College of Engineering, Construction and Living Sciences
Bachelor of Information Technology
IN608: Intermediate Application Development Concepts
Level 6, Credits 15
Practical 12 Django 6: Authentication

Due Date: 07-05-2020 at 5pm

In this practical, you will complete a series of tasks covering today's lecture. This practical is worth 2% of the final mark for the IN608: Intermediate Application Development Concepts course.

Before you start, in your practicals repository, create a new branch called **12-practical**.

Task 1

Create a Django project called `quiz`. `cd` to `quiz`, create a virtual environment & install Django. Create an app called `practical12quizcreator`. Please ensure you configure your app in `quiz/settings.py` & `quiz/urls.py`. Install `Crispy Forms` Python module by running the command: `pipenv install django-crispy-forms`. Add `crispy_form` to the `INSTALLED_APPS` in `quiz/settings.py`.

In the `practical12quizcreator` directory, create a directory called `templates` & two sub-directories called `practical12quizcreator` & `registration`. In `templates/practical12quizcreator`, create three HTML files called `base.html`, `index.html` & `details.html`. In `templates/registration`, create two HTML files called `login.html` & `signup.html`.

In `models.py`, create two model classes called `User` which extends `AbstractUser` & `Quiz` which extends `model.Model`. In `User`, declare the following field names with their types:

```
username = models.CharField(max_length=25, unique=True, blank=False)
email = models.EmailField(unique=True, blank=False)
first_name = models.CharField(max_length=200, blank=False)
last_name = models.CharField(max_length=200, blank=False)
```

For `User`, create a `__str__` method which returns `first_name` & `last_name`.

In `Quiz`, declare the following choices:

```
CATEGORIES = [(' ', 'Any Category'),
```

```
              ('9', 'General Knowledge'),
              ('18', 'Science: Computers'),
              ('21', 'Sport'),
              ('22', 'Geography'),
              ('27', 'Animals')]
DIFFICULTIES = [(' ', 'Any Difficulty'),
               ('easy', 'Easy'),
               ('medium', 'Medium'),
               ('hard', 'Hard')]
```

If you want to add more categories, view the following API: https://opentdb.com/api_category.php.

Below the two choices, declare the following field names with their types:

```
creator = models.ForeignKey(User, on_delete=models.CASCADE)
name = models.CharField(max_length=200, null=False)
category = models.CharField(max_length=200, choices=CATEGORIES, null=False)
difficulty = models.CharField(max_length=200, choices=DIFFICULTIES, null=False)
```

For `Quiz`, create a `__str__` method which returns `name`

Create a file called `forms.py`, create two form classes called `SignupForm` which extends `auth_forms.UserCreationForm` & `QuizForm` which extends `forms.ModelForm`. **Note:** please ensure you declare the following:

```
from django import forms
from django.contrib.auth import forms as auth_forms
from .models import Quiz, User
```

`SignupForm` contains the model `User` & the fields `username`, `first_name`, `last_name` & `email`. `QuizForm` contains the model `Quiz` & the fields `name`, `category` & `difficulty`.
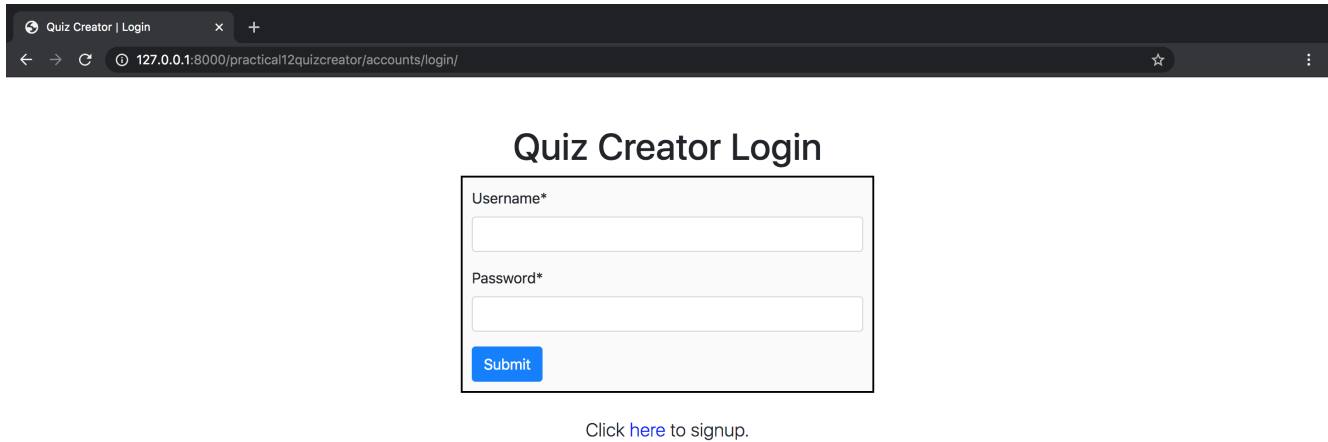
In `views.py`, create three classes called `SignupView` which extends `generic.FormView`, `IndexView` which extends `generic.FormView` & `DetailsView` which extends `generic.ListView`. `SignupView` has a method called `form_valid` which creates a `User`, logs the `User` in & redirects to `index.html`. `IndexView` has a method called `form_valid` which creates a `Quiz` & redirects to `index.html`. `DetailsView` has a method called `get_queryset` which returns a `QuerySet` containing all `Quiz` objects.

Create a file called `urls.py` in the `practical12quizcreator` app directory. In `urls.py`, set the `app_name` to `practical09dogsearch` & create three URLs which map to the `SignupView` class, `IndexView` class & `DetailsView` class in `views.py`. Create two more URLs for `LoginView` & `LogoutView` auth views.

Expected Output

Run the command `python manage.py runserver` then navigate to http://127.0.0.1:8000/practical12quizcreator/

`login.html` - login using a valid username & password (both fields are required). If successful, the user is redirected to `index.html`. If a user wants to signup, they can click the `here` link which will redirect the user to `signup.html`.
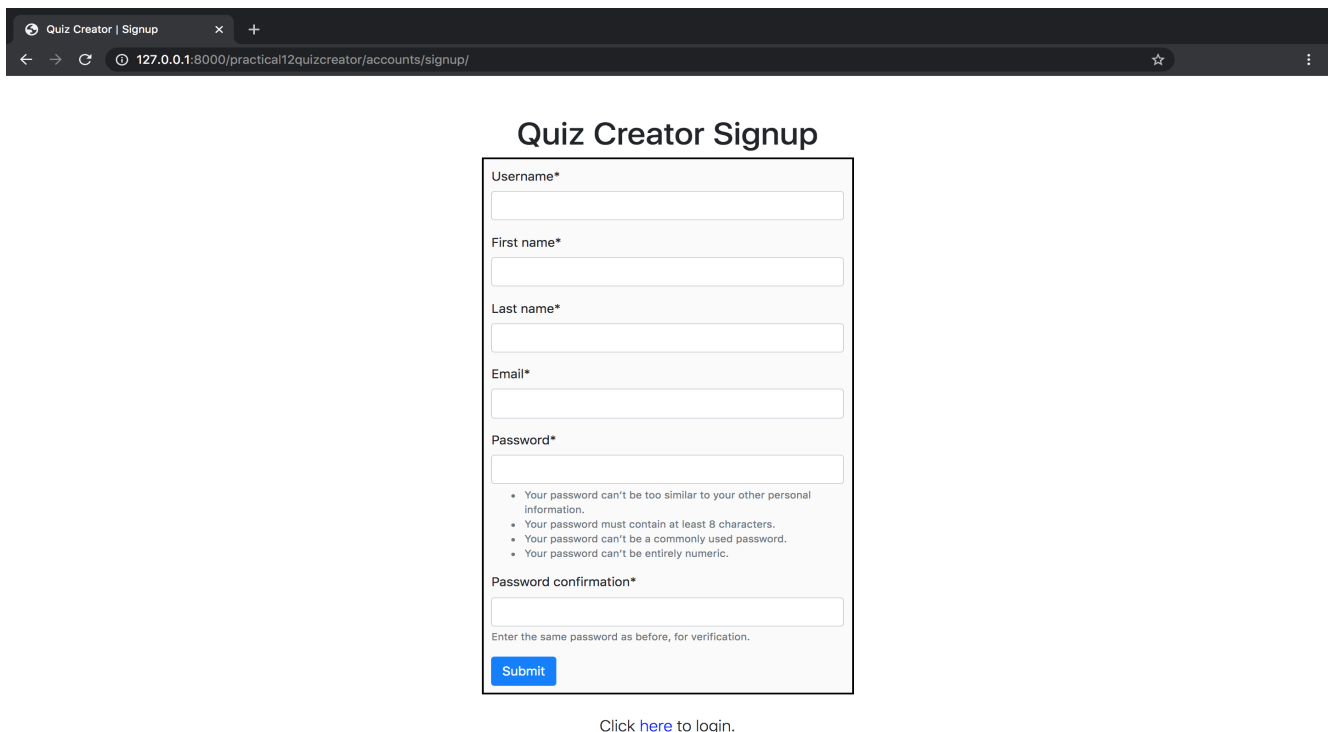
signup.html - all fields are required. If an email already exists, then the user must enter a different email. Password has additional validation:

- Can not be too similar to your other personal information

- Must contain at least 8 characters

- Can not be a commonly used password

- Can not be entirely numeric

If successful, the user is redirected to **index.html**.

index.html - all fields are required. If a user navigates to index.html & is not authenticated, the user is redirected to login.html. Select drop downs are populate via choices in models.py. If the form is valid, a Quiz is created. If a user wants to logout, they can click the Logout link which will redirect the user to login.html.

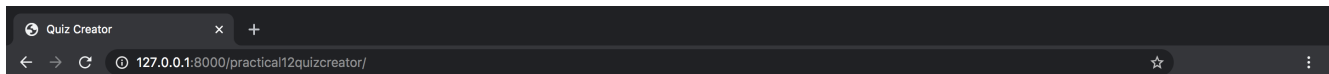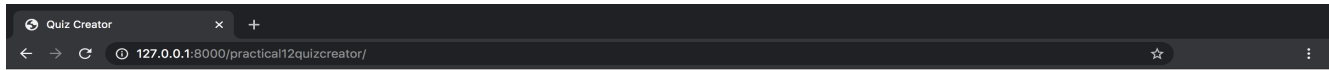**details.html** - display all `Quiz` objects in a nicely formatted HTML table.
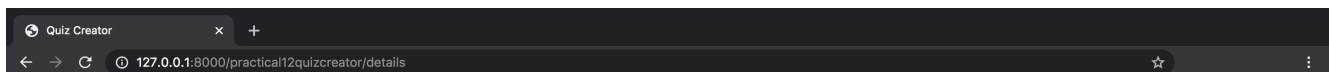
Deployment link: https://int-app-dev-practical-12.herokuapp.com/practical12quizcreator/

Task 2

Write the following unit tests:

- Signup is successful & unsuccessful

- Login is successful & unsuccessful

- Quiz is successfully created

Write the following end-to-end test:

- Login, create a quiz & view all quizzes