BTP/MTP Faculty Allocation System

Rahul Ray

October 26, 2025

Contents

1 Introduction

The BTP/MTP Faculty Allocation System is a Streamlit-based web application designed to automatically allocate students to faculty based on their CGPA and faculty preferences. The system provides a downloadable CSV of allocations along with a faculty preference summary. It is Docker-enabled for easy deployment across different environments and includes logging for error tracking.

2 Project Structure

The project follows a modular and containerized structure:

```
btp_allocation_app/

 app.py                  # Streamlit application
 requirements.txt        # Python dependencies
 Dockerfile              # Docker image for the app
 docker-compose.yml      # Docker Compose setup
 logs/
     app.log             # Log file for errors
```

3 app.py: Streamlit Application and Allocation Logic

The main Python script `app.py` handles the following:

- Streamlit UI for file upload and download
- Logging of errors using Python's `logging` module
- Dynamic detection of faculty columns
- Round-robin allocation algorithm based on CGPA
- Generation of faculty preference summary

3.1 Key Code Snippets

3.1.1 Logger Setup

```
import logging

logging.basicConfig(
    filename="logs/app.log",
    level=logging.INFO,
    format="%(asctime)s - %(levelname)s - %(message)s"
)
logger = logging.getLogger()
```

3.1.2 Allocation Logic

```python
def process_allocation(input_df):
    cols = list(input_df.columns)
    cgpa_index = cols.index("CGPA")
    faculty_cols = cols[cgpa_index + 1:]
    n_faculties = len(faculty_cols)

    students = input_df.sort_values(by="CGPA", ascending=False).
       reset_index(drop=True)

    allocations = []
    for i, row in students.iterrows():
        cycle_pref_index = i % n_faculties + 1
        allocated_fac = None
        for fac in faculty_cols:
            if row[fac] == cycle_pref_index:
                allocated_fac = fac
                break
        if not allocated_fac:
            allocated_fac = faculty_cols[i % n_faculties]
        allocations.append(allocated_fac)

    output_df = students[["Roll", "Name", "Email", "CGPA"]].copy
       ()
    output_df["Allocated"] = allocations

    pref_counts = pd.DataFrame({"Fac": faculty_cols})
    for pref_rank in range(1, n_faculties + 1):
        pref_counts[f"Count Pref {pref_rank}"] = [
            (input_df[fac] == pref_rank).sum() for fac in
               faculty_cols
        ]

    return output_df, pref_counts
```

4 Requirements

```
streamlit
pandas
```

5 Docker Setup

5.1 Dockerfile

```
FROM python:3.10-slim
WORKDIR /app
```

```
COPY . .
RUN pip install --no-cache-dir -r requirements.txt
EXPOSE 8501
CMD ["streamlit", "run", "app.py", "--server.port=8501", "--
    server.address=0.0.0.0"]
```

5.2 docker-compose.yml

```
version: '3.9'

services:
  btp_allocation_app:
    build: .
    container_name: btp_allocation_app
    ports:
      - "8501:8501"
    volumes:
      - .:/app
      - ./logs:/app/logs
    restart: always
```

6 Running Instructions

6.1 Locally Without Docker

1. Install dependencies: `pip install -r requirements.txt`
2. Run the app: `streamlit run app.py`

6.2 Using Docker Compose

1. Install Docker Desktop (includes Docker Compose)
2. Navigate to the project directory
3. Build and run the container: `docker-compose up --build`

7 Conclusion

This project provides a robust, containerized solution for BTP/MTP student-faculty allocation. It supports dynamic faculty detection, CGPA-based sorting, and detailed faculty preference analysis, all with an intuitive web interface.