Coursework

# Introduction

In this task you will build an application that makes use of modern application development frameworks and development techniques. You will demonstrate your understanding of the technical opportunities provided by such frameworks.

This application should be built to meet the needs of the clients in the case study that is available on

Blackboard at ​<https://tinyurl.com/y9awdc6p>

# The task

A system for the storage, cataloguing and retrieval of ​documents, media assets and other files​.

This application is a prototype and proof-of-concept for Morten Skjeggestad. Morten wants a custom application that allows his creative staff to store and share multimedia assets on a cloud repository. His staff need a user-friendly application that lets them upload and store different types of media files. They need to be able to attach metadata such as title, author, date and a keywords or tags to each item as it is uploaded. Users should be able to search for items using the metadata fields. The result of a search should be to display a list of files and their associated metadata.

You must build a system that TMS can use to manage their digital assets. You do not need to handle the files themselves because they would normally be held on a dedicated media server. Your application must

* Handle the metadata that is used to describe each file.
* Manage multiple versions for each file. As files are edited and used new versions will be created, for example images at different resolutions. Users must be able to track these versions and your tests should show how new ones are created.
* Allow for the check-in and check-out of files by users so that different people are not editing the same file concurrently.
* Be usable by project teams split across locations.
* Manage user authentication and sessions. You do not need to have a registration form. Assume that user registration is done elsewhere and you are simply checking credentials against records held in a database.

You might think of other features that you wish to add.

In this task we require that you use JavaScript[[1]](#footnote-1) to demonstrate the following aspects of the application:

1. An application that runs on the node.js server.
2. A data access layer that wraps a MongoDB database, presenting records and tuples as JavaScript objects in the form of JSON documents. This must implement a full set of CRUD operations.
3. A middleware layer that accepts HTTP requests (at least GET and POST), validates those requests and routes them to the appropriate URL endpoints.
4. Implement views using a framework such as Vue, React or Angular.

# Learning outcomes

By engaging successfully with this module a student will be able to:

1. Critically assess the architectures of popular frameworks and the role which such frameworks have in modern system development.
2. Implement software using a variety of frameworks.
3. Examine comparatively systems architectures and demonstrate appropriate uses for them.
4. Compare and contrast the frameworks used during the module.

# Demonstrating your work

Your work will be marked either at walkthrough or via a screencast video. ​~~Because there are so many~~ ~~students taking the module the video is the preferred option. Please talk to us if you wish to do a~~ ~~walkthrough.~~ ​A Doodle poll will be made available before the submission date on which you should register if you want to have a walkthrough.

Alternatively you​ ~~You​ ​must~~​ ​may submit a video presentation in which you demonstrate the​ functionality of your application and describe its implementation. The video and walkthrough will both follow the script that is given at the end of this assignment specification.

The video may be via desktop capture such as Open Broadcaster (​https://obsproject.com/)​ or you may use a camera/phone to capture both you and your screen. In either case it is important that the code and output are both clearly readable by a viewer. The video must be in a platform independent format such as MP4 and must be no longer than 15 minutes.

# Submission

The deadline for this task is ​**3 p.m. on Thursday 21**​**st**​ **February, 2019**​.

* You must submit an archive of your code to Blackboard.
* Do not include the contents of the ​node\_modules folder or the contents of your database.​
* You must include a readme file with a link to your video and instructions for building and running your application. Your readme must be plain text or simply formatted using markdown[[2]](#footnote-2)
* You must include instructions or a script for rebuilding and repopulating the database.
* Include any unit, integration or UI tests that you developed.

# Marking Scheme

We will be using a grade-based system rather than a numeric one. You can find details of this approach on the module’s Blackboard site and at ​<http://tinyurl.com/zkej95m>​.

You will be assessed on:

* The quality and completeness of your application.
* The quality of your JavaScript coding.
* Use of best-practice architecture and implementation approaches.
* Your use of unit tests.
* Your understanding of your own code as demonstrated at a walkthrough.

In the following table we give an incomplete list of the types of thing that we want you to demonstrate in your video or walkthrough.

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Marks**  **available** | **We are looking for** |
| Architecture of the framework | 10 | You use MVC or a variant thereof  Architectural layers are clearly defined and separate  Code is in appropriate places |
| Routing and the external interface of the middleware | 10 | URLs are RESTful  Routes are clear and can be deduced once the basic structure is understood  The set of routes is coherent  HTTP verbs are used appropriately |
| Implementation of the middleware layer | 5 | The logic of the application is clearly separate from model and view You use JavaScript modules and classes to enforce separation of concerns |
| Completeness of the work | 5 | All major aspects of the program are completed The work may extend beyond the brief |
| User interface | 10 | Your stylesheets are reactive  You create a clean, simple and usable UI  You adhere to principles of good UI design |
| Examples that use the framework | 5 | URLs are demonstrated in Postman  The examples are chosen well to demonstrate the range of of the implementation |
| Database Access Layer and database structure | 10 | The database holds documents that have a sensible structure  Document structures are mapped onto JavaScript objects  All CRUD operations are performed on all types  Deletion is handled sensibly |
| Document structures | 5 | You use JSON documents where you can Document structures relate to the domain |
| Handling of user accounts | 10 | ~~Accounts can be created, deleted etc.~~  Users are able to log on and off  Actions that users perform are related to account type and log on status  Sessions are used where necessary |
| Validation and verification | 5 | Are inputs verified  How are errors handled |
| Use of unit tests | 10 | The tests you have written  The framework or library you used |
| Quality of the JavaScript code | 5 | Naming conventions  Coding style and layout  Use of comments |
| Student’s  understanding of the code at walkthrough | 10 | Preparation  Quality of explanation  Level of detail  Appropriateness of the explanations and discussion, Your ability to respond to questions (at walkthrough), Your coverage of the work. |

# Walkthrough script

This script is to be followed in your walkthroughs with Chris.

1. Show us your code in action.
   1. Explain ​what​ it does but not how​ ​ it does those things.
   2. Be clear about the examples you have created.
2. Run your unit (or other) tests.
   1. What are you testing?
   2. What do the results mean?
   3. Why are you testing those things?
3. Talk us through the URLs which you implemented.
   1. Are they REST​*ish*​?
   2. How are the different HTTP verbs used?
   3. Show the document structure or data which is returned from each request?
   4. How do you parse requests?
   5. How are requests handed off to other code?
4. Show us the data structures you use.
   1. We are interested in your database tables.
   2. We are interested in your JSON documents.
   3. How is the database wrapped by the Data Access Layer?
   4. What does the DAL do?
   5. Can you return both single objects and sets of objects?
   6. Are your data structures and documents extendible?
5. Talk us through your architecture.
   1. How do you use callbacks and events?
   2. Is your interface clear and simple?
   3. Does your interface create a coherent API?
   4. How might your framework be extended?
   5. What design or architectural patterns have you used?

1. Using the application
   1. Clearly explain the user interface.
   2. How are user accounts managed?
   3. How do users authenticate on to the system?

1. You will also be assessed on
   1. Your management of the walkthrough.
   2. The quality of your code.
   3. The understanding that you demonstrate and your answers to questions.

1. TypeScript or CoffeeScript may also be used if they better fit your framework. [↑](#footnote-ref-1)
2. ​<https://daringfireball.net/projects/markdown/syntax> [↑](#footnote-ref-2)