Transfer Management Services(TMS)

@Author : Anshu Kumar

Publish Date : 28 / 08 / 2018

Table of Content :

1. Introduction :

This document explains the general solution design, technology choices, limitations, deployment instructions for Transfer Management Services (TMS).

TMS core function is to implement REST services for the core banking transaction management between two accounts. It should enable creation of banking accounts and allow transfer between two accounts.

2. Installation & Getting Started :

Below steps needs to be followed in order to run the application.

1. Clone the source code from git in local directory (for example : C:\users)

   $ git clone https://github.com/anshukumar100/TMS.git

2. Move to the root directory of the TSM project (C:\users\TSM) and build the source code using maven command

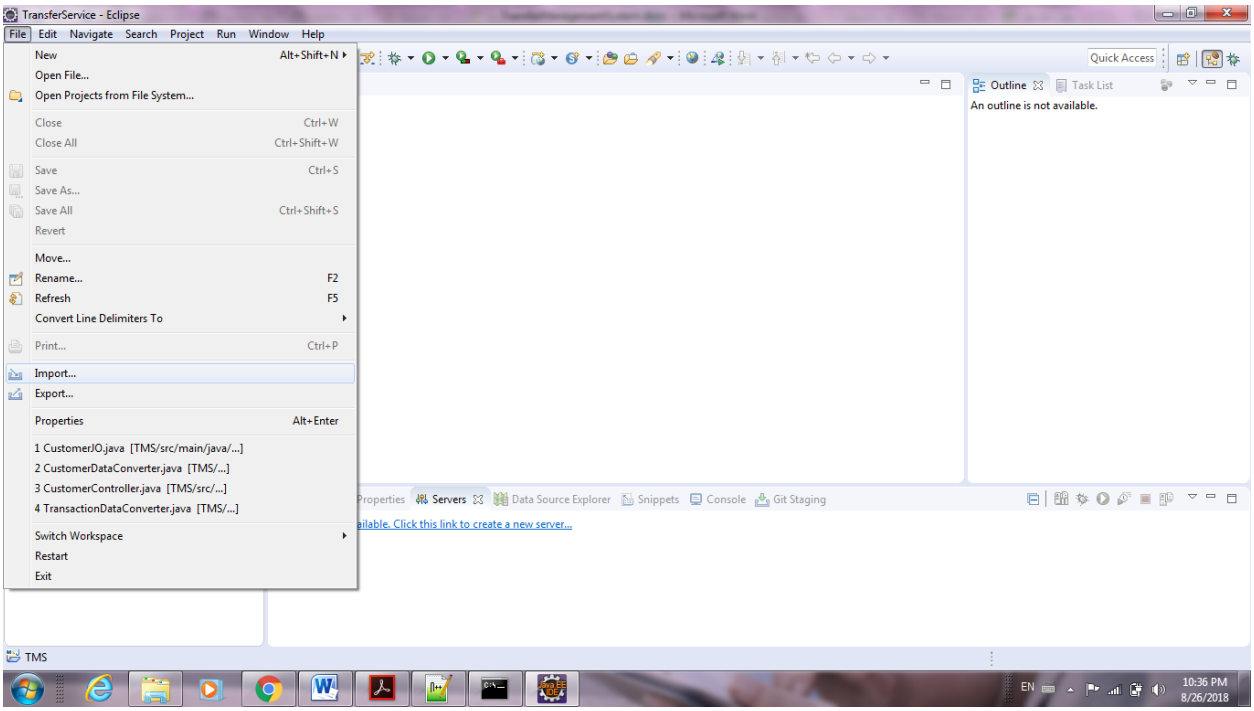
   $ mvn clean install

3. To start the application below command can be used

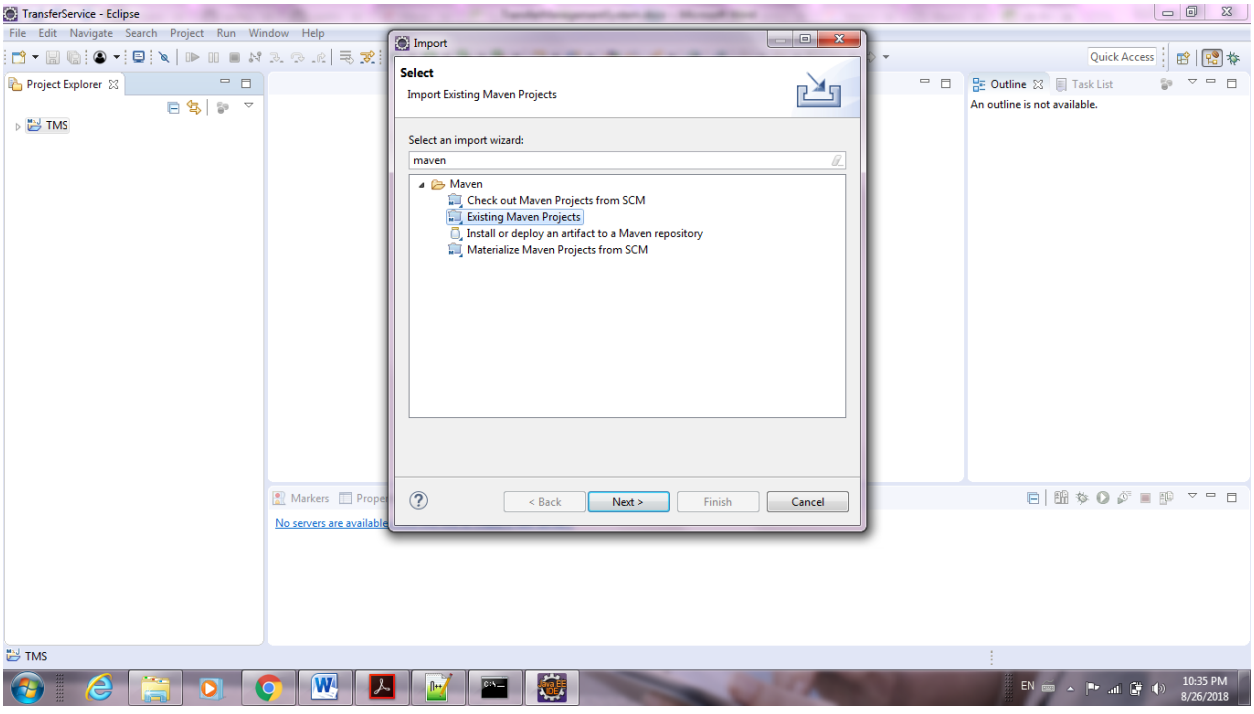
   $ java -jar target/TMS-0.0.1-SNAPSHOT.jar

Alternatively, the source code can be imported to any of the IDE like eclipse and it can be run Java application : steps for which are shown in screenshots

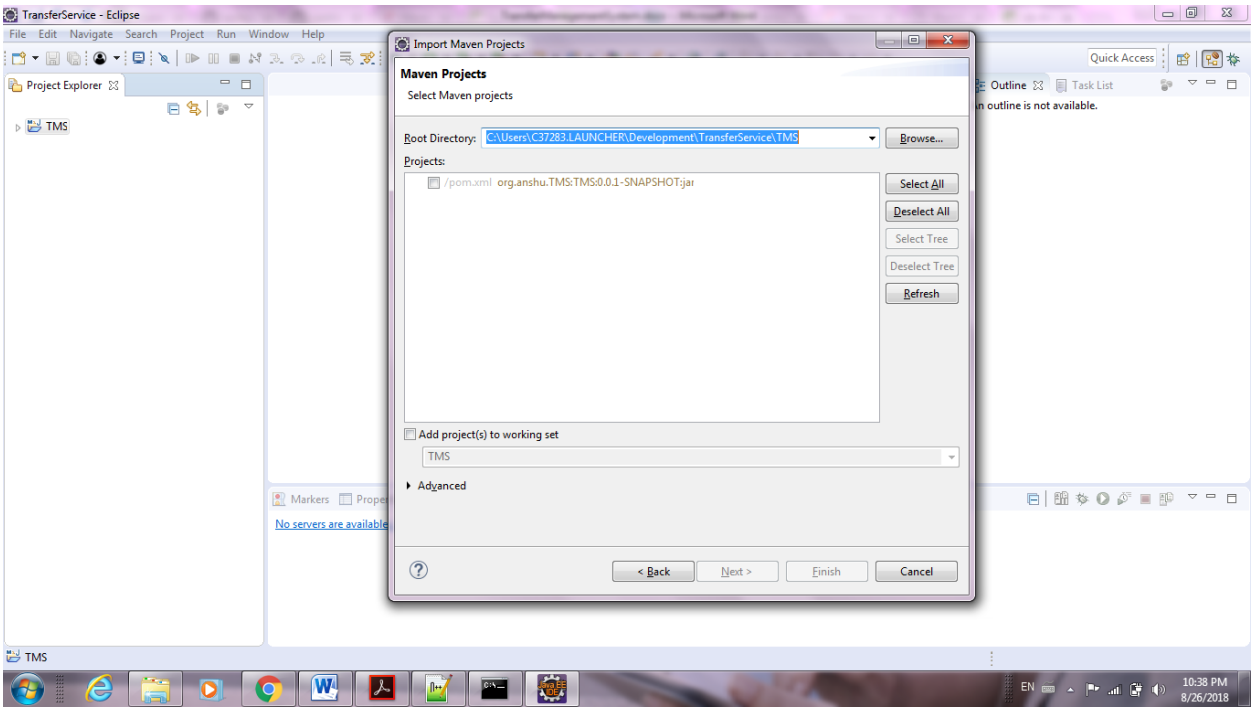
1. Import project as Maven project (screenshot 1 - 3)
2. Then project can be run as Java application (screenshot 4)


TransferService - Eclipse
File Edit Navigate Search Project Run Window Help
New Alt+Shift+N
Open File...
Open Projects from File System...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Import...
Export...
Properties Alt+Enter
1 CustomerJO.java [TMS/src/main/java/...]
2 CustomerDataConverter.java [TMS/...]
3 CustomerController.java [TMS/src/...]
4 TransactionDataConverter.java [TMS/...]
Switch Workspace
Restart
Exit
Quick Access
Outline
Task List
An outline is not available.
Properties
Servers
Data Source Explorer
Snippets
Console
Git Staging
TMS
EN
10:36 PM
8/26/2018

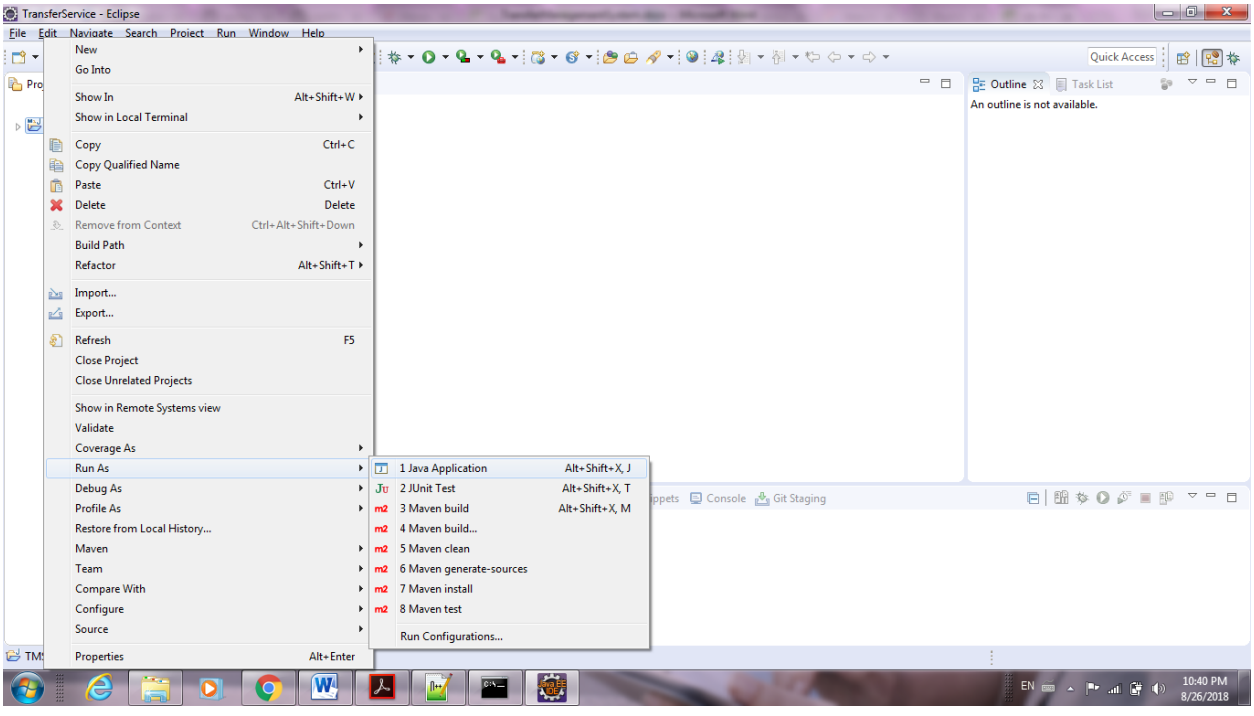
(fig 1)


TransferService - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
TMS
Import
Select
Import Existing Maven Projects
Select an import wizard:
maven
Maven
Check out Maven Projects from SCM
Existing Maven Projects
Install or deploy an artifact to a Maven repository
Materialize Maven Projects from SCM
< Back
Next >
Finish
Cancel
Quick Access
Outline
Task List
An outline is not available.
Markers
No servers are available
TMS
EN
10:35 PM
8/26/2018

(fig 2)


TransferService - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
TMS
Import Maven Projects
Maven Projects
Select Maven projects
Root Directory: C:\Users\C37283.LAUNCHER\Development\TransferService\TMS
Browse...
Projects:
/pom.xml org.anshu.TMS:TMS:0.0.1-SNAPSHOT.jar
Select All
Deselect All
Select Tree
Deselect Tree
Refresh
Add project(s) to working set
TMS
Advanced
< Back
Next >
Finish
Cancel
Quick Access
Outline
Task List
An outline is not available.
Markers
No servers are available
TMS
EN
10:38 PM
8/26/2018

(fig 3)


TransferService - Eclipse
File Edit Navigate Search Project Run Window Help
New
Go Into
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Show in Remote Systems view
Validate
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Maven
Team
Compare With
Configure
Source
Properties Alt+Enter
1 Java Application Alt+Shift+X, J
2 JUnit Test Alt+Shift+X, T
3 Maven build Alt+Shift+X, M
4 Maven build...
5 Maven clean
6 Maven generate-sources
7 Maven install
8 Maven test
Run Configurations...
Quick Access
Outline
Task List
An outline is not available.
Console
Git Staging
EN
10:40 PM
8/26/2018

(fig 4)

3. Technology Stack:

Major technology stack used for building Transfer management services have been outlined below along with the advantages and limitations :

1. Spring Boot
2. Maven
3. H2 in memory database
4. JPA (spring-data-jpa)
5. RESTful Web Service
6. Apache tomcat server
7. Java 8

1. Spring Boot :

To implement TMS spring boot has been used in order to avoid initial the server set-up, spring configurations and enable rapid development so that time to market can be reduced significantly. Also it can be easily integrated with any of the public cloud like Amazon AWS, Microsoft Azure etc. At the same time in-build database and tomcat server can be used to run the application without the need of installing them.

Advantages :

- ✓ Spring Boot makes it easy to create spring-powered, production ready application and services.
- ✓ Enables radically faster and widely accessible experience for all Spring development
- ✓ Provide a range of non-functional features that are common to large classes of projects (e.g. embedded servers, security, metrics, health checks, externalized configuration)
- ✓ Absolutely no code generation and no requirement for XML configuration

Limitations :

- ✓ Time consuming process to convert existing or legacy Spring Framework projects into Spring Boot Applications.
- ✓ Spring Boot is limited to work with Spring based applications (Java/groovy).

2. **Maven :**

Maven has been used as build tool for the application. It is used to manage dependencies of the application. It is essentially a project management and comprehension tool and as such provides a way to help with managing Builds, Dependencies, Documentation, Releases and Distribution.

3. **H2 in memory Database:**

Typical databases involve a lots of setup like install the database, setup schema, setup the tables, application to database connection. Therefore for the TMS in memory H2 database has been used which is provided by spring boot.

Advantages

- ✓ Zero project setup or infrastructure
- ✓ Zero Configuration or maintenance
- ✓ Easy to use for Learning, POCs and Unit Tests
- ✓ Spring Boot provides Simple Configuration to switch between a real database and an in memory database like H2

Limitations :

- ✓ poor query optimizer
- ✓ Concurrency issues.

4. **JPA : Spring data JPA:**

To implement the data access layer the JPS spring-data-jpa has been utilized in the application. It helps in reducing boiler-plate code and reduces the effort and amount needed to maintain data access layer.

Advantages :

- ✓ Support to build repositories based on Spring and JPA
- ✓ Easy migration to other JPAs for data access layer.
- ✓ Input validation support for entity and objects.

5. **RESTful Web Service :**

Rest is an architecture style for a web service to deal with client/server architecture and is designed to use a stateless communication protocol, typically HTTP. Simplicity, lightweight and faster response makes it more likeable option for services.

Advantages :

- ✓ Resource identification using URI.
- ✓ Uniform interface.
- ✓ Self-descriptive messages
- ✓ Stateless interactions through hyperlinks

Limitations :

- ✓ Only works on top of the HTTP protocol.
- ✓ Hard to enforce authorization and security on top of it.
- ✓ Hard to enforce a strict contract between client and server.

6. **Apache tomcat server**

Apache tomcat is an open source implementation of JSP, Java Servlet, Java Expression Language and Java WebSocket technologies. It powers

numerous large-scale, mission-critical web applications across a diverse range of industries and organizations. Open source and easy integration with spring boot has made it the first choice for server implementation. In addition it do not require any additional set-up.

Advantages :

- ✓ Incredibly lightweight, Highly flexible and Open-source.
- ✓ Offers additional security.

Limitations :

- ✓ Less robust than paid servers.
- ✓ Support for server and Configuration challenges

7. Java 8

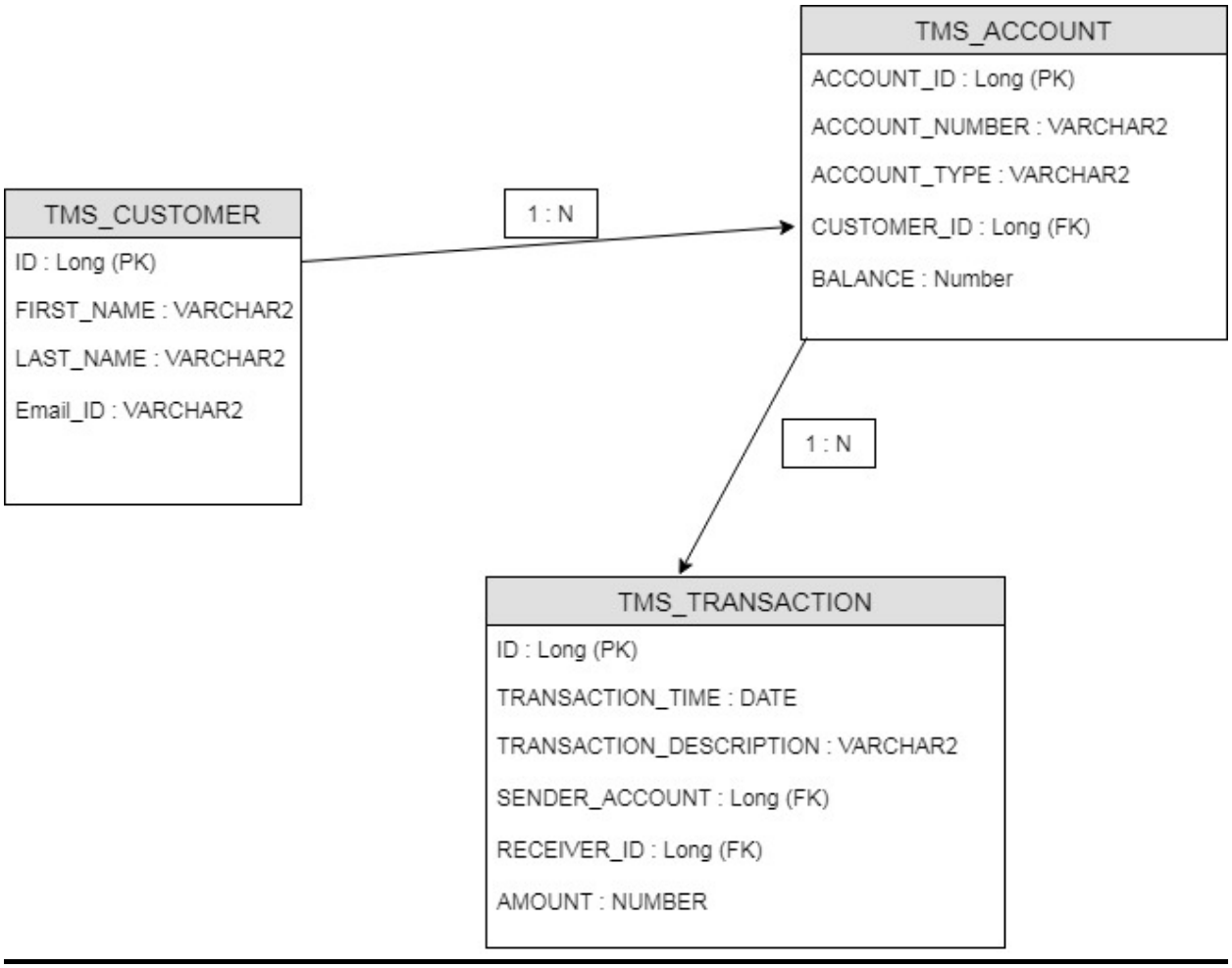
Java 8 is one of the major milestone in the history of Java programming. It has enabled lots of developers across world to reduce boilerplate code and focus more on implementation of business logic. Improved performance, better data structure, memory management has made it developers first choice.

Advantages :

- ✓ Improved performance.
- ✓ Better memory management.
- ✓ Improved productivity.

4. Data Model :

To implement Transfer Management services minimum set of relational tables have been used with mandatory fields, which is described below :



5. Rest Services :

Following the REST convention and HTTP protocol specification, the REST APIs of post-service are designed as the following table.

1. Customer Controller :

1.1

URI : /customers/{customerId}
Description : To get a specific customer
HTTP Method : GET
Request : NA
Response : Status : 200


```
{
        "id": 1,
        "firstName": "Anshu",
        "lastName": "Kumar",
        "emailId": "anshu.kumar@gmail.com",
        "accounts": [
                {
                        "accountId": 2,
                        "accountNumber": "10000000001",
                        "accountType": "SAVING",
                        "balance": 10000
                }
        ],
}
```


1.2

URI : /customers
Description : To get all the customer
HTTP Method : GET

Request : NA
Response : Status : 200


```
[
	{
		"id": 1,
		"firstName": "Anshu",
		"lastName": "Kumar",
		"emailId": "anshu.kumar@gmail.com",
		"accounts": [
			{
				"accountId": 2,
				"accountNumber": "10000000001",
				"accountType": "SAVING",
				"balance": 10000
			}
		],
	},
	{
		"id": 3,
		"firstName": "Ajay",
		"lastName": "Sahoo",
		"emailId": "ajau.sahoo@gmail.com",
		"accounts": [
			{
				"accountId": 4,
				"accountNumber": "10000000000",
				"accountType": "SAVING",
				"balance": 20000
			}
		],
	}
]
```

1.3

URI : /customers
Description : Create a new customer and account
HTTP Method : POST
Request :

```
{
    "firstName":"Ajay",
    "lastName":"Sahoo",
    "emailId":"ajau.sahoo@gmail.com",
    "accounts" :[
        {
            "accountNumber" : "10000000000",
            "accountType" : "SAVING",
             "balance" : "20000"
        }
    ]
}
```

Response : Status : 200

```
{
    "id": 1,
    "firstName": "Anshu",
    "lastName": "Kumar",
    "emailId": "anshu.kumar@gmail.com",
    "accounts": [
        {
            "accountId": 2,
            "accountNumber": "10000000001",
            "accountType": "SAVING",
            "balance": 10000
        }
    ],
}
```

2. Transaction Controller

2.1

URI : /transactions
Description : To create a new transaction between two accounts
HTTP Method : POST
Request :

```
{
        "transactionDescription" : "test transaction",
        "senderAccount" :
        {
                "accountNumber" : "10000000000"
        },

        "receiverAccount" :
        {
                "accountNumber" : "10000000001"
        },

        "amount" : "100"
}
```

Response : Status : 200

6. References :

While building the application and selecting the technology stack below references have been used :

1. https://maven.apache.org
2. https://spring.io/projects/spring-boot
3. https://projects.spring.io/spring-data-jpa/
4. https://spring.io/guides/gs/rest-service/
5. https://www.apache.org/
6. https://www.oracle.com/technetwork/java/javase/overview/java8-2100321.html
7. http://www.springboottutorial.com/spring-boot-and-h2-in-memory-database