



## **FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY**

### **BMIS2003 Blockchain Application Development**

#### **Assignment**

**Academic Session: 202405**

**Programme: RDS3 & RIS3**

**Tutorial Group: Group 2 & Group 5**

#### **Team Members:**

<b>No</b>	<b>Student Name &amp; Photo</b>	<b>Student ID</b>	<b>Signature</b>
<b>1</b>	<b>Ong Weng Kai</b>	<b>22WMR03309</b>	<i>KAI</i>
<b>2</b>	<b>Yong Zee Lin</b>	<b>22WMR03770</b>	<i>YongZL</i>
<b>3</b>	<b>Ryan Kho Yuen Thian</b>	<b>22WMR04097</b>	<i>RyanK</i>
<b>4</b>	<b>Chong Zi Feng</b>	<b>23WMR11730</b>	<i>ZF</i>
<b>5</b>	<b>Lim Jing Shaw</b>	<b>23WMR09236</b>	<i>SHAW</i>

# 1.0 Introduction (Timelock DApp)

## **Problem statement**

Parenting encompasses a range of responsibilities, from nurturing emotional growth to teaching practical life skills. A paramount task among these is instilling financial wisdom in children, a necessity that prepares them for the complexities of future financial independence. This educational responsibility goes beyond traditional methods like piggy banks; it requires a structured approach that integrates comprehensive financial lessons suited to the evolving economic environment. The importance of early financial education cannot be overstated. By introducing concepts of money management from a young age, parents lay a robust foundation for lifelong financial behaviors. This approach involves teaching children the intrinsic value of money, the benefits of saving, and the discipline required for budgeting. Such education equips children with the tools to make informed financial decisions, thereby setting them up to manage their finances adeptly and build a stable future (Edge, 2021).

The objective of introducing a parental allowance system is multifaceted. This system is designed to teach children about earning, saving, and spending through a series of structured allowances linked to age-appropriate financial tasks. These tasks, which mimic real-world financial situations, provide a practical framework within which children can apply their learning. For instance, younger children might start with simple saving tasks, while older children could be introduced to budgeting for larger goals like technology purchases or charitable donations (*Teaching Children About Financial Responsibilities* | AmMetLife, n.d.).

The expected impact of implementing a structured financial education program through a parental allowance system extends beyond financial literacy. It aims to cultivate essential life skills, such as self-discipline, strategic thinking, and empathy. These skills are integral to personal success and responsible community involvement. By fostering these capabilities, the program not only prepares children for personal financial success but also shapes them into well-rounded, community-minded individuals. Ultimately, this structured approach to financial education through a parental allowance system represents a vital tool in the parental toolkit, essential for nurturing a generation that is financially literate and socially responsible. This comprehensive strategy ensures that children grow up with a balanced understanding of money management, prepared to navigate the financial challenges of adulthood (W et al., 2023).

## **Proposed Solution**

To solve those issues and instill such disciplines, we propose to create a decentralized application (dApp) that uses both Ethereum Sepolia testnet and Ganache to deploy time-

locked smart contracts to carry out transactions from parents to their children in a timely and efficient manner. To achieve the “time-locked” functionality, we will deploy a “get the latest timestamp” smart contract on Ethereum Sepolia testnet, which will take advantage of two services: **Oracle** and **Upkeep** services. The Chainlink nodes from LinkWell Nodes will provide the Oracle service, which returns the latest timestamp (off-chain data) from an external API to ensure transactions execute at the correct time. For the Upkeep, we will apply Chainlink’s Upkeep service on the “get the latest timestamp” smart contract, which will get the latest timestamp at regular intervals. All other smart contracts will be deployed on Ganache. Note that the reason for using Ethereum Sepolia testnet as well is because Chainlink services can only be used on the mainnet and testnet. Moreover, setting up a local Chainlink node is difficult as it involves a lot of configurations (Chainlink, n.d.-a; Chainlink, n.d.-b; LinkWell Nodes, n.d. ).

Timelock smart contracts are a specific kind of smart contract that imposes a delay or time-based limitation on the execution of particular actions or transactions. They are sometimes referred to as time-based or delayed contracts. In contrast to ordinary smart contracts, which operate immediately, timelock contracts need a waiting period before a certain action may be taken. Timelock contracts are usually programmed with a predefined delay that prevents users from doing the specified activity. This waiting period is an important safeguard against malevolent activity and rash decisions, as well as a means of maintaining security and transparency.

### **Smart Contract Design:**

- **Parameters:** Define the allowance amount, the time of release and the intended recipient
- **Time Lock Mechanism:** Implement a time lock that releases funds at predetermined intervals. The contract should ensure that funds are only accessible according to the specified schedule.

One way this solution could be used is for parents to set time-locked transactions for their children, who are university students, for their daily/weekly/monthly allowance. This will help teach financial management and a way to not forget to do the transactions if the parents are busy.

The time-locked smart contract application will manage and automate the allowance distribution process, providing transparency and reducing administrative overhead. Parents

can set up allowances with specific conditions and schedules, while students receive their funds as intended, encouraging financial responsibility.

## 2.0 Policy/Business Rules

### a) Parents (with Administrative Privileges)

**Role:** Generally, the system allows parents to manage the scheduling and execution of allowances to their grown-up children. The system also allows parents to oversee the entire blockchain application, monitor the transaction integrity and handle potential problems.

#### **Rules/Permissions:**

- The system shall require parents to register first if they have not registered.
- The system shall require parents to sign in before they can perform any actions.
- The system shall allow parents to queue allowances, which means setting up future allowances by specifying the amount, timing and recipient details.
- Before the parent can queue the transaction, the parent will be authenticated to ensure that the parent is authorized to create the transaction and has the necessary funds.
- The system shall allow parents to adjust the terms or cancel scheduled allowances before they are executed.
- If the parent cancels a time-locked transaction, the fund will be returned back to the parent.
- The system shall allow parents to adjust the time, recipient and/or the fund to be sent.
- If the parent increases the fund to be sent to his/her child, the additional fund needed will be sent from the parent to the smart contract.
- If the parent decides to reduce the fund to be sent to his/her child, some of the fund will be withdrawn from the smart contract back to the parent.
- The system shall allow parents to access detailed information on all queued, cancelled and completed transactions, which ensures transparency and control.

### b) Students

**Role :** These refer to students, who are currently studying or already complete study at the university, and act as the recipients of the allowances managed by their parents.

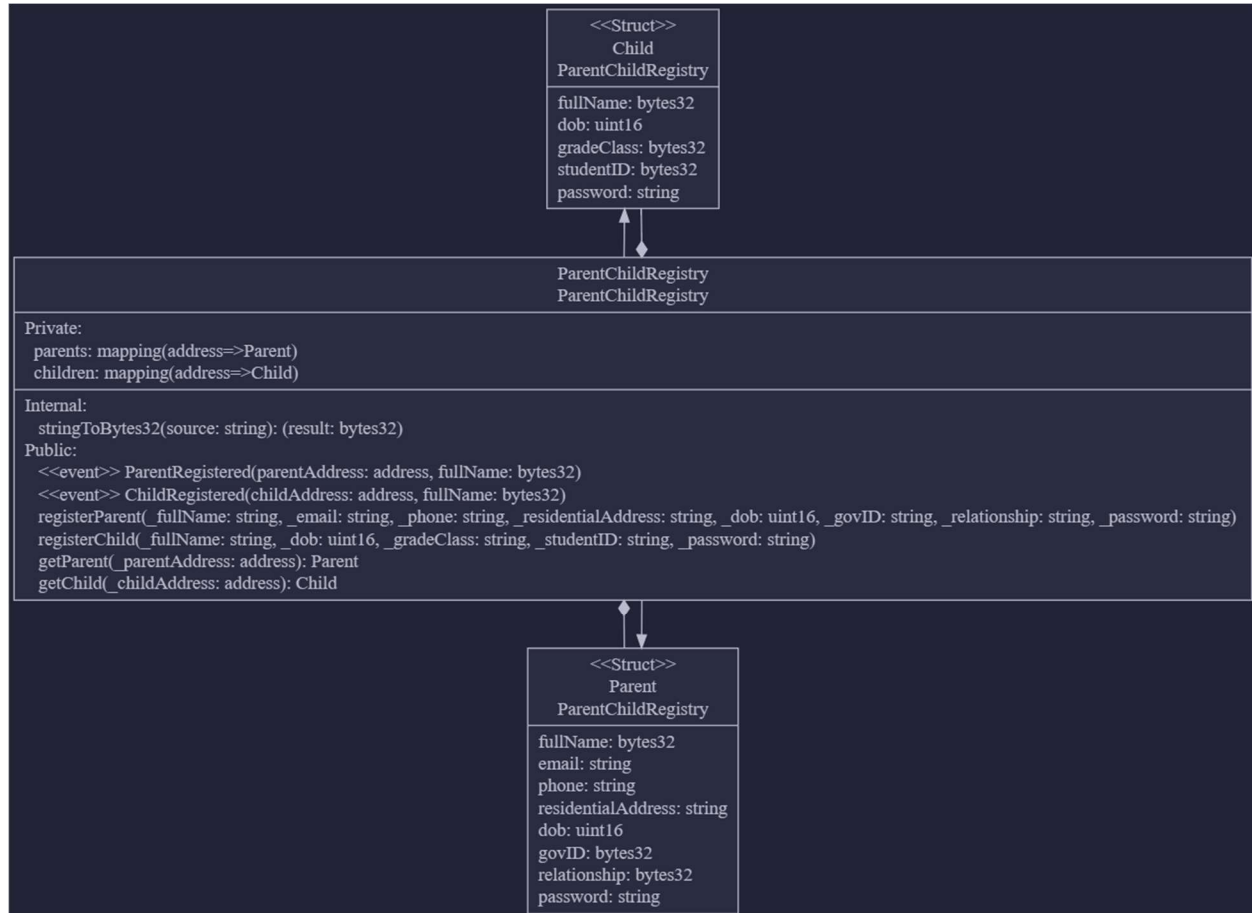
#### **Rules/Permissions:**

- The system shall require students to register first if they have not registered.

- The system shall require students to sign in before they can perform any actions.
- The system allows students to access details of their scheduled allowances such as the amount and timing.
- The system allows students to view the history of their received allowances.
- The system allows students to automatically receive funds according to the predefined schedule set by their parents.
- The system allows students to submit requests to their parents for reallocating funds to different purposes.

## 3.0 Contract Diagram

### Smart Contract 1: For Registration and Login of Parent and Child/Student



**Smart Contract 2:** For obtaining the latest time off-chain via Chainlink Oracle from an external api (worldtimeapi.org)

```
LinkWellUint256ConsumerContractExample
Timestampv5

Private:
  oracleAddress: address
  jobId: bytes32
  fee: uint256
Public:
  response: uint256

Public:
  <<event>> RequestFulfilled(requestId: bytes32, response: uint256)
  constructor()
  request()
  fulfill(requestId: bytes32, data: uint256)
  setOracleAddress(_oracleAddress: address)
  getOracleAddress(): address
  setJobId(_jobId: string)
  getJobId(): string
  setFeeInJuels(_feeInJuels: uint256)
  setFeeInHundredthsOfLink(_feeInHundredthsOfLink: uint256)
  getFeeInHundredthsOfLink(): uint256
  withdrawLink()
```

This contract will be deployed on Ethereum Sepolia Testnet and registered under Chainlink's Upkeep service. (Note: Actually, there are many other smart contracts associated with this contract. For simplicity, only the smart contract that needs its code to be edited to retrieve the current time off-chain is shown here.). The contract had to be modified to select the correct Request Type, Operator Contract, Job ID and api and to indicate which data in the returned JSON object is the desired one, for example the epoch timestamp. Therefore, we had to choose the oracle and Job ID that would retrieve the timestamp as a uint256 data type via HTTP GET request.

Using an oracle service for time data provides more accuracy and consistency than relying on block.timestamp, which can be affected by miner manipulation. Oracles can fetch time from reliable external sources, ensuring that the DApp operates based on precise and consistent time standards.

References for the Oracle Service Code and Guidelines:

- Chainlink (n.d.-b)
- Infura (2024)
- LinkWellNodes (n.d.)

- LinkWell Nodes (n.d.)

References for the Chainlink Upkeep Service Guidelines:

- Chainlink (n.d.-a)

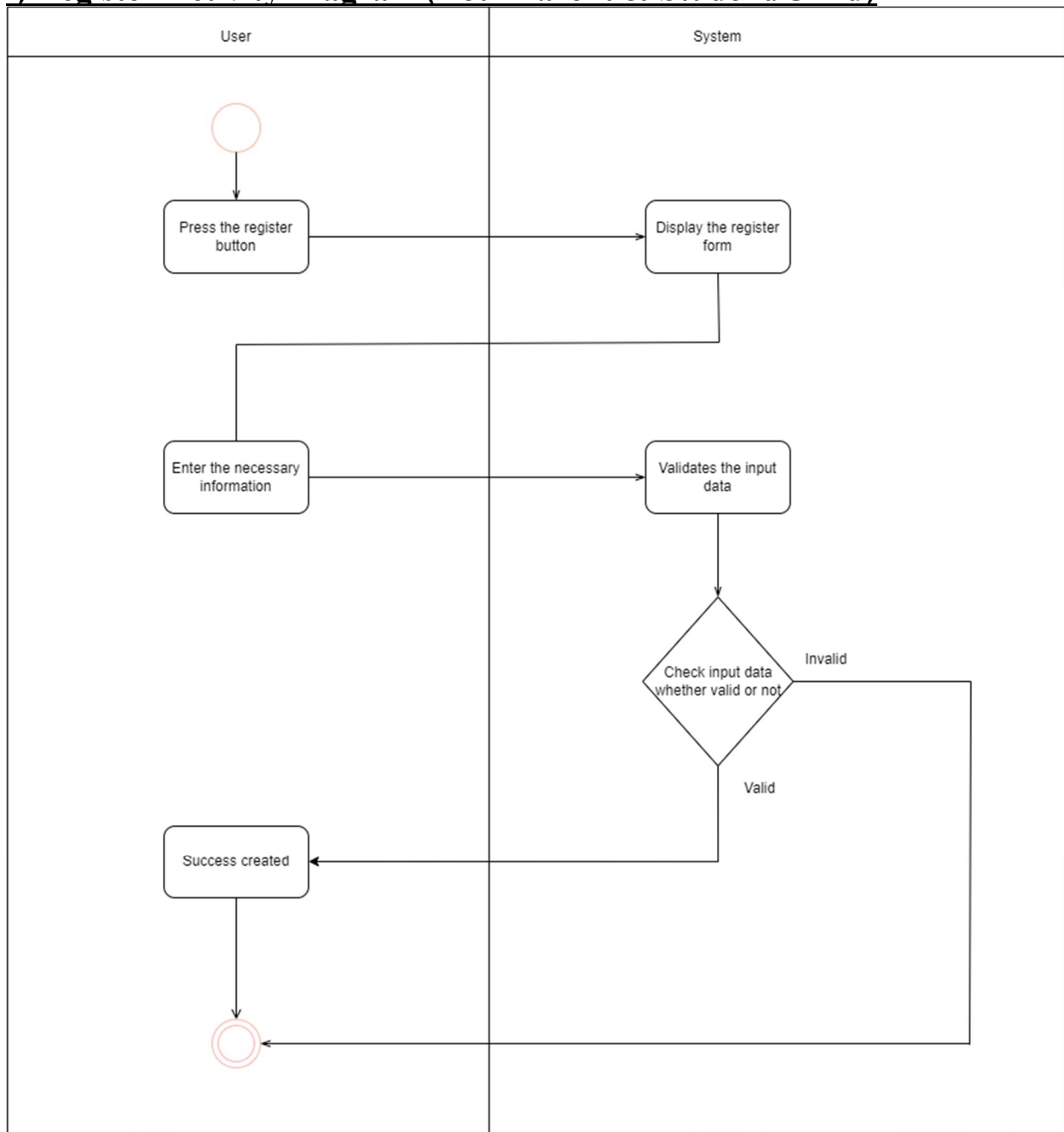
### Smart Contract 3: Timelock Logic for Parent and Child



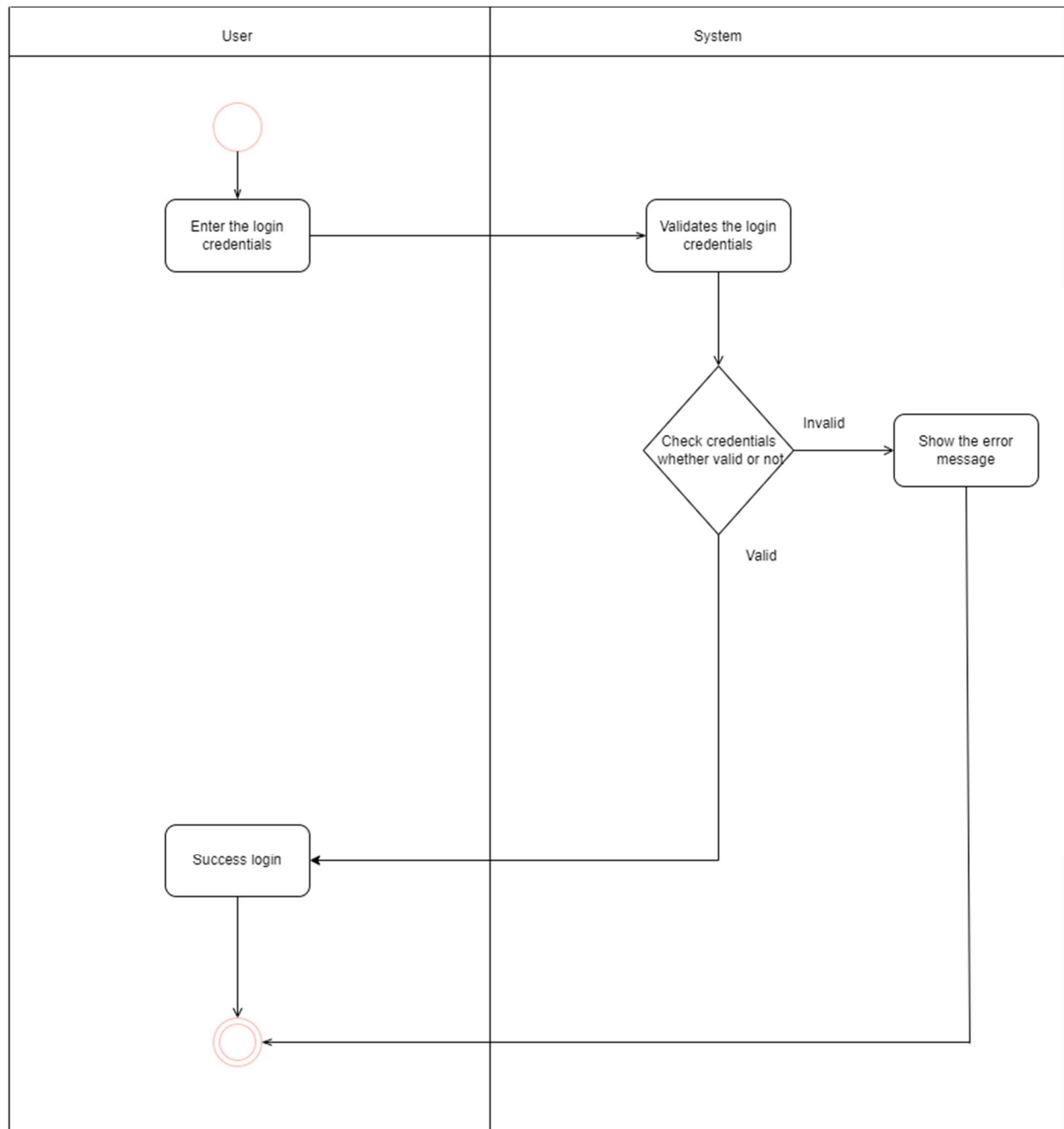


## 4.0 Activity Diagram

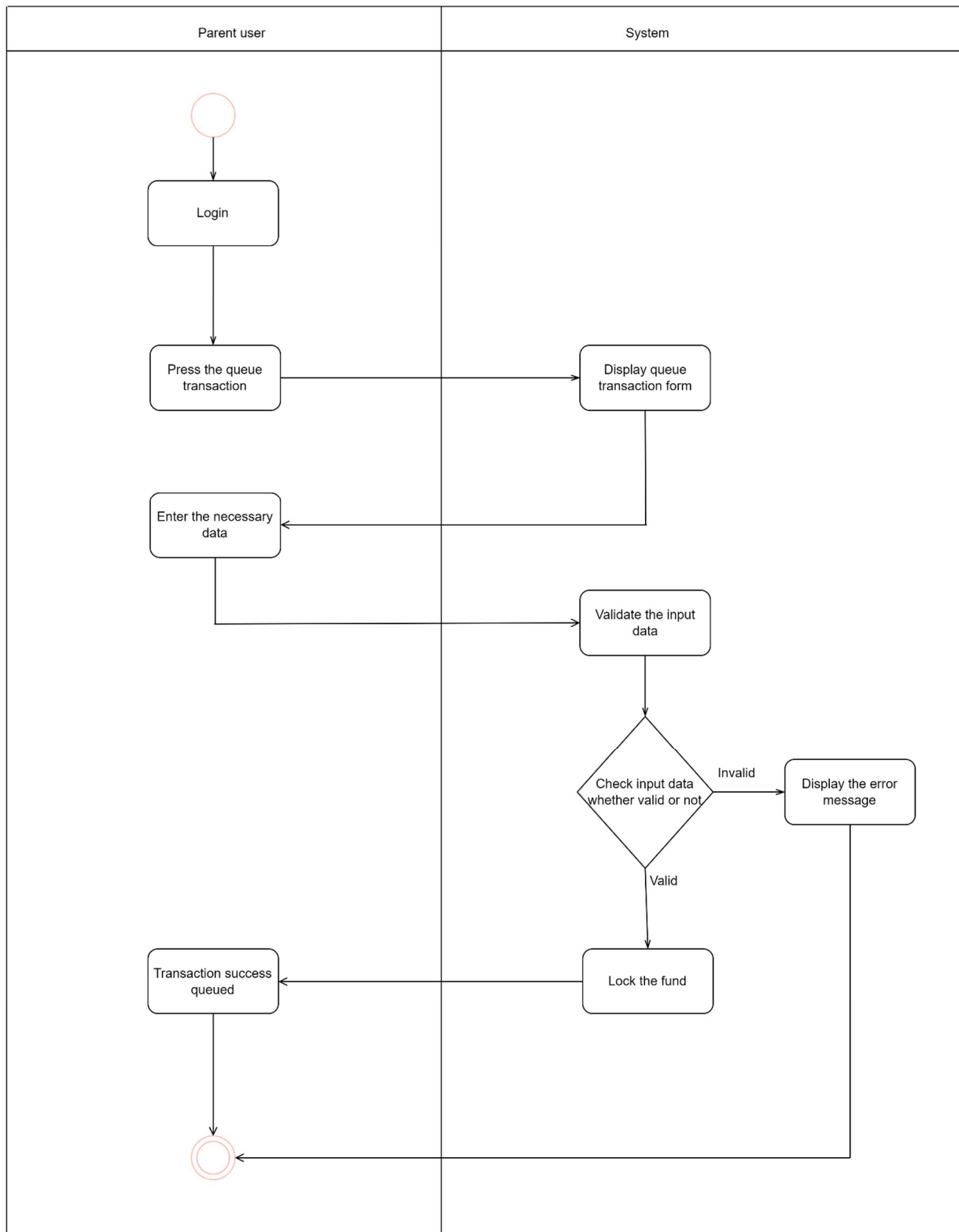
### i) Register Activity Diagram (Both Parent & Student/Child)



## ii) Login Activity Diagram

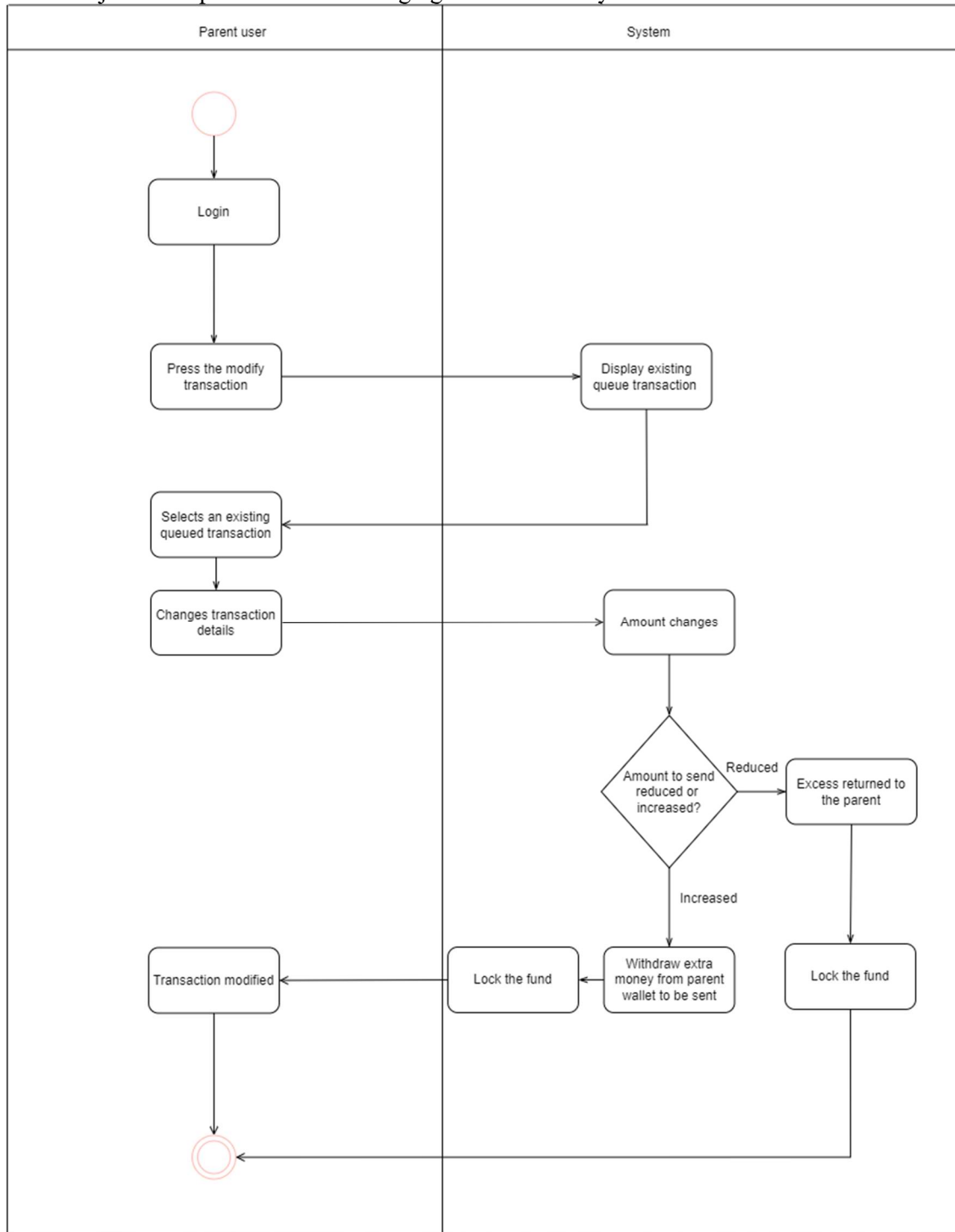


### iii) Queue Transaction Activity Diagram

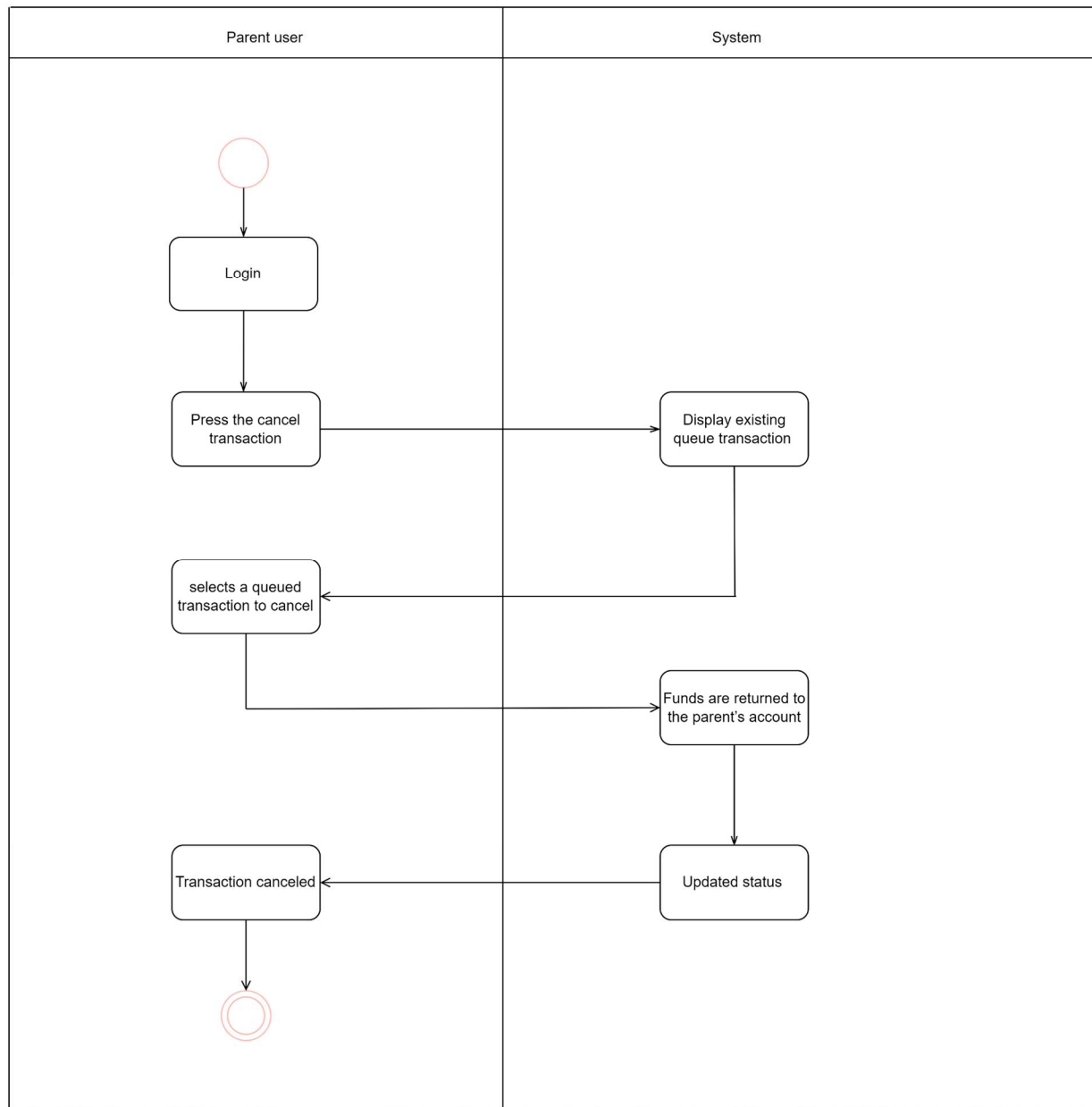


#### iv) Modify Transaction Activity Diagram

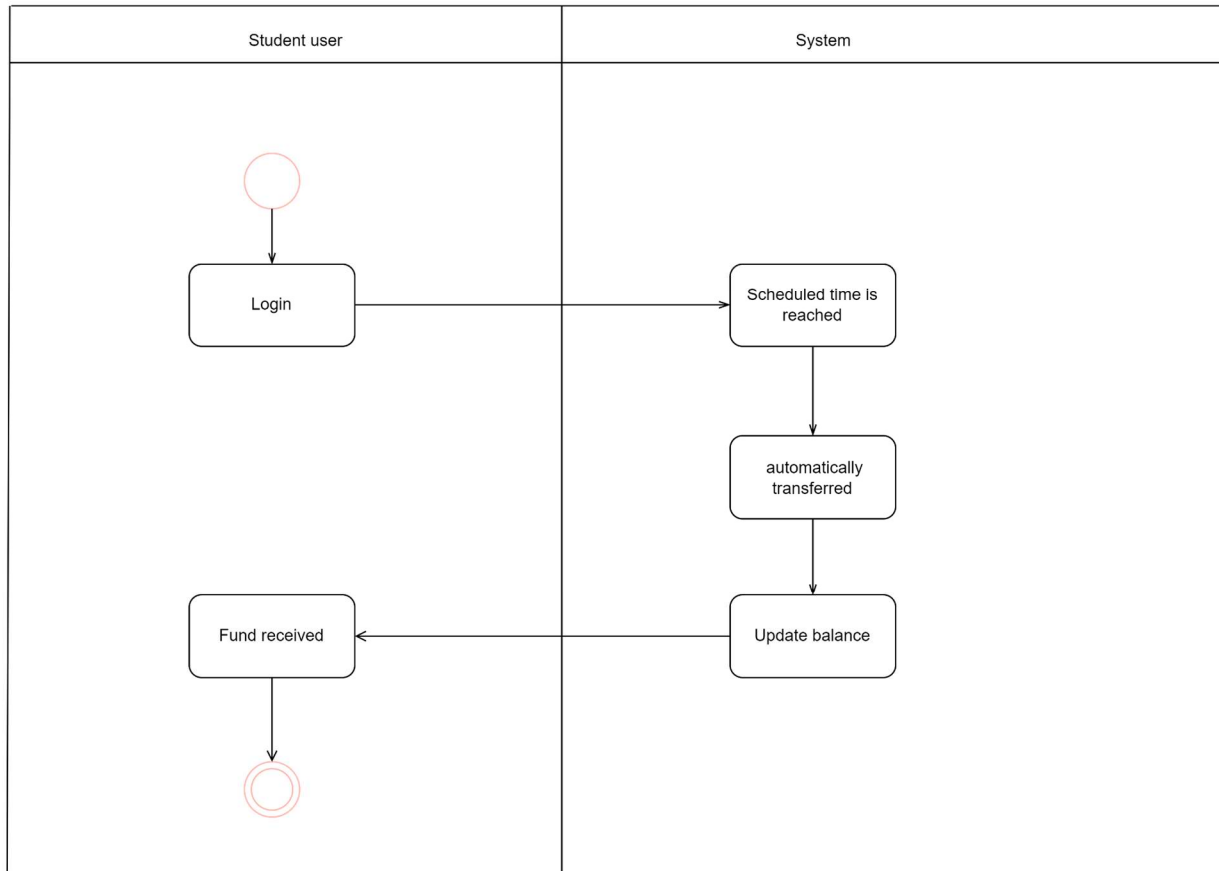
Note that our program actually allows the modification of the fund transferred, recipient and/or time. Below is just a sample scenario of changing the amount only.



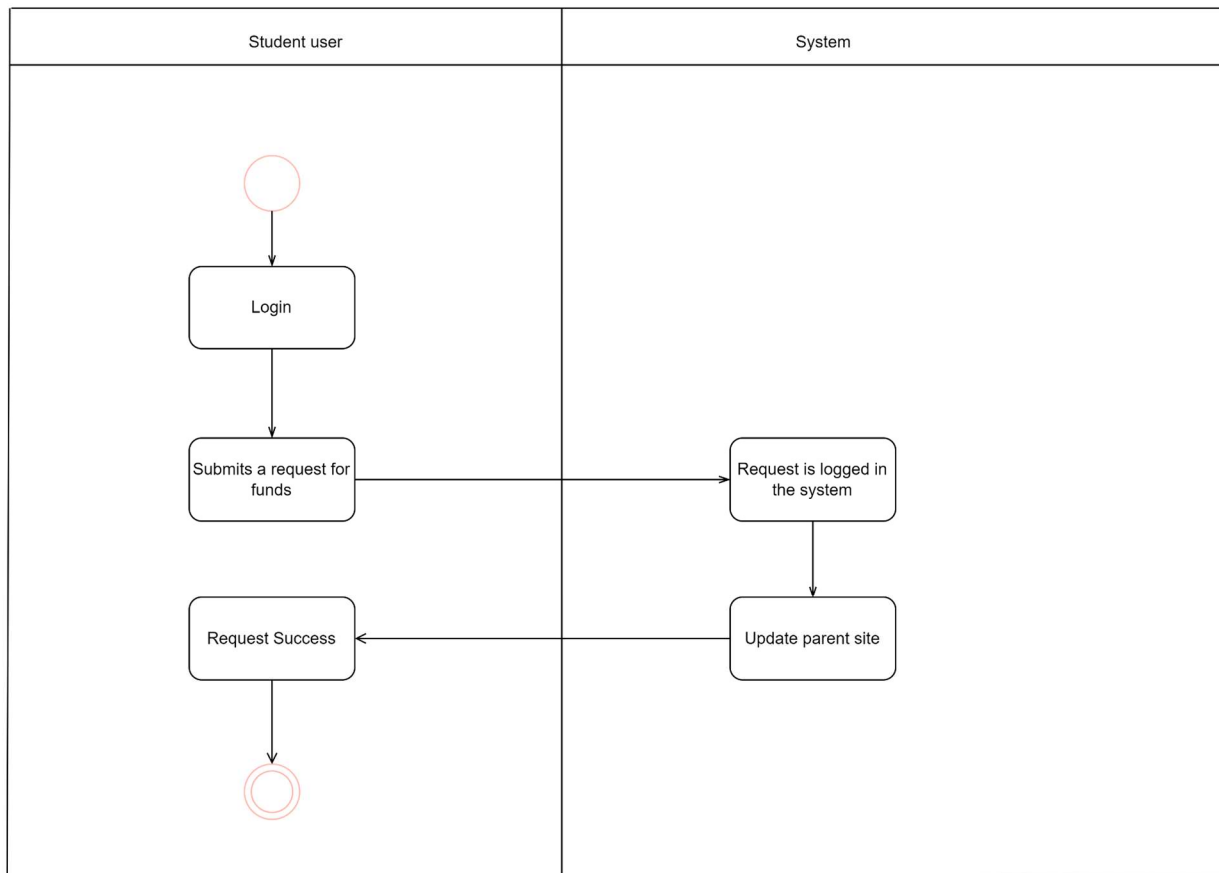
### v) Cancel Transaction Activity Diagram



## vi) Receive Fund Activity Diagram

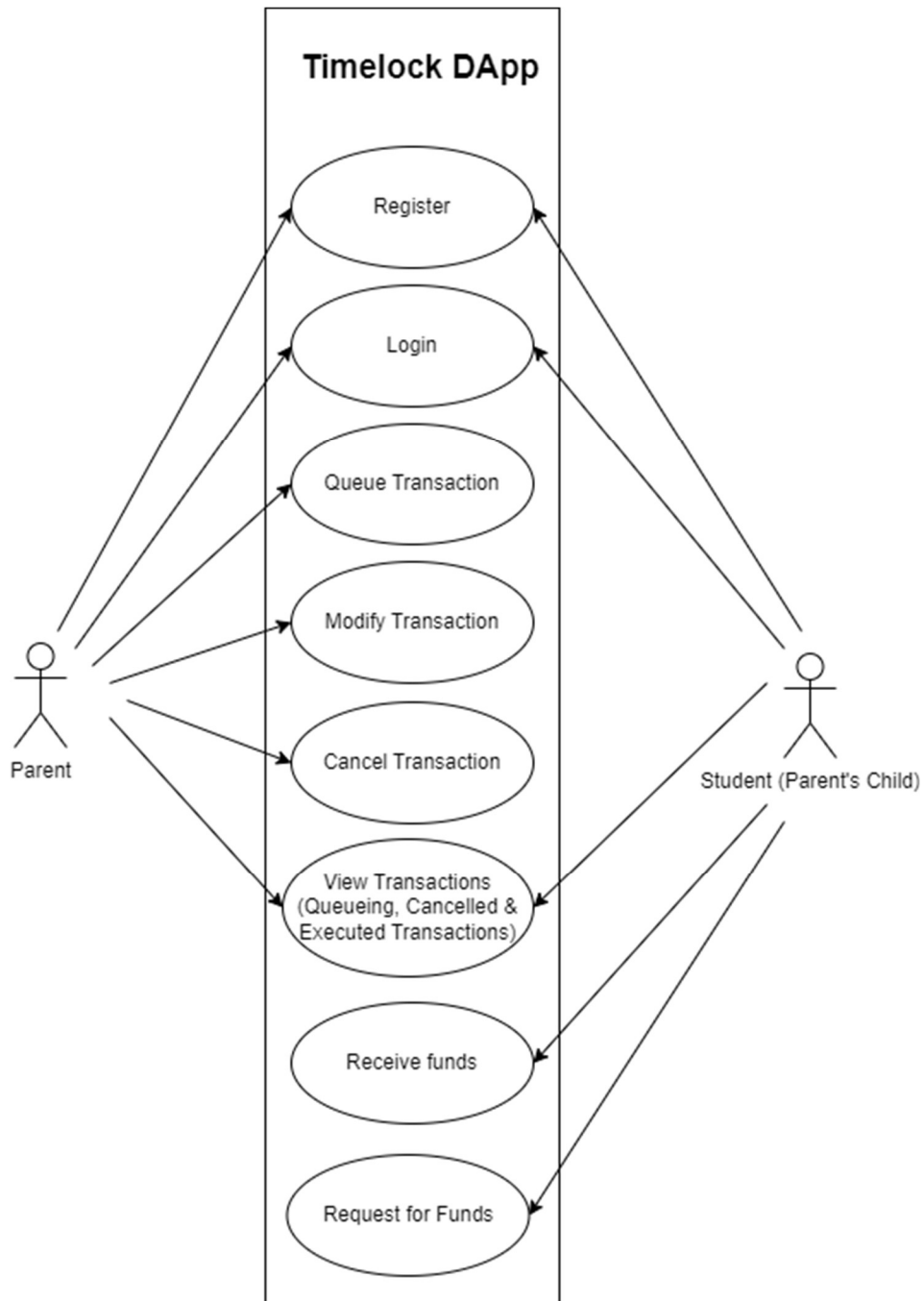


### vii) Request Fund Activity Diagram



## 5.0 Use Case Diagram

The figure below shows the various functions / tasks that can be performed by each kind of user (Parent and Student).





## **Category 1: Parent and Student Use Case**

### **Register (Parent and student)**

Both parents and students must first register within the system to gain access. The registration process ensures that each user has a unique identity and account. For parents, registration is a prerequisite to managing allowances, while for students, it is required to receive funds. The system will validate the user's information and store it securely, allowing each user to log in and interact with the platform based on their designated roles. This step is crucial to ensure that only authorized individuals (parents and students) can use the system's features.

### **Login (Parent and student)**

Once registered, both parents and students need to log in to access their respective dashboards and features. The login process involves verifying the user's credentials (such as username and password) to authenticate their identity. Parents are redirected to a dashboard where they can view and manage allowances, while students can view their upcoming or past allowances. Successful login is required for any subsequent actions, such as queuing transactions for parents or receiving funds for students, ensuring that access to the system is secure and controlled.

### **View Transactions (Queued, Canceled, Executed) (Parent and Student)**

Both parents and students can view the history and status of all their transactions. Parents can see the allowances they have queued (scheduled for the future), canceled (withdrawn before execution), and executed (completed allowances where funds were disbursed to the student). Students have similar access to view queued allowances, which they will receive in the future, as well as canceled or executed transactions. This feature provides transparency for both parties, allowing them to track the financial flow and history of the allowances at any time.

## **Category 2: Parent-Specific Use Case**

### **Queue Transaction (Parent Only)**

Only parents have the ability to queue future transactions (or allowances) for their children. This involves setting up a scheduled allowance by specifying the amount, the recipient (student), the purpose (e.g., education or savings), and the timing (release date). The funds are locked in the smart contract when the transaction is queued, and the allowance will be automatically disbursed to the student when the release time is reached. Parents use this feature to manage periodic or one-time financial support to guide their children's spending habits.

### **Modify Transaction (Parent Only)**

Parents are allowed to modify the details of any scheduled (queued) transactions before they are executed. They can adjust the amount of the allowance, the recipient, or the scheduled release time. If the parent increases the allowance amount, they must add more funds, and if they reduce it, excess funds are returned to their account. This flexibility allows parents to adjust the allowance according to any changes in their financial situation or their child's needs, ensuring dynamic control over the queued allowances.

### **Cancel Transaction (Parent Only)**

Parents can cancel a queued transaction at any time before it is executed, effectively stopping the scheduled transfer of funds. When a parent cancels a transaction, the funds that were locked in the smart contract are returned to the parent's account, and the transaction is marked as canceled. This feature ensures that parents retain control over their finances in case they change their mind or if circumstances require the allowance to be halted. The cancellation is logged in both the parent's and the student's transaction history for transparency.

## **Category 3: Student-Specific Use Case**

### **Receive Funds (Student Only)**

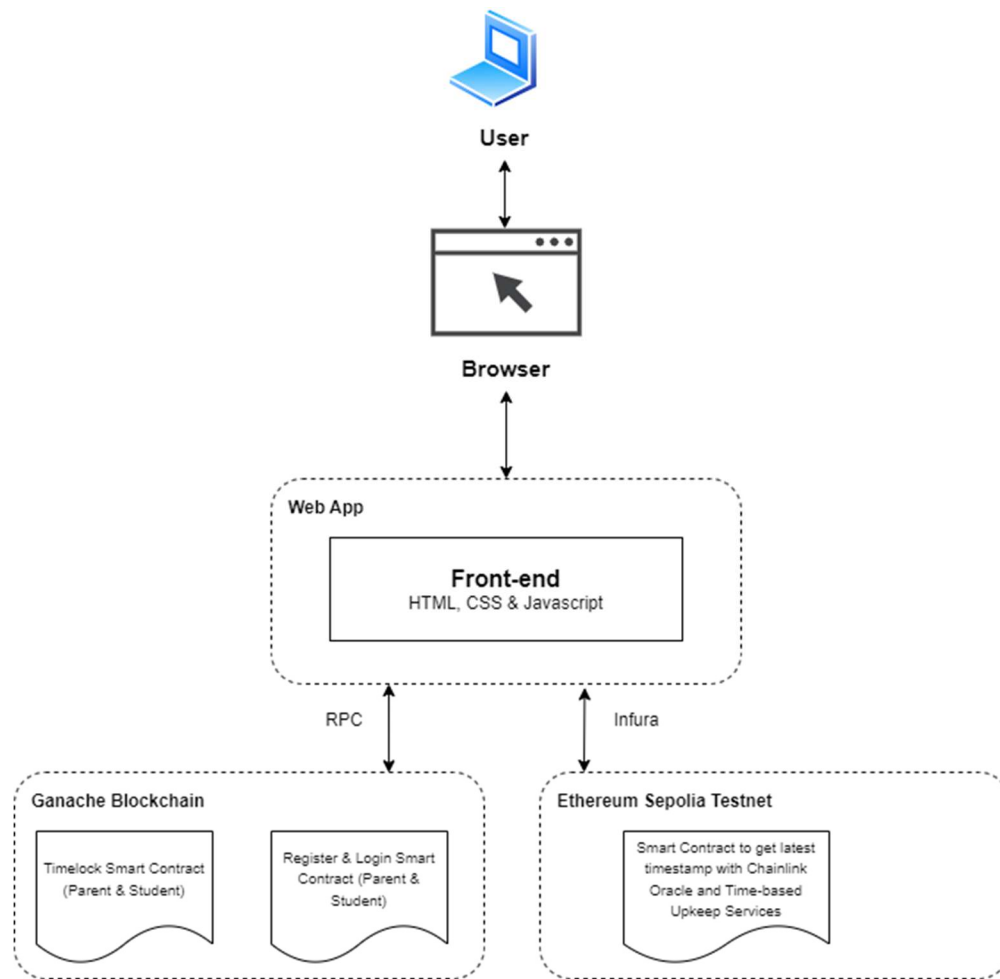
Students are the only ones who can receive the funds, and this occurs automatically according to the schedule set by the parent. Once the release time of a queued allowance is reached, the funds are transferred from the smart contract to the student's account. The student can then use the funds for the specified purpose (e.g., education or entertainment). This process is automated, ensuring that students receive their allowances without manual intervention once it is queued by the parent.

### **Request Funds (Student Only)**

Students can submit requests to their parents for funds or reallocation of existing allowances. This request doesn't automatically alter the transaction but is logged in the system as a communication between the student and the parent. It allows students to indicate a change in their needs, such as requesting an allowance for a different purpose (e.g., shifting funds from entertainment to education). The parent can review these requests and decide whether to approve or adjust the allowances accordingly. This feature fosters financial dialogue between students and parents.

## 6.0 System Architecture/Design Structure

The figure below shows the proposed system architecture for the Timelock DApp.



On the Ethereum Sepolia testnet, there will be a smart contract designed to obtain the latest timestamp from an external API (worldtimeapi). Since the data type for timestamp would be uint256, the contract will be performing HTTP GET request for uint256 data. This is made possible by using the Chainlink Oracle provided by Link Well Nodes. To automate retrieval of time at regular intervals, this smart contract will be under Chainlink's Time-based Upkeep service. The Upkeep service will periodically trigger a function within the smart contract at regular intervals to fetch the most recent timestamp from the specified API.

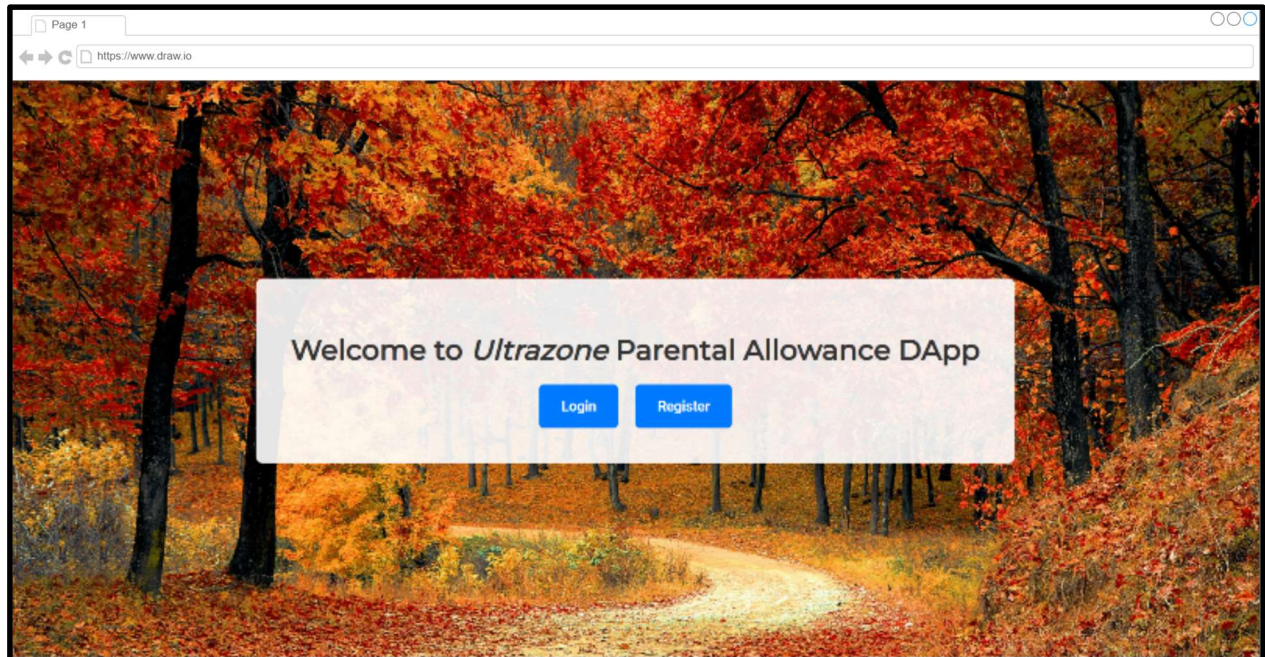
For the Ganache local blockchain, there will be a smart contract that contains the timelock logic and operations related to transactions. Another smart contract will be used for login and registration of users, facilitating user account creation.

A Node.js server will be used to access both the Ganache blockchain and the Ethereum

Sepolia testnet blockchain. There will be one event listener for the Ethereum Sepolia testnet blockchain to detect the emitted event “RequestFulfilled” that emits the latest timestamp off-chain, which will be sent to the Ganache Blockchain.

## 7.0 UI Design

### 7.1 Proposed UI Design for the Home Page



The user interface (UI) in the image is simple and clean, designed for an app called "Ultrazone Parental Allowance DApp."

In the middle of the screen, there's a message that says, "Welcome to *Ultrazone* Parental Allowance DApp", with *Ultrazone* in italic to highlight the name of the app. Below the message, there are two big buttons: **Login** and **Register**. These buttons are blue with white text, making them easy to see and click.

The whole design uses blue and white, which gives a calm and trustworthy feeling. The layout is very straightforward, focusing on two main actions—logging in or signing up—without any extra distractions.

This UI is designed to be simple and easy to use, likely aiming to make it quick for parents to manage allowances for their kids.

## 7.2 Proposed UI Design for Register Interface

Page 1  
https://www.draw.io

### UltraZone Parental Allowance DApp

Parent/Guardian Child

#### Parent/Guardian Registration

Full Name

Email Address

Phone Number (Optional)

Address

mm/dd/yyyy

Government ID (e.g., Passport)

Connect MetaMask No account connected

Password

Relationship to Child

Register as Parent/Guardian

The UI in the image is a registration form for the "Ultrazone Parental Allowance DApp". It's designed for parents or guardians to sign up.

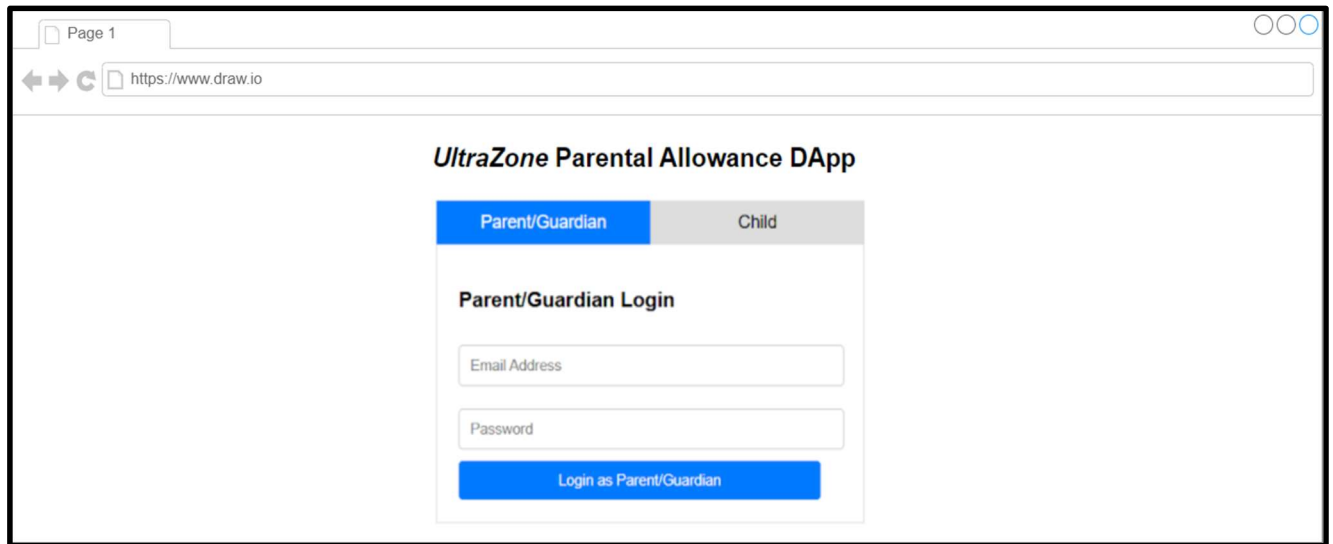
At the top, there are two tabs which are **Parent/Guardian** (active) and **Child** to let users choose who they are registering as. The form asks for basic information like:

- Full name,
- Email address,
- Phone number (optional),
- Address,
- Date of birth, and
- Government ID (e.g., passport).

There's also a **Connect MetaMask** button, allowing users to link their MetaMask wallet. Below that, users can set a password and choose their relationship to the child from a drop-down list.

Finally, there's a **Register as Parent/Guardian** button to complete the registration. The design is simple, with clear fields and a clean look, making it easy to fill out.

## 7.3 Proposed UI Design for Login Interface



The UI in this image is a login page for the "Ultrazone Parental Allowance DApp". It has a clean and simple design.

At the top, there are two tabs which are **Parent/Guardian** (active) and **Child** to allow users to choose the appropriate login type.

The form has two fields:

- **Email Address:** For the parent or guardian's email.
- **Password:** For entering the account password.

Below the fields, there is a blue button labeled **Login as Parent/Guardian** to submit the login details.

The overall design is straightforward and user-friendly, focusing on easy navigation and quick access for logging in. The blue and white colors make the page look clean and professional.

## 7.4 Proposed UI Design for the Parent Interface

Page 1

https://www.draw.io

### UltraZone Parental Allowance DApp

**Parent** Queue Transaction Modify Transaction Cancel Transaction View Transactions

#### Queue Transaction

Child's Wallet Address:

Allowance Amount:

Currency:

Wei

Release Time:

dd/mm/yyyy --:--:--

Set Allowance Clear

#### Modify Transaction

Transaction ID:

Find Transaction Clear

#### Cancel Transaction

Transaction ID:

Cancel Transaction Clear

#### View Transactions

View All Transactions Clear

#### View Child Requests

View Requests Clear

Connect Metamask Clear

Connect to Smart Contract Clear

The UI in the image is for the "Ultrazone Parental Allowance DApp", it is designed to help parents manage allowances.

- Queue Transaction: Parents can enter the child's wallet address, allowance amount, choose currency, and set a release time. There's a button to confirm the allowance or clear the form.
- Modify Transaction: Input a transaction ID to find and change an existing allowance.
- Cancel Transaction: Enter a transaction ID to cancel it.

- View Transactions: Click to see all past transactions.
- View Child Requests: Click to see all child requests

On the right side, buttons allow users to connect their MetaMask wallet and link to the Smart Contract for blockchain features.

The overall design is clean, simple, and focused on easy allowance management.

## 7.5 Proposed UI Design for the Student/Child Interface

The UI shown in the image is for the "Ultrazone Parental Allowance DApp" designed for students. Here's a simple breakdown:

- Current Balance: Displays the student's current balance in their wallet.
- Transaction History: A button to view past transactions related to their allowance.
- Request Funds:



- The student can enter their wallet address.
  - Specify the allowance amount.
  - Select the currency.
  - Set the release time for when they want the funds.
  - There are two buttons: one to request the allowance and another to clear the form.
  - View My Request: Displays all the students' requests
- On the right side, similar to the parent's UI, there are buttons to:
- Connect MetaMask: Link the DApp to the MetaMask wallet.
  - Connect to Smart Contract: For connecting to the smart contract for transactions.

The layout is clean and simple, focusing on key actions like viewing balances, transaction history, and requesting funds.

## **7.6 Proposed UI Design for Approve Transaction Interface**

### Admin Actions

Register as Admin

Click below to see the latest admin information:

Check Admins

### Admin Information

Admin Role	Admin Address
Admin 1	0x00
Admin 2	0x00

Approve

Deny

This UI is a simple interface for managing admins. It has a section called "Admin Actions" with two buttons.

- "Register as Admin": lets a user sign up as an admin.
- "Check Admins": shows the latest information about the current admins.

Below that, there's a table labeled "Admin Information," which lists the roles of two admins (Admin 1 and Admin 2) and their addresses. The addresses are in a format starting with "0x," which is commonly used in blockchain systems. Right now, both admin addresses are just placeholders filled with zeroes, meaning no real admins are shown yet.

At the bottom, there are two more buttons: "Approve" and "Deny." These are likely for approving or denying admin actions or requests. The design is simple and focuses on basic admin tasks.

## 8.0 Limitations/challenges faced

- Delay in Execution: Transactions are executed only after a set delay. This time restriction can create challenges for users who need immediate execution of transactions, reducing flexibility.
- Trust in Oracle & Upkeep service: Since the DApp relies on an oracle service and an Upkeep service to trigger the execution after the time-lock period, it is subject to the reliability and security of that service. An unreliable oracle could lead to delayed or failed executions.
- Deployment in both Ganache and Testnet: If we had enough ETH for Ethereum Sepolia testnet in our MetaMask wallets, we would have just deployed on the testnet itself to simplify deployment.
- User Experience: Users may find the waiting period frustrating if they are not used to time delays. There is also a risk of users forgetting about scheduled transactions if the UI does not adequately remind them or offer updates.
- Interoperability with Other Smart Contracts: Since the DApp relies on external smart contracts, changes in those contracts during the time-lock period can result in transaction failures or mismatched data when execution happens.

## 9.0 References

- Chainlink. (n.d.-a). Chainlink Documentation | Chainlink Documentation. Chainlink Documentation. <https://docs.chain.link/chainlink-automation/guides/job-scheduler>
- Chainlink. (n.d.-b). *Testnet Oracles* | *Chainlink Documentation*. Chainlink Documentation. <https://docs.chain.link/any-api/testnet-oracles>
- Edge, M. (2021, October 1). *Teaching your kids about money and good financial habits*. Merrill Edge. <https://www.merrilledge.com/article/teaching-kids-about-money-financial-responsibility>
- Infura. (2024, September 17). *Supported networks* | *INFURA*. <https://docs.infura.io/api/networks/ethereum/choose-a-network>
- LinkWell Nodes. (n.d.). *ChainLink Oracle Smart Contract Examples (Testnets)* | *LinkWell Nodes*. <https://docs.linkwellnodes.io/services/direct-request-jobs/testnets/Ethereum-Sepolia-Testnet-Jobs>
- LinkWellNodes. (n.d.). *Documentation/docs/services/direct-request-jobs/testnets/Ethereum-Sepolia/uint256/uint256.sol at main · LinkWellNodes/Documentation*. GitHub. <https://github.com/LinkWellNodes/Documentation/blob/main/docs/services/direct-request-jobs/testnets/Ethereum-Sepolia/uint256/uint256.sol#L23-L28>
- Solidity by example*. (n.d.). <https://solidity-by-example.org/app/time-lock/>
- Smartcontractkit. (n.d.). *smart-contract-examples/timelocked-contracts/src/Contract.sol at main · smartcontractkit/smart-contract-examples*. GitHub. <https://github.com/smartcontractkit/smart-contract-examples/blob/main/timelocked-contracts/src/Contract.sol>
- W, C., W, C., & W, C. (2023, November 2). *Teaching Kids about Money: Instilling Financial Responsibility*. Indepth Research Institute - Transforming People and Organizations in Africa Since 2003. <https://blog.indepthresearch.org/teaching-kids-about-money-instilling-financial-responsibility/>
- Teaching Children about Financial Responsibilities* | *AmMetLife*. (n.d.). <https://www.ammethlife.com/future/blog/for-loved-ones/teaching-children-about-financial-responsibilities/>