% django-admin

Available subcommands:

[python django-admin manage.py *subcommand*]

|  |  |
| --- | --- |
| **Subcommand** | |
| check | Check the Django project for potential problems |
| compilemessages | Compiles .po files to .mo files for use with builtin gettext support |
| createcachetable | Creates the tables needed to use the SQL cache backend |
| dbshell | Runs the command-line client for specified database or the default database |
| diffsettings | Displays differences between the current settings.py and Django's default settings |
| dumpdata | Output the contents of the database in the given format |
| flush | Removes all data from the database, including data added during migrations |
| inspectdb | Introspects the tables in a given database and outputs a Django model module |
| loaddata | Installs the named fixture(s) in the database |
| makemessages | Runs over the entire source tree of the current directory and pulls all strings marked for translation |
| makemigrations | Create new migration(s) for apps |
| migrate | Updates database schema. Manages both apps with migrations and those without |
| runserver | Starts a lightweight Web server for development |
| sendtestemail | Sends a test email to the email addresses specified as arguments |
| shell | Run a Python interactive interpreter |
| showmigrations | Shows all available migrations for the current project |
| sqlflush | Returns a list of SQL statements required to return all tables to the state they were in when installed |
| sqlmigrate | Prints the SQL statements for the named migration |
| sqlsequencereset | Prints the SQL statements for resetting sequences for the given app name(s) |
| squashmigrations | Squashes an existing set of migrations (from first until specified) into a single new one |
| startapp | Creates a Django app directory structure for the given app name |
| startproject | Creates a Django project directory structure for the given project name |
| testserver | Runs a development server with data from the given fixture(s) |

**Create basic web setup:**

1. Bring up PyCharm (currently Version 2019.3.5)
2. Select Create New Project

Graphical user interface, application, Teams

Description automatically generated

1. Enter Location. Use existing interpreter (Python 3.8). Click Create.  
     
     
   Graphical user interface, text, application

   Description automatically generated

|  |
| --- |
| $ cd ~/src  $ django-admin startproject [projectname]  $ cd [projectname]  $ python manage.py runserver |

**Testing basic setup:**

Using a browser, go to <http://localhost:8000>, and the following page should appear:

Diagram, text

Description automatically generated

Using a browser, go to <http://localhost:8000/admin> , and the following page should appear:

Graphical user interface, application, website

Description automatically generated

|  |  |
| --- | --- |
|  | $ cd ~/src  $ django-admin startproject [projectname]  $ cd [projectname  $ python manage.py startapp blog |

|  |
| --- |
| $ cd ~/src  $ django-admin startproject [projectname]  $ cd [projectname]  $ python manage.py startapp blog |

Configure django to use MAMP MySQL.

**DATABASES = {**

**'default': {**

**'ENGINE': 'django.db.backends.mysql',**

**'NAME': 'projectdb',**

**'USER': 'root',**

**'PASSWORD': 'root',**

**'HOST': '/Applications/MAMP/tmp/mysql/mysql.sock',**

**'PORT': '8888',**

}

}