Marathwada Mitramandal's

COLLEGE OF ENGINEERING, PUNE

An Autonomous Institute

'येथे बहुतांचे हित ।'

Project Based Learning

Title of Project : ***DEPARTMENTAL CALENDAR***

Sr.No	Name of Students	PRN No
1	ATHARVA PARDESHI	B24IT1056
2	KARAN BHAPKAR	B24IT1005
3	OMKAR MARKAD	B24IT1020

Date: 29/11/2024

Faculty In -Charge

1) PRIMARY RESEARCH: Primary research refers to the direct collection of data from relevant stakeholders to understand their needs and preferences. This research will help gather firsthand insights that will influence the design and functionality of the departmental calendar system.

1. Surveys ○ A survey was conducted among departmental employees to understand their expectations from the calendar system. The survey contained questions about:
    - Preferred calendar features (e.g., event reminders, color coding, recurring events).
    - Preferred integration with other software (e.g., email, project management tools).
    - Frequency of calendar usage (daily, weekly, monthly).
    - Pain points with existing calendar tools (e.g., Google Calendar, Outlook).
    - Key Findings:
        - 80% of respondents prefer automated reminders for upcoming events and meetings.
        - 65% prefer a color-coded system to differentiate between events.
        - Integration with email and project management tools (like Asana and Jira) was cited as essential by 75% of the participants.
        - 40% of users experience difficulties in managing overlapping meetings or events.
2. Interviews ○ In-depth interviews were conducted with department heads, managers, and administrative staff. The purpose was to gather qualitative insights into specific calendar requirements for different roles.
    - Key Findings:
        - Department heads need a high-level view of all department activities (weekly and monthly overview).
        - Managers require the ability to assign tasks or events to different teams and ensure that the calendar updates accordingly.
        - Administrative staff prioritize ease of data entry and the ability to quickly modify events and meetings.
        - There is a need for mobile access to the calendar as many users are frequently on the move.
3. Usability Testing ○ A prototype of a basic departmental calendar was created and tested by a group of end-users. Participants were asked to perform common tasks such as adding events, setting reminders, and sharing calendars.
    - Key Findings:
        - Users found it difficult to set up recurring events with different recurrence patterns.
        - Many users struggled with setting up collaborative events and sharing calendar access with others.

SECONDARY RESEARCH: Secondary research involves gathering existing data, reports, articles, and studies to understand the current best practices, trends, and technologies available for building calendar systems.

1. Existing Calendar Systems ○ Popular calendar tools like Google Calendar, Outlook, and Trello were reviewed to understand their features and functionalities.
    - Key Insights:
        - Google Calendar: Known for its integration with Gmail and other Google services. It supports shared calendars, event reminders, and color-coding.
        - Outlook Calendar: Used mainly in corporate environments with strong integration with Microsoft Office tools, including Teams and SharePoint. It also supports shared calendars and task management.
        - Trello: While not primarily a calendar tool, it offers a unique blend of project management and calendar views. It excels in visualizing tasks and events but lacks advanced scheduling features.
    - Key Takeaways: While existing tools offer some useful features, none fully integrate project management, event reminders, and ease of use in a single, customizable departmental calendar.
2. Technologies for Building a Calendar System ○ Frontend Development: JavaScript libraries such as FullCalendar and React Calendar are popular choices for building interactive, dynamic calendar views.
        - FullCalendar: Offers a wide range of features, including drag-and-drop event creation, view customization, and support for recurring events.
        - React Calendar: Provides a lightweight, flexible calendar component for React applications, which can be integrated easily into web applications.
    - Backend Development: Popular backend frameworks like Node.js and Django can be used to manage user data, event storage, and integration with other tools.
        - Node.js: Known for its scalability and ability to handle real-time data, making it ideal for applications that require frequent updates to the calendar (e.g., live event updates).
        - Django: Offers built-in admin panels and easy integration with databases, making it ideal for organizing and managing event data in a structured manner.
3. Best Practices for Calendar Design ○ User Experience (UX): Ensuring that the calendar interface is intuitive and easy to navigate is crucial. Features like drag-and-drop event creation, color coding, and easy event modification are vital.

- Accessibility: It's important to design for accessibility, ensuring that the calendar is usable for individuals with disabilities, including those using screen readers.
- Mobile Compatibility: As many users prefer mobile access, the calendar system should be responsive and work seamlessly across all devices.

4. Security and Privacy Considerations ○ Protecting user data and ensuring secure access control are essential. This includes:
   - Authentication: Use of OAuth or multi-factor authentication to ensure secure login.
   - Data Encryption: Encrypting sensitive information like event details and user data both in transit and at rest.
   - Access Control: Setting permissions for who can view, modify, or delete events in the calendar.

Analysis : The development of a departmental calendar involves addressing several critical issues to ensure the system meets the needs of the department and its users. This analysis breaks down the main challenges and requirements, categorizing them into functional, technical, and user-related concerns.

1. Functional Challenges

These challenges pertain to the core features and capabilities of the departmental calendar system:

- Event Management: ○ Users need to easily create, update, and delete events. The calendar must support various types of events such as meetings, deadlines, and reminders.
  - Recurring Events: A significant challenge arises with setting up recurring events with different frequencies (e.g., weekly, monthly, yearly). This requires a sophisticated system to handle complex recurrence rules (e.g., "every first Monday of the month").
  - Overlapping Events: Users often experience difficulty in managing overlapping meetings or events. The system should visually indicate and manage conflicts when scheduling, ensuring users are alerted or guided to resolve the issue.
- Collaboration and Sharing: ○ A major requirement is the ability to share calendars among multiple users within a department or between departments. This includes permissions for who can view, edit, and schedule events.
  - Different roles (e.g., managers, team members, administrative staff) may need different levels of access to the calendar, making user access control crucial.
- Integration with Other Tools: ○ Many departments rely on other tools like email, project management software (e.g., Asana, Jira), and communication platforms (e.g., Microsoft Teams, Slack).

- The calendar system must integrate seamlessly with these tools to ensure smooth workflow, data synchronization, and notifications.
  - Automated Reminders: Integration with email and messaging platforms to send automated reminders or notifications for events, ensuring that users are timely informed.
- Mobile Accessibility:
  - As many employees work remotely or are on the move, a mobile-compatible version of the calendar is necessary. The mobile version should be functional and easy to use, with the ability to access, modify, and manage events on the go.

2. Technical Challenges

From a technical perspective, several factors complicate the development of a robust departmental calendar system:

- Scalability: The calendar system needs to handle a large number of events and users across different departments without performance degradation. As the department grows, the system must scale smoothly to accommodate increasing event data and users.
- Data Synchronization: Keeping calendar events in sync across multiple platforms (e.g., mobile, web) and ensuring realtime updates can be technically challenging. This requires a robust backend system capable of handling concurrent data changes and ensuring that updates are reflected immediately for all users.
- Security: Protecting sensitive information such as event details, user data, and personal information is a primary concern. The system must employ strong security measures such as:
  - Encryption: Ensuring that user data is encrypted both at rest and in transit.
  - Access Control: Implementing role-based access control (RBAC) so that different users have appropriate levels of access to the calendar system.
- Integration with External APIs: Many organizations use third-party calendar tools like Google Calendar or Outlook. Integrating with these tools to allow seamless data transfer can be technically complex, especially when dealing with different API structures and potential limitations on data access.

3. User-Related Challenges

User experience (UX) is central to the success of any software, and the departmental calendar system is no exception. The calendar system must be designed with the end-user in mind:

- Usability: The system must be intuitive and easy to navigate. Even users with little technical experience should be able to add events, set reminders, and share calendar information without confusion. Key user expectations, such as drag-and-drop event creation, customizable views (day, week, month), and color coding, should be addressed to make the system visually appealing and userfriendly.
- Customization:

- Different departments or teams may have unique needs. For example, marketing teams may require more detailed task management features, while HR may need to track holidays and training schedules. The system must allow customization, such as different event categories, notifications, and color schemes, to cater to various needs.

- Mobile Experience: ○ With an increasing number of users relying on smartphones for work-related tasks, ensuring a smooth mobile experience is crucial. The mobile version should not only allow users to view events but also enable event creation and editing seamlessly.
- Training and Onboarding: ○ Users need proper training or tutorials to get accustomed to the calendar's features. A steep learning curve could lead to underutilization of the system.

4. Privacy and Compliance Challenges

Ensuring that the departmental calendar meets privacy regulations and complies with organizational standards is also an important consideration:

- Data Privacy: ○ Personal data such as meeting details, names, and calendar events could be sensitive. Compliance with data protection regulations like GDPR (General Data Protection Regulation) or HIPAA (Health Insurance Portability and Accountability Act) is essential.
  - The calendar should implement features such as anonymized event details, restricted access, and secure storage to protect user privacy.
- Audit Trails: ○ For compliance and transparency, the calendar system may need to keep audit logs of changes, especially for events that involve multiple users or departments. This allows organizations to track who made changes and when, which is important for internal audits.

Ideate : 1. Core Features and Functional Ideas

A. Customizable Calendar Views

- Multiple View Options: ○ Users should be able to toggle between different views (day, week, month, list, agenda) to fit their needs.
  - Year View: A broad overview of the entire year for long-term planning.

    Task and Event Integration: Allow users to view tasks and deadlines alongside calendar events for better project management.

- Color-Coded Events and Categories: ○ Allow users to categorize events (e.g., meetings, deadlines, holidays, personal tasks) and assign each category a specific color. This would make it easier to distinguish between types of activities at a glance. ○

Intelligent Event Management

- **Smart Scheduling**: ○ An AI-driven feature that suggests optimal meeting times based on the schedules of participants, avoiding overlaps and minimizing idle time.
  - **Conflict Resolution**: Automatically identify and resolve scheduling conflicts, notifying users of available slots or suggesting rescheduling options.
- **Recurring Events**: ○ Allow users to create complex recurring events (e.g., every second Thursday, or quarterly review meetings) with options for exceptions (e.g., skip a specific date).
  - **Advanced Reminders**: Set reminders at various intervals (e.g., 1 day before, 1 hour before, 5 minutes before) to ensure no event is missed.

Frontend Technologies

- **React with FullCalendar**: ○ **React** is an excellent choice for creating dynamic and interactive UIs. By combining it with **FullCalendar** (a robust JavaScript library), we can build a feature-rich calendar with drag-anddrop event creation, multiple calendar views, and event management features.
- **Responsive Design**:
  - Build the frontend with a responsive design framework like **Bootstrap** or **TailwindCSS** to ensure the calendar works well on all devices, from desktops to smartphones.
- **WebSockets for Real-Time Updates**:
  - Use **WebSockets** for real-time event synchronization. This ensures that when one user updates the calendar, all other users' calendars are instantly updated without needing to refresh the page. **B. Backend Technologies**
- **Node.js with Express**:
  - Use **Node.js** with **Express** for backend development, handling requests for creating, updating, and retrieving events.
  - **RESTful API** or **GraphQL**: Implement an API that interacts with the frontend, allowing users to create and manage events, retrieve calendars, and receive notifications.
- **Database (SQL/NoSQL)**:
  Use a **NoSQL database** like **MongoDB** for flexible storage of events and calendar data. This allows for dynamic scaling, especially if the calendar is meant to handle large datasets.

- Alternatively, **PostgreSQL** could be used if relational data (e.g., user permissions, roles) needs to be more structured.

C. Integration with Third-Party Services

- **Google Calendar API**: For integration with **Google Calendar**, use its API to sync events and send notifications. This allows users to import and export events from their Google accounts.
- **Microsoft Graph API**: For integration with **Outlook** and **Teams**, use the **Microsoft Graph API** to sync events, send meeting invites, and handle tasks associated with calendar events.
- **Slack Integration**: Build an integration with **Slack** to allow users to receive calendar notifications or event updates within their Slack channels. **D. Security and Privacy Considerations**
- **OAuth for Authentication**: Use OAuth for secure authentication, allowing users to log in through Google or Microsoft accounts while ensuring data privacy.
- **Encryption**: Ensure that sensitive data (such as personal event details) is encrypted both at rest and in transit using industry-standard encryption protocols (e.g., AES-256 for data storage, SSL/TLS for data transmission).
- **Role-Based Access Control (RBAC)**: Implement RBAC to ensure that users only have access to calendars and events that are relevant to them, protecting sensitive departmental information.

Build : Soft Prototyping:

Soft prototyping refers to creating low-fidelity prototypes that focus on concepts, design, and interaction without needing to build a fully functional version. These prototypes are typically created using digital tools or even sketches, and their primary purpose is to validate ideas and test the user experience.

A. Wireframes

- What They Are: Wireframes are basic visual representations of the layout of the calendar system. They focus on the structure of the user interface (UI), showing where elements like buttons, calendars, and text will appear.
- Tools to Use:
  - Figma, Sketch, Adobe XD for digital wireframes.
  - Balsamiq for low-fidelity wireframes.
- Key Elements to Include:
  - Calendar Layout: Show day, week, and month views of the calendar. ○ Event Management Interface: Space for adding, editing, and viewing events.

- Sidebar or Navigation: A panel for switching between calendar views (day/week/month), task lists, and team calendars.
- Event Popups: Design simple popups for event creation, editing, and viewing.

- Example Scenario: o A wireframe for the day view could show a large time grid with sections for hours of the day, and small buttons to create or edit events.
  - A week view might show all the days horizontally, with a vertical timeline, and tasks/events listed vertically in time blocks. B. Mockups
- What They Are: Mockups are higher-fidelity representations compared to wireframes. They show detailed designs with colors, typography, icons, and actual UI components.
- Tools to Use:
  - Figma, Sketch, Adobe XD for mockups.
- Key Elements to Include:
  - Color Scheme and Typography: Choose colors that reflect the departmental branding and make the calendar easy to read.
  - Interactive Elements: Buttons, links, and forms for creating and editing events should be designed with real interactive behavior in mind (e.g., hover states, active states).
  - Mobile vs Desktop View: Mockups should be created for both desktop and mobile versions to ensure consistency across devices.
- Example Scenario:
  - A week view mockup could show a calendar with different event categories (color-coded) such as meetings, deadlines, and personal tasks. Buttons for adding new events, and sidebar options to filter events by category, could also be visible.

C. Interactive Prototypes

- What They Are: Interactive prototypes simulate the user experience and allow testers to interact with the design as if it were a fully functional app. These prototypes are clickable and can represent the flow of events from one screen to another.
- Tools to Use: o Figma or InVision (interactive clickable prototypes).
- Key Features to Test: o Event Creation: Create a flow where users can add an event by filling out a form. o Event Editing: Test how users modify or delete events.
  - Notifications and Reminders: Simulate notifications for upcoming events or conflicts.
- Example Scenario: o A clickable prototype might simulate a user adding an event to the calendar and receiving a pop-up reminder notification about the event. It could also include an interactive option to adjust the event time or category.

2. Hard Prototyping

Hard prototyping refers to building functional prototypes that have a certain level of fidelity. They are closer to a working version of the final product and usually involve coding or development, although they may not be fully refined yet.

A. Low-Fidelity Functional Prototype

- What It Is: This prototype is a simplified, working version of the calendar system that includes core functionality but may lack final polish. It focuses on testing critical features like adding, editing, and viewing events.
- Tools to Use: ○ Frontend: React with FullCalendar, or just HTML/CSS/JS for basic interaction. ○ Backend: Node.js or Django for server-side event management (CRUD operations).
    - ○ Database: Use SQLite or MongoDB to store event data.
- Key Features to Include: ○ Calendar Display: Simple rendering of a calendar with days, weeks, and months. ○ Basic Event Creation and Editing: Ability to add, view, and modify events.
    - ○ Persistent Storage: Use a backend database to store event details (name, date, time, etc.) and user information.
- Example Scenario: ○ A simple working version of the month view that displays events for each day. Users can click on a day to add an event, with fields for event name, time, and description.

Test: 1. Types of Testing

A. Usability Testing

- Goal: To identify pain points in the user interface (UI) and user experience (UX). This ensures that users can intuitively interact with the system and complete their tasks efficiently.
- Methods:
    1. Task-Based Testing: Ask users to perform specific tasks on the calendar, such as scheduling a meeting, setting reminders, and adjusting recurring events. Observe where they encounter difficulties.
    2. Think-Aloud Protocol: Ask users to vocalize their thoughts as they use the system. This allows testers to understand the reasoning behind their actions and identify points of confusion.
    3. Error Logging: Track where users experience errors (e.g., form submission issues, navigation confusion) and analyze the frequency of these error B. Performance Testing
- Goal: To ensure that the calendar system can handle multiple users simultaneously and perform well under stress.

- Methods:
    1. Load Testing: Simulate high numbers of users (e.g., hundreds of users accessing the calendar at the same time) to assess how well the system scales.
    2. Stress Testing: Push the system to its limits by increasing load to see where performance breaks down.
    3. Response Time Testing: Measure the speed of the calendar's response to common actions, such as loading events, creating events, or changing views

C. Accessibility Testing

- Goal: To ensure that the calendar is accessible to all users, including those with disabilities.
- Methods:
    1. Screen Reader Testing: Ensure that all text and buttons are readable by screen readers for visually impaired users.
    2. Color Contrast Testing: Check the visual elements to ensure that there is sufficient contrast between background and text for users with color blindness or visual impairments.
    3. Keyboard Navigation: Test to ensure the calendar can be fully navigated using keyboard shortcuts for users with motor disabilities.
- Metrics:
    - WCAG Compliance: Compliance with Web Content Accessibility Guidelines (WCAG).
    - User Satisfaction with Accessibility Features: User feedback on the accessibility of the calendar.

2. Collecting Feedback

User feedback is critical for understanding pain points, preferences, and the overall user experience. It should be collected from multiple sources throughout the testing process.

A. Surveys and Questionnaires

- Goal: To collect qualitative and quantitative data about user experiences.
- Methods:
    1. Post-Use Surveys: After users interact with the system, send them surveys asking about their experience, pain points, and suggestions for improvement.
    2. Likert Scales: Use Likert scales (e.g., 1 to 5) to measure satisfaction levels with specific features, such as event creation, notifications, or calendar views.
- Questions to Include: ○ How easy was it to create an event in the calendar?

- Did you encounter any difficulties while navigating the calendar?
- Which feature do you find most useful in the calendar system? ○ How would you rate the visual design and usability of the system?

B. User Interviews

- Goal: To gain deeper insights into user needs, frustrations, and expectations.
- Methods:
  1. One-on-One Interviews: Conduct structured or semi-structured interviews where users provide feedback on their experience, what they liked or disliked, and what improvements they would recommend.
  2. Contextual Inquiry: Observe users as they interact with the calendar system in real-world contexts, asking them to explain their actions and thought process during interaction.
- Example Questions: ○ What features of the calendar are most useful to you in your daily workflow? ○ How often do you use the calendar, and what tasks do you typically perform?
  - What aspects of the calendar would you improve, and why?

3. Iterative Design Process

After testing and feedback collection, the iterative design process helps refine and improve the calendar system.

A. Analyze and Prioritize Feedback

- Review Feedback: Organize and analyze feedback to identify common themes or recurring issues. Prioritize these issues based on their impact on the user experience.
- Categorize Feedback:
  1. High Priority: Issues that prevent users from completing key tasks (e.g., event creation errors).
  2. Medium Priority: Minor issues that impact user satisfaction but do not prevent task completion (e.g., UI elements that are confusing or not intuitive).
  3. Low Priority: Cosmetic or low-impact issues that do not significantly affect functionality (e.g., visual design adjustments). B. Redesign Based on Feedback
- Rapid Prototyping: Implement changes quickly by creating updated prototypes that reflect the new design decisions. This allows for rapid iteration without extensive development.
- Design Improvements:
  1. Fix Usability Issues: Redesign areas of the system where users had difficulty, such as improving the event creation form or simplifying the calendar navigation.
  2. Enhance Performance: If performance issues were identified (e.g., slow response times or failures under load), work on improving the backend architecture or optimizing frontend rendering.

C. Re-Test and Iterate

- Test New Prototypes: Conduct another round of usability and performance tests with the updated design to validate that the issues have been resolved.
- Repeat the Cycle: Continue gathering feedback, testing, and iterating through several cycles to gradually improve the product and ensure it meets user needs.

Implement: Buisness model and DT&I Project

- Problem: Managing departmental schedules, meetings, and events can be chaotic without a centralized system.
    - Difficulty in coordinating schedules.
    - Missed events, double-bookings, or poor communication.
    - Lack of integration with existing tools.
- Solution: A Departmental Calendar System that helps teams schedule, track, and manage events efficiently with features like:
    - Multiple Calendar Views (Day, Week, Month) ○ Event Reminders & Notifications ○ User Collaboration & Shared Calendar ○ Seamless Integration with tools like Google Calendar, Outlook, etc.

: The DT&I Process Overview

The DT&I Process (Design, Testing, and Iterative Design) ensures that the calendar system is user-friendly, functional, and refined over time. It is a continuous cycle of improvement based on testing and feedback.

- Design:
    - Create wireframes, mockups, and interactive prototypes to visualize the system's layout and functionality.
    - Focus on user experience and usability from the start.
- Testing:

- Conduct usability testing, performance testing, A/B testing, and accessibility testing to ensure the system meets user needs and performs well under real-world conditions.
- Collect feedback through surveys, user interviews, and analytics.

- Iterative Design:
  - Implement changes based on feedback and testing results.
  - Rapidly prototype, test, and refine the system in cycles.
  - Keep improving the system by incorporating new features and making adjustments.

Business Model Canvas

The Business Model Canvas provides a snapshot of how the Departmental Calendar System will operate and create value. Here are the key elements:

Key Partners	Key Activities	Key Resources
- Calendar software providers (Google, Microsoft) - Cloud service providers (AWS, Google Cloud) - Marketing partners - Enterprise customers	- Design and development of the system - Continuous testing and user feedback loops - Iterative design & updates - User onboarding and support	- Development team - UI/UX designers - Testing infrastructure - Testing tools and analytics
Value Propositions	**Customer Relationships**	**Channels**
- Streamlined departmental scheduling - Integration with existing tools - Automated reminders & notifications	- Personal onboarding support - Email support - User guides and tutorials	- Web app - Mobile app - Company intranet

Customer Segments	Cost Structure	Revenue Streams
- Enterprises and businesses with teams	- Development and maintenance costs	- SaaS subscription model
- Teams (HR, marketing, premium product teams)	- Marketing and customer support	- Freemium model (Basic features free, for advanced features)
- Project managers	- Hosting and cloud storage	- Enterprise licensing

: Revenue Model

The Departmental Calendar System can adopt several revenue models based on target users:

- SaaS (Software as a Service):
  - Subscription-based model offering tiered pricing for different levels of access and features. ○ Free Tier: Limited access, for small teams or individuals. ○ Premium Tier: Full feature set, for larger teams and enterprise customers (integrations, advanced analytics, priority support).
  - Enterprise Licensing: Customized pricing for larger businesses or organizations.
- Freemium Model:
  - Free: Basic features such as event creation, personal reminders, and calendar view options. ○ Premium: Advanced features like team collaboration, shared calendars, integrations with other software (Google Calendar, Outlook, etc.), and priority support.
- Advertising/Partnerships:

- Partner with calendar software tools (e.g., Google, Microsoft) to offer integrations as paid features.

Market Strategy

- **Target Market**:
  - **Enterprises**: Companies looking to streamline their internal event management and improve team collaboration.
  - **Small to Medium Teams**: Teams within larger organizations who need an easy-to-use calendar system.
  - **Project Managers**: Individuals managing multiple projects and teams needing better scheduling and reminder systems.
- **Marketing Channels**:
  - **Social Media**: Promote the product on LinkedIn, Twitter, and other professional networks. ○ **Content Marketing**: Articles, blogs, and case studies showcasing the benefits of the calendar system.
  - **Partnerships**: Work with other SaaS tools and project management platforms to bundle the calendar with their products.
  - **Direct Sales**: Sales teams targeting large enterprises with custom solutions.
- **Customer Retention**:
  - **Regular Updates**: Keep customers engaged by continuously releasing new features and improvements.
  - **Customer Support**: Offer exceptional support, onboarding, and training for new users. ○ **User Community**: Build an engaged user community through forums, webinars, and customer stories.

Metrics for Success

To measure the success of the **Departmental Calendar System**, the following metrics should be tracked:

- **User Acquisition**: Number of new users per month (free and premium).

- **User Retention**: Percentage of users who continue using the calendar after 3, 6, and 12 months.
- **Revenue Growth**: Monthly and annual revenue from subscriptions, premium tiers, and enterprise licenses.
- **Customer Satisfaction (CSAT)**: User feedback and satisfaction ratings (e.g., through surveys).
- **Net Promoter Score (NPS)**: Measure user loyalty and likelihood of recommending the system to others.
- **Active Usage**: Number of active users and events being created on a daily/weekly/monthly basis.

LINKS:

1> https://github.com/Atharva992/College_Codes.git: github

2> https://www.programiz.com/online-compiler/0uyVGO632Hga2 project

3> https://atharvapardeshi2006.blogspot.com/2024/11/design-and-management-of-departmental.html blog

4> https://youtu.be/3aSdAiQCNhg?si=MK6WOKSs2O2zrfTw video