TiffinTrails: Software Documentation

CSC-510: Software Engineering Project 2

Version 1.0.0

Authors:

Yash Dhavale
Shreeya Ranwadkar
Aditya Deshpande
Ravi Goparaju

Instructor: Professor Tim Menzies
Institution: North Carolina State University

Date: November 3, 2025

Contents

1 Introduction

TiffinTrails is part of the CSC-510 Software Engineering course. The project demonstrates software design, data analysis, and modular development in Python. It focuses on generating, cleaning, and analyzing synthetic restaurant delivery datasets to evaluate efficiency using quantitative performance metrics.

The software emphasizes code modularity, documentation, and reproducible data workflows in line with professional software engineering standards.

Excess food waste and inefficient delivery are major challenges for restaurants and food delivery platforms, increasing costs and environmental impact. **TiffinTrails** tackles this with a simple reward-based system that tracks portion sizes, leftover food, and delivery efficiency. Restaurants and delivery staff earn points for minimizing waste and completing timely deliveries, while minor penalties highlight practices that create excess food or delays. Based on these scores, **TiffinTrails** provides actionable recommendations to improve operations, helping restaurants reduce waste, customers enjoy fresher meals, delivery staff work more efficiently, and the environment benefit from reduced food waste. This system engages all key stakeholders—restaurants, delivery staff, customers, and platform administrators—to collaboratively promote smarter, sustainable food delivery.

Motivation

In the modern food delivery landscape, efficiency metrics are essential for evaluating performance and ensuring customer satisfaction. This project aims to simulate real-world restaurant data pipelines, offering insights into delivery optimization and performance benchmarking.

Objectives

The primary objectives of this project are:

- To design modular, reusable Python components for data analysis.
- To generate and process synthetic datasets that mimic real-world restaurant and delivery operations.
- To compute quantitative efficiency scores for restaurants.
- To demonstrate reproducibility, clarity, and maintainability through strong documentation.

2 Software Overview

2.1 Project Structure

The folder structure below represents the modular organization of the project and its data-driven workflow:

```
Proj2/
```

```
data/                # Contains dataset files
src/                 # Source code
   data_generator.py
   data_loader.py
   delivery_metrics.py
   efficiency_scoring.py
test.txt
README.md
```

2.2 Purpose of Each Module

Each Python script is independent, yet integrates into a unified processing pipeline:

- **data_generator.py:** Generates realistic synthetic datasets for restaurant and delivery analysis.
- **data_loader.py:** Loads, validates, and integrates all generated data into a single clean dataset.
- **delivery_metrics.py:** Computes performance metrics such as average delivery time, on-time rate, and distance.
- **efficiency_scoring.py:** Merges metrics and metadata to compute an overall restaurant efficiency score.

Each component contributes to a seamless data flow from raw synthetic data generation to final analytical insights.

3 Quick Start Guide

This section provides a concise "quick start" walkthrough for new users to get TiffinTrails running locally.

3.1 Prerequisites

- Python 3.8 or higher
- Required libraries:

  ```
  pandas
  numpy
  ```

- Ensure that Git and pip are properly configured on your system.

3.2 Installation

Follow these steps to set up and deploy the software:

```
# Clone the repository
git clone https://github.com/YashDhavale/CSC-510-SE-Project.git

# Navigate to project directory
cd CSC-510-SE-Project/Proj2

# Install dependencies
pip install -r requirements.txt
```

After installation, verify the environment by executing:

```
python --version
pip list
```

3.3 Running the Project

The complete execution workflow is divided into four clear stages:

```
# Step 1: Generate synthetic datasets
python src/data_generator.py

# Step 2: Load, clean, and integrate datasets
python src/data_loader.py

# Step 3: Compute delivery performance metrics
python src/delivery_metrics.py

# Step 4: Calculate efficiency scores for restaurants
python src/efficiency_scoring.py
```

Each of these stages can be run independently or as part of an automated pipeline. The generated outputs are stored under the 'data/' directory for transparency and reproducibility.

4 Detailed Documentation

This section provides comprehensive explanations of each source file, their major functions, parameters, and intended outputs.

4.1 data_generator.py

Key Functions:

- `generate_waste_data(n)` – Creates weekly food waste dataset.
- `generate_restaurant_metadata()` – Builds restaurant profiles.
- `generate_customer_feedback()` – Simulates customer review data.

- `generate_menu_portions()` – Generates menu item data.
- `generate_delivery_logs()` – Produces delivery activity logs.

Usage:

```
from src.data_generator import main
main()  # generates all datasets under /data
```

Each generated dataset is saved as a CSV file under the `data/` directory and can be reused across modules.

4.2 data_loader.py

Key Functions:

- `load_csv(name)` – Loads CSV files into pandas DataFrames.
- `basic_clean(df)` – Cleans string values and removes duplicates.
- `integrate_all()` – Merges all datasets into one unified table.

Usage:

```
from src.data_loader import integrate_all
merged_df = integrate_all()
```

This module ensures dataset integrity before analytical processing.

4.3 delivery_metrics.py

Purpose: Processes delivery data to calculate vendor KPIs such as:

- Average delivery time
- On-time delivery rate
- Average delivery distance
- Deliveries per day

Usage:

```
from src.delivery_metrics import compute_delivery_metrics
compute_delivery_metrics("data/Delivery_Logs.csv", "data/
   vendor_delivery_metrics.csv")
```

4.4 efficiency_scoring.py

Purpose: Combines delivery metrics with restaurant metadata to compute a normalized efficiency score.

Formula:

Efficiency Score = $0.4(\text{on\_time\_rate}) + 0.3(1 - \text{norm\_delivery\_time}) + 0.2(1 - \text{norm\_distance}) + 0.1(\text{norm\_del}$

Usage:

```
from src.efficiency_scoring import compute_efficiency_scores
compute_efficiency_scores("data/vendor_delivery_metrics.csv",
                          "data/Restaurant_Metadata.csv",
                          "data/vendor_efficiency_scores.csv")
```

This computation allows the evaluation of delivery efficiency in a quantitative, comparable form.

5 Troubleshooting

- **ModuleNotFoundError:** Ensure all dependencies are installed via `pip install -r requirements.txt`.
- **FileNotFoundError:** Run `data_generator.py` before executing downstream modules.
- **Empty Dataset:** Check that the data directory is correctly generated and accessible.

If persistent issues occur, verify your Python environment or file paths, and consult GitHub issues for support.

6 Release History

- **v1.0.0 (Initial Release)** Complete modular pipeline for generating, cleaning, and analyzing synthetic data for restaurant efficiency scoring.

Future versions may include:

- Visualization dashboards for performance analysis.
- Expanded dataset schema for real-world integration.
- API endpoints for interactive access.

7 Revision Control

Documentation and source code are managed under GitHub version control. All revisions, commits, and updates are publicly visible in: `https://github.com/YashDhavale/CSC-510-SE-Project`

Each commit is associated with a descriptive message that records bug fixes, enhancements, or documentation improvements. Version control ensures traceability and collaborative reliability.

8 Conclusion

TiffinTrails integrates data engineering and analytics to evaluate restaurant performance efficiently. The modular architecture and documentation follow modern software engineering standards, supporting reusability, clarity, and open-source best practices.

This project highlights the team's ability to design structured, traceable, and extensible software, combining automation with analytics for better data-driven decisions.

9 References

- Python Software Foundation. Python Language Reference, version 3.10.
- `https://pandas.pydata.org/`
- `https://numpy.org/`