Fullstack Development

CS571: Building User Interfaces

Cole Nelson & Yuhang Zhao

Before Lecture

1. Clone today's starter code and run `npm install` in the starter and solution folders.
2. Download & install Docker

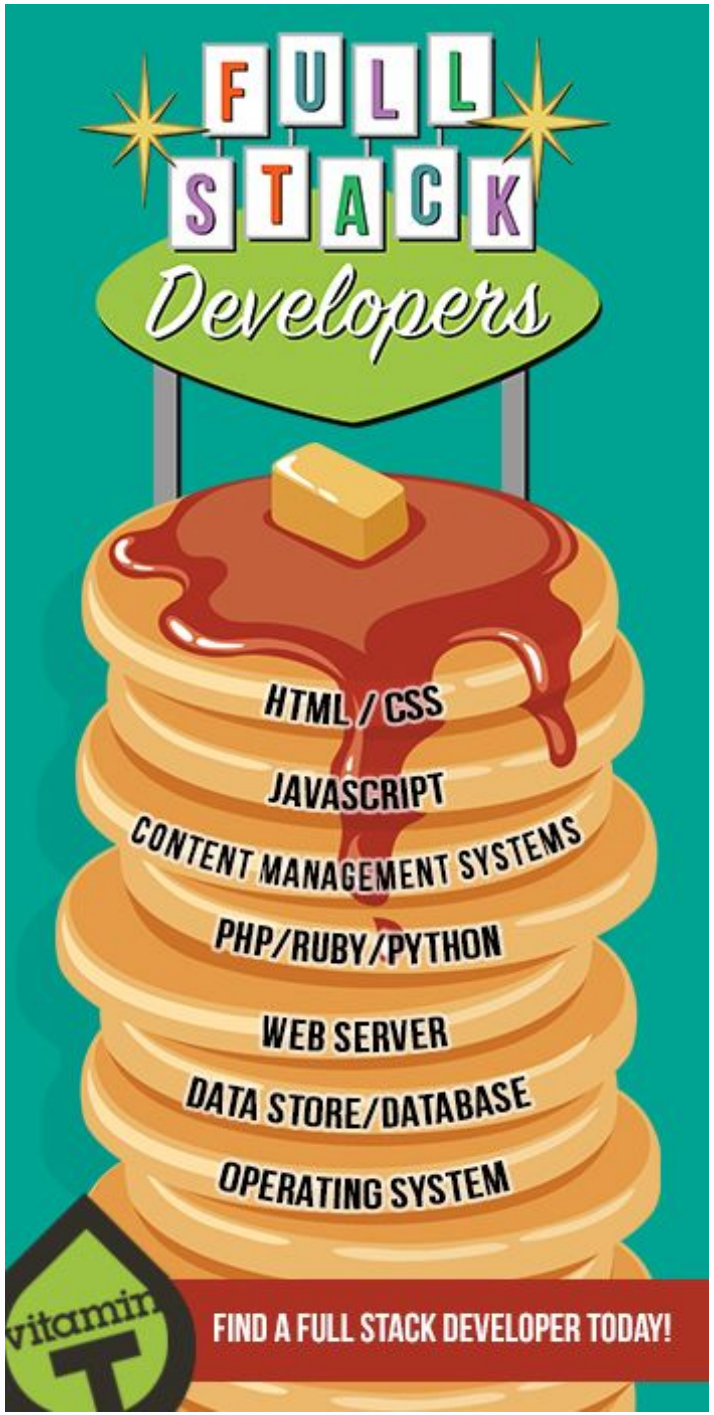
What will we learn today?

- What is the software stack?
- How can we develop a backend?
- How can we persist data?
- How can we use containerization?
- What are other considerations?

Software Stack

Think of software like a stack of pancakes...

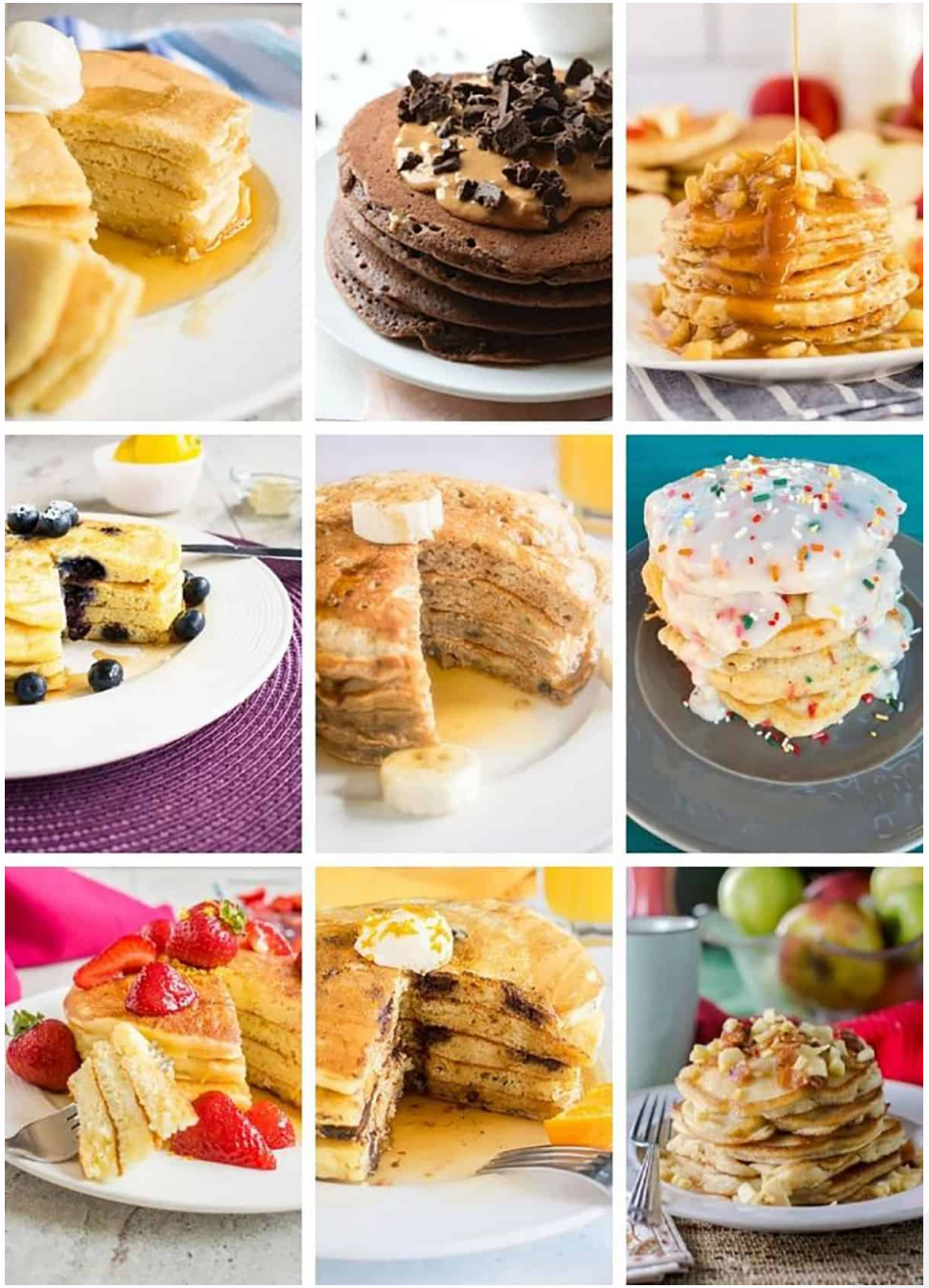
Image Source



Software Stack

... where each pancake can be its own flavor...

Image Source



Software Stack

... and can be cooked its own way...

Image Source

Software Stack

... with as many or as few as we want!

Image Source

Our Software Stack

JavaScript and React for frontend development.

JavaScript and Express for backend development.

When you build your project, you get to choose your software stack!

Setting up your own React project!

Use vite (or similar). create-react-app is dead.

```
npm create vite@latest my-cool-app -- --template react
```

Don't forget about Bootstrap!

```
npm install bootstrap react-bootstrap
```

Bootstrap also has additional setup.

BadgerChat Mini

Building the UI.

How to persist data?

Building a backend!

Creating a Backend Server

Many, many, many options!

- Google Cloud Functions
- AWS Lambdas
- C# & .NET
- Java & Spring
- Python & Flask
- **JavaScript & Express**

BadgerChat Mini

Building the API.

How to persist data?

Let's use SQLite

SQLite

- SQL, but lite!
- Creates a `.db` file on your machine
- Is not a "hosted" database, but is good for quick projects and hacks!
- Handles our concurrency issues.

BadgerChat Mini

Building the DB.

This is great but...

...how can we deploy this?

Deployment

Generally accepted: a server is a piece of computer hardware or software that provides functionality for other programs or devices, called "clients". - Wikipedia

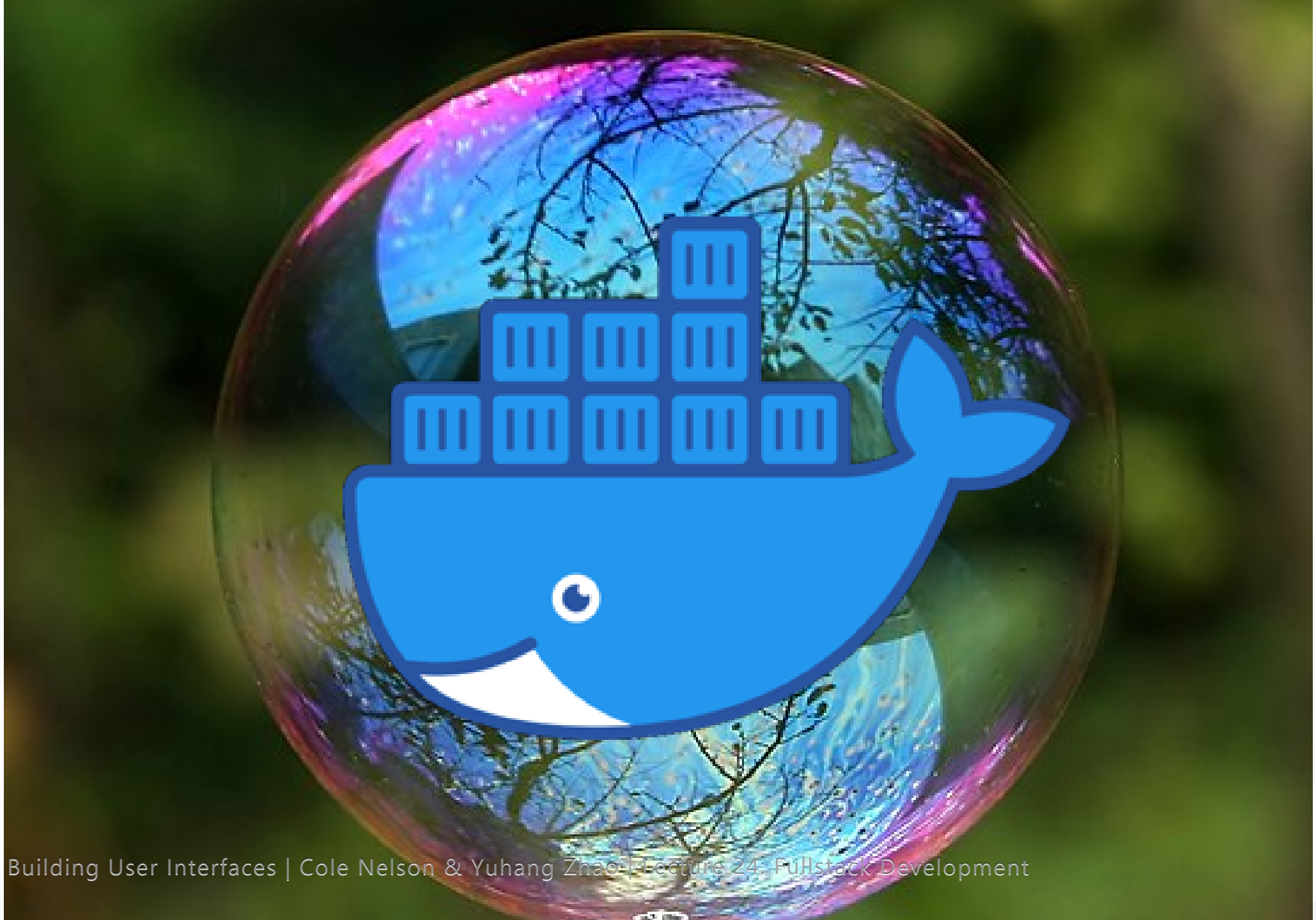
More cynical: a server is just another computer with a bunch of holes in its firewall.

Deployment

Run the setup commands, then...

1. Open the ports on your machine & router (or use a reverse proxy tool like ngrok).
2. Open the ports on a remote machine.

Still... how do we *isolate* ourselves? How do we make the environment *portable*? **Use a VM or a container!**



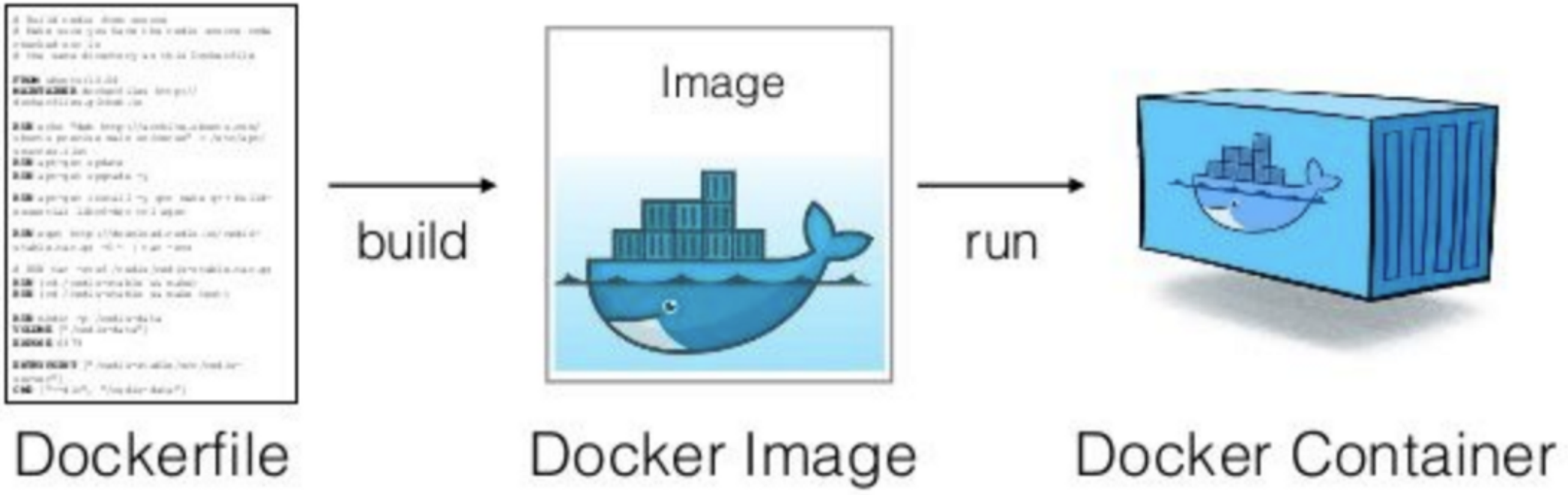


Image Source

BadgerChat Mini

Creating and deploying the image.

Backend Server Hosting

Not an endorsement of any particular service.

Other Considerations

- Use Jenkins or some other CICD platform to create a build and deploy pipeline.
  - Include testing as an automated step.
- Use HTTPS for a secure HTTP connection.
  - Consider LetsEncrypt.
- Buy a domain name?
  - Completely optional!

What did we learn today?

- What is the software stack?
- How can we develop a backend?
- How can we persist data?
- How can we use containerization?
- What are other considerations?

Questions?