module
- Route: `/api/exchange-rates`
- Functionality: Displays historical trends of currency exchange rates.
- Backend Code: Fetches time-series data for the selected currencies from the SQLite database, processes it using Pandas, and generates a JSON response. Optionally, Plotly is used on the server-side for visualizations sent to the frontend.

4. CurrencyBasket Module
- Route: `/api/currency-basket`
- Functionality: Calculates the value of a user-defined currency basket.
- Backend Code: Receives selected currencies and weights from the frontend, performs calculations based on current exchange rates stored in the database, and returns the calculated basket value.

5. Volatility Module

- Route: `/api/volatility`
- Functionality: Rates currencies based on volatility over a specified period.
- Backend Code: Fetches exchange rate data for the selected time range, calculates volatility using Pandas, and returns risk ratings (high, medium, low) for each currency.

6. Prediction Module

- Route: `/predict`
- Functionality: Predicts future exchange rates.
- Backend Code: Uses the Prophet library to model historical data and generate future currency exchange rate predictions. The forecast data (including confidence intervals) is returned as JSON for visualization.

Each module is built as a Flask route that processes the request, interacts with the SQLite database, and manipulates the data using Pandas (and Prophet for prediction) before sending JSON responses back to the React frontend.