Planet Database

Version 1.0

04/03/2019

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 04/01/2019 | 1.0 | Initial Version | Saji Vijayakumari Sadhasivam |

**Sign-Off**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Role** | **Print Name** | **Date** | **Version Number** | **Signature** |
| Project Manager | Damien Hanna |  | 1.0 |  |
| Senior Software Engineer | Saji Vijayakumari Sadhasivam |  | 1.0 |  |

Table of Contents

[1. Introduction 5](#_Toc5197851)

[2. Architecture 5](#_Toc5197852)

[3. Functional Directives 5](#_Toc5197853)

[4. Technical Directives 6](#_Toc5197854)

[5. Technical Components 6](#_Toc5197855)

[5.1.1 Databse Components 6](#_Toc5197856)

[5.1.2 UI Components 7](#_Toc5197857)

[5.1.3 Backend Components 7](#_Toc5197858)

[5.1.4 GET API JSON Format 7](#_Toc5197859)

[6. Screen Details 9](#_Toc5197860)

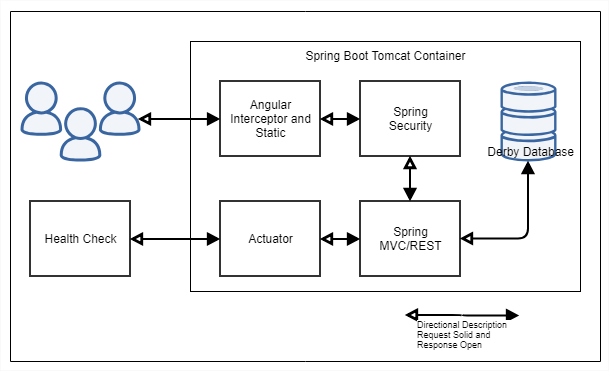
[7. Build and Run Application 10](#_Toc5197861)

[8. Assumptions 10](#_Toc5197862)

# Introduction

Objective of Planet Database application is to provide users with information and knowledge about the planets in our solar system. Current scope of the project is to display the distance of each planets from Sun.

# Architecture



Planet Database application has a Front-end component built with AngularJs which will display the UI to the user. Backend REST API along with Security and Health Check will be built in Spring boot framework. Derby in memory database will be used to store planet data. GetAllPlanets API will be used to retrieve the planet details from database and expose as JSON Response. Basic authentication will be provided for the GET API. One-time query will be fired by the application during startup to insert Planet data to DB. Health check API will be provided to check the status of the GET API. Current implementation will have both the UI and Backend component running on the same tomcat server provided by spring boot. But these components can be easily decoupled to run on separate servers as well.

Below tools will be used to build thee application.

| **Component Name** | **Purpose/Description** |
| --- | --- |
| Angularjs | 1.7.8 |
| Bootstrap | 3.4.0 |
| Java | 1.8 |
| Spring boot | 2.1.3 |

# Functional Directives

|  |  |
| --- | --- |
| **Id** | **Description** |
| 1 | Users will be displayed with a landing page on entering the URL (http://localhost:8085). Landing page will have Header, footer and body section. Header will display the logo, title and Menus. Footer will display the copyright information. Body will have two different sections, one to display the application details and other to display the Planet details |
| 2 | Right side of the body will display the list of planets. On click of each Planet name, a short description will be displayed with distance of the planet from the Sun. Description for all the planets can be viewed by clicking on the title bar of the Planet List. Description will be displayed with Expand and collapse functionality. |

# Technical Directives

|  |  |
| --- | --- |
| 1 | UI will be developed in AngularJS 1.7, UI design will be done in plain html and CSS. UI operations will be performed through controller and service. Inreceptor will be created to bypass all http request and add basic auth header. Directive will be created to add expand and collapse functionality. |
| 2 | Maven project will be created with dependencies for Spring boot started Web, Security, JPA and Derby.  Web security will be provided using Basic authentication. JPA entities and Repositories will be created to retrieve Planet details. Rest controller will serve the API in JSON Format through a service. |
| 3 | Actuator framework will be used to provide the Health status and info of the API. |
| 4 | data.sql file will be created for a One time insert of planet details. A POST API will also be provided to insert data. Schema and table will be created by the application on startup. |
| 5 | Jenkins file will be provided to build a pipeline |

# Technical Components

Code Base will be available in GITHUB over the below URL.

<https://github.com/sajivijaysadas/PlanetDatabase>

## Database Components

PLANET\_DETAILS – New Table, Class name – Planet.java

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Element** | **Data Type** | **Database Field** | **Data Type** |
| planetid | LONG | PLANET\_ID | NUMBER , PRIMARY KEY |
| planetName | STRING | PLANET\_NAME | VARCHAR |
| IMAGE | STRING | IMAGE | VARCHAR |
| distance | STRING | DISTANCE | VARCHAR |

**Insert Entries:-Query** data.sql

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (1,'Mercury','Mercury.JPG','200,900,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (2,'Venus','Venus.JPG','300,800,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (3,'Earth','Earth.JPG','400,700,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (4,'Mars','Mars.JPG','200,900,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (5,'Jupiter','Jupiter.JPG','300,800,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (6,'Saturn','Saturn.JPG','400,700,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (7,'Uranus','Uranus.JPG','200,900,000');

**INSERT** **INTO** PLANET\_DETAILS (PLANET\_ID,PLANET\_NAME,IMAGE,DISTANCE) **VALUES** (8,'Neptune','Neptune.JPG','300,800,000');

## UI Components

| **Component Name** | **Purpose/Description** |
| --- | --- |
| app.js | To initialize the angular module |
| auth.Interceptor.js | To intercept all http request and add basic auth header |
| expand.directive.js | To create a expand collapse directive |
| index.html | To design the landing page |
| planet.controller.js | To have Angular functionalities/operations |
| planetdata.service.js | Service to invoke the API calls |
| style.css | Provide style to the page |

## Backend Components

| **Component Name** | **Purpose/Description** |
| --- | --- |
| com.lexisnexis.PlanetDatabase.controller. PlanetDatabaseAppController.java | To serve the API request and response |
| com.lexisnexis.PlanetDatabase.model. Planet.java | Model Entity to map the table |
| com.lexisnexis.PlanetDatabase.Repository. PlanetDataRepository.java | Dao to get the planet details from DB |
| com.lexisnexis.PlanetDatabase.security.PlanetDatabaseSecurityConfig.java | To add security configuration-Basic Auth |
| com.lexisnexis.PlanetDatabase.service.PlanetDatabaseAppService.java | To add business logic |
| application.properties | Configure Actuator , server port , application info |
| Pom.xml | Dependency for spring boot , derby, tomcat. Available in repository |

## GET API JSON Format

[

{

"planetId": 1,

"planetName": "Mercury",

"image": "Mercury.JPG",

"distance": "200,900,000"

},

{

"planetId": 2,

"planetName": "Venus",

"image": "Venus.JPG",

"distance": "300,800,000"

},

{

"planetId": 3,

"planetName": "Earth",

"image": "Earth.JPG",

"distance": "400,700,000"

},

{

"planetId": 4,

"planetName": "Mars",

"image": "Mars.JPG",

"distance": "200,900,000"

},

{

"planetId": 5,

"planetName": "Jupiter",

"image": "Jupiter.JPG",

"distance": "300,800,000"

},

{

"planetId": 6,

"planetName": "Saturn",

"image": "Saturn.JPG",

"distance": "400,700,000"

},

{

"planetId": 7,

"planetName": "Uranus",

"image": "Uranus.JPG",

"distance": "200,900,000"

},

{

"planetId": 8,

"planetName": "Neptune",

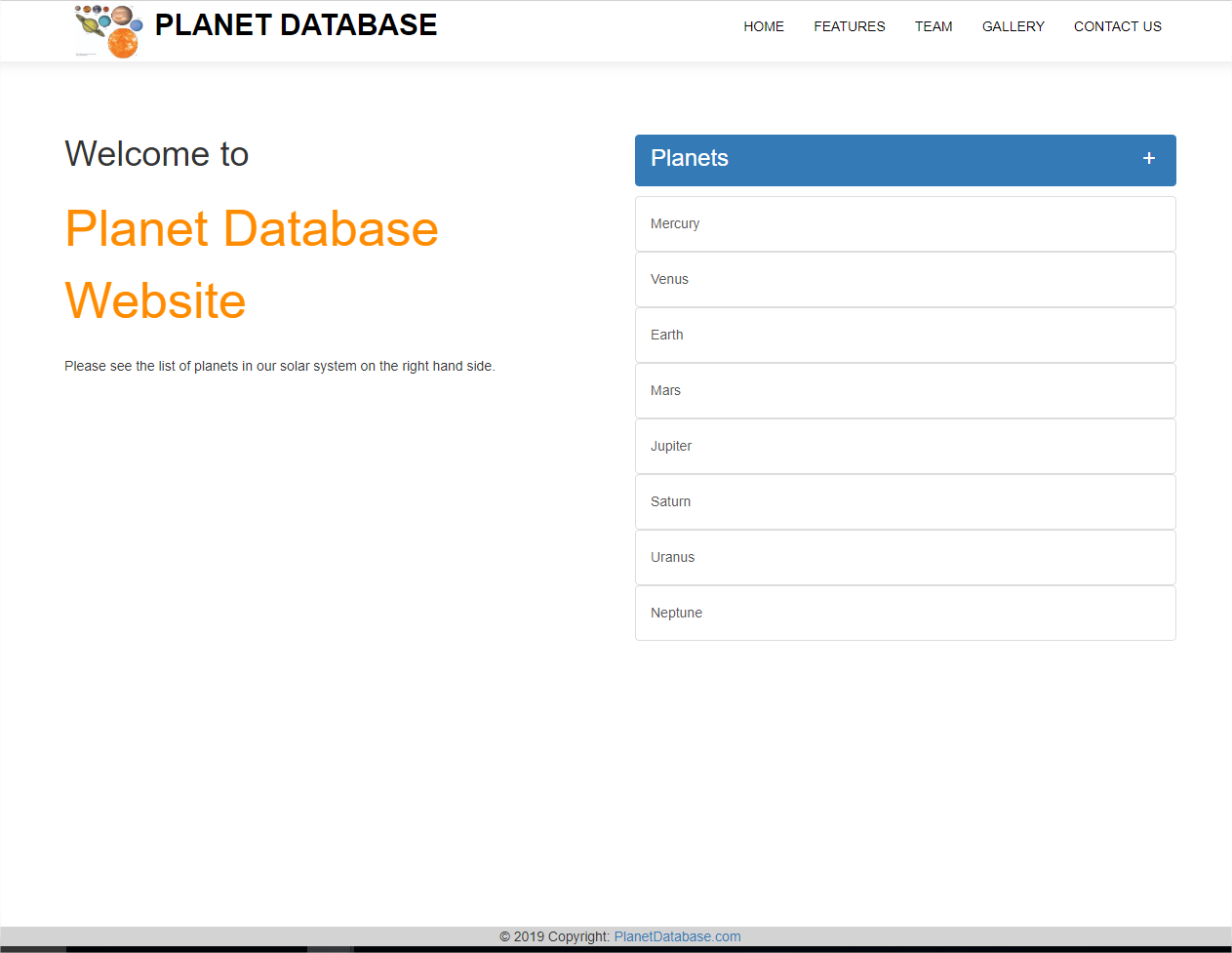
"image": "Neptune.JPG",

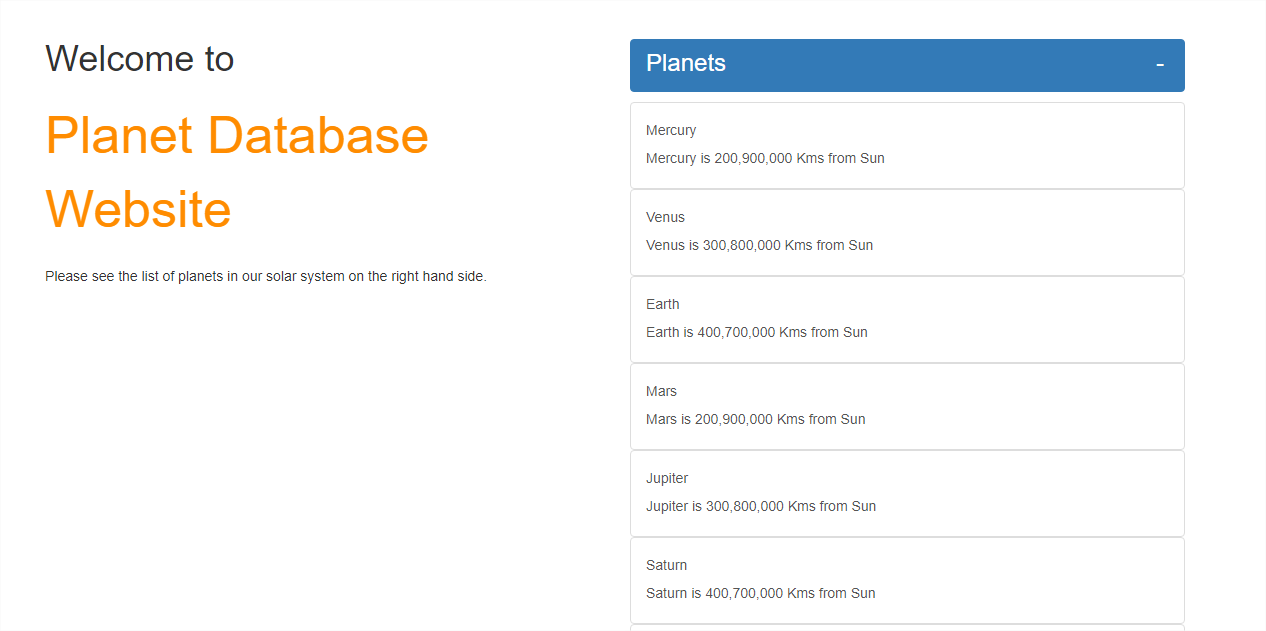
"distance": "300,800,000"

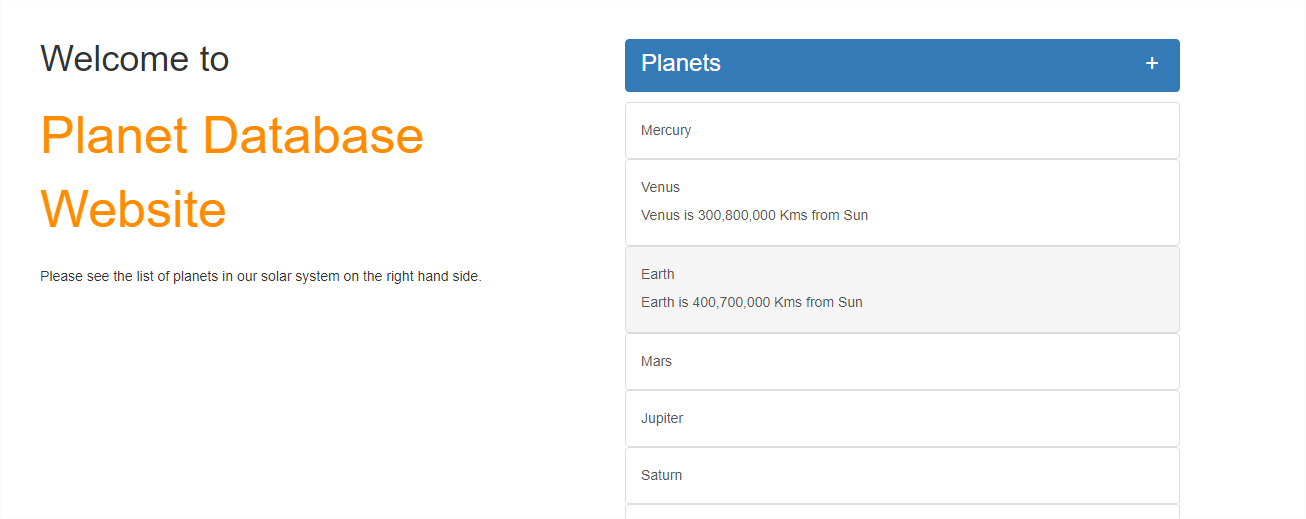
}

]

# Screen Details







# Build and Run Application

**Pre-requisites to build application – JDK 1.8, Maven 3.6**

Steps:-

1. Clone the Project from the GITHUB repository(<https://github.com/sajivijaysadas/PlanetDatabase.git> ).
2. In Eclipse , import as Existing Maven Project (Optional)
3. Open command Prompt and navigate to project location
4. Run mvn clean install to generate latest jar
5. Run java -jar target/PlanetDatabase-0.0.1-SNAPSHOT.jar
6. Hit <http://localhost:8085/>

Note: A copy of the Jar file is available in GITHUB repository under target folder. Run java -jar target/PlanetDatabase-0.0.1-SNAPSHOT.jar. But this wouldn’t be the latest version.

# Assumptions

|  |  |
| --- | --- |
| **Id** | **Description** |
| 1 | Application will only display the Planet details. Add, Update, Delete will not be supported in current version. |