

HackKU 2024 - Koala Pouch

Team Members: Alex Doebling, Nicholas Holmes, Kyle Spragg, Colin Traenor

Software Development Plan for XRPL Digital Wallet

Project Overview

Project Name: Koala Pouch: an XRPL Digital Wallet

Team Members:

Front-End:

- Nicholas Holmes
- Kyle Spragg

Back-End:

- Alex Doebling
- Colin Traenor

Hackathon Strategy

- Goal: Develop a minimum viable product (MVP) of an XRPL digital wallet with basic functionalities: viewing balances, sending XRP, and receiving XRP.
- Approach: Agile-like, continuous iterations with ongoing integration and testing.

Communication and Coordination

- Communication Tools: Mainly in-person communication along with group chats dedicated to scheduling and quick communication.
- Collaboration and Version Control: GitHub for real-time code sharing, updates, and collaboration among team members. Frequent commits to ensure progress is saved and accessible.

Front-End Development (Continuous Workflow)

Design Ideas

- Theme: Koala (cute marsupial, get it, it has a pouch haha)
- Design Platform: Figma

Technologies and Libraries

- Languages: JavaScript (ES6+)
- Framework: React

- Styling: CSS3

Hackathon Steps

Initial Setup:

- Quick setup of the React environment using Create React App.
- Establish initial repository and branch strategy on GitHub.

UI Development:

- Rapid prototyping and design implementation.
- Continuous testing and refinement of UI components.

Integration with Back-End:

- Implement real-time connection with the backend API for fetching and sending data.
- Continuous testing and tweaking based on backend updates and requirements.

Back-End Development (Continuous Workflow)

Technologies and Libraries

- Languages: Python 3
- Framework: Flask
- XRPL Interaction: xrpl-py library

Hackathon Steps

API Development:

- Setup Flask and basic route handling.
- Implement core functionalities: connect to XRPL, handle transactions, fetch account balances.

Continuous Integration:

- Regularly update API endpoints and test with frontend inputs.
- Adjust and optimize based on feedback and real-time testing results.

Testing and Integration

- Continuous Testing: Use tools like Jest for front-end and pytest for backend to continuously test components as they are developed.
- Integration Sessions: Hold frequent integration sessions to ensure front-end and back-end components work seamlessly together.

Deployment and Presentation

- Mock Deployment: Use services like Heroku for quick deployment to test the full application in a production-like environment.

- Final Run-through: Before presentation, conduct a complete run-through to ensure functionality and polish any UI/UX aspects.

Additional Notes

- Focus on MVP: Given the time constraint, prioritize core functionalities over advanced features.
- Documentation: Keep track of development steps and decisions for presentation and future reference.
- Health and Stamina: Schedule short breaks and ensure team members stay hydrated and energized.

Dependencies

Front-End Dependencies

Node.js and npm: As the runtime environment and package manager, these are essential for managing and installing other JavaScript packages.

- Download from [Node.js official website](https://nodejs.org/).

React: The main library for building the user interface.

- Install via npm:
npm create-react-app my-app
cd my-app

Redux (optional, depending on state management needs):

- Install via npm:
npm install redux react-redux

Axios (for making HTTP requests to the back-end):

- Install via npm:
npm install axios

Additional styling libraries (such as Bootstrap or Material-UI, if used):

- Bootstrap:
npm install bootstrap
- Material-UI:
npm install @mui/material @emotion/react @emotion/styled

Back-End Dependencies

Python: The programming language used for the back-end.

- Download from [Python official website](https://www.python.org/).

Flask: The web framework for handling HTTP requests and serving the API.

- Install using pip:
pip install Flask

xrpl-py: The official Python library for interacting with the XRP Ledger.

- Install using pip:
pip install xrpl-py

pytest (for testing the Python code):

- Install using pip:
pip install pytest

Virtual Environment:

- Install virtualenv if it's not installed:
pip install virtualenv
virtualenv myenv
source myenv/bin/activate – On Windows use `myenv\Scripts\activate`

Back-end Functions ([Transaction Methods — xrpl-py documentation](#))

Front end requirements:

- Login option if they already have an account
 - Create an account option if they don't have an account
- When logged in display the balance and previous transaction history

***seed -> the unique identifier for each wallet, used to get the information of an account

create_account():

Functionality: Called when user does not have an account

Creates a faucet wallet for user

Parameters:

None

get_account(seed):

Functionality: Called when user is signing back in (already has account)

Uses their seed to return their wallet that was already created

Parameters: seed

get_info(seed):

Functionality: returns a lot of data regarding the account

Parameters: seed

last_transaction(seed):

Functionality: when called gets the most recent transaction BUT if there was no recent transaction then will print an error

Parameters: seed

wallet_to_json(wallet):

Functionality: converts wallet data to json

Parameters: a users wallet

send_xrp(seed, amount, destination):

Functionality: send xrp to another user

Parameters: (seed of the account sending money, amount being sent, address of the destination account)

Design Process

Step 1) Border

- a. Establish border size to act as constant state of page
- b. Design Koala for homepage button
 - i. Arms separate object to hang over border
 - ii. Possibly have eye dots follow the cursor??
- c. Design bamboo shoots for right and left border
- d. Koala Wallet title head
- e. Settings button in top right corner

Step2) Login

- a. Login header
- b. Seed entry input box
- c. Login button
- d. Create new account button (automatically creates custom seed)

Step3) Main

- a. Split area into 3 columns.
- b. Left column displays contact names
- c. Center column is split into top and bottom halves. Top displays: balance, send & sell buttons; Bottom: graph
- d. Right column displays transaction history

Step4) Settings

- a. Public key display
- b. Private key display
- c. Message thanking users

Possible Errors

Send Button:

Possible Issue: User sends more money than available in account: Could be an overload or sending when account is empty

Fix: Will default to sending only .00001