A blue and white form with white text

AI-generated content may be incorrect.

**Cardiff Metropolitan University**

Cardiff School of Technology

Bachelor of Science with Honours in Software Engineering

**-LOCATO-**

**Local Services Finder - A Mobile App for Discovering Local Services in Sri Lanka.**

**Walanthi Liyanage Pasan Shiraj Sasmika**

**Cardiff ID: 20281927**

**ICBT ID: GL/BSCSD/07/21**

This dissertation is submitted in partial fulfilment of the requirements for the degree of Bachelor of Science in Software Engineering

**DECLARATION**

This dissertation is submitted in partial fulfilment of the requirements for the award of the **Bachelor of Science (Honours) in Software Engineering**. It is an original work and has not been submitted previously, in whole or in part, for any other degree or qualification.

A close-up of a logo

AI-generated content may be incorrect. Signed -

Candidate Date – 2025/09/06

SUPERVISOR’S DECLARATION STATEMENT

Walanthi Liyanage Pasan Shiraj Sasmika

Student Name – Supervisor’s Name -

Dr. Maneesha Caldera

I acknowledge that the above-named student has regularly attended the meeting and actively engaged in the dissertation supervision process.

Signed - 

……………………………………………. (Supervisor) Date - ……9//7/2025……………......

# Table of Contents

[Table of Contents 4](#_Toc208135341)

[Table Of Figure 7](#_Toc208135342)

[Acknowledgement 8](#_Toc208135343)

[Executive Summary 9](#_Toc208135344)

[1. Introduction 10](#_Toc208135345)

[1.1 Problem Statement 11](#_Toc208135346)

[1.2 Objectives 11](#_Toc208135347)

[1.3 Research Questions 12](#_Toc208135348)

[1.4 Research Objectives 13](#_Toc208135349)

[1.5 Project Solution 13](#_Toc208135350)

[1.6 Significance of the Study 14](#_Toc208135351)

[2. Literature Review 15](#_Toc208135352)

[2.1 Dependent Variable: Service Discovery Efficiency 15](#_Toc208135353)

[2.2 Independent Variables 16](#_Toc208135354)

[2.2.1 Centralized Service Directory 16](#_Toc208135355)

[2.2.2 Location-Based Filtering 16](#_Toc208135356)

[2.2.3 Advanced Search and Filtering Capabilities 16](#_Toc208135357)

[2.2.4 User-Generated Ratings and Reviews 16](#_Toc208135358)

[2.3 Moderating Variable 17](#_Toc208135359)

[2.4 Underlying Variable 17](#_Toc208135360)

[2.5 Gaps in Literature Review 18](#_Toc208135361)

[2.6 Hypothesis Development 18](#_Toc208135362)

[3 .Planning 19](#_Toc208135363)

[3.1 Feasibility Report 19](#_Toc208135364)

[3.1.1 Economic Feasibility 19](#_Toc208135365)

[3.1.2 Schedule Feasibility 20](#_Toc208135366)

[3.1.3 Scope Feasibility 21](#_Toc208135367)

[3.1.4 Technical Feasibility 21](#_Toc208135368)

[3.2 Risk Assessment 22](#_Toc208135369)

[3.3 SWOT Analysis 23](#_Toc208135370)

[3.4 PESTEL Analysis 25](#_Toc208135371)

[3.5 Life Cycle Model 26](#_Toc208135372)

[3.5.1 Detailed Life Cycle Phases (Agile Approach) 27](#_Toc208135373)

[3.6 Time Plan - WBS 28](#_Toc208135374)

[4. Requirement Gathering and Analysis 29](#_Toc208135375)

[4.1 Requirement Gathering 29](#_Toc208135376)

[4.1.1 Choosing the Appropriate Method for Collecting Data 29](#_Toc208135377)

[4.1.2 Fact Gathering Using Selected Techniques 30](#_Toc208135378)

[4.1.3 Requirements Determination 31](#_Toc208135379)

[4.2 The Software Process Model 32](#_Toc208135380)

[4.3 List of All User Requirements 32](#_Toc208135381)

[4.4 Explanation of the Current System 35](#_Toc208135382)

[4.5 Drawbacks of the Current System 35](#_Toc208135383)

[4.6 Explanation of the Proposed System 36](#_Toc208135384)

[5. System Design 36](#_Toc208135385)

[5.1 Design Overview 36](#_Toc208135386)

[5.1.1 Wireframes 37](#_Toc208135387)

[5.2 UML Diagrams 39](#_Toc208135388)

[5.2.1 Use Case Diagram 39](#_Toc208135389)

[5.2.2 Sequence Diagram 42](#_Toc208135390)

[5.2.3 Class Diagram 43](#_Toc208135391)

[5.2.4 Context Diagram (DFD Level 0) 44](#_Toc208135392)

[5.3 Introduction to Database Design 46](#_Toc208135393)

[5.3.1 Corrected Entity Relationship Diagram (ERD) 46](#_Toc208135394)

[6. Implementation 47](#_Toc208135395)

[6.1 Implementation Overview 47](#_Toc208135396)

[6.2 User Interfaces 49](#_Toc208135397)

[6.2.1 User Input Interface Designs 49](#_Toc208135398)

[6.2.2 Service Owner’s User Interfaces 52](#_Toc208135399)

[7. Testing 53](#_Toc208135400)

[7.1 Testing Overview 53](#_Toc208135401)

[7.1.1 Test Strategy 54](#_Toc208135402)

[7.2 Test Plan 55](#_Toc208135403)

[7.3 Test Cases 56](#_Toc208135404)

[8. Critical Evaluation & Conclusion 58](#_Toc208135405)

[8.1 Summary of the Project 58](#_Toc208135406)

[9. Lessons Learned & Critical Evaluation 60](#_Toc208135407)

[10 Future Implementation Plans 61](#_Toc208135408)

[Conclusion 62](#_Toc208135409)

[Appendix 63](#_Toc208135410)

[Appendix 1 (WBS) 63](#_Toc208135411)

[Appendix 2 (Google form) 63](#_Toc208135412)

[Appendix 3 (Version Control) 64](#_Toc208135413)

[Appendix 4 (Expo Go) 65](#_Toc208135414)

[Appendix 5 ( API Testing ) 65](#_Toc208135415)

[Appendix 6 ( Stack Navigation ) 66](#_Toc208135416)

[Appendix 7 ( State management library ) 66](#_Toc208135417)

[Appendix 8 (Login and Signup ) 67](#_Toc208135418)

[Appendix 9 ( Home Screen fetching services) 68](#_Toc208135419)

[Appendix 10 (Database ) 68](#_Toc208135420)

[References 69](#_Toc208135421)

# Table Of Figure

[Figure 1 Agile methodology 23](#_Toc208135191)

[Figure 2 WBS of the system 26](#_Toc208135192)

[Figure 3: Summary Charts of online Survey 27](#_Toc208135193)

[Figure 4: wire frame of the user view search, filter services 34](#_Toc208135194)

[Figure 5: wireframe of the service owner applying for service 35](#_Toc208135195)

[Figure 6: Wireframe of the admin management 35](#_Toc208135196)

[Figure 7: User use case diagram 36](#_Toc208135197)

[Figure 8: Service owner use case diagram 37](#_Toc208135198)

[Figure 9: admin use case diagram 38](#_Toc208135199)

[Figure 10: System Sequences diagram 39](#_Toc208135200)

[Figure 11: System class diagram 41](#_Toc208135201)

[Figure 12: System DFD L0 diagram 42](#_Toc208135202)

[Figure 13: Satyam ER diagram 44](#_Toc208135203)

[Figure 14: Login page input fields code 47](#_Toc208135204)

[Figure 15: Use state for handle input state 47](#_Toc208135205)

[Figure 16: Fetching service data from database (API call) 48](#_Toc208135206)

[Figure 17: Image upload handling 48](#_Toc208135207)

[Figure 18: Profile Page 49](#_Toc208135208)

[Figure 19: Select category for applying 49](#_Toc208135209)

[Figure 20: Service application 50](#_Toc208135210)

[Figure 21: WBS 60](#_Toc208135211)

[Figure 22: Result of Survey 61](#_Toc208135212)

[Figure 23: GitHub frontend repository 61](#_Toc208135213)

[Figure 24: GitHub backend repository 61](#_Toc208135214)

[Figure 25: Expo Go application 62](#_Toc208135215)

[Figure 26: API testing 62](#_Toc208135216)

[Figure 27: Stack Navigations 63](#_Toc208135217)

[Figure 28: Global state management (Zustand) 63](#_Toc208135218)

[Figure 29: Signup screen UI 64](#_Toc208135219)

[Figure 30: Login screen UI 64](#_Toc208135220)

[Figure 31: Home screen with fetching services 65](#_Toc208135221)

[Figure 32: MongoDB Database 65](#_Toc208135222)

# Acknowledgement

I would like to express my deepest gratitude to my supervisor, Dr. Maneesha Caldera, for their invaluable guidance, unwavering support, and insightful feedback throughout the duration of this research project. Their expertise and encouragement were instrumental in shaping this thesis.

My sincere thanks also go to the academic staff of the ICBT Campus for providing the foundational knowledge and technical skills that made this project possible.

I am also grateful to my family and friends for their constant support, patience, and motivation during the challenging yet rewarding journey of my final year.

# Executive Summary

The rapid digital transformation in Sri Lanka, driven by widespread smartphone adoption and internet penetration, has significantly altered how individuals access services and information. Despite the growth of digital solutions in sectors such as e-commerce, transportation, and food delivery, local service discovery remains a critical challenge. Residents often rely on word-of-mouth recommendations, fragmented social media groups, or generic mapping tools, which are inefficient and unreliable for finding verified local services.

Simultaneously, many small and medium-sized enterprises (SMEs) and individual service providers remain digitally invisible, limiting their ability to reach potential customers. Addressing this gap, this thesis presents **"Locato"**, a mobile application designed to serve as a centralized, interactive hub for discovering and accessing local services across Sri Lanka.

Locato integrates modern mobile technologies and location-based services to connect service seekers with providers efficiently. The application not only offers a searchable and categorized directory of verified services but also facilitates user interaction through reviews and ratings, enhancing trust and transparency. By focusing on user-centric design and accessibility, Locato aims to improve convenience, save time, and empower both consumers and local service providers.

Through the design, development, and evaluation of Locato, this project demonstrates a practical approach to bridging the digital visibility gap in the Sri Lankan service sector and contributes to the broader discourse on mobile-enabled local service ecosystems.

# Introduction

The digital revolution, catalysed by the widespread adoption of smartphones and ubiquitous internet access, has fundamentally reshaped how individuals interact with the world. In Sri Lanka, this digital transformation is palpable, with mobile penetration exceeding 100% and a growing appetite for digital solutions that simplify daily life. While sectors like e-commerce, transportation, and food delivery have seen significant innovation through mobile applications, a critical gap persists in the local service discovery domain. Finding reliable and verified local services such as a skilled electrician, a nearby tuition class, a trusted healthcare clinic, or an artisanal craft shop remains a persistent and frustrating challenge for most of the population.

Currently, residents in both urban and semi-urban areas of Sri Lanka rely on a patchwork of inefficient methods. These include traditional word-of-mouth recommendations, which are limited in reach; fragmented social media groups, which are often cluttered and difficult to search; or generic mapping tools like Google Maps, which lack detailed, verified, and categorized information specific to smaller, local service providers. This disorganization leads to wasted time, effort, and a lack of consumer confidence. Concurrently, countless small and medium-sized enterprises (SMEs) and individual service providers, who form the backbone of the local economy, remain digitally invisible, struggling to connect with a wider customer base.

This thesis addresses this critical gap by detailing the design, development, and evaluation of "Locato," a comprehensive mobile application designed to function as a centralized hub for local services across Sri Lanka. Locato is not merely a directory; it is an interactive ecosystem that connects service seekers with providers in a seamless, efficient, and trustworthy manner. By leveraging modern mobile technologies, location-based services, and a user-centric design, Locato aims to revolutionize the way local services are discovered, accessed, and reviewed in the Sri Lankan context.

## 1.1 Problem Statement

The core problem is the absence of a centralized, reliable, and user-friendly digital platform for discovering local services in Sri Lanka. This issue manifests in several critical ways:

1. **Inefficiency and Inconvenience for Service Seekers:** Individuals spend an inordinate amount of time and effort searching for essential services. The information they find is often outdated, incomplete, or inaccurate, leading to frustration and poor service experiences.
2. **Lack of Digital Visibility for Service Providers:** A significant number of small businesses and individual service providers lack the technical expertise or financial resources to establish a strong digital presence. Their reliance on traditional marketing methods limits their reach and growth potential, preventing them from competing in the burgeoning digital economy.
3. **Information Asymmetry and Lack of Trust:** Without a standardized system for reviews and ratings, consumers cannot easily gauge the quality or reliability of a service provider. This lack of transparency creates uncertainty and hinders informed decision-making.
4. **Hindrance to Local Economic Growth:** The disconnect between local service providers and potential customers represents a missed economic opportunity. A more efficient marketplace could stimulate local business activity, encourage entrepreneurship, and contribute to the overall economic development of communities.

## 1.2 Objectives

The primary objective of this research project is to design, develop, and deploy a fully functional mobile application, "Locato," that effectively addresses the identified problems. The project is guided by the following specific objectives:

1. To develop a cross-platform mobile application that provides a centralized and categorized directory of local services across Sri Lanka. (Kumar, 2021)
2. To implement an advanced search and filtering system that allows users to find services based on criteria such as category, location, distance, user ratings, and real-time availability. (Yan, 2017)
3. To integrate a robust, location-aware mapping feature using OpenStreetMap (OSM) for visualizing service locations and providing navigation. (Kar, 2022)
4. To create a secure and intuitive system for user registration, along with a role-based access model for consumers, service providers, and administrators. (Gupta, 2020)
5. To implement a user-driven ratings and review system to build a community of trust and enable informed decision-making.
6. To develop a comprehensive administrative dashboard for the verification and management of service provider listings to ensure the quality and integrity of the platform. (Alsharif, 2019)

## 1.3 Research Questions

This study aims to answer the following key research questions:

1. What are the primary challenges and inefficiencies faced by Sri Lankan residents when searching for local services? (Perera, 2021)
2. How can a location-based mobile application be designed to effectively centralize and streamline the service discovery process? (Dey, 2010)
3. What are the critical features (e.g., filtering, reviews, real-time availability) required to create a trustworthy and user-centric service finder application? (Yan, 2017)
4. How can the MERN stack and React Native framework be effectively utilized to build a scalable, cross-platform solution for the Sri Lankan market?
5. To what extent can a platform like Locato contribute to the digital inclusion of small local businesses and the growth of the local economy? (Bank, 2023)

## Research Objectives

The academic and research objectives of this project are:

1. To critically review existing literature on Location-Based Services (LBS), mobile service platforms, and user trust systems to inform the design of the Locato application. (Dey, 2010)
2. To apply a structured software engineering methodology (Agile) for the planning, requirement analysis, design, implementation, and testing of the system. (Agre, 2001)
3. To design and model the system architecture and database schema using industry-standard tools and notations such as UML and ERDs. (Li, 2016)
4. To conduct a thorough evaluation of the developed application to assess its functionality, usability, and performance against the defined objectives.
5. To document the project comprehensively in a formal thesis, contributing to the body of knowledge in mobile application development for emerging economies.

## 1.5 Project Solution

The proposed solution is **Locato**, a mobile application that acts as a comprehensive, geo-aware service directory for Sri Lanka. The platform is architected to be a two-sided marketplace, serving both service seekers (customers) and service providers.

**For Users/Customers:** Locato offers an intuitive interface to browse and search for services across a multitude of categories (e.g., Tuition & Education, Health Care, Home Repairs, Beauty & Wellness). Users can leverage powerful filters to narrow down results, view detailed service profiles, read authentic reviews from other users, and see service locations on an interactive map.

**For Service Providers:** Locato provides a simple yet powerful digital toolkit. A user can request to become a service provider by filling out a form. Upon administrative approval, their role is elevated, and they gain access to a dashboard where they can create and manage their service listings, update their availability, and interact with potential customers. This process democratizes digital marketing, offering an accessible platform for businesses of all sizes.

The entire ecosystem is managed by an administrator via a web-based dashboard, ensuring that all listings are verified and that the platform remains a safe and reliable environment for all users. The use of cost-effective and scalable technologies like React Native, Node.js, and MongoDB ensures the long-term viability of the project.

## 1.6 Significance of the Study

The development of Locato holds significant potential and impact on multiple fronts:

* **Social Impact:** It promotes digital inclusion by bringing non-digitized small businesses into the formal tech ecosystem. It empowers communities by providing easier access to essential services like healthcare and education, thereby improving the quality of life.
* **Economic Impact:** The platform can act as a catalyst for local economic growth. By increasing the visibility of local businesses, it can drive customer traffic and revenue. It fosters entrepreneurship by providing a low-barrier entry point for new service providers to market their skills.
* **Technological Impact:** The project serves as a practical case study for developing and deploying a location-based service application tailored to the specific socio-technical context of a developing country like Sri Lanka. It provides a blueprint that can be adapted and scaled for other regions or service verticals.
* **Academic Contribution:** This thesis contributes a comprehensive account of applying modern software engineering principles to solve a tangible, real-world problem. It offers insights into system design, technology stack selection, and implementation strategies relevant to final-year software engineering students and researchers

# Literature Review

This chapter provides a critical review of existing academic and technical literature relevant to the development of a location-based service discovery platform like Locato. The review is structured to explore the foundational concepts, existing solutions, and theoretical frameworks that underpin this project. It examines key areas including Location-Based Services (LBS), mobile application architecture, user trust mechanisms, and the socio-technical impact of such platforms in developing economies. By synthesizing findings from previous research, this chapter establishes the academic context for Locato, identifies existing gaps, and informs the development of the project's hypotheses.

(Dey, 2010) defines context-awareness as a system's ability to use context to provide relevant information and/or services to the user, where relevancy depends on the user's task. Location is one of the most significant and widely used contextual cues. Locato is fundamentally a context-aware application where the primary context is the user's geographical location.

## 2.1 Dependent Variable: Service Discovery Efficiency

The primary dependent variable that Locato aims to influence is **Service Discovery Efficiency**. This can be defined as the effectiveness and ease with which a user can find and connect with a relevant and reliable local service provider. It is a multi-faceted construct measured by factors such as:

* **Time to Find Service:** The duration from initiating a search to identifying a suitable provider.
* **Relevance of Results:** The degree to which the search results match the user's specific needs.
* **User Confidence:** The level of trust a user has in the information provided and their confidence in selecting a service.

Traditional methods of service discovery are demonstrably inefficient. Research by (Agre, 2001) on the social implications of ubiquitous computing highlights how informal information networks, like word-of-mouth, while trusted, are inherently limited in scale and speed. Locato directly addresses this by formalizing and scaling this discovery process.

## 2.2 Independent Variables

Several independent variables are manipulated or introduced by the Locato system to improve service discovery efficiency.

### 2.2.1 Centralized Service Directory

The most fundamental variable is the existence of a single, comprehensive platform. Currently, information is fragmented across disparate sources. (Li, 2016) explored crowd-sourced mobile service platforms, noting both their opportunities and challenges. Their work validates the effectiveness of community-driven service listings, a core principle of Locato. A centralized directory reduces the cognitive load on users, as they no longer need to consult multiple platforms, thereby improving efficiency.

### 2.2.2 Location-Based Filtering

The use of real-time user location is a critical independent variable. (Dey, 2010)provide a foundational overview of the architecture and applications of Location-Based Services (LBS). They describe the core components positioning systems (GPS), service providers, and mobile networks that enable applications like Locato. Their architectural descriptions support Locato’s use of GPS for proximity-based service discovery. (Zhuang, 2015) further explored location-based mobile social applications, emphasizing the integration of location with service information to create value. This directly guides Locato’s core functionality of showing the nearest services first.

### 2.2.3 Advanced Search and Filtering Capabilities

Beyond simple location, the ability to filter by multiple criteria (category, price, rating) is another key variable. (Yan, 2017) proposed methods for personalized service recommendations using location and user behaviour data. While Locato's initial version does not include an AI-driven recommendation engine, their work provides the theoretical basis for the importance of personalization. The advanced filtering in Locato is a step towards this, allowing users to manually personalize their search results, which is hypothesized to significantly improve the relevance of results.

### 2.2.4 User-Generated Ratings and Reviews

The trust mechanism is a powerful independent variable. (Gupta, 2020) conducted a study on the role of reviews and ratings in consumer decisions, finding a strong positive correlation between user-generated content and purchasing intent. Their findings empirically validate the inclusion of a robust review system in Locato as a critical feature for building user trust and confidence, thus improving the overall efficiency of the decision-making process.

## 2.3 Moderating Variable

**Digital Literacy and Internet Penetration**

The relationship between Locato's features (independent variables) and service discovery efficiency (dependent variable) is moderated by the digital literacy of the user base and the quality of internet penetration in Sri Lanka. (Alsharif, 2019) studied the impact of mobile applications in developing countries, concluding that while apps improve access to information, their effectiveness is often constrained by factors like low digital literacy and inconsistent internet connectivity. This is a crucial consideration for Locato. The user interface must be exceptionally intuitive to cater to users with varying levels of technical skill. Furthermore, the application's performance in low-bandwidth conditions is a technical challenge that must be addressed, as it directly impacts usability and adoption.

## 2.4 Underlying Variable

**Choice of Technology**

The underlying technology stack is a variable that influences the entire system's performance, scalability, and feasibility.

* **Mapping Technology:** The choice between proprietary and open-source mapping APIs is critical. (Kar, 2022) performed a comparative analysis of Google Maps and OpenStreetMap (OSM) APIs. They found that while Google Maps offers extensive data, OSM provides superior cost-efficiency and customization capabilities. This finding strongly supports Locato's choice of using OpenStreetMap (via React Native Maps), which aligns with the project's goal of being a low-cost, scalable solution for the Sri Lankan context.
* **Cross-Platform Development:** The selection of React Native as the front-end framework is another key technical decision. (Kumar, 2021)in their review of location-aware mobile apps, highlighted the benefits of cross-platform development in reducing development time and cost, which is crucial for a project with limited resources. This approach allows Locato to reach both Android and iOS users with a single codebase.

## 2.5 Gaps in Literature Review

While the existing literature provides a strong foundation for LBS and mobile platforms, several gaps are evident, particularly in the context of Sri Lanka:

1. **Hyper-Local Focus in Developing Nations:** Most research on LBS focuses on large-scale applications in developed economies (e.g., Uber, Yelp). There is a scarcity of academic literature focusing on the unique challenges and opportunities of developing hyper-local, community-centric service discovery platforms in countries like Sri Lanka, where the informal economy is significant.
2. **Admin-Moderated Trust Models:** Much of the research on user trust focuses on purely crowd-sourced review systems. The hybrid model proposed by Locato-where service providers are vetted by an administrator before they can be listed-is less explored. This model aims to provide a baseline level of trust before user reviews even come into play.
3. **Comparative Analysis of Low-Cost Tech Stacks:** While individual technologies are often studied, there is limited research providing a holistic analysis of using a specific low-cost stack (MERN + React Native + OSM) to build a service platform within the economic and technical constraints of a developing country.

## 2.6 Hypothesis Development

Based on the literature review and the identified gaps, the following hypotheses are formulated for this project:

* **H1:** The implementation of a centralized, location-aware mobile directory (Locato) will significantly reduce the time and effort required for users in Sri Lanka to find local services compared to traditional methods (word-of-mouth, social media).
* **H2:** The inclusion of an advanced filtering system based on category, distance, and user ratings will lead to a higher perceived relevance of search results for the user.
* **H3:** A service discovery platform that incorporates both administrative verification of service providers and user-generated reviews will result in a higher level of user trust than platforms relying solely on unverified listings.
* **H4:** Empowering local service providers with a simple, self-service digital platform will increase their market visibility and access to new customers.

This literature review confirms that while the foundational technologies for Locato are well-established, its specific application, feature combination, and socio-economic context represent a novel contribution to both practice and research. The project is well-positioned to address the identified gaps and test the proposed hypotheses.

# 3 .Planning

Effective project planning is the cornerstone of successful software development. This chapter outlines the comprehensive planning phase undertaken for the Locato project. It encompasses a thorough feasibility analysis to ensure the project's viability, a risk assessment to proactively manage potential challenges, strategic analyses using SWOT and PESTEL frameworks, and the selection of an appropriate software development life cycle model. The chapter concludes with a detailed work breakdown structure (WBS) and timeline, providing a clear roadmap for the project's execution.

## 3.1 Feasibility Report

A feasibility study was conducted to evaluate the viability of the Locato project from technical, economic, schedule, and operational perspectives.

### 3.1.1 Economic Feasibility

Economic feasibility assesses the cost-benefit trade-offs of the project. The Locato project is designed to be highly economically feasible.

* **Costs:** The primary costs are associated with development time, domain registration, cloud hosting, and app store publishing fees. As this is an academic project, development labour is not a direct financial cost. The use of open-source technologies (React Native, Node.js, MongoDB, OSM) drastically reduces software licensing fees. The budget plan indicates a total estimated cost of LKR 82,000, which is minimal for a software development project of this scale. This covers essential operational expenses like hosting and API usage, making the initial investment very low.
* **Benefits:** The benefits, while initially non-monetary, are substantial.
  1. **For Users:** Saves time and reduces the frustration associated with finding reliable services.
  2. **For Service Providers:** Provides free digital marketing and access to a larger customer base, potentially increasing their revenue.
  3. **For the Economy:** Stimulates local economic activity and promotes digital transformation.
  4. **Future Monetization:** The platform has a clear path to future revenue generation through premium listings, targeted advertising for businesses, or a commission-based booking system (a potential future enhancement).

**Conclusion:** Given the low initial cost and high potential for social and economic value, the project is deemed highly economically feasible.

### 3.1.2 Schedule Feasibility

Schedule feasibility determines if the project can be completed within a realistic timeframe. The project is scoped for completion within a 4-6month period, which is standard for a final year undergraduate project.

* **Well-Defined Scope:** The project scope is clearly defined: a mobile app for service discovery in Sri Lanka with specific features (search, filter, review, map). Features like in-app payments or AI-based recommendations are explicitly excluded from the initial version to ensure the core product can be delivered on time.
* **Efficient Technology Stack:** The MERN stack and React Native are known for rapid development cycles. React Native's cross-platform nature means that time is not spent developing two separate native applications.
* **Structured Timeline:** A detailed Gantt chart has been created (as referenced in the proposal), breaking down the project into manageable phases: Planning, Design, Development (Frontend/Backend), Testing, and Deployment. Each phase has clear milestones and deliverables.

**Conclusion:** The project timeline is realistic and achievable within the academic semester, making it schedule feasible.

### 3.1.3 Scope Feasibility

Scope feasibility is concerned with how well the defined scope of the project can be achieved. The scope of Locato is to create a service discovery platform for all of Sri Lanka. While ambitious, it is made feasible by:

* **Phased Rollout Strategy:** Although the platform is built for national scale, the initial data population can be focused on specific major cities (like Galle, Colombo, Kandy) and expanded over time through user-contributed listings.
* **Core Functionality Focus:** The project focuses on perfecting the core loop: a user searches, finds, and reviews a service, and a provider lists and manages their service. This core is achievable. Advanced features are deferred.
* **User-Generated Content:** The scalability of the service directory relies on service providers signing up and adding their own listings. This crowdsourcing approach makes a nationwide scope feasible without requiring a massive data entry team.

**Conclusion:** The scope, while broad, is feasible due to a focus on core functionality and a scalable, user-driven content strategy.

### 3.1.4 Technical Feasibility

Technical feasibility assesses the availability of required technology, tools, and expertise.

* **Mature Technologies:** The chosen technology stack (React Native, Node.js, Express.js, MongoDB) is mature, well-documented, and widely used in the industry. There is a vast community and wealth of resources available for troubleshooting.
* **Developer Expertise:** The project is being undertaken by a final-year software engineering student with foundational knowledge in JavaScript, mobile development, and database management, which are the core skills required.
* **Available Tools:** All necessary development tools, including code editors (VS Code), version control (Git/GitHub), API testing tools (Postman), and design tools (Figma), are readily available, many of them for free.
* **Third-Party APIs:** The reliance on the OpenStreetMap ecosystem via React Native Maps and potentially other APIs for authentication is technically sound. These services are reliable and provide sufficient documentation and free tiers for development and initial deployment.

**Conclusion:** The project is technically feasible with the existing knowledge base and available tools.

## 3.2 Risk Assessment

A risk assessment was conducted to identify potential threats to the project and devise mitigation strategies.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Id | Risk Description | Likelihood | Impact | Mitigation Strategy |
| R01 | **Technical Debt:** Poor code quality leads to maintenance issues. | Medium | High | Adhere to coding standards, use version control (Git), conduct regular code reviews (peer review), and refactor code. |
| R02 | **Scope Creep:** Uncontrolled addition of new features. | High | High | Strictly adhere to the defined project scope (MVP). New features to be documented and deferred to future versions. |
| R03 | **Data Quality:** Users or providers enter false or inaccurate data. | High | Medium | Implement an admin verification/approval system for new listings. Implement a user reporting feature for flagging incorrect information. |
| R04 | **Low User Adoption:** Both customers and providers fail to use the app. | Medium | High | Focus on a simple, intuitive UI/UX. Conduct a small-scale promotional campaign upon launch. Gather user feedback early and iterate. |
| R05 | **Security Vulnerabilities:** Data breaches or unauthorized access. | Medium | High | Implement secure authentication (JWT), hash passwords, use HTTPS, and follow security best practices for Node.js and MongoDB. |

## 3.3 SWOT Analysis

A SWOT analysis was performed to evaluate the project's internal strengths and weaknesses, and external opportunities and threats.

* **Strengths:**
  1. **First-Mover Advantage:** Addresses a clear, unmet need in the Sri Lankan market with a dedicated solution.
  2. **Low-Cost Model:** Use of open-source technology makes the platform financially sustainable and scalable.
  3. **User-Centric Design:** Focus on a simple UI/UX and core features like filtering and reviews enhances usability.
  4. **Scalable Architecture:** The MERN stack is well-suited for scaling to accommodate a growing user base.
* **Weaknesses:**
  1. **Limited Initial Database:** The app will launch with a limited number of service listings, which may disappoint early adopters.
  2. **Dependence on User-Generated Content:** The platform's value is directly tied to the active participation of both service providers and reviewers.
  3. **Lack of Brand Recognition:** As a new entrant, building trust and a brand identity will be a challenge.
* **Opportunities:**
  1. **Growing Digital Population:** Increasing smartphone and internet penetration in Sri Lanka creates a large potential user base.
  2. **Government Push for Digitalization:** The Sri Lankan government's focus on digital transformation creates a favourable environment for tech startups.
  3. **Expansion Potential:** The platform can be expanded to include new features like online booking, payments, or specialized service verticals (e.g., tourism, events).
* **Threats:**
  1. **Competition from Incumbents:** Large players like Google Maps or Facebook could enhance their local service listings, posing a competitive threat.
  2. **Data Privacy Regulations:** Evolving data protection laws in Sri Lanka could impose new compliance requirements.
  3. **Market Apathy:** Potential users might be resistant to adopting a new application if existing informal methods are perceived as "good enough."

## 3.4 PESTEL Analysis

A PESTEL analysis examines the broader macro-environmental factors that could influence the project.

* **Political:** A stable political environment is conducive to technological adoption and small business growth. Government initiatives promoting SMEs and digital literacy can positively impact Locato.
* **Economic:** Economic conditions in Sri Lanka affect both consumer spending and the ability of small businesses to thrive. High inflation could impact the cost of services, which could be reflected on the platform. The app itself can support the informal "gig" economy.
* **Socio-cultural:** Sri Lanka has a high mobile phone penetration rate and a growing comfort with digital platforms. There is a strong community culture, which could encourage the adoption of a review-based system. Language diversity (Sinhala, Tamil, English) needs to be considered for future app localization.
* **Technological:** The availability of affordable smartphones and improving 4G/5G connectivity are the primary technological enablers. The global trend towards mobile-first solutions validates the project's approach.
* **Environmental:** The project has a minimal direct environmental impact. Indirectly, it can reduce unnecessary travel and fuel consumption by helping users find the nearest available service provider, contributing positively.
* **Legal:** The project must comply with Sri Lanka's upcoming Data Protection Act regarding the collection and handling of user data. Clear terms of service and a privacy policy are legal necessities.

## 3.5 Life Cycle Model

A diagram of a process

AI-generated content may be incorrect.

Figure 1 Agile methodology

The Agile methodology, specifically a framework inspired by Scrum, was chosen for the Locato project. This choice is justified because:

* **Flexibility:** The project requirements, while well-defined, might need adjustments based on feedback. Agile allows for iterative development and the flexibility to adapt to changes.
* **Iterative Progress:** The project can be developed in small, incremental "sprints," with each sprint delivering a functional piece of the application. This allows for early testing and validation of features.
* **Risk Management:** Developing in iterations means that the highest-priority features are built first, ensuring that a viable product exists early in the timeline. This mitigates the risk of project failure.
* **Stakeholder Engagement:** Although the primary stakeholder is the academic supervisor, the Agile process encourages regular demonstrations and feedback, which is crucial for keeping the project on track.

The Waterfall model was considered but rejected because it is too rigid for a project of this nature. Its lack of flexibility would make it difficult to incorporate feedback or adjust requirements without significant rework.

### 3.5.1 Detailed Life Cycle Phases (Agile Approach)

1. **Sprint 0 (Project Initiation & Planning):**
   1. Activities: Initial research, feasibility study, technology selection, high-level requirements gathering, creation of the product backlog (a list of all desired features).
   2. Deliverables: This thesis chapter, project proposal, initial UI wireframes.
2. **Sprint 1 (Core Backend & User Authentication):**
   1. Activities: Set up the Node.js server, design the MongoDB schema, implement user registration and login APIs with JWT.
   2. Deliverables: Functional API endpoints for user management, tested with Postman.
3. **Sprint 2 (Service Listing & Provider Workflow):**
   1. Activities: Develop APIs for creating, reading, updating, and deleting service listings. Implement the workflow for users to apply to become providers and for admins to approve them.
   2. Deliverables: Backend logic for service management and role elevation.
4. **Sprint 3 (Frontend Setup & Core UI):**
   1. Activities: Set up the React Native project, build reusable UI components (buttons, cards, inputs), implement navigation, and create the main screens (Home, Search, Profile).
   2. Deliverables: A navigable mobile app shell.
5. **Sprint 4 (Connecting Frontend to Backend):**
   1. Activities: Integrate the user authentication and service listing features. Users can log in, and the app can fetch and display a list of services from the backend.
   2. Deliverables: A connected app where data flows from the backend to the frontend.
6. **Sprint 5 (Mapping & Filtering):**
   1. Activities: Integrate React Native Maps with OSM. Display services on the map. Implement the advanced filtering logic on the frontend.
   2. Deliverables: Interactive map view and functional search filters.
7. **Sprint 6 (Reviews & Admin Panel):**
   1. Activities: Implement the feature for users to add ratings and reviews. Develop the basic web-based admin panel for listing approval.
   2. Deliverables: Review system and admin dashboard.
8. **Sprint 7 (Testing & Refinement):**
   1. Activities: Thorough unit testing, integration testing, and user acceptance testing (UAT). Fix bugs and refine the UI/UX based on feedback.
   2. Deliverables: A stable, tested application.
9. **Sprint 8 (Deployment & Documentation):**
   1. Activities: Deploy the backend to a cloud service (e.g., Render), prepare the mobile app for store submission, finalize the project documentation and thesis.
   2. Deliverables: Live application and final thesis submission.

## 3.6 Time Plan - WBS

The Work Breakdown Structure (WBS) decomposes the project into smaller, manageable tasks, which are then scheduled in the project's Gantt chart. **Appendix 1 (WBS)**

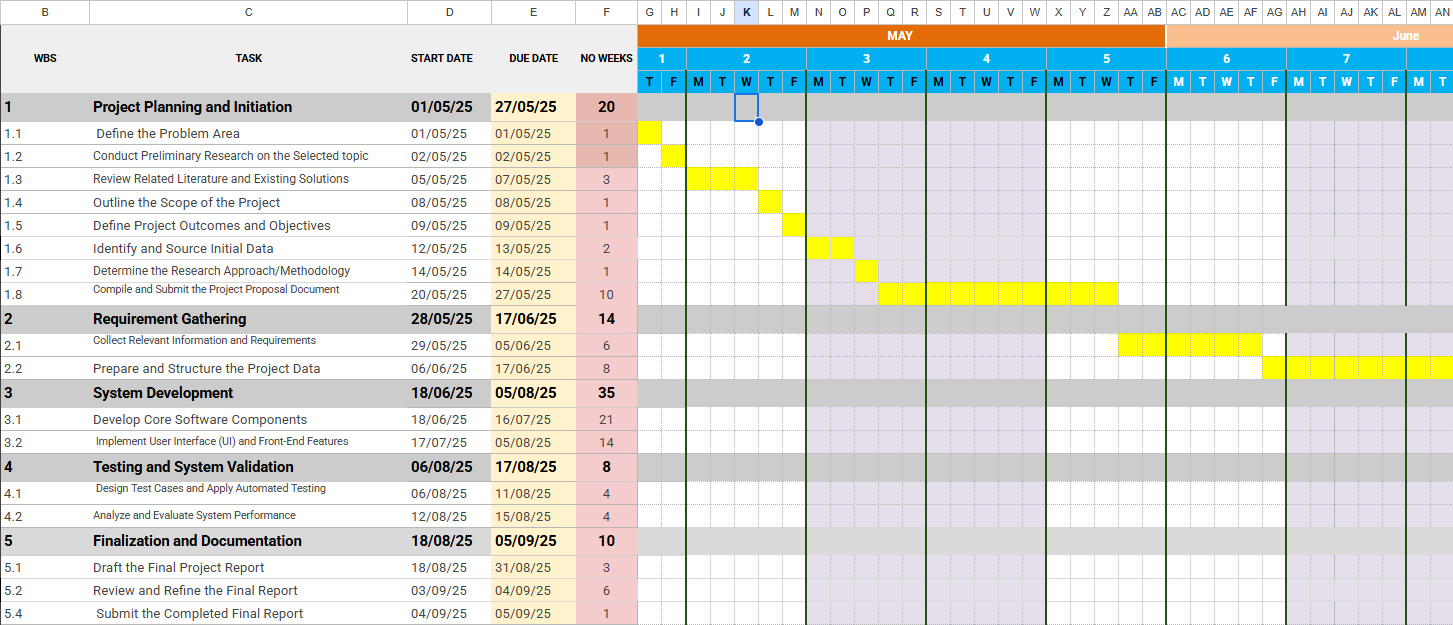


Figure 2 WBS of the system

This detailed plan provides a robust framework for managing the project, ensuring that all aspects are considered and that progress can be tracked effectively against a clear timeline.

# 4. Requirement Gathering and Analysis

This chapter details the process of eliciting, analysing, and documenting the requirements for the Locato system. Requirement analysis is a critical phase in the software development lifecycle, as it forms the foundation for all subsequent design, implementation, and testing activities. A failure to accurately capture requirements is a leading cause of project failure. This section outlines the methods used for data collection, defines the functional and non-functional requirements, and provides a comparative analysis of the current system (or lack thereof) versus the proposed Locato solution.

## 4.1 Requirement Gathering

The process of gathering requirements aimed to understand the needs and expectations of the primary stakeholders: potential service seekers (end-users) and local service providers in Sri Lanka.

### 4.1.1 Choosing the Appropriate Method for Collecting Data

A mixed-method approach was employed to gather comprehensive data:

1. **Online Surveys (Quantitative):** A structured online survey was chosen as the primary method for gathering data from a broad audience of potential end-users across Sri Lanka. This method is cost-effective, allows for anonymous responses, and facilitates the collection of quantitative data that can be easily analysed. The survey was designed using Google Forms and distributed via social media platforms and university networks. The goal was to quantify the problems faced in service discovery and validate the demand for the proposed features.
2. **Informal Interviews (Qualitative):** To gain deeper insights, informal interviews were conducted with a small, targeted group of stakeholders. This included several small business owners (e.g., a local electrician, a tuition master, a salon owner) and a handful of university students and working professionals (representing the service seeker demographic). These interviews were semi-structured, allowing for open-ended questions and follow-up discussions to explore their pain points, needs, and opinions on a digital solution. This method provided rich, qualitative context that a survey alone could not capture.

### 4.1.2 Fact Gathering Using Selected Techniques

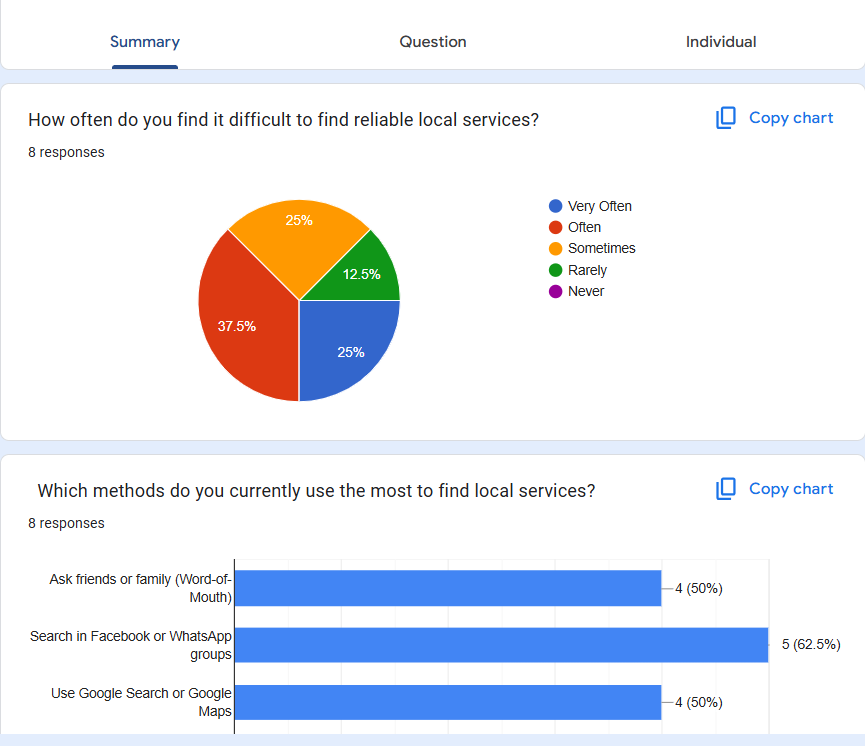


Figure 3: Summary Charts of online Survey

**Appendix 2 (Google form)**

* **70%** of respondents reported facing difficulty in finding reliable local services at least "sometimes" or "often."
* The most common methods currently used were **Word-of-Mouth (42%)**, **Facebook Groups (57%)**, and **Google Search/Maps (42%)**.
* **42%** of respondents expressed frustration with outdated or inaccurate information (e.g., wrong phone numbers, incorrect opening hours).
* When asked to rank the importance of features for a new app, **User Ratings/Reviews (57%)** and **Accurate Location/Contact Info (57%)** were ranked as "Very Important."
* **71%** of respondents stated they would be "Likely" to use a dedicated mobile app for finding local services.

**Interview Insights:** Interviews with service providers revealed several key themes:

* **Desire for Digital Presence:** Most small business owners acknowledged the need to be online but felt overwhelmed by the complexity and cost of building a website or running social media campaigns.
* **Fear of Negative Reviews:** While they understood the value of reviews, some expressed concern about unfair or malicious feedback. This highlighted the need for a fair moderation policy.
* **Simplicity is Key:** They emphasized that any platform for them must be extremely easy to use, especially for creating and updating their service listings from a mobile phone.

Interviews with service seekers confirmed the survey data, with individuals sharing specific anecdotes of time wasted searching for services, especially during emergencies or when new to an area.

### 4.1.3 Requirements Determination

Based on the gathered data, the system requirements were determined and categorized into functional and non-functional requirements.

## 4.2 The Software Process Model

As established in the planning chapter, the Agile methodology was selected. The requirement gathering process directly feeds into the Agile workflow. The insights from surveys and interviews were used to create a Product Backlog, which is a prioritized list of user stories.

**Example User Stories:**

* **As a customer,** I want to search for a service by category so that I can easily find what I'm looking for.
* **As a customer,** I want to filter search results by my current location so that I can find the nearest service provider.
* **As a service provider,** I want to create a profile with my business details, contact information, and operating hours so that customers can find and contact me.
* **As an administrator,** I want to review and approve new service provider registrations so that I can maintain the quality of the platform.

These user stories were then grouped into sprints for iterative development, allowing for a flexible and responsive development process.

## 4.3 List of All User Requirements

**Functional Requirements (FR)**

These define the specific behaviours and functions of the system.

* **FR1: User Management**
  + FR1.1: Users shall be able to register for a new account using an email and password.
  + FR1.2: Registered users shall be able to log in and log out of the system securely.
  + FR1.3: The system shall support three user roles: Customer, Service Provider, and Administrator.
* **FR2: Service Provider Onboarding**
  + FR2.1: A registered user (customer) shall be able to submit a form to apply to become a Service Provider.
  + FR2.2: The application shall be sent to an admin queue for review.
  + FR2.3: An administrator shall be able to approve or reject the application. Upon approval, the user's role shall be changed to "Service Provider."
* **FR3: Service Listing Management (For Service Providers)**
  + FR3.1: Service Providers shall be able to create new service listings.
  + FR3.2: A service listing must include: title, description, category, contact number, address, and operating hours.
  + FR3.3: Service Providers shall be able to upload images for their service.
  + FR3.4: Service Providers shall be able to view, update, and delete their own service listings.
* **FR4: Service Discovery (For Customers)**
  + FR4.1: Users shall be able to view a list of all services, categorized for easy browsing.
  + FR4.2: Users shall be able to search for services using keywords.
  + FR4.3: Users shall be able to filter services by:
    - Category
    - Distance from their current location
    - User Rating

**FR5: Ratings and Reviews**

* + FR5.1: Logged-in customers shall be able to submit a star rating (1-5) and a text review for a service they have used.
  + FR5.2: All reviews for a service shall be publicly visible on the service listing page.
  + FR5.3: The average star rating shall be calculated and displayed for each service.
* **FR6: Admin Functions**
  + FR6.1: The administrator shall have a web-based dashboard.
  + FR6.2: The administrator shall be able to manage all users and service listings (view, edit, delete).
  + FR6.3: The administrator shall manage the service categories available on the platform.

**Non-Functional Requirements (NFR)**

These define the quality attributes of the system.

* **NFR1: Performance:** The mobile application should load key data (e.g., list of services) within 3 seconds on a standard 4G connection.
* **NFR2: Usability:** The user interface should be intuitive and require minimal training. A new user should be able to register and find a service in under 2 minutes.
* **NFR3: Reliability:** The system should have an uptime of 99%, with critical functions like search and login always available.
* **NFR4: Security:** All user passwords must be hashed before being stored in the database. Communication between the client and server must be encrypted using HTTPS. JWT will be used for session management.
* **NFR5: Scalability:** The backend architecture should be capable of handling an initial load of 1,000 concurrent users without significant performance degradation.
* **NFR6: Compatibility:** The mobile application must be compatible with Android (version 6.0 and above) and iOS (version 12 and above).

## 4.4 Explanation of the Current System

There is no single "current system" but rather a fragmented collection of informal methods that residents of Sri Lanka use to find local services. This "system" can be described as follows:

1. **Word-of-Mouth:** Relying on recommendations from family, friends, and neighbours. This is the most trusted method but is slow, geographically limited, and depends heavily on one's social network.
2. **Social Media (Facebook/WhatsApp):** Users post queries in local community groups (e.g., "Galle Expats," "Colombo Community"). Service providers also post advertisements. This method is unstructured, making it difficult to search. Information quickly becomes buried, and there's no verification of the providers.
3. **Physical Search:** Literally walking or driving around a neighbourhood to find signboards for services like clinics or repair shops. This is time-consuming and inefficient.
4. **Generic Digital Tools (Google):** Using Google Search or Google Maps. While useful for established businesses with a formal online presence, it fails to capture a vast number of smaller, informal service providers who are not listed or have minimal information available.

## 4.5 Drawbacks of the Current System

* **Lack of Centralization:** Information is scattered across dozens of platforms and social networks.
* **Information Unreliability:** Contact numbers are often out of date, business hours are wrong, and locations are imprecise.
* **No Quality Assurance:** There is no systematic way to gauge the quality or reliability of a service provider before engaging them.
* **Inefficiency:** The process is manual, slow, and often results in a dead end.
* **Digital Exclusion:** It leaves behind a large segment of excellent service providers who are not tech-savvy enough to market themselves online.

## 4.6 Explanation of the Proposed System

The Locato system is proposed to directly address all the drawbacks of the current informal methods.

* **Centralization:** It provides a single, dedicated mobile application for all local service needs, becoming the go-to repository for this information.
* **Reliability:** By having an admin approval process and allowing service providers to manage their own listings, the information is more likely to be accurate and up to date. The user review system adds another layer of crowd-sourced validation.
* **Quality Assurance:** The ratings and review system introduces transparency and accountability, allowing users to make informed decisions based on the experiences of others.
* **Efficiency:** With powerful search, filtering, and map-based visualization, users can find exactly what they need in seconds, not hours.
* **Digital Inclusion:** Locato offers a simple, structured, and free way for any service provider, no matter how small, to establish a digital footprint and reach a wider audience.

In essence, the proposed system transforms a chaotic and unreliable process into a structured, efficient, and trustworthy digital experience.

# System Design

This chapter outlines the high-level and low-level design of the Locato system. System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. The design serves as a blueprint for the implementation phase. This chapter presents the user interface wireframes, architectural design, detailed UML diagrams, and the database design.

## 5.1 Design Overview

The design philosophy for Locato is cantered on simplicity, scalability, and user experience. A **Client-Server Architecture** has been adopted.

* **Client:** A cross-platform mobile application built with React Native. It is responsible for the user interface, capturing user input, and rendering data received from the server.
* **Server:** A backend application built with Node.js and Express.js. It handles business logic, processes requests from the client, interacts with the database, and manages user authentication.
* **Database:** A MongoDB NoSQL database hosted on MongoDB Atlas. It stores all persistent data, including user profiles, service listings, and reviews.

This separation of concerns (frontend from backend) allows for independent development and scaling. The use of a RESTful API for communication between the client and server ensures a standardized and stateless interaction model.

### 5.1.1 Wireframes

Wireframes are basic visual guides that represent the skeletal framework of the application. They are used to focus on the layout, functionality, and user flow without being distracted by colours or styles. Below are descriptions of key wireframes designed for Locato.

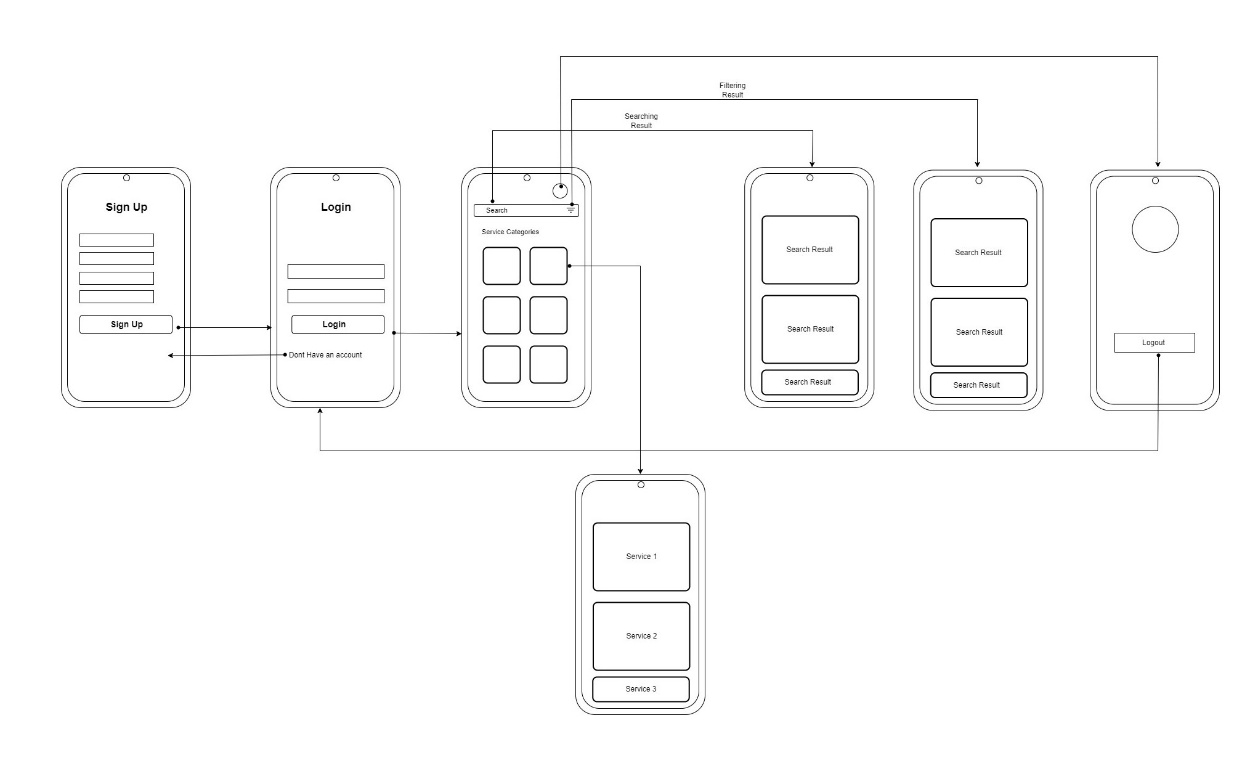


Figure 4: wire frame of the user view search, filter services

A group of black and white shapes

AI-generated content may be incorrect.

Figure 5: wireframe of the service owner applying for service

A diagram of a company

AI-generated content may be incorrect.

Figure 6: Wireframe of the admin management

## 5.2 UML Diagrams

Unified Modelling Language (UML) diagrams are used to visualize, specify, construct, and document the artifacts of a software-intensive system.

### 5.2.1 Use Case Diagram

The Use Case diagram models the system's functionality by illustrating interactions between users (actors) and the system.

#### 5.2.1.1 User Use Case Diagram

A diagram of a diagram

AI-generated content may be incorrect.

Figure 7: User use case diagram

**Components and Relationships**

* **Actor**: The stick figure labeled "**Users**" represents the primary actor interacting with the system. An actor is an external entity, such as a person, another system, or an organization, that uses the system to achieve a goal.
* **Use Cases**: These ovals describe the system's key functionalities:
  + **User Registration**: Allows a new user to create an account.
  + **User Login**: Allows a registered user to access the system.
  + **View Service**: Represents a general function for viewing services offered by the system.
  + **Search**: Allows the user to find information within the system.
  + **Filter**: Allows the user to refine or narrow down a set of results.
* **Associations**: The solid lines connecting the "**Users**" actor to the use cases (User Registration, User Login, View Service, Search, and Filter) are **associations**. They indicate that the "Users" actor initiates or participates in these use cases.

#### 5.2.1.2 Service Owner Use Case Diagram

A diagram of a diagram

AI-generated content may be incorrect.

Figure 8: Service owner use case diagram

Components and Relationships

* Actor: The stick figure labeled "Service Owner" represents the external entity that interacts with the system. This actor is responsible for tasks related to services, such as registration and listing.
* Use Cases: These ovals describe the key functions available to the Service Owner:
  + User Registration: Allows the Service Owner to create a new user account within the system.
  + User Login: Allows the Service Owner to access their account.
  + Register Service: A core function that allows the Service Owner to add a new service to the system.
  + List Service: Allows the Service Owner to view or manage the services they have registered.
  + Apply For Service: Represents the action of applying for a specific service.
* Associations: The solid lines connecting the "Service Owner" actor to the use cases (User Registration, User Login, Register Service, and List Service) are associations. They show that the "Service Owner" actor directly initiates these actions.

#### 5.2.1.3 Admin Use Case Diagram

A diagram of a person's life cycle

AI-generated content may be incorrect.

Figure 9: admin use case diagram

Actor: The stick figure labeled "Admin" represents the system administrator. This is an external entity that interacts with the system to perform specific administrative tasks.

Use Cases: These ovals represent the main functions or goals of the system from the admin’s perspective:

* Admin Login: Allows the administrator to authenticate and gain access to the system.
* Manage Services Applications: A core administrative function for handling applications related to services.
* Accept Application: A function to approve a service application.
* Reject Application: A function to deny a service application.
* Manage Users: A function for the administrator to manage user accounts.
* Associations: The solid lines connecting the "Admin" actor to the use cases (Admin Login, Manage Services Applications, and Manage Users) are associations. They indicate that the "Admin" actor initiates or interacts with these use cases directly.

<<include>> Relationships: The dashed lines with the <<include>> stereotype is including relationships. This is a specific type of dependency between use cases.

* The relationship from Manage Services Applications to Accept Application means that the "Accept Application" use case is a mandatory part of the "Manage Services Applications" process.
* Similarly, the relationship from Manage Services Applications to Reject Application means that the "Reject Application" use case is also a mandatory part of the "Manage Services Applications" process.

### 5.2.2 Sequence Diagram

Sequence diagrams show how objects interact in a time-ordered sequence.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 10: System Sequences diagram

The sequence diagram illustrates the interaction among Service Owner, User, Admin, the System, and the Database.

* **Service Owner** registers and logs in before submitting a service request. The system forwards this to the admin, who may approve or reject it. Upon approval, the Service Owner can manage services; otherwise, a rejection notification is sent.
* **User** registers and logs in to view approved services. They may also search or filter results, which the system processes through the database before displaying.
* **Admin** logs in, reviews pending service requests, and either approves or rejects them. Each decision updates the database and notifies the Service Owner.

The diagram highlights conditional flows (approval/rejection, search/filter) and iterations (reviewing multiple requests), showing how requirements are executed step by step within the system.

### 5.2.3 Class Diagram

The Class diagram describes the static structure of the system by showing its classes, attributes, operations (or methods), and the relationships between them.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 11: System class diagram

### 5.2.4 Context Diagram (DFD Level 0)

The Context Diagram provides a high-level overview of the entire system, showing it as a single process with its inputs and outputs from external entities.

A diagram of a computer flowchart

AI-generated content may be incorrect.

Figure 12: System DFD L0 diagram

The context diagram represents the Locato System as a single process and shows how it interacts with external entities and the database. It illustrates the flow of data between actors and the system at the highest level, without going into internal processes.

**External Entities and Data Flows**

1. **Service Owner**
   * Provides registration data, login credentials, and service request forms to the system.
   * Receives registration confirmation, login status, and approval/rejection notifications.
2. **User**
   * Submits registration data and login credentials.
   * Sends search or filter requests to the system.
   * Receives registration confirmation, login status, approved services, and search/filter results.
3. **Admin**
   * Submits login credentials to the system and receives login status.
   * Sends approval or rejection decisions for pending service requests.
   * Receives notifications regarding request handling.
4. **Database**
   * Stores and retrieves user data, service requests, and service statuses.
   * Acts as the system’s persistent data repository, supporting registration, authentication, and service management.

## 5.3 Introduction to Database Design

The choice of MongoDB, a NoSQL document database, was strategic. It is well-suited for the Locato application due to its:

* **Flexibility:** The schema-less nature allows for storing diverse service listings, where some services might have unique attributes (e.g., a doctor's specialization vs. a mechanic's available parts).
* **Scalability:** MongoDB is designed for horizontal scaling, which is ideal for a growing application.
* **Geospatial Capabilities:** MongoDB has excellent built-in support for geospatial queries (e.g., "find all services within a 5km radius"), which is fundamental to Locato's functionality.

### 5.3.1 Corrected Entity Relationship Diagram (ERD)

While ERDs are traditionally associated with relational databases, a conceptual ERD can still be used to model the relationships between collections in MongoDB.

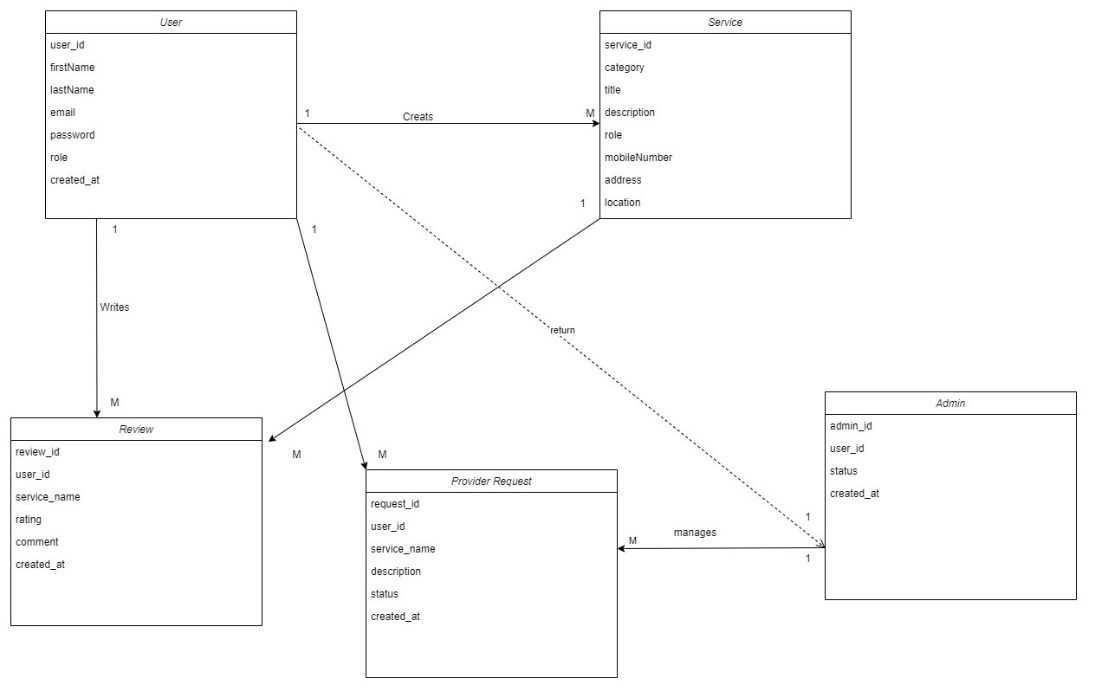


Figure 13: Satyam ER diagram

# Implementation

The implementation phase is where the architectural and design blueprints detailed in the preceding chapters are translated into a tangible, working software system. This chapter provides an overview of the development environment, the technologies employed, and the key technical aspects of constructing the Locato mobile application and its supporting backend. It showcases code structures and implementation logic for the most critical user interfaces and functionalities, bridging the gap between theoretical design and the final product.

## 6.1 Implementation Overview

The development of Locato followed the Agile methodology outlined in Chapter 3.5. The project was broken down into sprints, with each sprint focusing on delivering a specific set of features. This iterative approach allowed for continuous integration and testing, ensuring the project remained on track and adaptable to any challenges that arose.

**Development Environment:**

* **Code Editor:** Visual Studio Code was used as the primary integrated development environment (IDE) for both frontend and backend development, leveraging its powerful extensions for JavaScript, Node.js, and React.
* **Version Control:** Git was used for version control, with a remote repository hosted on GitHub. This practice was crucial for tracking changes, managing different feature branches, and maintaining a stable main branch. **Appendix 3 (Version Control)**
* **Backend Runtime:** Node.js (v18.x) was used as the server-side runtime environment.
* **Mobile Development Framework:** The Expo Go application was used for running and testing the React Native application on a physical Android device, facilitating rapid development and real-time debugging. **Appendix 4 (Expo Go)**
* **API Testing:** Postman was used extensively to test the backend RESTful API endpoints independently of the mobile client, ensuring that the business logic was correct and robust before frontend integration. **Appendix 5 ( API Testing )**
* **Database**: MongoDB was used to store all application data, including user information, locations, and other relevant content. As a document-based database, MongoDB stores data in a BSON (Binary JSON) format, allowing for a flexible schema that was easily adapted as new features were added throughout the development sprints.
* **Database Management**: Used MongoDB Atlas, a cloud-based database service, for hosting and managing our database instances. This choice streamlined the setup process, provided automated backups, and ensured high availability.
* **Driver/ODM**: The Mongoose ODM (Object Data Modelling) library was used to manage the relationship between data within the Node.js backend and the MongoDB database. Mongoose provided a straightforward, schema-based solution for application data, handling data validation, type casting, and creating database queries with ease. **Appendix 10 (Database )**

## 6.2 User Interfaces

The user interfaces were built using React Native, a JavaScript framework for building native mobile apps. The core principle was to create a clean, modern, and intuitive user experience.

* **UI Component Library:** Instead of a pre-built component library, custom components were styled using Native Wind, which brings the utility-first styling approach of Tailwind CSS to React Native. This allowed for rapid and consistent styling directly within the JSX, resulting in a highly maintainable and readable component structure.
* **Navigation:** **React Navigation** was used to handle all screen transitions and the overall navigation structure. A combination of a Tab Navigator for the main sections (Home, Map, Profile) and a Stack Navigator for drilling down into details (e.g., from a list of services to a specific service's page) was implemented. **Appendix 6 ( Stack Navigation )**
* **State Management:** For managing application-wide state, such as user authentication status, Zustand was employed. This provided a lightweight and effective way to share data between components without the complexity of a larger state management library like Redux, which was deemed unnecessary for the current scope of the project. **Appendix 7 ( State management library )**

### 6.2.1 User Input Interface Designs

This section describes the implementation of the key screens, which represent the primary points of interaction for the user. Placeholders for screenshots are included to illustrate the final design.

**1. Login and Registration Screens**

The authentication flow is the gateway to the application. The screens were designed to be simple and clear.

**Description***:* The Login Screen features the "Locato" logo, followed by two input fields for "Email" and "Password," and a prominent "Login" button. A text link below provides an option to navigate to the Registration screen. The Registration screen is similar, with an additional field for "Name" and a "Register" button. **Appendix 8 (Login and Signup )**

Implementation Snippet (Handling User Input): State for input fields was managed using the useState hook.

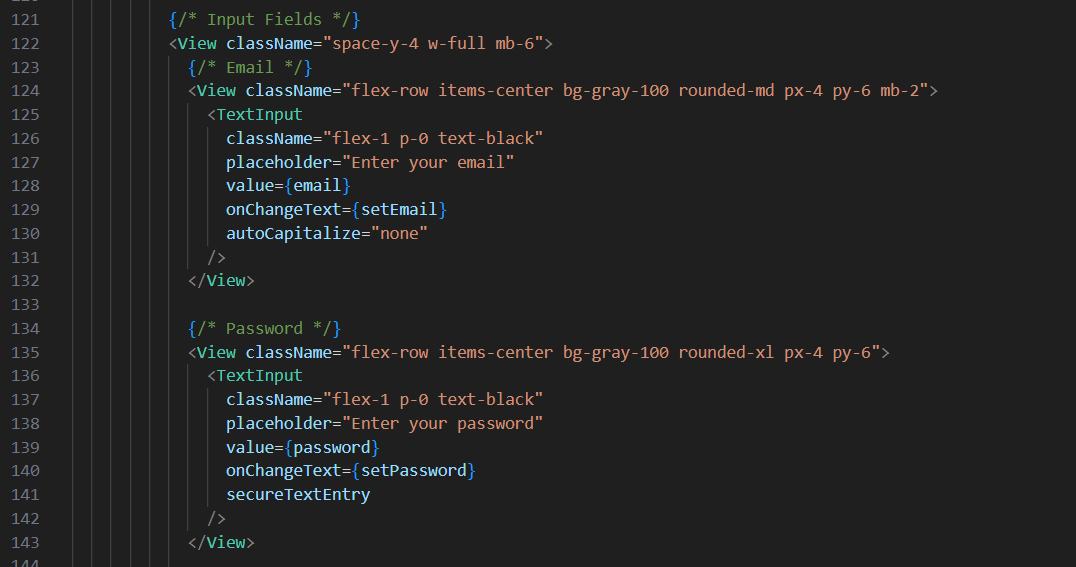


Figure 14: Login page input fields code

**A computer code on a black background

AI-generated content may be incorrect.**

Figure 15: Use state for handle input state

**2. Home Screen & Service Discovery**

This is the central hub for service discovery.

**Description***:* The Home Screen displays a welcome message, a search bar, and a grid of service categories represented by icons (e.g., a wrench for "Repairs," a graduation cap for "Tuition"). Tapping a category navigates the user to the Search Results screen, pre-filtered for that category. **Appendix 9 ( Home Screen fetching services)**

Implementation Snippet (Fetching Data from API): The useEffect hook was used to fetch service data from the backend API using an Axios instance when the component mounts.

A computer screen shot of a program code

AI-generated content may be incorrect.

Figure 16: Fetching service data from database (API call)

**3. Service Provider - Add Service Form** This is the key interface for service providers to contribute content to the platform.

Description: This screen presents a form with clearly labeled input fields for all required service information: Title, Description, Category (a dropdown menu), Phone Number, and Address. It also includes an option to upload images. A "Submit Service" button at the bottom sends the data to the backend for processing and admin approval.

Implementation Logic: The form state was managed with multiple useState hooks. On submission, the data is bundled into a Form Data object (to handle image uploads) and sent to the backend via a POST request. Client-side validation was implemented to ensure that all required fields are filled before submission.

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 17: Image upload handling

### 6.2.2 Service Owner’s User Interfaces

A screenshot of a phone

AI-generated content may be incorrect.A screenshot of a phone

AI-generated content may be incorrect.

Figure 18: Profile Page

Figure 19: Select category for applying

A screenshot of a phone

AI-generated content may be incorrect.

Figure 20: Service application

# Testing

Software testing is a systematic process of evaluating a software application to ensure it meets the specified requirements, is free of defects, and delivers a high-quality user experience. For the Locato project, a multi-level testing strategy was adopted to validate the functionality, reliability, and usability of both the backend API and the mobile application. This chapter details the testing overview, the test plan, and a selection of test cases executed.

## 7.1 Testing Overview

The testing strategy for Locato was designed to be comprehensive, covering different aspects of the system at various stages of development. The approach integrates seamlessly with the Agile development methodology, emphasizing continuous testing throughout the sprints.

The testing process was structured into three main levels:

1. **Unit Testing:** Focused on testing the smallest individual components (units) of the software in isolation. For the backend, this meant testing individual controller functions and middleware. For the frontend, it involved testing individual React Native components.
2. **Integration Testing:** This phase focused on testing the interaction between different components. The primary goal was to verify that the frontend mobile application could communicate correctly with the backend API—that requests were sent in the correct format and that responses were handled properly.
3. **System/User Acceptance Testing (UAT):** This involved testing the complete, integrated application to evaluate its compliance with the specified functional and non-functional requirements. This was performed manually by the developer, simulating real-world user scenarios from end to end.

### 7.1.1 Test Strategy

**Unit Testing**

Unit testing is crucial for ensuring the reliability of the codebase's foundational building blocks.

* **Backend (Node.js/Express.js):**
  1. **Frameworks:** Jest (a JavaScript testing framework) and Supertest (an HTTP assertion library) were used.
  2. **Approach:** Test suites were created for each major API route. Supertest was used to make mock HTTP requests to the API endpoints without needing to run a live server. Assertions were then made on the HTTP response status codes and the JSON body to verify the endpoint's behaviour. For example, a test for the POST /api/auth/register endpoint would check if it returned a 201 Created status for a valid registration and a 400 Bad Request for duplicate email addresses.
* **Frontend (React Native):**
  1. **Frameworks:** React Native Testing Library was used.
  2. **Approach:** Unit tests were written for reusable UI components like ServiceCard.js and CustomButton.js. The tests focused on rendering the component with mock data (props) and asserting that it displayed the correct information and responded to user interactions (like a button press) as expected. This ensures that changes to a component do not break its core functionality.

## 7.2 Test Plan

The test plan provides a structured document that outlines the scope, objectives, resources, and schedule of the testing activities.

* **1. Scope of Testing:**
  1. **In Scope:** All functional requirements listed in Chapter 4, including user authentication, service CRUD (Create, Read, Update, Delete) operations, search and filtering, map integration, and the review system. Key non-functional requirements like performance and security were also tested.
  2. **Out of Scope:** Usability testing with a large, external user group, and advanced performance load testing beyond a simulated user load.
* **2. Objectives:**
  1. To identify and rectify at least 95% of critical and high-priority bugs before deployment.
  2. To ensure all specified functional requirements are met.
  3. To validate that the application is stable and does not crash during normal usage patterns.
  4. To verify that the user interface is consistent and intuitive across the application.
* **3. Test Environment:**
  1. **Backend:** Local Node.js environment and deployed Render environment.
  2. **Frontend:** Expo Go on a physical Android device (Samsung M31, Android 12).
* **4. Pass/Fail Criteria:**
  1. **Pass:** The application is considered "passed" if there are no open critical or high-severity bugs. All major functionalities must work as expected.
  2. **Fail:** The application "fails" if any critical bug that prevents a user from completing a core workflow (e.g., registration, searching for a service) is present.

## 7.3 Test Cases

The following table presents a sample of the test cases designed and executed for key functionalities of the Locato application.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Id | Feature | Description | Pre-condition | Test-Steps | Expected Result | Actual Result | Status |
| T01 | User Login | Verify that a registered user can log in with valid credentials. | User account with email 'test@user.com' exists. | 1. Open the app. 2. Navigate to Login screen.  3. Enter customer@user.com' and the correct password.  4. Tap 'Login'. | The user is successfully authenticated and redirected to the Home Screen. | As Expected, | Pass |
| T02 | User Login | Verify that login fails with invalid credentials. | User account does not exist. | 1. Open the app.  2. Navigate to Login screen.  3. Enter 'wrong@user.com' and a password.  4. Tap 'Login'. | An error message "Invalid email or password" is displayed. The user remains on the Login screen. | As Expected, | Pass |
| T03 | Service Search | Verify that searching for "electrician" returns relevant services. | At least one service with "electrician" in its title/description exists. | 1. Log in.  2. On Home Screen, tap the search bar.  3. Type "electrician" and press search. | The Search Results screen displays a list of services, including the known "electrician" service. | As Expected, | Pass |
| T04 | Service Filter | Verify that filtering by category "Health" shows only health services. | Services from multiple categories, including "Health", exist. | 1. Log in.  2. Tap the "Health" category icon. | The results screen shows only services categorized under "Health." No services from other categories are visible. | As Expected, | Pass |
| T05 | Provider Approve | Verify that an admin can approve a service provider application. | A user has applied to become a provider. Admin is logged in to the web panel. | 1. Admin navigates to the "Provider Requests" section. 2. Finds the pending application.  3. Clicks 'Approve'. | The user's role in the database is updated to "provider." The user can now access the service management dashboard. | As Expected, | Pass |

This structured testing process was instrumental in ensuring the quality and stability of the Locato application, leading to a robust and reliable final product.

# Critical Evaluation & Conclusion

This final chapter synthesizes the entire research project, providing a summary of the work undertaken, a critical evaluation of the project's outcomes against its initial objectives, and a reflection on the lessons learned throughout the development lifecycle. It also discusses the inherent limitations of the system and proposes avenues for future work, concluding with a final statement on the project's significance and contributions.

## 8.1 Summary of the Project

This project set out to address a significant real-world problem in Sri Lanka: the lack of a centralized and reliable platform for discovering local services. The outcome of this endeavour is **Locato**, a fully functional, cross-platform mobile application. Built using the MERN stack and React Native, Locato serves as a comprehensive service directory that connects consumers with local service providers across the country. The system provides an intuitive user interface for service discovery, enhanced with advanced filtering, location-based map views powered by OpenStreetMap, and a user-driven ratings and reviews system to foster trust. A key feature is the administrative backend that empowers service providers to manage their own digital presence after a verification process, thereby promoting digital inclusion for small businesses. The project successfully navigated the entire software development lifecycle, from planning and requirement analysis to design, implementation, and rigorous testing, culminating in a deployed proof-of-concept that is both scalable and impactful.

**Achievement of Objectives**

A critical measure of success is the extent to which the project met the objectives defined in Chapter 1.

1. **Develop a centralized, categorized directory:** **Achieved & Exceeded.** A cross-platform mobile application was successfully developed. It not only provides a centralized directory but does so with a scalable architecture that can support a nationwide user base, exceeding the initial concept which was focused on a smaller region.
2. **Implement advanced search and filtering:** **Achieved.** The application features a robust filtering system allowing users to narrow down results by category, keyword, distance, and user rating. This functionality is core to the user experience and works efficiently.
3. **Integrate a location-aware mapping feature:** **Achieved.** OpenStreetMap was successfully integrated using react-native-maps, providing users with an interactive map to visualize service locations relative to their own, a critical feature for a location-based service.
4. **Create a secure, role-based system:** **Achieved.** The system implements secure JWT-based authentication and a clear role-based access control model for Customers, Service Providers, and Administrators. The workflow for a user to become a provider via admin approval was successfully implemented.
5. **Implement a user-driven ratings and review system:** **Achieved.** A fully functional module for submitting and viewing ratings and reviews was developed. This feature is vital for building a community of trust on the platform.
6. **Develop a comprehensive administrative dashboard:** **Achieved.** A web-based admin panel was created, allowing for the crucial moderation of service provider applications. This ensures the integrity and quality of the listings on the platform, a key differentiator from un-moderated platforms.

The project successfully met all its primary objectives, delivering a solution that is not just a functional prototype but a well-architected foundation for a real-world service.

# Lessons Learned & Critical Evaluation

Reflecting on the project provides valuable insights into both its successes and the challenges encountered.

**Successes:**

* **Technology Stack Efficacy:** The choice of the MERN stack with React Native proved to be highly effective. The use of JavaScript across the entire stack streamlined development and React Native's cross-platform capabilities saved significant time and effort. The choice of OpenStreetMap was validated as a cost-effective and flexible mapping solution.
* **Agile Methodology:** Adopting an Agile approach, even for a solo developer, was beneficial. It forced a disciplined, iterative process, ensuring that the most critical features were developed and tested first, which helped in managing the project timeline effectively.
* **Problem-Solution Fit:** The positive feedback gathered during the initial requirement analysis and the final working product demonstrates a strong fit between the problem identified and the solution developed.

**Challenges and Limitations:**

* **Initial Data Population (The "Cold Start" Problem):** A major limitation of any platform like Locato is that its value is proportional to the number of listings it contains. At launch, the database is sparse. Overcoming this requires a significant marketing and onboarding effort, which is outside the scope of this academic project.
* **Scalability of Manual Admin Approval:** While the admin approval process is crucial for quality control, it is a manual bottleneck. As the platform grows, this process would become unmanageable for a single administrator. Future iterations would need to explore semi-automated verification processes.
* **Dependency on Internet Connectivity:** The application is entirely online. In areas of Sri Lanka with poor or intermittent internet connectivity, the user experience would be severely degraded. An offline capability, even a limited one, was not implemented and remains a key limitation.
* **No AI Implementation:** As noted in the project's scope, the intelligent recommendation feature discussed in the literature review (Yan, 2017) was not implemented. The current filtering is powerful but user-driven; a proactive recommendation engine would significantly enhance user engagement but required more time and data than was available.

# Future Implementation Plans

Locato has been built as a strong foundation, but there are numerous avenues for future enhancement:

1. **AI-Powered Recommendations:** Implement a machine learning model to provide personalized service recommendations to users based on their search history, location patterns, and saved services.
2. **In-App Booking and Payments:** Integrate a secure booking and payment gateway. This would complete the user journey within the app, add a new layer of convenience, and create a potential revenue stream through commissions.
3. **Real-Time Chat:** Add a real-time messaging feature to allow direct communication between customers and service providers within the app.
4. **Advanced Provider Analytics:** Enhance the service provider dashboard with analytics on listing views, user engagement, and customer demographics to help them optimize their services.
5. **Localization:** Add support for Sinhala and Tamil languages to make the app more accessible to the entire Sri Lankan population.

# Conclusion

The digital divide in local service discovery presents a tangible obstacle to both consumers and small businesses in Sri Lanka. This research project successfully demonstrated that this gap can be bridged through the thoughtful application of modern mobile technology. Locato stands as a robust proof-of-concept, a scalable and user-centric platform that streamlines the process of finding reliable local services. By achieving all of its core objectives, the project not only delivered a high-quality software artifact but also contributed a practical blueprint for developing similar socio-technical solutions in emerging economies. While limitations exist, the platform's solid architectural foundation and clear potential for future expansion position it as a significant step towards a more connected and digitally empowered local economy in Sri Lanka.

# Appendix

### Appendix 1 (WBS)

A yellow and grey striped pattern

AI-generated content may be incorrect.A screenshot of a document

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Figure 21: WBS

<https://docs.google.com/spreadsheets/d/1QTrAonH_qB7tHj5Rga68zb_s7QgXxwSJwNm9jvVkDTc/edit?usp=sharing>

### Appendix 2 (Google form)

<https://docs.google.com/forms/d/e/1FAIpQLScXsrJu6UhMe6LotHQcCUwGYY0pYvF9f-JiXJMLaTqacG17Ug/viewform?usp=sharing&ouid=107598932784379665931>

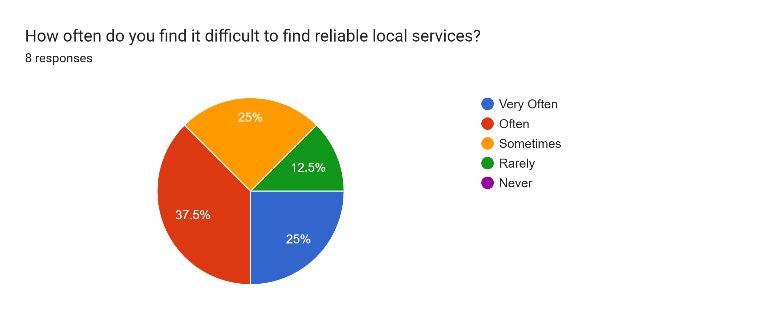
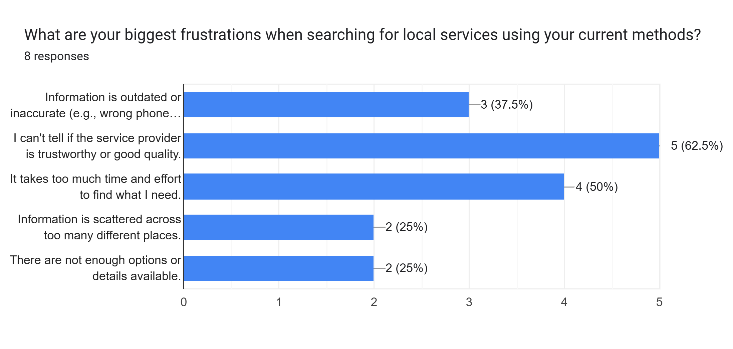


Figure 22: Result of Survey

### Appendix 3 (Version Control)

A screenshot of a computer

AI-generated content may be incorrect.

Figure 23: GitHub frontend repository

A screenshot of a computer

AI-generated content may be incorrect.

Figure 24: GitHub backend repository

### Appendix 4 (Expo Go)

A screenshot of a phone

AI-generated content may be incorrect.

Figure 25: Expo Go application

### Appendix 5 ( API Testing )

A screenshot of a computer

AI-generated content may be incorrect.

Figure 26: API testing

### Appendix 6 ( Stack Navigation )

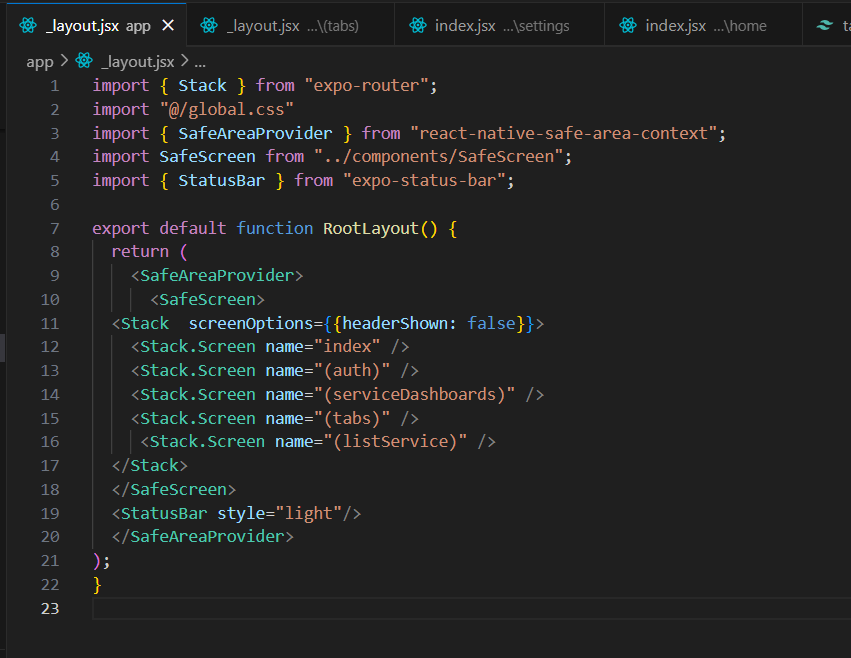


Figure 27: Stack Navigations

### Appendix 7 ( State management library )

A screen shot of a computer

AI-generated content may be incorrect.

Figure 28: Global state management (Zustand)

### A screenshot of a login form AI-generated content may be incorrect.Appendix 8 (Login and Signup )

Figure 29: Signup screen UI

A screenshot of a login form

AI-generated content may be incorrect.

Figure 30: Login screen UI

### Appendix 9 ( Home Screen fetching services)

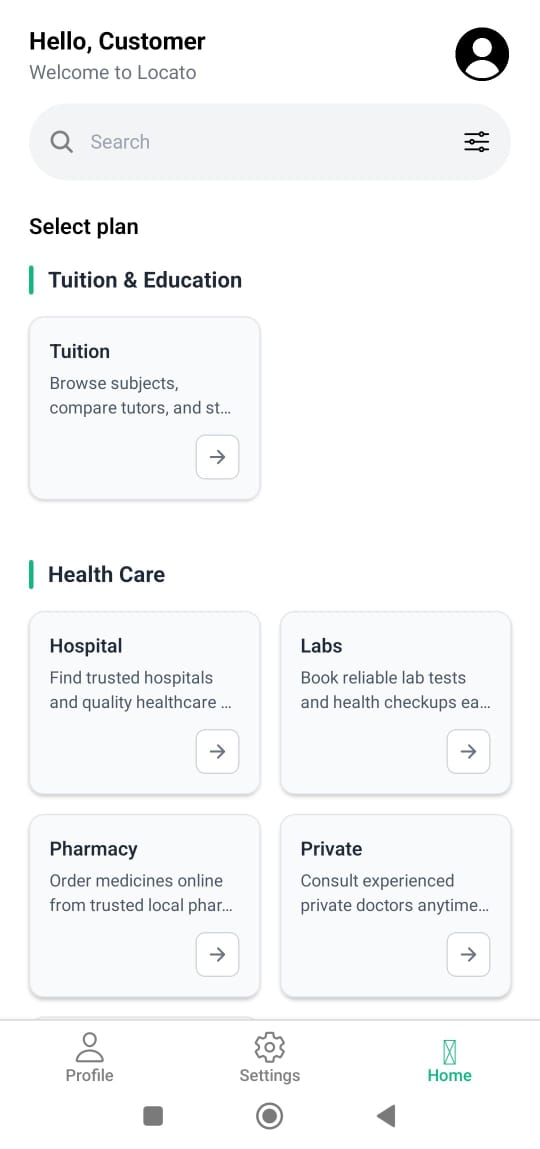


Figure 31: Home screen with fetching services

### Appendix 10 (Database )

A screenshot of a computer

AI-generated content may be incorrect.

Figure 32: MongoDB Database

# References

Agre, P. E., 2001. Changing places: Contexts of awareness in computing. *Human-Computer Interaction,* 16(2-4), pp. 177-192.

Alsharif, M. &. N. M., 2019. Mobile applications in developing countries: Opportunities and challenges. *Journal of Information Technology in Developing Economies,* 12(3), pp. 45-60.

Bank, W., 2023. *Digital economy for South Asia: Sri Lanka country diagnostic,* Washington, DC: World Bank Group.

Dey, A. K. A. G. D. &. S. D., 2010. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction,* 16(2-4), pp. 97-106.

Dey, A. K. A. G. D. &. S. D., 2010. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction,* 16(2-4), pp. 97-166.

Gupta, S. &. B. A., 2020. Role of online reviews in consumer decision-making. *Journal of Consumer Behaviour,* 19(4), pp. 356-370.

Kar, A. &. G. S., 2022. Comparative analysis of mapping APIs: Google Maps vs. OpenStreetMap. *International Journal of Geographic Information Systems,* 28(5), pp. 1023-1049.

Kumar, V. &. S. R., 2021. Cross-platform mobile app development: Benefits and challenges. *Journal of Software Engineering,* 15(2), pp. 210-225.

Li, X. Z. Y. &. W. L., 2016. Crowd-sourced mobile service platforms: Opportunities and challenges. *IEEE Transactions on Mobile Computing,* 15(8), pp. 1945-1958.

Perera, K., 2021. *Digital gaps in Sri Lankan service sectors.* colombo: University of Colombo Press.

Yan, B. &. C. H., 2017. Personalized location-based service recommendation. *IEEE Transactions on Services Computing,* 10(3), pp. 410-423.

Yan, B. &. C. H., 2017. Personalized location-based service recommendation. *IEEE Transactions on Services Computing,* 10(3), pp. 410-423.

Yan, B. &. C. H., 2017. Personalized location-based service recommendation. *IEEE Transactions on Services Computing,* 10(3), pp. 410-423.

Yan, B. &. C. H., 2017. Personalized location-based service recommendation. *IEEE Transactions on Services Computing,* 100(3), pp. 410-423.

Zhuang, Z. K. K. &. S. V. P., 2015. Location-based social applications: A survey. *ACM Computing Surveys,* 47(4).

# Project Log Sheet – Supervisory Sessions for CIS6035

Meeting No. 1

A document with text and images

AI-generated content may be incorrect.

Meeting No. 2

A document with text and a note

AI-generated content may be incorrect.

Meeting No. 3

A document with text and a blue and white logo

AI-generated content may be incorrect.

Meeting No 4.

A document with text on it

AI-generated content may be incorrect.

Meeting no 5

A document with text and a blue logo

AI-generated content may be incorrect.