DEBT COLLECTION MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

UMANG MISTRY

200170116025

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

In

Information Technology Engineering

VISHWAKARMA GOVERNMENT ENGINEERING COLLEGE

CHANDKHEDA, AHMEDABAD

Gujarat Technological University, Ahmedabad

April, 2024

VISHWAKARMA GOVERNMENT ENGINEERING COLLEGE CHANDKHEDA, AHMEDABAD

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **Debt Collection Management System** has been carried out by **Umang Mistry(Enrollment No: 200170116025)** under my guidance in partial fulfillment for the degree of Bachelor of Engineering in Information Technology Engineering, 8th Semester of Gujarat Technological University, Ahmadabad during the academic year 2023-24.

Prof. Dhaval Varia	Prof. Vibha D. Patel
Internal Guide	Head of the Department

OFFER LETTER

DocuSign Envelope ID: 74AC73F4-C141-4A9A-A11B-667E2643C3E8

codal

06th September 2023

Re: Offer of Training at Codal Systems Private Limited ("Company")

Dear Umang,

Congratulations! We are delighted to extend an offer of Training to you at the Company. We feel that your skills and background will be valuable assets in further enhancing and advancing the Company's business position and organization objectives. We have the pleasure of offering you a role in accordance with the following:

Position: The Company offers you the position of Trainee Software Engineer. The nature of the work, shift, role, and responsibilities will be as discussed during your interview.

Place of Work: You will be asked to report to the office in Ahmedabad.

Start Date: The Company requests you commence Training on 01st January 2024.

On your start date, you will be required to complete the joining formalities of the Company including but not limited to the execution of your Training agreement with the Company. The Training agreement will include the terms and conditions of your Training with the Company. Unless you have already provided these details, you are required to bring the following documents as a part of your joining formalities on your start date:

1. Valid ID proof (PAN card/ voter ID card/ passport/ driving license);
2. Passport size photographs.
3. Copy of Aadhar Card
4. Last semester copy

DocuSign Envelope ID: 74AC73F4-C141-4A9A-A11B-667E2643C3E8

Please note that a complete, accurate, adequate, and truthful submission of information is necessary for your successful on-boarding. Your employment is contingent upon completion of background screening checks.

If you have any questions, please feel free to get in touch with us!

All the best, and we hope to see you soon.

For and on behalf of Codal Systems Pvt. Ltd.

Mamta Goyal

Mamta Goyal
Director

ACKNOWLEDGEMENT

Read and accepted by

Trainee Name: Umang Mistri

Signature: _[candidate-sign-1]__

Date: 9/12/2023

COMPLETION CERTIFICATE

DocuSign Envelope ID: 27108A3D-8671-424D-A9F1-5A83CC5B0431

Date: 18th April, 2024
Name: Umang Mistry
Location: Ahmedabad

TO WHOM IT MAY CONCERN

This is to certify that **Mr. Umang Mistry**, IT student of Vishwakarma Government Engineering College, Gujarat Technological University is pursuing a six (6) months internship at Codal Systems Pvt. Ltd., Ahmedabad which was started from **8th January, 2024** and will be completed on **8th July, 2024.**

The project entitled **"Debt Collection Management System"** embodies the work done by Mr. Umang Mistry under the guidance of Mr. Arpit Patel during his above mentioned training period.

Till date 18th April, 2024 we have found him to be regular and diligent in his duties and responsibilities.

We wish him all the best for future assignments.

Regards,

Mamta Goyal

Authorized Signatory
For, Codal Systems Private Limited

VISHWAKARMA GOVERNMENT ENGINEERING COLLEGE
CHANDKHEDA, AHMEDABAD

DECLARATION

We hereby declare that the Project report submitted along with the Project entitled **Debt Collection Management System** submitted in partial fulfillment for the degree of Bachelor of Engineering in Information Technology to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me at Codal Inc under supervision of **Mr. Arpit Patel** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Mistry Umang Pareshbhai

Name of Student Sign of Student

ACKNOWLEDGEMENT

I would like to express our deepest appreciation to all those who provided us with the possibility to complete this report. A special gratitude we give to our Internal Guide **Prof. Dhaval Varia** whose contribution in stimulating suggestions and encouragement helped us to coordinate our project, especially in writing this report.

I express our sincere gratitude to our H.O.D. **Prof. Vibha D. Patel** for his support in our project completion. He provided us with immense knowledge. We are also thankful to our Principal **Dr. V. S. Purani** for providing us with great facilities for our project.

I would also like to extend my heartful gratitude to **Codal Systems Private Limited** for the internship opportunity. The experience and knowledge in the field have been invaluable in helping me to gain a deeper understanding of the industry. Company's constructive feedback and guidance have enabled me to develop my skills and become a more effective contributor to the organization.

In addition, my focus and dedication to learn something new helped me to clinch success in the internship and learn a lot of things and get hands on experience on live projects.

ABSTRACT

During my tenure as a **Software Engineer Trainee**, I focused on frontend development, honing my skills through practical contributions to real-world projects. My primary focus revolved around ReactJS, where I actively participated in the development of a **Debt Collection Management System** and an **E-commerce Application**. Embracing agile methodologies, I collaborated closely with experienced developers, engaging in various tasks such as user interface design, feature development, and frontend performance optimization. These experiences not only bolstered my proficiency in ReactJS but also instilled in me a comprehensive understanding of frontend development principles. By the conclusion of my internship, I had solidified my foundation in frontend development with ReactJS, equipped with the skills and knowledge necessary for a successful career in frontend development.

LIST OF FIGURE

Figure 1. 1 Organization Chart 2
Figure 3. 1 VS Code 5
Figure 3. 2 GitHub 5
Figure 3. 3 Jira 6
Figure 3. 4 HTML 6
Figure 3. 5 CSS 6
Figure 3. 6 JavaScript 7
Figure 3. 7 TypeScript 7
Figure 3. 8 ReactJS 7
Figure 3. 9 Redux 8
Figure 3. 10 NodeJS 8
Figure 3. 11 ExpressJS 8
Figure 3. 12 MYSQL 9
Figure 3. 13 Bookshelf ORM 9
Figure 3. 14 REST 10
Figure 3. 15 Ant Design 10
Figure 4. 1 React 13
Figure 4. 2 Django REST 13
Figure 4. 3 PostgreSQL 14
Figure 4. 4 Redis 14
Figure 4. 5 KendoUI 15
Figure 6. 1 Use Case Diagrame 24
Figure 6. 2 Data-Flow Diagram 25
Figure 6. 3 Process Diagram 26
Figure 6. 4 ER Diagram 27
Figure 7. 1 HOME Page 31
Figure 7. 2 Client Page 32
Figure 7. 3 Role Management 32
Figure 7. 4 Banking Page 33
Figure 7. 5 Invoice History 33
Figure 7. 6 Remittance History 34
Figure 7. 7 Quickbooks Deposits 34

LIST OF TABLE

Table 6. 1 Client Table....28
Table 6. 2 Debtor Table....28
Table 6. 3 Account Table....29
Table 6. 4 Debt Management Table....29
Table 6. 5 Debt Collection Table....30
Table 6. 6 Invoice Table....30
Table 8. 1 Test Case....36

TABLE OF CONTENT

DECLARATION i

ACKNOWLEDGEMENT ii

ABSTRACT iii

LIST OF FIGURE iv

LIST OF TABLE v

TABLE OF CONTENT vi

CHAPTER:1 OVERVIEW OF THE COMPANY 1

1.1 HISTORY 1

1.2 SCOPE OF WORK 1

1.2.1 eCommerce 1

1.2.2 Legacy Modernization 1

1.2.3 Product Strategy and Innovation 1

1.2.4 Data and AI: 2

1.3 ORGANIZATION CHART 2

1.4 CAPACITY OF COMPANY 2

CHAPTER:2 OVERVIEW OF DIFFERENT DEPARTMENTS 3

2.1 DEPARTMENTS 3

2.1.1 Development Team 3

2.1.2 Design Team 3

2.1.3 Project Management 3

2.1.4 Quality Assurance (QA) Team 3

CHAPTER :3 INTRODUCTION TO INTERNSHIP 4

3.1 INTERNSHIP SUMMARY 4

3.2 INTERNSHIP PLANNING AND APPROACH 4

3.2.1 Learning Period 4

3.2.2 Implementation Period 4

3.3 TOOLS AND TECHNOLOGIES 5

3.4 DEMO PROJECT 10

CHAPTER :4 INTRODUCTION OF PROJECT 11

4.1 INTRODUCTION 11

4.2 PURPOSE 11

4.3 OBJECTIVE 12

4.3.1 Optimize Debt Collection Processes 12

4.3.2 Enhance User Experience 12

4.3.3 Empower Data-Driven Decision-Making 12
4.3.4 Improve Debt Recovery Efficiency 12
4.3.5 Ensure Compliance and Data Security 12
4.3.6 Drive Organizational Excellence 12
4.4 TECHNOLOGY 13
4.4.1 React 13
4.4.2 Django Rest Framework 13
4.4.3 PostgreSQL 14
4.4.4 Redis 14
CHAPTER :5 SYSTEM ANALYSIS 16
5.1 STUDY OF THE CURRENT SYSTEM 16
5.2 WEAKNESS AND PROBLEM OF THE CURRENT SYSTEM 17
5.3 REQUIREMENTS FOR THE NEW SYSTEM 18
5.4 SYSTEM FEASIBILITY 19
5.5 REQUIREMENTS 19
5.5.1 Functional Requirements 19
5.5.2 Non-Functional Requirements 21
5.5.3 HARDWARE AND SOFTWARE REQUIREMENTS 23
CHAPTER :6 SYSTEM DESIGN 24
6.1 USE CASE DIAGRAM 24
6.2 DFD DIAGRAM 25
6.3 PROCESS DIAGRAM 26
6.4 ER DIAGRAM 27
6.5 DATA DICTIONARY 28
CHAPTER :7 SYSTEM IMPLEMENTATION 31
7.1 HOME PAGE 31
7.2 CLIENT PAGE 32
CHAPTER :8 SYSTEM TESTING 35
8.1 TESTING PLAN 35
8.2 TEST CASE AND RESULT 35
CHAPTER :9 CONCLUSION 37
9.1 FUTURE ENHANCEMENT 37
9.1.1 Advanced Reporting and Analytics: 37
9.1.2 Real-time Notifications using Firebase: 37
9.1.3 Online Payment Integration: 37
9.2 SUMMARY OF INTERNSHIP 37

9.3 CONCLUSION 38

CHAPTER:1 OVERVIEW OF THE COMPANY

1.1 HISTORY

Founded in 2009, Codal is an award-winning web design and development consultancy, specializing in digital transformation, eCommerce, UX/UI design, product strategy, data analytics, and AI. Our global team is committed to delivering innovative, data-driven digital products, supported by the unique perspectives and wide-ranging expertise of our clients, partners, and employees.

With offices in the US, the UK, Canada, and India, we work with enterprise organizations to solve complex business problems and discover new growth opportunities. Whether the goal is to modernize a legacy application, boost customer engagement, centralize data, or all of the above, Codal plans and executes tailored strategies for clients across industries—where we design, develop, test, launch, and maintain impactful, forward-thinking digital solutions.

1.2 SCOPE OF WORK

1.2.1 eCommerce: Codal dives deep into your existing website and technological infrastructure, analyzing the front- and back-end tools and workflows that power your eCommerce experience. Through tech stack audits, market research, stakeholder interviews, data analysis, and other exercises, our team is able to develop a roadmap to success that guarantees longevity for your business.

1.2.2 Legacy Modernization: Our team migrates your enterprise from its outdated legacy systems to modern solutions with a microservices-based architecture.

1.2.3 Product Strategy and Innovation: Our team creates and optimizes the delivery of business services, adding value for both your customers and employees. Codal's human-centered approach brings together various stakeholders, customers, employees, and partners to define, create, design, and deploy services that meet user needs and expectations.

1.2.4 Data and AI: Codal works closely with your business to understand the key factors necessary to leverage best-in-breed AI and ML libraries, building predictive models that inform and enhance your business growth strategy.

1.3 ORGANIZATION CHART

Figure 1. 1 Organization Chart

1.4 CAPACITY OF COMPANY

Codal is a dynamic digital solutions provider with a robust team of professionals spanning developers, designers, project managers, and IT experts. Our extensive experience encompasses collaborating with clients across diverse industries on a global scale. Leveraging our expertise in cutting-edge technologies including PHP, Python, React, Next, Node.js, and beyond, we deliver innovative and tailored solutions to meet the unique needs of each client.

CHAPTER:2 OVERVIEW OF DIFFERENT DEPARTMENTS

2.1 DEPARTMENTS

2.1.1 Development Team: The Development Team is responsible for coding, programming, and software development. They work closely with other departments to ensure that the developed software meets the design and functionality requirements. This coordination involves regular communication, sharing progress updates, addressing issues, and incorporating feedback from other teams.

2.1.2 Design Team: The Design Team focuses on UX/UI design, graphic design, and branding aspects of the project. They collaborate with the Development Team to translate design concepts into functional software interfaces. This coordination involves providing design assets, guidelines, and feedback to ensure that the final product aligns with the intended user experience and branding guidelines.

2.1.3 Project Management: Project Management oversees the planning, scheduling, and coordination of the entire development process. They facilitate communication and collaboration among different departments, allocate resources, track progress, and mitigate risks to ensure project success. This coordination involves conducting regular meetings, updating project plans, and resolving any conflicts or issues that may arise during development.

2.1.4 Quality Assurance (QA) Team: The QA Team is responsible for ensuring the quality and functionality of digital products through testing and feedback. They work closely with the Development Team to identify, report, and address any bugs, errors, or inconsistencies in the software. This coordination involves conducting various types of testing, including functional testing, performance testing, and usability testing, and providing feedback to developers for continuous improvement.

CHAPTER :3 INTRODUCTION TO INTERNSHIP

3.1 INTERNSHIP SUMMARY

Throughout my internship, I immersed myself in a dynamic tech stack, honing my skills in TypeScript, React, Next.js, Node.js, Express.js, and database management with MySQL and MongoDB. A significant focus was on seamless API integration, ensuring robust communication between frontend and backend systems. Moreover, I actively engaged with live tickets, applying my knowledge to real-world scenarios and contributing to the development of live projects.

3.2 INTERNSHIP PLANNING AND APPROACH

My internship is divided into 2 parts:

1. Learning Period
2. Implementation Period.

3.2.1 Learning Period

The learning period was the time taken to learn various languages needed for the understanding of the project. I learned Javascript, Typescript, ReactJs, Redux, NextJS, NodeJs, ExpressJs, Tailwind CSS, etc. Working in a team means collaborating on codes from various developers so for that I learned Git and GitHub. I Enhanced my Problem-Solving abilities.

3.2.2 Implementation Period

After completing our learning period, we were onboarded to live projects, where we had the opportunity to apply our newly acquired skills in real-world scenarios. Through comprehensive knowledge transfer sessions, we gained insights into the project architecture, business requirements, and best practices. Assigned tasks allowed us to collaborate closely with experienced team members, iterate on solutions, and deliver tangible results. This hands-on experience not only solidified our technical capabilities but also fostered a deep understanding of the software development lifecycle, making it a truly invaluable learning experience.

3.3 TOOLS AND TECHNOLOGIES

Tools:

3.3.1 VS Code:

Visual Studio Code is a lightweight but powerful source code editor that runs on your desktop. It comes with built-in support for JavaScript, TypeScript, and Node.js, and has a rich ecosystem of extensions for other languages such as Python, Java, and C++.

Figure 3. 1 VS Code

3.3.2 Git/GitHub:

Git is a distributed version control system used to track changes in source code during software development. GitHub is a platform built around Git, providing hosting for software development and version control using Git.

Figure 3. 2 GitHub

3.3.3 Jira:

Jira is a project management tool developed by Atlassian. It is commonly used for issue tracking, bug tracking, and project management. It offers features for agile project management, such as scrum and kanban boards.

Figure 3. 3 Jira

Technologies:

3.3.4 HTML:

HyperText Markup Language is the standard markup language for creating web pages and web applications. It defines the structure of content on a web page using elements and attributes.

Figure 3. 4 HTML

3.3.5 CSS:

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in HTML. It defines how HTML elements are displayed on screen, in print, or spoken.

Figure 3. 5 CSS

3.3.6 JavaScript (JS):

JavaScript is a programming language that is commonly used to create interactive effects within web browsers.

Figure 3. 6 JavaScript

3.3.7 TypeScript (TS):

TypeScript is a superset of JavaScript that adds optional static typing and other features to the language.

Figure 3. 7 TypeScript

3.3.8 ReactJS:

React is a JavaScript library for building user interfaces. It allows developers to create reusable UI components and manage the state of those components efficiently.

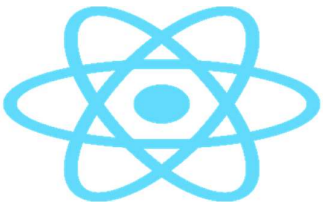

Figure 3. 8 ReactJS

3.3.9 Redux/Redux Toolkit:

Redux is a predictable state container for JavaScript apps. It helps manage the state of an application in a predictable way, making it easier to develop and debug complex applications. Redux Toolkit is the official, recommended toolset for building Redux applications.

Figure 3. 9 Redux

3.3.10 NodeJS:

Node.js is a runtime environment that allows you to run JavaScript on the server side. It is built on Chrome's V8 JavaScript engine and provides an event-driven, non-blocking I/O model that makes it lightweight and efficient.

Figure 3. 10 NodeJS

3.3.11 ExpressJS:

Express.js is a minimal and flexible Node.js web application framework that provides a robust set of features for building web and mobile applications. It is designed for building single-page, multi-page, and hybrid web applications.

Figure 3. 11 ExpressJS

3.3.12 MYSQL:

MySQL is an open-source relational database management system. It uses a structured query language (SQL) to manage and manipulate data stored in relational databases.

Figure 3. 12 MYSQL

3.3.13 Bookshlef ORM:

Bookshelf.js is a promise-based ORM for Node.js, built on top of the Knex.js SQL query builder. It is designed to work with PostgreSQL, MySQL, and SQLite databases, providing easy-to-use abstractions for interacting with databases in Node.js applications.

Figure 3. 13 Bookshelf ORM

3.3.14 REST APIs:

Representational State Transfer (REST) is a software architectural style that defines a set of constraints for building web services. RESTful APIs use HTTP requests to perform actions on resources, typically using GET, POST, PUT, and DELETE methods.

Figure 3. 14 REST

3.3.15 Ant Design:

Ant Design is a comprehensive design system and React UI library that facilitates the rapid development of user interfaces. It offers a wide range of components and utilities, allowing developers to build elegant and responsive designs with ease.

Figure 3. 15 Ant Design

3.4 DEMO PROJECT

During my internship, I implemented an E-Commerce Platform project that included key features such as a Products Page, Product Detail Page, Cart Page, Checkout Page, Payment Success Page, and an Offline Page. Additionally, I implemented user authentication for login and registration, allowing only logged-in users to add products to the cart or proceed with payment. Through this project, I gained hands-on experience in designing and developing a full-stack web application using ReactJS for the frontend, Node.js for the backend, and MYSQL for the database.

CHAPTER :4 INTRODUCTION OF PROJECT

4.1 INTRODUCTION

The Debt Collection Management System (DCMS) stands as a pivotal asset in navigating the complexities of financial management within modern business landscapes. Tailored to streamline critical debt administration operations, this system revolutionizes how debts are monitored, tracked, and resolved. By harnessing cutting-edge technologies and intricate functionalities, it sets a new benchmark for operational efficiency in debt management, transcending conventional approaches.

At its core, the DCMS serves as a cornerstone for organizational efficacy, delicately balancing the intricate dynamics between debtor accounts and client relationships. Seamlessly integrating state-of-the-art tools, it simplifies the once arduous task of debt handling into a user-friendly and intuitive process. Moreover, it offers robust reporting capabilities, empowering stakeholders with comprehensive insights into debt performance metrics and financial trends.

4.2 PURPOSE

The purpose of the Debt Collection Management System (DCMS) project is to revolutionize debt administration by providing a comprehensive and user-friendly platform. By leveraging cutting-edge technologies, the DCMS simplifies critical debt management processes such as debtor monitoring, transaction handling, and reporting. This project aims to enhance operational efficiency, streamline financial workflows, and empower users with actionable insights for informed decision-making. Additionally, it ensures regulatory compliance, improves debt collection effectiveness, and facilitates organizational growth by embracing digital innovation in financial management practices.

4.3 OBJECTIVE

4.3.1 Optimize Debt Collection Processes:

Efficiently streamline critical debt management tasks, including debtor monitoring, transaction handling, and reporting, to enhance operational efficiency and minimize administrative overhead.

4.3.2 Enhance User Experience:

Prioritize user-centric design principles to deliver a seamless and intuitive experience for debt collection personnel, clients, and debtors, thereby improving satisfaction and engagement across stakeholders.

4.3.3 Empower Data-Driven Decision-Making:

Equip stakeholders with comprehensive reporting capabilities and insightful analytics to facilitate data-driven decision-making, enabling strategic debt collection planning and performance evaluation.

4.3.4 Improve Debt Recovery Efficiency:

Leverage advanced technologies and data analytics to optimize debt recovery strategies, identify high-priority accounts, and expedite the resolution process, ultimately reducing outstanding debts and improving financial outcomes.

4.3.5 Ensure Compliance and Data Security:

Implement stringent security measures and compliance protocols to protect sensitive financial data and ensure adherence to regulatory requirements, safeguarding the integrity and confidentiality of debt-related information.

4.3.6 Drive Organizational Excellence:

Act as a catalyst for organizational excellence by fostering a culture of continuous improvement and innovation in debt collection practices, enabling enterprises to adapt to changing market dynamics and achieve sustainable growth.

4.4 TECHNOLOGY

4.4.1 React:

React is a JavaScript library for building user interfaces, widely known for its declarative and component-based approach. It allows developers to create interactive and dynamic web applications with ease by efficiently managing the UI state and rendering updates as data changes. React's virtual DOM and efficient reconciliation algorithm ensure optimal performance, making it ideal for building modern and responsive web interfaces.

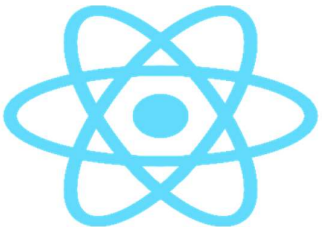

Figure 4. 1 React

4.4.2 Django Rest Framework:

Django Rest Framework (DRF) is a powerful toolkit for building Web APIs with Django, a high-level Python web framework. DRF simplifies the process of creating RESTful APIs by providing a set of reusable components and conventions for serializing and deserializing data, handling authentication, and implementing CRUD operations. It follows the principles of DRY (Don't Repeat Yourself) and encourages best practices in API design, making it a preferred choice for building scalable and maintainable APIs in Python-based web applications.

Figure 4. 2 Django REST

4.4.3 PostgreSQL:

PostgreSQL is an open-source relational database management system (RDBMS) known for its advanced features, extensibility, and reliability. It offers ACID compliance, support for complex data types, and powerful indexing capabilities, making it suitable for handling large volumes of data and complex queries. PostgreSQL's robust security features, including row-level security and SSL encryption, ensure data integrity and confidentiality, making it a preferred choice for storing sensitive information in web applications.

Figure 4. 3 PostgreSQL

4.4.4 Redis:

Redis is an in-memory data structure store known for its high performance, scalability, and versatility. It supports various data structures such as strings, lists, sets, and hashes, making it suitable for a wide range of use cases, including caching, session management, and real-time analytics. Redis's persistence options and replication capabilities ensure data durability and fault tolerance, making it a popular choice for building scalable and resilient web applications.

Figure 4. 4 Redis

4.4.5 Kendo-React:

KendoReact is a comprehensive library of React UI components designed to accelerate the development of modern web applications. It provides a wide range of customizable and accessible components, including grids, charts, inputs, and buttons, to streamline the creation of rich and interactive user interfaces. KendoReact's components are built with performance and responsiveness in mind, ensuring smooth user experiences across devices and browsers. With its extensive documentation and support for internationalization and accessibility standards, KendoReact is a versatile tool for building sophisticated web applications with React

Figure 4. 5 KendoUI

CHAPTER :5 SYSTEM ANALYSIS

5.1 STUDY OF THE CURRENT SYSTEM

5.1.1 CollectMax System:

CollectMax is a prominent debt collection management system (DCMS) recognized for its comprehensive features that streamline debt recovery processes. Its key features include robust debtor tracking, detailed debtor information recording, payment history tracking, and follow-up action scheduling.

The system's payment processing capabilities enable secure and efficient transactions, enhancing user experience by providing a convenient payment platform for debtors. CollectMax's reporting features offer valuable insights into debt collection activities, allowing organizations to optimize their strategies.

Moreover, CollectMax offers integration with various communication tools, such as email and phone systems, enabling debt collectors to communicate with debtors effectively. This integration enhances the efficiency of debt collection efforts by facilitating timely and targeted communication, increasing the likelihood of successful debt recovery.

Despite its strengths, CollectMax has been criticized for its complexity and lack of user-friendliness. Some users have reported difficulties in navigating the system and understanding its various features, which can hinder its effectiveness. Continuous improvement and investment in user training and system updates are recommended to address these challenges and ensure that CollectMax remains a powerful tool in the debt collection industry.

In conclusion, CollectMax offers a wide range of features designed to streamline debt recovery processes. Its robust debtor tracking, payment processing, and reporting functionalities make it a valuable asset for organizations looking to improve their debt collection strategies. While its complexity may present challenges, continuous improvement and user training can help maximize its effectiveness and ensure that it remains a leading DCMS in the industry.

5.2 WEAKNESS AND PROBLEM OF THE CURRENT SYSTEM

5.2.1 Complexity:

Debt Collection management software often comes with a multitude of features and functionalities, making it complex for users to navigate and utilize effectively. This complexity may require extensive training and onboarding for users to fully grasp the system.

5.2.2 Customization:

While Debt collection management software may offer customization options, users may find certain features or workflows challenging to customize to their specific requirements. This limitation can hinder adaptability and efficiency in managing different types of properties or transactions.

5.2.3 Integration:

Integrating debt collection management software like CollectMax with existing systems or third-party applications can be complex and resource-intensive. The process may require specialized knowledge and expertise, posing challenges for organizations without dedicated IT resources. Compatibility issues with legacy systems or specific third-party applications can further complicate integration, potentially leading to delays and increased costs

5.2.4 Cost:

Implementing and maintaining debt collection management software can incur significant costs, including licensing fees, customization expenses, training costs, and ongoing maintenance fees.

5.2.5 Reliabilty:

Debt collection management software, like any digital system, is susceptible to technical issues, downtime, or software bugs that can disrupt operations and impact reliability. Dependence on the software for critical tasks such as property listings, client communications

.

5.3 REQUIREMENTS FOR THE NEW SYSTEM

5.3.1 Customization Limitations:

The new debt collection management system should offer extensive customization options, allowing organizations to tailor workflows, communication templates, and reporting features to their specific needs. This customization capability will ensure that the system can adapt to evolving debt collection practices and regulatory requirements.

5.3.2 Integration Challenges:

Integration with third-party applications and systems is crucial for effective debt collection management. The new system should seamlessly integrate with accounting software, communication tools, and other relevant systems to facilitate data exchange and workflow automation. This integration will help reduce data silos and improve overall operational efficiency.

5.3.3 Scalability Concerns:

The new system should have robust scalability features to accommodate future growth and evolving debt collection demands. It should be able to handle a large volume of debtors and accounts without compromising performance or responsiveness. Additionally, the system should be flexible enough to scale up or down based on the organization's needs, ensuring efficient and effective debt collection operations now and in the future.

5.3.4 Cost Considerations:

The pricing options should be flexible, allowing organizations to choose the most suitable plan based on their budget and requirements. Clear communication about pricing details and any potential additional costs will help organizations make informed decisions and avoid unexpected expenses. Overall, a competitive and transparent pricing model will contribute to the success of the new system and its adoption by organizations seeking cost-effective debt collection management solutions.

5.4 SYSTEM FEASIBILITY

5.4.1 Alignment with Organizational Objectives:

Yes, the proposed debt collection management system aligns closely with the organization's objectives of improving debt recovery processes, enhancing data management, and ensuring compliance with regulatory requirements. By streamlining debt collection operations and providing comprehensive reporting capabilities, the system contributes significantly to achieving the organization's financial goals and operational efficiency.

5.4.2 Technological and Cost Considerations:

The proposed system can be implemented using existing technology infrastructure, leveraging commonly used debt collection management technologies and platforms. Additionally, the system falls within the organization's budget constraints, with allocated funds for development, implementation, and training. While there may be initial costs involved in implementing the new system, the long-term benefits, such as improved debt recovery rates and operational efficiency, justify the investment.

5.4.3 Integration Capability:

The proposed system has been designed with integration capabilities in mind, ensuring compatibility with other systems already in place within the organization. It can seamlessly integrate with existing tools such as payroll software, attendance management systems, and performance evaluation platforms, facilitating smooth data exchange and interoperability. While there may be some minor configuration adjustments required during the integration process, overall, the system's integration capabilities are robust and adaptable.

5.5 REQUIREMENTS

5.5.1 Functional Requirements

User Authentication and Access Control:

- Users should be able to securely log in with unique credentials.

- Role-based access control should be implemented to restrict access to sensitive information based on user roles and permissions.

Accounts Module:

- Track the status of each account, including outstanding balances, payment histories, and any relevant notes or communications.
- Assign accounts to specific debt collectors or teams for personalized management.
- Generate reports to analyze account performance and identify trends or patterns.

Remittance Module:

- Automate the remittance process to ensure timely and accurate payments to clients.
- Generate remittance reports to reconcile payments and track outstanding balances.
- Integrate remittance data with accounting systems for streamlined financial management.

Invoice Module:

- Generate invoices for outstanding balances, including details such as the amount owed, due date, and payment instructions.
- Customize invoices to include client-specific information or branding.
- Track invoice status and send reminders for overdue payments

QuickBooks Module:

- Sync financial data between the debt collection management system and Quickbooks to maintain accurate financial records.
- Ensure seamless integration to avoid duplication or errors in financial reporting.
- Utilize Quickbooks' reporting capabilities to analyze financial data and track performance.

Conversions Module:

- Convert data from various file formats or databases into a format compatible with the debt collection management system.

- Ensure data integrity and accuracy during the conversion process.
- Provide tools or utilities to assist with data conversion and migration.

Transactions Module:

- Record all financial transactions related to debt collection, including payments, fees, and adjustments.
- Ensure transactions are categorized correctly for accurate reporting and accounting purposes.

Clients Module:

- Maintain a comprehensive database of client information, including contact details, billing information, and communication history.
- Assign unique identifiers to each client for easy reference and tracking.
- Store any legal documents or agreements related to the client's debt collection.

Accessibility:

- Develop a responsive web interface that is accessible on various devices, including desktops, laptops, tablets.

5.5.2 Non-Functional Requirements

Performance:

- The debt collection management system should demonstrate high performance, with quick response times for user interactions such as accessing debtor information, updating account statuses, and processing payments.
- It should be capable of handling a large volume of concurrent users and transactions without experiencing degradation in performance.

Reliability:

- The system should be highly reliable, with minimal downtime and service interruptions.

- It should have built-in redundancy and failover mechanisms to ensure uninterrupted access to critical debt collection functionalities.

Scalability:

- The system should be scalable to accommodate the organization's growth in terms of both the number of debtors and the volume of debt.
- It should be capable of handling increased user load and data processing requirements without compromising performance or reliability.

Security:

- The system should enforce stringent security measures to protect sensitive debtor information from unauthorized access, modification, or disclosure.
- It should comply with industry standards and regulations related to data security and privacy, such as encryption of data at rest and in transit.

Usability:

- The system should have a user-friendly interface that is intuitive and easy to navigate, requiring minimal training for users to become proficient.
- It should adhere to principles of accessibility to ensure that all users can effectively utilize the system.

Compatibility:

- The system should be compatible with a variety of web browsers and operating systems commonly used by debt collectors and administrators.
- It should also be compatible with different screen sizes and resolutions to provide a consistent user experience across devices.

Maintainability:

- The system should be designed with modular architecture and well-documented code to facilitate ease of maintenance and future enhancements.
- It should support seamless updates and patches to address security vulnerabilities or introduce new features.

Interoperability:

- The system should be interoperable with other software applications and systems used within the organization's IT ecosystem.
- It should support standard data exchange formats and protocols to enable seamless integration with third-party systems.

5.5.3 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENT

- 8 GB RAM
- 256 GB SSD
- Processor: Intel i5 10th generation
- Mouse: Any Compatible
- Keyboard: Any Compatible
- Display: Any Compatible
- Device with decent internet connectivity

SOFTWARE REQUIREMENT

- 64-bit operating

*Given hardware and software are selected as per the company requirements, guidelines and policies.

CHAPTER :6 SYSTEM DESIGN

6.1 USE CASE DIAGRAM

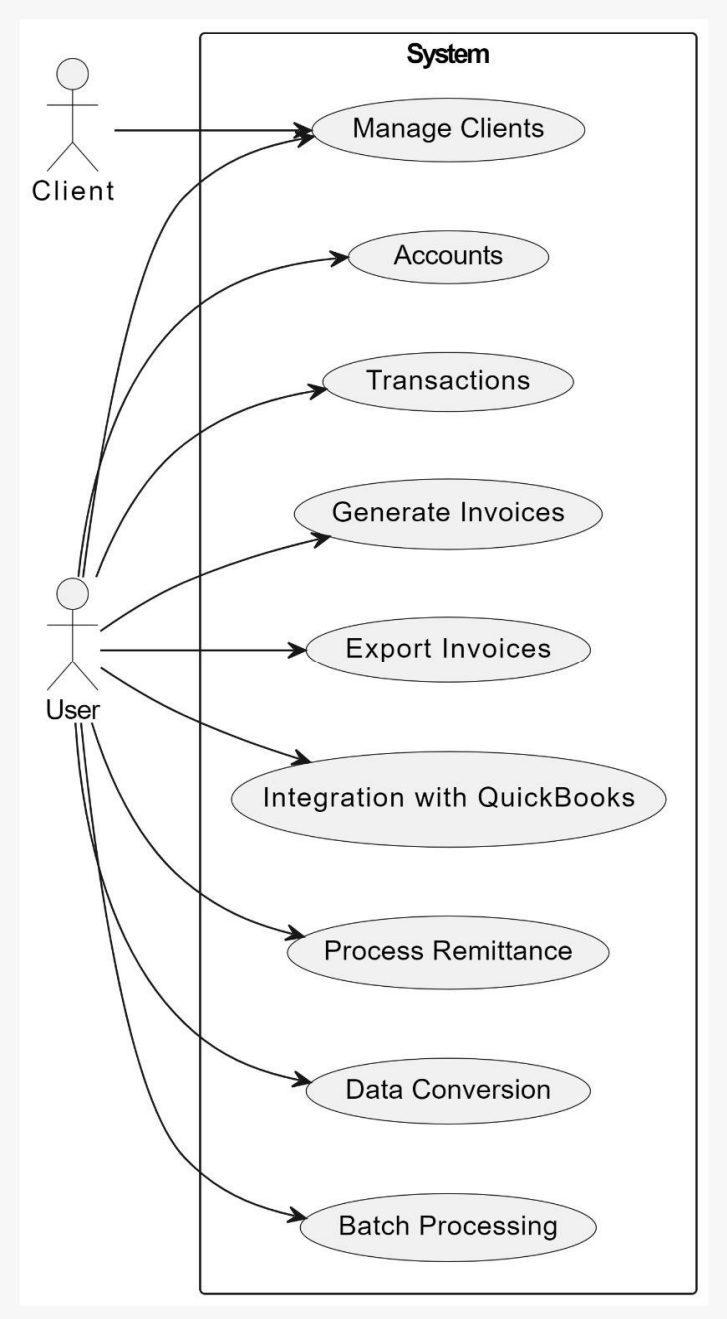

Figure 6. 1 Use Case Diagrame

6.2 DFD DIAGRAM

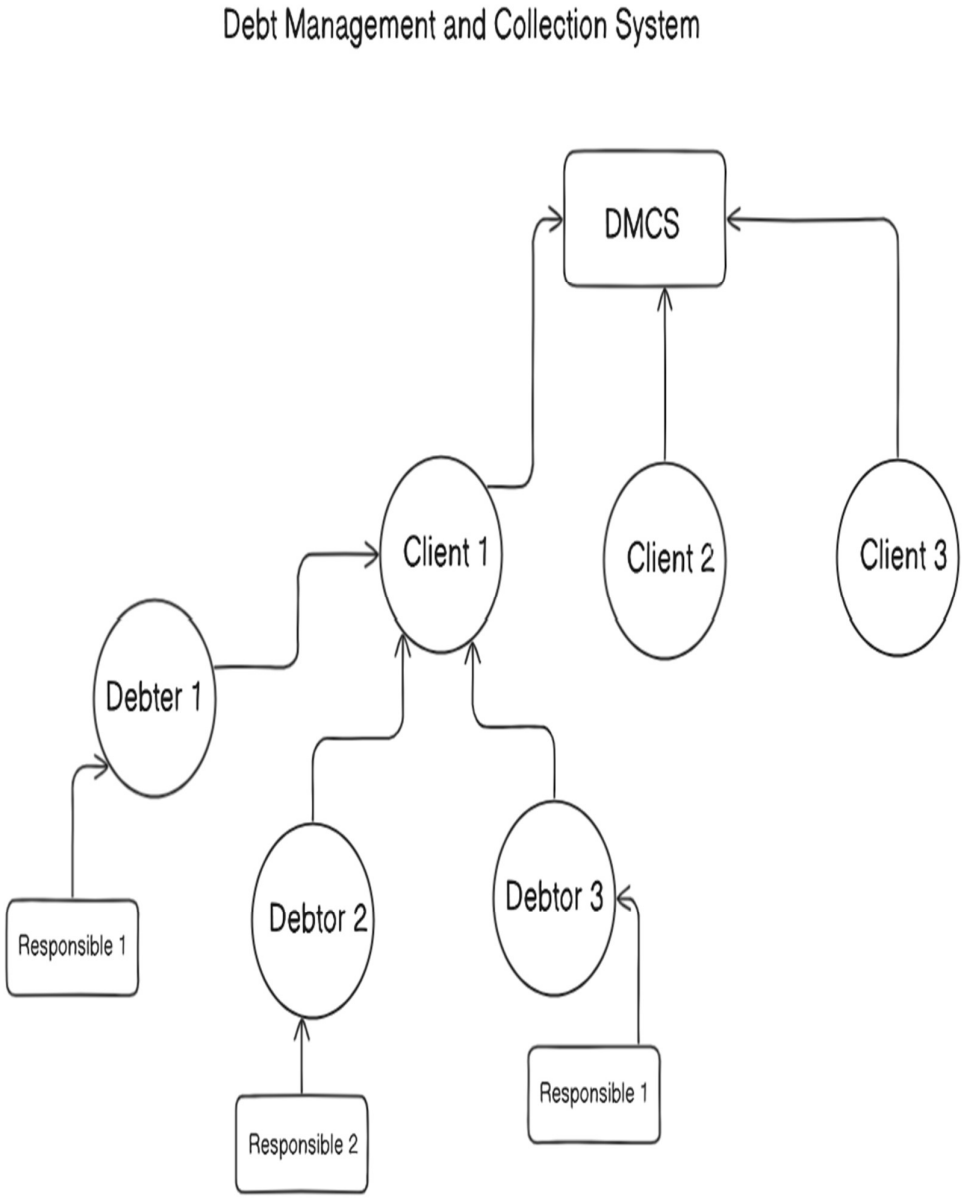

Figure 6. 2 Data-Flow Diagram

6.3 PROCESS DIAGRAM

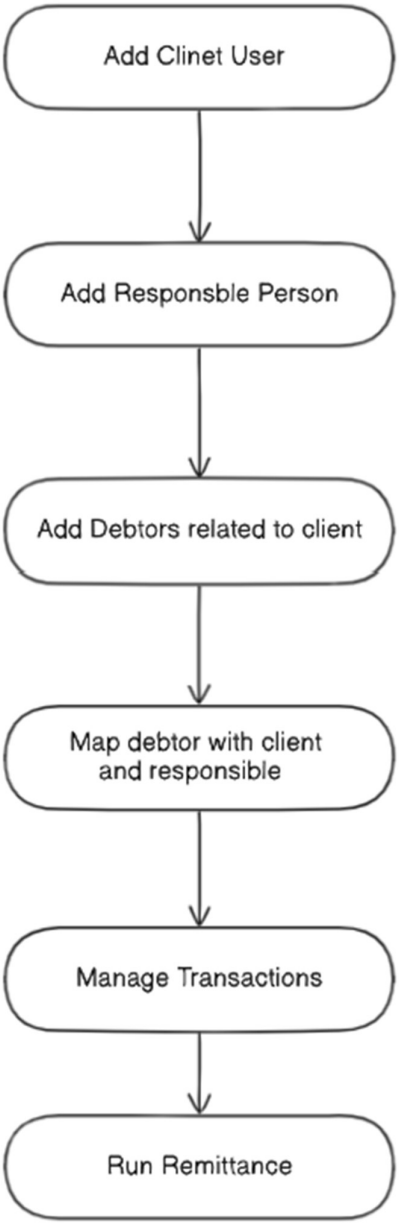

Figure 6. 3 Process Diagram

6.4 ER DIAGRAM

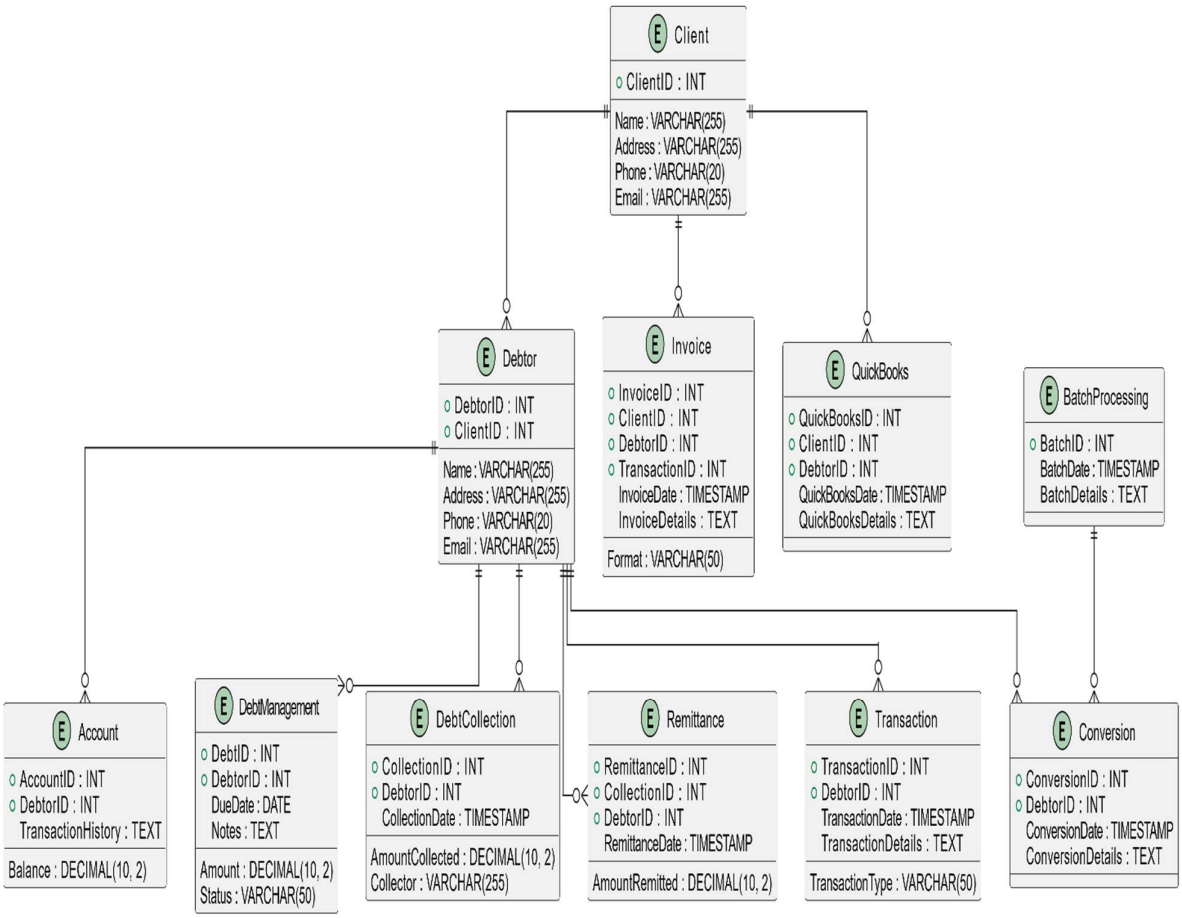

Figure 6. 4 ER Diagram

6.5 DATA DICTIONARY

Column Name	Data Type	Constraints	Description
ClientID	INT	PRIMARY KEY	Unique identifier for the client
Name	VARCHAR(255)		Name of the client
Address	VARCHAR(255)		Address of the client
Phone	VARCHAR(20)		Phone number of the client
Email	VARCHAR(255)		Email address of the client

Table 6. 1 Client Table

Column Name	Data Type	Constraints	Description
DebtorID	INT	PRIMARY KEY	Unique identifier for the debtor
ClientID	INT	FOREIGN KEY (Clients)	ID of the associated client
Name	VARCHAR(255)		Name of the debtor
Address	VARCHAR(255)		Address of the debtor
Phone	VARCHAR(20)		Phone number of the debtor
Email	VARCHAR(255)		Email address of the debtor

Table 6. 2 Debtor Table

Column Name	Data Type	Constraints	Description
AccountID	INT	PRIMARY KEY	Unique identifier for the account
DebtorID	INT	FOREIGN KEY (Debtors)	ID of the associated debtor
Balance	DECIMAL(10, 2)		Current balance of the account
TransactionHistory	TEXT		History of transactions associated with the account

Table 6. 3 Account Table

Column Name	Data Type	Constraints	Description
DebtID	INT	PRIMARY KEY	Unique identifier for the debt
DebtorID	INT	FOREIGN KEY (Debtors)	ID of the associated debtor
Amount	DECIMAL(10, 2)		Amount of the debt
DueDate	DATE		Due date for the debt
Status	VARCHAR(50)		Current status of the debt (e.g., outstanding, paid)
Notes	TEXT		Additional notes related to the debt

Table 6. 4 Debt Management Table

Column Name	Data Type	Constraints	Description
CollectionID	INT	PRIMARY KEY	Unique identifier for the collection
DebtorID	INT	FOREIGN KEY (Debtors)	ID of the associated debtor
CollectionDate	TIMESTAMP		Date and time of the collection
AmountCollected	DECIMAL(10, 2)		Amount collected during the collection
Collector	VARCHAR(255)		Name or ID of the collector

Table 6. 5 Debt Collection Table

Column Name	Data Type	Constraints	Description
InvoiceID	INT	PRIMARY KEY	Unique identifier for the invoice
ClientID	INT	FOREIGN KEY (Clients)	ID of the associated client
DebtorID	INT	FOREIGN KEY (Debtors)	ID of the associated debtor
TransactionID	INT	FOREIGN KEY (Transactions)	ID of the associated transaction
InvoiceDate	TIMESTAMP		Date and time of the invoice generation
Format	VARCHAR(50)		Format of the invoice (e.g., PDF, CSV)
InvoiceDetails	TEXT		Details included in the invoice

Table 6. 6 Invoice Table

CHAPTER :7 SYSTEM IMPLEMENTATION

7.1 HOME PAGE

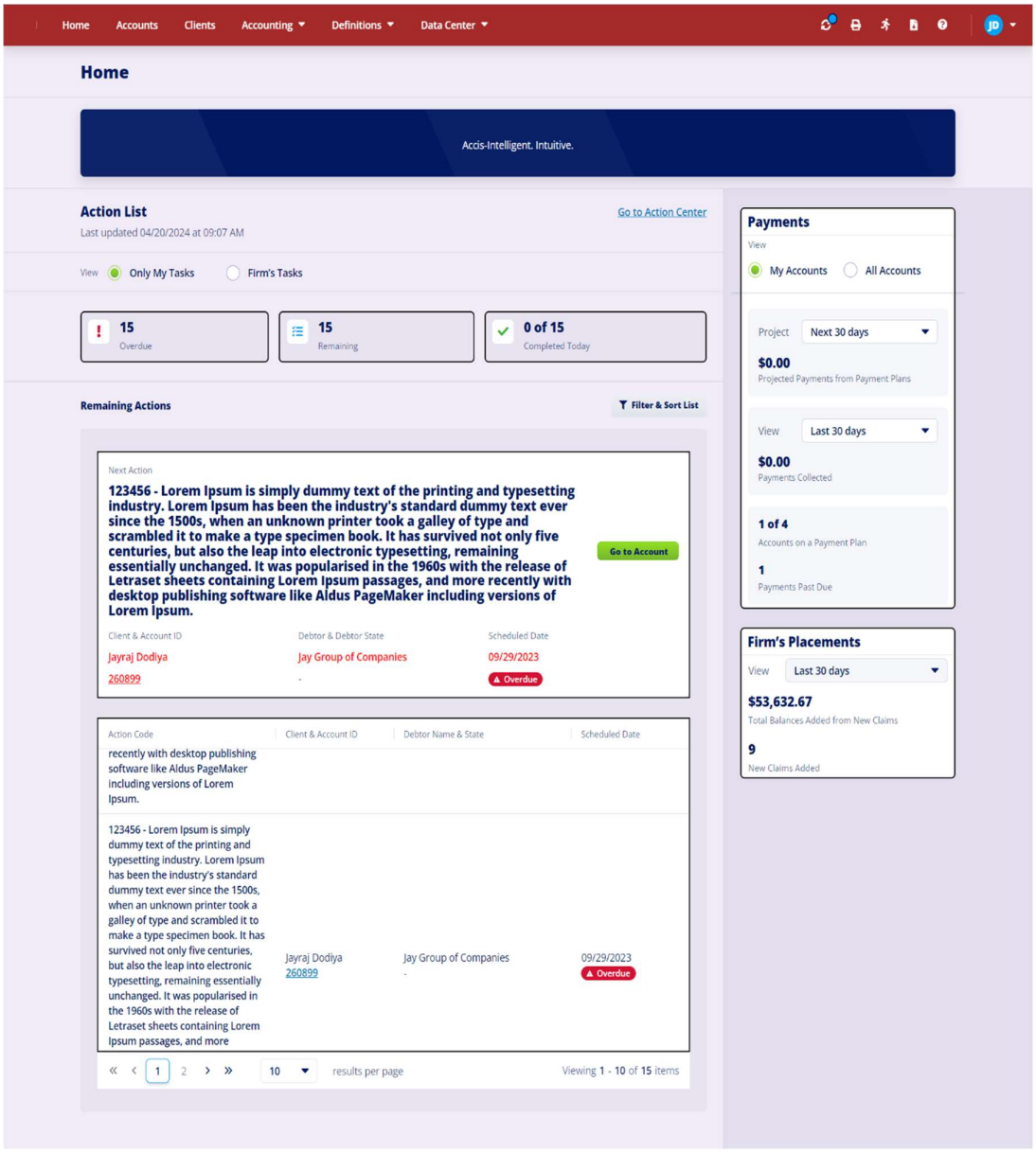

Figure 7. 1 HOME Page

7.2 CLIENT PAGE

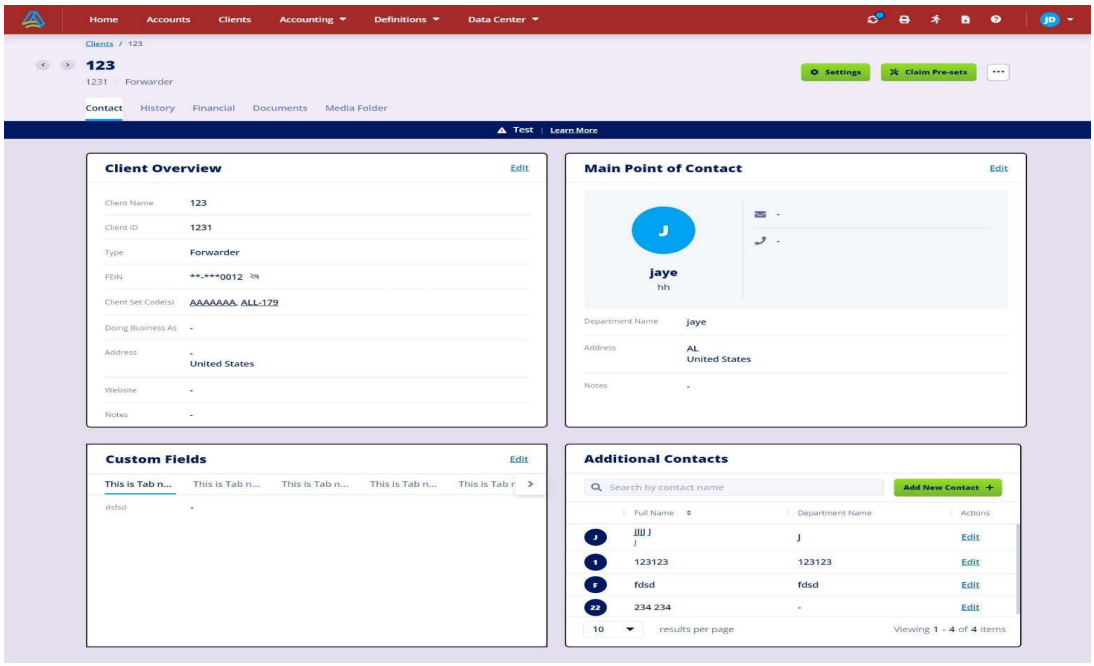

Figure 7. 2 Client Page

7.3 ROLE MANAGEMENT

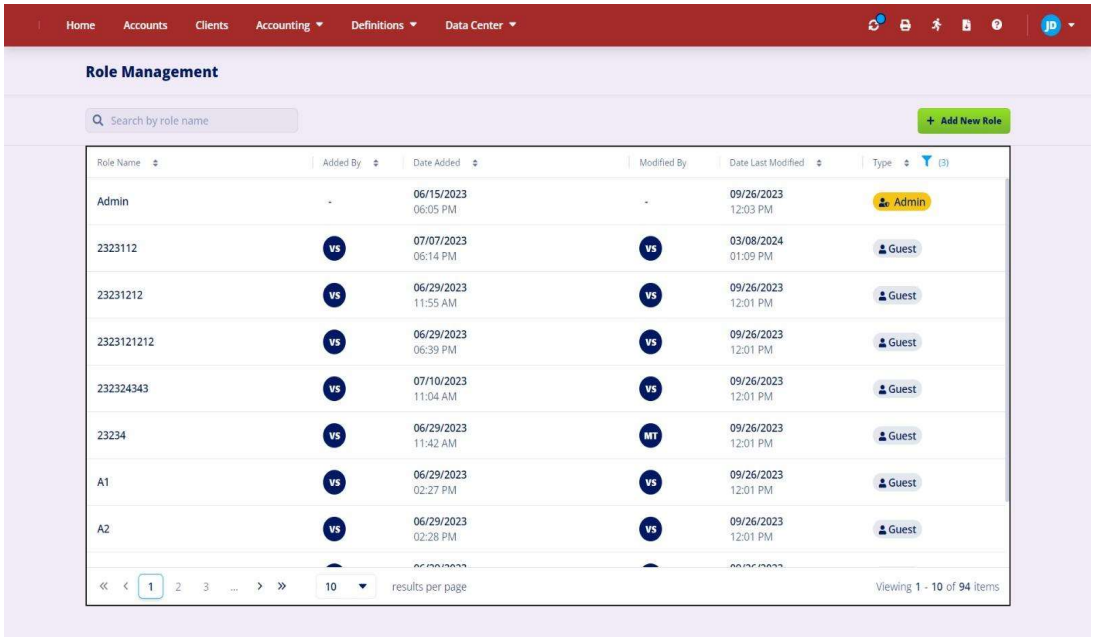

Figure 7. 3 Role Management

7.3 BANKING PAGE

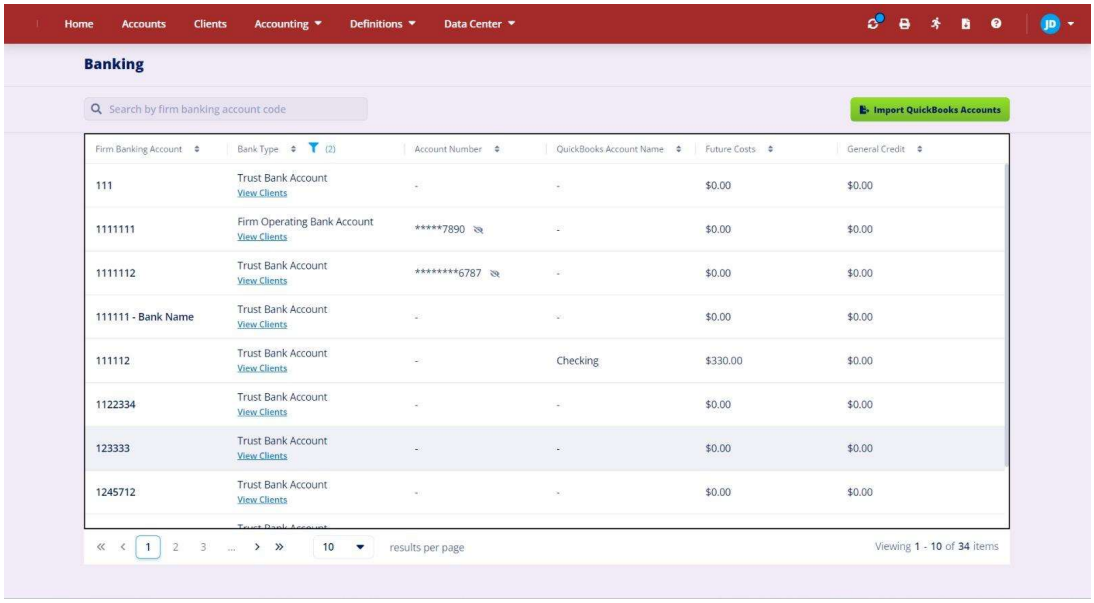

Figure 7. 4 Banking Page

7.4 INVOICE HISTORY

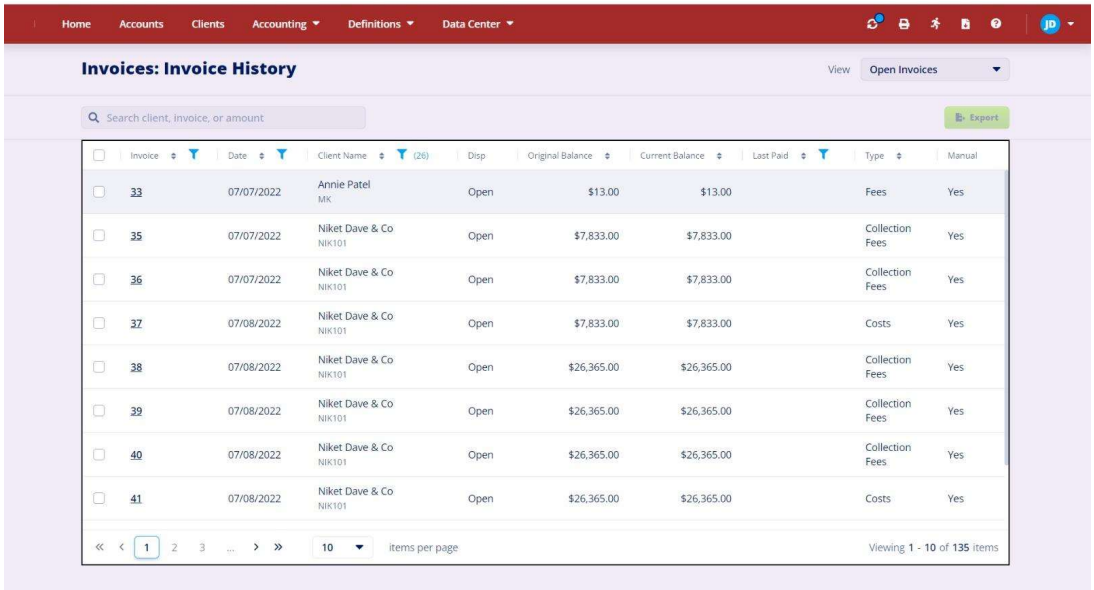

Figure 7. 5 Invoice History

7.5 REMITTANCE HISTORY

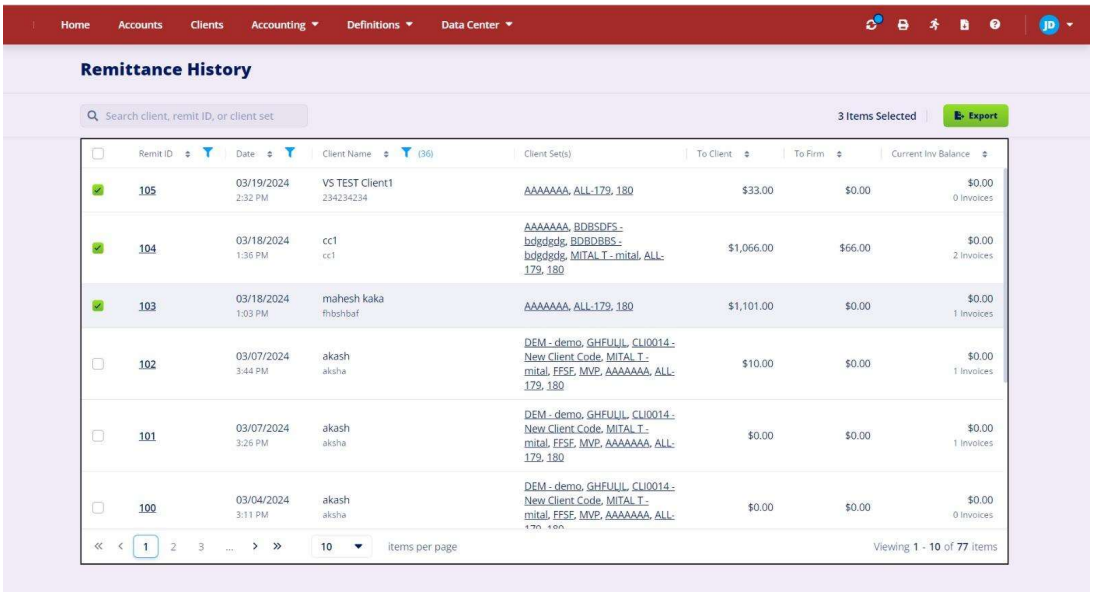

Figure 7. 6 Remittance History

7.6 QUICKBOOKS DEPOSITS

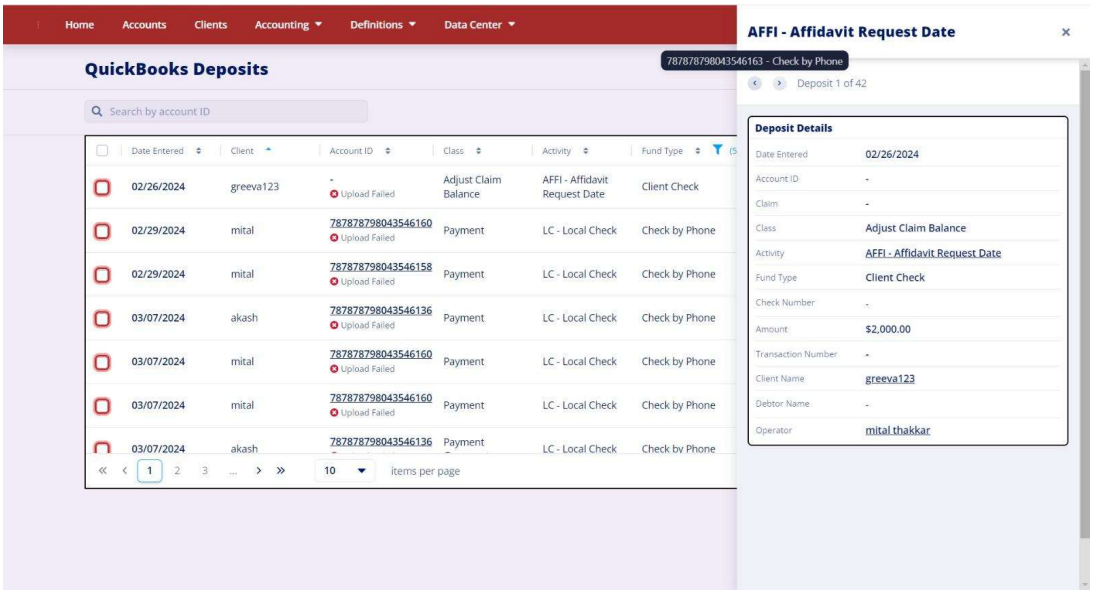

Figure 7. 7 Quickbooks Deposits

CHAPTER :8 SYSTEM TESTING

8.1 TESTING PLAN

8.1.1 Testing after completion of each module:

After completion of each module, thorough testing will be conducted to ensure that individual functionality are working as expected. This includes testing for individual behaviour, input validation, error handling and any specific requirements outlined for each module.

8.1.2 Integration Testing:

Once individual modules are tested and verified, integration testing will be performed to ensure that interconnected modules function correctly together. This involves testing data flow, communication pathways, and interactions between different modules to validate seamless integrations.

8.2 TEST CASE AND RESULT

Entity	Test Case	Expected Output	Actual Output	Result
Client Creation	User creates a new client	Client is successfully created	Client created successfully	Pass
Client Editing	User edits an existing client	Client details are successfully updated	Client details updated successfully	Pass
Client Deletion	User deletes a client	Client is successfully deleted	Client deleted successfully	Pass
Account Creation	User creates a new account for a client	Account is successfully created	Account created successfully	Pass
Account Editing	User edits an existing account	Account details are successfully updated	Account details updated successfully	Pass
Account Deletion	User deletes an account	Account is successfully deleted	Account deleted successfully	Pass
Debt Creation	User creates a new debt	Debt is successfully created	Debt created successfully	Pass
Debt Editing	User edits an existing debt	Debt details are successfully updated	Debt details updated successfully	Pass
Debt Deletion	User deletes a debt	Debt is successfully deleted	Debt deleted successfully	Pass
Debt Collection	User records a debt collection	Debt collection is successfully recorded	Debt collection recorded successfully	Pass
Remittance Processing	User processes a remittance	Remittance is successfully processed	Remittance processed successfully	Pass
Data Conversion	User converts data format	Data is successfully converted	Data converted successfully	Pass
Batch Processing	User processes data in batches	Data is successfully processed in batches	Data processed in batches successfully	Pass
Integration with QuickBooks	User integrates system with QuickBooks	Integration is successful	Integration with QuickBooks successful	Pass
Error Handling	User attempts invalid action	Error message is displayed	Error message displayed	Pass

Responsiveness	User accesses system on different devices	System interface adjusts smoothly for different devices	Interface adjusts smoothly for different devices	Pass
Payment Tracking	User records a payment for a debt	Payment is successfully recorded	Payment recorded successfully	Pass
Payment Adjustment	User adjusts the payment amount for a debt	Payment amount is successfully adjusted	Payment amount adjusted successfully	Pass
Payment Reversal	User reverses a payment made for a debt	Payment reversal is successfully processed	Payment reversed successfully	Pass
Debt Aging Report	User generates a report showing the aging of debts	Report showing aging of debts is generated	Debt aging report generated successfully	Pass
Payment Reminder	System sends a payment reminder to a debtor	Payment reminder is successfully sent	Payment reminder sent successfully	Pass
Debt Write-Off	User writes off a debt	Debt is successfully written off	Debt written off successfully	Pass
Debt Recovery	User initiates a debt recovery process	Debt recovery process is successfully initiated	Debt recovery initiated successfully	Pass
Audit Trail	User reviews the audit trail for a debt collection activity	Audit trail is displayed showing the activity history	Audit trail displayed successfully	Pass
System Backup	User performs a backup of the debt collection system data	Data backup is successfully completed	Data backup completed successfully	Pass
Data Restore	User restores the debt collection system data from a backup	Data is successfully restored from the backup	Data restored successfully	Pass

Table 8. 1 Test Case

CHAPTER :9 CONCLUSION

9.1 FUTURE ENHANCEMENT

9.1.1 Advanced Reporting and Analytics:

Enhance the Debt Collection Management System (DCMS) by incorporating advanced reporting and analytics capabilities. This could include the implementation of interactive dashboards, data visualization tools, and predictive analytics algorithms. These features would enable users to gain deeper insights into debt collection trends, identify patterns, and make data-driven decisions to optimize collection strategies.

9.1.2 Real-time Notifications using Firebase:

Integrate Firebase Cloud Messaging (FCM) to enable real-time notifications within the DCMS. By implementing live notifications, users can stay updated on critical events such as new debt assignments, payment status updates, account changes, and more. This feature will improve communication efficiency and enable timely actions, ultimately enhancing the overall effectiveness of debt collection efforts.

9.1.3 Online Payment Integration:

Enhance the Debt Collection Management System by integrating online payment functionality post creation of debt accounts. This feature would streamline the payment process for debtors, allowing them to make payments securely and conveniently through the system. Integration with popular payment gateways will ensure a seamless payment experience, reducing manual effort and improving debtor satisfaction.

9.2 SUMMARY OF INTERNSHIP

My internship journey as a Software Engineer Trainee at Codal Systems has been a remarkable opportunity for personal and professional development. I am immensely grateful for the chance to work alongside talented individuals and professionals who have guided me throughout this enriching experience. I extend my heartfelt appreciation to all those who have been part of this project and have supported me along the way.

Special acknowledgment goes to the following individuals:

- My internal guide, **Prof. Dhaval Varia**, whose insightful suggestions and encouragement have been invaluable in shaping my learning journey.

- **Mr. Arpit Patel**, my manager, for his mentorship and guidance, which have been instrumental in my growth as a software engineer.
- Throughout my internship, I explored ReactJS, Redux, React Router, and Axios. This experience improved my front-end development skills for creating dynamic web apps. It also taught me about teamwork, communication, and problem-solving in software development.

In summary, my internship at Codal Systems has been a transformative learning experience, equipping me with valuable skills and insights that I am eager to apply in future projects. I am deeply grateful for the opportunity and look forward to continuing my journey in the field of software engineering with enthusiasm and dedication.

9.3 CONCLUSION

The Debt Collection Management System (DCMS) project offers a comprehensive solution for efficient debt collection and management within organizations. Initiated by the debt collection department, the system facilitates seamless debtor tracking and account management, ensuring that debt collection efforts are well-coordinated and targeted. The DCMS provides features such as debtor information recording, payment history tracking, and follow-up action scheduling, enhancing debt collection efficiency. It also offers robust security measures to protect sensitive debtor information. Ultimately, the DCMS streamlines debt collection operations, fosters effective communication with debtors, and provides a scalable framework for future enhancements, contributing significantly to organizational success in debt recovery.