Read Me

Team Members

Ganesh Jasti
Lakshmi Sankari vissapragada

1. Prerequisites

- Python 3.9+
- Node.js 18+ and npm
- Access to a GMU Hopper for running Mistral-7B
- A Google Places API key

2. Backend Setup (FastAPI + Models)

2.1 Create and Activate Virtual Environment

From the project root:

```
cd backend
python3 -m venv .venv
source .venv/bin/activate
```

Install dependencies:

```
pip install -r requirements.txt
```

2.2 Environment Variables

Create a .env file inside backend/ with:

```
GOOGLE_PLACES_API_KEY=#####
```

google_places.py reads GOOGLE_PLACES_API_KEY to call the API.

2.3 Add Fine-Tuned Models

Create the models/ directory

```
mkdir -p models/bert_pref_extractor
mkdir -p models/mistral7b_lora_itinerary
```

Then:

- Copy **fine-tuned BERT classifier** into models/bert_pref_extractor/
- Copy **Mistral-7B LoRA adapter** into models/mistral7b_lora_itinerary/

2.4 Run the Backend

```
uvicorn main:app --host 0.0.0.0 --port 8000
```

3. Frontend Setup (React)

3.1 Install Dependencies

From the project root:

```
cd frontend
npm install
```

In frontend/src/App.js, the API_URL constant should be set to the currently running backend endpoint on GMU Hopper (this URL changes for each session).

```
const API_URL = "#####”
```

3.2 Production Build for ORC OnDemand Proxy

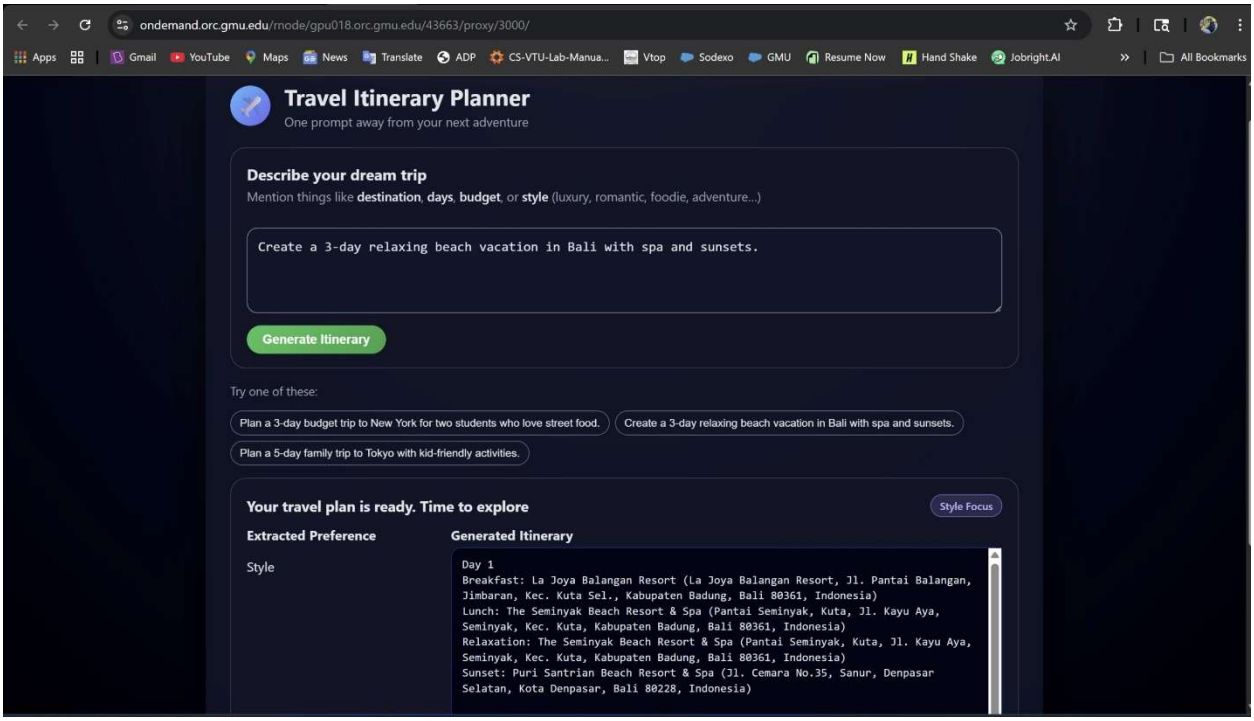
```
npm run build
cd build
python3 -m http.server 3000
```

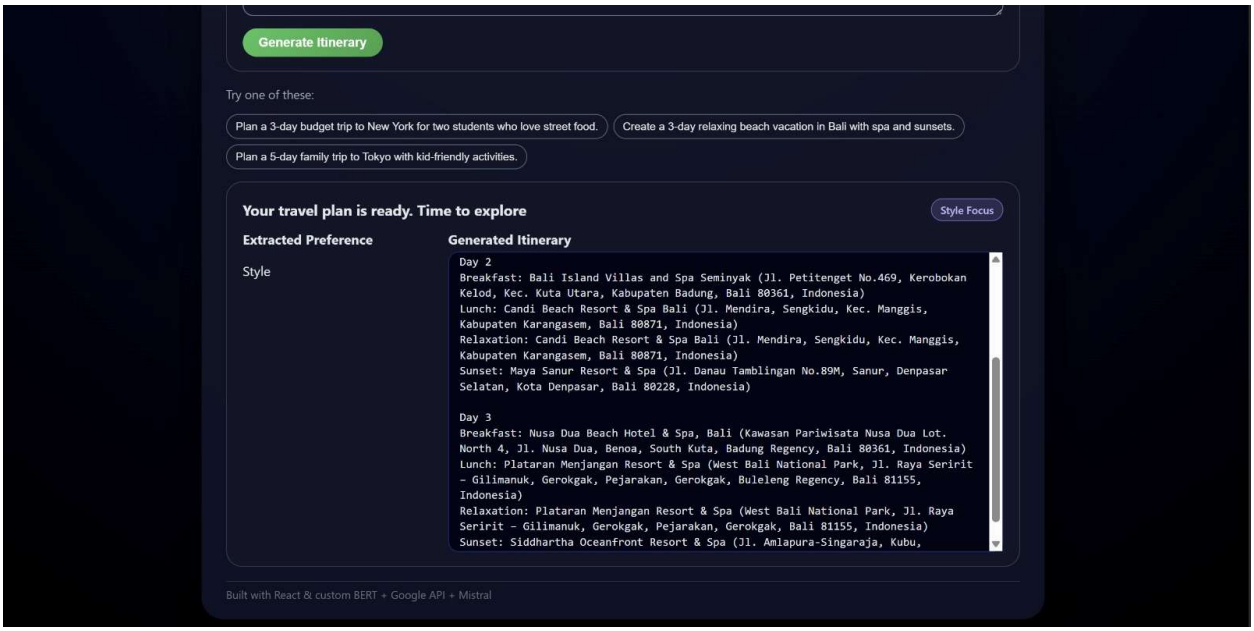
3.3 Run Frontend

```
python3 -m http.server 3000
```

The UI Displays:

1. Text box to type travel query.
2. Generate Itinerary Button.
3. View:
    - Extracted preference (Budget / Style / Destination / Days).
    - Day-by-day itinerary in a scrollable panel, with real venue names and addresses.





4. **Running Evaluation Scripts**

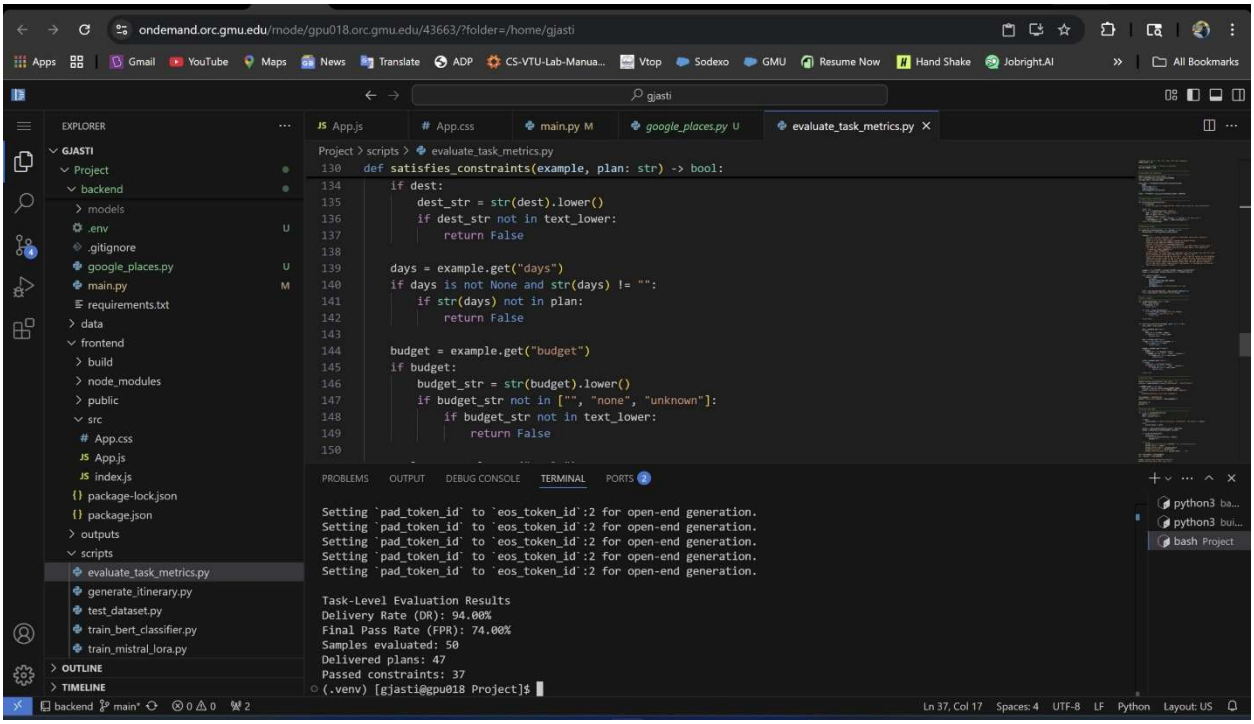
   cd scripts

   python evaluate_task_metrics.py

   This script:

   - Loads the fine-tuned Mistral-7B + LoRA model with extracted preference and google places.
   - Evaluates on a subset of the osunlp/TravelPlanner **test** split.
   - Reports:
     - **Delivery Rate (DR)** – fraction of queries where a structured itinerary with “Day 1 / Day 2 / …” is produced.
     - **Final Pass Rate (FPR)** – fraction of queries where destination, days, budget, and style constraints are satisfied.

   By default, it runs on the first 50 test examples to keep runtime and Google Places cost manageable.



4.1 Training Scripts

python scripts/train_bert_classifier.py

python scripts/train_mistral_lora.py

Here each script saves the best-performing model checkpoints automatically into the corresponding directory under backend/models/ (e.g., backend/models/bert_pref_extractor/ and backend/models/mistral7b_lora_itinerary/), so that the updated weights are picked up by the FastAPI backend without additional configuration.