

Software Quality Assurance (SQA) Plan

By Team 5: Ctrl Alt Elite

Date: 18 September 2025

Signature Page

Prepared by: Akshay Krishna & Aishwarya Anand **Date:** ____15/09/2025____

Reviewed by 1: ____Akshay Krishna____ **Date:** ____22/09/2025____

Reviewed by 2: ____Shen Jia Cheng____ **Date:** ____22/09/2025____

Approved by : ____Aishwarya Anand____ **Date:** ____22/09/2025____

Contents

1. Purpose and Scope.....	5
1.1. Purpose.....	5
1.2. Scope.....	5
2. Reference Documents.....	5
3. Management.....	5
3.1. Management Organisation.....	6
3.1.1. Project Management.....	6
3.1.2. Assurance Management.....	6
3.2. Tasks.....	6
3.2.1. Product Assessments.....	7
3.2.2. Process Assessments.....	7
3.3. Roles and Responsibilities.....	7
3.3.1. QAM.....	7
3.3.2. Software Quality Personnel.....	7
4. Documents.....	8
4.1. Purpose.....	8
4.2. Minimum Document Requirements.....	8
5. Standards, Practices, Conventions and Metrics.....	8
5.1. Purpose.....	8
5.2. Software Quality Programme.....	8
5.2.1. Standard Metrics.....	8
6. Software Reviews.....	9
6.1. Purpose.....	9
6.2. Minimum Software Reviews.....	9
7. Test.....	9
8. Problem Reporting and Corrective Action.....	10
9. Tools, Techniques and Methodologies.....	10
9.1. Software Quality Tools.....	10
10. Media Control.....	10
11. Supplier Control.....	10

12.	Record Collection, Maintenance, and Retention.....	10
13.	Training.....	11
14.	Risk Management.....	11
15.	SQA Plan Change Procedure and History.....	11

1. Purpose and Scope

1.1. Purpose

The purpose of this Software Quality Assurance (SQA) Plan is to establish the goals, processes, and responsibilities required to implement effective quality assurance functions for the NexusNTU project.

The Software Quality Assurance Plan provides the framework necessary to ensure a consistent approach to software quality assurance throughout the project life cycle. It defines the approach that will be used by the QAM and Software Quality (SQ) personnel to monitor and assess software development processes and products to provide objective insight into the maturity and quality of the software. The systematic monitoring of products, processes, and services will be evaluated to ensure they meet requirements and comply with policies, standards, and procedures, as well as applicable Institute of Electrical and Electronics Engineers (IEEE) and ISO standards.

1.2. Scope

The purpose of SQA is to ensure that the software developed does not deviate from the original intended product. SQA is also concerned to identify any errors, omissions, inconsistencies, and alternatives, enhancements or improvements that can be made at any stage of development.

NexusNTU is a web application that centralises essential resources for NTU's international and exchange students, bringing together a personalised information hub, campus navigation and nearby amenities, a 24/7 AI help chatbot, verified quick links, school/course hubs, and a small utility suite (login/OTP, profile, currency converter). The supplier of the software is SC3040 Group 5 "CTRL ALT ELITE" (NTU, CCDS). The plan covers the React PWA frontend, a Node.js/Express API, a MongoDB database, and the listed modules, plus platform services such as schedulers/caching, a LinkRegistry with automated link checks, simple data adapters to official sources (e.g., NTU Omnibus, Fullerton Health, One-Stop), feedback collection, and security (HTTPS, 2FA/OTP, secure tokens). The intended use is to help students quickly access official guidance, navigate campus, and find services anytime, while enabling administrators to keep links verified and manage school-specific hubs with built-in tools.

2. Reference Documents

- IEEE STD 730-2002, IEEE Standard for Software Quality Assurance Plans (http://standards.ieee.org/reading/ieee/std_public/description/se/730-2002_desc.html)
- ISO/IEC/IEEE 12207:2017 — Systems & Software Life Cycle Processes: [ISO page](#) / [IEEE page](#) [ISO](#)
- ISO/IEC 25010:2011 — Systems & Software Quality Models: [ISO page](#) [ISO](#)
- ISO/IEC 27001:2022 — Information Security Management Systems: [ISO page](#) [ISO](#)
- W3C WCAG 2.2 — Web Content Accessibility Guidelines: [W3C Recommendation](#) (overview: [What's new in 2.2](#)) [W3C](#)
- OWASP ASVS 4.x — Application Security Verification Standard: [Project page](#) / [v4.0.3 PDF](#) [OWASP Foundation](#)
- OpenAPI Specification 3.1 — REST API Description Standard: [OAS 3.1.1](#) [OpenAPI Initiative Publications](#)
- Google Material Design Guidelines — UI/UX Patterns: [Material Design \(M3\)](#) / [Material.io overview](#) [Material Design](#)
- ESLint/Prettier Coding Guidelines — JavaScript/TypeScript Style: [ESLint rules](#) / [Prettier docs](#) [eslint.org](#)
- GitHub Flow Guidelines — Branching, PRs, and Reviews: [GitHub Docs: GitHub flow](#) [GitHub Docs](#)
- ISO IEC 90003:2004 Software Standard (<http://praxiom.com/iso-90003.htm>)
- Project Plan
- System Requirement Specifications

3. Management

This section describes the management organizational structure, its roles and responsibilities, and the software quality tasks to be performed.

3.1. Management Organisation

The implementation of quality assurance system is the responsibility of the Quality Assurance Manager (QAM).

3.1.1. Project Management

The Project Manager will be responsible for approving:-

- The system requirement specification document
- The overall time scale for the project
- The choice of system development life cycle
- The choice of software development tools and techniques utilised
- The selection of project teams
- The training of project teams

3.1.2. Assurance Management

The QAM provides Project Management with visibility into the processes being used by the software development teams and the quality of the products being built. The QAM maintains a level of independence from the project and the software developers.

In support of software quality assurance activities, the QAM has assigned and secured Software Quality personnel from the pool of available SQ trainees to coordinate and conduct the SQ activities for the project and report back results and issues.

3.2. Tasks

This section summarizes the tasks (product and process assessments) to be performed during the development of software. These tasks are selected based on the developer's Project Plan and planned deliverables, and identified reviews.

3.2.1. Product Assessments

The following product assessments will be conducted by SQ personnel:

- Access & Profile (Auth/OTP) — registration, login, logout, password/phone change, 2FA/OTP flow; token handling and session security.
- Centralised Information Hub — content aggregation accuracy, freshness (≤ 24 h for official notices), de-duplication, and deep-link correctness.
- Quick Links Dashboard & LinkRegistry — coverage of essential links, verification TTLs, nightly automated link-checks, and broken-link rate.
- Navigation — campus search, route correctness, map rendering, error states.
- Nearest-Amenities Finder — POI accuracy, search/routing results and performance.
- AI Chatbot (24/7) — KB coverage, answer accuracy on in-scope queries, graceful fallbacks.
- Currency Converter — rate accuracy vs. source and latency.
- Settings & School/Course Hubs — template fidelity, role/visibility rules, admin overrides.
- API & Data Model — endpoint contracts, schema versioning, audits, and retention.

- PWA & Offline Readiness — installability, offline hotlines, cached fallbacks with “last updated”.
- Performance — boot/login/route/amenities response targets per plan; staging smoke results.
- Accessibility & UX — WCAG-aligned labels, keyboard nav, responsive layouts; usability fixes verified.
- Security & Privacy — TLS, JWT/Firebase Auth, 2FA/OTP, rate-limits; data minimisation (no passport/FIN/STP).
- Logging/Monitoring — client/server telemetry present and actionable for incidents.

3.2.2. Process Assessments

The following process assessments will be conducted by SQ personnel:

- Requirements Management — traceability from UCs/SRS to builds and tests.
- Architecture & Design Control — reviews of system/DB/API specs before implementation.
- Configuration & Change Management — version tagging, change logs, release notes.
- CI/CD & Build Verification — pipeline runs, artefact integrity, daily staging smoke (E2E).
- Code Quality & Reviews — adherence to coding standards, PR checklists, and static analysis.
- Test Management — unit/integration/E2E coverage, test data, and results sign-off before demos.
- Usability & Accessibility Validation — planned studies, issue triage, fix validation.
- Security Review — threat/abuse cases, auth flows, secrets handling, dependency scans.
- Data Governance — PII minimisation checks, retention policies, audit trails.
- Release & Deployment — cutover/rollback procedures, environment parity checks.

- Operations & Incident Response — monitoring thresholds, alerting, backup/restore drills.
- Stakeholder Reviews — sprint reviews against MVP contract and acceptance criteria.

3.3. Roles and Responsibilities

This section describes the roles and responsibilities for each assurance person assigned to the Project.

3.3.1. QAM

Responsibilities include, but are not limited to:

- Secure and manage SQ personnel resource levels
- Ensure that SQ personnel have office space and the appropriate tools to conduct SQ activities
- Provide general guidance and direction to the SQ personnel responsible for conducting software quality activities and assessments
- Assist SQ personnel in the resolution of any issues/concerns and/or risks identified as a result of software quality activities
- Escalate any issues/concerns/risks to project management

3.3.2. Software Quality Personnel

Responsibilities include, but are not limited to:

- Develop and maintain the project software quality assurance plan
- Generate and maintain a schedule of software quality assurance activities
- Conduct process and product assessments, as described within this plan
- Identify/report findings, observations, and risks from all software assurance related activities to the QAM

4. Documents

4.1. Purpose

This section identifies the minimum documentation governing the requirements, development, verification, validation, and maintenance of software that falls within the scope of this software quality plan. Each document below shall be assessed (reviewed) by SQ personnel.

4.2. Minimum Document Requirements

- System Requirement Specifications
- Use Case Diagrams and Descriptions
- Design report on software maintainability

- Software Requirement Specification
- Quality Management
- Software Model Prototype
- Configuration Management Plan
- Change Management Plan
- Release Plan
- Test Plan and Documentation

5. Standards, Practices, Conventions and Metrics

5.1. Purpose

This section highlights the standards, practices, quality requirements, and metrics to be applied to ensure a successful software quality program.

5.2. Software Quality Programme

These practices and conventions are tools used to ensure a consistent approach to software quality for all programs/projects across NexusNTU. For this project, the four most important qualities we will actively track are:

- **Functionality** – Core modules must do exactly what the use cases specify (e.g., Navigation shows a valid route with mode/distance/duration; Amenities Finder lists nearby services) and meet target specs (e.g., AI chatbot $\geq 85\%$ correct on in-scope queries; Information Hub surfaces NTU announcements within ≤ 24 h).
- **Reliability** – Service stays available and graceful under load: uptime $\geq 98\%$ with cached fallbacks and “last updated” stamps; scheduled refresh jobs keep data fresh; automated link-verification guards Quick Links. QA includes unit/integration and end-to-end smoke on staging.
- **Usability & Accessibility** – Interfaces are simple, fast, and inclusive: WCAG AA, keyboard accessibility, multilingual labels, and performance budgets on key screens; PWA installable with useful offline content.
- **Security & Privacy** – Protect users and minimise data: TLS, hashed passwords, 2FA/OTP, JWT rotation, and rate-limited auth; integrate via deep links and do not store passport/FIN/STP. REST over HTTPS with Firebase/JWT enforces request-level security.
- **Portability** – To be able to reach out to more users across different platforms such as Android and IOS devices, and smart watches in the future.

5.2.1. Standard Metrics

The following standard metrics are the minimum planned metrics that will be collected, reported, and maintained in the area of software quality assurance:

- Fan-in/fan-out
- Length of code
- Length of identifiers
- Cyclomatic complexity
- Depth of conditional nesting

6. Software Reviews

6.1. Purpose

This section identifies the number and type of system/subsystem reviews and engineering peer reviews that will be supported by the SQ Personnel. The project milestone chart, and the SQ Personnel resource levels determine the reviews that are supported.

6.2. Minimum Software Reviews

For each review, SQ will assess the review products to assure that review packages are being developed according to the specified criteria, the review content is complete, accurate, and of sufficient detail, and Requests for Action are captured, reviewed, and tracked to closure. In addition, SQ will assess the processes used to conduct the reviews to determine if appropriate personnel are in attendance, correct information is presented, entry and exit criteria are met, and appropriate documents are identified for update.

The following software reviews will be assessed by SQ:

- Project Plan Review
- Requirements Analysis Review
- Software Design Review
- Test Plan Review
- Acceptance Review
- Peer Review
- Post-Implementation Review

7. Test

SQ personnel will assure that the test management processes and products are being implemented per Test Plan. This includes all types of testing of software system components as described in the test plan, specifically during integration testing (verification) and acceptance testing (validation).

The testing activities covered by SQ personnel include:

1. Unit Testing – Verification of individual software components or modules to ensure they perform as designed. SQ personnel will review unit test cases, execution results, and defect reports.
2. Integration Testing – Verification of combined modules or components to ensure they interact correctly. SQ personnel will monitor integration test schedules, review integration test results, and ensure that interface defects are tracked and resolved.
3. System Testing – Validation of the complete, integrated system to confirm that it meets the specified requirements. SQ personnel will assess system-level test plans, test execution, and results for correctness, completeness, and compliance.
4. Validation Testing – Assurance that the developed system satisfies the intended use and user needs. SQ personnel will verify that validation tests are properly planned, executed, and documented.
5. User Acceptance Testing (UAT) – Confirmation by end-users that the system meets business requirements and is ready for deployment. SQ personnel will observe UAT sessions, ensure proper logging of user feedback, and verify that defects or change requests are addressed.

SQ personnel will monitor testing efforts to assure that test schedules are adhered to and maintained to reflect an accurate progression of the testing activities. SQ will assure that tests are conducted using approved test procedures and appropriate test tools, and that test anomalies are identified, documented, addressed, and tracked to closure. In addition, SQ will assure that assumptions, constraints, and test results are accurately recorded to substantiate the requirements verification/validation status. SQ personnel will review post-test execution related artifacts including test reports, test results, problem reports, updated requirements verification matrices, etc.

8. Problem Reporting and Corrective Action

SQ personnel generate, track, and trend assessment findings and observations in a centralized Reporting and Corrective Action System.

For this project, the system will be maintained as a secure Excel-based tracker stored on the project server with restricted access to SQ personnel, the QAM, and the Project Manager. All entries will be timestamped, categorized by severity, and assigned to the responsible team member for resolution.

The process for reporting and corrective action includes:

1. Identification – SQ personnel identify and records all defects, non-conformances, or deviations from project requirements discovered during assessments, testing, or reviews.
2. Tracking – Each issue is logged in the centralized system with a unique identifier, description, severity, status, assigned team member, and target date to resolve this issue.
3. Analysis – SQ personnel look for patterns in recurring problems to suggest improvements
4. Corrective Action – The responsible team members are informed and take steps to fix the issues. SQ personnel follow up to make sure problems are resolved.
5. Communication – Updates on problems and fixes are shared with the QAM and Project Manager every week. Serious issues are reported immediately*.
6. Closure Verification – Once a problem is fixed, SQ personnel check and ensure it's resolved before marking it complete.

This process provides traceability, ensures accountability, and allows for continuous improvement of both the software product and the development process.

9. Tools, Techniques and Methodologies

SQ personnel will require access to the following:

9.1. Software Quality Tools

- Microsoft Office tools (i.e., Word, Excel, and PowerPoint)
- GitHub
- Firebase
- Mapbox API
- Figma
- Visual Paradigm
- Draw.io

10. Media Control

SQ deliverables will be documented in one of the following Microsoft software applications: Word, Excel, or PowerPoint. Deliverables will be in soft copy, with the exception of completed checklists from process and product assessments. See Section 12 for additional details on the collection and retention of key records. Software Quality personnel will request space on the project's secured server for SQ records. This server is password protected and backed up nightly.

This project records and documentation will be maintained using the following services:

1. MediaWiki – A free, open-source platform for creating and maintaining project documentation. It supports collaborative editing and allows multiple team members to contribute simultaneously.
2. Google Drive – Used for storing and sharing soft copies of reports, checklists, and other project deliverables. It allows the team to access files from anywhere and supports real-time collaboration.
3. GitHub – Used for version control and collaborative development of source code and project documents. GitHub enables tracking changes, managing pull requests, and maintaining an organized history of revisions.

This approach ensures that all SQA deliverables are secure, organized, and easily accessible, while promoting collaboration and accountability among team members.

11. Supplier Control

[Not applicable for this project]

12. Record Collection, Maintenance, and Retention

SQ personnel will maintain records that document assessments performed on the project. Maintaining these records will provide objective evidence and traceability of assessments performed throughout the project's life cycle. There are two types of records that will be maintained: Hardcopy and Electronic. SQ personnel will maintain electronic or hard copies of all assessment reports and findings. SQ Project folders will contain hardcopies of the assessment work products such as completed checklists, supporting objective evidence, and notes.

The table below identifies the record types that will be collected, as well as the Record Custodian and Retention period

Record Title	Record Custodian	Record Retention
SQA Assessments	SQ Personnel	One Year
SQA Checklists	SQ Personnel	One Year
Deliverable Defects	SQ Personnel	One Year

13. Training

SQ personnel have fundamental knowledge in the following areas through prior experience, training, or certification in methodologies, processes, and standards:

- Audits and Reviews (Assessments)
- Risk Management
- Software Assurance
- Configuration Management
- Software Engineering
- ISO 9001, ISO 9000-3
- CMMI
- Verification and Validation

14. Risk Management

SQ personnel will assess the project's risk management process and participate in monthly risk management meetings and report any software risks to the QAM and the project manager.

The QA team has identified potential risks for the GoldFolks Project, along with their likelihood, severity, potential impact, and proposed mitigation strategies

Risk Type	Risk	Likelihood	Severity	Potential Impact	Mitigation Strategy
Resource	Missing the project deadline	Moderate	High	Could result in reduced functionality or delay the project deployment.	Conduct regular progress check-ins and meetings with the development team to monitor progress.
Hardware	Hardware failure, such as a corrupted hard drive	Low	High	Hardware failures, such as a corrupted hard drive, may prevent code from being written or executed, causing delays.	Maintain regular backups using version control tools like Git to minimize disruption.
Software	Scope creep or uncontrolled addition of new features	Moderate	High	Scope creep may lead to delays and difficulties in meeting project deadlines.	Ensure clear traceability between documentation and code to adhere strictly to the defined requirements.
Communication	Poor communication or lack of updates among team members	High	Moderate	Lack of communication could result in overlapping work, coding conflicts, or inaccuracies in	Promote transparency among team members when sharing updates to reduce miscommunication

				documentation.	on and conflicts.
--	--	--	--	----------------	-------------------

15. SQA Plan Change Procedure and History

SQ personnel are responsible for the maintenance of this plan. It is expected that this plan will be updated throughout the life cycle to reflect any changes in support levels and SQ activities. Proposed changes shall be submitted to the Quality Assurance Manager (QAM), along with supportive material justifying the proposed change.