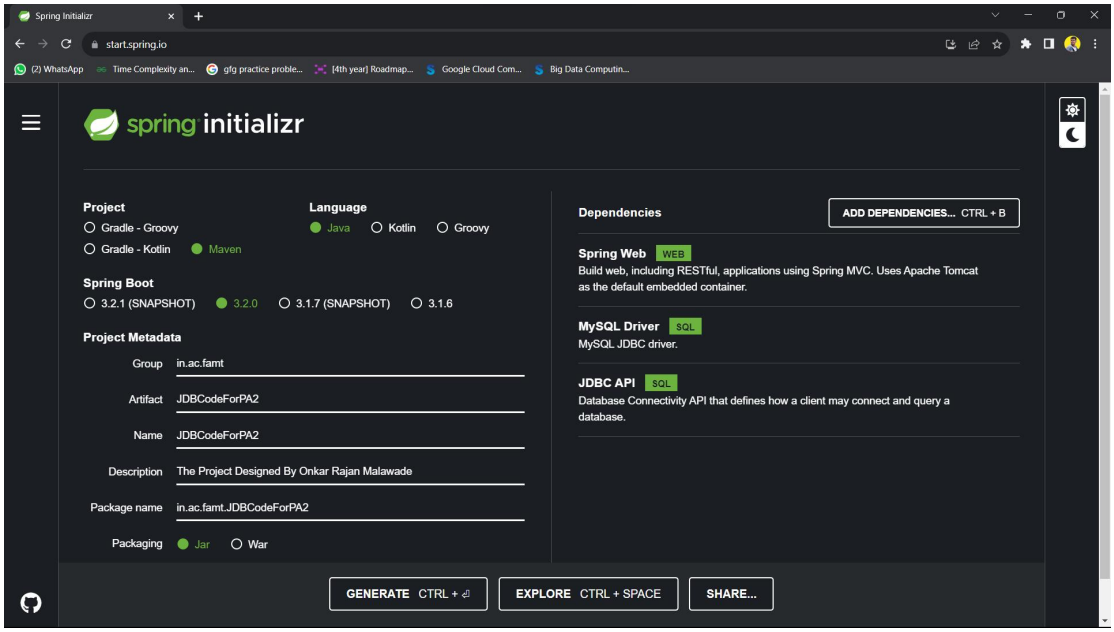
Q.2. Write a Java application for updating and selecting records from a database where BillHeader consists of information like billId, custNm, billdt, billAmt and BillDetails table consist of information like billId, billNo, prodCode, quantity, rate.

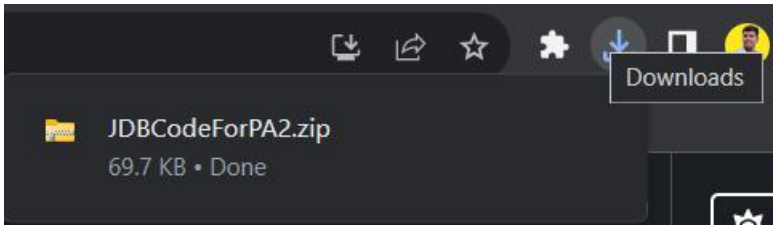
Program 2:

Step 1:Create Spring Boot Project

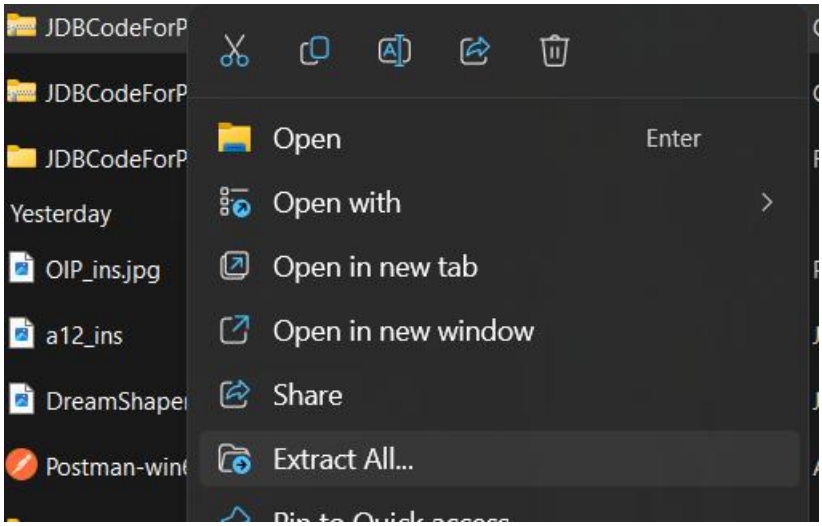


Step 2: Click on the "GENERATE"

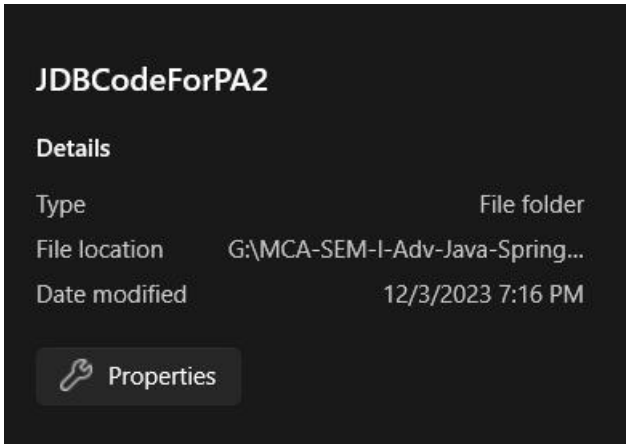
Step 3: It download Zip file from the Browser like given below :



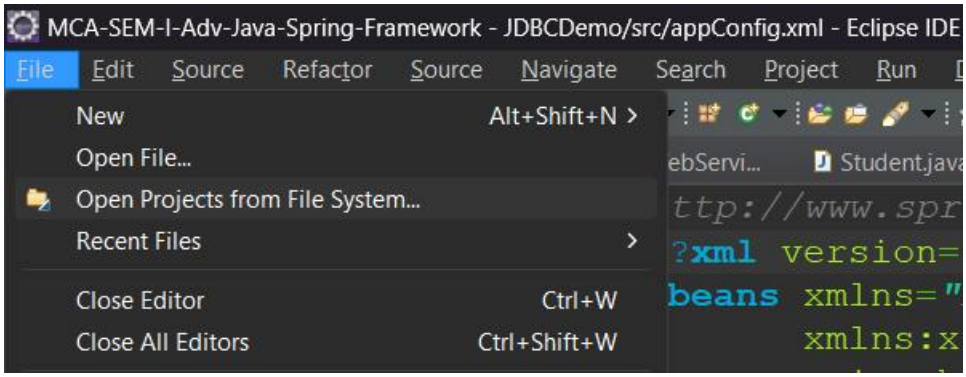
Step 4: Extract File and Open it.



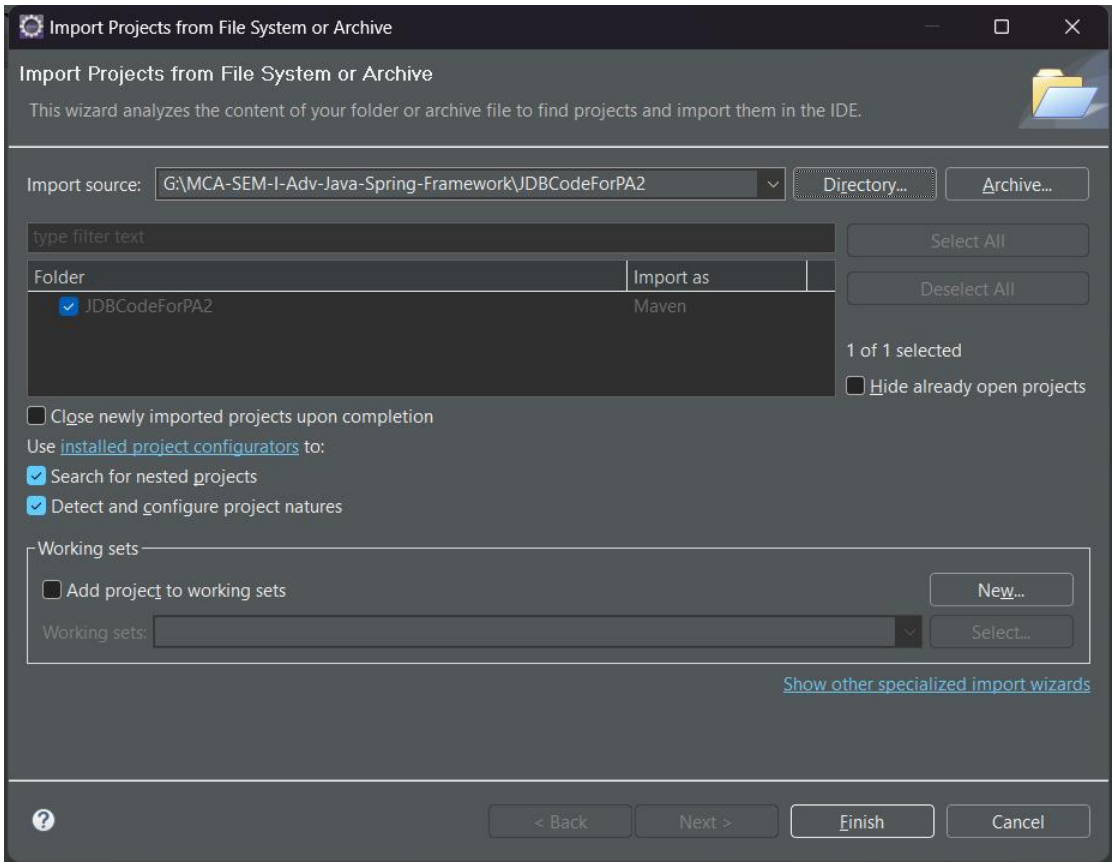
Step 5: Extracted file open and paste to your work-space(eclipse).



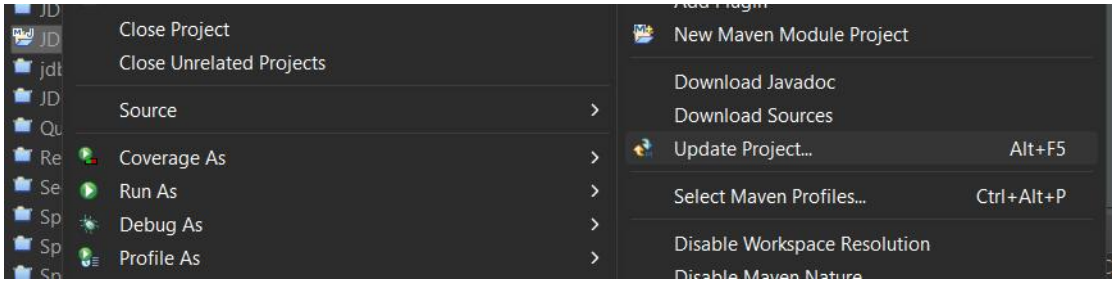
Step 6: Open Eclipse IDE and Click on the Open Projects from File System…



Step 7: Select folder in the Directory and Click Select Folder on it. After that Click on The Finish.

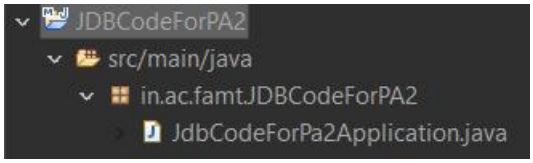


Step 8: Right Click to the Project Follow the given steps:



Step 9 : Your Project is Updated Successfully in the Work-Space.

Open Folder Structure -> Like Given below:



Step 10 : Create Class BillController.java

```
package in.ac.famt.JDBCodeForPA2;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/bill")
public class BillController {

@Autowired
private BillService billService;

// Update operation
@PutMapping("/update/{billId}")
public void updateBill(@PathVariable int billId,
@RequestParam String custNm,
@RequestParam double billAmt) {
billService.updateBill(billId, custNm, billAmt);
}

// Select operation
@GetMapping("/details/{billId}")
public List<BillDetails> getBillDetails(@PathVariable int billId) {
return billService.getBillDetails(billId);
}
}
```

Step 11 : Create Class BillDetails.java

```
package in.ac.famt.JDBCodeForPA2;

public class BillDetails {
private int billNo;
private String prodCode;
private int quantity;
private double rate;
```

```
public int getBillNo() {
return billNo;
}
public void setBillNo(int billNo) {
this.billNo = billNo;
}
public String getProdCode() {
return prodCode;
}
public void setProdCode(String prodCode) {
this.prodCode = prodCode;
}
public int getQuantity() {
return quantity;
}
public void setQuantity(int quantity) {
this.quantity = quantity;
}
public double getRate() {
return rate;
}
public void setRate(double rate) {
this.rate = rate;
}
}
```

Step 12 : Create Class BillHeader.java

```
package in.ac.famt.JDBCodeForPA2;

public class BillDetails {
private int billNo;
private String prodCode;
private int quantity;
private double rate;
public int getBillNo() {
return billNo;
}
public void setBillNo(int billNo) {
this.billNo = billNo;
}
public String getProdCode() {
return prodCode;
}
public void setProdCode(String prodCode) {
this.prodCode = prodCode;
}
public int getQuantity() {
return quantity;
}
public void setQuantity(int quantity) {
this.quantity = quantity;
}
public double getRate() {
return rate;
}
public void setRate(double rate) {
```

```
this.rate = rate;
}
}
```

Step 13 : Create Class BillRepository.java

```
package in.ac.famt.JDBCodeForPA2;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Date;
import java.util.List;

@Repository
public class BillRepository {

@Autowired
private JdbcTemplate jdbcTemplate;

// Update operation
public void updateBillDetails(int billId, String custNm, Date billdt, double billAmt) {
String updateHeaderSql = "UPDATE BillHeader SET custNm = ?, billdt = ?, billAmt
= ? WHERE billId = ?";
jdbcTemplate.update(updateHeaderSql, custNm, billdt, billAmt, billId);
}

// Select operation
@SuppressWarnings("deprecation")
public List<BillDetails> getBillDetails(int billId) {
String selectDetailsSql = "SELECT * FROM BillDetails WHERE billId = ?";
return jdbcTemplate.query(selectDetailsSql, new Object[]{billId}, new
BillDetailsRowMapper());
}

// RowMapper for BillDetails
private static class BillDetailsRowMapper implements RowMapper<BillDetails> {
@Override
public BillDetails mapRow(ResultSet rs, int rowNum) throws SQLException {
BillDetails billDetails = new BillDetails();
billDetails.setBillNo(rs.getInt("billNo"));
billDetails.setProdCode(rs.getString("prodCode"));
billDetails.setQuantity(rs.getInt("quantity"));
billDetails.setRate(rs.getDouble("rate"));
return billDetails;
}
}
}
```

Step 14 : Create Class BillService.java

```
package in.ac.famt.JDBCodeForPA2;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.Date;
import java.util.List;

@Service
public class BillService {

@Autowired
private BillRepository billRepository;

// Update operation
public void updateBill(int billId, String custNm, double billAmt) {
// Assume billdt is the current date
billRepository.updateBillDetails(billId, custNm, new Date(), billAmt);
}

// Select operation
public List<BillDetails> getBillDetails(int billId) {
return billRepository.getBillDetails(billId);
}
}
```

Step 15 : Create Class JdbCodeForPa2Application.java

```
package in.ac.famt.JDBCodeForPA2;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class JdbCodeForPa2Application {

        public static void main(String[] args) {

                SpringApplication.run(JdbCodeForPa2Application.class, args);

        }

}
```

Step 16 : Create Class application.properties

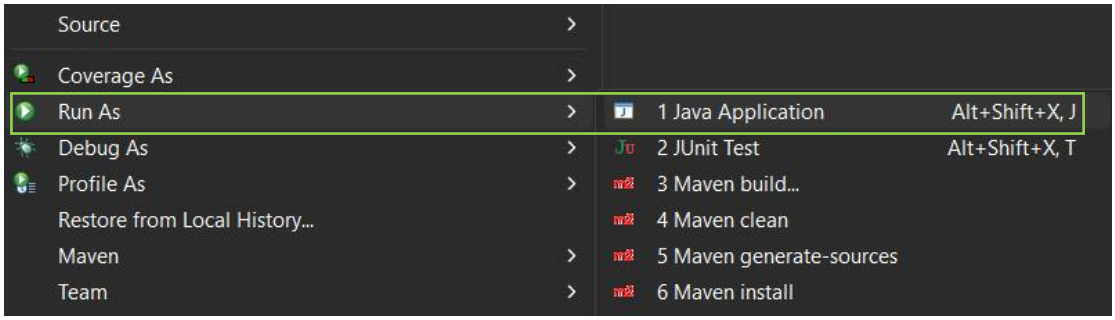
```
# Database Configuration
spring.datasource.url=jdbc:mysql://localhost:3306/billdb
spring.datasource.username=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

Note : Before Going to run this program Add Database in it.:

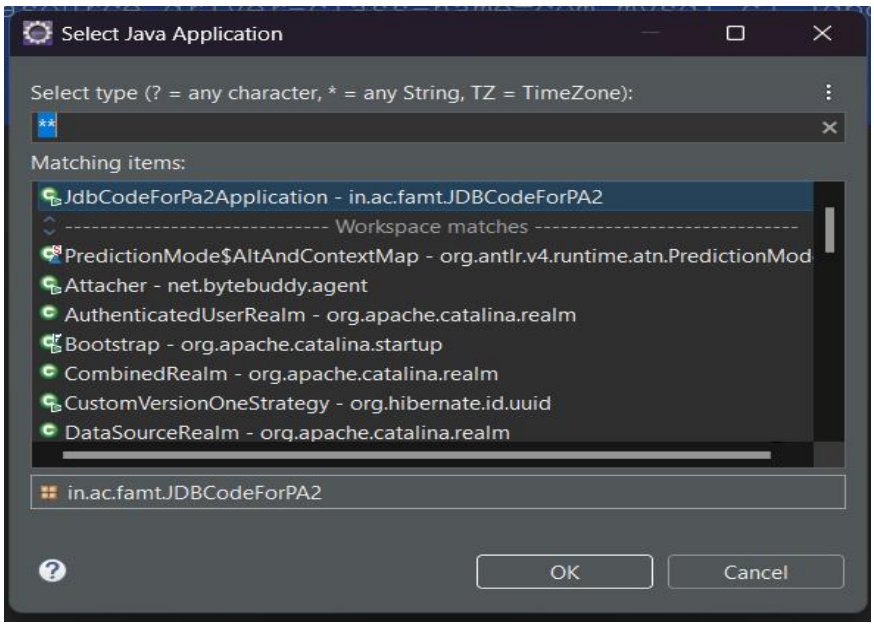
```
-- Create BillHeader table
CREATE TABLE BillHeader (
    billId INT PRIMARY KEY,
    custNm VARCHAR(255),
    billdt DATE,
    billAmt DOUBLE
);
-- Create BillDetails table
CREATE TABLE BillDetails (
    billId INT,
    billNo INT,
    prodCode VARCHAR(255),
    quantity INT,
    rate DOUBLE,
    PRIMARY KEY (billId, billNo),
    FOREIGN KEY (billId) REFERENCES BillHeader(billId)
);
```

And also Add values.

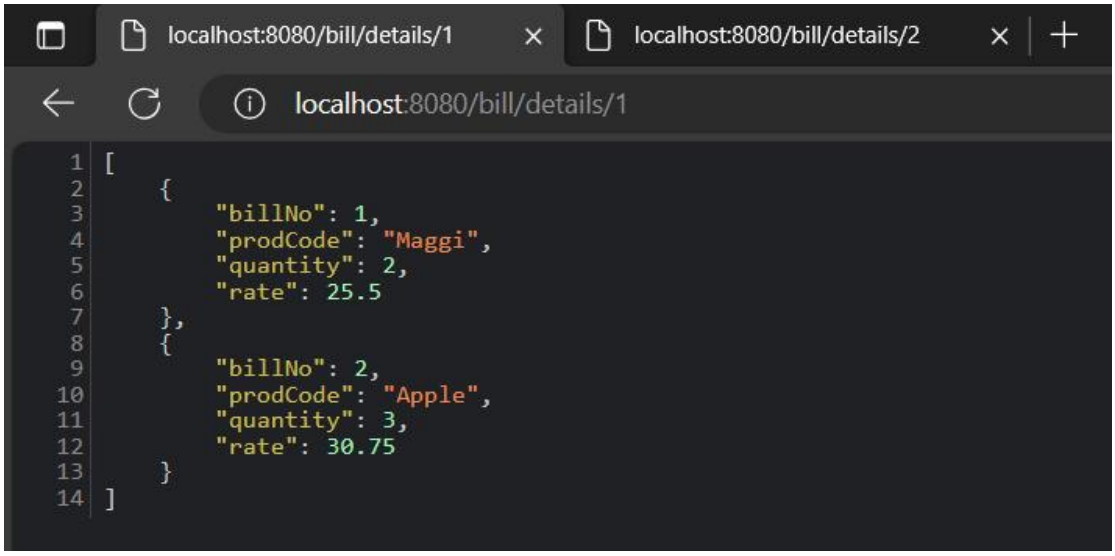
Step 17 : Run the Program with following steps:



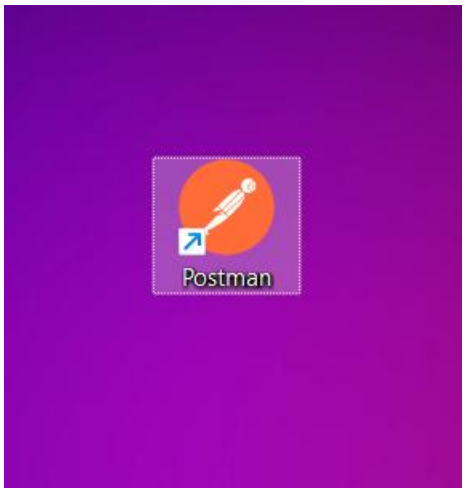
Step 18 : Select Our Application we want to Run and Click on it(OK).



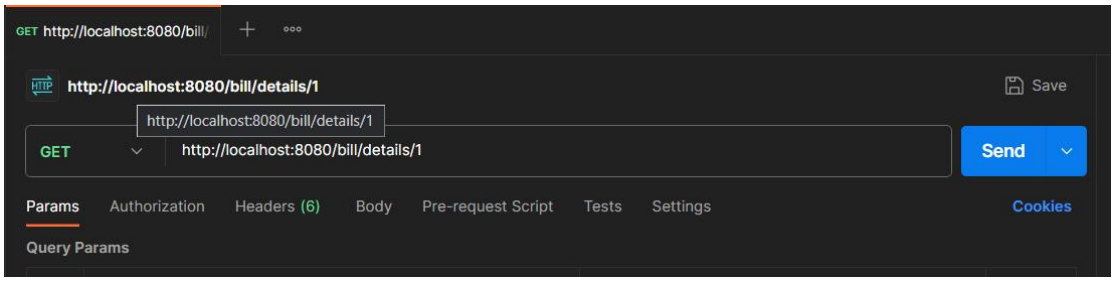
Step 19 : Open Browser: Type:



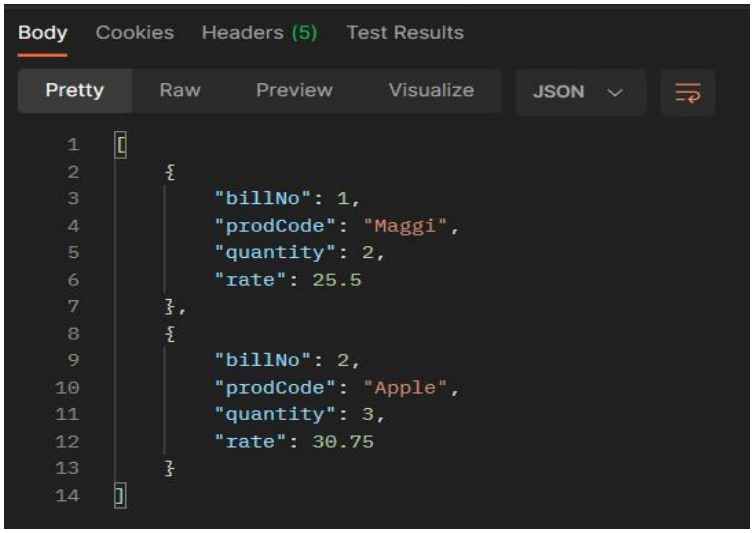
Step 20 : Another Method is Open Postman App



Step 21 : Select Get Method for fetching details from the Server as given below:



Step 22 : Select Get Method for fetching details from the Server as given below:



Step 23 : Select Put Method for Add details from the Server as given below: