My Alcohol Free Wine

Daniel Clark

1 Introduction

This document will provide an overview of the Architecture, Rationales, Design and Testing of this implementation of the MyAlcoholFreeWine website, and it's defined Suppliers. From this point onwards MyAlcoholFreeWine will simply be referred to as MAF. The MAF is a price comparison site for My Alcohol Free Wines.

2 Architecture and Rationales of MAF

Due to the inherent nature of Rails, it logically made sense for the MAF application to use the Model View Controller design pattern. Rails makes using MVC really easy by providing generators for each part and convenience methods for each one.

2.1 UI Design

2.2 Database Design

2.3 Software Design

3 Architecture of Suppliers

I chose to develop the wine supplier applications in Python. This was mainly because I had experience with creating web servers in Python using the BaseHTTPServer.

I chose my endpoints to: /wines - GET - Returns a JSON list of JSON wine objects /wines/0 - GET - Returns the JSON wine object of ID '0' (or any ID specified) /wines/0/name - GET - Returns the Name of the wine with ID '0' (or any ID specified) /orders - POST - Adds the posted order to the suppliers orders, and returns an ID number, a list of the ordered wines, and a list of any wines which couldn't be ordered (because they don't exist or are out of stock).

The simplicity of the task meant that my supplier server only needed to deal with HTTP GET requests, for the getting of the available wines, and HTTP POST requests, for the ordering of wines. If I had more time I would have implemented HTTP DELETE to remove an order and HTTP PATCH to modify part of an order, e.g. the delivery address.

4 Conformance to Functional Requirements

FR1 (Browse non-alcoholic wines) has been fully implemented to the extent that all the pieces of data outlined in the specification can be viewed as part of a paginated list of all the wines available. The only slight adjustment to that is the images, however that is purely because the URLs for the images (which are stored in a JSON file on the supplier side and are easy to change) don't point to anything. Those URLs can be changed on the supplier and the change will propagate into the MAF within a minute thanks to the Whenever Gem integrating with Cron to supply a minutely update of the MAF database from the available suppliers.

FR2 (Search) has been fully implemented such that any search term can be entered and the site will display a paginated list of any items which match the search term in any field.

FR3 (Display Detail) is also fully implemented except with the same issue regarding images as described above.

FR4 (Add to Basket) has been implemented mainly thanks to the "acts_as_shopping_cart" gem which provides a good framework for building the basket.

FR5 (Display Shopping Basket) was implemented again thanks to the ease of use of "acts_as_shopping_cart".

FR6a (Checkout With Log-In) was almost completed however the checkout functionality does everything except actually post to the suppliers. This was purely due to time constraints. Most of the functionality for logging in was implemented with use of the "Devise" gem which provides an excellent authentication solution.

FR6b (Checkout Without Log-In) was easily completed thanks again to "Devise".

FR7 (Login/Logout and Registration) was, after a reasonable amount of work modifying the "Devise" gem to do what I needed, completed entirely.

5 Test Strategy

6 Self Evaluation

6.1 Screencast

6.2 Design

6.3 Implementation of the MAF

I think my implementation of the MAF site fits the functional requirements well. There is only one minor sections missed, like displaying images correctly, and one rather major part missed, which is the posting of orders to the suppliers, but that section is still mostly implemented.

6.4 Implementation of Suppliers

I think my implementation of the wine suppliers fits the functional requirements well. The only place it falls short is that the Ordering endpoints have not been tested as the MAF application was not able to send POST requests to the suppliers. Endpoints involving each individual piece of information about a wine were also implemented, for example /wines/0/name which allows for easier management when only part of the data has changed.

6.5 Testing

6.6 Evaluation

6.7 Flair