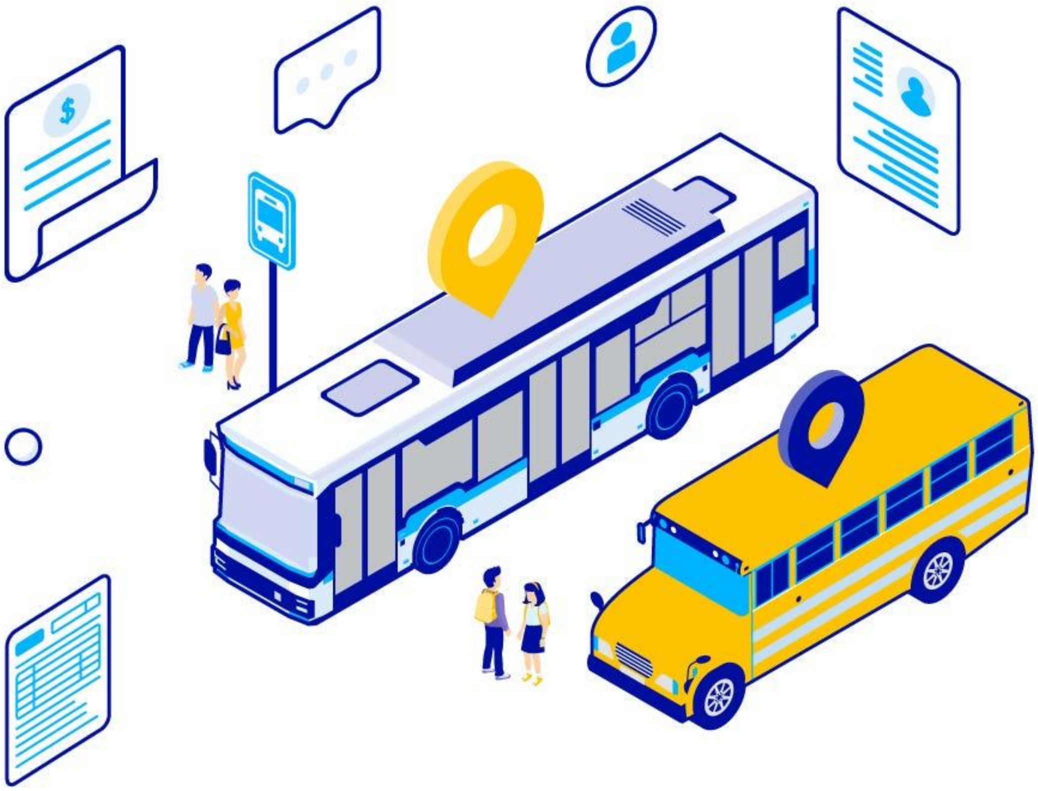


Public Transport Tracker Project Plan

Ndabenhle Dlamini – 202203349
Sisekelo Xaba – 202202154
Nkosingizwile Simelane - 202201752

Table of Contents

1. Introduction

Background and History of the Project

Towns in Eswatini are growing quickly; more people need reliable and efficient public transportation. Buses and mini-buses(kombis) are essential for daily travel, but they face problems like irregular schedules, no live updates, and are always chasing against time. These issues such as unpredictable arrival times, lack of real-time information, and limited fleet management tools cause stress for passengers and make it hard for passengers to commute smoothly through out their daily routes.

This project aims to solve such issues by developing a Real-Time Public Transport Tracking System. Using GPS technology, cloud systems, and mobile apps. Moreover, the system will show passengers the exact location of buses and their expected arrival times. Also transport operators will also get a dashboard to track and manage their vehicles better.

The birth of this idea comes from other countries where real-time tracking has made public transport more reliable and user-friendly. This project will follow a proven software engineering method (described in Sommerville's Software Engineering, 2016), the goal is to create a simple, strong, and adaptable system that fits Eswatini's transport needs.

Objectives

The specific objectives are as follows:

Create Easy-to-Use Apps - Build a mobile or website app with a simple design so passengers can check bus locations and times quickly.

Show Estimated Arrival Times - Use past traffic patterns and smart calculations to predict and display when buses will reach passenger's stop.

Operator Analytics Dashboard - Provide transport operators with tools to monitor fleet performance, vehicle performance, find better routes, and improve services. Improve service delivery.

Track Vehicles in Real Time – collaborate with GPS companies in the country to install devices in buses/kombis to send location updates every 30 seconds, giving passengers accurate, live info.

Aims

The primary aim of the project is to enhance the efficiency, reliability, and user experience of public transport services by implementing a real-time tracking application system. We aim to deliver a strong, scalable, and user-friendly solution personalized to the unique needs of Eswatini's public transport ecosystem. Furthermore, we aim at designing a system to support up to 1,000 vehicles while operating smoothly and ensuring secure data transmission. The project also aims to provide performance updates every 30 seconds. In conclusion we will look at ensuring a usable interface for low-tech users.

2. User/Client Involvement in the Public Transport Tracker Project for Eswatini

Effective execution of the Public Transport Tracker relies heavily on active collaboration with users and clients, whose contributions of information, services, resources, and facilities are phased to align with the project's timeline. The Eswatini Transport Regulatory Commission (ETRC) serves as a foundational partner, providing critical regulatory data such as route permits, vehicle registries, and historical delay patterns by Month 1. This information is vital during the project's initial planning phase, enabling accurate mapping of informal kombi routes and informing the design of real-time tracking algorithms. For instance, ETRC's data on flood-prone areas in Siteki directly supports the development of weather-responsive delay alerts, ensuring the system adapts to local environmental challenges.

Kombi associations, representing informal transport operators, contribute operational insights and permissions for GPS device installations by Month 3. Their cooperation is essential during the pilot phase in Mbabane, where 50 kombis are equipped with solar-powered tracking units. These associations also facilitate driver training workshops, ensuring smooth adoption of new technologies. By granting access to depots and fare structures, they enable the integration of fare transparency features into the app, aligning with the Eswatini Revenue Authority's formalization goals.

Telecom providers like MTN Eswatini and ECCELL deliver technical services critical to the project's hybrid tracking system. By Month 6, they supply discounted SMS/USSD bundles and cellular network support, ensuring rural users in regions like Nhlangano can access ETAs via low-tech channels. This partnership is pivotal during the testing phase, where USSD menus are validated for usability in areas with limited internet coverage. Telecom infrastructure also underpins the SMS-based delay alerts deployed during rainy seasons, enhancing commuter safety.

Academic institutions, notably the University of Eswatini (UNESWA), provide beta testers and research collaboration from Month 5. Students and faculty participate in usability testing, offering feedback on the app's multilingual interface (English/SiSwati) and offline functionality. Their input refines features like route visualization using OpenStreetMap, ensuring the tool meets the needs of both urban and rural commuters. UNESWA's involvement bridges technical development with community-centric design, fostering higher user adoption.

Local governments and community leaders contribute facilities and cultural legitimacy. Chiefs in rural areas, for example, grant access to community centers for driver training sessions by Month 4, while municipal authorities in Manzini allocate server hosting space via ESCCOM by Month 3. These resources ensure the project remains rooted in Eswatini's socio-cultural context, building trust among stakeholders.

Finally, commuters and drivers themselves are active participants, providing real-time feedback through USSD surveys and beta app testing from Month 6 onward. Their insights drive iterative improvements, such as optimizing notification timings for peak-hour congestion in Manzini. This direct engagement ensures the system remains responsive to evolving needs, solidifying its role as a public good.

By structuring user/client involvement around phased deliverables, the project mitigates delays and ensures resources align with each development stage—from initial data gathering to nationwide rollout. This collaborative framework not only adheres to Eswatini's National Digital Strategy but also embeds local ownership, a key factor in the system's long-term sustainability.

3. Potential Risks & Mitigation Strategies

Implementing a public transport tracking system in Eswatini involves navigating challenges such as limited rural connectivity, resistance from informal transport operators, budget limitations, and data privacy concerns. To ensure project resilience, hybrid tracking (GPS + SMS/USSD) will bridge connectivity gaps in remote areas, supported by partnerships with local telecom providers like MTN Eswatini. Resistance from *kombi* operators will be addressed through collaborative incentives, including training programs and fare subsidies, fostering stakeholder buy-in. Budget constraints will be mitigated by pursuing grants from regional initiatives (e.g., AU Smart Cities Program), while data privacy will be safeguarded through anonymization and compliance with Eswatini's Data Protection Act (2021). These strategies collectively align with local infrastructure capabilities and socio-economic priorities.

Risk	Mitigation
Low GPS/internet coverage	Hybrid tracking (SMS/USSD for rural areas). Partner with MTN Eswatini for coverage.
Resistance from kombi operators	Incentivize participation (e.g., priority parking, fare subsidies).
Budget constraints	Seek funding from AU Smart Cities Program or World Bank grants.
Data privacy concerns	Comply with Eswatini Data Protection Act (2021); anonymize user data.

4. Standards, guidelines and procedures

The success of the Public Transport Tracker in Eswatini project relies on all individuals involved to adhere to a set of standards, guidelines, and procedures that have been carefully defined and agreed upon They help keep work consistent, high-quality, and organized. Below is a detailed breakdown of these elements:

Standards

Documents and Code

Document Format: Use clear, organized templates for all reports, plans, and designs.

Coding Style: Write code neatly with consistent names, formatting, and comments.

User-Friendly Design: Make sure apps and tools are easy for everyone to use, including people with disabilities.

Guidelines

-Write requirements in simple language

-Talk to stakeholders often to confirm needs.

-Design code in modules for easy updates.

-Focus tests on the most critical parts first.

-Keep detailed records of test results and fixes.

-Share ideas openly and use Github to track tasks.

Procedures

Clear procedures for key tasks

How to Gather Requirements:

-Talk to stakeholders (e.g., interviews, workshops).

-Write down their needs in a standard template.

-Refine the list based on feedback.

How to Review Code:

-Assign a teammate to check new code.

-Ensure it follows coding rules.

-Fix issues before adding it to the main system.

Managing Risks:

-List possible risks (e.g., delays, tech issues).

-Rate how likely and severe they are.

-Create backup plans and monitor risks.

Handling Changes:

-Log all change requests in a central system.

-Check how changes affect timelines or costs.

-Get approval before making changes.

Testing the System:

-Set up a testing environment like the real system.

-Test parts individually, then together, then the whole system.

-Re-test after fixing issues.

This guidelines and procedures matter because the consistency allowing everyone works the same way. They also provide quality resulting to fewer errors and better results. furthermore, they enhance efficiency resulting to less rework and faster progress. In conclusion they provide compliance ensuring organizational and legal rules are meet

5. Project Organization

Relation to Other Projects

The Public Transport Tracker integrates with and supports several national and regional initiatives in Eswatini, enhancing collaborations and maximizing impact.
These include:

- *Eswatini's National Digital Strategy (2023–2028)*- This is a government-led initiative to digitize public services and infrastructure. The tracker directly contributes to the strategy's goal of "digital inclusion" by modernizing transport data systems and providing tech-driven solutions for rural and urban commuters.
- *AU Agenda 2063 (Sustainable Urban Mobility)*- this is a continental framework promoting sustainable transport systems in Africa. It aligns with Agenda 2063's focus on reducing traffic congestion and carbon emissions by optimizing public transport efficiency through real-time tracking.
- *Eswatini Smart Cities Framework (Mbabane & Manzini)*- Urban development plan to integrate IoT and data-driven infrastructure in cities. The tracker serves as a foundational component, feeding real-time transport data into broader smart city systems e.g., traffic light coordination, emergency response.
- Eswatini Revenue Authority's Transport Sector Formalization Project aims to formalize informal transport operators e.g., kombis for better regulation and taxation. The tracker provides data on routes and operator activity, aiding the Revenue Authority in monitoring compliance and improving fare transparency.

Project Organization & Roles of Personnel

The project is structured to align with Eswatini's unique operational context, balancing technical consistency with community engagement. Below is the organizational breakdown:

Team Structure: Core Roles & Responsibilities

1. ***Project Manager (PM)* -** Oversees timelines, budgets, and stakeholder coordination

   Key Tasks:

   -Liaise with government bodies (Ministry of Transport) and funders.

   -Mitigate risks like operator resistance or budget shortfalls.

-Ensure compliance with Eswatini Data Protection Act (2021).

2. ***Technical Lead* -** Manages system architecture, GPS/SMS integration, and backend development.

   Key Tasks:

   -Design hybrid tracking (GPS + USSD) for low-internet areas.
   -Partner with MTN Eswatini/ESCCOM for cellular network reliability.
   -Supervise developers (frontend/backend).

3. ***Data Analyst* -** Develops predictive algorithms for ETAs using local traffic patterns

   Key Tasks:

   -Analyze historical delay data from bus depots.
   -Optimize routes for rural corridors (e.g., Manzini-Nhlangano).

4. ***UI/UX Designer* -** Designs a user-friendly, multilingual app (English/SiSwati) for rural and urban users.

   Key Tasks:

   -Prioritize offline functionality and low-data usage.
   -Conduct usability tests with beta groups (e.g., UNESWA students).

5. ***QA Lead (Tester)* -** Ensures system reliability, security, and performance.

   Key Tasks:

   -Test GPS accuracy in mountainous regions (e.g., Mbabane).
   -Validate SMS/USSD fallback systems during network outages.

6. ***Community Liaison Officer* -** Bridges communication between the project and local stakeholders.

   Key Tasks:

   -Negotiate with kombi associations for GPS device adoption.
   -Organize driver training workshops in collaboration with local chiefs.

7. ***Field Technicians* -** Install and maintain GPS devices on kombis/buses.

   Key Tasks**:**

   -Troubleshoot hardware issues in remote areas.
   -Train drivers on device usage (e.g., solar-powered units).

Training Requirements

- **Technical Team**:

description	Duration (Months)
USSD/SMS integration workshops	2
GPS hardware maintenance training	3

- **Field Team**:

Description	Duration (Months)
Cultural sensitivity training for rural engagement	1

- **Kombi Drivers**:

Description	Duration (Months)
GPS device operation sessions	4 - 5

6. Project phases.

This project will be done in clear steps (using the Iterative Waterfall Model) to ensure it stays on track, meets deadlines, and uses resources well.

Start Phase

Tasks

-Hold meetings with stakeholders (like transport operators) to list needs.

-Define goals also what the project will deliver and possible risks.

-Create a rough timeline and budget.

Key Points

-Milestone 1: Project plan approved by stakeholders.

-Milestone 2: Final list of requirements ready.

Important Path

-Stakeholder meetings → Collecting requirements → Approval of plan.

Time & Cost

-2 weeks, E10 000

Design Phase

Tasks

-Plan the system structure (layers: GPS, app, database).

-Design parts like GPS setup, app features, and maps.

-Create diagrams to explain how everything works.

Key Points

-Milestone 3: System design approved.

-Milestone 4: Detailed designs for each part ready.

Important Path

-System structure → Part designs → Approval of design.

Time & Cost

-3 weeks, E15000 (for design tools and team).

Build Phase

Tasks

-Set up GPS devices and connect them to the system.

-Build the backend (server) and app (mobile/web).

-Set up the database for live location tracking.

Key Points

-Milestone 5: GPS working with the system.

-Milestone 6: Backend ready for testing.

-Milestone 7: App prototype ready for feedback.

Important Path

-GPS setup → Backend → App → Prototype done.

Time & Cost:

-12 weeks, E20000 (for devices, software, and team).

Testing Phase

Tasks

-Test each part alone (unit testing).

-Test parts working together (integration testing).

-Test the whole system (system testing).

-Test with real users (pilot testing).

Key Points

-Milestone 8: Unit tests done.

-Milestone 9: System tests passed.

-Milestone 10: Users approve the system.

Important Path

-Unit tests→ Integration tests → System tests → User approval.

Time & Cost

-4 weeks, E16,000 (for testing tools and team).

Launch Phase

Tasks

-Launch the system in one city (e.g., Mbabane).
-Train drivers and passengers to use the app.
-Fix issues after launch.

Key Points

-Milestone 11: Pilot launch successful.
-Milestone 12: Training done.

Important Path

-Pilot launch → Training → Fixing issues.

Time & Cost:

-2 weeks, E14000 (for training and launch costs).

Support Phase

Tasks

-Fix bugs and help users.
-Release updates every month.
-Expand the system nationwide.

Key Points

-Milestone 13: First update released.
-Milestone 14: System used across Eswatini.

Important Path

-Ongoing support → Monthly updates → National rollout.

Time & Cost

-Starts after Week 24 (ongoing), E10000/month (support team and tools).

7. Requirements Analysis and Design

To ensure the system aligns with Eswatini's unique transport dynamics and user needs, a structured approach to requirements analysis and design will be adopted. This phase integrates local context, technical feasibility, and regulatory compliance to deliver a scalable solution.

---

1. Requirements Analysis Methods & Techniques

Stakeholder Engagement

Structured Interviews & Workshops:

- **Method**: Conduct interviews with *kombi operators*, ETRC officials, and rural commuters to identify pain points (e.g., fare disputes, route unpredictability).
- **Tools**: Audio recorders, translated questionnaires (English/SiSwati), and Miro boards for collaborative mapping of informal routes.

- **Output**: Prioritized user stories (e.g., "As a rural commuter, I need SMS-based ETAs to plan my trip during rainy seasons").

Field Observations:

- **Method**: Shadow drivers and commuters in high-traffic corridors (e.g., Manzini-Mbabane) to document behavioral patterns and infrastructure gaps.
- **Tools**: Mobile data collection apps (KoBoToolbox) for offline note-taking in areas with poor connectivity.

Surveys & Beta Testing:

- **Method**: Deploy USSD-based surveys to gather feedback from rural users on preferred communication channels.
- **Tools**: Africa's Talking API for USSD integration, Google Forms for urban beta testers.

Functional & Non-Functional Requirements

Functional:

- Real-time GPS tracking with SMS/USSD fallback.
- Multilingual app (English/SiSwati) with offline maps.

Non-Functional:

- Low-data usage (<5MB/month for rural users).
- Compliance with Eswatini Data Protection Act (2021).

2. System Design Techniques

Architectural Design

Hybrid Tracking System:

- **Technique**: Layered architecture combining GPS data (primary) and cellular triangulation (fallback).
- **Tools**: AWS IoT Core for device management, OpenStreetMap APIs for route visualization.

Offline-First App Design:

- **Technique**: Cache critical data (e.g., route schedules) locally on users' devices.
- **Tools**: Flutter with Hive database for offline storage.

Interface Design

User-Centric Wireframing:

- **Technique**: Develop low-fidelity prototypes for rural users, emphasizing large buttons and voice-guided USSD menus.
- **Tools**: Figma for app mockups, InVision for stakeholder feedback.

Accessibility Standards:

- **Technique**: Follow WCAG 2.1 guidelines for color contrast and font sizes to accommodate low-literacy users.

Data Flow & Security Design

Data Anonymization:

- **Technique**: Strip personally identifiable information (PII) from commuter tracking data.
- **Tools**: Python scripts with Pandas for data preprocessing, AES-256 encryption.

Predictive ETA Algorithms:

- **Technique**: Machine learning models trained on historical traffic data from ETRC.
- **Tools**: TensorFlow, Jupyter Notebooks for prototyping.

3. Resources & Tools

Category	Resources/Tools	Purpose
Data Collection	KoBoToolbox, USSD surveys	Gather requirements from low-connectivity rural users.
Design & Prototyping	Figma, InVision, Flutter	Create intuitive interfaces for diverse literacy levels.
Backend Development	AWS IoT Core, Django REST Framework	Manage GPS data streams and API integrations.
Data Analysis	Python (Pandas, TensorFlow), Power BI	Model traffic patterns and generate ETA predictions.
Compliance	GDPR-compliant encryption tools, ETRC route registries	Ensure adherence to local and international regulations.

4. Contextual Adaptations for Eswatini

Low-Tech Prototyping:

- Use paper prototypes during workshops with rural chiefs to validate USSD menu flows.

Cultural Sensitivity:

- Partner with UNESWA linguists to ensure SiSwati translations are dialect-appropriate (e.g., Manzini vs. Shiselweni regions).

Infrastructure Constraints:

- Design for intermittent power supply: Solar-powered GPS devices with battery backups.

5. Validation & Iteration

Design Walkthroughs:

- Present wireframes to kombi associations for feedback on fare transparency features.

Pilot Testing:

- Deploy a minimal viable product (MVP) in Mbabane (Month 4) to validate GPS accuracy in mountainous terrain.

Continuous Feedback:

- Use Jira to log design changes based on commuter complaints (e.g., adjusting ETA refresh rates).

Alignment with National Goals

The requirements analysis and design phase directly support Eswatini's *National Digital Strategy* by prioritizing inclusive, low-cost solutions. By leveraging local partnerships (e.g., MTN Eswatini for USSD) and global tools (e.g., AWS), the project balances innovation with practicality, ensuring long-term viability in both urban hubs and rural communities.

8. Implementation.

The Implementation phase is an important stage in the software development lifecycle where the design of the system is transformed into executable code. This phase involves translating the requirements and design specifications into a functional software system that meets the needs of stakeholders. Below is a detailed breakdown of the implementation process for the Public Transport Tracker in Eswatini project, including the methods, tools, resources, and techniques to be used.

Methods and Techniques

To ensure a structured and efficient implementation, the following methods and techniques will be employed:

Object-Oriented Design (OOD)

The system will be implemented using an object-oriented approach, as outlined in Sommerville's Software Engineering (Chapter 7). Moreover, UML diagrams (e.g., class diagrams, sequence diagrams) will guide the implementation by clearly defining the relationships between components and their interactions. The key components include GPS Module Integration that is responsible for transmitting vehicle location data also Backend Server that processes GPS data, calculates ETAs, and manages user requests finally a Frontend Application that provides a user-friendly interface for commuters and operators.

Test-Driven Development (TDD)

Test-driven development will be adopted to ensure high-quality code and early detection of defects. For each feature or functionality, we will write an automated test case based on the specification also implement the functionality to pass the test furthermore the team will refactor the code for optimization while ensuring all tests continue to pass finally tools like JUnit (for Java) or Pytest (for Python) will be used to automate testing.

Agile Practices

The iterative nature of the Iterative Waterfall Model aligns well with agile practices such as pair programming and continuous integration. the code will be integrated into a shared repository (GitHub) regularly, and automated builds will verify its correctness.

Host-Target Development

Since the system involves embedded GPS modules, a host-target development environment will be used. the host environment is a development and testing of backend and frontend components on local machines. Also target environment deployment of GPS modules and backend servers to real-world vehicles and cloud infrastructure.

Resources and Tools

The following resources and tools will support the implementation phase.

Hardware

GPS Modules **-** Installed in vehicles to transmit location data.

Development Machines **-** Laptops or desktops for developers to write and test code.

Testing Devices **-** Smartphones and tablets for mobile app testing.

Software

Programming Languages:

Backend- Python or Node.js.

Frontend - React Native for cross-platform mobile apps.

Database -PostgreSQL with PostGIS for geospatial queries.

Integrated Development Environments (IDEs):

Visual Studio Code, PyCharm, or IntelliJ IDEA.

Version Control - GitHub for source code management.

Testing Frameworks - JUnit or Selenium for automated testing.

Cloud Platforms - AWS or Azure for hosting the backend server and database.

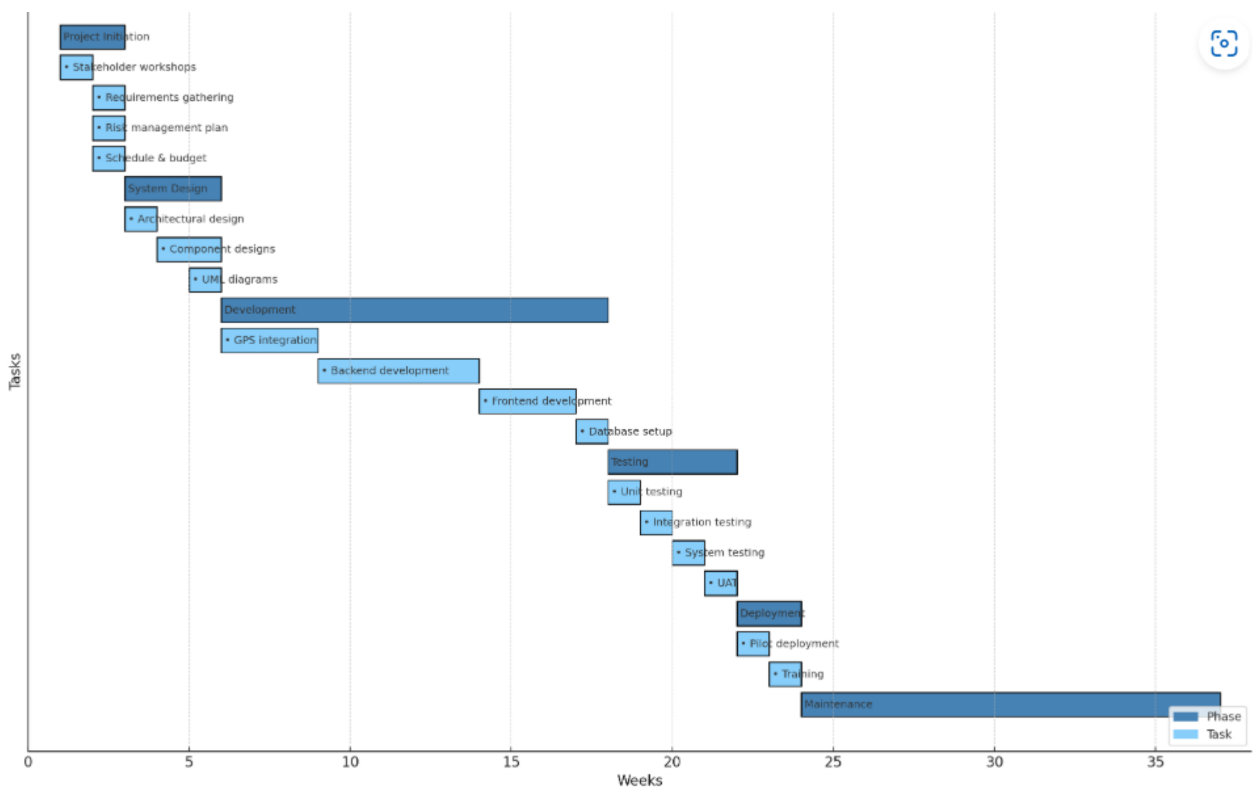
Libraries and APIs:

Mapping APIs - Google Maps API or OpenStreetMap for displaying vehicle locations.

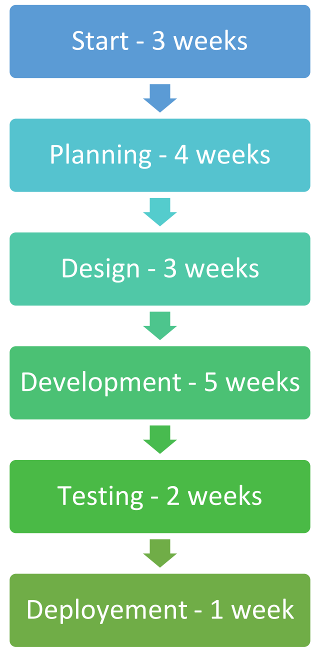
Push Notification Services - Firebase Cloud Messaging (FCM) for sending alerts to users.

Data Processing Libraries - Pandas (Python) for analyzing historical traffic data.

Gantt Chart



PERT Diagram



9. Testing

Test Setup

Category	Details
Hardware Setup	- GPS devices in vehicles - Test phones/tablets (iOS/Android) - Cloud server (AWS/Azure) - PostgreSQL database with PostGIS
Software Setup	- OS: Windows, macOS, Linux (backend), iOS/Android (mobile) - Browsers: Chrome, Firefox, Safari, Edge - Network: 3G, 4G, Wi-Fi simulations
Real-World Testing	- Pilot deployment in Mbabane with real users (commuters and operators)

Test Equipment/Tools

Tool Category	Tools Used
Automated Testing	JUnit, Pytest, Selenium
Performance Testing	Apache JMeter
Debugging	Browser dev tools, Visual Studio Code, PyCharm
Version Control	GitHub
Monitoring	Prometheus, Grafana

Scope of Testing

Test Type	Description
Functional Testing	Check features like real-time tracking and ETA calculations.
Non-Functional Testing	Test speed, security, and ease of use.
Integration Testing	Ensure GPS, server, and app work together.
System Testing	Test the entire system end-to-end.
User Acceptance (UAT)	Confirm the system meets user needs in Mbabane.

Order of Testing

Step	Phase	Details
1	Unit Testing	Test individual parts (e.g., GPS module).
2	Component Testing	Test combined parts (e.g., server + database).
3	Integration Testing	Test interactions (e.g., app ↔ server).
4	System Testing	Test full system in staging environment.
5	User Acceptance (UAT)	Real-world testing in Mbabane.

Testing Procedures

Test Phase	Focus	Example
Unit Testing	Small code parts (methods/functions)	Test GPS data errors.
Component Testing	Combined parts (APIs, server + database)	Check API endpoints for vehicle locations.
Integration Testing	Data flow between systems	GPS → server → app. Stress-test with 1,000 vehicles.
System Testing	End-to-end system validation	Test real-time updates with multiple users.
User Acceptance (UAT)	Real-world feedback and fixes	Train users in Mbabane and collect feedback.

Testing Guidelines

Guideline Type	Description
Partition Testing	Group similar tests (e.g., test different GPS coordinates).
Guideline-Based Testing	Test common errors (e.g., data overflow, repeated inputs).
Path Testing	Test all code paths (e.g., all "if" statement outcomes).

Challenges & Solutions

Challenge	Solution
Can’t perfectly copy real-world	Simulate real conditions (e.g., network speeds).
Parts not working together	Test connections step by step.
Limited time/resources	Prioritize critical tests and automate repetitive tasks.

10. Hardware and Tools Required for Project Execution

To ensure the successful implementation of the Public Transport Tracker in Eswatini, the following resources are critical:

Hardware Resources

GPS Tracking Devices:

- Solar-powered GPS units for rural *kombis* (to address unreliable power supply).
- Rugged, weatherproof devices for durability in mountainous regions (e.g., Mbabane).

Servers & Cloud Infrastructure:

- Local cloud servers (hosted in partnership with ESCCOM) for data storage and processing.
- Backup servers in Manzini for redundancy.

Mobile Devices:

- Low-cost smartphones/tablets for beta testing with commuters (offline-capable models).
- USSD/SMS-enabled feature phones for rural user testing.

Field Equipment:

- Installation kits (tools, cables) for GPS device deployment.
- Portable power banks and solar chargers for field technicians.

Software & Development Tools:

Tracking & Communication:

- Hybrid GPS-SMS/USSD integration tools (e.g., Twilio API for SMS, Africa's Talking for USSD).
- OpenStreetMap APIs for Eswatini-specific route visualization.

Development Frameworks:

- Flutter (cross-platform app development for Android/iOS).
- Python/Django for backend algorithms (ETA predictions, traffic analytics).

Testing Tools:

- Postman (API testing).
- JMeter (load testing for high user concurrency in urban hubs like Manzini).
- GPS simulation tools (e.g., NS-3) to mimic low-coverage rural areas.

Data Security:

- Encryption tools (AES-256) compliant with Eswatini Data Protection Act (2021).
- OAuth 2.0 for secure user authentication.

Collaboration & Management Tools

Project Management:

- Jira (Agile sprint tracking, bug reporting).
- Trello (stakeholder task coordination with kombi associations).

Communication:

- Slack/Microsoft Teams (internal team coordination).
- WhatsApp Business API (for driver/kombi operator updates).

Documentation:

- Confluence (technical wikis, user manuals in English/SiSwati).
- Google Workspace (real-time collaboration on reports).

Partnerships & External Resources

Telecom Providers:

- MTN Eswatini/ECCELL (cellular data/SMS bundles for USSD fallback).

Government Support:

- ETRC (access to vehicle registries, route permits).
- ESCCOM (server hosting, regulatory compliance).

Training Resources:

- Localized training modules for drivers (SiSwati audio/video guides).
- Workshops at UNESWA (University of Eswatini) for beta tester onboarding.

Contingency Resources

Backup Hardware:

- Reserve GPS devices (10% surplus) to replace faulty units.

Alternative Power Solutions:

- Portable solar panels for rural field teams.

Budget Reserves:

- 15% contingency fund (for hardware repairs or telecom cost overruns).

Procurement Timeline

Phase 1 (Months 1-2): Procure GPS devices, cloud infrastructure, and development licenses.

Phase 2 (Month 3): Deploy field equipment and beta-testing devices.

Phase 3 (Month 6): Secure bulk SMS/USSD packages from telecom partners

11. Quality Assurance

To ensure the software meets quality requirements and aligns with Eswatini's infrastructural and regulatory context, a rigorous QA framework will be implemented. This includes organizational structures, testing protocols, and compliance checks tailored to local challenges such as rural connectivity gaps and informal transport dynamics.

1. QA Organization & Roles

- **QA Lead**: Oversees all testing activities, coordinates with the Technical Lead, and ensures compliance with standards.
- **Testing Team**:
  - **Functional Testers**: Validate core features (e.g., GPS tracking accuracy, USSD/SMS reliability).
  - **Security Testers**: Ensure data encryption and compliance with the Eswatini Data Protection Act.
  - **Field QA Officers**: Conduct on-ground testing in rural areas (e.g., Nhlangano) and urban hubs (Manzini).
- **Compliance Officer**: Verifies adherence to national regulations (e.g., ETRC's Transport Code) and international standards (ISO/IEC 25010).

2. QA Procedures & Techniques

Testing Phases

Unit Testing:

- **Method**: Developers test individual modules (e.g., ETA algorithm, GPS data parser).
- **Tools**: JUnit (Java), Pytest (Python), SonarQube for code quality.
- **Focus**: Validate logic for edge cases (e.g., GPS signal loss in Mbabane's mountains).

Integration Testing:

- **Method**: Test interactions between GPS devices, backend APIs, and the commuter app.
- **Tools**: Postman (API testing), Selenium (UI automation).
- **Focus**: Ensure SMS/USSD fallback activates during cellular network outages.

System Testing:

- **Method**: End-to-end validation of the entire system under real-world conditions.
- **Tools**: JMeter (load testing for 10,000+ concurrent users in peak hours).
- **Focus**: Performance in low-bandwidth rural areas (e.g., Shiselweni).

User Acceptance Testing (UAT):

- **Method**: Beta testing with commuters, drivers, and ETRC officials.
- **Tools**: USSD surveys, in-app feedback forms.
- **Focus**: Validate usability for low-literacy users (e.g., voice-guided menus in SiSwati).

Automated vs. Manual Testing

Automated:

- Regression testing for frequent updates (e.g., fare calculation changes).
- Tools: Selenium, Appium (mobile app testing).

Manual:

- Exploratory testing for cultural context (e.g., validating route names in local dialects).
- Field testing of GPS devices in extreme weather (e.g., rainy season in Siteki).

3. Quality Standards & Compliance

Functional Standards:

- GPS accuracy: ≤50m deviation in urban areas, ≤100m in rural zones.
- ETA precision: ≤5-minute error margin during peak hours.

Security Standards:

- AES-256 encryption for commuter data.
- OAuth 2.0 authentication for admin dashboards.

Regulatory Compliance:

- Eswatini Data Protection Act (2021): Anonymize commuter location data.
- ESCCOM Telecom Guidelines: Ensure SMS/USSD services do not overload networks.

4. QA Documentation

- **Test Plans**: Outline scope, schedules, and responsibilities (e.g., testing GPS devices in Mbabane by Month 5).
- **Traceability Matrix**: Link test cases to requirements (e.g., connecting USSD fare alerts to user story).
- **Defect Reports**: Log issues in Jira with severity levels (e.g., critical = GPS device failure).
- **Compliance Checklists**: Verify adherence to ETRC and AU Agenda 2063 standards.

5. Tools & Resources

Category	Tools/Resources	Purpose
Test Automation	Selenium, Appium, JMeter	Validate app performance under load and network constraints.
Security Testing	OWASP ZAP, Burp Suite	Identify vulnerabilities in APIs and user authentication.
Field Testing	Portable network simulators, solar-powered devices	Replicate low-coverage scenarios in rural areas.
Compliance	GDPR-compliant audit tools, ETRC guidelines	Ensure alignment with local and international regulations.

6. Contextual Adaptations

Rural Testing:

- Use MTN Eswatini’s network throttling tools to simulate 2G speeds in regions like Nhlangano.
- Partner with local chiefs to recruit beta testers for USSD menus.

Cultural Validation:

- Ensure UI icons (e.g., bus symbols) are culturally recognizable.
- Test voice prompts in regional SiSwati dialects.

7. Metrics & Reporting

- **Defect Density**: Track bugs per module (target: <0.1 defects/function point).
- **Test Coverage**: Achieve ≥90% coverage for critical modules (e.g., ETA algorithm).
- **User Satisfaction**: Measure via post-UAT surveys (target: ≥85% approval).

- **Weekly QA Reports**: Shared with stakeholders, including ETRC and kombi associations.

8. Training & Capacity Building

QA Team:

- Workshops on security protocols (Month 2).
- Training on Eswatini's Data Protection Act (Month 1).

Field Testers:

- Sessions on GPS device troubleshooting (Month 3).

---

9. Separate QA Plan Document

A detailed **Quality Assurance Plan (QAP)** will outline:

- Risk-based testing strategies (e.g., prioritizing flood-prone routes).
- Roles/responsibilities matrix.
- Escalation procedures for critical defects (e.g., GPS data leaks).
- Disaster recovery protocols (e.g., server outages in Manzini).

Alignment with National Goals:

The QA framework supports Eswatini's *National Development Plan* by ensuring reliability and accessibility in public transport. By combining automated rigor with culturally sensitive manual testing, the system will meet the needs of diverse users—from tech-savvy urban com

12. Changes.

Changes during software development are unavoidable because business needs, user expectations, and technology are always evolving. To ensure changes don't disrupt the project, clear procedures must be in place to manage them in a structured way. These procedures are essential for maintaining quality, consistency, and meeting project goals.

Changes can come from different sources, such as requests from stakeholders, bug fixes, new regulations, changes in hardware or organizational priorities, adoption of new technologies, or feedback from users. According to Sommerville's Software Engineering, these changes often lead to additional costs because they require rework, like revisiting requirements, redesigning, modifying code, and testing again. Managing changes carefully is key to avoiding delays and extra expenses.

To handle changes effectively, several procedures are needed:

Change Request Process: Every proposed change must be submitted using a Change Request Form that explains the change, its reason, impact, and priority. The Change Control Board (CCB), which includes project managers, developers, testers, and stakeholders, evaluates the request to decide if it's feasible and aligns with project goals.

Change Implementation: Once approved, the change is assigned to the team for implementation. Developers update the affected components, ensuring no new issues arise, and document changes in the source code.

Testing and Validation: Modified components go through thorough testing to confirm the changes work and don't negatively affect existing functions. The system is validated to meet both original and new requirements.

Documentation Updates: All related documents, like requirements or designs, must reflect the changes. A record of the changes should be kept in the system's history, as suggested by Sommerville.

A Configuration Management Plan is essential for managing changes. This plan includes version control to track changes using tools like Git, automated build scripts to prevent deployment errors, clear change management procedures, and detailed release records showing what was updated and its effects.

Challenges in managing changes include inconsistent documentation, scope creep, and limited resources. These can be overcome with up-to-date records, prioritizing changes, and adjusting resources or timelines as needed. Managing changes properly ensures the software meets quality standards, avoids delays and extra costs, and maintains trust with stakeholders. Following the outlined procedures and using a robust Configuration Management Plan is vital for success.