Gamata Seva - ගමට සේවා

A Web-Based Platform Connecting Sri Lankan Micro-Entrepreneurs and Informal Workers with Local Customers

PROJECT REPORT

Individual Assignment - Software Engineering (CS2103)

Name: **V.J.K Rodrigo**

Index No: D/DBA/24/0024

Table of Contents

INTRODUCTION

In Sri Lanka, especially in rural and semi-urban areas, people struggle to access local services. Skilled workers like carpenters, masons, electricians, mechanics, tuition teachers, and household helpers often have a hard time reaching potential customers. They mainly rely on personal networks, word-of-mouth, or notice boards in villages. This lack of visibility stops them from fully using their skills and boosting their income.

Customers also have trouble finding reliable service providers. Without a central platform, they waste a lot of time asking neighbors, relatives, or friends for recommendations. This process is slow, often unreliable, and doesn't assure quality service.

Urban areas in Sri Lanka benefit from digital platforms like PickMe, Uber, and UrbanClap (now Urban Company). These services connect customers with various providers quickly and efficiently. However, these solutions target cities and rarely reach rural and village-level communities. As a result, rural populations miss out on the benefits of modern digital marketplaces.

To fill this gap, we propose Gamata Seva, a web-based and mobile-friendly platform that serves as a digital marketplace for rural services. This platform will give service providers more visibility and let customers search, book, and pay for services more easily. Gamata Seva will be designed to be simple, low-cost, and user-friendly, even for those with limited internet access.

This project report outlines the requirements and design aspects of the Gamata Seva platform. It includes the problem definition, objectives, scope, requirement engineering, system modeling, and architectural design.

SELECTING A SOFTWARE PROCESS MODEL

Developing a software system involves choosing the right software process model. This model guides the project through requirements, design, implementation, and testing. For the Gamata Seva project, we have selected a combined **Agile** and **Reuse-Oriented development** model.

Agile focuses on iterative development. The system is built in small increments and improved through constant feedback from stakeholders.

Advantages for Gamata Seva:

- It is flexible in response to changes in requirements. This flexibility is important for gathering feedback from rural users.
- It enables early prototyping and user testing, ensuring usability and simplicity.
- It allows for the gradual addition of features such as reviews, ratings, and payment methods.

Reuse-oriented development focuses on using existing software components to cut down on development time and costs.

Examples relevant to Gamata Seva:

- Payment modules (Mobile wallets, Bank APIs).
- Mapping services (Google Maps API) for service location tracking.
- Authentication modules for secure login and registration.

Advantages for Gamata Seva:

- Faster development and lower cost.
- Reliable and secure proven components.

Justification for Model Selection.

- A pure **Waterfall model** is not a good fit because it is inflexible and does not easily accommodate changes after the requirements are set.
- Using only **incremental models** could slow down the integration of existing components.
- The **Combined Agile and Reuse-Oriented model** offers both flexibility and efficiency. This approach suits the rural-user focus and low-cost nature of Gamata Seva.

By following this combined model, the project ensures rapid development, early user feedback, and cost-effective implementation while maintaining the quality and reliability of reused components.

REQUIREMENT ENGINEERING

Requirement Engineering is the process of identifying, analyzing, documenting, and validating the needs of the system stakeholders. For Gamata Seva, this makes sure that **service providers**, **customers**, and the **admin** can use the system effectively.

How we gathered requirements

To ensure that the system addresses real user needs, we applied multiple requirement elicitation techniques, with **interviews** being our primary method:

1. **Interviews**

   We conducted interviews with service providers and customers.

   Key Insights from Service Providers:
   - They struggle to attract regular customers because they lack marketing.
   - They need a simple platform since many are not tech-savvy.
   - They prefer a system that handles payments securely.

   Key Insights from Customers:
   - Customers find it hard to quickly locate skilled providers, particularly during emergencies.
   - They worry about trusting and relying on service providers they do not know.
   - There is a strong preference for mobile-based platforms, as smartphones are widely used in villages.

Stakeholder	**Problem Identified**	**Need Expressed**
Carpenter (Provider)	No steady customer flow	Wants visibility to reach more customers
Barber (Provider)	Difficulty reaching new customers	Wants a platform to attract local clients
Customer A	Cannot find reliable service providers quickly	Wants an easy way to search and book services
Customer B	Difficulty trusting unknown providers	Wants verified/approved service providers
Customer C	Prefers using smartphones over desktop	Wants a mobile-friendly platform

*The interviews were conducted in **Sinhala**, as our target participants were from rural areas. This ensured they could answer comfortably in their own language. The full questionnaire (translated into English) is provided in following link.*

- *https://tinyurl.com/2k79ewbb*

2. **Observation:**

   We studied how villagers currently connect with service providers — mostly through word-of-mouth, community notice boards, or personal recommendations.

3. **Brainstorming & Group Discussions:**

   Within the project team, we analyzed interview insights, discussed possible features, and refined the scope of the platform.

4. **Reference Study:**

   We reviewed existing platforms like PickMe, Uber, and UrbanClap. While these work well in urban areas, they are often too complex or data-heavy for rural contexts. Features were adapted into a simpler, low-cost, mobile-friendly model.

These findings directly informed the system's functional, non-functional, and domain requirements.

Functional Requirements

Actor	**Functions**
Customer	Login/Register Search for services View provider profiles Book services & make payments Rate & review providers
Service Provider	Login/Register List services Update profile details Manage bookings & Receive payments
Admin	Manage customers Manage service providers Verify providers Manage service categories Moderate reviews & complaints View reports & analytics

Non-Functional Requirements

- **Performance** – Service search should load within 3 seconds on 3G/4G.
- **Scalability** – Must support thousands of users across different rural regions.
- **Security** – Secure login, encrypted payments, and safe data storage.
- **Usability** – Simple design suitable for first-time smartphone users.
- **Compatibility** – Accessible on both mobile and desktop browsers.

Domain Requirements

- Must include common rural services (carpentry, tuition, plumbing, tailoring, etc.).
- Must support Sinhala/Tamil/English for wider adoption.
- Must integrate with local mobile payment systems (eZ Cash, mCash) and card payments.

After gathering requirements, the main **stakeholders** identified for the Gamata Seva are:

- **Customers**: People in rural areas who need reliable local services.
- **Service Providers**: Skilled individuals, such as carpenters, barbers, and electricians, looking for visibility and more clients.
- **Admin**: Platform managers who verify providers, manage service categories, moderate reviews, and generate reports.
- **Government / Investors (optional):** Entities interested in supporting rural development and local businesses.

These stakeholders are crucial for defining all functional, non-functional, and domain requirements of the system.

Based on the gathered requirements and identified stakeholders, the next step is to model the system interactions using **Use Case Diagrams**, which visually represent how customers, service providers, and admins will interact with the Gamata Seva platform.

USE CASE MODELING

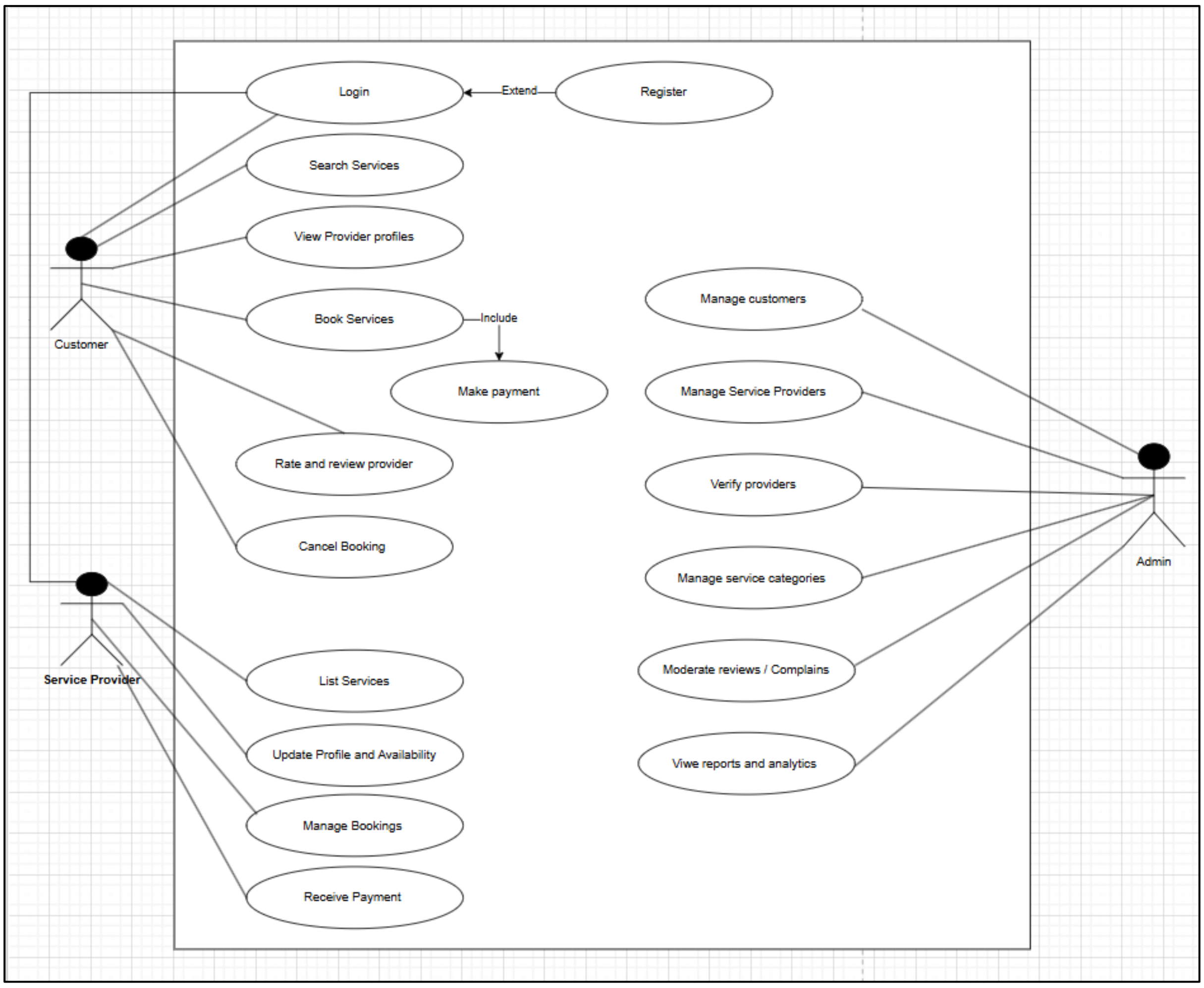
Use Case Modeling shows how the system interacts with its users and other stakeholders. It helps us understand what the system should do based on user needs. This process ensures that we capture all functional requirements.

For **Gamata Seva**, the main actors are:

- **Customer**
- **Service Provider**
- **Admin**

Use Case Diagram

The following Use Case Diagram for Gamata Seva has been created using Draw.io , visually representing the interactions between the system's actors (customers, service providers, and admin) and their respective use cases.



Use Case Specification Tables

Field	Details
Number	001
Name	Login
Summary	Allows users to login to the system.
Priority	5
Pre-Conditions	The user has registered in the system.
Post-Conditions	User enters the web app Home page.
Primary Actors	Customers & Service Providers
Secondary Actors	–
Triggers	User decides to login to the system.
Main Scenario / Actions	1. The actor selects **Register** or **Login**. 2. If **Register** → enters name, email, mobile, password, role (customer/provider). 3. If **Login** → enters email/username and password. 4. System validates credentials. 5. Users are granted access to the Home page.
Extensions	**2a.** During Registration (step 2), if username/email already exists → system displays "Already Exists". **3a.** During Login (step 3), if username/email is invalid → system prompts user to re-enter. **3b.** During Login (step 3), if password is invalid → system prompts user to re-enter. **3c.** During Login (step 3), if user has forgotten password → system provides option to reset password.
Open Issues	1. Should users confirm email/phone before first login? 2. Decide recovery method (email, SMS, or both).

Field	Details
Number	002
Name	Search Service
Summary	Allows users to choose a required service.
Priority	5
Pre-Conditions	Customer logged in.
Post-Conditions	Display the available services.
Primary Actors	Customers
Secondary Actors	Service Providers
Triggers	Customer clicks on "Search" to find a service.
Main Scenario / Actions	1. Customer opens the **Search Services** option. 2. System displays a search bar & filter options (category, location, price). 3. Customer enters keywords or selects filters. 4. System queries the database of service providers. 5. System displays a list of matching service providers with details. 6. Customer views results and may select one for booking.

Extensions	**3a.** Customer enters no keywords → system displays all available services by default. **4a.** No matching providers found → system displays message: “No services found”. **4b.** System/database error occurs → system shows error: “Unable to fetch services, please try again later.”
Open Issues	1. Should search support voice input or only text? 2. How much detail should be shown in search results (basic info vs. full provider profile)? 3. How will ranking of results be handled? (price low → high, ratings, sponsored services)

Field	**Details**
Number	003
Name	Book a Service
Summary	Customer books a service from a provider.
Priority	5
Pre-Conditions	Customer has searched for a service.
Post-Conditions	Booking record created; confirmation sent.
Primary Actors	Customers
Secondary Actors	Service Providers
Triggers	Customer selects a service and chooses "Book".
Main Scenario / Actions	1. Customer searches and selects a service provider from search results. 2. Customer chooses a date and time for the service. 3. Customer clicks “Book Now”. 4. System checks provider availability for the selected slot. 5. If available, system creates a booking record in the database. 6. System sends booking confirmation to customer (SMS/email/app notification). 7. Booking status is now “Confirmed” and provider is notified.
Extensions	**4a.** Selected slot unavailable → system suggests alternative dates/times. **5a.** Database error occurs during booking → system shows error: “Booking failed, please try again later.”
Open Issues	1. Maximum number of bookings per customer or per time slot? 2. Should cancellations be allowed anytime, or only before a cutoff?

Field	**Details**
Number	004
Name	View Provider Profiles
Summary	Allows a customer to view detailed profiles of service providers.
Priority	4
Pre-Conditions	Service providers must have active profiles in the system.
Post-Conditions	Customer successfully views provider details.
Primary Actors	Customers
Secondary Actors	Service Providers
Triggers	Customer selects a provider from the search results or service list.

Main Scenario / Actions	1. Customer navigates to provider search or service listing. 2. Customer selects a specific provider. 3. System displays provider profile with details (services, availability, pricing, reviews). 4. Customer browses through the information.
Extensions	**2a.** Selected provider profile not found → system displays error: "Profile not available." **3a.** Some details unavailable → system displays partial profile with a note: "Some information is currently unavailable."
Open Issues	1. Should provider contact details (phone/email) be visible or hidden until booking? 2. Should profile viewing history be tracked for personalization?

Field	**Details**
Number	005
Name	Rate and Review Provider
Summary	Allows a customer to provide feedback on a service provider after completing a booking by giving a star rating and writing a review.
Priority	4
Pre-Conditions	Customer must have completed at least one booking with the provider.
Post-Conditions	Updated average rating is reflected on the provider's profile.
Primary Actors	Customers
Secondary Actors	Service Provider (receives review and rating), Admin (moderates reviews)
Triggers	Customer chooses the option "Rate/Review Provider" after a booking is marked completed.
Main Scenario / Actions	1. Customer navigates to past bookings. 2. Customer selects a completed booking. 3. System prompts for a rating (1–5 stars) and optional written review. 4. Customer submits the rating/review. 5. System saves the review and updates the provider's profile.
Extensions	**3a.** Customer skips review → booking marked completed without feedback. **4a.** Customer submits inappropriate content → flagged for admin moderation.
Open Issues	1. Should customers be allowed to edit/delete reviews later? 2. Should anonymous reviews be supported? 3. Should providers be able to reply to reviews?

Field	**Details**
Number	006
Name	Cancel Booking
Summary	Allows a customer to cancel a service booking before the scheduled time, based on the platform's cancellation policy.
Priority	5
Pre-Conditions	Customer has an active booking that is not yet completed.
Post-Conditions	Booking status updated to "Cancelled."
Primary Actors	Customers
Secondary Actors	Service Provider
Triggers	Customer selects "Cancel Booking" option from their bookings list.

Main Scenario / Actions	1. Customer logs into the system. 2. Customer goes to “My Bookings.” 3. Customer selects an active booking. 4. System shows cancellation policy and refund eligibility. 5. Customer confirms cancellation. 6. System updates booking status and notifies the provider. 7. Refund is processed if applicable.
Extensions	**5a.** Cancellation after deadline (step 5) → system denies cancellation request. **6a.** Service provider cancels booking (step 6) → customer notified, refund processed automatically. **6b.** Dispute arises → admin intervention required.
Open Issues	1. Should partial refunds be allowed for late cancellations? 2. Should customers be allowed to reschedule instead of cancel? 3. Should providers be able to charge a cancellation fee?

Field	**Details**
Number	007
Name	List Services
Summary	Allows a service provider to add and manage the list of services they offer.
Priority	5
Pre-Conditions	Service provider must be registered and logged in.
Post-Conditions	Customers can view listed services.
Primary Actors	Service Provider
Secondary Actors	Customer
Triggers	Service provider selects “Add Service” or “Manage Services” option.
Main Scenario / Actions	1. Provider logs into the system. 2. Provider navigates to “My Services.” 3. Provider selects “Add New Service.” 4. System displays service details form. 5. Provider enters details (service type, description, price, location, availability). 6. Provider submits service listing. 7. System saves details and updates marketplace.
Extensions	**5a.** Missing/invalid input (step 5) → system requests correction. **6a.** Admin rejects service if it violates policies (step 6).
Open Issues	1. Should all new services require admin approval before going live?2. Should providers be able to upload service-related documents/licenses?3. Should providers be able to mark “seasonal” or “one-time” services?

Field	**Details**
Number	008
Name	Update Service Availability
Summary	Allows a service provider to update the availability status of their listed services.
Priority	4
Pre-Conditions	At least one service must already be listed.
Post-Conditions	Updated availability is reflected in the system.
Primary Actors	Service Provider
Secondary Actors	Customer
Triggers	Provider selects the "Update Availability" option from their service dashboard.
Main Scenario / Actions	1. Provider logs into the system. 2. Provider navigates to "My Services." 3. Provider selects a specific service. 4. Provider updates availability (time slots, days, or overall status).5. Provider submits changes. 6. System saves and updates service availability for customer view.
Extensions	**4a.** Provider cancels availability for specific dates (holidays, emergencies). **4b.** If provider forgets to update availability → system may auto-mark as "Unavailable" after inactivity.
Open Issues	1. Should availability sync with external calendars (e.g., Google Calendar)? 2. Should customers be notified when availability changes for a service they bookmarked?

Field	**Details**
Number	009
Name	Manage Bookings
Summary	Allows a service provider to view, accept, reject, or mark customer bookings as completed.
Priority	4
Pre-Conditions	Service provider has at least one active booking.
Post-Conditions	Booking statuses are updated (Accepted, Rejected, Completed).
Primary Actors	Service Provider
Secondary Actors	Customer
Triggers	Provider selects "Manage Bookings" from their dashboard.
Main Scenario / Actions	1. Provider logs into the system. 2. Provider navigates to "My Bookings." 3. System displays all pending and active bookings. 4. Provider selects a booking and chooses an action: Accept, Reject, or Mark Completed. 5. System updates booking status and notifies the customer.
Extensions	**4a.** Customer cancels before provider acts → booking marked Cancelled. **4b.** Provider tries to accept an already cancelled or completed booking → system shows error.
Open Issues	1. Should providers be allowed to partially accept bookings? 2. Should system allow accept/reject of multiple bookings at once?

Field	Details
Number	010
Name	Receive Payments
Summary	Allows a service provider to receive payments from customers for completed bookings via integrated payment systems.
Priority	5
Pre-Conditions	Customer booking must be marked as completed or ready for payment.
Post-Conditions	Payment is recorded in the system; booking status updated to "Paid."
Primary Actors	Service Provider
Secondary Actors	Customer
Triggers	Provider checks "Pending Payments" in dashboard.
Main Scenario / Actions	1. Provider logs into the system. 2. Provider navigates to "Pending Payments." 3. System displays all bookings pending payment. 4. Provider confirms receipt of payment or system updates automatically via gateway. 5. System updates booking status to "Paid" and adds transaction to provider account. 6. Notification sent to provider and customer.
Extensions	**4a.** Payment fails (insufficient balance, network error) → system prompts retry. **4b.** Refund or dispute requested → triggers admin intervention.
Open Issues	1. Should providers get instant transfer or scheduled batch payout? 2. Should system support automatic refunds for cancellations?

Field	Details
Number	011
Name	Manage Customers
Summary	Allows the admin to view, edit, suspend, or remove customer accounts to ensure platform integrity and compliance.
Priority	4
Pre-Conditions	Customer accounts exist in the system.
Post-Conditions	Customer account status is updated (active, suspended, deleted).
Primary Actors	Admin
Secondary Actors	Customer
Triggers	Admin selects "Manage Customers" from the admin dashboard.
Main Scenario / Actions	1. Admin logs into the system. 2. Admin navigates to "Manage Customers." 3. System displays a list of all customers. 4. Admin selects a customer and chooses an action: Edit, Suspend, or Delete. 5. System updates account status and logs the action.
Extensions	**4a.** Admin attempts to delete a customer with active bookings → system prevents deletion. **4b.** Admin suspends a customer → customer cannot book services; system notifies them.
Open Issues	1. Should deleted accounts be archived for future audits?

Field	Details
Number	012
Name	Manage Service Providers
Summary	Allows the admin to view, edit, suspend, or remove service provider accounts to maintain platform quality and compliance.
Priority	4
Pre-Conditions	Service provider accounts exist in the system.
Post-Conditions	Service provider account status is updated (active, suspended, deleted).
Primary Actors	Admin
Secondary Actors	Service Provider
Triggers	Admin selects "Manage Service Providers" from the admin dashboard.
Main Scenario / Actions	1. Admin logs into the system. 2. Admin navigates to "Manage Service Providers." 3. System displays a list of all providers. 4. Admin selects a provider and chooses an action: Edit, Suspend, or Delete. 5. System updates the account and logs the action.
Extensions	**4a.** Admin attempts to delete a provider with active bookings → system prevents deletion. **4b.** Admin suspends a provider → their services become unavailable to customers.
Open Issues	1. How to handle providers with pending payments or active bookings? 2. Should deleted accounts be archived for audits?

Field	Details
Number	013
Name	Verify Providers
Summary	Allows the admin to verify the authenticity and credentials of service providers before their services go live on the platform.
Priority	5
Pre-Conditions	Service provider has submitted required documents or information.
Post-Conditions	Provider account marked as verified or rejected.
Primary Actors	Admin
Secondary Actors	Service Provider
Triggers	Provider completes registration and submits documents for verification.
Main Scenario / Actions	1. Admin logs into the system.2. Admin navigates to "Verify Providers."3. System displays a list of providers pending verification.4. Admin reviews submitted documents and information.5. Admin approves or rejects verification.6. System updates provider status and notifies the provider.
Extensions	**4a.** Documents are incomplete → system requests additional info from provider. **5a.** Verification rejected → provider can resubmit after correction.
Open Issues	1. What exact documents are required for verification? 2. Should verification be automatic for certain trusted providers?

Field	**Details**
Number	014
Name	Manage Service Categories
Summary	Allows the admin to create, edit, or delete service categories to organize the marketplace and make it easier for customers to search services.
Priority	4
Pre-Conditions	At least one service category exists in the system.
Post-Conditions	Service categories are updated in the system.
Primary Actors	Admin
Secondary Actors	Service Provider
Triggers	Admin selects “Manage Service Categories” from the dashboard.
Main Scenario / Actions	1. Admin logs into the system. 2. Admin navigates to “Service Categories.” 3. System displays all existing categories. 4. Admin adds, edits, or deletes a category. 5. System updates categories and associated services.
Extensions	**4a.** Admin tries to delete a category with assigned services → system prevents deletion or asks for reassignment. **4b.** New category requires approval → system flags it for review.
Open Issues	1. Should categories have sub-categories for better granularity? 2. Should providers suggest new categories or admin creates them?

Field	**Details**
Number	015
Name	Moderate Reviews
Summary	Allows the admin to monitor, approve, or remove customer reviews to ensure appropriate content and maintain platform credibility.
Priority	4
Pre-Conditions	Reviews exist in the system.
Post-Conditions	Reviews are approved, flagged, or removed.
Primary Actors	Admin
Secondary Actors	Customer, Service Provider
Triggers	Customer submits a new review.
Main Scenario / Actions	1. Admin logs into the system. 2. Admin navigates to “Moderate Reviews.” 3. System displays new and flagged reviews. 4. Admin reviews content for appropriateness. 5. Admin approves, removes, or flags the review. 6. System updates the provider profile and notifies the customer if necessary.
Extensions	**4a.** Review contains inappropriate content → admin removes and optionally warns the customer.
Open Issues	1. Should providers be notified of moderation decisions? 2. Should customers be able to appeal moderation decisions?

Field	**Details**
Number	016
Name	View Reports & Analytics
Summary	Allows the admin to generate and view reports and analytics related to customers, service providers, bookings, payments, and platform usage to support decision-making.
Priority	4
Pre-Conditions	Relevant data (bookings, payments, reviews) exists in the system.
Post-Conditions	Reports and analytics are displayed or exported.
Primary Actors	Admin
Secondary Actors	-
Triggers	Admin selects “Reports & Analytics” from the dashboard.
Main Scenario / Actions	1. Admin logs into the system. 2. Admin navigates to “Reports & Analytics.” 3. System displays options for generating reports (e.g., bookings, payments, reviews, popular services). 4. Admin selects report type and applies filters (date, category, region). 5. System generates and displays the report or allows export (PDF/CSV).
Extensions	**4a.** No data available → system shows “No data for selected filters.”
Open Issues	1. Should reports be real-time or periodic? 2. Who has access to sensitive financial reports?

With all the use cases clearly defined, including primary and secondary actors, main scenarios, alternate flows, and open issues, we have a comprehensive understanding of the functional requirements of the Gamata Seva platform. This foundation allows us to now focus on **Behavioral Modeling**, which captures the dynamic interactions, workflows, and sequences within the system through diagrams like **Activity Diagrams**, **Sequence Diagrams**, and **State Diagrams**.

BEHAVIORAL MODELING

Behavioral modeling focuses on showing the dynamic aspects of the Gamata Seva system. It illustrates how the system reacts to various events and interactions. While use case modeling captures what the system should do, behavioral modeling explains how the system performs those functions over time.

It primarily uses three techniques:

- **Activity Diagrams**
- **Sequence Diagrams**
- **State Diagrams**

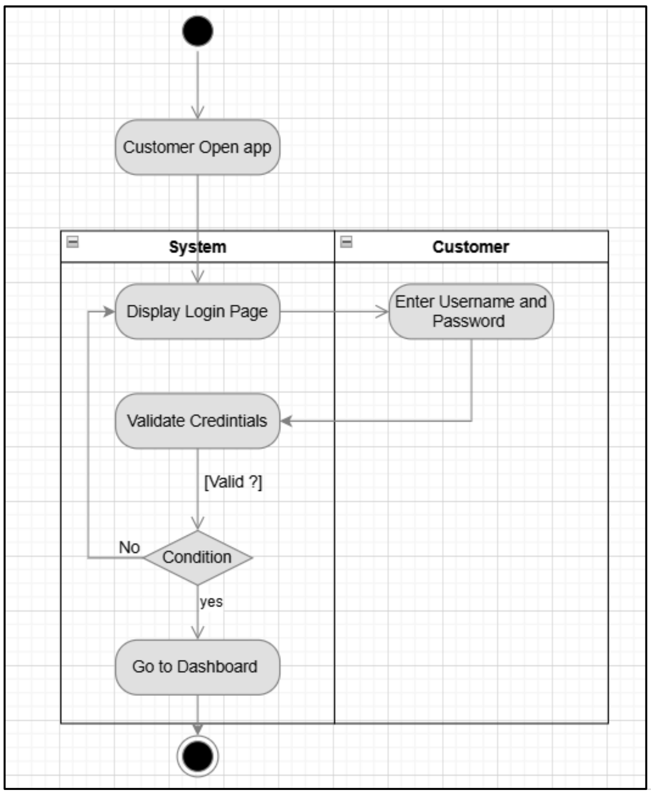
Activity diagrams

Activity diagrams are used to represent the flow of control or data within the Gamata Seva system for specific use cases. They show how various actors interact with the system, the sequence of actions, decision points, parallel activities, and the start/end of processes.

For clarity and simplicity, only the most critical activities are modeled, covering actions for **customers, service providers, and admins**. Swimlanes are used to separate responsibilities of different actors.

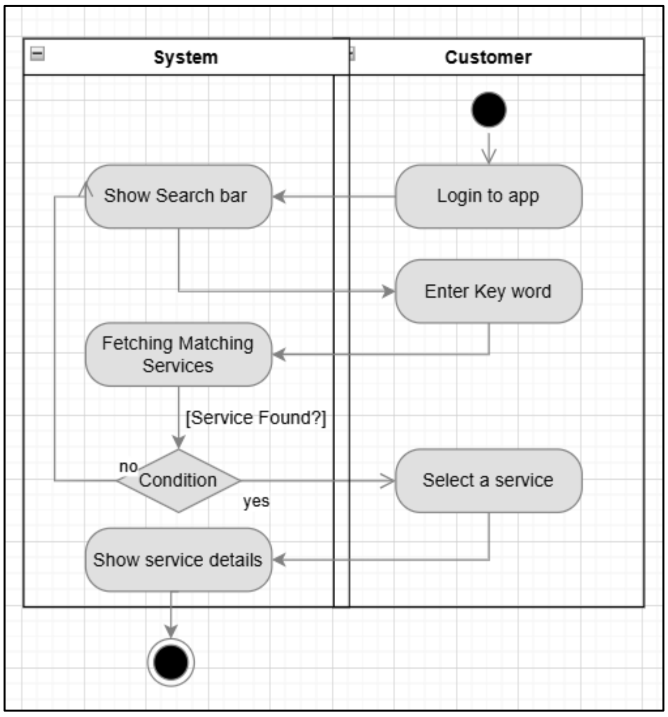
I used draw.io.

1. Login (Customer)



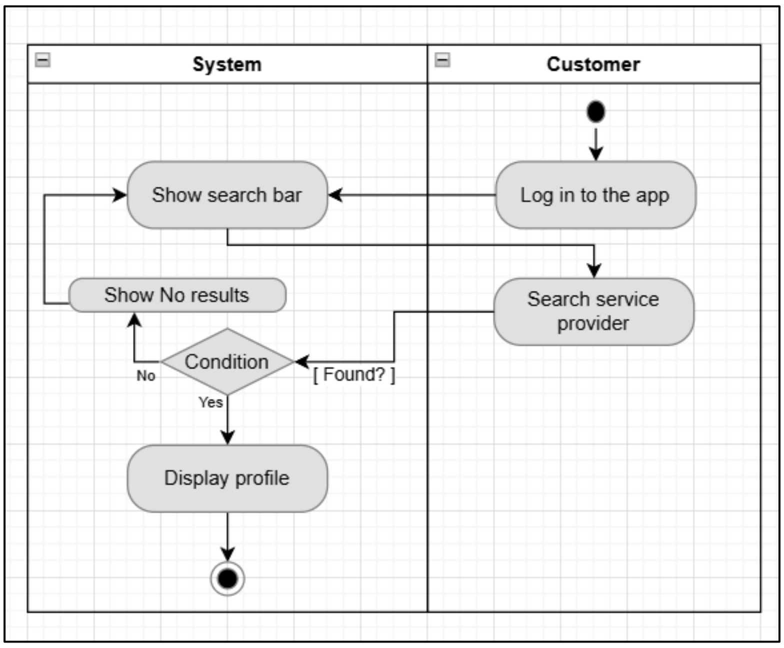
This activity diagram illustrates the customer login process in the Gamata Seva platform. The flow begins when the customer opens the app and is presented with a login page by the system. The customer then enters their username and password, which the system validates. If credentials are invalid, an error condition triggers a loop back to re-enter details. Upon successful validation, the system redirects the user to their personalized dashboard. Swimlanes clearly separate responsibilities between the Customer (user actions) and System (backend logic). This ensures secure, intuitive access while maintaining usability for rural users unfamiliar with complex interfaces.

2. Search for a Service (Customer)



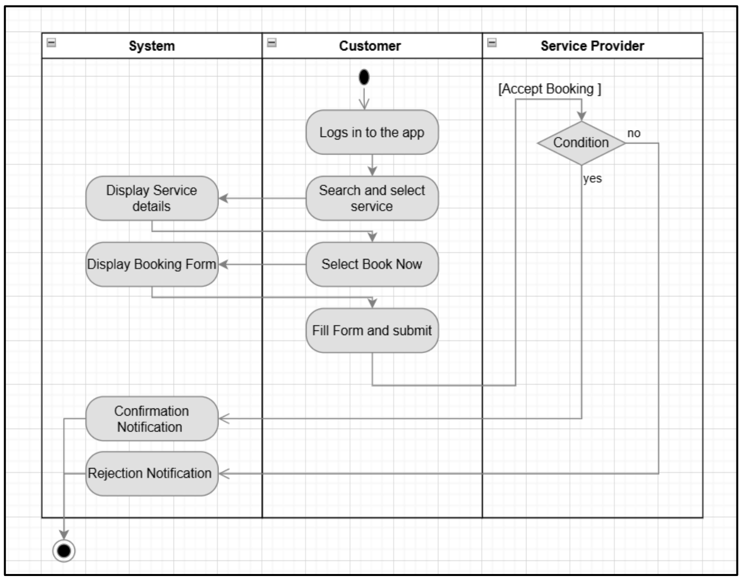
This diagram outlines how a customer searches for a service on Gamata Seva. The process begins after the customer logs into the app, triggering the system to display a search bar. The customer then enters a keyword (e.g., "plumber" or "tutor"). The system fetches matching services from its database and evaluates whether any results exist. If no services are found, the flow loops back to allow re-entry of keywords. If services are found, the customer selects one, prompting the system to display detailed information about that service provider. Swimlanes distinguish user actions (Customer) from platform operations (System), ensuring clarity in responsibilities and enhancing usability for rural users seeking local services efficiently.

3. View Provider Profile (Customer)



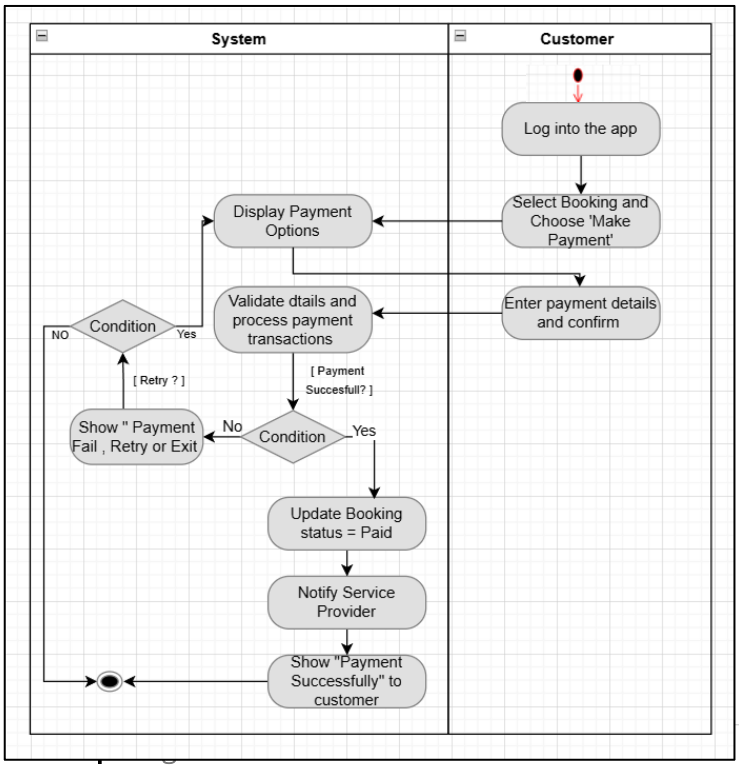
This diagram illustrates how a customer accesses a service provider's profile on Gamata Seva. After logging in, the customer searches for a specific provider. The system checks if the provider exists. If not found, it displays "No results" and returns to the search bar for refinement. If found, the system retrieves and displays the provider's full profile — including ratings, services offered, contact info, and availability. Swimlanes clearly separate user actions (Customer) from system responses (System), ensuring intuitive navigation. This flow empowers rural customers to make informed decisions by verifying provider credibility before booking, enhancing trust and usability in low-digital-literacy environments.

4. Book a Service (Customer)



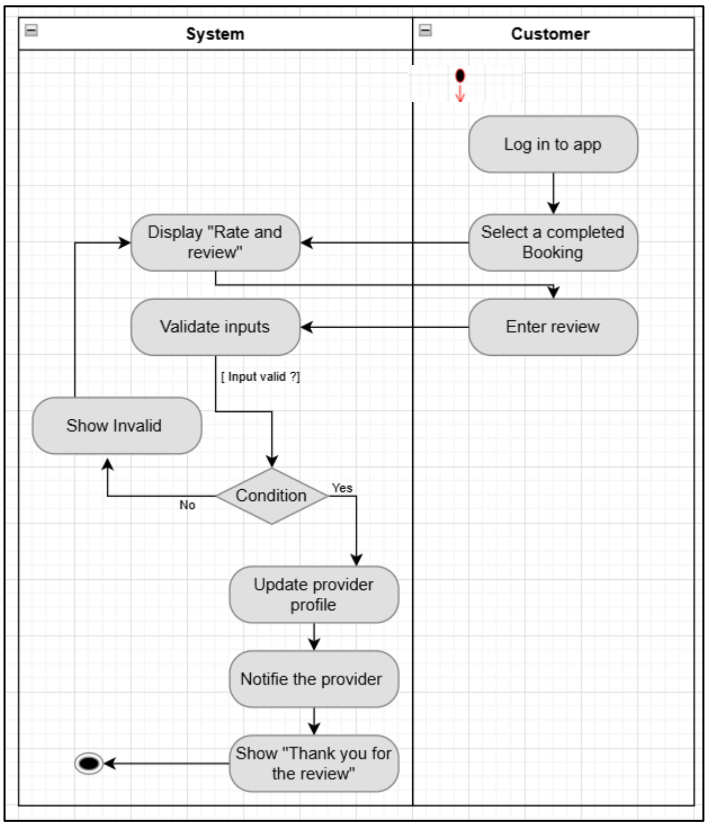
This diagram illustrates the end-to-end booking process for customers on Gamata Seva. After logging in, the customer searches and selects a service, then clicks "Book Now." The system displays a booking form, which the customer fills and submits. The request is sent to the Service Provider, who reviews it via a decision node labeled "Accept Booking?" If accepted, the system sends a confirmation notification to the customer; if rejected, a rejection notification is sent. Swimlanes clearly separate responsibilities among Customer, System, and Service Provider, ensuring transparency and accountability. This flow supports rural users by simplifying bookings while enabling providers to manage requests efficiently — building trust through clear communication.

5. Make Payment (Customer)


System
Customer
Log into the app
Select Booking and Choose 'Make Payment'
Display Payment Options
Enter payment details and confirm
Validate dtails and process payment transactions
[Payment Succesfull?]
Condition
No
Yes
Show " Payment Fail , Retry or Exit
[Retry ?]
Condition
NO
Yes
Update Booking status = Paid
Notify Service Provider
Show "Payment Successfully" to customer

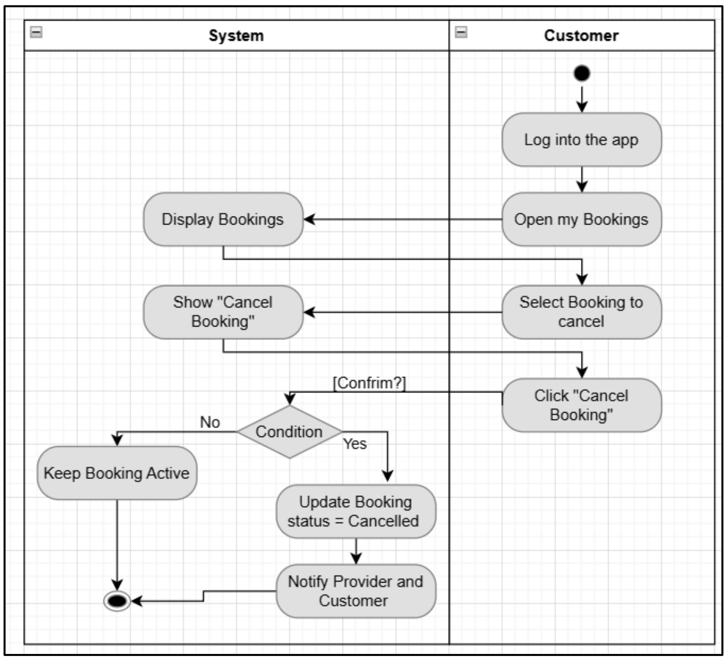
This diagram outlines the payment workflow for customers on Gamata Seva. After logging in, the customer selects a booking and chooses "Make Payment." The system displays available payment options. The customer enters payment details and confirms. The system validates and processes the transaction. If payment fails, the user is prompted to retry or exit. If successful, the booking status updates to "Paid," the service provider is notified, and the customer receives a "Payment Successfully" confirmation. Swimlanes clearly separate user actions from system responses, ensuring reliability and transparency.

6. Rate and Review



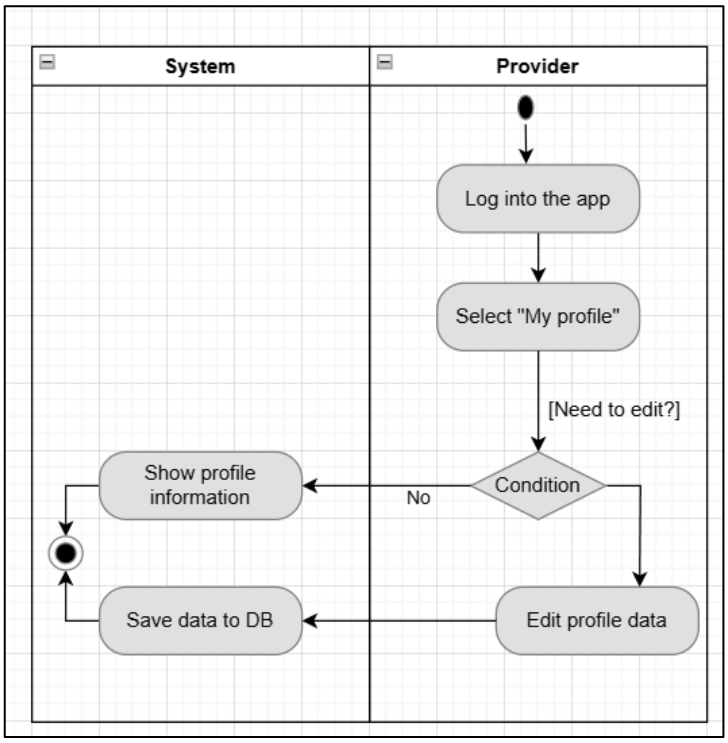
This diagram illustrates how customers rate and review service providers on Gamata Seva. After logging in, the customer selects a previously completed booking to review. The system displays a "Rate and Review" interface. The customer enters their feedback and rating, which the system validates for completeness and format. If input is invalid, an error message appears, prompting correction. Upon valid submission, the system updates the provider's profile with the new rating, notifies the provider, and displays a "Thank you for the review" message to the customer. Swimlanes clearly separate user actions from system logic, ensuring transparency. This process builds community trust by enabling honest, structured feedback — vital for rural users relying on reputation-based decisions.

7. Cancel Booking



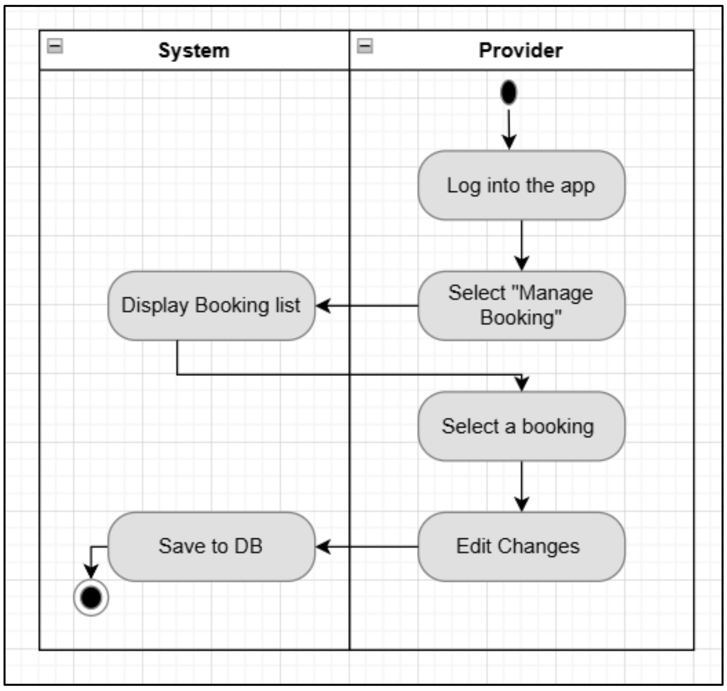
This diagram illustrates the customer's booking cancellation process on Gamata Seva. After logging in, the customer accesses "My Bookings," selects a specific booking, and clicks "Cancel Booking." The system prompts for confirmation via a decision node. If the customer declines ("No"), the booking remains active. If confirmed ("Yes"), the system updates the booking status to "Cancelled," then notifies both the service provider and customer to ensure transparency. Swimlanes clearly separate user actions from system responses, minimizing confusion. This structured flow empowers rural users to manage appointments confidently while maintaining accountability between parties — crucial for trust-building in community-based service ecosystems.

8. List and Update Profile (Service provider)



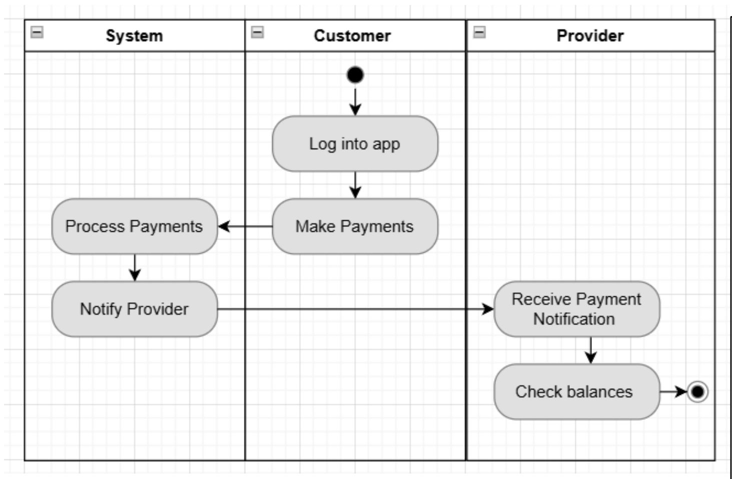
This diagram illustrates how a service provider manages their profile on Gamata Seva. After logging in, the provider accesses the dashboard and selects “My Profile.” The system checks if editing is needed. If not, it displays current profile information and ends the flow. If editing is required, the provider modifies details (e.g., contact, services, availability), and the system saves the updated data to the database. Swimlanes clearly separate provider actions from system responses, ensuring intuitive navigation. This streamlined process empowers rural providers — often with limited tech exposure — to maintain accurate, attractive profiles that attract customers, boosting visibility and trust within the local service ecosystem.

9. Manage Bookings (Service provider)



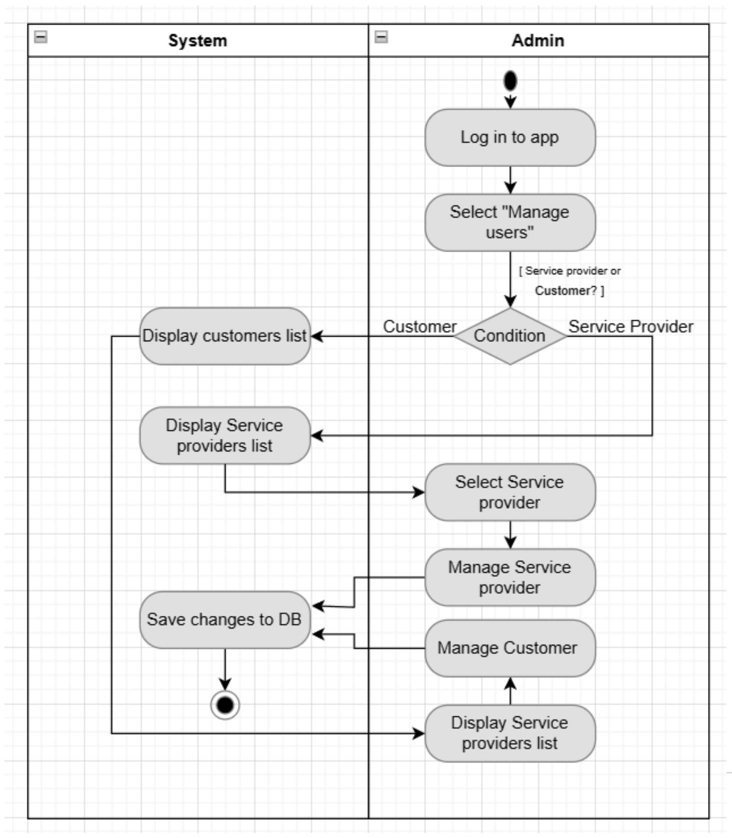
This diagram illustrates how a service provider manages customer bookings on Gamata Seva. After logging in, the provider selects “Manage Booking,” prompting the system to display their booking list. The provider then selects a specific booking and edits its details — such as date, time, or status — based on real-world availability. Once changes are made, the system saves the updated information to the database, ensuring synchronization across the platform. Swimlanes clearly separate provider actions from system responses, enabling intuitive control. This flow empowers rural providers to efficiently manage schedules, reduce no-shows, and maintain professionalism — critical for building customer trust and sustaining local livelihoods through digital tools.

10. Receive payments (Provider)



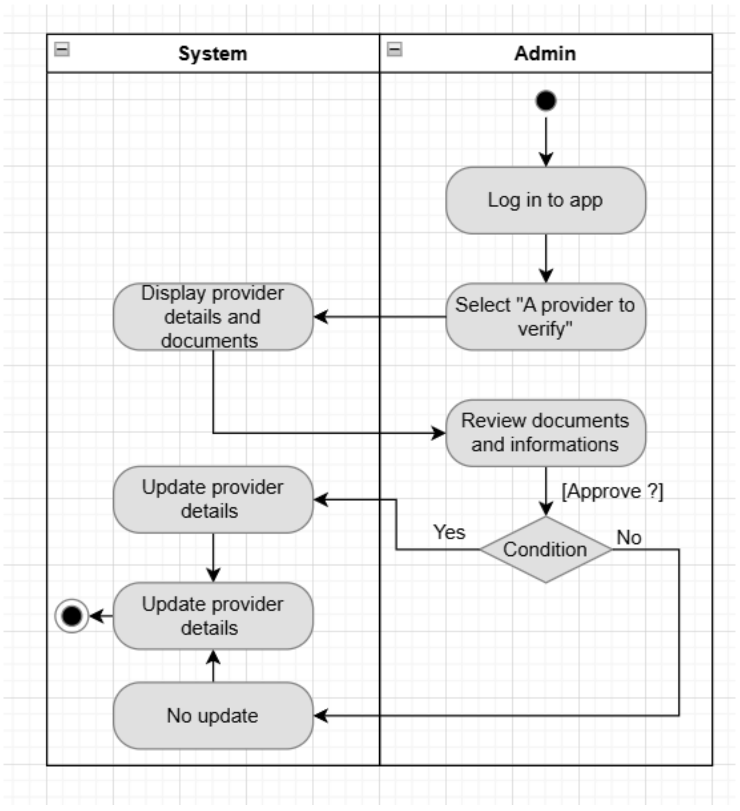
This diagram focuses on the provider's experience *after* payment — not making it, but receiving confirmation and accessing funds, as the "Make Payment" process is already detailed in a prior activity diagram (Customer flow). Here, once the customer completes payment (triggered externally), the system processes it and notifies the provider. The provider receives an in-app alert, views the payment notification, and checks their balance to confirm receipt. Swimlanes separate Customer (initiates payment), System (processes & notifies), and Provider (receives outcome). This design avoids redundancy while emphasizing financial transparency — vital for rural providers who rely on timely, visible income updates to manage livelihoods confidently.

11. Manage Customers and Service providers (Admin)



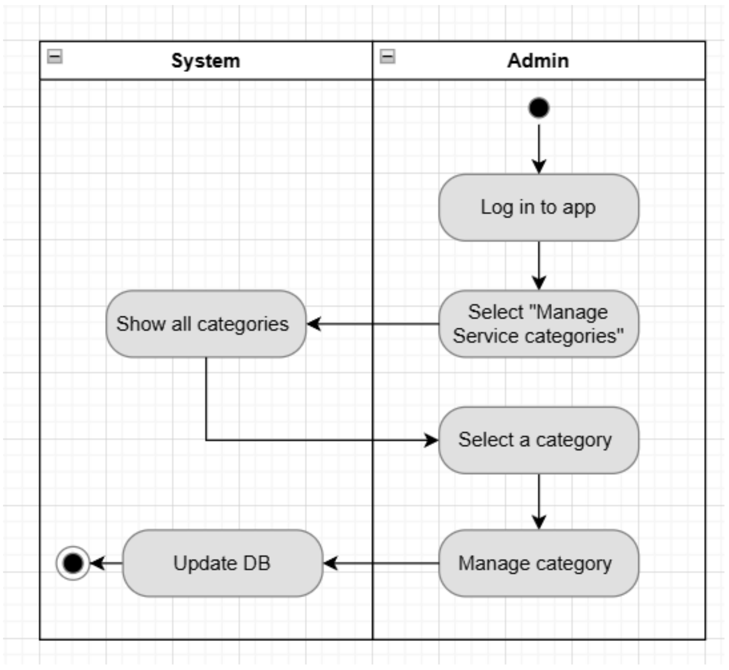
This diagram illustrates how the Admin manages users on Gamata Seva. After logging in, the Admin selects "Manage Users," prompting a decision: manage *Customers* or *Service Providers*. Based on selection, the system displays the respective list. The Admin can then select a specific user to view or edit details — such as status, contact, or verification level. Changes are saved to the database, ensuring data consistency. Swimlanes clearly separate Admin actions from System responses, enabling efficient oversight. This centralized control is vital for maintaining platform integrity, especially in rural areas where trust and accountability rely on verified, well-managed user profiles — empowering Admins to curate a safe, reliable service ecosystem.

12. Verify service providers (Admin)



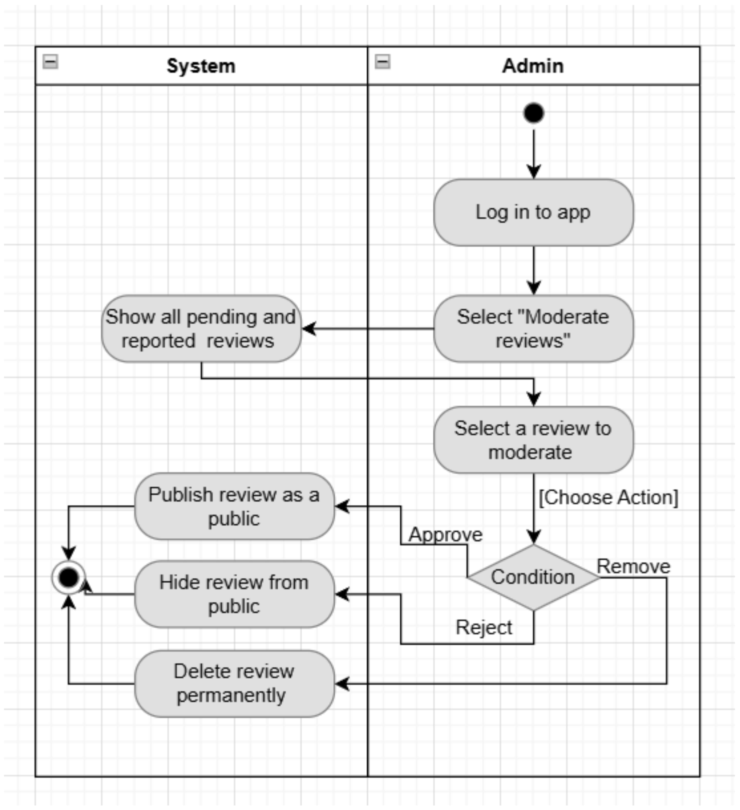
This diagram outlines the Admin's workflow for verifying service providers on Gamata Seva. After logging in, the Admin selects a provider to verify. The system displays the provider's submitted details and supporting documents. The Admin reviews them and decides whether to approve. If approved ("Yes"), the system updates the provider's status to "Verified," enabling visibility to customers. If rejected ("No"), no update occurs — the provider remains unverified. Swimlanes clearly separate Admin decisions from System actions, ensuring accountability. This verification gate is critical for rural trust-building, as it filters unreliable providers and ensures only credible, documented professionals serve the community — reinforcing platform integrity and user confidence.

13. Manage service categories (Admin)



This diagram illustrates how the Admin organizes and maintains service categories on Gamata Seva. After logging in, the Admin selects "Manage Service Categories," prompting the system to display all existing categories (e.g., "Plumbing," "Tutoring"). The Admin then selects a specific category to edit, add, or remove. Any changes are finalized through the "Manage Category" action and saved to the database via "Update DB." Swimlanes clearly separate Admin decisions from System responses, ensuring structured control over platform taxonomy. This functionality is vital for rural usability — accurate, localized categories help customers find relevant services quickly, while empowering Admins to adapt offerings based on community needs and emerging local trades.

14. Moderate reviews (Admin)



System
Admin
Log in to app
Select "Moderate reviews"
Show all pending and reported reviews
Select a review to moderate
[Choose Action]
Condition
Approve
Remove
Reject
Publish review as a public
Hide review from public
Delete review permanently

This diagram illustrates how the Admin ensures review integrity on Gamata Seva. After logging in, the Admin selects "Moderate Reviews," prompting the system to display all pending or reported reviews. The Admin selects one for moderation and chooses an action: *Approve* (publish publicly), *Reject* (hide from public view), or *Remove* (delete permanently). Each decision triggers a corresponding system response — publishing, hiding, or deleting the review. Swimlanes clearly separate Admin judgment from System execution, ensuring accountability. This moderation layer is vital for rural trust — it filters inappropriate or fake feedback, preserving platform credibility and encouraging honest, constructive user engagement in community-based service ecosystems.

While activity diagrams highlight the flow of actions within the system, sequence diagrams focus on the chronological order of interactions between actors and system components. Therefore, the next section presents sequence diagrams to illustrate how objects collaborate over time to accomplish key use cases in the Gamata Seva platform.

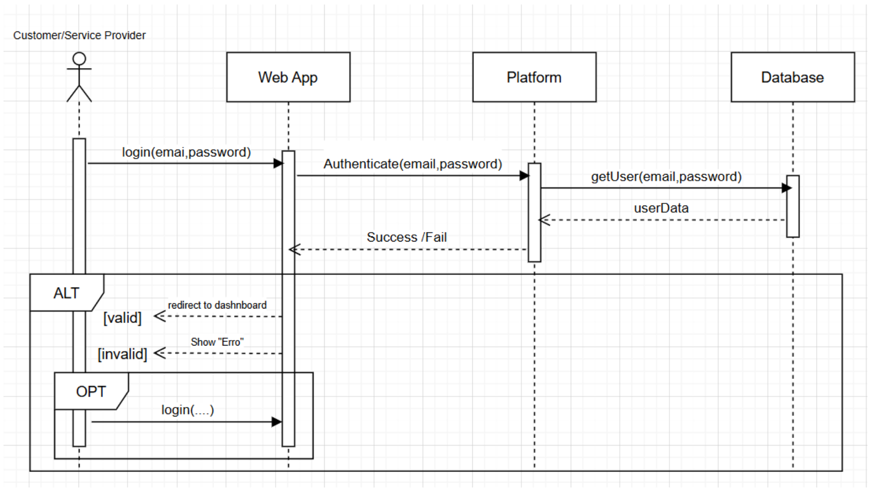
Sequence Diagrams

Sequence diagrams are a vital part of behavioral modeling as they emphasize the time-based interactions between system components and external actors. Unlike activity diagrams, which describe the flow of activities, sequence diagrams illustrate how different objects and actors collaborate through messages over time to achieve specific functionalities.

For the Gamata Seva system, these diagrams demonstrate the step-by-step communication between customers, service providers, and the admin with the platform, ensuring clarity on how requests are processed, validated, and completed.

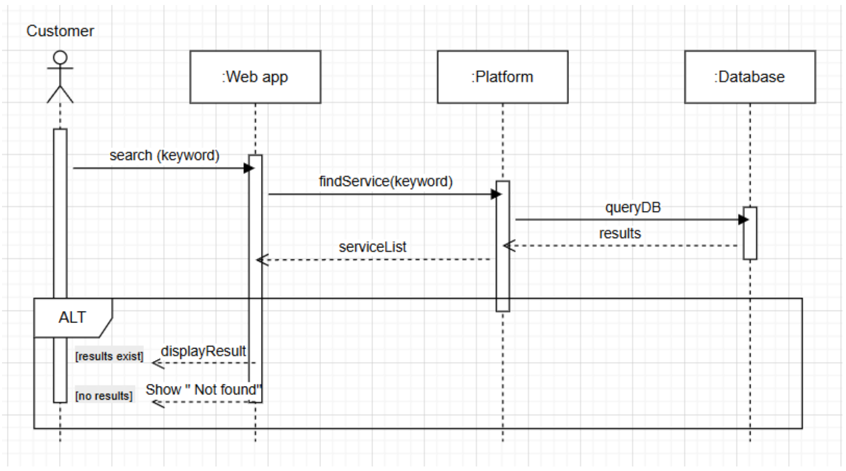
I used draw.io

1. Login (Customer /Service provider)



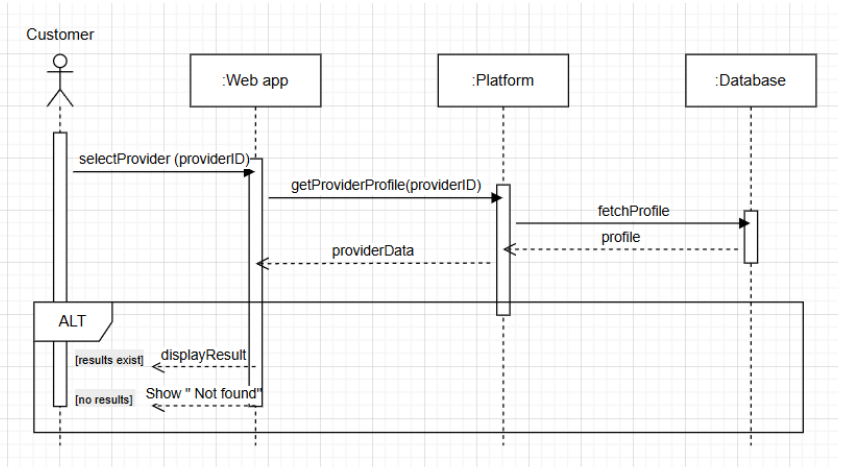
This diagram shows login flow for both users. After submitting credentials, the Web App authenticates via Platform and Database. On success, user is redirected to dashboard; on failure, an error is shown. Optional retry allows re-attempt. Ensures secure, intuitive access for rural users with minimal digital literacy.

2. Search Services (Customer)



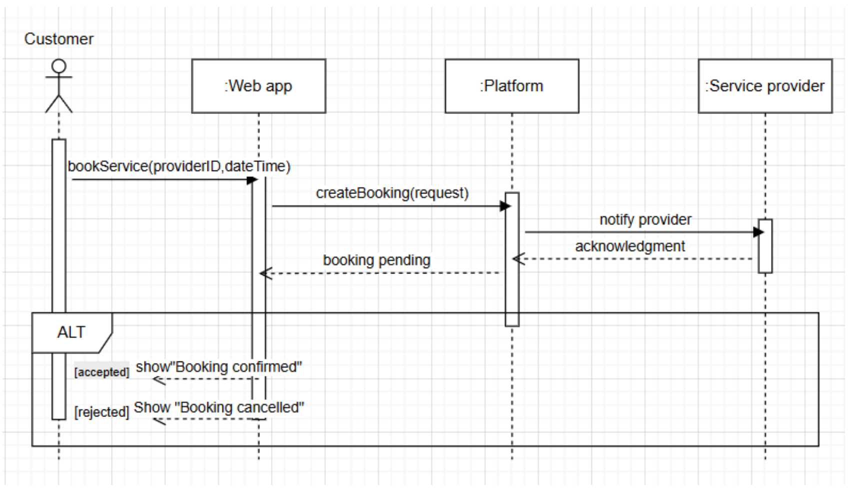
This diagram shows how a customer searches for services. The Web App forwards the keyword to the Platform, which queries the Database. Results are returned and displayed if found; otherwise, "Not found" is shown. The ALT fragment handles both outcomes clearly, ensuring rural users receive immediate, intuitive feedback during service discovery.

3. View provider profile (Customer)



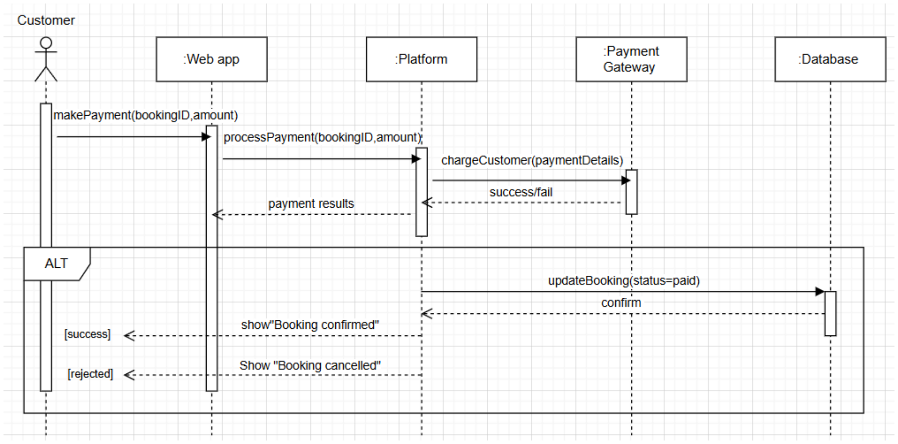
This diagram shows how a customer views a provider's profile. The Web App requests data from the Platform, which fetches it from the Database. If found, the profile displays; if not, "Not found" appears. The ALT fragment ensures clear, user-friendly feedback — critical for rural users navigating services with minimal tech experience.

4. Book a service



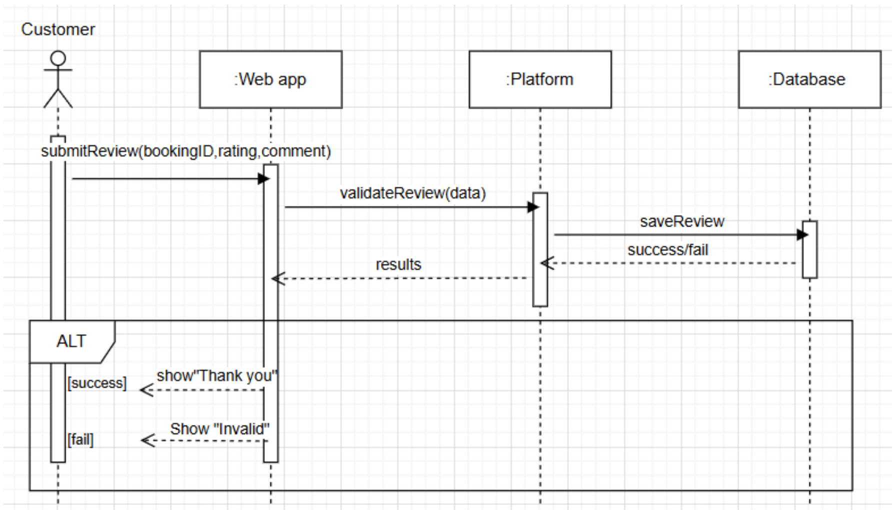
This diagram shows how a customer books a service. The Web App sends the request to the Platform, which notifies the provider. After acknowledgment, the Platform confirms or cancels the booking based on provider response. The ALT fragment ensures clear outcomes — vital for rural users needing transparent, reliable service scheduling without technical confusion.

5. Make payment (Customer)


Customer
:Web app
:Platform
:Payment Gateway
:Database
makePayment(bookingID,amount)
processPayment(bookingID,amount)
chargeCustomer(paymentDetails)
success/fail
payment results
ALT
updateBooking(status=paid)
confirm
[success] show"Booking confirmed"
[rejected] Show "Booking cancelled"

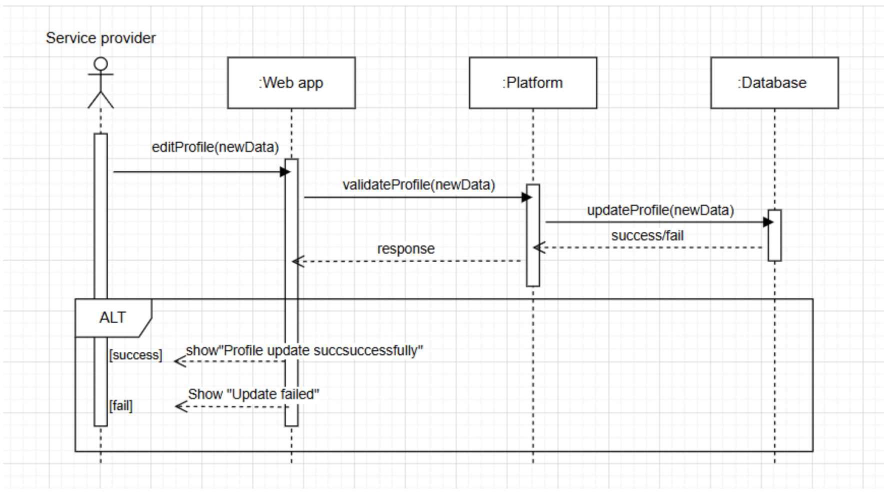
This diagram shows how a customer pays for a service. The Web App sends payment details to the Platform, which charges via Payment Gateway. On success, booking status updates in the Database and "Booking confirmed" appears. On failure, "Booking cancelled" is shown. The ALT fragment ensures clear outcomes — vital for trust in rural, low-tech environments.

6. Rate and review provider (Customer)



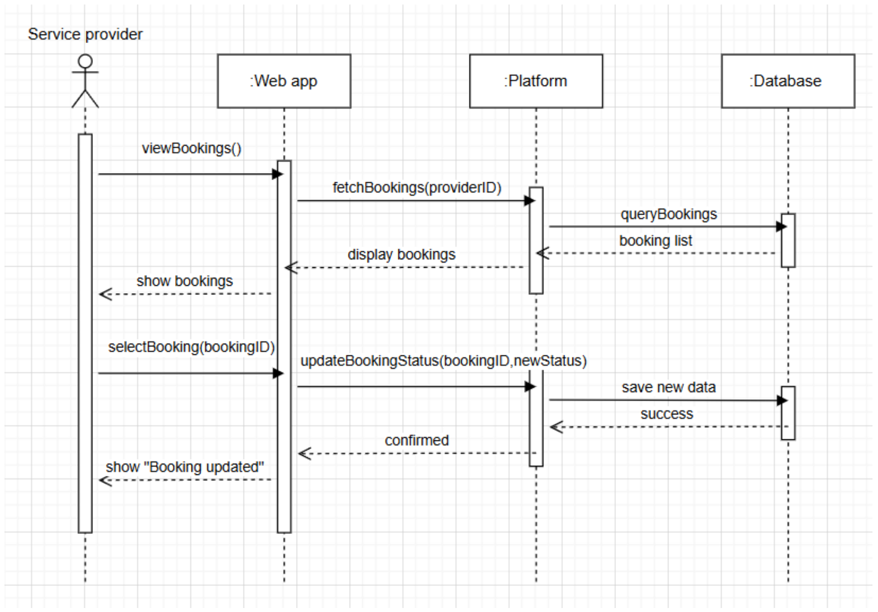
This diagram shows how a customer rates a provider after service. The Web App sends data to the Platform, which validates and saves it to the Database. On success, "Thank you" appears; on failure, "Invalid" is shown. The ALT fragment ensures clear feedback — essential for rural users submitting reviews with minimal tech experience and maintaining platform credibility.

7. Update profile (Service provider)



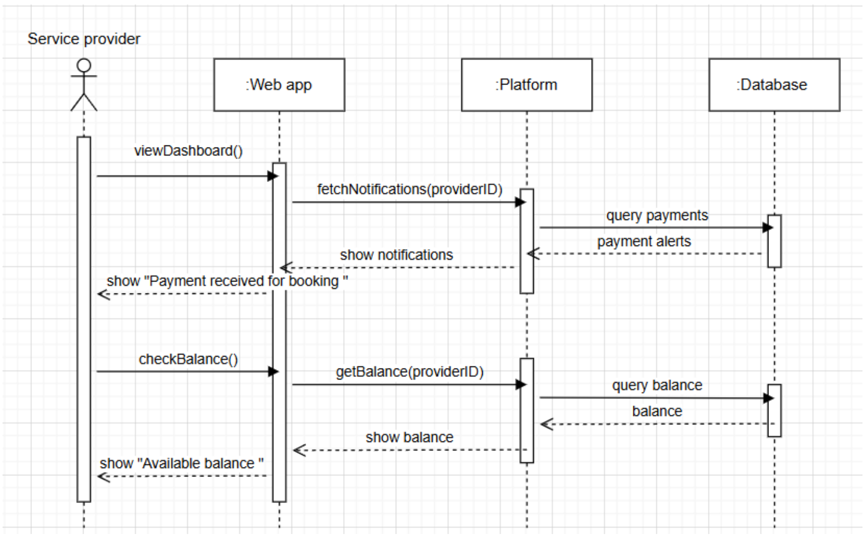
This diagram shows how a service provider updates their profile. The Web App sends data to the Platform, which validates and saves it to the Database. On success, “Profile updated successfully” appears; on failure, “Update failed” is shown. The ALT fragment ensures clear feedback — vital for rural providers managing their visibility with minimal tech experience.

8. Manage Bookings (Service provider)



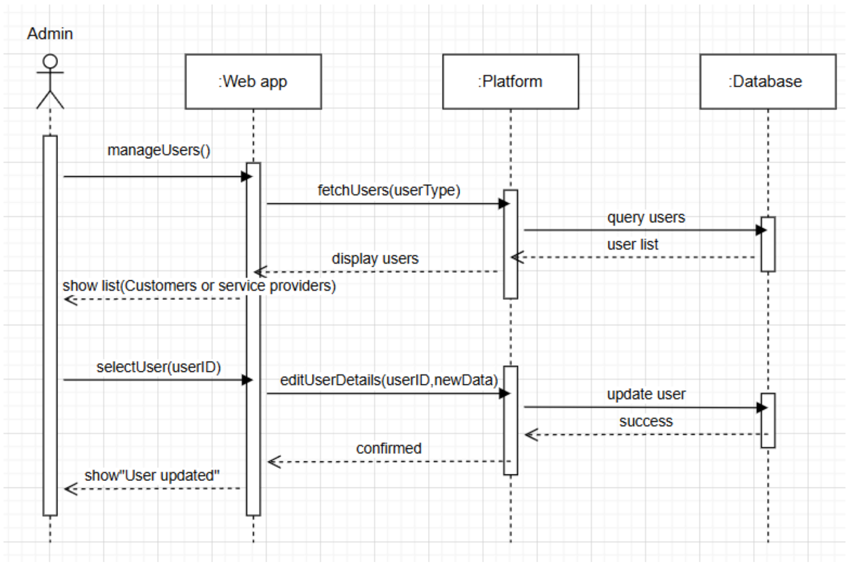
This diagram shows how a service provider manages bookings. The Web App fetches and displays their bookings from the Database via the Platform. After selecting one, they update its status — which is saved and confirmed. This streamlined flow empowers rural providers to manage schedules efficiently, ensuring timely communication and reliable service delivery in low-tech environments.

9. Receive payments (Service provider)



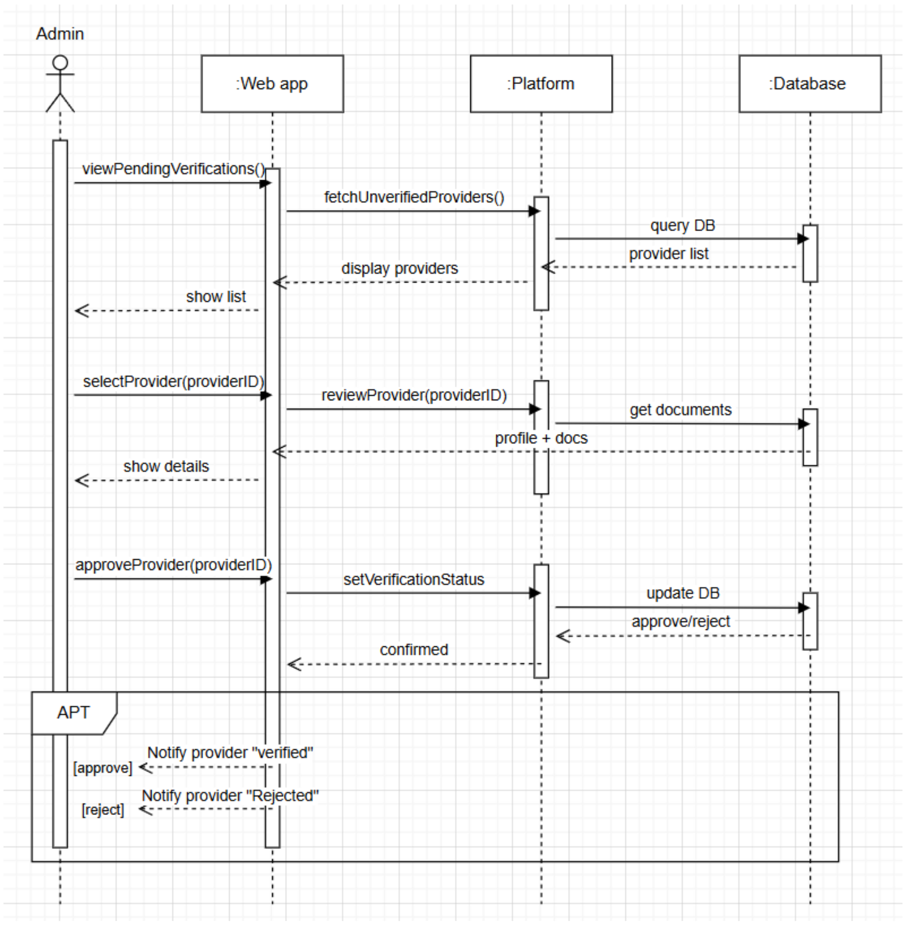
This diagram shows how a service provider receives payment notifications and checks balance. The Web App fetches alerts and balance from the Platform, which queries the Database. Clear, real-time updates empower rural providers to track earnings reliably — vital for financial transparency and trust in low-digital-literacy, cash-dependent communities.

10. Manage Customers and Service providers (Admin)



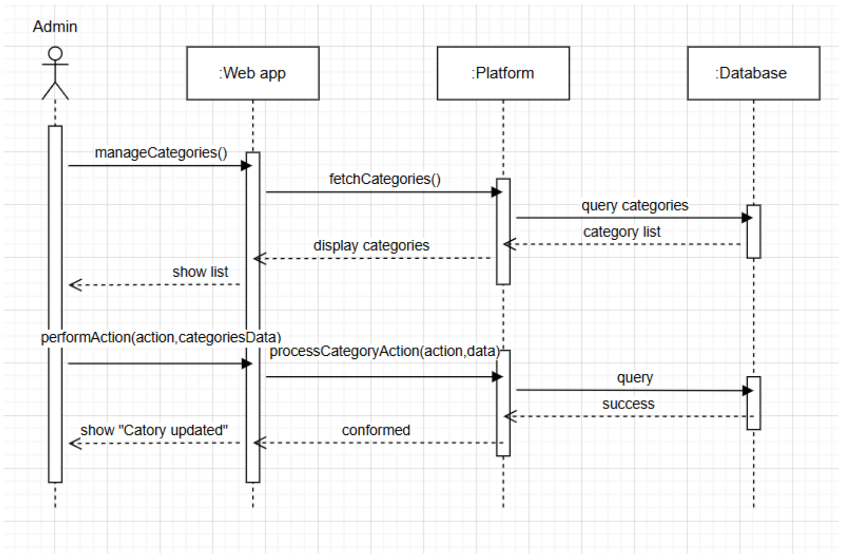
This diagram shows how the Admin manages users. The Web App fetches and displays customer or provider lists from the Database via the Platform. After selecting a user, edits are saved and confirmed. This centralized control ensures data accuracy and trust — vital for maintaining a safe, reliable service ecosystem in rural communities.

11. Verify service provider (Admin)



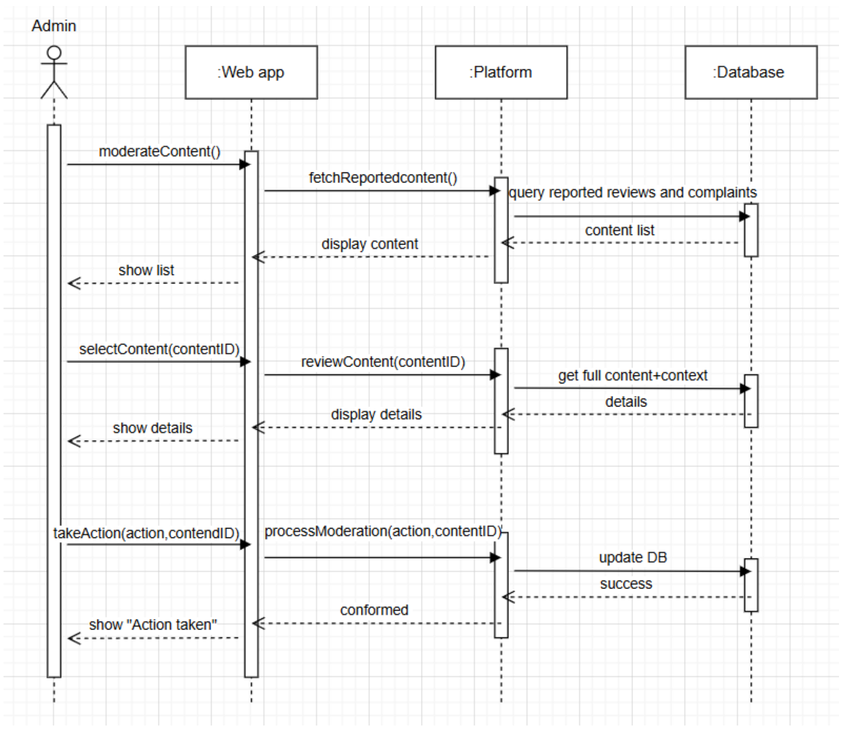
This diagram shows how the Admin verifies providers. The Web App fetches pending profiles, displays details, and processes approval/rejection. The Platform updates the Database and triggers notifications. An APT fragment ensures providers are informed of their status — vital for transparency and trust in rural communities where verified credentials build service credibility.

12. Manage service categories (Admin)



This diagram shows how the Admin manages service categories. The Web App fetches and displays the list from the Database via the Platform. After selecting an action (add/edit/delete), changes are processed and confirmed. This ensures categories stay relevant to rural needs — helping users find local services like farming or tutoring with ease.

13. Moderate reviews and complaints (Admin)



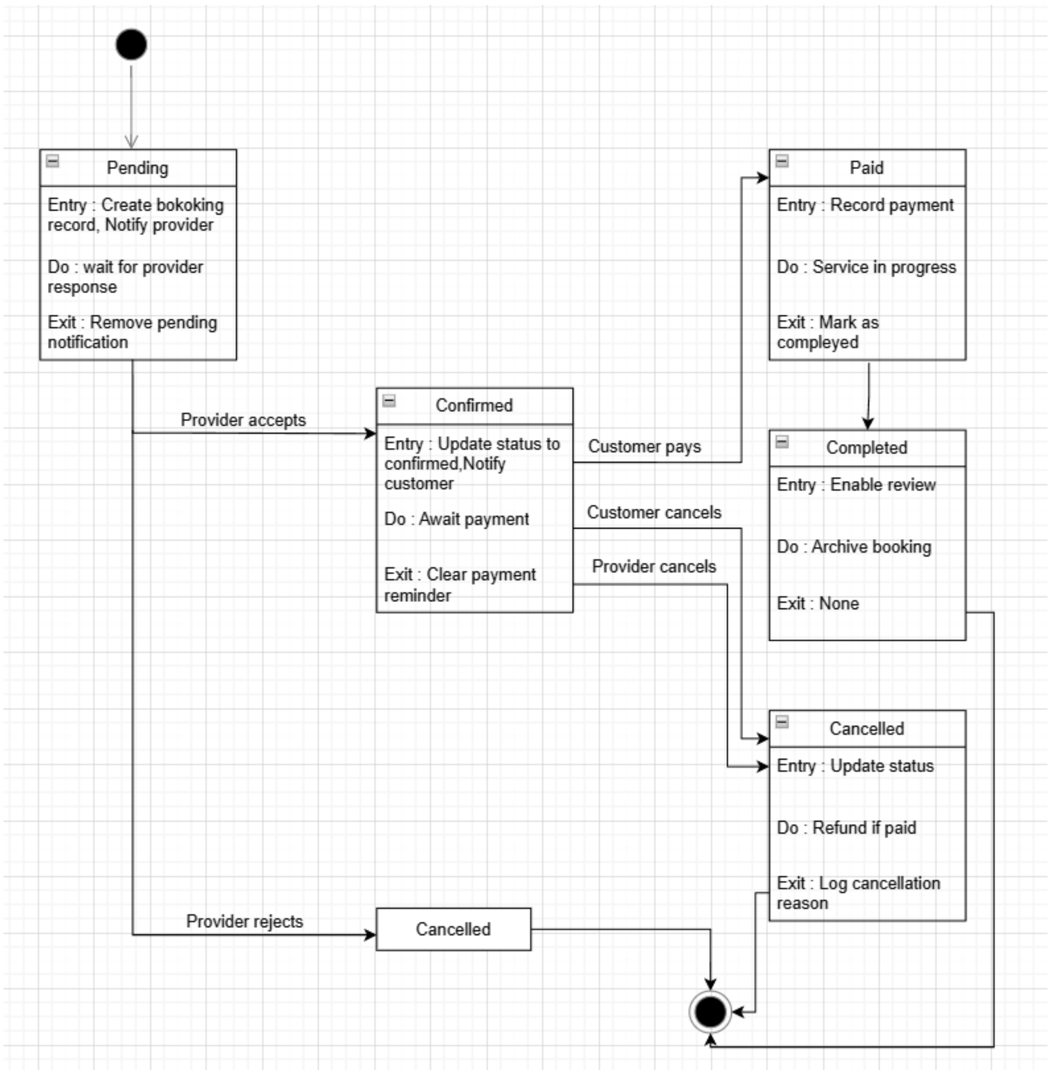
This diagram shows how the Admin moderates reviews and complaints. The Web App fetches reported content from the Database via the Platform, displays it for review, then processes the Admin's action (approve/reject/remove). A confirmation ensures accountability — vital for maintaining trust, safety, and honest feedback in rural service communities.

In summary, the sequence diagrams provide a detailed visualization of the interactions between customers, service providers, admins, and the system. They clearly demonstrate how different functions are coordinated in a time-ordered manner, ensuring that the Gamata Seva platform delivers seamless and efficient service flows.

The next section focuses on state modeling to capture how system entities evolve over time.

State Diagram

State diagrams are used to represent the different states that an object in the system can take and the transitions between these states based on events. In the Gamata Seva platform, state diagrams help to illustrate how system entities such as bookings, payments, or user accounts change status throughout their lifecycle. Unlike activity or sequence diagrams, which focus on processes and interactions, state diagrams emphasize the life cycle of individual objects. By modeling states, we can clearly understand how events trigger transitions and how the system ensures consistency and reliability across operations.



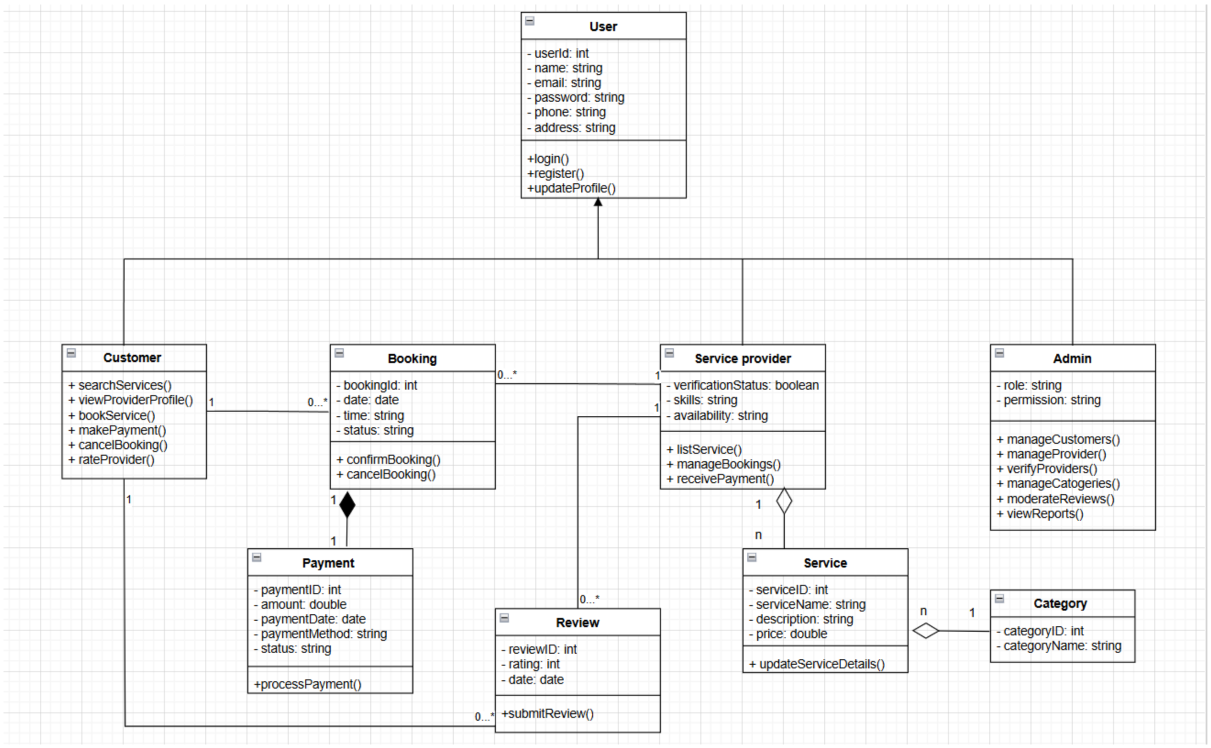
This state diagram models the complete lifecycle of a service booking in Gamata Seva, capturing all key interactions between customers and service providers. It begins with a *Pending* request, transitions to *Confirmed* upon provider acceptance, and advances to *Paid* after customer payment. The booking concludes as either *Completed* (enabling reviews) or *Cancelled* (from any active state, by either party). Each state includes entry, do, and exit actions—such as notifications, status updates, and refund handling—ensuring transparency and trust. This diagram integrates multiple use cases (booking, payment, cancellation, review) into a single, cohesive workflow, reflecting the real-world flexibility and reliability required in rural Sri Lankan service ecosystems.

After modeling behaviors, we now move to **structural modeling**, focusing on the system's components and their relationships.

STRUCTURAL MODELING

The structural modeling of the *Gamata Seva* system is represented using a **Class Diagram**, which illustrates the main classes, their attributes, operations, and the relationships among them. This diagram provides a clear understanding of the system's structure and how different entities interact within the application.

Class Diagram



This class diagram outlines Gamata Seva's core structure, defining key entities: User (parent), Customer, Service Provider, and Admin. It shows relationships like one Customer making many Bookings, each linked to one Payment and optional Review. Service Providers manage Services under Categories. Attributes and methods reflect functionality — from booking to payment and moderation. The diagram ensures system cohesion, scalability, and role-based access, vital for rural service delivery. Clear associations support database design and development, aligning user actions with backend logic while maintaining simplicity for low-tech users.

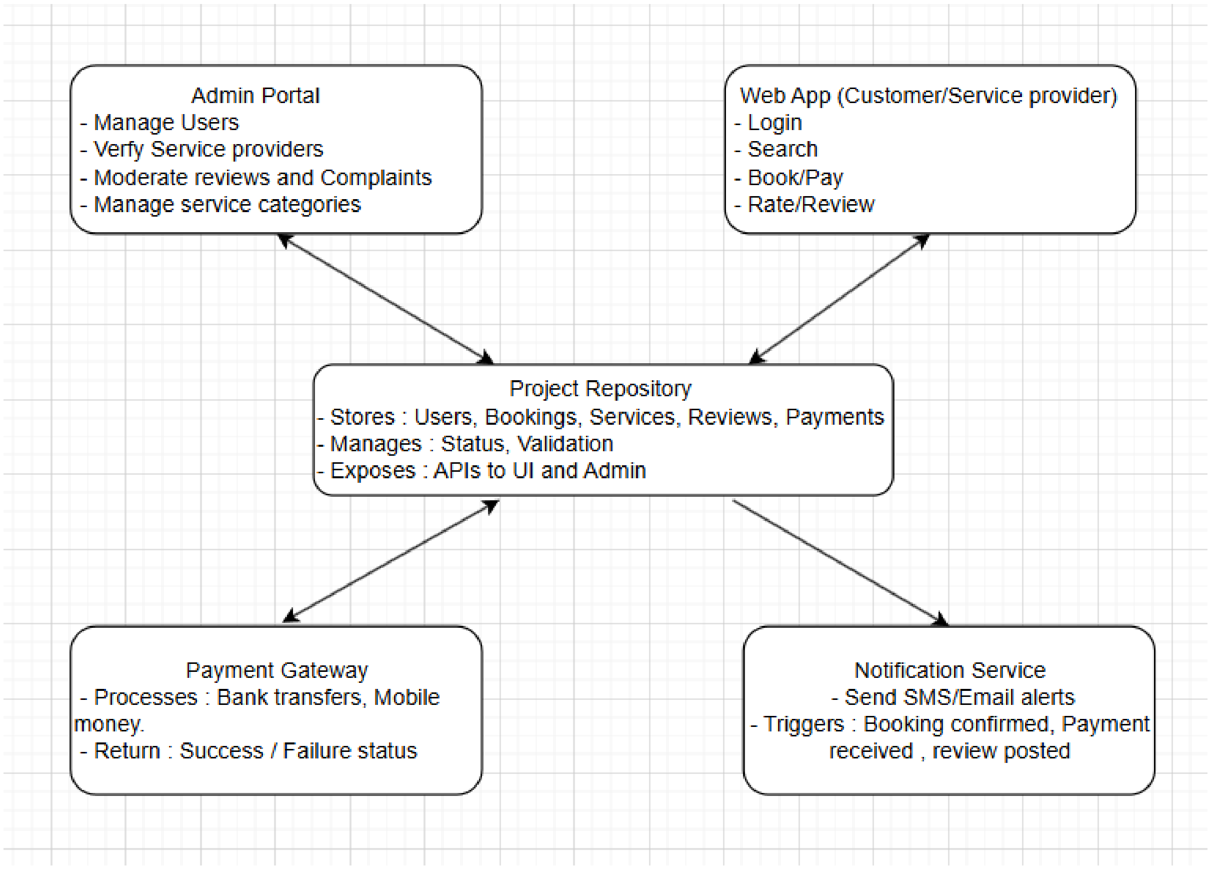
With the system's structure clearly defined through the class diagram, the next step focuses on designing the overall architecture of the Gamata Seva system.

SYSTEM ARCHITECTURE

The system architecture defines the overall structure and organization of the Gamata Seva system, showing how its components interact to achieve the required functionality.

For this project, the **Repository Architecture** was selected as it provides a centralized data management approach where all components access a shared data store. This architecture is ideal for Gamata Seva since multiple modules, such as customer management, service provider management, and booking. Need to share and update common data efficiently. It ensures consistency, easy maintenance, and smooth data flow across the entire system.

Repository Architecture



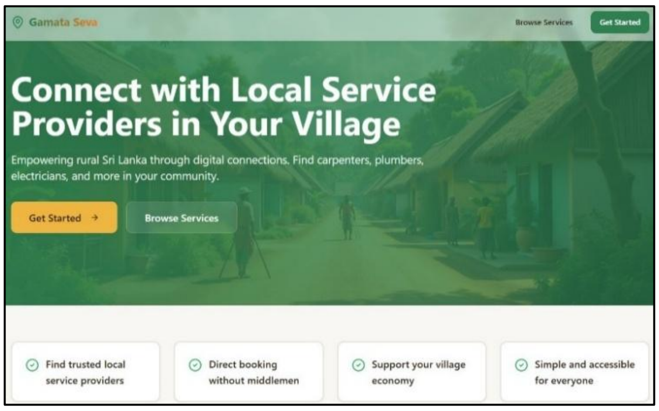
Admin Portal
- Manage Users
- Verfy Service providers
- Moderate reviews and Complaints
- Manage service categories
Web App (Customer/Service provider)
- Login
- Search
- Book/Pay
- Rate/Review
Project Repository
- Stores : Users, Bookings, Services, Reviews, Payments
- Manages : Status, Validation
- Exposes : APIs to UI and Admin
Payment Gateway
- Processes : Bank transfers, Mobile money.
- Return : Success / Failure status
Notification Service
- Send SMS/Email alerts
- Triggers : Booking confirmed, Payment received , review posted

This repository architecture centers on the Project Repository, storing core entities like Users, Bookings, Services, and Payments while managing validation and status. It exposes APIs to the Web App (for customers/providers) and Admin Portal (for moderation and verification). The Payment Gateway processes transactions via bank or mobile money, returning success/failure. Notification Service sends SMS/email alerts for key events like booking confirmation or payment receipt. Bidirectional arrows indicate data exchange between components, ensuring seamless interaction. This modular design supports scalability, security, and rural usability — aligning with Gamata Seva's mission to empower local service ecosystems through simple, reliable digital infrastructure.

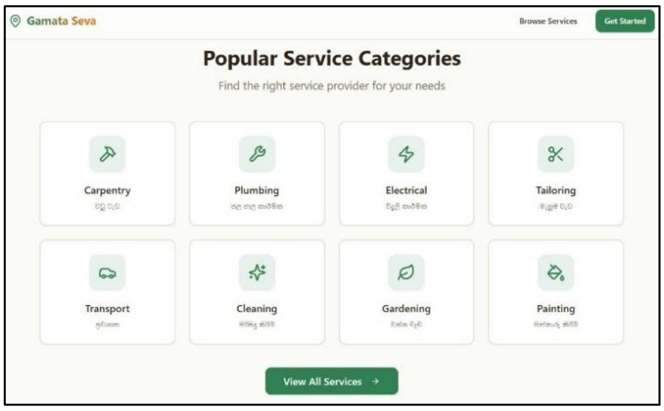
DESIGN

The following design section presents conceptual **UI wireframes and layouts** visualize the proposed interface of Gamata Seva. These mockups are not fully functional but serve to illustrate user flow and screen structure for stakeholders.

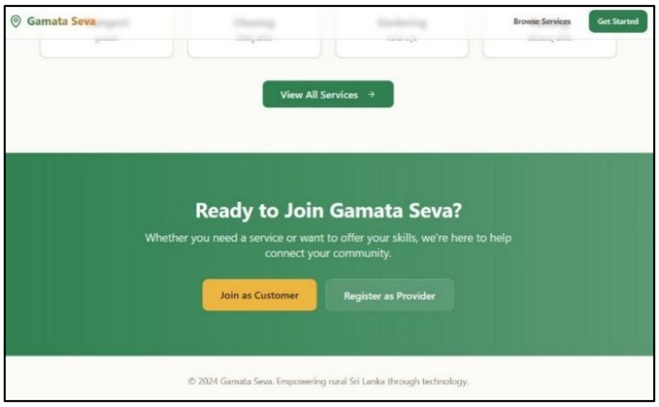
The focus remains on core system logic and rural usability — with future iterations aiming to implement responsive, accessible interfaces tailored for low-digital-literacy users.


Connect with Local Service Providers in Your Village

Empowering rural Sri Lanka through digital connections. Find carpenters, plumbers, electricians, and more in your community.
Gamata Seva
Browse Services
Get Started
Get Started
Browse Services
Find trusted local service providers
Direct booking without middlemen
Support your village economy
Simple and accessible for everyone

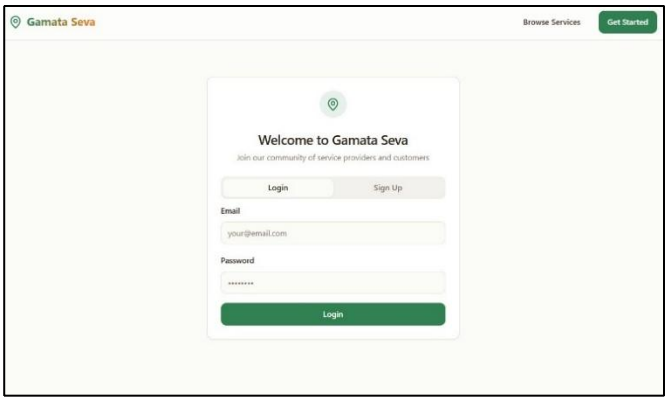

Popular Service Categories

Find the right service provider for your needs


Ready to Join Gamata Seva?

Whether you need a service or want to offer your skills, we're here to help connect your community.

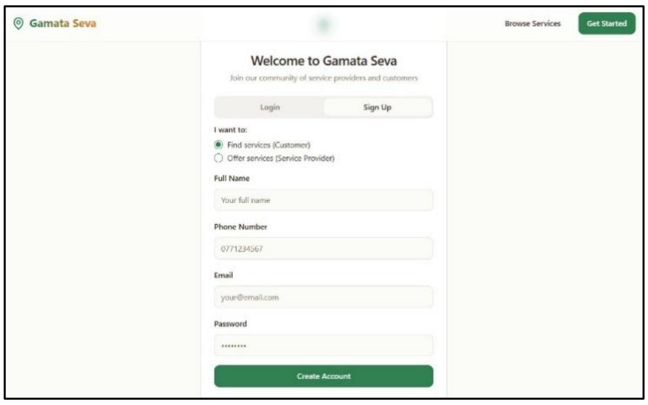
© 2024 Gamata Seva. Empowering rural Sri Lanka through technology.


Welcome to Gamata Seva

Join our community of service providers and customers

Email: your@email.com

Password: ••••••••


Welcome to Gamata Seva

Join our community of service providers and customers

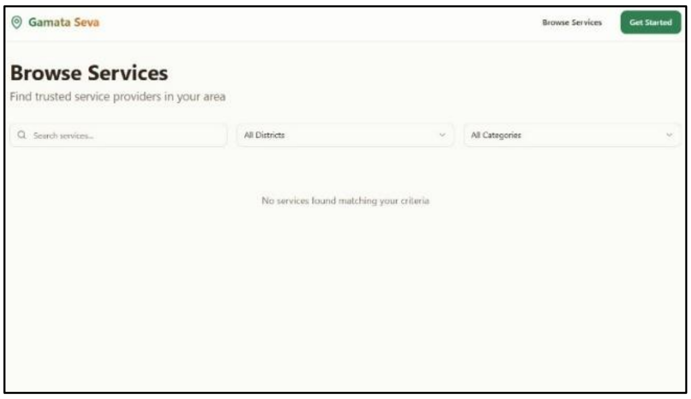
I want to:
- Find services (Customer)
- Offer services (Service Provider)

Full Name: Your full name

Phone Number: 0771234567

Email: your@email.com

Password: ••••••••


Browse Services

Find trusted service providers in your area

No services found matching your criteria

FUTURE TESTING PLANS

Although the *Gamata Seva* system is currently in its **design and prototype stage**, our team has outlined a structured plan for testing once development begins.

The main goal is to ensure the system's **functionality, reliability, and usability** across all modules, especially in real rural environments in Sri Lanka.

1. **Unit Testing:** Each individual component will be tested to verify its functionality.
   Example: Testing the ***Login Function*** *to ensure that when a valid username and password are entered, the system correctly redirects the user to their dashboard, while incorrect credentials trigger an error message.*
2. **Integration Testing:** This testing will focus on verifying that modules interact correctly once connected.
   Example: Testing the ***Booking and Payment modules*** *together to ensure that after a user confirms a booking, the payment page loads properly and the transaction is recorded in the system's central database.*
3. **System Testing:** After integration, the entire web application will be tested as a whole to check performance under real-world conditions.
   Example: Testing the ***full customer flow*** *from logging in, searching for a carpenter, booking the service, and paying online under* ***low-speed internet*** *common in rural Sri Lankan areas, ensuring the system remains stable.*
4. **Validation Testing:** This testing ensures the system meets the requirements collected during analysis.
   Example: Verifying that a ***customer can successfully register, search services, and make bookings*** *confirming that these features align with the project's user requirements defined in the requirement engineering phase.*
5. **Defect Testing:** Any bugs or errors discovered during testing will be documented, fixed, and retested to maintain reliability.
   Example: If the ***"Cancel Booking"*** *button doesn't remove the booking from the provider's schedule, the defect will be logged, corrected, and retested until it performs as expected.*

Through this planned testing approach, the *Gamata Seva* system aims to ensure a reliable and high-quality user experience before public deployment.

With a strong technical foundation established, the next step focuses on the **business model**, which defines how the platform will operate sustainably and deliver value to both rural service providers and customers across Sri Lanka.

BUSINESS MODEL

The Gamata Seva web application operates as a digital service bridge connecting rural service providers (such as carpenters, electricians, tailors, and tutors) with local and urban customers who seek reliable, affordable services. The business model is designed to be socially impactful while ensuring financial sustainability.

Value Proposition

- **For Customers:**
  A trusted, mobile-friendly platform to discover, compare, book, and review local service providers based on verified profiles, transparent pricing, real-time availability, and community ratings. It's eliminating reliance on word-of-mouth and reducing search friction.
- **For Service Providers:**
  A zero-barrier digital store that enables rural micro-entrepreneurs to build credibility, manage appointments, accept digital payments, and grow their client base without needing technical expertise or marketing budgets.
- **For Communities & Government:**
  A scalable tool to formalize informal labor, boost local economies, and advance digital inclusion in underserved regions. It's turning village skills into visible, valued services.

Revenue Model

Gamata Seva adopts a light-touch, ethical monetization strategy to ensure accessibility while generating sustainable income.

1. **Success-Based Commission (5–10%):**
   Charged only on completed and paid bookings, no fee for cancellations or failed transactions. This aligns platform incentives with user success.
2. **Freemium Provider Tiers:**
   - Basic: Free profile listing and booking management.
   - Premium (Rs. 100–200/month): Enhanced visibility (top of search), verified badge, analytics dashboard, and priority customer alerts.
3. **Localized Advertising:**
   Local agri-input shops, hardware stores, or vocational training centers can promote services to relevant provider segments (e.g., "New tool kits for carpenters in Galle").

4. **Public-Private Partnerships:**
   Collaborate with ICT Agency of Sri Lanka, Samurdhi, or NGOs to integrate Gamata Seva into national digital livelihood programs, potentially securing grant-based funding or co-branded rollout support.

Cost Structure

Key operational expenditures are kept lean through smart design:

- Cloud hosting & domain (using scalable, low-cost providers like AWS or local data centers)
- SMS/Notification costs (critical for low-internet users)
- Minimal marketing: Community-driven adoption via village leaders, co-op societies, and local youth tech ambassadors
- Maintenance & security: Automated updates, open-source frameworks, and volunteer developer support from universities
- Support: Tiered help system — FAQs + WhatsApp-based assistance (no call centers)

Target Market

- **Primary:**
    - Rural service providers aged 25–60 with skills but no digital presence
    - Semi-urban customers seeking reliable, affordable home services
- **Secondary:**
    - Urban migrants looking to support relatives' services back home
    - Local government units (LGUs) and development agencies seeking digital tools for rural employment programs
- **Tertiary:**
    - Micro-businesses (e.g., paint suppliers, tool shops) wanting to reach skilled laborers

Growth & Scalability Strategy

- **Phase 1 (Pilot):**
  Launch in One or Two districts with 200+ providers. Focus on user training, feedback loops, and offline onboarding (e.g., via village ICT centers).
- **Phase 2 (National):**
  Expand to all 9 provinces with Sinhala/Tamil language support, offline mode for low-connectivity areas, and integration with mobile money (e.g., FriMi, Genie).
- **Phase 3 (Innovation):**
  Introduce AI-powered matching (e.g., "You often book electricians — here's a top-rated one near you"), voice-based search for non-literate users, and skills certification partnerships with NVQ centers.

Social impact & National Alignment

Gamata Seva directly supports:

- **Smart Sri Lanka Initiative** – advancing digital infrastructure in rural areas
- **SDG 8 (Decent Work)** – formalizing informal labor and increasing income stability
- **SDG 10 (Reduced Inequalities)** – closing the urban-rural digital divide
- **Women's Economic Empowerment** – enabling female tailors, cooks, and tutors to work safely from home

By transforming invisible village talent into visible digital services, Gamata Seva doesn't just connect people. It revalues rural work in the digital age.

In essence, Gamata Seva is a self-sustaining social enterprise: it earns modest revenue to stay alive, but its true return is measured in jobs created, trust built, and villages empowered.

CONCLUSION

The Gamata Seva project was designed to address a critical challenge faced by rural Sri Lankan communities. The lack of digital connectivity between local service providers and customers. Through careful analysis, modeling, and design, this project demonstrates how technology can empower underrepresented workers and bring rural economies into the digital era.

By following an **Agile + Reuse-Oriented software process model**, the team ensured flexibility, iterative progress, and integration of existing reliable technologies such as payment gateways and Google Maps APIs. The detailed **requirement analysis**, **use case modeling**, **behavioral**, **structural**, and **architectural designs** laid a strong foundation for developing a practical, user-friendly system.

The **Repository Architecture** supports centralized data management, ensuring efficiency and scalability for future improvements. The design prototypes provide a clear vision of how the final system will look and function once developed.

Overall, Gamata Seva is more than a software project. It’s a social innovation aimed at reducing the urban–rural digital divide, creating new income opportunities, and promoting digital literacy across Sri Lanka. In the future, the platform could be enhanced with features like AI-based service recommendations, SMS booking options, and mobile application integration to reach even more communities.

In conclusion, the Gamata Seva system embodies the vision of a *digitally inclusive Sri Lanka*, empowering every village, one service at a time.

- END -