Justin Kabbes

Jkabb2

UIN 652894101

**Design and Implementation document**

**Justification**

I chose this approach for the final project because it meets all the project requirements and was the easiest for me to implement. Because the assumption is that each patient is assigned only to one doctor there was not a need for two datasets for patients and doctors each. Instead, I’ve assigned patients and doctors to one dataset with patients added into the same array string as their doctor, as I will discuss in more detail later in the document. If I were to build on this project further in the future I may consider separating doctors and patients in database files and assigning patients a doctor based on that doctor’s unique identifier (in this case the doctor’s ID).

**About**

This project uses the Restlet framework.

This project reads data about doctors and patients stored in an array and then outputs that data in a structured style in the formats plain text, XML and JSON.

Using the curl utility a user is able to perform GET, POST, PUT and DELETE operations against the data.

The project is deployed on a Tomcat Apache server using an ant script. To deploy, open the command line terminal in the restlet directory and execute the following command:

*ant -Dwar.name=doctors deploy*

The majority of the code is in the doctors directory, which contains 10 Java files:

* CreateResource.java
* JsonAllResource.java
* Main.java
* Patient.java
* Patients.java
* PlainResoure.java
* UpdateResource.java
* XmlAllResource.java
* XmlOneResource.java
* AdagesApplication.java

The array doctorsPatients, which stores the data for all doctors and patients, is located in Patients.java. That data is broken into three properties and passed to the Patient.java class. The name of the doctor is set to the property String doctor. The doctor’s ID is set to the property int id. Each doctor’s list of patients and their corresponding insurance ID numbers are set to the property array String[] words2.

That data is then structured in the toString() method in the Patient.java class. The method outputs each doctor’s ID and name using the doctor and id variables. It then separates the patient names and their insurance numbers using a for loop. The final product looks something like the following:

Dr. ID: 3 / Doctor name: JimDr

Patient #1: KimSick ==> CC123

Patient #2: EmilySick ==> CC456

Patient #3: JackSick ==> CC789

Dr. ID: 2 / Doctor name: JohnDr

Patient #1: RoseSick ==> CC111

Patient #2: LouSick ==> CC222

Use the following commands to perform CRUD operations against the web service:

* *curl --request GET http://localhost:8080/doctors/*
* *curl --request POST --data “doctor=EmberDr!1!JustinSick!CC101” http://localhost:8080/doctors/create*
* *curl --request PUT -d id=1 -d doctor=”TwinklesDr” http://localhost:8080/doctors/update*
* *curl --request DELETE http://localhost:8080/doctors/delete/1*

In order to perform POST operations the user must format the –data input in the following format

* [Doctor name] ! [Doctor ID] ! [Patient 1 name] ! [Patient 2 insurance ID] ! [Patient 2 name] ! [Patient 2 insurance ID] ! … [Patient x name] ! [Patient x insurance ID]
* All entities such as the names and IDs are separated using an exclamation point “!”.

**Troubleshooting**

The most difficult part of troubleshooting this assignment was determining how to break up the array data and pass it to the Patient class. I decided to break up the array into parts using *split.* Parts 1 and 2 are used to pass the doctor’s name and doctor’s ID. I then use a for loop to determine how many patients are assigned to each doctor and then split their names and insurance ID numbers into parts and pass each one to the Patient class.