Integrating Angular with Node.js RESTful Services

COURSE INTRODUCTION

Dan Wahlin
WAHLIN CONSULTING

@DanWahlin www.codewithdan.com

Module Overview

Pre-requisites to maximize learning

Learning goals

Server-side technologies and concepts

Client-side technologies and concepts

Running the sample application

Running the sample application with Docker

Pre-Requisites to Maximize Learning

Course Pre-Requisites

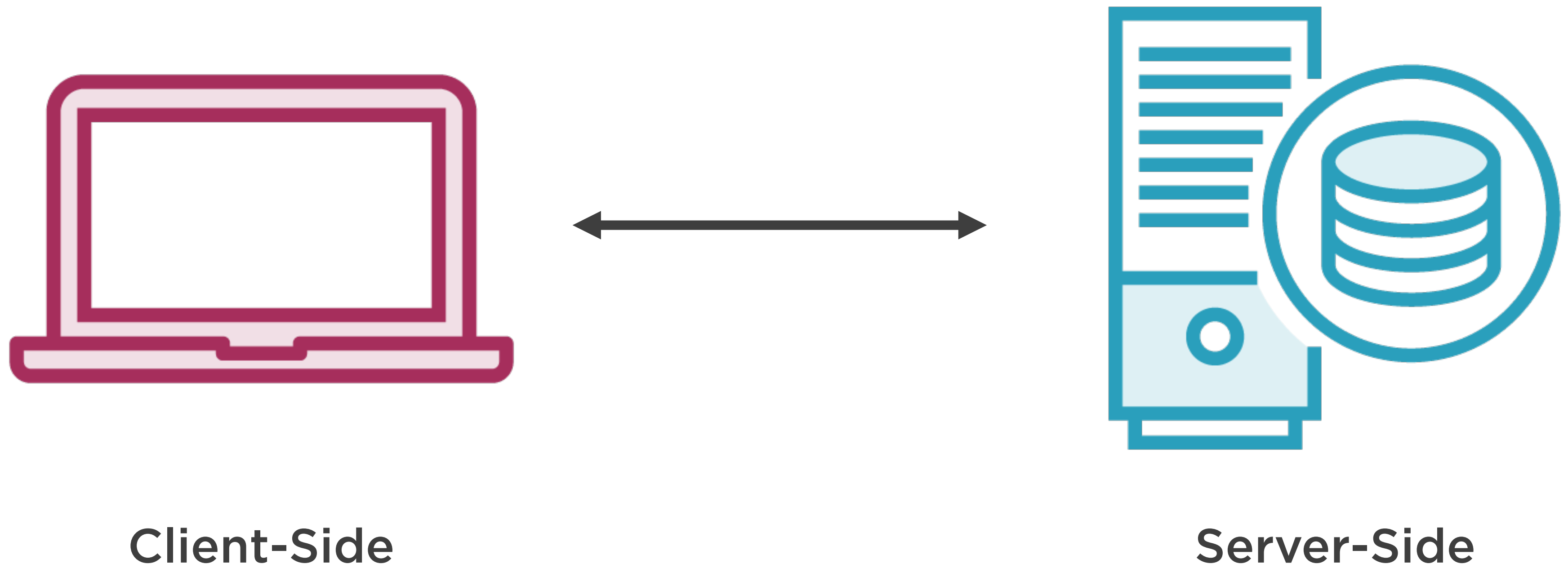
TypeScript

Fundamentals

Fundamentals

Learning Goals

Learning Goals



Server-Side Learning Goals

Learn how to use Node.js and Express to create a RESTful service

- Create convention-based routes
- Expose RESTful endpoints
- Integrate with a database

Client-Side Learning Goals

Learn how to use the Angular Http client to integrate with a RESTful service

- Understand the role of RxJS and observables
- Retrieve and display data from a RESTful service using Http
- Insert, update and delete data
- Page data

Server-Side Technologies and Concepts

Server-side Technologies and Concepts

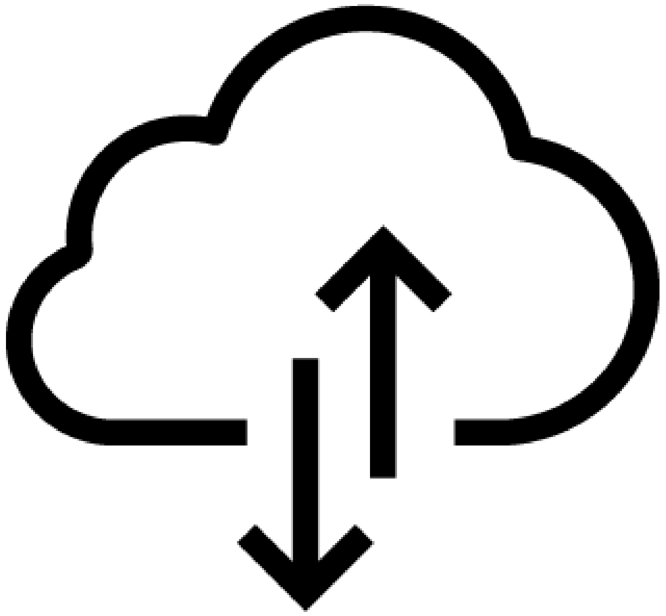
Node.js

MongoDB

HTTP

REST

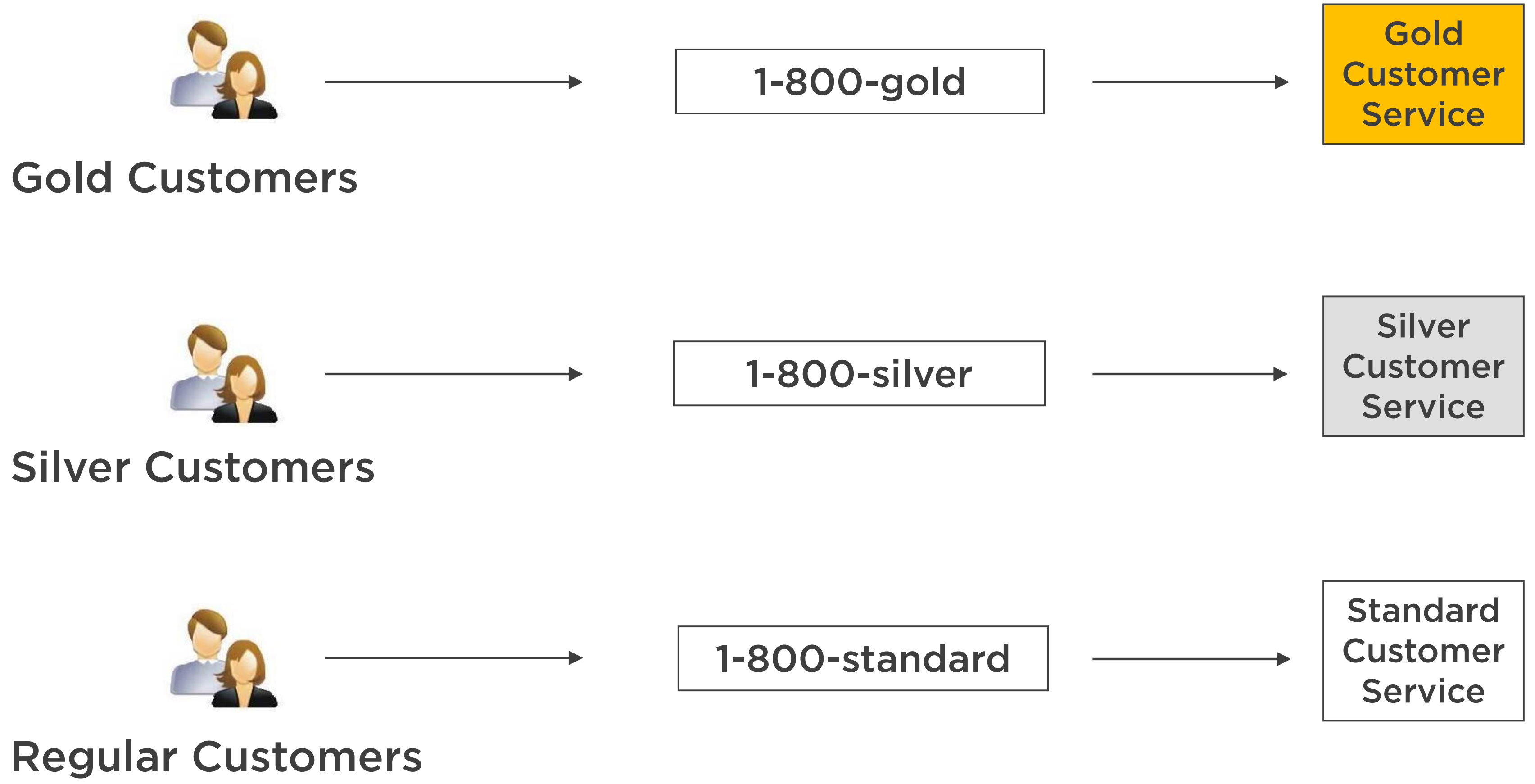
Introduction to REST



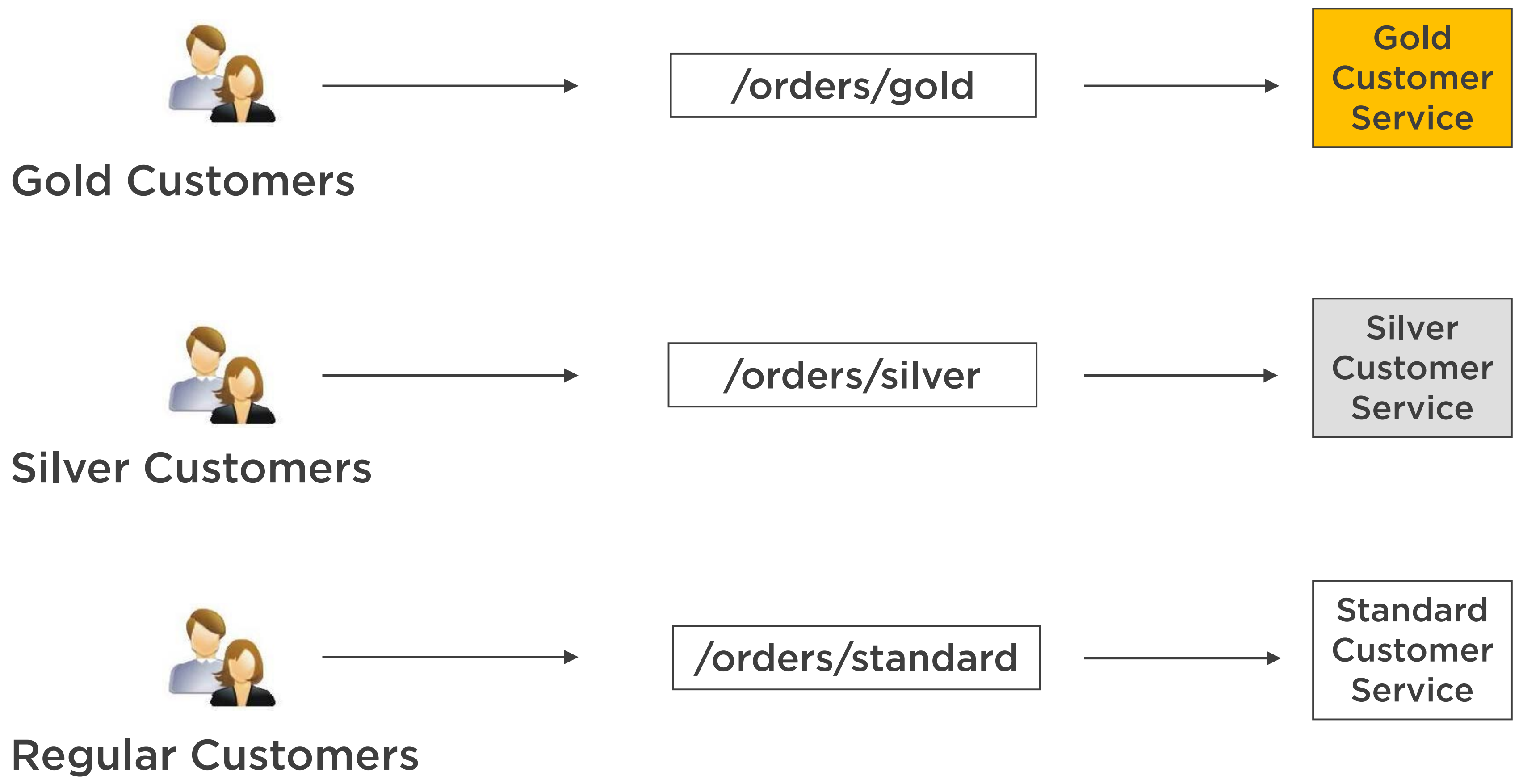
REST = Representational State Transfer

- Architectural style for distributed systems
- Exposes resources (state) to clients
- Resources identified with a URI
- Uses HTTP, URIs, MIME types

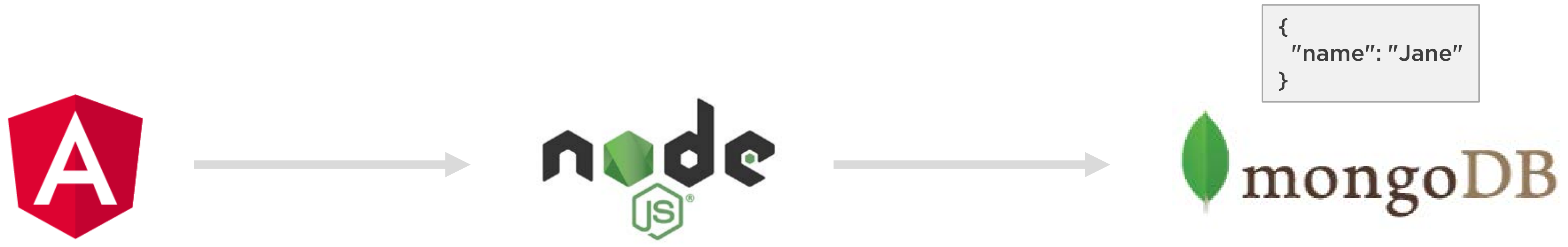
"RESTful" System Overview



RESTful Services and URIs

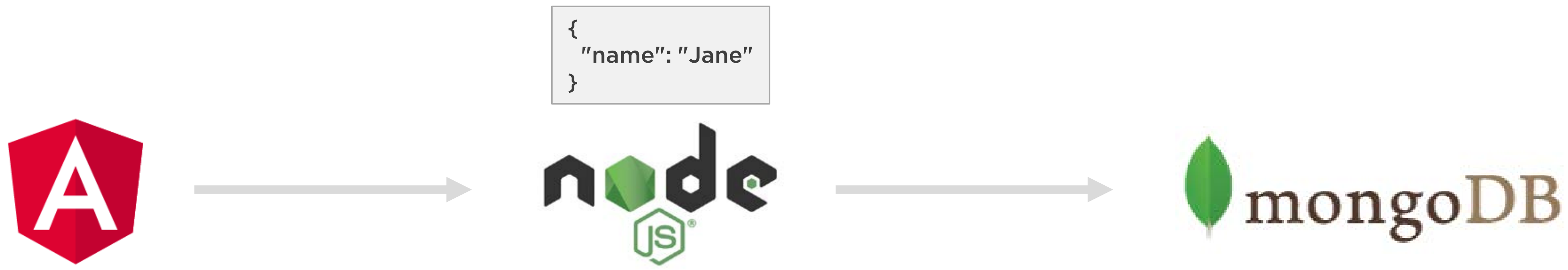


Key Technology Players



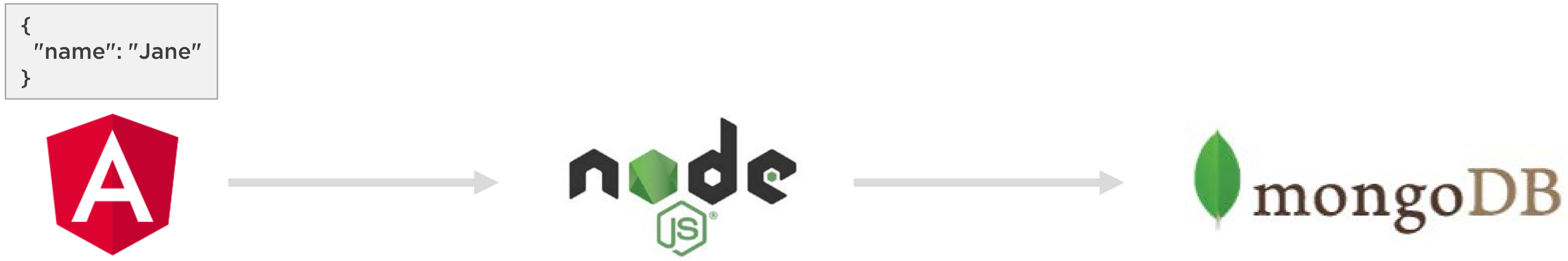
```
{
  "name": "Jane"
}
```

Key Technology Players



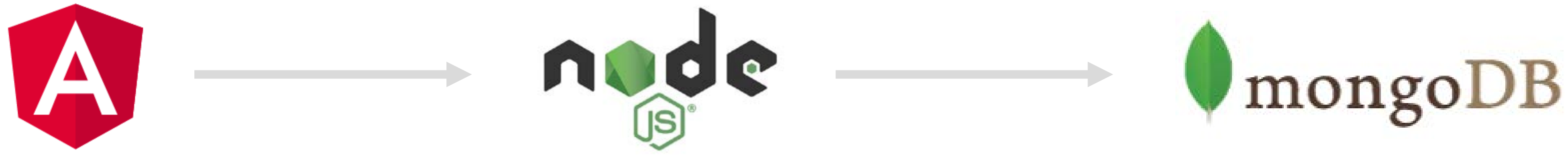
```
{
  "name": "Jane"
}
```

Key Technology Players



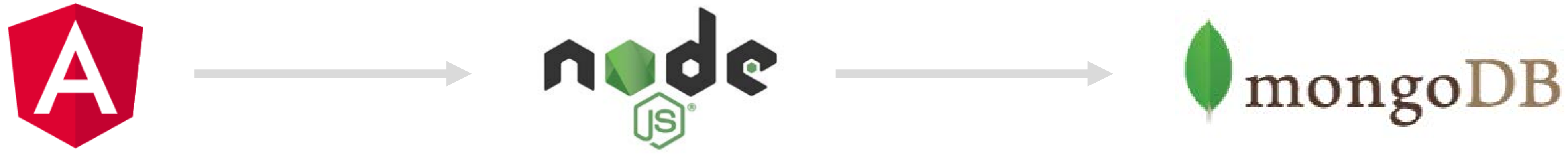
```
{
  "name": "Jane"
}
```

Key Technology Players



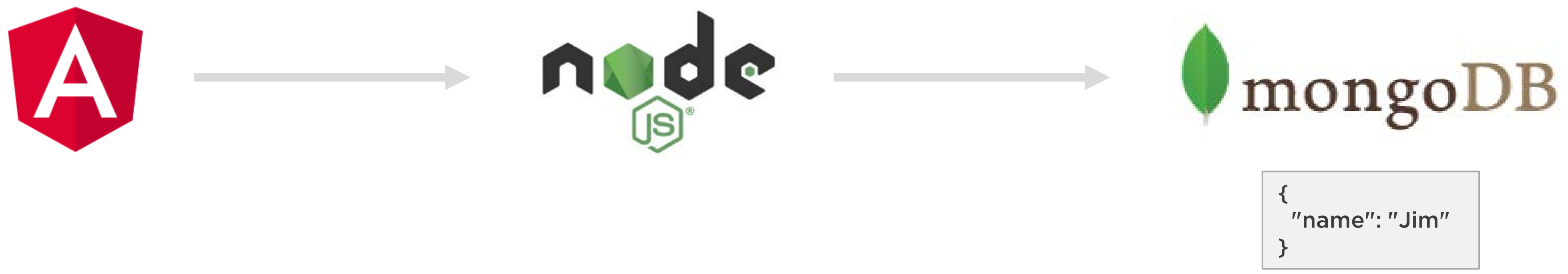
```
{
  "name": "Jim"
}
```

Key Technology Players



```
{
  "name": "Jim"
}
```

Key Technology Players



```
{
  "name": "Jim"
}
```

Client-Side Technologies and Concepts

Client-Side Technologies and Concepts

Angular

RxJS

XHR/HTTP

Observables

RxJS

http://reactivex.io/rxjs

Reactive Extensions for JavaScript

- Library for composing asynchronous and event-based programs
- Relies on observable sequences
- Used with Angular

Promises and Observables

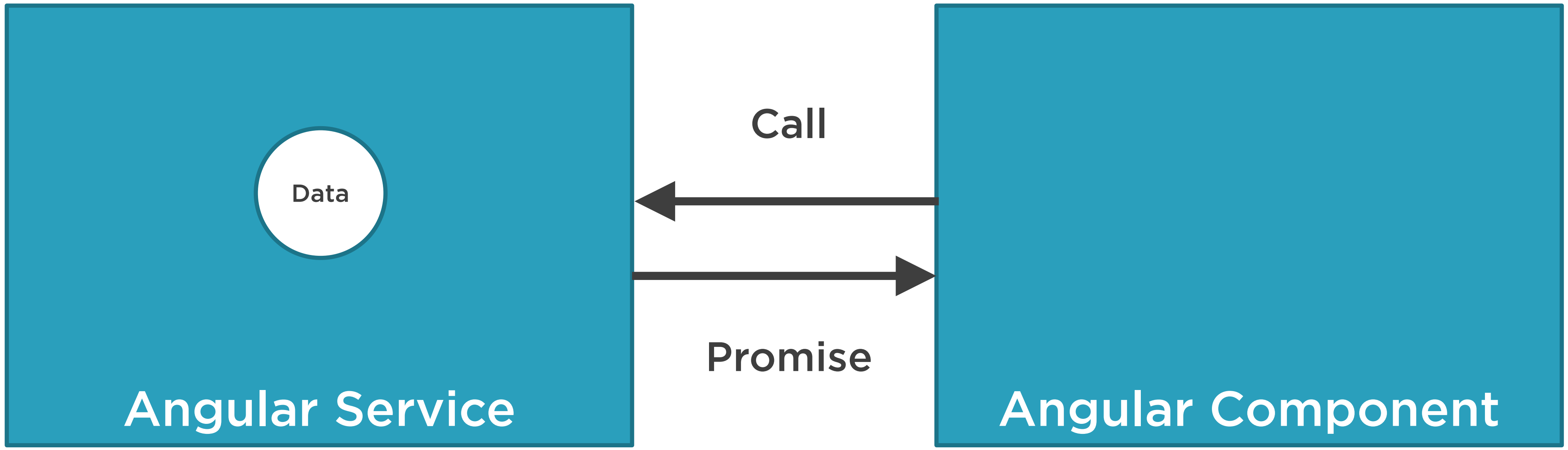
Promise

- An operation that hasn't completed yet, but is expected in the future
- Used with async/deferred operations
- Can be hooked to a callback

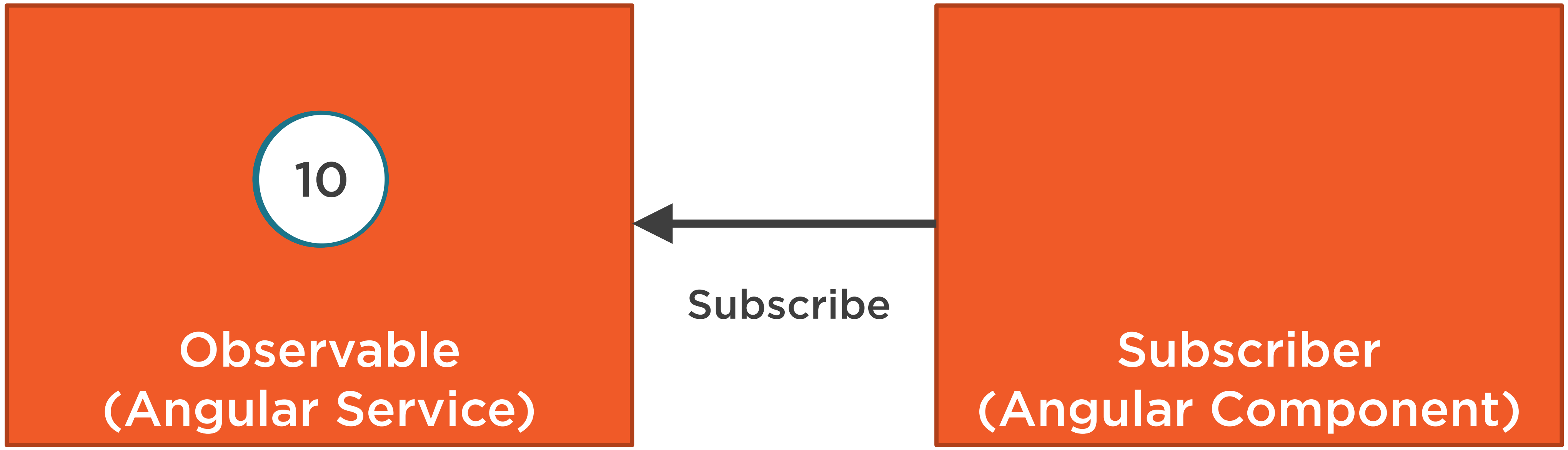
Observable

- An object that can be "subscribed" to by other objects
- Can return multiple values over time – an async data stream
- Event based

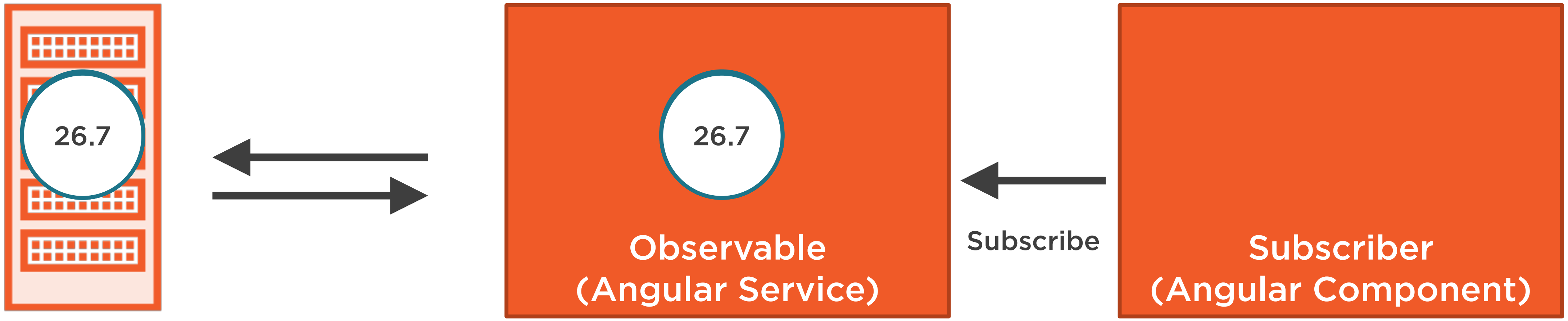
Promises Overview



Observables Overview



Observables and Async Streams



Promises and Observables Review

Promises	Observables
Returns a single value	Can return multiple values over time
Cannot cancel	Can cancel
Natively supported in browsers	Supports standard array functions (map, filter, reduce, etc.)
	Relies on a library such as RxJS

Running the Application

Software Requirements

https://code.visualstudio.com

https://nodejs.org

https://docs.mongodb.com/manual/installation

Running the Application with Docker

Software Installation

https://code.visualstudio.com

https://nodejs.org

https://docker.com

Summary

Key learning goals include understanding how to move data to and from a RESTful service

Key technologies and concepts

- Node/MongoDB/Http/REST
- Angular/RxJS/Observables/Http

Exploring the Node.js and Angular Application

Dan Wahlin
WAHLIN CONSULTING

@DanWahlin www.codewithdan.com

Module Overview

Exploring the project structure

Application modules

Configuring Node.js routes

Configuring the ES module loader

Creating Angular modules, components and services

Exploring the Project Structure

Application Modules

Configuring Node.js Routes

Configuring the ES Module Loader

Angular Modules, Components and Services

Summary

Project structure generated using express-generator

Standard npm modules for Node.js and Angular are used in the application

Application includes a custom "convention-based" routing feature

Multiple Angular modules used

Retrieving Data Using a GET Action

Dan Wahlin
WAHLIN CONSULTING

@DanWahlin   www.codewithdan.com

Module Overview

Creating GET actions using Express

Making GET requests with an Angular service

Displaying customers in a grid

Displaying a customer in a form

Creating a GET Action to Return Multiple Customers

Creating a GET Action to Return a Single Customer

Making GET Requests with an Angular Service

Displaying Customers in a Grid

Displaying a Customer in a Form

Converting to a "Reactive" Form

Summary

GET actions provide a way to return data to a variety of clients

Angular services provide reusable functionality

Angular supports different data binding and forms techniques

Inserting Data Using a POST Action

Dan Wahlin
WAHLIN CONSULTING
@DanWahlin www.codewithdan.com

Module Overview

Creating a POST action using Express

Making a POST request with an Angular service

Modify the customer form to support inserts

Explore the "Reactive" form

Creating a POST Action to Insert a Customer

Making a POST Request with an Angular Service

Modifying the Customer Form to Support Inserts

Exploring the "Reactive" Form

Summary

A POST action provides a way to perform an insert operation

Angular's Http client supports HTTP POST actions

Angular services provide a great way to reuse code in an application

Updating Data Using a PUT Action

Dan Wahlin
WAHLIN CONSULTING
@DanWahlin www.codewithdan.com

Module Overview

Creating a PUT action using Express

Making a PUT request with an Angular service

Modify the customer form to support updates

Explore the "Reactive" form

Creating a PUT Action to Update a Customer

Making a PUT Request with an Angular Service

Modifying the Customer Form to Support Updates

Exploring the "Reactive" Form

Summary

A PUT action provides a way to perform an update operation

Angular's Http client supports HTTP PUT actions

Template forms and Reactive forms integrate with Angular services

Deleting Data Using a DELETE Action

Dan Wahlin
WAHLIN CONSULTING

@DanWahlin www.codewithdan.com

Module Overview

Creating a DELETE action using Express

Making a DELETE request with an Angular service

Modify the customer form to support deletes

Explore the "Reactive" form

Creating a DELETE Action to Delete a Customer

Making a DELETE Request with an Angular Service

Modifying the Customer Form to Support Deletes

Exploring the "Reactive" Form

Summary

A DELETE action provides a way to perform a delete operation

Angular's Http client supports HTTP DELETE actions

Modal dialogs aren't the only way to confirm deletes!

Data Paging, HTTP Headers, and CSRF

Dan Wahlin
WAHLIN CONSULTING

@DanWahlin   www.codewithdan.com

Module Overview

Add a paging header to a RESTful service response

Access headers in an Angular service

Add paging support to a component

Adding a paging component

CSRF Overview

Add CSRF functionality with csurf

Using a csurf token in an Angular service

Adding a Paging Header to a RESTful Service Response

Accessing Headers and Data in an Angular Service

Adding Paging Support to a Component

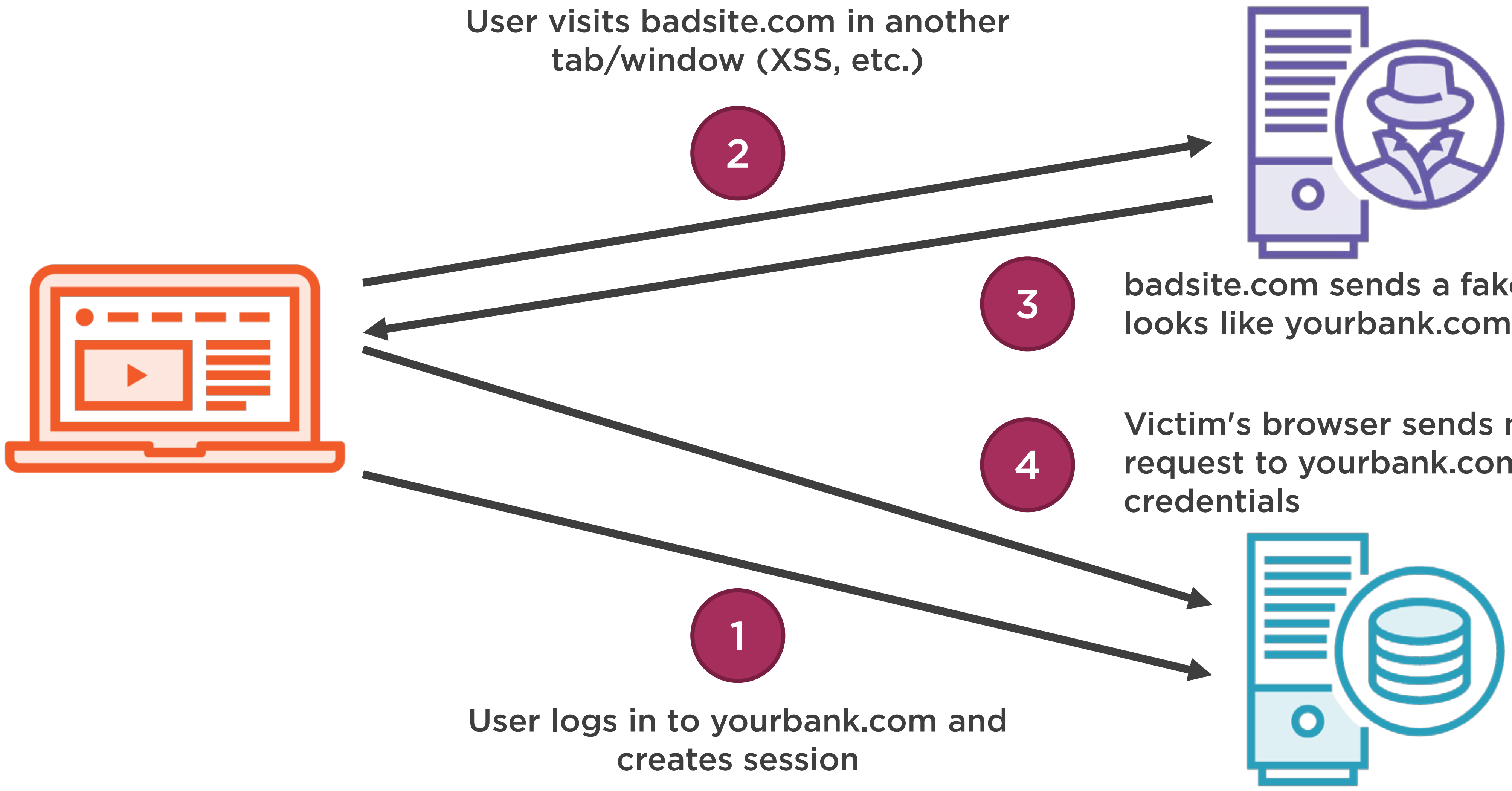
Adding a Paging Component

CSRF Overview

CSRF

Cross-Site Request Forgery

What Is CSRF?



1 User logs in to yourbank.com and creates session

2 User visits badsite.com in another tab/window (XSS, etc.)

3 badsite.com sends a fake page that looks like yourbank.com to the user

4 Victim's browser sends malicious request to yourbank.com with user credentials

How CSRF Works

Form Post from badsite.com

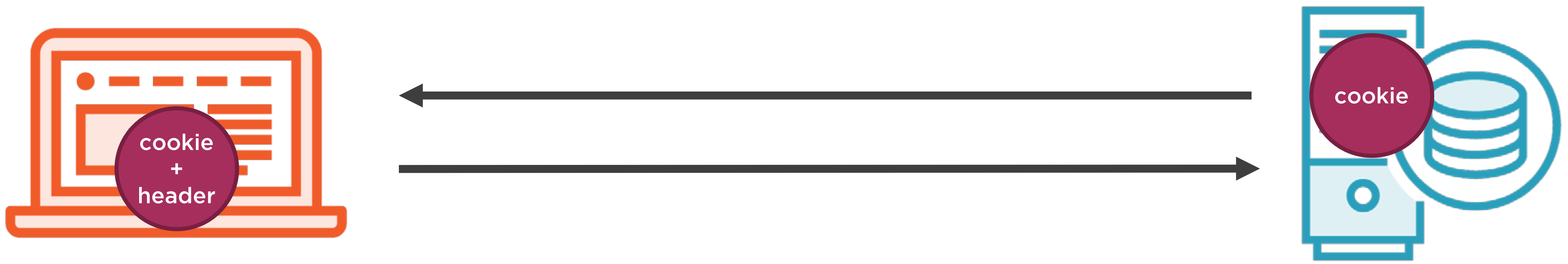
```
<form action="https://yourbank.com/someOperation" method="POST">
    <button type="submit">Click here for free money!</button>
</form>
<script>
  document.forms[0].submit();
</script>
```

Http/Ajax request from badsite.com

```
<script>
  http.post('http://yourbank.com/api/someOperation', data);
</script>
```

Adding CSRF Functionality with csurf

yourbank.com passes a cookie to the client
that contains a CSRF token value



Client sends the cookie back to yourbank.com
AND sets a header with the cookie value in it

Server compares cookie
value to header value

Using a csurf Token in an Angular Service

Summary

Headers can be used to send and receive data for paging and more

Components provide a reusable way to add paging functionality

CSRF attacks can be shutdown using modules such as csurf

Angular provides built-in CSRF functionality

Course Summary

Dan Wahlin
WAHLIN CONSULTING
@DanWahlin www.codewithdan.com