

DHS - Java Assignment

Planets Route Mapper

This project allow you to find the shortest path from point “A”, being Earth, through the galaxy to any of the planets represented by the other nodes. It calculates the distance on the basis of provided list with node names and their respective distances between their linked planets. In addition it helps you to find distance from any source to destination coordinates specified

Mahesh Gadupudi.

Date: 2018/11/22

Contents

[1. Project Overview 3](#_Toc333858467)

[2. Aims And Objectives 3](#_Toc333858468)

[3. Tools and Technologies 3](#_Toc333858469)

[4. Implementation Plan 3](#_Toc333858470)

[5. Project Structure Description 3](#_Toc333858471)

[6. Deployment Description 3](#_Toc333858472)

[7. User Guide For Deployment 3](#_Toc333858473)

[8. Related Screenshots 4](#_Toc333858474)

## Project Overview

**This project was designed according to the assigned task.**

## As per the graph given in assignment, which is related to interstellar transport system used by Earth’s inhabitants in the year 2145. This project was built in such way to find the shortest path from point “A”, being Earth, through the galaxy to any of the planets represented by the other nodes as XL sheet provided with a list of node names and their respective distances between their linked planets. In addition, you can find distance between any source and destination as coordinates specified.

## Aims and Objectives

## To build in-memory database to persist the graph node and values as provided.

## Persist the data using ORM tool.

## Generate Bindings

## Import the data from XL sheet into database (XL provided)

## Expose database as RESTful web service

## Expose curd operations to manage database

## Create the algorithm

## Object model to describe route

## Algorithm to determine shortest path between any nodes.

## Create Document Literal Web service

## Construct an XSD to constrain usage of the service

## Expose the Route Request document and return a Route Response document with the hops

## Create a front end to capture the source and destination and then print the shortest path

## Tools and Technologies

**Tools and technogies used according to the requirements:**

**Java Version**

* **Java: 1.7**

**Frameworks Used**

* **Action-based frameworks:** [**Spring MVC**](http://en.wikipedia.org/wiki/Spring_Framework)
* **Component-based frameworks: Java Server Faces**

**Database**

* **Apache Derby DB**

**Object Relational Mapping Tools:**

* **Hibernate**

**Web Services:**

* **Document Literal Web service**
* **RESTful Web service**

**Algorithm**:

* **Dijkstra’s**

**IDE’S:**

* **Eclipse Luna**
* **NetBeans 8.0.2**

**Server**

* **Glassfish 4.1**

**Build Tools**

* **Apache maven**

## Implementation Plan

**As per my knowledge, I divided the assignment into two parts.**

**Part 1 (Restful Web services):**

* It helps us to create tables automatically by using hibernate
* It reads data from provided XL sheet using apache poi and import it into derby database.
* This project using spring web service to expose database.
* Spring MVC is used to build basic frontend which helps us navigate through the tables imported.
* Web Application provides CRUD operations to manipulate database.
* Spring web services are used to manipulate data.

**Part 2 (Planet Route Mapper):**

* This project helps to calculate shortest path between two nodes.
* XSD are created to map request and response objects.
* Dijkstra’s algorithm is used as main part of this application to calculate distance between two nodes
* Source and destinations are passed as request object to get distance.
* Document web services are built in this project to communicate to database which is created in first project.
* This project using spring 3, hibernate4 , JSF 2.2 to achieve the goals for the assignment

## Project Structure Description

Current two projects are working MVC based project.

* **Model** - The model represents data and the rules that govern access to and updates of this data. In enterprise software, a model often serves as a software approximation of a real-world process.
* **View** - The view renders the contents of a model. It specifies exactly how the model data should be presented. If the model data changes, the view must update its presentation as needed. This can be achieved by using a push model, in which the view registers itself with the model for change notifications, or a pull model, in which the view is responsible for calling the model when it needs to retrieve the most current data.
* **Controller** - The controller translates the user's interactions with the view into actions that the model will perform. In a stand-alone GUI client, user interactions could be button clicks or menu selections, whereas in an enterprise web application, they appear as GET and POSTHTTP requests. Depending on the context, a controller may also select a new view -- for example, a web page of results -- to present back to the user.

**Spring Web Flow**

* Spring Web Flow (SWF) aims to be the best solution for the management of web application page flow.
* SWF integrates with existing frameworks like Spring MVC and JSF, in both Servlet and Portlet environments. If you have a business process (or processes) that would benefit from a conversational model as opposed to a purely request model, then SWF may be the solution.
* SWF allows you to capture logical page flows as self-contained modules that are reusable in different situations, and as such is ideal for building web application modules that guide the user through controlled navigations that drive business processes.

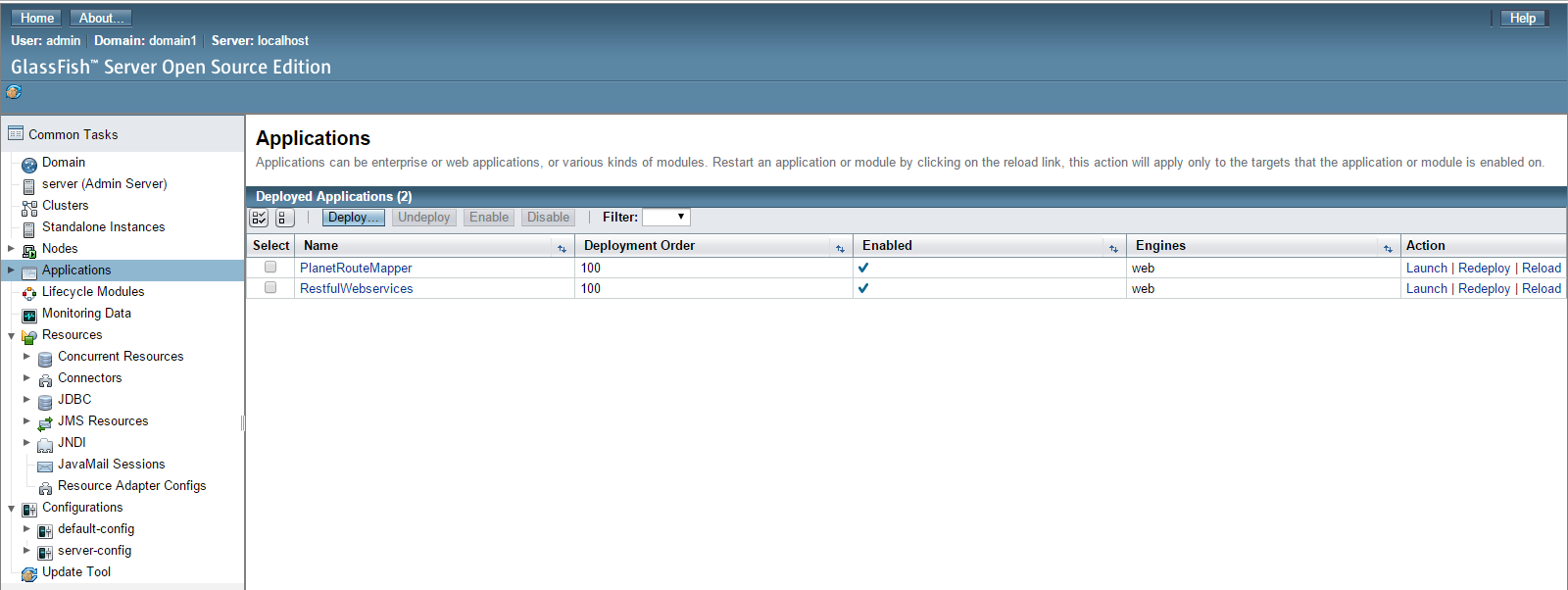
## Deployment Description

In this project I choose glassfish Server to deploy and run the application.

To Deploy the Current Project you can follow the below procedure.

1. Open glassfish folder and navigate to bin.
2. ./asadmin deploy path of the war/PlanetRouteMapper-1.0-SNAPSHOT.war
3. Open internet explorer or chrome. E.g.:**localhost:8080/PlanetRouteMapper**

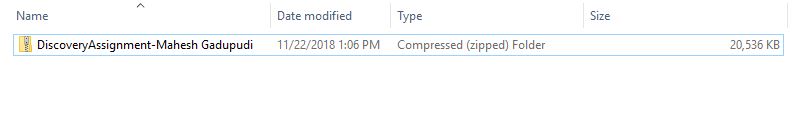
Graphical interface of deployed application



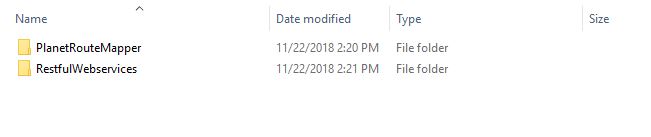
## User Guide

Screen 1:

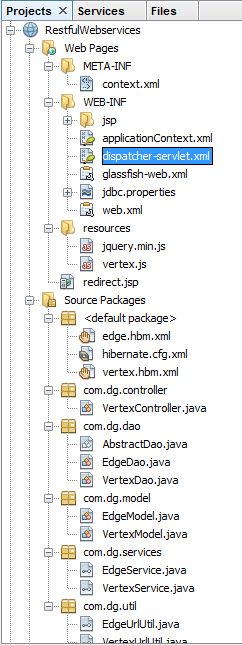
1. Open the downloaded project



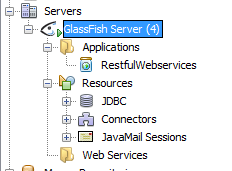
1. Extract it into a folder



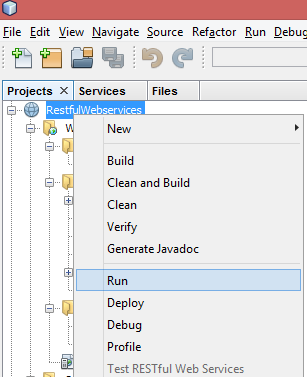
1. Import the project to workspace



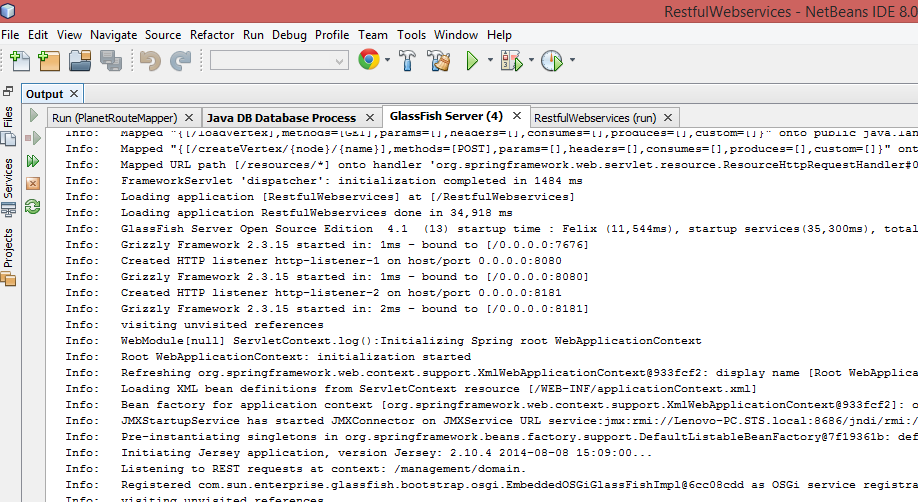
1. Configure Glassfish server



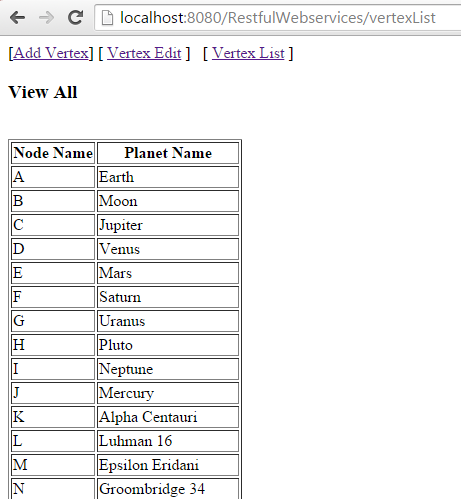
1. Right click on the project and select run



1. Console of deploying application

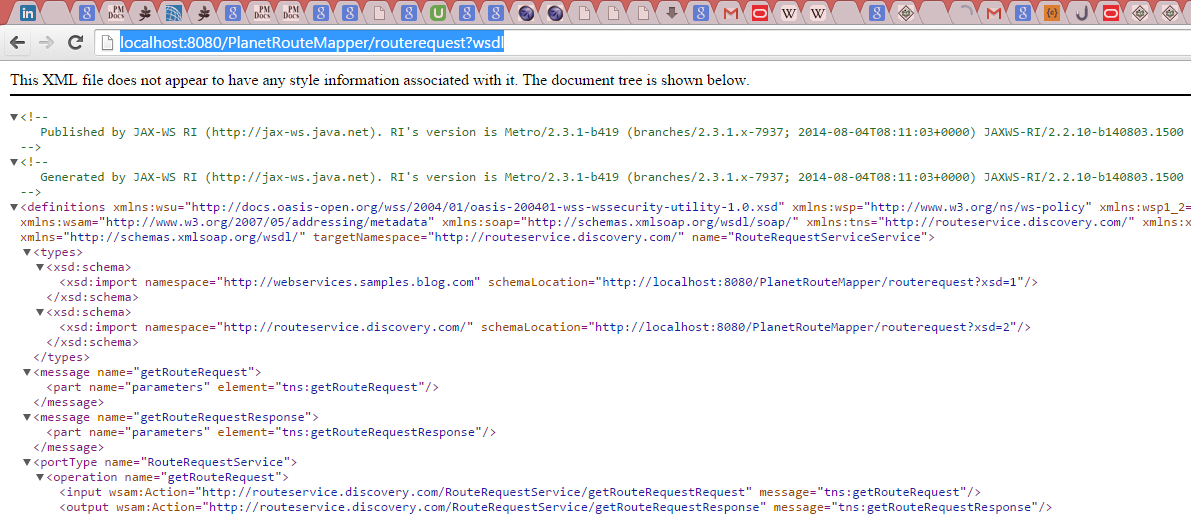


1. Final screen of deployed Application



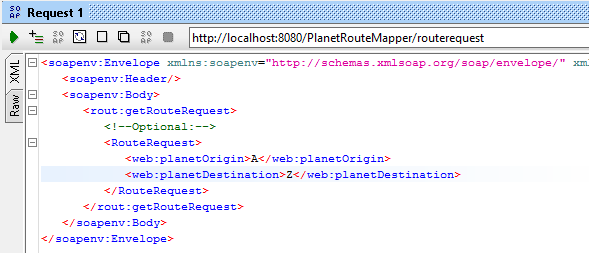
**Planet Route Mapping**

1. **Need to follow the same procedure to deploy. Please attached screen shots of deployed application**

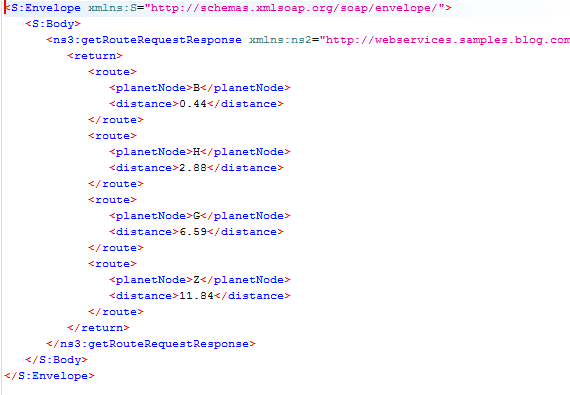


**Testing with SOAP UI:**

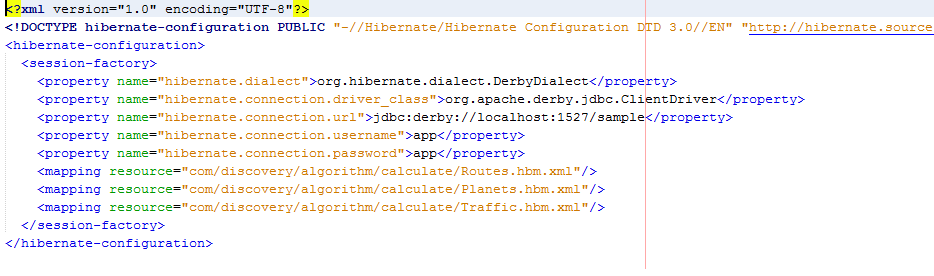
1. Send Request with Parameters: A, B

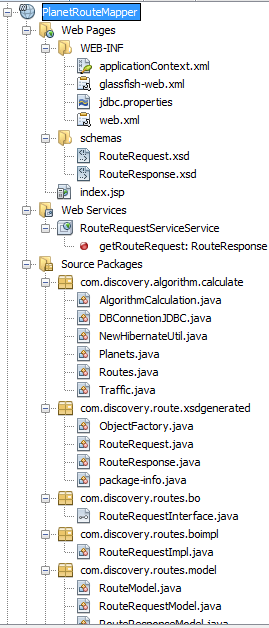


**Response From Server :**



## Related Screenshots







Mahesh Gadupudi 2018/11/22

Date: