

AI-Powered Test Failure Prediction Dashboard

Project Summary Report

1. Introduction

The **AI-Powered Test Failure Prediction Dashboard** is a machine learning-driven application designed to assist Quality Assurance (QA) teams in predicting potential test failures using historical test execution data.

The system analyzes past automation results and identifies high-risk test cases, enabling smarter test prioritization during regression cycles.

This project demonstrates the practical use of **Artificial Intelligence in Software Testing** by combining data analytics, machine learning, and an interactive dashboard.

2. Problem Statement

In large regression test suites:

- Executing all test cases is time-consuming
- Critical failures are often detected late
- Test prioritization is mostly manual or experience-based

This project addresses these challenges by **predicting which test cases are most likely to fail**, allowing teams to focus on high-risk areas first.

3. Objectives

- Predict test case failure probability using historical data
 - Identify high-risk test cases before execution
 - Provide an interactive dashboard for QA insights
 - Reduce regression testing time and effort
 - Enable data-driven QA decision-making
-

4. System Overview

The application consists of:

- A **machine learning model** trained on historical test execution data
 - A **Streamlit-based dashboard** for visualization and interaction
 - CSV-based input and output for easy integration with QA workflows
-

5. Key Features

5.1 Test Failure Prediction

- Uses a **Random Forest Classifier** to predict the likelihood of test failure
- Generates probability scores for each test case

5.2 Interactive Dashboard

- Upload historical test execution data (CSV)
- View raw test data and prediction results
- Identify high-risk test cases
- Module-wise risk analysis
- Download prediction reports

5.3 Model Evaluation

- Accuracy
 - Precision
 - Recall
 - Confusion Matrix for performance assessment
-

6. Technology Stack

Layer	Technology
Programming Language	Python
Machine Learning	Scikit-learn (Random Forest)
Dashboard Framework	Streamlit
Data Processing	Pandas, NumPy
Visualization	Matplotlib, Seaborn

7. Workflow

1. Upload historical test execution CSV file
 2. Data preprocessing and feature preparation
 3. Model training using Random Forest algorithm
 4. Failure probability prediction for each test case
 5. Visualization of high-risk test cases
 6. Downloadable prediction report
-

8. Sample Input Data

Typical input fields include:

- Test Case ID
 - Module Name
 - Execution Count
 - Last Run Status
 - Failure History
 - Execution Duration
-

9. Benefits

- Faster regression cycles
 - Improved test coverage on critical areas
 - Reduced manual decision-making
 - Early detection of high-risk failures
 - Improved release confidence
-

10. Use Cases

- Regression test prioritization
 - CI/CD pipeline risk analysis
 - QA reporting and analytics
 - Automation test optimization
-

11. Future Enhancements

- Integration with CI/CD tools (Jenkins, GitHub Actions)
 - Real-time test execution data ingestion
 - Explainable AI (feature importance visualization)
 - PDF and executive-level reports
 - Cloud deployment and scalability
-

12. Conclusion

The **AI-Powered Test Failure Prediction Dashboard** showcases how machine learning can significantly improve software testing efficiency. By predicting test failures in advance, QA teams can optimize effort, reduce risks, and deliver higher-quality software faster.