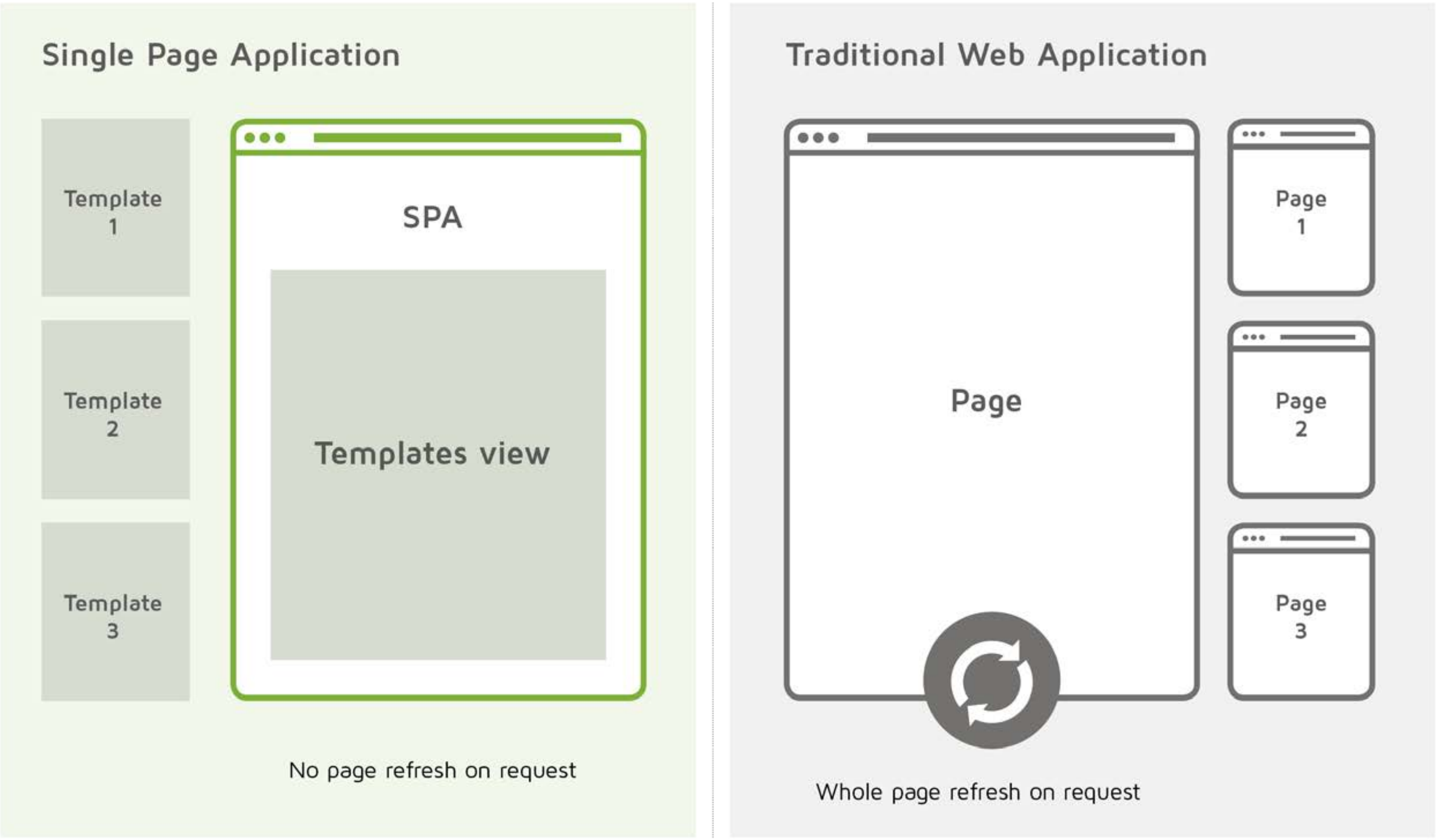
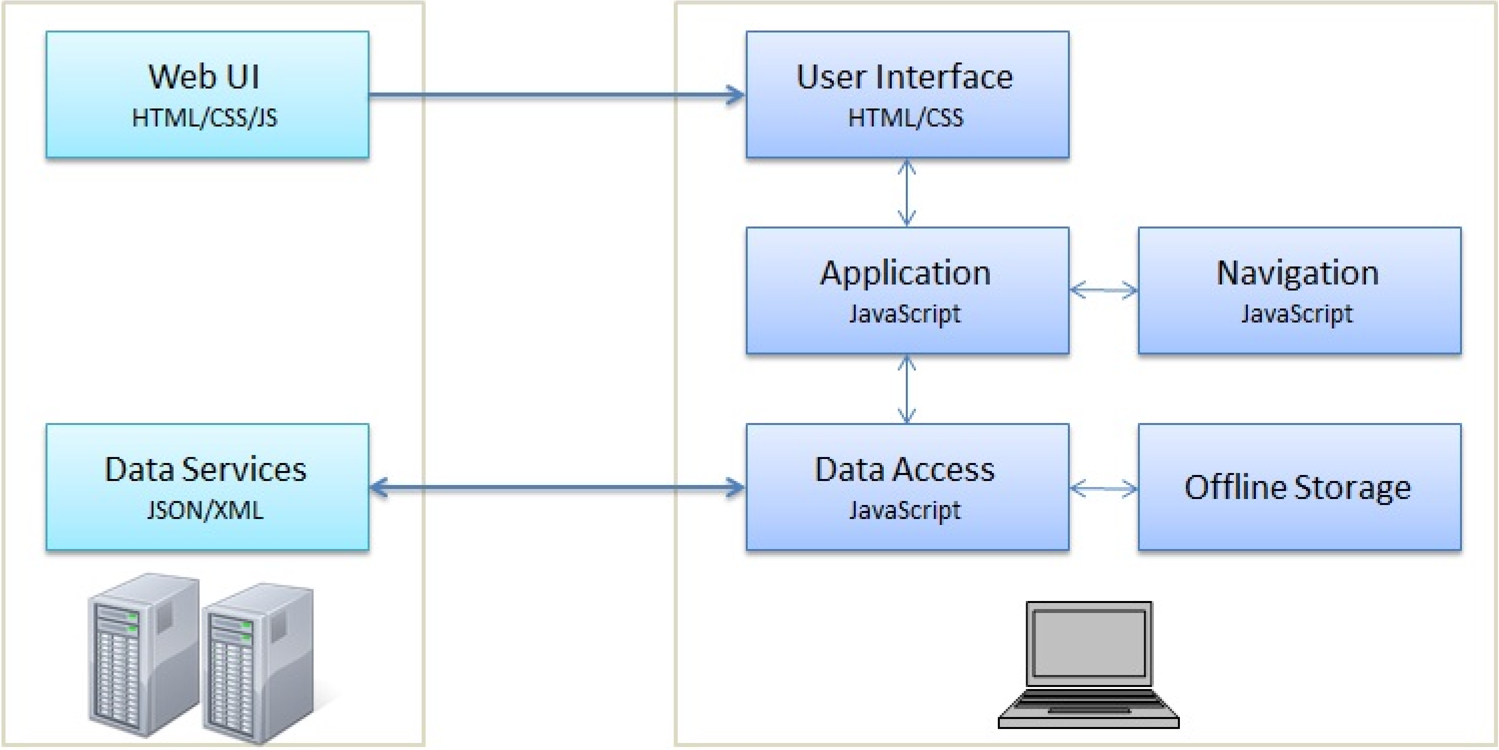
JS/Angular

CS252

Why frameworks?



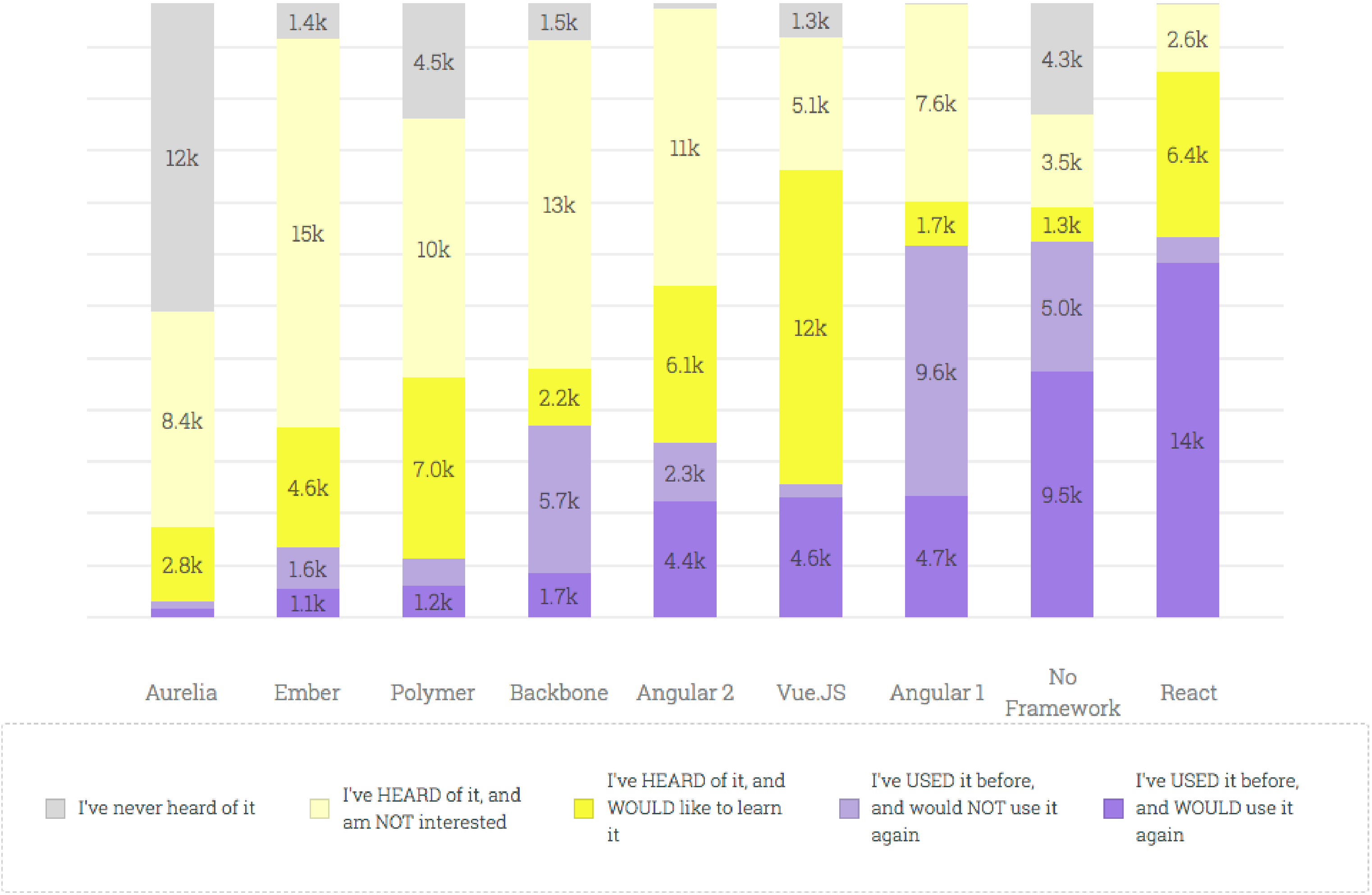
Complex non-linear navigation



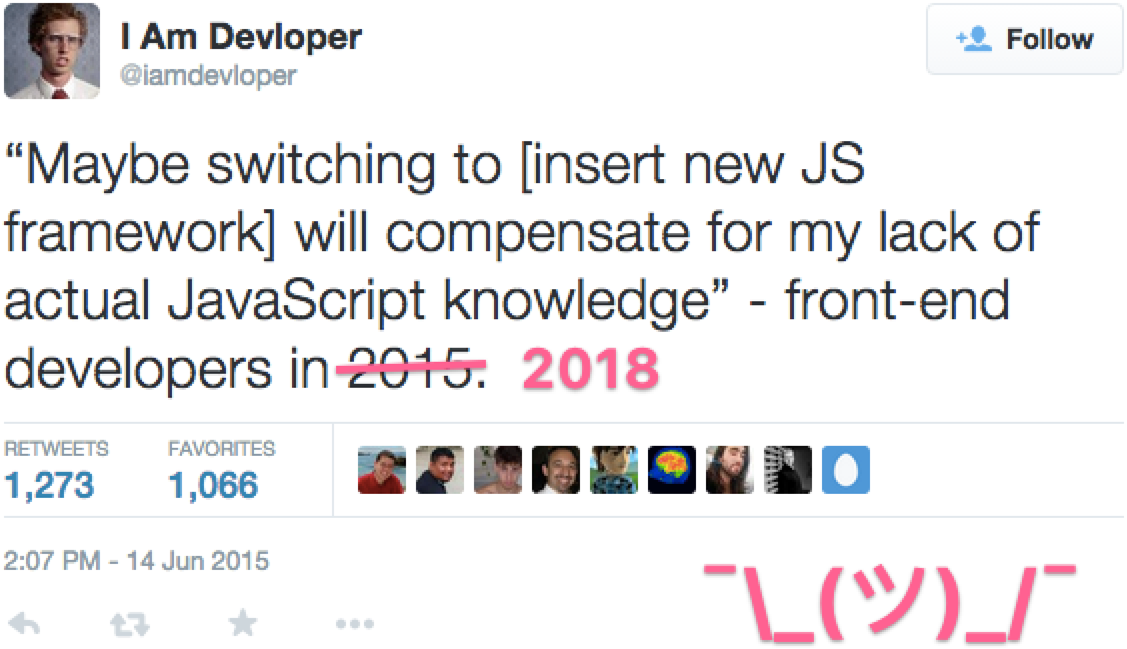
Server architectures

- Thin server
  - All data processing shifted to the client
- Thick stateful server
  - Server processes data and sends desired changes to client
  - Server maintains a record of state of client page
- Thick stateless server
  - Server processes data and sends desired changes to client
  - Server doesn’t maintain a record of the client page

Which framework to use?



Best answer is often *none*



I Am Devloper
@iamdevloper

"Maybe switching to [insert new JS framework] will compensate for my lack of actual JavaScript knowledge" - front-end developers in ~~2015.~~ 2018

RETWEETS 1,273
FAVORITES 1,066

2:07 PM - 14 Jun 2015

¯_(ツ)_/¯

Frameworks are just libraries. You need to know the language if you want to get things done.

Example: element selection

- The jQuery way
  - Require jquery during page load
  - $('.my-class');
- vanillaJS
  - No dependencies
  - document.querySelectorAll('.my-class');

Other actions

Action	VanillaJS
Set text	el.textContent = string
Set style	el.style.background-color = #FF32AB
Parse JSON	JSON.parse(json-string)
Set HTML	el.innerHTML = string
For each	Array.prototype.forEach.call(selected, function(sel, i){ })
Add class	el.classList.add(className)

VanillaJS keeps improving every year. Keep up, if you want to program for the web.

Frameworks 101

- We will look at two
  - Angular
  - React
- There are many others
  - Ember
  - Vue
  - Meteor

Angular basics

- No installation needed, its just a library call from your HTML code
- Angular directives extend the functionality of HTML code
  - *ng-app* signals the start of an angular application
  - *ng-init* initializes application data
  - *ng-model* binds values of the application data to HTML inputs
  - *ng-bind* binds application data to HTML tags
- Bindings are two-way
  - you can modify HTML content at will using program logic
  - You can enter data into JS via HTML inputs
- Bindings are real-time

Simple demo app

```
<html>
<head>
    <title>AngularJS First Application</title>
            <script src =
    "https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></
    script>
</head>
<body>
    <h1>Sample Application</h1>
    <div ng-app = "">
            <p>Enter your Name: <input type = "text" ng-model = "name"></p>
            <p>Hello <span ng-bind = "name"></span>!</p>
    </div>
</body>
</html>
```

Angular expressions

- Expressions bind specific application variables to HTML
  - Angular syntax: always use expressions within double braces like so – {{expression}}
- <p> Expense : Rs {{cost*quantity}} </p>
- <p> This is {{person.firstname + “ ” + person.lastname + “.”}} </p>
- <p> Score: {{score[3]}} </p>

Demo app with non-trivial directives

```
<html>
<head>
    <title>AngularJS Directives</title>
 </head>
<body>
    <h1>Sample Application</h1>
    <div ng-app = "" ng-init = "countries = [{locale:'en-US',name:'United States'}, {locale:'en-GB',name:'United Kingdom'}, {locale:'en-FR',name:'France'}]">
        <p>Enter your Name: <input type = "text" ng-model = "name"></p>      <p>Hello <span ng-bind = "name"></span>!</p>
        <p>List of Countries with locale:</p>
        <ol>
            <li ng-repeat = "country in countries"> {{ 'Country: ' + country.name + ', Locale: ' + country.locale }} </li>
        </ol>
    </div>
<script src = "https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
</body>
</html>
```

Angular controllers

- Angular controllers are JavaScript objects containing attributes and functions
- Each controller accepts scope as a parameter
  - Identifies which module the controller has to control
- Controllers allow arbitrary combinations of inputs and outputs into HTML display

Controller demo - HTML

```
<div ng-app = “testApp" ng-controller =
   "studentController">
Enter first name: <input type = "text" ng-model
   = "student.firstName">
Enter last name: <input type = "text" ng-model =
   "student.lastName">
 You are entering: {{student.fullName()}}
</div>
```

Controller demo - JS

```
<script>
    var testApp = angular.module(“testApp", []);
    testApp.controller('studentController', function($scope) {
            $scope.student = {
                        firstName: “Nisheeth",
                        lastName: “Srivastava",
                        fullName: function() {
                                    var studentObject;
                                    studentObject = $scope.student;
                                    return studentObject.firstName + " " +
    studentObject.lastName;
                        }
            };
     });
 </script>
```

Angular filters

- Filters are data selection operators that can be added to expressions or directives
- Examples
  - <li ng-repeat = "subject in student.subjects | filter: subjectName">
  - <li ng-repeat = "subject in student.subjects | orderBy:'marks'">

DOM directives

Name	Function
Ng-disabled	Disables a particular control
Ng-show	Shows a particular control
Ng-hide	Hides a given control
Ng-click	References a click event

Event directives

Ng-click	Ng-mousemove
Ng-dbl-click	Ng-mouseover
Ng-mousedown	Ng-keydown
Ng-mouseup	Ng-keyup
Ng-mouseenter	Ng-keypress
Ng-mouseleave	Ng-change

Getting data from the server

```
function studentController($scope,$https:) {
    var url = "data.txt";
    $https:.get(url).success(function(response) {
        $scope.students = response;
    });
 }
```

SPAs using Angular

- Can make single page applications using ng-view and ng-template directives
- The ng-view directive creates a placeholder within the HTML for a potential view
- The ng-template directive is used to create the corresponding view

Creating the placeholder and view

```
<div ng-app = "mainApp">
 ...
   <div ng-view> </div>
<script type = "text/ng-template" id =
   "addStudent.htm">
   <h2> Add Student </h2>
   {{message}}
</script>
</div>
```

Routing

```
var mainApp = angular.module("mainApp", ['ngRoute']);
mainApp.config(['$routeProvider', function($routeProvider) {
        $routeProvider.
        when('/addStudent', { templateUrl: 'addStudent.htm',
   controller: 'AddStudentController' }).
        when('/viewStudents', { templateUrl:
   'viewStudents.htm', controller: 'ViewStudentsController' }).
        otherwise({ redirectTo: '/addStudent' });
   }
]);
```

Multiple controllers on a page

```
<script>
 var mainApp = angular.module("mainApp", []);

 mainApp.controller("shapeController", function($scope) {
                  $scope.message = "In shape controller";
                 $scope.type = "Shape";
      }
);
 mainApp.controller("circleController", function($scope) {   $scope.message = "In circle controller";
      }
);
mainApp.controller("squareController", function($scope) { $scope.message = "In square controller";
                 $scope.type = "Square";
      }
);
</script>
```

Multiple controllers on a page

```
<div ng-app = "mainApp" ng-controller =
  "shapeController">
  <p>{{message}} <br/> {{type}} </p>
  <div ng-controller = "circleController">
      <p>{{message}} <br/> {{type}} </p>
  </div>
  <div ng-controller = "squareController">
      <p>{{message}} <br/> {{type}} </p>
  </div>
</div>
```

Services

- Services are JavaScript functions accessible to controllers
- Built-in services in Angular are prefixed with $, such as $https:
  - Look in docs for others

```
mainApp.service('CalcService', function(MathService){
    this.square = function(a) {
        return MathService.multiply(a,a);
       }
    });
```

Using a service

```
mainApp.controller(‘ShapeController',
   function($scope, CalcService) {
        $scope.square = function() {
         $scope.result =
   CalcService.square($scope.number);
         }
   });
```

Next week in lab

- Things to do
  - Make a location-aware mobile app using Ionic + angular
  - Hook your mobile app up to the Google Maps API reusing code I’ve uploaded to the course website
  - Route the express app you worked with last week using angular instead of express*
- First two components are your Assignment 3
  - Deadline October 20th
- No class next Thursday
- Work hard on your course project over the mid-sem break