Coding challenge

A hotel chain has a lot of hotels functioning under it, from across the world. Each hotel gets its own customer reviews from various review websites like Tripadvisor, HolidayIQ, Booking.in, makemytrip, etc. For a hotel owner to manage all these reviews and see them at a single place is a challenge. Today we are going to solve that challenge for one hotel chain, called Sterling Holiday Resorts.

Sterling Resorts has many hotels across the country. Three of which are : White Mist Manali, Villagio Goa, and Dindi by the Godavari. Of the two data files given to you, one of them contains all reviews of Sterling Resorts pertaining to these three hotels. The other contains the relationship between the parent (Sterling Resorts) and its 3 child units. Your task is to make it easier for the owner to look at feedback coming in from his three different hotels.

Make a small Python Flask application which makes use of reviews.json and relation.json to show two things —

1. Overall sentiments of reviews for Sterling Resorts. You can make a pie chart or bar graph for the three sentiments (Positive, Negative, and Neutral).
2. A page each for each of the three child hotels. On this hotel page, one should be able to see the sentiment graph (similar to the overall graph above) as well as all reviews of that hotel.

On running your application, we must first see the pie chart / bar graph for the overall sentiments of the 900 or so reviews. On clicking on one of the three buttons for hotels, we must be routed to a page where we see the sentiment graph for that hotel, followed by reviews for that hotel. If you do not wish to make multiple pages, you may combine task 1 and 2 in one single page itself. Make sure every section has its appropriate title.

Instructions

You must use Python 3.5 and Mongo 3.0.4
Import the two data files into your MongoDB database and get started with building the app!

What we are looking for :
1. Basic API calls / routing knowledge
2. Good knowledge of data structures and object oriented programming.
3. Basic use of any front-end Javascript framework (bonus points for using AngularJS)
4. Reading from mongo using Python

*What we are **not** looking for* :
1. Too much focus on designing and beautifying the frontend. A presentable view is just fine.
2. Unnecessary multiple calls to database. Keep it simple. At the same time make intelligent mongo queries and avoid loading more data from your server than you need!
3. We don't want any log-in feature at this point.

Push your project to github. Name the repository - YourName_SurvaiderTask
Alternately, you may also email the zipped folder of your submission to us. Keep in mind, in both cases, to have a requirements.txt file. You may include an installation README if you like.