Database Connectivity Individual Project:

Part 1: In-Class Tasks

You must complete these three tasks **IN-CLASS** on the specified day. The professor will release a document containing the task during the lecture. If you are not in class, you will get 0 for that task. There is no drop-box for the tasks, so you must show the professor your work before you leave.

Task 10: Creating an Angular Application – Due: ***1 mark***

Task 11: Angular Components and Routing – Due: ***1 mark***

Task 12: Forms and Angular Routing with Parameters – Due: ***1 mark***

Part 2: Main Component

Set-up:

1. Continue working on the project you created for the previous assignments
2. Create a new folder called APP_PUBLIC in the root of your project
3. Set up a new Angular project inside this folder

Service and Object:

1. Generate a new TypeScript class based on your main model class. Yours might be "Book" or "Movie". It should have the same fields as your Mongoose model (including an _id). ***2 marks***
2. Generate a service which allows you to access your API. It should include the ability to get a single item, get the list of items, and create a new item. ***2 marks***
3. Add delete and update functionality to the service. ***1 mark***

Components:

1. Generate components designed to make reusable blocks of code for your front-end. Chapter 9 of the textbook is very useful here (page 308). Your goal should be to think of what parts of your page can be separated into components. A good starting point is the framework, the block for each page of content (Home, About, etc), and a header. ***1 mark***
2. Set up your components so the user can view a list of every object in the database, as well as see the details of a particular entry. The user should also be able to create new entries with a form. ***2 marks***
3. On the details page, allow the user to update or delete the currently selected object. For example, we might be on the page for the book with id 1. The user should be able to update and delete it from this page. ***1 mark***

Angular Routes and SPA implementation:

1. Set up Angular routing and create all the basic routes your site needs. You should have a home page route for "", a route for "create", and a route for the individual entries. You may need more routes depending on your page structure. ***2 marks***
2. Modify the project so it does not run through Pug anymore. You will edit app.js to reference the "build" folder and make a few other modifications like we did for the in-class task. ***1 mark***

Design:

Here you will be graded on a number of factors:

1. The uniqueness of your site compared to what we've done in-class, as well as what other students have done.
2. The code should have comments and be easy to read.
3. The site should be well designed. It should look pleasing to the eye and use appropriate colours and contrast.
4. The site runs without any issues.

5 marks

TOTAL: 20 marks