Web Development

Prototype

21 August 2019

# 

[**Design**](#_26stetpoeox) **3**

[Technical decision](#_wepug2a6tmj1) 3

[Django](#_hroh70g3enm3) 3

[Bootstrap 4](#_g4puf5gjvnya) 3

[AppEngine Flex, CloudSQL, Cloud Storage](#_j14kixz3sgzb) 3

[Common contents](#_6nior3jxrzhk) 3

[Navbar](#_7br4z3ktwy20) 3

[Main pages](#_g66z3isa77er) 3

[index.html](#_ga5sothuiwg) 3

[contact.html](#_y38vsv7fsfwz) 4

[about.html](#_b26sa0dlr13n) 4

[Implementation Plan](#_rmrr6l63aqwp) 4

[**Development**](#_d9wnzbyqmm5y) **5**

[**Technologies**](#_lwnb839g9kbm) **5**

[**Further Exploration and conclusion**](#_qud8yeeqj916) **5**

[**Video Assignment Part 2**](#_dhjv3ayqnppv) **5**

[**References**](#_q2ne65dx66am) **5**

[**Appendix**](#_3p0ar6sb9wm1) **6**

[Run locally](#_k7mgng8a881) 6

[Tree directory](#_5sws2xok9wmf) 6

[Python files main](#_fqm2brqr88s0) 8

[main.views.py](#_r24daix7pl9) 8

[main.urls.py](#_t3c2m6ol8w5q) 10

[main.models.py](#_dnihcfqp4rz7) 11

[main.forms.py](#_35s5a9ei66zv) 12

[main.query.py](#_ur30vg4rxli6) 12

[Main Templates](#_eoq5i9n620g6) 14

[main.Templates.base.html](#_dpn4o4893f21) 14

[main.templates.navbar.html](#_ynqxgw4oo6lm) 15

[main.templates.footpage.html](#_oppwa55ryexi) 16

[main.template.index.html](#_46v4i3jwx7rl) 16

[main.templates.contact.html](#_1eqqns11f7dq) 18

[main.templates.about.html](#_3yls8te5hky8) 20

[main.templates.search.html](#_3718pkdxropn) 22

[main.templates. detail.html](#_wts9tokhi6n7) 23

[Python files accounts](#_m7oa3bt25pmi) 23

[accounts.views.py](#_8zfthu3d6njl) 23

[accounts.urls.py](#_oxhlk0jihulu) 25

[accounts.models.py](#_b0bbvkka9g4l) 25

[accounts.forms.py](#_79mzwfcq6e07) 26

[Account templates](#_uvlmbxwif3ao) 28

[accounts.templates.confirmation.html](#_a94kckyak6xf) 28

[accounts.templates.edit\_profile.html](#_bs44elgfgcpv) 28

[accounts.templates.signup.html](#_qy6f7cjbjt6) 29

[accounts.templates.view\_profile.html](#_qg9l97jj9k6z) 30

[Account registration templates](#_cmgeea45d7nr) 31

[accounts.templates.registration.login.html](#_x0ew24lj5ryd) 32

[accounts.templates.registration.password\_rest\_complete.html](#_jqye7msnwmj8) 32

[accounts.templates.registration.password\_rest\_confirm.html](#_lrgort4yfcbz) 33

[accounts.templates.registration.password\_rest\_done.html](#_2agh67zi8sk6) 33

[accounts.templates.registration.password\_rest\_form.html](#_76kxmk54i229) 34

# Design

The application will be a website to display the travel offer of the company B&B containing main pages about the offers, and the company details, a profile section, and an employee restricted area.

## Technical decision

#### Django

Django was chosen as a framework, as written in python a language that the developer master, the framework allow to simply create basic templates, that can extend one another like with class extension, this avoids duplicate code in the HTML pages, decreasing the risk of typo or copy past mistake between pages. As well a change will be reflected in all of the different views that use a specific template.

Django offer as well a wonderful way to handle the database creation through the models.py, declaring manyTomany or primary Key directly in the model file that will then be reflected in the database. Last but not least, django as an important community, it is then really easy to find full app modules that could potentially be added to the project. An app used in this project is to give an example django-crispy-forms that adapt the automatically created form (present in django) to be viewed with bootstrap properties.

#### Bootstrap 4

Bootstrap allow to easily create a responsive design that will be handled in most of the devices size. The framework allows the definition of different behaviours for each component depending on the device size that should normally be defined with @media.

#### AppEngine Flex, CloudSQL, Cloud Storage

These technologies have been used as many companies deploy now their products directly into Cloud provider. App engine can be easily deployed, and static files stored in Cloud Storage. The CLoud SQL support MySQL.

## Database Design

The database as been designed to handle more information from the User, that is defined in django, by using the model UserPorfile. The orders have a user as a foreigner key to enable to link orders to the user that made it, and a many to many relationship between the items and orders as orders can have multiple items on them.



## Common contents

#### Navbar

The navbar display a menu at the top of the page to allow easy navigation through the website, different button will appear depending on if the user is logged in or out (profile/ sign in/ log in/ log out).

#### Search input

The search input located in the navbar is available from all pages, and will search in the database for offers that contain the specific term used by the user.

## Main pages

#### index.html

Contain all of the current major offers in the carousel at the beginning of the page. And then article with more detailed offered, these articles are directly saved in the database and can then be retrieved and the number to display on the website can be changed.

#### contact.html

The contact page will contain the important information to contact the company as:

address, phone number, Google map, and a contact form that will send a message to a specific email that could possibly be further designed

#### about.html

The about page contain the description about the company and its team

## Admin page

The pre build admin page in django can show all of the desired described models and allow easy creation, modification or deletion of the object. Only superuser or is\_staff members have access to the page and then no further restriction was needed for this prototype, however what the enhancement of the employee section in further development the access to this page could possibly be restricted further.

## Account section

Then a full sub application that will handle the registration, login and user profile, and all of the pages for restoring a lost password.

This section should contain all the templates and views that will be needed to realise a full registration process, excluding the cancellation process that could possibly be handled later on.

The profile view is accessible only when the user is logged in, and display his profile information, as well as all the orders made by him.

## Employee section

By restricting part of the view in the user profile to employee only by using is\_staff we allow these users to get access to detailed view of all the orders, all the users, and all the items. These pages could easily be enhanced with new search modules that will allow employees to query the database with more specific query.

## Implementation Plan

The development of the website will be done in the order described below, in three phases. We will first implement the main pages, to then move to the accounts section and finalise the prototype with the employee section and the search bar. Colour background will differentiate the phases.

The tests were not prioritise in this development phase, as this implementation is considered a prototype that can’t run in a production environment and will need further development in term of security before being considered a finished product.

|  |  |  |
| --- | --- | --- |
| 1 | Create basic django application with app engine | ok |
| 2 | Connect it to CloudSQL | ok |
| 3 | Create base common files as   * base.html * navbar.html | ok |
| 4 | Create main pages, and links them to the app (views, urls, forms)   * index.html * about.html * contact.html | ok |
| 5 | Create models for Database   * Article for the index page * Order * Item | ok |
| 6 | Include article offers to the index page | ok |
| 7 | Connect models to the admin page to easily access the database and be in a position to delete, modify, create objects | ok |
| 8 | Create accounts app | ok |
| 9 | Create signup page with an automatic form   * views * urls * forms * templates | ok |
| 10 | Add login page | ok |
| 11 | create all pages related to password reset | ok |
| 12 | Create view\_profile page with an automatic form   * views * urls * forms * templates | ok |
| 13 | Extend basic django User model to include UserProfile   * address * city * phone Number * picture? | ok |
| 14 | Display edit form into the user profile | ok |
| 15 | display order made by a user into the profile | ok |
| 16 | limit access to some part of the profile page to is\_staff only | ok |
| 17 | add all objects pages for employees only | ok |
| 18 | add search bar for items | ok |
| 19 | testing |  |

# Further Exploration and conclusion

This remains a prototype that can’t be used in production, modification need to be adopted, in term of security, potential phishing attacks. As well as correct testing needs to be implemented to ensure that there is no unexpected behaviour from the app.

# 

# 

# 

# References

This references list contains the main website used as references during this development

Google Cloud. (2019). Running Django in the App Engine flexible environment | Python | Google Cloud. [online] Available at: https://cloud.google.com/python/django/flexible-environment?hl=en [Accessed 21 Aug. 2019].

Stack Overflow. (2019). *Stack Overflow - Where Developers Learn, Share, & Build Careers*. [online] Available at: https://stackoverflow.com [Accessed 21 Aug. 2019].

Mark Otto, a. (2019). *Bootstrap*. [online] Getbootstrap.com. Available at: https://getbootstrap.com/ [Accessed 21 Aug. 2019].

Docs.djangoproject.com. (2019). *Django documentation | Django documentation | Django*. [online] Available at: https://docs.djangoproject.com/en/2.2/ [Accessed 21 Aug. 2019].

Reddit.com. (2019). *Creating user profile page : django*. [online] Available at: https://www.reddit.com/r/django/comments/5jhusv/creating\_user\_profile\_page/ [Accessed 21 Aug. 2019].

Wsvincent.com. (2019). *Django Login/Logout Tutorial (Part 1) - William Vincent*. [online] Available at: https://wsvincent.com/django-user-authentication-tutorial-login-and-logout/ [Accessed 21 Aug. 2019].

# 

# Appendix

## Run locally

How to run locally the application is described at:

<https://cloud.google.com/python/django/flexible-environment?hl=en>

## Tree directory

|  |
| --- |
| | app.yaml | cloud\_sql\_proxy.exe | manage.py | README.md | requirements.txt | tree.txt |  +---accounts | | admin.py | | apps.py | | forms.py | | models.py | | tests.py | | urls.py | | views.py | | \_\_init\_\_.py | |  | +---migrations | | | 0001\_initial.py | | | 0002\_auto\_20190818\_2130.py | | | 0003\_userprofile\_image.py | | | 0004\_auto\_20190819\_2335.py | | | \_\_init\_\_.py | | |  | +---templates | | | confirmation.html | | | edit\_profile.html | | | signup.html | | | view\_profile.html | | |  | | \---registration | | login.html | | password\_reset\_complete.html | | password\_reset\_confirm.html | | password\_reset\_done.html | | password\_reset\_form.html | |  +---main  | | admin.py  | | apps.py  | | forms.py  | | models.py  | | my\_tags.py  | | query.py  | | urls.py  | | views.py  | | \_\_init\_\_.py  | |  | +---migrations  | +---templates  | | about.html  | | base.html  | | carousel.html  | | contact.html  | | detail.html  | | footpage.html  | | index.html  | | navbar.html  | | search.html  | | thanks.html  | |  | +---templatetags  | | | my\_tags.py  | | | \_\_init\_\_.py  | |  +---mysite  | | settings.py  | | urls.py  | | wsgi.py  | | \_\_init\_\_.py  | |  +---static  | | index1.css  | | logo.PNG  | | logo\_small.png  | | plan.jpg  | | \---carousel  | architecture-buildings-city-290595.jpg  | by-the-sea-greece-holiday-1365991.jpg  | images (1).jpg  | images.jpg |

## Python files main

#### main.views.py

|  |
| --- |
| import os from django.core.mail import BadHeaderError, EmailMessage from django.contrib.auth.decorators import user\_passes\_test from django.conf import settings from django.http import HttpResponse, Http404 from django.shortcuts import render, redirect from django.urls import reverse\_lazy from .models import Article, Order, Item from accounts.models import UserProfile from .forms import ContactForm from .query import get\_query  def index(request):  """Function collecting the data needed to send to the UI for index page.  """  context = {"home\_page": "active"}  try:  p = Article.objects.order\_by('pub\_date')[0:2]  except Article.DoesNotExist:  raise Http404("Poll does not exist")  files = os.listdir(os.path.join(settings.STATIC\_ROOT,"carousel"))  return render(request, 'index.html', {'files' : files, 'articles': p}, context)  def about(request):  """Function collecting the data needed to send to the UI for about page.  """  context = {"about\_page": "active"}  return render(request, 'about.html', context)  def contact(request):  """Function collecting the data needed to send to the UI for contact page.  """  if request.method == 'GET':  form = ContactForm()  else:  form = ContactForm(request.POST)  if form.is\_valid():  contact\_name = form.cleaned\_data['contact\_name']  contact\_email = form.cleaned\_data['contact\_email']  content = form.cleaned\_data['content']  try:  email = EmailMessage(contact\_name,  content,  contact\_email,  ['youremail@gmail.com'],   reply\_to=[contact\_email],  )  email.send()  except BadHeaderError:  return HttpResponse('Invalid header found.')  return redirect('./thanks/')  context = {"contact\_page": "active"}   return render(request, 'contact.html', {'form': form}, context)   def thanks(request):  """Function collecting the data needed to send to the UI for thanks page.  """  return render(request, 'thanks.html')  @user\_passes\_test(lambda u:u.is\_staff, login\_url=reverse\_lazy('login')) def all\_orders(request):  """Function collecting the data needed to send to the UI for detail page.  """  obj = Order.objects.all()  return render(request, 'detail.html', {'obj': obj})  @user\_passes\_test(lambda u:u.is\_staff, login\_url=reverse\_lazy('login')) def all\_items(request):  """Function collecting the data needed to send to the UI for detail page.  """  obj = Item.objects.all()  return render(request, 'detail.html', {'obj': obj})  @user\_passes\_test(lambda u:u.is\_staff, login\_url=reverse\_lazy('login')) def all\_users(request):  """Function collecting the data needed to send to the UI for detail page.  """  obj = UserProfile.objects.all()  return render(request, 'detail.html', {'obj': obj})  def search(request):  """Function collecting the data needed to send to the UI for search page.  """  query\_string = ''  found\_entries = None  if ('q' in request.GET) and request.GET['q'].strip():  query\_string = request.GET['q']  entry\_query = get\_query(query\_string, ['title', 'description',])  items = Item.objects.filter(entry\_query)  return render(request, 'search.html', { 'query\_string': query\_string, 'items': items })  else:  return render(request, 'search.html', { 'query\_string': 'Null', 'found\_entries': 'Enter a search term' }) |

#### main.urls.py

|  |
| --- |
| from django.contrib import admin from django.urls import path, include from django.conf.urls import url from . import views  urlpatterns = [  path('', views.index, name='index'),  path('about/', views.about, name='about'),  path('contact/', views.contact, name='contact'),  path('thanks/', views.thanks, name='thanks'),  path('all\_items/', views.all\_items, name='all\_items'),  path('all\_users/', views.all\_users, name='all\_users'),  path('all\_orders/', views.all\_orders, name='all\_orders'),  path('accounts/', include('accounts.urls')),  path('search/', views.search, name="search") ] |

#### main.models.py

|  |
| --- |
| from uuid import uuid4 from django.db import models from django.contrib.auth.models import User   class Article(models.Model):  """  Model declaring the fileds for article object.  """  title = models.CharField(max\_length=100)  text = models.CharField(max\_length=5000)  pub\_date = models.DateTimeField('date published')   def \_\_str\_\_(self):  return self.title   class Order(models.Model):  """  Model declaring the fields for order object.  """  user = models.ForeignKey(  User,  on\_delete=models.CASCADE,)  mykey = models.CharField(max\_length=100, unique=True, default=uuid4, editable=False)  date\_order = models.DateTimeField('date paid')  items = models.ManyToManyField("Item", related\_name="order\_items", blank=True)   class Item(models.Model):  """  Model declaring the fields for item object.  """  title = models.CharField(max\_length=50)  description = models.CharField(max\_length=200)   def \_\_str\_\_(self):  return self.title   def \_\_unicode\_\_(self):  return self.title |

#### main.forms.py

|  |
| --- |
| from django import forms from django.contrib.auth.models import User  class ContactForm(forms.Form):  """Class extending Form with the required fields.  """  contact\_name = forms.CharField(required=True, label="Name")  contact\_email = forms.EmailField(required=True, label="Email")  content = forms.CharField(  required=True,  widget=forms.Textarea,  label="Message"  ) |

#### main.query.py

|  |
| --- |
| import re  from django.db.models import Q  def normalize\_query(query\_string,  findterms=re.compile(r'"([^"]+)"|(\S+)').findall,  normspace=re.compile(r'\s{2,}').sub):  ''' Splits the query string in invidual keywords, getting rid of unecessary spaces  and grouping quoted words together.  Example:   >>> normalize\_query(' some random words "with quotes " and spaces')  ['some', 'random', 'words', 'with quotes', 'and', 'spaces']   '''  return [normspace(' ', (t[0] or t[1]).strip()) for t in findterms(query\_string)]  def get\_query(query\_string, search\_fields):  ''' Returns a query, that is a combination of Q objects. That combination  aims to search keywords within a model by testing the given search fields.   '''  query = None # Query to search for every search term   terms = normalize\_query(query\_string)  for term in terms:  or\_query = None # Query to search for a given term in each field  for field\_name in search\_fields:  q = Q(\*\*{"%s\_\_icontains" % field\_name: term})  if or\_query is None:  or\_query = q  else:  or\_query = or\_query | q  if query is None:  query = or\_query  else:  query = query & or\_query  return query    '''  taken from https://www.julienphalip.com/blog/adding-search-to-a-django-site-in-a-snap/  ''' |

## Main Templates

#### main.Templates.base.html

|  |
| --- |
| <!DOCTYPE html> {% **load** static %} <**html** lang="en"> <**head**>  <**meta** charset="UTF-8">  <**meta** name="viewport" content="width=device-width, initial-scale=1.0">  <**meta** http-equiv="X-UA-Compatible" content="ie=edge">  <**link** rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">  <**link** rel="stylesheet" type="text/css" href="{% **static** 'index1.css' %}">   <**script** src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></**script**>  <**script** src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></**script**>  <**script** src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></**script**>  </**head**> <**body**>  {% **include** 'navbar.html' %}  {% **block** content %}{% **endblock** %}  {% **include** 'footpage.html' %} </**body**> </**html**> |

#### main.templates.navbar.html

|  |
| --- |
| <**nav** class="navbar navbar-expand-lg navbar-light bg-light">  <**button** class="navbar-toggler navbar-toggler-right" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">  <**span** class="navbar-toggler-icon"></**span**>  </**button**>  {% **load** static %}  <**a** class="navbar-brand" href="{% **url** 'index' %}"> <**img** src="{% **static** 'logo\_small.png' %}"width="50" height="30" alt=""></**a**>  <**div** class="collapse navbar-collapse" id="navbarSupportedContent">  <**ul** class="navbar-nav">  <**li** class="nav-item">  <**a** class="nav-link" href="{% **url** 'index' %}">Home</**a**>  </**li**>  <**li** class="nav-item">  <**a** class="nav-link" href="{% **url** 'about' %}">About</**a**>  </**li**>  <**li** class="nav-item">  <**a** class="nav-link" href="{% **url** 'contact' %}">Contact</**a**>  </**li**>  <**li** class="nav-item">  {% **if** user.is\_authenticated %}  <**a** class="nav-link" href="{% **url** 'logout' %}">Logout</**a**>  {% **else** %}  <**a** class="nav-link" href="{% **url** 'login' %}">Login</**a**>  {% **endif** %}  </**li**>    <**li** class="nav-item">  {% **if** user.is\_authenticated %}  <**a** class="nav-link" href="{% **url** 'view\_profile' %}">Profile</**a**>  {% **else** %}  <**a** class="nav-link" href="{% **url** 'signup' %}">Sign up</**a**>  {% **endif** %}  </**li**>    </**ul**>  </**div**>  <**form** class="form-inline" action="/search/" method="get">  <**input** class="form-control mr-sm-2" name="q" type="search" placeholder="Search" aria-label="Search">  <**button** class="btn btn-outline-success my-2 my-sm-0" type="submit" >Search</**button**>  </**form**> </**nav**> |

#### main.templates.footpage.html

|  |
| --- |
| <footer class="footer font-small pt-2 bg-secondary">  <**div** class="container">  <**ul** class="list-inline text-center pb-2">  <**li** class="list-inline-item">  <**h5** class="mb-1">Register to the newletters to stay informed!</**h5**>  </**li**>  <**li** class="list-inline-item">  <!-- (TODO)install django newletter module -->  <**a** href="#!" class="btn btn-info btn-rounded">Sign up</**a**>  </**li**>  </**ul**>   </**div**> </footer> |

#### main.template.index.html

|  |
| --- |
| {% extends "base.html" %} {% load static %} {% block content %} <title>Home</title> {% include 'carousel.html' %} <div class="container">  <div class="row">  <div class="col">  <h1>We are here for your dream</h1>  <p>Even though using "lorem ipsum" often arouses curiosity due to its resemblance to classical Latin, it is not intended to have meaning. Where text is visible in a document, people tend to focus on the textual content rather than upon overall presentation, so publishers use lorem ipsum when displaying a typeface or design in order to direct the focus to presentation. "Lorem ipsum" also approximates a typical distribution of spaces in English.   The most common lorem ipsum text reads as follows: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.   The text is derived from Cicero's De Finibus Bonorum et Malorum (On the Ends of Goods and Evils, or alternatively [About] The Purposes of Good and Evil ). The original passage began: Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit (Translation: "Neither is there anyone who loves grief itself since it is grief and thus wants to obtain it"). It is not known exactly when the text acquired its current standard form; it may have been as late as the 1960s. The passage was discovered by Richard McClintock, a Latin scholar who is the publications director at Hampden-Sydney College in Virginia, by searching for citings of the rarely used word 'consectetur' in classical literature.</p>  </div>  </div>  <div class="row">   <img class="img-fluid" style="min-width:100%; max-height:400px;min-height:400px; !important;" src="{% static 'plan.jpg' %}" alt=""> </div> <div class="row">  <h1>Last publication about our travel offering</h1>  </div>  {% for article in articles %} <div class="row">  <div class="col">  <h2>{{article.title}}</h2>   <div class="fakeimg text-center">Fake Image</div>  <p>{{article.text}}</p>  </div>  </div> {% endfor %} </div> {% endblock %} |

#### main.templates.contact.html

|  |
| --- |
| {% **extends** "base.html" %} {% **block** content %} <**title**>Contact</**title**> {% **load** widget\_tweaks %}  <!-- Contact Us Section -->  <**div** class="container">  <**div** class="row">  <!-- Section Titile -->  <**div** class="col-md-12">  <**h1** class="section-title">Love to Hear From You</**h1**>  </**div**>  </**div**>  <**div** class="row">  <**div** class="col-md-6 mt-3">  <**p**>It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using Content.</**p**>   <**div** class="find-widget">  Company: <**a** href="https://hostriver.ro">HostRiver</**a**>  </**div**>  <**div** class="find-widget">  Address: <**a** href="#">4435 Berkshire Circle Knoxville</**a**>  </**div**>  <**div** class="find-widget">  Phone: <**a** href="#">+ 879-890-9767</**a**>  </**div**>    <**div** class="find-widget">  Website: <**a** href="https://uny.ro">www.uny.ro</**a**>  </**div**>  <**div** class="find-widget">  Program: <**a** href="#">Mon to Sat: 09:30 AM - 10.30 PM</**a**>  </**div**>  <!--Google map-->  <**div** class="embed-responsive embed-responsive-16by9">  <**iframe** src="https://maps.google.com/maps?q=Barcelona&t=&z=13&ie=UTF8&iwloc=&output=embed"  style="border:0" class="embed-responsive-item" allowfullscreen></**iframe**>  </**div**>  </**div**>  <!-- contact form -->  <**div** class="col-md-6" >  <**form** class="form-horizontal" role="form" action="." method="post">  {% **csrf\_token** %}  {% **for** field in form %}  {% **if** field.errors %}  <**div** class="form-group has-error">  <**label** class="col-md-3 control-label" for="id\_{{ field.name }}">{{ field.label }}</**label**>  <**div** class="col-md-6">  {{ field|attr:"class:form-control" }}  <**span** class="help-block">  {% **for** error in field.errors %}{{ error }}{% **endfor** %}  </**span**>  </**div**>  </**div**>  {% **else** %}  <**div** class="form-group">  <**label** class="col-md-3 control-label" for="id\_{{ field.name }}">{{ field.label }}</**label**>  <**div** class="col-md-6">  {{ field|attr:"class:form-control" }}  {% **if** field.help\_text %}  <**p** class="help-block"><**small**>{{ field.help\_text }}</**small**></**p**>  {% **endif** %}  </**div**>  </**div**>  {% **endif** %}  {% **endfor** %}  <**div** class="form-group">  <**div** class="col-md-offset-3 col-md-6">  <**button** type="submit" class="btn btn-primary">Submit</**button**>  </**div**>  </**div**>  </**form**>  </**div**>  </**div**>  </**div**> {% **endblock** %} |

#### main.templates.about.html

|  |
| --- |
| {% extends "base.html" %} {% load static %} {% block content %} <div class="image-aboutus-banner"style="margin-top:70px">  <div class="container">  <div class="row">  <div class="col-md-12">  <img class="img-fluid w-100" src="{% static "logo.PNG" %}" alt="">  <h1 class="lg-text">About Us</h1>  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Sed ullamcorper morbi tincidunt ornare massa eget egestas purus viverra. Quis auctor elit sed vulputate mi sit. Non pulvinar neque laoreet suspendisse interdum consectetur libero id. Enim ut sem viverra aliquet eget. Purus gravida quis blandit turpis cursus in hac habitasse. Gravida neque convallis a cras semper auctor neque vitae tempus. Malesuada nunc vel risus commodo viverra. Lectus vestibulum mattis ullamcorper velit sed. Turpis in eu mi bibendum neque egestas congue quisque. Consequat id porta nibh venenatis cras sed felis eget velit. Vitae congue mauris rhoncus aenean. Lectus sit amet est placerat in egestas erat imperdiet sed. Sit amet tellus cras adipiscing enim eu turpis egestas pretium. Quis vel eros donec ac odio tempor. Nisi est sit amet facilisis magna etiam tempor orci eu.</p>   <p>Fermentum iaculis eu non diam phasellus vestibulum lorem sed. Sit amet consectetur adipiscing elit ut. Est ullamcorper eget nulla facilisi etiam dignissim diam. Mollis aliquam ut porttitor leo a. Libero nunc consequat interdum varius sit. Nibh mauris cursus mattis molestie a. Eget dolor morbi non arcu risus. Amet volutpat consequat mauris nunc congue nisi vitae suscipit tellus. Donec ultrices tincidunt arcu non sodales neque sodales ut. Id consectetur purus ut faucibus.</p>   <p>Enim ut sem viverra aliquet eget. Bibendum neque egestas congue quisque egestas diam in. Ullamcorper a lacus vestibulum sed arcu. Molestie at elementum eu facilisis sed odio morbi. Viverra vitae congue eu consequat ac felis. Adipiscing elit duis tristique sollicitudin. In metus vulputate eu scelerisque felis imperdiet proin. A lacus vestibulum sed arcu. Eleifend quam adipiscing vitae proin sagittis nisl rhoncus. In eu mi bibendum neque egestas. Felis donec et odio pellentesque diam volutpat commodo. Rhoncus dolor purus non enim. Pulvinar sapien et ligula ullamcorper malesuada proin. Tempus urna et pharetra pharetra massa. Enim nulla aliquet porttitor lacus luctus accumsan tortor.</p>   <p>Nibh tellus molestie nunc non blandit. Elementum pulvinar etiam non quam lacus. In mollis nunc sed id semper risus in. Arcu vitae elementum curabitur vitae nunc sed. Cursus eget nunc scelerisque viverra. Morbi quis commodo odio aenean sed adipiscing diam donec. Viverra aliquet eget sit amet tellus cras adipiscing enim. Diam phasellus vestibulum lorem sed risus ultricies. Etiam sit amet nisl purus in mollis nunc sed id. Ut eu sem integer vitae. Sit amet commodo nulla facilisi nullam vehicula ipsum a arcu. Tellus id interdum velit laoreet id donec. Suspendisse interdum consectetur libero id faucibus. Pellentesque habitant morbi tristique senectus. Tristique senectus et netus et malesuada fames ac. Et odio pellentesque diam volutpat commodo. Facilisis magna etiam tempor orci eu lobortis elementum nibh tellus. Eget nunc scelerisque viverra mauris in aliquam sem fringilla ut.</p>   <p>Fringilla urna porttitor rhoncus dolor purus non enim praesent elementum. Dui faucibus in ornare quam viverra orci sagittis. Adipiscing tristique risus nec feugiat in fermentum posuere urna nec. Interdum velit euismod in pellentesque massa placerat duis ultricies lacus. Imperdiet dui accumsan sit amet nulla facilisi morbi. Dui ut ornare lectus sit amet. Dolor sed viverra ipsum nunc. Sed odio morbi quis commodo odio aenean sed. Condimentum id venenatis a condimentum vitae sapien. Consectetur adipiscing elit ut aliquam purus sit amet luctus.</p>  </div>  <div class="col-md-12">  <h1 class="lg-text">The team </h1>  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Sed ullamcorper morbi tincidunt ornare massa eget egestas purus viverra. Quis auctor elit sed vulputate mi sit. Non pulvinar neque laoreet suspendisse interdum consectetur libero id. Enim ut sem viverra aliquet eget. Purus gravida quis blandit turpis cursus in hac habitasse. Gravida neque convallis a cras semper auctor neque vitae tempus. Malesuada nunc vel risus commodo viverra. Lectus vestibulum mattis ullamcorper velit sed. Turpis in eu mi bibendum neque egestas congue quisque. Consequat id porta nibh venenatis cras sed felis eget velit. Vitae congue mauris rhoncus aenean. Lectus sit amet est placerat in egestas erat imperdiet sed. Sit amet tellus cras adipiscing enim eu turpis egestas pretium. Quis vel eros donec ac odio tempor. Nisi est sit amet facilisis magna etiam tempor orci eu.</p>  </div>  </div> </div> </div>   {% endblock %} |

#### main.templates.search.html

|  |
| --- |
| {% **extends** "base.html" %}  {% **block** content %}  <**div** class="container">  <**div** class="row">  <**div** class="col-md-12">  <**h1**>Search results</**h1**>  <**p**>Items containing '{{query\_string}}'</**p**>  {% **if** query\_string == 'Null' %}<**p**>Please enter a term to query</**p**> {% **endif** %}  {{items.lenght}}   {% **for** i in items %}  <**div**>  <**h2** class="page-title">{{ i.title }}</**h2**></**a**>  <**div** class="item">  {{ i.description }}  </**div**>  </**div**>  {% **endfor** %}  </**div**>  </**div**> </**div**> {% **endblock** %} |

#### main.templates. detail.html

|  |
| --- |
| {% **extends** "base.html" %} {% **load** my\_tags %} {% **block** content %} <**div** class="container">  <**div** class="col">  {% **for** i in obj %}  <**div** class="row">  <**ul**>  {% **for** key, val in i|get\_fields %}  <**li**>{{ key }}: {{ val }}</**li**>  {% **endfor** %}  </**ul**>  </**div**>  {% **endfor** %}  </**div**> </**div**> {% **endblock** %} |

## Python files accounts

This section includes part of the tutorial of Wsvincent.com. 2019.

#### accounts.views.py

|  |
| --- |
| from django.contrib.auth.decorators import login\_required from django.shortcuts import render, redirect from datetime import datetime from django.contrib.auth.forms import UserCreationForm from accounts.forms import EditProfileForm, ProfileForm, RegisterForm from django.contrib.auth import update\_session\_auth\_hash from django.contrib.auth.models import User from django.shortcuts import get\_object\_or\_404 from .models import UserProfile from main.models import Order  def signup(request):  """Function collecting the data needed to send to the UI for Signup page.  """  if request.method == 'POST':  form = RegisterForm(request.POST)  if form.is\_valid():  new\_user = form.save()  return redirect("confirmation")  else:  form = RegisterForm()  return render(request, 'signup.html', { 'form': form })  def confirmation(request):  """Function collecting the data needed to send to the UI for Confirmation page.  """  context = {"about\_page": "active"}   return render(request, 'confirmation.html', context)  @login\_required def view\_profile(request):  """Function collecting the data needed to send to the UI for View\_profile page, require a user to be logged in.  """  orders = Order.objects.filter(user=request.user)  user = get\_object\_or\_404(User, username=request.user)  profile = get\_object\_or\_404(UserProfile, user=request.user)  return render(request, 'view\_profile.html', {"user":user, "profile": profile, "orders": orders})  @login\_required def edit\_profile(request):  """Function collecting the data needed to send to the UI for View\_profile page, require a user to be logged in.  """  if request.method == 'POST':  form = EditProfileForm(request.POST, instance=request.user)  profile\_form = ProfileForm(request.POST, request.FILES, instance=request.user.userprofile)    if form.is\_valid() and profile\_form.is\_valid():  user\_form = form.save()  custom\_form = profile\_form.save(False)  custom\_form.user = user\_form  custom\_form.save()  return redirect('/')  else:  profile\_form = ProfileForm(instance=request.user.userprofile)  form = EditProfileForm(instance=request.user)  args = {}  args['form'] = form  args['profile\_form'] = profile\_form  return render(request, 'edit\_profile.html', args) |

#### accounts.urls.py

|  |
| --- |
| from django.urls import path, include from django.conf.urls import url from . import views  urlpatterns = [  path('accounts/', include('django.contrib.auth.urls')),  path('signup/', views.signup, name='signup'),  path('view\_profile/', views.view\_profile, name='view\_profile'),  path('edit\_profile/', views.edit\_profile, name ='edit\_profile'),  path('confirmation/', views.confirmation, name='confirmation') ] |

#### accounts.models.py

|  |
| --- |
| from django.db import models from django.contrib.auth.models import User from django.db.models.signals import post\_save   class UserProfile(models.Model):  """Model declaring the fields for userprofile object.  """  user = models.OneToOneField(User, on\_delete=models.CASCADE)  address = models.CharField(max\_length=100, default='')  city = models.CharField(max\_length=100, default='')  phone\_number = models.IntegerField(default=0)  image = models.ImageField(upload\_to='profile\_image' , blank=True)  first\_name = models.CharField(max\_length=100, default='')  last\_name = models.CharField(max\_length=100, default='')   def \_\_str\_\_(self):  return self.user.username  def create\_profile(sender,\*\*kwargs ):  """Function creating automatically the UserProfile when a new User is created.   Arg: sender that send the signal in this case current user  """  if kwargs['created']:  user\_profile=UserProfile.objects.create(user=kwargs['instance'])  post\_save.connect(create\_profile,sender=User) |

#### accounts.forms.py

|  |
| --- |
| from django import forms from django.contrib.auth.models import User from django.contrib.auth.forms import UserCreationForm, UserChangeForm from django.forms import ModelForm from accounts.models import UserProfile   class EditProfileForm(ModelForm):  """Class extenfing ModelForm to include the User object needed.  """  class Meta:  model = User  fields = (  'email',  'first\_name',  'last\_name'  )   class ProfileForm(ModelForm):  """Class extenfing ModelForm to include the UserProfile object needed.  """  class Meta:  model = UserProfile  fields = ('city', 'address', 'phone\_number', 'image')    class RegisterForm(UserCreationForm):  """Class extenfing UserCreationForm to include the fields needed.  """  username = forms.CharField(label="Your Username")  password1 = forms.CharField(label="Your Password")  password2 = forms.CharField(label="Repeat Your Password")  email = forms.EmailField(label = "Email Address")  first\_name = forms.CharField(label = "Name")  last\_name = forms.CharField(label = "Surname")    class Meta:  model = User  fields = ("first\_name", "last\_name", "email", "username", "password1", "password2", )    def save(self, commit=True):  """When commit is done from the UI, save the data from the form, and include field to User that are not standard.  """  user = super(RegisterForm, self).save(commit=False)  user.email = self.cleaned\_data["email"]  user.first\_name = self.cleaned\_data["first\_name"]  user.last\_name = self.cleaned\_data["last\_name"]    if commit:  user.save()  return user |

## Account templates

#### accounts.templates.confirmation.html

|  |
| --- |
| {% **extends** "base.html" %} {% **block** content %} <**div** class="image-aboutus-banner"style="margin-top:70px">  <**div** class="container">  <**div** class="row">  <**div** class="col-md-12">  <**h1** class="lg-text">Thanks!</**h1**>  <**p** class="image-aboutus-para">You are now register, please follow the link to Log In  </**div**>  </**div**> </**div**> </**div**>   {% **endblock** %} |

#### accounts.templates.edit\_profile.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **load** crispy\_forms\_tags %} <**title**>Profile</**title**>  {% **block** content %} <**div** class="container">  <**div** class="row mt-3"></**div**>  <**div** class="row justify-content-center">  <**div** class="col-md-6">  <**div** class="card">  <**div** class="card-header">  Modify your Profile  </**div**>  <**div** class="card-body">  <**form** method="post" enctype="multipart/form-data">  {% **csrf\_token** %}  {{ form|crispy }}  {{ profile\_form|crispy }}  <**button** class="btn btn-primary float-right" type="submit">Submit</**button**>  </**form**>  </**div**>  </**div**>  </**div**>  </**div**> </**div**>  {% **endblock** %} |

#### accounts.templates.signup.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **load** crispy\_forms\_tags %} {% **block** title %}Sign Up{% **endblock** %}  {% **block** content %} <**div** class="container">  <**div** class="row mt-3"></**div**>  <**div** class="col-md-6 align-self-center">  <**div** class="card">  <**div** class="card-header">  <**h2**>Sign up</**h2**>  </**div**>  <**div** class="card-body">  <**form** method="post">  {% **csrf\_token** %}  {{ form|crispy }}  <**button** type="submit" class="btn btn-primary">Sign Up</**button**>  </**form**>  </**div**>  </**div**>  </**div**> </**div**>  {% **endblock** %} |

#### accounts.templates.view\_profile.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **block** content %} {% **if** request.user.is\_authenticated %} <**title**>Profile</**title**> <**div** class="container">  <**div** class="row mt-3"></**div**>  <**div** class="row">  <**div** class="col-md-3">  <**div** class="card">  <**div** class="card-header">  Your Profile  </**div**>  <**div** class="card-body">  <**div** class="profile-name">  {{user.first\_name}} {{user.last\_name}}  </**div**>  <**div** class="profile-address">  {{profile.address}}  </**div**>  <**div** class="profile-city">  {{profile.city}}  </**div**>  <**div** class="profile-email">  {{user.email}}  </**div**>  <**div** class="profile-phone">  {{profile.phone\_number}}  </**div**>  <**a** href="{% **url** 'edit\_profile' %}" class="btn btn-dark float-right">Edit</**a**>  </**div**>  </**div**>  </**div**>  <**div** class="col-md-7">  <**div** class="card">  {% **if** request.user.is\_staff %}  <**div** class="card-header">  For employees only!  </**div**>  <**div** class="card-body">  <**a** href="{% **url** 'all\_orders' %}" class="btn btn-primary" role="button" >Orders</**a**>  <**a** href="{% **url** 'all\_items' %}" class="btn btn-secondary" role="button" >Items</**a**>  <**a** href="{% **url** 'all\_users' %}" class="btn btn-dark" role="button" >Users</**a**>  </**div**>  {% **else** %}   <**div** class="card-header">  List of your orders:  </**div**>  <**div** class="card-body">  {% **for** order in orders %}   <**h5**>Order ID {{order.id}} - {{order.date\_order}}</**h5**>  <**p**></**p**>  {% **for** i in order.items.all %}  <**p**>art. {{ i.id }} - {{ i.title }}</**p**>  <**p**>{{ i.description }}</**p**>  {% **endfor** %}  {% **endfor** %}  </**div**>  {% **endif** %}  </**div**>  </**div**>  </**div**> </**div**> {% **endif** %}  {% **endblock** %} |

## Account registration templates

#### accounts.templates.registration.login.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **load** crispy\_forms\_tags %} {% **block** title %}Login{% **endblock** %}  {% **block** content %}  <**div** class="container">  <**div** class="row mt-3"></**div**>  <**div** class="d-flex justify-content-center">  <**div** class="col-3 align-self-center" >  <**h2**>Login</**h2**>  <**form** method="post">  {% **csrf\_token** %}  {{ form|crispy }}  <**button** type="button submit" class="btn btn-info">Login</**button**>  </**form**>  <**p**><**a** href="{% **url** 'password\_reset' %}">reset password</**a**></**p**>  </**div**>  </**div**> </**div**>  {% **endblock** %} |

#### accounts.templates.registration.password\_rest\_complete.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **block** title %}Password reset complete{% **endblock** %}  {% **block** content %} <**div** class="container">  <**div** class="row">  <**div** class="col"> <**h1**>Password reset complete</**h1**> <**p**>Your new password has been set. You can log in now on the <**a** href="{% **url** 'login' %}">log in page</**a**>.</**p**> </**div**> </**div**> </**div**> {% **endblock** %} |

#### accounts.templates.registration.password\_rest\_confirm.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **block** title %}Enter new password{% **endblock** %}  {% **block** content %} <**div** class="container">  <**div** class="row">  <**div** class="col">  <**h1**>Set a new password!</**h1**> <**form** method="POST">  {% **csrf\_token** %}  {{ form.as\_p }}  <**input** type="submit" value="Change my password"> </**form**> </**div**> </**div**> </**div**> {% **endblock** %} |

#### accounts.templates.registration.password\_rest\_done.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **block** title %}Email Sent{% **endblock** %}  {% **block** content %} <**div** class="container">  <**div** class="row">  <**div** class="col">  <**h1**>Check your inbox.</**h1**>  <**p**>We've emailed you instructions for setting your password. You should receive the email shortly!</**p**> </**div**> </**div**> </**div**> {% **endblock** %} |

#### accounts.templates.registration.password\_rest\_form.html

|  |
| --- |
| {% **extends** 'base.html' %}  {% **block** title %}Forgot Your Password?{% **endblock** %}  {% **block** content %} <**div** class="container">  <**div** class="row">  <**div** class="col">  <**h1**>Forgot your password?</**h1**>  <**p**>Enter your email address below, and we'll email instructions for setting a new one.</**p**>   <**form** method="POST">  {% **csrf\_token** %}  {{ form.as\_p }}  <**input** class="btn btn-primary" type="submit" value="Send me instructions!">  </**form**> </**div**> </**div**> </**div**>  {% **endblock** %} |