Distributed Systems Assignment 2021

Cian Browne – A00254587

**Project Overview**

**Main Window**



The projects holds a HSQLDB containing the table holding the data used in the project. The server side files of the project run on a Tomcat v9.0 server. The database contains information on people and has columns for their ID, first name, last name, car, city and country. The ID value auto increments so an id does not need to be passed when creating a new user. Information is retrieved using GET requests and is processed using an XMLPullParse. The server uses a JAX:RS api to access and display the data. The client side makes use of the opencsv jar to export data to a csv file.

Main Functions:

Get: Return a list containing all the users in the database

Get id: Returns and individual person based on the id passed.

Put: Takes an id value and updates an existing person using the data from the other text fields.

Post: Takes the data from all non id fields and creates a new person the database.

Delete: Takes an id value and deletes the person from the database with the matching id.

Export: Sends a get request to the server for a list of all users, the list is then processed and printed in a data.csv file.

Requirements:

**Database:**

A table is created after setting up HSQLDB using an Ant script.

Table creation script:

create table people (

id INTEGER IDENTITY PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

car VARCHAR(50),

city VARCHAR(50),

country VARCHAR(50)

);

**Server side:**

A tomcat server is started to run the PersonResouce class that contains the necessary code to handle REST requests.

GET:

There are two get functions based on whether the client wants all users or a specific user. For all users the server calls the showAll function in the dao and returns a response with an entity containing a list of all users.

For a specific user the same process occurs but the getPerson function in the dao is called so that is returns a person object rather than a list of people.

DELETE:

The server takes in an id using a PathParam from the URI, this id calue is then passed to the deleteAccount method and deletes the user with the matching id in the database.

PUT:

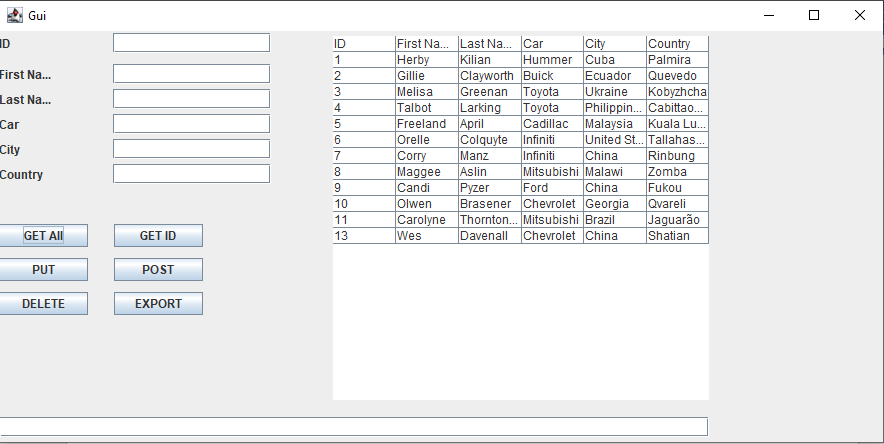
The server takes in an id using a PathParam then the remaining values in a Person object using FormParams, it then creates a person object to send to updatePerson method which it uses to update the values of a user with the matching id.

POST:

The server takes in all values to create a Person object outside of the id using FormParams, it then creates a Person object and sends it to the addPerson method. A new person is added to the database, the id value is added automatically using auto increment in the database.

**Client Side:**

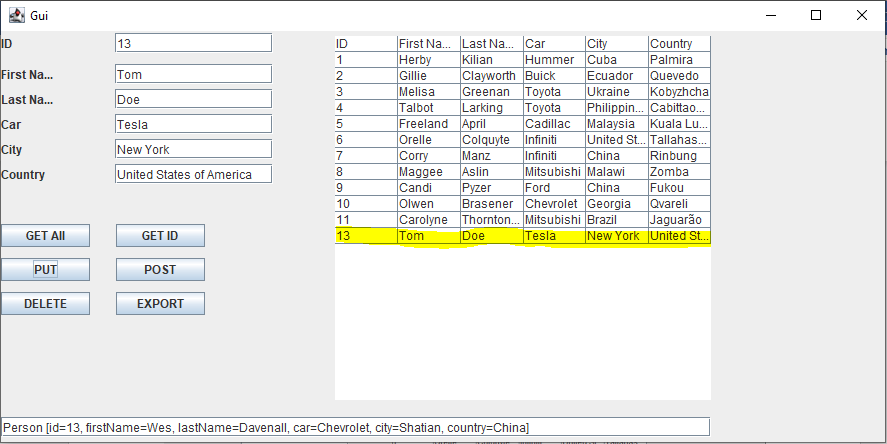
The Client side uses the buttons on the GUI to send http requests to the server. A Uri is constructed based on the button pressed setting the appropriate path, port, host, scheme and headers for the type of request taking place.



The table above displays the output of the current table stored in the database after clicking the GET All button. When the client receives the data it is parsed using an XMLPullParser and then the parsed result is sent to a fillTable method to output the information onto the JTable.

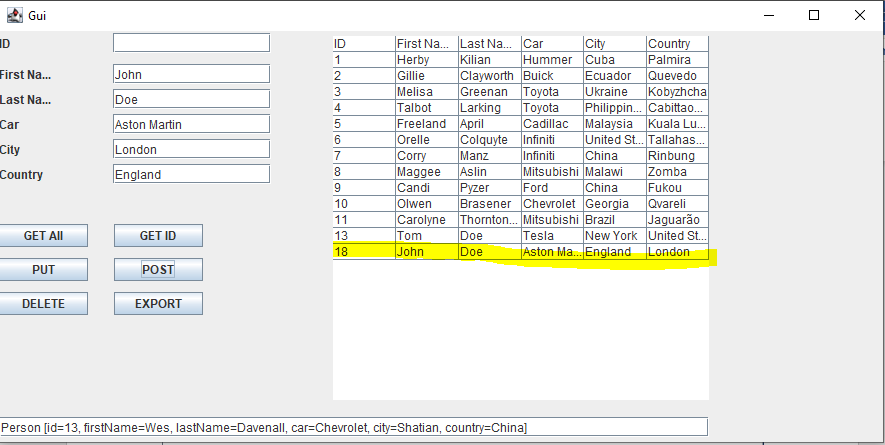


The text field at the bottom displays the Person object for a specified user, in this case the user with an id of 13.

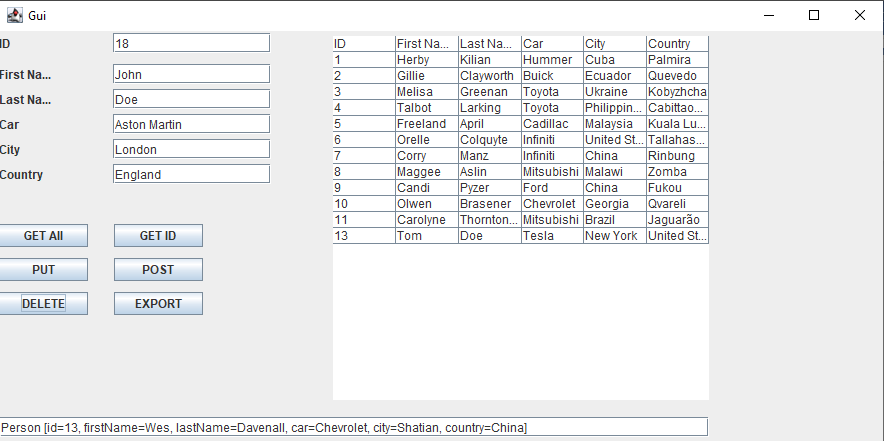


The table refreshes after sending a PUT, POST and DELETE request to the server so that it is always displaying the most up to date information.

POST



DELETE



EXPORT

