

Detailed Role-Based Functionality

1. Manager (Employee with elevated privileges)

- **Part Management:**
 - Add/Edit/Delete **Part** records (Name, Description, Category, Supplier).
 - Manage **StockQuantity** (manual adjustment or via inventory system).
 - Manage Part Pricing: Update **CurrentCost** on **Part**. This automatically creates a new record in **PartPriceHistory** with **EffectiveFrom** set to now. *This ensures future invoices use the correct historical cost.*
- **Repair Task Management:**
 - Define standard **RepairTasks** (Name, EstimatedDuration, DefaultLaborCost).
 - Link **RepairTasks** to **Parts** using **RepairTaskPart** (specifying which parts are typically required for this task, without quantity).
- **Work Order Creation & Assignment:**
 - Create a new **WorkOrder** for a specific **Vehicle**.
 - Assign one or more **Employees** to the **WorkOrder** via **WorkOrderEmployee** (specifying **HoursWorked** - initially estimated, later updated). *System should check employee availability (optional feature).*
 - Add specific **RepairTasks** to the **WorkOrder** via **WorkOrderRepairTask**. When added, the system captures the current **DefaultLaborCost** from the **RepairTask** and stores it as **LaborCostAtUse** in the linking table. *This locks in the labor rate at the time of assignment, preventing future price increases from affecting this order.*
 - Add specific **Parts** to the **WorkOrder** via **WorkOrderPart**. When added, the system captures the current **CurrentCost** from the **Part** and stores it as **UnitPriceAtUse** in the linking table. *This locks in the part cost at the time of use, ensuring accurate invoicing even if prices change later.* The **QuantityUsed** is also recorded here.
- **Invoice Generation:**
 - Trigger invoice creation when a **WorkOrder** is marked as **Completed**.
 - Calculate **Subtotal** = Sum of (**QuantityUsed** * **UnitPriceAtUse**) for all **WorkOrderPart** + Sum of (**LaborCostAtUse** for each **WorkOrderRepairTask**).
 - Apply **Discount** (if any).
 - Calculate **TaxAmount** = **Subtotal** * **TaxRate**.
 - Set **TotalAmount** = **Subtotal** - **Discount** + **TaxAmount**.
 - Set **PaymentStatus** to **Pending** and **IssuedAt** to current datetime.
 - Critical Rule: Prices (**UnitPriceAtUse**, **LaborCostAtUse**) are captured at the time of adding the item to the **WorkOrder**, not at invoice generation. This protects against price fluctuations and matches your requirement: *"the price increase... will not increase"*.

2. Employee (Mechanic / Laborer)

- **Work Order Management:**

- View assigned **WorkOrders** (filtered by their **EmployeeId** in **WorkOrderEmployee**).
- Update the **State** of a **WorkOrder** they are assigned to:
 - **Scheduled** -> **InProgress** (when starting work).
 - **InProgress** -> **Completed** (when finished).
 - **InProgress** -> **Canceled** (if work cannot be completed, requires manager approval?).
- **Record Hours Worked:** Update the **HoursWorked** field in **WorkOrderEmployee** for their assignment. This directly impacts the final labor cost calculation on the invoice.
- **Part Usage Tracking:**
 - Record the actual **QuantityUsed** for **Parts** in **WorkOrderPart** as they are consumed during the repair. *This updates the **StockQuantity** in the **Part** table automatically.*

3. Customer

- **Account Management:**
 - Register/Login via the **User** system.
 - Manage personal profile (linked to **Customer** record).
- **Vehicle Management:**
 - Add/Edit/Delete **Vehicle** records associated with their account.
- **Service Request & Tracking:**
 - Initiate a service request (creates a **WorkOrder** or **Appointment**).
 - Track Status: View the current **State** of their **WorkOrder** (Scheduled, InProgress, Completed, Canceled).
 - View Invoice: Access the generated **Invoice** for completed **WorkOrders**.
 - (Future) Schedule Appointments: Book a specific time slot via the **Appointment** system.
 - (Future) Notifications: Receive alerts (email/push) for status changes or appointment reminders.

Key Business Logic & Rules (Derived from ERD & Description)

1. **Price Locking:** All costs (**UnitPriceAtUse**, **LaborCostAtUse**) are captured and frozen at the moment a **Part** or **RepairTask** is added to a **WorkOrder**. This ensures billing accuracy regardless of future price changes.
2. **Inventory Management:** **StockQuantity** in **Part** is decremented when **QuantityUsed** is recorded in **WorkOrderPart** during the repair process.
3. **Labor Cost Calculation:** The total labor cost for a **WorkOrder** is the sum of (**HoursWorked** * **HourlyRate** at the time of work) for each assigned **Employee**. The **HourlyRate** used should be retrieved from **EmployeeSalaryHistory** based on the **WorkOrder's StartAt** date to ensure historical accuracy. *Note: Your ERD doesn't explicitly*

link `WorkOrderEmployee.HoursWorked` to `EmployeeSalaryHistory.HourlyRate` directly; this logic needs to be implemented in the application layer.

4. **Work Order State Transitions:** Defined workflow: `Scheduled` -> `InProgress` -> `Completed` or `Canceled`. Only `Employees` can change the state from `Scheduled` to `InProgress` and onwards.
 5. **Invoice Trigger:** Invoices are generated only when a `WorkOrder` reaches the `Completed` state. The `Invoice` record is linked 1:1 to the `WorkOrder`.
-

Workflow Summary

1. Customer Signs Up -> Creates `User` + `Customer` record.
2. Customer Adds Vehicle -> Creates `Vehicle` record linked to `Customer`.
3. Manager Creates Work Order -> Selects `Vehicle`, assigns `Employees`, adds `RepairTasks` (capturing `LaborCostAtUse`), adds `Parts` (capturing `UnitPriceAtUse`).
4. Employee Starts Work -> Changes `WorkOrder.State` to `InProgress`, records `HoursWorked` and `QuantityUsed`.
5. Employee Completes Work -> Changes `WorkOrder.State` to `Completed`.
6. System Generates Invoice -> Calculates `Subtotal`, applies `Discount`, calculates `Tax`, sets `TotalAmount`, `PaymentStatus=Pending`, `IssuedAt=Now`.
7. Customer Views Status/Invoice -> Can see `WorkOrder.State` and access the `Invoice`.