

# Global Startup

## Tech Meet-up 1: Intro to Django

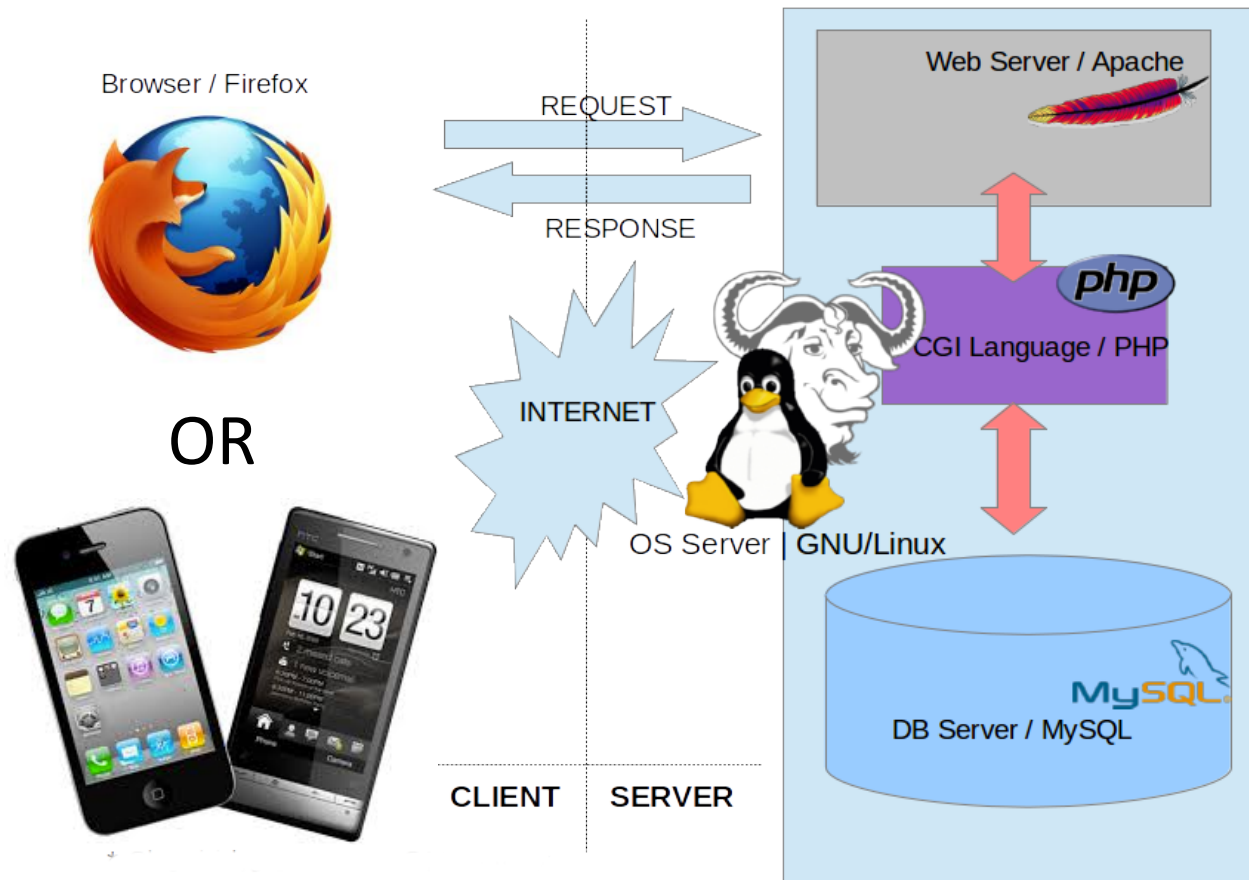


# Today's Meet-up

- The Big Picture
- Django – A Web Application Framework
- Your First Django App
- Models in Django
- Today's Assignment

# The Big Picture

# Once upon a time...



# Nowadays...



# Implementing the Backend

We need to implement the backend:

- A web application framework, like
  - MonoRail, CppCMS, Apache Click, Grails, Spring, Stripes, multiple, Catalyst, CakePHP, Drupal, Symfony, CherryPy, Django, web2py, Ruby on Rails, Compujure
- A host, like
  - Heroku, Google App Engine
- A RESTful API to communicate with frontend

# Web Application Framework

- A framework (code libraries) to help you make web applications or websites
- Supports you with
  - Handling HTTP requests
  - Templates for common HTML layouts
  - URL mapping
  - Database communication
  - Session management
  - Site security

# Django

## Web Application Framework



# Why Django?

# Why Django?

- Fast and easy development of web applications
  - Modular and re-useable. Don't Repeat Yourself (DRY) principle
  - Built-in database management
- Active development and wide community support
- Supported by Google App Engine & Heroku

*Pinterest*



Atlassian



**Bitbucket**

**DISQUS**

**theguardian**

# Django vs. PHP

# PHP SELECT Statement

```
<?php
// This could be supplied by a user, for example
$firstname = 'fred';
$lastname = 'fox';

// Formulate Query
// This is the best way to perform an SQL query
// For more examples, see mysql_real_escape_string()
$query = sprintf("SELECT firstname, lastname, address, age FROM friends
WHERE firstname='%s' AND lastname='%s'",
mysql_real_escape_string($firstname),
mysql_real_escape_string($lastname));

// Perform Query
$result = mysql_query($query);

// Check result
// This shows the actual query sent to MySQL, and the error. Useful for debugging.
if (!$result) {
    $message = 'Invalid query: ' . mysql_error() . "\n";
    $message .= 'Whole query: ' . $query;
    die($message);
}

// Use result
// Attempting to print $result won't allow access to information in the resource
// One of the mysql result functions must be used
// See also mysql_result(), mysql_fetch_array(), mysql_fetch_row(), etc.
while ($row = mysql_fetch_assoc($result)) {
    echo $row['firstname'];
    echo $row['lastname'];
    echo $row['address'];
    echo $row['age'];
}

// Free the resources associated with the result set
// This is done automatically at the end of the script
mysql_free_result($result);
?>
```

# Django SELECT Statement

```
# Retrieve all Kaylas
kaylas = User.objects.filter(firstname = 'Kayla')

# Print out firstname, lastname, and kayaks rolled
for k in kaylas:
    print k.firstname, k.lastname, k.kayaks_rolled_count
```

Focus on building your  
product!



VS



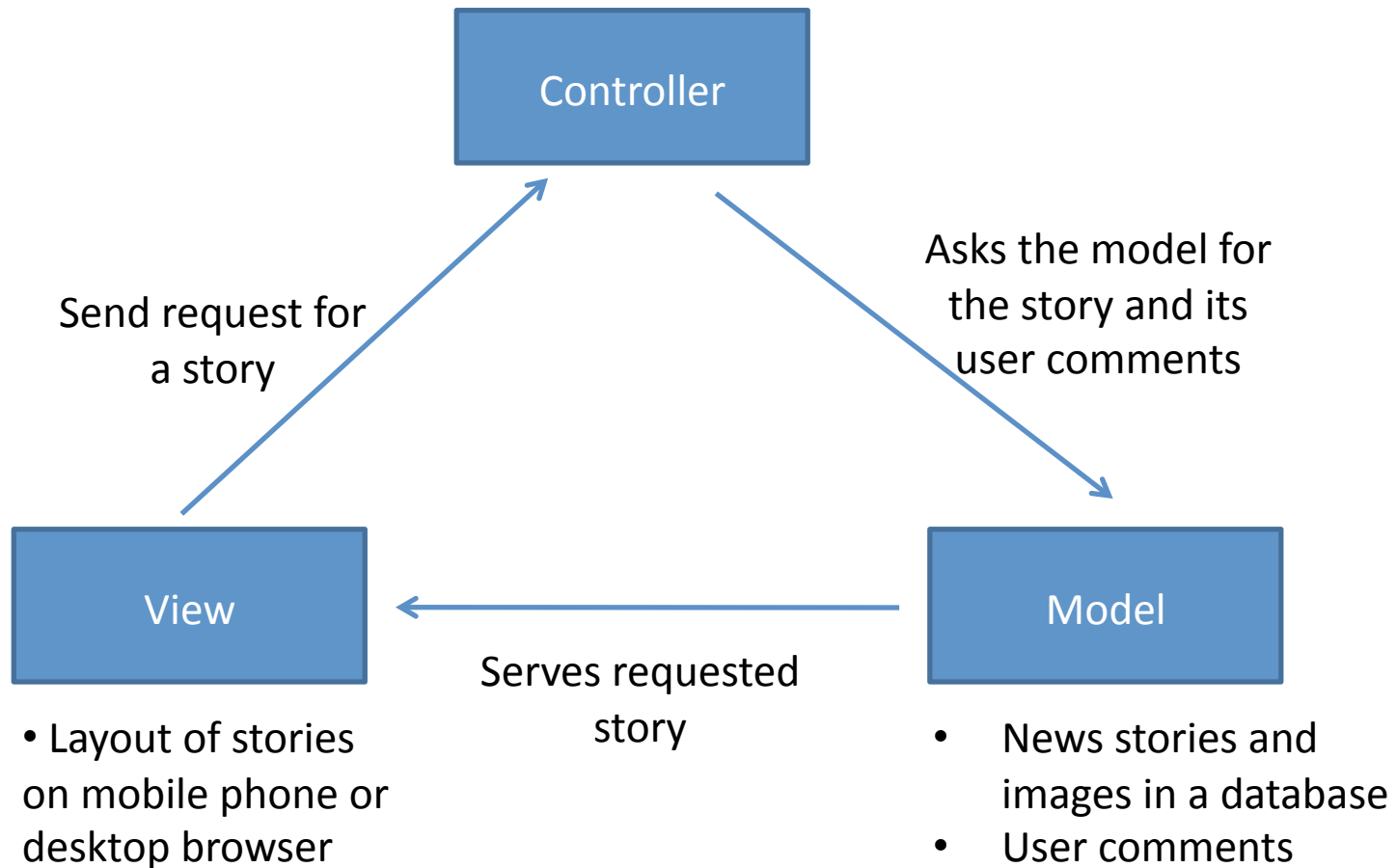


# Django Architecture

# Model-View-Controller (MVC)

- A pattern for organizing code often seen in web app frameworks
- Main idea:
  1. Separate storage and manipulation of data (model) from presentation of data (view)
  2. Controller communicates between model and view
- Advantages
  - Develop and test model and view independently
  - Easier for others to understand (modularity)

# Model-View-Controller (MVC) (news site example)

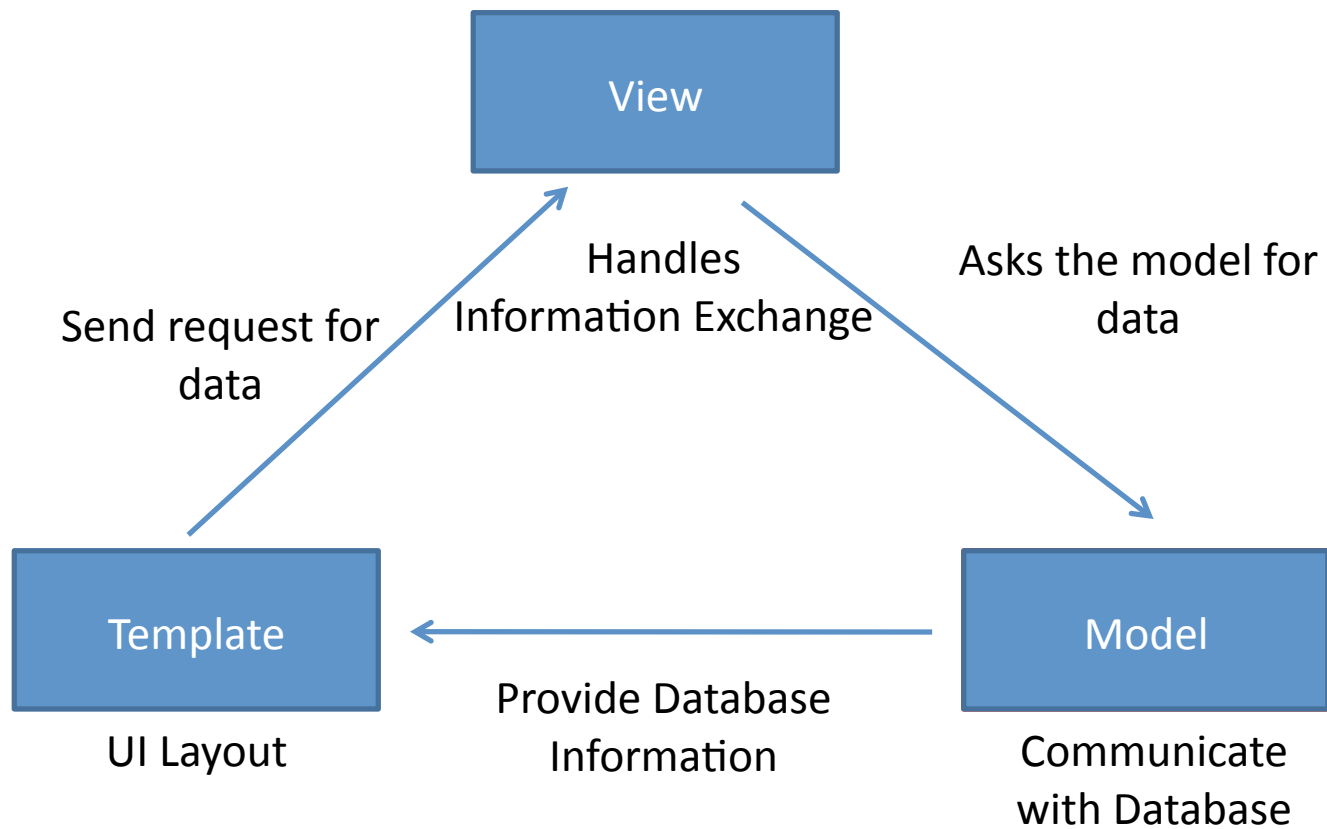


# Model-Template-View

In Django: Model-Template-View  
(similar to MVC pattern)

- Model
  - describes database information
- Template
  - decides how to present information
- View
  - manages what information to output based on request

# Model-Template-View

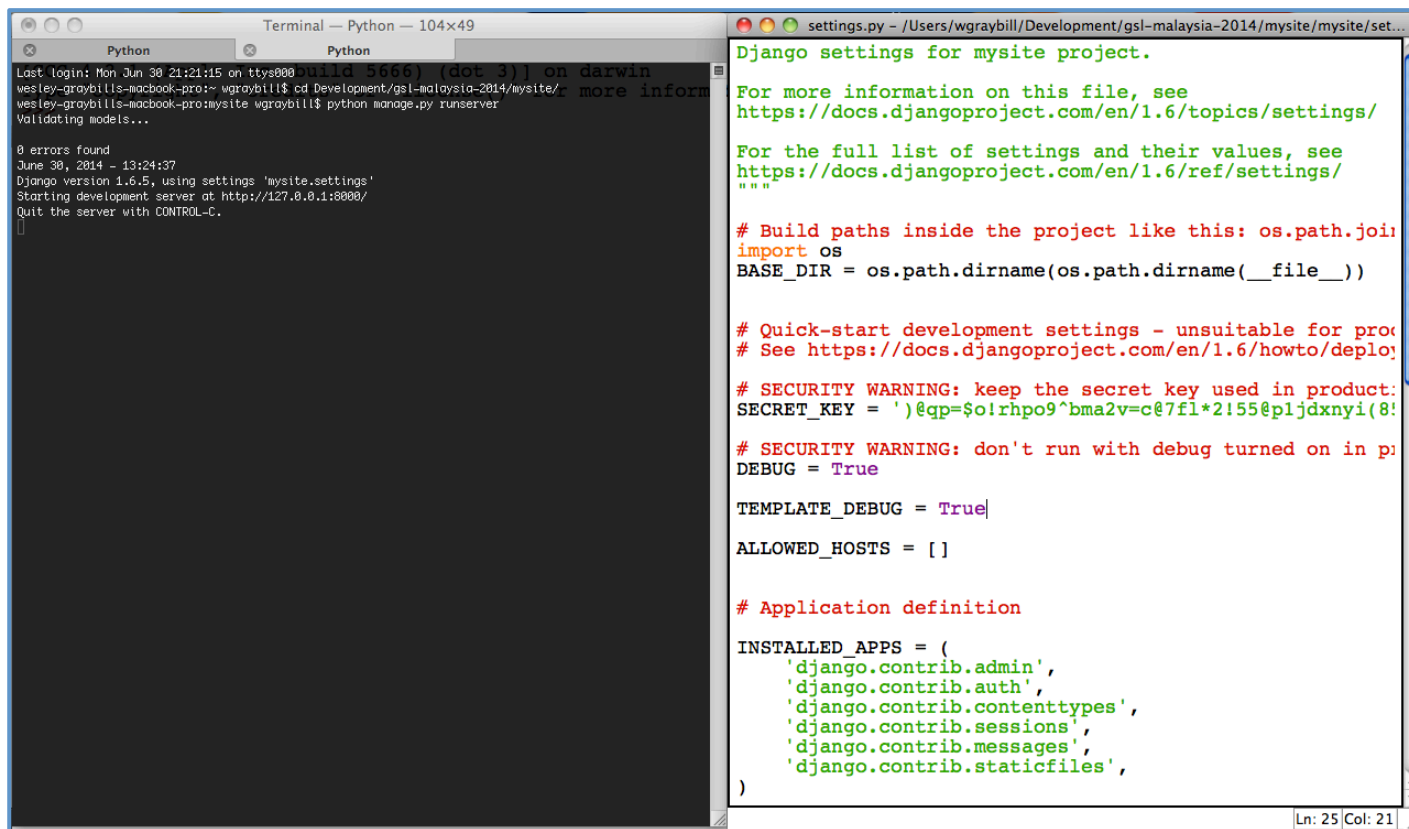


# Your First Django App

# Programming Interface

Terminal / Command Prompt

Python IDLE or Favorite editor



The image shows two side-by-side windows. The left window is a terminal titled 'Terminal - Python - 104x49'. It shows the output of running 'python manage.py runserver', including the Django version (1.6.5) and the server URL (http://127.0.0.1:8000/). The right window is a text editor titled 'settings.py - /Users/wgraybill/Development/gsl-malaysia-2014/mysite/mysite/set...'. It displays the Django settings for the 'mysite' project, including paths, security warnings, and application definitions.

```
Terminal - Python - 104x49
Python
Python
Last login: Mon Jun 30 21:21:15 on ttys000
wlesley-graybill@wlesley-graybill-macbook-pro: ~
wlesley-graybill@wlesley-graybill-macbook-pro: ~$ cd Development/gsl-malaysia-2014/mysite/
wlesley-graybill@wlesley-graybill-macbook-pro:mysite$ python manage.py runserver
Validating models...

0 errors found
June 30, 2014 - 13:24:37
Django version 1.6.5, using settings 'mysite.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

settings.py - /Users/wgraybill/Development/gsl-malaysia-2014/mysite/mysite/set...
Django settings for mysite project.

For more information on this file, see
https://docs.djangoproject.com/en/1.6/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/1.6/ref/settings/
"""

# Build paths inside the project like this: os.path.join
import os
BASE_DIR = os.path.dirname(os.path.dirname(__file__))

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/1.6/howto/deployment

# SECURITY WARNING: keep the secret key used in production
SECRET_KEY = ')@qp=$o!rhpo9^bma2v=c@7fl*2!55@pljdxnyi(8!

# SECURITY WARNING: don't run with debug turned on in production
DEBUG = True

TEMPLATE_DEBUG = True

ALLOWED_HOSTS = []

# Application definition


INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
)

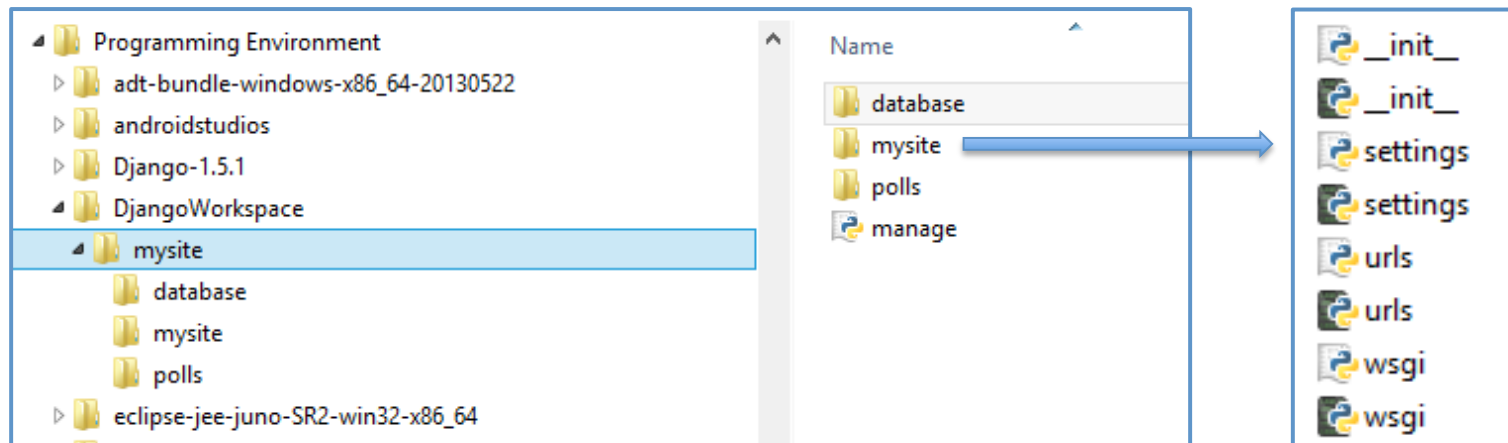
Ln: 25 Col: 21
```

# New Project

## Project Structure:

Whole project in one folder (mysite)

- MySite Python package 
  - Applications (polls)
  - (Database)
  - manage.py
    - Use for interaction with your project
- MySite Python Package
    - `__init__.py`
    - `settings.py`
    - `urls.py`
    - `wsgi.py`





# First Django Setup

After a couple commands in terminal...

**it worked!**

Congratulations on your first Django-powered page.

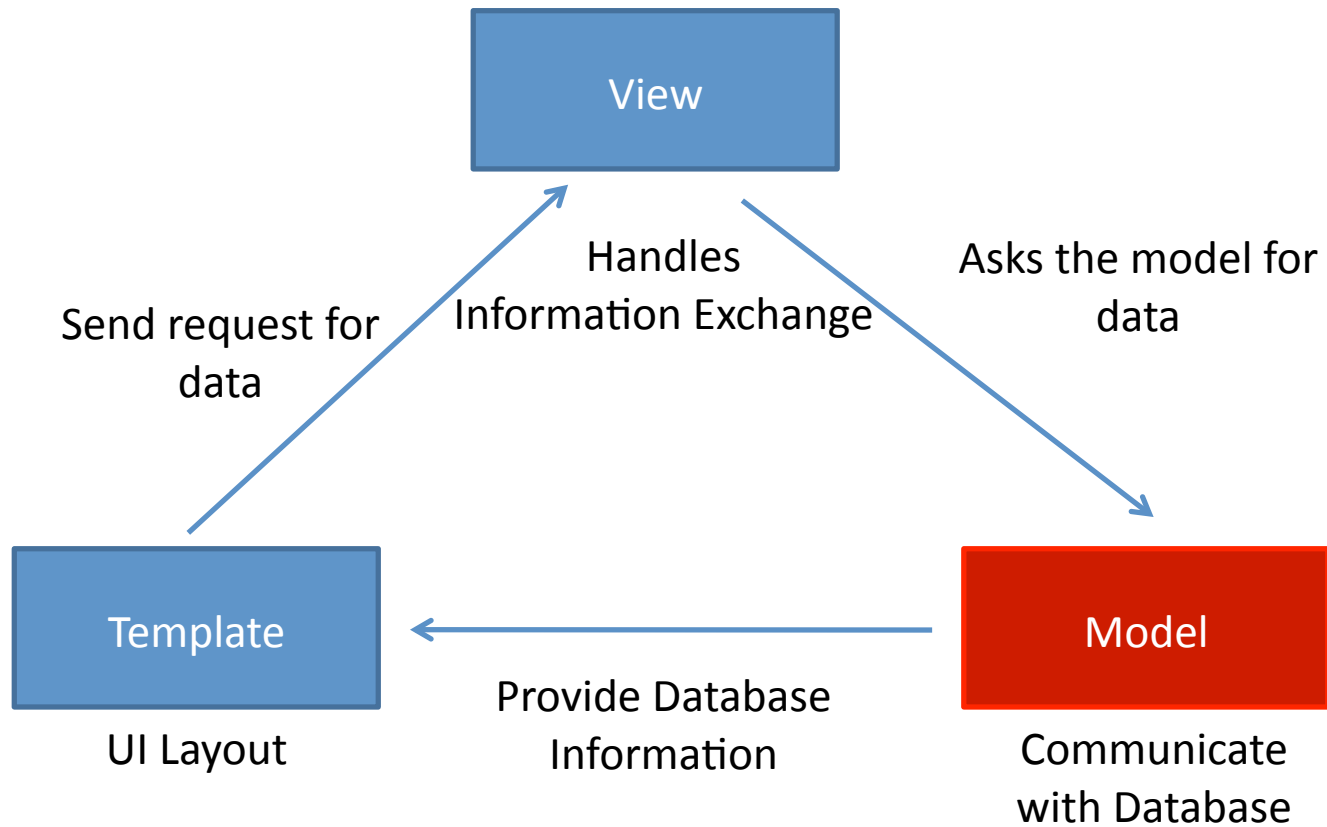
Of course, you haven't actually done any work yet. Here's what to do next:

- If you plan to use a database, edit the `DATABASES` setting in `mysite/settings.py`.
- Start your first app by running `python manage.py startapp [appname]`.

You're seeing this message because you have `DEBUG = True` in your Django settings file and you haven't configured any URLs. Get to work!

# Models

# Models



# What is a Model

- Python class describing data in your application
  - Subclass of `models.Model`
- Assigns attributes to each data field
- Avoid direct work with the database
  - No need to handle database connections, timeouts, etc. – Let Django do it for you!
  - Provides Schema for database

# Django Model Syntax

Import statements

SubClass of  
models.Model Class

Define fields

`__unicode__`  
corresponds to  
python `__str__`

Can define more  
functions

```
from django.db import models
import datetime
from django.utils import timezone

# Create your models here.

class Poll(models.Model):
    question = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')

    def __unicode__(self):
        return self.question

    def was_published_recently(self):
        return self.pub_date >= timezone.now() - datetime.timedelta(days=1)
```

# Django Fields

We can define fields directly in our model class

- No need to define manually in database

Example: create two fields in our Poll class

```
class Poll(models.Model):  
    question = models.CharField(max_length=200)  
    pub_date = models.DateTimeField('date published')
```

Define Type of Field

- E.g. `models.CharField`

Define arguments of field

- E.g. `max_length=200`

Django will  
automatically create  
fields in database

# Important Django Field Types

- BooleanField
  - Checkbox
- CharField(max\_length)
  - Single-line textbox
- DateField
  - Javascript calendar
- DateTimeField
  - Javascript calendar, time picker
- DecimalField(max\_digits, decimal\_places)
  - Decimal numbers
- EmailField
  - Charfield that validates email address
- FileField
  - File upload, stores path in database
- FloatField
  - Floating point numbers
- IntegerField
  - Integer textbox
- PositiveIntegerField
  - Integer textbos for positive integers
- TextField
  - Multi-line textbox

# Rules of Django Models

- When you update a model, ALWAYS RUN **python manage.py syncdb**
- All classes extend **models.Model**
- Models only live in **Apps**
- Django doesn't save objects until you call **save()** method

```
>>>a1 = Album(...)
```

```
# a1 is not saved to the database yet!
```

```
>>>a1.save()
```

```
# Now it is.
```



# Today's Assignment

# Today's Assignment

## Get started with Django!

- If you haven't already, install Django 1.6
- Create your first Django app:

[Writing your first Django App – Part 1](#)

## Helpful Documentation:

- Models section of [Django docs](#)