React for Data Science

A series from **AI-5**:
Deep Learning, Development, and Operations (MLOps)

Shivas Jayaram

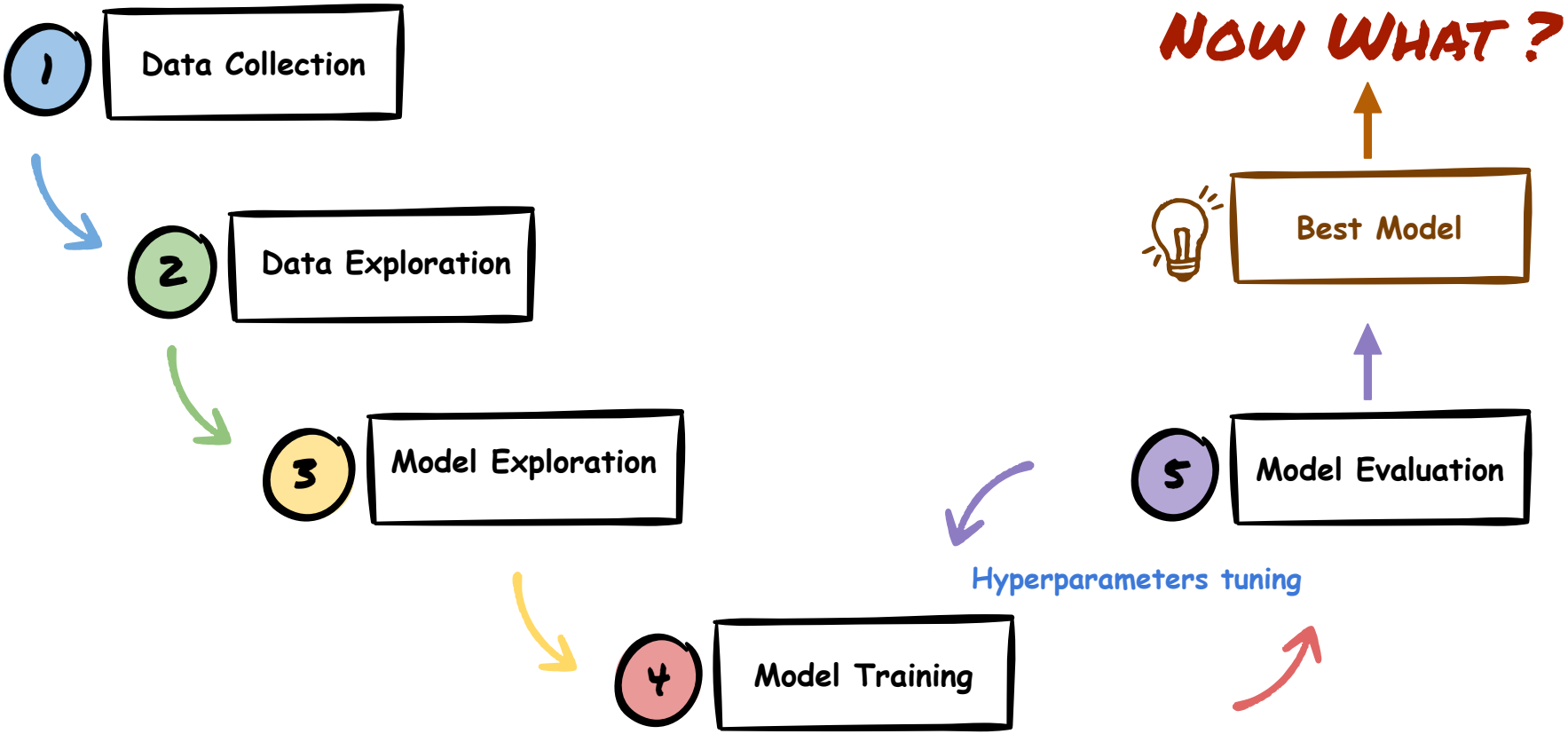
Outline

1. Motivation
2. App Frontend
3. Frontend Frameworks
4. React App

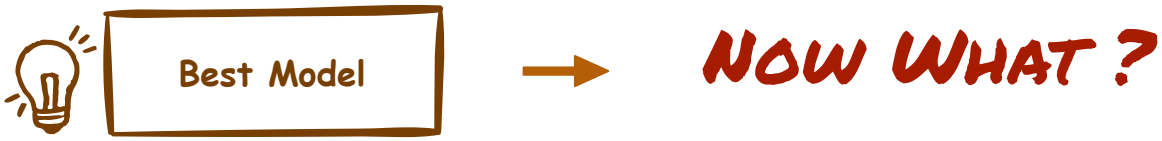
Outline

1. **Motivation**
2. App Frontend
3. Frontend Frameworks
4. React App

Motivation: Deep Learning Flow



Motivation: Best Model



Motivation: Your role

Researcher

- Focus on data and modeling
- Working on new SOTA model research
- Frontend could help to showcase your work

Data Scientist
(Enterprise)

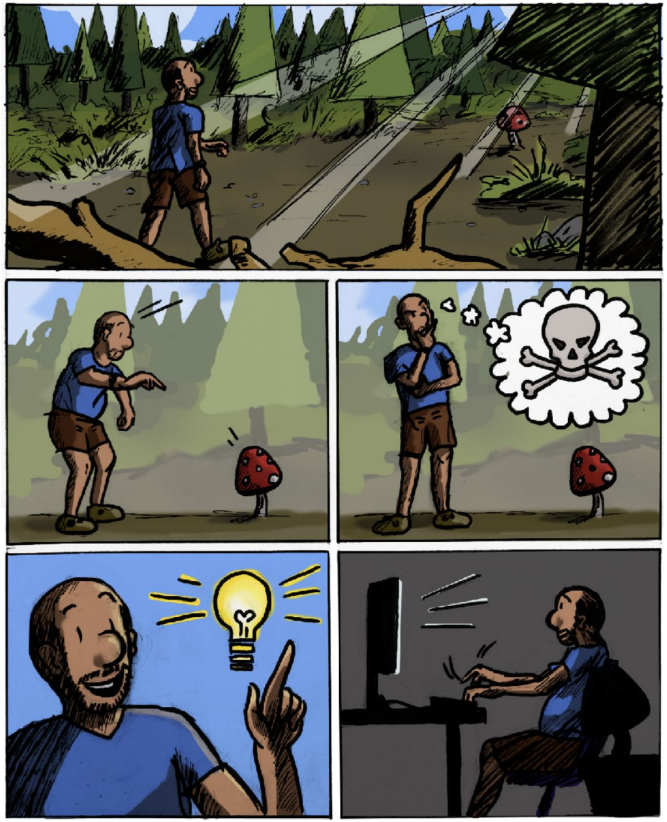
- Focus on data and modeling
- Familiar with model deployment
- Work with frontend team to build products using your model

Data Scientist
(Startup)

- Focus on data and modeling
- Working on product to showcase your idea
- Familiar with frontend and backend
- Familiar with Ops

Motivation: Example Project

- Pavlos likes to go for mushroom picking
- Some mushrooms can be poisonous
- Help build an app to identify mushroom type and if poisonous or not

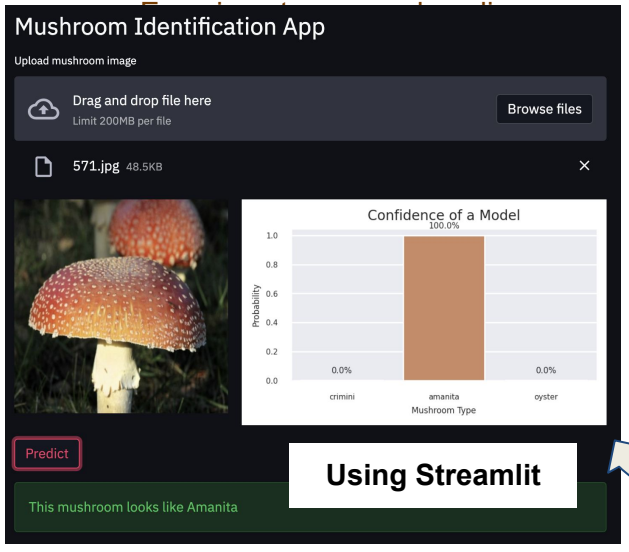


Credit: Nikolas Protopapas

Motivation: Example Project Scope

Proof Of Concept (POC)

- Scrap mushroom data
- Verify images



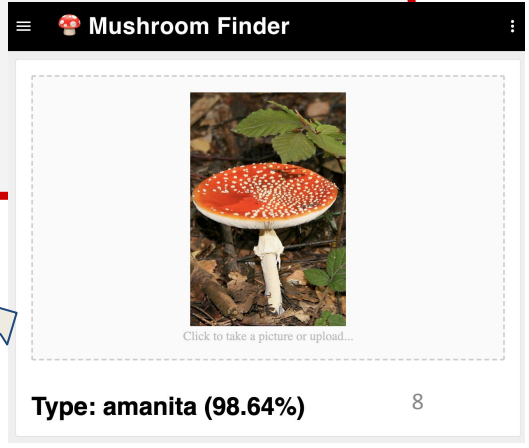
Using Streamlit

Prototype

- Create a mockup of screens to see how the app could look like
- Deploy one model to serve model predictions as an API

Minimum Viable Product (MVP)

- Create App to identify Mushrooms
- API Server for uploading images and predicting using best model



Motivation: How do we build an App?

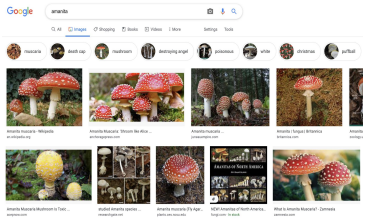
Data Collection

Data Exploration

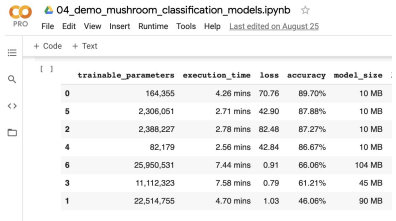
Model Exploration

Model Training

Model Evaluation



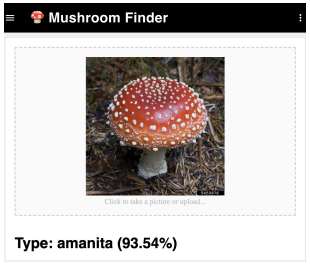
Python Script



	trainable_parameters	execution_time	loss	accuracy	model_size
0	164,355	4.26 mins	70.76	89.70%	10 MB
5	2,306,051	2.71 mins	42.90	87.88%	10 MB
2	2,388,227	2.78 mins	82.48	87.27%	10 MB
4	82,179	2.56 mins	42.84	86.67%	10 MB
6	25,950,531	7.44 mins	0.91	66.06%	104 MB
3	11,112,323	7.58 mins	0.79	61.21%	45 MB
1	22,514,755	4.70 mins	1.03	46.06%	90 MB

Colab

App Frontend



Rest API

Covered in AI-5

IDE / Code Editor

Outline

1. Motivation
2. **App Frontend**
3. Frontend Frameworks
4. React App

App Frontend

HTML

- Is Hyper Text Markup Language (Remember Markdowns)
- Browsers use HTML to display web pages

CSS

- Cascading style sheets
- Used to format & style web pages

Javascript

- Programming language understood by browser

App Frontend

Markdown

```
# 🍄 Mushroom Identifier

🍄 Welcome to the mushroom identification App!
```

3 lines (2 sloc) | 76 Bytes

🍄 Mushroom Identifier

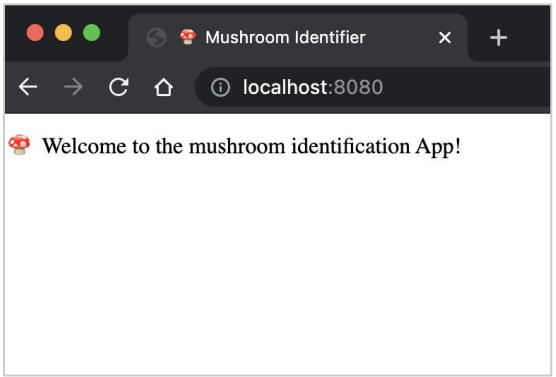
🍄 Welcome to the mushroom identification App!

Markup

```
<!DOCTYPE html>
<html>
<head>
    <title>🍄 Mushroom Identifier</title>
</head>
<body>
    🍄 Welcome to the mushroom identification
App!
</body>
</html>
```

Browser Title

Web page details



Frontend Vs Backend

Frontend

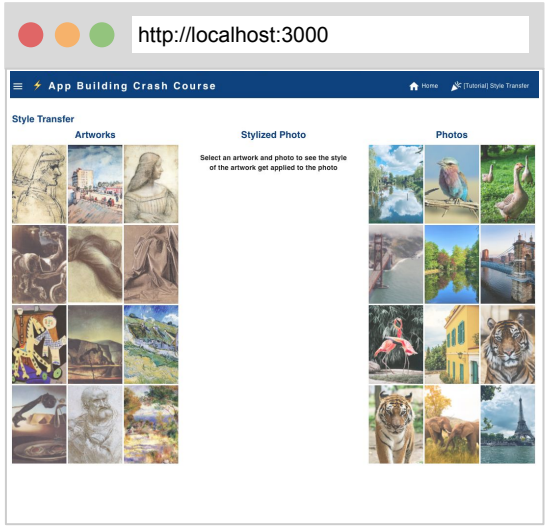
- Runs on user browser
- Written in HTML, Javascript, CSS
- Has no access to model and data
- Memory and processing constrained

Backend

- Runs on server
- Written in Python
- Has access to model and data
- Memory and processing not constrained

How does the App work

Frontend



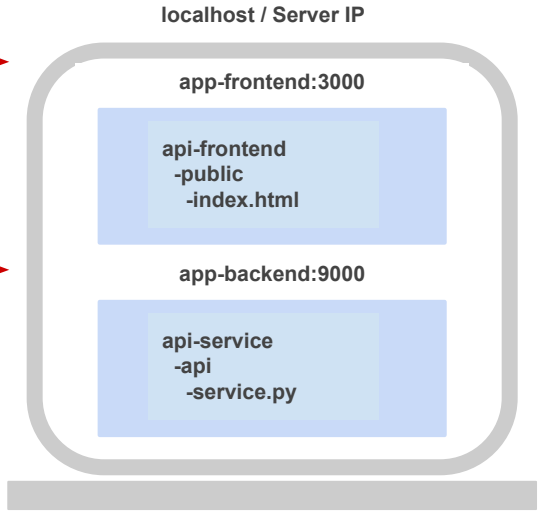
HTTP request to http://localhost:3000

/index.html will be sent back to browser

HTTP request to http://localhost:9000

Images are sent back to populate the artwork and photos grid

Backend

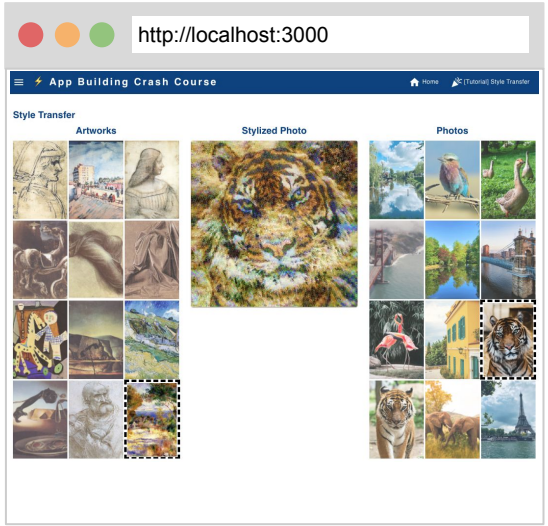


Browser

Local computer / Server

How does the App work

Frontend



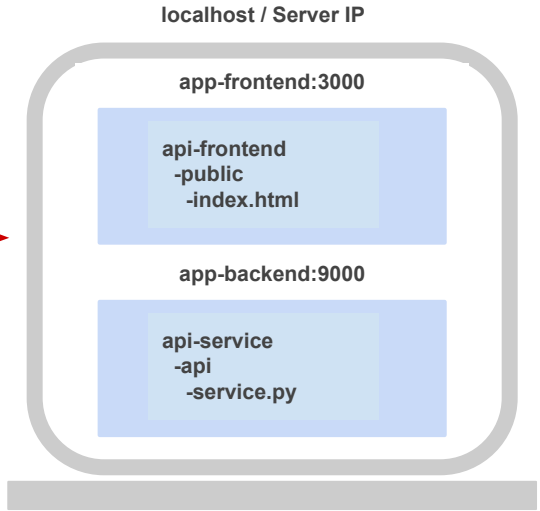
User selects a style image and a photo

Browser

HTTP request to http://localhost:9000
passing style image and photo

Backend uses the input style and uses a pre-trained model to apply style transfer and returns the “stylized” image

Backend



Local computer / Server

Frontend

If we build our frontend using just HTML and Javascript, we need a page for each component:

- index.html
- data.html
- predict.html

Frontend

If we build our frontend using just HTML and Javascript, we need a page for each component:

- index.html
- data.html
- predict.html

Problems:

- Each of these have its own HTML, Javascript, CSS
- How do we share/reuse code across pages?
- Each page is loaded separately in browser (Slow)

Frontend

Problems:

- Each of these had its own HTML, Javascript, CSS
- How do we share/reuse code across pages
- Each page is loaded separately in browser (Slow)

Solution:

- Create a single page app that manages HTML, Javascript, CSS as components
- Frontend App Frameworks to the rescue

Outline

1. Motivation
2. App Frontend
3. **Frontend Frameworks**
4. React App

Frontend Frameworks

The major frontend app frameworks are:

- Angular (Google)
- React (Facebook)
- Vue

React

- React has a rich set of pre-built components
- Build intuitive UIs and feature rich Dashboards easily
- Lot of app templates available

React

- Everything is a Component
- Uses JSX instead of Javascript
- JSX is an extension to JavaScript
- JSX is like a template language, but it comes with the full power of JavaScript

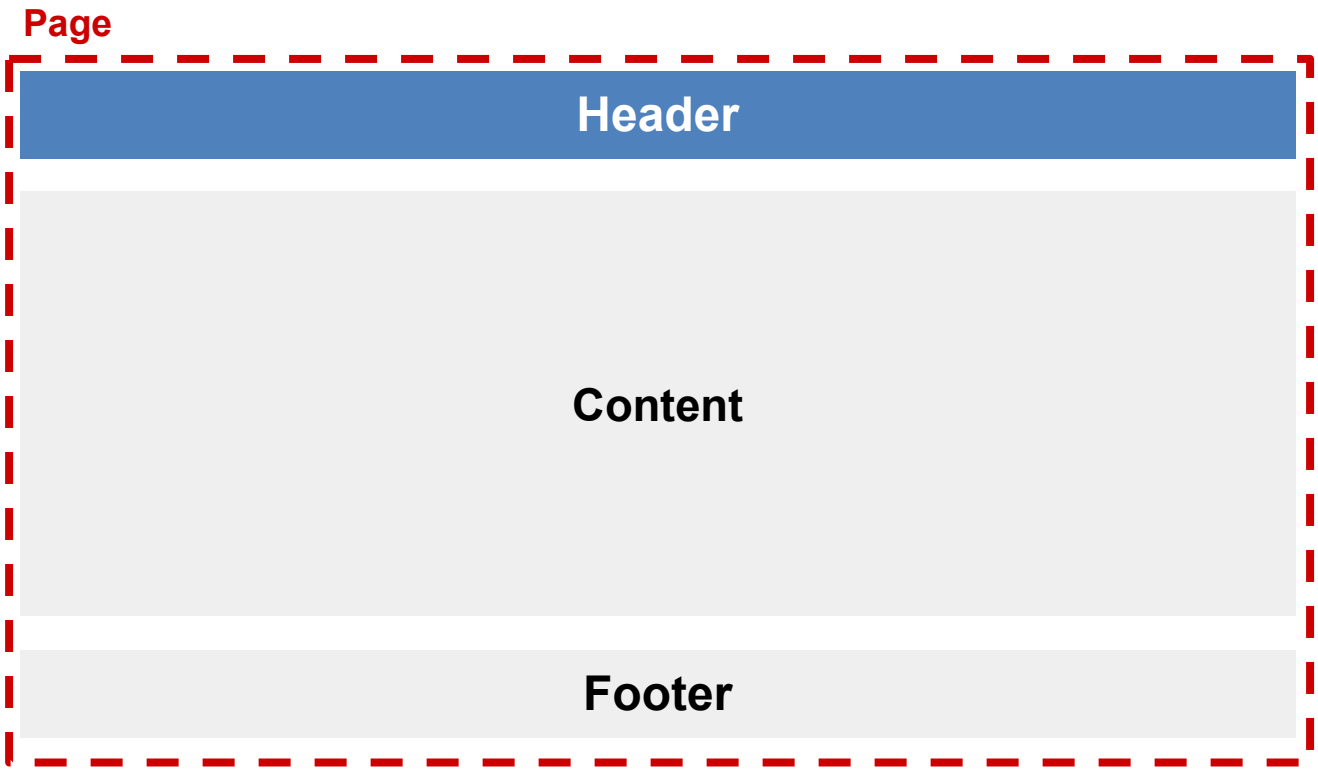
Outline

1. Motivation
2. App Frontend
3. Frontend Frameworks
4. **React App**

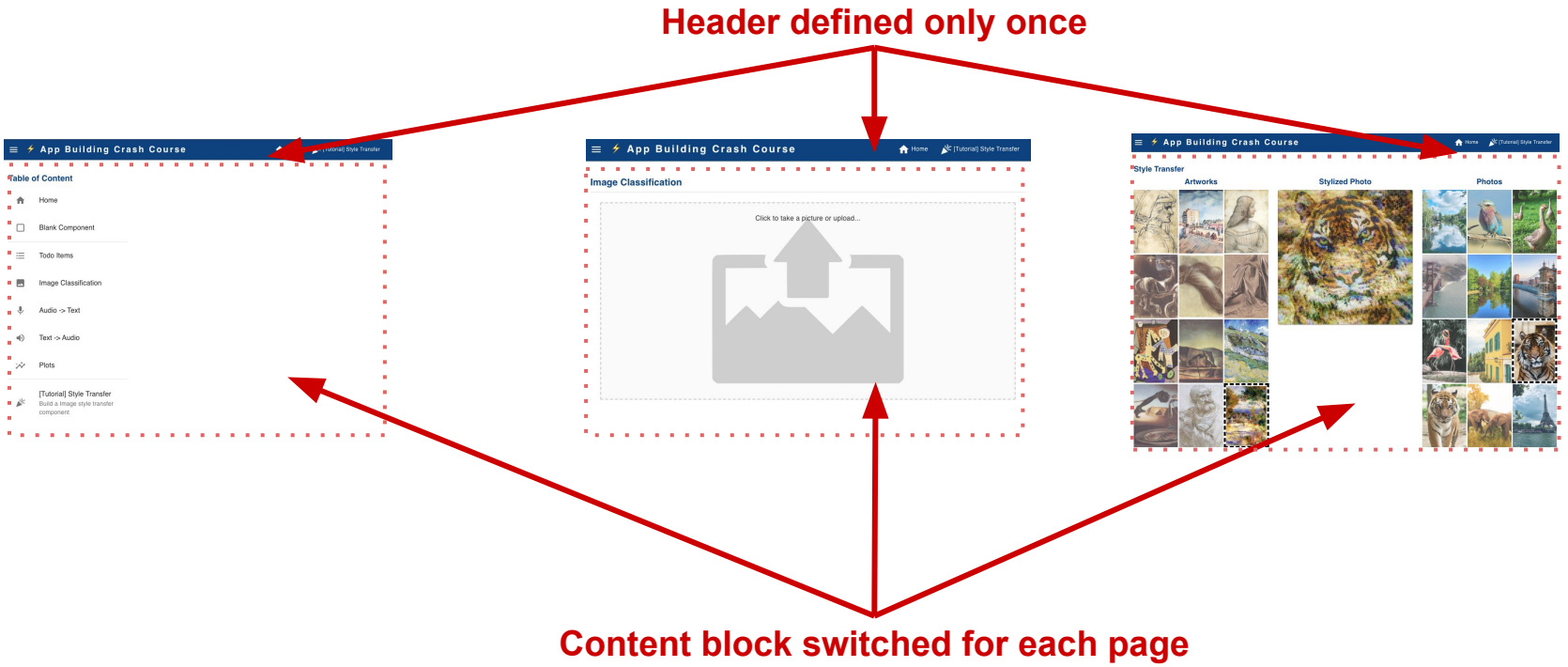
React App: Getting Started

- Intro to HTML: https://www.w3schools.com/html/html_intro.asp
- Intro to Javascript: https://www.w3schools.com/js/js_intro.asp
- Intro to React: https://medium.com/javascript-scene/the-missing-introduction-to-react-62837cb2fd76
- Getting started with React: https://reactjs.org/docs/getting-started.html

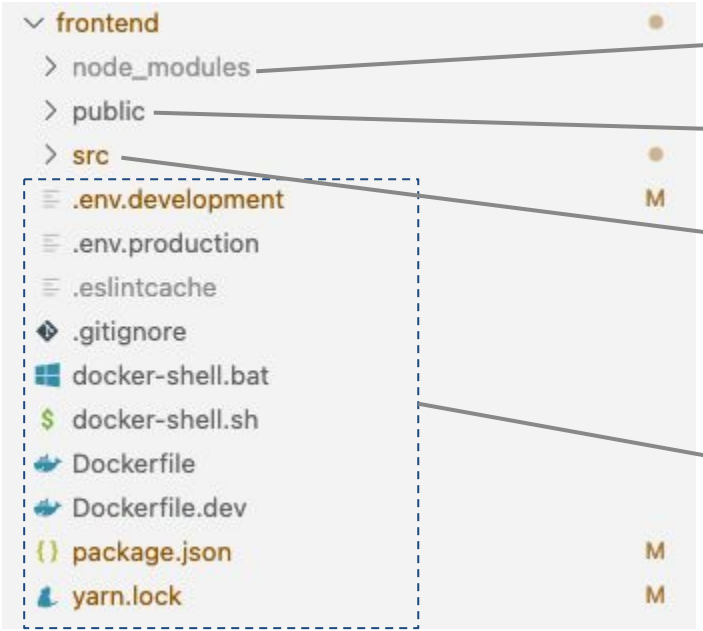
React App: Components



React App: Components



React App: Project Structure



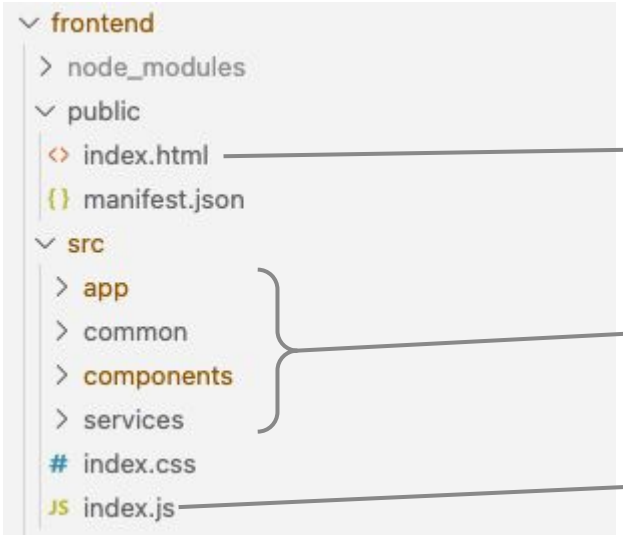
Installed packages

Entry point to serve the App

React Code

App operations files

React App: Project Structure




```
<!DOCTYPE html>
<html lang="en">
<head>…
</head>
<body>
  <noscript>You need to enable JavaScript
  <div id="root"></div>
</body>      Shivas Jayaram, 7 months ago
</html>
```

Initial page sent to browser which includes the React App

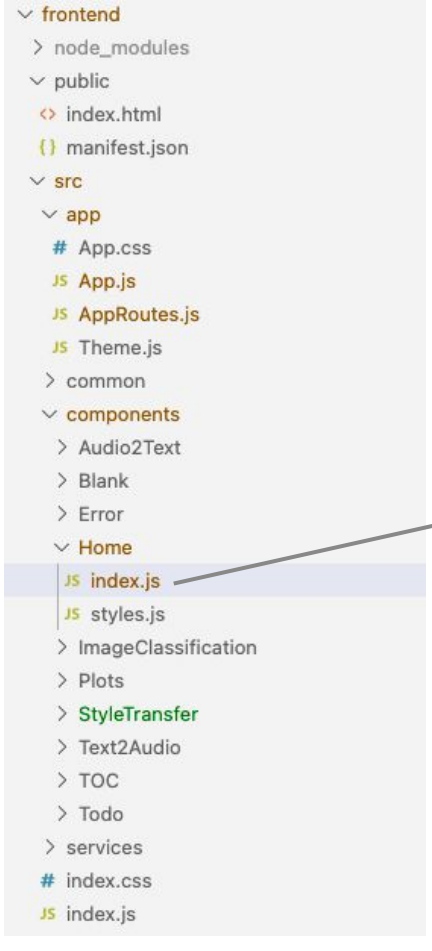
React Components

React App loader

```
import App from './app/App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

React App: Component




```jsx
import React, { useEffect, useRef, useState } from 'react';
import { withStyles } from '@material-ui/core';
import Container from '@material-ui/core/Container';
import Grid from '@material-ui/core/Grid';
import styles from './styles';
import TOC from '../TOC';

const Home = (props) => {
    const { classes } = props;

    // Component States

    // Setup Component

    // Handlers

    return (
        <div className={classes.root}>
            <main className={classes.main}>
                <Container maxWidth={false} className={classes.container}>
                    <Grid container spacing={3}>
                        <Grid item xs={4}>
                            <TOC />
                        </Grid>
                        <Grid item xs={4}></Grid>
                        <Grid item xs={4}></Grid>
                    </Grid>
                </Container>
            </main>
        </div>
    );
};

export default withStyles(styles)(Home);      Shivas Jayaram, 7 months ago • Components
```

React App: Project Structure


```
const styles = theme => ({
    root: {
        flexGrow: 1,
        minHeight: "100vh"
    },
    grow: {
        flexGrow: 1,
    },
    main: {

    },
    container: {
        backgroundColor: "#ffffff",
        paddingTop: "30px",
        paddingBottom: "20px",
    },
});

export default styles;      Shivas Jayaram, 7 months ago • Components
```

React App: Code Tutorial

- GitHub Repo: https://github.com/dlops-io/app-building-crashcourse
-

THANK YOU