**Capstone Project – Notes and References**

**Always refer to Microsoft Teams for the latest version of the document.**

**Points to be noted**

1. Implement the minimum basic features as mentioned in the requirements document and have a working application. After this, you can progressively start adding additional features and have a working application at every stage.
2. To make the user interface better, you should use **Bootstrap or Material**. Spend some time to think and make the UI intuitive and attractive.
3. Appropriate input fields should be used to make it user friendly. For example, where date input is required, you should use a date field and not a string field.
4. Validations should be done, wherever applicable, and make it user friendly.
5. It is mandatory to use Test Frameworks. The test cases developed to test the client side as well as server side code should also be checked into Gitlab repository during final submission.
6. **Initial data displayed when the application comes up for the first time should look like real meaningful data.** Don’t have data like “Test123”, “Dummy123” etc.

**Below are some references for implementing authentication, charts and for test frameworks.** You could go through these references to get an idea about how to implement these features in your capstone project.

**Authentication**

1. **Passport** - Express compatible authentication middleware for Node.js

[**http://www.passportjs.org/**](http://www.passportjs.org/)

There are around **300+ strategies**. You can read about **passport-local** (or) **passport-jwt** for user password authentication **AND** **passport-twitter** (or) **passport-linkedin** for OAuth authentication.

1. **JSON Web Token (JWT) -** Read about JWT in the below link. On successful authentication (in case of user password or OAuth authentication), you should return JWT and use it as identification token in further requests of the application.

[**https://jwt.io/introduction/**](https://jwt.io/introduction/)

1. Username, password authentication (with JWT)

[**https://stackoverflow.com/questions/20228572/passport-local-with-node-jwt-simple**](https://stackoverflow.com/questions/20228572/passport-local-with-node-jwt-simple)

**(This link explains about authenticating using passport-local strategy and on success, returning a JWT).**

**(OR)**

[**https://jonathanmh.com/express-passport-json-web-token-jwt-authentication-beginners/**](https://jonathanmh.com/express-passport-json-web-token-jwt-authentication-beginners/)

**(This link explains about authenticating using passport-jwt strategy)**

**Note:** Use **bcrypt** npm module for encrypting password and storing in DB (Do not store in plain text form).

1. OAuth Authentication - Twitter, Linkedin

Refer to **passport-twitter** and **passport-linkedin** strategies in passportjs.org and also go through the below link to get an idea how to implement.

[**http://mherman.org/blog/2015/09/26/social-authentication-in-node-dot-js-with-passport/**](http://mherman.org/blog/2015/09/26/social-authentication-in-node-dot-js-with-passport/)

**On successful OAuth authentication, you can return a JWT (JSON Web Token).**

**Charts**

1. Chart.js - <http://www.chartjs.org/docs/>

**Test Frameworks**

1. REST API - Testing with Mocha, Chai, Supertest

[**http://developmentnow.com/2015/02/05/make-your-node-js-api-bulletproof-how-to-test-with-mocha-chai-and-supertest/**](http://developmentnow.com/2015/02/05/make-your-node-js-api-bulletproof-how-to-test-with-mocha-chai-and-supertest/)

1. Angular testing

**https://angular.io/guide/testing**

1. React testing

**https://facebook.github.io/jest/docs/en/tutorial-react.html**