<u>Documentation For Kirana Register</u>

Kirana register is a spring boot application which **provides API's** to manage transaction and store information in the database. Here input is given with amount and any specified currency which is then later **converted into USD** with the help of an API and stored in database with information such as User's name, transaction type(credited/debited), date of transaction.

Now firstly, let's **look at all the API's** that are provided and the functionalities that they provide and **after that** how the **code** and the **test cases** have been configured and implemented.

<u>API's available</u>-

1) GET: *<u>localhost:8085/transactions/{id}/</u>*
In this API request, a simple Json output is given which is fetched based on the id given in API url.

2) GET: *<u>localhost:8085/transactions/credited/</u>*
In this API request, a list of Json output is given where only credited transactions i.e. transactions with credited transaction type are shown.

3) GET: *<u>localhost:8085/transactions/debited/</u>*
In this API request, a list of Json output is given where only debited transactions i.e. transactions with debited transaction type are shown.

4) POST: *<u>localhost:8085/transactions/</u>*
In this API request, a Json input is given where the Json has following value i.e. user (name), transactionType(either credited or debited), amount, currencyType(here the type of currency is given

which is then converted to USD with the help of an API) and date of transaction.
Json Input:
```
{
    "user": "Shivam",
    "amount": 500,
    "currencyType": "INR",
    "transactionType": "debited",
    "date": "2024-02-01"
}
```

5) PUT: *localhost:8085/transactions/{id}/*
In this API request, the existing data is updated with an id value in the url and a Json input is given where the Json has following value i.e. user (name), transactionType(either credited or debited), amount, currencyType(here the type of currency is given which is then converted to USD with the help of an API) and date of transaction where id in the link and transaction id should be same.

```
{
    "transactionID": 28,
    "user": "Shivam",
    "amount": 200,
    "currencyType": "INR",
    "transactionType": "credited",
    "date": "2024-02-01T11:24:24.396+00:00"
}
```

6) DELETE: *localhost:8085/transactions/{id}/*

In this API, the existing data is deleted on the basis of id that is given in the url.

__Code Implementation__**-**
In this project the application’ structure and code is implemented in the desired descriptive names for variables, classes and functions and abstraction as asked and for preprocessing in the database for table creation data.sql is added in the resource folder.

And test cases are written in the same structure inside “**…/src/test/**” package. And I have covered all of the test cases using MockExtension and the mock API. The controller and the services both have their test cases written in their own different packages as you will see in the project.