

The background of the slide features a photograph of a silver laptop on a light-colored desk. The laptop is open, and its screen displays a colorful abstract pattern. To the right of the laptop, a white computer mouse is visible. The entire image is overlaid with several thin, dark grey geometric lines, including a prominent diagonal line on the left and a curved line on the right. The text is positioned over these elements.

FINAL PROJECT:

Ami's Kitchen

PREPARED BY
ABDUL KAREEM
ASHHAL AHMED
ALI HYDER SHAH

CONTENTS

•.... Introduction	3
•.... Overview of the Python Recipe Organizer	4
•.... Installation Guide	5
•.... Appendices	6

Introduction:

In today's busy world, it's tough to find time and ideas for tasty meals. But with Ami's Kitchen, cooking becomes a breeze. It's made for everyone, whether you're a cooking pro or just starting out.

This documentation is here to guide you through using Ami's Kitchen. Whether you want to sort your recipes, plan your weekly meals, or find new dishes to try, you'll learn it all right here, in simple terms. Let's make cooking fun and stress-free with Ami's Kitchen!

Overview of the Python Ami's Kitchen:

Python Ami's Kitchen is like having your own cooking helper right on your computer! It's a special program made to help you with cooking and preparing meals. Whether you're a beginner or a pro in the kitchen, this program is here to make things easier for you.

Here are some cool things it can do:

1. **Organize Recipes:** You can keep all your recipes neat and tidy in one place. No more searching through messy piles of papers!
3. **Ingredients:** It keeps track of what ingredients you have at home and even makes a shopping list for you based on the recipes you choose. No more forgetting to buy something important!
4. **Find New Recipes:** It suggests new recipes based on what you like to eat. So, if you're feeling adventurous, you can try something new and exciting!

Python Ami's Kitchen makes cooking fun and stress-free. Whether you're cooking for yourself, your family, or friends, it's like having a culinary expert right by your side. So, let's get cooking and enjoy delicious meals together with Python Ami's Kitchen!

Installation Guide:

1. *Installation guide about Django:*

To install the Django framework using pip, follow these steps:

- Open your command-line interface (CLI). This could be Command Prompt on Windows, Terminal on macOS, or any other terminal emulator you prefer.
- Type the “pip install Django” command and press Enter.

2. *Installation guide about pip-install pillow:*

To install the Pillow library using pip, follow these steps:

- Open your command-line interface (CLI). This could be Command Prompt on Windows, Terminal on macOS, or any other terminal emulator you prefer.
- Type the “Python -m pip install Pillow” command and press Enter.

NOTE: If no error messages are displayed, Pillow is successfully installed and ready to use in your Python environment.

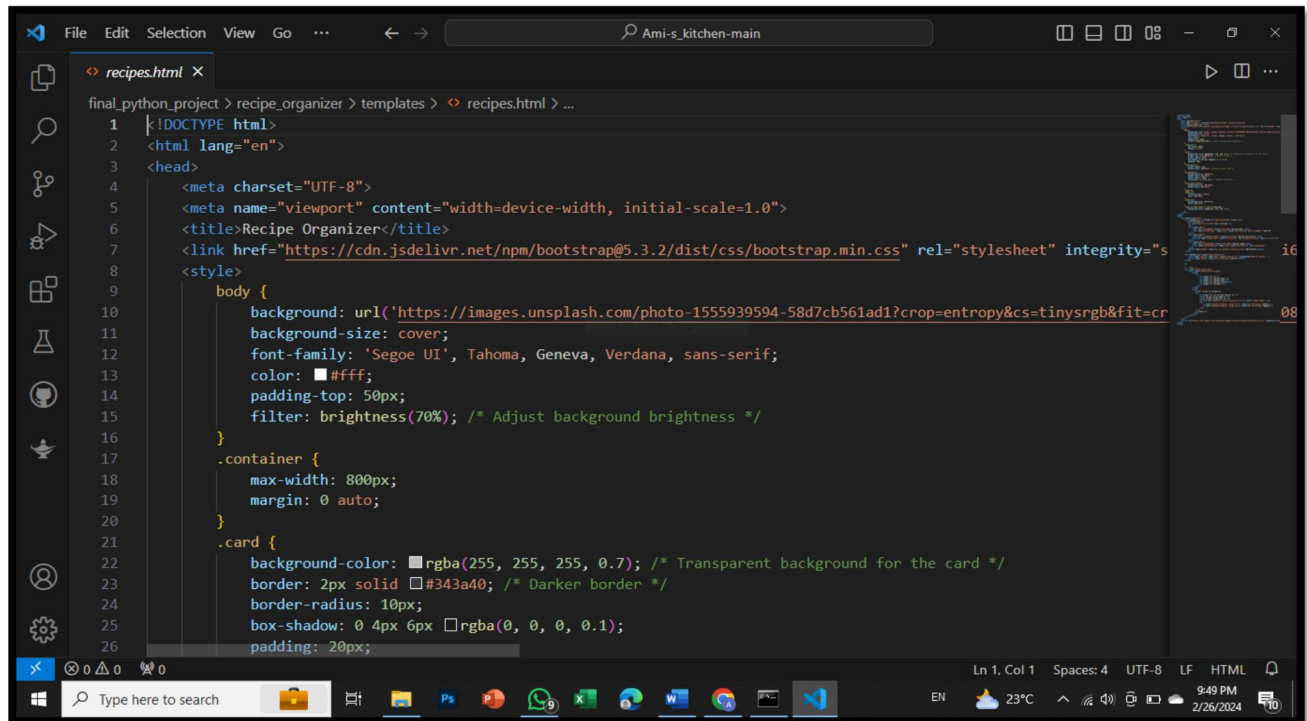
3. *Installation guide about boot up server:*

- Open your command-line interface (CLI). This could be Command Prompt on Windows, Terminal on macOS, or any other terminal emulator you prefer.
- Type the “python manage.py runserver” command and press Enter.

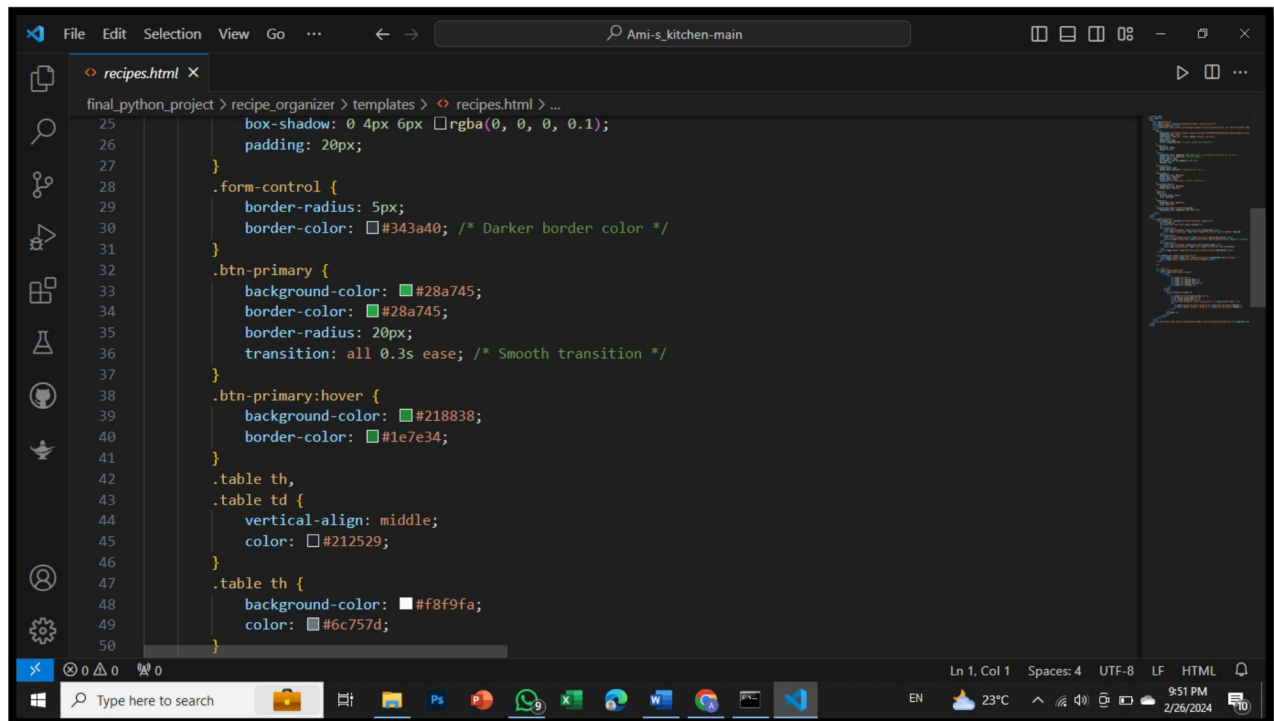
Appendices:

FRONTEND-CODING:

reciepe.html



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Recipe Organizer</title>
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="s
8 <style>
9   body {
10     background: url('https://images.unsplash.com/photo-1555939594-58d7cb561ad1?crop=entropy&cs=tinsrrgb&fit=cr
11     background-size: cover;
12     font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
13     color: #ffff;
14     padding-top: 50px;
15     filter: brightness(70%); /* Adjust background brightness */
16   }
17   .container {
18     max-width: 800px;
19     margin: 0 auto;
20   }
21   .card {
22     background-color: rgba(255, 255, 255, 0.7); /* Transparent background for the card */
23     border: 2px solid #343a40; /* Darker border */
24     border-radius: 10px;
25     box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
26     padding: 20px;
```



```
25     box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
26     padding: 20px;
27   }
28   .form-control {
29     border-radius: 5px;
30     border-color: #343a40; /* Darker border color */
31   }
32   .btn-primary {
33     background-color: #28a745;
34     border-color: #28a745;
35     border-radius: 20px;
36     transition: all 0.3s ease; /* Smooth transition */
37   }
38   .btn-primary:hover {
39     background-color: #218838;
40     border-color: #1e7e34;
41   }
42   .table th,
43   .table td {
44     vertical-align: middle;
45     color: #212529;
46   }
47   .table th {
48     background-color: #f8f9fa;
49     color: #6c757d;
50   }
```

update.html

```
update.html X
final_python_project > recipe_organizer > templates > update.html > ...
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1">
6     <title>Bootstrap demo</title>
7
8     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="s
9   </head>
10  <body>
11    <div class="container mt-5">
12
13
14    <form method="post" enctype="multipart/form-data" class="card col-6 mx-auto p-3">
15      {% csrf_token %}
16      <h2>UPDATE RECIPE</h2>
17      <div class="mb-3">
18        <label for="exampleInputEmail1" class="form-label">RECIPE age</label>
19        <input name="recipe_name" value="{{recipe.recipe_name}}" type="text" class="form-control">
20      </div>
21      <div class="mb-3">
22        <label for="exampleInputPassword1" class="form-label">RECIPE DESCRIPTION</label>
23        <textarea name="recipe_text" class="form-control">{{recipe.recipe_text}}</textarea>
24      </div>
25      <div class="mb-3">
26        <label for="exampleInputPassword1" class="form-label">RECIPE IMAGE</label>
```

```
update.html X
final_python_project > recipe_organizer > templates > update.html > ...
11 <div class="container mt-5">
12
13
14    <form method="post" enctype="multipart/form-data" class="card col-6 mx-auto p-3">
15      {% csrf_token %}
16      <h2>UPDATE RECIPE</h2>
17      <div class="mb-3">
18        <label for="exampleInputEmail1" class="form-label">RECIPE age</label>
19        <input name="recipe_name" value="{{recipe.recipe_name}}" type="text" class="form-control">
20      </div>
21      <div class="mb-3">
22        <label for="exampleInputPassword1" class="form-label">RECIPE DESCRIPTION</label>
23        <textarea name="recipe_text" class="form-control">{{recipe.recipe_text}}</textarea>
24      </div>
25      <div class="mb-3">
26        <label for="exampleInputPassword1" class="form-label">RECIPE IMAGE</label>
27        <input name="recipe_picture" value="{{recipe.recipe_picture}}" type="file" class="form-control">
28      </div>
29      <button type="submit" class="btn btn-primary">UPDATE IT</button>
30    </form>
31
32    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6Rzs
33  </body>
34 </html>
35
```

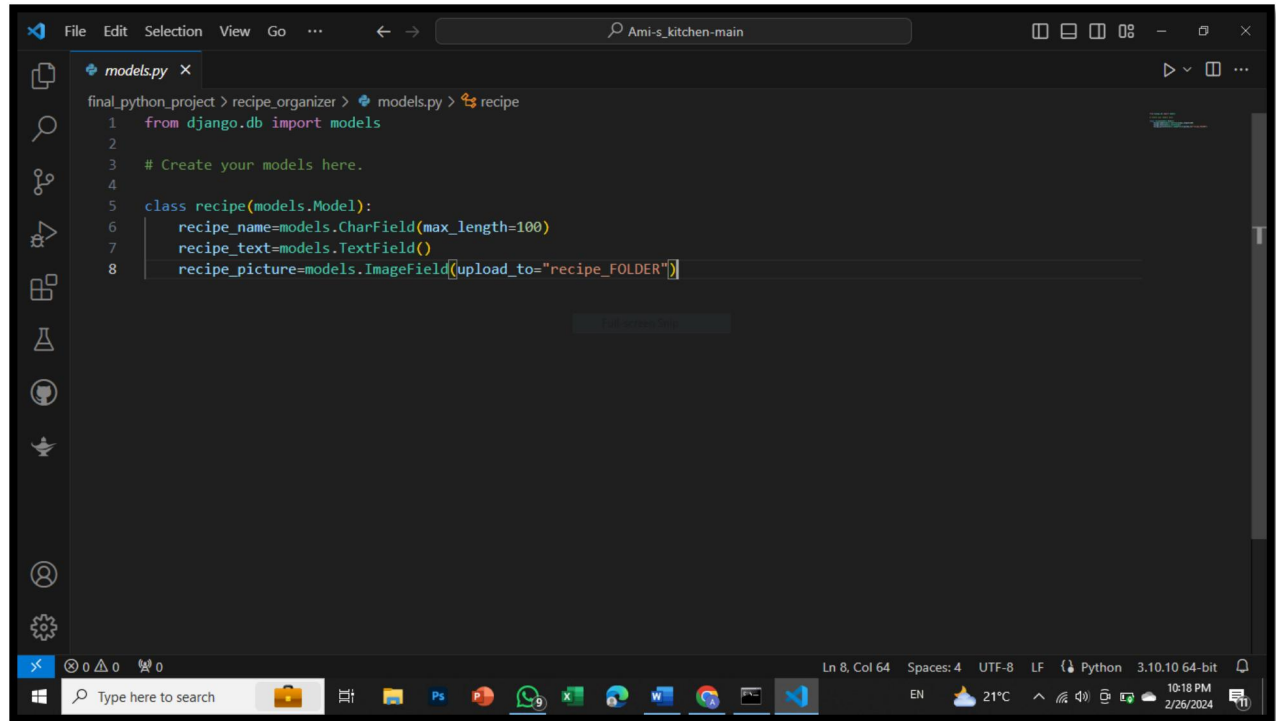
BACK-END CODING:

Views.py

```
views.py
final_python_project > recipe_organizer > views.py > ...
1 from django.shortcuts import render
2 from django.http import HttpResponseRedirect
3 from .models import *
4 from django.shortcuts import render, redirect
5 from django.shortcuts import get_object_or_404
6 # Create your views here.
7 def recipes(request):
8     if request.method=="POST":
9         data=request.POST
10        recipe_name=data.get("recipe_name")
11        recipe_text=data.get("recipe_text")
12        recipe_picture=request.FILES.get("recipe_picture")
13
14        recipe.objects.create(
15            recipe_name=recipe_name,
16            recipe_picture=recipe_picture,
17            recipe_text= recipe_text
18        )
19
20
21    datalist=recipe.objects.all()
22    if request.GET.get('searchs'):
23        datalist=datalist.filter(recipe_name__icontains = request.GET.get('searchs'))
24    get_data={'recipes':datalist}
25    return render(request,'recipes.html',get_data)
26
```

```
views.py
final_python_project > recipe_organizer > views.py > ...
24 get_data={'recipes':datalist}
25 return render(request,'recipes.html',get_data)
26
27 def delete_recipe(request, id):
28     recipe_instance = recipe.objects.get(id=id)
29     recipe_instance.delete()
30     return redirect('recipes')
31
32 def update_recipe(request, id):
33     recipe_instance = get_object_or_404(recipe, id=id)
34
35     if request.method == "POST":
36         data = request.POST
37         recipe_name = data.get("recipe_name")
38         recipe_text = data.get("recipe_text")
39         recipe_picture = request.FILES.get("recipe_picture")
40
41         recipe_instance.recipe_name = recipe_name
42         recipe_instance.recipe_text = recipe_text
43
44         if recipe_picture:
45             recipe_instance.recipe_picture = recipe_picture
46
47         recipe_instance.save()
48         return redirect('recipes')
49
50     get_data = {'recipe': recipe_instance}
```


models.py



```
models.py
final_python_project > recipe_organizer > models.py > recipe
1 from django.db import models
2
3 # Create your models here.
4
5 class recipe(models.Model):
6     recipe_name=models.CharField(max_length=100)
7     recipe_text=models.TextField()
8     recipe_picture=models.ImageField(upload_to="recipe_FOLDER")
```

Ln 8, Col 64 Spaces: 4 UTF-8 LF Python 3.10.10 64-bit

10:18 PM 2/26/2024