A logo for a company

Description automatically generated with low confidence

**A**

**PROJECT REPORT**

**On**

**“EdifyLabs Business Tool”**

**Submitted By**

**Tanmay Kisan Barvi (5053)**

**Under the guidance of**

**Prof. Milind Deshkar**

**SUBMITTED TO**



**Savitribai Phule Pune University**

**As a partial fulfilment for the award of the degree of**

**MASTER OF COMPUTER APPLICATIONS**

**At**

**ASM’S Institute of Business Management and Research,**

**MCA Institute, Chinchwad, Pune- 411019**

**A.Y 2022-2023**

ACKNOWLEDGEMENT

It is a pleasure to acknowledge the assistance of several people and institutions in this effort. This project has turned me into a debtor. First and foremost, I feel indebted to my guide **Prof. Milind Deshkar** and all the faculty members **of Department of MCA, ASM’s IBMR, MCA** Institute Chinchwad for their valuable guidance, continuous support and advice and constant encouragement throughout my project work.

I would like to extend my gratitude to **honorable Dr. V. P. Pawar Sir, Director, ASM’s IBMR, MCA** Institute, Chinchwad for being a constant source of inspiration.

Finally, I would like to extend my thanks to all those who have contributed, directly or indirectly, to make this project successful.

Tanmay Kisan Barvi (5053)

|  |  |  |
| --- | --- | --- |
| **Chapter No** | **Details** | **Page No.** |
| **1** | **Introduction** | 1 |
| 1.1 | Company Profile | 2 |
| 1.2 | Abstract | 4 |
| 1.3 | Existing System and Need for System | 5 |
| 1.4 | Scope of System | 7 |
| 1.5 | Operating Environment - Hardware and Software | 8 |
| 1.6 | Brief Description of Technology Used | 9 |
| **2** | **Proposed System** | 13 |
| 2.1 | Study of Similar Systems | 14 |
| 2.2 | Feasibility Study | 16 |
| 2.3 | Objectives of Proposed System | 17 |
| 2.4 | User Requirements | 19 |
| **3** | **Analysis and Design** | 21 |
| 3.1 | System Requirements (Functional and Non-Functional requirements) | 22 |
| 3.2 | Data Flow Diagram (DFD) | 28 |
| 3.3 | Entity Relationship Diagram (ERD) | 34 |
| 3.4 | Table Structure | 35 |
| 3.5 | Use Case Diagrams | 40 |
| 3.6 | Activity Diagram | 41 |
| 3.7 | Sequence Diagram | 42 |
| 3.8 | Deployment Diagram | 43 |
| 3.9 | Component Diagram | 44 |
| 3.10 | Sample Input and Output Screens | 45 |
| 3.11 | Report | 49 |
| **4** | **Coding** | 51 |
| 4.1 | Algorithms | 52 |
| 4.2 | Code snippets | 54 |
| **5** | **Testing** | 63 |
| 5.1 | Test Strategy | 64 |
| 5.2 | Test Case / Test Script | 66 |
| 5.3 | Defect report / Test Log | 72 |
| **6** | **Limitations of Proposed System** | 75 |
| **7** | **Proposed Enhancements** | 77 |
| **8** | **Conclusion** | 79 |
| **9** | **Bibliography** | 81 |
| **10** | **User manual** | 83 |

**CHAPTER 1**

**Introduction**

**1.1 Client’s Profile**

**EdifyLabs**

**COMPANY NAME :** Edify Labs

A picture containing text, font, logo, graphics

Description automatically generated

**ABOUT COMPANY :**

Edify Labs in Thane, Mumbai is one of the leading businesses in the Career Counseling Canters. Also known for Career Counseling Canters, Aptitude Test Canters, Life Coaching Services, Corporate Training Services, Motivational Activity Organizers, Personal Coaching and much more.

**FOUNDER:**

Founded by Nitesh Gawade, is an Multiple Award Winning Strategic Learning Partner to India’s Top Corporates and Best Selling Author. Nitesh has trained more than 2,90,000 participants across India in the last 10 years. Nitesh is a coach to authors , entrepreneurs and potential CXOs.

**SERVICES OFFERED (COURSES) :**

Edify Labs offers personalized coaching programs such as "Make It Happen - Personal Coaching," certification courses like "Licensed Purpose Coach," and a coaching program for aspiring teenage entrepreneurs called the "Teenpreneur Fellowship Personal Coaching Program." They also provide courses in career guidance, soft skills training, leadership development, personality development, interview preparation, team building, and motivational workshops. The courses offered by Edify Labs are priced in the range of 1 lakh to 7 lakhs, depending on the program and its duration.

**DETAILS:**

Website : www.livingtheunlived.com

Email : nitesh.gawade@edifylabs.com Customer Care : 80 80 024 024

Address : EDIFY LABS , Fourth Floor, WEWORK Zenia, Hiranandani Estate, Thane West.

**1.2 Abstract**

The Edify Labs HR Software System is a comprehensive and user-friendly solution designed to streamline HR processes and enhance efficiency in growing organizations. With advancements in technology and the increasing complexity of HR tasks, there is a need for a personalized and data-driven tool that can simplify and automate HR activities. Edify Labs aims to fill this gap by providing HR professionals with an easy-to-use software system that offers a tailored and comprehensive HR experience.

The software system is built on the Python Django web framework, leveraging its robust features and flexibility. It offers various modules and functionalities to meet the diverse needs of HR professionals and their clients. The system includes an HR dashboard for managing client information and performing CRUD operations. Scheduling capabilities enable users to schedule meetings and appointments, ensuring effective time management. The filtration feature allows for easy searching and management of data across different sections.

Data visualization tools provide graphical representations of HR-related information, enabling users to track and analyze data effectively. The report generation feature enables HR professionals to generate customized reports for record-keeping and analysis. Invoicing functionality facilitates the creation of payment invoices for clients, supporting various payment methods.

The system ensures secure authentication and data protection measures, allowing authorized users to access the system and ensuring the confidentiality of sensitive HR information. Integration with databases and external networks ensures accurate and up-to-date information availability for HR professionals.

**1.3 Existing System and Need for System**

**Existing System:**

The existing HR software systems often lack personalization and tailored experiences for individual clients. This means that HR professionals may struggle to effectively address the specific needs and requirements of each client they work with. The lack of customization can limit their ability to provide a high level of service and support to clients.

Furthermore, some HR software systems can be complex and difficult to navigate. This complexity may result in a steep learning curve for HR professionals, leading to reduced efficiency and productivity. The user interface may not be intuitive, making it challenging for HR professionals to quickly perform tasks and access the information they need.

Another limitation of many existing HR software systems is the lack of in-depth data analysis capabilities. While these systems may store and manage data related to clients' HR activities, they may not provide robust analytical tools to gain insights from this data. Without access to comprehensive data analysis features, HR professionals may struggle to identify trends, patterns, and opportunities for improvement within their clients' HR processes.

To overcome these limitations, there is a need for a more personalized, user-friendly, and data-driven HR software solution. A modern HR software system should allow for customization and flexibility, enabling HR professionals to tailor the system to the unique needs of each client. This could involve customizing workflows, reports, and user interfaces to align with specific client requirements.

**Need for System:**

Existing HR software systems have proven to be valuable tools for managing clients data and related activities. However, they often fall short in meeting the evolving needs of HR professionals and fail to provide a comprehensive solution for optimizing HR processes. This creates a compelling need for a new and innovative HR software system, such as Edify Labs, to address the following key challenges:

Lack of Personalization: Existing systems often lack the ability to provide personalized experiences for individual clients. HR professionals require a system that can tailor its functionalities and features to meet the unique needs of clients, enabling them to effectively address their concerns and provide a more engaging HR experience.

Complex and Inefficient User Interfaces: Some HR software systems suffer from complex user interfaces and navigation structures, leading to reduced efficiency and productivity. HR professionals need a user-friendly system with intuitive interfaces that simplify tasks, streamline processes, and enhance overall usability.

Limited Data Analysis Capabilities: Existing systems may lack advanced data analysis features, preventing HR professionals from gaining valuable insights into employee behavior, performance, and engagement. A new HR software system should provide robust data analysis tools and visualizations to empower HR professionals with actionable intelligence for informed decision-making.

Ineffective Process Improvement: Without comprehensive data analysis and reporting capabilities, HR professionals may struggle to identify areas for process improvement and efficiency gains. A new system should enable HR professionals to identify trends, patterns, and bottlenecks within HR processes, facilitating continuous improvement initiatives.

**1.4 Scope of System**

The scope of the Edify Labs Business Tool project encompasses the development of a web-based application that specifically caters to the needs of HR departments. The primary objective is to provide HR professionals with an intuitive and user-friendly solution for efficiently managing client details, course scheduling, and various related activities. The application will feature an HR dashboard that serves as a centralized hub for accessing and organizing client information. HR professionals will be able to add, update, and manage client details effortlessly, ensuring accurate and up-to-date records. The system will also include advanced filtration capabilities, enabling HR professionals to search and manage client data with ease. Data visualization will play a crucial role in the application, allowing HR professionals to gain insights from the collected data. Graphs, charts, and other visual representations will provide a comprehensive overview of key HR metrics, empowering professionals to make informed decisions and identify trends or patterns.

The reporting functionality of the system will enable HR professionals to generate customized reports, ensuring that crucial information is readily available for analysis and record-keeping purposes. This feature will streamline the process of generating reports, saving time and effort for HR professionals.

The application will also support payment invoicing, allowing HR professionals to generate invoices for clients who have made payments through various methods such as RTGS, UPI, or NFT. Additionally, the system will facilitate data downloading, enabling HR professionals to extract required data in formats such as CSV for further analysis or integration with external tools.

**1.5 Operating Environment - Hardware and Software**

**HARDWARE REQUIREMENTS**

* Operating System: operates on multiple platforms, including Windows, macOS, and Linux
* Web Browser: Google Chrome, Mozilla Firefox, or Safari.
* RAM: 4GB +
* Storage: 1-2 GB

**SOFTWARE REQUIREMENTS**

* FRONTEND: HTML5 , CSS3 , JavaScript , Bootstrap
* BACKEND: Python, Django web framework, ORM
* DATABASE: SQLite, Postgres
* EDITOR: VScode

**1.6 Brief Description of Technology Used**

Operating System:

* The HR software application is designed to be compatible with multiple operating systems such as Windows, macOS, and Linux. This compatibility ensures that HR professionals can use the application regardless of their preferred platform, increasing accessibility and usability.
* The application leverages platform-specific features and APIs to provide a seamless user experience and take advantage of system capabilities.

Web Browser:

* The application is developed to work seamlessly on popular web browsers such as Google Chrome, Mozilla Firefox, and Safari. Compatibility with these browsers ensures that HR professionals can access and use the application without any compatibility issues, enhancing user experience and convenience.
* The application utilizes modern web standards, adheres to W3C recommendations, and incorporates progressive enhancement techniques to ensure broad browser compatibility.

RAM:

* The recommended minimum RAM of 4GB is based on the application's resource requirements and performance considerations. Sufficient RAM ensures smooth and efficient operation, allowing the application to handle complex operations, process large data sets, and respond quickly to user interactions.
* With ample RAM, the application can cache frequently accessed data, optimize memory usage, and deliver a responsive user experience.

Storage:

* The allocated 1-2 GB of storage space is designed to accommodate the storage needs of the application. It allows for the storage of necessary files, databases, and related assets, ensuring seamless operation and preventing issues related to insufficient disk space.
* The application may store user data, documents, media files, configuration files, and other resources, providing a comprehensive and scalable solution for HR professionals' needs.

Frontend:

The frontend of the application utilizes HTML5, CSS3, JavaScript, and Bootstrap to create a modern and interactive user interface.

* HTML5 provides a rich set of elements, APIs, and semantic markup, allowing for the structuring and presentation of content effectively.
* CSS3 offers extensive styling capabilities, including advanced layout techniques, animations, and responsive design features, resulting in a visually appealing and consistent UI.
* JavaScript adds interactivity and dynamic functionality to the frontend, enabling features such as form validation, data manipulation, real-time updates, and client-side data processing.
* Bootstrap, as a CSS framework, provides pre-designed components, grids, and responsive utilities, ensuring a consistent and mobile-friendly UI across different devices and screen sizes.

Backend:

* Python serves as the backend programming language due to its simplicity, readability, and extensive ecosystem of libraries and frameworks.
* Django, a powerful Python web framework, is utilized to build the backend infrastructure of the application. It offers a high-level abstraction for handling common web development tasks, including URL routing, request handling, database management, and user authentication.
* Django's ORM simplifies database interactions by mapping database tables to Python objects, reducing the need for manual SQL queries and enhancing productivity.
* The combination of Python and Django provides a robust, secure, and scalable backend foundation for the HR software application.

Database:

* The HR software application supports both SQLite and Postgres as database management systems.
* SQLite is a lightweight, file-based database suitable for small-scale applications. It is self-contained, requiring no separate server setup, and simplifies deployment and management.
* Postgres, on the other hand, is a feature-rich and scalable relational database management system suitable for larger applications. It offers advanced database features, performance optimizations, and support for concurrent access and high data volumes.
* The choice of database systems provides flexibility based on the application's scale, performance requirements, and specific use cases.

Editor:

* Visual Studio Code (VScode) is recommended as the code editor for developing and customizing the HR software application.
* VScode offers a rich set of features, including syntax highlighting, code completion, linting, debugging, and version control integration.
* The extensibility of VScode allows HR professionals to customize their development environment with various extensions, enhancing productivity and collaboration.
* With support for multiple programming languages, including Python, VScode provides an efficient and user-friendly code editing experience.

Jasmine:

* Jasmine is a popular JavaScript testing framework that can be integrated into Django applications to enable automated testing of frontend JavaScript code.
* With Jasmine, developers can write test cases and specifications to verify the behavior and functionality of JavaScript code used in the frontend of the Django application.
* Jasmine provides a syntax that resembles natural language, making it readable and expressive.
* By using Jasmine in Django, developers can ensure the correctness and robustness of the frontend JavaScript code, which contributes to the overall stability and quality of the HR software application.

**CHAPTER 2**

**Proposed System**

**2.1 Study of Similar Systems**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr No.** | **Refereed From** | **Application Name** | **Name of Modules** | **How it Works** |
| 1 | HRMS Guide | Workday | Core HR, Payroll, Benefits, Time and Attendance, Talent Management | Workday offers a comprehensive suite of modules for managing core HR processes, payroll, benefits administration, time and attendance tracking, and talent management. The Core HR module handles employee data, including personal information, employment history, and organizational structure. The Payroll module automates payroll processing, tax calculations, and reporting. Benefits administration facilitates the management of employee benefits and open enrollment processes. Time and Attendance module tracks employee attendance, leave accruals, and absence management. Workday's cloud-based solution integrates these modules, providing HR professionals with a unified platform for managing HR processes. |
| 2 | G2 | BambooHR | Employee Database, Time Off Tracking, Performance Management, Recruitment, Onboarding | BambooHR focuses on core HR functions and employee management. The Employee Database module allows HR professionals to maintain employee records, including personal information, employment history, and documents. Time Off Tracking simplifies leave management by providing a centralized system for requesting, approving, and tracking time off. The Performance Management module helps streamline performance evaluations, goal setting, and feedback processes. The Recruitment module assists in job posting, applicant tracking, and interview scheduling. The Onboarding module ensures a smooth transition for new hires by automating the onboarding process and managing necessary documentation. |
| 3 | Capterra | ADP Workforce Now | Payroll, Time and Attendance, Benefits Administration, HR Management | ADP Workforce Now is a comprehensive HR software solution designed for small to mid-sized businesses. The Payroll module handles payroll processing, tax calculations, and direct deposit management. Time and Attendance module allows employees to clock in/out, tracks attendance, and provides scheduling capabilities. The HR Management module covers employee records management, onboarding, performance management, and compliance tracking. ADP Workforce Now offers integration with external systems and provides self-service features for employees and managers. The system streamlines HR processes and ensures compliance with payroll and HR regulations. |

**2.2 Feasibility Study**

There are mainly three kind of feasibility study that are equally important for this report:

* Technical Feasibility
* Economic Feasibility
* Operational Feasibility

**Operational Feasibility**

The operational feasibility of Edify Labs Business Tool is high. The system will provide a userfriendly and intuitive interface for HR professionals to manage their clients' HR-related data and activities. The system will be designed to automate and streamline HR-related tasks, such as scheduling, data visualization, and report generation, saving time and improving efficiency. The system will also be designed to be customizable, allowing users to tailor the system to their specific needs and preferences.

**Economic Feasibility**:

The economic feasibility of Edify Labs Business Tool is high. The system will provide a costeffective HR solution for growing organizations. The system will be designed to reduce manual HR-related tasks, such as data entry and report generation, which can save time and improve productivity. The system will also be designed to reduce the need for manual labor, such as HR administrative staff, which can result in cost savings for organizations. Additionally, the system will be priced competitively, making it accessible for small and medium-sized businesses.

**Technical Feasibility**

The technical feasibility of Edify Labs Business Tool is high. The system will be built on the Python Django web framework, which is a reliable and widely-used framework for building web applications. The system will be designed to be scalable, modular, and flexible, allowing for future updates and modifications. The system will also be designed to integrate with other HR-related software, such as payroll management software, to provide a comprehensive HR solution.

**2.3 Objectives of Proposed System**

The objective of Edify Labs Business Tool is to provide HR professionals with an easy-to-use web-based platform to manage their clients' information, courses, scheduling, and related activities. The platform is designed to streamline HR processes, reduce administrative workload, and increase efficiency, making it easier for HR professionals to get their work done in less time. Additionally, the platform is intended to provide valuable insights into client data, helping Edify Labs to make informed decisions and drive growth.

The objective of the Edify Labs Business Tool is to empower HR professionals by providing them with a comprehensive and user-friendly web-based platform to manage their clients' information and related activities. The tool aims to simplify HR processes and streamline operations, ultimately reducing administrative workload and increasing overall efficiency.

By offering features such as an HR dashboard, scheduling, filtration, data visualization, report generation, payment invoicing, and data downloading, the platform equips HR professionals with the necessary tools to effectively manage their clients' information. The HR dashboard serves as a centralized hub where HR professionals can access and update client details, track progress, and perform various CRUD (Create, Read, Update, Delete) operations.

The scheduling feature enables HR professionals to efficiently manage course schedules and appointments with clients. They can easily view, create, and modify schedules, ensuring that client sessions and meetings are organized and well-coordinated. This helps in optimizing time management and ensuring effective communication with clients.

The platform's data visualization capabilities provide HR professionals with valuable insights into client data. By presenting data in visually appealing graphs and charts, the platform facilitates the analysis of trends, patterns, and performance metrics. These insights enable HR professionals to make data-driven decisions, identify areas for improvement, and tailor their services to meet client needs more effectively.

Report generation functionality allows HR professionals to generate comprehensive reports on client activities, progress, and performance. These reports serve as a documentation tool, providing a clear overview of client engagements and outcomes. HR professionals can leverage these reports for internal purposes, client communication, and performance evaluation.

Payment invoicing features enable HR professionals to generate invoices for clients who have made payments for services rendered. This simplifies the billing process and ensures accurate financial records. Additionally, the platform supports downloading data in various formats, such as CSV, facilitating further analysis and integration with external tools.

**2.4 User Requirements**

* User-Friendly Interface: The software system should have an intuitive and user-friendly interface that is easy to navigate and understand, even for users with limited technical expertise.
* Client Management: The system should provide a centralized hub, such as an HR dashboard, for HR professionals to efficiently manage client information. It should allow users to add, update, and organize client details effortlessly.
* CRUD Operations: The system should support CRUD (Create, Read, Update, Delete) operations for managing client information, enabling HR professionals to easily perform necessary actions on client data.
* Scheduling Capabilities: The system should have scheduling features that allow HR professionals to schedule meetings, appointments, and other HR-related activities. This functionality should support effective time management and enable users to set reminders.
* Data Filtration: The system should include advanced filtration capabilities that enable HR professionals to search and manage client data efficiently. Users should be able to filter data based on specific criteria and easily retrieve the desired information.
* Data Visualization: The system should provide data visualization tools, such as graphs, charts, and other visual representations, to present HR-related information in a graphical format. This functionality should help HR professionals track and analyze data effectively and identify trends or patterns.
* Customized Reporting: The system should allow HR professionals to generate customized reports for analysis and record-keeping purposes. Users should be able to select specific data fields and generate reports in a format that is easy to understand and share.
* Invoicing Functionality: The system should support the creation of payment invoices for clients. It should facilitate the generation of invoices using various payment methods, such as RTGS, UPI, or NFT, and ensure accuracy in billing and payment tracking.
* Data Security: The system should implement secure authentication and data protection measures to ensure that only authorized users can access the system. It should also ensure the confidentiality of sensitive HR information.
* Integration and Data Availability: The system should integrate with databases and external networks to ensure accurate and up-to-date information availability for HR professionals. It should support data synchronization and provide seamless integration with external tools, if required.
* Data Downloading: The system should allow HR professionals to download required data in formats such as CSV for further analysis or integration with external tools. This functionality should enable users to extract and manipulate data as needed.

**CHAPTER 3**

**Analysis and Design**

**3.1 System Requirements (Functional and Non-Functional requirements)**

**Functional Requirements:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case Name** | **Id** | **Trigger** | **Precondition** | **Basic Path** | **Post Condition** |
| User-Friendly Interface | UC-001 | HR professional wants to navigate the system easily. | HR professional has appropriate system access | 1. HR professional logs into the system . 2. HR professional is presented with an intuitive and user-friendly interface. 3. HR professional can easily navigate through different sections and modules of the system | HR professional can perform tasks and access information efficiently. |
| Data Analysis | UC-002 | HR professional wants to gain insights from HR-related data. | HR professional has appropriate system access and relevant data available. | 1. HR professional logs into the system. 2. HR professional selects the desired data for analysis, such as employee performance, leave records, or training data. 3. HR professional uses the data analysis tools provided by the system to derive insights, identify trends, and uncover areas of improvement. | HR professional gains valuable insights from the HR data. |
| Data Filtration | UC-003 | HR professional wants to filter and search HRrelated data. | R professional has appropriate system access and data available for filtration. | 1. HR professional selects the desired data section or module. 2. HR professional specifies the filtration criteria, such as client name, date range, or payment status. 3. The system filters the data HR professional can efficiently search and manage HR data based on specific criteria on the specified criteria and presents the filtered results to the HR professional | HR professional can efficiently search and manage HR data based on specific criteria. |
| Reporting | UC-004 | HR professional wants to generate customized reports. | HR professional has appropriate system access and relevant data available. | 1. HR professional selects the desired data or report template. 2. HR professional customizes the report parameters, such as data fields, date range, or clientspecific information. 3. The system generates the customized report, incorporating the specified parameters and data. | HR professional obtains a customized report that can be used for analysis, recordkeeping, or sharing with clients. |
| Scheduling Meetings | UC-005 | HR professional needs to schedule meetings or appointments | HR professional as appropriate system access and meeting details | HR professional navigates to the scheduling module. 2. HR professional enters the meeting details, including date, time, participants, and location. 3. The system checks for conflicts and availability of participants. 4. If there are no conflicts, the system saves the meeting details and sends notifications to the participants. | The meeting appointment is successfully scheduled, and participants are notified about the details. |
| Customization | UC-006 | HR professional wants to customize the system for a specific client | HR professional has appropriate system access and client information. | 1. HR professional logs into the system. 2. HR professional selects the client they want to customize the system for. 3. HR professional identifies the customization requirements based on client needs. 4. HR professional modifies workflows, reports, and user interfaces to align with client requirements | The system is customized according to the specific needs of the client |

**Non-Functional Requirements:**

**Usability:**

The system should have a user-friendly interface with intuitive navigation and clear instructions.

It should be easy for HR users to learn and use the system efficiently.

**Performance:**

The system should provide fast response times, ensuring a smooth user experience even with a large number of clients and data.

It should handle concurrent user requests and database transactions efficiently.

**Security:**

The system should implement robust security measures to protect client data and user authentication. User access should be restricted based on roles and permissions to maintain data confidentiality.

**Scalability:**

The system should be designed to handle an increasing number of clients, courses, and user accounts. It should be scalable to accommodate future growth and additional features.

**Reliability:**

The system should be highly reliable, with minimal downtime and data loss.

Regular backups and data recovery mechanisms should be in place to ensure data integrity.

**Compatibility:**

The system should be compatible with common web browsers such as Google Chrome, Mozilla Firefox, and Safari. It should be responsive and accessible on different devices, including desktops, laptops, and tablets.

**Data Privacy:**

The system should comply with data privacy regulations and ensure the secure storage and handling of client information.

Personal data should be encrypted and protected against unauthorized access.

**Documentation and Support:**

The system should have comprehensive documentation to guide users on its usage, features, and troubleshooting. Adequate support channels, such as email or online chat, should be available to assist users in case of any issues or inquiries.

**3.2 Data Flow Diagrams** (**DFD)**

**A close-up of a white background

Description automatically generated**

**A diagram of a company

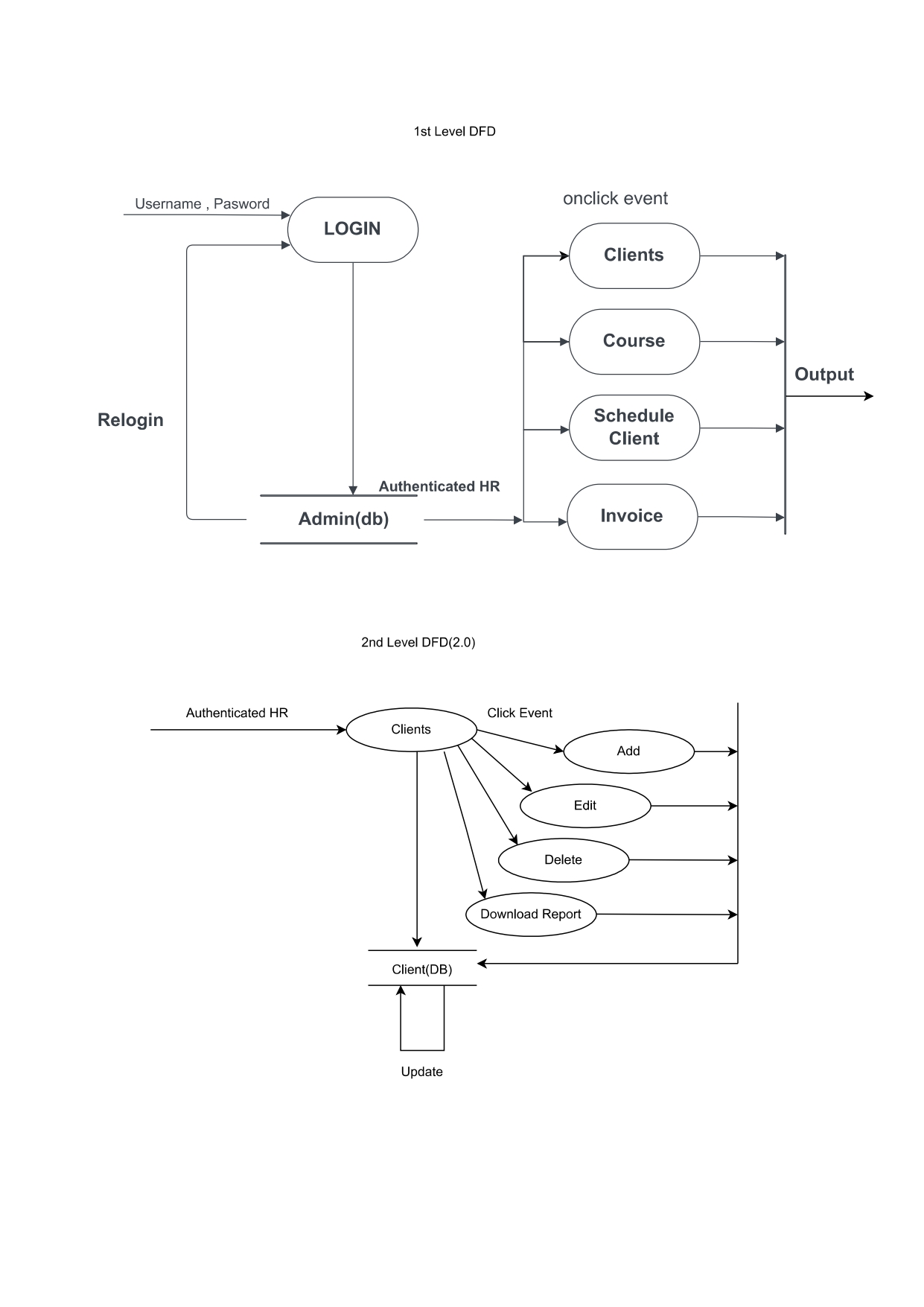
Description automatically generatedFirst level Dataflow diagram (1 (a))**

**First level Dataflow diagram (1 (b))**

**A diagram of a login

Description automatically generated**

**Second level Dataflow diagram (2a (i) )**

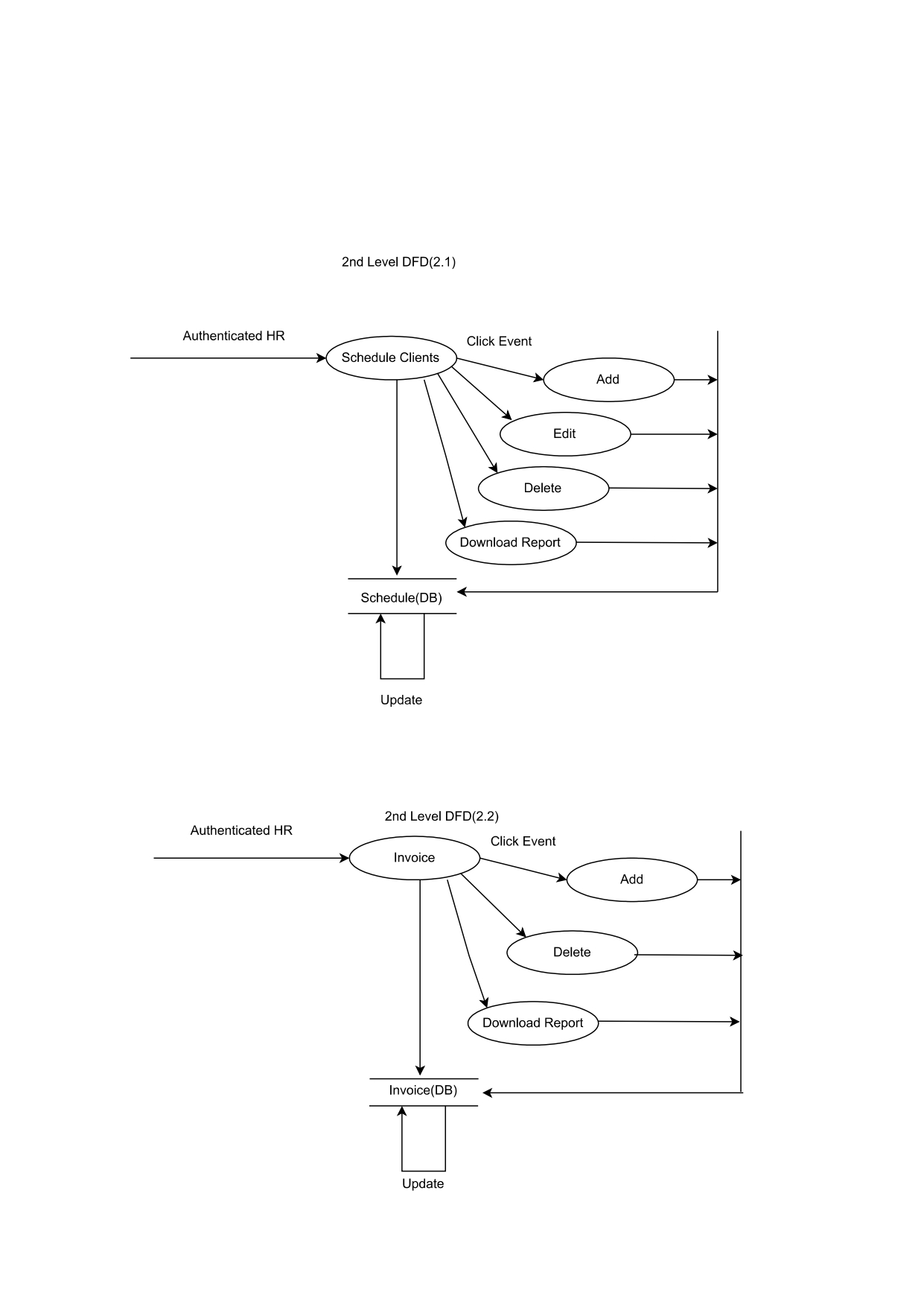
****

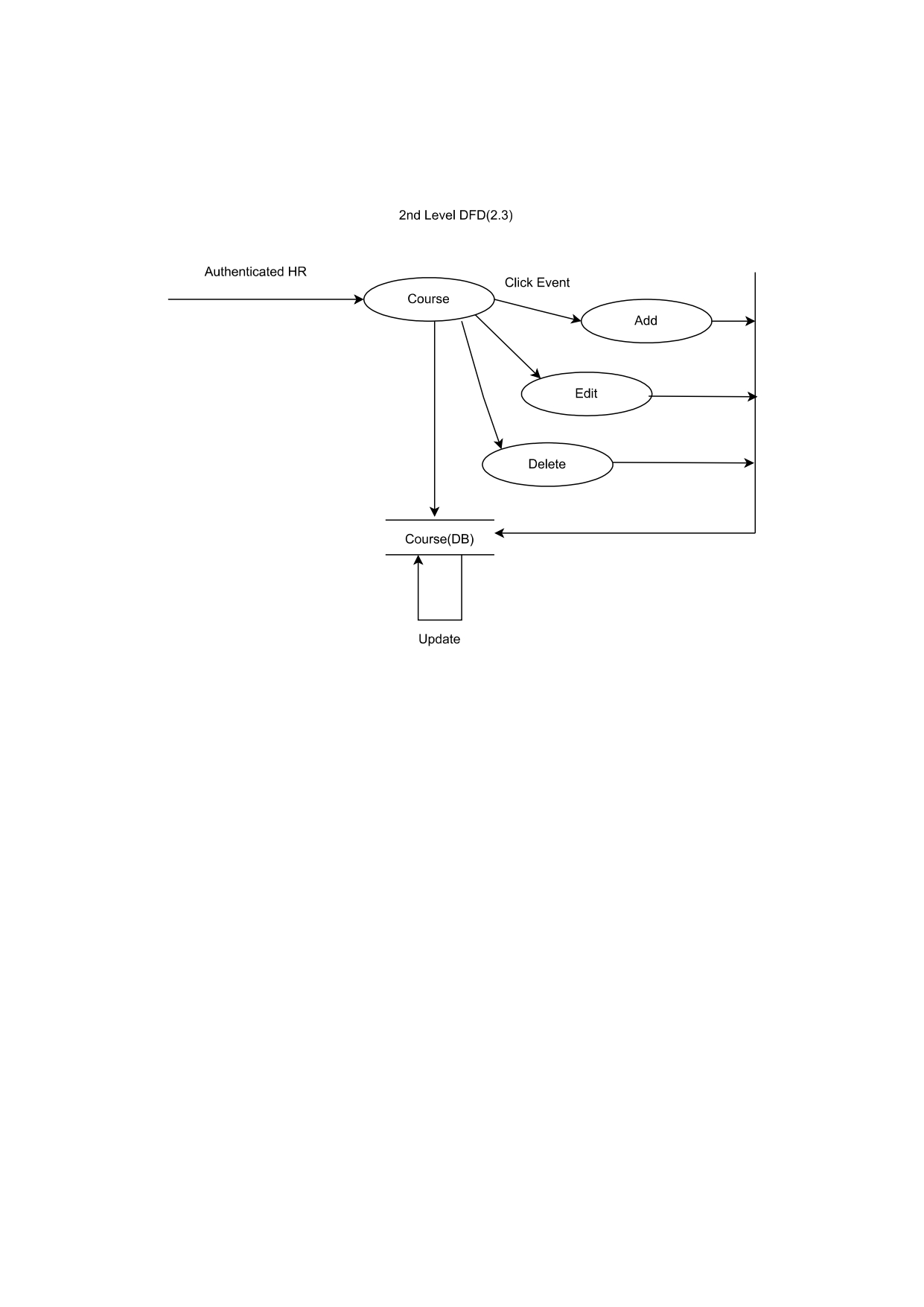
**Second level Dataflow diagram (2a (ii) )**

**A diagram of a company

Description automatically generated**

**Second level Dataflow diagram (2a (iii) )**

****

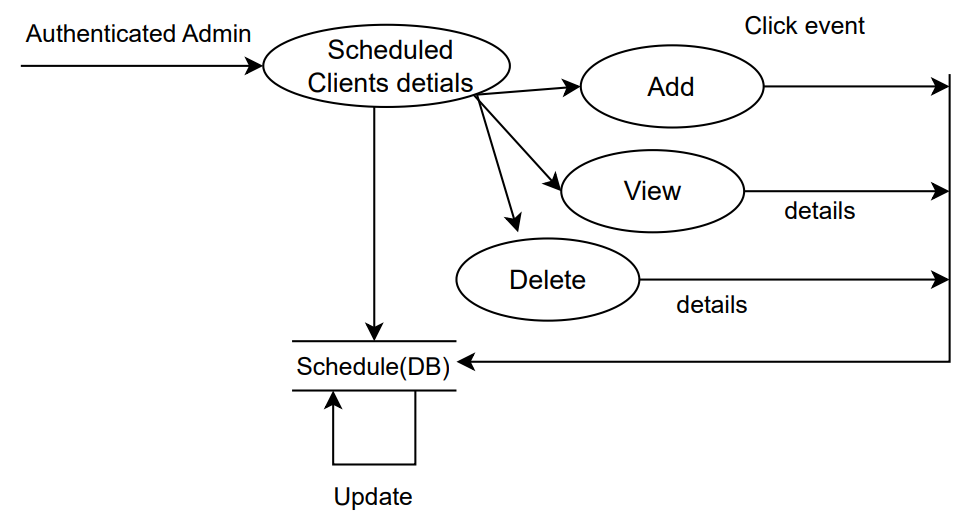
**Second level Dataflow diagram (2a (iv))**

**Second level Dataflow diagram (2b (i))**

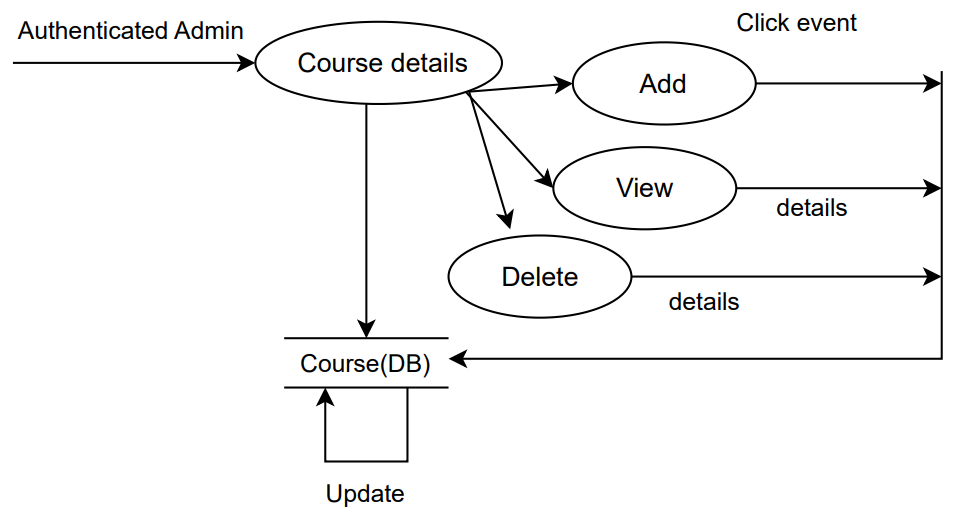
**A diagram of a software system

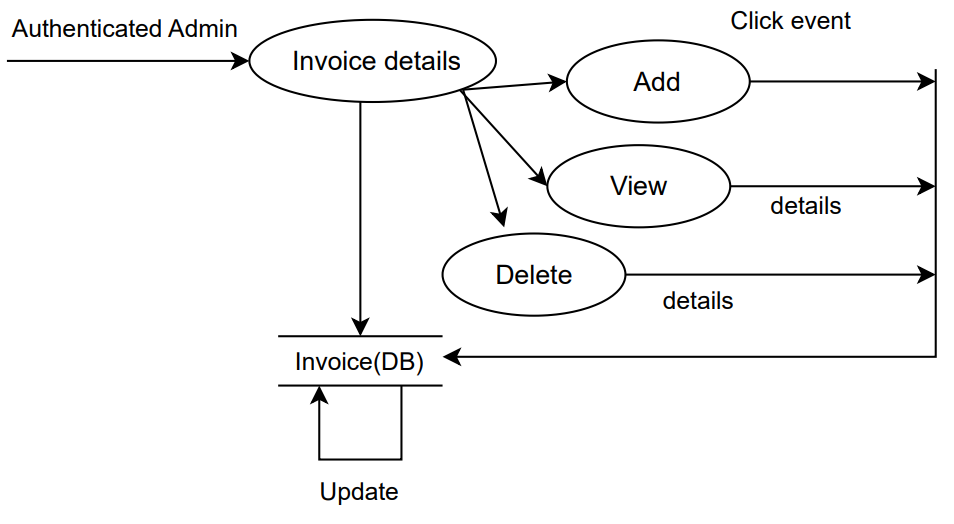
Description automatically generated**

**Second level Dataflow diagram (2b (ii))**

****

**Second level Dataflow diagram (2b (iii))**

****

**Second level Dataflow diagram (2b (iv))**

**Second level Dataflow diagram (2b (v))**

**A diagram of a system

Description automatically generated**

**3.3 Entity Relationship Diagram (ERD)**

A picture containing sketch, drawing, diagram, plan

Description automatically generated

**3.4 Table Structure**

Table Name: clientsapp\_course

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Field Name | Data Type | Constraints | Description |
| 1 | id | INTEGER | PRIMARY KEY AUTOINCREMENT | Identifier for the course |
| 2 | title | VARCHAR(100) | NOT NULL | Title of the course |
| 3 | desc | VARCHAR(100) | NOT NULL | Description of the course |
| 4 | desc | DATETIME | NOT NULL | Date of the course |

Table Name: clientsapp\_clients

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Field Name | Data Type | Constraints | Description |
| 1 | id | INTEGER | PRIMARY KEY AUTOINCREMENT | Identifier for the client |
| 2 | first\_name | VARCHAR(100) | NOT NULL | First name of the client |
| 3 | last\_name | VARCHAR(100) | NOT NULL | Last name of the client |
| 4 | gender | VARCHAR(1) | NOT NULL | Gender of the client |
| 5 | company\_name | VARCHAR(100) | NOT NULL | Company name of the client |
| 6 | email\_id | VARCHAR(100) | NOT NULL | Email ID of the client |
| 7 | contact | VARCHAR(13) | NOT NULL | Contact number of the client |
| 8 | date\_of\_birth | VARCHAR(100) | NOT NULL | Date of birth of the client |
| 9 | address | TEXT | NOT NULL | Address of the client |
| 10 | date\_added | DATETIME | NOT NULL | Date and time when the client was added |
| 11 | coaching\_type\_course\_id | BIGINT | NOT NULL | Foreign key reference to the course |

Table Name: scheduler\_scheduleclients

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Field Name | Data Type | Constraints | Description |
| 1 | id | INTEGER | PRIMARY KEY AUTOINCREMENT | Identifier for the scheduled client |
| 2 | first\_name \_S | VARCHAR(100) | NOT NULL | First name of the client |
| 3 | last\_name\_S | VARCHAR(100) | NOT NULL | Last name of the client |
| 4 | review\_call | BOOL | NOT NULL | Indicates if a review call is required |
| 5 | sessions | VARCHAR(100) | NOT NULL | Sessions for the scheduled client |
| 6 | my\_time\_field | TIME | NOT NULL | Time field for the schedule |
| 7 | date\_added | DATETIME | NOT NULL | Date and time when the schedule was added |
| 8 | coaching\_type\_course\_id | BIGINT | NOT NULL | Foreign key reference to the course |
| 9 | client\_count | INTEGER unsigned | NOT NULL CHECK (client\_count >= 0) | Number of clients |

Table Name: invoice\_invoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Field Name | Data Type | Constraints | Description |
| 1 | id | INTEGER | PRIMARY KEY AUTOINCREMENT | Identifier for the scheduled invoice |
| 2 | full\_name\_invoice | VARCHAR(100) | NOT NULL | Full name for invoice |
| 3 | address\_invoice | VARCHAR(100) | NOT NULL | Address for invoice |
| 4 | GST\_invoice | BOOL | NOT NULL | GST for invoice |
| 5 | invoice\_number | VARCHAR(100) | NOT NULL | Invoice number |
| 6 | amount\_invoice | DECIMAL | NOT NULL | Amount for the invoice |
| 7 | date\_added\_invoice | DATE | NOT NULL | Date and time when the invoice was added |
| 8 | any\_discount\_invoice | DECIMAL | NOT NULL | Any discount for the invoice |
| 9 | coaching\_type\_course\_  \_invoice\_id | BIGINT | NOT NULL | Foreign key reference to the course |
| 10 | cgst | INTEGER unsigned | NOT NULL CHECK (cgst >= 0) | CGST amount |
| 11 | sgst | INTEGER unsigned | NOT NULL CHECK (sgst >= 0) | SGST amount |
| 12 | total\_amount\_invoice | DECIMAL | NOT NULL | Total amount for the invoice |
| 13 | cgst\_amount | DECIMAL | NOT NULL | CGST amount in the invoice |
| 14 | sgst\_amount | DECIMAL | NOT NULL | SGST amount in the invoic |
| 15 | total\_amount\_payable\_invoice | DECIMAL | NOT NULL | Total amount payable for the invoice |
| 16 | particular\_invoice | TEXT | NOT NULL | Particulars for the invoice |

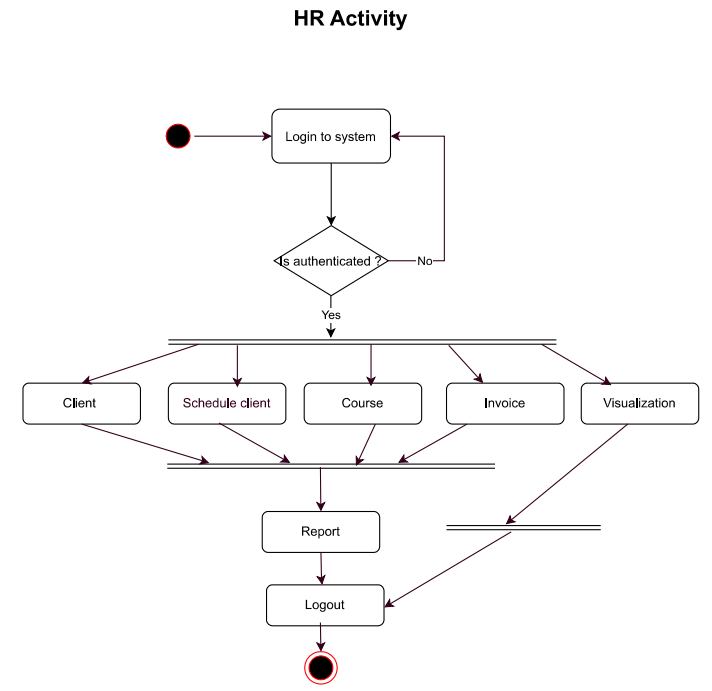
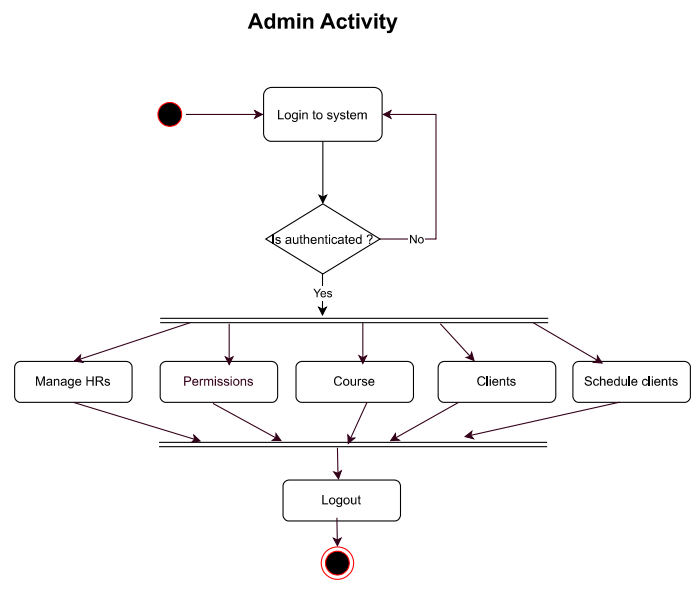
Table Name: auth\_user

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Field Name | Data Type | Constraints | Description |
| 1 | id | INTEGER | PRIMARY KEY AUTOINCREMENT | Identifier for the user |
| 2 | username | VARCHAR(100) | NOT NULL UNIQUE | User's username |
| 3 | first\_name | VARCHAR(100) | NOT NULL | User's first name |
| 4 | last\_name | VARCHAR(100) | NOT NULL | User's last name |
| 5 | email | VARCHAR(254) | NOT NULL | User's email address |
| 6 | date\_joined | TIME | NOT NULL | Date and time when the user joined |
| 7 | last\_login | DATETIME | NOT NULL | Last login date and time |
| 8 | password | VARCHAR(128) | NOT NULL | User's password |
| 9 | is\_superuser | BOOL | NOT NULL | Indicates if the user is a superuser |

A picture containing sketch, drawing, diagram, line art

Description automatically generated**3.5 Use Case Diagrams**

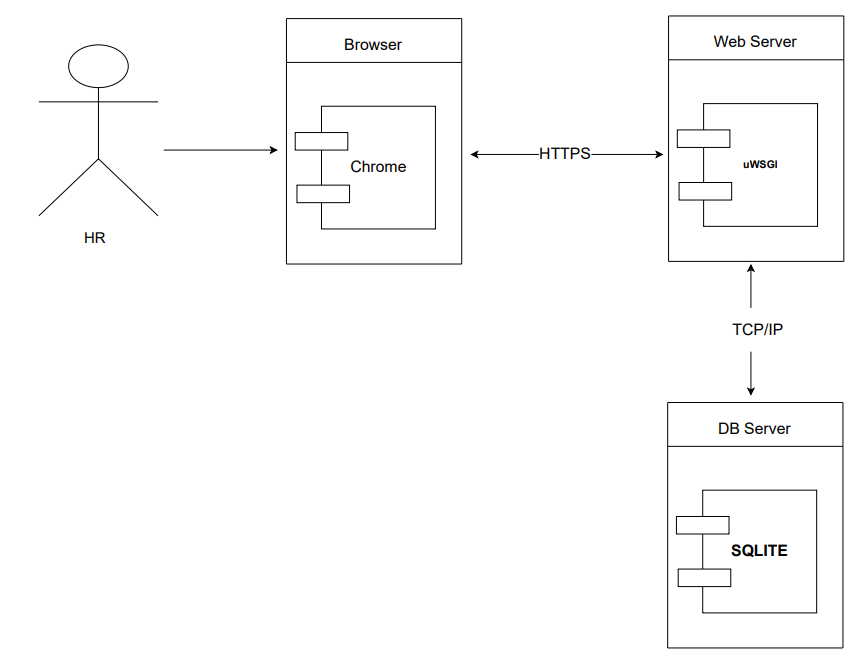
**3.6 Activity Diagram**

****

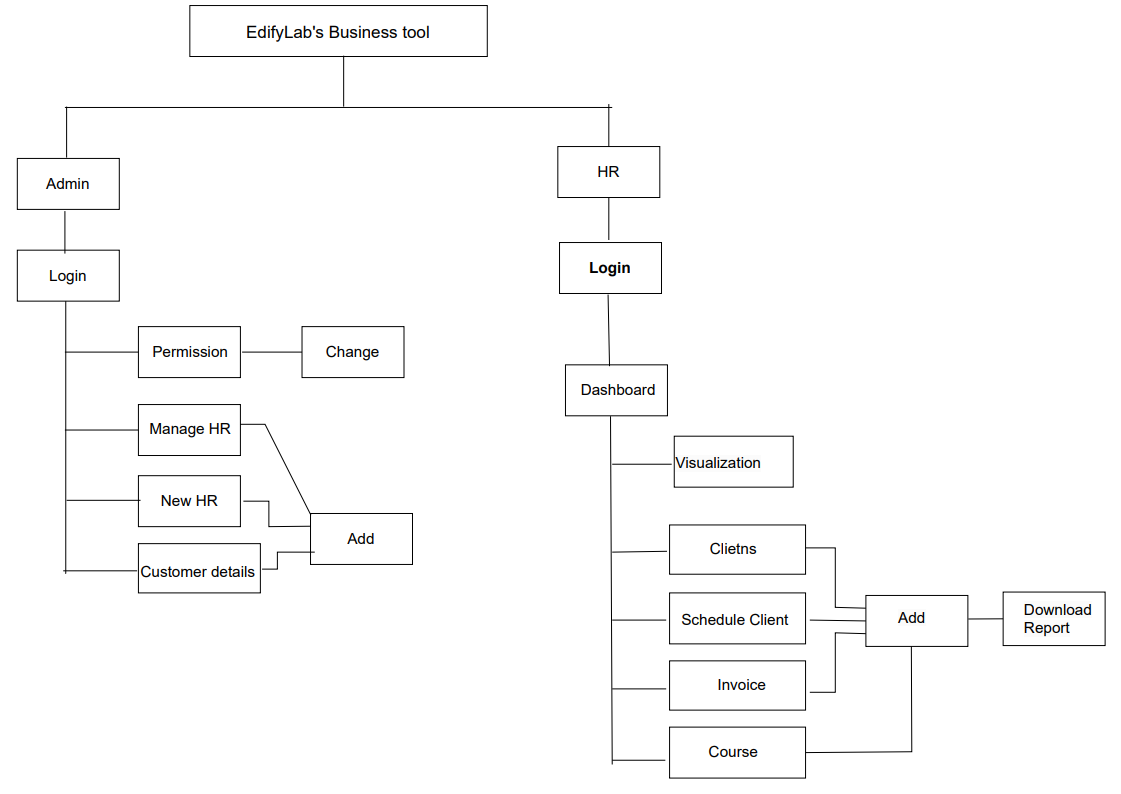
**3.7 Sequence diagram**



**3.8 Deployment Diagram**

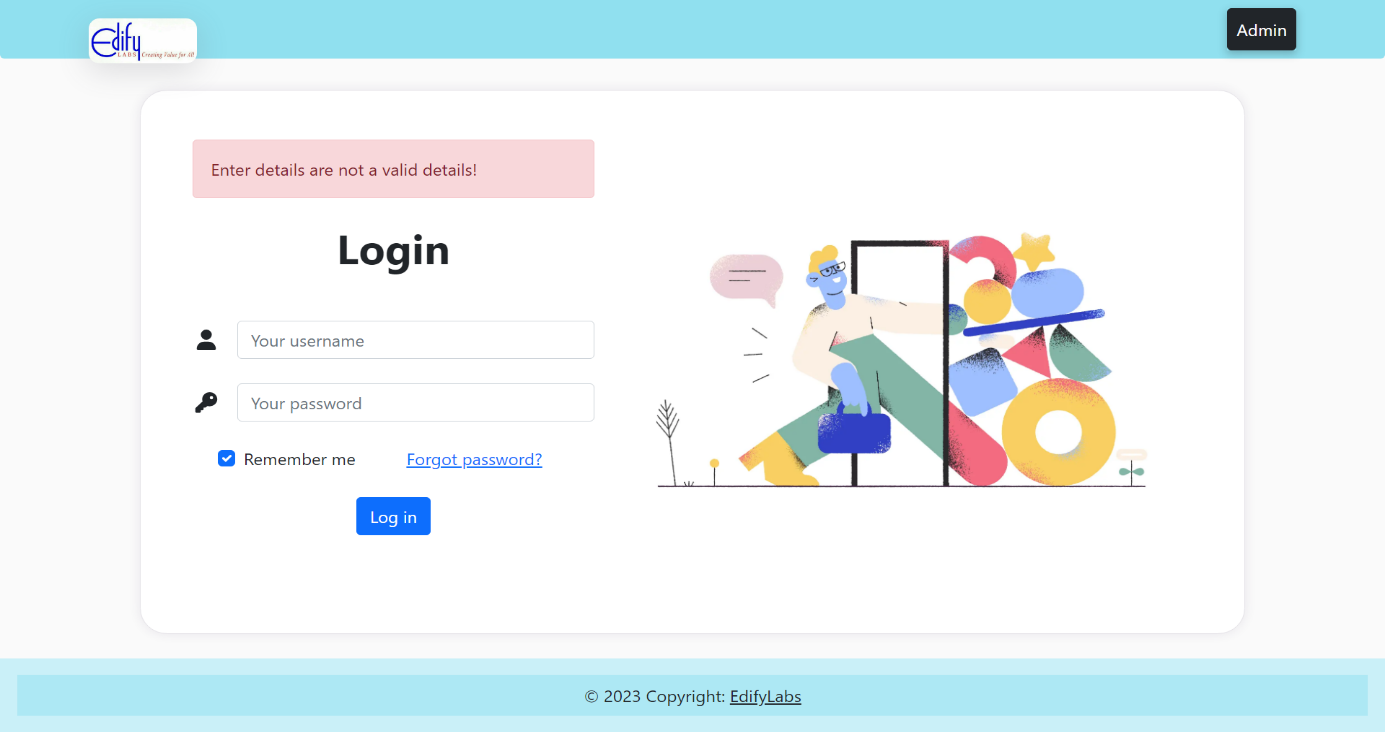


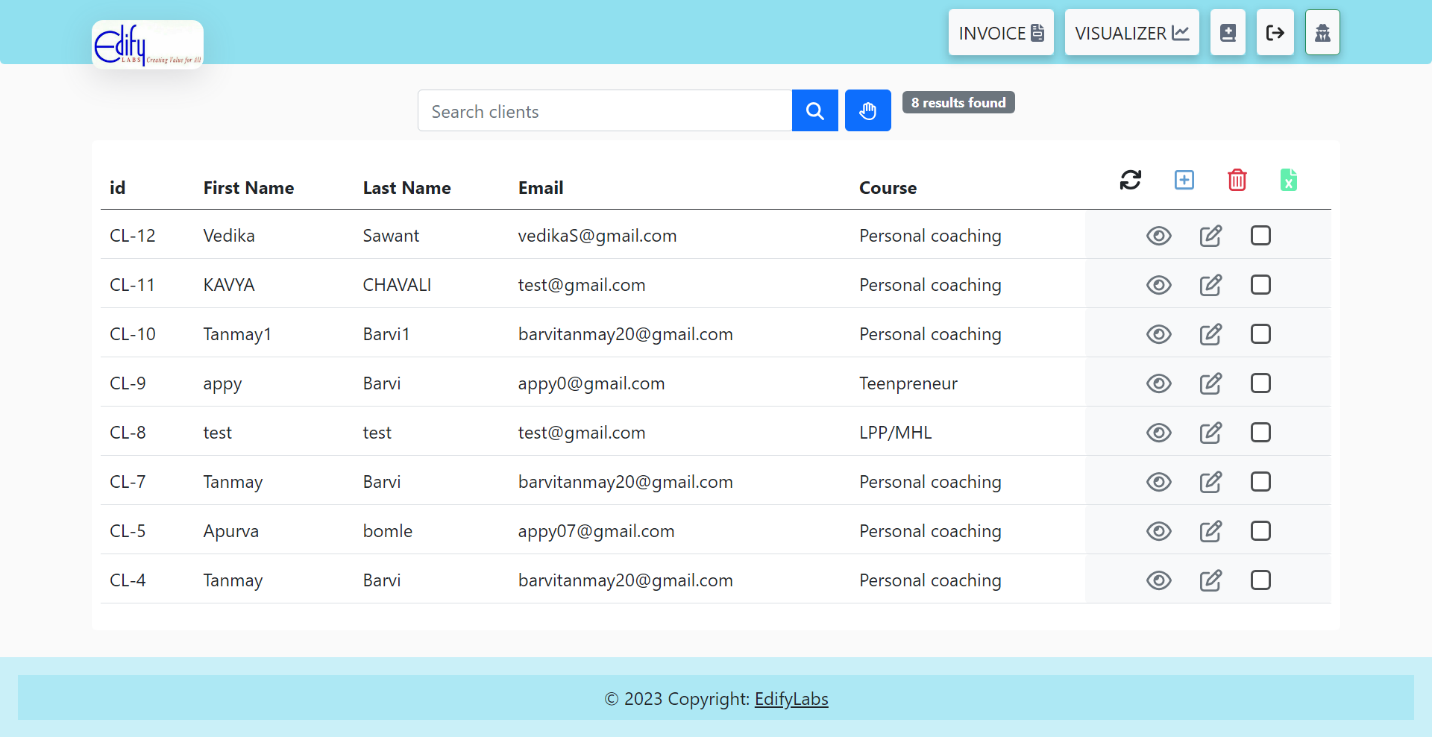
**3.9 Component Diagram**

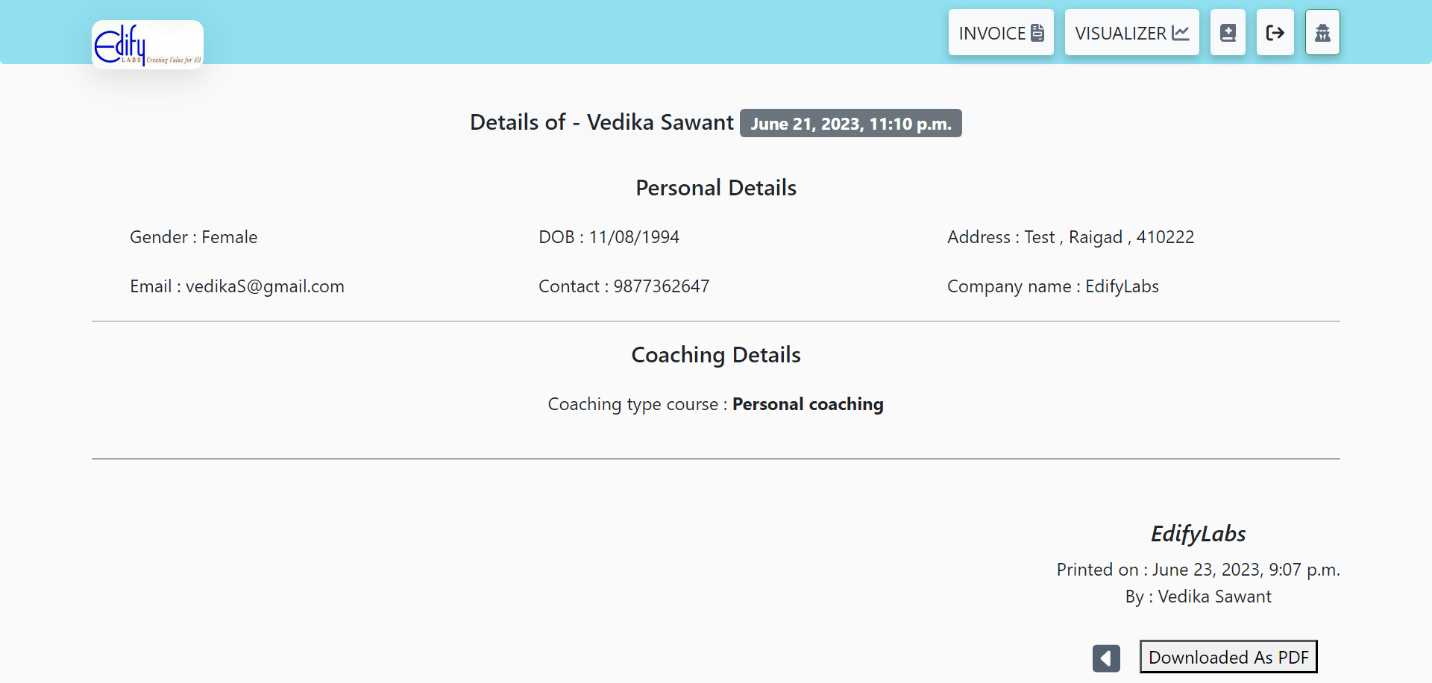


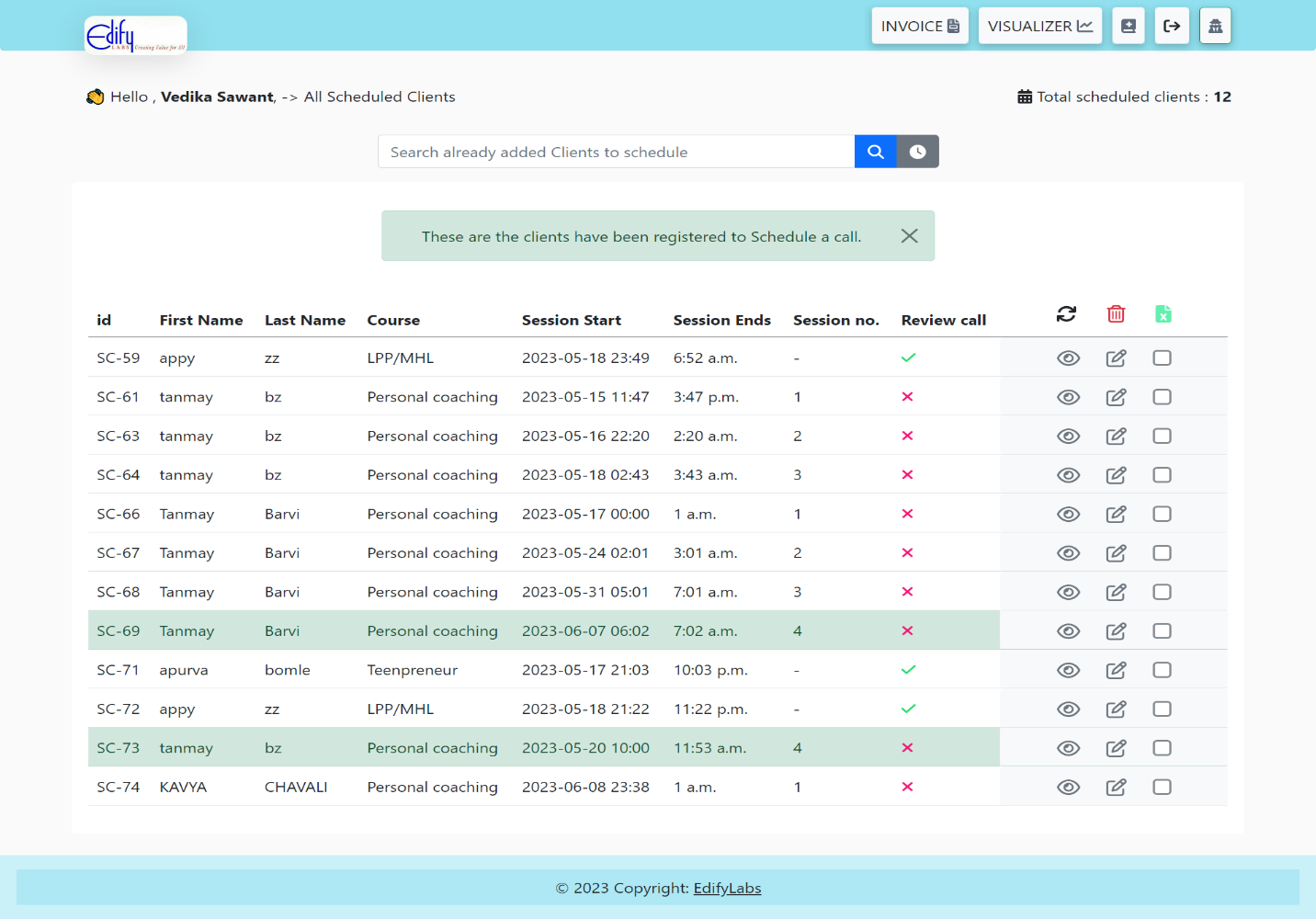
**3.10 Sample Input and Output Screen**

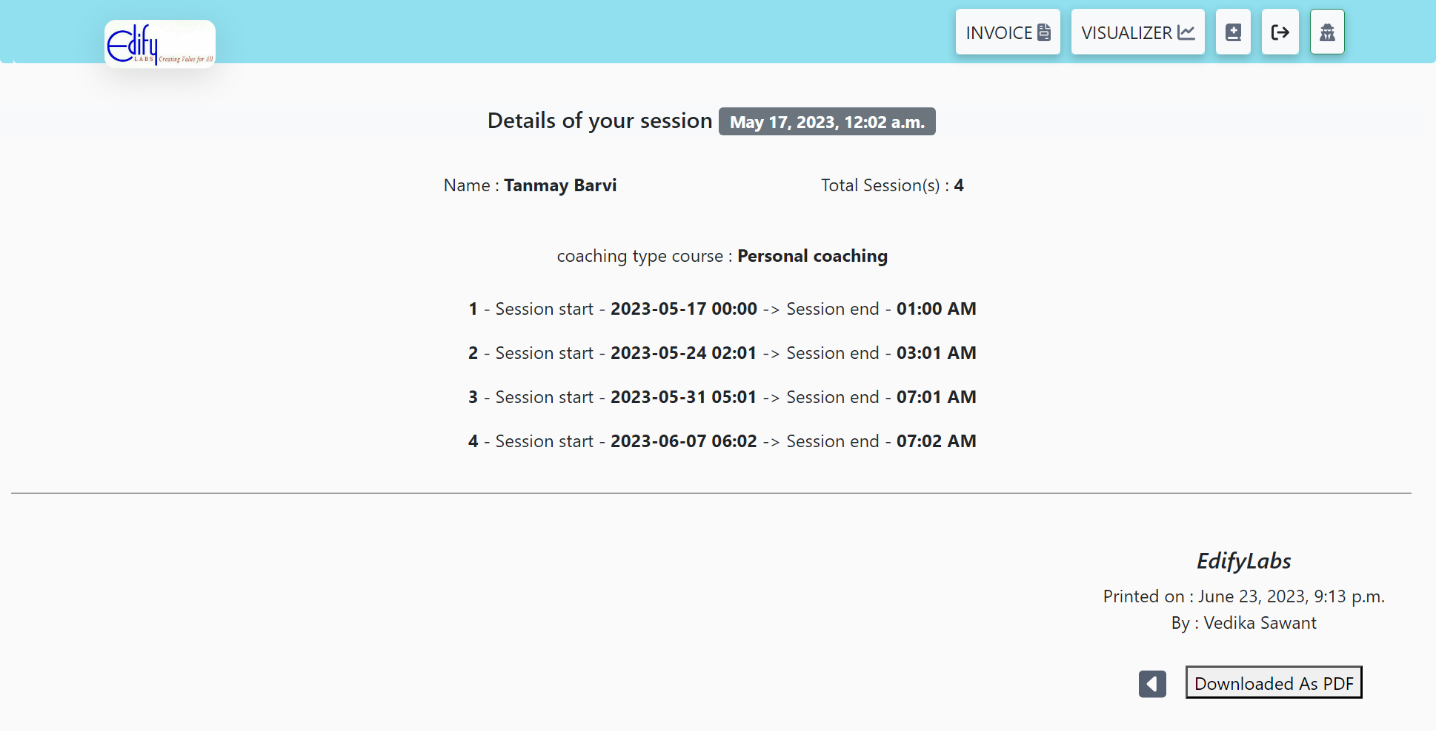
**Input -> HR login screen (with invalid inputs)**

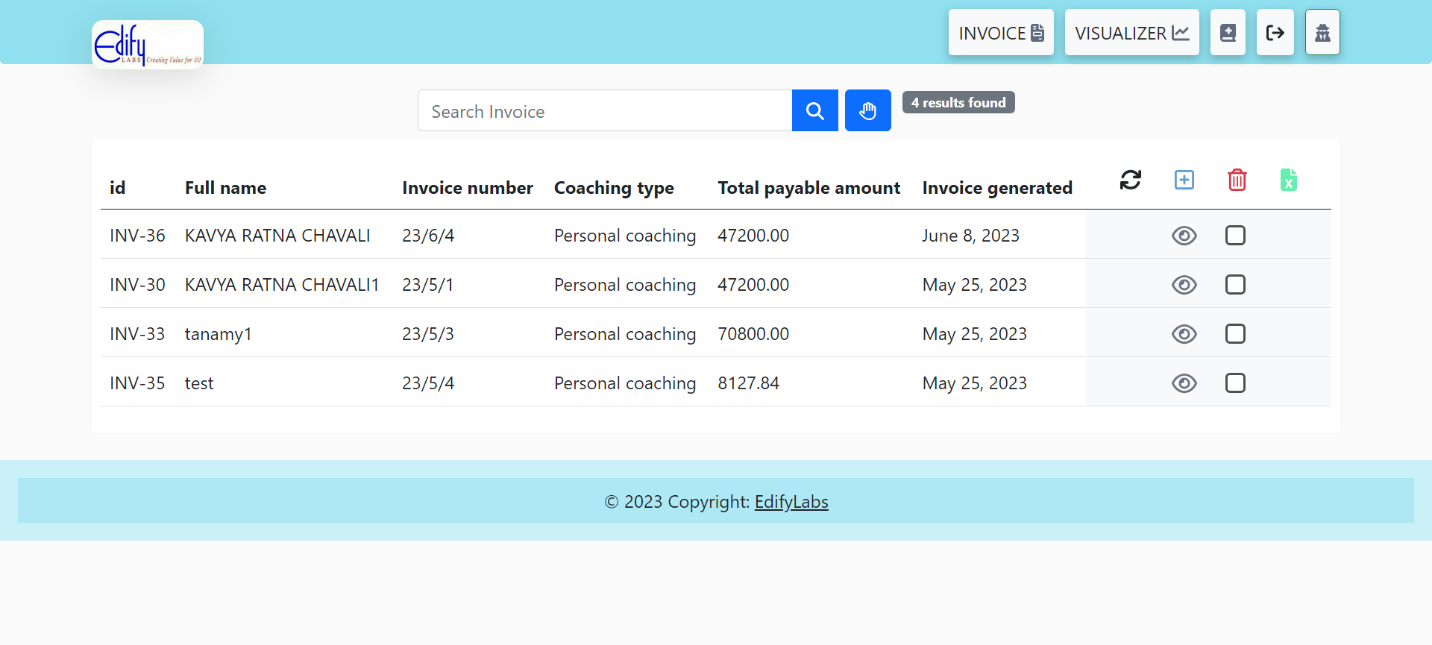
**Output -> Invalid message**

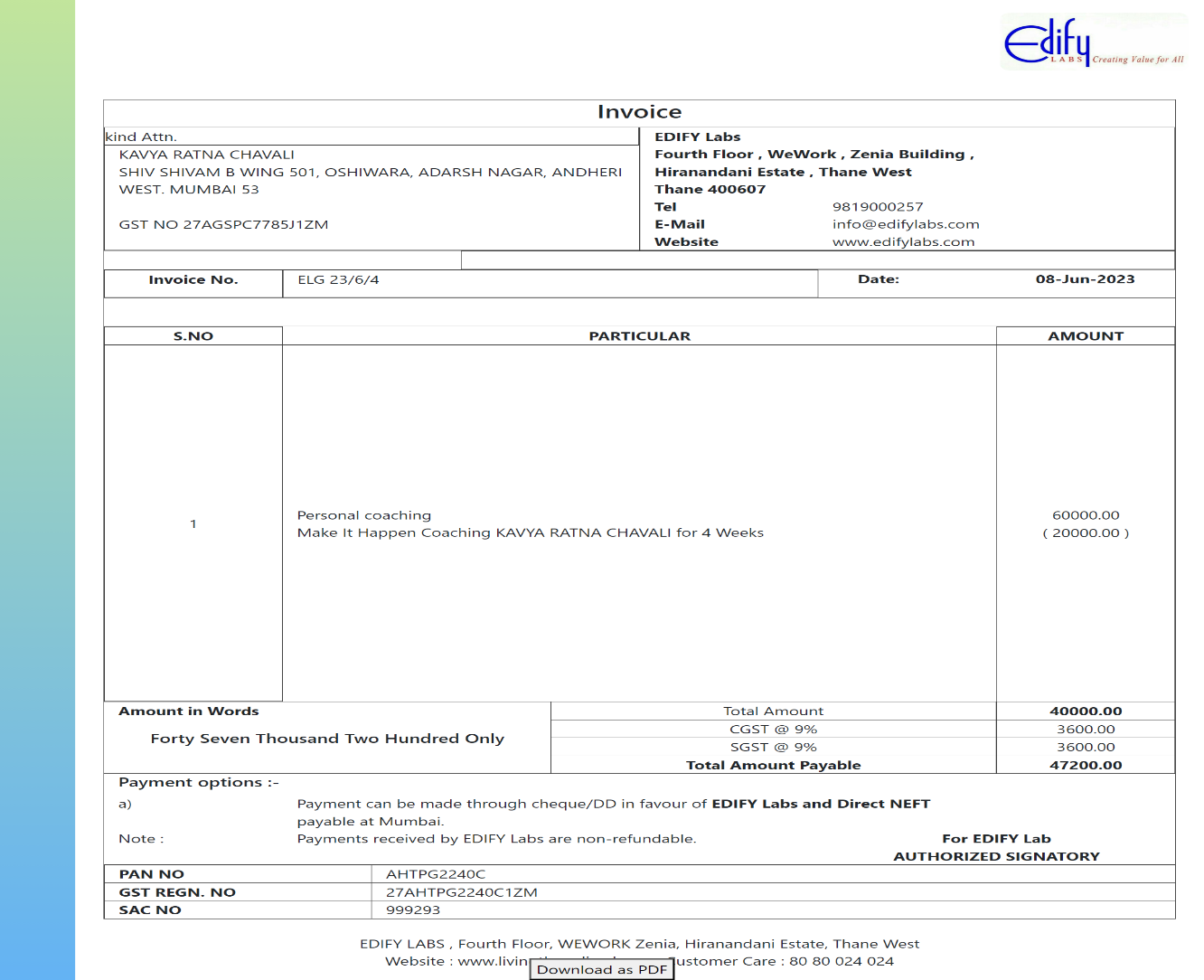
**Input -> Client’s screen**

**Output -> Download Client details(particular)**

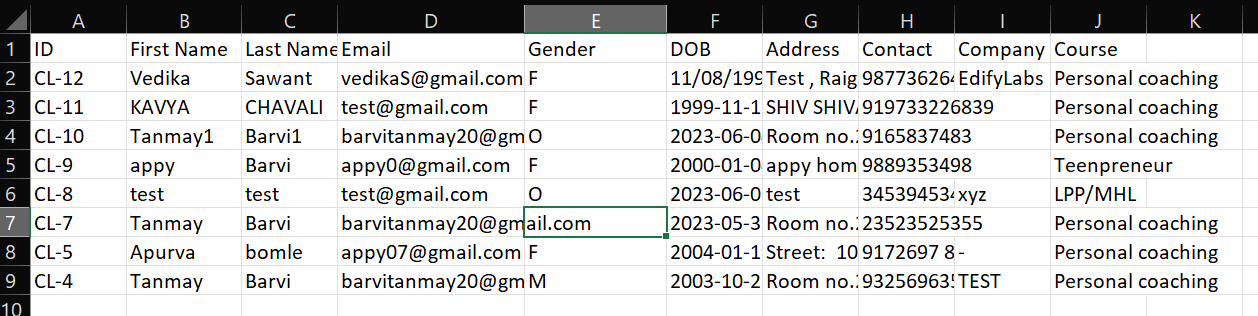
**Input -> Scheduled Clients screen**

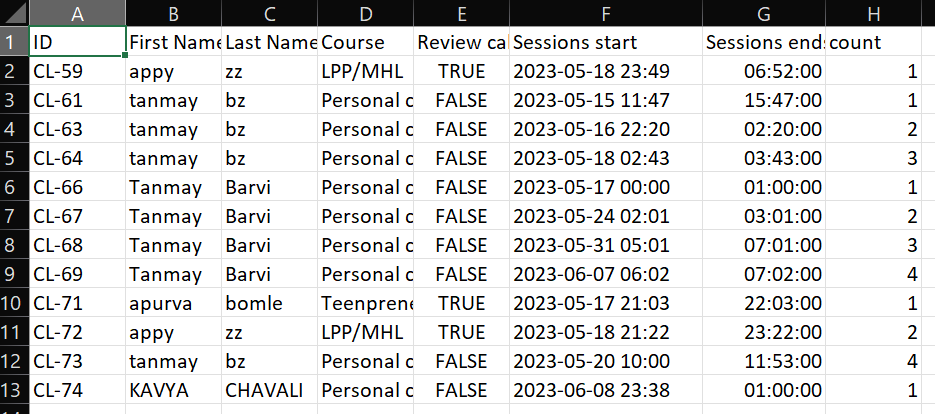
**Output -> Download Scheduled Client details(particular)**

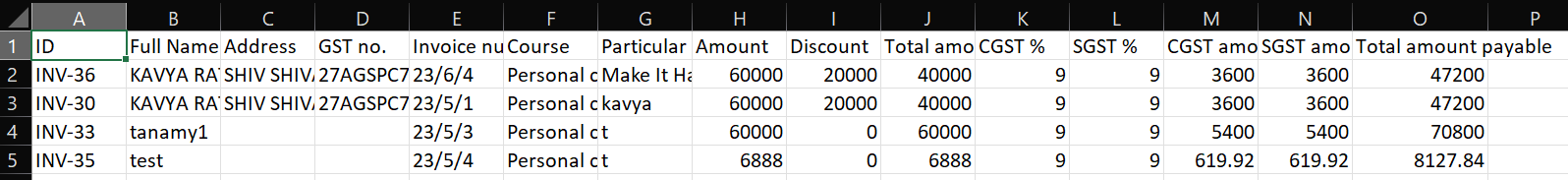
**Input -> Invoice lists screen**

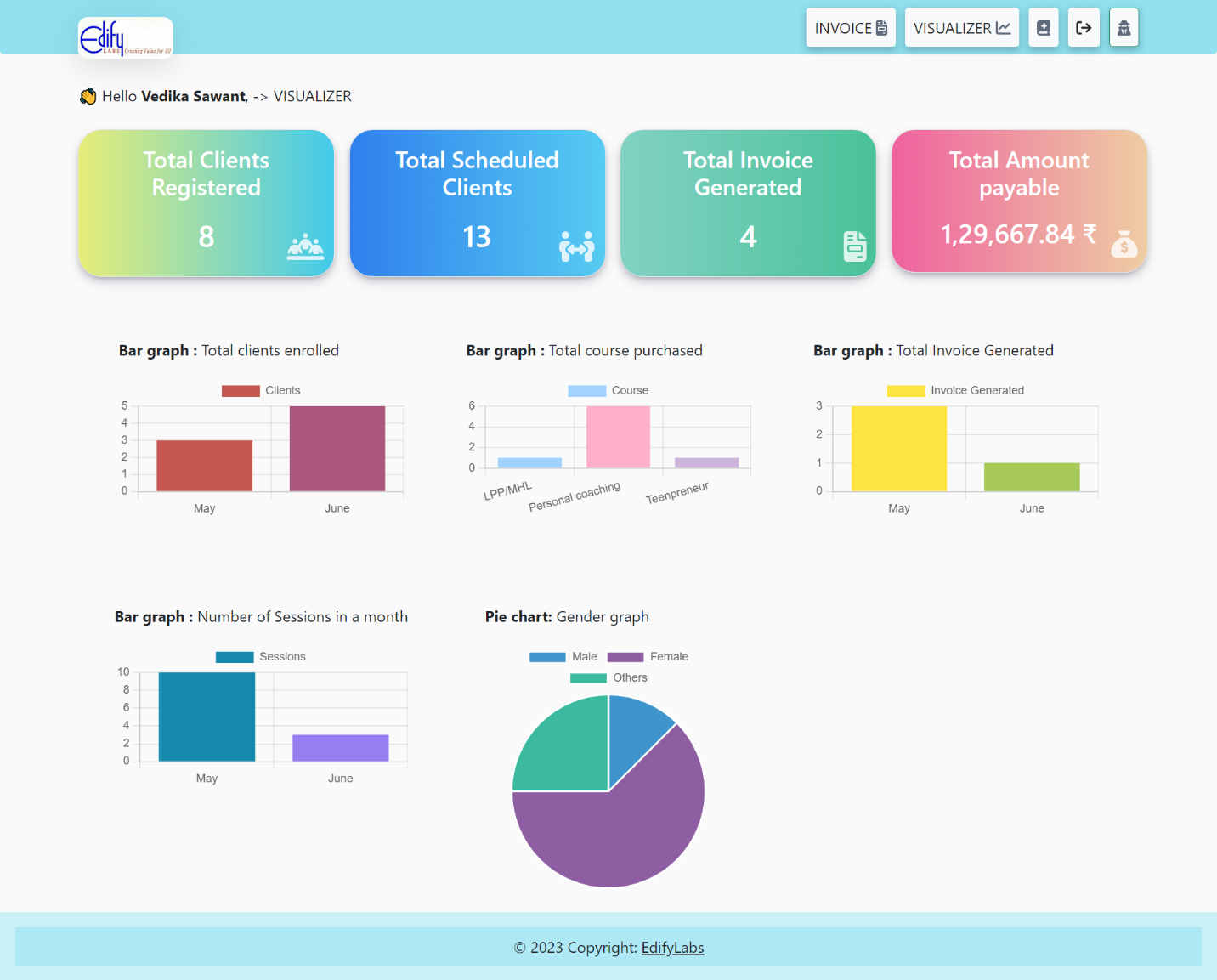
**Output -> Download invoice (particular)**

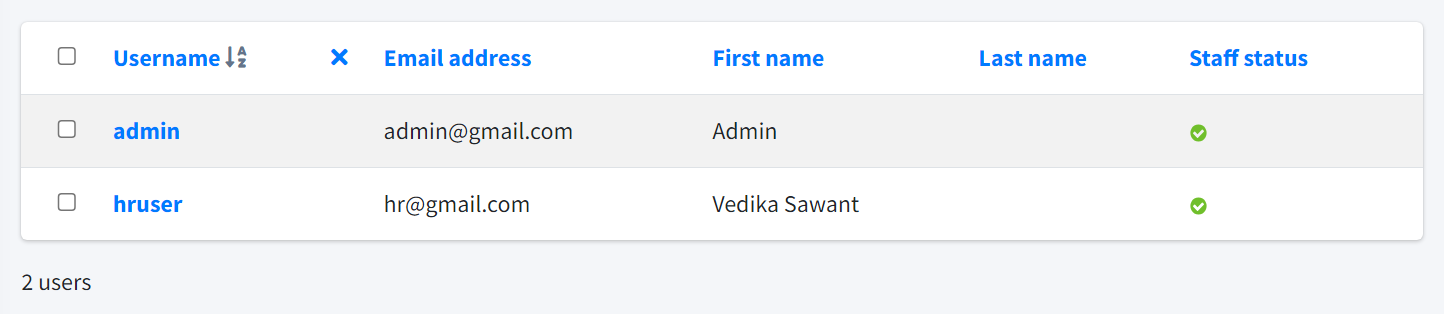
**3.11 Reports**

**Output -> Download to excel (All clients)**

**Output -> Download to excel (All Scheduled clients)**

**Output -> Download to excel (All Scheduled clients)**

**Output -> Visualization charts reports**

**Output -> Users status (Admin panel)**

**CHAPTER 4**

**Coding**

**4.1 Algorithms**

Algorithm for the login process in the Edify Labs HR Software System:

1. Start.

2. Declare variables `username` and `password`.

3. Read values of `username` and `password` from the user through the login form.

4. Validate that both `username` and `password` fields are not empty.

a. If either field is empty, display an error message and go to Step 9.

5. Query the database to find the user with the provided `username`.

a. If the user is not found in the database, display an error message, and go to Step 9.

6. Retrieve the hashed password associated with the found user.

7. Validate the provided `password` against the hashed password stored in the database.

a. If the validation fails, display an error message, and go to Step 9.

8. If all the fields are validated and the username is a valid user then show the Dashboard page , and go to Step 9

9. End.

Algorithm for the filtration inEdify Labs HR Software System:

The **filter algorithm** in Django involves applying conditions on a queryset to retrieve a subset of objects that match the specified criteria. The algorithm follows these steps:

* Import the necessary Django models for the data you want to filter.
* Define a function, let's call it filter\_data, that takes in the filter\_criteria as a parameter.
* Initialize a variable, let's call it filtered\_data, with the initial queryset or model manager. For example, filtered\_data = Client.objects.all().
* Iterate over each attribute-value pair in the filter\_criteria:

Use the Django ORM's filter() method on filtered\_data to apply the filter based on the attribute and value.

For example, if the attribute is 'client\_name' and the value is 'John Doe', you would filter like this: filtered\_data = filtered\_data.filter(client\_name='John Doe').

* Return the filtered\_data.

**4.2 Code Snippets**

"""

>> LOGIC FUNCTIONS FOR CLIENTS

"""

from django.shortcuts import render, redirect

from django.contrib.auth.decorators import login\_required

from clientsapp.models import Clients

from .forms import ClientForm

from django.db.models import Q

import datetime

import openpyxl

from django.http import HttpResponse

# HOME -> FORM DISPLAY - ADD CLIENT - UPDATE CLIENT

@login\_required(login\_url='loginPage')

def home(request, id=0):

if request.method == "GET":

auth\_user = request.user.first\_name

all\_clients = Clients.objects.all()

male\_count = Clients.objects.filter(gender='M').count()

female\_count = Clients.objects.filter(gender='F').count()

other\_count = Clients.objects.filter(gender='O').count()

print("Male count:", male\_count)

print(female\_count)

print(other\_count)

gender\_count = [male\_count , female\_count , other\_count]

gender\_name = ["Male" , "Female" , "Others"]

if id == 0:

form = ClientForm()

else:

client = Clients.objects.get(pk=id)

form = ClientForm(instance=client)

context = {"auth\_user": auth\_user, "page": "HOME | CLIENT | Edify Lab's Business Tool ", "ClientForm": form, "total\_clients": len(all\_clients) , "gender\_count" : gender\_count , "gender\_name" : gender\_name }

return render(request, "home.html", context)

else:

if id == 0:

form = ClientForm(request.POST)

else:

client = Clients.objects.get(pk=id)

form = ClientForm(request.POST, instance=client)

if form.is\_valid():

form.save()

return redirect('/clients-list/')

# DISPLAYS ALL CLIENTS

@login\_required(login\_url='loginPage')

def clients\_list(request):

all\_clients = Clients.objects.all()

page = "All clients | Edify Lab's Business Tool"

if len(all\_clients) != 0:

context\_clients\_list = {

"clients\_list": all\_clients, "total\_clients": len(all\_clients) , "page" : page}

else:

error = "NO SILENTS 0"

context\_clients\_list = {"error": error , "page":page}

return render(request, "clients\_template/clients\_list.html/", context\_clients\_list)

# DELETE A SELECTED CLIENT

@login\_required(login\_url='loginPage')

def item\_delete(request):

if request.method == 'POST':

item\_ids = request.POST.getlist('item\_ids')

print(item\_ids , "SELECTED")

if item\_ids:

Clients.objects.filter(id\_\_in=item\_ids).delete()

return redirect('/clients-list/')

# SEARCH CLIENT -> FNAME , LNAME , EMAIL , COURSE

@login\_required(login\_url='loginPage')

def search\_client(request):

if request.method == "POST":

page = "Search clients | Edify Lab's Business Tool"

title = request.POST["title"]

lookups = Q(first\_name\_\_icontains=title) | Q(

coaching\_type\_course\_\_title\_\_icontains=title) | Q(last\_name\_\_icontains=title) | Q(email\_id\_\_icontains=title)

filter\_title = Clients.objects.filter(lookups)

print(len(filter\_title), lookups)

if title != '' and title is not None and len(filter\_title) != 0:

context\_clients\_list = {

"clients\_list": Clients.objects.all(), "filter\_title": filter\_title, "filter\_title\_total": len(filter\_title) , "page":page}

else:

error = "NO MATCH FOUND"

context\_clients\_list = {"error": error , "page":page}

return render(request, "clients\_template/clients\_list.html/", context\_clients\_list)

else:

print("---------------SOMETHING WENT WRONG IN SEARCH CLIENT-----------------------")

return redirect('/home/')

# DOWNLOAD TO PDF

@login\_required(login\_url='loginPage')

def download\_client(request, id):

if request.method == "GET":

auth\_user = request.user.first\_name

client = Clients.objects.get(pk=id)

now = datetime.datetime.now

context = {"auth\_user": auth\_user,

"page": f"EdifyLabs | download - {client.first\_name}-{client.last\_name}-{client.id}", "clients": client, "DT": now}

return render(request, "clients\_template/download\_client.html", context)

# DOWNLOAD TO EXCEL

def download\_clients\_excel(request):

clients = Clients.objects.all()

workbook = openpyxl.Workbook()

sheet = workbook.active

header\_row = ["ID", "First Name", "Last Name", "Email", "Gender", "DOB" , "Address", "Contact" , "Company" ,"Course" ]

sheet.append(header\_row)

for client in clients:

data\_row = [f"CL-{client.id}", client.first\_name, client.last\_name, client.email\_id, client.gender, client.date\_of\_birth,client.address , client.contact , client.company\_name, client.coaching\_type\_course.title]

sheet.append(data\_row)

response = HttpResponse(content\_type='application/vnd.openxmlformats-officedocument.spreadsheetml.sheet')

response['Content-Disposition'] = f'attachment; filename=clients\_{datetime.datetime.now().strftime("%Y%m%d%H%M%S")}.xlsx'

workbook.save(response)

return response

"""

>> LOGIC FUNCTIONS FOR INVOICE GENERATION

---------------------------------------------------------------------

"""

from django.shortcuts import redirect, render

from .models import Invoice

from .forms import InvoiceForm

from django.contrib.auth.decorators import login\_required

from num2words import num2words

from django.db.models import Q

import datetime

import openpyxl

from django.http import HttpResponse

# INVOICE DASHBOARD - DISPLAY FORM

@login\_required(login\_url='loginPage')

def invoice\_dashboard(request):

user = request.user.first\_name

form = InvoiceForm()

all\_invoice = Invoice.objects.all()

highest\_invoice = Invoice.objects.order\_by('-invoice\_number').first()

print(highest\_invoice)

context = {"page": "INVOICE | Edify Lab's Business Tool ","InvoiceForm" : form, "auth\_user" :user , "total\_invoice": len(all\_invoice)}

return render(request, "invoice\_template/invoice\_dashboard.html" , context)

# DISPLAYS ALL THE INVOICES FROM THE DB

@login\_required(login\_url='loginPage')

def invoice\_display(request):

all\_invoice = Invoice.objects.all()

print(all\_invoice)

if len(all\_invoice) != 0:

context\_invoice\_list = { "page": "ALL INVOICE | Edify Lab's Business Tool ",

"invoice\_display": all\_invoice, "total\_invoice": len(all\_invoice)}

else:

error = "NO INVOICE - 0"

context\_invoice\_list = {"error": error , "page": " ALL INVOICE | Edify Lab's Business Tool "}

return render(request, "invoice\_template/invoice\_display.html/", context\_invoice\_list)

# ADD A NEW INVOICE TO THE DB FROM DASHBOARD

@login\_required(login\_url='loginPage')

def invoice\_add(request):

if request.method == 'POST':

form = InvoiceForm(request.POST)

try:

if form.is\_valid():

form.save()

return redirect('invoice\_display')

except:

print('Form is not valid')

else:

form = InvoiceForm()

return render(request, 'invoice\_template/invoice\_dashboard.html', {'InvoiceForm': form})

# DELETE A SELECTED INVOICE

@login\_required(login\_url='loginPage')

def invoice\_item\_delete(request):

if request.method == 'POST':

item\_ids = request.POST.getlist('item\_ids\_invoice')

print(item\_ids , "seleceted")

if item\_ids:

Invoice.objects.filter(id\_\_in=item\_ids).delete()

return redirect('invoice\_display')

# GO TO DOWNLOAD PAGE INVOICE

@login\_required(login\_url='loginPage')

def invoice(request, id):

auth\_user = request.user.first\_name

invoice\_of\_id = Invoice.objects.get(pk=id)

payable\_amount = invoice\_of\_id.total\_amount\_payable\_invoice

payable\_amount\_text = num2words(payable\_amount, lang='en').title() + ' Only'

payable\_amount\_text = payable\_amount\_text.replace('-', ' ').replace(',', '')

now = datetime.datetime.now

context = {"auth\_user": auth\_user,

"title": f"EdifyLabs | invoice - {invoice\_of\_id.full\_name\_invoice}-{invoice\_of\_id.invoice\_number}", "invoice\_of\_id": invoice\_of\_id, "payable\_amount\_text": payable\_amount\_text}

return render(request, "invoice\_template/download\_invoice.html", context)

# SEARCH INVOICE

@login\_required(login\_url='loginPage')

def search\_invoice(request):

if request.method == "POST":

title = request.POST["title"]

lookups = Q(full\_name\_invoice\_\_icontains=title) | Q(

coaching\_type\_course\_invoice\_\_title\_\_icontains=title) | Q(invoice\_number\_\_icontains=title) | Q(total\_amount\_payable\_invoice\_\_icontains=title)

filter\_title = Invoice.objects.filter(lookups)

print(f" INVOICE {len(filter\_title)} {lookups}")

if title != '' and title is not None and len(filter\_title) != 0:

context\_invoice\_list = {

"invoice\_list\_all": Invoice.objects.all(), "filter\_title": filter\_title, "filter\_title\_total": len(filter\_title)}

else:

error = "NO MATCH FOUND"

context\_invoice\_list = {"error": error}

return render(request, "invoice\_template/invoice\_display.html/", context\_invoice\_list)

else:

print("---------------SOMETHING WENT WRONG IN SEARCH INVOICE-----------------------")

return redirect('/home/')

# DOWNLOAD TO EXCEL

def download\_invoice\_excel(request):

invoice\_all = Invoice.objects.all()

workbook = openpyxl.Workbook()

sheet = workbook.active

header\_row = ["ID", "Full Name", "Address", "GST no.", "Invoice number", "Course" , "Particular detail", "Amount" , "Discount" , "Total amount" , "CGST %" , "SGST %" , "CGST amount" ,"SGST amount" , "Total amount payable"]

sheet.append(header\_row)

for inv in invoice\_all:

data\_row = [f"INV-{inv.id}", inv.full\_name\_invoice, inv.address\_invoice, inv.GST\_invoice , inv.invoice\_number , str(inv.coaching\_type\_course\_invoice), inv.particular\_invoice, inv.amount\_invoice, inv.any\_discount\_invoice , inv.total\_amount\_invoice , inv.cgst , inv.sgst , inv.cgst\_amount , inv.sgst\_amount, inv.total\_amount\_payable\_invoice ]

sheet.append(data\_row)

response = HttpResponse(content\_type='application/vnd.openxmlformats-officedocument.spreadsheetml.sheet')

response['Content-Disposition'] = f'attachment; filename=invoice\_{datetime.datetime.now().strftime("%Y%m%d%H%M%S")}.xlsx'

workbook.save(response)

return response

**CHAPTER 5**

**Testing**

**5.1 Test Strategy**

1. Objective: The objective of the test strategy is to ensure the quality and reliability of the Edify Labs Business Tool by performing comprehensive testing across its modules. The strategy aims to identify and resolve any defects, validate the system's functionality, and verify that it meets the specified requirements.
2. Testing Approach: The testing approach will follow a combination of manual and automated testing techniques. Manual testing will be used for functional testing, user interface testing, and exploratory testing. Automated testing will be employed for regression testing, data-driven testing, and performance testing.
3. Test Levels: The testing will be conducted at the following levels: a. Unit Testing: Individual units or components will be tested in isolation to ensure their correctness. b. Integration Testing: The integration of various modules will be tested to ensure their proper functioning together. c. System Testing: The entire system will be tested to validate its compliance with functional and non-functional requirements. d. Acceptance Testing: The system will be tested by end-users or stakeholders to ensure it meets their expectations.
4. Test Types: The following test types will be performed: a. Functional Testing: Validate the functionality of each module, including HR authentication, HR dashboard, scheduling, filtration, data visualization, report generation, invoice generation, and data downloading. b. User Interface Testing: Verify the user interface elements such as forms, buttons, menus, and navigation for usability and consistency. c. Performance Testing: Measure the system's performance under expected and peak loads to ensure it can handle the required user interactions. d. Security Testing: Validate the system's security measures, including user authentication, data encryption, and protection against common vulnerabilities. e. Regression Testing: Re-test previously implemented functionality after any changes or enhancements to ensure that existing features are not negatively affected. f. Compatibility Testing: Verify the compatibility of the system with different browsers, operating systems, and devices. g. Usability Testing: Evaluate the system's ease of use, intuitiveness, and overall user experience. h. Integration Testing: Test the integration of different modules and components to ensure proper data flow and functionality.
5. Test Data: Test data will be carefully selected to cover a wide range of scenarios, including valid and invalid inputs, boundary values, and edge cases. Realistic and representative data will be used to ensure the accuracy and effectiveness of the tests.
6. Test Environment: A dedicated test environment will be set up that closely resembles the production environment. This environment will include the necessary hardware, software, databases, and network configurations to support testing activities.
7. Test Deliverables: The following deliverables will be produced as part of the testing process: a. Test Plan: A comprehensive document outlining the test strategy, objectives, approach, scope, and schedule. b. Test Cases: Detailed test cases that cover all functional and non-functional requirements. c. Test Data: Well-defined test data sets used for testing different scenarios. d. Test Logs: Logs documenting the execution of test cases, including test results, issues encountered, and any additional observations or notes. e. Defect Reports: Reports documenting any defects found during testing, including their severity, priority, and steps to reproduce.
8. Test Schedule: A detailed test schedule will be created, considering the project timeline, development milestones, and resource availability. The schedule will allocate sufficient time for each testing phase, including test planning, test case development, test execution, and defect management.
9. Risk and Issue Management: A risk assessment will be conducted to identify potential risks and issues that may impact the testing process. Mitigation strategies will be defined to minimize the impact of these risks, and regular communication channels will be established to report and address any issues that arise during testing.
10. Test Execution and Reporting: Test execution will be carried out according to the defined test plan. Test progress and results will be regularly reported to the project stakeholders. Test logs, defect reports, and other relevant documentation will be maintained and shared with the development team for issue resolution.
11. Test Completion Criteria: The following criteria will be used to determine when testing is considered complete: a. All planned test cases have been executed and passed. b. High-priority defects have been resolved or mitigated. c. Acceptance criteria defined by stakeholders have been met. d. Test coverage for critical functionality is satisfactory. e. Test exit criteria, as defined in the test plan, have been fulfilled.

**5.2 Test Case/Test Script**

**#Test code for Login Authentication**

from django.test import TestCase, Client

from django.urls import reverse

from django.contrib.auth.models import User

class LoginPageTestCase(TestCase):

    def setUp(self):

        self.client = Client()

        self.signup\_url = reverse('signup')

        self.login\_url = reverse('loginPage')

        self.logout\_url = reverse('logoutPage')

        self.ourstory\_url = reverse('ourstory')

        self.user = User.objects.create\_user(username='testuser', password='testpassword')

    def test\_signup(self):

        response = self.client.get(self.signup\_url)

        self.assertEqual(response.status\_code, 200)

        data = {

            'name': 'testuser',

            'email': 'testuser@example.com',

            'pwd': 'testpassword',

            'pwdfinal': 'testpassword'

        }

        response = self.client.post(self.signup\_url, data)

        self.assertEqual(response.status\_code, 200)

        self.assertContains(response, 'LOGIN')

    def test\_login\_page(self):

        response = self.client.get(self.login\_url)

        self.assertEqual(response.status\_code, 200)

        data = {

            'login\_username': 'testuser',

            'login\_pass': 'testpassword'

        }

        response = self.client.post(self.login\_url, data)

        self.assertEqual(response.status\_code, 302)

        self.assertRedirects(response, reverse('home'))

        # Attempt login with incorrect credentials

        data = {

            'login\_username': 'testuser',

            'login\_pass': 'incorrectpassword'

        }

        response = self.client.post(self.login\_url, data)

        self.assertEqual(response.status\_code, 200)

        self.assertContains(response, 'Error\_message\_unauthorized')

    def test\_logout\_page(self):

        self.client.login(username='testuser', password='testpassword')

        response = self.client.get(self.logout\_url)

        self.assertEqual(response.status\_code, 302)

        self.assertRedirects(response, reverse('loginPage'))

    def test\_ourstory\_page(self):

        response = self.client.get(self.ourstory\_url)

        self.assertEqual(response.status\_code, 200)

    def test\_authenticated\_home\_page(self):

        self.client.login(username='testuser', password='testpassword')

        response = self.client.get(reverse('home'))

        self.assertEqual(response.status\_code, 200)

        self.assertContains(response, 'OkNoted | HOME')

    def test\_unauthenticated\_home\_page(self):

        response = self.client.get(reverse('home'))

        self.assertEqual(response.status\_code, 302)

        self.assertRedirects(response, reverse('loginPage'))

class SignupPageTestCase(TestCase):

    def setUp(self):

        self.client = Client()

        self.signup\_url = reverse('signup')

    def test\_signup\_page(self):

        response = self.client.get(self.signup\_url)

        self.assertEqual(response.status\_code, 200)

**#Test code for Client’s task**

from django.test import TestCase

from django.urls import reverse

from clientsapp.models import Clients

class ClientsAppTestCase(TestCase):

    def setUp(self):

        self.client1 = Clients.objects.create(

            first\_name='Tanmay',

            last\_name='Barvi',

            email\_id='Tanmay.Barvi@example.com',

            gender='M',

            date\_of\_birth='2000-08-13',

            address='xyz test',

            contact='9325949697',

            company\_name='ABC Company',

            coaching\_type\_course='Personal coaching'

        )

    def test\_home\_view(self):

        response = self.client.get(reverse('home'))

        self.assertEqual(response.status\_code, 200)

        self.assertTemplateUsed(response, 'home.html')

    def test\_clients\_list\_view(self):

        response = self.client.get(reverse('clients\_list'))

        self.assertEqual(response.status\_code, 200)

        self.assertTemplateUsed(response, 'clients\_template/clients\_list.html/')

    def test\_item\_delete\_view(self):

        # Prepare data for testing deletion

        item\_ids = [self.client1.id]

        initial\_count = Clients.objects.count()

        response = self.client.post(reverse('item\_delete'), {'item\_ids': item\_ids})

        self.assertEqual(response.status\_code, 302)  # Redirect status code

        self.assertEqual(Clients.objects.count(), initial\_count - 1)

    def test\_search\_client\_view(self):

        # Prepare data for testing search

        search\_term = 'Tanmay'

        response = self.client.post(reverse('search\_client'), {'title': search\_term})

        self.assertEqual(response.status\_code, 200)

        self.assertTemplateUsed(response, 'clients\_template/clients\_list.html/')

    def test\_download\_client\_view(self):

        client\_id = self.client1.id

        response = self.client.get(reverse('download\_client', args=[client\_id]))

        self.assertEqual(response.status\_code, 200)

        self.assertTemplateUsed(response, 'clients\_template/download\_client.html')

    def test\_download\_clients\_excel\_view(self):

        response = self.client.get(reverse('download\_clients\_excel'))

        self.assertEqual(response.status\_code, 200)

        self.assertEqual(response['Content-Type'], 'application/vnd.openxmlformats-officedocument.spreadsheetml.sheet')

**5.3 Defect Report/Test Log**

========================================

Test Log

========================================

Date: 2023-06-24

Time: 09:30 AM

Environment: Production

----------------------------------------

Test Case 1: Signup - Successful

----------------------------------------

Result: Pass

Duration: 0.5s

User successfully signed up with valid credentials.

----------------------------------------

Test Case 2: Signup - Error (Missing fields)

----------------------------------------

Result: Pass

Duration: 0.3s

Validation error triggered for missing required fields.

----------------------------------------

Test Case 3: Signup - Error (Username already taken)

----------------------------------------

Result: Pass

Duration: 0.4s

Validation error triggered for an already taken username.

----------------------------------------

Test Case 4: Login - Successful

----------------------------------------

Result: Pass

Duration: 0.6s

User successfully logged in with valid credentials.

----------------------------------------

Test Case 5: Login - Error (Invalid credentials)

----------------------------------------

Result: Pass

Duration: 0.5s

Authentication error triggered for invalid credentials.

----------------------------------------

Test Case 6: Logout

----------------------------------------

Result: Pass

Duration: 0.2s

User successfully logged out.

========================================

Test Summary

========================================

Total Test Cases: 6

Passed: 6

Failed: 0

Errors: 0

Skipped: 0

Total Duration: 3.2s

All tests passed without any errors or failures.

**CHAPTER 6**

**Limitations of**

**Proposed System**

1. Reliance on internet connectivity, as it is a web-based application. If a user is unable to connect to the internet, they may not be able to access the system or perform certain tasks.

2. Integration with other software systems: Edify Labs may not have seamless integration capabilities with other HR or business software systems, which could limit its overall usefulness for companies that rely on multiple tools.

3. While Edify Labs Business Tool aims to provide a more personalized and tailored HR experience, it may not be suitable for larger organizations with complex HR needs, as the system is designed with a focus on smaller-scale operations.

4. Learning curve for new users: The tool may have a learning curve for users who are unfamiliar with the system or have limited technical expertise. Adequate training and documentation should be provided to help users effectively navigate and utilize the tool's features.

**CHAPTER 7**

**Proposed Enhancements**

1. Mobile application: Currently, Edify Labs Business Tool is a web-based application. A mobile application could be developed to allow HR professionals to manage client data on the go.

2. Advanced analytics: While Edify Labs Business Tool offers data visualization and reporting features, incorporating advanced analytics tools like predictive analytics and machine learning could help HR professionals identify trends and patterns in employee data, enabling them to make more informed decisions.

3. Integration with video conferencing: The system could be enhanced to allow for video conferencing with employees, providing a more personalized and interactive experience.

4. Customization: The system could be enhanced to allow for greater customization and personalization, providing HR pr professionals.

5. Employee engagement and feedback: Implement features to measure and enhance client engagement, such as surveys, feedback mechanisms, and recognition programs. This can help HR professionals understand client’s satisfaction and take proactive steps to improve engagement and overall process.

**CHAPTER 8**

**Conclusion**

In conclusion, the Edify Labs project has been a significant endeavor aimed at developing a comprehensive and user-friendly business tool for HR professionals. Throughout the project, I have successfully created a web-based application that simplifies and streamlines various HR processes, including client management, data analysis, and reporting.

The Edify Labs Business Tool offers numerous benefits, such as centralized data management, intuitive user interfaces, and data visualization capabilities, enabling HR professionals to make informed decisions and optimize their HR operations. By automating routine tasks, the tool helps save time and reduce administrative burdens, allowing HR teams to focus on strategic initiatives and employee engagement.

During the development process, I implemented key features such as client management, search functionality, data export to Excel, and user authentication. I also ensured the security of sensitive information through password hashing and user authentication mechanisms.

While the project has achieved significant milestones, there are opportunities for future enhancements. These include the development of a mobile application, incorporation of advanced analytics, integration with video conferencing, and customization options to cater to diverse HR requirements.

I am confident that the Edify Labs Business Tool will empower HR professionals, streamline their workflows, and contribute to the success of organizations in managing their human resources effectively.

**CHAPTER 9**

**Bibliography**

Django Documentation:

Official Django documentation: https://docs.djangoproject.com/

Python Documentation:

Official Python documentation: <https://docs.python.org/>

Web Development Resources:

Mozilla Developer Network (MDN) Web Docs: https://developer.mozilla.org/en-US/docs/Web

W3Schools: https://www.w3schools.com/

Stack Overflow: <https://stackoverflow.com/>

Books:

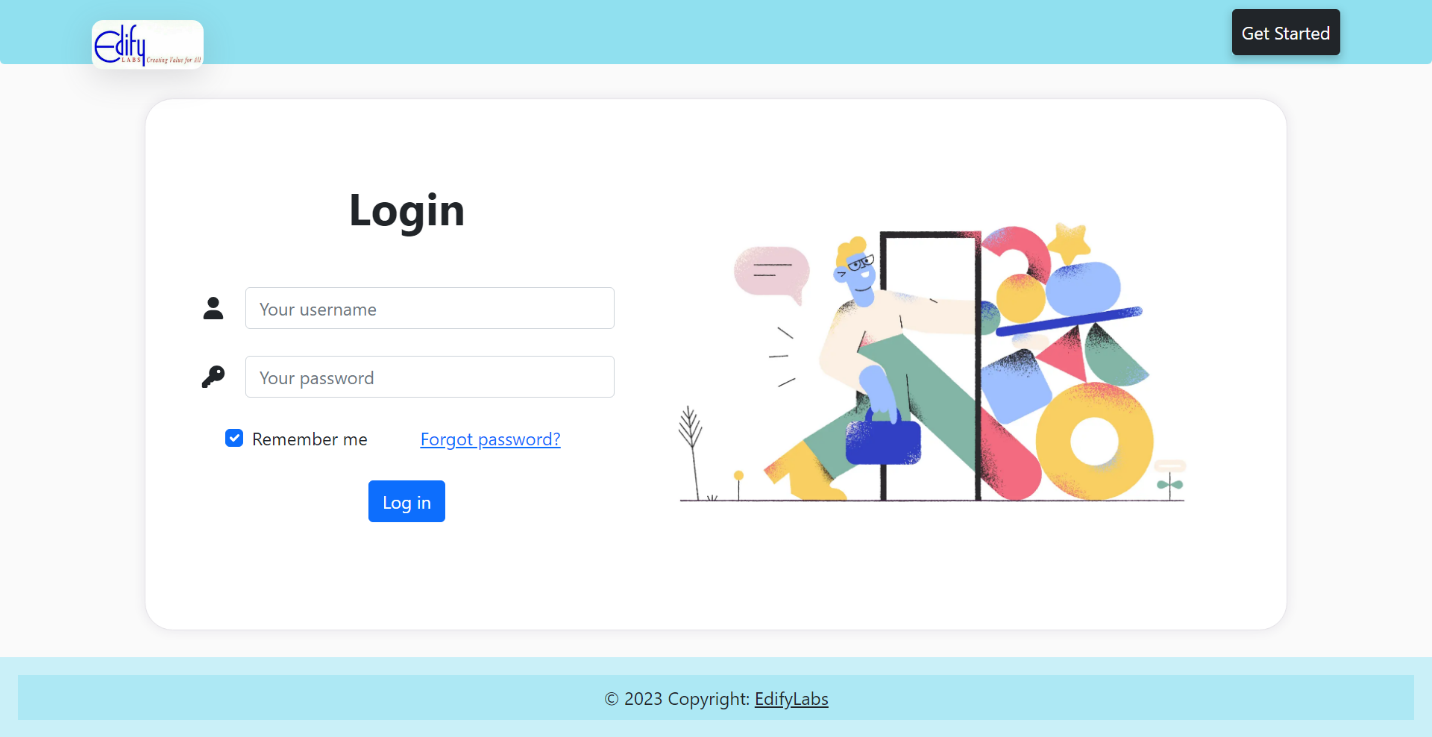
"Django for Beginners" by William S. Vincent

Online Tutorials and Courses:

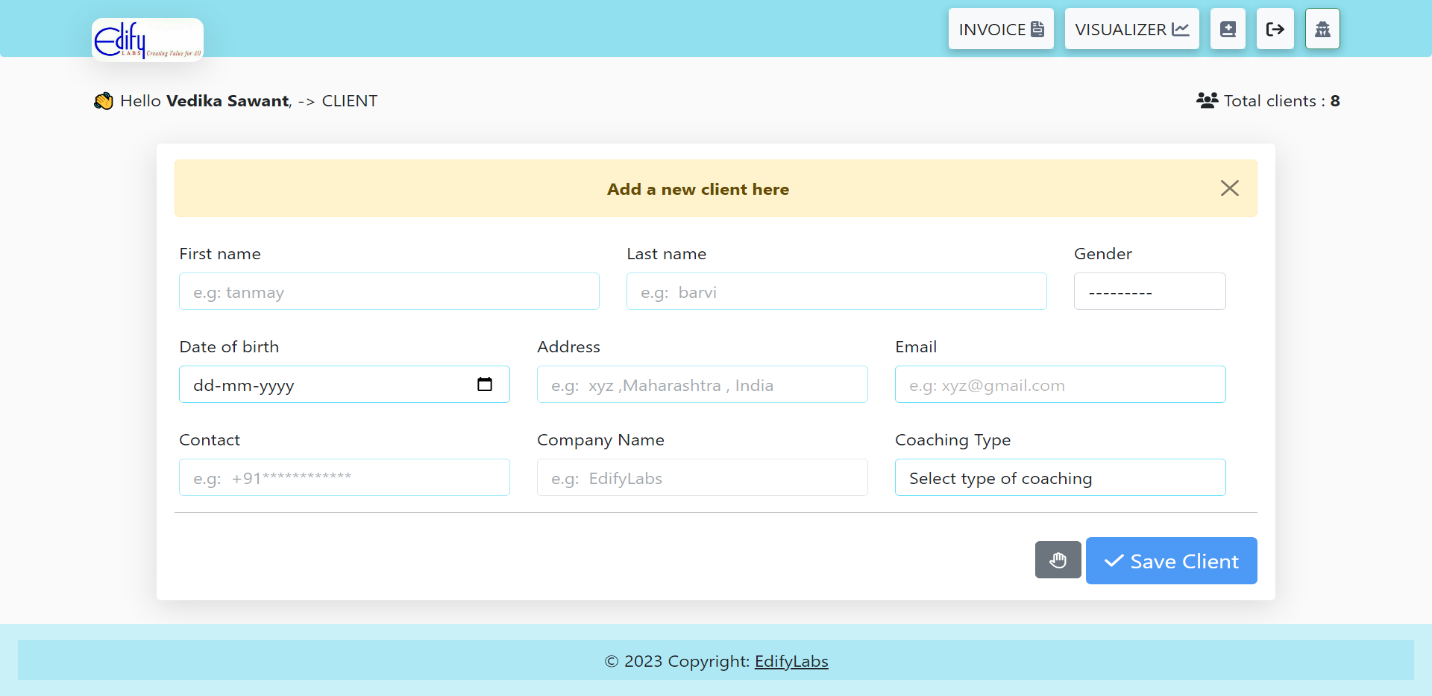
Django for Beginners: https://djangoforbeginners.com/

**CHAPTER 10**

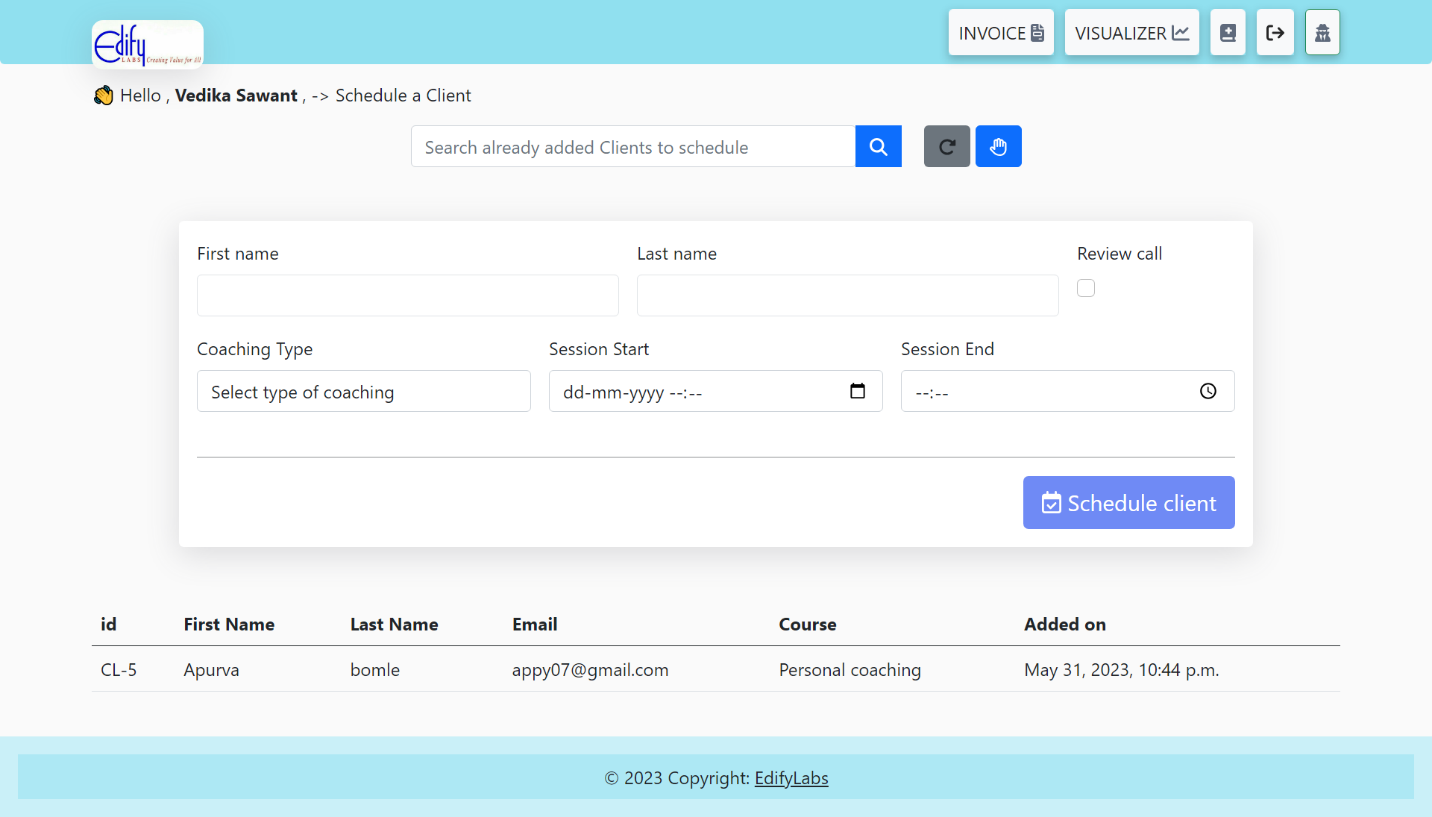
**User Manual**

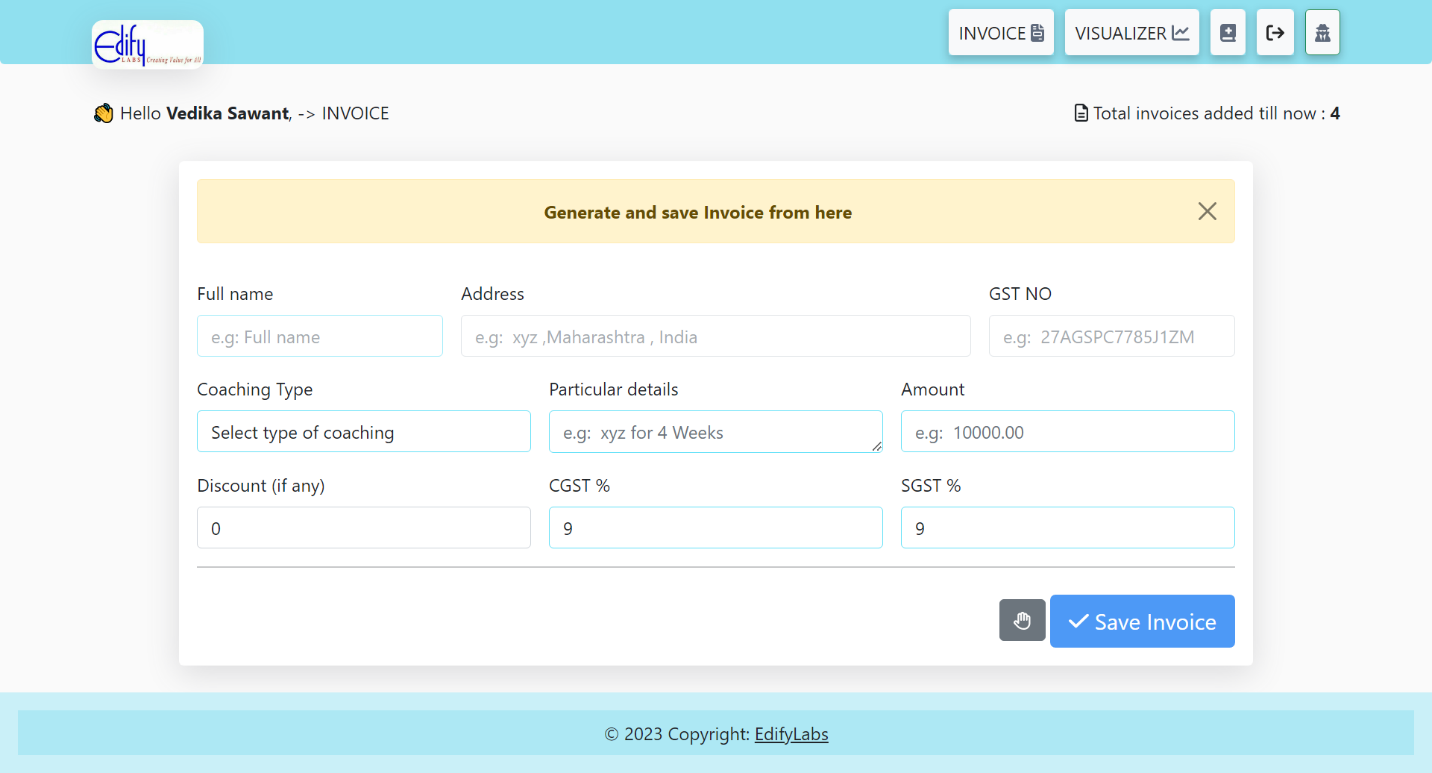
**Login Page** - From here HR can login

**HR Dashboard** – Here new clients can be added and redirected to various pages

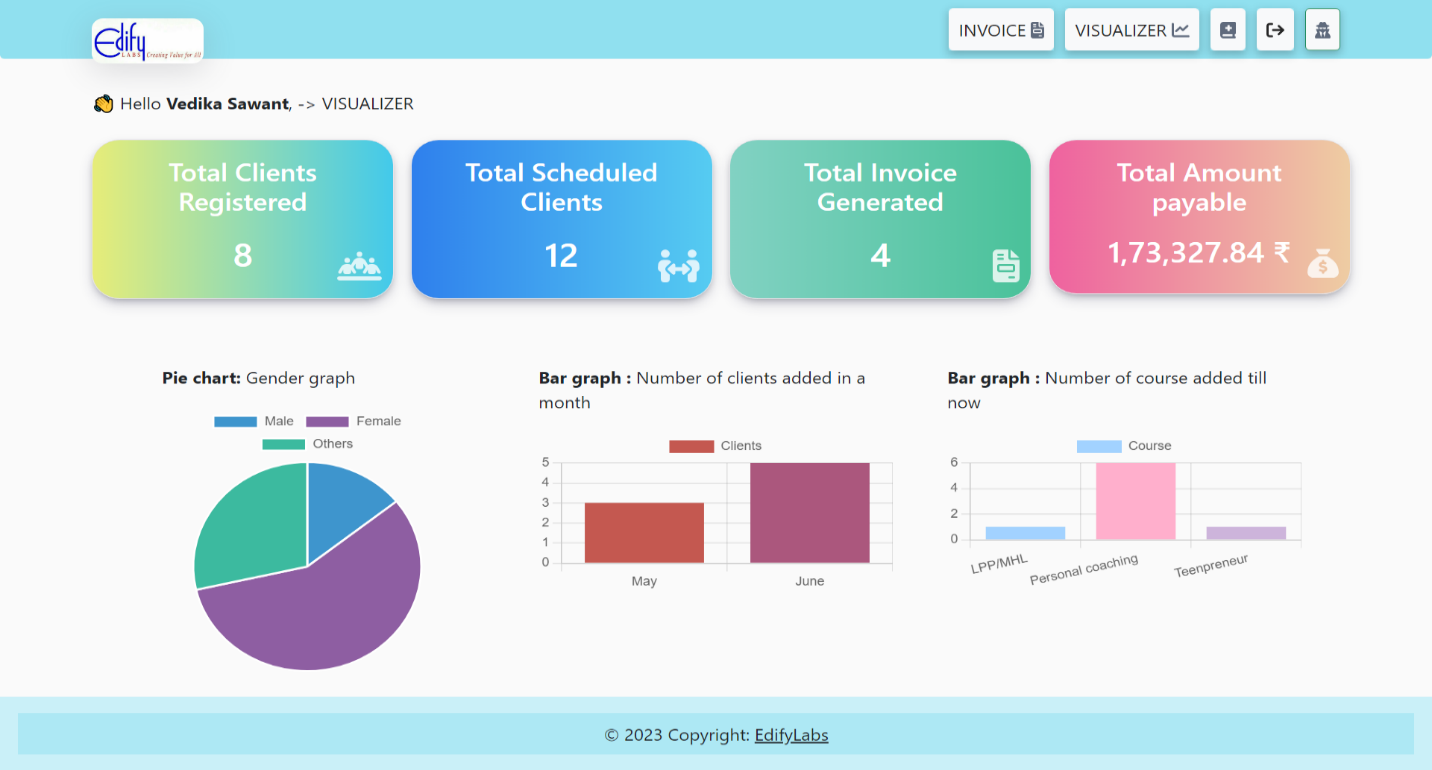
****

**Schedule page** – Here HR Schedules **a** call or a meeting if a client is already added

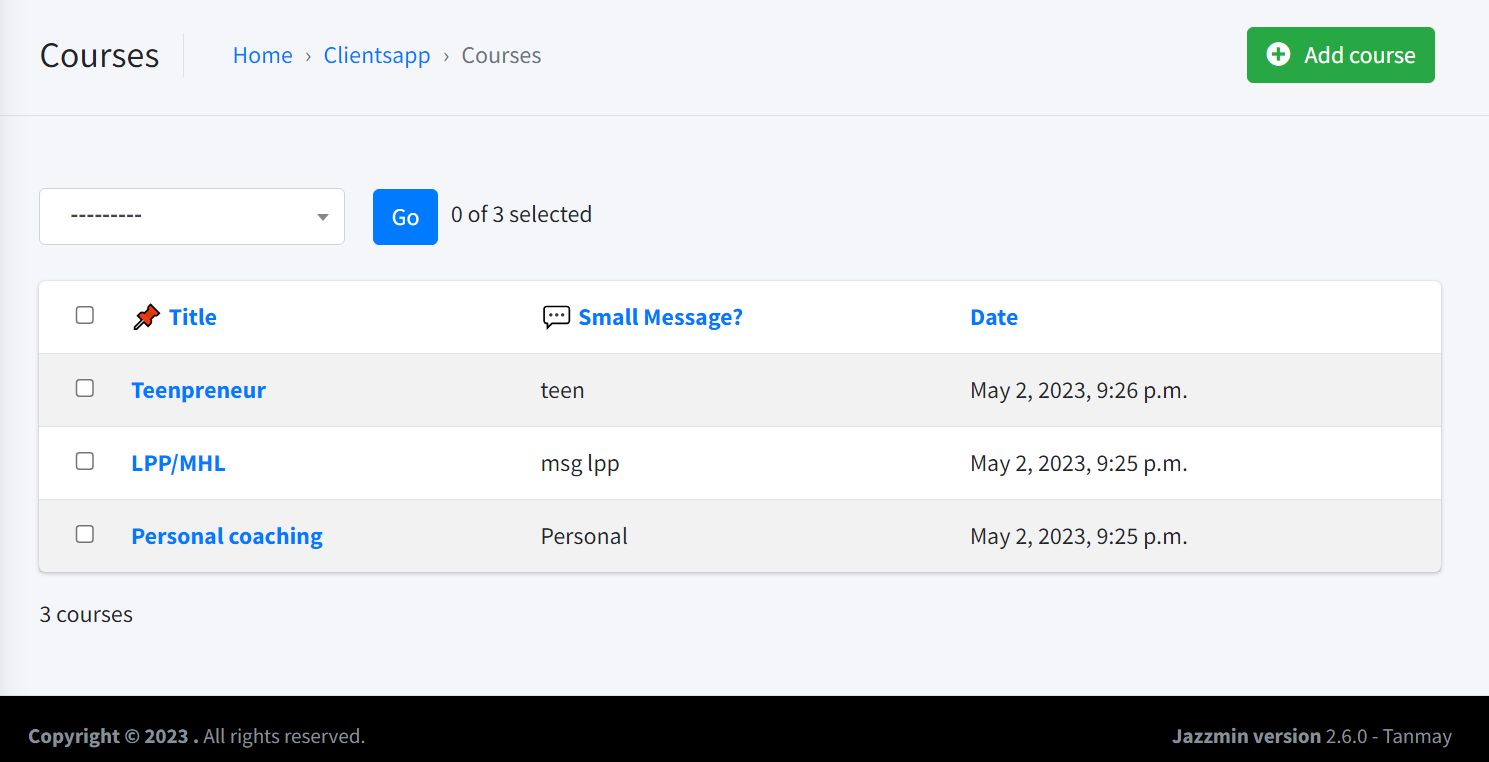
****

**Invoice page** – Here HR generates invoices

**Visualization page** – Here HR can see the analysis

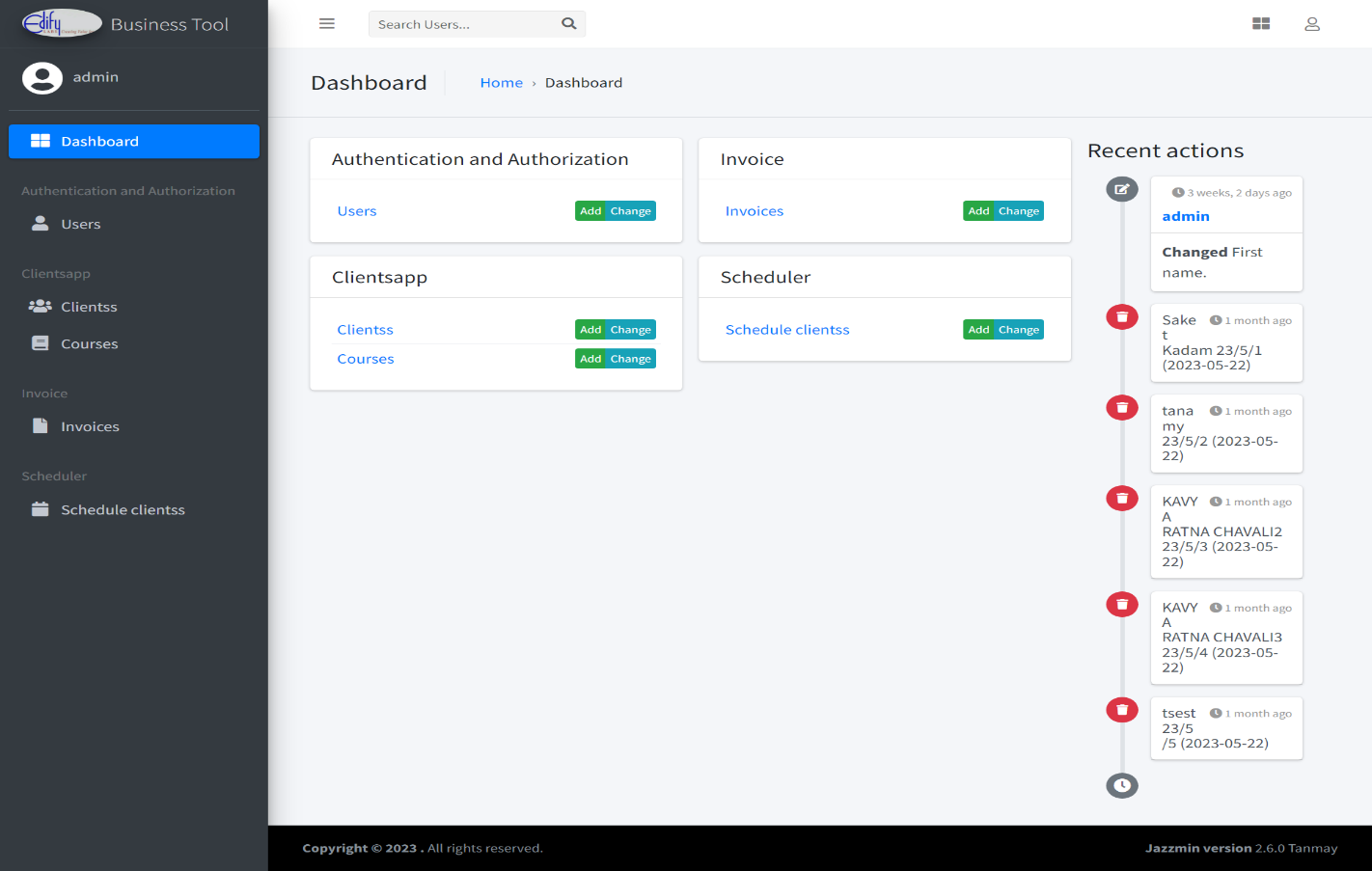


**Add course** – Here HR can add new courses

****

**Visualization page** – A screenshot of a login form

Description automatically generatedAdmin login page

**Admins page** – Here Admin manages all the system