

# **CRYPTO VOLATILITY AND RISK ANALYSER (CVARA)**

## **TEAM – B**

### **TEAM MEMBERS :**

1. DEVESH GAUTAM
2. BHUMIKA KASTURE
3. SHIRAPTINI T C
4. ATTURU JAGANMOHAN REDDY
5. BRIJENDRA SRIVASTAVA

### **TABLE OF CONTENTS**

- Problem Statement
- Objectives
- Proposed Solution
- Architecture
- Tech Stack Used
- Installation Steps
- How to Run
- Features
- Sample Use-Case
- Future Scope
- Acknowledgements

## 1. PROBLEM STATEMENT :

Cryptocurrency markets are **highly volatile, fast-changing, and unpredictable**, which makes **risk assessment and investment decision-making extremely challenging** for investors, traders, and financial analysts. Rapid price fluctuations, high market uncertainty, and limited transparency often result in **uninformed or high-risk investment decisions**.

Although many existing platforms provide **raw price data**, they lack a **comprehensive and integrated analytical system** that can:

- Quantitatively measure market volatility
- Evaluate risk-adjusted returns
- Visually analyze price behavior
- Classify cryptocurrencies based on risk levels
- Provide controlled and secure user access

Most tools operate in isolation and do not combine **data collection, risk computation, visualization, classification, and secure user management** into a single unified platform.

Additionally, the absence of **user authentication and database integration** limits personalization, data security, and controlled access to analytical features, making such systems unsuitable for real-world deployment.

## OBJECTIVES :

To address these challenges, this project proposes the development of an **end-to-end Crypto Volatility and Risk Analyzer (CVARA)** that:

- Collects **real-time and historical cryptocurrency data**
- Computes **statistical risk metrics** such as volatility, Sharpe ratio, and beta
- Visualizes **price trends and risk-return relationships** using interactive dashboards
- Classifies cryptocurrencies into **High, Medium, and Low risk categories**
- Implements a **secure login system** with **MySQL database integration** for authenticated access

The system provides a **structured, data-driven, secure, and user-friendly solution** that enables reliable cryptocurrency risk analysis and supports informed investment decision-making.

## 2. PROPOSED SOLUTION :

To overcome the challenges of cryptocurrency market volatility and the lack of integrated analytical tools, this project proposes the development of an **end-to-end Crypto Volatility and Risk Analyzer (CVARA)**.

The proposed system is designed as a **secure, data-driven, and interactive web application** that combines data acquisition, risk computation, visualization, classification, and user authentication into a single unified platform.

### Key Components of the Proposed Solution

#### *1 Secure User Authentication*

- A **login and registration system** is implemented to ensure authorized access.
  - User credentials are securely stored and validated using a **MySQL database**.
  - Prevents unauthorized access and enables controlled usage of analytics features.
- 

#### *2 Automated Data Acquisition*

- Fetches **real-time and historical cryptocurrency price data** using APIs such as **CoinGecko / Binance**.
  - Supports multiple cryptocurrencies including Bitcoin, Ethereum, Solana, Cardano, and Dogecoin.
  - Ensures continuous and up-to-date data availability without manual intervention.
- 

#### *3 Risk Metrics Computation*

- Computes essential **statistical risk indicators**, including:
    - Volatility (daily and annualized)
    - Sharpe Ratio (risk-adjusted return)
    - Beta coefficient (market sensitivity)
  - Uses mathematical and statistical methods to ensure accurate risk estimation.
- 

#### *4 Interactive Data Visualization*

- Displays **price trends, volatility trends, and risk-return relationships** using Python visualization libraries such as **Matplotlib and Plotly**.
- Provides dynamic and user-friendly dashboards for easy interpretation.
- Enables multi-crypto comparison and time-range selection.

---

## 5 Risk Classification & Reporting

- Classifies cryptocurrencies into **High, Medium, and Low risk categories** based on volatility distribution.
  - Uses data-driven thresholds to ensure fair and adaptive classification.
  - Generates summary reports and visual insights for decision-making.
- 

## Benefits of the Proposed Solution

- Integrates **analysis, visualization, classification, and security** in one system
- Reduces manual effort and improves decision accuracy
- Enhances usability through interactive dashboards
- Provides a **real-world, scalable, and industry-oriented solution**

## 3. ARCHITECTURE :

### System Architecture – Crypto Volatility and Risk Analyzer (CVARA)

## Architectural Overview

The CVARA system follows a **multi-layered architecture** that integrates **user authentication, data acquisition, processing, visualization, and risk classification** into a unified and secure web-based platform.

---

## Architecture Layers

### 1 Presentation Layer (User Interface)

- Web-based user interface built using **HTML, CSS, JavaScript / Streamlit**.
  - Includes:
    - Login & Registration page
    - Home Page (central navigation hub)
    - Milestone dashboards (1 to 4)
  - Allows users to interact with visual dashboards and analytics features.
- 

### 2 Authentication & Access Control Layer

- Handles **user login, registration, and session management**.

- Validates credentials against a **MySQL database**.
  - Ensures only authenticated users can access analytical features.
- 

### *3 Application Layer (Backend Logic)*

- Developed using **Python (Flask / Streamlit)**.
  - Controls application workflow and milestone navigation.
  - Manages API calls, data processing, and communication between UI and database.
- 

### *4 Data Acquisition Layer*

- Fetches **real-time and historical cryptocurrency data** from:
    - CoinGecko API
    - Binance API
  - Supports multiple cryptocurrencies.
  - Automatically refreshes data at defined intervals.
- 

### *5 Data Processing & Analytics Layer*

- Performs:
    - Data cleaning and preprocessing
    - Log return calculation
    - Volatility, Sharpe ratio, and Beta computation
  - Prepares processed datasets for visualization and classification.
- 

### *6 Visualization & Dashboard Layer*

- Uses **Matplotlib and Plotly** to generate:
    - Price trend graphs
    - Volatility trends
    - Risk-return scatter plots
  - Provides interactive and comparative analytics dashboards.
- 

### *7 Risk Classification & Reporting Layer*

- Classifies assets into **High, Medium, and Low risk categories**.
- Generates:
  - Risk summary tables

- Distribution charts
  - Exportable reports (CSV)
- 

#### *8 Database Layer*

- **MySQL database** stores:
  - User credentials
  - Authentication data
  - (Optional) user activity and preferences
- Ensures data persistence and security.

### **4. TECH STACK USED :**

The **Crypto Volatility and Risk Analyzer (CVARA)** project is developed using a modern and efficient technology stack to support real-time data processing, statistical analysis, and interactive visualization.

#### **Frontend**

- **HTML5** – Structure and layout of web pages
- **CSS3** – Styling, responsive UI, and modern design
- **JavaScript** – Dynamic content updates and user interaction

#### **Backend**

- **Python** – Core programming language for application logic
- **Flask** – Web framework for routing, API handling, and backend integration

#### **Data Analytics & Processing**

- **NumPy** – Mathematical and statistical computations
- **Pandas** – Data cleaning, processing, and transformation

#### **Visualization**

- **Matplotlib** – Static data visualization
- **Plotly** – Interactive and dynamic dashboards

#### **Authentication & Database**

- **MySQL** – Database for storing user credentials and authentication data
- **SQL** – Query language for database operations
- **Flask-MySQL Connector** – Backend database connectivity

## APIs

- **CoinGecko API** – Real-time and historical cryptocurrency price data
- **Binance API** – Market price and volume data

## Risk Metrics & Analytics

- **Volatility Calculation** – Risk measurement using price returns
- **Sharpe Ratio** – Risk-adjusted return evaluation
- **Beta Coefficient** – Market sensitivity analysis

## Development & Tools

- **VS Code** – Development environment
- **Git** – Version control
- **Web Browser** – Application testing and visualization

## 5. INSTALLATION STEPS :

### Installation Steps

#### 1 System Requirements

- Operating System: Windows / Linux / macOS
  - Python Version: **Python 3.8 or above**
  - Internet connection (for live crypto data APIs)
  - Web browser (Chrome / Edge / Firefox)
- 

#### 2 Install Required Software

- Install **Python** from the official website
  - Install **MySQL Server** for database storage
  - Install **MySQL Workbench** (optional, for database management)
- 

#### 3 Project Setup

- Download or clone the project source code
- Navigate to the project directory

```
cd crypto_project
```

---

## 4 Install Python Dependencies

Use `pip` to install required libraries:

```
pip install flask
pip install streamlit
pip install pandas
pip install numpy
pip install requests
pip install plotly
pip install mysql-connector-python
```

---

## 5 Database Configuration

- Create a **MySQL database**
- Create a `users` table for login authentication
- Update database credentials in the backend configuration file

```
CREATE DATABASE crypto_db;
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50),
    password VARCHAR(255)
);
```

---

## 6 Run the Application

- Start the backend Flask server:

```
python app.py
```

- Or run the Streamlit dashboard:

```
streamlit run app.py
```

---

## 7 Access the Application

- Open browser and navigate to:
  - `http://localhost:5000`
  - Login using registered credentials
  - Access Home Page and all project milestones
-



## 8 Successful Deployment

- Login authentication verified
- Live crypto data fetched successfully
- Risk metrics calculated
- Interactive dashboards displayed
- Risk classification completed

## 6. HOW TO RUN ? :

### Step 1: Prerequisites

Ensure the following software is installed on the system:

- Python 3.8 or above
  - MySQL Server
  - MySQL Workbench (optional, for database viewing)
  - Internet connection (for live crypto API data)
- 

### Step 2: Clone / Download Project

Download or clone the project folder to your local system.

Example:

```
crypto_project/
├── app.py
├── utils.py
├── milestone3_streamlit.py
├── templates/
├── static/
├── database.sql
└── requirements.txt
```

---

### Step 3: Install Required Python Libraries

Open terminal or command prompt inside the project folder and run:

```
pip install -r requirements.txt
```

Required libraries include:

- Flask
- Streamlit

- Pandas
  - NumPy
  - Plotly
  - Requests
  - MySQL Connector
- 

## Step 4: Configure MySQL Database

1. Start MySQL Server
2. Create a database (example: `crypto_users`)
3. Create a `users` table to store login credentials
4. Update database connection details in the project code:
  - Host
  - Username
  - Password
  - Database name

This enables **secure login authentication**.

---

## Step 5: Run the Flask Application

To start the main application (Home Page + Milestones 1, 2, 4):

```
python app.py
```

Once running, open browser and go to:

```
http://localhost:5000
```

---

## Step 6: Login to the System

- Enter registered username and password
  - Credentials are validated using **MySQL database**
  - After successful login, user is redirected to **Home Page**
-

## Step 7: Access Milestones

From the Home Page:

- Milestone 1 → Live & Historical Data Fetching
  - Milestone 2 → Risk Metrics Calculation
  - Milestone 3 → Visualization Dashboard (Streamlit)
  - Milestone 4 → Risk Classification & Reporting
- 

## Step 8: Run Milestone 3 (Streamlit Dashboard)

Open a **new terminal window** and run:

```
streamlit run milestone3_streamlit.py
```

Streamlit dashboard opens automatically in the browser and provides:

- Price trends
- Volatility trends
- Risk-return analysis
- KPI metrics

## 7. FEATURES :

### Secure User Authentication

- Login and registration system
  - User credentials stored securely in **MySQL database**
  - Prevents unauthorized access
  - Foundation for future user-specific analytics
- 

### Real-Time & Historical Data Collection

- Fetches live cryptocurrency prices
  - Retrieves historical price data using APIs (CoinGecko / Binance)
  - Supports multiple cryptocurrencies (Bitcoin, Ethereum, Solana, Cardano, Dogecoin)
-

## Risk Metrics Computation

- Calculates **daily and annualized volatility**
  - Computes **Sharpe Ratio** for risk-adjusted returns
  - Computes **Beta coefficient** relative to Bitcoin
  - Uses log-return formulas for accurate financial analysis
- 

## Interactive Data Visualization

- Time-series price charts
  - Volatility trend analysis
  - Risk–return scatter plots
  - Rolling-window volatility visualization
  - Interactive charts using Plotly
- 

## Risk Classification System

- Automatically classifies cryptocurrencies into:
    - **High Risk**
    - **Medium Risk**
    - **Low Risk**
      - Classification based on volatility thresholds
      - Color-coded dashboard for easy interpretation
- 

## Centralized Home Page Navigation

- Single unified interface for all milestones
  - Easy navigation without restarting the application
  - Clear milestone separation for modular analysis
- 

## Analytical Dashboard (Streamlit)

- Responsive dashboard layout
- Filters for cryptocurrency selection and date range
- KPI cards for volatility, returns, and Sharpe ratio
- Suitable for analysts, traders, and researchers

---

## Reporting & Export Capability

- Generates summarized risk reports
  - Supports CSV export of processed data
  - Ready for documentation, academic review, or further analysis
- 

## Modular & Scalable Architecture

- Milestone-based modular development
- Easy to extend with:
  - Additional cryptocurrencies
  - More risk metrics
  - User roles and preferences
    - Industry-oriented design

## 8. SAMPLE USE CASE :

### Use Case 1: Secure User Login & Access Control

**Actor:** Registered User (Investor / Analyst)

**Objective:** Securely access the Crypto Volatility and Risk Analyzer dashboard.

**Description:**

The user accesses the CVARA application through a login page. The system validates the user's credentials using a MySQL database before granting access. Unauthorized users are restricted from accessing analytics features.

**System Features Used:**

- Login authentication
- MySQL database validation
- Secure credential handling

**Outcome:**

Only authenticated users gain access, ensuring data security and controlled usage of the platform.

---

## Use Case 2: Live Cryptocurrency Data Analysis

**Actor:** Crypto Investor

**Objective:** Analyze real-time and historical cryptocurrency prices.

**Description:**

The investor logs into the system and selects a cryptocurrency such as Bitcoin or Ethereum. The system fetches live and historical price data using APIs like CoinGecko/Binance and displays updated market values.

**System Features Used:**

- API-based data fetching
- Live price updates
- Historical data retrieval

**Outcome:**

The investor gains real-time insights into market movements, enabling informed investment decisions.

---

## Use Case 3: Risk Metric Evaluation

**Actor:** Financial Analyst

**Objective:** Evaluate risk metrics for selected cryptocurrencies.

**Description:**

The analyst selects a time period and cryptocurrency. The system calculates volatility, Sharpe ratio, and beta coefficients using statistical formulas and presents them in tabular and graphical form.

**System Features Used:**

- Volatility calculation
- Sharpe ratio computation
- Beta analysis

**Outcome:**

The analyst understands the risk-adjusted performance of cryptocurrencies, supporting accurate financial assessment.

---

## Use Case 4: Risk Classification of Cryptocurrencies

**Actor:** Portfolio Manager

**Objective:** Classify cryptocurrencies based on risk levels.

**Description:**

The portfolio manager uses the risk classification module to view cryptocurrencies grouped into High, Medium, and Low risk categories based on volatility thresholds. The system automatically assigns categories using processed data.

**System Features Used:**

- Risk classification logic
- Color-coded risk categories
- Summary risk dashboard

**Outcome:**

The manager can balance portfolios by selecting assets according to risk tolerance.

---

## Use Case 5: Risk Monitoring for Crypto Platforms

**Actor:** Crypto Analytics Team

**Objective:** Monitor overall market risk levels.

**Description:**

An analytics team uses the risk summary dashboard to monitor overall market volatility and risk distribution across cryptocurrencies. The dashboard provides a consolidated view of market stability. Alerts for sudden changes can be added as a future enhancement.

**System Features Used:**

- Risk distribution charts
- Volatility monitoring
- Dashboard reporting

**Outcome:**

Early identification of market instability, enabling proactive decision-making.

## 9. FUTURE SCOPE :

### Future Scope: Advanced Risk Intelligence & Platform Expansion

This project lays a strong foundation for cryptocurrency risk analysis. In the future, the system can be enhanced in the following ways:

#### 1 Integration of Real-Time Alerts

**Objective:** Notify users about sudden risk changes.

**Description:**

The system can be extended to generate real-time alerts when volatility or risk levels cross predefined thresholds. Notifications can be sent via email or dashboard alerts.

**Enhancements:**

- Volatility threshold alerts
- Risk-level change notifications
- Market crash or spike warnings

**Outcome:**

Users can react quickly to market instability and reduce potential losses.

#### 2 Machine Learning-Based Risk Prediction

**Objective:** Predict future risk trends.

**Description:**

Machine learning models such as LSTM or Random Forest can be integrated to forecast future volatility and risk levels based on historical data patterns.

**Enhancements:**

- Predictive volatility modeling
- Risk trend forecasting
- Smarter investment guidance

**Outcome:**

Moves the system from descriptive analytics to predictive analytics.

#### 3 Portfolio-Level Risk Analysis

**Objective:** Analyze combined risk across multiple assets.

**Description:**

Instead of analyzing single cryptocurrencies, the system can evaluate user-defined portfolios and calculate overall portfolio volatility and risk-adjusted returns.



**Enhancements:**

- Portfolio volatility
- Diversification impact analysis
- Portfolio Sharpe ratio

**Outcome:**

Helps investors build balanced and optimized crypto portfolios.

## **4 Role-Based User Access & Personalization**

**Objective:** Improve security and user experience.

**Description:**

The login and database module can be extended to support multiple user roles such as Admin, Analyst, and Investor, along with personalized dashboards.

**Enhancements:**

- Role-based access control
- User-specific dashboards
- Saved preferences and history

**Outcome:**

Improves scalability and professional deployment readiness.

## **5 Multi-Exchange & Asset Expansion**

**Objective:** Broaden market coverage.

**Description:**

Future versions can integrate data from multiple exchanges (Binance, Coinbase, Kraken) and include additional assets such as NFTs, DeFi tokens, and stablecoins.

**Enhancements:**

- Multi-exchange data aggregation
- Wider crypto coverage
- Comparative exchange analytics

**Outcome:**

Creates a comprehensive crypto risk analytics platform.

## 10.ACKNOWLEDGEMENT :

We express our sincere gratitude to everyone who contributed to the successful completion of the **Crypto Volatility and Risk Analyzer (CVARA)** project.

We would like to thank our **project guide and faculty mentors** for their continuous guidance, valuable suggestions, and encouragement throughout the development of this project. Their support played a vital role in shaping the project's technical and analytical direction.

We extend our heartfelt thanks to our **external mentor** for providing industry-oriented insights and constructive feedback, which helped enhance the practical relevance and real-world applicability of this project.

We are grateful to our **institution and department** for providing the necessary infrastructure, resources, and learning environment required to carry out this project effectively.

We also acknowledge the developers and contributors of various **open-source tools and APIs**, including Python, CoinGecko, Flask, Streamlit, Plotly, and MySQL, which enabled efficient data acquisition, analysis, visualization, and system deployment.

Finally, we would like to thank our **friends and peers** for their cooperation, support, and motivation throughout the project development process.