**High-level documentation of our project:**

We used the basic architecture provided by play framework. We have a AKKA server managing client’s request, and slick manages the interaction with the database.

We defined the server to work with 20 threads.

When we get an HTTP request, the “routes” file provides the right function to deal with it, the corresponding functions are all in the toDoListController.scala file.

There- we process the request, handle the data base using slick queries, and return the appropriate response.

Diagram

Description automatically generated

**Dealing with errors:**

When using scala, it is very natural to use pattern matching to deal with input. This way we can send the right response in case the request was handled properly, and return “bad request” or another error in cases where we faced problems.

In order to minimize errors while querying the database, slick provides as with a mechanism of running a sequence of data base queries as a transaction, which means the data base is being manipulated only if **all** of the queries succeed.

Slick gives us the option to define default action on two tables entangled with a foreign key.

So when we delete a person, the tasks associated with it get deleted automatically, since we set the tasks’ ownerId as a foreign key of the primary key id of a person.

**How to run the system:**

Running the system is simple and consists of a few steps only!

1. Make sure you hav e JDK 8 or above, if not install from

<https://www.oracle.com/java/technologies/downloads/>

1. Install the sbt command line tool - <https://www.scala-sbt.org/>
2. Copy / download the contents of this git repository into a folder.
3. Open terminal, go to the directory of the folder from step 3 and run the command –

sbt run

1. Send HTTP requests to your heart’s content! (e.g. curl localhost:9000/api/people)