

Django

Build an organized web application

Presented by Youssef BADDI

Department of Computer Science, EST SB, Chouaib Doukkali University

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
4. Tutorial
5. Conclusion

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
4. Tutorial
5. Conclusion

Introduction

What is Django?
A high-level Python Web Framework

Introduction

What is Framework?

Front-end & Back-end co-development

Introduction

What is Framework like?
A stage company for performers

Introduction

Why should I use Django?
Clean & Rapid development

Introduction

Who use Django?

Instagram & Pinterest & Bitbucket

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
4. Tutorial
5. Conclusion

Outline

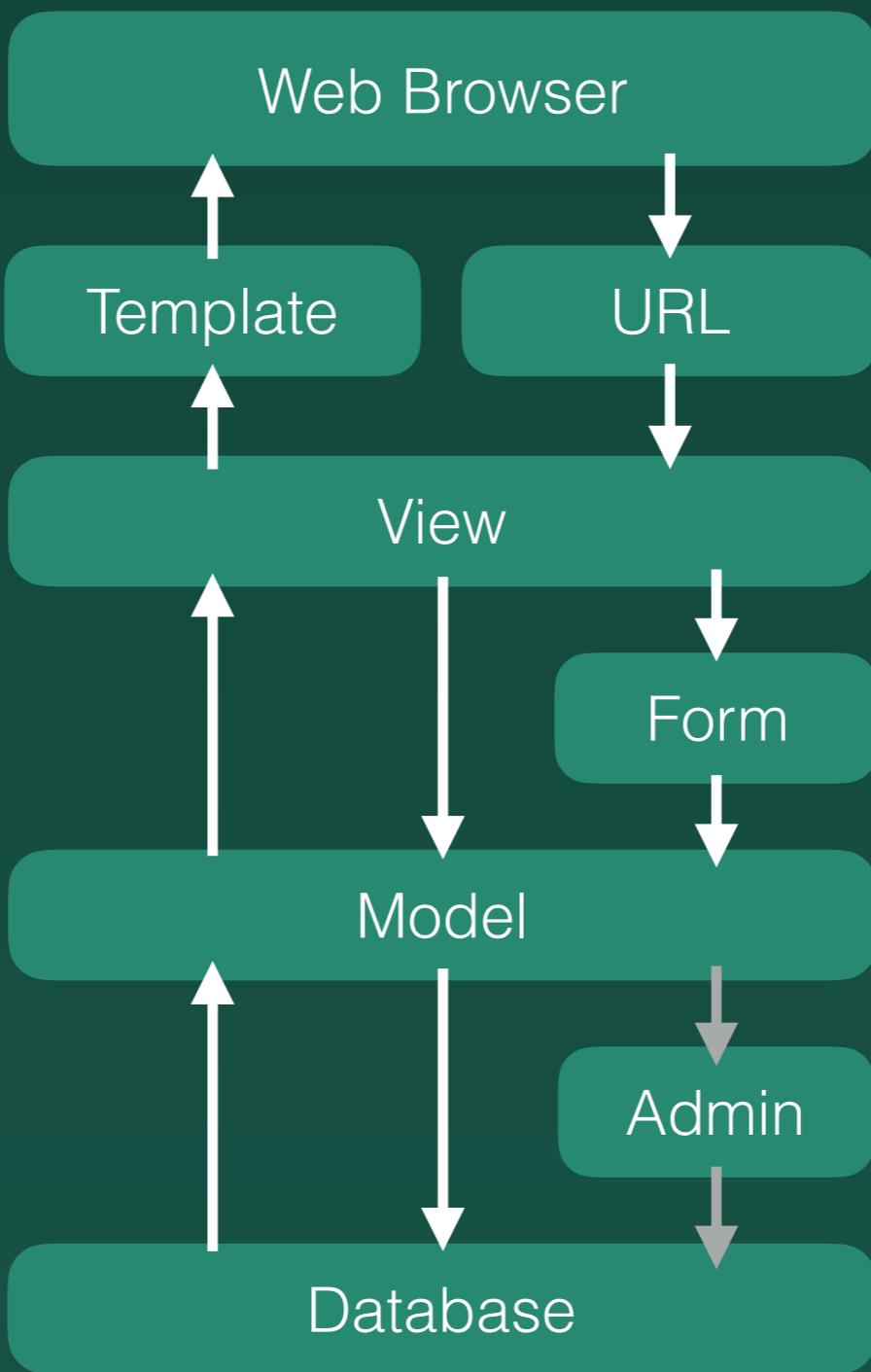
1. Introduction
- 2. Basic Knowledge**
3. Technical Details
4. Tutorial
5. Conclusion

Basic Knowledge

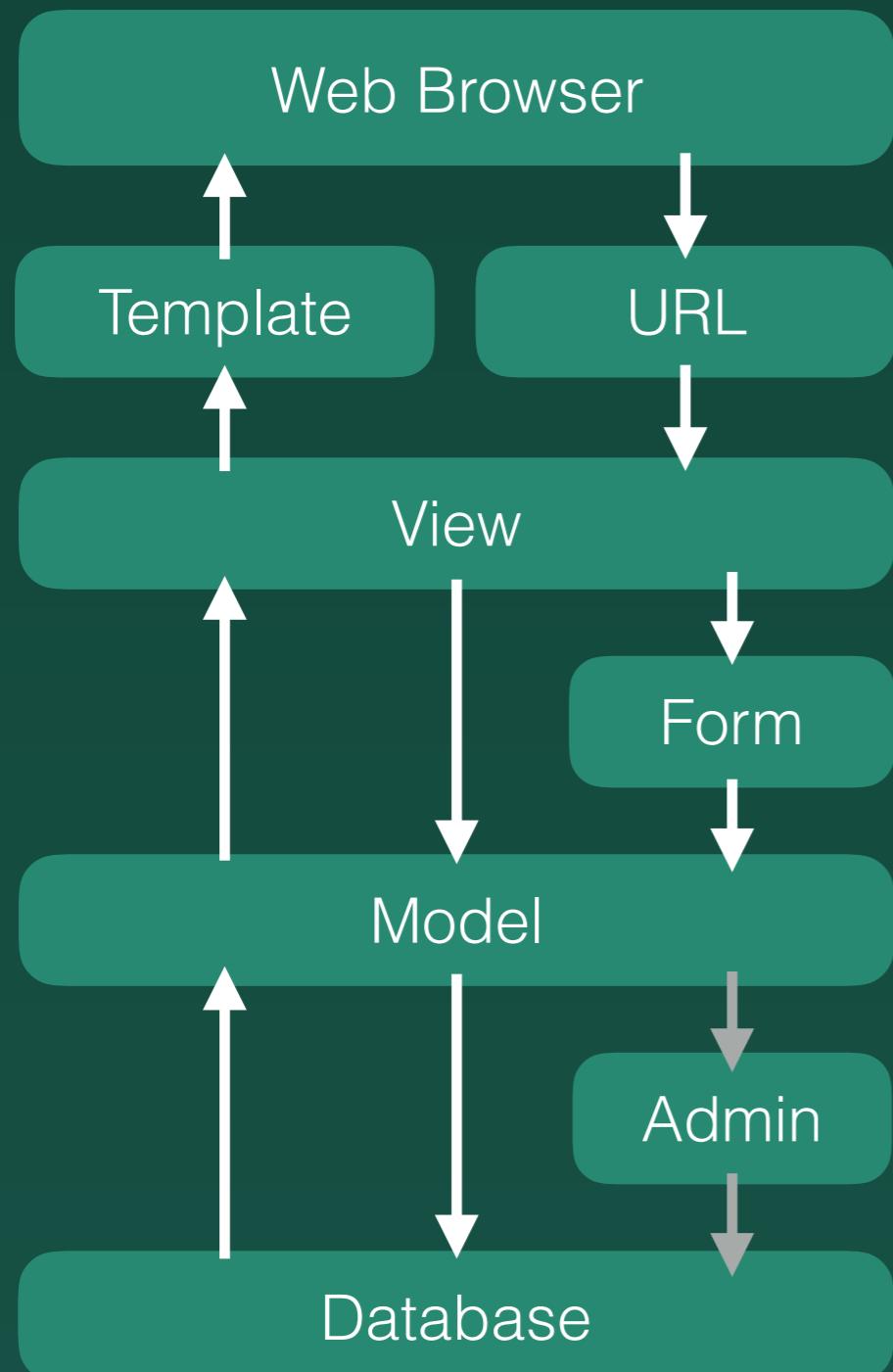


Basic Knowledge

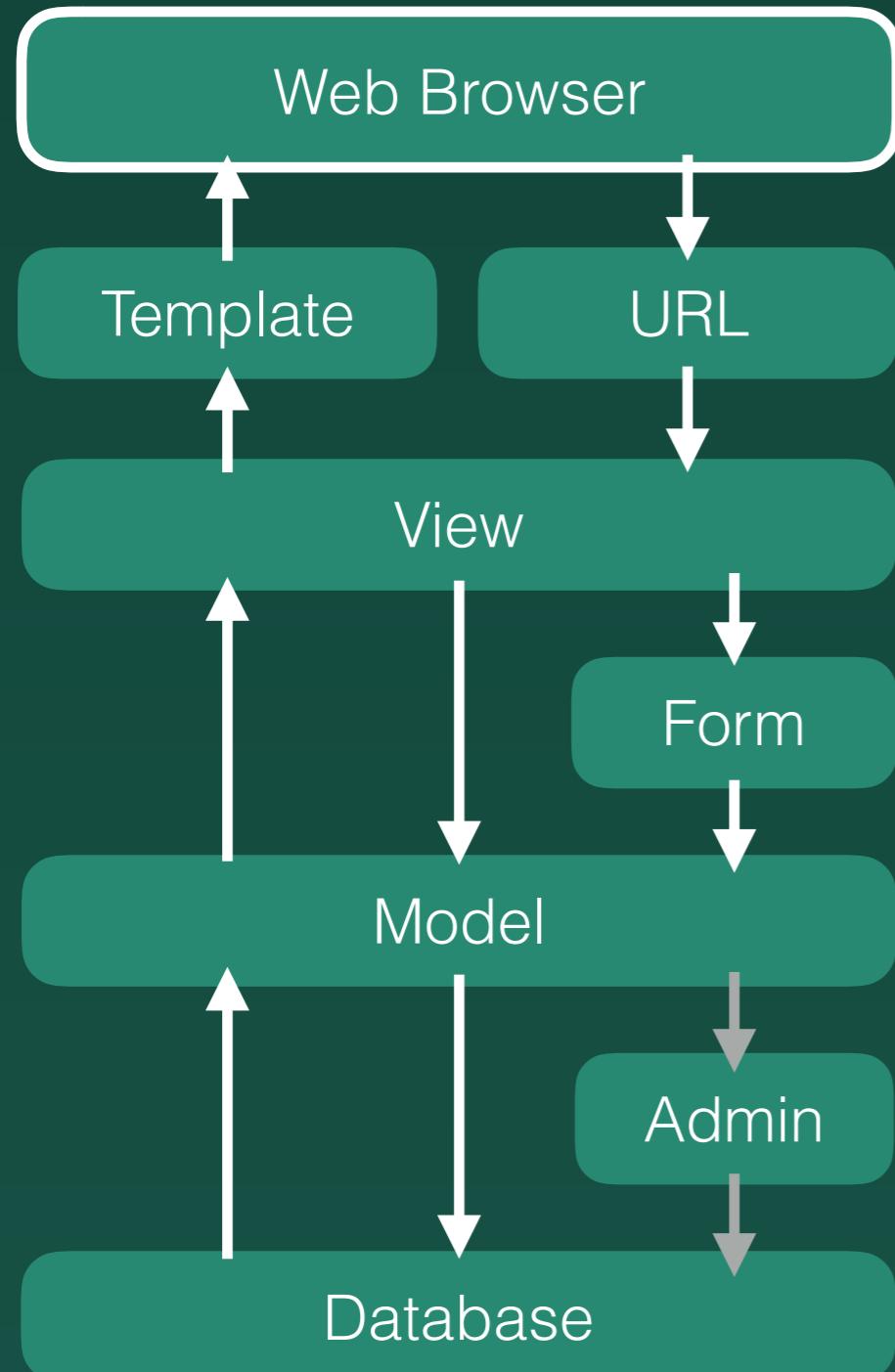
Workflow



Basic Knowledge



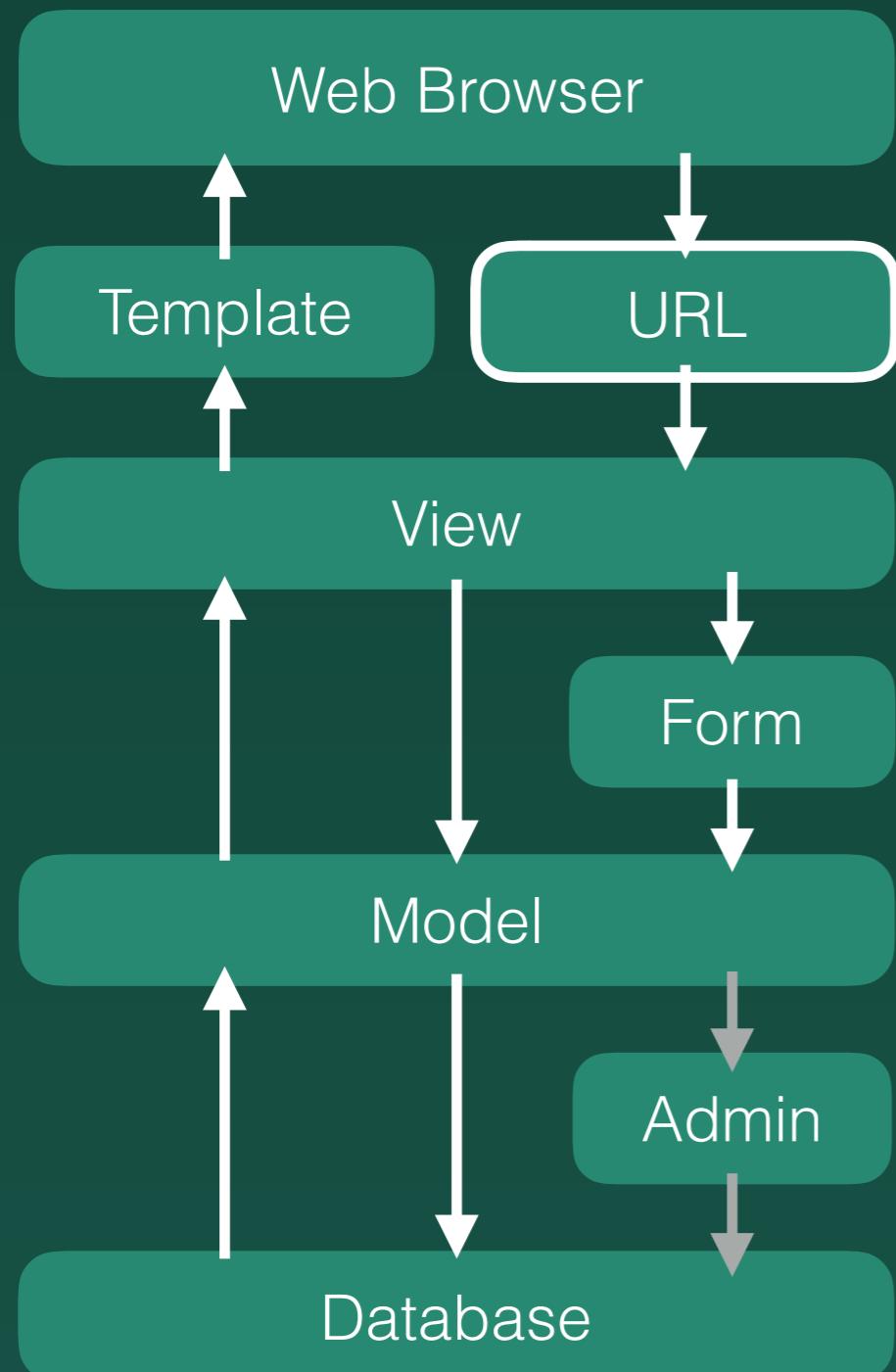
Basic Knowledge



Web Browser

- What users actually see
- Responds to what users do when they
 - 1. Click
 - 2. Type
 - 3. Press Enter

Basic Knowledge



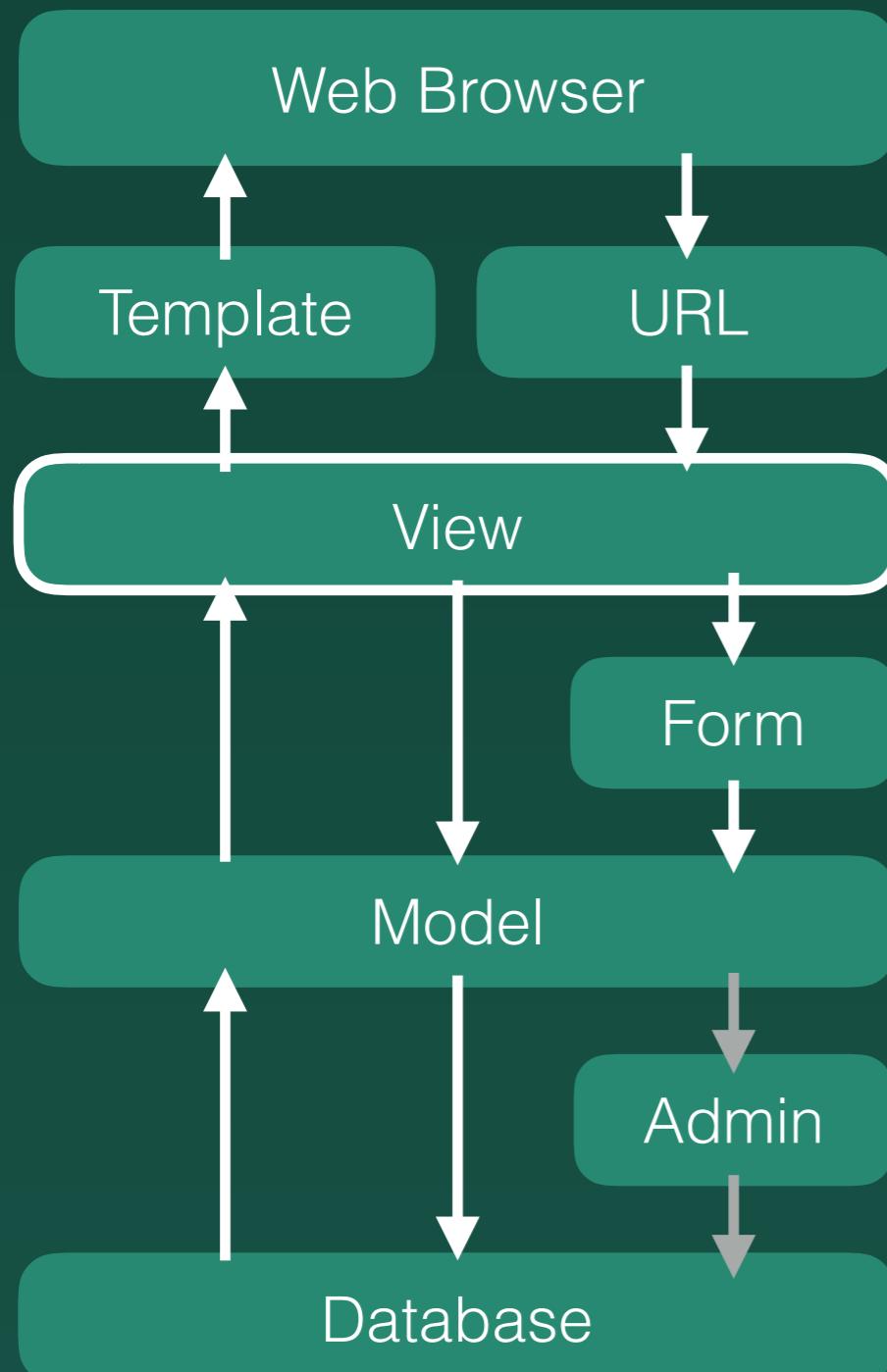
URL

- Often referred as the ‘web address’
- Provides **mapping** to View

What is mapping?

The act of assigning
functions to URLs

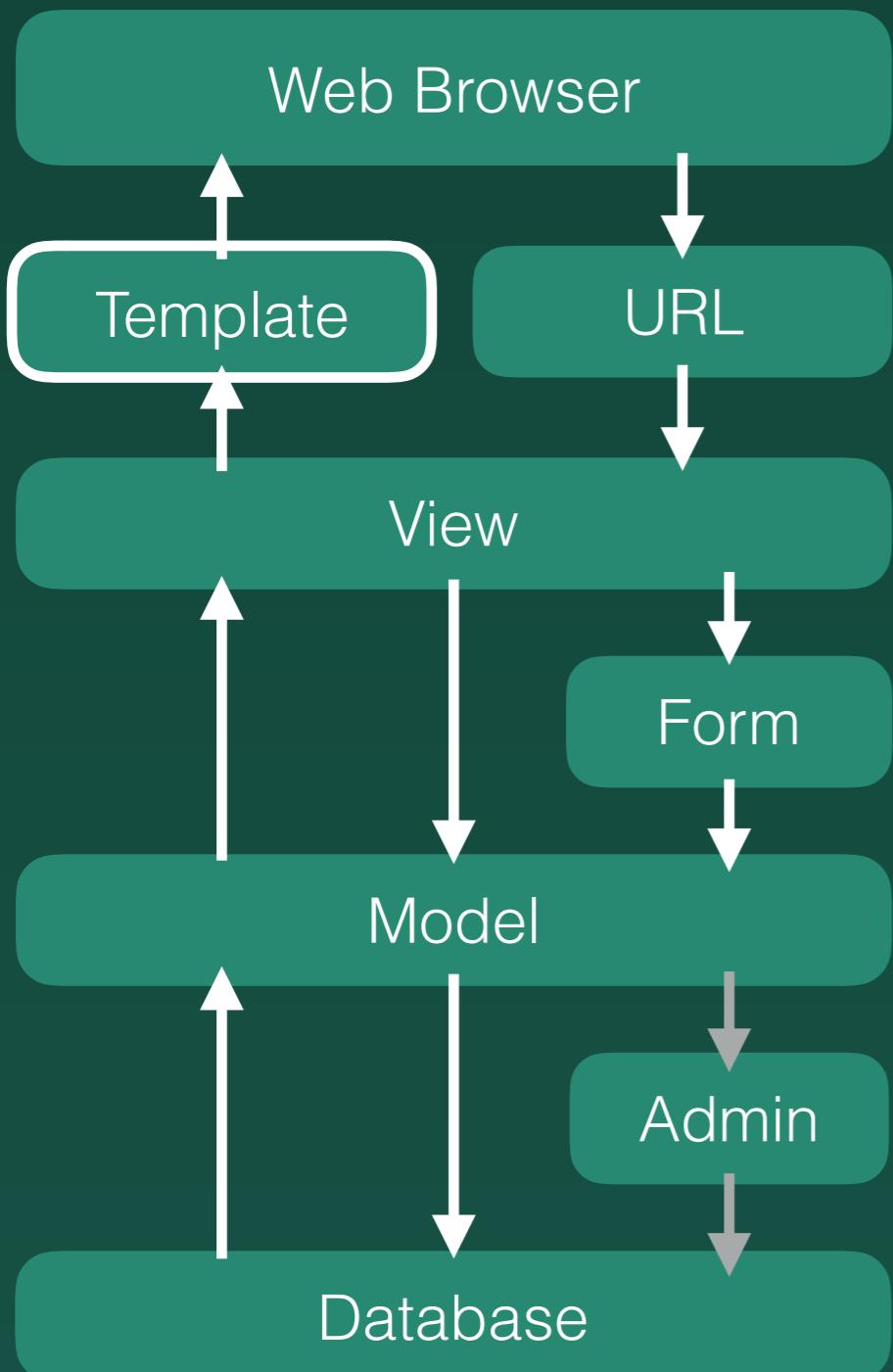
Basic Knowledge



View

- Where all the functions are written
- Render content to **Template**
- Get information from **Model** before rendering content
- Put information into **Model** and into **Database** through **Form**

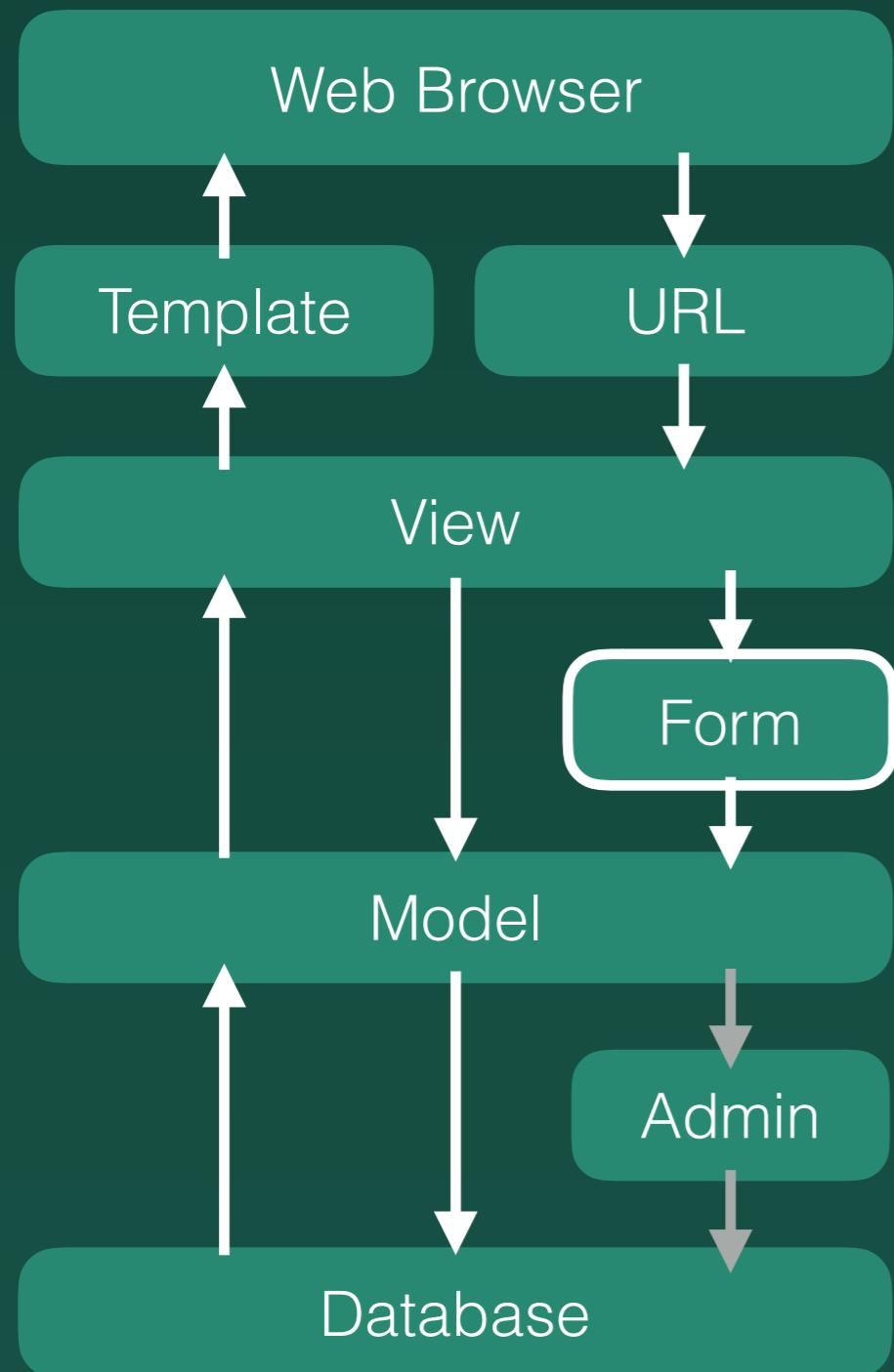
Basic Knowledge



Template

- The place you systematically store all of your **Html** files
- You will have a ‘static’ folder to store other **CSS** files, **Javascript** files, or Images

Basic Knowledge



Form

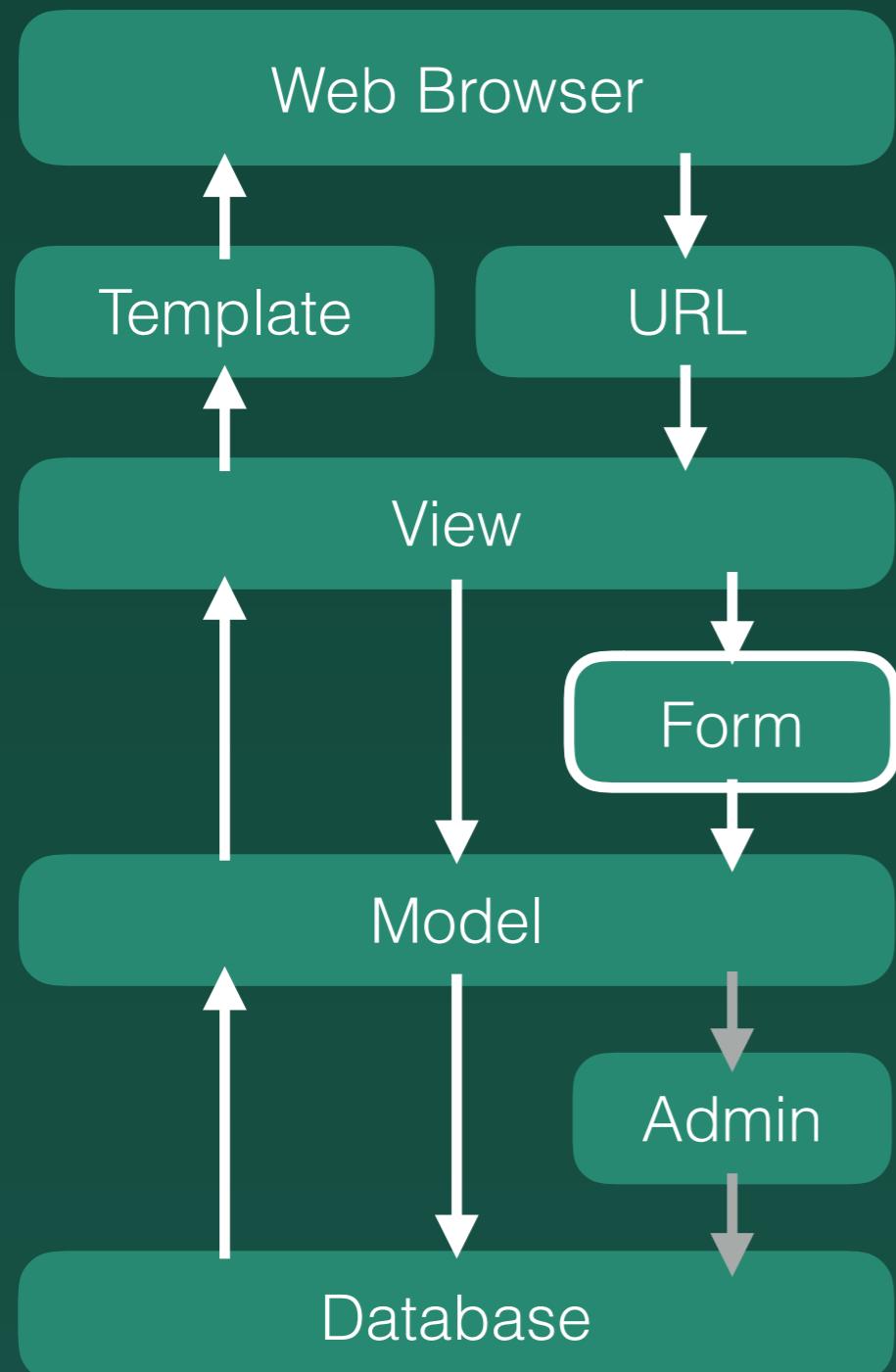
HTML Form \neq Django Form

HTML Form

Consists of

- input element
- checkbox
- submit button
- radio button
- and much more

Basic Knowledge



Form

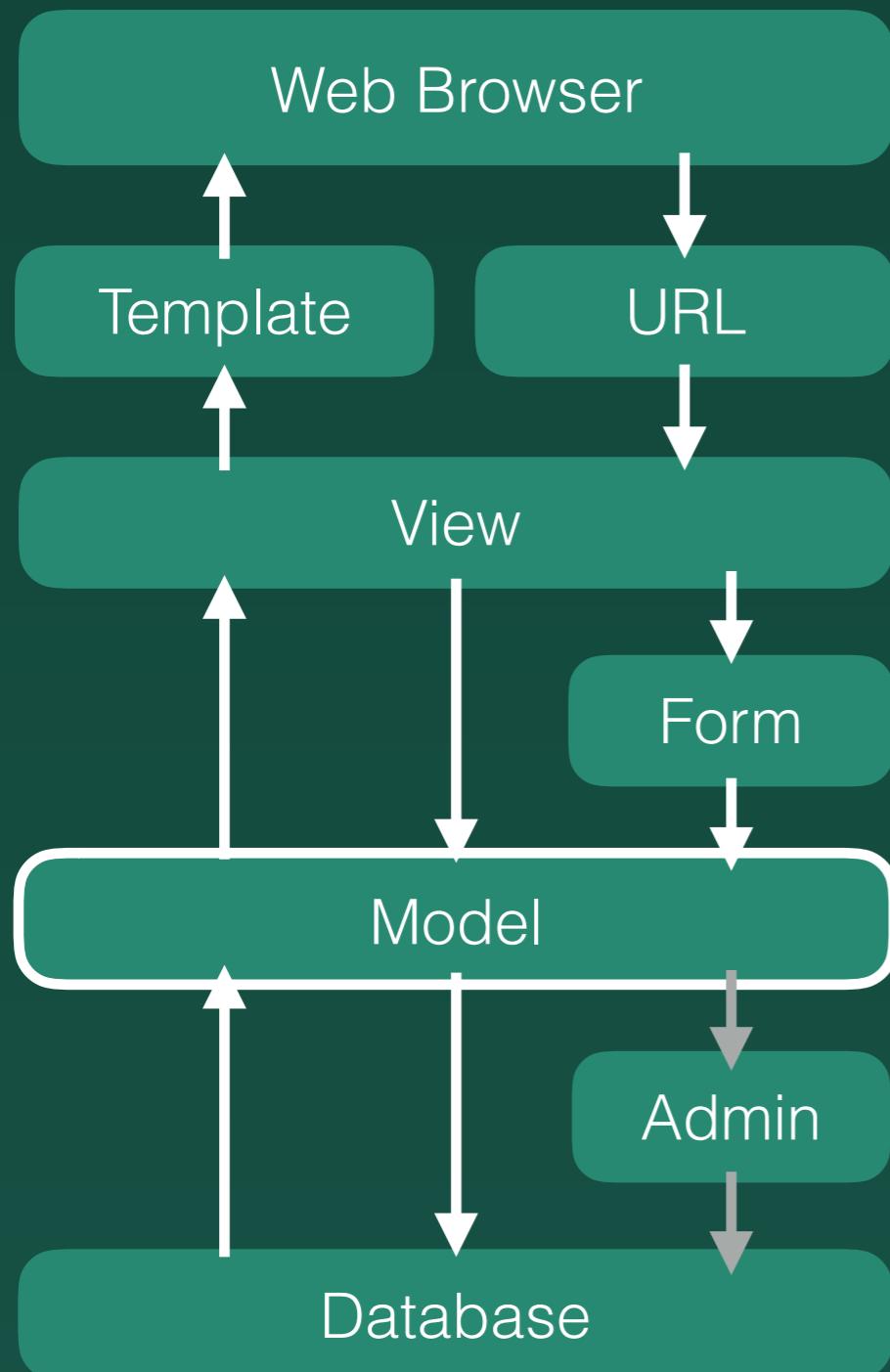
HTML Form \neq Django Form

Django Form

Aims to

- fetch data from html form
- helps to connect to **Model**

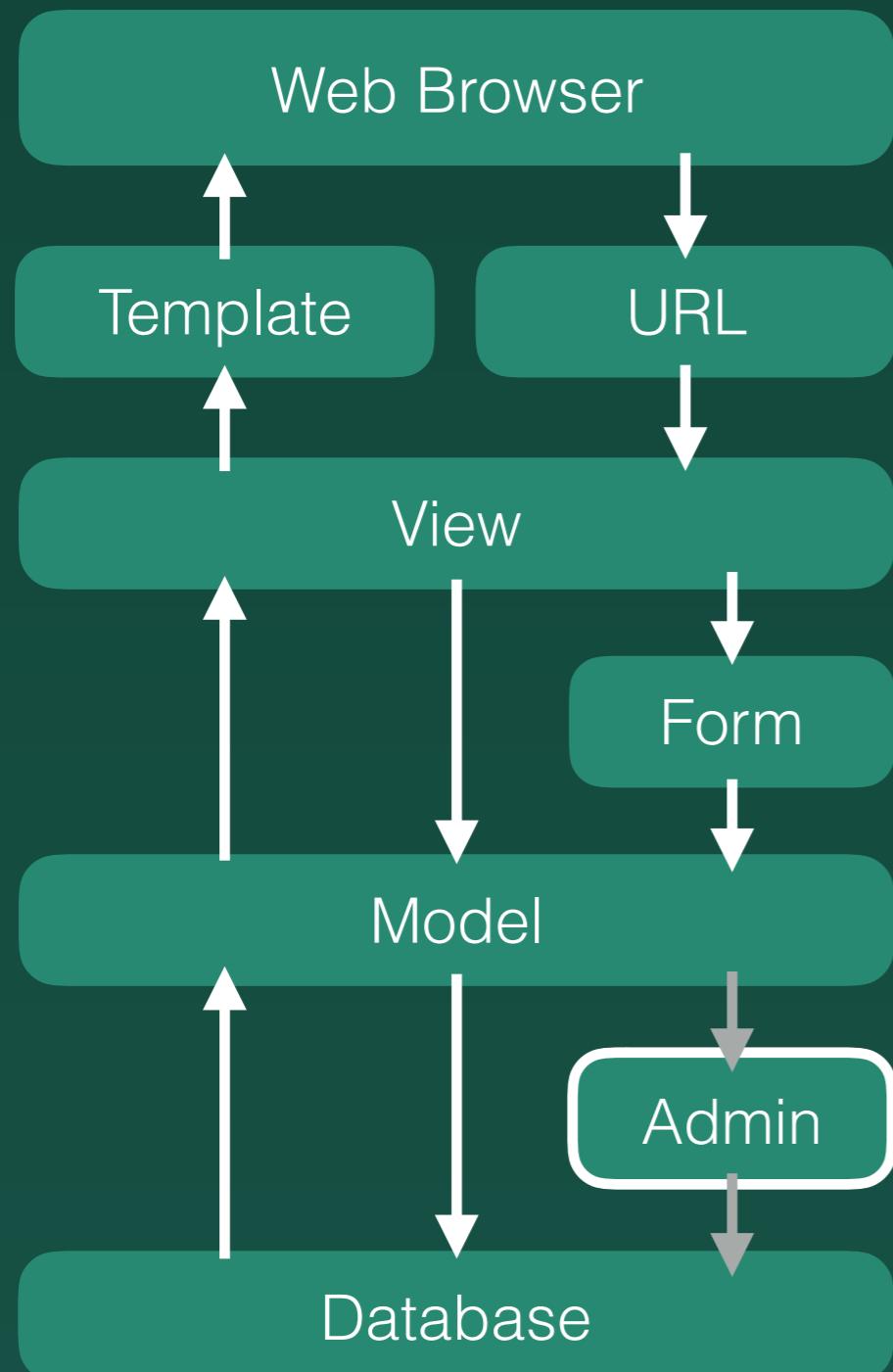
Basic Knowledge



Model

- Describes the structure of an object you want to store in **Database**
- Goes straight to **Database** and create & edit & request information as you wish

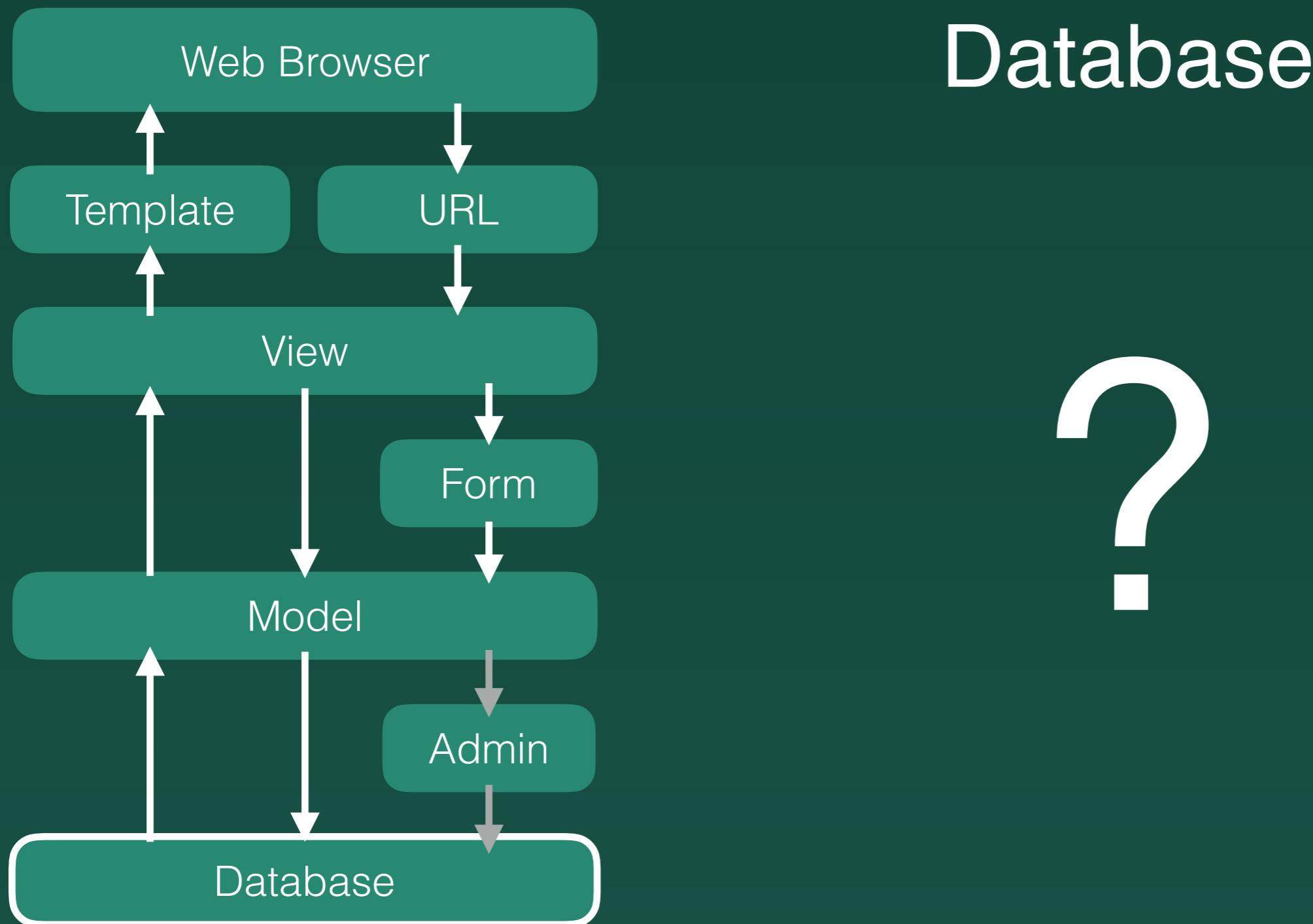
Basic Knowledge



Admin

- Helps to register your object in your **Model** so you can manage data in **Database**
- The registration has to be done in the first place

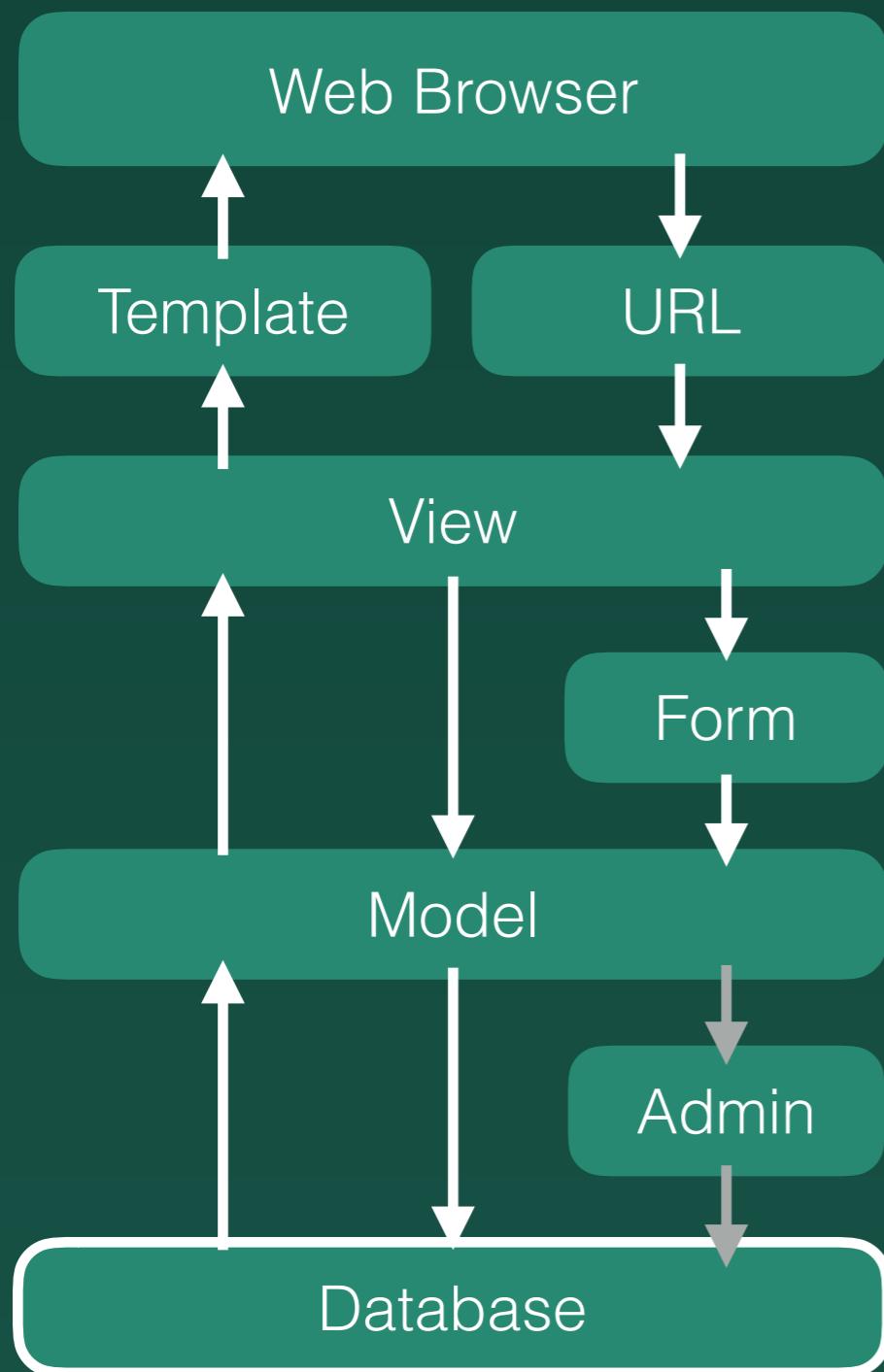
Basic Knowledge



Database



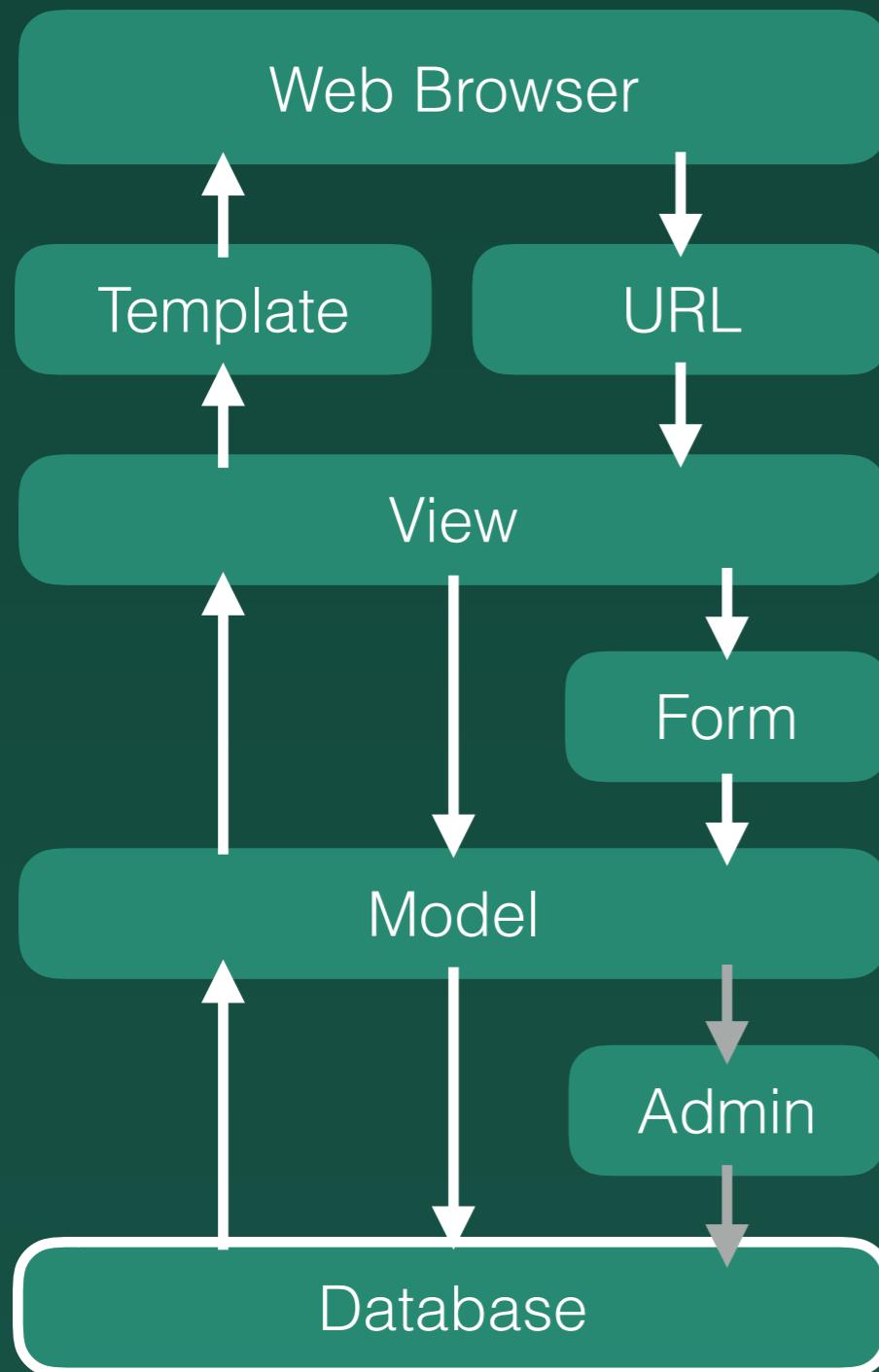
Basic Knowledge



Database

Collection of data

Basic Knowledge

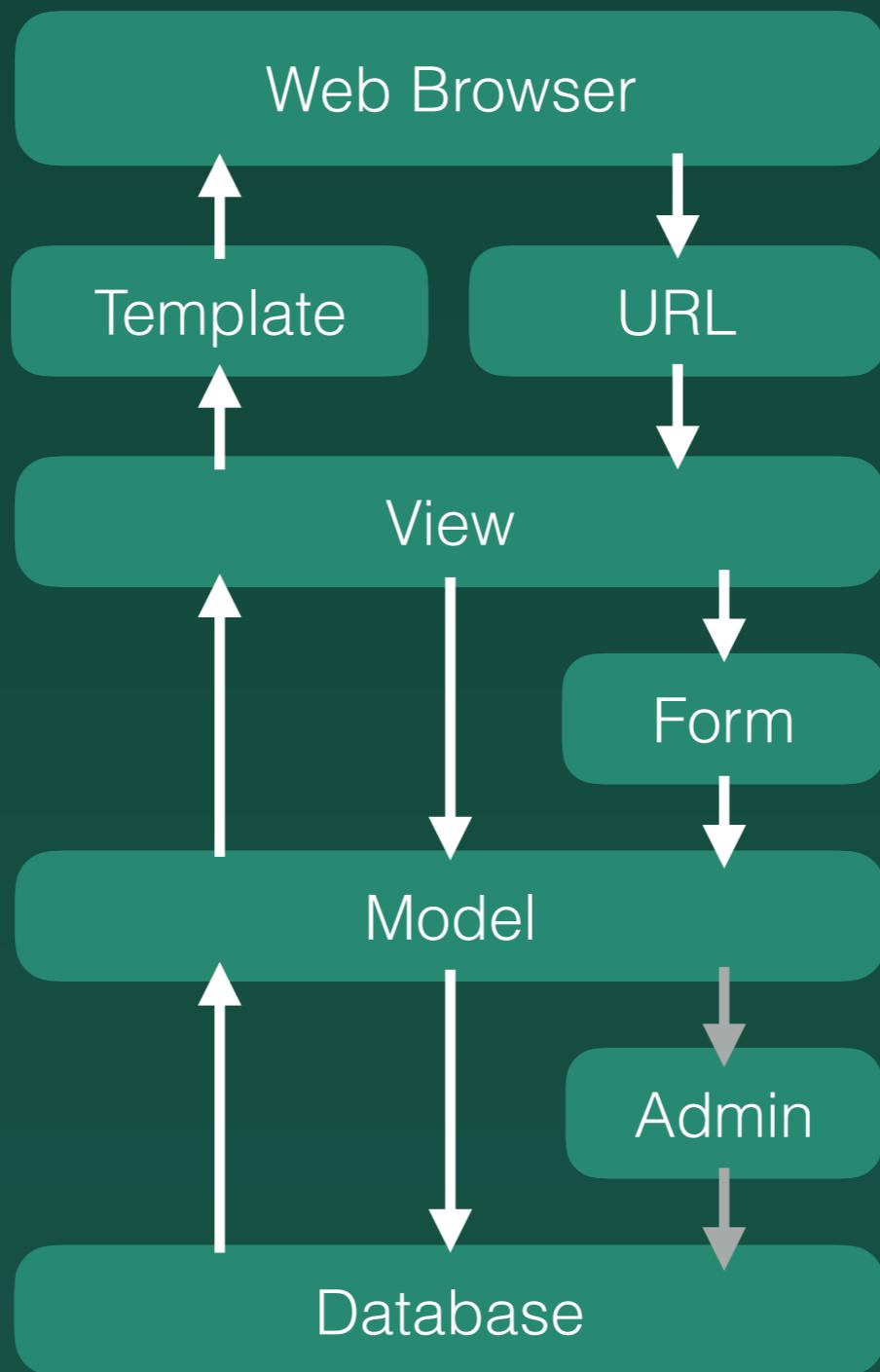


Database

- Collection of data
- Provided with a wonderful back-end platform for easy management

Basic Knowledge

Workflow



Outline

1. Introduction
- 2. Basic Knowledge**
3. Technical Details
4. Tutorial
5. Conclusion

Outline

1. Introduction
2. Basic Knowledge
- 3. Technical Details**
4. Tutorial
5. Conclusion

Technical Details

1. Directory Architecture
2. Module
3. Commands
4. Template Tags
5. High-level URL Configuration

Technical Details

Directory Architecture

project_name/	Container of your entire project, which often referred as ' workspace '
manage.py	The command-line utility to interact with your Django project E.g1. python manage.py help E.g2. python manage.py runserver -h
your_project/	The name of your Django project
__inti__.py	The file required for Python to treat this directory as a package
settings.py	Configuration for this Django project
url.py	Management of URLs to provide mapping to view.py
your_app/	One of the web applications of this Django project
__inti__.py	
migration/	The file which stores all the variations in your database
static/	The file which stores all of your CSS, JS, images
templates/	The file which stores all of your Html
admin.py	It reads your model and provides interface to your database
form.py	It is used to fetch data and performs validation
model.py	Description of the format or structure of an object stored in Database
views.py	All the functions needed to process or respond user's request
db.sqlite3	Your database

Technical Details

1. Directory Architecture
2. Module
3. Commands
4. Template Tags
5. High-level URL Configuration

Technical Details

Module

```
mypackage/  
    __init__.py  
    mymodule.py
```

- A module is a python source file
- A package is a directory of Python module(s)

Technical Details

1. Directory Architecture
2. Module
3. Commands
4. Template Tags
5. High-level URL Configuration

Technical Details

Commands

```
$ python manage.py startproject project_name
```

To start an project

Technical Details

Commands

```
$ python manage.py startapp app_name
```

To start an application

Technical Details

Commands

```
$ python manage.py createsuperuser
```

To create a superuser

Technical Details

Commands

```
$ python manage.py run server 0.0.0:8080
```

To start up your server

Note: 0.0.0:8080 → address:port

Technical Details

Commands

\$ python manage.py makemigrations

To create new migration based on the changes you made in your models

\$ python manage.py migrate

To apply migration into your database

Note: Migration is Django's way of propagating the changes you made into your database schema

Technical Details

Commands

\$ python manage.py -h

To ask for what kind of command can be used

\$ python manage.py runserver -h

To ask for what kind of command can be used

after runserver

Technical Details

1. Directory Architecture
2. Module
3. Commands
4. Template Tags
5. High-level URL Configuration

Technical Details

Template Tags

```
{% for object in objects %}
```

```
    {{ object.attribute }}
```

```
{% endfor %}
```

Django saves you the trouble of repeating your codes

Technical Details

Template Tags

```
{{ form.as_p }}
```

This is Django's way of rendering Html Form

Note: Remember to wrap it within your HTML form tag

Technical Details

Template Tags

`{{ csrf_token }}`

This is used prevent ill-intentioned hacker from
hacking into your database

Note: Remember to wrap it within your HTML form tag

Technical Details

1. Directory Architecture
2. Module
3. Commands
4. Template Tags
5. High-level URL Configuration

Technical Details

High-level URL Configuration

url(r'^\$', views.function)

?P< v > : to take **v** and sent to view as a variable

^ : beginning of the url

\$: end of the url

() : to capture part of the pattern

+ : to indicate the previous item should repeat at least once

{} : to indicate a specific number of repetition

Technical Details

High-level URL Configuration

```
url(r'^anything/(?P<variable>[0-9]+)$', views.function)
```

?P< v > : to take **v** and sent to view as a variable

^ : beginning of the url

\$: end of the url

() : to capture part of the pattern

+ : to indicate the previous item should repeat at least once

{} : to indicate a specific number of repetition

Technical Details

High-level URL Configuration

`url(r'^(?P<variable>[0-9]{n})/$', views.function)`

?P< v > : to take `v` and sent to view as a variable

^ : beginning of the url

\$: end of the url

() : to capture part of the pattern

+ : to indicate the previous item should repeat at least once

{ } : to indicate a specific number of repetition

Technical Details

1. Directory Architecture
2. Module
3. Commands
4. Template Tags
5. High-level URL Configuration

Outline

1. Introduction
2. Basic Knowledge
- 3. Technical Details**
4. Tutorial
5. Conclusion

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
- 4. Tutorial**
5. Conclusion

Tutorial

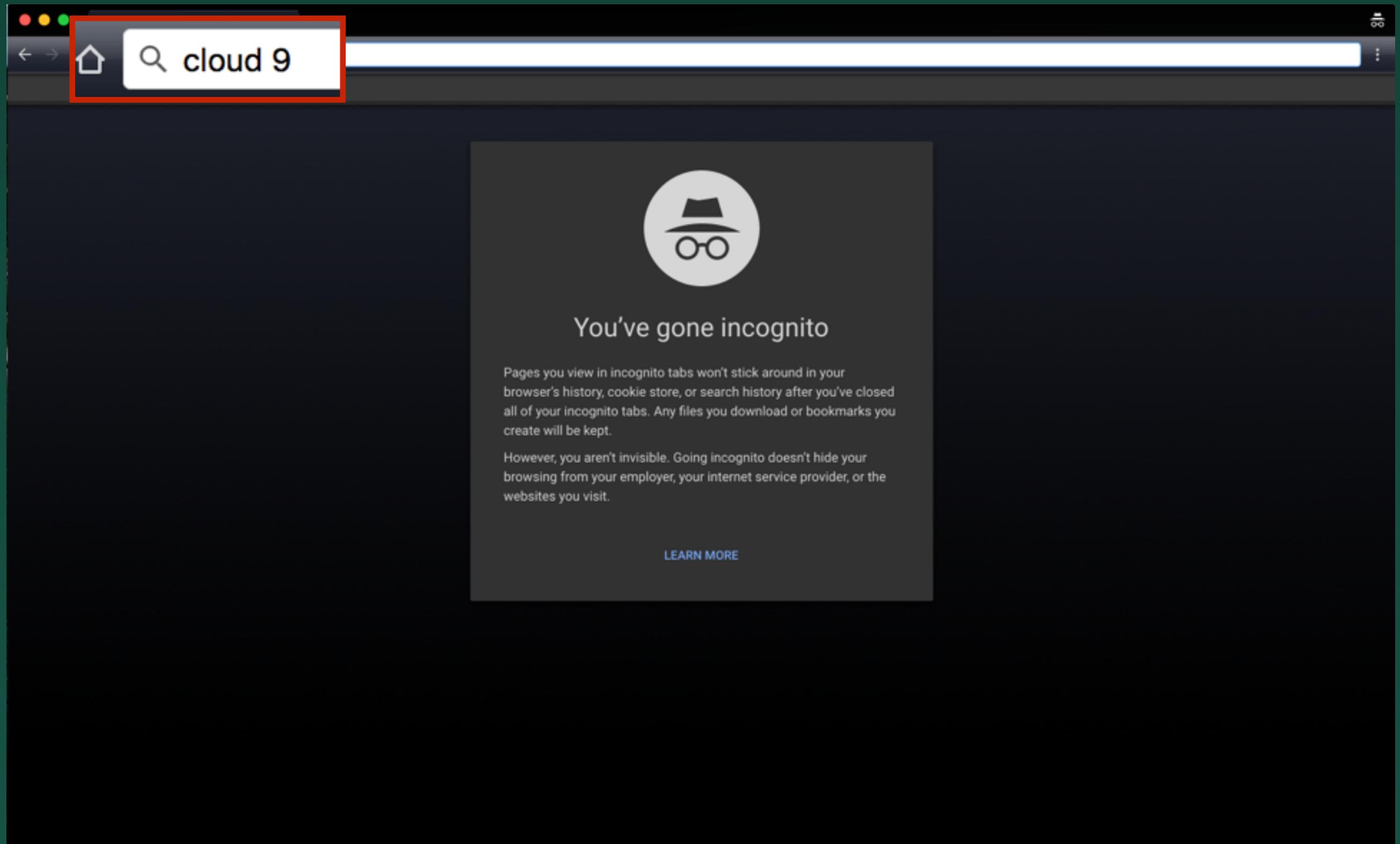
Build an online contact list

https://github.com/Denffer/django_tutorial

Find your Google Chrome



Google “Cloud 9”



Find “Cloud 9”

A screenshot of a Google search results page for the query "cloud 9". The search bar at the top contains "cloud 9". Below the search bar, there are filter options: 全部 (All), 圖片 (Images), 影片 (Videos), 新聞 (News), 地圖 (Maps), 更多 (More), and 搜尋工具 (Search tools). A red box highlights the first search result.

Cloud9 - Your development environment, in the cloud
<https://c9.io/> ▾ 翻譯這個網頁

Cloud9 combines a powerful online code editor with a full Ubuntu workspace in the cloud. Cloud9 supports more than 40 languages, with class A support for ...

cloud9.gg/ ▾ 翻譯這個網頁
Official Cloud9 Website. ... Season 2 Finals | Highlights. Cloud9 and Calle Danielsson are proud to present the StarSeries Season 2 Finals Highlights.

Teams — Cloud9
cloud9.gg/teams/ ▾ 翻譯這個網頁
Official Cloud9 Website. ... Cloud9 · Home · News · Store · Guides · Teams · Streams · Partners · Home · News · Store · Guides · Teams · Streams · Partners.

LOL — Cloud9
cloud9.gg/lol-na-cloud9/ ▾ 翻譯這個網頁
Official Cloud9 Website. ... COD · CS:GO · HS · LOL · LOL Challenger · OW · SSBM · Streamers · Smash4 · Vainglory. Cloud 9 HyperX Team Banner.png.

Cloud9 - Leaguepedia - Competitive League of Legends eSports Wiki
lol.gamepedia.com/Cloud9 ▾ 翻譯這個網頁
2016年10月15日 - Cloud9 (C9) is a North American League of Legends team competing in the North American League Championship Series. The team is ...
History · Timeline · Player Roster · Organization

Cloud9 | LoL Esports
www.lolesports.com/en_US/worlds/world_championship.../cloud9 ▾ 翻譯這個網頁

Click on Github logo

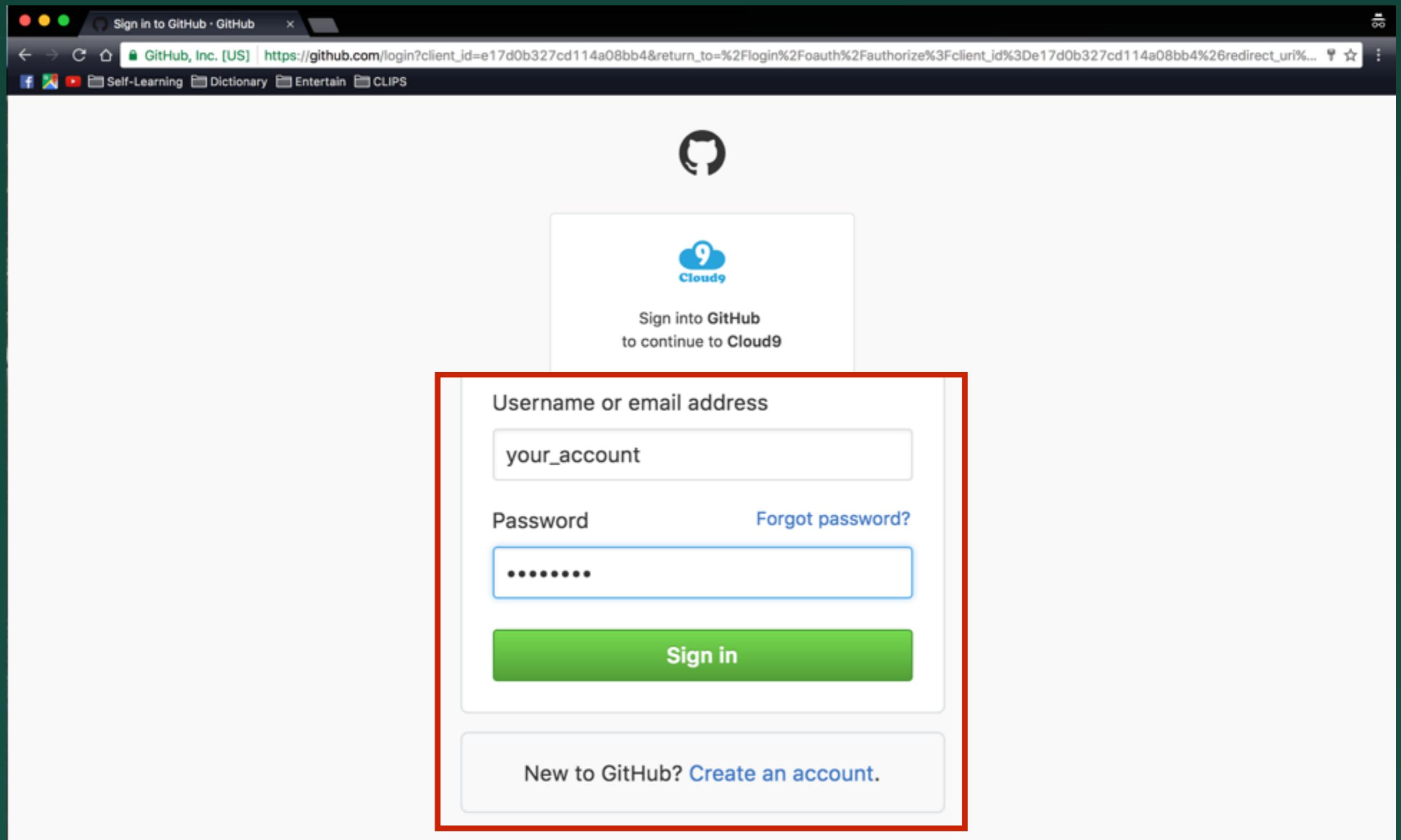
The screenshot shows a Cloud9 IDE interface. At the top, there's a navigation bar with links for PRICING, EDUCATION, SALESFORCE, COMMUNITY, BLOG, and SIGN IN. The SIGN IN button is highlighted with a red box. Below the navigation bar, a large heading says "Language tools that make you smarter". A sub-section below it states "Cloud9 supports more than 40 languages, with class A support for PHP, Ruby, Python, JavaScript, Go, and more". There are input fields for "Your e-mail address" and a "Sign Up" button.

The main workspace shows a file named "edit_session.js" with the following code:

```
151     this.$foldData = [];
152     this.$foldData.toString = function() {
153         return this.join("\n");
154     }
155     this.on("changeFold", this.onChangeFold.bind(this));
156     this.$onChange = this.onChange.bind(this);
157
158     if (typeof text != "object" || !text.getLine)
159         text = new Document(text);
160
161     this.setDocument(text);
162     this.selection = new Selection(this);
```

A tooltip appears over the line "this.selection = new Selection(this);", indicating an "Undeclared variable". The tooltip lists several suggestions from a dropdown menu, including "EditSession(text, mode)", "toString()", "this.changeFold", "this.setDocument(doc)", "this.getDocument()", "this.\$resetRowCache(docRow)", "this.\$getRowCacheIndex(cacheArray, val)", "this.resetCaches()", "this.onChangeFold(e)", "this.onChange(e)", "this.setValue(text)", "this.getValue", "this.toString()", and "this.getSelection()".

Login your Github



Create a new workspace

The screenshot shows a web browser window for the c9.io Denffer workspace. The URL in the address bar is <https://c9.io/denffer>. The page displays a list of workspaces and a central 'Create a new workspace' button.

Denffer (Profile Picture)

Workspaces

- Shared With Me
- Repositories

YOUR INDIVIDUAL SUBSCRIPTIONS

Free [Upgrade](#)

Create a new workspace

+ (Large Red Box)

django_contacts
python workspace
Add a description [here](#).
dated a few seconds ago.
[Clone](#) [Open](#)

tripadvisor1
Add a description [here](#).
Updated 13 days ago.
[Clone](#) [Open](#)

easy_map_test
Cloned from kmike/django-easy-maps
Add a description [here](#).
Updated 5 months ago.
[Clone](#) [Open](#)

lookhere
python workspace
Add a description [here](#).
Updated 10 months ago.
[Clone](#) [Open](#)

Create a new workspace

1.

Workspace name

my_project

option of your workspace

Salesforce

Private

This is a workspace for your eyes only

Public

This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional)

e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Choose a template

HTML5

Node.js

PHP, Apache ...

CSS

Wordpress

Balsamiq Mockups

2.

django

Django

3.

Create workspace

Welcome to Django on Cloud 9

The screenshot shows the Cloud 9 IDE interface. At the top, a browser window displays the URL https://ide.c9.io/denffer/my_project#openfile-README.md. The main workspace shows a file named README.md with the content "Django on Cloud 9". Below this, a welcome message reads: "Welcome to your Django project on Cloud9 IDE! Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '[<https://my-project-denffer.c9users.io/>](https://my-project-denffer.c9users.io/)' and the admin page from '[<https://my-project-denffer.c9users.io/admin>](https://my-project-denffer.c9users.io/admin)'". A terminal window at the bottom shows the command \$ python manage.py migrate.

Cloud9 File Edit Find View Goto Run Tools Window Support Preview Run Project MEMORY CPU DISK Share ...

Workspace my_project / README.md +

Commands Navigate

Collaborate Outline Debugger

Welcome to your Django project on Cloud9 IDE!

Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '<https://my-project-denffer.c9users.io/>' and the admin page from '<https://my-project-denffer.c9users.io/admin>'.

Starting from the Terminal

In case you want to run your Django application from the terminal just run:

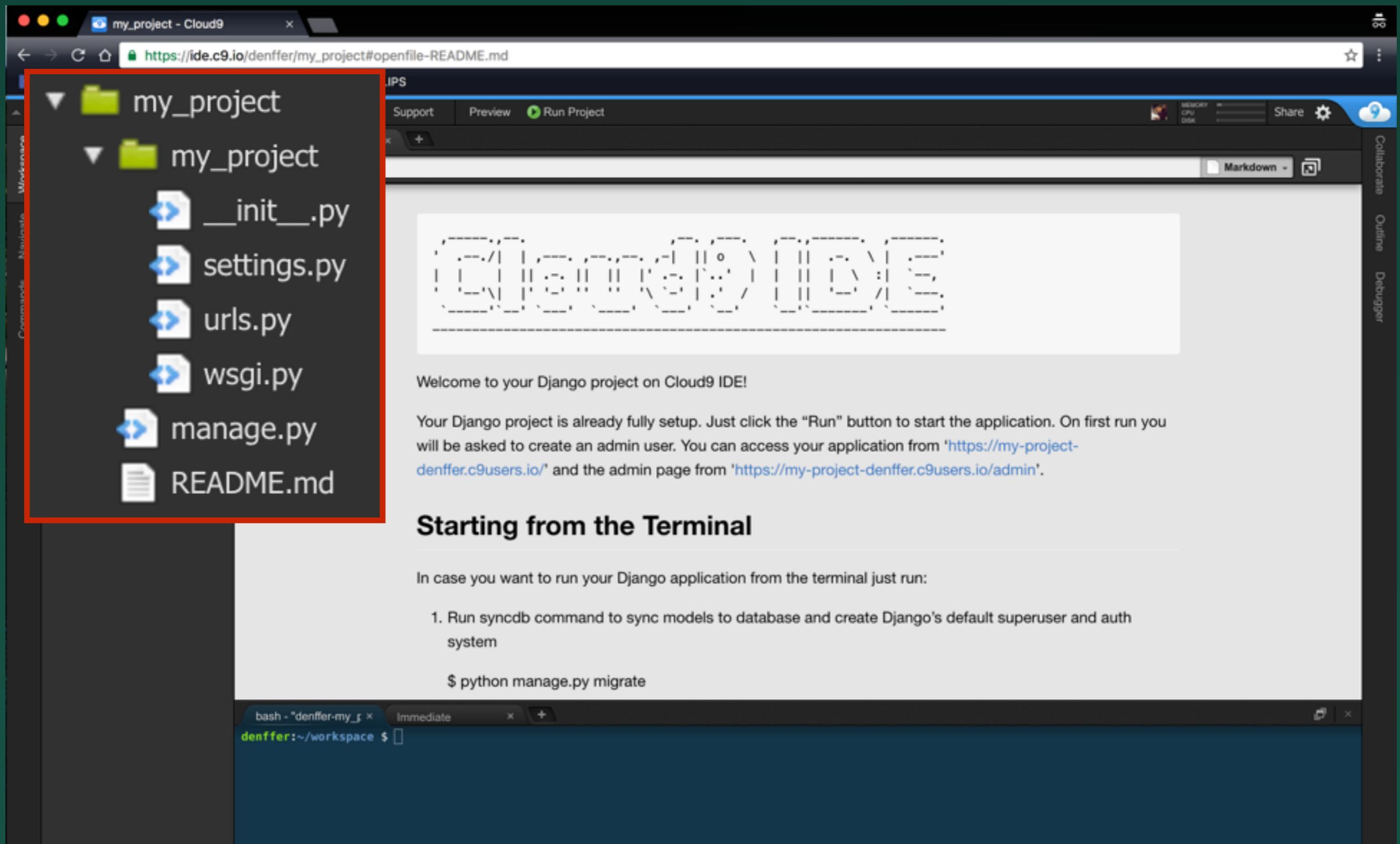
1. Run syncdb command to sync models to database and create Django's default superuser and auth system

```
$ python manage.py migrate
```

bash - "denffer-my_5" x Immediate x +

denffer:~/workspace \$

Click on 'my_contacts' and see what are the things initially given by Django



Anything beyond these should be added or created by yourself

The screenshot shows the Cloud9 IDE interface with a Django project named 'my_project'. A red box highlights the project structure on the left, which includes files like `__init__.py`, `settings.py`, `urls.py`, `wsgi.py`, and `manage.py`. The main workspace displays a welcome message for the Django project, indicating it is fully setup and providing URLs for access. Below this, a section titled 'Starting from the Terminal' provides instructions for running the application via the terminal. The terminal window at the bottom shows a command prompt with the user's name and workspace path.

my_project - Cloud9

https://ide.c9.io/denffer/my_project#openfile-README.md

IPS

Support Preview Run Project

my_project

my_project

__init__.py

settings.py

urls.py

wsgi.py

manage.py

README.md

Welcome to your Django project on Cloud9 IDE!

Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '<https://my-project-denffer.c9users.io/>' and the admin page from '<https://my-project-denffer.c9users.io/admin>'.

Starting from the Terminal

In case you want to run your Django application from the terminal just run:

1. Run syncdb command to sync models to database and create Django's default superuser and auth system

```
$ python manage.py migrate
```

bash - "denffer-my_5" x Immediate x +

denffer:~/workspace \$

Once your workspace is created on Cloud 9,
your project is created as well.

So there is **NO** need for you to

```
$ python manage.py startproject my_project
```

Now, create your first Django application

The screenshot shows the Cloud9 IDE interface. The top navigation bar includes tabs for File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run Project. The left sidebar has sections for Workspace, Navigate, and Commands. The workspace shows a file tree for a 'my_project' directory containing __init__.py, settings.py, urls.py, wsgi.py, manage.py, and README.md. A central editor window displays the contents of README.md, which includes a welcome message and setup instructions for a Django project. Below the editor is a terminal window with a red box highlighting the command '\$ python manage.py startapp mycontacts'. The terminal also shows the prompt 'bash - denffer\$'.

Welcome to your Django project on Cloud9 IDE!

Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '<https://my-project-denffer.c9users.io/>' and the admin page from '<https://my-project-denffer.c9users.io/admin>'.

Starting from the Terminal

In case you want to run your Django application from the terminal just run:

1. Run syncdb command to sync models to database and create Django's default superuser and auth system

```
$ python manage.py migrate
```

```
$ python manage.py startapp mycontacts
```

There you go. Your first Django application

The screenshot shows the Cloud9 IDE interface. On the left, the workspace sidebar displays a folder named 'mycontacts' which is highlighted with a red box. The main workspace shows a terminal window at the bottom with the command '\$ python manage.py startapp mycontacts' and its output. Above the terminal, there is a welcome message for a Django project. The top navigation bar shows the project name 'my_project - Cloud9' and the URL 'https://ide.c9.io/denffer/my_project#openfile-README.md'. The top right corner includes memory, CPU, and disk usage monitors.

Welcome to your Django project on Cloud9 IDE!

Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '<https://my-project-denffer.c9users.io/>' and the admin page from '<https://my-project-denffer.c9users.io/admin>'.

Starting from the Terminal

In case you want to run your Django application from the terminal just run:

1. Run syncdb command to sync models to database and create Django's default superuser and auth system

```
$ python manage.py migrate
```

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Check on what is inside

The screenshot shows the Cloud9 IDE interface with a Django project named "my_project". A red box highlights the project structure in the left sidebar, which includes a "mycontacts" app folder containing "migrations", "__init__.py", "admin.py", "apps.py", "models.py", "tests.py", and "views.py". The main workspace displays a welcome message for the Django project, instructions for starting from the terminal, and configuration options. A terminal window at the bottom shows the command \$ python manage.py startapp mycontacts being run.

my_contacts

- migrations
- __init__.py
- admin.py
- apps.py
- models.py
- tests.py
- views.py

Welcome to your Django project on Cloud9 IDE!

Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '<https://my-project-denffer.c9users.io/>' and the admin page from '<https://my-project-denffer.c9users.io/admin>'.

Starting from the Terminal

In case you want to run your Django application from the terminal just run:

1. Run syncdb command to sync models to database and create Django's default superuser and auth system
\$ python manage.py migrate
2. Run Django
\$ python manage.py runserver \$IP:\$PORT

Configuration

You can configure your Python version and `PYTHONPATH` used in Cloud9 > Preferences > Project Settings >

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

See that? Your application is actually a package, and it contains the modules we've seen earlier.

The screenshot shows the Cloud9 IDE interface. On the left, the file tree displays a Django project structure with a red box highlighting the 'mycontacts' app directory. Inside 'mycontacts', there are files: migrations, __init__.py, admin.py, apps.py, models.py, tests.py, and views.py. The main workspace shows a welcome message for a Django project on Cloud9 IDE. It provides instructions for running the application from the terminal, including commands like syncdb and runserver. Below this, a configuration section allows users to set Python version and PATH. At the bottom, a terminal window shows the command \$ python manage.py startapp mycontacts being run.

my_contacts

migrations

__init__.py

admin.py

apps.py

models.py

tests.py

views.py

Welcome to your Django project on Cloud9 IDE!

Your Django project is already fully setup. Just click the "Run" button to start the application. On first run you will be asked to create an admin user. You can access your application from '<https://my-project-denffer.c9users.io/>' and the admin page from '<https://my-project-denffer.c9users.io/admin>'.

Starting from the Terminal

In case you want to run your Django application from the terminal just run:

1. Run syncdb command to sync models to database and create Django's default superuser and auth system
\$ python manage.py migrate
2. Run Django
\$ python manage.py runserver \$IP:\$PORT

Configuration

You can configure your Python version and `PYTHONPATH` used in Cloud9 > Preferences > Project Settings >

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Open your urls.py

The screenshot shows the Cloud9 IDE interface. In the top left, there's a browser window with the URL https://ide.c9.io/denffer/my_project#openfile-README.md. The main area is the workspace, which displays a file tree for a project named 'my_project'. A context menu is open over the 'urls.py' file, with the 'Open' option highlighted. To the right of the workspace, a preview window shows the contents of 'urls.py' and 'wsgi.py', both with a large red 'Open' button overlaid. Below the workspace, a terminal window titled 'bash - "denffer-my_..."' shows the command: `denffer:~/workspace $ python manage.py startapp mycontacts`.

Starting from the Terminal

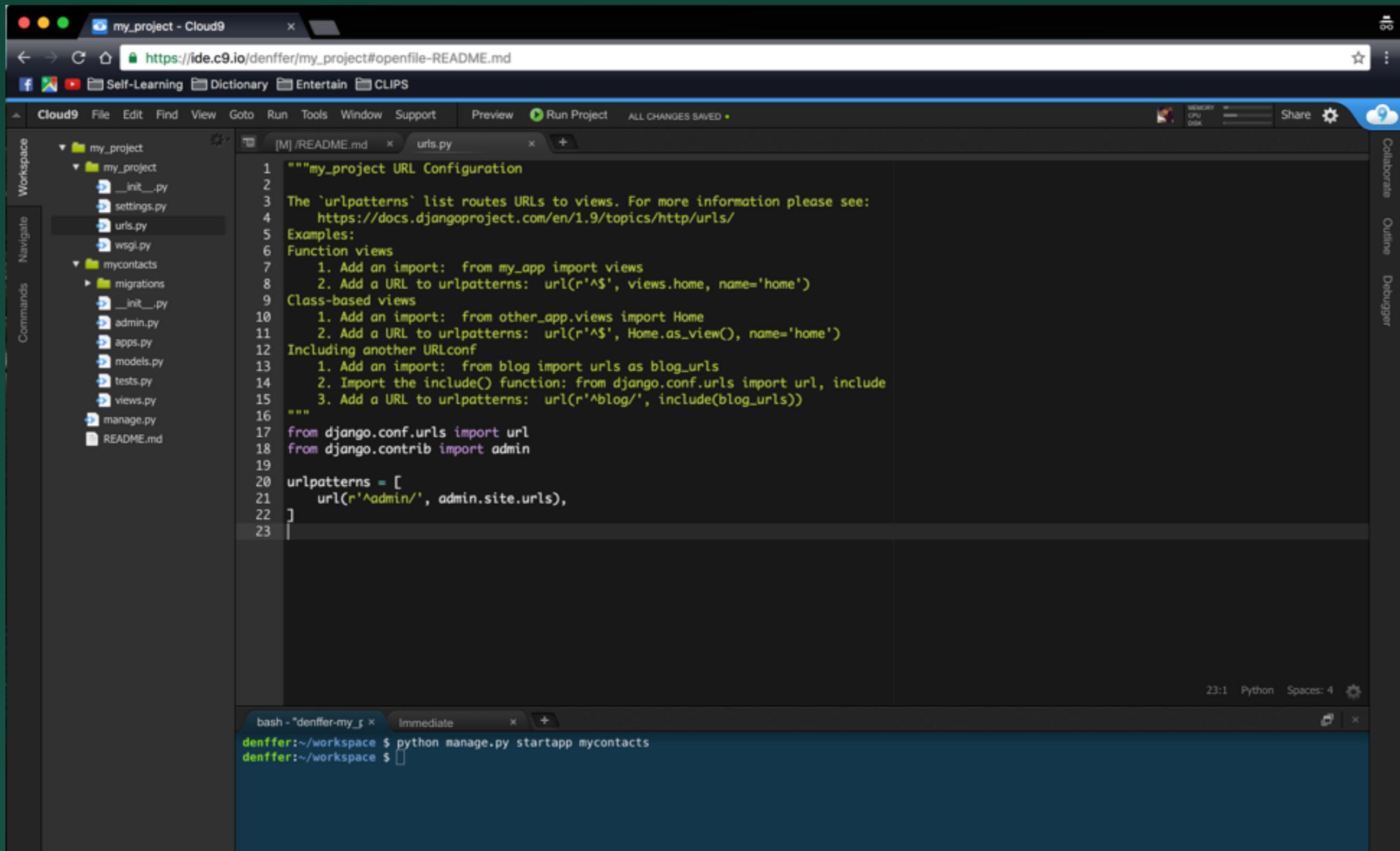
In case you want to run your Django application from the terminal just run:

1. Run syncdb command to sync models to database and create Django's default superuser and auth system
\$ python manage.py migrate
2. Run Django
\$ python manage.py runserver \$IP:\$PORT

Configuration

You can configure your Python version and `PYTHONPATH` used in Cloud9 > Preferences > Project Settings >

The default url given by Django is the web address connecting to your back-end management platform



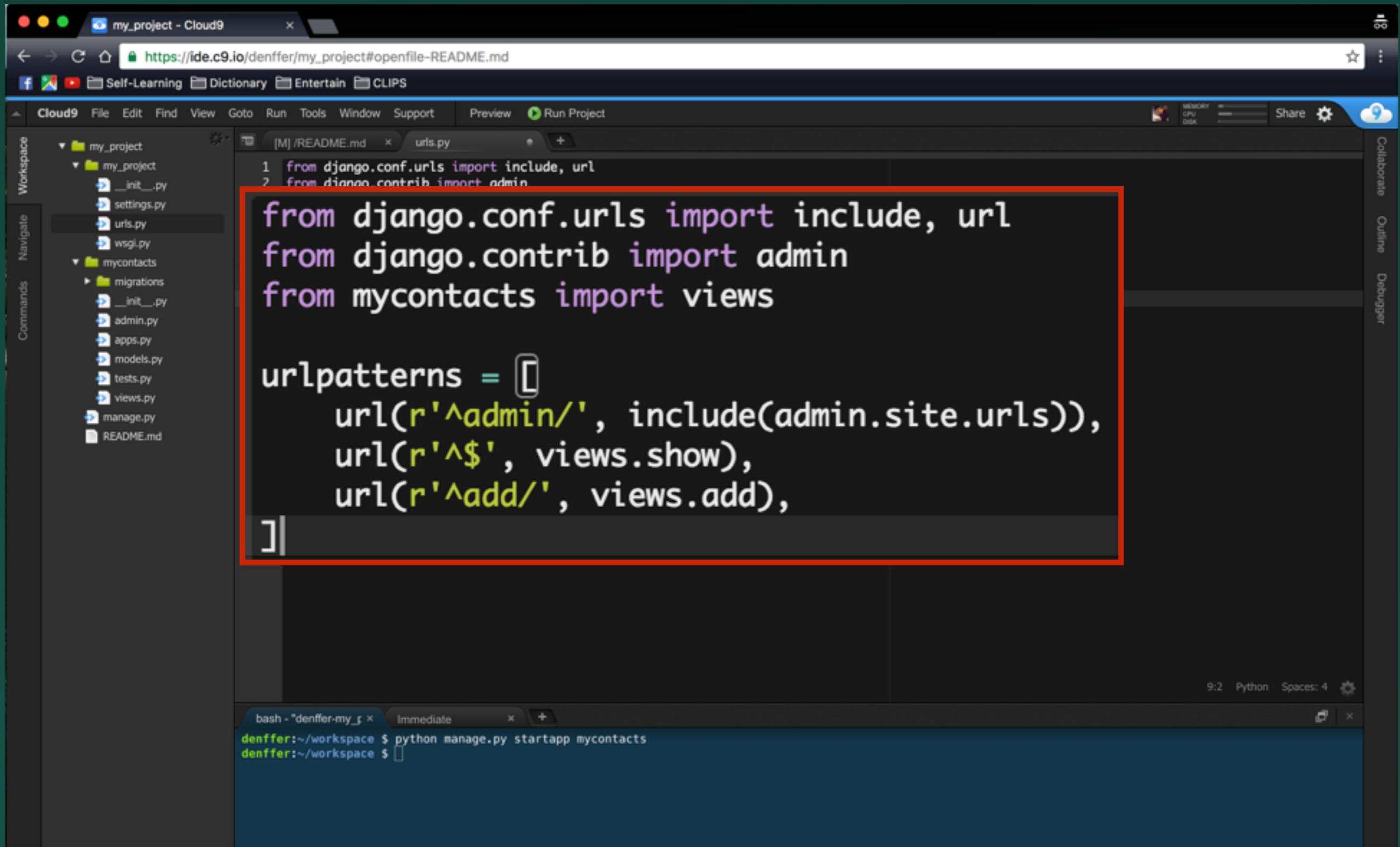
The screenshot shows the Cloud9 IDE interface with a Django project named "my_project". The project structure on the left includes "my_project" (containing __init__.py, settings.py, urls.py, and wsgi.py), "mycontacts" (containing migrations, __init__.py, admin.py, apps.py, models.py, tests.py, views.py, and manage.py), and README.md.

The main editor window displays the contents of urls.py:

```
1 """my_project URL Configuration
2
3     The `urlpatterns` list routes URLs to views. For more information please see:
4         https://docs.djangoproject.com/en/1.9/topics/http/urls/
5     Examples:
6         Function based views
7             1. Add an import: from my_app import views
8                 2. Add a URL to urlpatterns: url(r'^$', views.home, name='home')
9
10    Class-based views
11        1. Add an import: from other_app.views import Home
12            2. Add a URL to urlpatterns: url(r'^$', Home.as_view(), name='home')
13
14    Including another URLconf
15        1. Add an import: from blog import urls as blog_urls
16            2. Import the include() function: from django.conf.urls import url, include
17            3. Add a URL to urlpatterns: url(r'^blog/', include(blog_urls))
18
19
20    from django.conf.urls import url
21    from django.contrib import admin
22
23    urlpatterns = [
24        url(r'^admin/', admin.site.urls),
25    ]
```

At the bottom, a terminal window shows the command: `denffer:~/workspace $ python manage.py startapp mycontacts`.

Now, copy and paste the codes in **urls.py** from the repository you cloned from Github



The screenshot shows the Cloud9 IDE interface. The top bar displays the project name "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The left sidebar shows the project structure:

- my_project (selected)
- my_project (containing __init__.py, settings.py, urls.py, wsgi.py)
- mycontacts (containing migrations, __init__.py, admin.py, apps.py, models.py, tests.py, views.py)
- manage.py
- README.md

The main editor window shows the contents of "urls.py":

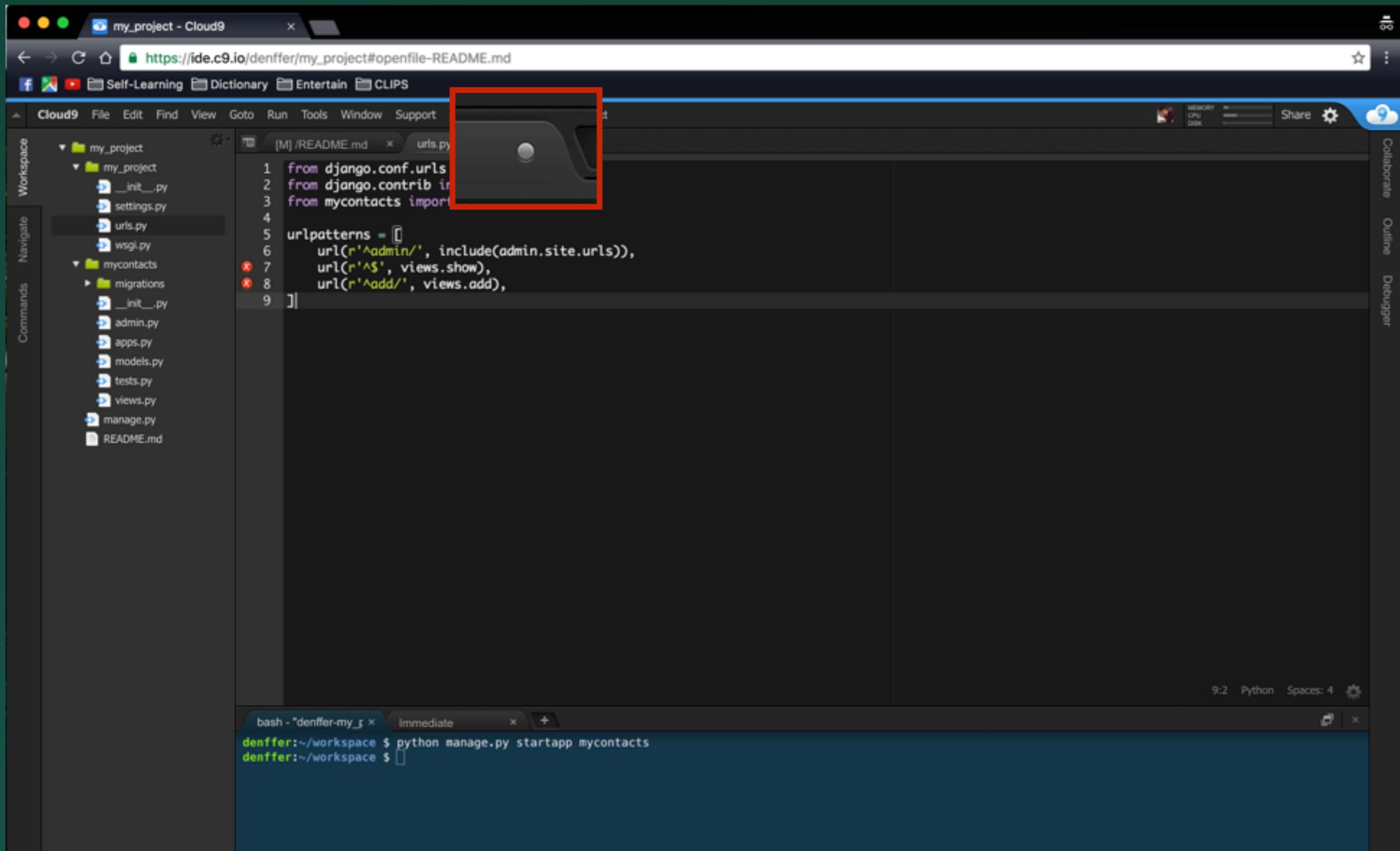
```
from django.conf.urls import include, url
from django.contrib import admin
from mycontacts import views

urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^$', views.show),
    url(r'^add/$', views.add),
]
```

A red box highlights the entire code block in the editor. Below the editor is a terminal window titled "bash - denffer-my_5" showing the command: "denffer:~/workspace \$ python manage.py startapp mycontacts".

Do you see the dot over here?

This means that the file is **NOT** saved

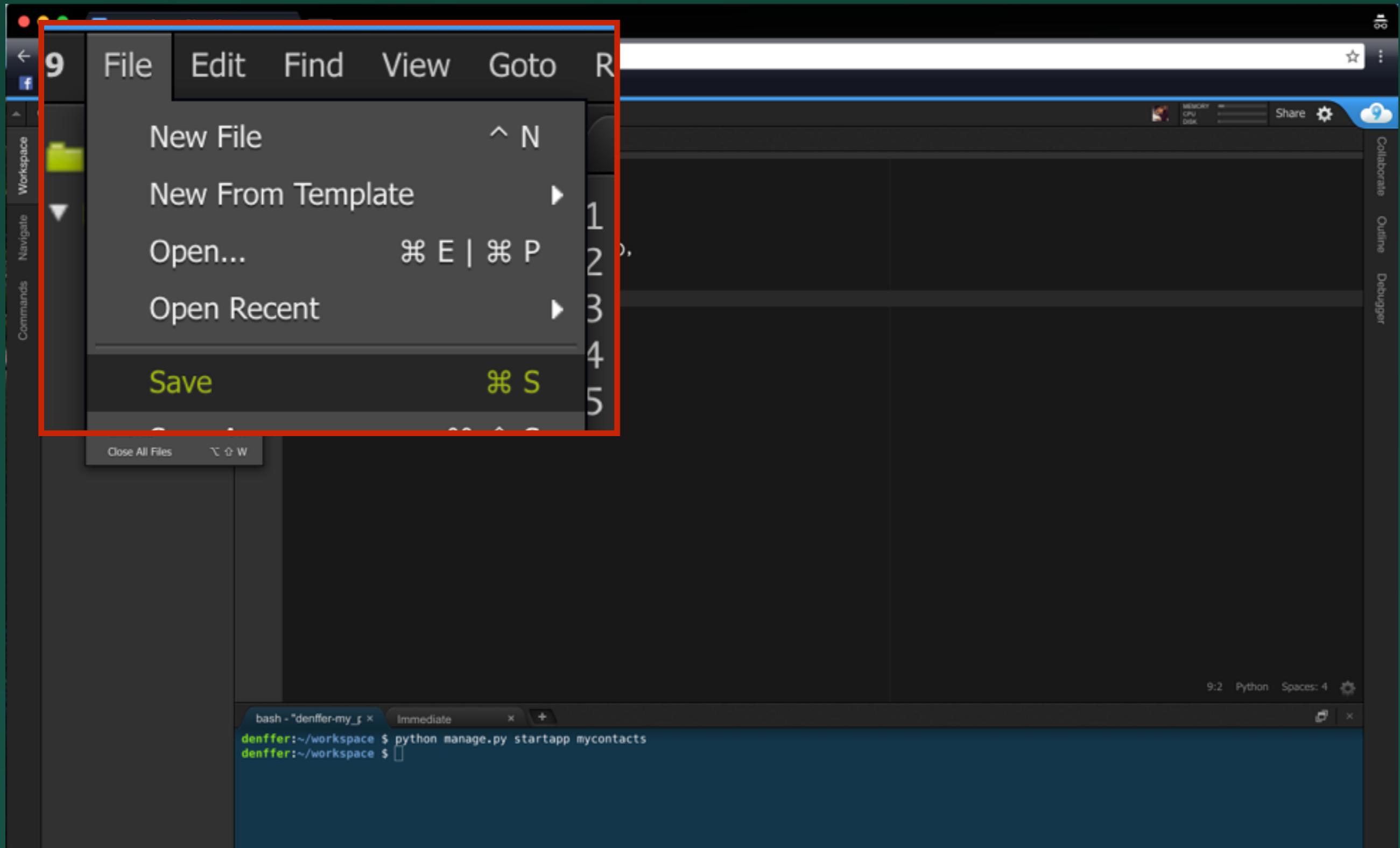


The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The left sidebar has sections for "Workspace", "Navigate", and "Commands". The workspace shows a project structure with files like __init__.py, settings.py, urls.py, wsgi.py, mycontacts, migrations, admin.py, apps.py, models.py, tests.py, views.py, manage.py, and README.md. The "urls.py" file is open in the main editor area, containing Python code for Django URLs. A red box highlights the small circular save icon located at the top right of the code editor window. Below the editor is a terminal window showing the command "python manage.py startapp mycontacts" being run. The bottom status bar indicates "9:2 Python Spaces: 4".

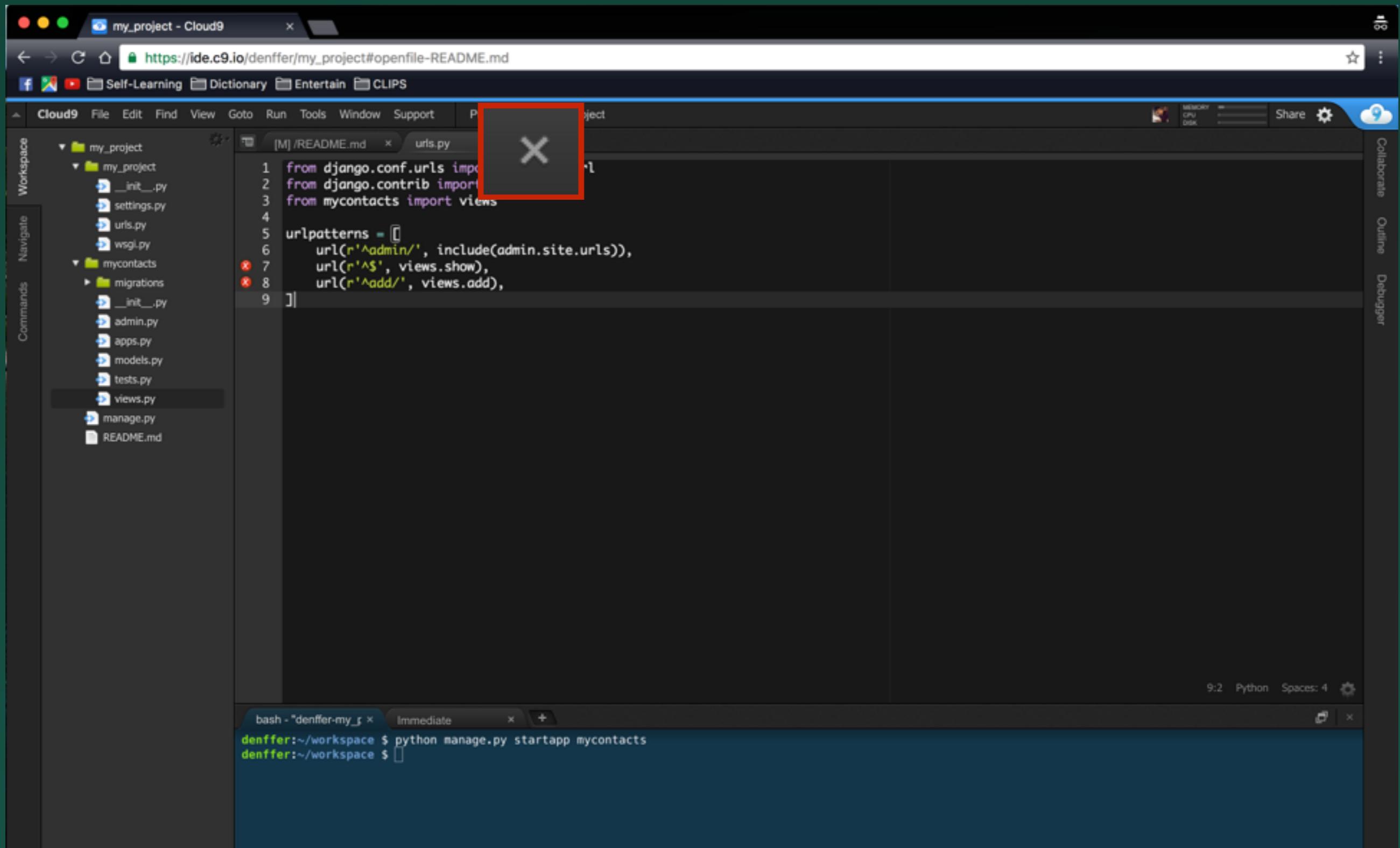
```
from django.conf.urls
from django.contrib import
from mycontacts import
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^$', views.show),
    url(r'^add/$', views.add),
]
```

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

From here on, remember to save your file after every edition



Great! The cross shown here means that you have already saved the file



The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The workspace sidebar on the left lists the project structure:

- my_project (selected)
- my_project (containing __init__.py, settings.py, urls.py, wsgi.py)
- mycontacts (containing migrations, __init__.py, admin.py, apps.py, models.py, tests.py, views.py)
- manage.py
- README.md

The main editor window shows the content of "urls.py":

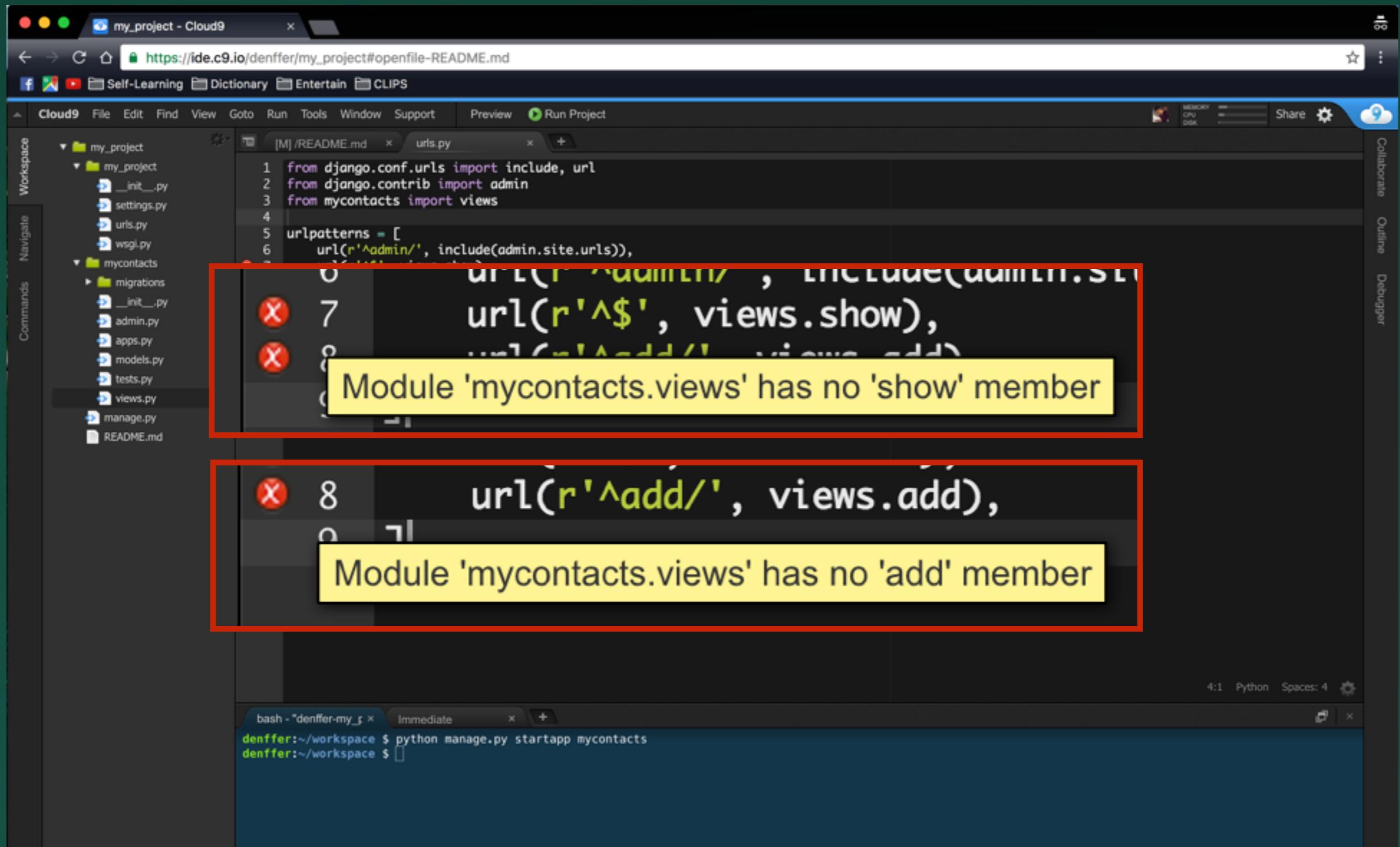
```
from django.conf.urls import *
from django.contrib import *
from mycontacts import views

urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^$', views.show),
    url(r'^add/$', views.add),
]
```

A red box highlights the small gray square with a white "X" icon in the top right corner of the code editor, indicating the file has been saved.

At the bottom, a terminal window titled "bash - denffer-my_5" shows the command: "denffer:~/workspace \$ python manage.py startapp mycontacts".

Whoops! It says there are some problems with your “View”



The screenshot shows a Cloud9 IDE interface with a Python project named "my_project". The "urls.py" file is open, displaying the following code:

```
from django.conf.urls import include, url
from django.contrib import admin
from mycontacts import views

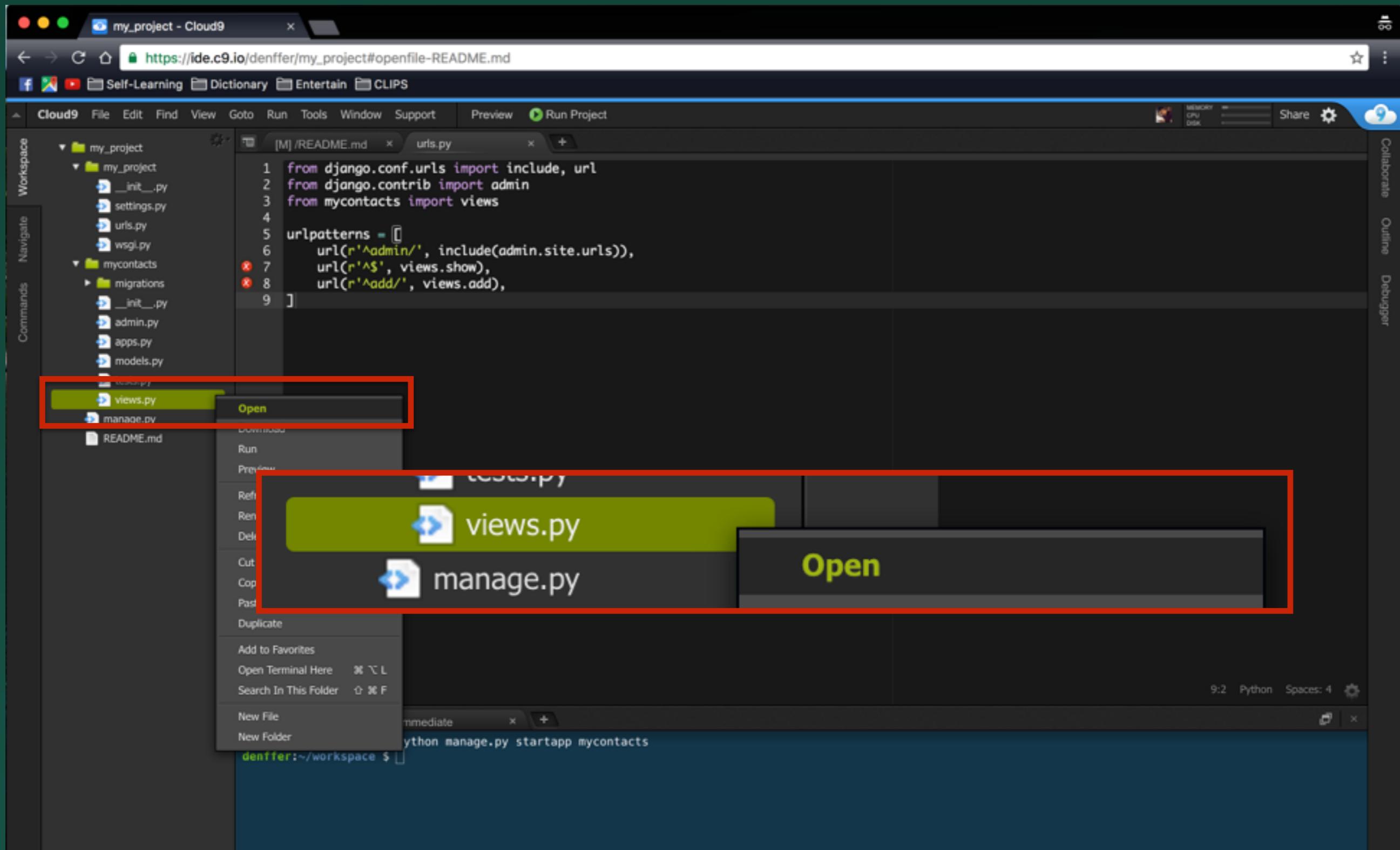
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^$', views.show),
    url(r'^add/$', views.add),
```

Two errors are highlighted with red circles and crossed-out lines:

- Line 7: `url(r'^$', views.show),` - Error message: "Module 'mycontacts.views' has no 'show' member"
- Line 8: `url(r'^add/$', views.add),` - Error message: "Module 'mycontacts.views' has no 'add' member"

The Cloud9 interface includes a sidebar with "Workspace", "Navigate", and "Commands" tabs, and a bottom navigation bar with "bash - denffer-my_5", "Immediate", and "File" tabs.

Let's deal with it. Now go to `views.py`



Remember? This is where you put all the functions

The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run Project. The workspace sidebar on the left lists the project structure:

- my_project (selected)
- my_project (subdir)
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
- mycontacts (subdir)
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - views.py (selected)
- manage.py
- README.md

The main editor area shows the contents of the "views.py" file:

```
1 from django.shortcuts import render
2
3 # Create your views here.
4
```

The bottom terminal window shows the command "python manage.py startapp mycontacts" being run in a bash shell:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Now, copy and paste the codes in `views.py` from the repository you cloned from Github

The screenshot shows the Cloud9 IDE interface. The left sidebar displays the project structure:

```
my_project
  my_project
    __init__.py
    settings.py
    urls.py
    wsgi.py
  mycontacts
    migrations
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    views.py
  manage.py
  README.md
```

The main workspace shows the contents of `views.py`:

```
1  from django.shortcuts import render
2  from .forms import AddForm
3  from .models import Contact
4  from django.http import HttpResponseRedirect
5
6  def show(request):
7      """
8          This function gets all the members in your Database through your Model
9          Any further usage please refer to: https://docs.djangoproject.com/en/1.10/ref/models/querysets/
10         """
11     contact_list = Contact.objects.all()
12     return render(request, 'mycontacts/show.html',{'contacts': contact_list})
13
14 def add(request):
15     """
16         This function is called to add one contact member to your contact list in your Database """
17     if request.method == 'POST':
18
19         django_form = AddForm(request.POST)
20         if django_form.is_valid():
21
22             """
23                 Assign data in Django Form to local variables """
24             new_member_name = django_form.data.get("name")
25             new_member_relation = django_form.data.get("relation")
26             new_member_phone = django_form.data.get('phone')
27             new_member_email = django_form.data.get('email')
28
29             """
30                 This is how your model connects to database and create a new member """
31             Contact.objects.create(
32                 name = new_member_name,
33                 relation = new_member_relation,
34                 phone = new_member_phone,
35                 email = new_member_email,
36             )
```

The bottom terminal window shows the command `python manage.py startapp mycontacts` being run.

Remember to save your file

The screenshot shows the Cloud9 IDE interface with a Django project named "my_project". The workspace sidebar on the left lists files and folders: README.md, manage.py, my_contacts (migrations, __init__.py, admin.py, apps.py, models.py, tests.py, views.py), my_project (__init__.py, settings.py, urls.py, wsgi.py), and README.md. The main editor window displays "views.py" with Python code for a Django application. The code defines two functions: "show" and "add". The "show" function retrieves all contacts from the database and renders them. The "add" function handles POST requests to add new contacts, extracting data from a Django form and saving it to the database. Below the editor is a terminal window showing the command "python manage.py startapp my_contacts" being run. The top bar shows the URL "https://ide.c9.io/denffer/my_project#openfile-README.md" and various Cloud9 navigation icons.

```
from django.shortcuts import render
from .forms import AddForm
from .models import Contact
from django.http import HttpResponseRedirect

def show(request):
    """
    This function gets all the members in your Database through your Model
    Any further usage please refer to: https://docs.djangoproject.com/en/1.10/ref/models/querysets/
    """
    contact_list = Contact.objects.all()
    return render(request, 'mycontacts/show.html', {'contacts': contact_list})

def add(request):
    """
    This function is called to add one contact member to your contact list in your Database """
    if request.method == 'POST':
        django_form = AddForm(request.POST)
        if django_form.is_valid():

            """
            Assign data in Django Form to local variables """
            new_member_name = django_form.data.get("name")
            new_member_relation = django_form.data.get("relation")
            new_member_phone = django_form.data.get('phone')
            new_member_email = django_form.data.get('email')

            """
            This is how your model connects to database and create a new member """
            Contact.objects.create(
                name = new_member_name,
                relation = new_member_relation,
                phone = new_member_phone,
                email = new_member_email,
            )
```

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Whoops! It says there are some problems with your “Model”

The screenshot shows a Cloud9 IDE interface with a Python project named "my_project". The "views.py" file is open, displaying the following code:

```
1 from django.shortcuts import render
2
3 from .models import Contact
4
5 contact_list = Contact.objects.all()
6 return render(request, 'mycontacts/show.html', {'contacts': contact_list})
7
8 def add(request):
9     """ This function is called to add one contact member to your contact list in your Database """
10    if request.method == 'POST':
11
12        django_form = AddForm(request.POST)
13        if django_form.is_valid():
14
15            """ Assign data in Django Form to local variables """
16            new_member_name = django_form.data.get("name")
17            new_member_relation = django_form.data.get("relation")
18            new_member_phone = django_form.data.get('phone')
19            new_member_email = django_form.data.get('email')
20
21            """ This is how your model connects to database and create a new member """
22            Contact.objects.create(
23                name = new_member_name,
24                relation = new_member_relation,
25                phone = new_member_phone,
26                email = new_member_email,
27            )
28
29
30
31
32
33
```

A red box highlights the line `from .models import Contact`, which is underlined in purple. A yellow tooltip window appears over this line with the text: "No name 'Contact' in module 'mycontacts..models'". The status bar at the bottom right shows "1:36 Python Spaces: 4".

In the bottom left terminal window, the command `python manage.py startapp mycontacts` is run, resulting in the output:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Let's deal with it. Now go to models.py

The screenshot shows the Cloud9 IDE interface. The file browser on the left lists files and folders under 'my_project'. The 'models.py' file is selected and has a context menu open, with the 'Open' option highlighted. The code editor in the center displays Python code for a Django application. The terminal at the bottom shows the command to start a new app named 'mycontacts'.

```
from django.shortcuts import render
from .forms import AddForm
from .models import Contact
from django.http import HttpResponseRedirect

def show(request):
    """
    This function gets all the members in your Database through your Model
    Any further usage please refer to: https://docs.djangoproject.com/el/1.10/ref/models/querysets/
    """
    contact_list = Contact.objects.all()
    return render(request, 'mycontacts/show.html', {'contacts': contact_list})
```

This is how your model connects to database and create a new member ""

```
Contact.objects.create(
    name = new_member_name,
    relation = new_member_relation,
    phone = new_member_phone,
    email = new_member_email,
)
```

denffer:~/workspace \$ python manage.py startapp mycontacts
denffer:~/workspace \$

Remember? You have to construct your model so it can really connects to your database

The screenshot shows the Cloud9 IDE interface with a dark theme. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run Project. The workspace sidebar on the left lists the project structure:

- my_project (selected)
- my_project (containing __init__.py, settings.py, urls.py, wsgi.py)
- mycontacts (containing migrations, __init__.py, admin.py, apps.py, models.py, tests.py, views.py)
- manage.py
- README.md

The main editor area shows the contents of the "models.py" file:

```
1 from __future__ import unicode_literals
2
3 from django.db import models
4
5 # Create your models here.
```

The bottom terminal window shows the command "python manage.py startapp mycontacts" being run in a bash session:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Now, copy and paste the codes in **models.py** from the repository you cloned from Github

The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run Project. The workspace sidebar shows the project structure:

- my_project
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
- mycontacts
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - views.py
- manage.py
- README.md

The main editor area shows the content of models.py:

```
1 From django.db import models
2
3 class Contact(models.Model):
4     """ For other types of fields for different purpose, please refer to: https://docs.djangoproject.com/ja/1.10/ref/models/fields/ """
5
6     name = models.CharField(max_length=200)
7     relation = models.CharField(max_length=200)
8     phone = models.CharField(max_length=200)
9     email = models.CharField(max_length=200)
```

The bottom right corner of the editor shows "1:1 Python Spaces: 4". Below the editor is a terminal window titled "bash - denffer-my_5" with the command "python manage.py startapp mycontacts" entered and executed.

Remember to save your file

The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, Run Project, and a status message "ALL CHANGES SAVED". The left sidebar has sections for Workspace (with "my_project" folder), Navigate, and Commands. The main workspace shows the project structure:

- my_project/
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
- mycontacts/
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - models.py (selected)
 - tests.py
 - views.py
- manage.py
- README.md

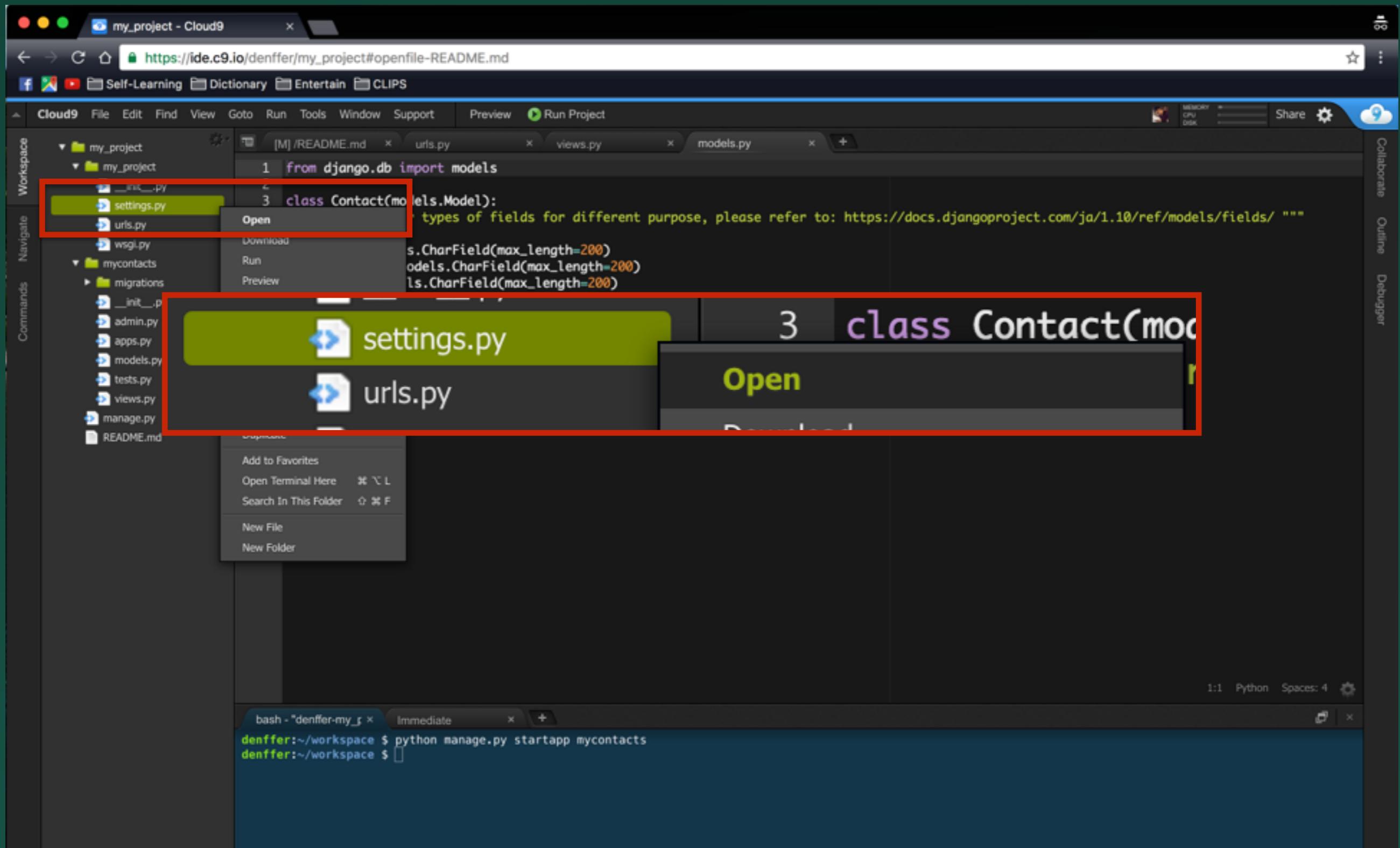
The code editor window displays the contents of models.py:

```
1  From django.db import models
2
3  class Contact(models.Model):
4      """ For other types of fields for different purpose, please refer to: https://docs.djangoproject.com/ja/1.10/ref/models/fields/ """
5
6      name = models.CharField(max_length=200)
7      relation = models.CharField(max_length=200)
8      phone = models.CharField(max_length=200)
9      email = models.CharField(max_length=200)
```

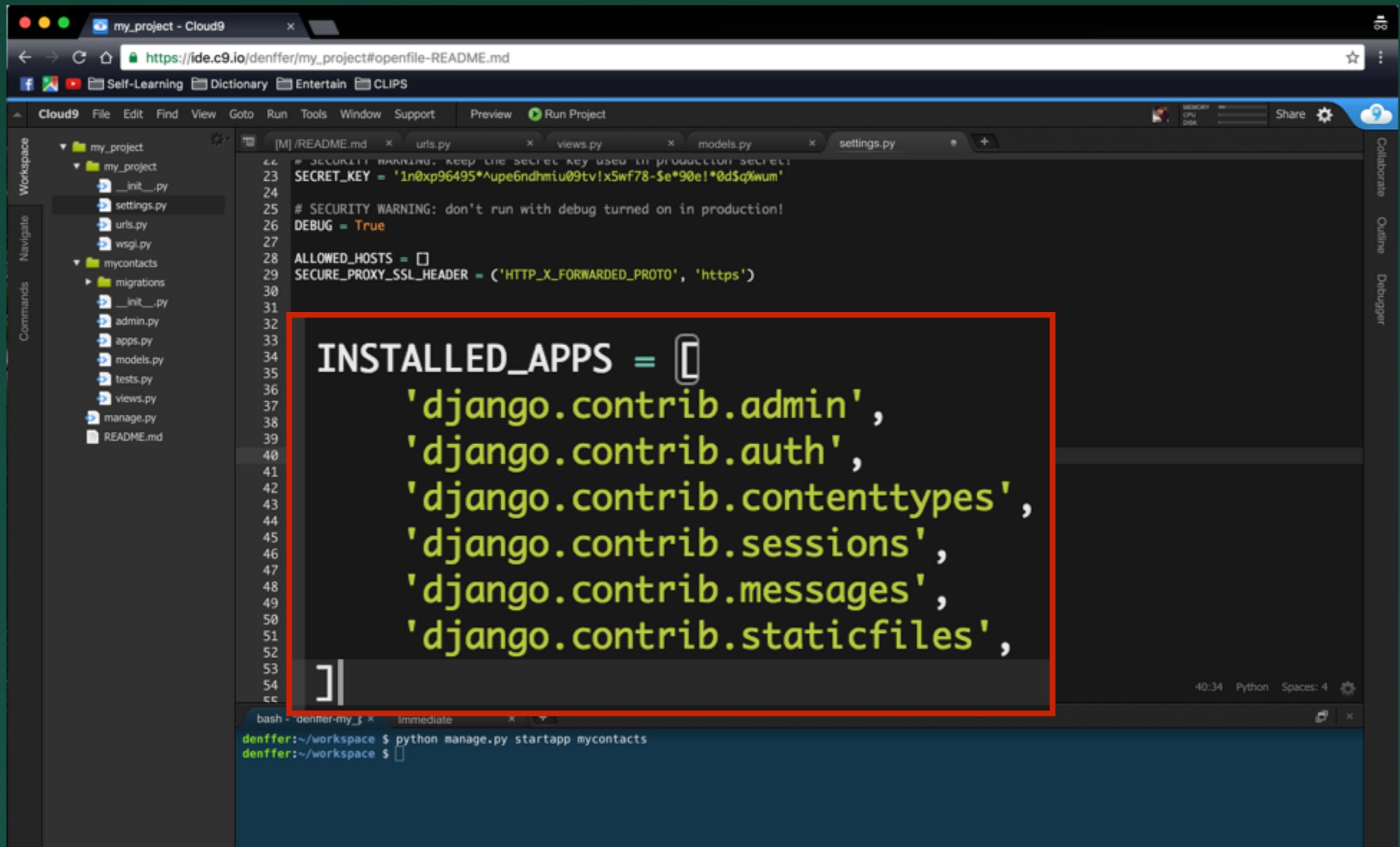
The bottom right corner of the code editor shows "1:1 Python Spaces: 4". Below the code editor is a terminal window titled "bash - denffer-my_5" showing the command:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Let's go to `settings.py`



Find 'INSTALLED_APPS'



The screenshot shows the Cloud9 IDE interface with the project 'my_project' open. The 'settings.py' file is the active tab, displaying configuration settings. A red box highlights the 'INSTALLED_APPS' section, which lists several Django contrib modules. The code editor has syntax highlighting and line numbers. Below the editor, a terminal window shows the command to start the app.

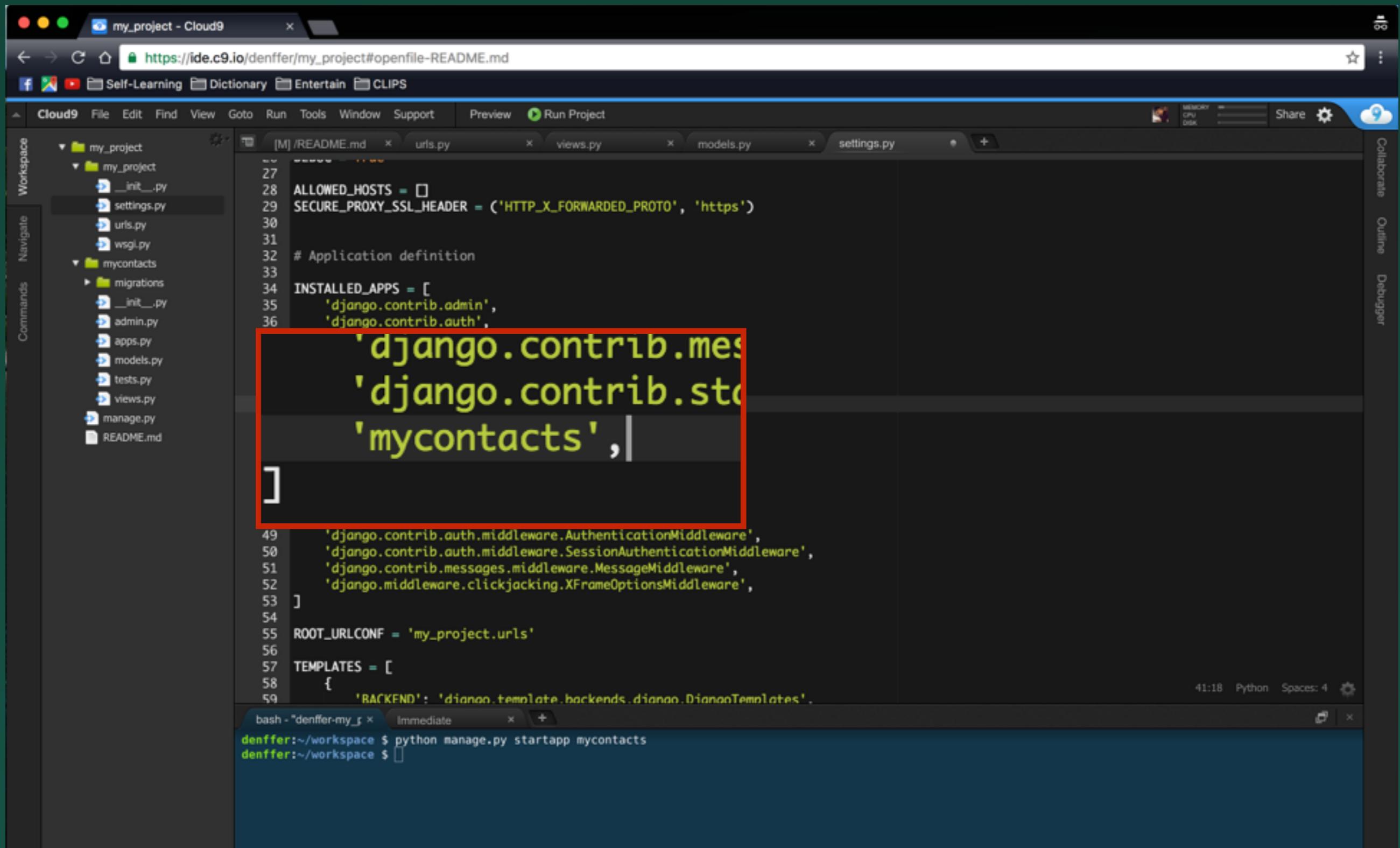
```
SECRET_KEY = '1n0xp96495*^upe6ndhmiu09tv!x5wf78-$e*90e!*0d$q%wum'
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []
SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTO', 'https')

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

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

State your application in INSTALLED_APPS



The screenshot shows the Cloud9 IDE interface with the project 'my_project' open. The 'settings.py' file is the active tab, displaying Python code for a Django project. A red box highlights the 'INSTALLED_APPS' list, which contains three entries: 'django.contrib.messages', 'django.contrib.staticfiles', and 'mycontacts'. The code also includes configurations for allowed hosts, secure proxy SSL header, and middleware.

```
27
28 ALLOWED_HOSTS = []
29 SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTO', 'https')
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.messages',
37     'django.contrib.staticfiles',
38     'mycontacts',
39 ]
40
41     'django.contrib.auth.middleware.AuthenticationMiddleware',
42     'django.contrib.auth.middleware.SessionAuthenticationMiddleware',
43     'django.contrib.messages.middleware.MessageMiddleware',
44     'django.middleware.clickjacking.XFrameOptionsMiddleware',
45 ]
46
47 ROOT_URLCONF = 'my_project.urls'
48
49 TEMPLATES = [
50     {
51         'BACKEND': 'dianan.template.backends.dianan.DiananTemplates',
52     },
53 ]
```

In the bottom terminal window, the command `python manage.py startapp mycontacts` is run, confirming the application's creation.

Remember to save your file

The screenshot shows the Cloud9 IDE interface with a Django project named 'my_project'. The project structure on the left includes 'my_project' with files __init__.py, settings.py, urls.py, and wsgi.py; and 'mycontacts' with files __init__.py, admin.py, apps.py, models.py, tests.py, views.py, and manage.py. A README.md file is also present. The code editor on the right displays the contents of settings.py. The terminal at the bottom shows the command 'python manage.py startapp mycontacts' being run.

```
SECRET_KEY = '1n0xp96495*^upe6ndhmiu09tv!x5wf78-$e*90e!*0d$q%wum'
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTO', 'https')

# Application definition

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

MIDDLEWARE_CLASSES = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.auth.middleware.SessionAuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'my_project.urls'

# Internationalization
# https://docs.djangoproject.com/en/1.11/topics/i18n/
# LANGUAGE_CODE = 'en-us'
# TIME_ZONE = 'UTC'
# USE_I18N = True
# USE_L10N = True
# USE_TZ = True
```

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Django does not provide **Form** by default

So you have to create it by yourself

Create a file named ‘forms.py’ in your application ‘mycontacts’

The screenshot shows the Cloud9 IDE interface with a Django project named 'my_project'. The project structure on the left includes files like README.md, urls.py, views.py, models.py, and settings.py. A context menu is open over the 'mycontacts' application directory, with 'Open' selected. The main code editor shows the contents of settings.py, which includes comments about Django settings and a production secret key. Below the code editor is a terminal window showing the command 'python manage.py startapp mycontacts' being run.

```
1 """
2 Django settings for my_project project.
3 
4 Generated by 'django-admin startproject' using Django 1.9.
5 
6 For more information on this file, see
7 https://docs.djangoproject.com/en/1.9/topics/settings/
8 
9     Full list of settings and their values, see
10    https://docs.djangoproject.com/en/1.9/ref/settings/
11 
12    Paths inside the project like this: os.path.join(BASE_DIR, ...)
13    os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
14 
15    Secret development settings - unsuitable for production
16    https://docs.djangoproject.com/en/1.9/howto/deployment/checklist/
17 
18    production secret!
19    -$e*90e!*0d$q%wum'
20 
21    on in production!
```

```
29 SECURE_PROXY_SSL_HEADER = ('HTTP_X_FORWARDED_PROTO', 'https')
30
31
32 # Application definition
33
```

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

There you go. Open it.

The screenshot shows the Cloud9 IDE interface with a Django project named "my_project".

File Explorer (Workspace):

- my_project
- my_project

 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py

- mycontacts

 - migrations
 - __init__.py

- forms.py
- models.py
- manage.py
- README.md

Code Editors:

- /README.md
- urls.py
- views.py
- models.py
- settings.py

Terminal:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Remember? Django Form is used to fetch data from Html Form

The screenshot shows the Cloud9 IDE interface with a Django project named 'my_project'. The project structure in the workspace includes files like `__init__.py`, `settings.py`, `urls.py`, `wsgi.py`, and `forms.py` within the `mycontacts` app. The `forms.py` file is currently selected. The terminal at the bottom shows the command `python manage.py startapp mycontacts` being run successfully.

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Now, copy and paste the codes in **forms.py** from the repository you cloned from Github

The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run Project. The workspace sidebar shows the project structure:

- my_project
 - my_project
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
 - mycontacts
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - forms.py
 - models.py
 - tests.py
 - views.py
 - manage.py
 - README.md

The main editor window displays the contents of the "forms.py" file:

```
1 from django import forms
2 from .models import Contact
3
4 class AddForm(forms.Form):
5     class Meta:
6         model = Contact
7         fields = ('name', 'relation', 'phone', 'email',)
```

At the bottom, a terminal window titled "bash - denffer-my_5" shows the command: "denffer:~/workspace \$ python manage.py startapp mycontacts".

Remember to save your file

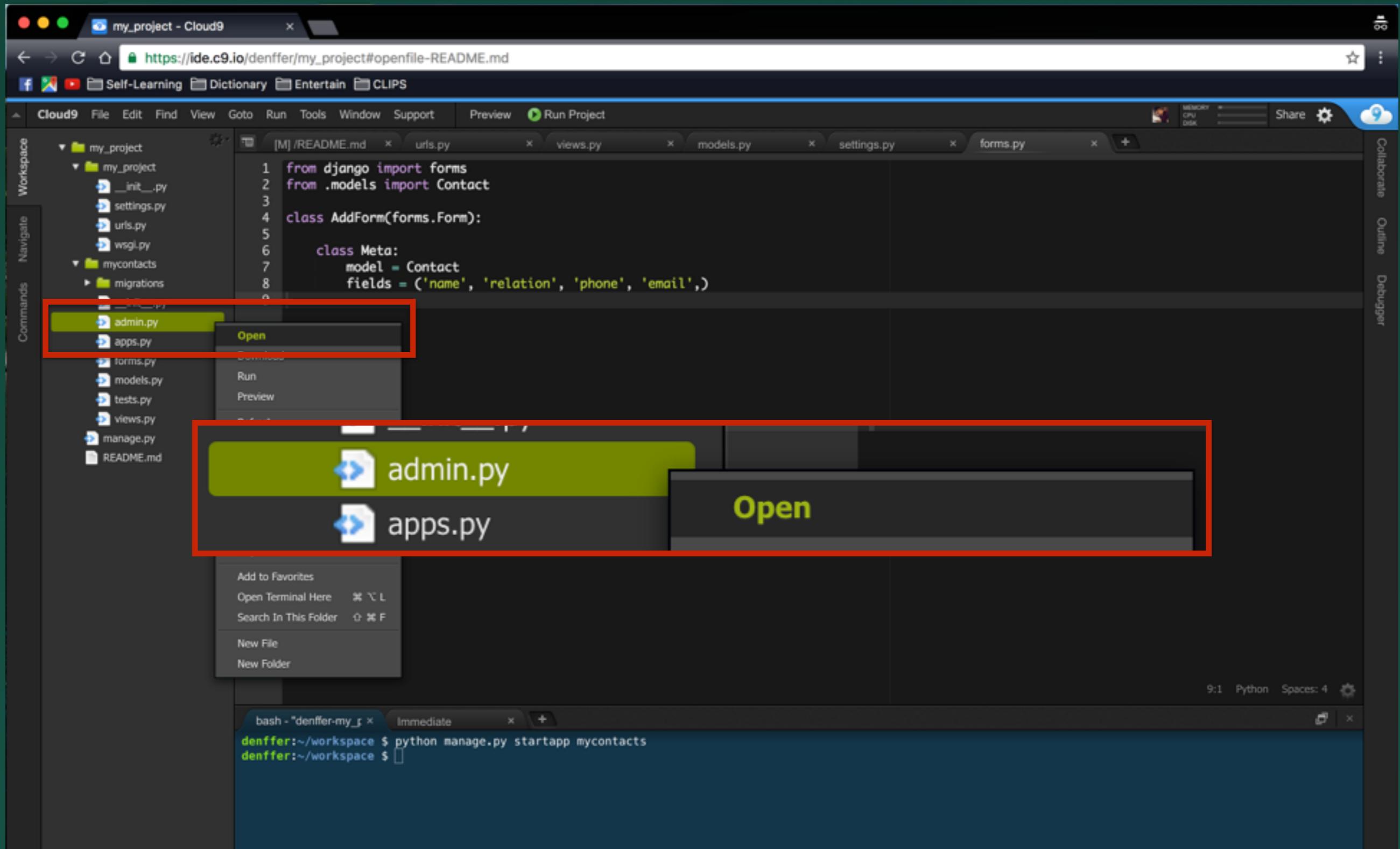
The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, Run Project, and a status message "ALL CHANGES SAVED". The left sidebar has sections for Workspace, Navigate, and Commands. The Workspace section shows a project structure with folders "my_project" and "mycontacts", and files like "__init__.py", "settings.py", "urls.py", "wsgi.py", "migrations", "admin.py", "apps.py", "forms.py", "models.py", "tests.py", "views.py", "manage.py", and "README.md". The main editor area shows Python code for a Django form:

```
1 from django import forms
2 from .models import Contact
3
4 class AddForm(forms.Form):
5     class Meta:
6         model = Contact
7         fields = ('name', 'relation', 'phone', 'email',)
```

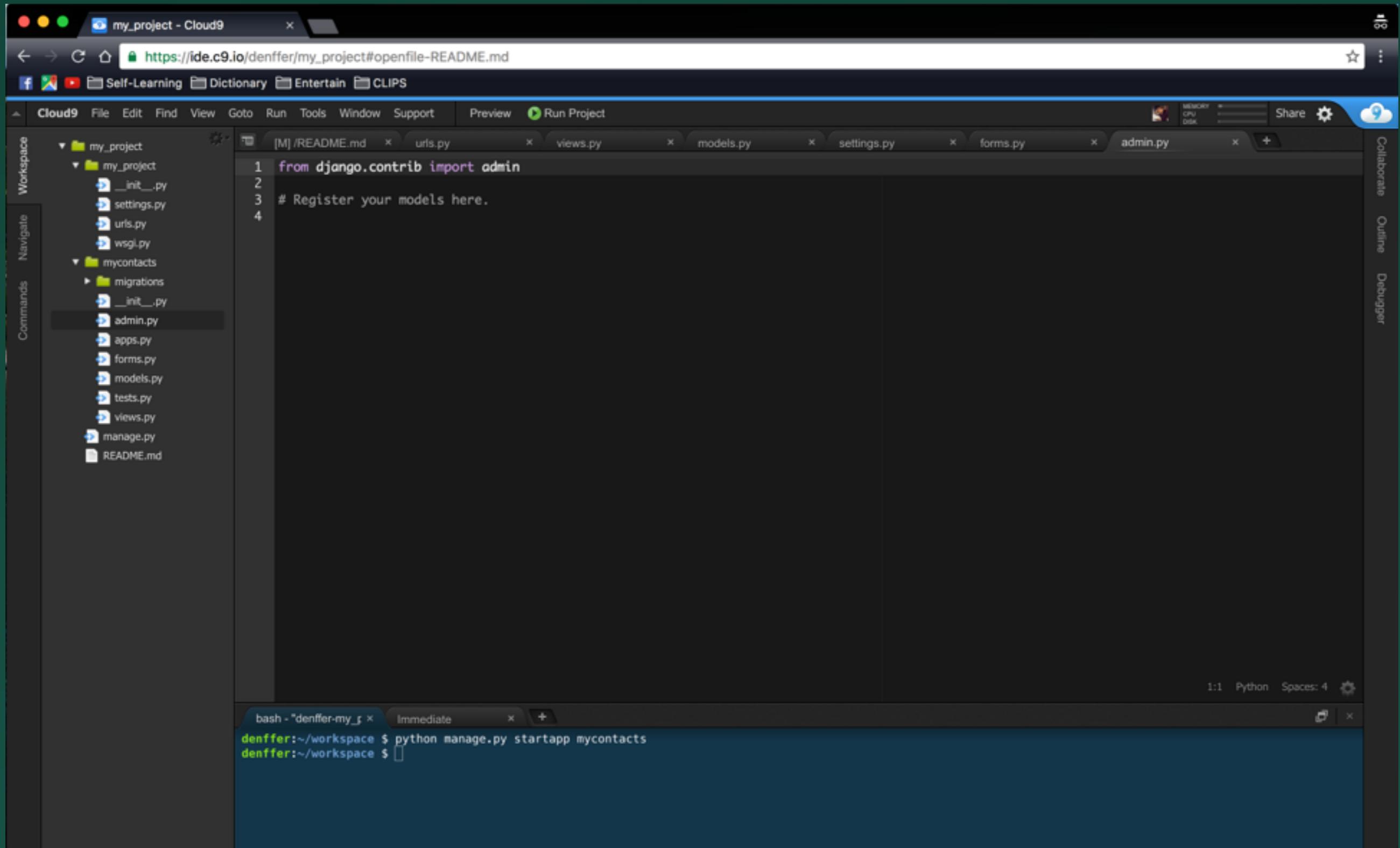
The bottom terminal window shows a bash session:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

One more step. Let's go to admin.py



Remember? Admin is used to register your Model in your Database



The screenshot shows the Cloud9 IDE interface with a Django project named "my_project". The workspace sidebar on the left lists files and folders: README.md, __init__.py, settings.py, urls.py, wsgi.py, mycontacts (with migrations, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py), manage.py, and README.md. The code editor window displays the contents of admin.py:

```
1 from django.contrib import admin
2
3 # Register your models here.
4
```

Below the code editor is a terminal window titled "bash - denffer-my_5" showing the command: `denffer:~/workspace $ python manage.py startapp mycontacts`. The status bar at the bottom right indicates "1:1 Python Spaces: 4".

Now, copy and paste the codes in **admin.py** from the repository you cloned from Github

The screenshot shows the Cloud9 IDE interface with a dark theme. The top bar displays the project name "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The menu bar includes Cloud9, File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run Project. The workspace sidebar shows the project structure:

- my_project (selected)
- my_project (containing __init__.py, settings.py, urls.py, wsgi.py)
- mycontacts (containing migrations, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py, manage.py, README.md)

The main editor area shows the content of the "admin.py" file in the "mycontacts" app:

```
1 from django.contrib import admin
2 from .models import Contact
3
4 admin.site.register(Contact)
5
```

The bottom terminal window shows the command "python manage.py startapp mycontacts" being run in a bash session:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $
```

Remember to save your file

The screenshot shows the Cloud9 IDE interface with a dark theme. At the top, the title bar reads "my_project - Cloud9". The address bar shows the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". Below the address bar is a toolbar with icons for File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, Run Project, and a status message "ALL CHANGES SAVED". On the left, there's a sidebar with tabs for Workspace, Navigate, and Commands. The Workspace tab is active, displaying the project structure:

```
my_project
  my_project
    __init__.py
    settings.py
    urls.py
    wsgi.py
  mycontacts
    migrations
      __init__.py
    admin.py
    apps.py
    forms.py
    models.py
    tests.py
    views.py
  manage.py
  README.md
```

The main workspace shows several open files in tabs: README.md, urls.py, views.py, models.py, settings.py, forms.py, and admin.py. The admin.py file contains the following Python code:

```
1 from django.contrib import admin
2 from .models import Contact
3
4 admin.site.register(Contact)
5
```

At the bottom, there's a terminal window titled "bash - denffer-my_5" showing the command "python manage.py startapp mycontacts" being run.

Great!

Now you have to make a migration so that your **Model** and be recognized and stored in **Database**

Go to command line and type in the command

The screenshot shows the Cloud9 IDE interface. On the left, the workspace sidebar displays the project structure:

- my_project (selected)
- my_project (containing __init__.py, settings.py, urls.py, wsgi.py)
- mycontacts (containing migrations, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py)
- manage.py
- README.md

In the center, several code editor tabs are open for README.md, urls.py, views.py, models.py, settings.py, forms.py, and admin.py. Below the editors is a terminal window titled "bash - 'denffer-my_'" showing the command:

```
denffer:~/workspace $ python manage.py startapp mycontacts  
denffer:~/workspace
```

The command `$ python manage.py makemigrations` is highlighted with a red box.

Great! You have just created your first model

The screenshot shows the Cloud9 IDE interface with a Django project named 'my_project'. The project structure on the left includes files like README.md, manage.py, and several files under my_contacts (migrations, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py). The code editor shows a file named 'models.py' with the following content:

```
1 from django.contrib import admin
2 from .models import Contact
3
4 admin.site.register(Contact)
```

Below the code editor is a terminal window showing the command-line output of running migrations:

```
dentfer:~/workspace $ python manage.py startapp mycontacts
dentfer:~/workspace $ python manage.py makemigrations
```

A red box highlights the terminal output, specifically the message:

Migrations for 'mycontacts':
0001_initial.py:
- Create model Contact

Now, type in the command to write your model into your **Database**

The screenshot shows the Cloud9 IDE interface. On the left, the workspace sidebar displays the project structure:

- my_project (selected)
- my_project (containing __init__.py, settings.py, urls.py, wsgi.py)
- mycontacts (containing migrations, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py)
- manage.py
- README.md

In the center, several code editor tabs are open: README.md, urls.py, views.py, models.py, settings.py, forms.py, and admin.py. The models.py tab contains the following Python code:

```
1 from django.contrib import admin
2 from .models import Contact
3
4 admin.site.register(Contact)
5
```

Below the code editors is a terminal window titled "bash - 'denffer-my_'" showing the following command history:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $ python manage.py makemigrations
Migrations for 'mycontacts':
  0001_initial.py:
    - Create model Contact
denffer:~/workspace
```

The command `$ python manage.py migrate` is highlighted with a red box.

Great! Your first database is just created with your model written in it

The screenshot shows the Cloud9 IDE interface with a Django project named "my_project". The project structure includes files like README.md, urls.py, views.py, models.py, settings.py, forms.py, and admin.py. A terminal window displays the command-line process of creating a new app "mycontacts", running migrations, and viewing the SQLite database "db.sqlite3".

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $ python manage.py makemigrations
Migrations for 'mycontacts':
  0001_initial.py:
    - Create model Contact
denffer:~/workspace $ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, contenttypes, mycontacts, auth, sessions
  Applying auth.0001_initial... 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 mycontacts.0001_initial... OK
  Applying sessions.0001_initial... OK
denffer:~/workspace $
```

A red box highlights the "db.sqlite3" file in the terminal output, indicating the newly created database.

Let's check on our database
and see how it looks like

However, before that, we have to
create a superuser account

Type in the command to create a superuser

The screenshot shows a Cloud9 IDE interface with a Python Django project named 'my_project'. The project structure on the left includes files like README.md, __init__.py, settings.py, urls.py, wsgi.py, mycontacts (with migrations, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py), db.sqlite3, manage.py, and README.md. The terminal window at the bottom contains the following command:

```
$ python manage.py createsuperuser
```

This command is highlighted with a red border.

The terminal output shows the migration process for the 'mycontacts' app:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $ python manage.py makemigrations
Migrations for 'mycontacts':
  0001_initial.py:
    - Create model Contact
denffer:~/workspace $ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, contenttypes, mycontacts, auth, sessions
Running migrations:
  Rendering model states... DONE
  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 mycontacts.0001_initial... OK
  Applying sessions.0001_initial... OK
denffer:~/workspace
```

This should be easy

The screenshot shows a Cloud9 IDE interface with a terminal window displaying the creation of a Django superuser.

Workspace:

- my_project
- my_contacts
- migrations
- __init__.py
- admin.py
- apps.py
- forms.py
- models.py
- tests.py
- views.py
- db.sqlite3
- manage.py
- README.md

Terminal Output:

```
python - "denffer-my" Immediate +  
denffer:~/workspace $ python manage.py startapp my_contacts  
denffer:~/workspace $ python manage.py makemigrations  
Migrations for 'my_contacts':  
    0001_initial:  
        - Create model Contact  
denffer:~/workspace $ python manage.py migrate  
Operations to perform:  
  Apply all migrations: admin, contenttypes, my_contacts, auth, sessions  
Running migrations:  
  Rendering model states... DONE  
  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 my_contacts.0001_initial... OK  
  Applying sessions.0001_initial... OK  
denffer:~/workspace $ python manage.py createsuperuser
```

Output (highlighted by a red box):

```
Username (leave blank to use 'ubuntu'): your_name  
Email address: your_email@gmail.com  
Password:  
Password (again):  
Superuser created successfully.
```

Now comes the interesting part!

Let's us launch our server!

Type in the command to launch the server

The screenshot shows a Cloud9 IDE interface with a terminal window containing migration commands and a highlighted command to start the development server.

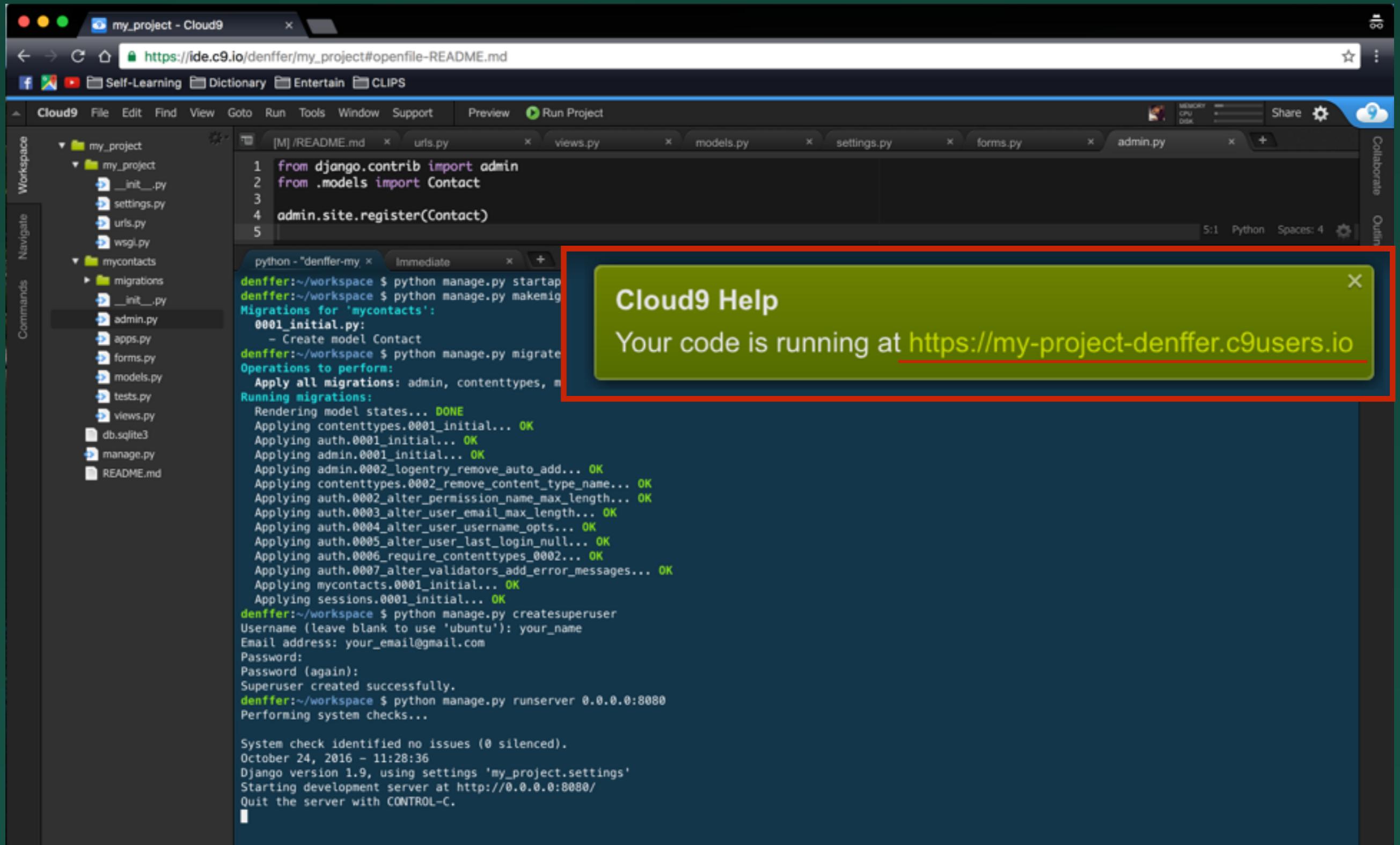
Terminal Output:

```
denffer:~/workspace $ python manage.py startapp mycontacts
denffer:~/workspace $ python manage.py makemigrations
Migrations for 'mycontacts':
  0001_initial:
    - Create model Contact
denffer:~/workspace $ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, contenttypes, mycontacts, auth, sessions
Running migrations:
  Rendering model states... DONE
  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 mycontacts.0001_initial... OK
  Applying sessions.0001_initial... OK
denffer:~/workspace $ python manage.py createsuperuser
Username (leave blank to use 'ubuntu'): your_name
Email address: your_email@gmail.com
Password:
Password (again):
Superuser created successfully.
denffer:~/workspace
```

Command Line:

```
$ python manage.py runserver 0.0.0.0:8080
```

Click on the link to your website!



Welcome to your first Django error

The screenshot shows a web browser window with the title bar "my_project - Cloud9" and the URL "https://my-project-denffer.c9users.io". The main content area displays a large red-bordered box containing the text "TemplateDoesNotExist". Below this, detailed error information is provided:

Exception Type: TemplateDoesNotExist
Exception Value: mycontacts/show.html
Exception Location: /usr/local/lib/python2.7/dist-packages/django/template/loader.py in get_template, line 43
Python Executable: /usr/bin/python
Python Version: 2.7.6
Python Path: ['/home/ubuntu/workspace', '/usr/lib/python2.7', '/usr/lib/python2.7/plat-x86_64-linux-gnu', '/usr/lib/python2.7/lib-tk', '/usr/lib/python2.7/lib-old', '/usr/lib/python2.7/lib-dynload', '/usr/local/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages/PILcompat', '/usr/lib/python2.7/dist-packages/gtk-2.0', '/usr/lib/pymodules/python2.7']
Server time: Mon, 24 Oct 2016 11:32:43 +0000

Template-loader postmortem

Django tried loading these templates, in this order:

```
Using engine django:  
* django.template.loaders.app_directories.Loader: /usr/local/lib/python2.7/dist-packages/django/contrib/admin/templates/mycontacts/show.html (Source does not exist)  
* django.template.loaders.app_directories.Loader: /usr/local/lib/python2.7/dist-packages/django/contrib/auth/templates/mycontacts/show.html (Source does not exist)
```

Traceback [Switch to copy-and-paste view](#)

```
/usr/local/lib/python2.7/dist-packages/django/core/handlers/base.py in get_response  
    149.             response = self.process_exception_by_middleware(e, request)  
...  
▶ Local vars  
/usr/local/lib/python2.7/dist-packages/django/core/handlers/base.py in get_response  
    147.             response = wrapped_callback(request, *callback_args, **callback_kwargs)  
...  
▶ Local vars  
/home/ubuntu/workspace/mycontacts/views.py in show
```

Don't worry. Let's edit and add '/admin/' to your URL

The screenshot shows a web browser window with a red box highlighting the URL bar. The URL is `https://my-project-denffer.c9users.io/admin/`. The main content area displays a Django error page for a missing template.

Request Method: GET
Request URL: `https://my-project-denffer.c9users.io/`
Django Version: 1.9
Exception Type: TemplateDoesNotExist
Exception Value: `mycontacts/show.html`
Exception Location: `/usr/local/lib/python2.7/dist-packages/django/template/loader.py` in `get_template`, line 43
Python Executable: `/usr/bin/python`
Python Version: 2.7.6
Python Path: `['/home/ubuntu/workspace', '/usr/lib/python2.7', '/usr/lib/python2.7/plat-x86_64-linux-gnu', '/usr/lib/python2.7/lib-tk', '/usr/lib/python2.7/lib-old', '/usr/lib/python2.7/lib-dynload', '/usr/local/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages/PILcompat', '/usr/lib/python2.7/dist-packages/gtk-2.0', '/usr/lib/pymodules/python2.7']`
Server time: Mon, 24 Oct 2016 11:32:43 +0000

Template-loader postmortem

Django tried loading these templates, in this order:

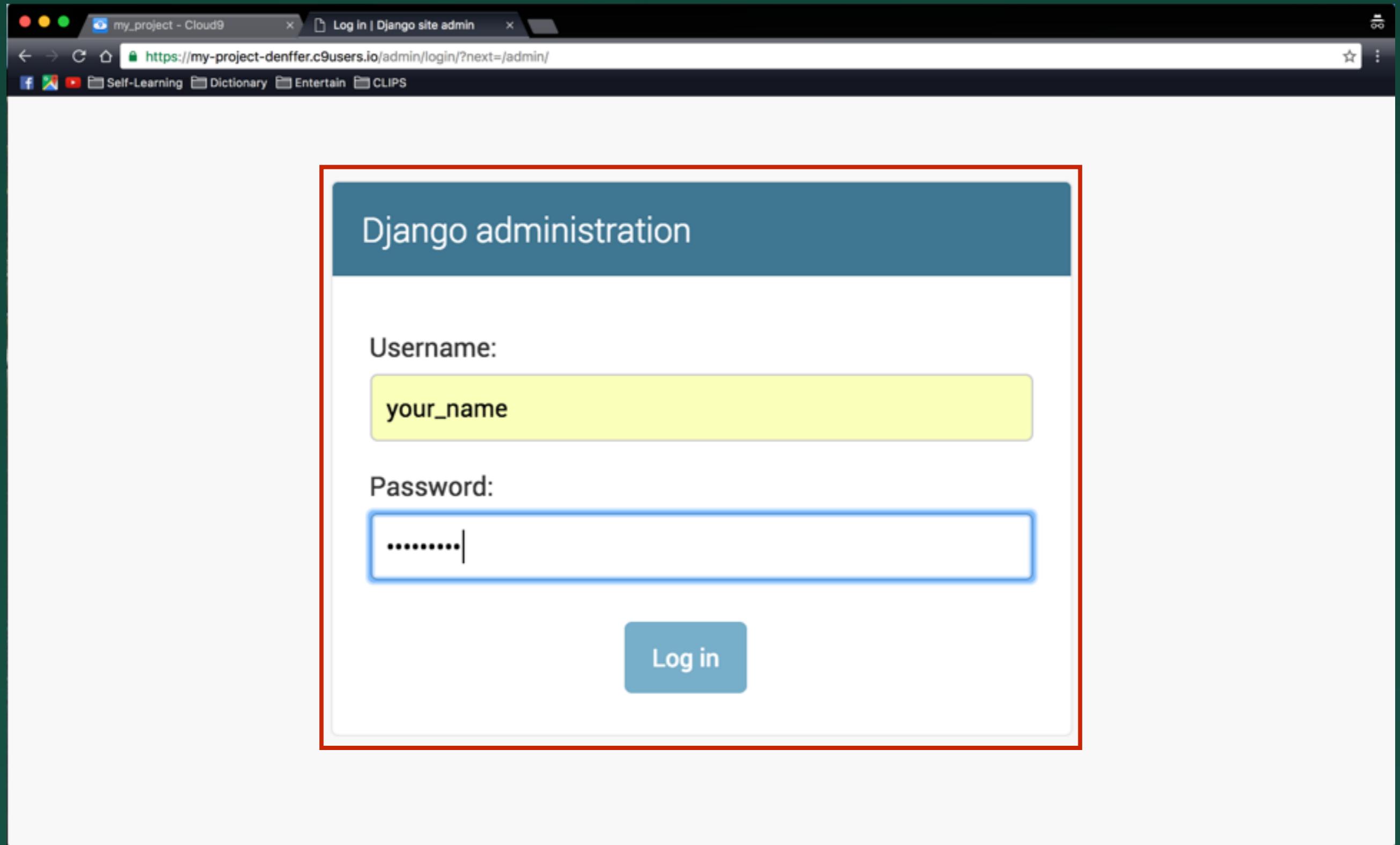
Using engine django:

- `django.template.loaders.app_directories.Loader: /usr/local/lib/python2.7/dist-packages/django/contrib/admin/templates/mycontacts/show.html` (Source does not exist)
- `django.template.loaders.app_directories.Loader: /usr/local/lib/python2.7/dist-packages/django/contrib/auth/templates/mycontacts/show.html` (Source does not exist)

Traceback [Switch to copy-and-paste view](#)

```
/usr/local/lib/python2.7/dist-packages/django/core/handlers/base.py in get_response
    149.             response = self.process_exception_by_middleware(e, request)
...
▶ Local vars
/usr/local/lib/python2.7/dist-packages/django/core/handlers/base.py in get_response
    147.             response = wrapped_callback(request, *callback_args, **callback_kwargs)
...
▶ Local vars
/home/ubuntu/workspace/mycontacts/views.py in show
```

Log in with your superuser account



You should be able to see your model

The screenshot shows the Django administration site. At the top, there's a header bar with the title "Site administration | Django site" and a URL "https://my-project-denffer.c9users.io/admin/". Below the header, the main navigation bar has links for "Self-Learning", "Dictionary", "Entertain", and "CLIPS". The main content area is titled "Django administration" and "WELCOME, YOUR_NAME. VIEW SITE / CHANGE PASSWORD / LOG OUT". On the left, there's a sidebar with "AUTHENTICATION AND AUTHORIZATION" section containing "Groups" and "Users" with "Add" and "Change" buttons. To the right, there's a "Recent Actions" sidebar with "My Actions" and a "Log out" link. The main content area has a red border around a section titled "MYCONTACTS" which contains a "Contacts" list and "Add" and "Change" buttons.

Django administration

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

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups + Add Change

Users + Add Change

MYCONTACTS

Contacts + Add Change

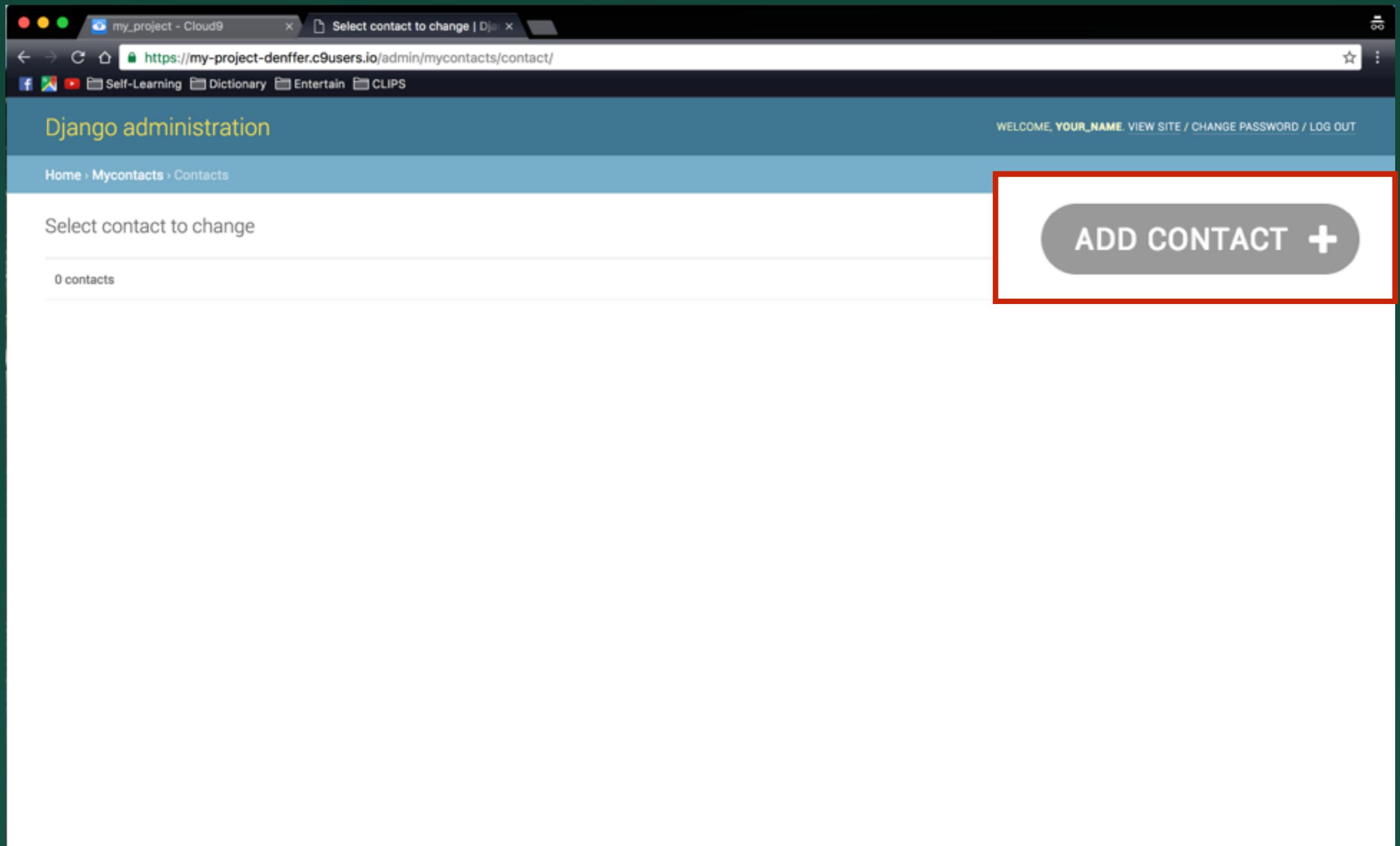
But it is now empty

A screenshot of a web browser window displaying the Django administration interface. The title bar shows the tab is titled "my_project - Cloud9" and the sub-tab is "Select contact to change | Django". The URL in the address bar is <https://my-project-denffer.c9users.io/admin/mycontacts/contact/>. The browser's toolbar includes icons for back, forward, search, and refresh, along with a star for bookmarks and a menu icon.

The main content area has a blue header bar with the text "Django administration" on the left and "WELCOME, YOUR_NAME. VIEW SITE / CHANGE PASSWORD / LOG OUT" on the right. Below this is a navigation bar with links to "Home", "Mycontacts", and "Contacts".

The main body of the page is titled "Select contact to change" and features a message "0 contacts" below a horizontal line. In the top right corner of this section, there is a button labeled "ADD CONTACT" with a plus sign icon.

Click on ‘ADD CONTACT’ button



Let's add 9 contact members

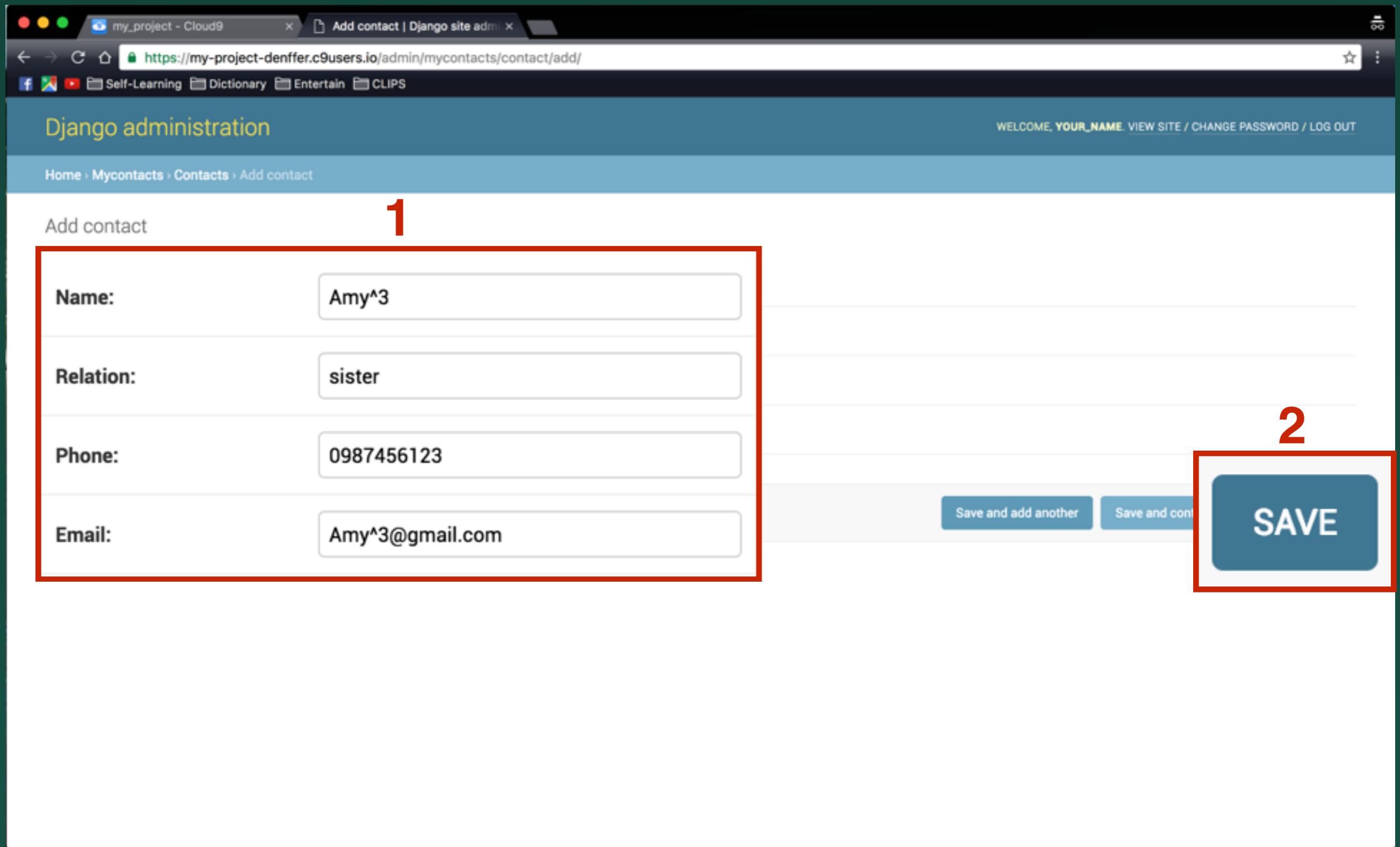
The screenshot shows the Django administration interface for adding a new contact. The page title is "Add contact | Django site admin". The URL in the browser is <https://my-project-denffer.c9users.io/admin/mycontacts/contact/add/>. The page header includes "Django administration", "WELCOME, YOUR_NAME. VIEW SITE / CHANGE PASSWORD / LOG OUT", and a navigation bar with "Home > Mycontacts > Contacts > Add contact".

The main form has the following fields:

- Name:** Tom (highlighted with a red box)
- Relation:** Friend
- Phone:** 0912345678
- Email:** Tom@gmail.com (highlighted with a red box)

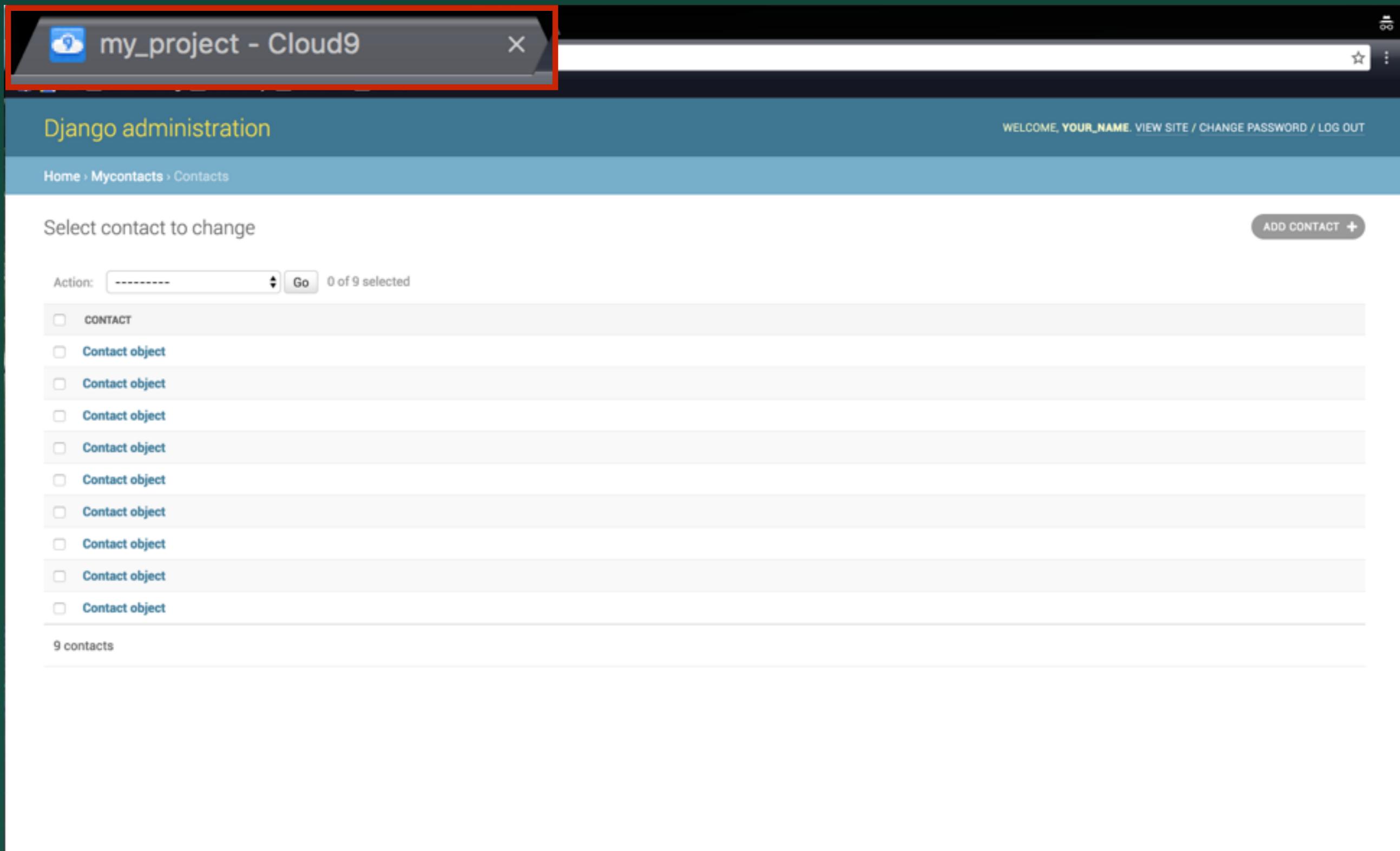
At the bottom right of the form area, there is a blue button labeled "Save and add another" (highlighted with a red box). To its right are two smaller buttons: "Continue editing" and "SAVE".

Click ‘SAVE’ button on the last contact member

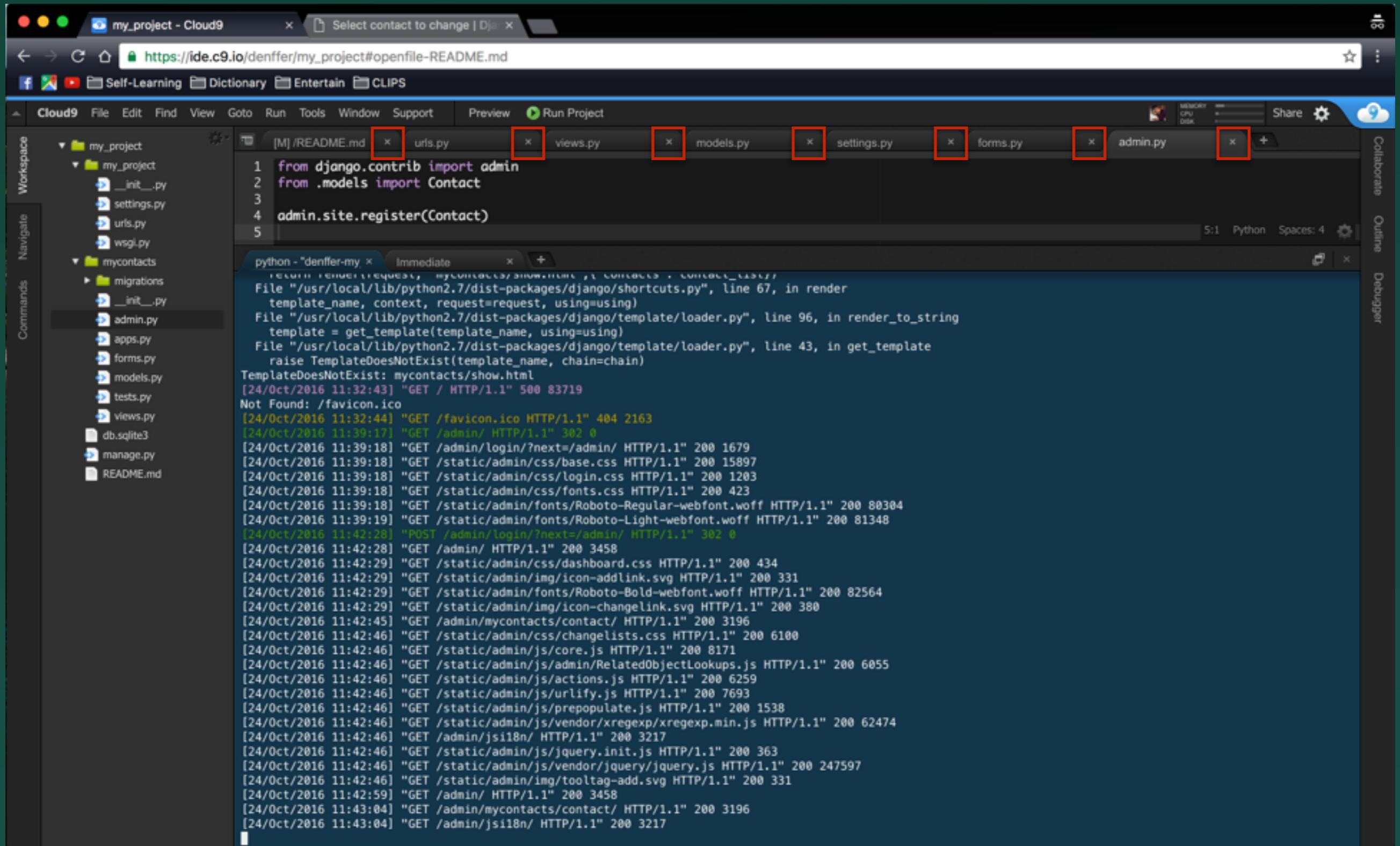


Now you should see 9 contact members

Let's go back to 'my_project'



Too messy. Let's kill all the tabs



Great! Now let's deal with the 'template' problem

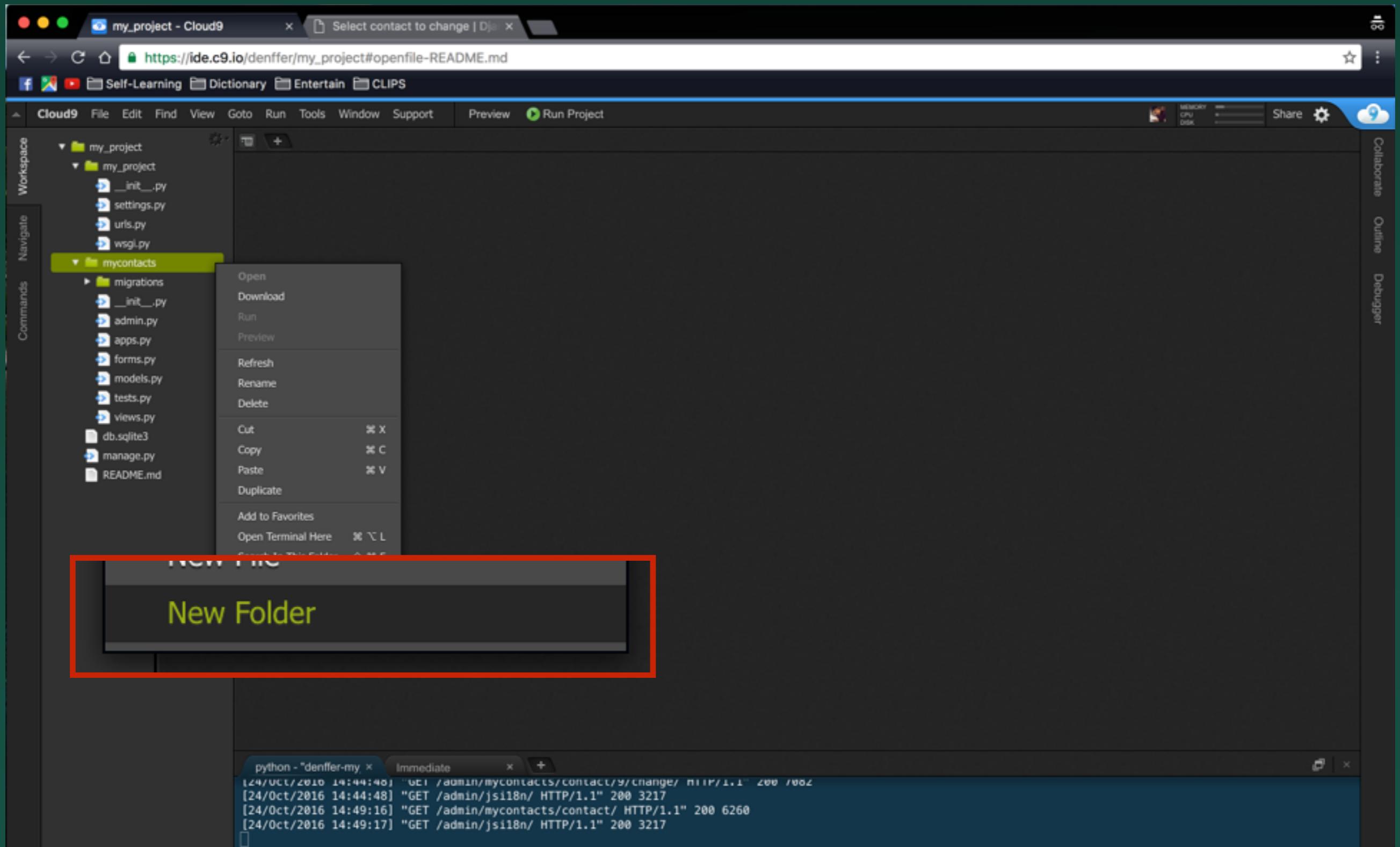
The screenshot shows the Cloud9 IDE interface with a Django project named 'my_project'. The project structure is visible in the workspace:

- my_project (root)
 - my_project (app)
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
 - mycontacts (app)
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - forms.py
 - models.py
 - tests.py
 - views.py
 - db.sqlite3
 - manage.py
 - README.md

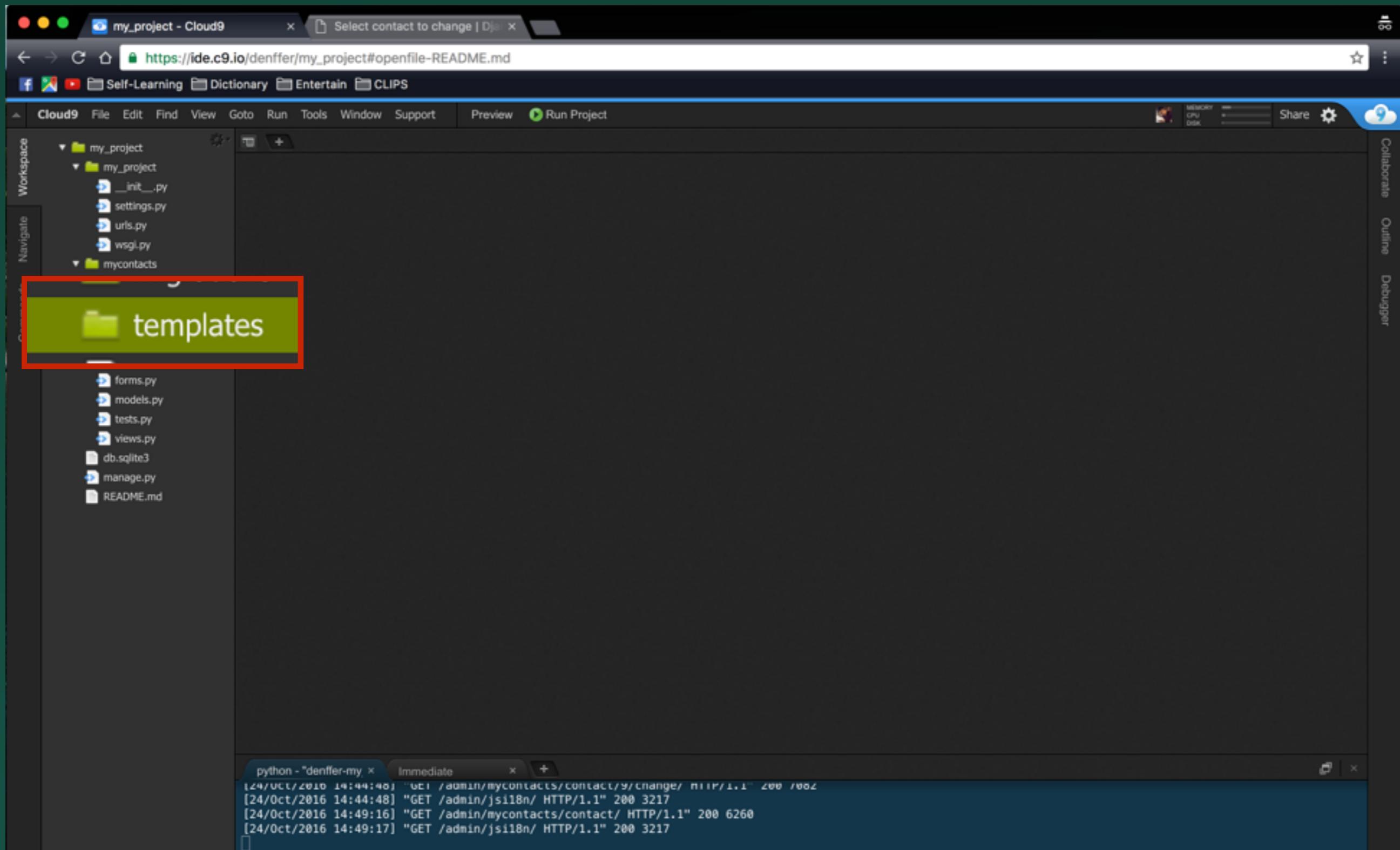
The 'admin.py' file under 'mycontacts' is currently selected. In the bottom right corner, there is a terminal window showing the following log output:

```
python - "denffer-my" x Immediate x +  
[24/Oct/2016 11:42:40] "GET /static/admin/img/tooltag-add.svg HTTP/1.1" 200 331  
[24/Oct/2016 11:42:59] "GET /admin/ HTTP/1.1" 200 3458  
[24/Oct/2016 11:43:04] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 3196  
[24/Oct/2016 11:43:04] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
```

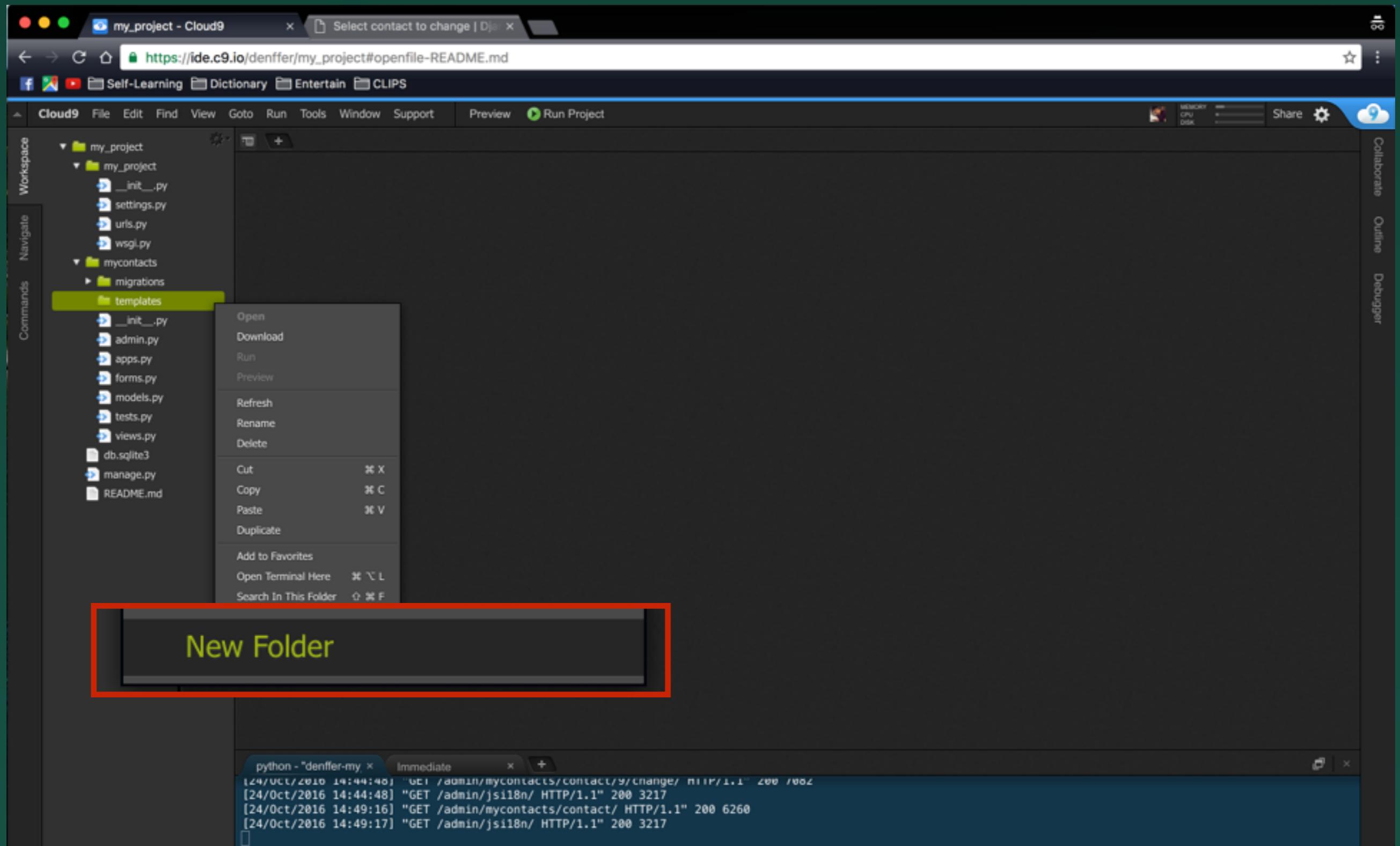
Create a folder named ‘templates’ under your application ‘mycontacts’



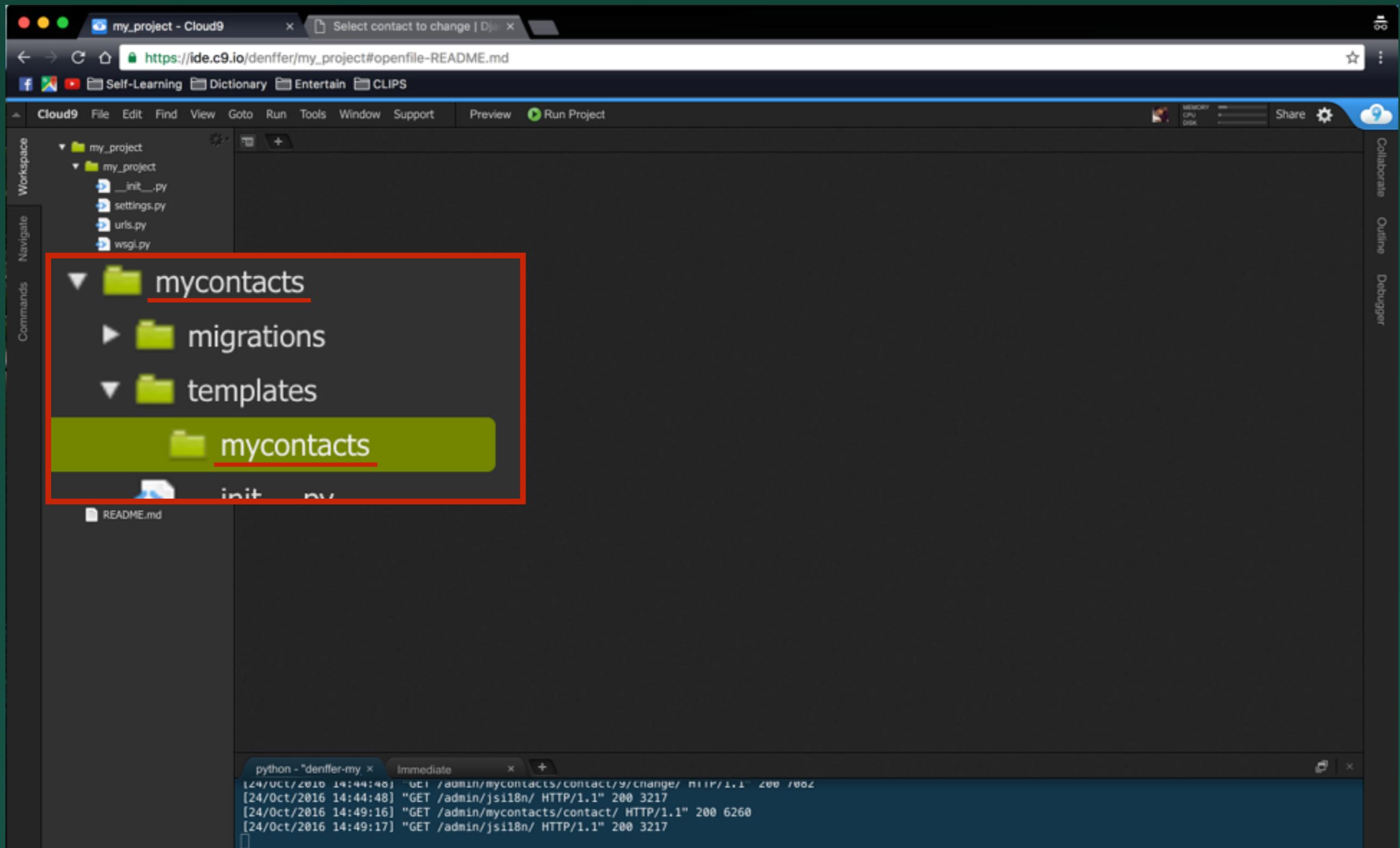
Remember? templates are used to contain all of your Html files



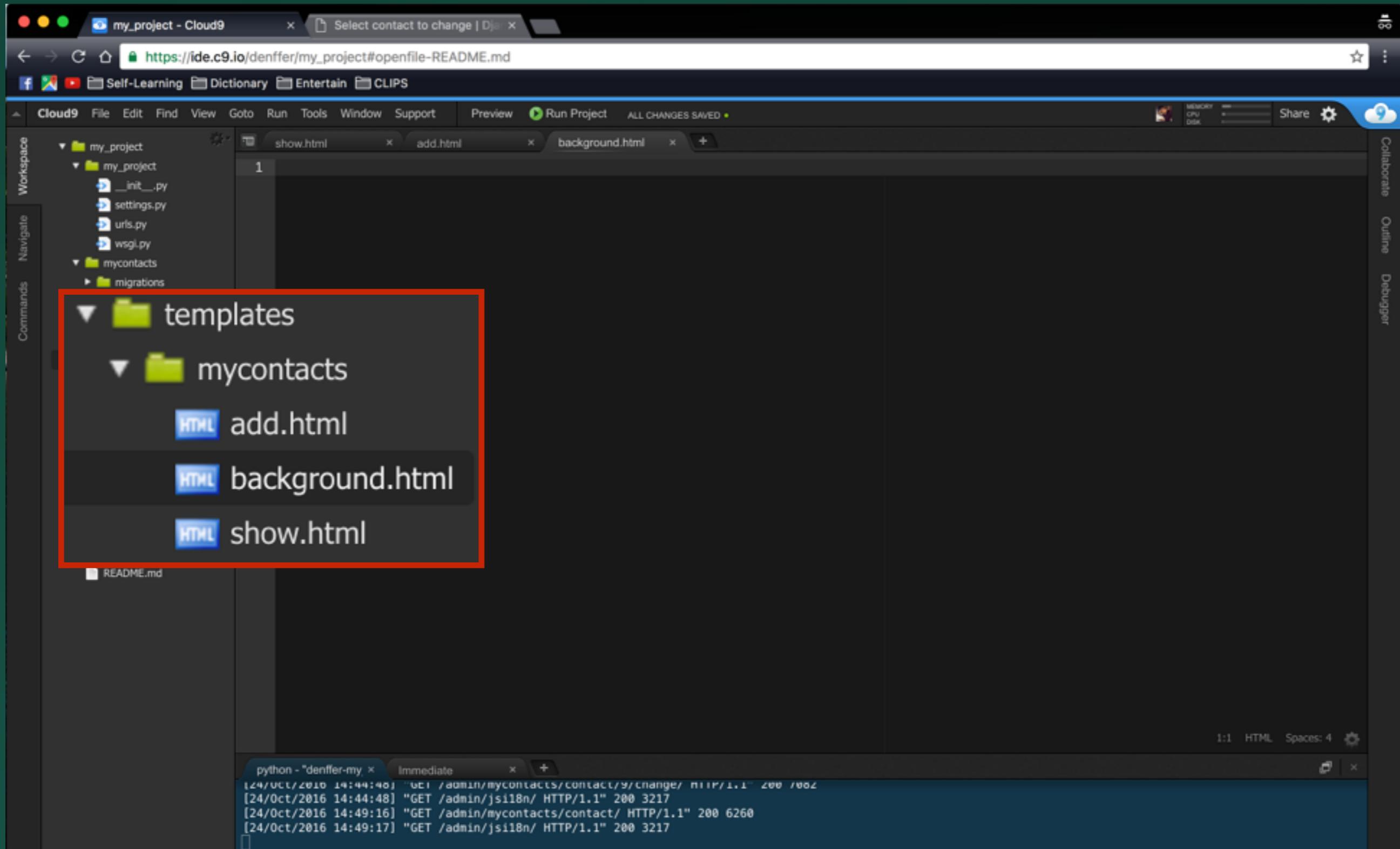
Now create another folder named 'mycontacts'
in the folder 'templates'



Remember the name of this folder should be the
same as the name of your application



Create ‘show.html’, ‘add.html’, and ‘background.html’ in the directory ‘mycontacts/templates/mycontacts’



Respectively copy and paste the codes in the html files from the repository you cloned from Github

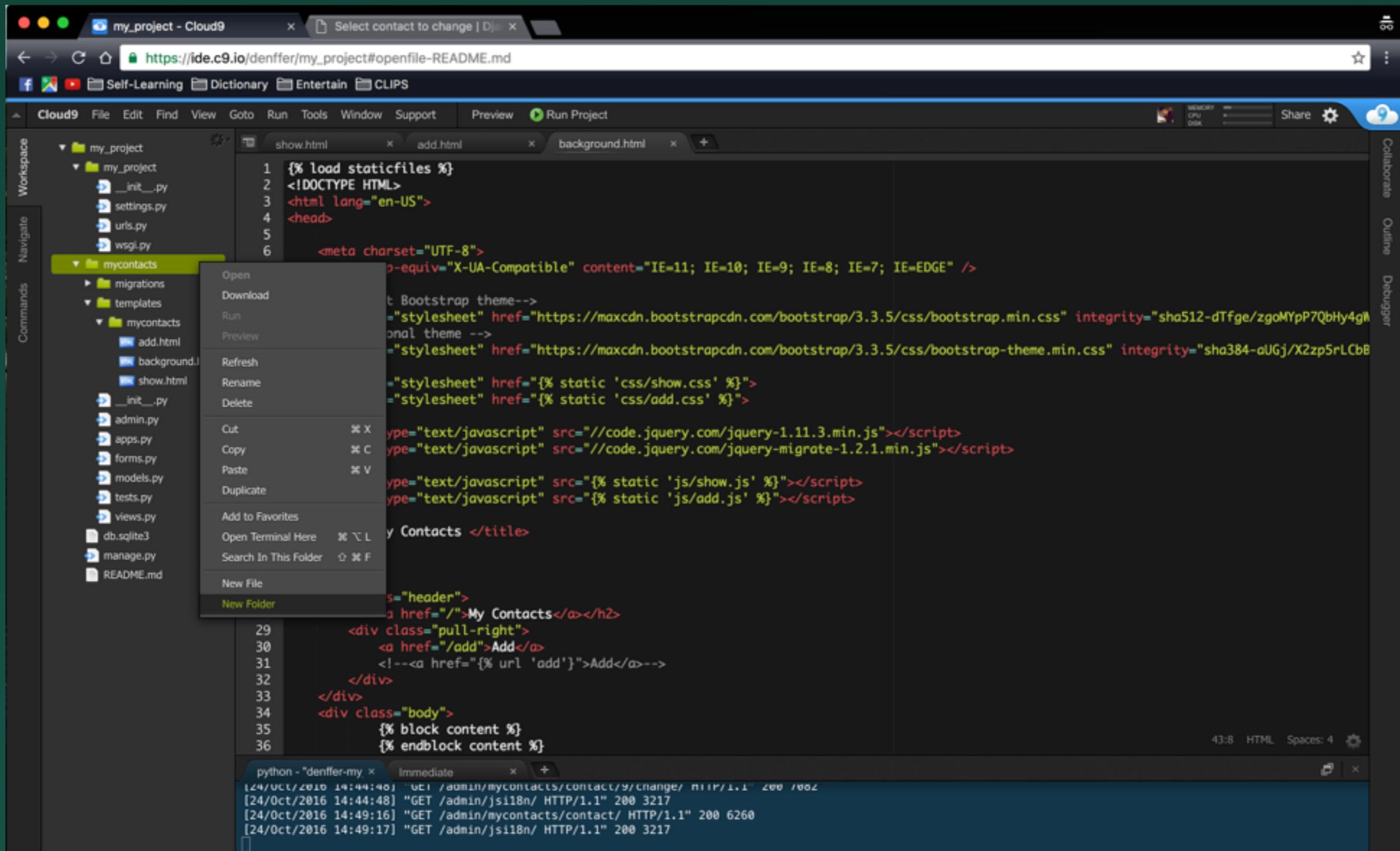
The screenshot shows the Cloud9 IDE interface. The top bar displays the title "my_project - Cloud9" and the URL "https://ide.c9.io/denffer/my_project#openfile-README.md". The left sidebar includes sections for Workspace, Navigate, Commands, and Collaborate. The workspace shows a file tree for a "my_contacts" application, including "my_contacts", "migrations", "templates", "models.py", "views.py", and "db.sqlite3". The "templates" folder contains "add.html", "background.html", and "show.html". The "show.html" file is currently open in the main editor area, displaying the following code:

```
1  {% load staticfiles %} 2  <!DOCTYPE HTML> 3  <html lang="en-US"> 4  <head> 5  6      <meta charset="UTF-8"> 7      <meta http-equiv="X-UA-Compatible" content="IE=11; IE=10; IE=9; IE=8; IE=7; IE=EDGE" /> 8  9      <!--latest Bootstrap theme--> 10     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css" integrity="sha512-dTfge/zgoMYpP7QbHy4gW/oPJ3mGU2fpopJ�" 11     <!-- Optional theme --> 12     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap-theme.min.css" integrity="sha384-aUGj/X2zp5rLcbB 13 14     <link rel="stylesheet" href="{% static 'css/show.css' %}"> 15     <link rel="stylesheet" href="{% static 'css/add.css' %}"> 16  17     <script type="text/javascript" src="//code.jquery.com/jquery-1.11.3.min.js"></script> 18     <script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script> 19  20     <script type="text/javascript" src="{% static 'js/show.js' %}"></script> 21     <script type="text/javascript" src="{% static 'js/add.js' %}"></script> 22  23     <title> My Contacts </title> 24  25 26     </head> 27     <body> 28         <div class="header"> 29             <h2><a href="/">My Contacts</a></h2> 30             <div class="pull-right"> 31                 <a href="/add">Add</a> 32                 <!--<a href="{% url 'add' %}">Add</a>--> 33             </div> 34         </div> 35         <div class="body"> 36             {% block content %} 37             {% endblock content %}
```

The bottom of the screen shows a terminal window with the following log output:

```
[24/OCT/2016 14:44:40] "GET /admin/mycontacts/contact/9/change/ HTTP/1.1" 200 7002  
[24/OCT/2016 14:44:48] "GET /admin/jsi18n/ HTTP/1.1" 200 3217  
[24/OCT/2016 14:49:16] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 6260  
[24/OCT/2016 14:49:17] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
```

Next, create a folder named ‘static’ in your application



The screenshot shows the Cloud9 IDE interface with a project named 'my_project'. The 'Workspace' sidebar on the left lists files and folders: __init__.py, settings.py, urls.py, wsgi.py, mycontacts (which is selected), migrations, templates, add.html, background.html, show.html, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py, db.sqlite3, manage.py, and README.md. The main editor window displays an HTML file named 'show.html' with code related to Bootstrap and static files. A context menu is open over the 'mycontacts' folder, listing options: Open, Download, Run, Preview, Refresh, Rename, Delete, Cut, Copy, Paste, Duplicate, Add to Favorites, Open Terminal Here, Search In This Folder, New File, and New Folder. The bottom of the screen shows a terminal window with log entries from October 24, 2016, at 14:44:40, 14:44:48, 14:49:16, and 14:49:17.

```
{% load staticfiles %}
<!DOCTYPE HTML>
<html lang="en-US">
<head>
    <meta charset="UTF-8">
    <!-->
        <meta http-equiv="X-UA-Compatible" content="IE=11; IE=10; IE=9; IE=8; IE=7; IE=EDGE" />
    <!-- Bootstrap theme-->
    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css" integrity="sha512-dTfge/zgoMYpP7QbHy4gWjxGZUcIUEoJ3qOJi6PvBjzRYlGbYyYm2tqEj/2" rel="stylesheet">
    <!-- Optional theme -->
    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap-theme.min.css" integrity="sha384-aUGj/X2zp5rLcbB" rel="stylesheet">
    <!-->
    <link href="{% static 'css/show.css' %}" rel="stylesheet">
    <link href="{% static 'css/add.css' %}" rel="stylesheet">
    <!-->
    <script type="text/javascript" src="//code.jquery.com/jquery-1.11.3.min.js"></script>
    <script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
    <script type="text/javascript" src="{% static 'js/show.js' %}"></script>
    <script type="text/javascript" src="{% static 'js/add.js' %}"></script>
<title>My Contacts </title>
```

```
[24/OCT/2016 14:44:40] "GET /admin/mycontacts/contact/9/change/ HTTP/1.1" 200 7002
[24/OCT/2016 14:44:48] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[24/OCT/2016 14:49:16] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 6260
[24/OCT/2016 14:49:17] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
```

Remember? ‘static’ is used to contain all of your CSS files and Javascript files

The screenshot shows the Cloud9 IDE interface with a Python project named "my_project". The project structure is as follows:

- my_project (root)
 - my_project (subdir)
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
 - mycontacts
 - migrations
 - static (highlighted with a red border)
 - css
 - show.css
 - add.css
 - js
 - show.js
 - add.js
 - templates
 - mycontacts

The "static" directory is highlighted with a red border. The "show.html" file in the templates folder contains code that includes static files:

```
{% load staticfiles %}
<!DOCTYPE HTML>
<html lang="en-US">
<head>
    <meta charset="UTF-8">
    <!--X-UA-Compatible content="IE=11; IE=10; IE=9; IE=8; IE=7; IE=EDGE" /-->
    <!--rap theme-->
    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css" integrity="sha512-dTfge/zgoMYpP7QbHy4gMElvjyZNanJqFwQ5cVb6e6BvT/5J37mP1H6zjyA/4Q+EQKxhpo7涵" rel="stylesheet">
    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap-theme.min.css" integrity="sha512-aUGj/X2zp5rLCbB" rel="stylesheet">
    <link href="{% static 'css/show.css' %}" rel="stylesheet">
    <link href="{% static 'css/add.css' %}" rel="stylesheet">
    <script src="//code.jquery.com/jquery-1.11.3.min.js"></script>
    <script src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
    <script src="{% static 'js/show.js' %}"></script>
    <script src="{% static 'js/add.js' %}"></script>
```

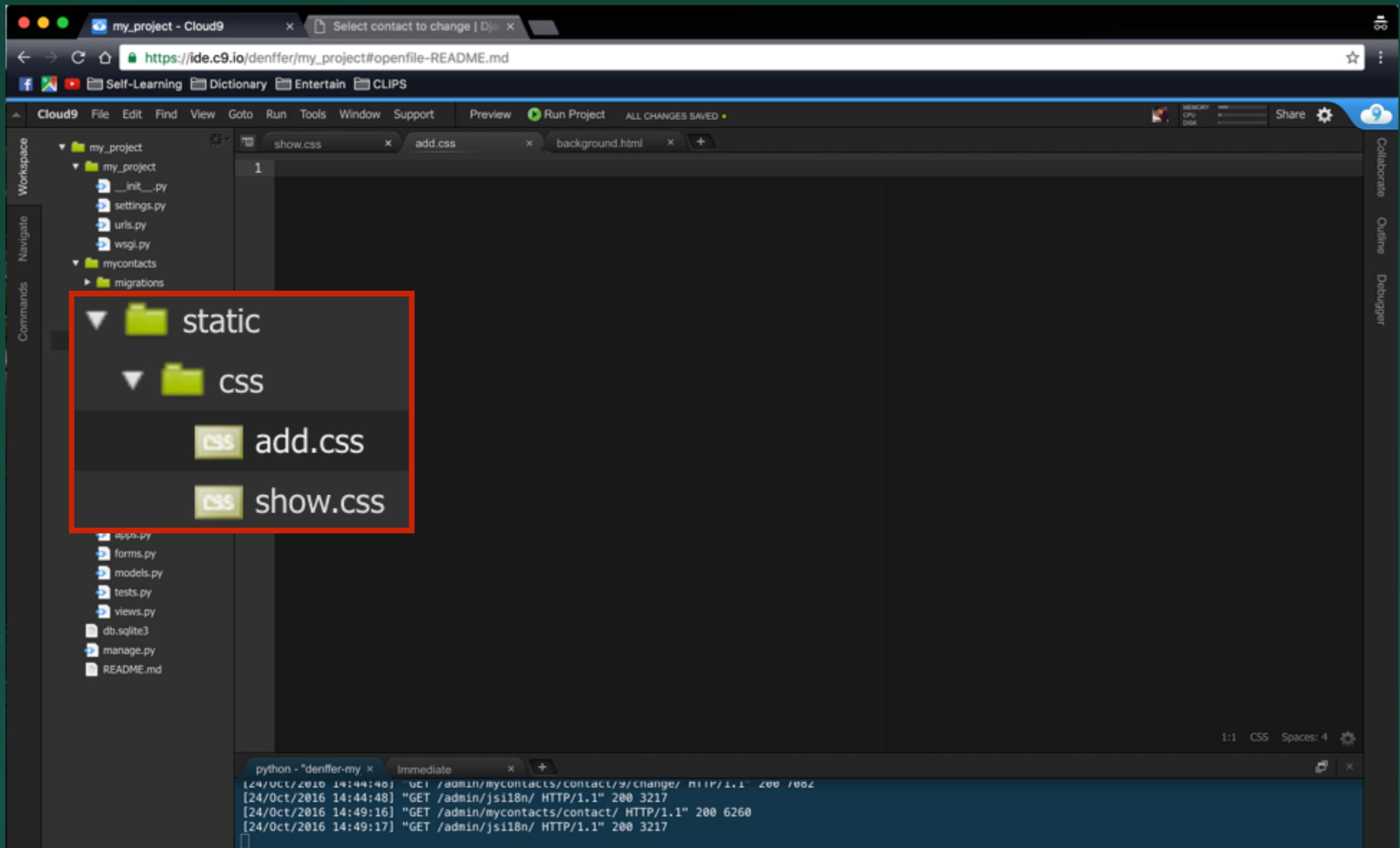
The bottom right corner of the code editor shows "43:8 HTML Spaces: 4".

The bottom panel shows a terminal window with the following log output:

```
[24/OCT/2016 14:44:40] "GET /admin/mycontacts/contact/9/change/ HTTP/1.1" 200 7002
[24/OCT/2016 14:44:48] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[24/OCT/2016 14:49:16] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 6260
[24/OCT/2016 14:49:17] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
```

Next, create two folders named ‘css’ and ‘js’
in the folder ‘static’

Create ‘show.css’ & ‘add.css’ in the folder ‘css’



Respectively copy and paste the codes in the CSS files from the repository you cloned from Github

The screenshot shows the Cloud9 IDE interface. The left sidebar displays the project structure:

```
my_project
  my_project
    __init__.py
    settings.py
    urls.py
    wsgi.py
  mycontacts
    migrations
    static
      css
        add.css
        show.css
      js
    templates
      mycontacts
        add.html
        background.html
        show.html
      __init__.py
      admin.py
      apps.py
      forms.py
      models.py
      tests.py
      views.py
    db.sqlite3
    manage.py
  README.md
```

The main workspace shows the content of the `show.css` file:

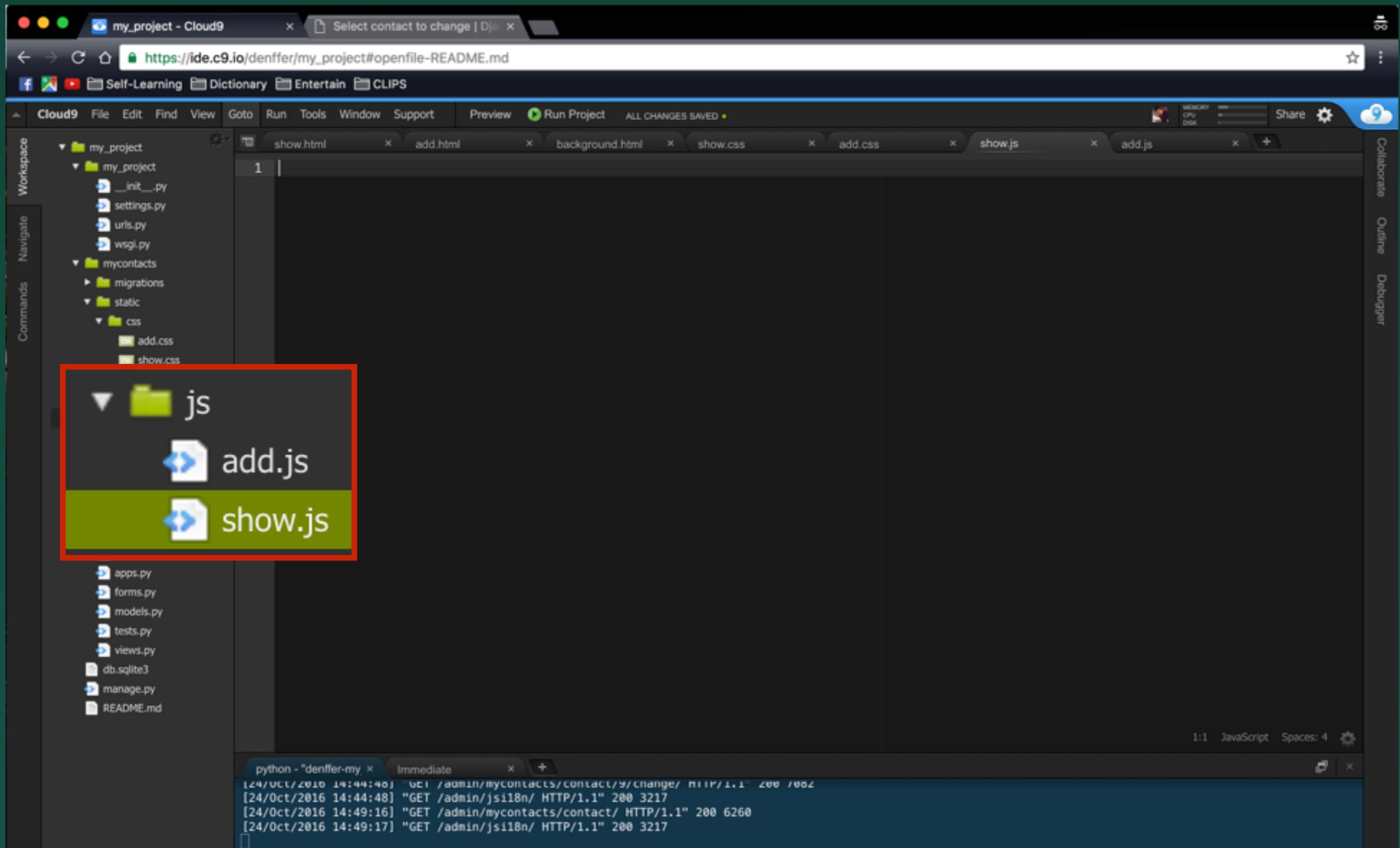
```
1  textarea {
2    resize: none;
3  }
4
5  .container-add{
6    padding-top: 15%;
7    padding-left: 15%;
8  }
9
10 .form-label {
11   font-size: 13px;
12   color: #083B39;
13   margin: 0;
14   display: block;
15   opacity: 1;
16   -webkit-transition: .333s ease top, .333s ease opacity;
17   transition: .333s ease top, .333s ease opacity;
18 }
19
20 .form-control {
21   border-radius: 0;
22   border-color: #ccc;
23   border-width: 0 0 2px 0;
24   border-style: none none solid none;
25   box-shadow: none;
26 }
27
28 .form-control:focus {
29   box-shadow: none;
30   border-color: #083B39;
31 }
32
33 .js-hide-label {
34   opacity: 0;
35 }
36
```

The bottom status bar indicates the file is 35:2 CSS with 4 spaces.

The terminal window at the bottom shows the following log output:

```
[24/Oct/2016 14:44:40] "GET /admin/mycontacts/contact/9/change/ HTTP/1.1" 200 7002
[24/Oct/2016 14:44:48] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[24/Oct/2016 14:49:16] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 6260
[24/Oct/2016 14:49:17] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
```

Create ‘show.js’ & ‘add.js’ in the folder ‘js’



Respectively copy and paste the codes in the JS files from the repository you cloned from Github

The screenshot shows the Cloud9 IDE interface. On the left, the workspace sidebar displays a project structure for 'my_project' containing files like __init__.py, settings.py, urls.py, wsgi.py, mycontacts, static/css/add.css, static/css/show.css, js/add.js, js/show.js, templates/mycontacts/add.html, templates/mycontacts/background.html, templates/mycontacts/show.html, __init__.py, admin.py, apps.py, forms.py, models.py, tests.py, views.py, db.sqlite3, manage.py, and README.md. The 'show.js' file is currently selected and open in the main editor area. The code in show.js is as follows:

```
$document).ready(function() {
    // Test for placeholder support
    $support.placeholder = (function(){
        var i = document.createElement('input');
        return 'placeholder' in i;
    });

    // Hide labels by default if placeholders are supported
    if($support.placeholder) {
        $('.form-label').each(function(){
            $(this).addClass('js-hide-label');
        });

        // Code for adding/removing classes here
        $('.form-group').find('input, textarea').on('keyup blur focus', function(e){

            // Cache our selectors
            var $this = $(this),
                $parent = $this.parent().find("label");

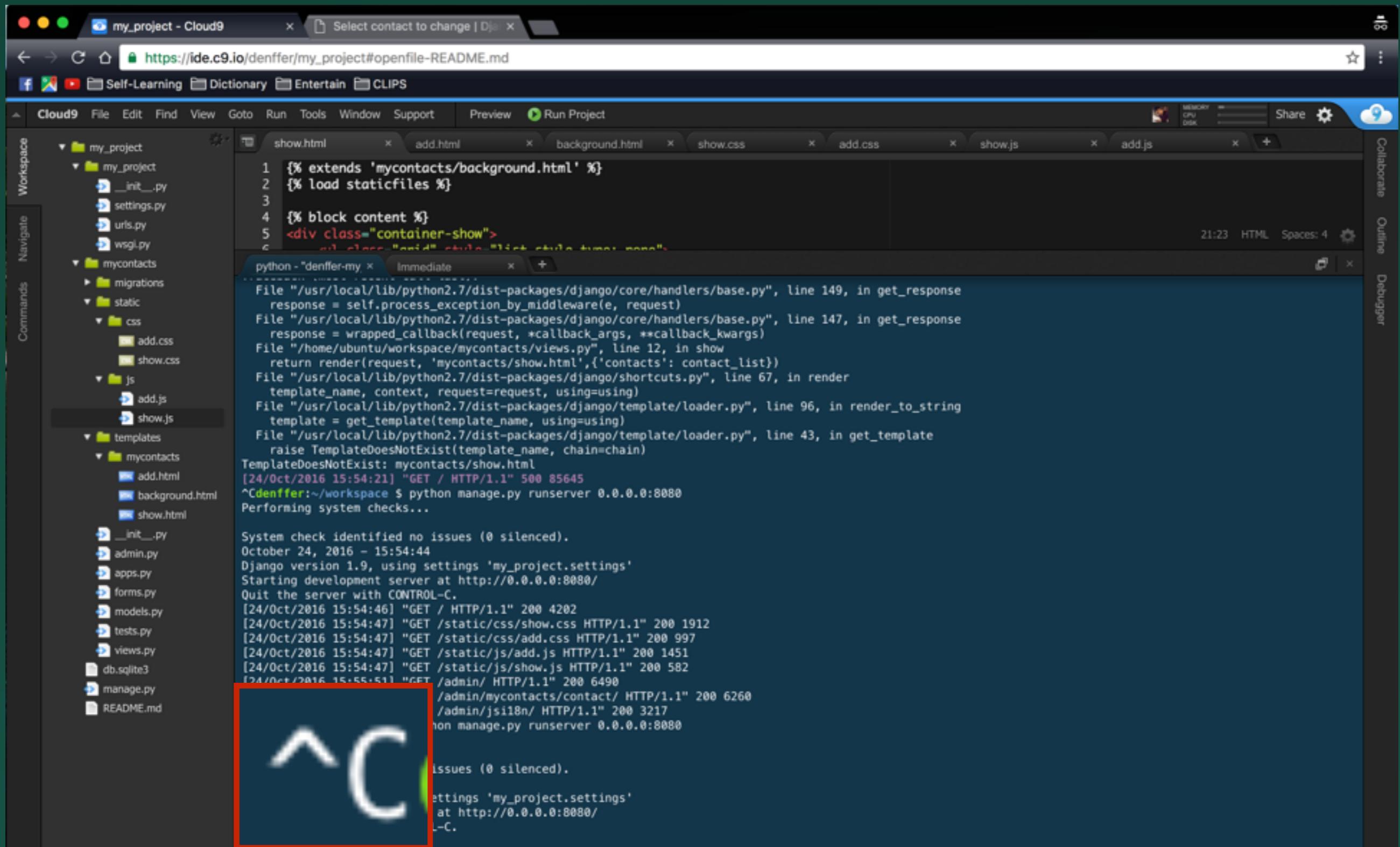
            if (e.type == 'keyup') {
                if( $this.val() == '' ) {
                    $parent.addClass('js-hide-label');
                } else {
                    $parent.removeClass('js-hide-label');
                }
            }
            else if (e.type == 'blur') {
                if( $this.val() == '' ) {
                    $parent.addClass('js-hide-label');
                }
                else {
                    $parent.removeClass('js-hide-label').addClass('js-unhighlight-label');
                }
            }
            else if (e.type == 'focus') {

```

At the bottom of the screen, there is a terminal window showing log output:

```
[24/OCT/2016 14:44:40] "GET /admin/mycontacts/contact/9/change/ HTTP/1.1" 200 7002
[24/OCT/2016 14:44:48] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[24/OCT/2016 14:49:16] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 6260
[24/OCT/2016 14:49:17] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
```

Go to command line. Hit ‘control’ + ‘c’ to kill the server



my_project - Cloud9 Select contact to change | Django my_project

https://ide.c9.io/denffer/my_project#openfile-README.md

Cloud9 File Edit Find View Goto Run Tools Window Support Preview Run Project

Workspace Navigate Commands

Cloud9 Cloud9 Cloud9 Cloud9

File Edit Find View Goto Run Tools Window Support Preview Run Project

Cloud9 Cloud9 Cloud9 Cloud9

Workspace my_project my_contacts static js templates my_contacts add.html background.html show.html __init__.py admin.py apps.py forms.py models.py tests.py views.py db.sqlite3 manage.py README.md

show.html add.html background.html show.css add.css show.js add.js

python - "denffer-my" Immediate

```
1  {% extends 'mycontacts/background.html' %} 2  {% load staticfiles %} 3  4  {% block content %} 5  <div class="container-show"> 6      <div class="row" style="background-color: #f2f2f2; height: 144px; margin-bottom: 20px;"> 7  File "/usr/local/lib/python2.7/dist-packages/django/core/handlers/base.py", line 149, in get_response 8      response = self.process_exception_by_middleware(e, request) 9  File "/usr/local/lib/python2.7/dist-packages/django/core/handlers/base.py", line 147, in get_response 10     response = wrapped_callback(request, *callback_args, **callback_kwargs) 11  File "/home/ubuntu/workspace/mycontacts/views.py", line 12, in show 12      return render(request, 'mycontacts/show.html', {'contacts': contact_list}) 13  File "/usr/local/lib/python2.7/dist-packages/django/shortcuts.py", line 67, in render 14      template_name, context, request=request, using=using) 15  File "/usr/local/lib/python2.7/dist-packages/django/template/loader.py", line 96, in render_to_string 16      template = get_template(template_name, using=using) 17  File "/usr/local/lib/python2.7/dist-packages/django/template/loader.py", line 43, in get_template 18      raise TemplateDoesNotExist(template_name, chain=chain) 19  TemplateDoesNotExist: mycontacts/show.html 20  [24/Oct/2016 15:54:21] "GET / HTTP/1.1" 500 85645 21  ^Cdenffer:~/workspace $ python manage.py runserver 0.0.0.0:8080 22  Performing system checks... 23  System check identified no issues (0 silenced). 24  October 24, 2016 - 15:54:44 25  Django version 1.9, using settings 'my_project.settings' 26  Starting development server at http://0.0.0.0:8080/ 27  Quit the server with CONTROL-C. 28  [24/Oct/2016 15:54:46] "GET / HTTP/1.1" 200 4202 29  [24/Oct/2016 15:54:47] "GET /static/css/show.css HTTP/1.1" 200 1912 30  [24/Oct/2016 15:54:47] "GET /static/css/add.css HTTP/1.1" 200 997 31  [24/Oct/2016 15:54:47] "GET /static/js/add.js HTTP/1.1" 200 1451 32  [24/Oct/2016 15:54:47] "GET /static/js/show.js HTTP/1.1" 200 582 33  [24/Oct/2016 15:55:51] "GET /admin/ HTTP/1.1" 200 6490 34  /admin/mycontacts/contact/ HTTP/1.1" 200 6260 35  /admin/jsi18n/ HTTP/1.1" 200 3217 36  non manage.py runserver 0.0.0.0:8080 37  Issues (0 silenced). 38  settings 'my_project.settings' 39  at http://0.0.0.0:8080/ 40  L-C.
```

Start up the server again!

The screenshot shows the Cloud9 IDE interface. On the left is the workspace sidebar with project files like `my_project`, `mycontacts`, `static`, `js`, `templates`, and `db.sqlite3`. The main area has several tabs open: `show.html`, `add.html`, `background.html`, `show.css`, `add.css`, `show.js`, and `add.js`. Below these tabs is a terminal window displaying command-line output.

Cloud9 Help (highlighted with a red box)

Your code is running at <https://my-project-denffer.c9users.io>

1 \$ python manage.py runserver 0.0.0.0:8080

2 Your code is running at <https://my-project-denffer.c9users.io>

```
python - "denffer-my" Immediate +  
File "/usr/local/lib/python2.7/dist-packages/django/template/  
    template = get_template(template_name, using=using)  
File "/usr/local/lib/python2.7/dist-packages/django/template/  
    raise TemplateDoesNotExist(template_name, chain=chain)  
TemplateDoesNotExist: mycontacts/show.html  
[24/Oct/2016 15:54:21] "GET / HTTP/1.1" 500 85645  
^Denffer:~/workspace $ python manage.py runserver 0.0.0.0:8080  
Performing system checks...  
  
System check identified no issues (0 silenced).  
October 24, 2016 - 15:54:44  
Django version 1.9, using settings 'my_project.settings'  
Starting development server at http://0.0.0.0:8080/  
Quit the server with CONTROL-C.  
[24/Oct/2016 15:54:46] "GET / HTTP/1.1" 200 4202  
[24/Oct/2016 15:54:47] "GET /static/css/show.css HTTP/1.1" 200 1912  
[24/Oct/2016 15:54:47] "GET /static/css/add.css HTTP/1.1" 200 997  
[24/Oct/2016 15:54:47] "GET /static/js/add.js HTTP/1.1" 200 1451  
[24/Oct/2016 15:54:47] "GET /static/js/show.js HTTP/1.1" 200 582  
[24/Oct/2016 15:55:51] "GET /admin/ HTTP/1.1" 200 6490  
[24/Oct/2016 15:55:54] "GET /admin/mycontacts/contact/ HTTP/1.1" 200 6260  
[24/Oct/2016 15:55:54] "GET /admin/jsi18n/ HTTP/1.1" 200 3217  
^Denffer:~/workspace $ python manage.py runserver 0.0.0.0:8080  
Performing system checks...  
  
System check identified no issues (0 silenced).  
October 24, 2016 - 15:57:33  
Django version 1.9, using settings 'my_project.settings'  
Starting development server at http://0.0.0.0:8080/  
Quit the server with CONTROL-C.  
denffer:~/workspace  
Performing system checks...  
  
System check identified no issues (0 silenced).  
October 24, 2016 - 16:01:50  
Django version 1.9, using settings 'my_project.settings'  
Starting development server at http://0.0.0.0:8080/  
Quit the server with CONTROL-C.
```

Ladies and gentlemen ...



It worked! Feel free to play around!

The screenshot shows a web browser window titled "My Contacts" with the URL <https://my-project-denffer.c9users.io>. The page displays six contact entries arranged in two rows of three. Each contact card includes the contact's name, their relationship status (e.g., Friend or brother), their phone number, and their email address.

Row	Column	Contact Name	Relationship	Phone Number	Email Address
1	1	Tom	Friend	0912345678	Tom@gmail.com
1	2	TomTom	Friend	0912345678	TomTom@gmail.com
1	3	Tom^3	Friend	0912345678	Tom^3@gmail.com
2	1	John	brother	0987654321	John@gmail.com
2	2	JohnJohn	Friend	0987654321	JohnJohn@gmail.com
2	3	John^3	brother	0987654321	John^3@gmail.com

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
- 4. Tutorial**
5. Conclusion

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
4. Tutorial
5. Conclusion

Conclusion

Alignment & Consistency

Two basic rules for a successful design

Conclusion

Error is your friend

Learn to embrace it

Conclusion

Simple is good

Try not to squeeze everything in a single page

Conclusion

Don't play hero

Learn to cooperate with one another

Outline

1. Introduction
2. Basic Knowledge
3. Technical Details
4. Tutorial
5. Conclusion

Thank you

Presented by Denffer

Department of Computer Science, University of Taipei

Django

Build an organized web application

Presented by Denffer

2017@Department of Computer Science, University of Taipei