Computing

IT 5007: Software Engineering on Application Architecture

Lecture-5

Prasanna Karthik Vairam
Lecturer
Department of Computer Science
NUS School of Computing

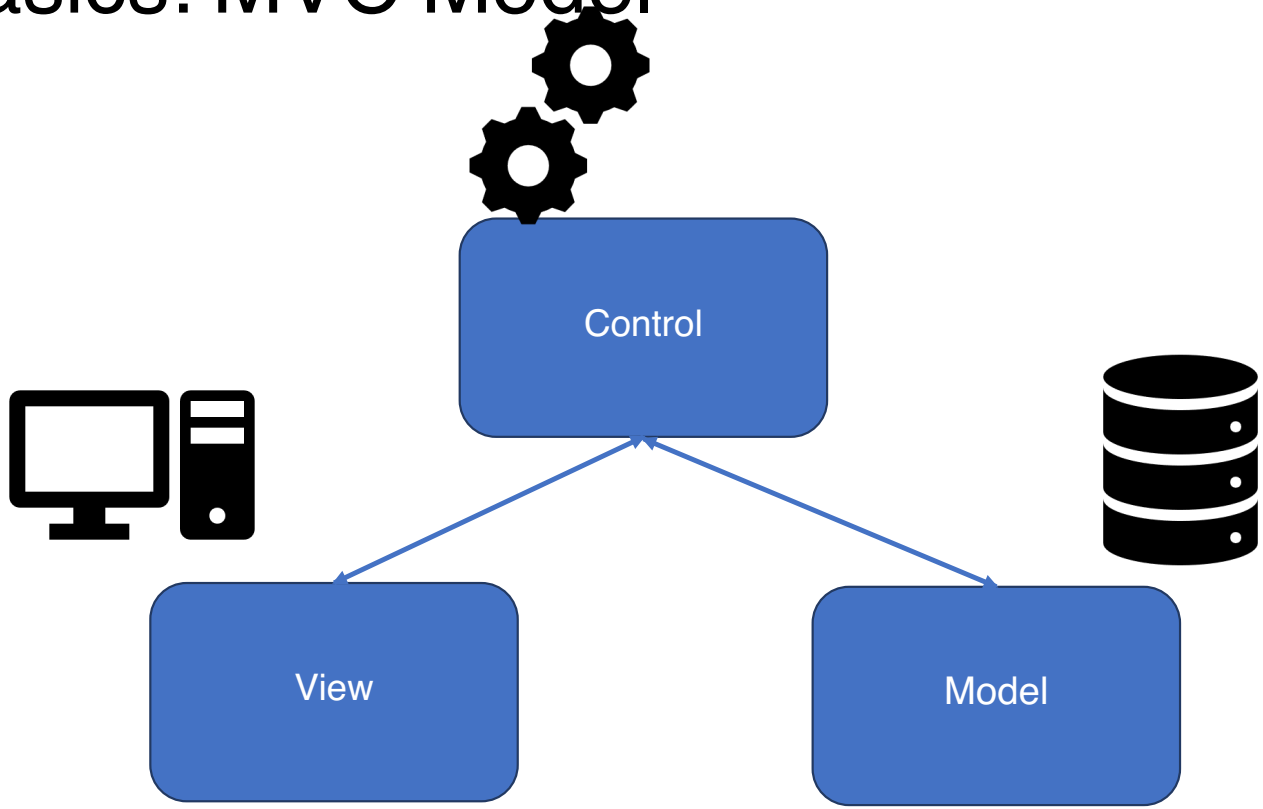
Computing

React: Motivation

What exactly is React?

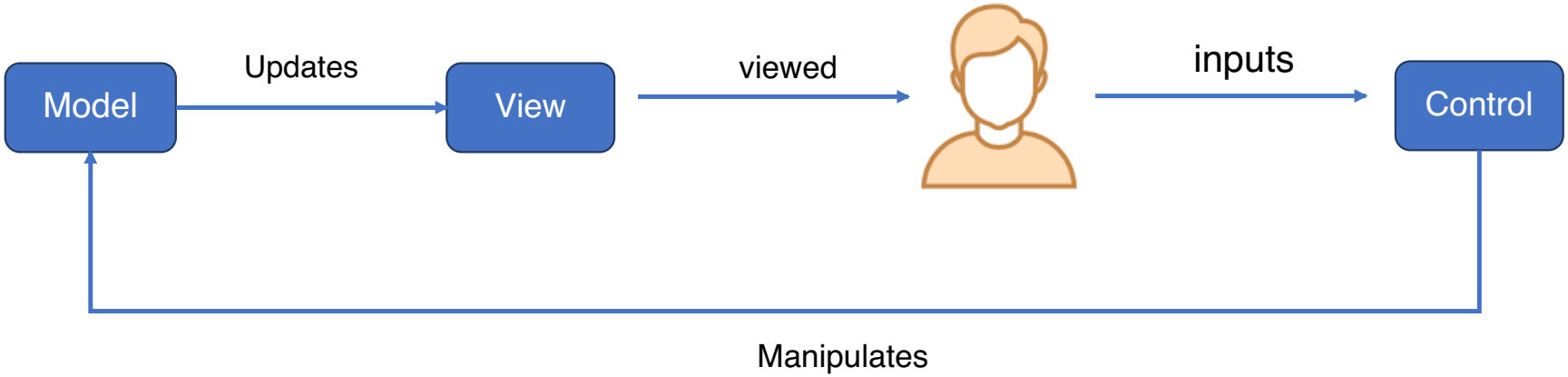
- React may sound like a lot of things.
- React is a high level language that generates JS code.
- It is just a JS library that you include to your HTML file
- **Digression**: What is a Library?
- **Why is it popular?:** React Provides numerous off-the-shelf libraries (components) for you to use. (DOM query, charting, math, etc.)
- **Importantly**: React allows you to create your own custom components from these off-the-shelf components.
- **More importantly**: React defines some nice rules that make coding and debugging easier.
- **Most importantly**: It binds the View and Model to boost performance.
- What is the intuition? Primevideo example.

Basics: MVC Model



Isn’t this what JS does already?

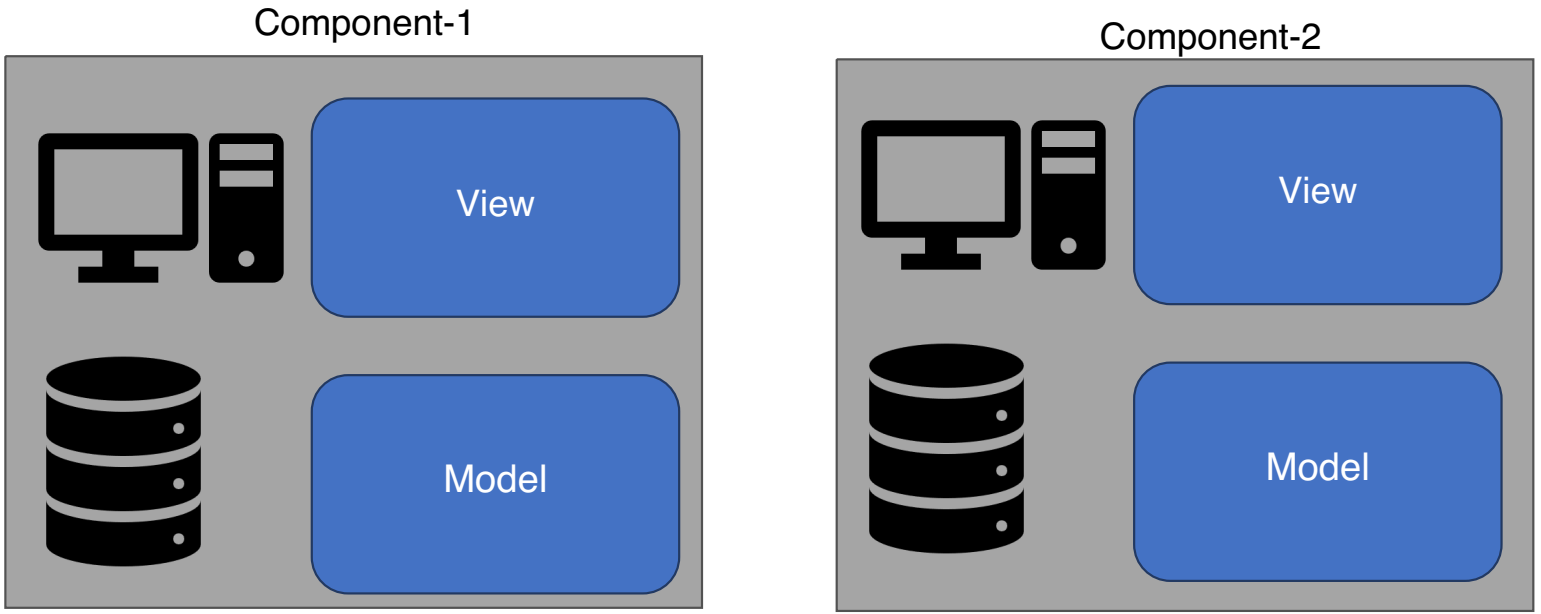
Javascript: The FB Ad story



Curious case of cascading updates!

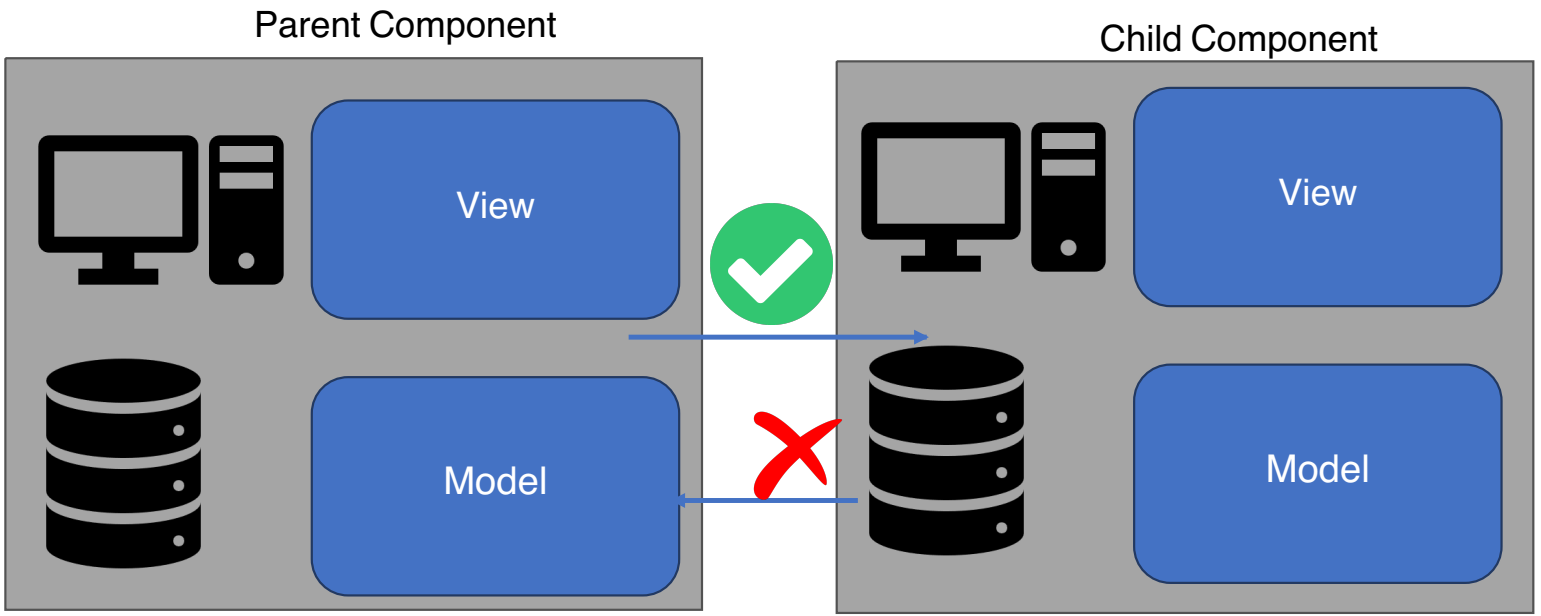
React: Features

Motivation for React: Component based



Components are self-contained and take care of themselves.

Motivation for React: Data flow Rules



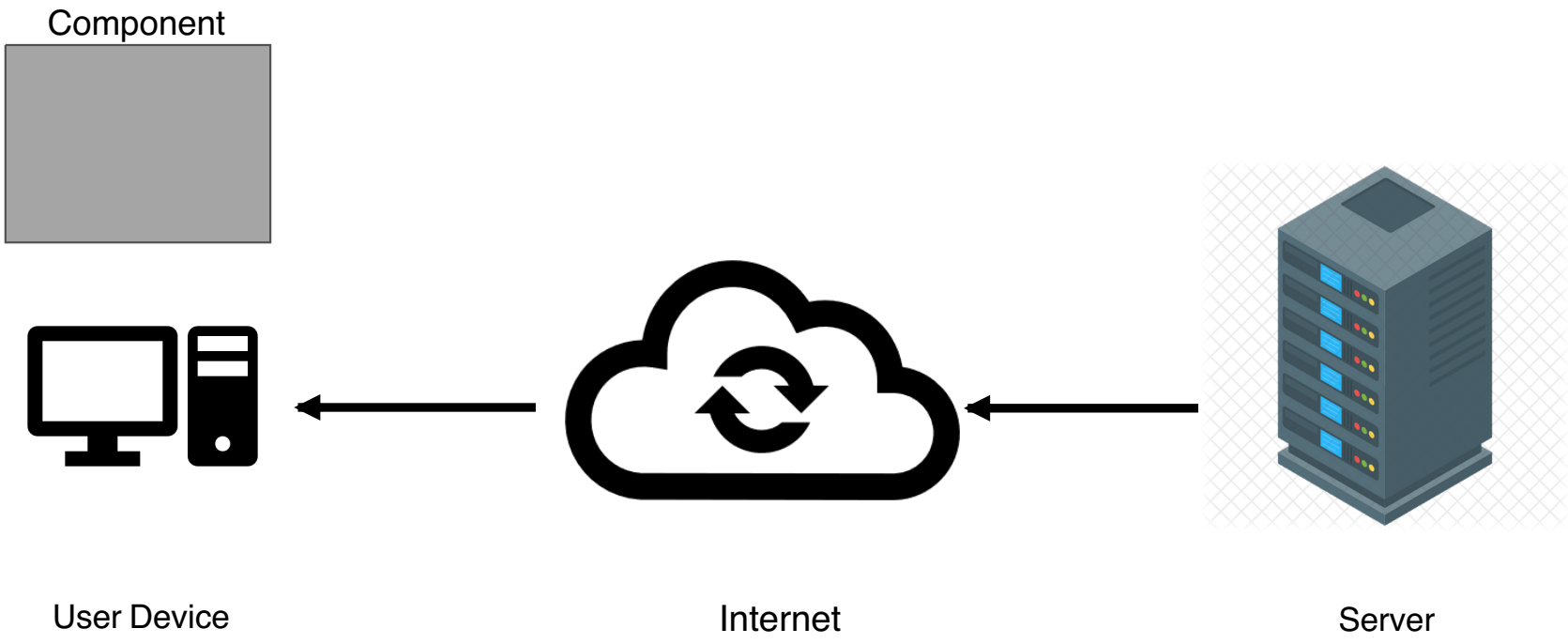
Unlike JS, we do not allow data to flow randomly!

Motivation for React: Data flow Rules

That’s like how most developers handle data in JS!

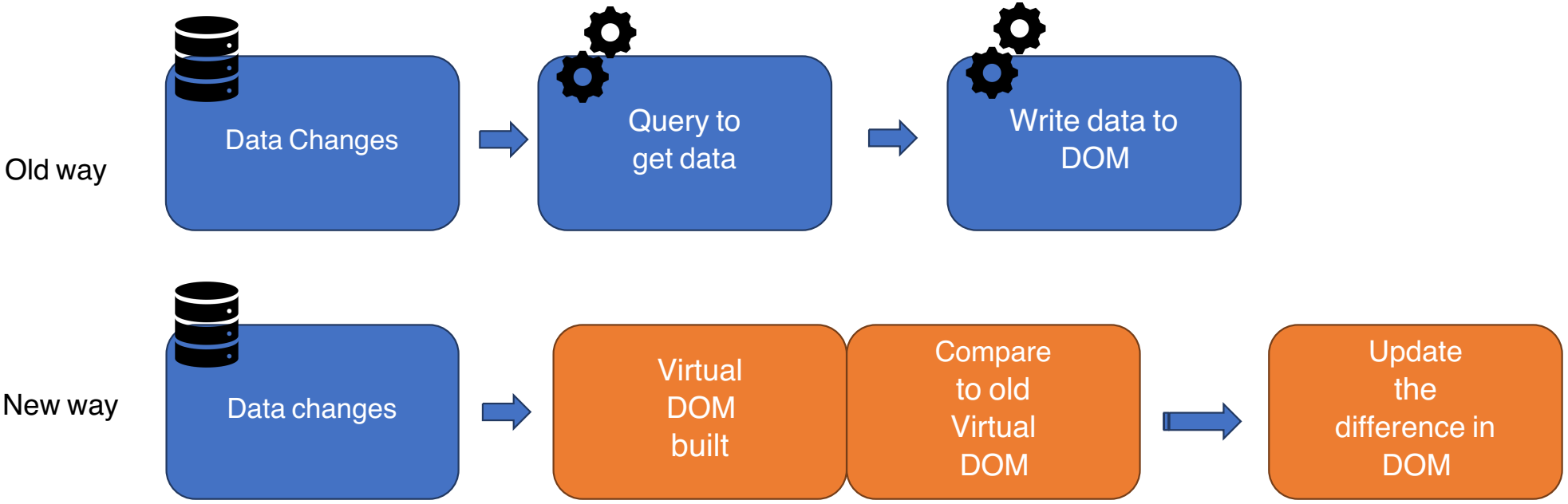
Motivation for React: Isomorphic

Write once – decide where to host later!



Motivation for React: Efficiency

Assume you are creating a <table> from the data.



Can you guess what happens in Facebook?

Computing

React/JS: JS Class Basics and Class Components

Class Definition

- Class is a **generic template** for creating many objects/instances.
- E.g., a Student class can be used to create 100 real student objects/instances.
- Class is Student and s1/s2 are **instances**.
- **Constructor** is called when an instance is created and its objective is to initialize the object.
- Class can have **variables** and **functions**.
- *this.name* refers to the current objects name.
  - E.g., when s1 is created, this.name refers to s1's name.

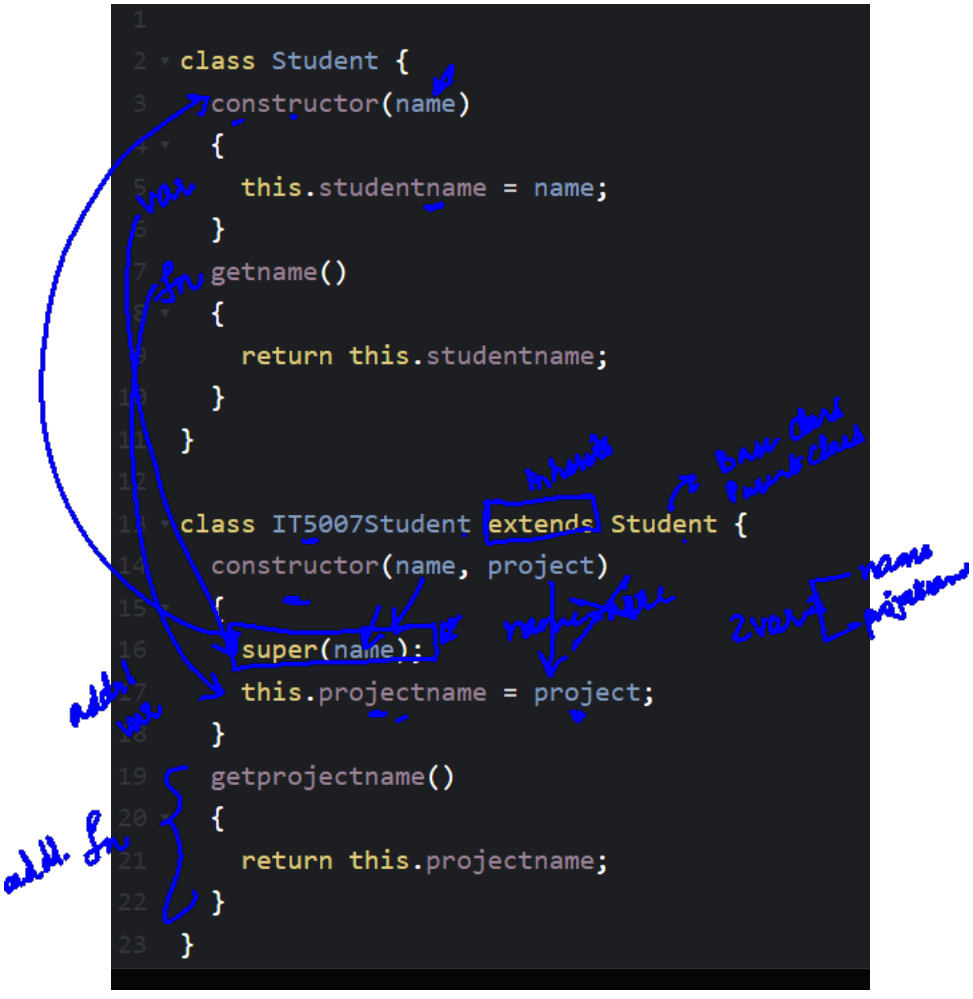
```
class Student {
  constructor(x_name, x_matricno)
  {
    this.name = x_name;
    this.matricno = x_matricno;
  }
  getname()
  {
    return this.name;
  }
  getmatricnumber()
  {
    return this.matricno;
  }
}

const s1 = new Student("Prasanna", 1);
const s2 = new Student("Karthik", 2);
```

CodePen – Example 1

- Example1: https://codepen.io/pkarthik88/pen/oNdjoNm
- Exercise:
  - Add one more field (NUSNETID) to student class.
  - Add get function for new field.
  - Update the instance creation code (s1 and s2).

Inheritance

- Canvas requires some details of students – Student class
- IT5007 may require “additional” information from students. – IT5007Student class.
- The idea is to **reuse** the variables/functions in Student class and add more variables/functions.
- We use *extends* to inherit from a base/parent class.
- *Super* is used to call the base class constructor. Why?



CodePen – Example 2

- Example2: https://codepen.io/pkarthik88/pen/LYmpKQo
- Write code to extend IT5007Student class to create a new class called IT5007StudentUnofficial, where you have an additional field called WeChat_ID.

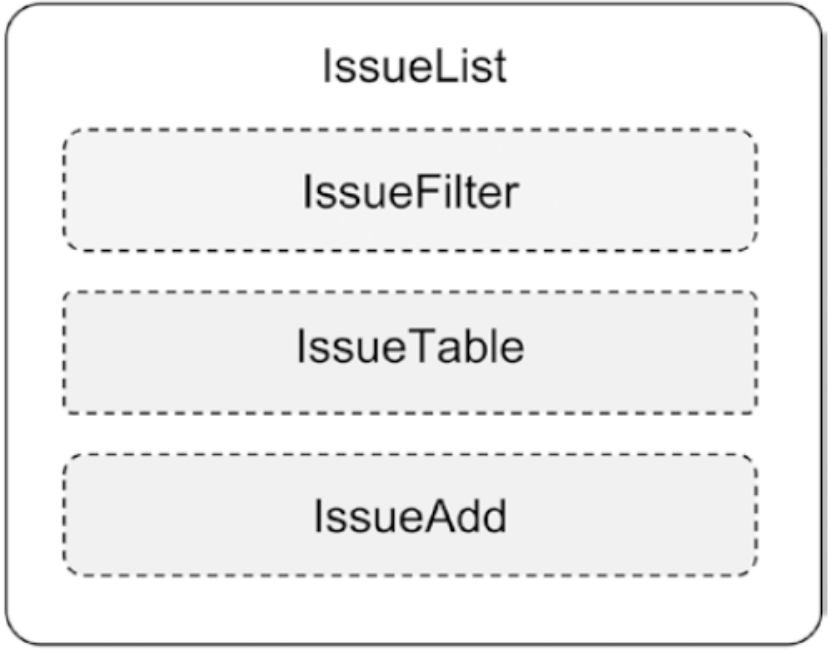
Computing

MVC: View (Display)

Realizing views in React

- We don't write HTML code for view.
- React code is written in JS or JSX (JavascriptXML) files.
- React components should ***render*** JS code which in-turn ***renders*** HTML code.
- **React package** responsible for generating HTML code is the ReactDOM.
- The **react function** responsible is ReactDOM.render().

Realizing views in React



React Class Components

- **Built-in** class components. (e.g., <h1>,)
- **Custom class** components built from many built-in classes. (e.g., <IssueList>, <HelloWorld>)
- Custom react classes always *extend React.Component*.
- React.Component gives essential variables and functions that we will use later (Props, Lifecycle Methods)!
- How to componentize?
  - Each component has a single idea.
  - How to know if your component has a single idea?
    - If you are passing too many data items to a class.

```
class IssueFilter extends React.Component {
  render() {
    return (
      <div>This is a placeholder for the issue filter.</div>
    );
  }
}
```

```
class IssueTable extends React.Component {
  render() {
    return (
      <div>This is a placeholder for a table of issues.</div>
    );
  }
}
```

Codepen Example 3

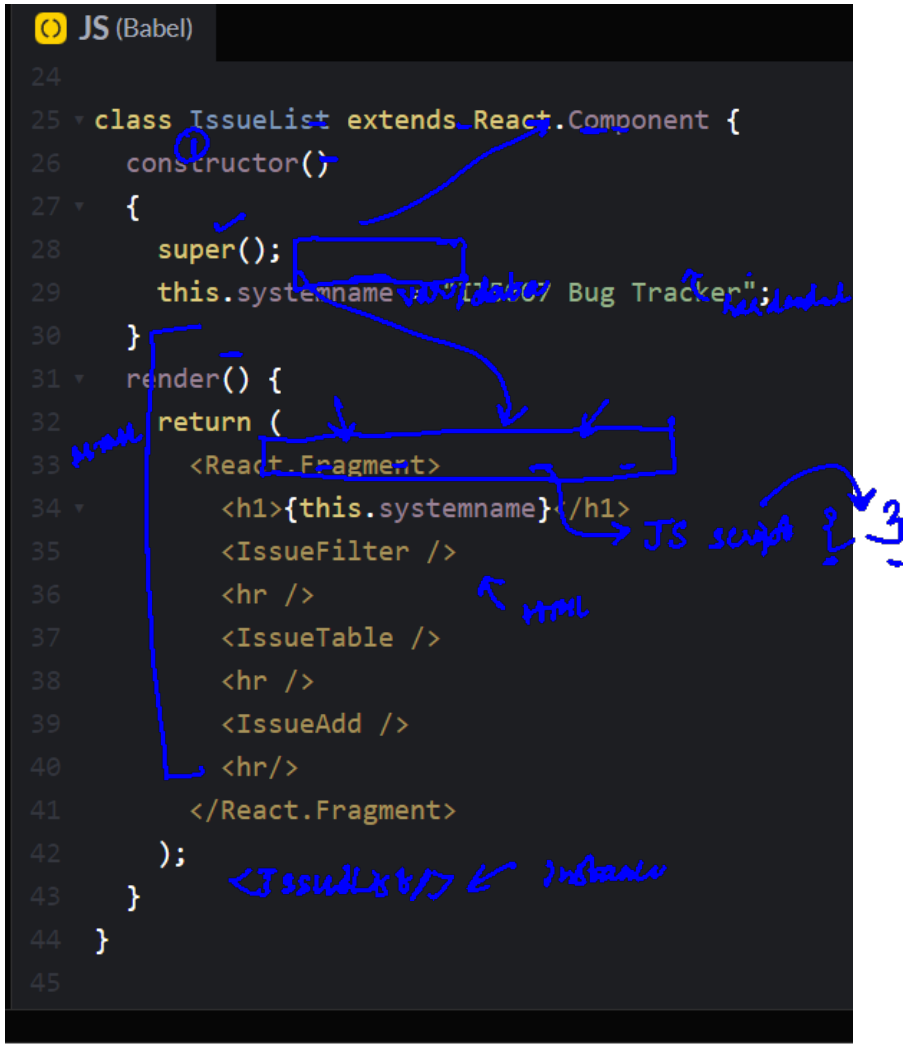
- https://codepen.io/pkarthik88/pen/oNdjrEr
- Change <React.Fragment> to and see if there is any change to the view.
- Name the components that only use build-in react components. Which built-in component does it use?

Computing

MVC: Model (Data)

Model-View Integration

- Class Component has
  - render() function that handles view.
  - Constructor() that handles data.
  - **Integration**: We can use JS/React data variables in the view code.



Codepen Example 4

- https://codepen.io/pkarthik88/pen/OJZyeem
- Create one more class variable called ‘footer’ and use it to add a copyright statement to bottom of the View.

Building Data models in React

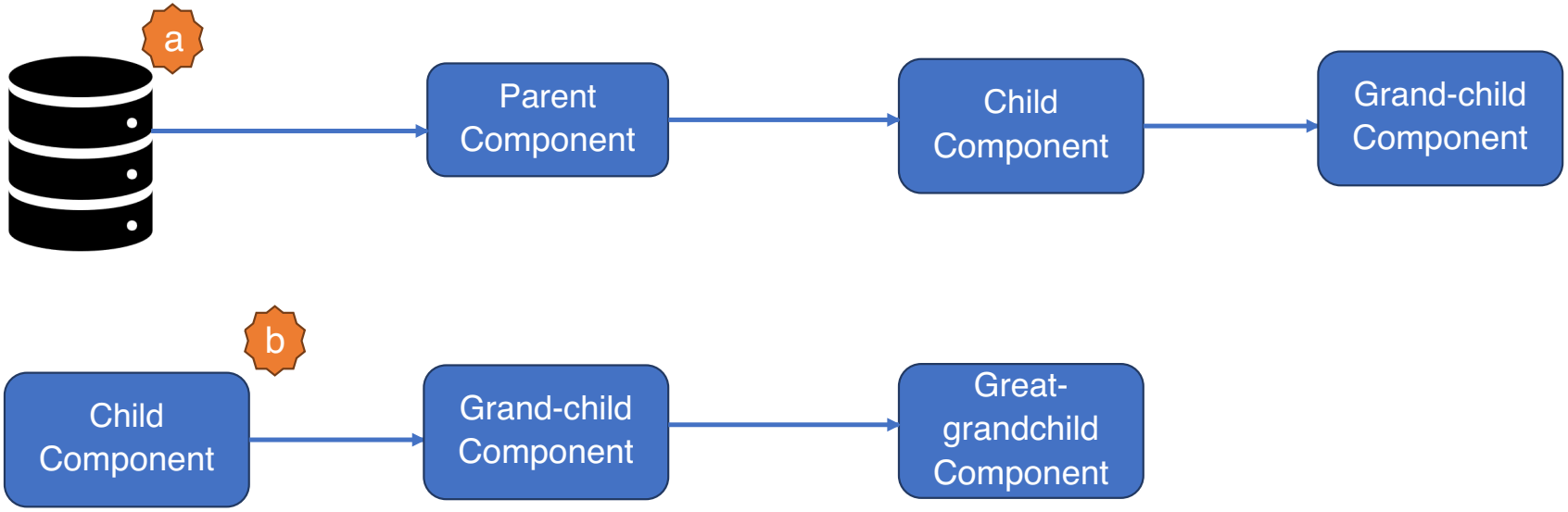
- **Data model** is different from database.
- DB will be introduced in Lecture 7. For now, let us use simple data-structures that resemble a database (e.g., an array of items).
- Data model refers to the structure/shape of data, how they relate to a real-world system, and how they relate to one other.
  - IssueList is an array of issues.
  - IssueList has a real-world application called bug tracker.
  - IssueList is related to EmployeeList the same way as real-world issues/bugs are related to employees.

3 Types of Data in React (Any web/mobile platform)

- **Domain data:** The dataset or static data that your application works with.
  - For example, Netflix works with a list of movies.
- **(optional) App State:** The current state of the App at this instant of time. As time goes on, the app state will change.
  - For example, Netflix maintains a list of seen movies, movies that are added as favorites for each user profile.
- **(optional) UI State:** The current state of the Netflix UI. As time goes on, the UI state will change.
  - For example, Has the user scrolled to the next page?

MVC: Model (Data) for Domain Data

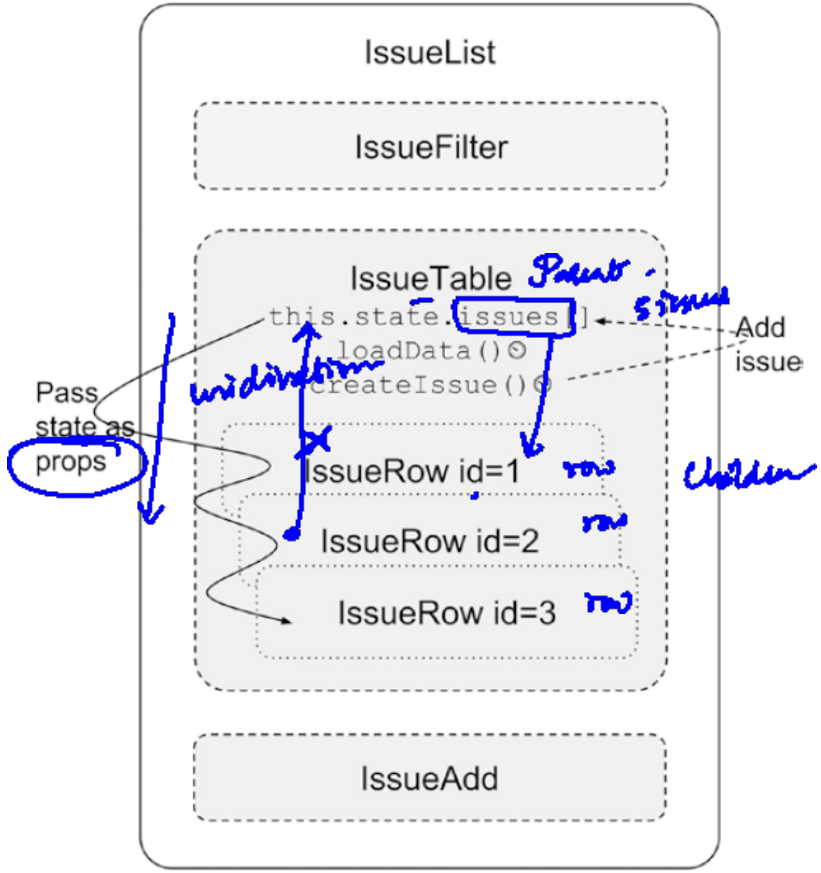
Data Flow rules



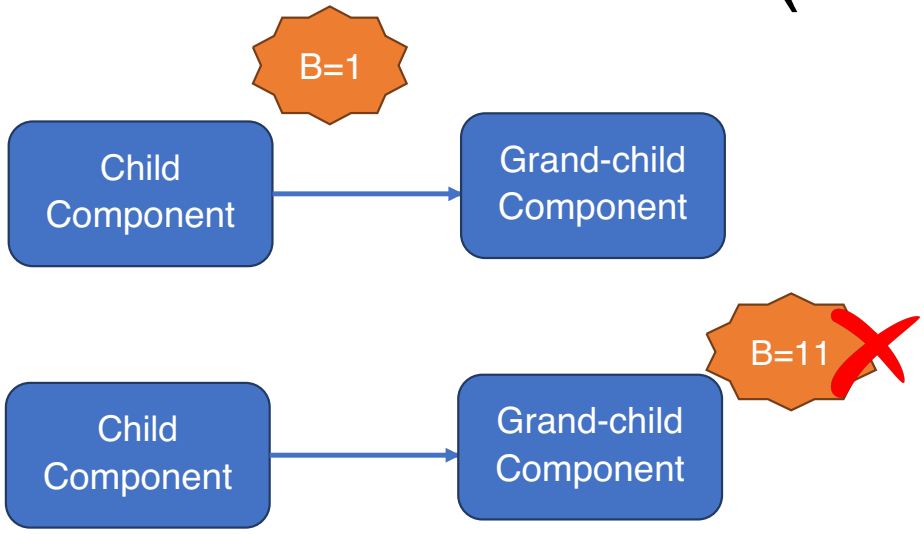
- Data flow is unidirectional
- Data cannot flow from child to parent.

Example: IssueTracker Application

- Data is passed down from IssueTable to IssueRow.
- Idea is to draw/render one row at a time.
- **New term:** props

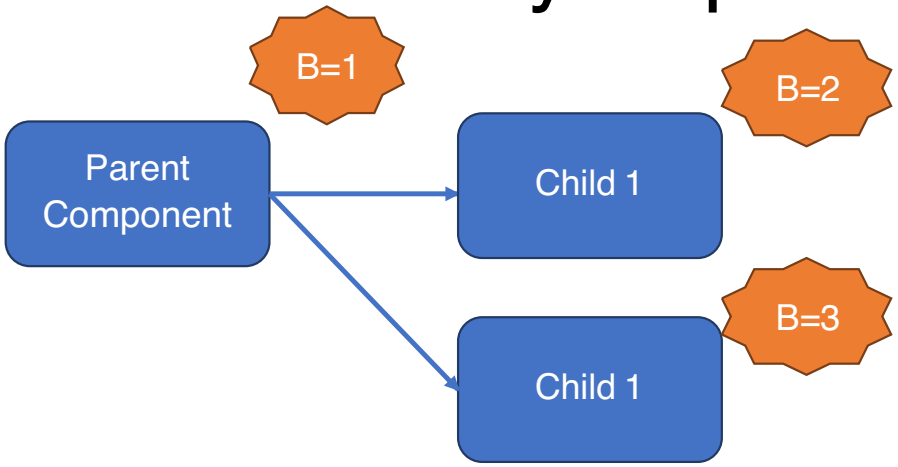


Data Modification Rules (Mutation)



- Data cannot be modified in child component.
- Why? Isn’t this a huge restriction?

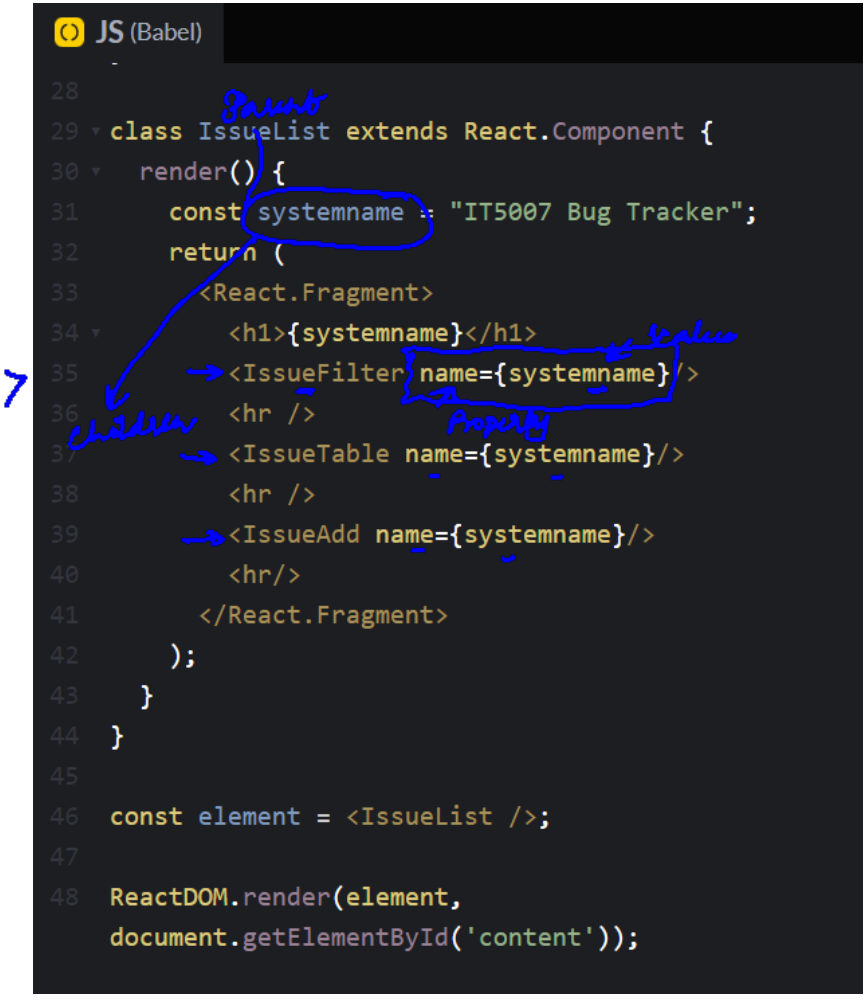
Why is immutability required?



Bottomline: This will lead to inconsistencies in data values.

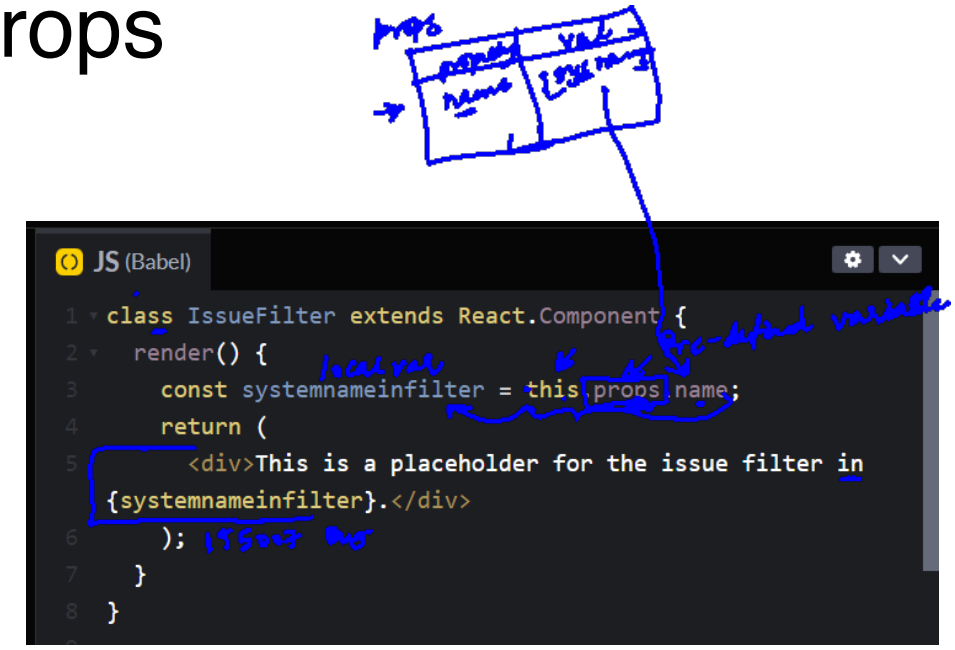
Passing Domain data: props

- The idea is to pass some data from a parent to a child.
- Props: property attribute.
- **Parent class** passes the variable and its value to the child.
- Example: IssueList is the parent trying to pass *name={systemname}* to its children.



Passing Domain data: props

- **Child class** uses **props** to access it.
- At the child, e.g., IssueFilter
- Variable is accessed as: *this.props.name*



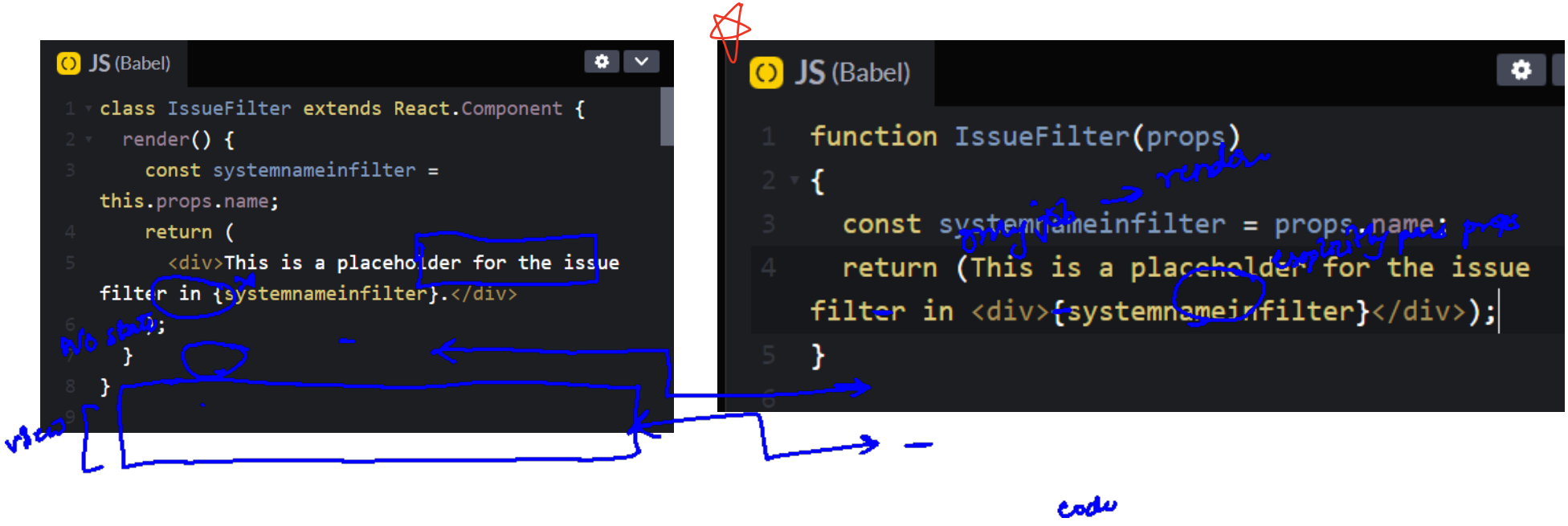
Codepen Example 5

- https://codepen.io/pkarthik88/pen/PoePMZj

React: Function Components (Efficiency)

Function Components for Efficiency

- Industry standard for implementing components.



CodePen Example 6

- https://codepen.io/pkarthik88/pen/YzjMajX
- Convert the footer class component into a function component.

Computing

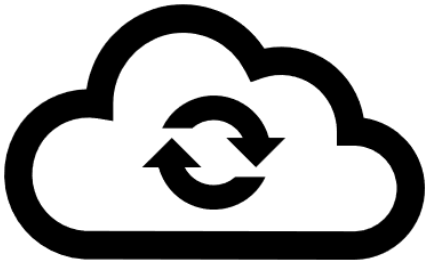
The Setup

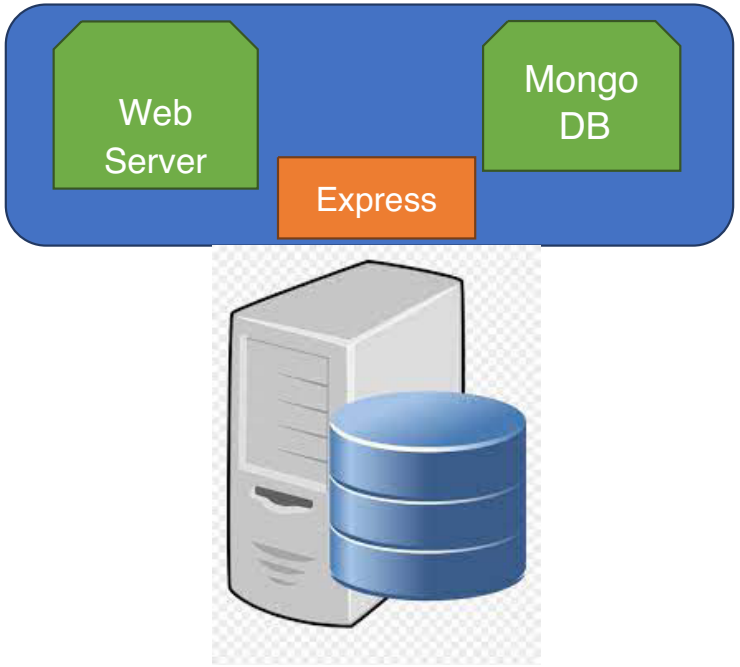
Developing without Servers

- Internet, Server, Client: Are we rich enough to have a server for ourselves?
- Instead, Try to **emulate** the server environment within your laptop.
- Our laptop serves as both server and client
- Virtualization – VM vs. Containers. We will be using **containers**.
- **Server**: Docker Container (runs as an application)
- **Client**: Web browser (runs as an application)
- Why not just code it like Tutorial 2? Have all files in one folder and double click on HTML file!

Real World Applications

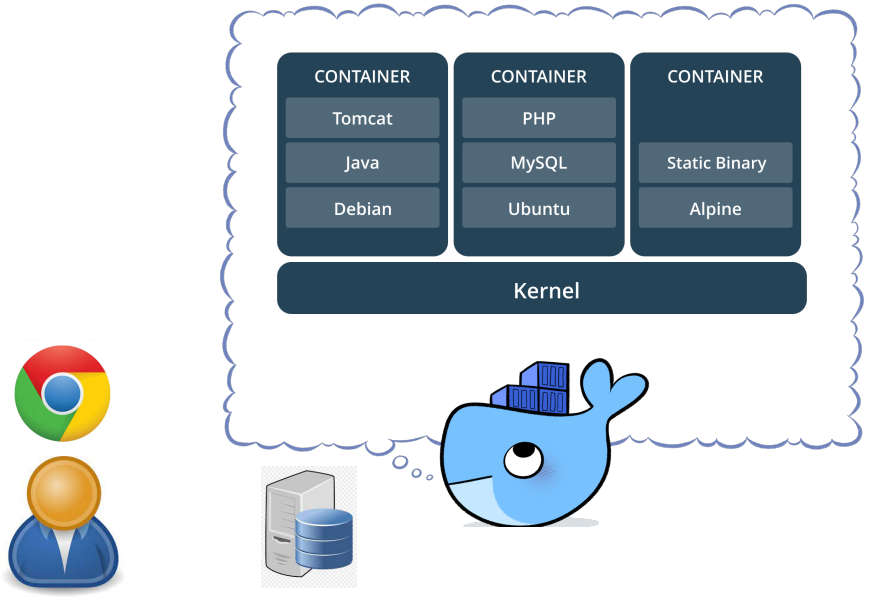
Client

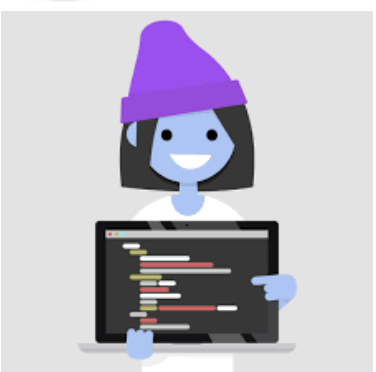




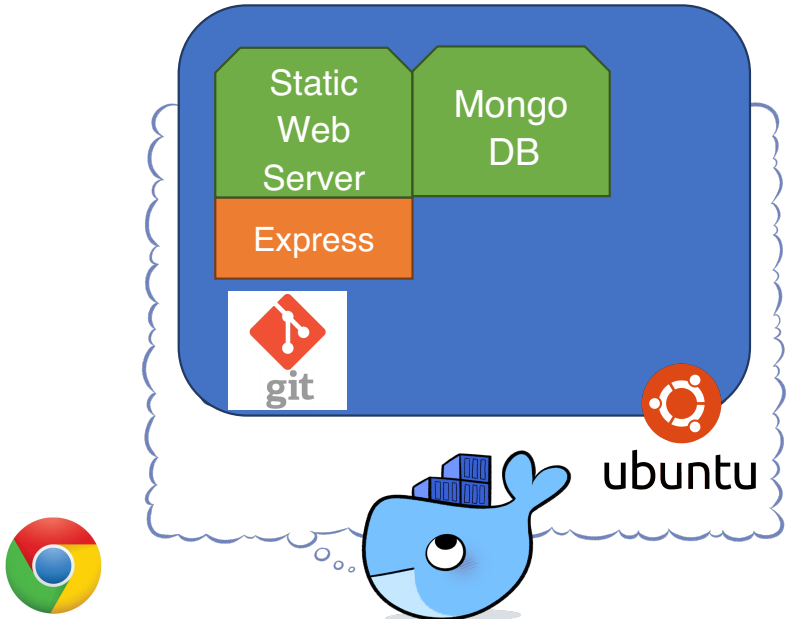
Server

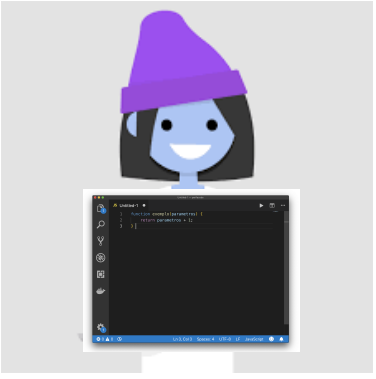
Developer’s Setup





Developer’s Setup





Setup Walkthrough Session

Follow instructions in
https://github.com/pkarthik88/IT5007-Lecture5.git