CHAPTER – 1

INTRODUCTION

During my internship, I had the opportunity to work on a web development project called Material Dashboard, an admin panel built using modern frontend technologies like React.js and Material UI (MUI). Web development is the process of creating and maintaining websites and web applications, and it typically involves both frontend and backend development.

Frontend development focuses on the parts of a website that users interact with, using technologies like HTML, CSS, JavaScript, and frontend frameworks such as React. Backend development, on the other hand, deals with server-side logic, databases, and application functionality. In this project, I was primarily involved in the frontend aspect, where I learned how to build responsive, interactive, and visually appealing user interfaces. Material Dashboard follows Material Design principles introduced by Google, which emphasize clean, modern, and consistent design across platforms.

The project includes features such as navigation menus, cards, data tables, charts, and forms—all essential for creating a professional and functional admin dashboard. I contributed by developing reusable components, styling elements using MUI, and ensuring the responsiveness of the interface across different devices. This experience helped me improve my understanding of React's component-based architecture, state management, and props handling, as well as how to use version control systems like GitHub for collaboration. Overall, this project not only enhanced my technical skills in web development but also gave me practical exposure to building real-world user interfaces with industry-standard tools.

To sum up the introduction, this internship served as a bridge between classroom learning and industry application, offering a practical lens into the technical and design dimensions of web development. It reaffirmed that building user-friendly and high-performance web applications is not merely a technical challenge, but a multidisciplinary mission that demands awareness, vigilance, and action from every individual involved in the web development lifecycle.

CHAPTER -2

ABOUT THE ORGANIZATION

2.1 Introduction

Established in 2013 and headquartered in Bengaluru, Karnataka, TechCiti Software Solutions is a leading IT Managed Services Provider (MSP) specializing in delivering end-to-end IT infrastructure solutions. The company caters primarily to small and medium-sized businesses (SMBs) across India, offering services that extend beyond traditional IT support to encompass comprehensive technology road mapping, planning, and strategy. With a presence in over 12 major cities and a diverse portfolio of satisfied corporate clients, TechCiti is committed to ensuring that investments in information technology yield maximum return on investment (ROI) for its stakeholders.

Techciti Software Private Limited offers a comprehensive range of services, including software development, IT consulting, and technical training. The company's training programs cover high-demand fields such as software development, cybersecurity, cloud computing, and data analytics. These programs are designed to align with global certification standards, making them highly relevant to industry needs. In addition to technical training, Techciti places a strong emphasis on job placement and employability enhancement, providing dedicated support to help trainees secure positions in top-tier organizations.

2.2 Vision

TechCiti envisions a future where individuals and organizations can fully realize their potential by embracing cutting-edge technology. The company seeks to become a key driver of digital transformation by enabling smart solutions that redefine how businesses engage with customers, data, and growth opportunities.

2.3 Mission

The mission of TechCiti is to become a leading provider of end-to-end IT infrastructure and service solutions, recognized for innovation, reliability, and customer focus. By continuously delivering quality-driven and cost-effective IT services, the company aims to support business success through technology, while fostering long-term relationships built on trust, excellence, and performance.

The specific objectives of TechCiti's mission include:

- **Deliver End-to-End IT Services:** Provide comprehensive infrastructure and software solutions that support clients throughout their digital transformation journey.
- **Ensure Customer Satisfaction:** Build long-term client relationships by consistently meeting or exceeding service expectations.
- **Promote Innovation:** Encourage continuous improvement and the adoption of cutting-edge technologies in all solutions offered.
- **Support Business Growth:** Help client businesses scale efficiently by aligning IT strategies with their goals.
- **Maintain Service Quality:** Adhere to high standards in project delivery, technical support, and client communication.
- **Optimize Cost-Effectiveness:** Offer affordable and efficient solutions that deliver strong ROI for small and medium enterprises (SMEs).
- **Strengthen Team Capability:** Invest in training and professional development to maintain a skilled and knowledgeable workforce.

2.4 Services

- **IT Infrastructure & Data Center Solutions:**
  TechCiti provides tailored IT infrastructure services including server setup, cloud solutions, disaster recovery, and managed IT services to ensure secure, scalable, and reliable systems for businesses.

- **Cybersecurity Services:**
  The company offers robust cybersecurity solutions such as network security, endpoint protection, data encryption, security audits, and incident response to safeguard businesses from digital threats.
- **Networking Solutions:**
  TechCiti designs and implements network infrastructures including wired, wireless, and cloud networks, ensuring businesses have secure and efficient connectivity.
- **Business Software Development:**
  Custom software solutions are provided for managing CRM, accounting, HR, inventory, and project management processes, helping businesses streamline their operations.
- **Application Development (SAP and Non-SAP):**
  TechCiti offers SAP and non-SAP application development, integration services, and security solutions, tailored to modernizing and optimizing business processes.
- **End User Computing & Peripherals:**
  TechCiti supplies desktops, laptops, printers, monitors, and other peripherals to enhance business operations, with a focus on reliability and performance.
- **IT Training Services:**
  The company offers various IT training programs for individuals and businesses to enhance technical skills, including certifications and online courses.

2.4.1 Corporate Training Programs, Certificate Programs, and Online Courses:

TechCiti offers **corporate training programs** tailored to upskill employees in areas such as cloud computing, cybersecurity, and software development. The **certificate programs** provide recognized qualifications to boost career prospects, while **online courses** offer flexible learning options for individuals seeking to improve their IT skills at their own pace.

2.4.2 Workshops and Seminars:

TechCiti organizes **workshops and seminars** that provide in-depth knowledge and hands-on experience on topics such as emerging technologies, best practices, and industry trends. These are designed to help businesses stay competitive and adopt new technologies effectively.

2.4.3 Consultancy Services:

TechCiti offers expert **consultancy services** to help businesses navigate their digital transformation journey. This includes IT strategy planning, infrastructure management, software development consulting, and advising on the best technology solutions to meet business goals.

Fig.2.4 TechCiti Softwate Private Limited logo

CHAPTER -3

ABOUT THE DEPARTMENT

3.1 Introduction

At TechCiti Software Solutions, the Web Development Department plays a key role in creating responsive, interactive, and scalable web applications tailored to client requirements. The department specializes in both frontend and backend development, utilizing modern technologies such as React.js, Material UI, and web services. It works collaboratively with design, development, and IT teams to build seamless digital experiences. Core responsibilities include designing user interfaces, coding interactive features, integrating backend services, testing, and deploying web applications.

During my internship, I was part of this department, contributing to the development of the Material Dashboard project. I focused on frontend development using React and Material UI, gaining hands-on experience in building professional, real-world applications.

- Specializes in designing and developing dynamic, responsive, and user-friendly websites and web applications.
- Primarily uses **React.js**, **Material UI**, **JavaScript**, **HTML5**, **CSS3**, and version control tools like **Git** and **GitHub**.
- Builds intuitive and interactive user interfaces that are responsive across different devices and browsers.
- Collaborates with backend developers to connect frontend components with APIs and databases for dynamic data handling.
- Follows modern design standards (such as Material Design) to ensure clean layout, ease of navigation, and high usability.
- Involves in planning, coding, testing, debugging, and deployment of web applications as per client or internal requirements.

- Works closely with other departments like UI/UX design, testing, and software development for cross-functional integration.
- Uses tools like **VS Code**, **Chrome Developer Tools**, **Figma** (for design), and **GitHub** for version control and collaboration.
- Often follows **Agile or Scrum** frameworks to manage projects efficiently in iterations with regular feedback loops.
- Developed projects like **Material Dashboard**, a modern admin interface that follows Material Design principles and is built for real-time data display and admin control.

3.2 Scope of the Domain

The Web Development domain encompasses the process of building, deploying, and maintaining websites and web applications that are accessible via the internet or an intranet. Its scope is vast and continually expanding due to rapid advancements in technology, growing internet usage, and increasing demand for digital solutions across industries.

Key Areas within the Scope:

- Frontend Development: Focuses on the design and development of the visual aspects of websites and web apps using technologies like HTML, CSS, JavaScript, and frameworks like React, Angular, and Vue.js.
- Backend Development: Involves server-side programming, databases, and APIs to support the functionality of web applications using languages such as Node.js, Python, PHP, and Java.
- Full Stack Development: Combines frontend and backend development skills, allowing developers to build complete web applications from end to end.

- Responsive & Mobile-First Design: Ensures websites are optimized for all screen sizes and devices to provide a seamless user experience.
- Web Security: Involves implementing measures to protect web applications from threats such as data breaches, hacking, and malware.
- E-Commerce and CMS Platforms: Supports businesses through platforms like WordPress, Magento, and custom e-commerce solutions for online transactions and content management.
- Web Performance Optimization: Enhancing site speed, load times, and overall user engagement.
- Cross-Browser & Cross-Platform Compatibility: Ensures web applications work efficiently on different browsers and operating systems.
- Deployment and Maintenance: Covers hosting, monitoring, and updating web applications to keep them secure and functional over time.

Industry Scope and Opportunities:

- Applicable across various industries such as e-commerce, education, healthcare, finance, and media.
- High demand for skilled web developers globally.
- Critical for digital transformation initiatives and online business growth.

CHAPTER – 4

INTERNSHIP DOMAIN

4.1 Introduction

My internship at TechCiti Software Solutions was focused on the domain of Web Development, a vital area in the IT industry that involves creating and maintaining websites and web applications. This domain combines both technical and creative skills to develop user-friendly, responsive, and efficient digital platforms. During the internship, I worked within the Web Development Department, where I gained hands-on experience in frontend development using technologies like React.js, Material UI, JavaScript, HTML, and CSS. I contributed to the development of the Material Dashboard Project, an admin interface designed with Material Design principles to provide a responsive and modern user experience. This opportunity enhanced my understanding of full-cycle web application development, from designing interfaces to integrating them with real-world functionalities.

Course Objectives of My Project

1. To understand the structure and workflow of professional web development projects.
2. To learn and apply frontend technologies such as React.js and Material UI in real-time application development.
3. To build responsive and interactive user interfaces that follow Material Design principles.
4. To gain experience in using version control tools like Git and GitHub for collaborative development.
5. To enhance debugging, problem-solving, and optimization skills within a production environment.
6. To work in a team setting, understanding the coordination between UI/UX designers, backend developers, and testers.

4.2 Project Management

At TechCiti Software Solutions, project management is a structured approach used to plan, execute, monitor, and deliver technology projects efficiently and on time. The organization follows best practices to ensure that every project—whether in web development, software solutions, or IT services—is handled systematically with clear goals, timelines, resource allocation, and team coordination.

The project management process typically involves multiple stages such as requirement gathering, planning, design, development, testing, deployment, and maintenance. TechCiti emphasizes agile methodologies, especially in web development projects, to ensure flexibility, faster delivery, and continuous improvement based on client feedback.

Key Features of Project Management at TechCiti include:

- Agile Methodology: Projects are executed in iterations (sprints) with regular feedback loops. Ensures adaptability to changes and continuous delivery of value.
- Team Collaboration: Cross-functional teams including developers, designers, testers, and project managers collaborate effectively. Uses tools like Slack, Trello, or Jira for communication and task tracking.
- Client-Centric Planning: Projects begin with detailed requirement gathering and client consultation. Regular updates and demos are provided to ensure alignment with client expectations.
- Time and Resource Management: Tasks are assigned based on skill sets and availability to ensure timely delivery. Milestones and deadlines are clearly defined and monitored.
- Use of Version Control Systems: Git and GitHub are widely used for source code management, allowing team collaboration and code tracking.
- Quality Assurance and Testing: Rigorous testing phases are conducted to ensure functionality, performance, and security. Bugs and issues are tracked and resolved before deployment.

- Documentation and Reporting: Every phase is documented thoroughly to ensure transparency and ease of future maintenance. Regular project reports and status updates are shared with stakeholders.
- Deployment and Maintenance: Final products are deployed with proper configuration, and ongoing support is provided post-deployment.

CHAPTER 5

SYSTEM REQUIREMENTS

5.1 Hardware Requirements

1. **Processor:** Minimum: Intel Core i3 or AMD Ryzen 3 is recommended to handle Intel Core i5 or AMD Ryzen 5 and above

2. **RAM**: Minimum: 4 GB is recommended to handle 8 GB or more for smoother development

3. **Storage**: Minimum: 500 MB of free disk space is recommended to handle 1 GB SSD space for faster build and run times

5.2 Software Requirements

1. **Operating System**: Windows 10/11, macOS, or a modern Linux distribution (e.g., Ubuntu)

2. **Node.js**: Version 14.x or higher (preferably LTS)

3. **npm or Yarn**: npm v6+ or Yarn v1+ for dependency management

4. **Angular CLI**: Version 13 or higher

5. **Web Browser**: Latest version of Chrome, Firefox, or Microsoft Edge (for testing UI)

6. **Code Editor**: Visual Studio Code (recommended) or any IDE that supports Angular

7. **Version Control**: Git for project cloning and version management 8. Backend/API: RESTful API (either mock server or real backend service)

CHAPTER 6

SYSTEM DESIGN

6.1 Existing System

Before the implementation of the Material Dashboard project, the organization either used a basic admin panel or relied on third-party tools that lacked customization and integration with their internal systems. These traditional dashboards were often built with outdated technologies or limited UI frameworks, resulting in rigid interfaces and inefficient workflows.

The existing system mainly offered:

- Static interfaces with minimal interactivity.
- Poor responsiveness across different devices.
- Limited data visualization and user control.
- Weak integration capabilities with modern APIs or backend services.

This created challenges for both internal teams and clients who needed real-time insights, easy navigation, and a modern user experience.

6.1.1 Disadvantages of Existing System

- Outdated User Interface: The old UI lacked modern design principles and was not user friendly, leading to poor user experience.
- Limited Responsiveness: The admin panel did not adapt well to mobile or tablet screens, reducing accessibility for users on the go.
- No Real-Time Updates:The system lacked features like dynamic data rendering, causing delays in reflecting updated information.
- Poor Maintainability: The codebase was difficult to maintain or extend, as it was not built using modular or reusable components.
- Lack of Customization: Admins and users had limited control over settings, layouts, and data views.

6.2 Proposed System

To address the limitations of the existing admin interface, the Material Dashboard was proposed and developed as a modern, responsive, and scalable web application. Built using React.js and Material UI, the new system follows Material Design principles, offering a clean and intuitive user interface. The proposed system is designed to enhance user experience through responsive layouts, real-time data handling, and modular components that are easy to maintain and customize. It includes features such as interactive dashboards, dynamic charts, tables, and role-based access control to ensure security and personalized access. Unlike the previous static and non-responsive system, the Material Dashboard allows seamless integration with backend APIs and provides better performance, faster load times, and cross-platform compatibility. Overall, the new system improves functionality, aesthetics, and usability, making it a future-ready solution for managing administrative tasks efficiently.

6.2.1 Problem Statement

The organization's existing admin panel was built using outdated technologies and lacked the features required to meet modern business and user expectations. The system had a static interface with minimal interactivity, making it difficult for administrators to perform real-time data operations or manage content efficiently. It was not responsive across different devices, limiting accessibility for users on mobile phones and tablets. Additionally, the user interface lacked visual clarity and did not follow modern design principles, which negatively impacted usability and navigation.

Furthermore, the system had limited support for role-based access control, which posed a risk in managing user permissions securely. It also did not provide dynamic data visualization tools such as charts or tables, which are essential for monitoring key metrics and making informed decisions. Integration with backend services was either manual or inefficient, leading to delays in updating or retrieving important data. Maintenance and scalability were also a challenge, as the code structure was not modular or easy to extend.

Due to these limitations, there was a clear need for a modern, flexible, and performance-driven admin dashboard. The project aimed to develop a new system—Material Dashboard—that would address these shortcomings by providing a responsive, interactive, and visually appealing web interface using technologies like React.js and Material UI.

6.2.2 System Architecture Overview

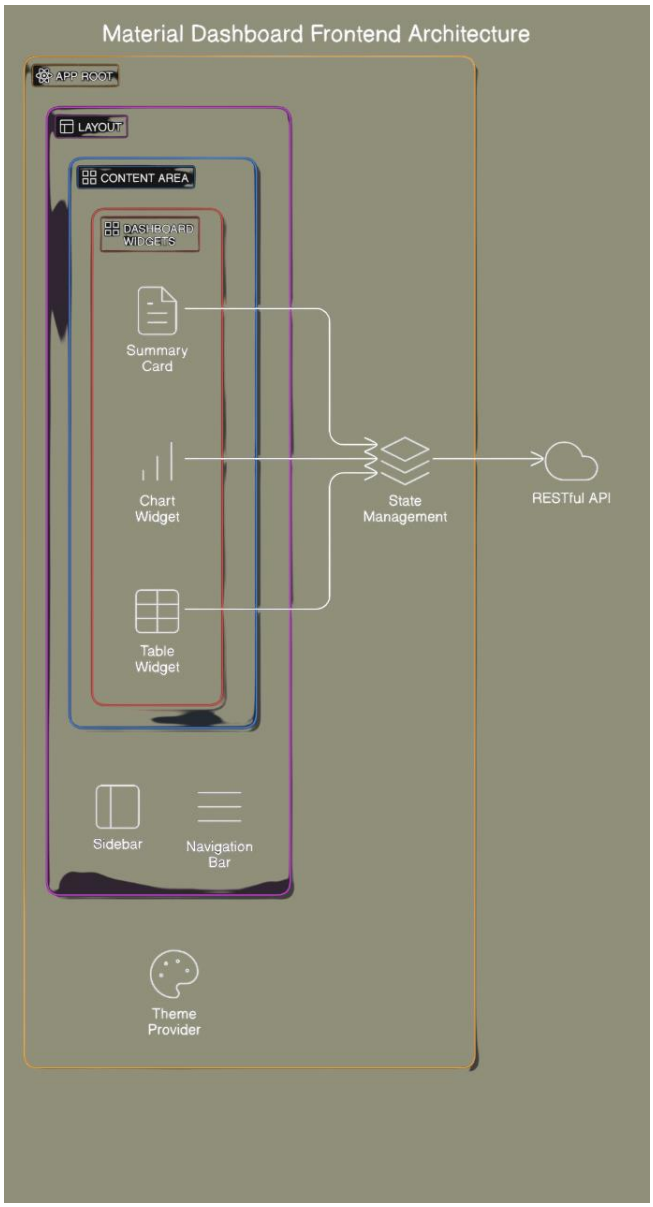


Fig. 5.2 System Architecture

The system architecture of the Material Dashboard-based project is designed to provide an interactive, component-driven user interface for real-time analytics and data visualization. Built using Angular and Material UI principles, the dashboard operates through modular components for maintainability, responsiveness, and clean state management. The application is fully integrated with backend services through RESTful APIs to retrieve and display dynamic data.

The architecture comprises the following core components:

- App Root: This is the entry point of the Angular application and wraps the entire structure.
- Layout: Defines the overall page structure, including the placement of navigation elements and content containers.
- Content Area: Hosts the main interactive elements and dashboard widgets where real-time data is rendered.
- Dashboard Widgets: These are reusable UI components:
    - Summary Card – Displays key metrics or counts.
    - Chart Widget – Renders charts (e.g., line, bar, or pie) for visual analytics.
    - Table Widget – Shows tabular data with sorting, filtering, and pagination capabilities.
- Sidebar & Navigation Bar: Offers intuitive navigation between different dashboard views or modules.
- State Management: Manages the application state using Angular services or libraries (e.g., RxJS or NgRx). This layer mediates between frontend widgets and the backend by maintaining shared application state and handling API responses.
- RESTful API Integration: Backend APIs serve as the data source for dashboard widgets. These APIs are responsible for CRUD operations and serve data in JSON format.

6.2.3 Working

This project provides a responsive, modular, and customizable dashboard interface built with Angular and Material Design principles. Its goal is to display real-time analytics using dynamic widgets that interact with backend RESTful APIs. The system follows a layered architecture to separate UI, logic, and data handling, ensuring maintainability and scalability.

The key steps in the working of the system are:

1. **Application Initialization:** The app root initializes the Angular environment and loads core layout components such as the sidebar, navigation bar, and theme provider. These elements provide the skeleton for the dashboard's visual structure and user navigation.
2. **User Interaction:** Users interact with the dashboard via navigation components, selecting views such as summary cards, charts, or tables. Each widget is context-aware and loads specific data from the backend.
3. **State Management:** Angular services manage application state, caching data where needed and synchronizing it across widgets. Services are responsible for handling API calls and passing data to components in a reactive manner (e.g., using RxJS Observables).
4. **Data Fetching via REST API:** When a widget is activated (e.g., a chart or table), a service sends a request to the backend RESTful API. The server responds with JSON data, which is processed and visualized in real-time.
5. **Data Visualization: Summary Cards** show brief metrics like counts, percentages, or status flags. **Chart Widgets** visualize trends over time using bar, line, or pie charts. **Table Widgets** display detailed records with features like pagination, sorting, and filtering.
6. **Dynamic Theming and Layout Management:** The Theme Provider ensures consistent design across all components. Layouts adjust responsively based on screen size and user preferences, enhancing usability across devices

6.2.4 Advantages

1. The component-based design (e.g., Summary Cards, Charts, Tables) makes the application highly modular, which simplifies maintenance, testing, and scalability.
2. Clear separation between UI components, state management, and API services improves code readability and reusability, following best practices of Angular development.
3. The use of RESTful APIs and reactive state management allows for real-time data updates, making the dashboard suitable for monitoring systems, analytics, or admin panels.
4. Built using Angular Material, the interface is clean, user-friendly, and responsive across devices, providing a consistent experience on desktops, tablets, and phones.
5. The Theme Provider allows dynamic theme switching and customization, making the dashboard visually adaptable to different branding or user preferences.
6. New widgets or modules can be easily added to the dashboard without disrupting existing functionality, supporting future expansion.
7. Centralized state management improves performance and ensures all widgets reflect the latest data, reducing redundant API calls.
8. The layout, navigation bar, and sidebar collectively provide an intuitive structure for users (especially admins) to manage and monitor data.
9. The architecture mirrors that of professional enterprise dashboards, giving you experience that's directly transferable to industry-level Angular applications.

CHAPTER 7

CONCLUSION

The internship project based on the Material Dashboard has successfully achieved its objective of creating a responsive and user-friendly web-based dashboard for data monitoring and visualization. Utilizing Angular and Material Design principles, the project offers a modular and scalable front-end solution that can be easily adapted to a variety of use cases. Integration with RESTful APIs allows the system to dynamically display real-time data, making it suitable for applications such as admin panels, analytics dashboards, and monitoring tools. The component-based structure ensures ease of maintenance and future expansion, while the clean interface enhances overall usability. This project not only enhanced technical skills in front-end development but also provided valuable experience in building scalable systems aligned with modern web development practices.

FUTURE ASPECTS

The project has strong potential for future development and integration across various domains. As a modular and scalable dashboard system, it can be extended to support advanced analytics, real-time data streaming, and AI-powered insights. By integrating backend services like Node.js, Django, or cloud-based APIs, the dashboard can become a central hub for monitoring live systems, whether in IoT, traffic management, or enterprise operations. Additionally, user authentication, role-based access control, and data export features can be incorporated to make it suitable for enterprise-level deployment. With support for real-time WebSocket integration, mobile responsiveness, and internationalization (i18n), the dashboard can evolve into a comprehensive solution for smart city applications, business intelligence platforms, and administrative control panels.

REFERENCES

[1] Optimizing Angular Dashboards for Real-Time Data Analysis *International Journal of Modern Innovations in Research & Management (IJMIRM), 2024.* This paper explores methods for optimizing Angular-based dashboards specifically for real-time data analytics, reviewing performance tuning, component design, and integration patterns.

[2] A Framework for the Structural Analysis of REST APIs *ResearchGate, 2017.* Introduces a framework for analyzing REST APIs based on their description documents, facilitating a comprehensive and well-structured analysis approach.

[3] Design and Implementation of REST API for Academic Information System *IOP Conference Series: Materials Science and Engineering, 2020.* Discusses the development and performance analysis of REST APIs for academic information systems, highlighting the importance of data exchange between systems.

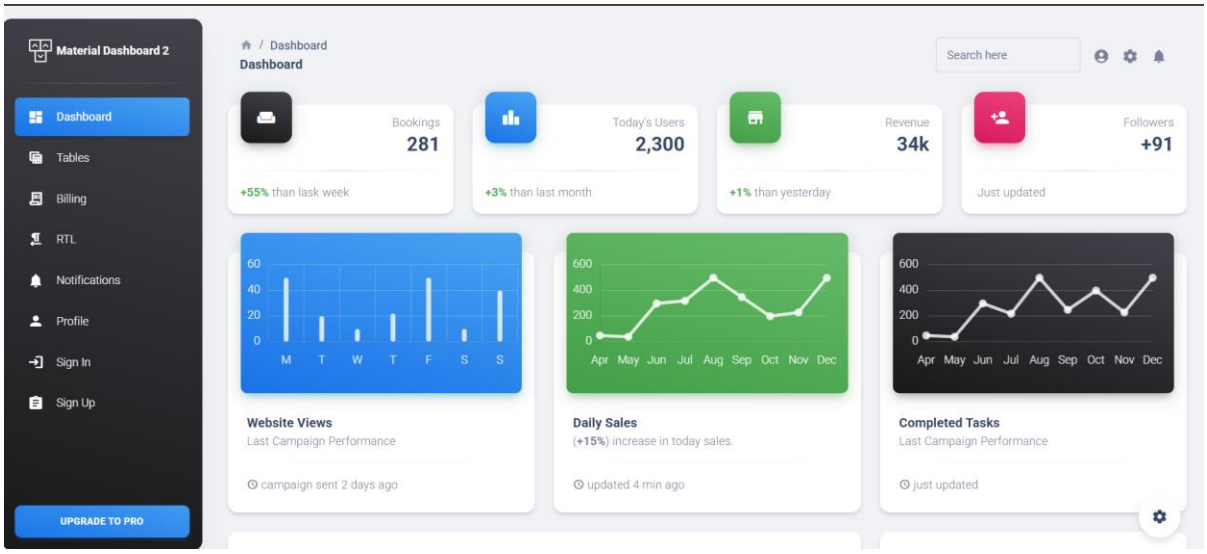
[4] Angular, Unit Testing, Figma, and Numerical Analysis: Building a Dashboard for Our App Using Angular Material *ResearchGate, 2021.* Explores the integration of Angular, unit testing, and design tools like Figma in building dashboards using Angular Material components.

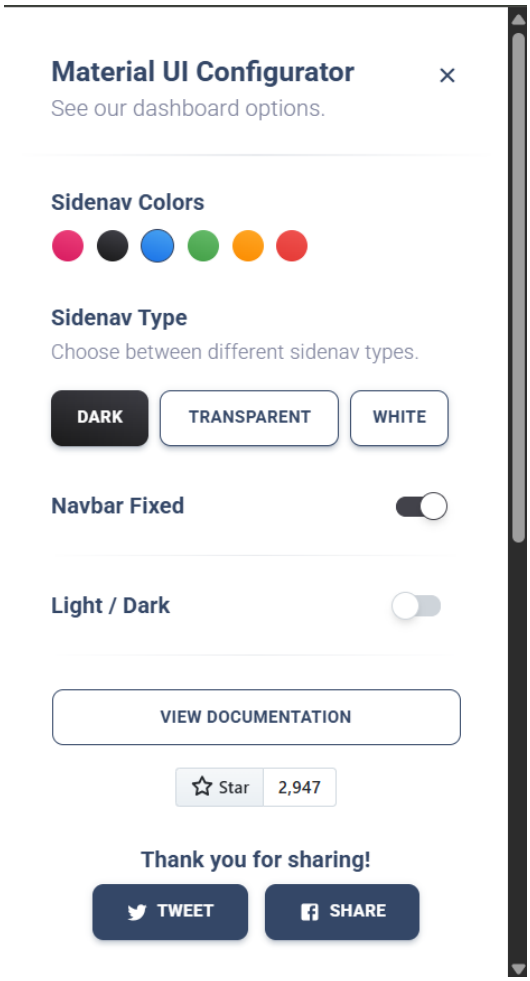
[5] Building Dynamic Dashboards in Angular with Material and Bootstrap: A Complete Guide *Moldstud, 2025.* Provides a comprehensive guide on creating dynamic dashboards in Angular using Material and Bootstrap, covering key techniques and best practices for effective dashboard design.

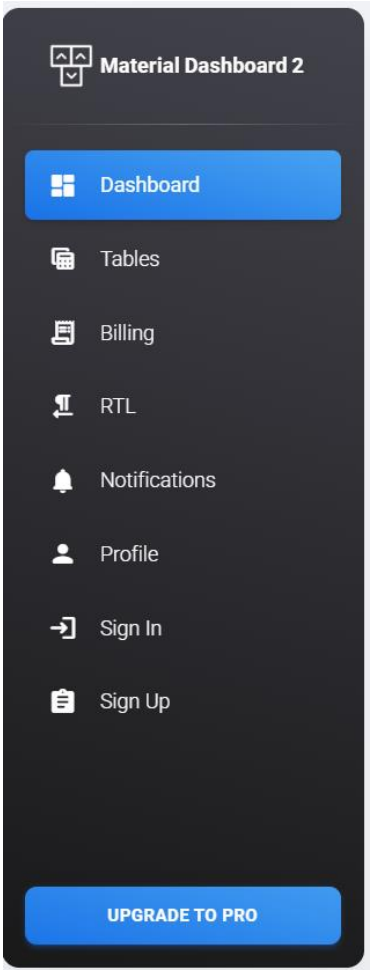
[6] Create an Angular Dashboard Application *DevExpress Documentation, 2025.* A tutorial on creating and configuring a client Angular application that contains a web dashboard, integrating with a server application to handle data requests.

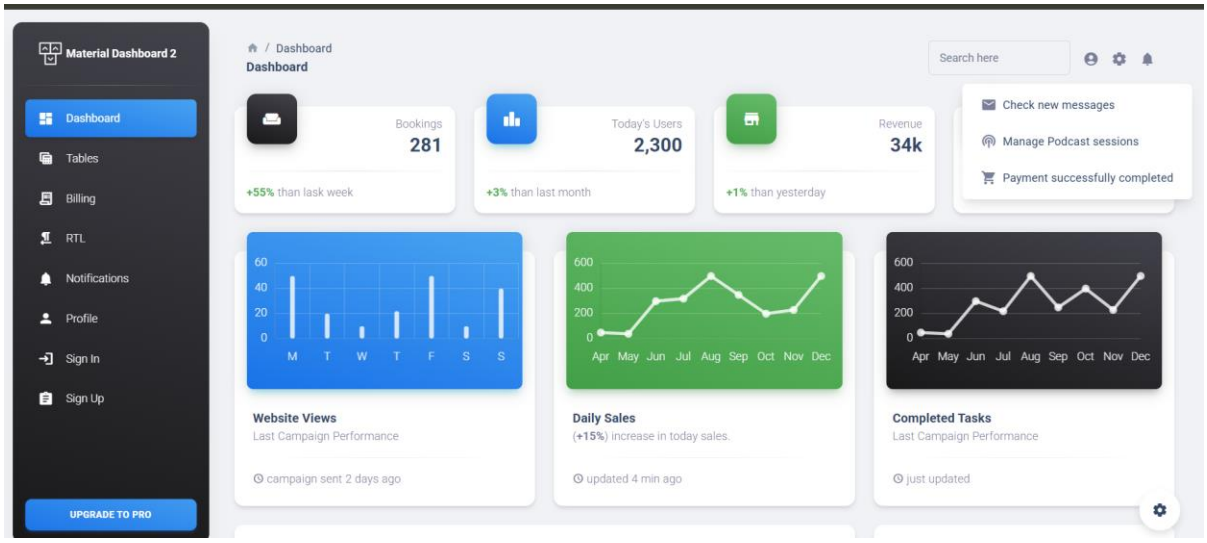
[7] Create a Responsive Dashboard with Angular Material and ng2-Charts *Smashing Magazine, 2020.* Demonstrates how to build responsive dashboards using Angular Material and ng2-charts, leveraging schematics to streamline development. [8] Angular Dashboard Libraries: Which One to Use in 2025? *Luzmo Blog, 2023.* Analyzes various Angular dashboard libraries, providing insights into their features and suitability for different project requirements.

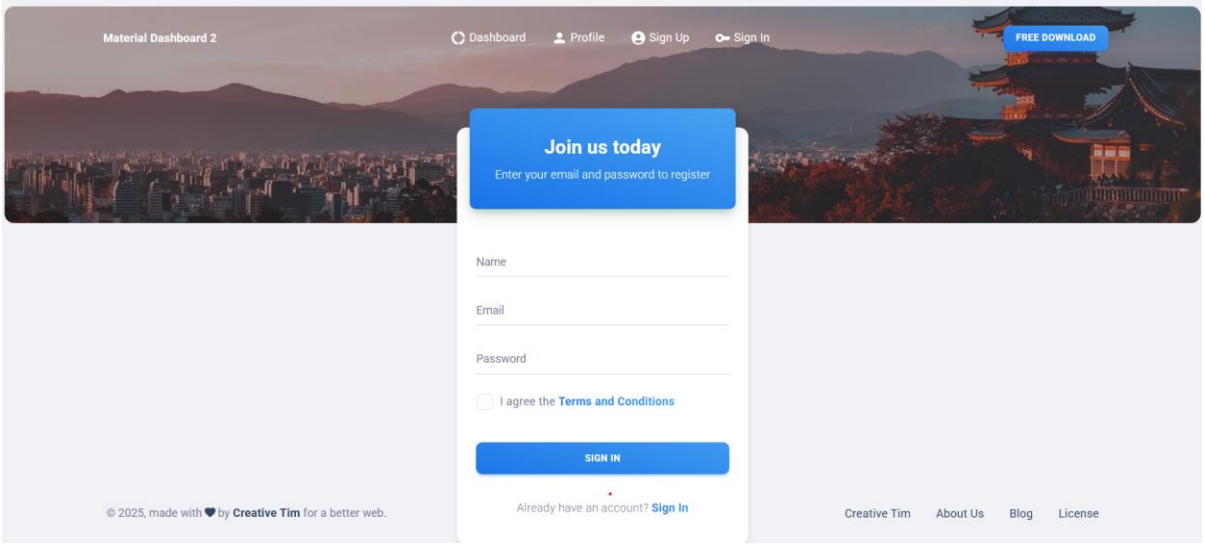
APPENDIX

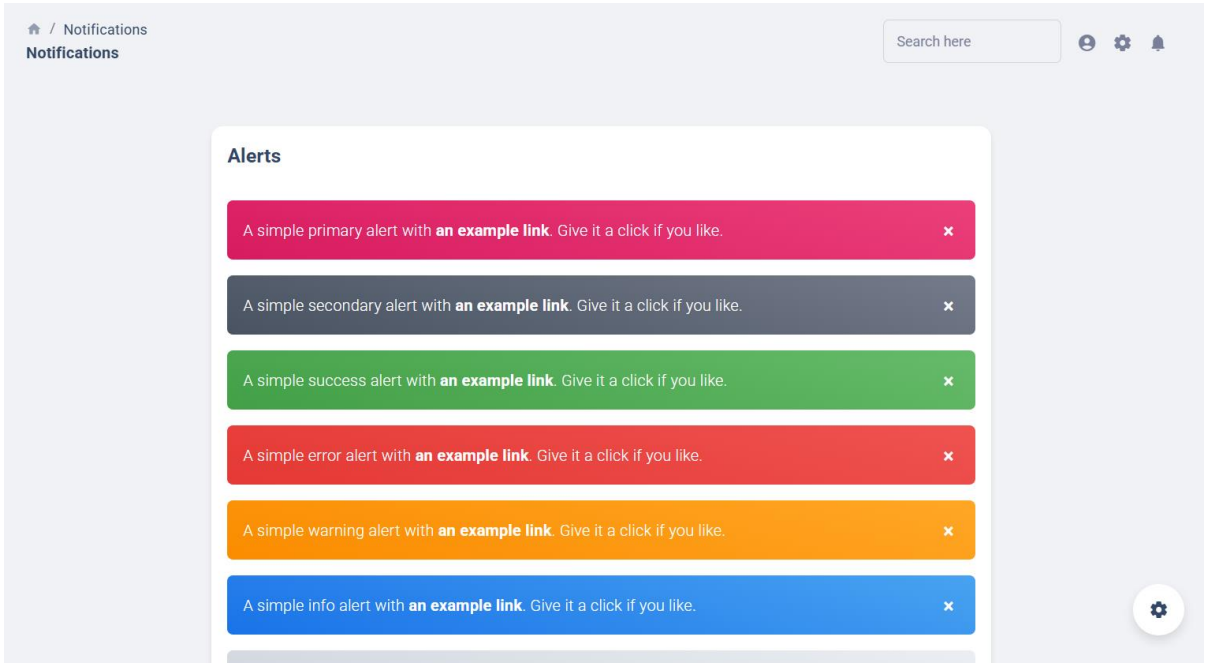

Material Dashboard 2
Dashboard
Tables
Billing
RTL
Notifications
Profile
Sign In
Sign Up
UPGRADE TO PRO
Dashboard
Search here
Bookings
281
+55% than lask week
Today's Users
2,300
+3% than last month
Revenue
34k
+1% than yesterday
Followers
+91
Just updated
Website Views
Last Campaign Performance
campaign sent 2 days ago
Daily Sales
(+15%) increase in today sales.
updated 4 min ago
Completed Tasks
Last Campaign Performance
just updated


Material UI Configurator
See our dashboard options.
Sidenav Colors
Sidenav Type
Choose between different sidenav types.
DARK
TRANSPARENT
WHITE
Navbar Fixed
Light / Dark
VIEW DOCUMENTATION
Star
2,947
Thank you for sharing!
TWEET
SHARE


Material Dashboard 2
Dashboard
Tables
Billing
RTL
Notifications
Profile
Sign In
Sign Up
UPGRADE TO PRO


Material Dashboard 2
Dashboard
Tables
Billing
RTL
Notifications
Profile
Sign In
Sign Up
UPGRADE TO PRO
/ Dashboard
Dashboard
Search here
Bookings
281
+55% than lask week
Today's Users
2,300
+3% than last month
Revenue
34k
+1% than yesterday
Check new messages
Manage Podcast sessions
Payment successfully completed
60
40
20
0
M T W T F S S
Website Views
Last Campaign Performance
campaign sent 2 days ago
600
400
200
0
Apr May Jun Jul Aug Sep Oct Nov Dec
Daily Sales
(+15%) increase in today sales.
updated 4 min ago
600
400
200
0
Apr May Jun Jul Aug Sep Oct Nov Dec
Completed Tasks
Last Campaign Performance
just updated


Material Dashboard 2
Dashboard
Profile
Sign Up
Sign In
FREE DOWNLOAD
Join us today
Enter your email and password to register
Name
Email
Password
I agree the Terms and Conditions
SIGN IN
Already have an account? Sign In
© 2025, made with ♥ by Creative Tim for a better web.
Creative Tim
About Us
Blog
License


/ Notifications
Notifications
Search here
Alerts
A simple primary alert with an example link. Give it a click if you like.
A simple secondary alert with an example link. Give it a click if you like.
A simple success alert with an example link. Give it a click if you like.
A simple error alert with an example link. Give it a click if you like.
A simple warning alert with an example link. Give it a click if you like.
A simple info alert with an example link. Give it a click if you like.