Design Plan Stream 3

PROJECT BRIEF

Now for the final project you're going to have to build a web app that will include as much of the topics that we've covered throughout the three streams of the fullstack course as possible.

PROJECT GUIDELINES

- Build a web app that fulfills some actual (or imagined) real-world need. This can be of your own choosing and may be domain specific.
- The project must be a brand new Django project, composed of multiple apps (an app for each reusable component in your project).
- The project should include an authentication mechanism, allowing a user to register and log in, and there should be a good reason as to why the users would need to do so. E.g. a user would have to register in order to persist their shopping cart between sessions (otherwise it would be lost).
- At least one of your Django apps should contain some kind of e-commerce functionality using Stripe and/or Paypal. This may be a shopping, or subscriptions, or single payments, etc.
- Include at least one form with validation, that will allow users to create and edit models in the backend (in addition to the authentication mechanism).
- The project will need to connect to an SQL database using Django's ORM, or to a Document-Oriented database (e.g. MongoDB) using pymongo.
- The UI should be responsive, use either media queries or a responsive framework such as Bootstrap to make sure that the site looks well on all commonly-used devices.
- As well as having a responsive UI, the app should have a great user experience.
- The frontend should contain some JavaScript logic to enhance the user experience.
- Whenever relevant, the backend should integrate with third-party Python/Django packages, such as Disqus, Django Rest Framework, etc. Strive to choose the best tool for each purpose and avoid reinventing the wheel, unless your version of the wheel is shinier (and if so, consider also releasing your wheel as a standalone open source project).
- Make sure to test your project extensively. In particular, make sure that no unhandled exceptions are visible to the users, in any circumstances. Use automated Django tests wherever possible. For your JavaScript code, consider using Jasmine tests.
- Write a README.md file for your project (in Markdown format) that explains what the project does and the need that it fulfils. It should also describe the functionality of the project, as well as the technologies used. Detail how the project was deployed and tested and if some of the work was based off other code, explain what was kept and/or how it was changed to fit your need. A project submitted without a README.md file will FAIL.
- In addition to the README.md file, you may include in your repository supplementary documentation and/or other relevant supporting material for the assessor in any format that is automatically handled by web browsers, such as html, pdf, jpg, etc. Files in proprietary formats such as Microsoft doc/docx will be ignored; but this is generally not a hindrance, since the vast majority of formats can be easily exported to PDF.
- Use Git & GitHub for version control. Each new piece of functionality should be in a separate commit.
- Deploy the final version of your code to a hosting platform such as Heroku.

IDEA

Irish Camping Forum App

- ***Allows users to create a profile with stripe payment – €1.99***
- ***Login/Log out function***
- ***Post threads to different forum topics***
- ***Edit/Delete threads & posts***
- ***Vote on Polls***

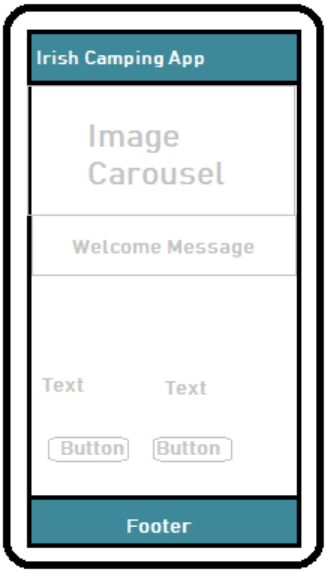
BUILD

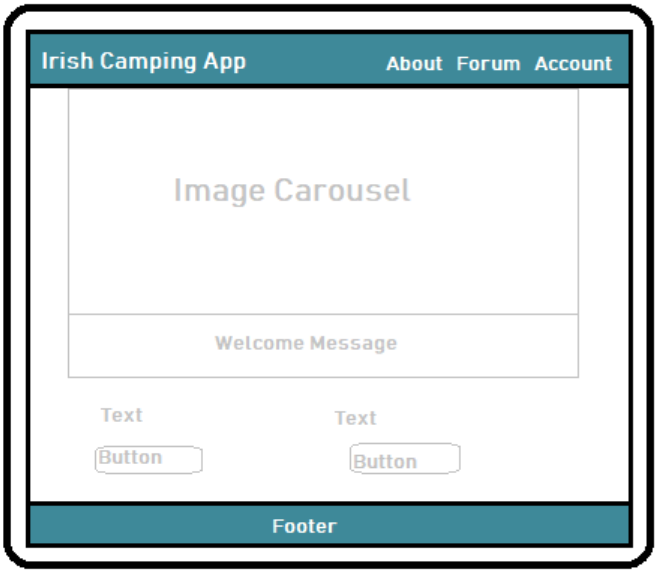
Django App

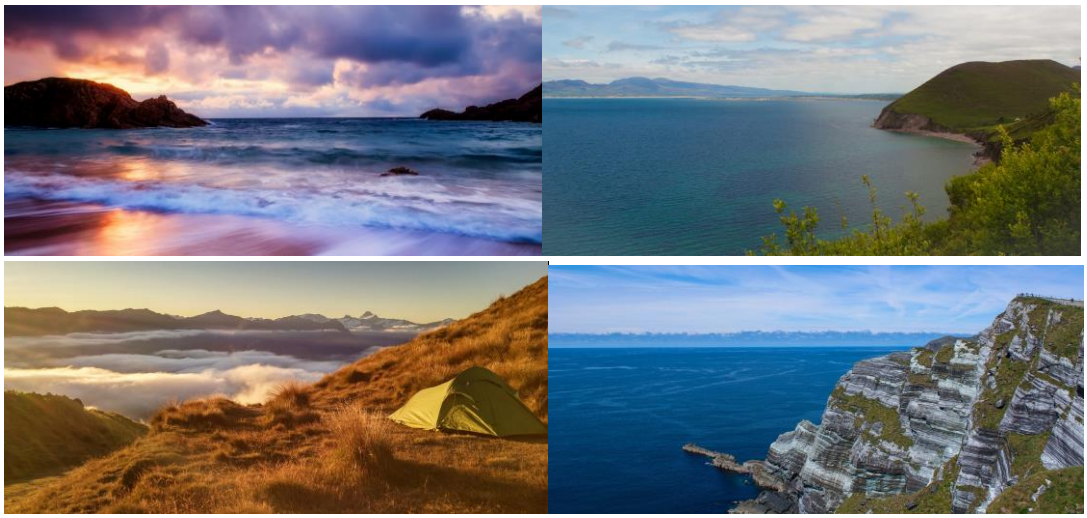
- ***Main base html***
- ***Template html***
- ***SQL data stored in ClearDB***
- ***Hosted on Heroku fetching source code from Github repo***

LAYOUT AND COLOR SCHEME

- ***Inspired by camping/outdoor photos***
- ***Use of my own photos and stock images***
- ***Forum layout and comment style using Tinymce and Font-Awesome***







-

- ***Forum Layout***