Republic of Iraq
Ministry of Higher Education
Babylon University
Information Technology
Software Department

Design and implementation of software package to coordinate supervisor meetings

This Research

Submitted to the College of Information Technology / University of Babylon as Partial Fulfillment of the Requirements for the Degree of Baccalaureus in Software Department

Research by

Mohammed Al-Mustafa Adel Abd

Supervisor

Prof. Dr. Ahmed Saleem Abbas

جمهورية العراق
كلية تكنولوجيا المعلومات
قسم البرمجيات

Republic of Iraq
Information Technology collage
Dept. of software

بحث التخرج لطلبة المرحلة الرابعة

للعام الدراسي ٢٠٢٢ - ٢٠٢٣

أسم الطالب الثلاثي : محمد المصطفى عادل عبد

أسم التدريسي المشرف : ا.د. أحمد سليم الصفار

رئيس القسم : ا.د. احمد سليم الصفار

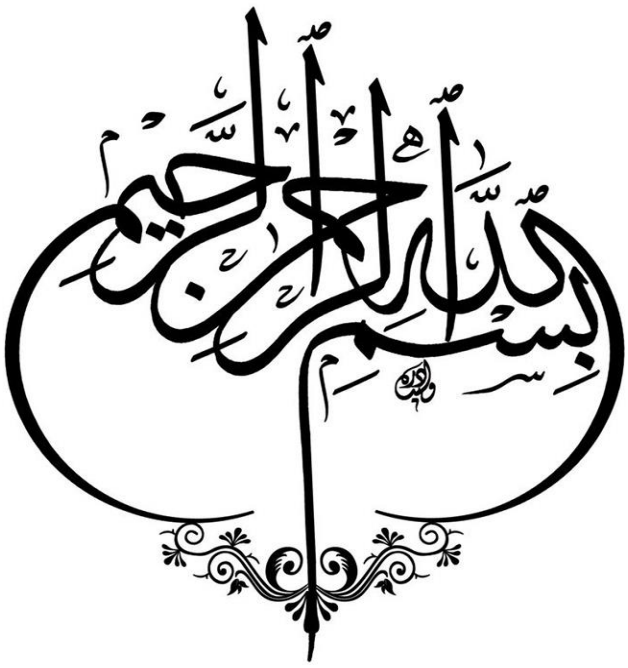
عنوان البحث : تصميم و تنفيذ تطبيق لأدارة اجتماعات المشرفين

التاريخ : ١٥\٢٠٢٣

توقيع التدريسي المشرف :

التاريخ : ١٥\٢٠٢٣

توقيع رئيس القسم :



(قَالَ رَبِّ اشْرَحْ لِي صَدْرِي * وَيَسِّرْ لِي أَمْرِي * وَاحْلُلْ عُقْدَةً مِنْ لِسَانِي * يَفْقَهُوا قَوْلِي)

صَدَقَ اللَّهُ الْعَلِيُّ الْعَظِيمُ

سورة طه (٢٥- ٢٨)

الأهداء

أهدي هذا المشروع، وبكل الحب والتقدير، إلى والدي العزيزين الذين وقفا بجانبي في كل خطوة أخطوها في هذه الرحلة. لكما، أمي وأبي، كل الشكر على الدعم الذي لا ينضب والحب الذي لا يعد ولا يحصى.

أهديه أيضاً لعائلتي العزيزة التي أعطتني القوة والشجاعة لتحقيق أحلامي. لكم، أقدم هذا العمل كعلامة صغيرة على مدى امتناني لكم.

لأصدقائي، الذين ساندوني وساهموا في تحقيق هذا النجاح، أهدي هذا المشروع لكم. إنكم كنتم العون والسند في الأوقات الصعبة والسهلة، وأنا مدين لكم بالكثير.

وأخيراً، وليس آخراً، أهدي هذا المشروع لكل الأساتذة الذين وجهوني ودعموني في رحلتي الأكاديمية. شكراً لكم على المعرفة التي قدمتموها لي والتشجيع الذي أعطيتموه لي لأصل إلى هذا المكان.

هذا المشروع هو نتيجة الحب والدعم الذي تلقيته منكم جميعًا. أتمنى أن يعكس العمل الذي قمت به في هذا المشروع الاحترام والتقدير الذي أشعر به لكل واحد منكم.

الشكر و التقدير

أود أن أبدأ بالشكر العميق والتقدير الصادق لمشرفي الرائع، الدكتور أحمد سليم عباس. لقد كانت توجيهاتك، مشورتك، ودعمك الثابت مصدر إلهام لا يقدر بثمن خلال هذا العمل. أعتز بالفرصة التي منحتني إياها للتعلم من خبرتك العميقة والثرية.

لا يمكنني تجاهل الدور الحاسم الذي لعبته عائلتي في هذا الإنجاز. حبكم ودعمكم كانا القوة الدافعة وراء كل جهد بذلته. في أوقات الشك والتحدي، كنتم تشعلون شرارة الأمل في قلبي. أشكركم لثقتكم العميقة بي ولدعمكم المستمر لطموحاتي.

لكل أصدقائي الذين ساندوني ودعموني، لكم جزيل الشكر. كانت تشجيعاتكم والدعم المستمر مصدر قوة لا يمكن التغاضي عنه.

وأخيرًا، أود أن أشكر كل من ساهم ودعمني في هذه المسيرة، بدءًا من الأقران الأكاديميين، والمحاضرين، والموظفين . أنتم جميعًا جزء لا يتجزأ من هذا الإنجاز، ولن أنسى أبدًا المساهمات القيمة التي قدمتموها.

ABSTRACT

This report provides a comprehensive review of a sophisticated web application built using ASP.NET Core 6.0, Entity Framework Core, ASP.NET Core Identity, and SQL Server databases. The application includes a robust authentication system, allowing users to sign in via email and password, and an administrative interface for managing user accounts. User roles, such as admin, student, and supervisor, offer distinct capabilities, ranging from managing other users, scheduling meetings, to tracking tasks. The system ensures secure password management with automated generation and email delivery. Moreover, a detailed profile system allows users to update personal information, schedule free time, and change passwords. The application also features a comprehensive meeting and supervision system, including elements of task management, mutual availability matching, and attendance recording. The report examines the user interface, highlighting components like meeting cards and detailed reports, that provide essential information to users. Further, it explores advanced reporting tools and system logging, crucial for effective administration and system optimization. The application demonstrates the potential of ASP.NET Core in creating secure, scalable, and feature-rich web applications.

Table of Contents

Chapter One
Introduction

INTRODUCTION

In the realm of higher education, productive communication and effective coordination between students and their academic supervisors are integral to success. However, traditional systems and methods of arranging meetings and managing tasks often fall short in providing a streamlined, efficient, and transparent experience. Addressing this challenge requires an innovative, digital solution that can simplify and enhance these interactions.

Leveraging the power of ASP.NET Core and SQL Server, a comprehensive software package has been developed to redefine this crucial aspect of the educational journey. Utilizing .NET 6 and Entity Framework Core, the platform offers a robust, reliable, and scalable solution that addresses the unique needs of students, supervisors, and administrators.

1.1 Background:

Historically, the process of managing meetings and supervisions has often been manual and time-consuming, riddled with inefficiencies and miscommunication. The lack of a centralized system made it challenging to track progress, manage tasks, and maintain transparency. In light of these issues, the need for a user-friendly, comprehensive, and efficient system emerged - one that could automate administrative tasks, provide a platform for efficient communication, and ensure a seamless user experience.

1.2 Advantages:

- **Efficiency**: By automating various administrative tasks, such as meeting scheduling and task assignment, the system significantly enhances efficiency, allowing users to focus more on their core academic or administrative duties.
- **Accessibility**: The platform's digital nature ensures that users can access it anytime, anywhere, providing a level of convenience and flexibility that traditional systems cannot match.
- **Transparency**: With features allowing comprehensive tracking and reporting, the system promotes transparency, enabling users to monitor progress, access relevant information, and make data-driven decisions.

- **Customizability**: The system's design accommodates customizable parameters, ensuring it caters to the unique needs and preferences of various institutions.
- **Security**: The software package ensures robust data security with stringent authentication mechanisms and secure data handling practices.
- **Scalability**: Designed with scalability in mind, the platform can accommodate an increasing number of users without compromising performance, making it suitable for institutions of varying sizes.

1.3 Aims:

- To develop a comprehensive user management system with specific roles that can be utilized in an educational or supervisory context.
- To streamline the process of user account creation and management by automatically generating secure passwords and facilitating easy updates to account settings.
- To enable efficient scheduling and management of meetings between supervisors and students, taking into account their respective availability.
- •To enhance the visibility of tasks, user information, and meetings through intuitive user profiles and meeting cards.
- To support system administrators in managing users and generating useful reports for analysis and decision making.
- To implement an effective logging and tracking system to monitor user activities and system performance.

1.4 Objectives:

- Implement a sign-in process using email and password, with three user types (admin, student, supervisor).
- Automate password generation and emailing of account credentials during account creation.
- Facilitate user account updates including personal information, profile image, and availability.
- Enable the assignment and management of student-supervisor relationships by the admin.

- Build a system for scheduling meetings between supervisors and students based on mutual availability, with features to add meeting descriptions and tasks.
- Create user interfaces for meeting viewing, task tracking, and attendance registration.
- Provide comprehensive user profiles showing relevant information and activities.
- Design and generate insightful reports for both general and advanced views, providing information on absence counts, active student counts, meeting histories, etc.
- Utilize Serilog and Seq for logging user activities and system performance, aiding system maintenance and improvement.
- Optimize system performance, ensuring responsive page loads and data requests.

1.5 Challenges:

Despite the significant advantages, the development of the software package was not without its challenges. Ensuring the system was user-friendly, secure, and reliable required careful design and rigorous testing. Accommodating the diverse needs of various user roles while maintaining a consistent and intuitive user experience posed a considerable challenge. Moreover, ensuring the system could scale efficiently to cater to an expanding user base while maintaining optimal performance required a robust and flexible architecture.

In conclusion, the integration of ASP.NET Core, SQL Server, and .NET 6 has led to the development of a transformative software package. This package aims to revolutionize the way meetings and supervisions are managed in higher education. By fostering efficient communication, transparent reporting, and effective coordination, this digital platform promises to significantly enhance the educational experience for students, supervisors, and administrators alike. Through continuous refinement and improvement, it aspires to set a new standard in educational coordination and management.

1.6 Integration and Collaboration:

One of the key elements of the system is its ability to foster a collaborative environment. It acknowledges the different roles in an educational institution - students, supervisors, and administrators - and provides functionalities tailored to

each. This approach ensures a harmonious integration of all parties involved, enhancing the overall efficiency of academic pursue

1.7 User Experience:

Prioritizing user experience, the platform is designed to be intuitive and easy to navigate. Whether it's scheduling a meeting, managing tasks, or viewing supervision details, every process is streamlined to ensure users can perform their required tasks with minimal effort. Furthermore, the system provides real-time notifications and updates, keeping users informed about upcoming meetings, task deadlines, and other important information.

1.8 Security and Privacy:

One of the key aspects of the platform is its strong emphasis on security and privacy. Leveraging the robust security features of ASP.NET Core, the system provides secure authentication mechanisms, ensuring only authorized users can access sensitive information [1]. User passwords are automatically generated with a mix of uppercase, lowercase, numbers, and symbols, providing an added layer of security. Furthermore, administrators have the power to manage user roles, providing an extra degree of control over the access to various system features.

1.9 Adaptability:

The system's adaptability is another aspect that sets it apart. Its ability to adjust to the varying demands of different educational institutions is a significant advantage. Parameters like the maximum number of students a supervisor can handle, the duration of supervision extensions, and report settings can all be customized based on the specific needs of the institution. This adaptability ensures the platform remains relevant and effective across different contexts and settings.

1.10 Performance and Scalability:

Developed using .NET 6 [2], the latest major release of Microsoft's open-source platform, the system is built for performance. It is designed to handle an increasing number of users without compromising on speed or reliability. This scalability is critical for institutions looking to grow and expand their student base. Furthermore, using Serilog for logging allows for thorough performance tracking and system load analysis, providing valuable insights for further optimization and improvements.

Chapter Tow
Theoretical Background

Theoretical Background

Software requirements

Software requirements refer to the specifications that detail what a software system or application should do, how it should perform, and what constraints it must operate within. They serve as the blueprint for the design and development of a software product.

2.1 Functional Requirements

Functional requirements are features that allow the system to function as it was intended. Put another way, if the functional requirements are not met, the system will not work. Functional requirements are product features and focus on user requirements , the software functional requirements as flowing.

- **User Authentication and Authorization:**
  - The system is required to authenticate users through email and password.
  - The system should support the creation of new user accounts, differentiated into three types: admin, student, and supervisor.
- **User Management:**
  - The system needs to provide admin users with the ability to manage other users, including search functionality by name.
  - User profile settings should be editable, including attributes like full name, phone number, email address, user role, stage, and profile picture.
  - The system must allow users to set and modify their free days and hours within the week.
  - Users should be able to change their password, and admins should have the authority to reset passwords.
- **Student-Supervisor Management:**
  - Admin users should be able to assign students to supervisors for supervision.
  - Admins should have the ability to extend the supervision end date and terminate supervision relationships.

- **Meeting Management:**
  - Supervisors should be able to create, view, and delete meetings with their students.
  - The system needs to prevent meeting overlaps.
- **Task Management:**
  - Tasks should be assignable to meetings by supervisors.
- **Reporting:**
  - The system should generate detailed reports for admins, supervisors, and students.

2.2 Non-functional Requirements

Non-functional requirements specify how the system should do it. Non-functional requirements do not affect the basic functionality of the system, the software Non-functional requirements as flowing.

- **Security:**
  - For new user accounts, the system should auto-generate a robust password and securely deliver it to the user's registered email.
- **Performance:**
  - During data request or page access operations, the system should display a loading indicator.
  - The system should log user activities and system performance metrics using Serilog.
- **Usability:**
  - Uploaded user profile images must not exceed 300kb in size post front-end compression.
  - The system should present information in an easily digestible format, like displaying meetings as cards.
- **Scalability:**
  - The system should allow modification of certain parameters like the maximum students a supervisor can handle, supervision extension time, etc., through a system settings file.

2.3 Domain Requirements

Domain requirements reflect the environment in which the system operates so, when we talk about an application domain we mean environments, Domain requirements can be functional or nonfunctional.

- The system must support distinct user roles, each with unique access rights and functionalities - admin, student, supervisor.
- The system should allow for supervision setup with a defined duration, including the option for extension.
- The system should facilitate scheduling of meetings, taking into account the availability of both students and supervisors.
- The system must provide a task management system, allowing tasks to be assigned, completed, or marked as incomplete.
- The system should generate comprehensive reports, offering insights into users, meetings, tasks, and more.

Chapter Three
Design and Implementation

Design and Implementation

3.1 Users Use Cases

A use case is a methodology used in system analysis to identify, clarify and organize system requirements [1]. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The method creates a document that describes all the steps taken by a user to complete an activity, see **Figure 1** for more information

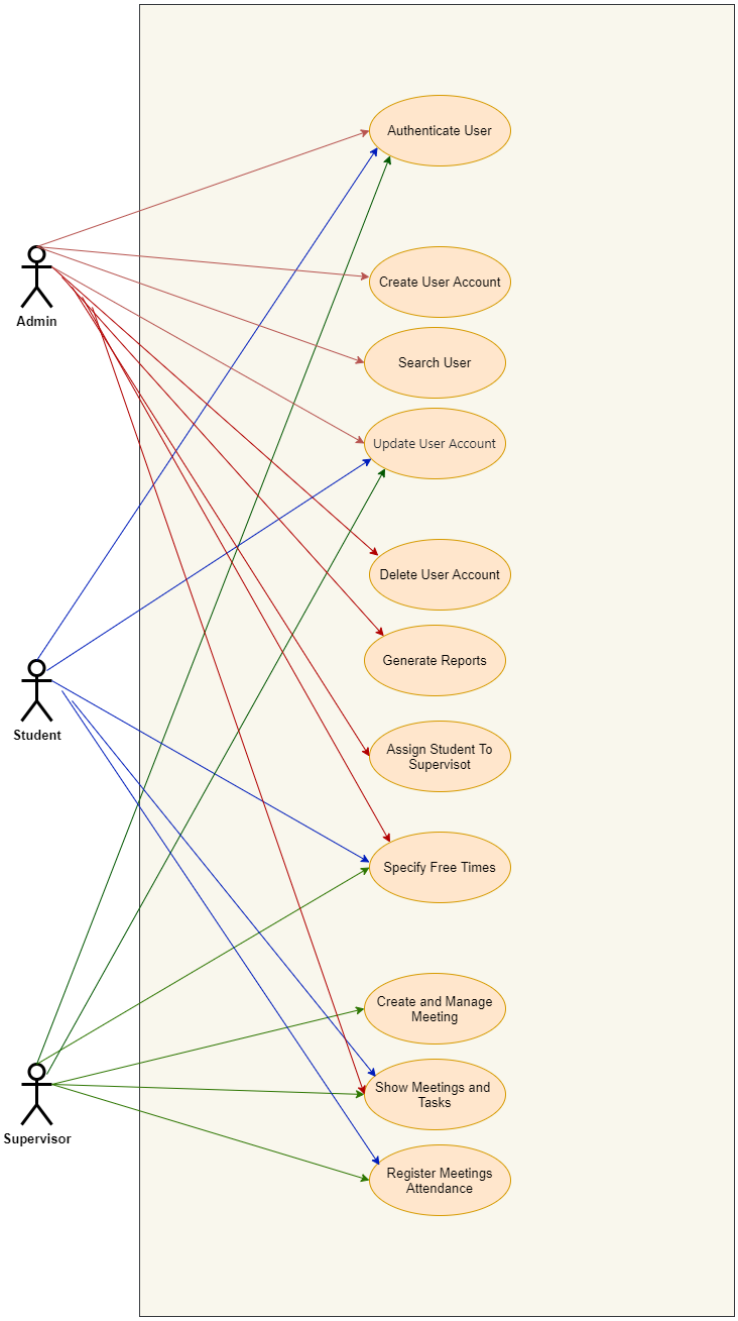


Figure 1 Users User Cases

3.2 State Transition Diagram (STD)

A diagram that indicates the possible states of a finite-state automaton and the allowable transitions between such states. There are several different dialects of STDs. Each one depicts the states, transitions, and event(s) that can cause each, see **Figure 2** , **Figure 3** and **Figure 4** for more information

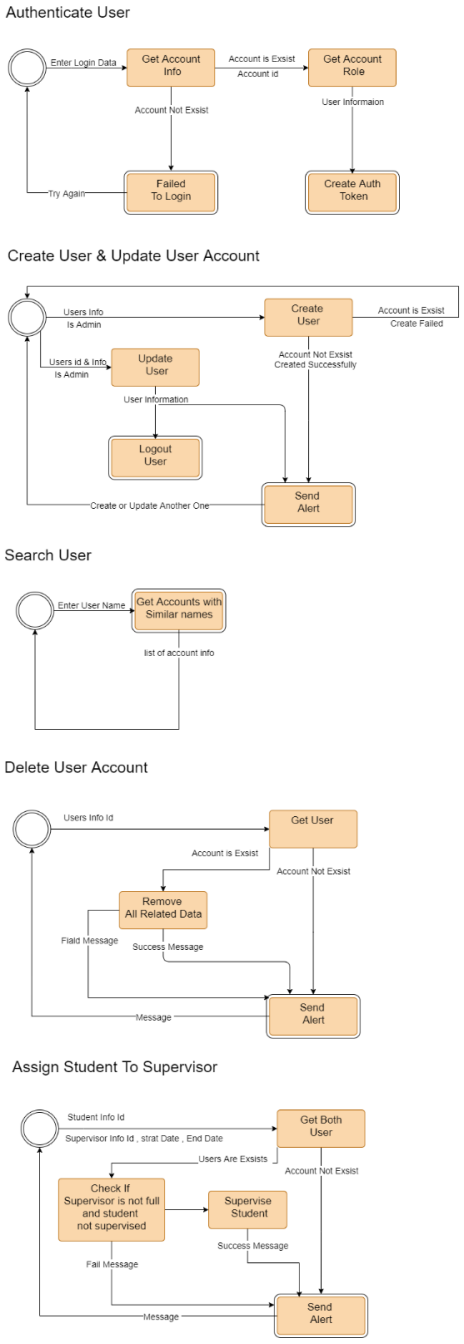


Figure 2 STD (Authenticate User, Create & Update User Account , Search for Name, Delete User Account , Assign Student To Supervisor)

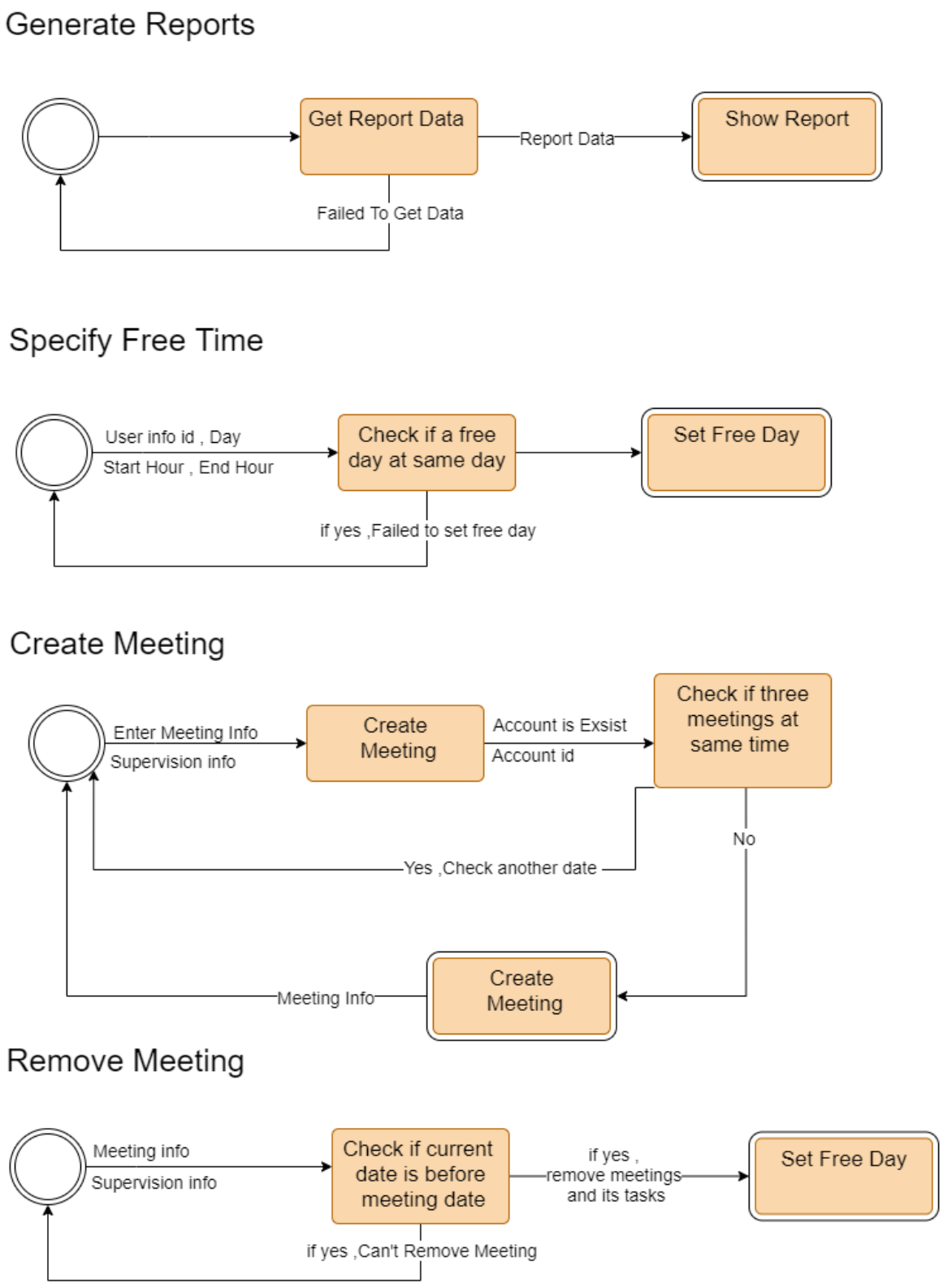


Figure 3 STD (Generate Reports , Specify Free Time , Create Meeting , Remove Meeting)

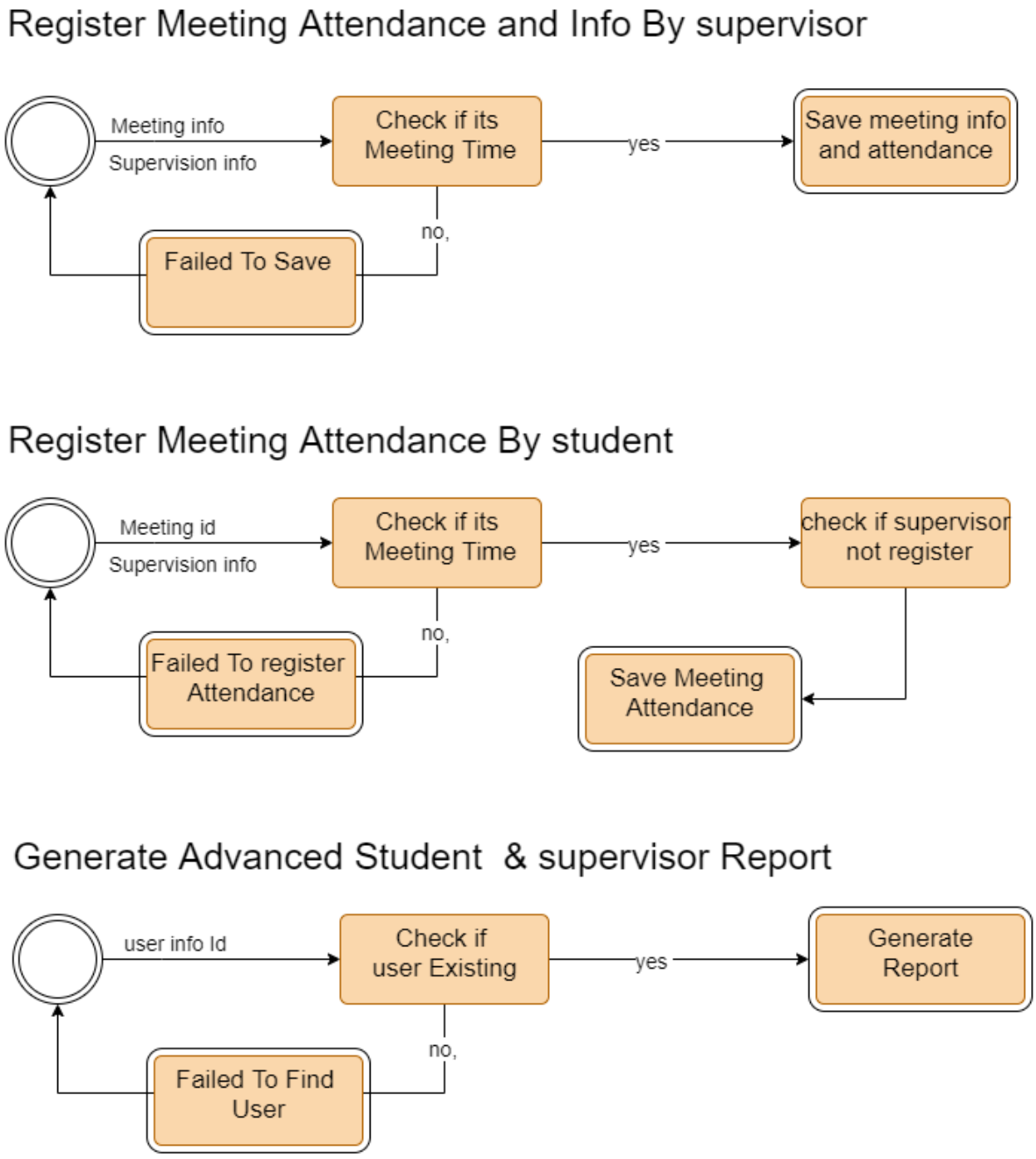


Figure 4 STD (Register Meetings Attendance , and Info By Supervisor , Register Meeting Attendance By Student , Generate Advanced Reports)

3.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both, its contains three levels (0,1,2) .

3.3.1 Level Zero

It is also known as fundamental system model, or context diagram represents the entire software requirement as a single bubble with input and output data, see **Figure 5** for more information

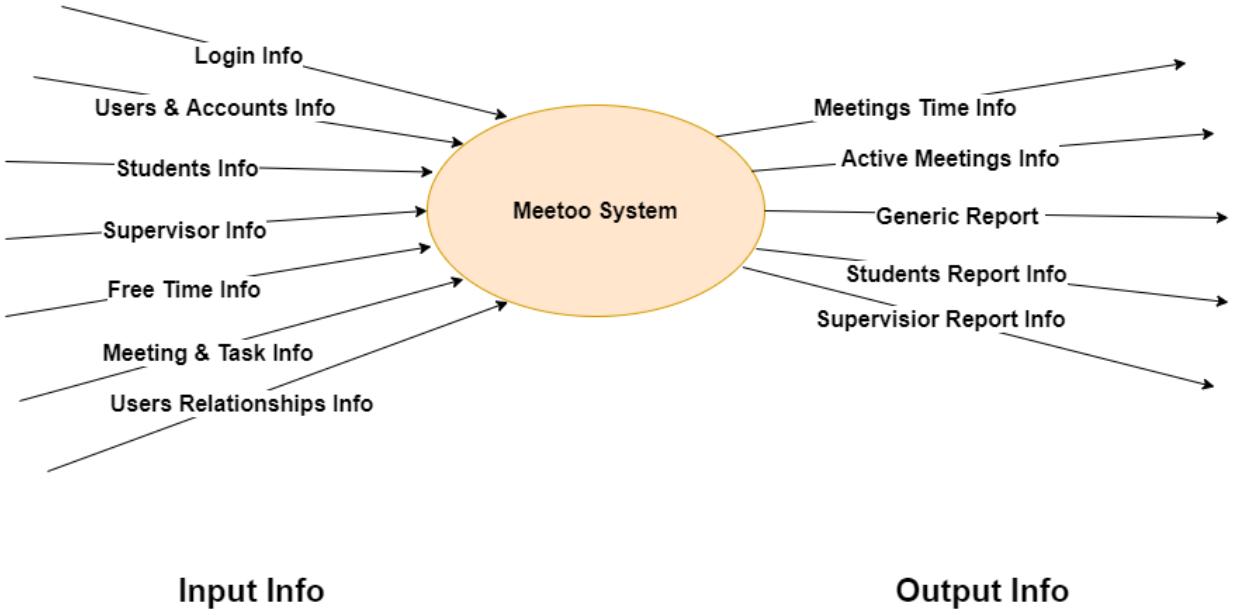


Figure 5 DFD Level Zero

3.3.2 Level One

In 1-level DFD, a context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main objectives of the system and breakdown the high-level process of 0-level DFD into subprocesses, see **Figure 6Figure 6** for more information

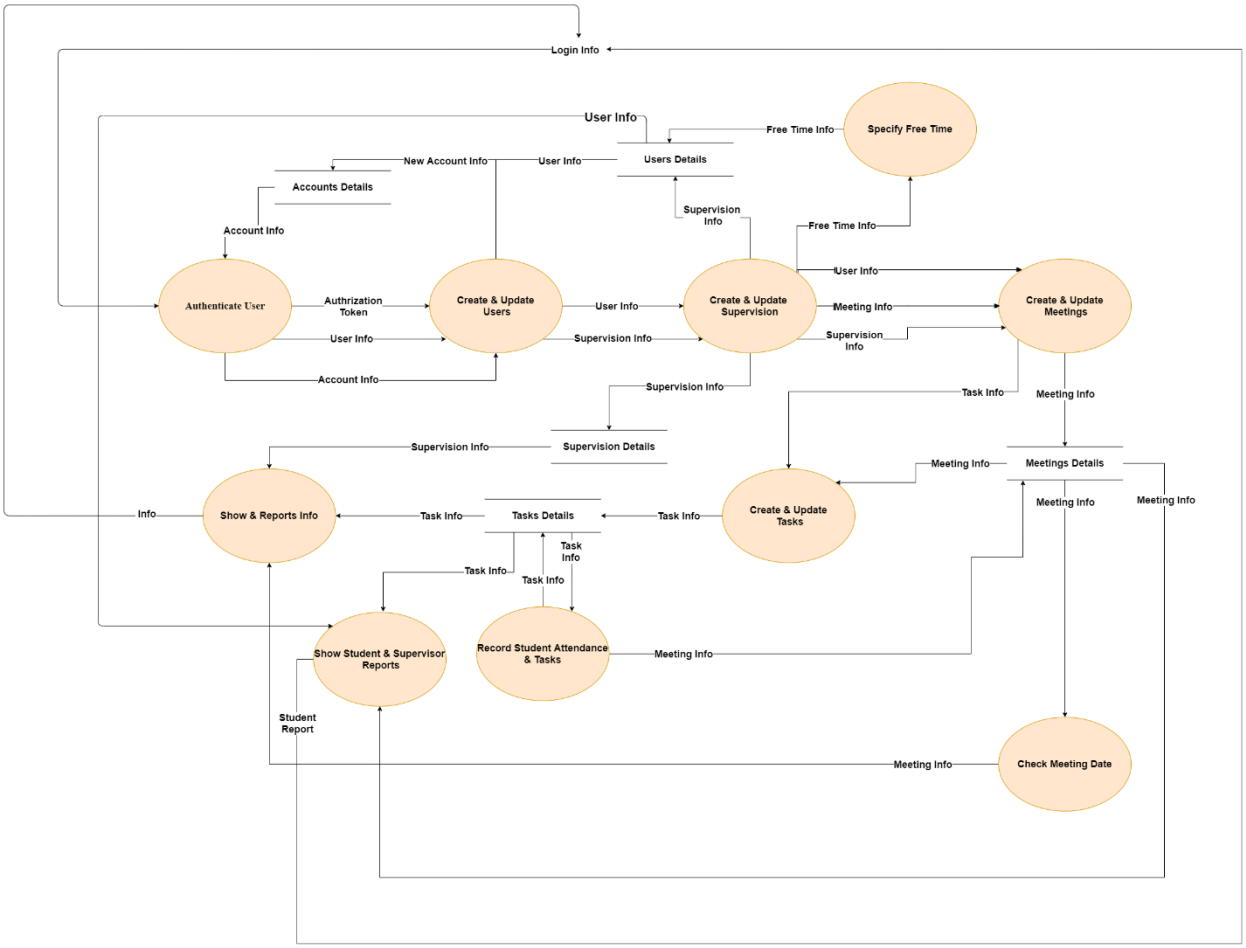


Figure 6 DFD Level One

3.3.3 Level Tow

2-level DFD goes one process deeper into parts of 1-level DFD. It can be used to project or record the specific/necessary detail about the system's functioning, see **Figure 7** for more information.

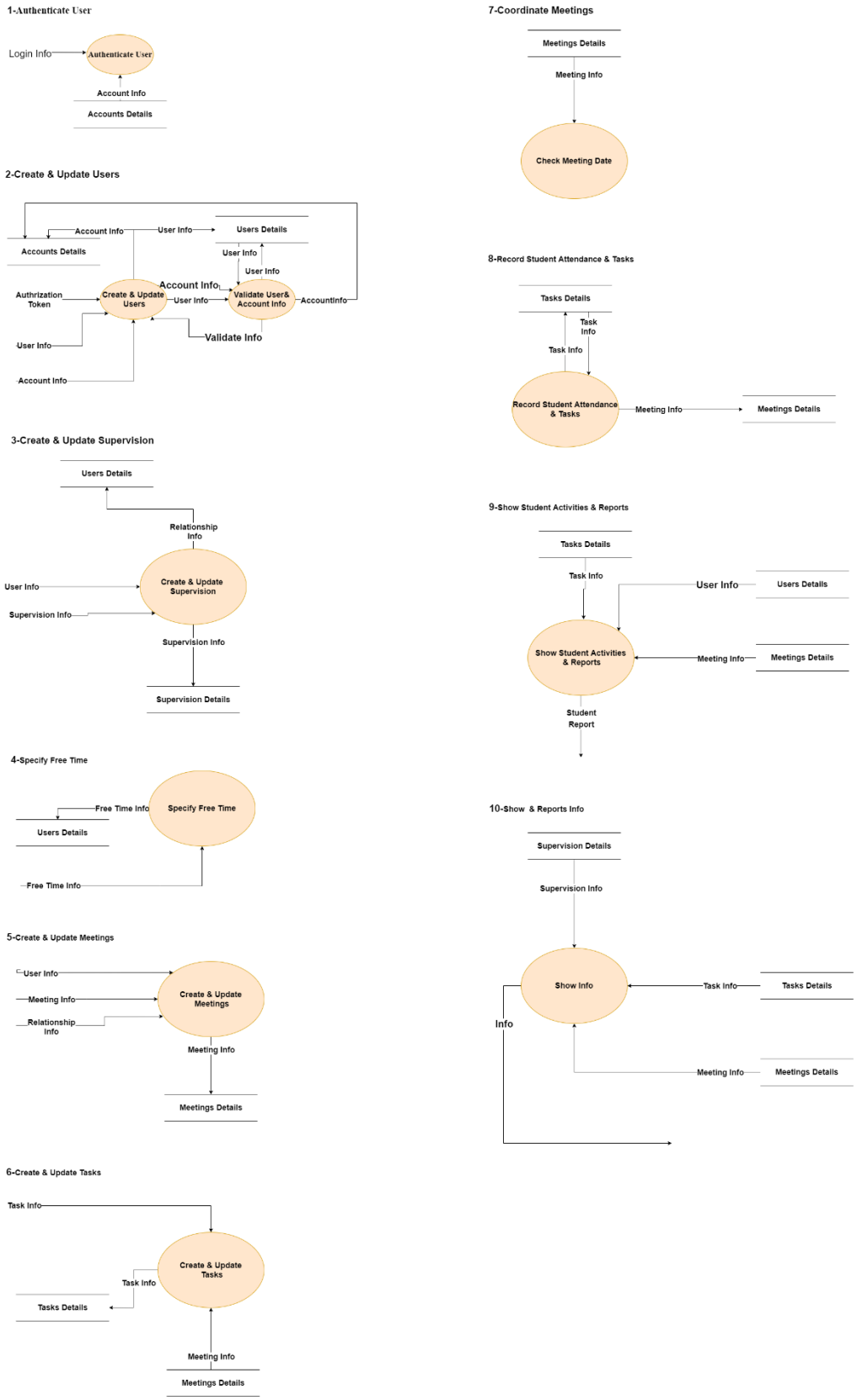


Figure 7 DFD Level Tow

3.4 Entity Relationship Diagram (ERD)

An Entity Relationship Diagram (ERD) [4] is a visual representation of different entities within a system and how they relate to each other. It is a tool used to design and model relational databases, and shows the logical structure of the database. ER diagrams use symbols to represent entities, attributes, and relationships, which help to illustrate the relationships between the entities in the database. ER diagrams are commonly used in software engineering and database design to help developers and stakeholders understand and design complex databases .

There are three basic elements in an ER Diagram: entity, attribute, relationship. There are more elements which are based on the main elements. They are weak entity, multi valued attribute, derived attribute, weak relationship, and recursive relationship. Cardinality and ordinality are two other notations used in ER diagrams to further define relationships , see **Figure 8** for example

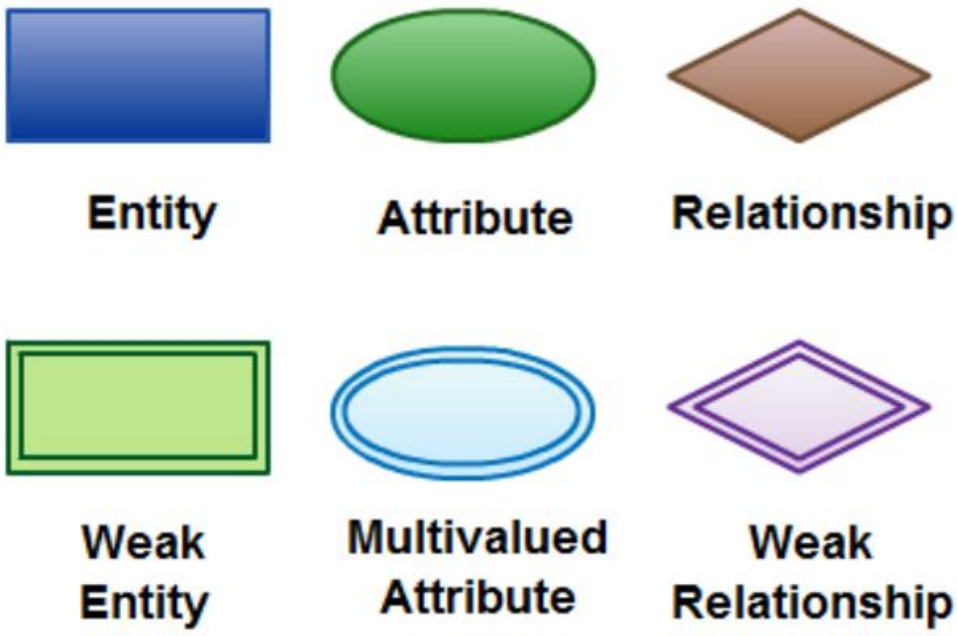


Figure 8 ERD Symbols [4]

Our Entity Relationship Diagram (ERD) in **Figure 9**

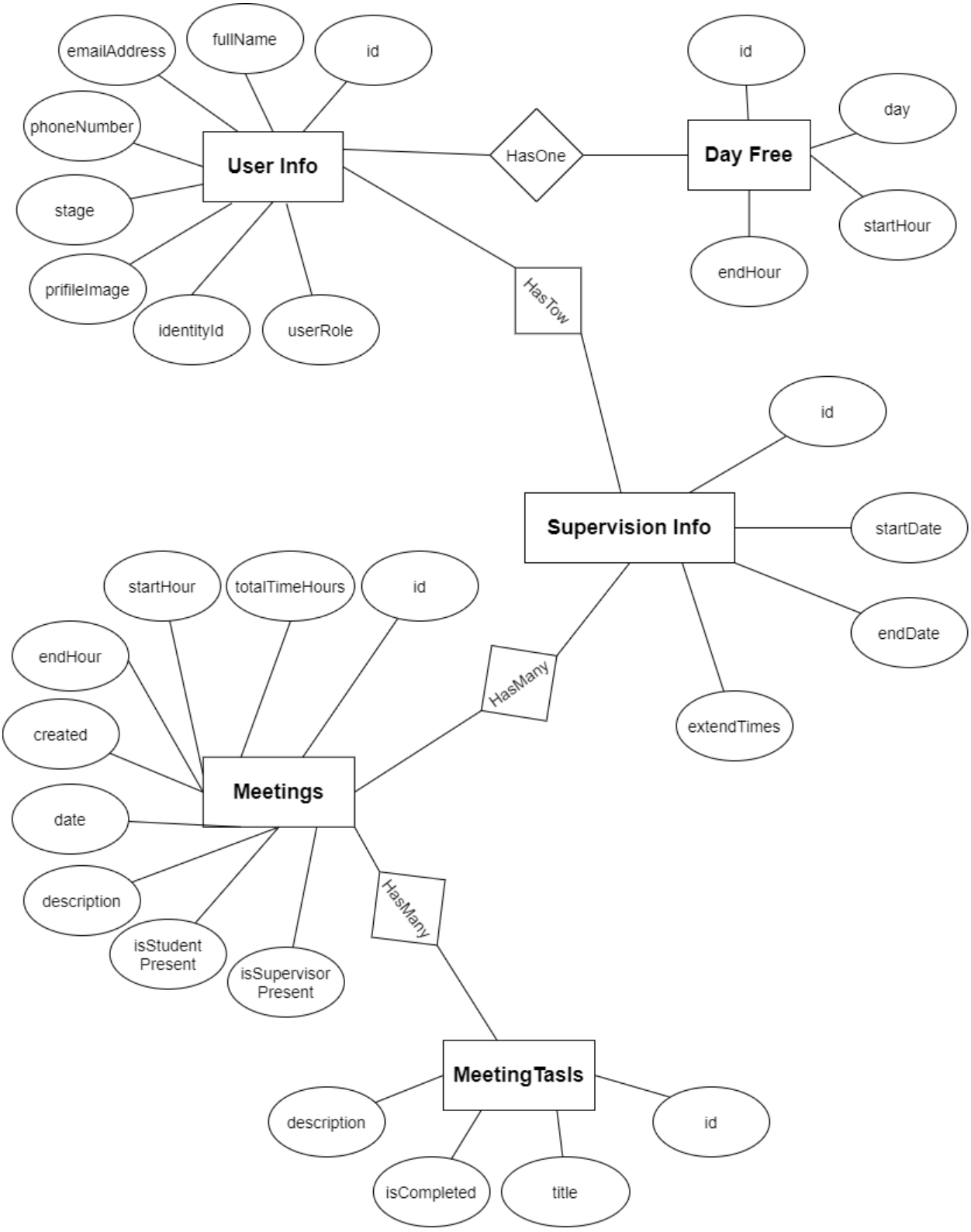


Figure 9 Entity Relationship Diagram (ERD)

Implementation

3.5 Software Architecture Pattern:

As a matter of fact, software architectural [5] patterns help specify the primary characteristics and behaviors of a software. To select the appropriate architectural pattern that meets your specific business goals, understanding the characteristics, strengths, and weaknesses of architectural patterns is a requisite issue. Even though a number of architectural patterns have been recommended in software development, the layered architecture pattern is well-known due to ease of development and test.

3.5.1 Layered architecture pattern:

The concept of layered architecture or n-tier architecture, a prevalent design pattern in contemporary software development. It elaborates on the division of an application into distinct, interdependent layers with distinct responsibilities.

Overview of Layers:

- **Presentation Layer:** This section explains how the presentation layer interacts with users, providing user interfaces, and API endpoints to capture their inputs and feedback.
- **Business Layer:** Discusses the critical role of the business layer, also known as the domain or service layer, in housing the business logic of the software. Here, the report elaborates on the execution of data manipulations and algorithms following the software's domain-specific rules.
- **Repository Layer:** This part explores how the repository layer serves as an intermediary between the business and data layers, abstracting data storage and retrieval mechanisms to streamline interaction with data.
- **Data Layer:** This section focuses on the data layer's role in data persistence, involving databases, file systems, and external APIs.

3.5.2 Benefits of Layered Architecture:

This section outlines the main advantages of layered architecture, including:

- **Separation of Concerns:** Layered architecture simplifies understanding, maintaining, and developing systems by dividing them into clear, separate sections.

- **Independent Testing and Scaling:** Each layer can be tested, optimized, and expanded independently, enhancing the system's overall reliability and performance.
- **Enhanced Reusability:** The architecture supports modification or replacement of a layer without impacting others, fostering reusability.
- **Flexibility:** Layered architecture provides flexibility by allowing different technologies or strategies to be used for each layer, based on their unique needs.

Each Layer contains Interfaces as infrastructure and there implantation so it will be easy to switch between deferent implantation for same interface without make any change in code that user that interface.

Closed Layered Architecture

In a closed layered architecture, a layer can only call the next layer immediately below it , see **Figure 10**.

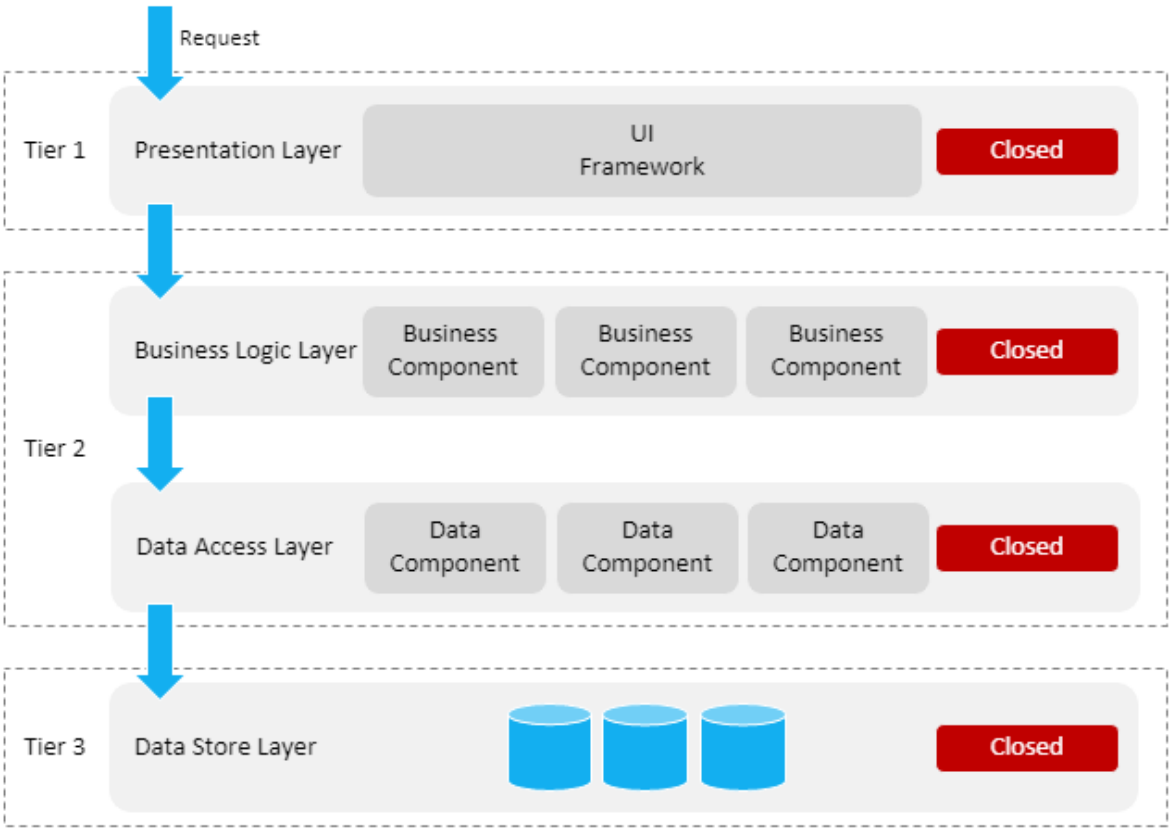


Figure 10 Closed Layered Architecture [6]

3.5.2 Tools and Technologies Used

In the completion of this project, various tools and technologies have been utilized across different layers of the application to ensure optimal functionality and efficient coding practices. They are categorized as follows:

1. **C# (Primary Programming Language):** C# is a multi-paradigm programming language developed by Microsoft as part of its .NET initiative. Offering simplicity, expressiveness, and a strong type system, C# allows developers to create a wide variety of secure and robust applications that run on the .NET platform. The language itself is object-oriented, providing key features like encapsulation, inheritance, and polymorphism. It's designed to be easy to use and efficient, making it a highly popular language for various applications from web services to desktop applications, games, and even mobile apps. It's also fully integrated with the .NET library, which provides thousands of classes and routines to developers, massively speeding up the development process and increasing productivity. Moreover, C# includes features like garbage collection, exception handling, and support for asynchronous programming, which make it a strong choice for both beginner and experienced developers.

2. **.NET Core 6 & ASP.NET Core MVC** [7]**:** .NET Core is a free, open-source, general-purpose development framework maintained by Microsoft and the .NET community on GitHub. It's cross-platform, supporting Windows, macOS, and Linux, and can be used in device, cloud, and embedded/IoT scenarios. .NET Core 6 is part of the .NET 6 release, providing developers with improvements in performance, deployment, and continued runtime improvements. On the other hand, ASP.NET Core MVC [8] is a lightweight, testable framework built on top of .NET Core for building web applications. It follows the Model-View-Controller design pattern, ensuring the separation of application logic and user interface. This separation makes managing complex applications easier because developers can focus on one aspect of implementation at a time. The loosely coupled nature of the MVC pattern also improves the testability of applications and provides a clean, organized way of coding.

3. **Dependency Injection:** Dependency Injection (DI) [8] is a software design pattern that helps in achieving loose coupling between objects and their dependencies. Rather than hard-coding the dependencies, DI provides a way

to supply the dependent objects from the outside. This pattern is essential for building scalable, testable, and maintainable applications. In the context of .NET Core, DI is built into the framework. When an application starts, it creates a container of services that are used throughout the application's lifetime. Whenever a service or application needs a dependent service, the .NET Core DI engine injects the required service automatically. This way, the DI pattern eliminates the need for manual wiring up of dependencies, improves code maintainability, and simplifies testing.

4. **Entity Framework Core:** Entity Framework (EF) [10] Core is a lightweight, extensible, and open-source version of the popular Entity Framework data access technology. As an Object-Relational Mapping (ORM) framework for .NET, it simplifies data access by allowing developers to work with databases using .NET objects, eliminating the need for most of the data-access code that developers usually need to write. EF Core supports a wide variety of databases, and it includes APIs for building queries, saving data, managing transactions, and tracking changes to objects. By reducing the need for writing and maintaining data-access code, EF Core improves the efficiency of database operations and enables developers to focus on the core functionality of their applications.

5. **ASP.NET Core Identity:** ASP.NET Core Identity [1] is a membership system that provides login functionality to ASP.NET Core applications. It supports user registration, authentication, authorization, roles, claims, tokens, password reset, account confirmation, two-factor authentication, and more. User identities are stored in a database, and password are hashed for security. ASP.NET Core Identity allows you to add login functionality to your application and makes it easy to customize data about the logged in user. It's an essential tool for managing user access and securing your ASP.NET Core applications.

6. **Humanizer Core:** Humanizer is a .NET library that helps in manipulating and displaying strings, enums, dates, times, timespans, numbers, and quantities in a human-friendly format. It fills the gap between system-like data structures and the way humans perceive information. For instance, it can transform numeric quantities into words, change casing, turn symbols into spoken phrases, and much more. This library can make your application's user

interface more friendly and intuitive, by ensuring that system or data-oriented information is presented in a way that is easy to understand for users.

7. **MailKit & MimeKit:** MailKit is a fully-featured and robust mail library used to send, receive and parse email from .NET applications using mail protocols such as IMAP, POP3, and SMTP. MimeKit is an accompanying library to MailKit, providing extensive email construction and parsing functionality. Together, these libraries offer a powerful, efficient, and easy-to-use system for handling email in .NET applications. They also include features for dealing with attachments, multipart messages, encoding/decoding mechanisms, and even encryption/decryption of emails.

8. **Serilog, Serilog Timings, Serilog Seq:** Serilog is a powerful structured logging [11] library for .NET applications. Unlike traditional logging libraries, which write only text messages, Serilog is able to capture complex data types and serialize them as structured data. When combined with storage and visualization systems, this allows for more meaningful analysis of logged data. The Serilog Timings library provides simple, readable timing blocks to measure and record methods, and the Serilog Seq sink allows log events to be viewed and queried in Seq, a structured log viewer. These tools together make it easy to capture, store, and analyze log data, helping to identify and diagnose problems quickly.

9. **OneOf:** OneOf is a C# library that provides discriminated unions, a concept often found in languages like F# and Rust. This feature allows a variable to hold one of several potential types, with the current type often governing the available operations and semantics. It's useful for scenarios where a method can return different types of responses. This aids in writing expressive, concise code that's easy to read and reason about.

10. **HTML:** HTML, or HyperText Markup Language, is the standard language used to create web pages. It's a markup language that tells a browser how to layout content. HTML structures information by tagging and categorizing content such as "heading", "paragraph", "table", and so forth. This allows browsers to present information in a structured and meaningful way. Moreover, HTML can embed scripts written in languages such as JavaScript, which affect the behavior and content of web pages. The inclusion of CSS (Cascading Style Sheets) with HTML gives web pages their look and formatting.

11. **CSS & Bootstrap:** CSS, or Cascading Style Sheets, is a style sheet language used for describing the look and formatting of a document written in HTML or XML. CSS [12] brings style to web pages by adjusting the colors, fonts, layout, and everything else that pertains to the look and feel. Bootstrap [12], on the other hand, is a free and open-source front-end framework for designing websites and web applications. It contains HTML and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. It aims to ease the development of dynamic websites and web applications by providing a well-structured, responsive grid system and a variety of plug-ins.

12. **JavaScript & JQuery:** JavaScript is the programming language of the web. It's an interpreted, high-level, multi-paradigm language that allows the creation of highly responsive interfaces, improves user interaction and provides dynamic functionality to a website without needing to reload pages. It's used for things like web form validation, user interactivity, animations, and much more. jQuery [12], on the other hand, is a fast, small, and feature-rich JavaScript library. It simplifies things like HTML document traversal and manipulation, event handling, animation, and Ajax with an easy-to-use API that works across a multitude of browsers.

13. **Material Design 2.2:** Material Design is a design system created by Google, intended to help teams build high-quality digital experiences for Android, iOS, Flutter, and the web. Material Design 2.2 provides design guidelines, components, tools, and best practices that reflect the latest in tech trends. It combines the classic principles of good design with the innovation of technology and science, providing a unified system that allows for a consistent user experience across platforms. Material Design's components are crafted to ensure better usability, accessibility, and interaction, making it easier for designers and developers to build beautiful, interactive user interfaces.

14. **Google Fonts & Font Awesome Icons:** Google Fonts is a library of over 800 free licensed font families and an interactive web directory for browsing these fonts. It provides a simple and robust platform for the discovery and use of fonts in a variety of projects, from print to web to mobile. It also includes APIs for conveniently using the fonts via CSS and Android. Font Awesome is a font and icon toolkit based on CSS and Less. It gives you scalable vector icons that

can instantly be customized in terms of size, color, drop shadow, and anything else that can be done with the power of CSS.

15. **Visual Studio Community 2022 & Visual Studio Code:** Visual Studio Community 2022 is a full-featured, extensible, free IDE for creating modern applications for Android, iOS, Windows, as well as web applications and cloud services. It's an all-in-one solution that bundles a rich set of tools including an advanced code editor, debugger, profiler, and designer. Visual Studio Code, on the other hand, is a powerful, open-source code editor developed by Microsoft. It has built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes. It also features a rich set of tools and integrations with modern web technologies and frameworks, making it easier to build and test applications.

16. **Microsoft SQL Server Management Studio 18:** Microsoft SQL Server Management Studio (SSMS) is a Windows software tool for managing Microsoft SQL Server. It's a comprehensive environment for managing any SQL infrastructure, from SQL Server to Azure SQL Database. SSMS provides tools to configure, monitor, and administer instances of SQL. It also allows you to deploy, monitor, and upgrade the data-tier components used by your applications, such as databases. With SSMS, you can query, design, and manage your databases and data warehouses, wherever they are - on your local machine, or in the cloud.

17. **Git & GitHub:** Git is a distributed version control system designed to handle everything from small to very large projects with speed and efficiency. It's easy to learn and has a tiny footprint with lightning fast performance. Git tracks changes in a set of files as you code, and allows multiple developers to work on the same project without overwriting each other's changes. GitHub, on the other hand, is a web-based hosting service for version control using Git. It's a platform where software developers can store their projects and network with like minded people. GitHub provides access control and several collaboration features such as bug tracking, feature requests, task management, continuous integration and more for every project.

18. **Seq:** Seq is a self-hosted server for structured log search, analysis, and alerting. It can ingest huge amounts of structured log data, and then display and analyze it in a clean, efficient web-based dashboard. It can handle

complex queries, and its intelligent alerts can catch problems before they spread. It's a crucial tool for developers seeking insights from their log data.

19. **Class Libraries (Data, Repository, Business layers):** Class libraries in .NET are reusable collections of classes, interfaces, and value types that are integrated into applications to provide some kind of functionality. They provide a way to encapsulate and distribute your code across different parts of your application. By separating the data, repository, and business layers into different class libraries, your code becomes more organized, more manageable, and more testable. This architectural approach promotes the Single Responsibility Principle and Separation of Concerns, which makes your codebase much easier to understand and maintain.

By using these various tools and technologies, I ensured the successful implementation and management of the project, and adopted best practices in both front-end and back-end development.

3.5.3 Logging Reasons :

1. **Error Detection & Diagnostics:** Logs help in identifying and debugging software errors, reducing the time and effort required to resolve issues.
2. **Performance Monitoring:** Through logging, developers can measure and optimize the performance of various processes within the application.
3. **Auditing and Compliance:** Logs act as an audit trail for regulatory compliance, recording important activities within the software application.
4. **Understanding User Behavior:** User activities captured in logs provide insights into user behavior and interaction with the application, aiding in feature improvement and user experience optimization.
5. **Security:** Logs can detect suspicious activities, acting as a security measure to identify potential breaches.
6. **Predictive Analysis and Forensics:** Logs can reveal trends and anomalies, assisting in predictive analysis and post-incident investigations.
7. **System Health Monitoring:** Logs provide vital information about the system's state, assisting in assessing overall health and identifying potential future issues.

Chapter Four
Results and Discussion

Results and Discussion

In this chapter, we delve into the key outcomes of our web application, combining tangible results with insightful discussion. Here, you'll find a comprehensive analysis of each page, enriched by relevant screenshots and detailed descriptions.

4.1 Sign in

In this page users can insert their email address and password to use application, if the user enter correct email and password he/she will have access to our system if not message will be shown , if user don't have and account he can request one by press on Request Account and send email to system owner ,see **Figure 11** and **Figure 12**for more information.

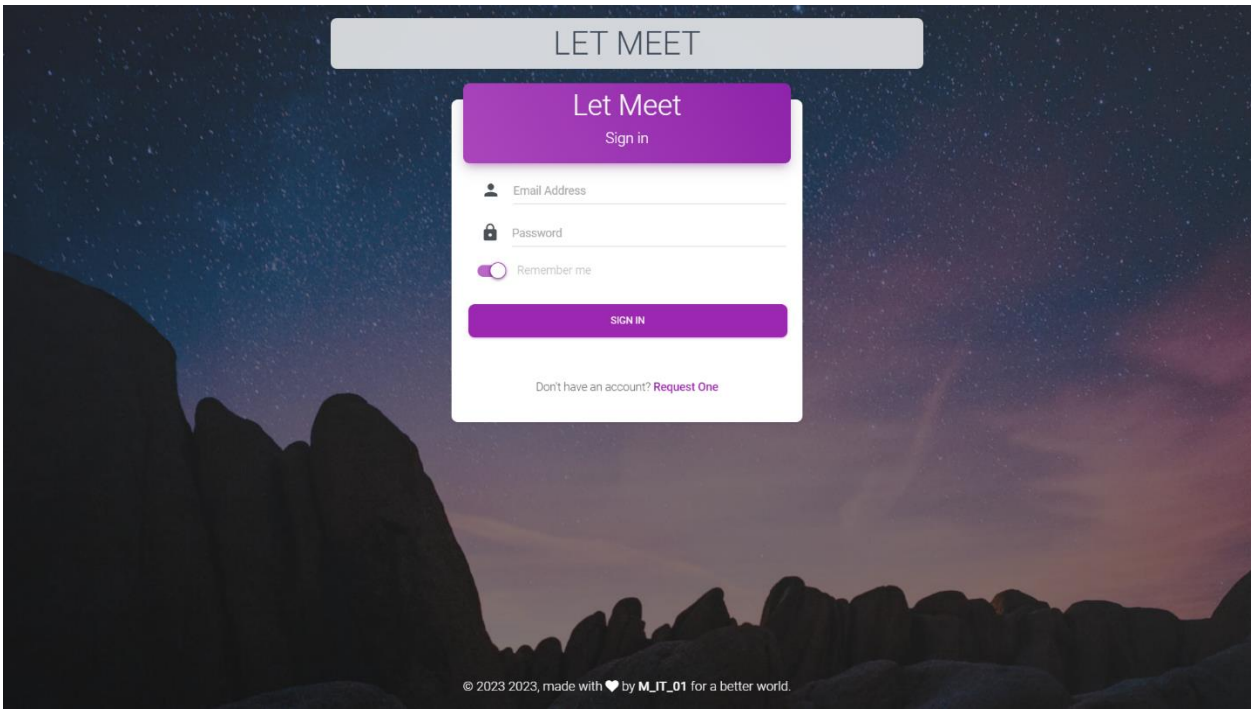


Figure 11 Sign in Page

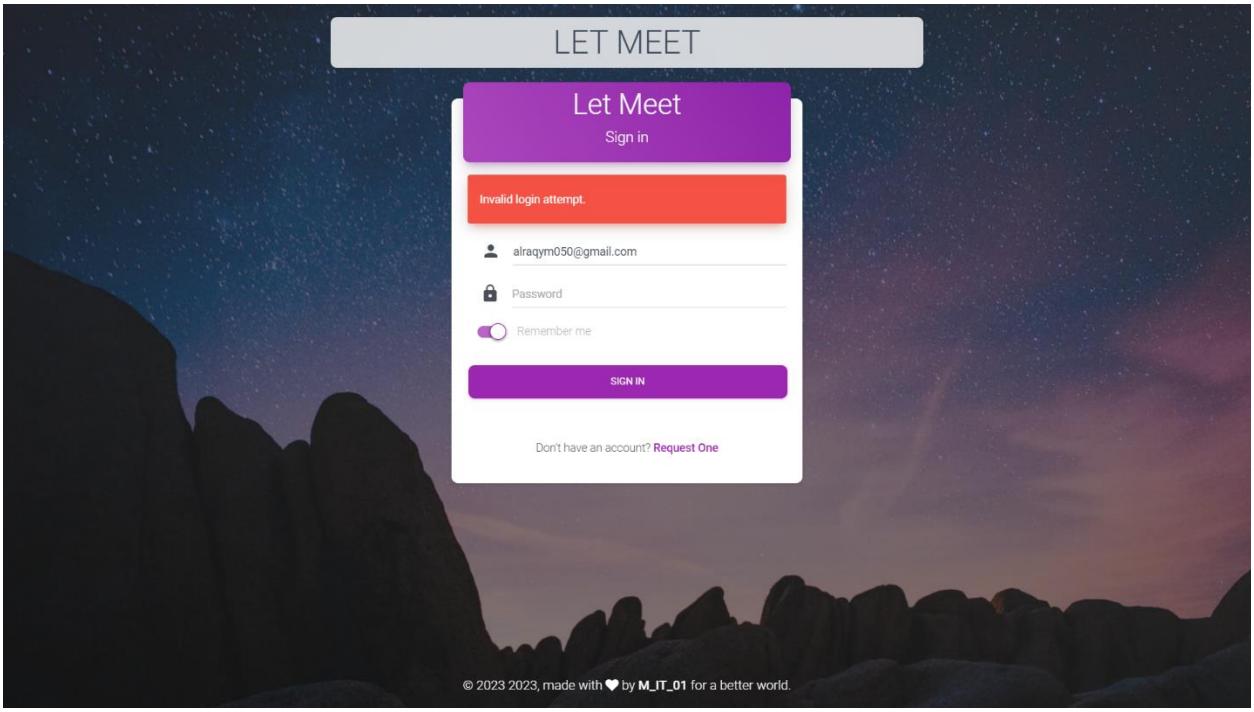


Figure 12 Sign in Page (invalid)

4.2 Manage Accounts

If user is an admin, he can manage all users and update their information's like update their account email, rest password, name etc.., also admin can remove entire user from system, admin can add student to supervisors to supervise , admin can see only specific users per page he can move through pages by press arrows in center of page, see **Figure 13** for more information.

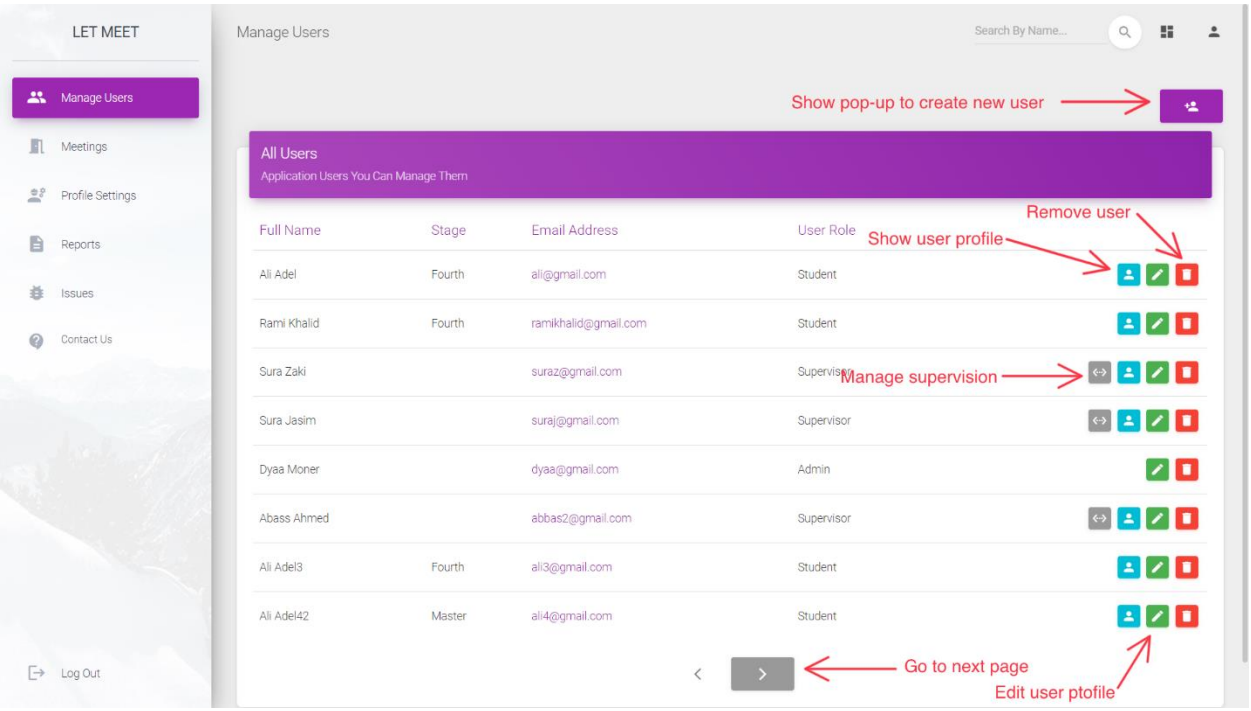


Figure 13 Manage Users (admin page)

4,3 Create Account

If user is an admin, he can create new user, by press create user icon and pop-up will be shown to specify information's about user like full name, email address, user stage and role, if email is already in use or any error happen, he will get message to notify him with it, see **Figure 14** for more information's.

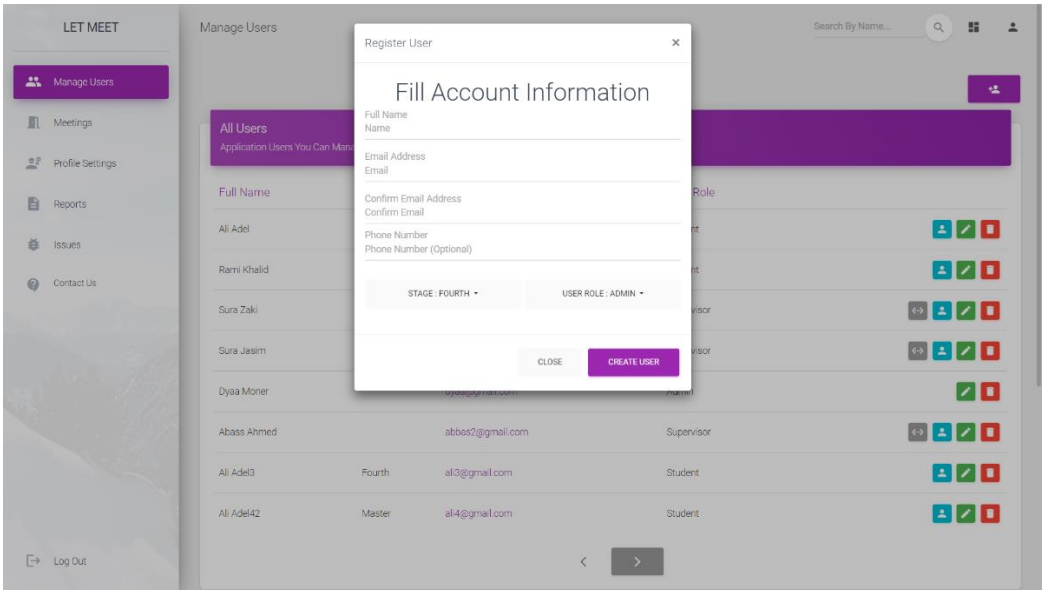


Figure 14 Create New User (admin page)

4.4 Show and Edit Profile

User can see and update only his account , profile image and his password by specify the old one but he can't change his email and role , admin can edit all accounts , roles and rest passwords , admins don't have profile because profile has info about user tasks and meetings , see **Figure 15** and **Figure 16** for more information.

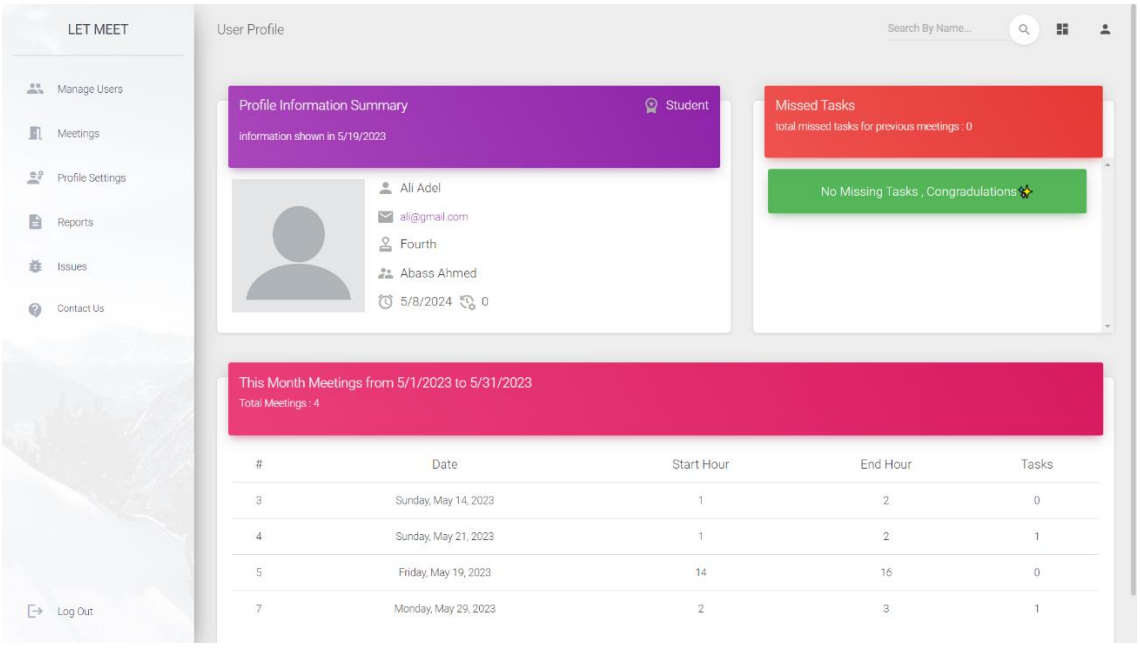


Figure 15 User Profile

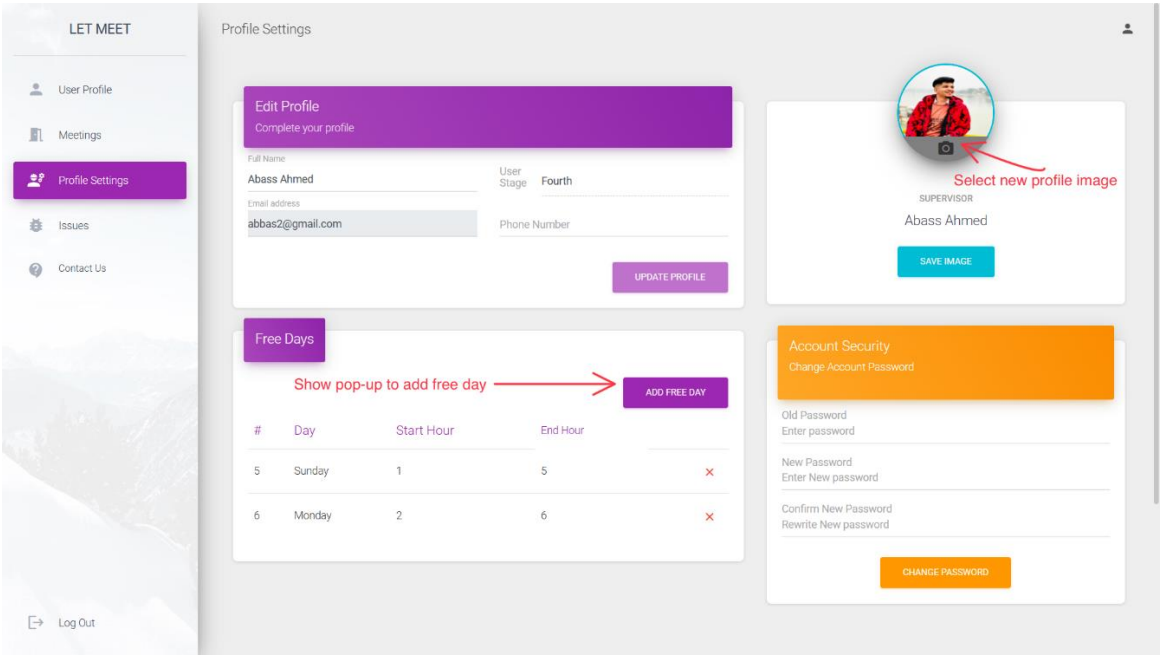


Figure 16 Profile Settings

4.5 Manage Free Days

User can specify and remove his free days by select day , start free hour and end free hour using 24 day hours format , see **Figure 17** and **Figure 18** for more information.

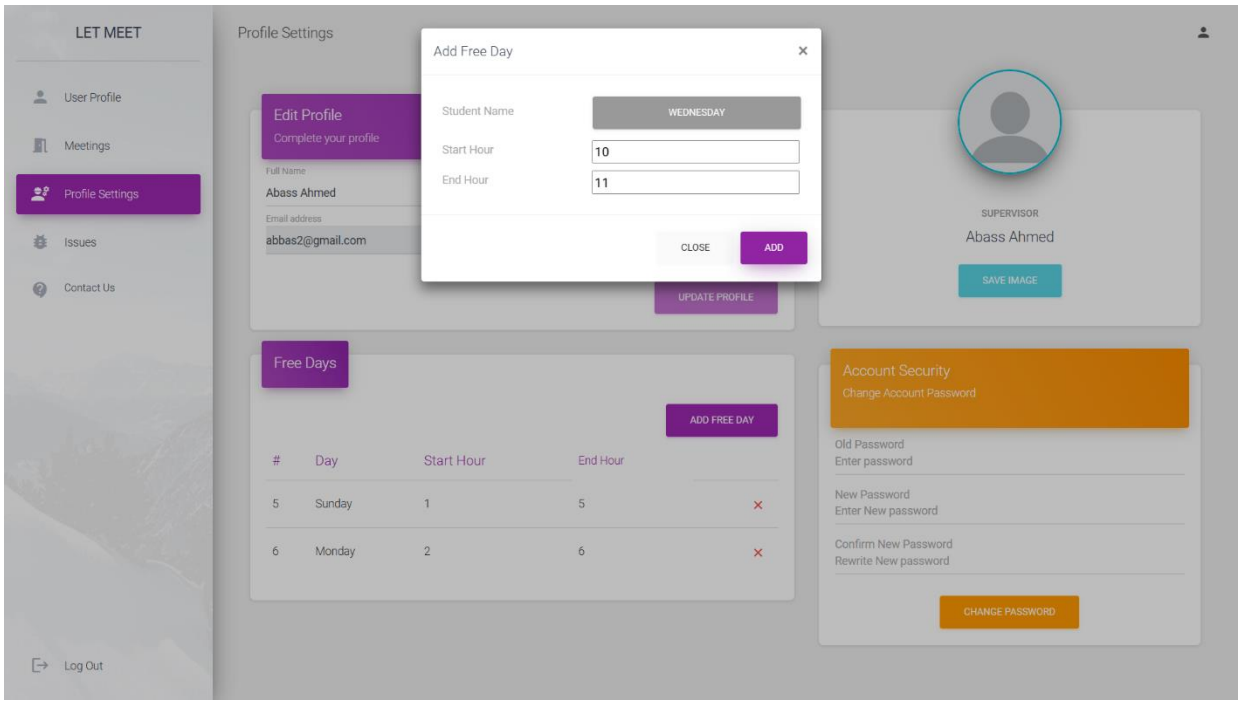


Figure 17 Add Free Day

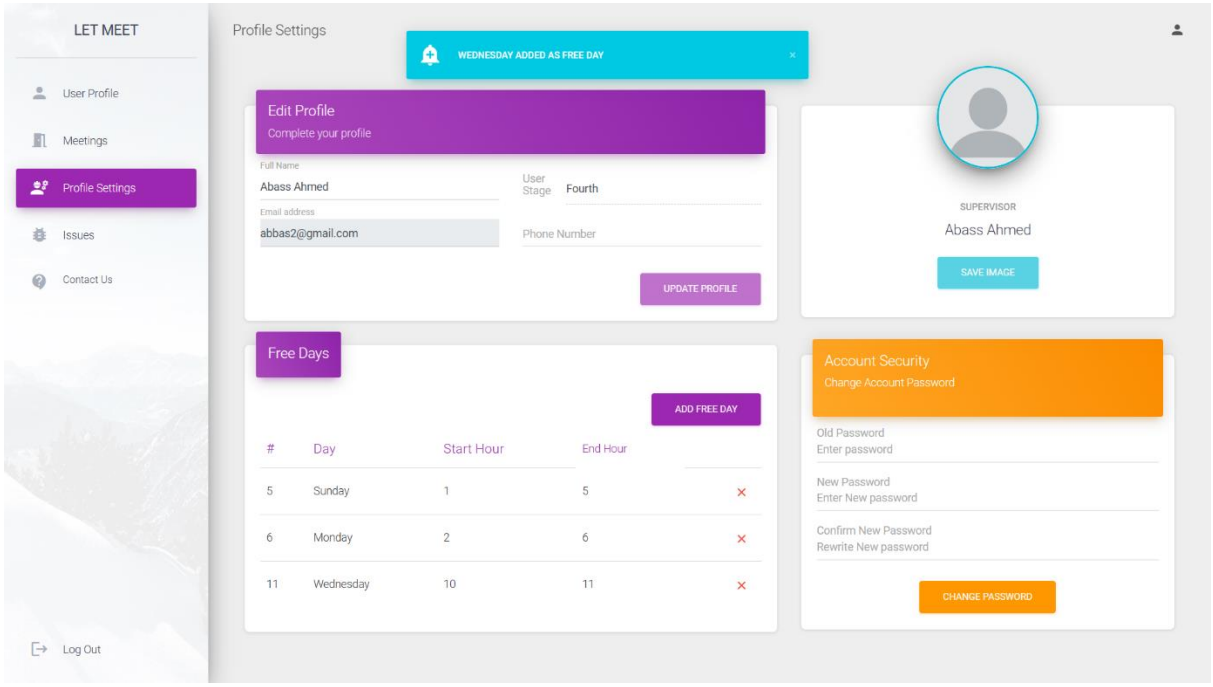


Figure 18 Update Free Day

4.6 Manage Supervision

Admin only can add student to supervisor to supervisor and specify start and end supervision date , admin can extend supervision end date with 6 months (can be changed from appsetting file) , manage supervision can be accessed from manage users page by click on supervision icon corresponding to supervisor see **Figure 19** and **Figure 20** for more information.

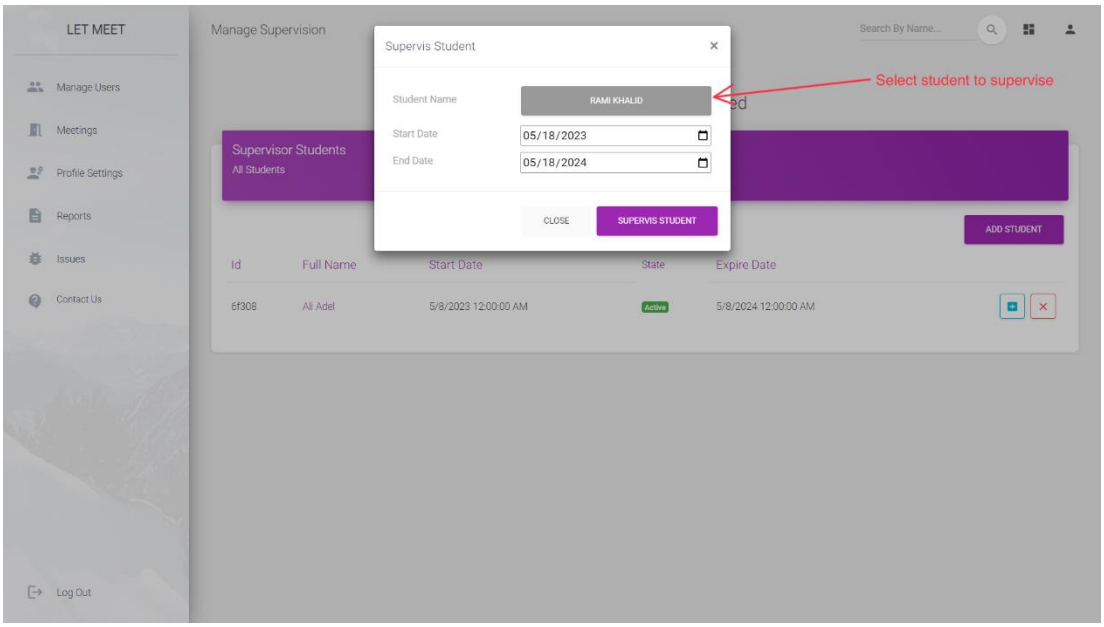


Figure 19 Supervise New Student

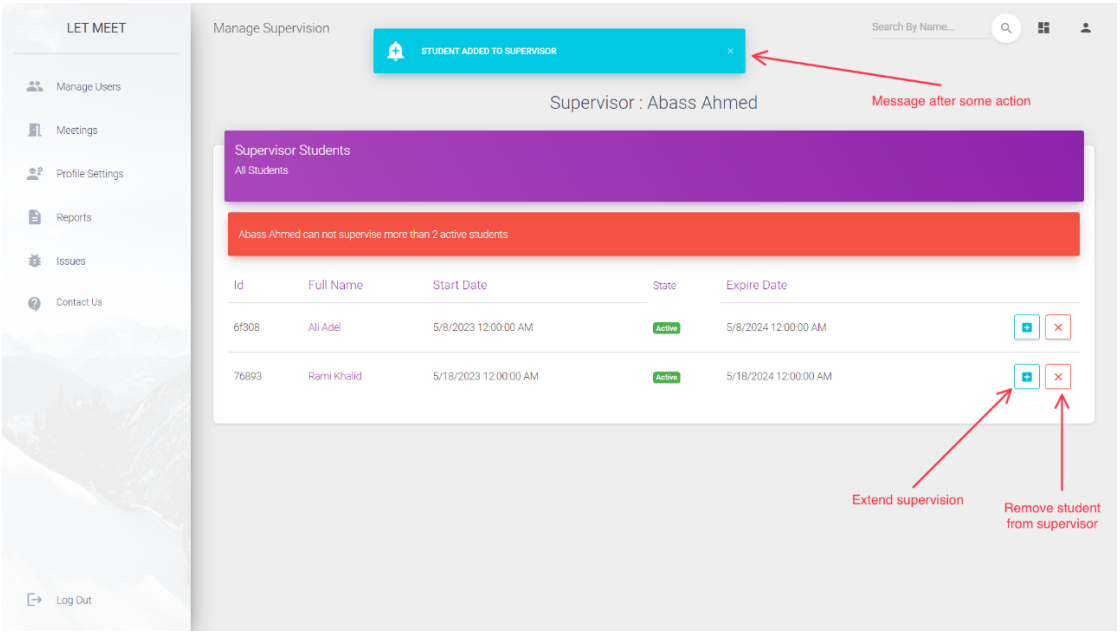


Figure 20 Manage Supervisions

4.7 Create Meeting

If user is a supervisor he can create meeting for his student by press on add meeting icon in meetings page then select student name after press go button user will redirect to create meeting page see **Figure 21** and **Figure 22** for more information.

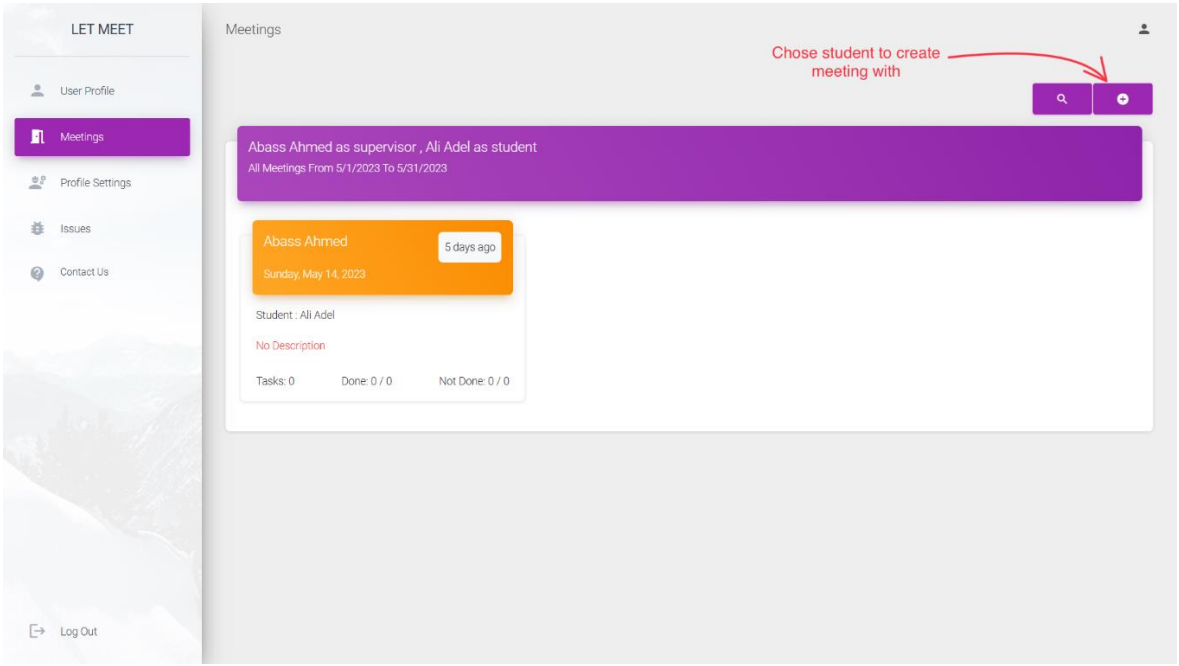


Figure 21 Chose Student to Create Meeting

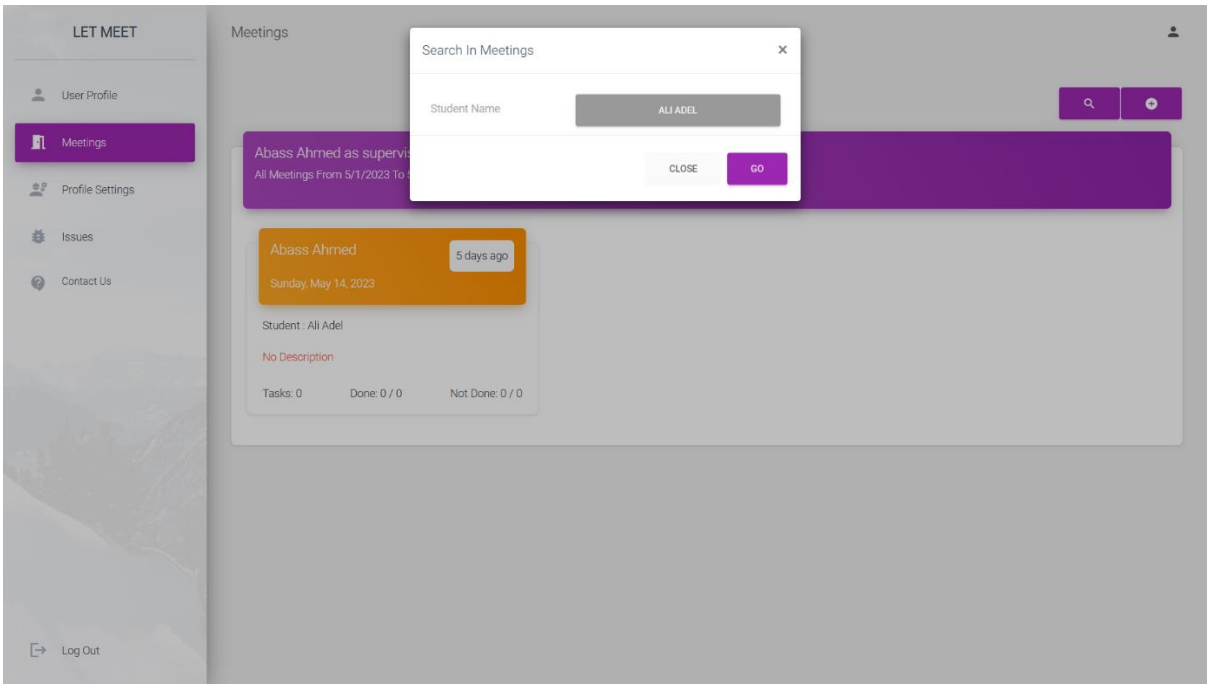


Figure 22 Select Student Name for Meeting

After select student to create meeting inside this page supervisor can specify meeting details, day based on mutual free days and hours between him and his student, meeting description and meeting tasks if exist, if there is a meeting at same date or user select date not matches one of free days message will be shown, see **Figure 23** , **Figure 24** , **Figure 25** ,and **Figure 26** for more information.

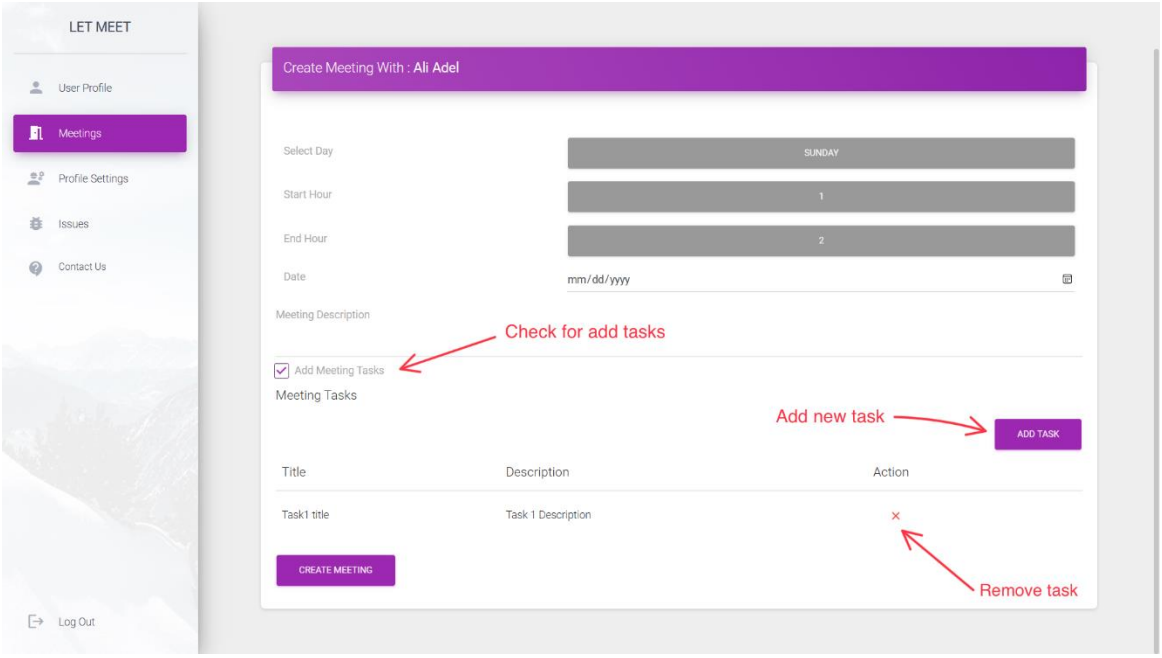


Figure 23 Create New Meeting (supervisor page)

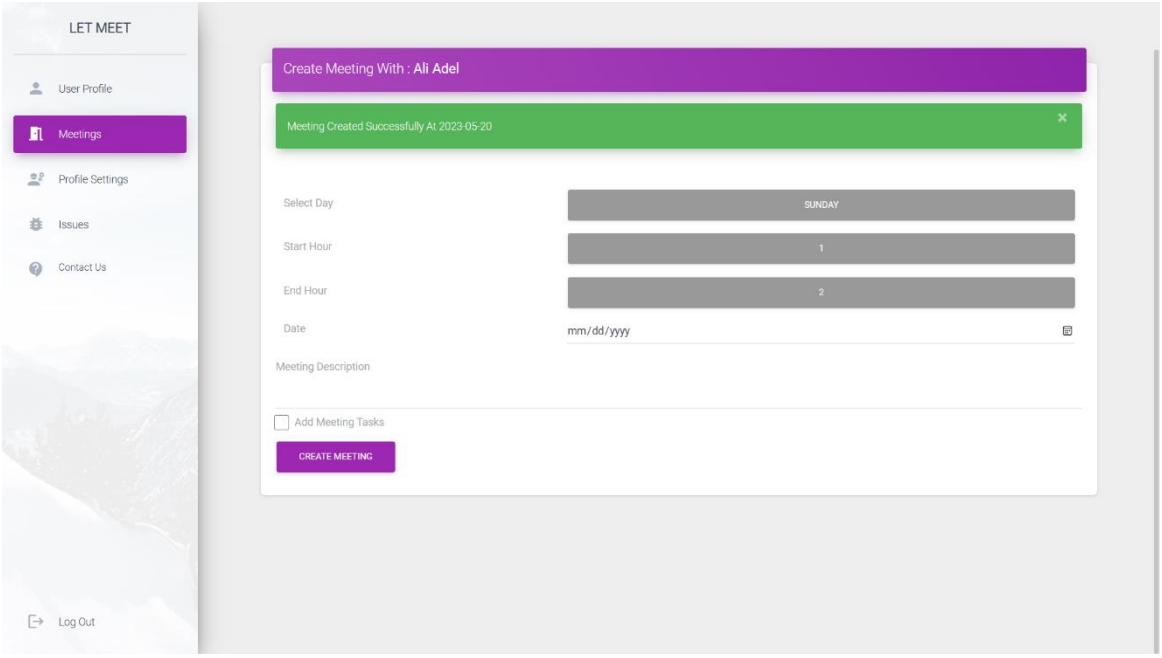


Figure 24 Create New Meeting Success (supervisor page)

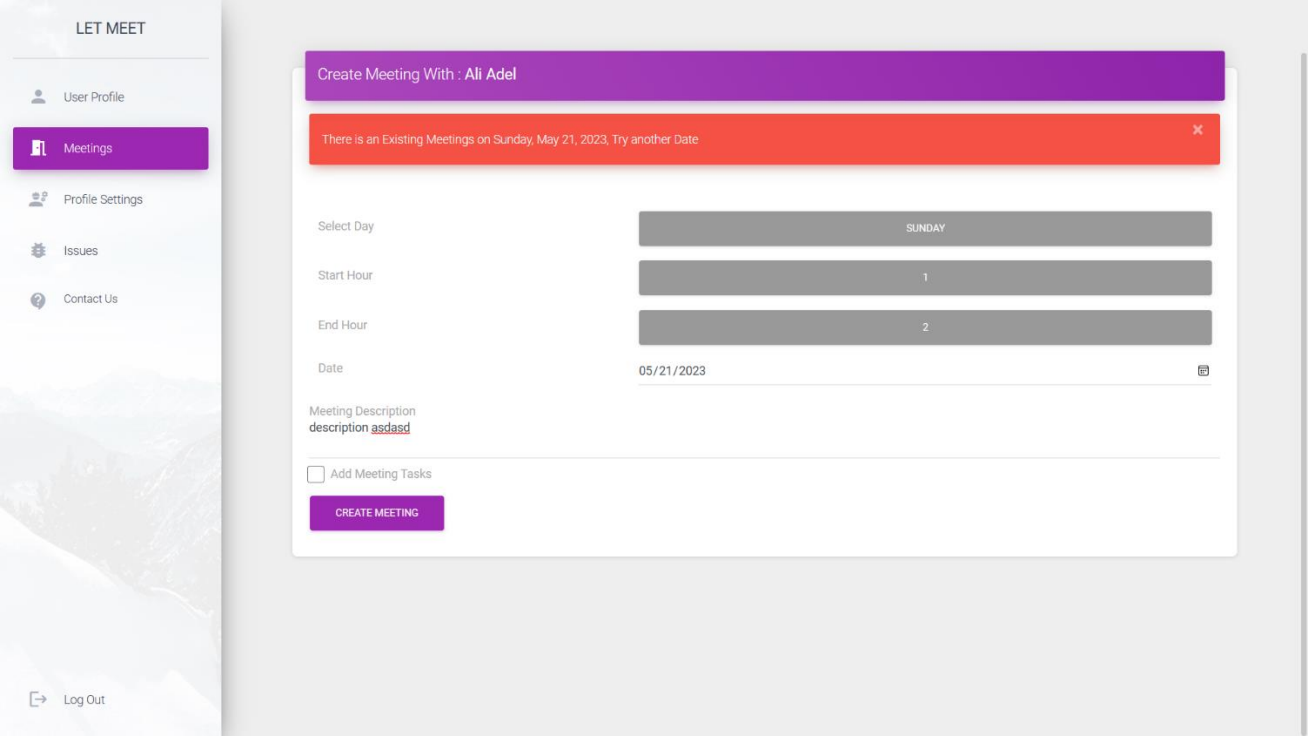


Figure 25 Create New Meeting Fail by Same Date (supervisor page)

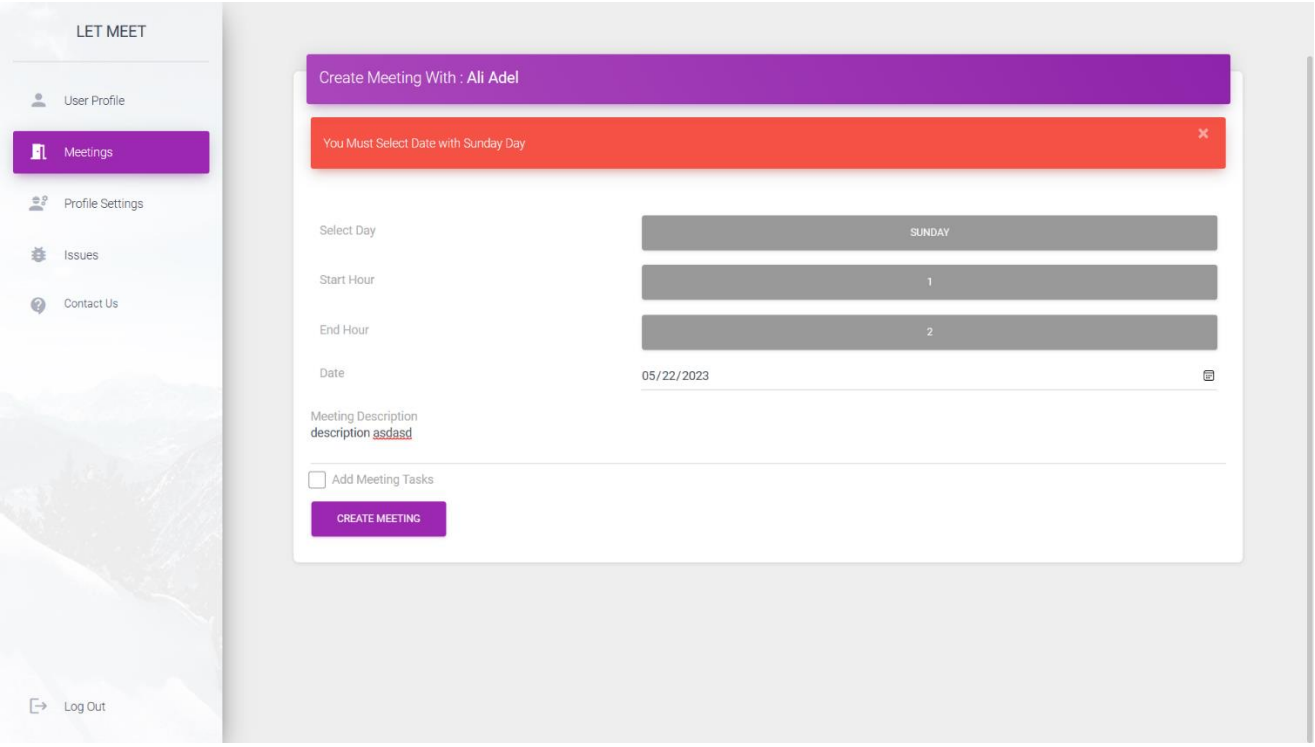


Figure 26 Create New Meeting Fail by Wrong Date (supervisor page)

4.8 Show and Query Meeting

If user is a supervisor he can select student, start and end date for see meetings , or if he was student he only specify start and end date, if admin he can select supervision and see meetings , meeting will be shown with different colors depends on meeting start date if date is end it will be yellow if sky blue if meeting time not coming , and green if meeting time is now, supervisor and admin can remove meeting before meeting time come by padding hours like 1 hour (can be changed from appsettings file) ,see **Figure 27** , **Figure 28** , **Figure 29** and **Figure 30** for more information's .

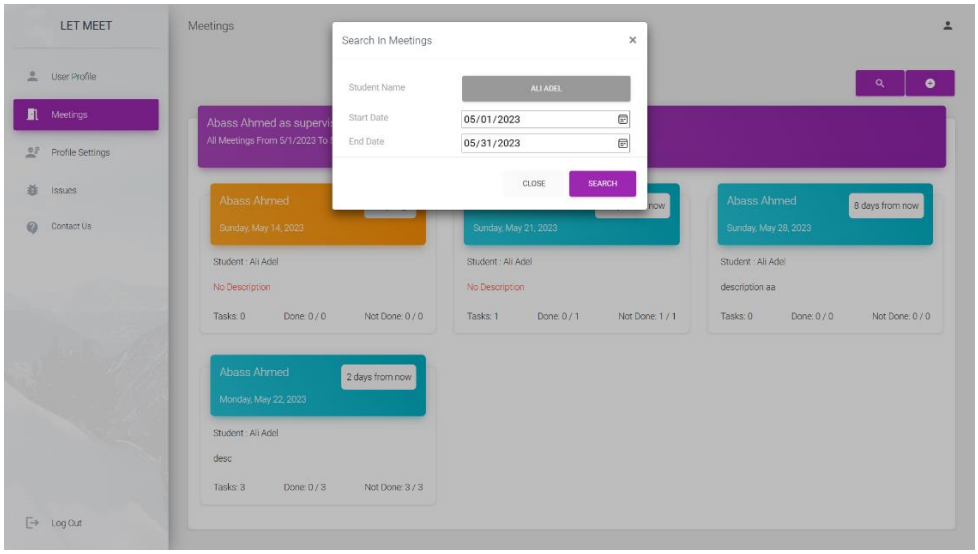


Figure 27 Select Student to Query meetings

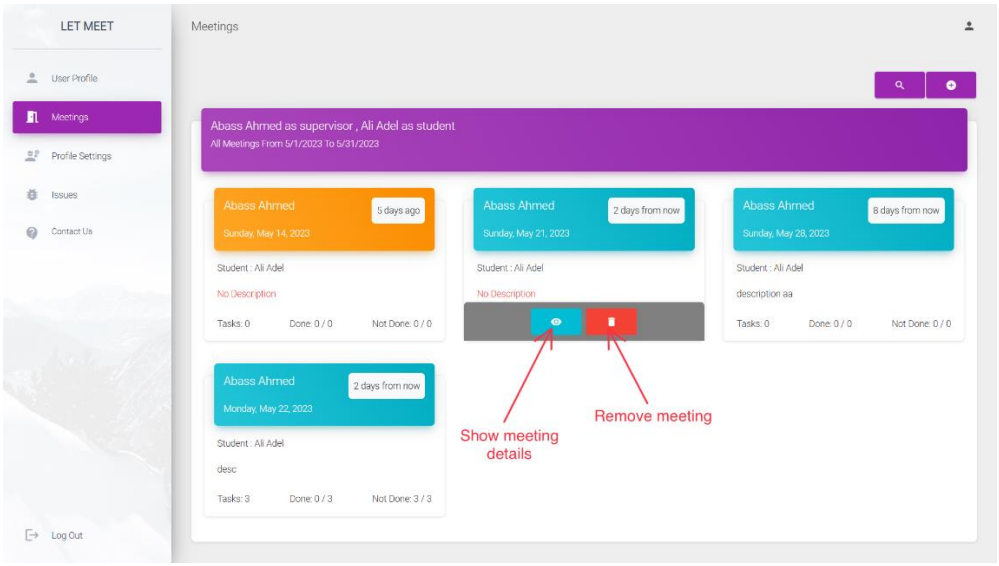


Figure 28 Meeting Actions Preview and Remove

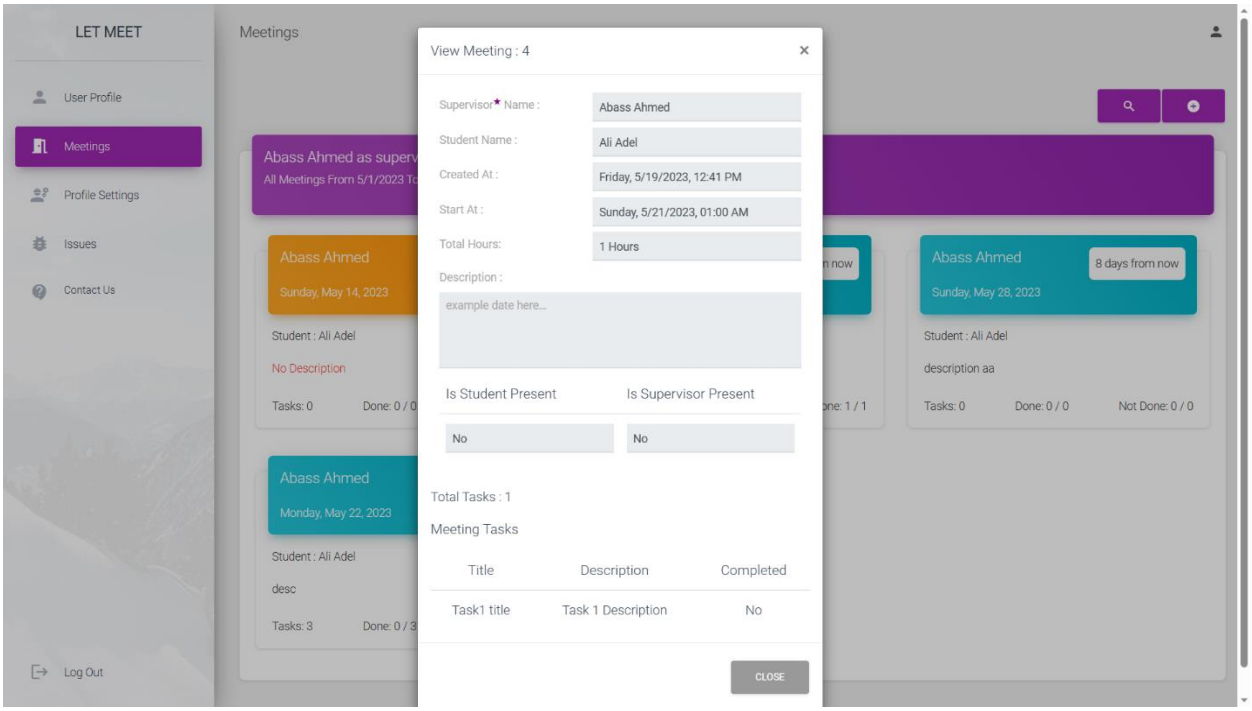


Figure 29 Meeting Details

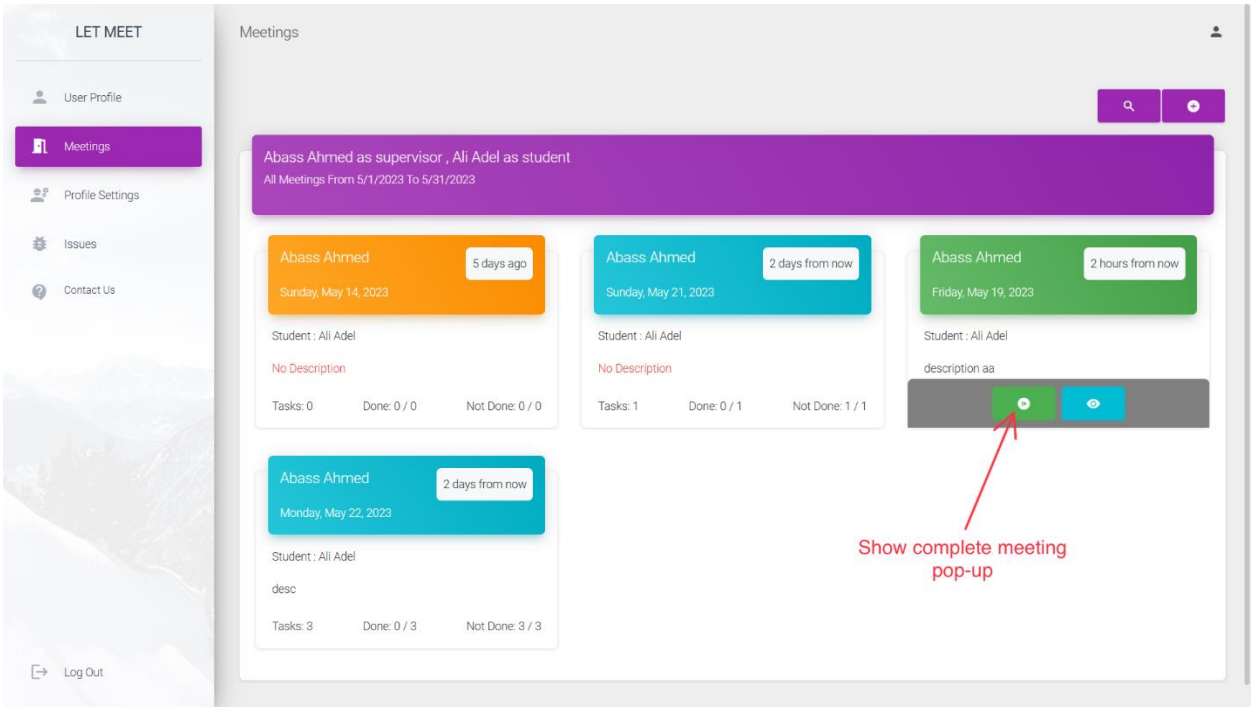


Figure 30 Complete Action

Supervisor can specify student presents and done tasks, see **Figure 31** for more information.

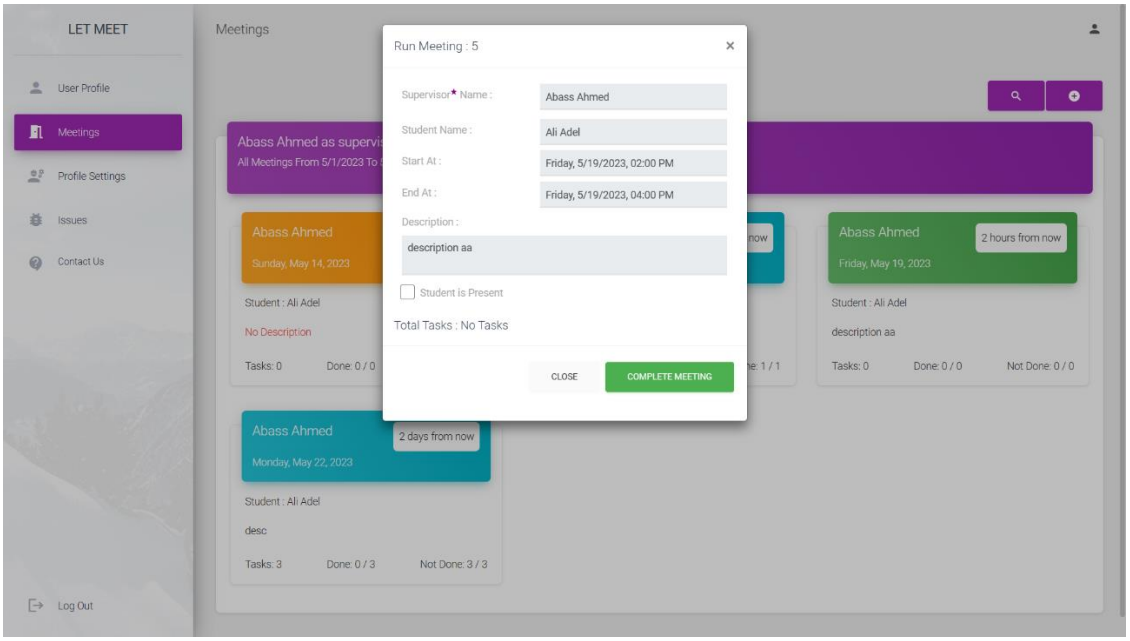


Figure 31 Complete Meeting Window

Student can register his presents if supervisor didn't specify meeting info , see **Figure 32** for more information's .

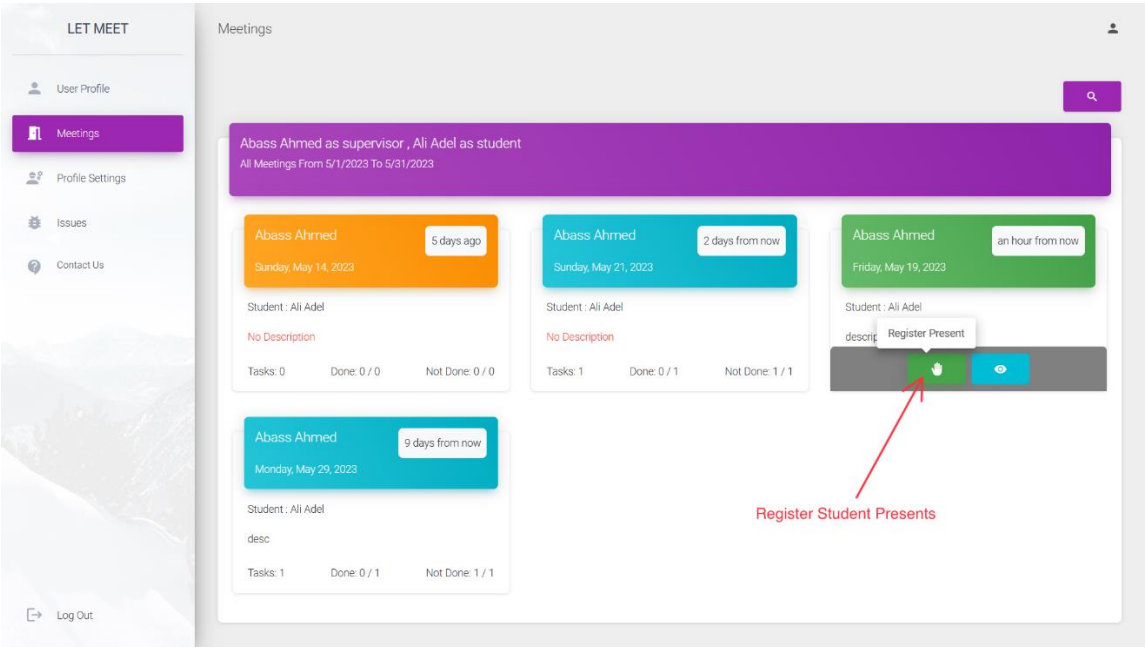


Figure 32 Register Student Presents

4.9 Reports

Admin can generate report about serval things ,report about most absence student and supervisor ,report about supervisor who achieved max number of students ,report about idle supervisors who have no student currently and there is advanced report about students and supervisor for more information about their activities and presents ,admin can print any type of report by pressing print icon above report he want, see **Figure 33** , **Figure 34** and **Figure 35** for more information's.

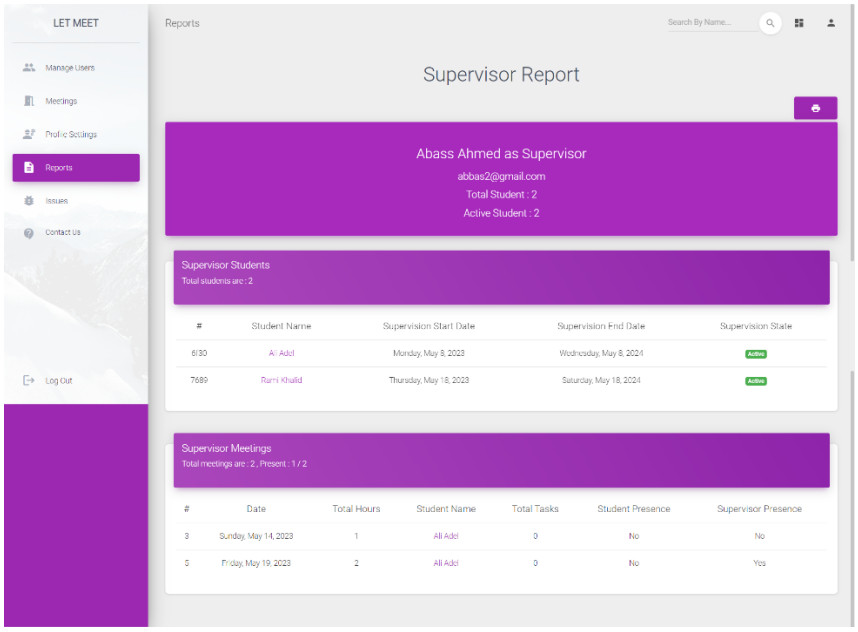


Figure 33 Supervisor Advanced Report

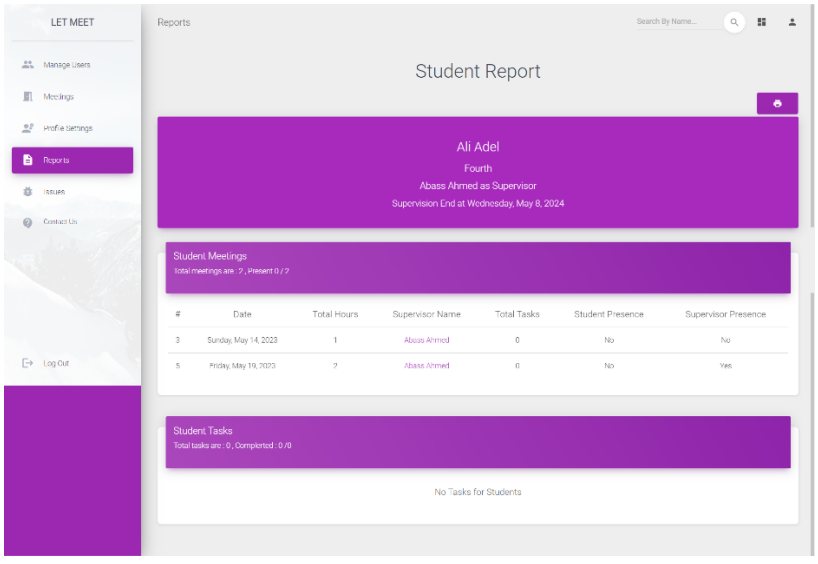


Figure 34 Student Advanced Report

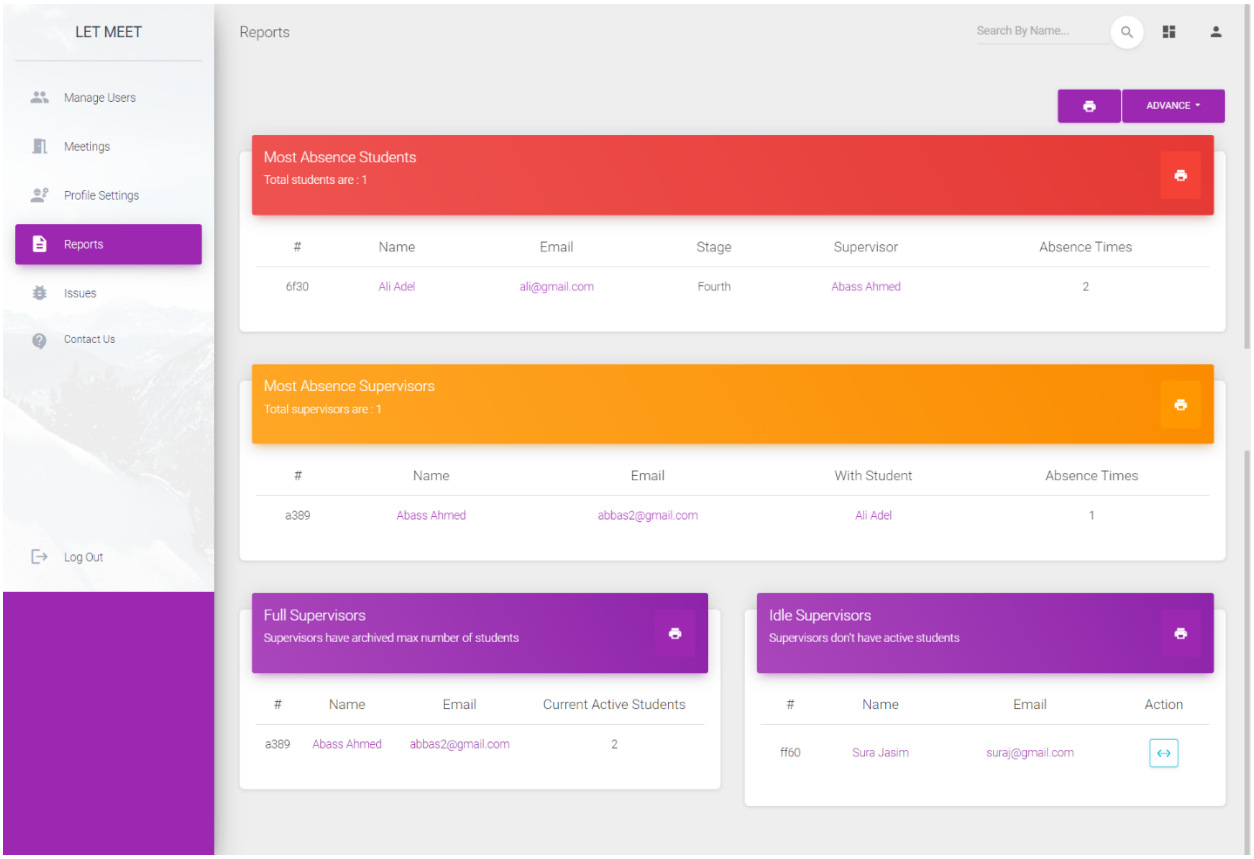


Figure 35 Generic Report

4.10 Seq Logging Monitor

Server Admin can see and query logs to trace and analyses all behavior , check for errors and for security ,using request id and user id can follow all users interaction, see **Figure 36** for more information.

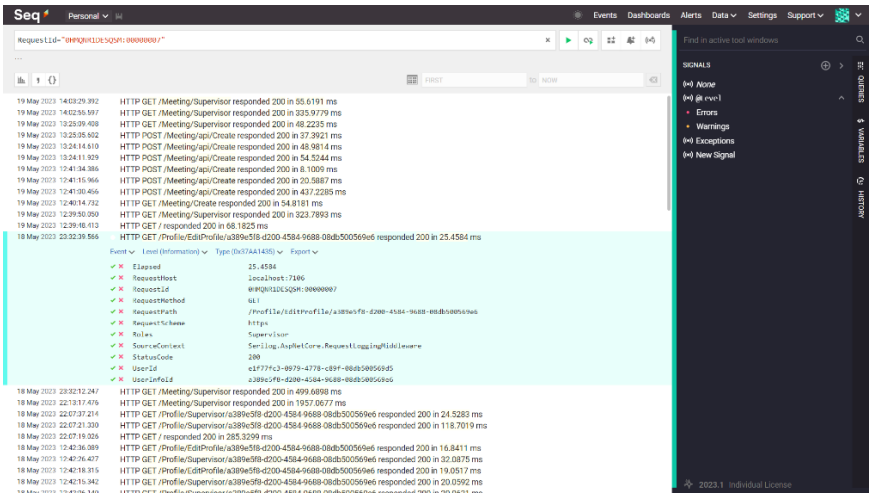


Figure 36 Seq Logging Monitor

Chapter Five
Conclusion and Future Work

Conclusion and Future Work

5.1 Conclusion

In conclusion, we have successfully implemented a comprehensive system using ASP.NET Core Web App, .NET 6, ASP.NET Core Identity, and Entity Framework Core, underpinned by SQL Server databases. This multifunctional system efficiently handles user management, meeting coordination, and task allocation, serving as a powerful tool for administrators, supervisors, and students alike.

The system is enriched by several robust features, from user sign-in and account creation to sophisticated role allocation and account management. Its capabilities extend to coordinating meetings, aligning schedules, and assigning tasks within meetings, showcasing a thorough understanding of practical needs in an educational setting.

A standout feature of our system is the sophisticated reporting functionality. This provides invaluable insights into user activities and system metrics, underpinning effective decision-making and future improvements.

The integration of advanced technologies like Serilog has also played a pivotal role in enhancing system efficiency, enabling tracking of user activities and system performance. This, along with user-friendly design elements such as front-end compression techniques for image upload and loading indicators, significantly enhances the overall user experience.

In essence, our system stands as a landmark contribution to the field of software engineering, fulfilling demanding requirements while offering sophisticated reporting and performance monitoring features. It serves as a testament to effective application development and offers a promising platform for future research and development.

5.2 Future Work

Here potential avenues for future work for our software

1. **Integration with External Platforms:** Future work could explore integrating the system with external platforms such as online calendars or educational tools. This would allow users to synchronize their schedules across different platforms and could make the system even more convenient and effective.
2. **Artificial Intelligence Enhancements:** Implementing artificial intelligence (AI) algorithms for schedule optimization and task management could be an exciting development. AI could potentially recommend optimal meeting times based on users' schedules or predict task completion times to better plan future meetings.
3. **Mobile Application Development:** As usage of mobile devices continues to grow, developing a mobile version of the system could increase accessibility for users. This would allow users to manage their profiles, view meeting details, and receive notifications on the go.
4. **Advanced Reporting and Analytics:** While the current system includes comprehensive reporting features, future work could involve developing more advanced analytics capabilities. This could include more detailed performance metrics, user behavior analysis, or predictive analytics to forecast future trends.
5. **User Feedback and Continuous Improvement:** Finally, ongoing user feedback should be collected to continuously improve the system. Users are the best source of information for understanding which features are most useful, what can be improved, and what new capabilities might be needed in the future. This could be implemented as a feedback feature within the system itself

References

[1] Microsoft, "Introduction to Identity on ASP.NET Core," 2022. [Online]. Available: https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-6.0&tabs=visual-studio. [Accessed 2023].

[2] Microsoft, "What's new in .NET 6," 2022. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/core/whats-new/dotnet-6. [Accessed 2023].

[3] K. Brush, "TechTarget," 2020. [Online]. Available: https://www.techtarget.com. [Accessed 10 4 2023].

[4] Nishadha, "Entity Relationship Diagram (ERD)," 11 12 2022. [Online]. Available: https://creately.com/. [Accessed 2023].

[5] K. Kaseb, "Medium," 27 May 2022. [Online]. Available: https://medium.com/kayvan-kaseb/. [Accessed May 2023].

[6] TheCodeReaper, "TheCodeReaper," 22 August 2020. [Online]. Available: https://thecodereaper.com/2020/08/22/. [Accessed May 2023].

[7] S. Shirhatti, "ASP.NET Core updates in .NET 6," 2021. [Online]. Available: https://devblogs.microsoft.com/dotnet/asp-net-core-updates-in-net-6-preview-1/.

[8] R. T. D. Roth, Pro ASP.NET Core 3: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages, Apress, 2020.

[9] M. Collins, Dependency Injection in .NET Core 3.1, Independent, 2020.

[10] J. Smith, Entity Framework Core in Action, Manning Publications, 2020.

[11] Microsoft, "Logging in .NET Core and ASP.NET Core," 2022. [Online]. Available: https://learn.microsoft.com/en-us/aspnet/core/fundamentals/logging/?view=aspnetcore-6.. [Accessed 2023].

[12] S. Lambert, Pro Bootstrap 4, Apress, 2019.

[13] B. Dayley, jQuery and JavaScript Phrasebook, Addison-Wesley Professional, 2021.

[14] D. Fowler, ASP.NET Core in Action, Manning Publications, 2021.

[15] S. Allen, "Mastering Identity and Access Management with Microsoft Azure," Packt Publishing, 2016.

[16] B. Morris, Designing Evolvable Web APIs with ASP.NET: Harnessing the Power of the Web, O'Reilly Media, 2014.

[17] J. Locke, Learn ASP.NET Core 3: Develop modern web applications with ASP.NET Core 3, Razor Pages, Bootstrap 4, and Azure, Apress, 2019.

[18] B. Conroy, "Performance improvements in ASP.NET Core 6," 2022. [Online]. Available: https://devblogs.microsoft.com/dotnet/performance-improvements-in-aspnet-core-6/. [Accessed 2023].

[19] Microsoft, "Tutorial: Get started with EF Core in an ASP.NET MVC web app," 2021. [Online]. Available: https://learn.microsoft.com/en-us/aspnet/core/data/ef-mvc/intro?view=aspnetcore-7.0.

[20] D. Makovetskiy, Responsive Web Design with HTML5 and CSS: Develop future-proof responsive websites using the latest HTML5 and CSS techniques, 3rd Edition, Packt Publishing, 2020.