Project Title	**Amazon Delivery Time Prediction**
Skills take away From This Project	Python scripting, data cleaning, Exploratory Data Analysis (EDA), Machine Learning,Regression modeling,MLflow, Streamlit,
Domain	Domain E-Commerce and Logistics

Problem Statement

This project aims to predict delivery times for e-commerce orders based on a variety of factors such as product size, distance, traffic conditions, and shipping method. Using the provided dataset, learners will preprocess, analyze, and build regression models to accurately estimate delivery times. The final application will allow users to input relevant details and receive estimated delivery times via a user-friendly interface.

Business Use Cases

Enhanced Delivery Logistics:

- Predict delivery times to improve customer satisfaction and optimize delivery schedules.

Dynamic Traffic and Weather Adjustments:

- Adjust delivery estimates based on current traffic and weather conditions.

Agent Performance Evaluation:

- Evaluate agent efficiency and identify areas for training or improvement.

Operational Efficiency:

- Optimize resource allocation for deliveries by analyzing trends and performance metrics.

Approach

1. **Data Preparation:**
    - Load and preprocess the dataset.
    - Handle missing or inconsistent data.
    - Perform feature engineering (e.g., calculating distance between store and drop locations).
2. **Data Cleaning:**
    - Remove duplicates and handle missing values.
    - Standardize categorical variables (e.g., weather, traffic).
3. **Exploratory Data Analysis (EDA):**
    - Analyze trends in delivery times, agent performance, and external factors.
    - Visualize the impact of traffic, weather, and other variables on delivery times.
4. **Feature Engineering:**
    - Calculate geospatial distances using store and drop coordinates.
    - Extract time-based features (e.g., hour of day, day of the week).
5. **Regression Model Development:**
    - Train multiple regression models, including:
        - Linear Regression
        - Random Forest Regressor
        - Gradient Boosting Regressor
    - Evaluate models using metrics like RMSE, MAE, and R-squared.
    - Compare models and track performance metrics using MLflow.
6. **Application Development:**
    - Build a user interface using Streamlit to:
        - Input order details (e.g., distance, traffic, weather, etc.).
        - Display predicted delivery times.
7. **Model Comparison and Tracking:**
    - Use MLflow to log, compare, and manage different regression models.
    - Document the hyperparameters, performance metrics, and model versions.
8. **Deployment:**
    - Deploy the application in streamlit for accessibility and scalability.

Dataset

amazon_delivery.csv

Dataset Explanation

The dataset contains detailed information about orders, agents, and delivery conditions:

- **Order_ID:** Unique identifier for each order.
- **Agent_Age**: Age of the delivery agent.
- **Agent_Rating:** Rating of the delivery agent.
- **Store_Latitude/Longitude:** Geographic location of the store.
- **Drop_Latitude/Longitude:** Geographic location of the delivery address.
- **Order_Date/Order_Time:** Date and time when the order was placed.
- **Pickup_Time:** Time when the delivery agent picked up the order.
- **Weather:** Weather conditions during delivery.
- **Traffic:** Traffic conditions during delivery.
- **Vehicle:** Mode of transportation used for delivery.
- **Area:** Type of delivery area (Urban/Metropolitan).
- **Delivery_Time:** Target variable representing the actual time taken for delivery (in hours).
- **Category:** Category of the product being delivered.

Data Flow and Architecture

1. **Data Preparation:**
    - Load and preprocess the dataset.
    - Store processed data locally.
2. **Processing Pipeline:**
    - Perform feature engineering and data preprocessing.
    - Train and save regression models.
3. **Model Training:**
    - Use libraries like scikit-learn and XGBoost for model development.

    - Track model training and evaluation in MLflow.
    - Save trained models for deployment.
4. **Deployment:**

    - Use Streamlit to create a user-friendly front end.

Exploratory Data Analysis (EDA)

Analyze trends and insights in the dataset, including:

- Distribution of delivery times.
- Impact of weather and traffic on delivery times.
- Relationship between distance and delivery time.
- Agent performance across various conditions.

Key Visualizations:

- Bar charts for delivery times by product category.
- Scatter plots for distance vs. delivery time.
- Heatmaps to visualize correlations (e.g., agent rating and delivery time).

Results

By the end of this project, learners will achieve:

1. A cleaned and processed dataset ready for modeling.
2. Multiple regression models developed and tracked using MLflow.
3. Insights into the factors influencing delivery times.
4. A functional delivery time prediction system accessible via a Streamlit interface.

Project Evaluation Metrics:

Data Preparation:

- Evaluate how well the dataset is cleaned and processed.

Feature Engineering:

- Assess the quality of derived features like distance and time-based variables.

Model Performance:

- Evaluate prediction accuracy using metrics like RMSE, MAE, and R-squared.

Application Functionality:

- Ensure the Streamlit interface is user-friendly and responsive.

Model Tracking:

- Verify that MLflow accurately logs and tracks all models and their metrics.

Technical Tags: Python, Regression Modeling, Data Cleaning, Feature Engineering, Streamlit, MLflow

Deliverables:

1. **Source Code:**
    - Python scripts for data preparation, EDA, feature engineering, and model development.
2. **Processed Data:**
    - Cleaned and prepare dataset ready for analysis.
3. **Regression Models:**
    - Trained and evaluated models for delivery time prediction.
4. **Application Code:**
    - Streamlit app code with functional prediction capabilities.
5. **Documentation:**
    - Detailed documentation explaining the implementation and results.
6. **Model Tracking::**
    - MLflow logs and comparison of all regression models.

Timeline (14days) : To be defined based on the project’s milestones and submission deadlines.