Presents

Django Workshop

Rishabh Rathi

Aditya Sharma

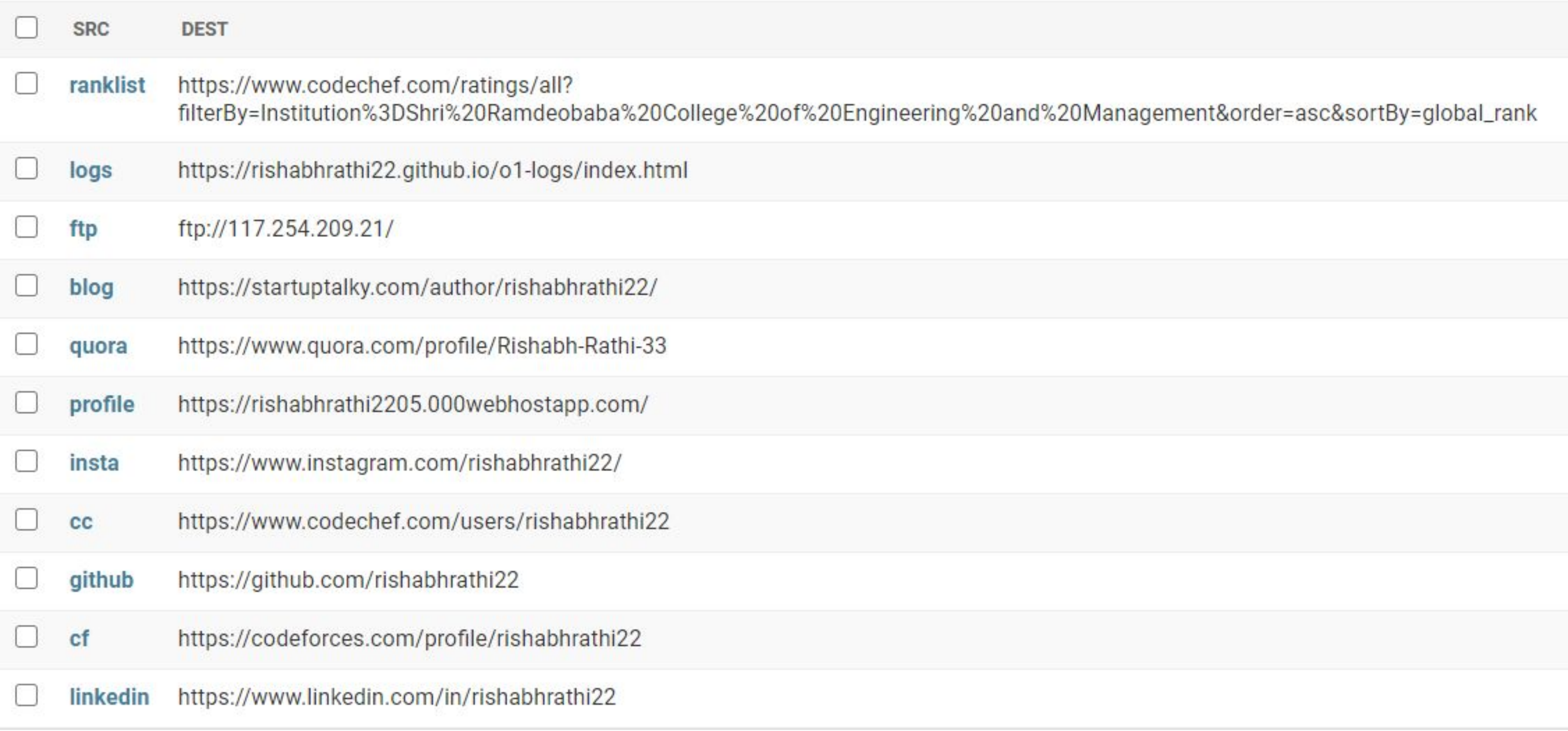
Today's Agenda

- Database Handling
- Django Models
- Django Admin
- Manage Static Files
- Complete URL Shortening view
- Redirect view
- Create basic analytics

DATABASE HANDLING

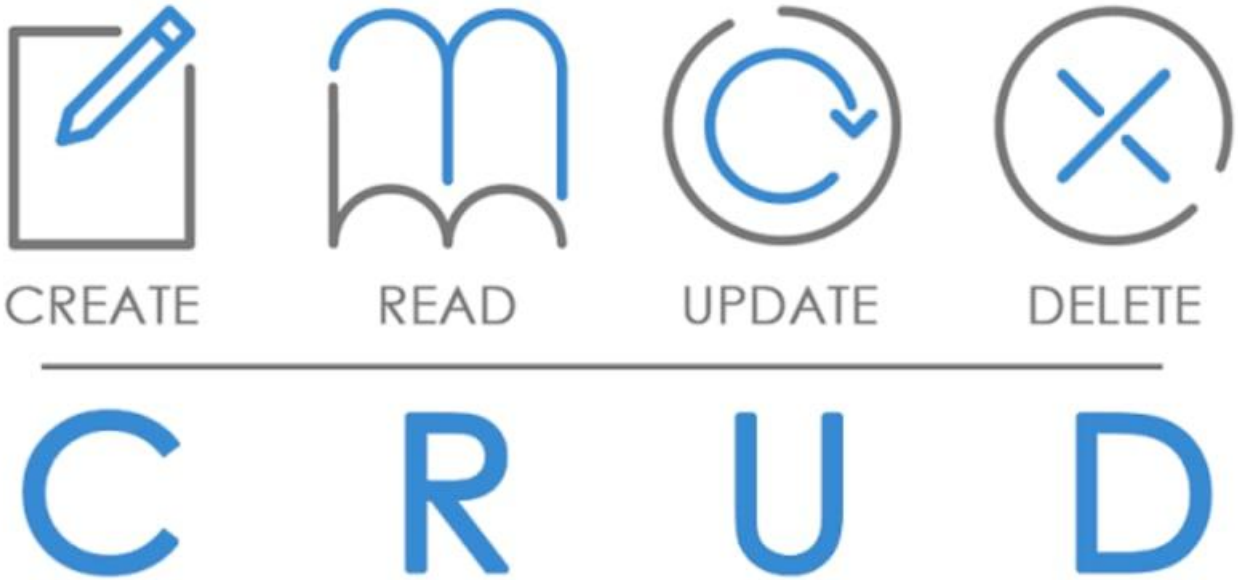
- We are working with Relational Database.
- A database consists of several tables.
- In simple terms, a database table can be thought of as consisting of rows and columns.
- Language used to interact with relational db - Structured Query Language (SQL).

Sample Database



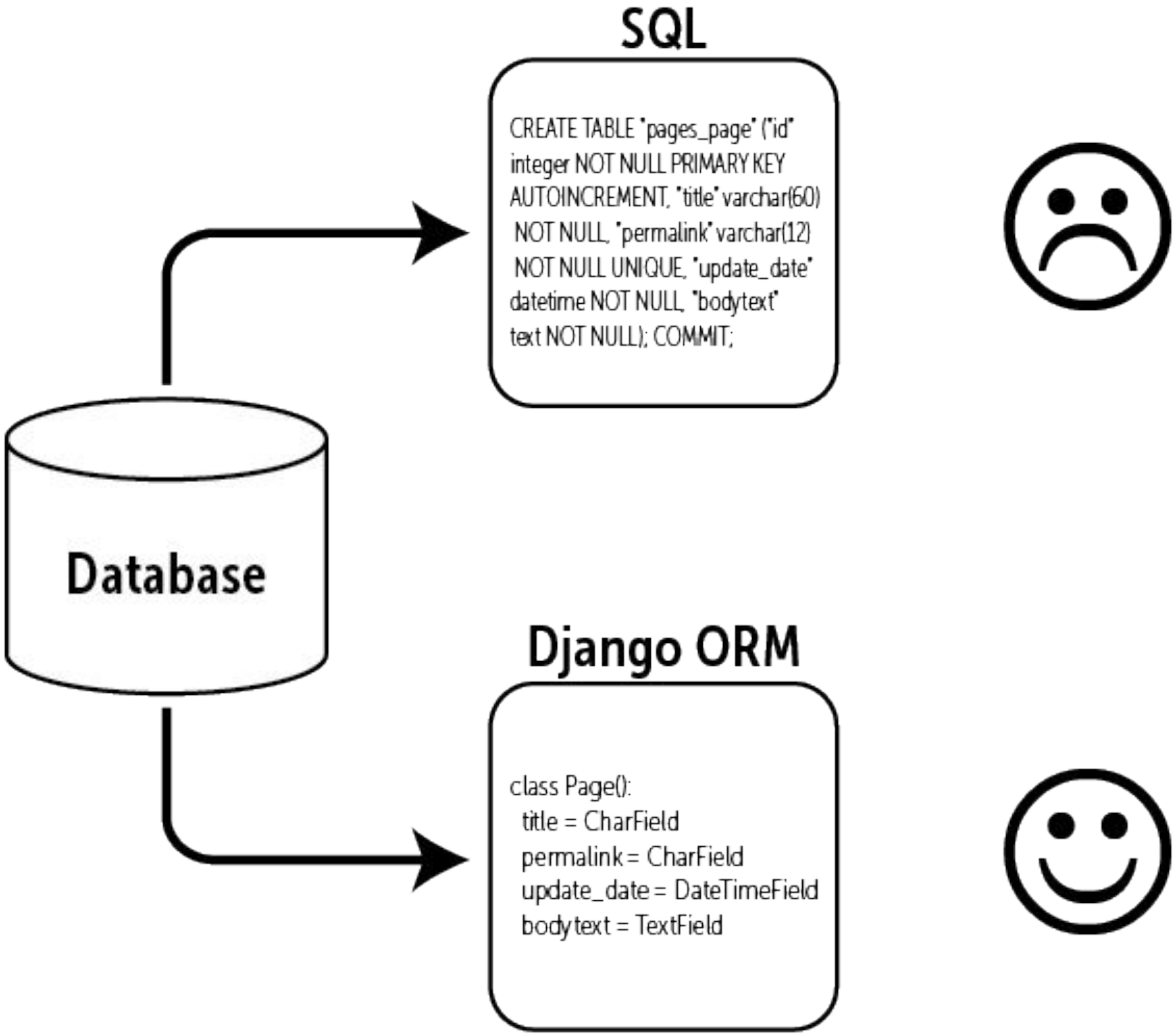
□	SRC	DEST
□	ranklist	https://www.codechef.com/ratings/all?filterBy=Institution%3DShri%20Ramdeobaba%20College%20of%20Engineering%20and%20Management&order=asc&sortBy=global_rank
□	logs	https://rishabhrathi22.github.io/o1-logs/index.html
□	ftp	ftp://117.254.209.21/
□	blog	https://startuptalky.com/author/rishabhrathi22/
□	quora	https://www.quora.com/profile/Rishabh-Rathi-33
□	profile	https://rishabhrathi2205.000webhostapp.com/
□	insta	https://www.instagram.com/rishabhrathi22/
□	cc	https://www.codechef.com/users/rishabhrathi22
□	github	https://github.com/rishabhrathi22
□	cf	https://codeforces.com/profile/rishabhrathi22
□	linkedin	https://www.linkedin.com/in/rishabhrathi22

Operations on Database



Don't know SQL? - Django ORM

- ORM is an acronym for the Object-Relational mapper.
- The ORM automates transmission of data between database and application, such that the developer need not write any SQL.
- Django is shipped with its own ORM (very efficient).
- Querysets are Django’s way to retrieve data from the database.
- A Queryset is a list of objects of a model. We use Querysets to filter and arrange our data.



Database

SQL

```
CREATE TABLE "pages_page" ("id"
integer NOT NULL PRIMARY KEY
AUTOINCREMENT, "title" varchar(60)
 NOT NULL, "permalink" varchar(12)
 NOT NULL UNIQUE, "update_date"
datetime NOT NULL, "bodytext"
text NOT NULL); COMMIT;
```

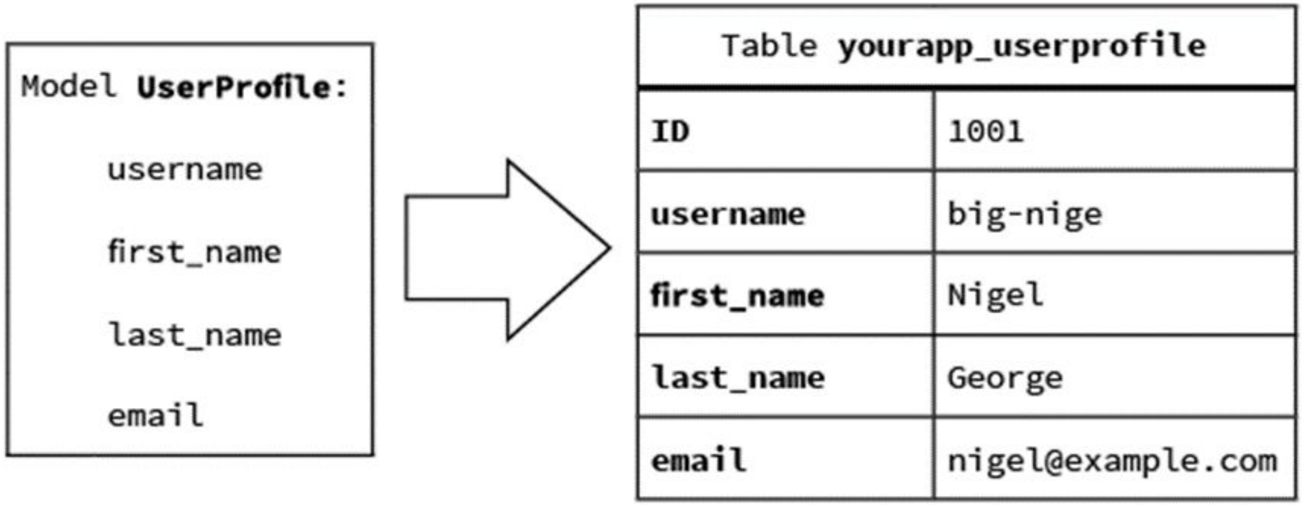
Django ORM

```
class Page():
  title = CharField
  permalink = CharField
  update_date = DateTimeField
  bodytext = TextField
```

Django Models

- A **Django model** is the built-in feature that Django uses to create tables, their fields, and various constraints. In short, Django Models is the SQL of Database one uses with Django.

- SQL (Structured Query Language) is complex and involves a lot of different queries for creating, deleting, updating or any other stuff related to database. Django models simplify the tasks and organize tables into models. Generally, each model maps to a single database table.



Model **UserProfile:**

username

first_name

last_name

email

Table **yourapp_userprofile**

ID	1001
username	big-nige
first_name	Nigel
last_name	George
email	nigel@example.com

Django Model Fields

Field Name	Description
BooleanField	A true/false field.
CharField	A field to store text-based values.
DateField	A date, represented in Python by a datetime.date instance
DateTimeField	It is used for date and time, represented in Python by a datetime.datetime instance.
EmailField	It is a CharField that checks that the value is a valid email address.
IntegerField	It is an integer field.
URLField	A CharField for a URL, validated by URLValidator.

How to create Models?

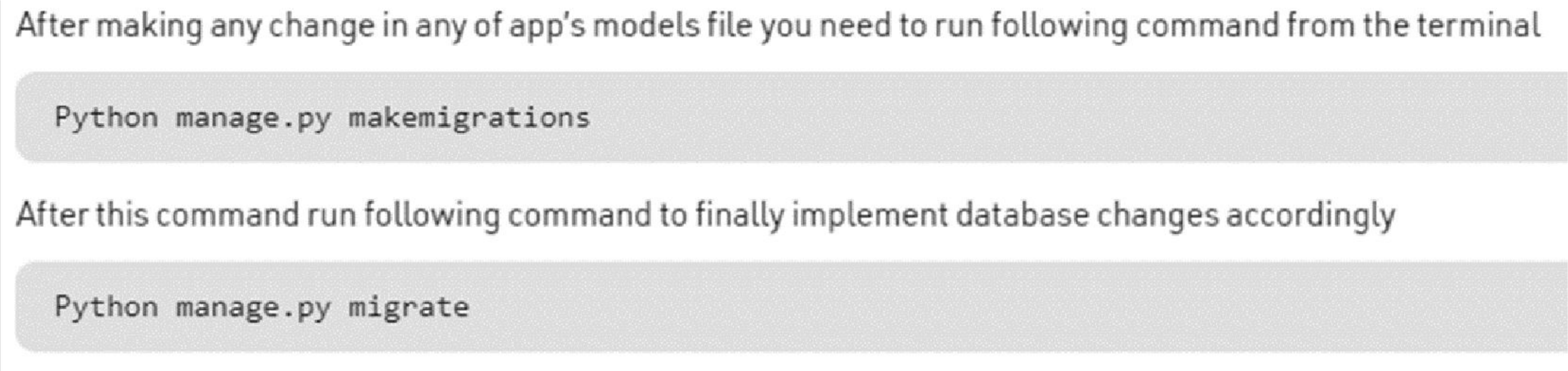
Step 1 : Add class in models.py

```
class URL_table(models.Model):
    long_url = models.URLField(max_length = 500)
    custom_name = models.CharField(max_length = 50, unique = True)
    created_date = models.DateField(auto_now_add = True)
    visit_count = models.IntegerField(default = 0)
```

Step 2 : Register our model to admin site (admin.py)

```
admin.site.register(URL_table)
```

MUST-DO Steps



After making any change in any of app's models file you need to run following command from the terminal

```
Python manage.py makemigrations
```

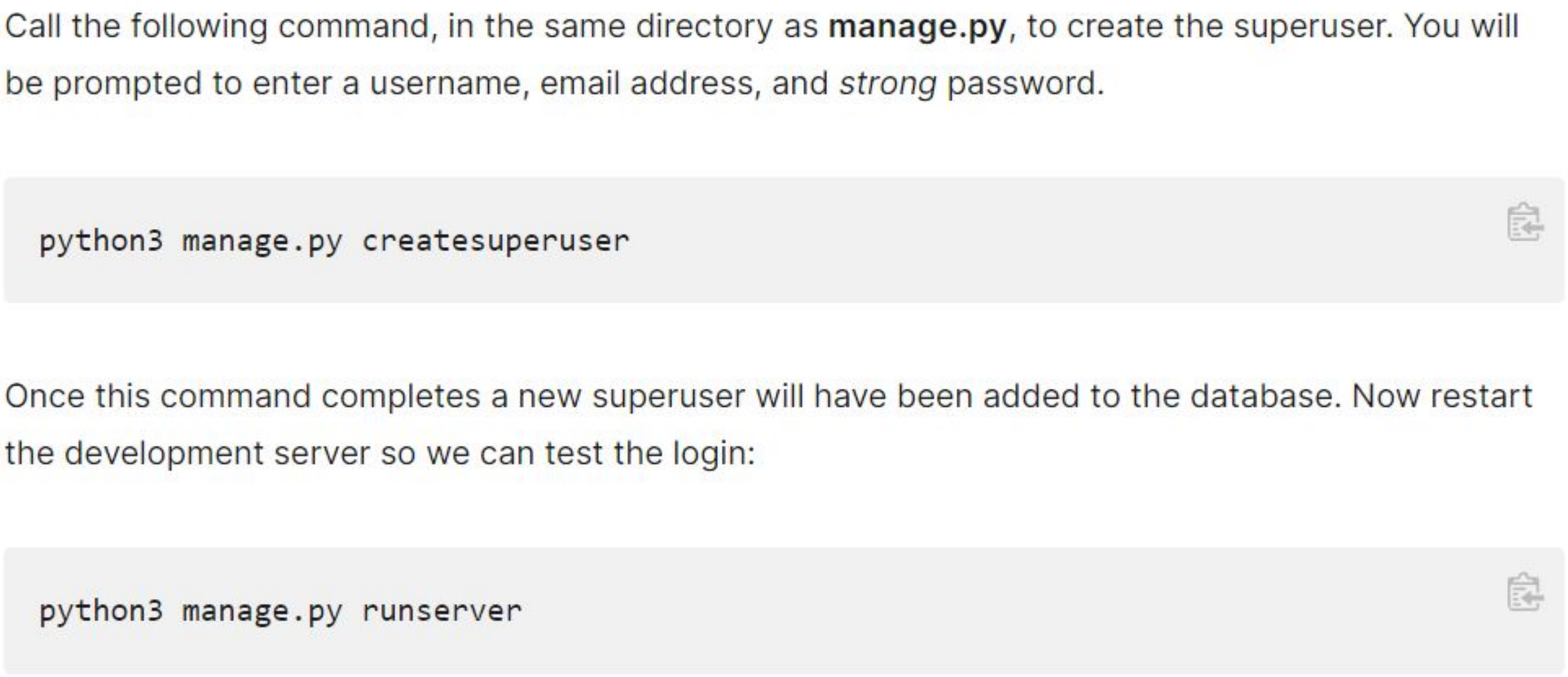
After this command run following command to finally implement database changes accordingly

```
Python manage.py migrate
```

Django Admin

- One of the most powerful parts of Django is the automatic admin interface.
- The Django admin application can use your models to automatically build a site area that you can use to create, view, update, and delete records.

Creating a Super User



Call the following command, in the same directory as **manage.py**, to create the superuser. You will be prompted to enter a username, email address, and *strong* password.

```
python3 manage.py createsuperuser
```

Once this command completes a new superuser will have been added to the database. Now restart the development server so we can test the login:

```
python3 manage.py runserver
```

Creating Objects (Adding Rows)

- To create an object, instantiate it using keyword arguments to the model class, then call save() to save it to the database.

```
>>> obj = URL_table(
      long_url = 'https://drive.google.com/file/d/1RsuU24ud',
      custom_name = 'day2video')

>>> obj.save( )
```

- This performs an INSERT SQL statement behind the scenes.

Read Rows

- Retrieving all objects - use the all( ) method

```
>>> all_entries = URL_table.objects.all()
```

- Retrieving specific objects with filters - use the filter( ) method

```
>> some_entries = URL_table.objects.filter(shorturl = 'myprofile')
```

- Retrieving a single object with get( )

```
>> one_entry = URL_table.objects.get(shorturl = 'myprofile')
```

*** If you know there is only one row that matches your query, you can use the get() method.**
If there is no row or there is more than 1 row matching the condition - It will give error.

Update Rows (objects)

- To save changes to an object that's already in the database, use save().

- Given a URLTable instance **obj** that has already been saved to the database, this example **changes its visit count** and updates its record in the database:

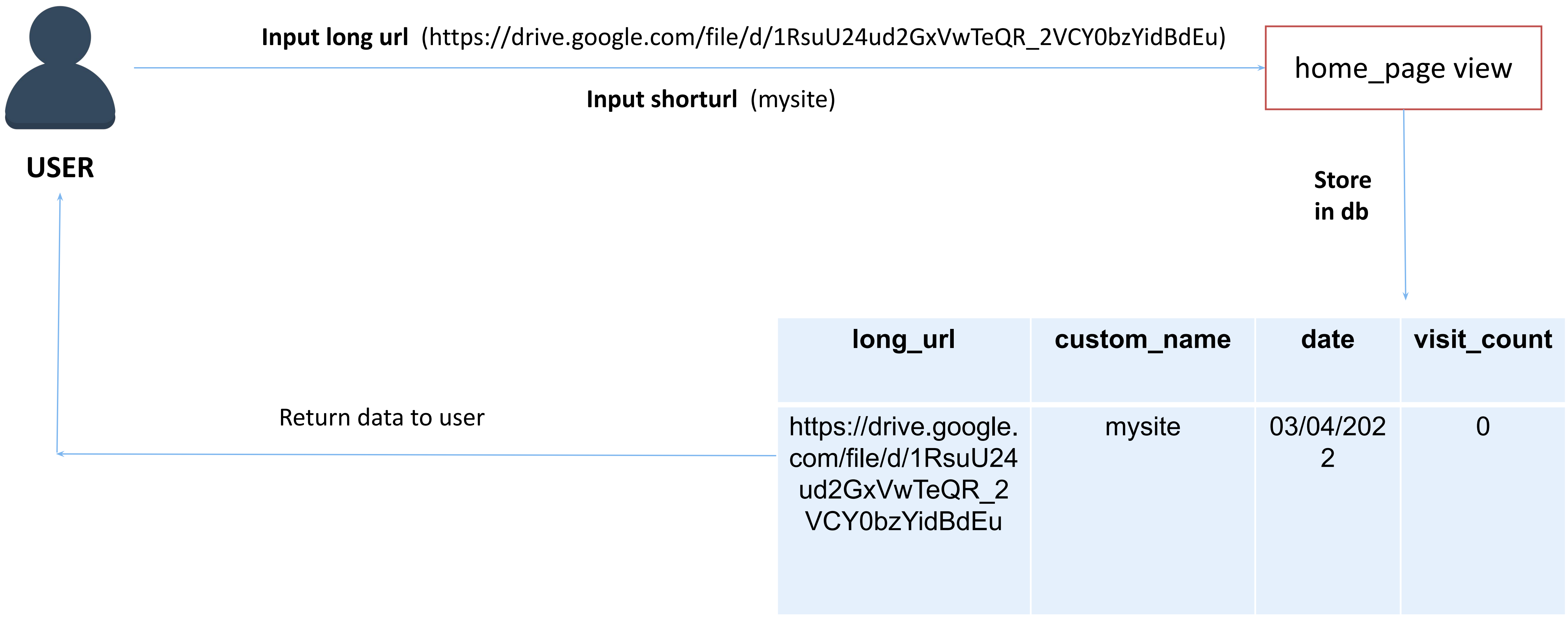
```
>>> obj.clicks = obj.clicks + 1
>>> obj.save( )
```

Delete Rows (objects)

- The delete method, conveniently, is named delete().
- This method immediately deletes the object and returns the number of objects deleted.
- Example: This code deletes all links with longurl 'http://google.com'.

```
>>> URLTable.objects.filter(longurl = 'http://google.com').delete()
```

View - Shorten



USER

Input long url (https://drive.google.com/file/d/1RsuU24ud2GxVwTeQR_2VCY0bzYidBdEu)

Input shorturl (mysite)

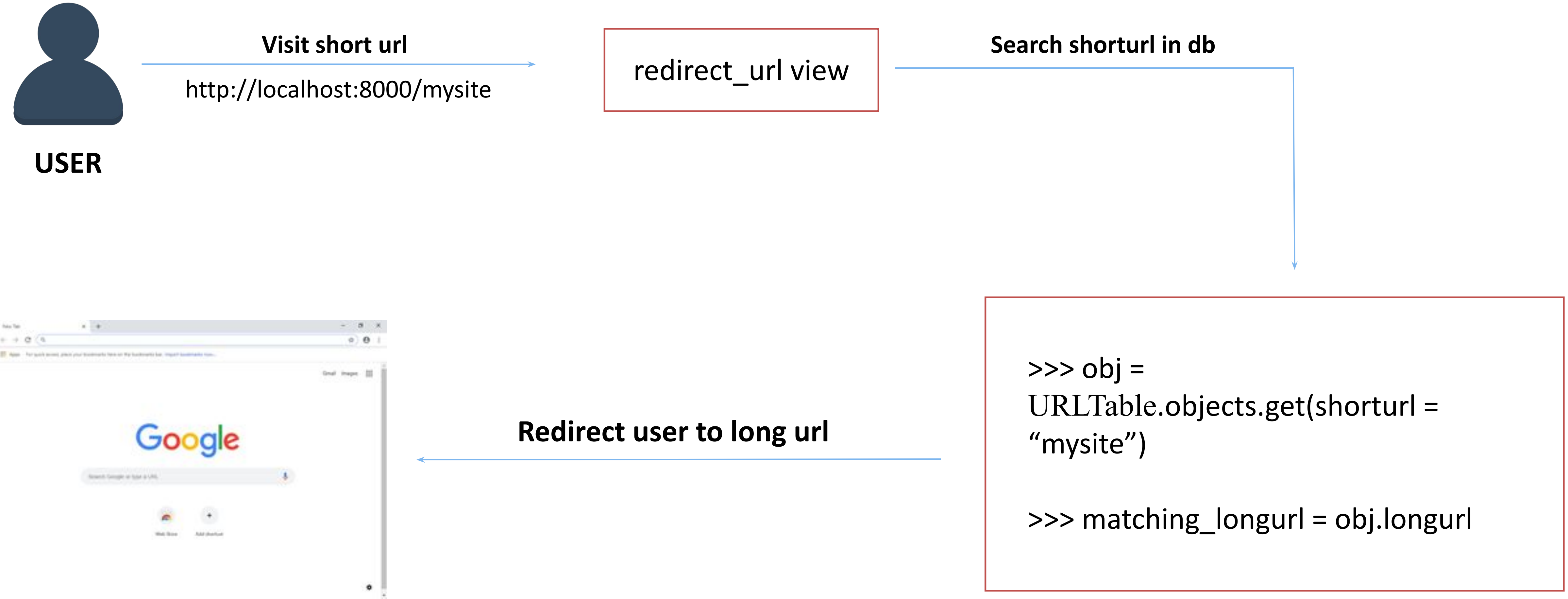
home_page view

Store in db

long_url	custom_name	date	visit_count
https://drive.google.com/file/d/1RsuU24ud2GxVwTeQR_2VCY0bzYidBdEu	mysite	03/04/2022	0

Return data to user

View - Redirect



USER

Visit short url

http://localhost:8000/mysite

redirect_url view

Search shorturl in db

```
>>> obj =
URLTable.objects.get(shorturl =
“mysite”)

>>> matching_longurl = obj.longurl
```

Redirect user to long url

Workshop Completion Certificate

ToDo - Post your Project video / Session screenshots on LinkedIn and tag o1codingclub.

- Recruiters will value your skills and work which you post on LinkedIn, it will be very much beneficial for your career

- You can use Workshop Completion Certificate in your resume

Surprise Task (Internship Opportunity)

- **ToDo -** Improve the website that we created (You can beautify frontend, you can add some more analytics apart from visit count, you can add any new feature you want)

- **Team Size** - Individual or 2 members

- **Deadline** - 6th November 2022 (Sunday), 11:59 PM.

- **To submit your entry**, post 10-15 secs screen recording video of your website on Linkedin and tag **o1codingclub.**

- Certificate of Appreciation to Top 5 Performers (individual or team).

- Internship opportunities with O(1) Coding Club if we like your work.

THANK YOU

SCAN ME

Share your feedback -
bit.ly/django-feedback