ASP .Net Web APIS

Identity

TomerBu

ASP.NET Core Web API

המשך הפרוייקט משיעור קודם:

מסמך אפיון שלAPI לכרטיסיות:

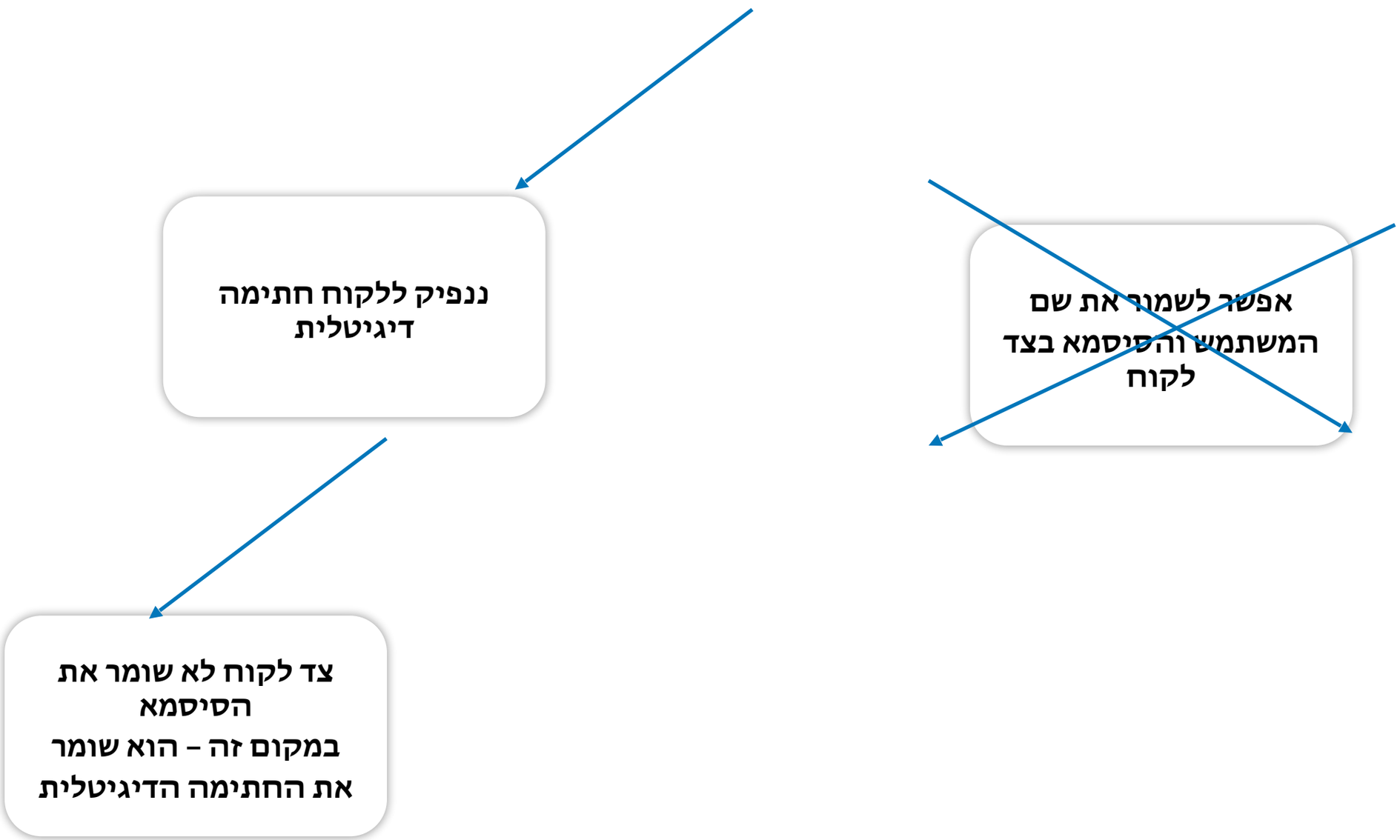
כרטיסיות:

https://documenter.getpostman.com/view/25008645/2s9YXcd5BE

משתמשים
https://documenter.getpostman.com/view/25008645/2s9YXcd5BL

שימוש בJWT
לא נרצה לבקש מהמשתמש
שם וסיסמא בכל אינטראקציה

Remember me:



שימוש בJWT
לא נרצה לבקש מהמשתמש
שם וסיסמא בכל אינטראקציה

במקום זה - ניתן ללקוח JWT

Encoded PASTE A TOKEN HERE


```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ikpva
G4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKx
wRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```


Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) ☐ secret base64 encoded
```

הנפקה של JWT
הקובץ AppSettings.JSON

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "JwtSettings": {
    "SecretKey": "This is the secret key(must be long)",
    "Issuer": "http://localhost:5000",
    "Audience": "http://localhost:500"
  },
  "ConnectionStrings": {
    "DefaultConnection": "mongodb://127.0.0.1:27017/lec3db"
  },
  "DatabaseName": "lec3db",
  "AllowedHosts": "*"
}
```

מזהה את המנפיק

מזהה למי הונפק
או לאיזו מטרה

במקום להזריק Iconfiguration אפשר להזריק מידע מ- AppSettings.JSON
עם תמיכה ב OOP

1

```
public class JWTSettings
{
        public required string SecretKey { get; set; }
        public required string Issuer { get; set; }
        public required string Audience { get; set; }

        public static JWTSettings newInstance()
        {
            return new JWTSettings()
            {
                SecretKey = "",
                Issuer = "",
                Audience = ""
            };
        }
}
```

ניצור גם factory method נשתמש בה בהמשך

2

נכריז בProgram.cs

```
builder.Services.Configure<JWTSettings>(builder.Configuration.GetSection("JWTSettings"));
```

3

```
public class JwtTokenService(
    UserManager<AppUser> userManager,
    IOptions<JWTSettings> options
  ): IJwtTokenService
{
    JWTSettings jwtSettings = options.Value;
```

כך מזריקים את זה - נראה את זה בעמוד הבא

IOptions<JWTSettings>מזריקים כ
(IConfiguration במקום)

Valueואז מחלצים ממנו את ה

הנפקה של JWT

```
public interface IJwtTokenService
{
  Task<string> CreateToken(AppUser user);
}
```

הזרקה בבנאי:

```
public class JwtTokenService(
    UserManager<AppUser> userManager,
    IOptions<JWTSettings> options
  ): IJwtTokenService
{
    JWTSettings jwtSettings = options.Value;
```

כשעובדים עם מחלקה T
מזריקים Options<T>
ואז כדי לקבל את האובייקט
משתמשים בoptions.Value

מתודה להנפקה של JWT

```
using ApisModuleLec3.Models;
using Microsoft.AspNetCore.Identity;
using System.Security.Claims;
using System.IdentityModel.Tokens.Jwt;
using Microsoft.IdentityModel.Tokens;
using System.Text;
```

```
async Task<string> IJwtTokenService.CreateToken(AppUser user)
{
    if (user is null || user.UserName is null)
        throw new ArgumentNullException(nameof(user));
    //TODO: use roles
    var isAdmin = user.IsAdmin; // await userManager.IsInRoleAsync(user, "Admin");

    List<Claim> claims =
    [
        new(JwtRegisteredClaimNames.Sub, user.UserName),
        new(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString()),
    ];
    if (isAdmin)
    {
        claims.Add(new Claim("isAdmin", "true"));
    }

    var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwtSettings.SecretKey));
    var creds = new SigningCredentials(key, SecurityAlgorithms.HmacSha512Signature);
    var token = new JwtSecurityToken(
        audience: jwtSettings.Audience,
        issuer: jwtSettings.Issuer,
        claims: claims,
        expires: DateTime.Now.AddMinutes(30),
        signingCredentials: creds
    );

    return new JwtSecurityTokenHandler().WriteToken(token);
}
```

הנפקה של JWT

הרשמה של השירות שלנו
בקובץ appsettings.json

```
// Configure JWT authentication

//our small service for generating jwt tokens:
builder.Services.AddScoped<IJwtTokenService, JwtTokenService>();
```

שימוש בAuthController
פעולת התחברות:

1 נוסיף הזרקה בבנאי של:

```
IJwtTokenService jwtTokenService
```

```
[HttpPost("login")]
public async Task<IActionResult> Login([FromBody] LoginDto dto)
{
    var user = await userManager.FindByEmailAsync(dto.Email);
    if (user == null)
    {
        return Unauthorized();
    }
    var result = await signInManager.PasswordSignInAsync(
        dto.Email,
        dto.Password,
        false,
        false
    );

    if (result.Succeeded)
    {
        var token = await jwtTokenService.CreateToken(user);
        return Ok(new {token});
    }

    return Unauthorized(result);
}
```

2 אחרי התחברות נשלח JWT

המתודה signInManager.PasswordSignInAsync
שולחת גם Cookie באופן אוטומטי. כדי להמנע מכך אפשר להשתמש ב:
userManager.CheckPasswordAsync(user, dto.Password)

```
[HttpPost("login")]
public async Task<IActionResult> Login([FromBody] LoginDto dto)
{
    var user = await userManager.FindByEmailAsync(dto.Email);
    if (user == null)
    {
        return Unauthorized();
    }

    var granted = await userManager.CheckPasswordAsync(user, dto.Password);

    if (granted)
    {
        var token = await jwtTokenService.CreateToken(user);
        return Ok(new { token });
    }

    return Unauthorized(new { message = "Invalid credentials" });
}
```

הגדרות לבדיקה של JWT ע"י הספריה Identity

```
using System.Text;
using ApisModuleLec3.Models;
using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.IdentityModel.Tokens;

namespace ApisModuleLec3.Auth
{
    public static class Utils
    {
        public static void SetupJwt(WebApplicationBuilder builder)
        {
            var jwtSettings = JWTSettings.newInstance();
            builder.Configuration.Bind("JwtSettings", jwtSettings);

            builder
                .Services.AddAuthentication(options =>
                {
                    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
                    options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
                    options.DefaultScheme = JwtBearerDefaults.AuthenticationScheme;
                })
                .AddJwtBearer(options =>
                {
                    options.TokenValidationParameters = new TokenValidationParameters
                    {
                        ValidateIssuer = true,
                        ValidateAudience = true,
                        ValidateLifetime = true,
                        ValidateIssuerSigningKey = true,
                        ValidIssuer = jwtSettings.Issuer,
                        ValidAudience = jwtSettings.Audience,
                        IssuerSigningKey = new SymmetricSecurityKey(
                            Encoding.UTF8.GetBytes(jwtSettings.SecretKey)
                        )
                    };
                });
        }
    }
}
```

חילוץ הגדרות מהקובץ appsettings

הגדרות authentication עבור JWT

הגדרות לJWT:

שימוש בהגדרות האלה בProgram.cs

```
builder.Services.AddSingleton<IRepository<Card>, CardRepository>();

ApisModuleLec3.Auth.Utils.SetupJwt(builder);

var app = builder.Build();

app.UseAuthentication();
app.UseAuthorization();
```

1 קריאה למתודה

2 הפעלה של בדיקות auth

אותנתיקציה: זיהוי המשתמש (האם המשתמש הוא מי שהוא טוען)
אותוריזציה = האם המשתמש מורשה לבצע פעולה

מתודה שמאפשרת רק למשתמש מחובר להכנס עם JWT
בתוך המתודה אפשר לבדוק את הClaims של המשתמש – האם הוא admin

הClaims מגיעים מJWT

```
[HttpPost]
[Authorize]
public async Task<IActionResult> Post([FromBody] CardAddRequest dto)
{
    if (!ModelState.IsValid)
    {
        return BadRequest("BadBad");
    }
    if (User.Claims.Any(c => c.Type == "isAdmin" && c.Value == "true"))
    {
        Console.WriteLine("Welcome, Admin");
    }
    var card = dto.ToCard("TheUserId");
    var result = await cardsRepo.AddAsync(card);

    return CreatedAtAction(nameof(Get), new { id = result.Id }, result);
}
```

בדיקה בקובץ http:

קבלת JWT:

```
### login a User:
POST {{ApisModuleLec3_HostAddress}}/api/auth/login
Content-Type: application/json

{
  "email": "john.doe@example.com",
  "password": "Abc!123Abc"
}
```

בדיקה בקובץ http:

שליחת בקשה עם Header של Authroization

```
POST {{ApisModuleLec3_HostAddress}}/api/cards
Content-Type: application/json
Authorization: bearer
eyJhbGciOiJodHRwOi8vd3d3LnczLm9yZy8yMDAxLzA0L3htbGRzaWctbW9yZSNobWFjLXNoYTUxMiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJ0b21lcmJ
1QGdtYWlsLmNvbSIsImp0aSI6ImU1NzUwOGQzLWEwNTctNGQyMy1iNzU4LTBhYWVhMDUyNzMxZiIsImh0dHA6Ly9zY2hlbWFzLm1pY3Jvc29mdC5jb20
vd3MvMjAwOC8wNi9pZGVudGl0eS9jbGFpbXMvcm9sZSI6WyJVc2VyIiwiQWRtaW4iXSwiaXNBZG1pbiI6InRydWUiLCJleHAiOjE3MjI0MzE0NTgsIml
zcyI6Imh0dHA6Ly9sb2NhbGhvc3Q6NTAwMCIsImF1ZCI6Imh0dHA6Ly9sb2NhbGhvc3Q6NTAwIn0.95KfRvIF828PEJLRHJMzIYSJGPVEl62jDr2-
va7y6GrZ3W7qIHxv63x_p8uGf1nt9EO0DFseekkSmzOX6FnuOA
```

```
{
  "title": "Pizza Card",
  "subtitle": "a pizza card",
  "description": "a test value for new card\na test value for new card\n",
  "phone": "012-3211234",
  "email": "pizza@gmail.com",
  "web": "www.pizza.com",
  "image": {
    "url": "https://img.izismile.com/img/img13/20201030/640/you_have_never_seen_something_like_this_640_36.jpg",
    "alt": "image of something"
  },
  "address": {
    "state": "IL",
    "country": "Israel",
    "city": "Arad",
    "street": "Shoham",
    "houseNumber": 5,
    "zip": 8920435
  }
}
```