# Introduction

Web Applications and Services
Spring Term

Naercio Magaia

UNIVERSITY
OF SUSSEX

# Contents



UNIVERSITY OF SUSSEX

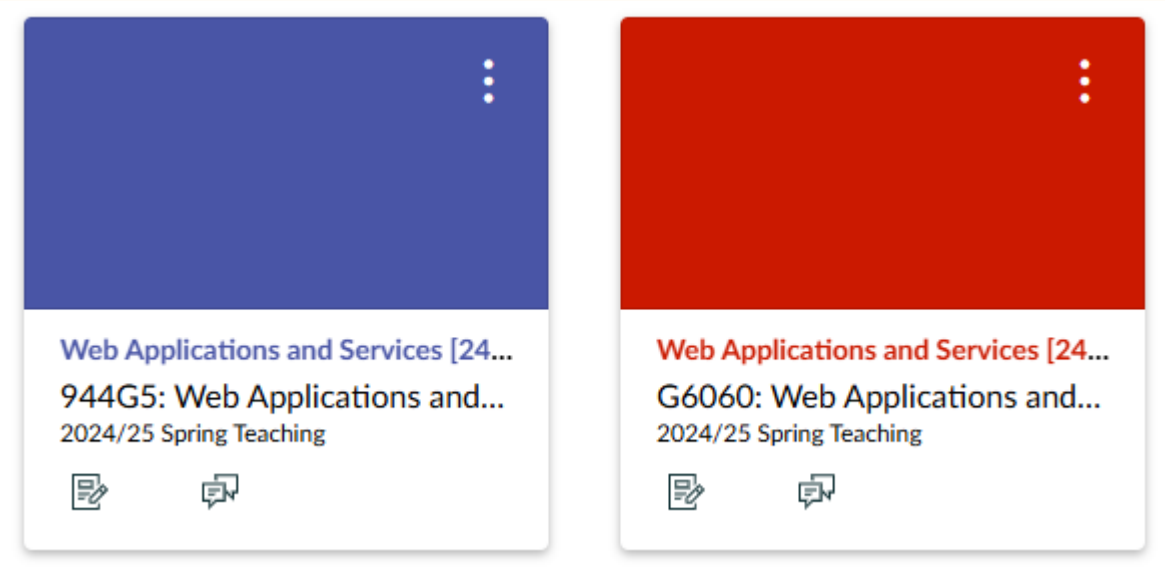
# Teachers

- Module Convenor
  - Naercio Magaia
    - Office: Chichester 2R220
    - Email: N.Magaia@sussex.ac.uk
    - Office Hours: Mondays 11:00 – 12:00 (on Microsoft Teams and in-person). Please email first.
- Lab Tutor(s)
  - Renhui Ying
    - Email: R.Ying@sussex.ac.uk
  - Xinyuan Hu
    - Email: xh233@sussex.ac.uk
  - Weibo Chen
    - Email: wc296@sussex.ac.uk

UNIVERSITY OF SUSSEX

# Teaching Materials

- All module teaching materials are at G6060: Web Applications and Services
- Lectures are shared by undergraduate (G6060) and postgraduate students (944G5)




UNIVERSITY OF SUSSEX

# Teaching Sessions

- Two lectures per week
  - Monday (Arts A A02) 9:00-10:00
  - Friday (Chichester 1 LT) 12:00-13:00

UNIVERSITY OF SUSSEX

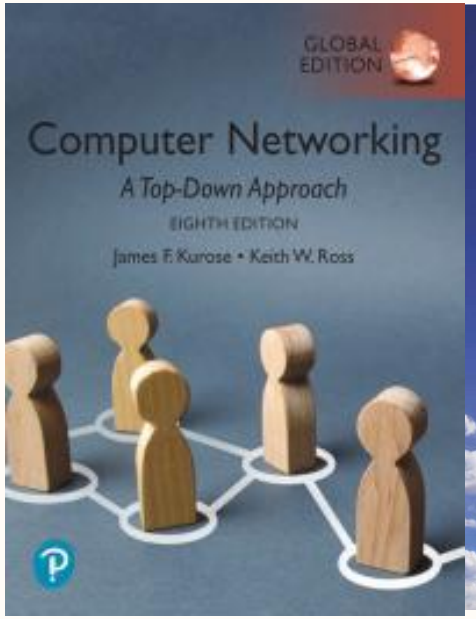
# Assessment

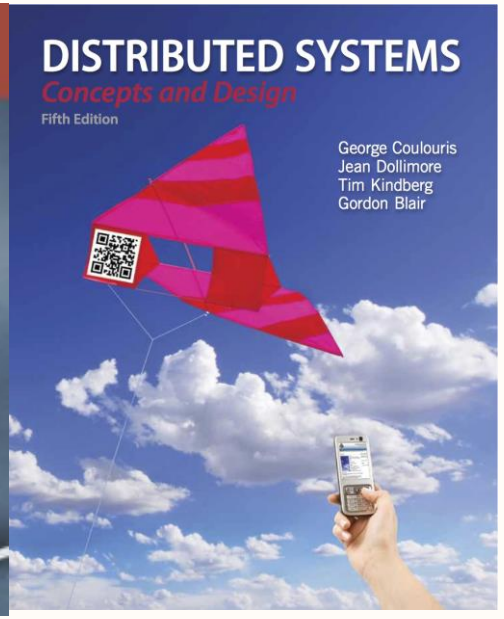
- Coursework (50% of total module marks)
  - Coursework will build on the labs, to be released later in the term
  - Due date: Week 11
- Exam (50% of total module marks)
  - Exam can draw from any of the material in the lectures or the labs
  - Due Date: TBA
- Ensure to always double-check assessment deadlines in Sussex Direct and Canvas.

UNIVERSITY OF SUSSEX

# Readings

- Readings are on Canvas and the module reading list as eBooks and physical books:
  - Distributed Systems: Concepts and Designs, George Coulouris, 2012
  - Computer Networking: A Top-Down Approach, Kurose and Ross, 2021
  - Django framework
    - https://docs.djangoproject.com/en/5.1/
- The module draws upon much of the supplied material from these books




python™ django

UNIVERSITY OF SUSSEX

# Lab classes

- Lab classes will be programming-based and connected with the lectures
- It is important to understand the technologies and work with your assignment
- Technologies:
  - Python, Django, Apache Thrift, ZooKeeper
  - PyCharm Pro
    - Lab PCs: Software Hub and look for “PyCharm Pro “.
    - Laptops: https://www.jetbrains.com/community/education/#students

UNIVERSITY OF SUSSEX

# Requirements

- You will attend lectures and lab classes
- You will attempt to complete each lab exercise during the week if there isn't enough time during class
- In case of doubts, please ask:
  - during the lecture/labs
  - drop-in sessions
  - post on forum
  - send an email

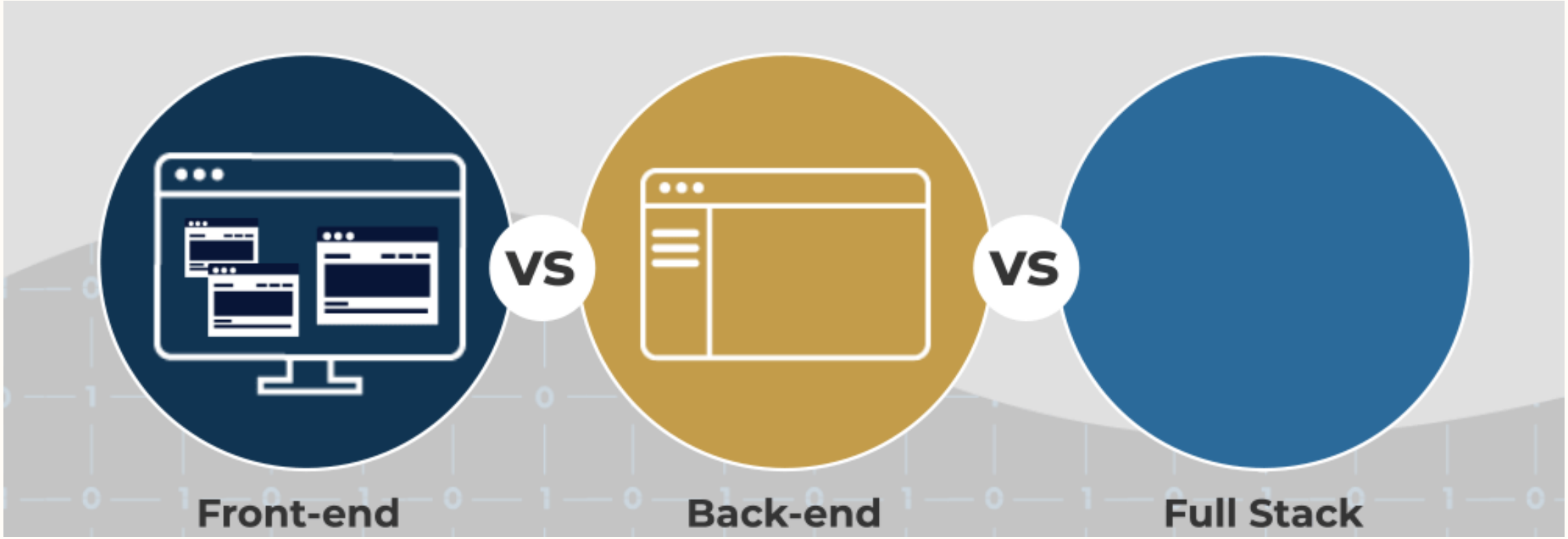
UNIVERSITY OF SUSSEX

# Module Outline

| Week | Lecture | Lecture | Lab Class | Assignment |
|---|---|---|---|---|
| 1 | Introduction | HTTP, Caching, and CDNs | AWS | |
| 2 | HTTP, Caching, and CDNs | Views | HTTP Server | |
| 3 | Templates | Forms | Views | |
| 4 | Models | Models | Templates | |
| 5 | Security | Security | Forms | **release** |
| 6 | Security | Transactions | Models | |
| 7 | Transactions | Remote Procedure Call | Security | |
| 8 | Web Services | Web Services | Transactions | |
| 9 | Time | Elections | Apache Trift | |
| 10 | Group Communication | Coordination and Agreement | Web Services | |
| 11 | Coordination and Agreement | Revision | ZooKeeper Lab | **submission** |

UNIVERSITY OF SUSSEX

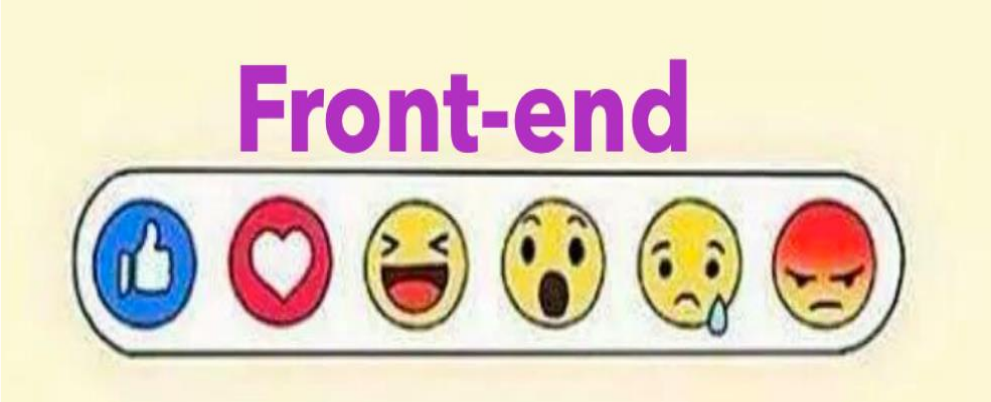
# Web Development




UNIVERSITY OF SUSSEX

# Web Development: Front-end

- Everything that a user sees in your web app is a part of front-end or client-side web app development.




- Front-end technologies:
  - HTML (Hypertext Markup Language)
  - CSS (Cascading Style Sheet)
  - JavaScript, etc.

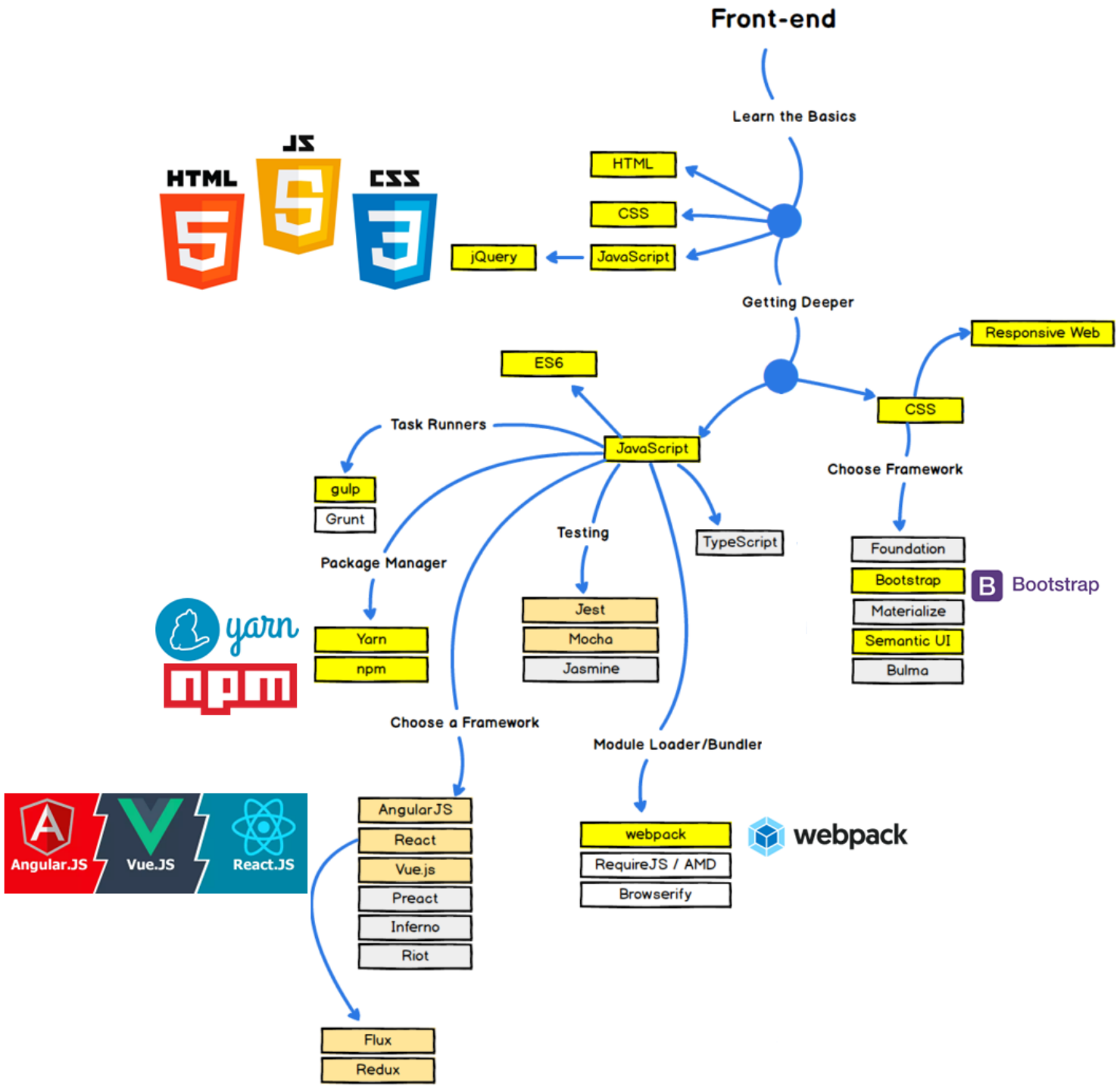

Headings, paragraphs, lists, etc.

Font, colour, border, etc.

Dynamic display. User interaction, click to open a popup, etc.

UNIVERSITY OF SUSSEX

# Front-end Roadmap

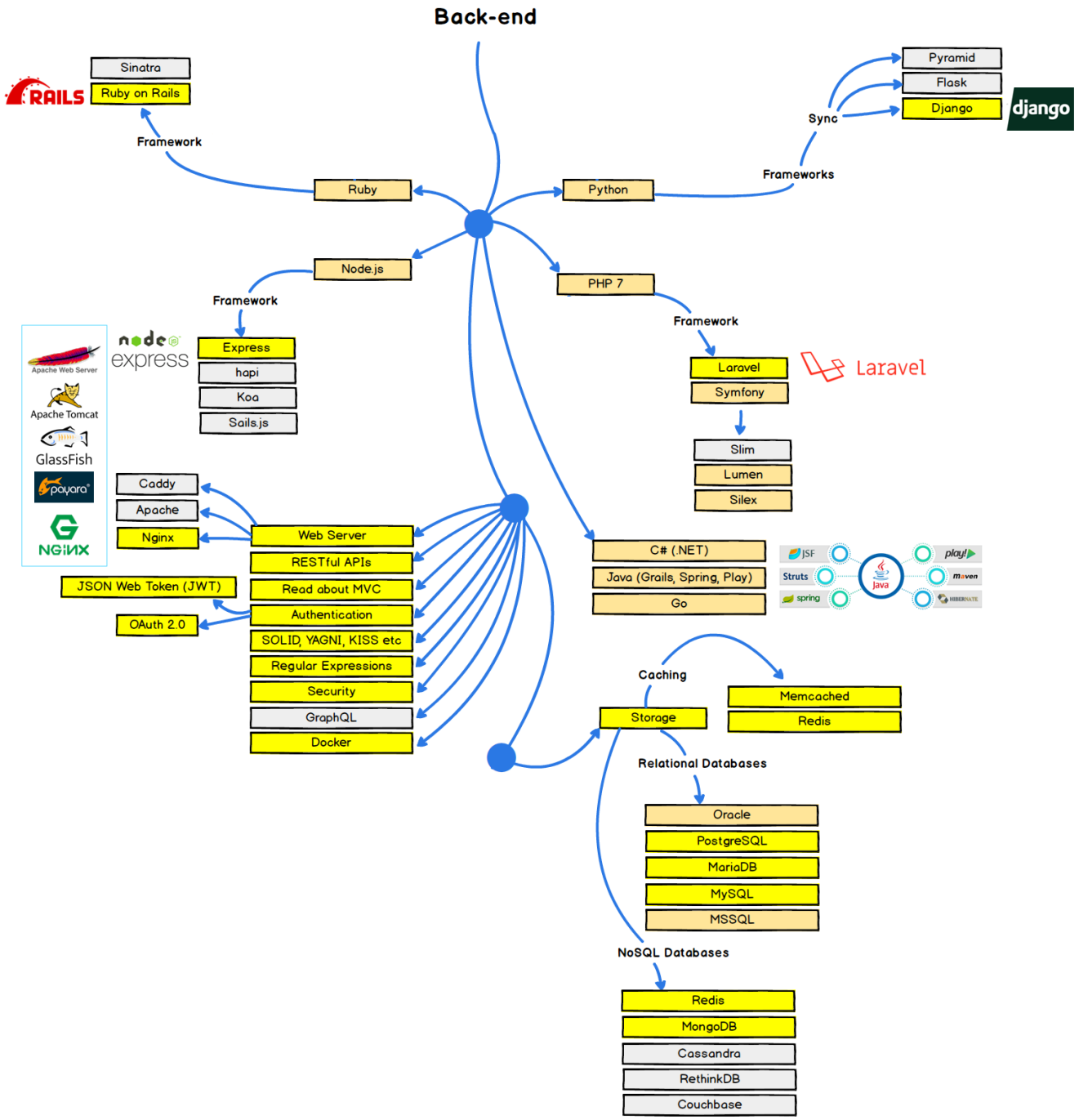


# Web Development: Back-end

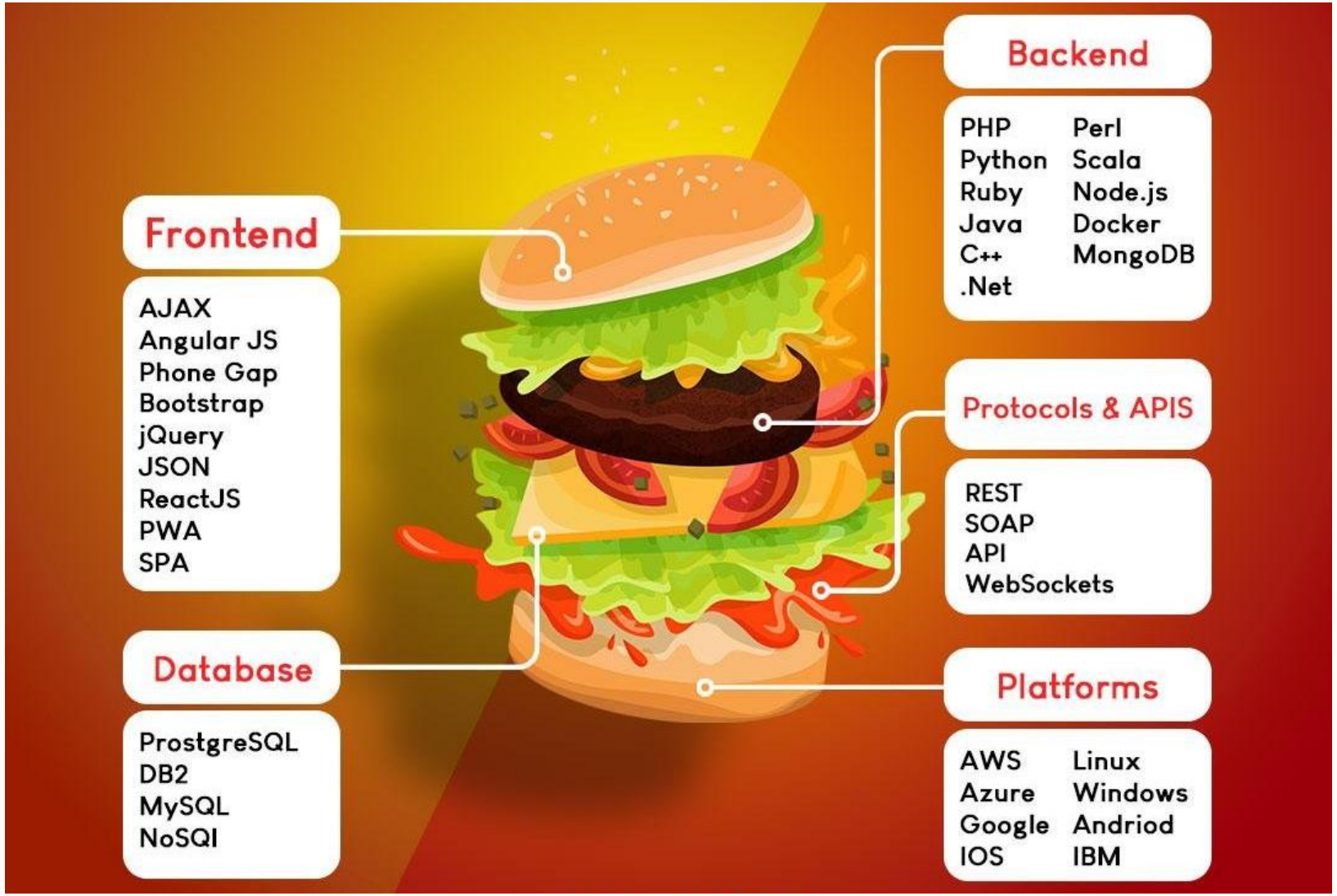
- Back-end, or server-side development, is responsible for how your web app functions.
- Back-end Programming Languages:
  - Java, PHP, Python, JavaScript, Ruby, etc.


UNIVERSITY OF SUSSEX

# Back-end Roadmap

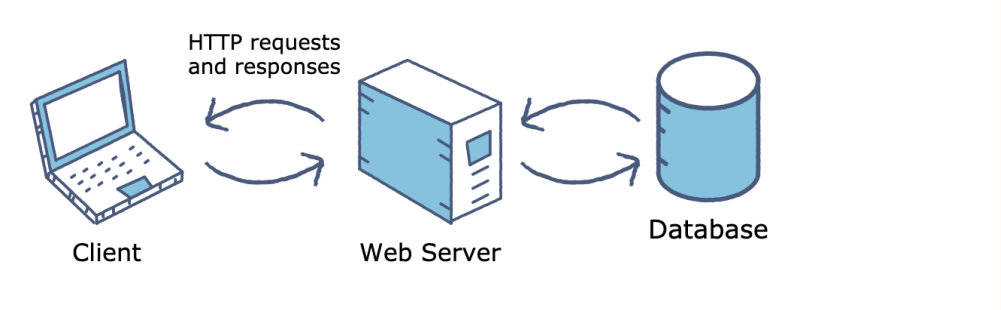


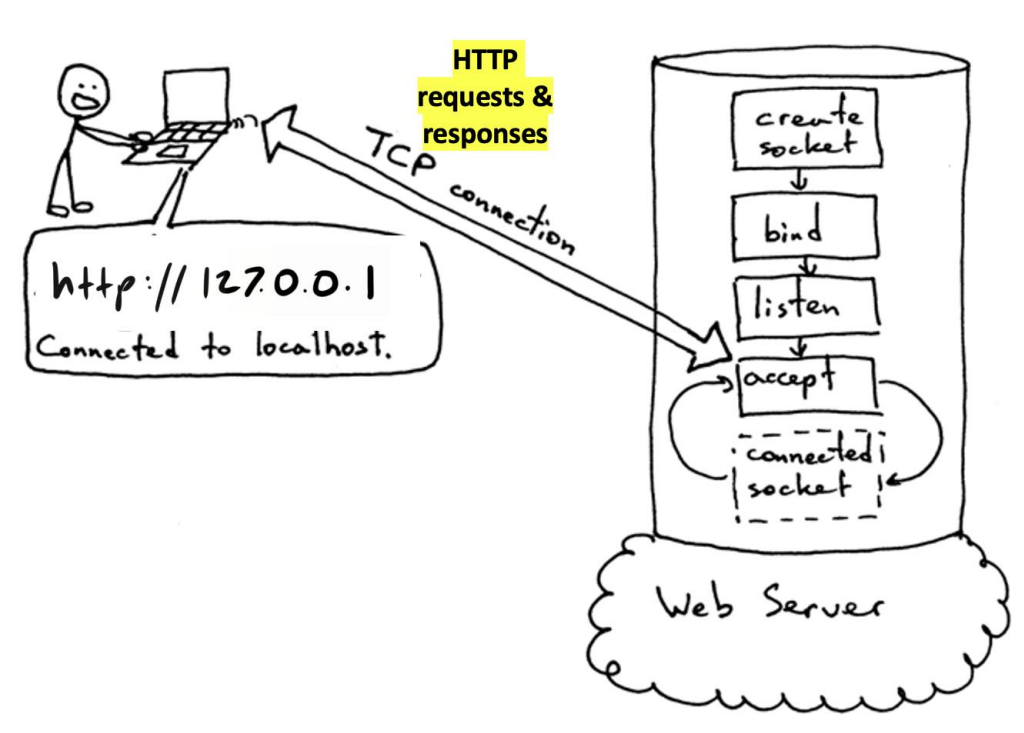
# Full Stack Developer



# Web Servers

- A web client (e.g., a web browser) communicates with a web server using HTTP protocol.
- The web server receives HTTP requests and sends back the requested content (i.e., HTML pages, files, images, ..).
- The client interprets the returned HTTP response and displays it appropriately.
- Most web servers support server interfaces used to generate dynamic content by web applications.

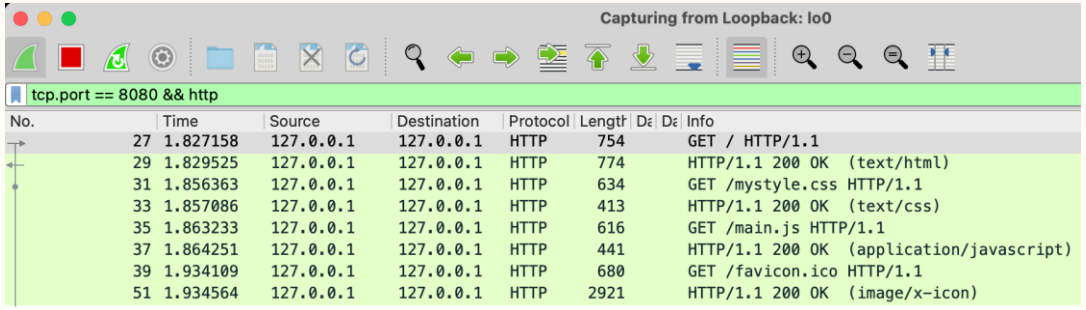


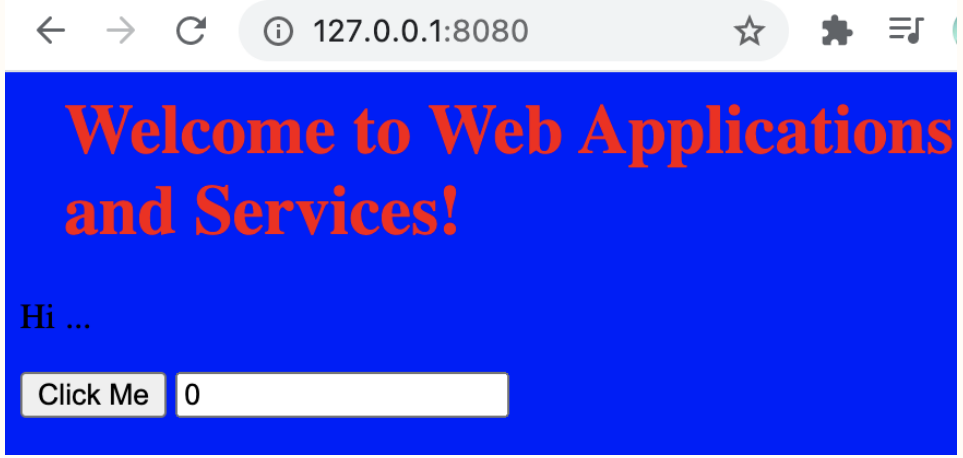

UNIVERSITY OF SUSSEX

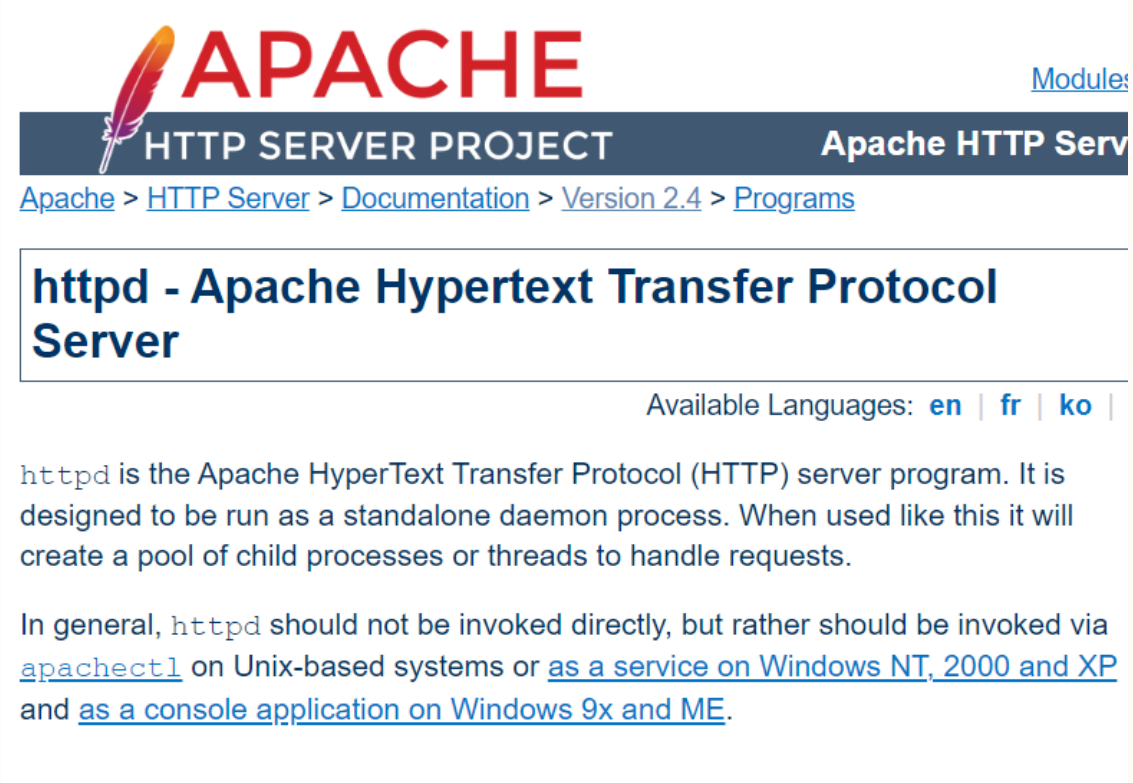
# Web Servers: demo

- http://localhost or http://127.0.0.1




| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 27 | 1.827158 | 127.0.0.1 | 127.0.0.1 | HTTP | 754 | GET / HTTP/1.1 |
| 29 | 1.829525 | 127.0.0.1 | 127.0.0.1 | HTTP | 774 | HTTP/1.1 200 OK (text/html) |
| 31 | 1.856363 | 127.0.0.1 | 127.0.0.1 | HTTP | 634 | GET /mystyle.css HTTP/1.1 |
| 33 | 1.857086 | 127.0.0.1 | 127.0.0.1 | HTTP | 413 | HTTP/1.1 200 OK (text/css) |
| 35 | 1.863233 | 127.0.0.1 | 127.0.0.1 | HTTP | 616 | GET /main.js HTTP/1.1 |
| 37 | 1.864251 | 127.0.0.1 | 127.0.0.1 | HTTP | 441 | HTTP/1.1 200 OK (application/javascript) |
| 39 | 1.934109 | 127.0.0.1 | 127.0.0.1 | HTTP | 680 | GET /favicon.ico HTTP/1.1 |
| 51 | 1.934564 | 127.0.0.1 | 127.0.0.1 | HTTP | 2921 | HTTP/1.1 200 OK (image/x-icon) |



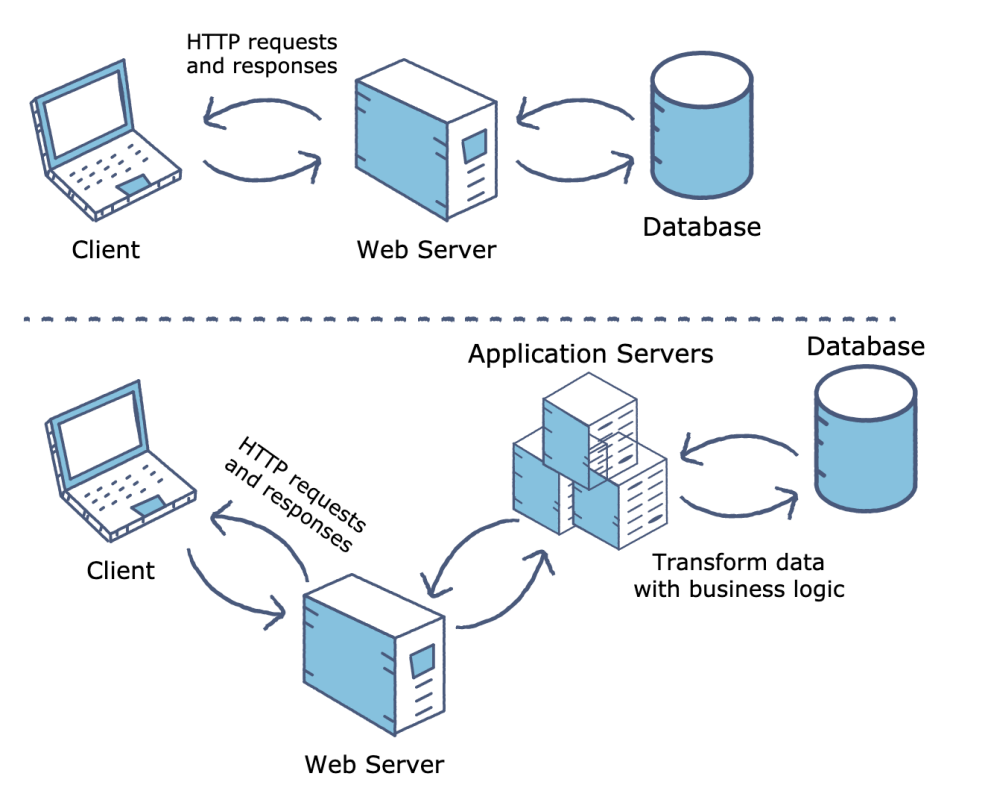




https://httpd.apache.org/docs/2.4/en/programs/httpd.html

UNIVERSITY OF SUSSEX

# Application Servers

- An application server exposes business logic to the clients, which generates dynamic content.
- It is a software framework that transforms data to provide the specialised functionality offered by a business, service, or application.



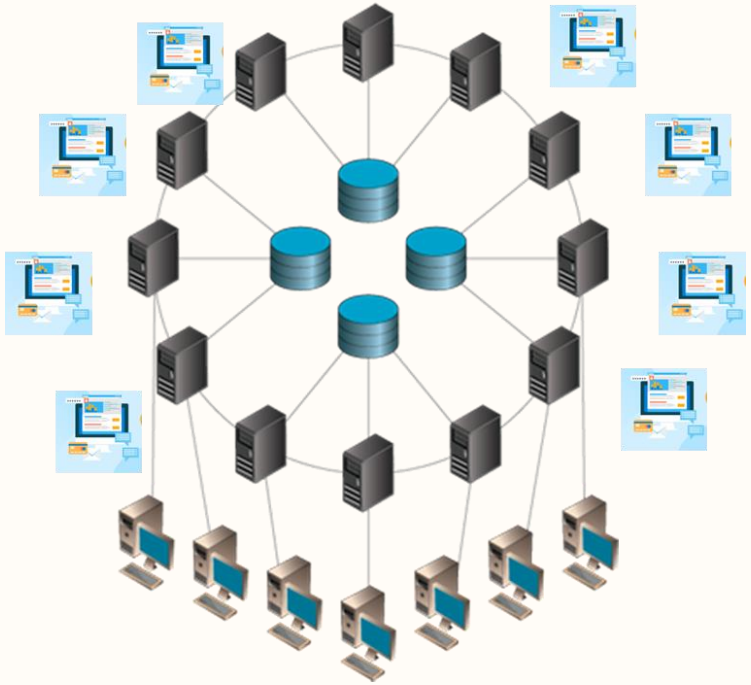

UNIVERSITY OF SUSSEX

# Web Applications vs Distributed Systems

- Web Applications

**Django, Java EE, etc.**

- Distributed Systems



Distributed information (e.g., banking), Distributed file systems (e,g., HDFS), Parallel computation (e.g., MapReduce), Distributed rendering in computer graphics

**Apache Thrift, Zookeeper, etc.**

UNIVERSITY OF SUSSEX

# Cloud Platforms

- Data centres
  - Thousands of servers
    - Explore a Google data center with Street View

  - Amazon Web Services (AWS)
    - For example, Amazon Elastic Compute Cloud (EC2)
    - AWS global infrastructure
    - EC2 On-Demand Instance Pricing

UNIVERSITY OF SUSSEX

# Django Framework

- Is a leading open-source backend framework based on the Python programming language
- Aims primarily to easy the creation of complex, database-driven websites
- Used in well-known global companies, e.g., Instagram, National Geographic, Mozilla, Spotify, Pinterest, Disqus, Bitbucket, Eventbrite, and Prezi.

| Version | Release Date |
|---|---|
| 0.90 | Nov/05 |
| ... | |
| 3.2 (LTS) | Apr/21 |
| 4.0 | Dec/21 |
| 4.1 | Aug/22 |
| 4.2 (LTS) | Apr/23 |
| 5.0 | Dec/23 |
| 5.1 | Aug/24 |

# Why Django? (1/2)

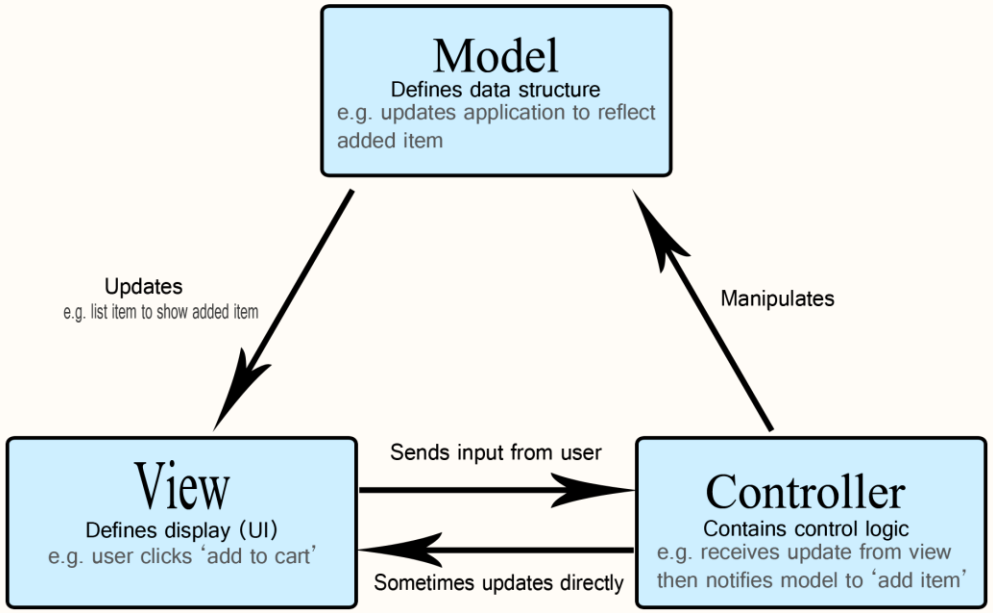
- Django remains in the top ten for the most commonly used web framework.
- Simple
  - Using the framework for development is fast and simple.
  - Pluggability and reusability mean that developers can take parts of the codebase and repurpose them elsewhere in their programming.
- Easy
  - Depends on Python, the most wanted programming language. The Python philosophy emphasizes code readability.

# Why Django? (2/2)

- Comprehensive
  - Equipped with most (if not all) of the libraries and tools that one'll ever want, including a template engine, Django ORM, multi-site support, authentication, HTTP libraries, and more.
- Secure
  - Built-in mitigation for attacks such as cross-site request forgery, cross-site scripting, clickjacking, SQL injection, among others
  - Releases new security patches in a timely manner.
- Scalable
  - Allows using clustering or load-balancing to distribute the application across servers

# Django Architecture (1/2)

- *Relies* on traditional model-view-controller (MVC) architecture
  - Model (i.e., the data-access portion) is handled by Django's database layer.
  - View (i.e., the portion that selects which data to display and how to display it) is handled by views and templates.
  - Controller (i.e., the portion that delegates to a view depending on user input) is handled by the framework itself by following the URLconf and calling the appropriate Python function for the given URL.




UNIVERSITY OF SUSSEX

# Django Architecture (2/2)

- However, it has been referred to as an MTV framework, as most in Django happens in models, templates and views. In such development pattern,
  - The data access layer (i.e., Model) contains **anything and everything about the data**: how to access it, how to validate it, which behaviors it has, and the relationships between the data.
  - The presentation layer (i.e., Template) contains **presentation-related decisions**: how something should be displayed on a Web page or other type of document.
  - The business logic layer (i.e., View) contains **the logic that access the model and defers to the appropriate template(s)**. You can think of it as the bridge between models and templates.

# Django Framework

- Django includes
  - a lightweight and standalone web server for development and testing
  - a form serialization and validation system that can translate between HTML forms and values suitable for storage in the database
  - a template system that utilizes the concept of inheritance borrowed from object-oriented programming
  - a serialization system that can produce and read XML and/or JSON representations of Django model instances
  - an interface to Python's built-in unit test framework

# Learning Outcomes

- Learn technologies underpinning the design of distributed web applications.
- Solve problems in the design of web applications and services, using a variety of related technologies.
- Build programs using a range of services available within the Django web framework.
- Understand the problems of security and of concurrency and synchronisation across replicated services.

# Next Lecture ...

- ✓ Introduction
- ➢ **HTTP, Caching, and CDNs**
- Views
- Templates
- Forms
- Models
- Security
- Transactions
- Remote Procedure Call
- Web Services
- Time
- Elections and Group Communication
- Coordination and Agreement

UNIVERSITY OF SUSSEX