



# ABDULLAH GÜL ÜNİVERSİTESİ

**COMP204 Term Project  
Phase 2**

**Instructor  
Samet Tonyalı**

**Prepared By  
Nihat Yalçın  
Ahmet Sezgin  
Muhammed Eşit  
Yusuf Bilgin**

## **Table of Contents**

Abstract	3
Summary	3
General Description	4
Requirement Analysis	4
Specification	5
Tool/ IDEs	6
Diagram /Users	7
ER	8
Philosophy	9
High-Level Design	10
ER to Relational Mapping	11
Normalization	13
More Details About The Application	14
Database Schemas	15
Table Creation Scripts:	18
Constraints of Tables:	21
Screenshots	22

## **Abstract**

This article contains the means of implementation of an internship management database system. It is essential for universities to be connected to their students after their graduation and during their internship since they can be valuable assets to the said university. Moreover, automation in the process of internship management will save time and reduce paperwork. The aim of the project is to make a desktop application that allows the university to stay in contact with the students during and after their internship process. What will be outlined in this article are the ER diagram, database design, system requirements, tools used, analysis and specifications in detail.

## **Summary**

This report is based on a project that created a database to make connections between University students and companies and universities faster and more effectively. In the realization of this project, programs such as SQL for creating the connection between people and systems, and Pyqt5 for desktop applications will be used. In the following sections of the report, the general description of the project, requirements analysis, specifications, IDEs, UML, High-level diagram, E-R diagram, design philosophy, cardinalities and user permissions are mentioned in detail.

## General Description

The purpose of the application is to automate paperwork of the internship process and save time with the help of the MySQL database management system. The main data to be stored in the database will be the students' and companies' data. For example, the contact information of the students and companies, student information (e.g. id, department, past experiences and so on), type of internship (whether it is long-term or compulsory, voluntary). The app will be used by the university, companies and students. Each will have an account and log in with the required data to the app. How they will utilize the app listed below:

- ❖ **University:** The university can track the students' internships and job life. This means it can follow the career of the students. Also, it can approve the required internship papers, and then upload some needed papers to the app.
- ❖ **Company:** Companies can see the profile of the students so as to reach suitable candidates.
- ❖ **Student:** Students can create a profile with their educational information. They can upload or prepare CVs and share their experiences. In addition, they can see other students' profiles, including graduated ones. Moreover, they can upload required documents for the internship approval process and trace the process.

## Requirement Analysis

Thanks to the internship database application, students will be able to easily continue the necessary steps before and during the internship. The required documents will be seen in the application and when the student completes them, they will be uploaded to the system. This situation prevents disruptions and delays in the document delivery process.

The school will start the student's insurance after the completion of the required documents and this will be shown by the system. After this approved stage, the student internship process will begin.

In addition to the current status of the students currently studying on the system, the companies where the graduate students have done their internship and are currently working will be listed. This will provide networking opportunities for students in search of internships and jobs in the next period. In addition, the effect of communication with graduate students will be great at the stage of learning the necessary criteria

## **Specification**

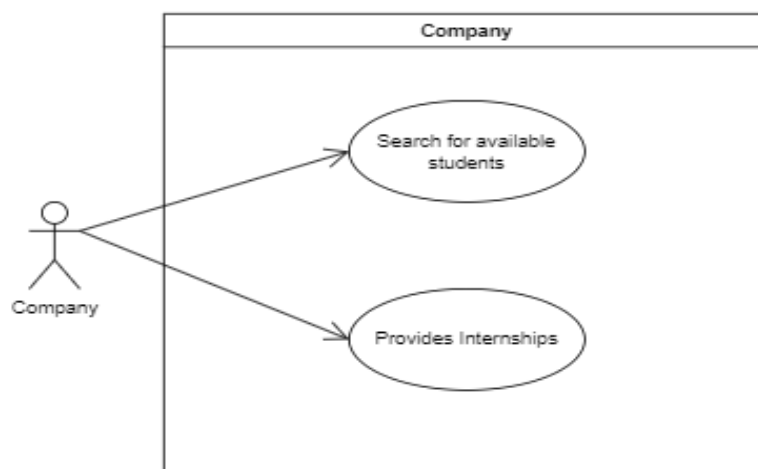
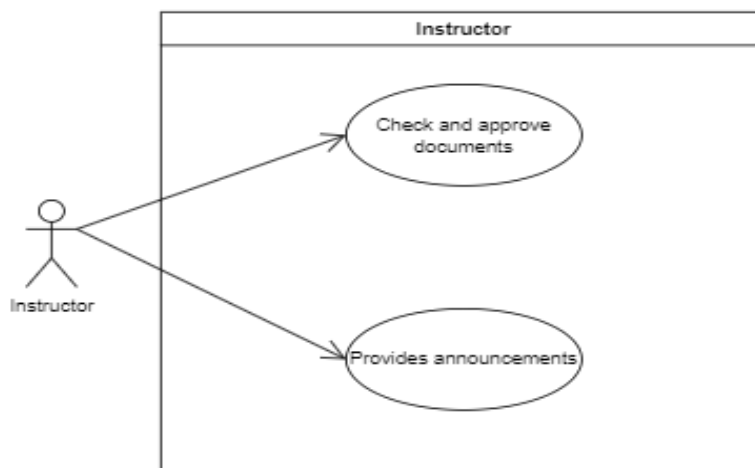
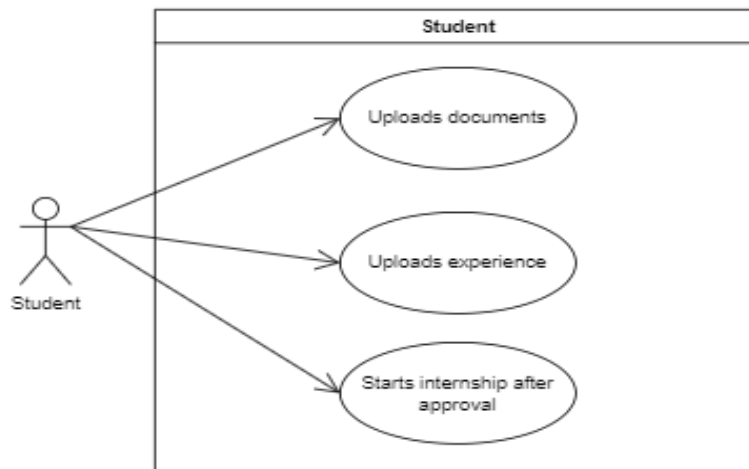
Students do voluntary and compulsory internships in order to spend their summer periods productively. After the stage of finding a company, which is a difficult process for students, the approval process of the required documents between the school and the company begins. After the documents requested by the school are filled in by the student, they must be approved by the school. Then the document that goes to the company for approval is sent back to the school. Delays and disruptions occur in the process of performing this process manually or via e-mail. With the application presented within the scope of the project, this process will take place mutually between the school and the student through a single platform. This will prevent time and document losses. When all the documents are completed, the school will approve the process by the system and start the necessary insurance procedures. The student, who sees the insurance approval in the system, can send the required document to the company quickly and easily. Since each student has his own user login, the student will be able to create his own profile. Thus, the company will reach the right student within the scope of the required criteria.

Finally, the registration of graduate students in the system will make the process more efficient. School graduates will be able to follow their careers easily. Students will also be able to take an example for their career processes by examining the alumni profiles. In addition, networking opportunities will be provided by reaching the students who work in the companies they want or who have done their internships before.

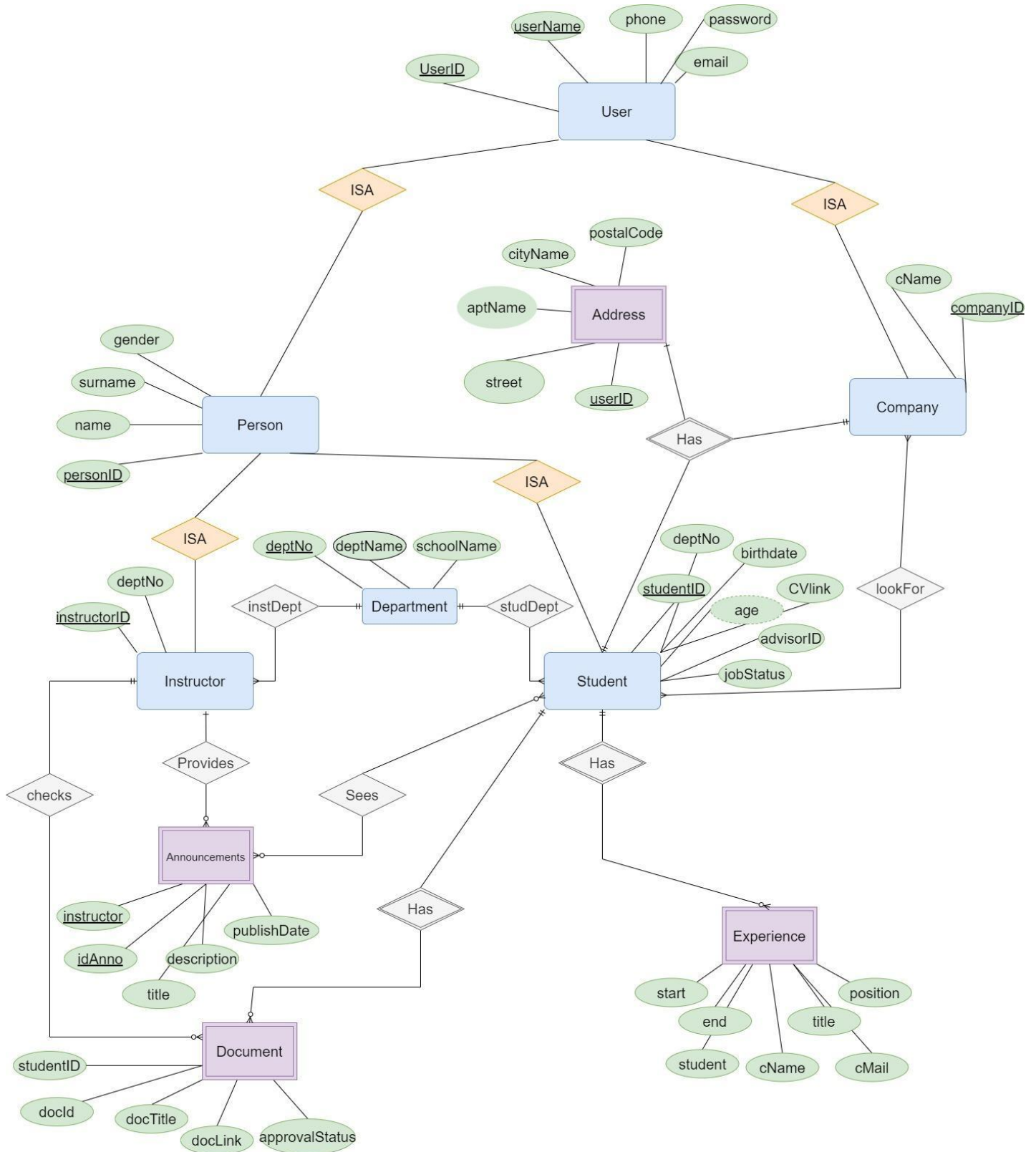
## **Tool/ IDEs**

- ❖ **MySQL Workbench:** MySQL Workbench will be used to create and manage the databases used in this project.
- ❖ **Python PyQt5:** PyQt5 library will be used to create and manage the GUI of our application.
- ❖ **Python MySQL Connector:** It allows us to manage database operations in python.
- ❖ **Visual Studio:** Visual Studio will be used as an IDE to manage the coding process.
- ❖ **Trello:** Trello will be used to design and list the requirements for this project. It will also help us distribute the workload among us.
- ❖ **Github:** Github will be used to synchronize our codes and manage our version control.

## Diagram /Users



# ER





## Philosophy

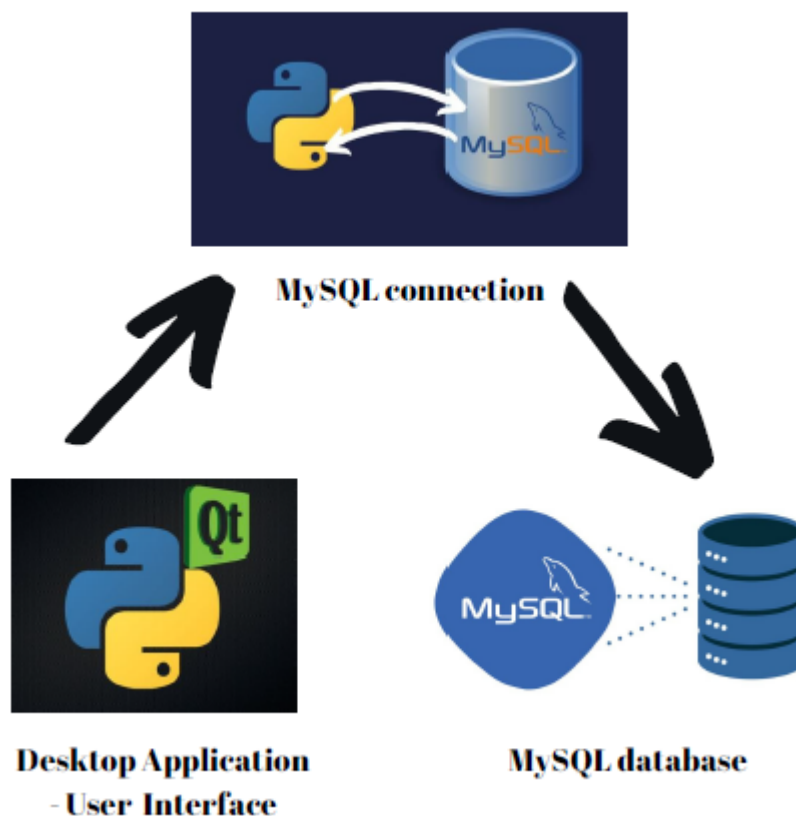
In this part, the ER diagram will be explained as much as explicitly to avoid confusion in the diagram.

### Entities:

- ❖ **User:** The user table keeps general information about the users of the app. There are four attributes: `userId`, `userName`, `phone` and `email`. The `userId` is the primary key for the table `user`. In addition, the user is the superclass for the tables, `person` and `company`.
- ❖ **Person:** `Person` is one of the subclasses of the table `user`. In the app, a user can be either a person or a company. `Person` is super-class for `instructor` and `student`. There are five attributes of a person: `personId`, inherited from `user` table, `name`, `surname` and `gender`. The primary key for the table `person` consists of `personId`.
- ❖ **Company:** As discussed above, a user can be a company. A company has two attributes: `companyID`, and `cName` (standing for company name). `companyID` is the primary key.
- ❖ **Instructor:** `Instructor` is the person who publishes announcements and checks the students' documents required for an internship. It has only one attribute, `instructorID` which is the primary key. In addition, every instructor has a department whose data is kept in the table `department`.
- ❖ **Announcements:** `Announcements` table is a weak entity. It has a number of attributes such as `status`, `idAnno` and `deadline`.
- ❖ **Student:** As discussed above a user can be a student. A student sees the announcements provided by instructors. Also, a student can have documents required for an internship and have address data. Moreover, a student does an internship at a company. The attributes that the students table has are `studentID`, `addressID`, `jobStatus`, `birthdate`, and `age`. The attribute `age` is a derived attribute from `birthdate`. `studentID` is the primary key for the table `student`.
- ❖ **Document:** A student can have zero or many documents for its internship. The documents are checked by the instructor. When approved, the student will be able to do their internship. The attributes of the document are `docId`, `docTitle`, `docDetail`, `approvalStatus`.

- ❖ **Experience:** Experience is a weak entity. A student can have zero or many experiences. Besides, experience is a superclass for internship. An experience has the attributes start, end, duration, title, and description. The attributes start and end are used to calculate the duration of the experience.
- ❖ **Department:** Department indicates the department of the instructors or students. Each instructor and student can have only one department. The attributes of the table department consist of deptName and schoolName. The primary key is deptName. The faculty name where the department belongs is kept as schoolName.
- ❖ **Address:** The table address is used to keep the address data of students and companies. An address consists of userID, cityName, postalCode, apartmentNo and street attributes. Every company and student can have only one address.

## High-Level Design



## **ER to Relational Mapping**

### **Normal Entities:**

User (UserID, userName, phone, email)

Person ( personID, name, surname, gender)

Company ( companyID, cName)

Student (studentID, jobStatus, birthdate, age)

Instructor (UserID, personID, instructorID, idAnno)

Department (deptName, schoolName)

### **Weak Entities:**

Address (addressId, street, apartmentNo, cityName, postalCode)

Experience (personID, start, end, duration, title, description)

Announcements (announcementID, instructorID, status, deadline)

Document (docID, docTitle, docDetail, approvalStatus)

## **CARDINALITIES**

### **ONE TO ONE**

Between User and Person

Between User and Company

Between Person and Instructor

Between Person and Student

Between Internship and Experience

### **ONE TO MANY**

Between Student and Address

Between Student and Internship

Between Student and Document

Between Student and Experience

Between Internship and Company

Between Internship and Document

Between Instructor and Document

Between Company and Internship

Between Instructor and Announcement

### **MANY TO MANY**

Between Student and Announcement

Between Student and Instructor

Between Student and Company

## **SPECIALIZATION**

### **Super Class**

User (UserID, userName, phone, email)

### **Sub-Class**

Person (UserID, personID, name, surname, gender)

Company (UserID, companyID, cName, addressID)

## Normalization

Normalization is a design technique that reduces data redundancy and improves data consistency. There are different layers to normalization and a database needs to fulfill each required criteria of a given normal form to be considered appropriate for that normal form. If normalization is implemented properly, it can optimize the speed and size of the database.

There are 3 different normal forms that our database needs to be subjected to. These forms are called 1NF, 2NF and finally 3NF.

In order for a database to be considered as 1NF, these criteria need to be fulfilled:

- Every column in a table must be unique.
- Each table must be identified with a primary key.
- Cells can't contain multiple values.

According to our ER model, we paid attention to the naming of cells so the names of cells are different from each other. Also, they contain a single value. The third rule states

that all tables should have a primary key and in our model, we put a primary key in each

cell. In addition, there are two weak entities which are called ADDRESS, ANNOUNCEMENTS, DOCUMENTS, EXPERIENCE and INTERNSHIP. The weak entities can not have a unique primary key, they connect to the other table with their primary key according to the information,

Thus the weak entities in our model don't have unique primary keys. Furthermore, the primary keys of the rest of entities are shown below;

**USER:** userID, userName

**PERSON:** userID, personID

**COMPANY:** userID, companyID

**INSTRUCTOR:** userID, personID, instructorID

**STUDENT:** userID, personID, studentID

**DEPARTMENT:** deptName

For a database to be considered as 2NF these criteria need to be fulfilled:

- Database needs to be in first normal form.
- Database can't contain a Partial Dependency

Since our database model satisfies the 1NF requirements, we can apply 2NF rules. The rules indicate that there should be no partial dependency between relations. Our database fulfills these conditions, thus it is in 2NF.

For a database to be considered as 3NF these criteria need to be fulfilled:

- Database needs to be in second normal form.
- Database can't contain any Transitive Partial Dependency.

Since our database model satisfies 2NF requirements, we can check to see if it satisfies the rules of 3NF. A relation contains Transitive Partial dependency when a non-prime attribute determines another non-prime attribute. This is the case for all of our tables except the ADDRESS table which should be divided into two separate tables since the zip code in our address table determines the city in that table.

## **More Details About The Application**

In this project, it is aimed that the school can easily track graduates and interns. In addition, within the scope of the project, the issue of approval and follow-up of documents, which is a common problem faced by the school and students during the internship, was discussed. In the project, it is aimed to create a desktop application that allows the university to keep in touch with students during and after the internship.

Our tool for this desktop application will be python and mysql. In this report, user, person, company, student, instructor, announcement, internship, experience, document and address are outlined. In addition, the ER diagram has been prepared, thus making it easier to prepare the database. Moreover, in the project, the software we will use to facilitate users' access to the data in the database will be the desktop application. MySQL will be used to communicate with the database and will be developed in Python.

The start page will open when the program is run for the first time. After the login and sign up options, the user will be able to log in to their page. Our users (student, school and graduate) and company. Registration will be created in line with the required personal information. During the internship, students will upload the necessary documents to the required platform. After the documents are approved by the school, the insurance start will be made and the insurance document will be sent to the student through the necessary platform. In addition, previous internship and work experiences of graduates will be included in their user profiles. This will be instrumental in ensuring the internship and job follow-up of students and graduates. Companies will be able to access the profiles of students and graduates and enable them to provide the necessary employment.

## Database Schemas

USER TABLE				
Column	Data Type	Null	Reference	Type
UserID	int	NN	Unique identification number of users	PK
userName	VARCHAR(20)	NN	USER NAME	PK
phone	int	N	USER PHONE NUMBER	
email	VARCHAR(80)	N	USER'S EMAIL	
password	VARCHAR(20)	N	Password of user	

PERSON TABLE				
Column	Data Type	Null	Reference	Type
Gender	VARCHAR(6)	N	Gender of person	
surname	VARCHAR(20)	N	surname of person	
name	VARCHAR(20)	N	name of person	
personID	int unsigned	NN	Unique identifier of the person	PK, FK

STUDENT TABLE				
Column	Data Type	Null	Reference	Type
studentID	int unsigned	N	Unique identifier of the student	PK, FK
birthdate	DATETIME	N	Birthdate of the student	
age	int	N	Age of the student	
jobStatus	VARCHAR(3)	N	Job status of the student	
Cvlink	VARCHAR(255)	N		
advisorID	int unsigned	NN	Unique identifier of the person	FK
deptNo	int unsigned	NN	Specific number of department	FK

COMPANY TABLE				
Column	Data Type	Null	Reference	Type
CompanyID	int	NN	Unique identifier of company table	PK, FK
cName	VARCHAR(20)	N	Name of company	

INSTRUCTOR TABLE				
Column	Data Type	Null	Reference	Type
instructorID	int unsigned	NN	Unique identifier of the instructor	PK, FK
deptNo	VARCHAR(25)	NN	Specific number of department	FK

DEPARTMENT TABLE				
Column	Data Type	Null	Reference	Type
deptName	VARCHAR(20)	N	Name of department	PK
facultyName	VARCHAR(20)	N	Name of faculty	
deptNo	int unsigned	NN	Specific number of department	

ADDRESS TABLE				
Column	Data Type	Null	Reference	Type
addressID	int	NN	Identification number of Address	FK, PK
postalcode	int	N	Postal code of district	
cityName	VARCHAR(20)	N	Name of city	
apartmentNo	VARCHAR(20)	N	Number of apartment	
street	VARCHAR(20)	N	Name of street	
userID	int unsigned	N	ID of user	



ANNOUNCEMENT TABLE				
Column	Data Type	Null	Reference	Type
idAnno	int unsigned	NN	Unique identifier of the announcement	PK
instructorID	int unsigned	NN	Unique identifier of the instructor	FK
title	VARCHAR(20)	N	Status of announcement	
description	VARCHAR(255)	N	Description	
publishDate	date	N	Date of when announcement published	

EXPERIENCE TABLE				
Column	Data Type	Null	Reference	Type
studentID	int	NN	Unique identifier of the student	FK
start	DATETIME	N	Start date of internship	
end	DATETIME	N	End date of internship	
position	VARCHAR(20)	N	Which position he or she works/worked	
title	VARCHAR(2)	N	Type of experience (work or internship)	
companyName	VARCHAR(25)	N	Name of company	
companyMail	VARCHAR(25)	N	Mail of company	

DOCUMENT TABLE				
Column	Data Type	Null	Reference	Type
studentID	int unsigned	NN	Unique identifier of the student	FK
docID	int unsigned	NN	Unique identifier of the document	PK
docTitle	VARCHAR(20)	N	Title of document	
link	VARCHAR(255)	N	Link of documents	
Approval Status	VARCHAR(20)	N	Status of approval	

## Table Creation Scripts:

### User Table:

```
CREATE TABLE `user` (  
  `userID` int unsigned NOT NULL AUTO_INCREMENT,  
  `userName` varchar(20) NOT NULL,  
  `password` varchar(255) NOT NULL,  
  `phone` varchar(11) NOT NULL,  
  `email` varchar(25) NOT NULL,  
  PRIMARY KEY (`userID`,`userName`,`password`),  
  UNIQUE KEY `userID` (`userID`),  
  UNIQUE KEY `userName` (`userName`),  
  UNIQUE KEY `phone` (`phone`),  
  UNIQUE KEY `email` (`email`)  
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci;
```

### Person Table:

```
CREATE TABLE `person` (  
  `personID` int unsigned NOT NULL,  
  `name` varchar(20) NOT NULL,  
  `surname` varchar(20) NOT NULL,  
  `gender` varchar(6) NOT NULL,  
  PRIMARY KEY (`personID`),  
  UNIQUE KEY `personID` (`personID`),  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

### Department Table:

```
CREATE TABLE `department` (  
  `deptNo` int unsigned NOT NULL AUTO_INCREMENT,  
  `deptName` varchar(6) NOT NULL,  
  `facultyName` varchar(12) NOT NULL,  
  PRIMARY KEY (`deptNo`),  
  UNIQUE KEY `deptNo` (`deptNo`),  
  UNIQUE KEY `deptName` (`deptName`),  
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci;
```

### **Instructor Table:**

```
CREATE TABLE `instructor` (  
  `instructorID` int unsigned NOT NULL,  
  `deptNo` int NOT NULL,  
  PRIMARY KEY (`instructorID`),  
  UNIQUE KEY `instructorID` (`instructorID`),  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

### **Student Table:**

```
CREATE TABLE `student` (  
  `studentID` int unsigned NOT NULL,  
  `deptNo` int unsigned DEFAULT NULL,  
  `birthdate` date NOT NULL,  
  `age` int NOT NULL,  
  `jobStatus` varchar(3) NOT NULL,  
  `advisorID` int unsigned DEFAULT NULL,  
  `CVlink` varchar(255) DEFAULT NULL,  
  PRIMARY KEY (`studentID`),  
  UNIQUE KEY `studentID` (`studentID`),  
  UNIQUE KEY `deptNo` (`deptNo`),  
  UNIQUE KEY `advisorID` (`advisorID`),  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

### **Company Table:**

```
CREATE TABLE `company` (  
  `companyID` int unsigned NOT NULL,  
  `cName` varchar(25) NOT NULL,  
  PRIMARY KEY (`companyID`),  
  UNIQUE KEY `companyID` (`companyID`),  
  UNIQUE KEY `cName` (`cName`),  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

### **Document Table:**

```
CREATE TABLE `document` (  
  `IDstudent` int unsigned NOT NULL,  
  `docID` int unsigned NOT NULL AUTO_INCREMENT,  
  `docTitle` varchar(25) NOT NULL,  
  `link` varchar(255) NOT NULL,  
  `approvalStatus` varchar(3) NOT NULL,  
  UNIQUE KEY `docID` (`docID`),
```

```

KEY `IDstudent` (`IDstudent`),
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

```

### **Experience Table:**

```

CREATE TABLE `experience` (
  `student` int unsigned NOT NULL,
  `start` date NOT NULL,
  `end` date NOT NULL,
  `title` varchar(25) NOT NULL,
  `position` varchar(25) NOT NULL,
  `companyName` varchar(25) NOT NULL,
  `companyMail` varchar(25) NOT NULL,
  KEY `student` (`student`),
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

### **Announcement Table:**

```

CREATE TABLE `announcement` (
  `instructor` int unsigned NOT NULL,
  `idAnno` int unsigned NOT NULL AUTO_INCREMENT,
  `title` varchar(30) NOT NULL,
  `description` varchar(255) NOT NULL,
  `publishDate` date NOT NULL,
  PRIMARY KEY (`instructor`,`idAnno`),
  UNIQUE KEY `instructor` (`instructor`),
  UNIQUE KEY `idAnno` (`idAnno`),
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

```

### **Address Table:**

```

CREATE TABLE `address` (
  `userID` int unsigned NOT NULL,
  `postalcode` varchar(5) NOT NULL,
  `city` varchar(20) NOT NULL,
  `street` int NOT NULL,
  `aptName` varchar(20) NOT NULL,
  PRIMARY KEY (`userID`),
  UNIQUE KEY `userID` (`userID`),
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

## **Constraints of Tables:**

### **Constraint of Table PERSON:**

```
CONSTRAINT `personID` FOREIGN KEY (`personID`) REFERENCES `user`  
(`userID`) ON DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT `person_chk_1` CHECK (((`gender` = _utf8mb4'male') or (`gender`  
= _utf8mb4'female')))
```

### **Constraint of Table DEPARTMENT:**

```
CONSTRAINT `department_chk_1` CHECK (((`deptName` = _utf8mb4'COMP') or  
(`deptName` = _utf8mb4'EE') or (`deptName` = _utf8mb4'IE') or (`deptName` =  
_utf8mb4'ME') or (`deptName` = _utf8mb4'BA') or (`deptName` = _utf8mb4'POL'))),  
CONSTRAINT `department_chk_2` CHECK (((`facultyName` =  
_utf8mb4'Engineering') or (`facultyName` = _utf8mb4'Others')))
```

### **Constraint of Table INSTRUCTOR:**

```
CONSTRAINT `instructorID` FOREIGN KEY (`instructorID`) REFERENCES  
`person` (`personID`) ON DELETE CASCADE ON UPDATE CASCADE
```

### **Constraint of Table STUDENT:**

```
CONSTRAINT `advisorID` FOREIGN KEY (`advisorID`) REFERENCES  
`instructor` (`instructorID`) ON DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT `deptNo` FOREIGN KEY (`deptNo`) REFERENCES `department`  
(`deptNo`) ON DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT `studentID` FOREIGN KEY (`studentID`) REFERENCES `person`  
(`personID`) ON DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT `student_chk_1` CHECK (((`jobStatus` = _utf8mb4'yes') or  
(`jobStatus` = _utf8mb4'no')))
```

### **Constraint of Table COMPANY:**

```
CONSTRAINT `companyID` FOREIGN KEY (`companyID`) REFERENCES  
`user` (`userID`) ON DELETE CASCADE ON UPDATE CASCADE
```

### **Constraint of Table DOCUMENT:**

```
CONSTRAINT `IDstudent` FOREIGN KEY (`IDstudent`) REFERENCES `student`  
(`studentID`) ON DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT `document_chk_1` CHECK (((`approvalStatus` = _utf8mb4'yes') or  
(`approvalStatus` = _utf8mb4'no')))
```

### Constraint of Table EXPERIENCE:

CONSTRAINT `student` FOREIGN KEY (`student`) REFERENCES `student` (`studentID`) ON DELETE CASCADE ON UPDATE CASCADE

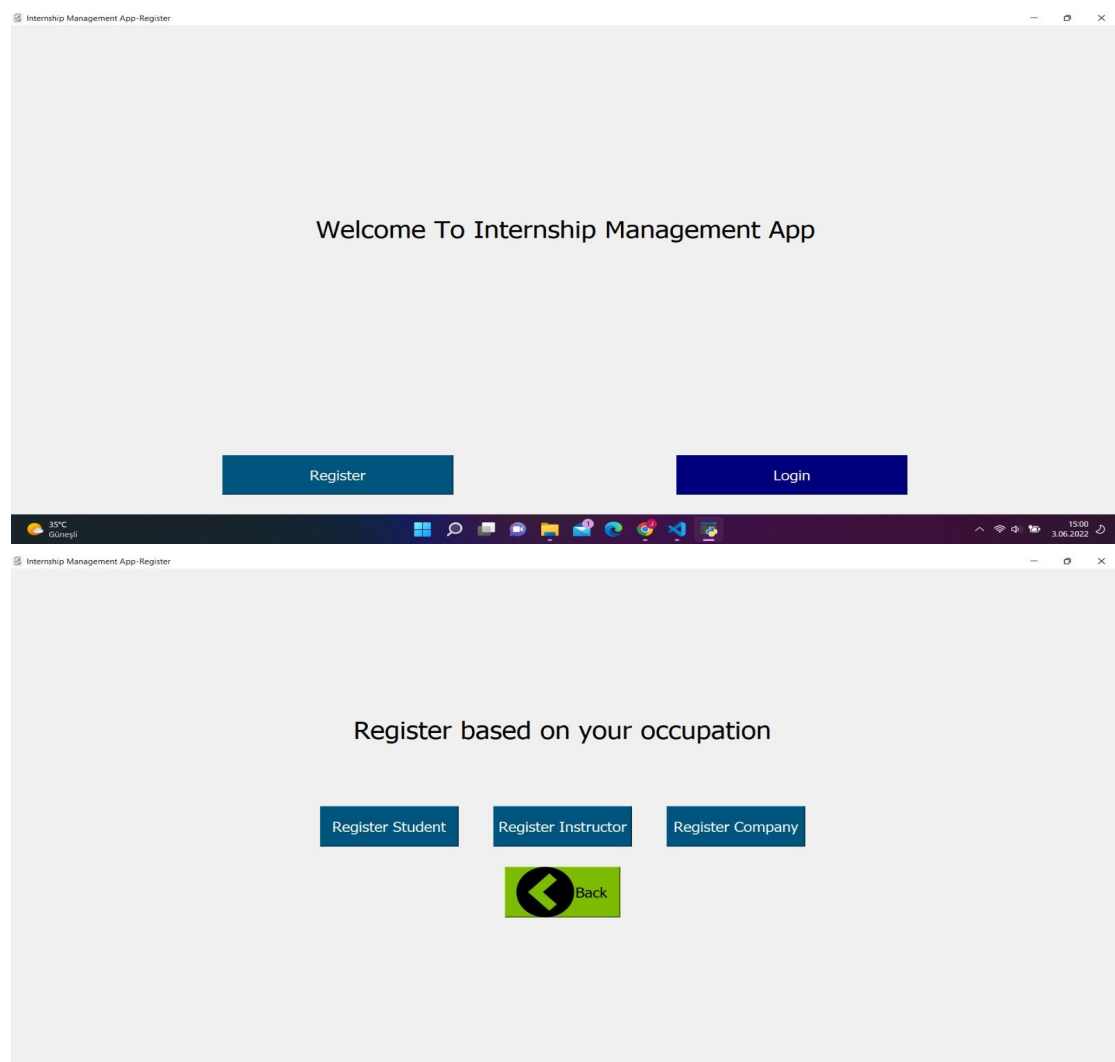
### Constraint of Table ANNOUNCEMENT:

CONSTRAINT `instructor` FOREIGN KEY (`instructor`) REFERENCES `instructor` (`instructorID`) ON DELETE CASCADE ON UPDATE CASCADE

### Constraint of Table ADDRESS:

CONSTRAINT `userID` FOREIGN KEY (`userID`) REFERENCES `user` (`userID`) ON DELETE CASCADE ON UPDATE CASCADE

## Screenshots



Student Register Page

USERNAME  
|

PASSWORD

CONFIRM PASSWORD

PHONE NUMBER

EMAIL

NAME

SURNAME

GENDER  
☐ MALE ☐ FEMALE

BIRTHDATE  
1.01.2000

DEPARTMENT NAME

FACULTY NAME

REGISTER GO BACK

Instructor Register Page

USERNAME  
|

PASSWORD

CONFIRM PASSWORD

PHONE NUMBER

EMAIL

NAME

SURNAME

GENDER  
☐ MALE ☐ FEMALE

DEPARTMENT NAME

FACULTY NAME

REGISTER GO BACK

Company Register Page

USERNAME  
|

PASSWORD

CONFIRM PASSWORD

PHONE NUMBER

EMAIL

COMPANY NAME

REGISTER GO BACK

Login Page

Student Login

Enter Username: example

Enter password:

Student Login

Company Login

Enter Username:

Enter password:


Company Login

Instructor Login

Enter Username:

Enter password:

Instructor Login



Student Home Page

Welcome Name Surname

Your information

Phone number: 05426147696

Email Address: testuser@gmail.com

→ LOGOUT

Profile

Documents

Announcements

Experiences

Student Documents Page

List Documents

Add Documents

→ Go Back

Your Documents

Document1: Document Title documentlink.com

Document2: Document Title documentlink.com

Document3: Document Title documentlink.com



Student Profile Page

UPDATE INFORMATIONS [→ Back](#)

POSTAL CODE

CITY

STREET

enter positive integer

APARTMENT NAME

UPDATE ADDRESS

EMAIL

UPDATE EMAIL

PHONE

UPDATE PHONE NUMBER

Student Details

NAME: My Name

SURNAME:

CITY:

CV LINK:

EMAIL:

DEPTNAME:

PHONE:

EXPERIENCES:

[→ Back](#)

Student Experiences Page

**EXPERIENCES**

[→ Back](#)

title

decription

start

1.01.2000

end

1.01.2000

status

☐ working now

position

ADD EXPERIENCE

Instructor Home Page

WELCOME NAME SURNAME

Publish Announcement

Check Student Documents

Logout

Profile

Approval Page

Approve Documents

Go Back

List Documents

Results

approve

disapprove

Approval Page

Publish Announcement

Title:

Description:

Add

Go Back

Company Home Page

WELCOME Cname

enter department name to search:

List available students

Results

show details

Profile settings

→ Logout