Environment and installation

This is just simple app for storing information about specific book and reads information about the book. **Java 8** is used for service, **postgresql** for DB and build tool **Maven**. Application is tested with **appache maven** server.

First is needed to checkout project from github **git clone https://github.com/boksha85/library.git**

and run **mvn clean install** to install dependencies.

Next we need to create database with name **library** for user/password boksha/boksha.
If different names are used for db and user/password then its needed updating **config.yml** file with those values.

After setting database we need to create table Book and add 10 records. For this is used **dropwizard-migration** feature. **migration.xml** contains infomration how table should look like, constraints for table and records that should be stored.
For creating table and storing records we need to run this command:

java -jar target/library-0.0.1-SNAPSHOT.jar db migrate config.yml

After this we can start our application with

java -jar target/library-0.0.1-SNAPSHOT.jar server config.yml

In browser we should run

localhost:8080

and we will be routed to **Home page**, which on load will call ***/api/books/lastfive*** to retrieve last five books stored

Front part is done in using **jqury, ajax, html and css**.

From header we can redirect to **Show all books**, which will show as all books from database

Then we have **Get Book**, where we can search book by title.

And **Add new book** from where we can store new book, if book si stored successfully we will be redirect to **Home page.**

APIs, Validations and Testing

4 APIs are exposed:

GET /api/*books* – *returns all books from database*

GET /api/books/lastfive – *returns last five inserted records. This is used for home page*

GET /api/books/get/{title} – *returns all books that contains search parameter sent as title*

POST /api/books/add/ *- store new record in database, in body we should send Book as json object.*

In project root folder in “postman” folder we have postman collections which can be imported in postamn app for testing APIs.

Validations are done in front and back. Advice **“Never trust the client”** :)

All validations are covered with **Junit** tests.