BUILDNEST

Construction Project Management System

*Mini Project Report Submitted by*

# Amal Jose Kurian

**Reg. No.: AJC22MCA-2014**

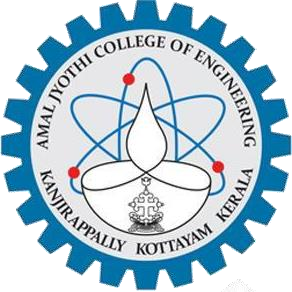
*In Partial Fulfillment for the Award of the Degree of*

**MASTER OF COMPUTER APPLICATIONS**

# (MCA TWO YEAR)

[Accredited by NBA]

# APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

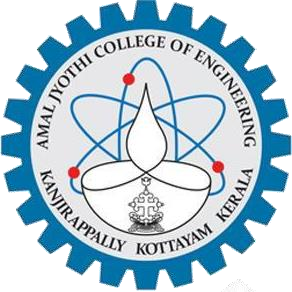


**AMAL JYOTHI COLLEGE OF ENGINEERING KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC. Koovapally, Kanjirappally, Kottayam, Kerala – 686518]

# 2023-2024

**DEPARTMENT OF COMPUTER APPLICATIONS AMAL JYOTHI COLLEGE OF ENGINEERING KANJIRAPPALLY**



CERTIFICATE

This is to certify that the Project report, “**BUILDNEST”** is the bona fide work of **Amal Jose Kurian (Regno: AJC22MCA-2014)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

**Mr. Rony Tom Ms. Meera Rose Mathew**

**Internal Guide Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose Head of the Department**

## DECLARATION

I hereby declare that the project report **“BUILDNEST”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

**Date: AMAL JOSE KURIAN**

**KANJIRAPPALLY Reg: AJC22MCA-2014**

# ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Ms. Rony Tom** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moralsupport, suggestions, and encouragement to make this venture a success.

AMAL JOSE KURIAN

# ABSTRACT

In the dynamic landscape of the construction industry, efficient project management is paramount for the advancement of established entities and the initiation of novel ventures. This essay explores a web application tailored to the unique needs of construction project management, embodying user-friendly features and functionalities. These aspects aim to refine and augment project oversight by addressing communication gaps, coordination issues, and progress tracking challenges encountered by industry professionals.

The proposed web application encompasses a set of key features to ensure comprehensive project management. The inclusion of user registration and authentication establishes a secure and personalized environment. Labor management and progress monitoring contribute to efficient project execution, while e-commerce capabilities facilitate seamless transactions. The responsive user interface enhances accessibility, and profile updation ensures the accuracy of user information. The system is structured into three modules, each catering to specific user needs. The administrative module focuses on user management, overseeing user registration, login, and the monitoring of other accounts. Clients benefit from functionalities such as user registration, login, and profile updates. Setting up preferences allows clients to articulate building specifications and requirements. The dashboard and analytics feature provides real-time insights into project progress, milestones achieved, and budget calculations. Contractors engage in user management tasks, including registration, login, and profile updates. Labor management involves detailing worker information and scheduling tasks and instructions to optimize workforce productivity. Engineers are granted user management capabilities, allowing for registration, login, and profile updates. They play a crucial role in managing client preferences, viewing proposals, and accepting or rejecting them. Additionally, engineers propose plan fees and manage payments for their services. The purchase manager module involves user registration and login, focusing on inventory control by ordering raw materials and regularly updating the inventory. Secure and timely payment transactions ensure a streamlined procurement process.

The incorporation of augmented reality for prototype building visualization, chatbots for enhanced communication, payment gateways for secure transactions, and a report generator for comprehensive analytics further enrich the capabilities of the web application. The web application is built on a robust foundation, utilizing HTML, CSS, and JavaScript for the frontend, while the backend is powered by the Python Django framework. This combination ensures a seamless and efficient user experience.

**CONTENT**

|  |  |  |
| --- | --- | --- |
| **SL. NO** | **TOPIC** | **PAGE NO** |
| **1** | **INTRODUCTION** |  |
| **1.1** | **PROJECT OVERVIEW** |  |
| **1.2** | **PROJECT SPECIFICATION** |  |
| **2** | **SYSTEM STUDY** |  |
| **2.1** | **INTRODUCTION** |  |
| **2.2** | **EXISTING SYSTEM** |  |
| **2.3** | **DRAWBACKS OF EXISTING SYSTEM** |  |
| **2.4** | **PROPOSED SYSTEM** |  |
| **2.5** | **ADVANTAGES OF PROPOSED SYSTEM** |  |
| **3** | **REQUIREMENT ANALYSIS** |  |
| **3.1** | **FEASIBILITY STUDY** |  |
| **3.1.1** | **ECONOMICAL FEASIBILITY** |  |
| **3.1.2** | **TECHNICAL FEASIBILITY** |  |
| **3.1.3** | **BEHAVIORAL FEASIBILITY** |  |
| **3.1.4** | **FEASIBILITY STUDY QUESTIONNAIRE** |  |
| **3.2** | **SYSTEM SPECIFICATION** |  |
| **3.2.1** | **HARDWARE SPECIFICATION** |  |
| **3.2.2** | **SOFTWARE SPECIFICATION** |  |
| **3.3** | **SOFTWARE DESCRIPTION** |  |
| **3.3.1** | **PHP** |  |
| **3.3.2** | **MYSQL** |  |
| **4** | **SYSTEM DESIGN** |  |
| **4.1** | **INTRODUCTION** |  |
| **4.2** | **UML DIAGRAM** |  |
| **4.2.1** | **USE CASE DIAGRAM** |  |
| **4.2.2** | **SEQUENCE DIAGRAM** |  |
| **4.2.3** | **STATE CHART DIAGRAM** |  |
| **4.2.4** | **ACTIVITY DIAGRAM** |  |
| **4.2.5** | **CLASS DIAGRAM** |  |
| **4.2.6** | **OBJECT DIAGRAM** |  |
| **4.2.7** | **COMPONENT DIAGRAM** |  |

|  |  |  |
| --- | --- | --- |
| **4.2.8** | **DEPLOYMENT DIAGRAM** |  |
| **4.2.9** | **COLLABORATION DIAGRAM** |  |
| **4.3** | **USER INTERFACE DESIGN USING FIGMA** |  |
| **4.4** | **DATABASE DESIGN** |  |
| **5** | **SYSTEM TESTING** |  |
| **5.1** | **INTRODUCTION** |  |
| **5.2** | **TEST PLAN** |  |
| **5.2.1** | **UNIT TESTING** |  |
| **5.2.2** | **INTEGRATION TESTING** |  |
| **5.2.3** | **VALIDATION TESTING** |  |
| **5.2.4** | **USER ACCEPTANCE TESTING** |  |
| **5.2.5** | **AUTOMATION TESTING** |  |
| **5.2.6** | **SELENIUM TESTING** |  |
| **6** | **IMPLEMENTATION** |  |
| **6.1** | **INTRODUCTION** |  |
| **6.2** | **IMPLEMENTATION PROCEDURE** |  |
| **6.2.1** | **USER TRAINING** |  |
| **6.2.2** | **TRAINING ON APPLICATION SOFTWARE** |  |
| **6.2.3** | **SYSTEM MAINTENANCE** |  |
| **7** | **CONCLUSION & FUTURE SCOPE** |  |
| **7.1** | **CONCLUSION** |  |
| **7.2** | **FUTURE SCOPE** |  |
| **8** | **BIBLIOGRAPHY** |  |
| **9** | **APPENDIX** |  |
| **9.1** | **SAMPLE CODE** |  |
| **9.2** | **SCREEN SHOTS** |  |

## List of Abbreviation

* IDE - Integrated Development Environment
* HTML - Hyper Text Markup Language
* CSS - Cascading Style Sheet
* SQL - Structured Query Language
* UML - Unified Modelling Language
* JS - JavaScript
* AJAX - Asynchronous JavaScript and XML Environment
* URL - Uniform Resource Locator
* PY – Python
* 1NF - First Normal Form
* 2NF - Second Normal Form
* 3NF - Third Normal Form
* PK - Primary Key
* FK - Foreign Key
* ORM - Object-Relational Mapping
* MVT - Model-View-Template
* MVC - Model-View-Controller
* RDBMS - Relational Database Management System

# CHAPTER 1 INTRODUCTION

### PROJECT OVERVIEW

The Construction Project Management System outlined in the abstract represents a comprehensive and innovative solution to the challenges prevalent in the construction industry. Encompassing four distinct modules—Project Admin, Engineers, Contractors, Clients, and Purchase Managers—the application integrates user-friendly features to enhance communication, coordination, and progress tracking. With key functionalities such as user registration, authentication, labor management, progress monitoring, and e-commerce capabilities, the system caters to a diverse range of stakeholders, including administrators, clients, engineers, contractors, and purchase managers. The proposed software, built on HTML, CSS, JavaScript for the frontend, and Python Django for the backend, leverages additional technologies like augmented reality, chatbots, and payment gateways. This project is poised to revolutionize construction project management, offering a streamlined and efficient platform for overseeing building projects and infrastructural development.

### PROJECT SPECIFICATION

The Construction Project Management System is a web application designed to facilitate seamless interactions among clients, engineers, contractors, purchase manager. It encompasses a range of functionalities tailored to meet the diverse needs of each user group.

### Admin:

* + - **User Management:**

User Registration and Login

Monitoring and managing other user accounts

### Site Management

Listing of Active Construction Sites Uploading Building permits and sanctions

### Authorization

Manage reports from Contractors

Authorizing for Construction (Approval or Reject)

### Clients:

* + - **User Management**

User Registration and Login Updating Profile

* + - **Setting up Preferences** Choosing the type of buildings Requirements and Specifications

### Dashboard and Analytics

Progress of Tasks Milestones Completed Budget Calculation

### Contractors:

* + - **User Management**

User Registration and LoginBasic Details

Updating Profile

### Labor Management

Details of Worker

Schedule Tasks and Instructions

### Engineer:

* + - **User Management**

User Registration and Login Updating Profile

* + - **Managing Client Preferences** Viewing client preferences Accept or Reject the Proposals
    - **Proposing Plan fee** Proposing a fee for plan Payments

### Purchase Manager:

* + - **User Management**

User Registration and Login

### Inventory Control

Ordering Raw materials Check and Updates Inventory

### Transactions

Secure and timely Payments

### SOFTWARE SPECIFICATION:

Frontend: HTML, CSS, JavaScript Backend: Django.

Database: SQLite.

Payment Gateway Integration: Razorpay.

# CHAPTER 2 SYSTEM STUDY

### INTRODUCTION

In the current Construction field, the process of initiating successful project is lacks a proper project management system. The Construction Project Management System initiates a comprehensive exploration into the evolving landscape of the construction sector. This investigation aims to assess the challenges faced by industry professionals in communication, coordination, and progress tracking, leading to the conceptualization of an innovative solution. By delving into the intricacies of the five modular components—Project Admin, Engineers, Contractors, Clients, and Purchase Managers—the system study endeavors to establish a foundation for a user-centric, technologically advanced platform that redefines project oversight in the construction industry.

### EXISTING SYSTEM

The existing system in the construction project management domain often grapples with challenges related to communication, coordination, and progress tracking. Traditional methods lack a cohesive and centralized approach, leading to inefficiencies in project execution. This underscores the need for a more streamlined and technologically advanced solution, prompting the conceptualization of a comprehensive web application that addresses these shortcomings and enhances the overall management of construction projects.

### NATURAL SYSTEM STUDIED

The natural system studied in the context of construction project management reflects the organic challenges and complexities inherent in the industry. Characterized by communication gaps, coordination issues, and a lack of centralized oversight, the current natural system often hinders efficient project execution. Recognizing the need for a more adaptive and responsive approach, the proposed web application seeks to emulate and enhance the natural system's functionality. By introducing modules tailored to the roles of Project Admin, Engineers, Contractors, Clients, and Purchase Managers, the system aims to seamlessly integrate with the existing dynamics of the construction sector, fostering a more harmonious and efficient project management ecosystem.

### DESIGNED SYSTEM STUDIED

The envisioned construction project management system is a strategic solution addressing limitations in the current natural system. Structured as a comprehensive web application with five modules—Project Admin, Engineers, Contractors, Clients, and Purchase Managers—it tailors functionalities to streamline stakeholder roles. Project Admin ensures efficient monitoring with a focus on user management and site oversight. Clients enjoy user-friendly features for preference setting and project tracking, while Contractors optimize labor management. Engineers handle client

preferences and payments, and Purchase Managers ensure effective inventory control. Key features such as user registration, authentication, and e-commerce bolster the system, complemented by advanced technologies like augmented reality and chatbots.

### DRAWBACKS OF EXISTING SYSTEM

* **Communication Gaps:** The existing system suffers from communication gaps, impeding the flow of information between stakeholders. This lack of effective communication can lead to misunderstandings, delays, and errors in project execution.
* **Coordination Challenges:** Coordination issues persist within the current system, hindering seamless collaboration among project participants. This can result in a disjointed workflow, impacting the overall efficiency of the construction process.
* **Limited Oversight:** The system lacks a centralized approach to project oversight, making it challenging to monitor and manage various aspects simultaneously. This limitation can lead to difficulties in tracking progress and ensuring adherence to project timelines.
* **Inefficient Progress Tracking:** The existing method of progress tracking is inefficient, relying on manual and disparate processes. This can result in inaccuracies, making it difficult to obtain real-time insights into project milestones and overall advancement.
* **Lack of User-Friendly Features:** The current system lacks user-friendly features, making it less intuitive for stakeholders to navigate and utilize. This can contribute to a steep learning curve for users, potentially slowing down processes and reducing overall system adoption.
* **Ineffective Budget Calculation:** Budget calculations within the current system are prone to errors and may lack real-time accuracy. This can lead to financial discrepancies, impacting the overall financial health of construction projects.
* **Limited Data Analytics:** The absence of a robust analytics component in the existing system hampers the ability to derive meaningful insights. This limitation restricts the capacity to make data-driven decisions for improved project management and optimization.
* **Security Concerns:** The current system may have security vulnerabilities, potentially risking unauthorized access and compromising sensitive project data. Addressing security concerns is crucial for maintaining the confidentiality and integrity of project information.

### PROPOSED SYSTEM

The proposed construction project management system represents a forward-thinking solution aimed at overcoming the drawbacks of the existing system. Through a comprehensive web application structured into five modules—Project Admin, Engineers, Contractors, Clients, and Purchase Managers—the system addresses communication gaps, coordination challenges, and limited oversight that characterize the current approach. With a focus on user-friendly features, efficient progress tracking, and advanced technologies like augmented reality and chatbots, the proposed system aims to revolutionize project management. Built on a robust software architecture using HTML, CSS, JavaScript for the frontend, and Python Django for the backend, the system is poised to enhance collaboration, streamline workflows, and provide a cohesive and technologically advanced platform for construction project stakeholders.

### ADVANTAGES OF PROPOSED SYSTEM

* **Enhanced Communication:** The proposed system addresses communication gaps, providing a platform for seamless and efficient communication among stakeholders. This improvement ensures that all team members are well-informed, reducing the likelihood of misunderstandings and delays.
* **Streamlined Coordination:** By structuring the system into distinct modules for different stakeholders, the proposed system facilitates streamlined coordination. This allows for better collaboration and ensures that each participant can contribute to the project with clarity and purpose.
* **Centralized Project Oversight:** Unlike the existing system, the proposed system offers centralized project oversight. This feature enables administrators to monitor and manage multiple aspects of construction projects concurrently, leading to more effective decision- making and project control.
* **Real-Time Progress Tracking:** The proposed system introduces efficient progress tracking mechanisms, leveraging technology to provide real-time insights into project milestones. This feature enables stakeholders to monitor progress dynamically and make timely adjustments as needed.
* **Intuitive User-Friendly Features:** Addressing the limitation of the current system's complexity, the proposed system emphasizes user-friendly features. This design choice ensures that stakeholders can easily navigate and utilize the system, promoting a more intuitive and user-friendly experience.
* **Accurate Budget Calculations:** Through improved functionalities, the proposed system

enhances the accuracy of budget calculations. Real-time data and advanced algorithms contribute to precise financial tracking, reducing the likelihood of errors in budget estimates.

* **Comprehensive Data Analytics:** The proposed system introduces a robust analytics component, enabling stakeholders to derive meaningful insights from project data. This feature supports data-driven decision-making, enhancing project management strategies and overall optimization.
* **Heightened Security Measures:** Recognizing the importance of data security, the proposed system incorporates heightened security measures. This ensures the confidentiality and integrity of project data, safeguarding against unauthorized access and potential breaches.

# CHAPTER 3 REQUIREMENT ANALYSIS

## FEASIBILITY STUDY

Feasibility is the degree to which a project can be carried out successfully. A feasibility study is conducted to assess the solution's viability, which establishes whether it is viable and implementable in the program. The feasibility study considers details like the availability of resources, software development costs, the advantages of the software to the business once it is built, and the costs associated with maintaining it. The outcome of the feasibility study should be a report recommending whether the requirements engineering, and system development process should be continued. A system is of no real value to a corporation if it does not serve its goals. Even though this may seem obvious, many organizations create systems that do not support their goals, either because they lack a clear statement of these goals, because they fail to specify the system's business requirements, or because other organizational or political factors have an impact on the procurement of the system.

### Economic Feasibility

The economic feasibility of the proposed construction project management system is underscored by its potential to yield substantial cost savings and efficiency gains. By addressing communication gaps, streamlining coordination, and centralizing project oversight, the system aims to reduce the likelihood of costly errors and delays. The real-time progress tracking and accurate budget calculations further contribute to financial prudence, ensuring that projects adhere closely to financial plans. Additionally, the emphasis on user-friendly features promotes quicker adaptation, minimizing training costs for stakeholders. The comprehensive data analytics component enhances decision-making, potentially leading to optimized resource allocation and improved project outcomes. Overall, the proposed system demonstrates strong economic feasibility through its capacity to enhance operational efficiency, reduce risks, and contribute to more cost-effective construction project management.

### Technical Feasibility

The Technical feasibility of the proposed construction project management system is evident in its adept integration of advanced technologies and a robust software architecture. Structured with HTML, CSS, and JavaScript for the frontend, and Python Django for the backend, the system ensures a stable, responsive, and scalable platform. The incorporation of augmented reality, chatbots, and a report generator reflects a commitment to cutting-edge technical solutions, enhancing user experience and project oversight. The system's ability to handle user registration, authentication, and e-commerce transactions demonstrates its technical prowess, while the emphasis on security measures underscores a commitment to safeguarding sensitive project data.

Overall, the proposed system exhibits strong technical feasibility, positioning itself as a technologically advanced solution for effective construction project management.

### Behavioral Feasibility

The behavioral feasibility of the proposed construction project management system is rooted in its user-centric design and emphasis on user-friendly features. By introducing distinct modules tailored to the roles of Project Admin, Engineers, Contractors, Clients, and Purchase Managers, the system aims to accommodate the diverse needs and preferences of its users. The intuitive interfaces, real-time progress tracking, and streamlined coordination contribute to a positive user experience, potentially reducing resistance to system adoption. The incorporation of advanced technologies like augmented reality and chatbots aligns with modern user expectations, enhancing engagement and collaboration. Overall, the behavioral feasibility of the proposed system is evident in its efforts to promote user acceptance, satisfaction, and effective utilization within the construction project management.

### Feasibility Study Questionnaire

Here's a questionnaire containing 11 questions to collect details about the "Construction Project Management System" project:

1. What are the common challenges you face in construction project management?

Answer: Common challenges include communication gaps, project delays, and resource allocation issues.

1. How do you currently manage project documentation, such as building permits and sanctions? Answer: We usually handle these documents manually or through email.
2. What difficulties do you encounter in managing labor and worker schedules on construction sites?

Answer: It can be challenging to track worker attendance and assign tasks efficiently.

1. How do you currently handle the procurement of raw materials and inventory management? \ Answer: We often rely on reports and manual tracking for inventory control.
2. What features or tools would you like to see in a construction project management system to improve your project oversight?

Answer: Real-time progress tracking, automated reporting, and effective communication tools would be valuable.

1. What challenges do you anticipate in adopting a digital project management system within your organization?

Answer: Training staff, data migration, and ensuring data security are potential challenges.

1. How can the system ensure secure and efficient financial transactions related to project expenses and material procurement?

Answer: Secure payment gateways and detailed financial reporting would be essential.

1. What additional functionalities would enhance your experience as a construction professional or contractor using this system?

Answer: Integration with scheduling tools, task reminders, and subcontractor management features would be beneficial.

1. How can the system contribute to improved client satisfaction and collaboration?

Answer: Providing clients with real-time project updates, milestone tracking, and transparent budget management would enhance collaboration.

1. In what ways do you envision the system assisting in project risk management and issue resolution?

Answer: Early issue detection, risk assessment tools, and a centralized issue tracking system would be valuable for risk management.

1. How can the system facilitate better communication and coordination among project stakeholders?

Answer: Features like in-app messaging, document sharing, and collaborative project dashboards would improve communication.

## 3.1 SYSTEM SPECIFICATION

* + 1. **Hardware Specification** Processor - Intel Core i3 RAM - 8 G B

Hard disk - 1 T B

### Software Specification

Front End - HTML5, CSS, Bootstrap

Back End - Django, SQLite

Client on PC - Windows 7 and above.

Database - SQLite

Technologies used - Django, HTML5, Bootstrap, JS, jQuery

## SOFTWARE DESCRIPTION

### DJANGO

Django, a leading open-source web framework, epitomizes the pinnacle of modern web development with its efficiency and versatility. Built on Python, Django follows the Model-View- Controller architecture, prioritizing simplicity and flexibility. Noteworthy features include its Object-Relational Mapping system, streamlined URL routing, and a user-friendly template engine. The built-in admin interface simplifies data management, while robust security measures and middleware support enhance application integrity. Django scales effortlessly, accommodating growing datasets and traffic demands. Its REST Framework extension further extends its utility to API development. Supported by an active community and extensive documentation, Django stands as a powerful toolkit, seamlessly shaping the development of dynamic and secure web applications for diverse purposes, from content management systems to Restful API. In essence, Django empowers developers with a comprehensive and adaptable framework for crafting sophisticated and salable web solutions**.**

### SQLite

SQLite, a lightweight and server less relational database management system, is celebrated for its simplicity, portability, and efficiency. Operating as a self-contained, single-file database, SQLite requires minimal setup and eliminates the need for a separate server process. This design makes it a preferred choice for embedded systems, mobile applications, and small to medium-scale projects. With zero configuration and a server less architecture, SQLite streamlines the database management process. It supports ACID transactions, ensuring data integrity and reliability, even in the face of system interruptions. The dynamic typing feature allows flexible data storage, accommodating various data types within the same column.

Noteworthy is SQLite's cross-platform compatibility, making it versatile across different operating systems and environments. Its wide adoption is evident in applications ranging from mobile apps to web browsers, where it is utilized for local data storage. SQLite stands out as a go- to solution for projects requiring a lightweight, self-contained, and easily deployable database system, embodying simplicity without compromising on functionality and reliability.

# CHAPTER 4 SYSTEM DESIGN

## INTRODUCTION

Any designed system or product's development process begins with the design phase. An efficient system depends on a well-executed design, which is a creative process. It entails utilizing a variety of techniques and concepts to completely specify a procedure or system for it to be implemented. Regardless of the development paradigm used in software engineering, the design phase is essential. It seeks to produce the architectural detail needed to design a system or product and acts as the process's technical backbone. To maximize every aspect of efficiency, performance, and accuracy, this program underwent a comprehensive design phase. A user-oriented document is converted into a document for programmers or database workers throughout the design phase.

## UML DIAGRAMS

Software engineering uses the Unified Modeling Language (UML), a standardized visual language, to model, develop, and document software systems. UML diagrams provide a concise and organized approach to represent different facets of a software system, serving as a common communication tool for developers, stakeholders, and designers. There are many different varieties of UML diagrams, including class diagrams, sequence diagrams, use case diagrams, and more. Each is designed to communicate a particular piece of knowledge about the design, operation, and interactions of the system. UML diagrams are essential to the software development process because they help with the visualization, analysis, and design of complex systems, which in turn makes the process more effective and efficient.

* + - Use case diagram
    - Sequence diagram
    - State diagram
    - Activity diagram
    - Class diagram
    - Object diagram
    - Component diagram
    - Deployment diagram

## USE CASE DIAGRAM

The Use Case Diagram serves as a pivotal element in the design phase, offering a visual representation of the intricate interactions taking place among various components within the system. Utilizing the Unified Modeling Language (UML), this diagram serves as a blueprint that illustrates how different actors engage with the system to accomplish specific objectives. In our scenario, the term "actors" encompasses entities such as students, teachers, and administrators, each assigned distinct roles or "use cases" within the system.

### Elements of the Use Case Diagram:

* + - * **Boundary**: Defines the system's scope in relation to its external environment.
      * **Actors**: Represent individuals or roles interacting with the system.
      * **Use Cases**: Depict specific roles played by actors within and around the system.
      * **Relationships**: Illustrate connections between actors and use cases.



Fig1. Use Case Diagram

## SEQUENCE DIAGRAM

The Sequence Diagram, a vital tool in software design, visually captures dynamic interactions among system elements over time. In the Unified Modeling Language (UML), it depicts collaborative exchanges among components, showcasing how they work together for specific tasks. Objects are "lifelines," with vertical lines indicating their existence. Arrows show the sequential flow of calls or messages, revealing the sequence of operations for each object in a use case. Sequence Diagrams unravel system behaviors, offering a detailed view of how components engage. In our project, they serve as indispensable guides, providing real-time insights into interactions for a precise understanding of system dynamics.

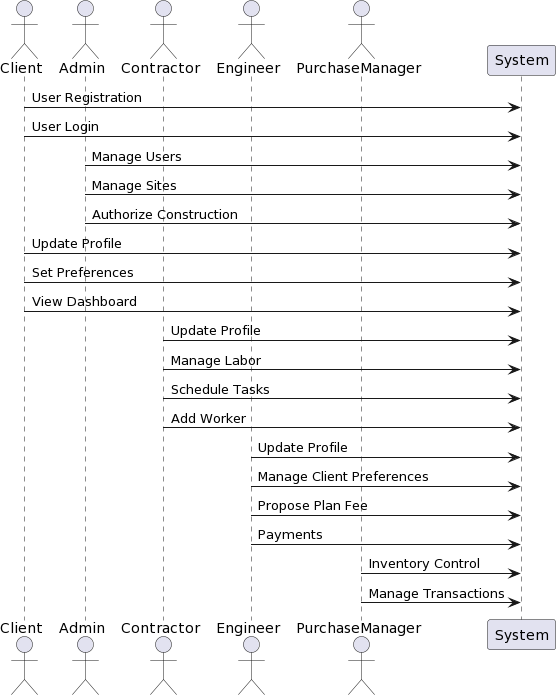


Fig2. Sequence Diagram

## State Chart Diagram

The State Chart Diagram, a distinctive facet of the Unified Modeling Language (UML), serves as an insightful illustration delineating the various states an object can traverse and the transitions between these states. Within the context of our software engineering project, an object signifies a specific entity within the program, embodying the fundamental unit of code representing that entity.

In essence, a state in this diagram encapsulates a distinct stage in the evolutionary or behavioral journey of an object. Unlike a flowchart, which elucidates the processes altering the state of an object within a system, a state diagram meticulously articulates the state changes themselves. To construct a state chart diagram, the initial and final states of the system are first identified, and subsequently, all conceivable intermediate states are strategically positioned in relation to the commencement and conclusion. The diagram is then enriched by incorporating labeled transition elements, articulating the events that act as catalysts for state alterations.

This visual representation becomes an invaluable tool for comprehending the dynamic states an object undergoes and the seamless transitions between these states. By capturing the essence of object evolution, the State Chart Diagram aids in fostering a profound understanding of the intricacies embedded within the operational fabric of our software system.

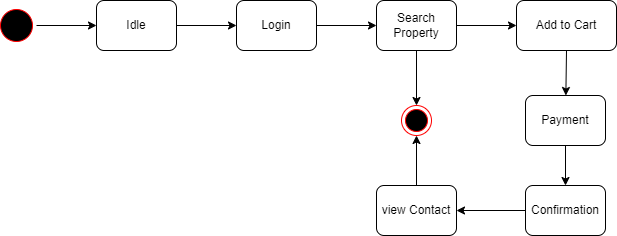


Fig3. State Chart Diagram

## Activity Diagram

The Activity Diagram, a dynamic portrayal of system behavior in UML, serves as an advanced flowchart depicting the sequential progression of operations. In our project, where intricate system activities demand clarity, this diagram becomes instrumental. It showcases how one operation triggers the next, employing features like fork and join for various flow control scenarios. Operating as a behavioral diagram, it illustrates the flow of activities, incorporating parallel, concurrent, and branched pathways. This visual tool encapsulates the nuanced dynamics of our system's behavior, providing a comprehensive understanding of the sequential execution of activities from initiation to completion.

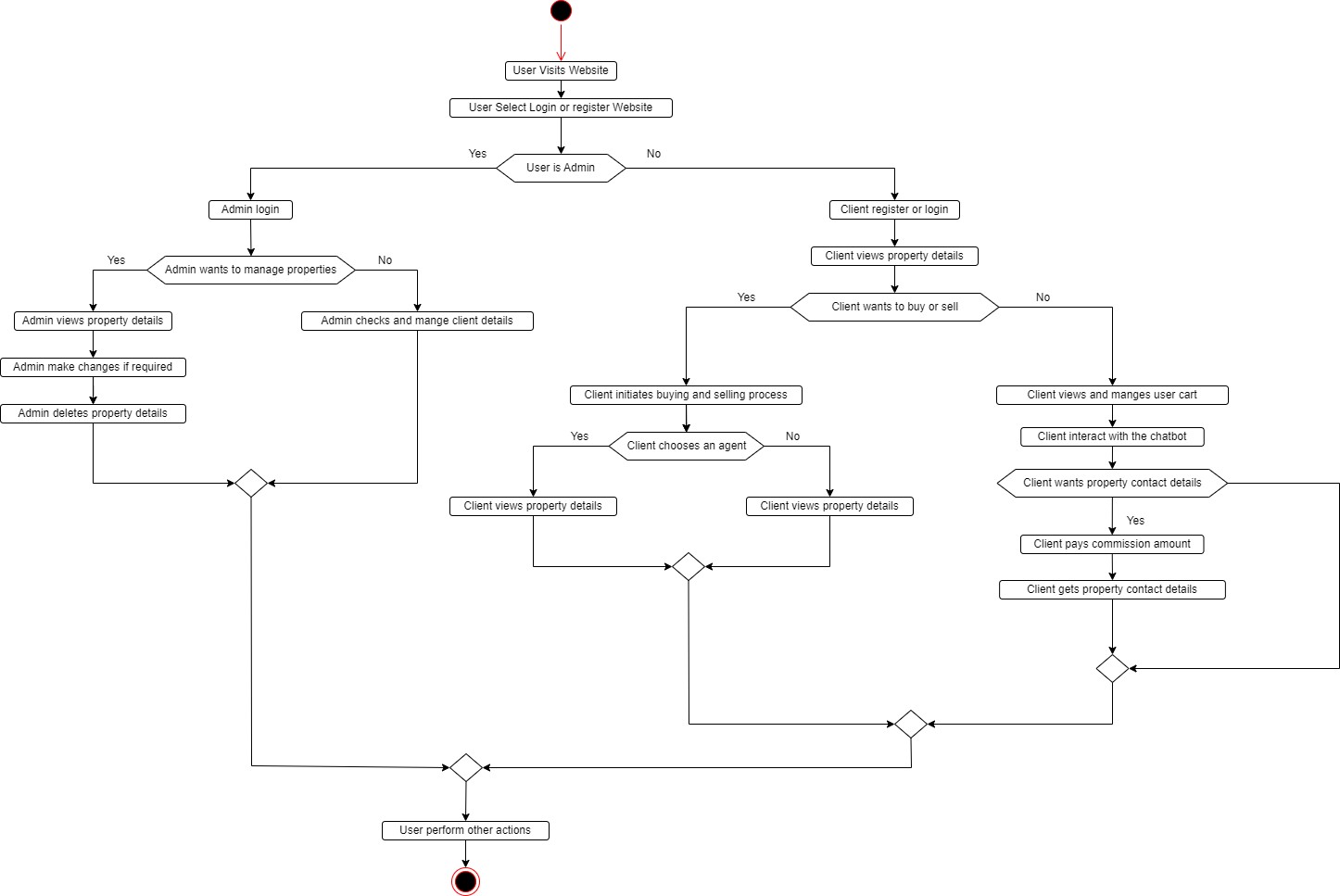


Fig4. Activity Diagram

## Class Diagram

In our software engineering venture, the Class Diagram stands as a cornerstone, serving as the primary architectural blueprint for object-oriented modeling. This diagram not only captures the essence of the application's structural design but also acts as a bridge between conceptual modeling and concrete programming code. Beyond its role in conceptualizing the system's structure, the Class Diagram proves invaluable in data modeling, articulating the relationships between objects and delineating their functionalities and services. As a versatile tool, it aids us in various stages of system design. During the analysis phase, it facilitates a profound understanding of problem domain requirements and component identification. In subsequent stages, particularly in an object-oriented software project, the Class Diagrams evolve from conceptual models to concrete representations, laying the foundation for actual software classes and objects. This dynamic evolution ensures that the diagrams remain relevant snapshots, detailing how our system functions, elucidating relationships at multiple levels, and outlining the roadmap for effective implementation.

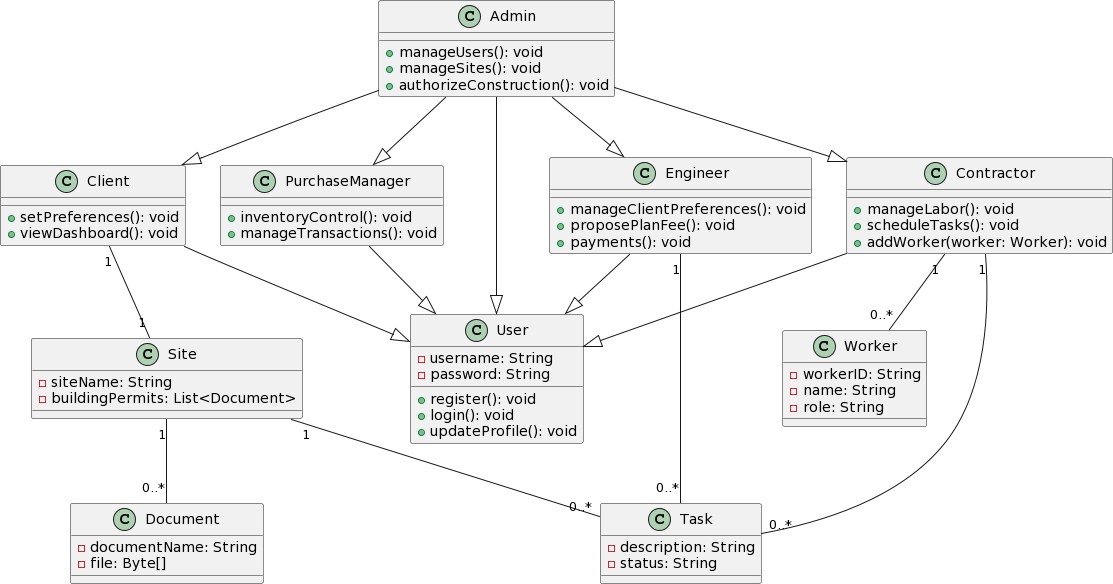


Fig5. Class Diagram

## Object Diagram

In the realm of system modeling, object diagrams are intricately tied to class diagrams, with the latter serving as their foundation. Considered a prerequisite, class diagrams lay the groundwork for the instantiation showcased in an object diagram—an illustrative snapshot capturing the system's essence at a specific moment. Both class and object diagrams seamlessly integrate fundamental ideas, fostering a cohesive modeling approach. While class diagrams offer a timeless, foundational perspective, object diagrams provide a dynamic, instance-driven angle, enabling the representation of the system's static view as a momentary snapshot. The application of object diagrams becomes instrumental in portraying a specific group of interconnected items within a system, contributing to a comprehensive understanding of its architecture.

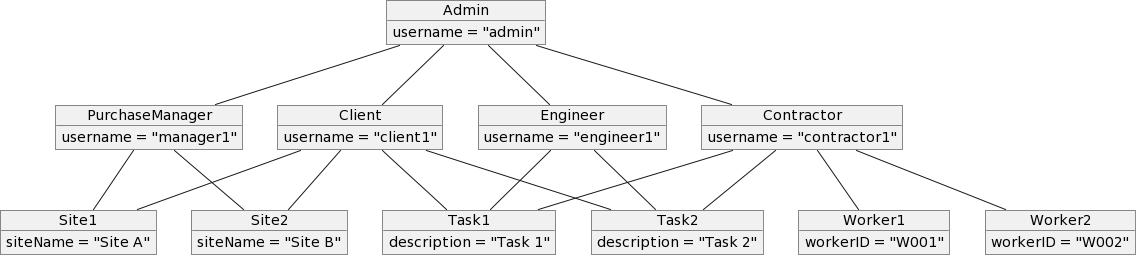


Fig6. Object Diagram

## Component Diagram

UML diagrams that serve as indispensable tools for visualizing and comprehensively documenting object-oriented systems, the component diagram holds a pivotal role. While class diagrams capture the static behavior of a system, component diagrams step in to portray the intricate web of interconnections among components, providing a visual representation of how they intricately come together to forge more extensive components or complete software systems. These diagrams prove instrumental in illustrating the structural composition of systems, shedding light on their inherent complexities. In the intricate dance of real multitasking systems, each component assumes a distinct role, fostering a coordinated symphony where communication is confined to essential components, ensuring efficiency in the overall functioning. The component diagram, therefore, stands as a powerful tool to unravel and articulate the structural intricacies of arbitrarily complex systems, contributing to a holistic understanding of their composition and interactions.

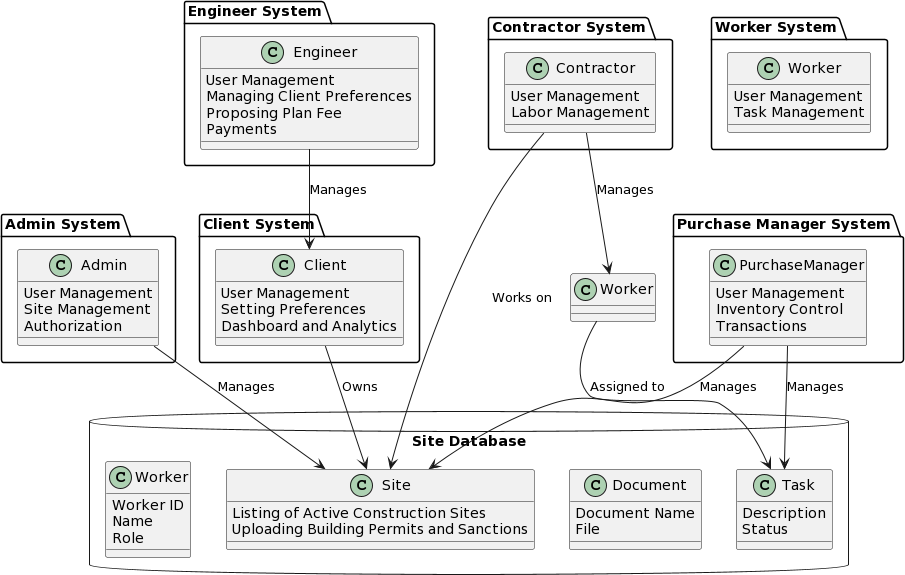


Fig7. Component Diagram

## 4.2.8 Deployment Diagram

Within the realm of UML diagrams that serve as indispensable tools for visualizing and comprehensively documenting object-oriented systems, the component diagram holds a pivotal role. While class diagrams capture the static behavior of a system, component diagrams step in to portray the intricate web of interconnections among components, providing a visual representation of how they intricately come together to forge more extensive components or complete software systems. These diagrams prove instrumental in illustrating the structural composition of systems, shedding light on their inherent complexities. In the intricate dance of real multitasking systems, each component assumes a distinct role, fostering a coordinated symphony where communication is confined to essential components, ensuring efficiency in the overall functioning. The component diagram, therefore, stands as a powerful tool to unravel and articulate the structural intricacies of arbitrarily complex systems, contributing to a holistic understanding of their composition and interactions.

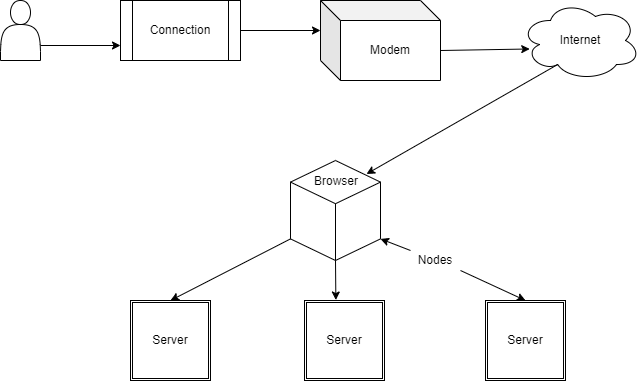
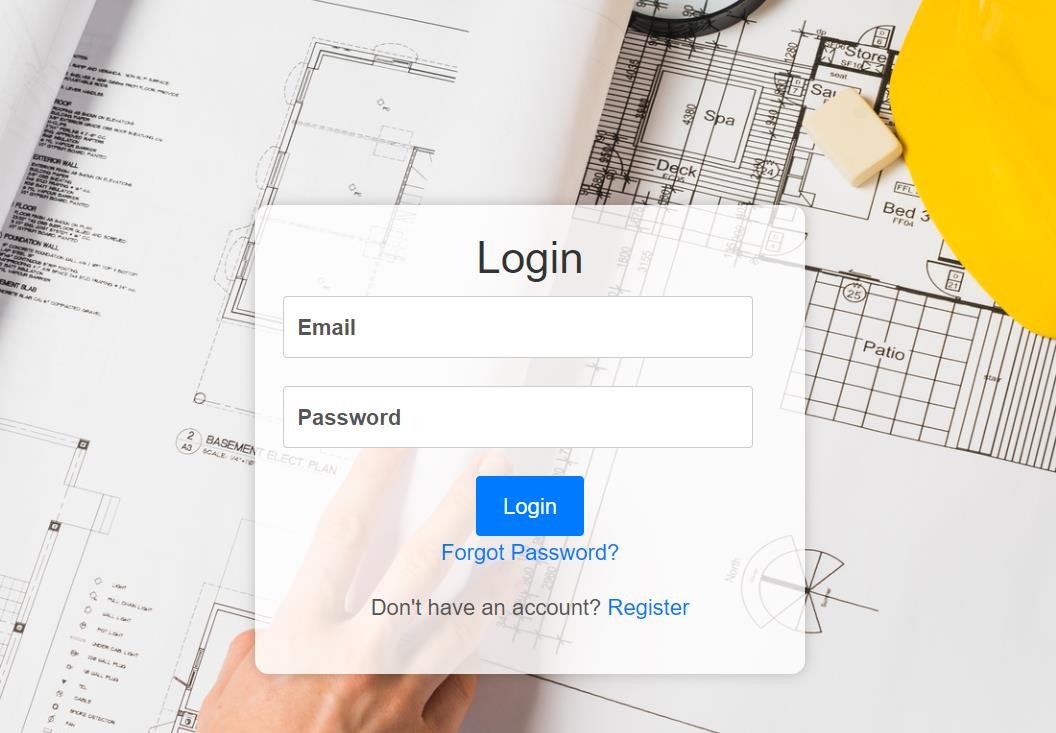
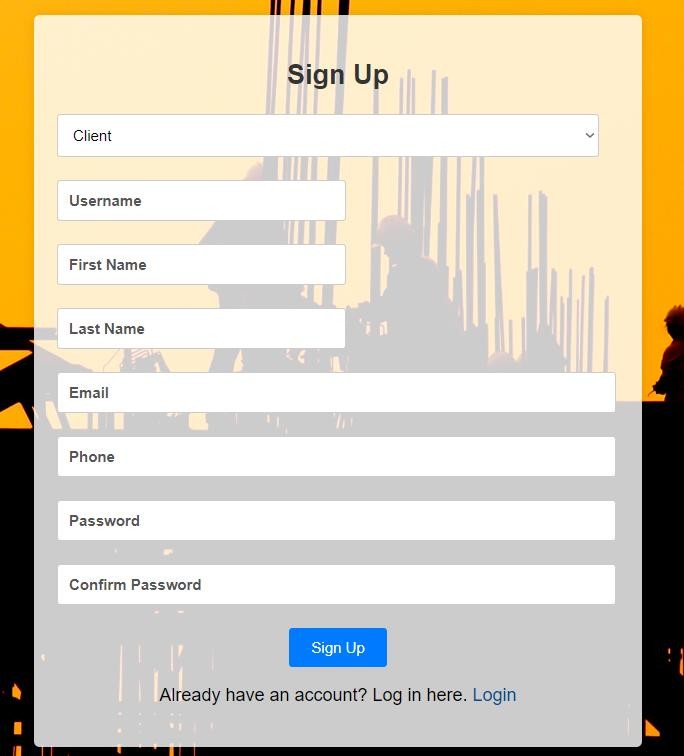


Fig8. Deployment Diagram

## USER INTERFACE DESIGN USING FIGMA







* 1. **DATABASE DESIGN**

A database is a collection of data that has been organized to make it simple to manage and update. Information security could be one of the main aims of any database. The database design process is divided into two steps. The goal of the first step is to gather user requirements to create a database that as clearly as possible satisfies user needs. It is known as information-level design and is carried out without the aid of any DBMS. An information-level design is changed to a specific DBMS design that will be used to develop the system in the following stage. The physical- level design phase is where the characteristics of the specific DBMS are considered.

### RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)

A Relational Database Management System (RDBMS) is a critical software application for organizing and managing data in a structured manner. It stores data in tables with rows and columns, where each row represents a record, and each column is a specific attribute or field. RDBMS systems like MySQL, Oracle, or Microsoft SQL Server ensure data integrity, enforce relationships between tables, and allow for efficient data retrieval and manipulation through Structured Query Language (SQL). These systems are widely used in various applications, such as e-commerce websites, financial systems, and inventory management, due to their robustness, scalability, and ability to handle complex data structures, making them essential for modern data- driven environments.

A Relational Database Management System (RDBMS) is a cornerstone of modern data management. It structures data into tables, where each row represents a unique entry, and each column corresponds to a specific attribute, ensuring a logical and organized storage system. RDBMS systems employ complex algorithms for data retrieval and manipulation, guaranteeing data consistency and adherence to predefined relationships between tables. Popular RDBMS software, including PostgreSQL, MySQL, and Microsoft SQL Server, provides robust features for transactions, data security, and scalability. These systems are integral to a myriad of applications, from healthcare records and customer databases to inventory control and financial platforms, supporting the efficient storage, retrieval, and analysis of vast datasets, making them essential tools in today's data-driven world.

### NORMALIZATION

Normalization is a database design technique used in relational database management systems (RDBMS) to eliminate data redundancy and improve data integrity. It involves organizing data in a way that reduces data duplication and ensures that relationships between tables are defined and maintained.

The primary goals of normalization are:

* + - * **Minimizing Data Redundancy**: By breaking down data into separate tables and ensuring that each piece of data is stored only once, normalization helps reduce the chances of data inconsistencies or errors.
      * **Ensuring Data Integrity:** Normalization enforces the rules of referential integrity, which means that relationships between tables are well-defined and maintained. This ensures that data remains accurate and consistent.

Normalization typically involves dividing a database into multiple related tables and using primary keys and foreign keys to establish relationships between these tables. There are several normal forms, from First Normal Form (1NF) to Fifth Normal Form (5NF), each with specific rules and requirements. The level of normalization achieved depends on the specific needs of the database and the trade-off between data redundancy and query performance.By applying normalization principles, designers can create efficient and reliable database structures that support data integrity and ease data maintenance and manipulation.

There are several normal forms (NF) that define specific rules and requirements for achieving progressively higher levels of normalization. Here are the most common normal forms, from First Normal Form (1NF) to Fifth Normal Form (5NF):

First Normal Form (1NF):

* + - * Each table has a primary key.
      * All columns contain atomic (indivisible) values.
      * There are no repeating groups or arrays in columns. Second Normal Form (2NF):
      * The table is in 1NF.
      * All non-key attributes are fully functionally dependent on the entire primary key. In other words, all non-key attributes must be dependent on the entire primary key, not just part of it.

Third Normal Form (3NF):

* + - * The table is in 2NF.
      * There is no transitive dependency, meaning that non-key attributes are not dependent on other non-key attributes.

Boyce-Codd Normal Form (BCNF):

* + - * A stricter version of 3NF.
      * It enforces that every non-trivial functional dependency involves a super key. Fourth Normal Form (4NF):
      * Addresses multi-valued dependencies.
      * Eliminates any multi-valued dependencies within the data. Fifth Normal Form (5NF):
      * Also known as Project-Join Normal Form (PJ/NF).
      * Handles cases where data can be derived by joining multiple tables.

### SANITIZATION

In Python Django, sanitization refers to the process of cleaning and validating data to ensure that it is safe and free from malicious content before it is used or stored in a database. Sanitization is a crucial security practice to prevent various forms of attacks, such as cross-site scripting (XSS) and SQL injection, which can compromise the security and integrity of a web application.

### INDEXING

Indexing in Django is a fundamental database optimization technique. It involves creating data structures, or indexes, on specific fields to expedite data retrieval. These indexes act as signposts that enable the database to swiftly locate and fetch relevant data, especially in tables with substantial amounts of information. Django offers automatic index creation for primary keys, unique fields, and foreign keys. Additionally, developers can define custom indexes for fields frequently used in filtering, sorting, or searching. The choice of proper indexing plays a pivotal role in improving query performance, resulting in more responsive and scalable web applications. It's essential to consider application-specific query patterns and create indexes accordingly, as well as to understand the capabilities and limitations of the chosen database backend. Effective indexing is a cornerstone of efficient database operations in Django, contributing to enhanced application performance.

* 1. **TABLE DESIGN**

## Registration Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Field name** | **Datatype (Size)** | **Key Constraints** | **Description of the field** |
| 1 | UserId | Int | PK | Id for Registration |
| 2 | UserType | String | NN | Type of User for eg: Client, Contractor |
| 3 | Username | Unique | NN | Name of user |
| 4 | Password | Password | NN | Users Password |
| 5 | FirstName | String | NN | User First Name |
| 6 | Last Name | String | NN | Users Last name |
| 7 | Email | String | NN | Users Email ID |
| 8 | Phone | Number | NN | Users Phone no |
| 9 | Confirm Password | Password | NN | Confirm Password |

1. **User Login Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Field name** | **Datatype (Size)** | **Key Constraints** | **Description of the field** |
| 1 | User ID | Int | PK | User ID |
| 2 | Email | Unique | FK | Email ID for Login |
| 3 | Password | Password | NN | Login Password |

## Project Data Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Field name** | **Datatype (Size)** | **Key Constraints** | **Description of the field** |
| 1 | Client ID | Int | PK | ID of the Client |
| 2 | No of Floors | Int | NN | No of Floors |
| 3 | Work Area Required | Boolean | NN | Work Area |
| 4 | Store Room Required | Boolean | NN | Store Room |
| ll5 | Dining Room Required | Boolean | NN | Dining Room |
| 6 | Kitchen Type | Text | NN | Kitchen Type |
| 7 | Additional Amenities | Text | NN | Additional Amenities |

1. **Floor Details Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Field name** | **Datatype (Size)** | **Key Constraints** | **Description of the field** |
| 1 | Client ID | Int | PK | ID of the Client |
| 2 | Floor number | Int | NN | Floor Number |
| 3 | No of Bedrooms | Int | NN | No of Bedrooms |
| 4 | No of Bathrooms | Int | NN | No of Bathrooms |
| 5 | Project ID | Int | FK | ID of the Project |

1. **Plot Data Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No:** | **Field name** | **Datatype (Size)** | **Key Constraints** | **Description of the field** |
|  | User ID | Int | FK | ID of Client |
| 1 | Plot Base Data ID | Int | PK | ID of Plot Data |
| 2 | Plot User ID | Int | FK | ID of Plot User |
| 3 | Project Data ID | Int | FK | Project Data ID |
| 4 | Contractor ID | Int | FK | Contractor ID |
| 5 | Engineer ID | Int | FK | Engineer ID |

# CHAPTER 5 SYSTEM TESTING

## INTRODUCTION

Software testing is a way to check if the computer program works like it's supposed to. We use testing to make sure the software does what it is supposed to do. Validation means checking or testing things like software to make sure they meet the requirements and standards they are supposed to follow. Software testing is a way to check if a program works well. It goes along with other methods like checking and walking through the program. Validation means making sure that what the user wanted is what they got. There are several rules that can serve as testing objectives.

They are:

Testing is a process of executing a program with the intent of finding an error.

* A good test case is one that has high possibility of finding an undiscovered error.
* A successful test is one that uncovers an undiscovered error.

If a test works well and follows its goals, it can find mistakes in the software. The test showed that the computer program is working like it's supposed to and is doing well.

There are three ways to test program.

* For correctness
* For implementation efficiency
* For computational complexity

## TEST PLAN

A test plan suggests several required steps that need be taken to complete various testing methodologies. The activity that is to be taken is outlined in the test plan. A computer program, its documentation, and associated data structures are all created by software developers. It is always the responsibility of the software developers to test each of the program's separate components to make sure it fulfills the purpose for which it was intended. To solve the inherent issues with allowing the builder evaluate what they have developed, there is an independent test group (ITG). Testing's precise goals should be laid forth in quantifiable language. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test.

The levels of testing include:

* + - Unit testing
    - Integration Testing
    - Data validation Testing & Output Testing

### Unit Testing

Unit testing concentrates verification efforts on the software component or module, which is the smallest unit of software design. The component level design description is used as a guide when testing crucial control paths to find faults inside the module's perimeter. The level of test complexity and the untested area determined for unit testing. Unit testing is white-box focused, and numerous components may be tested simultaneously. To guarantee that data enters and exits the software unit under test properly, the modular interface is tested. To make sure that data temporarily stored retains its integrity during each step of an algorithm's execution, the local data structure is inspected. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Before starting any other test, tests of data flow over a module interface are necessary. All other tests are irrelevant if data cannot enter and depart the system properly. An important duty during the unit test is the selective examination of execution pathways. Error circumstances must be foreseen in good design, and error handling paths must be put up to cleanly reroute or halt work when an error does arise. The final step of unit testing is boundary testing. Software frequently fails at its limits.

In the Sell-Soft System, unit testing was carried out by treating each module as a distinct entity and subjecting them to a variety of test inputs. The internal logic of the modules had some issues, which were fixed. Each module is tested and run separately after coding. All unused code was eliminated, and it was confirmed that every module was functional and produced the desired outcome.

### Integration Testing

Integration testing is a methodical approach for creating the program's structure while also carrying out tests to find interface issues. The goal is to construct a program structure that has been determined by design using unit tested components. The program is tested. Correction is challenging since the size of the overall program makes it challenging to isolate the causes. As soon as these mistakes are fixed, new ones arise, and the process repeats itself in an apparently unending cycle. All the modules were integrated after unit testing was completed in the system to check for an interface inconsistency. A distinctive program structure also developed when discrepancies in program structures.

### Validation Testing or System Testing

The testing process comes to an end here. This involved testing the entire system in its entirety, including all forms, code, modules, and class modules. Popular names for this type of testing include system tests and black box testing. The functional requirements of the software are the main emphasis of the black box testing approach. That example, using Black Box testing, a software engineer can create sets of input conditions that will fully test every program requirement. The following sorts of problems are targeted by black box testing: erroneous or missing functions, interface errors, data structure or external data access errors, performance errors, initialization errors, and termination errors.

### Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm’s need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required.

This done with respect to the following points:

* Input Screen Designs,
* Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future.

### Automation Testing

Automation Testing is a software testing technique that performs using special automated testing software tools to execute a test case suite. Essentially, it’s a test to double-check that the equipment or software does exactly what it was designed to do. It tests for bugs, defects, and any other issues that can arise with product development. Although some types of testing, such as regression or functional testing can be done manually, there are greater benefits of doing it automatically. Automation testing can be run at any time of the day. It uses scripted sequences to examine the software. It then reports on what has been found, and this information can be compared with earlier test runs. Automation developers generally write in the following programming languages: C#, JavaScript, and Ruby.

### Selenium Testing

Selenium is an open-source automated testing framework used to verify web applications across different browsers and platforms. Selenium allows for the creation of test scripts in various programming languages such as Java, C#, and Python. Jason Huggins, an engineer at Thought Works, developed Selenium in 2004 while working on a web application that required frequent testing. He created a JavaScript program called "JavaScriptTestRunner" to automate browser actions and improve testing efficiency. Selenium has since evolved and continues to be developed by a team of contributors. In addition to Selenium, another popular tool used for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior- driven development (BDD). It allows for the creation of executable specifications in a human- readable format called Gherkin. One of the advantages of using Cucumber is its ability to bridge the gap between business stakeholders and technical teams. By using a common language, Cucumber facilitates effective communication and collaboration during the testing process. It promotes a shared understanding of the requirements and helps ensure that the developed software meets the intended business goals. Cucumber can be integrated with Selenium to combine the benefits of both tools. Selenium is used for interacting with web browsers and automating browser actions, while Cucumber provides a structured framework for organizing and executing tests. This combination allows for the creation of end-to-end tests that verify the behavior of web applications across different browsers and platforms, using a business-readable and maintainable format.

### Test Case 1: User - Login Code

package testdefinitions;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import io.cucumber.java.en.And;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

**import static** org.junit.Assert.*assertTrue*;

**public class** loginsteps {

WebDriver driver = **new** EdgeDriver(); @Given("User is on the login page")

**public void** userIsOnLoginPage() { driver.get("http://127.0.0.1:8000/accounts/login/");

}

@When("User enters valid email and password")

**public void** userEntersValidCredentials() { driver.findElement(By.*id*("email")).sendKeys(["r](mailto:raoofvenad123@gmail.com)a[oofvenad123@gmail.com](mailto:raoofvenad123@gmail.com)"); driver.findElement(By.*id*("password")).sendKeys("Raoof@123"); driver.findElement(By.*id*("login-btn")).click();

}

@Then("User should be redirected to the home page")

**public void** userIsRedirectedToHomePage() {

// Pause for a moment to ensure the redirect completes (adjust as needed)

**try** {

Thread.*sleep*(2000);

} **catch** (InterruptedException e) { e.printStackTrace();

}

// Get the current URL after successful login String currentUrl = driver.getCurrentUrl();

// Verify that the current URL contains the expected part of the URL

*assertTrue*(currentUrl.contains("http://127.0.0.1:8000/client/"));

driver.quit();

}

**public static** WebDriver getDriver() {

// **TODO** Auto-generated method stub

**return null**;

}

}

### Screenshot

**Test Report**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case 1** | | | | | |
| **Project Name:** BuildNest | | | | | |
| **Login Test Case** | | | | | |
| **Test Case ID: 1** | | | **Test Designed By:** Amal Jose Kurian | | |
| **Test Priority (Low/Medium/High):** High | | | **Test Designed Date: 02-12-2023** | | |
| **Module Name**: Login Screen | | | **Test Executed By:** Mr. Rony Tom | | |
| **Test Title:** User Login | | | **Test Execution Date:** 05-12-2023 | | |
| **Description:** Verify login with valid email and password | | |  | | |
| **Pre-Condition: User** has valid username and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected**  **Result** | **Actual**  **Result** | **Status (Pass/**  **Fail)** |
| 1 | Navigation to login page. |  | Login Page should be displayed | Login Page be displayed | Pass |
| 2 | Provide valid Email | Email: amalj5634@g  mail.com | User should be able to login | User is logged in and navigated to corresponding Home Page | Pass |
| 3 | Provide Valid  Password | Password |
| 4 | Click on  Login button |  |
| **Post-Condition:** User is validated with database and successfully login into account. The Account session details are logged in database | | | | | |

**Test Case 2: Update Profile**

**Code:**

package testdefinitions;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import io.cucumber.java.en.And;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

public class update {

WebDriver driver = CommonDriver.getDriver();

@Given("user is on the settings page")

public void user\_is\_on\_the\_settings\_page() {

driver.get("http://127.0.0.1:8000/userprofile");

}

@When("user enters a new full name {string}")

public void user\_enters\_a\_new\_full\_name(String string) {

WebElement fullname\_input = driver.findElement(By.id("new\_first\_name"));

fullname\_input.clear();

fullname\_input.sendKeys("Abcd");

}

@And("user submits the update form")

public void user\_submits\_the\_update\_form() {

WebElement submitButton = driver.findElement(By.id("submitt"));

submitButton.click();

}

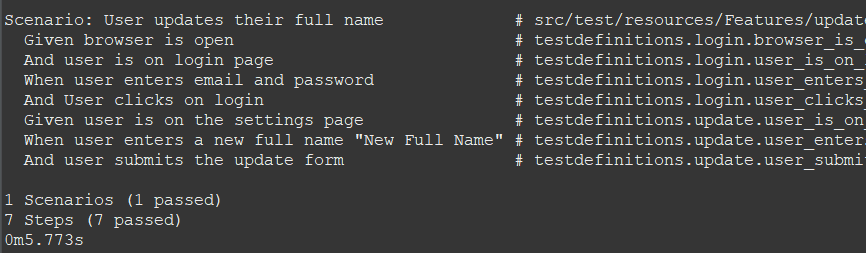
@Then("the full name should be updated to {string}")

public void the\_full\_name\_should\_be\_updated\_to(String string) {

}

}

**Screenshot**



**Test report**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case 2** | | | | | |
| **Project Name:BuildNest** | | | | | |
| **Update Profile** | | | | | |
| **Test Case ID: 2** | | | **Test Designed By:** Amal Jose Kurian | | |
| **Test Priority (Low/Medium/High):** High | | | **Test Designed Date:** 04-12-2023 | | |
| **Module Name**: User profile | | | **Test Executed By:** Mr. Rony Tom | | |
| **Test Title:** Update | | | **Test Execution Date:** 05-12-2023 | | |
| **Description:** User updates their first name | | |  | | |
| **Pre-Condition: User** has valid email and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/Fail)** |
| 1 | Navigation to  Login Page |  | Home should be  displayed | Login page  displayed | Pass |
| 2 | Provide  Valid  email | Email: amalj5634@gmail.com | User should be able to  Login | User Logged in  and navigated to  Player Home | Pass |
| 3 | Provide  Valid  Password | Password: Amal@123 |
| 5 | Click on profile | Button click | User should able to update the firstname | User firstname changed | Pass |
| 6 | View profile and edit firstname | Newfirstname:  ” ABCD” |
| **Post-Condition:** After logging in, the user accesses the home page, then in the profile section user updates their firstname. | | | | | |

**Test Case 3: Deactivating User**

**Code:**

package testdefinitions;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

public class function3 {

WebDriver driver = CommonDriver.getDriver();

@Given("user is on the login page")

public void user\_is\_on\_the\_login\_page() {

driver.get("http://127.0.0.1:8000/user\_login");

}

@When("admin enters email and password")

public void admin\_enters\_email\_and\_password() {

WebElement emailInput = driver.findElement(By.id("email"));

WebElement passwordInput = driver.findElement(By.id("password"));

emailInput.clear();

emailInput.sendKeys("sportigoplayspot@gmail.com");

passwordInput.clear();

passwordInput.sendKeys("sportigo");

}

@Then("user should be logged in")

public void user\_should\_be\_logged\_in() {

WebElement usersDiv = driver.findElement(By.id("userss"));

usersDiv.click();

WebElement normalDiv = driver.findElement(By.id("normalDiv"));

normalDiv.click();

}

@When("user deactivates a normal user")

public void user\_deactivates\_a\_normal\_user() {

WebElement suspendCheck = driver.findElement(By.id("suspendCheck"));

suspendCheck.click();

WebElement statusUpd = driver.findElement(By.id("statusUpd"));

statusUpd.click();

}

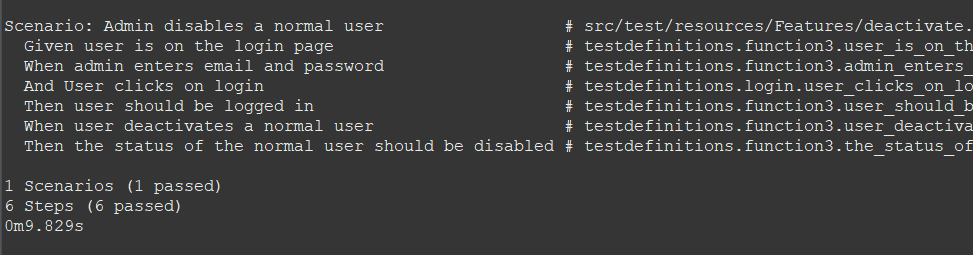
@Then("the status of the normal user should be disabled")

public void the\_status\_of\_the\_normal\_user\_should\_be\_disabled() {

}

}

**Screenshot**



**Test report**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case 3** | | | | | |
| **Project Name: BuildNest** | | | | | |
| **Admin deactivating a user** | | | | | |
| **Test Case ID: 3** | | | **Test Designed By:**Amal Jose Kurian | | |
| **Test Priority (Low/Medium/High):** High | | | **Test Designed Date:**04-12-2023 | | |
| **Module Name**: admin home | | | **Test Executed By:** Mr. Rony Tom | | |
| **Test Title:**Deactivate a User | | | **Test Execution Date:**05-12-2023 | | |
| **Description:**Admin deactivates a user | | |  | | |
| **Pre-Condition: Admin** has valid email and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/Fail)** |
| 1 | Navigation to  Login Page |  | Home should be  displayed | Login page  displayed | Pass |
| 2 | Provide  Valid  email | Email:  buildnest97@gmail.com | Admin should be able to  Login | Admin Logged in  and navigated to  Admin Home | Pass |
| 3 | Provide  Valid  Password | Password: buildnest |
| 5 | Click on manage users |  | Admin should able to see every user and can deactivate user | Admin | Pass |
| 6 | Click on client user | User email: amalj5634@gmail.com |
| 7 | Deactivate a user |  | User should be deactivated | The user is deactivated | Pass |
| **Post-Condition:**Admin login to their home page, see the list of users from that deactivating a user from normal user category | | | | | |

# CHAPTER 6 IMPLEMENTATION

## INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion. Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover. The implementation state involves the following tasks:

* Careful planning.
* Investigation of system and constraints.
* Design of methods to achieve the changeover.

## IMPLEMENTATION PROCEDURES

The implementation procedures for this project involve a systematic and phased approach to bring the online crime reporting portal to fruition. This complex system, designed to cater to the needs of various stakeholders, requires careful planning and execution.

To commence the implementation, the project team will initiate a detailed requirements analysis phase. This stage involves comprehensive discussions with law enforcement officials, control room staff, prison wardens, and potential end-users to determine their specific needs and expectations. The development phase will follow, involving the creation of the portal's architecture, databases, and user interfaces. It's during this stage that the functionalities outlined in the project's scope, including user management, crime reporting, and communication features, will be integrated into the system. The portal will be designed with a user-friendly interface to ensure ease of use for all categories of users.

Simultaneously, a dedicated team will work on the incorporation of advanced technologies, such as machine learning capabilities for crime trend analysis and predictive features. These technologies will be implemented carefully to ensure the portal's robustness and security.

Once the development phase is complete, rigorous testing will be conducted to identify and rectify any bugs or issues. The portal will undergo extensive quality assurance and user acceptance testing to ensure that it meets all the necessary requirements and is free of vulnerabilities.

Deployment of the system will be carried out in a controlled manner, ensuring minimal disruption to the existing processes, and allowing for gradual adaptation by the involved stakeholders. Comprehensive training sessions will be conducted to familiarize law enforcement personnel, control room staff, prison wardens, and registered users with the portal's functionality and features. Post-deployment, the project team will continue to provide support and maintenance, addressing any issues that may arise and making necessary improvements as the system matures. Regular updates and security measures will be implemented to safeguard user data and maintain the portal's efficiency.

### User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database, and call up routine that will produce reports and perform other necessary functions

### Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or

part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

### System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than “Finding Mistakes”.

# CHAPTER 7 CONCLUSION AND FUTURE SCOPE

## CONCLUSION

In conclusion, the proposed construction project management system presents a comprehensive and innovative solution to address the limitations of the existing natural system. With its structured modules catering to specific stakeholders, emphasis on enhanced communication, streamlined coordination, and centralized project oversight, the system stands poised to revolutionize project management in the construction sector. The integration of advanced technologies, user-friendly features, and robust security measures further underscores its potential to optimize operational efficiency. The economic feasibility, technical prowess, and behavioral considerations woven into the system design collectively position it as a forward- thinking and adaptable solution. As the construction industry continues to evolve, this proposed system not only addresses current challenges but also lays a foundation for efficient, collaborative, and technologically advanced project management practices.

## FUTURE SCOPE

The proposed construction project management system opens doors to a promising future scope with avenues for continual enhancement and adaptation to evolving industry needs. Future developments could involve the integration of artificial intelligence for predictive analytics, enabling more accurate project forecasting and risk management. The system could evolve to incorporate machine learning algorithms for automated decision-making, further streamlining project oversight and coordination. Additionally, expanding the application to support mobile platforms could enhance accessibility for stakeholders in the field. As the construction industry embraces emerging technologies, there is potential for the inclusion of Internet of Things (IoT) devices to monitor real-time construction site data, providing a more comprehensive and dynamic view of project progress. The project's future scope also extends to continuous updates to stay abreast of technological advancements and industry best practices, ensuring its sustained relevance and effectiveness in the ever-evolving landscape of construction project management.

.

# CHAPTER 8 BIBLIOGRAPHY

**Amal Jyothi College of Engineering, Kanjirappally Department of Computer Applications**

### REFERENCES:

* https:/[/www.99a](http://www.99acres.com/articles/step-by-step-guide-for-building-construction.html)c[res.com/articles/step-by-step-guide-for-building-construction.html](http://www.99acres.com/articles/step-by-step-guide-for-building-construction.html)
* <https://stonemarkcm.com/blog/a-detailed-guide-to-construction-planning/>
* <https://www.w3schools.com/django/django_update_model.php>
* Greenfeld, Daniel Roy, and Audrey Roy Greenfeld. Two Scoops of Django: Best Practices for Django 1.11. Two Scoops Press, 2017.
* Vincent, William S. Django for Beginners. William S. Vincent, 2018.
* Django Software Foundation. Django Documentation. [Online] Available at: <https://docs.djangoproject.com/>

# CHAPTER 9 APPENDIX

## 9.1 Sample Code

## 9.1 Screen Shots

