CourseForces

An online course assessment platform

Group 22
Shubh Agrawal (190828)
Tarun Kanodia (190902)

Mission and Vision

We have come up with CourseForces, an ed-tech app, in which users can register and then create courses and quizzes and invite other users as a student or professors and then conduct quizzes. Users registered as students can attempt the quiz and a professor in the course can evaluate their performance. In our app, we bring together the feature that a user can take different roles in different courses. A user who is taking some course as a professor can be a student in some other course and expand his/her knowledge. In a course, a professor can create quizzes with any combination of questions types. Question types can be single correct MCQ, multi correct MCQ, and subjective type. A user can test his knowledge based on the quiz.

Overview

CourseForces is a portal where professors can create courses, add students, conduct quizzes for the students, and then get some analytics. Quizzes can contain a variety of types of questions like subjective questions as well as objective questions with the option of having Single correct(one option) or Multi correct(one or more than one option(s) correct) questions. We also have email notifications sending system.

Entities

MyUser: These are the people associated with a particular course: professors, teaching assistants, or students. The role of a person is defined by the role attribute of the relationship "user_in_course" with the entity Course. MyUser has a one-to-many relationship with the Notification identity as well as the QuizAttempt entity.

Course: It contains information about the courses offered by professors. It has various attributes like course_code, course_name, offered_year, etc. It has a one to many relationship with the Quiz entity since a course can have many quizzes while a particular quiz can belong to only one course.

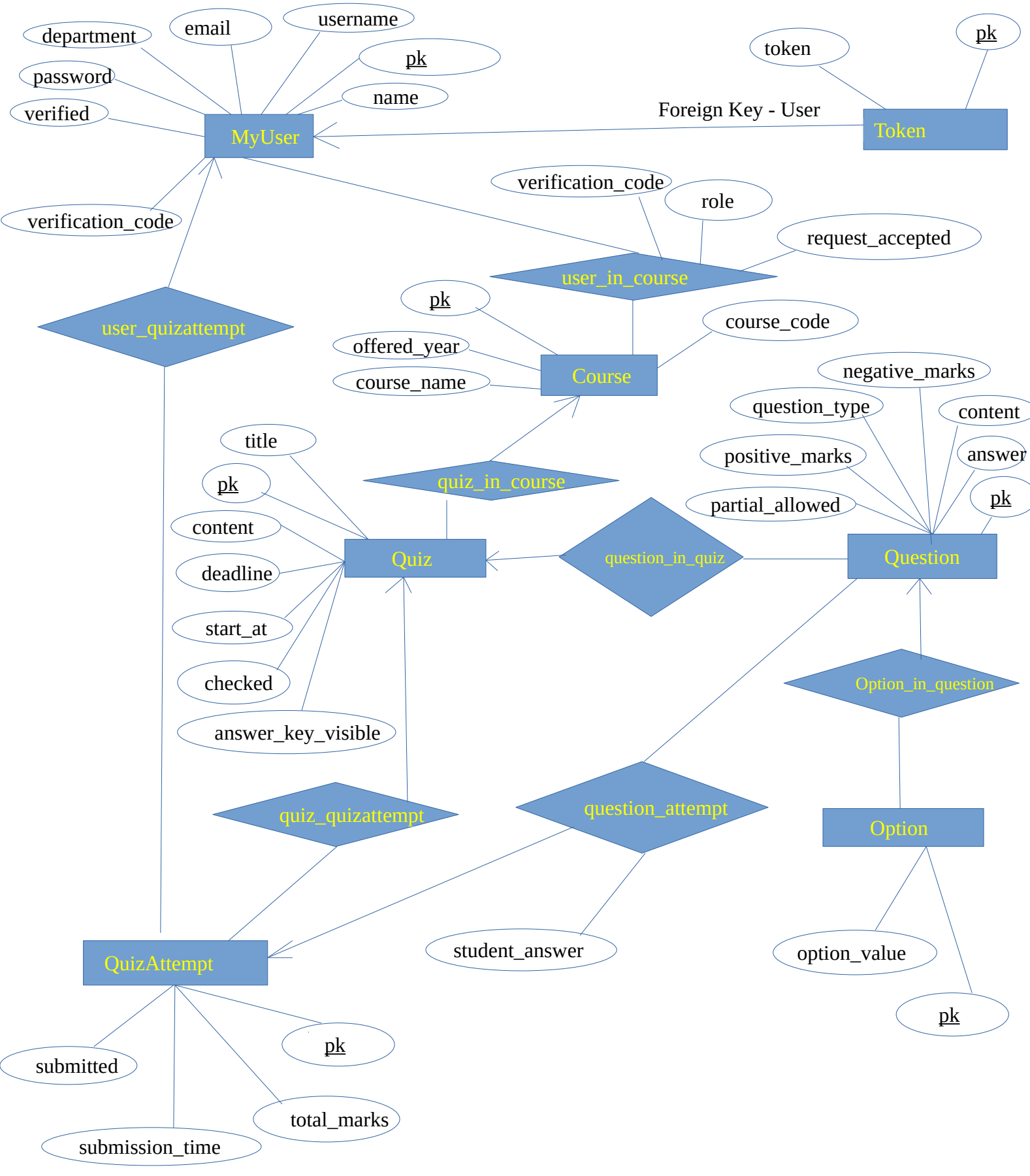
Quiz: It contains information about a particular quiz. It has attributes like content, start time, deadline, checked (whether graded or not), etc. It has a one-to-many relationship with the Question entity, which contains information about the questions in the quiz. It also has a one-to-many relationship with the QuizAttempt entity, which contains information about the submissions of the questions in the quiz made by the students.

Question: It contains information about the question in the quiz, whether the question has partial or negative marking in it, the answer to the question and the marks to be awarded. It has a one-to-many relationship with the Option entity.

Option: It contains options corresponding to questions in the Question entity.

QuizAttempt: It contains information about an attempt of a particular student like the marks scored by the student, the submission time of the quiz, and whether the quiz was submitted or not.

ER Diagram


department
email
username
pk
password
name
verified
MyUser
verification_code
Foreign Key - User
token
pk
Token
verification_code
role
request_accepted
user_in_course
user_quizattempt
pk
course_code
offered_year
course_name
Course
negative_marks
question_type
content
positive_marks
answer
partial_allowed
pk
title
pk
quiz_in_course
content
Quiz
question_in_quiz
Question
deadline
start_at
checked
answer_key_visible
Option_in_question
quiz_quizattempt
question_attempt
Option
student_answer
option_value
QuizAttempt
pk
submitted
total_marks
submission_time

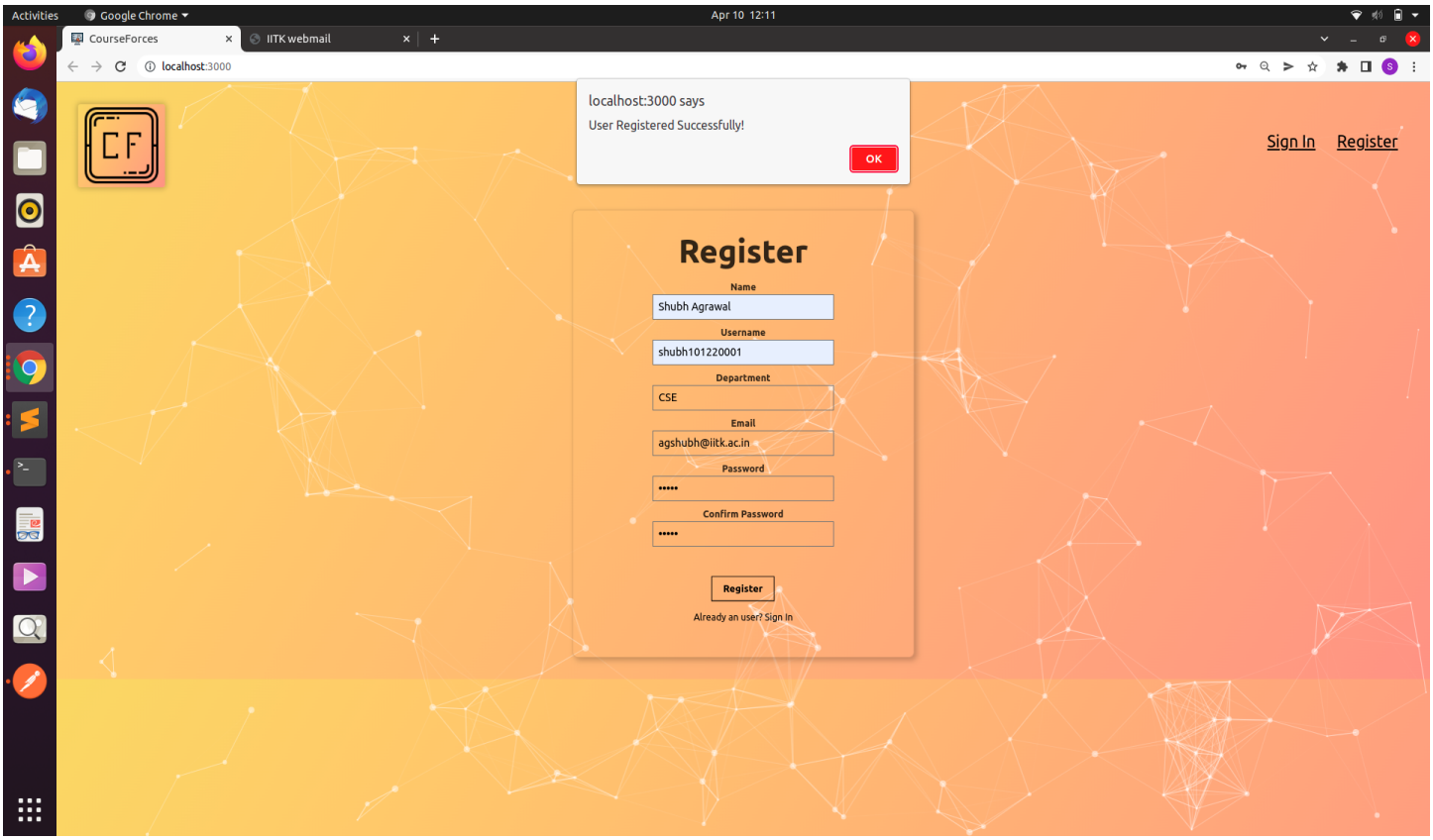
Implementation Details

The web app uses ReactJS for the frontend along with the tachyons library for functional CSS.

The backend is written using the python framework, Django. The backend also uses the Django REST framework. We have used SQLite as the database, which is the default database with Django.

The details of the screens are as follows:

1.) Register Screen:



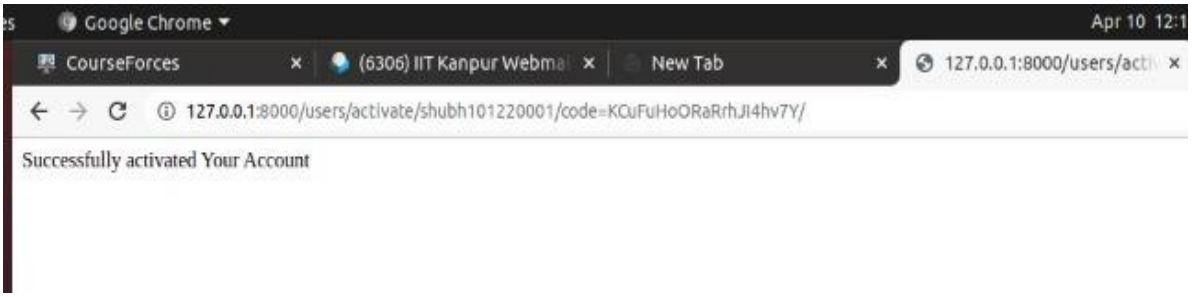
The first screen is the registration screen where a user can register on our app. Here the following details are asked from the user: Name, Username, Department, Email, and Password. The password is asked twice for confirmation. On clicking the register button, it calls a backend API which registers the user in the database, and the frontend gets routed to the SignIn page.

Now, this API call returns a status code of 200 if the API call was successful else this API returns an error code with the error.

In case of errors, those errors are shown as an alert message.

If the API showed a status of 200, this means the user was registered successfully and in this case, we send a mail with an activation link to the user on their email address so as to verify the email address. An example screen of the same email is shown below.

The mail is sent from the following mail id: cs315legends@gmail.com (lol :p). On clicking on the activation link, you get the following, provided you had not already activated your account and the link copied is correct.

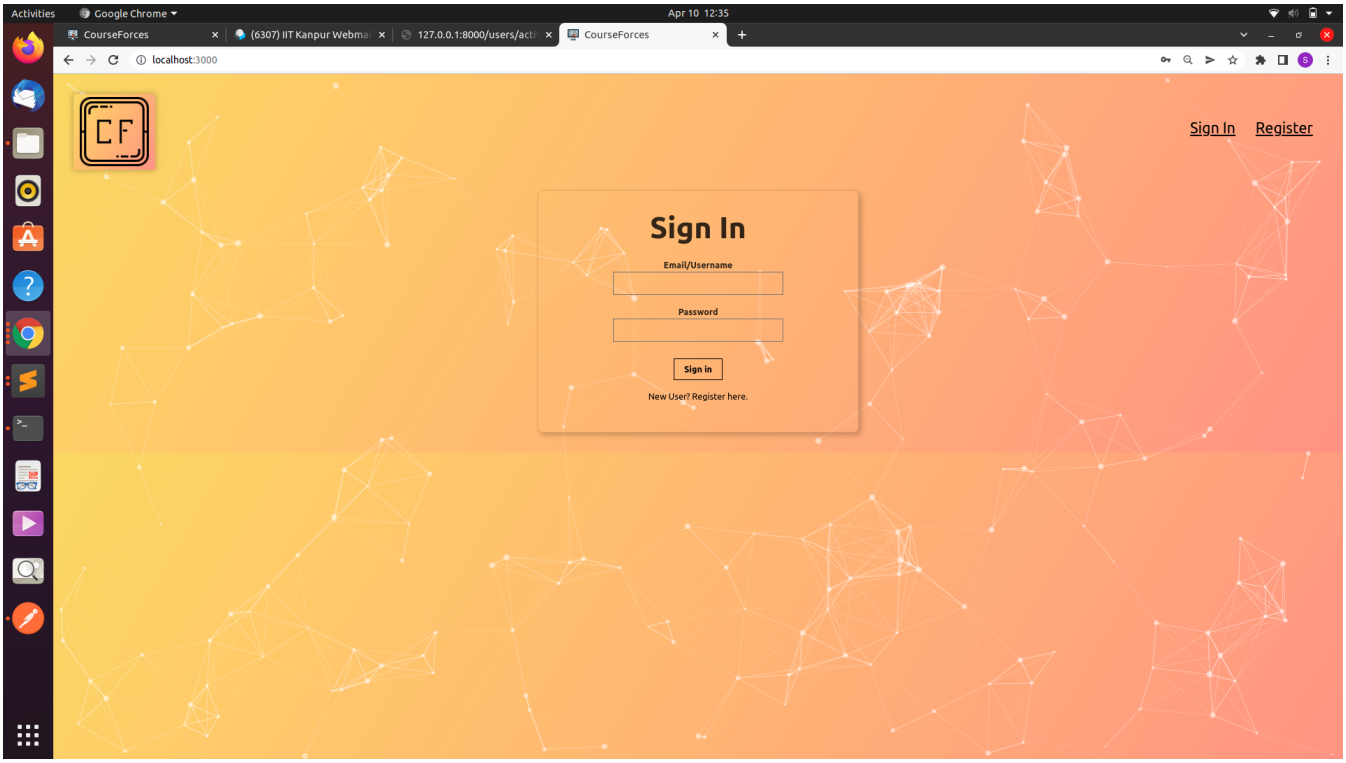


The state of the database in the backend after successful activation is as follows:

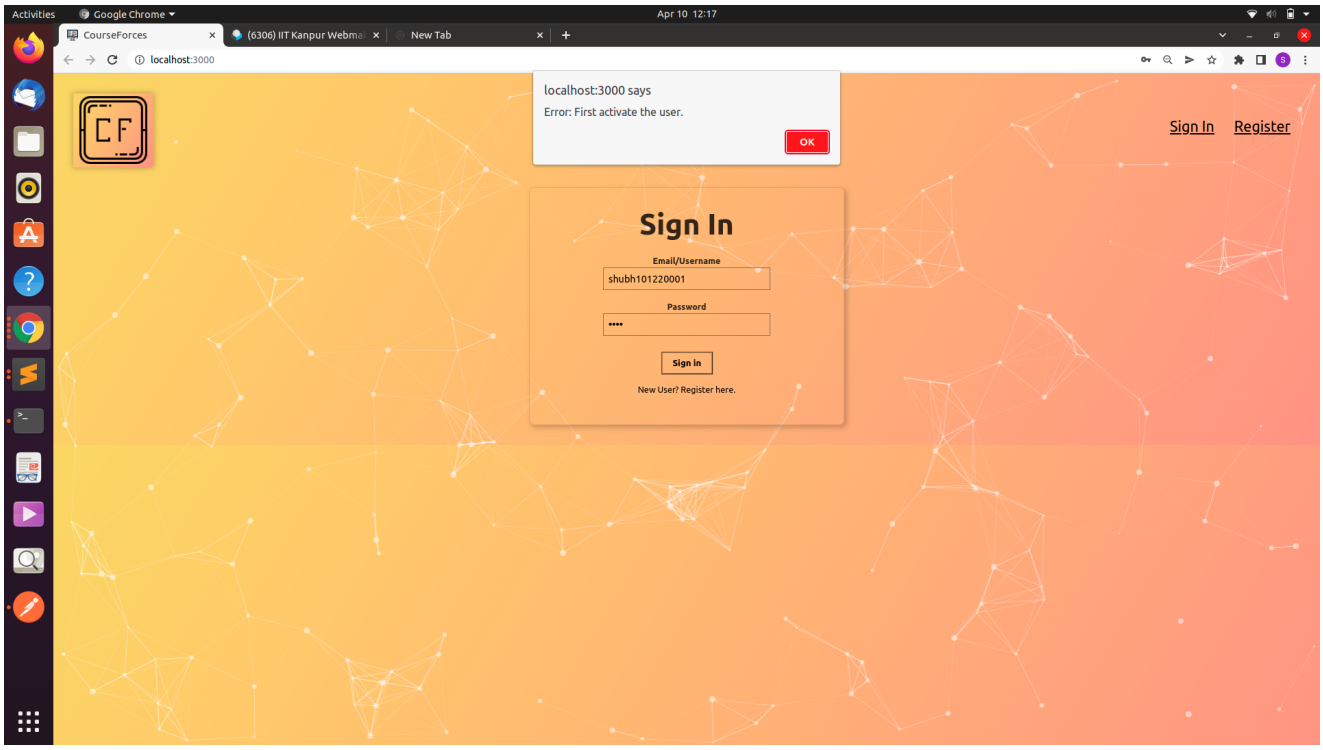
Now the user is registered into the app.

2.) Sign In Page

Users can use their username/email with the password to log in. In case the user is registered, the frontend gets routed to the home page containing the list of courses in which the user is a student or professor. On successful sign-in, we also receive an Authorization token from the backend which is saved in the state in the frontend. This token is sent to the backend in the subsequent API calls while the user is logged in.

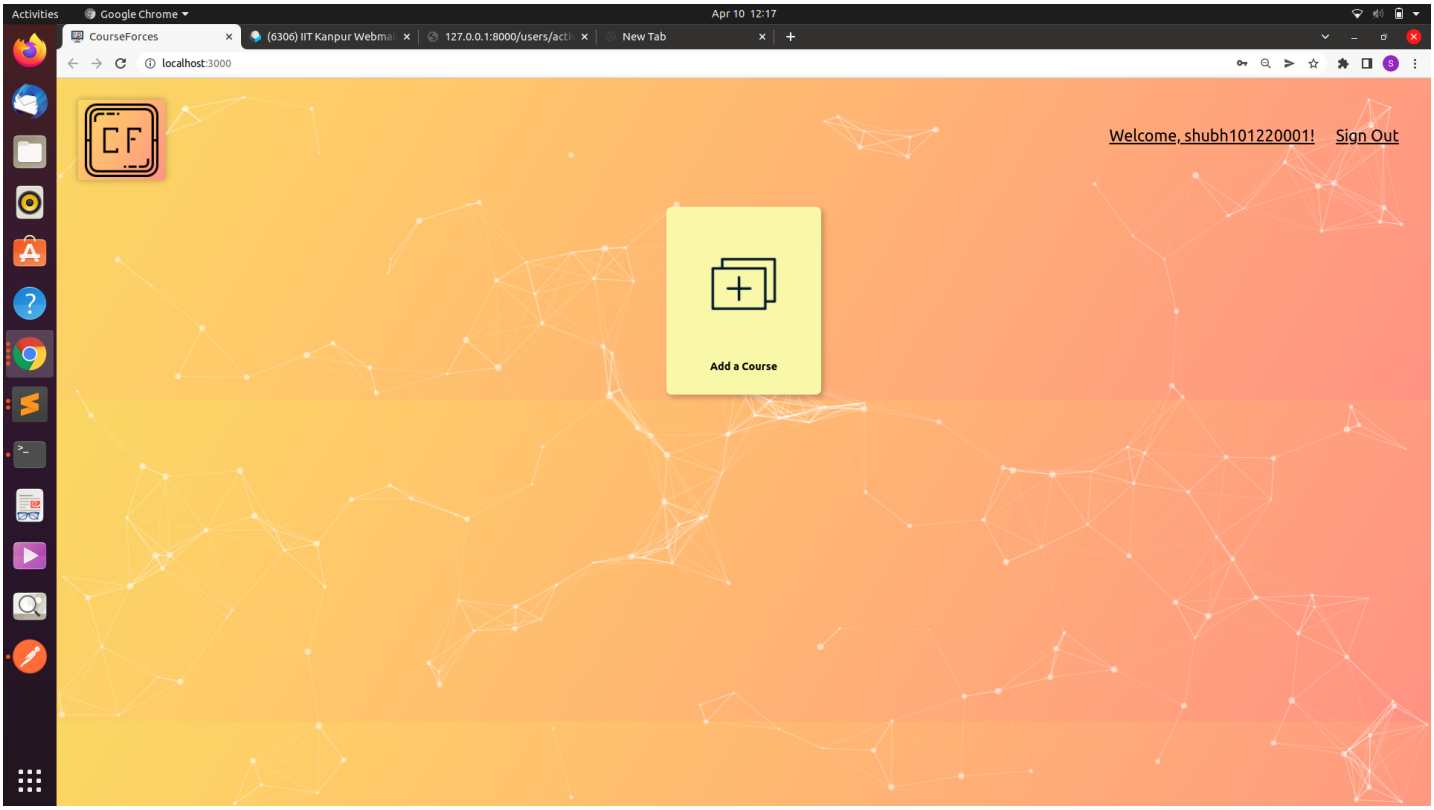


Note: If the user has not activated his account by clicking on the link sent on mail, he won't be able to log in and will receive an error message as an alert as follows:



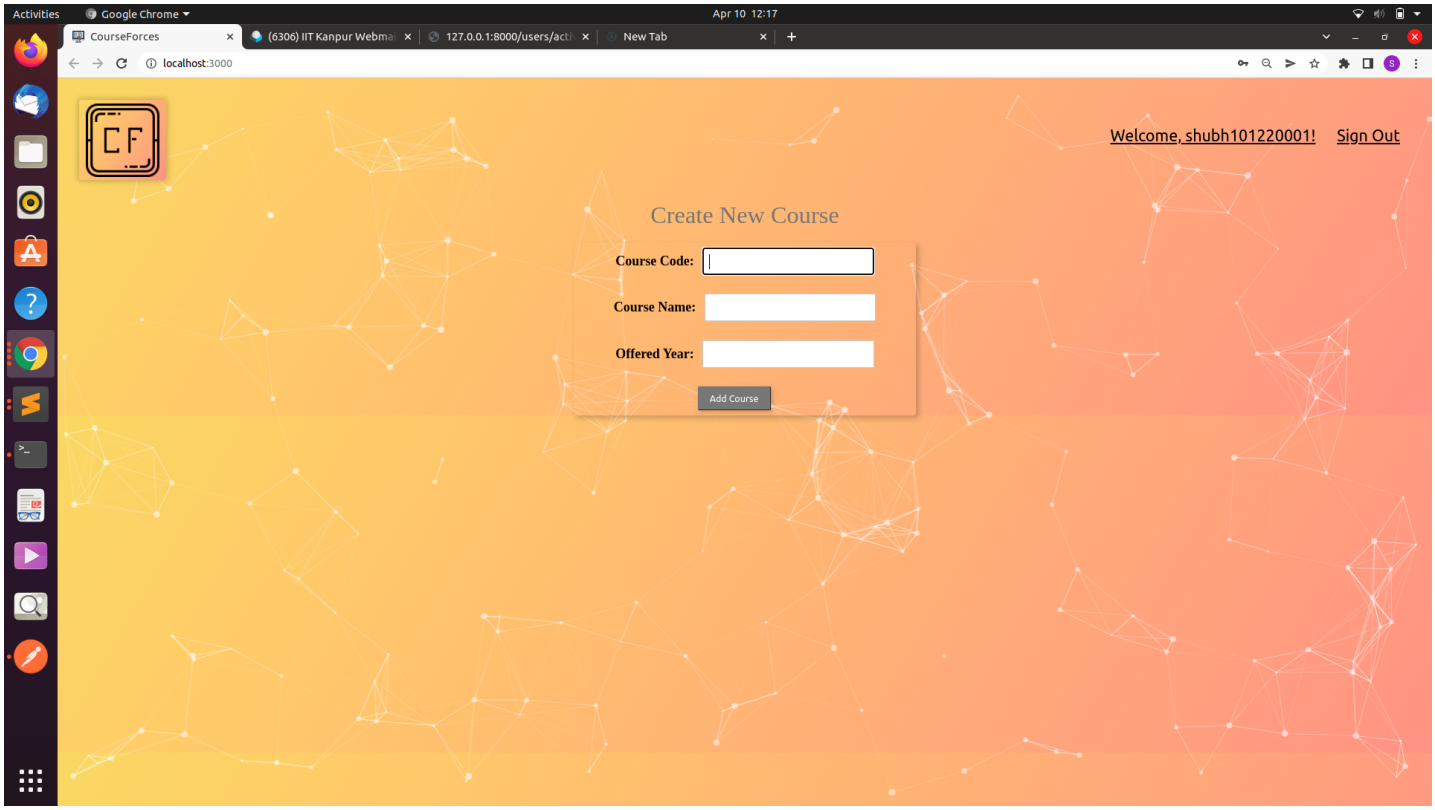
Note: Here the navigation component uses React Routing to display the page, on clicking register he would get the registration page. On successful login, the username is displayed on the navbar, which on clicking routes it to the home page. Moreover, the signout buttons log out the user.

3.) Initial Course List

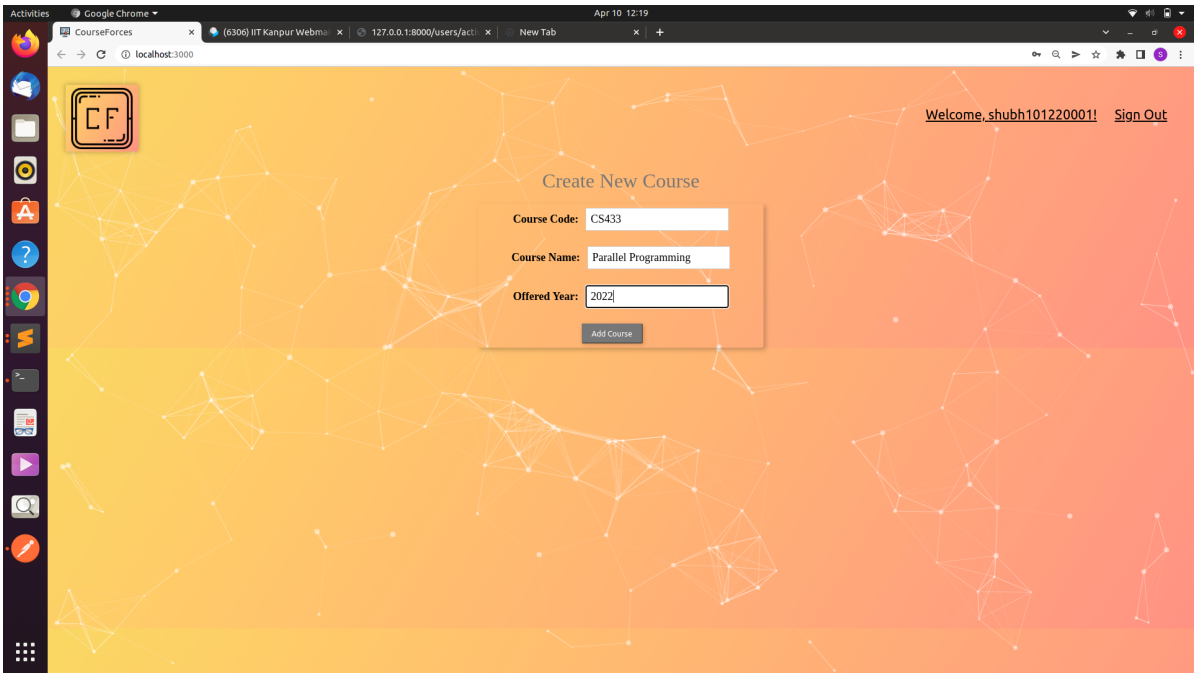


Since the user has just registered on the app, there is no course in which he is registered. Here he can create a new course using Add a Course button. Here a user will get the list of all those courses which he has created or has accepted the course request (more information on courses will be there in the upcoming pages)

4.) Creating a new course

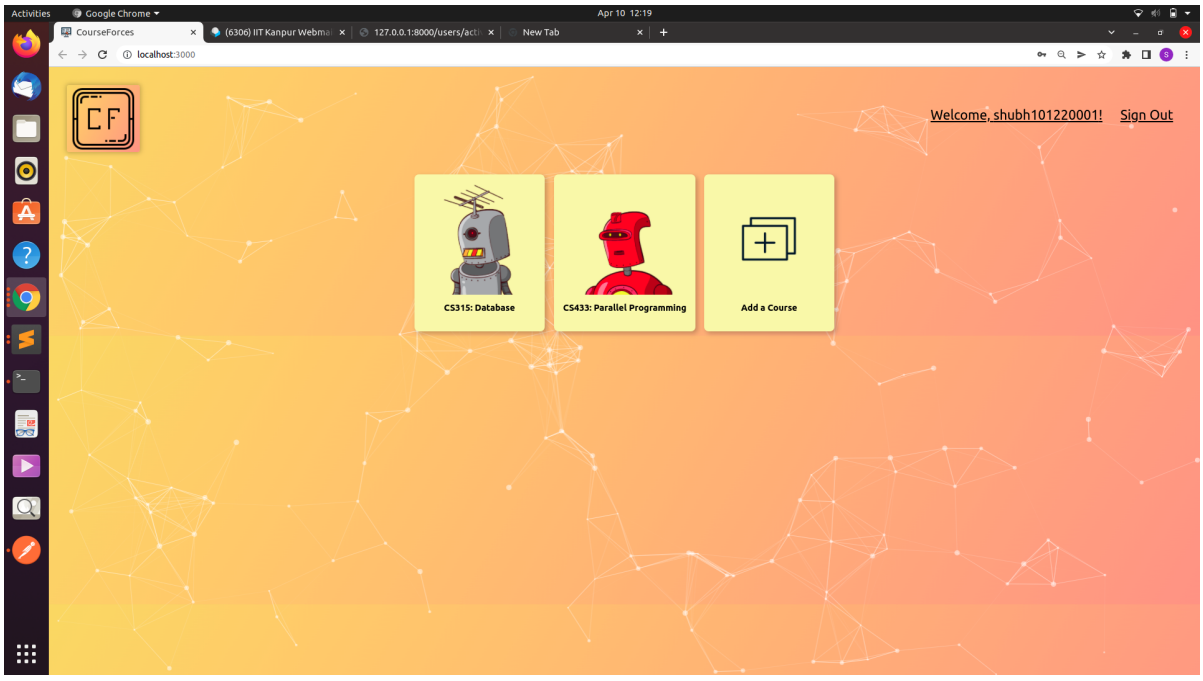


To create a new course, a user has to enter the course code, course name, and the offered year (only numbers are allowed). On clicking add a course, it calls a backend API to add the course to the database. By default, if a user has created a course, then he will be a professor in that course.



5.) View of the Course list

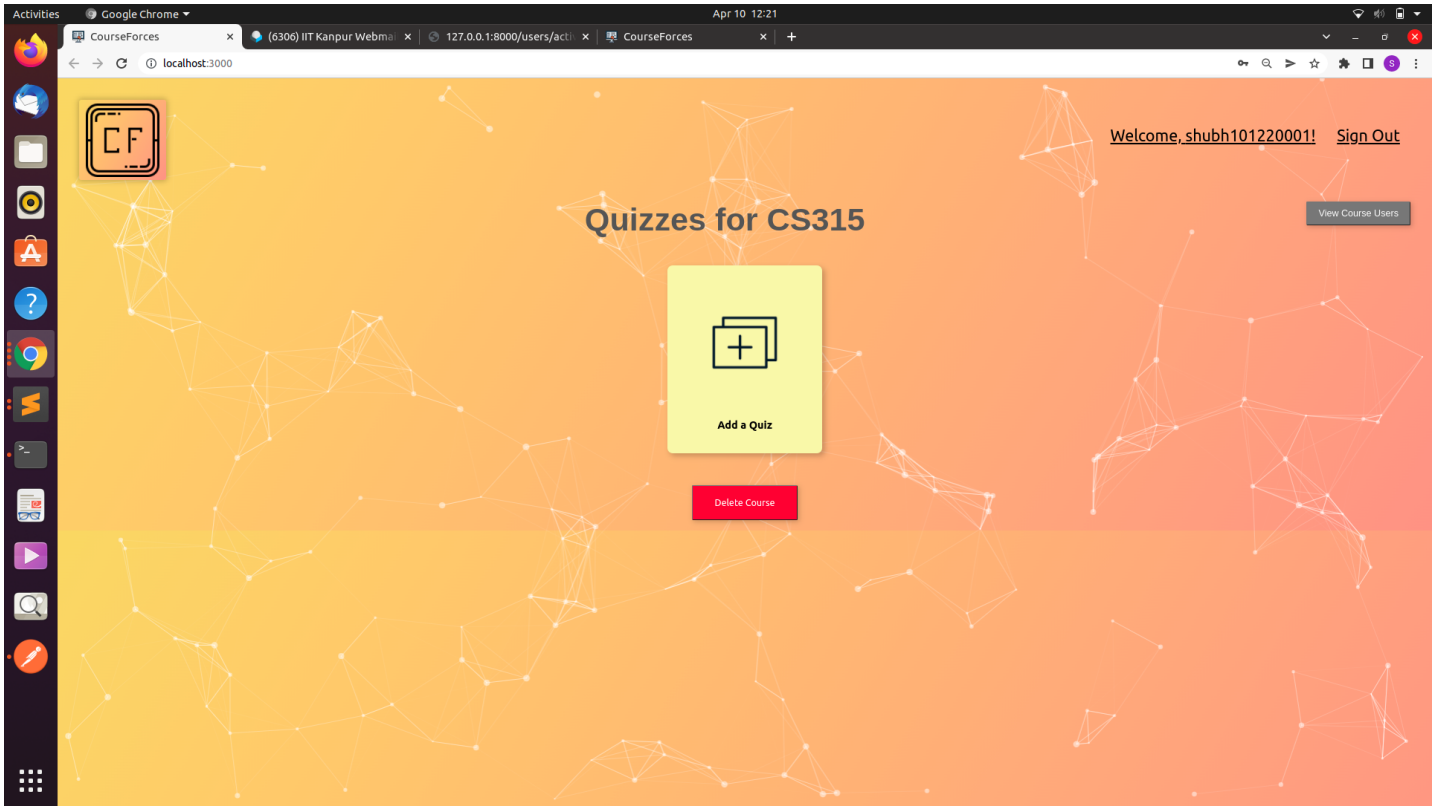
When there are a few courses in the database corresponding to the user, the screen view changes to the following:



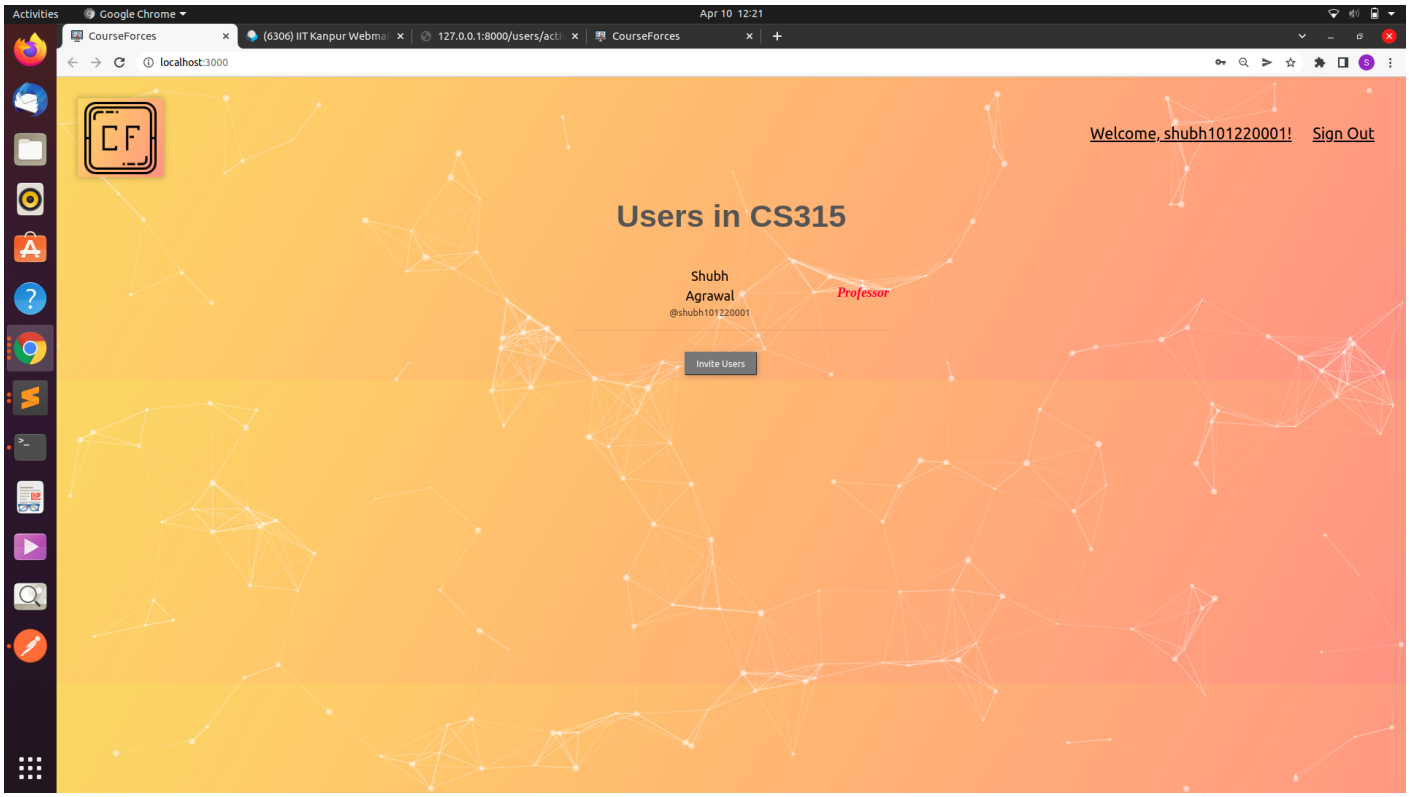
On a fun note, the robot images are fetched from “robohash.org”, using the course code as the text required to generate the robot, so it shows the same robot for each user of the course.

6.) **View of quiz list for the professor of a course**

Initially, there is no quiz in the course, so all we can see are buttons for adding a quiz, viewing the list of users in the course, and a button to delete the course.

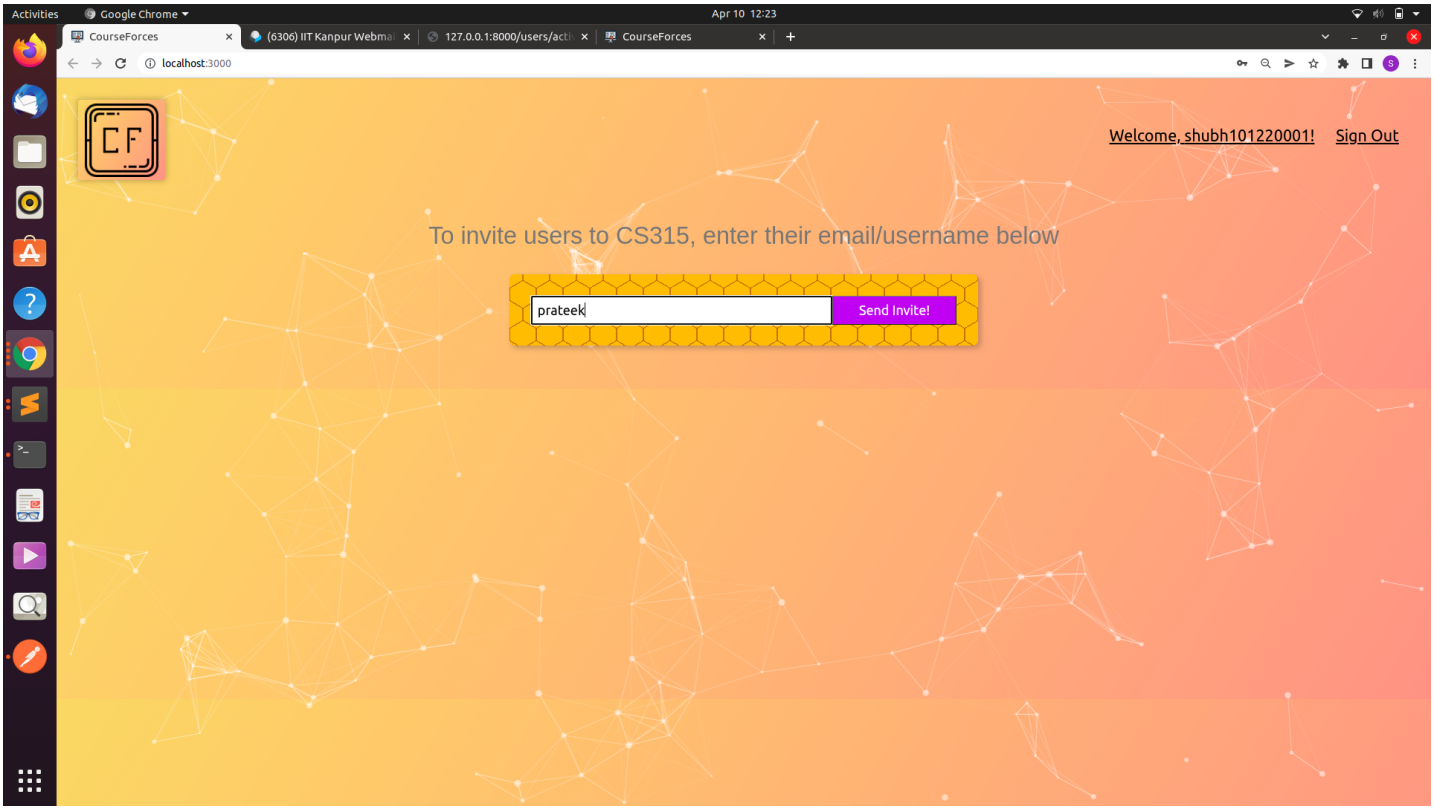


7.) Users in the course

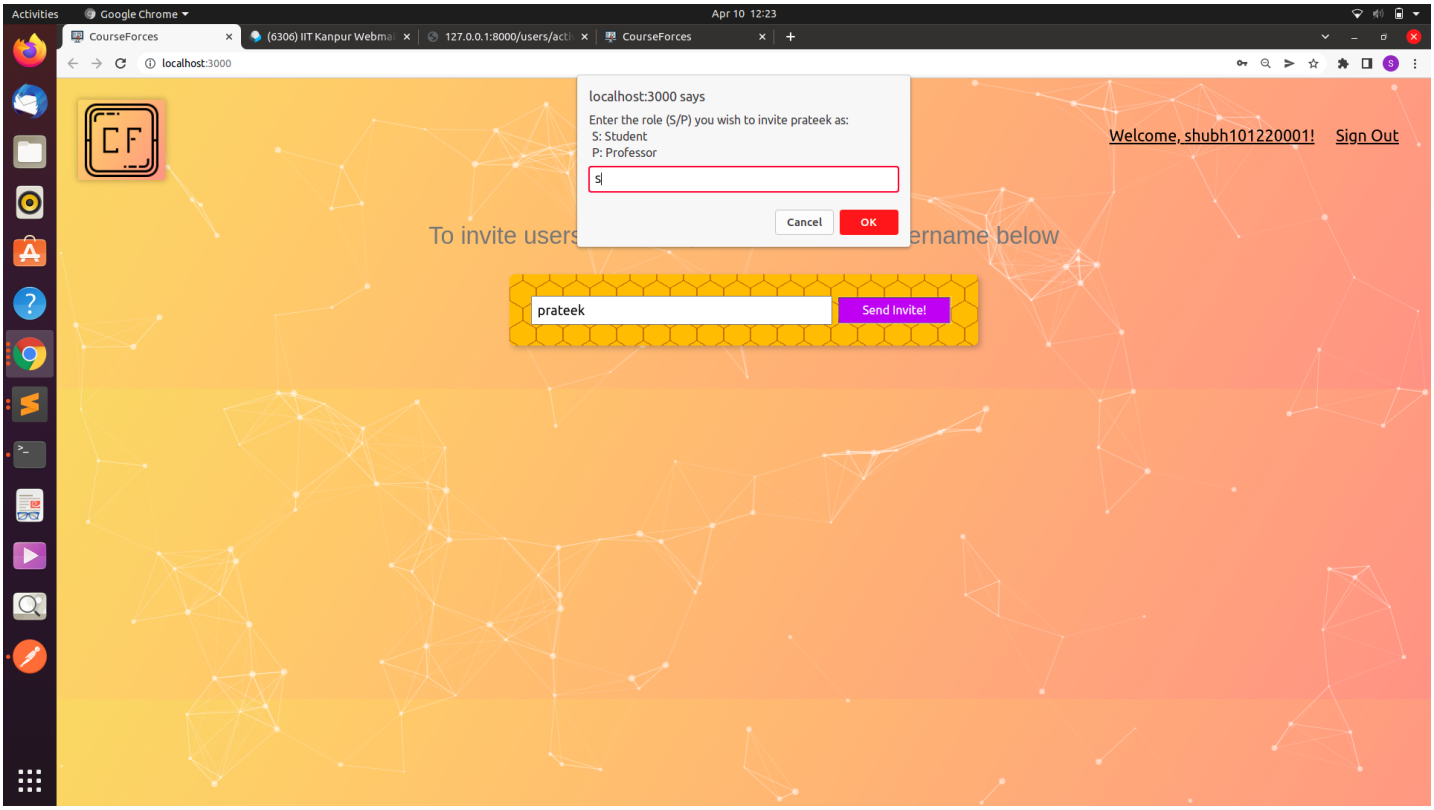


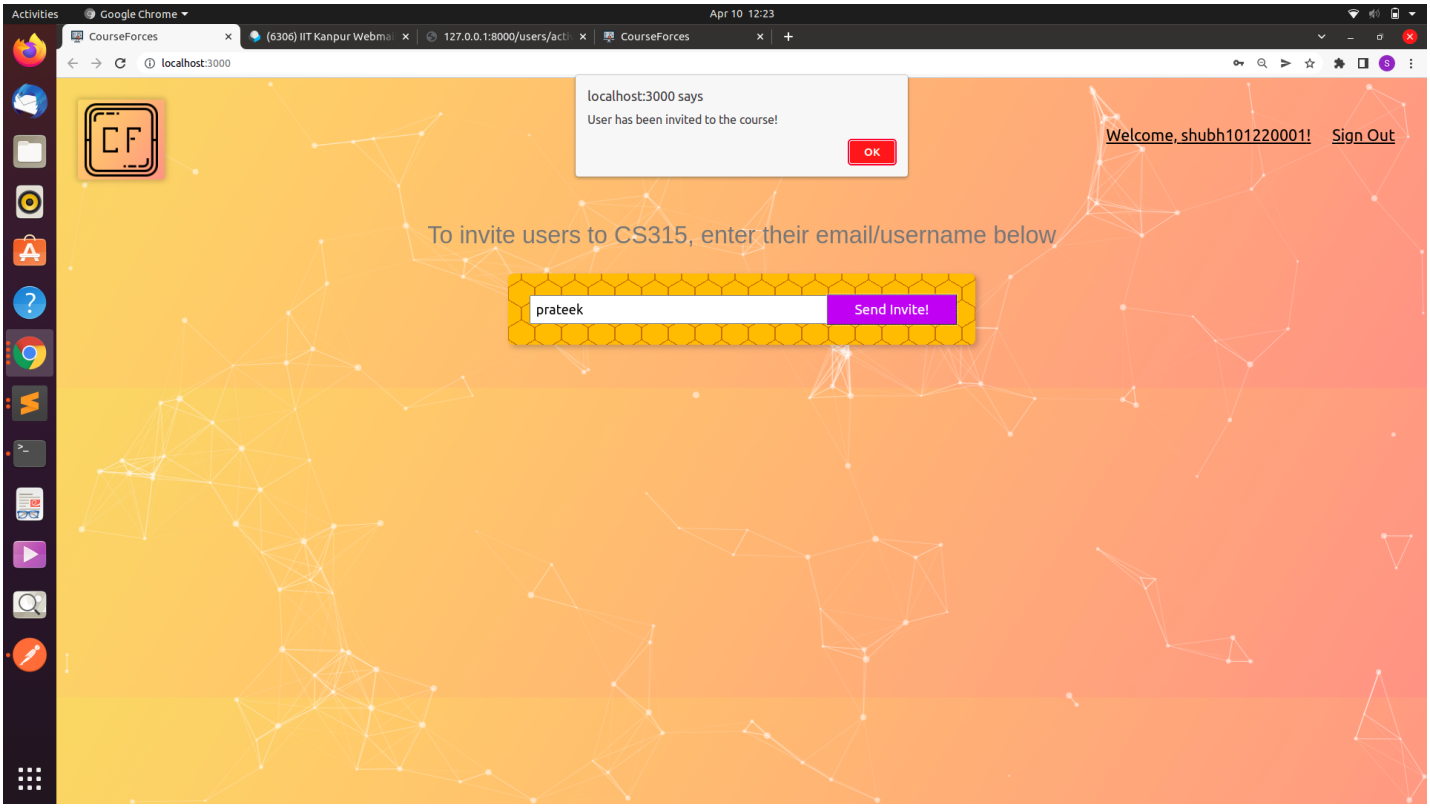
This page is visible to the professor only, here he can see the list of students as well as other professors in the course. Further, he can also invite students or professors in this course using the invite users button, as shown below.

8.) Inviting a user to the course



On this page, a professor can enter the username of the user to be invited to the course. On clicking the send invite button, there is an alert that asks whether to invite the user as a student or as a professor. The screen view is shown below.


Activities
Google Chrome
Apr 10 12:23
CourseForces
(6306) IIT Kanpur Webmail
127.0.0.1:8000/users/activate/
CourseForces
localhost:3000
localhost:3000 says
Enter the role (S/P) you wish to invite prateek as:
S: Student
P: Professor
s
Cancel
OK
Welcome, shubh101220001!
Sign Out
To invite users
ername below
prateek
Send Invite!

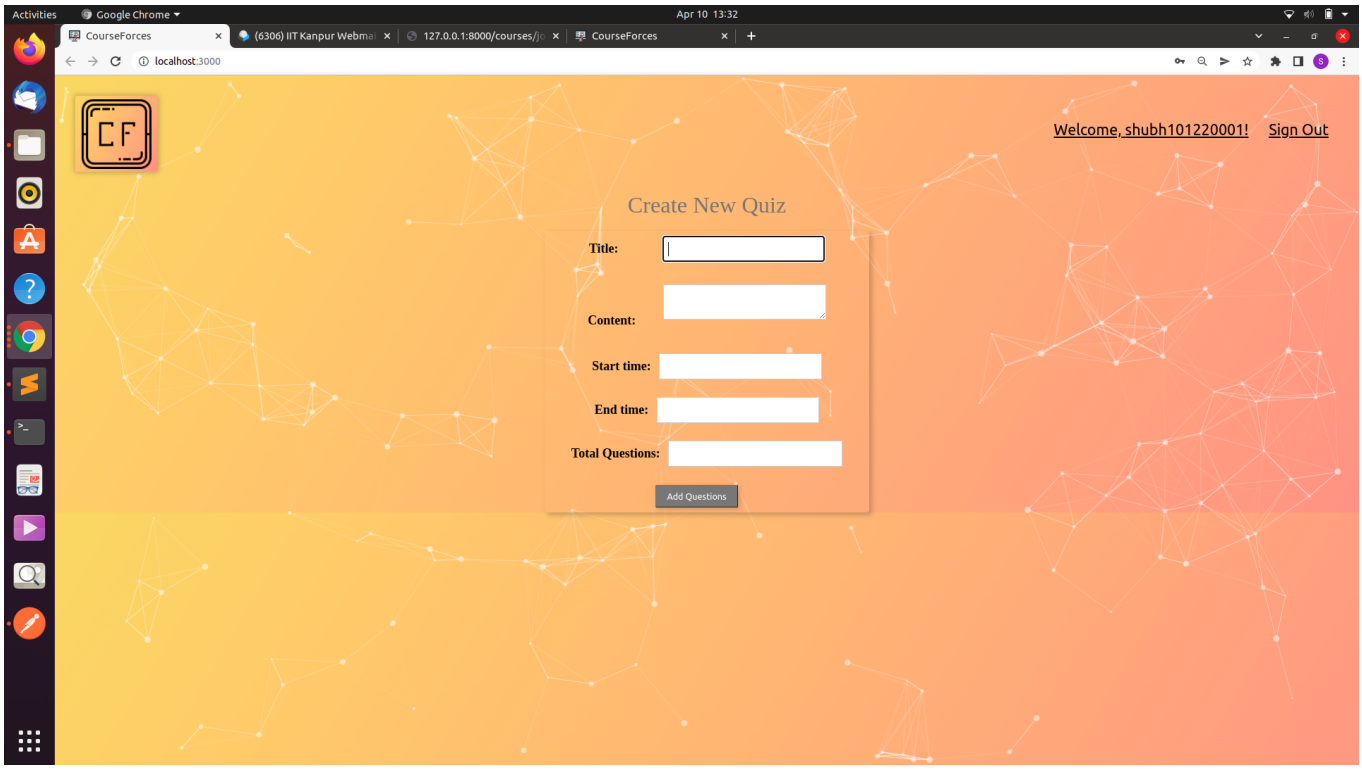

Activities
Google Chrome
Apr 10 12:23
CourseForces
(6306) IIT Kanpur Webmail
127.0.0.1:8000/users/activate/
CourseForces
localhost:3000
localhost:3000 says
User has been invited to the course!
OK
Welcome, shubh101220001!
Sign Out
To invite users to CS315, enter their email/username below
prateek
Send Invite!

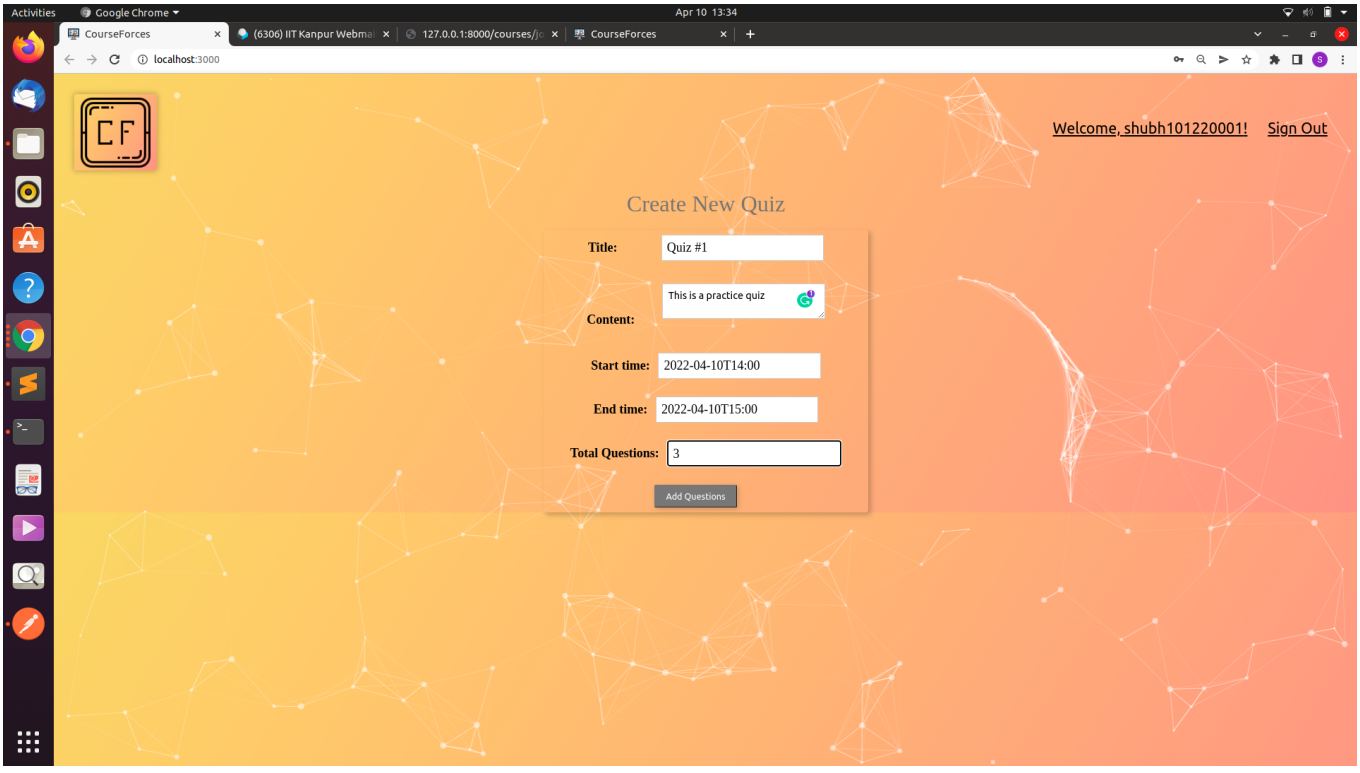
An alert is sent when successful login is shown on the screen, else the error message is shown in the alert.

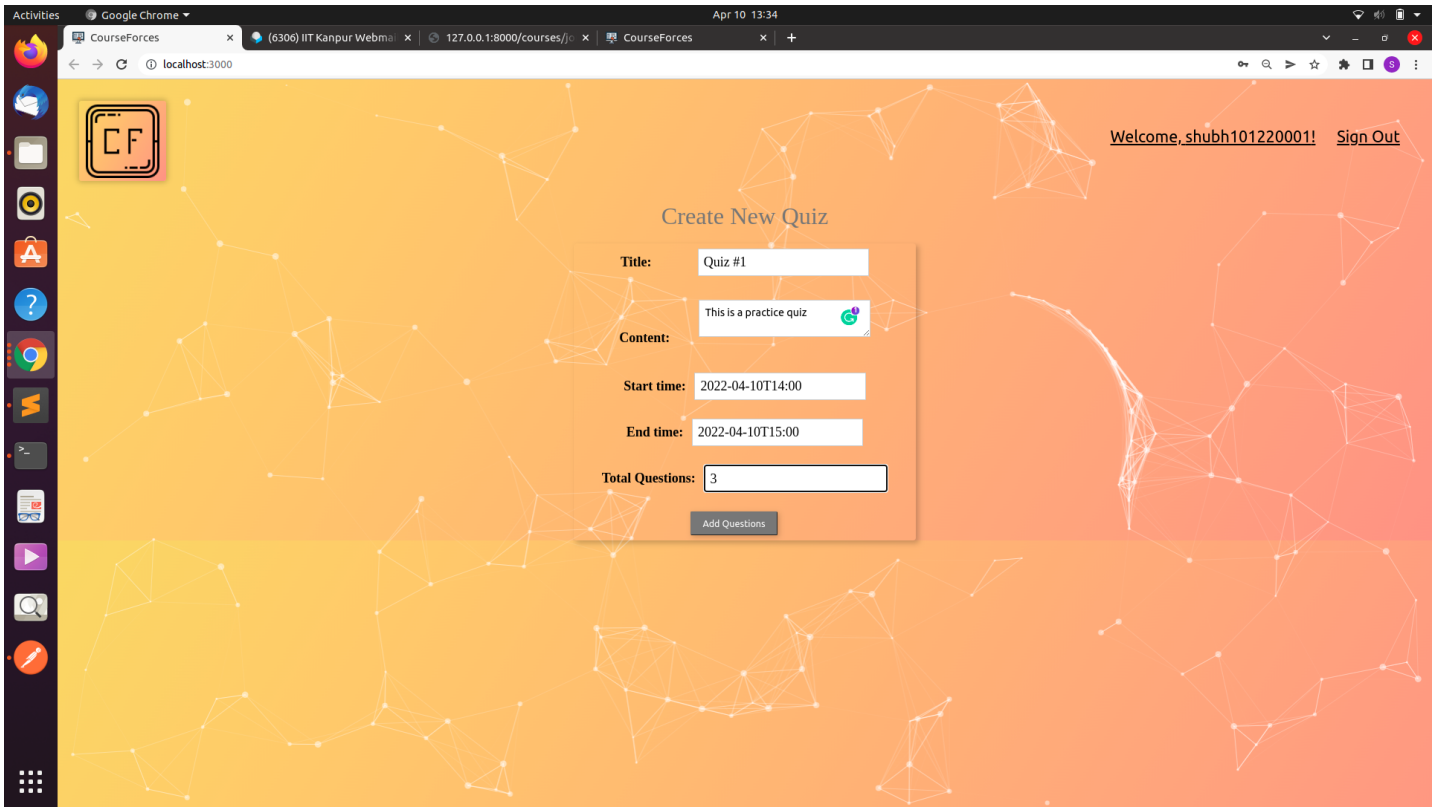
A mail with a registration link is sent to the user on a successful invite. When the user copies the link to the browser, the user has accepted the course.

9.) Create Quiz

Professor can create a quiz on this page. The Quiz title is something like “Quiz 1” which is displayed to students. The content of the quiz contains instructions to be displayed, like “Please score positive marks!” (:p). The format of start time and end time are as follows: <YYYY-MM-DDThh: mm> (Example: 2022-12-12T11:11:00Z).

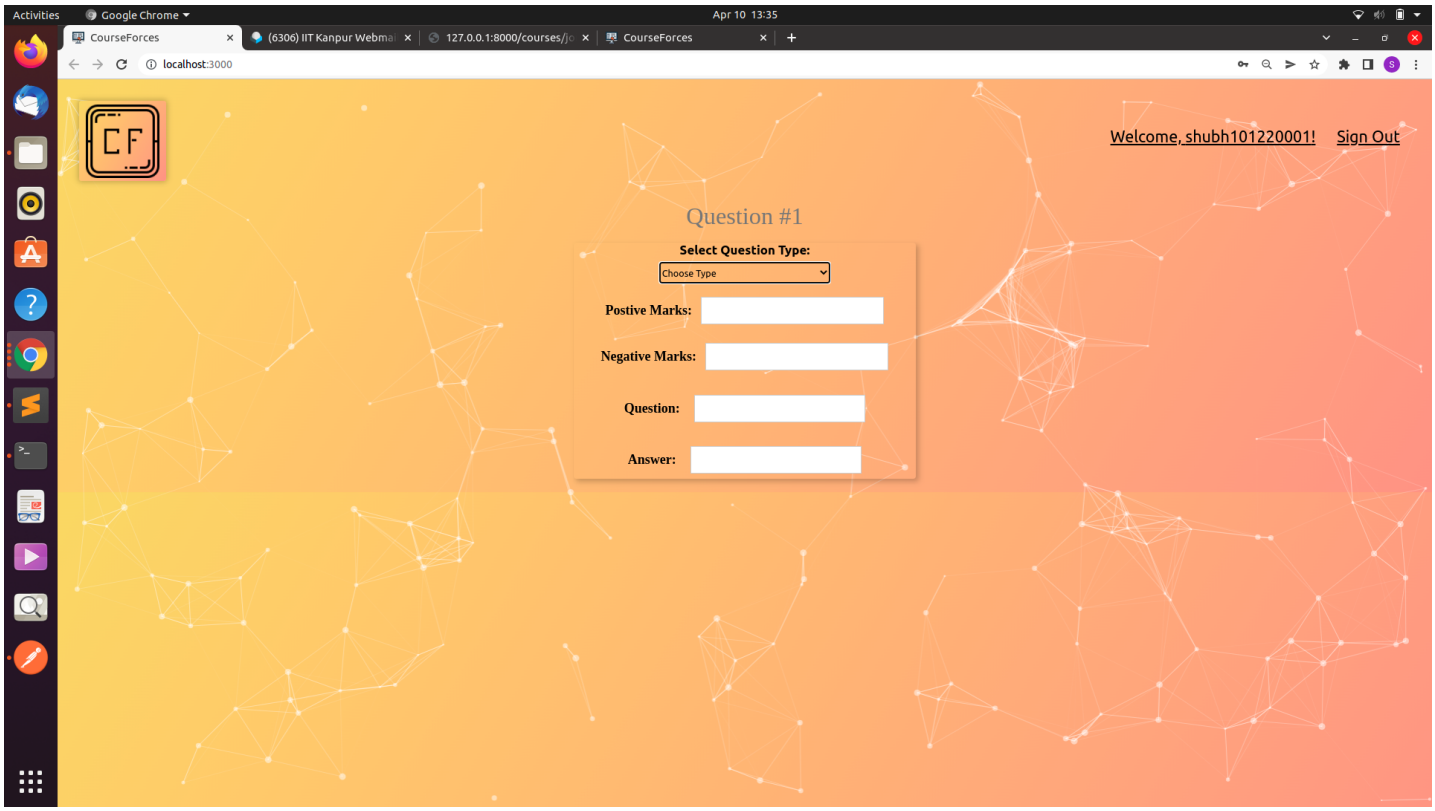

Activities
Google Chrome
Apr 10 13:32
CourseForces
(6306) IIT Kanpur Webmai
127.0.0.1:8000/courses/jc
CourseForces
localhost:3000
CF
Welcome, shubh101220001!
Sign Out
Create New Quiz
Title:
Content:
Start time:
End time:
Total Questions:
Add Questions


Activities
Google Chrome
Apr 10 13:34
CourseForces
(6306) IIT Kanpur Webmai
127.0.0.1:8000/courses/jc
CourseForces
localhost:3000
CF
Welcome, shubh101220001!
Sign Out
Create New Quiz
Title:
Quiz #1
Content:
This is a practice quiz
Start time:
2022-04-10T14:00
End time:
2022-04-10T15:00
Total Questions:
3
Add Questions



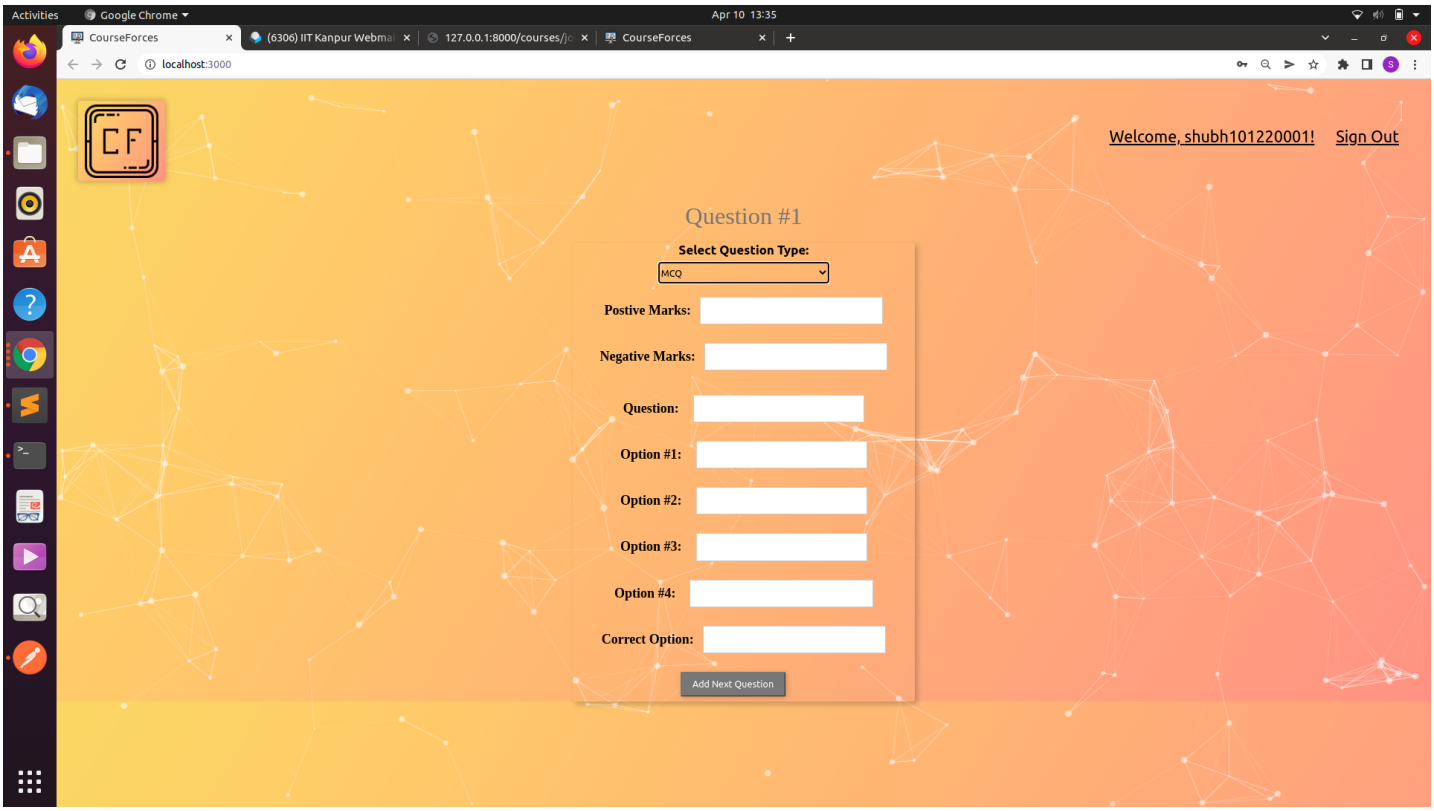
After adding total questions, the page is routed to add questions page, where details of the questions are asked as shown below.

10.) Question Page

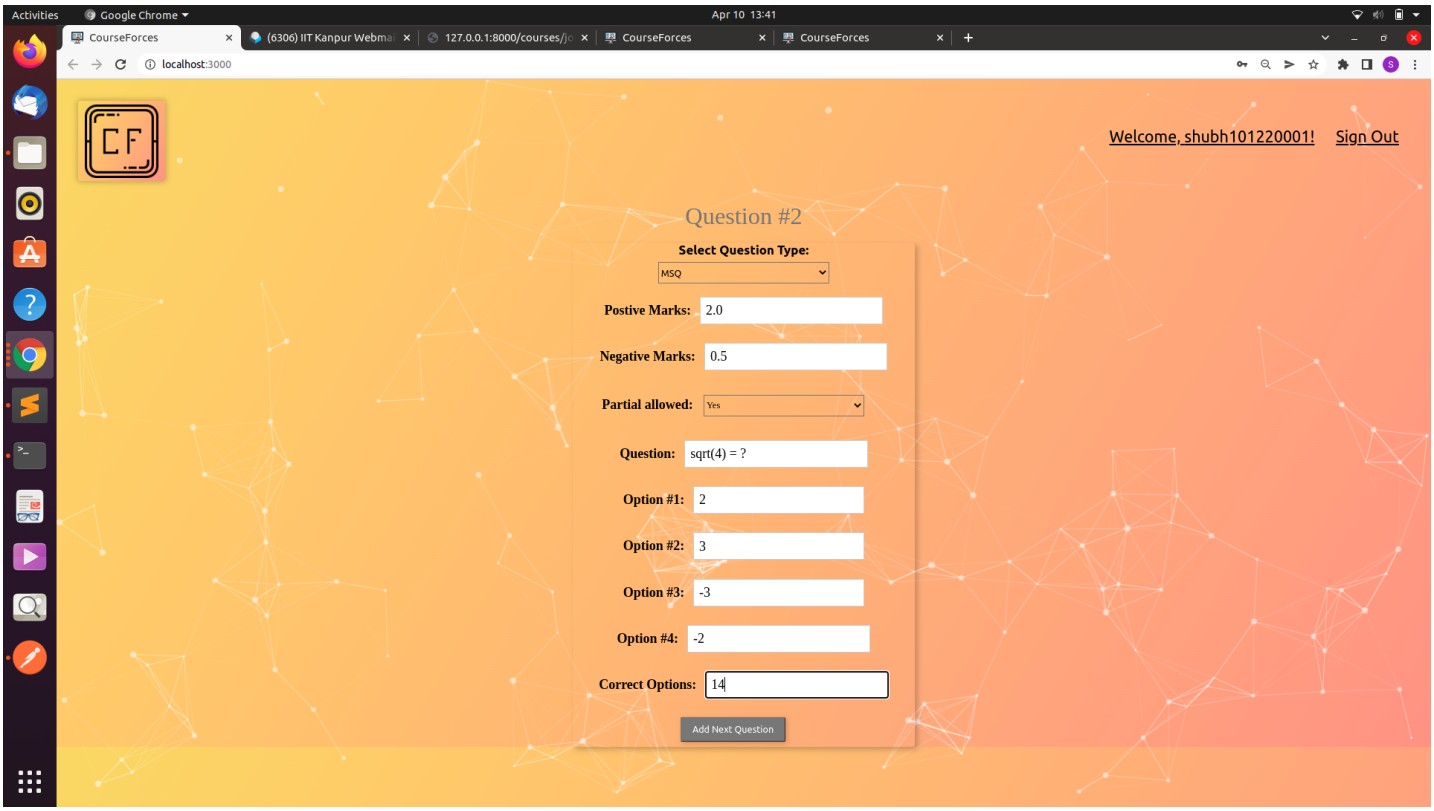


Some fields are common for all question types like Positive Marks, and Negative Marks.

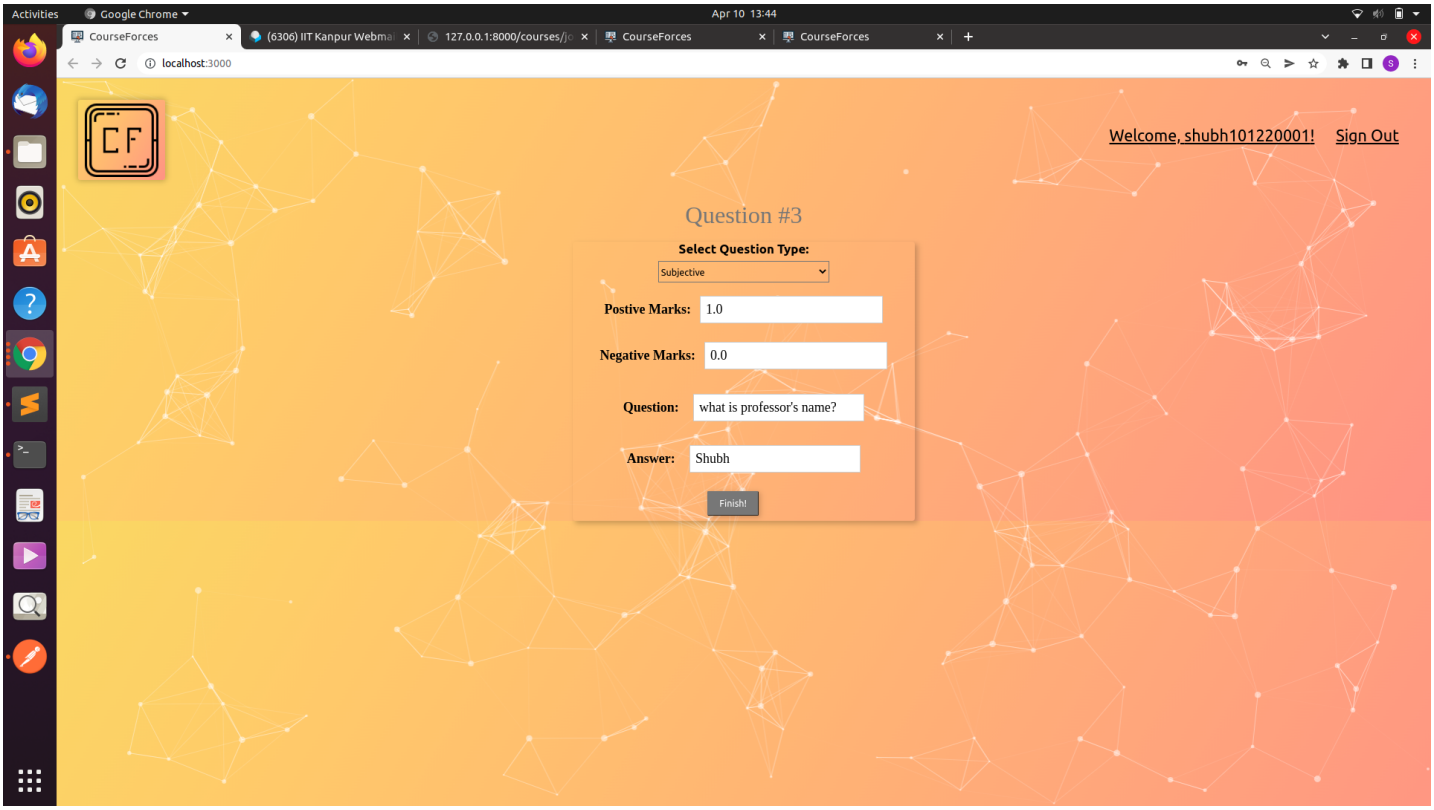
MCQ type question: In this question type, four options and the correct answer are asked. The correct answer has to be entered like an option number.



MSQ type question: This is similar to an MCQ type question, just that the correct answer has to be a string of option numbers. Example: (124 if options 1, 2, and 4 are correct.)

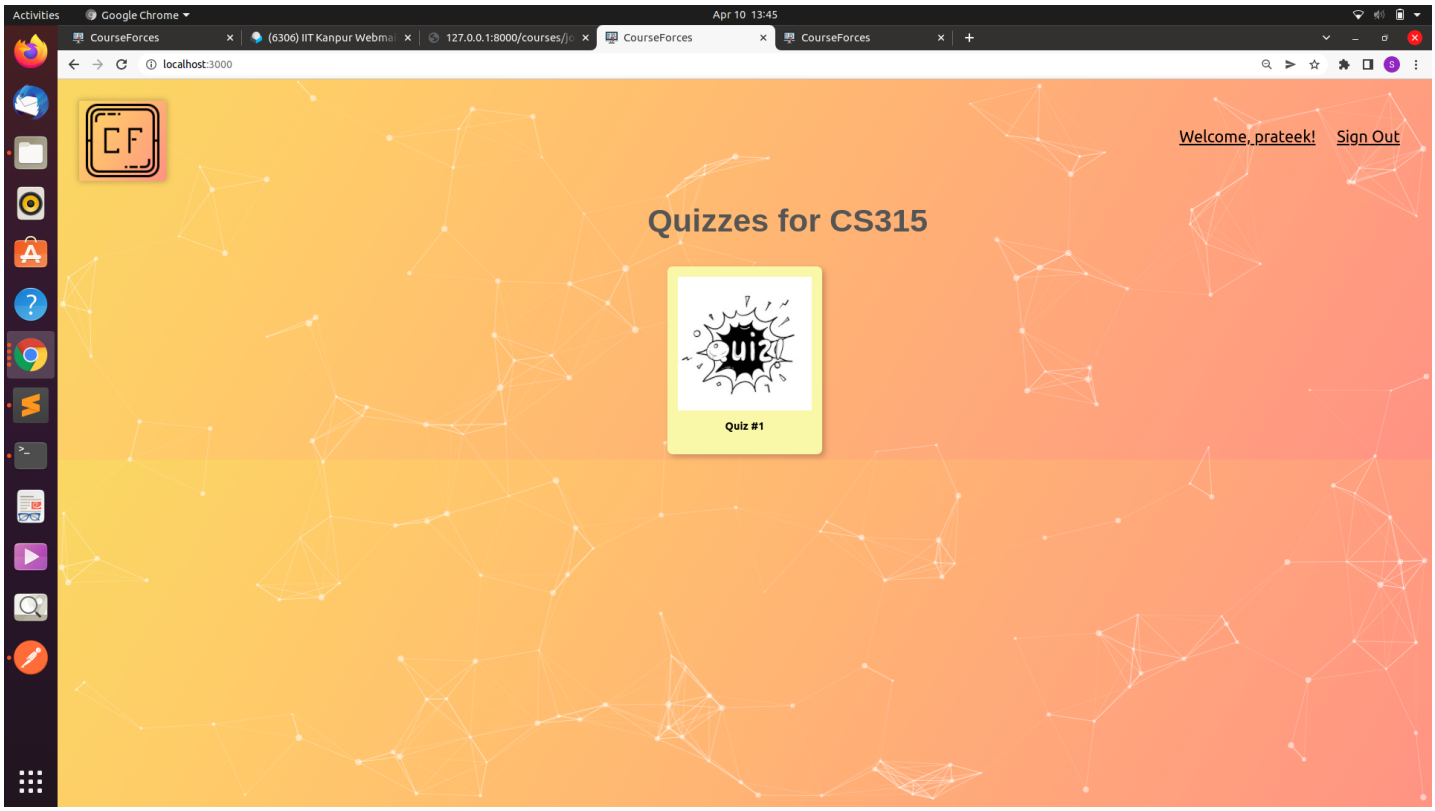


Subjective Question: Here only question and correct answer fields are asked.


Activities
Google Chrome
Apr 10 13:44
CourseForces
(6306) IIT Kanpur Webmail
127.0.0.1:8000/courses/jc
CourseForces
CourseForces
localhost:3000
CF
Welcome, shubh101220001!
Sign Out
Question #3
Select Question Type:
Subjective
Postive Marks:
1.0
Negative Marks:
0.0
Question:
what is professor's name?
Answer:
Shubh
Finish!

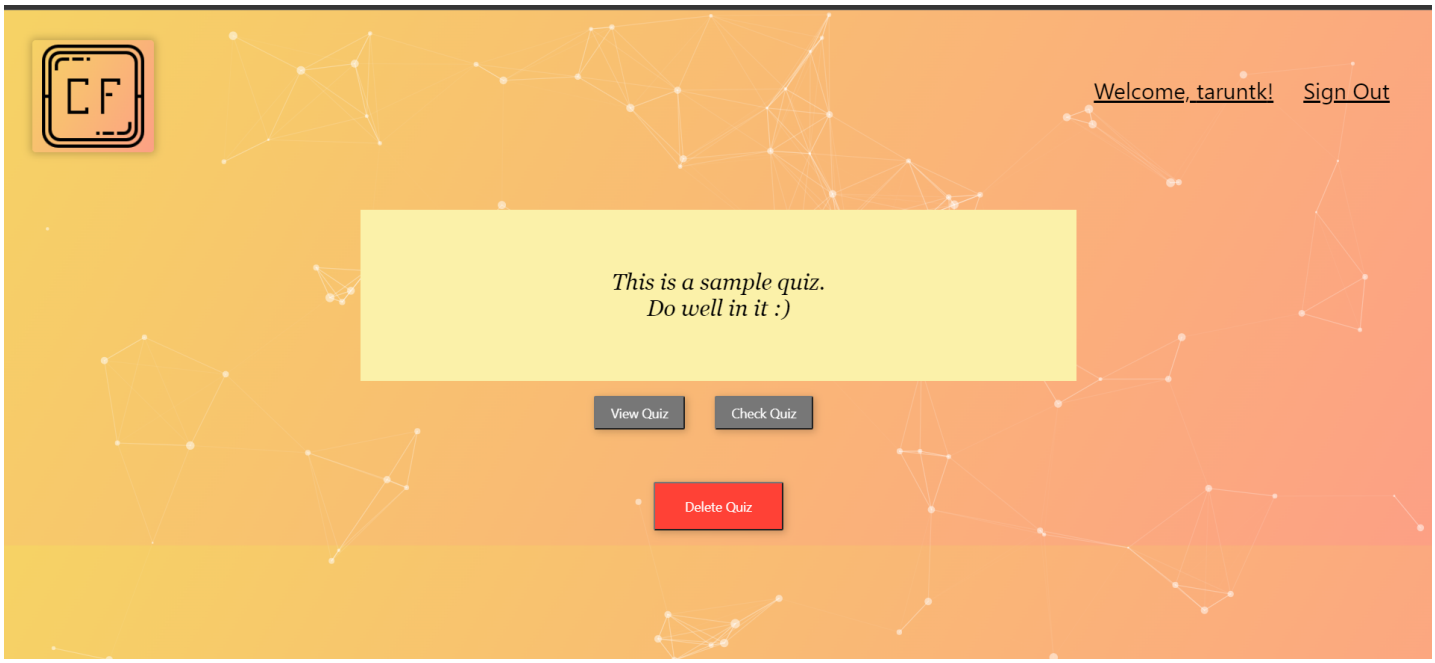
11.) Quiz list view for Student

This page contains the quiz list for the course created by the professor. Note that the delete course button is not available as it was for the professor.



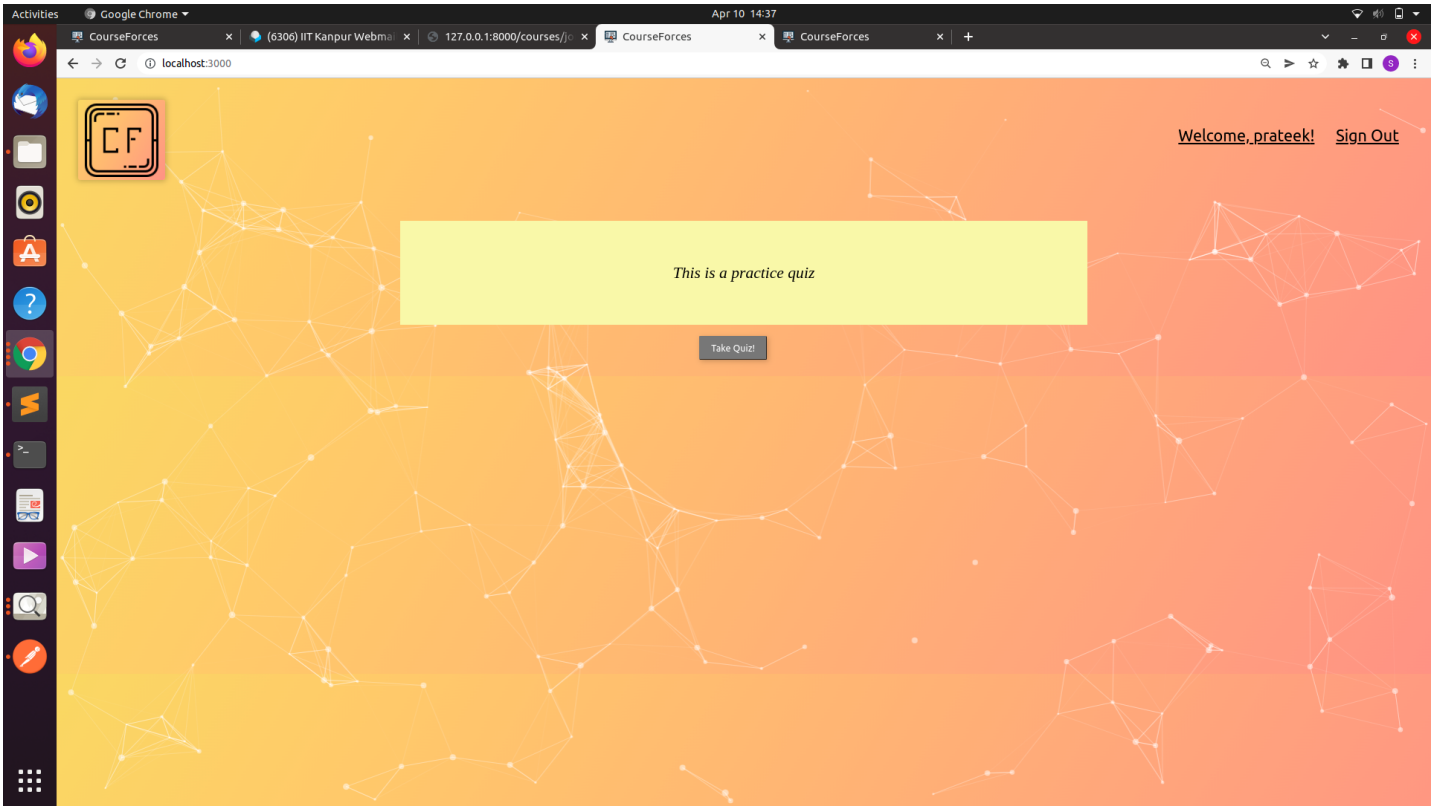
12.) Quiz Information Page:

Professor's View:



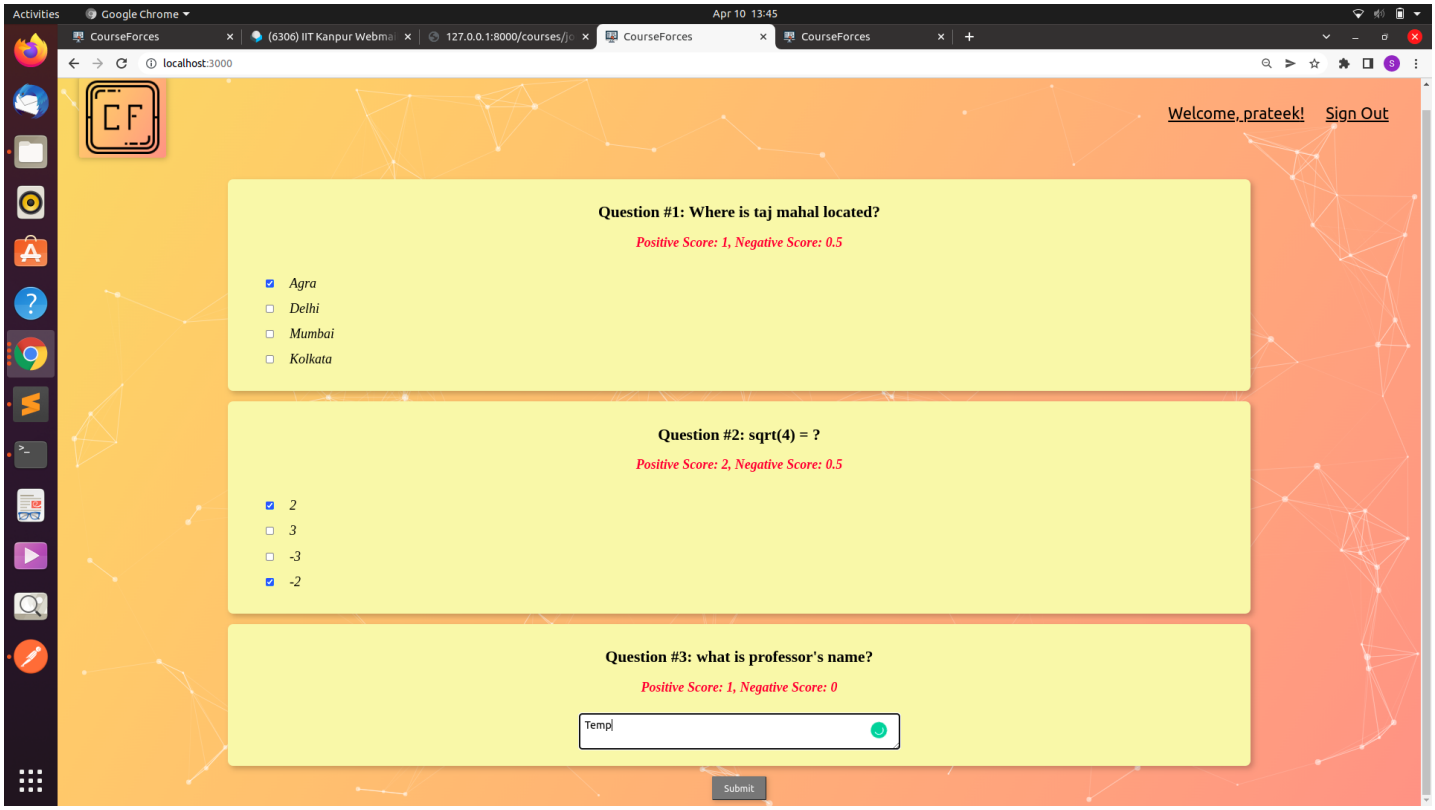
Here, after checking the quiz, the button gets changed to show the mark list button, where the professor can view the marks of students in the course.

Student's View:

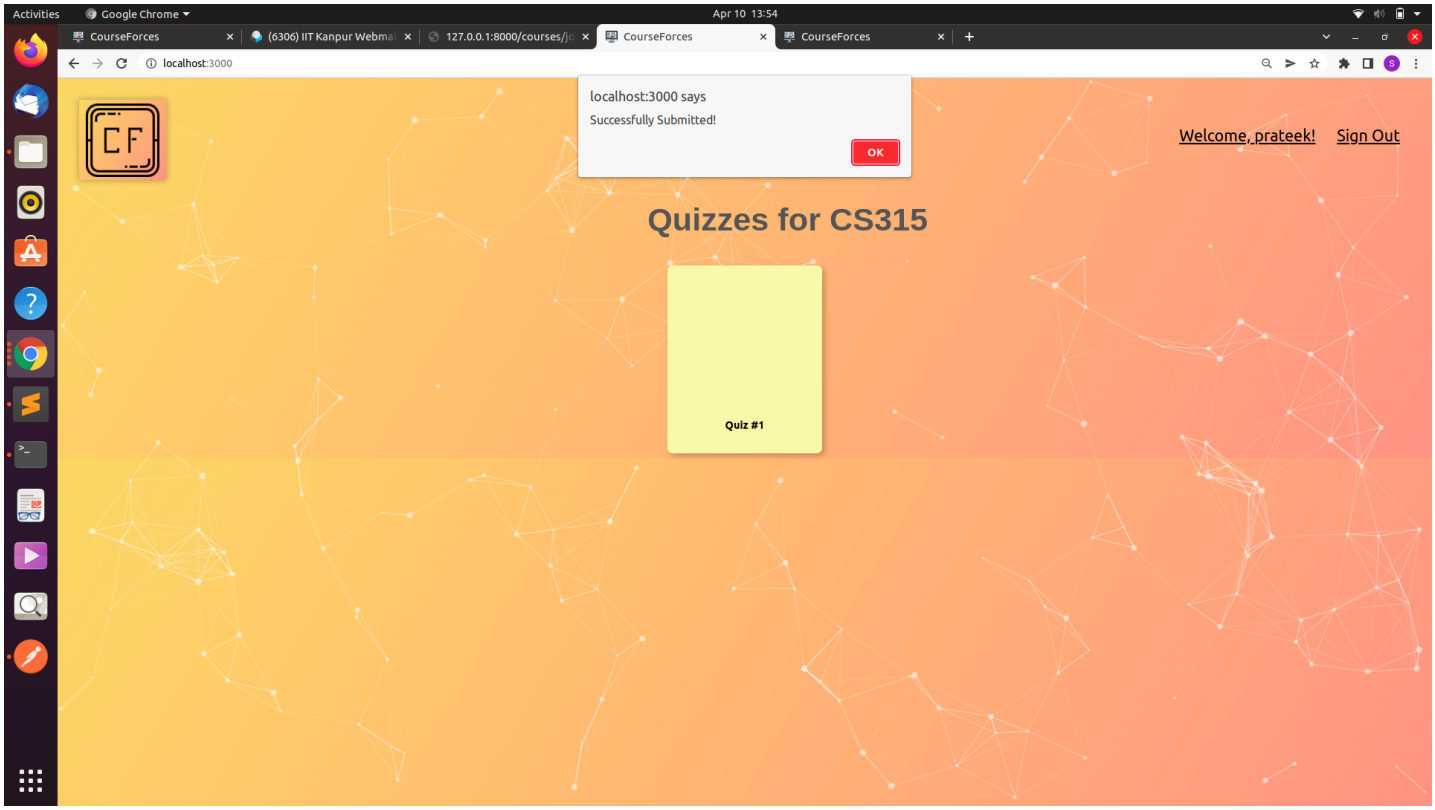


13.) Quiz Page:

The screen view when attempting a quiz.



After successful submission of a quiz :



14.) Marks List:

After checking the quiz, the professor can view the mark list of students, as shown below in the sample. This displays the name, username, and marks obtained by a user, (along with a robot face fetched from robohash.org).