1. Overview

The Career Guidance System is designed to help students identify suitable career paths and connect with teachers for guidance. The system provides functionalities for user registration, profile management, scheduling meetings, and viewing teacher/student lists, all underpinned by a Model-View-Controller (MVC) architecture. This design accommodates the .NET (ASP.NET Core MVC) development framework.

2. Core Modules Description

1. **User Management**: Responsible for the creation, modification, and retrieval of user information.
2. **Teacher Management**: Manages the profiles and availability of teachers.
3. **Student Management**: Manages student profiles and their meeting requests.
4. **Meeting Scheduling**: Handles scheduling and managing meetings between students and teachers.
5. **Profile Viewing**: Allows students to view teacher profiles and vice versa.

3. Module-level Design

3.1 User Management

Responsible for the creation, modification, and retrieval of user information.

- **Controller**: UserController
    - registerUser()
    - login()
    - logout()
    - getUserProfile()
- **Service**: UserService
    - Handles logic for user registration, authentication, and profile management.
- **Model**: User Entity
    - **Attributes**:
        - userId (PK)
        - username

        - password
        - email
        - role

3.2 Teacher Management

Manages the profiles and availability of teachers.

- **Controller**: TeacherController
    - createTeacherProfile()
    - updateTeacherProfile()
    - getTeacherById()
    - getAllTeachers()
- **Service**: TeacherService
    - Handles logic for creating, updating, and retrieving teacher profiles.
- **Model**: Teacher Entity
    - **Attributes**:
        - teacherId (PK)
        - name
        - qualification
        - experience
        - availability

3.3 Student Management

Manages student profiles and their meeting requests.

- **Controller**: StudentController
    - createStudentProfile()
    - updateStudentProfile()
    - getStudentById()
    - getAllStudents()

    - requestMeeting()
- **Service**: StudentService
    - Handles logic for creating, updating, and retrieving student profiles and managing meeting requests.
- **Model**: Student Entity
    - **Attributes**:
        - studentId (PK)
        - name
        - email
        - desiredCareer
        - meetingRequests

3.4 Meeting Scheduling

Handles scheduling and managing meetings between students and teachers.

- **Controller**: MeetingController
    - scheduleMeeting()
    - cancelMeeting()
    - getMeetingById()
    - getAllMeetings()
- **Service**: MeetingService
    - Handles logic for scheduling, updating, and retrieving meetings.
- **Model**: Meeting Entity
    - **Attributes**:
        - meetingId (PK)
        - studentId (FK)
        - teacherId (FK)
        - meetingDateTime

        - status

3.5 Profile Viewing

Allows students to view teacher profiles and vice versa.

- **Controller**: ProfileController
    - viewTeacherProfile()
    - viewStudentProfile()
- **Service**: ProfileService
    - Handles logic for viewing profiles.
- **Model**: Profile Entity
    - **Attributes**:
        - profileId (PK)
        - userId (FK)
        - profileType

4.Database schema

4.1 Table definitions

User Table:

```
CREATE TABLE User(
userId INT AUTO_INCREMENT PRIMARY KEY,
username VARCHAR(50) UNIQUE,
password VARCHAR(255),
email VARCHAR(100),
role ENUM('ADMIN', 'STUDENT')
);
```

Teacher Table:

```
CREATE TABLE Teacher(
teacherId INT AUTO_INCREMENT PRIMARY KEY,
```

```
  name VARCHAR(100),
  qualification VARCHAR(255),
  experience INT,
  availability BOOLEAN
);
```

Student Table:

```
 CREATE TABLE Student(
  studentId INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100),
  email VARCHAR(100),
  desiredCareer VARCHAR(100),
  meetingRequests TEXT
);
```

Meeting Table:

```
 CREATE TABLE Meeting(
  meetingId INT AUTO_INCREMENT PRIMARY KEY,
  studentId INT,
  teacherId INT,
  meetingDateTime DATETIME,
  status ENUM('SCHEDULED', 'CANCELLED'),
  FOREIGN KEY (studentId) REFERENCES Student(studentId),
  FOREIGN KEY (teacherId) REFERENCES Teacher(teacherId)
);
```

Profile Table:

```
 CREATE TABLE Profile(
  profileId INT AUTO_INCREMENT PRIMARY KEY,
```

```
    userId INT,
    profileType ENUM('TEACHER', 'STUDENT'),
    FOREIGN KEY (userId) REFERENCES User(userId)
);
```

6.Conclusion

By adhering to this design document, you can successfully develop and implement a Career Guidance System using ASP.NET Core MVC. This system will empower students to explore and identify suitable career paths based on their individual interests, skills, and preferences. Additionally, it will facilitate seamless interaction between students and teachers, ensuring that students receive the guidance and support they need to make informed career decisions.