Elasticsearch

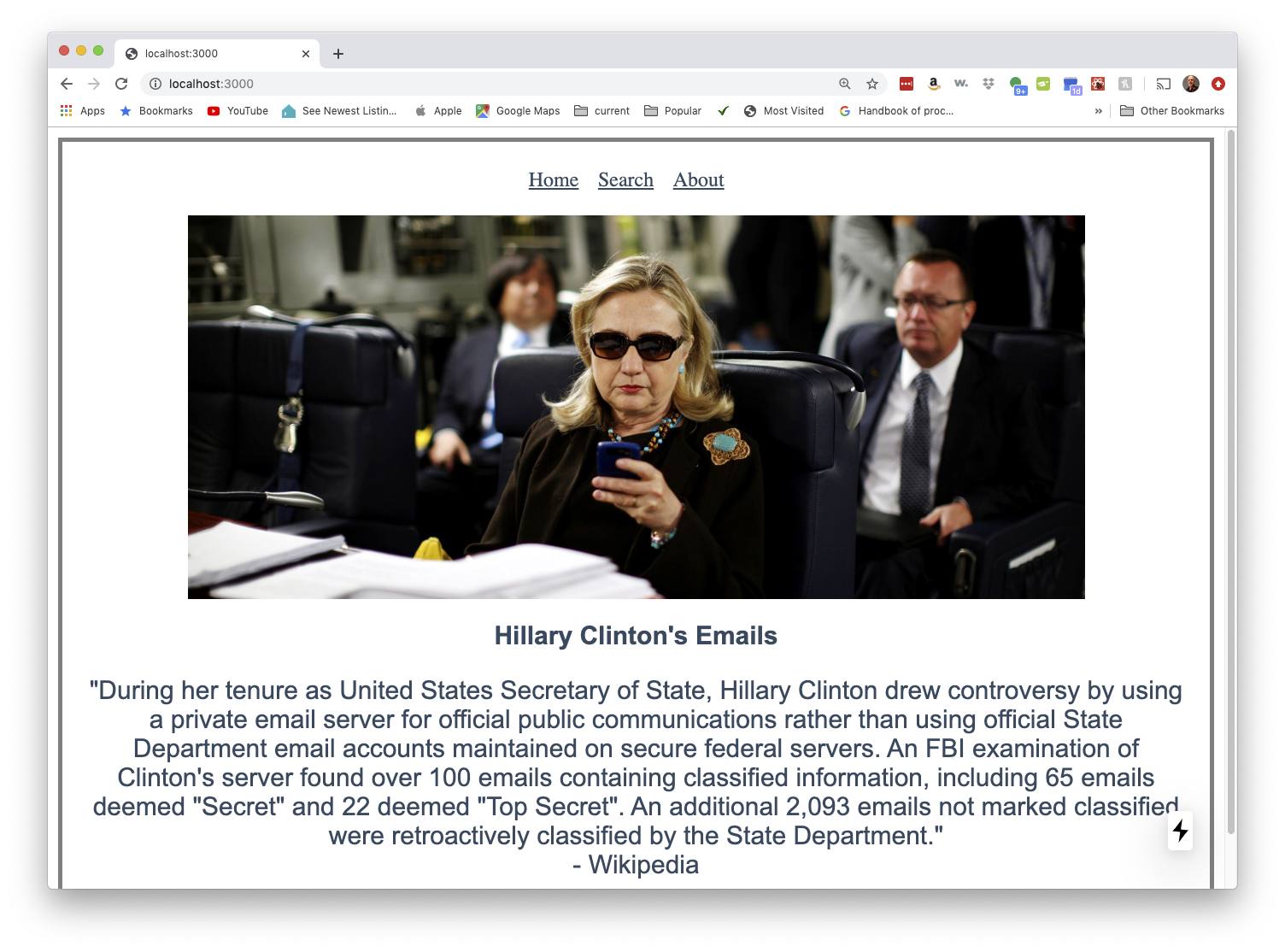
You must have demonstrated at least the complete web app before demoing this rudiment.

## Project Description.

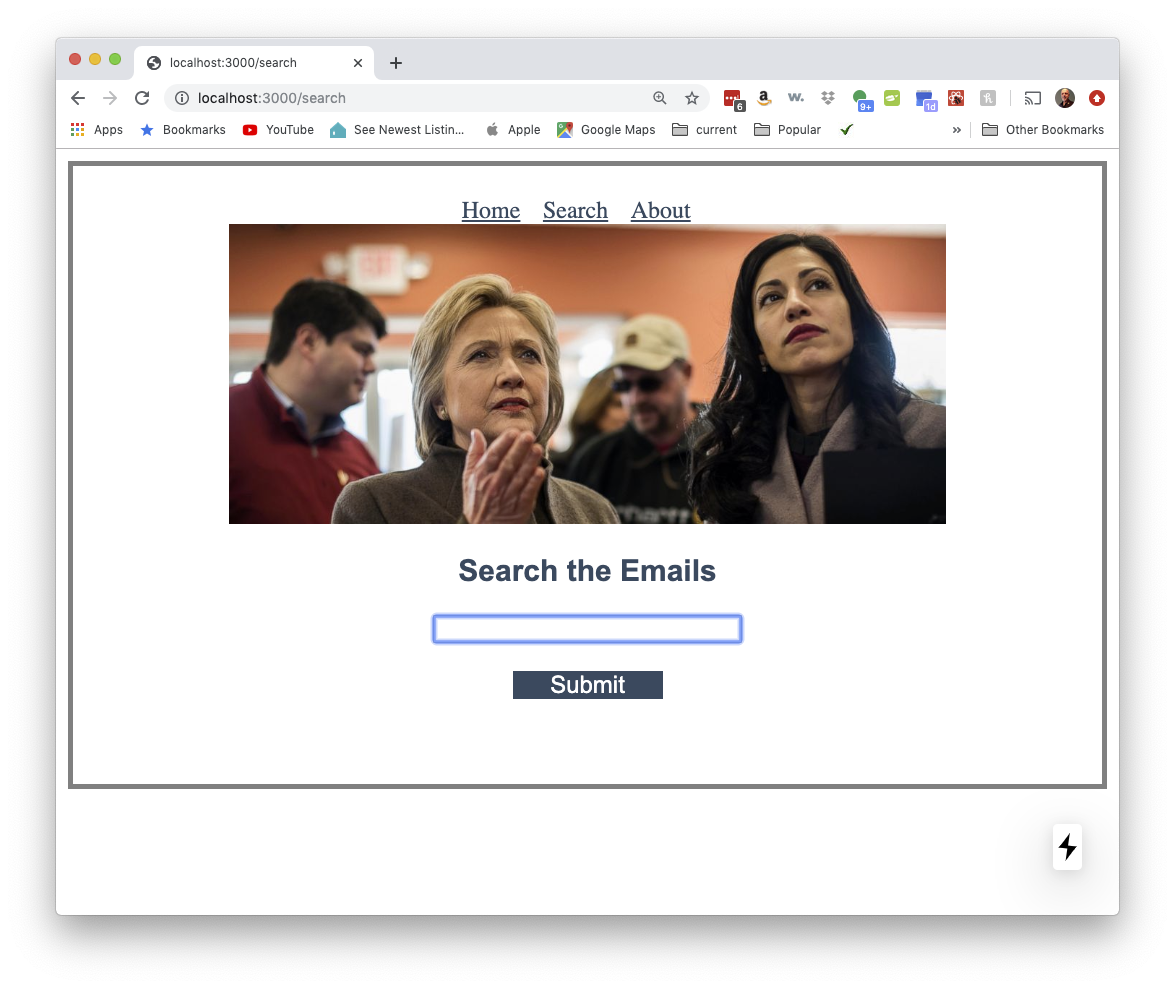
You are to write an Elasticsearch powered web application that allows a user to search a collection of Hillary Clinton emails. This dataset is the largest public email dataset in the world.

Alternatively, you can select a different dataset to use. If you have an idea for an alternative, talk to me.

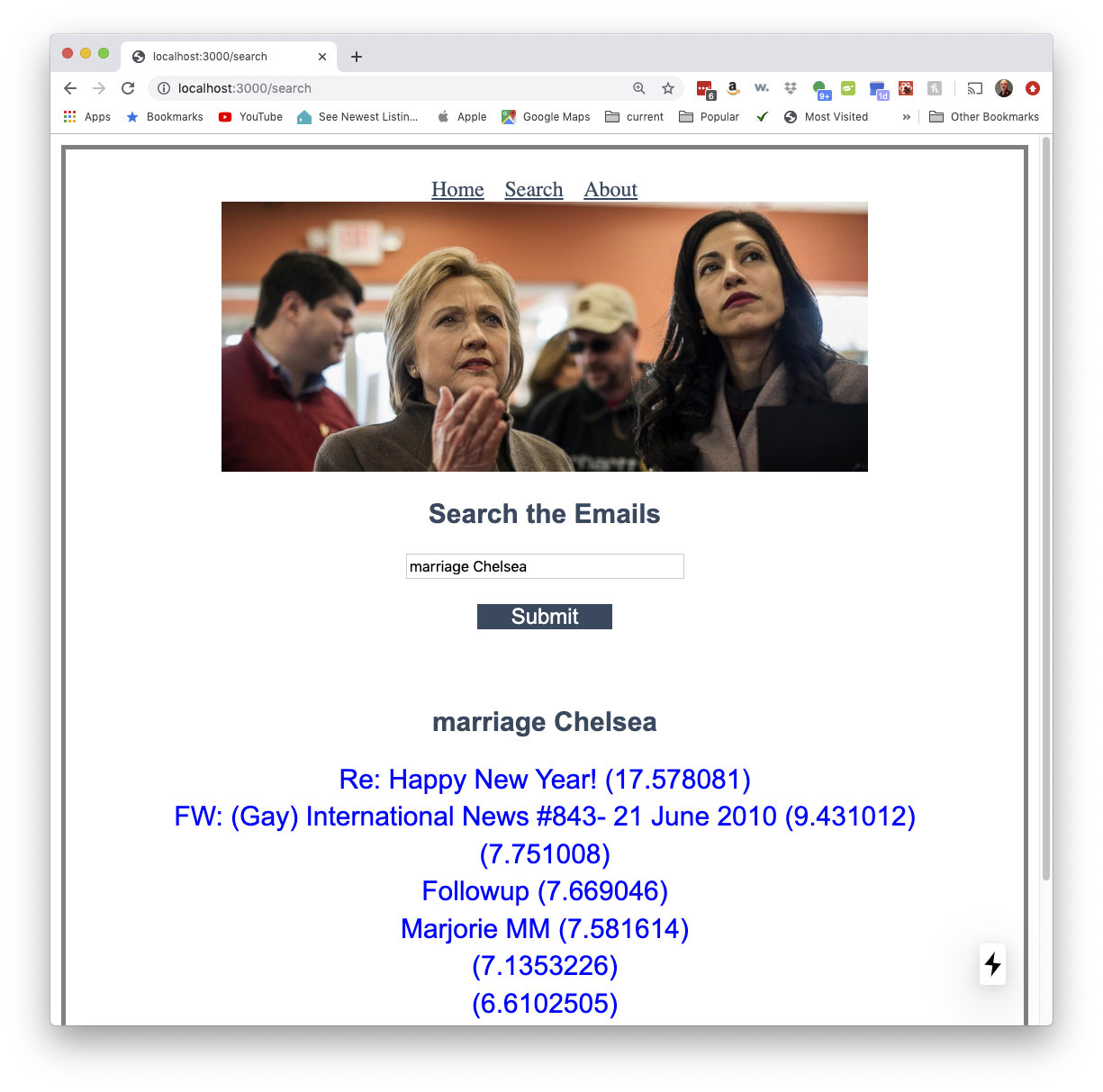
### Minimum Requirements



Minimally users should be able to search the body of the emails:

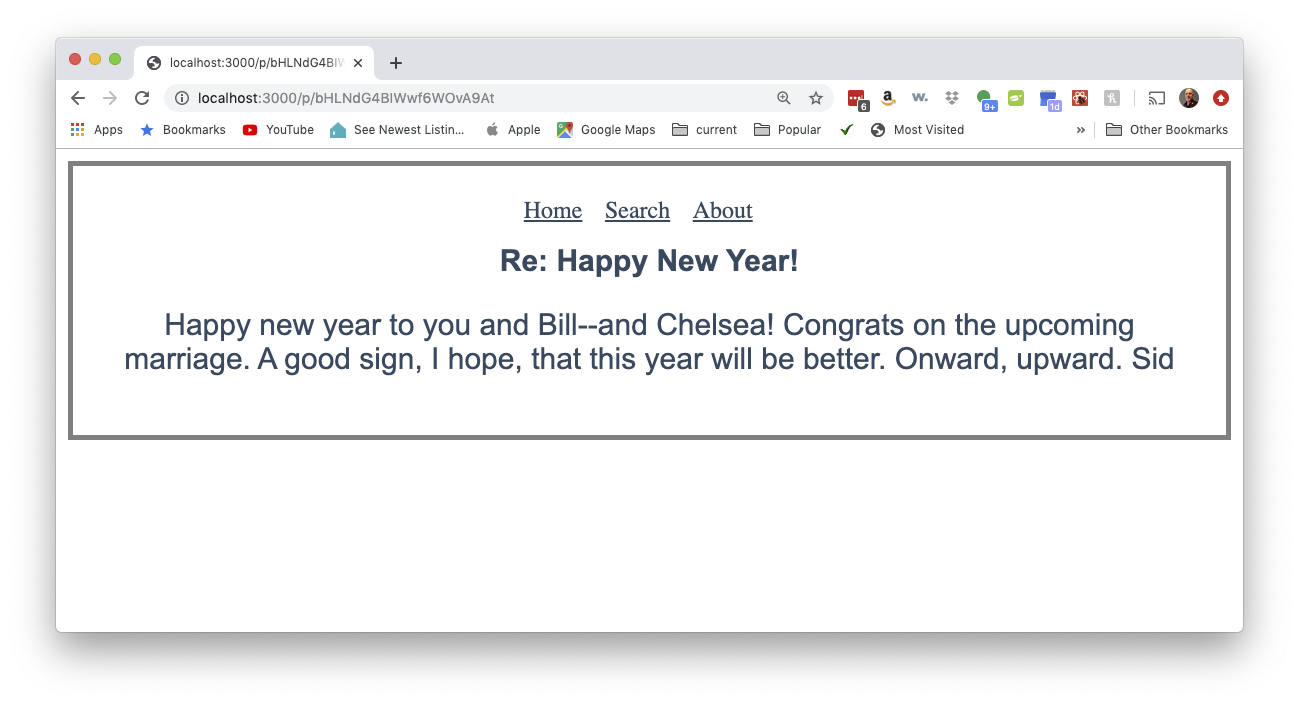


When a user types in their search string and clicks submit they should see a list of results:

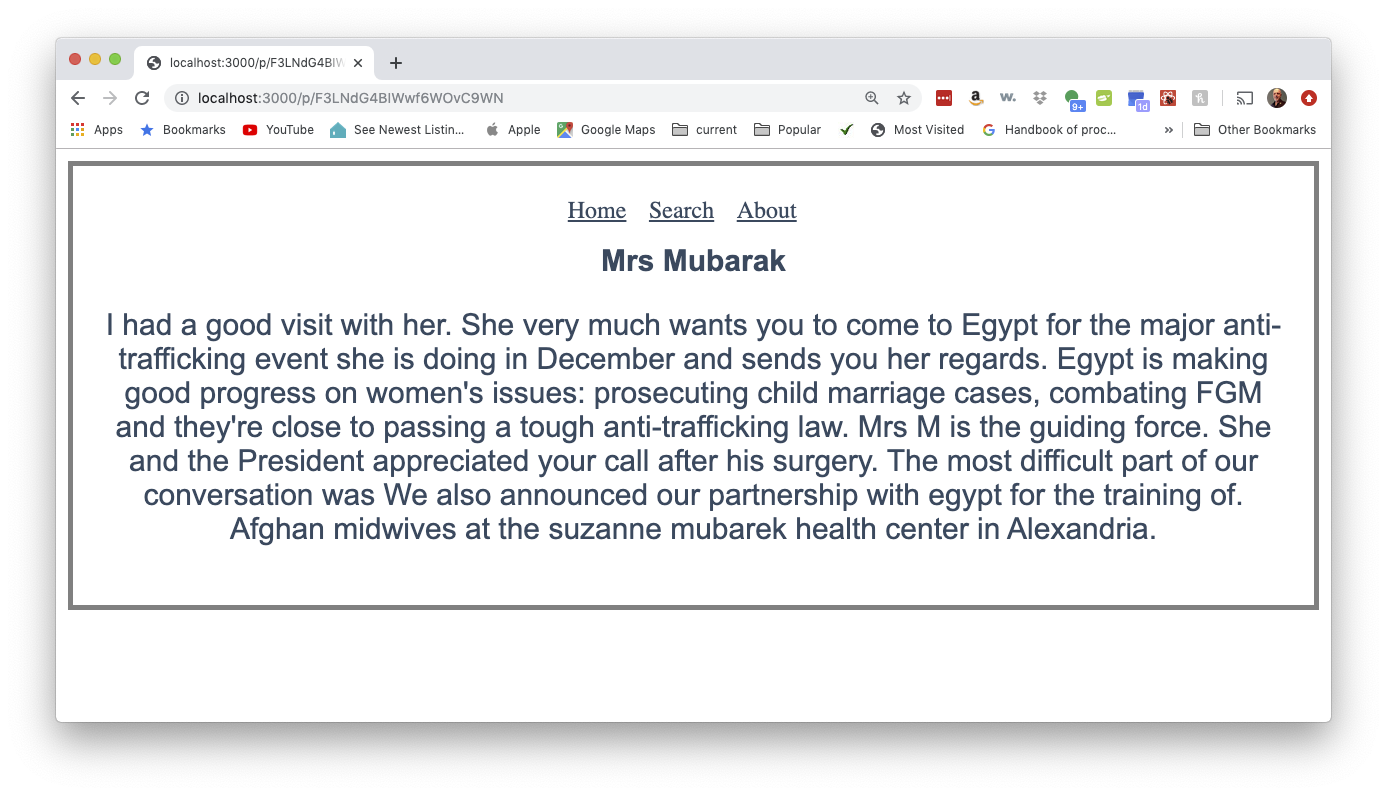


The example above only has the subject line of the email and the Elasticsearch rating. You can have what you want.

When a user clicks on an email they see the entire email.



## 



## Resources

### Reasonable video about Elasticsearch (but in Python)

[Elasticsearch in an hour](https://www.youtube.com/watch?v=UPkqFvjN-yI&t=561s) - I found the first 20 minutes or so very helpful.

### [The email dataset](https://www.kaggle.com/kaggle/hillary-clinton-emails/data)

You need to create a free account first. The dataset is in both a sqlite file and a csv file.

### Importing the data

I spent over 8 hours exploring ways of importing the data and by far the easiest is described in [Importing CSV and Log Data into Elasticsearch with File Data Visualizer](https://www.elastic.co/blog/importing-csv-and-log-data-into-elasticsearch-with-file-data-visualizer). I renamed some of the fields of the csv file within the visualizer just to make it easier later on.

This import method uses Kibana. You can also use Kibana to try out some initial queries. You don’t need Kibana for production. I only used it to import the data.

### Elasticsearch in Nodejs

I used [the official Elastic node module](https://github.com/elastic/elasticsearch-js). After going through the examples on that page, I aftered it so it would work with the Clinton email index -- just to print a few examples in the terminal.

Next, I wrote the backend code.

#### Front end code

I based my front end code [on part 6 of the learn nextjs tutorial](https://nextjs.org/learn/basics/fetching-data-for-pages).

## Demo and xp

You can demo several times. First, to get points for the baisc system and then once you add any improvements.

|  |  |
| --- | --- |
| description | xp |
| Demo basic functions on laptop | 95 |
| Demo on Cloud | +75 |
| Professional looking page (subjective) | +15 to +35 |
| Running multinode Elasticsearch | +35 |
| Allow a user to have more complex queries. For example,   * google has the option to say results must include the searchterm * Allow the user to search multiple fields of the email (in a particular date range or text in the subject. | +40 for something equivalent equivalent to one of the options mentioned. Up to +100 for a full blown search engine. |
| Different dataset | Potential for more xp |
| Other explorations (for ex., using a docker image on the cloud) | More xp (for ex., using Google Cloud Buld - 50xp) |

# Other misc. info

## In Docker

### 

#### Pull the images

docker pull docker.elastic.co/elasticsearch/elasticsearch:7.4.2

docker pull docker.elastic.co/kibana/kibana:7.4.2

#### 

#### 

#### 

#### Create a network called ‘elastic’

docker network create elastic

#### Spin up both elasticsearch and kibana

docker run -d --name elasticsearch --net elastic -p 9200:9200 -p 9300:9300 -e "discovery.type=single-node" docker.elastic.co/elasticsearch/elasticsearch:7.4.2

docker run --link es01 -p 5601:5601 --name kitana --net elastic docker.elastic.co/kibana/kibana:7.4.2

### Then

docker stop elasticsearch

docker start elasticsearch

#### Shakespeare data

curl -H 'Content-Type: application/x-ndjson' -XPOST 'http://localhost:9200/shakespeare/\_bulk?pretty' --data-binary @shakespeare.json