

# TALLER:

## *“Proyectos Web en Python”*

django



Jorge Díaz M.  
Jorge.diazma@sansano.usm.cl  
Segundo Semestre 2019

# Sesión 2

## Motivación

Patrón MVC y Django



# Patrón MVC

(Modelo – Vista – Controlador )



# Sesión 2.1

¿Qué es el Patrón MVC?









## MVC

- Es un patrón de la arquitectura de software muy conocido y popular, utilizado y probado en el mundo de aplicaciones web.
- Es uno de los más comunes tipos de implementación Web.
- Este se puede diferenciar en 3 secciones: Modelo – Vista – Controlador.



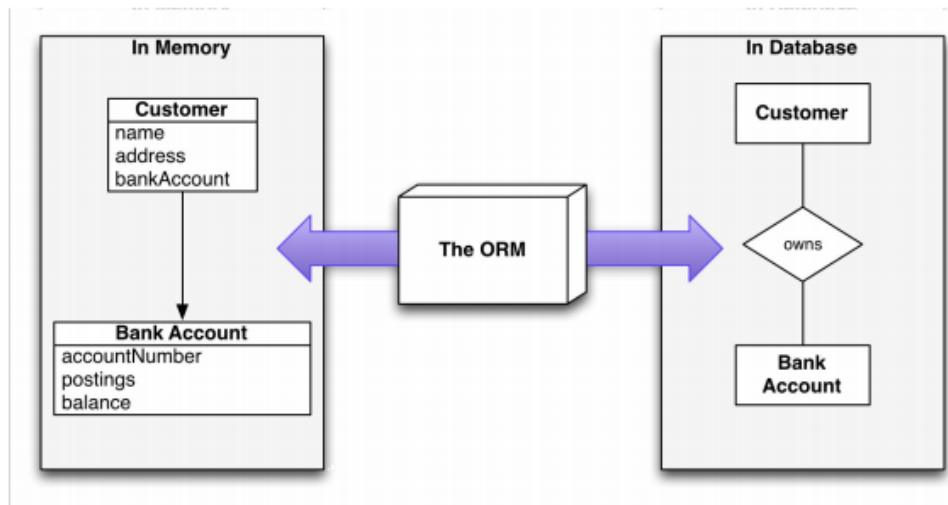
# Modelo

CREATE READ UPDATE DELETE

---

C R U D





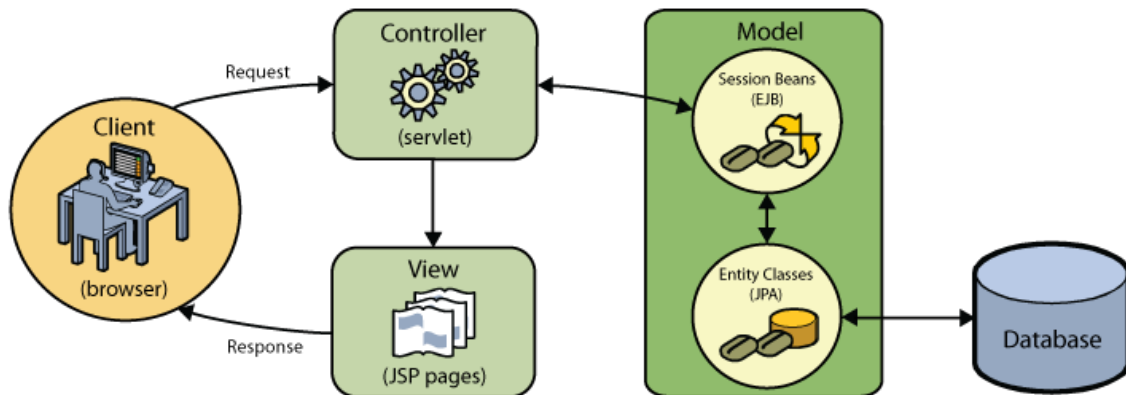
## Vista

- Representa la interfaz gráfica de la aplicación Web.
- Puede o no utilizar datos proporcionados desde el modelo.
- Típicamente HTML, CSS, JavaScript, etc.





# Controlador



- Define el comportamiento de la aplicación Web.
- Procesa las peticiones HTTP (GET, POST, PUT, DELETE).
- Es el ente intermedio entre Modelo y Vista.



# Sesión 2.2

¿Qué es Django?

**django**



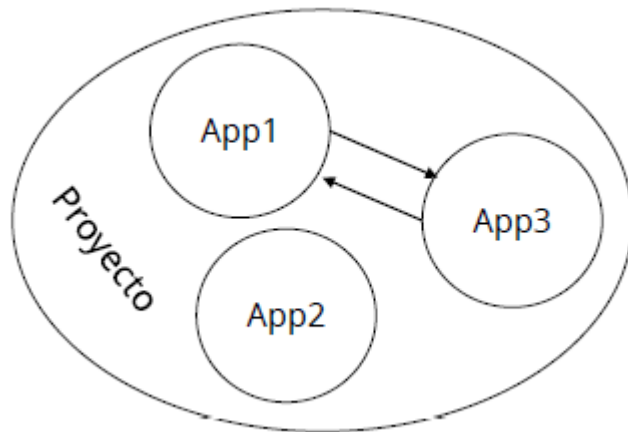
## Django

- Es un framework de desarrollo web de código abierto que se lanzó el 2005.
- Respeta (o se basa) en el patrón MVC.
- Es compatible con diversos framework de Python, como lo es REST para el desarrollo en base a microservicios u otras librerías de apoyo.



## ¿Cómo está compuesto un proyecto en Django?

- Está compuesto por un conjunto de módulos o Apps.
- Cada módulo tiene su vista, modelo y controlador
- Estas apps pueden o no interactuar entre sí.





## ¿Cómo creamos un proyecto?

- Una vez que activamos el entorno virtual, ejecutamos el siguiente comando:

**py -m django startproject nombreproyecto (o)**

**python3 -m django startproject nombreproyecto**

Con el comando anterior, tendremos nuestra carpeta del proyecto creado. La pregunta ahora es: ¿Cómo está organizado un proyecto en Django?



# Sesión 2.3

Creando un proyecto  
en Django

**django**

# Creamos el Entorno Virtual e instalamos Django

Recuerden que el entorno virtual es muy importante para el desarrollo de su proyecto.

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.17763.678]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\jdiaz>cd Desktop

C:\Users\jdiaz\Desktop>py -m venv env

C:\Users\jdiaz\Desktop>pip3 install django
Requirement already satisfied: django in c:\users\jdiaz\appdata\local\programs\python\python36\lib\site-packages (2.2)
Requirement already satisfied: pytz in c:\users\jdiaz\appdata\local\programs\python\python36\lib\site-packages (from django) (2019.1)
Requirement already satisfied: sqlparse in c:\users\jdiaz\appdata\local\programs\python\python36\lib\site-packages (from django) (0.3.0)
You are using pip version 18.1, however version 19.2.2 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Users\jdiaz\Desktop>
```

## Creamos nuestro Proyecto

Para esta ocasión, el nombre de nuestro proyecto será “Taller”.

```
Simbolo del sistema
Microsoft Windows [Versión 10.0.17763.678]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\jdiaz>cd Desktop

C:\Users\jdiaz\Desktop>py -m venv env

C:\Users\jdiaz\Desktop>pip3 install django
Requirement already satisfied: django in c:\users\jdiaz\appdata\local\programs\python\python36\lib\site-packages (2.2)
Requirement already satisfied: pytz in c:\users\jdiaz\appdata\local\programs\python\python36\lib\site-packages (from django) (2019.1)
Requirement already satisfied: sqlparse in c:\users\jdiaz\appdata\local\programs\python\python36\lib\site-packages (from django) (0.3.0)
You are using pip version 18.1, however version 19.2.2 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Users\jdiaz\Desktop>py -m django startproject Taller

C:\Users\jdiaz\Desktop>.
```

“

*¿Cómo quedan las  
carpetas?*



./taller



taller



manage.py

./taller/taller



\_\_init\_\_.py



settings.py



urls.py



wsgi.py

./taller

./taller/taller



taller



manage.py



Configuración  
Global del  
Proyecto



\_\_init\_\_.py



settings.py



urls.py



wsgi.py

./taller

./taller/taller



taller



manage.py

Direcciones  
generales del  
Proyecto



\_\_init\_\_.py




settings.py



urls.py



wsgi.py



### **<proyecto>/manage.py**

Es similar a *django* pero se encuentra configurado para definir algunas configuraciones en relación al directorio del proyecto donde vive

### **<proyecto>/<proyecto>/settings.py**

Es el corazón del proyecto, contiene todas las configuraciones principales

### **<proyecto>/<proyecto>/urls.py**

Es el ruteador del proyecto, indica que debe cargarse al entrar a determinada url en el navegador



## Comandos importantes en Django

### # Ejecutar el servidor para Django

- `py manage.py runserver <host:puerto>`

### # Crear un superusuario

- `py manage.py createsuperuser`

### # Buscar cambios sin aplicar al modelo de la Base de Datos

- `py manage.py makemigrations <app>`

### # Aplicar cambios a la Base de Datos

- `py manage.py migrate`



## Algunas configuraciones importantes de **settings.py**

<b>ALLOWED_HOSTS</b>	Servidores autorizados para funcionar, * = comodín, todos los servidores
<b>INSTALLED_APPS</b>	Aplicaciones que funcionan bajo esta configuración de proyecto, hay algunas definidas por defecto que se encargan del sistema de usuarios (django.contrib.auth) y el administrador (django.contrib.admin)
<b>ROOT_URLCONF</b>	Paquete de Python que controla el ruteo principal de esta configuración
<b>TEMPLATES</b>	Configuración del sistema de templates (vistas), especifica como se cargan y con que opciones
<b>DATABASES</b>	Configuración de la conexión a la base de datos, por defecto SQLite3, se puede cambiar por MySQL u otras.
<b>LANGUAGE_CODE</b>	Idioma para el que se traducirán los textos que cuenten con internacionalización
<b>TIME_ZONE</b>	Zona horaria del proyecto para cuando se trabaja con horas



## Antes de tener nuestro primer paso del proyecto

Las rutas del proyecto están en el archivo `urls.py`

Ruteador por defecto **`urls.py`**

```
from django.conf.urls import url
from django.contrib import admin
```

```
urlpatterns = [
    url(r'^admin/', admin.site.urls),
]
```





## Antes de tener nuestro primer paso del proyecto

Es necesario la Base de Datos exista y tenga los modelos básicos del proyecto.

Primero, veremos cuáles apps ya están incluidas en el proyecto por defecto.

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
]
```

*En settings.py*





## Casi listos

Ahora que sabemos las apps,  
utilizaremos el comando para  
incluir las datos  
predeterminados en la Base  
de Datos

***py manage.py migrate***

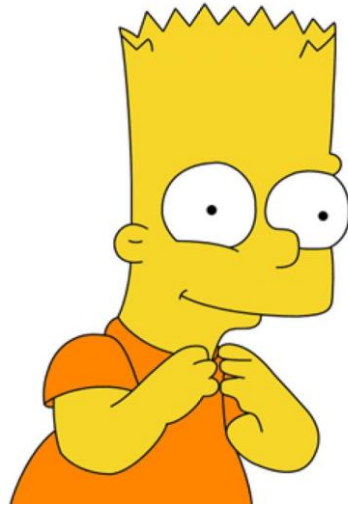


## Casi listos

Ahora que sabemos las apps, utilizaremos el comando para incluir las datos predeterminados en la Base de Datos

### *py manage.py migrate*

```
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying sessions.0001_initial... OK
```



**¡Ahora, a correr  
nuestro proyecto!**



**py manage.py runserver**

Ahora, en su navegador, ingrese la dirección  
<http://127.0.0.1:8000>



The install worked successfully! Congratulations!

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.



**Django Documentation**

Topics, references, & how-to's



**Tutorial: A Polling App**

Get started with Django



**Django Community**

Connect, get help, or contribute



## Listo

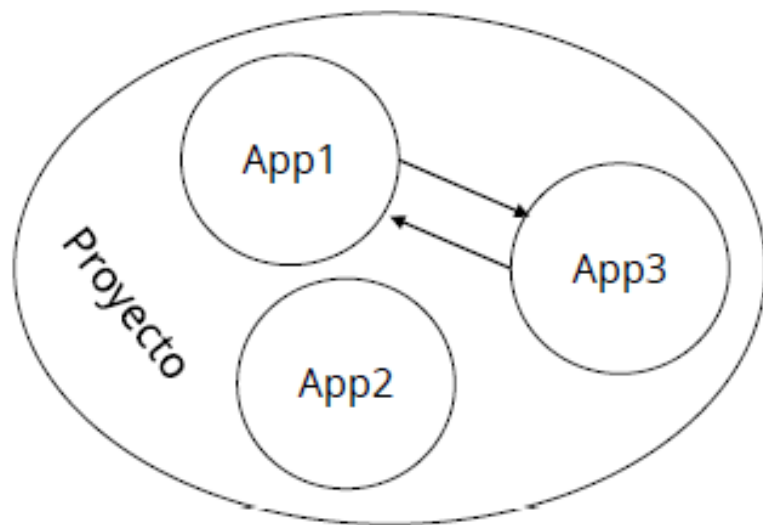
Ya tenemos nuestro proyecto funcionando. Ahora, tenemos que agregar las apps necesarias para hacer funcionar nuestro proyecto y agregar los modelos de datos que son necesarios.

Por ahora, agregaremos una app básica para tener una vista diferente a la por defecto.

# Sesión 2.4

## Apps en Django

- ❑ Recordar que Django funciona con apps







## ¿Qué es una app en django?

- Es una separación física del código (carpetas), que nos permite ordenar nuestro proyecto de forma inteligente.
- También, nos permite hacer un mejor seguimiento de los procedimientos que tenemos en nuestro proyecto.



En otras palabras, es para organizar de mejor manera el código



## Agregando una App

■ Para agregar una app utilizamos el siguiente comando

```
py manage.py startapp <nombre_app>
```



## Agregando una App

■ Para agregar una app utilizamos el siguiente comando

`py manage.py startapp <nombre_app>`



**Este comando, nos creará una carpeta llamada <nombre\_app> con algunos archivos, entre ellos `models.py` y `views.py` (importantes)**



## Agregando una App

Para agregar esta carpeta sea reconocida por nuestro proyecto, tenemos que acordarnos de algo:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
]
```



En settings.py



## Agregando una App

Para agregar esta carpeta sea reconocida por nuestro proyecto, tenemos que acordarnos de algo:

Para agregar la app debemos agregarla a la lista

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
]
```

*Ejemplo:  
Si la app creada se llama prueba,  
deberíamos agregar la línea:*

*'prueba.apps.PruebaConfig',*





## Ocupando la nueva app

- Con lo anterior, nuestro proyecto ya reconoce nuestra nueva app, pero, ¿cómo le damos uso?
- Descarguen el archivo zip de Moodle llamado “sesión 2” de Moodle, y copie el archivo urls.py en la carpeta de su nueva app.

```
File Edit Selection Find View Goto Tools Project Preferences Help
urls.py
1 from django.conf.urls import url, include
2
3
4 urlpatterns = [
5     url(r'^$', inicio),
6
7
8 ]
9
10 |
```



## Ocupando la nueva app

Agreguen la siguiente línea al archivo en la fila 2:

*from <nombre\_app>.views import \**

En este ejemplo, la app se llama proyectos

```
File Edit Selection Find View Goto Tools Project Preferences Help
urls.py
1 from django.conf.urls import url, include
2 from proyectos.views import *
3
4 urlpatterns = [
5     url(r'^$', inicio),
6
7
8 ]
9
```



## Ocupando la nueva app

Ahora vamos al archivo views.py

```
File Edit Selection Find View Goto Tools Project Preferences Help
views.py x
1 from django.shortcuts import render
2
3 # Create your views here.
4
```





## Ocupando la nueva app

Deben agregar lo siguiente:

Lo que estamos haciendo acá, es definir que si entran en la función inicio, les retornaremos la vista inicio.html



```
File Edit Selection Find View Goto Tools Project Preferences Help
views.py
1 from django.shortcuts import render
2
3 # Create your views here.
4
5 def inicio(request):
    return render(request, 'inicio.html')
```



## Ocupando la nueva app

```
File Edit Selection Find View Goto Tools Project Preferences Help
urls.py
1 from django.conf.urls import url, include
2
3
4 urlpatterns = [
5     url(r"^\$", inicio),
6
7
8 ]
9
10 |
```

Lo que hicimos acá, fue decir que si se entra a la dirección `http://<ip>:<puerto>/` retornaremos lo que está en la función `inicio` (que fue lo que acabamos de definir)



**Pregunta:**  
**¿Qué estamos omitiendo?**

*No hemos definido donde  
estarán nuestros templates  
o HTML !!!!*





## Agregar carpeta Templates

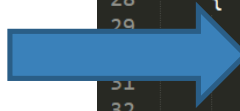
- Copien la carpeta llamada templates que está en el zip, en la raíz del proyecto.
- Ahora debemos agregar esta dirección a la configuración del proyecto.

```
File Edit Selection Find View Goto Tools Project Preferences Help
urls.py settings.py
25 ]
26 ROOT_URLCONF = '<nombre_proyecto>.urls'
27 TEMPLATES = [
28     {
29         'BACKEND': 'django.template.backends.django.DjangoTemplates',
30         'DIRS': [],
31         'APP_DIRS': True,
32         'OPTIONS': {
33             'context_processors': [
34                 'django.template.context_processors.debug',
35                 'django.template.context_processors.request',
36                 'django.contrib.auth.context_processors.auth',
37                 'django.contrib.messages.context_processors.messages',
38             ],
39         },
40     ],
```



## Agregar carpeta Templates

*Ahí no hay nada*



```
File Edit Selection Find View Goto Tools Project Preferences Help
urls.py settings.py
25 ]
26 ROOT_URLCONF = '<nombre_proyecto>.urls'
27 TEMPLATES = [
28     {
29         'BACKEND': 'django.template.backends.django.DjangoTemplates',
30         'DIRS': [],
31         'APP_DIRS': True,
32         'OPTIONS': {
33             'context_processors': [
34                 'django.template.context_processors.debug',
35                 'django.template.context_processors.request',
36                 'django.contrib.auth.context_processors.auth',
37                 'django.contrib.messages.context_processors.messages',
38             ],
39         },
40     },
41 ]
```

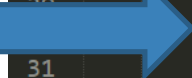


## Agregar carpeta Templates

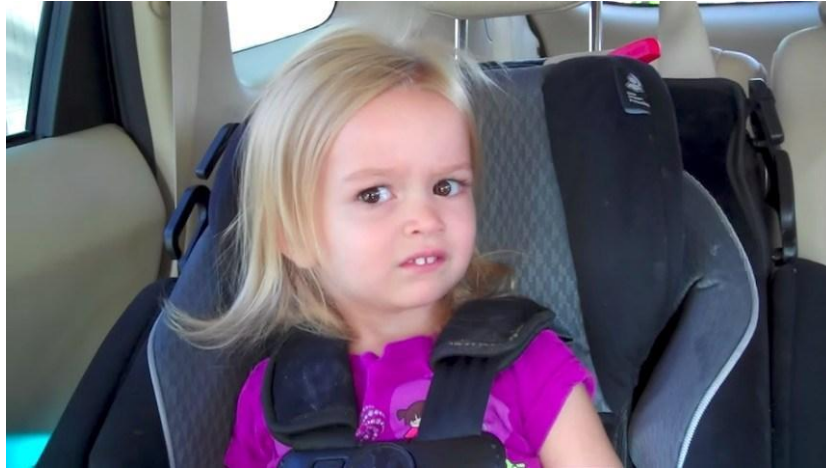
Agreguen esto:

`os.path.join(BASE_DIR, 'templates')`

El comando anterior le dice al proyecto, que para los templates vaya a la dirección base del proyecto (o raíz) y busque en la carpeta templates



```
File Edit Selection Find View Goto Tools Project Preferences Help
<< >> uris.py x settings.py
25 ]
26 ROOT_URLCONF = '<nombre_proyecto>.urls'
27 TEMPLATES = [
28     {
29         'BACKEND': 'django.template.backends.django.DjangoTemplates',
30         'DIRS': [os.path.join(BASE_DIR, 'templates')],
31         'APP_DIRS': True,
32         'OPTIONS': {
33             'context_processors': [
34                 'django.template.context_processors.debug',
35                 'django.template.context_processors.request',
36                 'django.contrib.auth.context_processors.auth',
37                 'django.contrib.messages.context_processors.messages',
38             ],
39         },
40     ],
41 ]
```



**Lamento decirles,  
pero aún falta algo**





## Agregar las direcciones a la URL generales del proyecto

Esto tenemos en el archivo `urls.py` de la carpeta que lleva el nombre del proyecto.

Debemos agregar que, si entra determinada url, vaya a las urls de nuestra aplicación

```
Project  Preferences  Help
< >  urls.py  x
13      1. Import the include() function: from d
14      2. Add a URL to urlpatterns:  path('blog/
15      """
16      from django.contrib import admin
17      from django.urls import path
18
19      urlpatterns = [
20          path('admin/', admin.site.urls),
21      ]
22  |
```



## Agregar las direcciones a la URL generales del proyecto

Esto tenemos en el archivo `urls.py` de la carpeta que lleva el nombre del proyecto.

Debemos agregar que, si entra determinada url, vaya a las urls de nuestra aplicación

```
Project  Preferences  Help
< >  urls.py  x
13      1. Import the include() function: from d
14      2. Add a URL to urlpatterns:  path('blog/
15      """
16      from django.contrib import admin
17      from django.urls import path
18
19      urlpatterns = [
20          path('admin/', admin.site.urls),
21      ]
22  |
```




## Agregar las direcciones a la URL generales del proyecto

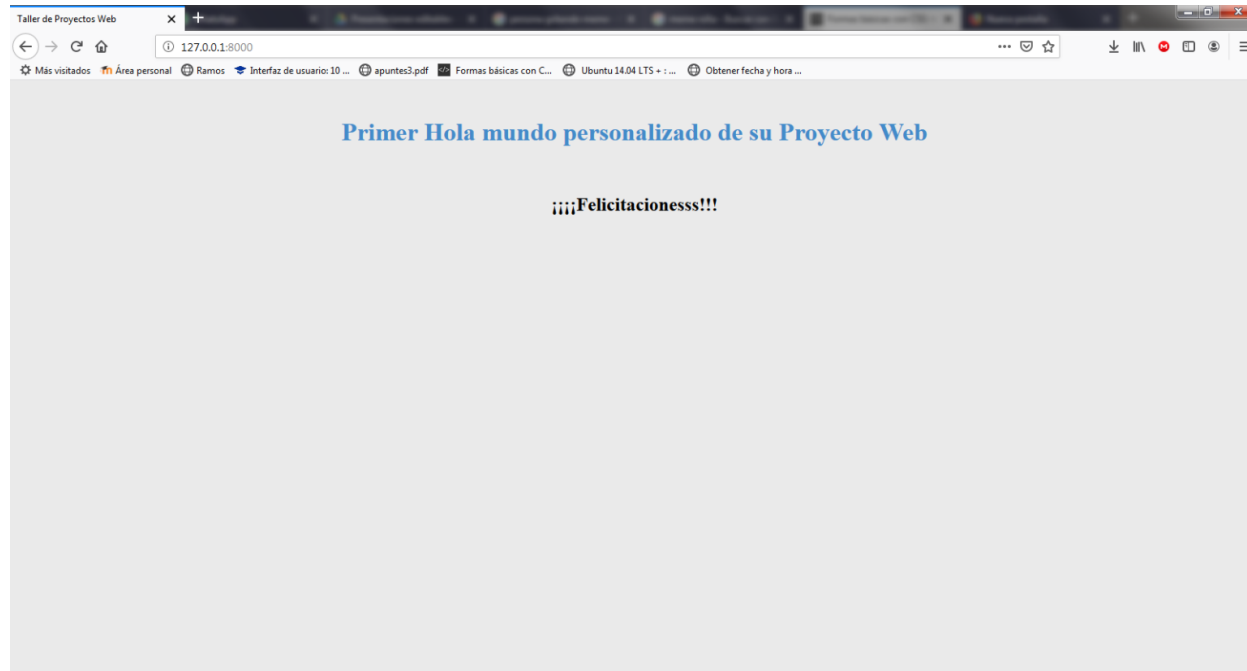
Agregamos los que si entra url vacío, vaya a buscar a los urls de proyectos.urls que es nuestra app anteriormente creada

```
Project  Preferences  Help
urls.py — taller\taller  urls.py — MasFondos\...\Pagina  x
14 2. Add a URL to urlpatterns: path('blog/', i
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21     path(r'', include('proyectos.urls'))
22 ]
23
```

Agregamos la función include



**Corremos nuestro  
proyecto, vamos a  
localhost:8000 y  
tenemos...**





## Actividad Evaluada 2

1. Cree su proyecto Django
2. Agregue una app.
3. Haga todos los pasos que hicimos para agregar una vista
4. Ponga su nombre o rol más algo característico en el html, y verifique que funciona correctamente
5. Suba su proyecto comprimido a Moodle



# Gracias por venir!

¿Preguntas?

La próxima clase, mejoraremos la calidad de las vistas 😊

**[Jorge.diazma@sansano.usm.cl](mailto:Jorge.diazma@sansano.usm.cl)**