SE31520
Developing Internet-Based Applications

Enhancing the CS-Alumni Application

Submission Due 4pm Monday, Decemeber 8, 2014

Thomas Mark Rosier (THR2)
110113188

Contents

Document Summary

Application Design

CS-Alumni Chrome Browser Extension

Rational

My rational behind deciding to developing a browser extension for this project, was that they have been a technology that I have been following for a while and I have not seen them used that regularly for small little applications that interface with a bigger application.

Another reason for my choice to develop a browser extension was that I wanted to learn about implementing some of the more advanced features within the Chrome SDK for example the use of 'Background Pages' which are used to keep a task running in the background while the extension is closed.

One of the newer features within the Chrome SDK had also caught my attention in the weeks before starting this assignment was the use of Chrome desktop notifications where you can create a notification within your application that will be displayed to the end user via the Chrome UI's notification draw.

I wanted to see if it was possible to encapsulate some of the key management features of the CS-Alumni application into an attractive and well presented browser extension that also notified the user of any new broadcasts that had been made.

Why did I choose to develop a Chrome extension over a Firefox, Safari or Internet Explorer, the main reason was due to Chris Loftus's preference of Chrome over the other browsers. In addition to that I felt that due to developing for Chrome also meant that the code will work without modification in Opera this a bonus in the real world as it means we can support a greater population of users without investing more money into development.

I also feel that Chrome has a much nicer development environment for developing browser extensions over its competitors, along with having some newer and more unique features that allow the Chrome based extension to stand out.

If this was a real world product it would not take that much resources to port the extension to work on Firefox and Safari aswell but this out of the scope of this assignment.

Module Diagrams

Program Operation

When the user opens the browser extension they will be presented with the home screen, this shows a small blurb about the application along with a disclaimer.

Due to this being an assignment this is essentially a development version of the browser extension thus I have included a link to the Unit tests on the home page to make it easily accessible during the development and testing of the application, if this extension was intended to be released as production code this link would be removed or turned off within the configuration.

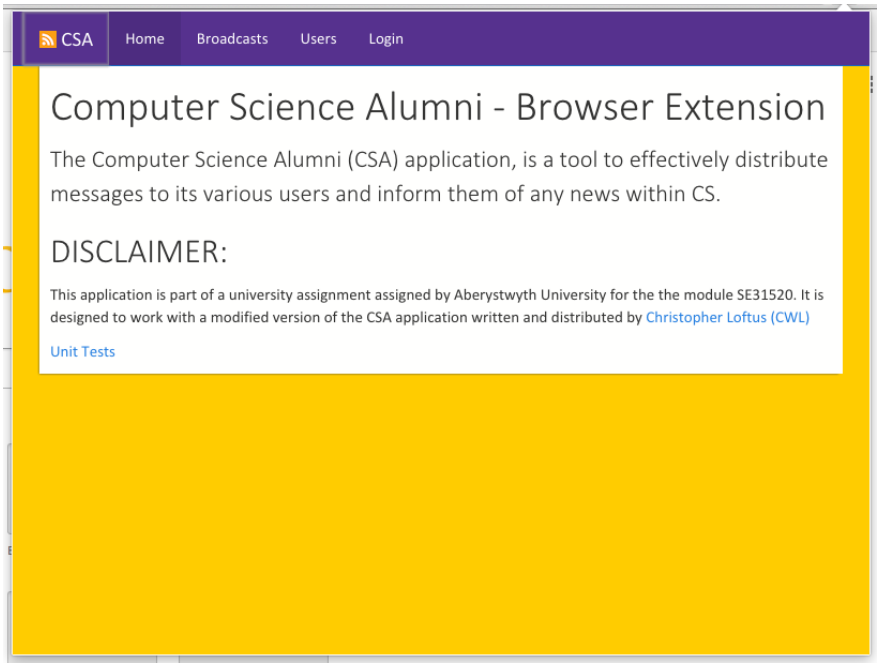

Figure 1: The screenshot above is the landing screen for the browser extension.

From the home page the user can navigate to the 'Broadcasts' tab, in the figure captioned there is no broadcasts available to the user so they are show a message that says that there is no broadcasts.

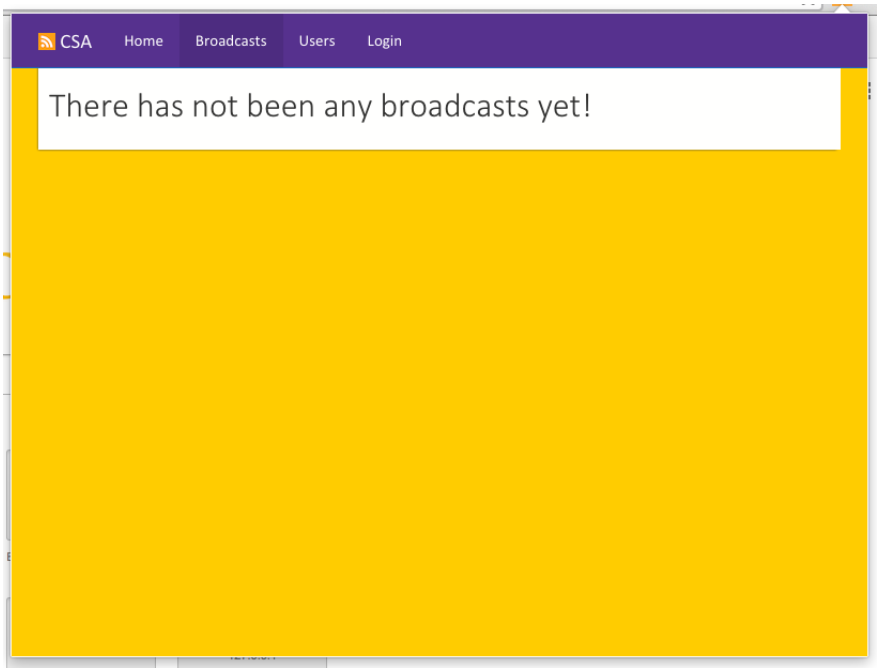

Figure 2: Above we see the broadcasts screen with no broadcasts available.

Again below we see the 'Broadcasts' tab this time we see how the page appears when there is broadcasts.

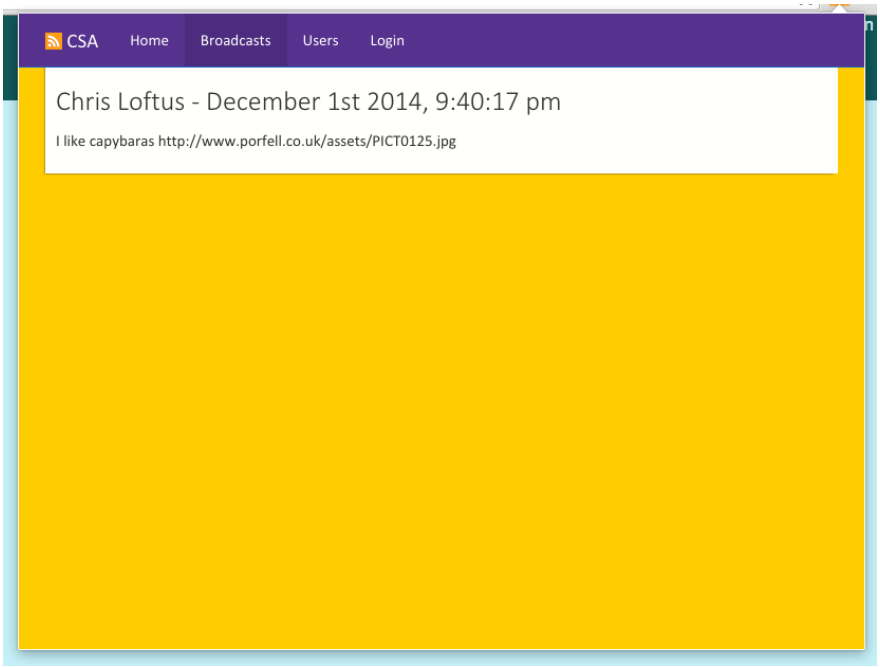

Figure 3: This is what the populated broadcast screen looks like.

This is the 'users' tab of the browser extension here the user can view a specific users details and edit there information if it is deemed appropriate.

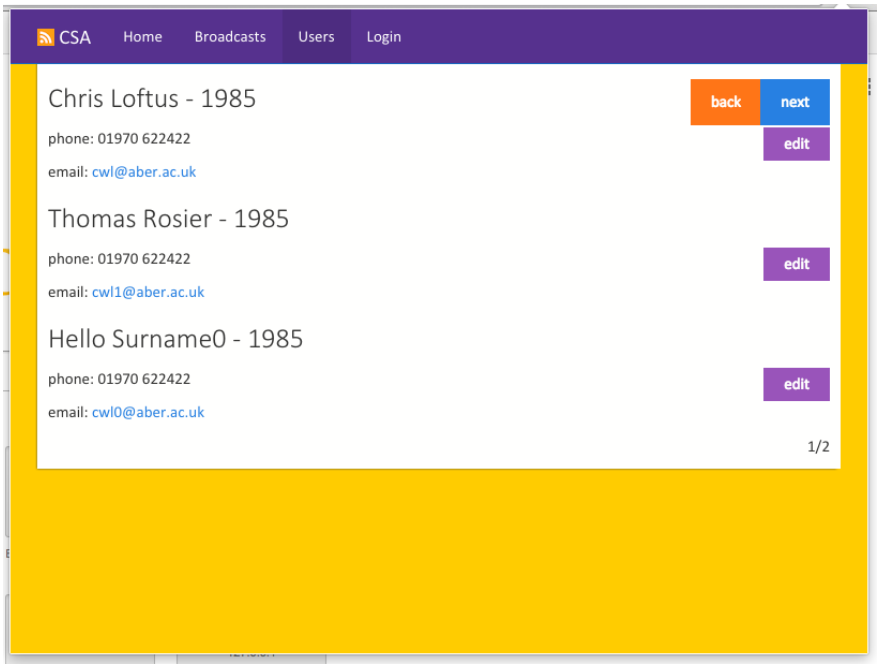

Figure 4: Here we have a screenshot of the user's screen.

When the edit button has pressed on the a specific user it shows this modal window that allows the user to edit the details that are kept on the current user they are editing.

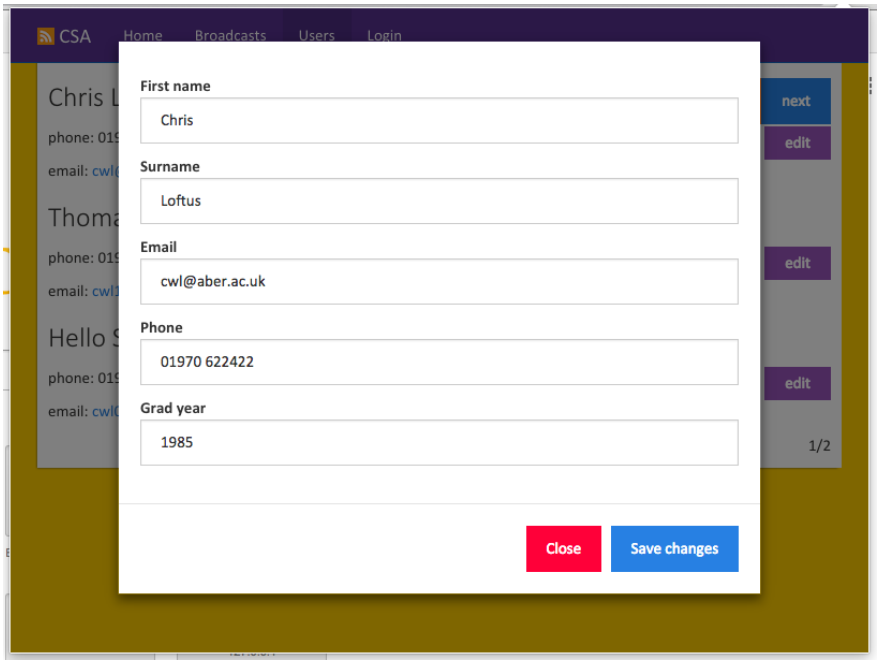

Figure 5: Above we have a screenshot showing the modal window we used to edit the user information.

The 'Login' tab is where the user enters there login credentials that will be used to authenticate the user against the CS-Alumni application.

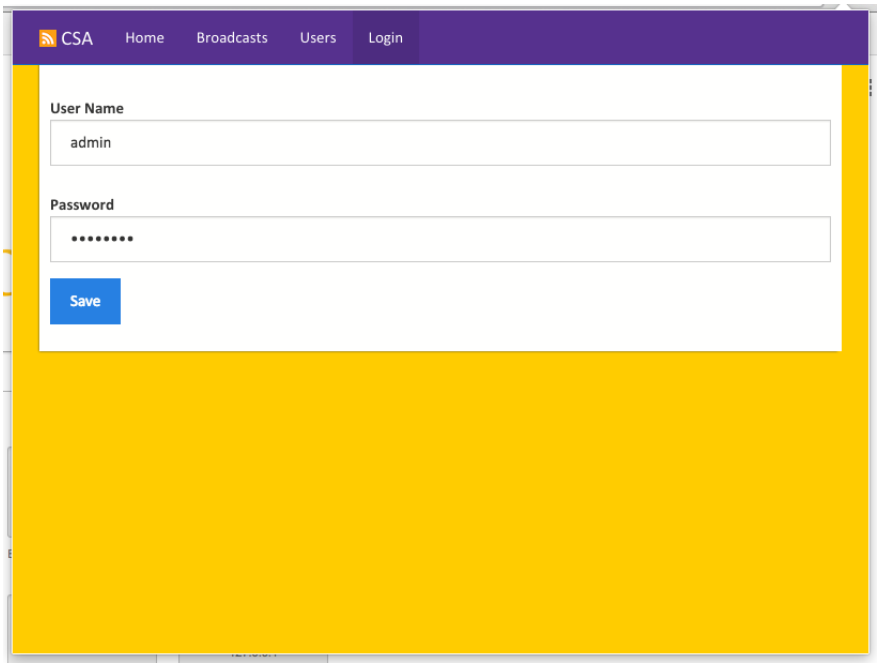

Figure 6: This is a screenshot of the login screen.

In the screenshot below i have created a new broadcast within the CS-Alumni application to to showcase the notification features within the browser extension.

Note: The destination list differs from the standard CSA application. They have been modified to remove the networks that do not work along with adding in the functionality to target the browser extension specifically.

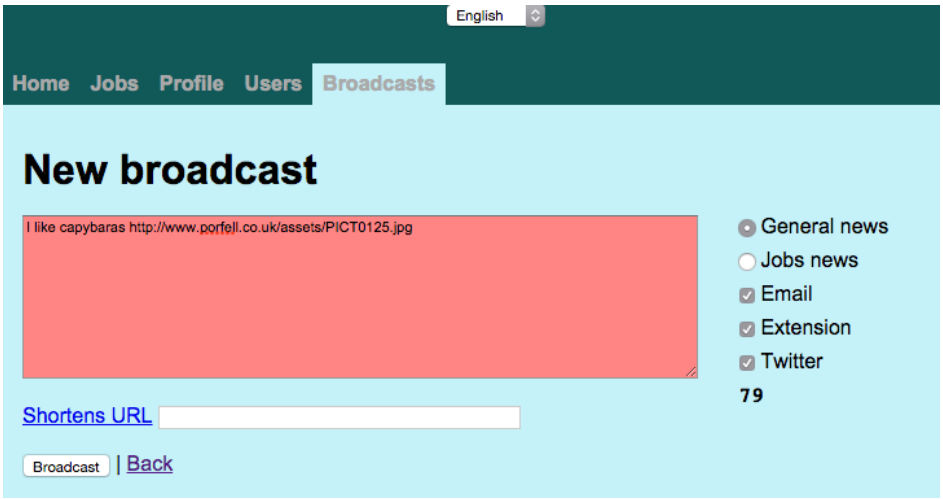

Figure 7: Creating a broadcast within the CSA application.

Confirmation of the broadcast being created with the parameters that I gave on the previous screenshot is pictured in the screenshot that has been included below.

You can see in the list of feeds that it includes the feed named 'Extension' which signifies that we want to alert users that are using the browser extension.

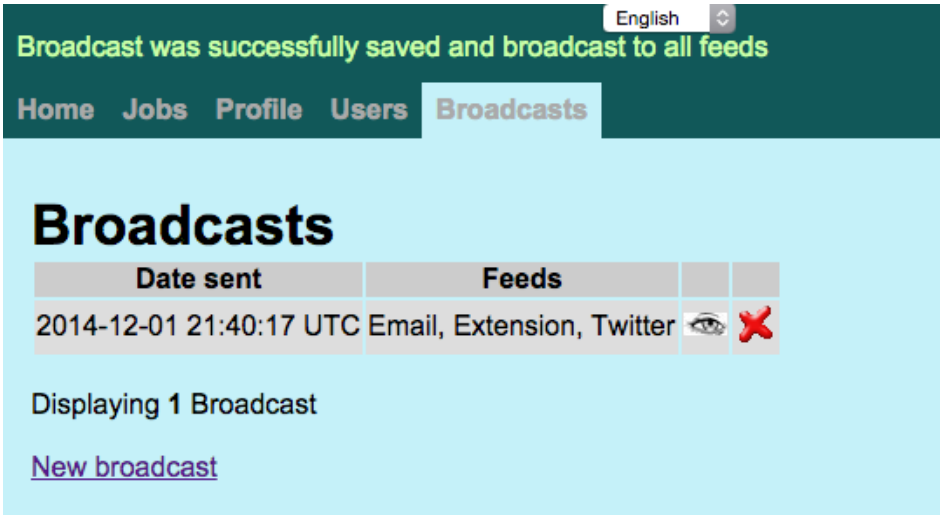

Figure 8: Here is the confirmation that a new broadcast has been created.

Here is shown a Chrome Notification for the broadcast that we just created, this notification will only be shown once when it has been created then is stored and will not be shown again to the user which will prevent them from being annoyed by repeated notifications telling them the same information over and over.

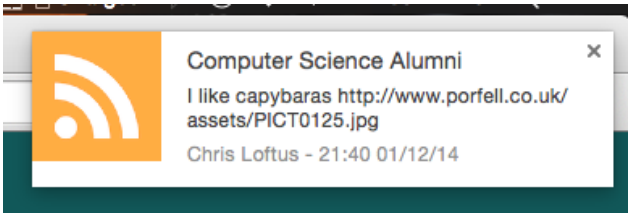

Figure 9: Here is a Chrome notification showing the latest broadcast.

Due to the nature of how the browser Opera is designed it users the same core frameworks as Chrome thus allowing it to use the same browser extensions without any modifications the screenshot below shows the same broadcast that we created earlier also being displayed within Opera's notification system.

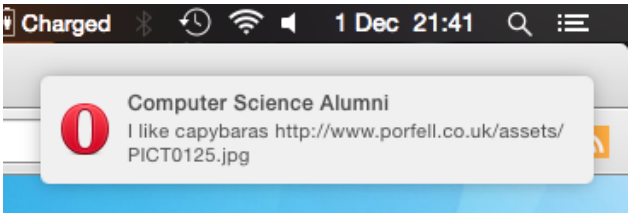

Figure 10: Same again but for Opera this time just to show multi platform support.

Just to confirm that this is a real broadcast and has not broken any of the other functionality within the CS-Alumni application this is a screenshot of the broadcast on the Twitter website.

Figure 11: Proof that the broadcasts still get sent to Twitter.

CS-Alumni Rails Application

New Additions

New Database View

New RESTful interface

Modifications

Changes to Seeds

Changes to Broadcasts

Changes to Security

Changes to REST implementation

Changes to Users

Communication Between Applications

RESTful Web Interfaces

Data Flow Diagrams

Application Testing

QUnit Tests

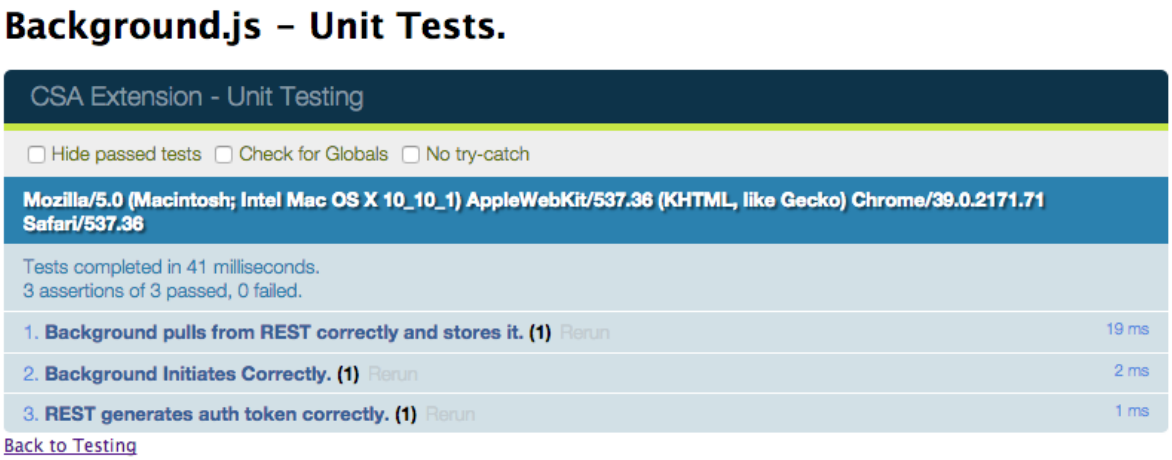

Figure 12: This is the unit tests for the background.js module for the browser extension.

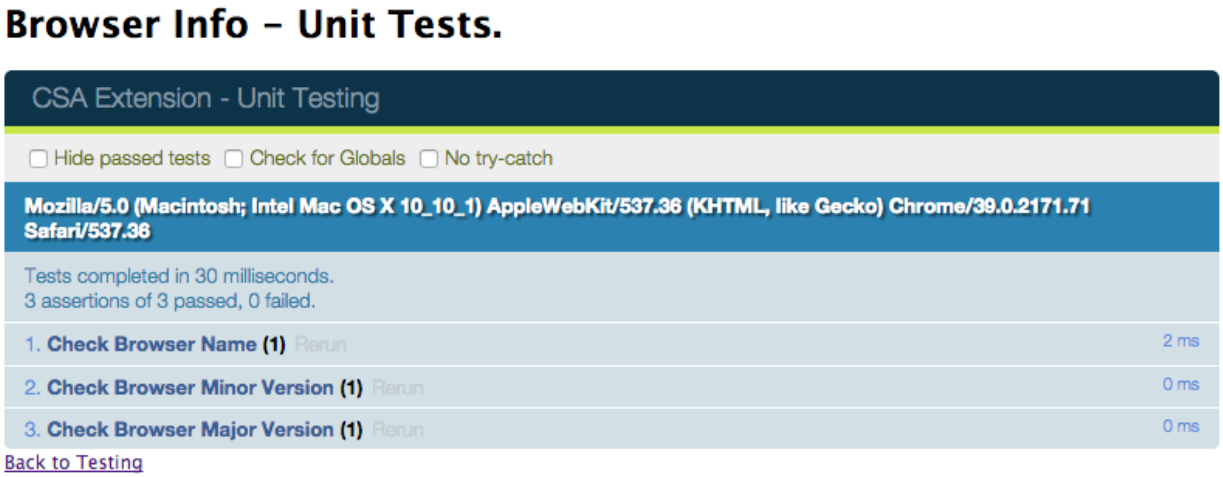

Figure 13: This images shows the browserinfo.js module completing its tests successfully.

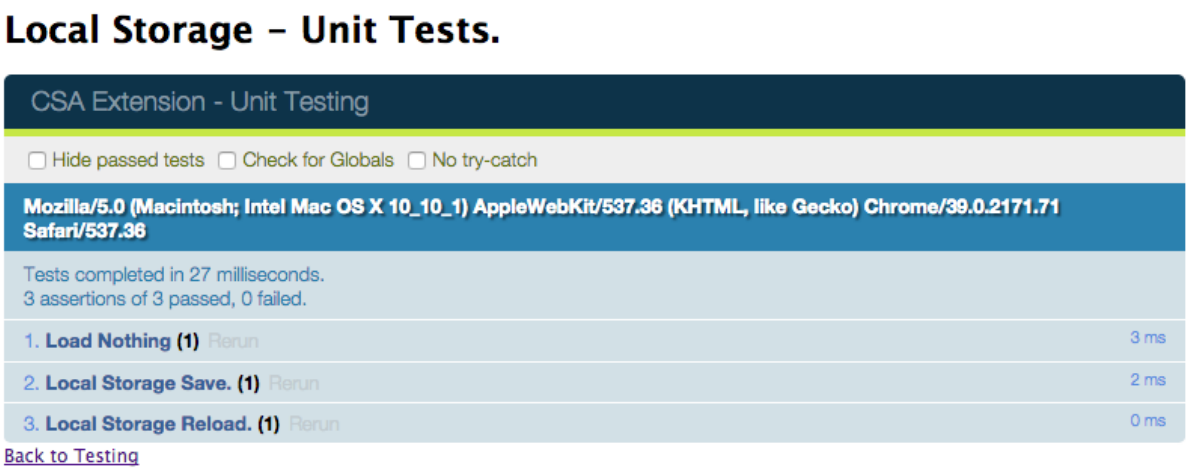

Figure 14: This shows the LocalStorage.js module passing all its unit tests.

Figure 15: Above we see the RestToolKit.js module successfully completing all its unit tests.

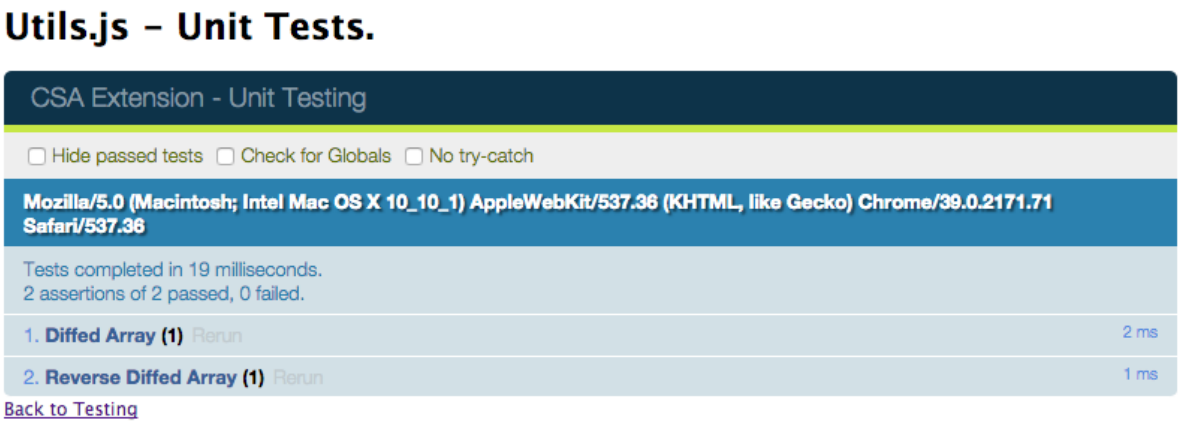

Figure 16: Included above we see the Utils.js module efficiently complete all of its unit tests.

Technologies

CS-Alumni Chrome Browser Extension

Javascript

Chrome Extensions API

Bootstrap

jQuery

Moment.js

Alertify.js

Qunit

CS-Alumni Rails Application

Ruby

Ruby on rails

Qunit

Communication Between Applications

RESTful Interfaces

CORS

Basic Authentication

Evaluation

Something Extra

Figure 17: Above is a image of the Pebble smartwatch running the CSA application.

Figure 18: This is a screenshot taken from the Pebble smartwatch.

Attributions

References

[1] Mark McDonnell. *SQLite and ActiveRecord*. Oct. 2013. URL: http://www.integralist.co.uk/posts/sqlite-and-activerecord/.

[2] Daniel Morrison. *Building Awesome Rails APIs: Part 1*. June 2013. URL: http://collectiveidea.com/blog/archives/2013/06/13/building-awesome-rails-apis-part-1/.