SWEN30006 Project 2

Wrangling Data

The Task

Your task is to develop a Rails application that will download and parse data from two different sources and store that data in a coherent abstract data format that does not expose the original format of either data sources.

Your rails application should, whilst running, use the two sources we provide to download all required weather data related to the Greater Metropolatin area of Melbourne every 10 minutes. It should then provide a basic web interface to view this data.

This project continues the work on building toward our final weather prediction application by familiarising you with data processing techniques and working with APIs in Ruby. Like project 1, this is an **individual** project.

Data Sources

You will be required to import data from two different sources for this project:

- The Beureu of Meteorology (www.bom.gov.au) - The Australian Government Weather Service
- Forcast.io (www.forecast.io) - A 3rd Party Weather Data Aggregator that provices a JSON API.

Both data sources will require a different method of data extractions (discusses in detail further in this specification), but have very similar data attributes and fields. You should start by investigating the data that each provider offers and the different formats it comes in. This will guide your design for your class diagram. Specfically, you will be required to store the following data:

- Rainfall Amount (in mm)
- Current Temperature
- Dew Point
- Wind Direction (if present)
- Wind Speed (in km/h if present)

Abstract Data Types and Class Diagram

An Abstract Data Type is a mathematical model of a collection of data and the functions that operate on that data. In object-oriented programming languages, we tend to represent these as classes. A large part of the challenge in this project is determining what an appropriate abstract data type is for the data you are storing.

After investigating the data sources you will be required to ingest, you must complete a class diagram for the structure you intend to use to hold your data. Given your program is a Rails Application, this will require the creation of several model objects and the relations between them.

Your class diagram must demonstrate the entirety of your model layer, all the attributes of each class and all the methods associated. You must also demonstrate class methods if you intend to use them.

Scraping HTML

The data provided by the Bureau of Meteorology is sadly, not availble in a nice convienient API. As such, you will need to develop a web parser that can read the HTML from their website and process it accordingly. All the data you will need can be found here:

http://www.bom.gov.au/vic/observations/melbourne.shtml

That page contains all of the data for the latest Melbourne weather observations and is updated every 10 minutes. Your task will be to download this data and parse the HTML for each value, you will need to then store the data in your appropriate abstract data type.

You may use any gems you like to help with this task, a popular option is the gem 'nokogiri', which can be used as follows:

```ruby
# Define the URL we are opening
URL = 'http://www.bom.gov.au/vic/observations/melbourne.shtml'
require 'nokogiri'
require 'open-uri'
# Open the HTML link with Nokogiri
doc = Nokogiri::HTML(open(URL))
# As an example, find an element
puts doc.css('#tMELBOURNE')
```

Nokogiri allows easy access of elements with css selectors, which for this data should be more than adequate. You are however free to use any tools you would like to. Just ensure that any gems you use are included in your 'gemfile'

Parsing JSON

Forecast.io, our second data source, requires that you register for an API key here: (https://developer.forecast.io/register) to start. The API is free to use provided you do not make more than 1000 calls a day. Given this restriction, you should make as many calls up until this API limit per day for the Forecast.io data.

After you have registered for an API key, you will find the API documentation located here: (https://developer.forecast.io/docs/v2) to be indispensible. An example of how to download a JSON file into a ruby hash would be:

```ruby
require 'nokogiri'
require 'open-uri'
require 'json'

# Define the URL
API_KEY = 'a3df59710af62078ae232f3a8184fbe1'
LAT_LONG ='37.8267,-122.423'
BASE_URL = 'https://api.forecast.io/forecast'

# Make sure you down forget the .read on the open file.
forecast = JSON.parse(open("#{BASE_URL}/#{API_KEY}/#{LAT_LONG}").read)
```

You will then be able to request the values out of the Ruby hash using the standard hash methods. Consult the Ruby docs if you an unfamiliar with this. **Note** the Forecast.io requires longitude and lattitude in it's API calls, you should endeavour to make these values as close as possible to the weather stations from the Bureau of Meteorology for this exercise.

API Key In Example

> *Note that the API Key in the provided example is only temporary and will be disabled before we test your*

> *applications. If you use it instead of acquiring your own, your project will* ***not*** *work when we test it.*

Web View

You must provide a basic web view that allows us to fully explore the data that you scrape. We do not care what this looks like as long as it is functional (a good example might be to follow the lead of the simple table that the Bureau of Meteorology uses).

This webview should be accessible through the url

```
http://localhost:3000/weather/data
```

Submission Instructions

Project submission will be enabled through the LMS. We will require you upload a zip file of both your rails application and class diagram to the submission link we provide.

The rails app you provide will be run as follows:

```
bundle install
rake db:create
rake db:migrate
rails s
```

Should your application require additional commands in order to set up, you **must** include a README file in your project detailing the additional commands.

If your program does not run using these commands (or those provided in your README), you will lose **2 mark** and we will attempt to fix it. If we cannot make your rails app run at all, you will receive 0 marks for JSON, HTML and Web View criteria.

Marks

This project will account for 8 marks out of the total 100 available for this subject. These will be broken down as follows:

Criterion	Percentage
Class Diagrams of Data Storage Design	3%
Implementation of JSON Download and Parsing	2%
Implementation of HTML Scraping and Parsing	2%
Web View of Data	1%

Class Diagrams will be marked on correctness and adapatbility of your abstract data type. We also expect good object oriented design principles and functional decomposition.

JSON and HTML Scrapers will be tested manually, as will the web view of data.

On Plagiarism

> *We take plagiarism very seriously in this subject. You are not permitted to submit the work of others under your own name. This is an **individual** project. More information can be found here: (https://academichonesty.unimelb.edu.au/advice.html).*

Submission Date

This project is due at **11:59 p.m. on the 24th of April.** Any late submissions will incur a 1 mark penalty per day unless you have supporting documents. If you have any issues with submission, please email Mat at mathew.blair@unimelb.edu.au, before the submission date.