PROJECT PROPOSAL:

Teacher Feedback System

CS232

Sir. Mohsin Zafar

Group 6:
Atesam Abdullah
(2021114)
Hamza Hasan
(2021197)
Muhammad Musa
(2021421)
Muhammad Zulfiqar Ali
(2021493)

Synopsis

The main objective of the Teacher Feedback System is to provide the students of the institute access to an interface where they can give feedback about their courses and course instructors. The system is built for the administrative end thus the administrator has access to it. This project is developed to provide feedback in a quick , easy, and incognito manner.

User Story

- As a user, I can be a student, admin, or faculty.
- As a user, I have my own login id and password.
- As a user, I can have different access rights.
- As an admin, I will be able to view the statistics generated by the system.
- As a faculty, I will provide access to students for giving feedback.
- As a student, I will fill in the feedback form.

Functional Requirements

- The system must allow users to login to their account.
- The system allows students to give feedback on each subject and faculty.
- The system should provide a structured result to the admin.
- The system must show updated results of feedback to the admin.
- The system must allow faculty to ensure all students submit feedback.

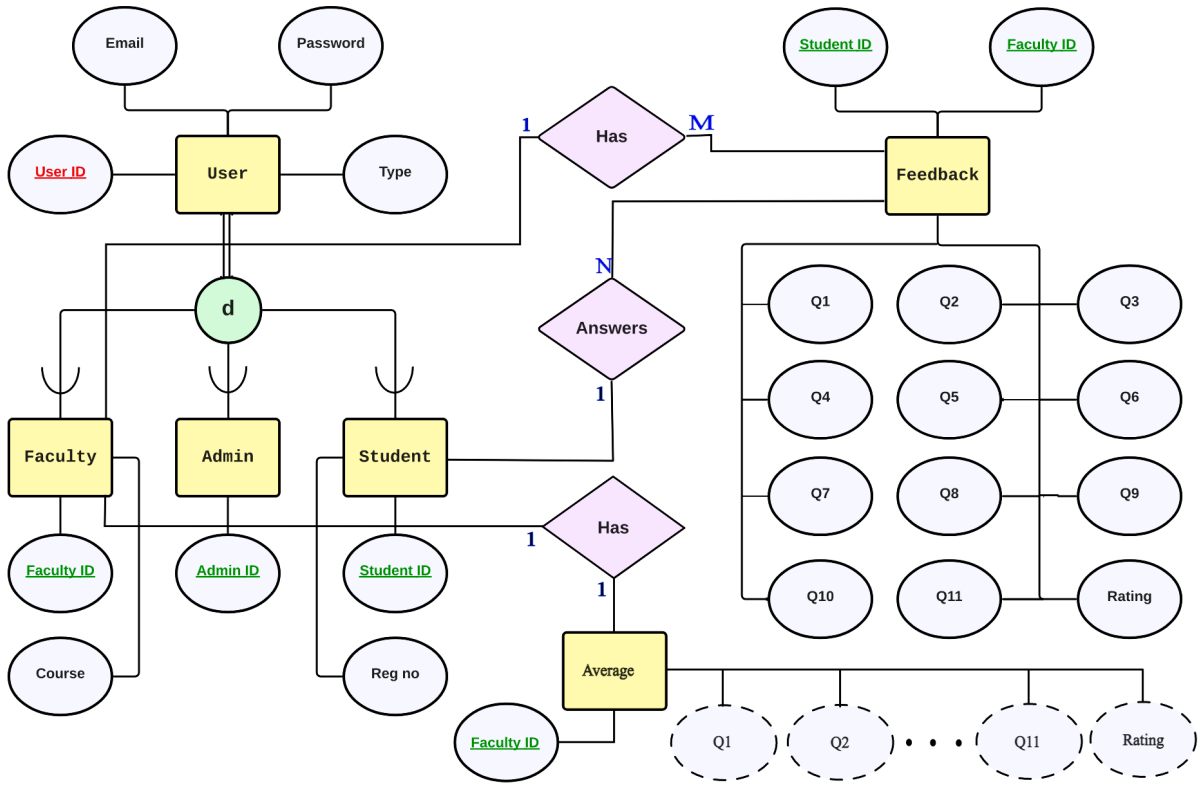
Stack

Operating System: Windows
Front-end: HTML/CSS/Javascript
Back-end: Flask, SQLAlchemy
Database: SQLite

Github Repo Link

https://github.com/hamzahasann/TeacherFeedbackSystem

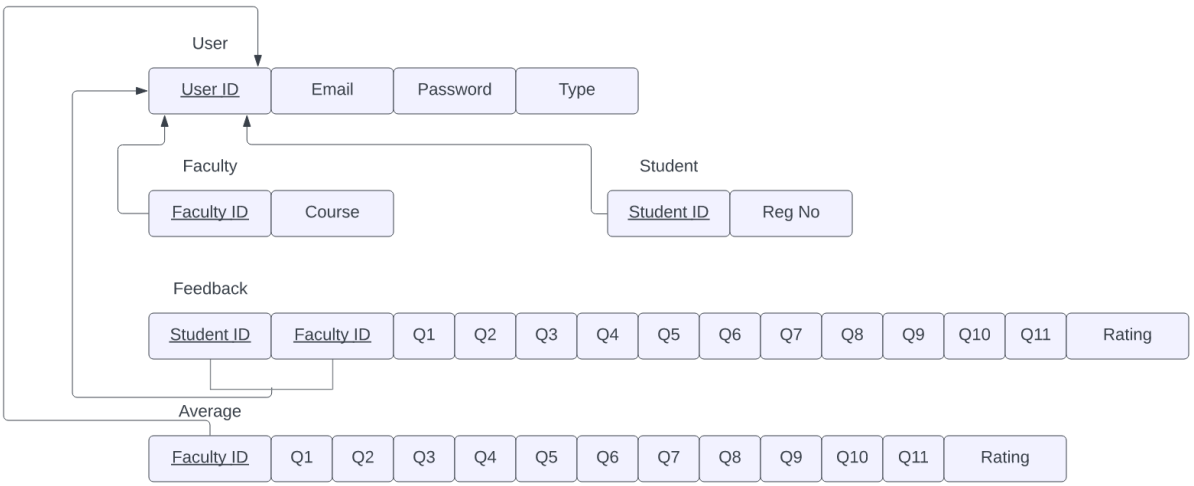
EERD:



Assumptions:

1) There can be only 3 type of users: Faculty, Admin and Student.
2) 1 Faculty User has only 1 course.
3) Feedback form is same for all courses.

Relational Schema:



Relations:

I. User,Faculty, Student, Feedback, Average

Primary Keys:

I. UserID(User)

II. FacultyID(Faculty)

III. StudentID(Student)

IV. [StudentID,FacultyID](Feedback)

V. FacultyID(Average)

Foreign Keys:

I. FacultyID(Faculty) references UserID(User)

II. StudentID(Student) references UserID(User)

III. StudentID(Feedback) references UserID(User)

IV. FacultyID(Feedback) references UserID(User)

V. FacultyID(Average) references UserID(User)

Dummy Data:

Users:

	U_ID	NAME	EMAIL	PASSWORD	TYPE
1	0	admin	admin@giki.edu.pk	0000	admin
2	1	Tom	tom@giki.com	1234	faculty
3	2	Holland	holland@giki.com	1234	faculty
4	3	Batman	batman@giki.com	1234	faculty
5	4	Moosa	moosa@giki	1111	student
6	5	Ahmed	ahmed@giki.com	1111	student
7	6	Atesam	atesam@giki.com	1111	student
8	7	Hamza	hamza@giki.com	1111	student
9	8	Zulfiqar	zulfiqar@giki.com	1111	student
10	9	Qari sahab	qari@giki.com	1111	student

Students:

	S_ID	RE...
1	4	2021421
2	5	2021123
3	6	2021114
4	7	2021197
5	8	2021442
6	9	2021198

Faculty:

	F_ID	COURSE
1	1	CS 232
2	2	CS 221
3	3	MT 201

Average:

	S_ID	F_ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	RATING	COMMENTS
1	4	1	3	4	5	1	2	3	4	5	1	2	3	6	OK
2	4	2	2	4	5	1	2	1	4	1	1	2	3	3	OK
3	4	3	5	5	5	1	2	3	5	5	1	2	3	9	Good
4	5	1	3	4	5	1	2	3	4	5	1	2	3	6	OK
5	5	2	5	3	5	3	5	3	5	3	5	3	5	3	Good
6	5	3	5	5	5	1	2	3	5	5	1	2	3	6	Good
7	6	1	3	4	5	1	2	3	4	5	1	2	3	6	OK
8	6	2	2	4	5	1	2	1	4	1	1	2	3	3	OK
9	6	3	5	5	5	1	2	3	5	5	1	2	3	9	Good
10	7	1	3	4	5	1	2	3	4	5	1	2	3	6	OK
11	7	2	2	4	5	1	2	1	4	1	1	2	3	3	OK
12	8	2	5	5	5	1	2	3	5	5	1	2	3	9	Good
13	8	1	3	4	5	1	2	3	4	5	1	2	3	6	OK
14	9	3	2	4	5	1	2	1	4	1	1	2	3	3	OK
15	9	2	5	5	5	1	2	3	5	5	1	2	3	9	Good

Basic SQL:

```
--Check if user is signedin
select * from users where email='admin@giki.edu.pk' and password='0000' and type='admin' ;

--list of courses student enrolled in
select course from feedback
inner join faculty on faculty.f_id=feedback.f_id
where s_id = 5;
```

```
--regno of student
select regno from student where s_id=5;

--fetch course taught by teacher
select course from faculty where f_id=2;

--distinct courses in feedback
select distinct course from feedback
inner join faculty on faculty.f_id=feedback.f_id;

--get f_id for a course
select f_id from faculty where course ='MT 201';

--check if student has been granted access
select case
         when exists (select * from feedback where s_id = 5 and f_id = 2)
         then 'Yes'
         else 'No'
         end
from dual;

--update feedbcak b/c s_id,f_id were already inserted
update feedback set q1=5,q2=3,q3=5,q4=3,q5=5,q6=3,q7=5,q8=3,q9=5,q10=3,q11=5,rating=3,comments='Good'
where s_id=5 and f_id=2;

----register student if not registered
--find s_id from reg
select s_id from student where regno =2021114;
--check if student has been granted access
select * from feedback where s_id = 5 and f_id = 2;
--insert if not prsent
insert into feedback(s_id,f_id) values(5,2);



----Function to update course average
create or replace procedure update_average(Tcourse faculty.course%type)
AS
         Tf_id faculty.f_id%type;
         temp number;

         Tq1 number:=0;
         Tq2 number:=0;
         Tq3 number:=0;
         Tq4 number:=0;
         Tq5 number:=0;
         Tq6 number:=0;
         Tq7 number:=0;
         Tq8 number:=0;
         Tq9 number:=0;
         Tq10 number:=0;
         Tq11 number:=0;
         Trating number:=0;
BEGIN
         --find f_id for course
         select f_id into Tf_id from faculty where course = Tcourse;

         --find averages for course
         select
avg(q1),avg(q2),avg(q3),avg(q4),avg(q5),avg(q6),avg(q7),avg(q8),avg(q9),avg(q10),avg(q11),avg(rating)
         into Tq1,Tq2,Tq3,Tq4,Tq5,Tq6,Tq7,Tq8,Tq9,Tq10,Tq11,Trating
         from feedback
         where f_id = Tf_id;

         --check if f_id/course already in
         select case
         when exists (select 1
                        from average
                        where f_id = Tf_id)
         then 1
         else 0
         end into temp
```

```
        from dual;

        --if not exist create record
        if(temp=0) then
        insert into average(f_id) values(Tf_id);
        end if;

        --update average table
        update average
        set q1=Tq1,q2=Tq2,q3=Tq3,q4=Tq4,q5=Tq5,q6=Tq6,q7=Tq7,q8=Tq8,q9=Tq9,q10=Tq10,q11=Tq11,rating=Trating
        where f_id=Tf_id;

END;
/

--Call update_average when admin requests data
begin
        update_average('MT 201');
end;
/
select * from average;
--fetch average from table to show admin
create view admin_view as (select * from average where f_id=3);
```

Advanced SQL:

```
--Check if user is signedin
select * from users where email='admin@giki.edu.pk' and password='0000' and type='admin' ;

--list of courses student enrolled in
select course from feedback
inner join faculty on faculty.f_id=feedback.f_id
where s_id = 5;

--regno of student
select regno from student where s_id=5;

--fetch course taught by teacher
select course from faculty where f_id=2;

--distinct courses in feedback
select distinct course from feedback
inner join faculty on faculty.f_id=feedback.f_id;

--get f_id for a course
select f_id from faculty where course ='MT 201';

--check if student has been granted access
select case
        when exists (select * from feedback where s_id = 5 and f_id = 2)
        then 'Yes'
        else 'No'
        end
from dual;

--update feedbcak b/c s_id,f_id were already inserted
update feedback set q1=5,q2=3,q3=5,q4=3,q5=5,q6=3,q7=5,q8=3,q9=5,q10=3,q11=5,rating=3,comments='Good'
where s_id=5 and f_id=2;

----register student if not registered
--find s_id from reg
select s_id from student where regno =2021114;
--check if student has been granted access
select * from feedback where s_id = 5 and f_id = 2;
--insert if not prsent
insert into feedback(s_id,f_id) values(5,2);
```

```
----Function to update course average
create or replace procedure update_average(Tcourse faculty.course%type)
AS
        Tf_id faculty.f_id%type;
        temp number;

        Tq1 number:=0;
        Tq2 number:=0;
        Tq3 number:=0;
        Tq4 number:=0;
        Tq5 number:=0;
        Tq6 number:=0;
        Tq7 number:=0;
        Tq8 number:=0;
        Tq9 number:=0;
        Tq10 number:=0;
        Tq11 number:=0;
        Trating number:=0;
BEGIN
        --find f_id for course
        select f_id into Tf_id from faculty where course = Tcourse;

        --find averages for course
        select
avg(q1),avg(q2),avg(q3),avg(q4),avg(q5),avg(q6),avg(q7),avg(q8),avg(q9),avg(q10),avg(q11),avg(rating)
        into Tq1,Tq2,Tq3,Tq4,Tq5,Tq6,Tq7,Tq8,Tq9,Tq10,Tq11,Trating
        from feedback
        where f_id = Tf_id;

        --check if f_id/course already in
        select case
        when exists (select 1
                        from average
                        where f_id = Tf_id)
        then 1
        else 0
        end into temp
        from dual;

        --if not exist create record
        if(temp=0) then
        insert into average(f_id) values(Tf_id);
        end if;

        --update average table
        update average
        set q1=Tq1,q2=Tq2,q3=Tq3,q4=Tq4,q5=Tq5,q6=Tq6,q7=Tq7,q8=Tq8,q9=Tq9,q10=Tq10,q11=Tq11,rating=Trating
        where f_id=Tf_id;

END;
/

--Call update_average when admin requests data
begin
        update_average('MT 201');
end;
/
select * from average;
--fetch average from table to show admin
create view admin_view as (select * from average where f_id=3);
```