Backend Code Review

Basic Code Review Checklist

Am I able to **understand** the code easily?

The php is pretty straightforward since the error handling system was implemented(try, catch, exceptions as well as response codes), and there would be a message or error message indicated for every code , although there are no code comments so if the reader is not familiar with certain methods then it would be hard to understand.

Is the code written following the **coding standards/guidelines**?

There are three different php versions used, one is For CRUD purposes, it uses Php PDO coding standards , other one is LDAP module for login and the last one does not follow any coding standards.

Is the same code **duplicated** more than twice?

Yes for the normal version of Php since it was the first one implemented, the goal was only to get it working, some parts of it were later converted to Php PDO versions although it is not yet complete.

Can I **unit test / debug** the code easily to find the root cause?

Debug codes are still present in the normal version e.g. print_r($x);

Also used php unit testing.

Is this function or class **too big**? If yes, is the function or class having too many responsibilities?

Yes the class User is responsible for logging in, signing up, creating and listing all user project.

Detailed Code Review Checklist

Is the code scalable to support a huge number of users?

Using local storage to improve scalability

Is Security taken care?

Html special chars are used to avoid mysql injection.

For jwt authentication a secret key was implemented as well it is using HS256 algorithm

Is performance acceptable with huge data?

Is the code meeting non functional requirements?

Is code follows defined architecture

Is the code maintainable easily?

Single responsibility principle

Open closed principle

Is the code following ooad?

Dependency injection

PDO is meant to separate database and classes therefore classes are not dependent of database

Is the static code analysis metrics applicable?