COLLEGE CODE : 6109

COLLEGE NAME : JAYALAKSHMI INSTUTION OF THECHNOLOGY

DEPARTMENT : COMPUTER SCIENE AND ENGINEERING

STUDENT NM-ID : 849dbfcce321f459f0148603bb827f00

ROLL NO : 05

DATE :07/10/2025

Completed the project named as

Phase _5_ TECHNOLOGY PROJECT

NAME : IBM-NJ-Event Scheduler App

SUBMITTED BY,

NAME : Mahendiran M

MOBILE NO : +91 9080640249

Phase 5 —project Demonstration&Documentation

1. Final Demo Walkthrough

- **Description:**
  A step-by-step demonstration of the Event Scheduler App's core functionalities. This includes:

- **User authentication (signup/login)**
- **Creating, editing, and deleting events**
- **Viewing upcoming events in a calendar or list format**
- **Setting reminders/notifications for events**
- **Sharing events with others (if applicable)**
- **Admin dashboard or user profile (if implemented)**
- **Mobile responsiveness or app deployment (optional)**

  - **Purpose:**
    To visually showcase how a user would interact with the application from start to finish, demonstrating a smooth and bug-free user experience.

Format:

- **3–5 min screen recording/video walkthrough (hosted on YouTube, Loom, or embedded in README)**
- **Optionally, a live demo link for reviewers to test the app themselves.**

---

2. Project Report

- **Contents:**
  - **Event Scheduler App**
  - **Project Title: Team Members: Names and roles**

- **Frontend: React / Vue / Angular / HTML-CSS-JS**

- **Backend: Node.js + Express / Django / Flask**

- **Database: MongoDB / PostgreSQL / Firebase**
- **Other: JWT for auth, Bootstrap/Tailwind CSS, Cron jobs for reminders, etc.**

- **Objectives:**

  Allow users to efficiently schedule and manage event

  Provide reminders and event categorization

  Support CRUD operations for events with real-time updates

- **Timeline Summary: Brief breakdown of weekly progress**

- **Future Scope: Notifications via email/SMS, recurring events, calendar sync (Google/Outlook), etc.**

---

3. Screenshots / API Documentation

Screenshots:

- **Landing/Login Page**
- **Dashboard**
- **Event Creation Form**
- **Calendar/List View of Events**
- **Edit/Delete Events**
- **Notification or Reminder UI (if any)**
- **Responsive/Mobile View (optional)**

API Documentation (if backend is custom-built):

Document each endpoint clearly:

Example:

POST /api/events

→ Creates a new event

→ Body Params: { title, description, date, time, userId }

GET /api/events/:userId

→ Returns all events for a specific user

PUT /api/events/:eventId

→ Updates an event

DELETE /api/events/:eventId

→ **Deletes an event**
Use tools like Swagger or Postman for visual documentation (optional).

.

4. Challenges&Solutions

challenge	solution
Handling time zone differences	**Used moment.js or Intl.DateTimeFormat to standardize time display**
Implementing real-time updates	**Integrated Socket.IO or used setInterval polling**
Deploying backend + frontend	**Used Render / Vercel / Netlify + MongoDB Atlas for smooth deployment**
Managing state in a complex form	**Used React Context or Redux for scalable state management**
Authentication security	**Used JWT tokens, bcrypt for hashing passwords**

🔷 5. GitHub README & Setup Guide

README should include:

- **Project Title & Description**
- **Live Demo Link**
- **Tech Stack**
- **Features**

- **Installation Steps:**

- **Installation Steps:**

cd event-scheduler
npm install
npm run dev

- **Screenshots**
- **Usage Instructions**
- **Known Issues / Future Improvements**
- **License**
- **Contributors**

---

6. Final Submission (Repo + Deployed Link)

What to Submit:

- ✅ **GitHub Repository (well-organized, clean code, proper commits)**
  - **Include all source code (frontend + backend if applicable)**
  - **.env.example for API keys**
  - **README as outlined above**
- ✅ **Deployed Link:**
  - **Frontend (e.g., Netlify, Vercel)**
  - **Backend (if separate, e.g., Render, Railway)**
  - **Working live app with test credentials (optional)**