Project Report

(Project Semester Jan-June 2020)

An

“(E-Commerce website)”

Submitted by

AKASH RAI



Under the Guidance of

Industry Guide -Mr. Bhanu Pratap

Designation: Full stack developer

**Declaration**

I hereby declare that the project work entitled “E-Commerce website” is an authentic record of my own work carried out at CDAC ATC as requirements of six months project semester for the award of degree of Master of Computer Application University of Rajasthan, under the guidance of Mr.Bhanu Pratap Singh Chouhan.

AKASH RAI

Certified that the above statement made by the student is correct to the best of our knowledge.

Mr.Bhanu Pratap

Industrial Guide

**Acknowledgment**

Many people have supported me in different ways during the internship project. I would like to thank our Industry Guide Mr. Bhanu Pratap Singh Chouhan valuable guidance during the work process. My family for offering me their unconditional support throughout the internship project. However, it would not have been possible without the kind support of many individuals and organizations. I would like to extend my sincere thanks to each and every member of CDAC ATC.

**Table of Contents**

Topic Page No.

Company Profile 6

Abstract 7

1) Aim and Objective 8

2) Introduction 9

3) Feasible Study 11

4) Main Components of the website 12

5) Resources Required 13

5.1) Software Requirements 15

5.2) Hardware Requirements 17

6) Python 24

6.1) Python Features 25

7) Django 28

7.1) Uses of Django 30

8) Component Architecture 33

9) Role/Contribution 50

10) Learnings 51

11) Future Enhancement 52

12) Conclusion 53

**List of illustrations**

The “List of Illustrations” gives systematic information about Tables, Charts, Figures and Graphs used in the report. It is usually included if the number of Illustrations are more than 10. Figure below contains a Sample List of Illustrations

|  |  |  |
| --- | --- | --- |
| **Figure** |  |  |
|  |  |  |
| Figure 1 | Flowchart | 26 |
|  |  |  |
| Figure 2 | Template | 28 |
|  |  |  |
| Figure 3 | Views.py | 30 |
|  |  |  |

**Company Profile**



**About C-DAC ATC NETCOM Jaipur**

Centre for Development of Advanced Computing (C-DAC) is the premier R&D organization of the Department of Electronics and Information Technology (DeitY), Ministry of Communications & Information Technology (MCIT) for carrying out R&D in IT, Electronics and associated areas. Different areas of C-DAC, had originated at different times, many of which came out as a result of identification of opportunities.While C-DAC was being setup for the indigenous design, development and delivery of the supercomputing technologies for the country, the mandate given was to not only develop the supercomputing technologies in the shortest possible time, but also continue to develop the high quality human resource, which will continue to develop such advanced technologies. C-DAC's Advanced Computing Training School (ACTS) is dedicated to creating high quality manpower for C-DAC in particular and the IT industry in general through the designing and delivering various courses. The projects are offered through a network of Authorized Training Centres (ATC's) as well as C-DAC's own centers.

NETCOM is having its own Software Development Company “NetParam Technologies Pvt. Ltd.” which is completely involved in IT products and service delivery. NTPL develops world class Websites, Android Apps, Iphone Apps, Cloud Based Application, Hybrid Mobile Apps & Big Data and Analysis. It works for the government and private projects and here the resources are placed from the top of the world. It is one of the most comprehensive and global leader in IT services that helps its clients to simplify, strengthen and transform their businesses. NETCOM is equipped with the latest infrastructure which provides a congenial and a healthy learning

environment to the students. In Netcom, Experts and Developers mentors the students and develops the professional quality in them.



**About Netparam Technologies Pvt. Ltd.**

Innovation and proper aim are key factors in the growth of technology for any ICT solutions provider companies. Here at Netparam Technologies Private Limited encourage innovation throughout the organization and emphasizes strategic planning in starting and developing operations.

We start with the data. We conduct independent research and draw on the latest technology to develop new insights and recommendations. Our rigorous analysis identifies risks, unveils opportunities, and informs smart strategies. We focus our efforts on influential and emerging economies where the future of sustainability will be determined.

We use our research to influence government policies, business strategies, and civil society action. We test projects with communities, companies, and government agencies to build a strong evidence base. Then, we work with partners to deliver change on the ground that alleviates poverty and strengthens society. We hold ourselves accountable to ensure our outcomes will be bold and enduring.

We don’t think small. Once tested, we work with partners to adopt and expand our efforts regionally and globally. We engage with decision-makers to carry out our ideas and elevate our impact. We also measure success through human well-being that improves people’s lives and sustains a healthy environment.

**Vision :**

We envision to be a recognized platform to create a world where the actions of the government, business, and communities combine to eliminate chaotic ICT solutions.

**Mission :**

* To encourage innovation throughout the organization and emphasizes strategic planning in starting and developing operations.
* To identify risks unveils opportunities and informs smart strategies.
* To influence government policies, business strategies, and civil society action.
* To work with partners to deliver change on the ground that alleviates poverty and strengthens society.
* To promote national and international collaboration with partners.

**Abstract**

The online food ordering system is one of the latest servicers most fast food restaurants in the western world are adopting. With this method, food is ordered online and delivered to the customer. The development of online food ordering system involved many phases. The approach used is a top-down one concentrating on *what* first, then *how* and moving to successive levels of details. The first phase startedwith a detailed study of the problems and prospects of ordering in Breakfast.com. In the course of this study, many problems were discovered to have hindered the effectiveness of the existing manual system. These problems, information needs and activities were documented and later used as the basis for system design, which immediately followed the first phase. The design phase was concerned primarily with the specification of the system elements in manner that best met the organization’s business needs. During this phase, strict adherence was made on proven software engineering principles and practices. To implement this design, a computer program was then written and tested in Django environment. It is hoped that effective implementation of this software product would eliminate many problems discovered during systems investigation.

**1. Aim and Objective :**

The development of online food ordering system involved many phases. The approach used is a top-down one concentrating on *what* first, then *how* and moving to successive levels of details. The first phase started with a detailed study of the problems and prospects of ordering in Breakfast.com. In the course of this study, many problems were discovered to have hindered

the effectiveness of the existing manual system.A project on django web framework website to to improve the skill of the user and make them learn about the different technology and enhance the knowledge by of the user by studying from it.

E-Commerce website include the different subjects ,login ,feedback and etc. The subject have easy to understand language. So the user can understand it easily and that will help them to get the goals of life.

**2. Introduction :**

It is a project on python Django framework to order food online to providē ease for users.Due to the great increase in the awareness of internet and the technologies associated with it, several opportunities are coming up on the web. So many businesses and companies now venture into their business with ease because of the internet. One of such business that the internet introduced is an online food ordering system. In today’s age of fast food and take out, many restaurants have chosen to focus on quick preparation and speedy delivery of orders rather than offering a rich dining experience. Until recently, most of this delivery orders were placed over the phone, but there are many disadvantages to this system.

It is possible for anybody to order any goods via the internet and have the goods delivered at his/her doorsteps. But while trying to discuss the transfer method of the goods and services, attention is focused on the payment mode. In other words, how possible is it to pay for goods and services via the internet? This then leads to the discussion of the economic consequences of digital cash. What are the implementations from the view point of economic? Since the world is fast becoming a global village, the necessary tool for this process is communication of which telecommunication is a key player. A major breakthrough is the wireless telephone system which comes in either fixed wireless telephone lines or the Global System of Mobile communication (GSM).

**STATEMENT OF PROBLEM**

As industries are fast expanding, people are seeking for more ways to purchase products with much ease and still maintain cost effectiveness. The vendors need to purchase the products in order to sell to end users. The manual method of going to their local food sales outlets to purchase food is becoming obsolete and more tasking. Food can be ordered through the internet and payment made without going to the restaurant or the food vendor. So there is need for a wide range of publicity and enabling direct order, processing and delivering of food through online system. For this system, there will be a system administrator who will have the rights to enter the menu with current prevailing prices

**3. Feasible Study :**

The study of project include increased customer satisfaction, improvement in product quality better decision making timeliness of information, improved accuracy of operations, better understand of knowledge , faster retrieval of information etc.

Proposed project is beneficial only if it can be turned into information systems that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to implementation? Is there sufficient support for the project from management from users? Are the current business methods acceptable to the user?

The feasibility study of project includes variety of issues that must be considered during while doing a project. Firstly, understand the various technologies involved for creating the website effective and must be very clear about what technologies are to be required for the development of the website. Determine whether the organization currently using the desired technologies or not? Is that the required technology available with the organization?

In Tutorial website, main technologies which are to be used are Python 3 , Django web framework of python and MySQL for maintaining the database.

**4. Main components of the website:**

* 1. **Subjects :** Generally there are so many subject to read and some time the user don’t havethe proper material to study this website provide the study material to the user so he/she can easily learn from there.
  2. **User :** User can login in to the website and excess the website by register there so whenever new subject is uploaded the admin can drop a mail to the user
  3. **Admin :** Admin is largely managing the functionality of management of website thatwhich subject will uploaded and can remove the dead user from login.
  4. **MySQL** : MySQL Database is used to store each activity of the website and integratingdatabase connectivity with the website using django is helpful in making website more efficient.
  5. **Login**: User can login after registration to access the website completely.
  6. **Registration**: User can register to access the website completely without registration theuser is not able to access the website

1. **Resources Used :**

|  |  |  |
| --- | --- | --- |
| **5.1 Software Used :** |  |  |
| Operating System | : | Windows 10.1 |
| User Interface | : | HTML , CSS 3,Bootstap |
| Text Editor | : | Atom |
| Programming Language | : | Python 3 |
| Web Framework | : | Django |
| Database | : | MySQL |

|  |  |  |
| --- | --- | --- |
| **5.2 Hardware Used :** |  |  |
| Processor | : | Intel core i5 7th Generation |
| Hard Disk | : | 1TB |
| RAM | : | 8GB |

**6. Python :**

Python is an object-oriented, high-level programming language with integrated dynamic semantics primarily for web and app development. Python laid its foundation in the late 1980s. The implementation of Python was started in the December 1989 by Guido Van Rossum at CWI in Netherland. In February 1991, van Rossum published the code (labeled version 0.9.0) to alt.sources. In 1994, Python 1.0 was released with new features like: lambda, map, filter, and reduce. Python 2.0 added new features like: list comprehensions, garbage collection system. On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was absolutely designed to rectify fundamental flaw of the language.

**6.1 Python Features :**

1. Easy to Learn and Use
2. Expressive Language
3. Interpreted Language
4. Cross-platform Language
5. Free and Open Source
6. Object-Oriented Language

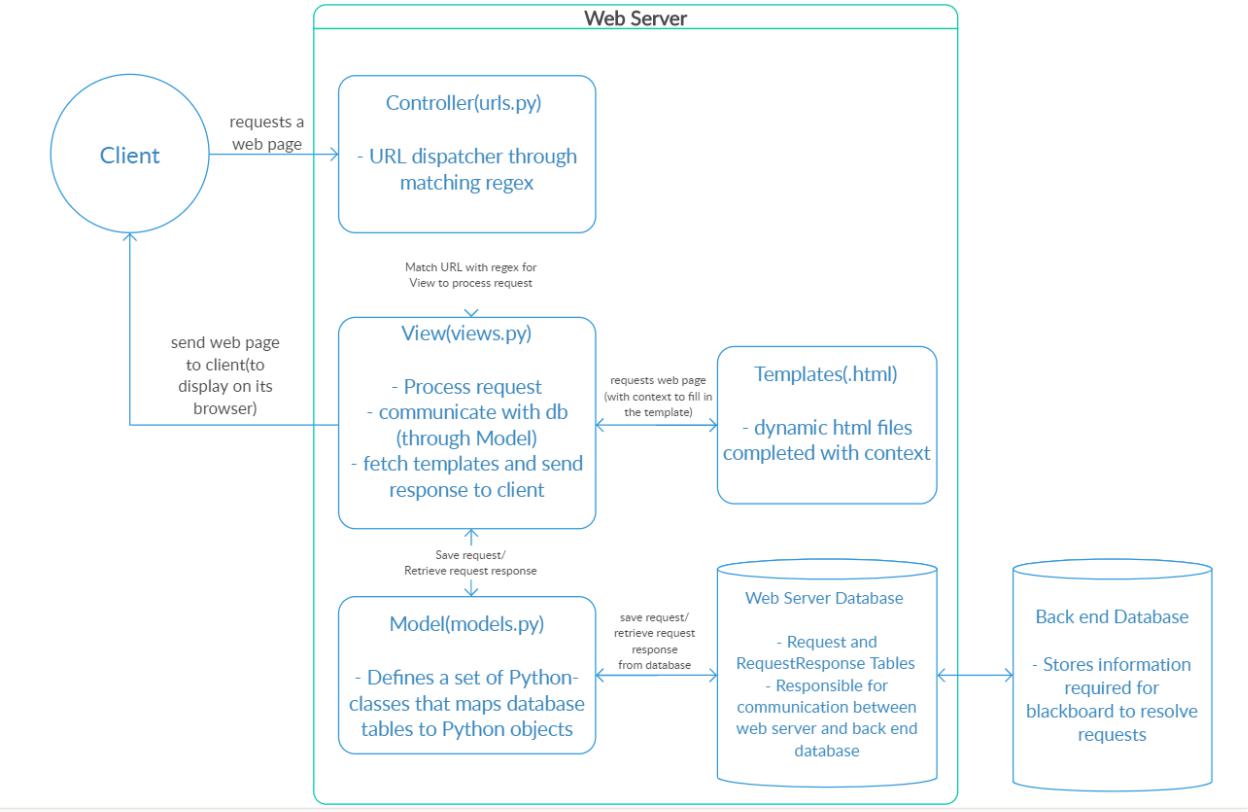
1. Extensible
2. Large Standard Library
3. GUI Programming Support
4. Integrated

**7. Django :**

Django is a free and open source web application framework written in Python. A framework is nothing over a collection of modules that make development easier. They are grouped together, and permit you to form applications or websites from an existing source, instead of from scratch.

This is how websites - even simple ones designed by an single person - can still include advanced functionality like authentication support, management and admin panels, contact forms, comment boxes, file upload support, and more. In other words, if you were creating a website from scratch you had have to develop these components yourself. By employing a framework instead, these components are already built, you only have to configure them properly to match your site. The official project site describes Django as "a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the effort of Web development, so you may consider writing your app with no need to reinvent the wheel. It’s free and open source."

Django offers an enormous collection of modules which you may use in your own projects. Primarily, frameworks exist to save lots of developers a lot of wasted time and headaches and Django isn’t any different.



**Figure 1 : Web Server**

**7.1 Why Django?**

Django was invented to meet fast-moving newsroom deadlines, while satisfying the tough requirements of experienced Web developers.

With Django, you can take Web applications from concept to launch in a matter of hours. Django takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

**1)Ridiculously fast.**

Django was designed to help developers take applications from concept to completion as quickly as possible.

**2)Fully loaded.**

Django includes dozens of extras you can use to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds, and many more tasks — right out of the box.

**3)Reassuringly secure.**

Django takes security seriously and helps developers avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery and clickjacking. Its user authentication system provides a secure way to manage user accounts and passwords.

**4)Exceedingly scalable.**

Some of the busiest sites on the planet use Django’s ability to quickly and flexibly scale to meet the heaviest traffic demands.

**5)Incredibly versatile.**

Companies, organizations and governments have used Django to build all sorts of things — from content management systems to social networks to scientific computing platforms.

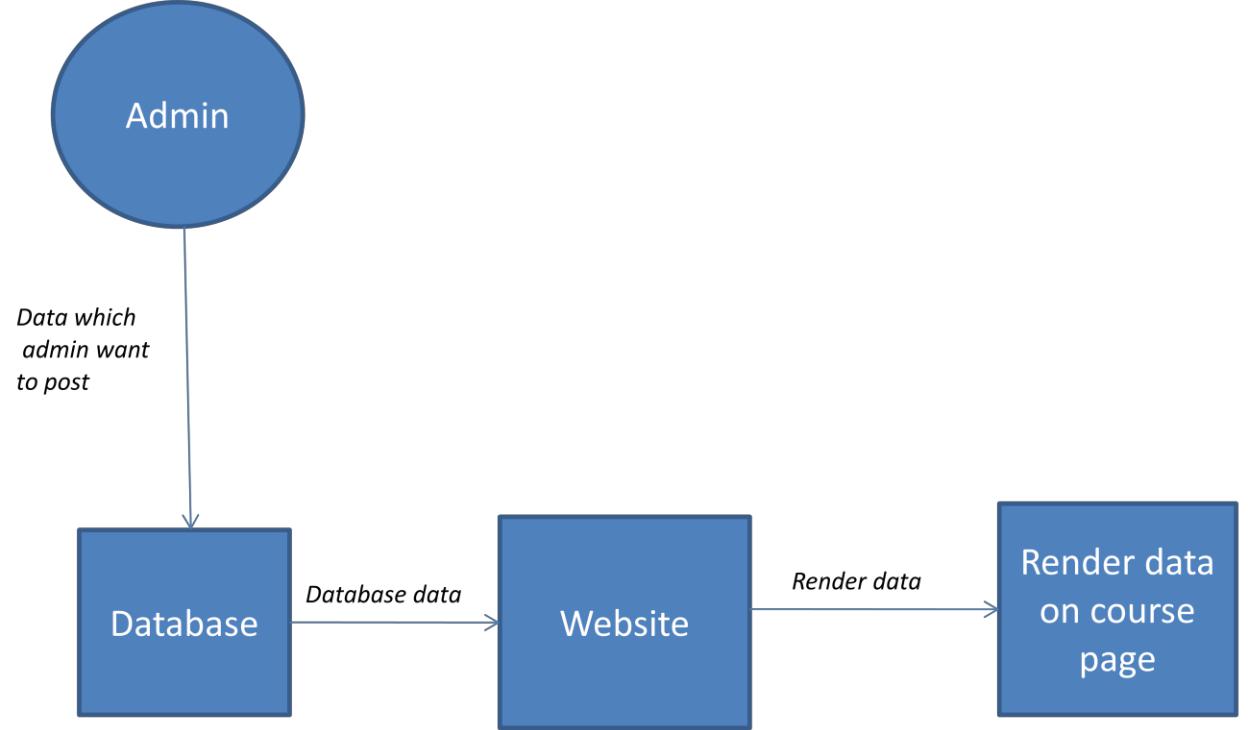
**7.2 Uses of Django :**

1. Django is time-tested
2. Application Development
3. Easy to Use
4. Operating System Dependent
5. Excellent Documentation for real-world application
6. Scalable and Reliable
7. Community Support
8. Django Features:
   1. Template layers.
   2. Forms, development process.
   3. Views layers, security.
   4. Model layers, python compatibility.
   5. Localization, performance, and optimization.
   6. Geographic framework, common tools for web application development.

g. Other core functionalities required for websites.

As Django can be used to build any variety of website with help of its frameworks like content management, Wikipedia pages, social networking applications, chat applications, and websites like Mozilla, Instagram, Pinterest, BitBucket etc. Django can work with any client-server applications and able to deliver content in any form (HTML, text, JSON, XML,etc.). So Django is used for completing your web application or website within a short duration with full security and offers fast, reliable and scalable services based on traffic. Many good websites are developed using Django framework like **Instagram, BitBucket, and Mozilla** etc.

1. **Files that use:-**



**Figure 2 : Flowchart**

**Writing views**

A view function, or view for short, is a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image . . . or anything, really. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as it’s on your Python path. There’s no other requirement–no “magic,” so to speak.

For the sake of putting the code somewhere, the convention is to put views in a file called views.py, placed in your project or application directory.

A simple view

Here’s a view that returns the current date and time, as an HTML document:

from django.http import HttpResponse

import datetime

def current\_datetime(request):

now = datetime.datetime.now()

html = "<html><body>It is now %s.</body></html>" % now

return HttpResponse(html)

Let’s step through this code one line at a time:

First, we import the class HttpResponse from the django.http module, along with Python’s datetime library.

Next, we define a function called current\_datetime. This is the view function. Each view function takes an HttpRequest object as its first parameter, which is typically named request.

Note that the name of the view function doesn’t matter; it doesn’t have to be named in a certain way in order for Django to recognize it. We’re calling it current\_datetime here, because that name clearly indicates what it does.

**Templates**

Being a web framework, Django needs a convenient way to generate HTML dynamically. The most common approach relies on templates. A template contains the static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted.

A Django project can be configured with one or several template engines (or even zero if you don’t use templates). Django ships built-in backends for its own template system, creatively called the Django template language (DTL), and for the popular alternative Jinja2. Backends for other template languages may be available from third-parties.

Django defines a standard API for loading and rendering templates regardless of the backend. Loading consists of finding the template for a given identifier and preprocessing it, usually compiling it to an in-memory representation. Rendering means interpolating the template with context data and returning the resulting string.

The Django template language is Django’s own template system. Until Django 1.8 it was the only built-in option available. It’s a good template library even though it’s fairly opinionated and sports a few idiosyncrasies. If you don’t have a pressing reason to choose another backend, you should use the DTL, especially if you’re writing a pluggable application and you intend to distribute templates. Django’s contrib apps that include templates, like django.contrib.admin, use the DTL.

For historical reasons, both the generic support for template engines and the implementation of the Django template language live in the django.template namespace.

Warning

The template system isn’t safe against untrusted template authors. For example, a site shouldn’t allow its users to provide their own templates, since template authors can do things like perform XSS attacks and access properties of template variables that may contain sensitive information.

Support for template engines¶

Configuration¶

Templates engines are configured with the TEMPLATES setting. It’s a list of configurations, one for each engine. The default value is empty. The settings.py generated by the startproject command defines a more useful value:

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [],

'APP\_DIRS': True,

'OPTIONS': {

# ... some options here ...

},

},

]

BACKEND is a dotted Python path to a template engine class implementing Django’s template backend API. The built-in backends are django.template.backends.django.DjangoTemplates and django.template.backends.jinja2.Jinja2.

Since most engines load templates from files, the top-level configuration for each engine contains two common settings:

DIRS defines a list of directories where the engine should look for template source files, in search order.

APP\_DIRS tells whether the engine should look for templates inside installed applications. Each backend defines a conventional name for the subdirectory inside applications where its templates should be stored.

While uncommon, it’s possible to configure several instances of the same backend with different options. In that case you should define a unique NAME for each engine.

Here’s an example of the search algorithm. For this example the TEMPLATES setting is:

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [

'/home/html/example.com',

'/home/html/default',

],

},

{

'BACKEND': 'django.template.backends.jinja2.Jinja2',

'DIRS': [

'/home/html/jinja2',

],

},

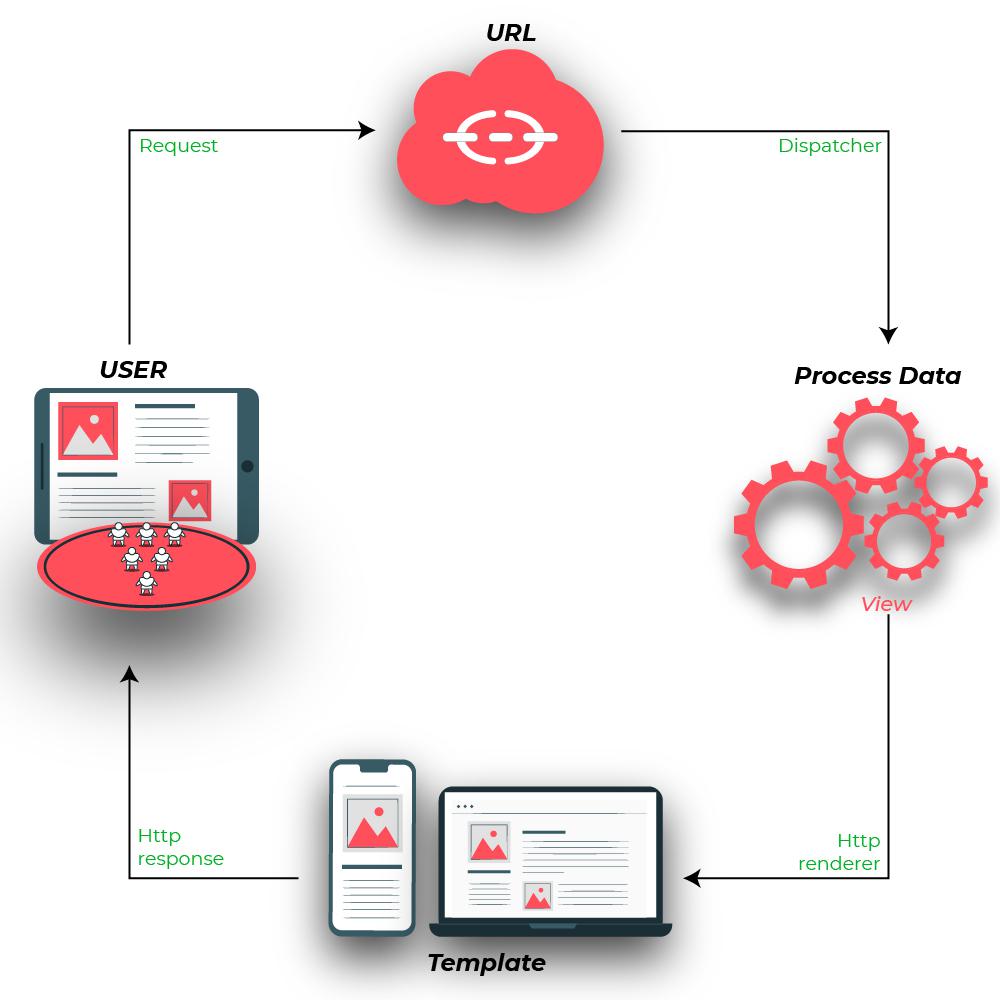
]

If you call get\_template('story\_detail.html'), here are the files Django will look for, in order:

/home/html/example.com/story\_detail.html ('django' engine)

/home/html/default/story\_detail.html ('django' engine)

/home/html/jinja2/story\_detail.html ('jinja2' engine)

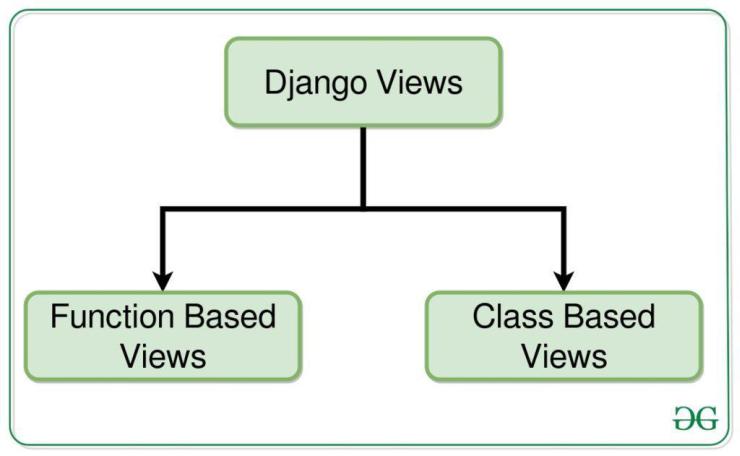


**Figure 3 : Template**

**The view**

Form data sent back to a Django website is processed by a view, generally the same view which published the form. This allows us to reuse some of the same logic.

To handle the form we need to instantiate it in the view for the URL where we want it to be published:



**Figure 4 : views.py**

from django.http import HttpResponseRedirect

from django.shortcuts import render

from .forms import NameForm

def get\_name(request):

* if this is a POST request we need to process the form data if request.method == 'POST':

* create a form instance and populate it with data from the request: form = NameForm(request.POST)
* check whether it's valid:

if form.is\_valid():

* process the data in form.cleaned\_data as required
* ...
* redirect to a new URL:

return HttpResponseRedirect('/thanks/')

* if a GET (or any other method) we'll create a blank form else:

form = NameForm()

return render(request, 'name.html', {'form': form})

If we arrive at this view with a GET request, it will create an empty form instance and place it in the template context to be rendered. This is what we can expect to happen the first time we visit the URL.

If the form is submitted using a POST request, the view will once again create a form instance and populate it with data from the request: form = NameForm(request.POST) This is called “binding data to the form” (it is now a bound form).

We call the form’s is\_valid() method; if it’s not True, we go back to the template with the form. This time the form is no longer empty (unbound) so the HTML form will be populated with the data previously submitted, where it can be edited and corrected as required.

If is\_valid() is True, we’ll now be able to find all the validated form data in its cleaned\_data attribute. We can use this data to update the database or do other processing before sending an HTTP redirect to the browser telling it where to go next.

The template

We don’t need to do much in our name.html template:

<form action="/your-name/" method="post">

{% csrf\_token %}

{{ form }}

<input type="submit" value="Submit">

</form>

All the form’s fields and their attributes will be unpacked into HTML markup from that {{ form }} by Django’s template language.

**Admin**

One of the most powerful parts of Django is its automatic admin interface. It reads metadata in your models to provide a powerful and production-ready interface that content producers can

immediately use to start managing content on your site. It’s easy to set up and provides many hooks for customization.

from django.contrib import admin

from bands.models import Band, Member

class MemberAdmin(admin.ModelAdmin):

"""Customize the look of the auto-generated admin for the Member model"""

list\_display = ('name', 'instrument')

list\_filter = ('band',)

admin.site.register(Band) # Use the default options

admin.site.register(Member, MemberAdmin) # Use the customized options

**django-admin and manage.py**

django-admin is Django’s command-line utility for administrative tasks. This document outlines

all it can do.

In addition, manage.py is automatically created in each Django project. It does the same thing as django-admin but also sets the DJANGO\_SETTINGS\_MODULE environment variable so that it points to your project’s settings.py file.

The django-admin script should be on your system path if you installed Django via pip. If it’s not on your path, you can find it in site-packages/django/bin within your Python installation. Consider symlinking it from some place on your path, such as /usr/local/bin.

For Windows users, who do not have symlinking functionality available, you can copy django-admin.exe to a location on your existing path or edit the PATH settings (under Settings - Control Panel - System - Advanced - Environment...) to point to its installed location.

Generally, when working on a single Django project, it’s easier to use manage.py than django-admin. If you need to switch between multiple Django settings files, use django-admin with DJANGO\_SETTINGS\_MODULE or the --settings command line option.

The command-line examples throughout this document use django-admin to be consistent, but any example can use manage.py or python -m django just as well.

Usage¶

$ django-admin <command> [options]

$ manage.py <command> [options]

$ python -m django <command> [options]

command should be one of the commands listed in this document. options, which is optional, should be zero or more of the options available for the given command.

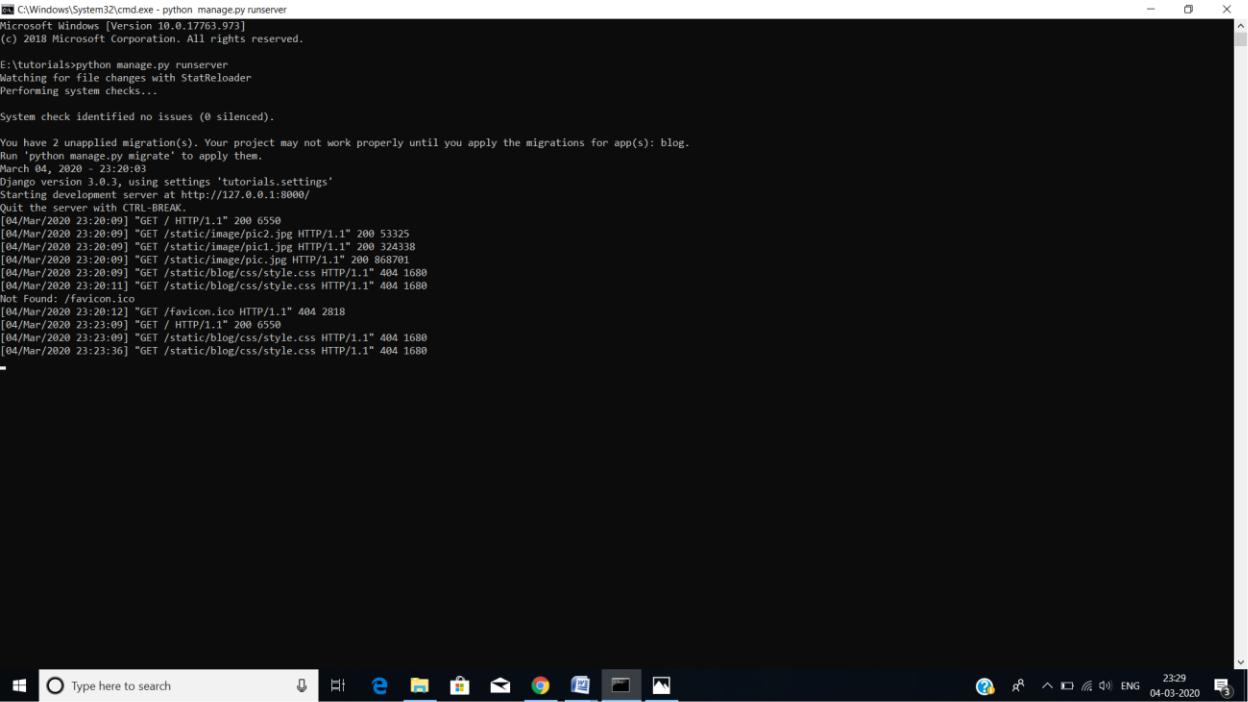
For example:

django-admin startproject myproject /Users/jezdez/Code/myproject\_report

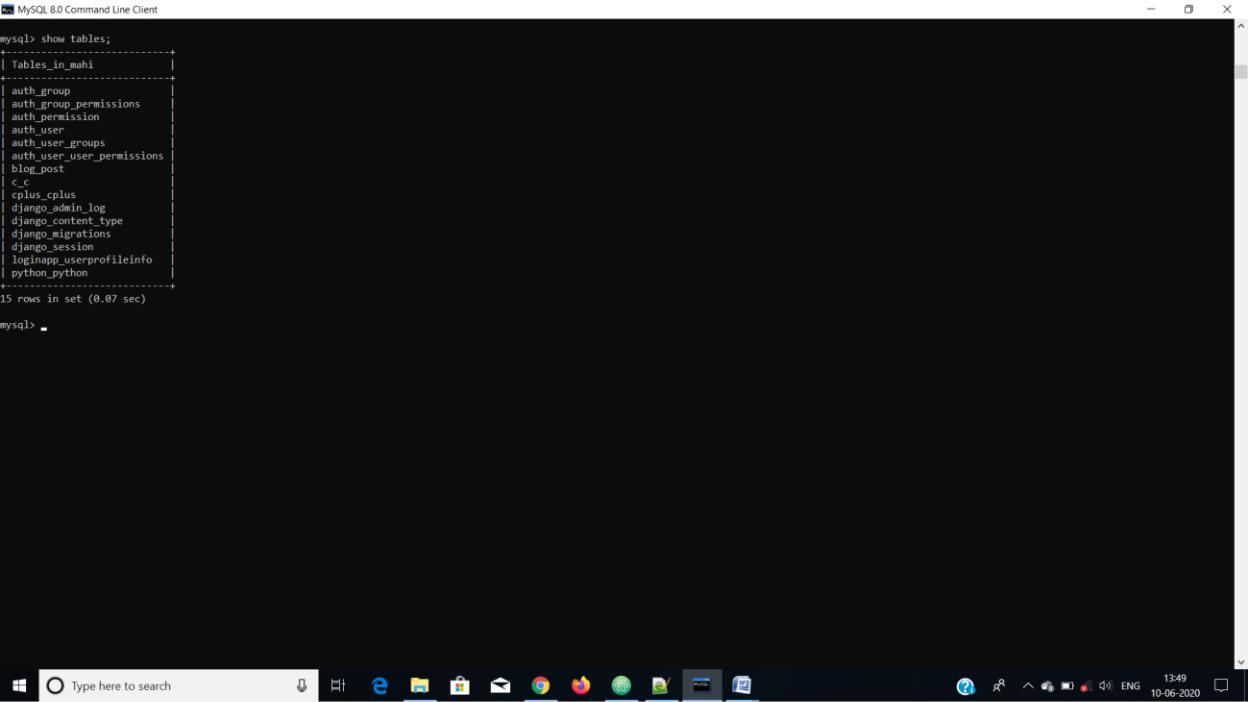
|  |  |
| --- | --- |
| **FILE NAME** | **DESCRIPTION** |
|  | A view function, or “view” for short, is simply |
| Views.py | a Python function that takes a web request and |
|  | returns a web response. This response can be |
|  | the HTML contents of a Web page, or a |
|  | redirect, or a 404 error, or an XML document, |
|  | or an image, etc |
|  | admin.py file are use to display your models in |
| Admin.py | django admin pannel also you can customise |
|  | your admin panel using admin.py file. Django |
|  | has a builtin admin interface that reads |
|  | metadata from your models, such as fields, and |
|  | lets you perform CRUD operations for free |
|  | In url.py, the most important thing is the |
| Urls.py | "urlpatterns" tuple. It's where you define the |
|  | mapping between URLs and views. A mapping |
|  | is a tuple in URL patterns like − from django. |
|  | conf. urls import patterns, include, url from |
|  | django. |
|  | In url.py, the most important thing is the |
| Forms.py | "urlpatterns" tuple. It's where you define the |
|  | mapping between URLs and views. A mapping |
|  | is a tuple in URL patterns like − from django. |
|  | conf. urls import patterns, include, url from |
|  | django. |
|  | This file is created to help the user include any |
| Apps.py | application configuration for the app. Using |
|  | this, you can configure some of the attributes |
|  | of the application. From Application |
|  | Configuration documentation: Application |
|  | configuration objects store metadata for an |
|  | application. |
|  | Django web applications access and manage |
| Models.py | data through Python objects referred to as |
|  | models. Models define the structure of stored |
|  | data, including the field types and possibly also |
|  | their maximum size, default values, selection |
|  | list options, help text for documentation, label |
|  | text for forms, etc. |
|  | pytest supports execution of unittest test cases. |

|  |  |  |
| --- | --- | --- |
| Test.py |  | The real advantage of pytest comes by writing |
|  |  | pytest test cases. pytest test cases are a series of |
|  |  | functions in a Python file starting with the |
|  |  | name test\_. |
|  |  | The \_init\_.py files are required to make Python |
| Init.py |  | treat the directories as containing packages; |
|  |  | this is done to prevent directories with a |
|  |  | common name, such as string, from |
|  |  | unintentionally hiding valid modules that occur |
|  |  | deeper on the module search path. |
|  |  | ASGI, or the Asynchronous Server Gateway |
| Asgi.py |  | Interface, is the specification which Channels |
|  |  | and Daphne are built upon, designed to untie |
|  |  | Channels apps from a specific application |
|  |  | server and provide a common way to write |
|  |  | application and middleware code. |
|  |  | settings.py is a core file in Django projects. It |
| Settings.py |  | holds all the configuration values that your |
|  |  | web app needs to work; database settings, |
|  |  | logging configuration, where to find static |
|  |  | files, API keys if you work with external APIs, |
|  |  | and a bunch of other stuff |
|  |  | The Web Server Gateway Interface (WSGI, |
| Wsgi.py |  | pronounced whiskey) is a simple calling |
|  |  | convention for web servers to forward requests |
|  |  | to web applications or frameworks written in |
|  |  | the Python programming language. The current |
|  |  | version of WSGI, version 1.0. 1, is specified in |
|  |  | Python Enhancement Proposal (PEP) 3333. |
|  |  | manage is a thin wrapper around django-admin |
| Manage.py |  | that takes care of two things for you before |
|  |  | delegating to django-admin: It puts your |
|  |  | project's package on sys. path. It sets the |
|  |  | DJANGO\_SETTINGS\_MODULE |
|  |  | environment variable so that it points to your |
|  |  | project's settings file |
|  | **Table 2 : Django Files** | |

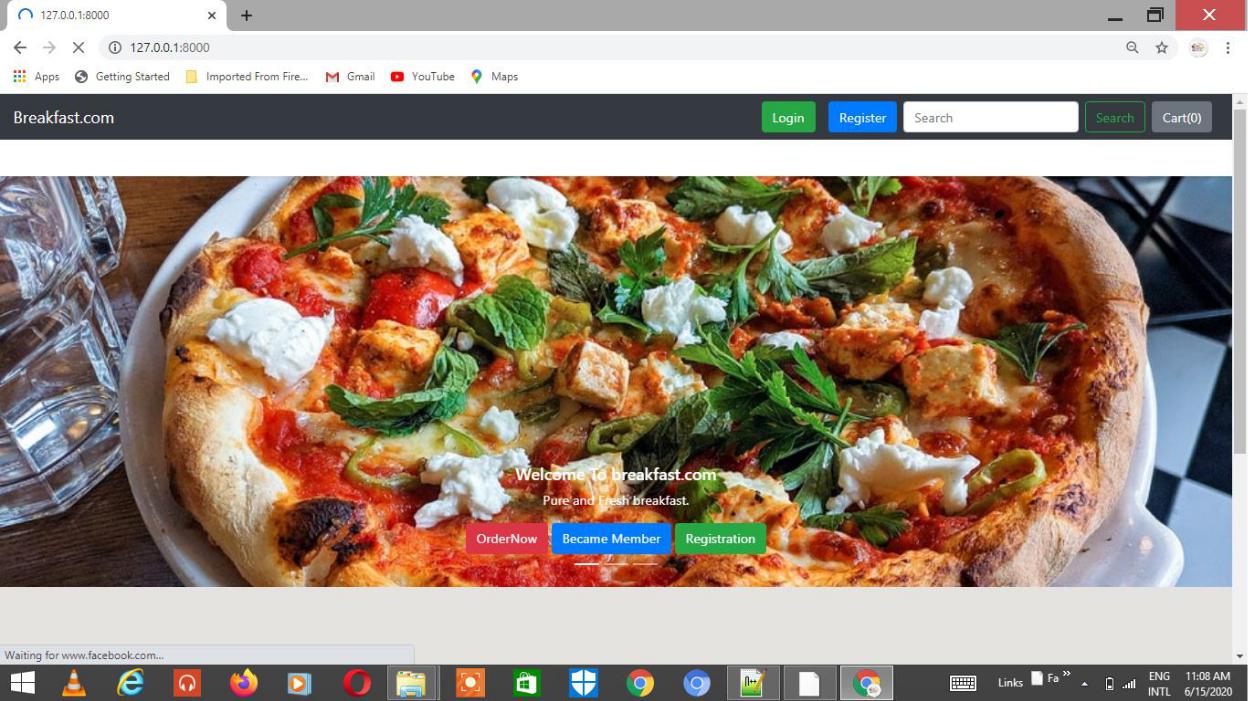
**9. Component Architecture (Screenshots)**



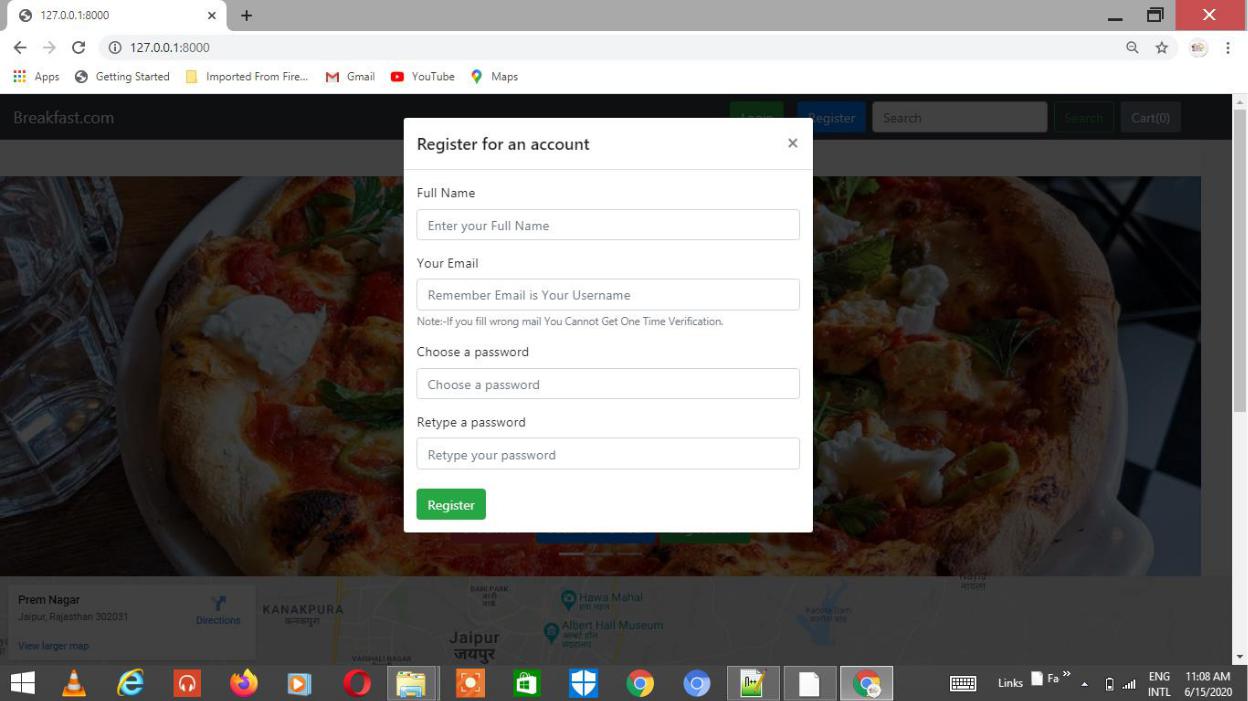
**1.Server**



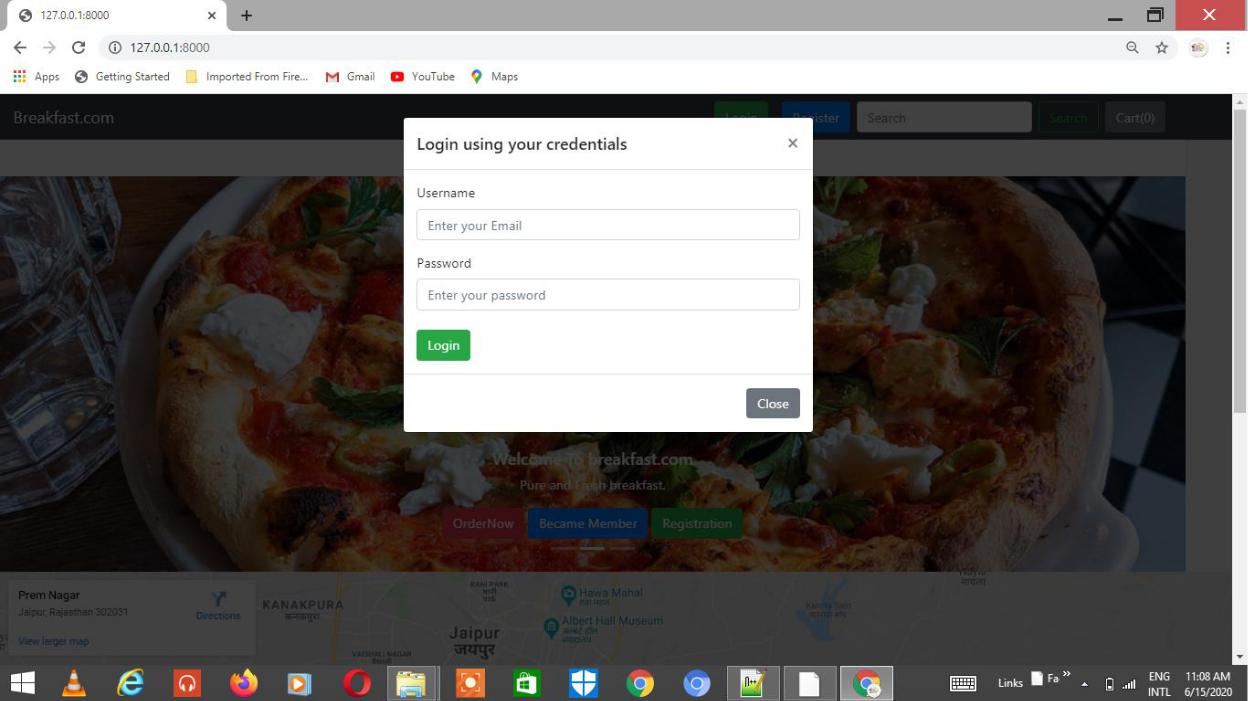
**2.Database**



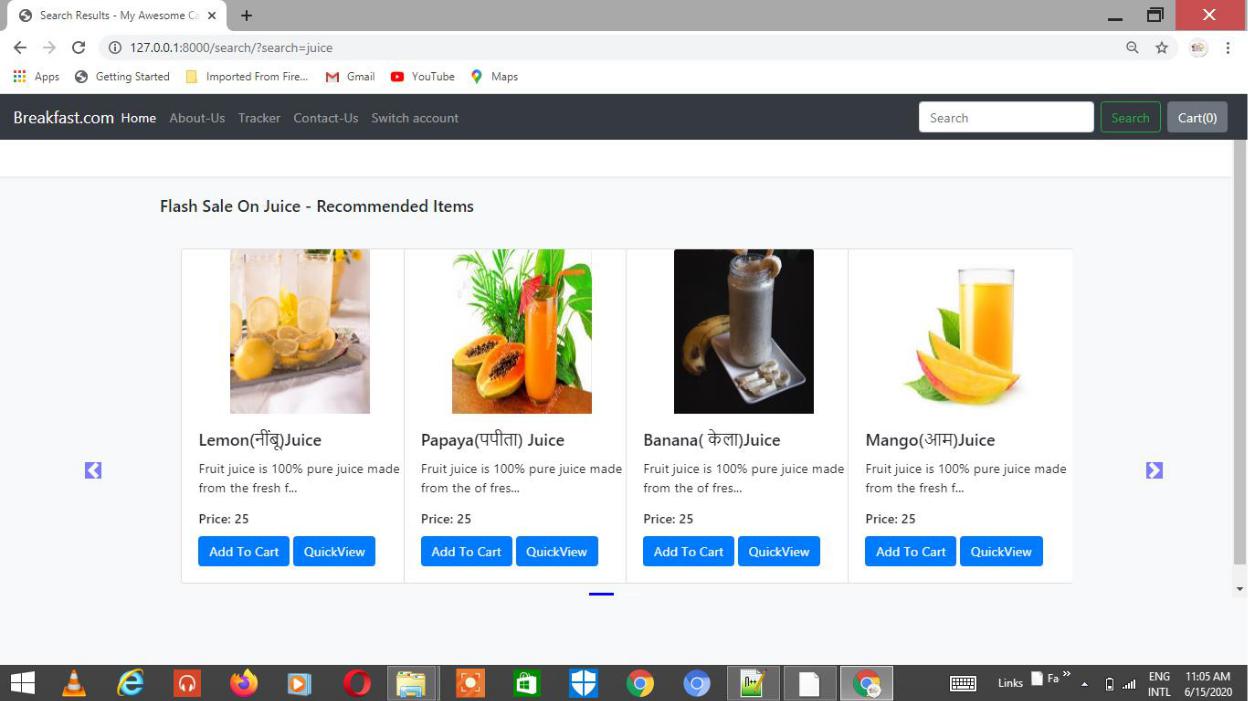
**Homepage**



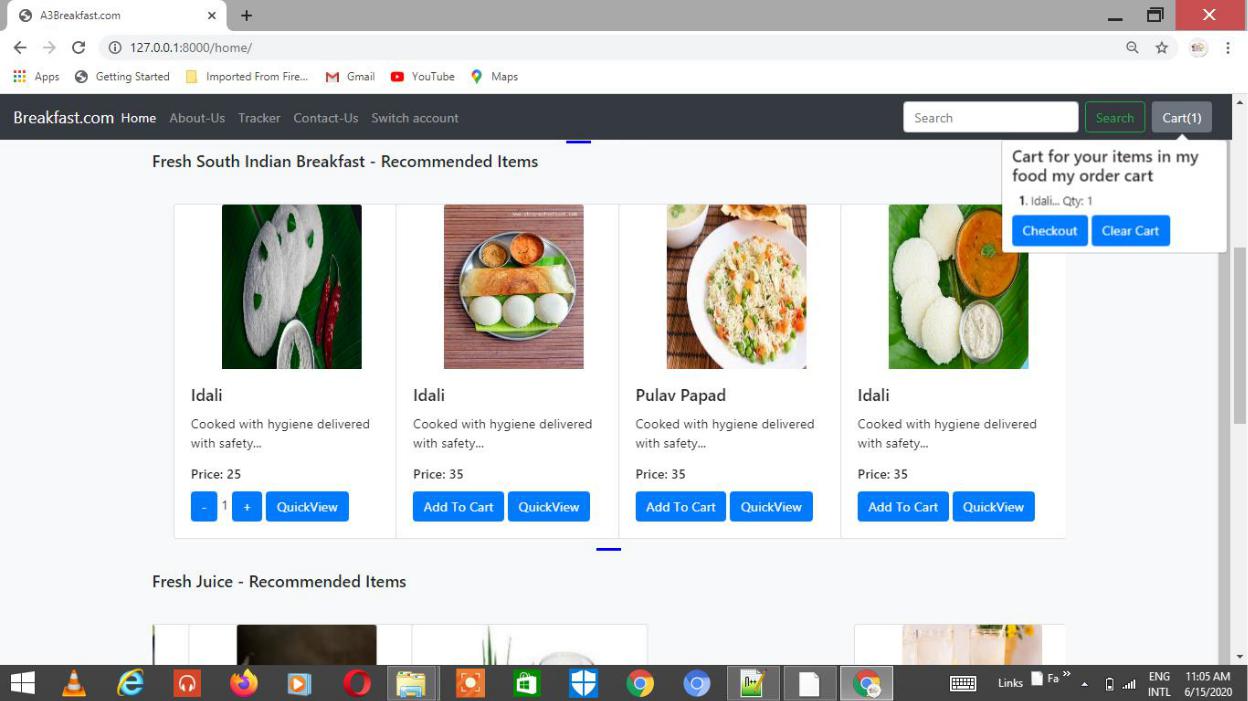
**Registration for new user**



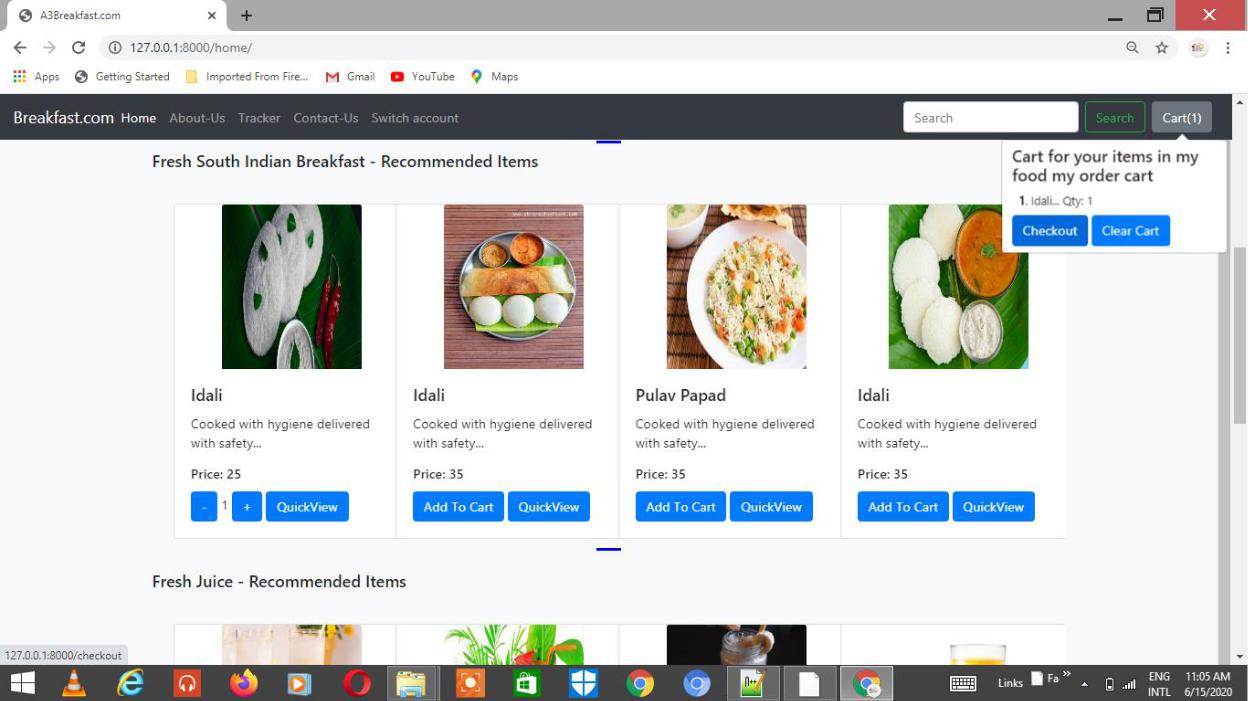
**Login**



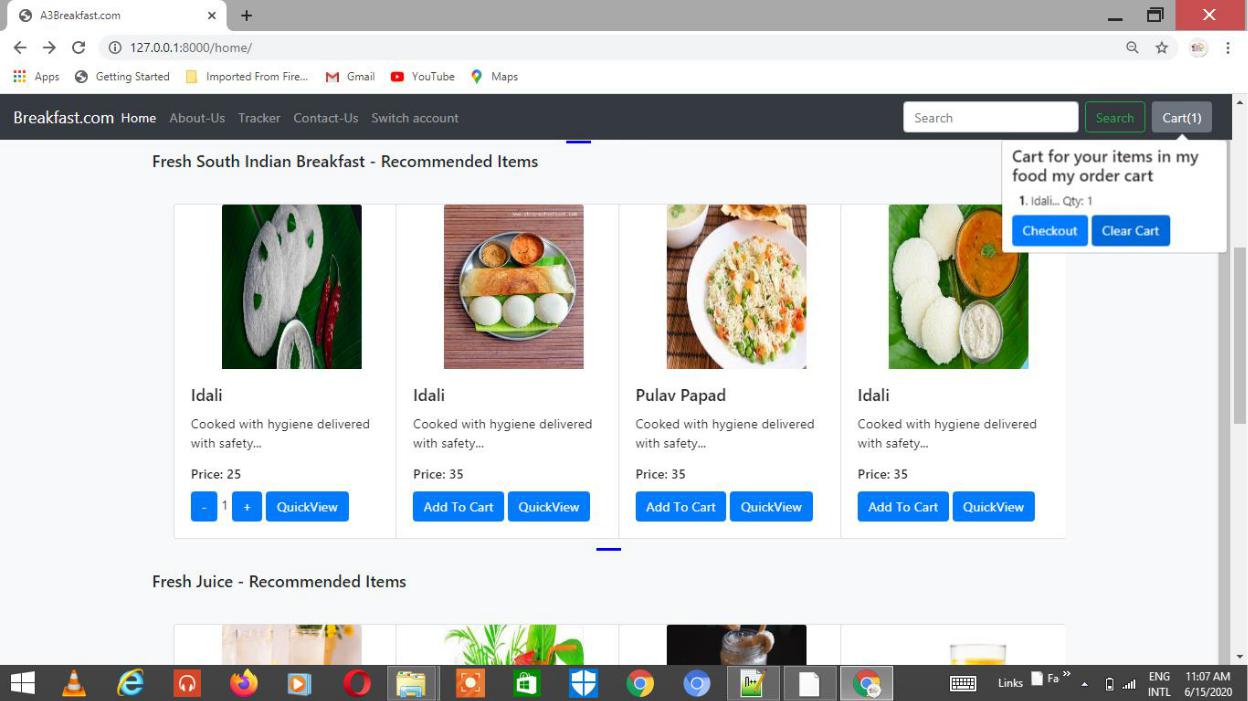
**Item list 1**



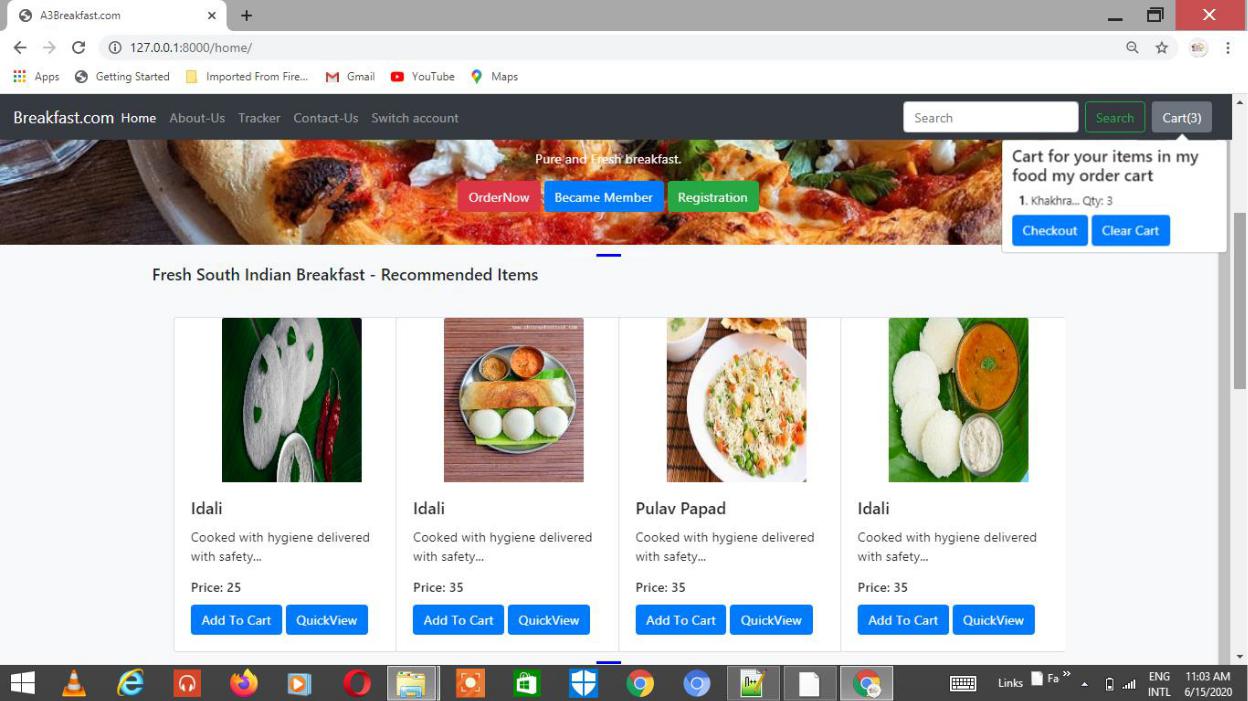
**Item list 2**



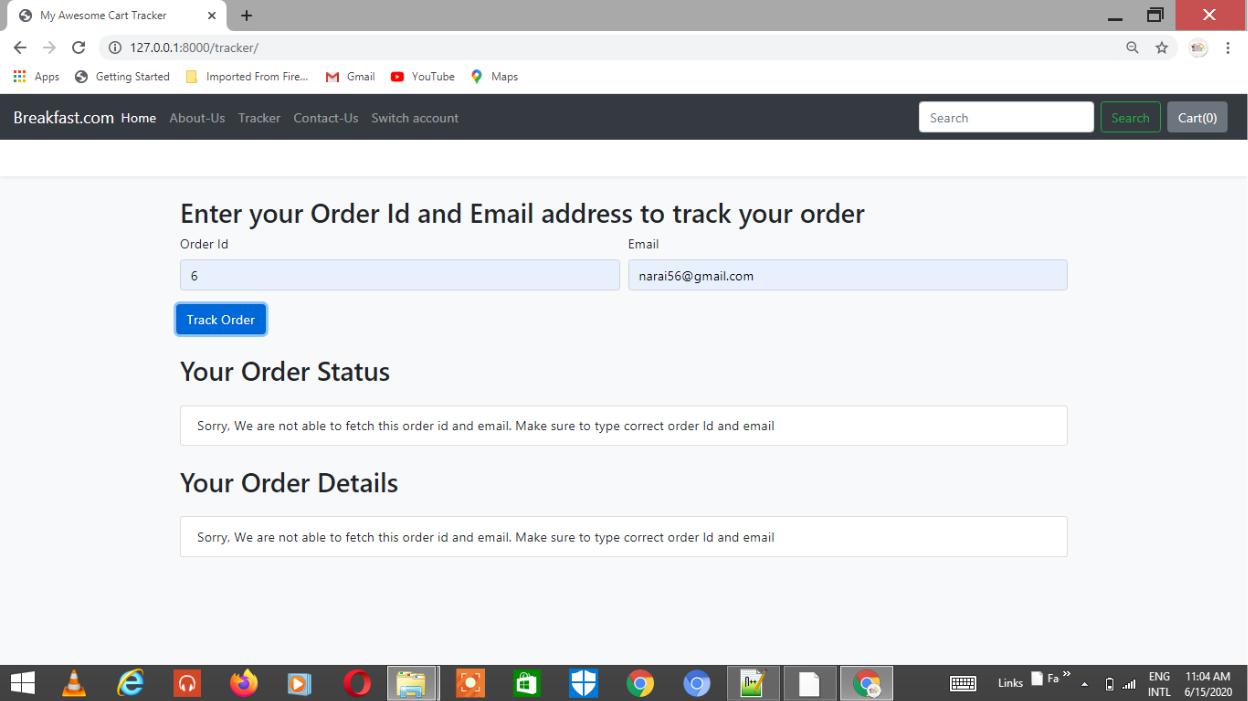
**Item list 3**



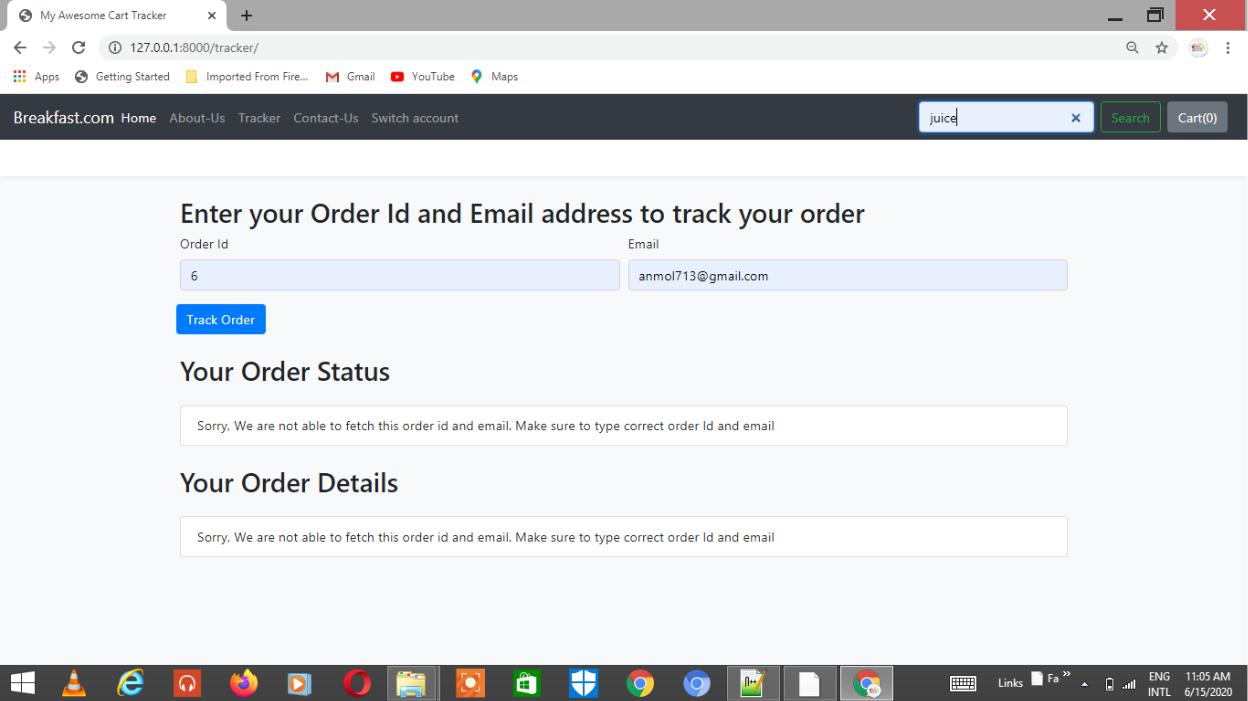
**Item list 4**



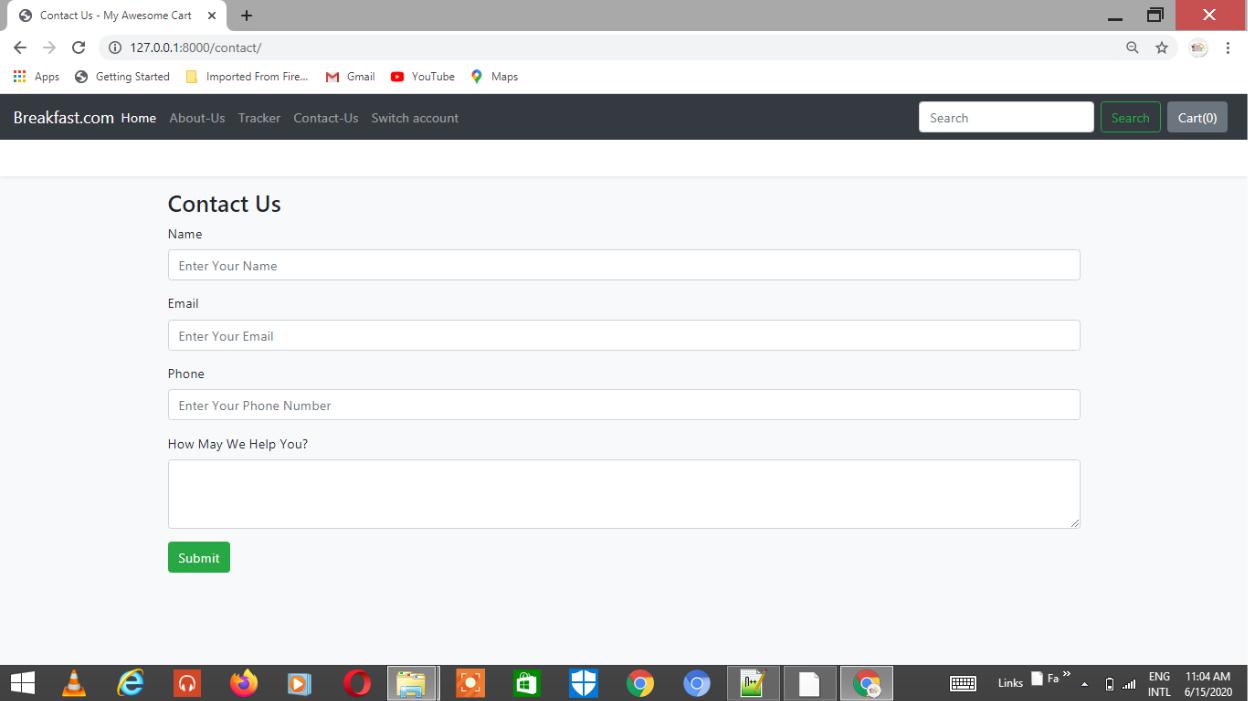
**Adding Products to cart**



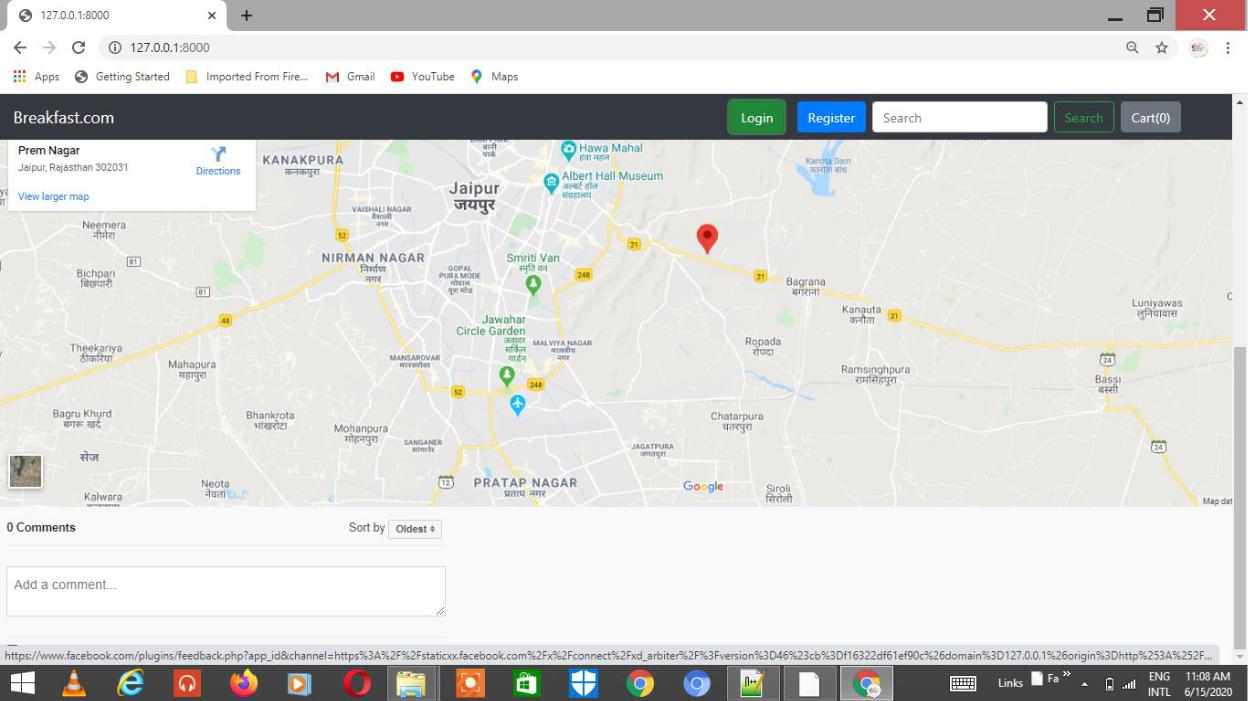
**To Track order status**



**Entering Wrong credentials**

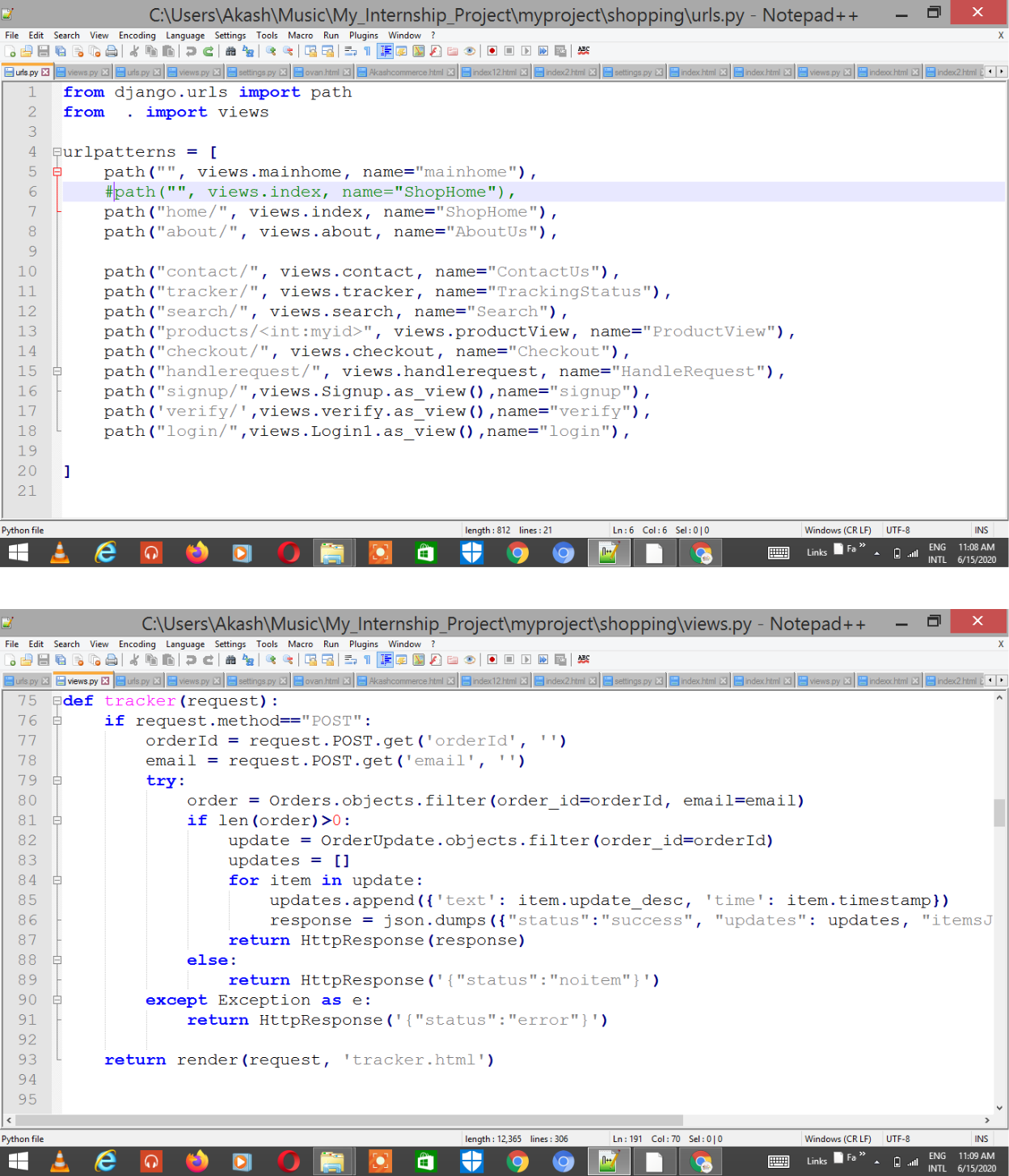


**Customer Care and support**

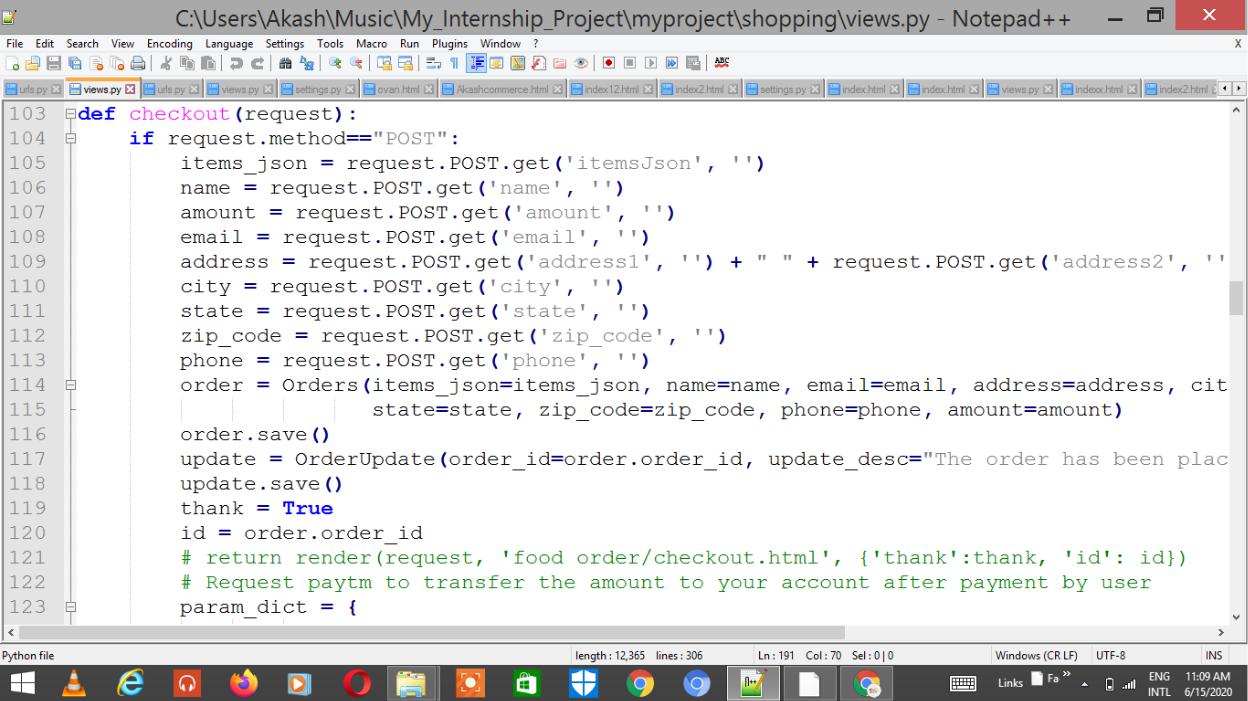
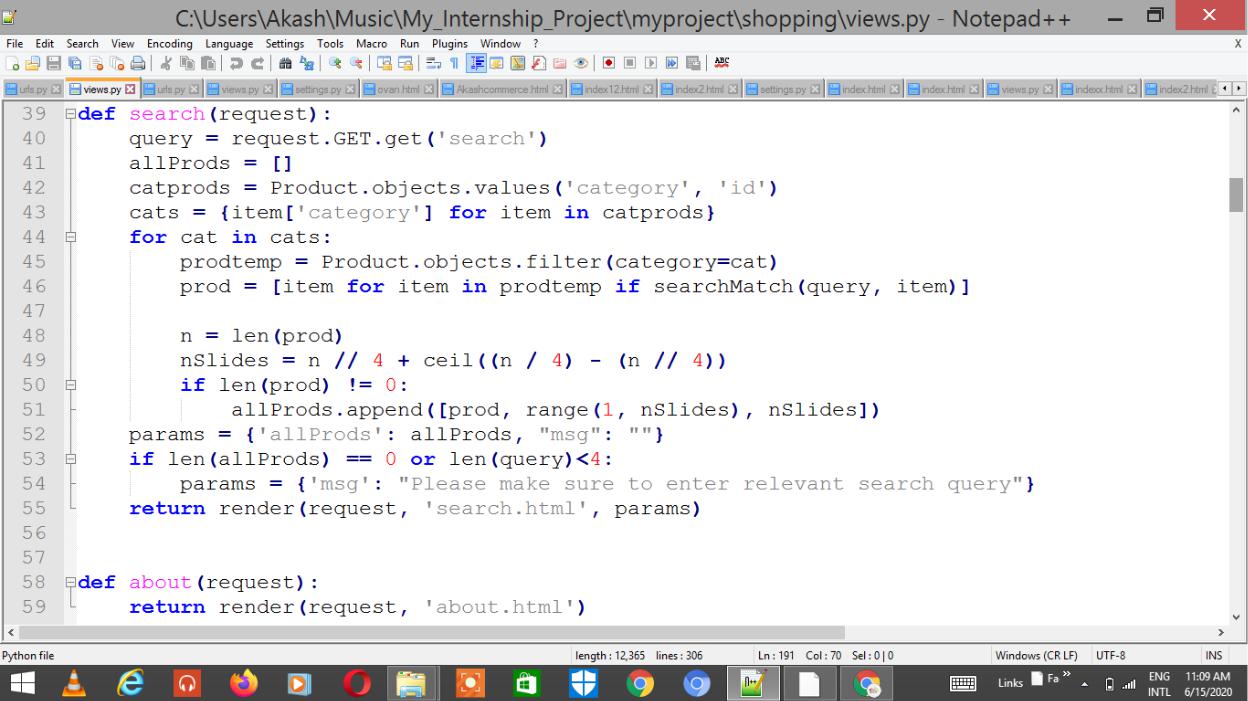


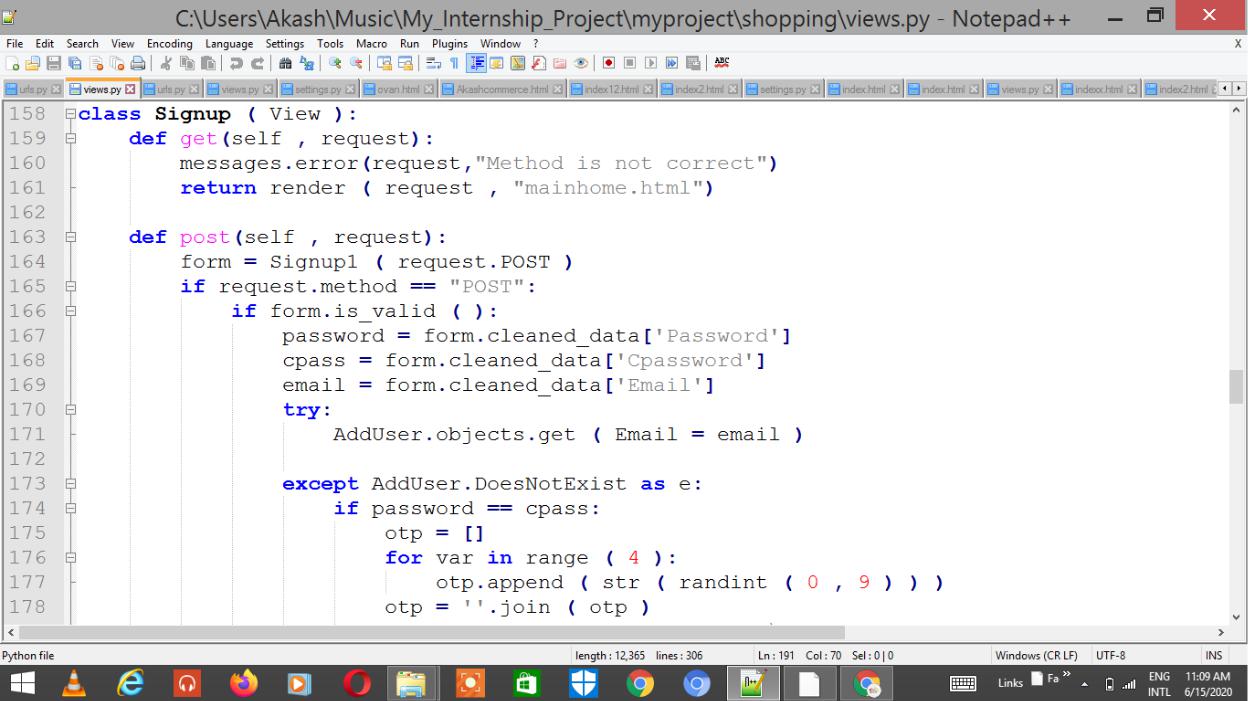
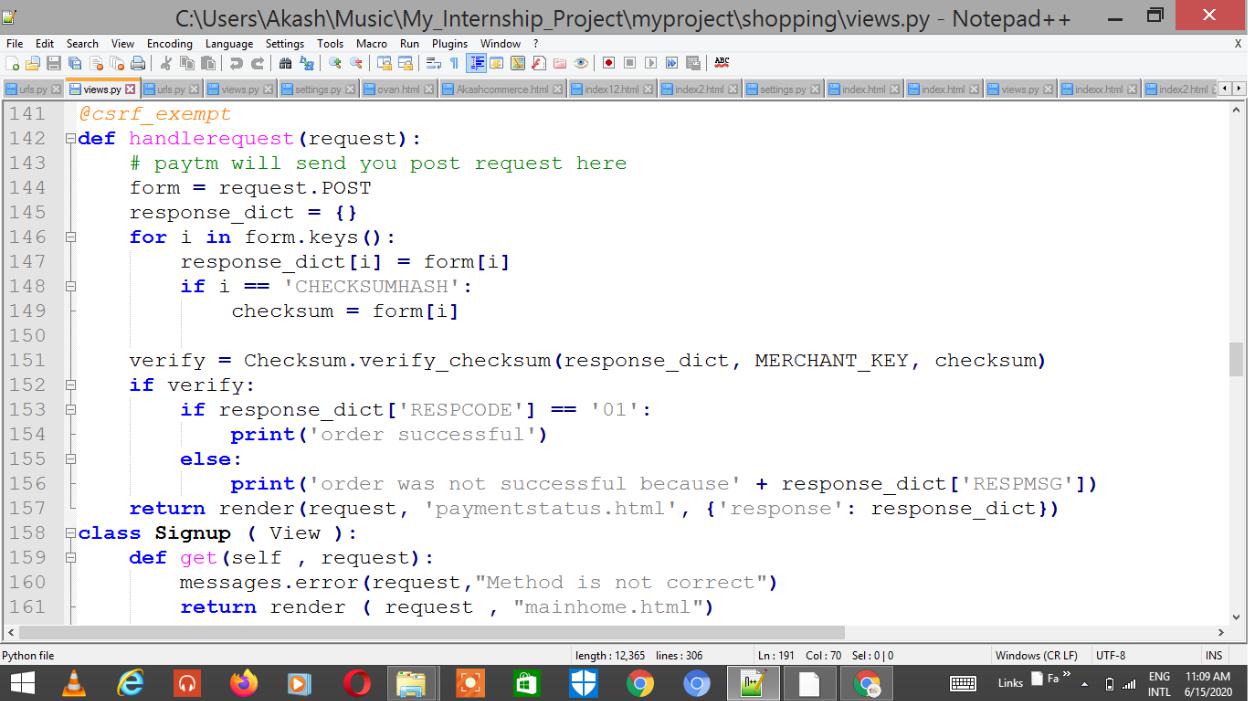
**Locating**

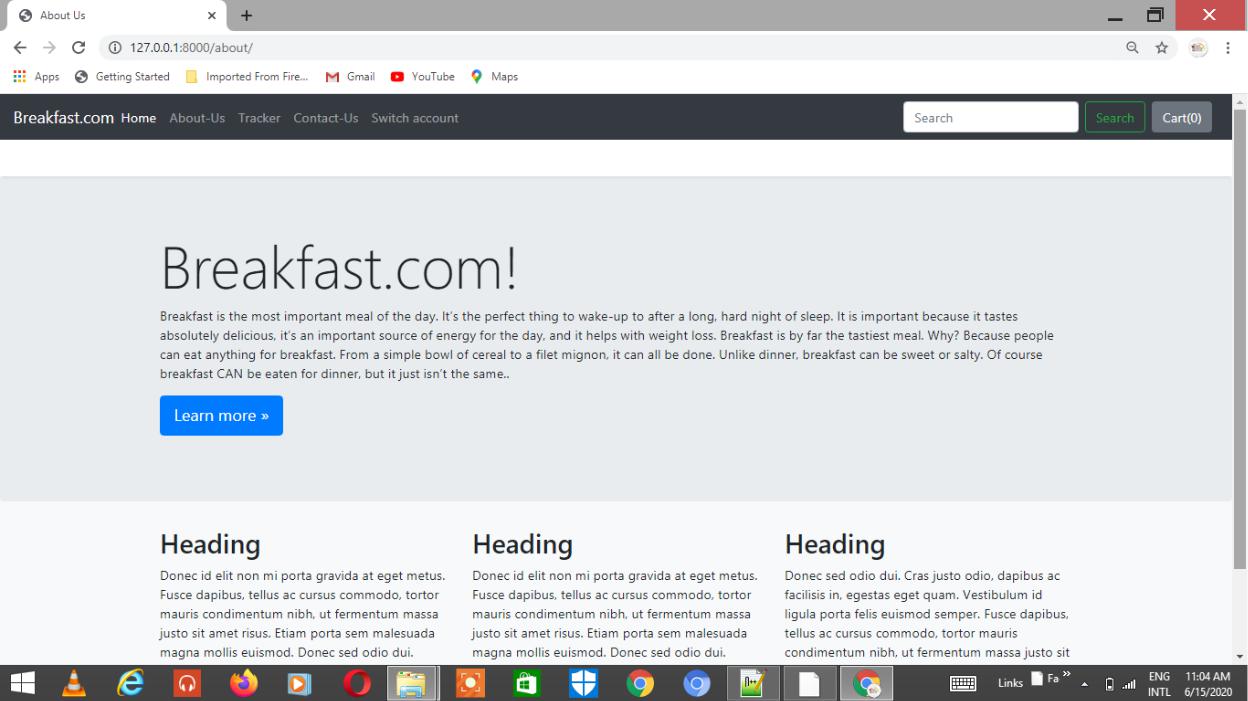
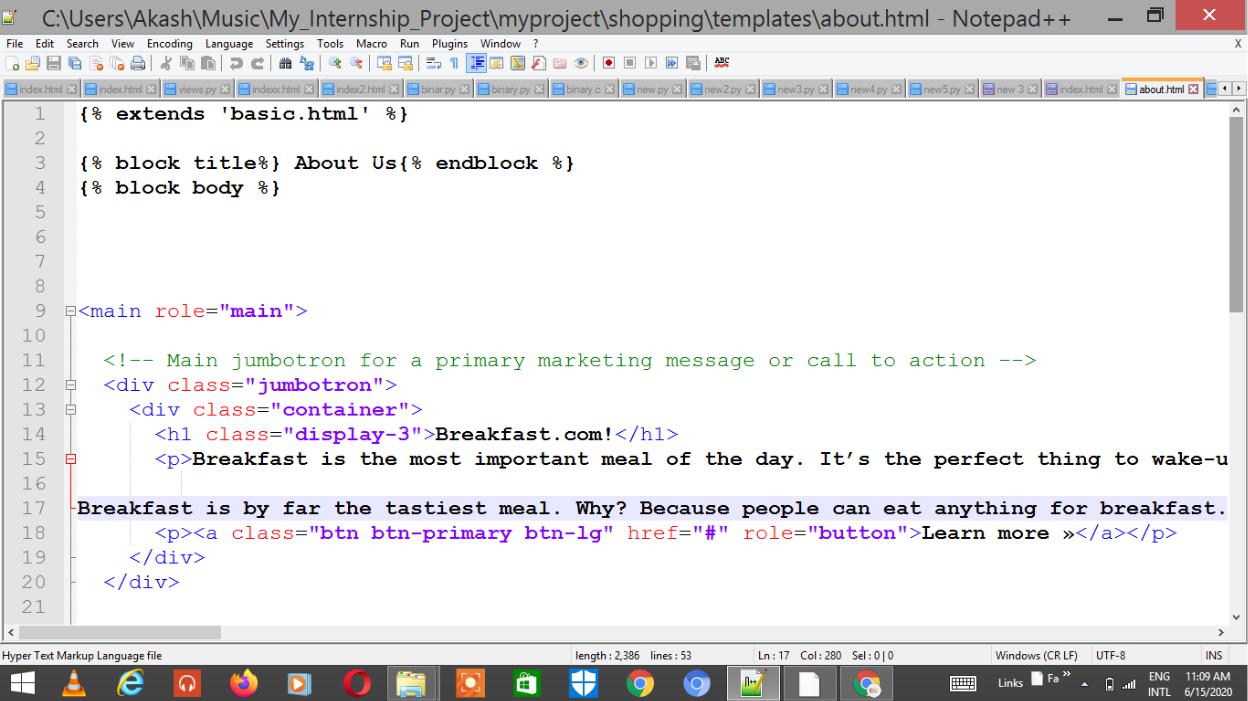
**Models, Views, Urls:-**



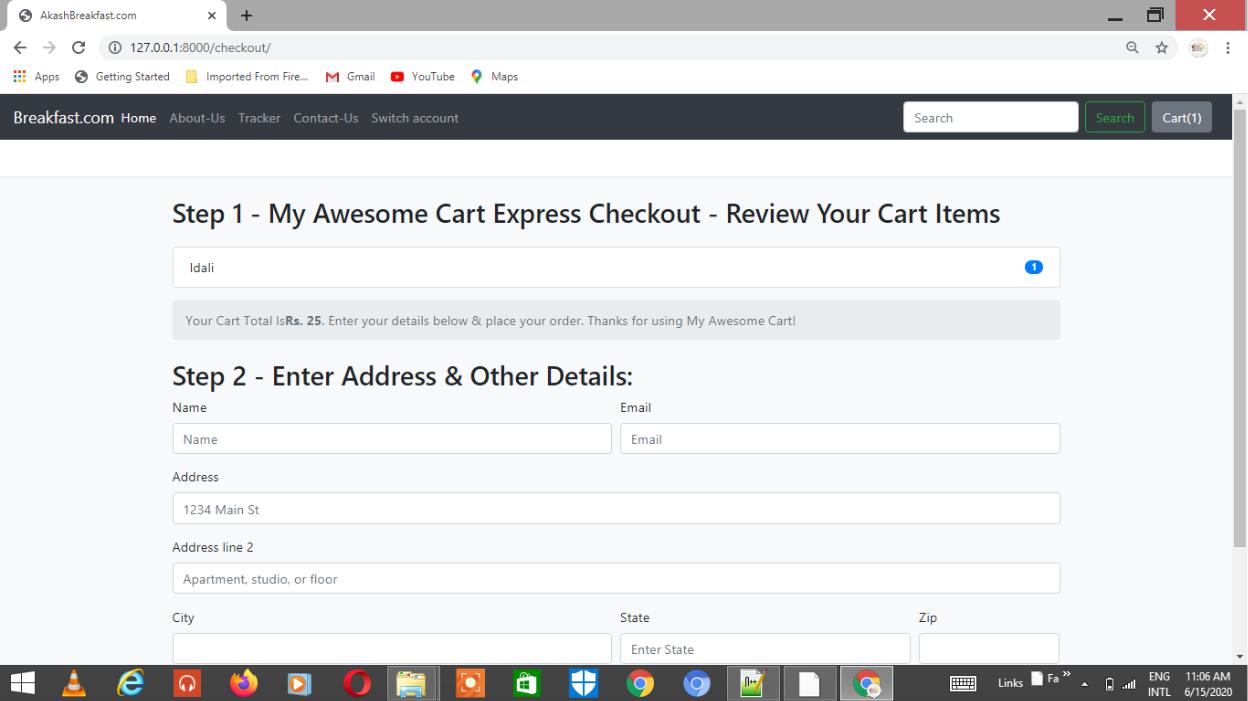
**Views.py of blog**



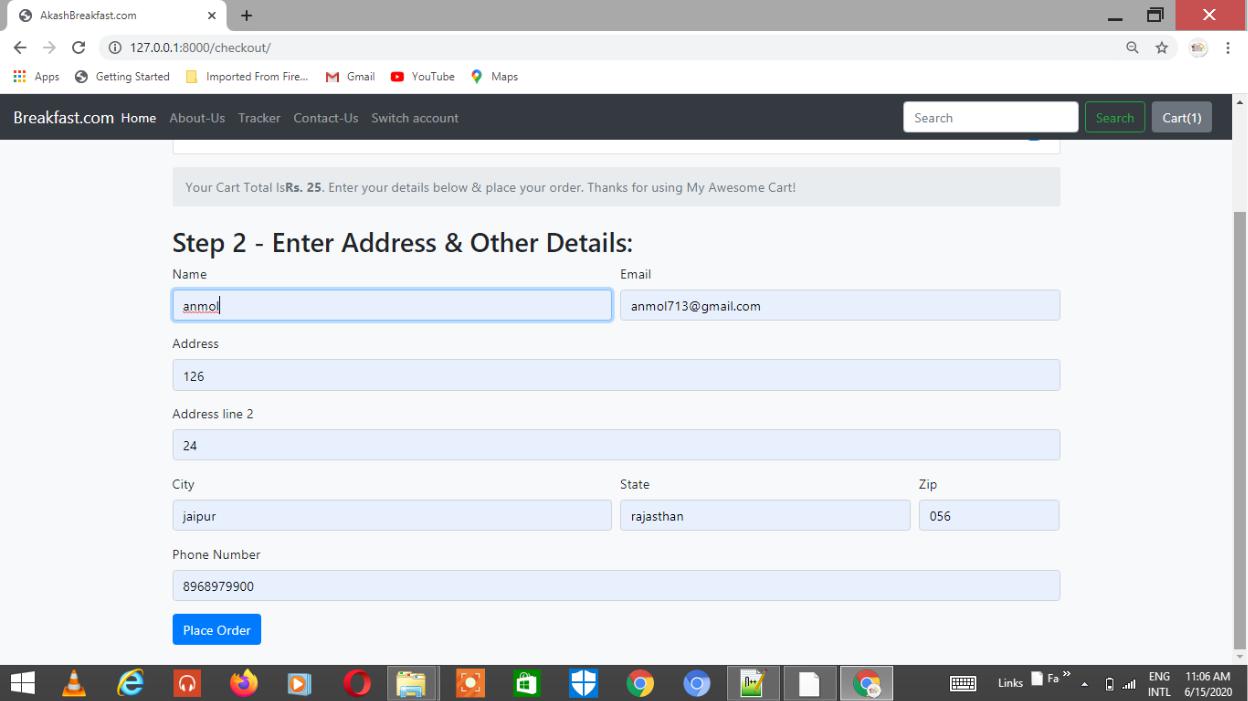


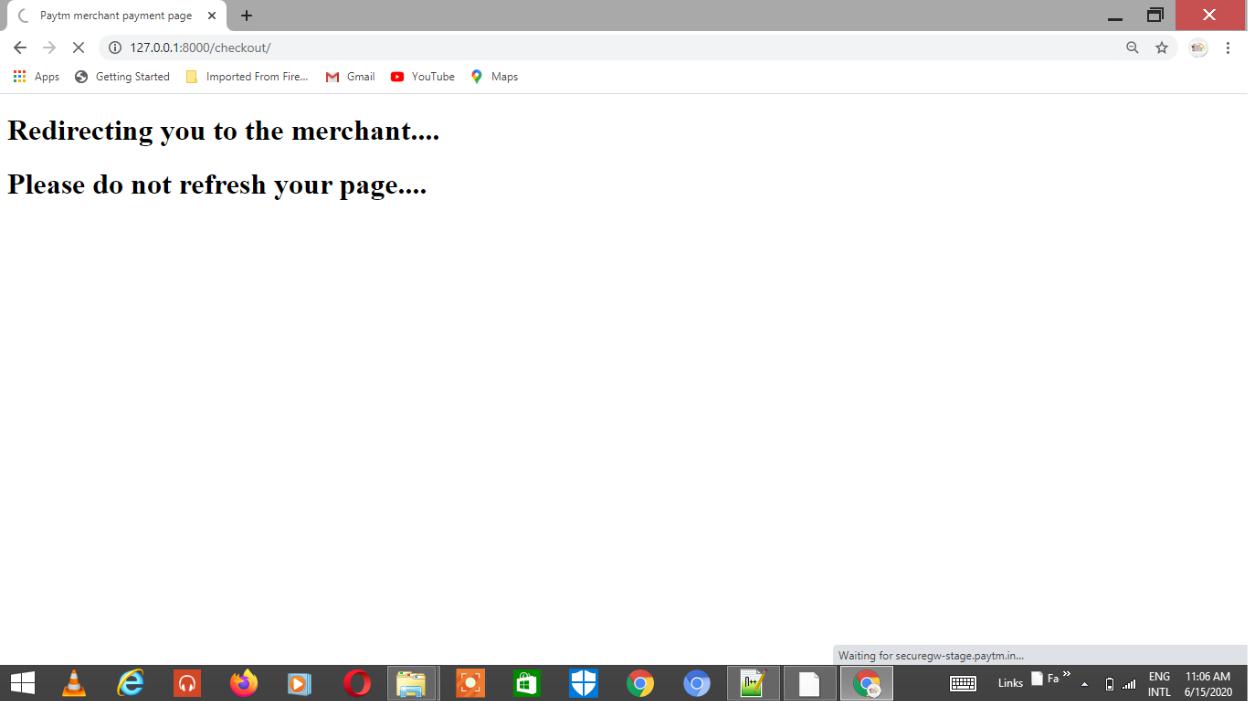


**Tagline**

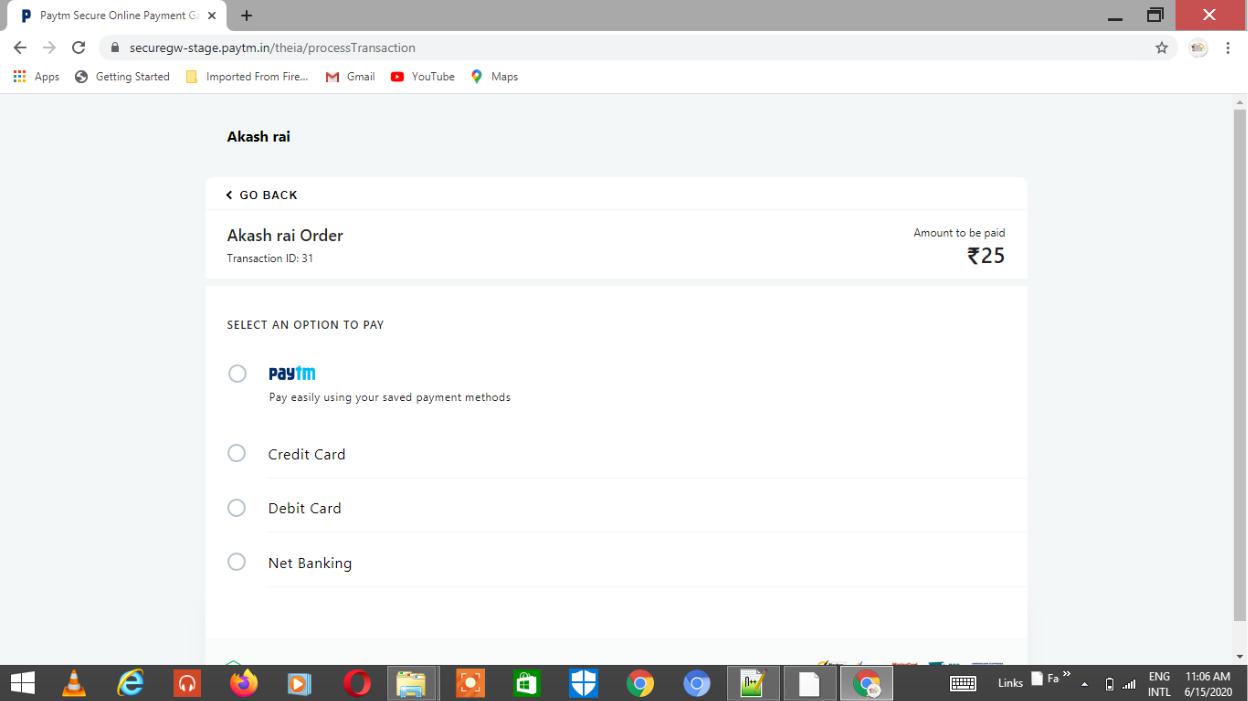


**Cart Experience**

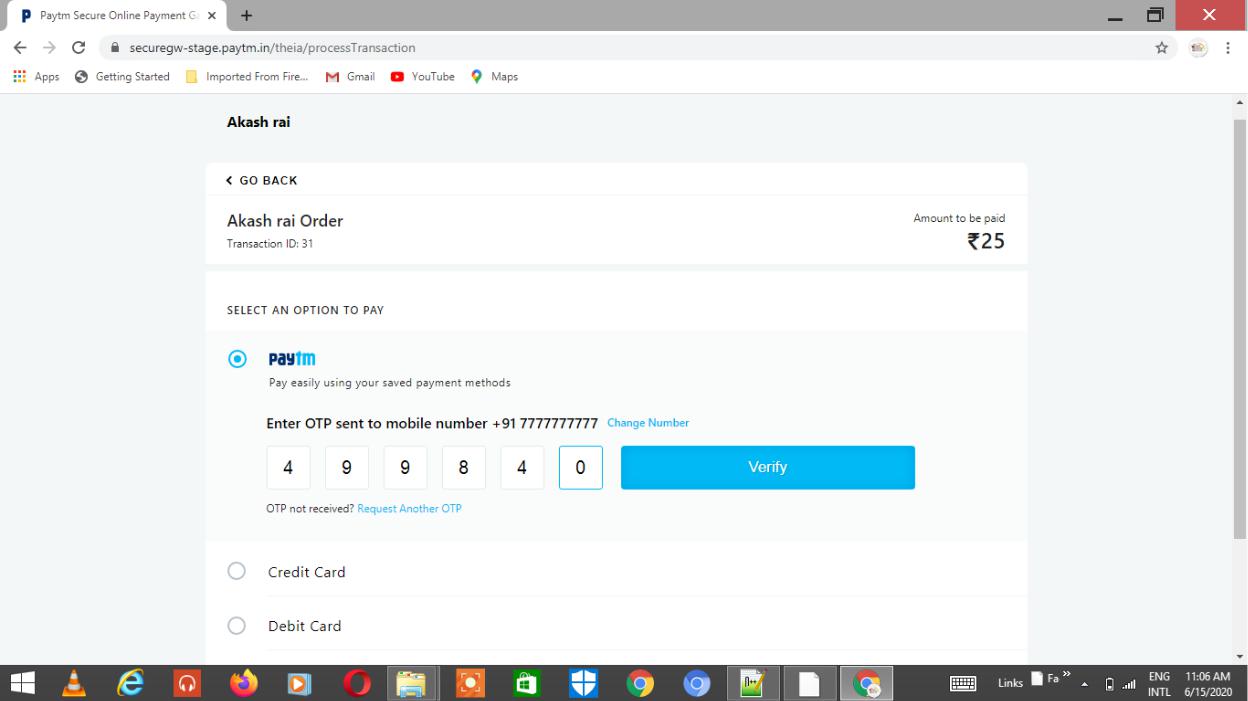
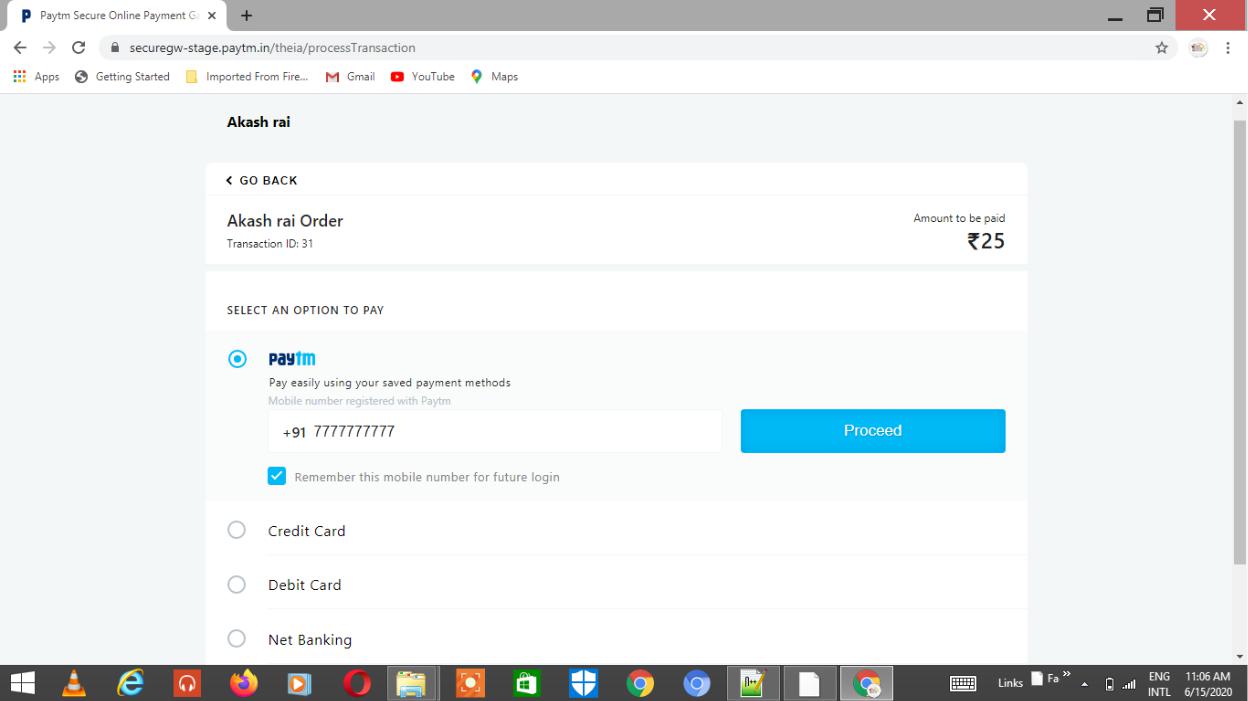




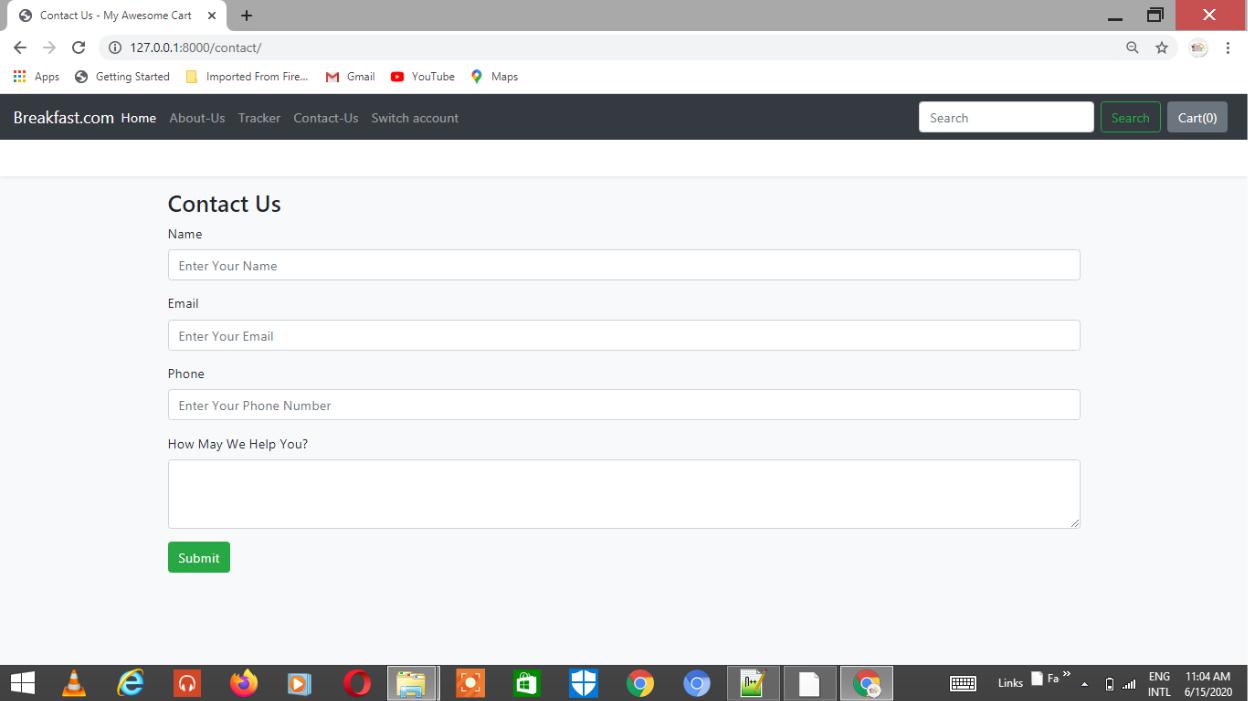
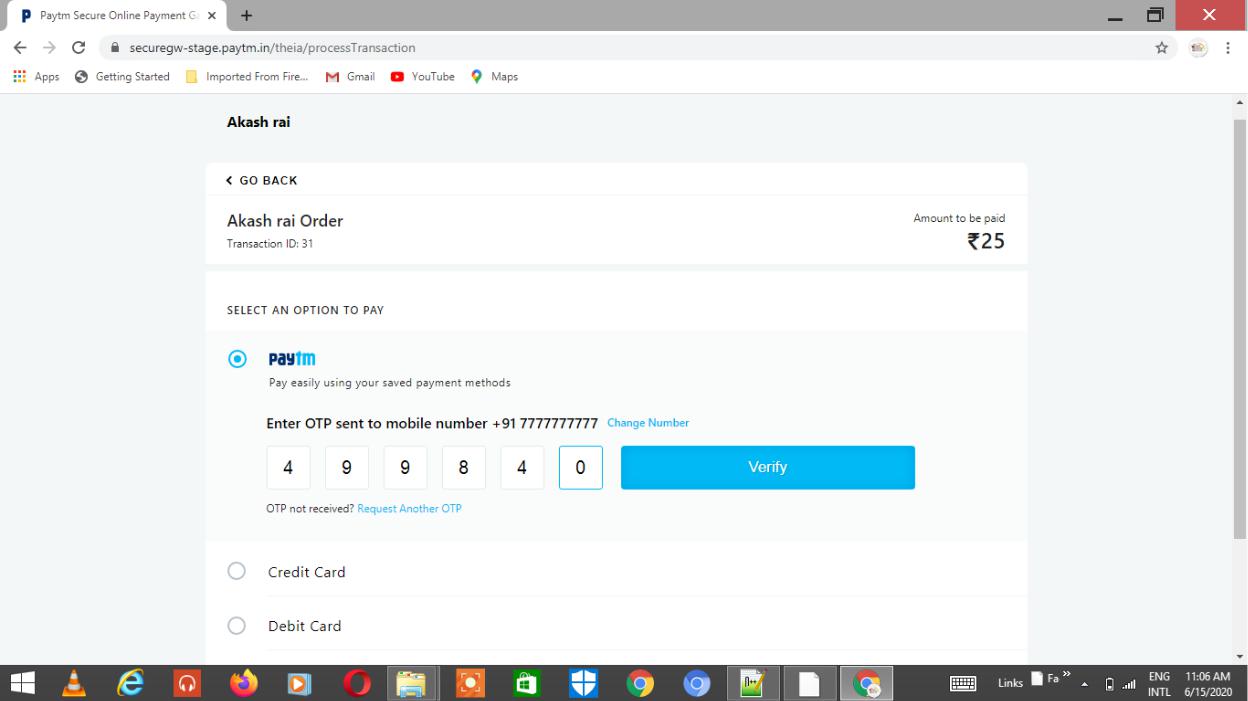
**Redirecting to payment portal**



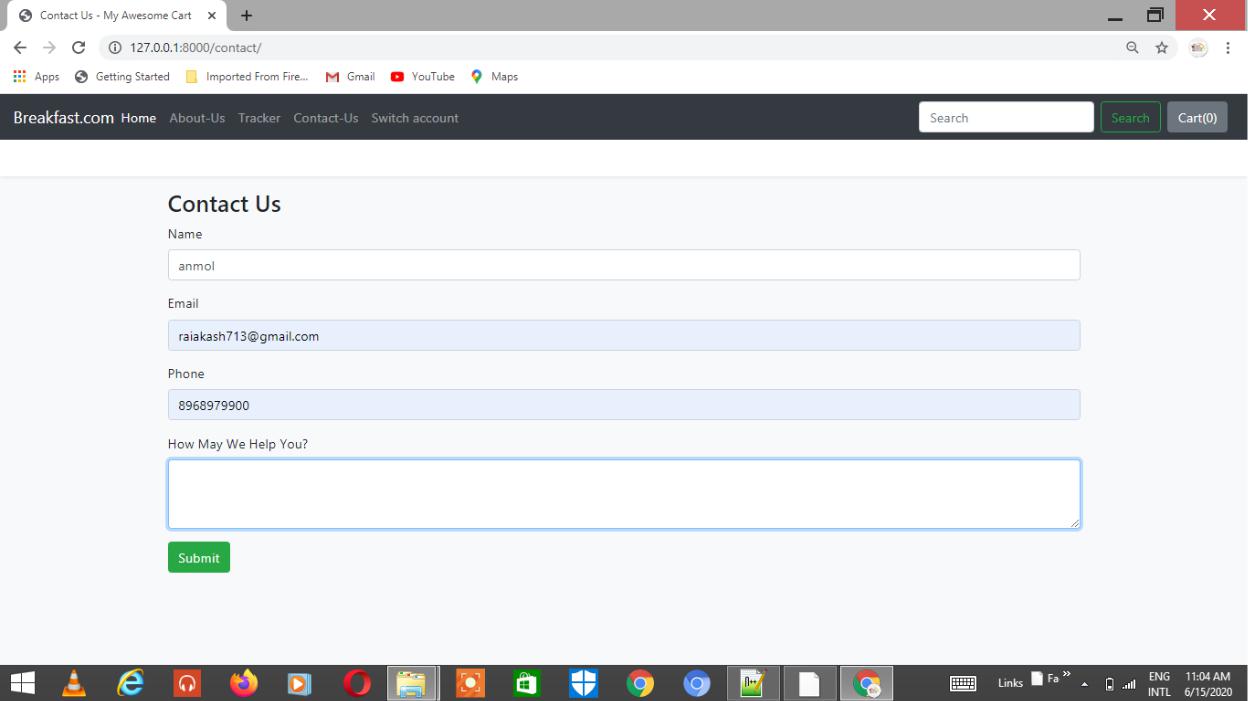
**Payment portal**



**Getting OTP through mobile number**



**FeedBack**



**9. Role / Contribution**

In Tutorial website project which is a project of the company and my responsibility for the project is to implement the desired objective of the project in the python django framework. As by using django for this project, would be a good choice because django provides better and efficient front-end and back-end development of the website.

Pre-requisite learnings needed for working on project are :

1. **Python** : Having a basic fundamental knowledge of the core python
2. **Advanced Python** : It basically includes the connectivity of python programs with thedatabase. Like a tkinter registration form with database connectivity

* 1. **MySQL** : The database included in the project is MySQL, so must have a proper andbasic fundamental knowledge of the mysql statements, database properties, E-R diagrams,etc.
  2. **Django** : The web framework used in project is django and must have a properknowledge of deploying Django server and django is a MTV (Model template View) architecture ,i.e, Models defines the logic of the project, templates contains the web pages of the website and views option give the final view of the website with styling by CSS.
  3. **Atom** : It is an advanced code editor used for editing various types of files like pythonfiles, html files, css files, etc. In creating this project, atom is good to use for editing and accessing the files.

1. **Learning :**

Pre-requisite learnings required for the project :

1. Installation : python, django, code editor.
2. Python : Fundamentals of the python language.
3. Django : Fundamentals of the django (web framework) which are the basic necessity for making django website are as follows :
   1. Django Models
   2. Django Admin
   3. Django server deployment
   4. Django URLs

* 1. Django views
  2. HTML
  3. CSS
  4. Django ORM (Querysets)
  5. Django Templates
  6. Template Extending
  7. Django Forms

1. MySQL : Fundamentals of the MySQL database connectivity.
2. HTML 5 : It is used for creating web pages of the website.
3. CSS 3 : It is used for designing the web pages and for giving better presentation to the website.

**11. Future Enhancements :**

User requirements keep changing as the project is being used. Some of the future enhancements that can be done to this project are :

* As the technology emerges, it’s possible to upgrade the project and may be adaptable to desired environment.
* Based on the future security issues, security may be improved using emerging technologies.

* Sub admin module may be added.

**12 SUMMARY**

At the end of this project work, I was able to design and develop software

that can successfully handle online food ordering and product order

In the process of the design, first hand information on fast food businesses was obtained. This work also will serve as a stepping-stone for people who wish to research more on this topic. Other benefits are:

* 1. Provision of facility for handling text electronically using powerful and sophisticated word processors to produce elegant and error free documents.
  2. In addition to storing the organization’s operational data on disk backing storage, other forms of data used by the organization could also benefit from storage on such medium.
  3. With the installed software, product ordering and delivery was made easier. The systematic approaches used during each phase of the software development provides a clear road map that would be of immense help to anyone carrying out research work in this area.

1. **Conclusion :**

It is hoped that effective implementation of this software product would

eliminate many problems discovered during systems investigation.

The development of online food ordering system involved many phases. The approach used is a top-down one concentrating on *what* first, then *how* and moving to successive levels of details.

The first phase started with a detailed study of the problems and prospects of ordering in breakfast.com. In the course of this study, many problems were discovered to have hindered the effectiveness of the existing manual system. These problems, information needs and activities were documented and later used as the basis for system design, which immediately followed the first phase.

The design phase was concerned primarily with the specification of the system elements in manner that best met the organization’s business needs.

As project is developed using Python Django as front-end and MySQL as back-end.

The goals that are achieved by the software are:-

* Instant access.
* Improved productivity.
* Optimum utilization of resources.
* Simplification of the operations.
* Less time interval and getting required information.
* User friendly.
* Portable and versatile for further enhancement.