Brandon Sinclair

cscu9yw assignment report

Contents

[Outline of the Problem 2](#_Toc67341871)

[Description of the problem 2](#_Toc67341872)

[Solution Overview 2](#_Toc67341873)

# Outline of the Problem

## Description of the problem

The assignment was to create a contact database service that stores details of people. The person details should include the person’s full name, address (street, town/city, postcode) and telephone number with the assumption that the phone number is a unique identifier which can be used to identify a contact.

The webservice should provide methods to retrieve, add, edit and delete contact with advance features include interacting with a group of contacts such as a group of contacts from the same town.

The webservice may also have robust handling potential error cases (adding an already existing contact, deleting one which does not exist). Some credit will be available for such features.

# Solution Overview

The client is successfully able to retrieve, add, edit and delete contacts from a database using SOAP (Simple Object Access Protocol) based web service that connect to an external database.

* The client can retrieve a contact using a contact’s phone number.
* The client can add new contacts to the database
* The client can edit existing contacts if they have that contacts phone number, they can edit the street address, postcode and city. They cannot change their first name, surname or phone number.
* The client can delete contacts if they know that contacts phone number
* The client can retrieve a list of contacts from a specific city (advanced feature)

Robust error handling has also been added to the webservice, this sends appropriate error messages based on what operation is being used for example searching for a contact with no matching phone number will result in the webservice return the error message “no match found matching Contact Number: ” to the client.

Using the WSDL (Web Services Description Language) file, Axis2 and WSDL2Java tool, were used to convert the WSDL service file into java code. The java code generated created and outline for the service and client, the complex types that are created are mapped to java classes and then used within the web service as class objects.

A basic UI for the client was built using windows builder, the window class acts as the main UI which allows the client to navigate to other frames for different operations. Basic Error handling has also been added for the UI for example no entered inputs, wrong data types etc.

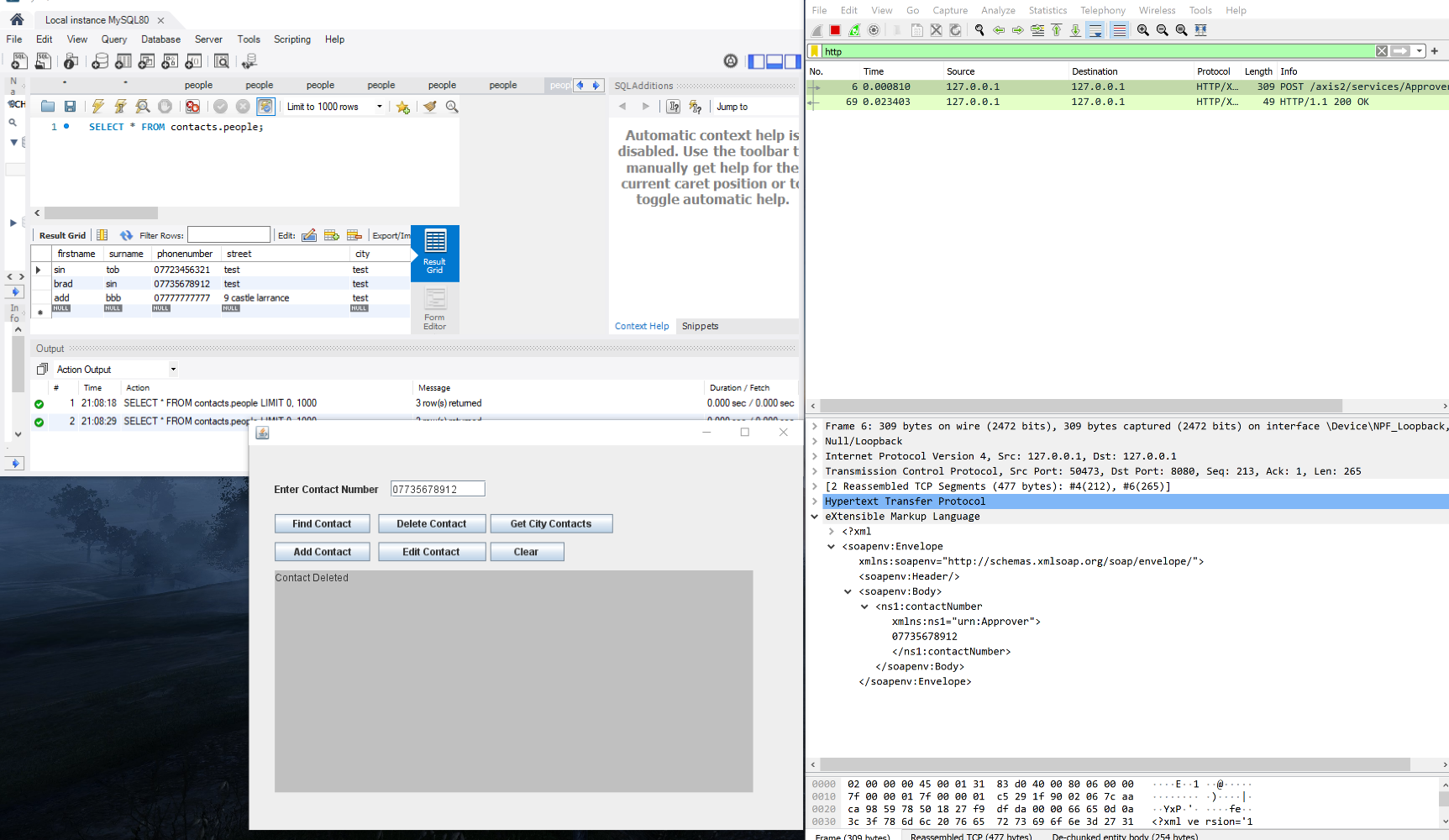
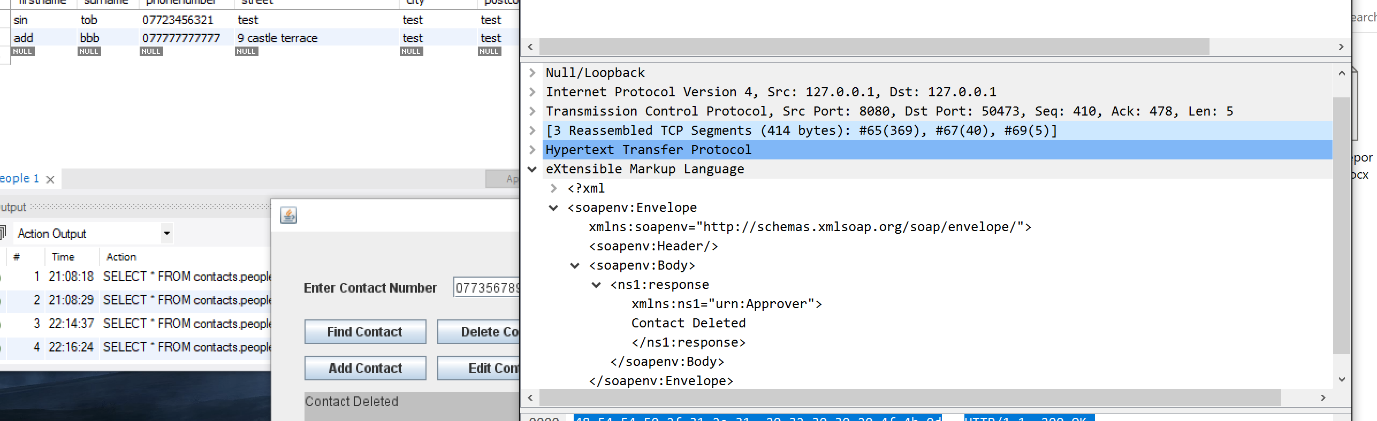
The external database was created using MySQL workbench, with a basic person table which include the fields firstname, surname, phonenumber, street, city and postcode.

# Solution with Screen Shots

## Retrieve Contact by Phone Number

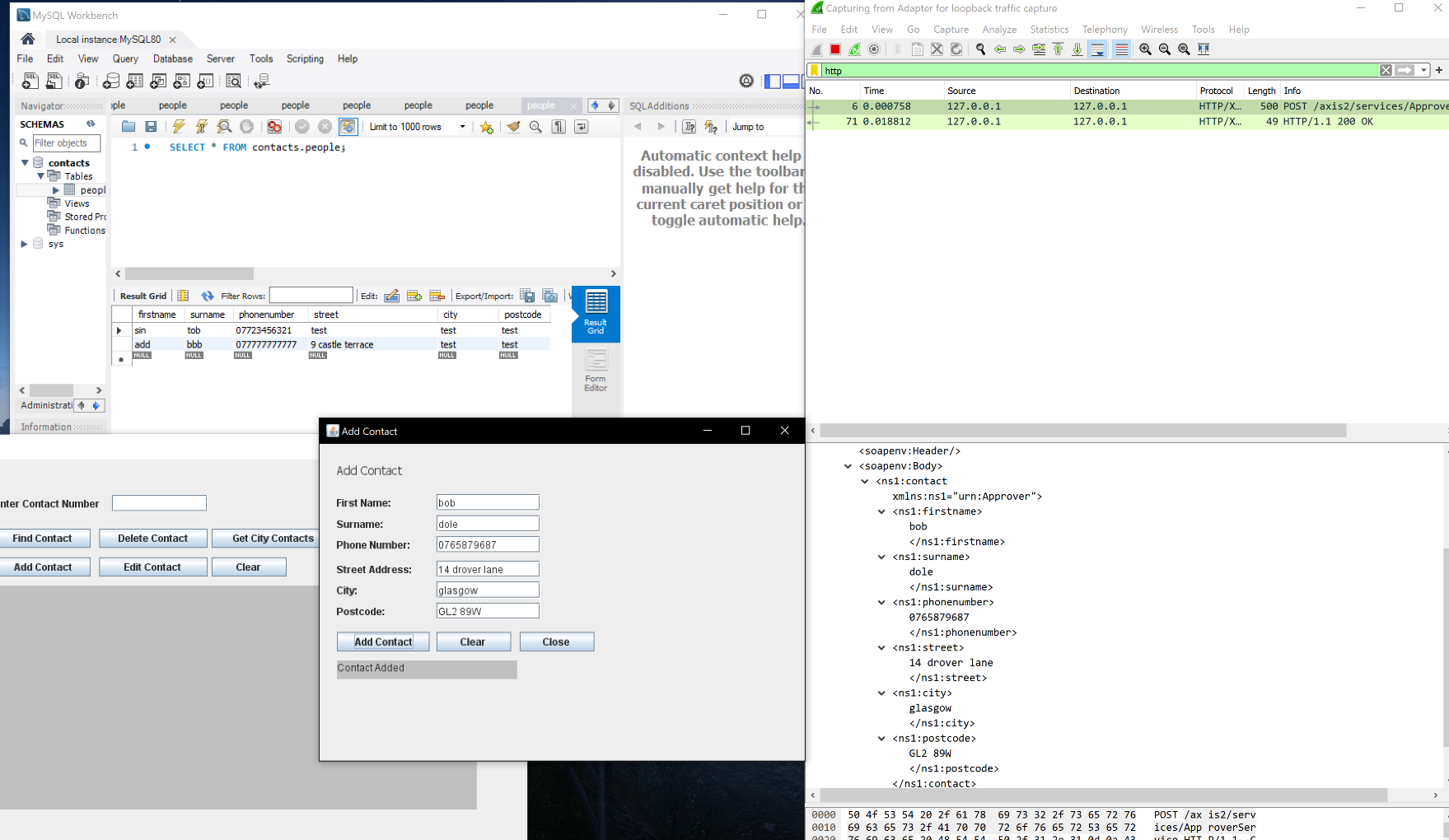
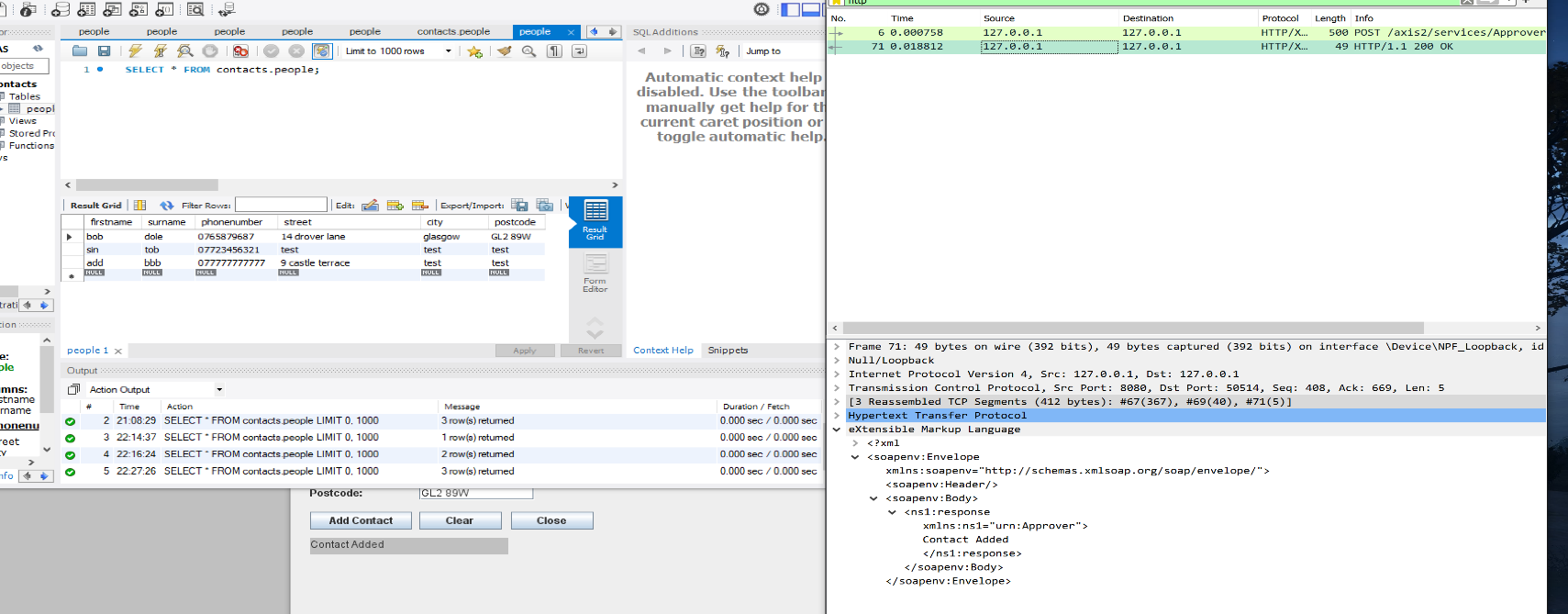
For retrieving the contact by phone operation, the client enters the contact number inside the enter contact number field, then the client will click the “find contact” button. This will then send a request to the service which we can see in the wireshark POST message. The service will then search for a matching contact in the database and sent that back to the client as a response, we can see this in the response message in wireshark.

## Delete Contact Screenshots



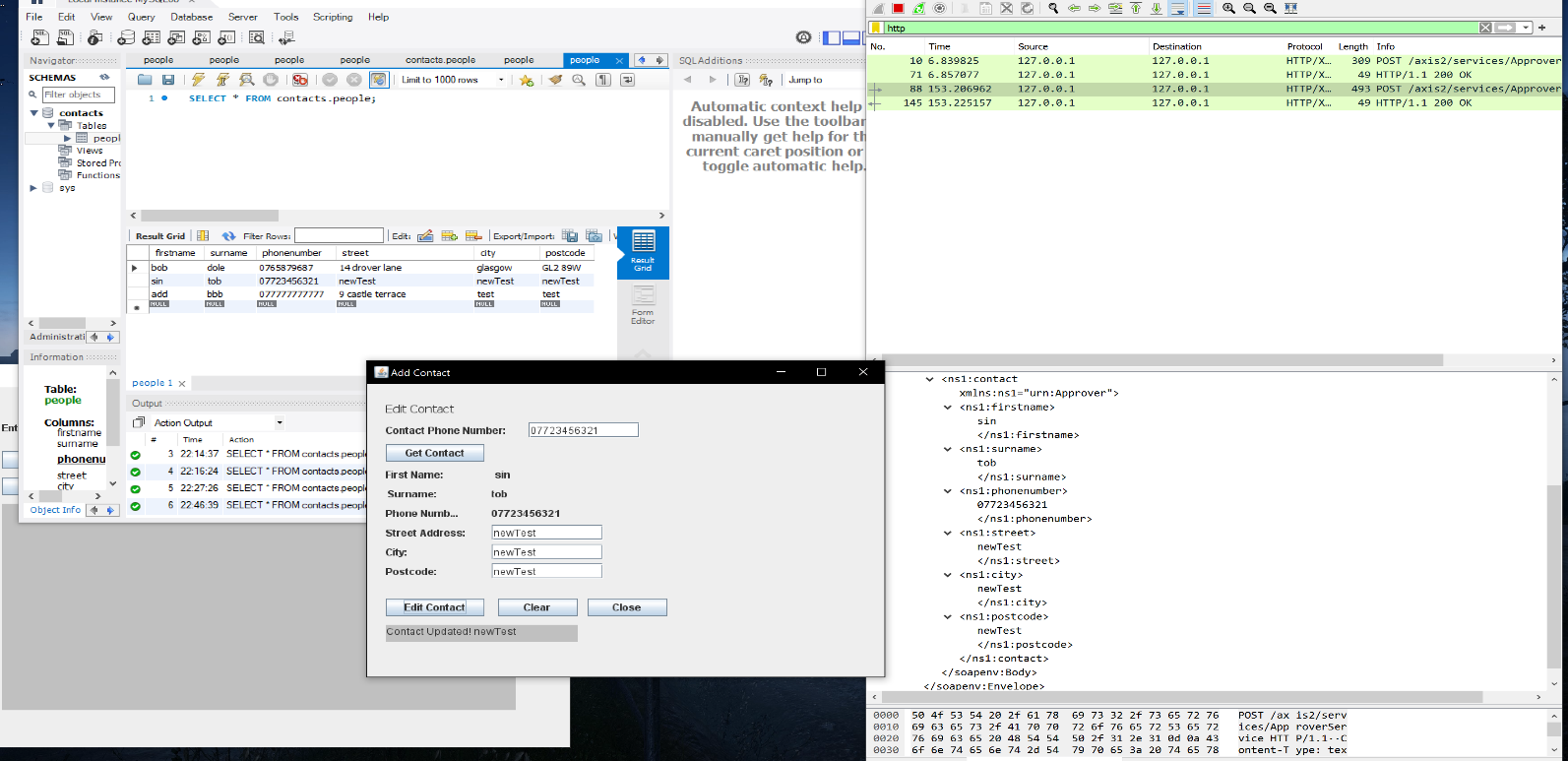
The client deletes a contact by entering that contact’s number in the contact number text field then pressing the “delete contact” button, which sends a request to delete the contact to the webservice. If the contact is deleted successfully the response will send back a response msg saying “contact deleted”

## Add Contact Screenshots



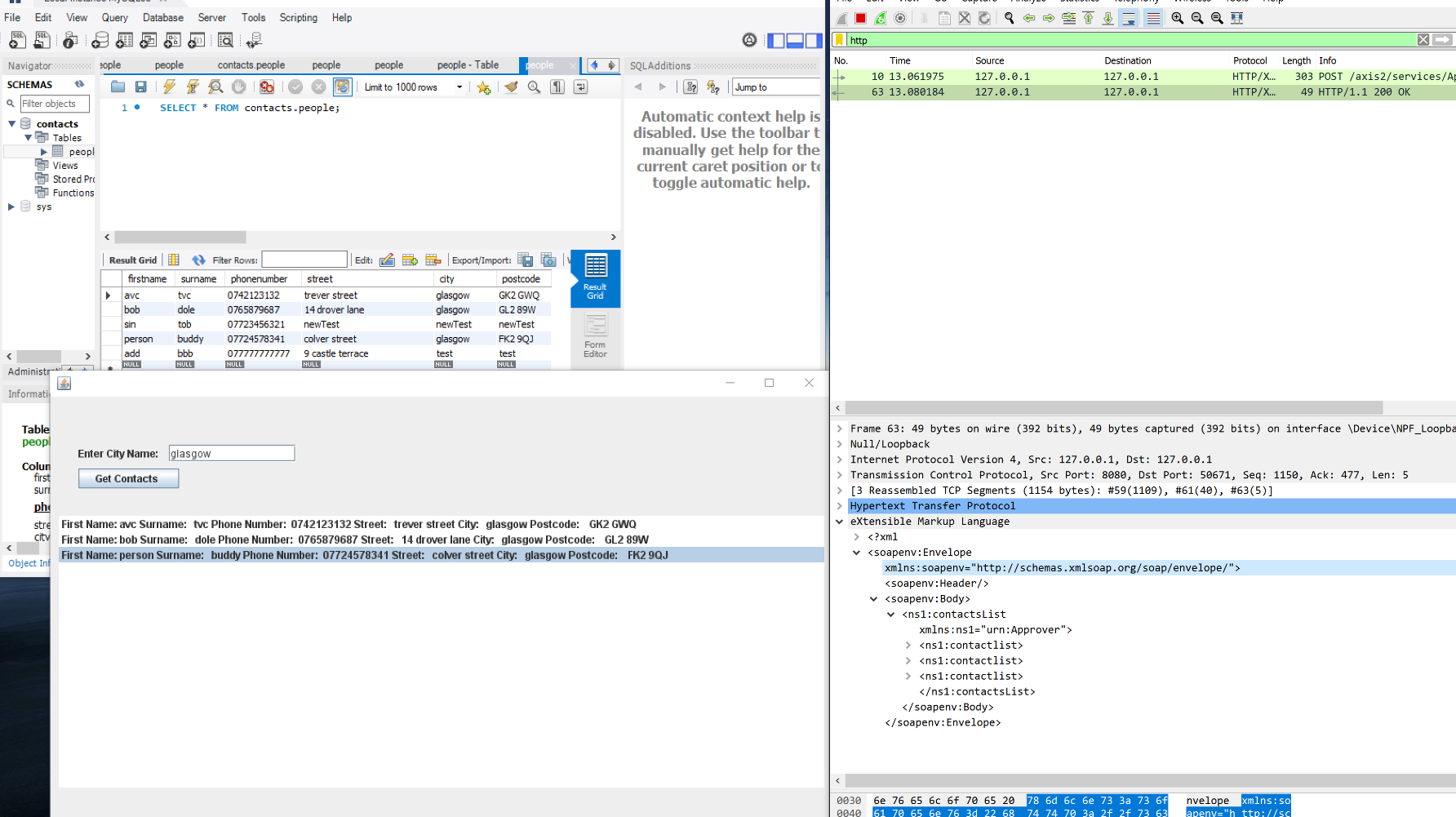
The client adds a new contact by selecting the “add contact” button from the main window, this will open a new add contact window. From this window the client can add the contact details, once the client clicks the “add contact” button which sends a request to the service which includes a person object containing the entered details, if accepted (no errors found) a response msg is sent back to the client saying “Contact Added”

## Edit Contact Screen Shots

To start the client must retrieve a contact, they do this by entering a contact number in the phone number text field and selecting the “get contact” button, this will update the rest of the fields in the window with that contact’s details. From here the client can edit the street address, city and post code.

When the client selects the “edit contact” after editing the desired fields, it sends a request with the person object to the webservice, if successful a response message will return a response message which contains the new person object details which are displayed.

## Get City Contacts Screen Shots

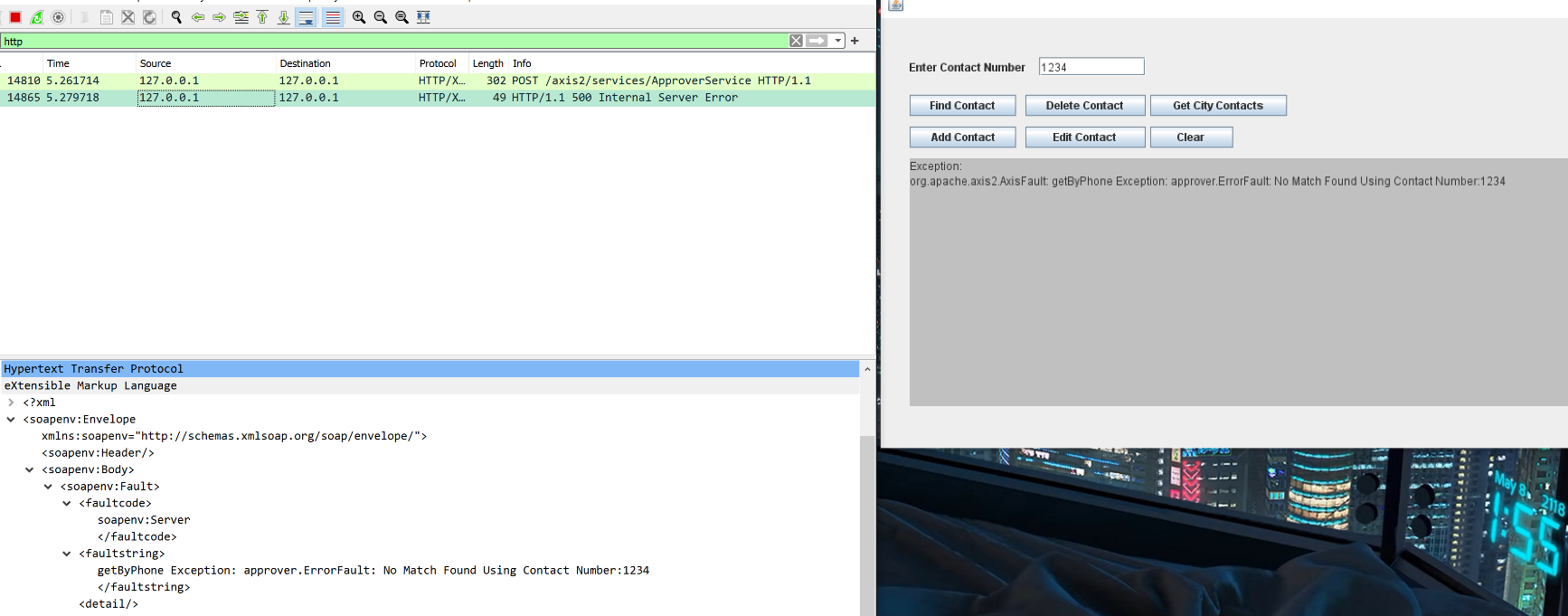


The client gets a list of contacts that are from the same city but entering that city in the city name field and selecting the “get contacts” button, this will send a request containing a city name to the webservice in response a list of contacts will be returned that match that city. Inside the contact list is the contact details.

# Error Handling Screen Shots

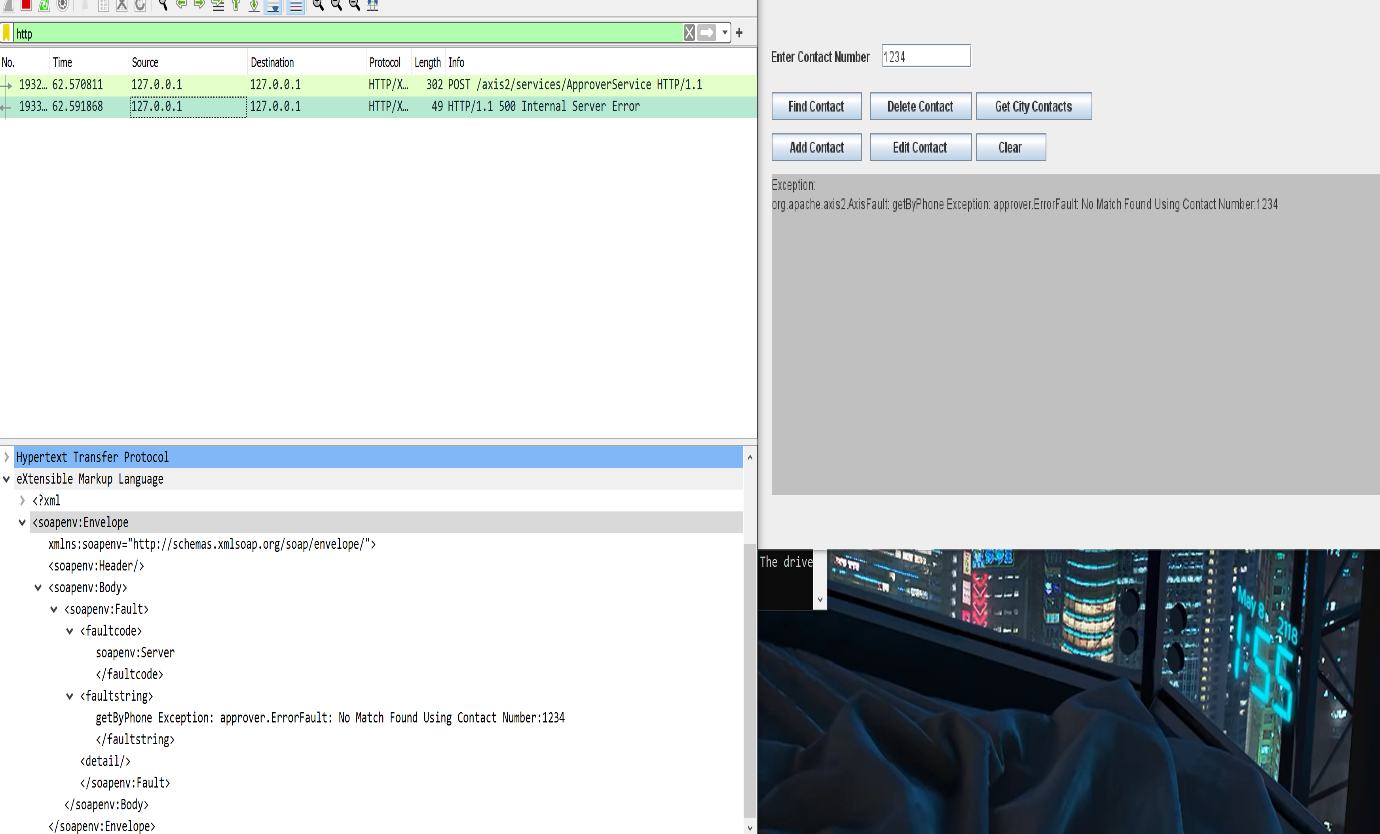
# 

## Find Contact Error Screen Shots



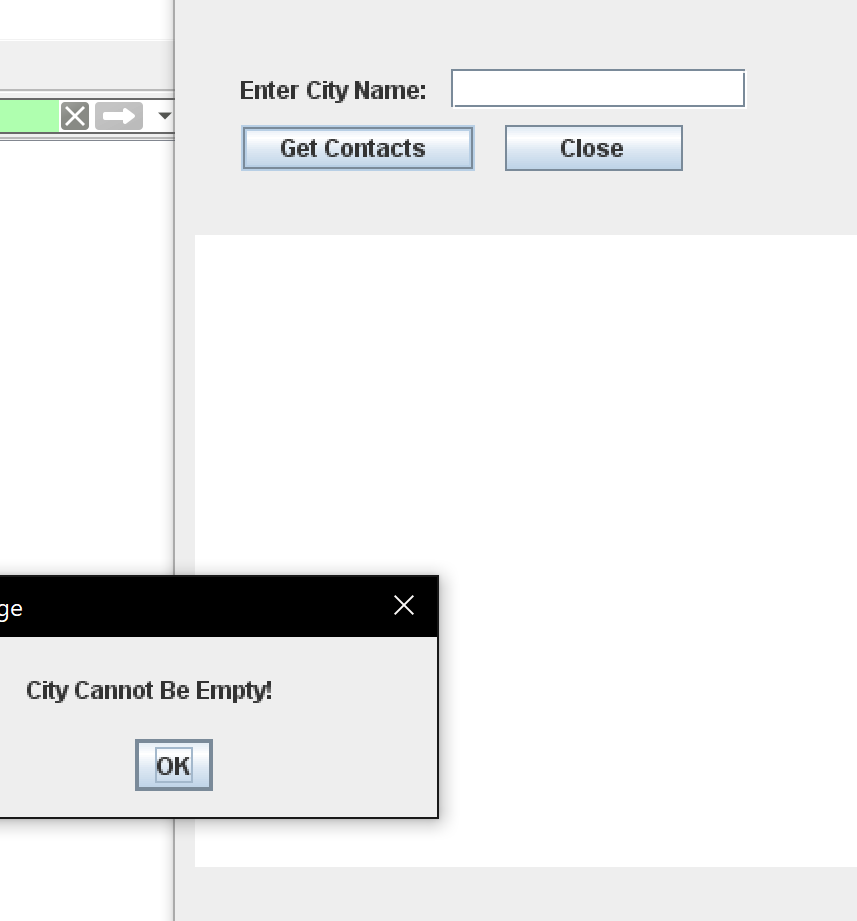
When the client requests a contact number that is not in the database this error fault message is returned and displayed.

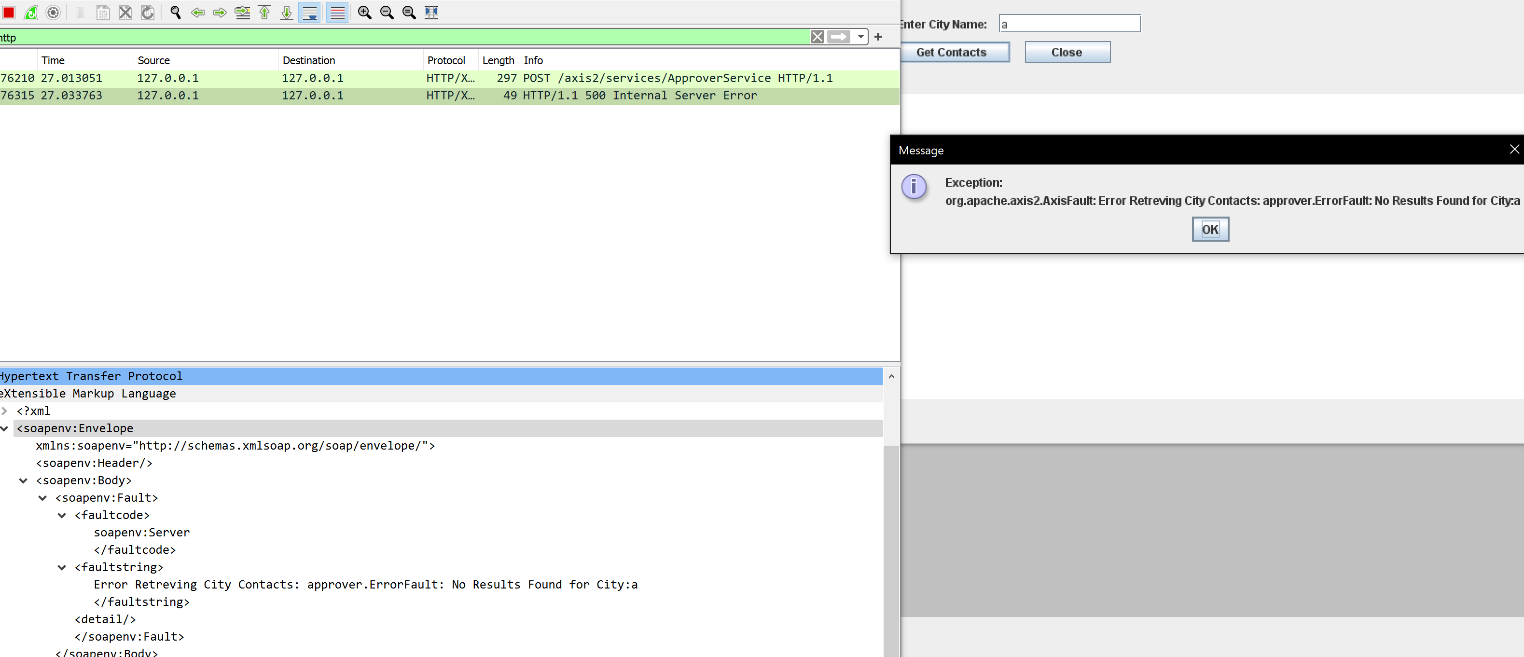
## Delete Contact Error Screen Shots



When a client tries to delete a contact that does not exist the same error message is thrown as the find contact operation. This is because before a contact is deleted the find contact operation is first run to determine if it exists. The find operation is ran locally on the server side and so does not send any requests to the client but the response can be sent from the find operation if an error fault is caught (notice the getByPhone Exception) is shown to determine the location of where the fault occurred.

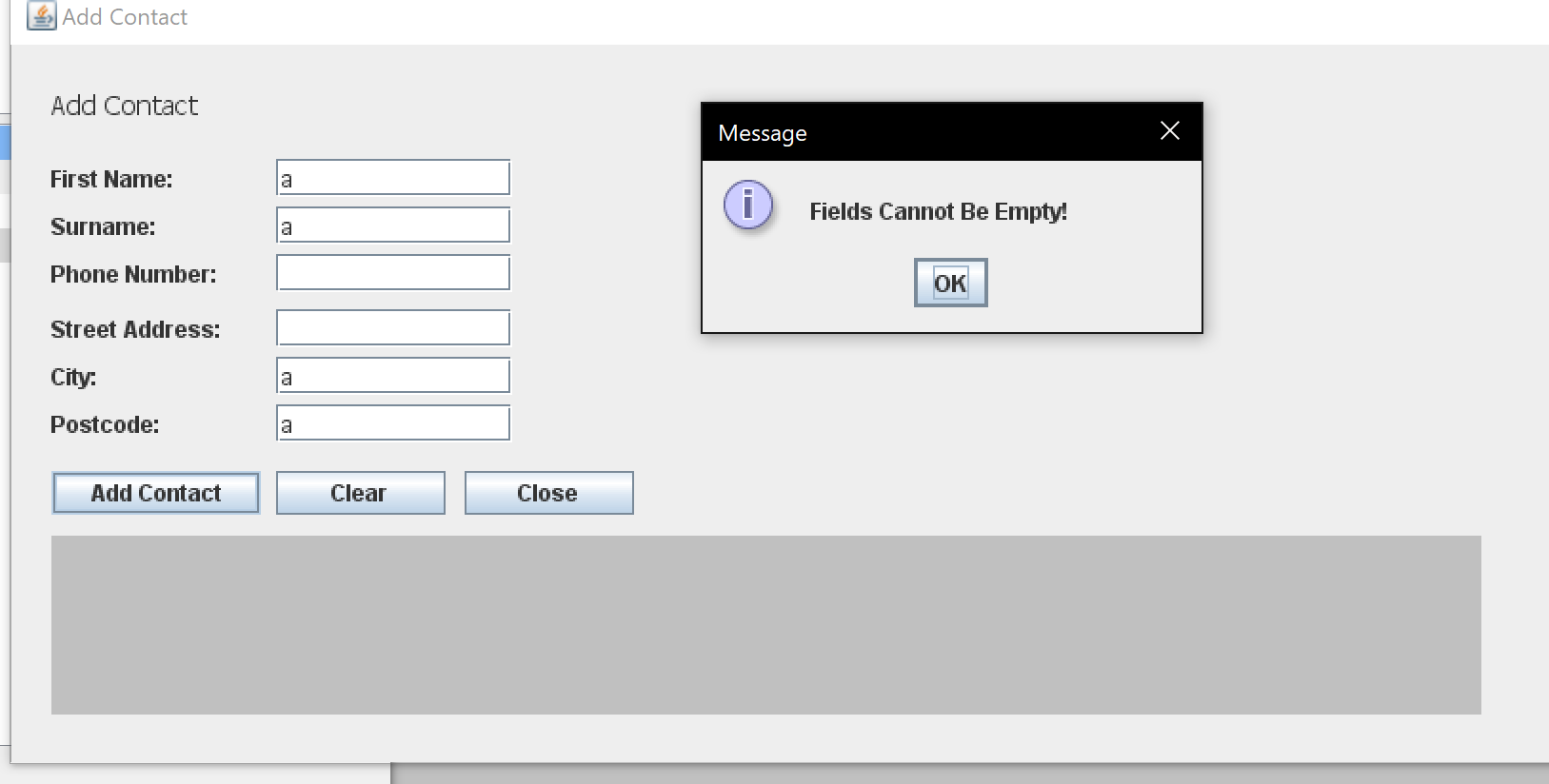
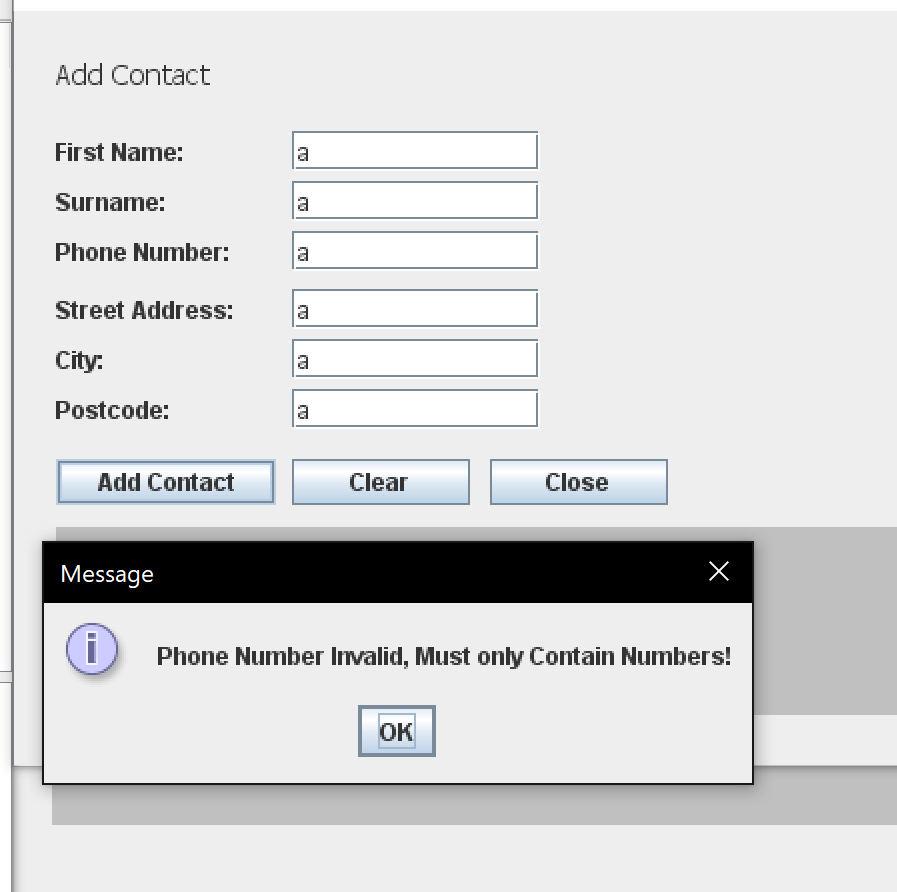
## Get City Contacts Error Screenshots

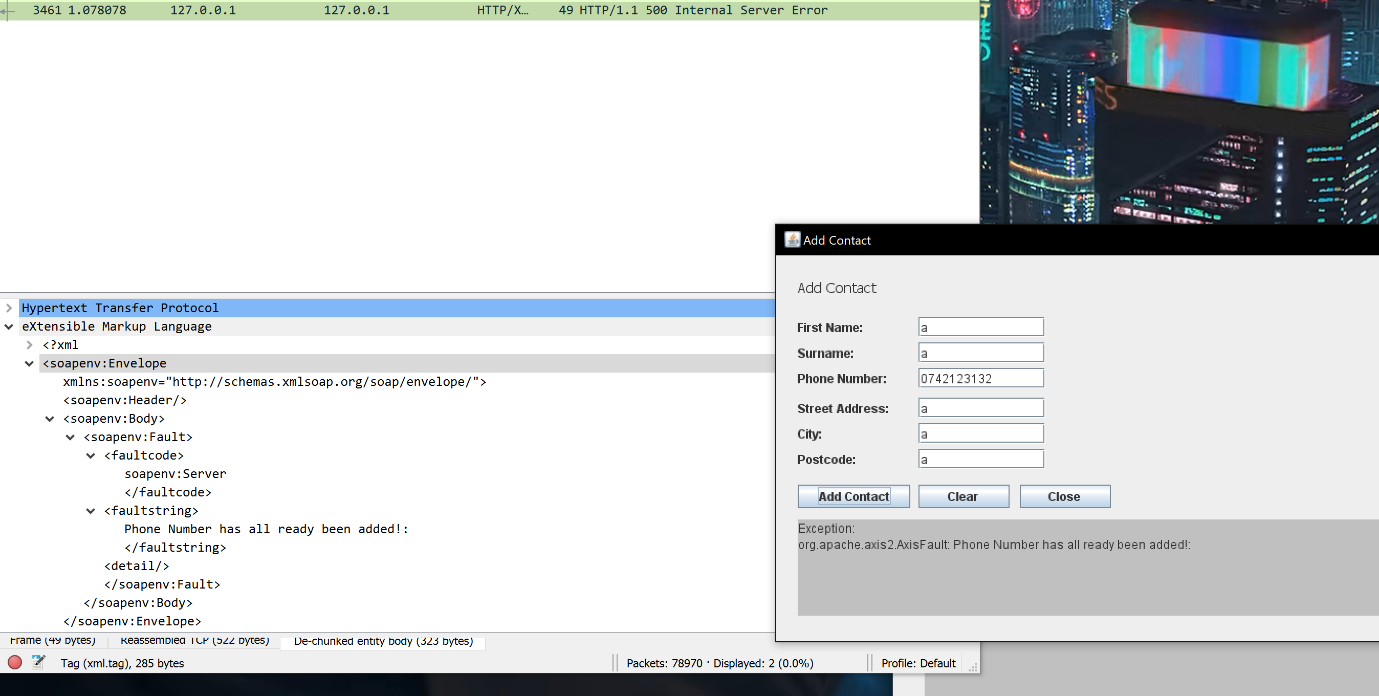




When a client searches for a list of contacts from a city that contains no contacts, the No result found for city: “ ”, is sent as a response.

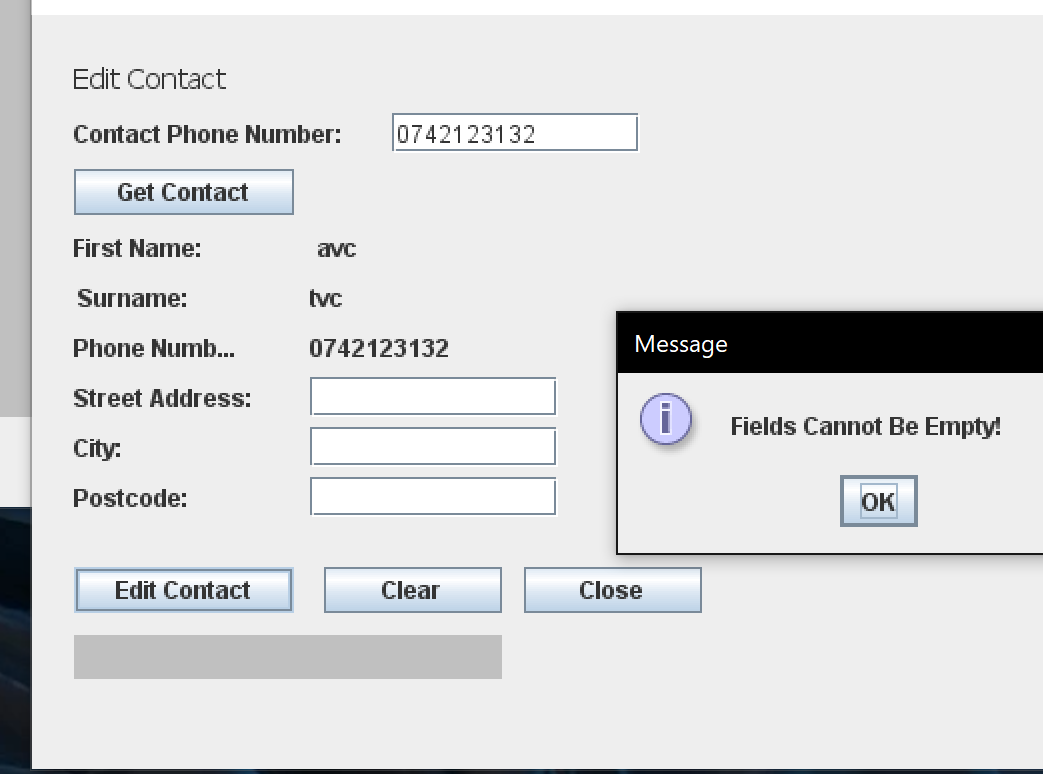
## Add Contacts Error Screenshots

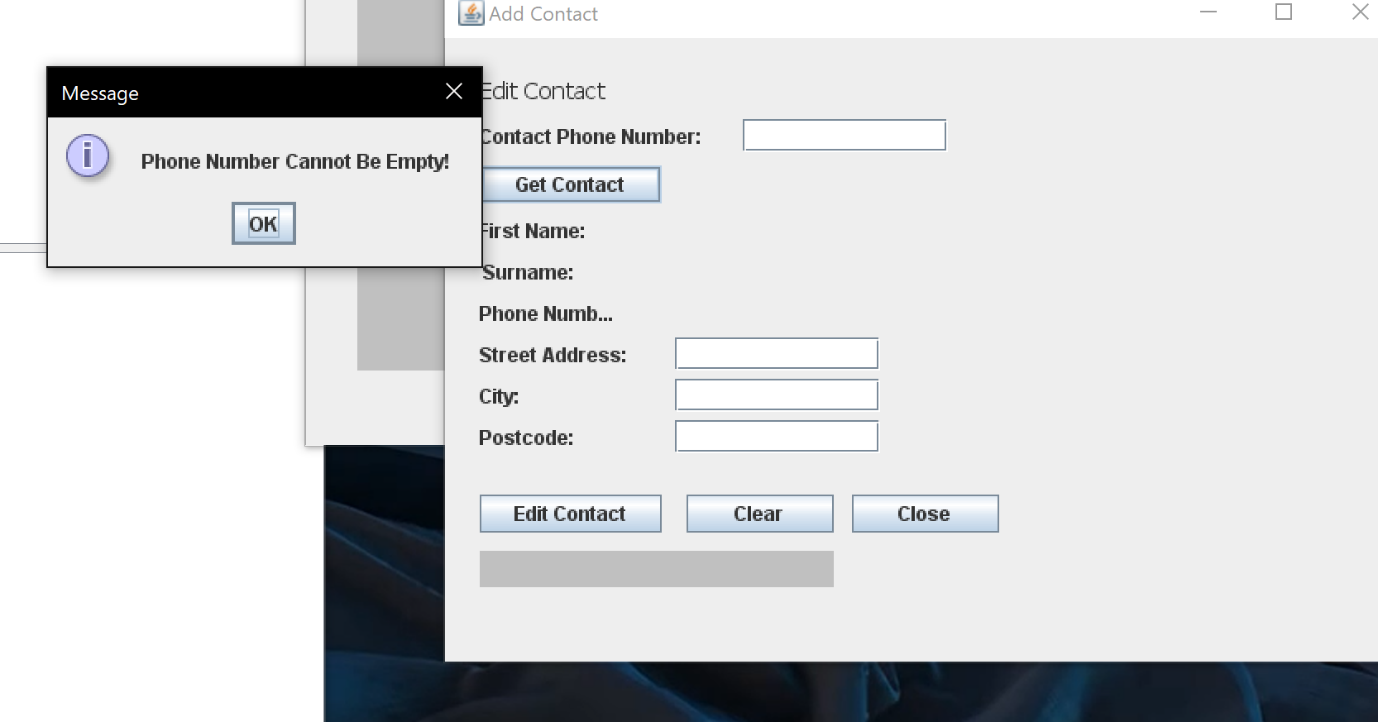




When a client tries to add a contact with a phone number that already exists in the database, the exception is thrown and is sent back to the client as a response. This exception is caught because the phone number is the primary key in the database and thus cannot be duplicated.

## Edit Contact Screen Shots





There is simple UI error prevention for this operation as no other error can occur if these errors are prevented (within reason).

# Appendix

This section includes the program’s code excluding any code that was generated from the wsdl file.

## ApproverWSDL File

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<**definitions** name=*"ApproverDefinitions"*

targetNamespace=*"urn:Approver"* xmlns=*"http://schemas.xmlsoap.org/wsdl/"*

xmlns:app=*"urn:Approver"* xmlns:xsd=*"http://www.w3.org/2001/XMLSchema"*

xmlns:soap=*"http://schemas.xmlsoap.org/wsdl/soap/"*>

<**types**>

<**schema** elementFormDefault=*"qualified"*

targetNamespace=*"urn:Approver"*

xmlns=*"http://www.w3.org/2001/XMLSchema"*>

<**complexType** name=*"personType"*>

<**sequence**>

<**element** name=*"firstname"* type=*"xsd:string"* />

<**element** name=*"surname"* type=*"xsd:string"* />

<**element** name=*"phonenumber"* type=*"xsd:string"* />

<**element** name=*"street"* type=*"xsd:string"* />

<**element** name=*"city"* type=*"xsd:string"* />

<**element** name=*"postcode"* type=*"xsd:string"* />

</**sequence**>

</**complexType**>

<**complexType** name=*"contactDetails"*>

<**sequence**>

<**element** name=*"contactlist"* type=*"app:personType"*

maxOccurs=*"unbounded"* minOccurs=*"0"* />

</**sequence**>

</**complexType**>

<**element** name=*"contact"* type=*"app:personType"* />

<**element** name=*"response"* type=*"xsd:string"* />

<**element** name=*"contactsList"* type=*"app:contactDetails"*/>

<**element** name=*"contactNumber"* type=*"xsd:string"* />

<**element** name=*"contactsCity"* type=*"xsd:string"* />

<**element** name=*"Error"* type=*"xsd:string"* />

</**schema**>

</**types**>

<**message** name=*"contactRequest"*>

<**part** name=*"request"* element=*"app:contactNumber"* />

</**message**>

<**message** name=*"contactResponse"*>

<**part** name=*"response"* element=*"app:contact"* />

</**message**>

<**message** name=*"delContactRequest"*>

<**part** name=*"request"* element=*"app:contactNumber"* />

</**message**>

<**message** name=*"delContactResponse"*>

<**part** name=*"response"* element=*"app:response"* />

</**message**>

<**message** name=*"addContactRequest"*>

<**part** name=*"request"* element=*"app:contact"* />

</**message**>

<**message** name=*"addContactResponse"*>

<**part** name=*"response"* element=*"app:response"* />

</**message**>

<**message** name=*"editContactRequest"*>

<**part** name=*"request"* element=*"app:contact"* />

</**message**>

<**message** name=*"editContactResponse"*>

<**part** name=*"response"* element=*"app:contact"* />

</**message**>

<**message** name=*"contactCityRequest"*>

<**part** name=*"request"* element=*"app:contactsCity"* />

</**message**>

<**message** name=*"contactCityResponse"*>

<**part** name=*"response"* element=*"app:contactsList"* />

</**message**>

<**message** name=*"errorFault"*>

<**part** name=*"errorMessage"* element=*"app:Error"* />

</**message**>

<**portType** name=*"loanPort"*>

<**operation** name=*"getByPhone"*>

<**input** message=*"app:contactRequest"* />

<**output** message=*"app:contactResponse"* />

<**fault** name=*"Error"* message=*"app:errorFault"* />

</**operation**>

<**operation** name=*"getCityContacts"*>

<**input** message=*"app:contactCityRequest"* />

<**output** message=*"app:contactCityResponse"* />

<**fault** name=*"Error"* message=*"app:errorFault"* />

</**operation**>

<**operation** name=*"delByPhone"*>

<**input** message=*"app:delContactRequest"* />

<**output** message=*"app:delContactResponse"* />

<**fault** name=*"Error"* message=*"app:errorFault"* />

</**operation**>

<**operation** name=*"editContact"*>

<**input** message=*"app:editContactRequest"* />

<**output** message=*"app:editContactResponse"* />

<**fault** name=*"Error"* message=*"app:errorFault"* />

</**operation**>

<**operation** name=*"addContact"*>

<**input** message=*"app:addContactRequest"* />

<**output** message=*"app:addContactResponse"* />

<**fault** name=*"Error"* message=*"app:errorFault"* />

</**operation**>

</**portType**>

<**binding** name=*"loanBinding"* type=*"app:loanPort"*>

<**soap:binding** style=*"document"*

transport=*"http://schemas.xmlsoap.org/soap/http"* />

<**operation** name=*"getByPhone"*>

<**soap:operation** soapAction=*""* />

<**input**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**input**>

<**output**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**output**>

<**fault** name=*"Error"*>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**fault**>

</**operation**>

<**operation** name=*"addContact"*>

<**soap:operation** soapAction=*""* />

<**input**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**input**>

<**output**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**output**>

<**fault** name=*"Error"*>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**fault**>

</**operation**>

<**operation** name=*"getCityContacts"*>

<**soap:operation** soapAction=*""* />

<**input**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**input**>

<**output**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**output**>

<**fault** name=*"Error"*>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**fault**>

</**operation**>

<**operation** name=*"editContact"*>

<**soap:operation** soapAction=*""* />

<**input**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**input**>

<**output**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**output**>

<**fault** name=*"Error"*>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**fault**>

</**operation**>

<**operation** name=*"delByPhone"*>

<**soap:operation** soapAction=*""* />

<**input**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**input**>

<**output**>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**output**>

<**fault** name=*"Error"*>

<**soap:body** use=*"literal"* namespace=*"urn:Approver"*

encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* />

</**fault**>

</**operation**>

</**binding**>

<**service** name=*"ApproverService"*>

<**port** name=*"ApproverLoan"* binding=*"app:loanBinding"*>

<**soap:address**

location=*"http://127.0.0.1:8080/axis2/services/ApproverService"* />

</**port**>

</**service**>

</**definitions**>

## ApproverService.WSDL File

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<**definitions** name=*"ApproverDefinitions"* targetNamespace=*"urn:Approver"* xmlns=*"http://schemas.xmlsoap.org/wsdl/"* xmlns:app=*"urn:Approver"* xmlns:xsd=*"http://www.w3.org/2001/XMLSchema"* xmlns:soap=*"http://schemas.xmlsoap.org/wsdl/soap/"*>

<**types**>

<**schema** elementFormDefault=*"qualified"* targetNamespace=*"urn:Approver"* xmlns=*"http://www.w3.org/2001/XMLSchema"*>

<**complexType** name=*"personType"*>

<**sequence**>

<**element** name=*"firstname"* type=*"xsd:string"*/>

<**element** name=*"surname"* type=*"xsd:string"*/>

<**element** name=*"phonenumber"* type=*"xsd:string"*/>

<**element** name=*"street"* type=*"xsd:string"*/>

<**element** name=*"city"* type=*"xsd:string"*/>

<**element** name=*"postcode"* type=*"xsd:string"*/>

</**sequence**>

</**complexType**>

<**complexType** name=*"contactDetails"*>

<**sequence**>

<**element** maxOccurs=*"unbounded"* minOccurs=*"0"* name=*"contactlist"* type=*"app:personType"*/>

</**sequence**>

</**complexType**>

<**element** name=*"contact"* type=*"app:personType"*/>

<**element** name=*"response"* type=*"xsd:string"*/>

<**element** name=*"contactsList"* type=*"app:contactDetails"*/>

<**element** name=*"contactNumber"* type=*"xsd:string"*/>

<**element** name=*"contactsCity"* type=*"xsd:string"*/>

<**element** name=*"Error"* type=*"xsd:string"*/>

</**schema**>

</**types**>

<**message** name=*"delContactResponse"*>

<**part** name=*"response"* element=*"app:response"*>

</**part**>

</**message**>

<**message** name=*"addContactRequest"*>

<**part** name=*"request"* element=*"app:contact"*>

</**part**>

</**message**>

<**message** name=*"editContactResponse"*>

<**part** name=*"response"* element=*"app:contact"*>

</**part**>

</**message**>

<**message** name=*"contactRequest"*>

<**part** name=*"request"* element=*"app:contactNumber"*>

</**part**>

</**message**>

<**message** name=*"editContactRequest"*>

<**part** name=*"request"* element=*"app:contact"*>

</**part**>

</**message**>

<**message** name=*"delContactRequest"*>

<**part** name=*"request"* element=*"app:contactNumber"*>

</**part**>

</**message**>

<**message** name=*"addContactResponse"*>

<**part** name=*"response"* element=*"app:response"*>

</**part**>

</**message**>

<**message** name=*"contactResponse"*>

<**part** name=*"response"* element=*"app:contact"*>

</**part**>

</**message**>

<**message** name=*"contactCityResponse"*>

<**part** name=*"response"* element=*"app:contactsList"*>

</**part**>

</**message**>

<**message** name=*"contactCityRequest"*>

<**part** name=*"request"* element=*"app:contactsCity"*>

</**part**>

</**message**>

<**message** name=*"errorFault"*>

<**part** name=*"errorMessage"* element=*"app:Error"*>

</**part**>

</**message**>

<**portType** name=*"loanPort"*>

<**operation** name=*"getByPhone"*>

<**input** message=*"app:contactRequest"*>

</**input**>

<**output** message=*"app:contactResponse"*>

</**output**>

<**fault** name=*"Error"* message=*"app:errorFault"*>

</**fault**>

</**operation**>

<**operation** name=*"getCityContacts"*>

<**input** message=*"app:contactCityRequest"*>

</**input**>

<**output** message=*"app:contactCityResponse"*>

</**output**>

<**fault** name=*"Error"* message=*"app:errorFault"*>

</**fault**>

</**operation**>

<**operation** name=*"delByPhone"*>

<**input** message=*"app:delContactRequest"*>

</**input**>

<**output** message=*"app:delContactResponse"*>

</**output**>

<**fault** name=*"Error"* message=*"app:errorFault"*>

</**fault**>

</**operation**>

<**operation** name=*"editContact"*>

<**input** message=*"app:editContactRequest"*>

</**input**>

<**output** message=*"app:editContactResponse"*>

</**output**>

<**fault** name=*"Error"* message=*"app:errorFault"*>

</**fault**>

</**operation**>

<**operation** name=*"addContact"*>

<**input** message=*"app:addContactRequest"*>

</**input**>

<**output** message=*"app:addContactResponse"*>

</**output**>

<**fault** name=*"Error"* message=*"app:errorFault"*>

</**fault**>

</**operation**>

</**portType**>

<**binding** name=*"loanBinding"* type=*"app:loanPort"*>

<**soap:binding** style=*"document"* transport=*"http://schemas.xmlsoap.org/soap/http"*/>

<**operation** name=*"getByPhone"*>

<**soap:operation** soapAction=*""*/>

<**input**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**input**>

<**output**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**output**>

<**fault** name=*"Error"*>

<**soap:body** encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"* use=*"literal"*/>

</**fault**>

</**operation**>

<**operation** name=*"addContact"*>

<**soap:operation** soapAction=*""*/>

<**input**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**input**>

<**output**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**output**>

<**fault** name=*"Error"*>

<**soap:body** encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"* use=*"literal"*/>

</**fault**>

</**operation**>

<**operation** name=*"getCityContacts"*>

<**soap:operation** soapAction=*""*/>

<**input**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**input**>

<**output**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**output**>

<**fault** name=*"Error"*>

<**soap:body** encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"* use=*"literal"*/>

</**fault**>

</**operation**>

<**operation** name=*"editContact"*>

<**soap:operation** soapAction=*""*/>

<**input**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**input**>

<**output**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**output**>

<**fault** name=*"Error"*>

<**soap:body** encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"* use=*"literal"*/>

</**fault**>

</**operation**>

<**operation** name=*"delByPhone"*>

<**soap:operation** soapAction=*""*/>

<**input**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**input**>

<**output**>

<**soap:body** use=*"literal"* encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"*/>

</**output**>

<**fault** name=*"Error"*>

<**soap:body** encodingStyle=*"http://schemas.xmlsoap.org/soap/encoding/"* namespace=*"urn:Approver"* use=*"literal"*/>

</**fault**>

</**operation**>

</**binding**>

<**service** name=*"ApproverService"*>

<**port** name=*"ApproverLoan"* binding=*"app:loanBinding"*>

<**soap:address** location=*"http://127.0.0.1:8080/axis2/services/ApproverService"*/>

</**port**>

</**service**>

</**definitions**>

## ApproverService.java File

package approver;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.SQLIntegrityConstraintViolationException;

import java.sql.Statement;

public class ApproverService extends ApproverServiceSkeleton {

/\*\*

\* connection to database

\* @return connection

\* @throws SQLException

\* @throws ErrorFault

\*/

public Connection createConnection() throws SQLException, ErrorFault {

try {

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

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/contacts", "root", "Bdsred123");

//return connection

return con;

} catch (Exception e) {

throw new ErrorFault("SQLException: " + e);

}

}

/\*\*

\* get a contact by taking a contact number

\* @return

\*/

public approver.Contact getByPhone(approver.ContactNumber key) throws ErrorFault {

try {

//query will return the contact details from a matching contact number

String query = "SELECT \* FROM contacts.people where phonenumber = " + key.getContactNumber();

Contact contact = new Contact();

PersonType prop = new PersonType();

Connection con = createConnection();

Statement state = con.createStatement();

ResultSet rs = state.executeQuery(query);

if (rs != null) {

while (rs.next()) {

prop.setFirstname(rs.getString("firstname"));

prop.setCity(rs.getString("city"));

prop.setPhonenumber(rs.getString("phonenumber"));

prop.setPostcode(rs.getString("postcode"));

prop.setSurname(rs.getString("surname"));

prop.setStreet(rs.getString("street"));

contact.setContact(prop);

con.close();

//returns contact object

return contact;

}

throw new ErrorFault("No Match Found Using Contact Number:" + key.getContactNumber());

} else

throw new ErrorFault("Error getting results, getByPhone Operation");

} catch (Exception e) {

throw new ErrorFault("getByPhone Exception: " + e);

}

}

/\*\*

\* get a list of contacts based on city

\*/

public approver.ContactsList getCityContacts(approver.ContactsCity city) throws ErrorFault {

//query will return contacts with the matching city

String query = "SELECT \* FROM contacts.people where city = '" + city.getContactsCity() + "'";

try {

Connection con = createConnection();

Statement state = con.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE, 0);

ResultSet rs = state.executeQuery(query);

// move cursor to last point

rs.last();

// get last row number

int rscount = rs.getRow();

if (rscount == 0) {

throw new ErrorFault("No Results Found for City:" + city.getContactsCity());

}

PersonType[] propList = new PersonType[rscount];

rs.beforeFirst();

int i = 0;

while (rs.next()) {

PersonType prop = new PersonType();

prop.setFirstname(rs.getString("firstname"));

prop.setCity(rs.getString("city"));

prop.setPhonenumber(rs.getString("phonenumber"));

prop.setPostcode(rs.getString("postcode"));

prop.setSurname(rs.getString("surname"));

prop.setStreet(rs.getString("street"));

propList[i] = prop;

i++;

}

ContactDetails contact = new ContactDetails();

contact.setContactlist(propList);

ContactsList contactList = new ContactsList();

contactList.setContactsList(contact);

//returns the contactList to the client

return contactList;

} catch (Exception e) {

throw new ErrorFault("Error Retreving City Contacts: " + e);

}

}

/\*\*

\* adds a contact to the database

\*/

public approver.Response addContact(approver.Contact contact) throws ErrorFault {

PersonType prop = contact.getContact();

//query to insert a new contact

String query = "INSERT into people (firstname, surname, phonenumber, street, city, postcode)"

+ "values (?, ? , ?, ? , ?, ?)";

try {

Connection con = createConnection();

PreparedStatement statement = con.prepareStatement(query);

statement.setString(1, prop.getFirstname());

statement.setString(2, prop.getSurname());

statement.setString(3, prop.getPhonenumber());

statement.setString(4, prop.getStreet());

statement.setString(5, prop.getCity());

statement.setString(6, prop.getPostcode());

statement.execute();

con.close();

Response resp = new Response();

resp.setResponse("Contact Added");

//returns the response to the client

return resp;

} catch (SQLIntegrityConstraintViolationException e) {

throw new ErrorFault("Phone Number has all ready been added!: ");

}catch(Exception f) {

throw new ErrorFault("Exception addContact: " + f);

}

}

/\*\*

\* edit a contact number from matching phone number

\*/

public approver.Contact editContact(approver.Contact contact) throws ErrorFault {

try {

PersonType prop = contact.getContact();

//query will update details to the matching contact

String query = "UPDATE people SET firstname= ?, surname= ?, phonenumber= ?, street= ?, city = ?, postcode= ? WHERE phonenumber = ?";

Connection con = createConnection();

PreparedStatement statement = con.prepareStatement(query);

statement.setString(1, prop.getFirstname());

statement.setString(2, prop.getSurname());

statement.setString(3, prop.getPhonenumber());

statement.setString(4, prop.getStreet());

statement.setString(5, prop.getCity());

statement.setString(6, prop.getPostcode());

statement.setString(7, prop.getPhonenumber());

statement.execute();

//returns contact object to the client

return contact;

} catch (Exception e) {

throw new ErrorFault("Failed updating Contact " + e);

}

}

/\*\*

\* deletes a contact based on contact number

\*/

public approver.Response delByPhone(approver.ContactNumber contact) throws ErrorFault {

try {

//will check to see if matching number exists, if not a error fault is thrown

Contact cn = getByPhone(contact);

//query will delete contact based on phone number

String query = "DELETE FROM contacts.people where phonenumber = " + contact.getContactNumber();

Connection con = createConnection();

Statement state = con.createStatement();

state.execute(query);

Response resp = new Response();

resp.setResponse("Contact Deleted");

con.close();

//returns a response message

return resp;

} catch (SQLException e) {

throw new ErrorFault("Failed Deleting Contact " + contact.getContactNumber() + e);

}

}

}

## Client

package approver;

import approver.ApproverServiceStub.Response;

import approver.ApproverServiceStub.ContactDetails;

import java.rmi.RemoteException;

import approver.ApproverServiceStub.Contact;

import approver.ApproverServiceStub.PersonType;

import approver.ApproverServiceStub.ContactNumber;

import approver.ApproverServiceStub.ContactsCity;

import approver.ApproverServiceStub.ContactsList;

public class Client {

private String result = "";

/\*\*

\* gets contact from service by sending phone number

\* @param contactNum contacts phone number

\* @return

\* @throws Exception

\*/

public PersonType getByPhone(String contactNum) throws Exception {

try {

//service stub

ApproverServiceStub stub = new ApproverServiceStub();

ContactNumber contact = new ContactNumber();

contact.setContactNumber(contactNum);

//send contact number to service, returns a response object

Contact resp = stub.getByPhone(contact);

PersonType prop = resp.getContact();

result = "FirstName: " + prop.getFirstname() + " " + "Surname:" + prop.getSurname() + " " + "Phone Number: " + prop.getPhonenumber() + " " + "Street: " + prop.getStreet() + " " + "City: "+ prop.getCity() + " " + "Postcode: " + prop.getPostcode();

//return propType

return prop;

} catch (Exception e) {

result = "Exception: " + "\n" + e;

return null;

}

}

/\*\*

\* gets a contact to edit from service

\* @param approve

\* @return returns contact

\* @throws RemoteException

\* @throws ErrorFault

\*/

public Contact editContact(Contact approve) throws RemoteException, ErrorFault {

try {

//service stub

ApproverServiceStub stub = new ApproverServiceStub();

//sends contact to service, expects a contact object returned

Contact result = stub.editContact(approve);

//return contact result

return result;

} catch (Exception e) {

result = "Exception: " + "\n" + e;

System.out.println("Exception:" + e);

return null;

}

}

/\*\*

\* gets a list of contacts with matching city value from service

\* @param cityName

\* @return return personType

\* @throws Exception

\*/

public PersonType[] getCityContacts(String cityName) throws Exception {

try {

//service stub

ApproverServiceStub stub = new ApproverServiceStub();

ContactsCity city = new ContactsCity();

city.setContactsCity(cityName);

//sends city string to service, expects a contact list returned

ContactsList contactList = stub.getCityContacts(city);

ContactDetails contactDetails = contactList.getContactsList();

PersonType[] prop = contactDetails.getContactlist();

//returns personType

return prop;

} catch (Exception e) {

System.out.println("Exception:" + e);

result = "Exception: " + "\n" + e;

return null;

}

}

/\*\*

\* sends a contact to service to be added to database

\* @param prop

\* @throws Exception

\*/

public void addContact(PersonType prop) throws Exception {

try {

//service stub

ApproverServiceStub stub = new ApproverServiceStub();

Contact approve = new Contact();

approve.setContact(prop);

//sends contact to service, expects a response back

Response resp = stub.addContact(approve);

result = resp.getResponse();

} catch (Exception e) {

result = "Exception: " + "\n" + e;

}

}

/\*\*

\* deletes a contact by sending a contactNum to service

\* @param contactNum

\* @throws Exception

\*/

public void delByPhone(String contactNum) throws Exception {

try {

//service stub

ApproverServiceStub stub = new ApproverServiceStub();

ContactNumber contact = new ContactNumber();

contact.setContactNumber(contactNum);

Response resp = new Response();

//sends contact num to service, expects response msg back

resp = stub.delByPhone(contact);

result = resp.getResponse();

} catch (Exception e) {

result = "Exception: " + "\n" + e;

}

}

/\*\*

\* return a string

\* @return

\*/

public String getResults() {

return result;

}

}

## Window (Main UI)

package approver;

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JButton;

import java.awt.BorderLayout;

import java.awt.event.ActionListener;

import java.util.Scanner;

import java.awt.event.ActionEvent;

import javax.swing.JToolBar;

import javax.swing.JTextField;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JTextArea;

import java.awt.Color;

public class Window {

private JFrame frame;

private JTextArea resultTextArea;

private JTextField phoneTextField;

Client client = new Client();

/\*\*

\* Launch the application.

\*/

public static void main(String[] args) {

Scanner in = new Scanner(System.in); // Create a Scanner object

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

Window window = new Window();

window.frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

public Window() {

initialize();

}

public void showResults() {

resultTextArea.setText(client.getResults());

}

/\*\*

\* Initialize the contents of the frame.

\*/

private void initialize() {

frame = new JFrame();

frame.setBounds(100, 100, 903, 500);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.getContentPane().setLayout(null);

JButton btnNewButton = new JButton("Find Contact");

btnNewButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (!phoneTextField.getText().isEmpty()) {

try {

client.getByPhone(phoneTextField.getText());

showResults();

} catch (Exception e1) {

e1.printStackTrace();

}

}

else {

JOptionPane.showMessageDialog(null, "Phone Number Cannot Be Empty!");

}

}

});

btnNewButton.setBounds(31, 82, 114, 23);

frame.getContentPane().add(btnNewButton);

phoneTextField = new JTextField();

phoneTextField.setBounds(170, 42, 114, 20);

frame.getContentPane().add(phoneTextField);

phoneTextField.setColumns(10);

JLabel lblNewLabel = new JLabel("Enter Contact Number");

lblNewLabel.setBounds(31, 45, 129, 14);

frame.getContentPane().add(lblNewLabel);

resultTextArea = new JTextArea();

resultTextArea.setBackground(Color.LIGHT\_GRAY);

resultTextArea.setBounds(31, 150, 829, 266);

frame.getContentPane().add(resultTextArea);

JButton btnNewButton\_1 = new JButton("Delete Contact");

btnNewButton\_1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

try {

if (!phoneTextField.getText().isEmpty()) {

client.delByPhone(phoneTextField.getText());

showResults();

}else {

JOptionPane.showMessageDialog(null, "Phone Number Cannot Be Empty!");

}

} catch (Exception e1) {

e1.printStackTrace();

}

}

});

btnNewButton\_1.setBounds(155, 82, 129, 23);

frame.getContentPane().add(btnNewButton\_1);

JButton btnNewButton\_2 = new JButton("Clear");

btnNewButton\_2.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

resultTextArea.setText("");

phoneTextField.setText("");

}

});

btnNewButton\_2.setBounds(289, 116, 89, 23);

frame.getContentPane().add(btnNewButton\_2);

JButton btnNewButton\_3 = new JButton("Add Contact");

btnNewButton\_3.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

addContactWindow addWindow = new addContactWindow();

addWindow.main();

}

});

btnNewButton\_3.setBounds(31, 116, 114, 23);

frame.getContentPane().add(btnNewButton\_3);

JButton btnNewButton\_4 = new JButton("Edit Contact");

btnNewButton\_4.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

editContactWindow editWindow = new editContactWindow();

editWindow.main();

}

});

btnNewButton\_4.setBounds(155, 116, 129, 23);

frame.getContentPane().add(btnNewButton\_4);

JButton btnNewButton\_5 = new JButton("Get City Contacts");

btnNewButton\_5.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

cityWindow cityWindow = new cityWindow();

cityWindow.main();

}

});

btnNewButton\_5.setBounds(289, 82, 147, 23);

frame.getContentPane().add(btnNewButton\_5);

}

}

## AddContactWindow

package approver;

import java.awt.EventQueue;

import approver.ApproverServiceStub.PersonType;

import javax.swing.JFrame;

import javax.swing.JButton;

import java.awt.event.ActionListener;

import java.awt.event.WindowEvent;

import java.util.Scanner;

import java.awt.event.ActionEvent;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import java.awt.Font;

import javax.swing.JTextField;

import javax.swing.JTextArea;

import java.awt.Color;

public class addContactWindow {

private JFrame frmAddContact;

private JTextField firstNameTextField;

private JTextField surnametextField;

private JTextField phoneNumbertextField;

private JTextField streetAddresstextField;

private JTextField citytextField;

private JTextField postcodetextField;

private JTextArea resultTextArea;

private Client client = new Client();

/\*\*

\* Launch the application.

\*/

public static void main() {

EventQueue.*invokeLater*(new Runnable() {

public void run() {

try {

addContactWindow window = new addContactWindow();

window.frmAddContact.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

public addContactWindow() {

initialize();

}

public void sendContact() {

PersonType prop = new PersonType();

prop.setCity(citytextField.getText());

prop.setFirstname(firstNameTextField.getText());

prop.setPhonenumber(phoneNumbertextField.getText());

prop.setStreet(streetAddresstextField.getText());

prop.setSurname(surnametextField.getText());

prop.setPostcode(postcodetextField.getText());

try {

client.addContact(prop);

resultTextArea.setText(client.getResults());

} catch (Exception e) {

e.printStackTrace();

}

}

/\*\*

\* Initialize the contents of the frame.

\*/

private void initialize() {

frmAddContact = new JFrame();

frmAddContact.setTitle("Add Contact");

frmAddContact.setBounds(100, 100, 990, 416);

frmAddContact.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

frmAddContact.getContentPane().setLayout(null);

JLabel lblNewLabel = new JLabel("Add Contact");

lblNewLabel.setFont(new Font("Tahoma", Font.***PLAIN***, 14));

lblNewLabel.setBounds(21, 11, 93, 41);

frmAddContact.getContentPane().add(lblNewLabel);

JLabel lblNewLabel\_1 = new JLabel("First Name: ");

lblNewLabel\_1.setBounds(21, 63, 93, 14);

frmAddContact.getContentPane().add(lblNewLabel\_1);

JLabel lblNewLabel\_2 = new JLabel("Surname:");

lblNewLabel\_2.setBounds(21, 88, 65, 14);

frmAddContact.getContentPane().add(lblNewLabel\_2);

JLabel lblNewLabel\_3 = new JLabel("Phone Number:");

lblNewLabel\_3.setBounds(21, 113, 93, 14);

frmAddContact.getContentPane().add(lblNewLabel\_3);

JLabel lblNewLabel\_4 = new JLabel("Street Address:");

lblNewLabel\_4.setBounds(21, 142, 110, 14);

frmAddContact.getContentPane().add(lblNewLabel\_4);

JLabel lblNewLabel\_5 = new JLabel("City:");

lblNewLabel\_5.setBounds(21, 167, 48, 14);

frmAddContact.getContentPane().add(lblNewLabel\_5);

JLabel lblNewLabel\_6 = new JLabel("Postcode:");

lblNewLabel\_6.setBounds(21, 192, 93, 14);

frmAddContact.getContentPane().add(lblNewLabel\_6);

firstNameTextField = new JTextField();

firstNameTextField.setBounds(139, 60, 124, 20);

frmAddContact.getContentPane().add(firstNameTextField);

firstNameTextField.setColumns(10);

surnametextField = new JTextField();

surnametextField.setBounds(139, 85, 124, 20);

frmAddContact.getContentPane().add(surnametextField);

surnametextField.setColumns(10);

phoneNumbertextField = new JTextField();

phoneNumbertextField.setBounds(139, 110, 124, 20);

frmAddContact.getContentPane().add(phoneNumbertextField);

phoneNumbertextField.setColumns(10);

streetAddresstextField = new JTextField();

streetAddresstextField.setBounds(139, 139, 124, 20);

frmAddContact.getContentPane().add(streetAddresstextField);

streetAddresstextField.setColumns(10);

citytextField = new JTextField();

citytextField.setBounds(139, 164, 124, 20);

frmAddContact.getContentPane().add(citytextField);

citytextField.setColumns(10);

postcodetextField = new JTextField();

postcodetextField.setText("");

postcodetextField.setBounds(139, 189, 124, 20);

frmAddContact.getContentPane().add(postcodetextField);

postcodetextField.setColumns(10);

JButton btnNewButton = new JButton("Add Contact");

btnNewButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (firstNameTextField.getText().isEmpty() && citytextField.getText().isEmpty()

&& phoneNumbertextField.getText().isEmpty() && streetAddresstextField.getText().isEmpty()

&& surnametextField.getText().isEmpty() && postcodetextField.getText().isEmpty()) {

JOptionPane.*showMessageDialog*(null, "Fields Cannot Be Empty!");

} else if (!phoneNumbertextField.getText().matches("[0-9]+")) {

JOptionPane.*showMessageDialog*(null, "Phone Number Invalid, Must only Contain Numbers!");

} else

sendContact();

}

});

btnNewButton.setBounds(21, 224, 110, 23);

frmAddContact.getContentPane().add(btnNewButton);

resultTextArea = new JTextArea();

resultTextArea.setBackground(Color.***LIGHT\_GRAY***);

resultTextArea.setBounds(21, 258, 751, 94);

frmAddContact.getContentPane().add(resultTextArea);

JButton btnNewButton\_1 = new JButton("Clear");

btnNewButton\_1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

firstNameTextField.setText("");

surnametextField.setText("");

phoneNumbertextField.setText("");

streetAddresstextField.setText("");

citytextField.setText("");

postcodetextField.setText("");

resultTextArea.setText("");

}

});

btnNewButton\_1.setBounds(139, 224, 89, 23);

frmAddContact.getContentPane().add(btnNewButton\_1);

JButton btnNewButton\_2 = new JButton("Close");

btnNewButton\_2.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

frmAddContact.setVisible(false);

frmAddContact.dispose();

}

});

btnNewButton\_2.setBounds(238, 224, 89, 23);

frmAddContact.getContentPane().add(btnNewButton\_2);

}

}

## CityWindow

package approver;

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JTextField;

import approver.ApproverServiceStub.PersonType;

import javax.swing.DefaultListModel;

import javax.swing.JButton;

import javax.swing.JTextArea;

import java.awt.event.ActionListener;

import java.util.Scanner;

import java.awt.event.ActionEvent;

import java.awt.List;

import javax.swing.JList;

import javax.swing.JOptionPane;

import javax.swing.border.BevelBorder;

import javax.swing.border.LineBorder;

import java.awt.Color;

public class cityWindow {

private JFrame frame;

private JTextField cityTextField;

private JButton btnNewButton;

Client client = new Client();

private JList<String> list;

DefaultListModel<String> listModel = new DefaultListModel<>();

/\*\*

\* Launch the application.

\*/

public static void main() {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

cityWindow window = new cityWindow();

window.frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

public cityWindow() {

initialize();

}

public void getCityContacts() {

try {

PersonType[] prop = client.getCityContacts(cityTextField.getText());

if (prop == null) {

JOptionPane.showMessageDialog(null, client.getResults());

} else {

String[] result = new String[prop.length];

// loops through a list of person's adding each to a result array to be

// displayed

for (int i = 0; i < prop.length; i++) {

PersonType pr = prop[i];

String msg = " First Name: " + pr.getFirstname() + " " + "Surname: " + pr.getSurname()

+ " " + "Phone Number: " + " " + pr.getPhonenumber() + " "

+ "Street: " + pr.getStreet() + " " + "City: " + pr.getCity() + " "

+ "Postcode: " + " " + pr.getPostcode() + "\n";

result[i] = msg;

list.setListData(result);

}

}

} catch (Exception e) {

System.out.println("Exception:" + e);

}

}

/\*\*

\* Initialize the contents of the frame.

\*/

private void initialize() {

frame = new JFrame();

frame.setBounds(100, 100, 923, 548);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.getContentPane().setLayout(null);

JLabel lblNewLabel = new JLabel("Enter City Name: ");

lblNewLabel.setBounds(33, 59, 137, 14);

frame.getContentPane().add(lblNewLabel);

cityTextField = new JTextField();

cityTextField.setBounds(138, 56, 148, 20);

frame.getContentPane().add(cityTextField);

cityTextField.setColumns(10);

btnNewButton = new JButton("Get Contacts ");

btnNewButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (!cityTextField.getText().isEmpty()) {

getCityContacts();

} else

JOptionPane.showMessageDialog(null, "City Cannot Be Empty!");

}

});

btnNewButton.setBounds(33, 84, 117, 23);

frame.getContentPane().add(btnNewButton);

list = new JList<String>();

list.setBounds(10, 139, 891, 316);

frame.getContentPane().add(list);

JButton btnNewButton\_1 = new JButton("Close");

btnNewButton\_1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

frame.setVisible(false);

frame.dispose();

}

});

btnNewButton\_1.setBounds(165, 84, 89, 23);

frame.getContentPane().add(btnNewButton\_1);

}

}

## EditContactWindow

package approver;

import java.awt.Color;

import java.awt.EventQueue;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JTextArea;

import javax.swing.JTextField;

import approver.ApproverServiceStub.Contact;

import approver.ApproverServiceStub.PersonType;

public class editContactWindow {

private JFrame frame;

private JTextField streetAddresstextField;

private JTextField citytextField;

private JTextField postcodetextField;

private JTextArea resultTextArea;

private Client client = new Client();

private JTextField contactPhoneNumTextField;

private JLabel firstNameLable;

private JLabel surnameLable;

private JLabel phoneLable;

/\*\*

\* Launch the application.

\*/

public static void main() {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

editContactWindow window = new editContactWindow();

window.frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

public void getContact() {

try {

PersonType prop = client.getByPhone(contactPhoneNumTextField.getText());

firstNameLable.setText(prop.getFirstname());

surnameLable.setText(prop.getSurname());

phoneLable.setText(prop.getPhonenumber());

streetAddresstextField.setText(prop.getStreet());

citytextField.setText(prop.getCity());

postcodetextField.setText(prop.getPostcode());

resultTextArea.setText("");

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void editContact() {

PersonType prop = new PersonType();

Contact approve = new Contact();

prop.setFirstname(firstNameLable.getText());

prop.setSurname(surnameLable.getText());

prop.setPhonenumber(phoneLable.getText());

prop.setCity(citytextField.getText());

prop.setStreet(streetAddresstextField.getText());

prop.setPostcode(postcodetextField.getText());

resultTextArea.setText(client.getResults());

approve.setContact(prop);

try {

approve = client.editContact(approve);

prop = approve.getContact();

resultTextArea.setText("Contact Updated! " + prop.getStreet());

} catch (Exception e) {

e.printStackTrace();

}

}

/\*\*

\* Create the application.

\*/

public editContactWindow() {

initialize();

}

/\*\*

\* Initialize the contents of the frame.

\*/

private void initialize() {

frame = new JFrame();

frame.setTitle("Add Contact");

frame.setBounds(100, 100, 593, 416);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.getContentPane().setLayout(null);

JLabel lblNewLabel = new JLabel("Edit Contact");

lblNewLabel.setFont(new Font("Tahoma", Font.PLAIN, 14));

lblNewLabel.setBounds(21, 11, 93, 41);

frame.getContentPane().add(lblNewLabel);

JLabel lblNewLabel\_1 = new JLabel("First Name: ");

lblNewLabel\_1.setBounds(21, 109, 93, 14);

frame.getContentPane().add(lblNewLabel\_1);

JLabel lblNewLabel\_2 = new JLabel("Surname:");

lblNewLabel\_2.setBounds(23, 134, 65, 14);

frame.getContentPane().add(lblNewLabel\_2);

JLabel lblNewLabel\_4 = new JLabel("Street Address:");

lblNewLabel\_4.setBounds(21, 184, 110, 14);

frame.getContentPane().add(lblNewLabel\_4);

JLabel lblNewLabel\_5 = new JLabel("City:");

lblNewLabel\_5.setBounds(21, 209, 48, 14);

frame.getContentPane().add(lblNewLabel\_5);

JLabel lblNewLabel\_6 = new JLabel("Postcode:");

lblNewLabel\_6.setBounds(21, 234, 93, 14);

frame.getContentPane().add(lblNewLabel\_6);

streetAddresstextField = new JTextField();

streetAddresstextField.setBounds(139, 181, 124, 20);

frame.getContentPane().add(streetAddresstextField);

streetAddresstextField.setColumns(10);

citytextField = new JTextField();

citytextField.setBounds(139, 206, 124, 20);

frame.getContentPane().add(citytextField);

citytextField.setColumns(10);

postcodetextField = new JTextField();

postcodetextField.setText("");

postcodetextField.setBounds(139, 231, 124, 20);

frame.getContentPane().add(postcodetextField);

postcodetextField.setColumns(10);

JButton btnNewButton = new JButton("Edit Contact");

btnNewButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (!citytextField.getText().isEmpty() && !streetAddresstextField.getText().isEmpty()

&& !postcodetextField.getText().isEmpty()) {

editContact();

}else

JOptionPane.showMessageDialog(null, "Fields Cannot Be Empty!");

}

});

btnNewButton.setBounds(21, 276, 110, 23);

frame.getContentPane().add(btnNewButton);

resultTextArea = new JTextArea();

resultTextArea.setBackground(Color.LIGHT\_GRAY);

resultTextArea.setBounds(21, 310, 214, 22);

frame.getContentPane().add(resultTextArea);

JButton btnNewButton\_1 = new JButton("Clear");

btnNewButton\_1.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

firstNameLable.setText("");

surnameLable.setText("");

phoneLable.setText("");

streetAddresstextField.setText("");

citytextField.setText("");

postcodetextField.setText("");

resultTextArea.setText("");

}

});

btnNewButton\_1.setBounds(146, 276, 89, 23);

frame.getContentPane().add(btnNewButton\_1);

JButton btnNewButton\_2 = new JButton("Close");

btnNewButton\_2.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

frame.setVisible(false);

frame.dispose();

}

});

btnNewButton\_2.setBounds(246, 276, 89, 23);

frame.getContentPane().add(btnNewButton\_2);

JLabel lblNewLabel\_7 = new JLabel("Contact Phone Number:");

lblNewLabel\_7.setBounds(21, 52, 149, 14);

frame.getContentPane().add(lblNewLabel\_7);

contactPhoneNumTextField = new JTextField();

contactPhoneNumTextField.setBounds(180, 49, 124, 20);

frame.getContentPane().add(contactPhoneNumTextField);

contactPhoneNumTextField.setColumns(10);

JButton btnNewButton\_3 = new JButton("Get Contact");

btnNewButton\_3.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (!contactPhoneNumTextField.getText().isEmpty()) {

getContact();

} else

JOptionPane.showMessageDialog(null, "Phone Number Cannot Be Empty!");

}

});

btnNewButton\_3.setBounds(21, 77, 110, 23);

frame.getContentPane().add(btnNewButton\_3);

JLabel lblNewLabel\_3 = new JLabel("Phone Number:");

lblNewLabel\_3.setBounds(21, 159, 81, 14);

frame.getContentPane().add(lblNewLabel\_3);

firstNameLable = new JLabel("");

firstNameLable.setBounds(143, 109, 120, 14);

frame.getContentPane().add(firstNameLable);

surnameLable = new JLabel("");

surnameLable.setBounds(139, 134, 120, 14);

frame.getContentPane().add(surnameLable);

phoneLable = new JLabel("");

phoneLable.setBounds(139, 159, 124, 14);

frame.getContentPane().add(phoneLable);

}

}