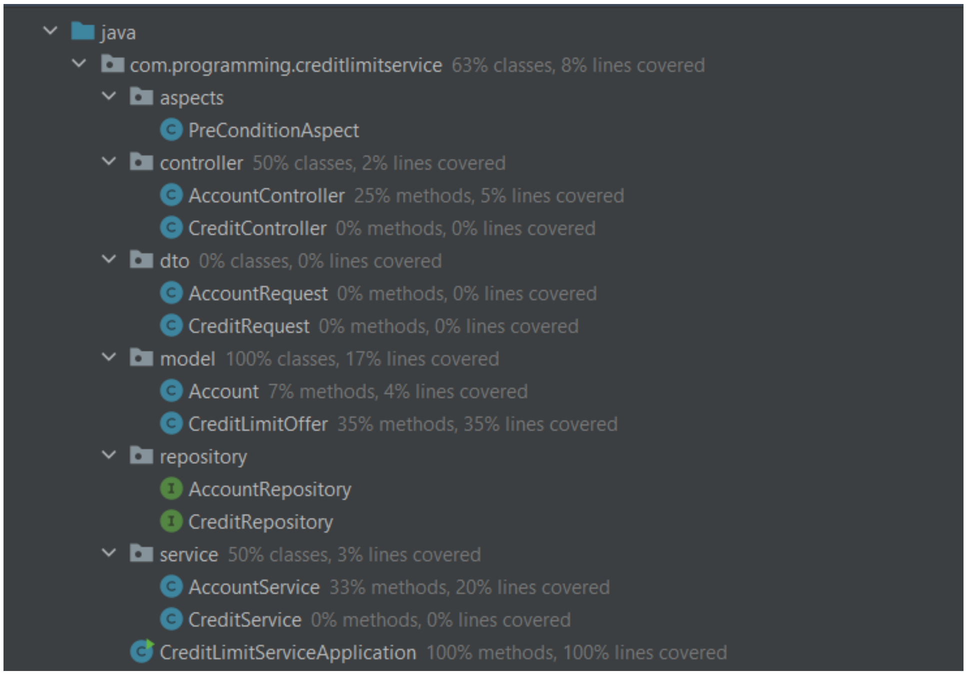
For this project we used Spring Boot, Java, with JPA and Hibernate and PostgreSQL Database in background.

Folder Structure: -



```
java
  com.programming.creditlimitservice  63% classes, 8% lines covered
    aspects
      PreConditionAspect
    controller  50% classes, 2% lines covered
      AccountController  25% methods, 5% lines covered
      CreditController  0% methods, 0% lines covered
    dto  0% classes, 0% lines covered
      AccountRequest  0% methods, 0% lines covered
      CreditRequest  0% methods, 0% lines covered
    model  100% classes, 17% lines covered
      Account  7% methods, 4% lines covered
      CreditLimitOffer  35% methods, 35% lines covered
    repository
      AccountRepository
      CreditRepository
    service  50% classes, 3% lines covered
      AccountService  33% methods, 20% lines covered
      CreditService  0% methods, 0% lines covered
    CreditLimitServiceApplication  100% methods, 100% lines covered
```

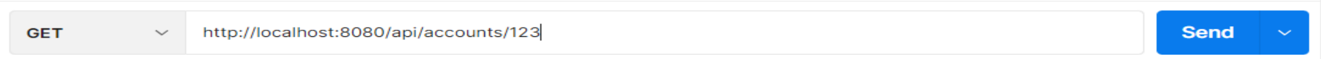
As we can see aspects contains PreConditionAspect class basically which check pre-conditions with the request object received when URL was hit. Also, Controller, Service and Repository classes are kept separate for each of the service and they talk with the help of dto and model classes.

There are two controller class AccountController and CreditController class.

AccountController class takes care of two apis :-

`CreateAccount()` - Basically this is Post Request API and it needs unique Customer ID for creating account. Unique Customer Id can be Aadhar or Pan Card. Since this is very sensitive information, this is hashed and then saved into Database.

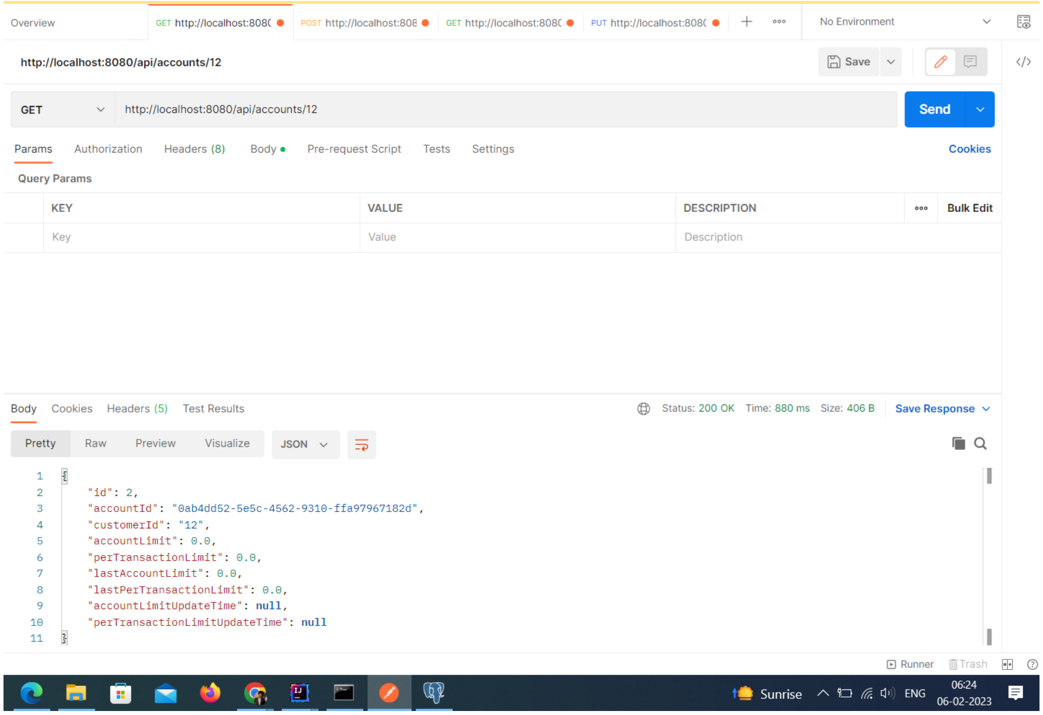
One of the Json Provided during the test: -



GET http://localhost:8080/api/accounts/123

Multiple Testing were done with Postman to see if the provided customer ID is null or "", we should not proceed. And if it's already present in the Database, then we should not create another account for the customer.

`GetAccountByCustomerId()` - This Api takes the customer ID and return back the accounts associated with that customer ID. `If it is not present then BAD REQUEST is returned otherwise the Account in Json Format.`



http://localhost:8080/api/accounts/12

GET http://localhost:8080/api/accounts/12

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 200 OK Time: 880 ms Size: 406 B

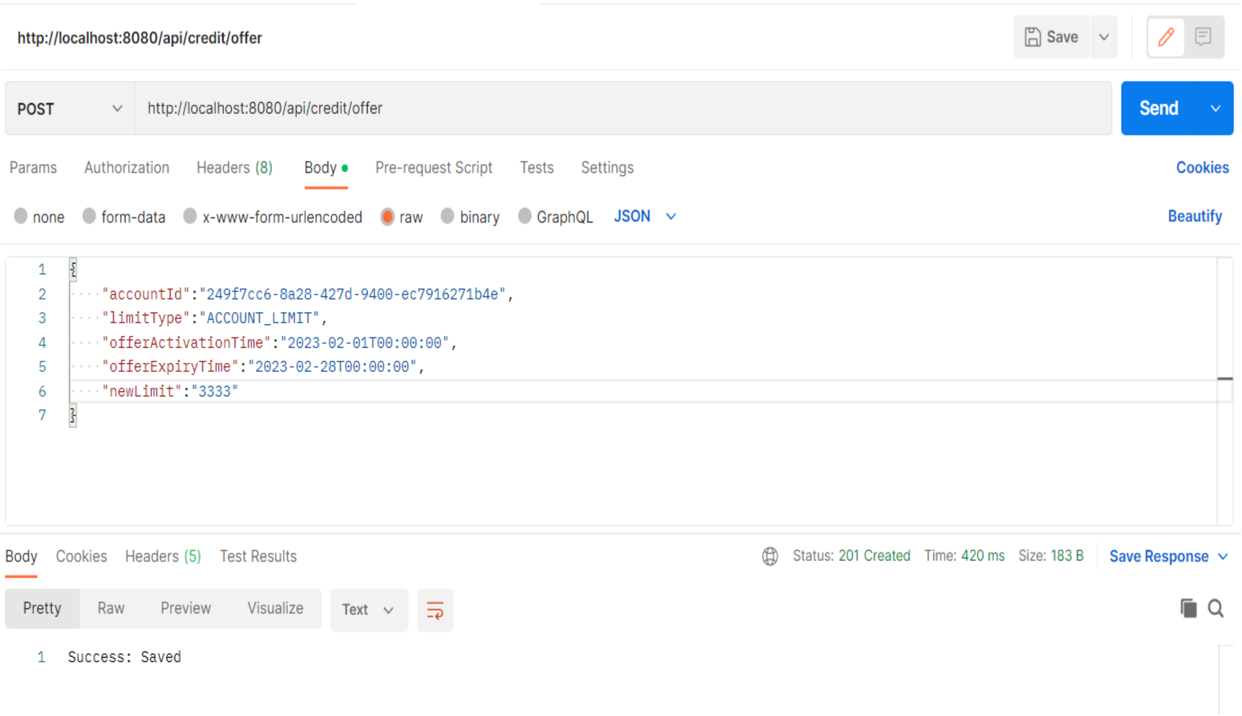
```
{
    "id": 2,
    "accountId": "0ab4dd52-5e5c-4562-9310-ffa97967182d",
    "customerId": "12",
    "accountLimit": 0.0,
    "perTransactionLimit": 0.0,
    "lastAccountLimit": 0.0,
    "lastPerTransactionLimit": 0.0,
    "accountLimitUpdateTime": null,
    "perTransactionLimitUpdateTime": null
}
```

CreditController class has three API's: -

- `createCreditRequest()` - This API takes creditLimitOffer Model class type Request Object and then create request with initial status of PENDING. There is different error check done like Precondition's are specified for arguments, then create only if the limit Offer Value > all the limit offer that the account holds with the status of ACCEPTED.

Also, expiry date should be greater that offer creation date. You can go through the code to check more error handling.

An example of POST REQUEST: -



http://localhost:8080/api/credit/offer

POST http://localhost:8080/api/credit/offer

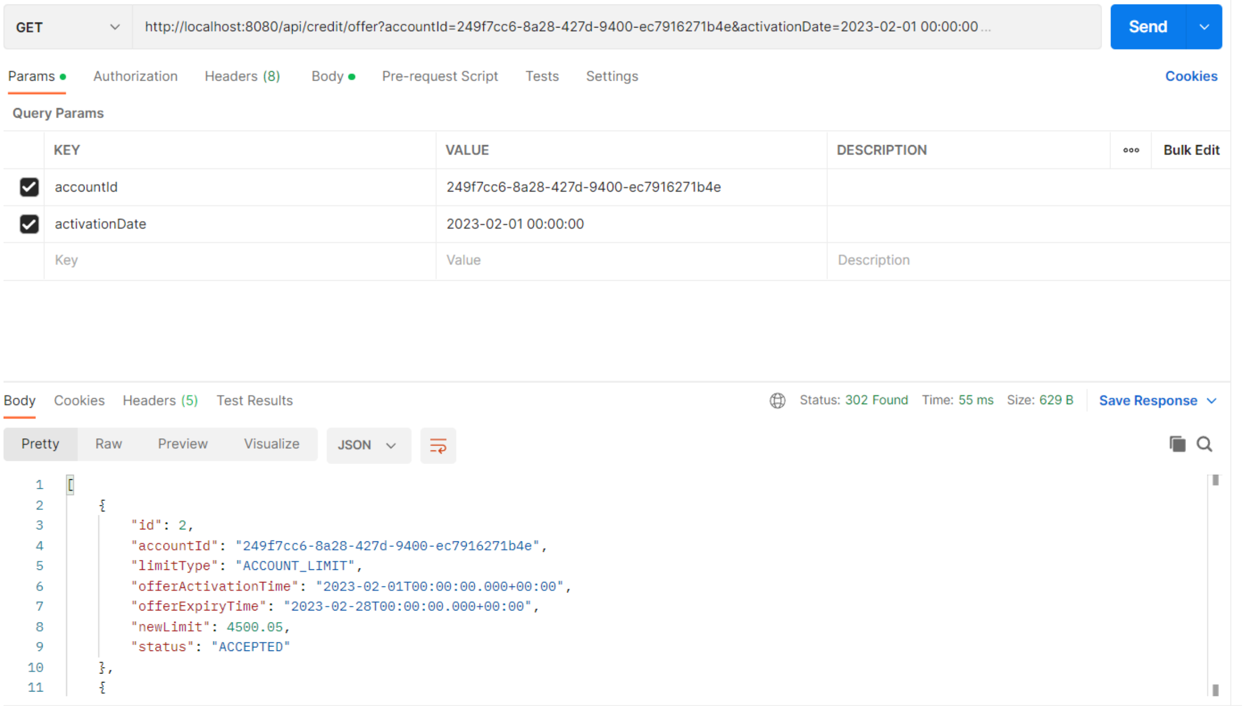
```
{
    "accountId":"249f7cc6-8a28-427d-9400-ec7916271b4e",
    "limitType":"ACCOUNT_LIMIT",
    "offerActivationTime":"2023-02-01T00:00:00",
    "offerExpiryTime":"2023-02-28T00:00:00",
    "newLimit":"3333"
}
```

Status: 201 Created Time: 420 ms Size: 183 B

```
Success: Saved
```

`GetActiveCreditRequest()` - This is get API which takes two parameters – accountID and active Date. There are many checks and error for this API which can be found in the code.

Snippet of one of testing via postman :-



GET http://localhost:8080/api/credit/offer?accountId=249f7cc6-8a28-427d-9400-ec7916271b4e&activationDate=2023-02-01 00:00:00 ...

Query Params

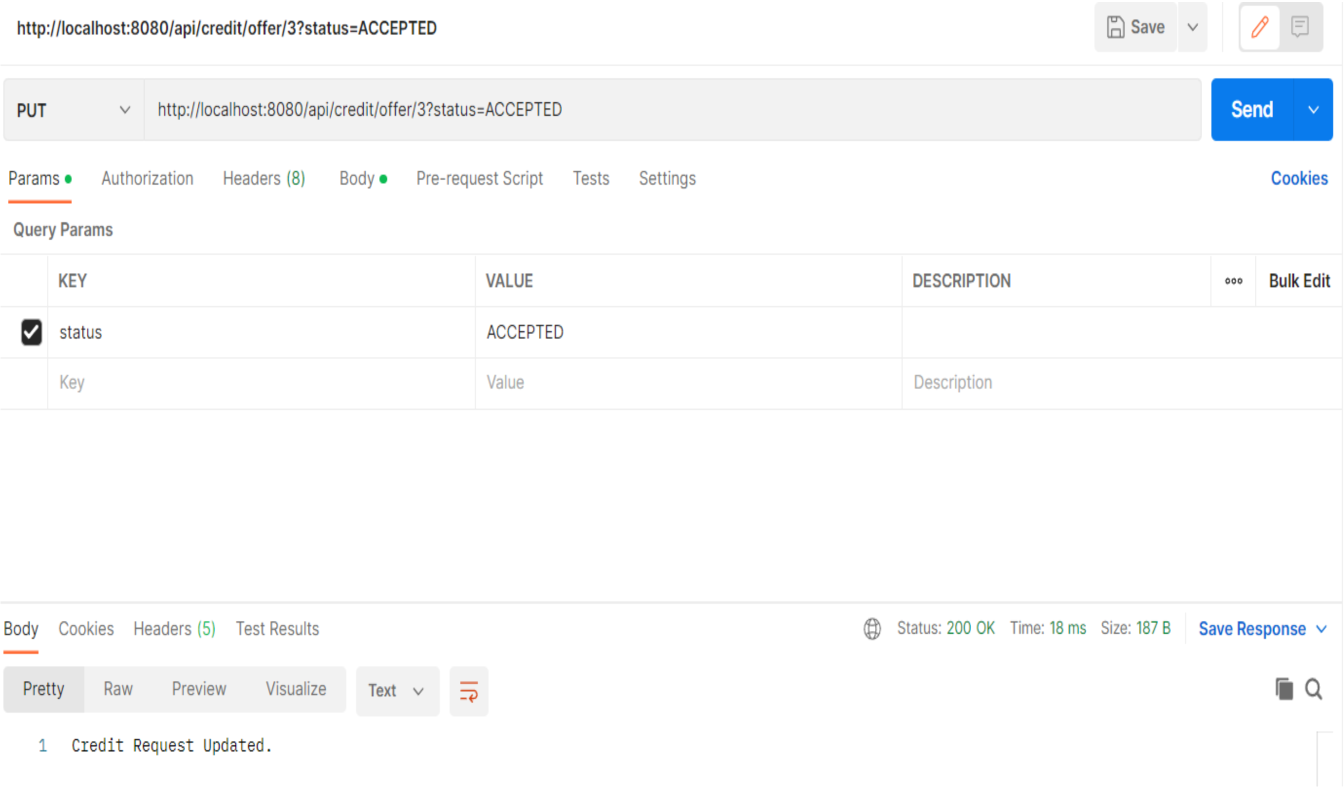
	KEY	VALUE	DESCRIPTION
☑	accountId	249f7cc6-8a28-427d-9400-ec7916271b4e	
☑	activationDate	2023-02-01 00:00:00	
	Key	Value	Description

Status: 302 Found Time: 55 ms Size: 629 B

```
[
    {
        "id": 2,
        "accountId": "249f7cc6-8a28-427d-9400-ec7916271b4e",
        "limitType": "ACCOUNT_LIMIT",
        "offerActivationTime": "2023-02-01T00:00:00.000+00:00",
        "offerExpiryTime": "2023-02-28T00:00:00.000+00:00",
        "newLimit": 4500.05,
        "status": "ACCEPTED"
    },
    {
```

`UpdateCreditRequest` – This is a PUT request, basically it updates in the CreditOfferLimit table, whether that credit offer is ACCEPTED OR REJECTED from customer. Initially when Credit offer are added against a account, its status is made PENDING.

Added the snip of one of the multiple testing via Postman: -



http://localhost:8080/api/credit/offer/3?status=ACCEPTED

PUT http://localhost:8080/api/credit/offer/3?status=ACCEPTED

Query Params

	KEY	VALUE	DESCRIPTION
☑	status	ACCEPTED	
	Key	Value	Description

Status: 200 OK Time: 18 ms Size: 187 B

```
Credit Request Updated.
```

At last, unit test has not been added because of time constraints and github push was done at once.