Applications Architectures and Frameworks

Task one

# Background

JavaScript has become important not only as a language that is used in the development of Web applications, but as a platform for a range of developments across devices. The Node.js lightweight Web server is at the forefront of these changes. In this assignment you will design, build, test and demonstrate a JavaScript framework that runs within Node.js and which provides the infrastructure required to build targeted, domain-specific applications.

This task is worth 60% of the marks for this module.

# 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.

# Outline of the task

In this task we require that you use JavaScript to write an original framework which supports the following aspects of an application:

1. A data access layer that wraps a database server of your choice, presenting records and tuples as JavaScript objects in the form of JSON documents.
2. A middleware layer that accepts HTTP requests (at least GET and POST), validates those requests and routes them to the appropriate URL end-points.

## Notes

* You must build a test implementation that demonstrates the use of your framework~~.~~
* Your code **must** run on the Node.js server.

# The domain

The domain for this task is the ever-popular world of online cookery sites and recipe books. You must develop a framework that could be used by other developers to create mobile and Web application such as community-driven recipe books. You are to develop a simple application of your own but only to demonstrate the flexibility, functionality and completeness of your framework. You will not be assessed on the abstract quality of this sample application no on its UI.

Your framework must support:

* User registration and authentication.
* The editing of recipes. ( need to add ingredients editing)
* Validation of newly entered or edited recipes (so that quantities, for example, are validated in some way).
* A search function through which client applications can find recipes by name or ingredient (you may want to consider the difficulty users might have spelling aubergine). NEED TO SEARCH FOR INGREDIENTS THOUGH
* The storage of recipes, comments and user details in a NoSQL style database (preferably MongoDB).
* A tagging system for recipes.
* The inclusion of links to Web resources such as images, videos or blogs within recipes and comments.
* The addition of shopping lists to individual recipes.
* The addition of notes to recipes.

*Note: for this module, the validity of the demonstration, interactions are not directly assessed; the focus is on the validity of the framework. But beware that an invalid application is unlikely to demonstrate a valid framework.*

# 

# Submission details

You must upload a zip file of your original code to the assignment handler on Blackboard by **midday on Friday 19th February, 2016**. Do not upload the node server or any node libraries or frameworks you have used. Do not include databases in your uploaded file.

# Marking

To avoid clashes with the writing and submission of your final year project this work will be marked in the two weeks after submission. Your work will be marked by walkthrough and code review in the weeks starting 21st February, 2016 and 28th February, 2016.

# Marking scheme

You will be assessed on:

* The quality and completeness of your framework.
* 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.

|  |  |  |  |
| --- | --- | --- | --- |
| **Facet** | **Marks available** | **At pass level** | **At distinctive level** |
| Architecture of the framework | 10 | Code has some structure with some attempt to separate into modules based on functionality/use | Good use of idioms and patterns to structure the code |
| Clarity of the interface to the framework | 10 | Interface is clear and can be used sensibly | The interface works either as a DSL or as a clear API |
| Completeness of the framework | 25 | Important aspects of the framework are implemented but not all | All of the required functions are implemented successfully |
| Examples that use the framework | 15 | Few are provided and are of poor quality | A high-quality set of examples shows how to use the framework. |
| Quality of the JavaScript code | 10 | Code is readable with sensible variable names | The conventions of JavaScript are used correctly. Code is of professional quality. |
| Use of unit tests | 10 | Limited evidence of unit testing | Comprehensive unit testing is demonstrated |
| Student’s understanding of the code at walkthrough | 20 | Some understanding shown but detail is lacking. | Good explanations, the student is confident and in control of the walkthrough. |