P1 Considerations

Webapp

The webapp should listen for incoming HTTP requests and invoke the appropriate service layer methods to interact with the ORM

The webapp should prepare the connection information in order to invoke the ORM connection logic.

The webapp should process the incoming HTTP requests and prepare a response to be sent.

The webapp should be in charge of basically all other logic and workflow of the project, anything not specifically manipulating the database.

ORM

The ORM should create and maintain the connection to the database. But, the ORM should make this connection based on information dependency injected into a parameter list of a method, like:

connect(String connectionString)

or:

Connect(String host, String port, String dbName, String username, String password)

The ORM should create SQL statements to parameterize.

The ORM should parameterize and execute SQL scripts to facilitate CRUD.

P1 Considerations

Where does Postman fit into all this?

Postman will simulate end users using some client that consumes our API.

Because we are eschewing a UI to interact with the user, we need some way to interact with our app. Instead of using a console or web browser or something, we will just pretend that our webapp is simply a service and the client is written by some other group. So, instead of having a client with which to enter info and see feedback, we just send along HTTP requests with postman, and receive responses.

We can make these up, it's all pretend. Let's make up enough requests to showcase our apps functionality.

How do we put our two projects together?

We can take our PORM and package it up like any other maven dependency. We won't publish it online, but we will load it into our maven cache locally. Maven checks the cache for dependencies first, and if it cannot find what it wants, it looks online and downloads it to the cache. So we just put our ORM in that cache.

This is as simple as using the maven install goal.

We type: mvn install

This packages our project and loads it into the cache depending on the project parameters GrpoupId, ArtifactId, version.

Now we just need to include that same info in our webapp as a dependency.