Zach Dilliha, Tyler Clark

Dr. Xia

CS 396 Intermediate Software Project

29 November 2020

Information System for Air Quality

# PROJECT DESCRIPTION

The goal of this project is to create a forum for the collection of information relating to air quality and air pollution. Users access this website to post on boards with different topics relating to air quality and pollution and to view data that relates to those topics. They can upload/download files that relate to the topic. The users can also search through pollution data in either from a City, from Vehicles, or from Power Plants as the data is maintained by the administrators. This project was completed using Django, a web framework for creating web apps with Python.

# FUNCTIONS IMPLEMENTED

Functions that we have implemented include:

1. There is an administrator that can create/delete Boards, Topics, and Posts, as well as deleting or “promoting” users to the level of administrator through the admin site, accessed by logging into the admin account and going to the dropdown bar by the username, clicking on admin, and using the admin site.
2. A Signup/Login function for users to register and log in with.
3. Any user can create a Topic within a Board, and reply to those topics.
4. Any user can upload and download files on a dedicated page.
5. The number of posts and views is kept updated.
6. Files are stored internally within the site.
7. There are buttons on the top navigation bar that link to other sites for additional information
8. The website interface is user friendly.
9. Database containing City, Vehicle, and Powerplant pollution data.
10. Search functionality on each table.
11. Ability to insert new data from the administration page.

New Functions Implemented

1. Data visualization using matplotlib to display database information.

# TECHNICAL DETAILS

1. Database Server
   1. The first database that we use is internal to the website, and holds the Boards, Topics, and Posts, as well as the user login credentials.
   2. The second database is also internal to the website, and it holds the City, Vehicle, and Power Plant data
2. Software
   1. For this project we used Django 3.1.1, a web framework for creating web apps with Python.
3. Language/Scripts
   1. For this project, we used Python and HTML as my languages.
4. Data Visualization
   1. For this project, we used Matplotlib to plot data over various types of graphs.

# FEATURES

1. Edit Posts
   1. Any user can edit their post after it has been posted. Pressing the edit button on the post takes the user to a new form where the original text is displayed so that it can be edited. After the post is edited, it will return the user to the topic page they were on originally so that they can see the edit. On the post section, it displays the time the post was originally created, as well as, if applicable, the time the post was edited. This serves to show that the post was edited, as without it, it can be very misleading if the post is edited after having other users reply to it.
2. Last Post Link
   1. When the user is on the home page, they will see a list of Boards that they can look through, and displayed alongside that is a link to the last post in that Board, on whatever Topic. This allows the visitor to see what new activity has happened since they last logged in.
3. Markdown Incorporation
   1. When creating a topic/post or replying to a post, or editing their post, users can use Markdown, a text-to-HTML conversion tool that allows for text customization. If the user is knowledgeable about Markdown, then they can utilize it when creating their posts.
4. Pollution Data Search Functionality
   1. The user can search through the databases containing the pollution data using the City name, state, or date for the City page, Automaker or Trim for Vehicles, and Name and State for the Power Plants.

# DATABASE DESIGN

This project utilized a database in order to store data on pollution levels in various forms. The way this database was designed was simple, with 3 tables: Cities, Vehicles, and Power Plants. Each table had its own fields for data that was unique to them. Each City had a Name, State, Latitude, Longitude, Date, Regmark, Temperature, Humidity, Wind Speed, Wind Direction, Speed, Accuracy, Minimum and Maximum of different particulate levels, and EFCO, EFHC, and EFNO. The Vehicles had Model Year, Automaker, Trim, PM2.5 levels, Engine Family, Engine Code, Engine Model, Manufacturer, Part Name, Part Description, Part Number, and whether or not it has a sensor. The Power Plants have fields for Name, State, SO2, NOx, CO2 and Hg emissions in tons, and Latitude and Longitude.

# DISCUSSION

For this project, we used Atom, a free and open-source text and source code editor. It was developed by GitHub and has Git Control embedded into the program. Besides its use as an editor, it is very useful for our collaborative purposes since Git Control is essentially a built-in way for Atom to push the current build of the project to a GitHub repository. For this project, HTML and Python worked well together to do what we needed to do. In order to complete this project, we researched and read up on Django and Python, as this was my first time using both. To prepare, we completed many small tutorial apps with Django and Python to get a beginning feel for the software and language. When starting the project, we consulted many sources for the variety of tasks the site was expected to do, and combined them into the final project. We believe that Django, Python, HTML, and JavaScript are sufficient for completing a project of this type.

HTML and CSS are used to provide the visual aspects of the website, allowing for customization of different pages and simplification of what the user sees. The HTML takes info from the Python and displays it to the user.

Python is used to do higher-level functions that interact with data that the website needs, such as the login/signup, file upload, and posting. The Python uses the HTML to gather/display information and functionality that the user needs. JavaScript was used for queries, saving data to the database, and for gathering info from the database for the various charts that were displayed on the site. Our Python scripts interacted with JavaScript to initiate and produce the info that needed to be displayed on the site for our purposes.