ICS213 Database Management Systems

PROJECT REPORT

VEHICLE SERVICE CENTER MANAGEMENT SYSTEM

DATE: 5 NOVEMBER 2025

MEMBERS:

- LAKSHYA DUBEY **2024BCS0339**
- ARMAAN Z HUSSAIN **2024BCS0323**
- S R JAI VISVAS VARSHAN **2024BCD0075**

INDEX

Database Schema

The database is structured into four main categories:

1. **Master Tables:** Lookup tables for predefined values (e.g., Roles, Vehicle Brands).
2. **Actor Tables:** Tables defining the users and their specific roles (e.g., Customer, Employee, Mechanic).
3. **Core Object Tables:** Core business entities (e.g., Vehicle, SparePart).
4. **Transactional Tables:** Tables that record business activities (e.g., Service, Invoices).

Entities (Tables)

1. Master Tables

- **`masters_master_role`**
  - `id` (Primary Key)
  - `name`
- **`masters_master_pay_grade_level`**
  - `id` (Primary Key)
  - `level`
  - `base_salary`
  - `description`
  - `role_id` (ForeignKey to `masters_master_role`)
- **`masters_master_expertise_area`**
  - `id` (Primary Key)
  - `name`
  - `description`
  - `role_id` (ForeignKey to `masters_master_role`)
- **`masters_master_vehicle_brand`**
  - `id` (Primary Key)
  - `name`
- **`masters_master_vehicle_category`**
  - `id` (Primary Key)
  - `name`
  - `licence_type`
  - `wheels`
- **`masters_master_vehicle_model`**
  - `id` (Primary Key)
  - `name`
  - `engine`
  - `weight`
  - `description`
  - `brand_id` (ForeignKey to `masters_master_vehicle_brand`)
  - `category_id` (ForeignKey to `masters_master_vehicle_category`)
- **`masters_master_spare_part_type`**
  - `id` (Primary Key)
  - `name`
  - `description`

2. Actor Tables

- **`auth_user`** (Base User Model)
  - `id` (Primary Key)
  - `username`
  - `first_name`
  - `last_name`
  - `email`
  - `password`
  - `is_staff`
  - `is_active`
- **`core_customer`** (Profile)
  - `user_id` (Primary Key, OneToOne to `auth_user`)
  - `mobile_number`
  - `address`
  - `description`
- **`core_employee`** (Base Employee Profile)
  - `id` (Primary Key)
  - `user_id` (OneToOne to `auth_user`)
  - `role_id` (ForeignKey to `masters_master_role`)
  - `pay_grade_id` (ForeignKey to `masters_master_pay_grade_level`)
  - `DOB`
  - `Address`
  - `mobile_number`
  - `shift`
- **`core_mechanic`** (Employee Sub-type)
  - `employee_id` (Primary Key, OneToOne to `core_employee`)
  - `expertise_area_id` (ForeignKey to `masters_master_expertise_area`)
  - `years_of_experience`
  - `certifications`
- **`core_accountant`** (Employee Sub-type)
  - `employee_id` (Primary Key, OneToOne to `core_employee`)
  - `education`
- **`core_cashier`** (Employee Sub-type)
  - `employee_id` (Primary Key, OneToOne to `core_employee`)
  - `counter_number`
- **`core_advisor`** (Employee Sub-type)
  - `employee_id` (Primary Key, OneToOne to `core_employee`)
  - `education_desc`

3. Core Object Tables

- **`core_vehicle`**
  - `id` (Primary Key)
  - `customer_id` (ForeignKey to `core_customer`)
  - `model_id` (ForeignKey to `masters_master_vehicle_model`)
  - `registration_number`
  - `chassis_number`
  - `year`
  - `description`

- **core_sparepart**
  - id (Primary Key)
  - vehicle_model_id (ForeignKey to masters_master_vehicle_model)
  - type_id (ForeignKey to masters_master_spare_part_type)
  - name
  - quantity_in_stock
  - price

4. Transactional Tables

- **core_service**
  - id (Primary Key)
  - customer_id (ForeignKey to core_customer)
  - vehicle_id (ForeignKey to core_vehicle)
  - mechanic_id (ForeignKey to core_mechanic)
  - status
  - description_of_service
  - labor_cost
  - service_date
- **core_servicepart** (Many-to-Many join table)
  - id (Primary Key)
  - service_id (ForeignKey to core_service)
  - part_id (ForeignKey to core_sparepart)
  - quantity
  - unit_price
- **core_serviceinvoice**
  - id (Primary Key)
  - service_id (OneToOne to core_service)
  - cashier_id (ForeignKey to core_cashier)
  - date
  - amount
  - payment_method
  - status
- **core_inventoryinvoice**
  - id (Primary Key)
  - spare_part_id (ForeignKey to core_sparepart)
  - supplier
  - quantity_received
  - date
  - cost

Relationships and Cardinality

This section details the explicit relationships between the tables.

One-to-One Relationships

- **`auth_user` (1) → (1) `core_customer`**: A base user can be one customer. The `user_id` in `core_customer` is both a Primary Key and a Foreign Key.
- **`auth_user` (1) → (1) `core_employee`**: A base user can be one employee. The `user_id` in `core_employee` is a Unique Key.
- **`core_employee` (1) → (1) `core_mechanic`**: An employee can be a mechanic. The `employee_id` is the Primary Key.
- **`core_employee` (1) → (1) `core_accountant`**: An employee can be an accountant. The `employee_id` is the Primary Key.
- **`core_employee` (1) → (1) `core_cashier`**: An employee can be a cashier. The `employee_id` is the Primary Key.
- **`core_employee` (1) → (1) `core_advisor`**: An employee can be an advisor. The `employee_id` is the Primary Key.
- **`core_service` (1) → (1) `core_serviceinvoice`**: A service can have exactly one invoice. The `service_id` in `core_serviceinvoice` is a Unique Key.

One-to-Many Relationships

- **`masters_master_role` (1) → (M) `core_employee`**: One role (e.g., "Mechanic") can be assigned to many employees.
- **`masters_master_role` (1) → (M) `masters_master_pay_grade_level`**: One role can have multiple pay grade levels.
- **`masters_master_role` (1) → (M) `masters_master_expertise_area`**: One role (like "Mechanic") can have multiple expertise areas (e.g., "Engine," "Transmission").
- **`masters_master_pay_grade_level` (1) → (M) `core_employee`**: One pay grade level can be assigned to many employees.
- **`masters_master_expertise_area` (1) → (M) `core_mechanic`**: One expertise area can be held by many mechanics.
- **`masters_master_vehicle_brand` (1) → (M) `masters_master_vehicle_model`**: One brand (e.g., "Toyota") can have many models.
- **`masters_master_vehicle_category` (1) → (M) `masters_master_vehicle_model`**: One category (e.g., "SUV") can include many models.
- **`masters_master_vehicle_model` (1) → (M) `core_vehicle`**: One vehicle model (e.g., "Corolla") can be owned by many customers (as many vehicles).
- **`masters_master_vehicle_model` (1) → (M) `core_sparepart`**: One vehicle model can be associated with many different spare parts.
- **`masters_master_spare_part_type` (1) → (M) `core_sparepart`**: One part type (e.g., "Filter") can apply to many spare parts.
- **`core_customer` (1) → (M) `core_vehicle`**: One customer can own multiple vehicles.
- **`core_customer` (1) → (M) `core_service`**: One customer can have multiple services over time.
- **`core_vehicle` (1) → (M) `core_service`**: One vehicle can undergo many services.
- **`core_mechanic` (1) → (M) `core_service`**: One mechanic can perform many services.

- **`core_cashier` (1) → (M) `core_serviceinvoice`**: One cashier can process many invoices.
- **`core_sparepart` (1) → (M) `core_inventoryinvoice`**: One spare part can be restocked through many inventory invoices.

Many-to-Many Relationships

- **`core_service` (M) ↔ (N) `core_sparepart`**: This relationship is managed by the **`core_servicepart`** join table.
    - One service (`core_service`) can use many spare parts (`core_sparepart`).
    - One spare part (`core_sparepart`) can be used in many different services.

Database Triggers and Automated Business Logic

To ensure data integrity and automate key business processes, the database implements several triggers. These are stored procedures that automatically execute (fire) when specific data manipulation (DML) events—such as `INSERT`, `UPDATE`, or `DELETE`—occur on a table.

This section details the triggers implemented in the system, their purpose, and their SQL definition.

1. Trigger: Inventory Control on Service Usage

- **Purpose:** To automatically decrement or adjust the `quantity_in_stock` in the `core_sparepart` table whenever parts are used, returned, or modified in a service record. This is crucial for maintaining accurate inventory levels.
- **Events:** `AFTER INSERT`, `AFTER UPDATE`, `AFTER DELETE` on `core_servicepart`.

SQL Implementation

On `INSERT` (Part added to service): Decreases stock by the quantity of parts used.

```
DELIMITER $$

CREATE TRIGGER trg_UpdateStockOnServicePart
AFTER INSERT ON core_servicepart
FOR EACH ROW
BEGIN
    UPDATE core_sparepart
    SET quantity_in_stock = quantity_in_stock - NEW.quantity
    WHERE id = NEW.part_id;
END$$

DELIMITER ;
```

On `DELETE` (Part removed from service): Increases stock, returning the parts to inventory.

```
DELIMITER $$

CREATE TRIGGER trg_UpdateStockOnServicePartDelete
AFTER DELETE ON core_servicepart
FOR EACH ROW
BEGIN
    UPDATE core_sparepart
    SET quantity_in_stock = quantity_in_stock + OLD.quantity
    WHERE id = OLD.part_id;
END$$

DELIMITER ;
```

On `UPDATE` (Part quantity changed in service): Adjusts the stock based on the *difference* between the old and new quantities.

```
DELIMITER $$

CREATE TRIGGER trg_UpdateStockOnServicePartUpdate
AFTER UPDATE ON core_servicepart
FOR EACH ROW
BEGIN
    UPDATE core_sparepart
    SET quantity_in_stock = quantity_in_stock - (NEW.quantity -
OLD.quantity)
    WHERE id = NEW.part_id;
END$$

DELIMITER ;
```

2 Trigger: Inventory Restocking

- **Purpose:** To automatically increment the `quantity_in_stock` in the `core_sparepart` table when a new shipment of parts is logged via an inventory invoice.
- **Event:** `AFTER INSERT` on `core_inventoryinvoice`.

SQL Implementation

```
DELIMITER $$

CREATE TRIGGER trg_RestockInventory
AFTER INSERT ON core_inventoryinvoice
FOR EACH ROW
BEGIN
    UPDATE core_sparepart
    SET quantity_in_stock = quantity_in_stock + NEW.quantity_received
    WHERE id = NEW.spare_part_id;
END$$

DELIMITER ;
```

3. Trigger: Service Part Price Snapshot

- **Purpose:** To "snapshot" the current price of a spare part at the moment it is added to a service. This ensures that the `core_servicepart.unit_price` is fixed and will not change if the part's price in `core_sparepart` is updated later, guaranteeing historical accuracy for invoices.
- **Event:** `BEFORE INSERT` on `core_servicepart`.

SQL Implementation

```
DELIMITER $$

CREATE TRIGGER trg_SetServicePartPrice
BEFORE INSERT ON core_servicepart
FOR EACH ROW
BEGIN
    DECLARE v_current_price DECIMAL(10, 2);

    -- Fetch the current price from the main parts table
    SELECT price INTO v_current_price
    FROM core_sparepart
    WHERE id = NEW.part_id;

    -- Set the unit_price on the new service part record
    SET NEW.unit_price = v_current_price;
END$$

DELIMITER ;
```

4. Trigger: Automatic Invoice Amount Calculation

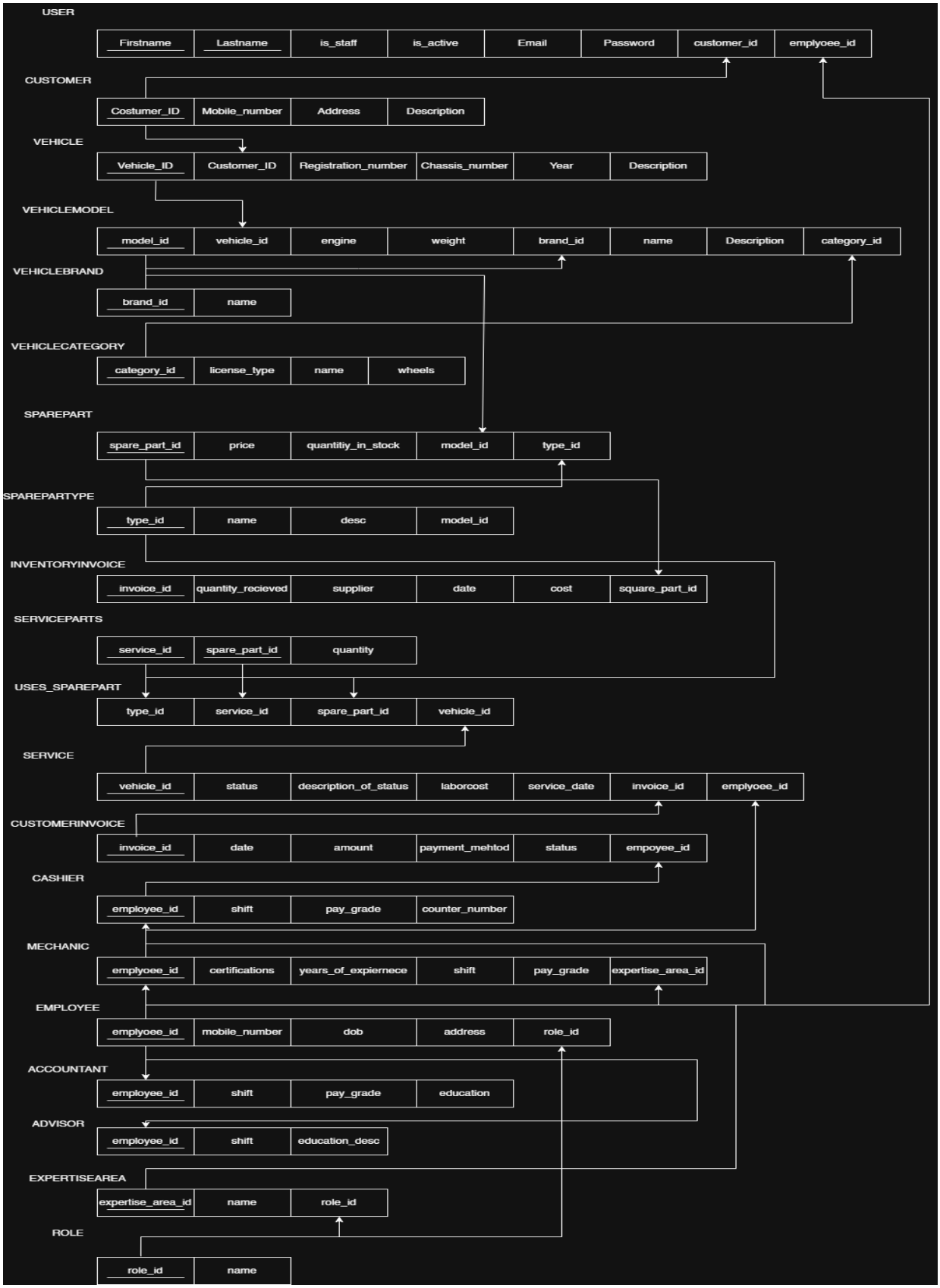
- **Purpose:** To automatically calculate the final `amount` for a `core_serviceinvoice` by summing the `labor_cost` from the parent `core_service` and the total cost of all parts (quantity * unit_price) from the associated `core_servicepart` records.
- **Event:** `BEFORE INSERT` on `core_serviceinvoice`.

SQL Implementation

```
DELIMITER $$

CREATE TRIGGER trg_CalculateInvoiceAmount
BEFORE INSERT ON core_serviceinvoice
FOR EACH ROW
BEGIN
    DECLARE v_labor_cost DECIMAL(10, 2);
    DECLARE v_parts_total DECIMAL(10, 2);

    -- 1. Get the labor cost from the related service
    SELECT labor_cost INTO v_labor_cost
    FROM core_service
    WHERE id = NEW.service_id;

    -- 2. Calculate the total cost of all parts for that service
    SELECT SUM(quantity * unit_price) INTO v_parts_total
    FROM core_servicepart
    WHERE service_id = NEW.service_id;
```

```
    -- 3. Set the amount on the new invoice
    -- (IFNULL handles cases with no parts)
    SET NEW.amount = v_labor_cost + IFNULL(v_parts_total, 0);
END$$

DELIMITER ;
```

SCHEMA DIAGRAM



SQL QUERIES

```
CREATE DATABASE IF NOT EXISTS vsc_erp;
USE vsc_erp;

--
-- Master Data Tables (Lookup Tables)
--

CREATE TABLE masters_master_role (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(50) NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY name (name)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE masters_master_pay_grade_level (
  id bigint NOT NULL AUTO_INCREMENT,
  level varchar(10) NOT NULL,
  base_salary decimal(10,2) NOT NULL,
  description longtext,
  role_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY Masters_master_pay_grade_level_role_id_level_3194a9d6_uniq
(role_id,level),
  CONSTRAINT Masters_master_pay_g_role_id_691cc7b2_fk_Masters_m FOREIGN KEY
(role_id) REFERENCES masters_master_role (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE masters_master_expertise_area (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(100) NOT NULL,
  description longtext,
  role_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY name (name),
  KEY Masters_master_exper_role_id_82dc56c3_fk_Masters_m (role_id),
  CONSTRAINT Masters_master_exper_role_id_82dc56c3_fk_Masters_m FOREIGN KEY
(role_id) REFERENCES masters_master_role (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE masters_master_vehicle_brand (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(50) NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY name (name)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```sql
CREATE TABLE masters_master_vehicle_category (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(50) NOT NULL,
  licence_type varchar(20) NOT NULL,
  wheels smallint unsigned NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY name (name),
  CONSTRAINT masters_master_vehicle_category_chk_1 CHECK ((wheels >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE masters_master_vehicle_model (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(100) NOT NULL,
  engine varchar(50) NOT NULL,
  weight int unsigned DEFAULT NULL,
  description longtext NOT NULL,
  brand_id bigint NOT NULL,
  category_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY Masters_master_vehicle_model_brand_id_name_c8eaf8de_uniq
(brand_id,name),
  KEY Masters_master_vehic_category_id_46c005b1_fk_Masters_m (category_id),
  CONSTRAINT Masters_master_vehic_brand_id_2c81c645_fk_Masters_m FOREIGN KEY
(brand_id) REFERENCES masters_master_vehicle_brand (id),
  CONSTRAINT Masters_master_vehic_category_id_46c005b1_fk_Masters_m FOREIGN KEY
(category_id) REFERENCES masters_master_vehicle_category (id),
  CONSTRAINT masters_master_vehicle_model_chk_1 CHECK ((weight >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE masters_master_spare_part_type (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(50) NOT NULL,
  description longtext,
  PRIMARY KEY (id),
  UNIQUE KEY name (name)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```sql
--
-- User and Actor Tables (Users, Customers, Employees)
--

CREATE TABLE auth_user (
  id int NOT NULL AUTO_INCREMENT,
  password varchar(128) NOT NULL,
  last_login datetime(6) DEFAULT NULL,
  is_superuser tinyint(1) NOT NULL,
  username varchar(150) NOT NULL,
  first_name varchar(150) NOT NULL,
  last_name varchar(150) NOT NULL,
  email varchar(254) NOT NULL,
  is_staff tinyint(1) NOT NULL,
  is_active tinyint(1) NOT NULL,
  date_joined datetime(6) NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY username (username)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_customer (
  user_id int NOT NULL,
  mobile_number varchar(128) NOT NULL,
  address longtext,
  description longtext,
  PRIMARY KEY (user_id),
  CONSTRAINT CORE_customer_user_id_50d0b9cd_fk_auth_user_id FOREIGN KEY (user_id)
REFERENCES auth_user (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_employee (
  id bigint NOT NULL AUTO_INCREMENT,
  DOB date NOT NULL,
  Address longtext NOT NULL,
  mobile_number varchar(128) NOT NULL,
  shift varchar(10) DEFAULT NULL,
  pay_grade_id bigint DEFAULT NULL,
  role_id bigint DEFAULT NULL,
  user_id int NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY user_id (user_id),
  KEY CORE_employee_role_id_d059113d_fk_Masters_master_role_id (role_id),
  KEY CORE_employee_pay_grade_id_dcce3e6e (pay_grade_id),
  CONSTRAINT CORE_employee_pay_grade_id_dcce3e6e_fk_Masters_m FOREIGN KEY
(pay_grade_id) REFERENCES masters_master_pay_grade_level (id),
  CONSTRAINT CORE_employee_role_id_d059113d_fk_Masters_master_role_id FOREIGN KEY
(role_id) REFERENCES masters_master_role (id),
  CONSTRAINT CORE_employee_user_id_b261e990_fk_auth_user_id FOREIGN KEY (user_id)
REFERENCES auth_user (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
CREATE TABLE core_accountant (
  employee_id bigint NOT NULL,
  education longtext,
  PRIMARY KEY (employee_id),
  CONSTRAINT CORE_accountant_employee_id_1040ddd2_fk_CORE_employee_id FOREIGN KEY
(employee_id) REFERENCES core_employee (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_advisor (
  employee_id bigint NOT NULL,
  education_desc longtext,
  PRIMARY KEY (employee_id),
  CONSTRAINT CORE_advisor_employee_id_c11f806c_fk_CORE_employee_id FOREIGN KEY
(employee_id) REFERENCES core_employee (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_cashier (
  employee_id bigint NOT NULL,
  counter_number varchar(10) NOT NULL,
  PRIMARY KEY (employee_id),
  CONSTRAINT CORE_cashier_employee_id_82ba154b_fk_CORE_employee_id FOREIGN KEY
(employee_id) REFERENCES core_employee (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_mechanic (
  employee_id bigint NOT NULL,
  years_of_experience smallint unsigned NOT NULL,
  certifications longtext,
  expertise_area_id bigint NOT NULL,
  PRIMARY KEY (employee_id),
  KEY CORE_mechanic_expertise_area_id_5a7f3546_fk_Masters_m (expertise_area_id),
  CONSTRAINT CORE_mechanic_employee_id_45c7346e_fk_CORE_employee_id FOREIGN KEY
(employee_id) REFERENCES core_employee (id),
  CONSTRAINT CORE_mechanic_expertise_area_id_5a7f3546_fk_Masters_m FOREIGN KEY
(expertise_area_id) REFERENCES masters_master_expertise_area (id),
  CONSTRAINT core_mechanic_chk_1 CHECK ((years_of_experience >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
--
-- Core Business Object Tables (Vehicles, Parts)
--

CREATE TABLE core_vehicle (
  id bigint NOT NULL AUTO_INCREMENT,
  registration_number varchar(20) NOT NULL,
  chassis_number varchar(17) NOT NULL,
  year smallint unsigned NOT NULL,
  description longtext NOT NULL,
  customer_id int NOT NULL,
  model_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY registration_number (registration_number),
  UNIQUE KEY chassis_number (chassis_number),
  KEY CORE_vehicle_customer_id_370e1bb0_fk_CORE_customer_user_id (customer_id),
  KEY CORE_vehicle_model_id_79d8b126_fk_Masters_m (model_id),
  CONSTRAINT CORE_vehicle_customer_id_370e1bb0_fk_CORE_customer_user_id FOREIGN KEY
(customer_id) REFERENCES core_customer (user_id),
  CONSTRAINT CORE_vehicle_model_id_79d8b126_fk_Masters_m FOREIGN KEY (model_id)
REFERENCES masters_master_vehicle_model (id),
  CONSTRAINT core_vehicle_chk_1 CHECK ((year >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_sparepart (
  id bigint NOT NULL AUTO_INCREMENT,
  name varchar(100) NOT NULL,
  quantity_in_stock int unsigned NOT NULL,
  price decimal(10,2) NOT NULL,
  type_id bigint DEFAULT NULL,
  vehicle_model_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY CORE_sparepart_vehicle_model_id_name_32caaead_uniq
(vehicle_model_id,name),
  KEY CORE_sparepart_type_id_60c590bf_fk_Masters_m (type_id),
  CONSTRAINT CORE_sparepart_type_id_60c590bf_fk_Masters_m FOREIGN KEY (type_id)
REFERENCES masters_master_spare_part_type (id),
  CONSTRAINT CORE_sparepart_vehicle_model_id_9af94d87_fk_Masters_m FOREIGN KEY
(vehicle_model_id) REFERENCES masters_master_vehicle_model (id),
  CONSTRAINT core_sparepart_chk_1 CHECK ((quantity_in_stock >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
--
-- Transactional Tables (Services, Invoices)
--

CREATE TABLE core_service (
  id bigint NOT NULL AUTO_INCREMENT,
  status varchar(20) NOT NULL,
  description_of_service longtext NOT NULL,
  labor_cost decimal(10,2) NOT NULL,
  service_date datetime(6) NOT NULL,
  customer_id int NOT NULL,
  vehicle_id bigint NOT NULL,
  mechanic_id bigint DEFAULT NULL,
  PRIMARY KEY (id),
  KEY CORE_service_vehicle_id_79f35b9e_fk_CORE_vehicle_id (vehicle_id),
  KEY CORE_service_mechanic_id_9b8019d7_fk_CORE_mechanic_employee_id (mechanic_id),
  KEY CORE_service_customer_id_64546910_fk_CORE_customer_user_id (customer_id),
  CONSTRAINT CORE_service_customer_id_64546910_fk_CORE_customer_user_id FOREIGN KEY
(customer_id) REFERENCES core_customer (user_id),
  CONSTRAINT CORE_service_mechanic_id_9b8019d7_fk_CORE_mechanic_employee_id FOREIGN
KEY (mechanic_id) REFERENCES core_mechanic (employee_id),
  CONSTRAINT CORE_service_vehicle_id_79f35b9e_fk_CORE_vehicle_id FOREIGN KEY
(vehicle_id) REFERENCES core_vehicle (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_servicepart (
  id bigint NOT NULL AUTO_INCREMENT,
  quantity int unsigned NOT NULL,
  unit_price decimal(10,2) NOT NULL,
  service_id bigint NOT NULL,
  part_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY CORE_servicepart_service_id_part_id_d7a69a0e_uniq (service_id,part_id),
  KEY CORE_servicepart_part_id_4a1c2323_fk_CORE_sparepart_id (part_id),
  CONSTRAINT CORE_servicepart_part_id_4a1c2323_fk_CORE_sparepart_id FOREIGN KEY
(part_id) REFERENCES core_sparepart (id),
  CONSTRAINT CORE_servicepart_service_id_19b7f66e_fk_CORE_service_id FOREIGN KEY
(service_id) REFERENCES core_service (id),
  CONSTRAINT core_servicepart_chk_1 CHECK ((quantity >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_serviceinvoice (
  id bigint NOT NULL AUTO_INCREMENT,
  date datetime(6) NOT NULL,
  amount decimal(10,2) NOT NULL,
  payment_method varchar(10) NOT NULL,
  status varchar(10) NOT NULL,
  service_id bigint NOT NULL,
  cashier_id bigint NOT NULL,
  PRIMARY KEY (id),
  UNIQUE KEY service_id (service_id),
```

```
  KEY CORE_serviceinvoice_cashier_id_41d6c839_fk_CORE_cash (cashier_id),
  CONSTRAINT CORE_serviceinvoice_cashier_id_41d6c839_fk_CORE_cash FOREIGN KEY
(cashier_id) REFERENCES core_cashier (employee_id),
  CONSTRAINT CORE_serviceinvoice_service_id_068dcbf3_fk_CORE_service_id FOREIGN KEY
(service_id) REFERENCES core_service (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE core_inventoryinvoice (
  id bigint NOT NULL AUTO_INCREMENT,
  supplier varchar(100) NOT NULL,
  quantity_received int unsigned NOT NULL,
  date datetime(6) NOT NULL,
  cost decimal(10,2) NOT NULL,
  spare_part_id bigint NOT NULL,
  PRIMARY KEY (id),
  KEY CORE_inventoryinvoic_spare_part_id_91069a8a_fk_CORE_spar (spare_part_id),
  CONSTRAINT CORE_inventoryinvoic_spare_part_id_91069a8a_fk_CORE_spar FOREIGN KEY
(spare_part_id) REFERENCES core_sparepart (id),
  CONSTRAINT core_inventoryinvoice_chk_1 CHECK ((quantity_received >= 0))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

SOFTWARES USED

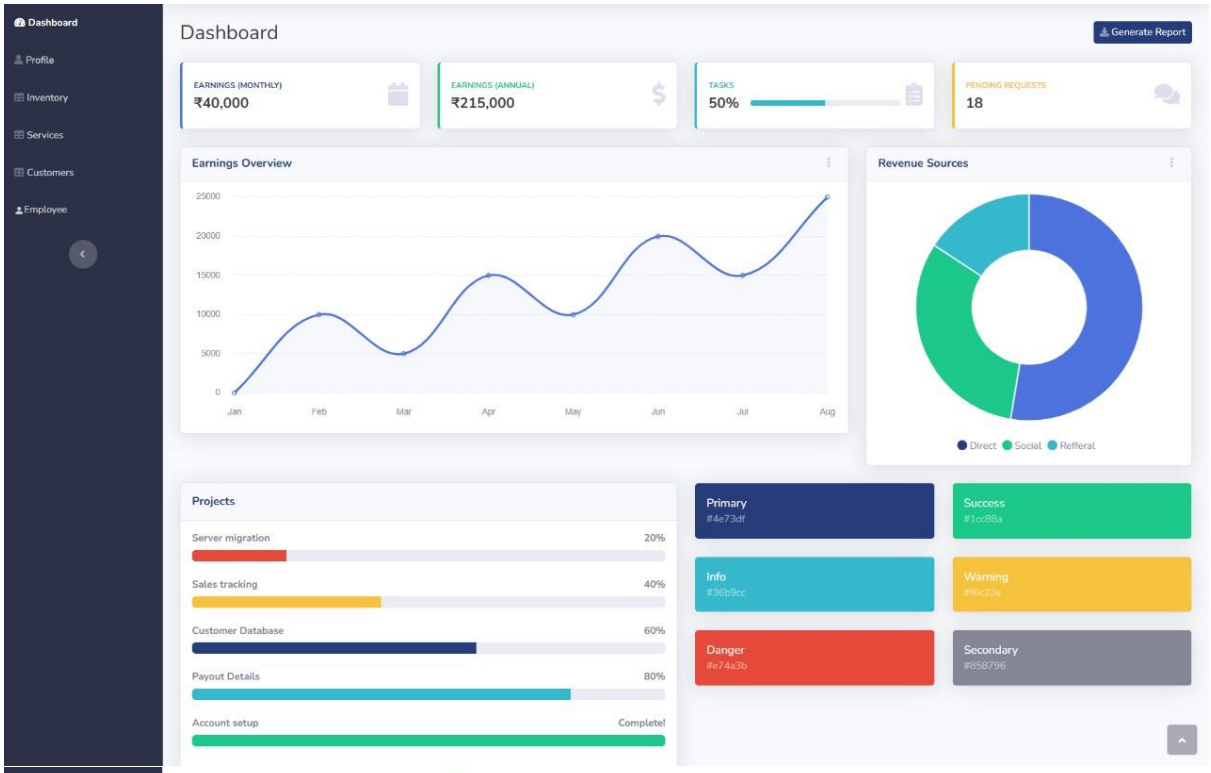
Backend

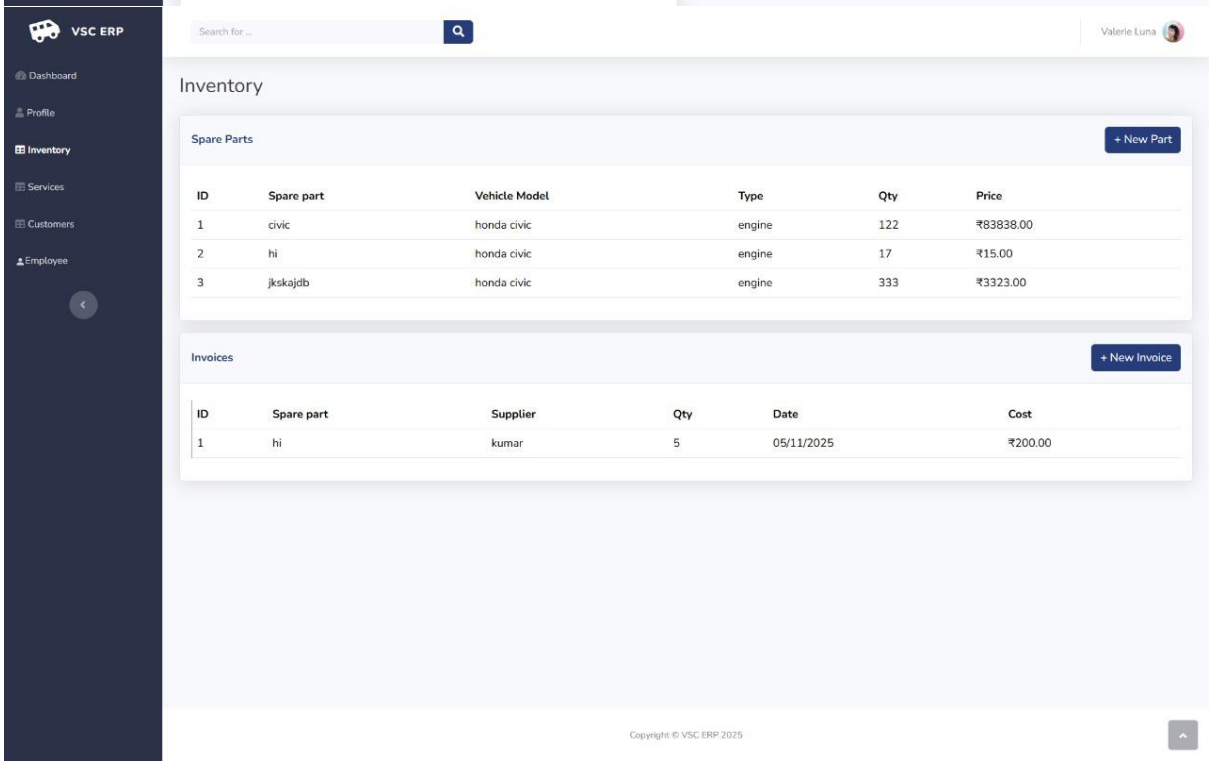
- Django (Web Framework)
- Django-formtools (For multi-step wizards
- Django-crispy-forms (For form styling)
- Crispy-bootstrap5 (To make crispy_forms use Bootstrap 5)
- Phonenumber_field (For phone number validation in models)
- Smart_selects (For chained dropdowns in models)
- MySQL

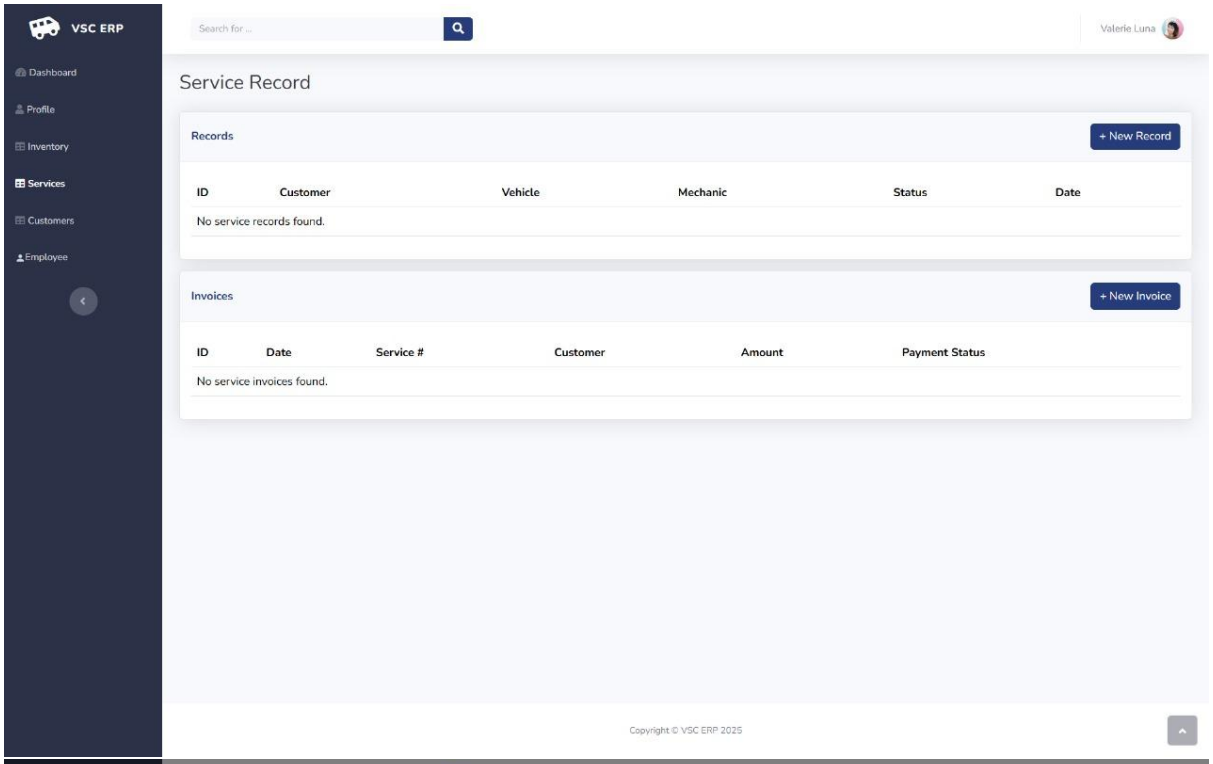
Frontend

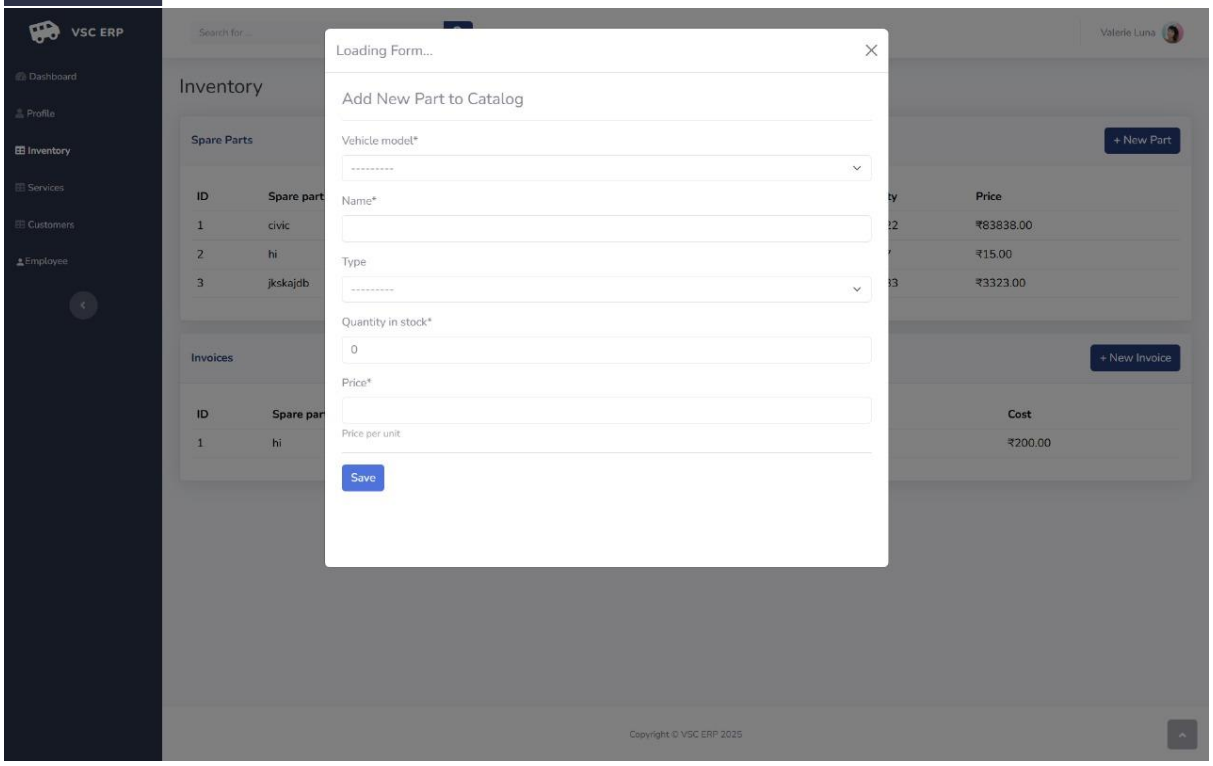
- HTML5
- CSS3
- JavaScript
- Bootstrap 5 (For styling, components, and modals)
- Popper.js (Required for Bootstrap modals)
- Chart.js (For the graphs on your dashboard)

SCREENSHOTS



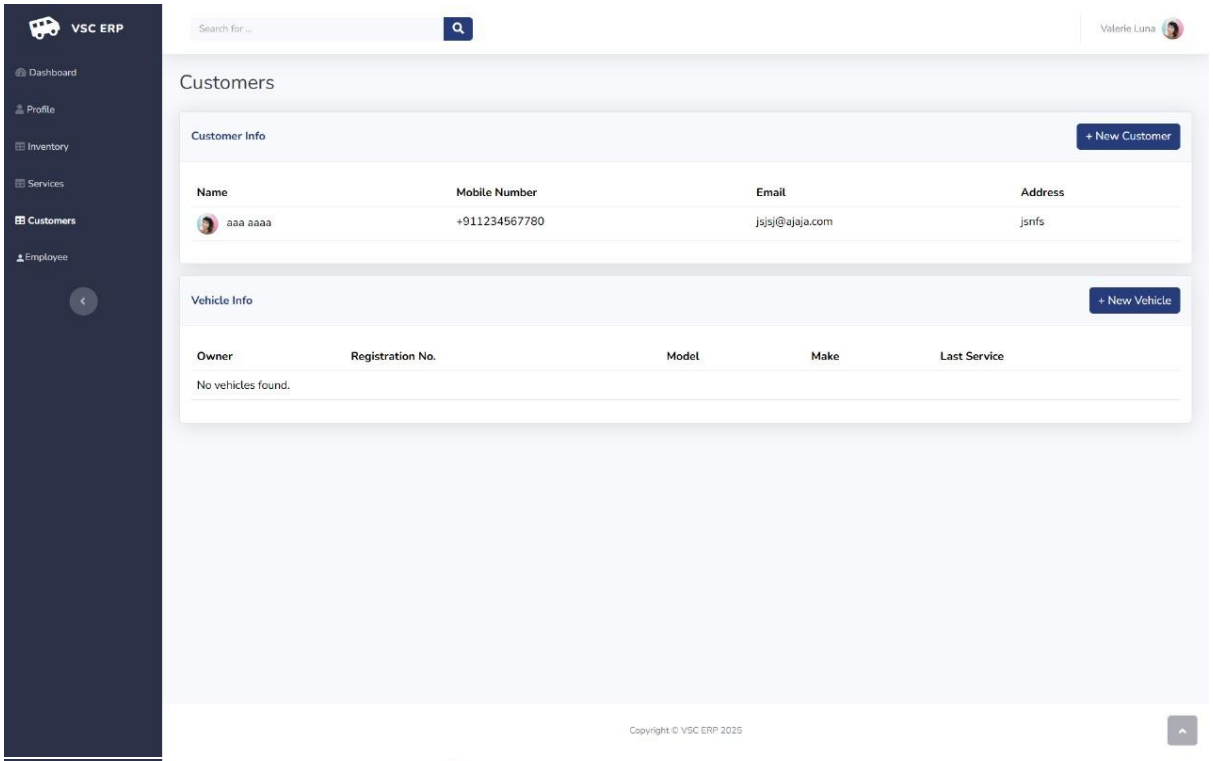


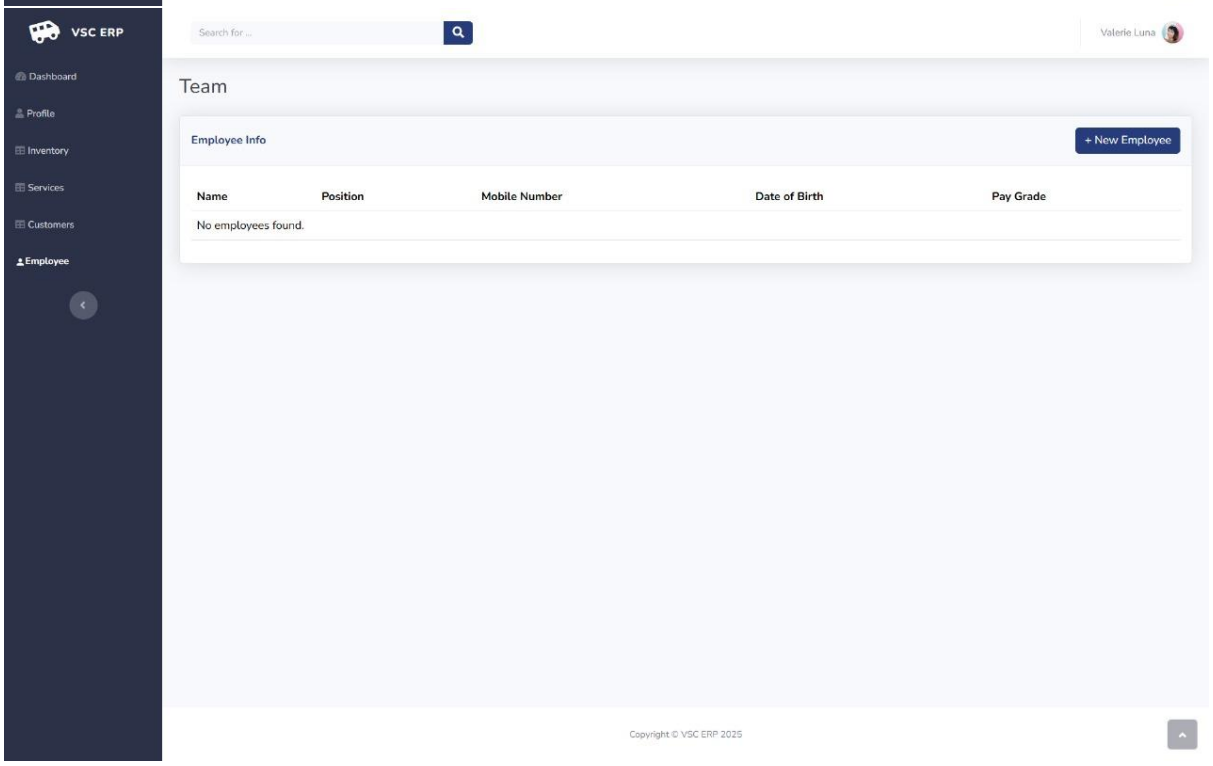

VSC ERP
Search for ...
Valerie Luna
Dashboard
Profile
Inventory
Services
Customers
Employee
Service Record
Records
+ New Record
ID
Customer
Vehicle
Mechanic
Status
Date
No service records found.
Invoices
+ New Invoice
ID
Date
Service #
Customer
Amount
Payment Status
No service invoices found.
Copyright © VSC ERP 2025

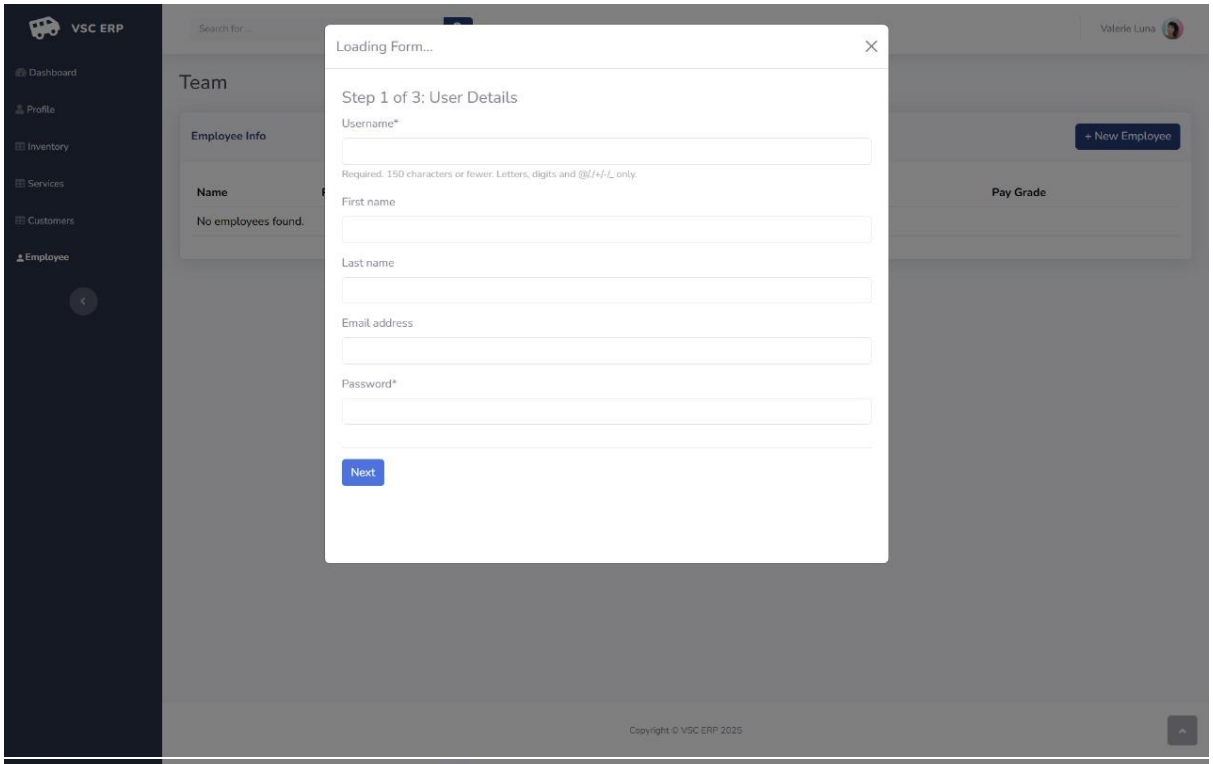

VSC ERP
Search for ...
Valerie Luna
Dashboard
Profile
Inventory
Services
Customers
Employee
Inventory
Spare Parts
+ New Part
ID
Spare part
Price
1
civic
₹83838.00
2
hi
₹15.00
3
jkskajdb
₹3323.00
Invoices
+ New Invoice
ID
Spare part
Cost
1
hi
₹200.00
Loading Form...
Add New Part to Catalog
Vehicle model*

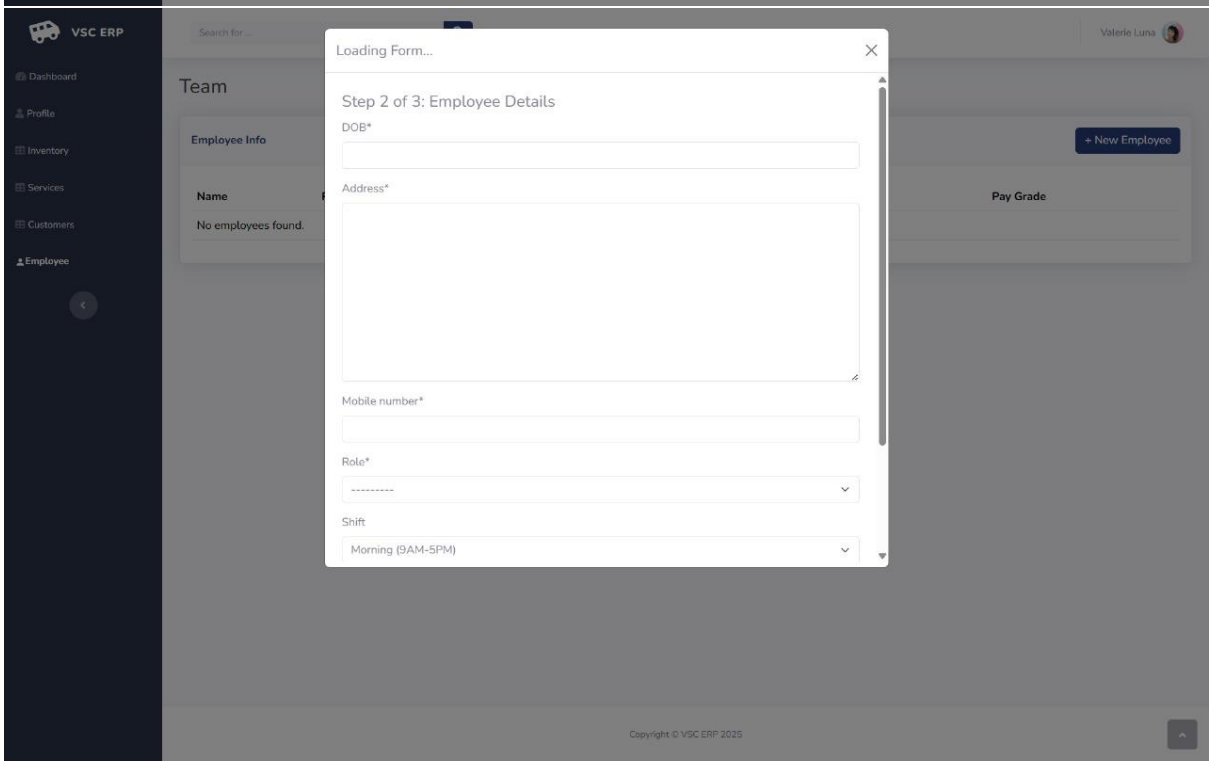
Name*
Type

Quantity in stock*
0
Price*
Price per unit
Save
Copyright © VSC ERP 2025


VSC ERP
Search for ...
Valerie Luna
Dashboard
Profile
Inventory
Services
Customers
Employee
Customers
Customer Info
+ New Customer
Name
Mobile Number
Email
Address
aaa aaaa
+911234567780
jsjsj@ajaja.com
jsnfs
Vehicle Info
+ New Vehicle
Owner
Registration No.
Model
Make
Last Service
No vehicles found.
Copyright © VSC ERP 2025


VSC ERP
Search for ...
Valerie Luna
Dashboard
Profile
Inventory
Services
Customers
Employee
Team
Employee Info
+ New Employee
Name
Position
Mobile Number
Date of Birth
Pay Grade
No employees found.
Copyright © VSC ERP 2025


VSC ERP
Dashboard
Profile
Inventory
Services
Customers
Employee
Search for...
Valerie Luna
Team
Employee Info
+ New Employee
Name
Pay Grade
No employees found.
Loading Form...
Step 1 of 3: User Details
Username*
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.
First name
Last name
Email address
Password*
Next
Copyright © VSC ERP 2025


VSC ERP
Dashboard
Profile
Inventory
Services
Customers
Employee
Search for...
Valerie Luna
Team
Employee Info
+ New Employee
Name
Pay Grade
No employees found.
Loading Form...
Step 2 of 3: Employee Details
DOB*
Address*
Mobile number*
Role*

Shift
Morning (9AM-5PM)
Copyright © VSC ERP 2025