**Copa DB**

Project: Copa 2015 Website

Group Members:

Juan Yu

Kevin Chaorenworawat

Paul Milla

Vincent Nguy

Table of Contents

**Introduction:**

The purpose of this report is to document the process for building a full-fledged website which is dynamic in nature. The website is meant to function similar to IMDB where various data models can be linked together as needed. The website will be created using a mixture of Python, Django, Bootstrap, HTML/CSS, and Javascript. The report will go over the steps taken for each process in the build.

**Problem:**

The goal of the website is to take data that can be found spread out on the internet and consolidate it into a single website. One such use for this website is Copa 2015. There are many websites which have new information related to the event but not a single website dedicated to Copa 2015 itself. As such we sought out to build that single website which will maintain and display all the pertinent information to the even. The information can range anywhere from how man goals a particular player made to how many goals the entire team made during the event. CopaDB hopes to create a site where every question can be answered through CopaDB.me:8000.

**Uses:**

CopaDB.me:8000 can be used in order to look up statistics and results for Copa 2015. The site allows for navigation between matches, teams/countries, and players. The match landing page will contain the list of attributes each match has as well as a table for all the matches which took place. The teams landing page will contain the list of attributes each team has as well as a table for all the participating countries. The players landing page will contain the list of attributes each player has as well as a table for all the participating players. Each individal match will link over to the two involved teams. Each country will link over to its captain's player page and its top scorer's player page. Each player will link over to the country which they play for.

**Data Scraping:**

**Data Models:**

In order to store the various information scraped off the internet, we made models based off the Django format. These models represented how the data would be stored in our databse at a later date. There were three distinct models for Copa 2015: the Match, the Country, and the Player. The Match model contains the match name, each Country, the ending score, the man of the match, the match location, and the match date. Three foreign keys are maintained which represent each of the two Countries involved in the match and the man of the match which is a Player. The \_\_str\_\_ method is used to retrun the name of the match. The Country model contains the country name, number of total goals made, the team coach, the team's captain, and the team's top scorer. Foreign keys are maintained for the country's captain and country's top scorer which are Players. The \_\_str\_\_ method is used to return the name of the country. The Player model contains the player name, country they played for, their position, their dob, the number of goals they made, and their original club. A foreign key named country is maintained for the relation of Player to Country. The \_\_str\_\_ method is used to return the name of the player. The use of foreign keys through the models allow for the database to be linked to existing data files.

**Test Cases:**

For the first phase of the project the test cases were trivial in nature as they only tested the correct creation of the data models defined in models.py. There are a total of nine tests where each model has three individual tests. Each test consists of creating a test framework using Django and creating dummy model objects with various data we had scraped. For example a Country object is made with the country variable set to Mexico and the coach set to Miguel Herrera. Another Country object is then made using Country.objects.get(country=”Mexico”). Finally the test can be carried out by asserting that the object gets and returns the country variable as Mexico and the coach as Miguel Herrera. This example is extended for all attributes for each of the three models.

**Front End:**

For the first phase of this project static html files were used. The website allows for navigation between the main splash page, matches, teams, players, and the about section. So far we have written an about.html, players.html, teams.html, and matches.html. Each html file includes the same navigation code to include the necessary html files for each link to work. As such clicking on a player's country will directly link over to the created country. The website currently has three separate pages for each of the three data models.

The layout to the site is provided by Bootstrap with a starter template from <http://getbootstrap.com/getting-started/>. The template includes a nav-bar for linking the pages of the site and a carousel for extra functionality.

In the second phase we began changing our static html pages in order to allow data to be dynamically integrated. For example the syntax of inserting a player name into a page would be <p>{{Player.player}}</p>.

With the addition of Django’s Template inheritance the base.html file contains the html that is included in all pages on our site. Below is a minimal template that illustrates the basics. A template contains variables, which get replaced with values when the template is evaluated, and tags, which control the logic of the template.

{% extends base.html %}

{% block title %} Title {% end block %}

{% block content %}

……

{% end block %}