BEng/MEng Degree Examinations 2013/14

DEPARTMENT OF COMPUTER SCIENCE

Interactive Application Programming Techniques (IAPT)

Open Individual Assessment

Issued: Wednesday 13 November, 2013 (Aut/7)
Submission due: 12 noon, Wednesday 4 December, 2013 (Aut/10)
Feedback and marks due: 27 January 2014 (Spr/4)

All students should prepare their assessment as per the details for web submission found at the following link: http://www.cs.york.ac.uk/support/pages/Website_Submission. Note this is a new submission system for the department. Do not use the standard online submission system that we use in the department for non-web assessments.

All programming files, database export files and accompanying documents for this assessment must be in that folder at **12 noon, Wednesday 4 December, 2013**. At that time, the permissions on the folder will be converted to be visible to the Lecturer for marking. Any changes made to files after that point will be noted as being late. If an assessment (or part of an assessment) is changed after the deadline that whole assessment will be marked initially as if it had been handed in on time, but the Board of Examiners will normally apply a lateness penalty to that whole assessment.

The feedback and marks date is guided by departmental policy but, in exceptional cases, there may be a delay. In these cases, all students expecting feedback will be emailed by the module owner with a revised feedback date. The date that students can expect to see their feedback is published on the module descriptor: http://www.cs.york.ac.uk/modules/

Your attention is drawn to the Guidelines on Mutual Assistance and Collaboration in the Departmental Statement on Assessment:
http://www.cs.york.ac.uk/student/assessment/policies/#AcademicMisconduct

Any queries on this assessment should be addressed to Dr. Christopher Power (christopher.power@york.ac.uk)

Answers that apply to all students will be posted on the IAPT webpage: http://www-module.cs.york.ac.uk/iapt/.

Rubric:

The following is the Individual Open Assessment for the module Interactive Application Programming Techniques (IAPT) 2013.

The assessment has two components:

Part A: Interactive Web Application (65 marks)

The student will produce an interactive web application meeting the requirements prescribed in the System Requirements Specification below.

Part B: Design Document (35 marks)

The student will provide a report on design decisions he/she made during the process of creating the interactive web application in Part A, specifically a set of models of the interactive application.

Submission Checklist:

- All source code for the components of your web application. (HTML/CSS/PHP/Javascript)
- An export of your MySQL database. (SQL)
- A PDF document with your design document

Part A: Interactive Web Application (65 marks)

You will implement a content management system (CMS) for an online magazine. You will produce a website for online magazine for a domain of media of your choosing (e.g. books, films, comics) to demonstrate the effectiveness of your CMS. The following is the System Requirements Specification (approximating the IEEE standard).

1 System Requirements Document

1.1 Scope

This document describes the infrastructure for an online magazine. Each online magazine will present recent and popular articles on its home page, and allow readers to: view individual articles, search for articles, comment on articles and like/dislike articles. Each magazine may be on a topic of the student's choosing.

This infrastructure for the magazine includes content management system that will allow writers to submit articles to be published. Editors will manage an iterative workflow of submission and quality checking in collaboration with the writers ensuring only high quality articles are published.

Each online magazine will have readership from a wide audience from around the world; however, it can be assumed that content will only be provided in English.

1.2 Definitions

- Article: A piece of content consisting of the following: one or more Writers, a title, a text body between 0 and 2000 words, publishing metadata, content metadata and a cover image.
- Article Status: An article can have 4 different article statuses: Submitted, Under Review, Awaiting changes, Published
- Column: A named set of articles on one particular topic.
- Column Article: An article that belongs to a column.
- Review: A review is an article that provides an evaluation of something within the domain of interest (e.g. films, books). Each review includes a rating of quality from 1 to 5.
- Metadata: This is data that describes other data.
- Publishing metadata: These are descriptors of the writer of the piece, the type of piece (column, article, review) and the date the piece was published.
- Content metadata: These are descriptors about the content of the article. These are specific to the domain being written about. They can be thought of as keywords that describe the piece. For one specific online magazine there should be a standard set of content related metadata that can be chosen.
- Cover image: All articles should have an image that can be used with the article. This image will be displayed in thumbnail form, and in full size.
- Writer: Someone who is approved to submit articles for publication
- Editor: Someone who checks the quality of articles and ultimately publishes them to the online magazine.
- Publisher: A person in charge of the online magazine. Someone who has all the powers of editors, but also has the authority to add new writers and editors.
- Reader: Someone who reads articles on the online magazine.
- Subscriber: A reader with permission to leave comments and like/dislike articles on the website.

2 Overall System Description

This section contains the description of system to be produced in terms of its functionality and the different user and system interfaces that are to be provided.

2.1 Product Perspective

The CMS to be produced in this assessment is only intended to support the publishing of an online magazine. It is not expected to be a general purpose CMS for publishing any website, such as is provided Drupal or Joomla. Content for any website managed by this CMS is limited to the types listed in the definitions above.

2.2 Product Functions

CMS must support a cycle of writing and editing, where a *writer* submits an article for publication and an *editor* reviews the article for quality. An editor can reject the article outright, return the article to the writer with comments on major changes that are required, or the editor can make minor changes and publish the article. A published article is immediately available to readers and subscribers through the online magazine.

A published article should be accompanied by metadata about who wrote the piece, the date of publication and the type of article it is, specifically an article, a column article, or a review. Further, the piece should have appropriate writer/editor chosen content metadata (e.g. if it is a review of a book, it might include the author of the book and the publisher).

The home page of the magazine will draw attention to new articles, popular articles, and specific content that editors choose to highlight. Subscribers will be able to read published material through the online magazine website, comment on material, and indicate if they like or dislike the article.

3 Specific Requirements

3.1 Component Requirements

1. All components must be bespoke implementations. No generators, model-driven drivers or other forms of automatic code generators may be used.
2. All components must use only the technologies specified in this section. No other APIs, frameworks or other pre-prepared programming may be used.
3. If you use an external sources to inform your work, for example a design pattern, please cite the source. Students should not copy templates from off the web verbatim. There must be substantial innovation in your own website code to be awarded full marks as per the discretion of the module leader.

Violation of these three requirements will cause the assessment to immediately incur a 40 mark penalty.

3.1.1 System Interfaces

4. The CMS must use a Model-View-Controller architecture.
5. Data for the CMS must be stored in a MySQL database.
6. Data in the MySQL database will be stored in 3rd-Normal Form.
7. The Model component must access the data using the PHP Data Object API (PDO).
8. The Model must be implemented using the Data Mapper pattern
9. The Controller and Model must be implemented using PHP5.

3.1.2 User Interfaces

10. The View must be implemented using a combination of PHP5, HTML5, CSS3 and Javascript.
11. The View may use the JQuery API and the JQuery plugin JQuery UI if you so choose.

3.2 Functional Requirements

3.2.1 Reader Requirements

12. Readers must be able to identify the top 5 most liked content pieces from the home page of the online magazine.
13. Readers must be able to identify the 5 newest content pieces from the home page of the online magazine for each type of: article, column article, review.
14. Readers must be able to identify the 5 most recently highlighted articles as chosen by Editors and Publishers.
15. Readers must be able to retrieve the most recent 10 articles with a particular piece of publishing metadata or content metadata.
16. Readers must be able to view and read a single article.
17. Readers should have a minimum of 20 articles to read on the website.

3.2.2 Subscriber Requirements

18. Subscribers have all requirements that Readers have, plus the following additional requirements.
19. Subscribers must be able to register with a unique user name to the online magazine.

20. Subscribers must be able to add a comment to the bottom of a article that can be publically viewed.
21. Subscriber comments should contain information about which subscriber submitted the comment.
22. Subscribers must be able to like/dislike content pieces.

3.2.3 Writer Requirements

23. Writers have all requirements that Subscribers have plus the following additional requirements.
24. Writers must be able to submit an article to the magazine for publication consisting of: title, text body, cover image, additional writers (other than themselves), publishing metadata, content metadata.
25. Writers must be able to choose what type of content piece they are submitting: article, column article, review.
26. Writers must be able to specify to what column a column article belongs.
27. Writers must be able to specify a review rating for a review.
28. Writers must be able to see a list of articles they have written with their current status.
29. Writers must be able to edit an article that currently has the status: Awaiting changes.
30. Writers must be able to read comments from editor(s) on all articles that Awaiting changes or Published.
31. Writers must be able to identify which editor(s) have provided comments on the articles throughout the publishing cycle.

3.2.4 Editor Requirements

32. Editors have all requirements that Writers have plus the following requirements.
33. Editors must be able to see all articles that currently have the status: Submitted.
34. Editors must be able to indicate they are reviewing an article by changing its status from Submitted to Under Review.
35. Editors must be able to edit articles and save their changes when an article is Under Review.
36. Editors must be able to append (not overwrite) comments on articles when an article is Under Review.
37. Editors must be able to publish an article.
38. Editors must be able to reject an article.
39. Editors must be able to return an article with comments to Writers.
40. Editors must be able to see a list of all articles that they have been involved in editing.
41. Editors must be able to see what articles are marked as highlighted on the front page.
42. Editors must be able to indicate that an article should be highlighted on the front page.

3.2.5 Publisher Requirements

43. Publishers have all requirements that Editors have plus the following requirements.
44. Publishers must be able to see a list of all Writers.
45. Publishers must be able to see a list of Editors.
46. Publishers must be able to assign Subscribers to be Writers or Editors.

3.3 Non-Functional Requirements

3.3.1 Usability Requirements

47. All pages must allow users to succeed in tasks nearly 100% of the time.
48. All pages should provide access to articles, columns and reviews through a navigation bar.
49. All pages that display an article should provide access to publisher metadata and content metadata for initiating search on that metadata.
50. All pages should apply the usability guidelines found at usability.gov, paying particular attention to chapters 5 through 14.
51. A user guide that describes the functionality of each user group will be provided.

3.3.2 Maintainability Requirements

52. All methods and functions should be documented with a statement as to what the method does, the input parameters and return values.
53. All PHP methods and functions should be documented using PHPDocumentor annotations.

54. All code should have internal documentation sufficient to understand what the functionality does.

3.3.3 Code Organisation Requirements

55. All model code should be contained in a subfolder called *models* in the root folder.
56. All controller code should be contained in a subfolder called *control* in the root folder.
57. All view code should be contained in a subfolder called *view* in the root folder, with the following two exceptions.
58. All CSS files should be in a subfolder called *css* in the root folder.
59. The homepage of the application should be located in the root folder of the website with the name: index.php

Part B: Design Document (35 marks)

Each student will provide a design document that will consist of a *maximum* of 10 pages. This document should contain the following sections.

1 Interactive Application Modelling Diagrams

For each model, students should provide a text justification for key design decisions made in producing the interactive system. This does not mean that students should explain the entire diagram. Students should concentrate on describing parts of the diagrams that highlight key aspects requested in the assessment description below.

1.1 Data and Mapping Models (10 marks)

Students will provide a UML class diagram describing the data organisation of their application and the mapping of that organisation into their application. In the justification provided for this model, students should discuss aspects of the model that support cohesion and decoupling in their code and how it meets the data requirements specified above.

1.2 Process Model (5 marks)

Students will provide a Business Process Modelling and Notation (BPMN) diagram describing the process of moving an article from initial writing through to publication in the online magazine. In the justification provided for this model, students should provide information regarding how the requirements of the process are met the specified process.

1.3 Interaction Flow Model (10 marks)

Students will provide interaction flow models describing the interfaces for:

- A writer submitting an article for publication.
- An editor undertaking a quality review of an article.

1.4 Usability Justification (10 marks)

Each student will choose a key and substantial piece of functionality from their interactive system. For that piece of functionality the students will justify, in no more than 500 words, the design choices they made to support usability of the interactive system.

Marking Criteria

Part A: Interactive Web Application (65 marks)

<table>
<tr><td>Subscriber/Reader Requirements</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for meeting all requirements specified for subscribers and readers. The majority of the marks will be allocated for readers being able to access content as described in the requirements brief, with subscriber functionality accruing smaller amounts of marks.</td></tr>
<tr><td>Writing/Publish Requirements</td><td>Marks: 0-25</td></tr>
<tr><td colspan="2">Marks will be awarded for meeting all requirements specified for Writers, Editors and Publishers.</td></tr>
<tr><td>Model Component Requirements</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for model components being implemented correctly including appropriate use of the Data Mapper pattern and implementation of the normal form.</td></tr>
<tr><td>Usability Requirements</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for students applying usability guidelines as discussed in class and as documented on usability.gov. Students should justify their design decisions in their documentation of the code for the view. This is a holistic mark, which will be awarded on the basis of the expert judgement of the Lecturer.</td></tr>
<tr><td>Maintainability Requirements</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for appropriate documentation is provided both for methods headers and internal code documentation. Marks will also be awarded for cohesion and decoupling of code, and the correct application of the MVC architecture.</td></tr>
</table>

Part B: Design Document (35 marks)

<table>
<tr><td>Data Models</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for providing a model that accurately describes the database provided in the project. This model must not be generated by an automated tool. Models that do not meet the data requirements, in normal form or content of the project will not be awarded full marks.</td></tr>
<tr><td>Process Model</td><td>Marks: 0-5</td></tr>
<tr><td colspan="2">Marks will be awarded for a model that appropriately describes the process of publishing an article to a sufficient level of detail that someone could describe the requirements of the process</td></tr>
<tr><td>Interaction Flow Model</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for the completeness of the representation of the interface provided in the project and appropriate use of modularity in the diagrams.</td></tr>
<tr><td>Usability Justification</td><td>Marks: 0-10</td></tr>
<tr><td colspan="2">Marks will be awarded for the adequacy of the justification of how usability principles were taken into account in the interface paying particular attention to topics discussed in lecture.</td></tr>
</table>