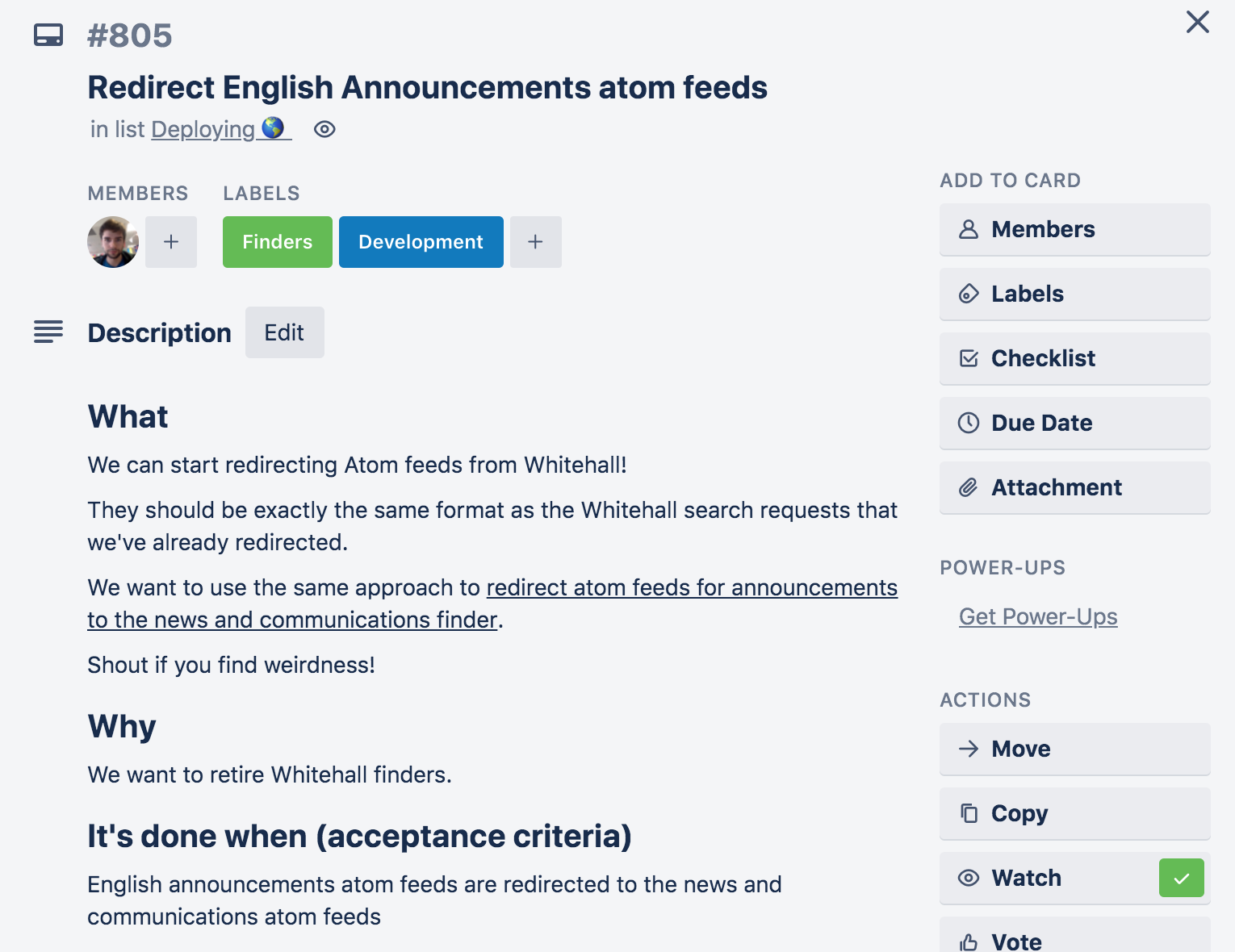
# Adding a Feed Subscription Redirect

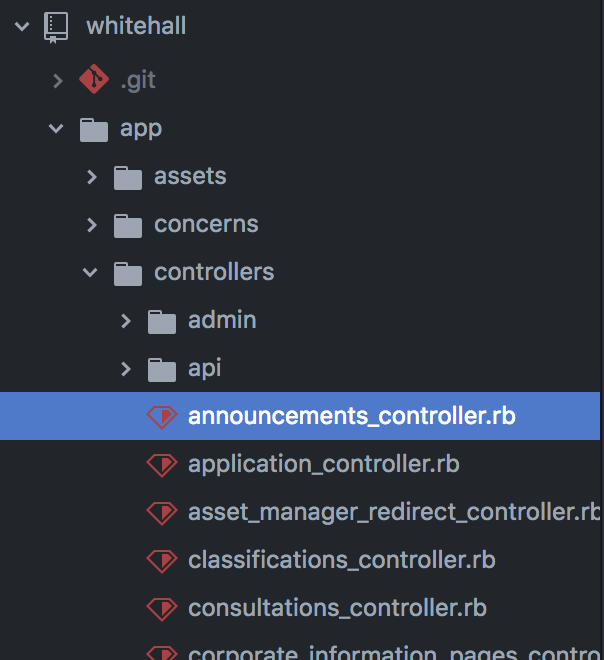
The main project I was working on that this ticket was involved with was a process that my team was going through, which involved replacing older search tools with more up to date equivalents. This meant that the older search tools would need to be gracefully decommissioned, and newer ones put in their place. As part of this, whenever one of the older tools were brought offline, there would need to be redirection logic introduced that ensured users that attempted to visit the old tool’s URL would be handed over to one of the newer alternatives.

This particular piece of work looked at a functionality that was included with this tool - the ability to obtain a web feed that would keep up to date with any new information that search tool encountered. One of the more common uses of this was to add it as an RSS subscription, though there were also other uses.

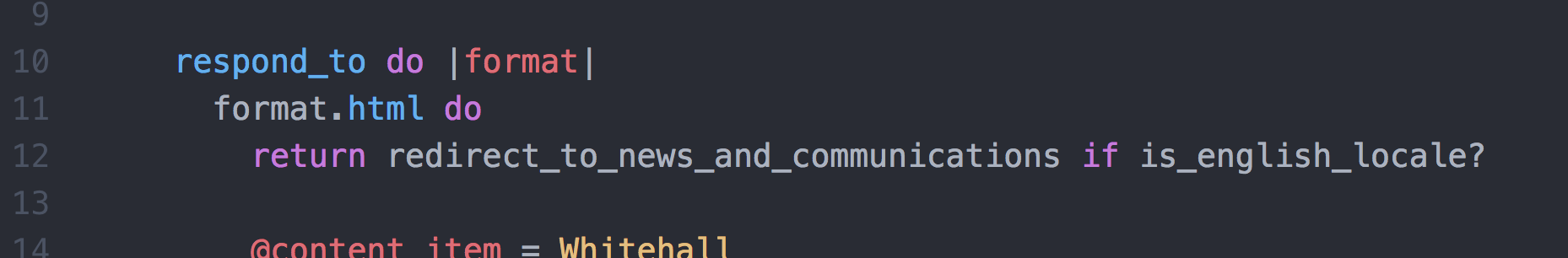
Initially, I picked this ticket up from our team backlog on Trello. Because I had already worked on redirection logic for other parts of the search tools, this was a good fit, as it expanded on things that I already knew about. I started by reading through the ticket to understand the requirements, and check out any of the related information. I then put my name on the ticket, and moved it into the ‘doing’ column to indicate I’d started working on it.

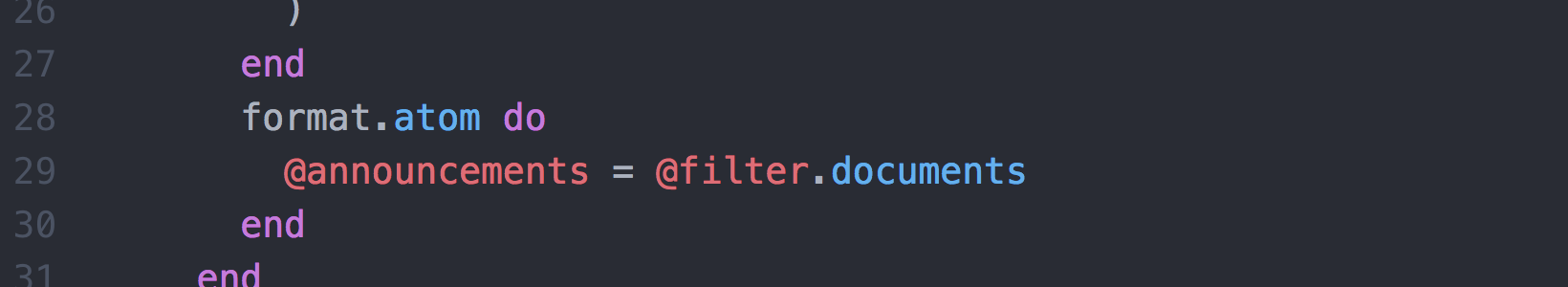


After I’d worked out roughly what was needed, I started to look into the code. I knew from my earlier work on the same project that the redirects were handled in the controller part of the code. I wanted to find the appropriate controller for the Announcements atom feeds, and this turned out to be the ‘announcements\_controller.rb’ file.



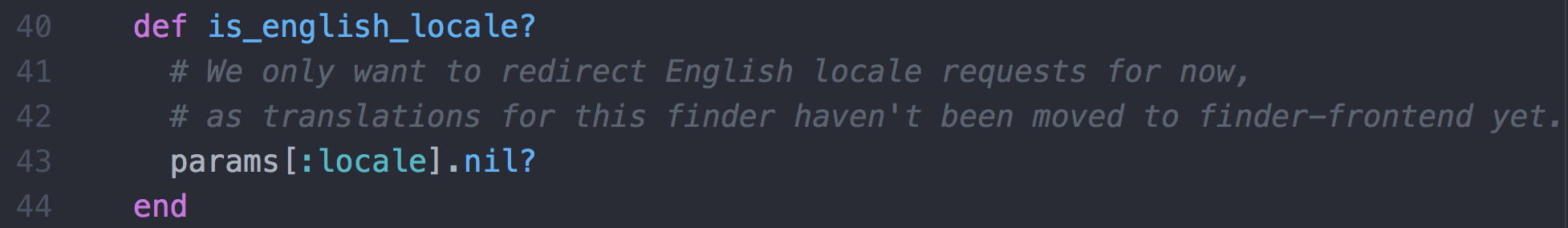
I then looked at the announcements controller to get a better understanding of how it worked. There was already a redirect in place for any requests that came in, but this would only happen if the request was in an html format (line 11). I also saw that later on, if the format was ‘atom’ (the feed format) then it wouldn’t redirect, and instead load in the current setup (line 28).





I then needed to work out what changes needed to be made so that when the format was atom, it would redirect to the more appropriate alternative. It would need to match the redirection logic for when the format was html, but ensure that a ‘.atom’ was included on the end of the request. Because of this, I decided that reusing the redirection logic was sensible, but I’d need to add some additional checks to determine whether the function should return an ‘atom’ style URL.

To do this, I looked into the code to see what the ‘redirect\_to\_news\_and\_communications’ function did. It looked like the end result of the function was to redirect to a particular path, while maintaining any additional parameters that might have been in the original request. Additionally, I checked to see what the ‘is\_english\_locale?’ function did. It turned out to perform a simple check to see if this was in anything other than the default English setting. There was a handy comment explaining that redirects should only be done when if it’s in English, as other languages hadn’t been set up in the new tool yet. I decided that I would need to run this check for any code I added, as I shouldn’t redirect without ensuring it was a request in English.

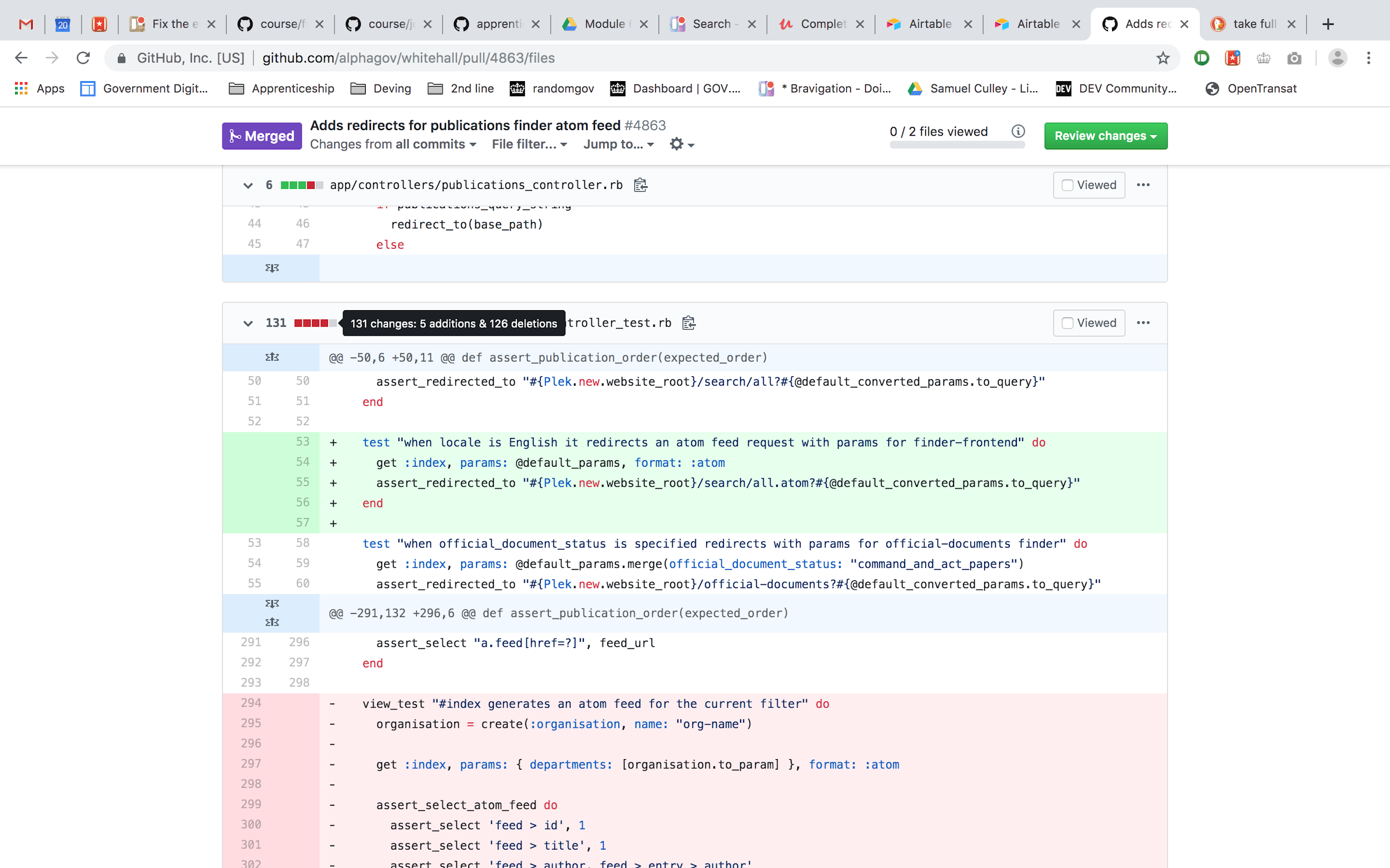


Next, I added a parameter check to the redirect function, which would default to an empty string. I named this parameter ‘format’, as this could potentially be reused by other parts of the code later, if there were other formats that needed to be accounted for. This would help keep the code readable.

I then added an additional line to the atom formatted response, which called the redirect function. I also added a parameter to this function call, that contained the additional ‘.atom’ string that would need to be added to the end of the path. I then ensured that the function interpolated the parameter to the end of the path, so that whenever the format was atom, the redirection would include the ‘.atom’ part of the path, and the redirect would point to the right spot.



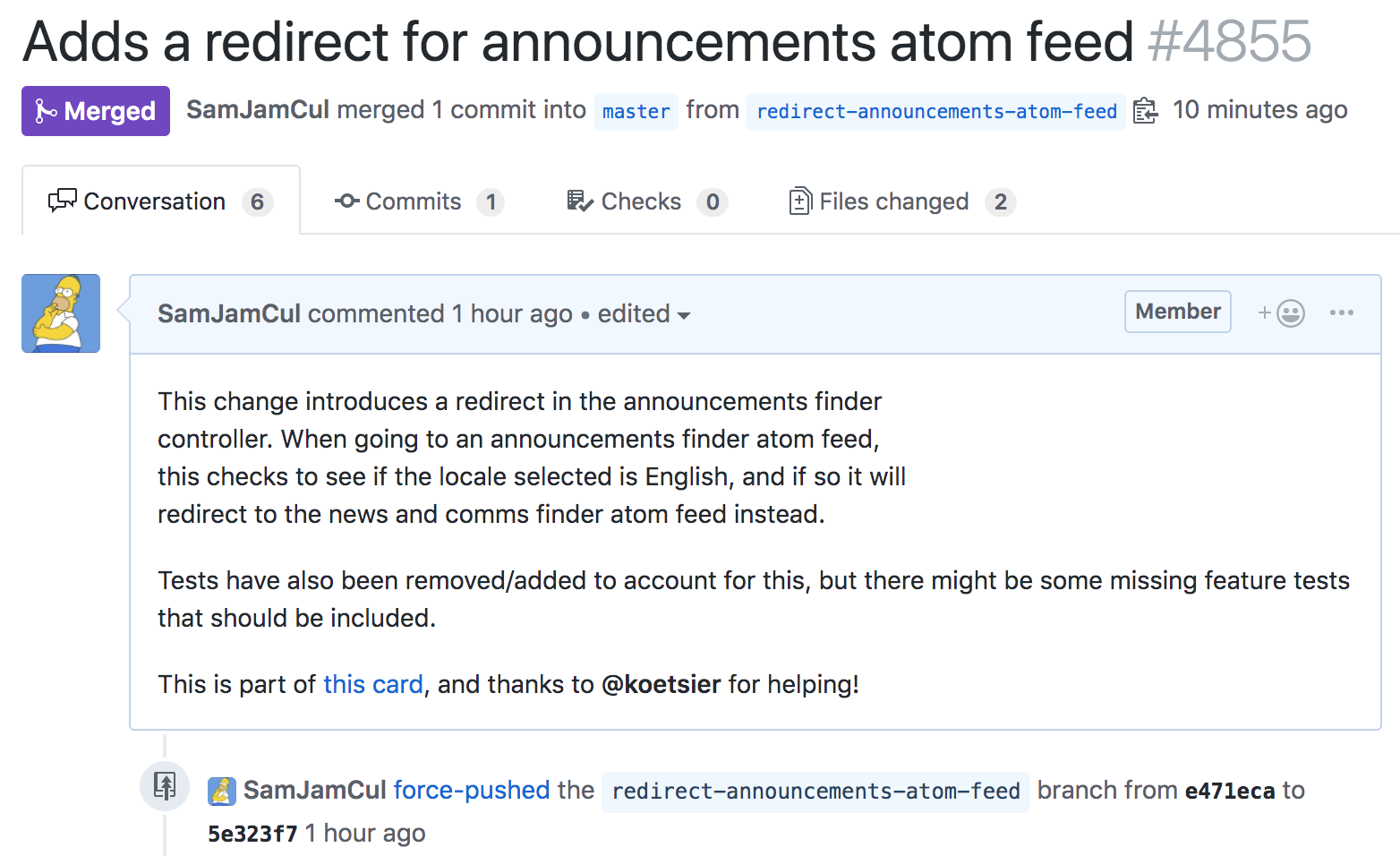
I then needed to write tests for the new code. This also meant I’d need to remove tests that would no longer pass as I’d changed how the atom format worked (when the request was in English). This ended up being over 100 lines of test code that were removed.



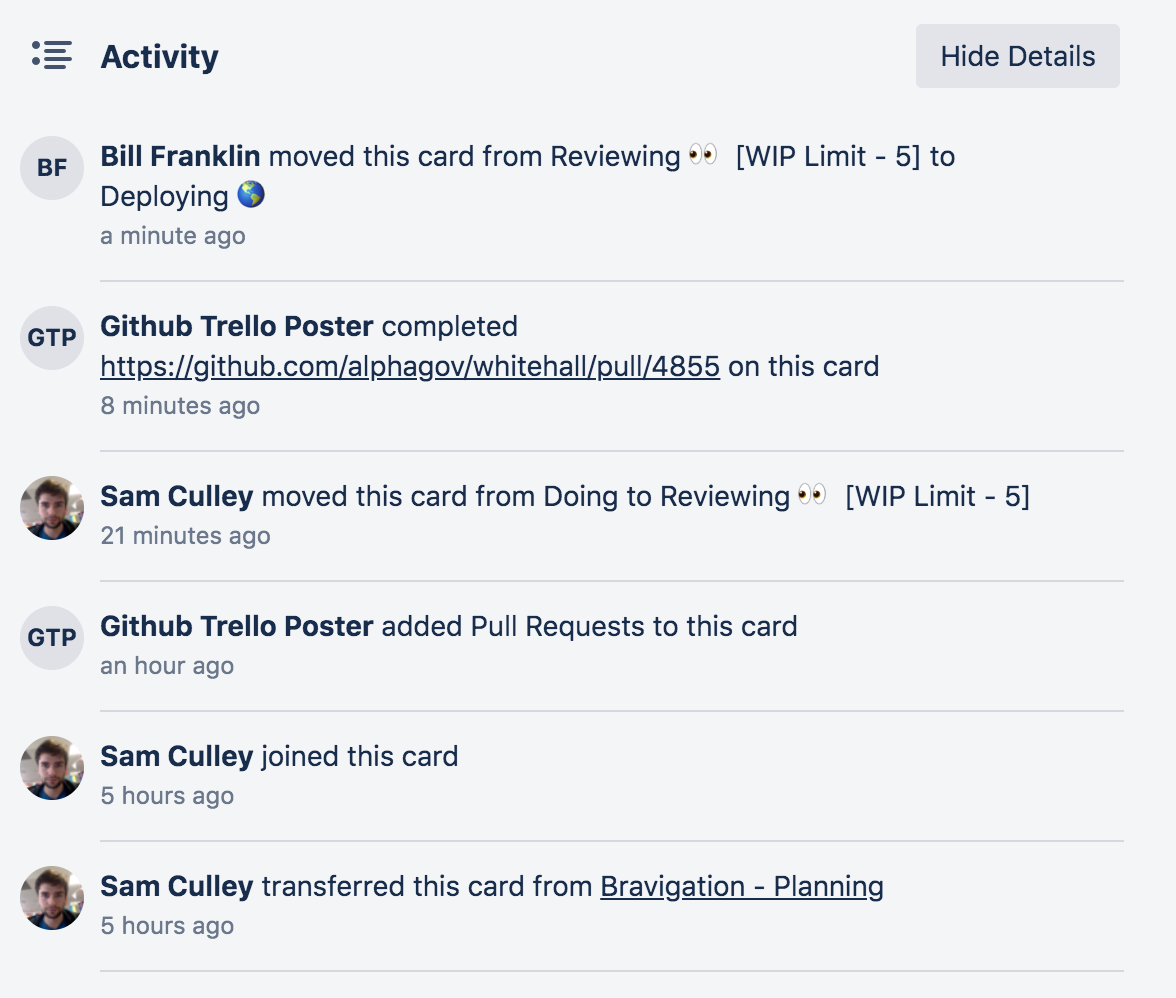
Because the new redirect only needed to point to another part of the codebase, it wasn’t necessary to test it in as much detail. The only thing that needed to be certain was that the redirection path was accurate. I added a single test that checked this, ensuring that the path was correct when the format was atom, and that any additional parameters would be carried across.

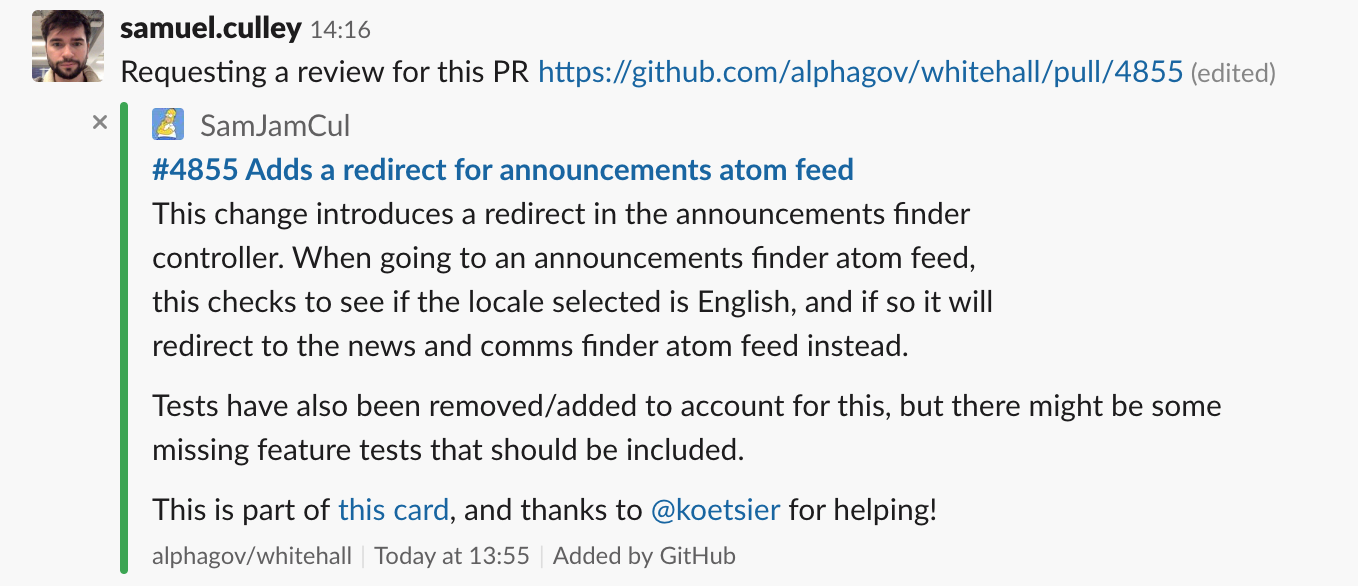


I committed my changes using Git version control, and pushed the new branch up to the remote repository. From here, I made a pull request, and summarised the changes that I’d made. I also linked to the original ticket where the requirements came from, and tagged a team member who had advised me earlier.



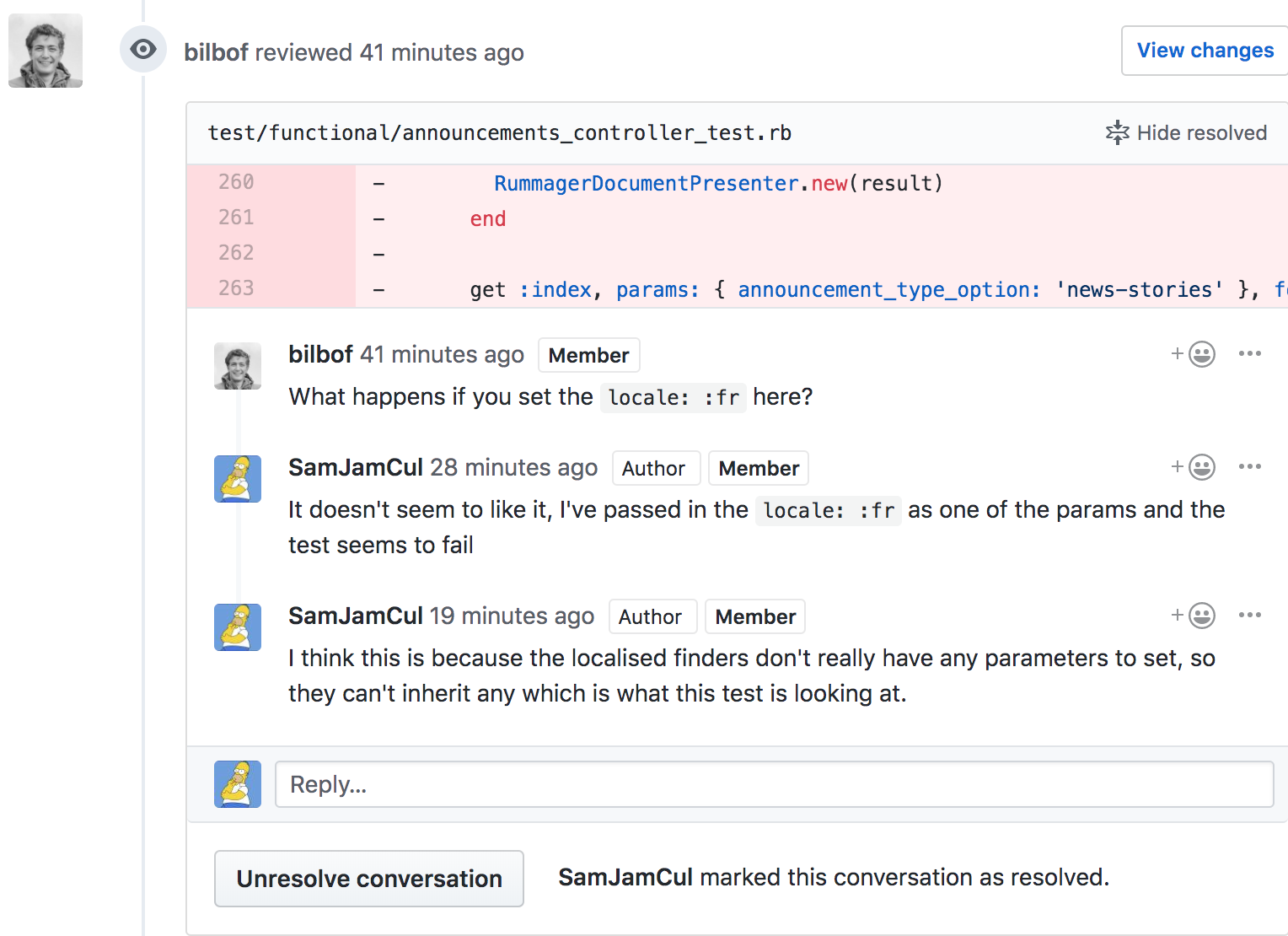
I also posted a request in the team Slack channel for someone to review the pull request. I also moved the Trello card into the reviewing column, to indicate that it had moved out of the doing phase.



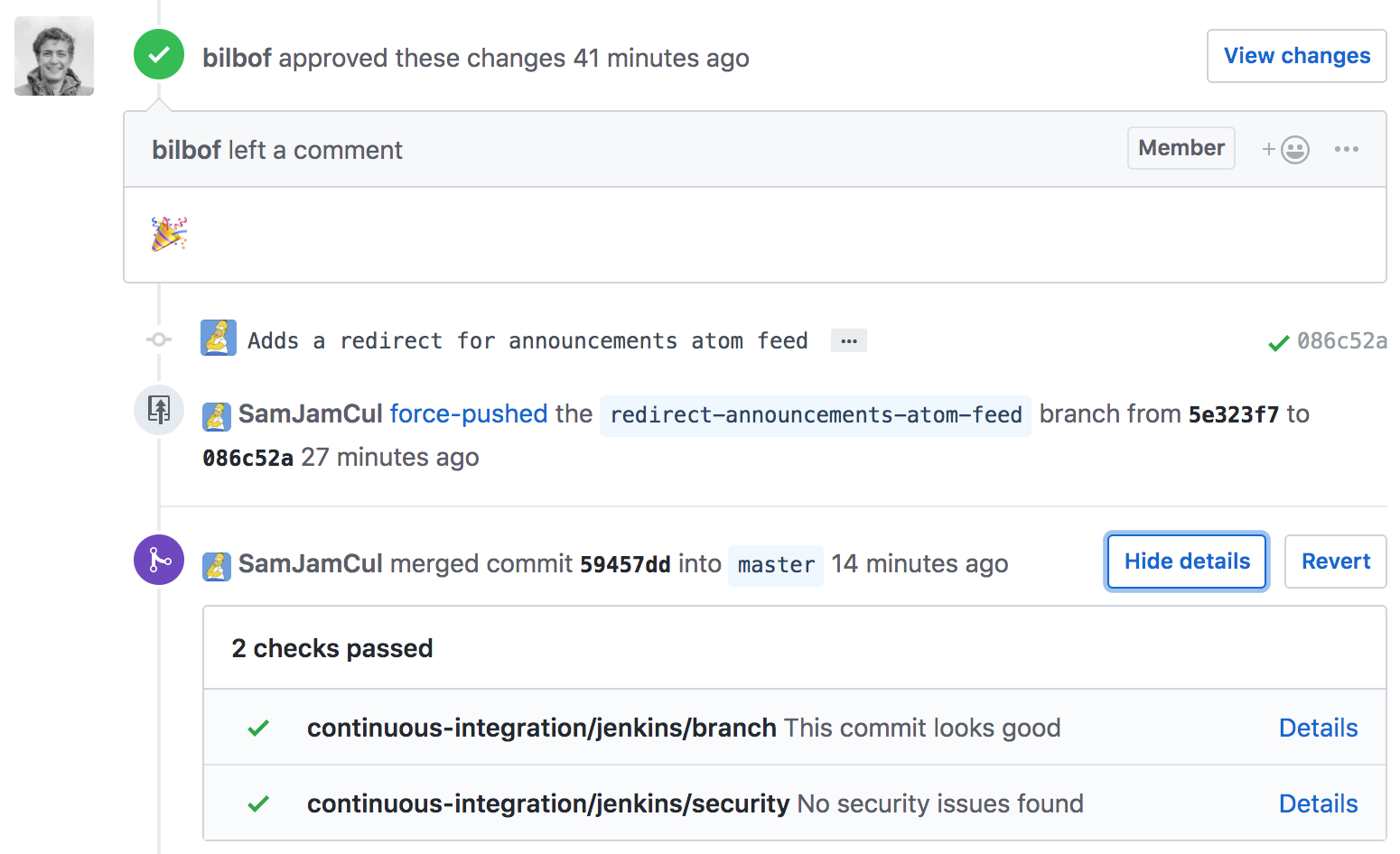


One of my colleagues reviewed the code, but also had a question about the test that I’d set up. He wanted to know how a test that I had removed would respond if the locale was set to ‘:fr’ (French). I modified my local version of the test so that the locale was different, and tried to run it. The test came back with an error, which turned out to be because the non-English versions of the tool don’t handle any additional parameters. The test was checking to see how the English version handled parameters, so changing the locale would cause it to fail.

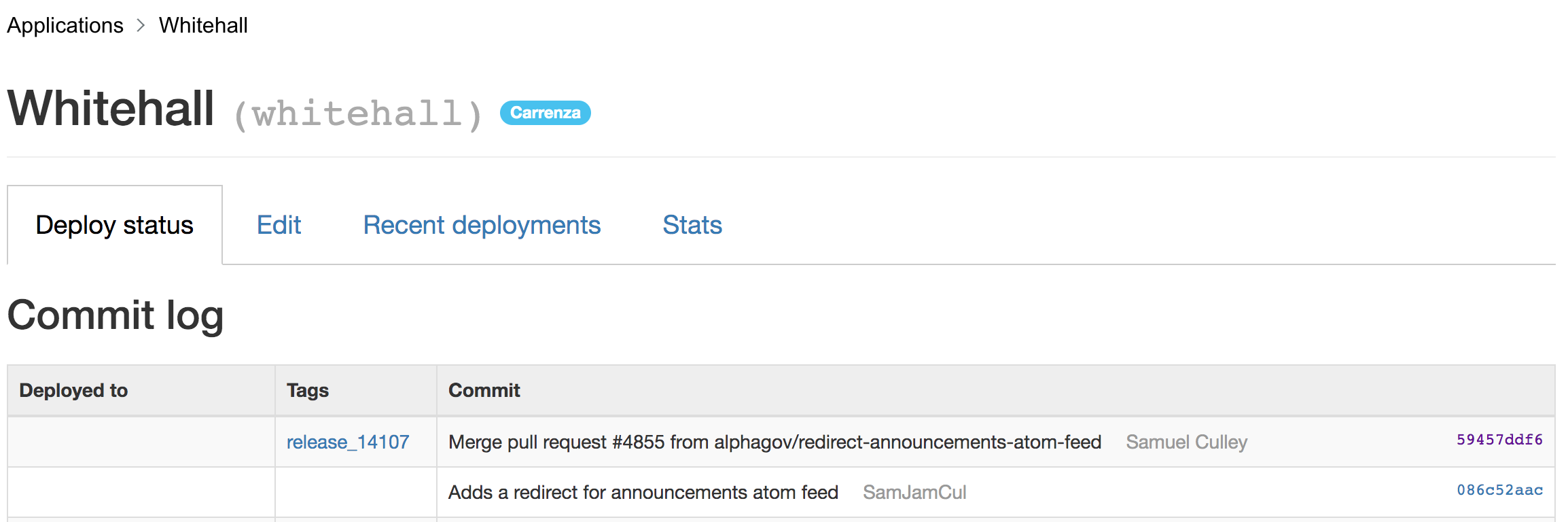
I responded to my colleagues’ question with the results of what I’d found out.



Now that my team member was satisfied with my response, he approved the overall PR. Once he had done this, I then checked that the continuous integration checks had also passed. This would have thrown up any errors related to code formatting, or if any other tests had started failing as a result of the changes I’d made. The checks actually did throw up some formatting issues, so I corrected those, and pushed the new changes up. The image below shows my corrections being submitted after the approval.



Once I had merged the branch, I then monitored the continuous integration process to watch my code move through the deployment environments. Initially the automated checks happened in the integration environment, but once it was merged, these moved to the staging environment, where it would be put through pre-live checks. After that, it moved to the live environment, where the changes would come into full effect. At this point, I moved the Trello card into the done column, as it required no further work.



## Summary

This project built on my knowledge from the previous redirection work, but it required me to layer things on top of pre-existing functionality. I found this more interesting, as it made me use more elegant code to try and squeeze more usage out of things that were already there. I also learned more about analysing the code base, as it required me to follow through the logic of the code from a starting point, and I had to make sure that I’d accounted for all the different behaviours.

I think I approached this project well, and I was able to reason out what the minimum change required was to meet the requirements. I definitely could have used a more long winded solution, and in some ways the ‘shortcut’ that I took might have affected readability in the long run. It might have been good to have added a comment to the code clarifying why there was a parameter in the function call, as this would have made it easier to understand later on.

The end result of this project was that users who utilised a feed from the old tool would now get an updated feed that instead drew from the new tool. Additionally, if anyone were to try and access the old tool’s atom feed, they would instead be shown the new tool.