**Project Design Phase**

**Problem Solution Fit**

|  |  |
| --- | --- |
| **Date** | 25 JUNE 2025 |
| **Team ID** | LTVIP2025TMID31613 |
| **Project Name** | Garage Management System |
| **Maximum Marks** |  |

**Problem – Solution Fit :**

**Problem Statement:**

* Manual booking and customer record tracking.
* Inefficient inventory management for spare parts and tools.
* Lack of real-time job tracking and service updates.
* Poor communication between mechanics, management, and customers.
* Paper-based billing and service history records.

**Solution Overview**:

* Provide **digital booking and appointment scheduling**.
* Automate **inventory tracking**, alerting when parts are low in stock.
* Offer a **real-time job progress dashboard** for internal use and customer updates.
* Enable **digital invoicing** and maintain a full service history.
* Integrate **customer communication tools** (SMS, email, or app notifications).

**Problem-Solution Fit**

When the system addresses the exact pain points of garage operators, it creates a seamless operational flow. Your project design should clearly map:

* **Each core problem → The feature that solves it**
* **Real-world user need → Usable, intuitive system function**

**Operational Pain Points**

* Manual scheduling leads to double bookings or missed appointments.
* Disorganized work orders cause delays in vehicle servicing.
* Limited visibility into job status for both managers and customers**.**

**Administrative Challenges**

* Inventory mismanagement, especially of fast-moving spare parts.
* Billing errors and lack of streamlined digital invoicing.
* Inconsistent record-keeping for customer history and vehicle data.

**Customer Experience Gaps**

* Poor communication about service progress or delays.
* No real-time updates or easy digital channels for customers to engage.
* Low customer retention due to impersonal or inefficient service.

**Staff Management Struggles**

* Workforce scheduling issues, especially with last-minute absenteeism.
* Lack of performance tracking for technicians or job efficiency.
* Training gaps leading to inconsistent service quality.

**Key Points:**

1. **User Persona Mapping**

* Small independent mechanics
* Multi-bay garage owners
* Fleet maintenance teams

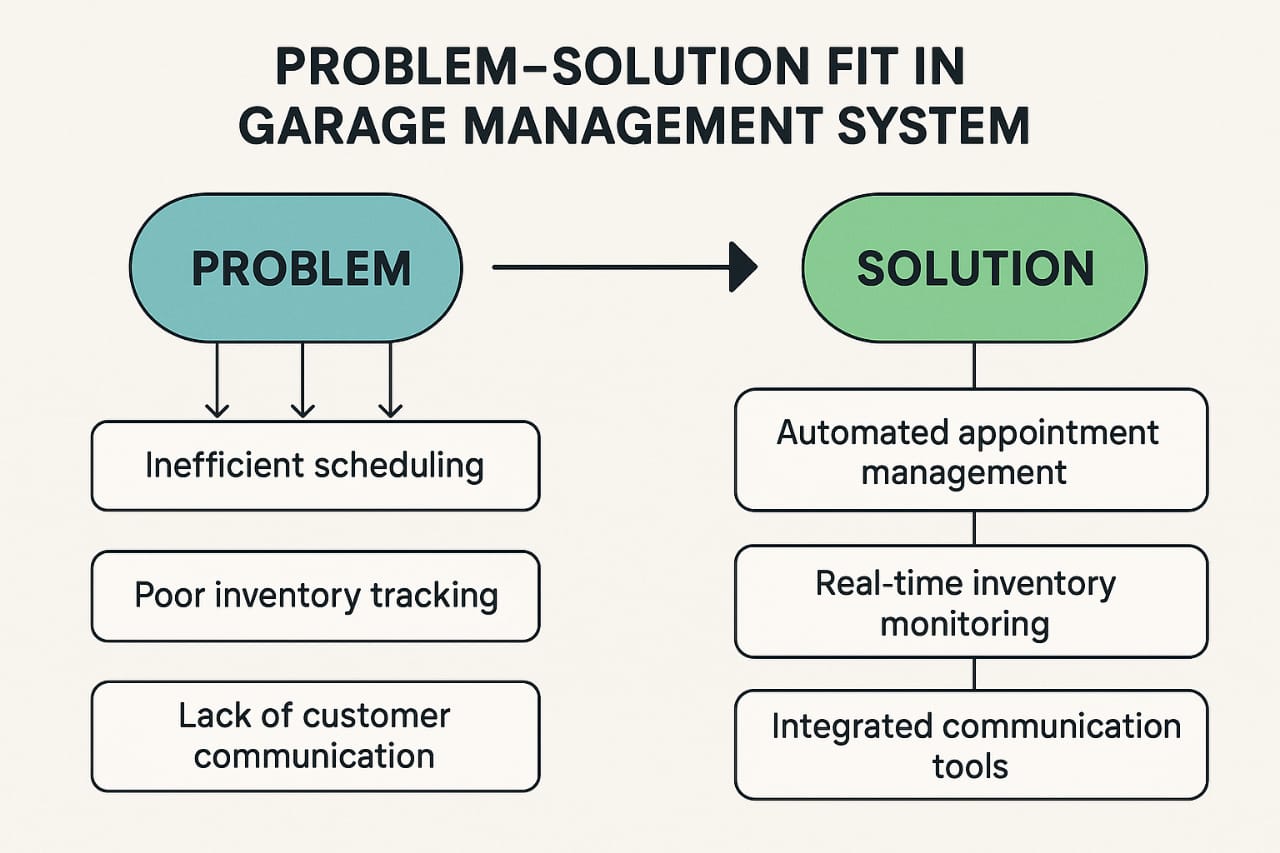
2. **Rapid Feedback Loops**

Once you've built a basic prototype or clickable UI:

* Share it with a few local garages
* Observe how they use it in real time
* Ask: “What do you wish this system could do?”

Their reactions will show you what resonates and what falls flat. This is your fastest path to alignment with real-world need

Diagram:



**Project Design Phase**

**Proposed Solution**

|  |  |
| --- | --- |
| **Date** | 25 JUNE 2025 |
| **Team ID** | LTVIP2025TMID31613 |
| **Project Name** | Garage Management System |
| **Maximum Marks** |  |

**Proposed Solution:**

**Proposed Solution for a Garage Management System**

**1. Core Modules:**

**1.Core Modules:**

* Customer Management: Store customer profiles, vehicle history, and contact details.
* Vehicle Intake & Service Scheduling: Digital intake forms, service appointment calendar, and reminders.
* Job Card Management: Auto-generate job cards with repair details, parts used, labor costs, and technician assignments.
* Inventory Management: Track parts and supplies, manage stock levels, reorder alerts.
* Billing & Invoicing: Integrated GST-compliant invoicing, discounts, and multiple payment modes.
* Technician Management: Track technician workload, shift scheduling, and performance analytics.
* Reports & Analytics: Daily income reports, parts usage, service trends, and customer feedback analysis.

**2. Technology Stack (example):**

* Frontend: React.js or Angular
* Backend: Node.js with Express or Django
* Database: PostgreSQL or MongoDB
* Hosting: Cloud-based (AWS, Azure, or Firebase)

**3. Key Features:**

* Mobile-responsive interface
* SMS/Email notifications
* Data backup and recovery
* Role-based access (admin, technician, receptionist)
* Barcode scanning for inventory

**4. Optional Add-ons:**

* Customer app for booking & tracking
* Loyalty points system
* Integration with insurance claim APIs
* Vehicle diagnostics integration (OBD data)

**Project Design Phase**

**Solution Architecure**

|  |  |
| --- | --- |
| **Date** | 25 JUNE 2025 |
| **Team ID** | LTVIP2025TMID31613 |
| **Project Name** | Garage Management System |
| **Maximum Marks** |  |

**Solution Architecture:**

**1.Presentation Layer (Frontend)**

**Handles user interactions.**

* Platforms: Web, Mobile App
* Technologies: React.js / Angular (Web), React Native / Flutter (Mobile)
* Features: UI for booking, service tracking, inventory updates, invoicing

**2. Application Layer (Backend Services)**

**Processes business logic and data flow.**

* API Gateway: Handles routing and security (e.g., API rate limiting, authentication)
* Business Services:
* Customer & Vehicle Management
* Service Request Handling
* Job Card Processing
* Billing & Payments
* Inventory Control

3. **Data Layer**

Stores and retrieves all necessary data.

* **Databases:**
  + PostgreSQL / MySQL (structured data)
  + MongoDB (flexible schemas for logs or service history)
* **File Storage:** AWS S3 or Azure Blob for invoice PDFs, vehicle images, etc.
* **aching:** Redis or Memcached for frequently accessed data

4. **Integration Layer**

Ensures communication with external systems.

* **Payment Gateway:** Razorpay, Stripe, etc.
* **SMS/Email Service:** Twilio, SendGrid
* **Insurance APIs** for claims management (optional)
* **OBD integration** for real-time diagnostics (optional)

5. **Security & Identity Management**

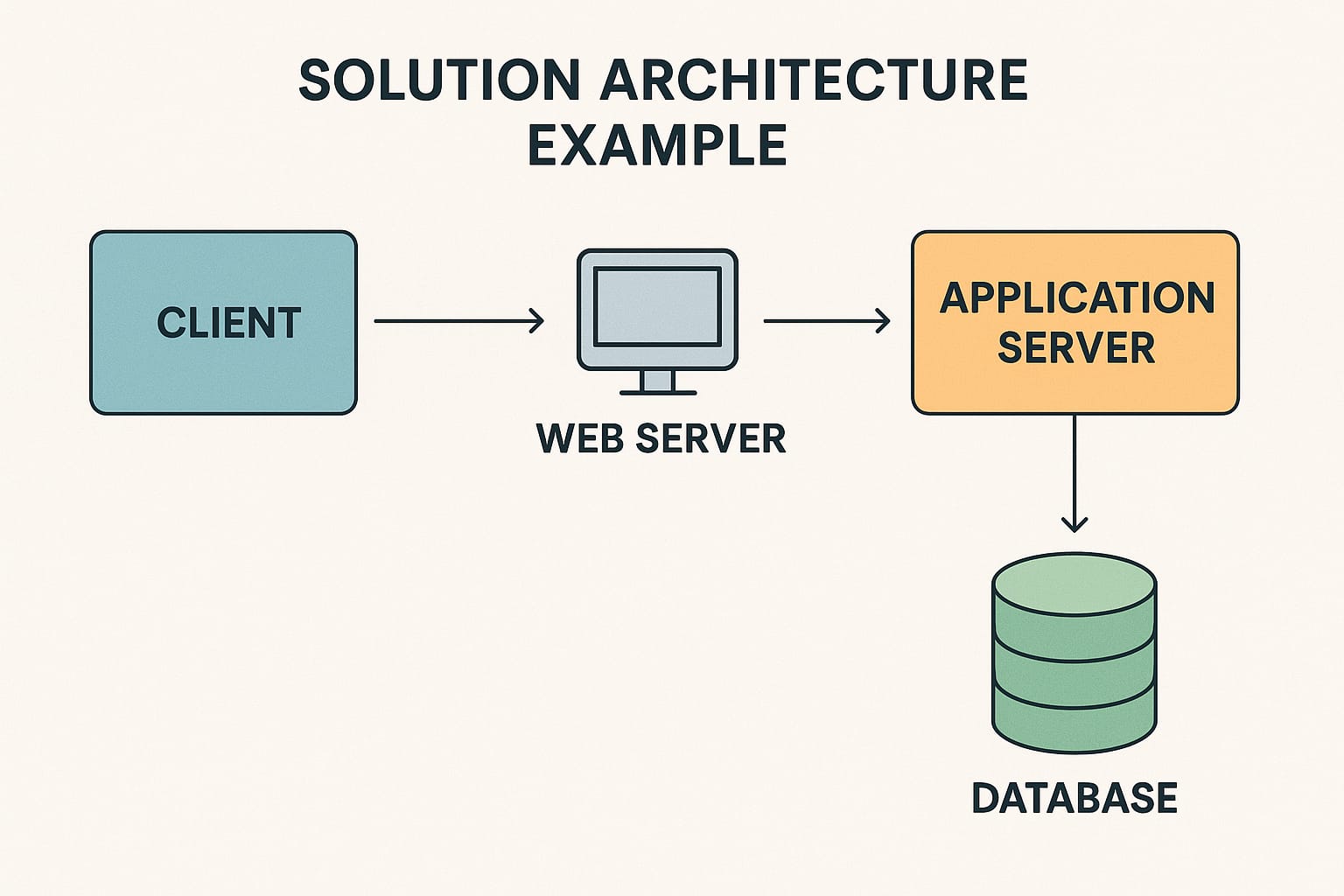
Protects user data and manages access.

* Role-Based Access Control (RBAC)
* OAuth2/JWT for secure login
* Encrypted communication via HTTPS

6. **Deployment & DevOps**

* **Containerization:** Docker
* **Orchestration:** Kubernetes
* **CI/CD Pipelines:** GitHub Actions, Jenkins
* **Monitoring:** Prometheus + Grafana, ELK Stack

**Solution Architecture Diagram** **Example:**

****