

<https://www.halvorsen.blog>

django

Django Admin

Hans-Petter Halvorsen



Contents

- Introduction
 - Short introduction to Django.
 - Short Introduction to the Django “Company” App created in a previous Tutorial.
- Django Admin
 - “Django Admin“ is included with Django.
 - Django provides a ready-to-use user interface for administrative activities.
 - Django automatically generates admin UI based on your Django Models.
 - “Django Admin” offers a CRUD user interface for all your Models.
 - CRUD is short for Create, Read, Update and Delete.

<https://www.halvorsen.blog>

Introduction



[Table of Contents](#)

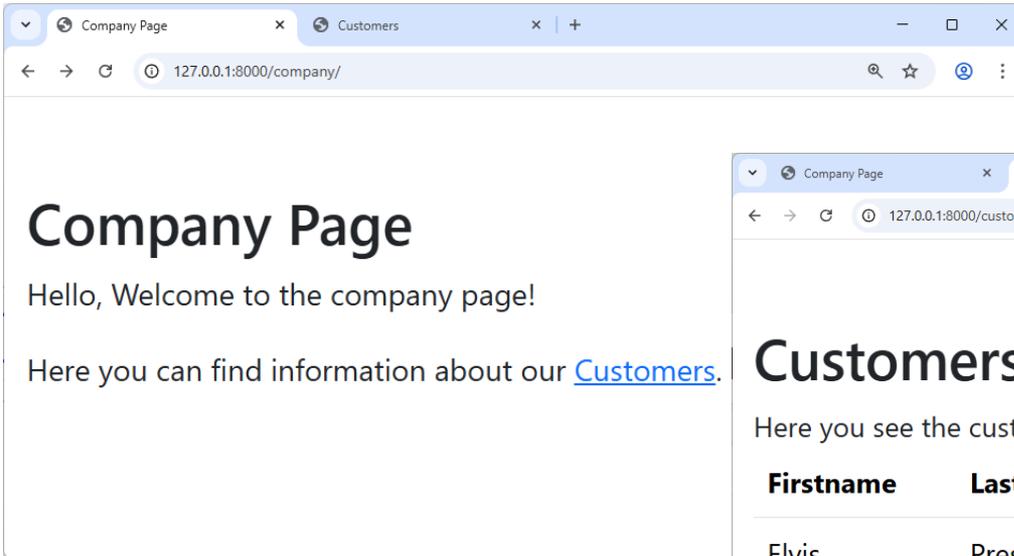
Hans-Petter Halvorsen

Django

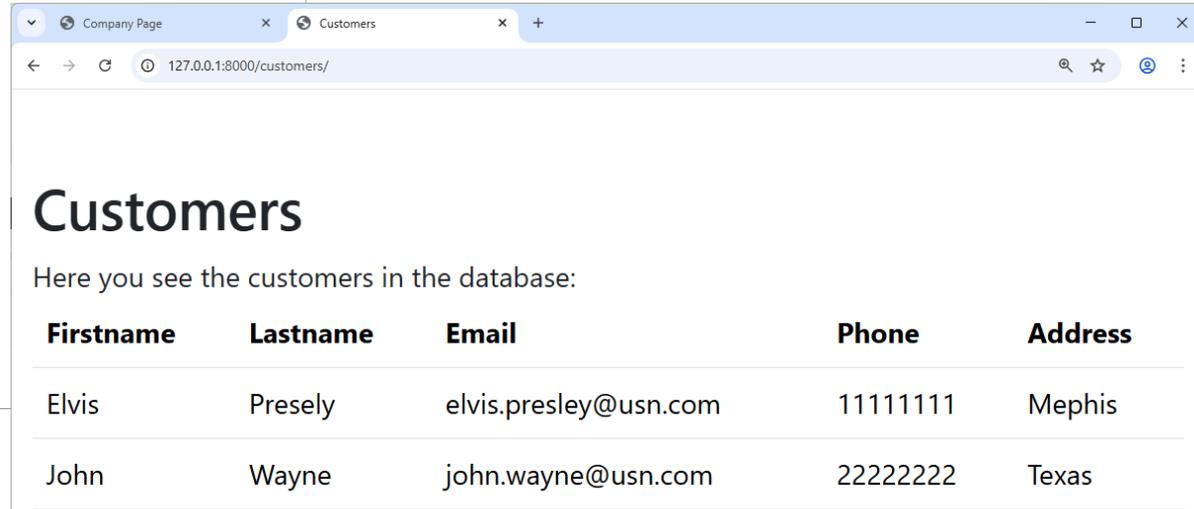
- Django is a Python web development framework.
- Django is a back-end/server-side web framework.
- Django has support for databases like SQLite, MySQL/MariaDB and PostgreSQL.
- Django includes an SQLite database, but for the others you need to install the database system you want to use from their respective websites and in addition install a Python package/driver for the specific database system.
- Django is free, open source and written in Python.
- Homepage: <https://www.djangoproject.com>

Django “Company” App

In the “Django Tutorial” we created the following Application with Django:



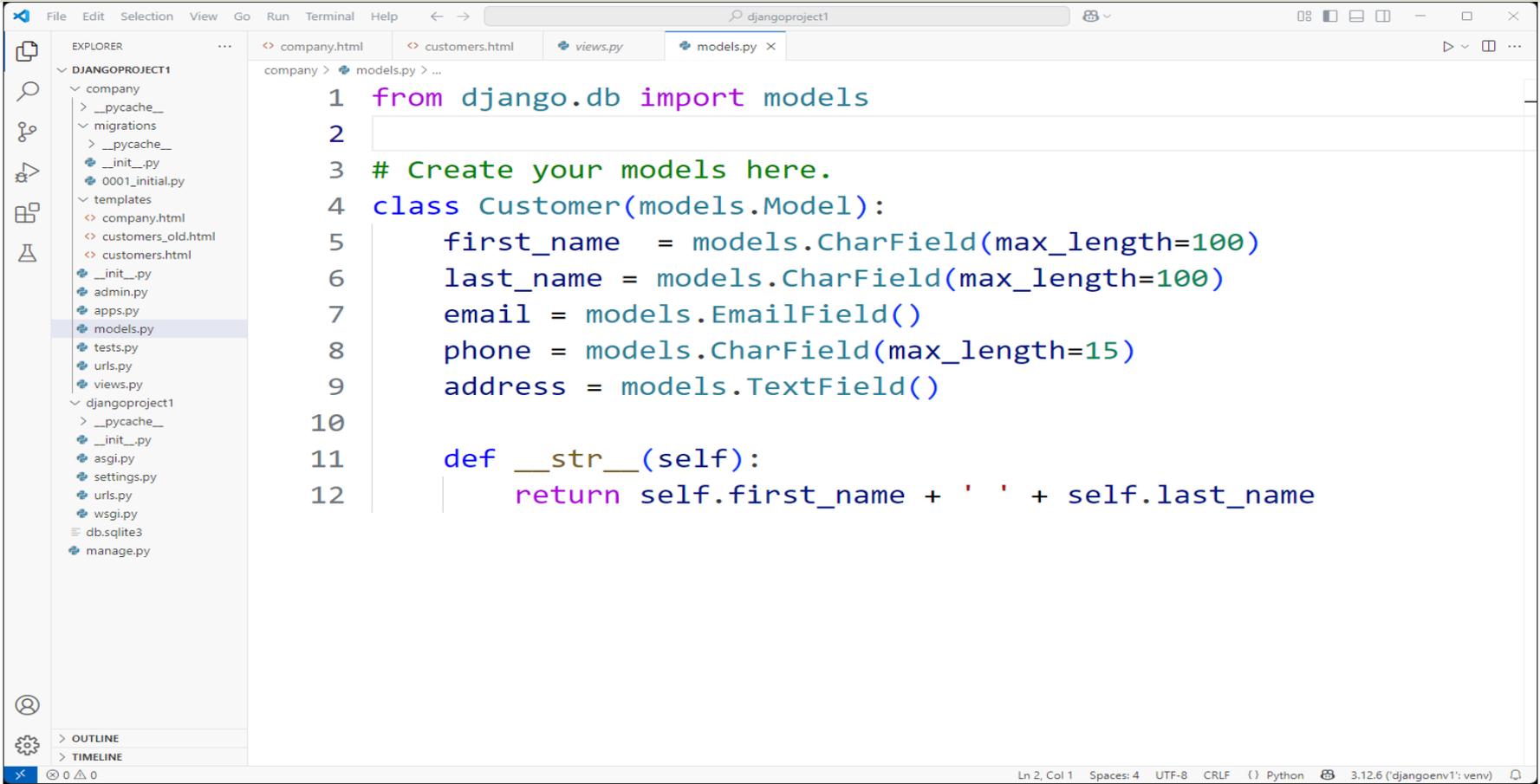
Subpage showing the “Customers” for the “Company” App:



Main “Company” App

We will use the “Django Admin” interface to create CRUD functionality for the “Customer”, i.e., Create (insert), Read (select), Update and Delete data from the database.

Django “Company” Project



The image shows a code editor window for a Django project named 'django-project1'. The Explorer on the left shows the project structure, including folders for 'company', 'migrations', 'templates', and 'django-project1'. The 'models.py' file is selected and open in the editor. The code in the editor defines a Django model named 'Customer' with fields for first_name, last_name, email, phone, and address. The model also has a custom string representation method.

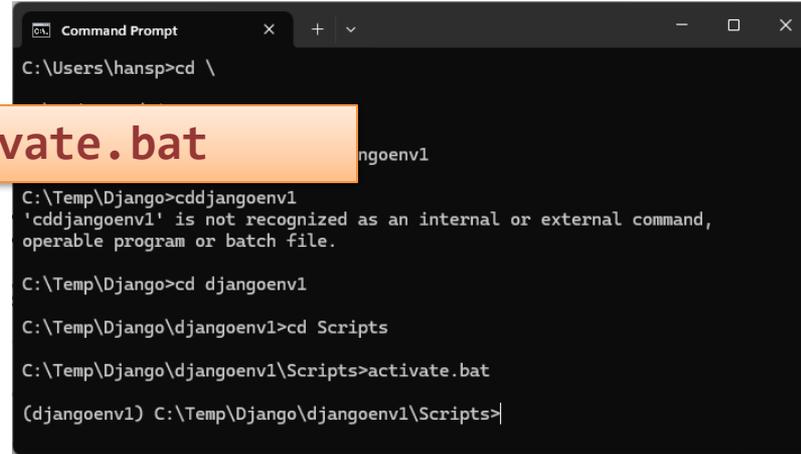
```
1 from django.db import models
2
3 # Create your models here.
4 class Customer(models.Model):
5     first_name = models.CharField(max_length=100)
6     last_name = models.CharField(max_length=100)
7     email = models.EmailField()
8     phone = models.CharField(max_length=15)
9     address = models.TextField()
10
11 def __str__(self):
12     return self.first_name + ' ' + self.last_name
```

Ln 2, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.6 ('djangoenv1': venv)

Start Django “Company” Project

Activate the Virtual Python Environment

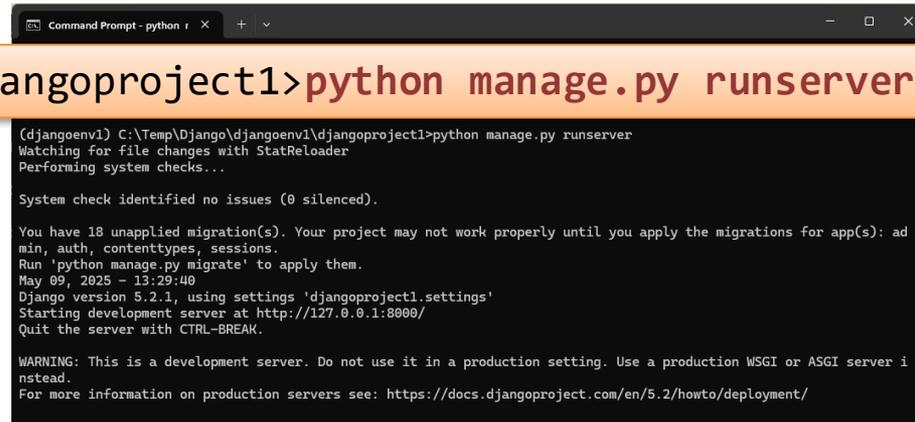
```
C:\Temp\Django\djangoenv1\Scripts>activate.bat
```



```
Command Prompt
C:\Users\hansp>cd \
C:\Temp\Django>cddjangoenv1
'cddjangoenv1' is not recognized as an internal or external command,
operable program or batch file.
C:\Temp\Django>cd djangoenv1
C:\Temp\Django\djangoenv1>cd Scripts
C:\Temp\Django\djangoenv1\Scripts>activate.bat
(djangoenv1) C:\Temp\Django\djangoenv1\Scripts>
```

Run the Django Project:

```
(djangoenv1) C:\Temp\Django\djangoenv1\djangoproject1>python manage.py runserver
```



```
Command Prompt - python r x
(djangoenv1) C:\Temp\Django\djangoenv1\djangoproject1>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): ad
min, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
May 09, 2025 - 13:29:40
Django version 5.2.1, using settings 'djangoproject1.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server i
nstead.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
```

Open your Web Browser and enter the
URL: **127.0.0.1:8000/company**

Django Tutorial and “Company App”

- Django Tutorial:
<https://www.youtube.com/playlist?list=PLdb-TcK6Aqj0enrpIQ0P4ISl8ofhk8Sza>
- This Tutorial gives an overview of Django and create a “Company” Django App step by step.
- The “Company” App is used as a starting point for this “Django Admin“ Tutorial.

<https://www.halvorsen.blog>

Django Admin

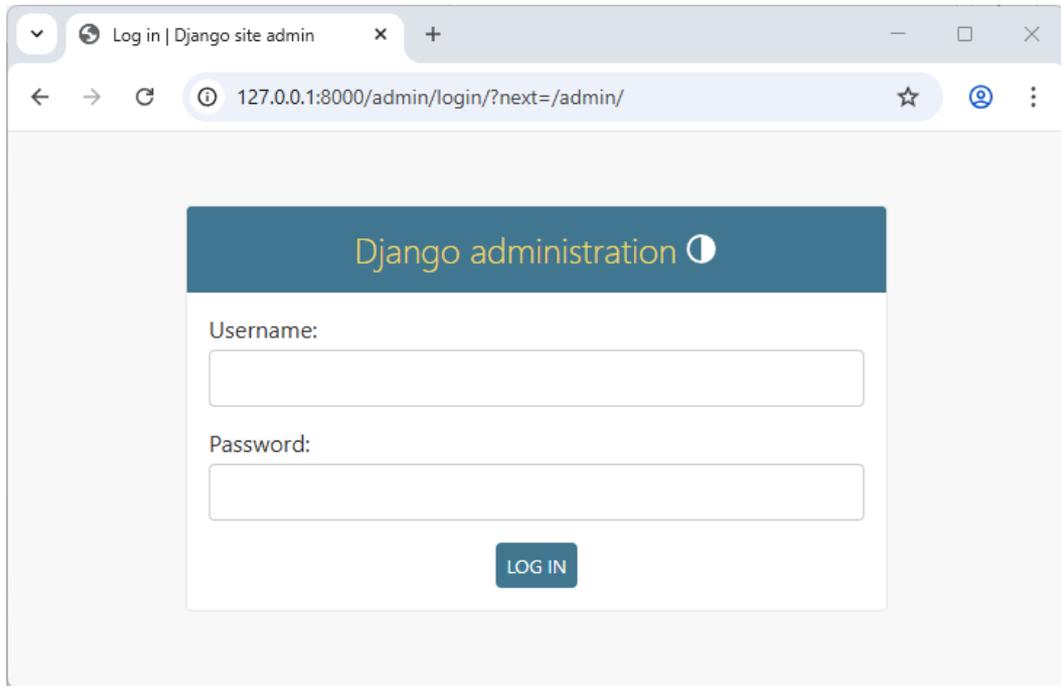


[Table of Contents](#)

Hans-Petter Halvorsen

Django Admin

<http://127.0.0.1:8000/admin>



The image shows a browser window with the title "Log in | Django site admin". The address bar contains the URL "127.0.0.1:8000/admin/login/?next=/admin/". The main content area displays the Django administration login form. At the top of the form is a dark blue header with the text "Django administration" and a moon icon. Below this, there are two input fields: "Username:" and "Password:". At the bottom of the form is a blue button labeled "LOG IN".

“Django Admin“ is included with Django.

Before we can start using “Django Admin”, we need to create a User.

Django Admin

- “Django Admin“ is included with Django.
- Django provides a ready-to-use user interface for administrative activities. Django automatically generates admin UI based on your Django Models.
- “Django Admin” offers a CRUD user interface for all your Models.
- CRUD is short for Create, Read, Update and Delete.
- The “Django Admin“ interface is intended for “superusers” and not for ordinary users of your application.
- To use it you need to create a User.

Django Admin - Create User

Type the following to create a new User:

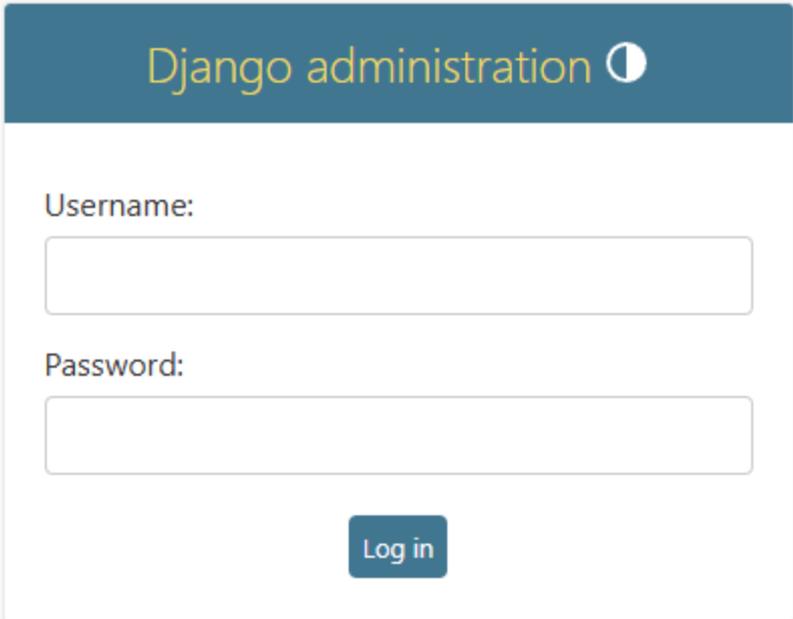
```
...>python manage.py createsuperuser
```

Then you are asked to create a Username and a Password.

Then run the server:

```
...>python manage.py runserver
```

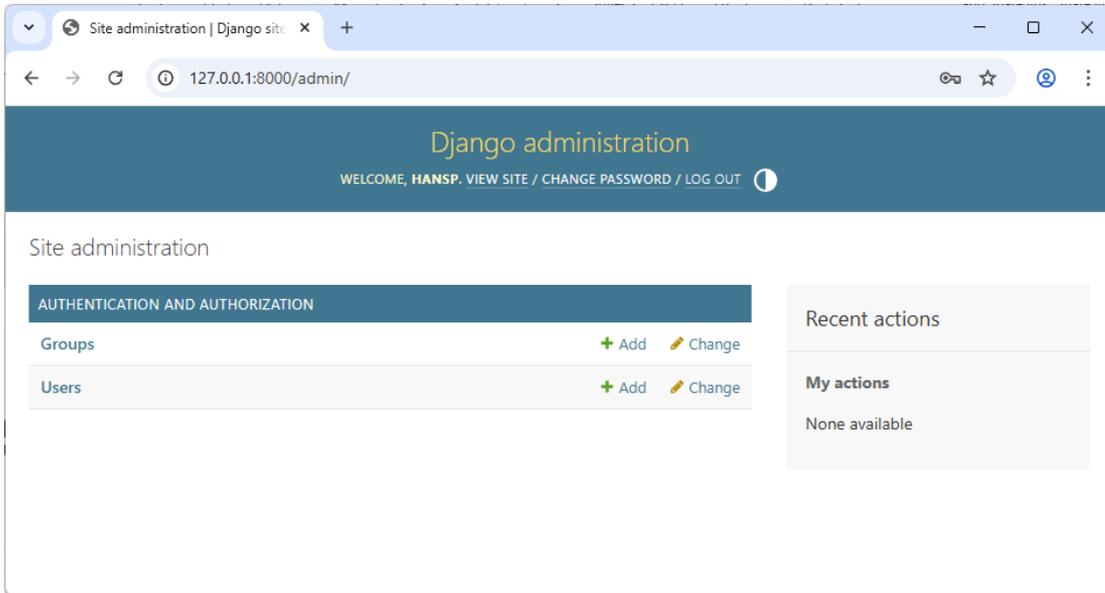
Goto <http://127.0.0.1:8000/admin> and enter your Username and Password:



The screenshot shows the Django administration login interface. At the top, there is a dark blue header with the text "Django administration" and a circular icon. Below the header, the page is white. There are two input fields: "Username:" followed by a text input box, and "Password:" followed by a text input box. At the bottom center, there is a dark blue button with the text "Log in".

Django Admin

The following will appear:



Here you can Add, Change and Delete (CRUD functionality) Groups and Users.

Existing Customer Model

```
models.py ×
company > models.py > ...
1 from django.db import models
2
3 # Create your models here.
4 class Customer(models.Model):
5     first_name = models.CharField(max_length=100)
6     last_name = models.CharField(max_length=100)
7     email = models.EmailField()
8     phone = models.CharField(max_length=15)
9     address = models.TextField()
10
11     def __str__(self):
12         return self.first_name + ' ' + self.last_name
```

By default, all Models created in the Django project will be created as tables in the default SQLite database (“db.sqlite3”) that is installed with Django.

Add “Customer” Model to Admin

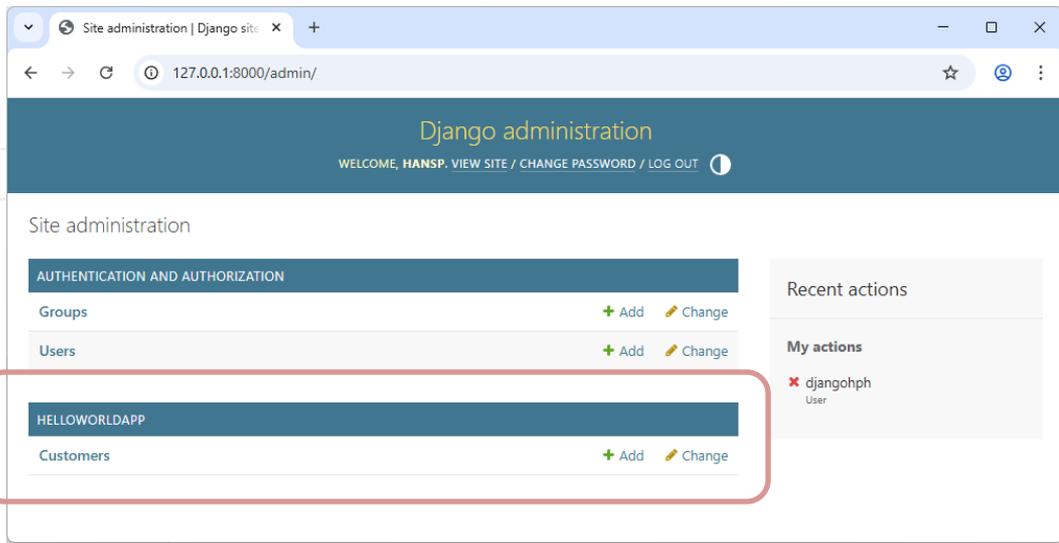
Django Admin offers a CRUD user interface for all your Models.
Let's add CRUD functionality for our Customer Model.

You need to register the Customer Model in the file “**admin.py**”:

```
admin.py x
helloworldapp > admin.py
1 from django.contrib import admin
2 from .models import Customer
3
4 # Register your models here.
5 admin.site.register(Customer)
```

Then go back to

<http://127.0.0.1:8000/admin>



Customer CRUD Interface

This screenshot shows the Django administration interface for the 'Customers' model. At the top, a green success message states: "The customer 'Donald Trump' was deleted successfully." Below this, the page title is "Select customer to change". There is a dropdown menu for "Action:" with a "Go" button and a "0 of 2 selected" indicator. A list of customers is displayed with checkboxes:

- CUSTOMER
- John Wayne
- Elvis Presely

At the bottom of the list, it says "2 customers". On the left sidebar, the "COMPANY" section is expanded, and "Customers" is highlighted with a green background and a "+ Add" button.

This screenshot shows the Django administration interface for the 'Add customer' form. The page title is "Add customer". The form contains the following fields:

- First name:**
- Last name:**
- Email:**
- Phone:**
- Address:**

At the bottom of the page, there are three buttons: "SAVE", "Save and add another", and "Save and continue editing". The left sidebar is identical to the previous screenshot, showing the "Customers" model selected under the "COMPANY" section.

Change Display Name

Select customer to change

Action: ----- Go 0 of 2 selected

- CUSTOMER
- Customer object (2)
- Customer object (1)

2 customers

Update the Customer Model in models.py:

```
models.py x
helloworldapp > models.py > ...
1 from django.db import models
2
3 class Customer(models.Model):
4     firstname = models.CharField(max_length=100)
5     lastname = models.CharField(max_length=100)
6     address = models.CharField(max_length=100)
7     city = models.CharField(max_length=100)
8     email = models.EmailField(max_length=100)
9     phone = models.CharField(max_length=20)
10
11     def __str__(self):
12         return f'{self.firstname} {self.lastname}'
```



Django administration

WELCOME, HANSP. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Company > Customers

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

- Groups + Add
- Users + Add

COMPANY

- Customers + Add

The customer "Donald Trump" was deleted successfully.

Select customer to change ADD CUSTOMER +

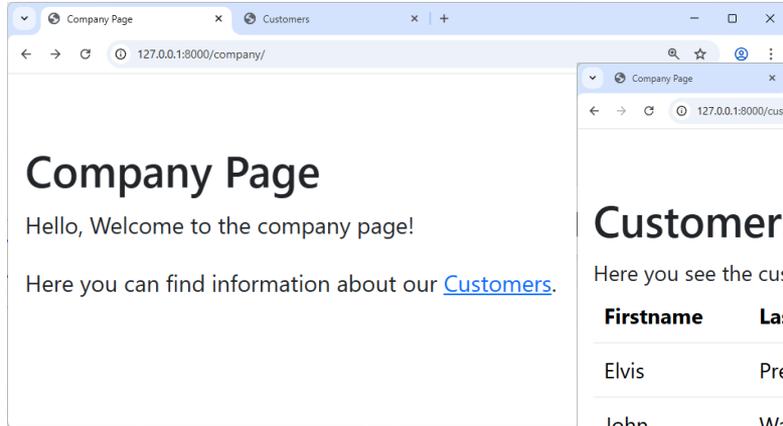
Action: ----- Go 0 of 2 selected

- CUSTOMER
- John Wayne
- Elvis Presely

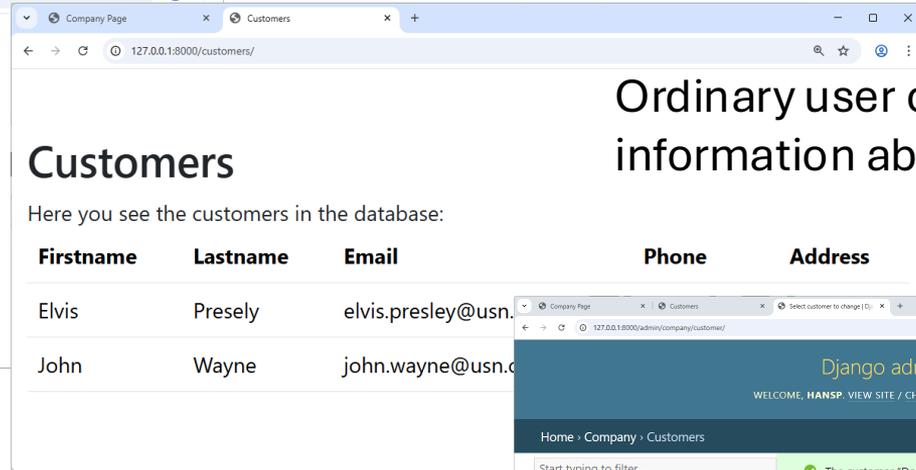
2 customers

Final Django “Company” App

Main “Company” App

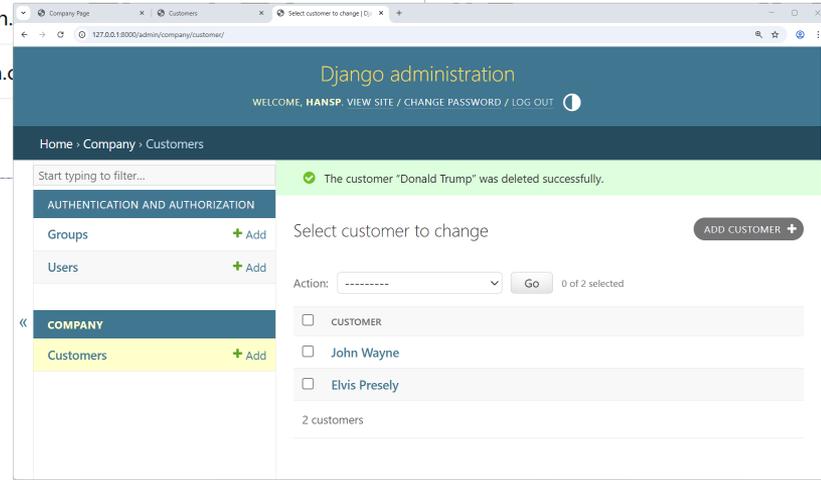


Subpage showing the “Customers” for the “Company” App:



Ordinary user can only see information about the customers

Admin interface for creating, changing and deleting Customers (this part is only available for superusers, and administrators and you need to login):



Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

