SE31520/CHM5820 Assessed Assignment 2014-15

Enhancing the CS-Alumni Application

Chris Loftus
Hand-out date: Tuesday 28th October 2014
Hand-in date: Monday 8th December 2014
50% of SE31520/CHM5820 assessment

The problem

During the course of the module we have been looking at building the CSA Ruby-on-Rails application. In this assignment you will take my most recent version of the prototype and enhance its functionality or add scenario based test support (described below). I want to give you the freedom to be creative and decide on possible enhancements. That said, I have provided a list of possible enhancements to choose from. If you decide to do something different, then check with me first. I want to make sure it is sufficiently challenging.

This assignment has been given out very early to give you time to think and develop the code incrementally. Some relevant lectures and worksheets will not be covered until several weeks into the project. In particular, lectures on Rails support for authentication and Ajax will not be dealt with until November (see the Blackboard schedule). This may influence your choice of enhancements.

If you don't use Cucumber and RSpec testing then you must provide Rails unit and functional tests. The current prototype's testing support is very weak and needs fixing and enhancing.

I also want you to analyse the existing prototype and provide a design section in your report that describes the final design. This must cover both my code and your extensions. Make it clear which is which.

You are allowed to refactor my code. However your report must justify the refactorings. However, please do not completely re-implement everything!

Please try to respect the REST way of doing things when adding new functionality.

This assignment should take in the order of 50 hours, however the exact figure depends on whether you undertook sufficient preparatory work and study (worksheets, reading and attending lectures) and your software development abilities.

When testing the application, be careful that you don't swamp our email servers with emails. They will be treated as spam and you will be prevented from emailing for a half an hour or so. Instead, record emails in the development log file and just do a small test to make sure emails really work. I have commented out the configuration code that allows emailing. Also, make sure that if you do send test emails then send them to yourself and not to me!

Possible enhancements

Here are a few ideas. Choose just one. Some are easier than others. The easier ones will need to be done really well to get good marks. However, don't feel constrained by these suggestions, but you must contact me if you want to do something different.

- Explore and add Cucumber and RSpec test suites. If you take this approach I expect to see a thorough set of tests.
- Develop a REST-based web-service client program that communicates with the CSA application, using JSON as the data representation. The client may be written in any language you wish. CSA's authentication support may make this more difficult than you might initially think. You might consider commenting out the authentication/authorisation code during initial development.
- Change the broadcast function to broadcast a URL to a weblog entry rather than the text itself. The HTML text should be saved then in a place pointed to by the URL.
- Implement the Facebook feed.
- Implement the RSS feed.
- Improve authorisation to enable multiple logins to be considered as administrators.
- Make the site responsive (Responsive Web Design principles) and support mobile platforms.

Steps

Undertake the following steps:

- Download the current prototype and get it to run. It uses the gems installed during the worksheets (apart from the OAuth gem used for Twitter interaction). These are declared in the *Gemfile* file.
- Analyse the existing application. You will find existing design documentation in my Requirements slides (on Blackboard).
- Design and implement your enhancement, or implement Cucumber and RSpec testing. If necessary, speak to me if you're not sure. You may also refactor my existing code.
- For those not doing Cucumber Testing, make sure you have implemented some examples of unit and functional tests that test my code and your additions. Just provide a few examples for each model class and controller class. Fix any of my tests that are currently broken.
- Write a 2000 to 3000 word report:
    - Write an introduction.
    - Write a section on the architecture of the application and rationale for decisions made. As part of this, produce a UML diagram(s) that shows the architecture of your application. You can base this on the design from my slides with your changes and enhancements highlighted. I drew mine using PowerPoint, but feel free to use another tool or even to draw neatly by hand!
    - Write a section on your test strategy. IMPORTANT: Provide a five-minute screencast of your application working (some free screen-casting tools can be found online) and that focuses on your enhancements.

    - Write a critical evaluation section. Say what you found hard or easy, and what was omitted and why. Provide a critique on your design and the appropriateness, or otherwise, of the technologies used. State what mark you think you should be awarded and why.
- Upload a zip (not an RAR file) of all the source code, screencast and your report to Blackboard. Give yourself plenty of time since the file is likely to be quite large.

Learning outcomes

By undertaking this assignment you will:

- Demonstrate that you know how to use Ruby-on-Rails technologies: ActiveRecord, view templates, controllers and REST-based web services.
- Demonstrate the application of design patterns.
- Demonstrate that you are able to produce a UML diagram(s) during the design process and then translate this (these) into code.
- Demonstrate that you know how to undertake suitable testing.
- Demonstrate an ability to maintain existing software.

Submission Date and Instructions

You must submit the files zipped into a .zip file (nothing else please! no .rars, .tars, .gzs etc.). **If I cannot read your files your project will not be marked.**
Your solution to this assignment must be uploaded to Blackboard by Monday 8th December 2014 4pm. If you are late then please complete a Late Assignment Submission form (http://www.aber.ac.uk/~dcswww/Dept/Teaching/late-assign.pdf) and send this in to the Department office. If Blackboard no longer accepts submissions then you will also have to email the zipped assignment to cwl@aber.ac.uk (or a link to Dropbox or similar if the file is too large to email).

Note: this is an "individual" assignment and must be completed as a one-person effort by the student submitting the work. This assignment is **not** marked anonymously.
By submitting your work to Blackboard, you agree to the statement about the Declaration of Originality – you can read that at http://www.aber.ac.uk/~dcswww/Dept/Teaching/Handbook/assign_cover.pdf. This statement will be visible when you submit your report. You **do not** need to submit a separate declaration form.

I will attempt to provide marks and feedback by the Wednesday 7th January 2015.

Mark breakdown

Assessment will be based on the assessment criteria described in Appendix AA of the Student Handbook. However, the following table gives you some indication of the weights associated with individual parts of the assignment. This will help you judge how much time to spend on each part. Note how Cucumber-based projects place more emphasis on testing and less on the design and documentation section.

Successfully running the provided CSA prototype	Does the screencast show the prototype running?	5%
Design and documentation (non-Cucumber test project)	Is there a design diagram(s)? Is there an associated textual description? Is there rationale for design choices made? Quality of the design. Documentation quality.	30%
Design and documentation (Cucumber test project)	Is there a design diagram(s)? Is there an associated textual description? Is there rationale for design choices made? Quality of the design. Documentation quality.	15%
Implementation	Is the enhancement implemented and running? Code quality: comments, identifier names ...	35%
Flair	You implemented and documented something in a way that really impressed me. I'll know it when I see it.	10%
Testing (non-Cucumber test project)	Good set of unit and functional tests. Discussion of results and test strategy. Inclusion of screencast is essential. I need to see the proof.	15%
Testing (Cucumber test project)	Excellent set of Cucumber and RSpec tests. Discussion of results and test strategy. Inclusion of screencast is essential. I need to see the proof.	30%
Evaluation	Critique of your performance. Critical discussion of approach taken and technologies used. Problems encountered and if and how you solved them. What did you learn? The mark you think you should be awarded and why.	5%