UEP Feedback – Employee & Manager

Problem Statement

Need a system to collect UEP feedback from employees to understand how the program helped individual to work better in project.

User Stories

As a unified engineer I should be able to provide feedback about unified engineer so that I can share my viewpoints on how the program helped me to improve my productivity

Acceptance criteria

- System should provide list of parameters given below against which the user can share his/her feedback.
- User can provide rating against each parameter. Rating should b/n 1 to 5 (1 being poor and 5 being excellent). This is mandatory.
- User should also provide comments for each parameter to describe on how the program helped him/her to improve.
- System should capture the information about the details like empid, name, and project id of the person who gives the feedback for analysis purpose. System should NOT ask the user to enter the details. These should be captured based on the user login
- As a unified engineer I should be able to view the feedback that I have given
- Manager should be able to see his team member UEP feedbacks that are given
- Manager should be able to give rating on individual

Requirement

Employee must be able to login.

Employee must be able to submit the feedback according to the category.

Employee must be able to provide the ratings.

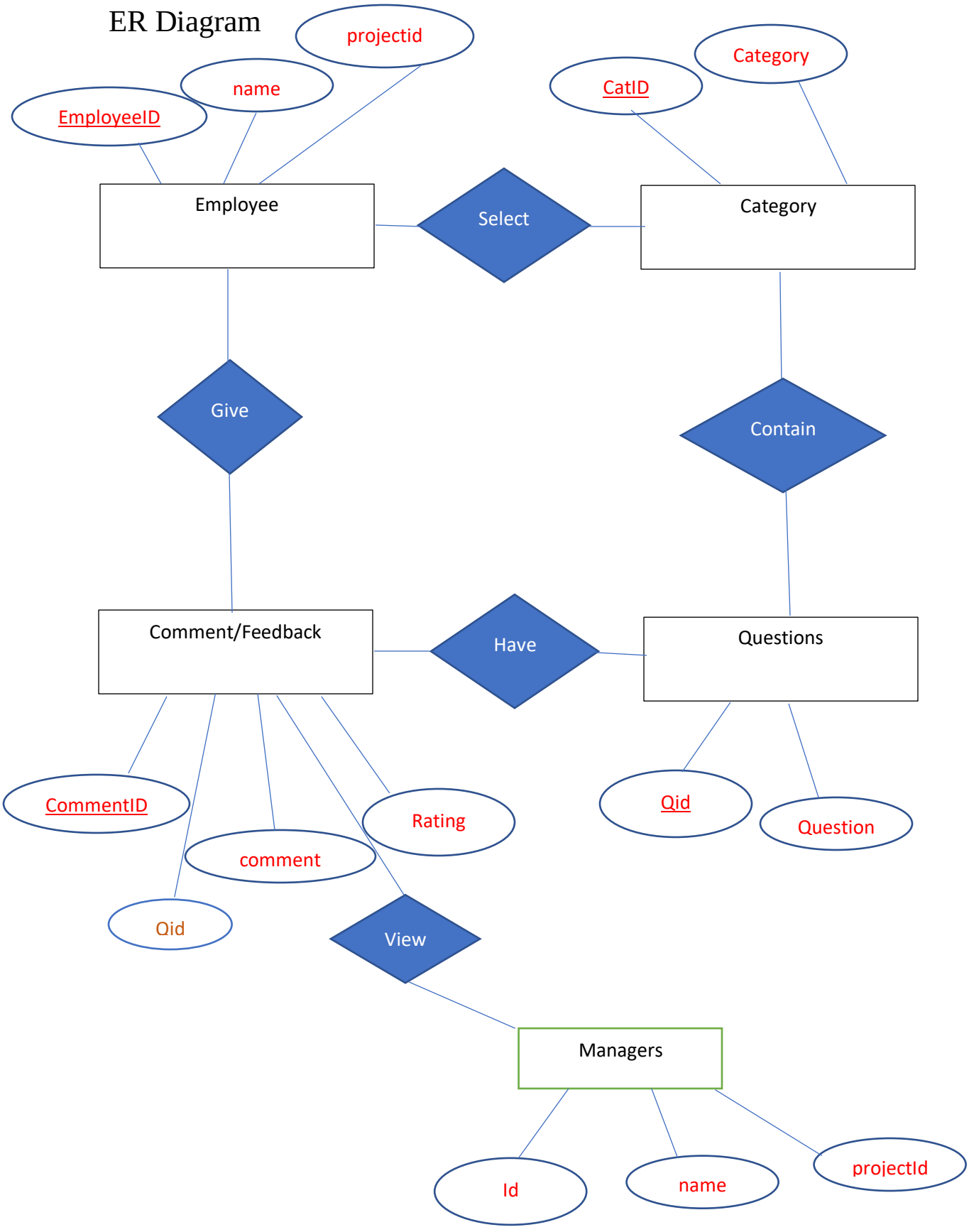
Employee must be able to view feedback page which is submitted by him/her.

Manager should be able to login

Manager should be able to see his team member UEP feedbacks that are given

Manager should be able to give rating on individual

ER Diagram


projectid
name
EmployeeID
Employee
Select
CatID
Category
Category
Give
Contain
Comment/Feedback
Have
Questions
CommentID
Rating
comment
Qid
View
Qid
Question
Managers
Id
name
projectId

Microservices

- User Management
- Employee
- Feedback
- Manager

Table Structure

- Employee
- Category
- Questions
- Feedback
- Manager
- Review

```
CREATE TABLE Manager (
id int IDENTITY(1,1) PRIMARY KEY,
[name] varchar(255),
[contactno] varchar(20),
projectid varchar(255),
[address] varchar(250),
[role] varchar(250)
)

CREATE TABLE Category (
CategoryId int IDENTITY(1,1) PRIMARY KEY,
CategoryName varchar(255)
)
CREATE TABLE Questions (
QuestionId int IDENTITY(1,1) PRIMARY KEY,
Question nvarchar(max),
CategoryId int FOREIGN KEY REFERENCES Category(CategoryId)
)
CREATE TABLE FeedBack (
```

```
FeedBackId int IDENTITY(1,1) PRIMARY KEY,
Feedback nvarchar(max),
Rating int,
EmpId int,
QuestionId int FOREIGN KEY REFERENCES Questions(QuestionId)
)

 CREATE TABLE Review(
  reviewId int IDENTITY(1,1) PRIMARY KEY,
  [Question]  nvarchar(max),
  Feedbacks nvarchar(max),
  Rating int,
  EmpId int
  )

CREATE TABLE Employee (
id int IDENTITY(1,1) PRIMARY KEY,
[name] varchar(255),
[contactno] varchar(20),
projectid varchar(255),
[address] varchar(250),
[role] varchar(250)
)
```

API Design

User Management
POST/api/Users/Login
POST/api/Users/Register
POST/api/Users/UpdatePassword

Employee
Get /api /Employee/GetEmployeeById
Post /api/Employee/PostEmployeeDetails
Get /api/Employee/GetAllByProjectId
PUT/api/Employee/UpdateEmployeeById

Category
Get /api / Category/GetCategory
POST/api/Category/PostCategory
Put/api/Category/UpdateCategory
Delete/api/Category/Delete

Questions
Get /api / Questions/GetByCategoryId
Put /api/ Questions /UpdateQuestion
Post /api/ Questions/PostQuestion
Delete/api/Questions/Delete

FeedBack
Get /api/ FeedBack /GetFeedbackByEmpId
Post /api/ FeedBack/PostFeedback

Managers
Get/api/ Managers/GetDetailsById
Post/api/ Managers/PostManagerDetails
Put/api/ Managers/UpdateDetailsById