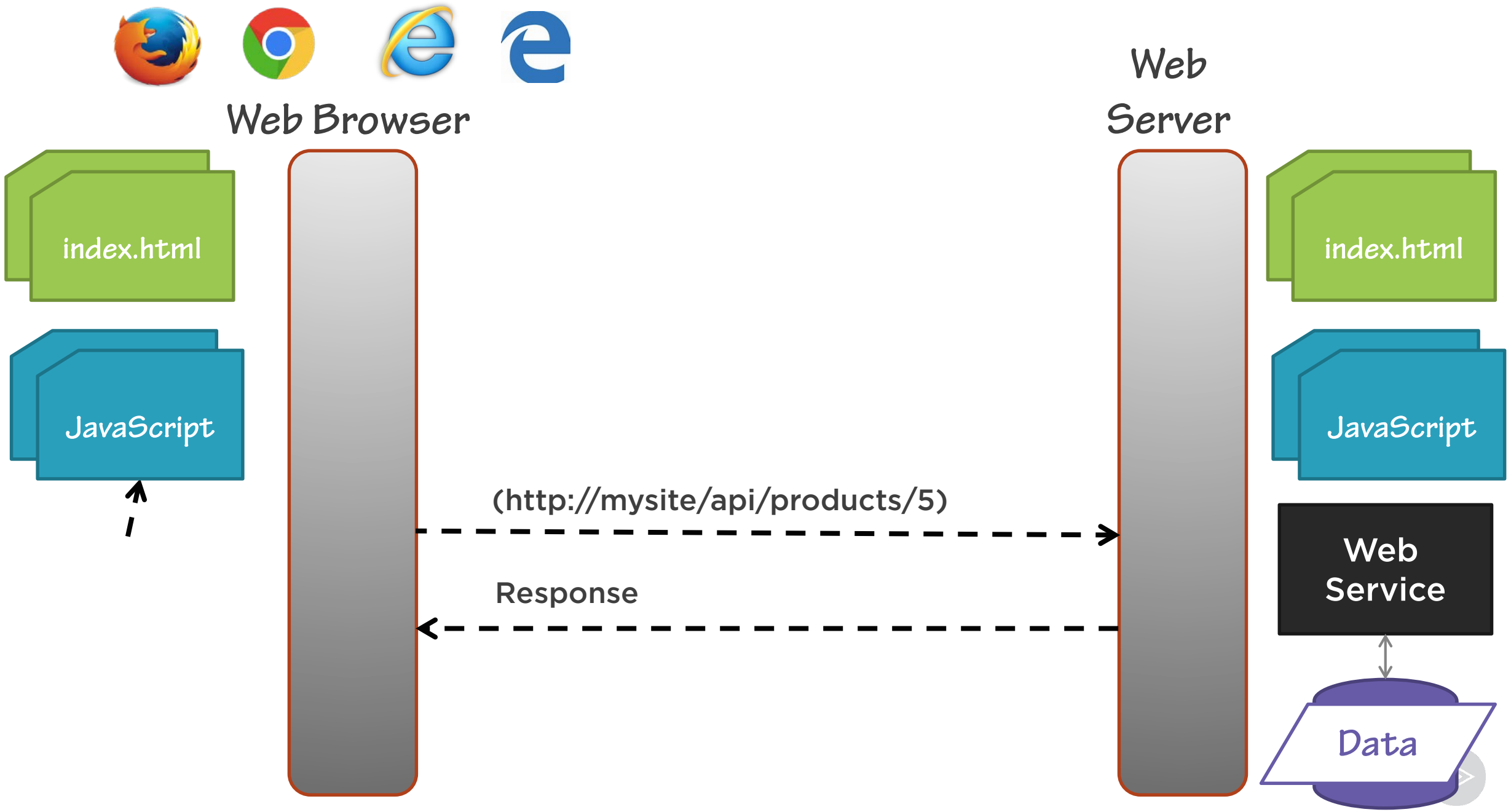
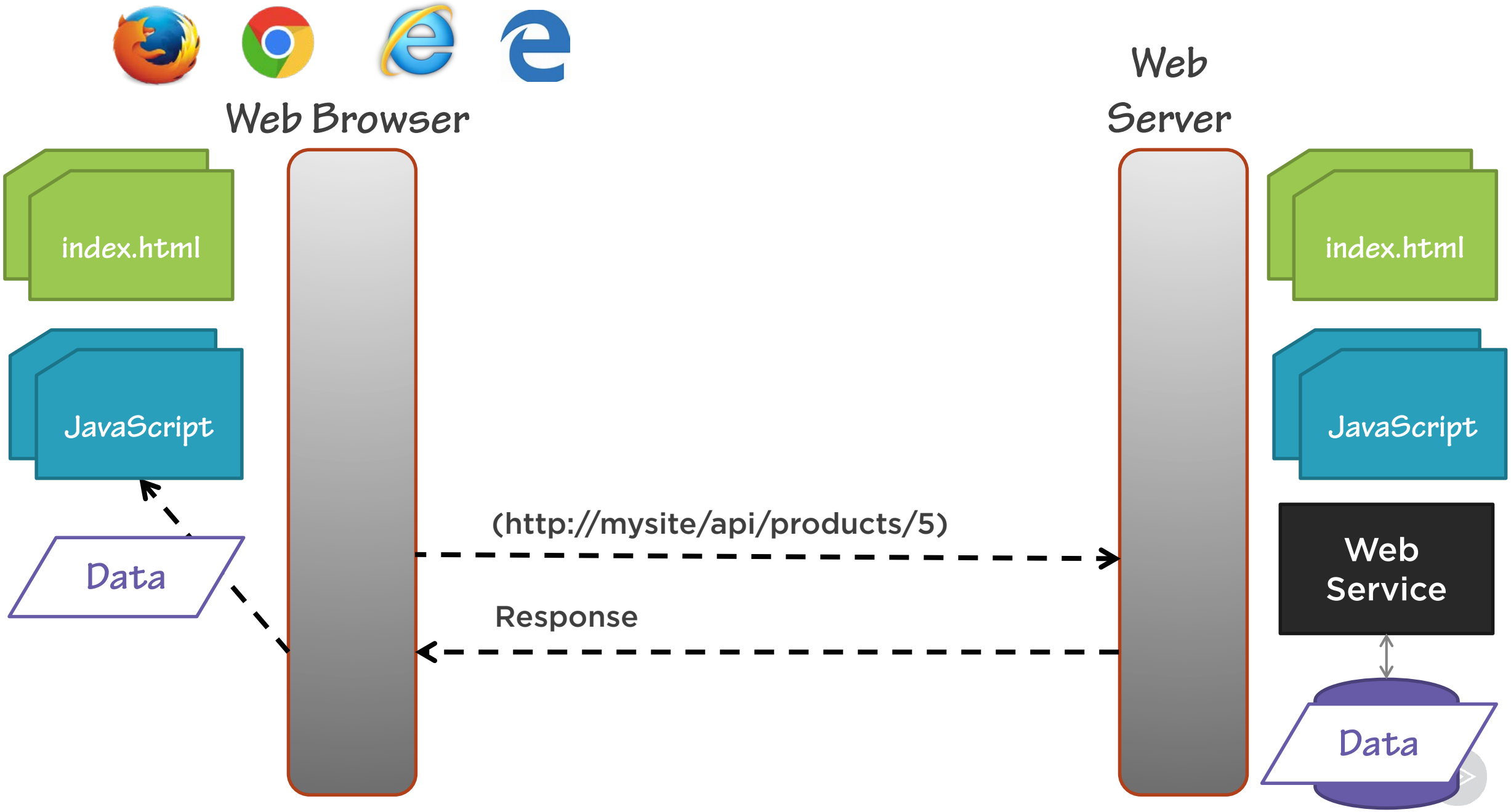
Create, Read, Update and Delete (CRUD) Using HTTP

Deborah Kurata
CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/


Web Browser
index.html
JavaScript
(http://mysite/api/products/5)
Response
Web Server
index.html
JavaScript
Web Service
Data


Web Browser
index.html
JavaScript
Data
(http://mysite/api/products/5)
Response
Web Server
index.html
JavaScript
Web Service
Data

Module Prerequisites

```
@Injectable()
export class ProductService { }
```

```
...
import { ProductService }
    from './product.service';

@NgModule({
  imports: [ ... ],
  providers: [ ProductService ]
})
export class ProductModule { }
```

```
constructor(private http: Http) { }
```

```
...
import { Observable }
    from 'rxjs/Observable';
import 'rxjs/add/operator/do';
import 'rxjs/add/operator/catch';
import 'rxjs/add/operator/throw';
import 'rxjs/add/operator/map';
...

 getProducts(): Observable<IProduct[]> {
  return this.http.get(this.baseUrl)
              .map(this.extractData);
 }
```

Module Overview

Data Access Service

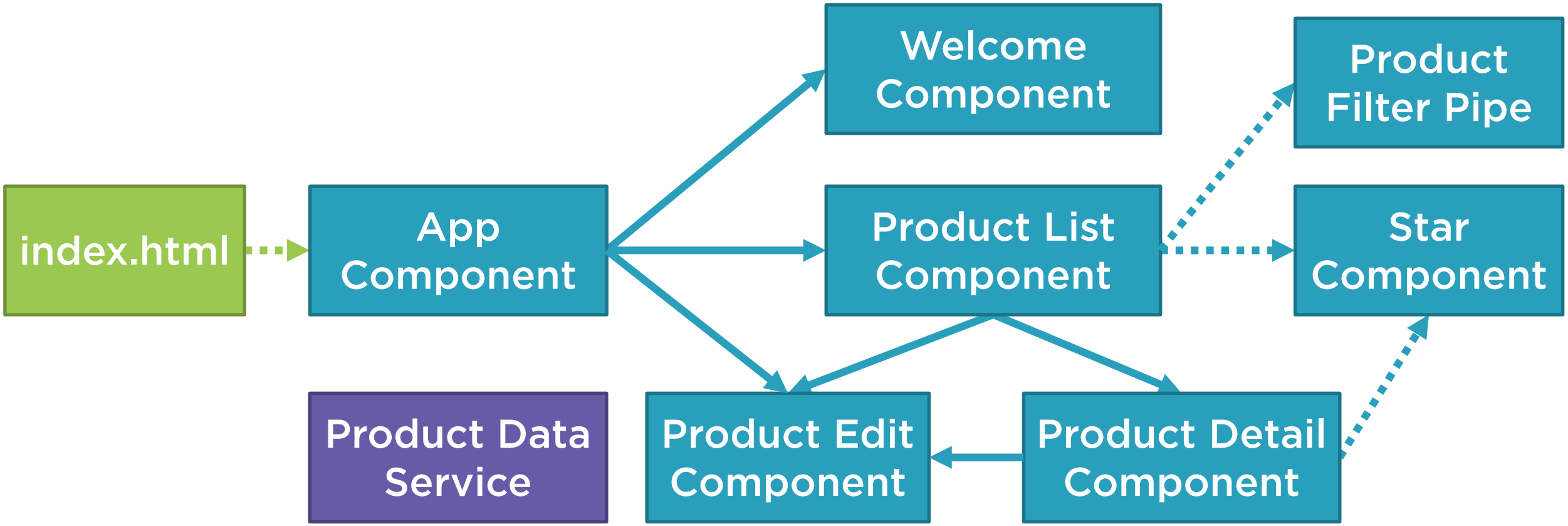
Creating Data

Reading Data

Updating Data

Deleting Data

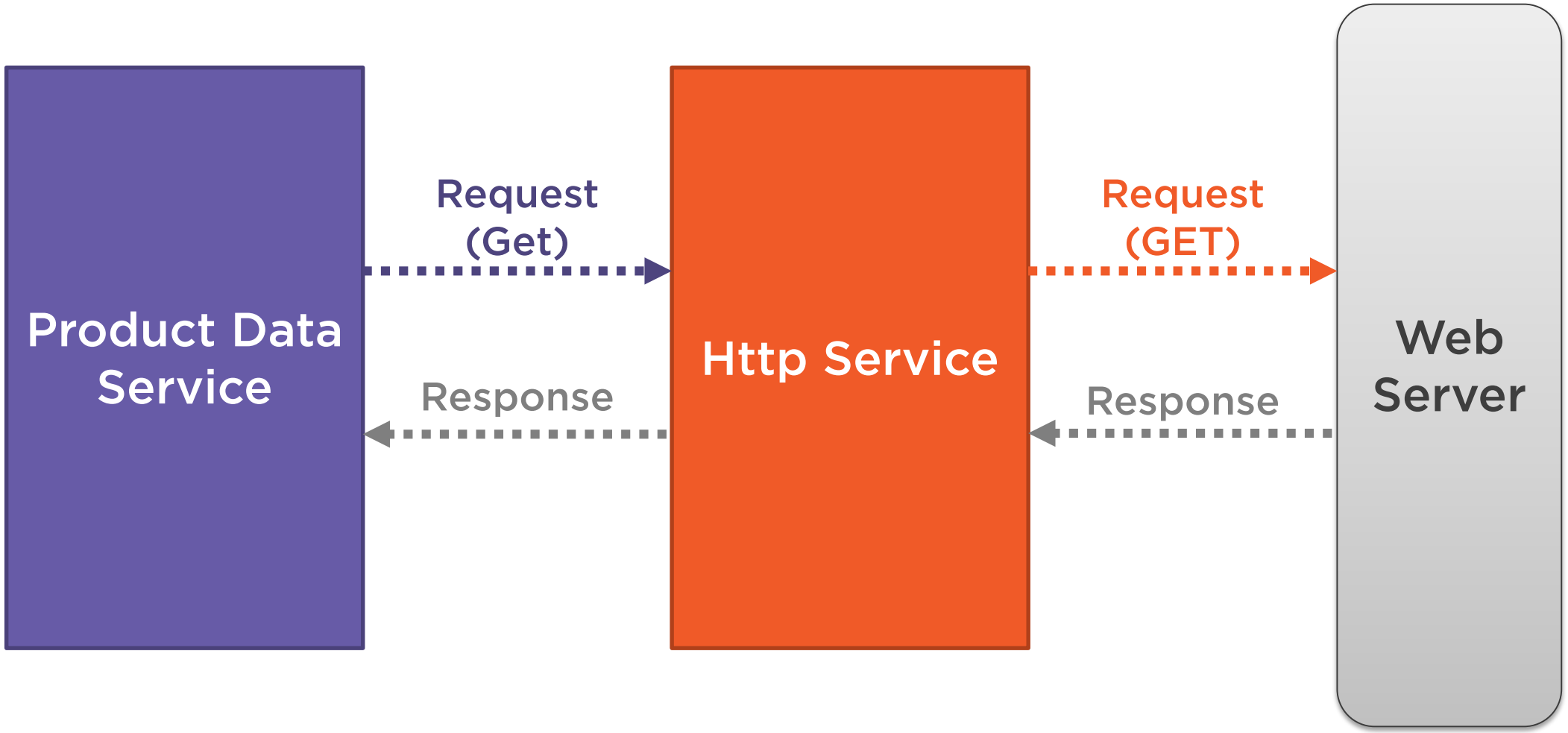
APM Sample Application Architecture



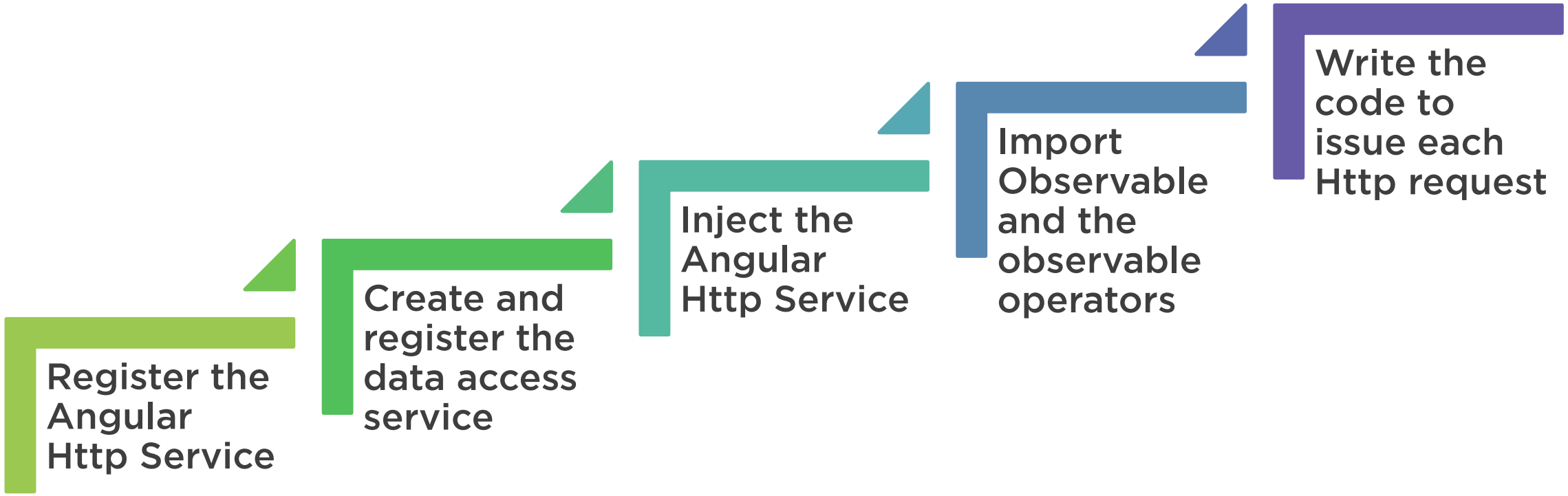
Why Build a Data Access Service?

- Separation of Concerns
- Reusability
- Data Sharing

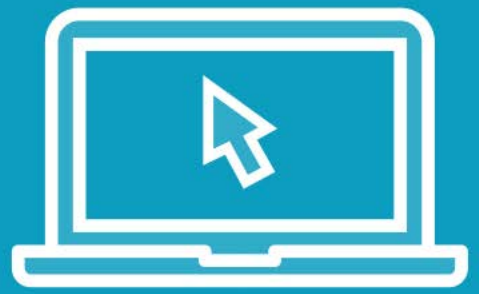
Sending an HTTP Request



Steps to Building a Data Access Service

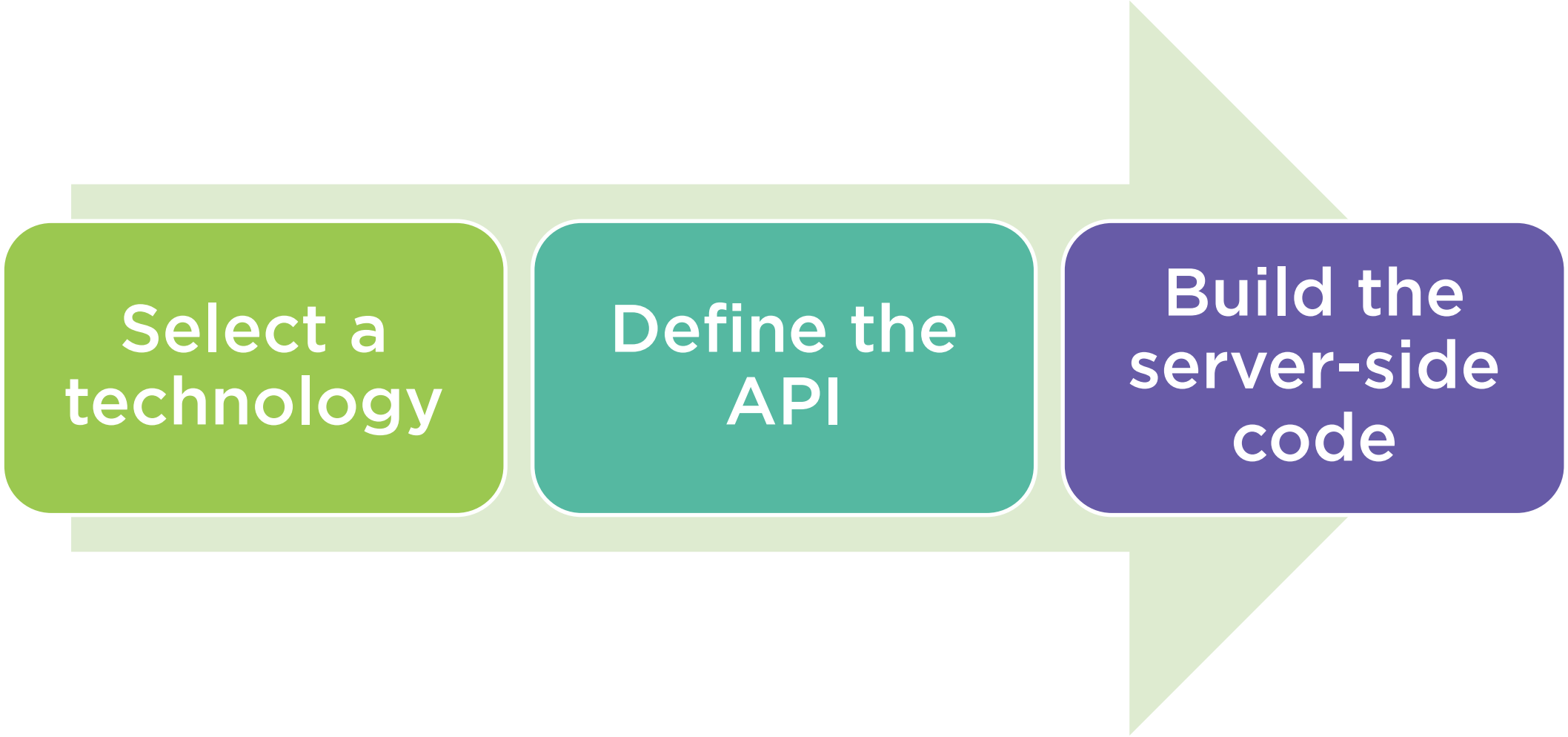


Demo



Building a Data Access Service

Setting up the Backend Server



Faking a Backend Server

Directly return hard-coded data

Use a JSON file

Write our own code using MockBackend

Use angular-in-memory-web-api

Populating the Form with Data

Sign Up!

First Name: First Name (required)

Last Name: Last Name (required)

Email: Email

Send me your catal

Address Type: Hom

Street Address 1: Street

Street Address 2: Street

City, State, Zip Code: City

Save

Edit Product: Hammer

Product Name: Hammer

Product Code: TBX-0048

Star Rating (1-5): 4.8

Add Tag

Tag: tools

Tag: hammer

Tag: construction

Description: Curved claw steel hammer

Save | Cancel | Delete

Product List

Filter by:

Hide Image

Product	Code	Available	Price	5 Star Rating	
Leaf Rake	gdn-0011	March 19, 2016	$19.95	★★★	Edit
Garden Cart	gdn-0023	March 18, 2016	$32.99	★★★★	Edit
Hammer	tbx-0048	May 21, 2016	$8.90	★★★★★	Edit
Saw	tbx-0022	May 15, 2016	$11.55	★★★½	Edit
Video Game Controller	gmg-0042	October 15, 2015	$35.95	★★★★½	Edit

HTTP Get Request

product.service.ts

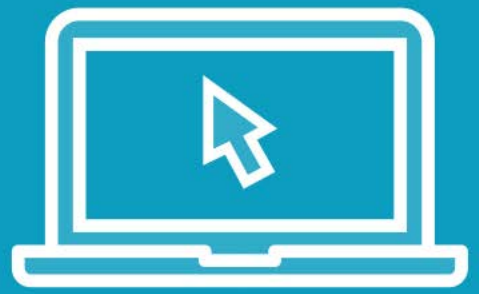
```
...
import { Http, Response} from '@angular/http';
import { Observable } from 'rxjs/Observable';
import 'rxjs/add/operator/map';

@Injectable()
export class ProductService {
  private baseUrl = 'www.myWebService.com/api/products';

  constructor(private http: Http) { }

  getProduct(id: number): Observable<IProduct> {
     const url = `${this.baseUrl}/${id}`;
     return this.http.get(url)
                 .map(this.extractData);
  }
}
```

Calling the Data Access Service

product-edit.component.ts

```
...

constructor(private productService: ProductService) { }
...

getProduct(id: number): void {
    this.productService.getProduct(id)
        .subscribe(
            (product: IProduct) => this.onProductRetrieved(product),
            (error: any) => this.errorMessage = <any>error
        );
}
```

Demo



Populating the Form with Data

Saving Edits

Edit Product: Hammer

Product Name Hammer

Product Code TBX-0048

Star Rating (1-5) 4.8

Add Tag

Tag tools

Tag hammer

Tag construction

Description Curved claw steel hammer

Save Cancel Delete

{ "productName": "Hammer", "productCode": "TBX-0048", "starRating": "4.6", "description": "Small curved claw steel hammer", "tags": ["tools", "hammer", "home maintenance"] }

Saving Edits

Edit Product: Hammer

Product Name: Hammer

Product Code: TBX-0048

Star Rating (1-5): 4.8

Add Tag

Tag: tools

Tag: hammer

Tag: construction

Description: Curved claw steel hamm

Save | Cancel | Delete

```
'id': 5,
```

```
'id': 5,
'productName': 'Hammer',
'productCode': 'TBX-0048',
'releaseDate': 'May 21, 2016',
'description': 'Small curved claw steel hammer',
'price': 8.9,
'starRating': 4.6,
'imageUrl': … ',
'tags': ['tools', 'hammer', 'home maintenance']
```

```
let p = Object.assign({}, this.product,
                          this.productForm.value);
```

{ "productName": "Hammer", "productCode": "TBX-0048", "starRating": "4.6", "description": "Small curved claw steel hammer", "tags": ["tools", "hammer", "home maintenance"] }

Post vs Put

POST (api/products)

Posts data for a resource or set of resources

Used to:

- Create a new resource when the server assigns the Id
- Update a set of resources

Not idempotent

PUT (api/products/5)

Puts data for a specific resource with an Id

Used to:

- Create a new resource when the client assigns the Id
- Update the resource with the Id

Idempotent

HTTP Put Request

product.service.ts

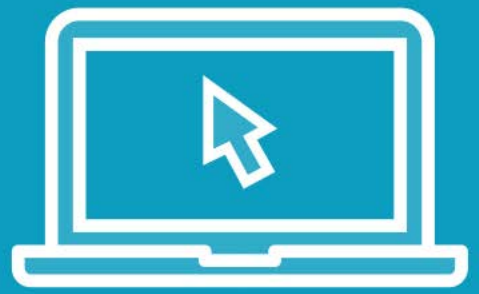
```
...

@Injectable()
export class ProductService {
  private baseUrl = 'www.myWebService.com/api/products';
  constructor(private http: Http) { }

  updateProduct(product: IProduct): Observable<IProduct> {
   let headers = new Headers({ 'Content-Type': 'application/json' });
   let options = new RequestOptions({ headers: headers });

    const url = `${this.baseUrl}/${product.id}`;
    return this.http.put(url, product, options)
                    .map(() => product);
  }
}
```

Calling the Data Access Service

product-edit.component.ts

```
editProduct: void {
    this.productService.updateProduct(p)
        .subscribe(
            () => this.onSaveComplete(),
            (error: any) => this.errorMessage = <any>error
        );
}
```

Demo



Saving Edits

Creating New Items

Acme Product Management Home Product List Add Product

Add Product

Product Name Name (required)

Product Code Code (required)

Star Rating (1-5) Rating

Add Tag

Tag Tag

Description Description

Save Cancel Delete

Initializing an Object

product.service.ts

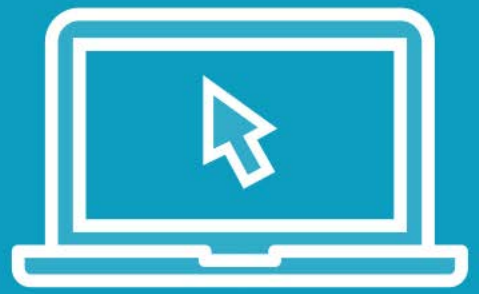
```
initializeProduct(): IProduct {
  return {
      id: 0,
      productName: null,
      productCode: null,
      tags: [''],
      releaseDate: null,
      price: null,
      description: null,
      starRating: null,
      imageUrl: null
  }
}
```

HTTP Post Request

product.service.ts

```
...

@Injectable()
export class ProductService {
  private baseUrl = 'www.myWebService.com/api/products';
  constructor(private http: Http) { }

  createProduct(product: IProduct): Observable<IProduct> {
   let headers = new Headers({ 'Content-Type': 'application/json' });
   let options = new RequestOptions({ headers: headers });

   return this.http.post(this.baseUrl, product, options)
              .map(this.extractData);
  }
}
```

Demo



Creating New Items

Deleting an Existing Item

Edit Product: Hammer

Product Name Hammer

Product Code TBX-0048

Star Rating (1-5) 4.8

Add Tag

Tag tools

Tag hammer

Tag construction

Description Curved claw steel hammer

Save Cancel Delete

Product List

Filter by:

Hide Image

Product	Code	Available	Price	5 Star Rating	
Leaf Rake	gdn-0011	March 19, 2016	$19.95	★★★	Edit
Garden Cart	gdn-0023	March 18, 2016	$32.99	★★★★	Edit
Hammer	tbx-0048	May 21, 2016	$8.90	★★★★★	Edit
Saw	tbx-0022	May 15, 2016	$11.55	★★★½	Edit
Video Game Controller	gmg-0042	October 15, 2015	$35.95	★★★★½	Edit

HTTP Delete Request

product.service.ts

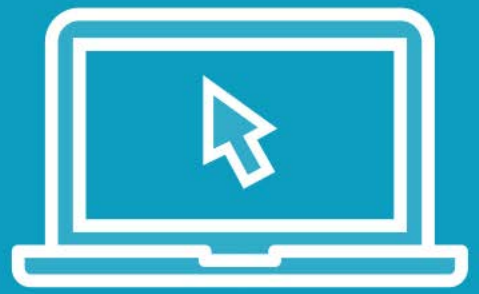
```
...

@Injectable()
export class ProductService {
  private baseUrl = 'www.myWebService.com/api/products';
  constructor(private http: Http) { }

  deleteProduct(id: number): Observable<Response> {
   let headers = new Headers({ 'Content-Type': 'application/json' });
   let options = new RequestOptions({ headers: headers });

   const url = `${this.baseUrl}/${id}`;
   return this.http.delete(url, options);
  }
}
```

Calling the Data Access Service

product-edit.component.ts

```
deleteProduct: void {
    this.productService.deleteProduct(this.product.id)
        .subscribe(
            () => this.onSaveComplete(),
            (error: any) => this.errorMessage = <any>error
        );
}
```

Demo



Deleting an Existing Item

CRUD Checklist: Register the Http Service

app.module.ts

```
...
import { HttpModule }
    from '@angular/http';

@NgModule({
  imports: [ HttpModule ],
  ...
})
export class AppModule { }
```

Add HttpModule to the imports array of one of the application's Angular Modules

CRUD Checklist: Data Access Service

Import what we need

product.service.ts

```
...
import { Http, Response} from '@angular/http';
import { Observable } from 'rxjs/Observable';
import 'rxjs/add/operator/map';
```

CRUD Checklist: Data Access Service

Import what we need

Define a dependency for the http client service

- Use a constructor parameter

product.service.ts

```
...
@Injectable()
export class ProductService {

  constructor(private http: Http) { }
}
```

CRUD Checklist: Data Access Service

product.service.ts

```
...

createProduct ...
getProduct ...
updateProduct ...
deleteProduct ...
```

Import what we need

Define a dependency for the http client service

- Use a constructor parameter

Create a method for each http request

CRUD Checklist: Data Access Service

product.service.ts

```
...

const url =
 `${this.baseUrl}/${id}`;
return this.http.get(url);
```

Import what we need

Define a dependency for the http client service

- Use a constructor parameter

Create a method for each http request

Call the desired http method, such as get

- Pass in the Url

CRUD Checklist: Data Access Service

product.service.ts

```
...

const url =
 `${this.baseUrl}/${id}`;
return this.http.get(url)
      .map(this.extractData);
```

Import what we need

Define a dependency for the http client service

- Use a constructor parameter

Create a method for each http request

Call the desired http method, such as get

- Pass in the Url

Map the Http response to a JSON object

CRUD Checklist: Data Access Service

product.service.ts

```
...

const url =
 `${this.baseUrl}/${id}`;
return this.http.get(url)
      .map(this.extractData)
    .catch(this.handleError);
```

Import what we need

Define a dependency for the http client service

- Use a constructor parameter

Create a method for each http request

Call the desired http method, such as get

- Pass in the Url

Map the Http response to a JSON object

Add error handling

CRUD Checklist: Using the Service

product-edit.component.ts

```
...

constructor(private ps: ProductService) { }
```

Inject the Data Access Service

CRUD Checklist: Using the Service

product-edit.component.ts

```
...

this.ps.getProduct(id)
 .subscribe();
```

Inject the Data Access Service

Call the subscribe method of the returned observable

CRUD Checklist: Using the Service

product-edit.component.ts

```
...

this.ps.getProduct(id)
 .subscribe(
  (product: IProduct) =>
    this.onRetrieved(product)
);
```

Inject the Data Access Service

Call the subscribe method of the returned observable

Provide a function to handle an emitted item

CRUD Checklist: Using the Service

product-edit.component.ts

```
...

this.ps.getProduct(id)
 .subscribe(
  (product: IProduct) =>
    this.onRetrieved(product),
  (error: any) =>
    this.errorMessage = error
);
```

Inject the Data Access Service

Call the subscribe method of the returned observable

Provide a function to handle an emitted item

Provide an error function to handle any returned errors

Summary

Data Access Service

Creating Data

Reading Data

Updating Data

Deleting Data