**UNIVERSITY OF TURKISH AERONAUTICAL ASSOCIATION**

**ENGINEERING FACULTY**

**DEPARTMENT OF COMPUTER ENGINEERING**

amblem, simge, sembol, logo, ticari marka içeren bir resim

Açıklama otomatik olarak oluşturuldu

**CENG 301 LAB Project**

**Project Title:**

Human Resources Management Application

**Project Member(s):**

Fevzi FİDAN 210444056

Hikmet ÇATAK 210444017

Ömer Faruk AZİLİ 210444026

**Project Advisor**

Ahmet Serkan KARATAŞ, Ayşe Beyza ÜNAL

**01.01.2025**

Table of Content

* 1. **Project Introduction  
     1.1 Project Overview**

Briefly describe the purpose and objectives of the database project.

The aim of our project is to create a human resources program. In this program, it is to keep records of employees for human resources and to ensure communication between employees and human resources. For this purpose, it is necessary to create a communication platform and record the necessary information and requests. This is exactly the aim of our project.

**1.2 Project Team**

**Roles and Responsibilities:**

Fevzi FİDAN: Responsible for creating the user interface and ensuring its usability. Designing the general structure and design of the Software. Deciding what the features of the project would be and ensuring that these features operate efficiently.

Hikmet Çatak: Responsible for general designing the database and writing codes to allow access to the database through the user interface. Decide coding structure for features part. Helping to decide what the features of the project would be and writing these features.

Ömer Faruk Azili: Responsible for designing the database and writing codes to allow access to the database through the user interface. To fix problems that may arise for the future and prepare future additions. To make the overall project structure better. Writing and deciding the features.

**Project Scope**

Clearly define the scope of the database project, including its intended functionalities and limitations.

I mentioned the purpose of our project above.The features we have determined for this project are sending e-mail, registering, changing passwords, creating permission-special requests, evaluation of permission-special requests and creating events-meetings. Also, you can hire or fire people and assign items to those people. As, expected, it displays information about all these things except passwords. We have divided all of these features into features that users and administrators can access. Employees can only create requests, send messages, change passwords, and see information about themselves. Managers can perform all the features mentioned except creating requests. This includes approving or rejecting incoming requests.

* 1. **Database Design  
     2.1 Introduction**

Provide an introduction to the database design, highlighting its purpose and significance in the context of the project.

For this human resources program, we have 9 tables and most of them have different primary keys. We needed to easily record large amounts of data and to search through them effectively, we decided to design with a small amount of joins. Of course, we didn't hesitate to create tables for this purpose. As you can see, all of our tables are related to the employess table, for this we use most of the time foreign key. Usually, one of our main goals is to have just the employee id to perform a sub-action, but as a result, you may need to use a different function to get some different information. However, since we designed the UI and queries efficiently, this was not a problem. In addition, the main table structure you see allows us to add other tables.

**2.2 Entity-Relationship Diagram (ERD)(** **Larger image on slide)**

metin, ekran görüntüsü, paralel, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. **Data Dictionary**

**Table:employees****:** Represents the employees in the company.  
  
**Columns:**

|  |  |
| --- | --- |
| **employee\_id** | int AI PK |
| first\_name | varchar(50) |
| last\_name | varchar(50) |
| date\_of\_birth | date |
| gender | varchar(50) |
| job\_title | varchar(50) |
| department | varchar(25) |
| salary | decimal(8,2) |
| hire\_date | date |
| email | varchar(50) |
| phone\_number | varchar(50) |
| password | varchar(50) |
| is\_active | tinyint(1) |

**Table:email: :** Represents the email records that sent.  
  
**Columns:**

|  |  |
| --- | --- |
| **id** | int AI PK |
| email\_title | varchar (255) |
| email\_description | text |
| **from\_emp\_id** | int |
| **to\_emp\_id** | int |

**-****An employee can send and get multiple email (many to many)**

**-From\_emp\_id, to\_****emp\_id** in the **email** is a **foreign key** that references the **employee\_id** column in the **employees** table.

**Table:employee\_items:** Represents the assigned items to employees.  
  
**Columns:**

|  |  |
| --- | --- |
| **employee\_id** | int |
| **item\_id** | int |
| assignment\_date | datetime |
| quantity | int |
| **assign\_id** | int AI PK |

**-** **More than one item can be assigned to more than one employee based on quantity. (many to many)**

**-employee\_id** in the **employee\_items** is a **foreign key** that references the **employee\_id** column in the **employees** table.

**Table:employee\_leaves:-Represent leave requests of employees**  
  
**Columns:**

|  |  |
| --- | --- |
| **leave\_request\_id** | int AI PK |
| **employee\_id** | Int |
| status\_of\_request | enum('Pending','Accepted','Rejected') |
| request\_date | Date |
| approved\_by | Int |
| answer\_date | Date |
| leave\_type | enum('Annual Leave','Health Leave','Unpaid Leave','Excuse Leave') |
| Start\_date | Date |
| end\_date | Date |
| total\_dates | Int |
| desc\_request | Text |
| created\_at | Timestamp |

-**An employee can create multiple leave requests(One to Many)**

**-****employee\_id** in the **employee\_leaves** is a **foreign key** that references the **employee\_id** column in the **employees** table.

**Table:events\_:Represent the events.**  
  
**Columns:**

|  |  |
| --- | --- |
| **id** | int AI PK |
| event\_name | varchar(255) |
| event\_text | text |
| event\_date | datetime |

**Table:items:Represent the items in company.**  
  
**Columns:**

|  |  |
| --- | --- |
| **id** | int AI PK |
| item\_name | varchar(20) |
| quantity | int |

**Table:messages:Represent the messages informations.**  
  
**Columns:**

|  |  |
| --- | --- |
| **id** | int AI PK |
| **from\_emp\_id** | int |
| **to\_emp\_id** | int |
| message\_text | text |
| is\_read | tinyint(1) |
| message\_date | datetime |
| subject | varchar(255) |

- **An employee can send and get multiple messages (many to many)**

**-****From\_emp\_id,to\_emp\_id** in the **messages** is a **foreign key** that references the **employee\_id** column in the **employees** table.

**Table:pending\_email:Represent the emails that will be send .**  
  
**Columns:**

|  |  |
| --- | --- |
| **id** | int AI PK |
| email\_title | varchar(255) |
| email\_description | text |
| **from\_emp\_id** | int |
| **to\_emp\_id** | int |
| send\_date | datetime |
| is\_sent | tinyint(1) |

**-From\_emp\_id,to\_emp\_id** in the **pending\_email** is a **foreign key** that references the **employee\_id** column in the **employees** table.

**Table:special\_requests:Represent the special requests of employees**  
  
**Columns:**

|  |  |
| --- | --- |
| **request\_id** | int AI PK |
| **employee\_id** | Int |
| request\_type | enum('Advance','Salary Increase','Payback','Other') |
| request\_amount | decimal(10,2) |
| request\_date | Date |
| status\_of\_request | enum('Pending','Accepted','Rejected') |
| approved\_by | Int |
| description | Text |
| answer\_date | Date |
| created\_at | Timestamp |

**-An employee can create multiple requests.(One to Many)**

**employee\_id** in the **special\_requests** is a **foreign key** that references the **employee\_id** column in the **employees** table.

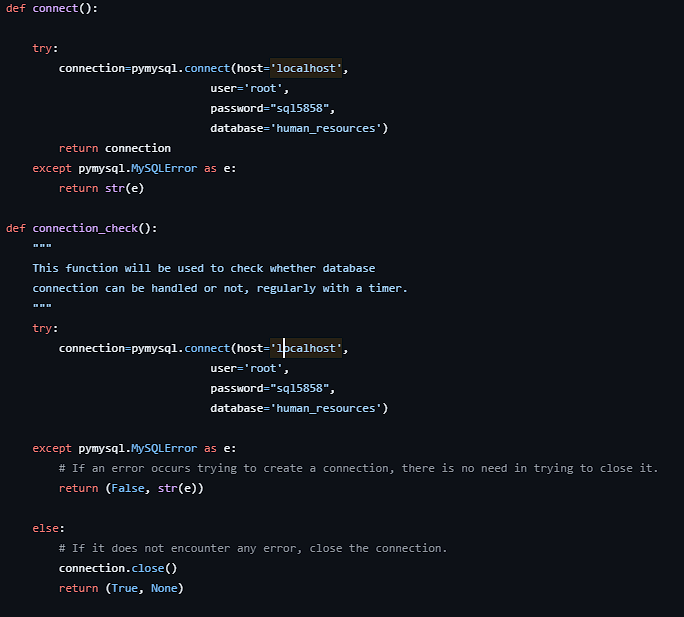
**View:employee\_items\_with\_names:** **Shows employees who have items assigned to them.**  
**Columns:**

|  |  |
| --- | --- |
| employee\_id | int |
| first\_name | varchar(50) |
| last\_name | varchar(50) |
| item\_id | int |
| item\_name | varchar(20) |
| quantity | int |
| assign\_id | int |
| assignment\_date | datetime |

* 1. **Database Implementation**  
     3.1 **Database Management System**

MySQL

* 1. **Database Connectivity**



We used pymysql library to connect database and implementing functions in app.

**Parameters:**

Host= Specifies the database server's hostname.

User= The username used to connect to the database.

Password= The password associated with the username **root** to authenticate the connection.

Database= Specifies the name of the database.

If the connection is successful, the connection object is returned. This object allows you to interact with the database (e.g., execute queries, fetch results).

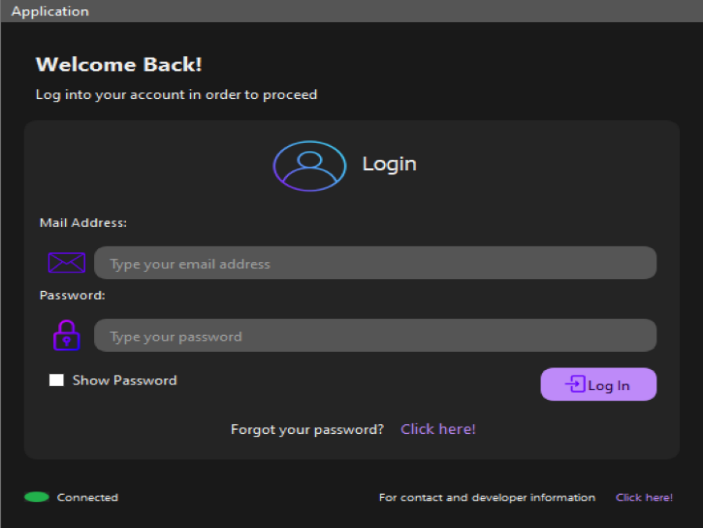
* 1. **Interface Design**  
     5.1 **User Interface Design**

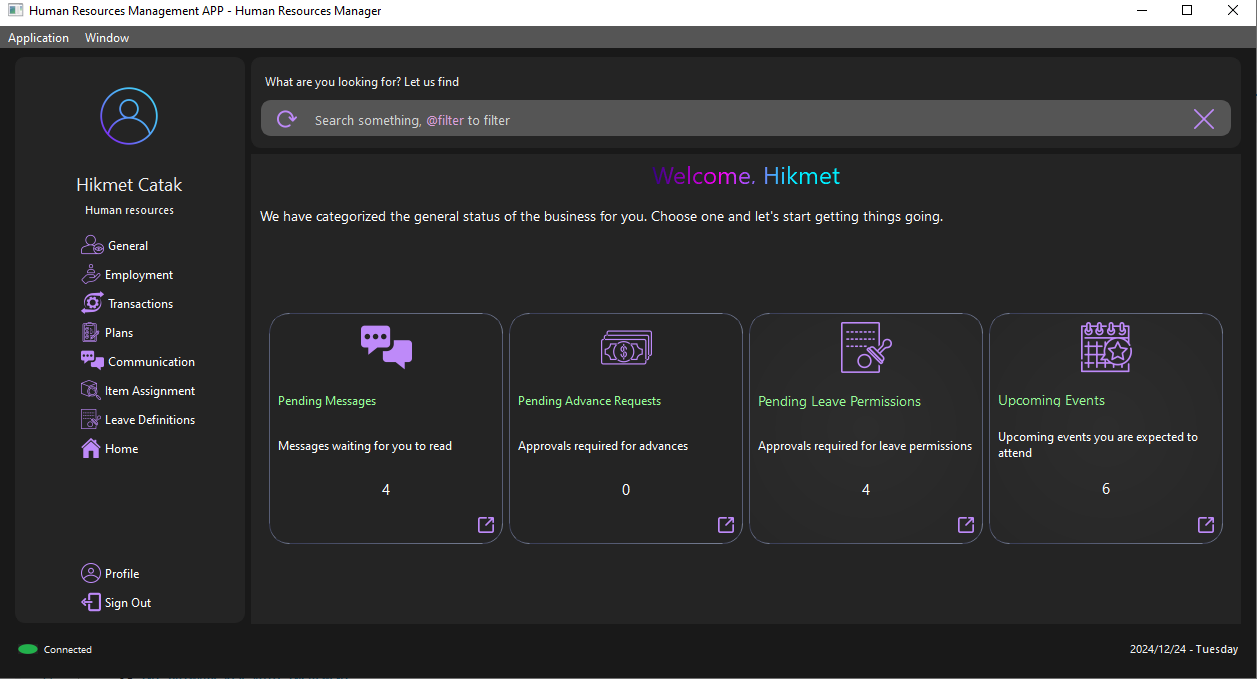
Specify the programming languages and tools used to create the interface (e.g., Java, C#, PHP, HTML, CSS, etc.).

The interface of the project has been developed in Python. We used PyQt6 GUI toolkit.

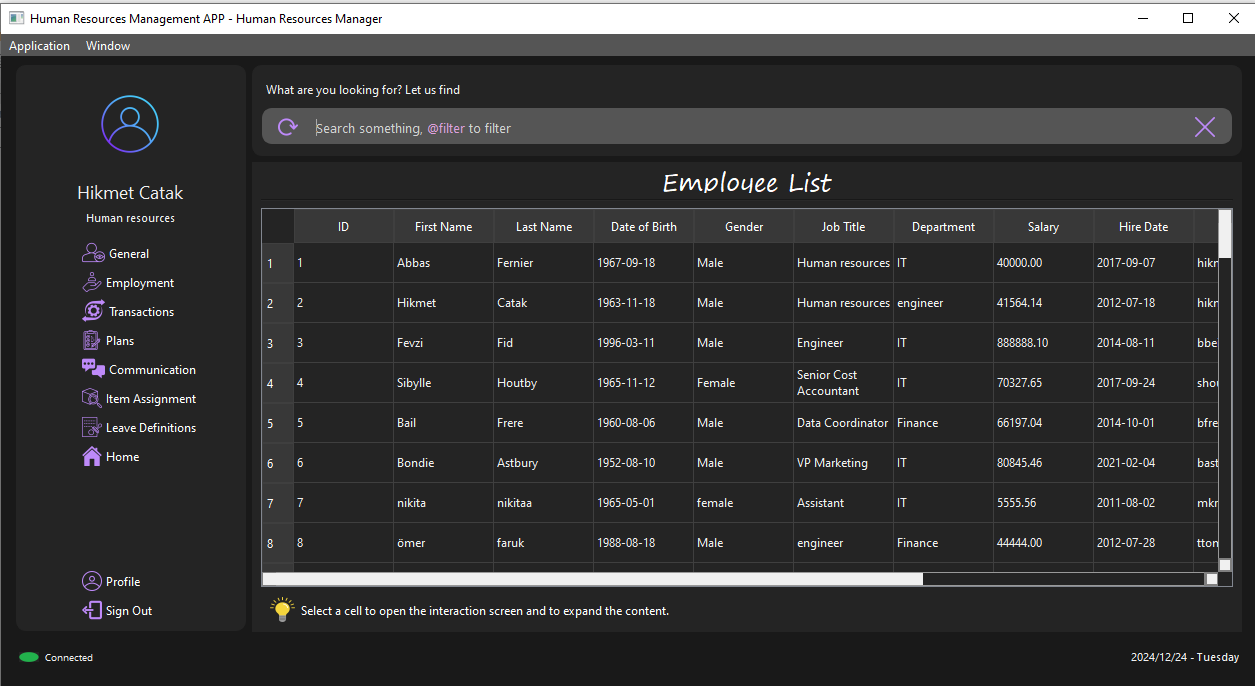
* 1. **Interface Overview**

Provide screenshots of each interface page, explaining its purpose and functionality. ( Examples: Login Page, Product Management Page (e.g., Add Product, Delete Product), Customer Information Page, Navigation and Buttons, ect…)



Login Page

Dashboard

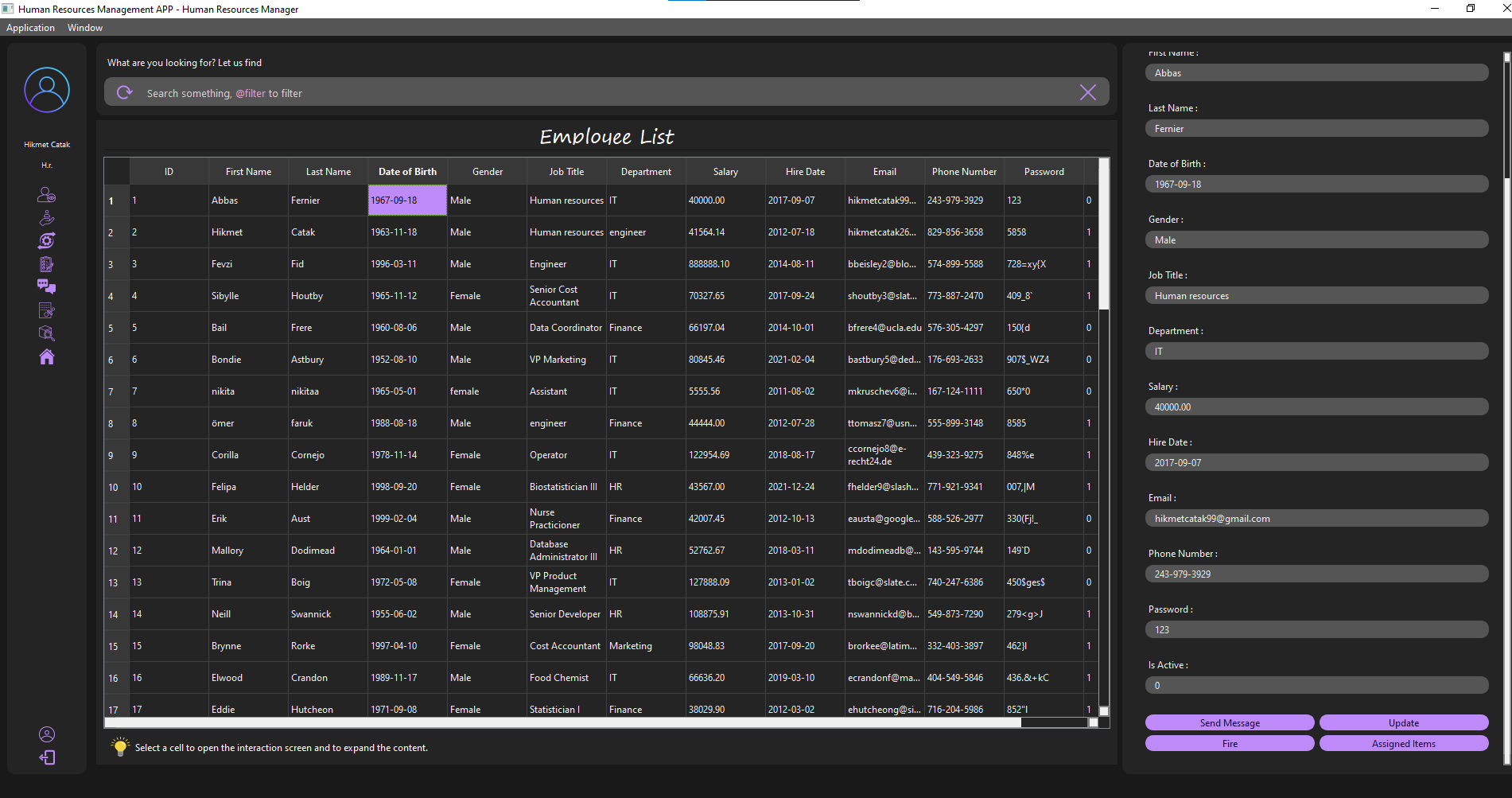


View Employees

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

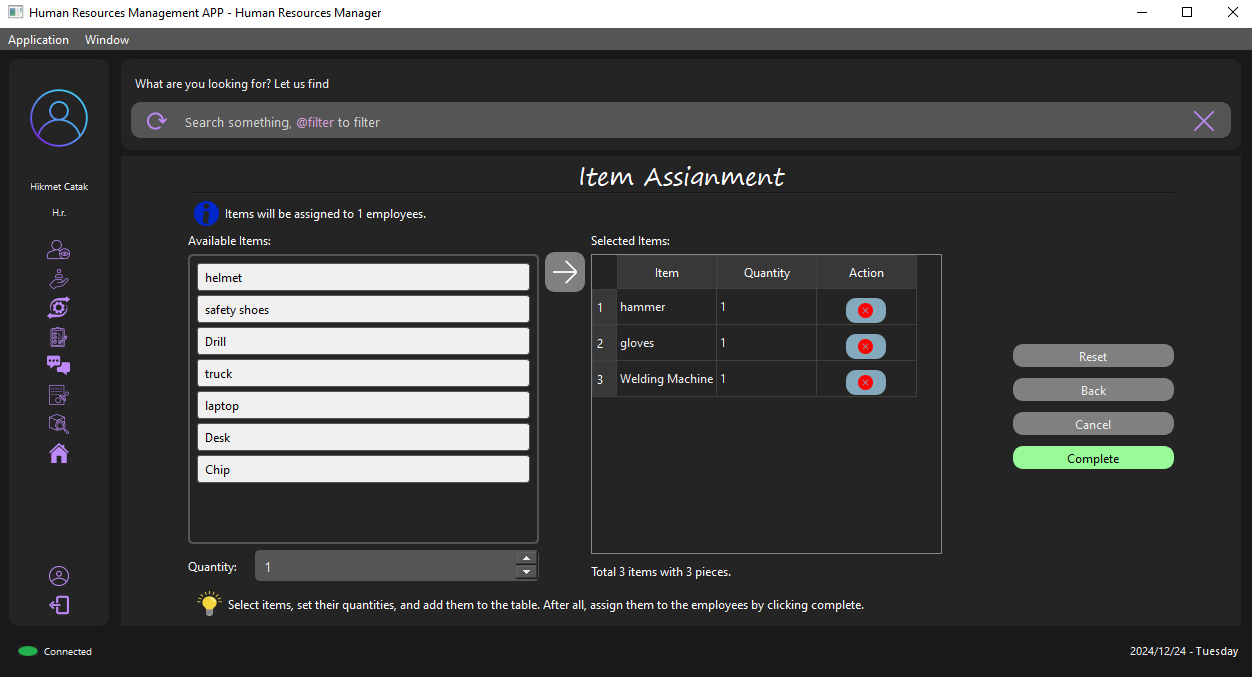
Hiring employee page



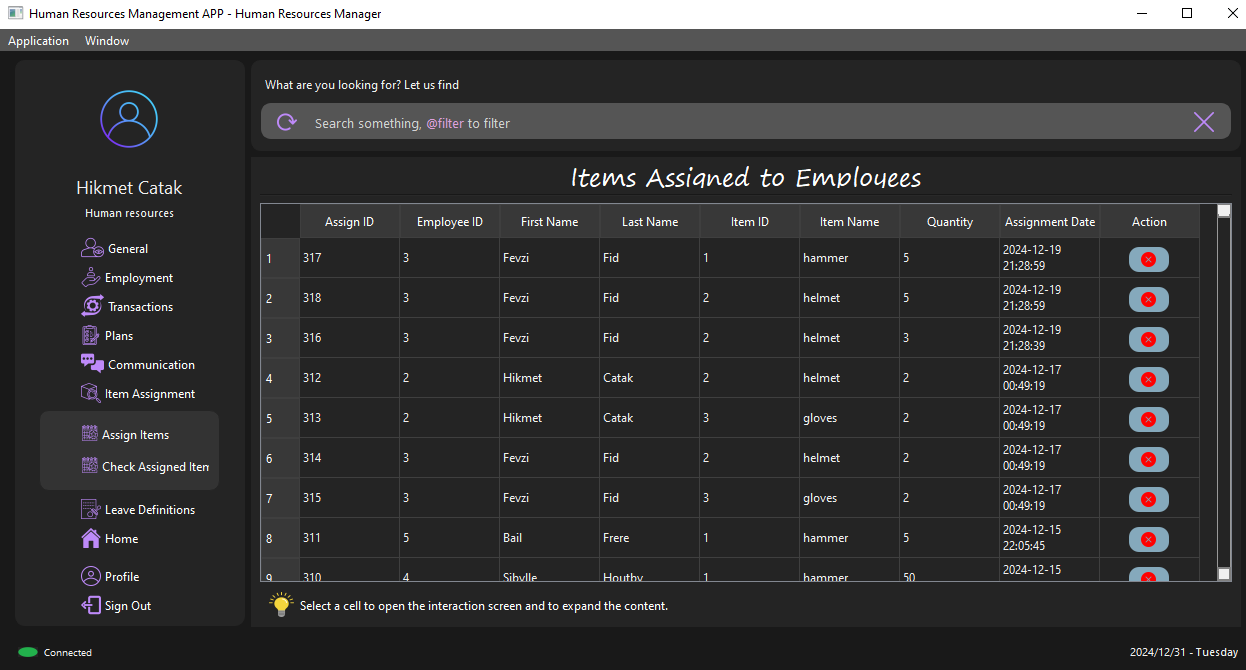
Update,fire,send message or see assigned items of employeemetin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Incoming messages page



Item assignment page=Can be assigned items to employees



Employee and assigned item page: It shows all records related to assigned items. It can be deleted assignment proces from actions .

ekran görüntüsü, yazılım, metin, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Update Salary page

ekran görüntüsü, yazılım, multimedya yazılımı, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturulduSpecial requests page of human resources

ekran görüntüsü, yazılım, metin, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Leave requests page of human resources

ekran görüntüsü, yazılım, metin, multimedya yazılımı içeren bir resim

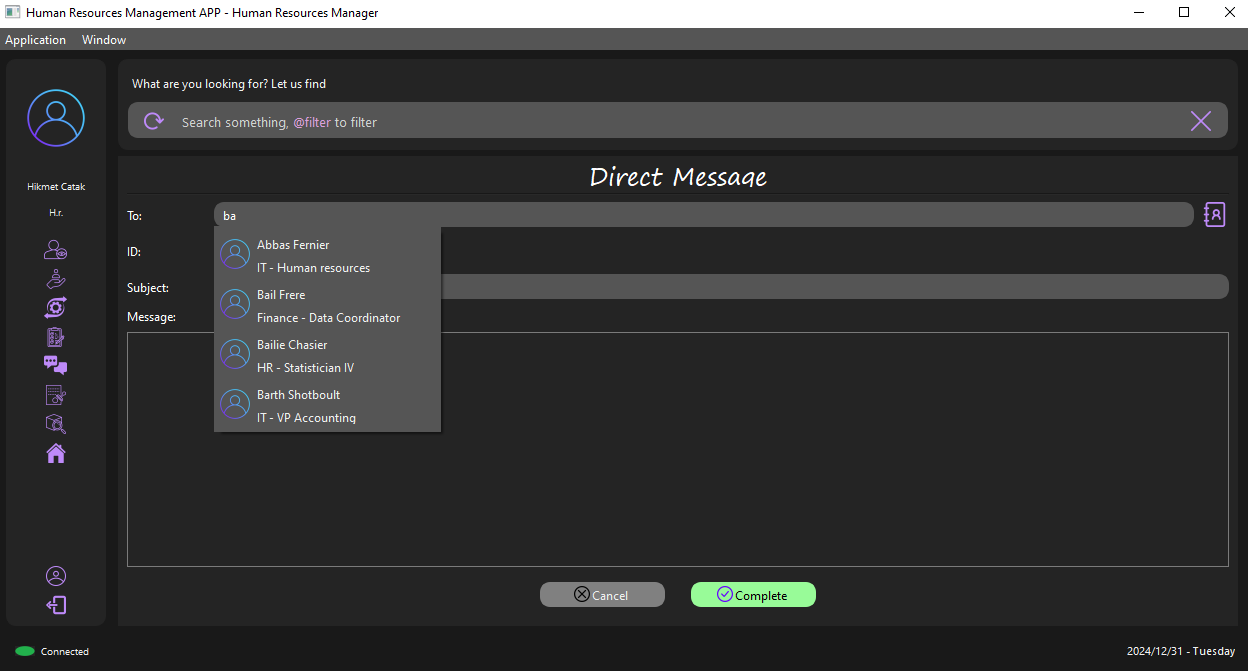
Açıklama otomatik olarak oluşturuldu

Send group email page: Emails can be sent email to more than one employees

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Send group message page: Messages can be sent messages to more than one employees

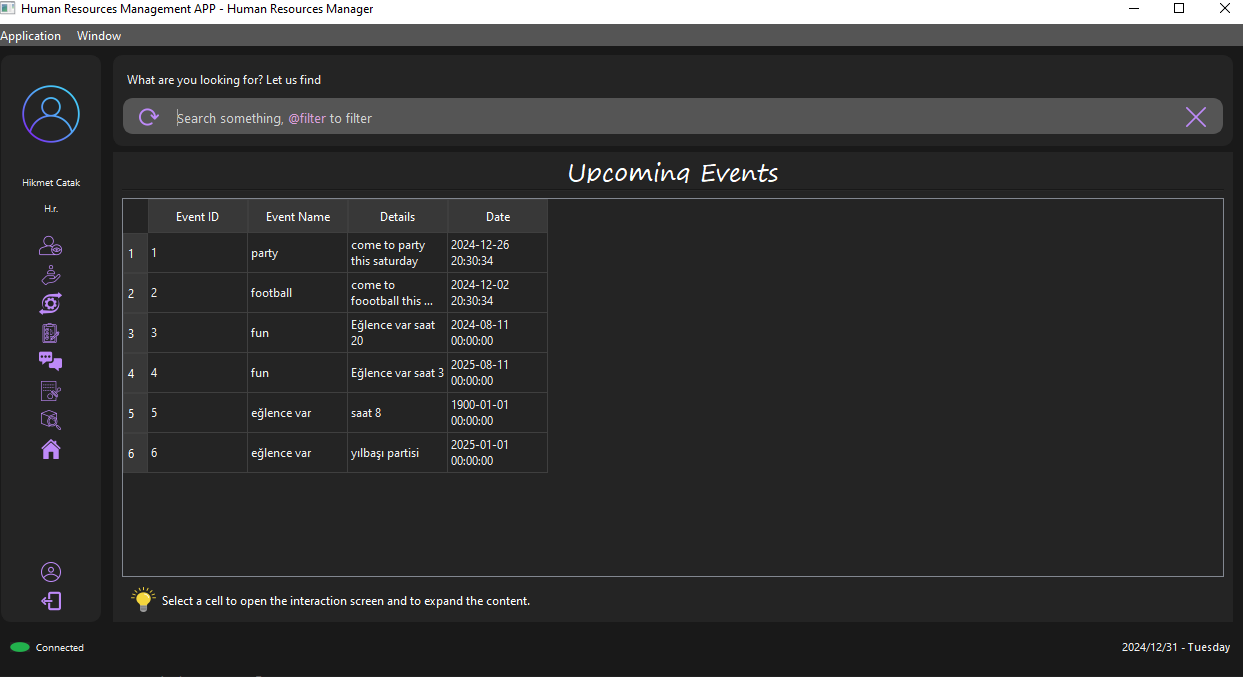


Send direct message page: It can be searched name or surname of employee on database to send message

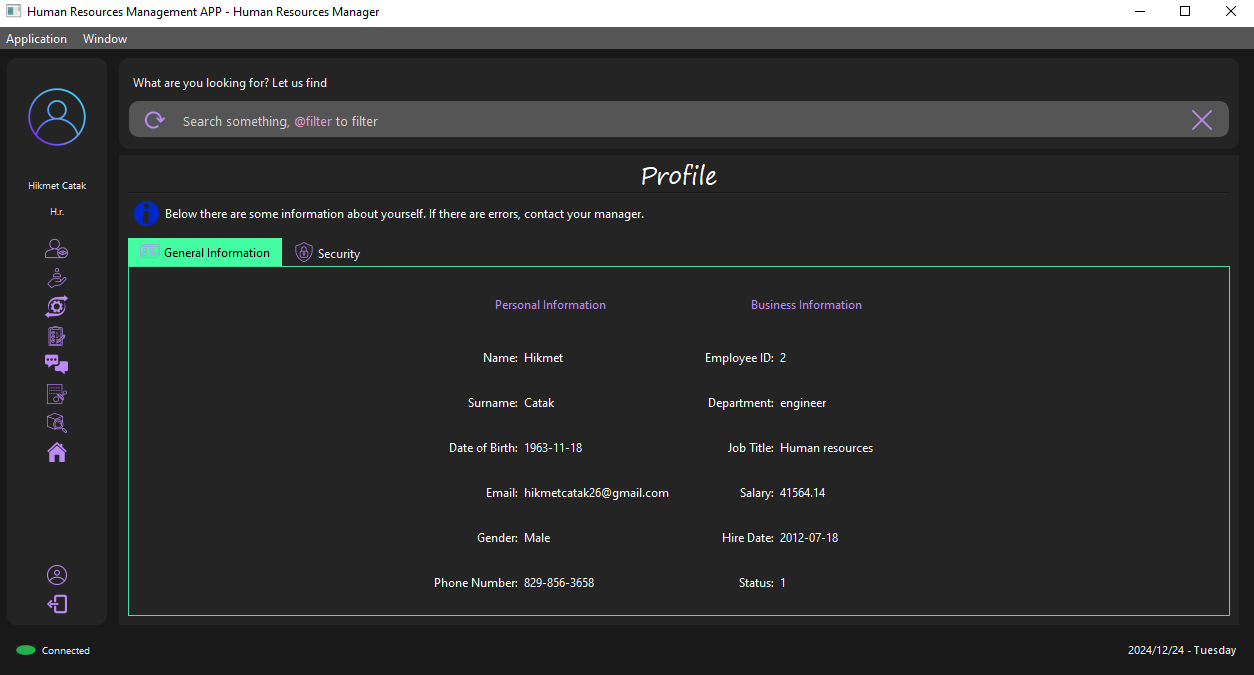
ekran görüntüsü, metin, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

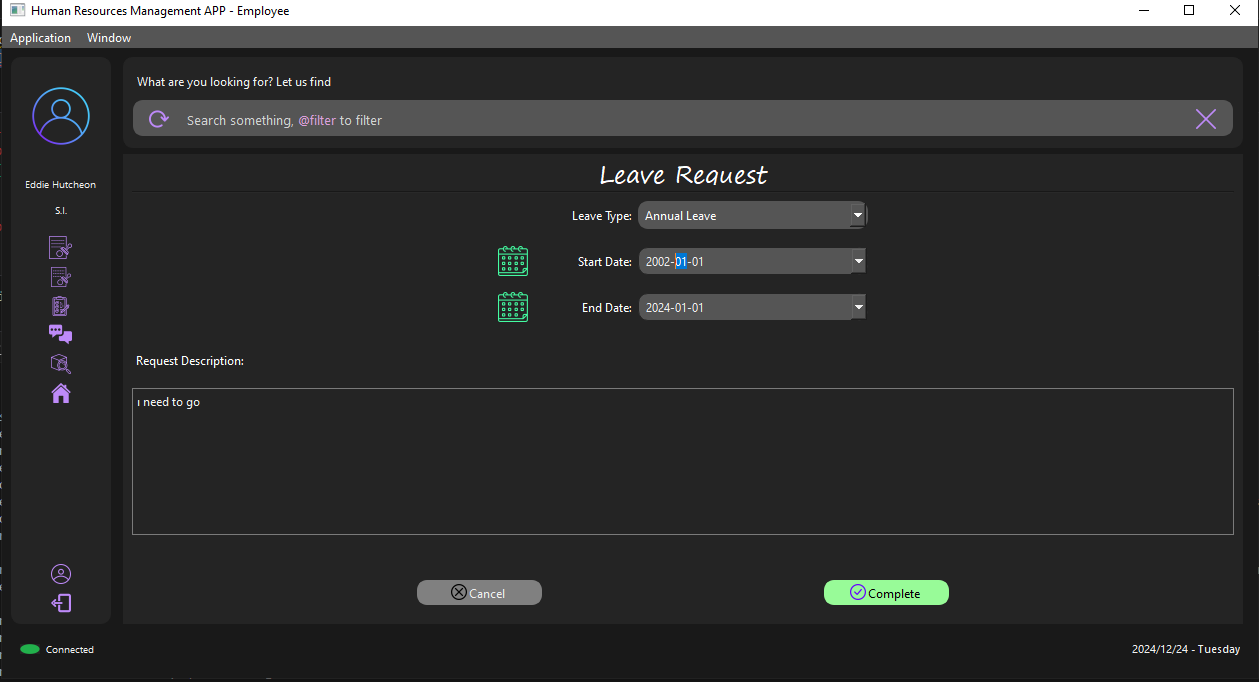
Creating Event Page:Events can be scheduled here



Upcoming events page: Shows future events



Profile page: Shows general information.

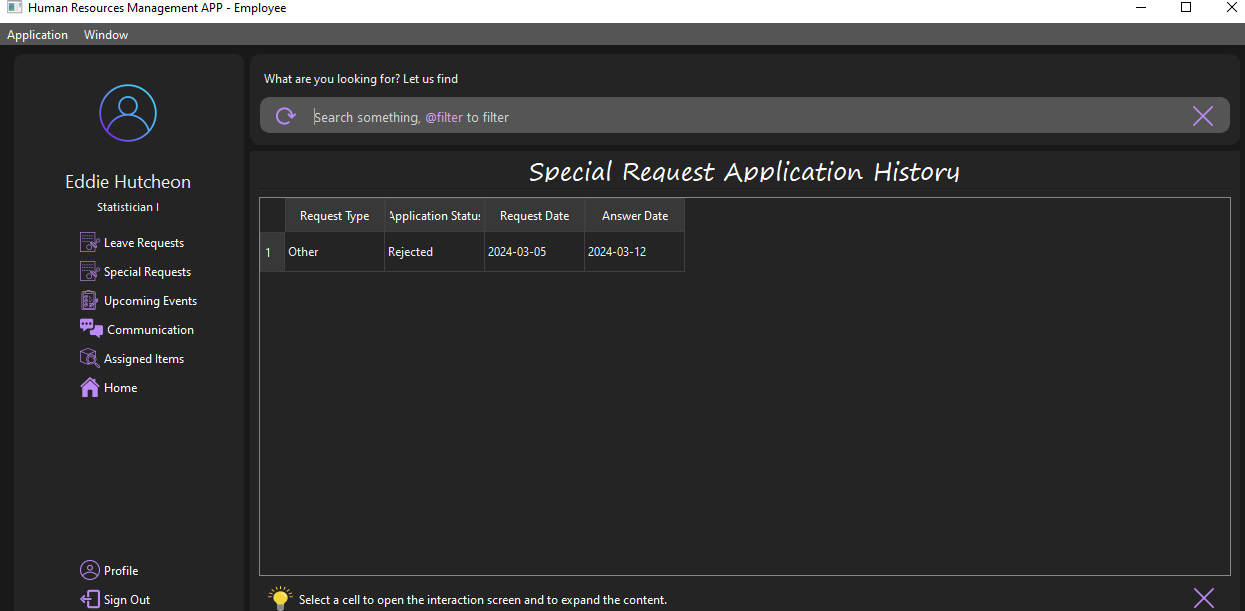


Demanding Leave Request page: Employees demand leave requests.

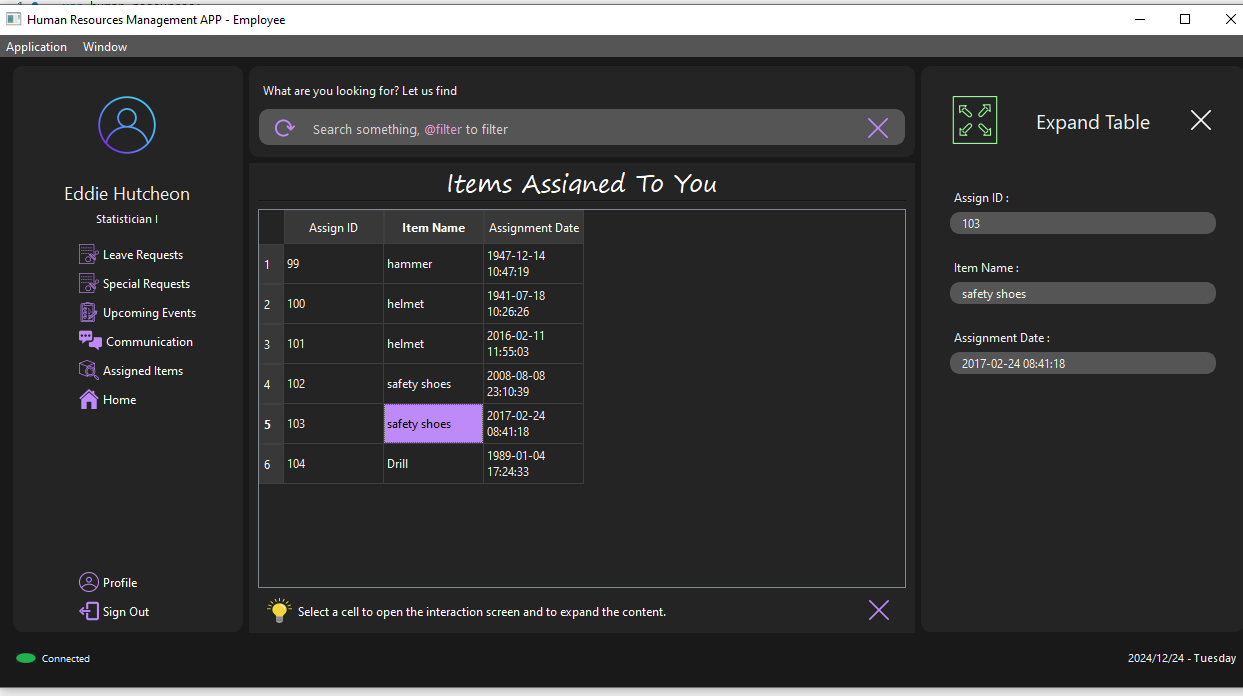
metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Leave Reuqest History Page:Shows past leave requests



Special Reuqest History Page: Shows past special requests



Assigned item Page for employees: Shows assigned items

* 1. **Appendices**
* Include all relevant code used for the project, organized by functionality (e.g., database schema, connectivity code, interface design scripts).
* Everything is in HR\_project folder

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

* 1. **System Testing and Validation**

**(No writing here—this section is for live demonstration!!)**

* After presenting all components, test the program in front of your classmates.
* Demonstrate functionalities like adding, deleting, and modifying records through the interface and confirm that the changes are reflected in the database.