Hi Claudia

Thanks for giving me the opportunity to attempt the Waracle Full-Stack Challenge.

It’s 7 years since I did any web work, and things have changed a lot. Both for the front-end and back-end. Many of the technologies in common use today didn’t exist 7 years ago or were in their infancy.

The challenge suggests it could be done in half a day, but since I was coming back to this after a long time away, and I’d be learning most aspects of the tech from scratch, I gave myself a week, 7 days, to get my head round everything.

The Challenge is tech agnostic, but I decided to use new tech wherever I could rather than looking back at anything I’d used before. I’ve created a few web-sites with jQuery and written back-ends in C# and PHP before, running under Apache on a Windows VPS with MySQL.

First, as a long term Visual Studio user, I determined that VS supports node.js, and whilst researching as much as I needed to at every stage, the path I followed was:

Install Windows 64-bit node.js, which also installs npm. Then I used “npm i” to install the following modules: express, ejs, mongodb, image-to-base64, react and react-dom. Also, babel, css-loader, webpack, lodash and busboy. I also used jQuery client-side, the only tech here that I’ve used before.

I also decided to use TypeScript server-side, use Git and GitHub and create a Progressive Web App.

How did I get on? I’d say not bad in general, although I’ve failed a couple of hurdles. I’ve got my head round Git and to some extent GitHub, so if you go to: <https://github.com/AndyWhyte/WaracleDemo>

and clone it, you’ll get my project. If you install all the node components above and run it with node.js and navigate to localhost:1234/home, you’ll see the mini-site I created. I hope you like it!

It runs with mongodb behind and is pre-populated with a bunch of cakes. Feel free to add and delete cakes as you wish. I know the Challenge said image picking and file uploading was not required, but I’d implemented file upload on both the C# and PHP servers, so I was thinking about trying it.

So, when I realised html would let me display base64 images directly on the client, and I could save the images as base64 strings in mongodb, and have the whole thing running from a local server and the db on the cloud, I just had to do it! I apologise for letting feature creep get the better of me, but I’d say it was worth it. I couldn’t find a base64 encoder for node though, not to encode from a buffer, so I wrote a version of btoa() myself.

Where did I fall down? I gave myself a week to learn these new concepts and frameworks, and in the end I couldn’t get webpack to build a bundle I could run. I’ve got a bundle, but it can’t find entry points for jQuery or my <body onload=”InitBody() function, so everything’s unbundled. I think there must be some export settings I couldn’t find.

Also, I couldn’t get Babel to transpile the .ts to js. I installed the version npm offered, v7, but when I ran it, it said it was using some v6 components from somewhere. Whether VS had installed some v6 components it was finding instead of the v7 ones in node\_modules I can’t be sure. But whatever the case, I couldn’t transpile and I ran out of time to debug the system, so I never got as far as trying Grunt or Gulp to build the project.

When I realised I couldn’t offer a build from the command line, I committed my server-side .js files, so it can run directly after cloning. Apart from that, someone would need Visual Studio 2017 (I used Community Edition) to rebuild the project.

The project itself? It’s for sure not at production code quality. I used Stack Overflow for reference, so in a perfect world I’d re-write much of the code, but I really enjoyed writing it all and creating the site. I wasn’t sure originally if I could do it as a single page app, but jQuery AJAX came through for me, both with uploading the images to the server and for partial page reloading.

After looking at Angular and realising it was too much to get hold of in the time I had, I’d settled on React. But when time was running out I resorted to vanilla with jQuery on the client, which is what I’d used before. I’d still like to create a React front-end as it looks an interesting bit of tech. Also, I had considered using Redux, but this project really didn’t need much state. I went old-school and held three hidden html elements on the single page to store the number of images, the names of all the images as a string and the id’s of all the images as a string. I think that was ok for this project.

I probably overused .js for page setup. Initially I intended for the display to be resizable, but that was not achievable in the time I’d given myself. Instead, I could probably have used css to greater effect than I did. But I think my code works fine in general. For a real project I’d try to organise my client side .js files a little better, but as a whole the site works well.

There is one bit of code I couldn’t find a good solution to. When adding a cake, having just uploaded all the data, I wanted to immediately go back to the home page, but I also wanted to reload the home page with the additional cake (or cakes). I’m quite happy with asynchronous coding, but the file upload function returned a value, so I couldn’t chain on to the back of it to reload the home page.

Initially I was getting some sort of db conflict and cakes which had just been added had disappeared on the home page, so I added a timer before making the second db call. This is definitely not production code! There is surely a better way. I’d try some handshaking with the server or have the server queue the db accesses, but that’ll have to be for another project!

Two small asides. The Challenge asked for GET, POST, PUT and DELETE to be used. But I couldn’t find a good reason for using PUT anywhere. Also, the Challenge specifies that the cake payload has an id, but I’m not sure if that’s the payload going to or from the server? I opted to use the mongodb option of allowing it to define ids itself. So ids were coming down and being used to view cakes and delete cakes, but I wasn’t creating them myself. This does mean that the same cake can be submitted a number of times, but I thought it was the best way for the demo as I didn’t want to be keeping track of ids when I was adding a new cake.

Also, having got the PWA system working, I must remember to always turn it off during development!

Thanks again for this opportunity. I’m still very interested in your Full-Stack position, or any similar position you may have, but it’s clear there are some areas where I couldn’t hit the ground running.

I’m familiar though with the repetitive growth pattern with any new area (or in this case several areas) of tech. After a few similar projects, with all the building blocks in place and upon learning to use React effectively, I’d expect that I could create something like this in a much shorter period of time. I do think I have potential as a mobile app developer! :0)

Thanks again

Andy