Quality Assurance Plan

For

Smart Traffic Lights Simulation

Version 3.0 approved

Prepared by

Rabia Latif 20-SE-024

Abdullah Asif 20-SE-067

Samrin Fatima 20-SE-068

HITEC University Taxila Cantt

10 th May,2024

**Project Advisor**

Dr. Saima Shaheen

Co Supervisor(Sir Irfan Haider)

**Particulars of the students:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Registration#** | **Name in Full**  Use Block Letters | **CGPA** | **Signatures** |
|  | **20-se-024** | **RABIA LATIF** | **3.90** |  |
| **2.** | **20-se-068** | **SAMRIN FATIMA** | **3.24** |  |
| **3.** | **20-se-067** | **ABDULLAH ASIF** | **3.00** |  |

**Advisor’s Consent**

I Dr.Saima Shaheen am willing to guide these students in all phases of the above-mentioned project/thesis as advisor. I have carefully seen the Title and description of the project/thesis and believe that it is of an appropriate difficulty level for the number of students named above.

|  |  |  |
| --- | --- | --- |
| **Note:**  Advisor can’t be changed without prior permission of the FYP In-Charge and the duration for completion of Research Project / Thesis is 10 months (approx.) from the date of Registration of Research Project/Thesis. | ***Signatures and Date***   |  | | --- | |  |   **Advisor** |

**Co-Advisor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| I have carefully read the project proposal and feel that the proposed project is a useful one and of a sufficient difficulty level to justify a one year work load of above mentioned students. | | | | | |
| Recommended | | | | Signatures and Date | |
| Yes |  | No |  |  |  |
|  | | | |  |

Table of Contents

[***Signatures and Date*** 2](#_Toc166190712)

[1 Quality Assurance & Testing 4](#_Toc166190713)

[1.1 Quality Assurance Plan 4](#_Toc166190714)

[1.2 QA Schedule 5](#_Toc166190715)

[1.3 Quality Control 5](#_Toc166190716)

[1.4 Testing Requirements 5](#_Toc166190717)

[1.5 Scope 5](#_Toc166190718)

[1.6 Acceptance Criteria 6](#_Toc166190719)

[2. System Testing and Integration 7](#_Toc166190720)

[2.1 INTRODUCTION 7](#_Toc166190721)

[2.2 STRATEGIC APPROACH TO SOFTWARE TESTING 7](#_Toc166190722)

[2.3 Unit Testing 8](#_Toc166190723)

[3. TEST PLAN 9](#_Toc166190724)

[3.1 Black Box Testing: 10](#_Toc166190725)

[3.2 Unit Testing: 10](#_Toc166190726)

[3.3 Integration Testing: 10](#_Toc166190727)

[3.4 System Testing: 10](#_Toc166190728)

[4. TEST CASES: 10](#_Toc166190729)

[4.1 Testing Environment 11](#_Toc166190730)

[4.1.1 Testing Sites 11](#_Toc166190731)

[4.1.2 Required Hardware and Firmware Configuration 11](#_Toc166190732)

[4.1.3 Participating Organizations 11](#_Toc166190733)

[4.1.4 Manpower Requirements 11](#_Toc166190734)

[4.1.5 Preparation and Training Required for the Test Team 11](#_Toc166190735)

[4.2 TEST CASES 11](#_Toc166190736)

[*4.2.1* TEST CASE 1:AI DRIVEN SIGNAL TIMING 11](#_Toc166190737)

[*4.2.2* TEST CASE 2:PERFORMANCE MONITORING 12](#_Toc166190738)

[4.2.3 TEST CASE 3: PRIORITIZATION OF VECHILE 13](#_Toc166190739)

[*4.2.4* TEST CASE 4:ADAPTIVE TRAFFIC CONTROL 14](#_Toc166190740)

[*4.2.5* TEST CASE 5:EMERGENCY VEHICLE PRIORITIZATION 15](#_Toc166190741)

[4.2.6 TEST CASE 6:PEDESTRAIN SAFETY 16](#_Toc166190742)

# 1 Quality Assurance & Testing

## Quality Assurance Plan

The quality assurance (QA) plan for Smart Traffic Lights Simulation includes the following key components:

1. **Quality Goals:** Establish clear, measurable quality goals aligned with user expectations and business objectives.
2. **Quality Metrics**: Define metrics for assessing the quality of the system, including defect densities, test coverage, and user satisfaction ratings.
3. **QA Processes:** Develop QA processes that encompass all stages of the software development lifecycle, from requirements analysis to post-release support.
4. **Testing Tools and Environments:** Identify and set up testing tools and environments needed to simulate real-world usage scenarios.
5. **Risk Management:** Implement a risk management strategy to identify potential quality risks early in the development process and plan for mitigation.
6. **Documentation:** Ensure all testing activities are thoroughly documented, including test plans, test cases, and test results.
7. **Continuous Improvement:** Use feedback from testing phases to continuously improve the system, focusing on areas that directly impact user satisfaction and engagement.

## QA Schedule

A detailed QA schedule will be integrated with the overall project timeline, ensuring sufficient time is allocated for thorough testing and quality control.

## Quality Control

The final quality control will be conducted before release, ensuring that the system meets all acceptance criteria.

## Testing Requirements

Establish a comprehensive testing strategy to ensure that all features of the Smart Traffic Lights Simulation System meet the predefined functional and non-functional requirements.

## Scope

* **Functional Testing:**

Ensuring that each feature in the Smart Traffic Lights Simulation System works as per the specified requirements. For example, verifying that real-time traffic data acquisition, emergency vehicle detection, and signal timing optimization operate correctly.

* **Usability Testing:**

Checking that the system is easy to use and understand, focusing on user-friendliness. This includes assessing the interface for simplicity and making sure it is accessible for all users, including children who might interact with the system.

* **Performance Testing:**

Validating the system's responsiveness, stability, and scalability under various conditions. This involves testing how well the system performs during peak traffic hours, emergency situations, or when handling a significant volume of data.

* **Security Testing:**

Confirming that user data is securely handled and the application is safeguarded against potential vulnerabilities. This ensures that sensitive information is protected, and the system is resilient against potential cyber threats.

* **Compliance Testing:**

Verifying that the Smart Traffic Lights Simulation System adheres to all regulatory requirements, especially those related to children's online privacy and data protection. This ensures that the system complies with legal standards and safeguards user privacy, especially in contexts involving minors.

## Acceptance Criteria

**Criteria**

**1. Resolution of Critical Bugs:**

All critical bugs identified during testing phases must be resolved before release.

**2. Fulfillment of Functional Requirements:**

The system must fulfill all functional requirements as outlined in the SRS.

**3. Application Stability:**

The application should not crash during typical user interactions and must handle errors gracefully.

**4. Performance Benchmarks:**

Performance benchmarks, including load times and response times, must be met or exceeded.

**5. Positive User Experience:**

Feedback from usability testing sessions must reflect a positive user experience.

**Conditions**

**1. Acceptance Testing Environment:** Acceptance testing will be conducted in an environment that mirrors the production setup.

**2. Usability Testing Participants:**

Both regular commuters and emergency service personnel will be part of usability testing to gather comprehensive feedback.

**3. Security Testing Best Practices:**

Security testing results must show that the application adheres to best practices for data encryption and user authentication.

# System Testing and Integration

## INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

## STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

UNIT TESTING

MODULE TESTING

SUB-SYSTEM TESING

SYSTEM TESTING

ACCEPTANCE TESTING

Component Testing

Integration Testing

UserTesting

## Unit Testing

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

* 1. WHITE BOX TESTING

This type of testing ensures that

* All independent paths have been exercised at least once
* All logical decisions have been exercised on their true and false sides
* All loops are executed at their boundaries and within their operational bounds
* All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

* 1. CONDITIONAL TESTING

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

* 1. DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The definition-use chain method was used in this type of testing. These were particularly useful in nested statements.

* 1. LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

* All the loops were tested at their limits, just above them and just below them.
* All the loops were skipped at least once.
* For nested loops test the inner most loop first and then work outwards.
* For concatenated loops the values of dependent loops were set with the help of connected loop.
* Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

# TEST PLAN

A Software Test Plan is a document describing the testing scope and activities. It is the basis for formally testing any software/product in a project, Test Plans are comprehensive evaluation of programs, and procedures.

**Test Plans:** Plans for each level of testing.

* Black Box Testing
* Program Testing/ Unit Testing
* System Testing
* Parallel Testing

## Black Box Testing:

In this testing functional design of the system will be checked, without regard to the internal program structure.

## Unit Testing:

In the testing individual programs are checked to extend of desired output i.e. program specifications. While checking boundary conditions, exceptional Handling, error Handling etc.

## Integration Testing:

It is the process of combining and multiple components together. To assure that the software units or components operate properly when combined together

## System Testing:

These are used to test all program, which together constitute the system.the system testing is conducted using synthetic data both valid and invalid transaction are used in this test.

The step used in the system testing is:

1. Decompose and analyze the requirement specification.

2. Partition the requirements into logical categories and for each component make a list of the detailed requirements.

# TEST CASES:

**Functionalities:**

Here's an overview of the Smart Traffic Light Simulation project functionalities:

* Automate Tasks of traffic warden
* AI-Based Traffic Analysis
* Adaptive Traffic Light Control
* Performance Evaluation
* Emergency Vehicle Prioritization

Certainly! Here's a detailed test plan for an Smart Traffic Light Simulation:

**Scope of the Tests**

**The software package to be tested**

* **Name**: Smart Traffic Light Simulation
* **Version**: 2.0
* **Revision**: N/A

**Documents providing the basis for the planned tests**

* System Requirements Specification (Version 1.3)
* Design Document (Version 2.3)
* User Manual (Version 1.1)

## Testing Environment

### Testing Sites

Testing will be conducted in a dedicated testing environment.

### Required Hardware and Firmware Configuration

* Server: Minimum 16GB RAM, 500GB HDD, Intel Xeon Processor
* Client Machines: Minimum 8GB RAM, 250GB HDD, Intel Core i7 Processor
* Operating System: Windows 10

### Participating Organizations

* Testing Team: QA Department, HITEC University
* Development Team: Software Solutions

### Manpower Requirements

* **Test Lead**: 1 person(Rabia)
* **Testers**: 2 persons(Abdullah ,Samrin)
* **Developers**: 3 persons (for error correction)(Rabia,Samrin,Abdullah)

### Preparation and Training Required for the Test Team

* Test team will be provided with training on the system functionality and test procedures.
* Testers will familiarize themselves with the system by referring to the user manual.
* Developers will receive a briefing on identified issues and necessary fixes.

## TEST CASES

### TEST CASE 1:AI DRIVEN SIGNAL TIMING

| **Element** | **Description** |
| --- | --- |
| Test case id | TC\_01 |
| Test case objective | To verify the system's ability to adjust signal timing based on real-time traffic flow. |
| Pre requisite | Traffic monitoring sensors are operational |
| Steps | * 1. Simulate heavy traffic conditions at intersection.   2. Ensure that the AI-driven system receives and processes real-time traffic data.   3. Monitor the traffic light control system's response in adjusting signal timing to alleviate congestion. |
| Input data | Simulated heavy traffic flow data at the intersection. |
| Expected output | The traffic light system should optimize signal timing to reduce congestion, allowing smoother traffic flow within a reasonable time frame |
| Actual output | Same as expected output |
| Status | Pass |

### TEST CASE 2:PERFORMANCE MONITORING

| **Element** | **Description** |
| --- | --- |
| Test case id | TC\_01 |
| Test case objective | To verify if the smart traffic light system adapts to high traffic flow efficiently. |
| Pre requisite | Traffic monitoring sensors installed and operational. |
| Steps | 1. Increase the simulated traffic flow in a specific area controlled by the smart traffic lights. 2. Observe the system's response time in adjusting signal durations based on the increased traffic. |
| Input data | Simulated high traffic flow. |
| Expected output | The traffic lights adapt by extending green light durations on the busy route and reducing them on lesser-used routes. |
| Actual output | As expected |
| Status | Pass |

### TEST CASE 3: PRIORITIZATION OF VECHILE

| **Element** | **Description** |
| --- | --- |
| Test case id | TC\_02 |
| Test case objective | To validate emergency vehicle prioritization by the smart traffic control system. |
| Pre requisite | Emergency vehicle detection mechanism integrated and functional |
| Steps | 1. Simulate the approach of an emergency vehicle in the vicinity of the traffic lights. 2. Monitor the system's response in clearing traffic by giving priority to the emergency vehicle. |
| Input data | Simulated emergency vehicle signal trigger. |
| Expected output | Traffic lights optimize timings or adapt to the decreased traffic flow to minimize unnecessary waiting times. |
| Actual output | As expected |
| Status | Pass |

### TEST CASE 4:ADAPTIVE TRAFFIC CONTROL

| **Element** | **Description** |
| --- | --- |
| Test case id | TC\_01 |
| Test case objective | Verify the system's ability to detect and respond to high traffic density. |
| Pre requisite | Smart traffic light system installed and operational. |
| Steps | 1. Simulate high traffic density conditions at a specific junction or intersection. 2. Monitor the system's response time in detecting the increased traffic load. 3. Observe the changes in traffic light timings and patterns. |
| Input data | Increase the number of vehicles passing through the intersection beyond the normal threshold. |
| Expected output | Traffic lights adapt to the increased traffic by adjusting timings or implementing alternative traffic control measures. |
| Actual output | As expected |
| Status | Pass |

### TEST CASE 5:EMERGENCY VEHICLE PRIORITIZATION

| **Element** | **Description** |
| --- | --- |
| Test case id | TC\_01 |
| Test case objective | To verify the system's ability to detect an approaching emergency vehicle and prioritize traffic lights accordingly. |
| Pre requisite | The system is operational and configured with emergency vehicle detection sensors. |
| Steps | 1. Simulate an approaching emergency vehicle within the system's detection range. 2. Observe the traffic light response and change sequence. |
| Input data | Simulated emergency vehicle signal triggering the system. |
| Expected output | The traffic light in the emergency vehicle's direction switches to green while other lights turn red to clear the path. |
| Actual output | As expected |
| Status | Pass |

### TEST CASE 6:PEDESTRAIN SAFETY

|  |  |
| --- | --- |
| **Element**    Test case id | **Description**  TC\_02 |
| Test case objective | To test the functionality of pedestrian crossing integration. |
| Pre requisite | Pedestrian detection and crossing signal integration in the system. |
| Steps | 1. Initiate pedestrian crossing request at designated crossings controlled by the smart traffic lights. 2. Analyze the system's response in managing traffic flow for safe pedestrian crossing. |
| Input data | Simulated pedestrian crossing request. |
| Expected output | Halting traffic and providing a safe window for pedestrians to cross. |
| Actual output | As expected |
| Status | Pass |

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***