

A

COAPPS PROJECT ON

CUSTOMER RELATIONSHIP MANAGEMENT

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

P.Vaasanthi

(20N81A6603)

D.Ganesh

(21N85A6705)

N.Venkata Ravi Kumar

(20N81A6744)

K.Niwas Reddy

(21N85A6206)

From

SPHOORTHY ENGINEERING COLLEGE



ABSTRACT

The point of departure for this study is the understanding of customer relationship management (CRM) as a set of technological solutions key for efficient business management, the benefits of which, highlighted by previous works, are presented and defined here as crucial for entrepreneurial success. Of particular interest for this purpose are the existing studies on sustainability, which provide a viable research model to assess and validate the potential effect of each CRM component (sales, marketing, and services) on the three dimensions of sustainability (economic, environmental, and social). Upon confirmation of our hypotheses, the subsequent validation of such model should bring a better understanding of the way in which CRM-related benefits may increase the positive impact of its components on each dimension of sustainability. CRM can hence be considered a sort of Green IT, oriented toward digital transformation and sustainable business model innovation. Indeed, this research model may be the basis for a more specific methodology to measure the impact and benefits of applying CRM, understood, as we will contend, both in terms of sustainable business models and innovation.

INDEX

Contents		Page No.
Abstract		2
Index		3
List of Figures		4
Chapters		
1.	Introduction	5
1.1.	Background	5
1.2.	Customer Relationship Management System	7
1.3.	Project Aim and Objective	7
1.4.	Motivation	7
1.5.	Existing System	8
1.6.	Proposed System	8
1.7.	Scope of the Study	9
1.7.	System Feasibility Study	10
2.	Literature Survey	
2.1.	Introduction	12
3.	Requirement Specification	
3.1.	Hardware Requirements	15
3.2.	Software Requirements	15
3.3	Version Control	15
3.4	React.js	15
3.5	Node.js	17
3.6	Express.js	18
3.7	MySQL	19
3.8	Languages used	20
3.9	Axios	23
4.	SYSTEM DESIGN	
4.1.	Implementation of System Design	25
4.2.	CRM Flowchart	25
4.3	ER Diagram	26
4.4	Use Case Diagram	27
4.5	Sequence Diagram	28
4.6	Activity Diagram	31
4.7	Data Flow Diagram	32
5.	Implementation	33
6.	Testing	36
7.	Output	38
8.	Conclusion	46

LIST OF FIGURES

Figure No.	Figure Name	Page No.
4.3	ER Diagram	27
4.4	Use Case Diagram	28
4.5.1	Sequence Diagram	29
4.5.2	Sequence Diagram for Admin	30
4.5.3	Sequence Diagram for Customer	31
4.6	Activity Diagram	32
4.7	DataFlow Diagram	32
7.2.1	Database Table Structure	40
7.2.2	Homepage	41
7.2.3	Logins	41
7.2.4	Signup	41
7.2.5	Admin Dashboard	42
7.2.6	Task Modules	42
7.2.7	User Dashboard	42
7.2.8	Creating Tickets	43

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

To embark on building a Customer Relationship Management (CRM) system, a comprehensive background study is essential. This study begins with a deep understanding of CRM systems, their pivotal role in businesses, and the diverse features they encompass, including operational, analytical, and collaborative functionalities. Market analysis becomes paramount, delving into current trends and demands for CRM solutions, assessing competitors, and pinpointing the specific needs of the target audience. Evaluating the technology stack is crucial, exploring frontend frameworks like React.js for dynamic user interfaces, backend options such as Node.js with Express for robust server-side logic, and database choices like MySQL for efficient data management. Concurrently, user experience (UX) research illuminates the importance of intuitive design, prompting surveys or interviews to grasp user pain points and preferences.

1.2 Customer Relationship Management System

CRM (Customer Relationship Management) comprises software tools designed to manage sales, marketing, and service interactions with customers. It has evolved as a crucial paradigm in customer relationship management, essential for modern businesses' success in adapting to market dynamics. The primary goal of CRM is to attract and retain economically valuable customers while prioritizing profitable relationships, emphasizing the need to meet customer needs, improve satisfaction, and enhance loyalty.

- **Definition and Scope:**

CRM comprises software tools to manage sales, marketing, and service interactions with customers.

- **Evolution and Importance:**

CRM has evolved as the latest paradigm in customer relationship management, critical for modern businesses' success in adapting to market dynamics.

- **Primary Goal:**

The primary goal of CRM is to attract and retain economically valuable customers while prioritizing profitable relationships.

- **Strategic Importance:**

CRM is considered crucial for long-term growth and sustainability, emphasizing the need to meet customer needs, improve satisfaction, and enhance loyalty.

- **Coordination of Strategies:**

CRM enhances a company's ability to coordinate marketing and service strategies to reach and retain long-term partnerships with customers.

- **Customer-Centered Focus:**

A customer-centered focus is key to business success, and CRM strategies aim to enhance customer loyalty.

- **Direct Benefits:**

Implementing CRM in SMEs brings direct benefits in terms of financial performance and daily business activity.

- **Customer Service and Support:**

CRM enhances customer service and support by facilitating a thorough understanding of consumer needs and streamlining processes such as order reception and project management.

1.3 PROJECT AIM AND OBJECTIVE

To develop a comprehensive CRM system that empowers businesses to efficiently manage and nurture relationships with their customers, resulting in improved customer satisfaction, increased retention rates, and ultimately, sustainable business growth.

The objectives of this project are to:

User-Friendly Interface: Design and implement a user interface that is intuitive, easy to navigate, and aesthetically pleasing, ensuring a positive user experience for all stakeholders.

- **Comprehensive Contact Management:** Develop functionality to efficiently store, organize, and retrieve customer contact information, including details such as names, addresses, phone numbers, and email addresses.
- **Effective Lead Tracking:** Implement features to track and manage leads throughout the sales pipeline, enabling users to capture, qualify, assign, and prioritize leads effectively.
- **Task Scheduling and Management:** Develop capabilities for scheduling and managing tasks related to customer interactions, follow-ups, meetings, and other activities, ensuring timely follow-through and accountability.
- **Customization and Personalization:** Offer customization options to tailor the CRM system to the unique needs and workflows of different businesses, allowing users to configure settings, fields, and workflows to align with their specific requirements.

1.4 MOTIVATION

- **Competitive Advantage:** In today's competitive landscape, delivering exceptional customer experiences is essential for differentiation. A CRM system equips businesses with the tools and insights needed to outperform competitors, delight customers, and become market leaders.

- **Customer Retention and Loyalty:** By proactively engaging with customers, addressing their needs, and anticipating their preferences, businesses can foster loyalty and retention. A CRM system facilitates ongoing communication and relationship-building, reducing churn and increasing customer lifetime value.
- **Efficient Task Management:** With task scheduling and management features, CRM systems help teams stay organized, prioritize activities, and ensure timely follow-up with leads and customers. This boosts productivity and reduces the risk of missed opportunities.

1.5 EXISTING SYSTEM

- The paper aims to establish Limited insights without complete data profiles.
- CRM systems often store basic contact details such as names, addresses, phone numbers, and email addresses. However, relying solely on this data may lead to incomplete customer profiles and limited insights into customer behavior and preferences.

DISADVANTAGES:

- Outdated or inaccurate data without proper customization.
- Customer Feedback and Surveys
- Time-consuming analysis, difficulties in identifying trends.
- Risk of data breaches, unauthorized access, and compliance violations.

1.6 PROPOSED SYSTEM

Our proposed CRM (Customer Relationship Management) system is designed to streamline and optimize the management of customer relationships for businesses of all sizes

- User-Friendly Interface
- Our CRM system will facilitate better customer service and support by

enabling users to track customer inquiries, issues, and requests. Users can assign tickets to team members, track ticket status, and ensure timely resolution of customer issues.

- Our CRM system will enable businesses to store, organize, and manage customer contact information effectively. Users can easily add, edit, and delete contacts, as well as track interactions and communications with each contact.

ADVANTAGES:

- Integration with External Systems
- Security and Data Privacy
- Customer Service and Support
- Sales Pipeline Management

1.7 SCOPE OF THE STUDY

The scope of study for the CRM system development project encompasses several key components aimed at effectively managing customer relationships and optimizing business processes. Functionally, the system will include features such as contact management for storing and updating customer information, sales pipeline management for tracking leads and opportunities, marketing campaign management for creating and tracking promotional activities, and customer service and support tools for managing inquiries and issues. User roles and permissions will be defined to ensure appropriate access levels and security measures. Data management will focus on collecting, storing, and processing customer data securely while integrating with external systems such as email clients and marketing automation platforms. The chosen technology stack will be carefully selected to meet scalability, performance, and security requirements. The geographical scope will consider where the system will be deployed and used, taking into account language, currency, and regulatory compliance. The project timeframe, resources, and limitations will be clearly defined to guide planning,

execution, and evaluation throughout the project lifecycle. Overall, this scope of study provides a comprehensive framework for developing a CRM system that meets the needs of businesses in managing customer relationships and driving growth.

The challenges faced during the project are as follows:

1.7.1 User Experience (UX): Designing a user-friendly interface and intuitive workflows that cater to the needs of diverse user roles and preferences can be challenging, especially when balancing usability with functionality.

1.7.2 Integration with External Systems

1.7.3 Scalability

1.8 SYSTEM STUDY FEASIBILITY STUDY

A feasibility study for a CRM (Customer Relationship Management) system evaluates the viability and potential success of implementing such a system within an organization.

Three key considerations involved in the feasibility analysis are

- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**
- **ECONOMICAL FEASIBILITY**

TECHNICAL FEASIBILITY

- Assessment of the organization's existing technical infrastructure and capabilities to support the implementation of a CRM system.
- Evaluation of the compatibility of the CRM system with existing hardware, software, databases, and network infrastructure.
- Consideration of technical requirements such as scalability, performance, security, and integration with other systems.

Financial Feasibility

- Cost-benefit analysis to evaluate the financial implications of implementing a CRM system, including initial investment, ongoing maintenance costs, and expected return on investment (ROI).
- Comparison of the costs of implementing a CRM system against potential savings and revenue gains resulting from improved customer relationships, increased sales, and enhanced productivity.
- Consideration of alternative financing options, such as budget allocation, external funding, or financing arrangements with CRM vendors.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

Risk Analysis

- Identification and evaluation of potential risks and uncertainties that could affect the success of the CRM implementation project, such as technical challenges, organizational resistance, budget overruns, and market volatility.
- Development of risk mitigation strategies and contingency plans to address potential risks and minimize their impact on the project.

CHAPTER-2

LITERATURE SURVEY

2.1 INTRODUCTION

The purpose of the literature review is to give a summary of facts and findings on this project. This can be done by studying or finding the references or related findings. This review will give a better understanding to the need for this project and also help in designing the methodology for this project. In this project methodology section will describe in detail about the selected methodology or approach that will be used in this project. By selecting the suitable methodology, the productivity and quality of the project will be increased and improved.

Customer Relationship Management (CRM) has garnered significant attention in both academic and business circles due to its pivotal role in fostering customer-centric strategies and enhancing organizational performance. Over the years, scholars and practitioners have extensively studied various aspects of CRM, including its definition, evolution, benefits, challenges, and implementation strategies.

BasePaper-1

Title: Understanding customer relationship management (CRM) by people, process and technology

Author: Chen and Popovich (Department of Operations Management and Business Statistics, College of Business Administration, Cleveland State University, Cleveland, Ohio, USA)

Objective:

- According to Chen and Popovich (2003), CRM encompasses a set of software tools and methodologies aimed at managing customer interactions across sales, marketing, and service functions. This definition underscores CRM's multifaceted nature, emphasizing its role in

facilitating seamless communication and collaboration between different departments to better serve customers.

Multifaceted Nature:

- CRM is described as multifaceted, indicating that it involves multiple aspects and functionalities.
- It is not limited to one specific function but encompasses a range of activities related to customer management.

Facilitating Communication and Collaboration:

- A significant emphasis is placed on CRM's role in facilitating seamless communication and collaboration.
- CRM systems are designed to enable interaction and cooperation between different departments within an organization.
- This collaboration is aimed at better serving customers by providing them with a cohesive and consistent experience across various touchpoints.

Integration Across Functions:

- CRM systems integrate sales, marketing, and service functions to provide a holistic view of customer interactions.

Enhancing Customer Service:

- One of the primary objectives of CRM is to enhance customer service by improving communication and coordination between departments.

BasePaper:2

Title : Customer relationship management: digital transformation and sustainable business

model innovation

Author : Scullin et al. (2002). Department of Computer Systems and Computation, Universitat Politècnica de Valencia, Valencia, Spain

Objective:

This paper proposes a research model to analyse how customer relationship management (CRM) brings small and medium enterprises (SMEs) a dual benefit, in terms of both customer knowledge management (CKM) and innovation. This confluence of interests and benefits is a key point to consider CRM a critical tool for business model innovation, driving SME efforts toward economic, social and environmental sustainability. Traditionally, SMEs have been the cornerstone of the European economy, comprising over 99% of all European companies, and two thirds of the private sector jobs (European Commission, 2013). Thus the impact of CRM on SMEs is of special interest given the social and economic relevance of this sector.

CHAPTER-3

REQUIREMENT SPECIFICATION

3.1 Hardware Requirements:

- Processor – Multi-core processor (e.g., Intel Core i5 or higher)
- RAM – Minimum 4GB
- Storage – Solid State Drive (SSD)

3.2 Software Requirements:

- Operating system - Windows8 or Above

3.2.1 Development Environment:

- React.js
- MySQL
- Node.js
- HTML, JavaScript, CSS (fundamental languages used for building frontend)
- Code Editor – Visual Studio Code

3.3 Version Control:

- Git

3.4 React.js

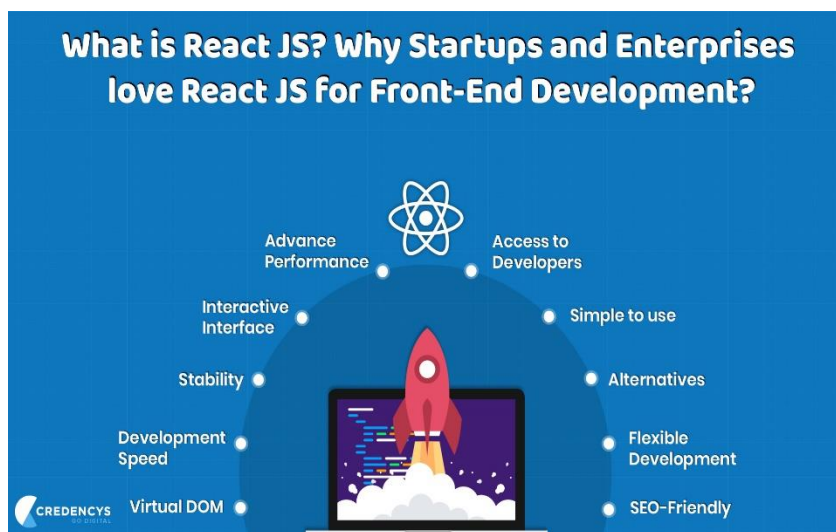
ReactJS is a JavaScript library primarily used for building user interfaces (UIs) for web applications. It was developed by Facebook and released as an open-source project. ReactJS allows developers to create reusable UI components and efficiently manage the state of their applications.

- **Component-Based Architecture:** ReactJS is based on a component-based architecture. A component is a reusable piece of code that

encapsulates the UI and its behavior. Components can be nested within each other to create complex UI structures. This approach promotes code reusability, maintainability, and scalability.

- **Virtual DOM:** One of the key features of ReactJS is its Virtual DOM (Document Object Model). Instead of directly manipulating the browser's DOM, React creates an in-memory representation of the DOM known as the Virtual DOM. When the state of a component changes, React compares the Virtual DOM with the previous state and efficiently updates only the parts of the actual DOM that have changed. This helps in improving performance and reducing unnecessary re-rendering of components.
- **JSX (JavaScript XML):** ReactJS introduces JSX, which is a syntax extension for JavaScript. JSX allows developers to write HTML-like code within JavaScript, making it easier to describe UI components. JSX gets transpiled into regular JavaScript by tools like Babel before it's served to the browser.
- **Unidirectional Data Flow:** React follows a unidirectional data flow, also known as one-way data binding. Data flows downwards from parent components to child components. This ensures predictable data flow and makes it easier to debug and understand the application's state.
- **State Management:** React components can have a local state which is managed using the `useState` hook (for functional components) or the `setState` method (for class components). State represents the data that can change over time within a component. By updating the state, React triggers re-rendering of the component, ensuring that the UI stays in sync with the data.
- **React Router:** React Router is a popular library for handling routing in React applications. It allows developers to define routes and map them to specific components. This enables the creation of single-page applications (SPAs) with multiple views or pages without full-page reloads.

- **React Hooks:** Introduced in React 16.8, hooks are functions that allow developers to use state and other React features in functional components. Hooks like `useState`, `useEffect`, `useContext`, etc., provide a way to use React features without writing class components. Hooks enable better code organization and reusability.
- **Community and Ecosystem:** ReactJS has a vast and active community, which has contributed to the development of numerous third-party libraries and tools. These include libraries for state management (Redux, MobX), UI component libraries (Material-UI, Ant Design), and development tools (React DevTools, Create React App).



3.5 Node.js

Node.js is a powerful open-source, cross-platform JavaScript runtime environment that executes JavaScript code outside of a web browser. It is built on Chrome's V8 JavaScript engine and was initially developed by Ryan Dahl in 2009.

- **JavaScript Runtime Environment:** Node.js provides an environment for running JavaScript on the server-side. Traditionally, JavaScript was primarily used for client-side scripting within web browsers.

- **Asynchronous and Event-Driven:** One of the key features of Node.js is its asynchronous, non-blocking I/O model. This means that Node.js can handle multiple connections simultaneously without getting blocked by I/O operations.
- **Single-Threaded Event Loop:** Node.js operates on a single-threaded event loop mechanism. This event loop allows Node.js to handle multiple concurrent requests efficiently. When a request is received, Node.js delegates the I/O operations to the system kernel and continues processing other requests.
- **NPM (Node Package Manager):** NPM is the default package manager for Node.js, providing access to a vast ecosystem of reusable JavaScript modules and libraries. It allows developers to easily install, manage, and share dependencies for their Node.js projects.
- **Cross-Platform:** Node.js is designed to be cross-platform, meaning it can run on various operating systems such as Windows, macOS, and Linux. This allows developers to write code once and deploy it across different environments without modification, enhancing code portability and reducing development time.
- **Server-Side Web Development:** Node.js is widely used for server-side web development, powering backend APIs, web servers, and real-time web applications.
- **Desktop Applications:** With frameworks like Electron, Node.js can be used to build desktop applications using web technologies (HTML, CSS, and JavaScript)

3.6Express.js

Express.js is a lightweight and flexible Node.js web application framework used to build web applications and APIs. It simplifies the process of handling HTTP requests and responses, managing middleware, and defining routes. With features like middleware support, routing, template engines, static file serving, error handling, and RESTful API creation, Express.js

provides developers with a powerful toolset for building server-side applications in Node.js.

3.7 MySQL

MySQL is an open-source relational database management system (RDBMS) that is widely used for managing structured data. It is developed, distributed, and supported by Oracle Corporation.

- **Relational Database Management System (RDBMS):** MySQL is an RDBMS, which means it organizes data into tables with rows and columns. It follows the relational model, where data is stored in tables, and relationships between tables are established using keys. This allows for efficient storage, retrieval, and manipulation of structured data.
- **Open Source:** MySQL is open-source software, licensed under the GNU General Public License (GPL). This means that it can be freely downloaded, used, modified, and distributed by anyone. The open-source nature of MySQL has contributed to its widespread adoption and the growth of its community.
- **Client-Server Architecture:** MySQL follows a client-server architecture, where multiple clients can connect to a MySQL server simultaneously to access and manipulate data. The server manages multiple databases and handles incoming requests from clients, executing SQL queries and returning results.
- **SQL (Structured Query Language):** MySQL uses SQL as its query language for interacting with databases. SQL is a standardized language for defining, querying, and manipulating relational databases. MySQL supports a wide range of SQL commands for tasks such as creating databases and tables, inserting, updating, and deleting data, and performing complex queries.
- **Data Types:** MySQL supports various data types to represent different types of data, including integers, floating-point numbers, strings, dates, times, and binary data. Each column in a MySQL table is assigned a

specific data type, which defines the kind of data that can be stored in that column.

- **Indexes and Keys:** MySQL allows the creation of indexes and keys to optimize data retrieval and enforce data integrity. Indexes are data structures that improve the speed of data retrieval operations by enabling faster lookup of values in columns. Keys, such as primary keys and foreign keys, are used to establish relationships between tables and enforce referential integrity.
- **Scalability and Replication:** MySQL is designed to be scalable, allowing it to handle large volumes of data and high concurrent loads. It supports various scalability features, including replication, sharding, and clustering. Replication enables data to be replicated across multiple MySQL servers, providing fault tolerance, load balancing, and scalability.

3.8 fundamental languages used for building frontend

3.8.1 HTML (Hypertext Markup Language):

- HTML is the standard markup language used for creating the structure and content of web pages.
- It consists of a set of elements or tags that define the different parts of a webpage, such as headings, paragraphs, images, links, forms, and more.
- HTML elements are represented by tags enclosed in angle brackets (< and >), and they typically come in pairs, with an opening tag and a closing tag. For example, <p> denotes the start of a paragraph, while </p> denotes the end of the paragraph.
- HTML elements can have attributes that provide additional information about the element, such as its appearance, behavior, or functionality. Attributes are specified within the opening tag of the element.
- HTML documents are structured as a hierarchy of nested elements, with the <html> element at the top, followed by <head> and <body> elements.
- HTML is not a programming language but rather a markup language used

to define the structure and semantics of web content.

3.8.2 JavaScript:

- JavaScript is a high-level, interpreted programming language that is primarily used for adding interactivity and dynamic behavior to web pages.
- It is a versatile language that can be used both on the client-side (in web browsers) and the server-side (with platforms like Node.js).
- JavaScript enables developers to manipulate HTML content, respond to user interactions, and dynamically update the appearance and behavior of web pages.
- JavaScript code is typically embedded within HTML documents using `<script>` tags, either inline or as external script files referenced by the `src` attribute.
- JavaScript syntax is similar to other programming languages such as Java and C, making it relatively easy to learn for developers familiar with those languages.
- JavaScript supports a wide range of features, including variables, data types, operators, control structures (such as loops and conditionals), functions, objects, arrays, and more.
- With the advent of modern JavaScript frameworks and libraries (such as React, Angular, and Vue.js), JavaScript has become a cornerstone of web development, enabling the creation of complex and interactive web applications.

3.8.3 CSS (Cascading Style Sheets):

- CSS is a stylesheet language used for describing the presentation and visual styling of HTML elements on web pages.
- It allows developers to control the layout, appearance, and design of web pages, including aspects such as colors, fonts, spacing, borders, and

positioning.

- CSS operates on a rule-based system, where styles are applied to HTML elements based on selectors that target specific elements, classes, or IDs.
- CSS rules consist of a selector (which identifies the elements to style) and one or more declarations (which define the styling properties and their values).
- CSS styles can be applied to HTML documents in various ways, including inline styles directly within HTML elements, internal styles embedded within `<style>` tags in the `<head>` section of HTML documents, or external stylesheets linked to HTML documents using the `<link>` tag.
- CSS supports a wide range of selectors, properties, and values, allowing developers to create complex layouts and designs while maintaining separation of concerns between content (HTML), behavior (JavaScript), and presentation (CSS).
- CSS preprocessors like Sass and Less provide additional features and enhancements to CSS, such as variables, mixins, nesting, and functions, to streamline the development process and improve code maintainability.



3.9 Axios

Axios is a popular JavaScript library used for making HTTP requests from web browsers or Node.js. It simplifies the process of sending asynchronous HTTP requests to web servers and handling responses

- **Simplicity and Ease of Use:**

Axios provides a simple and intuitive API for making HTTP requests, making it easy to use for developers. It offers a clean syntax for specifying request parameters, headers, data, and URL endpoints.

- **Browser and Node.js Support:**

Axios can be used both in web browsers and in Node.js environments, making it versatile and widely adopted across different platforms. In web browsers, Axios utilizes XMLHttpRequests (XHR) or the Fetch API to send requests.

In Node.js, Axios uses the http or https modules for making HTTP requests.

- **Promises-based:**

Axios is built on top of JavaScript promises, allowing developers to use async/await syntax or traditional promise chains for handling asynchronous operations. This makes it easier to manage asynchronous code and handle response data in a structured manner.

- **Interceptors:**

Axios provides a feature called interceptors, which allows developers to intercept and modify requests or responses before they are sent or processed.

- **Request and Response Handling:**

Axios allows developers to handle different types of HTTP requests, including GET, POST, PUT, DELETE, and more. It provides methods for specifying request parameters, headers, data, and URL

endpoints. Developers can also handle various types of response data, including JSON, binary data, or form data.

- **Error Handling:**

Axios simplifies error handling by automatically rejecting promises when requests fail (e.g., due to network errors, server errors, or timeouts). Developers can use try/catch blocks or `.catch()` methods to handle errors and implement error recovery strategies.

- **Cross-Origin Resource Sharing (CORS):**

Axios handles Cross-Origin Resource Sharing (CORS) by default, allowing developers to make requests to different origins or domains without running into security restrictions. CORS headers such as Access-Control-Allow-Origin are automatically handled by Axios, simplifying the process of making cross-origin requests.

SYSTEM DESIGN

4.1 Importance of System Design

The purpose of system design phase is to plan a solution of the problem specified by the required document. It is the process of defining software methods, functions, objects and overall structure and interaction of your code so that the resulting functionality will satisfy your users requirements. It allows you to do the best abstraction, to understand the requirements better and meet them better. This prevents redundancy and increases reusability. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed, design takes us towards how to satisfy the needs. The design of a system is perhaps the most critical factor affecting the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design.

4.2 CRM flowchart

Flowcharts are visual representations of processes or workflows, commonly used in various fields such as software development, business analysis, project management, and engineering. They use standardized symbols and connectors to depict the sequence of steps, decision points, and interactions within a process.

Creating a flowchart for a Customer Relationship Management (CRM) system involves visually representing the various processes and workflows involved in managing customer interactions, sales activities, marketing campaigns, and customer support.

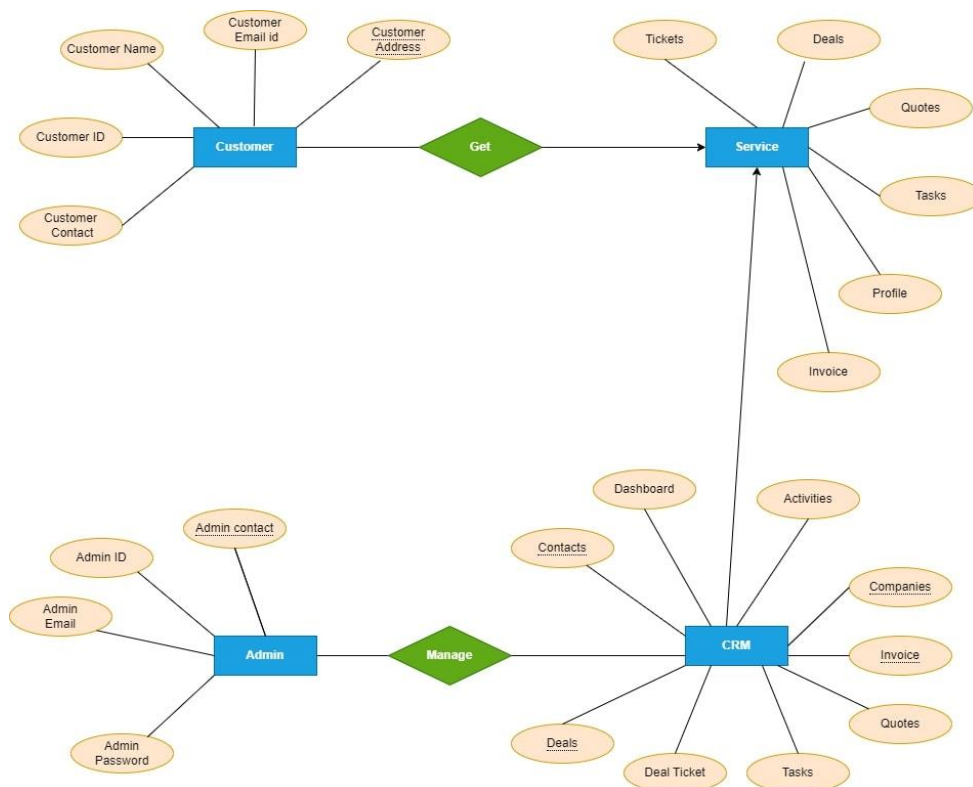


Fig 4.3 : ER Diagram

4.4 Use Case Diagram

Use Case Diagram is a vital tool in system design, it provides a visual representation of how users interact with a system. It serves as a blueprint for understanding the functional requirements of a system from a user's perspective, aiding in the communication between stakeholders and guiding the development process.

A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure. These behaviours, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications with its environment. A use case can include possible variations of its basic behavior, including

exceptional behavior.

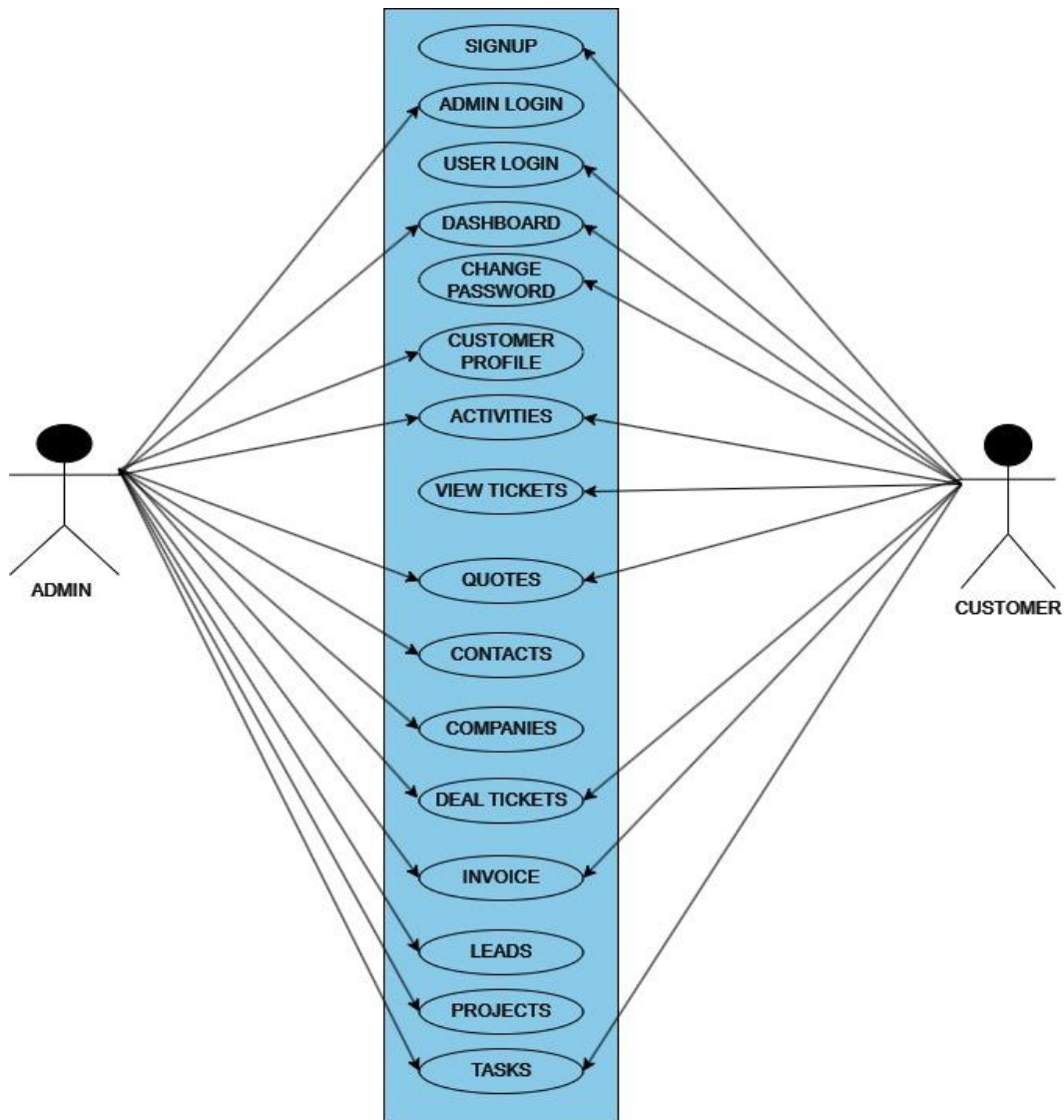


Fig 4.4 : Use Case Diagram

4.5 Sequence Diagram

A sequence diagram is a type of Unified Modeling Language (UML) diagram that illustrates the interactions and communication sequences between objects or components within a system over a specific period of time. It is a modeling language in the field of software engineering that aims to set standard ways to visualize the design of a system. Sequence diagrams are often used to model the dynamic behavior of a system, showing how various objects or components collaborate and exchange

messages to achieve a specific functionality or scenario.

4.5.1 Sequence Diagram for Admin and Customer Login

In a sequence diagram for admin and customer login, the process begins with either the admin or customer initiating the login request by providing their credentials. The system then verifies these credentials against stored data in the database. If the credentials are valid, the system proceeds to authenticate the user and determine their role (admin or customer). Depending on the role identified, the system grants appropriate access to the user. Once authenticated and authorized, the login process is completed, allowing the user to access the system's functionalities.

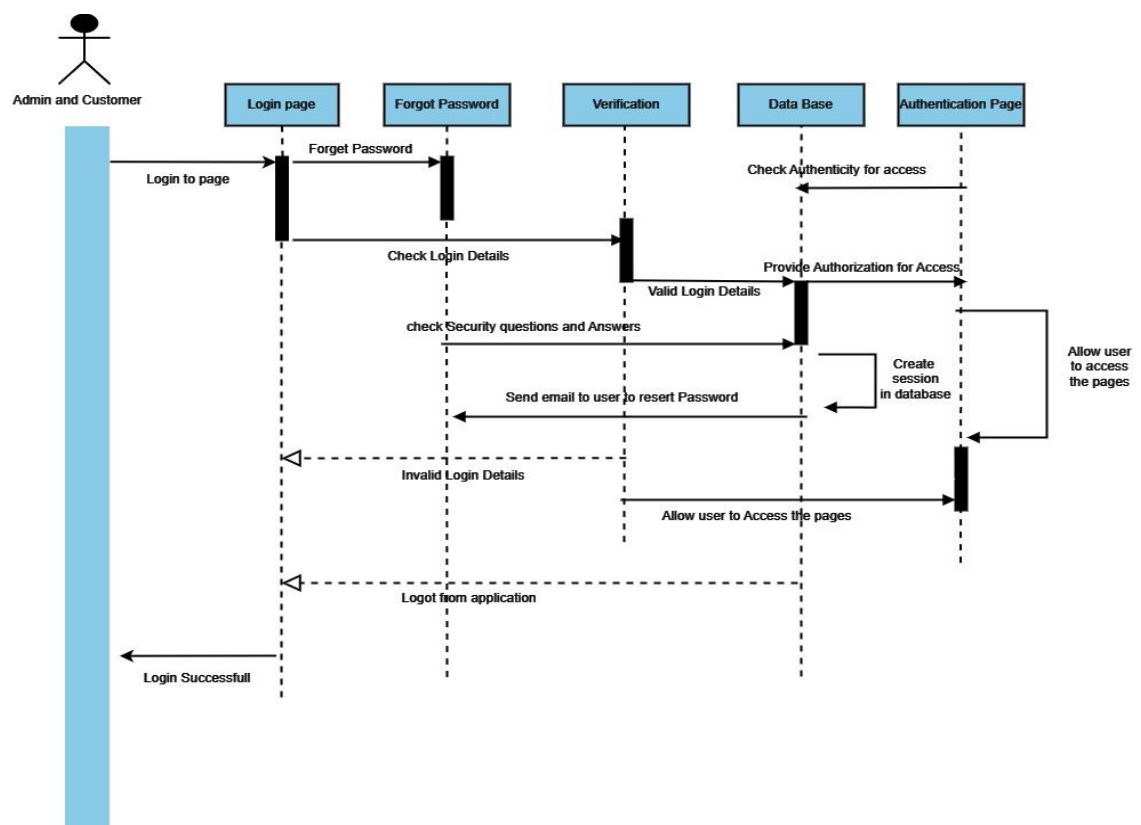


Fig.4.5.1 Sequence Diagram for Admin and Customer Login

4.5.2 Sequence diagram for Admin

In a sequence diagram for an admin, we depict the interactions between the admin actor and the system components involved in typical admin tasks

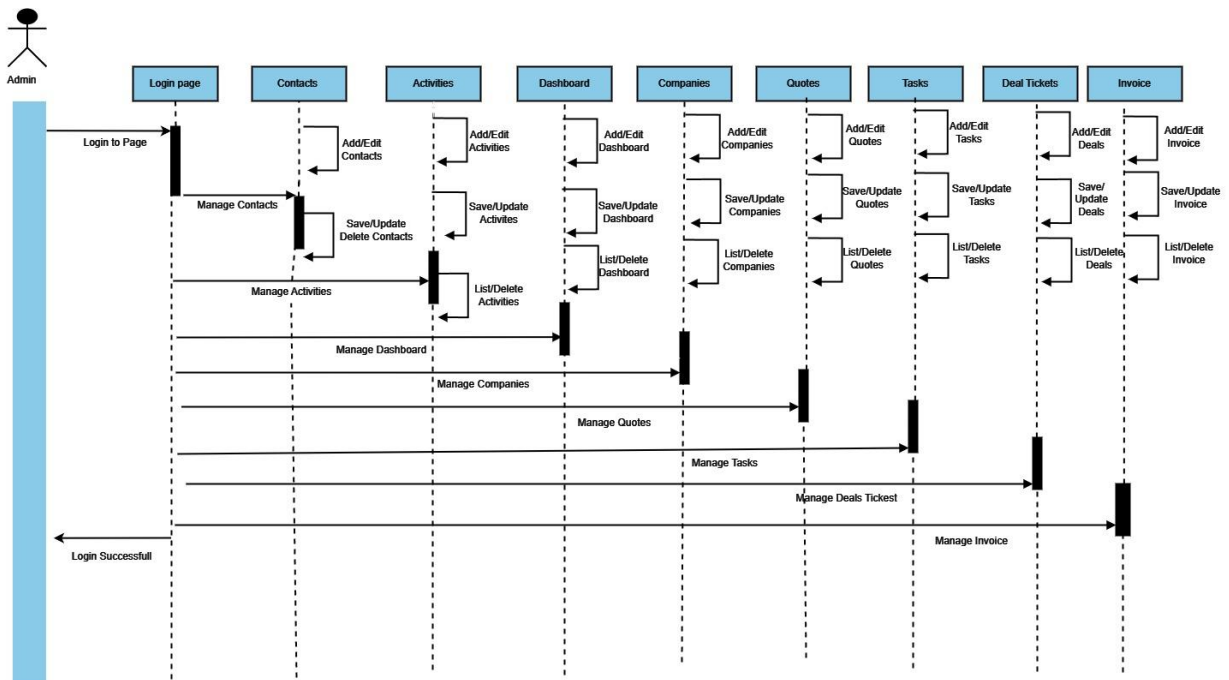


Fig.4.5.2 Sequence Diagram for Admin

4.5.3 Sequence diagram for Customer

In a sequence diagram for a customer, we illustrate the interactions between the customer actor and the system components involved in typical customer actions.

This sequence diagram outlines the typical flow of interactions between a customer actor and the system components involved in customer actions, such as browsing products, adding items to the cart, and completing a purchase.

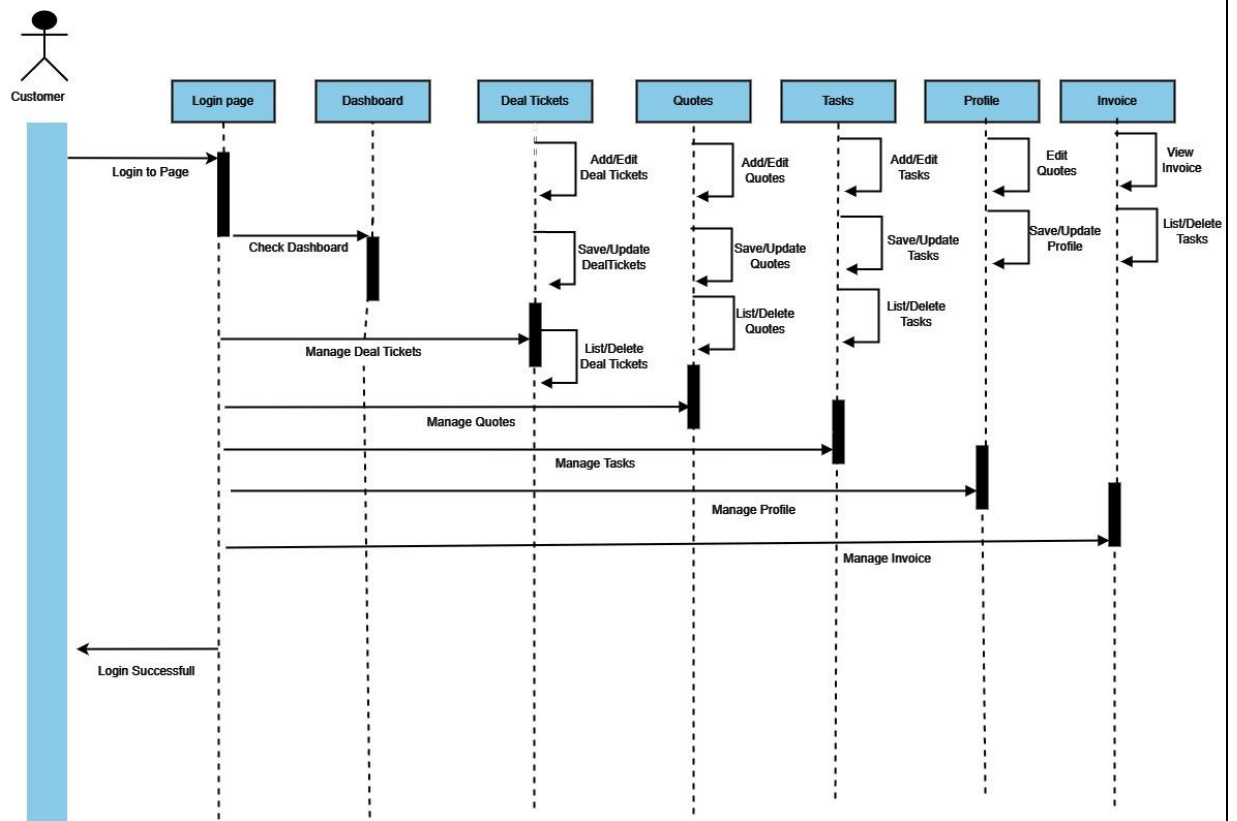


Fig 4.5.3 : Sequence Diagram for Customer

4.6 Activity Diagram

An activity diagram is a type of Unified Modeling Language (UML) diagram used to visualize and model the workflow, activities, and actions within a system or a specific business process. Activity diagrams provide a high-level view of the steps, decisions, and interactions that occur in a process, making them particularly useful for modeling business processes, software workflows, and other complex systems.

Activity Diagrams are used to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. It is a type of behavioral diagram and we can depict both sequential processing and concurrent processing of activities using an activity diagram ie an activity diagram focuses on the condition of flow and the sequence in which it happens.

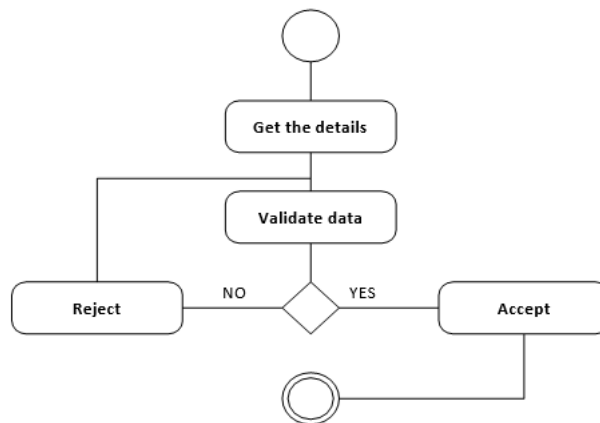


Fig 4.6 : Activity Diagram for User and Admin Login

4.7 Dataflow Diagram for CRM

A Data Flow Diagram (DFD) for a CRM (Customer Relationship Management) system visually represents the flow of data within the system and between external entities.

By creating a DFD for a CRM system, stakeholders can visualize the flow of data, understand the interactions between processes and external entities, and identify potential bottlenecks or areas for improvement in the system's data flow. DFDs serve as valuable tools for analyzing, designing, and communicating the structure and behavior of complex systems like CRM systems.

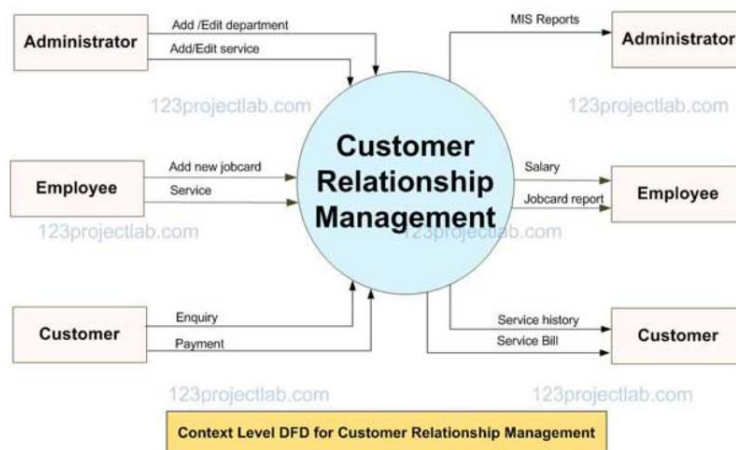


Fig 4.6 : Data Flow Diagram

CHAPTER 5

IMPLEMENTATION

5.1 Process involved in implementation

Implementing a CRM (Customer Relationship Management) system involves several steps to ensure successful adoption and integration into an organization's operations.

➤ **Assessment and Planning:**

- **Assess current processes and systems:** Evaluate existing processes for managing customer interactions, data, and relationships. Identify pain points, inefficiencies, and areas for improvement.
- **Define objectives and requirements:** Determine the goals and objectives of implementing a CRM system. Identify key requirements such as functionality, scalability, integration with existing systems, and user accessibility.
- **Select a CRM solution:** Research and evaluate CRM software options that best align with the organization's needs and budget. Consider factors such as features, customization options, vendor reputation, and customer support.

➤ **Customization and Configuration:**

- **Customize the CRM system:** Tailor the CRM software to meet the specific needs and workflows of the organization. Customize data fields, user interfaces, reports, and automation rules to align with business processes.
- **Configure user roles and permissions:** Define user roles and access levels based on job responsibilities and security requirements. Set up permissions to control access to sensitive data and functionalities within the CRM system.

➤ **Data Migration and Integration:**

- **Prepare data for migration:** Cleanse and prepare existing customer data for migration to the CRM system. Standardize data formats, resolve duplicates, and ensure data accuracy and completeness.
- **Migrate data to the CRM system:** Use data migration tools or services to transfer customer data from legacy systems, spreadsheets, or databases to the CRM platform. Verify data integrity and conduct testing to ensure successful migration.
- **Integrate with existing systems:** Implement integration solutions to connect the CRM system with other systems and applications used within the organization, such as ERP (Enterprise Resource Planning) systems, marketing automation platforms, email marketing tools, and customer support systems. Ensure seamless data flow and synchronization between systems.
- **Technology Stack:** Use technologies such as React.js for building dynamic user interfaces, Node.js for server-side development, and JavaScript for client-side scripting. Tools like MongoDB or MySQL can be used for database management.

➤ **Customization and Configuration according to technologies:**

- **Technology Stack:** Utilize React.js for frontend customization and Node.js with frameworks like Express.js for backend customization. Employ JavaScript for implementing custom functionalities and business logic within the CRM system.
- **Technology Stack:** Employ testing frameworks such as Jest and Mocha for automated testing of React.js components and Node.js APIs. Utilize tools like Postman for API testing and Selenium for end-to-end testing of the CRM system.

➤ **Training and Adoption:**

- **Provide training and support:** Conduct training sessions to educate users on how to use the CRM system effectively. Train users on data entry, navigation, reporting, and best practices for managing customer interactions.

➤ **Testing and Quality Assurance:**

- **Conduct testing:** Perform thorough testing of the CRM system to identify and resolve any issues or bugs. Test functionality, data integrity, performance, security, and usability across different devices and platforms.

➤ **Deployment and Go-Live:**

- **Plan deployment:** Develop a deployment strategy and schedule for rolling out the CRM system to users. Determine the sequence of deployment, user groups, and any downtime or disruptions.
- **Execute deployment:** Deploy the CRM system according to the planned strategy. Monitor the deployment process, address any issues or challenges that arise, and ensure a smooth transition to the new system.
- **Go-live and post-launch support:** Officially launch the CRM system to users and stakeholders. Provide ongoing support and monitoring to address any post-launch issues, optimize system performance, and ensure user satisfaction.

CHAPTER 6

TESTING

6.1 Introduction

Testing for a CRM (Customer Relationship Management) system involves verifying its functionality, performance, security, and usability to ensure that it meets the requirements and expectations of users and stakeholders. Here's an explanation of testing approaches commonly used for CRM systems:

✓ **Functional Testing:**

- **Unit Testing:** Test individual components, modules, or functions of the CRM system in isolation to ensure they behave as expected. Use testing frameworks like Jest or Mocha for JavaScript-based components.
- **Integration Testing:** Verify interactions between different modules or subsystems of the CRM system to ensure they work together seamlessly. Test data flow, communication protocols, and API integrations.
- **End-to-End Testing:** Validate the entire CRM system's functionality from the user interface to the backend. Automate user interactions and workflows using tools like Selenium or Cypress to simulate real-world scenarios.

✓ **Performance Testing:**

- **Load Testing:** Assess the CRM system's ability to handle concurrent user interactions and process requests under heavy load. Use load testing tools like Apache JMeter or LoadRunner to simulate high traffic and measure response times, throughput, and resource utilization.
- **Stress Testing:** Push the CRM system beyond its capacity limits to identify performance bottlenecks, stability issues, or resource

exhaustion. Gradually increase the load or stress levels until the system reaches its breaking point.

- **Scalability Testing:** Evaluate how well the CRM system scales with increased user loads, data volumes, or concurrent transactions. Test horizontal and vertical scaling strategies to ensure the system can handle growth without degradation in performance.

✓ **Security Testing:**

- **Vulnerability Scanning:** Use automated security scanning tools like OWASP ZAP or Nessus to identify potential security vulnerabilities in the CRM system, such as SQL injection, cross-site scripting (XSS), or insecure authentication mechanisms.
- **Penetration Testing:** Conduct manual or automated penetration tests to simulate real-world attacks and identify exploitable security weaknesses. Test authentication mechanisms, authorization controls, data encryption, and session management.
- **Data Protection:** Ensure that sensitive customer data stored in the CRM system is protected from unauthorized access, tampering, or disclosure. Implement encryption, access controls, and data masking techniques to safeguard customer information.

✓ **Usability Testing:**

- **User Experience (UX) Testing:** Evaluate the CRM system's user interface design, navigation, and overall user experience. Conduct usability testing sessions with representative users to gather feedback, identify usability issues, and improve user satisfaction.
- **Accessibility Testing:** Verify that the CRM system complies with accessibility standards and guidelines, ensuring it is usable by people with disabilities. Test screen reader compatibility, keyboard navigation, and color contrast ratios.

CHAPTER 7

OUTPUT

7.1 Output Design

The system after careful analysis has been identified to be presented with the following modules. The modules involved are:

- Admin
- Customer
- Activities Module
- Leads Module
- Tasks Module
- Contacts Module
- Invoice Module
- Dashboard
- Quotes Module

- **General Home Page:**

The system's landing page provides a general overview of the CRM application and its functionalities. It includes information about the system and provides links for users to navigate to different modules.

- **Admin Module:**

The Admin module allows administrators to manage all aspects of the CRM system. Administrators can add, edit, or delete users, manage user roles and permissions, access reports and analytics, and configure system settings.

- **Customer Module:**

The Customer module enables users to manage customer information and interactions. Customers can view and update their profiles, view their

activity history, create and manage leads, tasks, and contacts, and generate reports on customer interactions.

- **Activities Module:**

The Activities module allows users to schedule, track, and manage various activities such as meetings, calls, emails, and appointments. Users can view their activity calendar, set reminders, and track activity outcomes.

- **Leads Module:**

The Leads module enables users to capture, qualify, and manage potential sales opportunities. Users can add new leads, track lead status and progress through the sales pipeline, assign leads to sales representatives, and analyze lead conversion rates.

- **Tasks Module:**

The Tasks module allows users to create, assign, and track tasks related to customer interactions and sales activities. Users can prioritize tasks, set due dates, assign tasks to team members, and monitor task completion.

- **Contacts Module:**

The Contacts module stores and manages information about customers, prospects, and other contacts. Users can add new contacts, view contact details, track communication history, and segment contacts for targeted marketing campaigns.

- **Dashboard:**

The Dashboard provides users with a visual overview of key performance metrics, sales pipeline status, upcoming activities, and other important insights. Users can customize their dashboard to display relevant information and track progress towards goals.

- **Invoice Module:**

The Invoice module allows users to create, send, and track invoices for

products or services sold to customers. Users can generate invoices, manage payment statuses, track overdue invoices, and reconcile payments received.

- **Quotes Module:**

The Quotes module enables users to create and manage price quotes and proposals for potential sales deals. Users can generate quotes, customize pricing, send quotes to customers for approval, and track quote status and conversions.

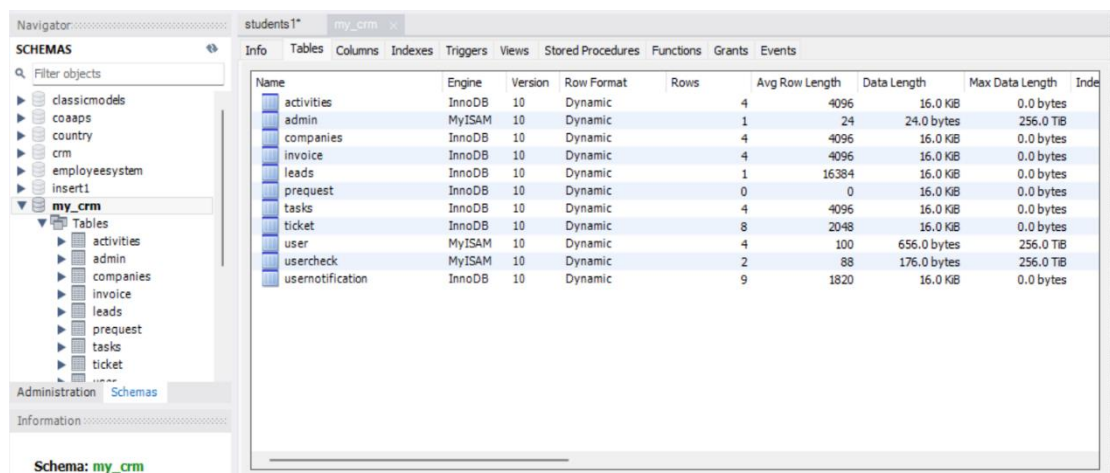
Profile Module:

The Profile module allows users to view and update their personal information, preferences, and settings. Users can manage their account details, update contact information, and configure notification preferences.

- **Authentication Module:**

The Authentication module handles user authentication and access control. Users must log in with their username and password to access the CRM system. Role-based access controls ensure that users only have access to modules and functionalities relevant to their roles.

7.2 The Presentation of Results/ Screen Shots



Name	Engine	Version	Row Format	Rows	Avg Row Length	Data Length	Max Data Length	Index
activities	InnoDB	10	Dynamic	4	4096	16.0 KiB	0.0 bytes	
admin	MyISAM	10	Dynamic	1	24	24.0 bytes	256.0 TiB	
companies	InnoDB	10	Dynamic	4	4096	16.0 KiB	0.0 bytes	
invoice	InnoDB	10	Dynamic	4	4096	16.0 KiB	0.0 bytes	
leads	InnoDB	10	Dynamic	1	16384	16.0 KiB	0.0 bytes	
prequest	InnoDB	10	Dynamic	0	0	16.0 KiB	0.0 bytes	
tasks	InnoDB	10	Dynamic	4	4096	16.0 KiB	0.0 bytes	
ticket	InnoDB	10	Dynamic	8	2048	16.0 KiB	0.0 bytes	
user	MyISAM	10	Dynamic	4	100	656.0 bytes	256.0 TiB	
usercheck	MyISAM	10	Dynamic	2	88	176.0 bytes	256.0 TiB	
usernotification	InnoDB	10	Dynamic	9	1820	16.0 KiB	0.0 bytes	

Fig : 7.2.1 Database Table Structure



Fig : 7.2.2 Homepage



Fig : 7.2.3 Logins

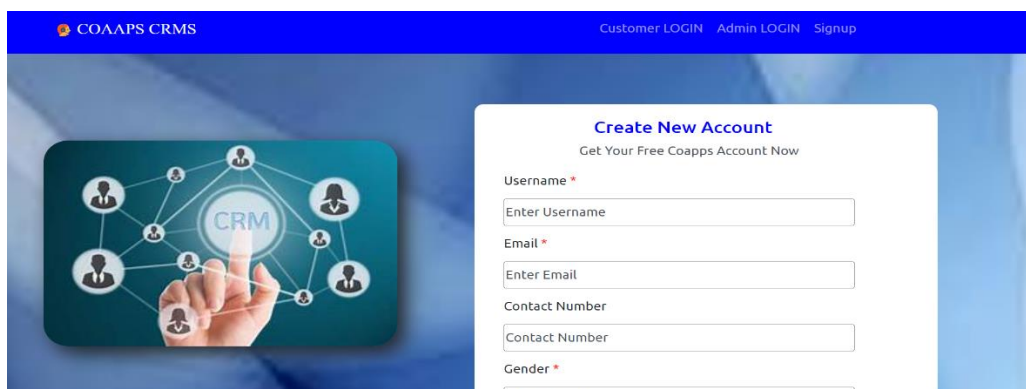


Fig : 7.2.4 Signup

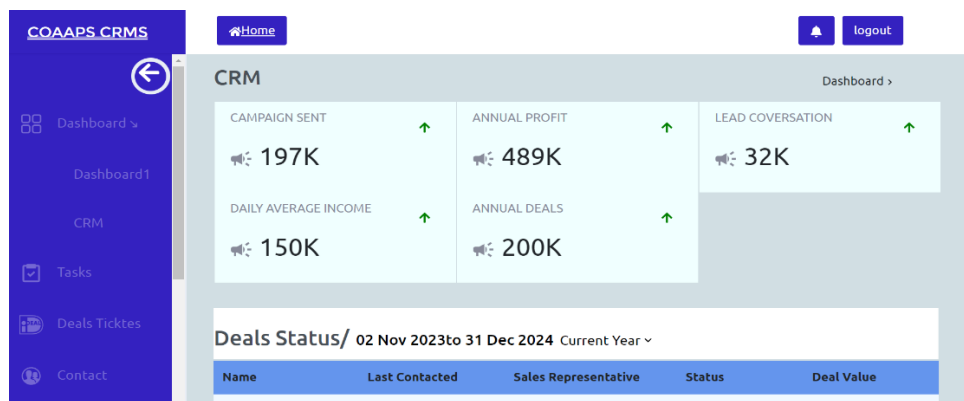


Fig : 7.2.5 Admin Dashboard

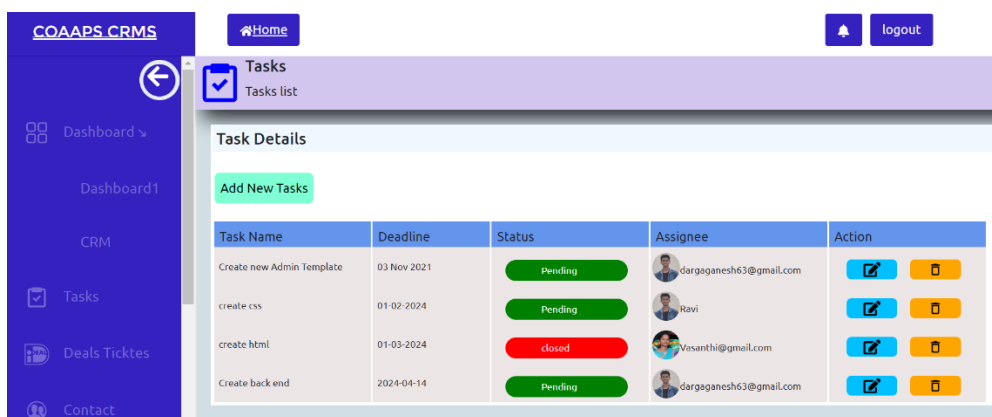


Fig 7.2.6 Task Module

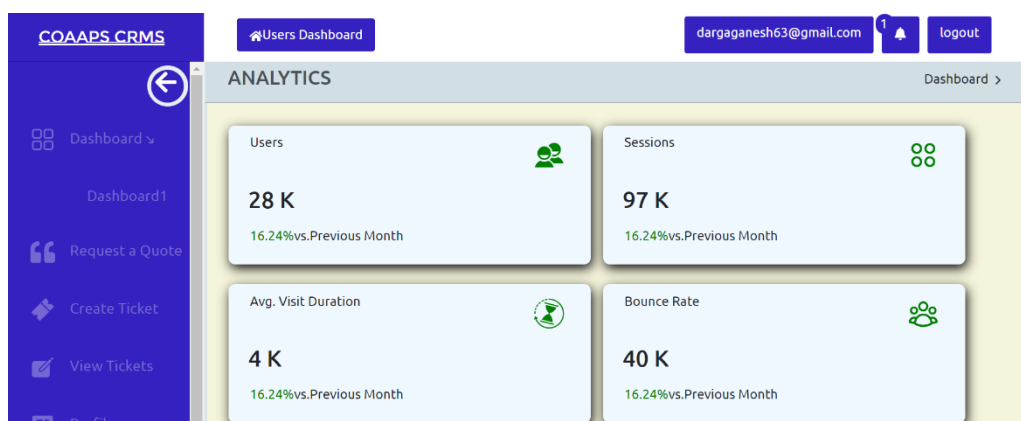


Fig 7.2.7 User Dashboard

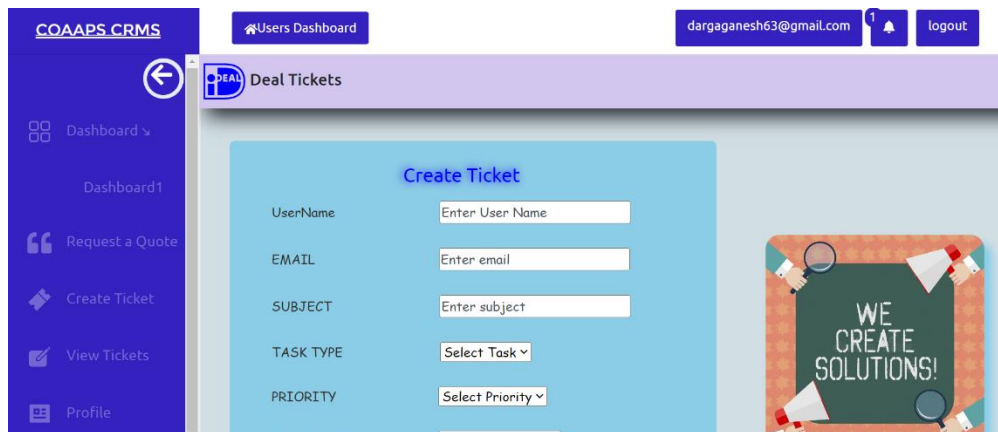


Fig 7.2.8 Creating Ticket

7.3 System Evaluation

System evaluation involves assessing the effectiveness, efficiency, usability, and overall performance of a CRM (Customer Relationship Management) system to determine its success in meeting organizational goals and user needs.

- **Effectiveness:**

User Satisfaction: Gather feedback from users through surveys, interviews, or feedback forms to assess their satisfaction with the CRM system. Evaluate user perceptions of ease of use, functionality, and overall experience.

- **Efficiency:**

Time and Resource Savings: Measure the time and resources saved by using the CRM system compared to manual processes or legacy systems. Assess efficiency gains in tasks such as data entry, lead management, and reporting.

Workflow Optimization: Evaluate the efficiency of workflows and processes within the CRM system. Identify bottlenecks, redundancies, or areas for improvement and implement optimizations to streamline operations.

- **Usability:**

User Interface Design: Evaluate the user interface design of the CRM system for clarity, consistency, and intuitiveness. Assess navigation, layout, and visual elements to ensure ease of use and accessibility for users.

- **Security:**

Data Protection: Evaluate the security measures implemented to protect sensitive customer data stored in the CRM system. Assess encryption, access controls, and data masking techniques to prevent unauthorized access or data breaches.

Compliance: Ensure that the CRM system complies with relevant data protection regulations and industry standards, such as GDPR, HIPAA, or PCI DSS. Conduct audits and assessments to verify compliance with legal and regulatory requirements.

- **Performance:**

System Performance: Evaluate the responsiveness and reliability of the CRM system under normal and peak load conditions. Measure system uptime, response times, and throughput to ensure optimal performance.

Scalability: Assess the CRM system's ability to scale with growing data volumes, user loads, and system complexity. Test scalability by simulating increased usage and monitoring system performance metrics.

Feedback and Iteration: Use feedback from users, stakeholders, and system metrics to identify areas for improvement and optimization. Implement iterative updates and enhancements to address user needs, technology advancements, and evolving business requirements.

Monitoring and Maintenance: Establish monitoring mechanisms and maintenance routines to proactively identify and address issues in

the CRM system. Monitor system performance, data quality, security posture, and user feedback to ensure ongoing reliability.

CHAPTER 8

CONCLUSION

Summary

CRM (Customer Relationship Management) is a strategy and software solution used by businesses to manage interactions with current and potential customers. It centralizes customer data, streamlines sales, marketing, and customer service processes, and enhances customer engagement. By providing a unified view of customer information, CRM helps businesses personalize interactions, improve customer satisfaction, and increase sales. Key features include contact management, lead tracking, opportunity management, and reporting. CRM systems can be cloud-based or on-premises, offering flexibility in deployment. They often integrate with other business systems such as ERP and marketing automation tools. Implementation involves assessing needs, selecting a suitable CRM solution, customization, data migration, and user training. Evaluation focuses on effectiveness, efficiency, usability, security, performance, and continuous improvement. Ultimately, CRM empowers businesses to build stronger customer relationships, drive revenue growth, and stay competitive in today's market.

Future Work

- **Advanced AI Integration:** Enhancing CRM systems with AI capabilities for predictive analytics, personalized recommendations, and automated customer interactions to improve sales forecasting, marketing campaigns, and customer service efficiency.
- **Omni-Channel Integration:** Integrating CRM with various communication channels such as social media, messaging apps, and IoT devices to provide seamless customer experiences across multiple touchpoints and channels.

- **Blockchain Technology:** Leveraging blockchain for secure and transparent management of customer data, transactions, and contracts, ensuring data integrity, privacy, and compliance with regulations such as GDPR.
- **Virtual and Augmented Reality:** Incorporating VR and AR technologies into CRM systems for immersive customer experiences, virtual product demos, and interactive training sessions, enhancing engagement and sales conversion rates.
- **Voice-enabled Interfaces:** Implementing voice recognition and natural language processing (NLP) capabilities to enable voice-controlled CRM interactions, voice search, and voice-based customer support, catering to the growing trend of voice-first technology.

Conclusion

The objective of our project has been successfully realized through the development of a comprehensive Customer Relationship Management (CRM) system. Utilizing a combination of React.js, JavaScript, HTML, CSS, and Node.js, we have designed and implemented a robust solution tailored to meet the needs of modern businesses.

Our CRM system serves as a centralized platform for managing customer interactions, sales processes, and marketing campaigns, facilitating improved customer engagement and driving business growth. By incorporating features such as user authentication, data management, and reporting functionalities, we have created a versatile tool for businesses to enhance customer relationships and streamline operations.

The CRM system offers a seamless user experience, with dynamic and responsive interfaces powered by React.js. Leveraging Node.js for the backend ensures scalability and flexibility, allowing the system to adapt to evolving business requirements and handle growing data volumes effectively.

Overall, our CRM project holds significant potential to impact business success by enabling organizations to build stronger customer relationships, improve sales efficiency, and make informed strategic decisions. Through continuous evaluation, feedback, and iteration, we are committed to refining and enhancing our CRM system to deliver maximum value to our users and stakeholders.

References

- Ayyagari, M. R. (2019) 'A Framework for Analytical CRM Assessments Challenges and Recommendations', International Journal of Business and Social Science. doi: 10.30845/ijbss.v10n6p2.
- Scullin et al. (2002). Department of Computer Systems and Computation, Universitat Politècnica de Valencia, Valencia, Spain
- Chen and Popovich (Department of Operations Management and Business Statistics, College of Business Administration, Cleveland State University, Cleveland, Ohio, USA)
- Berry, I. et al. (1995) 'Berry et al 1995.pdf', NeuroImage.
- Buttle, F. (2008) Customer relationship management (Manajemen Hubungan Pelanggan), Customer Relationship Management: Second Edition.
- Chiguvu, D. and Mahambo, C. (2021) 'The Psychometric Properties of Customer Relationship Management Framework in the Local Government Authorities in Zimbabwe', Dutch Journal of Finance and Management. doi: 10.21601/djfm/9354.
- Fahey, L. et al. (2001) 'Linking e-business and operating processes: The role of knowledge management', IBM Systems Journal. doi: 10.1147/sj.404.0889.
- Journal, A. and Basic, O. F. (2016) 'Australian Journal of Basic and', 10(April), pp.