Distributed System

a00258751 |

Mobile Management system

Aviral rajput

2019

Contents

[Abstract 2](#_Toc6227487)

[Main Functions: 2](#_Toc6227488)

[GUI: 2](#_Toc6227489)

[Screenshots: 2](#_Toc6227490)

[Main Page: 2](#_Toc6227491)

[1. Requirement 6](#_Toc6227492)

[Completion 6](#_Toc6227493)

[Table Creation: 6](#_Toc6227494)

[Maintain the database 7](#_Toc6227495)

[2. Requirement 7](#_Toc6227496)

[POST: 7](#_Toc6227497)

[PUT 8](#_Toc6227498)

[GET: 8](#_Toc6227499)

[3. Requirements 8](#_Toc6227500)

[Completion 9](#_Toc6227501)

[POST 9](#_Toc6227502)

[PUT 9](#_Toc6227503)

[Delete 10](#_Toc6227504)

# Abstract

This project implements the learning outcome of what all has been gained and learnt in distributed system module. This project implements client and server architecture with the use of rest service. Making using of controller, model and a service class in server and JAX api to display the data. and a client application to interact with the server where client makes GET,POST, PUT and DELETE requests. The project stores and reflects the information of mobiles running on Tomcat 8.5 server. The information in stores and queried on HSQLDB which has the table containing the information of Mobile Id, Name, Price and description. The information is retrieved are processes using XMLPullParser used to identify the tags.

# Main Functions:

**Get** – Get all the information from the database

**Delete** - Enter an Id number and it is deleted from the system.

**Put** – Enter the ID and can alter the other details and will be updated to table

**Post** – Enter the details for a new mobile and create it on the server and saves to database.

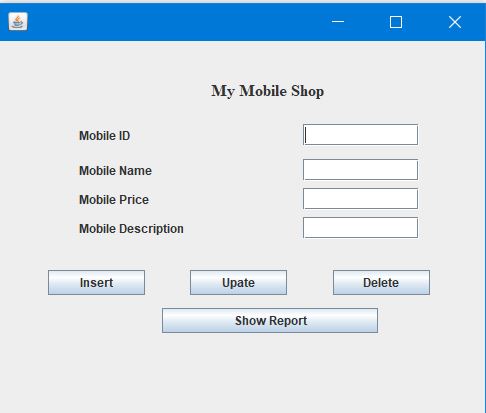
**Delete All** – Removes all data from the tables.

# GUI:

Client app is a single page application made with swing

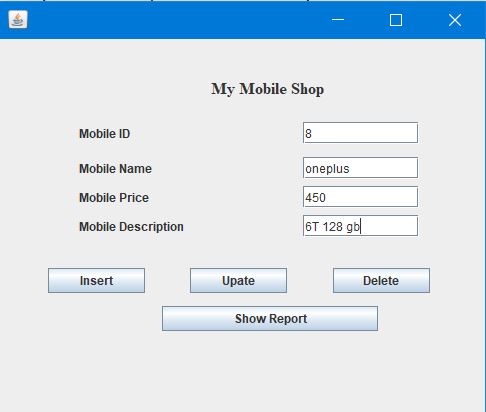
## Screenshots:

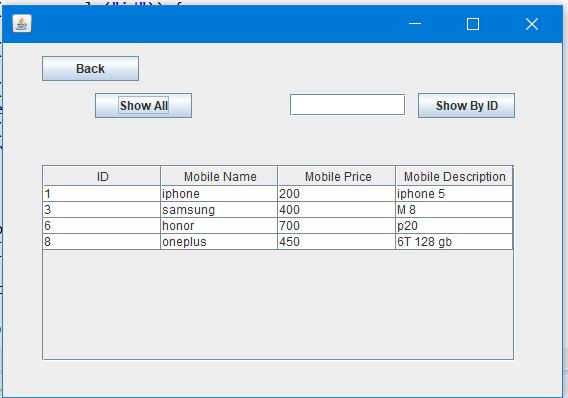
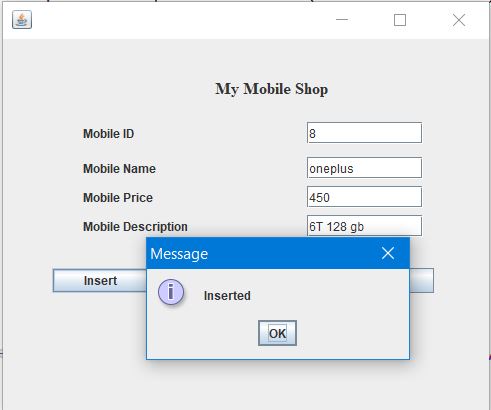
### Main Page:

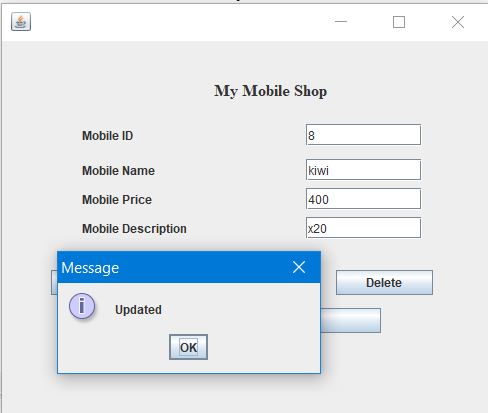
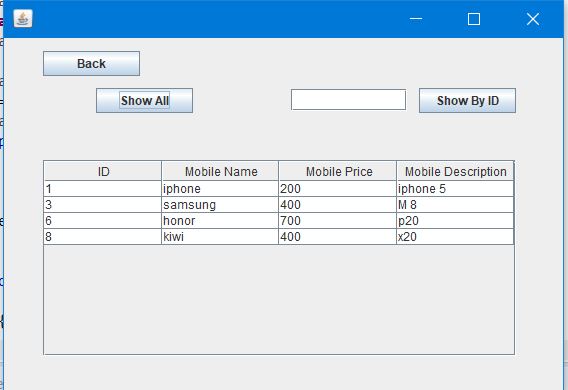


This is the main page which opens up when the application is started, this page has all the functionalities and is interacting with the server directly.

#### Insert Mobile Details Page:

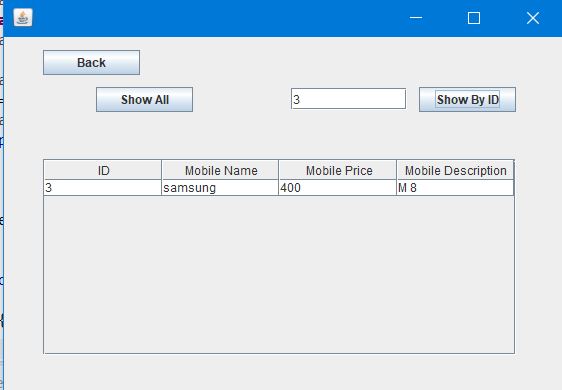
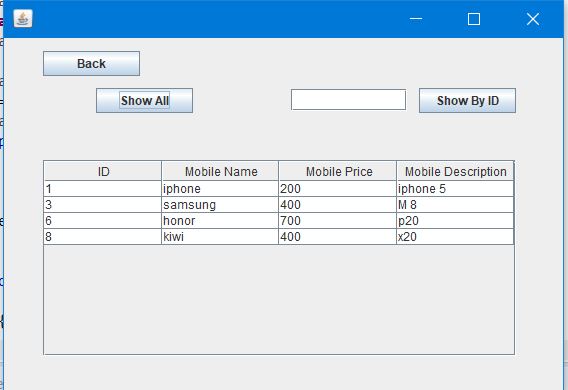
Here is how the data is inserted into the system by using post method



Update Mobile Details (PUT): ****

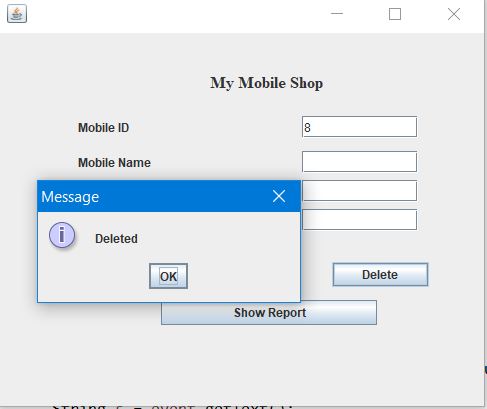
Based upon the data that already exists in the system we can take the id and update the associated details with update button using PUT method.

#### Show Details (GET):

****

In this page, the user has two options where he can chose to display the details by Id or display the all the details that are available in the mobile table with the use of GET.

#### Delete Mobile Details (DELETE):

****

In this page the user can delete a particular mobile from the table by using the ID.

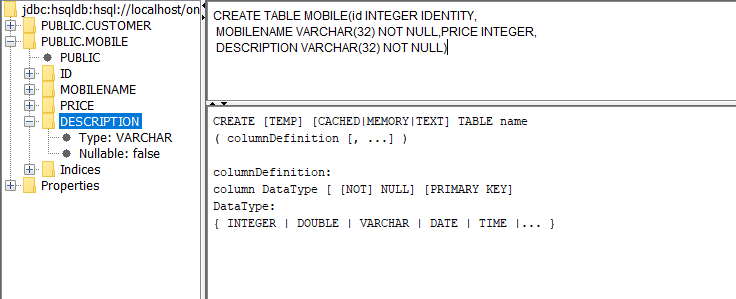
# Requirement

Tables created to interact with the server and handle the operations for insert, update and delete the data.

## Completion

### Table Creation:

Query used to create the table:

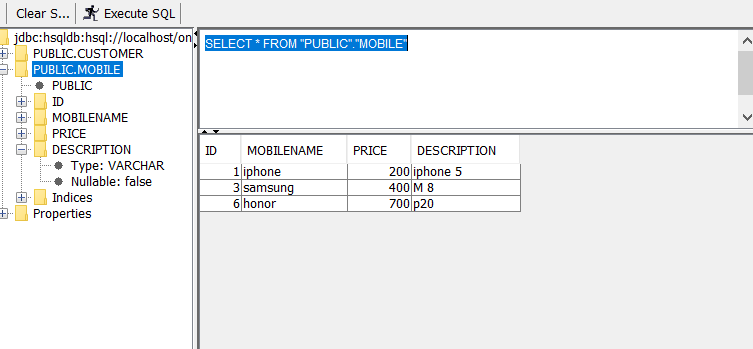


Creation SQL

CREATE TABLE MOBILE(id INTEGER IDENTITY, MOBILENAME VARCHAR(32) NOT NULL,PRICE INTEGER, DESCRIPTION VARCHAR(32) NOT NULL)

**Data in the Mobile Table:**

Select \* from MOBILE;



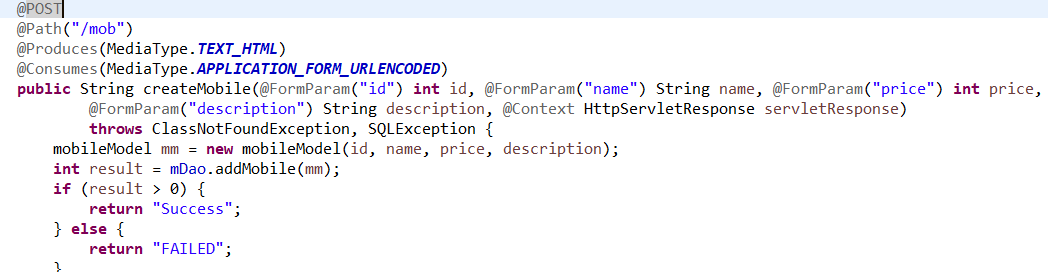
### Maintain the database

1. **Run the Database Ant File provided on moodle**
2. **Run the Server and the table will be active.**
3. **On ant file press manage to handle the database manually**
4. **Run main.java in Client to interact with the database**

# Requirement

"A Jax-Rs/Jersey client that sends all of the HTTP requests GET/PUT/POST/DELETE, parses the response XMLPullParser and outputs to the GUI".A tomcat server that responds to all of the HTTP requests.

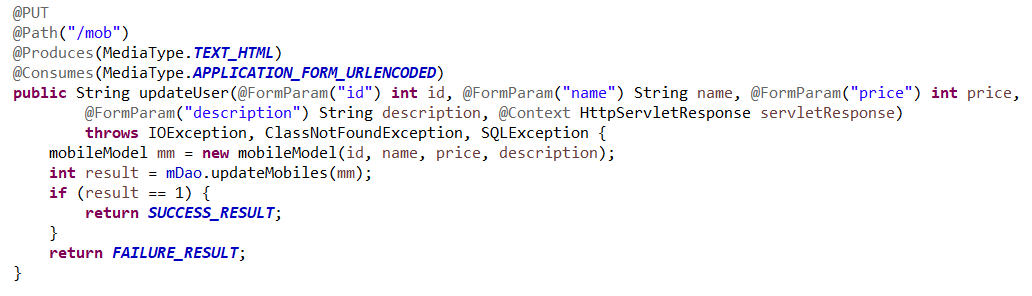
### POST:



This method accepts 4 parameters needed to insert in the database and then pass the object for POST, It return message of the success or failure of the insertion process. This method is called at the GUI with the respective parameters.

PUT**:**

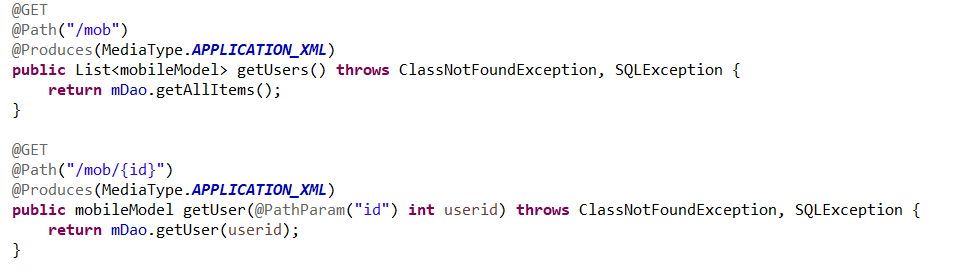
In the client application, the PUT function is written in the update the mobiles



This method accepts the parameter needed to update, in this the important parameter is ID, which is used to update the data. The ID acts as the reference for the update method and allows the changes to other respective fields.

### GET:

In the client application, the PUT function is written in the DefaultTable class, which holds two methods named,



The getUser method returns all the data in the table as a TableModel which is used to populate the JTable in the GUI.

# Requirements

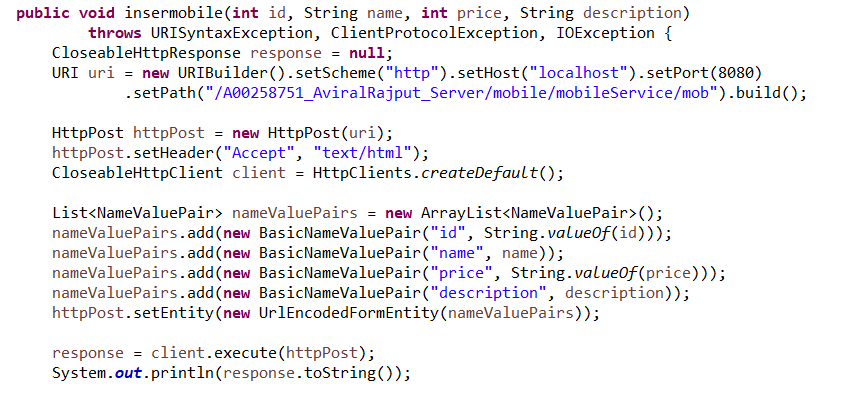
A GUI that handles GET/PUT/POST/DELETE functions

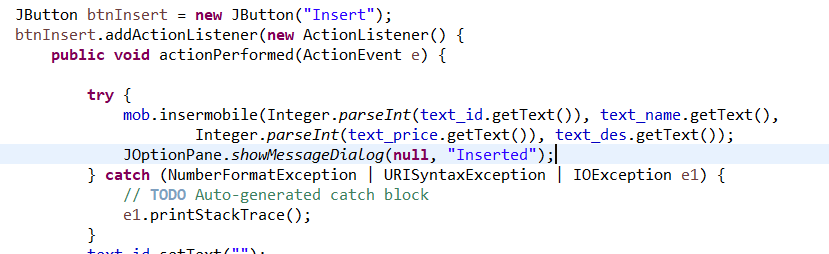
## Completion

UI screenshots have been attached above, this code runs at the back of the button accept the parameter values from UI.

### POST

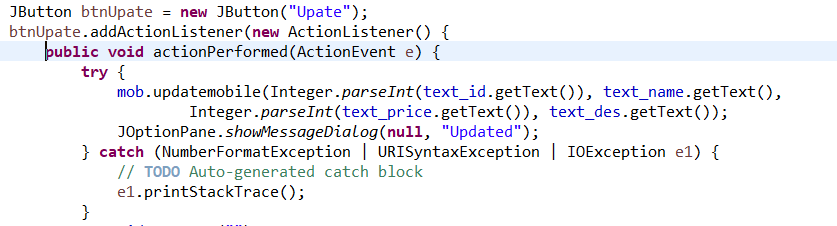
InsertMobile code interacts with the server to post the data to the server coming from textbox as input

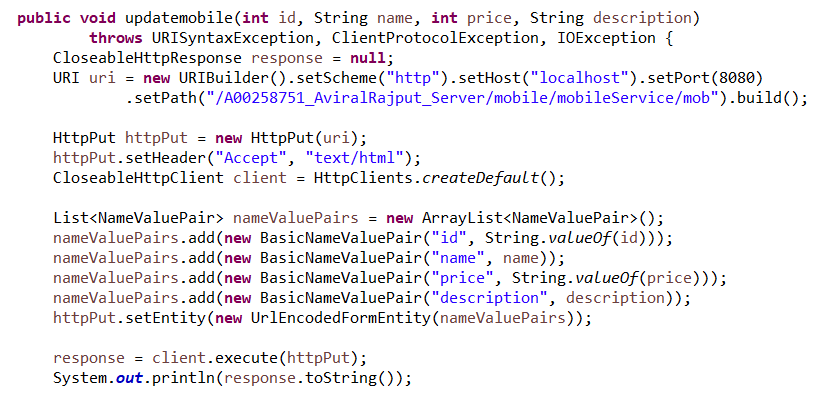




### PUT

UpdateMobiles take the relevant id and updates the associated field based upon input from user.





### Delete

Delete function on client side to delete the data based upon id

