SE31520/CHM5820 Assignment 1 2016-17

Part 2

Chris Loftus
Hand-out date: Tuesday 1st November 2016
Hand-in date: Monday 12th December 2016
Feedback date: Wednesday 11th January 2017
90% of Assignment 1

Part 1 helped you to acquire a basic understanding of the general structure of Rails applications. Part 2 will build on that experience by getting you to add to the CS-Alumni application.

Task

Your task is to extend and test the CSA application that was demonstrated in Workshop 3. The csa.zip file to build on is provided on the Blackboard assignment link. Note that you will need to have ImageMagick installed in order to run the existing app. It is also configured to use Rails 5. The first step is to get it to run.

There are three aspects to Assignment Part 2:

1. Develop a web service REST client.
2. Use ActionCable (Web Sockets)
3. Undertake some Cucumber testing.

Develop a REST web service client

Develop a simple command-line web service client that issues RESTful requests to the CSA application. You may use any language you wish for the client. This will require some restructuring of the CSA application so that it separates out the human web from the programmable web. A way to do this is described in lectures 6 and 7.

The web service API and client should only support the Broadcast feature and with this only the listing of existing broadcasts (no delete, show, or edit) and the ability to create a new one where you specify the feeds. You may wish to disable authentication / authorisation initially, until you have this running and until after I have discussed authentication in the CSA (lectures 16 and 17).

Use of ActionCable

Add a new broadcast "feed" to the CSA so that a user has a "notification feed" that displays CSA messages in real-time in a prominent area within the CSA site (the user can see it at all times). I have not talked about ActionCable, but essentially this new Rails 5 feature uses HTML5 web sockets to push information to a browser page. You will need to do a bit of research to work out how to achieve this. The two Rails books listed in the Aspire reading list for this module (see the Aspire link on the Blackboard page) have information on how to use ActionCable.

I'm happy for you to decide on the detailed design of the feed area on the CSA site. I'd imagine it would look a little like a scrollable Twitter stream (like we have on our

CS web page). The feed is only updated if it is selected as an output feed when doing the broadcast. However, when the whole page is refreshed the feed should show the last 20 messages, each with a time stamp and with the most recent at the top.

Cucumber testing

Provide stepped definitions that test the ActionCable notification feed to make sure it is operating correctly. You do not have to follow the behaviour-driven development approach, but you may if you wish.

I have attached a Cucumber Gherkin file to the Blackboard assignment link that you should use as the starting point for the testing. If you feel it needs modifying then please contact me first. Flair marks can be gained by adding more Cucumber tests and/or RSpec tests to test other CSA functionality. You may need to use Cucumber's support for Javascript. See my YouTube video https://www.youtube.com/watch?v=KBQhm-kMxyk on using Ajax and Javascript and Cucumber testing.

To upload

Write a 2000 to 3000 word report[1] (you will be penalised if you stray out of this range).

- Include a cover page with your name, email, date and version number.
- Write an introduction that describes what is to follow.
- Write a section on the architecture of the CSA application and rationale for decisions made when designing the ActionCable feed and redesign to support the RESTful web service API. As part of this, produce a UML diagram(s) that shows the architecture of the CSA. You can base this on the design from my slides with your changes highlighted; alternatively you may use your own design diagram.
- Write a section on the architecture of the RESTful client.
- Write a section on testing using the provided Cucumber Gherkin test.
- IMPORTANT: Provide a five-minute screencast of your application working and that focuses on running the client, the use of ActionCable in the application, and the tests running. Make sure the screencast has a voice-over.
- If you have gone for flair marks, highlight this work in a flair section. This will make sure that I don't overlook your additional work. Justify why you think this work is "flair".
- 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.

Learning outcomes

By undertaking this assignment you will:

[1] The word limit does not include the cover page, and table of contents, references, figure captions or figure annotations.

- Demonstrate that you know how to use Ruby-on-Rails technologies and REST concepts: 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

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.
If I cannot read your files your project will not be marked.

Your solution to this assignment must be uploaded to Blackboard by Monday 12th December 2016 4pm. You will be able to submit as a late submission after this deadline, but you will receive a zero mark. If you have a problem with upload to Blackboard then 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: as shown on the Blackboard assignment link.

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.

Screencast showing the running CSA app and the RESTful client.	Does the screencast, with voice over, show the CSA running and also the RESTful client running as specified above, plus tests?	10%
Design and documentation: RESTful web service	Quality of the documentation and the design.	10%
Design and documentation: ActionCable feature	Quality of the documentation and the design.	10%
Testing using Cucumber plus documentation.	Quality of the documentation. Quality of the testing implementation.	20%
Implementation of ActionCable feature	The ActionCable feed shows CSA news posts in real time.	20%
Implementation of RESTful client	The CSA functions are available from the client. Quality of code.	20%
Flair	You implemented and documented something in a way that really impressed me. Possibilities: more Cucumber tests, RSpec for controller and model testing, improving the CSA application. Also, deploying to Heroku and AWS. Check with me if you are not sure.	10%