Install and Run Login Server and Client

Run Server

To run the server, you can download the jar artifact from the url:
https://github.com/ecarrilloq/login-test/blob/master/artifacts/login-0.0.1-SNAPSHOT.jar

You need to have java installed on your PC.

Then open the folder where you save the jar file and, in a terminal, or a command window run the next command: java -jar login-0.0.1-SNAPSHOT.jar

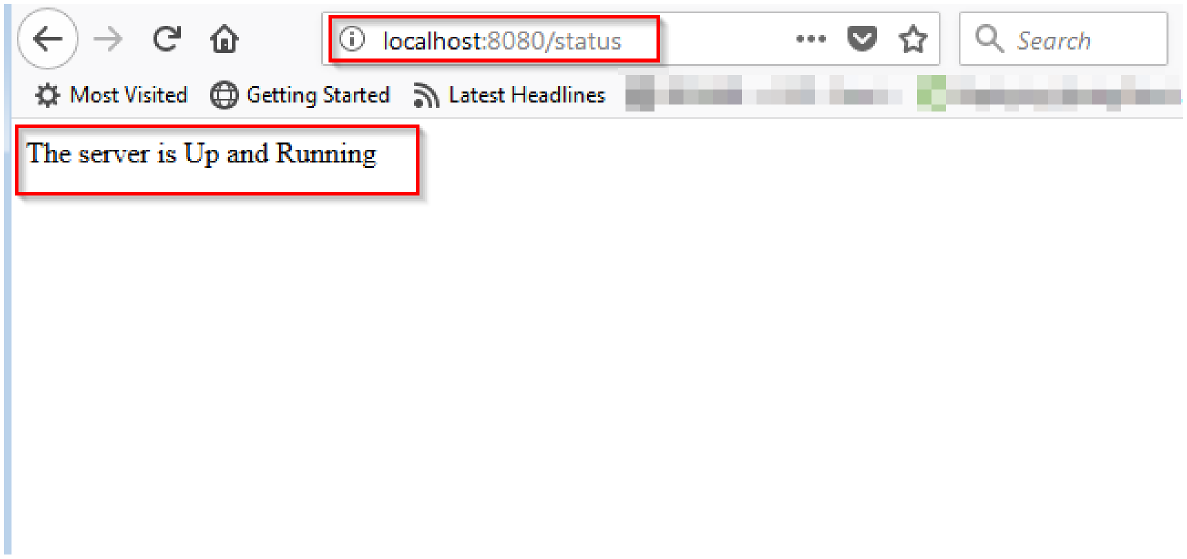
If the server runs ok, the console must show at the end a message like:

```
Adding User admin
Adding User user
…
```

The server runs on port 8080

To stop the server just press "CTRL+C" on the command window where the server is running.

The server can be tested on the url: http://localhost:8080/status



The server is Up and Running

Build Server on Terminal

To build the server you need to have installed on your PC git, java and maven.

The first step is to create a work folder and cd to it.

Then download the project from the repository "`https://github.com/ecarrilloq`" using the command: `git clone https://github.com/ecarrilloq/login-test.git .`

cd to the folder: **back-end/login**

build the project with maven using the command: **mvn clean install**

after finishing the build, cd to the folder target

run the server with the command **java -jar login-0.0.1-SNAPSHOT.jar**

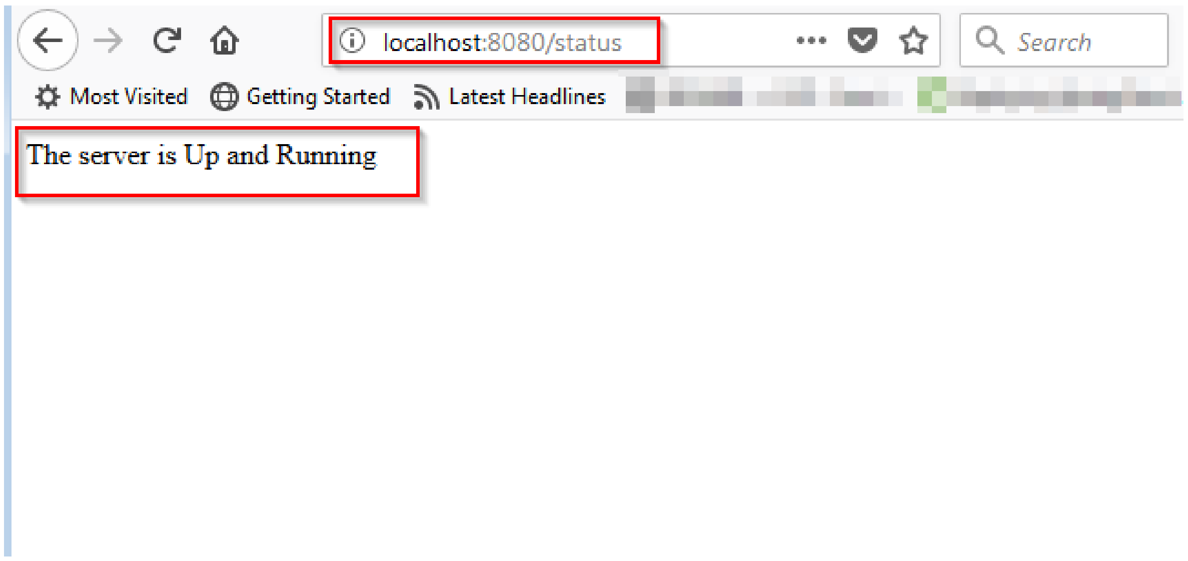
If the server runs ok, the console must show at the end a message like:

```
Adding User admin
Adding User user
…
```

The server runs on port 8080

To stop the server just press "CTRL+C" on the command window where the server is running.

The server can be tested on the url: http://localhost:8080/status

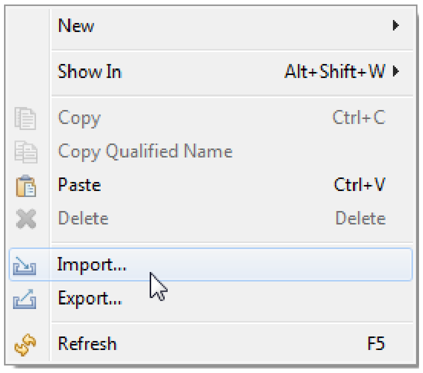


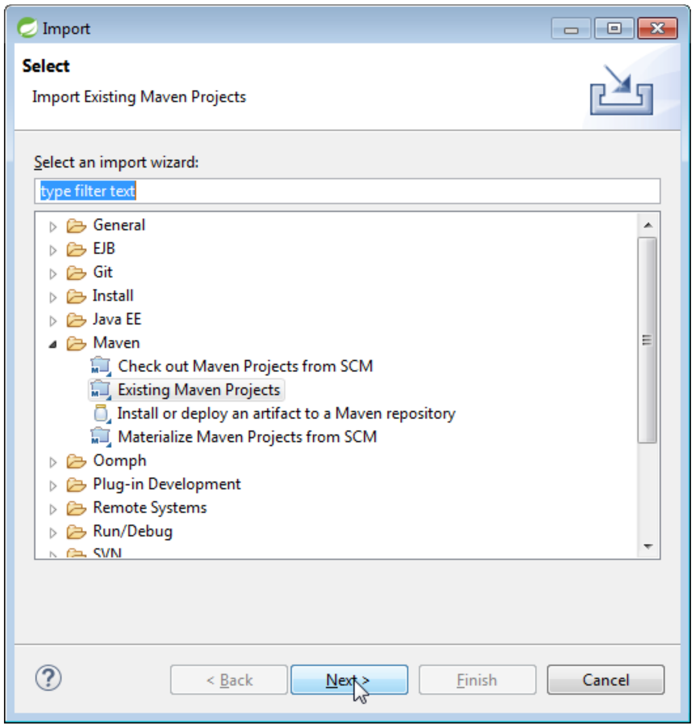
The server is Up and Running

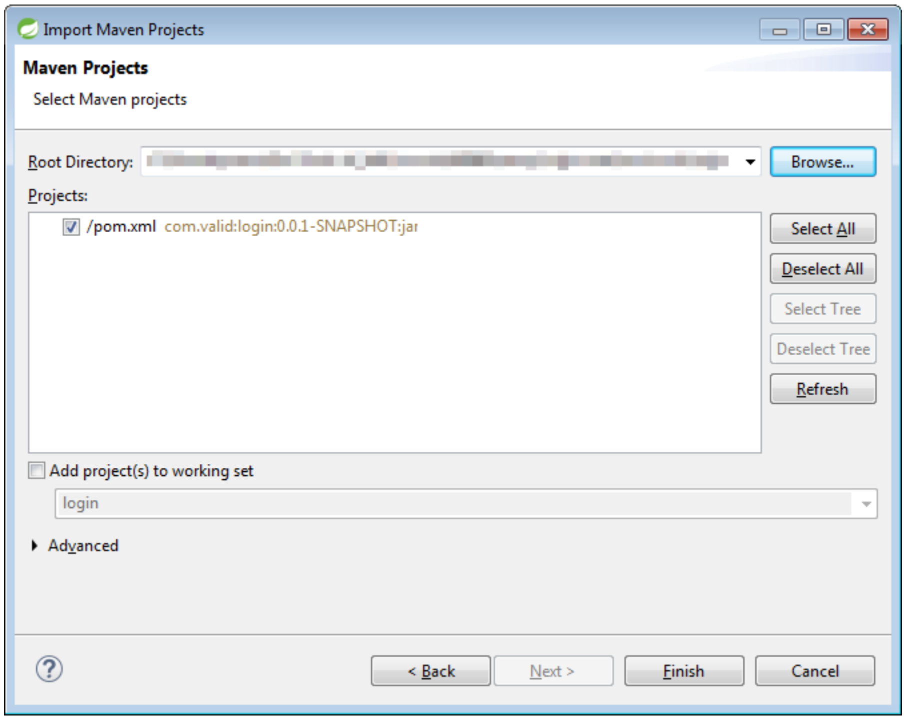
Build the server on an IDE

I recommend using Spring Tool Suite as IDE.

First create a workspace and then import maven project into the workspace



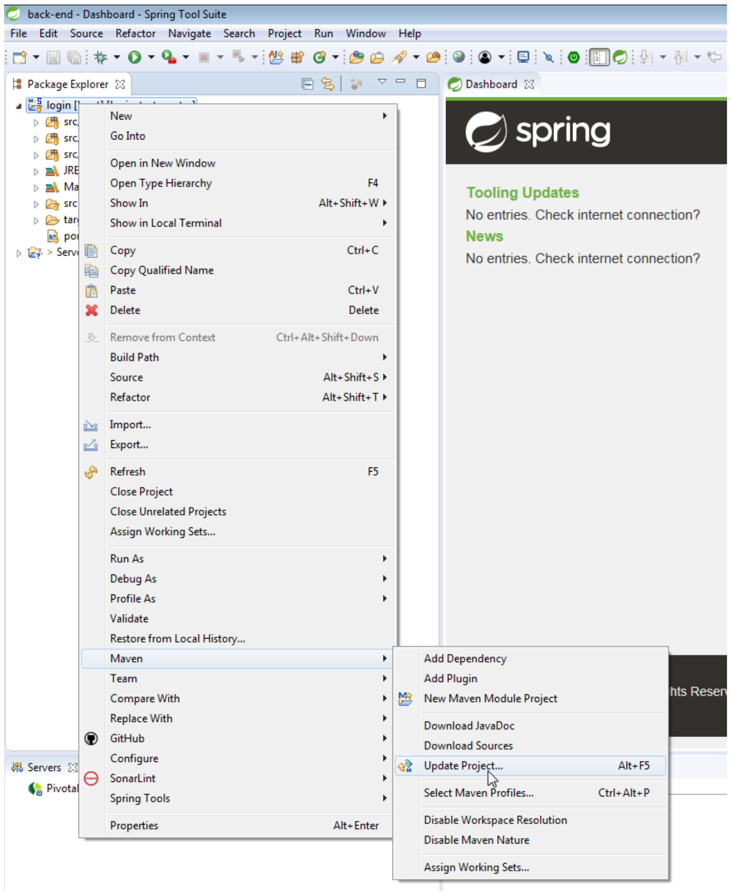


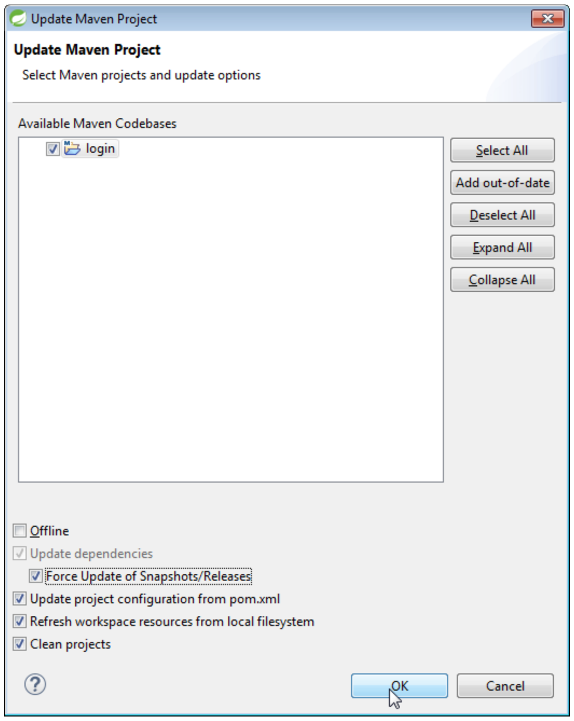


[✓] /pom.xml com.valid:login:0.0.1-SNAPSHOT:jar

[ ] Add project(s) to working set: login

After import the project run a maven update to download all the dependencies.





Update Maven Project

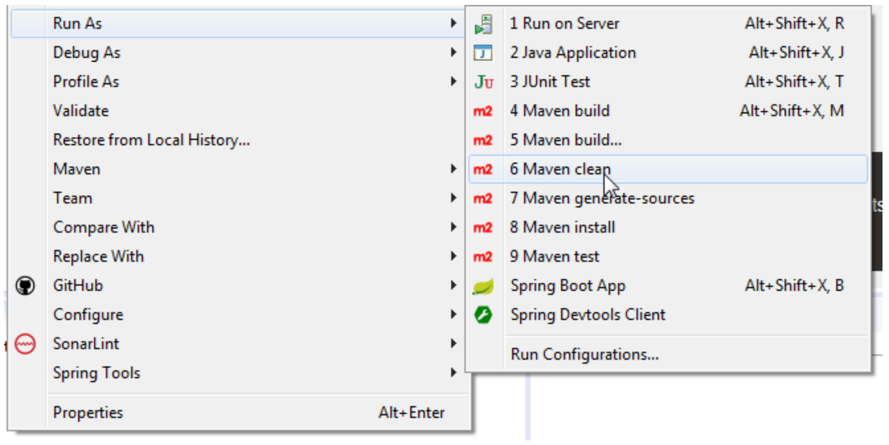
Select Maven projects and update options

Available Maven Codebases

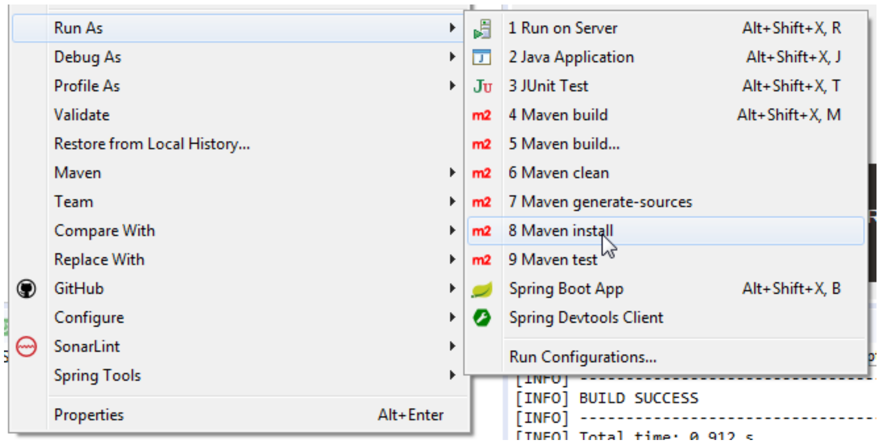
- [✓] login

- [ ] Offline
- [✓] Update dependencies
  - [✓] Force Update of Snapshots/Releases
- [✓] Update project configuration from pom.xml
- [✓] Refresh workspace resources from local filesystem
- [✓] Clean projects

Then run a Maven Clean.

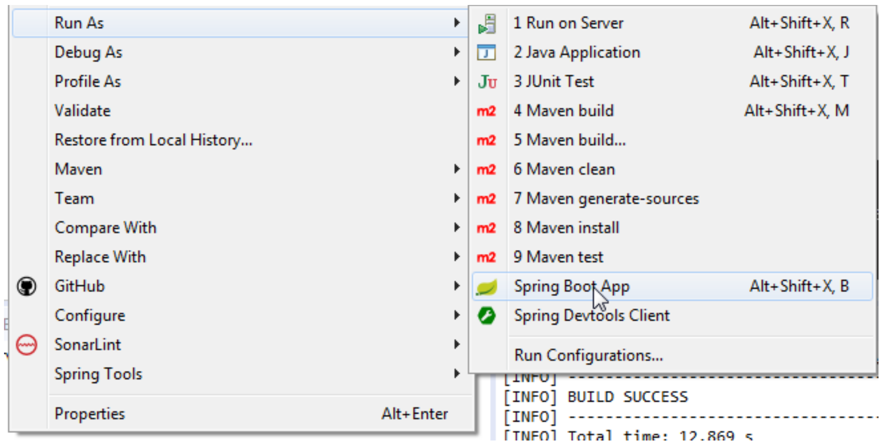


After the Maven Clean, run a Maven Install to compile the project and build the artifact.

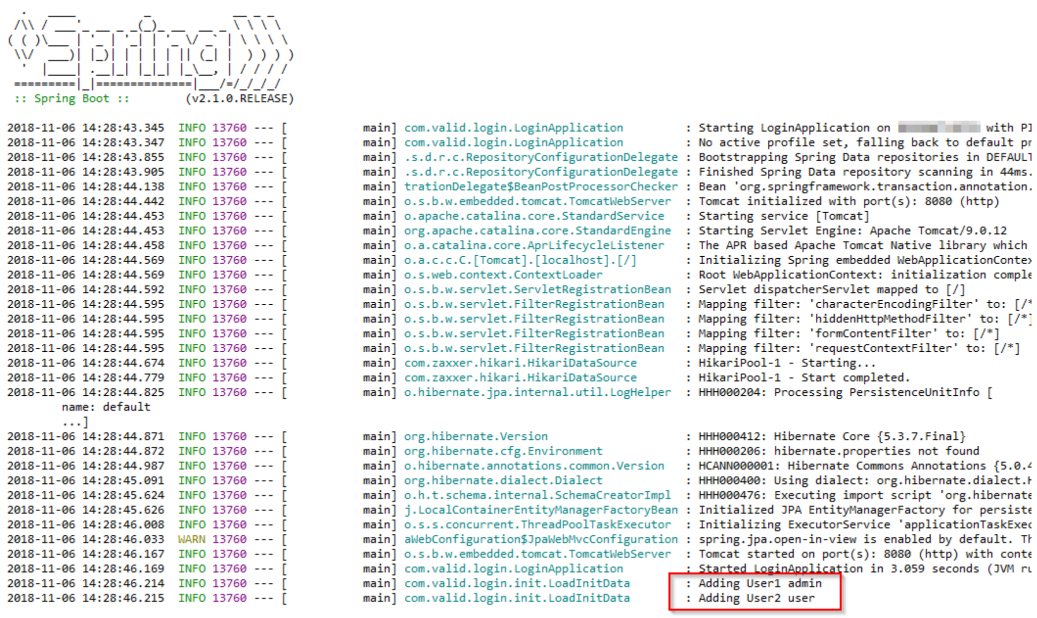


```
[INFO] ------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------
[INFO] Total time: 0.912 s
```

Finally, you can run the server from the IDE with the option “Run As > Spring Boot App”



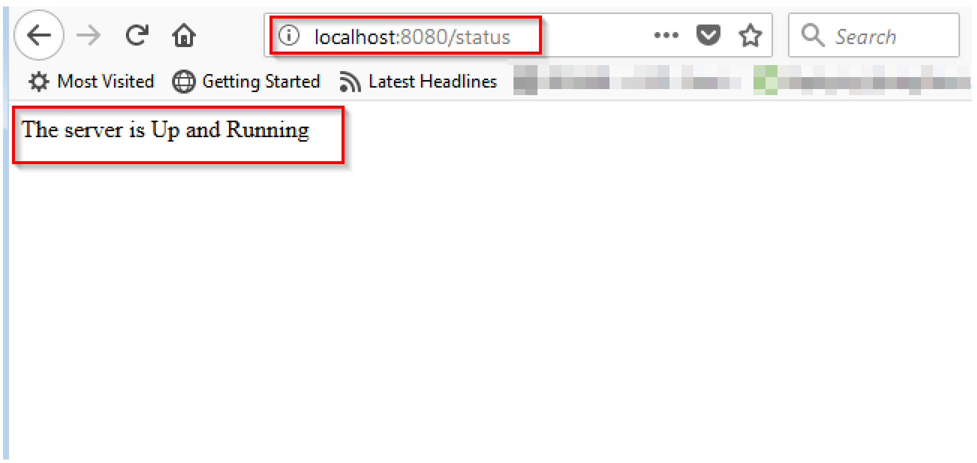
```
[INFO] ------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------
[INFO] Total time: 12.869 s
```



```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.1.0.RELEASE)

2018-11-06 14:28:43.345  INFO 13760 --- [           main] com.valid.login.LoginApplication         : Starting LoginApplication on [redacted] with PI
2018-11-06 14:28:43.347  INFO 13760 --- [           main] com.valid.login.LoginApplication         : No active profile set, falling back to default pr
2018-11-06 14:28:43.855  INFO 13760 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data repositories in DEFAULT
2018-11-06 14:28:43.905  INFO 13760 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 44ms.
2018-11-06 14:28:44.138  INFO 13760 --- [           main] trationDelegate$BeanPostProcessorChecker : Bean 'org.springframework.transaction.annotation.
2018-11-06 14:28:44.442  INFO 13760 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2018-11-06 14:28:44.453  INFO 13760 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2018-11-06 14:28:44.453  INFO 13760 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet Engine: Apache Tomcat/9.0.12
2018-11-06 14:28:44.458  INFO 13760 --- [           main] o.a.catalina.core.AprLifecycleListener   : The APR based Apache Tomcat Native library which
2018-11-06 14:28:44.569  INFO 13760 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationConte
2018-11-06 14:28:44.569  INFO 13760 --- [           main] o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization comple
2018-11-06 14:28:44.592  INFO 13760 --- [           main] o.s.b.w.servlet.ServletRegistrationBean  : Servlet dispatcherServlet mapped to [/]
2018-11-06 14:28:44.595  INFO 13760 --- [           main] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'characterEncodingFilter' to: [/*
2018-11-06 14:28:44.595  INFO 13760 --- [           main] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'hiddenHttpMethodFilter' to: [/*]
2018-11-06 14:28:44.595  INFO 13760 --- [           main] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'formContentFilter' to: [/*]
2018-11-06 14:28:44.595  INFO 13760 --- [           main] o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'requestContextFilter' to: [/*]
2018-11-06 14:28:44.674  INFO 13760 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2018-11-06 14:28:44.779  INFO 13760 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2018-11-06 14:28:44.825  INFO 13760 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [
        name: default
        ...]
2018-11-06 14:28:44.871  INFO 13760 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate Core {5.3.7.Final}
2018-11-06 14:28:44.872  INFO 13760 --- [           main] org.hibernate.cfg.Environment            : HHH000206: hibernate.properties not found
2018-11-06 14:28:44.987  INFO 13760 --- [           main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.0.4
2018-11-06 14:28:45.091  INFO 13760 --- [           main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.H
2018-11-06 14:28:45.624  INFO 13760 --- [           main] o.h.t.schema.internal.SchemaCreatorImpl  : HHH000476: Executing import script 'org.hibernate
2018-11-06 14:28:45.626  INFO 13760 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persiste
2018-11-06 14:28:46.008  INFO 13760 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExec
2018-11-06 14:28:46.033  WARN 13760 --- [           main] aWebConfiguration$JpaWebMvcConfiguration : spring.jpa.open-in-view is enabled by default. Th
2018-11-06 14:28:46.167  INFO 13760 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with conte
2018-11-06 14:28:46.169  INFO 13760 --- [           main] com.valid.login.LoginApplication         : Started LoginApplication in 3.059 seconds (JVM ru
2018-11-06 14:28:46.214  INFO 13760 --- [           main] com.valid.login.init.LoadInitData        : Adding User1 admin
2018-11-06 14:28:46.215  INFO 13760 --- [           main] com.valid.login.init.LoadInitData        : Adding User2 user
```

The server can be tested on the url: http://localhost:8080/status



The server is Up and Running

Server endpoints

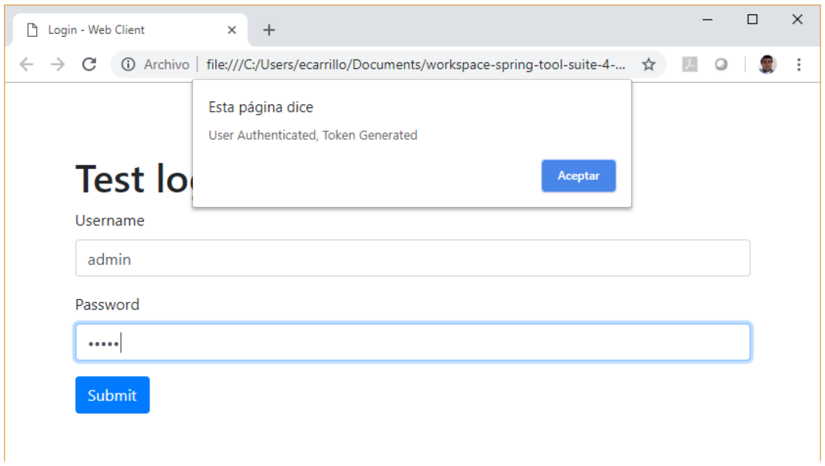
Endpoint	Http Method	Header
/status	GET (Without Authentication)	
/login	POST (If Authentication is ok returns token)	
/users	GET (Need token header to be called)	Authorization:token

Run Web Client

To run the eb client just download from the repository the compressed file web-client.zip from the url: https://github.com/ecarrilloq/login-test/blob/master/artifacts/web-client.zip

Extract the files and open in a browser the file index.html

The page is a login form with fields username and password, is the server is up and the values are ok, the page responds with an alert telling us that the user is authenticated, and the token was generated.



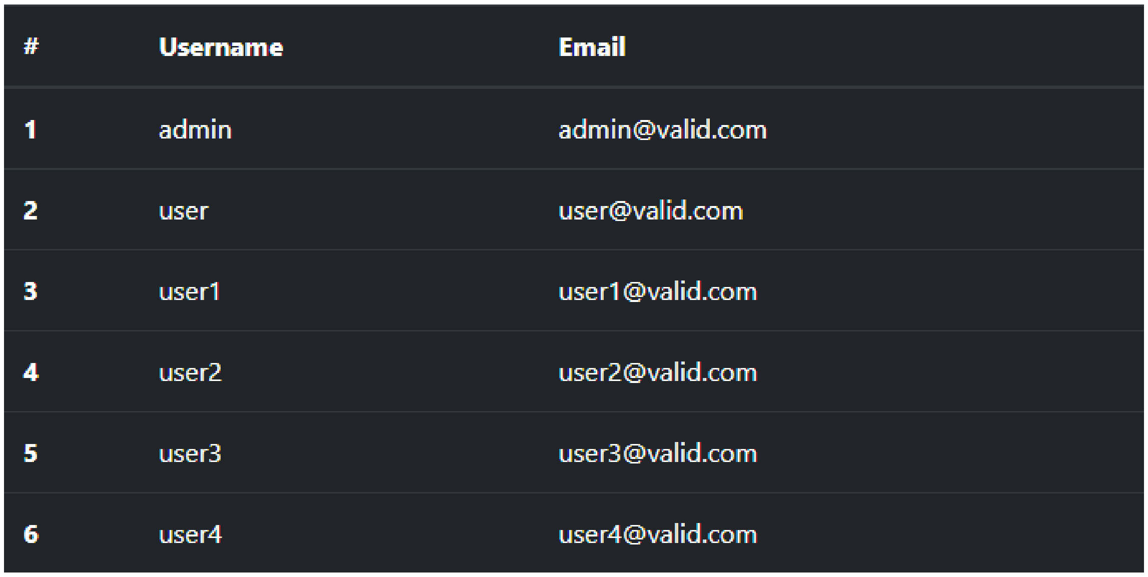
Login - Web Client

Archivo | file:///C:/Users/ecarrillo/Documents/workspace-spring-tool-suite-4-...

Esta página dice

User Authenticated, Token Generated

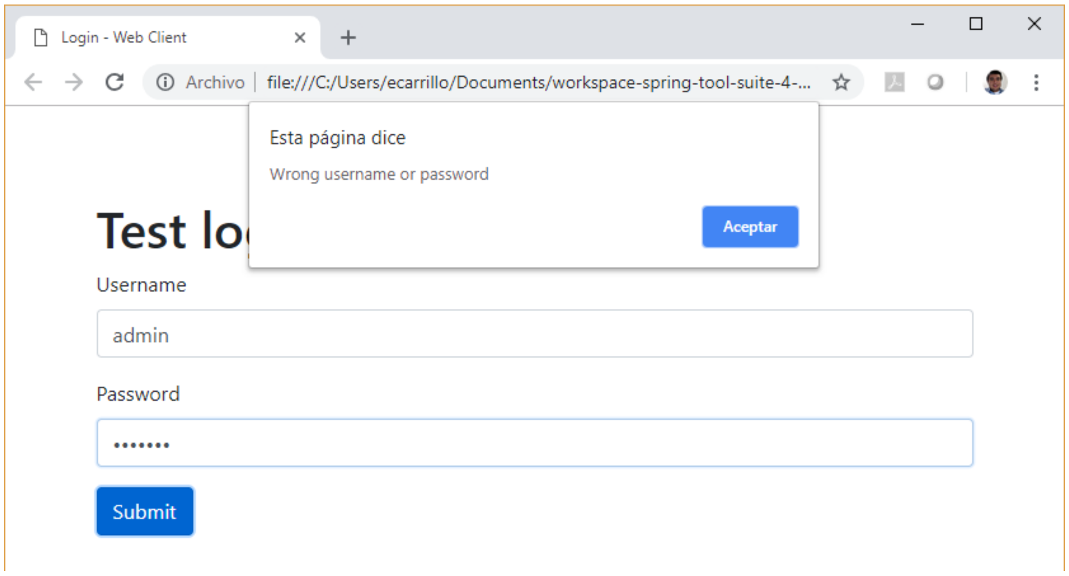
After press the accept button the page shows us the list of the initial users created on the database when the server is started.



#	Username	Email
1	admin	admin@valid.com
2	user	user@valid.com
3	user1	user1@valid.com
4	user2	user2@valid.com
5	user3	user3@valid.com
6	user4	user4@valid.com

For al the users, the password is equal to his username.

If the authentication parameters are wrong, the page shows us an alert with an error message.


Login - Web Client

Archivo | file:///C:/Users/ecarrillo/Documents/workspace-spring-tool-suite-4-...

Esta página dice

Wrong username or password

Test lo

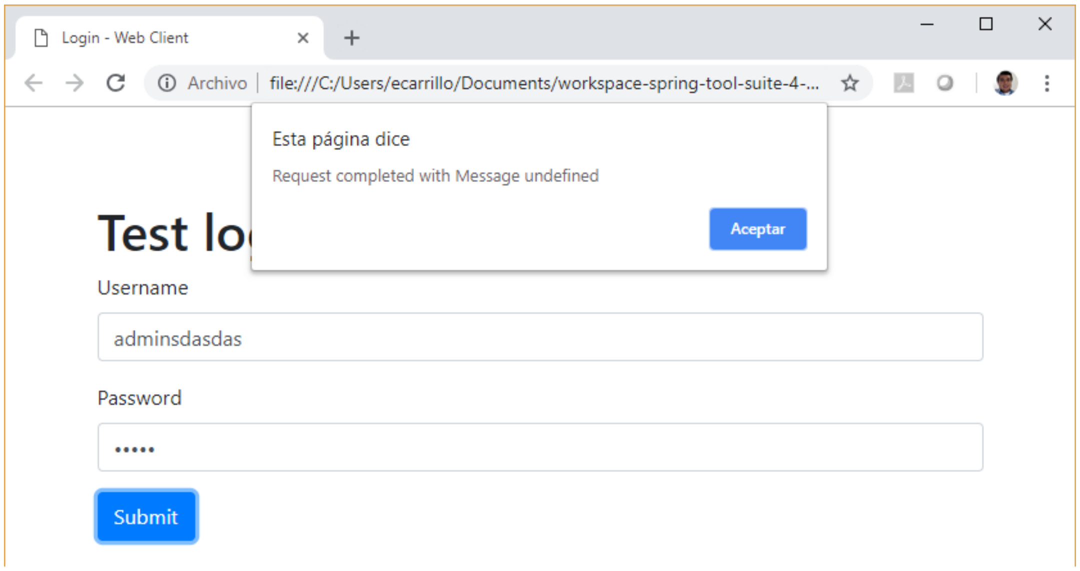
Username

admin

Password

•••••••

If the server is down the page shows us an error message:


Login - Web Client

Archivo | file:///C:/Users/ecarrillo/Documents/workspace-spring-tool-suite-4-...

Esta página dice

Request completed with Message undefined

Test lo

Username

adminsdasdas

Password

•••••

Project key aspects

Software Design Patterns and Good Practices:

The service is developed using an architecture with only two layers MODEL and CONTROLLER, on this case the service answer is wrote directly on the http response and not on a view technology.

The back end uses dependency injection to avoid issues when a dependency needs to be instantiated by another class.

The back application was developed thinking on a high cohesion and low coupling.

The classes and methods where developed thinking on single responsibility.

In the development process I use the Sonarlint plugin to avoid code smells.

The web client calls the back end using AJAX requests, those are asynchronous calls than don't freeze the browser.

With the usage of "Spring Data" the persistence to the database was simplified removing the DAO implementations I only need to extend a repository interface to have all the most common used crud methods.

I used a Business Service Facade to mask the complexity of another operations.

The back-end application doesn't write on the system out it uses a logger.

Security:

The token is generated using JWT one of the most currently used standards to secure rest services.

The /users service use the token for authentication on the server.

The server allows cross origin request from any origin for the demo purpose, on a production environment it could be limited just to some clients.

The login call is made using the Post method to hide the parameters from the URL.

For time, the username and password are not encoded, on a production environment the best is secure the communication over https with a protocol equals or superior to TLS 1.2

The password is stored on the database using a hash, for time I just used a SHA-512 provided by java native, but it could be improved using other libraries as spring security an generating the hash with salt.