Accelerating Information Technology Innovation

http://aiti.mit.edu

Ghana Summer 2013
Django Lecture 1– Web Frameworks

MISTI

Agenda

- Linux
- Django
- Python

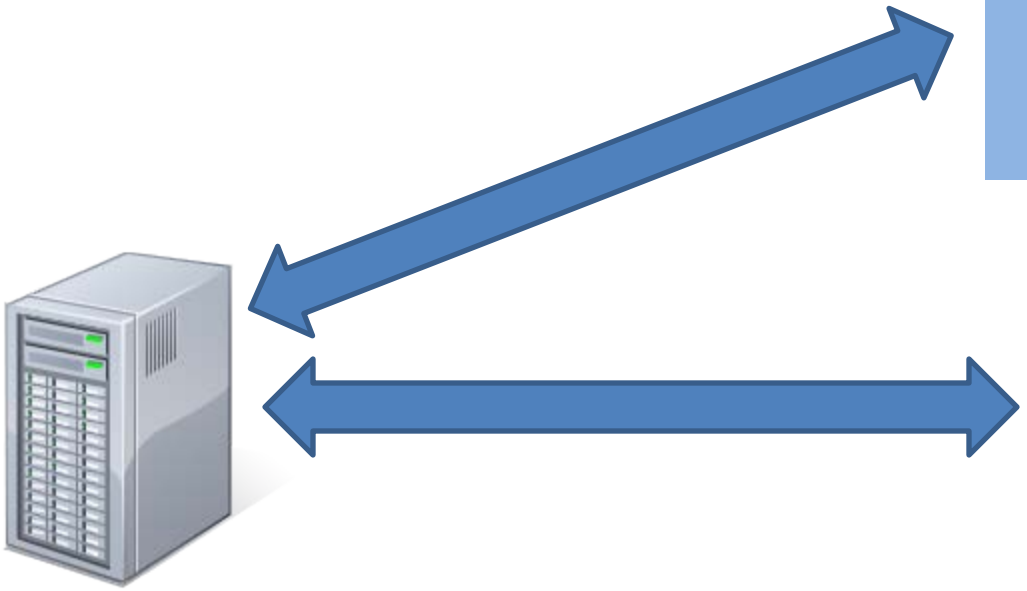
Linux

- Free and open source operating system
- Powerful Terminal
  - Navigate file system
  - Run programs
  - Download things from the internet

The Big Picture

Web browser

Your HTML web pages



Google App Engine

Your Django app (Python code)

Android OS

Your Android app

Simplest Web Page

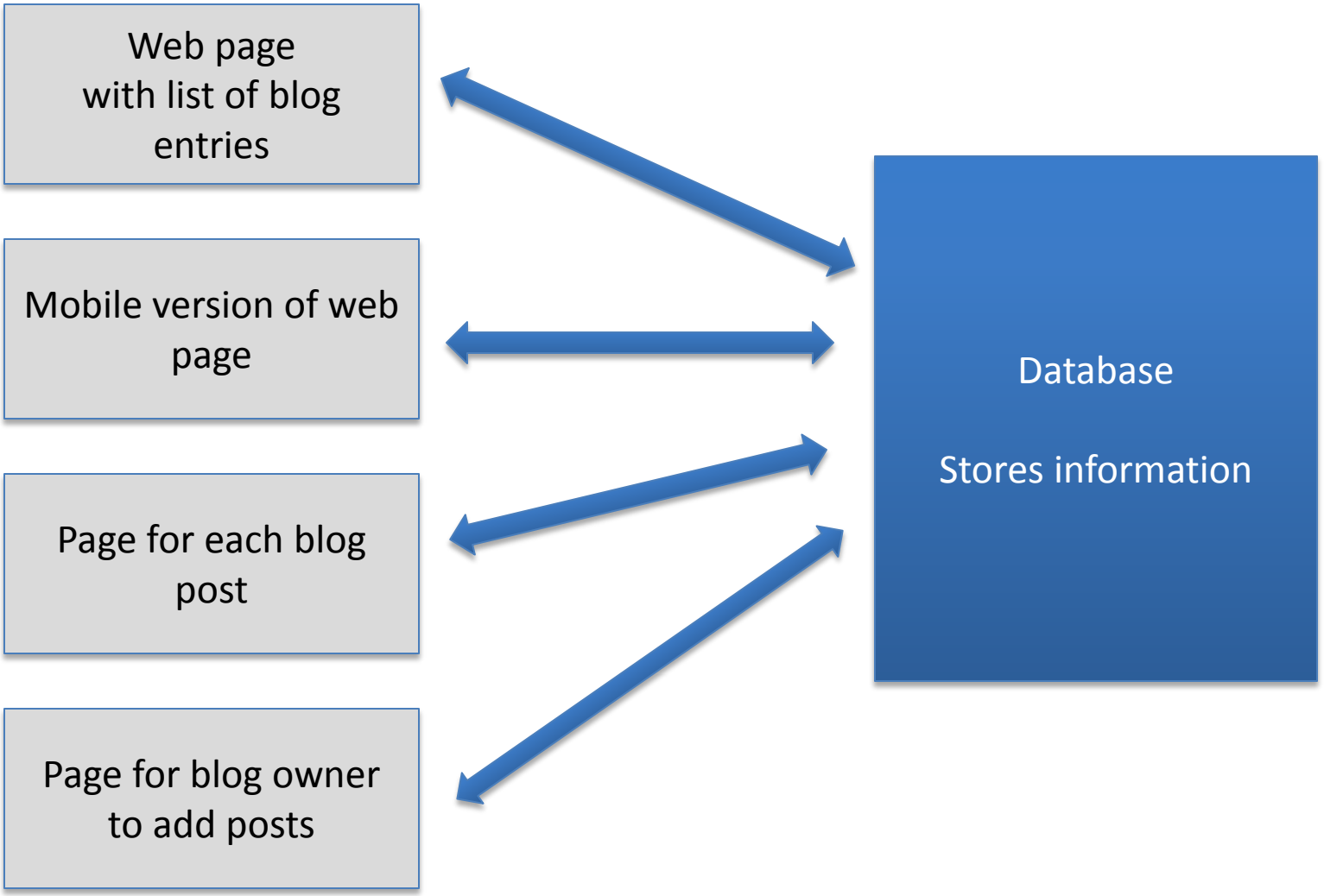
Blog Application

Blog web page

Database

Stores information

Blog Application



Web Application Framework

- A framework (a.k.a. code libraries) that provides functionality for common components in a website, web app, or web service.
- Eases coding for
  - Working with forms
  - Handling HTTP requests
  - Templates for common HTML layouts
  - URL mapping
  - Database communication
  - Session management
  - Site security
- Allows you to focus on design and functionality rather than small details.

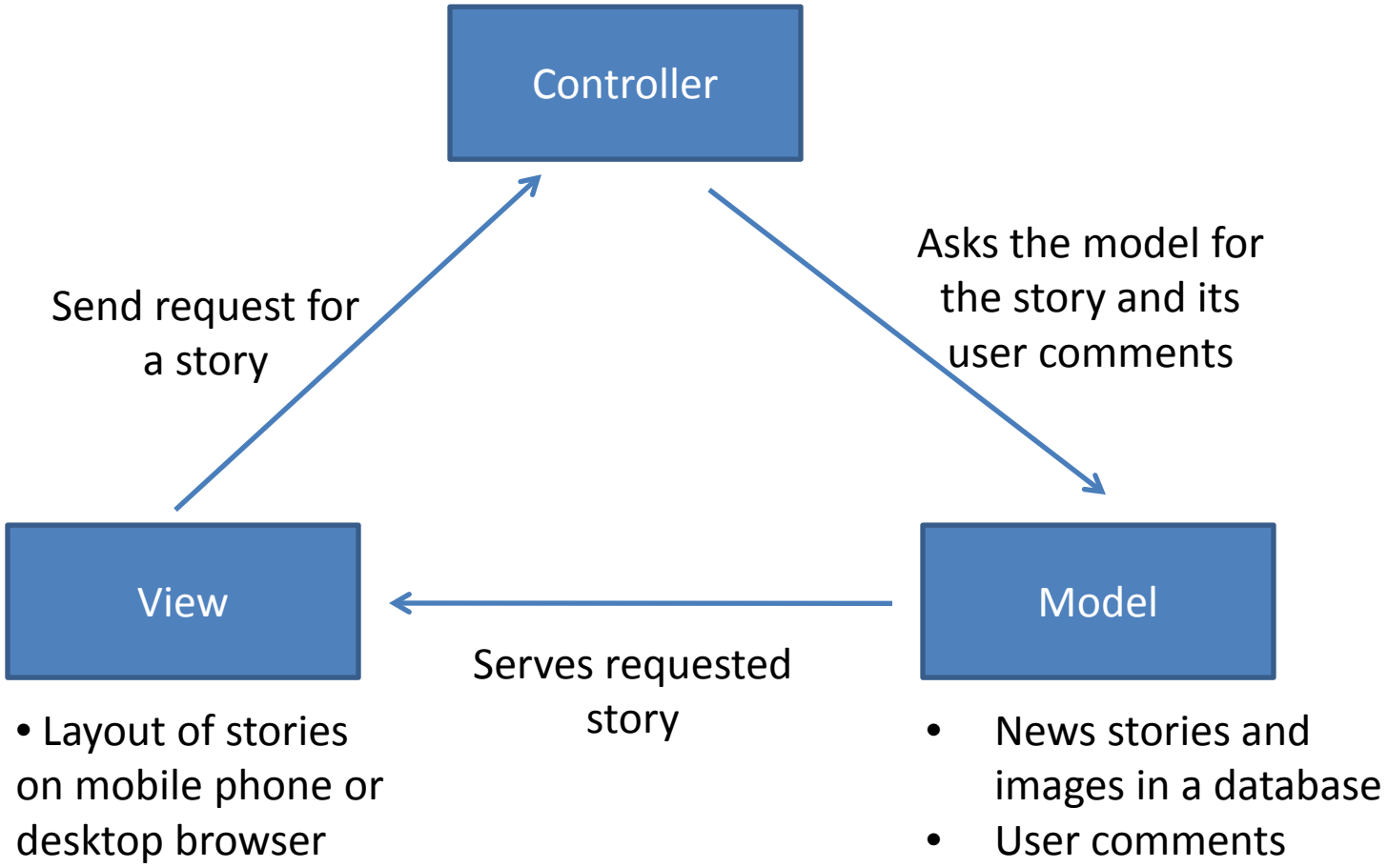
Django

- Web framework that handles many things for you
- You don’t have to understand how it works behind the scenes

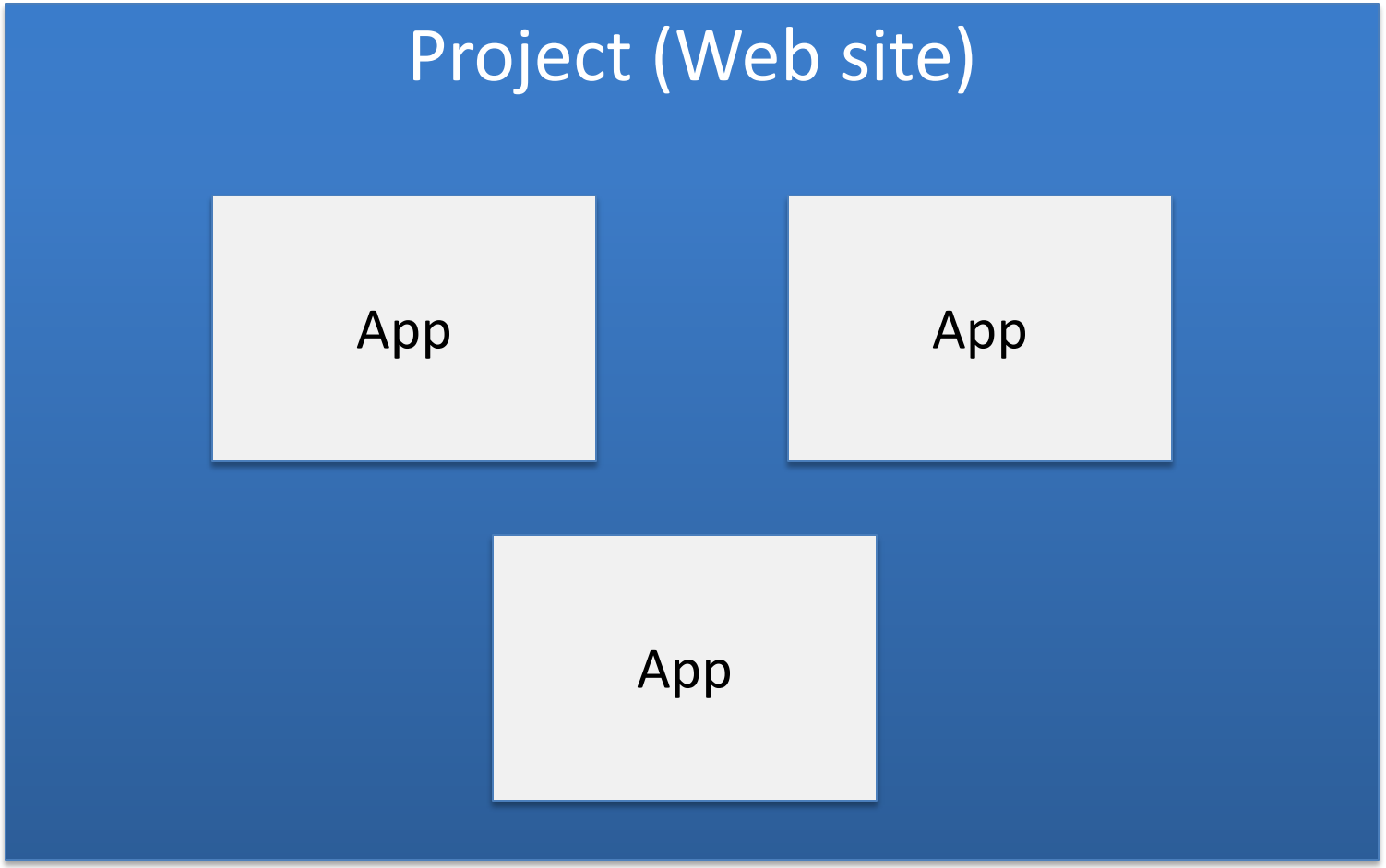
Model-View-Controller (MVC)

- A paradigm for organizing code often seen in web app frameworks
- Main idea is
  1. Separate the storage and manipulation of data (the model) and the presentation of data (view)
  2. Use the Controller to communicate between the model and view
- Advantages
  - Easier to develop and test model and view independently
  - Easier for others to understand

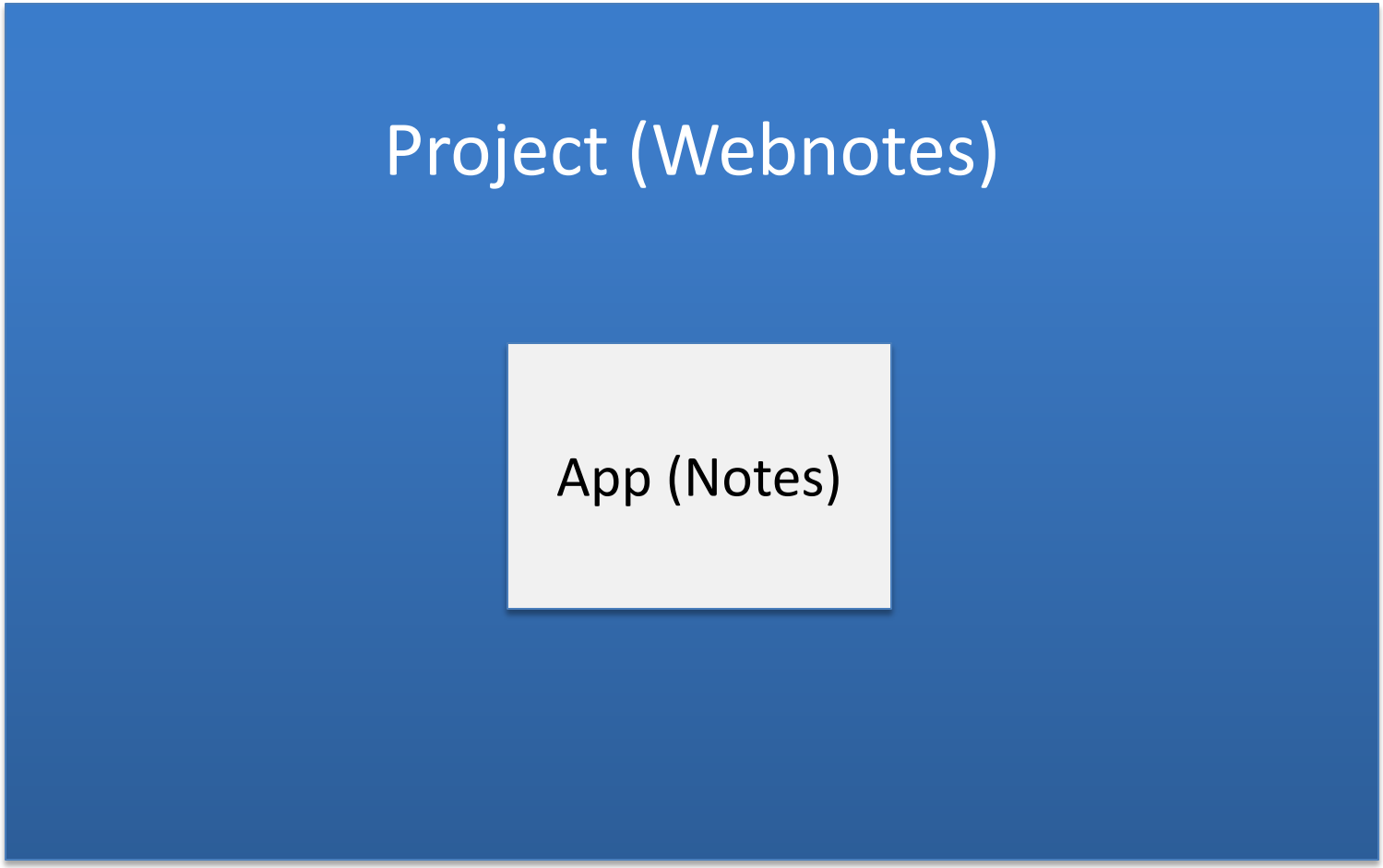
Model-View-Controller (MVC) (news site example)



Django Components



Today



Django components

- We will learn more about each individual component over the next week

HTML

- Hyper Text Markup Language

```
<h1>My Blog</h1>
<p>Welcome to
my blog!</p>
```

My Blog

Welcome to my blog!

Templates

- “View” part of MVC
- HTML pages that the site visitor will see
- Can contain content to be filled in by database

Templates

- “View” part of MVC
- HTML pages that the site visitor will see
- Can contain content to be filled in by database

```
<body>
    <div id="content”>
        {% block content %}
        <h2>{{ note.title }}</h2>
        <p>{{ note.content }}</p>
        {% endblock %}
    </div>
</body>
```

Python settings files

- Python file setting.py that tells Django where to look for things like the templates and database

- Python file urls.py that tells Django what view to use when the user goes to each URL

Tuples

- The settings file uses a data structure called a tuple
- Python immutable list
- a = (1,2,3)

Models

- Tells Django how to structure the database
- Python class
- Models are how the programmer communicates with the database
- You don’t use SQL

```
class Notes(models.Model):
    title = models.CharField(max_length=255)
    content = models.TextField()
```

Views

- “Controller” part of MVC
- Python code to tell the website which HTML (template) to display and which information to get from the database (Model)
- We will give you this code for now

```
def notes_list(request):
    note_list = Notes.objects.all()
```

Server

- You will run a mini local server on your own computer
- That way, you can open a browser and see your work!

Overview

- Templates
- Models
- Views
- Python files

Django Lab 1

- Read instructions carefully
- The instructions tell you exactly what to type
- You can copy and paste long chunks of code from the lab instructions
- Follow instructions very carefully