Изображение выглядит как текст

Автоматически созданное описание

…

Create urls.py in recipes, urls.py in addrecipe.

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание Изображение выглядит как текст

Автоматически созданное описание Изображение выглядит как текст, снимок экрана, экран

Автоматически созданное описание

Update settings.py in recipes:

Изображение выглядит как текст

Автоматически созданное описание

Create models for the app: Category, Recipe, Comment in models.py in addrecipe:

Изображение выглядит как текст

Автоматически созданное описание

-- Create model Category

CREATE TABLE "addrecipe\_category" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name" varchar(40) NOT NULL UNIQUE, "description" varchar(200) NOT NULL);

--

-- Create model Recipe

CREATE TABLE "addrecipe\_recipe" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "subject" varchar(700) NOT NULL, "lowsugar" bool NOT NULL, "cooking\_time" integer NOT NULL, "last\_updated" datetime NOT NULL, "category\_id" bigint NOT NULL REFERENCES "addrecipe\_category" ("id") DEFERRABLE INITIALLY DEFERRED, "starter\_id" integer NOT NULL REFERENCES "auth\_user" ("id") DEFERRABLE INITIALLY DEFERRED);

-- - Add field rec\_description to recipe

- Alter field subject on recipe

--

-- Create model Comment

CREATE TABLE "addrecipe\_comment" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "message" text NOT NULL, "created\_at" datetime NOT NULL, "updated\_at" datetime NULL, "created\_by\_id" integer NOT NULL REFERENCES "auth\_user" ("id") DEFERRABLE INITIALLY DEFERRED, "recipe\_id" bigint NOT NULL REFERENCES "addrecipe\_recipe" ("id") DEFERRABLE INITIALLY DEFERRED, "updated\_by\_id" integer NULL REFERENCES "auth\_user" ("id") DEFERRABLE INITIALLY DEFERRED);

CREATE INDEX "addrecipe\_recipe\_category\_id\_c4c08ee9" ON "addrecipe\_recipe" ("category\_id");

CREATE INDEX "addrecipe\_recipe\_starter\_id\_ee39c586" ON "addrecipe\_recipe" ("starter\_id");

CREATE INDEX "addrecipe\_comment\_created\_by\_id\_010d5933" ON "addrecipe\_comment" ("created\_by\_id");

CREATE INDEX "addrecipe\_comment\_recipe\_id\_ba91981a" ON "addrecipe\_comment" ("recipe\_id");

CREATE INDEX "addrecipe\_comment\_updated\_by\_id\_9bf3b033" ON "addrecipe\_comment" ("updated\_by\_id");

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

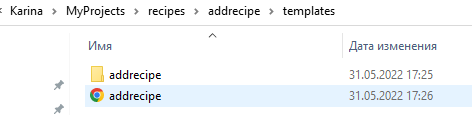
HTML: add app name in urls,

Изображение выглядит как текст

Автоматически созданное описание

Change views:

Изображение выглядит как текст

Автоматически созданное описание 

Изображение выглядит как текст

Автоматически созданное описание Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Do the first test:

Изображение выглядит как текст

Автоматически созданное описание

Go to [fonts.google.com](https://fonts.google.com/), type “Django Boards” or whatever name you gave to your project then click on **apply to all fonts**. Browse a bit, find one that you like.

Изображение выглядит как текст

Автоматически созданное описание

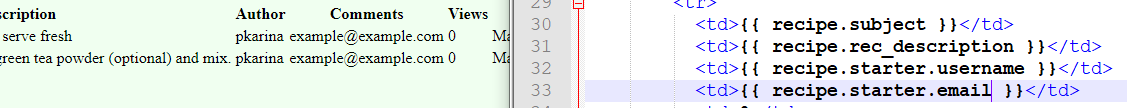


Изображение выглядит как текст

Автоматически созданное описание

The reason to use board.topics.all instead of just board.topics is because board.topics is a **Related Manager**, which is pretty much similar to a **Model Manager**, usually available on the board.objects property. So, to return all topics associated with a given board, we have to run board.topics.all(). To filter some data, we could do board.topics.filter(subject\_\_contains='Hello').

Fields in Model’s object may be accessed and displayed in html:



Update URLs, add pages and do the form.

Изображение выглядит как текст

Автоматически созданное описание

Add tests for the URLs, form in tests.py

**LOGIN, Sign in**

Account management will be in a separate app.



Add it in the in the **settings.py** file in installed apps, add in urls.py of recipes:



Do the function signup in views.py in accounts, use Django’s built-in **UserCreationForm**.

Изображение выглядит как текст

Автоматически созданное описание

Do the html template. Do tests.py in accounts.

Update signup function, check the new user is created and it is redirected to home page.

Create file forms.py in accounts to customize default form – add email field, update tests.

Change the project structure by separating tests in the folder.

**LOGOUT** – add url in recipes/urls.py: and use LogoutView.as\_view() - a Django’s class-based view. In recipes/settings.py: 

Add links and buttons in html.

**LOGIN**

Inside the as\_view() we can pass some extra parameters, so to override the defaults. In this case, we are instructing the **LoginView** to look for a template at **login.html**.

Add redirect in settings as for logout.

SKIP TEMPLATE TAGS

**Password Reset**

-emails will only be displayed to console, no real emails sent

Add to settings.py:

Изображение выглядит как текст

Автоматически созданное описание

Password reset process requires four views:

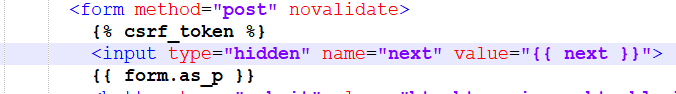
* A page with a form to start the reset process;
* A success page saying the process initiated, instructing the user to check their spam folders, etc.;
* A page to check the token sent via email;
* A page to tell the user if the reset was successful or not.

The views are built-in, just need to add the routes to the **urls.py** and create the templates.

**Password change**.

Tests updated, tests for password reset email confirmation fail (the whole file – 3 errors).

Update the views.py in the addrecipe: the new recipe may be added by logged in user only. Now while not logged in uiser wants to add new recipe, he is redirected to login page. By adding the line (<input>…. With the value of {{next}}…):



* After logging in this user will be redirected where he was – at the add new recipe page.

Set the User authentication in the addrecipe/views.py.new\_recipe file by choosing the correct user from the DB.

**COMMENTS for the recipe**

Add url to addrecipe/urls.py

Add function in views.py: recipe\_comments

Add html page.

Add tests and run the tests for ‘addrecipe’ app:

python manage.py test addrecipe

or the test for the new file:

python manage.py test addrecipe.tests.test\_view\_recipe\_comments

or the test for the new functions:

python manage.py test addrecipe.tests.test\_view\_recipe\_comments.*class.function*

Add the link to this page from the ‘recipes’ page.

REPLY – Comment view

Add url, new form in forms.py, view in views.py, add all links to corresponding pages.

Some tests fail, but the links are ok.

**UPDATE THE main ADDRECIPE page**

Add to models.py – to Category :

Functions for queries to get the number of recipes, the last comment.

Изображение выглядит как текст

Автоматически созданное описание

(Above is from the part-6 of tutorial <https://simpleisbetterthancomplex.com/series/2017/10/09/a-complete-beginners-guide-to-django-part-6.html> )

Skip pagination and Edit comment features. Skip class-based views.

Removed email from exposure: Изображение выглядит как текст

Автоматически созданное описание