FAQ for assignment 1

assignment #assignment1-faq

James L. Parry
B.C. Institute of Technology

Overall Assignment Goals

The purpose of the assignments, collectively, is to let you apply the techniques from the lessons, tutorials and labs.

In teams of up to four, you will be building a small but complete webapp, in three stages, with specific expectations for each stage. Each stage will be a separate assignment.

The framework and a number of supporting technologies have been pre-decided. The techniques for applying these will be the subject of upcoming lessons and labs.

Revisions?

If assignment details need clarifying, such changes will be made in this slide deck, and a note added to this page.

- Planning
- Design
- Process
- FAQ

Goals for This Assignment

The purpose of this assignment 1 is to get your webapp proposal together and approved, and then to get the basic pieces in the correct places expected for a CodeIgniter webapp.

This may sound suspiciously similar to lab #2! Hmmmmm.

Your Webapp

Each team will propose a webapp to build for their assignments.

Your webapp doesn't have to be "world-class" or even necessarily "real", but it needs sufficient complexity to be dynamic, personalized and scaleable/integrated.

Practically speaking, this means that your data will come partly from a database with at least two tables, and partly from a rich XML data structure, which might be stored as separate XML documents, or which might be stored as a column in one of your RDB tables.

What Could I Possibly Build?

Four typical webapps come to mind, as suggestions:

- A Mini-CMS (content management system)
- A simple order entry system
- A destination tourist guide
- A sports team fansite

You are welcome to propose your own ideas, too.

Possiblity #1: Mini-CMS

A mini-CMS would be a webapp where the primary role of your code/logic would be to make it easy for a content creator to create and edit rich content on a variety of topics, with the content posts presented and searchable in some structured fashion.

Examples of such a mini-CMS could be a blog, an informational site, or even a small forum.

The rich XML content could be the actual post contents.

Possiblity #2: Order Entry

An order entry webapp would present a simple menu, and capture/validate customer order details.

Such a webapp could be built for a real/hypothetical coffee shop, fast food oultet, or even a Ferrari dealership (collect all the colors, hehe).

The rich XML content could be the order details themselves.

Possiblity #3: Destination Guide

A destination guide is another suitable webapp. This might present attraction details for a chosen destination (country, planet or neighborhood), broken down into suitable categories (eat, sleep, play).

We pursued this theme in the fall of 2014, with some of teh destinations being Mykonos, Coruscant, and Ucleuelet.

The rich XML content would be the attraction descriptons, with different data for each category.

Possiblity #4: Sports Team Fansite

Such a site might present the team (roster), information about the league they play in, and then their schedule for upcoming games. The webapp would provide for updating the schedule with results, as games are played.

This would only work for a "real" team, and one whose season overlaps the course.

The rich XML content would likely be the schedule/results data.

Your Data

Whatever you choose to build for a webapp, you want to make sure that the data you plan is reasonable for the webapp, and reasonable for your team.

A handy rule of thumb would be to plan a database with at least as many tables as you have team members. This count would exclude any tables for sessions or user management.

Your XML data would be in addition to this (or in some cases inside one of your tables).

Keep your data plan/design simple!

Your Presentation

Your webapp will be evaluated on its functionality, not on how good it looks. It is better to have a consistent design, with elements appropriate to the purpose, than it is to have a pretty design.

Your job is to design and implement the back-end of such a site. That doesn't mean your site's appearance should be ugly or cringe-worthy - there are many freely available website templates online.

Your Usecases

Your webapp should have four broad usecases: three that each provide a different perspective or manipulation of your data, and a maintenance/admin one to add/update your data. This first assignment only need worry about the first three - the maintenance will be added in assignment 2, and it will be role-restricted in assignment 3.

In the case of a mini-CMS, your usecases might present two different perspectives on your data and a search function. In the case of a destination guide, they might present three kinds of attractions, with different kinds of information for each.

Each of your usecases would present a suitably ordered list of some sort, with the ability to select an item in the list to drill down for details on that item.

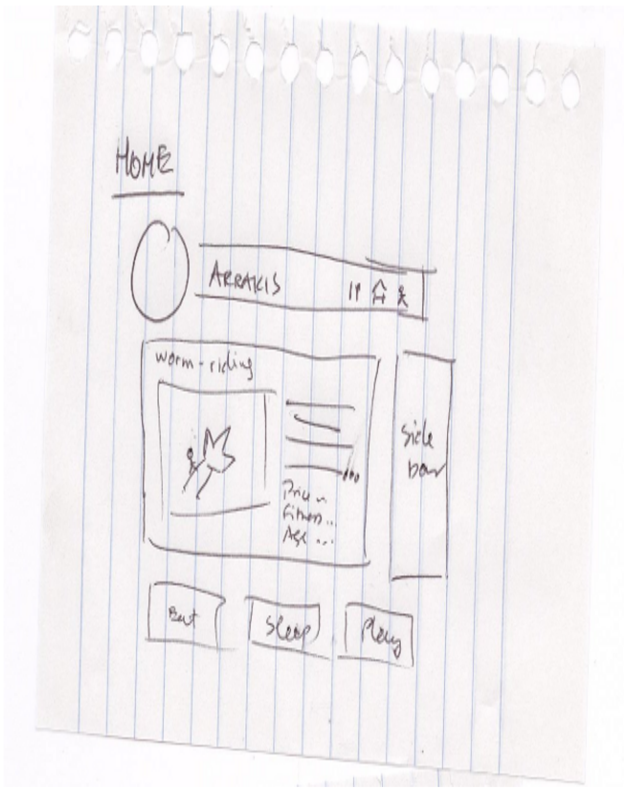
Your Sitemap

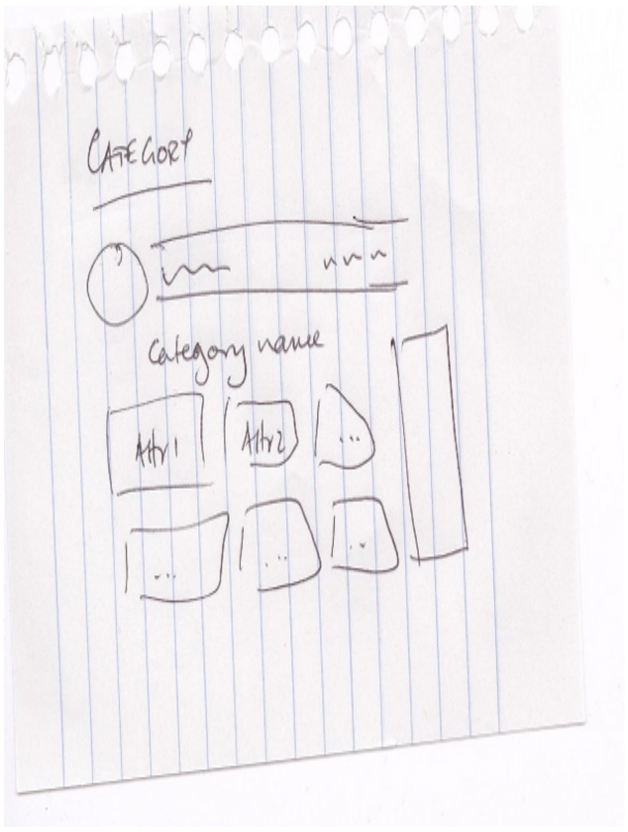
Your homepage should should provide some useful summary information (eg. most recent post, or current season statistics), as well as an elegant and obvious way to reach the three "main" pages.

If you are planning to use your webapp as part of a portfolio, I suggest adding an "about" page, where you can describe your design or your process.

A Sample Webapp

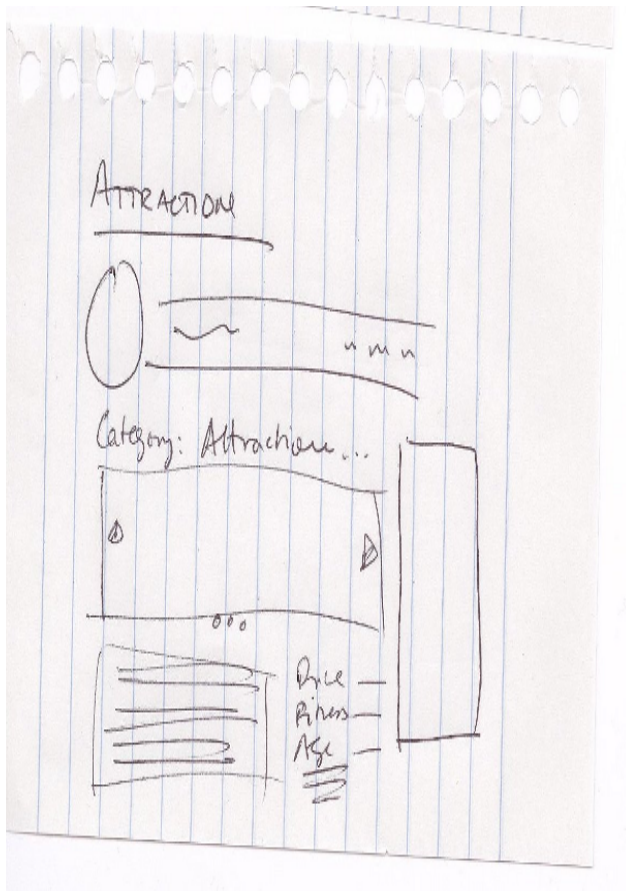
Here are the first two "napkin" sketches for a sample destination guide to Arrakis...

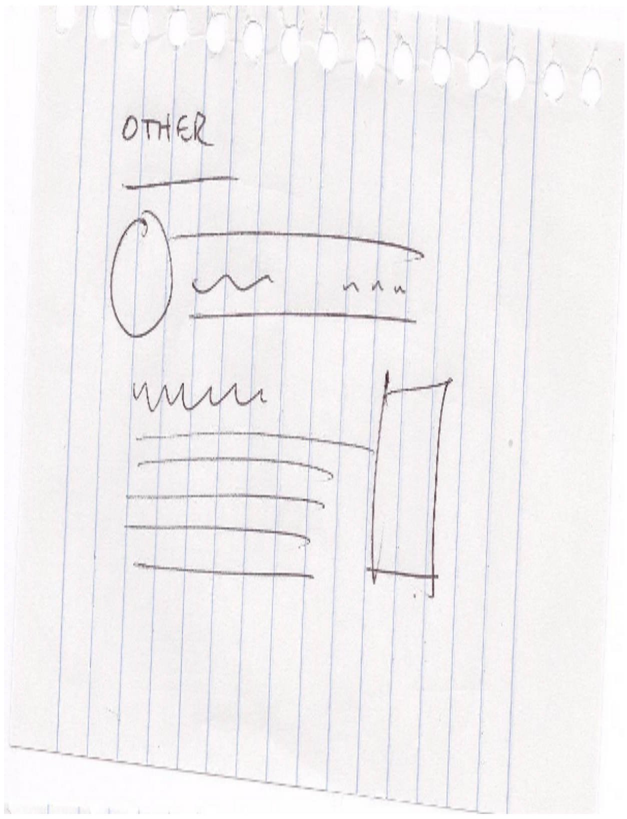




A Sample Webapp (Part 2)

And here are the last two sketches...





Webapp Stages

Your webapp will be built in three stages, each corresponding to an assignment.

Stage 1: basic webapp. The pieces will be there, in the right place, but the data and interaction are largely bogus. You might find it easier to mock the data in this stage, rather than providing a "real" database.

Stage 2: make the data "real", the usecases functional, and add maintenance.

Stage 3: Add the rich data and its maintenance, as well as role-based access control.

Your Design

Run your webapp proposal and design by your instructor, before embarking on its implementation. You want to make sure that the scope and usecases are appropriate, and that you haven't over-simplified or gotten carried away.

Typical design components include simple usecase writeups and wireframe layout(s) depicting how you see the site coming together.

This is *not* a formal design!

Process

Create a github organization for your team, in which you will then create a shared repository for your project. This repository will be used for all three stages.

Each team member should fork the project repo, and follow the gitflow workflow process for adding to or updating it. You may wish to give one team member maintainer responsibility for the repo, or you can all have it.

I would expect comparable contributions from each team member, whether you measure that by quantity or quality. If one team member does the bulk of the work for a given assignment, they will get the bulk of the marks for that assignment!

Submission

Dayschool students: submit a readme to the D2L assignment dropbox. This readme should have a link to your github repository.

Distance students: send me an email with a link to your github repository.

Due: Saturday, Jan 31, 23:59 PST

Evaluation

Assignments will be evaluated out of 20, according to the following breakdown:

- Design proposal/writeup (3)
- Controllers, home plus three (2)
- Models, with mock data (3)
- Views, template plus 4 layouts (3)
- Appropriate & consistent UI (2)
- Works properly? (3)
- Workflow process (2)
- Comments & programming style (2)

Your Code

You are programmers, and you want to be professional. Code like it.

That means clearly written and formatted code, properly commented.

Your views should have no PHP in them, apart from possibly comments.

Remember the golden rules!!!

Cautions

Don't get carried away, spending days coming up with “perfect” design or content, etc.

This is a course assignment, not a job, and not an industry-sponsored project. It is a vehicle to learn how to use CodeIgniter to build a simple webapp, incorporating good practices.

F.A.Q.

There are no questions yet, but I am sure there will be!

Congratulations!

You have completed assignment #assignment1-faq: FAQ for assignment 1

If you would take a minute to provide some feedback, we would appreciate it!

The next activity in sequence is: lesson05 Views

You can use your browser's back button to return to the page you were on before starting this activity, or you can jump directly to the course homepage, organizer, or reference page.