FGF Brands

HACKATHON

Team 5: Shaan, Navdeep, Jaimil, Nikhil, Simran

FGF

PROBLEM 1:

Meeting Schedule Automation

Our Algorithm:

- **Saves time** by automating scheduling and room selection
- **Reduces Errors** in scheduling and room allocation
- **Optimizes** use of resources (time and space)

FGF

INTRODUCING: ROOMMATE

How does it work?

- RoomMate uses predefined rooms, with their capacities: ex. 1235 Ormont Blueberry (6 people)
- User inputs:
  - Attendee's name
  - Email
  - # of attendees
  - Date
  - Time
  - Duration
- RoomMate uses inputted data to check room availability and books an available room

WHAT ELSE?

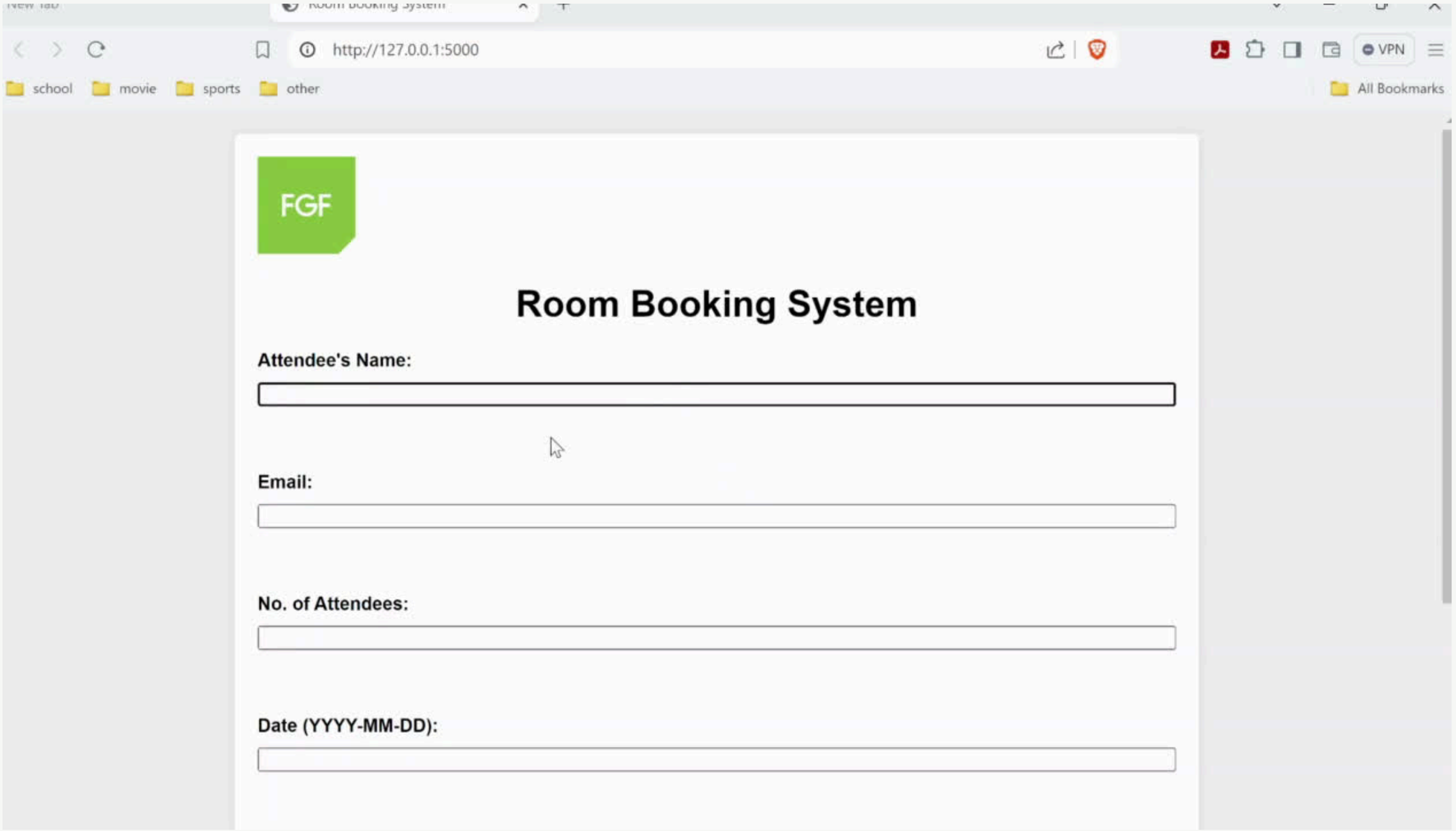
Date and Time Conversion

- Converts input date and time to timezone-aware datetime objects
- Book meetings from anywhere around the world!

Checks Room Availability + Allocation

- Uses Outlook calendar to check room availability
- Sorts + filters appointments to find conflicts
- If a suitable room is found, RoomMate **creates a new appointment and sends a meeting invite** to attendees emails.

USER INTERFACE



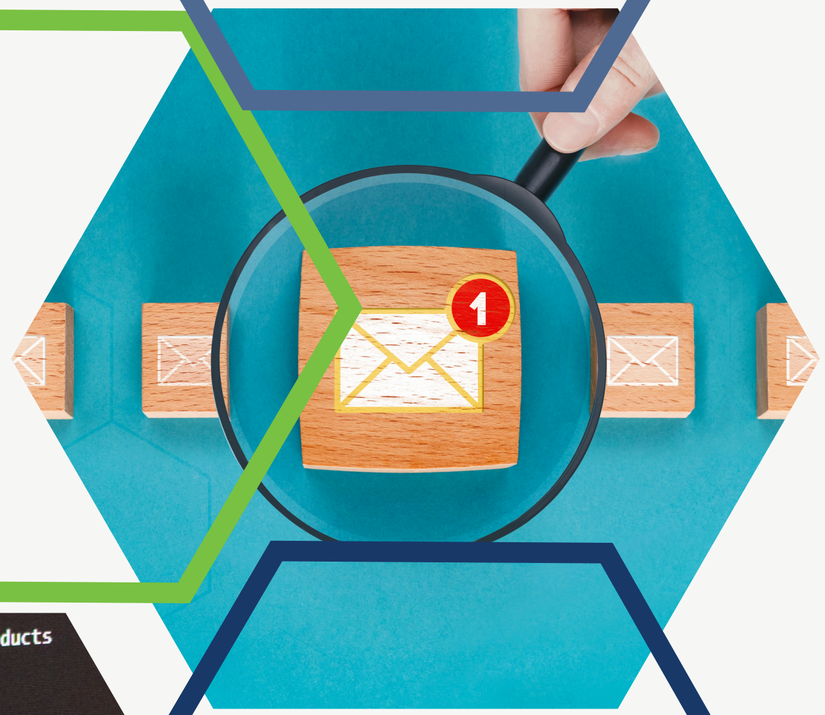
FGF

PROBLEM 5:

Email Overload Management

OUR ALGORITHM:

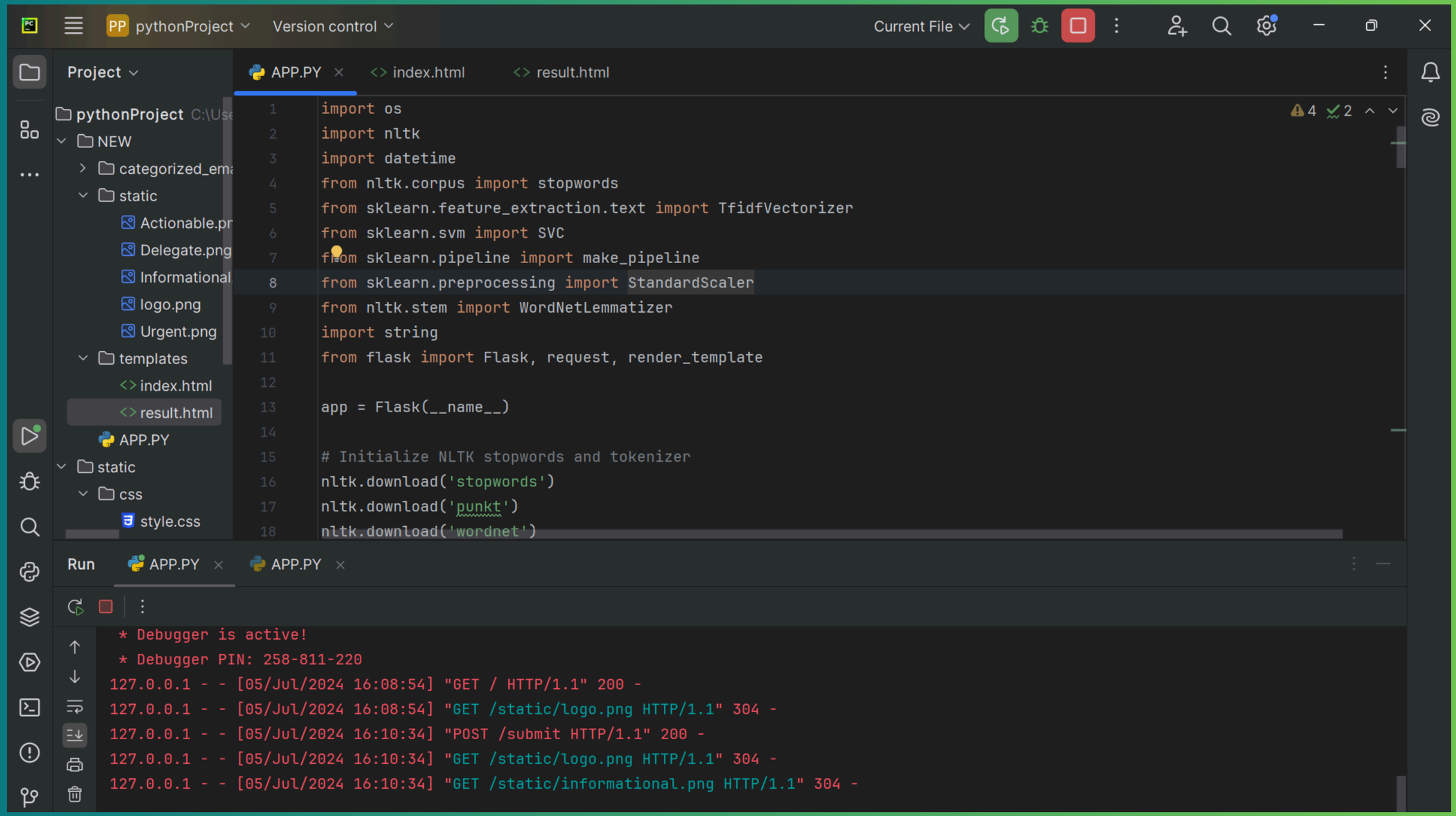
- **Automatically prioritizes** incoming emails based on urgency and relevance
- **Increases Productivity** for employees by displaying critical messages as a priority



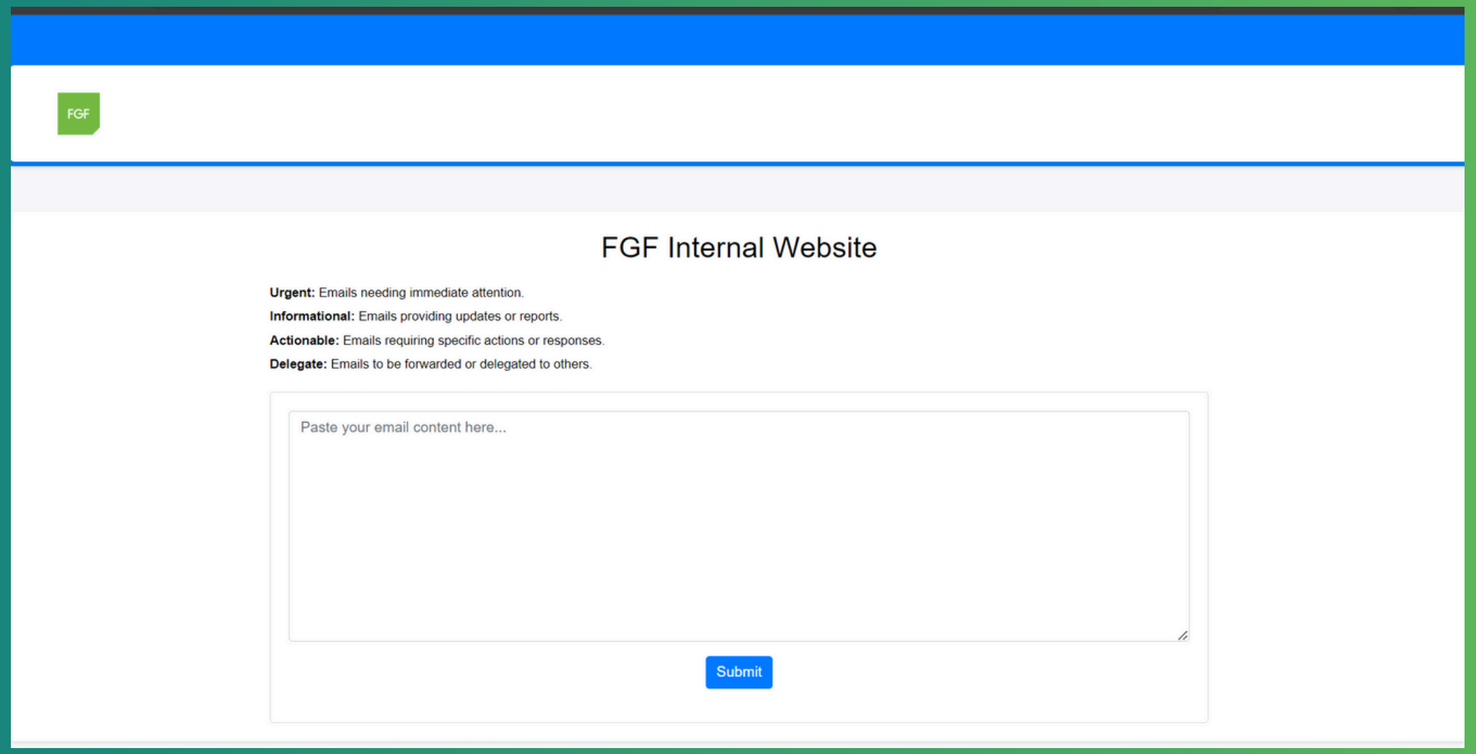
HOW DOES IT WORK?

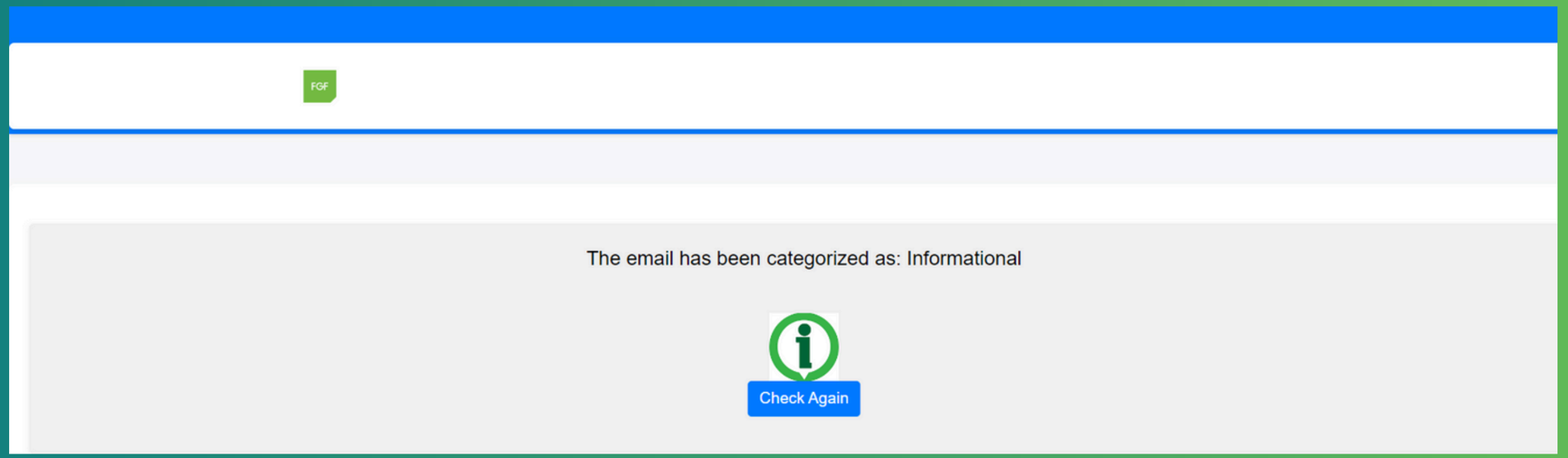
- Using **NLP** (Natural Language Processor) the algorithm categorizes incoming emails into four categories:
  - **Urgent, Informational, Actional, Delegate**
- The model **predicts categories based on email content**
- The user interface provides a form for users to paste email content
  - Upon submission, the **algorithm categorizes the email and saves it** in the appropriate category

CODE


pythonProject
Version control
Current File
Project
pythonProject C:\Use
NEW
categorized_ema
static
Actionable.pr
Delegate.png
Informational
logo.png
Urgent.png
templates
index.html
result.html
APP.PY
static
css
style.css
APP.PY
index.html
result.html
import os
import nltk
import datetime
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from nltk.stem import WordNetLemmatizer
import string
from flask import Flask, request, render_template
app = Flask(__name__)
# Initialize NLTK stopwords and tokenizer
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
Run
APP.PY
APP.PY
* Debugger is active!
* Debugger PIN: 258-811-220
127.0.0.1 - - [05/Jul/2024 16:08:54] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Jul/2024 16:08:54] "GET /static/logo.png HTTP/1.1" 304 -
127.0.0.1 - - [05/Jul/2024 16:10:34] "POST /submit HTTP/1.1" 200 -
127.0.0.1 - - [05/Jul/2024 16:10:34] "GET /static/logo.png HTTP/1.1" 304 -
127.0.0.1 - - [05/Jul/2024 16:10:34] "GET /static/informational.png HTTP/1.1" 304 -

USER INTERFACE





FGF

PROBLEM 3:

Email Reply Suggestions

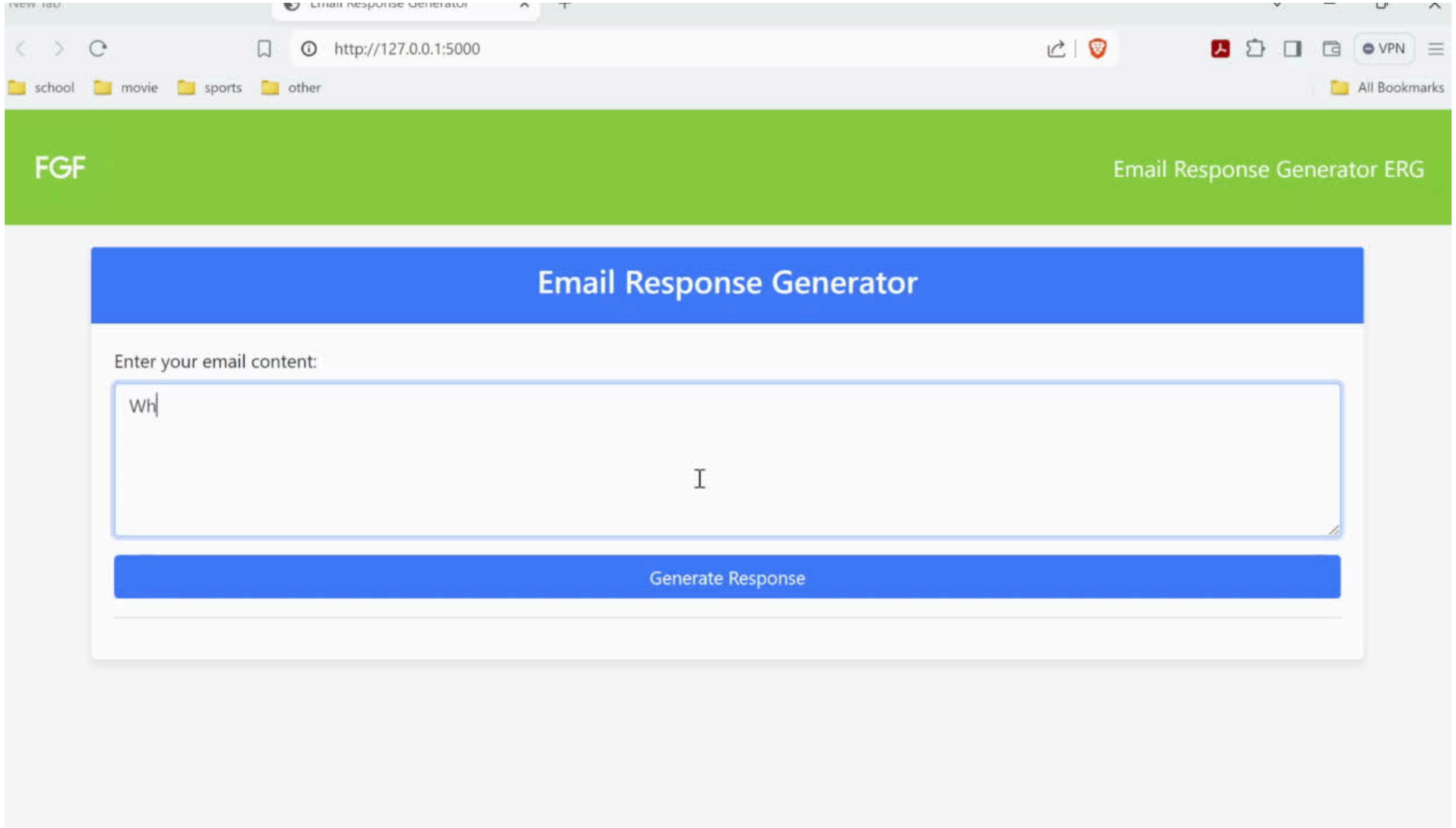
OUR ALGORITHM:

- **Saves Time** by suggesting relevant responses quickly
- **Ensures Consistency** through responses that align with the tone and intent of the email
- **Can Adapt** suggestions based on context of the sender

HOW DOES IT WORK?

- Using predefined patterns, the algorithm automates email responses based on keywords
- The user inputs the email on the interface
- Utilizes NLP using spacy API

USER INTERFACE:



THANK YOU!

-TEAM 5!