# Introduction

Our project steps, obstacles, and impending tasks are in this report. A user-friendly machine learning-powered web application with great data preprocessing and robust machine learning models is the project's goal. For back-end integration, data gathering, preparation, exploratory data analysis (EDA), and technological hurdles.

# Progress So Far

**Front-End Development**

Designing the static front-end pages in HTML and CSS was the initial step in our project. These technologies were chosen because of their simplicity and versatility to produce an intuitive end user experience. Pages with forms, navigation bars, and table/graph placeholders are front-end.

**Back-End Integration with Flask**

After static pages, we incorporated Flask in Python for the back-end. For its lightweight structure and web application development simplicity, Flask was chosen. User interface-application logic interaction and dynamic routing to static pages were added.

As noted in the difficulties section, integrating the pages proved difficult despite their simplicity.

**Data Collection**

The following crucial step was to create a data set for analysis and modelling. First, we used Kaggle, a popular site for diverse, high-quality datasets. Our dataset selection approach included choosing one that meets project goals and gives valuable insights. Our chosen data set was sufficient for exploration, but we needed more for comparative analysis to substantiate our results.

**Data Preprocessing**

Pretreatment ensured data set analysis and modelling. Some preprocessing:

* **Handling Missing Values:** Imputed missing entries or removed unneeded rows.
* **Outlier Detection:** Statistics handled outliers that could affect results.
* **Data Normalization:** Standardising and preparing the dataset for machine learning models required scale.

Solid data, low noise, and strong models need preprocessing.

**Exploratory Data Analysis (EDA)**

We understood the dataset via exploratory data analysis.

* Histogram and box plot analysis of data distribution were major tasks during this period.
* Finding correlations with scatterplots and heatmaps.
* Trending datasets.
* To proceed with feature selection and machine learning model creation, EDA provided background.

# Challenges Faced

**Dataset Selection**

Finding a good Kaggle dataset was hard. The platform has several datasets, however choosing one for our project was tricky. Some data sets lacked key properties, while others required substantial cleaning, making preprocessing more difficult.

**Web Application Integration**

Flask has trouble connecting front-end pages to back-end logic. Routing logic must be debugged and improved to ensure page navigation and communication.

# Next Steps

**Exploring AWS Datasets**

This step searches the AWS Open Data Registry for project-improvement data sets. Comparative analysis of these datasets will expand our machine learning models. The quality and relevance of AWS data sets are crucial to our project, making them crucial.

**Machine Learning Model Development**

We'll proceed to train and test machine learning models on EDA data. The expected steps are:

* **Model Selection:** Regression models, classification techniques, and neural networks are suitable.
* **Training and Testing:** To ensure thorough analysis, divide the data set into training and testing regions.
* **Model Evaluation:** Using accuracy, precision, recall, and F1 score to evaluate models.

**Model Deployment**

The next step is to incorporate trained models into the web application. Flask models make real-time predictions from data.

# AWS Login and Deployment Plan

**Logging into AWS**

The project will use AWS to host the web application, deploy models, and access datasets. The first step is AWS Management Console.

**Dataset Access**

We can discover complementary datasets in AWS Open Data Registry. These data sets will enhance analysis, making the end results more complete.

**Model Deployment**

Use AWS SageMaker and Lambda to implement machine learning models. Lambda supports serverless model inference, while SageMaker provides end-to-end machine learning workflow. To ensure easy integration, the given models will be linked to the Flask application.

**Web Application Hosting**

Web application will be hosted using AWS Elastic Beanstalk or EC2. These scalable and reliable services ensure a smooth user experience even with growing traffic.

**Monitoring and Maintenance**

Monitoring with AWS CloudWatch will ensure application performance. It monitors server uptime, response time, and issues.

# Conclusion

Designing and integrating the web application, data collecting, and exploratory analysis. While data set selection and page linking were difficult, we found answers and planned ahead. Integrating AWS datasets, machine learning models, and application delivery will be key. This ensures a user-friendly machine learning application.