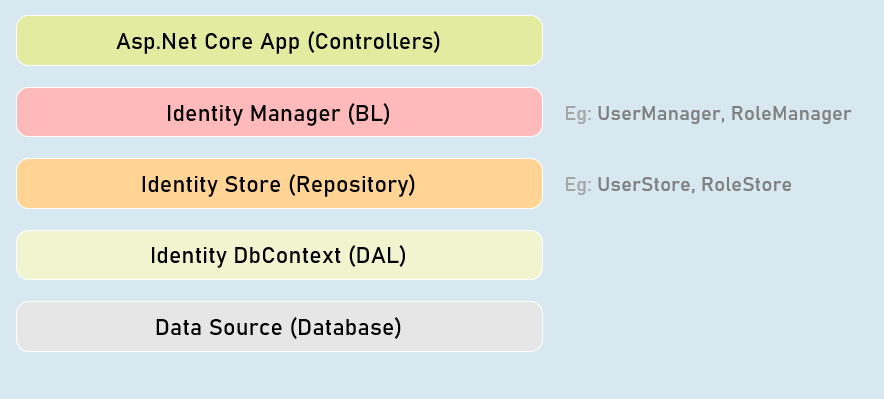
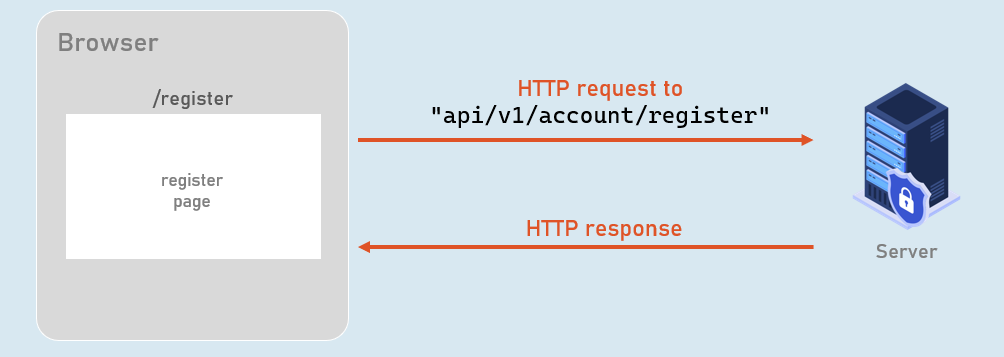
**Identity with Web API**

It is an API that manages users, passwords, profile data, roles, tokens, email confirmation, external logins etc.

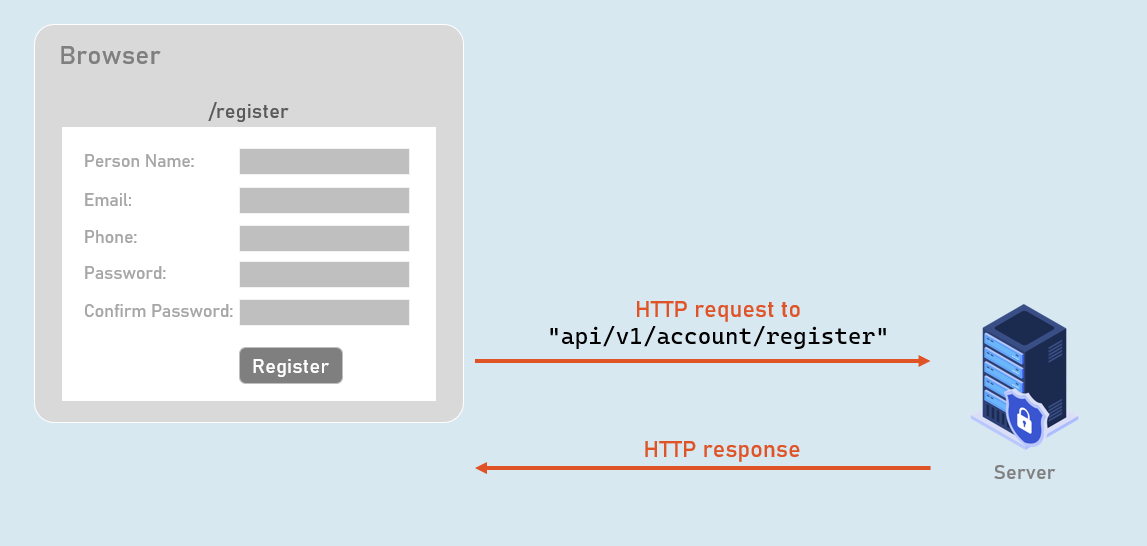
It is by default built on top of EntityFrameworkCore; you can also create custom data stores.



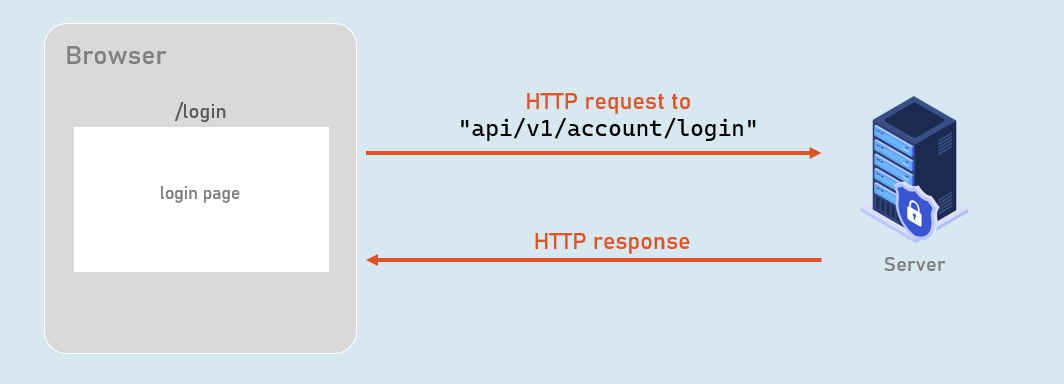
**Register Endpoint**



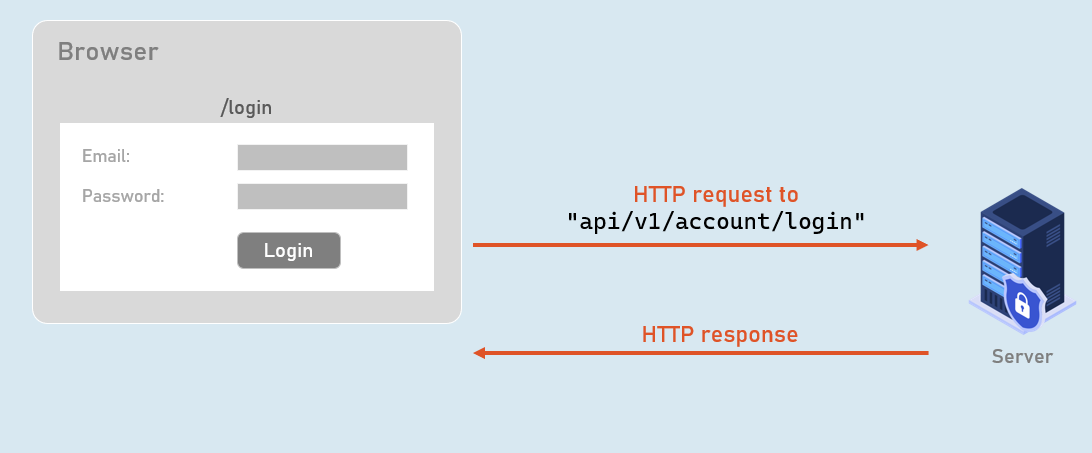
**Register UI**



**Login Endpoint**

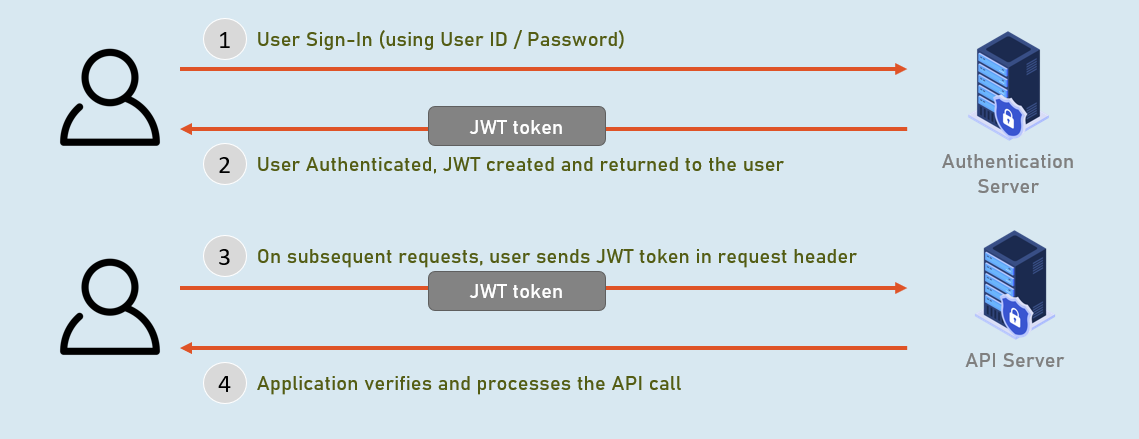


**Login UI**



**Introduction to JWT**

A JSON Web Token (JWT) is a compact and self-contained object for securely transmitting information between parties as a JSON object.



**Contents of JWT**

**1. Header (base 64 string)**

Defines the type of token and the signing algorithm used.

Eg: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9

**2. Payload (base 64 string)**

Contains user claims (user details such as name, email or user type).

Eg: { "userId": "b08f86af-35da-48f2-8fab-cef3904660bd"}

Eg: eyJ1c2VySWQiOiJiMDhmOZhZi0zNWRhLTQ4ZjItOOTA0NjYwYmQifQ

**3. Signature (base 64 string)**

It is used to verify to ensure that the message wasn't changed along the way.

It is usually signed by using a secret key (HMAC algorithm).

-xN\_h82PHVTA9vdoHrcZxH-x5

**JWT Algorithm**

Inputs:

**header**

{

"typ": "JWT",

"alg": "HS256"

}

**payload**

{

"userId": "b08f86af-35da-48f2-8fab-cef3904660bd"

}

**secret**

MySecret

**Algorithm:**

data = base64Encode( header ) + "." + base64Encode( payload )

hashedData = hash( data, secret )

signature = base64encode( hashedData )

jwtToken = data + "." + signature

**Example JWT token:**

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJiMDhmODZhZi0zNWRhLTQ4ZjItOOTA0NjYwYmQifQ.-xN\_h82PHVTCMA9vdoHrcZxH-x5mb11y1537t3rGzcM

**Refresh Tokens - JWT**

A refresh token is a token (base-64 string of a random number) that is used to obtain a new JWT token every time, when it is expired.

