

American International University-Bangladesh (AIUB)  
**Department of Computer Science  
Faculty of Science & Technology (FST)**

**BlockPay Money Transaction System (Blockchain)**

A Software Requirement Engineering Project Submitted

By

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Semester: Spring Spring\_22\_23** | | **Section: F** | **Group Number: 03** | |
| SN | Student Name | Student ID | Contribution (CO1+CO2) | Individual Marks |
| 10 | NABIL, HADIUR RAHMAN | 20-42095-1 | 40% |  |
| 15 | JOYA, NABILA CHOWDHURY | 20-42268-1 | 20% |  |
| 16 | CH., RASHIQUE HABIB | 20-42269-1 | 20% |  |
| 29 | NETU, KHAN NUSHRAT SULTANA | 20-43191-1 | 20% |  |
|  |  |  |  |  |

The project will be Evaluated for the following Course Outcomes

|  |  |  |
| --- | --- | --- |
| Evaluation Criteria | Total Marks (50) | |
|  | |
| Introduction, Format, Submission, Defense | [10 Marks] |  |
| System Overall Description & Functional Requirements | [10 Marks] |  |
| System Quality Attributes and Project Requirements | [10 Marks] |  |
| UML and E-R Diagram with Data Dictionary | [10 Marks] |  |
| UI/UX Prototyping | [10 Marks] |  |

Software Requirements Specification

for

< BlockPay Money Transaction System >

Version 1.0 approved

Prepared by < NABIL, HADIUR RAHMAN, NETU, KHAN NUSHRAT SULTANA, CH., RASHIQUE HABIB, JOYA, NABILA CHOWDHURY >

**<American International University-Bangladesh >**

<30.04.23 >

Table of Contents

[Revision History 3](#_Toc133711664)

[1. Introduction 4](#_Toc133711665)

[1.1 Purpose 4](#_Toc133711666)

[1.2 Document Conventions 4](#_Toc133711667)

[1.3 Intended Audience and Reading Suggestions 5](#_Toc133711668)

[1.3.1 Different types of readers 5](#_Toc133711669)

[1.3.2 Sequence for reading the document 5](#_Toc133711670)

[1.4 References 8](#_Toc133711671)

[2. Overall Description 9](#_Toc133711672)

[2.1 Product Perspective 9](#_Toc133711673)

[2.2 Product Functions 11](#_Toc133711674)

[2.3 User Classes and Characteristics 12](#_Toc133711675)

[2.4 Operating Environment 12](#_Toc133711676)

[2.5 Design and Implementation Constraints 13](#_Toc133711677)

[2.6 User Documentation 13](#_Toc133711678)

[3. System Requirements 14](#_Toc133711679)

[3.1 System Features 14](#_Toc133711680)

[3.2 Non-Functional/Quality Requirements 19](#_Toc133711681)

[3.3 Project Requirements 21](#_Toc133711682)

[4. Design and Interface Requirements 22](#_Toc133711683)

[4.1 UML Diagrams 22](#_Toc133711684)

[4.2 Data Dictionary 25](#_Toc133711685)

[4.3 UI/UX Design Specification 26](#_Toc133711686)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

This software requirement specification (SRS) document's goal is to outline the specifications for the "BlockPay" blockchain-based payment system. The software system used to enable safe, decentralized, and incredibly quick financial transactions using blockchain technology is the object detailed in this document. The system and all of its subsystems are covered by this SRS. Without the need of third-party organizations, the software system described in this article tries to provide a very safe and reliable way to execute financial transactions. It will give users access to a decentralized network that makes it easier for people or organizations to transfer money without the need for a central authority. The system's main goals are to protect transactions' privacy, reliability, and accessibility as well as to stop fraud. The key benefits of the software system include improved security, reduced transaction costs, and more transparency. The software application is in line with the business's goals of encouraging financial inclusion and leveling the playing field for people and companies. If there is a separate goal and scope document, this SRS document does not repeat its contents. However, it enhances the project's vision and scope by outlining the software requirements necessary to accomplish the stated objectives and goals.

## Document Conventions

This SRS document is done using the IEEE format and the project is a Blockchain-based money transaction system named “BlockPay”.

|  |  |
| --- | --- |
| Heading and Subheading | This part of the SRS describes the conventions used in the document, including font type which is Times New Roman and size which is 12, and space 1.0. These conventions ensure consistency and clarity throughout the document, making it easier to understand and evaluate the requirements. The size of the main heading is 18 and the subheading size is 14. |
| SRS | Software Requirement Specification |
| IEEE | [Institute of Electrical and Electronics Engineers](https://en.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers) |
| DeFi | Decentralized finance |

## Intended Audience and Reading Suggestions

### Different types of readers

**Developers:** This paper provides in-depth details regarding the data model, user interface, and system testing and validation method, as well as the functional and non-functional requirements of the "BlockPay" system. The information in this paper can be used by developers to design, create, test, and correct any errors in the "BlockPay" program.

**Project Managers:** The SRS document can be used as a guide for project managers for planning projects, scheduling them, and allocating resources. This document can be used by project managers to understand the project specifications and guarantee that the development team completely fulfills the project goals and objectives.

**Business analysts:** This SRS document informs business analysts about the "BlockPay" system's business needs and overall business strategy.

**Marketing Staff:** SRS documents will include information on the system's advantages, additional promotional personnel, and market potential that support the system's marketing-related activities.

**Users:** The "BlockPay" system's capabilities, restrictions, and user interface are all covered in detail in this SRS document, which also ensures user interaction. For example, locating features where consumers would expect to find them.

**Testers**: This SRS document contains information on the many kinds of tests that must be carried out to confirm the system's functionality. This knowledge can be used by testers to create ideal test cases and guarantee that the system performs as planned.

**Documentation Writers**: The paper may be reviewed while creating user guides, system functional specifications, system documentation, and other technical and theoretical materials. This document can be used by those who develop documentation to make sure that it appropriately reflects system functionality.

.

### Sequence for reading the document

**Overview**: The SRS document for the Blockchain-based/Decentralized Money Transaction System (SRS) is a comprehensive manual that specifies the requirements for a secure and reliable way to conduct financial transactions without the involvement of third parties. With the help of blockchain technology, the "BlockPay" system aims to offer customers a decentralized and open platform for financial transactions. Increased security, lower transaction costs, and improved transparency without the support of a third party are the main characteristics of this system.

**System Architecture:** The architecture is based on a decentralized peer-to-peer network, in which nodes converse with one another to verify transactions and keep track of a distributed record. The blockchain network, the user interface, the smart contracts, and other elements are included in the system architecture. The "BlockPay" system, which offers a safe and protection against unauthorized changes for recording all transactions, is built on the blockchain network. The front-end element that enables user interaction and transaction beginnings is the user interface. On the blockchain, smart contracts are self-executing programs that uphold the rules of transactions. This system design gives specifics regarding the system's dependability and credibility and supports high-volume transactions using the "BlockPay" system structure.

**Functional Requirements:** Functional Requirements for the Blockchain-based Defi System (BlockPay) can include:

* **User Registration and Authentication**: To use the platform, users must be able to sign up and authenticate themselves using the system.
* **Wallet Management:** To enable users to transmit, receive, and manage their digital assets like MetaMask, the system should offer a safe and user-friendly wallet management system.
* **Transaction Verification**: The system should check each transaction to make sure the owner of the digital asset has allowed it and that it is valid.
* **Transaction History**: For each user, the system must keep track of all transactions that have taken place on the platform.
* **Notification System**: The system has to have a notification system to inform users of their account activity and transactions.
* **Multi-Currency Support**: To give users rights in their transactions, the system should allow several currencies, including both cash and cryptocurrency.
* **Smart Contract Support**: The system needs to allow for the usage of smart contracts, which are self-executing contracts in which the conditions of the contract between the buyer and seller are directly encoded in lines of code (Solidity).
* **Refund and Dispute Resolution**: The system needs to provide a refund policy for illegal or fraudulent transactions and a dispute resolution process for user complaints. For example, a smart contract may automatically initiate a refund to a buyer if they pay for goods but do not receive it within a predetermined deadline.
* **Transaction Fees**: To ensure the long-term viability of the platform, the system should have an open and acceptable cost structure for all transactions.

**Non-Functional Requirements:**

* **Security:** To prevent fraud, hacking, and other unwanted access, the system must be extremely secure. To guarantee that private user information is protected, it should also feature strong encryption and access controls.
* **Performance:** The system needs to be able to process a large number of transactions without crashing or going down. Additionally, it must to be reachable to accommodate future growth in usage.
* **Availability:** The system must have little downtime for maintenance or updates and be accessible to users around the clock.
* **Reliability:** The system must offer accurate transaction records and be dependable. In case of unexpected failures, it must also have a backup strategy.
* **Compliance:** The system must adhere to all applicable rules and laws, particularly those about anti-money laundering.
* **Usability:** The system needs to have a user interface that is simple to navigate and understand. Additionally, it needs to give users clear instructions on how to finish transactions and solve problems.
* **Interoperability:** For cross-border transactions and interoperability, the system must be able to interface with other financial systems and platforms.
* **Sustainability:** The system must be created with a focus on environmental sustainability, reducing energy consumption, and maintaining waste.

.

**User Interface Requirements:**

* **User-friendly interface:** Users of all technical backgrounds should be able to use, navigate, and understand the "BlockPay" system's user interface.
* **Security features**: To ensure that users' money and personal information are protected, the user interface should include strong security protections.
* **Mobile responsiveness**: For users to access the system from their mobile devices, the system must have a mobile-responsive design.
* **Transaction history:** The user interface must give users access to a transaction history that lists all previous platform transactions.
* **Easy account management:** Users should have access to simple account management options, such as password reset and account recovery, through the user interface.
* **Multi-language support**: Support for many languages is necessary so that the user interface can accommodate users from various geographical areas.
* **Clear feedback and notifications**: Clear feedback and notifications: The user interface needs to inform users straightforwardly about the status of their transactions and any account modifications

## References

* "Blockchain Basics: A Non-Technical Introduction in 25 Steps" by Daniel Drescher, 2nd Edition, Wiley, September 2021.
* “Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained" by Imran Bashir, 2nd Edition, Packt Publishing, August 2020.
* "Ethereum: A Secure Decentralized Generalized Transaction Ledger" by Vitalik Buterin, November 2013.

# Overall Description

## Product Perspective

**Business requirements**

The "BlockPay" system is a decentralized, blockchain-based method of exchanging money that wants to offer users safe, quick, and affordable transactions. Users will be able to send and receive payments using a variety of cryptocurrencies, such as Bitcoin, Ethereum, and others, because of the system's design. Users will be able to safely store their digital assets using the system's built-in wallet feature. The system will enable a variety of user roles, each with its own set of capabilities and permissions, including buyers, sellers, and administrators.

**Context and Origin**

A brand-new system called BlockPay is being created to meet the increasing demand for reliable and actual blockchain-based DeFi systems. The Defi system is completely new with unique properties.

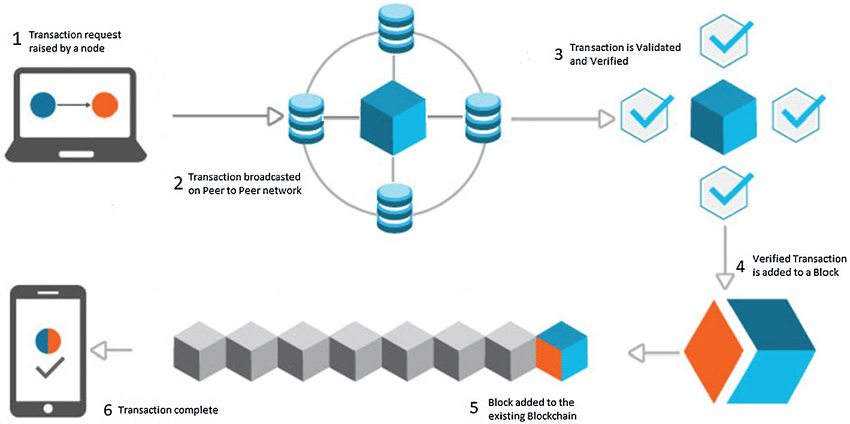


Fig-1: Transaction flow in a Blockchain

## Product Functions

* Create a new user account
* Log in to an existing user account
* View account balance and transaction history
* Send cryptocurrency to other users
* Receive cryptocurrency from other users
* Withdraw cryptocurrency to an external wallet
* Refund a transaction
* Raise a dispute for a transaction
* Resolve a dispute for a transaction
* View real-time exchange rates
* Generate a transaction receipt

**The flow of Data diagram in this system**

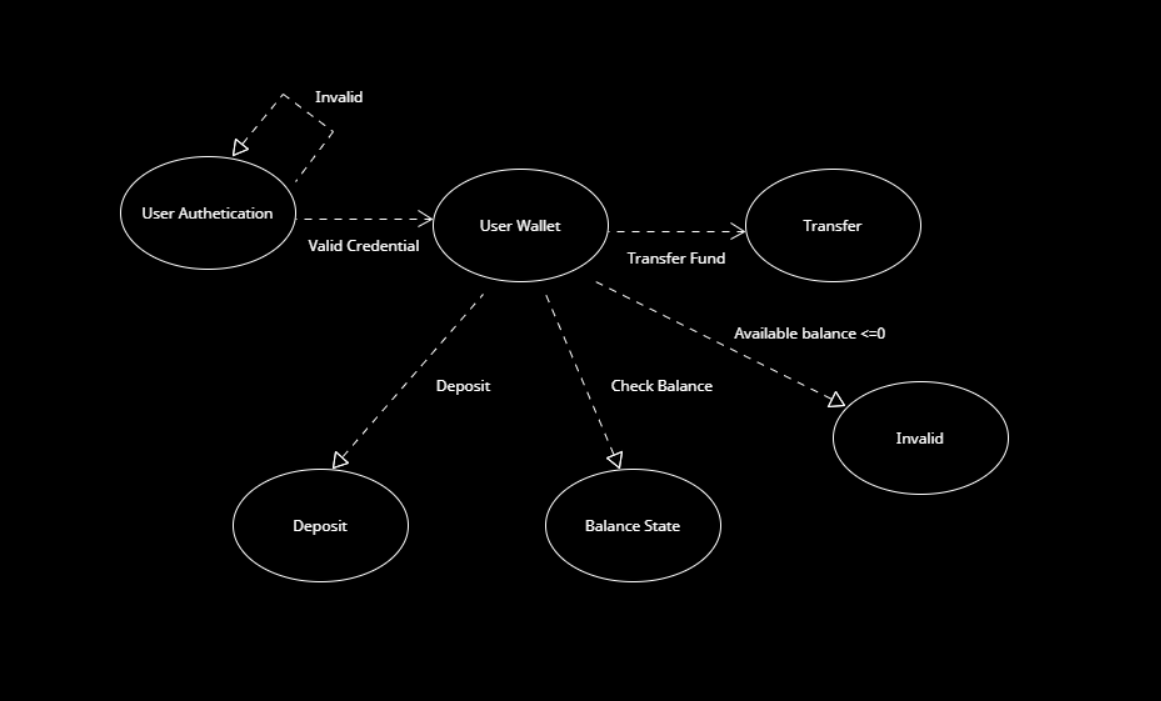
****

Fig-2: Data Flow Diagram

## User Classes and Characteristics

* **Customers**: Customers are people or organizations who use the system to send or receive money. They may use the system frequently or infrequently, and they may possess varied degrees of technical competence.
* **Merchants**: Businesses that accept payments through the system are known as merchants. They may use the system frequently or infrequently, and they may possess varying degrees of technical competence.
* **Administrators**: Administrators are those who are in charge of managing and operating the system. They are in charge of guaranteeing the system's functionality and security and possess a high level of technological skill.
* **Developers**: Developers are the people in charge of creating and maintaining the system's software. They are in charge of assuring the operation and reliability of the system and possess a high level of technical skill.

## Operating Environment

The operating environment for the blockchain-based decentralized money transaction system named BlockPay is as follows:

**Hardware Platform:**

“BlockPay” will operate on any hardware platform that supports the minimum hardware requirements specified below.

* Processor: Intel Core i3 (Core i5 recommended)
* RAM: 8GB or higher
* Hard Disk Space: 100GB or higher
* Network Interface Card: Ethernet (10/100/1000)
* Display: 1280x800 or higher resolution

**Operating System:**

* Windows 10 and above
* macOS High Sierra and above
* Ubuntu 18.04 and above
* Android 8.0 and above
* iOS 12.0 and above

**Software Components:**

* The latest version of a compatible web browser (e.g. Chrome, Firefox, Safari)
* The latest version of Ethereum wallet software (e.g. MetaMask)

## Design and Implementation Constraints

**Regulatory agreement:** The system is required to abide by all rules and regulations that are relevant to financial transactions, the security of personal information, and data privacy.

**Security considerations:** Strong security measures must be in place on the system to guard against unauthorized access, data breaches, and other security concerns.

**Technology requirements:** Specific technologies, tools, and databases, such as blockchain protocols, cryptocurrency wallets, and encryption algorithms, must all be compatible with the system for it to function.

**Interoperability with other systems:** The "BlockPay" system must be able to communicate with other software programs, hardware, or online shopping carts that are used in DeFi transactions.

**Performance requirements:** The system must be capable of supporting a specific number of users and transactions with the least amount of downtime or latency.

**User interface standards:** For user interface components like buttons, menus, and forms, the system must adhere to specific design standards or coding guidelines.

**Maintenance and support:** Whether performed by the development team or the customer's company, the system must be built and documented to make maintenance and support simple.

## User Documentation

**Installation guide:** The installation guide would provide instructions on how to install the software on the user's device, which may include specific requirements for hardware and operating system compatibility.

**User manual:** The user manual would describe how to use the software to perform decentralized money transactions, how to manage funds and wallets, and any other features unique to “BlockPay”.

**Quick start guide:** A quick start guide may be provided for users who want to get started using the software quickly without reading the full user manual.

**Help files:** Help files may be included in the BlockPay software itself, providing on-screen assistance with specific tasks or questions that users may have.

**Release notes:** Release notes may be included with each new version of the software, detailing any changes or updates made to the system.

**FAQs:** FAQs may also be provided to help users troubleshoot common issues or concerns.

**The known user documentation delivery formats and standards may contain:**

Depending on the user's demands, "BlockPay" user documentation may be given in PDF or HTML format. Additionally, video tutorials guiding users through using the software may be offered. These are especially helpful for new users who are unfamiliar with blockchain-based financial transactions. To satisfy the demands of users in various countries or areas, localization may also be thought of as the offering of documentation in several languages.

# System Requirements

## System Features

**1. Software Login**

**Functional Requirements (FRs)**

* 1. The software shall allow users to login with their given username and password.
  2. The login credentials (username and password) will be verified with database records.
  3. If the login successful, the home page of the user account will be displayed.
  4. If the username and/or password has been inserted wrong, the random verification code will be generated and sent to the user’s email address by the system to retry login.
  5. If the number of login attempt exceed its limit (3 times), the system shall block the user account login for one hour *[optional function]*

**Priority Level:** High **Precondition:** user have valid user id and password  
**Cross-references:** 4.1, 7.2, 9 (example)

1. **User Registration and Authentication**

**Functional Requirements (FRs)**

* 1. The software shall allow users to register with their personal information, including email addresses, phone numbers, and NID verification.
  2. The system shall verify the user's information against the database records to ensure uniqueness.
  3. The software shall provide a confirmation email or SMS message to the user to activate their account.
  4. The system shall allow users to reset their password with email verification.

**Priority Level**: High

**Precondition**: user provides valid personal information and email/phone number.

**Cross-references:** None

1. **Wallet Management**

**Functional Requirements (FRs)**

* 1. The software shall allow users to create and manage their digital wallets.
  2. The system shall enable users to add or remove digital assets from their wallets.
  3. The software shall provide users with the present exchange rates of digital assets in real time.

**Priority Level**: High

**Precondition**: user has a registered and verified account.

**Cross-references:** 2.1, 2.2

1. **Transaction Verification**

**Functional Requirements (FRs)**

* 1. The software shall allow users to verify their transactions on the blockchain.
  2. The system shall provide users with the transaction hash (encrypted) and status for each transaction.
  3. The software shall notify users of successful transactions or errors.

**Priority Level**: High

**Precondition**: user has initiated a transaction.

**Cross-references:** 2.1, 3.1

1. **Transaction History**

**Functional Requirements (FRs)**

* 1. The software shall allow users to view their transaction history.
  2. The system shall display transaction details, such as sender, recipient, amount, and status.
  3. The software shall allow users to filter and search their transaction history.

**Priority Level**: Medium

**Precondition**: user has made at least one transaction.

**Cross-references:** None

1. **Notification System**

**Functional Requirements (FRs)**

* 1. The software shall provide a notification system to alert users of important events, such as successful transactions or changes in account status.
  2. The system shall allow users to customize their notification preferences.

**Priority Level**: Medium

**Precondition:** user has a registered and verified account.

**Cross-references:** 2.1, 2.2

1. **Multi-Currency Support**

**Functional Requirements (FRs)**

* 1. The software shall support multiple digital assets and cryptocurrencies for transactions.
  2. The system shall allow users to exchange one digital asset for another.

**Priority Level**: Medium

**Precondition**: user has a registered and verified account.

**Cross-references:** 2.1, 3.

1. **Smart Contract Support**

**Functional Requirements (FRs)**

* 1. The software shall support the execution of smart contracts on the blockchain.
  2. The system shall allow users to create, deploy, and interact with smart contracts.

**Priority Level**: High

**Precondition**: user has a registered and verified account.

**Cross-references:** None

1. **Refund and Dispute Resolution**

**Functional Requirements (FRs)**

* 1. The software shall provide a mechanism for users to initiate refund requests.
  2. The system shall allow users to resolve disputes through a decentralized arbitration process.

**Priority Level**: High

**Precondition**: user has a registered and verified account.

**Cross-references:** 6.2, 7.1.

1. **Transaction Fees**

**Functional Requirements (FRs)**

* 1. The software shall charge transaction fees for each transaction, which will be paid in the digital asset used for the transaction.
  2. The system shall provide users with information on the transaction fees before the transaction is confirmed.

**Priority Level**: Medium

**Precondition**: user has initiated a transaction.

**Cross-references:** 3.1, 3.2, 4.1.

## Non-Functional/Quality Requirements

**QA1: Security:** The system shall utilize public-key cryptography to ensure the confidentiality and integrity of all transactions.

**Priority Level**: High

**Precondition**: N/A

**Cross-references**: N/A

**QA2: Performance**: The system shall be able to handle at least 1,000 transactions per second.

**Priority Level**: High

**Precondition**: N/A

**Cross-references**: N/A

**QA3: Availability**: The system shall have an uptime of at least 99.99%.

**Priority Level**: High

**Precondition**: N/A

**Cross-references**: N/A

**QA4: Reliability**: The system shall be able to handle network partitions and continue to operate correctly.

**Priority Level**: High

**Precondition**: N/A

**Cross-references**: N/A

**QA5: Compliance**: The system shall comply with all relevant regulations and laws regarding financial transactions and data privacy.

**Priority Level**: High

**Precondition**: N/A

**Cross-references**: N/A

**QA6: Usability**: The user interface shall be intuitive and easy to use, with clear instructions and feedback provided at all times.

**Priority Level**: Medium

**Precondition**: N/A

**Cross-references**: N/A

**QA7: Interoperability**: The system shall be able to interface with other blockchain-based systems and traditional financial systems.

**Priority Level**: Medium

**Precondition**: N/A

**Cross-references**: N/A

**QA8: Sustainability**: The system shall use energy-efficient consensus mechanisms to minimize its environmental impact.

**Priority Level:** Low

**Precondition:** N/A

**Cross-references**: N/A

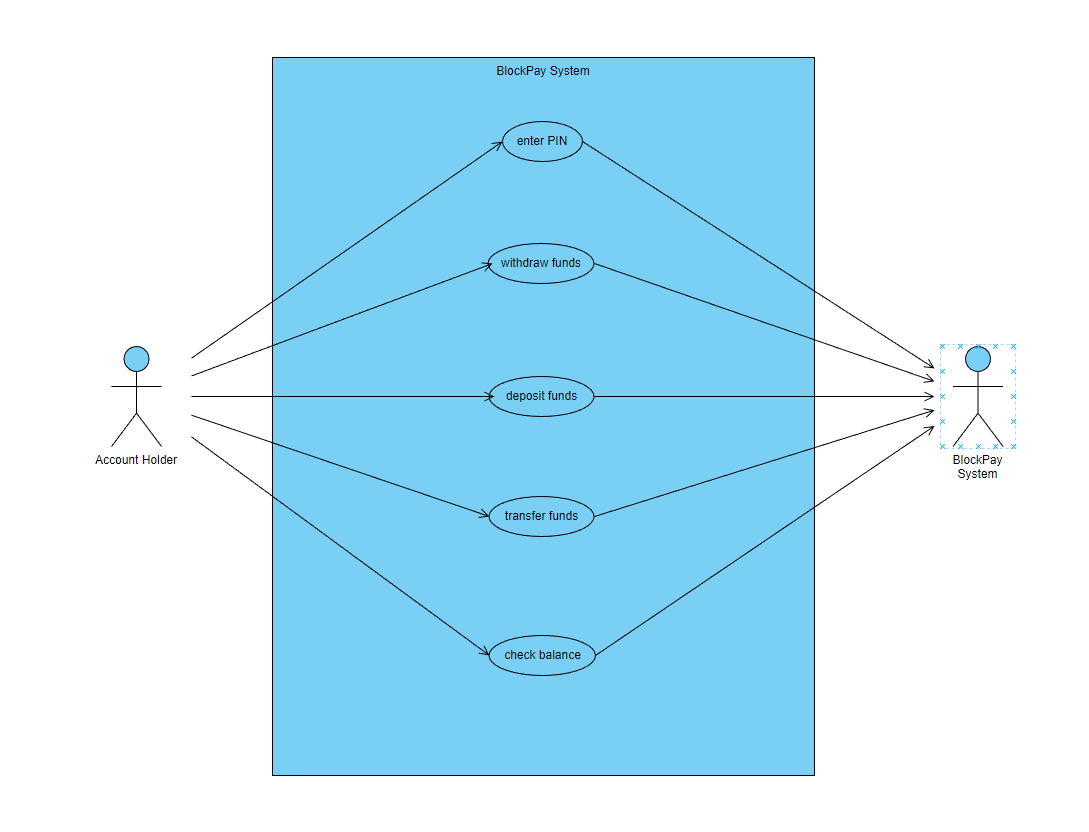
## Project Requirements

* **Development Tools**: The development team needs access to particular development tools, such as the Alchemy framework, the Solidity compiler, etc.
* **Security Tools**: To provide secure transactions and stop malicious behaviors, the system has to interface with security tools like MetaMask.
* **Payment Gateways**: To enable simple and safe transactions for users, the system must allow a variety of payment gateways. It can be necessary to integrate with well-known payment processors like PayPal, Stripe, or BitPay.
* **Analytics Tools**: To track user activity, transaction patterns, and other pertinent metrics, the system must integrate with analytics tools. “BockPay” can use programs like Google Analytics for this.
* **Scalability Tools:** The system needs to be able to handle a lot of transactions. It might be necessary to integrate scaling technologies.
* **Blockchain Network**: The system must path on a blockchain network that supports smart contract implementation, such as Ethereum, Binance, etc.
* **Node Hosting**: The system must have a dependable node hosting facility to ensure the uptime of the blockchain network. Services like Alchemy can be used for this resolution.
* **Wallet Integration**: The system must participate with popular wallets like MetaMask or Trust Wallet to allow users to store and accomplish their digital assets.
* **APIs**: The system must provide APIs for developers to build applications. APIs for transactions, asset management, and user verification may be required.

# Design and Interface Requirements

## UML Diagrams

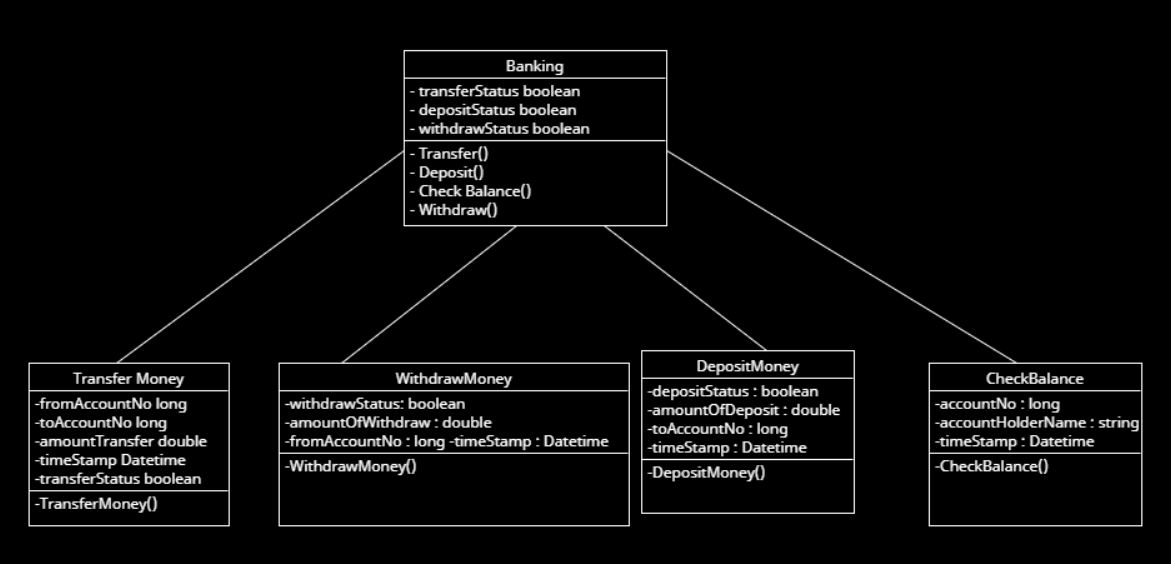
**Use Case:**



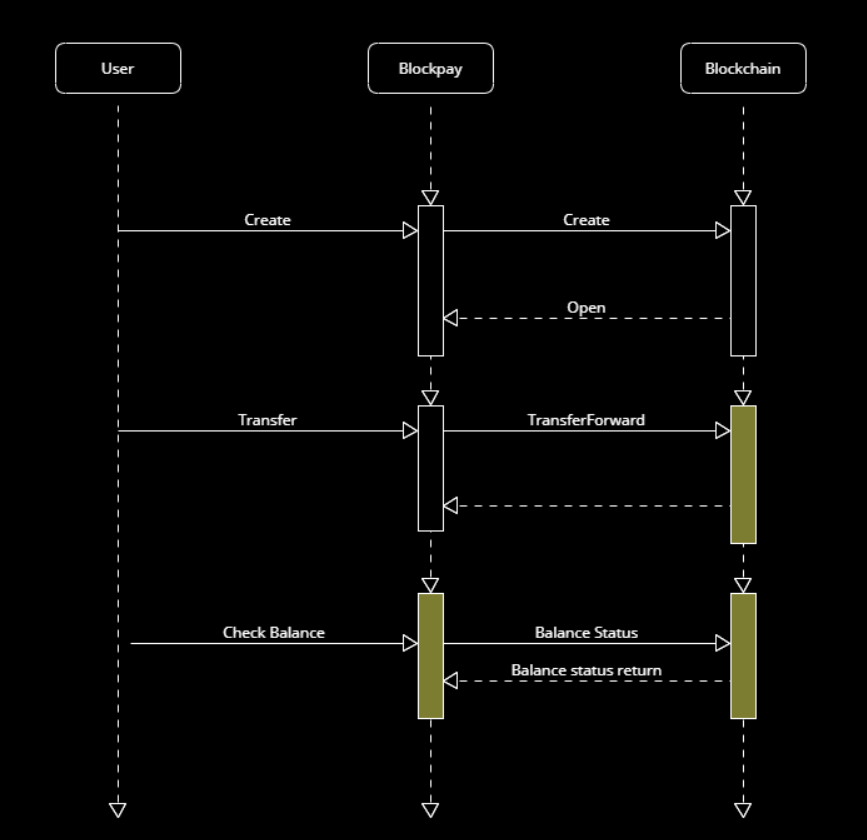
**Class Diagram**

**Scenario:** N is a customer of the BlockPay system and wants to transfer $X from his checking account to his savings account.

N logs in to the BlockPay system and selects the Transfer option. N enters his checking account number and savings account number. N enters the amount of $X to be transferred. N confirms the transaction and the system records the current date and time as the timestamp for the transaction. The system checks the balance of N's checking account to ensure that he has sufficient funds. If N has sufficient funds, the system updates the transfer status to "successful" and deducts $X from his checking account. The system then updates the deposit status to "successful" and adds $X to his savings account. John can then check his account balances to verify that the transfer was completed successfully.



**Sequence Diagram:**

****

## Data Dictionary

**Transaction Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Validation | Key |
| Transaction | TxID | String | Unique, alphanumeric | Primary Key |
| Transaction | SenderAddress | String | Alphanumeric | Foreign Key(Address.AddressID) |
| Transaction | RecieverAddress | String | Alphanumeric | Foreign Key (Address.AddressID) |
| Transaction | Amount | Decimal | Positive, non zero |  |
| Transaction | Currency | String | Alphabetic |  |
| Transaction | DateTime | DateTime |  |  |
| Transaction | Fee | Decimal | Positive, non zero |  |

**Address Table**

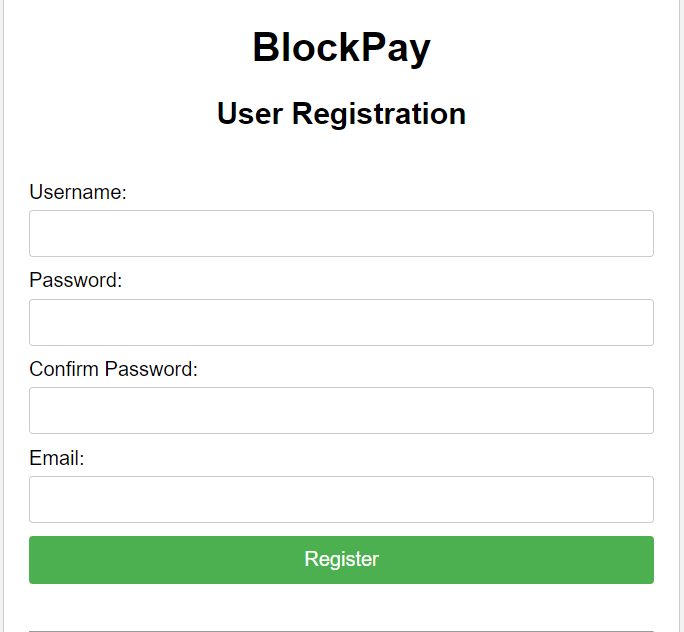
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Validation | Key |
| Address | AddressID | string | Unique, alphanumerical | Primary Key |
| Address | UserID | String | Unique, alphanumerical | Foreign Key (User.UserID) |
| Address | Currency | string | Alphabetic |  |

**User Table**

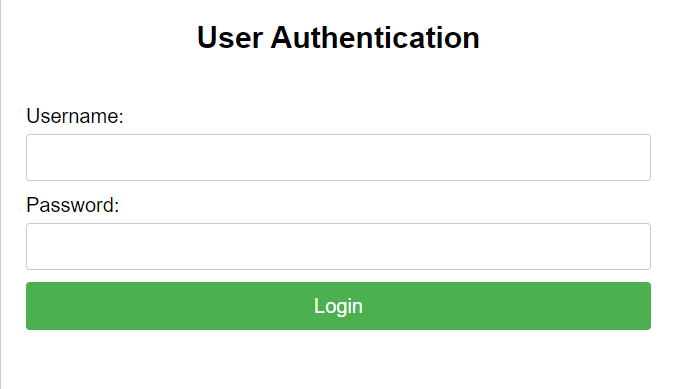
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Validation | Key |
| User | UserID | String | Unique, alphanumeric | Primary Key |
| User | Name | String | Alphabetic |  |
| User | Email | string | Valid email format |  |

## UI/UX Design Specification

1. **Registration**

****

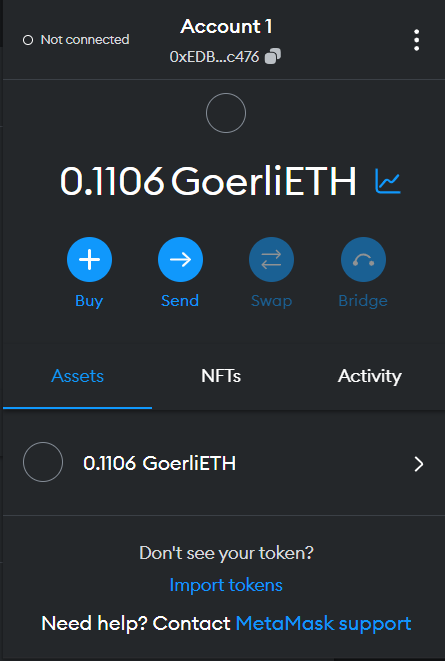
1. **User Authentication**

****

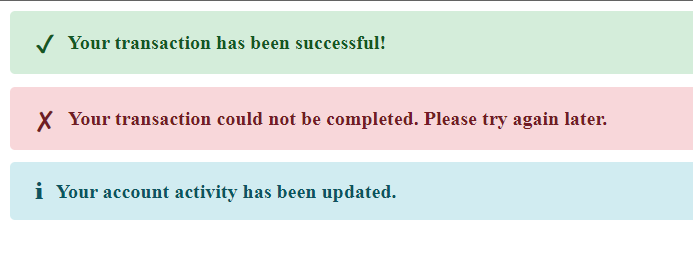
1. **Wallet Management**

****

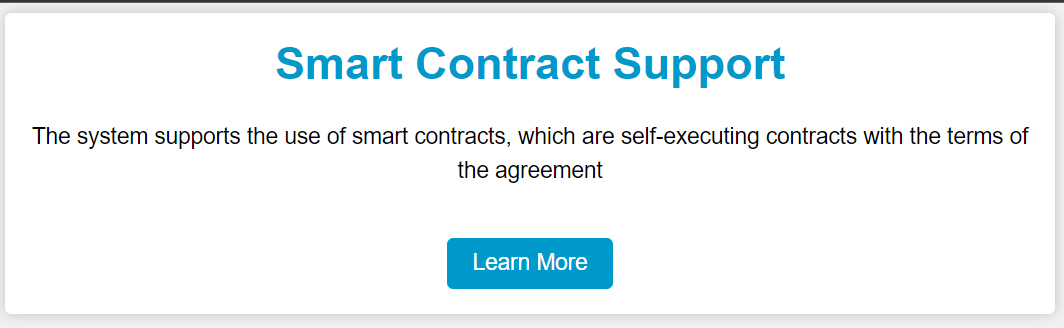
1. **Metamask**

****

1. **Notifaction messages**

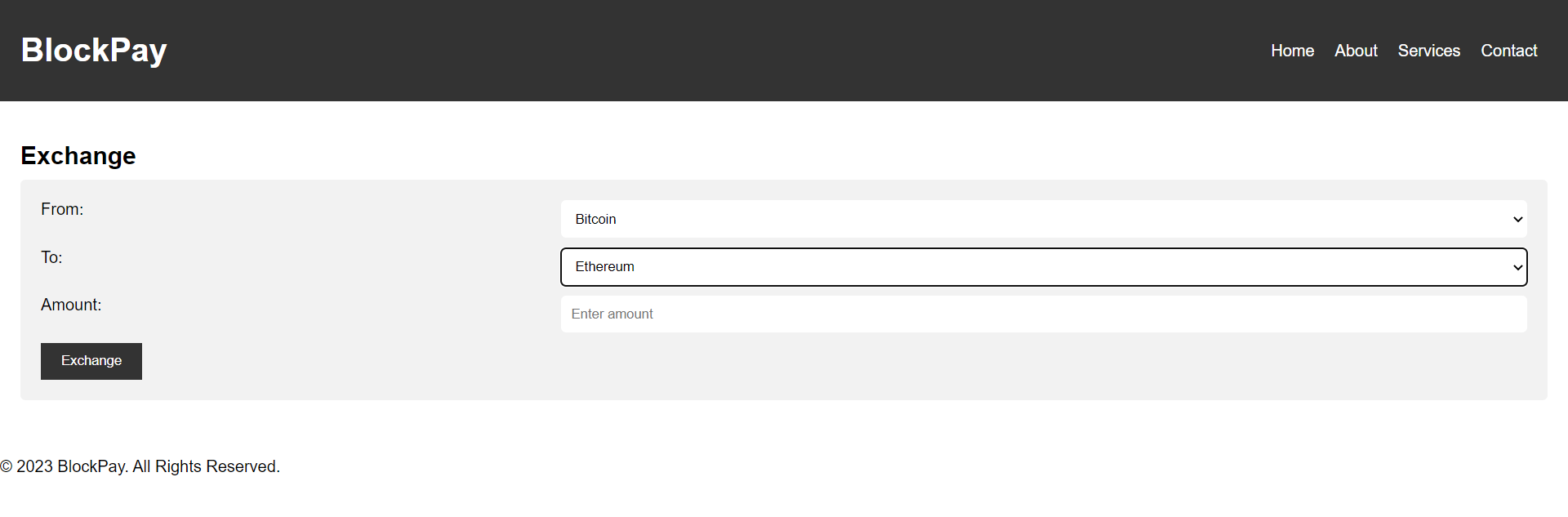
****

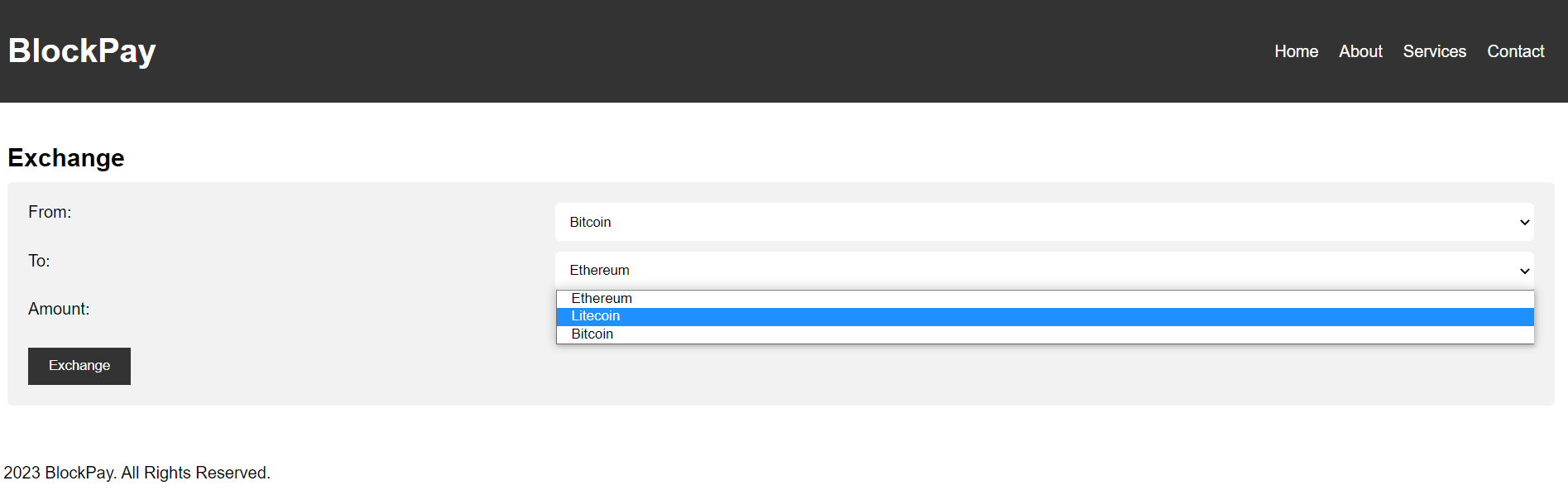
1. **Smart Contact Support policy**

****

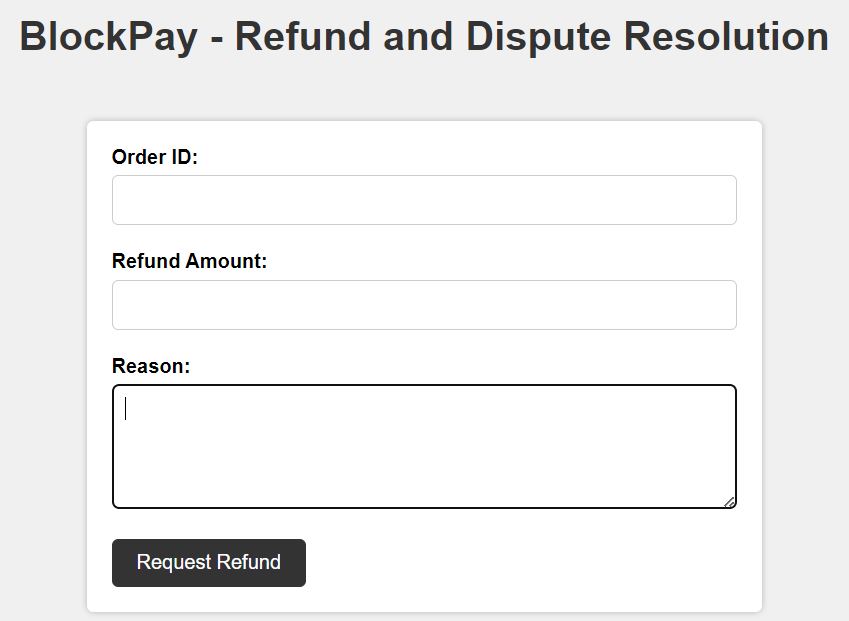
****

1. **Multi-Currency Support**

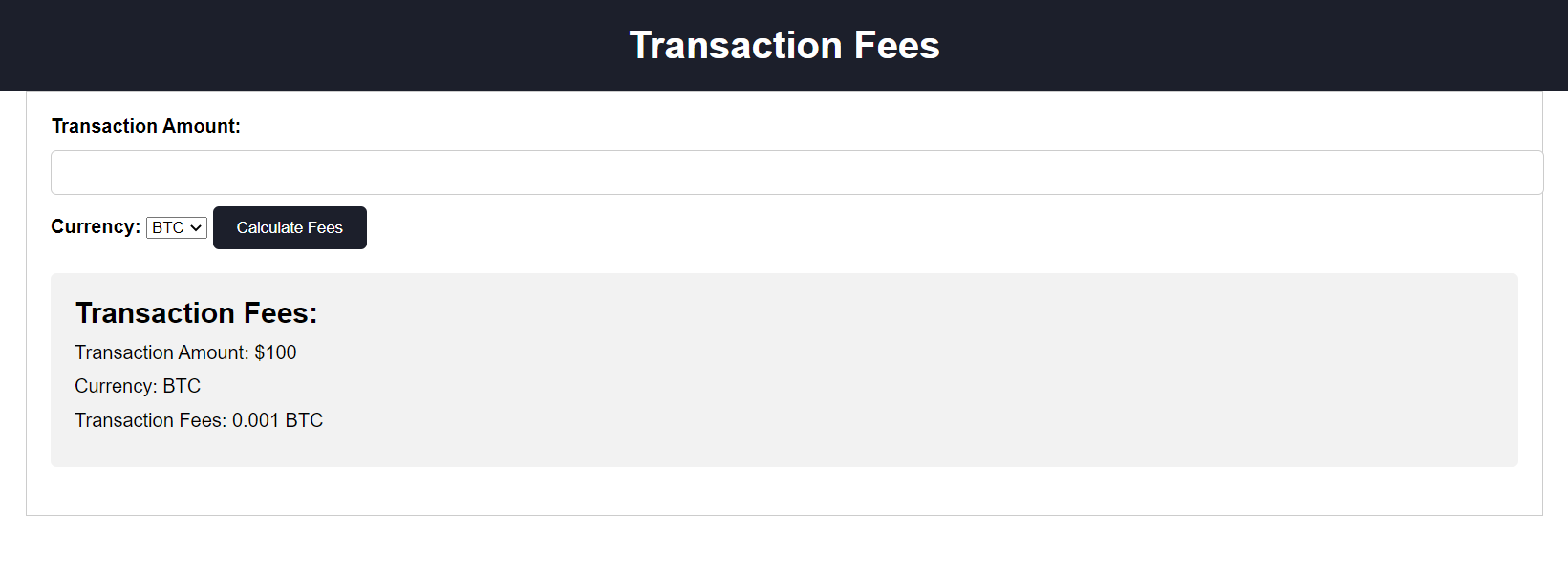
****

****

1. **Refund System**

****

1. **Transaction Fee**

****