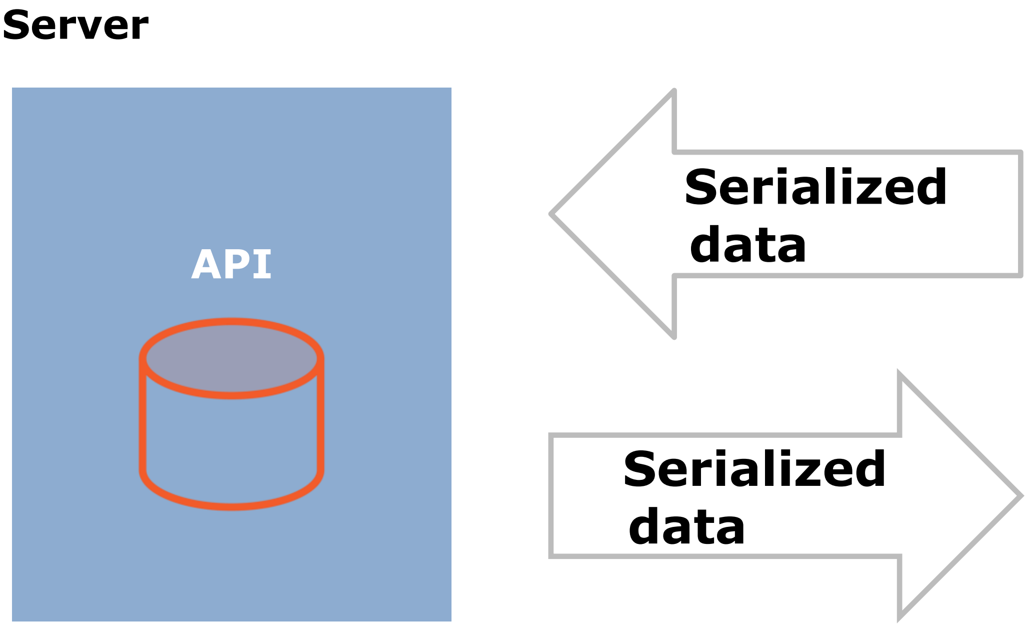
Web API Introduction

What is Web API?

- Part of ASP.NET Core
- Used to create an API (Application Programming Interface)
- The job of APIs is to enable other applications to retrieve data from it and let them manipulate the data.
- It manages the data and business rules.
- Web API applies the REST pattern to make this happen.

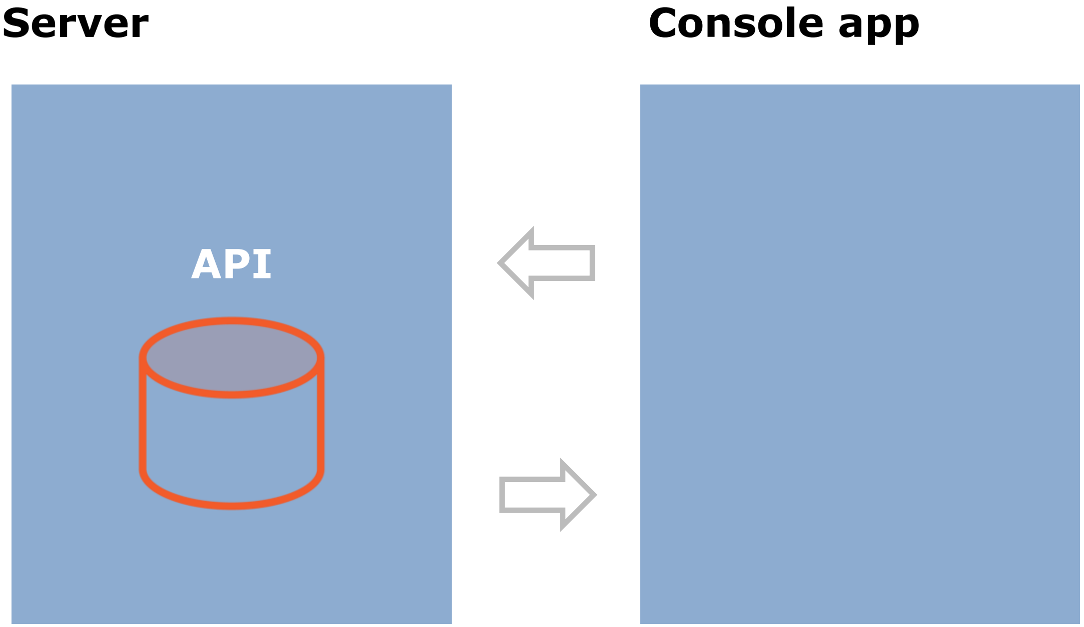
Backend API Applications



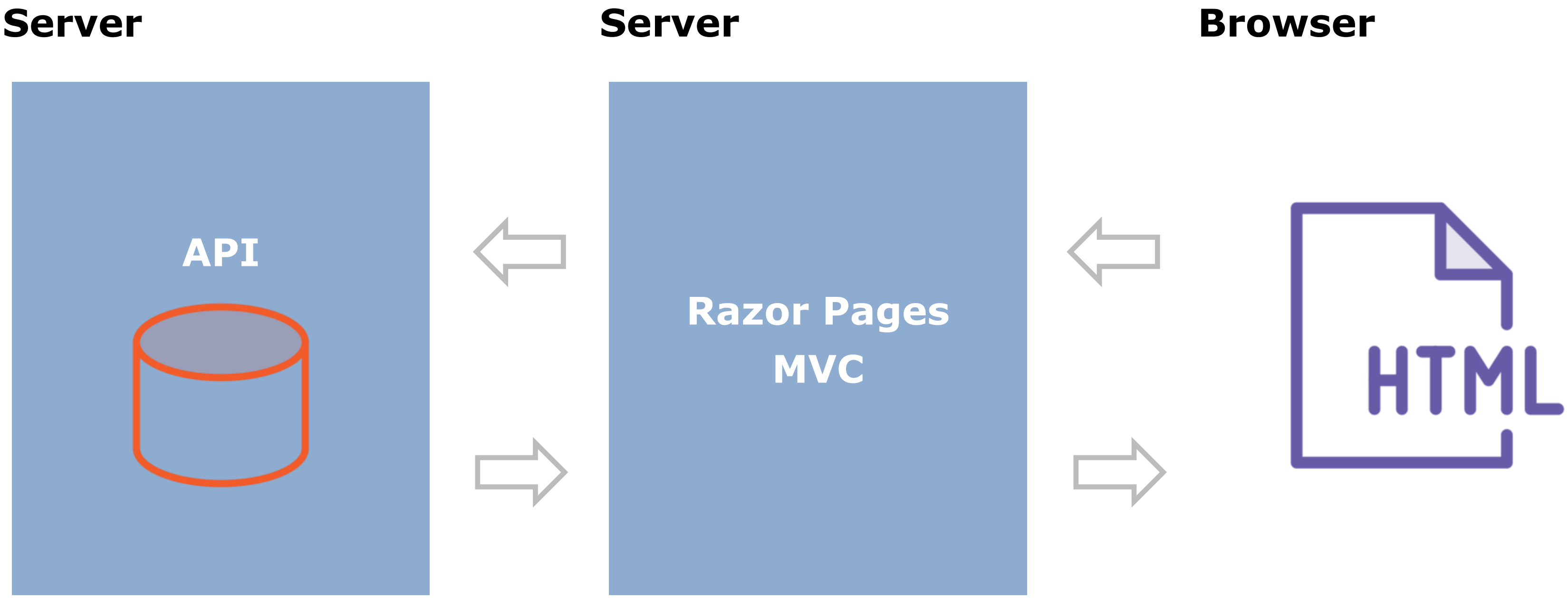
JSON Example

```
{
	"id": 1,
	"name": "Mountain Walkers",
	"price": 219.5,
	"stock": 12
}
```

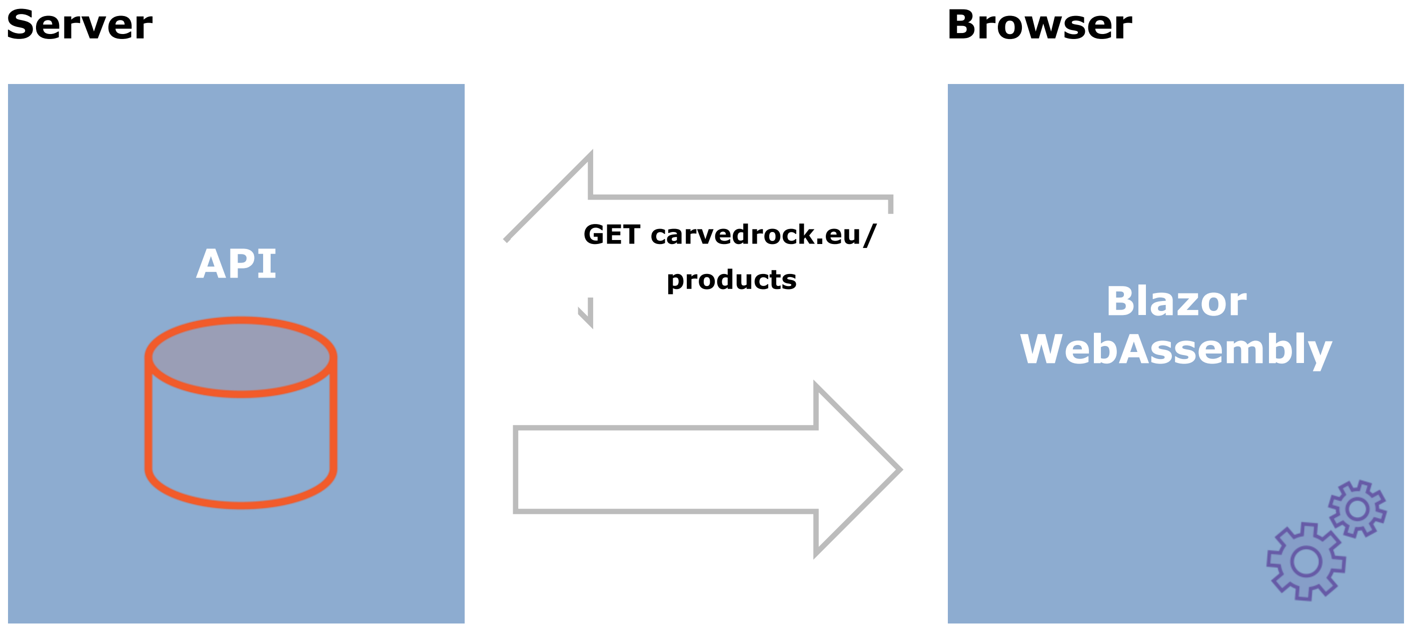
APIs With Console Applications



APIs With Server-Rendered Frontends



APIs With Client-Rendered Frontends



Demo: Adding Endpoints to An Empty Project

- Using a fairly new feature called “Minimal APIs”

Practicing Web API Basics

- Please see the AFirstEndpoint folder

- REST APIs With Web API

- Leverage HTTP protocol
- Each piece of data is available at a unique location which is called an endpoint
- HTTP methods are mapped to actions
- HTTP status codes are used to determine outcomes
- Response can also contain pointers on what to do next
- Hypermedia as the Engine of Application State (HATEOAS)

- Endpoint Examples

- https://shop.co.uk/products
- https://shop.co.uk/product/1

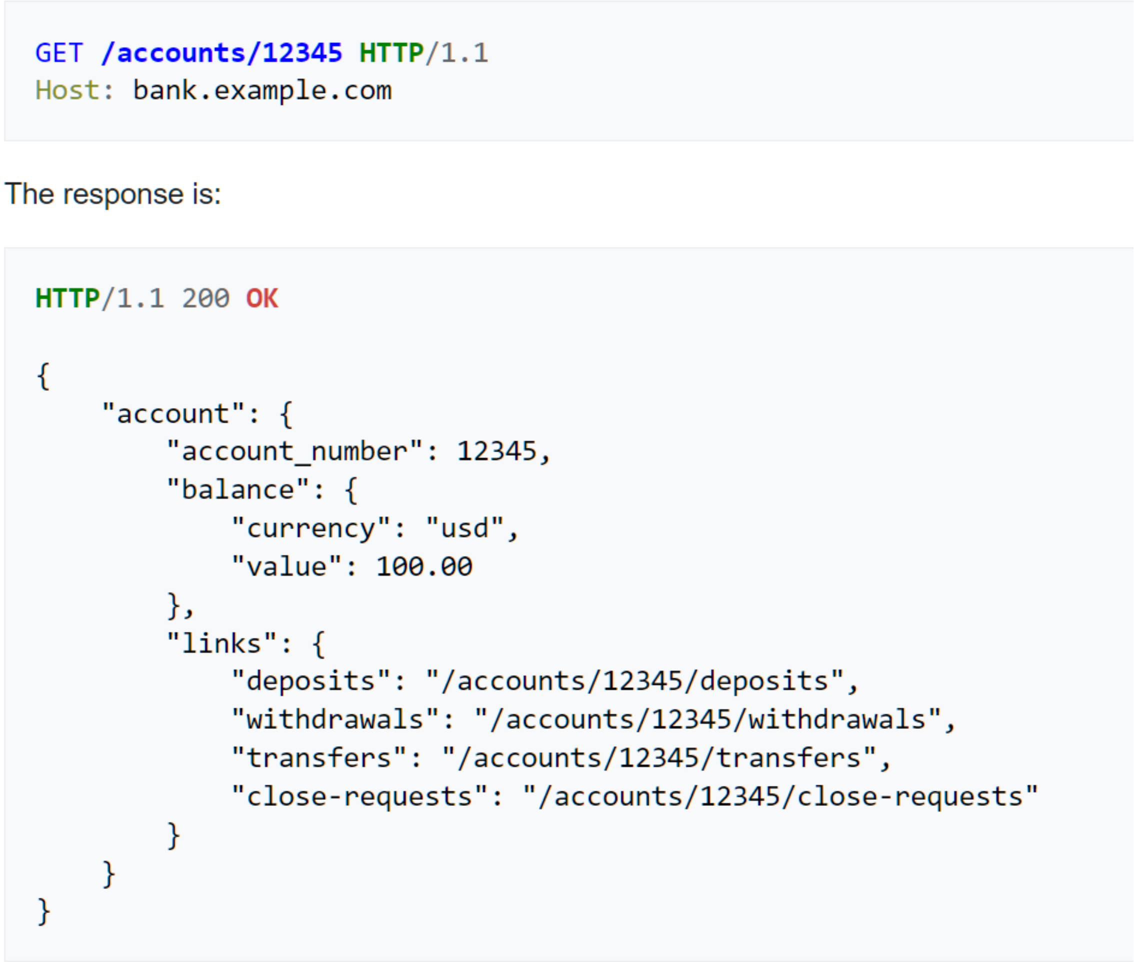
- HTTP Method Examples

- GET => https://shop.co.uk/products
- POST => https://shop.co.uk/products
- DELETE => https://shop.co.uk/products

- Status Code Examples

- 200 – OK
- 201 – Created
- 401 - Unauthorized
- 404 – Not found
- 500 – Server error

- HATEOAS Example



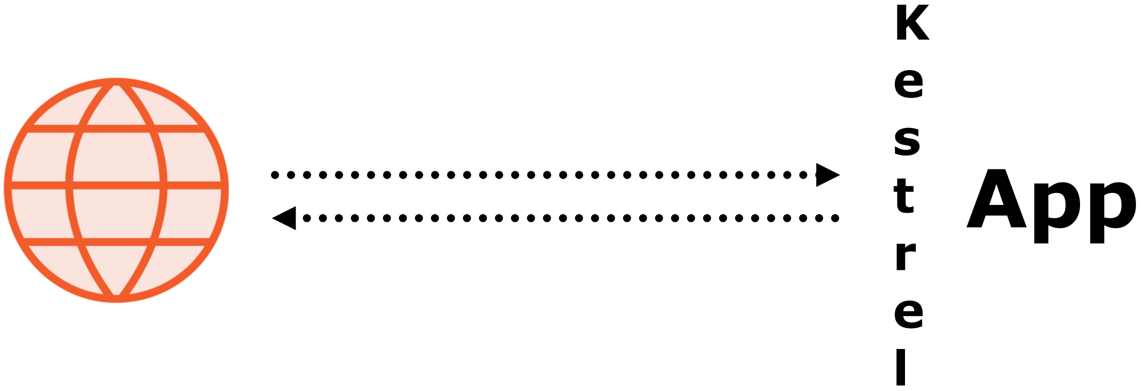
```
GET /accounts/12345 HTTP/1.1
Host: bank.example.com
```

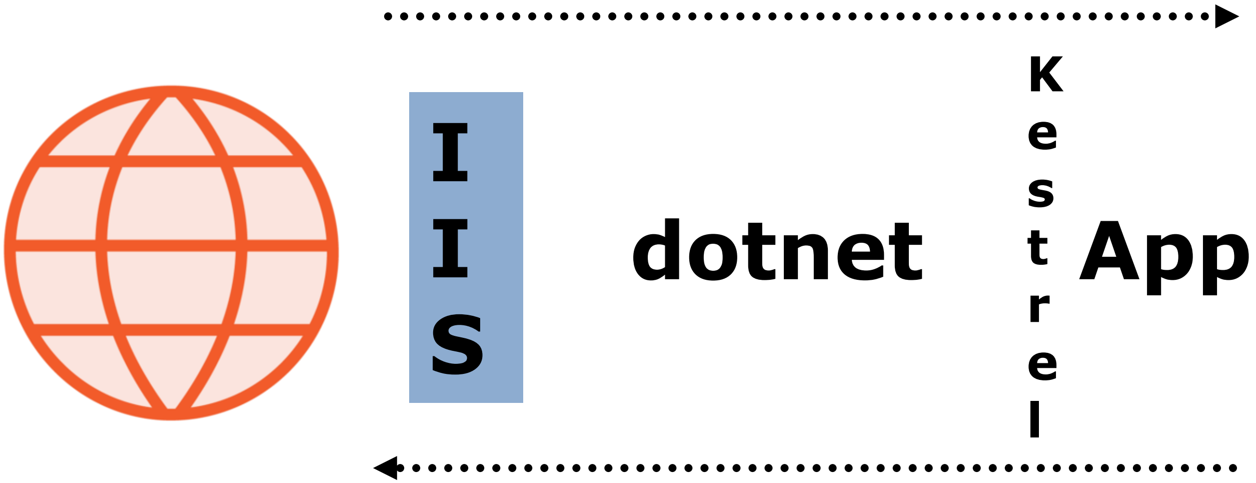
The response is:

```
HTTP/1.1 200 OK

{
    "account": {
        "account_number": 12345,
        "balance": {
            "currency": "usd",
            "value": 100.00
        },
        "links": {
            "deposits": "/accounts/12345/deposits",
            "withdrawals": "/accounts/12345/withdrawals",
            "transfers": "/accounts/12345/transfers",
            "close-requests": "/accounts/12345/close-requests"
        }
    }
}
```

The Built-In Kestrel Web Server





IIS

dotnet

Kestrel

App

Dependency Injection

Manages the lifetime of objects

Register

Instance

Object Lifetimes

Singleton

Transient

Scoped

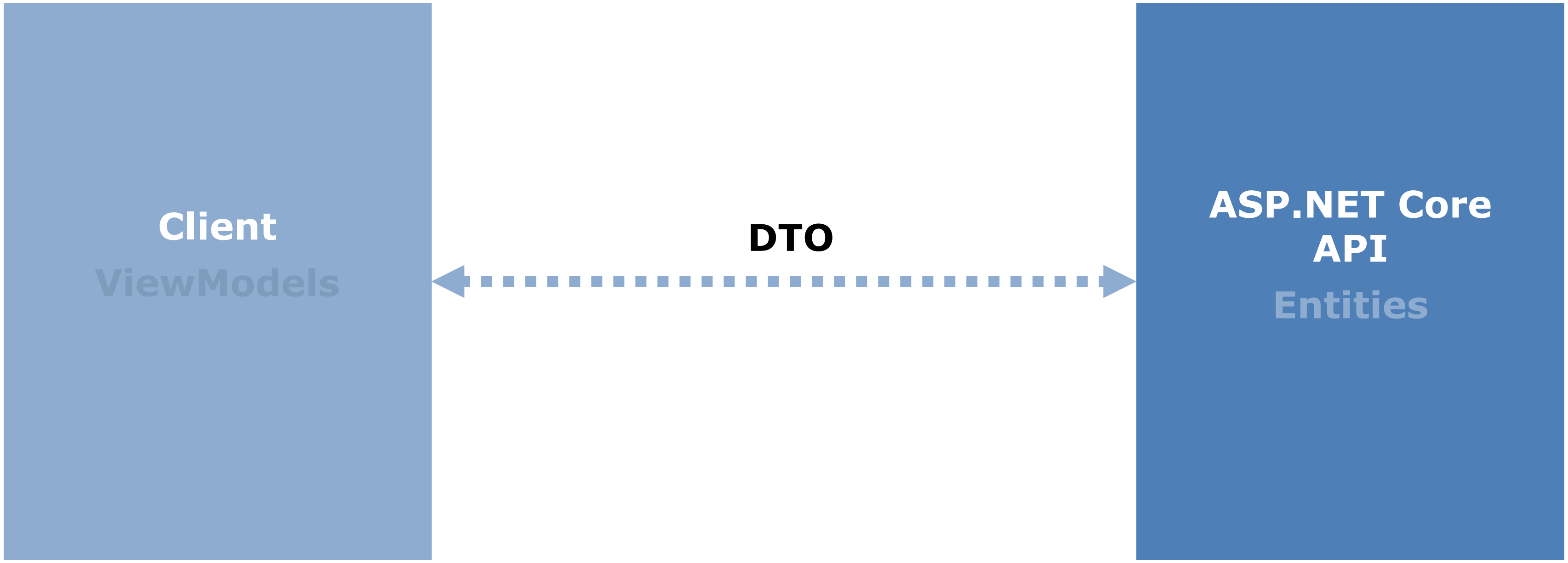
- Dependency Injection: Advantages

- No complexities around the lifetime of objects
- No "hard" dependencies between types

Practicing Dependency Injection

DependencyInjection folder

Web API and EF: Never Expose an Entity



Lab: Adding a Database

AddingADatabase folder

Route Parameters

- Routes can contain parameters
- /person/{personId}
- Can be caught as a parameter in the lambda

Working with HTTP Status Codes

- Default status codes: either 200 OK or 500 Server Error.
- Use Results type to deviate from defaults.

Lab: Adding a Parameterised Endpoint

RouteParams folder