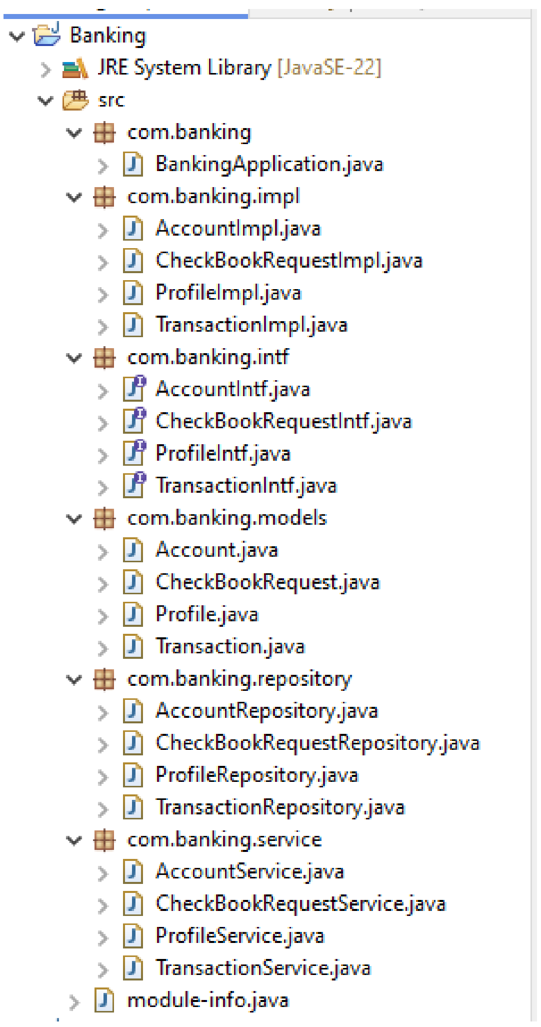
JAVA PROJECT

NET BANKING

- **Profiles:** The system displays the profile details, updates one profile, and shows the remaining profiles after an update.
- **Accounts:** The system adds an account, updates the balance, and then shows the remaining accounts after an update.
- **Transactions:** The system shows the initial transaction, updates one, deletes another, and displays the remaining transactions.
- **CheckBook Requests:** The system manages checkbook requests by showing the initial status, updating one, and displaying the remaining request after deletion.

Project structure:



```
Banking
  JRE System Library [JavaSE-22]
  src
    com.banking
      BankingApplication.java
    com.banking.impl
      AccountImpl.java
      CheckBookRequestImpl.java
      ProfileImpl.java
      TransactionImpl.java
    com.banking.intf
      AccountIntf.java
      CheckBookRequestIntf.java
      ProfileIntf.java
      TransactionIntf.java
    com.banking.models
      Account.java
      CheckBookRequest.java
      Profile.java
      Transaction.java
    com.banking.repository
      AccountRepository.java
      CheckBookRequestRepository.java
      ProfileRepository.java
      TransactionRepository.java
    com.banking.service
      AccountService.java
      CheckBookRequestService.java
      ProfileService.java
      TransactionService.java
    module-info.java
```

Profile Module:

Profile.java :

```
package com.banking.models;
public class Profile {
   private String id;
   private String name;
   private String email;
          // Getters and Setters
   public String getId() {
      return id;
   }
   public Profile(String id, String name, String email) {
                  this.id = id;
                  this.name = name;
                  this.email = email;
          }
          public void setId(String id) {
      this.id = id;
   }
   public String getName() {
      return name;
   }
   public void setName(String name) {
      this.name = name;
   }
   public String getEmail() {
      return email;
   }
   public void setEmail(String email) {
      this.email = email;
   }
}
```

ProfileIntf.java :

```
package com.banking.intf;
import java.util.List;
import com.banking.models.Profile;
public interface ProfileIntf {
   void addProfile(Profile profile);
   void addAllProfiles(List<Profile> profiles);
   Profile getProfile(String id);
   void updateProfile(String id, Profile profile);
   void deleteProfile(String id);
}
```

ProfileService.java :

```
package com.banking.service;
import com.banking.impl.ProfileImpl;
import com.banking.models.Profile;
import com.banking.repository.ProfileRepository;
import java.util.List;
public class ProfileService {
    public void manageProfiles() {
        ProfileImpl profileImpl = new ProfileImpl();
        ProfileRepository profileRepository = new ProfileRepository();
        // Add all profiles
        List<Profile> profiles = profileRepository.getSampleProfiles();
        profileImpl.addAllProfiles(profiles);
        // Read a profile
        Profile profile = profileImpl.getProfile("1");
        System.out.println("Profile Name: " + profile.getName());
        // Update a profile
        profile.setName("John Smith");
        profileImpl.updateProfile("1", profile);
        // Delete a profile
        profileImpl.deleteProfile("2");
    }
}
```

ProfileRepository.java :

```
package com.banking.repository;
import com.banking.models.Profile;
import java.util.ArrayList;
import java.util.List;
public class ProfileRepository {
    public List<Profile> getSampleProfiles() {
        List<Profile> profiles = new ArrayList<>();
        profiles.add(new Profile("1", "John Doe", "john@example.com"));
        profiles.add(new Profile("2", "Jane Doe", "jane@example.com"));
        return profiles;
    }
}
```

ProfileImpl.java :

```
package com.banking.impl;
import com.banking.intf.ProfileIntf;
import com.banking.models.Profile;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class ProfileImpl implements ProfileIntf {
   private Map<String, Profile> profiles = new HashMap<>();
   @Override
   public void addProfile(Profile profile) {
      profiles.put(profile.getId(), profile);
   }
   @Override
   public void addAllProfiles(List<Profile> profilesList) {
      for (Profile profile : profilesList) {
         profiles.put(profile.getId(), profile);
      }
   }
   @Override
   public Profile getProfile(String id) {
      return profiles.get(id);
   }
   @Override
   public void updateProfile(String id, Profile profile) {
      profiles.put(id, profile);
   }
   @Override
   public void deleteProfile(String id) {
      profiles.remove(id);
   }
}
```

Account Module:

- **Account.java**: The Account model class defines the structure of an account, including accountId, accountType, and balance.
- **AccountIntf.java**: The AccountIntf interface declares methods for managing accounts, including adding, retrieving, updating, and deleting accounts.
- **AccountImpl.java**: The AccountImpl class implements the AccountIntf interface and provides the actual logic for managing accounts using a HashMap.
- **AccountRepository.java**: The AccountRepository class returns sample account data that can be used to populate the system.
- **AccountService.java**: The AccountService class demonstrates how to use the AccountImpl class to manage accounts, including adding, reading, updating, and deleting accounts.

Account.java :

```
package com.banking.models;
public class Account {
   private String accountId;
   private String accountType;
   private double balance;
   public Account(String accountId, String accountType, double balance)                {
      this.accountId = accountId;
      this.accountType = accountType;
      this.balance = balance;
   }
   // Getters and Setters
   public String getAccountId() {
      return accountId;
   }
   public void setAccountId(String accountId) {
      this.accountId = accountId;
   }
   public String getAccountType() {
      return accountType;
   }
   public void setAccountType(String accountType) {
      this.accountType = accountType;
   }
   public double getBalance() {
      return balance;
   }
   public void setBalance(double balance) {
      this.balance = balance;
   }
}
```

AccountIntf.java :

```
package com.banking.intf;
import com.banking.models.Account;
import java.util.List;
public interface AccountIntf {
   void addAccount(Account account);
   void addAllAccounts(List<Account> accounts);
   Account getAccount(String accountId);
   void updateAccount(String accountId, Account account);
   void deleteAccount(String accountId);
}
```

AccountImpl.java :

```
package com.banking.impl;
import com.banking.intf.AccountIntf;
import com.banking.models.Account;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
public class AccountImpl implements AccountIntf {
   private Map<String, Account> accounts = new HashMap<>();
   @Override
   public void addAccount(Account account) {
      accounts.put(account.getAccountId(), account);
   }
   @Override
   public void addAllAccounts(List<Account> accountsList) {
      for (Account account : accountsList) {
         accounts.put(account.getAccountId(), account);
      }
   }
   @Override
   public Account getAccount(String accountId) {
      return accounts.get(accountId);
   }
   @Override
   public void updateAccount(String accountId, Account account) {
      accounts.put(accountId, account);
   }
   @Override
   public void deleteAccount(String accountId) {
      accounts.remove(accountId);
   }
}
```

AccountRepository.java :

```
package com.banking.repository;
import com.banking.models.Account;
import java.util.ArrayList;
import java.util.List;
public class AccountRepository {
   public List<Account> getSampleAccounts() {
      List<Account> accounts = new ArrayList<>();
      accounts.add(new Account("101", "Savings", 5000.0));
      accounts.add(new Account("102", "Checking", 1500.0));
      return accounts;
   }
}
```

AccountService.java :

```
package com.banking.service;
import com.banking.impl.AccountImpl;
import com.banking.models.Account;
import com.banking.repository.AccountRepository;
import java.util.List;
public class AccountService {
  public void manageAccounts() {
    AccountImpl accountImpl = new AccountImpl();
    AccountRepository accountRepository = new AccountRepository();
    // Add all accounts
    List<Account> accounts = accountRepository.getSampleAccounts();
    accountImpl.addAllAccounts(accounts);
    // Read an account
    Account account = accountImpl.getAccount("101");
    System.out.println("Account Type: " + account.getAccountType() + ", Balance: " +
account.getBalance());
    // Update an account
    account.setBalance(6000.0);
    accountImpl.updateAccount("101", account);
    // Delete an account
    accountImpl.deleteAccount("102");
    // Display all remaining accounts
    System.out.println("Remaining Accounts: ");
    Account remainingAccount = accountImpl.getAccount("101");
    System.out.println("Account Type: " + remainingAccount.getAccountType() + ",
Balance: " + remainingAccount.getBalance());
  }
}
```

Transaction Module:

- **Transaction.java**: The Transaction model class defines the structure of a transaction, including transactionId, accountId, amount, type (debit/credit), and date.
- **TransactionIntf.java**: The TransactionIntf interface declares methods for managing transactions, including adding, retrieving, updating, and deleting transactions.
- **TransactionImpl.java**: The TransactionImpl class implements the TransactionIntf interface and provides the actual logic for managing transactions using a HashMap.
- **TransactionRepository.java**: The TransactionRepository class returns sample transaction data that can be used to populate the system.
- **TransactionService.java**: The TransactionService class demonstrates how to use the TransactionImpl class to manage transactions, including adding, reading, updating, and deleting transactions.

Transaction.java:

```
package com.banking.models;
public class Transaction {
   private String transactionId;
   private String accountId;
   private double amount;
   private String type; // "debit" or "credit"
   private String date;
   public Transaction(String transactionId, String accountId, double amount, String type, 
String date) {
      this.transactionId = transactionId;
      this.accountId = accountId;
      this.amount = amount;
      this.type = type;
      this.date = date;
   }
   // Getters and Setters
   public String getTransactionId() {
      return transactionId;
   }
   public void setTransactionId(String transactionId) {
      this.transactionId = transactionId;
   }
   public String getAccountId() {
      return accountId;
   }
   public void setAccountId(String accountId) {
      this.accountId = accountId;
   }
   public double getAmount() {
      return amount;
   }
   public void setAmount(double amount) {
      this.amount = amount;
   }
   public String getType() {
      return type;
   }
   public void setType(String type) {
      this.type = type;
   }
   public String getDate() {
      return date;
   }
   public void setDate(String date) {
      this.date = date;
   }
}
```

TransactionIntf.java:

```
package com.banking.intf;
import com.banking.models.Transaction;
import java.util.List;
public interface TransactionIntf {
   void addTransaction(Transaction transaction);
   void addAllTransactions(List<Transaction> transactions);
   Transaction getTransaction(String transactionId);
   void updateTransaction(String transactionId, Transaction transaction);
   void deleteTransaction(String transactionId);
}
```

TransactionImpl.java:

```
package com.banking.impl;
import com.banking.intf.TransactionIntf;
import com.banking.models.Transaction;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
public class TransactionImpl implements TransactionIntf {
   private Map<String, Transaction> transactions = new HashMap<>();
   @Override
   public void addTransaction(Transaction transaction) {
      transactions.put(transaction.getTransactionId(), transaction);
   }
   @Override
   public void addAllTransactions(List<Transaction> transactionsList) {
      for (Transaction transaction : transactionsList) {
         transactions.put(transaction.getTransactionId(), transaction);
      }
   }
   @Override
   public Transaction getTransaction(String transactionId) {
      return transactions.get(transactionId);
   }
   @Override
   public void updateTransaction(String transactionId, Transaction transaction) {
      transactions.put(transactionId, transaction);
   }
   @Override
   public void deleteTransaction(String transactionId) {
      transactions.remove(transactionId);
   }
```

```
        public Map<String, Transaction> getTransactions() {
            return transactions;
        }
        public void setTransactions(Map<String, Transaction> transactions) {
            this.transactions = transactions;
        }
    }
```

TransactionRepository.java:

```
    package com.banking.repository;
    import com.banking.models.Transaction;
    import java.util.ArrayList;
    import java.util.List;
    public class TransactionRepository {
      public List<Transaction> getSampleTransactions() {
        List<Transaction> transactions = new ArrayList<>();
        transactions.add(new Transaction("T001", "101", 500.0, "credit", "2024-08-01"));
        transactions.add(new Transaction("T002", "101", 200.0, "debit", "2024-08-02"));
        transactions.add(new Transaction("T003", "102", 1000.0, "credit", "2024-08-03"));
        return transactions;
      }
    }
```

TransactionService.java:

```
    package com.banking.service;
    import com.banking.impl.TransactionImpl;
    import com.banking.models.Transaction;
    import com.banking.repository.TransactionRepository;
    import java.util.List;
    public class TransactionService {
      public void manageTransactions() {
        TransactionImpl transactionImpl = new TransactionImpl();
        TransactionRepository transactionRepository = new TransactionRepository();
        // Add all transactions
        List<Transaction> transactions = transactionRepository.getSampleTransactions();
        transactionImpl.addAllTransactions(transactions);
        // Read a transaction
        Transaction transaction = transactionImpl.getTransaction("T001");
        System.out.println("Transaction Type: " + transaction.getType() + ", Amount: " +
    transaction.getAmount());
        // Update a transaction
        transaction.setAmount(600.0);
        transactionImpl.updateTransaction("T001", transaction);
        // Delete a transaction
        transactionImpl.deleteTransaction("T002");
        // Display all remaining transactions
        System.out.println("Remaining Transactions: ");
        for (Transaction remainingTransaction : transactionImpl.getTransactions().values()) {
            System.out.println("Transaction ID: " + remainingTransaction.getTransactionId() + ", 
Amount: " + remainingTransaction.getAmount());
        }
    }
}
```

CheckBookRequest Module:

- **CheckBookRequest.java**: The CheckBookRequest model class defines the structure of a checkbook request, including requestId, accountId, numberOfLeaves, requestDate, and status.
- **CheckBookRequestIntf.java**: The CheckBookRequestIntf interface declares methods for managing checkbook requests, including adding, retrieving, updating, and deleting checkbook requests.
- **CheckBookRequestImpl.java**: The CheckBookRequestImpl class implements the CheckBookRequestIntf interface and provides the actual logic for managing checkbook requests using a HashMap.
- **CheckBookRequestRepository.java**: The CheckBookRequestRepository class returns sample checkbook request data that can be used to populate the system.
- **CheckBookRequestService.java**: The CheckBookRequestService class demonstrates how to use the CheckBookRequestImpl class to manage checkbook requests, including adding, reading, updating, and deleting requests.

CheckBookRequest.java :

```
package com.banking.models;
public class CheckBookRequest {
    private String requestId;
    private String accountId;
    private int numberOfLeaves;
    private String requestDate;
    private String status; // "pending", "approved", "rejected"
    public CheckBookRequest(String requestId, String accountId, int numberOfLeaves, String 
requestDate, String status) {
        this.requestId = requestId;
        this.accountId = accountId;
        this.numberOfLeaves = numberOfLeaves;
        this.requestDate = requestDate;
        this.status = status;
    }
    // Getters and Setters
    public String getRequestId() {
        return requestId;
    }
    public void setRequestId(String requestId) {
        this.requestId = requestId;
    }
    public String getAccountId() {
        return accountId;
    }
    public void setAccountId(String accountId) {
        this.accountId = accountId;
    }
    public int getNumberOfLeaves() {
        return numberOfLeaves;
    }
    public void setNumberOfLeaves(int numberOfLeaves) {
        this.numberOfLeaves = numberOfLeaves;
    }
    public String getRequestDate() {
        return requestDate;
    }
    public void setRequestDate(String requestDate) {
        this.requestDate = requestDate;
    }
    public String getStatus() {
        return status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
}
```

CheckBookRequestIntf.java:

```
package com.banking.intf;
import com.banking.models.CheckBookRequest;
import java.util.List;
public interface CheckBookRequestIntf {
    void addCheckBookRequest(CheckBookRequest request);
    void addAllCheckBookRequests(List<CheckBookRequest> requests);
    CheckBookRequest getCheckBookRequest(String requestId);
    void updateCheckBookRequest(String requestId, CheckBookRequest request);
    void deleteCheckBookRequest(String requestId);
}
```

CheckBookRequestImpl.java:

```
package com.banking.impl;
import com.banking.intf.CheckBookRequestIntf;
import com.banking.models.CheckBookRequest;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
public class CheckBookRequestImpl implements CheckBookRequestIntf {
   private Map<String, CheckBookRequest> checkBookRequests = new HashMap<>();
   @Override
   public void addCheckBookRequest(CheckBookRequest request) {
      checkBookRequests.put(request.getRequestId(), request);
   }
   @Override
   public void addAllCheckBookRequests(List<CheckBookRequest> requestsList) {
      for (CheckBookRequest request : requestsList) {
         checkBookRequests.put(request.getRequestId(), request);
      }
   }
   @Override
   public CheckBookRequest getCheckBookRequest(String requestId) {
      return checkBookRequests.get(requestId);
   }
   @Override
   public void updateCheckBookRequest(String requestId, CheckBookRequest request) {
      checkBookRequests.put(requestId, request);
   }
   @Override
   public void deleteCheckBookRequest(String requestId) {
      checkBookRequests.remove(requestId);
   }
         public Map<String, CheckBookRequest> getCheckBookRequests() {
                  return checkBookRequests;
         }
         public void setCheckBookRequests(Map<String, CheckBookRequest>
checkBookRequests) {
                  this.checkBookRequests = checkBookRequests;
         }
}
```

CheckBookRequestRepository.java:

```
package com.banking.repository;
import com.banking.models.CheckBookRequest;
import java.util.ArrayList;
import java.util.List;
public class CheckBookRequestRepository {
   public List<CheckBookRequest> getSampleCheckBookRequests() {
      List<CheckBookRequest> requests = new ArrayList<>();
      requests.add(new CheckBookRequest("C001", "101", 25, "2024-08-05", "pending"));
      requests.add(new CheckBookRequest("C002", "102", 50, "2024-08-06", "approved"));
```

```
        return requests;
    }
}
```

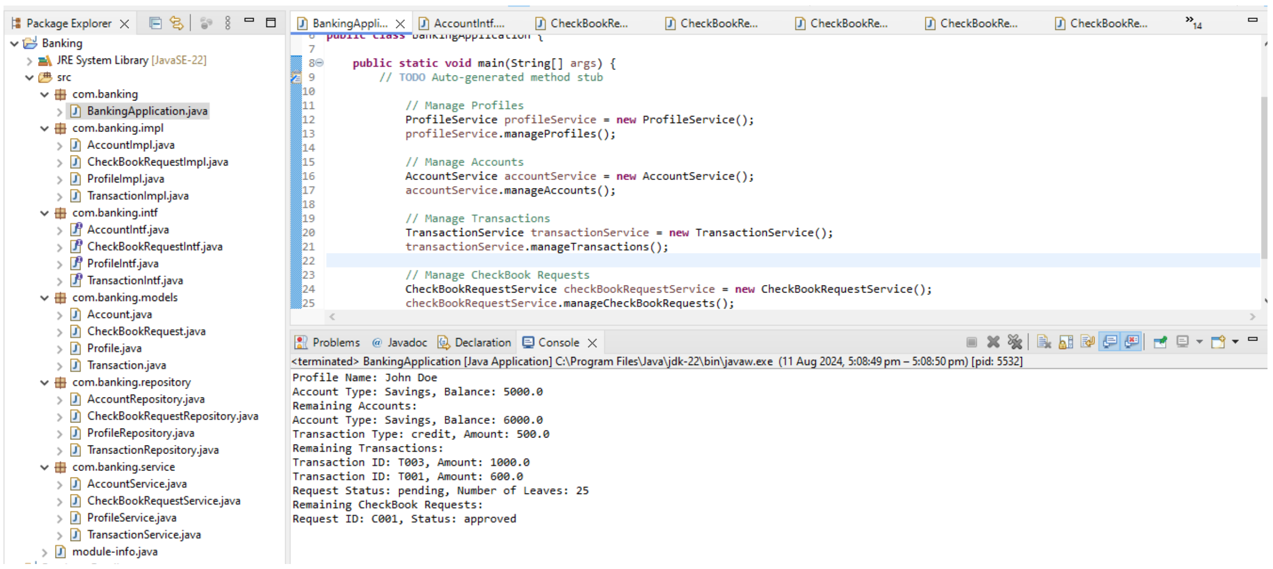
CheckBookRequestService.java:

```
package com.banking.service;
import com.banking.impl.CheckBookRequestImpl;
import com.banking.models.CheckBookRequest;
import com.banking.repository.CheckBookRequestRepository;
import java.util.List;
public class CheckBookRequestService {
    public void manageCheckBookRequests() {
        CheckBookRequestImpl checkBookRequestImpl = new CheckBookRequestImpl();
        CheckBookRequestRepository checkBookRequestRepository = new
CheckBookRequestRepository();
        // Add all checkbook requests
        List<CheckBookRequest> requests =
checkBookRequestRepository.getSampleCheckBookRequests();
        checkBookRequestImpl.addAllCheckBookRequests(requests);
        // Read a checkbook request
        CheckBookRequest request = checkBookRequestImpl.getCheckBookRequest("C001");
        System.out.println("Request Status: " + request.getStatus() + ", Number of Leaves: " +
request.getNumberOfLeaves());
        // Update a checkbook request
        request.setStatus("approved");
        checkBookRequestImpl.updateCheckBookRequest("C001", request);
        // Delete a checkbook request
        checkBookRequestImpl.deleteCheckBookRequest("C002");
        // Display all remaining checkbook requests
        System.out.println("Remaining CheckBook Requests: ");
        for (CheckBookRequest remainingRequest :
checkBookRequestImpl.getCheckBookRequests().values()) {
            System.out.println("Request ID: " + remainingRequest.getRequestId() + ", Status: " +
remainingRequest.getStatus());
        }
    }
}
```

Main Method (BankingApplication.java) :

```
package com.banking;
import com.banking.service.*;
public class BankingApplication {
        public static void main(String[] args) {
                // TODO Auto-generated method stub
            // Manage Profiles
            ProfileService profileService = new ProfileService();
            profileService.manageProfiles();
            // Manage Accounts
            AccountService accountService = new AccountService();
            accountService.manageAccounts();
            // Manage Transactions
            TransactionService transactionService = new TransactionService();
            transactionService.manageTransactions();
            // Manage CheckBook Requests
            CheckBookRequestService checkBookRequestService = new
CheckBookRequestService();
            checkBookRequestService.manageCheckBookRequests();
        }
}
```

OUTPUT:



```
    public static void main(String[] args) {
        // TODO Auto-generated method stub

            // Manage Profiles
            ProfileService profileService = new ProfileService();
            profileService.manageProfiles();

            // Manage Accounts
            AccountService accountService = new AccountService();
            accountService.manageAccounts();

            // Manage Transactions
            TransactionService transactionService = new TransactionService();
            transactionService.manageTransactions();

            // Manage CheckBook Requests
            CheckBookRequestService checkBookRequestService = new CheckBookRequestService();
            checkBookRequestService.manageCheckBookRequests();
```

```
<terminated> BankingApplication [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe  (11 Aug 2024, 5:08:49 pm – 5:08:50 pm) [pid: 5532]
Profile Name: John Doe
Account Type: Savings, Balance: 5000.0
Remaining Accounts:
Account Type: Savings, Balance: 6000.0
Transaction Type: credit, Amount: 500.0
Remaining Transactions:
Transaction ID: T003, Amount: 1000.0
Transaction ID: T001, Amount: 600.0
Request Status: pending, Number of Leaves: 25
Remaining CheckBook Requests:
Request ID: C001, Status: approved
```