CS 491/691 - Full Stack Web Development

Keith Lancaster, Ph.D.

Course Overview

- This course is intended to provide an introduction to full-stack web application development.
- We will focus on the tools and techniques required to create database-backed, multi-page dynamic applications.
- The web application framework chosen for the course is Phoenix, a relatively new framework built using the Elixir programming language
- Elixir is a functional language that runs on the Erlang VM.

As this is a new class, there are no formal prerequisites. Since the backgrounds of those in the class vary, we will cover the basics on most topics (for example, SQL, functional programming, database design, ...).

Lecture Format

- The class is 3 hours long, so there will be a 10 minute break around an hour or so after the start of class.
- In general, the first part of the class will include a short on-paper quiz, discussion, and introduction of the topic of the week. Slides, when used, will be provided in advance.
- The latter half of the class, again in general, will be in-class work that will be turned in at the end of class.

A Partial List of Topics

- HTTP fundamentals
- Web application architectures
  - The MVC design pattern
- Web frameworks
- Review / intro to HTML, JavaScript, and CSS
- CSS and JavaScript frameworks
- Design and development strategies for web applications
- Functional programming using Elixir
- Elixir, OTP, and distributed parallel systems
- Phoenix framework architecture
- LiveView and single-page applications

And now, the Syllabus

https://tinyurl.com/yckcf2nx

Why Phoenix and Elixir?

- Elixir is built upon the Erlang VM called BEAM
  - Erlang systems are extremely robust, with some applications in operation for over two decades without a reboot
  - Elixir compiles to Erlang, providing a much friendlier syntax that is heavily influenced by Ruby
  - An Elixir application can launch 10s of thousands of BEAM processes that run in parallel and can take advantage of all the cores on a machine, unlike most other stems
  - Due to the use of the actor model, developing concurrent systems is a breeze compared to many other languages

Why Phoenix and Elixir? (2)

- Phoenix was developed from the ground up to take advantage of the power of Elixir and BEAM
  - Applications built in Rails, for example, require additional machines to scale
  - Many report being able to move from 10 or more machines running Rails down to one machine running Elixir and Phoenix

Why Phoenix and Elixir? (3)

- Web development is changing
  - In the beginning, you could make 10k just doing a static HTML page
  - Building db-backed web applications was *painful*
  - Frameworks came along made building web apps much easier (Rails, etc)
- Where are we now?
  - Web builder apps can be used to quickly create CMS systems and complex apps *without* coding
  - *Opinion*: Basic web apps are a commodity. Where there is still a need is for languages, frameworks, and developers that can build highly concurrent systems capable of handling huge numbers of users and capable of processing large amounts of data.
    - Making a living building custom CMS systems for small companies is, again *IMO*, on the way out.

The Development Environment

- Elixir and Phoenix
  - We cannot install the system on the lab computers, so USB sticks will be provided with all the software you need. The OS used is Ubuntu
  - You can of course use your own laptop in class if you prefer. If you are on Windows, it is strongly recommended that you use a Linux VM or Docker to run the software.
  - Installing Elixir requires installing the Erlang/OTP. On macOS you can use brew or asdf. On other systems, see https://elixir-lang.org/install.html
    - *There is no need to run a Linux VM on macOS*
  - We will be making use of https://livebook.dev/ for both lecture and in-class exercises, so you will need to create a *free* account. You can run it in the cloud environment or you can download for Mac or Windows.
    - There are instructions on the site for installation on Linux/Docker

The Development Environment(2)

- Database
  - You will need to install PostgreSQL 14 or greater on your development machine. You can find instruction at https://www.postgresql.org/download/
  - For macOS users, the easiest approach is to install Postgresapp from postgresapp.com.
- Editor
  - You can use any editor you wish. Visual Studio Code has good support for elixir, as does Sublime Text, Vim, and Emacs.
  - You can use Github Copilot if you wish, however, you need to understand the Elixir code you are asked to write! As a practical matter, Copilot sometimes gets confused and recommends Ruby code for Elixir, leading to code that will not compile. (Example: Copilot might recommend using a `while` loop, but Elixir does not have loops!)

A Short Demo of Phoenix

- Phoenix Demo
  - LiveView app creation
  - Running using iex
  - Adding validations
  - Connecting via LiveView
  - Looking at the Erlang process using `:observer.start`