Building a Payment Splitter iOS Application Based on Blockchain

Yan-Da Chen / Po-Hsuan Chen / Che-Yu Chang
Ling-Yuan Chen / Wei-Ting Ho / Yan-Yu Huang
Outline

● Introduction

● Implementation
  ○ Frontend Development
  ○ Backend Development

● Risks and Difficulties

● Future Work

● Demo
Introduction

- Payment splitting applications to keep track of money
- Split Ether
- Cryptocurrency payment splitter application
Introduction

- Payment splitting applications to keep track of money
- Split Ether
- Cryptocurrency payment splitter application
Introduction - Payment Splitter Flow

Create Participants
Amy
Alan

Set Agreement

Check Balance

Create Expense
Amy
Alan
Me

Create Payment
Me
Implementation

Front-end

Back-end

Swift

node
Implementation - Blockchain

Smart Contract

Development Tool

SOLIDITY

TRUFFLE

Ganache
Frontend Development

- Swift UI
  - User interface toolkit
- Swift
  - iOS develop language
# Backend Development - API

<table>
<thead>
<tr>
<th>API</th>
<th>REST Verb</th>
<th>Request Type</th>
<th>Request Structure</th>
<th>Response Structure</th>
<th>Request Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/createAccount</td>
<td>POST</td>
<td>JSON</td>
<td>`{</td>
<td>[</td>
<td>Register a new user and create the password.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;name&quot;: &quot;alan&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;password&quot;: &quot;alan5411&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>/login</td>
<td>POST</td>
<td>JSON</td>
<td>`{</td>
<td>{</td>
<td>Login a user and get the login status.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;name&quot;: &quot;alan&quot;,</td>
<td>&quot;info&quot;: &quot;Login Successful!&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;password&quot;: &quot;alan5411&quot;</td>
<td>}</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
<td>OR</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{</td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;info&quot;: &quot;Wrong Password!&quot;</td>
<td>&quot;info&quot;: &quot;Wrong Password!&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>/showAmount/:name</td>
<td>GET</td>
<td>/showAccount/alan</td>
<td>/showAccount/alan</td>
<td>[</td>
<td>Enter the name to get the amount of that account.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{</td>
<td>&quot;amount&quot;: 99.43462224</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
<td>}</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>}</td>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
### Backend Development - API

<table>
<thead>
<tr>
<th>Method</th>
<th>Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/getAllParticipants</td>
<td>Get all participants' names and balance.</td>
</tr>
</tbody>
</table>

```json
{
    "info": "check all participants successful",
    "balanceInfo": [
        {
            "name": "amy",
            "balance": 1750000000000000000
        },
        {
            "name": "byron",
            "balance": -7500000000000000000
        },
        {
            "name": "sherry",
            "balance": 1250000000000000000
        },
        {
            "name": "alan",
            "balance": -7500000000000000000
        },
        {
            "name": "ginny",
            "balance": -1500000000000000000
        }
    ]
}
```
# Backend Development - API

<table>
<thead>
<tr>
<th>Path</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/showParticipant/:id</td>
<td>GET</td>
<td>Enter the id to get the name, balance, address from that account.</td>
</tr>
<tr>
<td>/createExpense</td>
<td>POST</td>
<td>Create an expense, including the title, amount, data, payer and payee.</td>
</tr>
</tbody>
</table>

```
/showParticipant/1
{
    "Name": "alan",
    "Balance": 0,
    "Address": "0xE1a62eF686fe820DE76191e88fBa648A37dE696d"
}
```

```
/createExpense
{
    "title": "food",
    "amount": 500000,
    "date": 2020,
    "payer": "Amy",
    "payee": "Sherry"
}
```
<table>
<thead>
<tr>
<th>Route</th>
<th>Method</th>
<th>Content-Type</th>
<th>Request Body</th>
<th>Response Body</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/createPayment</td>
<td>POST</td>
<td>JSON</td>
<td>`{</td>
<td>[</td>
<td>Make a payment from one account to another account.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;payer&quot;: &quot;Amy&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;payee&quot;: &quot;Sherry&quot;,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;amount&quot;: 50000 }</td>
<td>[</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&quot;sources&quot;: &quot;200&quot; }</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[</td>
<td></td>
</tr>
<tr>
<td>/agree</td>
<td>POST</td>
<td>JSON</td>
<td>`{</td>
<td>{</td>
<td>Set agreement to expense.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;name&quot;: &quot;amy&quot;,</td>
<td>&quot;info&quot;: &quot;amy agree expense 2&quot; }</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&quot;expenseID&quot;: 2 }</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Risks and Difficulties

Risks

- High latency of blockchain

Difficulties

- New development environment
- Integration of backend and smart contract
Future Work

● Add different splitting actions
  ○ split by the **number of shares**
  ○ split by the **percentage**
  ○ share by **exact amounts**

● Combine debts
  ○ Ex. A owes B $10 and B owes C $10 = A owes C $10
  ○ **avoid extra payments**
  ○ **reduce transaction gas**
Demo
Team Work

Yan-Yu Huang
- User interface design
- Front end development

Yan-Da Chen
- Smart contract test
- Node.js API test

Po-Hsuan Chen
- Smart contract design
- Node.js API test

Wei-Ting Ho
- Smart contract test
- Node.js API design

Che-Yu Chang
- Front back end integration
- Node.js API design

Ling-Yuan Chen
- Front back end integration
- Smart contract design
Thank You