Project Report
on
TIME TRACKER
at
Grownited Private Limited

Submitted in partial fulfillment of the requirement
for the award of the degree of
Bachelor of Technology
in
COMPUTER ENGINEERING
by

Prajapati Ansh (21012011113) Prajapati Yashkumar (21012011125)

Under the Guidance of

Internal Guide:
Prof. Kavindra Patel

External Guide:
Rahul Kirpekar

Semester VIII

Ganpat University
॥ विद्यया समाजोत्कर्षः ॥
U.V. Patel College of Engineering
National Assessment and Accreditation Council NAAC A Grade Accredited University

Submitted to
Department of Computer Engineering
Faculty of Engineering & Technology
Ganpat University, Ganpat Vidyanagar-384012,Gujarat
May 2025

U.V. PATEL COLLEGE OF ENGINEERING

Ganpat Vidyanagar-384012, Mehsana-Gandhinagar Highway,
District-Mehsana, Gujarat, INDIA

CERTIFICATE

This is to certify that **Mr. Prajapati Ansh Hasmukhbhai** student of **B.Tech. Semester VIII (Computer Engineering)** has completed his full semester project work titled "**Time Tracker**" satisfactorily in partial fulfillment of the requirement of Bachelor of Technology degree of Computer Engineering of Ganpat University, Ganpat Vidyanagar in the year 2024-2025.

Prof. Kavindra Patel
Internal Guide

Dr. Paresh Sholanki
Head of the Department
Computer Engineering

Stamp

Date of Examination: 10/05/2025

U.V. PATEL COLLEGE OF ENGINEERING

Ganpat Vidyanagar-384012, Mehsana-Gandhinagar Highway,
District-Mehsana, Gujarat, INDIA

CERTIFICATE

This is to certify that **Mr. Prajapati Yashkumar Dasharathbhai** student of **B.Tech. Semester VIII (Computer Engineering)** has completed his full semester project work titled "**Time Tracker**" satisfactorily in partial fulfillment of the requirement of Bachelor of Technology degree of Computer Engineering of Ganpat University, Ganpat Vidyanagar in the year 2024-2025.

Prof. Kavindra Patel
Internal Guide

Dr. Paresh Sholanki
Head of the Department
Computer Engineering

Stamp

Date of Examination: 10/05/2025

GROWNITED

Date: - 21/04/2025

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Ansh Hasmukhbhai Prajapati**, a student of UV Patel College Of Engineering Ganpat University Kherva has successfully completed his/her internship in the field of PYTHON from 6th January 2025 to 6th May 2025 17 Weeks under the guidance of **Rahul Kirpekar**.

His internship activities include Project Understanding & Requirement Analysis, Technical Documentation, Learning New Technologies, Development Activities, Testing & Quality Assurance, Soft Skills & Reporting.

During the internship period, he/she was exposed to various processes and was found to be diligent, hardworking, and inquisitive.

We wish him/her all the best in future endeavors.

For, Grownited Private Limited.

Rahul Kirpekar

(Authorised Signature)

301-305, 3rd Floor, Surbhi Complex, Nr. Municipal Market, Chimanlal Girdharlal Road, Navrangpura,Ahmedabad Gujarat- 380009, 7874014621, hr@grownited.com

GROWNITED

Date: - 19/04/2025

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Yashkumar Dasharathbhai Prajapati**, a student of U.V.Patel College Of Engineering Ganpat University,Kherva has successfully completed his/her internship in the field of PYTHON from 6th January 2025 to 6th May 2025 17 Weeks under the guidance of **Rahul Kirpekar**.

His internship activities include Project Understanding & Requirement Analysis, Technical Documentation, Learning New Technologies, Development Activities, Testing & Quality Assurance, Soft Skills & Reporting.

During the internship period, he/she was exposed to various processes and was found to be diligent, hardworking, and inquisitive.

We wish him/her all the best in future endeavors.

For, Grownited Private Limited.

Rahul Kirpekar

(Authorised Signature)

301-305, 3rd Floor, Surbhi Complex, Nr. Municipal Market, Chimanlal Girdharlal Road, Navrangpura,Ahmedabad Gujarat- 380009, 7874014621, hr@grownited.com

Acknowledgment

I would like to express our heartfelt gratitude to all those who have supported and guided us throughout the successful completion of our project, Time Tracker. First and foremost, I extend our sincere thanks to Prof. Kavindra Patel, our internal guide, for his continuous support, encouragement, and expert guidance at every stage of the project. His insights and feedback have been instrumental in shaping the direction and outcome of our work. I would also like to thank the Department of Information Technology, U.V. Patel College of Engineering, for providing us with the necessary resources, mentorship, and a conducive environment for learning and innovation. A special thanks to our friends and peers who provided valuable input and motivation throughout the development process. Lastly, I are deeply grateful to our families for their patience, encouragement, and unwavering support during the entire project journey. This project has been a rewarding experience that allow me to translate classroom knowledge into a real-world application while enhancing our technical, analytical, and teamwork skills.

Abstract

In today’s fast-paced software development and project management environments, efficient time tracking and task management are essential for maximizing productivity and ensuring timely delivery. The Time Tracker system is a web-based application designed to help teams log work hours, manage tasks, and analyze performance through an intuitive, role-based interface.

This system supports three key user roles: Admin, Project Manager, and Developer. Admins oversee the entire platform, generate comprehensive reports, and manage users. Project Managers can assign tasks, monitor progress, and evaluate team performance. Developers can log time in real-time or manually, track their tasks, and view individual productivity dashboards.

INDEX

LISTS OF FIGURE

7. System design

LIST OF TABLES

1. Introduction

1.1 Project Overview

A Time Tracking App is a crucial tool designed to improve time management, boost productivity, and enhance project tracking. This application helps admins, project managers, and developers monitor task durations, identify productivity trends, and optimize workflow efficiency.

1.2 Background

Time management is essential for software development teams to meet deadlines, allocate resources effectively, and ensure timely project completion. This system streamlines task assignments, records time spent, and generates insightful reports to facilitate better decision-making.

1.3 Purpose

1.3.1 Problem Statement

Managing time and tracking productivity manually can be inefficient and error-prone. A centralized system is needed to automate tracking and reporting.

1.3.2 Project Aim

The aim of this project is to develop an efficient time-tracking system that provides detailed reports on time usage and assists in identifying bottlenecks in the development process.

1.3.3 Project Objectives

- Enable project managers to assign tasks and track progress.
- Allow developers to log their working hours.
- Provide admins with analytical reports and insights.
- Offer visual charts to monitor deadlines and productivity levels.

1.4 Impact, Significance, and Contributions

- Helps teams manage workloads effectively.
- Improves transparency in project timelines.
- Enhances productivity tracking and performance evaluation.

1.5 Organization of Project Report

This document outlines the functional, non-functional requirements, user roles, system design, implementation strategies, and testing methodologies.

2. Project Scope

The proposed time tracking system is designed to streamline project and task management by offering robust time logging, tracking, and reporting capabilities. It aims to serve as a comprehensive tool for improving productivity, transparency, and efficiency within teams. The system will support a variety of stakeholders, including project managers, developers, and administrators, by providing them with role-based access and tailored functionalities.

Key Features and Functional Scope:

• Task Assignment and Monitoring by Project Managers

The system will empower **project managers** with the ability to create, assign, and manage tasks under specific projects. Managers will be able to allocate tasks to individual developers or team members, set deadlines, define priorities, and update task statuses (e.g., To-Do, In Progress, Completed). Real-time dashboards and activity views will enable managers to monitor task progress, track bottlenecks, and ensure alignment with project timelines and objectives. Managers can also reassign or update tasks dynamically based on project requirements.

• Developer Time Logging Functionality

Developers and team members will be able to log their working hours by starting and stopping timers for specific tasks. The system will support both manual and automatic time entry options, enabling users to input time logs retrospectively if needed. Each time log entry will be associated with a project, module, and task, allowing detailed tracking of how work hours are distributed. The interface will be designed for ease of use, enabling developers to efficiently track time with minimal disruption to their workflow.

• Comprehensive Reporting and Analytics

The system will generate detailed and customizable reports based on time logged across projects, tasks, and users. These reports will assist stakeholders in analyzing project progress, individual contributions, and time utilization efficiency. Reports can be exported in multiple formats (PDF, Excel, etc.) and may include features such as graphical representations (pie charts, bar graphs) and filters (by date, task, user, or project). Managers can use these reports for performance evaluation, invoicing, and project budgeting.

• Role-Based Access Control (RBAC)

To ensure security and data privacy, the system will implement role-based access control. Different types of users (e.g., Admin, Project Manager, Developer) will have access to distinct functionalities based on their roles. For example:

- **Admins** will have full control over user management, project settings, and system configuration.
- **Project Managers** will have permissions to create projects, assign tasks, monitor team activities, and generate reports.
- **Developers** will be able to view assigned tasks, log time, update task status, and view their own reports.

Each role will experience a personalized dashboard and feature set tailored to their responsibilities, promoting a more intuitive and secure user experience.

Overall Scope Summary:
This system is intended to bridge the gap between time management and project execution. By offering real-time visibility into work progress and providing analytical insights, it will support better planning, resource allocation, and accountability. Whether for small teams or growing enterprises, this time tracking solution will be scalable, user-centric, and adaptable to various project workflows.

3. Feasibility Analysis

Existing System	Technologies Used	Pros	Cons	Special Features	Ideal For
Clockify	Web, Desktop, Mobile Apps	- Easy to use - Free version available - Rich reporting features	- Limited advanced analytics - Manual time tracking	- Multi-platform access - Detailed time reports	Individuals, Small to Medium Teams
Time Doctor	Cloud-Based, Desktop App	- Advanced productivity monitoring - Distraction alerts	- User interface is complex - Expensive for premium features	- Productivity analytics - Distraction detection - Screenshots and activity tracking	Enterprises, Remote Teams
TimeCamp	Web App, Browser Extensions	- Automated time tracking - Attendance tracking features	- UI can be overwhelming - Limited integrations in free version	- Productivity & attendance tracking - App & website usage monitoring	Freelancers, Contractors
Hubstaff	Web, Desktop, Mobile, GPS	- Real-time tracking - GPS & idle detection - Payroll integration	- Privacy concerns with screenshot features - High learning curve	- GPS tracking - Idle time detection - Payroll and invoicing tools	Field Teams, Remote Businesses
RescueTime	Desktop App (Runs in background)	- Automatic tracking - Focus on productivity	- No manual time logging - Less suitable for team collaboration	- Silent tracking - Focus alerts - Productivity scoring and reports	Individuals, Productivity Seekers
Everhour	Web App, Integrations	- Strong PM integrations - Accurate	- Limited features without	- Asana, Trello, ClickUp	Agencies, Project-based Teams

Existing System	Technologies Used	Pros	Cons	Special Features	Ideal For
	(Trello, Asana, etc.)	invoicing features	integrations - Paid-only plans	integration - Budgeting and invoicing - Team workload insights	

3.1 Feasibility Analysis Table

Feasibility Analysis of Time Tracking System :-

3.2. Technical Feasibility

The technical feasibility is high.
All reviewed systems have been successfully implemented using modern technologies such as:

- Web-based platforms, Desktop apps, and Mobile apps (Clockify, Hubstaff)
- Cloud-based infrastructure for data storage, syncing, and accessibility (Time Doctor, Everhour)
- Features like automatic/manual tracking, GPS, screenshot capturing, and third-party integrations are technically viable using widely available tools and APIs.

This indicates that:

- Required tech stacks (React, Node.js, FastAPI, Firebase, etc.) are mature and well-supported.
- Hosting and scaling on platforms like AWS, Azure, or GCP is feasible.
- Mobile versions can be developed using Flutter or React Native for cross-platform deployment.

3.3. Time Schedule Feasibility

The project timeline is moderately feasible depending on scope.
Based on the features of existing tools, a basic MVP (Minimum Viable Product) with core functionalities (time tracking, reports, login system, task/project linking) can be built within 3–5 months with a small team.
Advanced features (GPS, integrations, billing, analytics) would require an extended timeline (6–12 months).

3.4. Operational Feasibility

Operational feasibility is high, as similar tools are actively used by various user types:

- Individuals & freelancers benefit from simple interfaces (RescueTime, TimeCamp).
- Remote teams & enterprises use tools like Hubstaff and Time Doctor for accountability.
- Agencies & project-based teams use tools like Everhour for PM integration and invoicing.

Your proposed system can operate efficiently if:

- It's user-friendly like Clockify
- Offers features relevant to the target group (e.g., freelancers, remote teams, or agencies)
- Has proper training material and onboarding guides

3.5. Implementation Feasibility

The implementation feasibility is good with careful planning.
Given that open-source libraries and SaaS models are common, implementing this system is achievable:

- Backend can be implemented using FastAPI or Node.js
- Frontend using ReactJS or Vue.js
- Integrations (e.g., calendar, Trello, GPS) via APIs
- Use of cloud platforms ensures scalability and deployment

Security, privacy (especially for screenshot and GPS features), and compliance (e.g., GDPR) should be handled carefully.

3.6. Economic Feasibility

Economic feasibility is positive, depending on scale and monetization.

- Development using open-source technologies can significantly reduce costs.
- A freemium model (like Clockify) is effective to attract users, with premium plans for advanced features (used by Time Doctor, Hubstaff).
- Ongoing costs involve hosting, maintenance, and support staff.
- Potential for revenue through subscriptions, API usage charges, or white-labeled solutions.

4. Software and Hardware requirement

4.1 Functional Requirements

4.1.1 User Management

- The system shall support role-based access control for three types of users: Admin, Project Manager, and Developer. Each role will have specific access rights and capabilities.
- The system shall allow users to register using a secure form capturing essential details (name, email, role, etc.).
- Users shall be able to log in securely using their credentials.
- Users shall be able to manage their profiles, including updating personal information and changing passwords.

4.1.2 Task and Project Management

- The Admin or Project Manager shall be able to create new projects, assigning them a name, description, start/end dates, and modules.
- The Project Manager shall be able to assign tasks to developers, with attributes such as title, description, priority, estimated time, project, and module linkage.
- The system shall allow developers to update task status (e.g., To-do, In Progress, Completed, Pending) as work progresses.
- Deadline management shall include visual indicators and notifications for upcoming or overdue tasks.

4.1.3 Time Tracking

- The system shall support both automatic and manual time logging. Developers can manually log time spent on tasks or use a timer-based approach.
- The system shall calculate the total time spent on each task, which will be displayed in both individual task views and reports.

4.1.4 Reports and Analytics

- The system shall allow Admin and Project Manager to generate project-specific and team-specific reports, exportable in Excel or PDF formats.
- The dashboard shall display interactive charts and graphs to provide insights into task status distribution, total hours logged, and weekly productivity.

4.2 Non-Functional Requirements

4.2.1 Performance Requirements

- The system shall be able to simultaneously support multiple users performing actions like task updates, time tracking, and report generation without performance degradation.

4.2.2 Security Requirements

- The system shall implement secure authentication methods (e.g., bcrypt password hashing).
- Access to system features shall be strictly role-based, preventing unauthorized access to sensitive information or actions.

4.2.3 Usability Requirements

- The UI shall be clean, intuitive, and responsive, ensuring users can navigate easily across different modules (Dashboard, Tasks, Reports, etc.).
- Tooltips, consistent icons, and clear button labels will enhance overall user experience.

4.2.4 Reliability Requirements

- The system must ensure accuracy in time logs, task status, and project data.
- Automatic backups and error handling shall be implemented to prevent data loss in case of unexpected failures.

4.2.5 Maintainability Requirements

- The software shall be designed with a modular and scalable architecture, allowing future enhancements or updates with minimal rework.
- Clear separation of concerns across the frontend, backend, and database layers will promote easier debugging and upgrades.

5. Process Model

The development of the Time Tracker application is based on the Iterative Process Model, which structures the software lifecycle into a series of repeated development cycles (iterations). This approach is ideal for the Time Tracker project as it supports progressive refinement, flexibility to accommodate changes, and continuous feedback integration from stakeholders like developers, project managers, and faculty.

Each iteration of development is treated as a self-contained cycle with the following phases:

5.1. Requirement Analysis

In every cycle, requirements are gathered based on user needs and feedback from the previous iteration. For example:

- In the initial iteration, requirements may focus on basic user authentication and role-based access.
- Later iterations cover task assignment, time logging, and report generation features.

This phase includes:

- Meetings with the guide or user representatives (acting as clients).
- Refining user stories and scenarios for Admin, Manager, and Developer roles.

5.2. Design

Once the requirements for the iteration are finalized, the design phase begins. This includes:

- UI/UX wireframes for components like dashboards, time logging interfaces, and report pages.
- Database schema design using MongoDB collections for users, projects, tasks, and time logs.
- API structure planning for FastAPI endpoints like /login, /create-task, /log-time.

Tools like draw.io, Figma, or ERDPlus may be used for designing diagrams and interfaces.

5.3. Implementation

Each iteration involves implementing the selected module(s) using the chosen tech stack:

- Frontend: React.js with Vite for fast development, Tailwind CSS for styling, and Axios for API calls.
- Backend API: FastAPI (Python) for high-performance, asynchronous APIs.
- Database: MongoDB to store user, project, task, and log details in a flexible document format.

5.4. Testing

Each module is tested rigorously during the same cycle it is developed. Testing includes:

- Unit Testing (e.g., FastAPI endpoints tested with pytest, React components with Jest).
- Integration Testing to ensure that frontend-backend communication is seamless.
- UI Testing using tools like Cypress for verifying user actions like task creation and log entry.

Bugs and inconsistencies found during this phase are fixed before the iteration is closed.

5.5. Evaluation and Feedback

After testing, the working module is shared with stakeholders (e.g., team members, guide) for feedback. During this evaluation:

- Functionality is validated.
- UI/UX improvements are suggested.
- Any missing edge cases are reported.

Feedback collected is documented and considered for implementation in the next cycle.

6. Project Plan

6.1 Objective

The objective of the Time Tracker system is to streamline time management and improve productivity tracking across a project team. The application enables Admins, Project Managers, and Developers to collaboratively track work hours, manage tasks, and generate insightful productivity reports. By providing real-time data and historical logs, the system empowers better planning, task allocation, and performance evaluation.

It aims to:

- Eliminate manual errors in time tracking.
- Improve transparency in project timelines.
- Offer data-driven insights for performance optimization.

6.2 Project Overview and Duration

The Time Tracker project is developed as part of the Capstone Project-III (Semester VII) curriculum and is scheduled over a 16-week development timeline, spanning from January 2025 to June 2025.

The development is structured using the Iterative Model, ensuring:

- Continuous delivery of incremental modules.
- Quick integration of stakeholder feedback.
- Early detection and resolution of issues.

Each iteration focuses on core functionality such as authentication, task management, time logging, and reporting, followed by design refinements and testing.

6.3 Scope of the Project

The system will encompass the following major functionalities:

User Role Management

- Role-based login for Admin, Project Manager, and Developer.
- Project and Task Management

- Creation of projects and assignment of tasks to developers.
- Task status tracking (To-Do, In-Progress, Completed).

Time Logging System

- Real-time tracking using timers and manual entry options.
- Daily and weekly logs for user accountability.

Report Generation and Analytics

- Visual dashboards for time utilization analysis.
- Exportable productivity reports (PDF format).

API-Driven Architecture

- FastAPI for backend REST APIs.
- Seamless frontend-backend communication using Axios in React.

The system will be scalable, modular, and easy to maintain, with flexibility to add future integrations like mobile app support or third-party tools.

6.4 Milestones and Timeline

Week	Milestone
Week 1–2	Requirement gathering, initial research, and feasibility study
Week 3–4	UI/UX prototyping using Figma and designing MongoDB schema
Week 5–6	Backend setup with FastAPI: Auth, User, and Task APIs
Week 7–8	React frontend structure, routing, login/register components
Week 9–10	Time logging feature implementation (timer/manual), log view
Week 11–12	Admin dashboard, analytics charts, and PDF report generation
Week 13–14	Testing: unit, integration, regression, and bug fixing
Week 15–16	Final polishing, deployment, documentation, and project demo

7. System Design

7.1 State Diagram

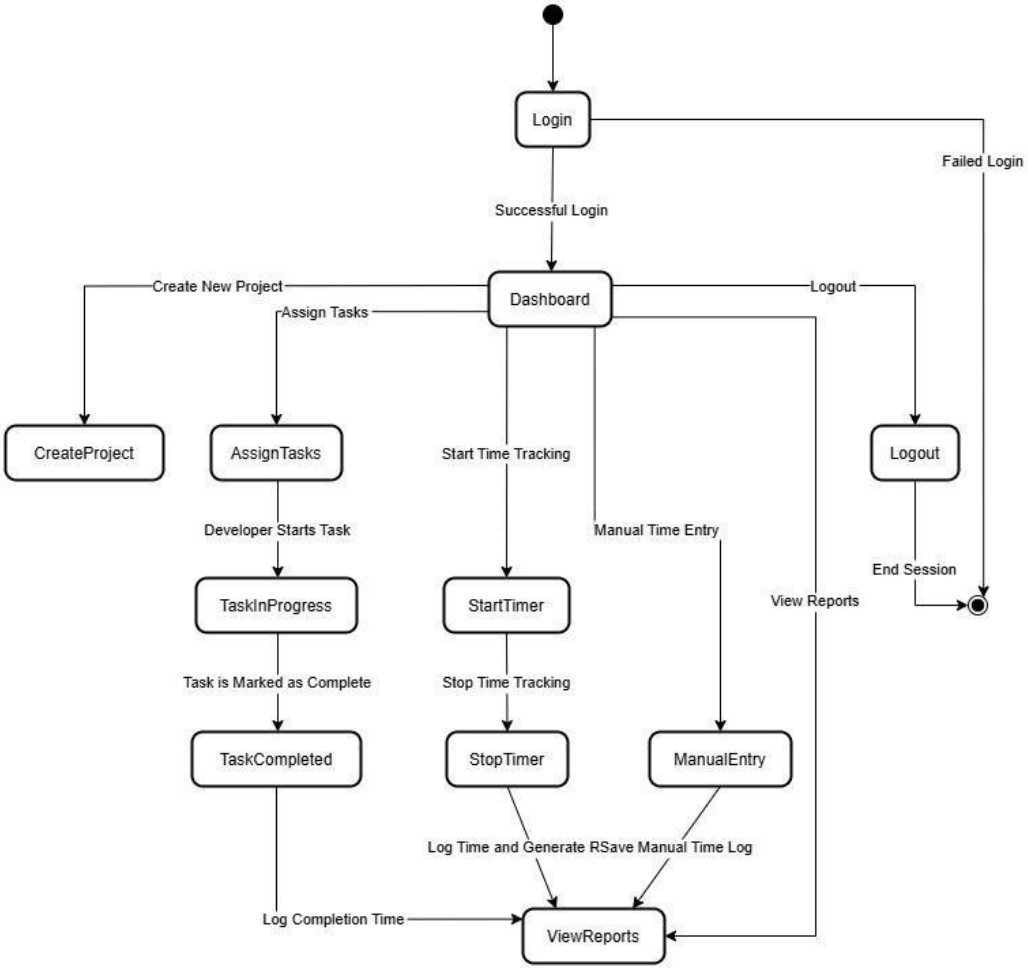

Figure 7.1 State Diagram

Description :-

This state diagram models the behavior of the Time Tracker System from the moment a user starts the application to the point they log out. It focuses on the user role-based navigation flow and system states after login. The system has three roles: Admin, Project Manager, and Developer. Here's a detailed breakdown of each state and transition:

1. Start
   - This is the initial state of the system. It represents the point where the application is launched and is waiting for user interaction.
   - The next action is for the user to attempt login.
2. Login → CheckCredentials
   - The user provides their login credentials (email and password), and the system enters the CheckCredentials state.
   - At this point, the system checks if the provided credentials are valid.

3. Invalid → ShowError
    - If the credentials are invalid, the system transitions to the ShowError state.
    - The user is shown an error message, and they are given the opportunity to retry login.
    - This forms a loop until valid credentials are entered or the application is closed.
4. Valid → Dashboard
    - Upon successful login, the system moves to the Dashboard state.
    - This is a role-routing hub, where the system decides which panel the user should be redirected to based on their role (Admin, Project Manager, Developer).
5. Admin Role → AdminPanel
    - If the user is an Admin, the system routes them to the AdminPanel.
    - From here, they can perform two main functions:
        - ManageUsers: Allows the admin to add, update, or remove users from the system.
        - GenerateReports: Enables report generation related to projects, time logs, and user performance.
6. Project Manager Role → ProjectManagerPanel
    - If the user is a Project Manager, they are directed to the ProjectManagerPanel.
    - From here, the project manager can access:
        - CreateProject: Create a new project with relevant details.
        - AssignTasks: Assign specific tasks to developers based on the project plan.
        - TrackProgress: Monitor task progress, developer logs, and overall project status.
7. Developer Role → DeveloperPanel
    - Developers are directed to the DeveloperPanel, where they manage their own task work.
    - The available options in this panel are:
        - ViewTasks: View the list of tasks assigned to them.
        - LogWorkHours: Log work hours by entering start and end times for tasks.
        - UpdateTaskStatus: Change the status of tasks (e.g., To-Do, In Progress, Completed).
8. Logout
    - From any role's panel (Admin, Project Manager, or Developer), the user has access to the Logout function.
    - Upon clicking logout, the system clears the session and transitions to the End state.
9. End
    - The system ends the current session and waits for the next user interaction.

7.2 Activity Diagram

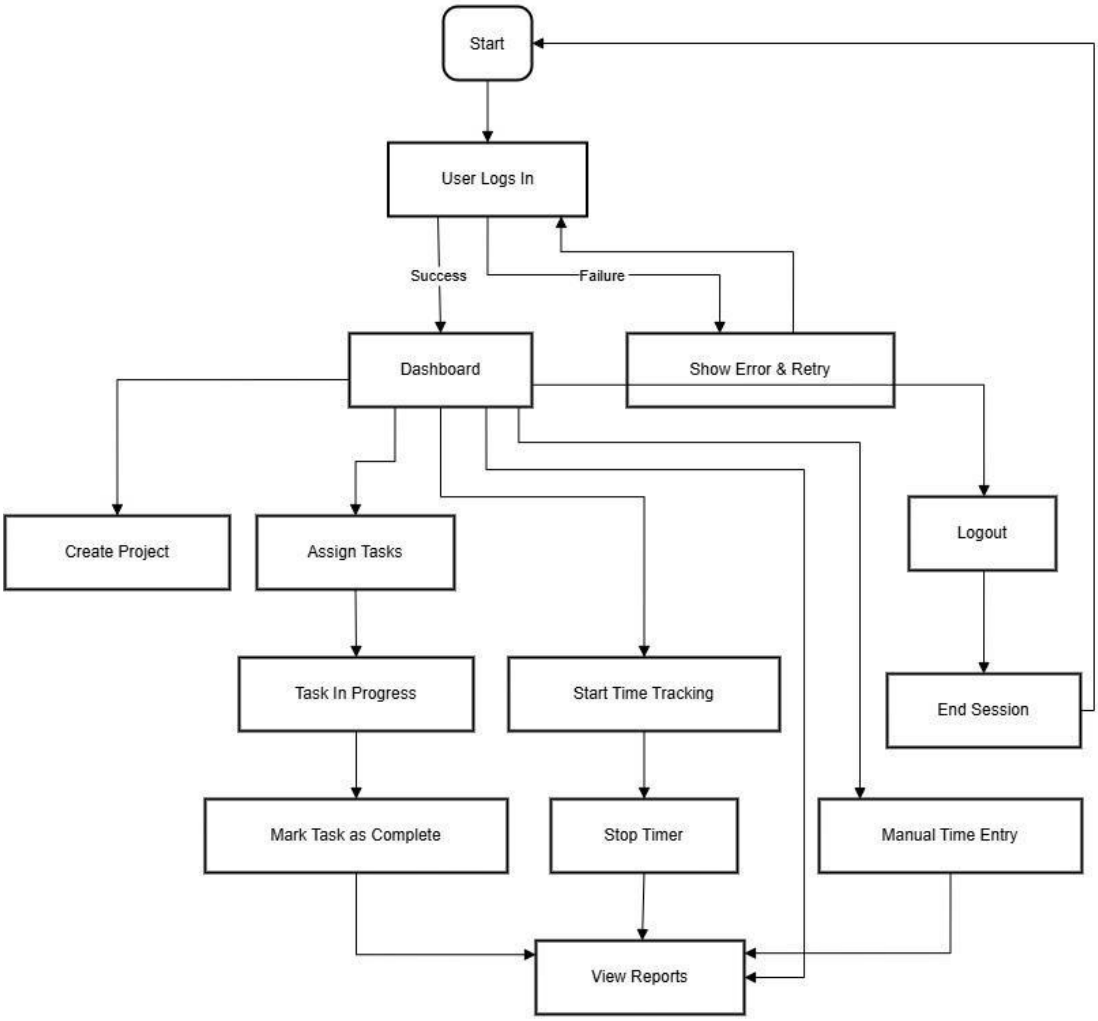

Figure 7.2 Activity Diagram

Description :-

This Activity Diagram represents the workflow of the Time Tracker System. It demonstrates how users interact with the system from login to logout based on their respective roles — Admin, Project Manager, and Developer. The diagram clearly separates the activities based on role and ensures a logical, step-by-step flow from system entry to exit.

1. Start
   - The initial activity of the system begins when the user accesses the application.
   - The first action a user can perform is the Login activity.
2. Login
   - The user is prompted to enter their credentials (email and password).

- Once submitted, the system proceeds to the CheckCredentials activity to verify the login data.

3. CheckCredentials
   - The system validates the credentials against the user database.
   - Two possible outcomes from this decision point:
     - Valid → Go to Dashboard
     - Invalid → ShowError and Retry
4. ShowError (Invalid Case)
   - If the credentials are incorrect, the user is shown an error message.
   - The activity loops back to the Login page, allowing the user to retry login.
5. Dashboard (Valid Login)
   - After successful login, the user is taken to a common Dashboard page.
   - From here, navigation is handled based on the user's role (determined at login):
     - Admin → AdminPanel
     - Project Manager → ProjectManagerPanel
     - Developer → DeveloperPanel
6. Admin Role → AdminPanel
   - The AdminPanel provides two key features:
     - ManageUsers: Admin can create, modify, or delete user accounts.
     - GenerateReports: Admin can generate time tracking and task completion reports.
   - After completing any task, the Admin has the option to Logout.
7. Project Manager Role → ProjectManagerPanel
   - Project Managers access their own panel with the following core actions:
     - CreateProject: Create a new project by entering details like name, start date, end date, and description.
     - AssignTasks: Assign specific tasks to developers for the created projects.
     - TrackProgress: Monitor the status of assigned tasks, completion rates, and developer activity.
   - All activities conclude with an option to Logout.
8. Developer Role → DeveloperPanel
   - Developers have access to task-related functionalities from the DeveloperPanel:
     - ViewTasks: View the list of assigned tasks along with statuses.
     - LogWorkHours: Log the time spent working on a task by entering start and end times.
     - UpdateTaskStatus: Change the status of a task (e.g., To Do → In Progress → Completed).
   - As with other roles, the user can Logout after completing tasks.
9. Logout
   - The Logout activity is accessible from every user role's panel.
   - Once a user clicks logout, the session is closed securely, and the system transitions to the end.
10. End
    - Marks the end of the system workflow.
    - The system is now ready for a new login session from the same or another user.

7.3 Use Case Diagram

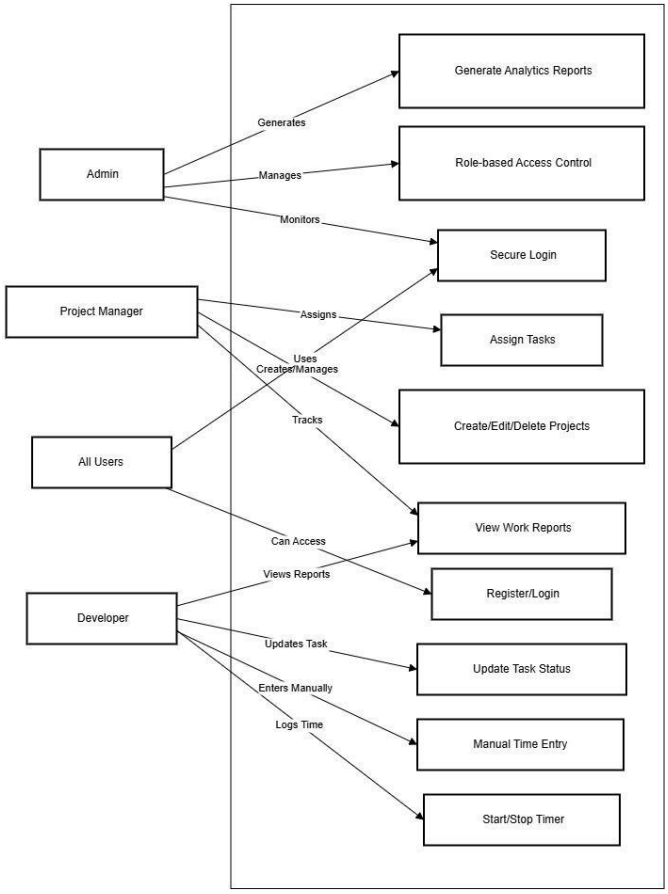

Figure 7.3 Use Case Diagram

Description :-

The Use Case Diagram of the Time Tracker System illustrates the core functionalities accessible by different types of users: Admin, Project Manager, and Developer. It visualizes the interaction between users (actors) and system features (use cases), providing a high-level overview of the system's capabilities and role-based access.

Actors Involved:

1. Admin
    - Has access to user management and system reporting functionalities.
2. Project Manager
    - Manages projects and tasks, and tracks work progress of developers.
3. Developer
    - Works on assigned tasks and logs work activity.

Use Cases & Actor Interactions:

Common Use Cases for All Roles:

- Login

    - All users must authenticate before accessing their respective dashboards.
- Logout
    - Ends the user session securely for all roles.

Admin Use Cases:

- Manage Users
    - The Admin can add, edit, or delete user accounts and assign roles (Developer or Project Manager).
- Generate Reports
    - Admin generates reports regarding user activity, time logs, and task status.

Project Manager Use Cases:

- Create Project
    - Allows the project manager to initiate new projects and define their scope.
- Assign Task
    - Tasks can be created and assigned to developers under specific projects.
- Track Progress
    - View real-time progress of projects, tasks, and logged hours.

Developer Use Cases:

- View Tasks
    - Developers can see the list of tasks assigned to them.
- Log Work Hours
    - Developers record start and end times for their daily tasks.
- Update Task Status
    - Developers can update the progress of their tasks (e.g., To-Do → In Progress → Completed).

System Behavior Summary:

- The diagram ensures role-based access control, where each actor is limited to functionalities relevant to their responsibilities.
- Authentication is a prerequisite for all other actions.
- Use cases are clearly distributed to avoid overlap between Admin, Manager, and Developer functionalities.

7.4 Sequence Diagram

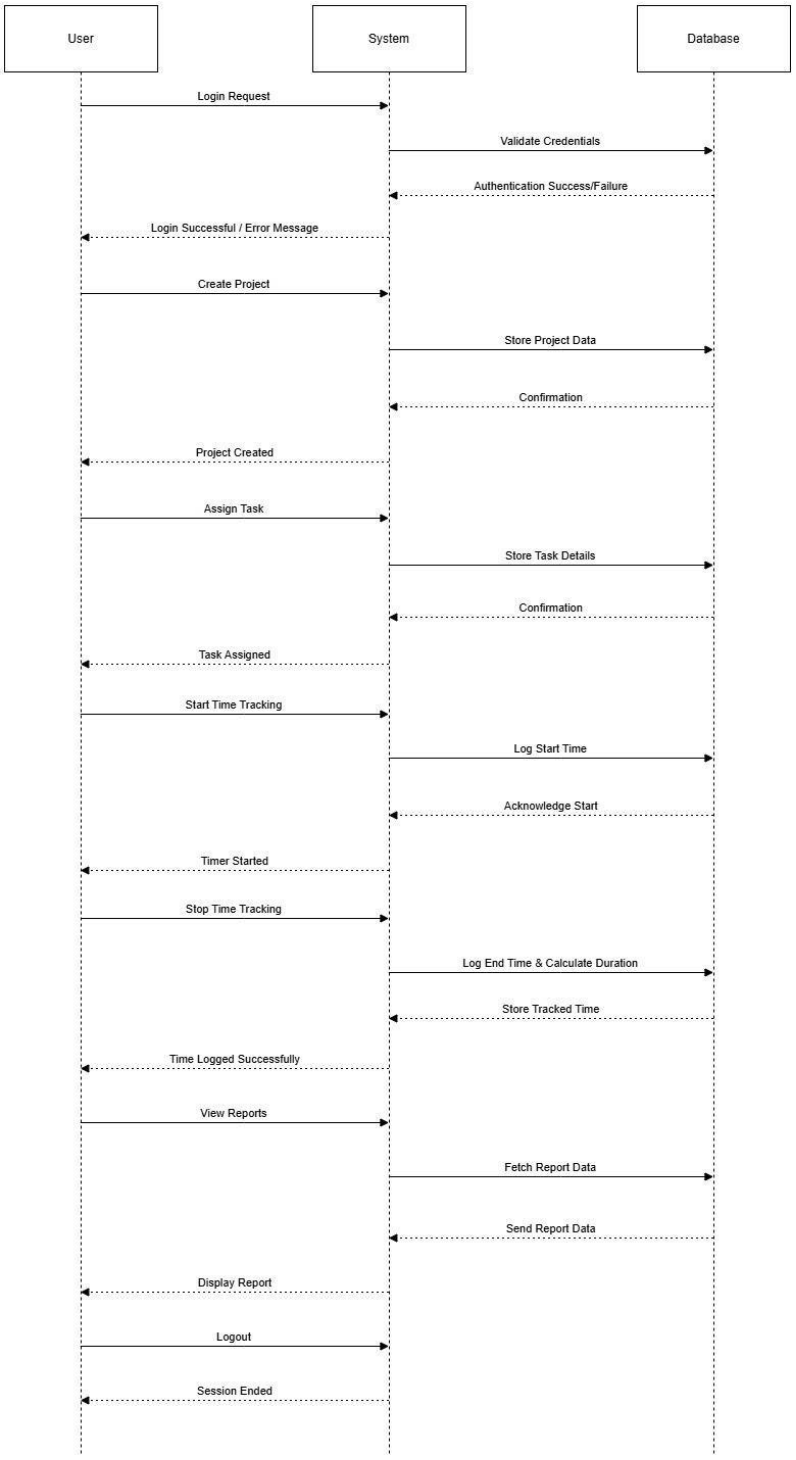

Figure 7.4 Sequence Diagram

Description :-

The Sequence Diagram represents the time-ordered interaction between different objects in the Time Tracker System during the login and role-based navigation process. It shows how messages are passed between components to achieve user authentication and redirection to the appropriate dashboard based on the user's role.

Participants (Objects / Lifelines):

1. User : The actor initiating the interaction — can be an Admin, Project Manager, or Developer.
2. Login Page : The UI interface where the user enters credentials.
3. Authentication Controller : Handles the business logic for checking user credentials.
4. Database : Stores and retrieves user data, including email, password, and role.
5. Dashboard : The common dashboard that displays after successful login, responsible for routing users to role-specific panels.
6. AdminPanel / ProjectManagerPanel / DeveloperPanel : The respective dashboards shown to the user based on their role.

Flow of Events (Step-by-Step Message Sequence):

1. User → Login Page
   - The interaction begins when the user opens the system and enters their email and password into the login form.
   - This information is passed to the backend for verification.
2. Login Page → Authentication Controller
   - The login page sends the user's credentials to the Authentication Controller through a method call like checkCredentials(email, password).
3. Authentication Controller → Database
   - The controller then queries the Database to validate the credentials.
   - A typical query might be SELECT * FROM users WHERE email = ? AND password = ?.
4. Database → Authentication Controller
   - If the user is found and credentials match, the database returns the user data along with their role (Admin, Project Manager, or Developer).
5. Authentication Controller → Dashboard
   - The Authentication Controller forwards the user information to the Dashboard component.
6. Dashboard → Role-Based Routing
   - Based on the user's role, the Dashboard redirects the user to the respective panel:
     - Admin → AdminPanel
     - Project Manager → ProjectManagerPanel
     - Developer → DeveloperPanel
   - This part may use conditional logic like:
7. User ← Role-Based Panel
   - The user finally gains access to their role-specific dashboard, where they can interact with available system features.

7.5 Class Diagram

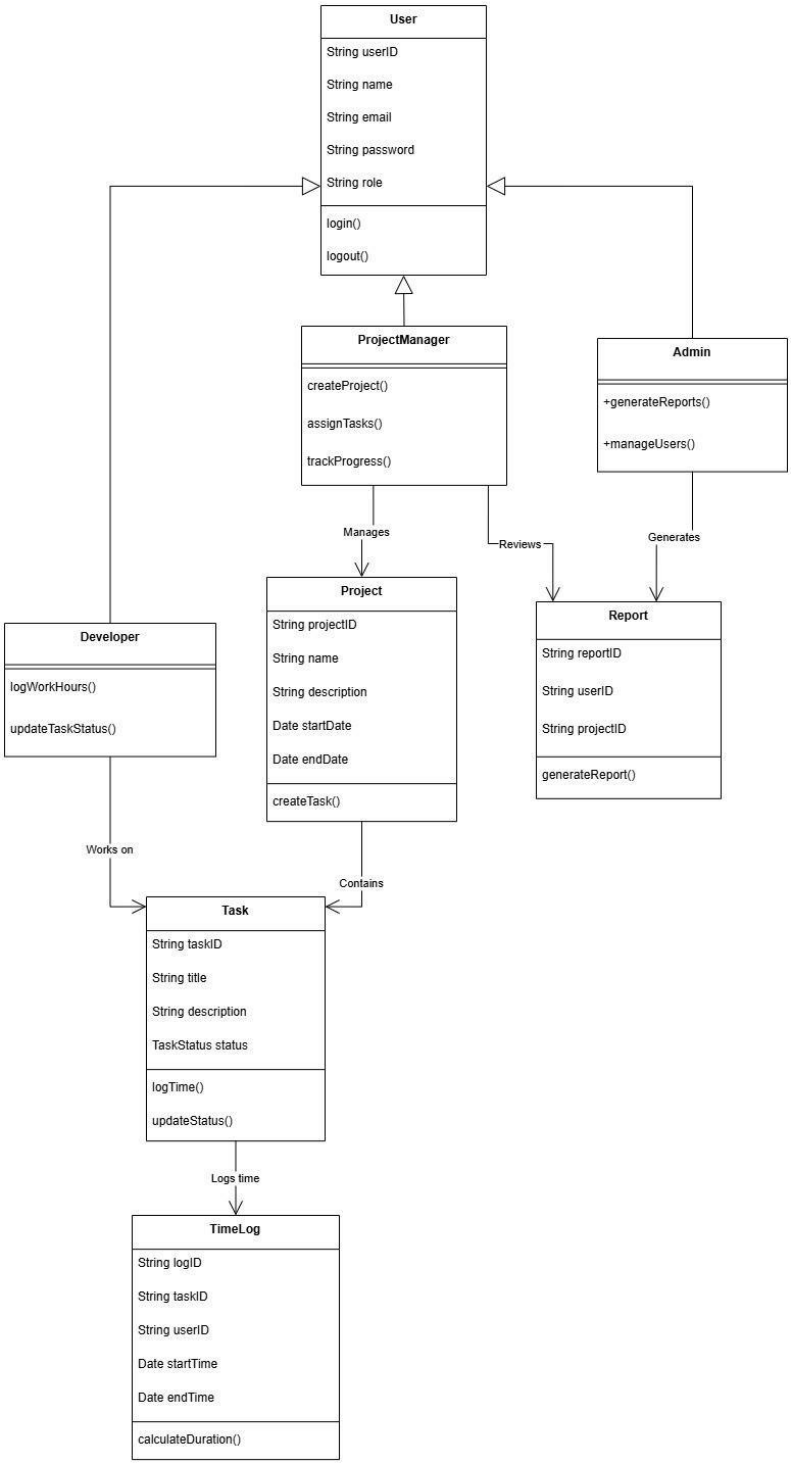

Figure 7.5 Class Diagram

Description :-

The Class Diagram of the Time Tracker System provides a structural view of the system by showing the classes, attributes, methods, and relationships among them. It models the system's core entities involved in user management, project/task tracking, and time logging functionality.

Main Classes and Descriptions:

1. User
   - Attributes:
     - userId: int – Unique identifier for the user.
     - name: String – Name of the user.
     - email: String – User's email address.
     - password: String – Encrypted password.
     - role: String – Specifies the user's role (Admin, Project Manager, or Developer).
   - Methods:
     - login() – Authenticates the user.
     - logout() – Ends the current user session.
   - Relationships:
     - Generalization: Inherited by Admin, ProjectManager, and Developer classes.
2. Admin (Inherits from User)
   - Methods:
     - manageUsers() – Add, update, or delete user accounts.
     - generateReports() – Generate system usage and time reports.
3. ProjectManager (Inherits from User)
   - Methods:
     - createProject() – Create new projects.
     - assignTask() – Assign tasks to developers.
     - trackProgress() – Monitor developer activity and task completion.
4. Developer (Inherits from User)
   - Methods:
     - viewTasks() – View list of assigned tasks.
     - logWorkHours() – Log daily hours against tasks.
     - updateTaskStatus() – Change the status of tasks (e.g., In Progress, Completed).
5. Project
   - Attributes:
     - projectId: int – Unique identifier for the project.
     - projectName: String – Name of the project.
     - description: String – Brief detail about the project.
     - startDate: Date
     - endDate: Date
   - Methods:
     - addModule() – Add modules to the project.
     - getStatus() – Retrieve project status.
   - Relationships:
     - Association: Managed by ProjectManager.

- Aggregation: Contains multiple Tasks and Modules.

6. Module
   - Attributes:
     - moduleId: int
     - moduleName: String
     - projectId: int
   - Methods:
     - assignTaskToDeveloper() – Link a task to a developer under the module.
   - Relationships:
     - Associated with Project (belongs to one).
     - Associated with Task (contains many).
7. Task
   - Attributes:
     - taskId: int
     - taskName: String
     - status: String
     - developerId: int
   - Methods:
     - updateStatus() – Change task progress.
     - trackTime() – Used to monitor time logged on the task.
   - Relationships:
     - Assigned to Developer.
     - Belongs to a Module.
8. TimeLog
   - Attributes:
     - logId: int
     - taskId: int
     - developerId: int
     - startTime: DateTime
     - endTime: DateTime
     - duration: Float
   - Methods:
     - calculateDuration() – Calculates time worked on a task.
   - Relationships:
     - Aggregated by Task, associated with Developer.

Relationships Summary:

- Inheritance:
  - Admin, ProjectManager, and Developer inherit from User.
- Associations:
  - ProjectManager ↔ Project
  - Project ↔ Module ↔ Task
  - Developer ↔ Task, TimeLog
- Aggregation / Composition:
  - Project aggregates Modules and Tasks.
  - Task is composed of TimeLog.

7.6 Flowchart Diagram

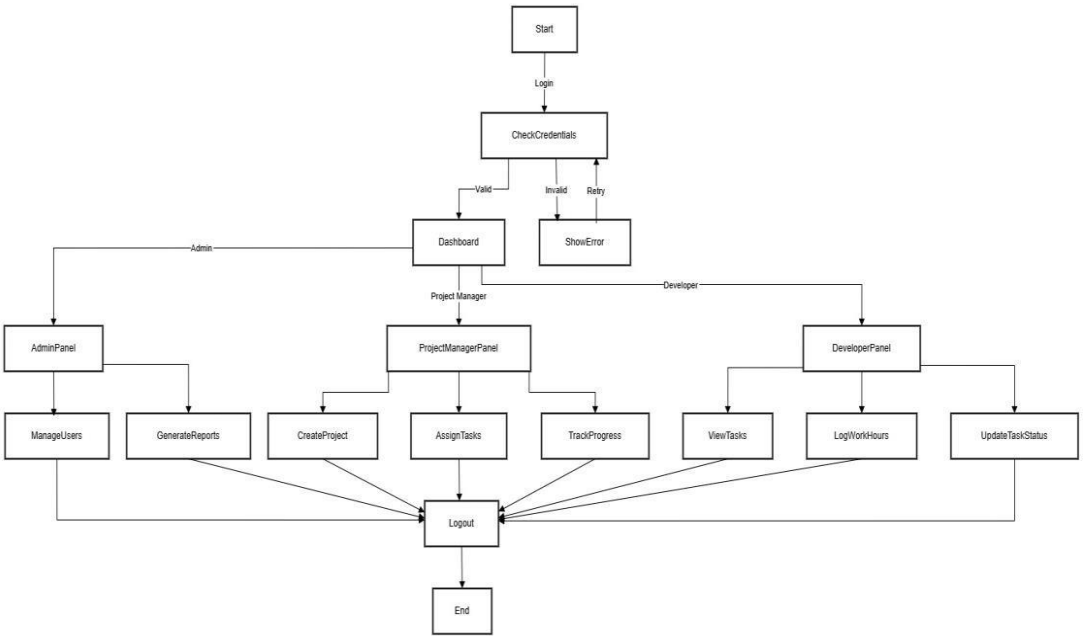

Figure 7.6 Flowchart Diagram

Description :-

The Flowchart Diagram of the Time Tracker System visually represents the logical flow of processes from user login to task management and logout, highlighting conditional decisions, system actions, and role-based behavior. It simplifies how users interact with the system through a step-by-step workflow.

Workflow Summary:

The flow begins when a user accesses the system and ends with their logout, depending on their role (Admin, Project Manager, or Developer). Each decision node branches the flow according to user roles and system outcomes.

Step-by-Step Process Description:

1. Start
   - The system process begins when a user opens the application.

2. Login Page
   - The user is presented with the login interface where they input their credentials (email and password).

3. Credential Validation
   - The system verifies the credentials with data stored in the database.

Decision: Are Credentials Valid?

- Yes: Continue to role check.
- No: Display an "Invalid Login" message and return to the login page.

4. Role Identification

- If credentials are valid, the system retrieves the role of the user (Admin / Project Manager / Developer).

Role-Based Navigation:
If User is Admin:

- Redirected to Admin Panel.
- Admin can:
  - Manage Users
  - View Reports
  - Oversee Project Data

If User is Project Manager:

- Redirected to Manager Panel.
- Manager can:
  - Create Projects
  - Assign Tasks to Developers
  - Track Task and Project Progress

If User is Developer:

- Redirected to Developer Panel.
- Developer can:
  - View Assigned Tasks
  - Start/Stop Time Tracker
  - Log Work Hours
  - Update Task Status

Step – Logout

- After performing their respective actions, the user may choose to log out.
- System ends the session and returns to the Start state.

Key Observations:

- The flowchart uses decision diamonds for validations and role checks.
- Ensures secure login and prevents unauthorized access with a proper feedback loop for invalid attempts.
- Implements clear separation of concerns based on roles, improving system navigation and usability.
- Follows a modular approach that makes it easier to integrate additional features or modify role behavior in the future.

Prototype:-

7.7 Register Page :-

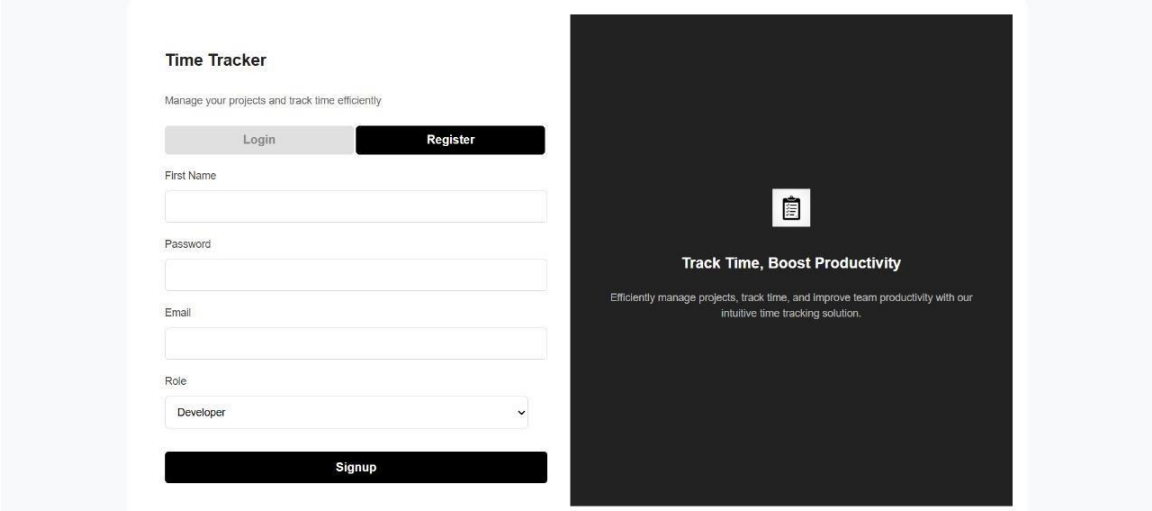

Figure 7.7 Register Page

Description :-
The Register Page of the Time Tracker System is designed to allow new users to sign up by providing essential information such as their identity details and role in the system. It supports a secure and structured registration flow, which ensures that each user is correctly categorized into their respective role: Admin, Project Manager, or Developer.

Form Components and Field Description:

1. **Name**
   - Input field for entering the full name of the user.
   - Validation ensures the name is not left empty and doesn't contain numeric characters.
2. **Email**
   - Input field to collect a valid email address.
   - Acts as the unique identifier for the user account.
   - Email format validation (e.g., abc@domain.com) is applied.
3. **Password**
   - Input field for setting a secure password.
   - Password strength validation is implemented (minimum length, use of special characters, etc.).
   - Optionally may include password visibility toggle for user convenience.
4. **Confirm Password**
   - Field to confirm the password entered above.
   - System checks for a match between both password fields to avoid typos or errors.

5. **Role Selection (Dropdown)**
    - A dropdown menu where the user selects their role from:
        - **Admin**
        - **Project Manager**
        - **Developer**
    - This selection determines their access level and dashboard navigation after login.
6. **Register Button**
    - Submits the form data to the backend system.
    - Triggers validation and stores the user data in the database upon success.

Security and Validation:

- **Input Validation:** Ensures all fields are correctly filled before form submission.
- **Password Security:** Passwords are hashed (using bcrypt or similar encryption) before storing in the database.
- **Email Uniqueness Check:** System checks if the email already exists in the database to prevent duplicate registrations.
- **Role Validation:** Ensures a valid role is selected before allowing account creation.

Flow of Actions (Behind-the-Scenes):

1. User fills in the registration form.
2. Upon clicking Register, frontend performs client-side validation.
3. Data is sent to the backend API endpoint like /register.
4. Backend:
    - Validates data.
    - Hashes password.
    - Stores the user in the database along with their role.
5. A confirmation response is sent back:
    - On success: User is redirected to login page with a success message.
    - On failure: Appropriate error message is shown (e.g., "Email already exists", "Passwords do not match").

7.8 Log in :-

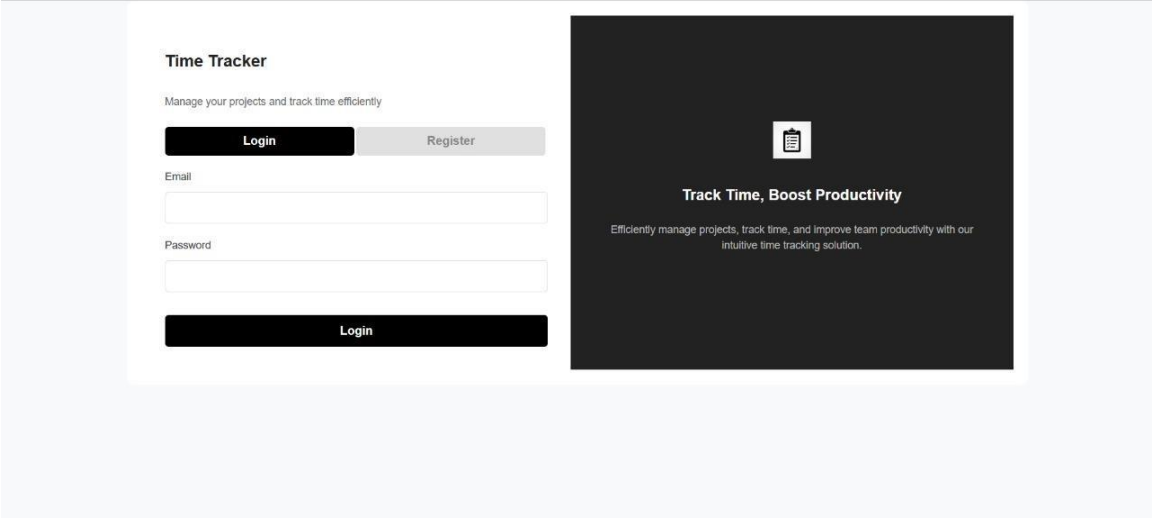

Figure 7.8 Log in

Description :-

The Login Page serves as the gateway to the Time Tracker System, allowing registered users to authenticate and access their role-based dashboards. It emphasizes security, usability, and access control, ensuring that only valid users can proceed.

Form Components and Field Description:

1. **Email**
   - Input field to enter the registered email address.
   - Acts as the primary identifier for the user.
   - Includes validation for proper email format (e.g., user@example.com).
2. **Password**
   - Secure input field for the user's password.
   - Password input is masked to enhance security.
   - May include a show/hide password toggle for ease of use.
3. **Login Button**
   - Triggers the login action.
   - On click, the form data is validated and sent to the backend for authentication.
   - Successful login redirects user to the appropriate dashboard based on their role.

Security and Validation Measures:

- **Email Validation:** Ensures the email is in proper format and exists in the system.
- **Password Verification:** Checks if the entered password matches the hashed password stored in the database.
- **Error Handling:**
  - If login fails, displays meaningful messages like:
    - "Invalid email or password"
    - "User not registered"

- **Data Protection:** Passwords are never stored or transmitted in plain text; they are encrypted using hashing algorithms (e.g., bcrypt).

Workflow and Backend Interaction:

1. User inputs email and password.
2. Clicks the Login button.
3. Frontend performs client-side validation.
4. Sends a POST request to backend API endpoint (e.g., /login).
5. Backend checks:
    - Whether the email exists.
    - Whether the password matches the stored hashed password.
6. If validated:
    - User is authenticated.
    - Role is identified (Admin / Project Manager / Developer).
    - User is redirected to their corresponding dashboard.
7. If validation fails:
    - Returns appropriate error response.
    - User stays on login page with error message displayed.

Role-Based Redirection After Login:

Upon successfully logging into the Time Tracker system, users are automatically redirected to their respective dashboards based on their assigned roles. This role-based redirection ensures that each user is presented with a tailored interface containing features relevant to their responsibilities.

- **Admin** users are redirected to the **Admin Dashboard**, where they can manage users, oversee all projects, and access comprehensive reports and analytics.
- **Project Managers** are redirected to the **Project Manager Dashboard**, which allows them to manage project workflows, assign tasks, and monitor developer performance.
- **Developers** are redirected to the **Developer Dashboard**, where they can view assigned tasks, log time, and track their progress.

This ensures users access only the functionalities that are permitted for their role.

Additional Features (Optional Enhancements):

- **"Forgot Password?" Link** – Allows users to initiate password recovery.
- **"Remember Me" Checkbox** – Keeps users logged in on the same device.
- **"New User? Register" Link** – Redirects to the registration page for new users.

7.9 ADMIN DASHBOARD

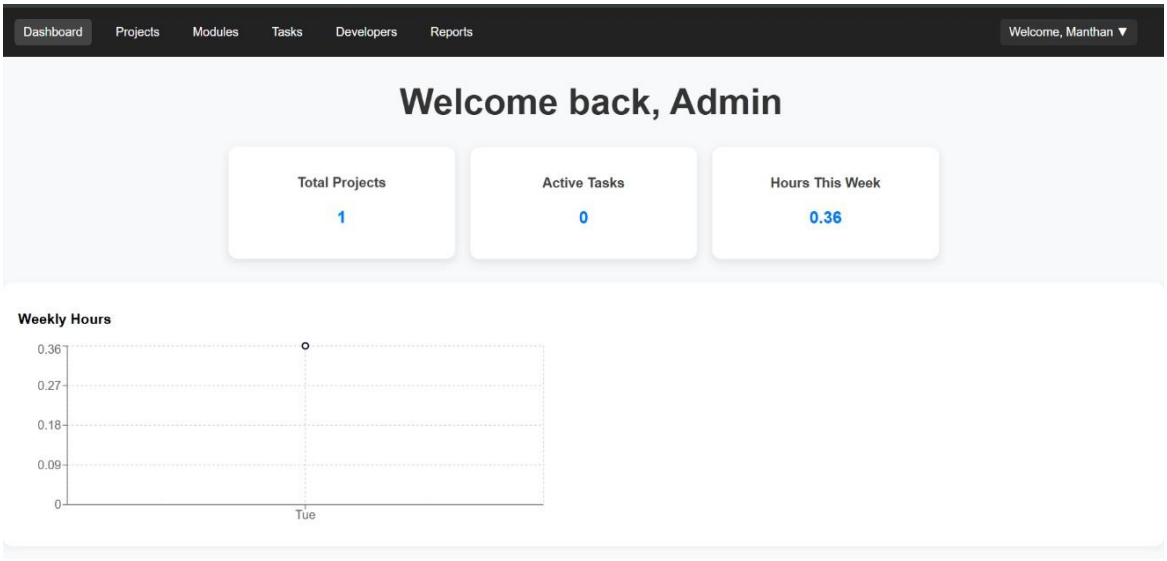

Figure 7.9 Admin Dashboard

Description :-

The Admin Dashboard is the central control panel for administrators, offering a summarized view of ongoing projects, active tasks, and tracked working hours. Designed with simplicity and clarity in mind, it helps admins monitor productivity and system usage efficiently.

Top Navigation

A fixed dark navigation bar at the top provides quick access to key sections:

- Dashboard, Projects, Modules, Tasks, Developers, Reports
  On the right, a user profile dropdown shows "Welcome, Manthan", indicating the logged-in admin user with access to account controls.

Welcome Message

At the center of the page, the greeting "Welcome back, Admin" personalizes the experience and confirms the user role. It reinforces that this view is tailored for admin-level monitoring and decisions.

Key Metrics Summary

Below the welcome message, three clean and compact info cards summarize current system stats:

1. Total Projects – *1*
   Indicates that one project has been added to the system.

2. Active Tasks – *0*

   Shows that there are currently no ongoing or in-progress tasks.

3. Hours This Week – *0.36*

   Reflects the total developer hours logged for the current week, helping track time efficiency.

Each stat is visually emphasized with bold text and blue highlights, allowing for easy scanning at a glance.

Weekly Hours Chart

A line chart labeled “Weekly Hours” gives a visual breakdown of time logged throughout the week. The X-axis represents days, while the Y-axis quantifies the hours. Currently, it shows 0.36 hours logged on Tuesday, with no activity on other days—helpful for spotting productivity trends or inactivity early.

Design & Usability

The layout is clean, responsive, and minimal, using clear fonts, modern card components, and a soft background to keep focus on data. It’s built for usability and supports future expansion like mobile responsiveness or added features.

7.10 ADMIN PROJECT

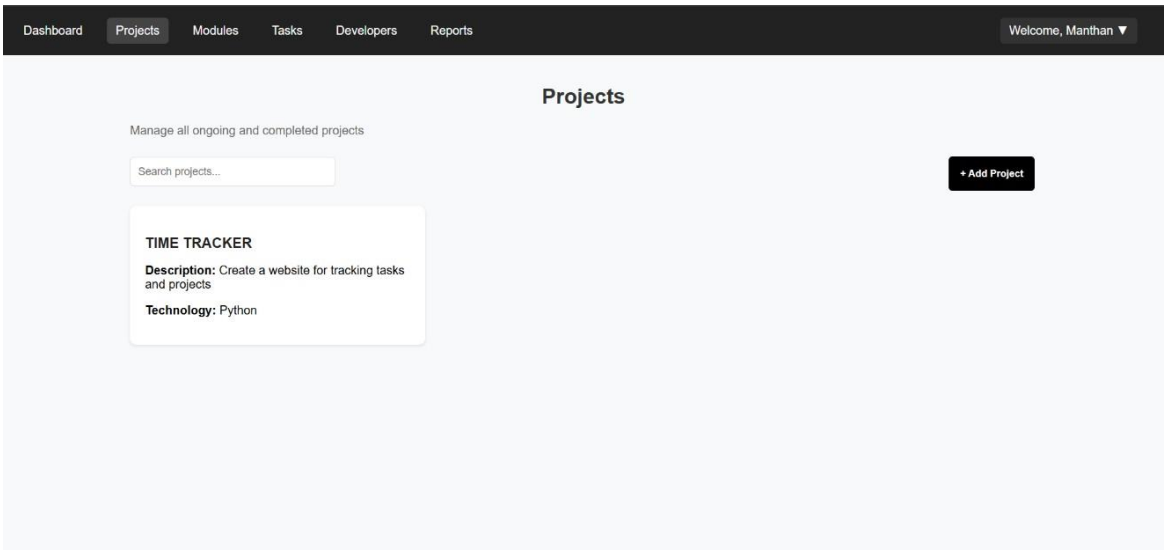

Figure 7.10 ADMIN PROJECT

Description :-

The Projects Page serves as the foundation for managing and organizing all task-related activities within the Time Tracker system. It allows administrators to view, add, and manage all ongoing and completed projects, ensuring that each initiative is properly structured and tracked from initiation to completion.

Navigation & Header

The page maintains a consistent top navigation bar with key sections: Dashboard, Projects, Modules, Tasks, Developers, Reports Here, the "Projects" tab is highlighted, indicating the current active section. On the top-right corner, a dropdown menu shows “Welcome, Manthan”, confirming the currently logged-in admin user.

Page Purpose & Description

Directly under the header, a subtle instruction: “Manage all ongoing and completed projects” helps orient the user and clarifies the page’s intent – it’s a central place to create and monitor projects that form the base of the module-task-developer hierarchy.

Project Search & Action Button

- Search Bar: Allows users to filter projects by name, enabling quick access even when handling multiple projects at once. This improves navigation efficiency.
- Add Project Button: Located prominently on the right side as a black button labeled “+ Add Project”, it allows users to initiate a new project. This would typically lead

to a form where users can input the project name, description, and associated technology stack.

Project Display Card

Each project is presented in a card layout, showcasing essential project details in a clear and compact manner. In the current view, one project is displayed:

- Project Name: TIME TRACKER – the title of the main project being managed.
- Description: "Create a website for tracking tasks and projects" – summarizing the core goal and functionality of the application.
- Technology Used: *Python* – specifying the backend technology stack, useful for team assignment and resource planning.

This modular card system makes the UI scalable for future enhancements like status indicators, progress bars, or quick-edit buttons.

Design & Usability

- Minimalist Aesthetic: The page uses a white background, simple text formatting, and neatly spaced elements to keep the focus on content.
- Responsiveness: The layout supports clear visibility across different devices, ensuring that admins can manage projects easily whether on desktop or mobile.
- Expandable Structure: The structure of the cards and controls is flexible enough to support future features like project timelines, file attachments, or integration indicators.

7.11 ADMIN MODULE

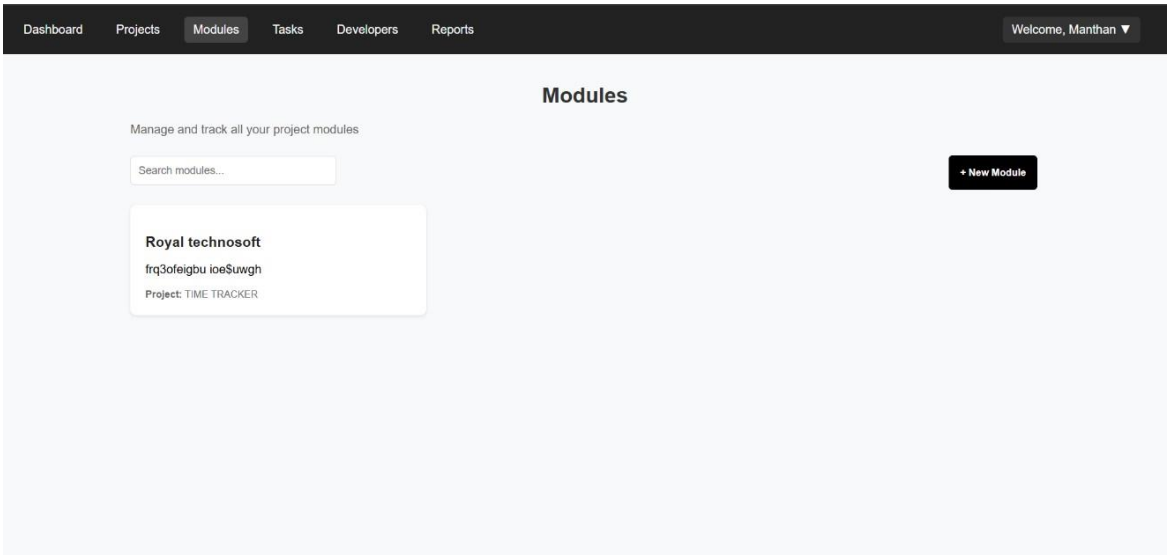

Figure 7.11 ADMIN MODULE

Description :-

The Modules page allows administrators and project managers to efficiently view, manage, and track individual modules associated with various projects. This section serves as an intermediate layer between project creation and task assignment, offering modular project structure and organization.

Navigation & User Interface

Like other pages in the application, the top features a persistent dark navigation bar with links to: Dashboard, Projects, Modules, Tasks, Developers, Reports .The "Modules" tab is highlighted, indicating the active page. On the top-right, the user dropdown displays "Welcome, Manthan", confirming the currently logged-in admin account.

Module Management Features

Beneath the page title **"**Modules", users are presented with a minimalistic and clean interface offering the following core features:

- **Search Bar**: A text input labeled "Search modules..." allows for quick filtering and retrieval of specific modules by name. This enhances usability when managing multiple modules across projects.
- **New Module Button**: Positioned at the top right is a distinct black button labeled "+ New Module". Clicking this would navigate to or open a form for creating a new module, likely associating it with an existing project and providing name/description fields.

Module Card Display

Below the search and control panel, existing modules are displayed using individual **module cards** for easy visual management. Each card includes:

- **Module Name**: Example: **Royal technosoft** – the title of the module, possibly referring to a client or project area.
- **Module Description**: A placeholder/auto-generated text: *"frq3ofeijgbu ioe$uwgh"* is shown (likely test or dummy data). In production, this would typically contain a short summary of the module's purpose or functionality.
- **Project Association**: Labeled at the bottom of the card as **"Project: TIME TRACKER"**, this shows the parent project under which the module falls. This structure helps users identify where the module fits in the overall system.

Usability & Design

- **Clean Layout**: White space is effectively used to reduce clutter and ensure that modules are readable and distinguishable.
- **Responsive Cards**: The modular card layout makes the UI scalable for future enhancements like editing, deletion, or task association buttons.
- **User-Centered**: The interface is tailored for quick actions and a smooth user experience, keeping module management fast and efficient.

7.12 ADMIN TASKS

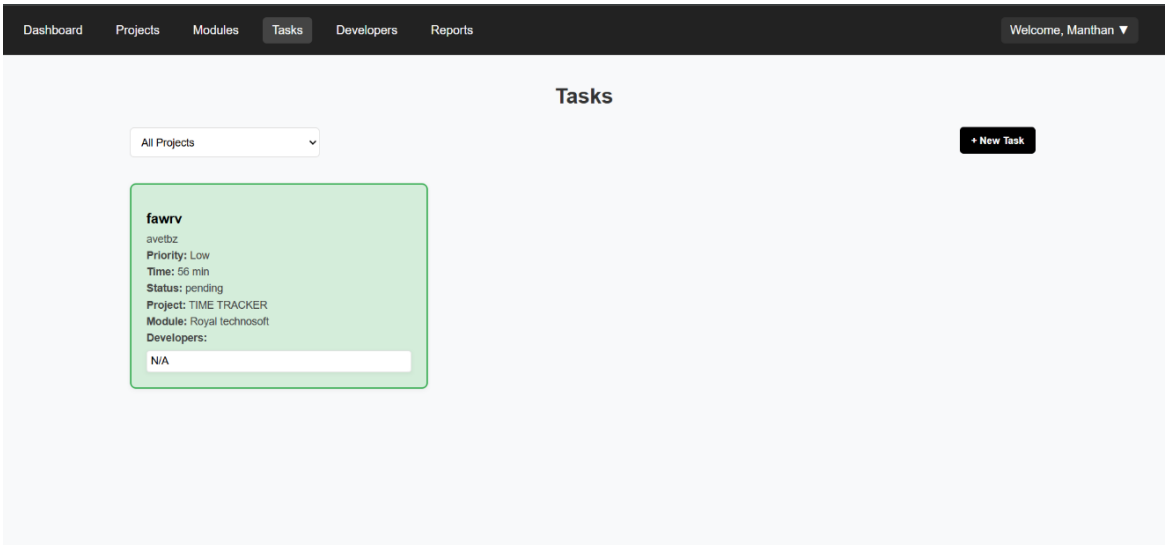

Figure 7.12 ADMIN TASKS

Description :-

The Tasks Page plays a pivotal role in the Time Tracker system by offering a streamlined interface for managing all tasks associated with different projects and modules. It acts as a real-time control panel for viewing, assigning, and updating tasks while tracking individual productivity and project progression.

Navigation & Layout

At the top of the page is a consistent navigation bar with tabs for Dashboard, Projects, Modules, Tasks, Developers, and Reports. The Tasks tab is actively highlighted, indicating the current section. The top-right corner includes the user dropdown labeled "Welcome, Manthan", confirming admin access.

Filter & Task Addition

- **Project Filter Dropdown:**
  Positioned on the left side, this dropdown lets users filter tasks by specific projects or view all tasks at once ("All Projects"). This is crucial for large teams handling multiple projects simultaneously, helping to maintain task visibility and reduce clutter.
- **+New Task Button:**
  Located on the right in bold black, this button initiates the creation of a new task. When clicked, it likely triggers a modal or a form where an admin can enter task title, description, assign developers, select module, set priority, and estimate the time.

Task Card Design

Each task is displayed as a distinct, colored card (in light green), presenting essential task details in a readable and organized layout. In the shown screenshot, one task is listed with the following data:

- Task Title: fawrv – the name of the task.
- Description/Note: avetbz – a possible short description or code associated with the task.
- Priority: Low – indicates urgency, which may influence scheduling and developer workload.
- Time: 56 min – estimated or logged time spent on the task.
- Status: Pending – shows the task's current state, helping track progress.
- Project: TIME TRACKER – links the task to the primary project.
- Module: Royal technosoft – shows the module under which the task falls.
- Developers: N/A – no developer has been assigned yet; this is expected to change when the task is delegated.

The structure is both compact and informative, enabling admins to quickly scan through multiple tasks and understand their current state.

Functionality & UX Highlights

- Modular Task Visualization: Each card is self-contained and color-coded for easy priority detection.
- Editable Interface: Although not shown directly, cards may include interactive elements (edit, delete, assign) in full implementation.
- Scalability: As more tasks are added, cards will dynamically populate the page and can be filtered by project.

Summary

The Tasks Page provides a focused and efficient environment for managing individual work units within the Time Tracker application. With built-in filtering, structured task cards, and a minimal interface, it helps administrators maintain workflow clarity and ensure timely task delegation. The layout balances simplicity with functionality, laying the groundwork for future enhancements like drag-and-drop task sorting, real-time collaboration, or visual timelines.

7.13 ADMIN DEVELOPERS

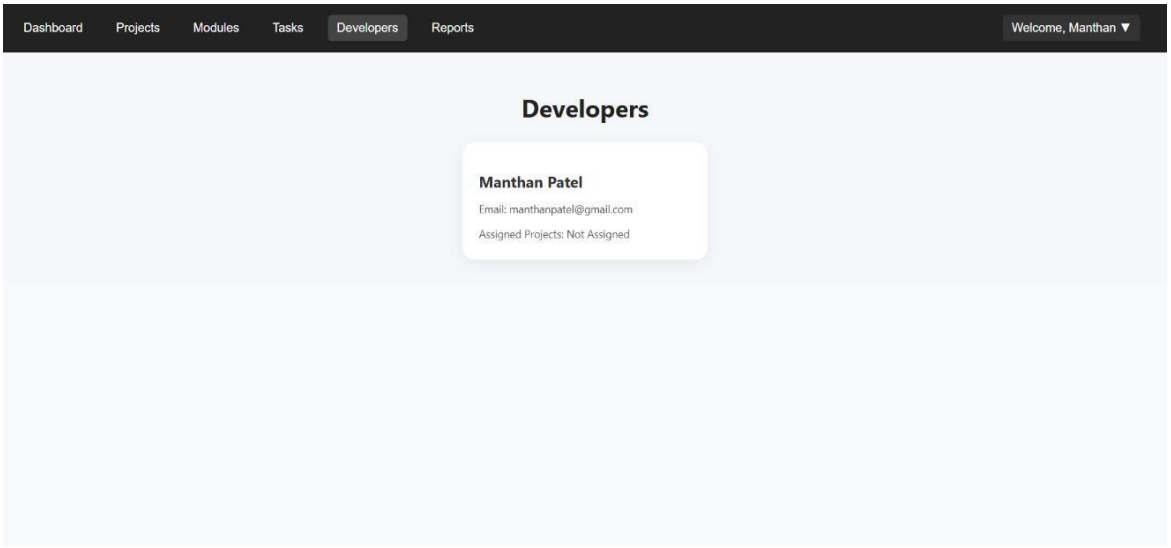

Figure 7.13 ADMIN DEVELOPERS

Description :-

The Developers Page is designed to manage and display information about the developers registered within the Time Tracker system. It provides a clear and simple overview of each developer, including their identity, contact information, and project assignments.

Navigation & Layout

At the top is a consistent navigation bar with key tabs: Dashboard, Projects, Modules, Tasks, Developers, and Reports. The Developers tab is currently active, indicating that the user is viewing the developer management section. On the top right corner, the user dropdown labeled "Welcome, Manthan" is present, signifying admin access or a logged-in user.

Developer Card

Each developer is displayed as an individual card on the screen. In the example provided, only one developer is listed:

- **Name**: Manthan Patel
- **Email**: manthanpatelr@gmail.com
- **Assigned Projects**: Not Assigned

This format delivers concise but important details about each developer, helping administrators quickly identify users and determine who is actively engaged in projects.

Features & Functionality

- **Visual Clarity**: The clean card layout makes it easy to browse developer information without clutter. Each developer's details are centered in a soft-white card with subtle shadows for visibility and separation.
- **Unassigned Status Highlight**: The tag "Not Assigned" under Assigned Projects makes it immediately clear that this developer is not currently involved in any active project. This can prompt the admin to assign roles or tasks accordingly.
- **Scalability**: Though currently displaying a single developer, the layout is scalable and can accommodate multiple developers listed in a scrollable or grid format.
- **Admin-Only View (assumption)**: Based on the layout, this page appears designed primarily for admin users to manage and assign developers to relevant projects or modules.

Expected Functional Enhancements (Future Scope)

While this page currently displays static developer details, possible enhancements could include:

- Buttons to Edit, Assign Projects/Modules, or Delete Developer.
- A search bar or filter options to find developers based on project/module.
- Display of current task workload, total time spent, or performance insights.

Summary

The Developers Page provides a minimal yet essential snapshot of the human resources behind project execution. By listing key information like name, email, and assigned projects, it serves as a foundational directory to track and manage developer involvement in the system. Its simplicity and clarity ensure that administrators can maintain oversight and optimize task distribution effectively.

7.14 ADMIN REPORTS

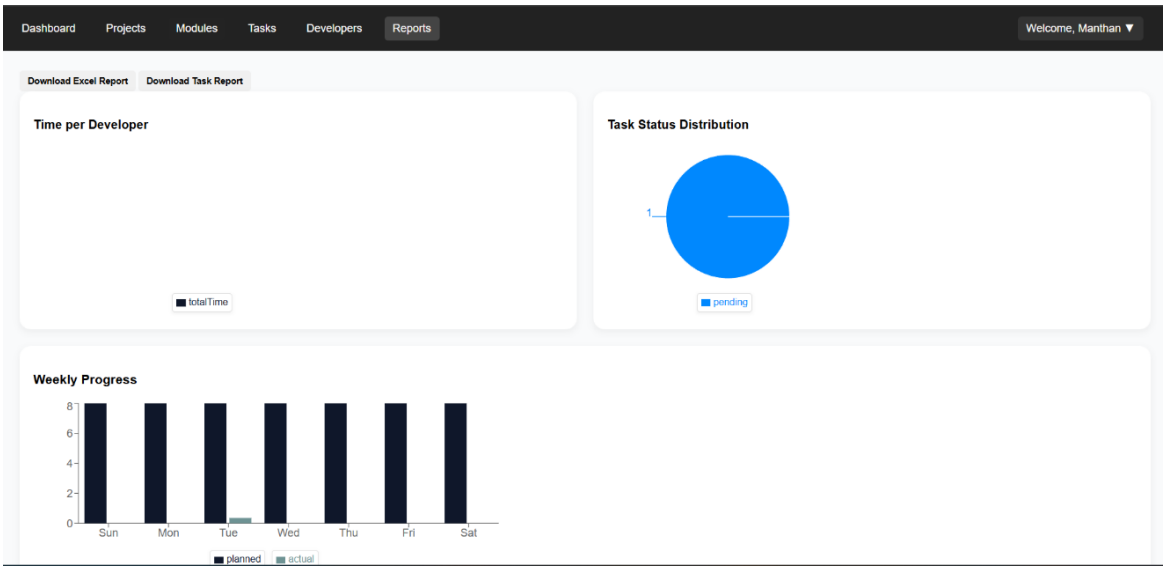

Figure 7.14 ADMIN REPORTS

Description :-

The Reports Page is a data-driven interface that visually summarizes team performance, task status, and work distribution. It's designed to help managers and developers monitor productivity and task flow through real-time graphical insights.

Navigation & Layout

The navigation bar remains consistent across the system, with Reports highlighted as the active section. The top-right corner shows a dropdown menu labeled "Welcome, Manthan", indicating a logged-in user/admin role.

Report Downloads

At the top of the page, two buttons provide quick data exports:

- **Download Excel Report** – Exports overall task/project stats in Excel format.
- **Download Task Report** – Likely exports task-specific details, potentially including time logs, status, developer names, etc.

These features allow offline analysis and backup of progress records.

Visual Report Sections

The page is split into three main report areas:

1. Time per Developer

- **Type**: Bar chart
- **Metric**: Displays the total time logged per developer under the label totalTime.
- **Observation**: Currently, the chart appears empty—suggesting that no developers have logged time yet or the data hasn't been fetched.

2. Task Status Distribution

- **Type**: Pie Chart
- **Insight**: Shows the current distribution of tasks based on their status.
- **Observation**: The chart is entirely blue, labeled pending, indicating all tasks in the system are currently in the pending state. This provides a quick alert to project managers that no tasks are marked as completed or in-progress.

3. Weekly Progress

- **Type**: Bar Graph
- **Metric**: Compares planned vs actual work across the week (Sunday to Saturday).
- **Insight**:
    - Each day shows a dark bar (planned) and a light bar (actual) for comparison.
    - Most days have only the planned bars, with minimal or no actual time recorded.
    - This suggests work is scheduled, but actual execution/logging is lagging—indicating inefficiencies or delays.

Summary

The Reports Page offers a centralized snapshot of overall team activity and project health using three powerful visuals:

- Developer time tracking
- Task status breakdown
- Planned vs. Actual weekly work

7.15 PROJECT MANAGER DASHBOARD

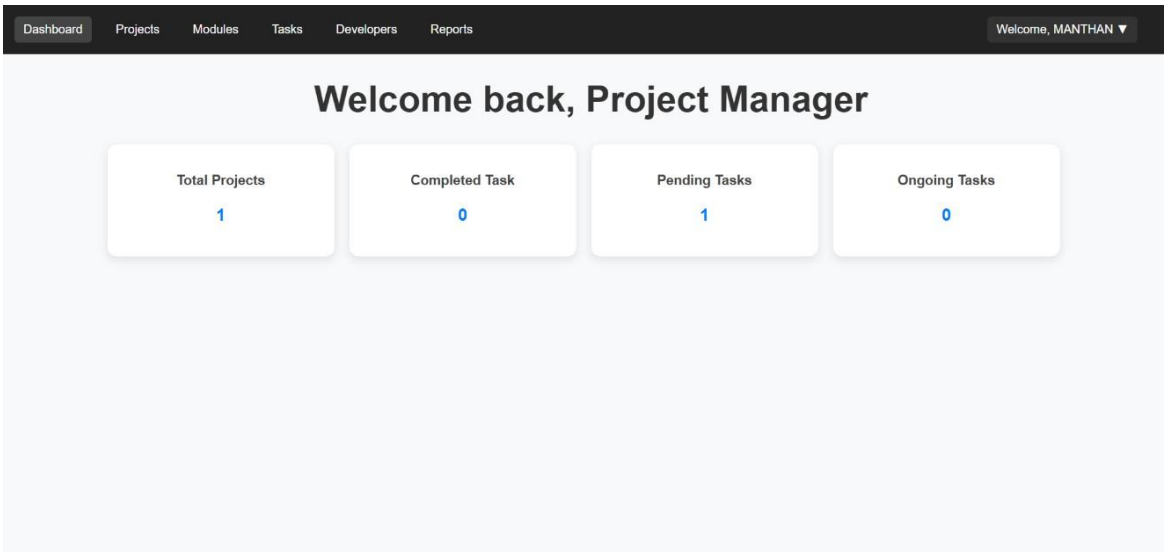

Figure 7.15 PROJECT MANAGER DASHBOARD

Description :-

The Dashboard Page is the central hub for the Project Manager in the Time Tracker system. It offers an at-a-glance overview of project and task metrics to help managers keep track of team progress and workloads.

Welcome Message

The top of the page greets the logged-in user with:

"Welcome back, Project Manager"

This confirms the user's role and gives a personalized touch to the interface.

Key Metrics Cards

There are four summary cards aligned horizontally, each giving a count of specific data:

1. **Total Projects**
   - **Value**: 1
   - Indicates there is one active or created project in the system.
2. **Completed Task**
   - **Value**: 0
   - No tasks have been marked as completed yet.
3. **Pending Tasks**
   - **Value**: 1
   - One task is currently in the pending state and not yet started.
4. **Ongoing Tasks**
   - **Value**: 0
   - No tasks are currently marked as "in progress".

Each card uses a clean white container with a blue numeric highlight, visually making key stats pop out for immediate recognition.

Navigation Bar

The top nav bar allows smooth switching between different system views:

- Dashboard
- Projects
- Modules
- Tasks
- Developers
- Reports

On the right side, it shows:

Welcome, MANTHAN ▼
Indicating the current logged-in user's identity with a dropdown (probably for profile or logout).

Summary

The Dashboard Page acts as a quick insight center showing:

- How many projects exist
- Task distribution across statuses (Pending, Completed, Ongoing)

This snapshot helps project managers immediately identify gaps (e.g., no tasks are completed or ongoing) and follow up accordingly.

7.16 PROJECT MANAGER PROJECTS

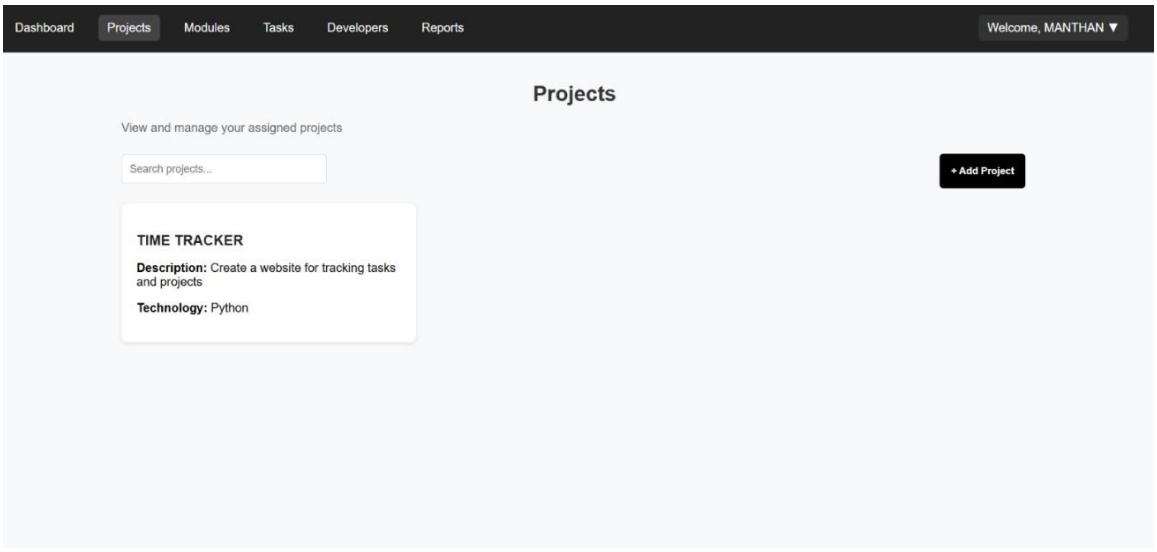

Figure 7.16 PROJECT MANAGER PROJECTS

Description :-

This is the Projects Module interface from your Time Tracker application. It serves as the central point for managing all projects within the system, providing an organized overview for users such as project managers, team leads, and developers.

Here's a detailed breakdown of the interface and its functionality:

Purpose:

To view, manage, and create new software development projects. Each project serves as a container for associated modules, tasks, developers, and time logs.

Navigation Context:

At the top, you'll see a consistent dark navigation bar, where the current active section is highlighted as Projects. Other options available include:

Dashboard , **Modules** , **Tasks** , **Developers** , **Reports**

Top-right:

- **Welcome, MANTHAN ▼** – Shows the logged-in user, allowing profile/account management.

Project List Panel:

- **Search Functionality**: A search input allows users to filter and locate projects quickly by typing project names or keywords.
- **Project Card (Example: TIME TRACKER)**:
    - **Project Name**: TIME TRACKER (bold)
    - **Description**: Create a website for tracking tasks and projects This gives context to the purpose and goal of the project.
    - **Technology**: Python
      Indicates the tech stack or primary language used — helpful for assigning developers and organizing modules.

Add Project Button:

- **Label**: + Add Project
- **Style**: Prominent black button on the right-hand side
- **Function**: Likely triggers a modal or form allowing users to:
    - Enter project name, description, technology stack
    - Assign developers or set deadlines

Inferred Functionalities & Benefits:

1. **Centralized Management**: View all projects in one place with quick access to related data.
2. **Scalability**: Ideal for teams working on multiple software projects or client-based work.
3. **Technology Tagging**: Helps filter or assign work based on skillsets.
4. **Simple UI**: Clean layout improves usability for both technical and non-technical users.
5. **Search Optimization**: Search bar boosts productivity by reducing navigation time.

Suggestions for Enhancement:

- **Add Filters**: Filter by technology, status (active/inactive), deadline, etc.
- **Project Status**: Show visual status indicators (e.g., in-progress, completed).
- **Team Assignments**: Display assigned developers on the project card.
- **Deadlines**: Add due dates to track time-sensitive projects.
- **Clickable Project Cards**: Link to a detailed dashboard view per project.
- **Pagination or Scroll**: For organizations with many projects, paginated display or infinite scroll will be helpful.

Use Case Summary:

This Projects Module is the foundational area of the Time Tracker system where:

- New projects are registered.
- Existing projects are tracked.
- Project data acts as a parent entity for modules, tasks, reports, and time entries.

7.17 PROJECT MANAGER MODULES

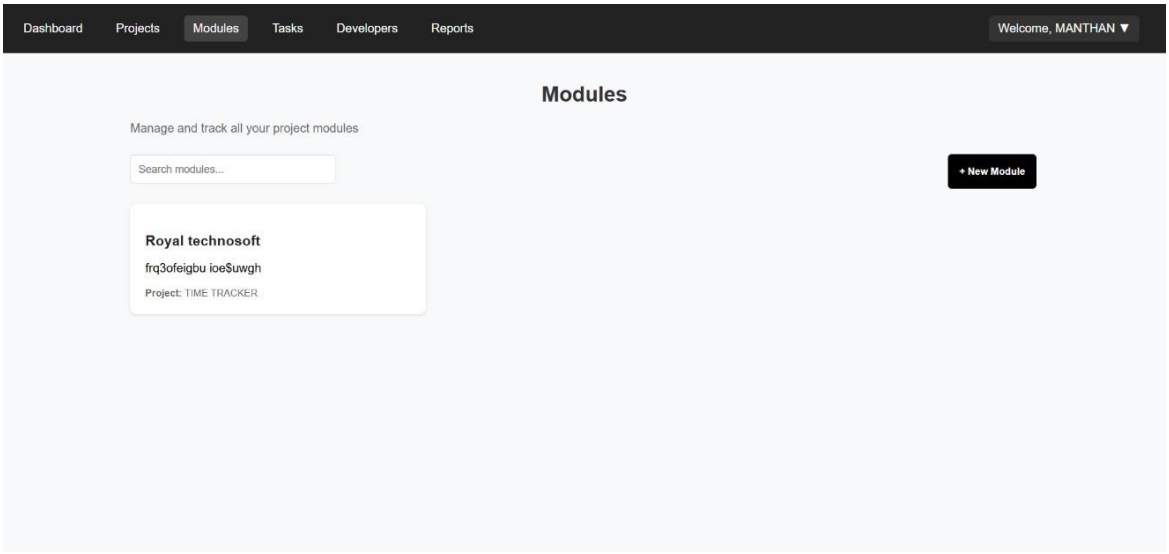

Figure 7.17 PROJECT MANAGER MODULES

Description :-

This is a Modules Management Interface of a web-based project management system, specifically part of the Time Tracker application. Here's a detailed breakdown of the screen and its components:

Page Overview:

- **Page Title:** "Modules"
- **Primary Purpose:** This interface is used to manage and track project modules—logical components or sections within a larger project, likely categorized by features or development stages.

Top Navigation Bar (Dark Theme):

This fixed top menu allows for seamless navigation across the platform's major areas:

- **Dashboard** – Likely the homepage with a summary of activity.
- **Projects** – Takes the user to a list or overview of ongoing projects.
- **Modules** – The current section being viewed; visibly highlighted to show it's active.
- **Tasks** – Possibly a breakdown of assigned or ongoing tasks within modules/projects.
- **Developers** – A section to manage developers, their profiles, or assignments.
- **Reports** – Could be used for generating visual or tabular analytics regarding time, tasks, or performance.

On the far right:

- **Welcome, MANTHAN ▼** – Indicates the logged-in user (probably with a dropdown menu for profile, settings, or logout).

Main Body:

1. Header Section

- **Title:** Modules (centered, bold, large font).
- **Subheading (left aligned):**
  - "Manage and track all your project modules" – providing users with context on this page's function.

2. Search Bar

- A module-specific search field to quickly locate a module by name. This is useful when handling large numbers of modules.

3. Module Card (Example Listed):

A single module card is displayed, containing:

- **Module Name:** "Royal technosoft"
- **Description (Random Text):** "frq3ofeibigb ioe$uwgh" – Appears to be placeholder or junk text; may not be finalized.
- **Project Association:** "Project: TIME TRACKER" – Suggests this module belongs to the "Time Tracker" project.

The card is styled with a white background and slight shadow, giving it a raised appearance against the light gray page background.

4. Add Module Button

- Located to the right of the page, it's a black button labeled:
  - + New Module
- This is likely used to create a new module. Clicking this would presumably open a form to input details such as the module name, description, associated project, deadlines, etc.

Functionality Implied:

1. **Module Management**:
   - Users can add, view, and search modules.
   - Each module links to a project and might include details like deadlines, assigned developers, and progress.
2. **Project-Level Navigation**:
   - The top menu suggests a hierarchy: Projects → Modules → Tasks.
3. **User Roles**:

- Given the "Welcome, MANTHAN" tag, the system supports user authentication and probably role-based access control (admin, manager, developer, etc.).

Possible Enhancements (Based on UI UX Best Practices):

- **Module Status/Progress Bar**: Indicating whether a module is in progress, complete, or delayed.
- **Action Buttons on Module Card**: Like “Edit”, “Delete”, or “View Details”.
- **Pagination/Scroll Management**: Useful if the number of modules grows.
- **Better Descriptions**: Placeholder text (like "frq3ofeibigb") should be replaced with meaningful summaries or tags.

7.18 PROJECT MANAGER TASKS

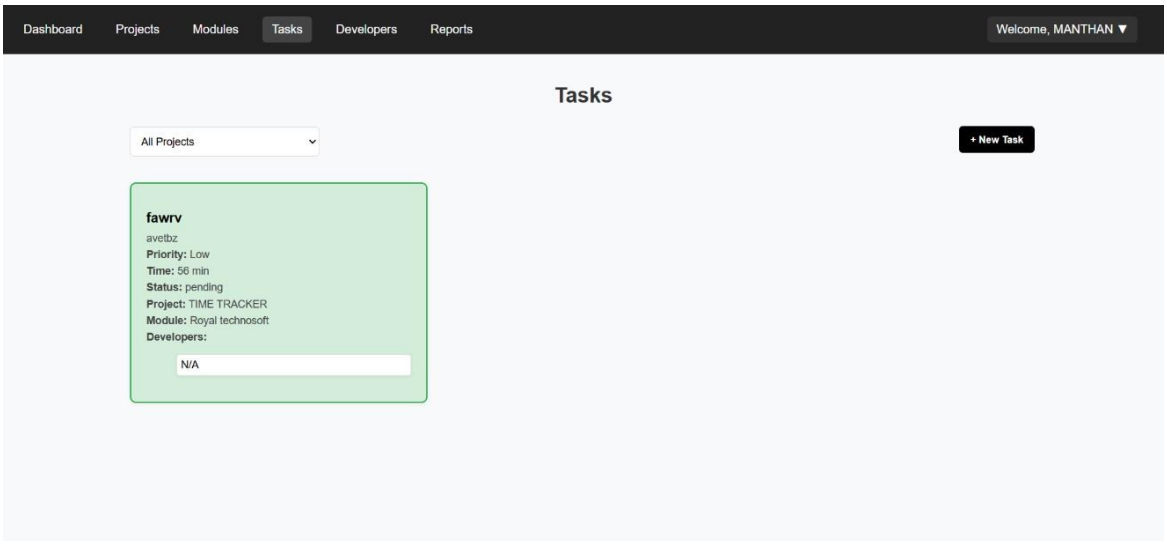

Figure 7.18 PROJECT MANAGER TASKS

Description :-

This is the Tasks Management Interface from the same Time Tracker web application shown earlier. The interface allows users to view and manage tasks linked to specific projects and modules. Here's a detailed breakdown of the elements and functionality represented in the screenshot:

Page Overview:

- **Page Title:** Tasks
- **Purpose:** Manage, view, and create tasks that belong to various projects and their associated modules.

Top Navigation Bar (Dark Theme):

Just like in the Modules page, the top bar provides access to:

- **Dashboard**
- **Projects**
- **Modules**
- **Tasks** (highlighted to indicate the current view)
- **Developers**
- **Reports**

On the top-right:

- **Welcome, MANTHAN ▼** – Currently logged-in user (with expected dropdown options).

Main Page Layout:

1. Dropdown Filter:

- **Label:** All Projects
- This dropdown allows users to filter tasks based on the selected project. Only tasks from the selected project will be displayed.

2. Task Card Displayed:

There's a single green-tinted card representing a task with the following details:

- **Title:** fawrv
- **Assigned To (likely a tag or label):** avethz
- **Priority:** Low
- **Time:** 56 min – Estimated or logged time for completion.
- **Status:** pending – Indicates the task hasn't been completed yet.
- **Project:** TIME TRACKER – Links the task to a specific project.
- **Module:** Royal technosoft – The module under which this task falls.
- **Developers:** NA – No developer assigned currently.

The green background likely symbolizes a low-priority or pending status, potentially color-coded for quick status recognition.

3. New Task Button:

- Located on the right side, styled in black:
  - + New Task
- Clicking this would likely open a form to create a new task with fields such as:
  - Task title
  - Assigned developer
  - Priority (Low/Medium/High)
  - Estimated time
  - Project/module association
  - Task status

Implied Functionalities:

- **Task Filtering:** Based on project selection.
- **Task Metadata Management:** Each task includes time estimates, priority levels, status, developer assignment, and organizational linkage.

- **Assignment Tracking:** The developer section is empty (NA), suggesting a future enhancement to assign developers dynamically.

Suggested Enhancements:

1. **Clickable Cards:** To allow detailed view/edit of each task.
2. **Color Codes for Priority or Status:**
    - Green: Low
    - Yellow: Medium
    - Red: High or Overdue
3. **Developer Dropdown/Auto Assign Field**
    - If no one is assigned, show an "Assign Developer" button.
4. **Time Logging Integration:**
    - Start/Stop button to track time dynamically instead of manual input.
5. **Sorting and Filtering:**
    - By priority, status, time, or module.
6. **Status Update Toggle:**
    - Allow inline marking as "Complete", "In Progress", etc.

Use Case Summary:

The task management page helps project managers or team leads:

- Monitor active tasks,
- Track time against deliverables,
- Maintain workflow visibility,
- Coordinate work within defined modules of larger projects.

7.19 PROJECT MANAGER DEVELOPERS

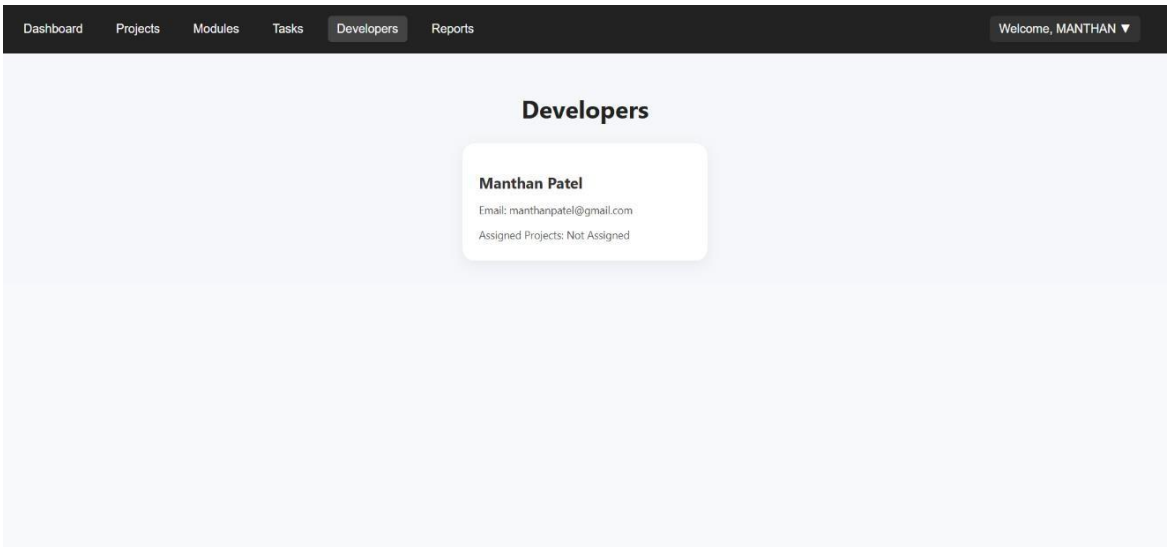

Figure 7.19 PROJECT MANAGER DEVELOPERS

Description :-

This screen represents the Developers Management Interface of the Time Tracker project management system. It's designed to show the list of developers registered in the system along with their basic details and assignment status.

Here's a detailed breakdown:

Page Overview:

- **Title:** Developers
- **Purpose:** To list all developers who are part of the platform and show their email and assigned projects.

Top Navigation Bar (Dark Theme):

Consistent with the rest of the system:

- **Dashboard**
- **Projects**
- **Modules**
- **Tasks**
- **Developers** (highlighted, showing it's the active tab)
- **Reports**

Top-right:

- **Welcome, MANTHAN ▼** – Logged-in user details.

Developer Card Displayed:

A single developer profile is shown, formatted as a card:

⬭ **Name: Manthan Patel**

⬭ **Email: manthanpatelr@gmail.com**

⬭ **Assigned Projects: Not Assigned**

This suggests that while the developer is registered in the system, they are currently not assigned to any projects.

The card is neatly styled with:

- A soft white background.
- Slight border-radius and shadow, giving a clean and minimal aesthetic.
- Simple, readable typography.

Inferred Functionalities:

- **Developer Directory:** Likely supports multiple developer cards (only one shown here).
- **Project Assignment Tracking:** Helps managers see which developers are unassigned or overloaded.
- **Email for Contact:** Indicates the system supports contacting developers directly (or notifying them of task/project assignment).

Possible Enhancements:

1. **Assignment Feature:**
    - A button or dropdown to assign a developer to a project/module directly from this screen.
2. **Role and Skill Tags:**
    - Add info such as developer roles (Frontend, Backend, Full Stack) or skills (React, FastAPI, MongoDB).
3. **Task Load Indicator:**
    - Show how many tasks are currently assigned to each developer.
4. **Clickable Card:**
    - Clicking the developer could show a detailed profile view with:
        - Assigned modules/tasks
        - Time logs

        - Performance/Activity history
5. **Search or Filter Option:**
    - Useful when there are many developers in the system.

Use Case Summary:

This interface helps team leads and project managers:

- View who is available for work.
- Monitor developer participation.
- Balance workload by identifying unassigned members.
- Maintain a structured team management flow across projects.

7.20 PROJECT MANAGER REPORTS

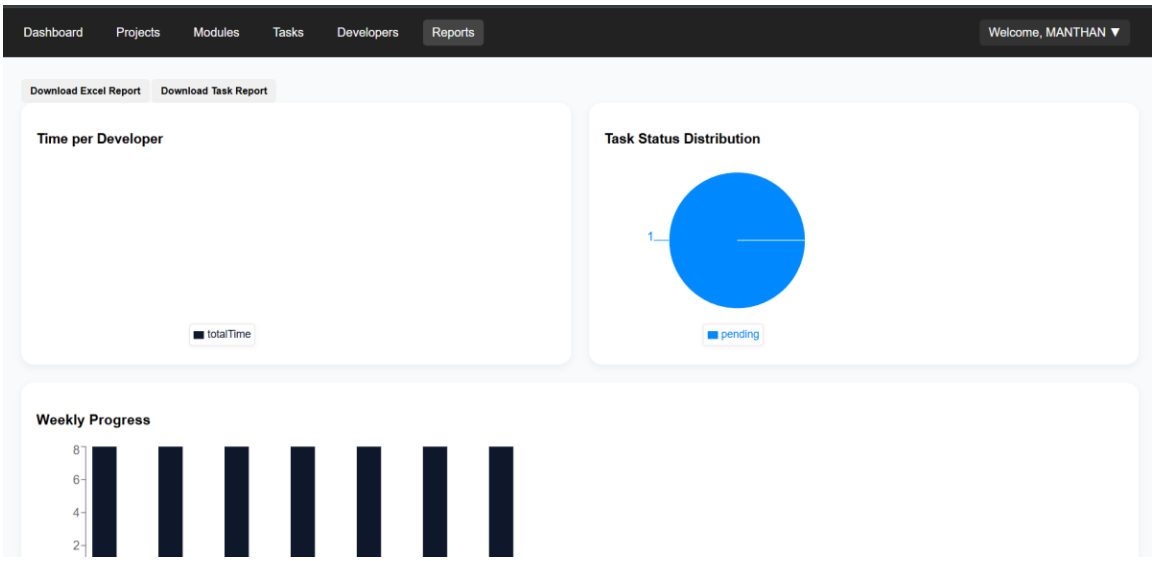

Figure 7.20 PROJECT MANAGER REPORTS

Description :-

This is the Reports Dashboard of the Time Tracker project management system. It provides a visual summary of productivity, task distribution, and developer engagement metrics using charts and reports.

Here's a comprehensive breakdown of its structure and purpose:

Page Overview:

- **Title:** Reports
- **Purpose:** To display analytical data visualizations and downloadable reports related to task progress, developer productivity, and overall project performance.

Top Navigation Bar (Dark Theme):

This remains consistent across the platform:

- **Dashboard**
- **Projects**
- **Modules**
- **Tasks**
- **Developers**
- **Reports** (highlighted as the current active tab)

Top-right:

- **Welcome, MANTHAN ▼** – Indicates the logged-in user.

Report Download Buttons:

At the top of the content area:

- Download Excel Report – Likely exports data like time logs, developer activity, or task stats in .xlsx format.
- Download Task Report – Likely focused on task details like status, duration, assigned personnel, and deadlines.

These buttons suggest integration with backend-generated downloadable reports, useful for audits or external analysis.

Visual Report Sections:

There are three key data visualizations shown on this page:

1. Time per Developer (Bar Chart)

- **Chart Type:** Bar Graph (vertical)
- **Label:** totalTime – This represents the total time logged by each developer.
- **Usage:** Helps managers understand how much effort each developer is contributing.
- **Current State:** Appears mostly empty, possibly due to:
  - Only one developer with limited data.
  - No time logs entered yet.

2. Task Status Distribution (Pie Chart)

- **Chart Type:** Pie Chart
- **Data:** Entire chart is blue, representing "pending" status.
- **Usage:** This helps visualize how many tasks are in each status category (e.g., pending, in progress, completed).
- **Current Insight:** All tasks in the system are currently pending — a potential alert that no progress is being made or no statuses have been updated.

3. Weekly Progress (Bar Chart)

- **Chart Type:** Bar Graph
- **Y-axis Scale:** From 0 to 8 (likely number of tasks or hours per week)
- **Usage:** Tracks weekly task completion or developer activity, helping project managers identify trends or performance bottlenecks.
- **Current Insight:** The bars suggest consistent but static progress, possibly due to placeholder or initial seed data.

Inferred Features and Functional Value:

- **Project Monitoring:** Visual indicators help understand project and team status at a glance.
- **Developer Workload Assessment:** Identify overworked or under-utilized team members.
- **Status Tracking:** The pie chart gives instant awareness about task flow.
- **Export Functionality:** Provides tools for documentation, reporting to stakeholders, or project reviews.

Suggestions for Enhancement:

1. **More Granular Filters:**
    - Allow filtering by project, date range, developer, or module.
2. **Status Trend Over Time:**
    - Line chart to show how tasks move between statuses weekly or monthly.
3. **Developer Performance Summary:**
    - KPI-style cards showing tasks completed, average time per task, etc.
4. **Interactive Charts:**
    - Hover effects, clickable sections (e.g., clicking "pending" shows list of pending tasks).
5. **Alerts/Notifications:**
    - Trigger warnings if tasks are overdue or progress is stagnant.

Use Case Summary:

The Reports Module serves as a critical dashboard for:

- Team leads and managers to review performance.
- Upper management to assess overall productivity.
- Generating documentation or exporting time logs and task statuses for audits or reviews.

7.21 DEVELOPER DASHBOARD

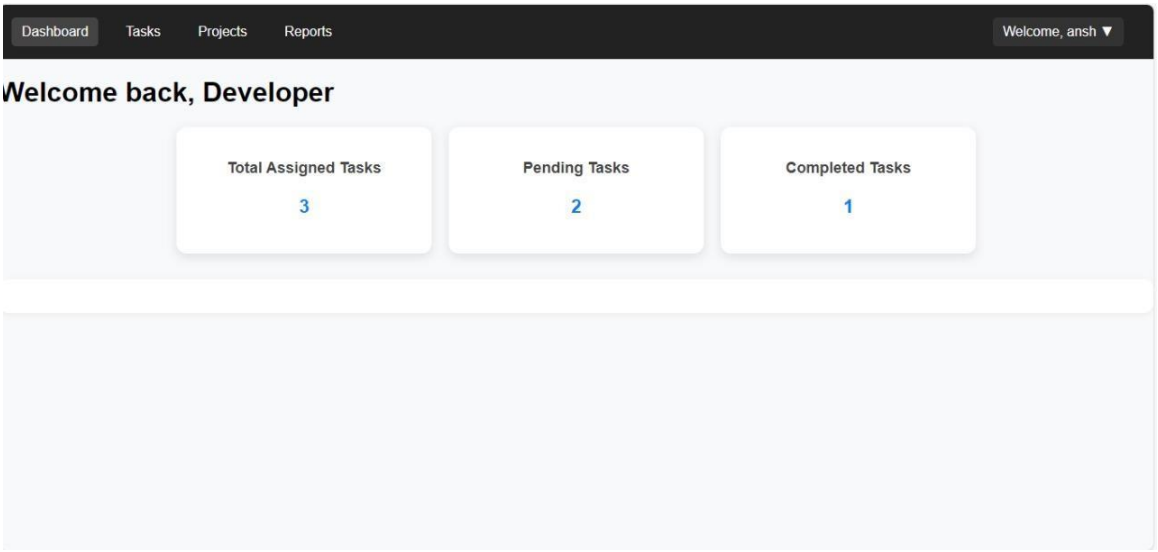

Figure 7.21 DEVELOPER DASHBOARD

Description :-

This screenshot showcases the Developer Dashboard of your Time Tracker application — the landing page a developer sees after logging in.
Here's a detailed breakdown of this dashboard interface:
Developer Dashboard – Summary
Primary Purpose:
To provide developers with a clear overview of their task assignments and statuses — enabling better task management and productivity tracking.

Top Navigation Bar:
Consistent with the rest of the application, it includes:

- **Dashboard** (active section)
- **Tasks**
- **Projects**
- **Reports**

Top-right corner:

- **User Info**: Welcome, ansh ▼ — Identifies the logged-in developer.

Greeting Section:

- **Message**: Welcome back, Developer
- Personal, friendly greeting to reinforce user role and login status.

Task Summary Cards:

Three neatly styled summary cards provide a snapshot of the developer's current workload:

1. **Total Assigned Tasks**
    - **Count**: 3
    - Shows the total number of tasks assigned to the developer.
2. **Pending Tasks**
    - **Count**: 2
    - Reflects tasks that are in progress or yet to be started.
3. **Completed Tasks**
    - **Count**: 1
    - Indicates how many tasks have been marked as finished.

These cards likely update dynamically based on task status changes.

Inferred Functionalities & Behavior:

- **Click Navigation**: Clicking on the counts (e.g., "2" under Pending Tasks) may redirect to a detailed list view filtered by task status.
- **Live Updates**: Task counts may refresh in real time or at regular intervals.
- **Role-Based UI**: Since this is the *developer* dashboard, the view is streamlined and personalized — no admin controls or management tools are visible.

Suggestions for Improvement:

- **Add Progress Visualization**: Include a pie chart or progress bar showing task status breakdown visually.
- **Task Deadline Alert**: Display urgent tasks due soon.
- **Recent Activity Feed**: Show recent updates like new assignments or completed tasks.
- **Filter Options**: Allow filtering tasks by project/module/date range directly on the dashboard.
- **Navigation Shortcuts**: Buttons for "View My Tasks", "Log Time", etc., to speed up access.

Use Case Summary:

The **Developer Dashboard** functions as a quick-glance control panel for individuals to:

- Monitor workload
- Track progress
- Stay updated on assigned responsibilities

7.22 DEVELOPER TASKS

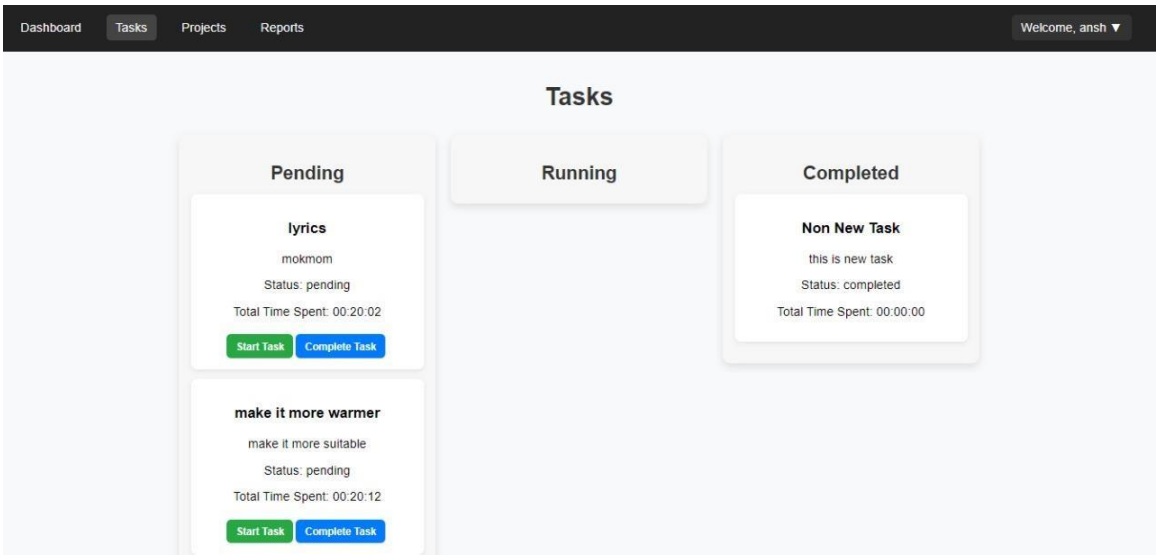

Figure 7.22 DEVELOPER TASKS

Description :-

This screen represents the Developer Task Management View in your Time Tracker application — where a developer can directly manage their assigned tasks.

Developer Tasks View – Breakdown

Location:

- From the top navigation, the Tasks tab is selected.
- Developer logged in: ansh.

Task Categorization Tabs:

The screen is divided into three task status categories, helping organize tasks by progress state:

1. **Pending** – Tasks yet to be started.
2. **Running** – (Currently no tasks in this section).
3. **Completed** – Tasks finished by the developer.

Task Cards per Section:

Pending Tasks:

Each card includes:

- **Title**: e.g., lyrics, make it more warmer
- **Description**: Short explanation of the task.
- **Status**: Always pending for this section.
- **Total Time Spent**: Actively tracked time (e.g., 00:20:02).
- **Action Buttons**:
    - Start Task: Likely changes the status to "Running".
    - Complete Task: Directly marks task as "Completed".

This dual-action design supports both immediate completion or gradual work tracking.

Running Tasks:

- This section is currently empty, but presumably when a task is started, it moves here with active tracking enabled.

Completed Tasks:

Contains tasks marked as done:

- Example: Non New Task
- Status: completed
- Time spent: 00:00:00

Shows task history or review data for the developer.

Inferred Functionality:

- Likely real-time updates on task duration once the status is "Running".
- Actions update both backend state and UI using status transitions.
- Switching status might persist via FastAPI backend and MongoDB storage.

Suggestions for Enhancement:

1. **Add Task Timers**:
    - Visual countdown or stopwatch for running tasks.
    - Pause/Resume button for real-world time tracking.
2. **Improve Running Section UX**:
    - Show active tasks with large timer, and control panel.
3. **Confirmation Prompts**:
    - Before marking a task "complete", confirm via modal dialog.
4. **Filter by Project/Module**:
    - Useful for developers working on multiple concurrent modules.
5. **Search & Sort Tasks**:
    - By priority, date, estimated duration, etc.
6. **Color Code by Priority**:
    - Add visual cues for task importance.

7.23 DEVELOPER PROJECTS

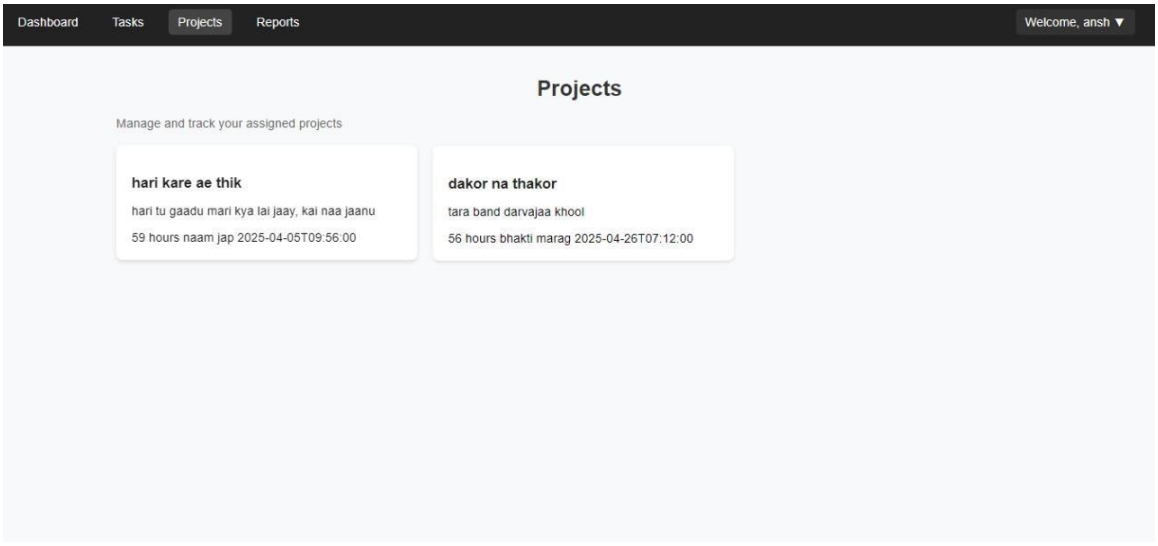

Figure 7.23 DEVELOPER PROJECTS

Description :-

Developer Projects View – Breakdown

Location:

- From the top navigation, the Projects tab is selected.
- Developer logged in: ansh.

Project Categorization (Implicit):

- Unlike the "Tasks" view with explicit "Pending," "Running," and "Completed" tabs, this "Projects" view categorizes projects implicitly through individual Project Cards. Each card seems to represent a distinct project.

Project Cards:

Each card provides a summary of a project. The visible information within each card includes:

- **Title (Bold Text):** This is the name of the project. Examples from the image are:
  - "hari kare ae thik"
  - "dakor na thakor"
- **Description (Regular Text):** A short piece of text associated with the project. This could be a brief description, a relevant phrase, or perhaps a current status message. Examples:
  - "hari tu gaadu man kya lai jaay, kai naa jaanu"
  - "tara band darvajaa khool"

- **Time-Related Information:** Each card displays information that seems to relate to time, potentially deadlines or progress tracking:
    - **Card 1:** "59 hours naam jap 2025-04-05T09:56:00"
    - **Card 2:** "56 hours bhakti marag 2025-04-26T07:12:00"
        - The numerical value followed by "hours" likely indicates a duration or time remaining.
        - The subsequent text ("naam jap," "bhakti marag") might be a label for this time tracking or a specific phase.
        - The timestamp in ISO 8601 format (YYYY-MM-DDTHH:MM:SS) likely represents a deadline, target end time, or a recorded completion time.

Inferred Functionality:

- **Project Overview:** The view provides a summarized overview of all projects assigned to the developer "ansh."
- **Potential for Project Details:** Clicking on a project card likely navigates the developer to a more detailed view for that specific project, where they can manage tasks, view progress, and access other relevant information.
- **Time Tracking Association:** The presence of time-related information within the project cards suggests that the application might allow for tracking time spent on specific projects or deadlines associated with them.
- **Implicit Status:** While not explicitly labeled, the different time-related information on each card might implicitly indicate the status or progress of the project. For example, a past deadline might suggest a completed or overdue project.

Suggestions for Enhancement (Based on the "Tasks" View Enhancements):

- **Visual Progress Indicators:** Incorporate visual cues like progress bars or color-coding to represent the completion status or urgency of projects.
- **Clearer Time Indicators:** Improve the clarity of the time-related information. Label whether it represents "Time Remaining," "Deadline," or "Total Time Spent."
- **Project-Specific Actions:** Add action buttons or links within each project card to allow developers to quickly access project details, add tasks, or update progress.
- **Filtering and Sorting:** Implement options to filter projects by status, due date, or other relevant criteria. Allow sorting of projects based on different parameters.
- **Search Functionality:** Add a search bar to easily find specific projects by title or keywords.
- **Project Grouping:** If developers work on multiple modules or categories of projects, consider allowing grouping or categorization of projects in the view.
- **Badges or Notifications:** Display badges or notifications on project cards to indicate urgent deadlines or overdue items.

7.24 DEVELOPER REPORTS

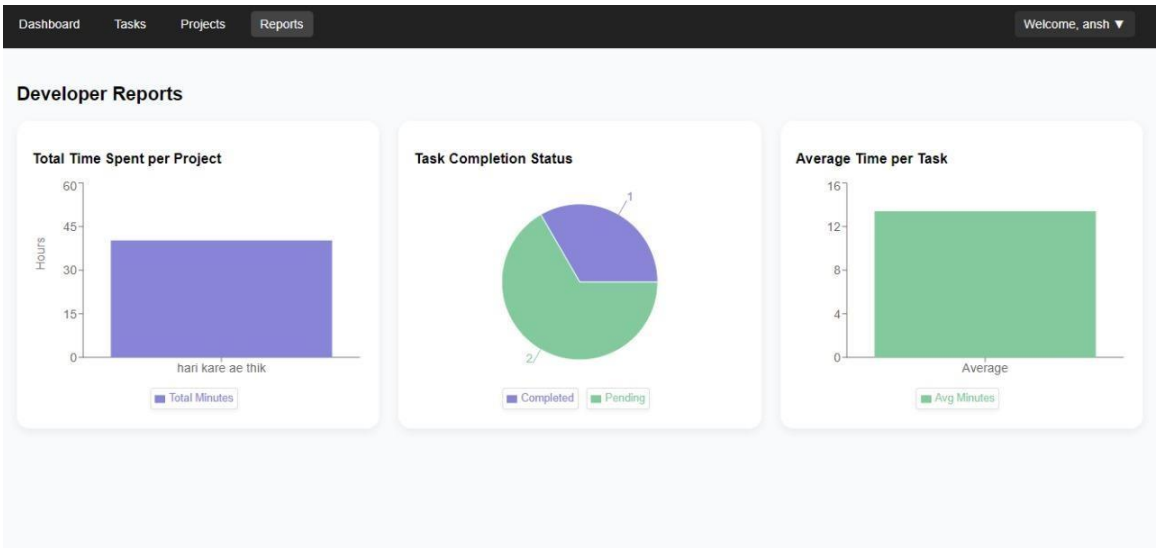

Figure 7.24 DEVELOPER REPORTS

Description :-

Developer Reports View – Breakdown

Location:

- From the top navigation, the Reports tab is selected.
- Developer logged in: ansh.

Report Sections (Visualized as Cards):

The screen is divided into three distinct report sections, each presented within a separate card, providing different insights into the developer's activity:

- **Card 1: Total Time Spent per Project**
  - **Title:** "Total Time Spent per Project"
  - **Visualization:** A vertical bar chart.
    - **Y-axis Label:** "Hours," ranging from 0 to 60 with increments of 15.
    - **X-axis Label:** "hari kare ae thik," which corresponds to one of the project titles seen in the "Projects" view. This indicates that the chart is displaying data for this specific project.
    - **Bar:** A single, solid purple bar extending up to approximately the 40-hour mark (a bit below 45).
    - **Legend:** A small purple square labeled "Total Minutes." This is inconsistent with the Y-axis label "Hours." It's likely a labeling error, and the bar probably represents total hours spent.

    - **Interpretation:** This chart visually represents the total amount of time the developer "ansh" has spent working on the project "hari kare ae thik." Based on the bar height, it's roughly 40 hours.
- **Card 2: Task Completion Status**
    - **Title:** "Task Completion Status"
    - **Visualization:** A pie chart.
        - **Slices:** The pie chart is divided into two slices: a larger light green slice and a smaller light purple slice.
        - **Labels:** Numbers "2" is associated with the light green slice, and "1" is associated with the light purple slice. These likely represent the count of tasks in each status.
        - **Legend:**
            - A purple square labeled "Completed." This likely corresponds to the smaller slice (labeled "1").
            - A green square labeled "Pending." This likely corresponds to the larger slice (labeled "2").
    - **Interpretation:** This pie chart shows the proportion of completed versus pending tasks for the developer. There are 2 pending tasks and 1 completed task, meaning a larger portion of the developer's tasks are currently pending.
- **Card 3: Average Time per Task**
    - **Title:** "Average Time per Task"
    - **Visualization:** A vertical bar chart.
        - **Y-axis Label:** "Average," ranging from 0 to 16 with increments of 4.
        - **X-axis Label:** "Average," indicating that this chart shows a single average value.
        - **Bar:** A single, solid light green bar extending up to approximately the 12-minute mark (a bit below 16).
        - **Legend:** A small green square labeled "Avg Minutes." This indicates that the bar represents the average time spent per task in minutes.
    - **Interpretation:** This chart displays the average time taken by the developer "ansh" to complete their tasks. The average time appears to be around 12 minutes per task.

Inferred Functionality:

- **Progress Monitoring:** The "Reports" view provides a visual summary of the developer's progress in terms of time spent on projects and task completion.
- **Performance Insights:** The "Average Time per Task" report offers a metric for assessing the developer's efficiency or the average effort required for their tasks.
- **Project-Specific Reporting:** The "Total Time Spent per Project" report allows the developer and potentially managers to understand the effort distribution across different projects.
- **Real-time or Periodic Updates:** These reports likely update periodically or in near real-time as the developer works on tasks and projects.

- **Potential for More Detailed Reports:** This view might be a summary, with the possibility to drill down into more detailed reports or filter data by date range, project, or task status.

Suggestions for Enhancement (Based on Previous Enhancements):

- **Date Range Filtering:** Allow the developer to filter the reports by specific date ranges (e.g., last week, this month, custom range) for more granular analysis.
- **Project Selection:** Enable the developer to select specific projects to view detailed reports for, especially if they are working on multiple projects.
- **Task-Specific Time Breakdown:** For the "Average Time per Task," consider providing a breakdown of average times for different types of tasks or within specific projects.
- **Trend Analysis:** Implement line charts or other visualizations to show trends in time spent or task completion over time.
- **Comparison Metrics:** Introduce comparison metrics, such as comparing the developer's average time per task to team averages or estimated times.
- **Interactive Elements:** Make the charts interactive, allowing developers to hover over elements for more specific data points.
- **Error Correction:** Fix the inconsistency in the "Total Time Spent per Project" legend ("Total Minutes" vs. Y-axis "Hours").
- **More Project-Specific Reports:** If there were multiple projects with tracked time, the "Total Time Spent per Project" card should ideally display data for all of them, perhaps as multiple bars.

Summary:

The "Developer Reports" view provides a visual overview of the developer "ansh's" activity through three key reports: total time spent on a specific project, task completion status (pending vs. completed), and average time taken per task. These reports offer insights into the developer's workload, progress, and efficiency. Enhancements such as date range filtering, project selection, and more detailed breakdowns could further improve the value and usability of this reporting section.

8. Implementation Details

The implementation of the Time Tracker system follows a modular, service-oriented architecture to ensure separation of concerns, maintainability, and scalability. Each component—frontend, backend, database, and admin interface—is designed as an independent module that interacts through secure APIs.

8.1 Frontend

The frontend is built using React.js with Vite, which allows for faster builds and hot module replacement during development. The user interface is clean and responsive, utilizing Tailwind CSS for component-based styling. It includes the following key components:

- Login/Register Pages – User authentication and onboarding.
- Dashboard – Role-specific views for Admin, Project Manager, and Developer.
- Task Management – Task creation, status update, and listing interfaces.
- Time Tracking UI – Timer widgets and manual entry forms.
- Reports Page – Graphs, tables, and PDF download buttons for productivity insights.

Axios is used to make secure, asynchronous calls to the backend APIs.

8.2 Backend API (FastAPI)

The core business logic and data handling are implemented using FastAPI (Python). This framework is chosen for its:

- High performance and asynchronous I/O capabilities.
- Automatic documentation (Swagger UI).
- Strong support for JSON and data validation using Pydantic.

FastAPI handles operations such as: CRUD operations for users, tasks, and time logs. Calculation of time spent on tasks and generating analytics.

Each API route is secured and validated using middleware to ensure role-based access control.

8.3 Database (MongoDB)

The system uses MongoDB, a NoSQL document-oriented database that provides:

- Schema flexibility for handling various types of logs and task structures.
- Scalability for future integration with mobile apps or team-level reports.
- Easy JSON-like structure which aligns well with FastAPI and React.js data flow.

Key collections include:

- users

- projects
- tasks
- time_logs
- reports

MongoDB Atlas or a local instance can be used depending on deployment requirements.

8.4 Authentication System

Roles supported:

- Admin: Access to user reports, team analytics, and user management.
- Project Manager: Can create projects, assign tasks, and monitor developers.
- Developer: Can log time, update task status, and view personal dashboards.

8.5 Containerization (Optional)

The system optionally supports Docker-based containerization to simplify:

- Deployment in isolated environments.
- Local development with docker-compose for orchestration.

Docker containers are configured with:

- Separate networks for API and database communication.
- Shared volumes for logging and config.
- Auto-restart policies for development stability.

8.6 API Documentation

All backend routes are auto-documented using Swagger (via FastAPI) for testing and reference purposes. This includes:

- Auth routes
- Task management
- Time logging
- Report generation

This implementation ensures that the Time Tracker system remains:

- Modular (easy to test and maintain)
- Scalable (support future features like billing, mobile apps)
- Fast and Responsive (using modern UI/UX practices)

9. Testing

Test Type	Purpose	Methods	Tools	Benefits
Unit Testing	Validate individual functions or components in isolation.	Test each function/class with edge cases, boundary conditions, and typical scenarios.	- pytest (Python) - Jest (React)	Catches bugs early, supports refactoring, ensures individual units function correctly.
Integration Testing	Ensure correct interactions between frontend and backend systems.	Simulate real user actions (e.g., form submission, API requests) and validate data flow and state transitions.	- Postman - Selenium	Ensures modules work together, reduces risk of system-level bugs, validates end-to-end workflows.
Regression Testing	Ensure that new changes do not break existing functionality.	Run full or partial tests after every sprint to check that existing features remain functional.	- Automated Regression Suites with Jest, pytest	Detects side effects of changes, maintains stability, prevents broken legacy features.
Performance Testing	Test the system's behavior under load and identify bottlenecks.	Simulate multiple users, monitor performance under load and stress, analyze response time and system stability.	- JMeter - Google Lighthouse	Ensures scalability and responsiveness, improves user experience under traffic.
Security Testing	Identify vulnerabilities and ensure data protection.	Test for SQL injection, XSS, CSRF, secure login, and validate RBAC (role-based access control).	- Fail2Ban - OWASP ZAP - SonarQube	Prevents unauthorized access, protects sensitive data, identifies security risks.
User Acceptance Testing (UAT)	Validate application from end-user perspective before release.	Manual testing by actual users based on real workflows; feedback collected and addressed before final release.	- Manual - Feedback via Google Forms, JIRA	Ensures the software meets user needs, enhances usability, reduces post-release issues.
Continuous Testing & Monitoring	Run automated tests on every	Automated tests triggered on code integration to run	- GitHub Actions -	Rapid bug detection, high code quality, and faster

Test Type	Purpose	Methods	Tools	Benefits
	code push as part of CI/CD.	unit, integration, regression, and performance checks.	Jenkins, CircleCI	deployments with fewer issues.

9.1 Testing Table

Testing Strategy for Time Tracker System :-

9.2 Unit Testing

Purpose: To test individual components or functions of the system in isolation. Application to Time Tracker:

- Test timer start/stop functions to ensure accurate time capture.
- Validate dropdown components for Project, Module, User, and Status.
- Check that task creation form correctly handles inputs, validations, and API calls.
- Ensure utility functions (like time formatting, status filters) work as expected.

Tools: Jest (for React), PyTest (for FastAPI backend)

9.3 Integration Testing

Purpose: To verify that different modules of the application work together correctly. Application to Time Tracker:

- Test the interaction between the frontend form (React) and the backend API (FastAPI) during task creation.
- Validate the connection between dropdown selections (e.g., selecting a project should dynamically show corresponding modules).
- Ensure timer data is correctly submitted and reflected in reports via API.
- Confirm that admin module addition reflects accurately in the task manager view.

Tools: React Testing Library, Postman/Newman (API testing), PyTest with HTTPX (FastAPI)

9.4. Regression Testing

Purpose: To ensure that new updates or bug fixes do not break existing functionality. Application to Time Tracker:

- After introducing new features like module management or filtering, verify that core features like task creation, timer logging, and report generation still function correctly.
- Re-run automated test suites after changes to the backend database schema or user roles.

Approach: Maintain a test case suite with automation scripts for repetitive checks.

9.5. Performance Testing

Purpose: To test the responsiveness, stability, and scalability of the system under different loads.
Application to Time Tracker:

- Simulate multiple users starting/stopping timers simultaneously and observe API latency.
- Check page load time of task dashboard when hundreds of tasks are loaded.
- Stress test report generation with heavy time log data.

Tools: JMeter, Locust (for backend load), Lighthouse (frontend performance)

9.6. Security Testing

Purpose: To identify and resolve vulnerabilities in the system.
Application to Time Tracker:

- Ensure secure login/signup (using bcrypt for password hashing).
- Validate role-based access control (RBAC) to prevent unauthorized access to admin or project data.
- Protect APIs from unauthorized POST/GET calls using token-based authentication (JWT).
- Prevent common attacks: SQL Injection, XSS, CSRF.

Tools: OWASP ZAP, Postman for token validation, manual access control testing

9.7. User Acceptance Testing (UAT)

Purpose: To validate that the system meets the business requirements and is usable by real end-users.
Application to Time Tracker:

- Involve developers, managers, and admins in testing real-world scenarios:
    - Creating tasks
    - Logging time
    - Viewing time logs and reports
    - Adding new modules and seeing them reflect immediately in dropdowns
- Gather feedback on usability, UI design, and feature completeness.

Output: Approval or change requests from stakeholders before final deployment.

9.8. Continuous Testing & Monitoring

Purpose: To ensure quality assurance throughout the development lifecycle and after deployment.
Application to Time Tracker:

- Set up CI/CD pipelines with automated test execution on every push (e.g., GitHub Actions).
- Use monitoring tools to track system health, response time, and uptime after deployment.
- Automatically run unit, integration, and regression tests during each deployment phase.

Tools: GitHub Actions / GitLab CI, Selenium for UI tests, Prometheus + Grafana for monitoring

10. User Manual

The Time Tracker System is designed to provide a seamless, intuitive experience to users based on their designated roles. This section outlines how users can access the system, utilize features tailored to their roles, and generate insightful analytics through dashboards and reports.

10.1 Accessing the System

To begin using the Time Tracker application:

1. Open a supported web browser (Google Chrome, Mozilla Firefox, etc.).
2. Navigate to the application's login page.
3. Enter your registered email address and password.
4. Click the Login button.
5. Upon successful authentication, the system automatically detects your role (Admin, Project Manager, or Developer) and redirects you to your role-specific dashboard.

Note:
New users must first register through the Signup page. Their access will remain inactive until approved and assigned a role by the Admin.

10.2 Core Functionalities by User Role

The Time Tracker system implements role-based access control, ensuring users only access the tools and features relevant to their responsibilities.

Admin

- Access the Admin Dashboard with full system visibility.
- Manage user accounts:
    - View, edit, deactivate users.
    - Assign or change roles (Admin, Project Manager, Developer).
- Create and manage all projects, modules, and tasks.
- Generate and download global productivity reports.
- Access detailed analytics across all teams.

Project Manager

- Access the Project Manager Dashboard.
- Create and manage projects and their corresponding modules.
- Assign tasks to developers with specified deadlines and priorities.
- Track developer performance and task progress in real-time.
- View dashboards with team analytics and task overviews.
- Review and verify developer time logs.

Developer

- Access the Developer Dashboard.
- View all assigned tasks with due dates and status.
- Log time using:
    - Real-time start/stop timer, or
    - Manual time entry per task.
- Update task statuses: *In Progress*, *Completed*, or *Blocked*.
- View personal productivity summaries with weekly logs and progress charts.

Each user interface is customized based on the assigned role, improving usability and reducing complexity.

10.3 Reporting and Dashboard Features

Visual Dashboards

Role-based dashboards present real-time, graphical data, including:

- Total and average time logged per task/project.
- Developer activity trends over time (daily/weekly).
- Task status distribution (pending, in progress, completed).

Report Generation

Admins and Project Managers have access to advanced reporting tools:

- Generate reports by:
    - User, Project, Task, or Date Range.
- Export reports as PDF files for documentation or performance review.
- Filter and analyze logs to identify trends and bottlenecks.

Reporting Technologies

- **Chart.js** (or similar libraries): For rendering bar charts, pie charts, and time-series graphs.
- **PDFKit / jsPDF**: For converting dashboards into downloadable PDF reports.

10.4 System Navigation

The system features a clean and accessible navigation structure:

- Use the top navigation bar or side menu to access:
    - Dashboard
    - Projects
    - Modules
    - Tasks

    - Time Logs
    - Developers
    - Reports
    - Settings
- Click Logout (top-right corner) to securely end your session.

11. Conclusion and Future Work

11.1 Conclusion

The Time Tracker project provides an innovative and effective solution for managing and analyzing developer productivity. By streamlining task assignment, time logging, and performance tracking, the project empowers teams to enhance their work efficiency and meet deadlines with greater precision. The system's modern, full-stack web approach integrates a user-friendly React interface with a powerful backend based on FastAPI, ensuring that it meets the needs of both individual developers and larger teams.

Key Takeaways:

- Modular Design for Maintainability: The system's architecture is designed to be modular, enabling easy updates and maintenance. New features and components can be added without disrupting the core functionality, ensuring long-term sustainability.
- Scalable Backend Using FastAPI: FastAPI offers high performance and scalability, handling increased traffic and a growing user base effortlessly. The backend is designed to accommodate future scaling needs, ensuring the system can grow alongside user demand.
- User-Friendly React Interface with Role-Specific Views: The front-end experience is tailored for various user roles, allowing developers, managers, and administrators to interact with the system according to their needs. The intuitive interface enhances productivity and reduces the learning curve for users, ensuring adoption across the team.

The system allows for real-time tracking of work statuses and module-level granularity, making it easy to understand task progress. With the integration of dropdown menus for various task statuses, projects, modules, and users, it simplifies project management and boosts visibility into team activities. The platform offers flexibility, allowing admins to easily manage modules and other project components.

Overall, the Time Tracker is designed to streamline workflows, enhance transparency, and improve productivity through a seamless integration of frontend and backend technologies.

11.2 Future Enhancements

While the current version of the Time Tracker project provides a robust solution for time tracking and task management, there are several potential enhancements that can make the system more versatile, interactive, and efficient. These enhancements focus on improving user engagement, expanding functionality, and providing deeper insights into team productivity.

1. Mobile Application Version (React Native):

- The development of a mobile application version using React Native will allow users to track their time and manage tasks on-the-go. This would make the platform accessible across various devices, providing flexibility for remote teams or developers working outside the office. Mobile access would also improve overall user engagement and ensure users can track their time at any point throughout their day.

2. Integration with Slack, Jira, or GitHub:
    - Adding integrations with popular tools like Slack, Jira, or GitHub would further streamline project management. For instance, time logs could be automatically created from Jira issues or GitHub commits, reducing manual input. Notifications in Slack could alert users to task updates or deadlines, enhancing communication and collaboration.
3. Gamification Features to Motivate Users:
    - Introducing gamification elements could increase user engagement and motivation. Features like progress bars, achievement badges, leaderboards, or challenges could make time tracking more enjoyable and rewarding. Developers could be encouraged to complete tasks more efficiently and meet deadlines by earning rewards or recognition within the platform.
4. AI-Based Analytics to Suggest Task/Time Optimization:
    - AI-based analytics could provide insights into how developers are spending their time and suggest ways to optimize task allocation and time management. By analyzing historical data, the system could highlight areas for improvement, such as frequently delayed tasks or inefficiencies in the workflow. Personalized recommendations for time optimization could lead to better productivity and smarter task management.
5. Invoice Generation for Freelance Teams:
    - For teams of freelancers or consultants, adding an invoice generation feature would simplify billing and payment processes. The system could automatically generate invoices based on the time logged for each task, project, or client, ensuring that freelance developers can easily track and invoice their work. This feature would be particularly useful for project-based work, where clients are billed according to time spent on specific tasks or milestones.

These enhancements would greatly expand the functionality of the Time Tracker, improving its value for different user groups and aligning with the evolving needs of teams and businesses. By integrating mobile access, external tools, and advanced analytics, the platform can evolve into an even more powerful tool for boosting productivity, collaboration, and overall project success. Future developments will focus on enhancing user engagement, streamlining workflows, and making the application more versatile and adaptable to different use cases. With these enhancements, the Time Tracker can become an indispensable tool for developers, managers, and teams striving for efficiency and productivity in their projects.

12. Annexure

12.1 Glossary of Terms and Abbreviations

Term / Abbreviation	Full Form / Definition
UI	User Interface – The layout and interactive elements of a software application.
API	Application Programming Interface – A set of rules that allows different software entities to communicate with each other.
CRUD	Create, Read, Update, Delete – The four basic operations of persistent storage.
MVC	Model-View-Controller – A software design pattern commonly used for developing web applications.
IDE	Integrated Development Environment – A software application that provides comprehensive facilities to computer programmers for software development.
HTTP	HyperText Transfer Protocol – Protocol used for transmitting web pages on the internet.
JSON	JavaScript Object Notation – A lightweight format for storing and transporting data.

12.2 Reference Link:

- **Clockify**
  https://clockify.me

- **Time Doctor**
  https://www.timedoctor.com

- **TimeCamp**
  https://www.timecamp.com

- **Hubstaff**
  https://hubstaff.com

- **RescueTime**
  https://www.rescuetime.com

- **Everhour**
  https://www.everhour.com

12.3 About Tools and Technologies Used

- **FastAPI**: A modern, high-performance web framework for building APIs with Python 3.6+ based on standard Python type hints.
- **ReactJS**: A JavaScript library for building user interfaces, particularly single-page applications.
- **MongoDB**: A NoSQL database used for storing application data in a flexible, JSON-like format.
- **Python**: A powerful, high-level programming language used for backend logic and data processing.
- **Chart.js / Recharts**: JavaScript libraries used to render interactive charts for reports and dashboards.
- **VS Code**: A widely used code editor with support for debugging, syntax highlighting, and Git integration.
- **Docker (if used)**: A tool designed to make it easier to create, deploy, and run applications by using containers.

12.4 About the Organization – Grownited Pvt. Ltd.

Grownited Pvt. Ltd. is a technology-driven company based in [City, State], committed to providing digital solutions and IT services that empower businesses. The company specializes in areas such as web development, mobile applications, software development, and IT consulting. With a mission to foster innovation and digital growth, Grownited offers a collaborative environment for interns and employees to work on real-world projects that deliver value to clients across industries.

Website: https://grownited.com

Mission: To grow businesses digitally with scalable and secure software solutions.

Internship Role: As an intern at Grownited Pvt. Ltd., the focus was on practical learning in full-stack development using modern technologies like FastAPI, React, and MongoDB, working under experienced mentors on live projects.

12.5 About College

U.V. Patel College of Engineering,

Ganpat University.

Ganpat University-U. V. Patel College of Engineering (GUNI-UVPCE) is situated in Ganpat Vidyanagar campus. It was established in September 1997 with the aim of providing educational opportunities to students from It is one of the constituent colleges of Ganpat University various strata of society. It was armed with the vision of educating and training young talented students of Gujarat in the field of Engineering and Technology so that they could meet the demands of Industries in Gujarat and across the globe.

The College is named after Shri Ugarchandbhai Varanasibhai Patel, a leading industrialist of Gujarat, for his generous support. It is a self-financed institute siter approved by All India Council for Technical Education (AICTE), New Delhi and the Commissionerate of Technical Education, Government of Gujarat.

The College is spread over 25 acres of land and is a part of Ganpat Vidyanagar Campus. It has six ultra-modern buildings of architectural splendor, class rooms, tutorial rooms, seminar halls, offices, drawing hall, workshop, library, well equipped departmental laboratories, and several computer laboratories with internet connectivity through 1 Gbps Fiber link, satellite link education center with two-way audio and one-way video link. The superior infrastructure of the Institute is conducive for learning, research, and training.

The Institute offers various undergraduate programs, postgraduate programs, and Ph.D. programs. Our dedicated efforts are directed towards leading our student community to the acme of technical excellence so that they can meet the requirements of the industry, the nation and the world at large. We aim to create a generation of students that possess technical expertise and are adept at utilizing the technical 'know-hows' in the service of mankind.

We strive towards these Aims and Objectives:

• To offer guidance, motivation, and inspiration to the students for well-rounded development of their personality.

• To impart technical and need-based education by conducting elaborated training programs.

• To shape and mold the personality of the future generation. • To construct fertile ground for adapting to dire challenges.