Team 3 (CarePax)

David Monti, Ryan King, Kevin Leung, Brent Zhang

CS 411 Section A1

October 13, 2015

Options for Web App Architecture

Naturally, when creating a web application that uses a database, there are many options for hosting platforms as well as general architecture of the site. This document analyzes the tradeoffs associated with two possibilities for architecture of our web app, CarePax.

Option 1:

One possibility for our website would be to host it on the Host Gator web servers, where Ryan King has an existing account. Here we could create a database using MySQL to manage the driver IDs, customer account information, and the inventory levels of the drivers. One big benefit to using MySQL is the ability to create relational databases, which ensures that data will not be duplicated as the platform scales. This data will be updated, and drawn upon through Ruby On Rails, which will be imbedded into our HTML code for database reads and writes. We selected Ruby On Rails because of its ability for rapid development and its Python-like syntax, which will make for a more familiar development environment for our team. Additionally, CSS and JavaScript code will be used to add aesthetics and functional performance to our web app. For CSS, we are choosing to write in a pre-processor called SASS, which will automatically convert our code to CSS. This will boost efficiency because SASS will allow us to use variables and hierarchical nesting, features that are not available in standard CSS. Finally, we will import the JQuery and Angular JS libraries into our web app to allow us to include various pre-programed front end animations and functions.

Option 2:

There are some alternatives that we are considering for our web application, primarily involving the back end database and hosting service. Instead of moving forward with Host Gator, we could open an account on Amazon Web Services and create our database there. If we were to use Amazon, we would want to create the database using Mongo. We like Mongo because we feel the code is easier to read and use than MySQL, and it is a newer method. However, Mongo does not use relational tables, so we do run the risk of data duplication. That being said, our user base is going to be relatively small starting out, so we do not feel that this is a major issue. For the data connector, the team believe Ruby On Rails will remain our best option regardless of what database style we use. Standard HTML, JavaScript, and CSS will be applied similarly to Option 1.

Finally, the team will use Foundation, which is installed on our computers, as a HTML framework for efficient and robust layouts, mostly in the form of a grid.