**EXPERIMENT-4**

D.Jaswanth Reddy

BU22CSEN0100330

1. UML DIAGRAMS OF ONLINE BANKING SERVICES

**USE CASE DIAGRAM**

@startuml

actor Customer

actor "Bank Employee" as Employee

actor Admin

rectangle "Online Banking System" {

(User Authentication) as Auth

(View Account Details) as ViewAccount

(Transfer Funds) as Transfer

(Bill Payments) as BillPay

(Apply for Loan) as Loan

(Deposit & Withdraw Money) as CashTransactions

(View Transaction History) as ViewHistory

(Manage Profile) as ManageProfile

(Manage Users) as ManageUsers

(Generate Reports) as Reports

(Loan Approval) as LoanApproval

Auth <|-- ViewAccount

Auth <|-- Transfer

Auth <|-- BillPay

Auth <|-- Loan

Auth <|-- CashTransactions

Auth <|-- ViewHistory

Auth <|-- ManageProfile

Loan <|-- LoanApproval : "extends"

Transfer <|-- ViewAccount : "includes"

BillPay <|-- ViewAccount : "includes"

Loan <|-- Reports : "includes"

}

Customer -- Auth

Customer -- ViewAccount

Customer -- Transfer

Customer -- BillPay

Customer -- Loan

Customer -- CashTransactions

Customer -- ViewHistory

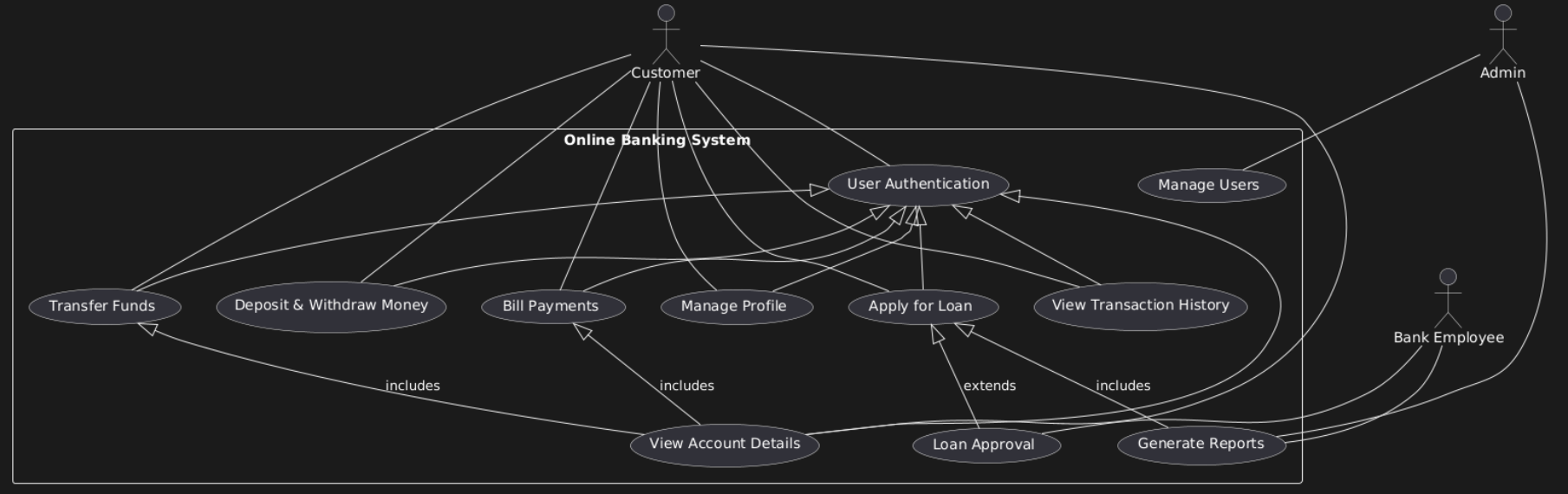
Customer -- ManageProfile

Employee -- LoanApproval

Employee -- Reports

Admin -- ManageUsers

Admin -- Reports

@enduml

**CLASS DIAGRAM**

@startuml

class Customer {

+Login()

+ViewAccountDetails()

+TransferFunds()

+PayBills()

+ApplyForLoan()

+ViewTransactionHistory()

+ManageProfile()

}

class BankEmployee {

+ApproveLoan()

+GenerateReports()

}

class Admin {

+ManageUsers()

+GenerateReports()

}

class Account {

-accountNumber: String

-balance: Double

+getBalance()

+updateBalance()

}

class Transaction {

-transactionId: String

-amount: Double

-date: Date

+processTransaction()

}

class Loan {

-loanId: String

-amount: Double

-interestRate: Double

+applyForLoan()

+approveLoan()

}

Customer --> Account : owns

Customer --> Transaction : initiates

Customer --> Loan : applies for

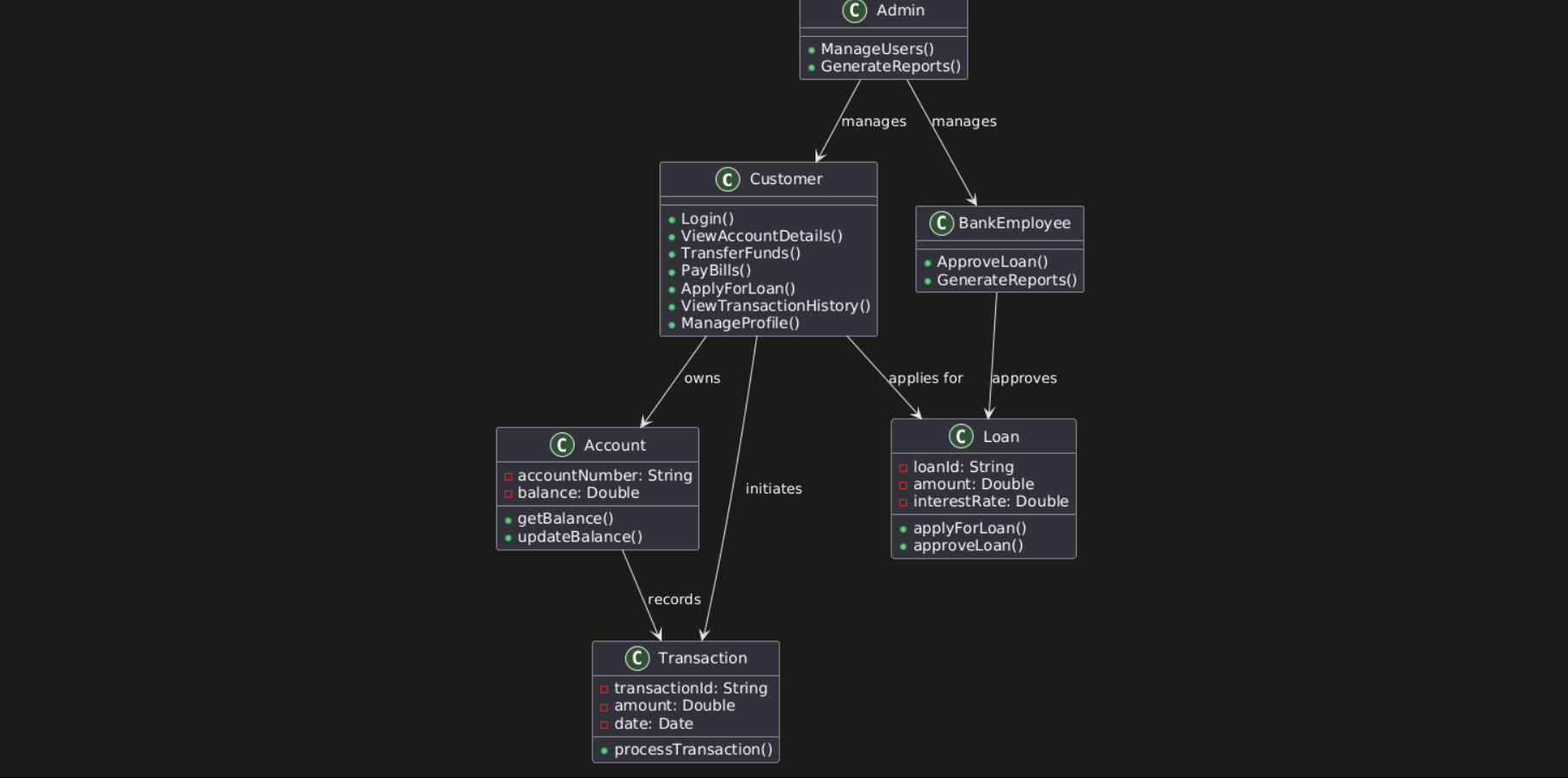
BankEmployee --> Loan : approves

Admin --> Customer : manages

Admin --> BankEmployee : manages

Account --> Transaction : records

@enduml



**SEQUENCE DIAGRAM**

@startuml

actor Customer

participant "Online Banking System" as OBS

participant Account

participant Transaction

participant BankEmployee

Customer -> OBS: Login()

OBS -> Account: GetAccountDetails()

Account --> OBS: ReturnDetails()

OBS --> Customer: DisplayDetails()

Customer -> OBS: TransferFunds(amount, targetAccount)

OBS -> Transaction: CreateTransaction()

Transaction -> Account: UpdateBalances()

Account --> Transaction: ConfirmUpdate()

Transaction --> OBS: TransactionSuccess()

OBS --> Customer: Transfer Successful

Customer -> OBS: ApplyForLoan(amount)

OBS -> BankEmployee: RequestApproval()

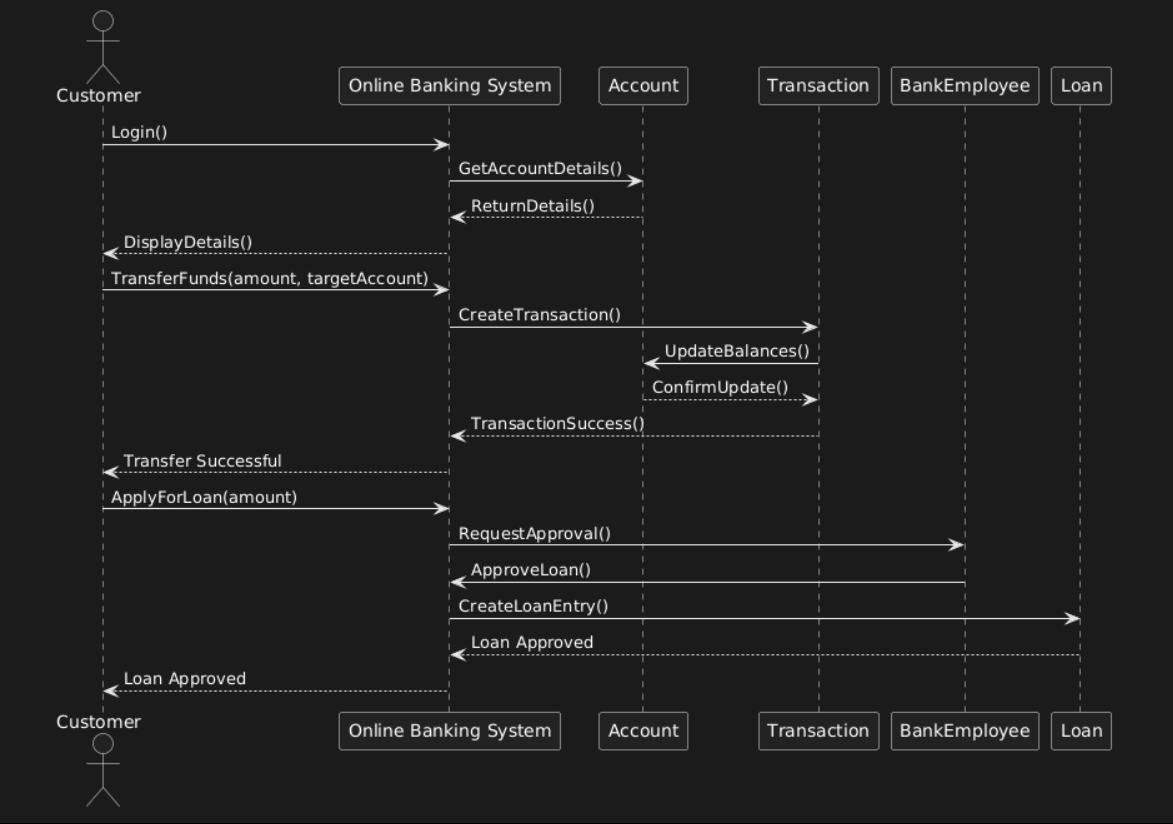
BankEmployee -> OBS: ApproveLoan()

OBS -> Loan: CreateLoanEntry()

Loan --> OBS: Loan Approved

OBS --> Customer: Loan Approved

@enduml



**COLLABORATION DIAGRAM**

@startuml

object Customer

object "Online Banking System" as OBS

object Account

object Transaction

object BankEmployee

object Loan

Customer -> OBS: Login

OBS -> Account: GetAccountDetails

Account -> OBS: ReturnDetails

OBS -> Customer: DisplayDetails

Customer -> OBS: TransferFunds(amount, targetAccount)

OBS -> Transaction: CreateTransaction

Transaction -> Account: UpdateBalances

Account -> Transaction: ConfirmUpdate

Transaction -> OBS: TransactionSuccess

OBS -> Customer: Transfer Successful

Customer -> OBS: ApplyForLoan(amount)

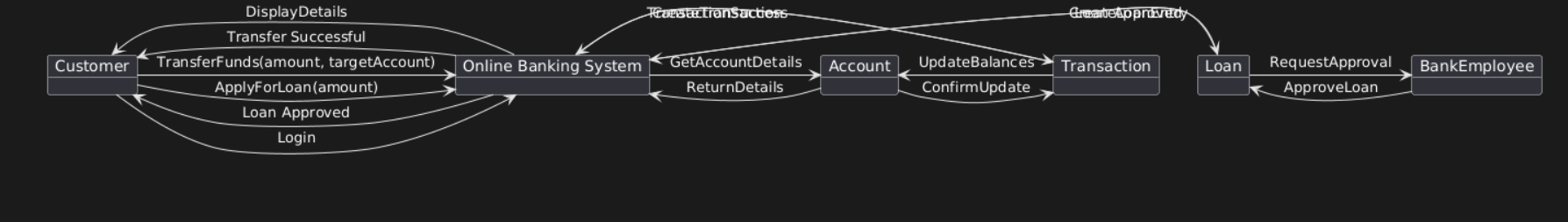
OBS -> Loan: CreateLoanEntry

Loan -> BankEmployee: RequestApproval

BankEmployee -> Loan: ApproveLoan

Loan -> OBS: Loan Approved

OBS -> Customer: Loan Approved

@enduml  
  


**STATE CHART DIAGRAM**

@startuml

state "Online Banking System" as OBS

[\*] --> LoggedOut

LoggedOut --> LoggedIn: User logs in

LoggedIn --> ViewingAccount: View Account Details

ViewingAccount --> LoggedIn: Back to Dashboard

LoggedIn --> TransferringFunds: Initiate Transfer

TransferringFunds --> TransactionProcessing: Process Transaction

TransactionProcessing --> TransactionSuccess: Transfer Successful

TransactionProcessing --> TransactionFailed: Transfer Failed

TransactionSuccess --> LoggedIn

TransactionFailed --> LoggedIn

LoggedIn --> ApplyingForLoan: Apply for Loan

ApplyingForLoan --> LoanApproval: Request Approval

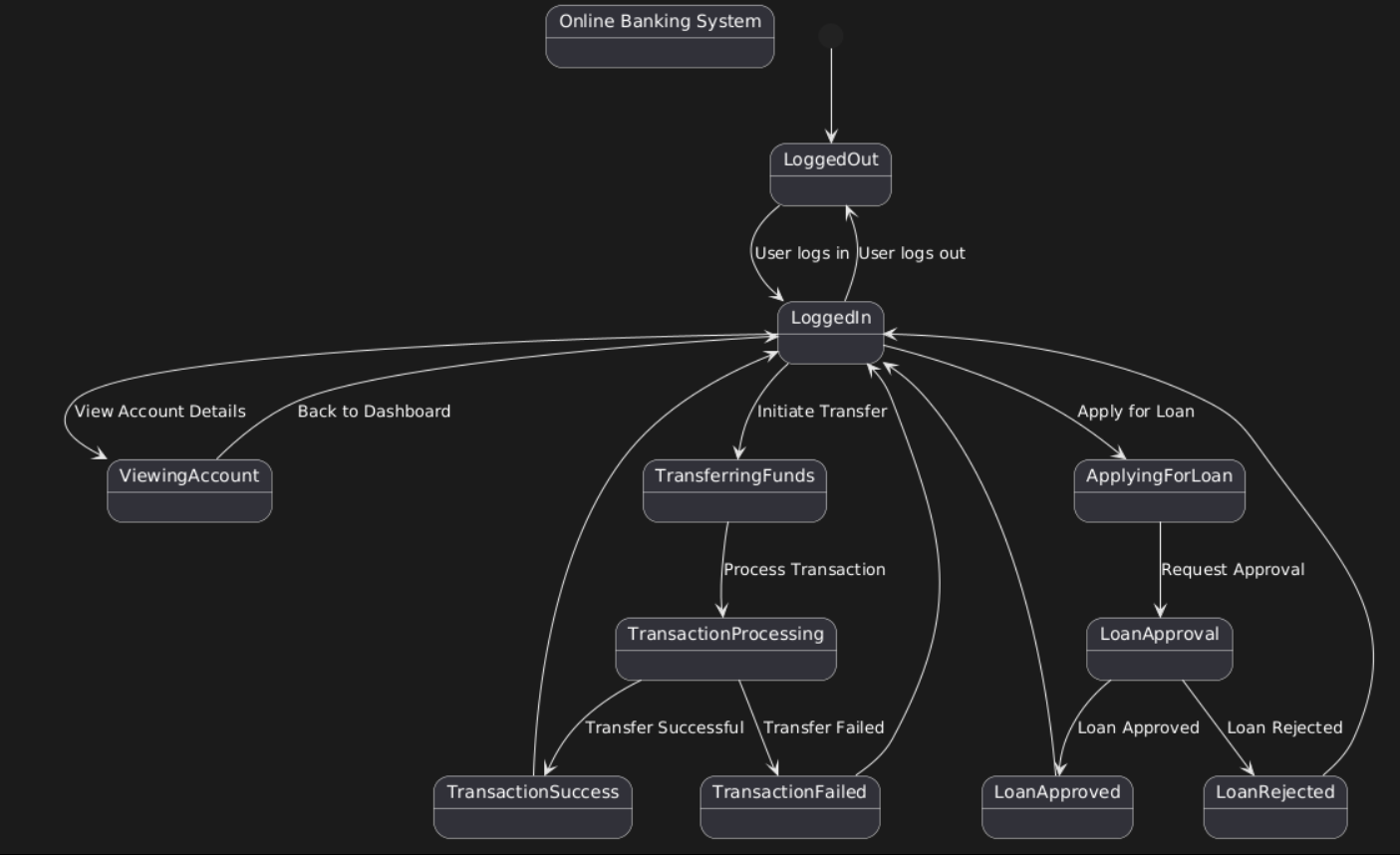
LoanApproval --> LoanApproved: Loan Approved

LoanApproval --> LoanRejected: Loan Rejected

LoanApproved --> LoggedIn

LoanRejected --> LoggedIn

LoggedIn --> LoggedOut: User logs out

@enduml  
  


**ACTIVITY DIAGRAM**

@startuml

start

:User logs in;

if (Authentication Successful?) then (Yes)

:Display Dashboard;

else (No)

:Show Error Message;

stop

endif

:Select Action;

switch (User Action)

case (View Account Details)

:Fetch Account Data;

:Display Account Details;

case (Transfer Funds)

:Enter Transfer Details;

:Validate Transfer;

if (Sufficient Balance?) then (Yes)

:Process Transaction;

:Update Balances;

:Show Success Message;

else (No)

:Show Insufficient Funds Message;

endif

case (Apply for Loan)

:Enter Loan Details;

:Submit Loan Application;

:Wait for Approval;

if (Loan Approved?) then (Yes)

:Notify User;

else (No)

:Show Rejection Message;

endif

case (Logout)

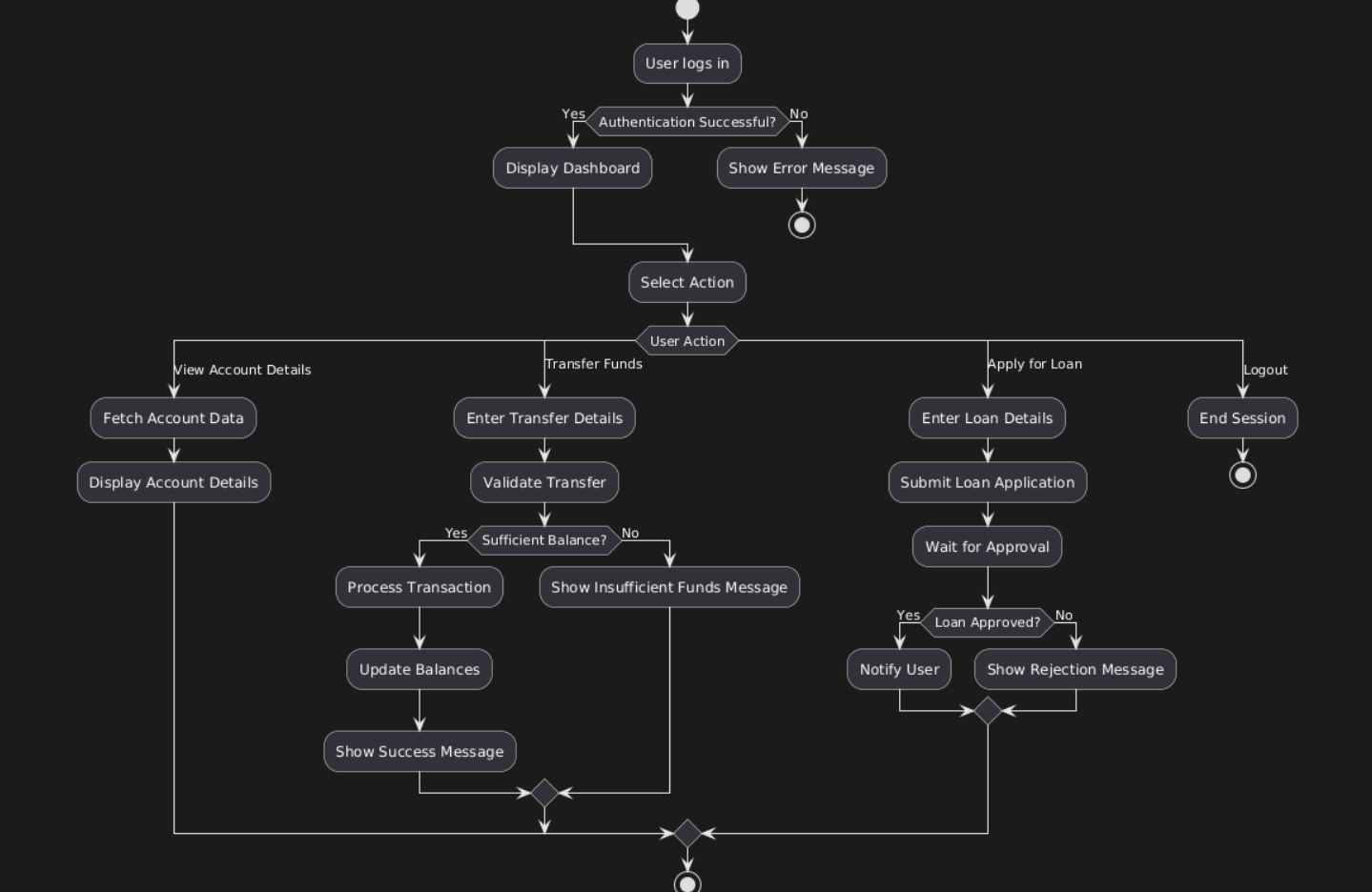
:End Session;

stop

endswitch

stop

@enduml



**COMPONENT DIAGRAM**

@startuml

component "User Interface" as UI

component "Authentication Service" as Auth

component "Banking Services" as Banking

component "Transaction Service" as Transaction

component "Loan Processing" as Loan

component "Database" as DB

UI --> Auth : Authenticate User

Auth --> DB : Validate Credentials

Auth --> UI : Return Authentication Status

UI --> Banking : Request Account Details

Banking --> DB : Fetch Account Information

Banking --> UI : Return Account Details

UI --> Transaction : Request Fund Transfer

Transaction --> Banking : Validate Transaction

Banking --> DB : Update Balances

Transaction --> UI : Return Transaction Status

UI --> Loan : Apply for Loan

Loan --> DB : Store Loan Application

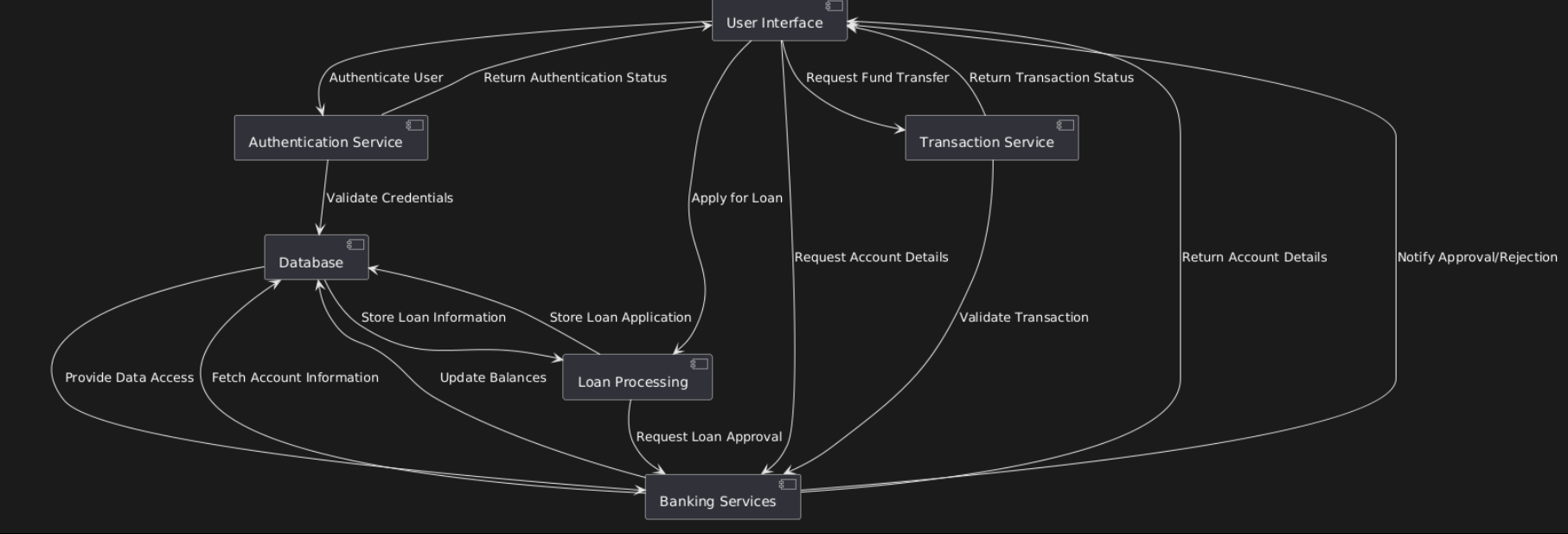
Loan --> Banking : Request Loan Approval

Banking --> UI : Notify Approval/Rejection

DB --> Banking : Provide Data Access

DB --> Loan : Store Loan Information

@enduml



**DEPLOYMENT DIAGRAM**

@startuml

node "Client Device" {

component "Web Browser" as Browser

component "Mobile App" as Mobile

}

node "Web Server" {

component "Application Server" as AppServer

component "Authentication Service" as AuthService

component "Banking Services" as BankingService

component "Transaction Service" as TransactionService

component "Loan Processing Service" as LoanService

}

node "Database Server" {

database "Banking Database" as DB

}

Browser --> AppServer : HTTPS Request

Mobile --> AppServer : API Request

AppServer --> AuthService : Authenticate User

AuthService --> DB : Validate Credentials

AppServer --> BankingService : Request Account Details

BankingService --> DB : Fetch Account Information

AppServer --> TransactionService : Process Fund Transfer

TransactionService --> DB : Update Balances

AppServer --> LoanService : Apply for Loan

LoanService --> DB : Store Loan Application

@enduml

