# Views and URLs

Once you've created the project by following the instructions, go to the root project folder ("devsite") and open the urls.py file.

You'll find the following initial code:

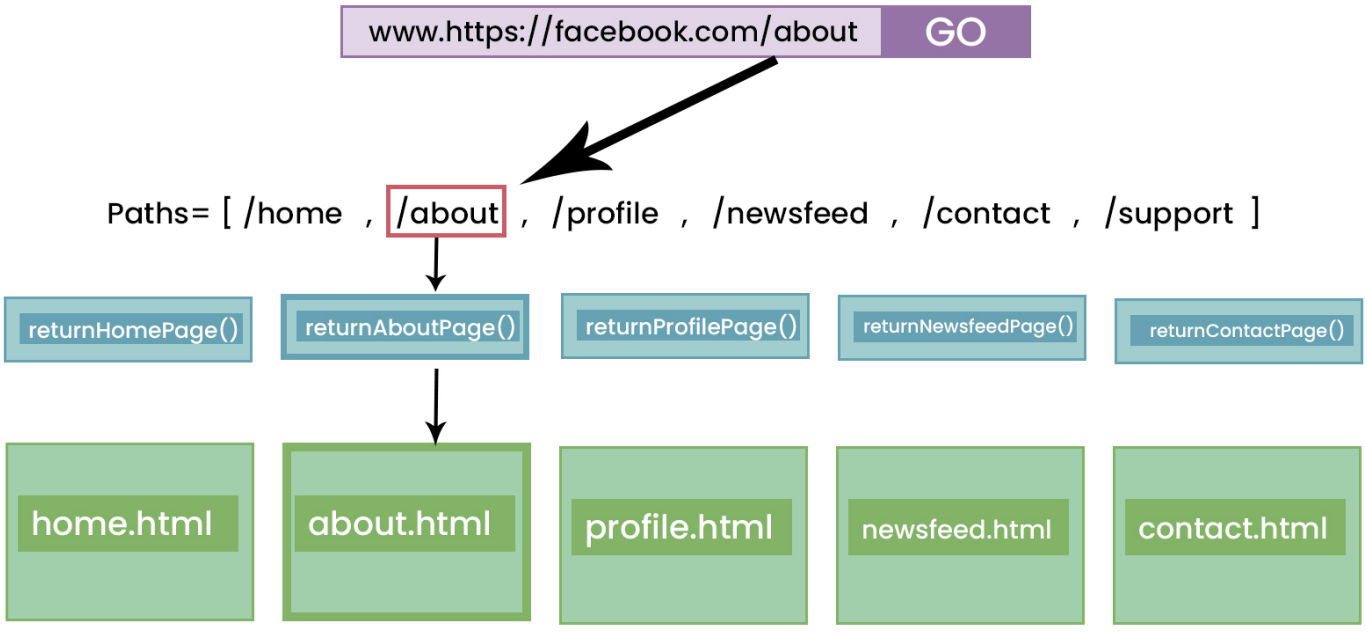
```
from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls)
]
```

We can actually exite a response for the site by simply adding a "View" to the same file.

In django, a view is really a response to a 'request'. This is overly simplified, but basically, we only have to provide a reply to a 'GET' request.

In the following image, we can think of the facebook site as a series of paths (or slugs) that provide a way to select what kind of reply has been requested from the website by accessing that particular address in the site:



for each specific 'slug' we have a corresponding reply, i. e., for /about, we retreive the function *returnAboutPage()* which is in charge or generating the about.html from a template and possible some variables.

So, let's modify the urls.py file and lets add the code:

```
from django.contrib import admin
from django.urls import path

from django.http import HttpResponse

def projects(request):
    return HttpResponse('These are our current projects')

urlpatterns = [
    path('admin/', admin.site.urls),
    path('projects/', projects, name="projects")
]
```

Now, run your server:

```
python manage.py runserver
```

and in your browser go to the page with the slug "projects":

```
127.0.0.1:8000/projects
```

you should see the text "These are our current projects" as a reply on the page.

## Parameterized Views

Views also accept parameters with their slugs. Lets change the urls.py in the project's root folder by adding another function:

```
from django.contrib import admin
from django.urls import path

from django.http import HttpResponse

def projects(request):
    return HttpResponse('These are our current projects')
```

```
    def project(request, pk):
        return HttpRequest(f'This is Project {pk}')

    urlpatterns = [
        path('admin/', admin.site.urls),
        path('projects/', projects, name="projects")
        path('project/<slug:pk>/, project, name="project")
    ]
```

Now, run your server again if you stopped it:

```
    python manage.py runserver
```

and in your browser go to the page with the slug "projects/<anything>":

```
    127.0.0.1:8000/project/1
    127.0.0.1:8000/project/39
    127.0.0.1:8000/project/Pluto
    127.0.0.1:8000/project/Hades
```

you should see the text "This is Project <anything>" as a reply on the page.

Since we don't really want to pollute the project's root folder with the view handling for each django application, we should create the first django application for the project:

```
    python manage.py startapp projects .
```

and let the application handle any view that may be constructed:

Open up the views.py file in the projects folder. The initial state should look like this:

```
    from django.shortcuts import render

    # Create your views here.
```

Lets add the code we used in the project's root folder urls.py file. In the views.py in the projects folder:

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.

def projects(request):
    return HttpResponse('These are our current projects')

def project(request, pk):
    return HttpRequest(f'This is Project {pk}')
```

and create the file urls.py in the same projects folder and move the routes for the projects/views.py:

```
from django.urls import path
from . import views

urlpatterns = [
    path('projects/', views.projects, name="projects")
    path('project/<slug:pk>/, views.project, name="project")
]
```

and finally, we need to modify the devsite/urls.py in orde to include the projects/urls.py

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('projects.urls')),
]
```

In the statement:

```
path('', include('projects.urls'))
```

the '' inidicates the url root path or url domain path, and the include statement includes the file on the particular folder, in this case, "/projects/urls.py" file.