CS795 Functional Programming

Functional Web Development Project 2

Marie-Louise Steenkamp 20722796
Hendrik van Heerden 20916892
Kieren Sinclair 19989059
Christoff Van Zyl 20072015

Introduction

The purpose of this project was to allow students to experience the shift in mindset and skills needed to move from everyday programming languages and frameworks to functional programming languages and frameworks. The goal of the project was to create or significantly modify a web site using functional programming languages and frameworks. The project was completed in a group of four students and implemented and documented over the period of less than a month. Our experience of this project as well as some details about our web site are mentioned in the sections below.

About our chosen stack

For this project we designed and created a classifieds website. We decided to use IHP as the development stack to create this website.
IHP is a self contained web development stack. IHP uses Haskell as its main development language for creating web sites and makes use of a standard sql database. IHP also uses nix as the package manager to ensure all needed dependencies are present. One of the main reasons that we decided to use IHP is because IHP was designed to be able to rapidly develop websites with minimal prior knowledge of Haskell. This seemed convenient, since we are relatively new to Haskell and the concept of functional programming as a whole. IHP achieves this convenience by providing a browser-based interface that is used to generate skeleton code for many different situations and it also allows you to build your sql database using this interface.
Another helpful aspect of IHP is that it has a live recompile. This was extremely helpful while coding and adapting to using Haskell for web development. The way the code is structured is also designed to make coding without prior knowledge easier.

Structure of the stack

IHP uses the well known model-view-controller structure. The previously mentioned ability to generate skeleton code can be used to create either a new view or controller, quickly and easily. In the case of generating a controller, for example, IHP creates a set of common database actions, then it creates a new view for the controller to affect and it makes entries in the routes file to allow navigating to the new view.
Each controller is made up of actions, which can be called by any view to perform some action such as updating the HTML or making a change to the database. Conveniently, the generated code for a controller includes the most common actions for a table in a database, namely the ability to create a new entry, update an entry, delete an entry and finally get the whole table.

Below you can see an example of a simple action that, when called, fetches the requested users entry in the database and updates the current page to display the users information.

```
action ShowUserAction { userId } = do
    user <- fetch userId
    render ShowView { .. }
```

Routing

When a new controller is generated all of its actions are automatically routed to their own unique urls.So if a NewListingAction is called you are automatically sent to a view that contains the needed form to create a new listing.

Default HTML layouts

IHP has the ability to set a default layout that is used across all pages and create other additional layouts that can be used on multiple pages. IHP does this by creating a type called “Layout”. This type takes in a View which is made up of HTML code, the type then modifies it and then returns the new View that is to be used as the base layout of any given page. For our Site the default layout was modified to give the site a different look to the original Blog that is used as an example of a functioning website which made use of the default layout.

Updating a Page

Updating the HTML of a page is quite simple. When a change is made to what is currently being shown on a page a simple function is used to update the page. An example of a situation that would require this, is when you search for a specific listing, the page has to be updated to only show the listings that meet the search criteria.

The function that achieves this is called render. Render takes in a view and updates the HTML to reflect any changes made.

```
action HomeAction = do
    listings <- query @Listing
        |> orderByDesc #createdAt
        |> fetch
    render HomeView { .. }
```

Above is an example of render being used. Here all the listings are fetched from the database. Then when the render function is called the HTML of the home view is updated to display all of the listings that are now stored in the listings variable.

Functionality of the site

- Register users
  New users are able to register. This allows new users to be able to log into the site and post listings. If a user is not registered they are still able to search through all available listings.
- Login
  Users, once registered for the website, are able to log in and then, when logged in, create new listings. Users are also prompted if they enter a username that is not in the database or if they have entered the incorrect password.
- Listings
  All listings on the site are visible to all users. Once a user has registered they are able to edit and delete any listings that were created by their account. A user that is not registered is not allowed to create a listing.
- Search
  Any user, registered or not, is able to search through all available listings. This is done from the home page and allows them to only view listings that they are interested in.

Our personal experience of using functional programing for web development

The Haskell programming language:

Learning to successfully code in Haskell was, and still is, a challenge. To code in any functional programming language requires the coder to understand, and be able to apply, multiple functional programming concepts. Having this knowledge about functional programming concepts is the basis of becoming a good functional programmer.

The IHP framework:

While it was intimidating at first, to make a website using a whole new style of coding and a language we had only recently picked up, once we started to get the hang of it, it got easier.
While IHP had many useful features that assisted in adapting to a functional based development stack there were many aspects that made it very difficult to learn how to code in the framework. One of the biggest problems is that the guide to IHP that is provided on their website has many problems. It is incomplete and does not cover all topics. There are sections that only contain “TODO” prompts that are still incomplete.

For example, view the following screenshot:

Above is a screenshot of the page that is meant to help you debug any errors that occur. This is a rather important part when trying to code anything using IHP. This incomplete section made using the framework much harder.
Some of their examples that try to explain how to achieve something in the framework often turns out to be too specific and not general enough to make it easy to then apply that knowledge in a use case that differs from the given one.
Often when going through the guide it does not give context for why something is being done or in what file the changes are being made, but rather it is left to the users to try figure it out themselves.
IHP helped a with adapting to their structure by providing lots of help generating the base code for any additions however as mentioned previously making certain changes to this generated code could be difficult and tedious. The way the code is separated into views which are acted on by each controller's actions was a simple and powerful way of separating the code to make adding any new features simple to understand.

Working as a team on a remote functional programming project:

Having to code, present and write a report on a new programming language, style and framework had some challenges. Firstly, collaborative coding was hard to accomplish as using the functional programming concepts, which are new to all group members, can be tricky to master. Using function based coding requires all group members to be aware of function structures, inputs, and outputs. This was a challenge, but we managed to collaborate well in the end. Writing reports and making presentations was easily managed by using online platforms for collaborative writing such as Google Docs and Overleaf. As for presenting, we divided the presentation up into equal parts and decided who was presenting in which order and this allowed each member to prepare individually.

Conclusion

Overall, this project was very challenging and time-consuming, but it was a very good learning experience. Our site was completed in the allocated time, with most of the basic functionality in place. All group members worked equally as hard and contributed an equal amount of time to the project. This project prepared us as students for future functional programming and challenges to come.