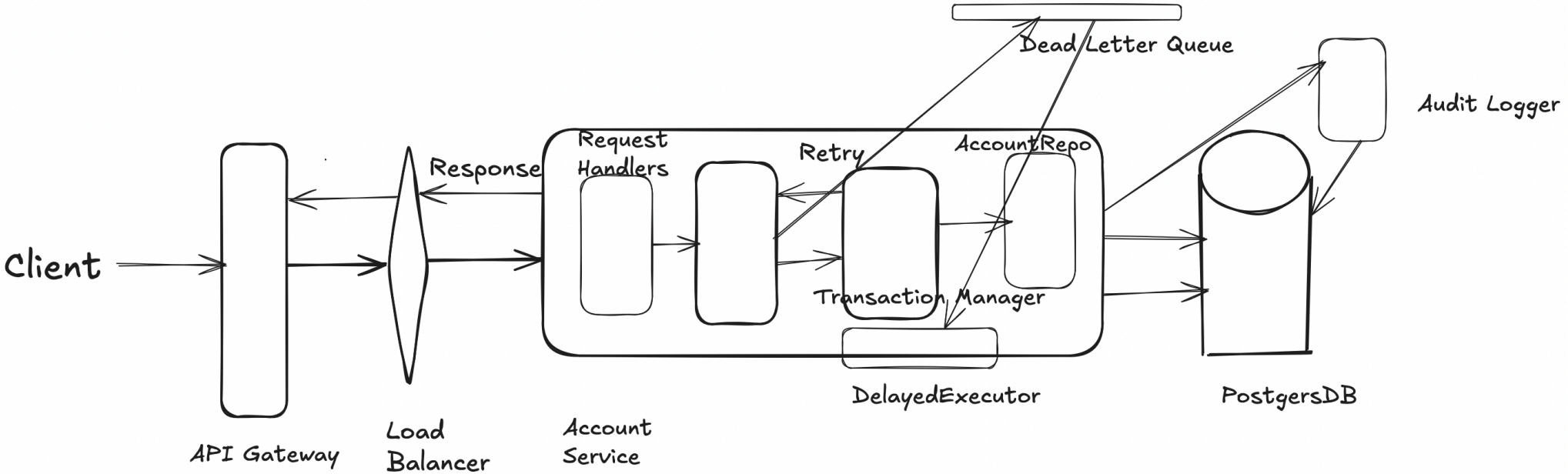
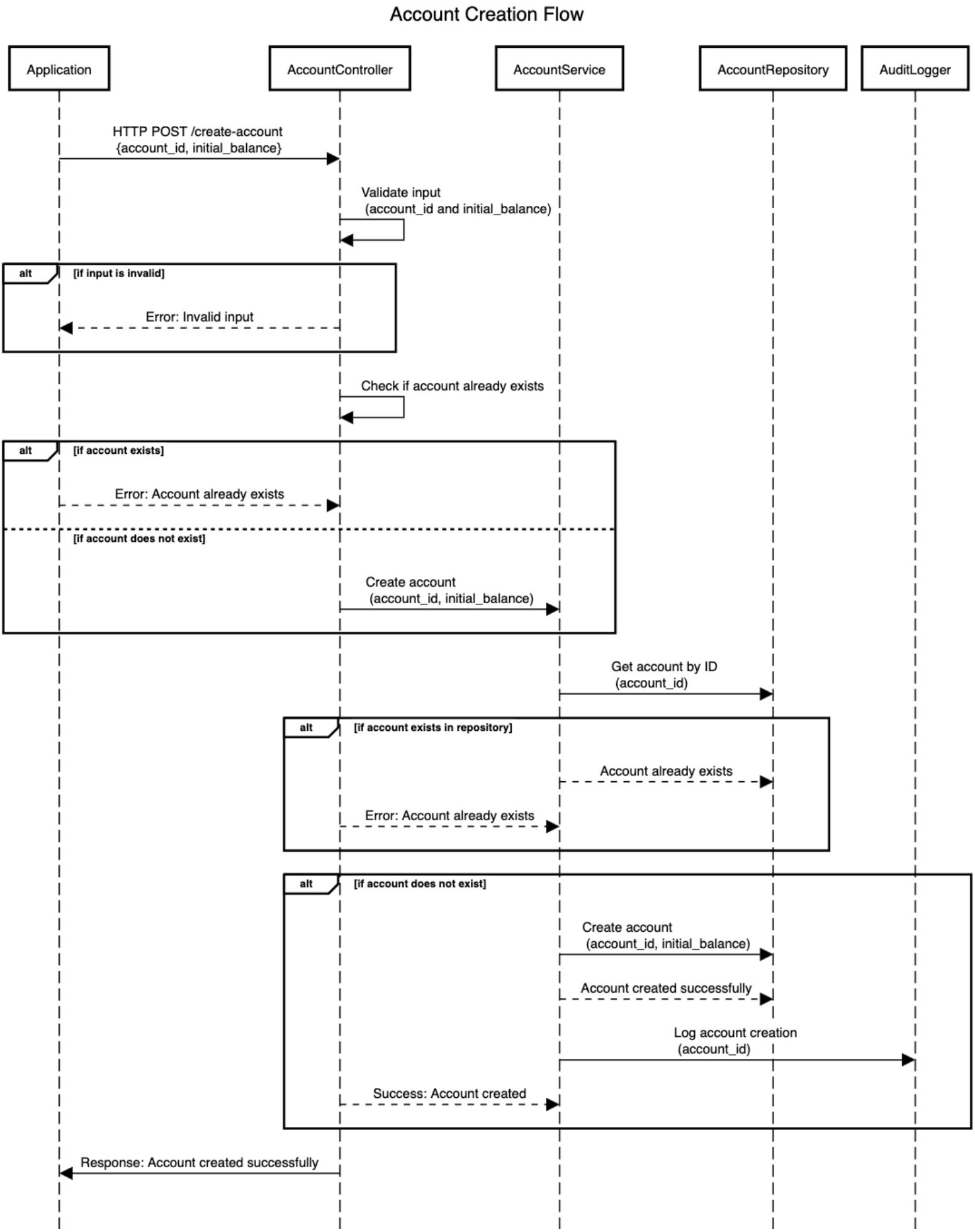
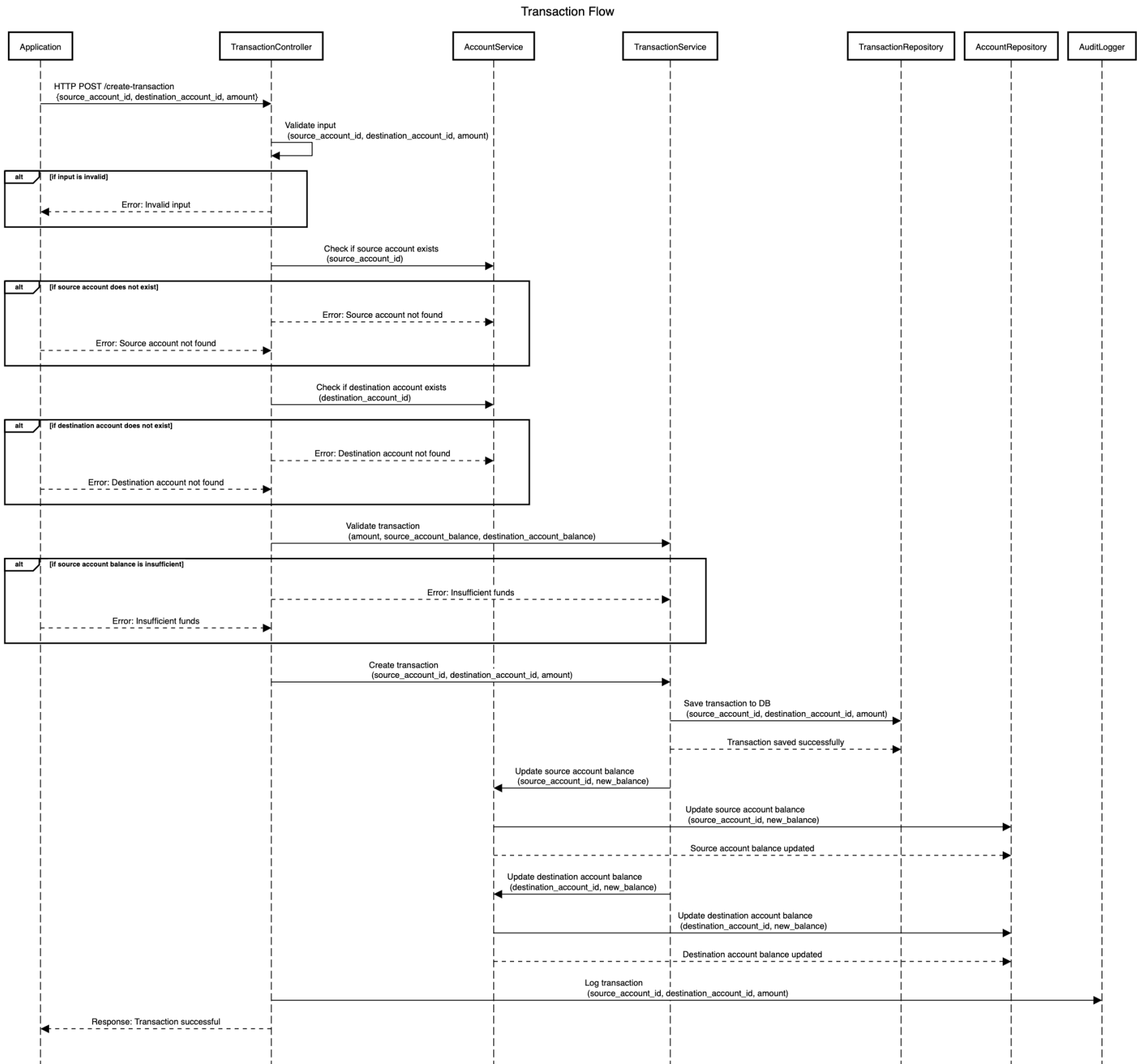


Transfer-System


Account Creation Flow
Application
AccountController
AccountService
AccountRepository
AuditLogger
HTTP POST /create-account
{account_id, initial_balance}
Validate input
(account_id and initial_balance)
alt
[if input is invalid]
Error: Invalid input
Check if account already exists
alt
[if account exists]
Error: Account already exists
[if account does not exist]
Create account
(account_id, initial_balance)
Get account by ID
(account_id)
alt
[if account exists in repository]
Account already exists
Error: Account already exists
alt
[if account does not exist]
Create account
(account_id, initial_balance)
Account created successfully
Log account creation
(account_id)
Success: Account created
Response: Account created successfully


Transaction Flow
Application
TransactionController
AccountService
TransactionService
TransactionRepository
AccountRepository
AuditLogger
HTTP POST /create-transaction
{source_account_id, destination_account_id, amount}
Validate input
(source_account_id, destination_account_id, amount)
alt
[if input is invalid]
Error: Invalid input
Check if source account exists
(source_account_id)
alt
[if source account does not exist]
Error: Source account not found
Error: Source account not found
Check if destination account exists
(destination_account_id)
alt
[if destination account does not exist]
Error: Destination account not found
Error: Destination account not found
Validate transaction
(amount, source_account_balance, destination_account_balance)
alt
[if source account balance is insufficient]
Error: Insufficient funds
Error: Insufficient funds
Create transaction
(source_account_id, destination_account_id, amount)
Save transaction to DB
(source_account_id, destination_account_id, amount)
Transaction saved successfully
Update source account balance
(source_account_id, new_balance)
Update source account balance
(source_account_id, new_balance)
Source account balance updated
Update destination account balance
(destination_account_id, new_balance)
Update destination account balance
(destination_account_id, new_balance)
Destination account balance updated
Log transaction
(source_account_id, destination_account_id, amount)
Response: Transaction successful

About the Project

```
/internal-transfers
├── /cmd
│   └── main.go                          # Entry point for the application
├── /common
│   ├── audit_logger.go                  # Implements the AuditLogger for logging application actions
│   ├── config.go                        # Configuration utilities, including loading environment variables
│   └── logger.go                        # Logger utilities for consistent logging across the application
├── /controller
│   ├── account_controller.go            # Handles HTTP requests for account-related operations (REST API controllers)
│   └── transaction_controller.go        # Handles HTTP requests for transaction-related operations
├── /model
│   ├── account.go                       # Defines the Account data structure
│   ├── transaction.go                   # Defines the Transaction data structure
│   └── input_validation.go              # Input validation logic and request data models (e.g., CreateAccountInput)
├── /persistence
│   ├── account_repository.go            # Implements data access methods for account operations (CRUD)
│   ├── transaction_repository.go        # Implements data access methods for transaction operations (CRUD)
│   └── db.go                            # Database connection and setup
├── /service
│   ├── account_service.go               # Business logic for account operations
│   ├── transaction_service.go           # Business logic for transaction operations
├── /tests
│   ├── unit
│   │   ├── account_service_test.go      # Unit tests for account service logic
│   │   ├── transaction_service_test.go  # Unit tests for transaction service logic
│   │   ├── account_controller_test.go   # Unit tests for account controller
│   │   └── transaction_controller_test.go # Unit tests for transaction controller
│   ├── mocks
│   │   ├── mock_account_repository.go   # Mock for AccountRepository for unit tests
│   │   ├── mock_transaction_repository.go# Mock for TransactionRepository for unit tests
│   │   └── mock_account_service.go      # Mock for AccountService for unit tests
└── go.mod                               # Go modules file for dependencies
```