

Digital Check Sheet

Submitted by

KADEEJATH SALAHA

2217029

Master of Computer Applications

**St. Aloysius Institute of Management and Information Technology (AIMIT)
Mangalore**

Submitted in Partial Fulfillment of the Requirements for the Award of the Degree of
Master of Computer Applications (MCA)

Under the guidance of

Mrs. Annapoorna Shetty

Asst. Professor
Dept. of MCA, AIMIT,
St Aloysius (Deemed to be University),
Beeri, Mangalore-575022

Mr. Raghavendra G Shetty

Managing Director,
RDL Technology Pvt. LTD.
Sahyadri Campus, Adyar,
Mangalore 575007

Submitted to



**ST. ALOYSIUS INSTITUTE OF MANAGEMENT AND INFORMATION
TECHNOLOGY (AIMIT)
ST ALOYSIUS COLLEGE (AUTONOMOUS)
MANGALURU, KARNATAKA**

2024



**ST ALOYSIUS INSTITUTE OF MANAGEMENT AND INFORMATION TECHNOLOGY
ST ALOYSIUS COLLEGE (AUTONOMOUS)
MANGALORE, KARNATAKA**

CERTIFICATE

This is to certify that the project titled

Digital Check Sheet

Submitted by

KADEEJATH SALAHA

2217029

Master of Computer Applications

**St. Aloysius Institute of Management and Information Technology (AIMIT)
Mangalore**

In partial fulfillment of the requirements for award of degree of Master of Computer Applications of Mangalore University, is a bonafide record of the work carried out at

RDL Technologies Pvt. LTD.

Sahyadri Campus, Adyar, Mangaluru

During the year

2024

Mrs. Annapoorna Shetty

Asst. Professor

Dept. of MCA, AIMIT,

St Aloysius (Deemed to be University),

Beerli, Mangaluru-575 022.

Dr. Hemalatha N

Dean, AIMIT,

PG Depts of Information Technology,

St. Aloysius College,

Beerli, Mangaluru-575 022.

Dr (Fr) Kiran Cotha S J

Director, AIMIT

St. Aloysius (Deemed to be University),

Beerli, Mangaluru-575 022.

Examiners

1.

2.



ST ALOYSIUS INSTITUTE OF MANAGEMENT AND INFORMATION TECHNOLOGY
ST. ALOYSIUS COLLEGE (AUTONOMOUS)
BEERI, MANGALORE – 575 022
P G DEPT OF INFORMATION TECHNOLOGY & BIOINFORMATICS

PROFORMA OF PROJECT PROPOSAL

Note : The student has to fill this form and approval has to be sought from the Dean of the faculty before starting the Industry Internship.(January I week)

- [1]. Student Register Number : 2217029
- [2]. Student Name : Kadeejath Salaha
- [3]. Title of the Project : Digital CheckSheet
- [4]. Name & Address of the Organization : RDL Technologies Pvt.Ltd. ,5th floor ,Sahyadri Campus, Adyar, Mangaluru ,575007, Karnataka.
- [5]. Telephone No. : +91 8088423347
- [6]. E-mail : sales@rdltech.in
- [7]. Website / URL : <https://rdltech.in/>
- [8]. Address of the Project Site : 5th floor ,Sahyadri Campus, Adyar, Mangaluru ,575007, Karnataka.
- [9]. Telephone No. : +91 8088423347
- [10]. E-mail : sales@rdltech.in
- [11]. Residential Address of the Student : Darul Maimana Moososdi, Uppala PO,
- [12]. Kasaragod, Kerala- 671322
- [13]. Telephone No : -
- [14]. Mobile Number : 919645176669
- [15]. E-mail : kadeejasalaha123@gmail.com

I hereby declare that the above furnished details are true to best of my knowledge and request you to permit me to join the Industry Internship for the Project work prescribed for MCA.

Signature of the Student

The Proposal for the project work has been accepted

Dean of the Faculty

CERTIFICATE OF AUTHENTICATED WORK

This is to certify that the project report entitled Digital Check Sheet submitted to Aloysius Institute of Management and Information Technology (AIMIT), St Aloysius College, Mangalore affiliated to Mangalore University in partial fulfilment of the requirement for the award of the degree of MASTER OF COMPUTER APPLICATIONS (MCA) is an original work carried out by **Ms. Kadeejath Salaha** Register no. **2217029** under my guidance. The matter embodied in this project is authentic and is genuine work done by the student and has not been submitted to this University, or to any other University / Institute for the fulfilment of the requirement of any course of study.

Signature of the Student:

Date:.....

Name and Address
of the student

.....

.....

.....

Register No.....

Signature of the Guide

Date:

Name, Designation
and Address of the Guide

.....

.....

.....

ABSTRACT

The Digital Check Sheet (DCS) application is a comprehensive digital platform designed to streamline the management and monitoring of routine tasks within organizations. By replacing traditional manual processes with an automated, centralized system, the DCS application enhances efficiency, accuracy, and accountability. The application features a user-friendly interface built with React.js and a robust backend powered by Node.js, ensuring seamless data handling and real-time updates. Key functionalities of the DCS application include automated scheduling, real-time notifications, and offline capability, which allow users to perform and record tasks even in areas with limited connectivity. The application supports role-based access control, providing different levels of functionality for Super Admins, Admins, and Users, thereby maintaining security and data integrity.

The DCS application significantly reduces the administrative burden by enabling efficient task management, real-time data entry and validation, and comprehensive monitoring of organizational activities. This digital transformation not only improves operational efficiency but also ensures compliance with organizational standards and enhances overall productivity.



RDL Technologies Pvt. Ltd

Date:2024-07-27

Ref-RDL/081/2024-25

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Ms. Kadeejath Salaha with Reg No: 2217029 a student of St. Aloysius College, Mangalore (AIMIT) is currently interning at our Company on a project “Digital Check Sheet”, showcasing dedication and professionalism in her work. The project is expected to be completed by 25/09/2024.

Thanking You

RDL Technologies Pvt Ltd. Authorized Signature

Authorized Signature





RDL Technologies Pvt. Ltd

Date:2024-04-10

Ref-RDL/081/2024-25

To,

Dean Academics (IT)

Aloysius Institute of Management and Information Technology (AIMIT)

Beerli, Kotekar,

Mangalore- 575022

We hereby accept Ms. Kadeejath Salaha (Reg No: 2217029) to carry out our project work at our organization, for the period of 6 months from 25th March 2024.

Thanking You

RDL Technologies Pvt Ltd.

Authorized Signatur



Project Proposal Report /Synopsis

for

Digital Check Sheet

KADEEJATH SALAHA

2217029

Master of Computer Applications

**St. Aloysius Institute of Management and Information Technology (AIMIT)
Mangalore**

Under the guidance of

Mrs. Annapoorna Shetty

Asst. Professor

Dept. of MCA, AIMIT,
St Aloysius (Deemed to be University),
Beeri, Mangalore-575022

Submitted to



**ST ALOYSIUS INSTITUTE OF MANAGEMENT AND INFORMATION
TECHNOLOGY (AIMIT)
ST ALOYSIUS COLLEGE (AUTONOMOUS)
MANGALORE, KARNATAKA**

Project Proposal / Synopsis

I. Title of the Project

Digital Check Sheet

II. Statement of the Problem

In today's digital age, the need for streamlined processes within organizations is ever-growing. Effective management of organizational tasks and activities is crucial for maintaining efficiency, productivity, and compliance. Traditional methods of handling these tasks often involve manual processes, paper-based records, and fragmented systems, leading to several critical issues:

- **Inefficiency and Time Consumption:**

Manual entry of check sheets is labor-intensive and time-consuming. Employees spend a significant amount of time filling out, verifying, and consolidating data from various sources, which could be better spent on more strategic activities.

- **Human Errors and Inconsistencies:**

The manual process is prone to human errors, such as data entry mistakes, omissions, and inconsistencies. These errors can lead to inaccurate records, affecting decision-making and overall operational efficiency.

- **Lack of Real-time Monitoring:**

Traditional check sheets do not provide real-time monitoring and updates, making it difficult for management to track progress and ensure timely completion of tasks. This lack of visibility can lead to missed deadlines and unmet compliance requirements.

- **Security Concerns:**

Paper-based records and unencrypted digital records are vulnerable to unauthorized access, loss, and damage. Ensuring the security and integrity of sensitive organizational data is a significant concern.

- **Inflexibility and Scalability Issues:**

Traditional methods lack flexibility and scalability. As organizations grow and their needs evolve, the existing systems may not adapt well to changing requirements, leading to inefficiencies and additional costs. To address these challenges, the Digital Check Sheet (DCS) project aims to develop a comprehensive solution that digitizes the management of check sheets, enhances real-

time data access and monitoring, automates scheduling and notifications, ensures data security, and supports offline functionality. By solving these problems, the DCS application seeks to improve overall operational efficiency, accuracy, and productivity within organizations.

III. Why this particular topic chosen?

The Digital Check Sheet (DCS) project was chosen due to its profound relevance in addressing the inefficiencies and challenges faced by organizations in managing routine tasks and activities. Traditional methods of handling check sheets often involve manual processes that are time-consuming, prone to errors, and lack real-time data access. These limitations hinder effective decision-making and timely issue resolution. By digitizing check sheets, the DCS application aims to enhance operational efficiency, accuracy, and productivity. It offers automated scheduling, real-time data entry and validation, and robust security measures, ensuring data integrity and confidentiality. Additionally, the application's offline functionality caters to areas with limited connectivity, ensuring uninterrupted task management. The structured roles and access controls within the DCS application further streamline task delegation and accountability. Choosing this topic aligns with the growing need for digital transformation in organizations, providing a scalable and customizable solution to improve overall management and control of organizational activities.

IV. Objective and Scope

Objective:

The primary objective of our project is to create a user-friendly platform that allows organizations to generate digital checklists and datasheets seamlessly. By automating this process, including the frequency checkup aspect, we aim to improve efficiency, reduce errors, and enhance overall productivity within organizations. The objective encompasses several key goals:

- **Enhance Efficiency and Productivity:**

Replace manual, paper-based processes with a streamlined digital system to reduce time consumption and increase productivity across the organization.

- **Ensure Data Accuracy and Consistency:**

Minimize human errors and inconsistencies by providing a reliable platform for real-time data entry and validation, ensuring accurate and consistent records.

- **Support Scalability and Customization:**

Design the application to be scalable and customizable to meet the diverse needs of different organizations, making it a versatile tool for various industries.

- **Define Clear Roles and Access Controls:**

Establish distinct roles (Super Admin, Admin, User) with specific access controls and functionalities to ensure efficient task delegation, accountability, and management within the organization.

By achieving these objectives, the DCS project aims to transform the way organizations manage check sheets, leading to improved operational efficiency, enhanced data accuracy, and better overall management of organizational activities.

Scope:

The scope of the Digital Check Sheet (DCS) project encompasses the design, development, implementation, and maintenance of a digital solution for managing and monitoring check sheets within organizations. The project aims to address the needs of various stakeholders, including Super Admins, Admins, and Users, providing them with specific functionalities to enhance task management and operational efficiency. The following aspects define the scope of the DCS project:

- **User Roles and Permissions:**

- Super Admin: Responsible for managing organization details, overseeing login activities of Admins, and ensuring overall system security and integrity.
- Admin: Capable of creating check sheets for daily, weekly, and monthly tasks, managing user accounts, and monitoring task completion.
- User: Responsible for selecting and completing check sheets as per their assigned tasks, entering data, and validating task completion.

- **Digital Check Sheet Management:**

- Development of a user-friendly interface for creating, editing, and managing check sheets.
- Implementation of automated scheduling for routine checks with customizable intervals (daily, weekly, monthly).
- Real-time data entry and validation to ensure accuracy and consistency.

- **Notifications and Reminders:**

- Automated notifications and reminders for upcoming tasks and overdue check sheets to ensure timely completion.

- **Real-time Monitoring and Reporting:**

- Real-time monitoring of task progress and completion status.
- Generation of detailed reports for analysis and decision-making.

V. Methodology

The methodology for the Digital Check Sheet (DCS) project encompasses a systematic approach to planning, designing, developing, testing, and deploying the application. The project will follow the Agile development methodology, which promotes iterative progress, collaboration, and flexibility.

VI. Process Description

The Digital Check Sheet (DCS) platform offers a streamlined approach to managing organizational tasks by leveraging digital check sheets. The process begins with the Super Admin, who is responsible for overseeing the system's security and managing organizational details. They create and assign Admin login credentials, ensuring that the system remains secure and organized. The Admin, in turn, creates and manages check sheets for various tasks—daily, weekly, and monthly—and assigns these tasks to Users based on their roles. Admins also track the progress of these tasks, validate the data entered by Users, and provide feedback as necessary. Users log into the platform to view and complete their assigned check sheets, submitting their work for review upon completion. The system supports automated scheduling and notifications to remind Users of upcoming tasks, ensuring timely completion. Additionally, the platform includes offline functionality, allowing Users to complete tasks without an internet connection, with data synchronization occurring once connectivity is restored. This comprehensive approach ensures efficient task management, accurate data handling, and robust security across the organization.

VII. Resources and Limitations

The successful development and deployment of the Digital Check Sheet (DCS) application will rely on a range of resources while facing certain limitations. Technological resources include essential development tools such as Integrated Development Environments (IDEs) like Visual Studio Code, and version control systems like Git, which are crucial for coding and managing versions. Frameworks and libraries such as React for the front end, Node.js for the back end, and MUI for UI components will be employed to build the application efficiently. For data management, MySQL will be used to handle storage and retrieval. Human resources comprise a dedicated project team including software developers, UI/UX designers, testers, and project managers. Regular input from stakeholders like Super Admins, Admins, and Users will ensure that the application aligns with user needs. Comprehensive documentation and training resources

will include user manuals, guides, and training sessions to support effective use and adoption of the platform. However, the project will encounter several limitations. Technological constraints may arise, such as compatibility issues with older browsers or operating systems and scalability challenges as the number of users and data volumes grow. Budgetary constraints could limit the scope of features, affect third-party tool choices, and impact the available time for development and testing. Additionally, user adoption might be hindered by resistance to change from traditional methods, necessitating extra training and support. Finally, offline functionality may present challenges with data synchronization, particularly in maintaining data integrity and consistency when transitioning between offline and online modes. Addressing these resources and limitations effectively will be key to the successful implementation and operation of the DCS application.

VIII. Testing Technologies

The testing of the Digital Check Sheet (DCS) application will involve a comprehensive approach to ensure the platform's reliability, functionality, and performance. Each component and module of the application will undergo rigorous testing using various technologies and methodologies.

Modules in this project will be tested using

- Unit Testing: Testing each module of software project is known as unit testing. To perform this kind of testing, knowledge of programming is necessary.
- Manual Testing: Manual testing includes testing a software manually, i.e., without using any automated tool or any script.
- Automation Testing: Automation testing makes use of specialized tools to control the execution of tests and compares the actual results against the expected result.

IX. Conclusion

The Digital Check Sheet system represents a significant advancement in the realm of organizational management, offering a comprehensive solution for the streamlined management of checklists and datasheets. By automating the generation of customized checklists, incorporating frequency checkup parameters, and providing intuitive user interfaces, the system empowers organizations to enhance efficiency, accuracy, and productivity.

ACKNOWLEDGEMENT

The satisfaction that accompanies the ongoing progress of any task would be impossible without mentioning the people who make it possible. Their constant guidance and encouragement are key to our efforts. This project has been the most practical and exciting part of my learning experience during the final semester, which will be an asset for me in my future career.

I sincerely appreciate all those people who have been instrumental in our progress so far. I am very grateful to **RDL Technologies Pvt Ltd** for providing us with this project, the necessary facilities, and a conducive environment.

I would also like to thank my External Guide, **Mr. Raghavendra G Shetty** of RDL Technologies Private Ltd, for her guidance and useful suggestions that have been invaluable in our ongoing work.

I would like to thank **Dr. (Fr.) Kiran Cotha S J**, Director of Aloysius Institute of Management and Technology (AIMIT), for their constant support and encouragement.

I am also highly obliged to **Dr. Hemalatha N**, Dean Academics (IT) at Aloysius Institute of Management and IT (AIMIT), for her help and instructions throughout our project work.

I would like to thank my Internal Guide, **Mrs. Annapoorna Shetty**, Asst. Professor of the Dept. of MCA, who has been very supportive throughout the project and who has provided me with all valuable inputs to put my efforts on right track.

Finally, I place a deep sense of gratitude to my family members and friends who have been a constant source of inspiration during the preparation and ongoing work of this project.

| TABLE OF CONTENTS | Page No |
|---|----------------|
| CHAPTER 1 INTRODUCTION | 16 |
| 1.1 Purpose | 17 |
| 1.2 Document Conventions | 17 |
| 1.3 Intended Audience and reading suggestions | 18 |
| 1.4 Project Scope | 18 |
| 1.5 Organization of Report | 18 |
| CHAPTER 2: SURVEY OF TECHNOLOGIES | 21 |
| 2.1 React.js | 22 |
| 2.2 Node.js | 22 |
| 2.3 HTML & CSS..... | 22 |
| 2.4 MySQL..... | 22 |
| 2.5 MUI..... | 23 |
| 2.6 Visual Studio..... | 23 |
| 2.7 API | 23 |
| CHAPTER 3: REQUIREMENTS AND ANALYSIS | 25 |
| 3.1 Problem Definition | 26 |
| 3.2 Requirement Specification | 26 |
| 3.2.1 Performance Requirements... .. | 26 |
| 3.2.2 Safety Requirements... .. | 27 |
| 3.2.3 Stable Internet Connection | 27 |
| 3.2.4 Security Requirements | 27 |
| 3.2.5 Software Quality Attributes... .. | 27 |
| 3.2.5.1 Reliability | 27 |
| 3.2.5.2 Availability | 28 |
| 3.2.5.3 Testability | 28 |
| 3.2.5.4 Maintainability..... | 28 |
| 3.3 Software and Hardware Requirements | 28 |
| 3.4 Preliminary Product Description | 29 |
| 3.5 Conceptual Model | 29 |

| | |
|--|-----------|
| 3.5.1 Data Flow Diagram | 29 |
| 3.5.2 Entity Relationship Diagram | 33 |
| CHAPTER 4: SYSTEM DESIGN..... | 34 |
| 4.1 Basic Models | 35 |
| 4.1.1 Super Admin Module | 35 |
| 4.1.2 Admin Module | 35 |
| 4.2.3 User Module... .. | 35 |
| 4.2 Data Design Project Structure | 35 |
| 4.2.1 Schema Design... .. | 35 |
| 4.2.2 Data Integrity and Constraints | 38 |
| 4.3 Procedural Design | 38 |
| 4.3.1 Logic Diagrams | 38 |
| 4.3.1.1 Use Case Diagram... .. | 39 |
| 4.3.1.2 Sequence Diagram... .. | 40 |
| 4.3.1.3 State Chart Diagram... .. | 42 |
| 4.3.1.4 Activity Diagram | 43 |
| 4.4 User Interface Diagram | 46 |
| 4.5 Security Issues | 55 |
| 4.6 Test Case Design | 55 |
| CHAPTER 5: IMPLEMENTATION AND TESTING | 57 |
| 5.1 Implementation Approaches | 58 |
| 5.2 Coding Details and Code Efficiency | 58 |
| 5.3 Testing Approach | 59 |
| 5.3.1 Unit Testing | 60 |
| 5.3.2 Integration Testing... .. | 60 |
| 5.3.3 User Acceptance Testing | 60 |
| 5.3.4 Performance Testing | 61 |
| 5.3.5 Load Testing/ Stress Testing | 61 |
| 5.3.6 Modification and Improvement | 61 |
| CHAPTER 6: RESULTS AND DISCUSSION | 62 |
| 6.1 Test Reports | 63 |
| 6.2 Identification..... | 65 |

| | |
|---------------------------------------|--------|
| 6.3 Type..... | 65 |
| 6.4 Function..... | 66 |
| 6.5 Interface | 66 |
| 6.6 Processing | 66 |
| 6.7 Data | 67 |
| 6.8 Dependency Description | 67 |
| 6.9 Interface Description | 67 |
| CHAPTER 7: CONCLUSION..... | 68 |
| 7.1 Conclusion | 69 |
| 7.2 Limitations | 69 |
| 7.3 Future Scope of the project | 70 |
| REFERENCES | 71 |
| GLOSSARY | 72 |
| FEEDBACK FORMS | 73 |

| Table of Figures | Page No |
|---|----------------|
| Figure 1: Level 0 DFD Diagram..... | 30 |
| Figure 2: Level 1 DFD for SuperAdmin..... | 31 |
| Figure 3: Level 1 DFD for Admin | 31 |
| Figure 4: Level 1 DFD for User | 32 |
| Figure 5: Level 2 DFD for Admin | 32 |
| Figure 6:ER Diagram | 33 |
| Figure 7: Use Case Diagram..... | 40 |
| Figure 8: Sequence diagram... .. | 41 |
| Figure 9: State chart..... | 43 |
| Figure 10: Activity Diagram..... | 45 |
| Figure 11: Super Admin Login Page..... | 46 |
| Figure 12: Admin Login Page | 46 |
| Figure 13: Organization Dashboard... .. | 46 |
| Figure 14: Add Organization | 47 |
| Figure 15: Admin Logs | 47 |
| Figure 16: Admin Dashboard | 48 |
| Figure 17: Department view | 48 |
| Figure 18: Section view..... | 49 |
| Figure 19: Add section | 49 |
| Figure 20: Title view..... | 50 |
| Figure 21: Confirm box while deleting | 50 |
| Figure 22: View Heading..... | 51 |
| Figure 23: Add Heading..... | 51 |
| Figure 24: Create Template..... | 52 |
| Figure 25: View checklist..... | 52 |
| Figure 26: Checksheet..... | 53 |
| Figure 27: Checksheet..... | 53 |
| Figure 28: Checksheet | 54 |
| Figure 29: User Signup..... | 54 |
| Figure 30: User Login | 55 |
| Figure 31: Forgot Password..... | 55 |

List of Tables

PageNo

| | |
|--|----|
| Table 1: Database Table for login | 35 |
| Table 2: Database Table for Admin login | 35 |
| Table 3: Database Table for Organizations..... | 36 |
| Table 4: Database Table for Department..... | 36 |
| Table 5: Database Table for Heading..... | 36 |
| Table 6: Database Table for Templates | 36 |
| Table 7: Database Table for Titles..... | 37 |
| Table 8: Database Table for Sections | 37 |
| Table 9: Database Table for User login | 37 |

Software Requirements Specification

for

Digital Check Sheet

KADEEJATH SALAHA

2217029

Master of Computer Applications

**St. Aloysius Institute of Management and Information Technology (AIMIT)
Mangalore**

Under the guidance of

Mrs. Annapoorna Shetty

Asst. Professor

Dept. of MCA, AIMIT,

St Aloysius (Deemed to be University),
Beerli, Mangalore-575022

Submitted to



**ST ALOYSIUS INSTITUTE OF MANAGEMENT AND INFORMATION
TECHNOLOGY (AIMIT)
ST ALOYSIUS COLLEGE (AUTONOMOUS)
MANGALORE, KARNATAKA**

CHAPTER 1

INTRODUCTION

CHAPTER 1 INTRODUCTION

1.1 Purpose

The Digital Check Sheet (DCS) application is designed to address the complexities and challenges associated with managing organizational tasks and processes. In today's fast-paced business environment, organizations face numerous hurdles, including maintaining accurate records, ensuring timely completion of routine checks, and managing data effectively. Traditional methods of handling these tasks—often reliant on paper-based systems or disparate digital tools—can be inefficient and prone to errors.

The primary purpose of the DCS application is to streamline and automate the management of check sheets and associated tasks within organizations. By digitalizing check sheet management, the application aims to enhance operational efficiency, improve data accuracy, and provide real-time visibility into task progress. It addresses the challenges of tracking task completion, managing schedules, and ensuring compliance with organizational standards.

With the DCS application, organizations can overcome issues such as manual data entry errors, difficulties in monitoring task status, and inefficiencies associated with traditional check sheet processes. The platform provides a unified solution for creating, assigning, and tracking check sheets, as well as automating notifications and reminders to ensure tasks are completed on time.

In essence, the DCS application seeks to modernize task management by offering a comprehensive digital solution that simplifies workflows, improves data management, and enhances overall productivity within organizations.

1.2 Document Conventions

- The document is justified with Line Spacing of 1.5

Convention for Title

- Font face: Times New Roman
- Font style: Bold
- Font Size: 14

Convention for Subtitle

- Font face: Times New Roman
- Font style: Bold
- Font Size: 12

Convention for Body

- Font face: Times New Roman
- Font Size: 12

1.3 Intended Audience and Reading Suggestions

This document is intended to help the developer to understand what must be achieved throughout the development process. It also helps the developer to analyze whether the application being developed meets the requirement or not. The document should be viewed through the scope and system features to get an overview of the entire project.

1.4 Project Scope

The scope of the Digital Check Sheet (DCS) project encompasses the development and deployment of a comprehensive digital solution for managing organizational check sheets and associated tasks. This project aims to streamline task management by providing a unified platform that allows for the creation, assignment, tracking, and completion of check sheets. The DCS application will facilitate automated scheduling of routine checks, send timely notifications and reminders, and support both online and offline functionality to ensure uninterrupted task management. The project includes designing a user-friendly interface for different roles, such as Super Admins, Admins, and Users, each with specific functionalities and permissions. Super Admins will have the capability to manage organizational details and oversee Admin logins, while Admins will create and manage check sheets, assign tasks, and monitor user progress. Users will interact with the platform to complete assigned tasks and submit check sheets for review. The project will involve a structured development process, including requirements gathering, design, development, testing, and deployment. The testing phase will incorporate various methodologies to ensure the application's performance, security, and usability. By focusing on these areas, the DCS project aims to enhance organizational efficiency, improve task management processes, and provide a reliable solution for monitoring and managing check sheets.

1.5 Organization of report

➤ Chapter 1: Introduction

This chapter introduces the system that is being developed. The project's history and setting are described in detail. The purpose, scope, and applicability of the project are described together with a concise summary of its goals and objectives.

➤ Chapter 2: Survey of technologies

All the technologies needed to execute the project are described in depth in this chapter. This shows that the project's team is aware of and knowledgeable about the technologies that are relevant to the project's theme.

➤ **Chapter 3: Requirements and Analysis**

This chapter discusses and evaluates the requirements that influenced the development of the system. This covers the definition of the problem, the specification of the requirement, and the planning and scheduling. In the problem definition phase, specifics about the main problem and its subproblems are given. The planning and scheduling phase specifies planning for the objectives as well as specific rules and limits. In the document, the hardware and software requirements are listed. The system's hardware requirements for running the product are spelled out in detail. This section contains a list of the software requirements, including the operating system and other software needed to link and install the software. The section on product description outlines the specifications and goals of the new system, as well as its features and methods of operation. The problem domain, which outlines the activities that may be carried out and the permitted sequences of those operations, is understood through the use of conceptual models.

➤ **Chapter 4: System Design**

System Design includes data design, functional and interface design, process diagrams, detailed function descriptions, and test case design in its clear description of features and operations. All of the modules and their features are explained briefly in the Basic Modules section. The Data Design section follows, and it describes how to manage, organize, and modify the data. Schema design and Data Integrity and Constraints are the second and third subsections of this section, respectively. The structure and justification of schemas are defined by schema design. All of the validity checks and constraints are listed in data integrity and constraints.

➤ **Chapter 5: Implementation and testing**

This chapter covers the project's implementation strategy, standards for implementation, code efficiency, testing methodology, integrated testing, and modifications and additions.

➤ **Chapter 6: Results and Discussions**

Here, the test reports provide an explanation of the test findings and case-based reports, demonstrating that the software is capable of handling any challenging circumstances and performing well under various settings. Additionally, the user manual explains how the software functions. With screenshots, it explains the many components and operations of the device.

➤ **Chapter 7: Conclusions**

The entire work is summed up in this chapter. It clarifies every argument put out in the prior chapters. It also lists all of the criticism heard during the software demonstration, along with the system's shortcomings. Additionally, it suggests the project's future scope, including any additional research topics required by changes. Further, it describes the system's limitations and potential future improvements.

CHAPTER 2

SURVEY OF TECHNOLOGIES

CHAPTER 2: SURVEY OF TECHNOLOGIES

2.1 React.js

React.js is a popular open-source JavaScript library developed by Facebook for building user interfaces, especially single-page applications where data changes dynamically. React allows developers to create reusable UI components, which enhances the efficiency of development and ensures consistency across the application. It utilizes a virtual DOM to optimize rendering performance, providing a fast and responsive user experience. ReactJS component-based architecture makes it easy to manage and scale complex user interfaces, facilitating the development of modern, interactive web applications.

2.2 Node.js

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to execute JavaScript code on the server side. Built on the V8 JavaScript engine, Node.js enables the creation of scalable network applications with a non-blocking, event-driven architecture. This makes it particularly well-suited for building real-time applications and APIs. Node.js is known for its high performance and ability to handle concurrent requests efficiently. It provides a rich set of built-in libraries and a robust package management system via npm (Node Package Manager), which simplifies dependency management and code reuse.

2.3 HTML & CSS

CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in HTML or XML. It controls the layout, colors, fonts, and overall visual appearance of web pages. CSS enhances the user experience by providing a way to apply consistent styles across multiple pages or components, enabling responsive design for various devices and screen sizes. Modern CSS features, such as Flexbox and Grid, facilitate complex layouts and improve the flexibility and maintainability of styles.

2.4 MySQL

MySQL is a freely available open-source Relational Database Management System (RDBMS) that uses Structured Query Language (SQL). SQL is the most popular language for adding, accessing, and managing content in a database. It is most noted for its quick processing, proven

reliability, ease and flexibility of use.

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is released under an open-source license. So, you have nothing to pay to use it. MySQL is a very powerful program. It handles a large subset of the functionality of the most expensive and powerful database packages. MySQL uses a standard form of the well-known SQL data language. MySQL works on many operating systems and with many languages including Python, JAVA, etc. MySQL works very quickly and works well even with large data sets. MySQL is very friendly to PHP, the most appreciated language for web development. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

2.5 MUI (Material-UI)

MUI (Material-UI) is a popular React component library that implements Google's Material Design principles. It provides a comprehensive suite of pre-built, customizable components, allowing developers to build aesthetically pleasing and responsive user interfaces quickly. MUI's component-based architecture promotes reusability and consistency across applications. The library includes a wide range of components, from basic elements like buttons and text fields to complex components like data grids and dialogs. MUI also supports theming and styling, enabling developers to tailor the look and feel of their applications to match specific design requirements. Its integration with React makes it an excellent choice for modern web development.

2.6 Visual Studio

Visual Studio is an integrated development environment (IDE) developed by Microsoft for building and debugging applications. It supports multiple programming languages, including C#, JavaScript, and SQL, and provides a comprehensive set of tools for coding, testing, and deployment. Visual Studio features advanced debugging, code analysis, and project management capabilities, which enhance developer productivity. It is commonly used for developing web applications, services, and database management systems, making it a versatile tool in the software development lifecycle.

2.7 API

An API, or Application Programming Interface, is a crucial tool in software development that

allows different applications to communicate and interact with each other. It defines a set of rules and protocols that dictate how requests and responses should be formatted and processed. APIs can be used in various contexts, such as web APIs for online services, library APIs for software components, operating system APIs for accessing system functions, and database APIs for data management. Web APIs, including REST and SOAP, enable seamless data exchange over the internet using HTTP/HTTPS protocols. REST APIs are known for their simplicity and use of standard HTTP methods, while SOAP APIs offer robust security features. GraphQL is another modern approach that allows clients to request specific data precisely. APIs play a vital role in integration, enabling different systems and services to work together, automating tasks, and facilitating data retrieval and manipulation.

CHAPTER 3

REQUIREMENTS AND

ANALYSIS

CHAPTER 3: REQUIREMENTS AND ANALYSIS

3.1 Problem Definition

The challenge addressed by the Digital Check Sheet (DCS) application stems from the inefficiencies and inaccuracies associated with traditional methods of managing organizational tasks and check sheets. Many organizations rely on paper-based systems or fragmented digital tools to track and manage routine tasks, leading to several issues:

- **Manual Errors:** Paper-based check sheets are prone to human error during data entry and tracking, resulting in inaccuracies that can impact task completion and overall operational efficiency.
- **Inefficient Tracking:** Traditional systems often lack real-time tracking capabilities, making it difficult to monitor the progress of tasks, identify bottlenecks, and ensure timely completion.
- **Lack of Automation:** Manual scheduling and reminders for routine checks can be time-consuming and prone to oversight, leading to missed tasks and delays in critical processes.
- **Limited Accessibility:** Paper-based or isolated digital systems may not support offline access, hindering the ability to complete and update check sheets in areas with limited connectivity.
- **Data Security Concerns:** Traditional methods may lack robust security measures, risking the exposure of sensitive information and increasing vulnerability to data breaches.
- **Integration Challenges:** Existing systems may not integrate seamlessly with other organizational tools, resulting in fragmented data management and inefficient workflows.

The Digital Check Sheet application seeks to address these problems by providing a comprehensive digital solution that enhances accuracy, efficiency, and security in managing check sheets and related tasks. By automating task scheduling, enabling real-time tracking, supporting offline functionality, and incorporating strong security measures, the DCS application aims to improve overall task management and operational effectiveness within organizations.

3.2 Requirement Specification

3.2.1 Performance Requirements

The performance of the Digital Check Sheet (DCS) application is influenced by both server and client-side factors. As a web-based solution, its performance is closely tied to network speed and server capacity. The application relies on a range of APIs for functionalities such as data retrieval,

task updates, and notifications. Therefore, network latency and bandwidth can impact the response times and overall user experience. The system is designed to optimize performance by employing efficient data handling techniques and minimizing latency through scalable cloud infrastructure.

3.2.2 Requirements for Safety

To ensure the safety of user data, the DCS application leverages a secure cloud platform that provides robust data protection mechanisms. The application utilizes multiple layers of security, including authentication tokens and encryption, to safeguard sensitive information. Authorization techniques ensure that only authenticated and authorized users have access to specific functionalities and data, thus maintaining the confidentiality and integrity of the data.

3.2.3 Stable Internet Connection

A stable internet connection is crucial for accessing and using the DCS application effectively. While the application is designed to handle intermittent connectivity by supporting offline functionality, optimal performance and real-time updates require a reliable internet connection. Users should ensure that their network environment provides adequate stability and speed to fully utilize the application's features.

3.2.4 Security Requirements

Data Privacy: All data within the DCS application must remain private and accessible only to authenticated users. User access is controlled through secure login credentials, including usernames and passwords.

Backend Access: The application's backend servers are restricted to access by authenticated administrators only. This ensures that sensitive system operations are protected from unauthorized access.

Role-Based Access Control: The system implements role-based access control (RBAC) to manage permissions. Each user is assigned a specific role with defined access constraints, ensuring that users only access data and functionalities pertinent to their responsibilities.

3.2.5 Software Quality Attributes

3.2.5.1 Reliability

The DCS application ensures high reliability by utilizing cloud infrastructure with built-in data

redundancy and backup capabilities. Regular backups and system monitoring are in place to maintain data integrity and system stability.

3.2.5.2 Availability

The application is designed to be highly available, with a target uptime of 99.9%. This is achieved through redundant systems, load balancing, and failover mechanisms to minimize downtime and ensure continuous access for users.

3.2.5.3 Testability

The application's modules are individually tested to ensure functionality and performance. Comprehensive test coverage is maintained through unit tests, integration tests, and system tests, which are automated where possible to facilitate continuous integration and deployment.

3.2.5.4 Maintainability

The DCS application is built with maintainability in mind, allowing for straightforward updates and enhancements. The codebase is designed to be modular and extensible, making it easy to integrate new features and adapt to emerging technologies. Regular maintenance practices are established to ensure cost-effectiveness and simplicity in keeping the application up-to-date.

3.3 Software and Hardware Requirements

| Services | Software and Versions required |
|-----------------------|--|
| Framework & Libraries | React.js, Node.js, Express.js, CSS, MUI |
| Database Server | MySQL server Ver 8.0.35 is recommended. |
| Browser | Chrome, Mozilla Firefox, or any other browsing application |
| IDE | Visual Studio |
| Language Used | React.js, Node.js, MySQL |

Hardware Requirements

- RAM: 4 GB (minimum), 8 GB (recommended)
- Storage: 5GB (minimum to accommodate database growth, logs, and static files).
- Operating System: Android, IOS, Linux, Windows 10+.

3.4 Preliminary Product Description

The primary goal of the Digital Check Sheet (DCS) application is to streamline the management and monitoring of organizational tasks through a digital platform. Traditionally, managing check sheets and routine tasks often involves manual processes that can be inefficient, error-prone, and difficult to track. This project aims to replace these cumbersome manual systems with a centralized digital solution that offers enhanced functionality and efficiency.

The DCS application provides a unified platform for creating, assigning, and tracking check sheets, which are essential for routine and administrative tasks within an organization. By digitizing these processes, the application ensures that tasks are scheduled automatically, reminders and notifications are sent in real-time, and progress is tracked accurately.



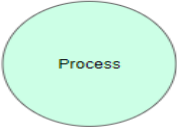
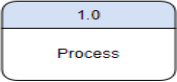



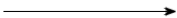
3.5 Conceptual Models

3.5.1 Data Flow Diagrams

Data flow diagram (DFD) is a graphical representation of the “flow” of data through an information system; it differs from the flowchart as it shows the data flow instead of the control flow of the program.

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

DFD is used to convey how data or information flows through the system, how data is transformed in the process from lower level to higher level where in project requirement and the process could be easily understood.

| | Yourdon and Coad | Gane & Sarson |
|-----------------|---|---|
| External Entity |  |  |
| Process |  |  |
| Data Store |  |  |
| Data Flow |  |  |

0-level DFD:

It is also known as a Context Flow Diagram. It's designed to be an abstract view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.

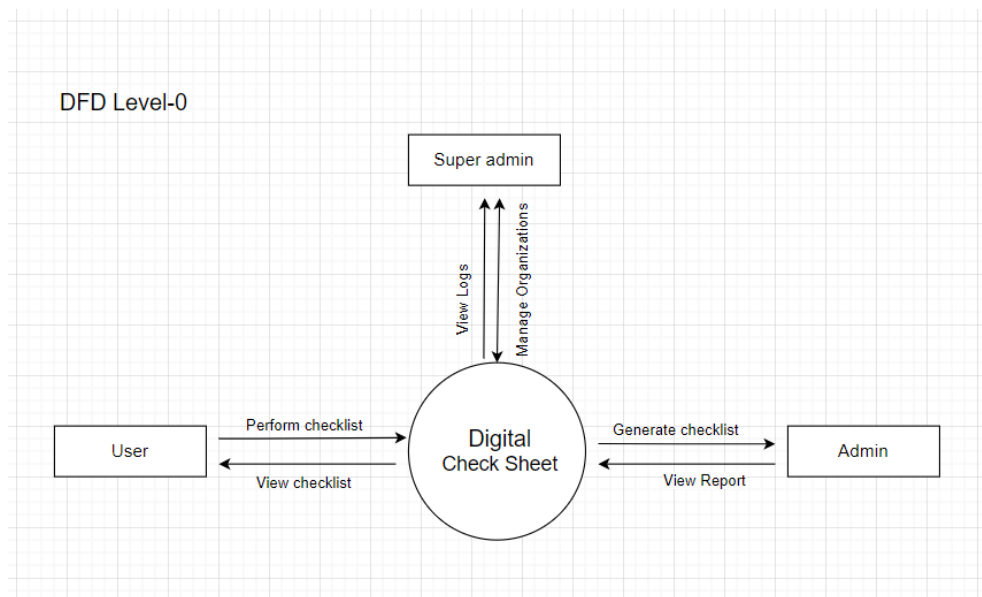


Figure 1: Level 0 DFD Diagram

1-level DFD:

In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and break down the high-level process of 0-level DFD into sub-processes.

Level-1 DFD Super Admin

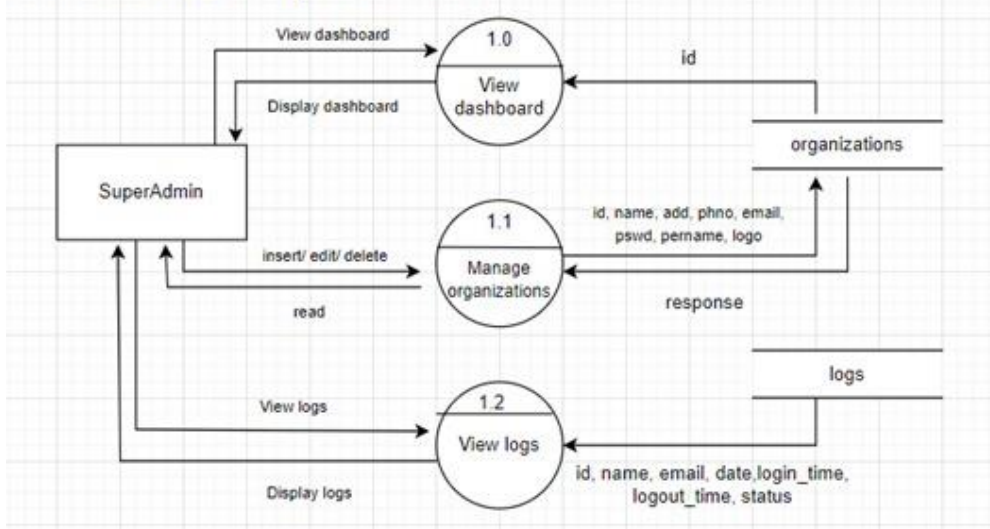


Figure 2: Level 1 Super Admin DFD Diagram

Level-1 DFD Admin

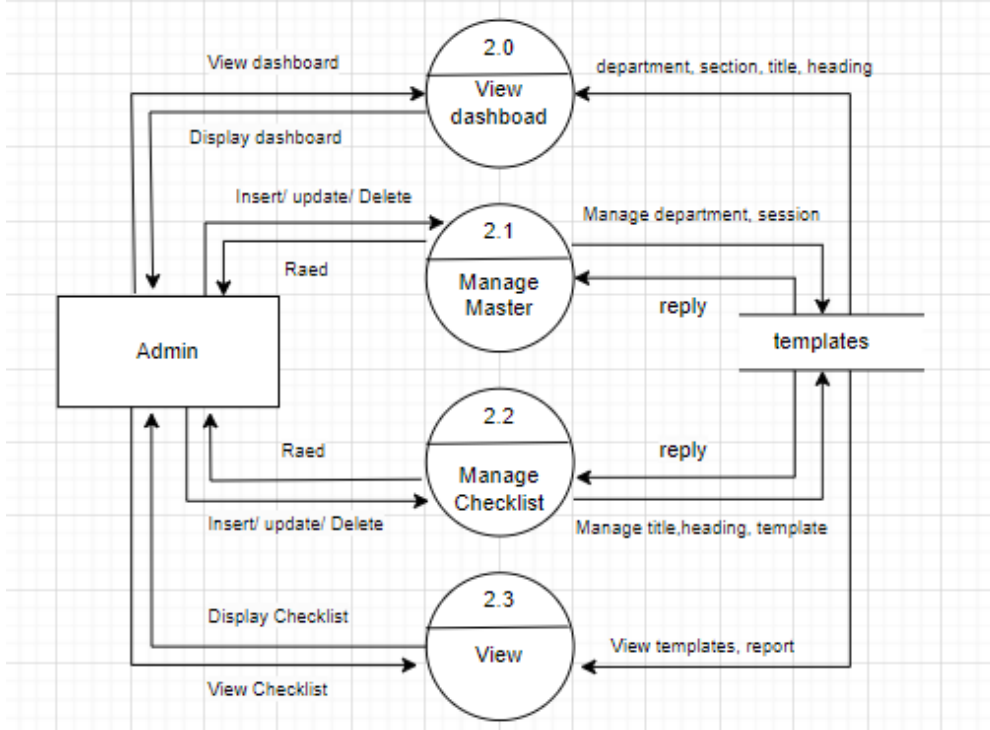


Figure 3: Level 1 Admin DFD Diagram

User DFD level-1

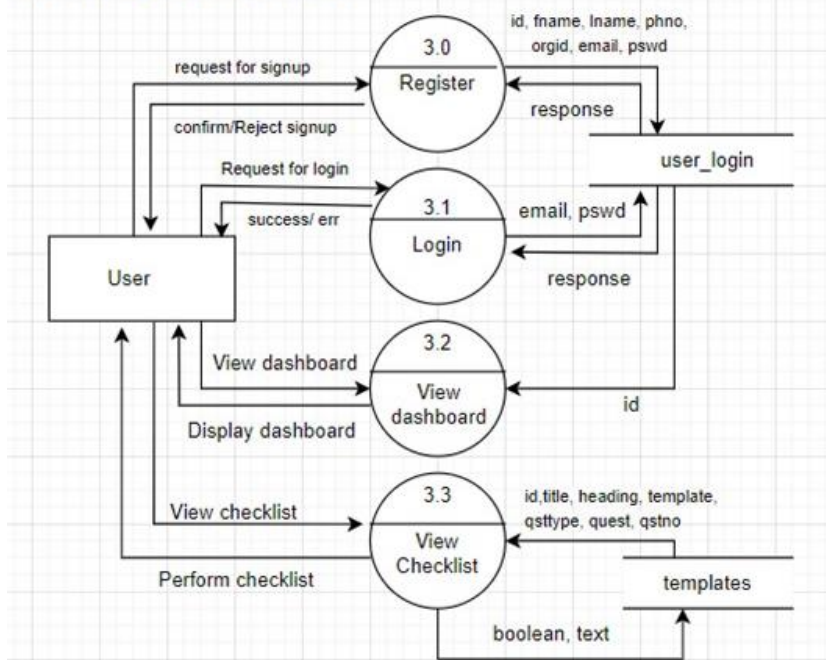


Figure 4: Level 1 User DFD Diagram

2-level DFD:

Level-2 DFD Admin

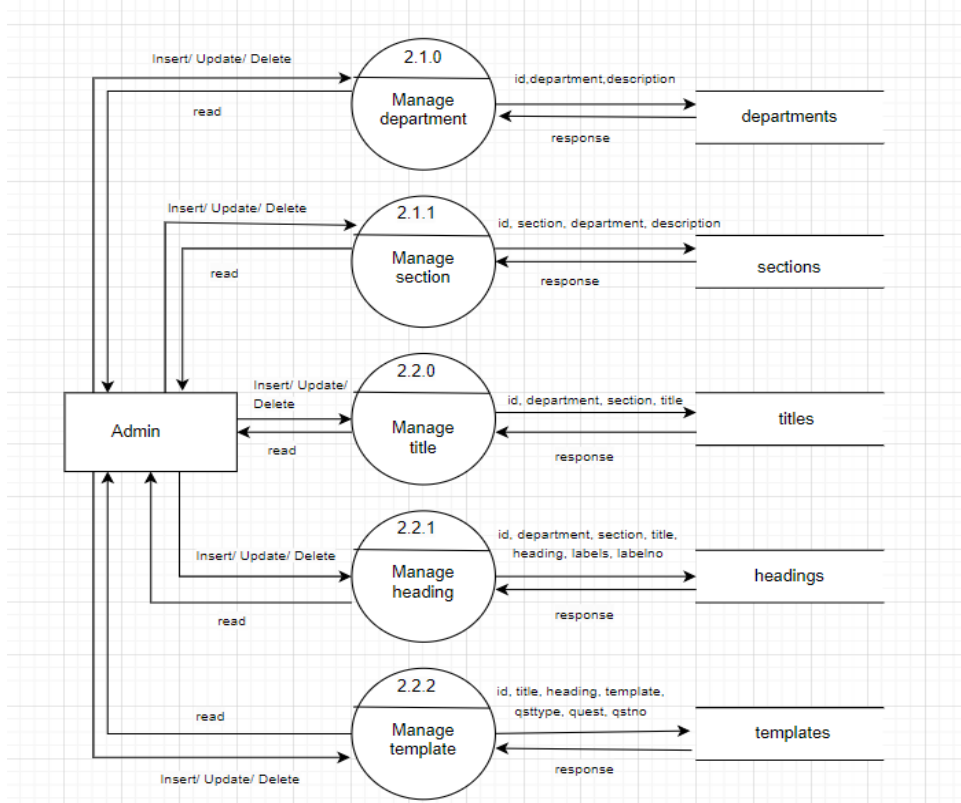


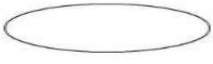





Figure 5: Level 2 Admin DFD Diagram

3.5.2 Entity Relationship Diagram

Entity-Relationship Diagram is a type of database model based on the notion of real-world entities and relationship among them. It is a specialized graphic that illustrates the inter-relationships among business entities.

| BASIC NOTATIONS | | |
|---|------------------------|--|
| SYMBOL | NOTATION | DESCRIPTION |
|  | Entity | An entity is an object or concept about which you want to store information. |
|  | Weak Entity | Weak entity is an entity that must be defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone. |
|  | Attribute | Represents the characteristic properties belonging to the entity. |
|  | Key Attribute | A key attribute is the unique, distinguishing characteristic of the entity. |
|  | Multi-Valued Attribute | A multi-valued attribute can have more than one value. |
|  | Derived Attribute | An attribute whose value is calculated (derived) from other attributes. The derived attribute may or may not be physically stored in the database |

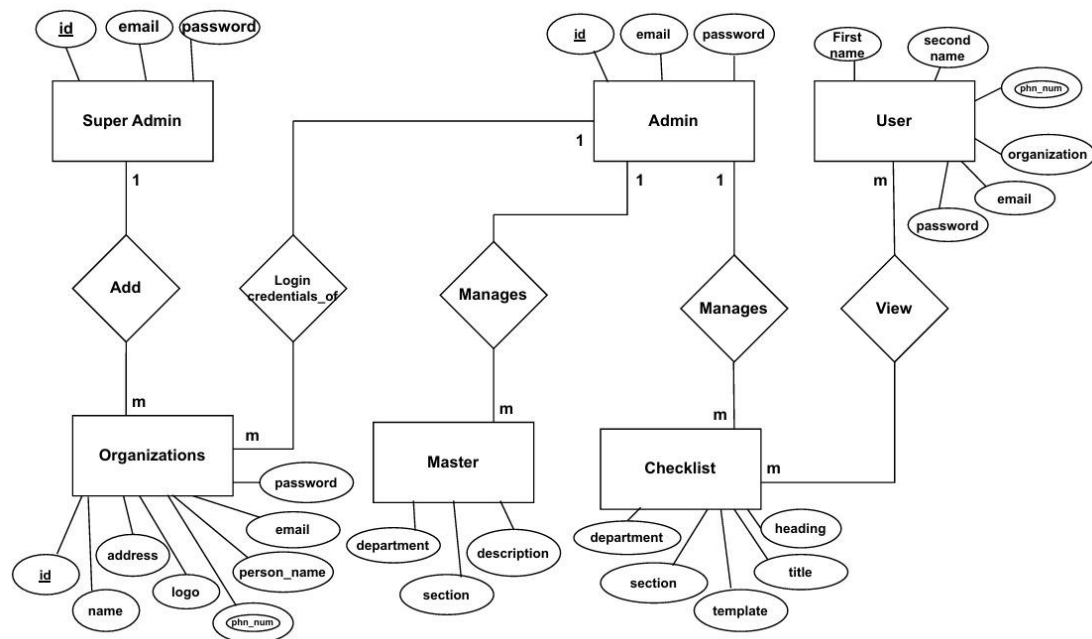


Figure 6: ER Diagram

System Design Document

for

Digital Check Sheet

KADEEJATH SALAHA

2217029

Master of Computer Applications

**St. Aloysius Institute of Management and Information Technology (AIMIT)
Mangalore**

Under the guidance of

Mrs. Annapoorna Shetty

Asst. Professor

Dept. of MCA, AIMIT,

St Aloysius (Deemed to be University),
Beerli, Mangalore-575022

Submitted to



**ST ALOYSIUS INSTITUTE OF MANAGEMENT AND INFORMATION
TECHNOLOGY (AIMIT)
ST ALOYSIUS COLLEGE (AUTONOMOUS)
MANGALORE, KARNATAKA**

CHAPTER 4 SYSTEM DESIGN

CHAPTER 4: SYSTEM DESIGN

4.1 Basic Modules

4.1.1 Super Admin Module: Can add new organizations by entering details such as organization name, phone number, email, password, and logo. Provides the admin's login credentials (email and password) for each organization. Upon Admin logins, the Super Admin can view logs that include the person's name, email, and login time. Manages overall system settings and oversees the administrative functions.

4.1.2 Admin Module: The Admin manages the organization-specific settings and monitors user activities. Uses the credentials provided by the Super Admin to log in. Manages the organization-specific settings.

- Master Page: Adds departments and sections within the organization.
- Checklist Page: Creates checklists by adding titles and headings.
- Checklist Type: Specifies the type of checklist (daily, weekly, monthly) and whether it is a task completion form or a feedback form.
- Reports: Views submitted checklists and generates reports based on user submissions.

4.1.3 User Module: The User interacts with the system by completing tasks and checklists. Registers under the desired organization to gain access to its checklists. Views and fills out the checklists assigned to the organization. Submits completed checklists which are then reviewed by the admin. Provides necessary feedback or completes tasks as required by the checklist.

4.2 Data Design Project Structure

4.2.1 Schema design

Table 1: Database Table for Login

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| EMAIL | NVARCHAR(255) | NOT NULL |
| PASSWORD | NVARCHAR(255) | NOT NULL |

Table 2: Database Table for Admin login

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-------------|-----------|-------------|
| ID | INT | PRIMARY KEY |

| | | |
|----------|---------------|----------|
| EMAIL | NVARCHAR(255) | NOT NULL |
| PASSWORD | NVARCHAR(255) | NOT NULL |

Table 3: Database Table for organizations

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|--------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| NAME | NVARCHAR(255) | NOT NULL |
| ADDRESS | NVARCHAR(255) | NOT NULL |
| PHONE NUMBER | NVARCHAR(50) | NOT NULL |
| EMAIL | NVARCHAR(255) | NOT NULL |
| PASSWORD | NVARCHAR(255) | NOT NULL |
| PERSON NAME | NVARCHAR(255) | NOT NULL |
| LOGO | NVARCHAR(255) | NOT NULL |

Table 4: Database Table for department

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| NAME | NVARCHAR(255) | NOT NULL |
| DESCRIPTION | TEXT | NULL |

Table 5: Database Table for Headings

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|--------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| DEPARTMENT | NVARCHAR(255) | NOT NULL |
| SECTION | NVARCHAR(255) | NOTNULL |
| TITLE | NVARCHAR(255) | NOT NULL |
| HEADING | NVARCHAR(255) | NOT NULL |
| LABEL NUMBER | INT | NULL |
| LABELS | TEXT | NULL |

Table 6: Database Table for Templates

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-------------|-----------|-------------|
|-------------|-----------|-------------|

| | | |
|-----------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| TITLE | NVARCHAR(255) | NOT NULL |
| HEADING | NVARCHAR(255) | NOTNULL |
| TEMPLATE | TEXT | NOT NULL |
| QUESTION TYPE | NVARCHAR(255) | NOT NULL |
| QUESTION NUMBER | INT | NULL |
| QUESTIONS | JSON | NOT NULL |

Table 7: Database Table for Titles

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| DEPARTMENT | NVARCHAR(255) | NULL |
| SECTION | NVARCHAR(255) | NULL |
| TITLE | NVARCHAR(255) | NULL |

Table 8: Database Table for Sections

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| DEPARTMENT | NVARCHAR(255) | NOT NULL |
| SECTION | NVARCHAR(255) | NOT NULL |
| DESCRIPTION | TEXT | NULL |

Table 9: Database Table for User login

| COLUMN NAME | DATA TYPE | CONSTRAINTS |
|-----------------|---------------|-------------|
| ID | INT | PRIMARY KEY |
| FIRST NAME | NVARCHAR(255) | NOT NULL |
| LAST NAME | NVARCHAR(255) | NOT NULL |
| PHONE | VARCHAR(20) | NOT NULL |
| ORGANIZATION ID | INT | MUL |
| EMAIL | NVARCHAR(255) | NOT NULL |
| PASSWORD | NVARCHAR(255) | NOT NULL |

4.2.2 Data Integrity and constraints

Primary key: The primary key of a table uniquely identifies each record in the table, so that an individual record can be located without confusion.

Foreign Key: It is a key used to link two tables together. Primary key is the target which a foreign key can reference. Foreign key identifies a column or a set of columns in one table that refers to set of columns in another table.

NOT NULL constraint: It allows you to specify that a column may not contain NULL values.

NULL constraint: It allows you to specify that a column may contain NULL values.

4.1 Object-Oriented Approach

UML can be described as the successor of object-oriented (OO) analysis and design. An object contains both data and methods that control the data. The data represents the state of the object. A class describes an object, and they also form a hierarchy to model the real-world system. The hierarchy is represented as inheritance and the classes can also be associated in different ways as per the requirement. Objects are the real-world entities that exist around us and the basic concepts such as abstraction, encapsulation, inheritance, and polymorphism all can be represented using UML. UML is powerful enough to represent all the concepts that exist in object-oriented analysis and design. UML diagrams are representation of object-oriented concepts only. Thus, before learning UML, it becomes important to understand OO concept in detail. OO can be defined as an investigation and to be more specific, it is the investigation of objects. Design means collaboration of identified objects. Thus, it is important to understand the OO analysis and design concepts. The most important purpose of OO analysis is to identify objects of a system to be designed. This analysis is also done for an existing system. Now an efficient analysis is only possible when we can start thinking in a way where objects can be identified. After identifying the objects, their relationships are identified and finally the design is produced. The purpose of OO analysis and design can describe as –

- Identifying the objects of a system.
- Identifying their relationships.
- Making a design, which can be converted to executables using OO languages

4.3 Procedural Design

4.3.1. Logic Diagrams

4.3.1.1 Use Case Diagrams

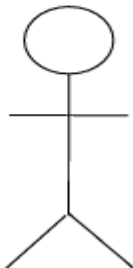
In the use case diagram, the representation of user's interaction with the system and specifications of a use case is done. It portrays the different types of users of a system and the various ways that they interact with the system.

Use case diagrams depict:

- **System:** Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.
- **Use Case:** Draw use cases using ovals. Label the ovals with verbs that represent the system's functions. A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.



- **Actors:** Actors are the users of a system. An actor is a person, organization, or external system that plays a role in one or more interactions with your system. Actors are drawn as stick figures. When system is the actor of another system, label the actor system with the actor stereotype.



Actor

- **Scenario:** A scenario is one hypothetical instance of how a particular use case might play out. A single use case thus inspires many different scenarios, in the same way that planning a driving trip from one city to another can involve many different routes.
- **Associations:** Associations between actors and use cases are indicated in use case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use case.
- **System boundary boxes (optional):** You can draw a rectangle around the use cases, called the system boundary box, to indicate the scope of your system.

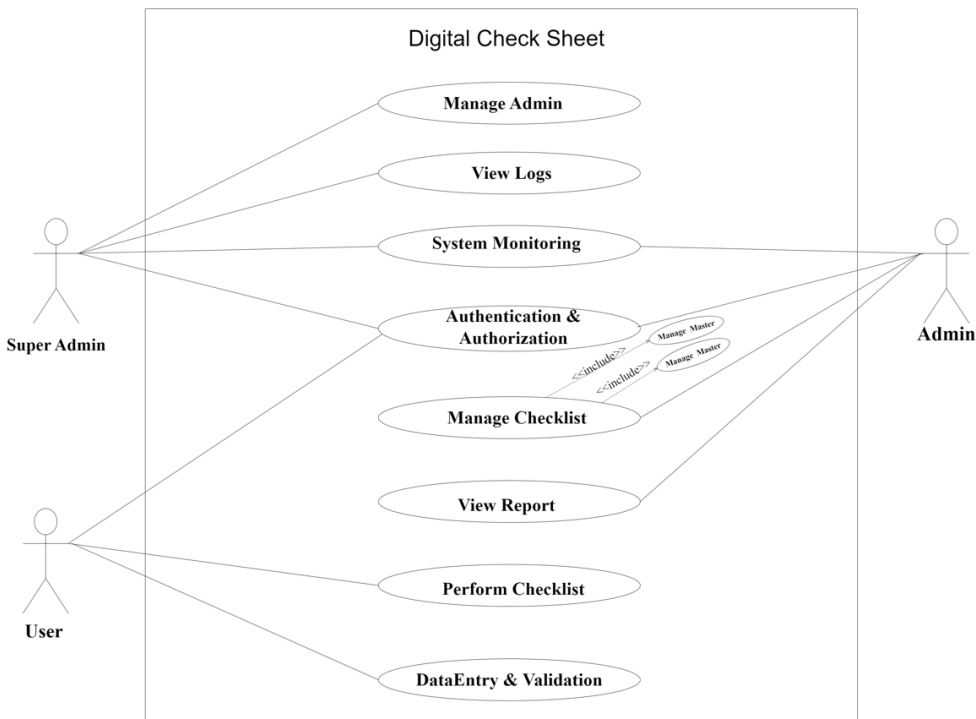


Figure 7: Use Case Diagram

4.3.1.2 Sequence Diagrams

A Sequence Diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. UML sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. 50 User or another system that will interact with the objects. Actors are placed in columns. The objects in the system that is being modelled and represented in rectangular boxes. Dashed line is lifeline which indicates the existence of the object over time. Rectangular box is Activation which indicates that the object is performing an action. To display the interaction, messages are used. These are horizontal arrows with the message name written above them.

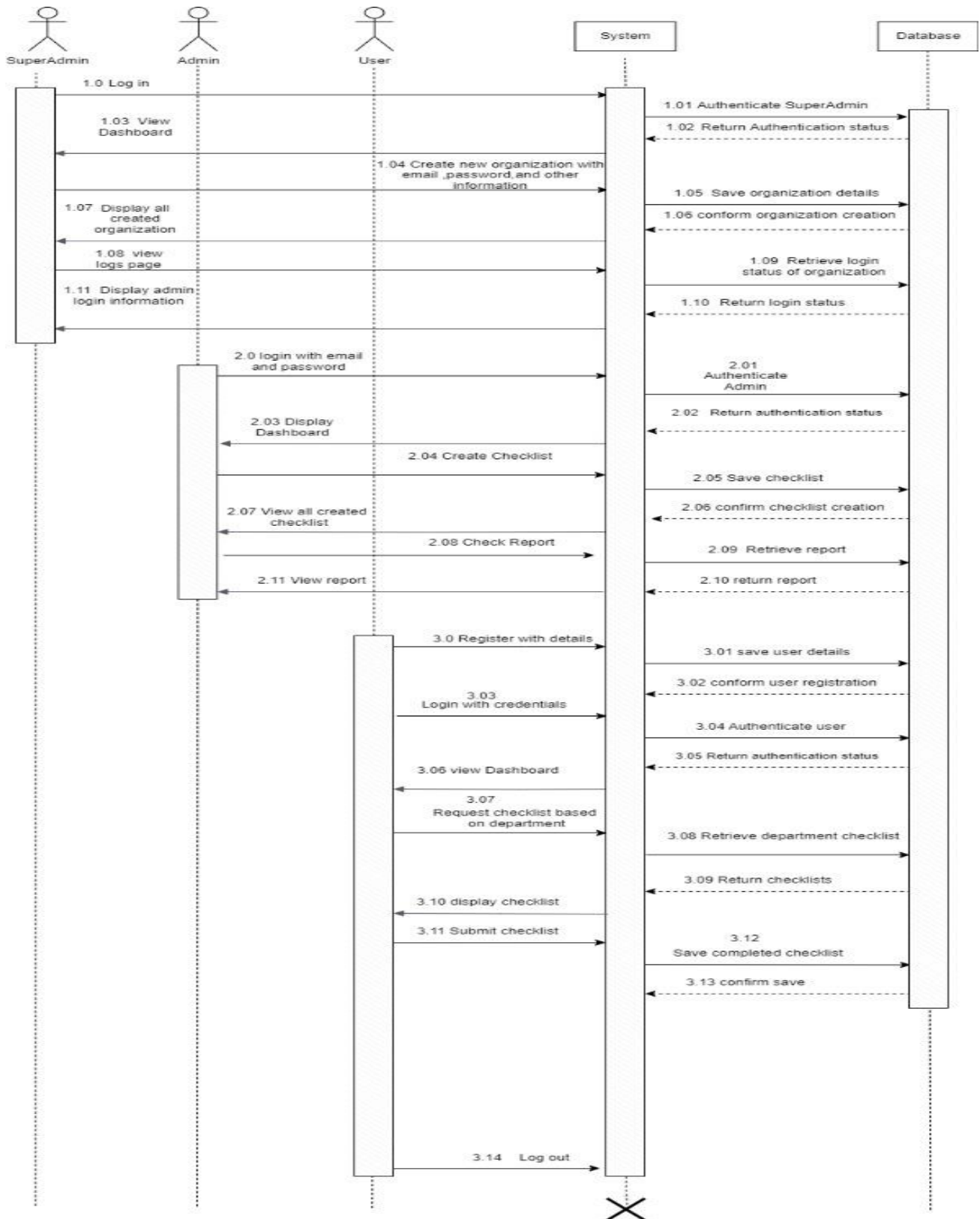


Figure 8: Sequence Diagram

4.3.1.3 State chart Diagrams

A state diagram shows the behaviour of classes in response to external stimuli. Specifically, a state diagram describes the behaviour of a single object in response to a series of events in a system.

1. **States** State represents situations during the life of an object. You can easily illustrate a state in Smart Draw by using a rectangle with rounded corners.



2. **Transition** A solid arrow represents the path between different states of an object. Label the transition with the event that triggered it and the action that results from it. A state can have a transition that points back to itself.



3. **Initial State** A filled circle followed by an arrow represents the object's initial state.



4. **Final State** An arrow pointing to a filled circle nested inside another circle represents the object's final state.



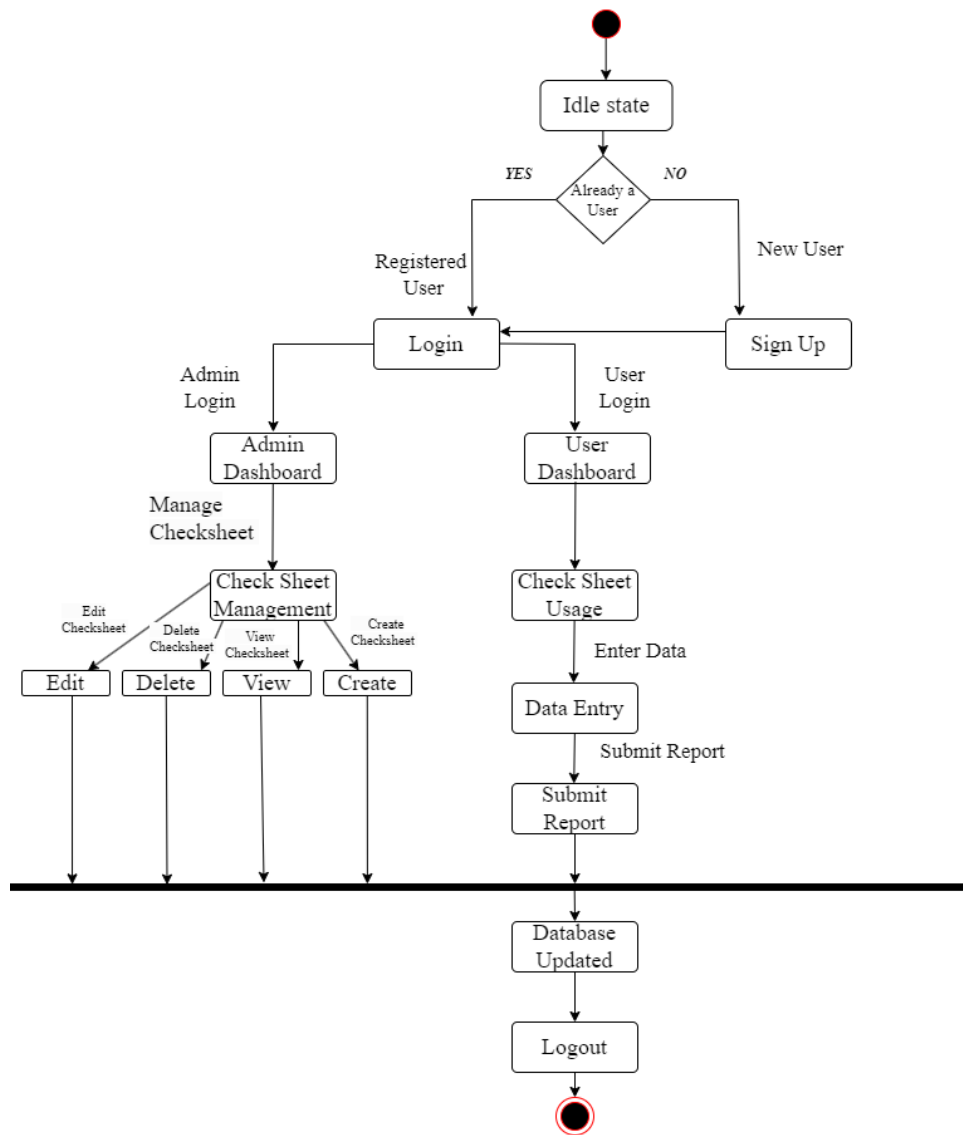





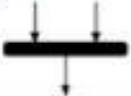





Figure 9: State Chart

4.3.1.4 Activity Diagram

Activity diagrams are used to describe the business and operational step-by- step workflows of components in a system. Rounded rectangles represent activities, Diamonds represent decisions, a black circle represents the start (initial state) of the workflow, and an encircled black circle represents the end (final state). Arrows run from the start towards the end and represent the order in which activities happen.

Basic Notations:

| | |
|--|---|
| Initial Node  | A black circle is the standard notation for an initial state before an activity takes place. It can either stand alone or you can use a note to further elucidate the starting point. |
| Activity  | The activity symbols are the basic building blocks of an activity diagram and usually have a short description of the activity they represent. |
| Control Flow  | Arrows represent the direction flow of the flow chart. The arrow points in the direction of progressing activities. |
| Branch  | A marker shaped like a diamond is the standard symbol for a decision. There are always at least two paths coming out of a decision and the condition text lets you know which options are mutually exclusive. |
| Fork  | A fork splits one activity flow into two concurrent activities |
| Join  | A join combines two concurrent activities back into a flow where only one activity is happening at a time. |
|  | The final flow marker shows the ending point for a process in a flow. The difference between a final flow node and the end state node is that the latter represents the end of all flows in an activity. |
|  Complete Activity Flow | The black circle that looks like a selected radio button is the UML symbol for the end state of an activity. As shown in two examples above, notes can also be used to explain an end state. |
| Notes  | The shape used for notes. |

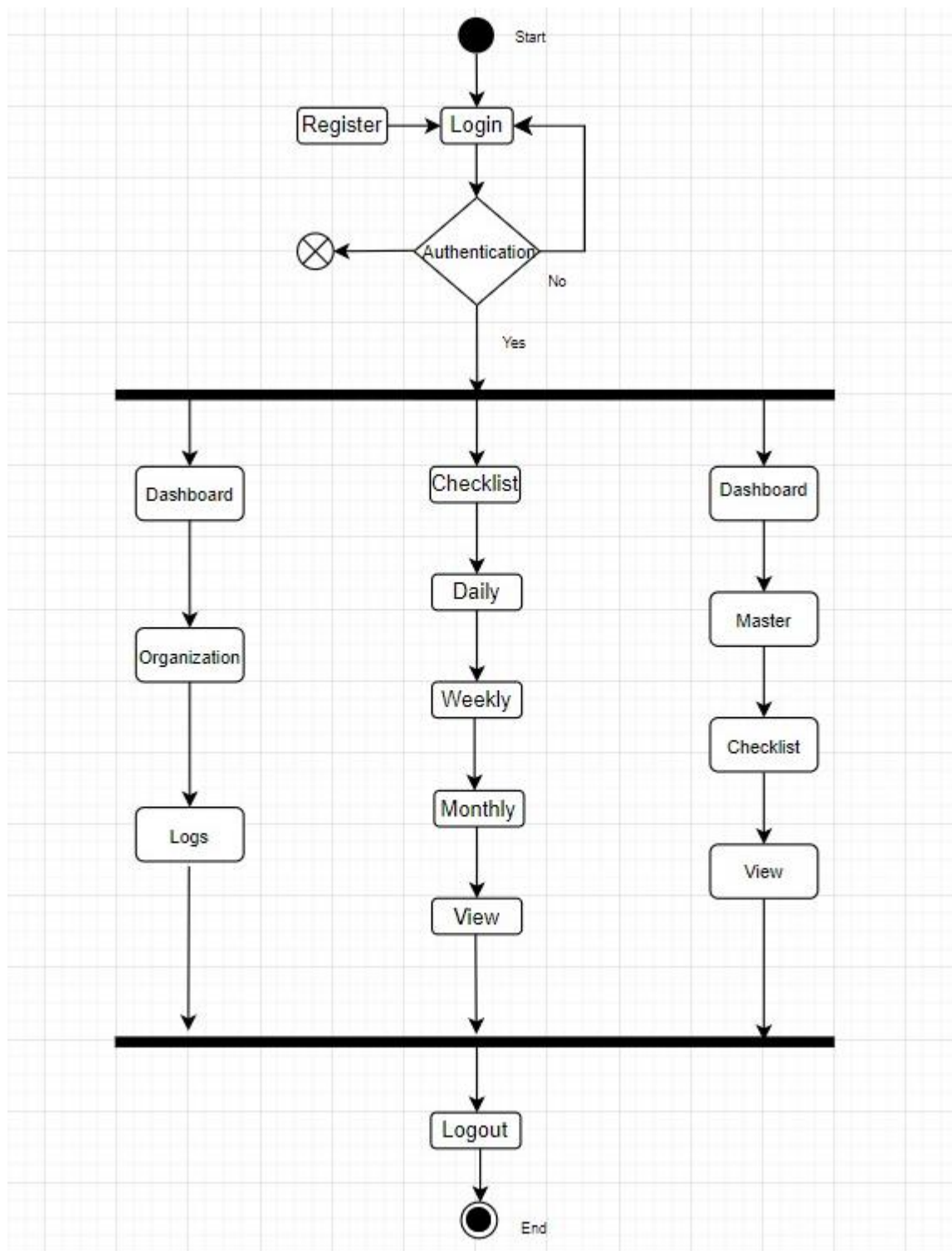


Figure 10: Activity Diagram

4.4. User interface design

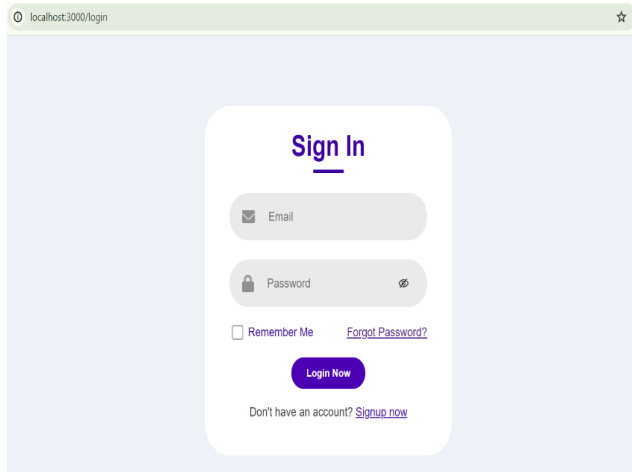


Figure11: Super Admin Login Page

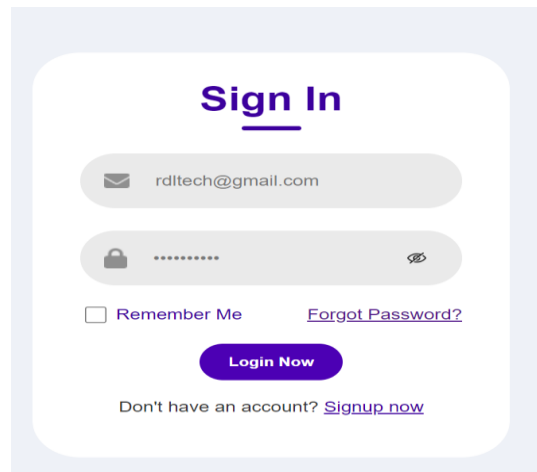


Figure 12: Admin login Page

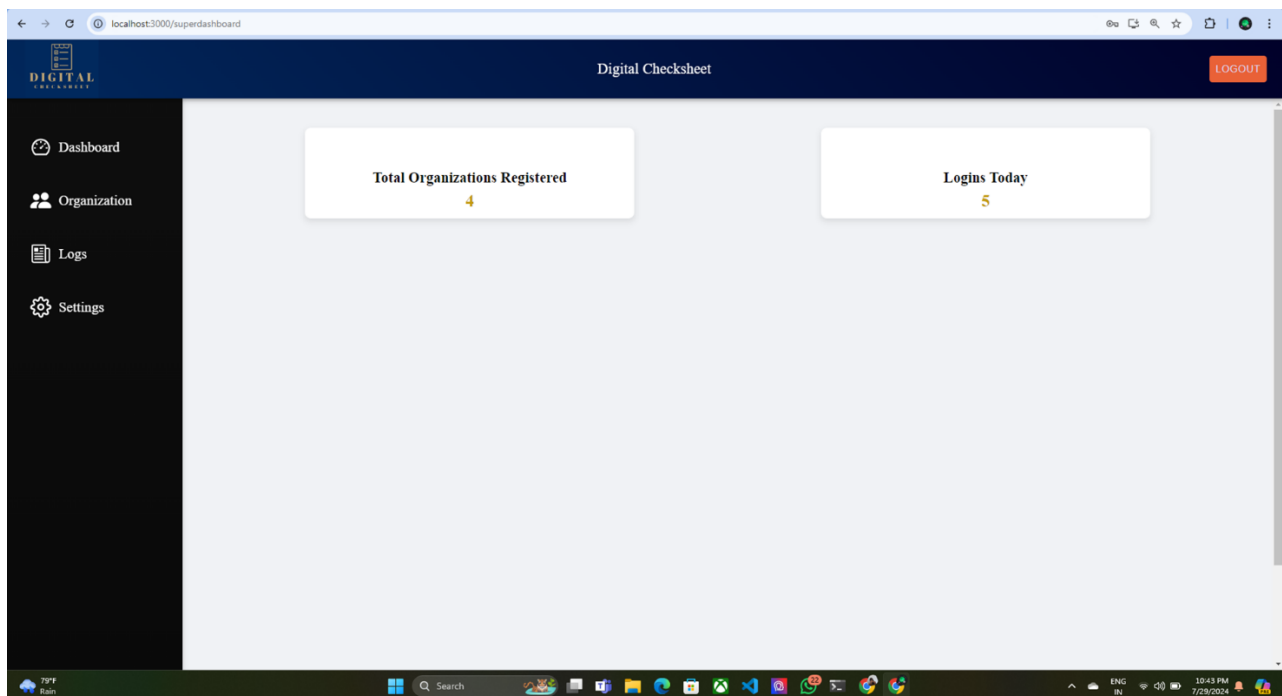


Figure 13: Organization Dashboard

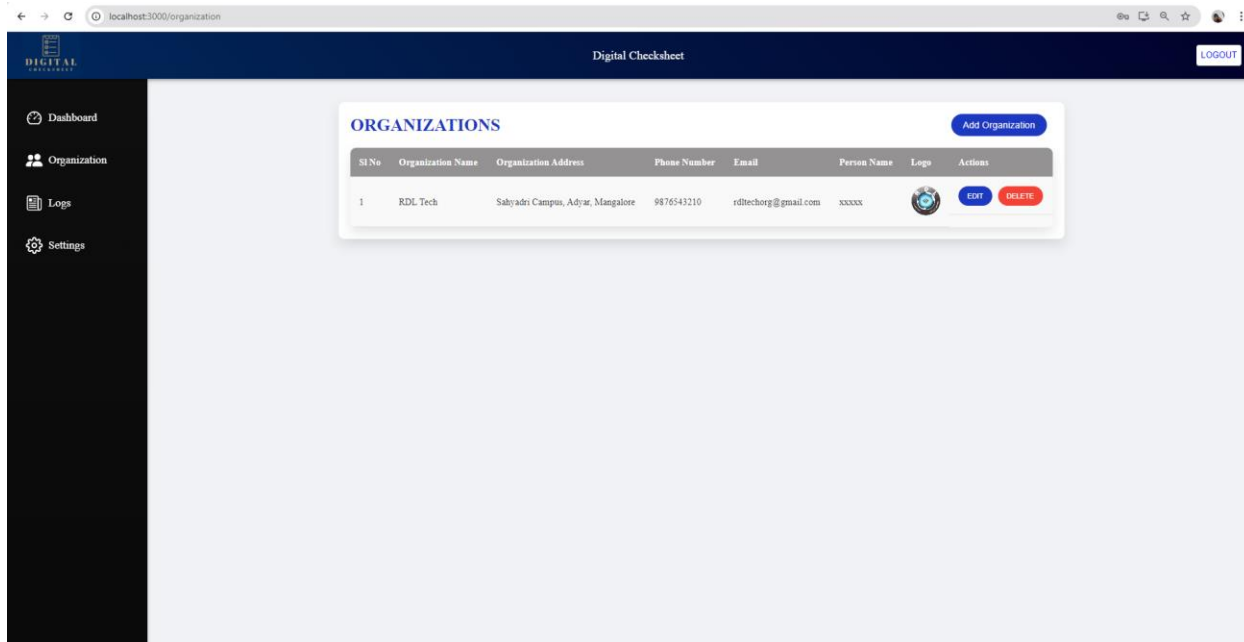


Figure14: Add Organization

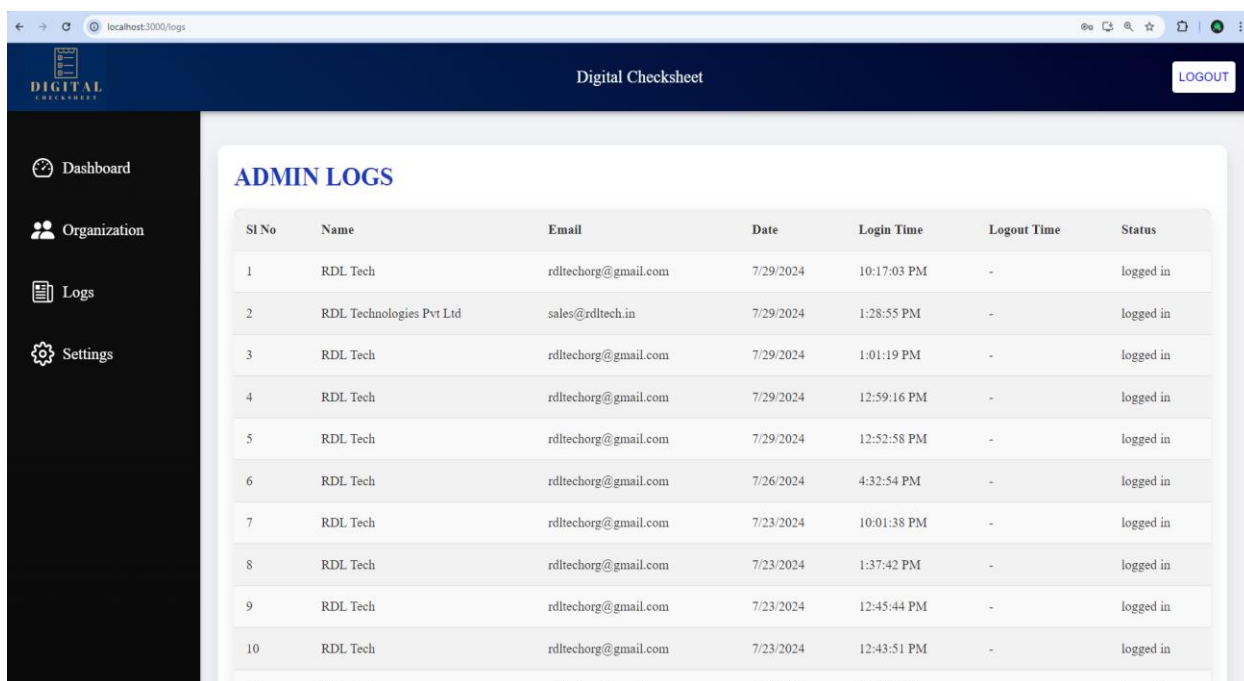


Figure15 Admin Logs

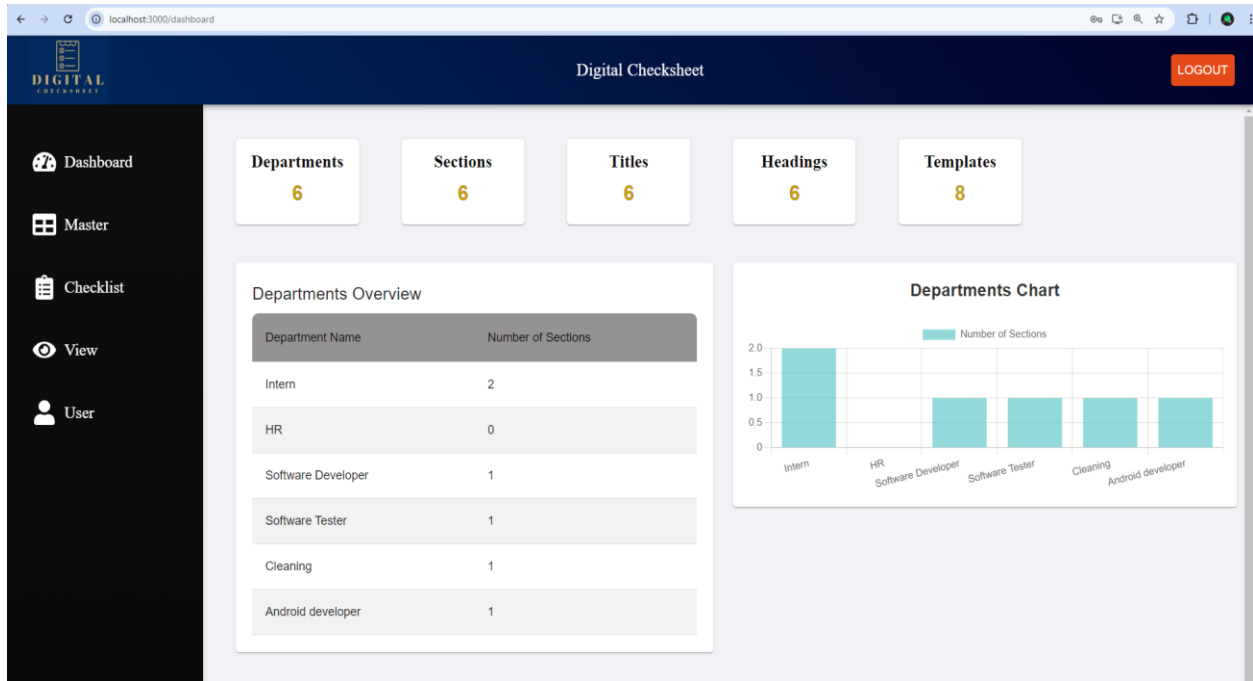


Figure16 Admin Dashboard

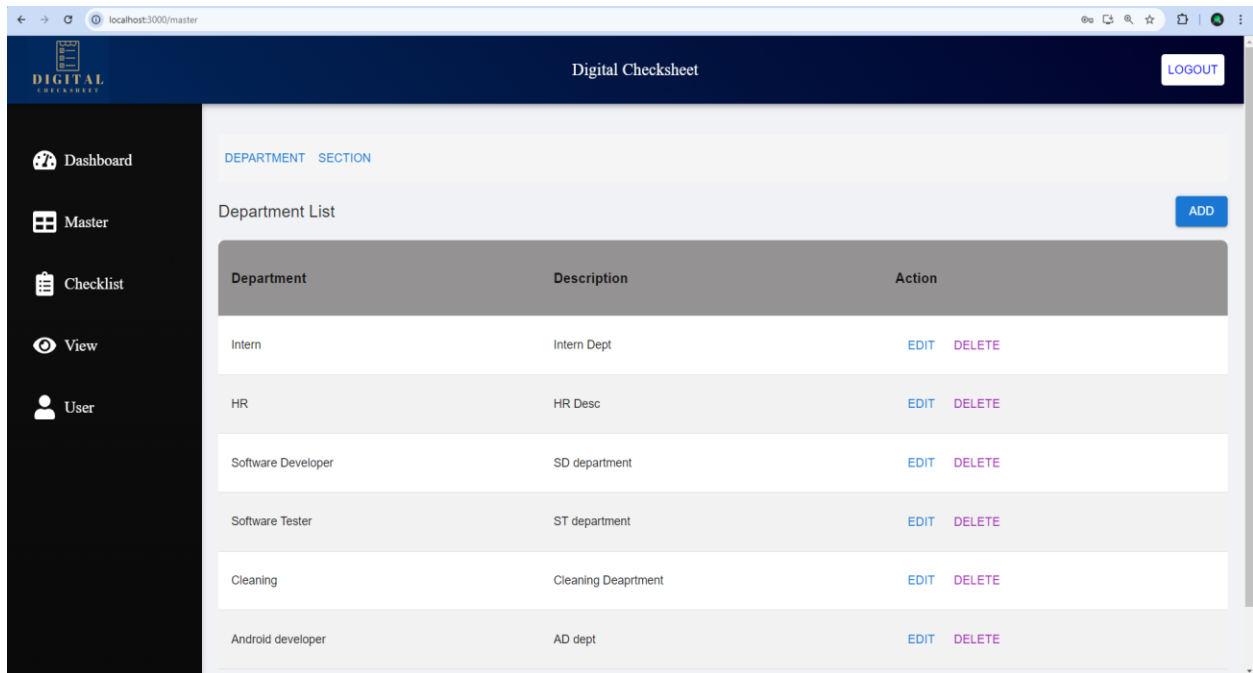


Figure 17 Department View

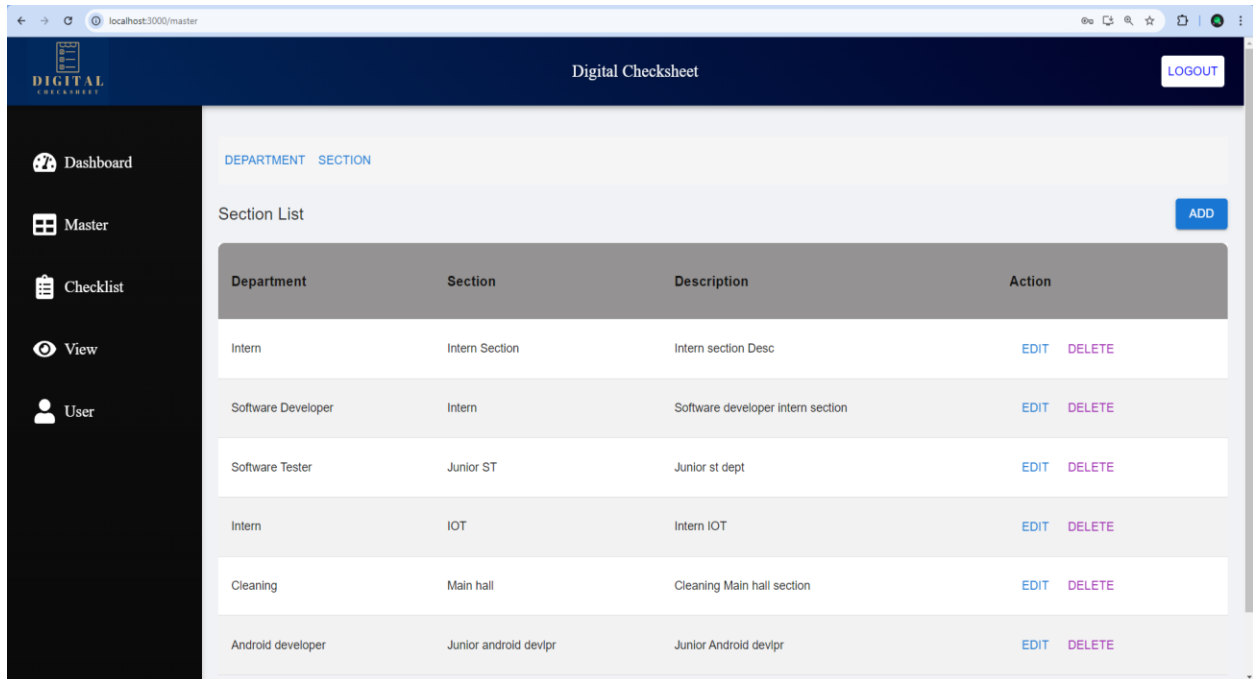


Figure 18 Section View

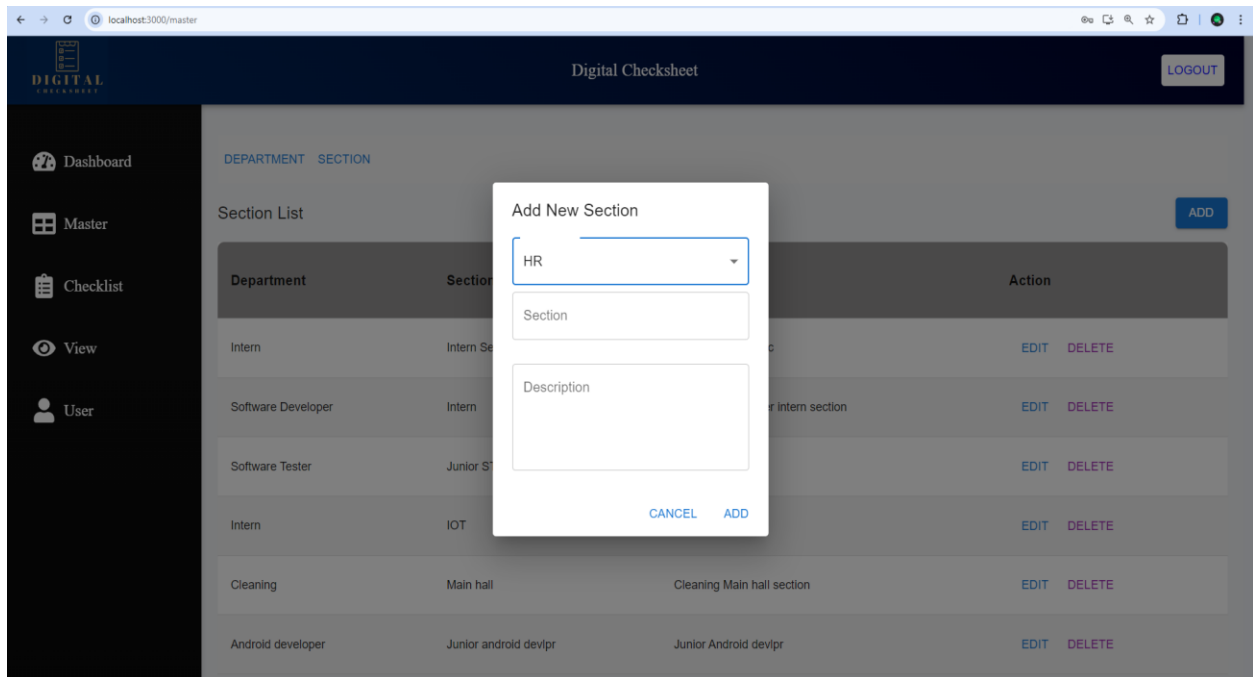


Figure 19 Add Section

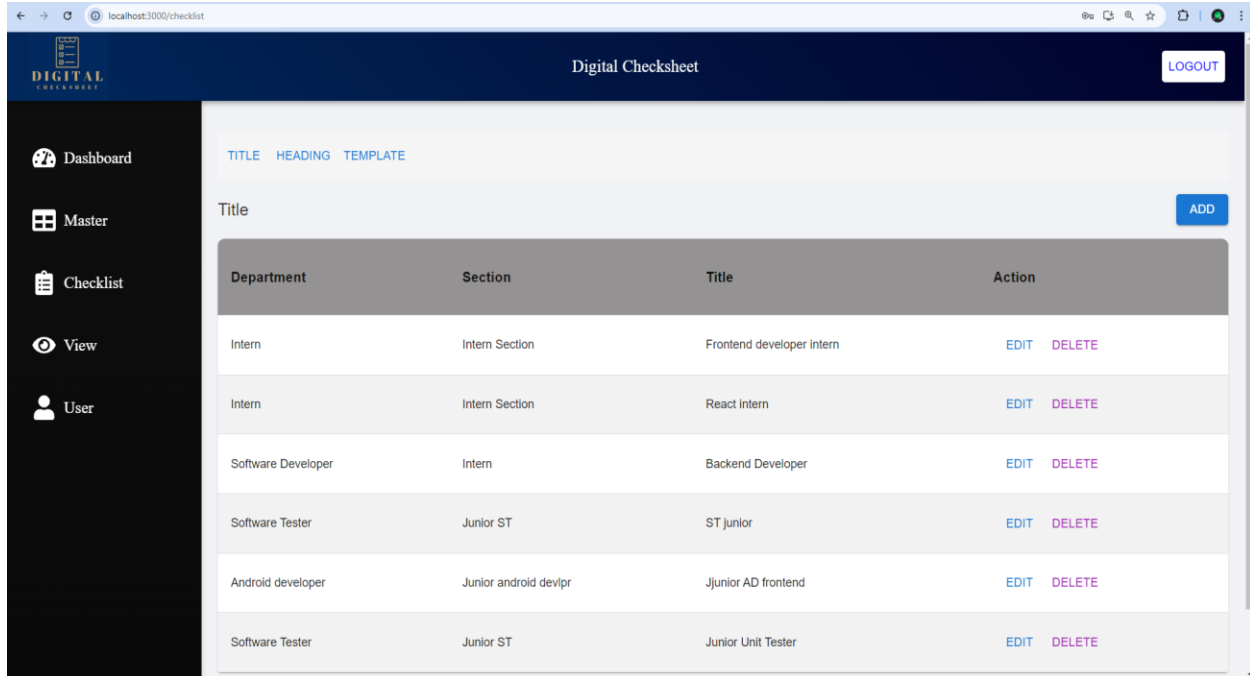


Figure 20 Title View

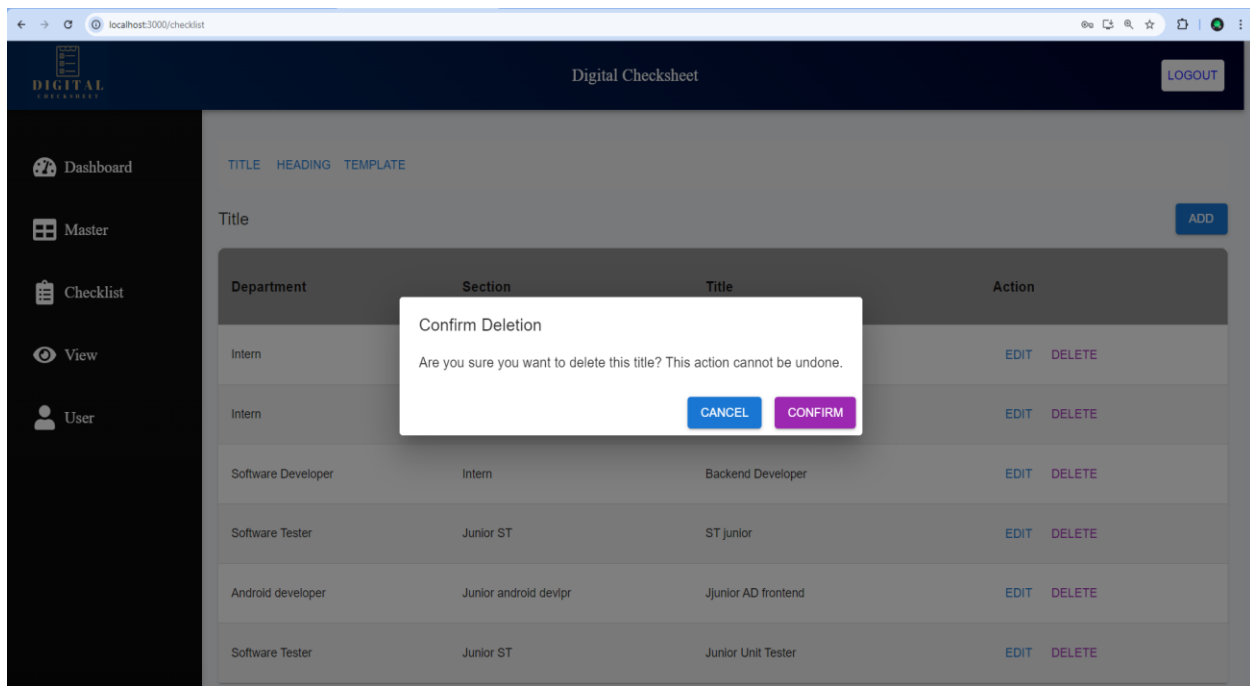


Figure 21 Confirm box while deleting

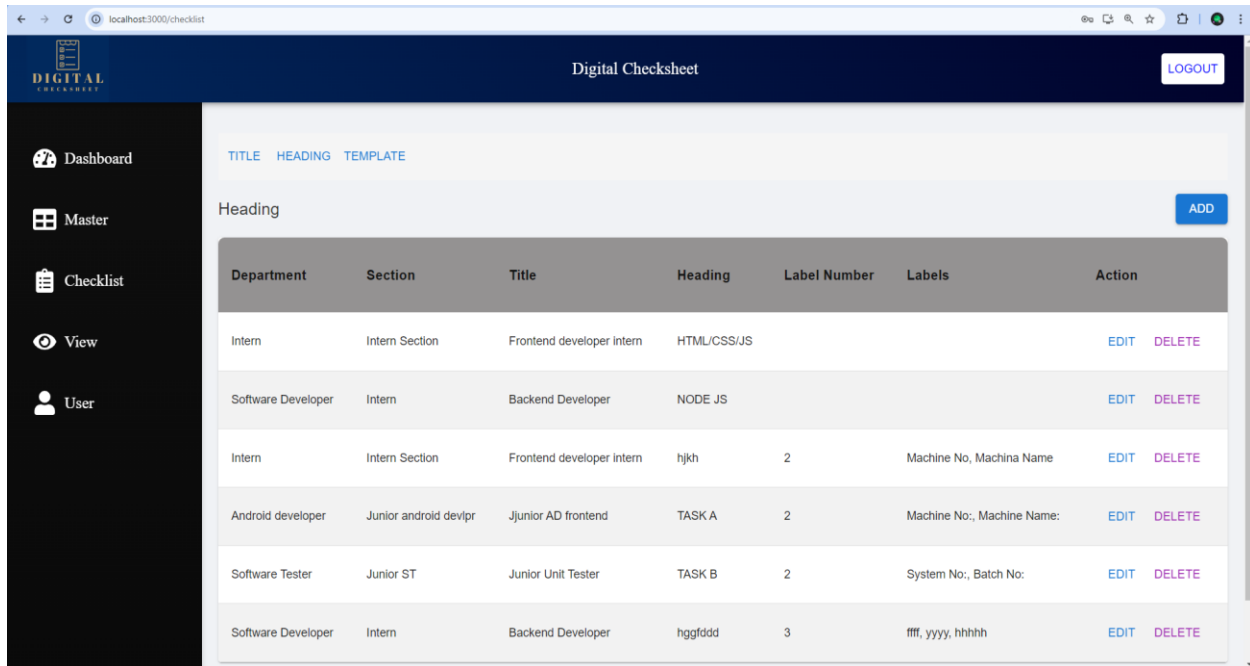


Figure 22 View Heading

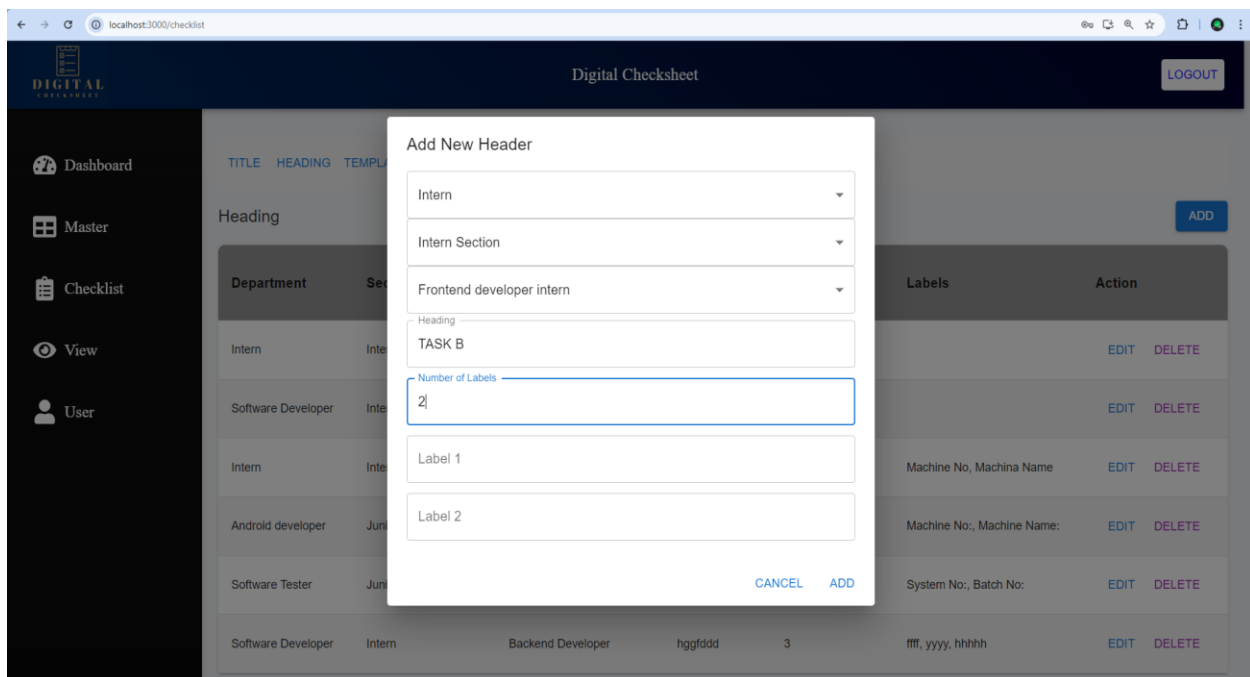


Figure 23 Add Heading

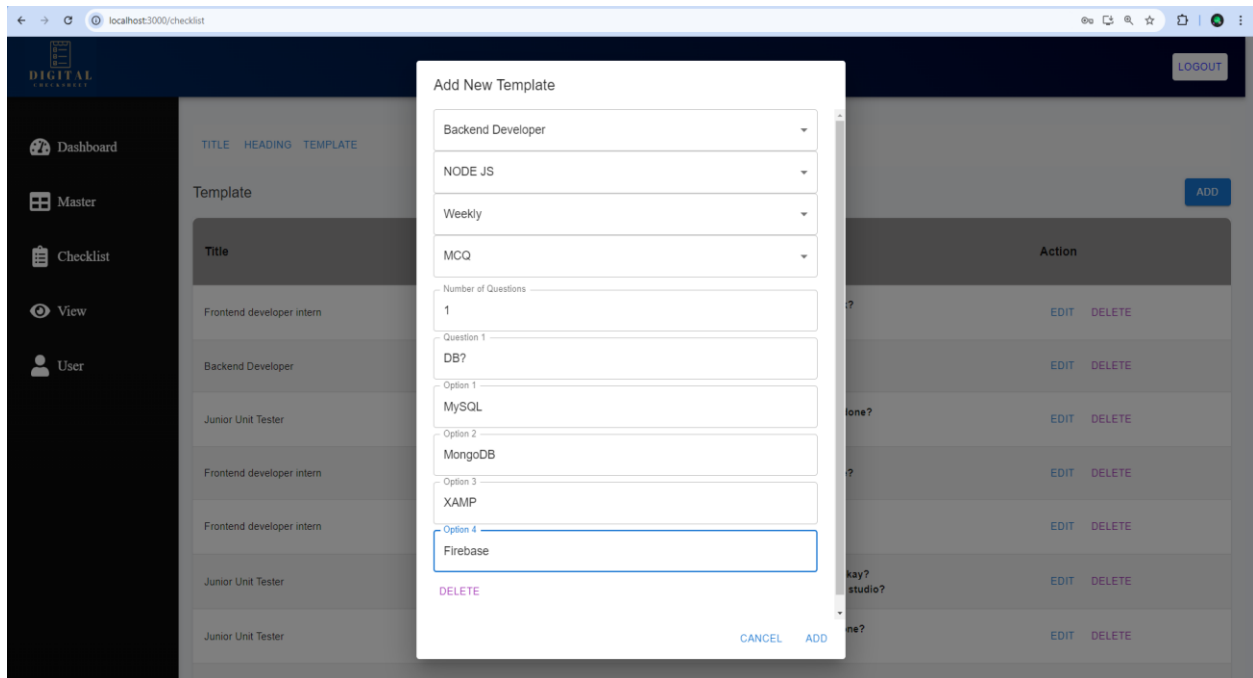


Figure 24 Create Template

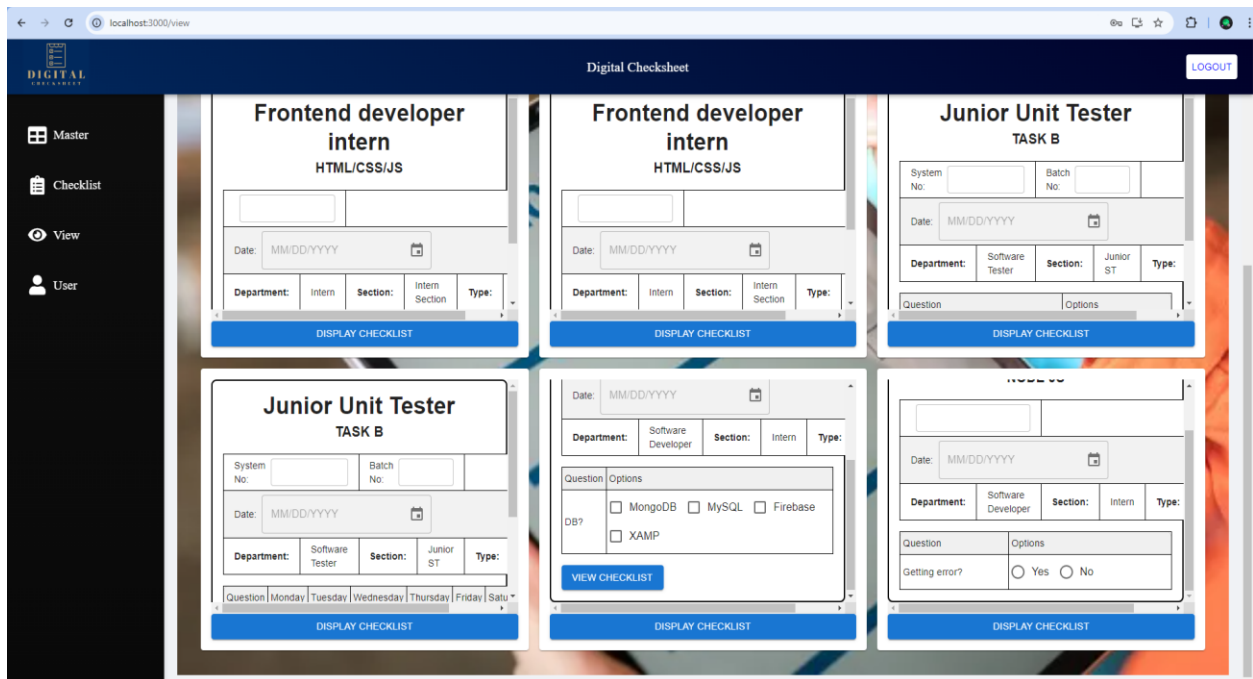


Figure 25 View Checklist

The screenshot shows a web application titled "Digital Checksheet" with a sidebar menu containing "Dashboard", "Master", "Checklist", "View", and "User". The main content area displays a checklist for "Junior Unit Tester TASK B". At the top, there is a "BACK TO CHECKLIST VIEW" button. Below the title, there are input fields for "System No:", "Batch No:", and "Date: MM/DD/YYYY". A table below these fields shows "Department: Software Tester", "Section: Junior ST", and "Type: weekly". The main checklist table has columns for "Question", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", and "Sunday". The first two rows of questions are "Which are the task you done?" and "Is there any issue?".

| Question | Monday | Tuesday | Wednesday | Thursday | Friday | Saturday | Sunday |
|------------------------------|--------|---------|-----------|----------|--------|----------|--------|
| Which are the task you done? | | | | | | | |
| Is there any issue? | | | | | | | |

Figure 26 Check sheet

The screenshot shows the same "Digital Checksheet" application, but with a checklist for "Backend Developer NODE JS". At the top, there is a "BACK TO CHECKLIST VIEW" button. Below the title, there are input fields for "System No:", "Batch No:", and "Date: MM/DD/YYYY". A table below these fields shows "Department: Software Developer", "Section: Intern", and "Type: daily". The main checklist table has columns for "Question" and "Options". The first row of questions is "DB?". The options listed are "MongoDB", "MySQL", "Firebase", and "XAMP". At the bottom, there is a "VIEW CHECKLIST" button.

| Question | Options |
|----------|---|
| DB? | <input type="checkbox"/> MongoDB <input type="checkbox"/> MySQL <input type="checkbox"/> Firebase <input type="checkbox"/> XAMP |

Figure 27 Check sheet

View Report

[BACK TO CHECKLIST VIEW](#)

Junior Unit Tester TASK B

System No: Batch No:

Date: 07/29/2024

Department: Software Tester Section: Junior ST Type: monthly

| Question | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|----------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Admin module backend done? | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| User module? | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

Figure 28 Check sheet

Users Management

Unverified Users


| ID | First Name | Last Name | Email | Actions |
|----|------------|-----------|----------------|------------------------|
| 4 | Zabra | Fathima | fath@gmail.com | Verify |

All Verified Users

| ID | First Name | Last Name | Phone | Organization ID | Email | Actions |
|----|------------|-----------|------------|-----------------|----------------------------|---|
| 1 | kadeejath | Salaha | 9876543210 | 5 | kadeejasalaha123@gmail.com | Edit Delete |
| 2 | Kadeejath | Salaha | 9645176669 | 5 | kadeejasalaha123@gmail.com | Edit Delete |
| 3 | Wazri | Shfq | 1234567890 | 5 | waazi@gmail.com | Edit Delete |


Fig 29 Manage User

Sign Up




Enter First Name

First Name is required




Enter Last Name

Last Name is required




Enter Phone Number

Phone number is required




Select Organization

Please select your organization




Enter Email

Email is required



Enter Password



Password is required


☒ I agree to the [Terms and Policies](#)

Sign Up


Already have an account? [Sign In](#)

Figure 29: User Signup


Sign In



fath@gmail.com



.....



Your account is not verified yet.

☐ Remember Me [Forgot Password?](#)


Login Now

Don't have an account? [Signup now](#)

Figure 30: User Login

Forgot Password

Email



rdluser@gmail.com

Send Reset Link

Figure 31: Forgot Password

Dashboard

Checklist

Digital University

Digital Checksheet

LOGOUT

Date: 09/01/2024

Department: Technical Support

Section: Non voice

Type: monthly

| Question | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| There any complaintraise? | | | | | | | | | | | | | | | | | | | | | | | | | |

SUBMIT

CLEAR

Backend

TASK C

System No:

Batch No:

Date: MM/DD/YYYY

Department: Backend Developer

Section: Intern

Type: daily

| Question | Response |
|---------------------|----------|
| Backend Technology? | |
| Version Control? | |

SUBMIT

VIEW CHECKLIST

CLEAR

Figure:32 User Submission

4.5. Security Issues

The system administrator provided non-shareable login credentials to other users so they could access it. This ensures that the users will receive accurate data by preventing the authenticity of the data and making sure that those without authentication are not permitted to change the data already present in the database.

4.6 Test Case Design Software testing:

It is the process of running a software programme with the goal of detecting problems. It is used to determine the correctness, completeness, and quality of generated computer software. Quality assurance of software: It entails the entire software development process being monitored and improved, as well as making sure that any established standards and processes are followed and that any issues are identified and fixed.

SOFTWARE DEVELOPMENT LIFE CYCLE:

- Requirements
- Analysis
- Design
- Coding
- Testing
- Maintenance

APPROACHES TO TESTING: There are two fundamental methods for testing:

- Black-Box Testing
- White-Box Testing

BLACK-BOX TESTING:

The program's structure is not considered during black box testing. The internals of the module or programme are not taken into consideration when selecting test cases; instead, test cases are chosen exclusively based on the requirements or specifications of the programme or module. When conducting black-box testing, the tester is only aware of the possible inputs and the intended output of the system. In functional testing, the requirements or specifications of the system or module serve as the basis for selecting test cases. Functional or behavioural testing are other names for this

type of testing.

WHITE-BOX TESTING:

Black-box testing is more concerned with functionality than actual programme implementation because white box testing is only concerned with the function that the tested programme is supposed to perform and ignores the internal structure of the programme that is responsible for implementing that function. On the other hand, white box testing focuses on evaluating how the programme is put into practise. The purpose of this testing is to put the various programming structures and data structures utilised in the programme to the test, not all possible input or output conditions (although that could be a by-product). Structural testing is another name for white-box testing.

CHAPTER 5
IMPLEMENTATION
AND TESTING

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1 Implementation Approaches

The project's implementation stage is when the theoretical design is translated into a functional system, providing users confidence that the new system will function successfully and efficiently. It requires thorough planning, research of the current system and its implementation limitations, design of techniques to achieve the changeover, evaluation of the changeover methods, and implementation of the changeover. Other than planning, the main tasks involved in getting ready for deployment include user education and training. Implementing a fully working system with objectives accomplished in priority order and with efficient training is the goal of the system implementation phase. This phase ends with a correctly operating system and users that have received the appropriate training. The effort required for system analysis and design just for implementation will increase in proportion to how complicated the system being implemented is. Quality, performance, baselines, libraries, and debugging issues are addressed during the implementation phase. The product itself is the final output. The system is constructed in the implementation phase in accordance with the specifications from the earlier phases. Code writing, code reviews, tests, component selection for integration, configuration, and integration are all included in this.

The following items are part of the implementation:

- Thorough planning
- System and constraint investigation.
- Build the strategies to accomplish the charge over
- Educating the staff about the modified phase.

5.2 Coding Details and Code Efficiency

The development of the Digital Check Sheet (DCS) application adhered to best practices and coding standards to ensure code consistency, efficiency, and maintainability. The following practices were implemented to enhance code quality and application performance:

Code Consistency and Efficiency: To achieve a lightweight and reliable application, coding standards were followed meticulously. This included the use of meaningful variable names, consistent indentation, and clear documentation.

IDE Features and Code Reviews: The IDE's built-in features and regular code reviews were utilized to identify and eliminate inefficiencies in the codebase. This process helped to address common issues such as:

- **Unused Local Variables:** Removal of variables that were declared but never used.

Empty Catch Blocks: Ensuring that exception handling blocks are properly managed or removed if unnecessary.

- **Unused Parameters:** Elimination of function parameters that were not used within the functions.
- **Empty 'If' Statements:** Removal of conditional statements that did not affect the code's execution.
- **Unused Private Methods:** Removal of private methods that were defined but not called within the code.
- **Inappropriate Variable and Method Names:** Ensuring variable and method names are descriptive and of appropriate length.

Code Coverage Tools: The IDE's code coverage tools were employed to assess the effectiveness of the coding practices. This included evaluating:

- **Branch Coverage:** Ensuring all branches of the code were tested.
- **Loop Coverage:** Verifying that loops within the code were adequately tested.
- **Strict Condition Coverage:** Checking that all conditions and combinations of conditions were evaluated during testing.

These practices and tools collectively contributed to the development of a robust and efficient DCS application, ensuring high code quality and reliable performance.

5.3 Testing Approach

The most crucial stage of software development is testing. The system is using test data that has undergone extensive testing. Errors are identified during testing and fixed. To use the created system, users must undergo training. Unit testing, integration testing, and system testing are all used to test the CRM product. With the unit testing method of software development, tests are created for every feature of the programmer. As and when the modules are developed, this is done. Each module has undergone extensive testing and integration before integration testing is carried out to determine whether the modules interact with one another as intended. It is carried out to

remove faults and errors brought up by integration.

Lastly, system testing examines how well a complete, integrated system complies with the requirements that are set forth for it. Black box testing includes system testing, which should not call for any understanding of the logic or code's inner structure. Prior to the start of the live operation, the system is tested to make sure it functions correctly. System testing makes the logical premise that the objective will be effectively attained if every component of the system is correct. Before the proposed system is ready for user acceptability testing, a number of tests are run on it.

Qualities of a Good Test:

- Tests are likely to find faults
- There is no redundancy
- It is both basic and complex enough

5.3.1 Unit Testing

The smallest unit of the module is the focus of unit testing. to determine whether each software module is functioning correctly and producing the required results when given the appropriate inputs. In the unit test, every validation and condition are checked at the module level. Control routes are put to the test to make sure information enters, exits, and flows appropriately into and out of the software unit being tested. To guarantee that the modules function at boundaries, boundary conditions are evaluated.

5.3.2 Integration Testing

Creating an incremental technique to reduce the complexity of full actions among components as they are added to the system is one of integration testing's main objectives. creating a component as it is added to the system, creating a timetable for its development and integration to make the modules accessible as required, and creating test cases to show the system's viability.

5.3.3 User Acceptance Testing

This form of testing is likely the most crucial because it is carried out by the Quality Assurance Team, which will determine whether the application satisfies the client's requirements and the intended specifications. The application will be tested by the QA team using a collection of pre-written scenarios and test cases.

5.3.4 Performance Testing

Instead of detecting software flaws, it is mostly utilized to pinpoint any performance or bottleneck issues.

5.3.5 Load Testing/Stress Testing

By subjecting a software to its maximum load in terms of accessing and modifying vast amounts of input data, it is tested for performance. Both regular and peak load situations are possible. This kind of testing identifies the software's maximum capacity and its peak-use behavior. The functionality of a piece of software is examined during stress testing. For instance, it can include removing certain resources or putting a load above the permitted load limit.

5.4. Modification and Improvement

We regularly check them once the application is put into production to make sure everything is operating as it should. When problems are discovered, they are fixed, thoroughly tested, and then incorporated into the current product.

CHAPTER 6

RESULTS AND DISCUSSION

CHAPTER 6: RESULTS AND DISCUSSION

6.1 Test Reports

Super Admin Login

| SL NO. | Test Case | Expected Result | Test Result |
|--------|-----------------------|--|-------------|
| 1 | Email is empty | Prompts email is empty | Success |
| 2 | Password is empty | Prompts password is empty | Success |
| 3 | Password is incorrect | Prompts entered information is incorrect | Success |
| 4 | Email is incorrect | Prompts entered information is incorrect | Success |

Add Organization

| SL NO. | Test Case | Expected Result | Test Result |
|--------|----------------------------------|--|-------------|
| 1 | If Organization Name is empty | Prompts the user to enter the Organization Name | Success |
| 2 | If Organization Address is empty | Prompts the user to enter the Organization Address | Success |
| 3 | If the Phone Number is empty | Prompts the user to enter the Phone Number | Success |
| 4 | If the email is empty | Prompts the user to enter the email | Success |
| 5 | If the Person Name is empty | Prompts the user to enter the Person Name | Success |
| 6 | If the Logo is empty | Prompts the user to select the logo | Success |

Admin Login

| SL NO. | Test Case | Expected Result | Test Result |
|--------|-----------------------|--|-------------|
| 1 | Email is empty | Prompts email is empty | Success |
| 2 | Password is empty | Prompts password is empty | Success |
| 3 | Password is incorrect | Prompts entered information is incorrect | Success |
| 4 | Email is incorrect | Prompts entered information is incorrect | Success |

Master

| SL NO. | Test Case | Expected Result | Test Result |
|---------------|--------------------------------------|---|--------------------|
| 1 | If Department Name is empty | Prompts the user to enter the Department Name | Success |
| 2 | If Description of department empty | Prompts the user to enter the Description of department | Success |
| 3 | If Department Name is not selected | Prompts the user to select the Department Name | Success |
| 4 | If Section Name is empty | Prompts the user to enter the Section Name | Success |
| 5 | If Description of Section Name empty | Prompts the user to enter the Section Name | Success |

Checklist

| SL NO. | Test Case | Expected Result | Test Result |
|---------------|------------------------------------|--|--------------------|
| 1 | If Department Name is not selected | Prompts the user to select the Department Name | Success |
| 2 | If Section Name is not selected | Prompts the user to select the Section Name | Success |
| 3 | If Title is empty | Prompts the user to enter the Title Name | Success |
| 4 | If Department Name is not selected | Prompts the user to select the Department Name | Success |
| 5 | If Section Name is not selected | Prompts the user to select the Section Name | Success |
| 6 | If Title is not selected | Prompts the user to select the Title | Success |
| 7 | If Heading is empty | Prompts the user to enter the Heading | Success |
| 8 | If number of label is empty | Prompts the user to enter the | Success |

| | | | |
|----|----------------------------------|--|----------------|
| | | number of labels | |
| 9 | If Name of label is empty | Prompts the user to enter the Name of label | Success |
| 10 | If Title is not selected | Prompts the user to select the Title | Success |
| 11 | If Heading is not selected | Prompts the user to select the Heading | Success |
| 12 | If Template is not selected | Prompts the user to select the Template | Success |
| 13 | If Question type is not selected | Prompts the user to select the Question type | Success |
| 14 | If number of Question is empty | Prompts the user to enter the number of Question | Success |
| 16 | If Question is empty | Prompts the user to enter the Questions | Success |

6.2 Identification

The Digital Check Sheet (DCS) application identifies key entities essential to its functionality and organization. These entities include the Super Admin, Admin, User, Check Sheets, Tasks, Organizations, and Notifications. Each entity has unique attributes and roles within the system. The Super Admin manages organizational details and monitors login activities of Admins. Admins create user logins, design, and manage check sheets, and assign tasks to users. Users select and complete check sheets, enter task data, and submit completed tasks for review. Check Sheets serve as templates for routine tasks, categorized into daily, weekly, and monthly schedules.

6.3 Type

The Digital Check Sheet (DCS) application describes the kind of entities it handles, specifying the nature and roles of each entity within the system.

6.3.1 Super Admin

The Super Admin manages overall organizational details and monitors login activities of Admins. They have the authority to add and remove organizations, monitor system usage, and ensure the security and integrity of the data within the application.

6.3.2 Admin

Admins have the ability to create user logins, design, and manage check sheets for daily, weekly, and monthly tasks. They can also assign tasks to users, oversee task progress, and ensure that check sheets are completed accurately and on time.

6.3.3 User

Users can view and select check sheets assigned to them, perform the necessary actions to complete tasks, and enter task data into the system. Once tasks are completed, users submit the data for review and approval by the Admin.

This structured approach to entity management ensures that each role within the DCS application is clearly defined and that responsibilities are efficiently allocated to maintain an organized and effective workflow.

6.4 Function

Entity functions manage the state of an entity explicitly, rather than implicitly representing state via control flow. Entities provide a means for scaling out applications by distributing the work across many entities, each with a modestly sized state.

6.5 Interface

Users can log in to the Digital Check Sheet (DCS) application using any device, providing a flexible and accessible interface. The DCS software facilitates the secure and efficient storage of organizational data, including check sheets, task details, and user information. With the help of the DCS software, employees can easily process check sheet data and track task progress. The software system offers an efficient way to store and recall necessary information, ensuring that all data is available as and when required. This user-friendly interface enhances productivity and streamlines the management of routine tasks within the organization.

6.6 Processing

The Digital Check Sheet (DCS) application operates as follows: The Super Admin is responsible for creating and managing Admin accounts. Admins handle the creation and management of check sheets, which are then assigned to Users. Users select their assigned check sheets and complete the tasks outlined in them. Once completed, the Users submit their check sheets for review. The

Admin reviews these submissions, approves or requests modifications, and updates the task status accordingly. This streamlined process ensures that tasks are completed efficiently and accurately within the system.

6.7 Data

The Digital Check Sheet (DCS) application has been designed with a user-friendly interface to enhance navigation and accessibility. The system allows users to efficiently manage and interact with data through intuitive features and easy-to-use options. The interface includes comprehensive guides and help sections for all available options, ensuring that users can easily access and utilize the various functionalities of the application. This design approach facilitates smooth data management and enhances the overall user experience.

6.8 Dependency Description

- **System Uptime:** The Digital Check Sheet (DCS) application must be operational 24/7 to ensure continuous data capture and accurate task management. All functions within the application should be fully operational at all times to support seamless workflow and task execution.
- **Data Logging:** Accurate data logging is essential for the smooth functioning of the application. Proper logging helps in tracking task progress, managing check sheets, and minimizing human errors, ensuring reliable and efficient operation of the system.

6.9 Interface Description

- This application works on Internet, so an uninterrupted internet connection is must for this system to operate.
- Validation is done by the backend as the user enters the details.

CHAPTER 7: CONCLUSION

CHAPTER 7: CONCLUSION

7.1 Conclusion

The digital checklist project serves as an effective tool for streamlining and automating the management of organizational tasks. It is designed with three main modules: Super Admin, Admin, and User, each with specific functionalities to ensure efficient workflow and accountability.

- **Super Admin Module:**

The Super Admin can create and manage organizations by providing necessary details such as organization name, phone number, email, password, and logo. They also generate login credentials for Admins and have the ability to view detailed logs of user activities, including names, emails, and login times.

- **Admin Module:**

Admins, using the credentials provided by the Super Admin, can access the master page to add departments and sections. They can create checklists by specifying titles, headings, and the type of checklist (daily, weekly, monthly). Admins can also choose between task completion forms and feedback forms for each checklist. They can view all created checklists and monitor the submissions made by users, generating reports based on the collected data.

- **User Module:**

Users register with the desired organization to access specific checklists. They complete the checklists and submit them, after which the Admin can view and analyse the submitted data in the report section.

7.2 Limitations

- **Data Loss Risk:**

The application relies on a stable internet connection for data submission and management. In case of internet outages or device failures during data saving, there is a risk of data corruption and potential loss of critical information.

- **Lack of Human Interaction**

While the system automates many processes, it reduces the human element essential for building strong client relationships.

- **Maintenance Overhead**

Managing data backups and ensuring data integrity adds to the operational costs.

Security Concerns

Although the application adheres to high security standards, centralized data storage poses a risk. Malicious activities by disgruntled employees or external threats could compromise data integrity.

7.3 Future Scope of the Project

Enhanced Communication: Implementing a chat module to facilitate real-time communication between Admins and Users.

Advanced Planning: Developing features like a stowage plan for more comprehensive task management.

Improved User Experience: Making the website more responsive and user-friendly.

Multimedia Support: Enabling users to add images to their checklists for better documentation.

Real-Time Tracking: Introducing real-time tracking features for tasks and projects, similar to voyage tracking.

REFERENCES

Books Referred

1. An Integrated Approach to Software Engineering by Pankaj Jalote.
2. Unified Modeling Language User Guide –Grady Booch, James Rumbaugh, Ivar Jacobson
3. IEEE standard 10161998 recommended practice for software design descriptions
4. Software Engineering, 5th Edition, Addison by Wesley – Ian Somerville.

Websites Referred

1. <https://www.tutorialspoint.com/uml/index.htm>
2. <https://www.geeksforgeeks.org/unified-modeling-language-uml-state-diagrams>
3. <https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams>
4. <https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams>
5. <https://stackoverflow.com/>

GLOSSARY

1. SDD - Software Design Document
2. SRS - Software Requirements Specification.
3. IDE – Integrated Development Environment.
4. SQL – Sequential Query Language
5. EFC - Entity Framework Core
6. RAM – Random-Access Memory.
7. GB – Giga Byte.
8. DFD - Data Flow Diagram
9. ER - Entity Relationship Diagram
10. IEEE - The Institute of Electrical and Electronics Engineers.
11. OS - Operating System
12. DCS- Digital Check Sheet
13. RDL Tech- Research Design Lab Technology
14. API- Application Programming Interface
15. SPA- Single Page Application