Homework #10 (Django) - Group + Individual

Part 1 (0 points, but required)

Task 1

This part is for you to practice developing locally on your machine.

- Install a VM on VirtualBox.
  - Any flavor is fine. Ubuntu 14.04 or 16.04 is recommended.
  - Mint Mate 17.3 (https://linuxmint.com/edition.php?id=206) is MOST recommend. It is based on Ubuntu 14.04 and it comes with VirtualBox tools installed, so you don't need to struggle with this.

Part 2 (3 points)

Task 1

Install postgresql (you have done this in a previous homework).

Task 2

For this to work with Django, you will need to update 'pg_hba.conf'

- Locate pg_hba.conf. For Ubuntu 14.04, it should be located at '/etc/postgresql/9.3/main/pg_hba.conf'. Sometimes it can be located in '/var/lib/pgsql/data/*'
- Open that file.
- Find line that says: host all all 127.0.0.1/32 md5
- Change the word 'md5' to 'trust'.
- Save the file.
- Restart postgresql.

Task 3

Setup your environment using 'virtualenv'.

- https://virtualenv.pypa.io/en/stable/

Task 4

- Install current release of Django (Your job to determine which one to install).

    - https://www.djangoproject.com/start/
- Create a Django application.
- Add your database information.
    - **NOTE: Django works out the box with SQLite database and doesn't require a database. However, we want you to use postgresql.**
- Make sure your installation can start (It will not start if Postgresql database is not setup correctly).
- Perform your migrations.
- Create an admin user.
- Log into the admin section of Django.

Task 5

Upload your code to bitbucket.

- Develop and deploy from this installation.
- **NOTE: Every team member should have at least 5 'significant' commits, as with the previous homework. We will check this. As a team, you will lose 2 points [part 2] if we do not see this. If participation is a problem in your group, discuss this with us.**

Task 6 **(IMPORTANT)**

To ensure that your group starts the homework early, we will require this portion of the homework to be done by class time on Wednesday 11/1. **Before class starts, one of your group members must show me that you can successfully log into the admin section of Django. Otherwise, your group will lose 2 points.**

Part 3 (7 points)

This section is to learn to use the Django by creating a 'blog type' application.
NOTE: Development should* take place on your local machine, however, similar steps will be used when deploying to production.

Unlike MEANJS, Django does not come with 'pre-built' articles module. You will need to create an app. This homework will be replicating the site you did in homework #9. You will be using Bootstrap for this homework.

Task 1

- Create a model called 'Articles'.
    - id: unique identifier for the articles (search whether you need to add this to your model).
    - slug-field: Human readable unique identifier for the articles.
        - **NOTE: This field should be unique.**

    - content: Text of the article.
    - tags: A list of tags that the user enters when creating the article.
    - author: **[FK field]** Who created the article.
    - views: **[int]** Number of times the article has been viewed.
    - comments: **[int]** Number of comments for this article.
    - created: **[datetime]** Timestamp when article was created.
    - updated: **[datetime]** Timestamp when article was updated.
- Create a model called 'Comments'.
    - id: unique identifier for the comment.
    - article: **[FK field]** id of the article this comment belongs to.
    - author: **[FK field]** author of the article.
    - comment: Text of the comment.
    - created: **[datetime]** Timestamp when the comment was created.

Task 2

Create the following pages.
NOTE: These pages must use server side 'templates'.
NOTE: You mush use class based views (http://ccbv.co.uk/ - Use the correct version based on what you have installed).

- create
    - This page allows a user to create an article.
    - Page should only be shown if they are logged in.
    - Only appropriate fields should be shown.
- update
    - This page allows a user to update an article (i.e. user must be logged in to see this).
    - **NOTE: Only the original creator should be able to update it.**
    - Only the appropriate fields should be editable.
- list **(landing page for the site)**
    - Divide your page into 2 (content + sidebar on the right).
    - Shows a list of all articles in a grid view (3 rows) on the content side.
        - Each cell includes the title, views, few characters from the body, author and date created.
        - Articles should be ordered by most recent article first.
    - A searchbar that searches the articles (by title).
        - **NOTE: Search should be done server side.**
    - Sidebar should show unique tags from all the articles.
        - Clicking on a tag should show only articles with that tag.
        - **NOTE: This should be done server side.**
    - A button at the top to reset all articles.
    - A button at the top (if they are logged in), to create an article.
- view

    - Shows the article.
    - Shows all the information about the article.
    - Shows the comments at the bottom (most recent first).
        - Form entry to add a comment at the top of comments (bottom of article) (if they are logged in).
        - **BONUS: Get up to a maximum of 2 points to offset homework #9!!!**
            - Implement the comments urls using Django Rest Framework (http://www.django-rest-framework.org/)
- delete
    - Not necessary.

Task 3

- Brand your site.
    - Add a 'cool' name for your site.
    - Make your site presentable.

Task 4 (1 point)

Deploy your application on your machine.

- https://www.digitalocean.com/community/tutorials/how-to-serve-django-applications-with-apache-and-mod_wsgi-on-ubuntu-14-04
- **NOTE: You should use port 8080**
- **NOTE: Your project should be in '/opt/django/'**

Task 5

Send your URL with your team's netids to gmadey@nd.edu and qzhi@nd.edu, with subject line CSE 40613 - group-number (i.e, your machine's hostname) - HW10

Bonus (Group)[0 or 2 points i.e. all or nothing. Due 11/2 at 11:59PM]

Create a report (Max 3 pages) with the following

- MEANJS
    - Create a visual (with a key) of how MEANJS works.
        - This must include: Main frameworks, main libraries, concepts like CRUD, Routes, Middleware, Controllers, etc.
- Django (This will also allow you to get ahead on HW#10).
    - Create a visual (with a key) of how Django works (Django Rest Framework not necessary here).

        - This must include: Routes, Class based views, functional views, middleware, templates, etc.
- Comparison (with a short explanation for each).
    - 5 things that are similar.
    - 5 things that are different.
        - Example: MEANJS is a stack. Django is a Framework. A stack combines (integrates) several different technologies and Frameworks that can work independently. For example, AngularJS can work independently from Express, but MEANJS brings them together without extra configuration.

Send your URL with your team's netids to gmadey@nd.edu and qzhi@nd.edu and samuel.w.njoroge.5@nd.edu, with subject line CSE 40613 - group-number (i.e, your machine's hostname) - Report