Team Members

1. Rahul Googikoll [1AM19CS158]
2. Rahul Mishra [1AM19CS159]
3. Saurav Kumar [1AM19CS195]

Introduction

1.1 Project outline

This application is going to be used by:

1. Employees
2. Customer

This application is meant for users (Employees and Customers) who can access the information from database about car deals. Our objective is so that executives and administrators can easily automate their work with less effort. For example they can generate quotations for any enquiry for a car in no time. They can also have the customer's feedback to the show room authority regarding show room services.

1.2 Project Goals

The System should be capable of performing the following:

- Store basic information regarding cars, employees, customers, accessories and services provided by the organization.
- Store salary information of employees (entered by the team leaders in each city) such as, working hours, salary per hour, salary before tax, tax percentage, total amount of tax paid , salary after tax, social security fee, on monthly basis
- Help employees automate the billing process by quickly fetching car/accessory details from the database.
- System should be able to generate invoices and maintain transaction details.
- Should be able to list the inventory of cars currently available in a dealership.
- Should help customers to access car details like model, make, engine no., price etc.

- Help the organization to maintain customer details like customer name, transaction/purchase details, car purchased, purchase medium, purchase date etc.
- Display details of various dealerships of the same franchise/organization like, profits, dealership name, number of cars in dealership, number of employees etc.

System Specifications

2.1 Hardware requirements

Minimum requirements :

- ***processor:*** `Intel Pentium 4/ AMD Athlon series`
- ***RAM:*** `512MB`
- ***Storage:*** `100MB`

Recommended requirements:

- ***processor:*** `Intel 11th generation I-series/AMD Ryzen Threadripper series`
- ***RAM:*** `8GB`
- ***Storage:*** `500GB`

2.2 Software requirements

OS: `Windows,GNU/Linux Distributions, Mac OS, BSD, 64-bit`

Tech stack used:

Front end: `React-JS, Tailwind-CSS`

- **React-JS**:

  React is a free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by Meta and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications

- **Tailwind-css**:
  Tailwind CSS is basically **a utility-first CSS framework for rapidly building**

custom user interfaces. It is a highly customizable, low-level CSS framework that gives you all of the building blocks you need to build bespoke designs without any annoying opinionated styles you have to fight to override.

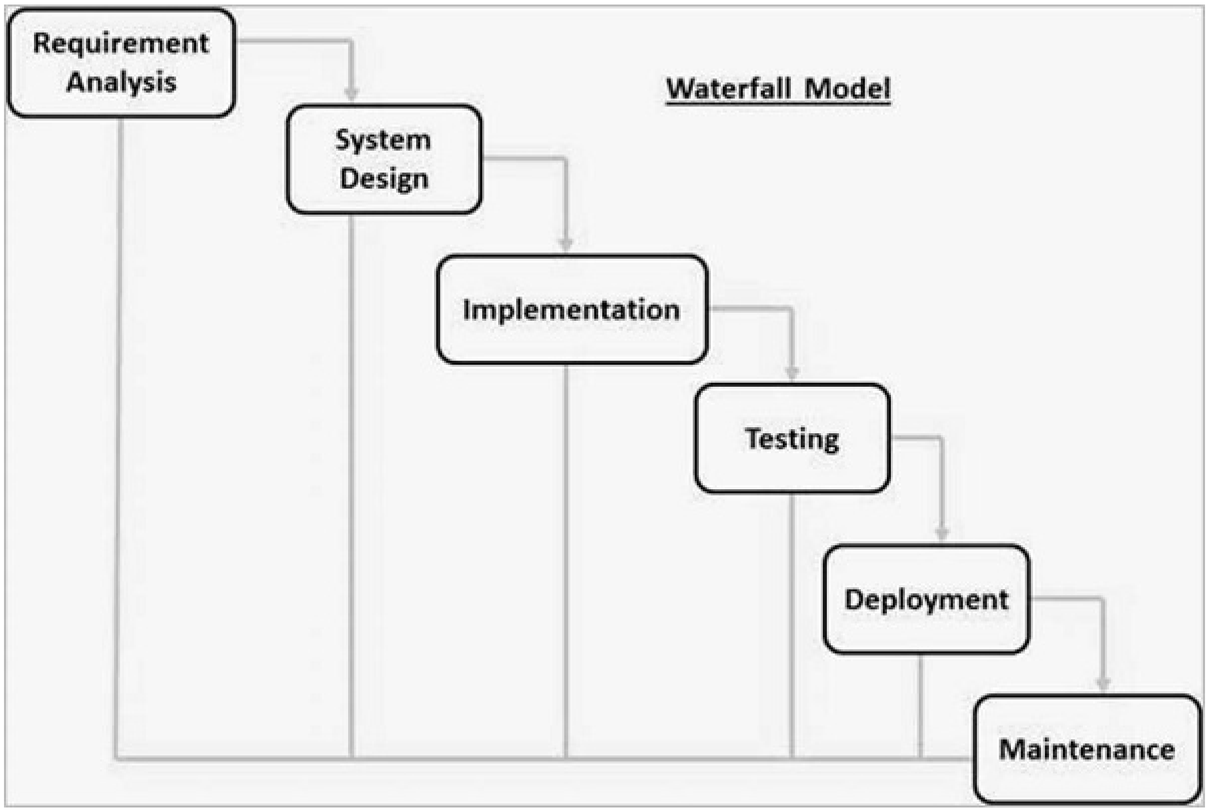
Back end: `Django, SQLite3`

- **Django**:
  Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

- **SQLite3**:
  SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. SQLite is the most used database engine in the world. SQLite is built into all mobile phones and most computers and comes bundled inside countless other applications that people use every day.

Design

3.1 Development model:



- **Requirement Analysis**:
  All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

- **Design stage**:
  The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

- **Implementation**:
  With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

- **Integration and Testing**:
  All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

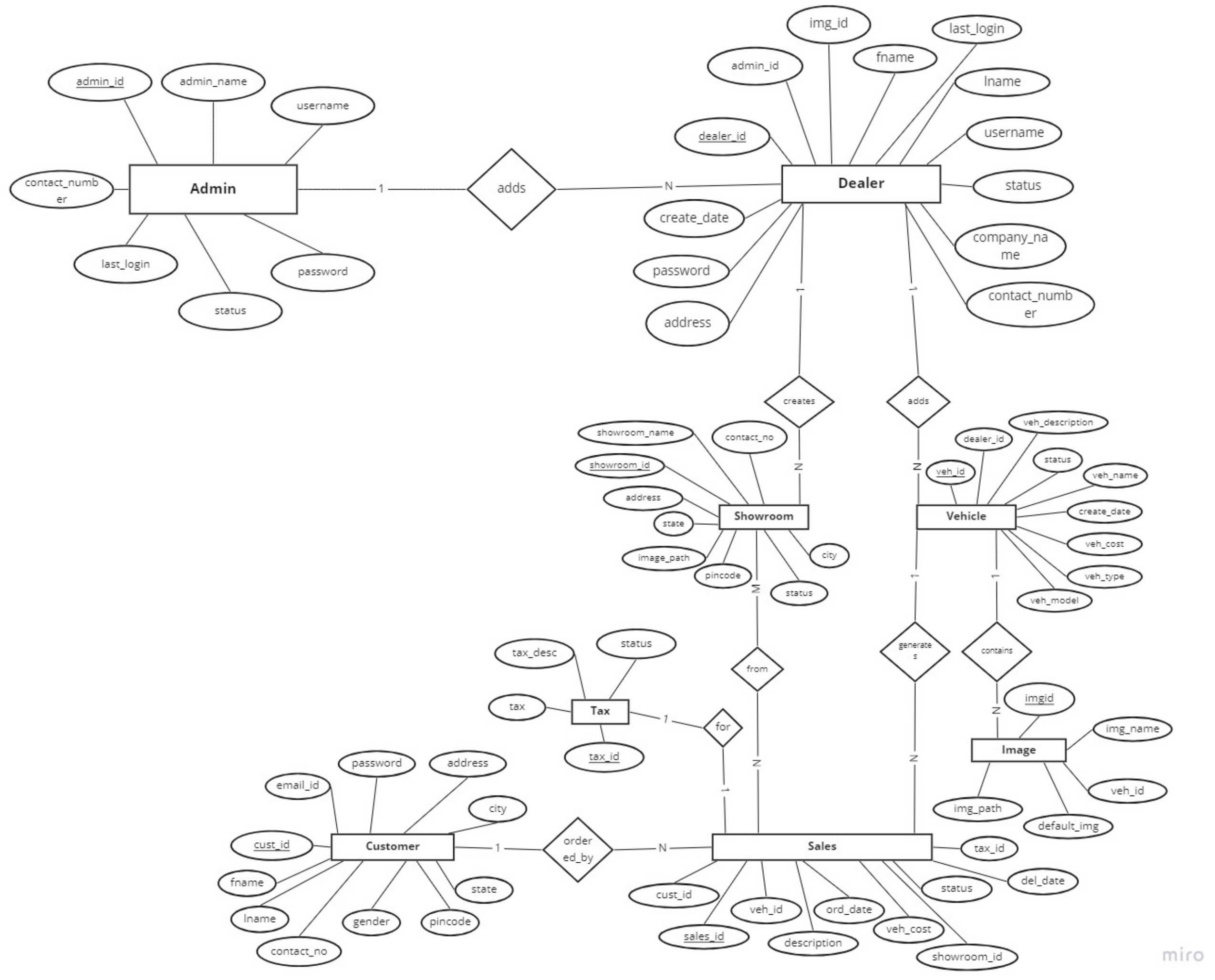
- **Deployment of System**:
  Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

- **Maintenance**:
  There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

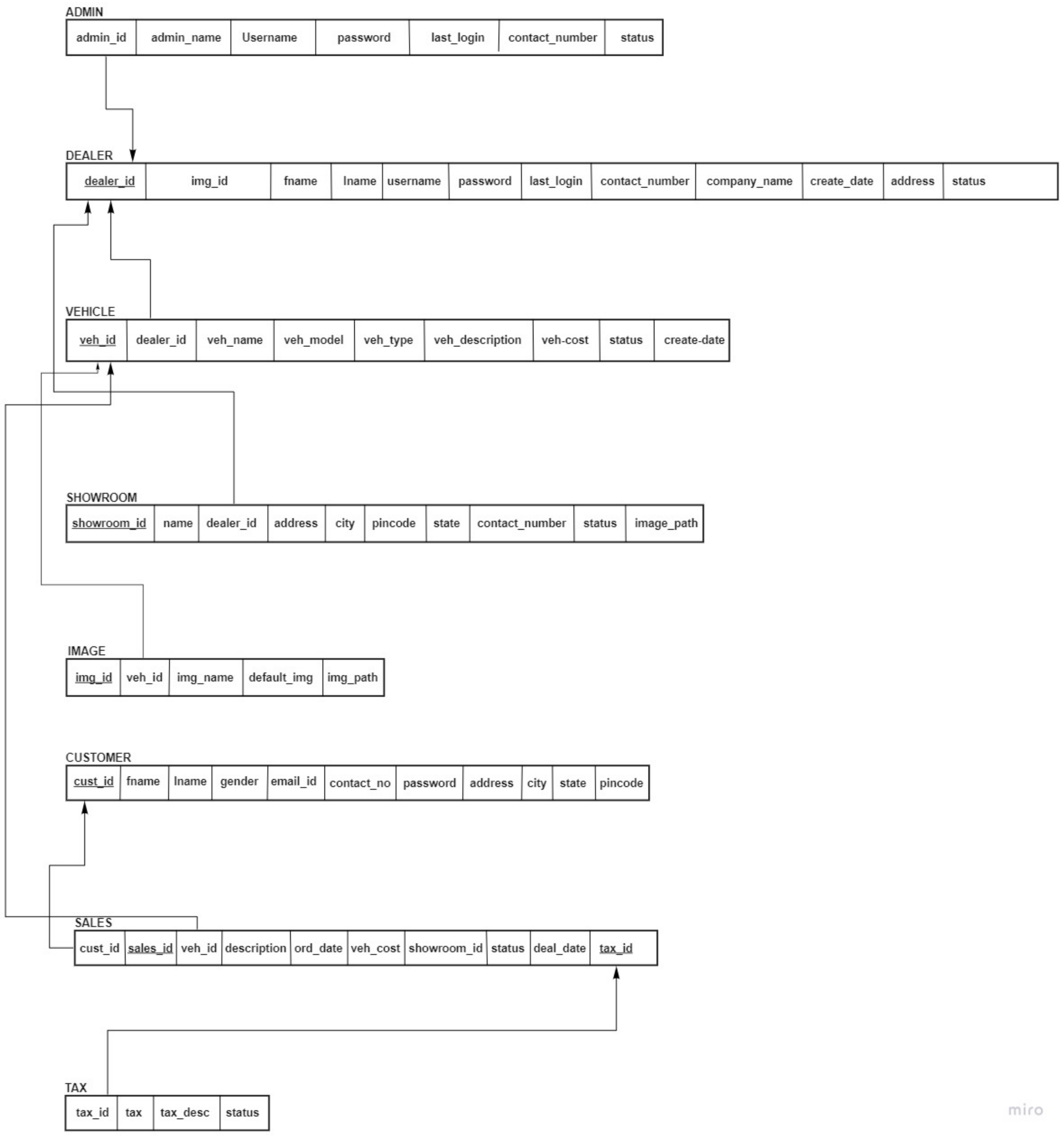
All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

3.2 Database description:

ER Diagram



Schema Diagram


ADMIN
admin_id | admin_name | Username | password | last_login | contact_number | status
DEALER
dealer_id | img_id | fname | lname | username | password | last_login | contact_number | company_name | create_date | address | status
VEHICLE
veh_id | dealer_id | veh_name | veh_model | veh_type | veh_description | veh-cost | status | create-date
SHOWROOM
showroom_id | name | dealer_id | address | city | pincode | state | contact_number | status | image_path
IMAGE
img_id | veh_id | img_name | default_img | img_path
CUSTOMER
cust_id | fname | lname | gender | email_id | contact_no | password | address | city | state | pincode
SALES
cust_id | sales_id | veh_id | description | ord_date | veh_cost | showroom_id | status | deal_date | tax_id
TAX
tax_id | tax | tax_desc | status