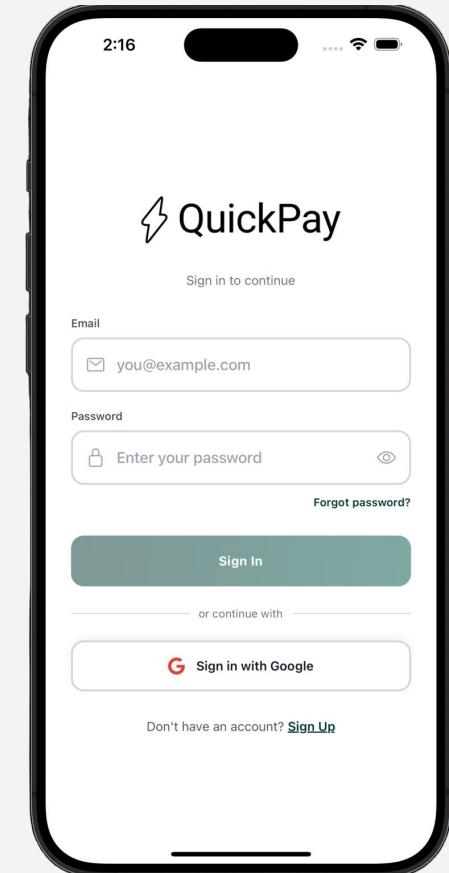


⚡ QuickPay

Team B – CSCI 441 Software Engineering

Members:

- Chanrattnak Mong - Frontend & Authentication Engineer
- Seanglong Lim - Backend Integration Engineer
- Seth Tharo Hour - Backend Integration Engineer
- Sok Sreng Chan - Frontend Engineer



Customer Problem & Requirements



Problem:

- Hard to track spending across several bank apps
- Manual updates → inaccurate balances
- Stress over bill-splitting and delayed payments

Customer Wants:

- One dashboard for all banks
- Real-time spending visualization
- QR/group payments & full transparency

Current System Specifications

Functional:

- QR Code & Split payments
- Flowchart visual budgeting

Non-Functional:

- Security - Clerk, PLAID, Stripe
- Uptime - Supabase Serverless
- Do not store sensitive user data



Tools & Technologies

Frontend



Backend (Underway)



Database



API Integration



Authentication



Deployment



Project Roadmap

The screenshot shows a project management interface with a dark theme. At the top, there's a navigation bar with icons for file, search, and user profile, followed by the path "CSCI441-QuickPay / Projects / @QuickPay's project". Below the navigation is a header with sections for "Backlog", "Team capacity", "Current iteration", "Roadmap", "My items", "New view", and search/filter fields. There are also buttons for "Add status update", "Insights", "Workflows 4", and a three-dot menu.

The main area displays four columns representing project phases:

- Design**: 2 / 5 Estimate: 0. Sub-tasks include "Design UX/UI and UML Diagram" and "System Design" (with items #2 and #3).
- Development**: 4 / 5 Estimate: 0. Sub-tasks include "Write Code and Create Database" and several sub-items under "Frontend setup" and "Backend setup" (items #7 through #13).
- Testing**: 5 Estimate: 0. Sub-tasks include "Quality Assurance and Security Testing" and several sub-items (items #8 through #12).
- Deployment**: 1 Estimate: 0. Sub-task is "Website Hosting" (item #13).

Each phase has a "+ Add item" button at the bottom. The interface uses color-coded circles to indicate the status of tasks: green for Design, blue for Development, orange for Testing, and purple for Deployment.

Project Challenges & Future Work

Challenges:

- Security Implementation
- Unfamiliar API Integrations
- Real-Time Data Handling
- Cross-Platform Development
(Android & iOS)

Future Work:

- Conduct Security Tests
- Implement Multi-account Linking
- Enable Request/Send Payments
- Support Real-Time Transactions



Competitive Analysis

Features/ Competitors	⚡ QuickPay	\$ Cash App	Rocket Money	CHASE
Non Custodial	YES	NO	YES	No
Multi-Account Linking	YES	NO	YES	NO
Expense Splitting	YES	Send / Receive Only	NO	Manual Transfer
Budgeting & Insights	<i>Interactive Budget Flow</i>	Basic Spend List	AI-based Budgeting	Basic Charts
Payment Method	<i>QR-Code Payment</i>	QR-Code Payment	NO	Bank transfers
Financial Alerts	<i>Real-time overspend/ low balance</i>	Send / Receive Only	Bill reminders	Standard alerts



Conclusion

- QuickPay aims to transform how people manage their daily finances by offering an all-in-one platform that connects multiple bank accounts, automates budgeting, and simplifies group payments.
- With our advance integration like **Plaid**, **Stripe**, and **Clerk**, our team designed a secure and user-friendly system that gives users real-time visibility into their financial activities.
- The project not only reduces the stress of manual tracking but also encourages responsible spending habits.
- Moving forward, we plan to enhance QuickPay features, improve its user interface, and prepare it for real-world deployment to help people take full control of their finances effortlessly.
- **Test our project:** <https://github.com/CSCI441-QuickPay/QuickPay-MobileApp>