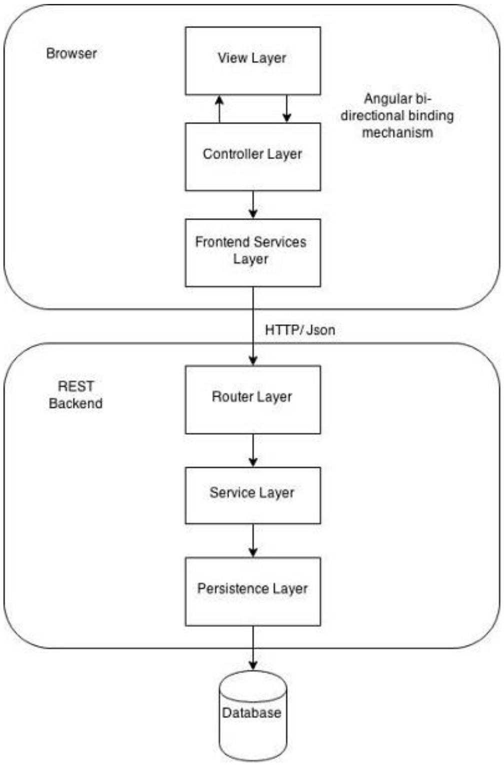
Currency Exchange App Design Document

Version 1

Diego Potapczuk
potapczuk@gmail.com

1 Architecture Layers

The application is designed in two parts, the client web application and the backend server. As show in the following diagram.



The client is divided in 3 layers, the View, Controller and Frontend Services.

- The View Layer is where the HTML and CSS elements are presents and reflect what the user should see in the application.

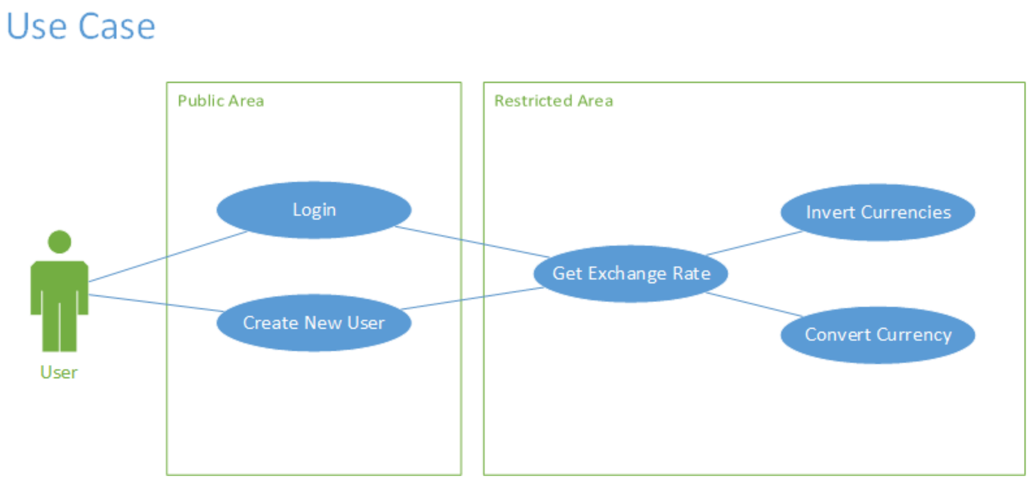
- The Controller Layer is responsible for controlling the model data of the application, it controls the application and information flow of the view and service layers.
- The Service Layer manage the more complex business logic of the Client App and interact with a REST server to consume data.

The Backend is also composed of 3 layers, the Router, Service and Persistence Layer.

- The router is responsible for providing the REST API for clients to consume.
- The Service Layer is responsible for the business logic of the backend application.
- The Persistence Layer interact with a database system to persist and retrieve data.

2 USE CASE

The application was design with a main user that has two actions in a public area, Login and Create New User. After this User is logged in or has created a new user, he can Get Exchange Rate for two currencies, Convert a value of a currency to another currency and Invert the Currencies. This is demonstrated in the following diagram.



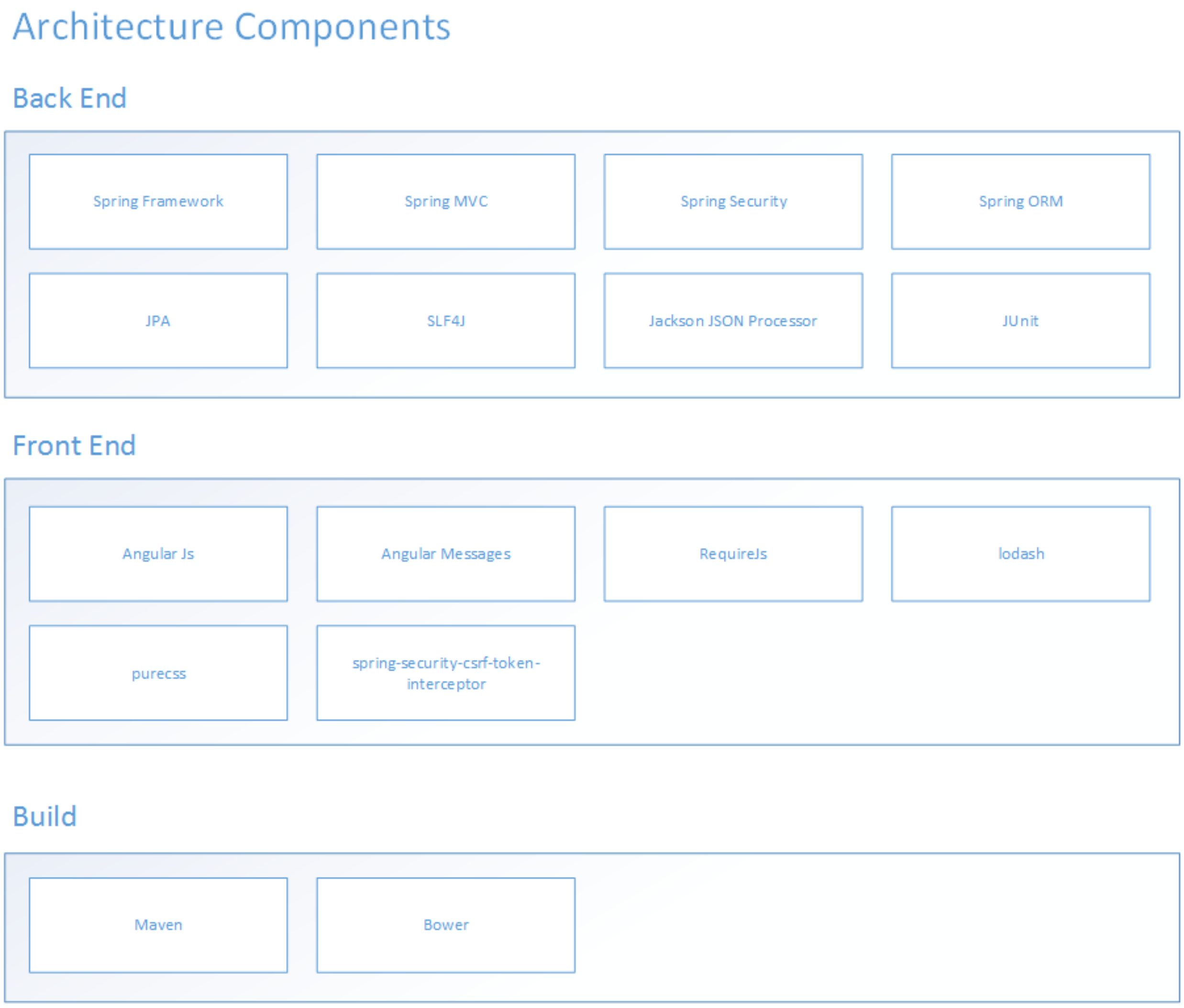
For a User to Login, he need a valid username and password.

For a User to Create New User, he need a valid username with more than 6 letters, a valid email and a password with at least 1 lower case letter, 1 upper case letter, 1 number and a minimum of 6 characters.

For a User to Get Exchange Rate, he need to choose 2 currencies from a list of currencies and they need to be different from each other.

3 COMPONENTS OF THE ARCHITECTURE

The application is composed of many components in 3 main parts, backend, frontend and build as show in the following image.



The backend was developed with the Java Language, using Java SE 8, Java EE 6 and some components such as:

- Spring Framework for the Context and Dependency Injection.
- Spring MVC for the REST, Controller and Web functionalities.
- Spring Security for the authentication and authorization.
- Spring ORM and JPA for the persistence.
- SLF4J for the logging in the application.
- Jackson JSON Processor to convert Java to JSON in the communication with clients.
- JUnit for the application automated tests.

The Frontend was developed using HTML, CSS and JavaScript and components such as:

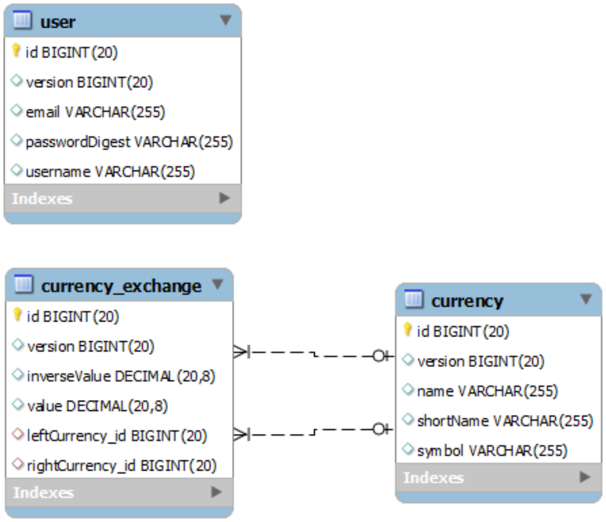
- Angular JS for an easier and better web client app development and the bi-directional binding between JS and the HTML.
- Angular Messages an extension of Angular JS with some functions to control messages display.
- RequireJS to make it easier to control the load of JS files for the client application.
- Lodash, a functional programming library to manipulate data in JS.
- Purecss, a Yahoo CSS framework to help in the construction of a responsible layout.
- Spring Security CSRF token interceptor, to help against CSRF attacks.

The Frontend dependency is managed using the bower and the backend using Maven.

The build is done using the Maven, and the application can be tested using a tomcat7 mavem plugin.

4 Entity Relationship Diagram

In the following diagram is show how the database is structured

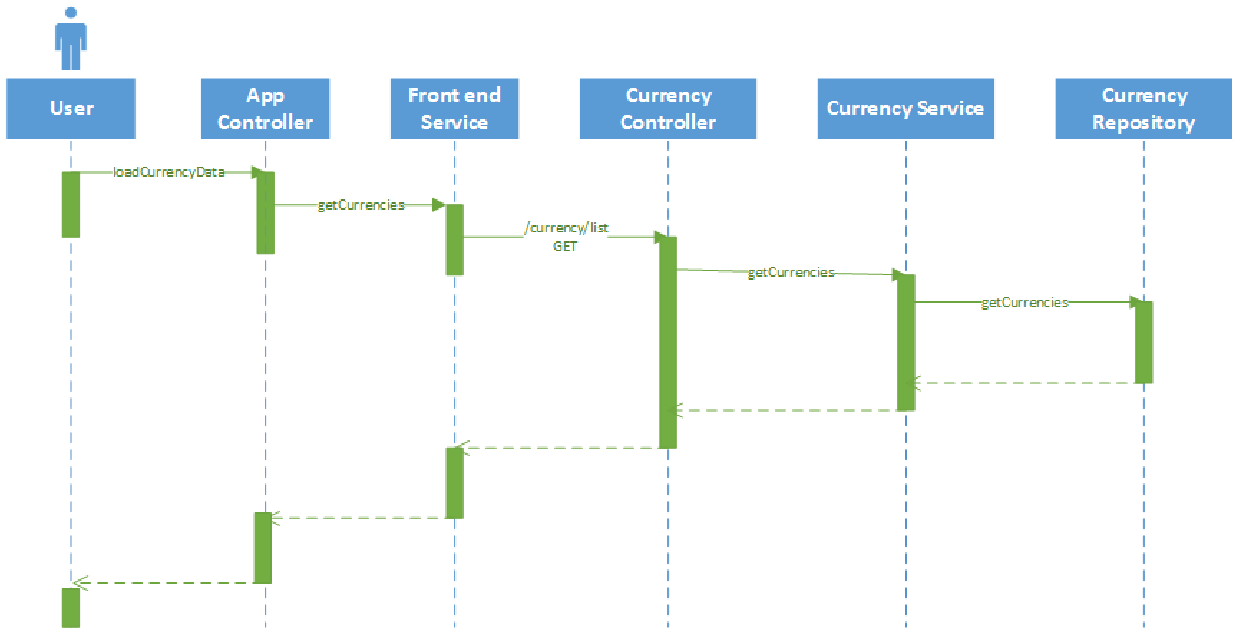


5 Sequence Diagram

The Get Currencies sequence of interactions between the application parts is displayed in the following diagram.

Sequence Diagram

Get Currencies



6 REST API

Authentication Service

Url	Verb	Description
/authenticate	POST	user authentication
/logout	POST	logout of the application

Currency Service

Url	Verb	Description
/currency/list	GET	get a list of currencies
/currency/exchage	GET	get the information of currency exchange rate between 2 currencies