**Django**

* Python framework to develop a web application
* Uses MVT (Model View Template) architecture

pip install django

django-admin startproject broadway .

python manage.py runserver

**Web Application**

* **Presentation Layer(UI/UX)**

***HTML*** – to define contents of a page

***CSS*** – to apply style to a page

***Bootstrap***– CSS library to make responsive pages

***JavaScript*** – to add user events to a page (to make a page interactive)

***JQuery*** – JavaScript library

***AJAX*** – Technique to load data to a particular component without reloading full page

***React JS, Vue JS, React Native, Flutter***

* **Business layer**
* **Database**

**HTML**

* Hypertext Markup Language
* Used to create web pages
* defines contents of a web page
* **tag** based language (<html></html>)
* tag has its properties called **attributes**
* it has tree like document structure called DOM (Document Object Model)

**Tags**

1. Heading Tags (h1-h6)
2. Paragraph Tag(p)
3. Formatting tags (b, I, u, strong, strike, em, pre, sub, sup)
4. Lists
   1. Unordered List (<ul><li></li></ul>)
   2. Ordered List (<ol><li></li></ol>)
   3. Definition List(<dl><dt></dt><dd></dd></dl>)
5. Multimedia
   1. Image (<img src=”” alt=”” />)

(<figure>

<img src=”” alt=”” />

<figcaption>Text</figcaption>

</figure>)

* 1. Audio (<audio controls><source src=”” type=”” /></audio>)
  2. Video (<video controls><source src=”” type=”” /></video>)

1. Hyperlink (anchor tag (<a href=””>Click</a>))
   1. Relative URL
   2. Absolute URL
   3. In page Link
2. Table in HTML

<table>

<tr>

<td></td>/<th></th>

</tr>

</table>

1. Division and Span (<div></div> <span></span>)
2. HTML5 Semantic elements

header, nav, section, article, aside, footer

1. Iframe
2. Form in HTML
   1. <form method=”” action=”” enctype=””></form>
   2. <label>Text</label>
   3. <input type=”” name=”” value=”” />

type=text/date/number/password/radio/checkbox/range/file/email/url/submit/button/reset

* 1. <select name=””>

<option value=””></option>

</select>

* 1. <textarea name=””></textarea>
  2. <button type=” submit/button/reset”></button>

**CSS - Selectors**

|  |  |
| --- | --- |
| Tag Selector | **p{} h1{} ul{}** |
| Class Selector | **.class\_name {}** |
| Id Selector | **#id\_name {}** |
| Descendant selector | **#id\_name p {} nav ul li{}** |
| Child Selector | **ul > li {} .div\_class > p {}** |
| Pseudo Element  Selector | **h1::first-letter{} p::first-line{}**  **h1::after{} h1::before{}** |
| Pseudo Class Selector | **a:hover{} a:active{} a:visited{} a:link{}** |
| Sibling selector | **div ~ p {}** |
| Adjacent sibling selector | **div + p {}** |
| nth child selector | **div:nth-child(3)** |
| Universal selector | **\*** |
| Multiple Element Sel | **P, h1, #id\_name {}** |

**Text Properties**

color, font-size, font-family, text-align, text-indent,

font-weight, text-decoration, font-style, text-shadow

**Box Properties**

height, width, background-color, background, margin, padding, border, border-radius

**Floating & Positioning**

* float: left/right
* clear: left/right/both
* position: absolute/fixed/relative/sticky/static
* display: none/block/inline/grid/flex

**Animation & Styles**

* transform: rotate/skew/scale
* animation

animation-name:

animation-duration:

* transition

**Database**

Stores data in the form of table. Provides different tools to manage data.

**CRUD Operation**

* CREATE DATABASE db\_name;
* SHOW DATABASES;
* USE db\_name;
* CREATE TABLE users(id int PRIMARY KEY AUTO\_INCREMENT, name varchar(100), dob Date, phone varchar(20), email varchar(100), username varchar(20) UNIQUE KEY, password varchar(20), type ENUM('admin', 'staff', 'customer'), status boolean);
* CREATE TABLE products (id int PRIMARY KEY AUTO\_INCREMENT,

category\_id int, name varchar(255),

details mediumtext, price double,

stock int,

FOREIGN KEY (category\_id) REFERENCES categories(id))

* CREATE TABLE orders (id int PRIMARY KEY AUTO\_INCREMENT,

user\_id int, product\_id int, qty int,

ordered\_date date, delivery\_address varchar(255),

FOREIGN KEY(user\_id) REFERENCES users(id),

FOREIGN KEY(product\_id) REFERENCES products(id))

* INSERT INTO users (name,dob,phone,email,username,password,type,status)

VALUES ('ram kumar', '2000-01-01', '9812345678', 'ram@email.com', 'ram123', 'ram123', 'admin', 1)

* INSERT INTO users

VALUES ('', 'ram kumar', '2000-01-01', '9812345678', 'ram@email.com', 'ram123', 'ram123', 'admin', 1)

* UPDATE users SET name='ram prasad', dob='2001-08-10' WHERE id=1
* DELETE FROM users WHERE id=3
* SELECT \* FROM users
* SELECT name, type FROM users
* SELECT \* FROM `users` WHERE name LIKE 'r%' AND dob < '2002-01-01';
* SELECT \* FROM products

JOIN categories ON categories.id=products.category\_id

* SELECT products.name, categories.title, products.price FROM products

JOIN categories ON categories.id=products.category\_id

WHERE categories.title = 'Kitchen Ware'

* TRUNCATE TABLE product
* DROP TABLE product
* ALTER : see yourself

**Web application**

**News Portal**

* **user**
  + name
  + email
  + username
  + password
  + type
  + status
* **category**
  + title
  + image
  + status
* **news/article**
  + category\_id
  + title
  + date
  + details
  + author
  + image
  + status
  + user\_id
* **advertisement**
  + company
  + product
  + title
  + image
  + details
  + category
  + status
  + user\_id
* **comment**
  + user\_id
  + news\_id
  + datetime
  + status
* **about**
  + title
  + details
  + status

pip install django

django-admin startproject broadway .

python manage.py runserver

django-admin startapp module\_name

python manage.py makemigrations

python manage.py migrate

python manage.py createsuperuser

pip install pillow

pip install mysqlclient

1. Install django

pip install django

1. Create django project

django-admin startproject project\_name .

1. Create all modules needed for the project

django-admin startapp app\_name

**Database Migrations**

1. Define Model class (required attributes with required datatypes)

**class Category(models.Model):  
 title = models.CharField(max\_length=255)  
 image = models.ImageField(max\_length=255, upload\_to='categories')  
 details = models.TextField()  
 status = models.BooleanField(default=True)**

**class News(models.Model):  
 title = models.CharField(max\_length=255)  
 date = models.DateTimeField()  
 details = models.TextField()  
 author = models.CharField(max\_length=255)  
 image = models.ImageField(upload\_to='news')  
 status = models.BooleanField(default=True)  
 category\_id = models.ForeignKey(Category, on\_delete=models.CASCADE)**

1. After completing all required model classes add your module to **installed apps** list in **settings.py**

INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
  
 'about.apps.AboutConfig',  
 'categories',  
 'news',  
 'comment'  
]

Then create migration files

python manage.py makemigrations

1. Migrate all the table to database using migration files created above

python manage.py migrate

**Django admin Panel**

1. We can access django admin panel at site\_url/admin
2. To register your model to django-admin go to admin.py of relevant module/app and add the highlighted line:

categories/admin.py

from django.contrib import admin  
from categories.models import Category  
  
admin.site.register(Category)

news/admin.py

from django.contrib import admin  
from news.models import News  
  
admin.site.register(News)

**To change admin panel displaying data :**

categories/admin.py after defining model admin class

class CategoryAdmin(admin.ModelAdmin):  
 list\_display = ('title', 'image', 'details', 'status')  
  
  
admin.site.register(Category, CategoryAdmin)

news/admin.py after defining model admin class

class NewsAdmin(admin.ModelAdmin):  
 list\_display = ('title', 'date', 'author', 'image',  
 'category', 'user', 'status')  
  
  
admin.site.register(News, NewsAdmin)

1. Setting for Static, Media files and Templates

Create directories/folders named static, media and templates inside project folder then add this to setting.py file.

**setting.py**

|  |  |
| --- | --- |
| STATIC\_URL = 'static/' STATICFILES\_DIRS = [os.path.join(BASE\_DIR, 'static')]   MEDIA\_URL = 'media/' MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media/') | TEMPLATES = [  {  'BACKEND': 'django.template.backends.django.DjangoTemplates',  'DIRS': [os.path.join(BASE\_DIR, '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',  ],  },  }, ] |

**Routing**

1. Routing

**project/urls.py**

from django.conf.urls.static import static  
from django.contrib import admin  
from django.urls import path, include  
from broadway import settings  
  
urlpatterns = [  
 path('admin/', admin.site.urls),

path('', views.show\_home),

path('about/', include('about.urls')),  
 path('news/', include('news.urls')),  
 path('comment/', include('comment.urls')),  
]

**about/urls.py**

from django.urls import path  
from about import views  
  
urlpatterns = [  
 *# http://127.0.0.1:8000/about/* path('', views.show\_about),  
  
 *# http://127.0.0.1:8000/about/contacts* path('contacts', views.show\_contacts)  
]

**news/urls.py**

from django.urls import path  
from news import views  
  
urlpatterns = [  
 *# http://127.0.0.1:8000/news/* path('', views.show\_news),  
  
 *# http://127.0.0.1:8000/news/category/3* path('category/<cid>',views.show\_news\_by\_category),  
  
 *# http://127.0.0.1:8000/news/single/2* path('single/<nid>', views.show\_single\_news),  
  
 *# http://127.0.0.1:8000/news/searched/election*]

1. Render a template and show data

**about/views.py**

from django.http import HttpResponse  
from django.shortcuts import render  
from categories.models import Category  
from news.models import News

def show\_home(request):  
 categories = Category.objects.all()  
 business\_news = News.objects.filter(category\_id=2).order\_by('-id')[:4]  
 return render(request, 'index.html', {'categories': categories,  
 'business\_news': business\_news})

**index.html**

{% for fn in flash\_news %}  
<div class="position-relative overflow-hidden" style="height: 500px;">  
 <img class="img-fluid h-100" src="media/{{ fn.image }}" style="object-fit: cover;">  
 <div class="overlay">  
 <div class="mb-2">  
 <a class="badge badge-primary text-uppercase font-weight-semi-bold p-2 mr-2"  
 href="">{{ fn.category.title }}</a>  
 <a class="text-white" href="">{{ fn.date }}</a>  
 </div>  
 <a class="h2 m-0 text-white text-uppercase font-weight-bold" href="">{{ fn.title }}</a>  
 </div>  
</div>  
{% endfor %}