Title: Question, Answer & Vote

Description: The application serves as a platform for users to ask and answer questions, and, through membership and active participation, to vote questions and answers up or down similar to Reddit/Stack overflow and edit questions and answers.

Users of application can earn reputation points and "badges" for example, a person is awarded 10 reputation points for receiving an "up" vote on a question or an answer to a question, and can receive badges for their valued contributions, which represents a gamification of the traditional Q&A website. Users unlock new privileges with an increase in reputation like the ability to vote, comment, and even edit other people's posts.

User Stories:

1. The application should have option to login to system.
2. The application should allow to register.
3. The application should allow to ask a question.
4. Should allow answer the question.
5. The application should show answers for the particular question

1. Feature: Login

Scenario (Valid):
When user enters valid email as "<email>"
And password as "<password>"
And credentials entered are correct
Then payload given
{
message: ”user logged in successfully”
}

Input
{
“username”:mail@naveen.com,
“password”: ”keepguessing”
}
Method Type: POST
Return Status:201

Scenario Outline (invalid):
When user entered unregistered email as "<email>"
And password as "<password>"
Then payload given
{
message: ”Sorry invalid credentials”
}

Input
{
“username”:nouser@naveen.com,
“password”: ”wrongpassword”
}
Method Type: POST
Return Status:401
URL: http://localhost:4000/login

2. Feature: Registration

Scenario Outline: Successful Registration
When user enters valid name as "<name>"
And valid email as "<email>"
And email is not already registered
And password as "<password>"
Then payload given

```
{
        message: ”User Registered Successfully”,
        “registration-name”:”Naveen”
}
```

Input

```
{
        “registration-name”:”Naveen”,
        “username”: mail@naveen.com,
        “password”: ”keepguessing”,

}
```

Method Type: POST
Return Status: 201
URL: http://localhost:4000/register

3. Feature: Ask Question

Scenario Outline (Valid): User asks question
When user enters valid question title as "<title>"
And valid question body as "<body>"
Then payload returned

```
{
        “message”:” Question posted successfully”,
        “question-id”:101

}
```

Input

```
{
    “user-details”: {
        “username”:mail@naveen.com,
        “password”:” keepguessing”
    },
    “question”: {
        “title:”:”how to write nodejs program”,
        “body”:”I’m trying to build rest api with node and express js, I get error as express is
        missing”
    }
}
```

Note: The application should take valid user details, and from the server side it must be validated and then the question to be posted.
Method Type: POST
Return Status: 201
URL: http://localhost:4000/question

4. Feature: Answer a question

4.1 *Background:*

Given user is logged in
And the question id is given

Scenario Outline (valid case): User wants to answer a question

When user wants to give answer "<answer>" for question "<question>"
And has not already answered the question
Then post the answer
And payload returned

```
{
        "message": "answer posted successfully",
        "question-id":101

}
```

Input

```
{
    "user-details": {
        "username":mail@naveen.com,
        "password":" keepguessing"
    },
    "question": {
        "question-id":101,
        "answer": "you need to check if the express is installed, check package.json file under dependencies, if express exists, if not execute : npm install express"
    }
}
```

Note: The application should take valid user details, and from the server side it must be validated and then the answer to be posted.
Method Type: POST
Return Status: 201
URL: http://localhost:4000/question /{questionid}/answer

4.2: *Background:*

Given user is logged in
And the question id is given

Scenario Outline (invalid case): User wants to answer a question

When user wants to give answer "<answer>" for question "<question>"
And has not already answered the question
Then post the answer
And payload returned

```
{
        "message": "sorry invalid user details",
        "question-id":101

}
```

Input

```
{
    "user-details": {
        "username":wrongname@naveen.com,
        "password":"wrongpassword"
    },
    "question": {
        "question-id":101,
        "answer": "some solution for the question"
    }
}
```

Note: The application should take valid user details, and from the server side it must be validated and then the answer to be posted, failing which the answer should not be posted.
Method Type: POST
Return Status: 201
URL: http://localhost:4000/question /{questionid}/answer

4.3 *Background:*

Given user is logged in
And the question id is given

Scenario Outline (valid case): User wants to updated the answer to a question

When user wants to update answer "<answer>" for question "<question>"
And has already answered the question
Then update the answer
And payload returned

```
{
        "message": "answer updated successfully",
        "question-id":101

}
```

Input

```
{
    "user-details": {
        "username":mail@naveen.com,
        "password":" keepguessing"
    },
    "question": {
        "question-id":101,
        "answer": "you need to check if the express is installed, check package.json file under dependencies, if express exists, if not execute : npm install express, and make sure your cmd line is where package.json"
    }
}
```

Note: The application should take valid user details, and from the server side it must be validated and then the answer to be updated.

Method Type: PUT/PATCH

Return Status: 200

URL: http://localhost:4000/question /{questionid}/answer

5. Feature: Get answers for all the questions

Background:
Given user is logged in
And the question id is given

Scenario Outline (valid case): User wants to get answers for a question
When user wants to get answers for a question
And payload returned

```
{
        {
                "question": "who invented java",
                "answers":[
                        {
                                "answer":"games gostling",
                        },
                        {
                                "answer":"peter james",
                        }
                ]
        },
        {
                "question": "who invented c++",
                "answers":[
                        {
                                "answer":"games gostling",
                        },
                        {
                                "answer":" Bjarne Stroustrup",
                        }
                ]
        },
}
```

Input

```
{
        "user-details": {
                "username":mail@naveen.com,
                "password":" keepguessing"
        }
}
```

Note: The application should take valid user details, and from the server side it must be validated and then the answers for the question to be shown

Method Type: GET
Return Status: 200
URL: http://localhost:4000/question /{questionid}