TYING EVERYTHING TOGETHER - THE THREE-TIER ARCHITECTURE

Lecture 10

Joe Appleton

CSEE

LAST WEEK WE

- Considered the role of a web framework in the development of a web application
- Considered the role of a web server in the development of a web application
- Used the Express framework to create a web application
- Use middleware in a web application
- Understand how to use `ejs` templates to create dynamic web pages
- Used `npm` to manage all of this

A CODE RECAP

THIS WEEK

- This week, we will conclude the web part of the module.
- Specifically, we will consider:
  - ✓ How to Create a full-stack web application using Node.js, Express, and **MySQL.**
  - ✓ The various compliance issues and best practices surrounding the development of web applications.
  - ✓ How to implement simple data validation on the server and client side.
  - ✓ Understand the three-tier architecture.

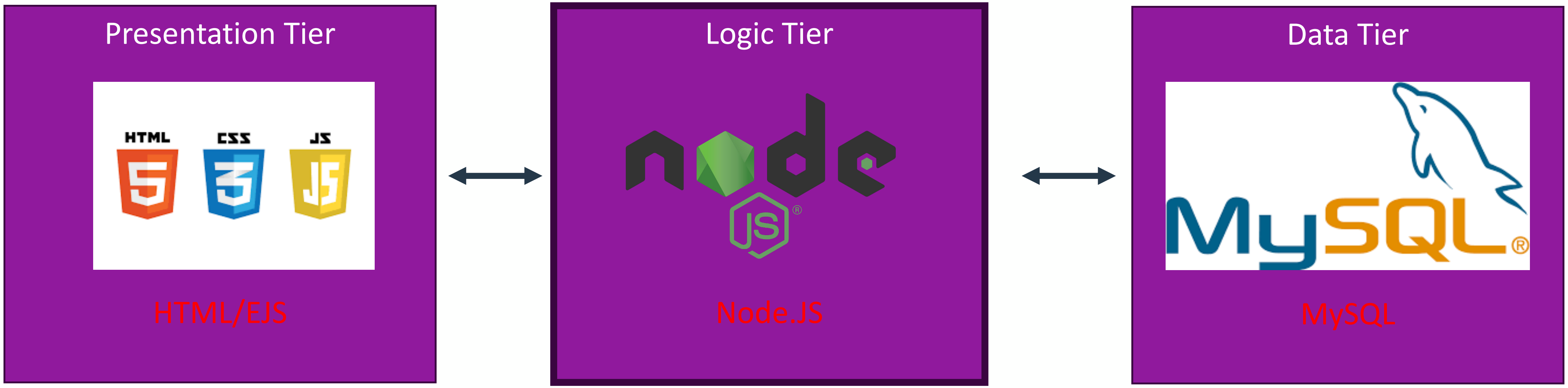
DISCLAIMER ABOUT THE LAB

- This week, I don’t expect you to complete the lab in one sitting.
- I will release a full, in-depth tutorial video on how to complete the lab.
- After completing the lab, you’ll be able to finish all non-advanced tasks of the assessment.

THE THREE-TIER ARCHITECTURE

WHAT IS A THREE TIER ARCHITECTURE?

- The three-tier architecture is a common architecture for web applications. It is a client-server architecture that separates the presentation layer, the application layer and the data layer.
- Improved scalability and maintainability.
- Clear separation of concerns.



Presentation Tier

HTML/EJS

Logic Tier

Node.JS

Data Tier

MySQL

THE PRESENTATION LAYER (CLIENT)

- is the layer that the user interacts with.
- responsible for rendering the user interface and handling user input.
- typically implemented using HTML, CSS and JavaScript.
- can create sophisticated presentation layers
- web Browsers: Browsers like Chrome, Firefox, and Safari are examples of clients that render and display web applications.
- mobile apps running on devices such as smartphones or tablets also serve as clients interacting with the presentation layer.

HTML CSS JS

APPLICATION LAYER (SERVER)

- Business logic and application processing
  - authentication,
  - data validation
  - business rules
- Communication between presentation and data layers.
- Examples: Node.js, Django, Flask.

DATA LAYER (DATABASE)

- manages the storage of structured data in a database system.
- ensures efficient and secure access to stored information.
- handle tasks such as data organisation, indexing, and ensuring data integrity.
- examples: MySQL, PostgreSQL, MongoDB.

POSSIBLE DRAWBACKS OF A 3-TIER ARCHITECTURE

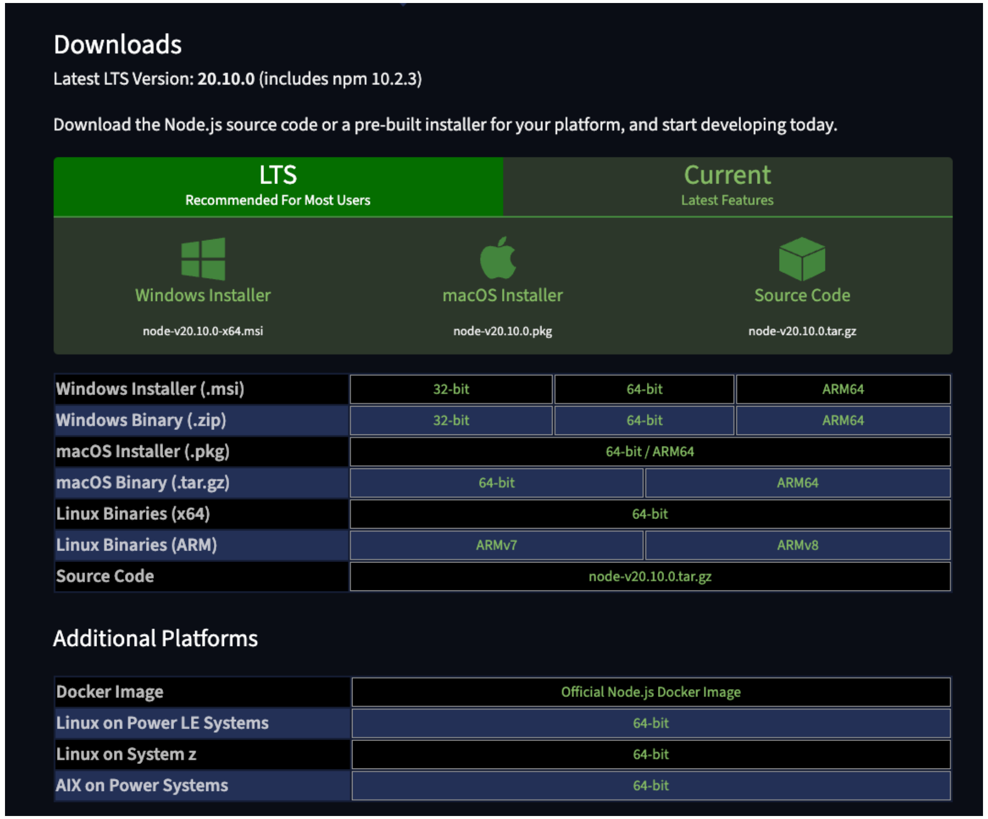
- Single point of failure.
- The communication between layers involves passing data back and forth. This can lead to increased overhead, especially if not managed efficiently.
- Developing a three-tier architecture can sometimes take more time compared to a simple HTML/CSS application.
- Changes in one layer may require corresponding adjustments in other layers

WIDER ARCHITECTURES

- **Microservices Architecture** involves breaking down a monolithic application into smaller, independent services that can be developed, deployed, and scaled independently.
- **Serverless Architecture** based on the concept of functions as a service (FaaS), where functions are triggered by events. These events can be HTTP requests, database changes, file uploads, etc.

CONNECTING TO A DATABASE

FIRST, WE NEED A DATABASE TO CONNECT TO AND NODEJS INSTALLED



Downloads

Latest LTS Version: **20.10.0** (includes npm 10.2.3)

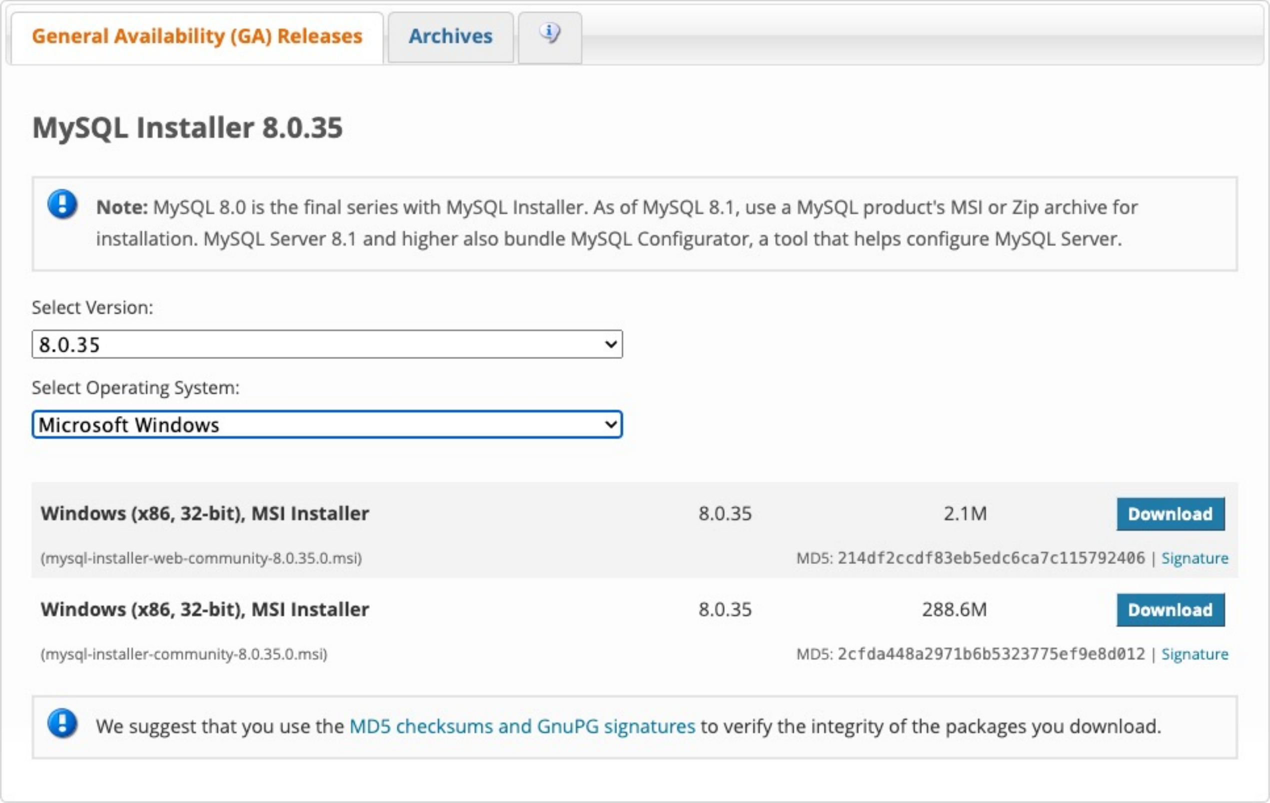
Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Windows Installer (.msi)	32-bit	64-bit	ARM64
Windows Binary (.zip)	32-bit	64-bit	ARM64
macOS Installer (.pkg)	64-bit / ARM64		
macOS Binary (.tar.gz)	64-bit	ARM64	
Linux Binaries (x64)	64-bit		
Linux Binaries (ARM)	ARMv7	ARMv8	
Source Code	node-v20.10.0.tar.gz		

Additional Platforms

Docker Image	Official Node.js Docker Image
Linux on Power LE Systems	64-bit
Linux on System z	64-bit
AIX on Power Systems	64-bit

https://nodejs.org/en/download



MySQL Installer 8.0.35

Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.

Select Version: 8.0.35

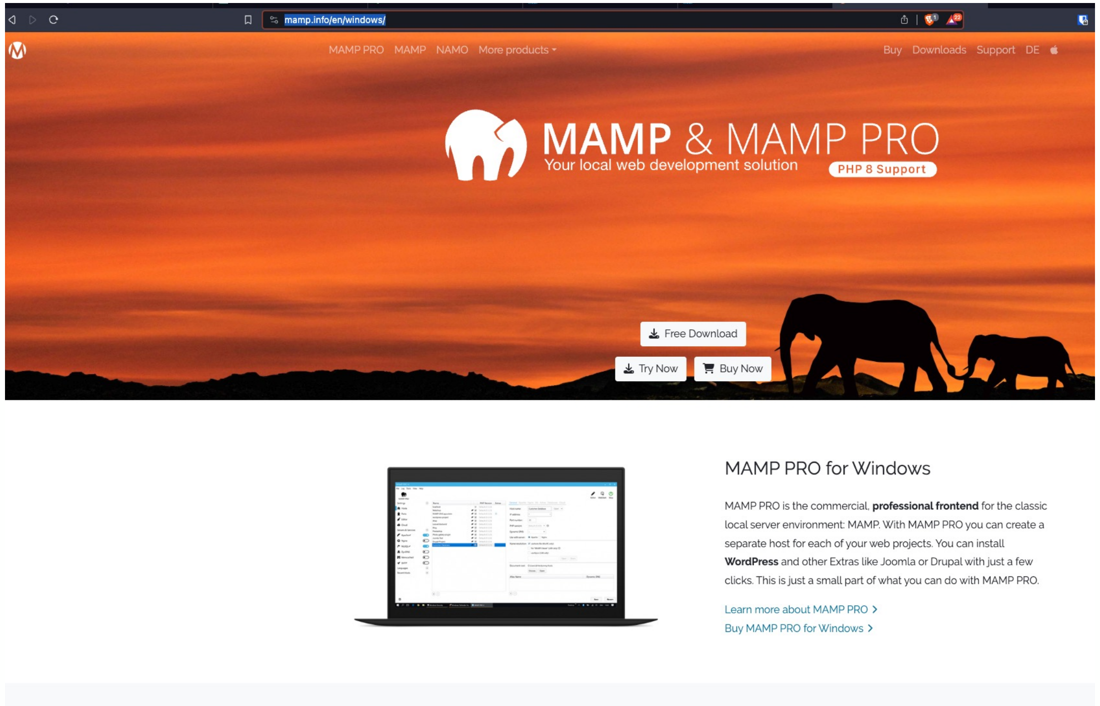
Select Operating System: Microsoft Windows

Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.35.0.msi)	8.0.35	2.1M	Download MD5: 214df2ccdf83eb5edc6ca7c115792406 | Signature
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.35.0.msi)	8.0.35	288.6M	Download MD5: 2cfda448a2971b6b5323775ef9e8d012 | Signature

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

https://dev.mysql.com/downloads/mysql/

YOU CAN ALSO INSTALL A BUNDLED MYSQL AND NODE PACKAGE



MAMP & MAMP PRO

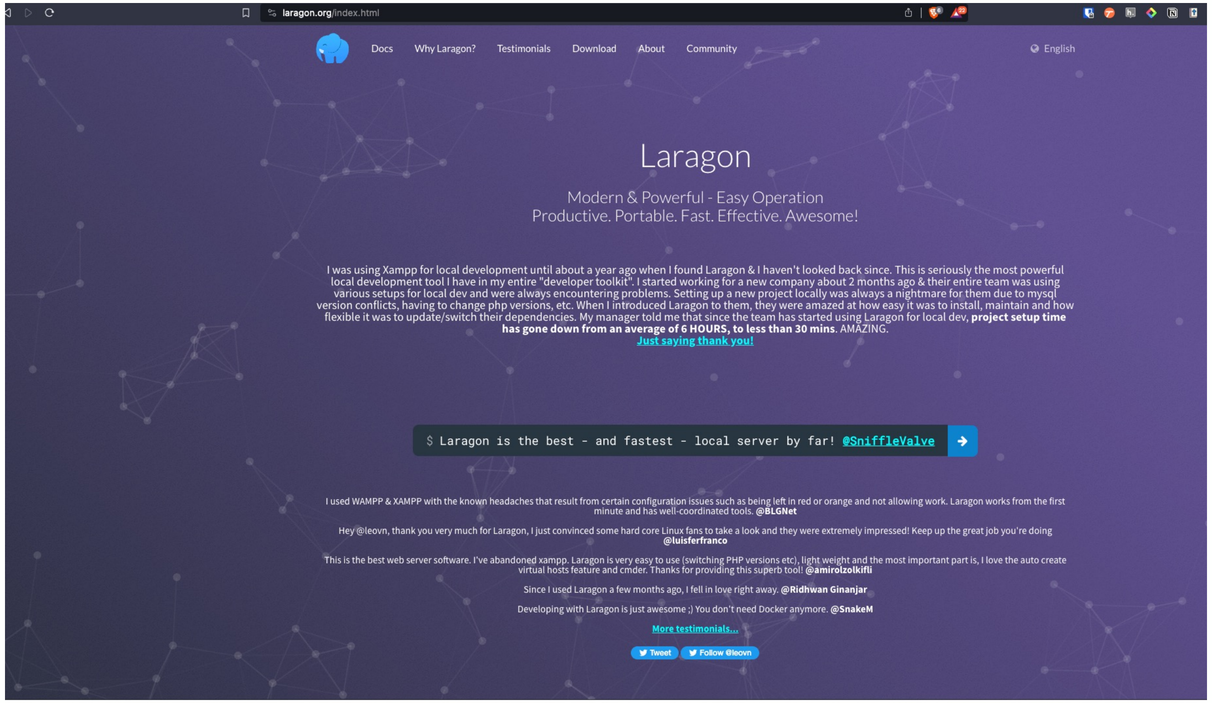
Your local web development solution

MAMP PRO for Windows

MAMP PRO is the commercial, **professional frontend** for the classic local server environment: MAMP. With MAMP PRO you can create a separate host for each of your web projects. You can install **WordPress** and other Extras like Joomla or Drupal with just a few clicks. This is just a small part of what you can do with MAMP PRO.

Learn more about MAMP PRO ›

Buy MAMP PRO for Windows ›



Laragon

Modern & Powerful - Easy Operation
Productive. Portable. Fast. Effective. Awesome!

I was using Xampp for local development until about a year ago when I found Laragon & I haven't looked back since. This is seriously the most powerful local development tool I have in my entire "developer toolkit". I started working for a new company about 2 months ago & their entire team was using various setups for local dev and were always encountering problems. Setting up a new project locally was always a nightmare for them due to mysql version conflicts, having to change php versions, etc. When I introduced Laragon to them, they were amazed at how easy it was to install, maintain and how flexible it was to update/switch their dependencies. My manager told me that since the team has started using Laragon for local dev, **project setup time has gone down from an average of 6 HOURS, to less than 30 mins**. AMAZING.
Just saying thank you!

```
$ Laragon is the best - and fastest - local server by far! @SniffleValve
```

I used WAMPP & XAMPP with the known headaches that result from certain configuration issues such as being left in red or orange and not allowing work. Laragon works from the first minute and has well-coordinated tools. **@BLGNet**

Hey @leovn, thank you very much for Laragon, I just convinced some hard core Linux fans to take a look and they were extremely impressed! Keep up the great job you're doing **@luisferfranco**

This is the best web server software. I've abandoned xampp. Laragon is very easy to use (switching PHP versions etc), light weight and the most important part is, I love the auto create virtual hosts feature and cmder. Thanks for providing this superb tool! **@amirolzolkifli**

Since I used Laragon a few months ago, I fell in love right away. **@Ridhwan Ginanjar**

Developing with Laragon is just awesome ;) You don't need Docker anymore. **@SnakeM**

More testimonials...

CONNECTING TO A DATABASE CODE

ONCE CONNECTED, WE CAN RUN MYSQL QUERIES

```
const studentCount = await connection.query("SELECT COUNT(*) as count
FROM Student");
```

- The above code runs the query "SELECT COUNT(*)as count FROM Student" and stores the result in a constant called studentCount.
- Notice how we use the **await** keyword. This is because the connection.query function is asynchronous.
- studentCount will be an array of objects. In this case, it will be an array with one object:

```
[
        {
                count: 10,
        },
]
```

```
console.log(studentCount[0].count)    // logs 10 to the console
```

QUERIES THAT RETURN MULTIPLE ROWS

```
const students = await connection.query("SELECT * FROM Student");
```

QUERIES THAT RETURN MULTIPLE ROWS

```
const students = await connection.query("SELECT * FROM Student");
```

```
[
    RowDataPacket {
      URN: 612345,
      Stu_FName: 'John',
      Stu_LName: 'Smith',
      Stu_DOB: 2002-06-19T23:00:00.000Z,
      Stu_Gender: 'M',
      Stu_Phone: '11111',
      Stu_Course: 201,
      Stu_Type: 'UG'
     },
    RowDataPacket {
      URN: 612345,
      Stu_FName: 'John',
      Stu_LName: 'Smith',
      Stu_DOB: 2002-06-19T23:00:00.000Z,
      Stu_Gender: 'M',
      Stu_Phone: '11111',
      Stu_Course: 201,
      Stu_Type: 'UG'
     },

    },
.....
```

ACCESSING VALUES IN AN ARRAY

```
students.forEach(student => { console.log(student) });
```

THE PRESENTATION LAYER

HTML AND CSS

- HTML and CSS form a powerful duo for building the presentation layer of web applications
- HTML: Provides a lightweight structure for content,
- CSS: Centralising styles in separate CSS files makes it easier to maintain and update the design of a website.
- simple to learn
- if you have not gone over my pre-learning, ensure you do this.

WHAT'S WRONG WITH JUST HTML

- Serving plain HTML files provides us with a means to present a website
- How do we inject data into our html pages?
- Shared components or layouts must be replicated across different HTML files
- As web applications grow in complexity, maintaining a large codebase written in pure HTML becomes increasingly challenging.
- The lack of a clear separation of concerns makes it difficult to manage the presentation layer independently from the application logic

EJS TEMPLATING TO THE RESCUE

- Templating allows for the dynamic generation of content in web applications

- Templating promotes code reusability by separating the HTML structure from the data

- Shared components

- Templating ensures consistent layouts across different pages of a website.

EJS ALLOWS HTML TO BE CHUNKED

```
<!-- views/common/header.ejs -->
<ul>
  <li><a href="home">Home</a></li>
  <li><a href="about"></a>About</li>
  <li><a href="contact">Contact</a></li>
</ul>
```

```
<!-- views/index.ejs -->
...
<body>
  <%- include('common/header'); %>
  <h1>Home Page</h1>
  <img src="test.jpeg" alt="wtf" />
</body>
...
```

YOU CAN ALSO INJECT DATA

```
// index.js
app.get('/', async (req, res) => {
        res.render('index’,
            {
                name: ”joe appleton”,
                location: “surrey”
            }
        ):
});
```

```
<!-- views/index.ejs -->
...
<h1> Hello World, <%= name %> </h1> <!– Hello World, joe appleton-->
<p> You are currently in <%= location %> </p> <!– You are currently in , brighton -->
...
```

YOU CAN ALSO INJECT DATA

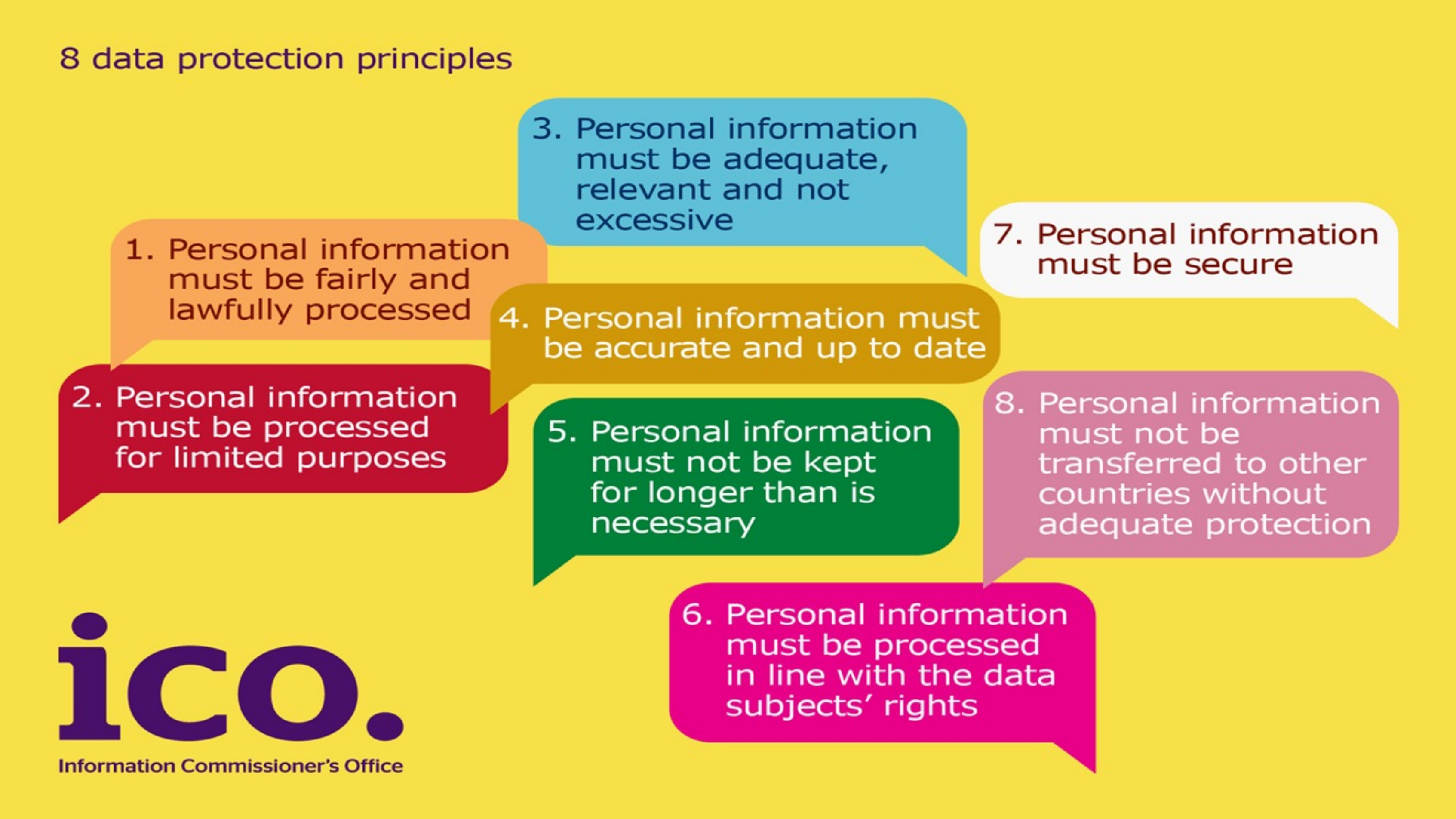
```
// index.js
app.get('/', async (req, res) => {
        res.render('index’, { students: [

                { name: "Joe Appleton", location: "Surrey" },
                { name: "Jane Doe", location: "London" },
                { name: "John Smith", location: "Manchester" },
               ] }):
});
```

```
<ul>
                <% students.forEach(function(student){ %>
                        <li>
                                <%= student.name %> - <%= student.location %>
                        </li>
                <% }) %>
</ul>
```

CONCERNS WHEN DEALING WITH USER DATA

LEGAL

Data Protection Act 1998 The eight principles



8 data protection principles

1. Personal information must be fairly and lawfully processed
2. Personal information must be processed for limited purposes
3. Personal information must be adequate, relevant and not excessive
4. Personal information must be accurate and up to date
5. Personal information must not be kept for longer than is necessary
6. Personal information must be processed in line with the data subjects' rights
7. Personal information must be secure
8. Personal information must not be transferred to other countries without adequate protection

ico.

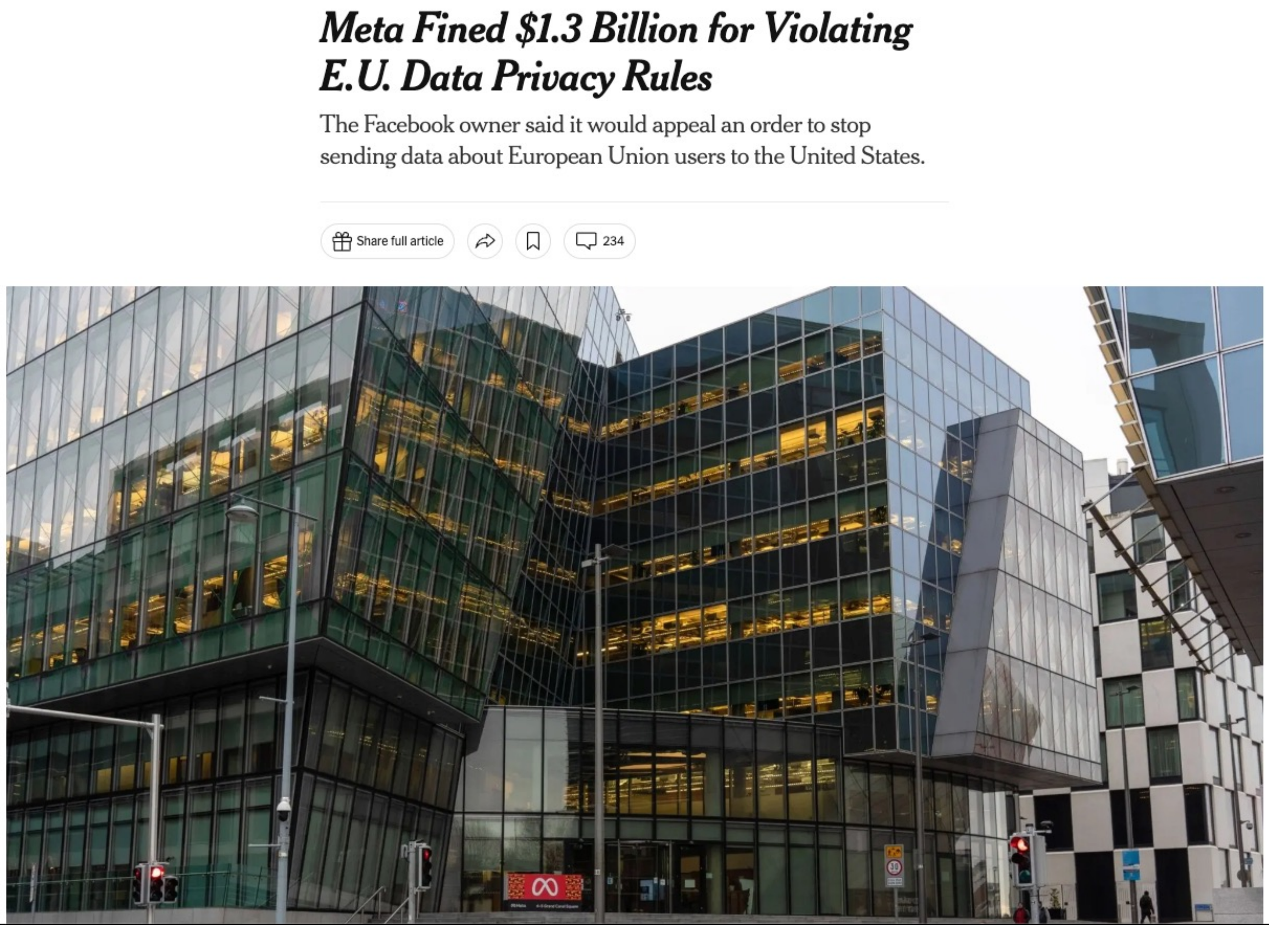
Information Commissioner's Office

British Pregnancy Advice Service

Date: 7 March 2014

British Pregnancy Advice Service fined £200,000. Hacker threatened to publish thousands of names of people who sought advice on abortion, pregnancy and contraception.

Abortion provider BPAS fined £200,000 for data breach



Meta Fined $1.3 Billion for Violating E.U. Data Privacy Rules

The Facebook owner said it would appeal an order to stop sending data about European Union users to the United States.

Meta settles Cambridge Analytica scandal case for $725m

23 December 2022

Facebook-Cambridge Analytica scandal

GETTY IMAGES

By Shiona McCallum
Technology reporter

Facebook owner Meta has agreed to pay $725m (£600m) to settle legal action over a data breach linked to political consultancy Cambridge Analytica.

The long-running dispute accused the social media giant of allowing third parties, including the British firm, to access Facebook users' personal data

THE GENERAL DATA PROTECTION REGULATION (GDPR)

Lawfulness, Fairness, and Transparency

Purpose Limitation

Data Minimisation

Accuracy

Integrity and Confidentiality (Security)

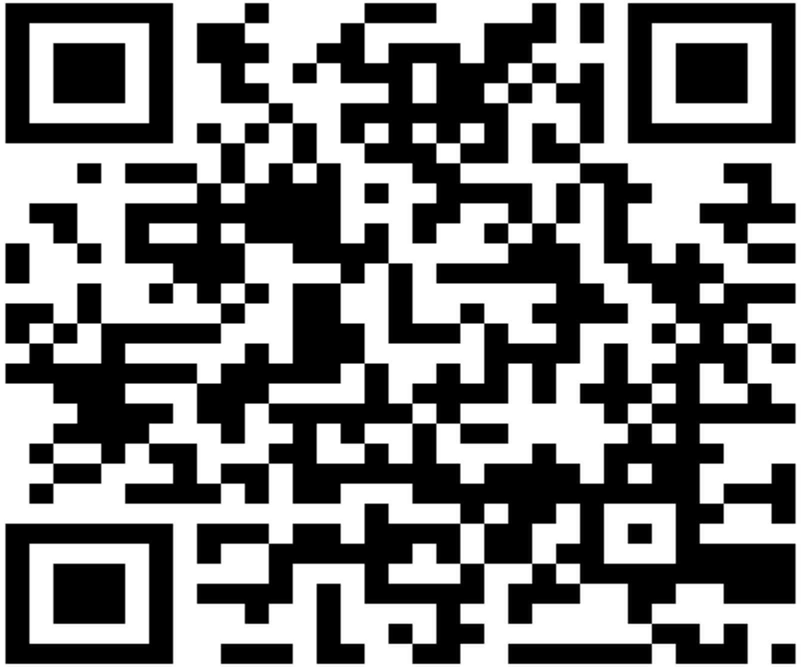
Accountability

TECHNICAL

SANITISING USER INPUT

NOT SANITIZING USER INPUT CAN LEAD TO VARIOUS SECURITY VULNERABILITIES

- ✓ **SQL Injection:**
- ✓ **Code Injection**
- ✓ **Command Injection**



Visit: https://http-server-leqm.onrender.com/

ESCAPE EJS TEMPLATES

```
<%= userName %> <!-– this sanitises the output -->

<%- userName %> <!-– this  this does not sanitise the output -->
```

USE PARAMETERIZED QUERY TO PREVENT SQL INJECTION

```
app.get('/students/view/:id', async (req, res) => {
	const student = await connection.query('SELECT * FROM STUDENT WHERE URN = ?', [req.params.id]);
	res.render('student_view', { student: student[0] });
});
```

END OF LECTURE

DEPLOYMENT