e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:01/Jauuary-2024 Impact Factor- 7.868 www.irjmets.com

TASK-ASSIST: PERSONAL PLANNER

Adivarekar Divesh Shrikrishna*1, Mahajan Shashank Prashant*2, Navale Prathamesh Rajendra*3, Chintalwar Rudra Vishal*4, Mrs. N.N. Kawale*5

*1,2,3,4Department Of Computer Engineering, JSPM's Rajarshi Shahu College Of Engineering, Polytechnic, Tathawade, Pune – 411033, India.

*5Co-Ordinator, Department Of Computer Engineering, JSPM's Rajarshi Shahu College Of Engineering, Polytechnic, Tathawade, Pune – 411033, India.

DOI : https://www.doi.org/10.56726/IRJMETS49028

ABSTRACT

Personal scheduling software has become a digital solution to help people manage their lives and be productive in today's world. The software serves a diverse user base, including professionals, students, housewives, and entrepreneurs, and its main features include management, good time, organization, and purpose. Users can customize their plans to meet specific needs, whether it's project management or exercise. A significant advantage is the provision of seamless connectivity between multiple devices and access anytime, anywhere. Alerts and reports help meet deadlines, while features such as goal tracking and data analysis help continuously improve and optimize daily work.

I. INTRODUCTION

Personal planner software is a versatile tool designed for users across various backgrounds, such as professionals, students, homemakers, and entrepreneurs. It focuses on effective time management, task organization, and goal setting, allowing users to easily create, modify, and prioritize tasks and events. The software's flexibility enables customization for specific needs like project management or fitness routines. A key advantage is its seamless synchronization across devices, ensuring accessibility from anywhere. Integration with calendars and emails streamlines event management, and reminders reduce the risk of missed commitments. Beyond time management, the software empowers users to set and monitor goals, fostering personal growth. Advanced features include data analytics, offering valuable feedback on productivity and time allocation for continuous improvement of daily routines and priorities.

II. METHODOLOGY

1. Project Initiation :

- Objective Definition: Clearly defined the objectives and purpose of the task management app, focusing on creating, organizing, and prioritizing tasks for personal and organizational planning.
- Stakeholder Identification: Identified key stakeholders and their roles in the project to ensure effective communication and collaboration.

2. Market Research and Requirements Gathering :

- User Stories and Use Cases: Conducted market research to understand user needs and expectations, leading to the identification and documentation of user stories and use cases.
- Feature Prioritization: Prioritized features based on market demands, ensuring alignment with the core functionality of creating, organizing, and prioritizing tasks.

3. Architecture and Design:

- System Architecture: Designed the system architecture to ensure scalability, performance, and maintainability.

4. Technology Stack Selection :

- Node.js and Express: Selected Node.js and Express as the backend framework for building a scalable and efficient server for handling API requests.
- Google APIs Integration: Integrated Google APIs, including Drive API, People API, and Calendar API, to leverage Google's services for file storage, user management, and calendar functionalities.

- Middleware Usage: Implemented middleware such as body-parser, cookie-parser, and dotenv to enhance request parsing, cookie handling, and configuration management.
- Development Environment: Utilized nodemon during development to automatically restart the server upon file changes, streamlining the development process.

5. UI/UX Design :

- Wireframes and Mockups: Created wireframes and design mockups for a user-friendly interface focused on task creation, organization, and prioritization.

6. Development:

- Backend Development: Implemented backend functionalities for task creation, assignment, and organization using Node.js and Express.

7. Integration of Features :

- API Integration: Integrated Google APIs, such as Drive API, People API, and Calendar API, to enhance the app's functionality and provide additional features.
- User Authentication: Implemented secure user authentication and authorization using cookies for session management.

8. User Testing :

- Unit Testing: Conducted unit tests for individual components to ensure they met specified requirements.
- Integration Testing: Performed integration testing to verify the proper functioning of the entire system, including API integrations.

9. Optimization and Performance :

- Code Optimization: Optimized code for improved performance, focusing on response times and resource utilization.
- Scalability Considerations: Ensured the system was designed to scale horizontally and vertically to handle increasing user loads.

10. Security and Privacy Considerations :

- Cookie Implementation: Implemented cookies for secure user session management, enhancing security and privacy.
- Data Encryption: Employed encryption methods to secure sensitive user data and communications.

11. Deployment :

- Cloud Hosting: Deployed the app on a cloud platform such as AWS, Google Cloud, or Heroku for scalability and accessibility.
- Environment Configuration: Configured production environment variables and settings to ensure the app functioned optimally in a live environment.

12. Documentation:

- Code Documentation: Documented the codebase, including functions, modules, and API endpoints, to facilitate collaboration and future development.
- User Documentation: Prepared user documentation to guide users on how to use the app effectively.

13. Post-Launch Support and Maintenance :

- Feedback Loop: Established a feedback loop to gather user feedback and continuously improved the app based on user needs and preferences.
- Version Control: version control (e.g., Git) to manage code changes and updates, allowing for easy rollbacks and collaborative development.
- Bug Fixes and Updates: Addressed post-launch issues, applied bug fixes, and provided ongoing support and maintenance to ensure a smooth user experience

III. MODELING AND ANALYSIS

1) Log-in Mechanism using Google's OAuth API :

- The OAuth login flowchart typically begins with a user attempting to access a protected resource, triggering a request for authorization. The user is then redirected to the OAuth provider's authorization endpoint, where they authenticate and grant permission. Once authorized, the OAuth provider issues an access token.

- The user is redirected back to the original application with the access token, and the application can then use this token to securely access the user's protected resources without exposing their credentials. This flow ensures a secure and standardized method for user authentication and authorization across different applications.

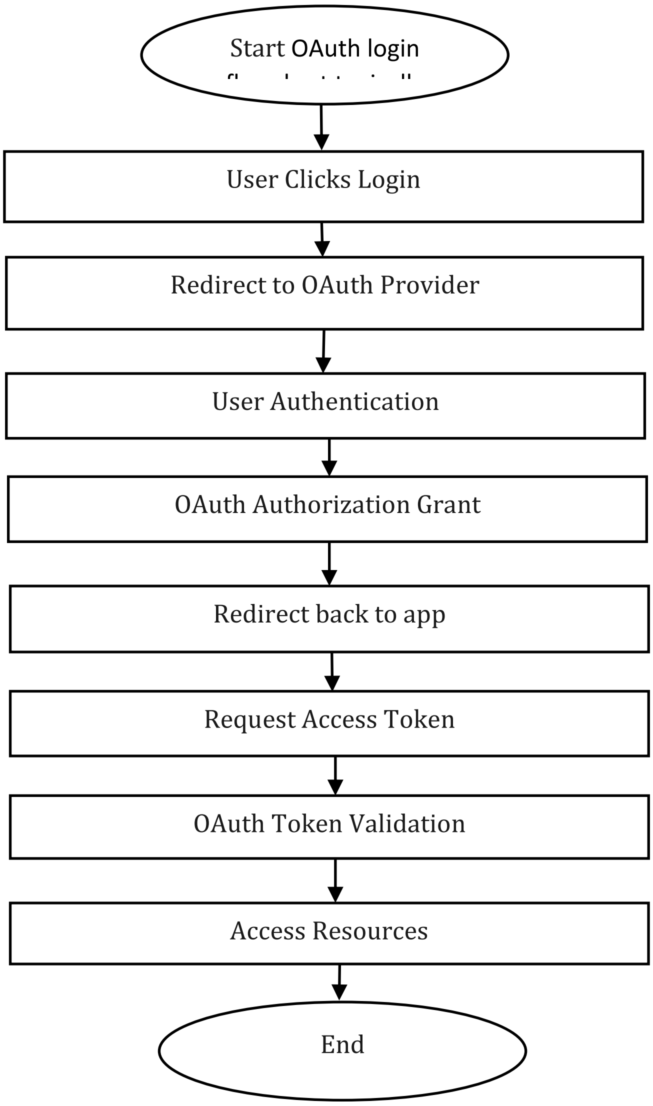


Fig 1. Flowchart of Google's OAuth API

2) Google's Drive API for Storing Data of User :

- Developers can include Google Drive features into their apps by using the Google Drive API. Developers can add, access, and edit files and folders in a user's Google Drive account using a collection of RESTful APIs.
- The API allows for smooth integration of cloud storage capabilities into a variety of applications by supporting features like file uploads, downloads, and sharing rights. This promotes a more connected and cooperative digital world by giving users the ease of managing and accessing their Google Drive files directly from third-party applications.

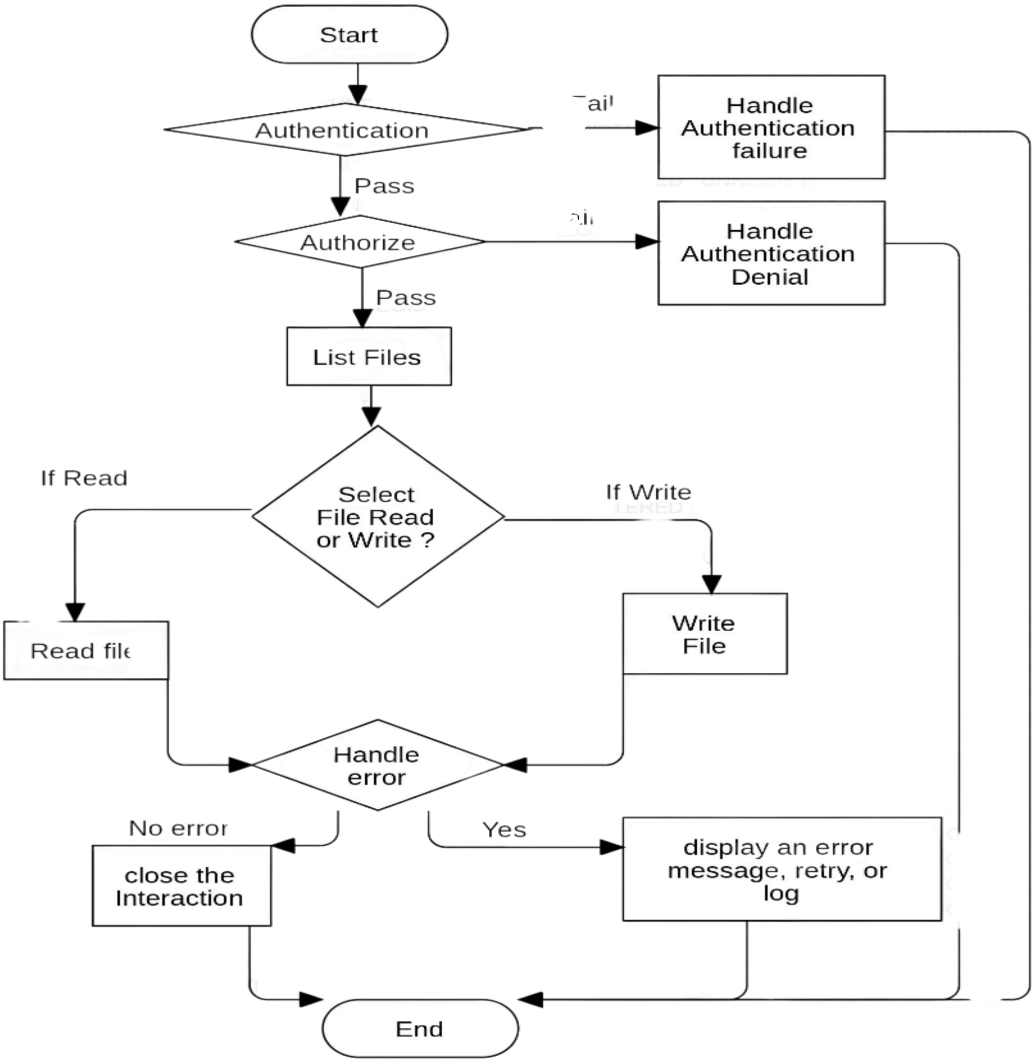


Fig 2. Flowchart of Google's Drive API

3) Google's Calendar API :

- The Google Calendar API provides RESTful access through HTTP calls or Google's client library, offering functionality akin to the Google Calendar web interface. The API encompasses various components, including:

○ Events :

- Events within the calendar encapsulate details like name, start/end times, and participants. These events can be classified as either 'event' or 'event', and are represented by event resources.

○ List :

- A compilation of events, each associated with metadata such as a calendar description or preset time zone. Calendar metadata is depicted by a 'calendar' resource.

○ Calendar List :

- An enumeration of all calendars in the user list within the calendar user interface. CalendarListEntry resource represents the metadata, incorporating user-specific features like calendar color and event notifications.

○ Settings :

- User preferences, reflecting individualized time settings within the calendar user interface, are embodied in service providers.

○ ACL (Access Control Policy) :

- An access control policy governing user or group access levels to the calendar. Each access control policy is portrayed as a resource ACL.

This breakdown illustrates the varied elements and functionalities within the Google Calendar API, encompassing events, lists, user preferences, and access control policies.

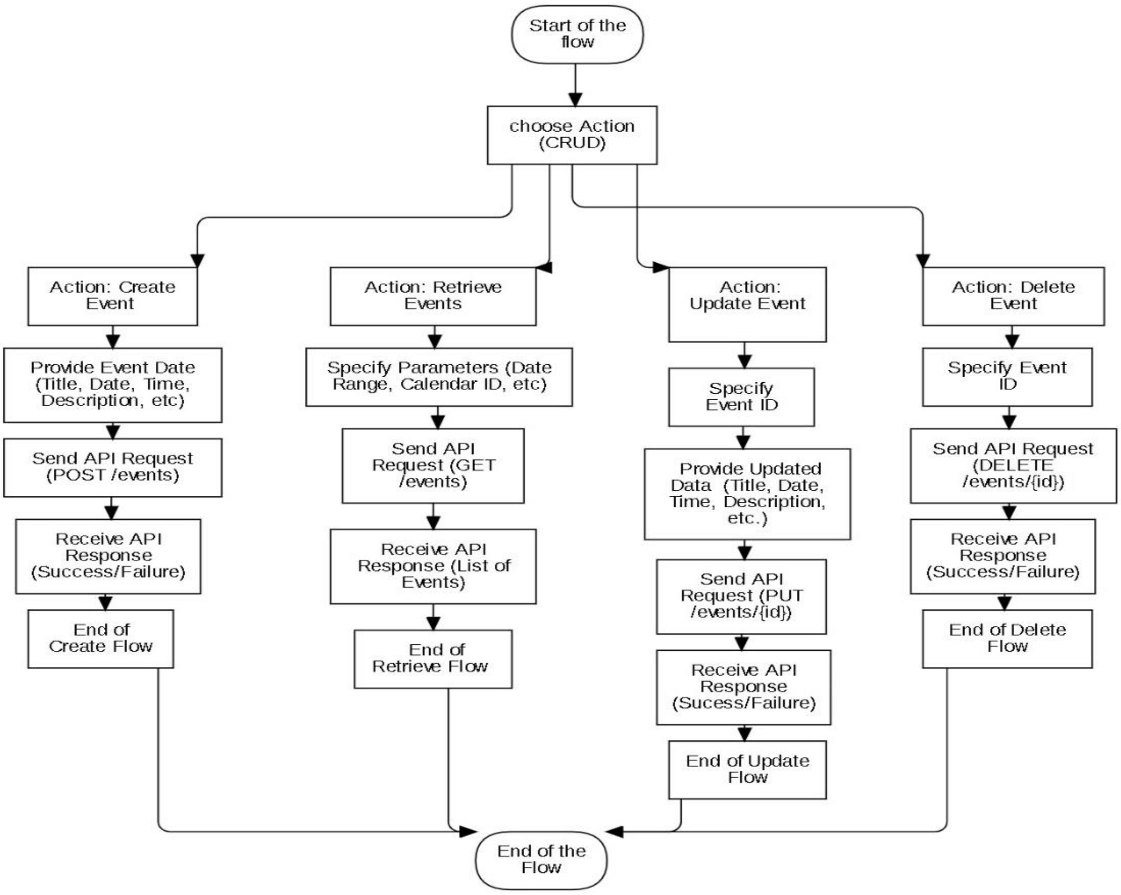

Start of the flow
choose Action (CRUD)
Action: Create Event
Provide Event Date (Title, Date, Time, Description, etc)
Send API Request (POST /events)
Receive API Response (Success/Failure)
End of Create Flow
Action: Retrieve Events
Specify Parameters (Date Range, Calendar ID, etc)
Send API Request (GET /events)
Receive API Response (List of Events)
End of Retrieve Flow
Action: Update Event
Specify Event ID
Provide Updated Data (Title, Date, Time, Description, etc.)
Send API Request (PUT /events/{id})
Receive API Response (Sucess/Failure)
End of Update Flow
Action: Delete Event
Specify Event ID
Send API Request (DELETE /events/{id})
Receive API Response (Success/Failure)
End of Delete Flow
End of the Flow

Fig 3. Flowchart of Google's Calendar API

IV. RESULTS AND DISCUSSION

1. Login Process :

The login process involves users accessing a secure Google login page using OAuth authentication. After entering credentials, the application receives an access token for limited data access. This token is securely stored on the server and included in headers for subsequent authenticated requests, ensuring a streamlined and secure interaction with Google APIs. This OAuth-driven approach enhances the security and integrity of the login process, crucial for the overall user experience and application security.

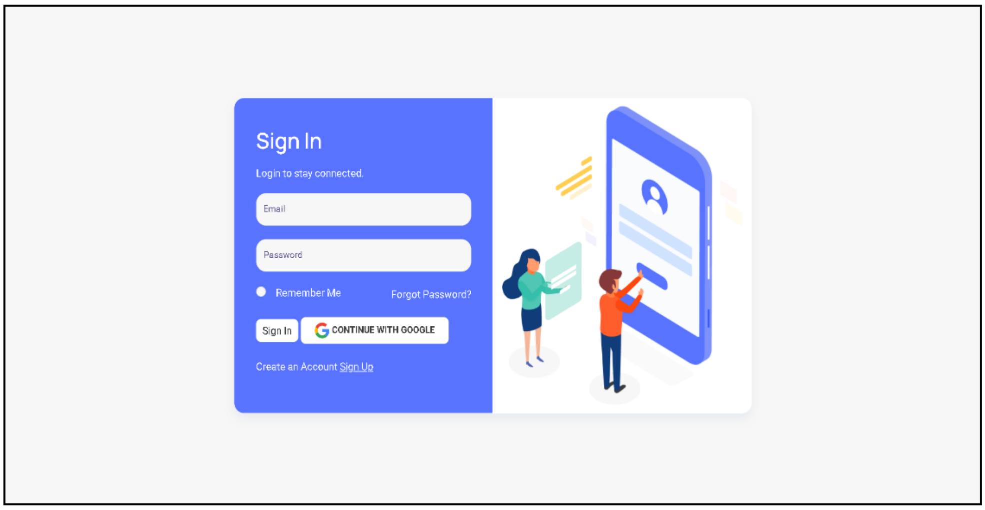
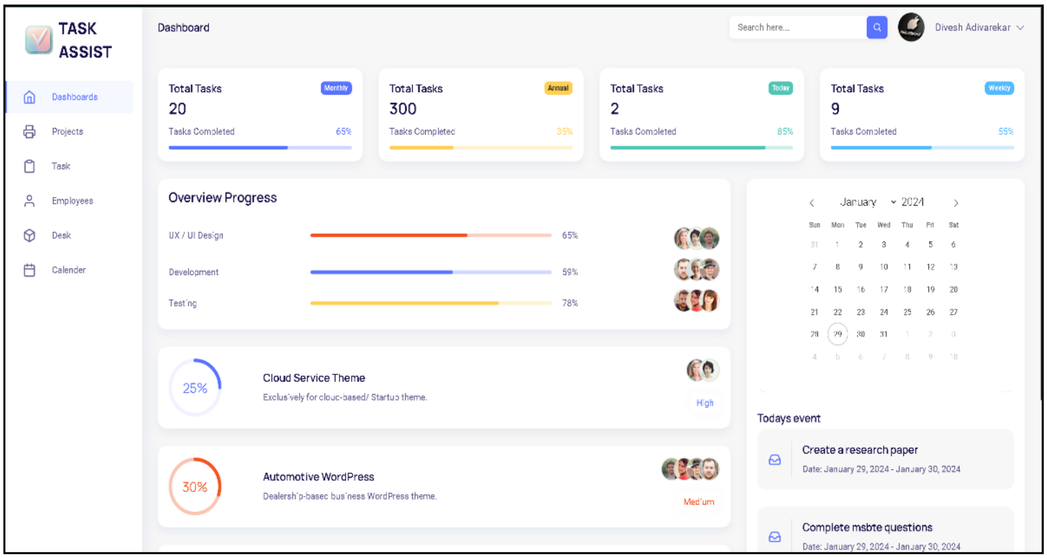


Fig 4. Login Page of Task-Assist

2. Home Page :

The main page provides a consolidated display of user tasks, offering insights into task progress, total tasks, and completed tasks. Users can easily access and control their calendar events through the Google Calendar API within the same interface, establishing a cohesive platform for both task management and event scheduling. This efficient approach aims to improve user productivity by presenting key information in a clear and organized format.



Total Tasks Monthly 20 Tasks Completed 65%

Total Tasks Annual 300 Tasks Completec 35%

Total Tasks Today 2 Tasks Completed 85%

Total Tasks Weekly 9 Tasks Completed 55%

Overview Progress

UX / UI Design 65%

Development 59%

Testing 78%

25% Cloud Service Theme
Exclusively for cloud-based/ Startup theme.
High

30% Automotive WordPress
Dealership-based business WordPress theme.
Medium

January 2024

Sun	Mon	Tue	Wed	Thu	Fri	Sat
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

Todays event

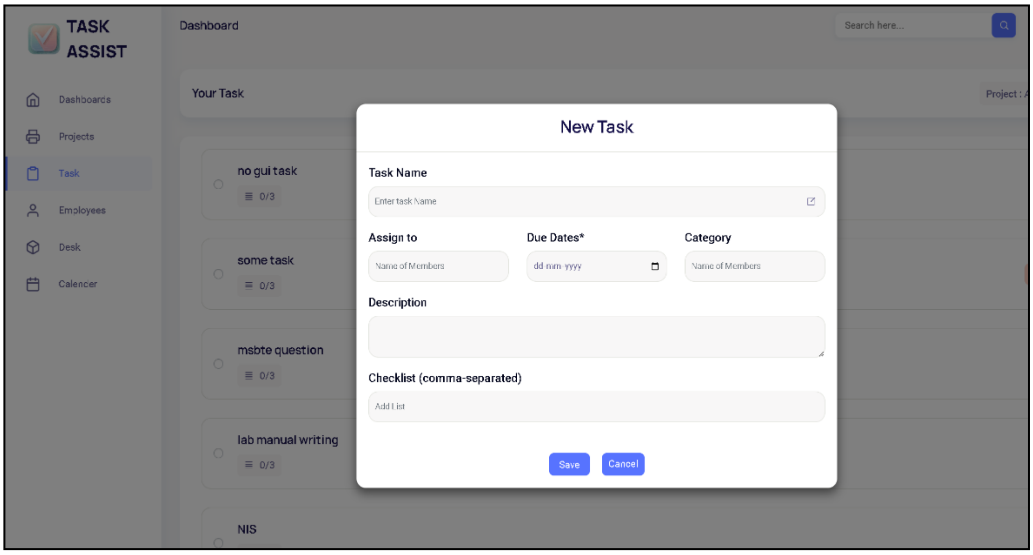
Create a research paper
Date: January 29, 2024 - January 30, 2024

Complete msbte questions
Date: January 29, 2024 - January 30, 2024

Fig 5. Home Page of Task-Assist

3. Adding a New Task :

Adding a new task in the application is a straightforward process. Users input task details, such as the task name, due date, and notes, using an intuitive interface. The submitted task seamlessly integrates into the user's task list, ensuring a quick and organized task management experience. This feature enhances the overall efficiency of task tracking within the application.



no gui task 0/3

some task 0/3

msbte question 0/3

lab manual writing 0/3

NIS

New Task

Task Name: Enter task Name

Assign to: Name of Members

Due Dates*: dd-mm-yyyy

Category: Name of Members

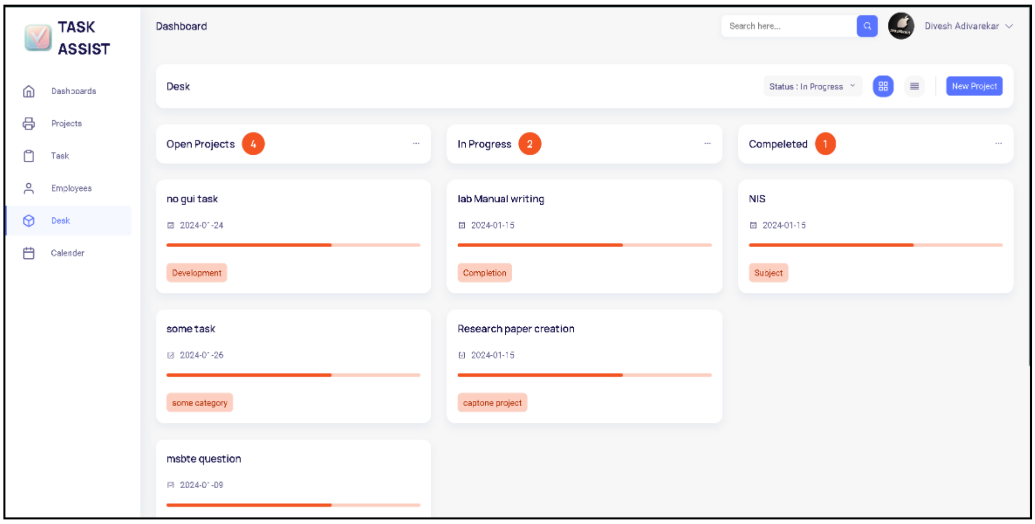
Description:

Checklist (comma-separated): Add List

Fig 6. Adding a New Task

4. Dashboard of all Tasks :

The application's dashboard offers a centralized perspective on tasks, encompassing the task list, progress status, and due dates. Users can efficiently oversee and handle tasks, obtaining rapid insights into completion and prioritizing based on due dates. This efficient functionality improves task visibility and organizational aspects within the application.



Open Projects 4

- no gui task — 2024-0‿-24 — Development
- some task — 2024-0‿-26 — some category
- msbte question — 2024-0‿-09

In Progress 2

- lab Manual writing — 2024-01-15 — Completion
- Research paper creation — 2024-01-15 — captone project

Compeleted 1

- NIS — 2024-01-15 — Subject

Fig 7. Dashboard of Task-Assist

V. CONCLUSION

In conclusion, the development and implementation of this web application, incorporating HTML, CSS, JavaScript, Node.js, Express, and various Google APIs, have yielded a robust and user-friendly platform for task management. The integration of Google's OAuth for secure login, Drive API for data storage, and Calendar API for seamless event coordination has significantly enhanced the application's functionality.

The home page offers a consolidated view of tasks and calendar events, promoting user efficiency, while features like task addition and the task dashboard contribute to a streamlined and organized task management experience. The systematic methodology employed in the development process has resulted in a cohesive and scalable application that not only meets user needs but also adheres to best practices in web development. Overall, this research and development endeavor have successfully addressed the complexities of task management, providing a valuable tool for users across diverse backgrounds.

VI. REFERENCES

[1] S. McKenna "Organisational Complexity and Perceptions of Task", Task Management: An International Journal Vol3

[2] YouTube : Look for channels dedicated to planner setups, organization tips, and productivity hacks.

- UI/UX Design : https://youtu.be/c9Wg6Cb_YlU?si=j6644sKf4Qx3WtKn
- JavaScript : https://youtu.be/hKB-YGF14SY?feature=shared

[3] Tech and Development Resources:

- GitHub Repositories : https://github.com/
- Stack Overflow : https://stackoverflow.com/
- Design Illustrations : https://designstripe.com/search/illustrations