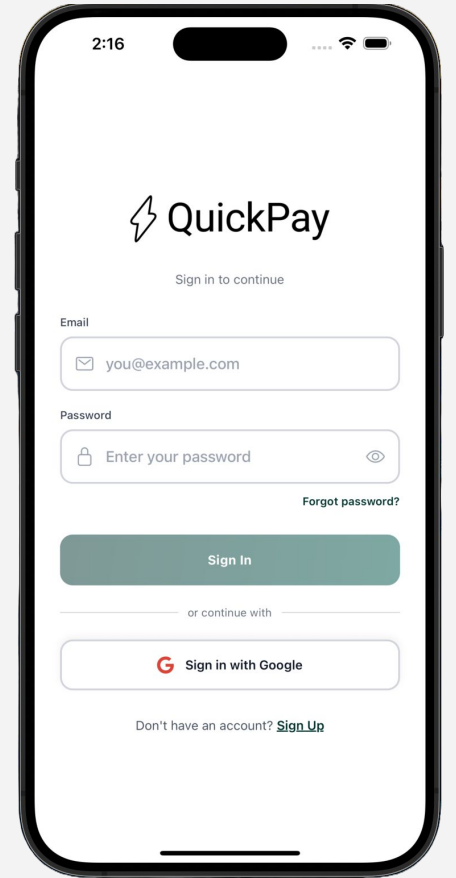




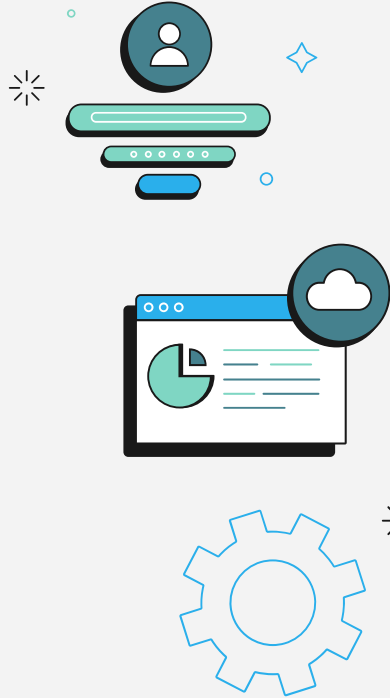
Team B – CSCI 441 Software Engineering

### 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

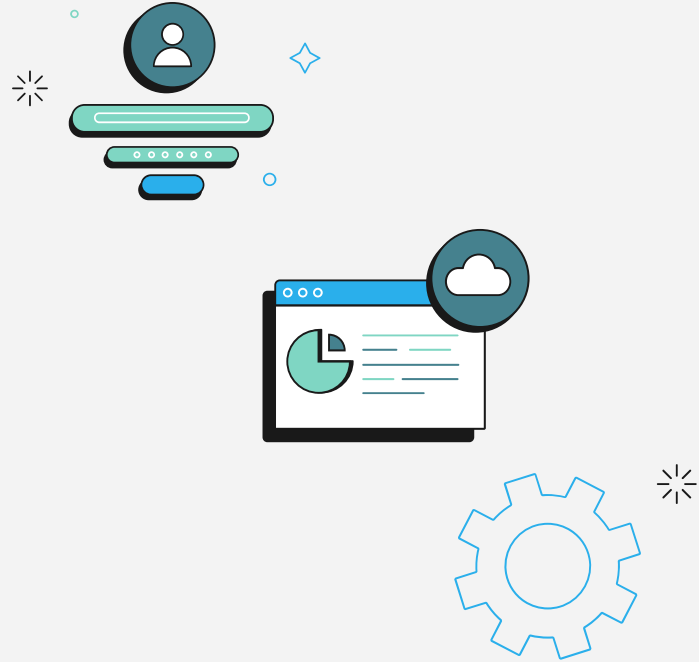
# Current System Specifications

## Functional:

- QR Code & Split payments
- Flowchart visual budgeting

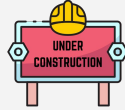
## Non-Functional:

- 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

CSCI441-QuickPay / Projects / @QuickPay's project

Q

Type to search

@QuickPay's project

Add status update

Insights

Workflows 4

Backlog

Team capacity

Current iteration

Roadmap

My items

New view

Filter by keyword or by field

Discard

Save

Design 2 / 5 Estimate: 0

...

Design UX/UI and UML Diagram

QuickPay-MobileApp #2

System Design

QuickPay-MobileApp #3

UI/UX Integration

+ Add item

Development 4 / 5 Estimate: 0

...

Write Code and Create Database

QuickPay-MobileApp #7

Frontend setup

P1 Iteration

QuickPay-MobileApp #4

Backend setup

QuickPay-MobileApp #5

Integrate Clerk Authentication.

QuickPay-MobileApp #6

QR code payments UI & scanning logic

+ Add item

Testing 5 Estimate: 0

...

Quality Assurance and Security Testing

QuickPay-MobileApp #8

Unit & integration testing for Multi-account linking & routing logic.

QuickPay-MobileApp #9

Unit & integration testing for QR payment speed

QuickPay-MobileApp #10

Unit & integration testing for Budget visualization accuracy

QuickPay-MobileApp #11

Usability testing on flow-chart clarity and navigation

QuickPay-MobileApp #12

Performance testing

+ Add item

Deployment 1 Estimate: 0

...

Website Hosting

QuickPay-MobileApp #13

Deployment on iOS using Expo Go

+ Add item

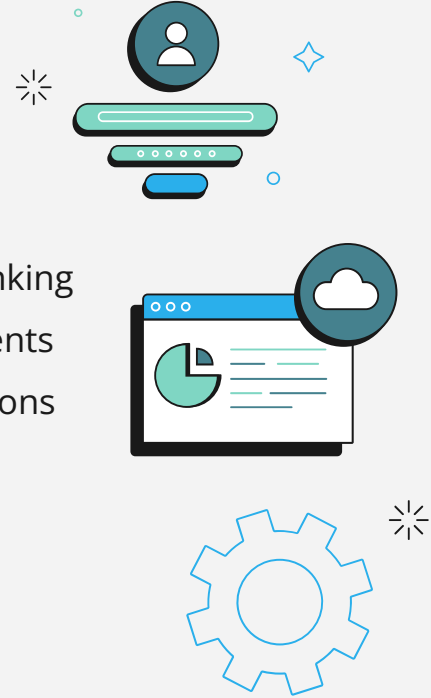
# Project Challenges & Future Work

## 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>