- **Project Overview**: Data anomalies detection
    - The objective of this project is to detect anomalies in a dataset containing 150K daily trade data entries.
- **Scope**:
    - Focus on identifying and mitigating potential outliers, errors, or unusual patterns in trading data to improve data quality and decision-making.

2. Objectives

- **Primary Goal**:
    - Develop a machine learning model to accurately detect anomalies in the dataset.
- **Secondary Goals**:
    - Improve data preprocessing techniques.
    - Establish robust evaluation metrics to measure the performance of anomaly detection.
    - Implement the model in a scalable and maintainable way for future use.

3. Project Milestones

1. **Data Collection and Exploration**:
    - **Deliverables**: Dataset collection, initial data exploration, and summary statistics.
2. **Data Preprocessing**:
    - **Deliverables**: Data cleaning, normalization, and feature engineering.
3. **Model Selection and Training**:
    - **Deliverables**: Selection of appropriate anomaly detection algorithms, model training, and initial testing.
4. **Model Evaluation and Optimization**:
    - **Deliverables**: Model evaluation, parameter tuning, and performance optimization.
5. **Implementation and Deployment**:
    - **Deliverables**: Deployment of the model in a production environment.

6. **Final Review and Documentation**:
    - **Deliverables**: Final report, project documentation, and stakeholder review.

4. Data Exploration

- **Data Source**:
    - Trading dataset with 150K daily trade entries.
- **Exploratory Data Analysis (EDA)**:
    - Summary statistics, visualizations, correlation analysis, and anomaly detection techniques.
- **Key Insights**:
    - Identified patterns, outliers, and correlations that will inform the model development phase.

5. Data Preprocessing

- **Data Cleaning**:
    - Handle missing values, correct data entry errors, and filter out irrelevant data.
- **Normalization**:
    - Normalize data to ensure uniformity across features.
- **Feature Engineering**:
    - Create new features or transform existing ones to improve model performance if required
- **Tools**:
    - Python, Pandas, NumPy, Scikit-learn.

6. Model Selection

- **Algorithms**: (yet to decide)
    - Considering algorithms such as Isolation Forest, One-Class SVM, or Autoencoders.
- **Model Training**:
    - Split the data into training and testing sets, then train the selected models.

- **Evaluation Metrics**:
    - Use metrics to evaluate model performance.

7. Model Evaluation and Optimization

- **Performance Metrics**:
    - Evaluate the model on unseen data using cross-validation techniques.
- **Optimization Techniques**:
    - Hyperparameter tuning, model ensembling, and feature selection.
- **Tools**:
    - Python, Scikit-learn.

8. Implementation and Deployment

- **Environment Setup**:
    - Setup the production environment using cloud services like AWS, Docker, and Kubernetes.
- **Model Integration**:
    - Integrate the model into the existing trading platform.
- **Monitoring**:
    - Implement logging and monitoring to ensure the model performs well in production.

9. Risk Management

- **Potential Risks**:
    - Data quality issues, model overfitting, computational resource constraints.
- **Mitigation Strategies**:
    - Regular data quality checks, model validation, and resource allocation planning.