Mini Wallet API Design Specification

1. Overview

The Mini Wallet API provides functionalities for managing virtual money transactions within a user's wallet. This includes depositing money into the wallet, withdrawing money from the wallet, and disabling the wallet.

2. Technical Stack

- **Programming Language:** Python 3.8+
- **Framework**: Flask
- **Database:** PostgreSQL for production, SQLite for development
- **Authentication:** API Key
- **Documentation:** Swagger

3. API Endpoints

Deposit Money

- **Endpoint:** POST /api/v1/wallet/deposits
- **Authorization**: API Key
- **Body:**
    - amount: The amount to deposit.
    - reference_id: A unique identifier for the deposit transaction.

Withdraw Money

- **Endpoint:** POST /api/v1/wallet/withdrawals
- **Authorization:** API Key
- **Body:**
    - amount: The amount to withdraw.
    - reference_id: A unique identifier for the withdrawal transaction.

Disable Wallet

- **Endpoint:** PATCH /api/v1/wallet
- **Authorization:** API Key
- **Body:**
    - is_disabled: Boolean value to disable the wallet.

4. Database Models

User

- **id**: Primary Key
- **username**: String
- **password**: String
- **token**: String for API Key authentication

Wallet

- **id**: Primary Key
- **user_id**: Foreign Key to User
- **balance**: Decimal
- **is_disabled**: Boolean

Transaction

- **id**: Primary Key
- **wallet_id**: Foreign Key to Wallet
- **amount**: Decimal
- **reference_id**: String, Unique
- **transaction_type**: Enum ('DEPOSIT', 'WITHDRAWAL')
- **created_at**: DateTime

5. Security Considerations

- Secure API Key authentication.
- Validate all input data to prevent vulnerabilities.
- Employ HTTPS for API communication.

6. Testing

- **Unit Tests:** For individual components.
- **Integration Tests**: For testing endpoint interactions.

7. Deployment

- Use Docker for containerization.
- Configure application using environment variables.