**A CRYPTOGRAPHY DASHBOARD**

Project Dissertation Submitted in University of Madras in Partial

Fulfilment of requirement for the

Award of Degree of

**BACHELOR OF COMPUTER APPLICATION**

Submitted

By

A.AJAY ANTONY (212206039)

A.BALAJI (212206040)

G.BALAJI (212206041)

B.BARATH (212206042)

DEPARTMENT OF COMPUTER APPLICATION



**AKSHEYAA**

COLLEGE OF ARTS AND SCIENCE

PULUDIVAKKAM, MADURANTAKAM-603 314

DEPARTMENT OF COMPUTER APPLICATION

AKSHEYAA COLLEGE OF ARTS AND SCIENCE

Puludivakkam, Madurantakam-603 314

Assistant Professor

Department of Computer Application

# **BONAFIDE CERTIFICATE**

This is to certify that the project dissertation entitled A CRYPTOGRAOHY DASHBOARD submitted to the University of Madras; Chennai for the Degree of BACHELOR OF COMPUTER APPLICATION is a bonafide record of the research work done by A.AJAY ANTONY (212206039) , A.BALAJI (212206040) , G.BALAJI (212206041) , B.BARATH (212206042) under my guidance. It is also certified that this work has not previously formed the basis for the award to the candidate of any Degree, and Associate ship, Fellowship or any other similar title and that the project works is an independent work on the part of the candidate.

HOD PRINCIPAL

Submitted to Viva-voice Examination Held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Internal Examiner External Examiner

# **DECLARATION**

I here by declare that the internship titled A CRYPTOGRAPHY DASHBOARD is an original work done by us and the project desertion submitted is for the fulfilment for the award of bachelor of computer application from university of madras.

Date :

Place :

A.AJAY ANTONY A.BALAJI

(212206039) (212206040)

G.BALAJI B.BARATH

(212206041) (212206042)

**E MAIL I’D :**

[ajayaantony76464@gmail.com](mailto:ajayaantony76464@gmail.com)

[balaji.ar3pilot55@gmail.com](mailto:balaji.ar3pilot55@gmail.com)

[balaji62626262@gmail.com](mailto:balaji62626262@gmail.com)

[bharathbabu881@gmail.com](mailto:bharathbabu881@gmail.com)

# **ACKNOWLEDGEMENT**

We are extremely grateful to our Honourable Chairman DR. AKILAN RAMNATHAN sir for his encouragement.

We express our sincere thanks to the college and respected Dean.

Dr.M.D. ANTONYARULPRAKASH, Aksheyaa College of Arts & Science,Maduranthagam, for providing all facilities to complete our work on time.

We are extremely thankful to our principal DR.M.MURUGADOSS sir for his care and affection in accomplishing the project.

We express our thanks to Mrs. S. NIRMALA, Head of the Department, Bachelor of Computer Science for his guidance and support during this project work.

We express our deep sense of gratitude to our guide Mrs. R.MONISHA Bachelor of Computer Application, for providing an opportunity to work on this project “A CRYPTOGRAPHY DASHBOARD”.

We also express our thanks to all staff members, Bachelor of Computer Application and my classmates for the support and suggestion during this project work.

We express our sincere thanks to our parents and siblings for their support during this project work.

**ABSTRACT**

**ABSTRACT**

A cryptography dashboard is a centralized tool designed to monitor, analyze, and manage cryptographic operations in real time. As cyber threats continue to evolve, organizations rely on cryptographic techniques such as encryption, hashing, and digital signatures to protect sensitive data. However, managing cryptographic security manually can be complex and error-prone. A cryptography dashboard simplifies this process by providing a visual representation of encryption activities, key management, compliance tracking, and security analytics. It enables organizations to monitor encryption strength, detect vulnerabilities, and ensure proper key lifecycle management, reducing risks associated with weak or compromised cryptographic implementations. Key features of a cryptography dashboard include real-time monitoring of cryptographic operations, automated key rotation tracking, anomaly detection, and regulatory compliance insights. By integrating machine learning and AI-driven analytics, these dashboards can predict potential security threats and enhance proactive defence mechanisms. Additionally, they help organizations maintain adherence to industry standards such as AES, RSA, ECC, and regulatory frameworks like GDPR and NIST guidelines. The implementation of a cryptography dashboard enhances data security, ensures compliance, and minimizes cryptographic failures that could lead to security breaches. In a rapidly digitalizing world, such dashboards are essential for strengthening cyber security strategies and safeguarding critical information.

## **INTRODUCTION**

**INTRODUCTION**

In today's digital era, data security is a critical concern for individuals, organizations, and governments. With the exponential growth of online transactions, cloud computing, and interconnected systems, protecting sensitive information from cyber threats has become a top priority. Cryptography, the practice of securing information through encryption and decryption techniques, plays a crucial role in ensuring data confidentiality, integrity, and authentication. However, managing cryptographic operations effectively requires constant monitoring, timely key management, and compliance with evolving security standards. This necessity has led to the development of cryptography dashboards, which provide a centralized platform for overseeing encryption mechanisms, cryptographic key lifecycles, security protocols, and threat detection.

A cryptography dashboard is an essential cybersecurity tool designed to offer real-time insights into cryptographic processes, helping organizations maintain secure communication, prevent data breaches, and adhere to regulatory requirements. Traditional cryptographic management often involves complex configurations, manual key tracking, and limited visibility into encryption performance. This can lead to security gaps, improper key handling, and increased risk of cryptographic failures. A cryptography dashboard addresses these challenges by automating key management, visualizing encryption performance, and detecting anomalies in cryptographic operations. It provides organizations with a

streamlined approach to monitor encryption algorithms, certificate lifecycles, secure key storage, and compliance adherence, reducing the chances of cryptographic misconfigurations or unauthorized access.

One of the key benefits of a cryptography dashboard is real-time monitoring. It enables security professionals to track cryptographic events such as encryption and decryption activities, key usage, and digital signature verifications. By offering detailed logs and reports, the dashboard helps organizations quickly identify potential weaknesses or breaches in their cryptographic implementations. Additionally, it supports key management, ensuring that cryptographic keys are generated, distributed, and revoked according to security best practices. Poor key management is one of the primary causes of encryption failures, leading to data leaks and unauthorized decryption. A cryptography dashboard mitigates these risks by automating key lifecycle processes, enforcing key rotation policies, and securely storing cryptographic keys.

Another critical aspect of cryptographic security is compliance with industry standards and regulations. Organizations must adhere to security frameworks such as AES (Advanced Encryption Standard), RSA (Rivest-Shamir-Adleman), ECC (Elliptic Curve Cryptography),

GDPR (General Data Protection Regulation), NIST (National Institute of Standards and Technology), and PCI-DSS (Payment Card Industry Data Security Standard). Failure to comply with these standards can result in legal penalties, reputational damage, and financial losses. A cryptography dashboard simplifies compliance management by continuously assessing cryptographic implementations, ensuring they align with regulatory guidelines, and generating compliance reports for audits.

Additionally, threat detection and risk mitigation are essential features of a cryptography dashboard. Cybercriminals constantly attempt to exploit vulnerabilities in encryption methods, such as weak cryptographic algorithms, outdated certificates, or compromised private keys. The dashboard incorporates machine learning and artificial intelligence (AI)-based analytics to identify unusual cryptographic behavior, detect potential security threats, and provide recommendations for strengthening encryption mechanisms. By proactively addressing vulnerabilities, organizations can prevent cryptographic attacks such as man-in-the-middle attacks, brute force decryption attempts, and cryptographic key theft.

Beyond security enhancements, cryptography dashboards improve operational efficiency by reducing the complexity of managing cryptographic assets. Security teams can easily monitor multiple cryptographic implementations across different systems, ensuring consistency and reliability in encryption practices. The dashboard also integrates with enterprise security solutions such as Security Information and Event Management (SIEM) systems, Public Key Infrastructure (PKI) platforms, and Cloud Access Security Brokers (CASB), providing a unified view of cryptographic security across an organization’s IT infrastructure.

A cryptography dashboard is a vital component of modern cybersecurity strategies, offering organizations enhanced visibility, control, and automation over their cryptographic operations. By enabling real-time monitoring, secure key management, compliance tracking, and proactive threat detection, it strengthens encryption security and ensures data protection against evolving cyber threats. As digital transformation accelerates and data security becomes increasingly complex, implementing a cryptography dashboard is no longer optional but a necessity for organizations aiming to safeguard sensitive information and maintain trust in their cybersecurity posture.

## 

## **LITERARTURE REVIEW**

**Literature Review for Cryptography Dashboard Project**

1. Introduction to Cryptography

Cryptography is the practice of securing communication and information through mathematical techniques. It plays a vital role in protecting data confidentiality, integrity, and authentication. Various cryptographic techniques, such as symmetric and asymmetric encryption, hash functions, and digital signatures, are widely used in modern applications.

1. Cryptographic Algorithms and Techniques

Numerous cryptographic algorithms are utilized in secure communication.

Some key algorithms include:

Symmetric Encryption:

AES (Advanced Encryption Standard), DES (Data Encryption Standard), and Blowfish.

Asymmetric Encryption:

RSA (Rivest-Shamir-Adleman), ECC (Elliptic Curve Cryptography), and DiffieHellman.

Hash Functions:

SHA (Secure Hash Algorithm) family, MD5 (Message Digest Algorithm 5), and

BLAKE2.

Digital Signatures:

RSA-based signatures, DSA (Digital Signature Algorithm), and ECDSA (Elliptic Curve Digital Signature Algorithm).

1. Cryptographic Libraries and Tools

Various cryptographic libraries facilitate secure implementation:

OpenSSL – A widely used open-source cryptographic library for encryption and SSL/TLS protocols.

PyCryptodome – A Python library for implementing cryptographic algorithms.

Bouncy Castle – A cryptography API for Java and C#.

Libsodium – A modern, easy-to-use cryptographic library designed for security and performance.

1. Cryptography in Dashboard Applications

* + Real-time encryption and decryption monitoring.

* + Key generation and management interfaces
  + Visualization of cryptographic performance and security metrics.

* + User authentication and access control mechanisms.

1. Security Challenges and Future Trends

Post-Quantum Cryptography – Developing algorithms resistant to quantum attacks.

Zero-Trust Security Models – Enforcing strict authentication and access controls.

Homomorphic Encryption – Enabling computations on encrypted data without decryption.

1. Conclusion

The literature highlights the importance of cryptographic techniques in securing digital assets. The integration of cryptography dashboards enhances security monitoring and management. Ongoing research focuses on improving cryptographic algorithms, security awareness, and usability in real-world applications.

**METHODOLOGY**

METHODOLOGY

Project Planning

1. \*Define Project Scope\*:

Identify the dashboard's purpose, target audience, and required features.

1. \*Conduct Market Research\*:

Analyze existing cryptocurrency dashboards, identifying strengths and weaknesses.

1. \*Establish Goals and Objectives\*:

Determine key performance indicators (KPIs) and metrics to track.

Data Collection and Integration

1. \*Select Data Sources\*:

Choose reliable APIs for cryptocurrency data (e.g., CoinMarketCap, CryptoCompare).

1. \*Design Data Pipeline\*:

Develop a data ingestion process to collect, process, and store data.

1. \*Integrate Data\*:

Combine data from various sources into a unified dataset.

Dashboard Design

1. \*Create Wireframes\*:

Sketch the dashboard's layout, navigation, and key components. 2. \*Design Visualizations\*:

Develop charts, graphs, and other visualizations to display data.

3. \*Choose Colour Scheme and Typography\*: Select a visually appealing and consistent design.

Development

1. \*Select Front-end Framework\*:

Choose a suitable framework (e.g., React, Angular, Vue.js).

1. \*Implement Data Visualization\*:

Use libraries like D3.js, Chart.js, or Highcharts.

1. \*Develop Dashboard Components\*

Build reusable components for data display.

Testing and Quality Assurance

1. \*Conduct Unit Testing\*:

Verify individual components function correctly.

1. \*Perform Integration Testing\*:

Test data flow and visualization rendering.

1. \*Conduct User Acceptance Testing (UAT)\*:

Validate the dashboard meets requirements.

Deployment and Maintenance

1. \*Deploy Dashboard\*:

Host the dashboard on a suitable platform (e.g., AWS, Google Cloud)

1. \*Configure Monitoring\*:

Set up monitoring tools to track performance and issues.

1. \*Plan Updates and Maintenance\*:

Schedule regular updates and maintenance.

**SYSTEM DESIGN**

**SYSTEM DESIGN**

A Cryptocurrency Dashboard system design can be structured as follows:

1. Frontend:
   * React.js or Vue.js for a dynamic and responsive UI.
   * Displays real-time cryptocurrency data (prices, trends, market cap) through interactive charts.
   * User authentication via OAuth (e.g., Google/Facebook) or email-based login.

1. Backend:
   * Node.js/Express or Django for RESTful API services.
   * Data aggregation from third-party APIs like CoinGecko or Binance for real-time cryptocurrency data.
   * WebSocket connection for live updates on cryptocurrency price changes.

1. Database:
   * PostgreSQL for storing user profiles, preferences, and historical transaction data.
   * Redis for caching real-time price data to reduce API calls.

1. Authentication:
   * JWT (JSON Web Tokens) for secure user authentication and session management.

1. API Layer:
   * + REST API for fetching historical data, user portfolio, and preferences.
     + Use GraphQL for querying specific, real-time data requests.
2. Security:
   * HTTPS for secure data transmission.
   * Rate limiting to prevent abuse of third-party APIs.

1. Hosting:
   * Frontend: Deployed on Netlify or Vercel.
   * Backend: Hosted on AWS or DigitalOcean.

1. Monitoring:
   * Use tools like New Relic or Datadog for system performance monitoring.

1. Error Handling:
   * Implement centralized error handling and logging using tools like Sentry or LogRocket.

1. Scalability:
   * Use Docker and Kubernetes for containerization and orchestration to handle high traffic efficiently.

## 

## **IMPLEMENTATION**

**IMPLEMENTATION**

1. Frontend Implementation:

* + UI Design and Setup:

Use React.js or Vue.js for building a dynamic and responsive interface

Create core components like a navbar, sidebar, price charts, portfolio tracker, and user profile.

* + Charts & Data Visualization:

Use Chart.js or Recharts to implement interactive charts displaying cryptocurrency prices, historical data, and trends.

* + State Management:

Use Redux or Context API for managing global states (e.g., user data, market prices).

* + Authentication:

Implement JWT-based authentication for user login and session management.Allow login via OAuth providers (Google/Facebook) or email/password.

1. Backend Implementation:
   * Setup API Server

Use Node.js with Express.js or Django for building a RESTful API to serve cryptocurrency data, user portfolios, and preferences.

* + Real-Time Data

Integrate WebSockets or Server-Sent Events (SSE) for pushing live price updates to the frontend.

* + Database Integration:

Use PostgreSQL for storing user data, transaction history, and portfolio information. Integrate Redis for caching real-time market data to reduce API calls.

* + Authentication:

Implement JWT (JSON Web Tokens) for secure, stateless authentication. Store user passwords securely using bcrypt for hashing.

1. External API Integration:

Integrate CoinGecko API or Binance API to fetch live cryptocurrency prices, market data, historical data, and trends.Fetch user-specified data(such as portfolio details) via the backend API, and sync it with the frontend.

1. Testing:

* + Unit Tests:

Write unit tests for frontend components (React) and backend API routes (Express or Django).

* + API Tests:

Use Postman or automated tests to check API responses.

* + E2E Tests:

Implement Cypress or Selenium for testing user flows like login, portfolio tracking, and real-time updates.

* + Performance Testing:

Use tools like Apache JMeter to stress-test the backend API, especially under high load.

1. Real-Time Updates:

Set up WebSocket connections to push real-time cryptocurrency price changes from the backend to the frontend.

1. Security Measures:

* + HTTPS:

Use SSL/TLS certificates to encrypt data transferred between frontend and backend.

* + Input Validation:

Sanitize inputs and use prepared statements to prevent SQL injection.

* + Rate Limiting:

Implement rate limiting for API calls to prevent abuse, using tools like express-rate-limit.

1. Deployment:

* + Frontend Deployment:

Deploy the React.js app on platforms like Netlify or Vercel for easy CI/CD integration.

* + Backend Deployment:

Host the backend on cloud services like AWS EC2, DigitalOcean, or Heroku.Use Docker to containerize the application for easy deployment and scalability.

* + Database Hosting:

Use Amazon RDS or DigitalOcean Managed Databases for PostgreSQL.Use Redis Labs or AWS Elasticache for caching with Redis.

1. Scalability and Monitoring:
   * Containerization:

Use Docker for containerizing the frontend and backend. Optionally, deploy with Kubernetes for scaling.

* + Monitoring:

Set up New Relic or Datadog for performance and uptime monitoring.

* + Logging:

Implement winston (Node.js) or Loguru (Python) for server-side logging to capture any errors or unusual behavior.

1. User Feedback and Iteration:

* + Analytics:

Integrate tools like Google Analytics to track user behavior and usage patterns.

* + User Feedback:

Implement a feedback system to gather user insights on the dashboard's functionality.

## **TESTING**

**TESTING**

1. Unit Testing:

Test individual components and functions, such as API data processing, currency conversion logic, and chart rendering.Use frameworks like Jest or Mocha for frontend and backend unit tests.

1. API Testing:

Ensure the API endpoints for fetching cryptocurrency data, user portfolio, and transactions return the expected results.Tools like Postman or Jest with supertest can help validate RESTful and GraphQL API responses.

1. Integration Testing:

Test how different components of the system (frontend, backend, and database) work together.

1. End-to-End (E2E) Testing:

Test complete user journeys, like logging in, viewing portfolios, adding new cryptocurrencies, and checking real-time data updates.Use tools like Cypress or Selenium for automated browser testing.

1. Security Testing:

Ensure that user data is encrypted and sensitive information (like passwords) is securely stored.

1. Performance Testing:

Measure the load time of the dashboard, API response time, and the performance of WebSocket connections for real-time updates.Tools like Apache JMeter or Locust can simulate high traffic and load testing.

1. Usability Testing:

Verify that the dashboard is user-friendly, responsive, and accessible.Conduct user tests with different devices and browsers to ensure cross-platform compatibility.

1. Smoke Testing:

Perform a basic check to ensure that the main functionalities (login, real-time data, portfolio view) work after updates or deployments.

1. Regression Testing:

Ensure that new changes (features, bug fixes) do not negatively affect the existing functionality of the platform.

1. Load Testing:

Test how the system handles high traffic, especially during high-volume trading times or major price shifts in the market.

## **RESULT AND DISCUSSION**

**RESULT**

Developing a cryptocurrency dashboard involves creating a user-friendly platform that aggregates real-time data, enabling users to monitor market trends and manage their cryptocurrency portfolios effectively. Key outcomes of such a project include:

1. Real-Time Market Data Integration:

Incorporating live data feeds ensures users have up-to-date information on cryptocurrency prices and market movements.

1. Comprehensive Portfolio Management:

Allowing users to track their cryptocurrency holdings, view historical performance, and analyze portfolio distribution aids in informed decision-making.

1. Advanced Data Visualizations:

Utilizing charts and graphs to represent market trends and individual asset performance enhances data comprehension.

1. Educational Resources:

Providing guides and tutorials within the dashboard can assist users, especially novices, in understanding cryptocurrency markets and investment strategies.

1. Customizable Alerts:

Implementing alert systems for price thresholds or significant market movements helps users stay informed about critical changes.By focusing on these aspects.

**DISCUSSION**

A cryptography dashboard is a powerful tool for monitoring, managing, and analyzing cryptographic processes. It can be used by security professionals, developers, and organizations to ensure encryption, key management, and data security compliance.

### 1. Purpose of the Dashboard

* Real-time Monitoring – Track encryption/decryption processes, key usage, and algorithm performance
* Key Management – Manage cryptographic keys (generation, rotation, expiration).
* Security Compliance – Ensure adherence to standards (AES, RSA, ECC, etc.) and industry regulations.
* Threat Detection – Identify anomalies, brute-force attempts, or expired certificates.

### 2. Essential Features

* Encryption & Decryption Statistics – Visualize usage patterns of different cryptographic algorithms.
* Key Lifecycle Management – Track key creation, expiration, revocation, and renewal.
* Algorithm Performance Metrics – Measure speed, efficiency, and load of encryption mechanisms.
* Certificate Management – Monitor SSL/TLS certificates for expiration and vulnerabilities.
* Logs & Auditing – Maintain logs of encryption events for security analysis and audits.
* User Access Controls – Define who can view and manage cryptographic elements.

### 3. UI/UX Considerations

* Intuitive Interface :

Use dashboards with clear charts, graphs, and alerts.

* Dark Mode Option :

Preferred for security professionals working long hours.

* Customizable Views :

Allow users to filter by encryption algorithms, key types, or usage.

* Real-time Alerts :

Notify users of security breaches, expired keys, or algorithm weaknesses.

### 4. Integration with Other Systems

* SIEM (Security Information and Event Management) Integrate with platforms like Splunk or IBM QRadar.
* Cloud & On-Prem Support

Ensure compatibility with AWS KMS, Azure Key Vault, and on-prem HSMs.

* API Support

Enable developers to automate cryptographic functions.

### 5. Security Challenges & Considerations

* Access Control –

Implement role-based access controls (RBAC) to restrict unauthorized access.

* Data Encryption :

Ensure stored data and logs are also encrypted.

* Zero Trust Approach :

Continuously validate users and systems interacting with cryptographic tools.

## **CONCLUSION**

**CONCLUSION**

Developing a cryptocurrency dashboard is essential for navigating the dynamic digital currency market. Such dashboards provide real-time data, enabling users to monitor market trends and make informed investment decisions. A well-designed dashboard enhances user experience by presenting complex data in an accessible format. Integrating educational components within the dashboard can further benefit users, especially those new to cryptocurrency. The "Crypto Educational Dashboard" project aims to assist beginners in entering the cryptocurrency space safely, promoting positive user experiences. However, it's crucial to acknowledge the risks associated with cryptocurrency investments, including fraud and high volatility. Investors should exercise caution and conduct thorough research before engaging with cryptocurrency markets.

## **FUTURE ENHANCEMENT**

**FUTURE ENHANCEMENT**

1. Security Enhancements

Post-Quantum Cryptography: Implement algorithms resistant to quantum attacks .

* + Zero-Knowledge Proofs (ZKPs):

Add support for ZKPs to verify data without revealing it.

* + Multi-Factor Authentication (MFA):

Strengthen access control using biometrics or hardware keys.

* + End-to-End Encryption (E2EE):

Ensure data remains encrypted from sender to recipient.

1. Performance & Scalability

* + GPU/TPU Acceleration:

Optimize cryptographic operations for faster processing.

* + Cloud & Edge Computing Support:

Distribute encryption/decryption workloads across cloud or edge nodes.

* + Blockchain Integration:

Secure data integrity and audit logs using blockchain.

* + Efficient Key Management:

Improve key storage with Hardware Security Modules (HSM) or Secure Enclaves.

1. Usability & UX Improvements

* + Intuitive UI/UX:

Create a more user-friendly and visually appealing dashboard.

* + Dark Mode & Accessibility Features:

Improve readability and support users with disabilities.

* + Drag & Drop Encryption Tools:

Simplify file encryption/decryption for non-technical users.

1. Support for Advanced Cryptographic Techniques

* + Homomorphic Encryption:

Allow computation on encrypted data without decryption.

* + Attribute-Based Encryption (ABE):

Implement role-based access control via encryption.

* + Secure Multi-Party Computation (SMPC):

Enable collaborative data analysis without sharing raw data.

* + Hybrid Encryption:

Combine symmetric and asymmetric encryption for better performance and security.

1. API & Interoperability Enhancements

* + REST & GraphQL APIs:

Allow external applications to integrate encryption services.

* + SDKs for Developers:

Provide SDKs for different programming languages (Python, Java, etc.).

1. Compliance & Governance

* + Regulatory Compliance:

Ensure adherence to GDPR, HIPAA, PCI-DSS, and NIST standards.

* + Automated Audit Logs & Reports:

Generate compliance reports for security audits.

* + Data Residency Control:

Allow users to choose where encrypted data is stored.

## **REFERENCES**

**REFERENCES**

1. Research Papers & Articles
   * "A Survey on Cryptography Techniques" –

Covers different cryptographic algorithms and their use cases. [DOI:

10.1109/ACCESS.2021.3051984]

* + "Post-Quantum Cryptography:

Current State and Future Directions" – Discusses quantum-safe cryptography. [Available on IEEE Xplore]

* + "Cryptography and Network Security Principles and Practices" by William Stallings –

A comprehensive book on cryptography concepts.

* + "Modern Cryptography:

Theory and Practice" by Wenbo Mao – Covers cryptographic protocols and real-world applications.

1. Open-Source Cryptography Dashboard Projects (GitHub)
   * CipherSweet Dashboard :

A PHP-based cryptographic dashboard for managing encryption.

GitHub: https://github.com/paragonie/ciphersweet

Rage (Rust Age Encryption) : A modern encryption dashboard using Rust.

GitHub: https://github.com/str4d/rage

Cryptii :Interactive cryptography tool for encryption/decryption.

GitHub: https://github.com/cryptii/cryptii

1. Tools & Libraries for Cryptography Dashboards CryptoJS – JavaScript library for hashing and encryption.

https://github.com/brix/crypto-js

PyCryptodome – Python library for cryptographic operations.

https://github.com/Legrandin/pycryptodome

OpenSSL – Command-line and library tool for encryption. https://www.openssl.org/

BCrypt – Secure password hashing library. https://github.com/kelektiv/node.bcrypt.js

1. Dashboard Development Frameworks

Dash (Python) + Plotly – Used to create interactive cryptography dashboards.

https://dash.plotly.com/

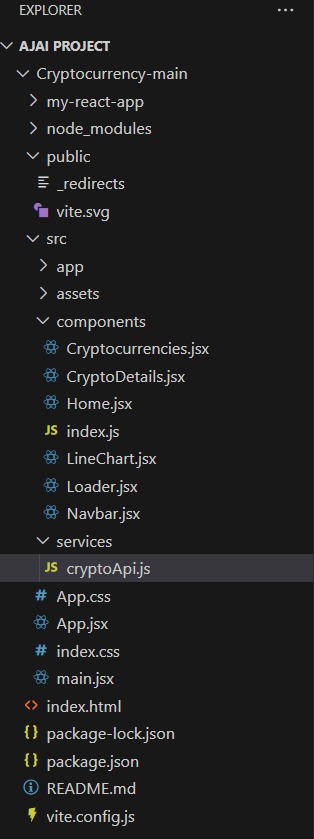
React + Chart.js – For building web-based cryptographic dashboards.

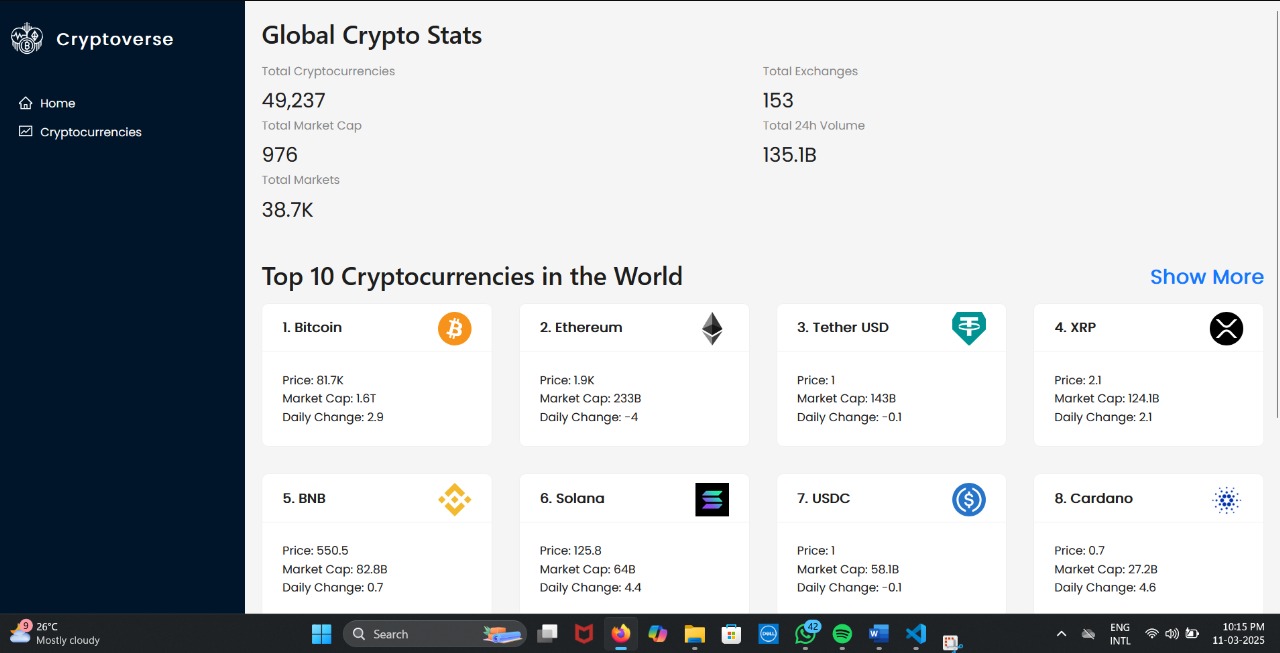
https://www.chartjs.org/

Grafana – For visualizing encryption statistics.

https://grafana.com/

**PROJECT STRUCTURE :**

****

**OUTPUT :**