Boray's Wallet: Technical Report

Group-T

Contents

Executive Summary

This report outlines the conception, development, and deployment of Borays—a secure, web-based blockchain wallet that operates using a dual-device system for digital asset signing. Created as a final-year capstone project at the University of Wollongong, Borays was developed in direct response to user-centric requirements focusing on security, usability, and shared trust through multi-device interaction.

Borays stands out by implementing a novel approach to private key management. The private key is never fully stored on a single device; instead, it is split and distributed using homomorphic Paillier encryption across two devices. Additionally, the application supports secure transaction signing through a two-party ECDSA mechanism. Real-time communication and state synchronization between the devices are enabled using Firebase, ensuring a seamless and secure user experience.

This document explores various development phases, starting from initial user requirement gathering through to implementation and validation. It details the use of web technologies including Flutter Web for frontend development and Firebase Firestore for backend management. Each development sprint was structured around Agile methodologies, allowing for rapid prototyping, continuous feedback, and incremental feature delivery.

The team encountered and overcame several critical challenges throughout development. These included interpreting abstract security requirements into usable features, securely handling mnemonic phrase recovery, encrypting key fragments between devices, building a responsive UI for web access, and ensuring a robust Firebase integration capable of handling real-time events. Comprehensive testing and feedback loops helped validate each feature and ensured the final product met expectations.

Borays now supports secure wallet generation, transaction co-signing via two trusted devices, and encrypted local and remote storage of sensitive data. The application is live-test ready, and all relevant system documentation, including design rationale and installation guidelines, are available to support future maintenance and possible upgrades.

Introduction

Purpose

This technical report presents the design journey and implementation of Borays, a secure, dual-device cryptographic wallet developed to enhance blockchain transaction safety. The core purpose of this document is to provide a comprehensive summary of the planning, development strategies, technological choices, and engineering methodologies used to realize a secure two-party digital signature system.

The report outlines the different development phases and highlights the cryptographic principles underpinning the application. It also discusses the architectural choices that shape the frontend and backend of the system and offers insight into the rationale behind creating a web-first wallet experience. Special focus is given to the two-party ECDSA model and how it was practically realized using accessible web technologies to meet user expectations for both security and usability.

Background

This project was undertaken as part of the CSIT321 capstone subject at the University of Wollongong. Its objective was to address some of the core vulnerabilities found in traditional single-device cryptocurrency wallets. To enhance wallet security and mitigate single points of failure, Borays incorporates both Paillier homomorphic encryption and a dual-device structure using the Elliptic Curve Digital Signature Algorithm (ECDSA).

Our exploration began with a review of common threat models in existing wallets, which informed the design of our core security features—distributed key custody, encrypted communication, and recovery through BIP-39 mnemonic standards. Based on user and stakeholder insights, we created a prototype using Flutter and Firebase that facilitated cryptographic coordination across two devices.

Borays was ultimately developed to meet the growing demand for secure, accessible, and decentralized wallet solutions in the blockchain space. By following secure development life cycle (SDLC) principles, the team ensured that the final product balanced cryptographic integrity with intuitive usability.

https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm

https://en.wikipedia.org/wiki/Paillier_cryptosystem

https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki

System Requirements

Functional and Non-Functional Requirements with Examples

Type	Requirement	Example
Functional	Support creation of wallets using 24-word mnemonic phrases	System must allow secure mnemonic generation to ensure future wallet recovery in case of data loss.
Functional	Split mnemonic between Device A (1–12) and Device B (13–24)	Enables distributed key management by dividing the secret between two physical endpoints.
Functional	Co-sign transactions from two devices	Transactions are validated only when both device shares contribute to the digital signature.
Functional	Secure device pairing using unique code	Time-sensitive pairing codes ensure that only trusted devices are authorized during setup.
Functional	Transactions must only complete after dual approval	Prevents single-point compromise by requiring both devices to confirm any blockchain transaction.

Non-Functional	Secure communication using Paillier encryption	Homomorphic encryption ensures no key fragments are exposed during transmission.
Non-Functional	Web-based and accessible through supported browsers	Application should be usable on latest Chrome/Firefox with no mobile installation requirement.
Non-Functional	Secure storage of user data using Firebase	Firebase Firestore securely stores wallet metadata and authorization states.
Non-Functional	Enforce strong password policy	Passwords must meet strength criteria to avoid brute-force or dictionary attacks.
Non-Functional	Provide real-time visual feedback and transaction status	UI must reflect transaction lifecycle to ensure transparency for the user.

Scope and Iteration Management

The development of Borays followed an Agile workflow that emphasized adaptability and iterative improvement. Weekly stand-up meetings and sprint reviews allowed the team to adapt quickly to evolving requirements, incorporate feedback, and ensure that key milestones were met on time.

- **Sprint 1** focused on laying the foundation, with the creation of a basic UI/UX prototype, initial Firebase setup, and integration of authentication and Firestore databases. This sprint provided a working frontend that was visually aligned with project goals and allowed user interaction.
- **Sprint 2** introduced wallet creation functionality using 24-word mnemonics, aligned with the BIP-39 standard. In this phase, device-side password storage and validation mechanisms were also developed, reinforcing the system's security model.
- **Sprint 3** tackled the implementation of device pairing. This included the generation and verification of one-time pairing codes and the syncing of device data via Firebase. Real-time updates and transaction status monitoring were tested extensively to improve reliability.
- **Sprint 4** marked the introduction of cryptographic modules. Key highlights included the integration of Paillier encryption to facilitate secure key exchanges and the ECDSA digital signature scheme for transaction validation. Complex edge cases and failure conditions were identified and addressed.

The final iteration was dedicated to refinement. The team optimized UI components, patched vulnerabilities, and ensured system stability through testing. Password policies were enforced, data encryption was verified, and the app was prepared for demonstration.

Web-Based Platform Implementation Requirements

To ensure that *Borays* functions reliably as an internet-based application, the system was designed exclusively for browser access, requiring active connectivity at all times.

Requirement Category	Description
Web-Based Access	The application is built using Flutter Web and requires an active internet connection.
No Offline Support	Offline capabilities are not provided due to reliance on Firebase-based real-time operations.
Browser Compatibility	The app is optimized for modern browsers like Chrome, Firefox, and Edge.
Responsive UI	Layouts auto-adjust to screen sizes for varied resolutions on laptops and desktops.
Visual Feedback	Users receive instant confirmation for transactions and status changes.
Secure Cloud Backend	Firebase Authentication and Firestore manage secure user sessions and data exchange.

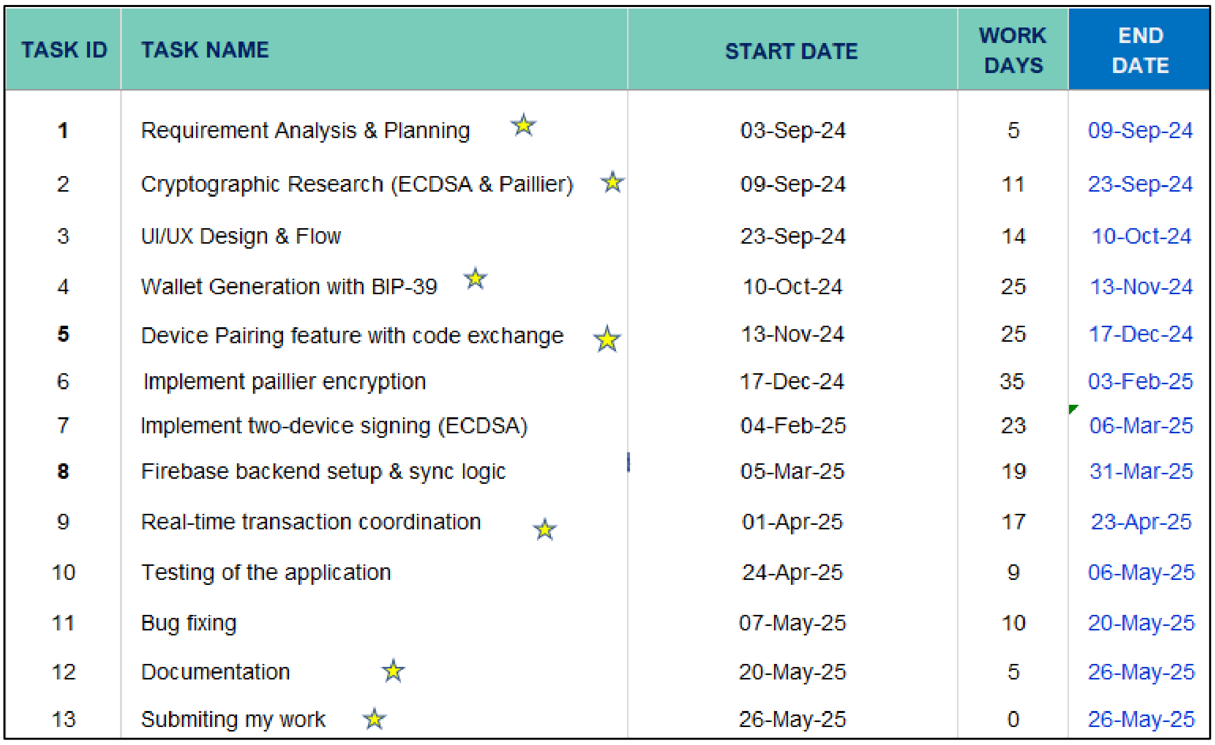
This implementation ensures real-time synchronization between Device A and Device B for all wallet operations, maintaining robust cryptographic security while leveraging web technology for accessibility and scalability.

Project Summary

The development of Borays was guided by a well-organized, Agile-inspired approach, broken down into five structured iterations. Each stage focused on building specific features that collectively shaped the final product. Starting with the foundation—user interface design and secure mnemonic generation—the team steadily moved toward more complex elements such as dual-device transaction signing and real-time Firebase integration. Progress was consistently tracked through team collaboration, feedback cycles, and iterative improvements, ensuring the wallet evolved in line with both technical goals and user needs.

Iteration Timeline

The Borays project progressed through a series of targeted development phases. The initial iterations were dedicated to planning, system design, and prototyping. As the project matured, the focus shifted to implementing essential features like secure wallet creation, device pairing, and encrypted communication for co-signing transactions. The later stages centered on integration testing, bug resolution, optimization, and preparing documentation and presentation material. Each iteration played a crucial role in refining the system's reliability, security, and usability.



TASK ID	TASK NAME	START DATE	WORK DAYS	END DATE
1	Requirement Analysis & Planning ☆	03-Sep-24	5	09-Sep-24
2	Cryptographic Research (ECDSA & Paillier) ☆	09-Sep-24	11	23-Sep-24
3	UI/UX Design & Flow	23-Sep-24	14	10-Oct-24
4	Wallet Generation with BIP-39 ☆	10-Oct-24	25	13-Nov-24
5	Device Pairing feature with code exchange ☆	13-Nov-24	25	17-Dec-24
6	Implement paillier encryption	17-Dec-24	35	03-Feb-25
7	Implement two-device signing (ECDSA)	04-Feb-25	23	06-Mar-25
8	Firebase backend setup & sync logic	05-Mar-25	19	31-Mar-25
9	Real-time transaction coordination ☆	01-Apr-25	17	23-Apr-25
10	Testing of the application	24-Apr-25	9	06-May-25
11	Bug fixing	07-May-25	10	20-May-25
12	Documentation ☆	20-May-25	5	26-May-25
13	Submiting my work ☆	26-May-25	0	26-May-25

Achievements

- **Secure Key Exchanges Using Paillier Encryption**
  Borays employs Paillier homomorphic encryption to safeguard the transmission of private key fragments between devices. This ensures that even if communication is intercepted, no critical cryptographic data can be reconstructed, preserving the confidentiality of wallet keys.
- **Enforced Dual-Device Signing Protocol**
  A major success of the project was implementing a dual-device approval system for all transactions. By splitting key ownership across two devices, the wallet reduces the risk of single-point compromise and mandates collaborative validation for every asset transfer.
- **Real-Time Pairing and Storage via Firebase**
  Firebase Firestore was used as a secure and scalable backend to manage device pairing, wallet metadata, and transaction status. Its real-time synchronization features enabled seamless coordination between devices, supporting a responsive and efficient user experience.
- **Functional End-to-End Application Deployment**
  Borays transitioned from a conceptual prototype into a fully functional web-based wallet. It includes secure mnemonic generation, encrypted storage, transaction validation using two-party ECDSA, and real-time feedback via Firebase. Continuous testing and user feedback played a vital role in refining the app’s performance and stability, confirming its practical readiness for real-world use.
- **Stable MVP and Current System Readiness**
  As of submission, the Borays wallet meets all the essential goals set during the planning phase. It supports secure wallet creation, dual-device pairing, and protected transaction signing—all via a web interface. Deferred features have been clearly documented, ensuring a path forward for future development without compromising the current system’s reliability.

Changes and Rationale

- **NFT Viewer Deferred for Future Integration**
  Originally, the project aimed to include a built-in NFT viewer to allow users to manage and view their Ethereum-based NFTs. However, the complexity of rendering token metadata and ensuring secure validation presented challenges that risked delaying the development timeline. To keep the core wallet features stable and reliable, the NFT functionality was strategically postponed and marked for future enhancement.
- **Shift from Native Mobile to Web-Only Deployment**
  While early plans considered support for Android and iOS through native apps, the team ultimately prioritized a browser-based implementation. This decision was driven by both time limitations and the desire to concentrate technical efforts on perfecting the cryptographic and real-time aspects of the wallet. As a result, the native mobile versions were deferred, enabling the team to deliver a dependable, web-accessible MVP without compromising performance or security.

This focused scope ensured that critical features such as two-device signing and secure key management were fully realized within the available timeframe, laying a strong foundation for future platform expansions.

Traceability Matrix

Requirement	Implemented	Notes
24-word wallet	Yes	BIP-39 standard used; split into 12+12 across devices
Device pairing	Yes	Firebase Connection establishment
Firebase backend	Yes	Real-time data sync and secure user management
Transaction approval	Yes	Transactions require dual device approval
NFT Viewer	No	Feature deferred for future updates

This matrix confirms that all essential functionality was implemented and tested successfully, aligning with the original goals of the project. Deferred features were excluded in a controlled manner to prioritize system stability and secure performance during evaluation.

System Design

Information Architecture

The design of the Borays wallet is built around the concept of secure, dual-device cryptographic communication. This approach ensures that sensitive key material is never fully exposed on a single device, significantly improving security. The system architecture consists of the following core components:

- **Device A**

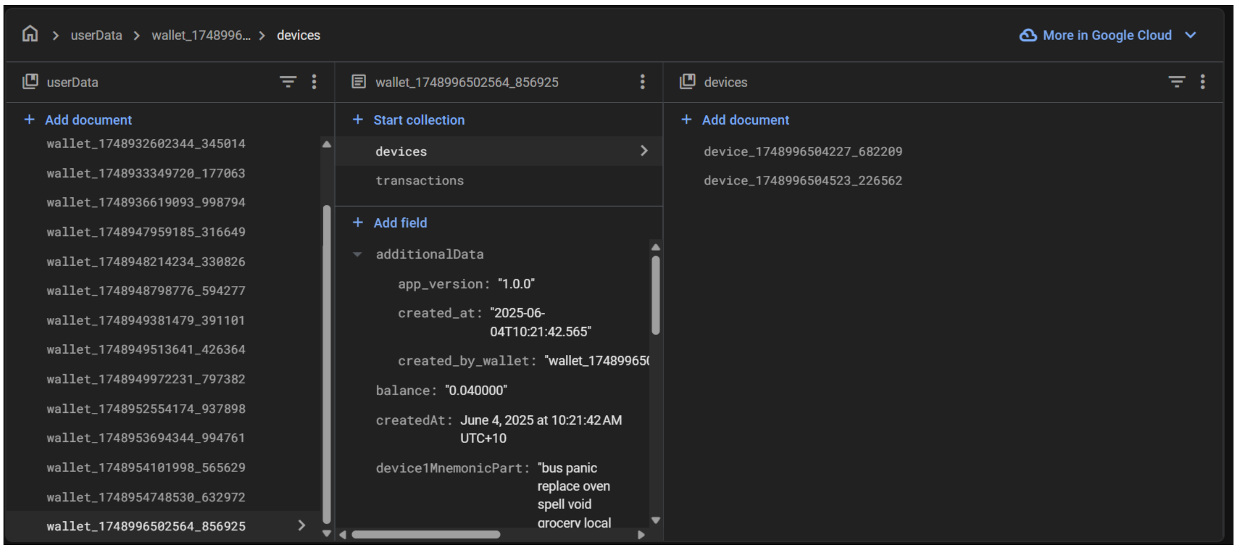
Device A manages the first half (12 words) of the mnemonic phrase. It acts as the user's primary interface, responsible for initiating transactions, managing sessions, and sending signing requests to Device B. It also handles communication with Firebase for real-time data syncing.

- **Device B**
  Device B securely stores the remaining 12 words of the mnemonic. Its sole purpose is to review, approve, and sign transactions initiated by Device A. It fetches transaction requests from Firebase, performs necessary validations, and returns signed data back to the cloud.
- **Firebase Firestore (Cloud Backend)**
  Firebase serves as the backend that coordinates communication between the two devices. It stores encrypted key shares, pairing metadata, transaction records, and device status information. Data entries are indexed by wallet and device IDs, ensuring quick look up and conflict-free updates.

This architecture guarantees that the complete private key never exists on a single device or in transit. Cryptographic operations are handled locally, while only encrypted payloads are exchanged via Firebase. Additionally, the use of asynchronous updates and atomic Firestore writes ensures reliable state management and prevents race conditions during simultaneous device interactions.

Firebase Collections

Borays uses two primary Firestore collections:



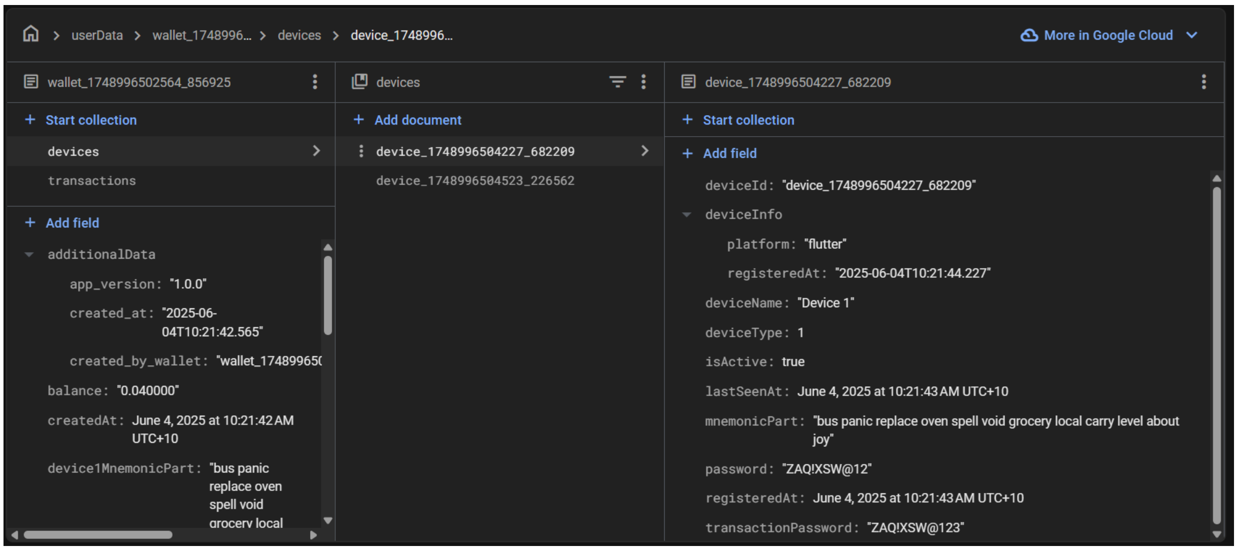
```
userData
wallet_1748932602344_345014
wallet_1748933349720_177063
wallet_1748936619093_998794
wallet_1748947959185_316649
wallet_1748948214234_330826
wallet_1748948798776_594277
wallet_1748949381479_391101
wallet_1748949513641_426364
wallet_1748949972231_797382
wallet_1748952554174_937898
wallet_1748953694344_994761
wallet_1748954101998_565629
wallet_1748954748530_632972
wallet_1748996502564_856925

wallet_1748996502564_856925
devices
transactions

additionalData
    app_version: "1.0.0"
    created_at: "2025-06-04T10:21:42.565"
    created_by_wallet: "wallet_17489965
balance: "0.040000"
createdAt: June 4, 2025 at 10:21:42 AM UTC+10
device1MnemonicPart: "bus panic replace oven spell void grocery local

devices
device_1748996504227_682209
device_1748996504523_226562
```

Figure 1 Firebase devices

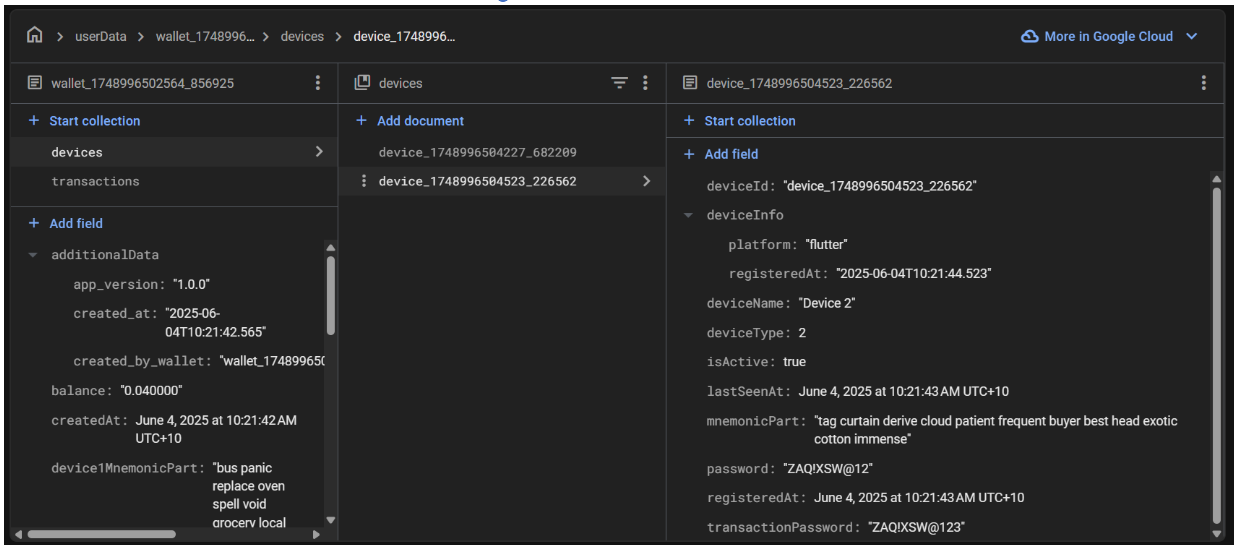

userData > wallet_1748996... > devices > device_1748996...
More in Google Cloud

wallet_1748996502564_856925
+ Start collection
devices
transactions
+ Add field
additionalData
app_version: "1.0.0"
created_at: "2025-06-04T10:21:42.565"
created_by_wallet: "wallet_17489965
balance: "0.040000"
createdAt: June 4, 2025 at 10:21:42 AM UTC+10
device1MnemonicPart: "bus panic replace oven spell void grocery local

devices
+ Add document
device_1748996504227_682209
device_1748996504523_226562

device_1748996504227_682209
+ Start collection
+ Add field
deviceId: "device_1748996504227_682209"
deviceInfo
platform: "flutter"
registeredAt: "2025-06-04T10:21:44.227"
deviceName: "Device 1"
deviceType: 1
isActive: true
lastSeenAt: June 4, 2025 at 10:21:43 AM UTC+10
mnemonicPart: "bus panic replace oven spell void grocery local carry level about joy"
password: "ZAQ!XSW@12"
registeredAt: June 4, 2025 at 10:21:43 AM UTC+10
transactionPassword: "ZAQ!XSW@123"

Figure 2 Firebase devices1


userData > wallet_1748996... > devices > device_1748996...
More in Google Cloud

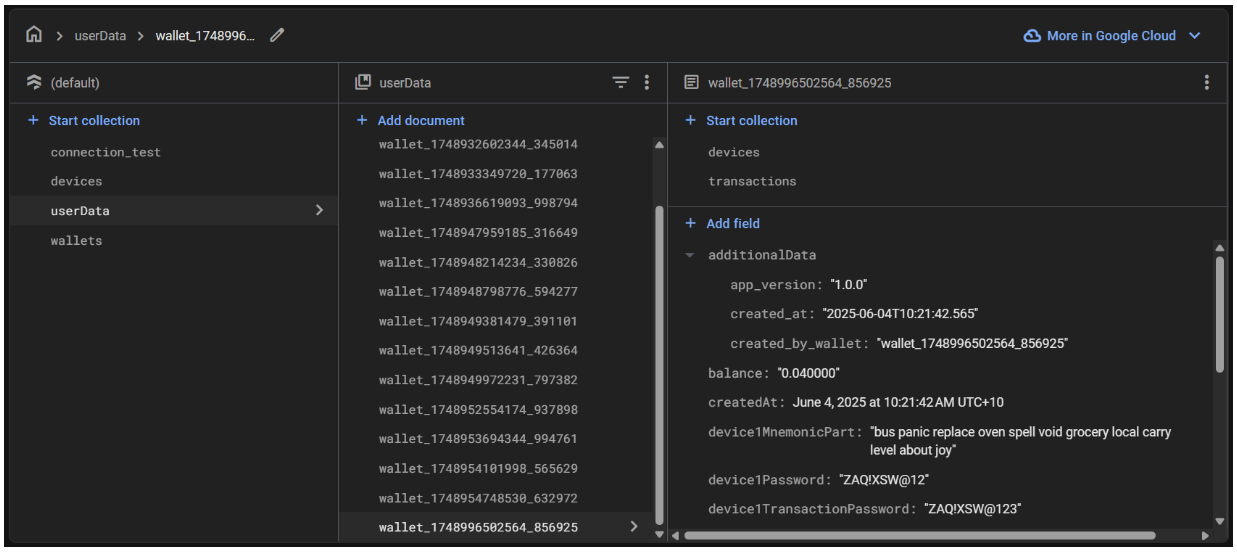
wallet_1748996502564_856925
+ Start collection
devices
transactions
+ Add field
additionalData
app_version: "1.0.0"
created_at: "2025-06-04T10:21:42.565"
created_by_wallet: "wallet_17489965
balance: "0.040000"
createdAt: June 4, 2025 at 10:21:42 AM UTC+10
device1MnemonicPart: "bus panic replace oven spell void grocery local

devices
+ Add document
device_1748996504227_682209
device_1748996504523_226562

device_1748996504523_226562
+ Start collection
+ Add field
deviceId: "device_1748996504523_226562"
deviceInfo
platform: "flutter"
registeredAt: "2025-06-04T10:21:44.523"
deviceName: "Device 2"
deviceType: 2
isActive: true
lastSeenAt: June 4, 2025 at 10:21:43 AM UTC+10
mnemonicPart: "tag curtain derive cloud patient frequent buyer best head exotic cotton immense"
password: "ZAQ!XSW@12"
registeredAt: June 4, 2025 at 10:21:43 AM UTC+10
transactionPassword: "ZAQ!XSW@123"

Figure 3 Firebase devices2

- **devices/:** Contains metadata for each registered device. Key fields include:
  - platform: Platform type (e.g., Web)
  - deviceType: Identifies the role (1 for Device A, 2 for Device B)
  - isActive: Boolean indicating if the device is currently online
  - mnemonicPart: Stores the encrypted half of the mnemonic phrase
  - password: Hashed local password used for login/authentication
  - transactionPassword: Additional password used during transaction approval



```
userData > wallet_1748996...                                   More in Google Cloud

(default)              userData                          wallet_1748996502564_856925
+ Start collection     + Add document                    + Start collection
connection_test        wallet_1748932602344_345014       devices
devices                wallet_1748933349720_177063       transactions
userData               wallet_1748936619093_998794       + Add field
wallets                wallet_1748947959185_316649       additionalData
                       wallet_1748948214234_330826           app_version: "1.0.0"
                       wallet_1748948798776_594277           created_at: "2025-06-04T10:21:42.565"
                       wallet_1748949381479_391101           created_by_wallet: "wallet_1748996502564_856925"
                       wallet_1748949513641_426364       balance: "0.040000"
                       wallet_1748949972231_797382       createdAt: June 4, 2025 at 10:21:42 AM UTC+10
                       wallet_1748952554174_937898       device1MnemonicPart: "bus panic replace oven spell void grocery local carry
                       wallet_1748953694344_994761                            level about joy"
                       wallet_1748954101998_565629       device1Password: "ZAQ!XSW@12"
                       wallet_1748954748530_632972       device1TransactionPassword: "ZAQ!XSW@123"
                       wallet_1748996502564_856925
```

Figure 4 Firebase Wallets

- **wallets/:** Contains session and identity information for each wallet:
  - publicKey: Reconstructed from both devices during setup
  - deviceCount: Tracks how many devices are currently paired
  - transactionCount: Number of transactions executed by the wallet
  - createdAt: Timestamp when the wallet was initialized
  - lastAccessedAt: Timestamp of the most recent device interaction

This structure supports real-time access control, secure authentication, and robust cryptographic validation.

Data Dictionary

Field	Type	Description
mnemonicPart	string	12-word segment of the mnemonic
deviceType	int	1 for Device A, 2 for Device B
password	string	Local password hash
publicKey	string	Reconstructed public key for the wallet
transactionPassword	string	Password used specifically for transaction authentication
isActive	boolean	Status of device availability
lastSeenAt	timestamp	Last recorded active status from the device

This system design supports modular, secure, and synchronized cryptographic wallet operations, minimizing attack surfaces while enhancing usability and real-time responsiveness.

Group Meeting Agendas and Minutes

Throughout the development of Borays, our team maintained consistent communication through structured weekly meetings, ensuring transparency, coordination, and accountability across all project stages. Each meeting focused on task updates, roadblocks, and critical decisions. The records below reflect a clear and logical progression from ideation to project finalization.

Meeting Structure:

- **Progress Updates:** Members shared completed tasks, blockers, and resolutions.
- **Sprint Planning:** Tasks were assigned for the upcoming sprint based on project milestones.
- **Technical Discussions:** We addressed issues involving Flutter integration, Firebase configuration, cryptography, and deployment.
- **Demo Reviews:** Functional modules were demonstrated, and improvements discussed.
- **Supervisor Feedback:** We discussed input received from supervisor and integrated key suggestions into upcoming plans.

Meeting Records

1. Initial Development Phase

- **Attendees:** Adhiraj, Puneeth, Sweeto, Vinay, Sarath
- **Agenda Items:**
  - Discuss Firebase integration and authentication logic.
  - Share progress on UI wireframes and frontend routing.
  - Assign mnemonic generation and BIP-39 standard research.
- **Decisions:**
  - Adopt BIP-39 standard for secure mnemonic creation.
  - Configure Firestore collections for users and transactions.
- **Next Steps:**
  - Complete password hashing mechanism.
  - Prepare components for Sprint 2 UI demo.

2. Cryptographic Module Integration

- **Attendees:** Puneeth, Sarath, Vinay.
- **Agenda Items:**
  - Integrate Paillier homomorphic encryption.
  - Design the two-party ECDSA signature protocol.
  - Define test cases for key share validation.
- **Decisions:**
  - Conduct internal tests with dummy transaction data.

3. Transaction Co-signing Implementation

- **Attendees:** Full team.
- **Agenda Items:**
  - Connect signing protocol between both devices.
  - Enable real-time transaction syncing using Firebase.
  - Establish dual-device verification logic.
- **Decisions:**
  - Finalize signing flow and transaction lifecycle logic.

    - Log approvals and rejections for user auditing.

4. UI/UX Streamlining

- **Attendees:** Full team.
- **Agenda Items:**
    - Analyse feedback from early testers.
    - Improve device pairing and onboarding sequence.
    - Refine transaction status indicators.
- **Decisions:**
    - Redesign linking screen for better clarity.
    - Address responsiveness issues across screen sizes.

5. Security Review and Protocol Validation

- **Attendees:** Sarath, Puneeth, Vinay.
- **Agenda Items:**
    - Review cryptographic modules for vulnerabilities.
    - Validate full mnemonic is never reconstructed or transmitted.
    - Perform signature stress tests.
- **Decisions:**
    - Confirm secure transmission logic.
    - Document edge cases and unexpected transaction behaviours.

6. Testing and QA Strategy

- **Attendees:** Adhiraj, Sweeto.
- **Agenda Items:**
    - Draft test plans for rejection scenarios and session losses.
    - Initiate multi-browser testing across devices.
    - Maintain bug-tracking log for team-wide visibility.
- **Decisions:**
    - Assign testing across different browser combinations.
    - Implement retry logic for signing failures.

7. Documentation and Submission Planning

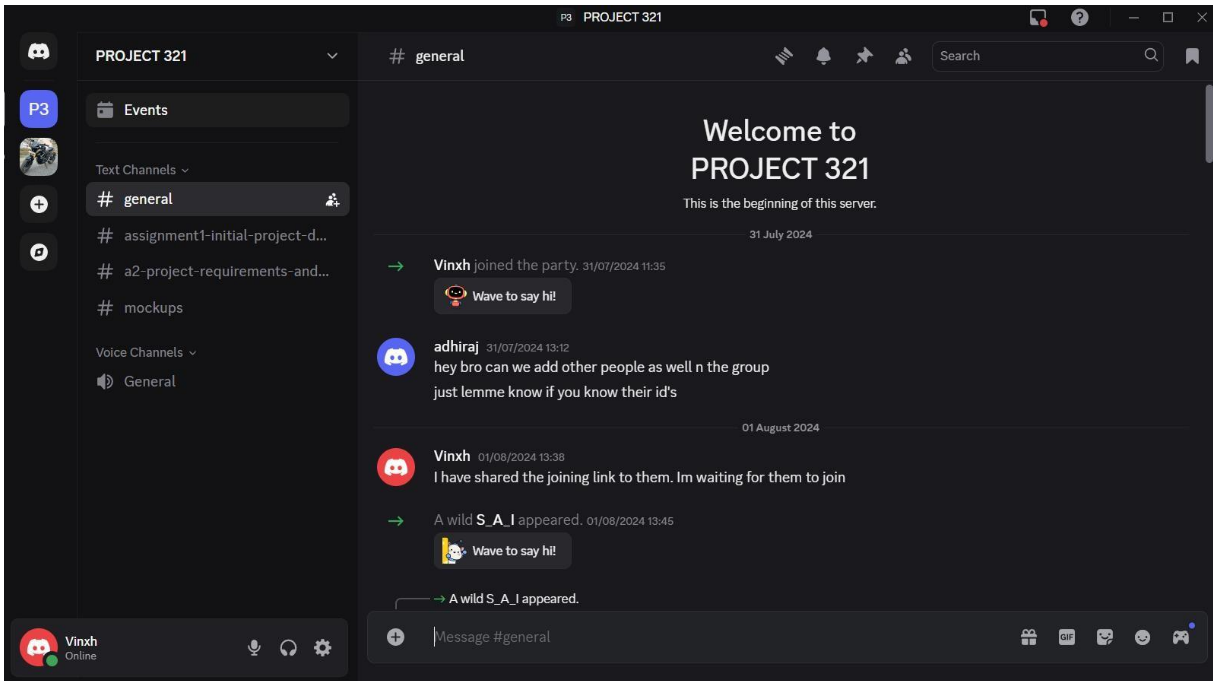
- **Attendees:** Full team.
- **Agenda Items:**
    - Finalize the technical documentation.
    - Delegate sections for the user manual and walkthrough.
    - Create system architecture diagrams and Firebase schema visuals.
- **Decisions:**
    - Submit all documents in required formats (PDF + HTML).
    - Complete draft review before submission window.

8. Final Touches and MVP Submission

- **Attendees:** Full team.
- **Agenda Items:**
  - Apply bug fixes and final visual tweaks.
  - Assign roles for technical presentation video.
  - Upload the final app build and documentation.
- **Decisions:**
  - Freeze the codebase for final submission.
  - Complete video and documentation.

9. Trade Show and Pitch Planning

- **Attendees:** Full team.
- **Agenda Items:**
  - Plan the live demo flow to highlight pairing and signing.
  - Anticipate and prepare answers for common technical questions.
- **Decisions:**
  - Run a mock walkthrough before the event.
  - Finalize slides and backup demo setup.



Welcome to PROJECT 321

This is the beginning of this server.

31 July 2024

Vinxh joined the party. 31/07/2024 11:35

adhiraj 31/07/2024 13:12
hey bro can we add other people as well n the group
just lemme know if you know their id's

01 August 2024

Vinxh 01/08/2024 13:38
I have shared the joining link to them. Im waiting for them to join

A wild S_A_I appeared. 01/08/2024 13:45

A wild S_A_I appeared.

Figure 5 Discord Server used for communication

To keep our development on track, we regularly held team meetings that allowed us to align on progress, clarify goals, and solve any blockers that arose. These meetings provided a space to update each other on completed tasks, plan upcoming work, and handle technical or design challenges as a team. Whether we were integrating Firebase, troubleshooting cryptographic operations, or preparing for submission, these sessions helped maintain a steady and collaborative workflow. The records below highlight how our teamwork evolved from initial setup to final delivery.

System Installation process

The Borays wallet is a Flutter-based web application designed for browser access. Follow the steps below to set up and run the application locally using Android Studio:

1. Install Required Software
   - Download and Install Android Studio
     - Ensure that the Flutter and Dart plugins are enabled during installation.
   - Install Flutter SDK
     - Download the latest stable release from: https://flutter.dev/docs/get-started/install
   - Set Environment Variables
     - Add the Flutter SDK path to the system PATH variable to allow Flutter commands from any terminal:
       - Example: C:\flutter\bin

2. Download the Project Files

- Clone or download the Borays wallet project from GitHub:
  https://github.com/Vickkysai/Borays_Crypto_Wallet.git

- Alternatively, download the ZIP file and extract it.

3. Open the Project

- Launch Android Studio.
- Select "Open an existing project" and navigate to the downloaded borays-wallet folder.

4. Get Dependencies

- Once the project is opened, run the following in the terminal or use "Pub get":
  - Example: flutter pub get

5. Run the Application

- Connect a supported browser (e.g., Google Chrome).
- In Android Studio, select Chrome (Web) as the target device.
- Click Run or use the command:
  - Example: flutter run -d chrome

6. Important Compatibility Note (Optional)

- Some systems may face a CardTheme-related error when launching the app.
- To fix this:
  - Open the file: lib/theme/app_theme.dart
  - Modify the following lines:
    - Line 43: Change CardTheme(...) to CardThemeData(...)
    - Line 134: Same change from CardTheme(...) to CardThemeData(...)
- Save the file and re-run the application.

Project Closeout

Lessons Learned

During the development of Borays, the team encountered a range of technical and usability challenges. These experiences helped shape the final product and will serve as valuable guidance for future development cycles:

2. **Firebase Strengths and Risks**
   Firebase proved to be a powerful tool for enabling real-time synchronization and scalable backend functionality. However, we learned that configuring Firestore security rules correctly is essential—any oversight could potentially expose sensitive wallet data. Future deployments must include rigorous rule audits and access testing.
3. **Managing Race Conditions in Real-Time Operations**
   As devices interacted simultaneously through Firebase, we discovered race conditions that could affect transaction state consistency. To address this, we implemented sequential write logic and additional validation checks, reinforcing the importance of handling concurrency in real-time applications.
4. **Strict On-Device Mnemonic Enforcement**
   Security requirements mandated that the full mnemonic phrase must never be stored or transmitted in one piece. Enforcing this split on-device added complexity but significantly strengthened the system's security posture, ensuring that key reconstruction is only possible when both devices are used together.
5. **Prioritizing Simplicity in the UI**
   Early user testing revealed that complex onboarding flows, especially during pairing, caused confusion. By simplifying the interface and focusing only on essential actions, we greatly improved user experience and minimized potential misuse or error.
6. **Balancing Cryptographic Security with Performance**
   Integrating Paillier encryption ensured secure communication, but it also introduced latency. We learned to optimize these operations by streamlining background processes and minimizing encryption cycles where possible, without compromising security.
7. **Cross-Browser Compatibility is Crucial**
   Since Borays runs as a web-only application, testing across various browsers was essential. We encountered and resolved compatibility issues, reinforcing the need for thorough cross-platform validation during QA.

Overall, these lessons informed important architectural and usability decisions. They helped the team deliver a wallet system that is not only secure but also practical and accessible for real-world use.

Project Acceptance

The Borays project has successfully met all core acceptance criteria outlined by stakeholders and course requirements, with final validation pending public demonstration:

8. The wallet system has been completed and is scheduled to be showcased at the upcoming university trade show, where it will be demonstrated to supervisors, peers, and evaluators.

9. The application fully supports all critical features, including wallet creation, mnemonic phrase generation, secure device pairing, dual-approval transaction signing, and transaction history tracking.
10. Functionality has been reviewed internally and verified by our academic mentor as part of pre-submission evaluation.
11. The final version passed peer testing and internal code reviews, meeting technical expectations for grading.
12. Supporting documentation—including architecture design, and a user manual—has been finalized and submitted alongside the application.

This indicates that Borays is academically sound, aligns with real-world blockchain security requirements, and is ready for demonstration and future deployment.

Future Improvements

To ensure Borays continues to evolve into a more robust and widely usable platform, a clear roadmap of future enhancements has been identified. These planned improvements aim to increase functionality, security, and accessibility while maintaining the wallet's core principles of distributed trust and strong cryptographic protection:

- **Cross-Platform Native App Development:** Future versions will include dedicated Android and iOS apps, developed using Flutter or React Native. This will offer users a native experience with better performance and access to mobile-specific security features like biometric authentication.
- **Browser Extension Integration:** A browser extension version of Borays is proposed for platforms like Chrome and Firefox. This will integrate the wallet directly into the dApp ecosystem and provide users with a familiar experience similar to MetaMask, but with added dual-device security.
- **Support for Multiple Blockchains:** Expanding beyond Ethereum, future updates will introduce support for other popular networks such as Binance Smart Chain, Polygon, and Solana. This will allow users to manage diverse assets in one secure environment.
- **Offline Transaction Queuing:** A planned feature will let users prepare and queue transactions while offline. These will be signed and submitted automatically once both devices are back online, increasing the wallet's reliability in unstable network conditions.
- **Smarter Notification System:** The notification framework will be improved to include real-time push alerts and optional email notifications for critical events such as pending transaction approvals, wallet activity, or potential security threats.
- **UI and UX Enhancements:** Ongoing design improvements will include dark mode support, improved responsiveness for smaller screens, accessibility compliance, and smoother transitions for a more intuitive user experience.
- **Adoption of Advanced Cryptography:** To further reduce communication overhead and improve transaction speed, Borays will explore implementing more advanced schemes like threshold ECDSA and BLS signatures, while preserving current security assurances.

These upgrades are designed to future-proof Borays and ensure that it remains secure, user-friendly, and adaptable in the fast-evolving landscape of decentralized applications and digital finance.

Conclusion

Borays began as a concept to tackle the common security pitfalls of single-device blockchain wallets, and it has now evolved into a functional, secure, and user-friendly dual-device wallet system. By splitting the private key across two devices and using strong cryptographic methods like Paillier encryption and two-party ECDSA, we created a system that ensures both safety and control remain in the user's hands.

Our development journey followed an iterative approach, allowing us to adjust and improve the system with each sprint. We focused on building core features such as wallet generation, secure device pairing, and transaction co-signing, all running smoothly on a web-based platform with real-time updates via Firebase.

With all major milestones completed and a stable MVP in place, Borays is ready for live testing and demonstration. It has been carefully designed to meet both technical and user experience expectations, while also being scalable for future improvements.

Looking ahead, there's clear potential to grow—whether that's through native mobile apps, browser extensions, or support for multiple blockchain networks. What we've built is a solid foundation for a more secure and flexible crypto wallet experience, and we're excited about what comes next.

Prepared by:

Vinay Chandra Kyatham, Sarath Sai Siddagunta, Puneeth Reddy Yanamala, Adhiraj Singh and Sweeto Babu.

CSIT321 – University of Wollongong – 2025