*Term Project*

*Instructor: Faisal Khan*

|  |  |  |
| --- | --- | --- |
| *Name* | *Student ID* | *Group#* |
| *Jiju Acharya* | *C0770937* | *1* |
| *Filgy Sali* | *C0765127* | *1* |
| *Edwin Tom* | *C0760831* | *1* |
| *Ebin M. Francis* | *C0765035* | *1* |

1-Use Documentation Comments where needed and use Javadoc to create HTML files for documentation.

2-In your project src folder add this word document with all your group members names and paste the screen shots of your output after each question.

3-Right Click on your Project Folder in Eclipse, Copy Option is available. Create a copy of the Project and upload the zipped folder using Moodle.

Create a web based application using Either JSF or JSPs & Servlets . Your web application should maintain

Bank Account Records. You can use a model class BankAccount

with fields like.

Account ID, Account Type(Checking/Saving), AccouHolderName,

AccountHolderContact, AccountBalance,AccountSetupDate.

You should be able to perform all these CRUD Operations.

a)Create a Database BankAccounts for your application.

Create a table named Accounts in this database.

Insert 5 to 6 Records in this table. 10

CREATE TABLE `cestar`.`bank` (

`accountId` INT NOT NULL,

`accountType` VARCHAR(45) NULL,

`accountName` VARCHAR(45) NULL,

`accountContact` VARCHAR(45) NULL,

`accountBalance` VARCHAR(45) NULL,

`accountDate` VARCHAR(45) NULL,

PRIMARY KEY (`accountId`));

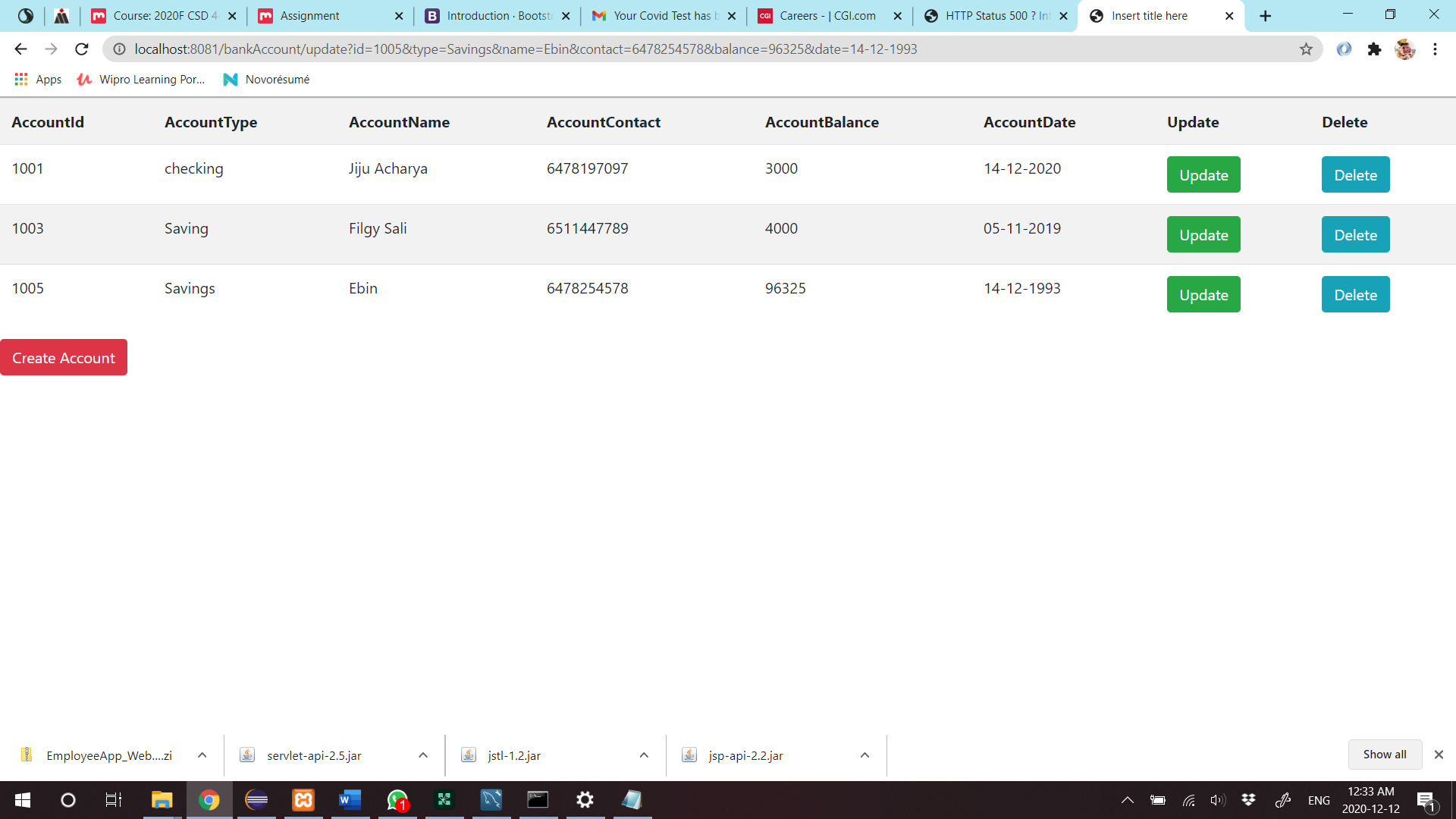
INSERT INTO `cestar`.`bank` (`accountId`, `accountType`, `accountName`, `accountContact`, `accountBalance`, `accountDate`) VALUES ('1001', 'checking', 'Jiju Acharya', '6478197097', '3000', '14-12-2020');

INSERT INTO `cestar`.`bank` (`accountId`, `accountType`, `accountName`, `accountContact`, `accountBalance`, `accountDate`) VALUES ('1002', 'Saving', 'Filgy Sali', '6511447789', '4000', '05-11-2019');

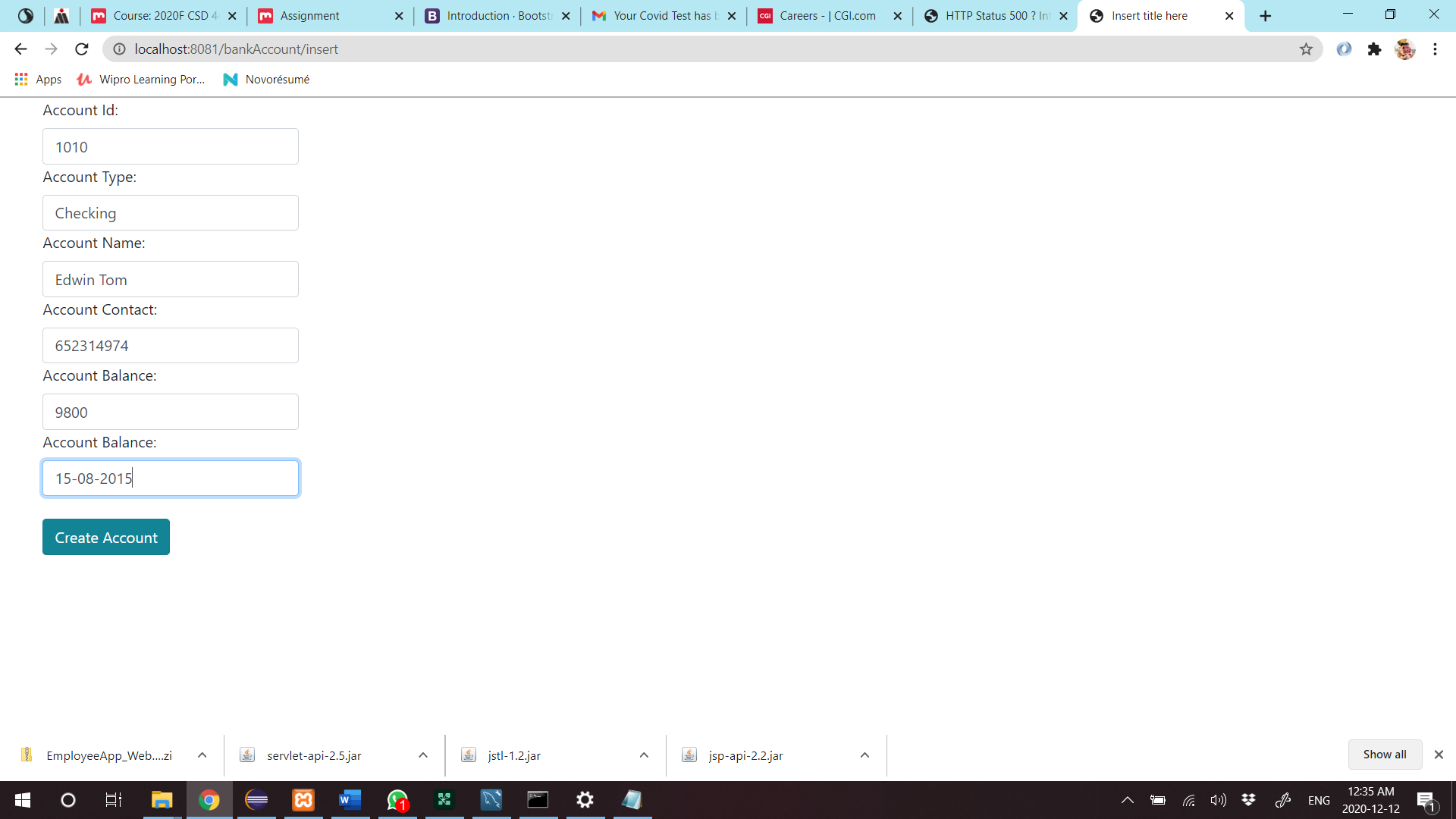
INSERT INTO `cestar`.`bank` (`accountId`, `accountType`, `accountName`, `accountContact`, `accountBalance`, `accountDate`) VALUES ('1005', 'checking', 'Ebin Francis', '6985123547', '2500', '14-07-2018');

INSERT INTO `cestar`.`bank` (`accountId`, `accountType`, `accountName`, `accountContact`, `accountBalance`, `accountDate`) VALUES ('1008', 'Saving', 'Edwin Tom', '6542147854', '6000', '25-11-2016');

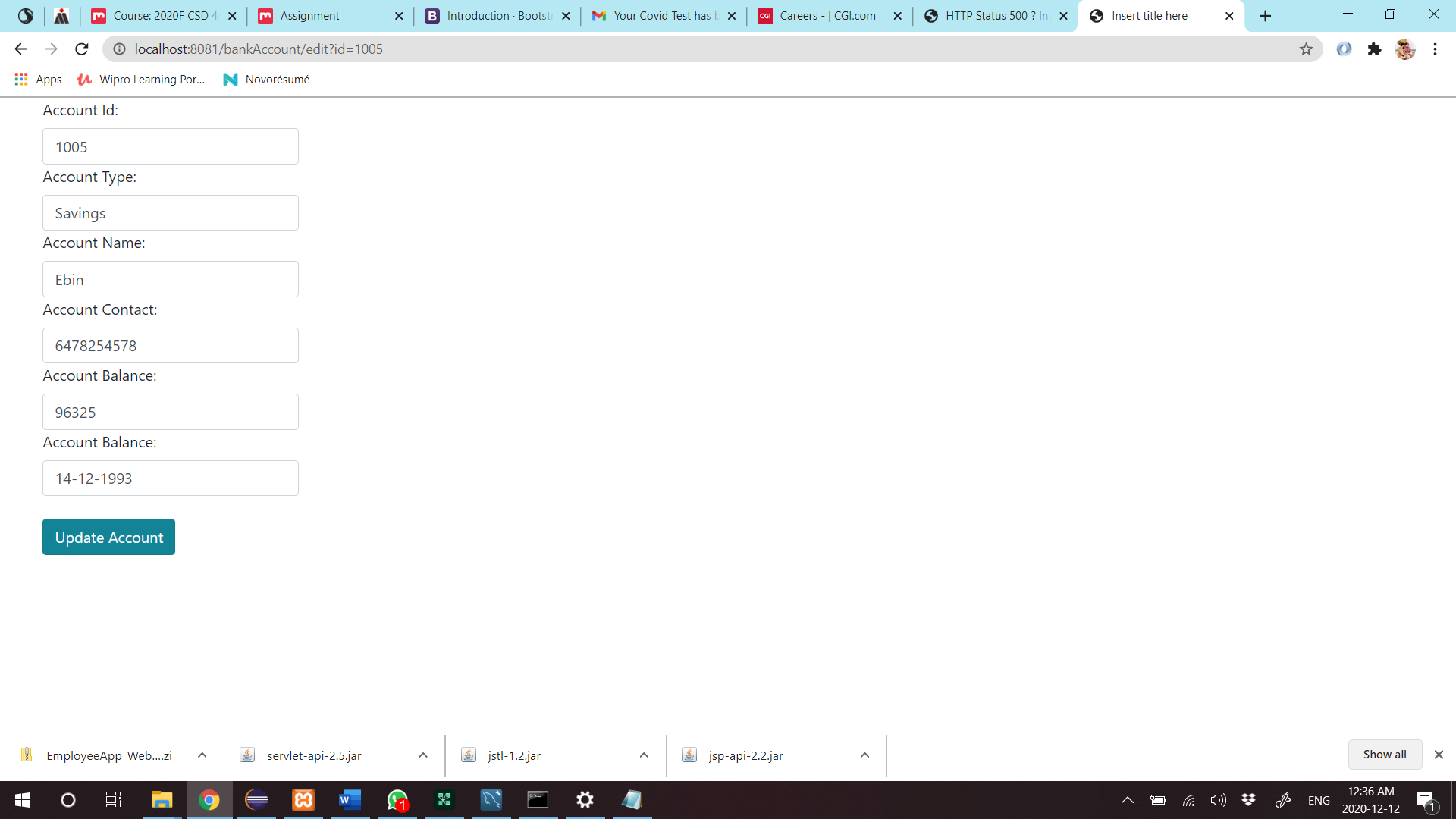
b) Display all the Bank Accounts in your xhtml page View or a JSP page as a table. 20



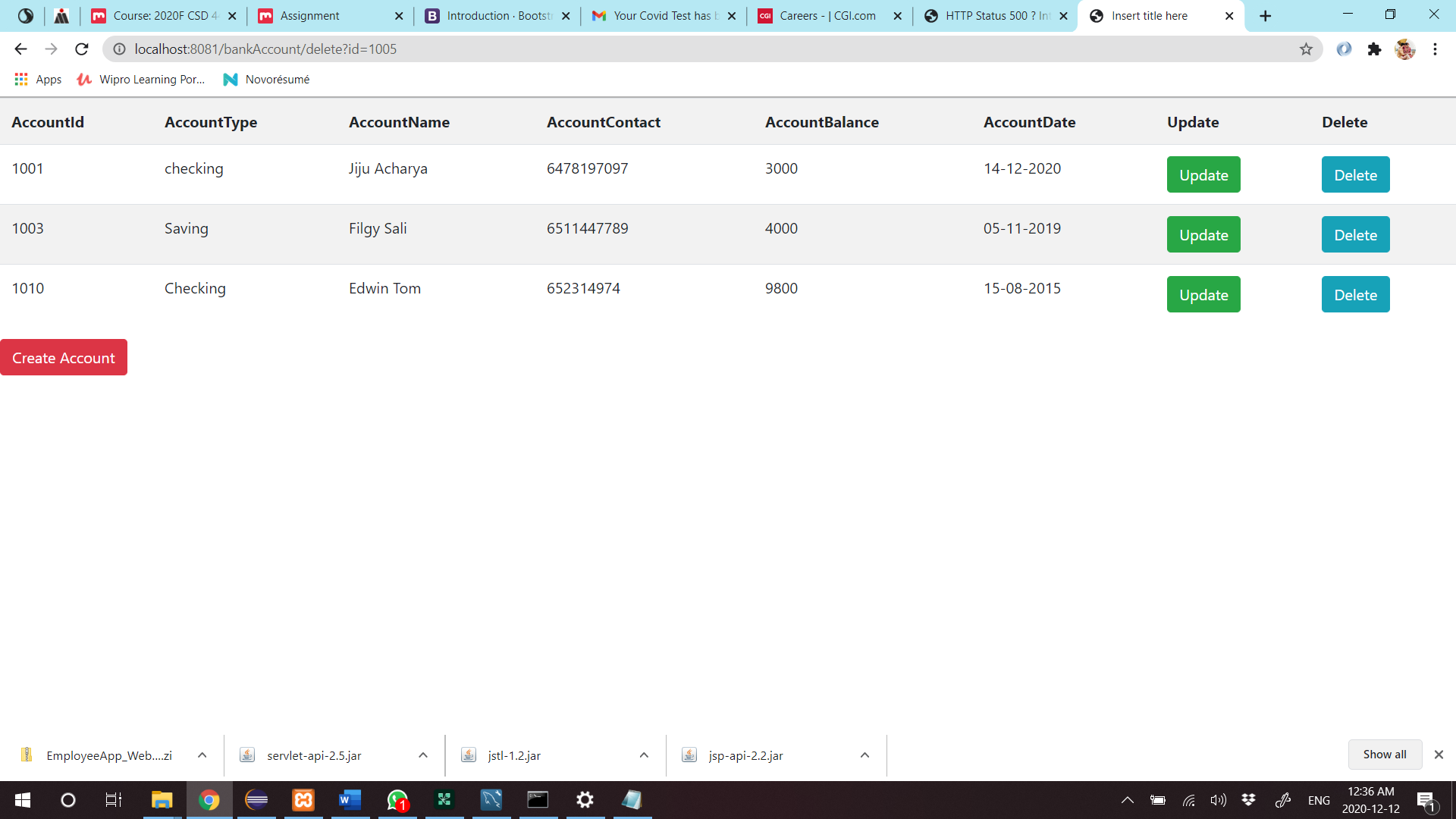
c) Add a new Bank Account in to the database using a button on your view page. 20



d)Update an existing Bank Account using an Update link on your view page. 20



e)Delete a Bank Account using a Delete Link on your view page.

 20

f) Use ManagedBean if using JSF with annotation and create separate DAO class for Crud operations or use a Simple Java Bean as your Model class if using JSPs and Servlets.Try to create separate methods for all crud operations. 10

package com.cestar.dao;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

import com.cestar.model.bank;

public class bankDao {

public Connection getConnection() {

String url = "jdbc:mysql://localhost:3306/cestar";

String user = "root";

String pwd = "";

Connection con = null;

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection(url, user, pwd);

System.out.println("Connection Successful!!!");

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return con;

}

public int addRec(bank b) {

Connection con = getConnection();

String sql = "insert into bank values (?,?,?,?,?,?)";

int status = 0;

try {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, b.getAccountId());

pstmt.setString(2, b.getAccountType());

pstmt.setString(3, b.getAccountName());

pstmt.setString(4, b.getAccountContact());

pstmt.setString(5, b.getAccountBalance());

pstmt.setString(6, b.getAccountDate());

status = pstmt.executeUpdate();

if (status > 0) {

System.out.println("Record Inserted Successfuly!");

} else {

System.out.println("Try Again!!");

}

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

return status;

}

public List<bank> displayAllRec() {

Connection con = getConnection();

String sql = "select \* from bank";

List<bank> bnk = new ArrayList<>();

try {

Statement stmt = con.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

bank b = new bank(rs.getInt("accountId"), rs.getString("accountType"), rs.getString("accountName"),

rs.getString("accountContact"), rs.getString("accountBalance"),rs.getString("accountDate"));

bnk.add(b);

}

System.out.println(bnk);

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return bnk;

}

public int updtaeRec(int old\_id,bank Updated\_bnk) {

Connection con = getConnection();

String sql = "update bank set accountId=?,accountType=?,accountName=?,accountContact=?,accountBalance=?, accountDate=? where accountId=?";

int status = 0 ;

try {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1,Updated\_bnk.getAccountId());

pstmt.setString(2,Updated\_bnk.getAccountType());

pstmt.setString(3,Updated\_bnk.getAccountName());

pstmt.setString(4,Updated\_bnk.getAccountContact());

pstmt.setString(5, Updated\_bnk.getAccountBalance());

pstmt.setString(6, Updated\_bnk.getAccountDate());

pstmt.setInt(7, old\_id);

status = pstmt.executeUpdate();

if(status>0) {

System.out.println("Record Updated:");

}

else {

System.out.println("Try Again:");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return status;

}

public void deleteRec(int id) {

Connection con = getConnection();

String sql = "delete from bank where accountId=?";

int status = 0 ;

try {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, id);

status = pstmt.executeUpdate();

if(status>0) {

System.out.println("Record deleted!");

}

else {

System.out.println("Try again!");

}

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public bank getRecById(int id) {

Connection con = getConnection();

String sql = "select \* from bank where accountId=?";

int status = 0;

bank bnk = null;

try {

PreparedStatement pstmt = con.prepareStatement(sql);

pstmt.setInt(1, id);

ResultSet rs = pstmt.executeQuery();

while(rs.next()) {

bnk = new bank(rs.getInt(1),rs.getString(2),rs.getString(3),rs.getString(4),rs.getString(5),rs.getString(6));

}

System.out.println(bnk);

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return bnk;

}

}