

Visvesvaraya Technological University, Belagavi – 590010



**Mini Project on Python
Cake Shop Management System
*Submitted by***

Alonie Jane Crasta

4SO17CS011

Under the guidance of

Ms. Eden Sequeira
(Assistant Professor, CSE Department)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**ST JOSEPH ENGINEERING COLLEGE
Vamanjoor, Mangaluru -575028, Karnataka
2019-2020**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, SJEC
Assignment Rubrics& Marks Sheet -2019-20

Class: VISemA section

Course Title: Python Application Programming

Course Code: 17CS664

Name of the Student: Aloinie Jane Crasta

USN: 4SO17CS011

Name of the Course instructor: **Ms. Eden Sequeira**

NOTE: Students are required to create a WebApplication usingpython programming and Django Framework for Assignment.

Topic can be chosen by students

Submission by: May 3rd 2020

Criteria	Below Average 1	Basic 2	Satisfactory 3	Good 4	Exceptional 5	Marks
Implementation (1)	Inefficient coding, Exceptions are not handled effectively, no database used, Framework features not used effectively	Efficient coding, Exceptions are not handled effectively, no database used, Framework features not used effectively	Efficient coding, Exceptions are handled effectively, no database used, Framework features not used effectively	Efficient coding, Exception are handled effectively, Database is used effectively, Framework features not used effectively	Efficient coding, Exception are handled effectively, Database is used effectively, Framework features used effectively	
Design (1)	Very poor design no proper alignments and only images used	Design has no look and feel of web page with no proper alignments and only images used	Design has a look and feel of web page with no proper alignments and standard design	Design has a look and feel of web page with proper alignments and standard design	Design has a look and feel of web page with proper alignments and exceptional design	
					Total Marks (10)	

Signature of the course instructor with date

CONTENTS

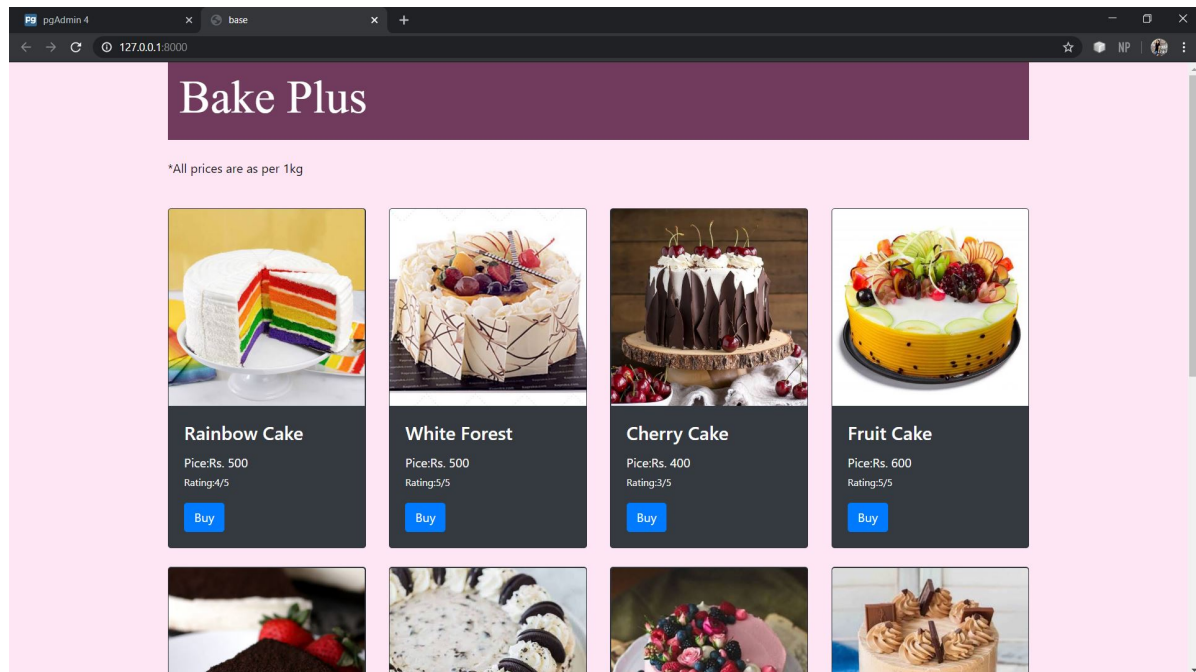
1. Introduction.....	01
2. Implementation.....	02-18
2.1 Urls code.....	02
2.2 Views Code	03-05
2.3 Settings code	05-08
2.4 Model Code.....	08
2.6 Template code	09-18
3. Screenshots of Design	18-23

1. Introduction

Cake Shop Management System App allows the users to choose a cake of their choice and buy the cake. It provides online ordering of the cake.

The cake shop owner can add cakes, delete the details or update the details.

The user first chooses his/her cake .The user provides contact details and the quantity of cake in kg .A invoice is created which allows users to verify the order and buy the cake.



2. Implementation

2.1 Urls Code

i. Main urls.py

```
from django.contrib import admin
from django.urls import path, include
from Cakeadmin import views
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path("", views.home),
    path('Cakeadmin/', include('Cakeadmin.urls')),
    path('details/<int:id>', views.details),
    path('payment/<int:id>', views.payment),
    path('thankyou', views.thankyou),
]
urlpatterns = urlpatterns + static(settings.MEDIA_URL,
    document_root=settings.MEDIA_ROOT)
```

ii. Cakeadmin/urls.py

```
from django.urls import path
from Cakeadmin import views
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('login', views.login),
    path('display', views.display),
    path('add', views.add),
    path('edit/<int:id>', views.edit),
    path('edit/update/<int:id>', views.update),
    path('delete/<int:id>', views.delete),
]
```

2.2 Views Code

```
from .models import CakesItems, Customers

# Create your views here.

def home(request):
    items = CakesItems.objects.all()
    return render(request, "Home.html", {'items': items})

def login(request):
    if request.method == "POST":

        name = request.POST['name1']
        pwd = request.POST['pass1']

        if name == 'admin' and pwd == 'pass':
            try:
                return redirect(display)
            except:
                pass
        return render(request, "Login.html")

def display(request):
    items = CakesItems.objects.all()
    return render(request, "Display.html", {'items': items})

def add(request):
    if request.method == "POST":
        form = CakeItemsForm(request.POST, request.FILES)
        if form.is_valid():
            try:
                form.save()
                return redirect(display)
            except:
                pass
    else:
        form = CakeItemsForm()
    return render(request, "Add.html", {'form': form})
```

```
def edit(request,id):
    cakeitem=CakesItems.objects.get(id=id)
    return render(request,"Update.html",{ 'cakeitem':cakeitem})

def update(request,id):
    cakeitem = CakesItems.objects.get(id=id)
    # form=CakeItemsForm(request.POST,request.FILES,instance=cakeitem)
    if request.method == "POST":
        des = request.POST['descrip']
        p = request.POST['price']
        q = request.POST['quan']

        CakesItems.objects.filter(id=id).update(Description=des,price=p,quantity=q)
    # if form.is_valid():
    #     form.save()
    return redirect(display)

    return render(request,"Update.html",{ 'cakeitem':cakeitem})

def delete(request,id):
    cakeitem = CakesItems.objects.get(id=id)
    cakeitem.delete()
    return redirect(display)

def details(request,id):
    if request.method=="POST":
        form = CustomersForm(request.POST)
        if form.is_valid():
            try:
                q=form.cleaned_data['quan']
                n=form.cleaned_data['name']
                add = form.cleaned_data['address']
                p = form.cleaned_data['phoneno']
                e= form.cleaned_data['email']

                itemid = CakesItems.objects.get(id=id)
                to=(float)(q * itemid.price)
                record =
Customers.objects.create(name=n,phoneno=p,email=e,address=add,quan=q,totalamt=to,cakeid=id
)
                record.save()
                i=record.id
```

```
        return redirect(payment,id=i)
    except:
        pass
    else:
        form = CustomersForm()
        itemid = CakesItems.objects.get(id=id)
        return render(request, "Buy.html", {'itemid': itemid,'form':form})

def payment(request,id):
    cus=Customers.objects.get(id=id)
    itemid=cus.cakeid
    item = CakesItems.objects.get(id=itemid)
    return render(request, "Payment.html", {'cus': cus,'item':item})

def thankyou(request):
    return render(request, "Thankyou.html")
```

2.3 Settings code

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
from typing import Union

BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'm#ijh0+f^kxf9g)l+jqtry9_*_)&2!*x0tqciwq99^7e+!2n_s'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
```



```
'Cakeadmin',
'django.contrib.admin',
'django.contrib.auth',
'django.contrib.contenttypes',
'django.contrib.sessions',
'django.contrib.messages',
'django.contrib.staticfiles',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'cake.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS':
[os.path.join(BASE_DIR, 'templates'), os.path.join(BASE_DIR, 'Cakeadmin/templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
                'django.template.context_processors.media',
            ],
        },
    },
]

WSGI_APPLICATION = 'cake.wsgi.application'

# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases
```

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'cake',
        'USER': 'postgres',
        'PASSWORD': '12345',
        'HOST': 'localhost',
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

# Internationalization
# https://docs.djangoproject.com/en/3.0/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

# Static files (CSS, JavaScript, Images)
```

<https://docs.djangoproject.com/en/3.0/howto/static-files/>

```
STATIC_URL = '/static/'
STATICFILES_DIRS=[
    os.path.join(BASE_DIR,'Cakeadmin/Static')
]
STATIC_ROOT=os.path.join(BASE_DIR,'Staticcontent')

MEDIA_ROOT=os.path.join(BASE_DIR,'media/')
MEDIA_URL='/media/'
```

2.4 Model Code

```
from django.db import models

# Create your models here.
Choices = (
    (0.50, '0.5'),
    (1.00, '1'),
    (2.00, '2'),
    (3.00, '3'),
)

class CakesItems(models.Model):
    Description=models.CharField(max_length=200)
    img=models.ImageField(upload_to='pics')
    quantity=models.IntegerField()
    price=models.IntegerField()

class Customers(models.Model):
    name=models.CharField(max_length=100)
    phoneno=models.CharField(max_length=10)
    email=models.EmailField()

totalamt=models.DecimalField(max_digits=8,decimal_places=2,null=True,blank=True,default=0.00)
quan=models.DecimalField(default='1.00',choices=Choices,max_digits=3,decimal_places=2)
address=models.TextField()
cakeid=models.IntegerField(default='1',blank=True)
```

2.5 Template code

i Login.html

```
<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Admin Login</title>
  <style>

  </style>
</head>
<body>
  {% block content%}

  <div style="text-align:center place-items: center vertical-align: middle">
    <form method="POST" action="login" >
      {% csrf_token %}

      <div class="form-group">
        <label >Name:</label>
        <div class="row">
          <div class="col-lg-3">
            <input type="text" class="form-control" id="name1" name="name1">
          </div>
        </div>
      </div>
      <div class="form-group">
        <label >Password</label>
        <div class="row">
          <div class="col-lg-3">
            <input type="password" class="form-control" id="Pass1" name="pass1">
          </div>
        </div>
      </div>
      <button type="submit" class="btn btn-primary">Submit</button>
    </form>
  </div>
  {% endblock%}
</body>
</html>
```

ii Add.html

```
<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Add</title>
  <style>
    .wrapper {
      text-align: center;
    }

  </style>
</head>
<body>
  {% block content %}
    <form method="POST" class="post-form" action="add" enctype="multipart/form-data">
      {% csrf_token %}
      <br>
      <br>

      <h4>Enter the following details</h4><br>

      <div class="form-group row">
        <label class="col-sm-2 "><font size="+1">Description:</font></label>
        <div class="col-sm-10">
          {{ form.Description }}
        </div>
      </div>

      <div class="form-group row">
        <label class="col-sm-2 "><font size="+1">Image:</font></label>
        <div class="col-sm-10">
          {{ form.img }}
        </div>
      </div>

      <div class="form-group row">
        <label class="col-sm-2 "><font size="+1">Rating:</font></label>
        <div class="col-sm-10">
          {{ form.quantity }}
        </div>
      </div>
    </form>
  {% endblock %}
</body>
</html>
```

```

<div class="form-group row">
  <label class="col-sm-2 " ><font size="+1"> Price:</font></label>
  <div class="col-sm-10 " >
    {{form.price}}
  </div>
</div>

<div class="form-group row wrapper" >
  <div class="col-sm-10">
    <button type="submit" class="btn btn-primary">Submit</button>
  </div>
</div>
</form>
{% endblock%}
</body>
</html>

```

iii Display.html

```

<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Display</title>
</head>
<body>
  {% block content%}

  <form method="POST" class="post-form" action="add" >
    {% csrf_token %}
    <button type="submit" name="Sub" value="Add New Record" class="btn btn-
      primary" style="margin:30px">Add new record</button>
  </form>

  <div class="table-responsive">
    <table class="table table-hover table-bordered">
      <thead class="thead-dark">
        <tr>
          <th scope="col">id</th>
          <th scope="col">Description</th>

```

```

        <th scope="col">Rating</th>
        <th scope="col">Price</th>
        <th scope="col">image</th>
        <th scope="col">Action</th>
    </tr>
</thead>

    {% for item in items %}
<tbody>
    <tr>
        <th scope="row">{{ item.id }}</th>
        <td>{{ item.Description }}</td>
        <td>{{ item.quantity }}</td>
        <td>{{ item.price }}</td>
        <td>{{ item.img.url }}</td>
        <td>
            <a href="edit/{{ item.id }}"><span class="glyphicon glyphicon-edit" aria-
            hidden="true"></span>Edit</a>
            <a href="delete/{{ item.id }}"><span class="glyphicon glyphicon-
            trash"></span>Delete</a>
        </td>
    </tr>
</tbody>
    {% endfor %}
</table>

</div>
{% endblock %}
</body>
</html>

```

iv Update.html

```

<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Add</title>
    <style>
        .wrapper {
            text-align: center;
        }
    </style>

```

```

</head>
<body>
    {% block content%}
        <form method="POST" class="post-form" action="update/{{ cakeitem.id }}"
        enctype="multipart/form-data">
            {% csrf_token %}
            <br>
            <br>
            <h5>Update the following details</h5><br>

            <div class="form-group row">
                <label class="col-sm-2 "><font size="+1">Description:</font></label>
                <div class="col-sm-3">
                    <input type="text" class="form-control" value="{{ cakeitem.Description }}"
name="descrip">
                </div>
            </div>

            <div class="form-group row">
                <label class="col-sm-2 "><font size="+1">Rating:</font></label>
                <div class="col-sm-3">
                    <input type="number" class="form-control" value="{{ cakeitem.quantity }}"
name="quan">
                </div>
            </div>

            <div class="form-group row">
                <label class="col-sm-2 " ><font size="+1"> Price:</font></label>
                <div class="col-sm-3 " >
                    <input type="number" class="form-control" value="{{ cakeitem.price }}"
name="price">
                </div>
            </div>

            <div class="form-group row " >
                <div class="col-sm-3">
                    <button type="submit" class="btn btn-primary">Update</button>
                </div>
            </div>
        </form>
    {% endblock%}
</body>
</html>

```


v Buy.html

```

<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Add</title>
  <style>
    .wrapper {
      text-align: center;
    }

  </style>
</head>
<body>
  {% block content %}
    <form method="POST" class="post-form" action="update/{{ cakeitem.id }}"
    enctype="multipart/form-data">
      {% csrf_token %}
      <br>
      <br>
      <h5>Update the following details</h5><br>

      <div class="form-group row">
        <label class="col-sm-2 "><font size="+1">Description:</font></label>
        <div class="col-sm-3">
          <input type="text" class="form-control" value="{{ cakeitem.Description }}"
name="descrip">
        </div>
      </div>

      <div class="form-group row">
        <label class="col-sm-2 "><font size="+1">Rating:</font></label>
        <div class="col-sm-3">
          <input type="number" class="form-control" value="{{ cakeitem.quantity }}"
name="quan">
        </div>
      </div>

      <div class="form-group row">
        <label class="col-sm-2 "><font size="+1"> Price:</font></label>

```

```

        <div class="col-sm-3 " >
            <input type="number" class="form-control" value="{{cakeitem.price}}"
name="price">
        </div>
    </div>

    <div class="form-group row " >
        <div class="col-sm-3">
            <button type="submit" class="btn btn-primary">Update</button>
        </div>
    </div>
</form>
{% endblock%}
</body>
</html>

```

vi Home.html

```

<!DOCTYPE html>
{% extends 'Base.html' %}
{% load static %}

<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Items</title>
    <style>
        .card-img-top {
            flex-grow: 1;
            object-fit:contain;
        }
    </style>
</head>
<body>

    {% block content%}
        <br>
        <p style=" color:#000000">*All prices are as per 1kg</p>
        <div class="row">
            {% for item in items %}
                <div class="col-lg-3">

                    <div class="card-deck mt-4">
                        <div class="card bg-dark text-white">
                            

```

```

        <div class="card-body">
            <h4 class="card-title">{{ item.Description }}</h4>
            <div class="card-text">Pice:Rs. {{item.price}}</div>
            <p class="card-text"> <small >Rating: {{ item.quantity }}/5</small></p>
            <a href="details/{{item.id}}" class="btn btn-primary card-link stretched-
link" methods="GET">Buy</a>
        </div>
    </div>

    </div>

    </div>
    {%endfor%}
</div>

{% endblock %}

</body>
</html>

```

vii Payment.html

```

<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Payment</title>
</head>
<body>
    {% block content%}

        <div class="card-body">
            <h5>{{ cus.name }}</h5>
            {{ cus.address }}
            <br>{{ cus.email }}
            <br><small class="text-muted">{{ cus.phoneno }}</small>
        </div>

        <div class="table-responsive">
            <table class="table">
                <thead class="thead-dark">
                    <tr>
                        <th scope="col">Description</th>

```

```
<th scope="col">Quantity(kg)</th>
<th scope="col">Price(per kg)</th>
</tr>
</thead>

<tbody>
<tr>
<td>{{item.Description}}</td>
<td>{{cus.quan}}</td>
<td>{{item.price}}</td>
</tr>
</tbody>

<tbody>
<tr>
<td></td>
<td></td>
<td>{{cus.totalamt}}</td>
</tr>
</tbody>

</table>
</div>
<div style="text-align:center">
<a href="/thankyou" class="btn btn-primary" methods="GET">Payment </a>
</div>

{% endblock%}
</body>
</html>
```

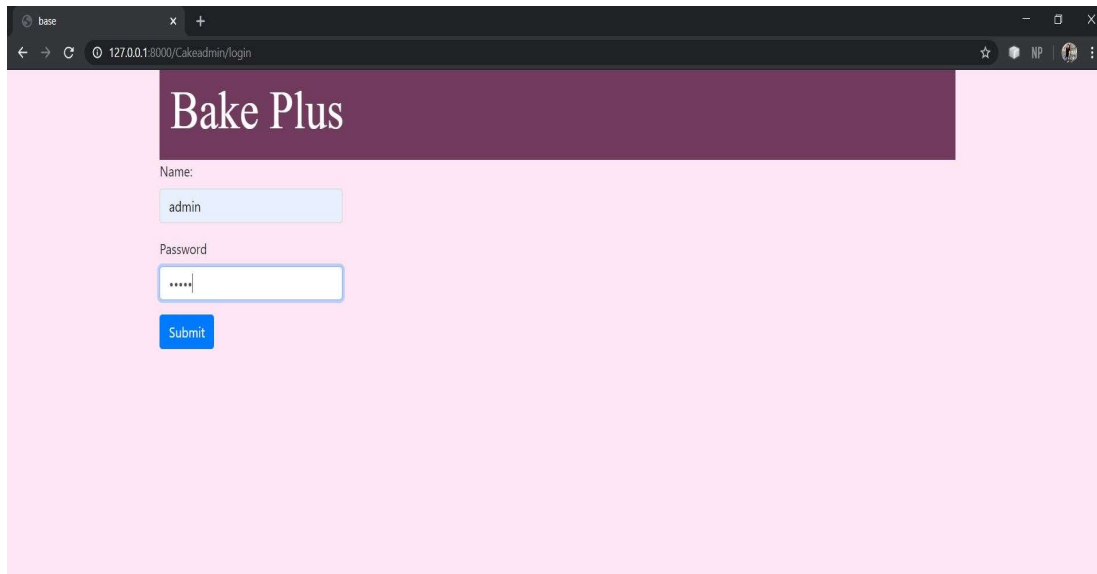
viii Thankyou.html

```
<!DOCTYPE html>
{% extends 'Base.html' %}
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Thankyou</title>
</head>
<body>
  {% block content%}
    <center>
      <br>
      <br>
      <h1>Thank you!!</h1>
      <h5>Your ordered will be placed soon </h5>
    </center>

    {% endblock%}
  </body>
</html>
```

3. Screenshots of Design

i Admin interfaces



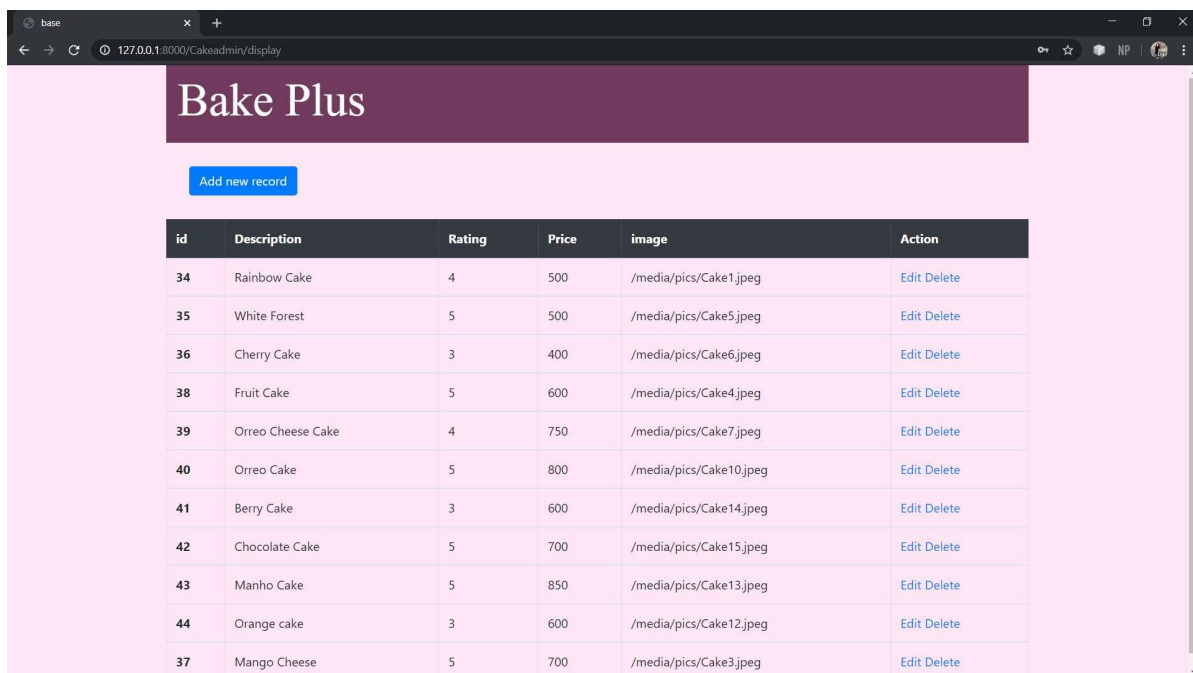
Bake Plus

Name:

Password:

[Submit](#)

a) AdminLogin

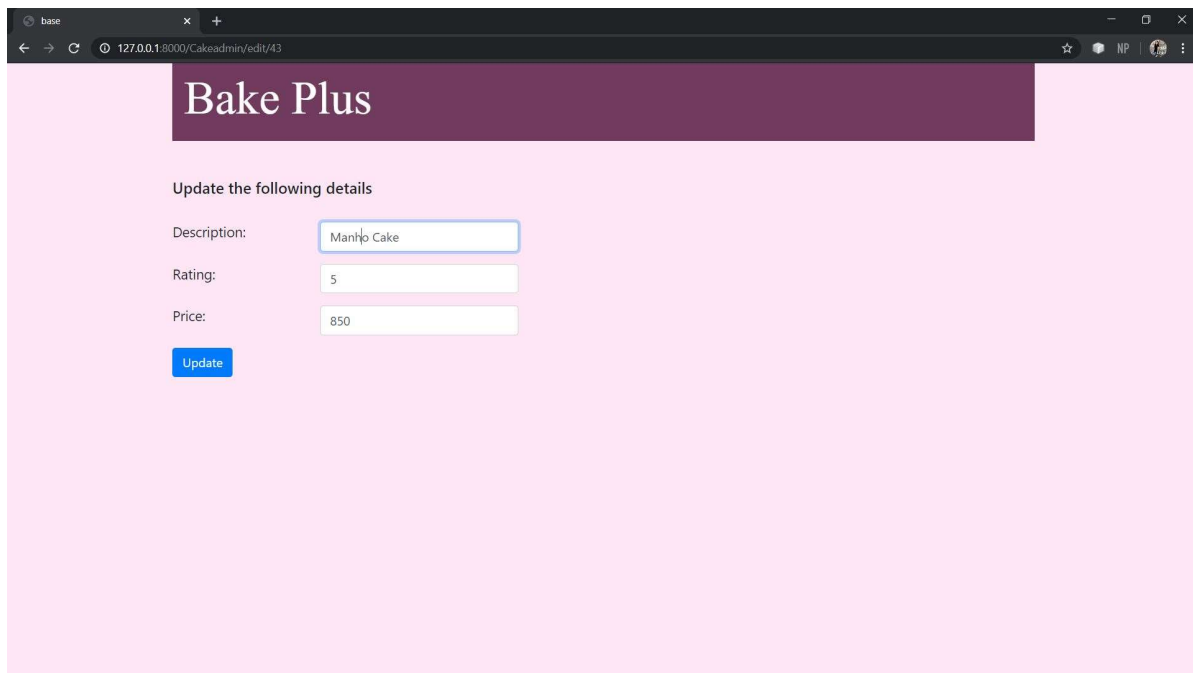


Bake Plus

[Add new record](#)

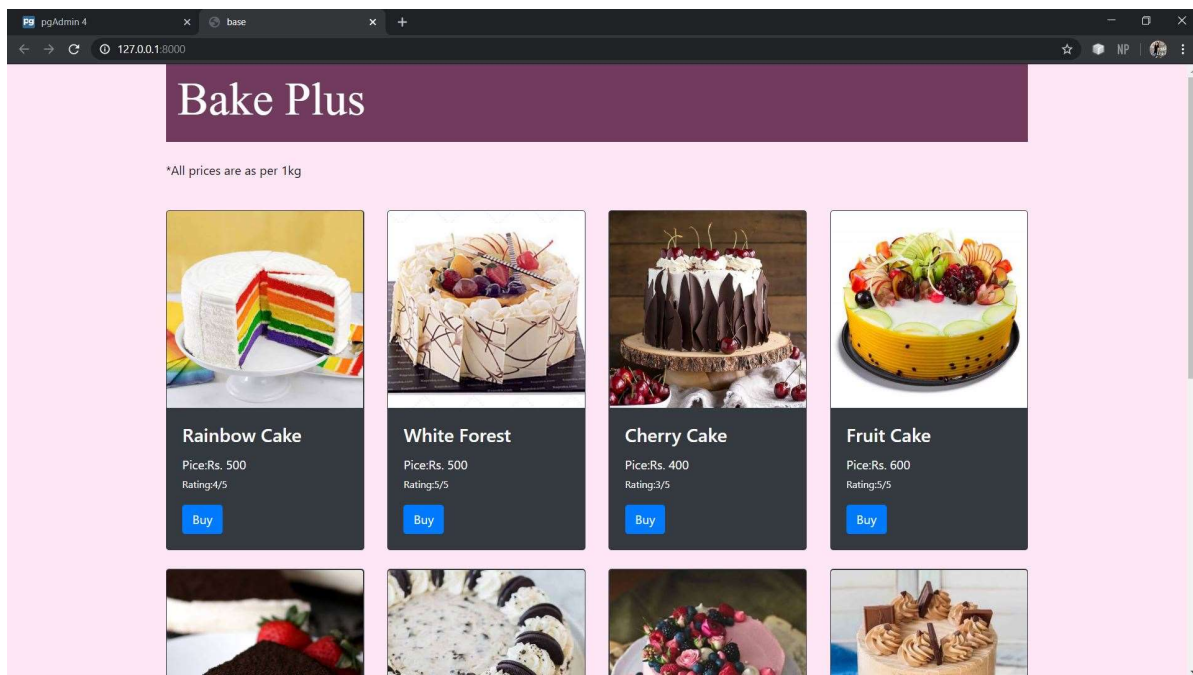
id	Description	Rating	Price	image	Action
34	Rainbow Cake	4	500	/media/pics/Cake1.jpeg	Edit Delete
35	White Forest	5	500	/media/pics/Cake5.jpeg	Edit Delete
36	Cherry Cake	3	400	/media/pics/Cake6.jpeg	Edit Delete
38	Fruit Cake	5	600	/media/pics/Cake4.jpeg	Edit Delete
39	Orreo Cheese Cake	4	750	/media/pics/Cake7.jpeg	Edit Delete
40	Orreo Cake	5	800	/media/pics/Cake10.jpeg	Edit Delete
41	Berry Cake	3	600	/media/pics/Cake14.jpeg	Edit Delete
42	Chocolate Cake	5	700	/media/pics/Cake15.jpeg	Edit Delete
43	Manho Cake	5	850	/media/pics/Cake13.jpeg	Edit Delete
44	Orange cake	3	600	/media/pics/Cake12.jpeg	Edit Delete
37	Mango Cheese	5	700	/media/pics/Cake3.jpeg	Edit Delete

b) AdminDisplay of Cakes



c) AdminUpdate of Cakes

ii User Interfaces



d) User home page

Bake Plus

Enter the following details

Name:

Phone Number:

Email id:

Address:

Quantity:

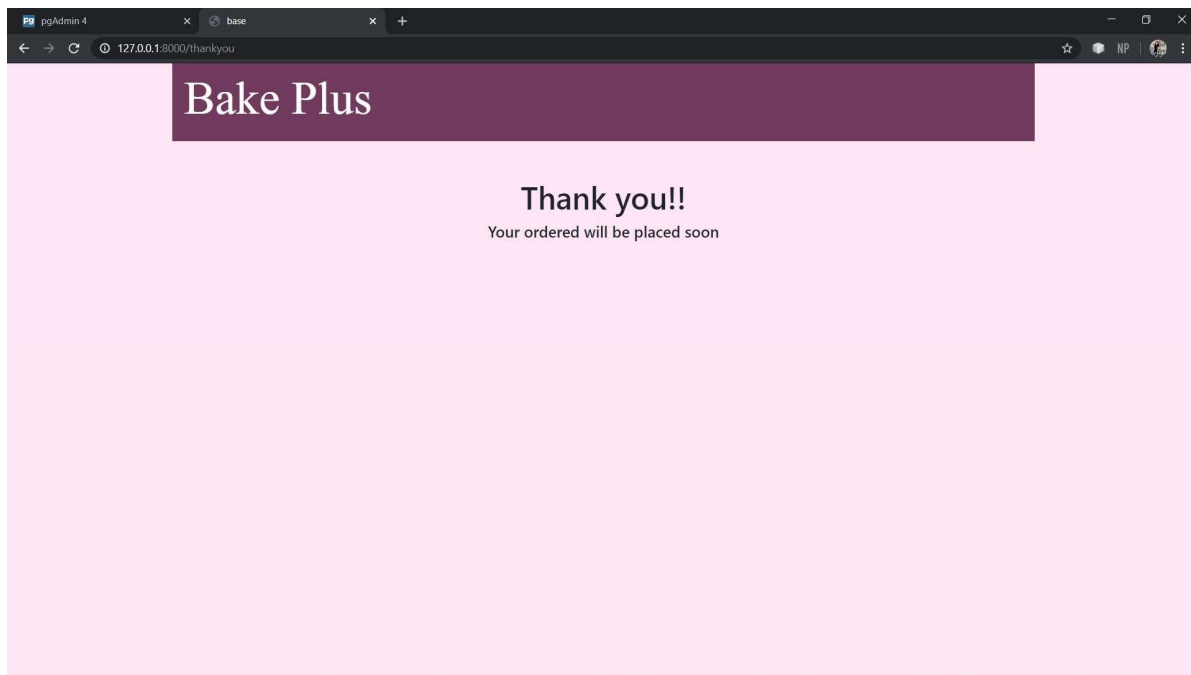
e) User details

Bake Plus

Alonie Jane Crasta
Building 45, Street 3, shantinagar,Bangalore
alrishjane@gmail.com
7878345678

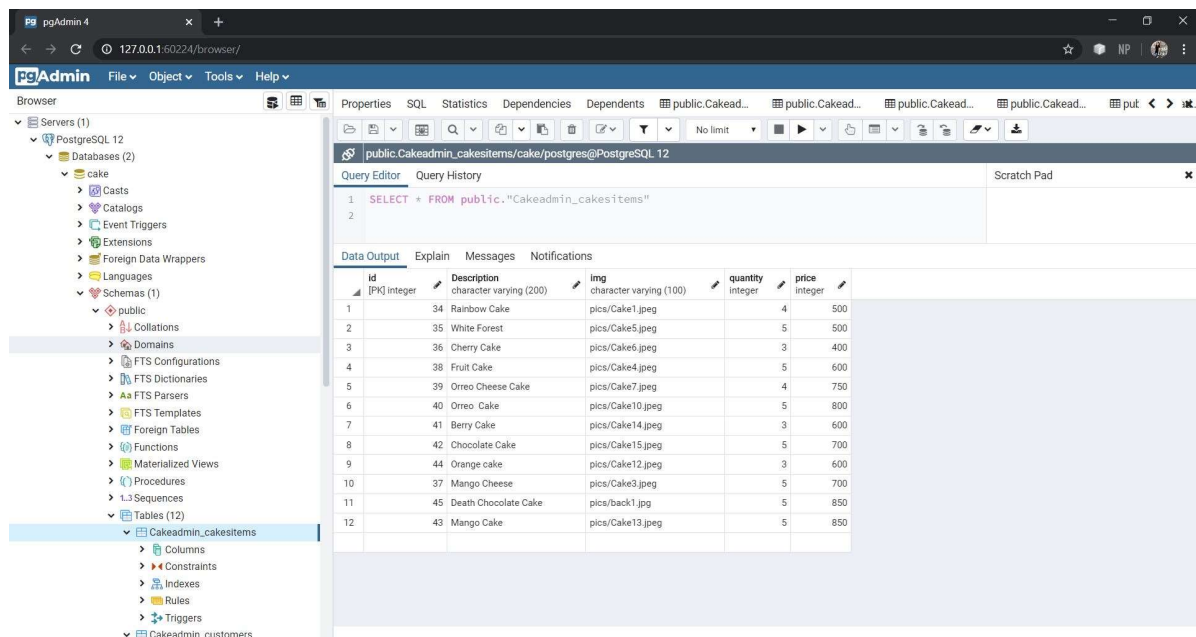
Description	Quantity(kg)	Price(per kg)
Orreo Cake	2.00	800
		1600.00

f) User Payment

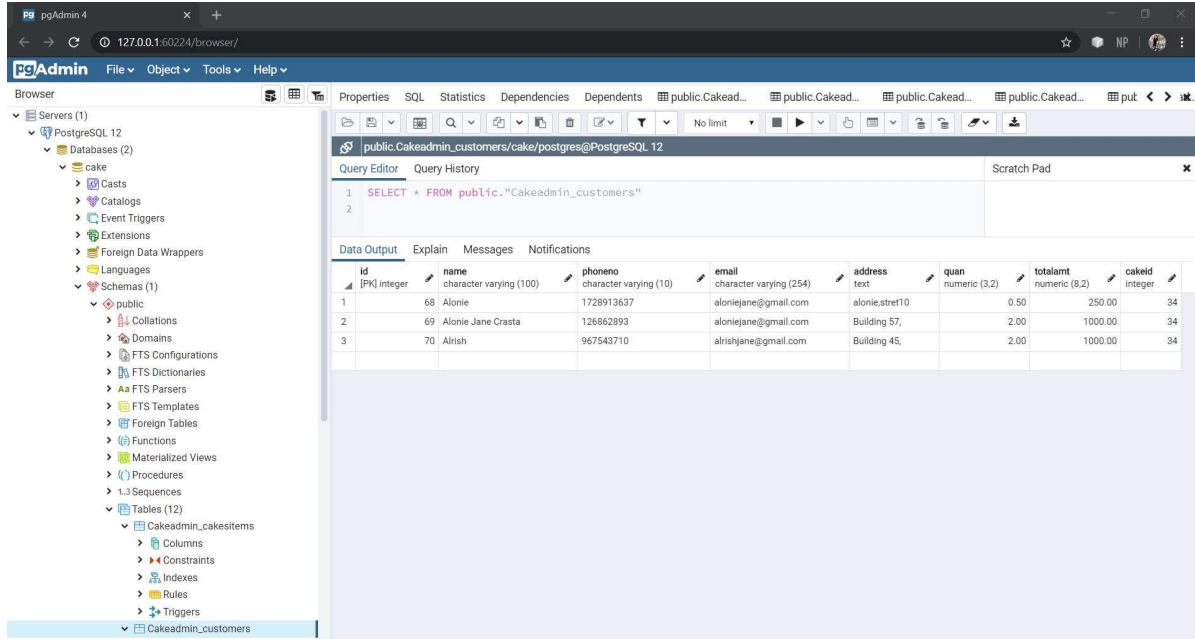


g) User thankyou

iii PostgreSQL Databse



h) Table cakeitems



The screenshot shows the pgAdmin 4 web interface. The left sidebar displays the database structure, including the 'public' schema and the 'Cakeadmin_customers' table. The main pane shows the 'Query Editor' with a SQL query: `SELECT * FROM public."Cakeadmin_customers"`. Below the query editor, the 'Data Output' tab displays the results of the query in a table format.

id	name	phoneno	email	address	quan	totalamt	cakeid
1	Alonie	1728913637	aloniejane@gmail.com	alonie,stret10	0.50	250.00	34
2	Alonie Jane Crasta	126862893	aloniejane@gmail.com	Building 57,	2.00	1000.00	34
3	Alrish	967543710	alrishjane@gmail.com	Building 45,	2.00	1000.00	34

i) Table Customers