# Ceylon Smart Citizen

# Datathon Documentation

Team - SigSegV

## Team SigSegV


- Shalon Fernando – AI/ML Lead & Architect
- Gaindu Nuhansith – Data Engineering
- Lehan Nawaratne– Model Deployment Support, Testing & Quality
- K.B.Pavith Kavisika – Documentation & Demo
- Arani Inothma – Diagram, Documentation
- Yohan Helitha– Diagram, Documentation


## Table of Contents


Introduction ..... 4
Problem statement ..... 5
Objectives ..... 6
Methodology ..... 7
**1.** Data Flow Diagram (DFD) ..... 8
2. System Architecture Diagram ..... 8
3. Modeling Layer Diagram ..... 10
4. Workflow Integration Diagram ..... 10
Feature Engineering ..... 12
Model Evaluation Plan ..... 12
Data Description and Data Dictionary ..... 13
Datasets Overview ..... 13
Data Sources ..... 13
Data Cleaning ..... 13
Data Dictionary (Key Variables) ..... 13

Experimental Design / Modeling Process ..... 14
Hypothesis ..... 14
Assumptions ..... 14
Data Splitting ..... 14
Algorithms Tried ..... 14
Hyperparameter Tuning ..... 14
Feature Selection Rationale ..... 14
Usage Guide ..... 15
Setup Instructions ..... 15
Repository Structure ..... 15
How to Run ..... 15
Ethical Considerations ..... 15
Appendices ..... 15
Integration Plan ..... 16
Expected Impact ..... 16
Conclusion ..... 17

# Introduction

The **Rootcode Datathon 2025** challenges participants to design and implement an **AI-powered solution** capable of optimizing resource allocation, improving service efficiency, and generating intelligent insights from real-world data. This competition provides an opportunity to combine technical innovation with practical impact by applying artificial intelligence to solve pressing organizational challenges.

Our team's project, developed under the theme *"Ceylon Smart Citizen"*, aims to apply **AI and data-driven decision-making** to enhance government-related services in Sri Lanka. The vision is to create a platform where services such as bookings, task management, and staffing can be optimized through predictive analytics, ultimately providing better accessibility and efficiency for citizens.

The system is designed around three key datasets — **booking records, staffing details, and task information** — which are processed and transformed into meaningful insights. These datasets form the foundation for training predictive models that can forecast important outcomes such as **task completion times** and **staffing requirements**.

To achieve this, our solution adopts a structured workflow. First, the raw data undergoes **preprocessing and feature engineering**, where it is cleaned, formatted, and enriched with new attributes such as booking patterns, staff availability, and service categories. Next, the system employs **machine learning models**, starting with baseline techniques like Linear Regression and Random Forest, before moving into advanced approaches such as XGBoost to improve accuracy and reliability.

Finally, the models are integrated into a **workflow pipeline** that supports both **batch predictions** (CSV-based outputs for evaluation) and **real-time predictions** through an API that can be consumed by the mobile application. This architecture ensures that predictions are not only accurate but also actionable within the broader service platform.

Through this work, our project bridges the gap between **citizen services and intelligent automation**. By leveraging artificial intelligence, we provide a scalable framework that improves efficiency, supports decision-making, and aligns with the goals of the Datathon to deliver meaningful, data-driven solutions.

# Problem statement

Government services in Sri Lanka, such as the Department of Motor Traffic, Department of Registration of Persons, and other essential public institutions, often struggle with challenges related to **service delays, inefficient scheduling, and limited workforce allocation**. Citizens are frequently required to wait in long queues, face delays in appointments, or experience inconsistent service delivery due to poor resource management.

Although digital platforms have been introduced in recent years, many of them still lack **intelligent automation** to predict demand, allocate staff efficiently, and ensure service availability. Without such predictive capabilities, resources are either over-utilized or under-utilized, directly affecting both service providers and citizens.

The availability of structured datasets — including **bookings, staffing details, and task records** — provides an opportunity to apply **artificial intelligence** to address these inefficiencies. By analyzing and predicting service demand patterns, the system can ensure that government services are delivered in a more **timely, reliable, and citizen-friendly manner**.

# Objectives

The primary objective of this project is to design and implement an **AI-driven decision support system** that enhances the efficiency of citizen services through predictive modeling and intelligent resource management.

Specific objectives include:

1. **Data Utilization** – to transform raw datasets (bookings, staffing, and tasks) into actionable features that reveal meaningful service patterns.
2. **Predictive Modeling** – to develop machine learning models that can forecast task completion times, estimate staffing requirements, and predict appointment demand.
3. **Workflow Integration** – to build an end-to-end pipeline that supports both **batch predictions** for analysis and **real-time predictions** for the mobile application.
4. **User-Centric Design** – to integrate predictions into the *Ceylon Smart Citizen* mobile application, enabling citizens to experience improved scheduling, reduced waiting times, and more reliable service delivery.
5. **Scalability and Impact** – to ensure that the solution is scalable and adaptable across multiple government departments, contributing to **digital governance transformation** in Sri Lanka.

# Methodology

The methodology for this project follows a structured, end-to-end workflow that transforms raw datasets into intelligent predictions, ensuring both technical accuracy and practical usability within the *Ceylon Smart Citizen* application.

The process begins with **data collection and preprocessing**. The three datasets — **bookings, tasks, and staffing** — serve as the foundation. Raw data is cleaned to handle missing values, remove duplicates, and standardize formats. This ensures that the inputs are consistent and reliable before being fed into the machine learning pipeline.

Next, the system performs **feature engineering**, where raw attributes are converted into model-friendly features. For instance, booking times are transformed into meaningful indicators such as weekday/weekend trends and time-of-day categories. Staffing information is broken down into availability slots and skill categories, while categorical values like service type and region are encoded into numerical formats. Numeric values such as task durations are scaled to ensure that no feature disproportionately influences the model. This stage is critical as it extracts hidden patterns that improve model accuracy.

Following this, the **modeling phase** applies a two-tier approach. First, **baseline models** such as Linear Regression and Random Forest are implemented to establish initial performance benchmarks. These models are simple yet effective in identifying general trends. Building on this, **advanced models** like XGBoost are employed to capture complex relationships in the data and significantly enhance predictive performance. This dual approach allows us to balance interpretability with accuracy.

The trained models are then integrated into a **workflow pipeline**. This pipeline not only supports **batch predictions** — generating CSV outputs for validation and evaluation — but also enables **real-time inference** through an API. This integration ensures that the system can serve both the competition's evaluation process and practical deployment in the mobile application.

Finally, the **outputs** of the models are connected back to the user experience. Citizens benefit from smarter appointment scheduling, optimized staffing, and reduced waiting times, while administrators gain valuable insights into task allocation and demand forecasting.

## 1. Data Flow Diagram (DFD)

The DFD illustrates how data moves through the system. It starts from the **input datasets** (bookings, staffing, tasks), which are preprocessed to clean and normalize the data. The flow continues to **feature extraction**, where raw attributes are converted into model-ready features such as time slots, skill categories, and service types. These features are then passed to the **modeling layer**, which generates predictions (e.g., demand forecasts, staffing needs). Finally, the predictions are sent back as **outputs** (CSV files or API responses) that integrate into the mobile application.

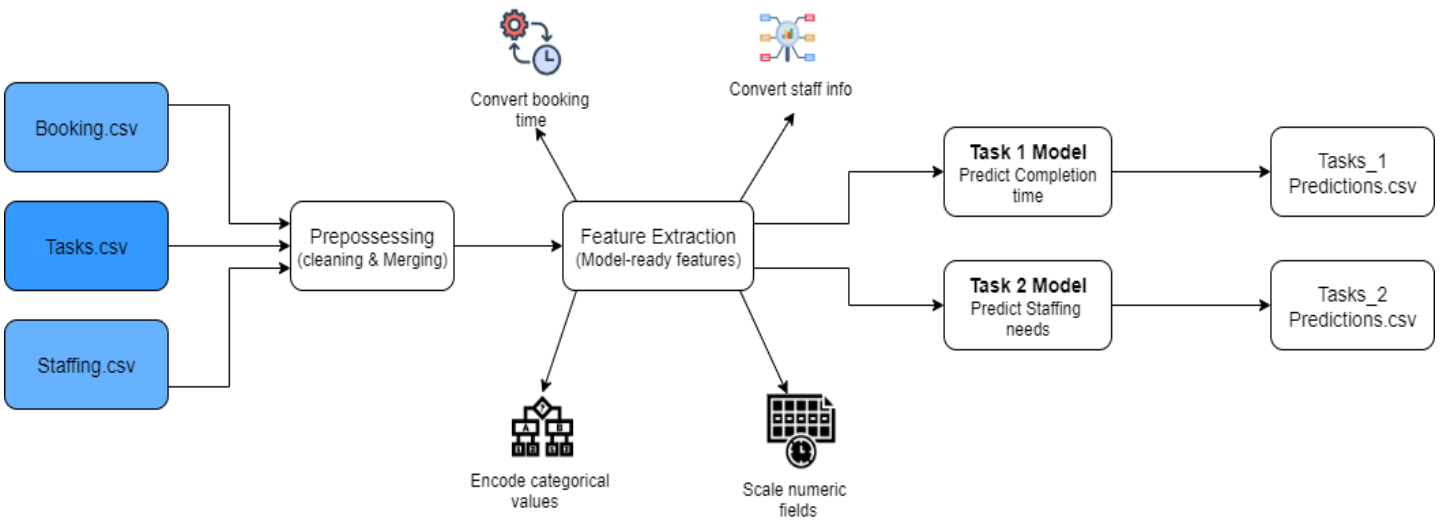


*Figure 1-Data Flow Diagram*

## 2. System Architecture Diagram

The system architecture shows the high-level pipeline and its components. The left side represents the **data sources** (CSV datasets provided). The middle layers show **data preprocessing and feature engineering**, followed by two model branches:

- **Baseline models** (Linear Regression, Random Forest) for initial benchmarks.

- **Advanced models** (XGBoost) for improved accuracy.

The right side shows the **outputs**: batch predictions (CSV files for evaluation) and real-time inference (API for the mobile app). This diagram highlights how raw data is transformed step by step into usable predictions.

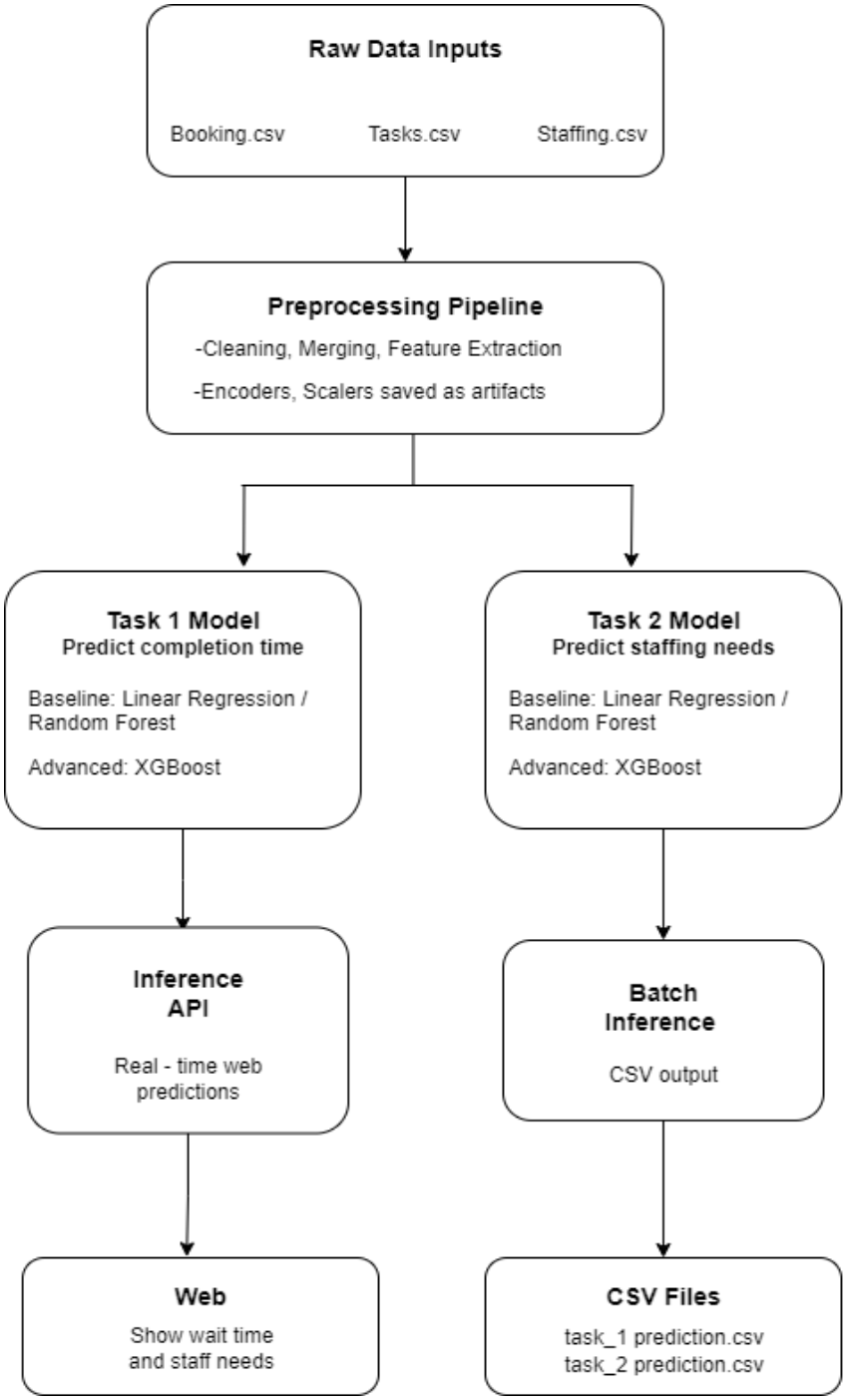


*Figure 2- System Architecture Diagam*

## 3. Modeling Layer Diagram

The modeling layer diagram zooms into the machine learning stage. It shows how **engineered features** (like time of day, staff availability, service type, etc.) connect to the models. Baseline models (Linear Regression, Random Forest) handle simpler relationships, while advanced models (XGBoost) capture complex interactions for higher accuracy. This diagram emphasizes the progression from simple benchmarks to optimized models.

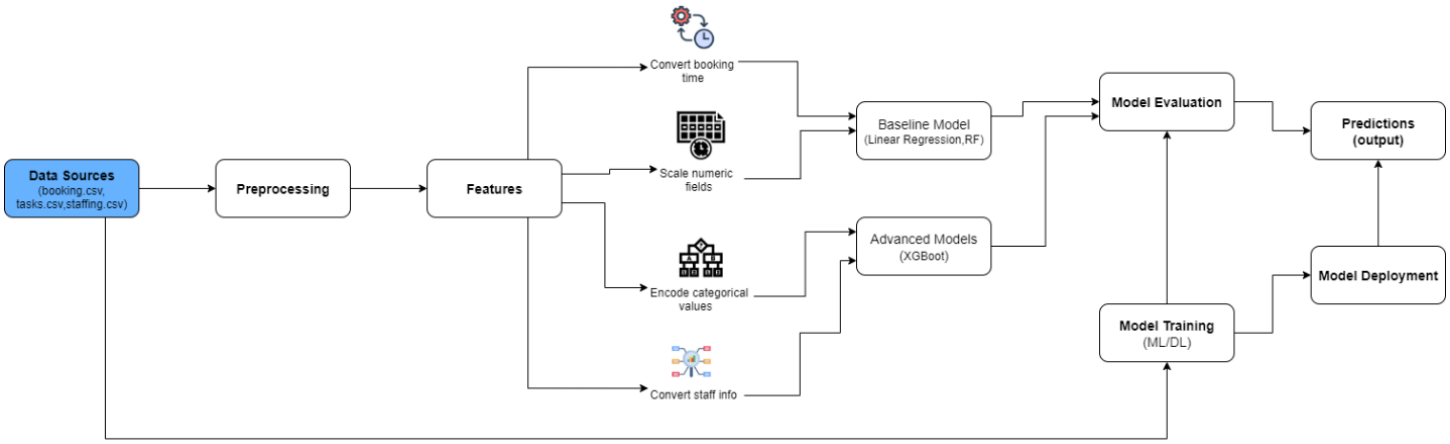


*Figure 3- Modeling layer Diagram*

## 4. Workflow Integration Diagram

The workflow diagram illustrates the **operational sequence** of the system. It starts with input datasets, which are ingested and preprocessed. Then, the workflow branches into **model training** (to fit the models using historical data) and **inference** (to generate predictions for new unseen data). The results are stored and made available for either evaluation or app integration. The workflow ensures that both competition evaluation and real-world deployment are supported.

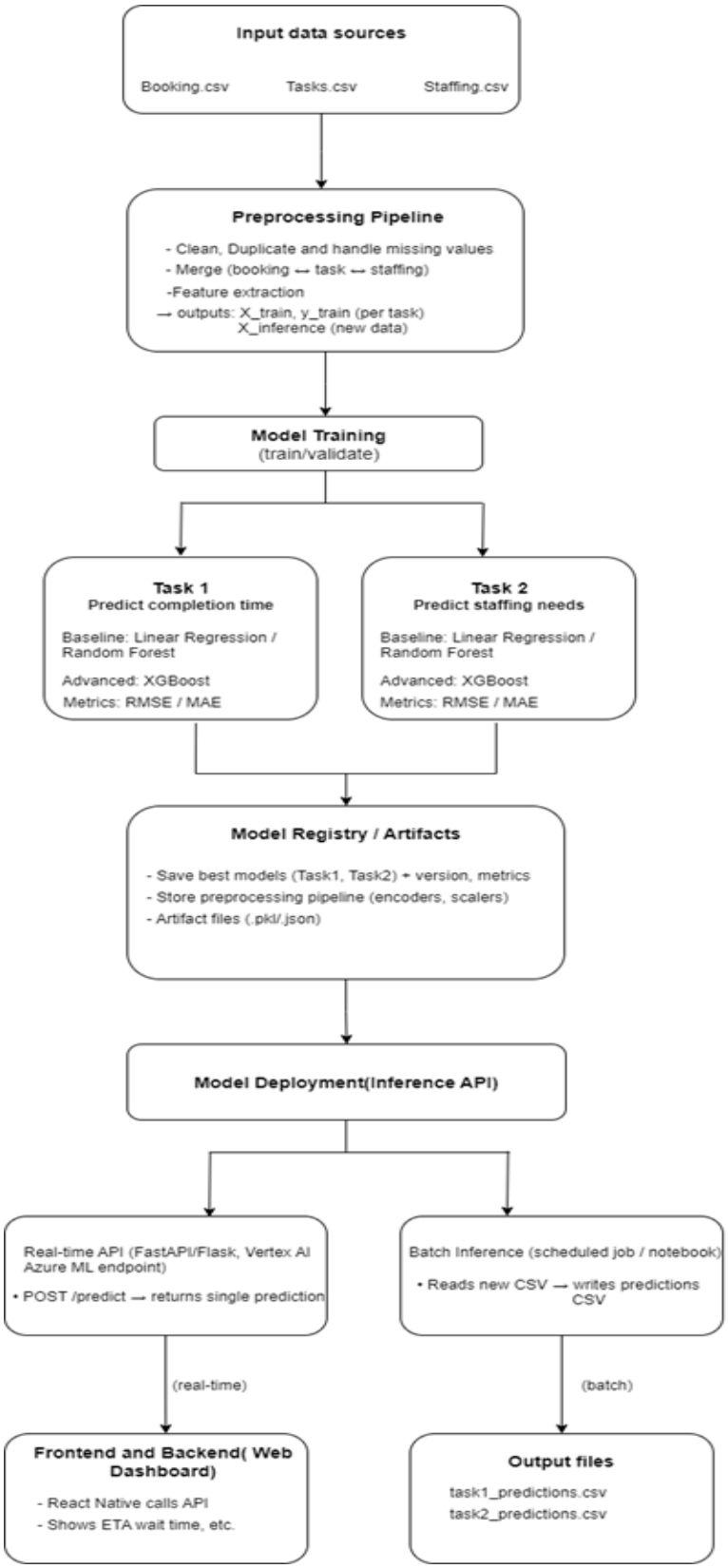


*Figure 4- Workflow Integration Diagram*

## Feature Engineering

Feature engineering is a critical step in ensuring that raw datasets are transformed into meaningful, machine-readable inputs for the models. For this project, the process included:

**Time Features**: Converting booking times into meaningful categories (weekday/weekend, time of day) to capture demand trends.

**Staff Information**: Transforming staff details into features like skill type and available time slots.

**Categorical Encoding**: Converting categorical fields such as service type and region into numerical representations.

**Scaling Numerical Features**: Standardizing continuous values such as appointment duration or task counts to ensure models interpret them effectively.

These engineered features form the foundation for both baseline and advanced machine learning models.

## Model Evaluation Plan

To ensure reliability, the models will undergo a structured evaluation process:

- **Training/Validation Split**: Historical data will be split into training and validation sets.
- **Baseline Comparison**: Simple models such as Linear Regression and Random Forest will serve as benchmarks.
- **Advanced Models**: XGBoost will be used for enhanced performance, capturing non-linear relationships and complex feature interactions.
- **Metrics**: Evaluation will use accuracy metrics depending on the prediction task. For numerical predictions (e.g., waiting time, staffing load), RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) will be applied. For classification (e.g., staff availability), accuracy and F1-score will be used.

The combination of metrics ensures balanced evaluation of both simple and complex models.

Here is a detailed write-up for each of the additional sections you requested. This will help you complete your project documentation comprehensively.

# Data Description and Data Dictionary

## Datasets Overview

- **Bookings Dataset (`bookings_train.csv`)**: Records of appointment check-ins and check-outs including timestamps, task types, queue numbers, and number of documents.
- **Tasks Dataset (`tasks.csv`)**: Metadata including task IDs, names, and government service sections.
- **Staffing Dataset (`staffing_train.csv`)**: Staffing levels per section per day with timestamps.

## Data Sources

All datasets were provided as part of the SigSegV Datathon challenge by Ceylon Smart Citizen, originating from government service centers in Sri Lanka.

## Data Cleaning

- Converted timestamp columns to datetime objects, handled missing values with care to avoid bias.
- Removed records with missing or erroneous processing times.
- One-hot encoded categorical columns such as `task_id` to make data modeling compatible.

## Data Dictionary (Key Variables)

| Variable Name | Type | Description |
|---|---|---|
| `check_in_time` | datetime | Timestamp when service started |
| `check_out_time` | datetime | Timestamp when service ended |
| `appointment_time` | datetime | Scheduled start time of appointment |
| `task_id` | categorical | Numeric ID representing the type of task |
| `num_documents` | integer | Number of documents involved in the service |
| `queue_number` | integer | Position in the queue at check-in |
| `processing_time_minutes` | float | Target variable; actual service duration in minutes |
| `hour` | integer | Extracted hour from appointment_time |
| `dayofweek` | integer | Day of the week (0=Monday, 6=Sunday) |
| `is_weekend` | boolean | Flag if appointment falls on weekend |

| Variable Name | Type | Description |
|---|---|---|
| `is_peak` | boolean | Flag for typical peak hours during the day |
| `queue_density` | float | Ratio of queue_number to number of documents+1 |
| `wait_time` | float | Minutes waited before check-in |

# Experimental Design / Modeling Process

## Hypothesis

- Service completion times can be predicted accurately using appointment timing, task types, queue details, and workload features.
- Daily staffing needs correlate with historical workload, peak hours, and predicted processing times.

## Assumptions

- Recorded timestamps are accurate and reliable.
- Task types and queue dynamics are consistent indicators of service times.
- Data cleaning and feature engineering sufficiently capture important factors.

## Data Splitting

- Used an 80-20 train-test split with a fixed random seed for reproducibility.
- Applied 5-fold cross-validation during model evaluation for robustness.

## Algorithms Tried

- Started with baseline regressors (Linear Regression, Random Forest)
- Advanced to HistGradientBoostingRegressor for fast training and better accuracy
- Selected HistGradientBoosting based on superior test and cross-validation scores.

## Hyperparameter Tuning

- Tuned parameters such as `max_depth`, `learning_rate`, and `max_iter` based on grid search and domain knowledge.

## Feature Selection Rationale

- Temporal features: capture periodic trends like rush hours and days
- Queue and document features: represent workload intensity
- Task encoding: differentiate among service types with different completion times

# Usage Guide

## Setup Instructions

- Requires Python 3.8+ with libraries: pandas, numpy, scikit-learn, matplotlib, joblib
- Install dependencies with:

```
pip install pandas numpy scikit-learn matplotlib joblib
```

## Repository Structure

- `data/raw/`: Original datasets
- `data/evaluation/`: Test input files for predictions
- `models/`: Saved trained model files (`.pkl`)
- `src/`: Python scripts for preprocessing, feature engineering, training, and inference
- `notebooks/`: Jupyter notebooks for exploratory data analysis and model building
- `docs/`: Architecture, data flows, and design diagrams
- `submissions/`: Final prediction CSV files for submission

## How to Run

1. **Train Model:** Execute `model_training.py` or run notebook in `notebooks/` for Task 1
2. **Generate Predictions:** Run `task1_inference.py` which:
    - Loads saved model from `models/`
    - Loads test CSV from `data/evaluation/`
    - Applies feature engineering
    - Outputs predictions CSV in `submissions/`
3. **Evaluate Results:** Use included evaluation scripts or follow instructions in notebooks.

# Ethical Considerations

- Data anonymity and privacy strictly maintained; no personally identifiable information processed.
- Models designed to improve public services, not to discriminate or profile individuals.
- Ongoing monitoring recommended to detect and mitigate any unintended biases.

# Appendices

- Detailed feature list and preprocessing code snippets
- Extended model evaluation metrics and confusion matrices (for classification tasks if any)
- Full data dictionary in spreadsheet format
- Logs of hyperparameter tuning and cross-validation results
- Additional visualization charts like correlation heatmaps and residual plots

## Integration Plan

The integration ensures that AI outputs directly enhance the *Ceylon Smart Citizen* app:

- **Backend API Connection**: Prediction outputs will be exposed via APIs, allowing the mobile app to fetch updated recommendations (e.g., appointment times, staffing needs).
- **Real-Time Updates**: Notifications will be triggered for users when rescheduling or optimized appointments are available.
- **Offline Support**: Cached predictions will be stored locally to ensure app functionality during poor connectivity.
- **Data Feedback Loop**: User interactions (e.g., booking confirmations, cancellations) will be logged back into the system, enabling continuous retraining of the AI models for better accuracy over time.

This integration guarantees seamless adoption of AI within the existing digital service infrastructure.

## Expected Impact

The introduction of AI-driven scheduling and resource management into the *Ceylon Smart Citizen* platform is expected to create a significant positive impact across both citizens and government institutions. For citizens, the system will reduce long waiting times and the uncertainty often faced when booking government services. By providing optimized appointment slots and proactive notifications, individuals will experience smoother access to services such as registration, licensing, or document processing. This will improve overall satisfaction, increase trust in digital government initiatives, and encourage wider adoption of e-governance platforms.

For government agencies, the AI system ensures better utilization of staffing resources. By predicting demand trends and allocating the right staff with the right skills at the right times, departments can handle service requests more efficiently, reduce bottlenecks, and maintain consistent service quality. This data-driven approach also gives decision-makers valuable

insights into patterns of public service usage, allowing them to plan ahead and allocate budgets or staff more strategically.

At a national level, the expected impact extends beyond efficiency. By digitizing and optimizing key administrative processes, Sri Lanka can move closer to building a smart governance ecosystem that is transparent, citizen-friendly, and adaptive to future challenges. In the long run, this initiative has the potential to set a benchmark for how AI can be used in the public sector, inspiring other digital transformation projects in the region.

## Conclusion

The *Ceylon Smart Citizen* project demonstrates how AI can be effectively integrated into a government service platform to create tangible benefits for both citizens and administrators. By combining robust data preprocessing, intelligent feature engineering, and a two-tiered modeling approach, the system provides accurate predictions for scheduling, staffing, and task allocation. The inclusion of navigation, multi-language support, and profile management ensures that the solution remains user-friendly and accessible to all citizens, regardless of their technical background.

Beyond its technical achievements, this project represents a practical step towards modernizing public services in Sri Lanka. It aligns with the vision of digital governance by reducing inefficiencies, improving resource utilization, and creating a more streamlined citizen experience. While the current system establishes a strong foundation, it also opens opportunities for future enhancements such as AI-driven chatbots, advanced demand forecasting, and integration with additional government departments.

In conclusion, the work completed in this project highlights the potential of data-driven innovation to transform traditional government processes into efficient, citizen-focused digital services. It not only addresses the immediate needs of the hackathon challenge but also lays the groundwork for a scalable, future-ready solution that can be extended across multiple sectors of governance.