TED UNIVERSITY

CMPE 492 / SENG 492 Senior Project

General AI Safety Systems Test Plan Report

Spring 2025

Team Members

Mustafa PINARCI 18853734706 Computer Engineering

Ege İZMİR 12584814676 Computer Engineering

Egemen Doruk SERDAR 71155167474 Software Engineering

Mustafa Boğaç MORKOYUN 44764509874 Software Engineering

Supervisor: Gökçe Nur YILMAZ

Jury Members

Eren ULU

Tansel DÖKEROĞLU

Tolga Kurtuluş ÇAPIN

Contents

1. Introduction

The purpose of this Test Plan is to describe the testing strategy and framework for the General AI Safety System. The aim is to ensure that each component—from facial recognition for attendance to dynamic route optimization, alerting, and data management—is rigorously tested to meet performance, security, and user experience expectations. This report outlines the test items, approach, resources, schedule, roles, responsibilities, risk factors, and references utilized in developing the plan.

2. Test Objectives

- **Functional Validation:** Verify that each module (Attendance Management, Route Optimization, Notification, and Data Management) operates as intended.
- **Integration Assurance:** Validate the interactions and data flow between subsystems.
- **Performance Benchmarking:** Ensure real-time processing (e.g., facial recognition and route re-calculation) meets performance criteria.
- **Security Verification:** Confirm that encryption, access control, and data storage adhere to the specified engineering standards.
- **User Acceptance:** Conduct testing with actual users (e.g., administrative staff) to confirm system usability and adherence to requirements.

3. Scope

This plan covers both functional and non-functional testing. It includes:

- **Unit Testing:** Testing individual components such as face detection algorithms and encryption functions.
- **Integration Testing:** Evaluating interactions between modules (e.g., FaceRecognition to AttendanceLogger).
- **System Testing:** Conducting end-to-end testing across the complete workflow.
- **Performance Testing:** Measuring response times and behavior under simulated load conditions.
- **User Acceptance Testing (UAT):** Involving end users to validate system usability and performance.
- **Beta Testing:** Limited deployment to collect real-world feedback and operational data.

4. Test Approach

4.1 Testing Methodologies

1. **Unit Testing:**
   - Objective: Validate each individual function or method.
   - Tools: Pytest, JUnit, or equivalent (based on project language).
   - Example: For function verifyStudent(image_path), assert that a known student image returns a positive match.
2. **Integration Testing:**
   - Objective: Ensure communication between components is reliable (e.g., data flows from FaceRecognition to AttendanceLogger are complete and accurate).
   - Techniques: Mock third-party API calls and simulate data streams.
3. **System Testing:**
   - Objective: Perform end-to-end tests across real-life scenarios (student boarding to alert notification).
   - Approach: Test on a staging environment configured similarly to production.
4. **Performance Testing:**
   - Objective: Verify that response times and throughput meet requirements (e.g., face verification under 2 seconds, optimal route calculation under 5 seconds).
   - Tools: Apache JMeter, Locust.
5. **User Acceptance Testing (UAT):**
   - Objective: Confirm usability and acceptance by representative users (administrators, drivers).
   - Method: Hands-on tests under controlled conditions with feedback collection.
6. **Beta Testing:**
   - Objective: Validate the system in a live, controlled environment with a limited user base.
   - Method: Monitor system logs, capture feedback, and track any discrepancies.

4.2 Test Environment

A dedicated environment is prepared to mimic production as closely as possible:

Component	Test Environment
Backend APIs	Docker-based containers on Linux
Face Recognition	System with NVIDIA GPU support
Database	PostgreSQL configured with AES-256 encryption
External API Clients	Simulated responses for Google Maps, OpenWeather
Messaging Services	Twilio API, SendGrid, Firebase for real-time SMS

5. Test Schedule

Phase	Estimated Dates	Activities
Unit Testing	April 15 – April 20, 2025	Execute module-level tests; use automated test scripts
Integration Testing	April 21 – April 25, 2025	Validate inter-module interactions
System Testing	April 26 – May 1, 2025	End-to-end scenerios on staging environment
Performance Testing	May 2 – May 3, 2025	Load and stress tests; benchmark against KPIs
User Acceptance Testing	May 4 – May 7, 2025	Real-world scenario tests with end users (parents,drivers, school administration)
Beta Testing	May 8 – May 12, 2025	Limited deployment; collect feedback and logs

6. Roles and Responsibilities

To maintain clear accountability, the following roles have been assigned:

Role	Assigned To	Responsibilities
Test Manager	Mustafa PINARCI	Overall test strategy, planning, and defect management
Unit Testing Lead	Egemen Doruk SERDAR	Oversee the development and execution of unit tests
Integration Testing Lead	Mustafa Boğaç MORKOYUN	Coordinate testing between different system modules
Test Engineer	Ege İZMİR	Write test cases, execute tests, and document outcomes
Supervisor	Gökçe Nur YILMAZ	Provide overall guidance and validate testing results

7. Risks and Mitigation Strategies

Identifying potential risks and defining plans to mitigate them is a critical component of the Test Plan:

Risk	Potential Impact	Mitigation Strategy
Failure in Real-Time Face Recognition	Delays in attendance logging; possible misidentification	Enhance image preprocessing; add fallback alerts; perform stress tests
API Downtime or Unreliable External Data	Route optimization failures; notification delays	Implement local caching and alternate path strategies
Performance Bottlenecks in High Load	Reduced system responsiveness affecting user experience	Stress testing and scaling strategies using cloud tools
Data Integrity and Encryption Issues	Compromise of sensitive data; non-compliance with standards	Regular integrity checks; scheduled backups; automated alerts
Inadequate User Feedback During UAT	Undetected usability issues leading to system rejection	Structured feedback collection; scheduled review sessions

8. Control Procedures

- **Defect Logging and Tracking:** Use Jira with issue severity levels and resolution SLAs.
- **Regression Testing:** Automated regression tests will run after each fix to confirm that new changes have not introduced new issues.
- **Version Control:** Both the test scripts and the test data will be version controlled using Git.
- **Change Management:** Formal change control procedures will be in place for any alterations in test cases or scope.
- **Periodic Reviews:** Weekly reviews to monitor progress, assess risks, and update stakeholders.

9. References

[1] ACM, "Code of Ethics and Professional Conduct," Association for Computing Machinery. [Online]. Available: https://www.acm.org/code-of-ethics

[2] European Union, "General Data Protection Regulation (GDPR)," [Online]. Available: https://gdpr.eu

[3] National Highway Traffic Safety Administration, "School Bus Safety," U.S. Department of Transportation. [Online]. Available: https://www.nhtsa.gov/school-bus-safety

[4] International Organization for Standardization, "ISO/IEC 27001 - Information security management," [Online]. Available: https://www.iso.org

[5] J. Redmon, "YOLOv3: Real-Time Object Detection," [Online]. Available: https://pjreddie.com/darknet/yolo/

[6] D. E. King, "Dlib Face Recognition Documentation," [Online]. Available: https://face-recognition.readthedocs.io

[7] OWASP Foundation, "OWASP Web Security Testing Guide," [Online]. Available: https://owasp.org/www-project-web-security-testing-guide/

[8] B. Bruegge and A. H. Dutoit, *Object-Oriented Software Engineering, Using UML, Patterns, and Java*, 2nd ed., Upper Saddle River, NJ: Prentice-Hall, 2004.

[9] Dlib, "HOG Face Detection Overview," [Online]. Available: http://dlib.net