**Readme File**

1. **Requirements to run the project**
   1. Java1.8, Spring Boot, JPA, H2
   2. Maven plugin in Eclipse
2. **Process to start the Projects**
   1. Download both projects ATMApi and SBIApi from the path given below
   2. Import the both projects in eclipse with option “Existing project into Eclipse”.
   3. Follow the following steps to build and run the both projects:
      1. Right click the project and select Run As -> Maven Clean to clean the project.
      2. Again right click the project and select Run As -> Maven Install to install the dependencies.
      3. Again right click the project and select Run As -> Maven Build with option Spring-boot:run or you can open the class file ATMApiApplication.java and right click and run as Java application for ATMApi project and SBIApiApplication.java for SBIApi application.
   4. For both application H2 database configurations are specified in application.properties file.
   5. H2 Console URL
      1. <http://localhost:8081/h2-console> (For ATMApi)
      2. <http://localhost:8082/h2-console> (For SBIApi)
   6. For SBIApi data.sql file already placed in the project structure which will insert the meta data information for 5 users.
3. **Process Flow with API details with Assumptions:**
   1. User will require a token to complete the transaction related to Balance Enquiry, Deposits and Withdrawal.
   2. To generate the token:

**URL:** [**http://localhost:8081/accountTransaction/gettoken**](http://localhost:8081/accountTransaction/gettoken)

**RequestType: POST**

**Payload JSON:**

**{**

**"debitCardNumber":"112233445566",**

**"code":"5569",**

**"bank":"SBI"**

**}**

**Assumption 1 :** All the fields are mandatory in above mentioned payload. The bank value always is “SBI” as I made assumption the ATM card will be of SBI only. And I created five profiles within SBIApi which can be used to make the transactions. So you can use any of five debit card numbers mentioned in SBIApi’s database table Account\_Details.

**Assumption 2 :** While generating token, firstly ATMApi application checks whether information of ATM is available or not in ATMApi. If it’s not available than ATMApi fetch the information of ATM from SBIApi and after validating the details of ATM new token will be generated and passed as string.

**Assumption 3 :** After getting the details from SBIApi, ATMApi checks whether Account is Active or not if it’s not return with proper message. If it’s active then it checks ATM is active or not.

**Note:** **Token is valid for only one transaction. For each new transaction, need to generate the new token.**

* 1. To check balance:

**URL:** [**http://localhost:8081/accountTransaction/checkbalance/:token/:debitcardnumber**](http://localhost:8081/accountTransaction/checkbalance/:token/:debitcardnumber)

**RequestType: GET**

After providing the valid token, this api return the current balance by fetching the current balance from SBIApi.

**Assumption 4 :** Account can be updated from anywhere else also, so we need to fetch live data from SBIApi.

* 1. To withdraw amount:

**URL:** [**http://localhost:8081/accountTransaction/withdraw/:token/:debitcardnumber/:amount**](http://localhost:8081/accountTransaction/withdraw/:token/:debitcardnumber/:amount)

**RequestType: GET**

After providing the valid token, this api will update the balance on SBIApi by checking the valid amount and sufficient fund. If there is no sufficient fund it will return with message “Insufficient fund”.

**Assumption 5 :** Withdrawal limit is 10000 per transaction only and without decimal value.

* 1. To deposit amount:

**URL:** [**http://localhost:8081/accountTransaction/deposit/:token/:debitcardnumber/:amount**](http://localhost:8081/accountTransaction/deposit/:token/:debitcardnumber/:amount)

**RequestType: GET**

After providing the valid token, this api will update the balance on SBIApi by checking the valid amount.

**Assumption 6 :** Deposit limit is 50000 per transaction only and without decimal value.

**Note: For each deposit and withdrawal I create an entry in ATMApi table ATM\_TRANSACTION\_DETAILS.**

1. **Executing Test Case:**

Test cases are created under ATMApi section. To execute the test cases open the file DepositWithdrawRestControllerTest.java file. Right click on file and select Run As -> Junit Test.