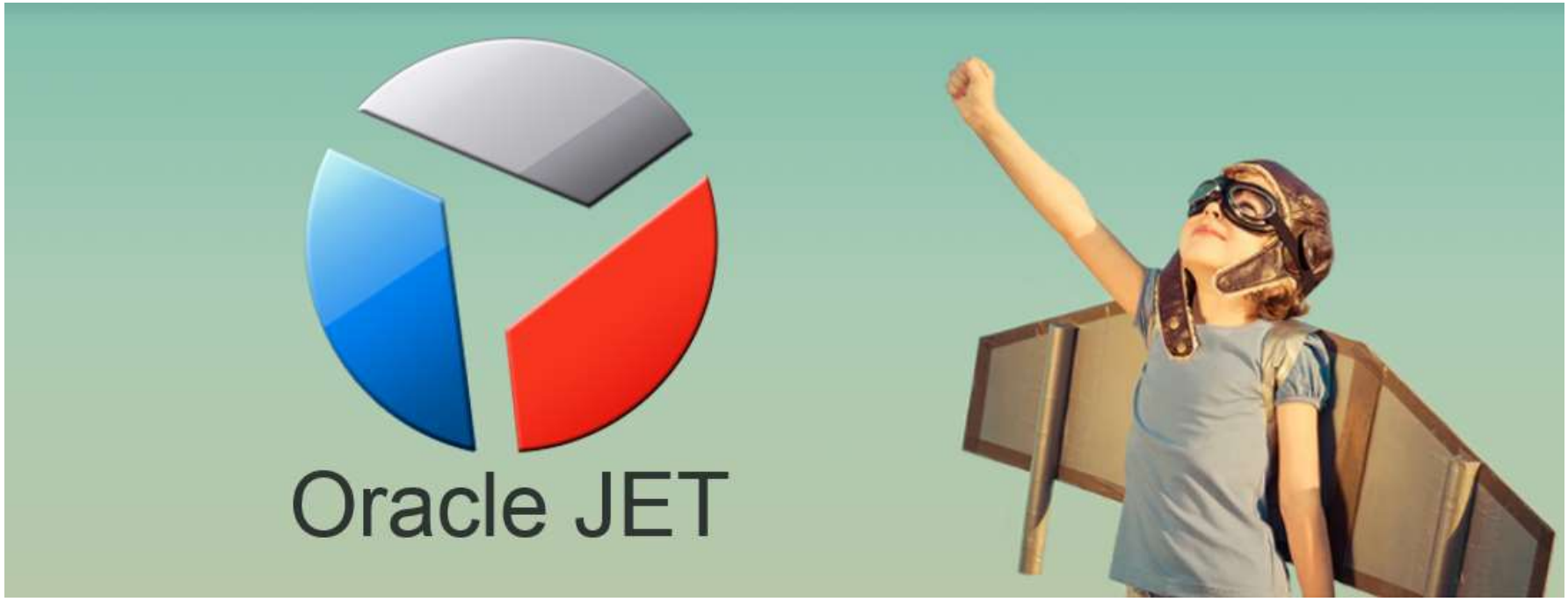
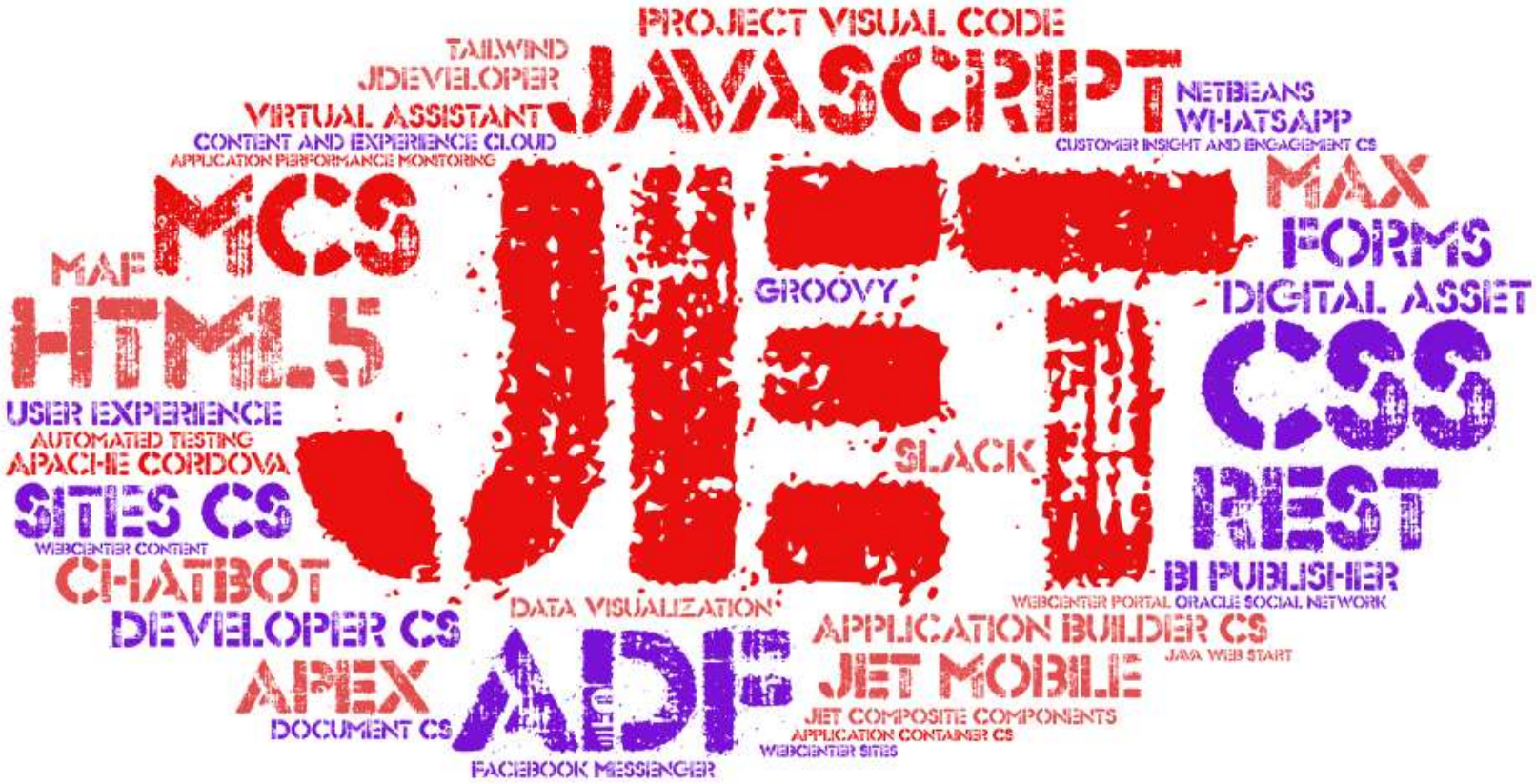
Oracle JET


Oracle JET

Oracle Solutions



OJET Toolkit

jQuery write less, do more.

jQuery user interface

REQUIRE.JS

APACHE CORDOVA™

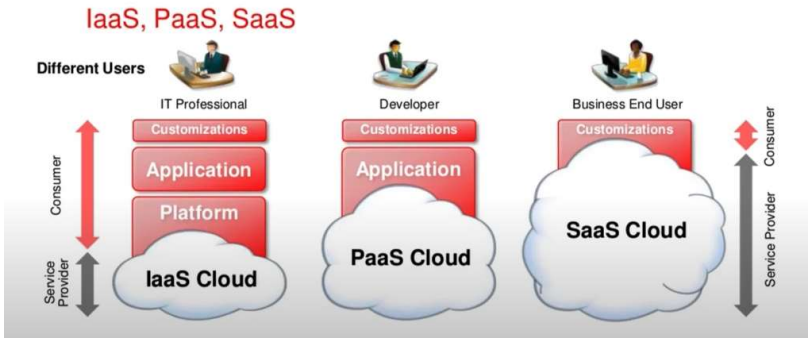
Knockout.

ORACLE JET

Javascript Ecosystem

- Vibrant, innovative ecosystem
- Increasing popularity of mobile devices
- Expense of native applications
- Popularity of hybrid mobile applications
- Challenges of a volatile, dynamic ecosystem
- Relevant and Maintainable over a number of years

Modern Business Applications and Cloud

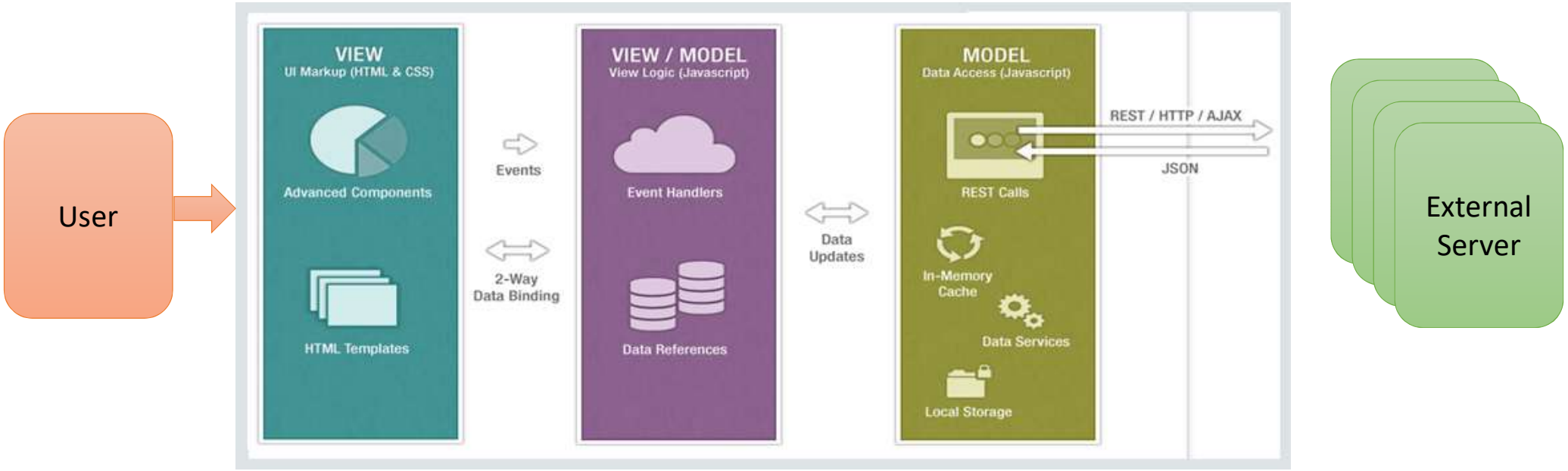


- JS is the native language of the internet
- Research with Cloud architectures
- OJET is the resultant of that research
- Oracle recommendations and best practices for enterprise JS

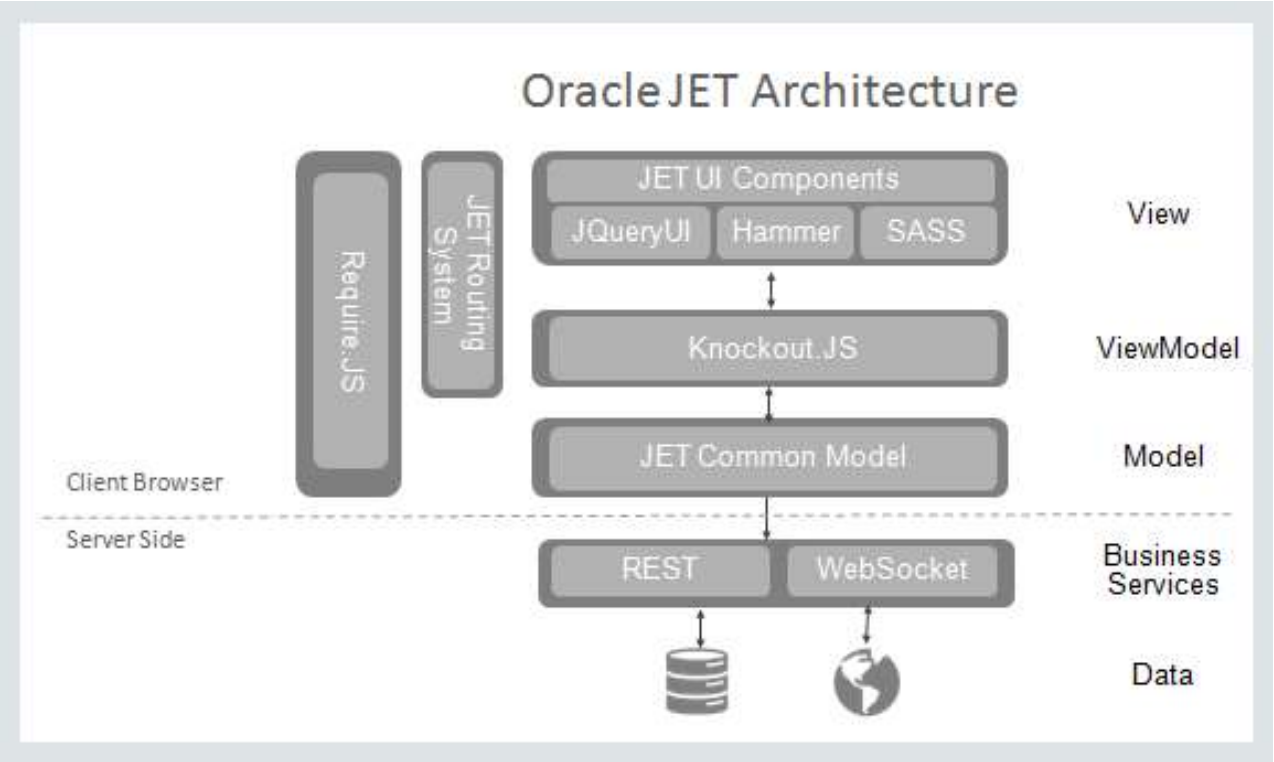
Oracle JET

- Free and Open Source
  - github.com/oracle/oraclejet
- Javascript Toolkit
- Useful Architectures, Patterns, Templates, Techniques, Components
- Focussed on Enterprise Applications
- Javascript, HTML5 and CSS3
- Result of research at Oracle in Javascript enterprise applications

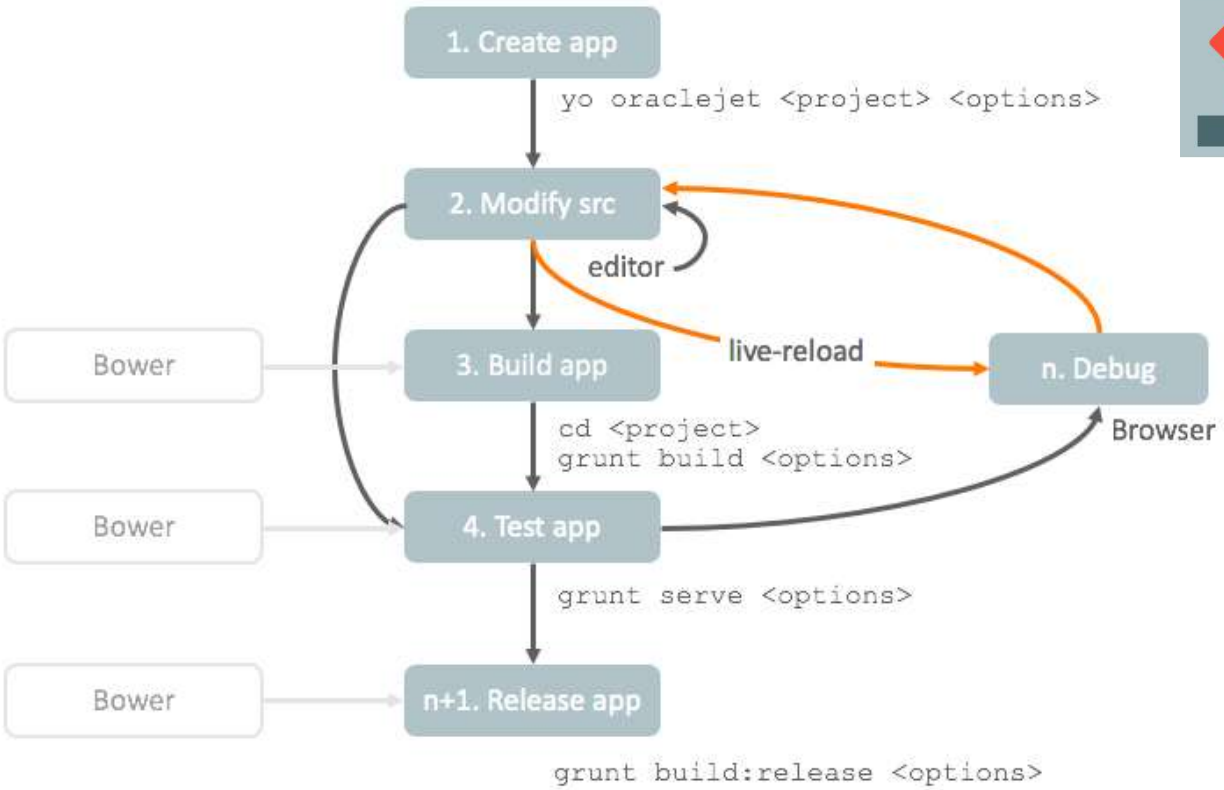
OJET MVVM Concepts



OJET Architecture



Development Process with OJET



Installation and Configuration

- Download and install node.js

  https://nodejs.org/en/download/

- Install OJET Client

  > (sudo) npm install -g @oracle/ojet-cli

- Verify the OJET CLI

  > ojet help
  > ojet --version

First OJET Web Application

- Developing client-side web application with Oracle JET is simple and efficient
- Create a starter template

```
> ojet create firstwebapp --template=navdrawer
```

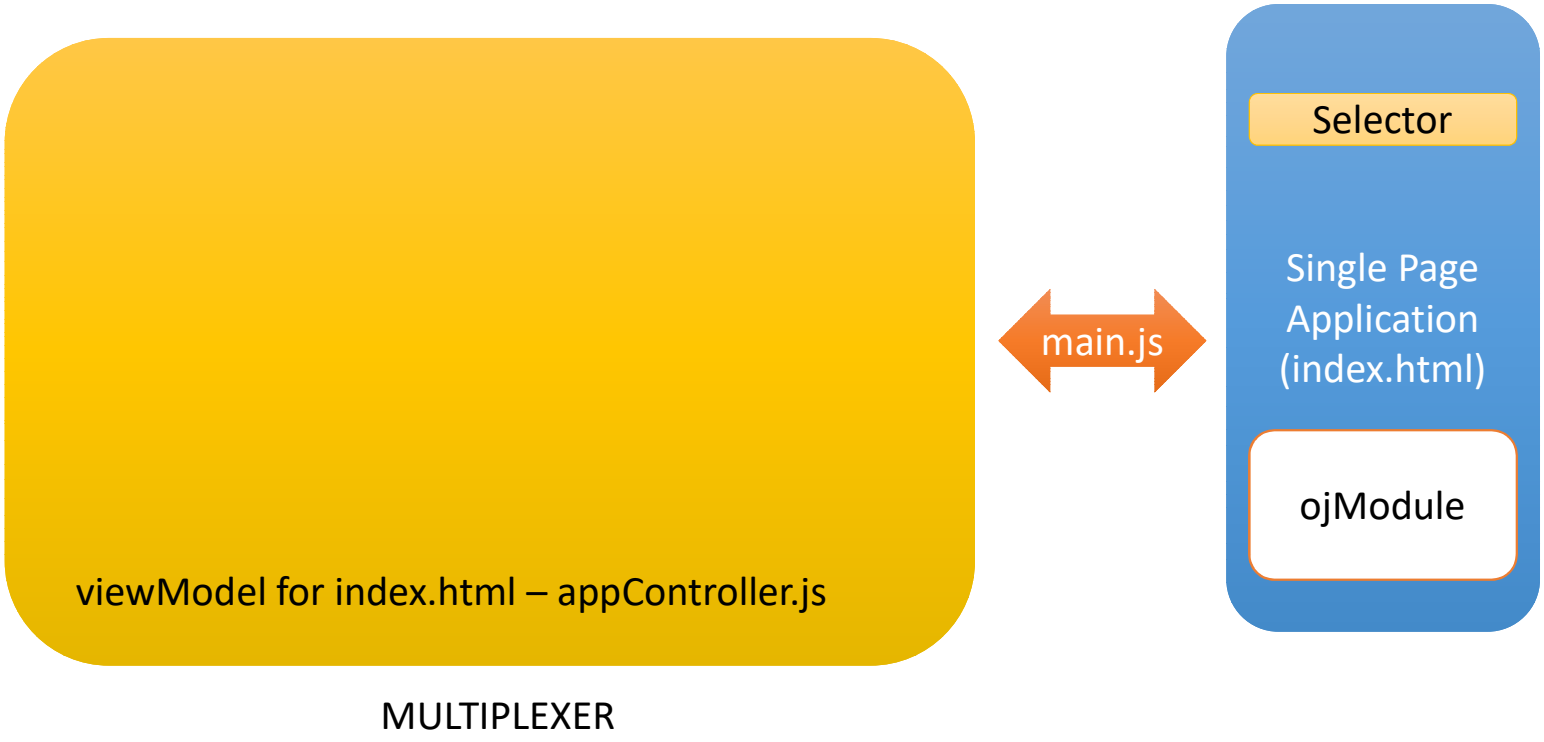
- Run the web application
  - Change the directory to firstwebapp
  - Execute the command

```
> ojet serve
```

LAB #1

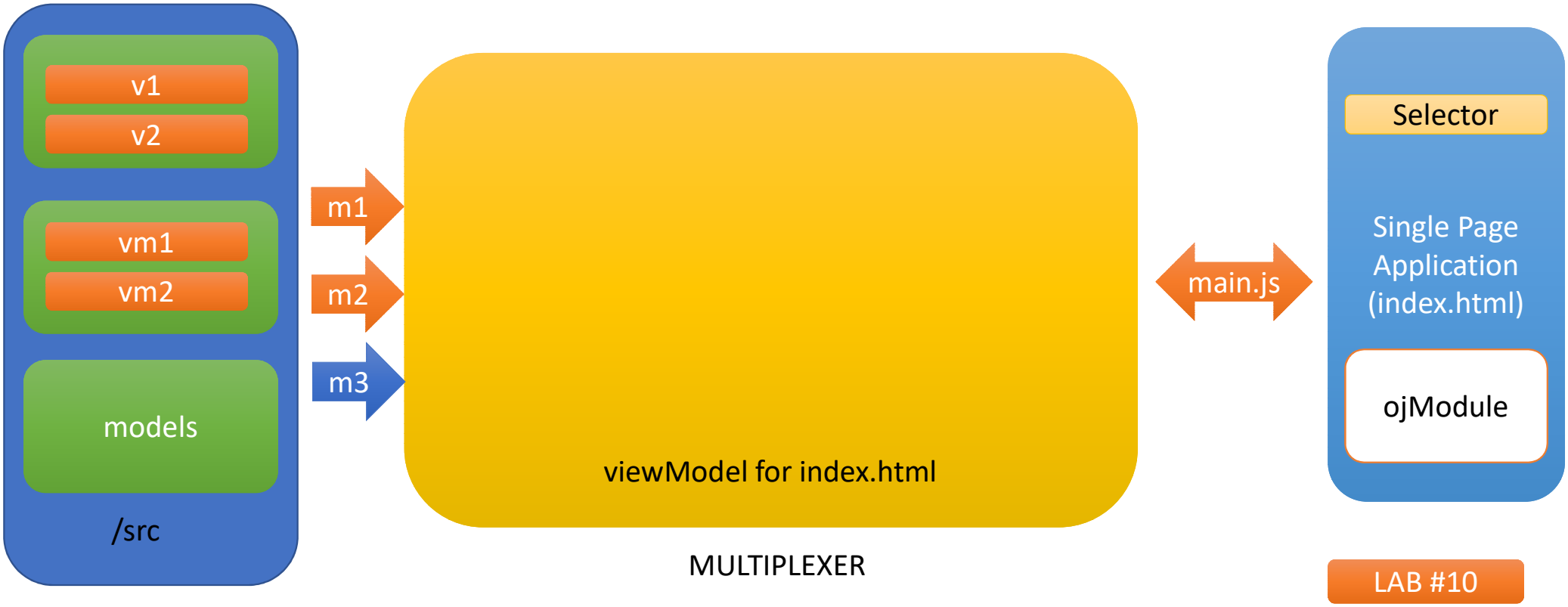
OJET Code Structure

Feature = Module = view + viewModel



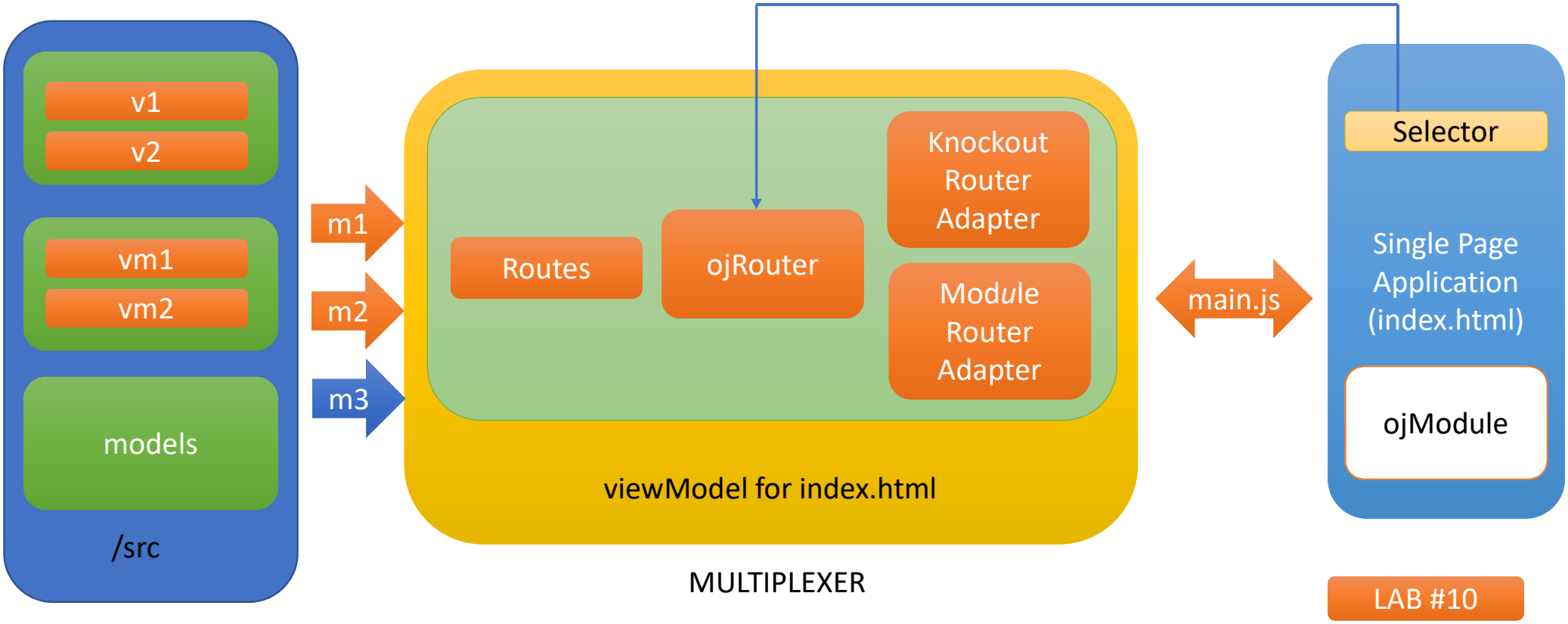
OJET Code Structure

Feature = Module = view + viewModel



OJET Code Structure

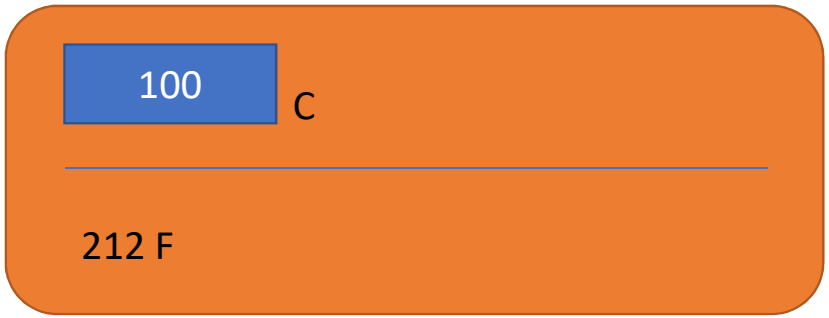
Feature = Module = view + viewModel



Evolution of Javascript Ecosystem

Using Require JS and Knockout JS in OJET

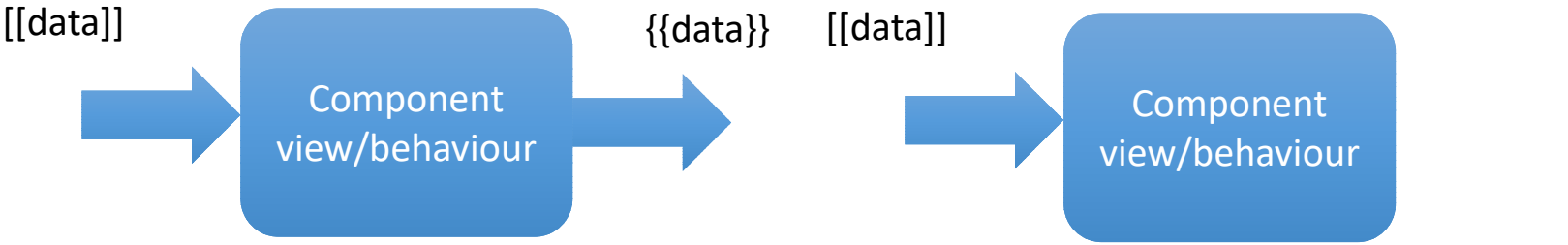
- Require JS helps in dependency management
- Knockout JS helps in data-binding and automatic tracking of value/property changes, it updates all references to that property automatically



LAB #2

Oracle JET Components

- Powerful and clean way of organizing UI code into self-contained reusable chunks
- Development patterns beneficial for large applications, simplifying by encapsulation
- Improve performance by incremental loading of resources



LAB #3

Loading Resources

- Require.js Text Plugin is an extension of Require.js
- Enables you to load text based resources
- Let’s you separate HTML templates from business logic in Javascript files

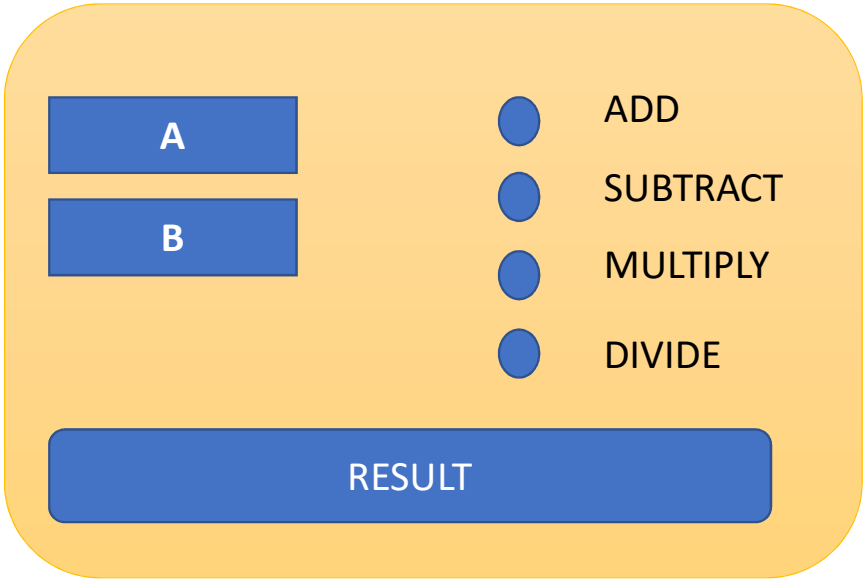
LAB #4

Creating Modules

- OJET applications are inherently modular
- An OJET module has its business logic in Javascript and its view in HTML
- An OJET module represents a section in your application providing a new feature to the user

LAB #5

Creative Lab: Add a new feature to the app



OJET Cookbook Components

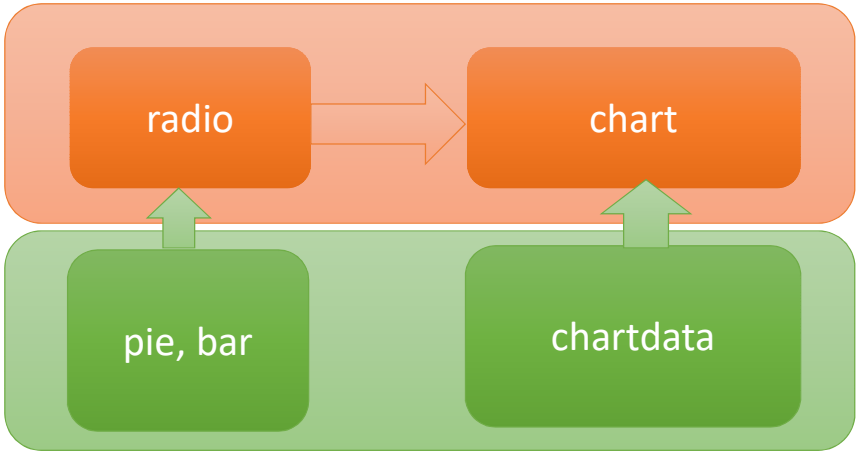
- Open sourced
- Online OJET Cookbook provides recipes for each component
  - HTML View
  - JS/TS Business Logic
- JSDoc describes each component

Explore Cookbook, Understand Workflow

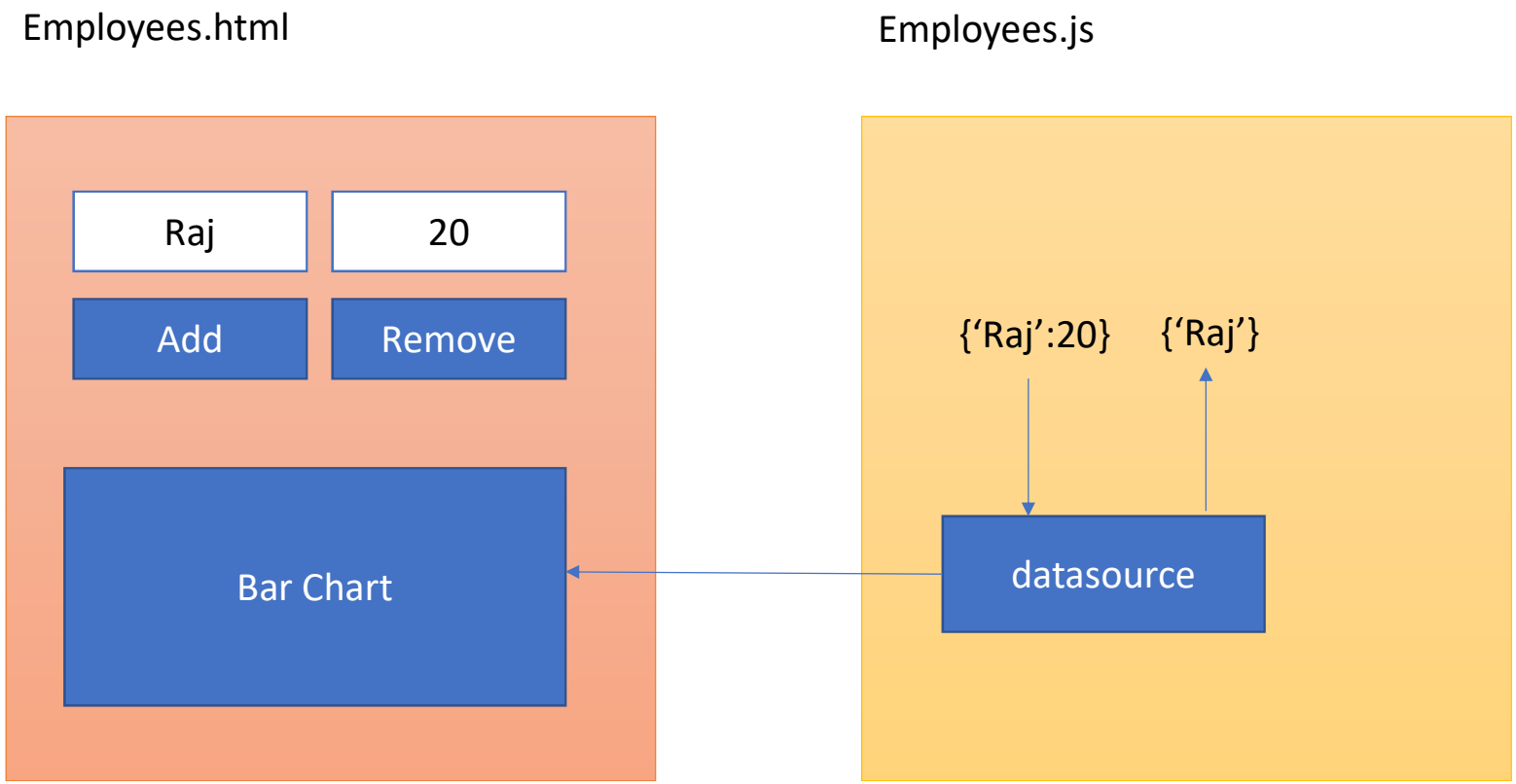
OJET Cookbook Components

view

viewModel



LAB #6


Employees.html
Employees.js
Raj
20
Add
Remove
Bar Chart
{'Raj':20}
{'Raj'}
datasource

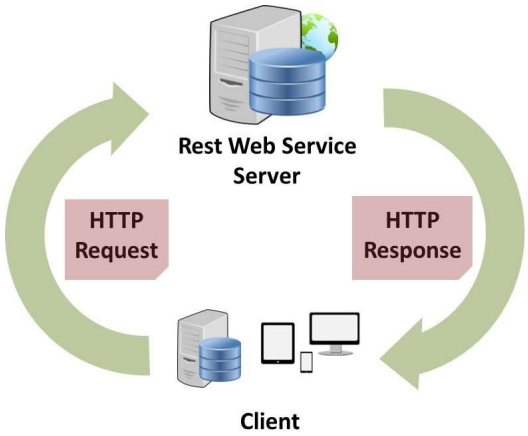
OJET Flex Layouts

- Oracle ALTA UI design systems
  https://www.oracle.com/webfolder/ux/middleware/alta/index.html
  - Set of CSS classes used to design modern Oracle applications
  - Default theme of Oracle JET
  - Design with mobile first approach
- Flex box layout is the standard part
  - Efficient way to layout, align and distribute space for applications
  - Helps accommodate variety of device visual spaces

LAB #7

Accessing and Integrating Data

- We access data using REST Services
- Representation State Transfer
  - Data and functionality are resources
  - Resources are accessed using URI
  - Benefit: Simplicity and it's a standard
  - Can expose data via Java EE, Spring Boot, Node, etc
- OJET allows you to create REST clients to integrate REST services
- jQuery API can be used for REST calls

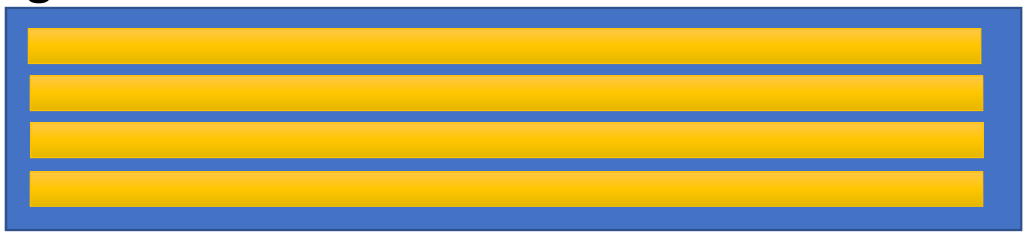


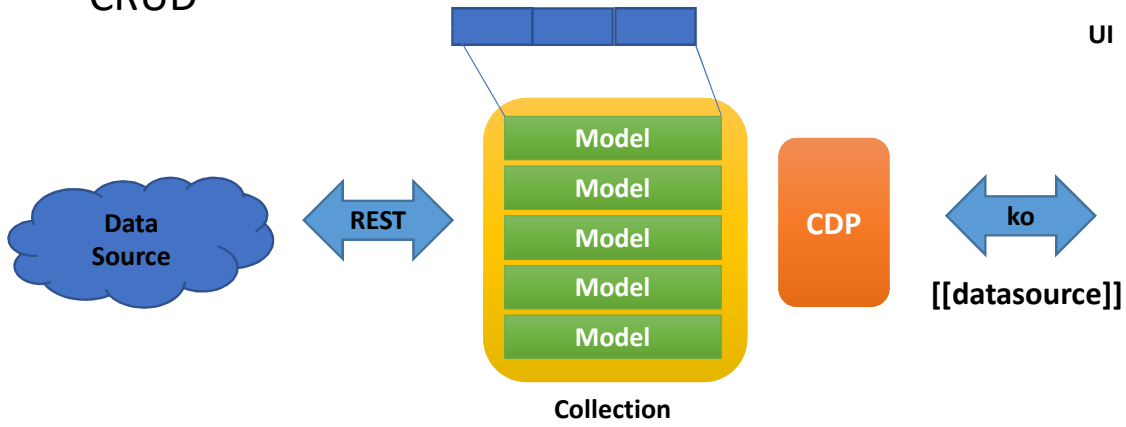
Experiment with a REST endpoint

Common Model

Rest Server

- Idiomatic alternative to working with data
- **Ordered set of models**
- Inspired by Backbone JS
  - Model and Collections
  - CRUD





Employees

Employee Id	First Name	Last Name	Email	Phone Number	Salary
105	David	Austin	DAUSTIN	590.423.4569	4800
106	Valli	Pataballa	VPATABAL	590.423.4560	4800
107	Diana	Lorentz	DLORENTZ	590.423.5567	5000
108	Nancy	Greenberg	NGREENBE	515.124.4569	12000
109	Daniel	Faviet	DFAVIET	515.124.4169	9000

Page 2 of 4 (6-10 of 20 items) 1 2 3 4

```
var collection = [
{“Employee ID”:105, “First Name”: David, “Last Name”:”Austin”, “Email”:”DAUSTIN”, “Phone Number”:590-423-4569”, “Salary”:4800},
…
]
```

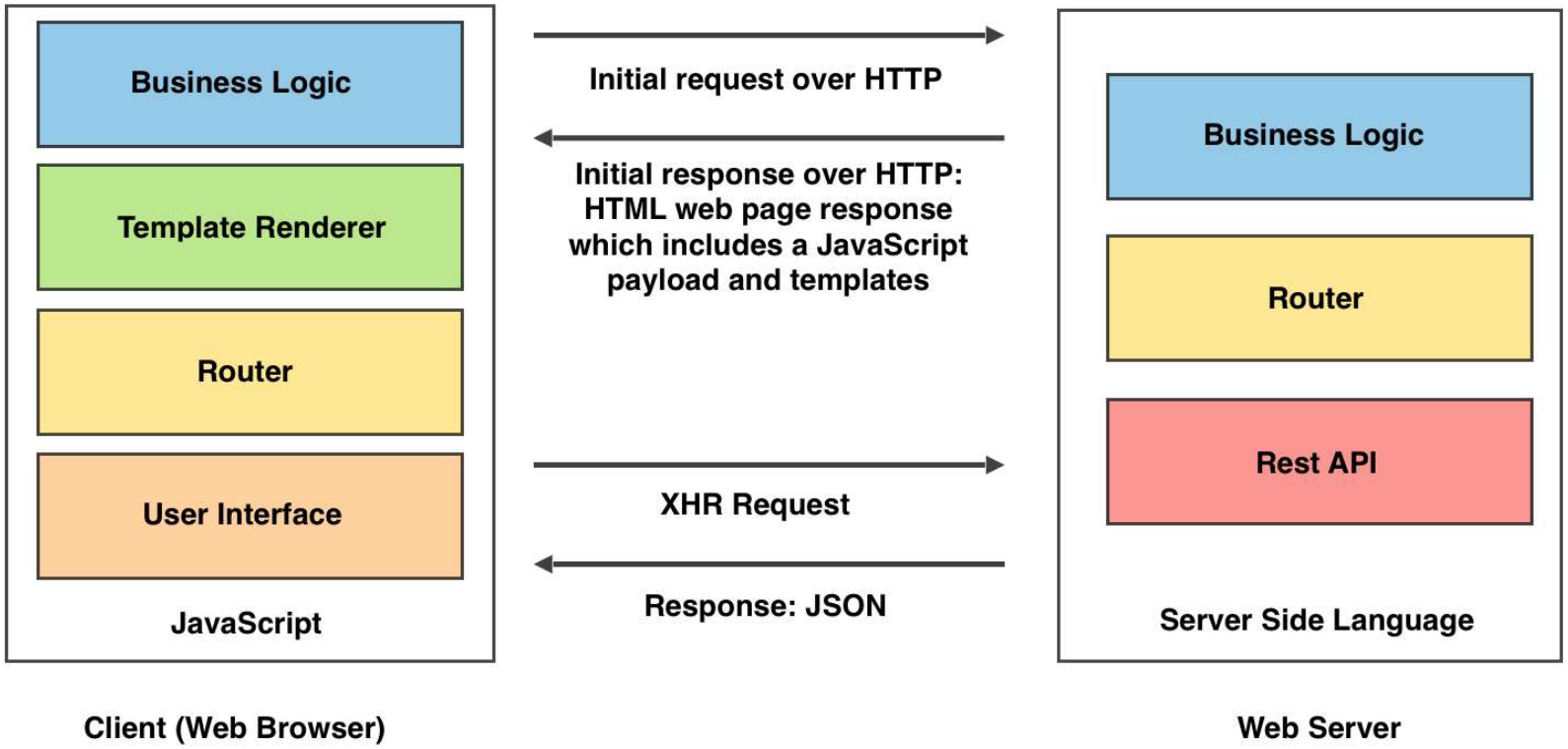
LAB #8

Single Page Applications

- A single-page application is a web application or website that interacts with the web browser by **dynamically rewriting the current web page with new data** from the web server, instead of the default method of the browser loading entire new pages
- https://www.excellentwebworld.com/what-is-a-single-page-application/

https://globekit.co/

LAB #9


Business Logic
Template Renderer
Router
User Interface
JavaScript
Client (Web Browser)
Initial request over HTTP
Initial response over HTTP: HTML web page response which includes a JavaScript payload and templates
XHR Request
Response: JSON
Business Logic
Router
Rest API
Server Side Language
Web Server

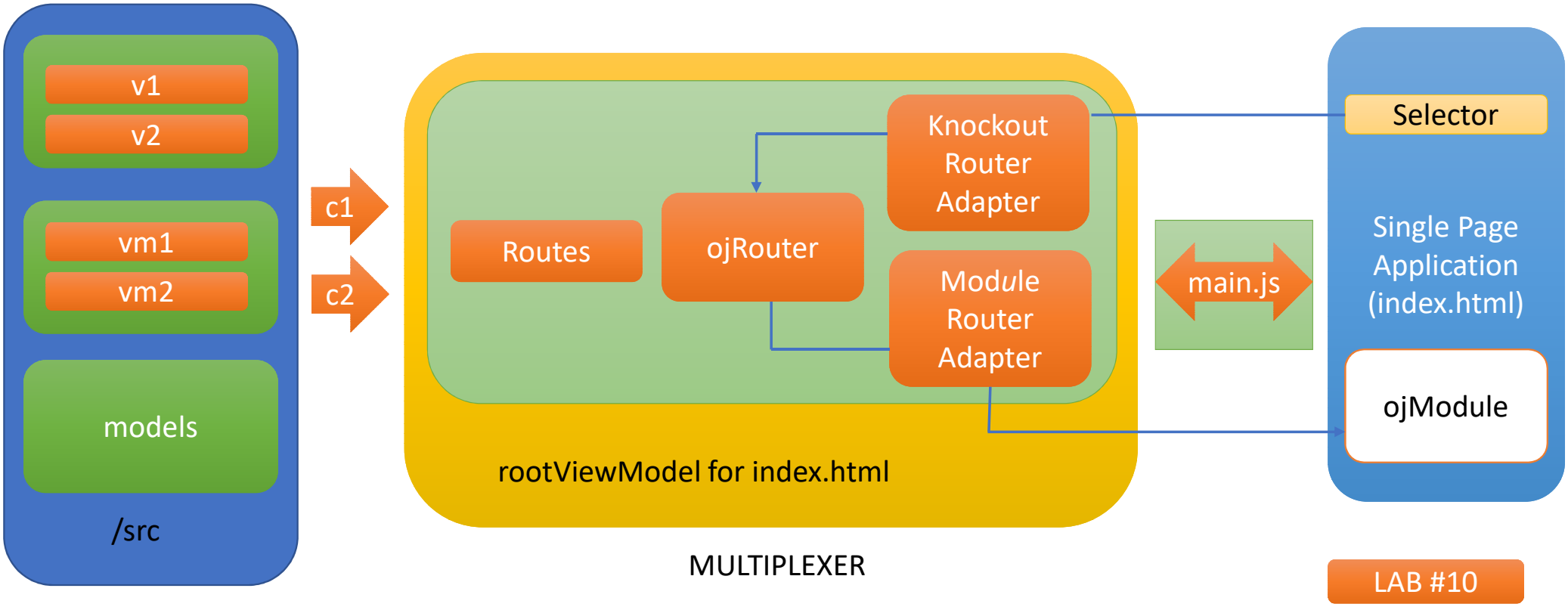
Building OJET Applications from Scratch

- Create a blank or basic application and proceed

```
> ojet create webapp –template=blank|basic
```

OJET Router

Feature = Module = view + viewModel



Custom Components: view and viewModel


```
<div class="oj-panel oj-margin">
    <div class="oj-lg-float-end" id='btcl'>
      <oj-bind-slot name='toolbar'>
      </oj-bind-slot>
    </div>
    <oj-table id='t1' aria-label='Employees Table'
          data='[[pagingDatasource]]'
          columns='[[tableColumns]]'
          selection-mode.row='single'
          on-selection-changed='[[rowSelectionListener]]'
          style='width: 100%;'>
        <oj-paging-control id="pg1" data='[[pagingDatasource]]' page-size='[[pageSize]]' slot='bottom'>
        </oj-paging-control>
    </oj-table>
</div>
```

```
function ExampleComponentModel(context) {
    var self = this;
    self.composite = context.element;

    self.restBackEndUrl = ko.observable();
    self.keyVal = ko.observable();
    self.pageSize = ko.observable();

    self.createModel = function () {
        var ModelData = oj.Model.extend({
            urlRoot: self.restBackEndUrl(),
            idAttribute: self.keyVal()
        });

        return new ModelData();
    };

    self.createCollection = function () {
        var CollectionData = oj.Collection.extend({
            url: self.restBackEndUrl(),
            fetchSize: self.pageSize(),
            model: this.createModel()
        });

        return new CollectionData();
    };

    context.props.then(function (propertyMap) {
        //Store a reference to the properties for any later use
        self.properties = propertyMap;

        //Parse your component properties here
        self.restBackEndUrl(self.properties.restBackendUrl);
        self.keyVal(self.properties.keyVal);
        self.pageSize(self.properties.pageSize);
        self.tableColumns = self.properties.columns;

        self.listData = self.createCollection();
        self.pagingDatasource = ko.observable(new oj.PagingTableDataSource(new oj.CollectionTableDataSource(self.listData)));
    });
```

Custom Components: properties

```
"properties": {
  "restbackendurl": {
    "description": "REST Backend URL",
    "type": "string"
  },
  "keyval": {
    "description": "REST Backend Key",
    "type": "string"
  },
  "pagesize": {
    "description": "REST Backend Page Size",
    "type": "number"
  },
  "columns": {
    "description": "Column structure",
    "type": "array",
    "writeback": false,
    "properties": {
      "headerText": {
        "description": "Column header name",
        "type": "string"
      },
      "field": {
        "description": "Column attribute mapping",
        "type": "string"
      }
    }
  }
},
```

```
"slots": {
  "toolbar": {
    "description": "Placeholder for table buttons"
  }
}
```

```
"methods": {
  "testMethod": {
      "description": "Test method",
      "params": [{
          "name": "param",
          "description": "Parameter",
          "type": "object"
      }]
  }
},
```

```
"events": {
  "handleSelection": {
    "description": "Triggered when table row is selected",
    "bubbles": true,
    "detail": {
      "value": {
        "description": "Selected row key",
        "type": "string"
      }
    }
  }
},
```

component.json

Custom Components: usage

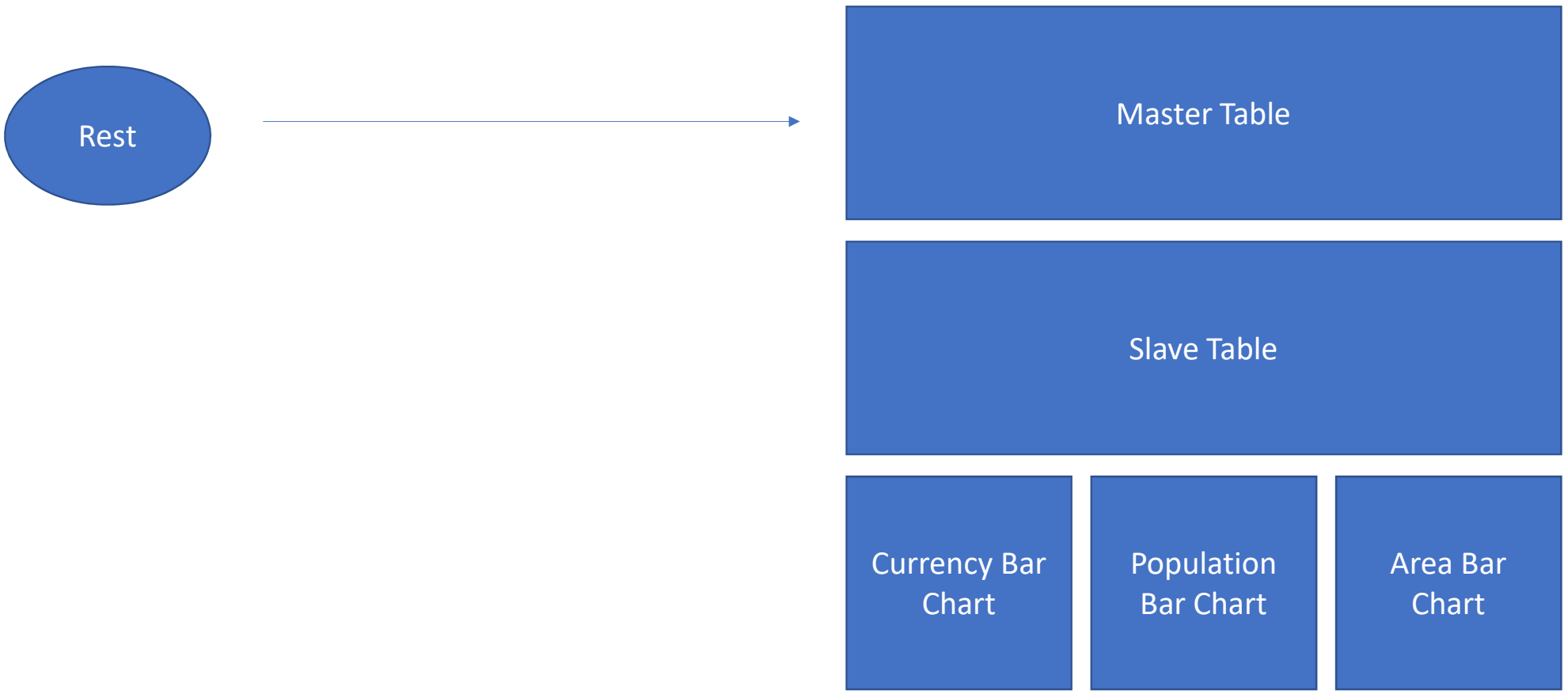

```
<div class="oj-hybrid-padding">
  <table-redsan id="t1" restbackendurl="[[urlEmployees]]"
    keyval="[[keyEmployees]]" pagesize="[[pagesizeEmployees]]"
    columns="[[tableColumns]]" on-handle-selection="[[handleTableRowSelection]]">
    <oj-button slot="toolbar" on-click="[[testMethodCall]]">Call Method</oj-button>
  </table-redsan>
</div>
```

```
define(['ojs/ojcore', 'knockout', 'jquery', 'ojs/ojcomposite', 'jet-composites/table-redsan/loader',
        'ojs/ojbutton'],
 function(oj, ko, $) {

    function DashboardViewModel() {
      var self = this;

      self.urlEmployees = "http://[redacted]/restapp/rest/1/Employees?onlyData=true";
      self.keyEmployees = "EmployeeId";
      self.pagesizeEmployees = 10;

      self.tableColumns = [];
      self.tableColumns.push({"headerText": "First Name", "field": "FirstName"});
      self.tableColumns.push({"headerText": "Last Name", "field": "LastName"});
      self.tableColumns.push({"headerText": "Email", "field": "Email"});
      self.tableColumns.push({"headerText": "Phone Number", "field": "PhoneNumber"});
      self.tableColumns.push({"headerText": "Salary", "field": "Salary"});

      self.testMethodCall = function(event) {
        document.getElementById('t1').testMethod('Hello');
      };

      self.handleTableRowSelection = function(event) {
        console.log('Selected key handling outside component: ' + event.detail.value);
      };
```

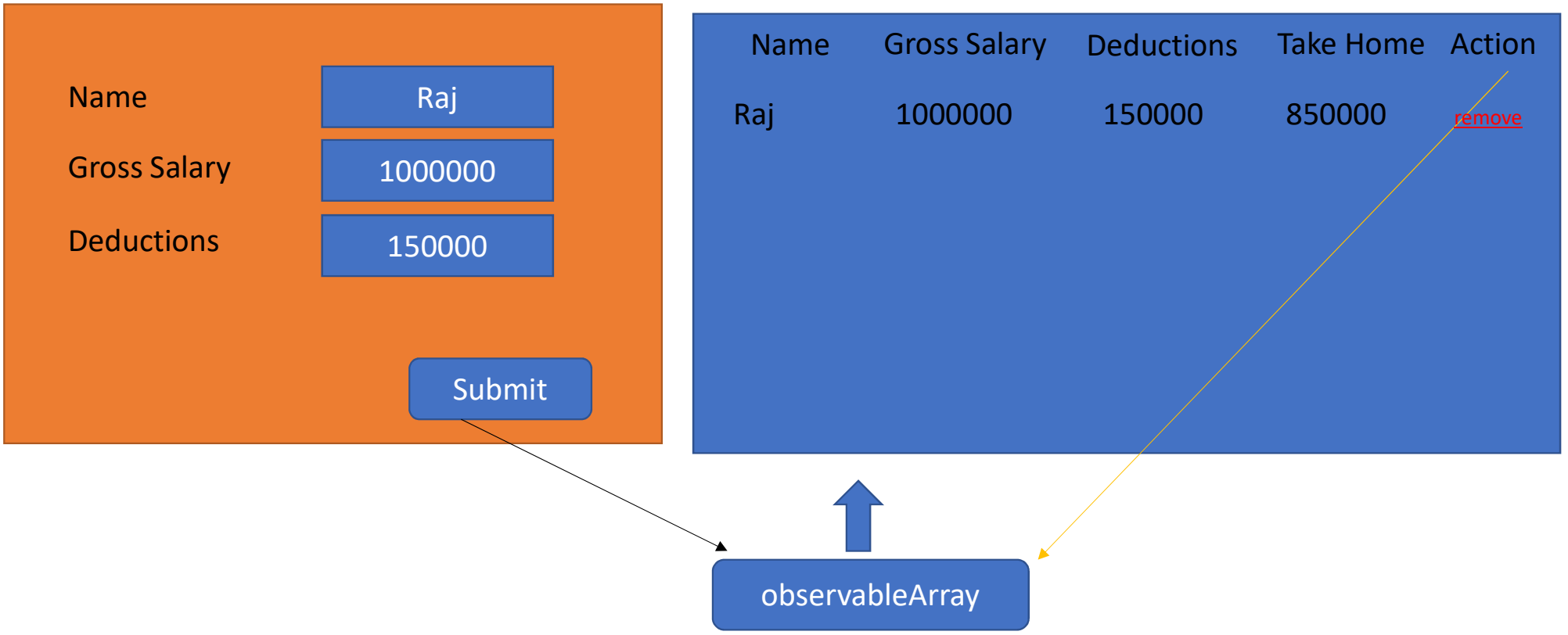
Oracle ADF vs Oracle JET

Oracle JET vs Oracle ADF – The Basics

	Oracle ADF	Oracle JET
Who is it for	Java Developer	JavaScript Client Developer
Where it runs	Server	Client
Development Experience	Declarative	Code centric
Development Tools	JDeveloper / OEPE	Any
Architecture	MVC	MVVM
Data Services	ADF BC, EJB/POJO, SOAP, REST	REST, JSON
Customization & Backward Compatibility	MDS and XML based	Develop Your Own
Based on	Java EE	Popular open source libraries
Integrated Solutions	SOA, BI, WebCenter	Application Builder Cloud Service
What Oracle Builds with it	SaaS applications	PaaS products


Rest
Master Table
Slave Table
Currency Bar Chart
Population Bar Chart
Area Bar Chart

Creative Lab: KO Application in OJET Project



oj-table