Django
Part 1

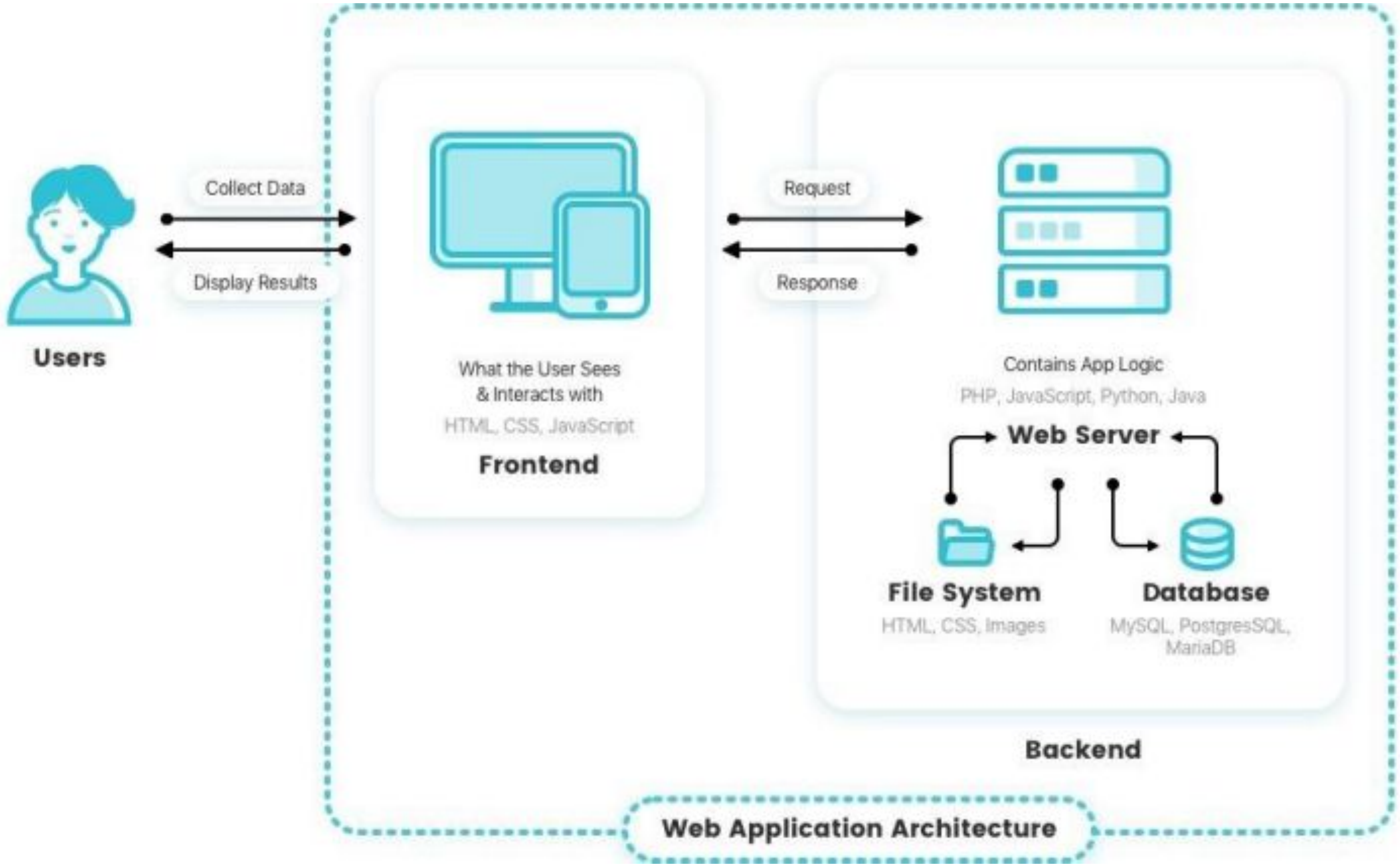
Table of Contents

1. What is a Web Application?
2. How does it work?
3. Stack
4. What is Django
5. Understanding MVC vs MVT Architecture
6. Object Relational Mapping
4. Understanding the Django Workflow

What is Web Application?

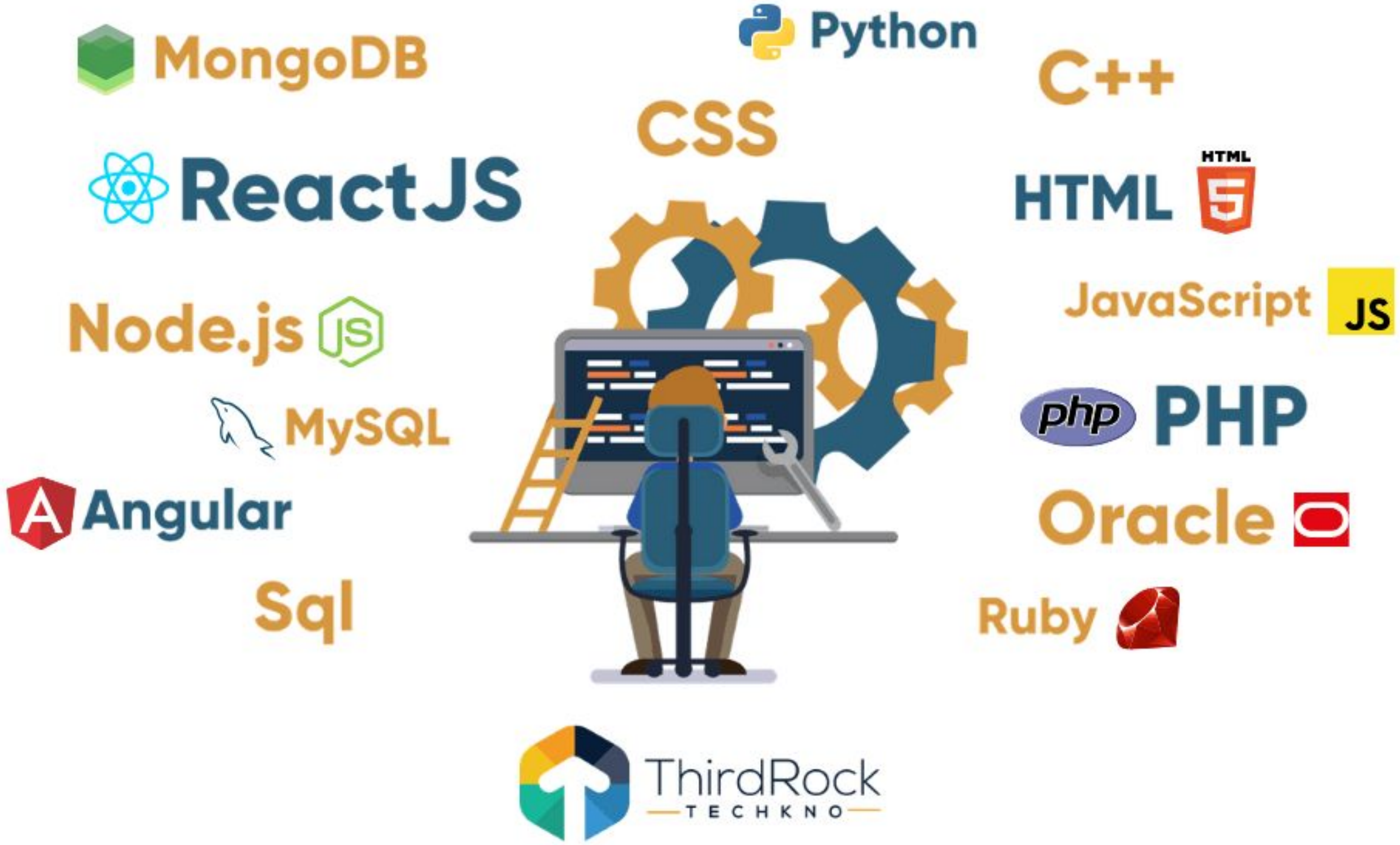
A web application is a software application that runs on a web server, as opposed to being installed on the local computer. Web applications are accessed by the user through a web browser with an active network connection.

How does it work?



Forward School

Stack



Forward School

What is Django?

1. Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.
2. 3 Characteristics of Django:

   1. **Ridiculously fast.**

      Django was designed to help developers take applications from concept to completion as quickly as possible.

   2. **Reassuringly secure.**

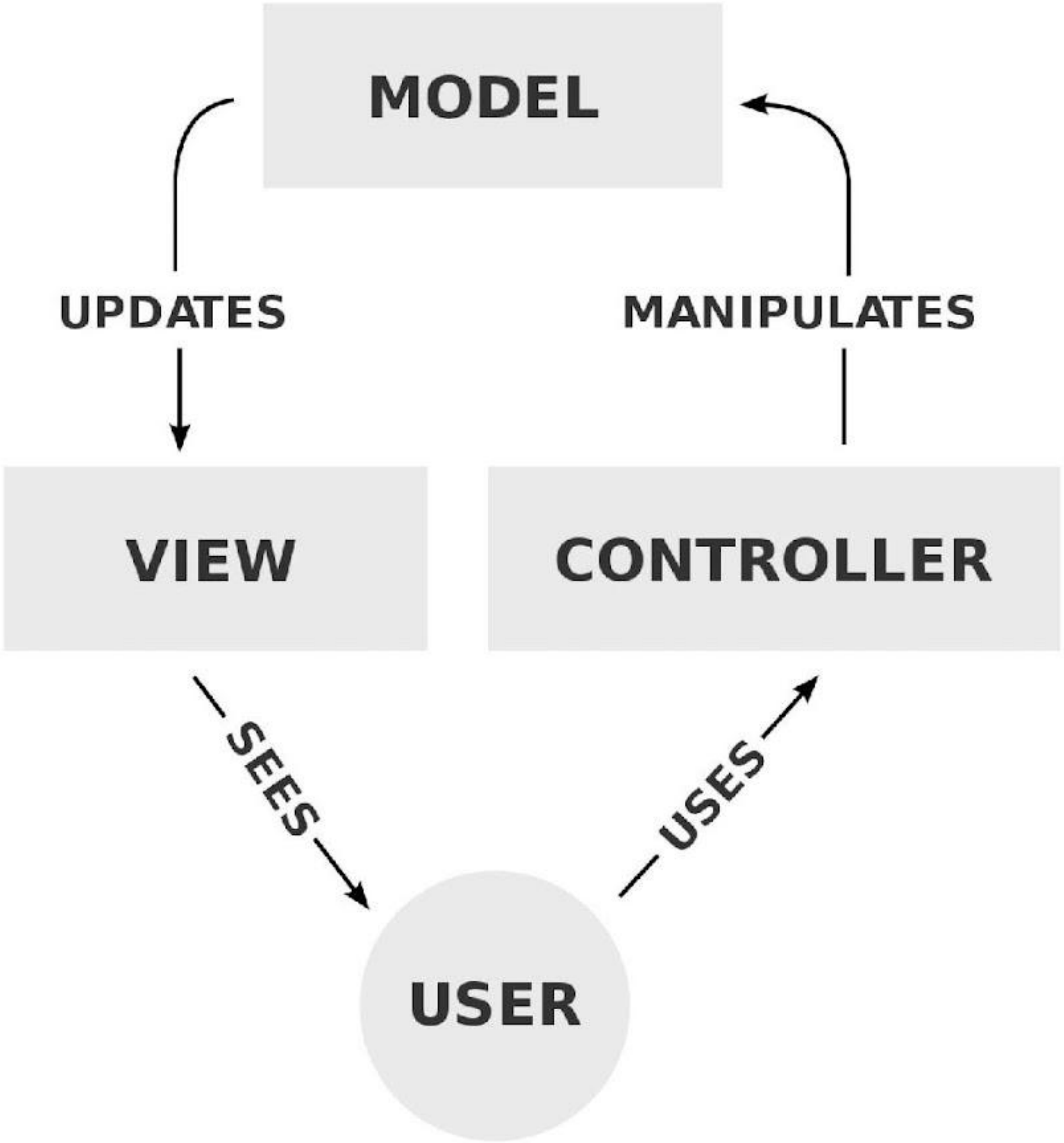
      Django takes security seriously and helps developers avoid many common security mistakes.

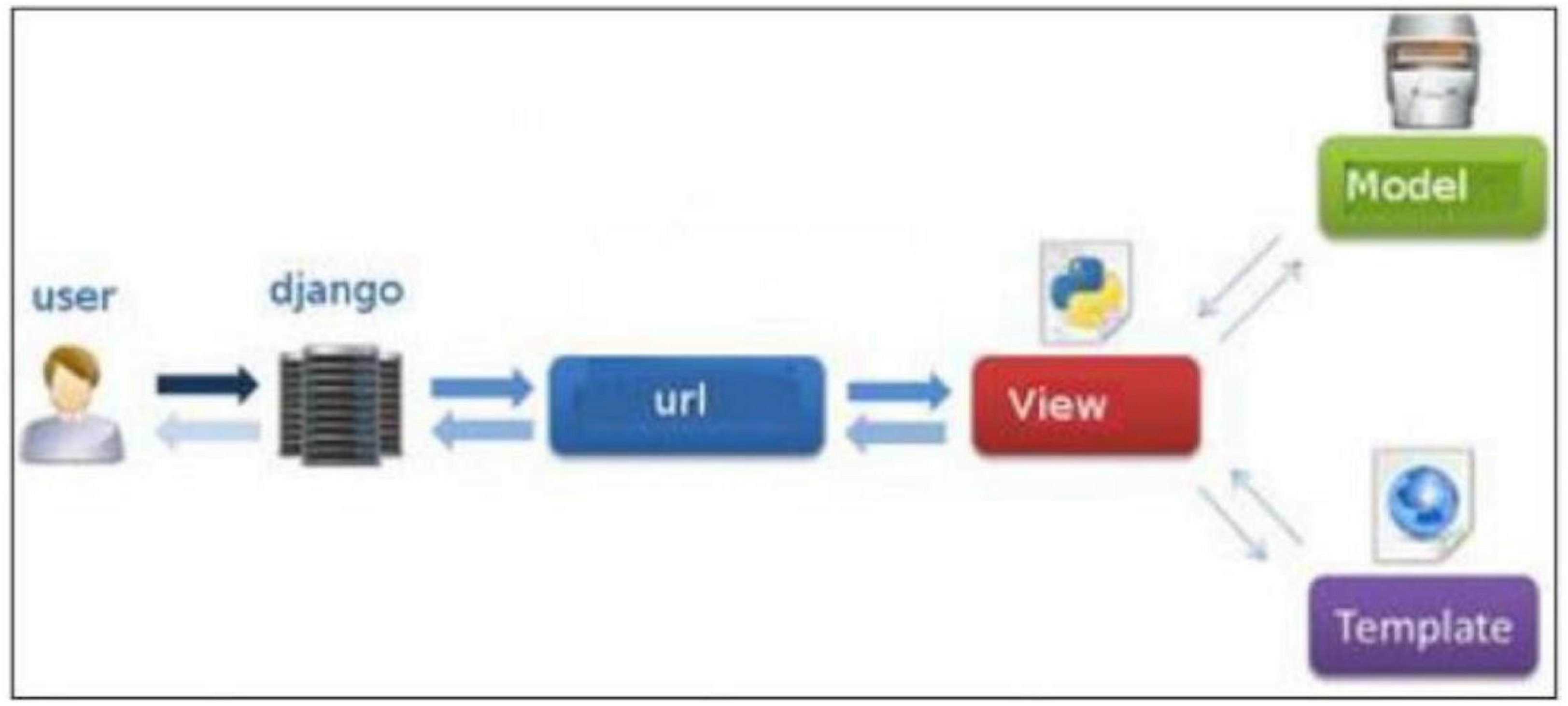
   3. **Exceedingly scalable.**

      Some of the busiest sites on the Web leverage Django’s ability to quickly and flexibly

Uses of Django in real world Applications

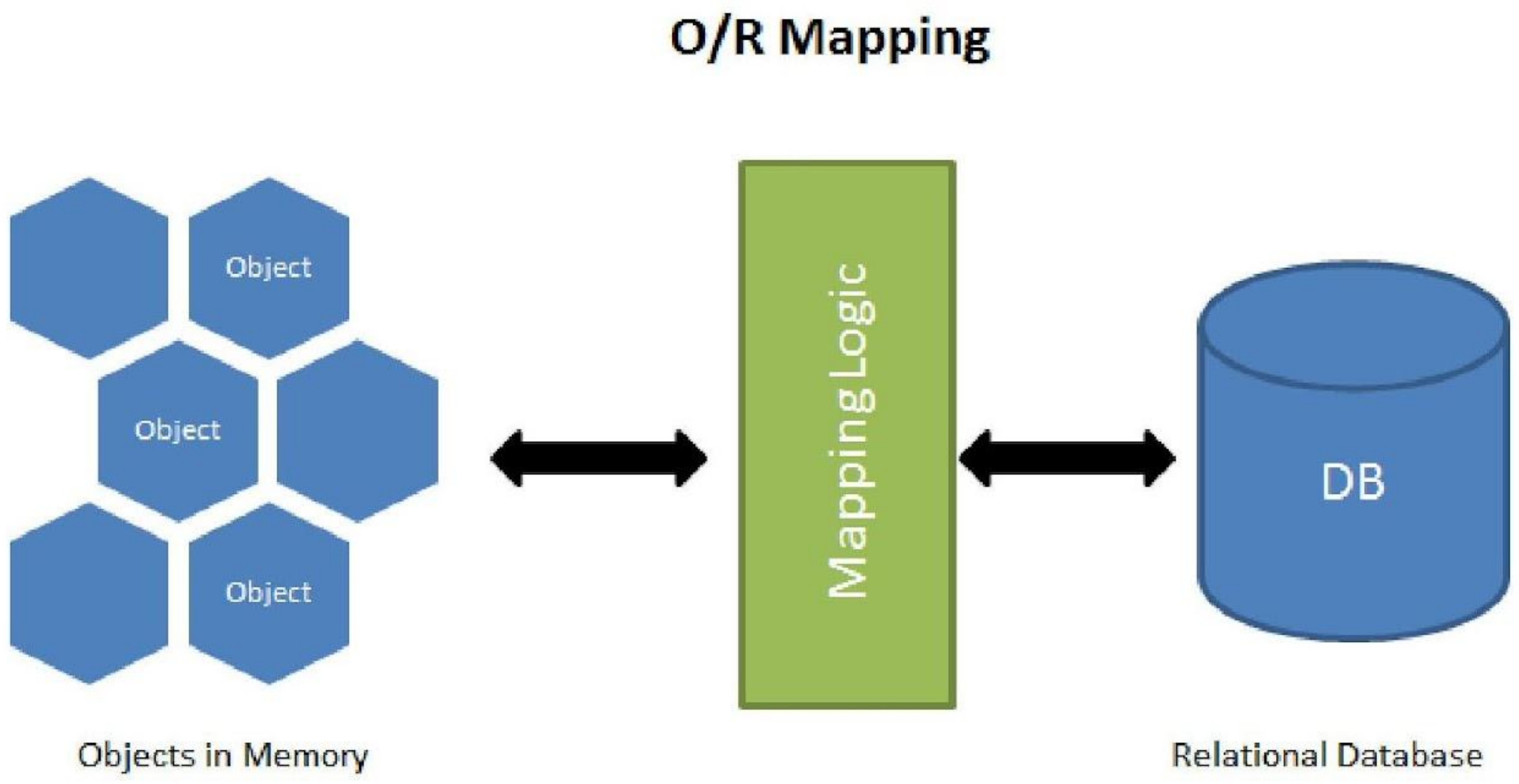
Understanding MVC vs MVT Architecture





Forward School

Object Relational Mapping (ORM)?



Application Layer (Object-Oriented)

- This layer consists of the objects used in your application. These objects are written in an object-oriented programming language like Java, Python, or C#.
- Example: `User` object, `Product` object.

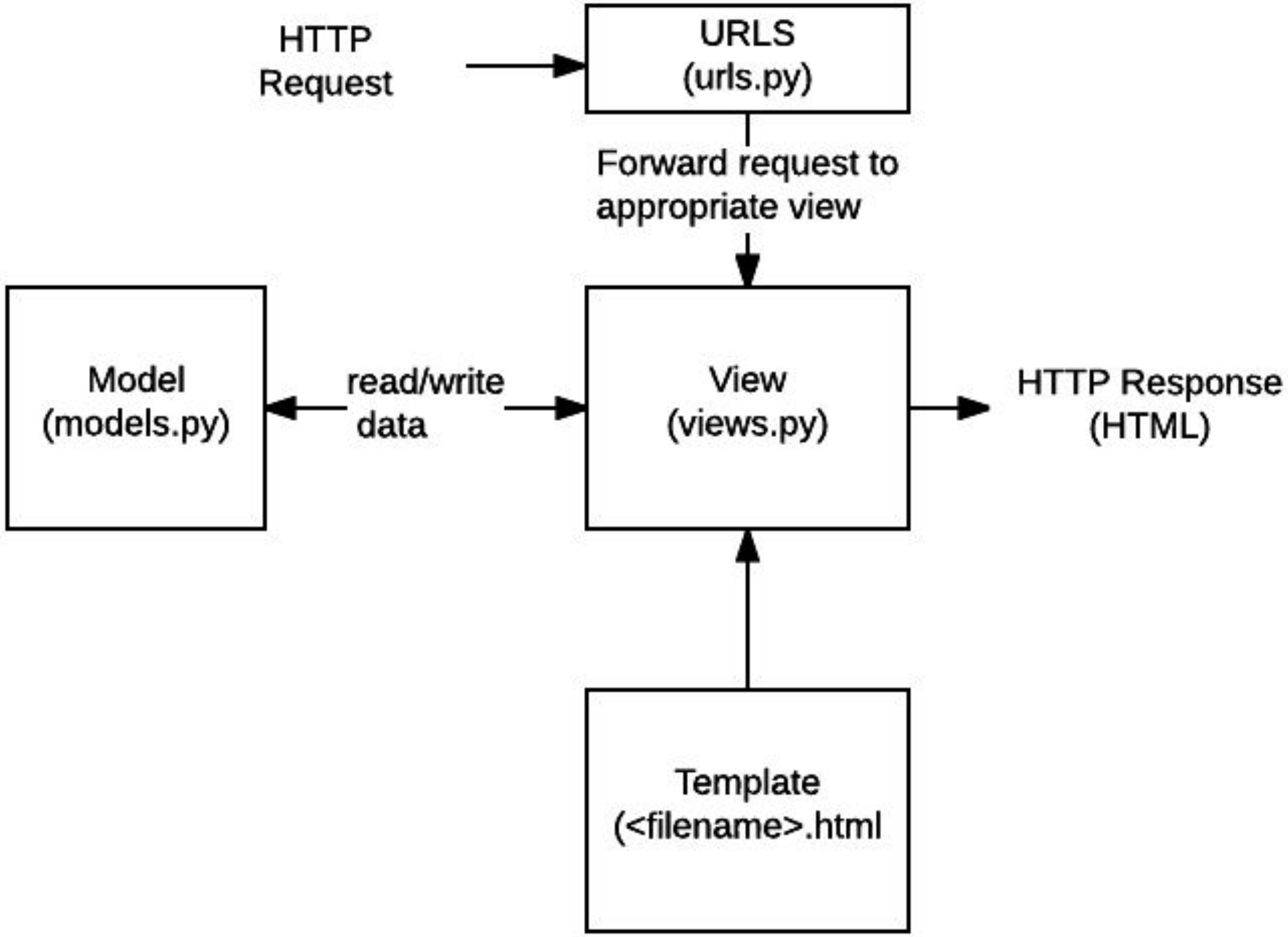
ORM Layer

- This is the middleware or the interface between the application and the database.
- The ORM software translates (maps) the objects in the application layer into tables and columns in the relational database.
- Example: An object `User` with attributes `name` and `email` is mapped to a table `users` with columns `name` and `email`.

Database Layer (Relational)

- This is the relational database where data is stored in tables.
- Example: Tables like `users`, `products`.

Understanding the process django


HTTP
Request
URLS
(urls.py)
Forward request to
appropriate view
Model
(models.py)
read/write
data
View
(views.py)
HTTP Response
(HTML)
Template
(<filename>.html