COMP 204

DATABASE MANAGEMENT SYSTEMS

FINAL REPORT

STUDENT INTERNSHIP DATABASE SYSTEM

08.06.2020

Prepared By

Enes Kaçan

Muhammed Raşit Aydın

Rıdvan Şahin

ABSTRACT

This paper explains the information about internship database system to be created for Abdullah Gul University. The aim of the project is to help Abdullah Gul University students in finding an internship by creating a database system containing past internship information. General description and requirement analysis of the project, the E-R diagram of the system, philosophy of design, required tools, high level and UML use case diagrams are defined and presented.

TABLE OF CONTENT

LIST OF FIGURES

REQUIREMENT ANALYSIS

Internship is an important step for students' future career and it is a great opportunity for them to gain experience in business life. However, it is not easy for students to get this important experience due to the fact that students have to find their own internship place. In order to ease this process for the students of Abdullah Gul University, we decided to create a database containing past internship information of students. At this point, it is crucial to store data of students' previous internships to help lower grade students to find internship opportunities in a shorter time period and in an easier way. This data includes information about companies or universities that internship has been conducted, contact persons, location, research field. Besides this, information about students is also important since people want to filter internship opportunities that done by ones with similar profiles and backgrounds with themself. So, student's department, GPA, research or interest area and their comments about the internship experience. Also, if the student allows, their contact information can be stored so internship searching students can directly contact them and get further information and help. Since this will promote cooperation between students, it will take an important role to create a university culture.

TOOLS and IDES

Android Studio: This IDE is used for writing flutter code and making an android application.

MySQL Workbench: This software will be used to design an extended entity relation diagram.

ERDPlus: This website will be used to create E-R diagrams.

Adobe XD: This is used for designing the UI of the app

Github: This is used as version control system and used for sharing files with group members

HIGH LEVEL DIAGRAM

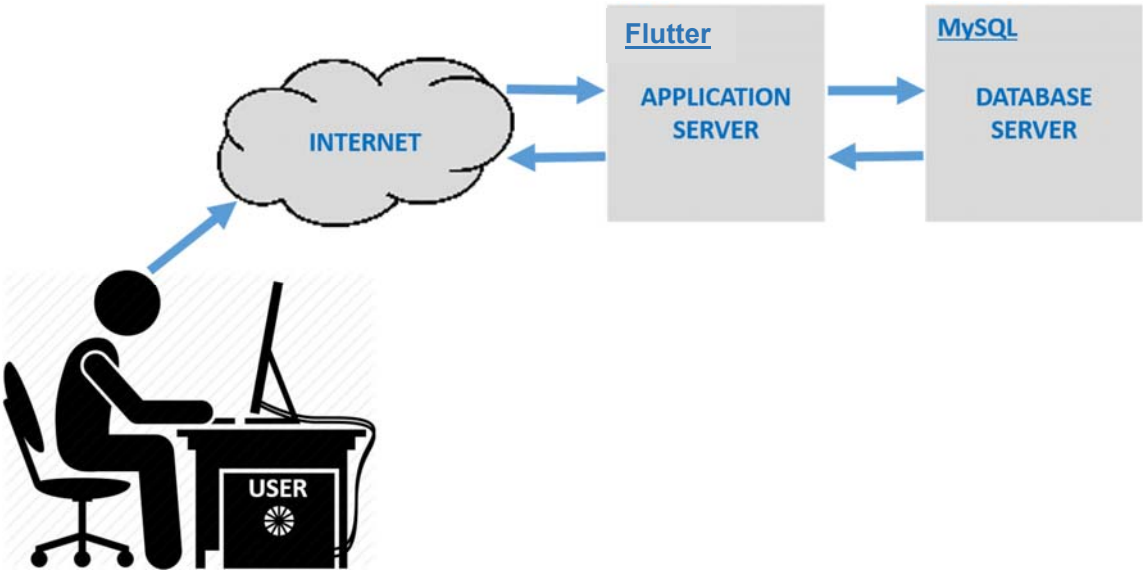

Figure 1. High Level Diagram

UML USE CASE DIAGRAM

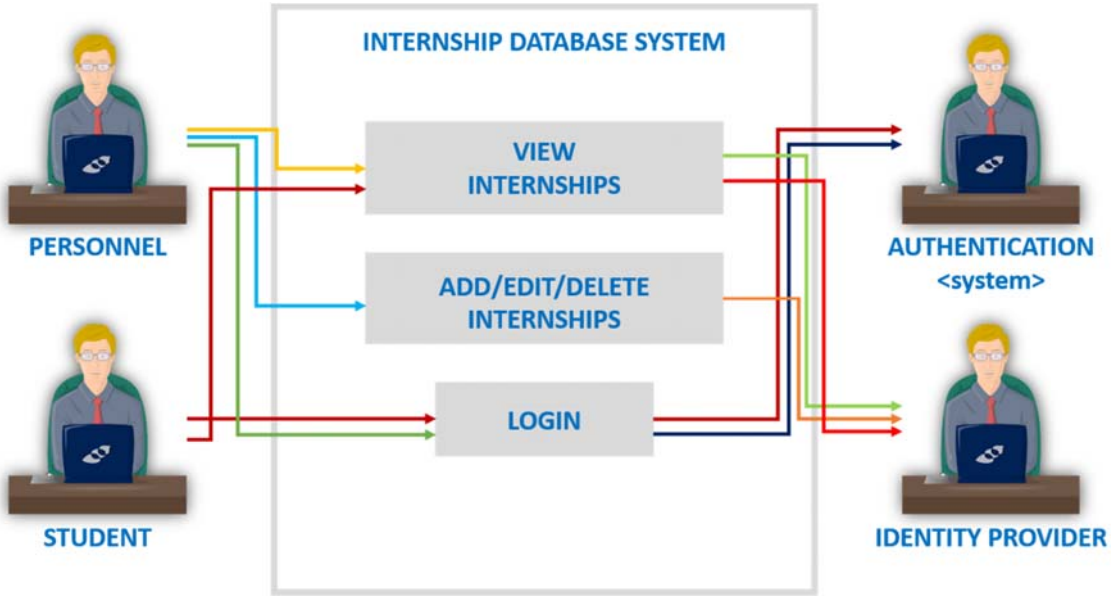

Figure 2. UML Use Case Diagram

E-R DIAGRAM

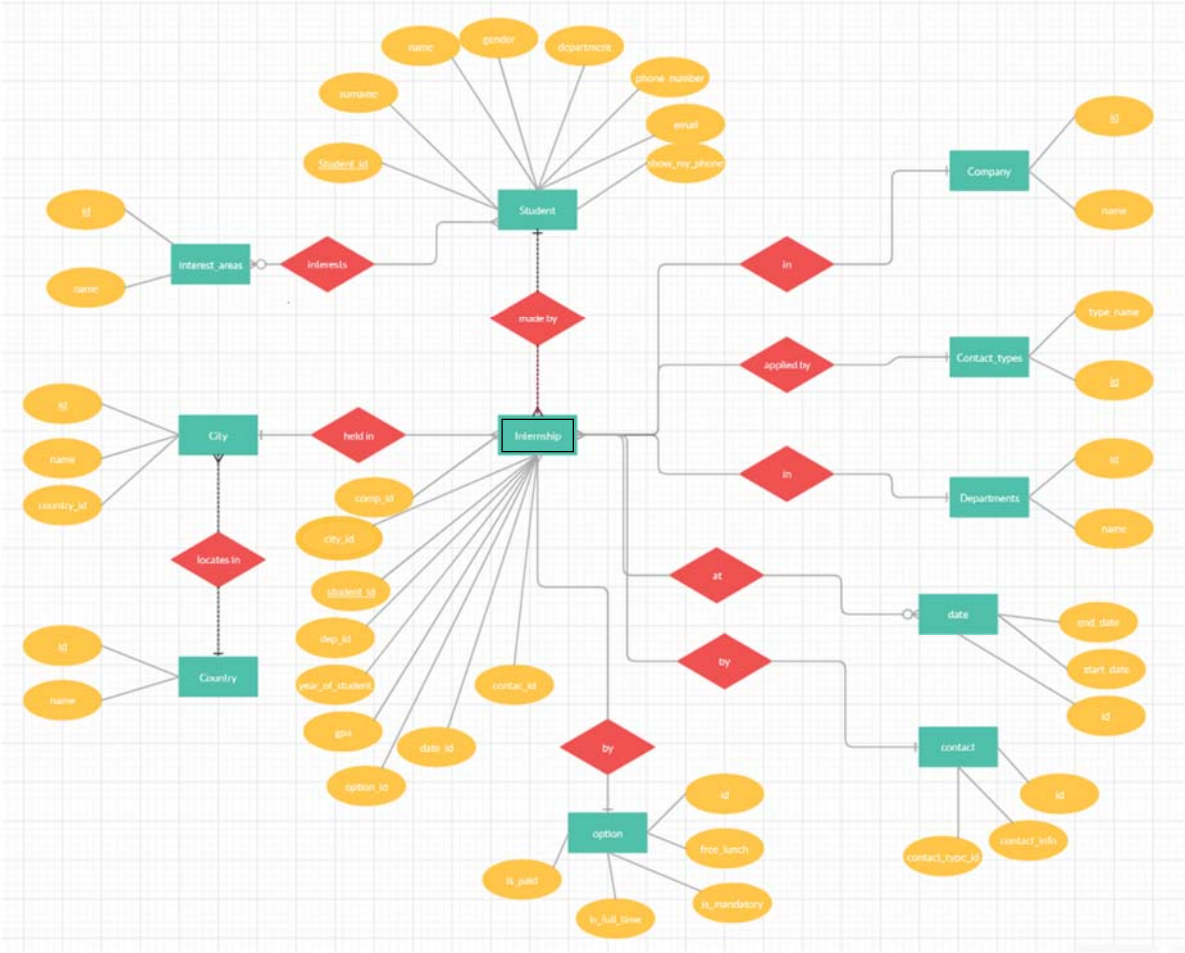

Figure 3. ER Diagram

DESIGN PHILOSOPHY

Our modeling has been tried to be designed as flexible as possible. It is aimed to eliminate unnecessary confusion by using simple and plain content. Considering future plans, changes can be made on model design.

Entities:

- internship_info:

  internship_info is the main table of the relationship. It has links with other tables. It is a weak entity.

- contact_types:

  contact_types is important to decide on the type of contact information. With this entity, the user can decide the contact way which is e-mail, phone, and website. Thus confusion is prevented.

- Country
- cities:

  Country and cities are informed about the location of the internship. They are linked to each other. Also city is weak entity.

- student:

  student has the general information about interns. Every student has unique ID which is giving from university.

- interest_areas:

  interest_areas is listing all the topics which is interested by all the students.

- interests:

  interests is showing all the interests of a student. It uses interest_areas to reach all topics and show student's interest areas.

- department
- company:

  company and department has information about companies. Every company has departments and departments show the area of the work.

Many to One Relations:

- internship_info and student
- internship_info and company
- internship_info and contact_types
- internship_info and departments
- internship_info and option
- internship_info and date
- internship_info and contact
- internship_info and city
- city and Country

Many to Many Relations:

- student and interest_areas

USER PERMISSION

Student:

Edit: Student

View: Internship

Personnel:

Add: Student, Internship

Delete: Student, Internship

Edit: Student, Internship

View: Student, Internship

E-R TO RELATIONAL MAPPING

Handling Entities

CITY(id, name, country_id)

COMPANY(id, name)

CONTACT(id, contact_info, contact_type_id)

CONTACT_TYPES(id, name)

COUNTRY(id, name)

DATE(id, start_date, end_date)

DEPARTMENTS(id, name)

INTEREST_AREAS(id, name)

OPTION(id, free_lunch, is_paid, is_full_time, is_mandatory)

STUDENT_INTERESTS(interest_areas_id, student_student_id)

INTERNSHIP(id, year_of_student, gpa, city_id, contact_id, option_id, date_id, departments_id, company_id, student_id)

STUDENT(student_id, name, surname, gender, department, phone_number, email, show_my_phone)

DATABASE SCHEMA

CITY TABLE

Column	Data Type	Null	Domain	Identification
id	INT	N	Sequence#	Unique identifier for a City class
name	VARCHAR(45)	N		City name
Country_id	INT	N		Unique identifier for a Country class

Example:

1	Kayseri	3

Unique Index: -

Purpose: Stores general information about cities.

COMPANY TABLE

Column	Data Type	Null	Domain	Identification
Id	INT	N	Sequence#	Unique identifier for a Company class
Name	VARCHAR(45)	N		Company name

Example:

3	Facebook

Unique Index: -

Purpose: Stores general information about companies.

CONTACT TABLE

Column	Data Type	Null	Domain	Identification
id	INT	N	Sequence#	Unique identifier for a Contact class
Contact_info	VARCHAR(45)	N		Contact info
Contact_type_id	INT	N		Unique identifier for a Contact Type class

Example:

1	x	3

Unique Index: -

Purpose: Stores general information about contacts.

CONTACT TYPES TABLE

Column	Data Type	Null	Domain	Identification
Id	INT	N	Sequence#	Unique identifier for a Contact Types class
Name	VARCHAR(45)	N		Contact type name

Example:

2	E-Mail

Unique Index: -

Purpose: Stores general information about contact types.

COUNTRY TABLE

Column	Data Type	Null	Domain	Identification
Id	INT	N	Sequence#	Unique identifier for a Country class
Name	VARCHAR(45)	N		Country name

Example:

12	Italy

Unique Index: -

Purpose: Stores general information about countries.

DATE TABLE

Column	Data Type	Null	Domain	Identification
id	INT	N	Sequence#	Unique identifier for a Date class
Start_date	DATE	N		date
End_date	DATE	N		date

Example:

1	06.06.2020	06.09.2020

Unique Index: -

Purpose: Stores general information about internship dates.

DEPARTMENTS TABLE

Column	Data Type	Null	Domain	Identification
Id	INT	N	Sequence#	Unique identifier for departments class
Name	VARCHAR(45)	N		Departments names

Example:

9	R&D

Unique Index: -

Purpose: Stores general information about departments.

INTEREST AREAS TABLE

Column	Data Type	Null	Domain	Identification
Id	INT	N	Sequence#	Unique identifier for Interest Areas class
Name	VARCHAR(45)	N		Interest Areas names

Example:

5	Artifical Intelligence

Unique Index: -

Purpose: Stores general information about interest areas.

OPTION TABLE

Column	Data Type	Null	Domain	Identification
id	INT	N	Sequence#	Unique identifier for a Date class
Free lunch	VARCHAR(45)	Y		Does Internship provide free-lunch?
Is paid	VARCHAR(45)	Y		Is the Internship paid?
Is full time	VARCHAR(45)	Y		Is the Internship mandatory?
Is mandatory	VARCHAR(45)	Y		Is the Internship mandatory?

Example:

1	Y	Y	N	N

Unique Index: -

Purpose: Stores general information about internship dates.

STUDENT INTERESTS TABLE

Column	Data Type	Null	Domain	Identification
Interests areas id	INT	N	Sequence#	Unique identifier for Student Interests class
Student student id	INT	N		Unique identifier for Student Interests class

Example:

4	14

Unique Index: -

Purpose: Stores general information about student interests.

INTERNSHIP TABLE

Column	Data Type	Null	Domain	Identification
Id	INT	N	Sequence#	Unique identifier for Internship class
Date id	INT	N		Unique identifier for Date class
Year of student	VARCHAR(45)	Y		Year of student
GPA	VARCHAR(45)	Y		GPA of student
City id	INT	N		Unique identifier for City class
Option id	INT	N		Unique identifier for option class
Departments id	INT	N		Unique identifier for Departments class
Contact id	INT	N		Unique identifier for Contact class
Company id	INT	N		Unique identifier for Company class
Student id	INT	N		Unique identifier for Student class

Example:

4	2	2.12	12	3	3	2	1	43	12

Unique Index: -

Purpose: Stores general information about internship.

STUDENT TABLE

Column	Data Type	Null	Domain	Identification
Student id	INT	N	Sequence#	Unique identifier for Student class
Name	VARCHAR(45)	N		Student name
Surname	VARCHAR(45)	N		Student surname
Gender	VARCHAR(45)	N		Student gender
Department	VARCHAR(45)	N		Student department
Phone number	VARCHAR(45)	Y		Student phone number
Email	VARCHAR(45)	Y		Student email
Show my phone	VARCHAR(45)	Y		Will student show his / her number?

Example:

23	Sönmez	Alev	M	IE	5073213245	sonmezalev@mail.com	Y

Unique Index: student_id, phone_number, email

Purpose: Stores general information about student.

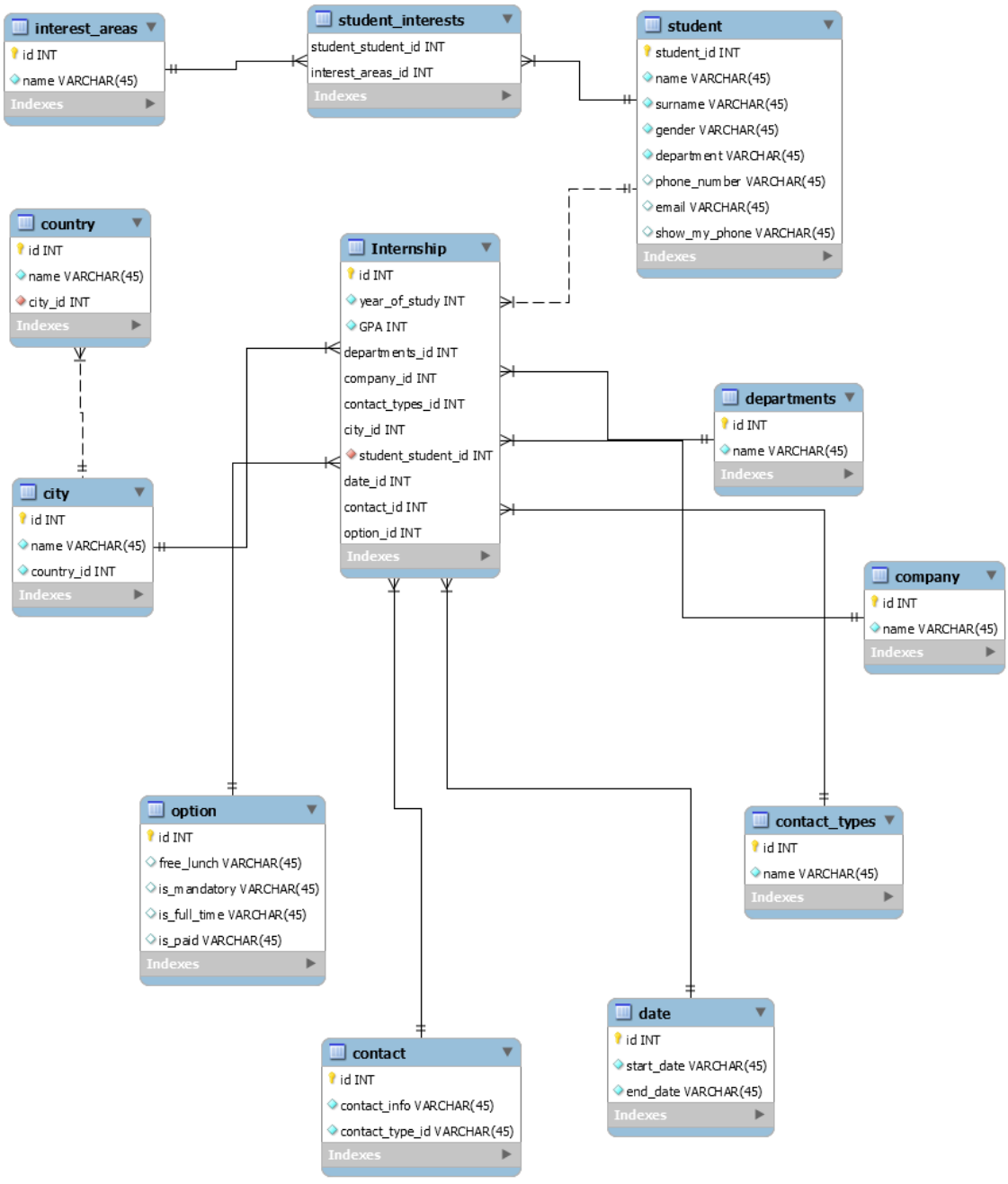
interest_areas
id INT
name VARCHAR(45)
Indexes
student_interests
student_student_id INT
interest_areas_id INT
Indexes
student
student_id INT
name VARCHAR(45)
surname VARCHAR(45)
gender VARCHAR(45)
department VARCHAR(45)
phone_number VARCHAR(45)
email VARCHAR(45)
show_my_phone VARCHAR(45)
Indexes
country
id INT
name VARCHAR(45)
city_id INT
Indexes
Internship
id INT
year_of_study INT
GPA INT
departments_id INT
company_id INT
contact_types_id INT
city_id INT
student_student_id INT
date_id INT
contact_id INT
option_id INT
Indexes
departments
id INT
name VARCHAR(45)
Indexes
city
id INT
name VARCHAR(45)
country_id INT
Indexes
company
id INT
name VARCHAR(45)
Indexes
option
id INT
free_lunch VARCHAR(45)
is_mandatory VARCHAR(45)
is_full_time VARCHAR(45)
is_paid VARCHAR(45)
Indexes
contact_types
id INT
name VARCHAR(45)
Indexes
date
id INT
start_date VARCHAR(45)
end_date VARCHAR(45)
Indexes
contact
id INT
contact_info VARCHAR(45)
contact_type_id VARCHAR(45)
Indexes

TABLE CREATION AND CONSTRAINT SCRIPTS

city

```
CREATE TABLE IF NOT EXISTS `mydb`.`city` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `country_id` INT NOT NULL,
  PRIMARY KEY (`id`, `country_id`),
  INDEX `fk_city_country1_idx` (`country_id` ASC) VISIBLE,
  CONSTRAINT `fk_city_country1`
    FOREIGN KEY (`country_id`)
    REFERENCES `mydb`.`country` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
```

company

```
CREATE TABLE IF NOT EXISTS `mydb`.`company` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
```

contact_types

```
CREATE TABLE IF NOT EXISTS `mydb`.`contact_types` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
```

country

```
CREATE TABLE IF NOT EXISTS `mydb`.`country` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
```

departments

```
CREATE TABLE IF NOT EXISTS `mydb`.`departments` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
```

date

```
CREATE TABLE IF NOT EXISTS `mydb`.`date` (
  `id` INT NOT NULL,
  `start_date` VARCHAR(45) NOT NULL,
  `end_date` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

contact

```
CREATE TABLE IF NOT EXISTS `mydb`.`contact` (
  `id` INT NOT NULL,
  `contact_info` VARCHAR(45) NOT NULL,
  `contact_type_id` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

interest_areas

```
CREATE TABLE IF NOT EXISTS `mydb`.`interest_areas` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
```

option

```
CREATE TABLE IF NOT EXISTS `mydb`.`option` (
  `id` INT NOT NULL,
  `free_lunch` VARCHAR(45) NULL,
  `is_mandatory` VARCHAR(45) NULL,
  `is_full_time` VARCHAR(45) NULL,
  `is_paid` VARCHAR(45) NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;
```

internship

```
CREATE TABLE IF NOT EXISTS `mydb`.`Internship` (
  `id` INT NOT NULL,
  `year_of_study` INT NOT NULL,
  `GPA` INT NOT NULL,
  `departments_id` INT NOT NULL,
  `company_id` INT NOT NULL,
  `contact_types_id` INT NOT NULL,
  `city_id` INT NOT NULL,
  `student_student_id` INT NOT NULL,
  `date_id` INT NOT NULL,
  `contact_id` INT NOT NULL,
  `option_id` INT NOT NULL,
  PRIMARY KEY (`id`, `departments_id`, `company_id`, `contact_types_id`, `city_id`, `date_id`,
`contact_id`, `option_id`),
  INDEX `fk_Internship_departments1_idx` (`departments_id` ASC) VISIBLE,
  INDEX `fk_Internship_company1_idx` (`company_id` ASC) VISIBLE,
  INDEX `fk_Internship_contact_types1_idx` (`contact_types_id` ASC) VISIBLE,
  INDEX `fk_Internship_city1_idx` (`city_id` ASC) VISIBLE,
  INDEX `fk_Internship_student1_idx` (`student_student_id` ASC) VISIBLE,
  INDEX `fk_Internship_date1_idx` (`date_id` ASC) VISIBLE,
  INDEX `fk_Internship_contact1_idx` (`contact_id` ASC) VISIBLE,
  INDEX `fk_Internship_option1_idx` (`option_id` ASC) VISIBLE,
  CONSTRAINT `fk_Internship_departments1`
    FOREIGN KEY (`departments_id`)
    REFERENCES `mydb`.`departments` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Internship_company1`
    FOREIGN KEY (`company_id`)
    REFERENCES `mydb`.`company` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Internship_contact_types1`
    FOREIGN KEY (`contact_types_id`)
    REFERENCES `mydb`.`contact_types` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
```

```
  CONSTRAINT `fk_Internship_city1`
    FOREIGN KEY (`city_id`)
    REFERENCES `mydb`.`city` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Internship_student1`
    FOREIGN KEY (`student_student_id`)
    REFERENCES `mydb`.`student` (`student_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Internship_date1`
    FOREIGN KEY (`date_id`)
    REFERENCES `mydb`.`date` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Internship_contact1`
    FOREIGN KEY (`contact_id`)
    REFERENCES `mydb`.`contact` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Internship_option1`
    FOREIGN KEY (`option_id`)
    REFERENCES `mydb`.`option` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

student

```
CREATE TABLE IF NOT EXISTS `mydb`.`student` (

  `student_id` INT NOT NULL,

  `name` VARCHAR(45) NOT NULL,

  `surname` VARCHAR(45) NOT NULL,

  `gender` VARCHAR(45) NOT NULL,

  `department` VARCHAR(45) NOT NULL,

  `phone_number` VARCHAR(45) NULL,

  `email` VARCHAR(45) NULL,

  `show_my_phone` VARCHAR(45) NULL DEFAULT 'NO',

  PRIMARY KEY (`student_id`),

  UNIQUE INDEX `student_id_UNIQUE` (`student_id` ASC) VISIBLE,
```

```
  UNIQUE INDEX `phone_number_UNIQUE` (`phone_number` ASC) VISIBLE,
  UNIQUE INDEX `email_UNIQUE` (`email` ASC) VISIBLE)
ENGINE = InnoDB
```

student_interests

```
CREATE TABLE IF NOT EXISTS `mydb`.`student_interests` (
  `interest_areas_id` INT NOT NULL,
  `student_student_id` INT NOT NULL,
  PRIMARY KEY (`interest_areas_id`, `student_student_id`),
  INDEX `fk_interest_areas_has_student_student1_idx` (`student_student_id` ASC) VISIBLE,
  INDEX `fk_interest_areas_has_student_interest_areas1_idx` (`interest_areas_id` ASC) VISIBLE,
  CONSTRAINT `fk_interest_areas_has_student_interest_areas1`
    FOREIGN KEY (`interest_areas_id`)
    REFERENCES `mydb`.`interest_areas` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_interest_areas_has_student_student1`
    FOREIGN KEY (`student_student_id`)
    REFERENCES `mydb`.`student` (`student_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
```

SECURITY

SQL INJECTION

In the database model whereArgs used to pass arguments to a where statement. Statements like where: “id = ${student.id}” is not used. Two parameters, where and whereArgs, are used together to substitute values in the where clause while preventing SQL injection. It returns the matching entries as a list.

ENCRYPTED PASSWORD

Md5 method has used for password security. The packages and method is below

```
import 'dart:convert';i

mport 'package:convert/convert.dart';

import 'package:crypto/crypto.dart' as crypto;


///Generate MD5 hash
generateMd5(String data) {
    var content = new Utf8Encoder().convert(data);
    var md5 = crypto.md5;
    var digest = md5.convert(content);
    return hex.encode(digest.bytes);
}
```

DART CODE EXAMPLES

```
Future<bool> login(User user) async {

 var dbClient = await db;

 int number = Sqflite.firstIntValue(await dbClient

   .rawQuery('SELECT COUNT(*) FROM user WHERE mail = "${user.mail}" AND
pass = "${user.pass}"'));

 if (number > 0) {

  return true;

 } else {

  return false;

 }

}
```

```
Future<List<Internship>> getInternship(Internship internship) async{
 var dbClient = await db;
 final String getInternship =
   "SELECT internship.id AS id, student.gender AS gender, start_date,
end_date, company.name AS company_id, contact_info, free_launch, "
   "year_of_student, gpa, is_paid, is_full_time, is_mandatory, city.name AS
city_id, country.name AS country_id, "
   "departments.name AS departments_id, contact_types.name AS
contact_types_id, internship.student_id AS student_id "
   "FROM internship, city, country, departments, contact_types, company,
student "
   "WHERE internship.city_id = city.id AND student.student_id =
internship.student_id AND country.id = city.country_id "
   "AND internship.departments_id = departments.id AND
internship.contact_types_id = contact_types.id "
   "AND internship.company_id = company.id AND "
   "country.name LIKE '%${internship.country ?? ""}%' AND "
   "city.name LIKE '%${internship.city ?? ""}%' AND "
   "departments.name LIKE '%${internship.department ?? ""}%' AND "
   "company.name LIKE '%${internship.company}%' AND  "
   "gpa <= ${internship.gpa ?? "0"} AND "
   "year_of_student LIKE '%${internship.yearOfStudent ?? ""}%' AND "
   "is_paid LIKE '%${internship.isPaid ?? ""}%' AND "
   "is_full_time LIKE '%${internship.isFullTime ?? ""}%' AND "
   "is_mandatory LIKE '%${internship.isMandatory ?? ""}%' AND "
   "free_launch LIKE '%${internship.freeLunch ?? ""}%';";
 List<Map> data = await dbClient.rawQuery(getInternship);
 List<Internship> result = List<Internship>();
 print(data);
 for(Map records in data){
  Internship x = Internship.fromMap(records);
  x.studentData = await getStudentViaID(x.student);
```

```
  result.add(x);
 }
 return result;
}
```

```
var countryResult = await dbClient.query("country", where: "name =
'${internship.country}'", limit: 1);

Country myCountry = countryResult.map((data) =>
Country.fromMap(data)).toList()[0];

add.country = myCountry.id;
```

SCREENSHOTS

Figure 4.Homepage

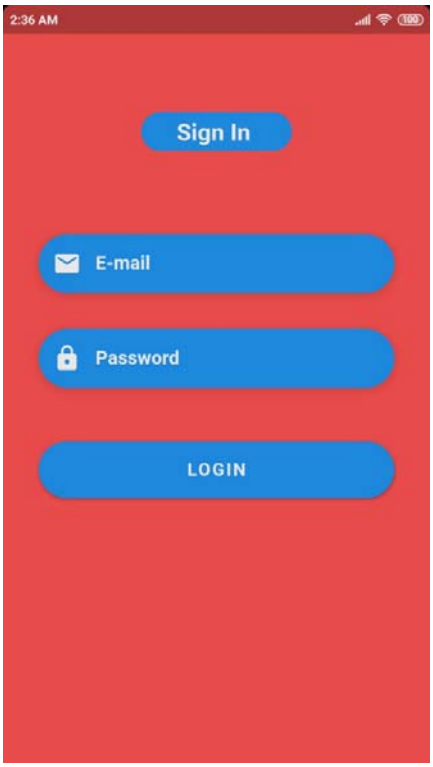

Figure 5.Sign In

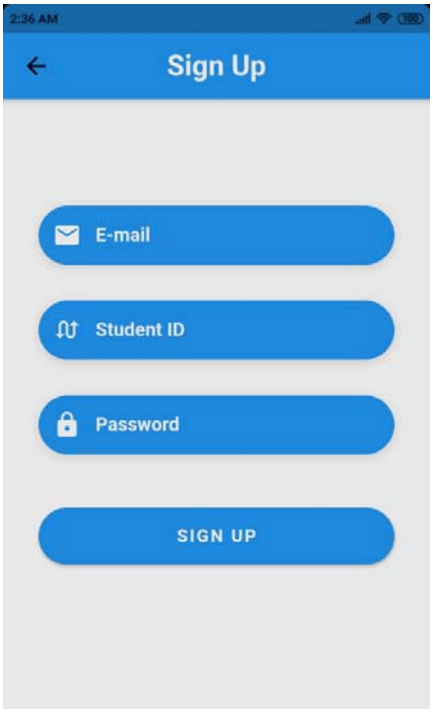

Figure 6.Sign Up

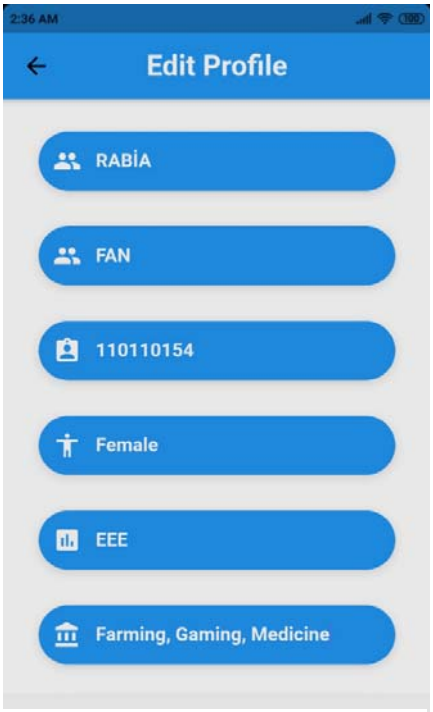

Figure 7.Edit Profile

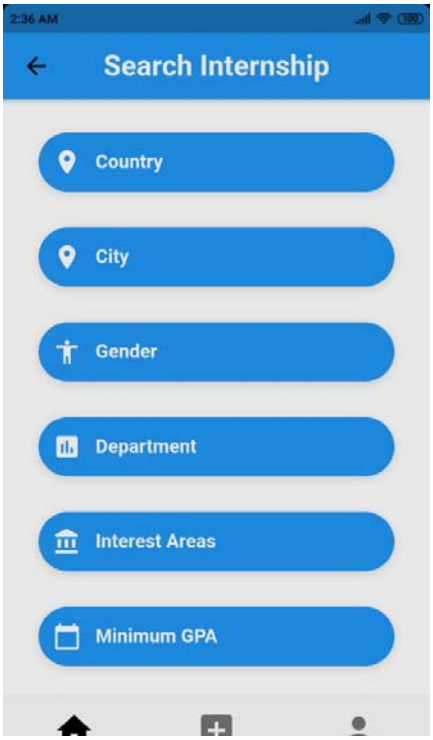

Figure 9.Search Internship page

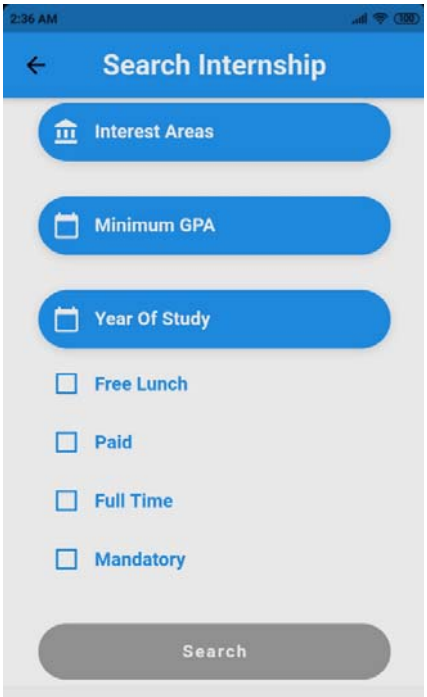

Figure 8.Search Internship

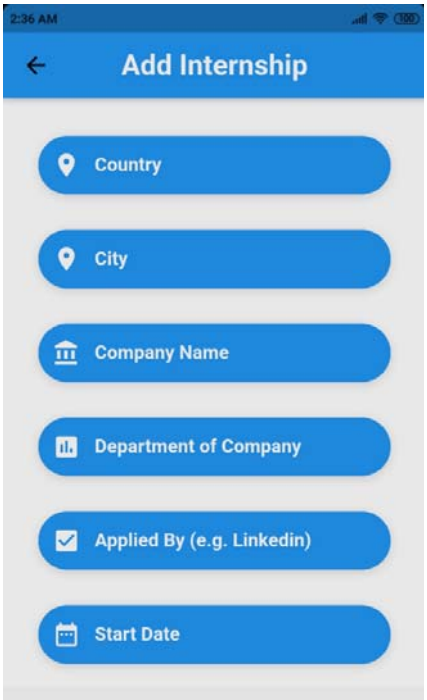

Figure 11.Add Internship

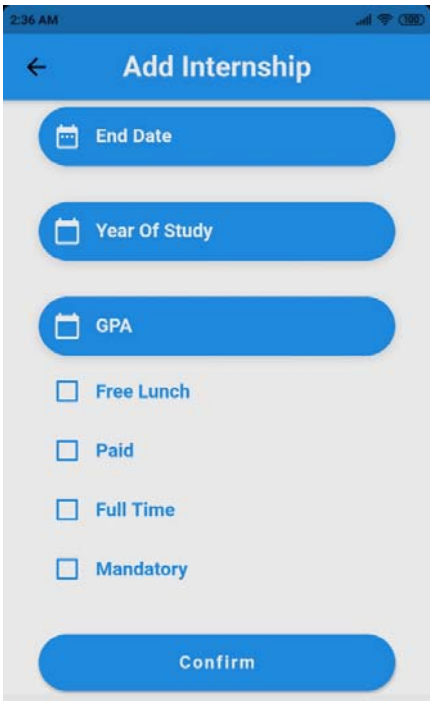

Figure 10.Add Internship page