Object Relational Mapping

Recitation 6

February 20th, 2020

جامعة كارنيجي ميلون في قطر
Carnegie Mellon University Qatar

Outline

- Introduction to Object Relational Mapping
- ORM case study (1): JPA
- ORM case study (2): Django ORM

Outline

- Introduction to Object Relational Mapping
- ORM case study (1): JPA
- ORM case study (2): Django ORM

Building a web application…

Create Student

ID:

Name:

GPA:

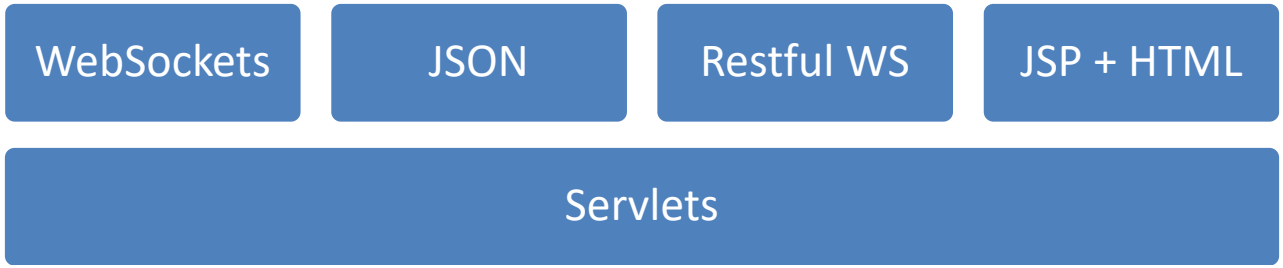
```
class Student
{
        int sid;
        String name;
        double gpa;
}
Student s = new Student (…)
```

Sid	Name	gpa
22	Adam	3.6
23	Joy	3.2
24	AG	4.0

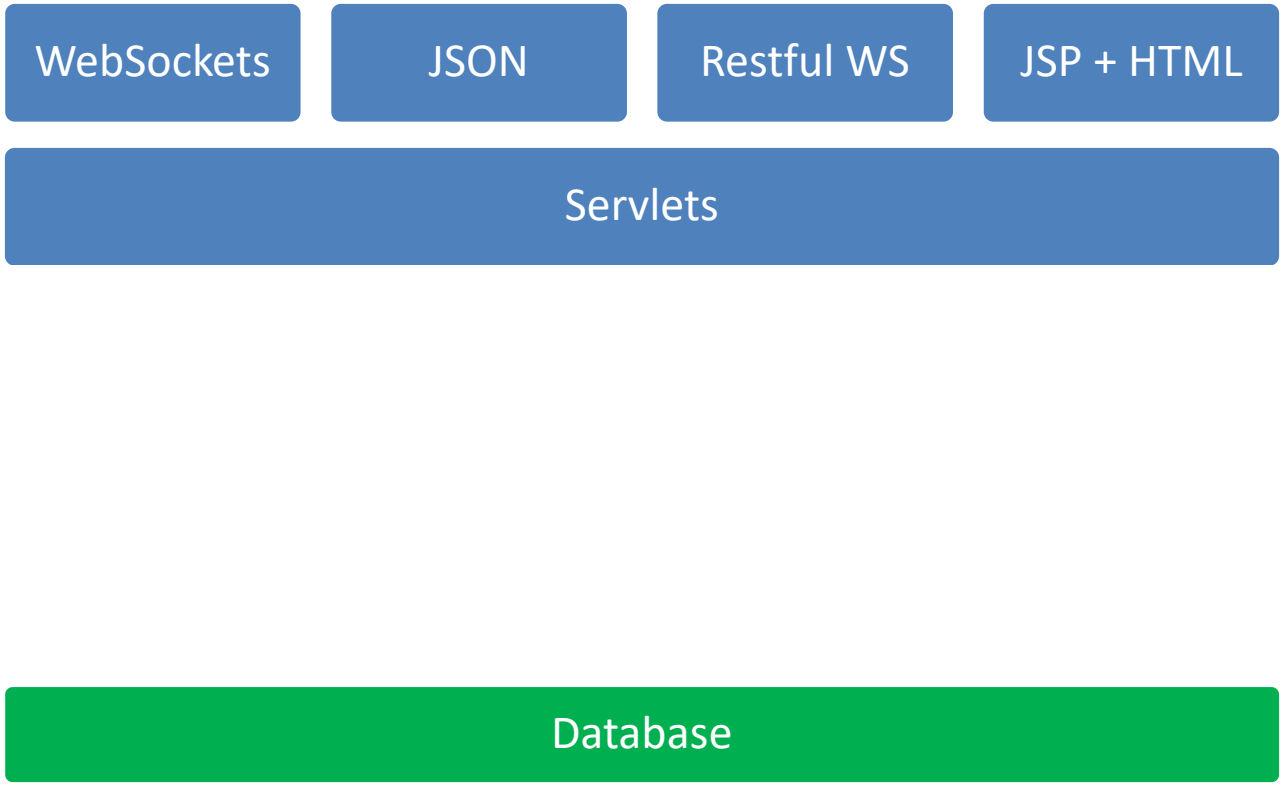
Building a web application…

WebSockets | JSON | Restful WS | JSP + HTML

Building a web application…



Building a web application…



Building a web application…

What happens here?

Building a web application…

```
class Student
{
        int sid;
        String name;
        double gpa;
}
Student s = new Student (…)
```

?

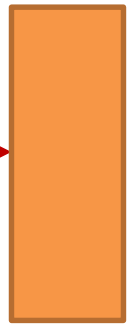
Sid	Name	gpa
22	Adam	3.6
23	Joy	3.2

What happens here?

Building a web application…

```
class Student
{
        int sid;
        String name;
        double gpa;
}
Student s = new Student (…)
```

JDBC
API



Sid	Name	gpa
22	Adam	3.6
23	Joy	3.2

What happens here?

JDBC Example:

```
Class.forName("org.postgresql.Driver");
connection = DriverManager.getConnection(jdbcURL,dbUser,dbPassword);
statement = connection.prepareStatement("INSERT INTO STUDENT (sid, name, gpa)
                                        VALUES ("22, "Adam", 3.6);");
ResultSet rs = statement.executeQuery();
```

Building a web application…

Static SQL code

What if we are inserting 100 records to two tables?

Student is an object, we need to break it down to an SQL code…

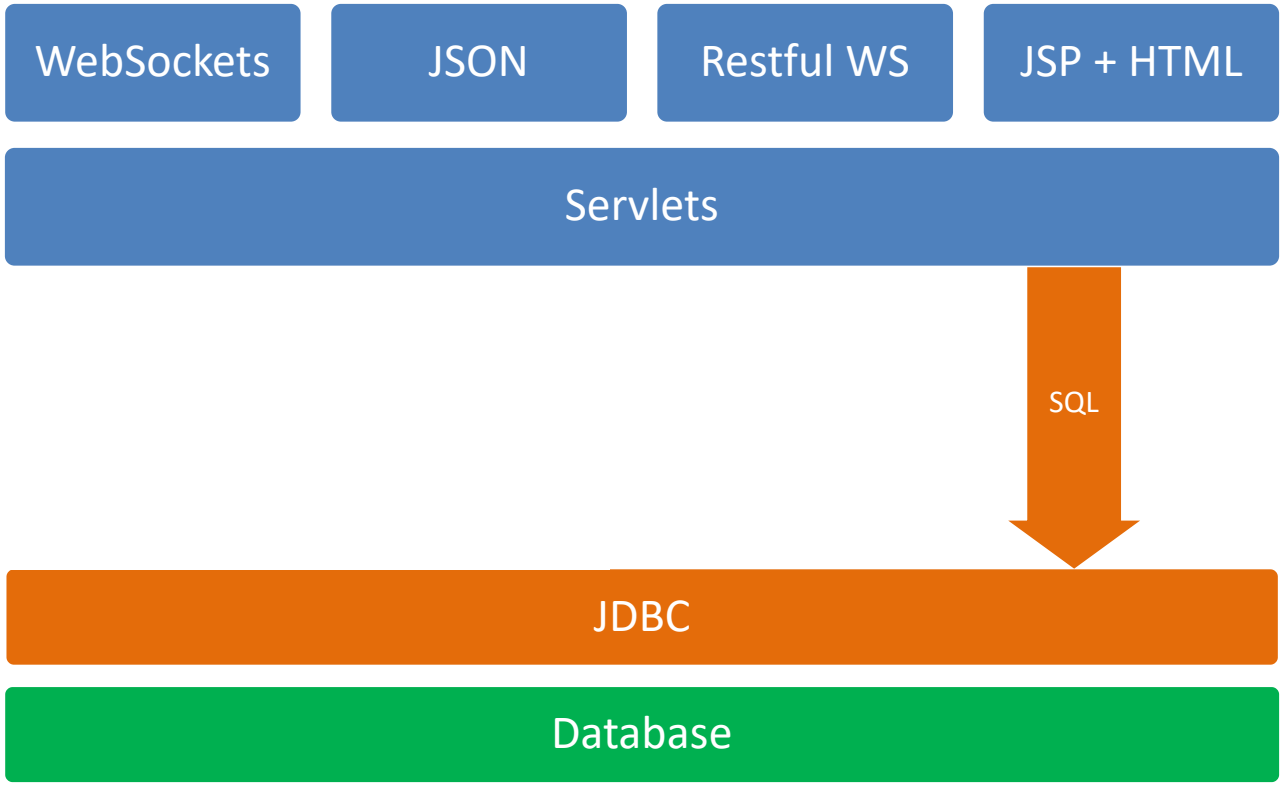
When I receive an SQL query result, how do I convert it back to an object?

What if I decide to change my database from SQL to Oracle?

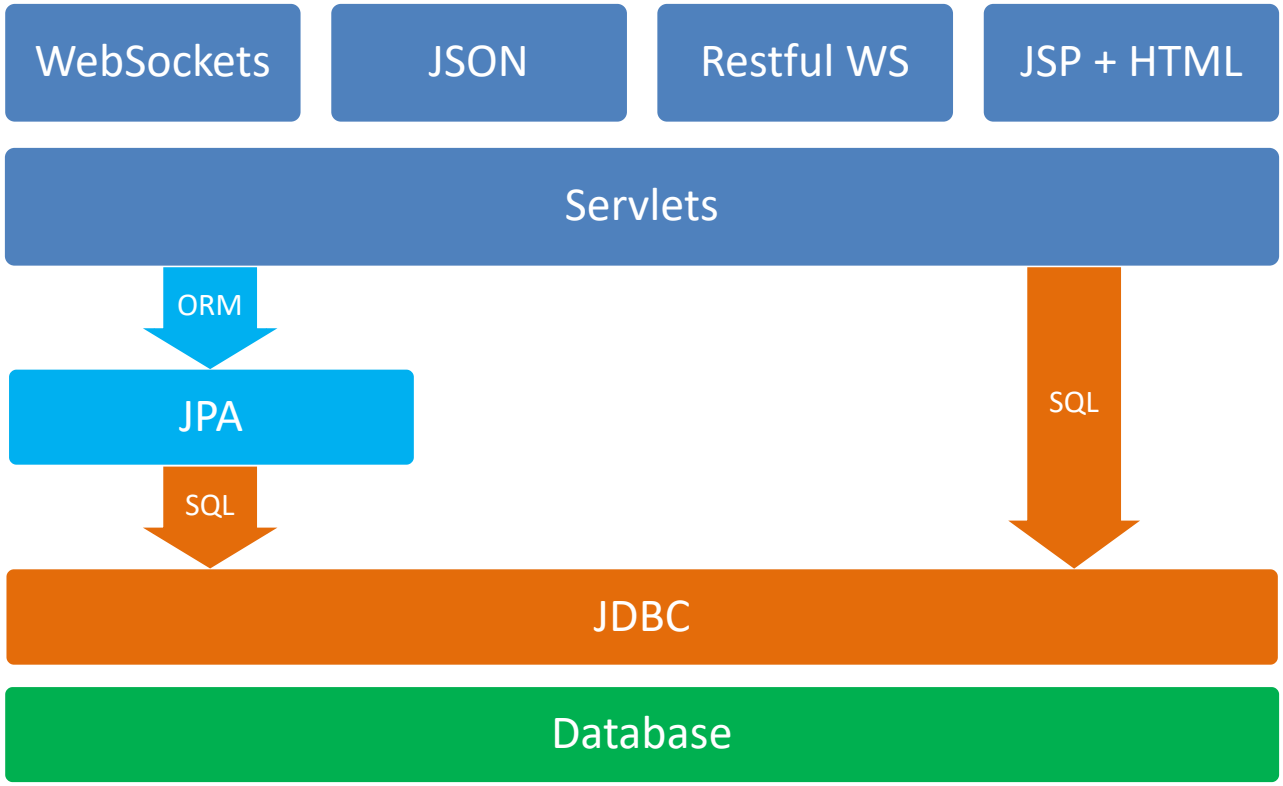
JDBC Example:

```
Class.forName(”org.postgresql.Driver”);
connection = DriverManager.getConnection(jdbcURL,dbUser,dbPassword);
statement = connection.prepareStatement(”INSERT INTO STUDENT (sid, name, gpa)
                                        VALUES (“22, “Adam”, 3.6);”);
ResultSet rs = statement.executeQuery();
```

Building a web application…



Building a web application…



“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A)⇔ Inheritance
Relationship ⇔ Composition / Aggregation

```
Sailors (sid: integer,
sname: string, rating:
integer, age: real)
```

Sid	Sname	Rating	Age

```
class Sailors {
        int sid;
        String sname;
        int rating;
        double age;
}
```

“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A)⇔ Inheritance
Relationship ⇔ Composition / Aggregation

```
Sailors (sid: integer,
sname: string, rating:
integer, age: real)
```

Sid	Sname	Rating	Age
123	Zein..	9	26

```
class Sailors {
        int sid;
        String sname;
        int rating;
        double age;
}

Sailors S1 = new Sailor()
S1.sid = 123;
S1.sname = Zeinab;
S1.rating = 9;
S1.age = 26
```

“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A)⇔ Inheritance
Relationship ⇔ Composition / Aggregation

```
Sailors (sid: integer,
sname: string, rating:
integer, age: real)
```

Sid	Sname	Rating	Age
123	Zein..	9	26
124	Oma..	10	32

```
class Sailors {
        int sid;
        String sname;
        int rating;
        double age;
}

Sailors S2 = new Sailor()
S2.sid = 124;
S2.sname = Omar;
S2.rating = 10;
S2.age = 32
```

“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A)⇔ Inheritance
Relationship ⇔ Composition / Aggregation

```
Sailors (sid: integer,
sname: string, rating:
integer, age: real)
```

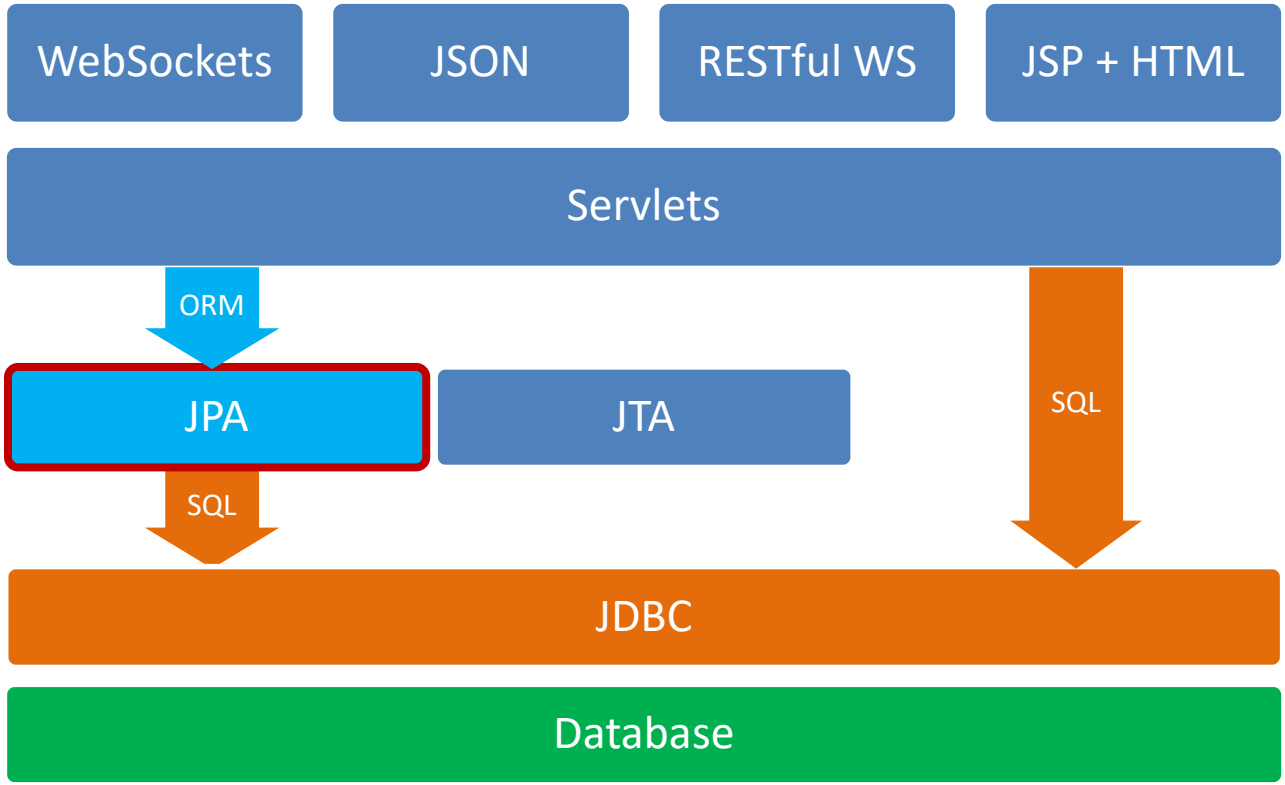
Sid	Sname	Rating	Age
123	Zein..	9	26
124	Oma..	10	32

```
class Sailors {
        int sid;
        String sname;
        int rating;
        double age;
}

Sailors S2 = new Sailor()
S2.sid = 124;
S2.sname = Omar;
S2.rating = 10;
S2.age = 32
```

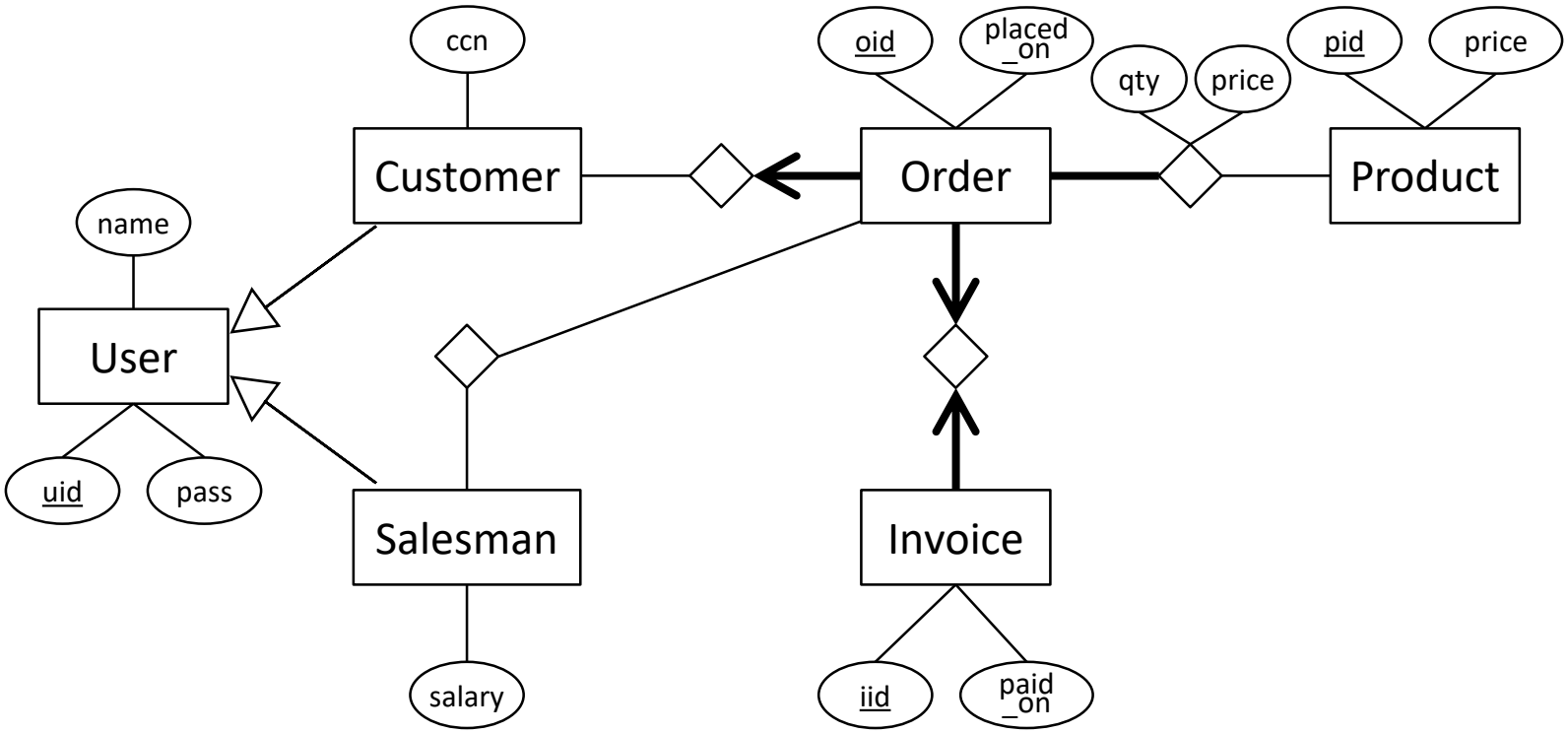
Outline

- Introduction to Object Relational Mapping
- ORM case study (1): JPA (Entity & Inheritance)
- ORM case study (2): Django ORM

Introduction to ORM

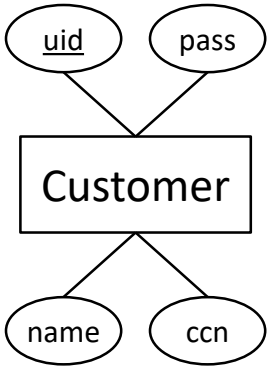


JPA Walkthrough



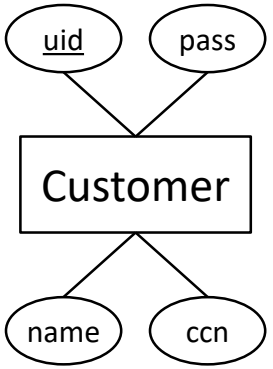
@Entity

```
class Customer
{

    String username;

    String password;

    String full_name;

    String cc_number;

    boolean logged_in;

    Set<Order> orders;
}
```



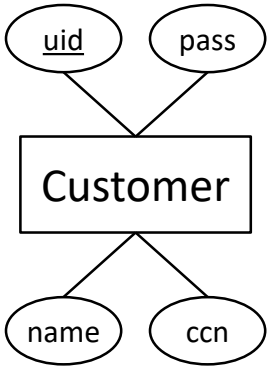
@Entity

```
@Entity

class Customer
{
    @Id

    String username;


    String password;


    String full_name;


    String cc_number;

    @Transient
    boolean logged_in;

    @...
    Set<Order> orders;
}
```

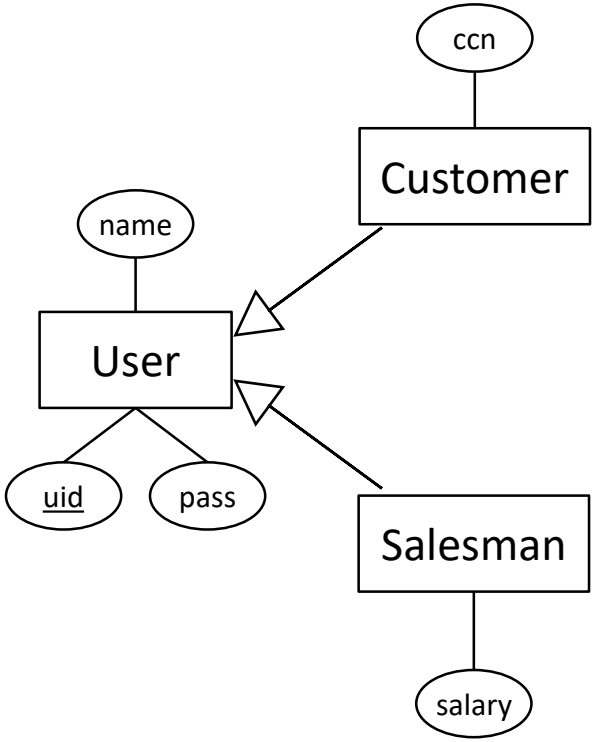


@Entity

```
@Entity
@Table (name = "Customers", schema = "V2")
class Customer
{
    @Id
    @Column (name = "cid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (name = "name", nullable = false)
    String full_name;

    @Column (name = "ccn")
    String cc_number;

    @Transient
    boolean logged_in;

    @...
    Set<Order> orders;
}
```

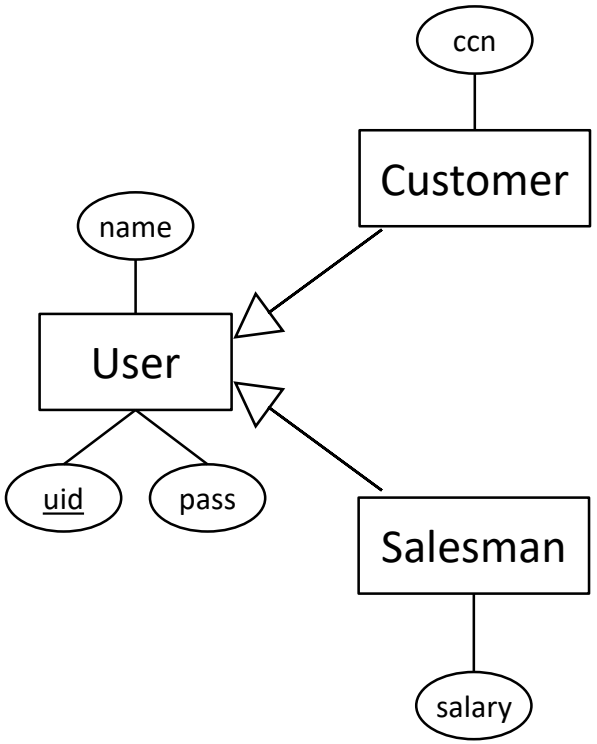


Inheritance



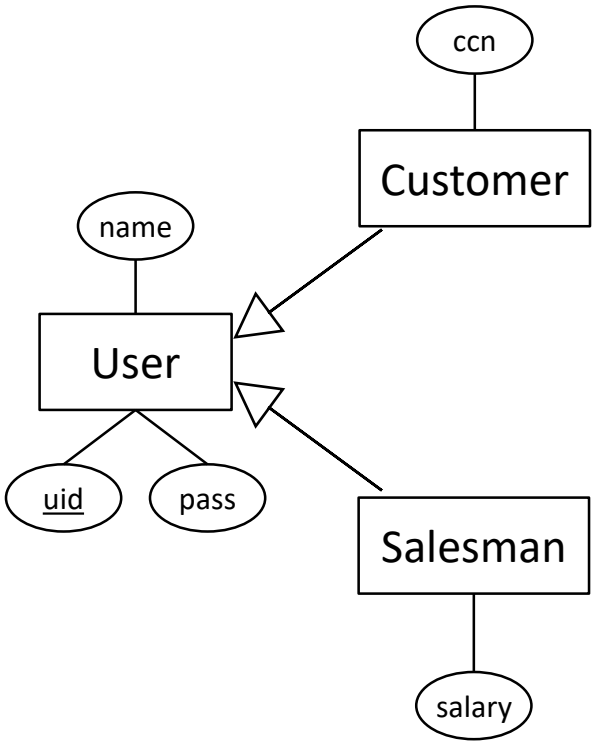
Inheritance

```
@Entity
@...
class User
{
    @Id
    @Column (name = "uid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (nullable = false)
    String full_name;

    @Transient
    boolean logged_in;
}
```



Inheritance

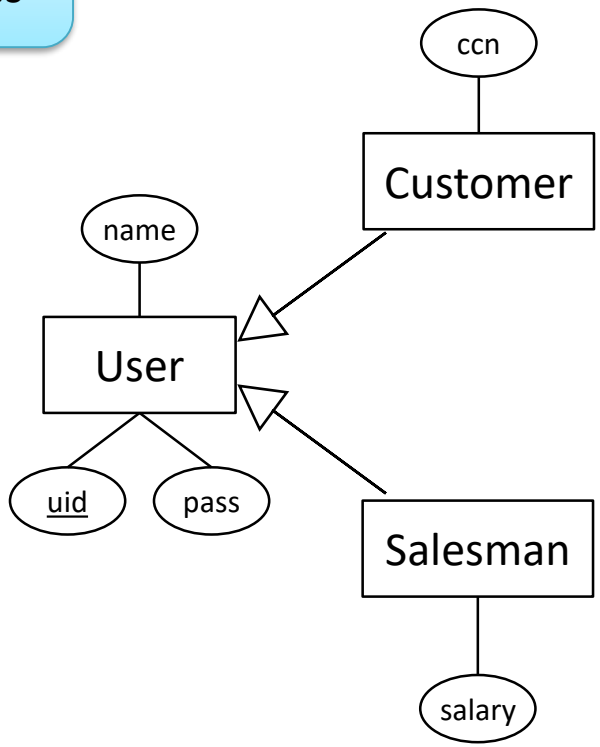
```
@Entity
class Customer extends User
{
    String cc_number;
}

@Entity
class Salesman extends User
{
    int salary;
}
```



Inheritance

```
@Entity
@MappedSuperClass

class User
{
    @Id
    @Column (name = "uid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (nullable = false)
    String full_name;

    @Transient
    boolean logged_in;
}
```

Fields get embedded with subclass entities



Customer

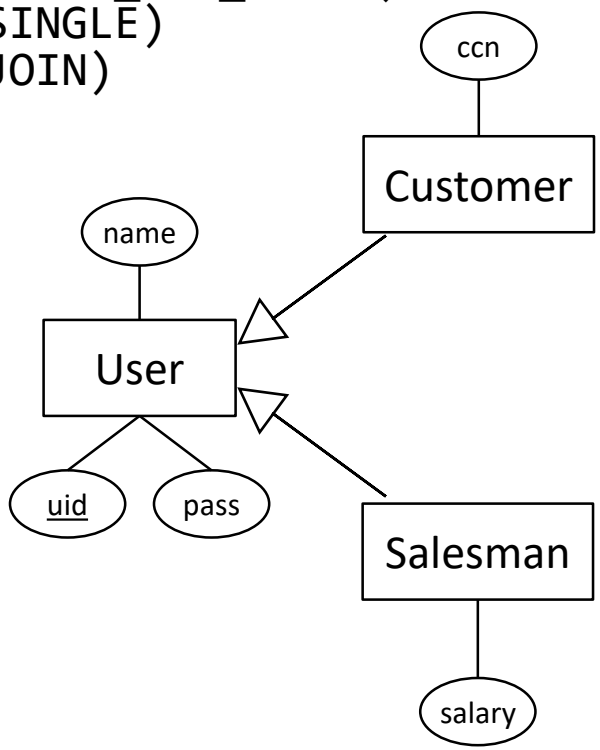
uid	pass	name	ccn
myahmad	1234	Yousuf	123456

Salesman

uid	pass	name	salary
tjab	p@$$	Tamim	

Inheritance

```
@Entity
| @Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
| @Inheritance(strategy = InheritanceType.SINGLE)
| @Inheritance(strategy = InheritanceType.JOIN)
abstract class User
{
    @Id
    @Column (name = "uid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (nullable = false)
    String full_name;

    @Transient
    boolean logged_in;
}
```

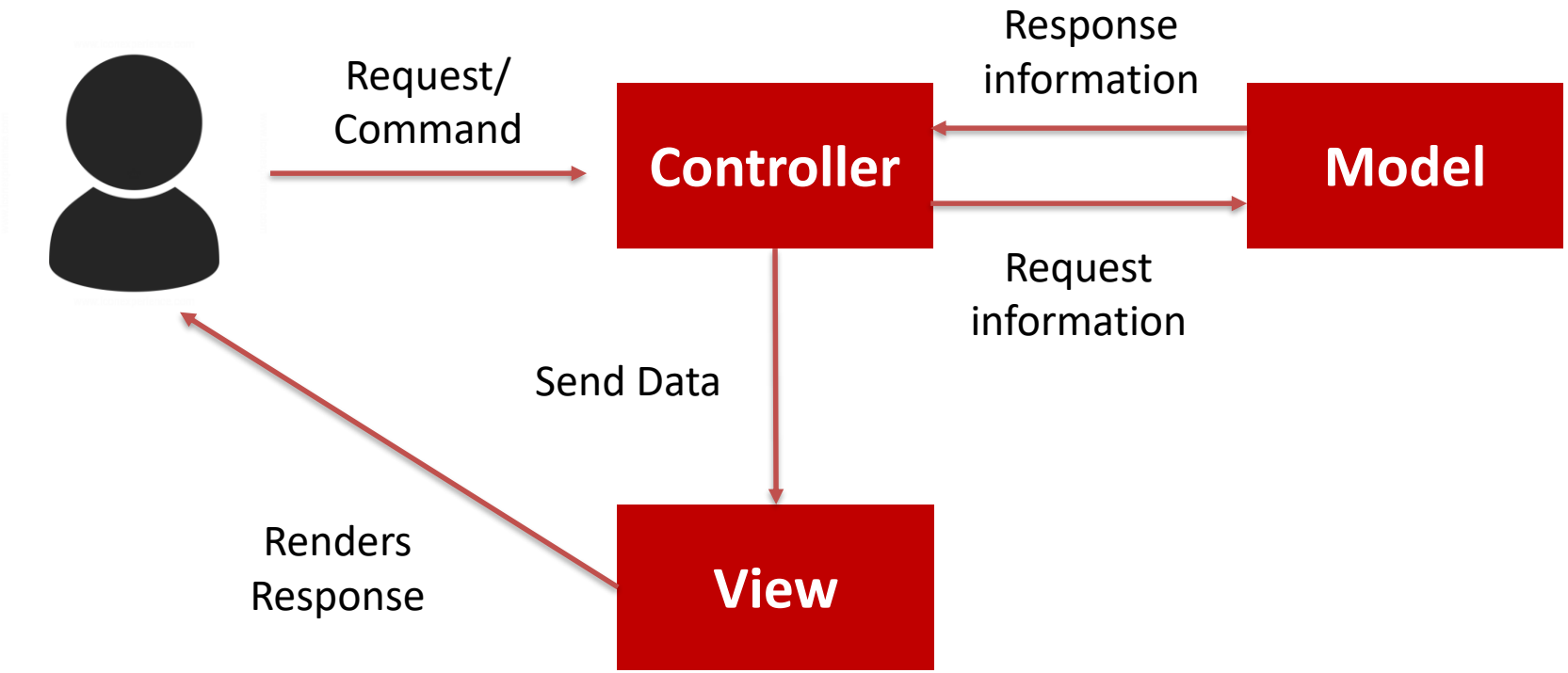


Outline

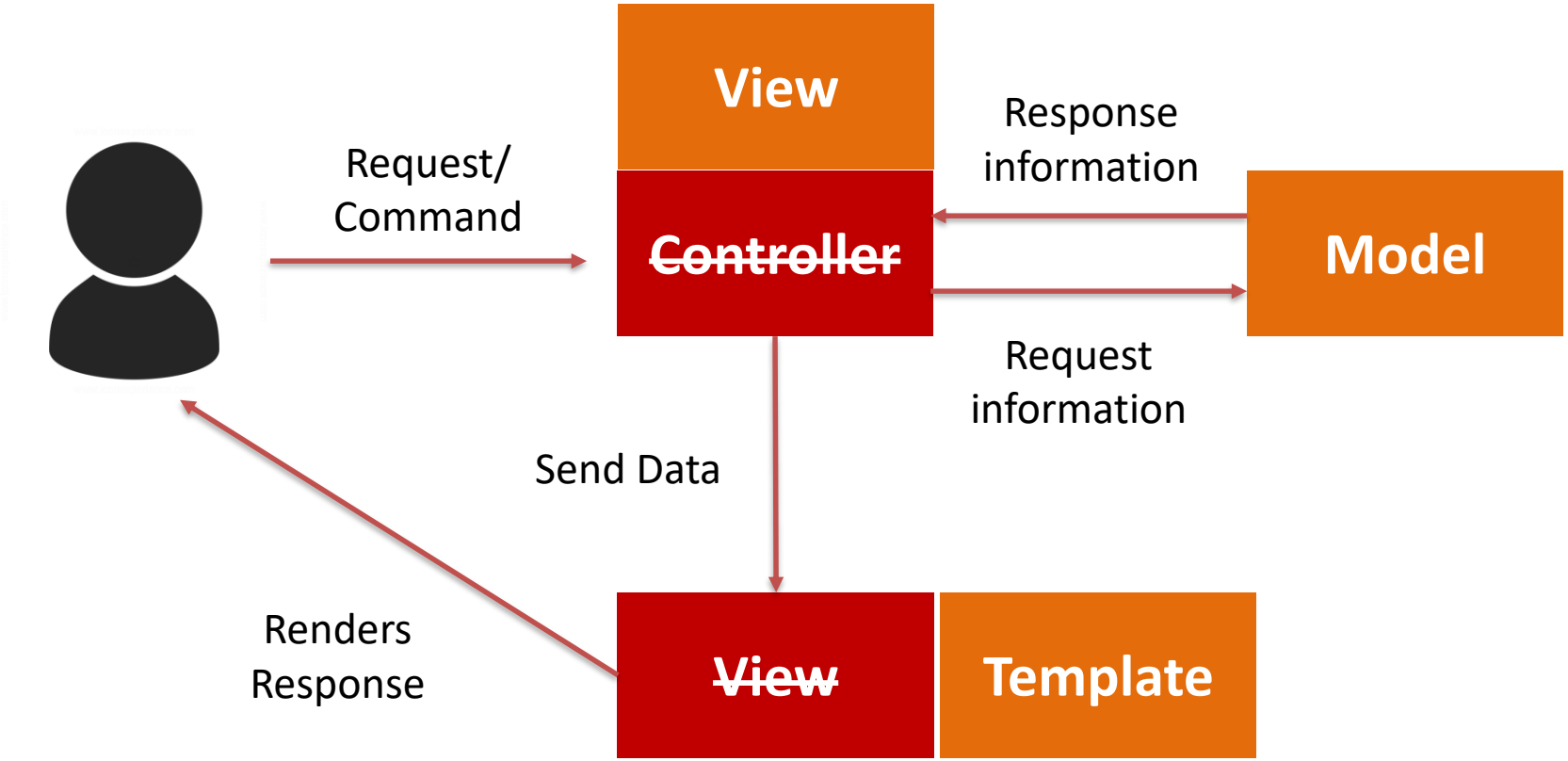
- Introduction to Object Relational Mapping
- ORM case study (1): JPA
- ORM case study (2): Django ORM:

  you will be using Django for your projects …

Model View Controller (MVC)

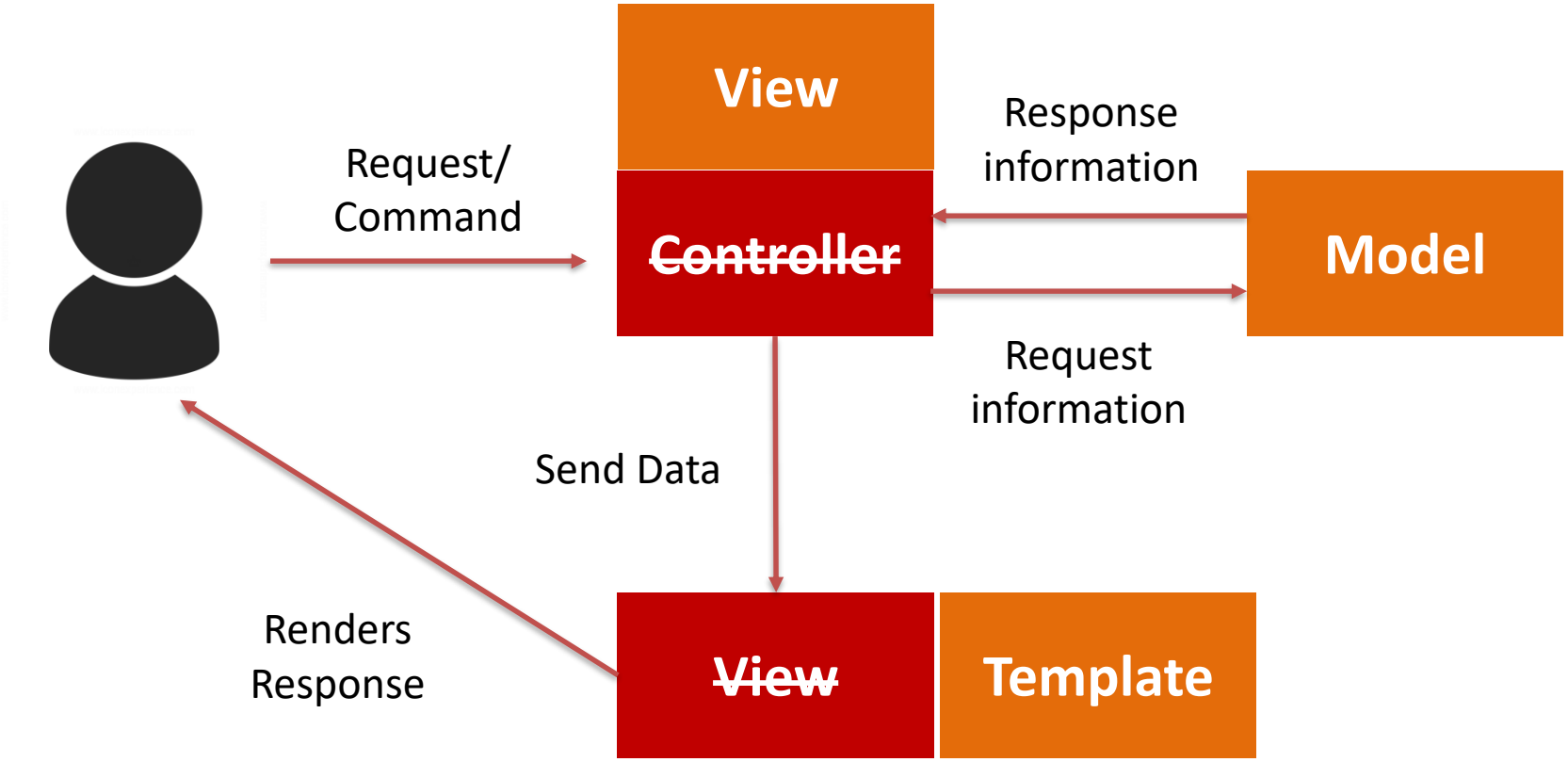


Model View Controller (MVC)

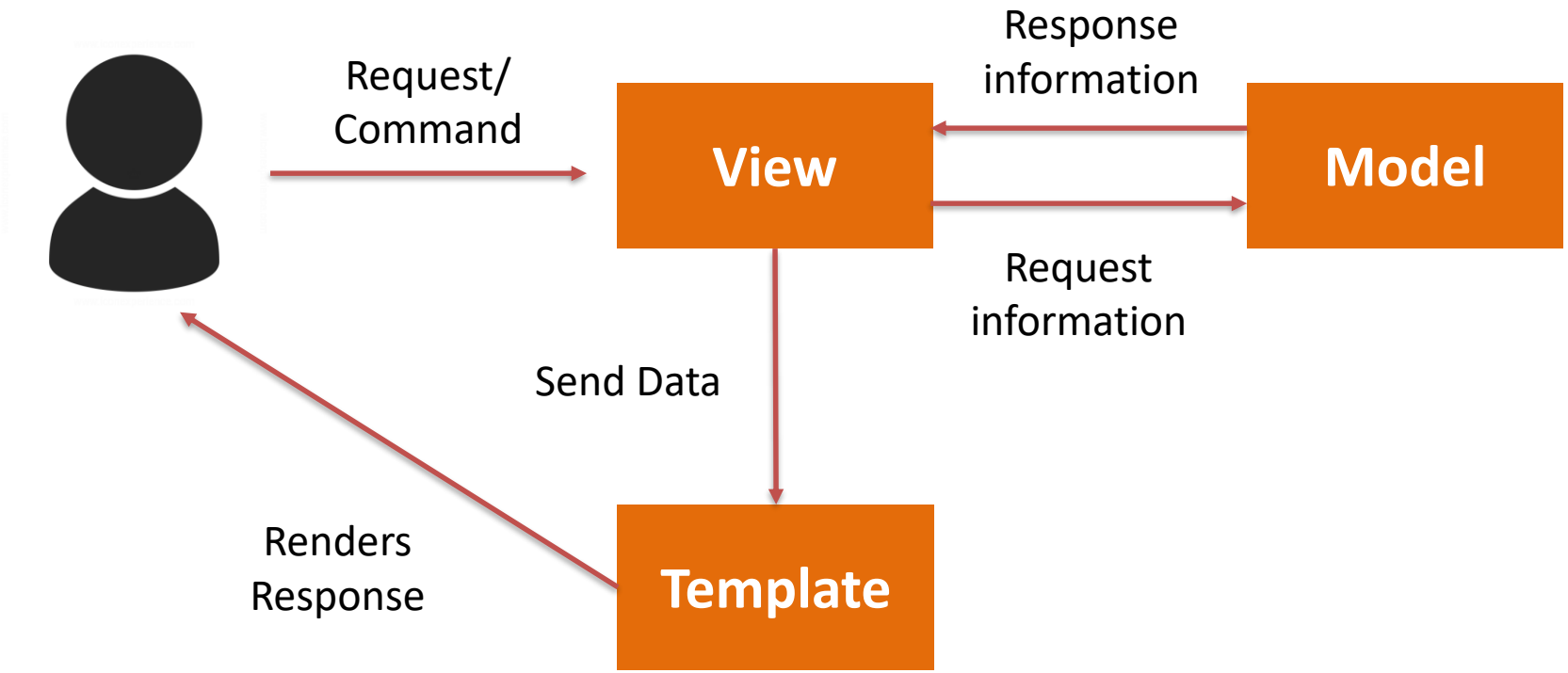


~~Model View Controller (MVC)~~

Model View Template (MVT)



Model View Template (MVT)



Next Recitation…

- Mapping relationships
- Practice with Django