Computer Science Final Year

Evaluation of a Haskell Web Framework

Junaid Ali Rasheed

April 22, 2018

Final Year Project Definition Form

Student's name Junaid Rasheed
Course Computer Science
Project title Evaluation of a Haskell Web Framework
What is the project about? Comparing a Haskell Web Framework with a more traditional framework. The two frameworks that will be compared are Yesod (Haskell) and Django (Python). A website similar to Twitter will be created with both frameworks and then both websites will be compared. The comparisons will include differences in page load speed, safety, reliability and whether type checking during compile time vs runtime helps reduce bugs, maintainability by comparing the ease of adding a new feature to both websites, and the ease of testing in both frameworks.
What is the project deliverable? Two websites that are functionally identical, one developed using Django, and the other using Yesod. Then a report will be written comparing the reliability, maintainability, speed, safety, and possibly the scalability of both frameworks. The report will evaluate the advantages and disadvantages of making a website using Yesod.
What is original about this project? There has not been any detailed comparisons between a Haskell Web Framework and a Web Framework in a more traditional object oriented language like PHP or Python. This project will provide enough detail to people looking into using a Haskell Web Framework to help them inform their decision.
Timetable showing main stages in work plan End of October: Comfortable with Yesod and Django, create a simple website with both frameworks. Create tests for the simple website. November: Start to create a simple Twitter clone in both frameworks. Ensure tests are created for new features. Debug any errors. End of January: Simple twitter clone finished, users can make posts, follow each other, make 'hashtags' (any word with a '#' preceding itself is linkable to other posts containing the 'hashtagged' word and looking up the word will show all posts containing the 'hashtagged' word in a paginated results view.). Tests created for all features, bugs debugged. End of February: Add a new feature. The feature that planned is a way to send private messages directly to other users. Users will have an area displaying all private messages sent and received and will be able to reply to other people's messages. Add tests for this feature and debug any bugs encountered. End of March: Record and resolve any bugs and errors in the framework, ensure that each framework has a sufficient number of unit tests and that all unit tests pass. Compare both frameworks, with a focus on speed, reliability, and safety. April: Begin and finish the Final Project report, prepare for live demos.

Student's signature [illegible] Date 19 Oct 2017

Supervisor's signature [illegible] Date 19 Oct 2017

Computer Science, SEAS, Aston University

Contents

List of Figures

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

1. Introduction

In this report, we will compare two web frameworks, one written in Haskell and another, more popular framework, written in a typical object oriented language. To perform the evaluation, a functionally identical website was created in both frameworks.

One issue that individuals face when looking into Haskell Web frameworks is the lack of detailed comparisons between these frameworks and more traditional frameworks that these individuals likely have experience in. This report will provide an in-depth look at the advantages and disadvantages of choosing to use a Haskell Web Framework, and will help these individuals come to an informed decision on whether or not a Haskell Web Framework is best for them.

1.1. The Chosen Frameworks

Yesod is a fully featured and modular web framework written in Haskell. Yesod claims to use features of the Haskell language to provide a fast, modular, and type safe web framework. By choosing Yesod as the web framework for the Haskell language, we will be able to determine whether or not Haskell's type safety, referential transparency, and lazy compiling is an advantage or a disadvantage for web developers.

> Yesod attempts to ease the web development process by playing to the strengths of the Haskell programming language. Haskell's strong compile-time guarantees of correctness not only encompass types; referential transparency ensures that we don't have any unintended side effects. Pattern matching on algebraic data types can help guarantee we've accounted for every possible case. By building upon Haskell, entire classes of bugs disappear. (Snoyman, 2012, Introduction)

Django is a Python Web Framework. Django, like Yesod, is a "batteries included" web framework, "instead of having to open up the language to insert your own power (batteries), you just have to flick the switch and Django does the rest.” (George, 2017). I chose Django for a number of reasons

- “Batteries included”, like Yesod
- Modular, like Yesod
- Python is now more common than PHP, second only to Node.js (George, 2017)

The Django and Yesod web frameworks have a similar set of features. This ensured that we could make a functionally identical site in both of these frameworks with a similar amount of effort. This enables us to make a fair comparison between Django and Yesod, enabling us to come to a conclusion on whether a Haskell web framework may be a good choice for a developer rather than a more tradition web framework.

1.2. The Report

The rest of this report will discuss any pieces of work similar to this project, the process of writing the code for both frameworks, including any preparation that had to be done, and a detailed evaluation of the websites that were produced. The evaluation will compare page load speeds, the reliability and maintainability of the websites, the ease of writing new code, the ease of debugging issues, the features including in each framework’s test suite, and whether the static typing and type safety of the Haskell language help or hinder the process of developing a website.

2. Background

In this chapter, we will discuss previous work and research pertaining to Haskell web frameworks and any real-world sites built using a Haskell web framework.

2.1. Similar Previous Work

Looking through scientific journals, online articles, and blog posts, you can find many individuals documenting their experiences and performing reviews of Haskell Web Frameworks. For example, in the IEEE Internet Computing journal, Collins and Beardsley have written an article giving an overview of Snap. According to the article, snap is a simple web framework written in Haskell where programming is done at a similar level of abstraction to Java servlets. The article instructs the reader on how to install Snap, walks the reader through some sample code, and shows a quick comparison between Snap and other major web frameworks. The comparison is a benchmark of each framework, recording the amount of time it takes for each framework to respond to a request. The benchmark results can be seen in Figure 2.1. (Collins & Beardsley, 2011)

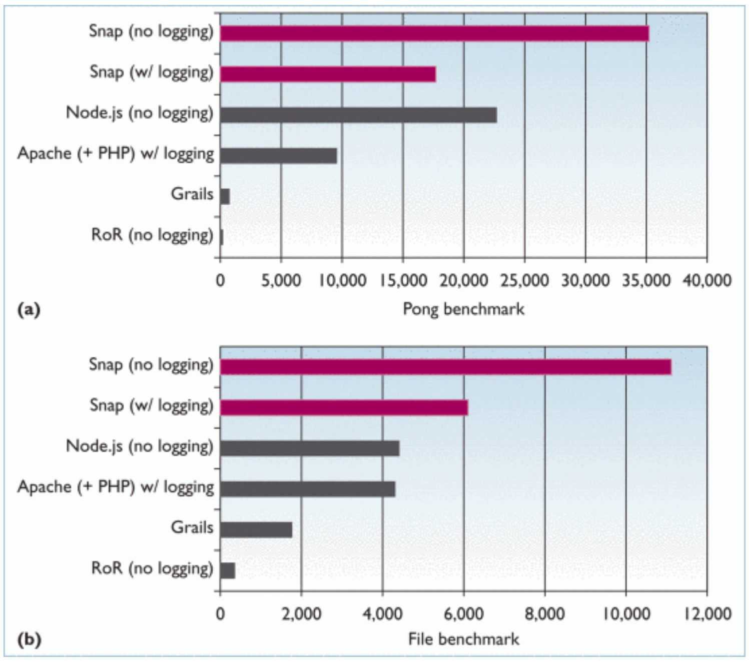

Figure 2.1.: Snap and other frameworks, values are in requests per second
© 2011 IEEE

Figure 2.1 shows the results of two benchmarks, one where a server responded to a request by sending the string "pong", and another that records how fast a server can send a 49 kilobyte image. As you can see in the graph, for the file benchmark, the Snap framework was faster than all other frameworks. Snap was only beaten by Node.js in the Pong benchmark when logging was turned on. Turning logging off dramatically increased the performance of Snap, resulting in Snap being 55% faster than Node.js in the Pong benchmark. (Collins & Beardsley, 2011)

So, from the research done by Collins and Beardsley, we can see that Haskell web frameworks can be significantly faster than more traditional web frameworks. This is because Haskell uses the Glasgow Haskell Compiler (GHC). GHC compiles Haskell programs into native machine code, ensuring high performance, especially for concurrent programs such as web servers. Because of this, any Haskell web framework that uses GHC, including Yesod, will serve requests faster than most traditional web frameworks. (Gamari, 2018)

Bajt published a blog post on his website with the title “Comparing Haskell Web Frameworks”. In the post, Bajt provides a quick comparison between some popular Haskell Web Frameworks, including Yesod and Snap. The comparisons include the installation process of each framework, the way they handle routing – i.e. pointing a URL to a piece of code that will produce a response, and the quality of documentation for each framework. (Bajt, 2014)

At the end of his comparison, Bajt found that Yesod was the best framework for his use case. One of the reasons for his choice was because of the great documentation available for Yesod. The creator of Yesod has written a book, *Haskell Programming from first principles*, that is very comprehensive. The book is available for free on the Yesod website. The amount of detail contained in this book is also one of the reasons the Yesod web framework was chosen for this project. (Bajt, 2014; Snoyman, 2012)

In “A Haskell Beginner’s Experience With Yesod”, Picciau discusses his experiences in using Yesod as a beginner to the Haskell programming language. He mentions the depth and thoroughness of the Yesod book when learning Yesod. However, when making an actual website, he came across difficulties when trying to implement features that required the use of functions not in the book. The author had to check the documentation of the functions on Hackage, a Haskell package archive. On Hackage, most functions contain a type signature and normally a one line description. The author mentions how the type signatures would probably be enough for experienced Haskell developers to work out how to use a function but, as a beginner, founding out how a function works using types was much more difficult. Because of this, the author had to spend a lot of time fixing type errors. (Picciau, 2018)

When first starting the project, I personally experienced the same issues described in Picciau’s blog post. I was not very good at reading the type signatures available on Hackage, resulting in spending a lot of time fixing unmatched type errors when trying to use functions not documented in the book. However, as I became more experienced in Yesod and Haskell, these problems became more and more rare as my understanding of Haskell type signatures increased.

2.2. Real World Haskell Sites

Some readers will be concerned about whether or not a Haskell web framework like Yesod is ready for production websites. This is a valid concern considering the relatively small amount of Haskell programmers when compared to mainstream programming languages. Readers will be pleased to know, however, that there are some high traffic sites that are built using the Yesod web framework.

Freckle, previously known as Front Row Education, is an education platform that provides a service to almost 10 million students (Alvarez, 2018). In 2015, Freckle migrated their site to the Yesod web framework and have been using it ever since. Kurilin, the CTO of Freckle, wrote an article on his experience of using Yesod for a high traffic website. (Kurilin, 2015)

In his article, Kurilin states that the reason they chose a Haskell Web Framework was because of the low resource usage and the ability to make quick iterations that the Haskell language gives you. The article also discusses how static typing saves time when writing unit tests. The developers at Freckle did not have to deal with checking for null exceptions, mismatched types, and other common bugs that are annoying to deal with. Spending less time dealing with dynamic typing gives developers more time in implementing their features. The modularity of Yesod also allows Freckle to reuse complex code, reducing potential mistakes by developers, reducing the amount of code that needs to be written, and allowing code to be updated quickly without the need to repeat changes. All of these advantages allow developers to be more efficient and write fewer bugs. (Kurilin, 2015)

However, there are some issues that the Freckle team came across during their migration. Haskell builds are normally quite slow, as all external libraries used have to be compiled during the build process. Kurilin mentions that builds took 5-10 minutes on their most powerful machines. The author does mention that the team could improve their build process by, for example, caching built files. The testing suite caused problems for the team because when a test fails, they could not determine which condition caused the failure when a test block has multiple conditions. The issue, however, was reported to developers behind the testing library and the current version of the testing suite does not have the issue mentioned in the article. The lack of documentation for some functions also caused some frustration to the development team, especially for the more junior developers who could not rely on type signatures. (Kurilin, 2015)

When Freckle switched their main API to Yesod, their CPU usage rose to 95%. This issue did not occur during testing and profiling, in fact, the Freckle team were the first to experience this particular issue. This is one issue when using a relatively niche language like Haskell, you have to be comfortable with the idea that you may be the first person to experience a particular issue. With other popular frameworks like Django, any issue you discover has most likely been found and fixed by other members of the community.

Despite these issues, Freckle decided to stick with Yesod due to the advantages of the Haskell compiler and the fact that the issues they experienced with regards to documentation and build time are improving. And although the community is small, you can almost always find help by asking on the StackOverflow or Google Groups pages or by visiting the #haskell-beginners IRC channel, an online chat room where developers new to Haskell can quickly and easily get help from some more experienced developers.

3. Preparation

In this chapter, we will discuss the work that needed to be done before work could be started on developing websites in Yesod and Django.

3.1. Planning the Website

Before any work was done, a plan was created that indicated the features the website should contain to ensure that the features of each framework are able to be fairly tested and evaluated. The website to be created was a twitter clone with the following features: a home page, authentication, a profile page, ability to post a message, ability to post ‘tagged’ messages (any words with a preceding ‘#’ turns into a link that leads to a search page), ability to search for messages and users, and an ability to follow other users. All features implemented would also have unit tests implemented using the testing tools available in each framework. After each site is feature complete, a new feature, the ability to message other users, should be implemented.

Implementing these features allows us to test page load speed by navigation to certain web pages. Safety can be tested by analysis how each framework deals with custom user input. We can evaluate how the static type checking and type safety features of Haskell affects reliability when compared to dynamic type checking in Python. Testing all of our implemented features allows us to fairly evaluate the features of the test suites built into each framework. Implementing a new feature once each site is feature complete will also allow us to test the maintainability of each framework.

By planning the website before any work was done, there was a clear vision of what the website should look like. This allowed development to focus on implementing the specified features rather than trying to create new features while developing at the same time, ensuring that functionally identical websites are created in both frameworks that can be fairly compared and evaluated.

3.2. Learning the Frameworks

Even after the planning was completed for both sites, before any development could be done, the basics of each framework must be learned. This ensures that code produced follows the latest standards of each framework, the built-in features of each framework that may help with development are understood and used appropriately, and code produced is of a high standard, maintainable, and readable.

3.2.1. Learning Django

Because of previous experience with Python and other object oriented web frameworks, Django was learned quickly by going through the official Django tutorials and documentation.

The Django tutorials themselves were straight forward for someone who has experience in Python and other web frameworks. The tutorials walk you through installing Django and creating your own app. In Django, an app resides in a project and is a web application that performs some function. An example of an app could be a web blogging system. A Django project is a collection of apps and configuration settings for a particular website. After completing the Django tutorials, work on the planned website was begun. ("Getting Started", 2018)

3.2.2. Learning Yesod

Coming from an object oriented background, it was not trivial to start using a functional programming language like Haskell. Before development with the Yesod framework could be started, the Haskell language had to be learned to an adequate level.

To help with learning Haskell, the book *Haskell Programming from first principles* (Allen & Moronuki, 2016) was used. This book walks the reader through learning the Haskell language beginning with the fundamentals. Reading through several chapters of the book gives you a basic understanding of programming with Haskell, allowing you to start learning Yesod itself and referring to the book to understand more advanced concepts that you may come across while learning Yesod.

To learn Yesod itself, the book *Developing Web Applications with Haskell and Yesod* (Snoyman, 2012), written by the person who wrote Yesod, was used. The book goes through all of the features of the Yesod framework in an easy to understand manner. After reading the book, development on the planned website using the Yesod framework began.

3.3. The Evaluation

Once the website was feature complete on both frameworks, a series of experiments were ran to compare the features that we planned to test during the planning phase. The raw data of these experiments can be found in the appendix and an evaluation of these results are discussed in chapter 5.

4. Deliverable

5. Evaluation

6. Conclusion

References

Allen, C. & Moronuki, J. (2016, July). *Haskell programming from first principles*. (Cit. on pp. 5, 8).

Alvarez, H. (2018, April 13). Say hello to freckle education! Retrieved April 22, 2018, from http://blog.freckle.com/say-hello-to-freckle-education. (Cit. on p. 5)

Bajt, A. (2014, February 23). Comparing Haskell web frameworks. Retrieved April 21, 2018, from http://www.edofic.com/posts/2014-02-23-haskell-web.html. (Cit. on pp. 4, 5)

Collins, G. & Beardsley, D. (2011, January). The Snap framework: A web toolkit for Haskell. *IEEE Internet Computing*, *15*(1), 84–87. doi:10.1109/MIC.2011.21. (Cit. on pp. 3, 4)

Gamari, B. (2018). The glasgow haskell compiler. Retrieved April 22, 2018, from https://www.haskell.org/ghc/. (Cit. on p. 4)

George, N. (2017, May). *Why Django? the Django book*. Retrieved January 19, 2018, from https://djangobook.com/tutorials/why-django/. (Cit. on p. 2)

Kurilin, A. (2015, April 25). Haskell at Front Row. Retrieved April 22, 2018, from https://github.com/commercialhaskell/commercialhaskell/blob/master/usage/frontrow.md. (Cit. on pp. 5, 6)

Getting started. (2018, January). Django Software Foundation. Retrieved April 22, 2018, from https://docs.djangoproject.com/en/2.0/intro/. (Cit. on p. 8)

Picciau, L. (2018, January 22). A Haskell beginner's experience with Yesod. Retrieved April 21, 2018, from https://itscode.red/posts/a-haskell-beginners-experiance-with-yesod/. (Cit. on p. 5)

Snoyman, M. (2012, April). *Developing web applications with Haskell and Yesod*. O'Reilly Media. (Cit. on pp. 1, 5, 8).

Bibliography

Allen, C. & Moronuki, J. (2016, July). *Haskell programming from first principles*.

Alvarez, H. (2018, April 13). Say hello to freckle education! Retrieved April 22, 2018, from http://blog.freckle.com/say-hello-to-freckle-education

Bajt, A. (2014, February 23). Comparing Haskell web frameworks. Retrieved April 21, 2018, from http://www.edofic.com/posts/2014-02-23-haskell-web.html

Collins, G. & Beardsley, D. (2011, January). The Snap framework: A web toolkit for Haskell. *IEEE Internet Computing*, *15*(1), 84–87. doi:10.1109/MIC.2011.21

Gamari, B. (2018). The glasgow haskell compiler. Retrieved April 22, 2018, from https://www.haskell.org/ghc/

George, N. (2017, May). *Why Django? the Django book*. Retrieved January 19, 2018, from https://djangobook.com/tutorials/why-django/

Kurilin, A. (2015, April 25). Haskell at Front Row. Retrieved April 22, 2018, from https://github.com/commercialhaskell/commercialhaskell/blob/master/usage/frontrow.md

Django Documentation. (2018). Retrieved April 21, 2018, from https://docs.djangoproject.com/en/2.0/

Getting started. (2018, January). Django Software Foundation. Retrieved April 22, 2018, from https://docs.djangoproject.com/en/2.0/intro/

Hackage. (2018). Retrieved April 20, 2018, from https://hackage.haskell.org/

Haskell Wiki. (2018). Retrieved April 21, 2018, from https://wiki.haskell.org/

Hoogle. (2018). Retrieved April 20, 2018, from https://www.haskell.org/hoogle/

Stackage. (2018). Retrieved April 20, 2018, from https://www.stackage.org/

The Haskell Tool Stack. (2018). Retrieved April 21, 2018, from https://docs.haskellstack.org/en/stable/README/

Yesod Cookbook. (2018). Retrieved April 21, 2018, from https://github.com/yesodweb/yesod-cookbook

Yesod Google Group. (2018). Retrieved April 21, 2018, from https://groups.google.com/forum/#!forum/yesodweb

Yesod StackOverflow Page. (2018). Retrieved April 21, 2018, from https://stackoverflow.com/questions/tagged/yesod

Picciau, L. (2018, January 22). A Haskell beginner's experience with Yesod. Retrieved April 21, 2018, from https://itscode.red/posts/a-haskell-beginners-experiance-with-yesod/

Snoyman, M. [Micahel]. (2018). Yesod git repository. Retrieved April 21, 2018, from https://github.com/yesodweb/yesod

Snoyman, M. [Michael]. (2012, April). *Developing web applications with Haskell and Yesod*. O'Reilly Media.

Snoyman, M. [Michael]. (2018). Yesod web framework for Haskell. Retrieved January 19, 2018, from https://www.yesodweb.com/

Staub, C. (2011). *A user interface for interactive security protocol design* (Diploma Thesis, Eidgenössische Technische Hochschule Zürich, Department of Computer Science). doi:10.3929/ethz-a-007554462

Watson, J. (2018, December 8). How does the Yesod web framework compare to more mature frameworks such as Rails and Django? Retrieved April 21, 2018, from https://www.quora.com/How-does-the-Yesod-web-framework-compare-to-more-mature-frameworks-such-as-Rails-and-Django-Is-Yesod-stable-enough-to-develop-a-production-website-What-do-you-lose-by-going-with-Yesod-rather-than-Rails-or-Django-What-do-you-gain

Appendices

A. Project Diary

A.1. Meeting 1 - 3rd October 2017

A.1.1. Meeting Notes:

Books

Real World Haskell
Haskell from first principles (haskellbook.com)
Web application development with Haskell and Yesod (out of date)

Frameworks / Tools

Haskell Servant package
Snap is alternative to Yesod
ghcjs haskell to js
haskell stack tool
hackage is like npm. Stack can use hackage.
Stackage is like stack on top of hackage
Use the latest LTS version of haskell from stackage
Atom could be useful with their plugins, compare with plugins available for code
ghc-mod available for haskell in atom, helpful when developing
ide-haskell, linter
There is a Haskell plugin for intellij which may work. Good because I would be familiar with the IDE.

Comparing the two frameworks

- Maintainability
  - Make a change to both
- Performance
- Scalability - could use tools, hard to do on your own
- People say Haskell is easier to write code with, less time debugging, once learnt
  - We could test this. How much the type checking helps. The different tools available
  - Can't use line by line debugging

Plan for next meeting

Do as much as possible for now
Come up with rough project definition form
Go through some haskell tutorials, haskellbook.com is recommended

A.2. Meeting 2 - 12th October 2017

A.2.1. Meeting Notes:

Look into getting GHC mod compile on save
Get the project proposal doc ready for next week
Learn Django and get it installed on the laptop
Make a basic page in Django and Haskell

A.3. Meeting 3 - 19th October 2017

A.3.1. Meeting Notes:

Carry on with the Haskell Programming from First principles book
Have some planning for the twitter clone ready

A.4. Meeting 4 - 24th October 2017

A.4.1. Meeting Notes:

Set up a basic homepage in Yesod and Django. Do this over the weekend.
Have a play around with the yesod site that's provided to see what you can focus on.
Carry on with the book
Setup Docker/Vagrant if you have time at the end, for instructions on setting up the repo
Topics important for yesod

- Quasi quotes, provided by yesod
- Yesod Typeclass could be useful to know

A.5. Meeting 5 - 10th November 2017

A.5.1. Meeting Notes:

I've created the homepages in both yesod and django. I've used tests in django to test a basic app not related to the project

Next week, I want to ensure both home pages are the same and to create tests in both frameworks. I want to progress more through the yesod and haskell book. Create User models in both yesod and django and create tests for them.

A.6. Meeting 6 - 16th November 2017

A.6.1. Meeting Notes:

I’ve created the homepages in yesod and django and ensured that they both have the same content and styling.

For django, I have added the functionality to allow users to create accounts and log in. I have added unit tests for this and they all pass.

For yesod, I have added the latest version of jquery and bootstrap to the project. I have tried to complete the user account functionality but I am blocked. I am trying to import yesod-auth-hashdb but cannot figure out how to do it. There is some documentation showing how to edit the cabal file but this is overwritten during the build, I believe the data comes from package.yml. Editing package.yml causes strange errors when I try to build the project but I don’t think I am doing it in the correct manner. Need to figure out how to edit the package.yml, edits would result in errors on my computer.

For next week, I want to fix the weird error and get some tests up.

Things to try to resolve the error, try to reproduce it on normal ubuntu. If you can’t resolve it, report it to yesod.

A.7. Meeting 7 - 23rd November 2017

A.7.1. Meeting Notes:

I’ve resolved the random error we had last week.
I’ve imported hashdb and have added functionality for users to create accounts and login on the yesod site.
Yesod forms rely on bootstrap 3, so downgraded from bootstrap 4 (beta) to 3.

For next time...
I want to figure out how to concatenate a Text data variable in Yesod. Have to figure out how to deal with overloaded strings?
Finish the user authentication functionality. Show appropriate messages and add extra validation to the yesod form (unique user and email, min and max length of fields).

Create tests for the user authentication functionality.
Change the forms on Django to use their form model rather than a HTML form. This will let me compare the pros and cons of Django's and Yesod's forms.
If there is time, add functionality to allow users to post messages. These messages should be saved in the database so that the user can see all the messages they've posted when they log in.

The user post message page should use ajax so when they post a message, the part of the div will just reload rather than the whole page.

A.8. Meeting 8 - 14th December 2017

A.8.1. Meeting Notes:

On the yesod site:
Have some tests working
Users can post messages, be signed up, see other users messages
Have some tests working, this is WIP

For next time…
Get Django messages working
Try to get ajax working on both sites, see https://www.yesodweb.com/blog/2013/02/ajax-with-scaffold

Interim report plan

- Intro
- Explain the choices of yesod and django
- Do some initial comparisons of the site
- My experiences with developing on both sites, what I found easy and hard on the different frameworks.
- Advantages and disadvantages of both frameworks.

A.9. Meeting 9 - 1st February 2018

A.9.1. Meeting Notes:

Worked mainly on the Django site. I have the messages working and have began comparing features between two sites such as

- The implementation of Handlers/Routes
- The way you can pass variables to templates
- How Haskell's 'maybe' reduces the number of errors you need to catch
- the ways you can implement AJAX in both frameworks

In the near future, refactor the messages implementation to use AJAX for retrieval of messages and creating new messages. This refactoring will help compare the ease of modifiability of both of these frameworks.
Whenever you come across a difficult error, try to compare the process of debugging in both frameworks.
Remember to focus on using different parts of the framework than just implementing new features on the site.
Try to resolve the textarea problem. If you can't send a screenshot of the error.

A.10. Meeting 10 - 15th February 2018

A.10.1. Meeting Notes:

Created AJAX functionality for getting messages

Resolved issue with using single template for profile by declaring the form stuff even if isCurrentUse is false, this is fine

What needs to be done

Add more tests this weekend for both frameworks. Does Haskell's type checking mean we need fewer tests compared to Python?

What to look at for evaluation

Evaluate:

- The ease of writing tests
- In python, you need lots of testing because there's no static type checking
  - Does this mean you need less tests in Haskell
  - Does this mean tests are easier to write in Haskell, or in Python because tests are more important in Python so they'd be easier to use
- What types of tests are important in the haskell world
- Some tests are unneeded for Haskell
- Is it cheaper to build a bullet proof app in Haskell or Python, maintainability? etc

Amount of users in Django makes it easier to find problems that others have experineced

Amount of users in Django means more tools for Django but this is improving on the Haskell on the side

The Yesod book written by the creator of Yesod is pretty good

Some of the problems written by people using Yesod are more detailed, users are probably more experienced in the programming world? more academic?

Evaluate the ease of adding a new feature after the site is complete

Evaluate how quick it is to debug something

Built in compiler and type checking in Haskell is very useful when debugging

Evaluate page load speeds, is Python slower because it runs the interpreter every time?

Scalability if you can

Some quantitative data?

Explain to the reader why some things make a big difference

- How long it takes to get a reliable application
  - Length of tests? Number of tests? Instances where types catch important errors? These are very useful
  - Times where the type checking or other similar features got in your way?
    - E.g. profile page was easier to code in Python, this was because of Haskell checking the scoping

A.11. Meeting 11 - 22nd March 2018

A.11.1. Meeting Notes:

Implemented type safety for some URLs

- How much does it help with testing / debugging?
- Does this reduce the amount of code you need (to check parameter types, etc.)

Joins cannot be done with Yesod Persistent (alternative pseudo SQL available) but restructuring logic may be more efficient than joins.

Plan for the holidays and beyond

- Finish the functionality of both websites by the end of week 2 of the holidays
- Produce a report by the end of week 3 of the holidays

- Get feedback and produce another version of the report the first week back from the holidays
- Get more feedback and produce a final version of the report the second week after the holidays

A.12. Meeting 12 - 19th April 2018

A.12.1. Meeting Notes:

Report draft, hand in by this weekend.

Book in a meeting on Tuesday to discuss the report.

Simulate mistakes and see how the haskell compilers help

Argue against how the compiler may make you write unnecessary code

- Haskell version, I couldn't do something like if not null, ignore the block
- You can try to use an undefined and force an exception
- You can do something like "Error - (loc) shouldn't happen"

Type safety saved a lot of time with checking variable types

But people like python because of the lack of type checking

It may be faster to ignore types when developing but it could cause problems later on

In Yesod, there's types for entity Ids, and if the entity Id does not exist, it will 404

Think about seperating language vs framework

For this report, it's more useful to focus on the framework, but think about anything to do with the language

For example, static type checking is to do with Haskell rather than Yesod

In the yesod-test package, you can use HTML selectors. That makes testing static content easy.

Django, you can test HTML page but selectors are not implemented. (If html page contains this string).

If you have time after getting the report done...

Load testing, deploy both apps to the same environment. Is there a tool for it. Send multiple queries from multiple machines, time responses.

If no tool, you can try requests from just your own machine. If you can't get it on Heroku, set up a VM.

Test the ease of deployments for both frameworks. Test the actual production mode, not dev mode.

A.13. Project Definition Form

A.13.1. 14th October 2017

First draught written up and sent to tutor via email for feedback

A.13.2. 15th October 2017

Tutor feedback implemented

A.13.3. 19th October 2017

Tutor and I signed form. Form is submitted electronically via Turnitin

A.14. Interim Report

A.14.1. 18th January 2018

Interim report started and then finished

A.14.2. 19th January 2018

Sent Ethics form to tutor

Proof-read and submitted the interim report

A.15. Software Development

This section records the development lifecycle of the Django and Yesod sites. More detailed commit messages can be found by running a *git log* on the git repo for both sites.

A.15.1. Yesod Site

1st November 2017

Created a basic homepage and a vagrantfile.

23rd November 2017

Implemented Login and Signup functionality

11th December 2017

Created a profile page

14th December 2017

Implemented message creation functionality and added some tests

15th February 2018

Modified profile page to retrieve messages using AJAX

19th March 2018

Implemented functionality to show other registered users on the profile page

20th March 2018

Basic functionality to follow users implemented

21st March 2018

Refactor profle page and follow functionality to use type safe URLs

22nd March 2018

Added functionality to allow users to follow eachother through normal usage of the site. Previously, users had to type in a URL to follow a user

31st March 2018

Refactored profile page. Separated pages used for logged in and anonymous users, reducing complexity. Refactoring also increased use of type safe URLs in the profile page and when posting messages.

1st April 2018

Updated stack resolver to latest point release for current resolver, ensuring we have the latest compatible updates for our packages. Fix some bugs and add tests.

2nd April 2018

Added tests for all route handlers and integrate tests with Travis CI.

4th April 2018

Implemented search functionality

5th April 2018

Tested the search functionality

6th April 2018

Added latest posted messages to the homepage and used a previously created datatype to help display posted messages on the frontend.

7th April 2018

Added hashtag functionality and fixed some bugs on the profile page. Hashtagged messages appear on the homepage.

10th April 2018

Added yesod prefix to database names so that the Yesod server can be ran the same time as the Django server.

11th April 2018

Removed unneeded visit button on profile page.

13th April 2018

Improved ordering of messages posted by other uses on your feed.

A.15.2. Django Site

9th November 2017

Created the homepage and a vagrantfile.

14th November 2017

Implemented login and signup functionality

15th November 2017

Refactored authentication functionality to use Django's built in models rather than custom models.

31st January 2018

Implemented profile page and message creation functionality.

10th April 2018

Downgrade Django bootstrap version to bootstrap 3. This is the version used by Yesod so using version 3 reduces the amount of template work to do which does not provide much insight into the benefits of either framework.

Implemented breadcrumbs and a base template file that all other template files use.

11th April 2018

Added AJAX functionality to the profile page.

12th April 2018

Completed the profile page and added search functionality

13th April 2018

Added tests and fixed any bugs discovered. Latest message and hashtagged messages added to the homepage.

A.16. Project Diary Final Report

A.16.1. 20th April 2018

Started write-up of the final report

A.16.2. 22nd April 2018

Completed report and sent to tutor for feedback

B. Ethics Form

Ethics form for student projects

SEAS group: Computer Science

Project title: Evaluation of a Haskell Web Framework

Supervisor name and email: Michal Konecny m.konecny@aston.ac.uk

Ethics questions

Please answer Yes or No to each of the following four questions:

1 - Does the project involve participants selected because of their links with the NHS/clinical practice or because of their professional roles within the NHS/clinical practice, or does the research take place within the NHS/clinical practice, or involve the use of video footage or other materials concerning patients involved in any kind of clinical practice? ***No***

2 - Does the project involve any i) clinical procedures or ii) physical intervention or iii) penetration of the participant's body or iv) prescription of compounds additional to normal diet or other dietary manipulation/supplementation or v) collection of bodily secretions or vi) involve human tissue which comes within the Human Tissue Act? (eg surgical operations; taking body samples including blood and DNA; exposure to ionizing or other radiation; exposure to sound light or radio waves; psychophysiological procedures such as fMRI, MEG, TMS, EEG, ECG, exercise and stress procedures; administration of any chemical substances)? ***No***

3 - Having re ected upon the ethical implications of the project and/or its potential ndings, do you believe that that the research could be a matter of public controversy or have a negative impact on the reputation/standing of Aston University? ***No***

4 - Does the project involve interaction with or the observation of human beings, either directly or remotely (eg via CCTV or internet), including surveys, questionnaires, interviews, blogs, etc? *Answer "no" if you are only asking adults to rate or review a product that has no upsetting or controversial content, you are not requesting any personal information, and the adults are Aston employees, students, or your own friends.* ***No***

Student's signature: Junaid Rasheed

Supervisor's signature: Michal Konečný M.K.