MICHIGAN STATE
UNIVERSITY

Project Plan
Leveraging SPAM to Make Bold Societal Predictions

The Capstone Experience

Team Proofpoint

Scott Newhard
Ben Kandel
Connor Southwell
Matthew Xu
Cameron Smith

Department of Computer Science and Engineering
Michigan State University

Fall 2020

From Students…
…to Professionals

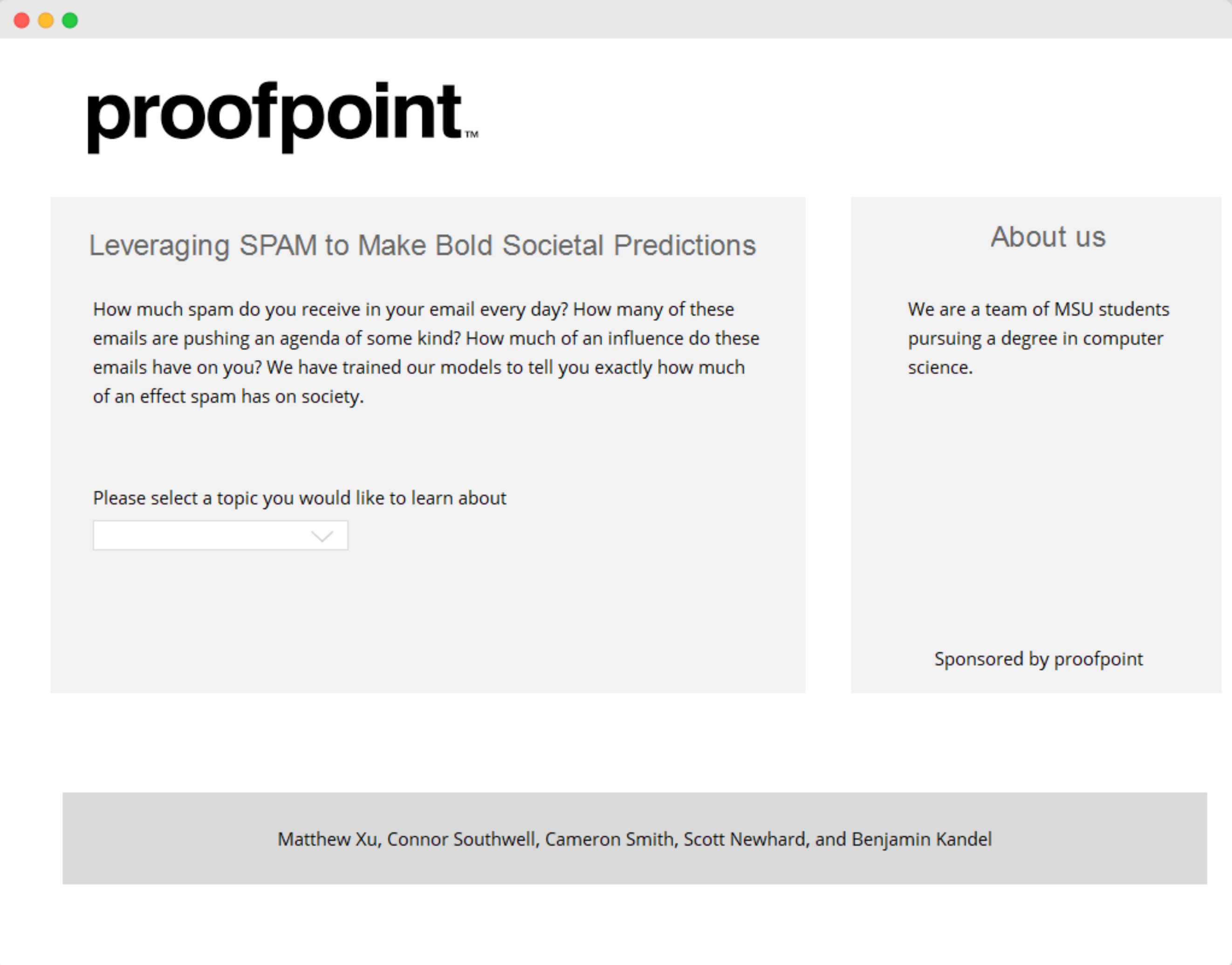
Functional Specifications

- The end goal of this project is to find ways to monetize the data currently being held by Proofpoint. This could come in several different forms of prediction, I.e. election outcomes, stock market rises and falls, and overall consumer sentiment
- We are using natural language processing to find the sentiment of these emails and use that in coordination with machine learning to find ways to holistically predict the outcome of societal events

Design Specifications

- Email Parsing: Proofpoint supplies us with terabytes of emails that will be dumped into a JSON object by our application
- Natural Language Processing: Once emails are stored in the database; sentiment analysis is done to gauge both the subject of the email as well as the overall sentiment toward that subject
- Machine Learning: Statistical analysis is then done on the sentiment collected from all emails contained in the database to find common topics as well as the sentiment portrayed about said topics
- API: The backend sets up an API for communication with our frontend
- Web Interface: The frontend is focused purely on showing information in a comprehensive manner. It will give the user a view of overall system health and data while also providing drill down features to allow users to dive into the specific data they want to see, as well as various stats and analysis on the data we have gathered

Screen Mockup: Landing Page



Leveraging SPAM to Make Bold Societal Predictions

How much spam do you receive in your email every day? How many of these emails are pushing an agenda of some kind? How much of an influence do these emails have on you? We have trained our models to tell you exactly how much of an effect spam has on society.

Please select a topic you would like to learn about

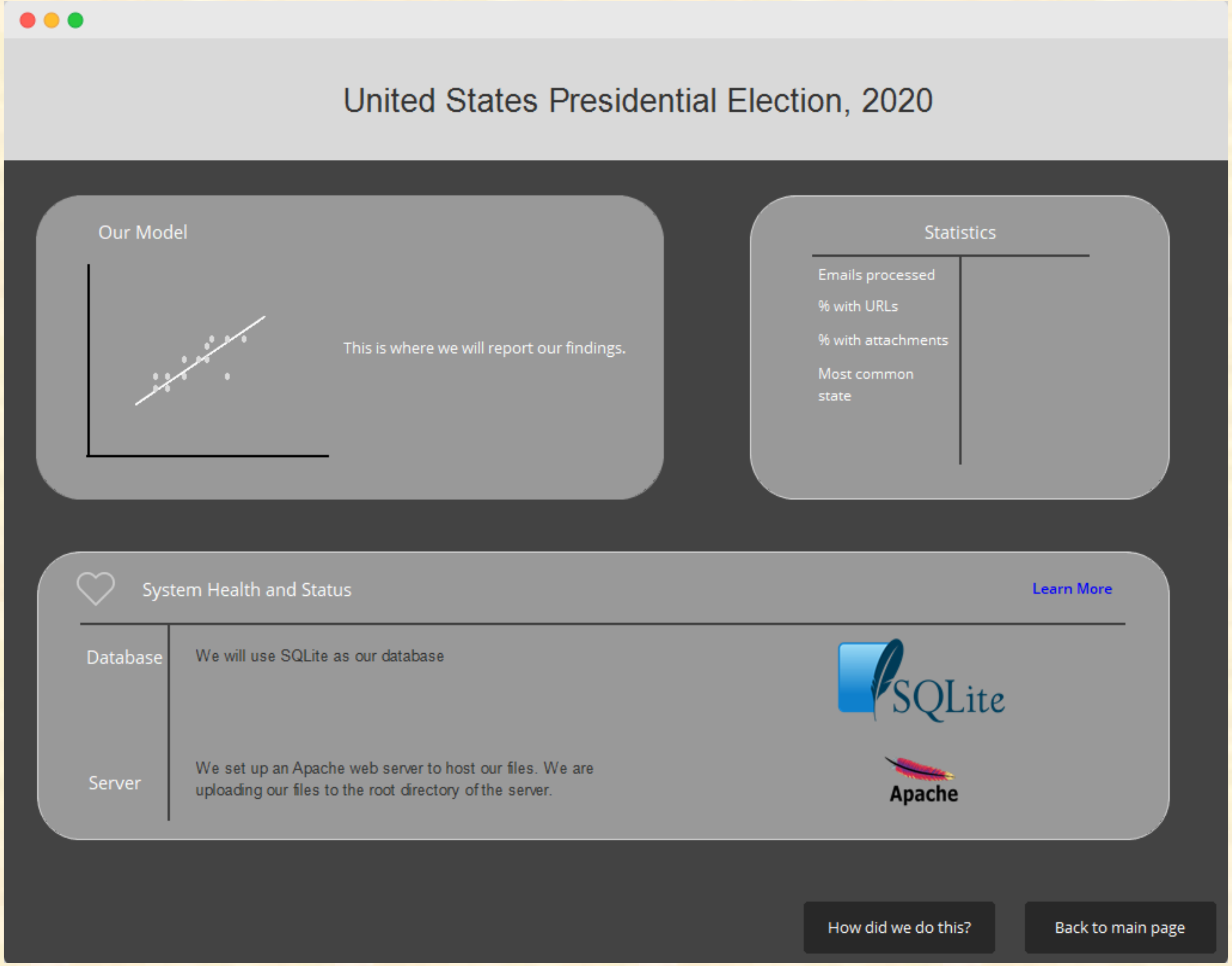
About us

We are a team of MSU students pursuing a degree in computer science.

Sponsored by proofpoint

Matthew Xu, Connor Southwell, Cameron Smith, Scott Newhard, and Benjamin Kandel

Screen Mockup: Dashboard



United States Presidential Election, 2020

Our Model

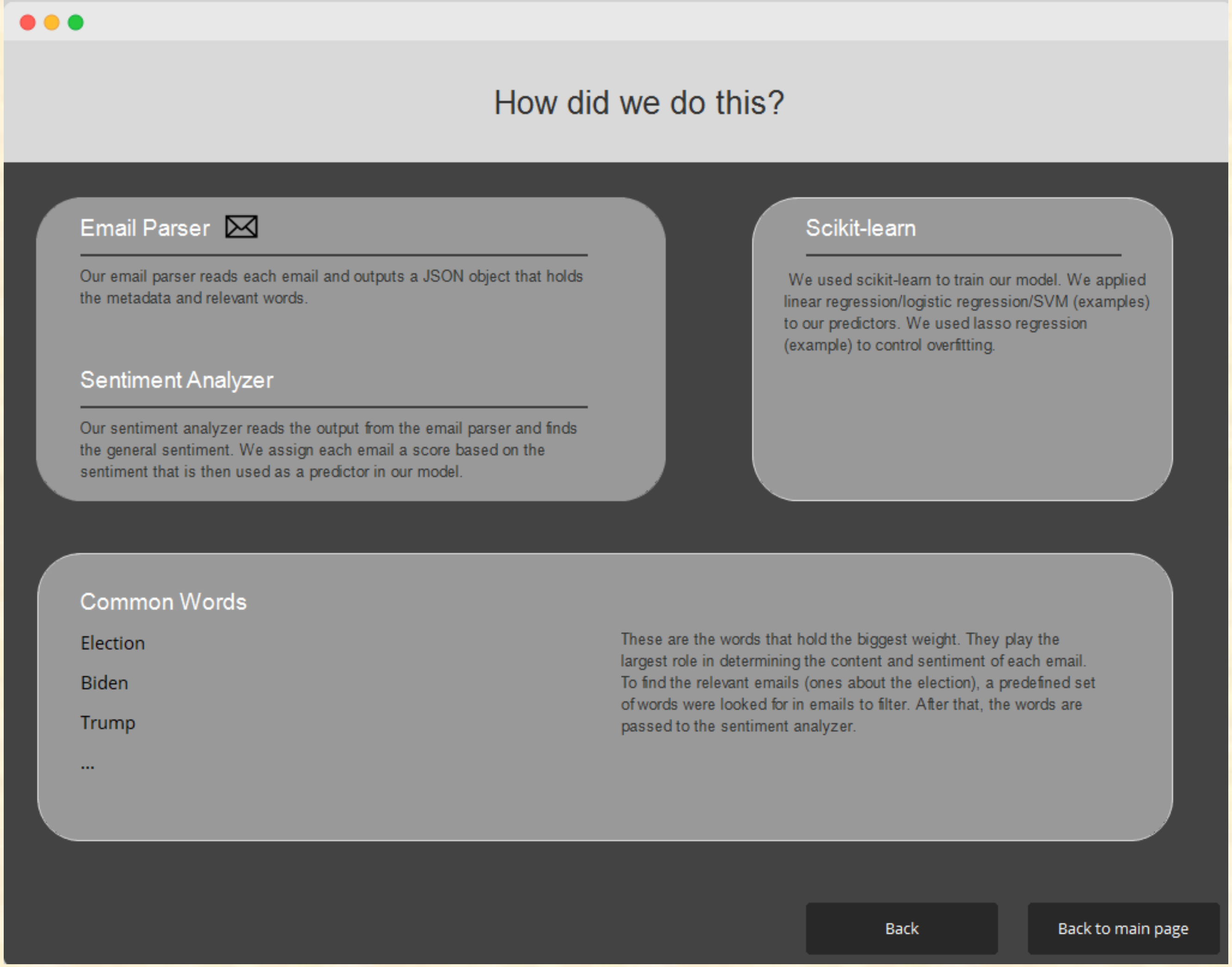
This is where we will report our findings.

Statistics	
Emails processed	
% with URLs	
% with attachments	
Most common state	

System Health and Status

Database	We will use SQLite as our database
Server	We set up an Apache web server to host our files. We are uploading our files to the root directory of the server.

Screen Mockup: How Did We Do This?



How did we do this?

Email Parser ✉

Our email parser reads each email and outputs a JSON object that holds the metadata and relevant words.

Sentiment Analyzer

Our sentiment analyzer reads the output from the email parser and finds the general sentiment. We assign each email a score based on the sentiment that is then used as a predictor in our model.

Scikit-learn

We used scikit-learn to train our model. We applied linear regression/logistic regression/SVM (examples) to our predictors. We used lasso regression (example) to control overfitting.

Common Words

Election

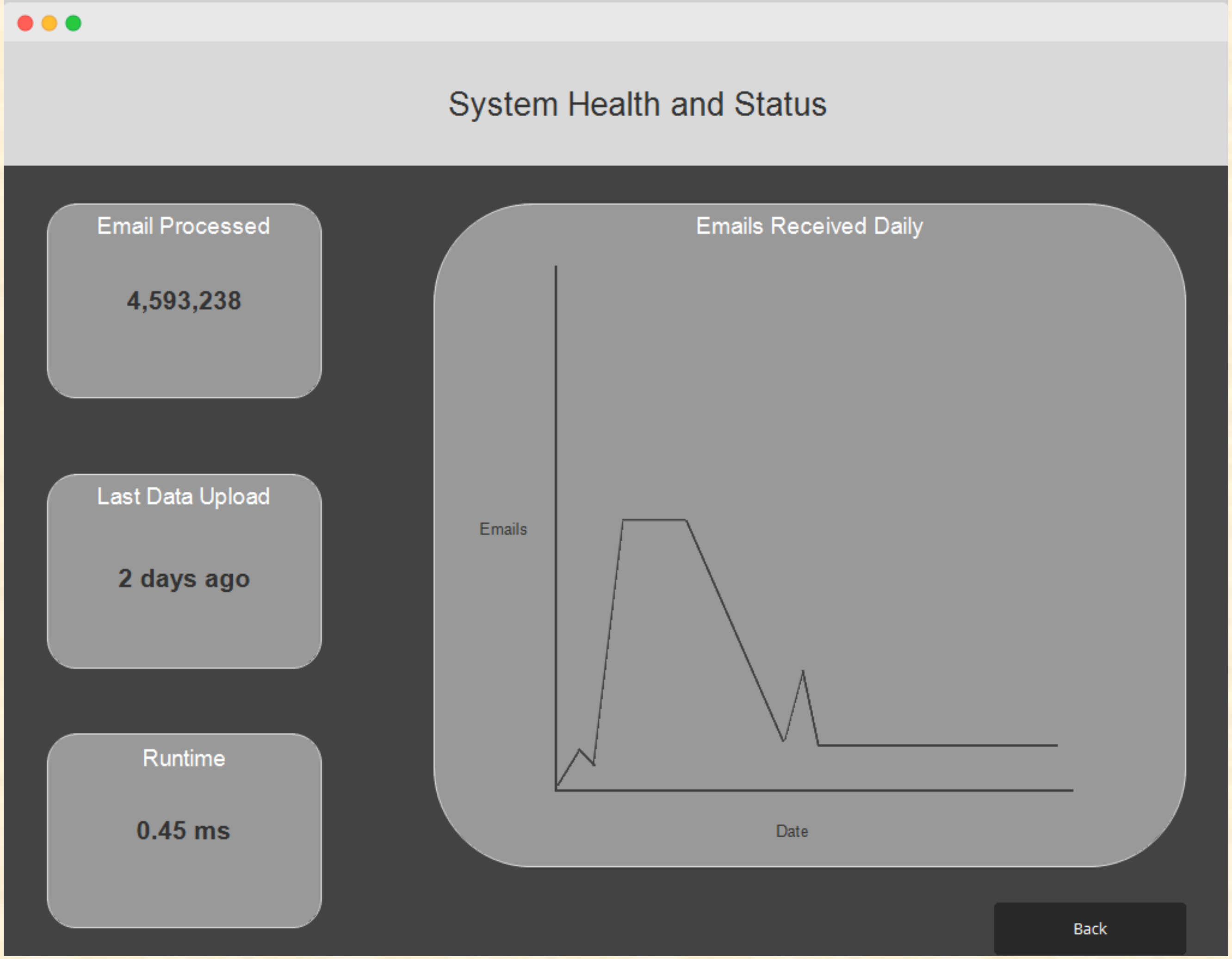
Biden

Trump

...

These are the words that hold the biggest weight. They play the largest role in determining the content and sentiment of each email. To find the relevant emails (ones about the election), a predefined set of words were looked for in emails to filter. After that, the words are passed to the sentiment analyzer.

Screen Mockup: System Health



System Health and Status

Email Processed

4,593,238

Last Data Upload

2 days ago

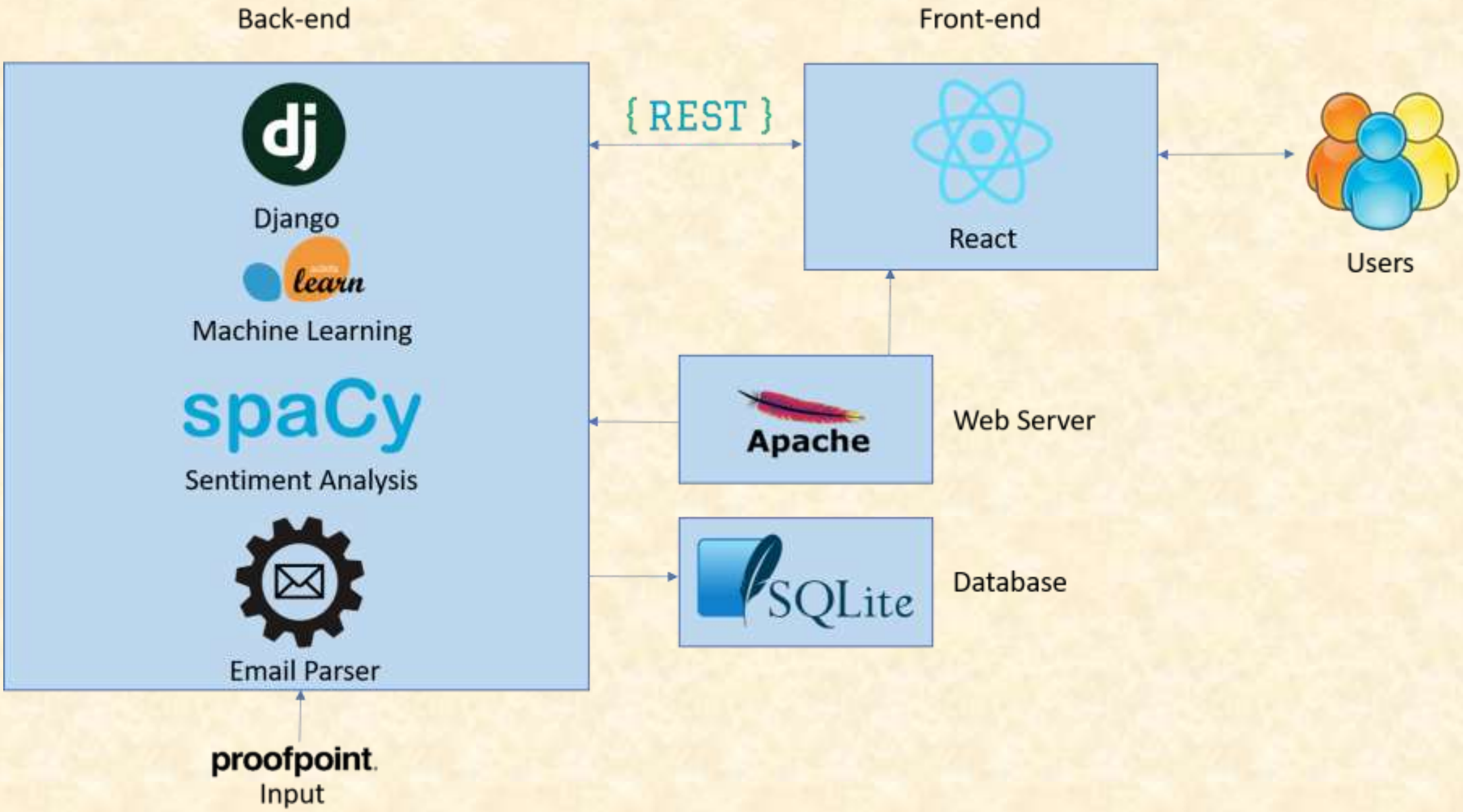
Runtime

0.45 ms

Technical Specifications

- Front-end: React
  - A JavaScript library for building UIs
- Back-end: Django
  - A Python web framework
- Machine Learning: Scikit-learn
  - A Python library
- Database: SQLite
  - Database is integrated via Django

System Architecture



System Components

- Hardware Platforms
  - iMacs (provided by MSU)
  - Personal Windows computers
  - Proofpoint local database servers
    - This is where the final version of the back-end code will run
- Software Platforms / Technologies
  - React
  - Django
  - Apache
  - SQLite
  - Scikit
    - Machine Learning in Python
  - EML Parser
  - SpaCy
    - Natural Language Processor

Risks

- Data shows no clear correlation
  - When using ML, a common risk is that measuring the trends leads to no correlation or that the correlation we find does not accurately predict events
  - Using multiple methods of training will provide several different ways to measure the data we are given. The different models, when trained on historical events, can then be compared against one another to find the most accurate model
- Data provided from only the last 30 days
  - Makes it difficult to train our model on historical data and gauge the accuracy of our current model
  - Use data from smaller local elections or smaller scale societal events to train the model and measure the accuracy of predictions
- Lack of direct access to database
  - For security reasons, we must give our code to the client to run on their machine which will hinder our ability to test our code on the full database
  - Sending our code in a timely matter to account for the delayed testing process. In addition, we will be setting up our own local database to hold sample .eml files we are given so that we are able to locally test our code
- User Interface
  - The description of the desired user interface is fairly open ended
  - Creating several mockups to show a variety of different possible outcomes for the user experience to find what works best for the client before the creation process gets too intense to make widespread changes

Questions?

? ? ? ?

? ?

? ?

?