**1.WebApi\_Handson**

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  // For more information on enabling Web API for empty projects, visit https://go.microsoft.com/fwlink/?LinkID=397860  namespace WebApplication1.Controllers  {  [Route("api/[controller]")]  [ApiController]  public class ValuesController : ControllerBase  {  // GET: api/<ValuesController>  [HttpGet]  public IEnumerable<string> Get()  {  return new string[] { "value1", "value2" };  }  // GET api/<ValuesController>/5  [HttpGet("{id}")]  public string Get(int id)  {  return "value";  }  // POST api/<ValuesController>  [HttpPost]  public void Post([FromBody] string value)  {  }  // PUT api/<ValuesController>/5  [HttpPut("{id}")]  public void Put(int id, [FromBody] string value)  {  }  // DELETE api/<ValuesController>/5  [HttpDelete("{id}")]  public void Delete(int id)  {  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

**2.WebApi\_Handson**

|  |
| --- |
| using Microsoft.OpenApi.Models;  var builder = WebApplication.CreateBuilder(args);  builder.Services.AddControllers();  builder.Services.AddEndpointsApiExplorer();  builder.Services.AddSwaggerGen(c =>  {  c.SwaggerDoc("v1", new OpenApiInfo  {  Title = "Swagger Demo",  Version = "v1",  Description = "TBD",  TermsOfService = new Uri("https://example.com/terms"),  Contact = new OpenApiContact  {  Name = "John Doe",  Email = "john@xyzmail.com",  Url = new Uri("https://www.example.com")  },  License = new OpenApiLicense  {  Name = "License Terms",  Url = new Uri("https://www.example.com")  }  });  });  var app = builder.Build();  if (app.Environment.IsDevelopment())  {  app.UseSwagger();  app.UseSwaggerUI(c =>  {  c.SwaggerEndpoint("/swagger/v1/swagger.json", "Swagger Demo");  });  }  app.UseHttpsRedirection();  app.UseAuthorization();  app.MapControllers();  app.Run(); |

A screenshot of a web page

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  namespace WebApplication1.Controllers  {  [Route("api/employee")]  [ApiController]  public class EmployeeController : ControllerBase  {  [HttpGet]  public IActionResult Get()  {  var employees = new[]  {  new { Id = 1, Name = "Alice", Role = "Developer" },  new { Id = 2, Name = "Bob", Role = "Tester" },  new { Id = 3, Name = "Charlie", Role = "Manager" }  };  return Ok(employees);  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  namespace WebApplication1.Controllers  {  [Route("api/emp")]  [ApiController]  public class EmployeeController : ControllerBase  {  [HttpGet]  public IActionResult Get()  {  var employees = new[]  {  new { Id = 1, Name = "Alice", Role = "Developer" },  new { Id = 2, Name = "Bob", Role = "Tester" },  new { Id = 3, Name = "Charlie", Role = "Manager" }  };  return Ok(employees);  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

**3.WebApi\_Handson**

|  |
| --- |
| using Microsoft.AspNetCore.Http;  using Microsoft.AspNetCore.Mvc;  using System;  using System.Collections.Generic;  using WebApplication1.Models;  namespace WebApplication1.Controllers  {  [Route("api/emp")]  [ApiController]  public class EmployeeController : ControllerBase  {  private List<Employee> GetStandardEmployeeList()  {  var devDepartment = new Department { Id = 1, Name = "Development" };  var hrDepartment = new Department { Id = 2, Name = "Human Resources" };  var csharpSkill = new Skill { Id = 101, Name = "C#" };  var sqlSkill = new Skill { Id = 102, Name = "SQL" };  var adminSkill = new Skill { Id = 103, Name = "Administration" };  var employees = new List<Employee>  {  new Employee  {  Id = 1,  Name = "John Doe",  Salary = 80000,  Permanent = true,  Department = devDepartment,  Skills = new List<Skill> { csharpSkill, sqlSkill },  DateOfBirth = new DateTime(1990, 5, 15)  },  new Employee  {  Id = 2,  Name = "Jane Smith",  Salary = 65000,  Permanent = true,  Department = hrDepartment,  Skills = new List<Skill> { adminSkill },  DateOfBirth = new DateTime(1992, 8, 21)  }  };  return employees;  }  [HttpGet]  [ProducesResponseType(StatusCodes.Status200OK)]  public IEnumerable<Employee> Get()  {  return GetStandardEmployeeList();  }  }  } |

A black rectangular object with white stripes

AI-generated content may be incorrect.

**CustomAuthFilter.cs**

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  using Microsoft.AspNetCore.Mvc.Filters;  using Microsoft.Extensions.Primitives;  namespace WebApplication1.Filters  {  public class CustomAuthFilter : ActionFilterAttribute  {  public override void OnActionExecuting(ActionExecutingContext context)  {  if (!context.HttpContext.Request.Headers.TryGetValue("Authorization", out StringValues authHeader))  {  context.Result = new BadRequestObjectResult("Invalid request - No Auth token");  return;  }  if (!authHeader.ToString().StartsWith("Bearer "))  {  context.Result = new BadRequestObjectResult("Invalid request - Token present but Bearer unavailable");  return;  }  base.OnActionExecuting(context);  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**CustomExeceptionFilter.cs**

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  using Microsoft.AspNetCore.Mvc.Filters;  using Microsoft.AspNetCore.Mvc.WebApiCompatShim; // Required for ExceptionResult  using System.IO;  using System.Web.Http;  namespace WebApplication1.Filters  {  public class CustomExceptionFilter : IExceptionFilter  {  public void OnException(ExceptionContext context)  {  string exceptionMessage = $"[Timestamp: {System.DateTime.Now}]" +  $"\nException Type: {context.Exception.GetType().Name}" +  $"\nMessage: {context.Exception.Message}" +  $"\nStackTrace: {context.Exception.StackTrace}\n\n";  File.AppendAllText("exception\_log.txt", exceptionMessage);  context.Result = new ExceptionResult(context.Exception, includeErrorDetail: true);  context.ExceptionHandled = true;  }  }  } |

A black rectangular object with green text

AI-generated content may be incorrect.

A close-up of a text

AI-generated content may be incorrect.

**4.WebApi\_Handson**

|  |
| --- |
| using Microsoft.AspNetCore.Http;  using Microsoft.AspNetCore.Mvc;  using System;  using System.Collections.Generic;  using WebApplication1.Models;  using WebApplication1.Filters;  namespace WebApplication1.Controllers  {  [Route("api/emp")]  [ApiController]  //[CustomAuthFilter]  public class EmployeeController : ControllerBase  {  private List<Employee> GetStandardEmployeeList()  {  var devDepartment = new Department { Id = 1, Name = "Development" };  var hrDepartment = new Department { Id = 2, Name = "Human Resources" };  var csharpSkill = new Skill { Id = 101, Name = "C#" };  var sqlSkill = new Skill { Id = 102, Name = "SQL" };  var adminSkill = new Skill { Id = 103, Name = "Administration" };  var employees = new List<Employee>  {  new Employee  {  Id = 1,  Name = "John Doe",  Salary = 80000,  Permanent = true,  Department = devDepartment,  Skills = new List<Skill> { csharpSkill, sqlSkill },  DateOfBirth = new DateTime(1990, 5, 15)  },  new Employee  {  Id = 2,  Name = "Jane Smith",  Salary = 65000,  Permanent = true,  Department = hrDepartment,  Skills = new List<Skill> { adminSkill },  DateOfBirth = new DateTime(1992, 8, 21)  }  };  return employees;  }  [HttpGet]  [ProducesResponseType(StatusCodes.Status200OK)]  [ProducesResponseType(StatusCodes.Status500InternalServerError)]  public IEnumerable<Employee> Get()  {  return GetStandardEmployeeList();  }  [HttpPut("{id}")]  public ActionResult<Employee> Put(int id, [FromBody] Employee updatedEmployee)  {    if (id <= 0)  {  return BadRequest("Invalid employee id");  }  var employeeList = GetStandardEmployeeList();  var employeeToUpdate = employeeList.FirstOrDefault(e => e.Id == id);  if (employeeToUpdate == null)  {  return BadRequest("Invalid employee id");  }  employeeToUpdate.Name = updatedEmployee.Name;  employeeToUpdate.Salary = updatedEmployee.Salary;  employeeToUpdate.Permanent = updatedEmployee.Permanent;  return Ok(employeeToUpdate);  }  }  } |

A screenshot of a video

AI-generated content may be incorrect.

A black and white striped background

AI-generated content may be incorrect.

**5.WebApi\_Handson**

**1.JsonWebToken**

**AuthController.cs**

|  |
| --- |
| namespace WebApplication1.Controllers  {  [Route("api/[controller]")]  [ApiController]  public class AuthController : ControllerBase  {  private string GenerateJSONWebToken(int userId, string userRole)  {  var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("this-is-my-super-long-and-secure-secret-key-for-jwt-256"));  var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);  var claims = new List<Claim>  {  new Claim(ClaimTypes.Role, userRole),  new Claim("UserId", userId.ToString())  };  var token = new JwtSecurityToken(  issuer: "mySystem",  audience: "myUsers",  claims: claims,  expires: DateTime.Now.AddMinutes(10),  signingCredentials: credentials);  return new JwtSecurityTokenHandler().WriteToken(token);  }  [HttpGet]  [AllowAnonymous]  public IActionResult Get()  {    var token = GenerateJSONWebToken(userId: 1, userRole: "Admin");  return Ok(new { token });  }  }  } |

**Program.cs**

|  |
| --- |
| var builder = WebApplication.CreateBuilder(args);  builder.Services.AddControllers(options =>  {  options.Filters.Add<CustomExceptionFilter>();  });  builder.Services.AddEndpointsApiExplorer();  builder.Services.AddSwaggerGen(c =>  {  c.SwaggerDoc("v1", new OpenApiInfo  {  Title = "Swagger Demo",  Version = "v1",  Description = "TBD",  TermsOfService = new Uri("https://example.com/terms"),  Contact = new OpenApiContact  {  Name = "John Doe",  Email = "john@xyzmail.com",  Url = new Uri("https://www.example.com")  },  License = new OpenApiLicense  {  Name = "License Terms",  Url = new Uri("https://www.example.com")  }  });  });  string securityKey = "this-is-my-super-long-and-secure-secret-key-for-jwt-256";  var symmetricSecurityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(securityKey));  builder.Services.AddAuthentication(x =>  {  x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;  x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;  x.DefaultSignInScheme = JwtBearerDefaults.AuthenticationScheme;  })  .AddJwtBearer(JwtBearerDefaults.AuthenticationScheme, x =>  {  x.TokenValidationParameters = new TokenValidationParameters  {  ValidateIssuer = true,  ValidateAudience = true,  ValidateLifetime = true,  ValidateIssuerSigningKey = true,  ValidIssuer = "mySystem",  ValidAudience = "myUsers",  IssuerSigningKey = symmetricSecurityKey  };  });  var app = builder.Build();  if (app.Environment.IsDevelopment())  {  app.UseSwagger();  app.UseSwaggerUI(c =>  {  c.SwaggerEndpoint("/swagger/v1/swagger.json", "Swagger Demo");  });  }  app.UseHttpsRedirection();  app.UseAuthentication();  app.UseAuthorization();  app.MapControllers();  app.Run(); |

A screenshot of a computer

AI-generated content may be incorrect.

1. **Use the JWT generated thru the AuthController to be used in POSTMAN request.**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Check for JWT expiration**

**AuthController.cs**

|  |
| --- |
| amespace WebApplication1.Controllers  {  [Route("api/[controller]")]  [ApiController]  public class AuthController : ControllerBase  {  private string GenerateJSONWebToken(int userId, string userRole)  {  var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("this-is-my-super-long-and-secure-secret-key-for-jwt-256"));  var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);  var claims = new List<Claim>  {  new Claim(ClaimTypes.Role, userRole),  new Claim("UserId", userId.ToString())  };  var token = new JwtSecurityToken(  issuer: "mySystem",  audience: "myUsers",  claims: claims,  expires: DateTime.Now.AddMinutes(2),  signingCredentials: credentials);  return new JwtSecurityTokenHandler().WriteToken(token);  }  [HttpGet]  [AllowAnonymous]  public IActionResult Get()  {    var token = GenerateJSONWebToken(userId: 1, userRole: "Admin");  return Ok(new { token });  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Add the roles to be authorized in the Authorize attribute.**

**EmployeeController.cs**

|  |
| --- |
| namespace WebApplication1.Controllers  {  [Route("api/emp")]  [ApiController]  //[CustomAuthFilter]  [Authorize(Roles ="POC")]  public class EmployeeController : ControllerBase  {  private List<Employee> GetStandardEmployeeList()  {  var devDepartment = new Department { Id = 1, Name = "Development" };  var hrDepartment = new Department { Id = 2, Name = "Human Resources" };  var csharpSkill = new Skill { Id = 101, Name = "C#" };  var sqlSkill = new Skill { Id = 102, Name = "SQL" };  var adminSkill = new Skill { Id = 103, Name = "Administration" };  var employees = new List<Employee>  {  new Employee  {  Id = 1,  Name = "John Doe",  Salary = 80000,  Permanent = true,  Department = devDepartment,  Skills = new List<Skill> { csharpSkill, sqlSkill },  DateOfBirth = new DateTime(1990, 5, 15)  },  new Employee  {  Id = 2,  Name = "Jane Smith",  Salary = 65000,  Permanent = true,  Department = hrDepartment,  Skills = new List<Skill> { adminSkill },  DateOfBirth = new DateTime(1992, 8, 21)  }  };  return employees;  }  [HttpGet]  [ProducesResponseType(StatusCodes.Status200OK)]  [ProducesResponseType(StatusCodes.Status500InternalServerError)]  public IEnumerable<Employee> Get()  {  return GetStandardEmployeeList();  }  [HttpPut("{id}")]  public ActionResult<Employee> Put(int id, [FromBody] Employee updatedEmployee)  {    if (id <= 0)  {  return BadRequest("Invalid employee id");  }  var employeeList = GetStandardEmployeeList();  var employeeToUpdate = employeeList.FirstOrDefault(e => e.Id == id);  if (employeeToUpdate == null)  {  return BadRequest("Invalid employee id");  }  employeeToUpdate.Name = updatedEmployee.Name;  employeeToUpdate.Salary = updatedEmployee.Salary;  employeeToUpdate.Permanent = updatedEmployee.Permanent;  return Ok(employeeToUpdate);  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

**EmployeeController.cs**

|  |
| --- |
| namespace WebApplication1.Controllers  {  [Route("api/emp")]  [ApiController]  //[CustomAuthFilter]  [Authorize(Roles ="Admin,POC")]  public class EmployeeController : ControllerBase  {  private List<Employee> GetStandardEmployeeList()  {  var devDepartment = new Department { Id = 1, Name = "Development" };  var hrDepartment = new Department { Id = 2, Name = "Human Resources" };  var csharpSkill = new Skill { Id = 101, Name = "C#" };  var sqlSkill = new Skill { Id = 102, Name = "SQL" };  var adminSkill = new Skill { Id = 103, Name = "Administration" };  var employees = new List<Employee>  {  new Employee  {  Id = 1,  Name = "John Doe",  Salary = 80000,  Permanent = true,  Department = devDepartment,  Skills = new List<Skill> { csharpSkill, sqlSkill },  DateOfBirth = new DateTime(1990, 5, 15)  },  new Employee  {  Id = 2,  Name = "Jane Smith",  Salary = 65000,  Permanent = true,  Department = hrDepartment,  Skills = new List<Skill> { adminSkill },  DateOfBirth = new DateTime(1992, 8, 21)  }  };  return employees;  }  [HttpGet]  [ProducesResponseType(StatusCodes.Status200OK)]  [ProducesResponseType(StatusCodes.Status500InternalServerError)]  public IEnumerable<Employee> Get()  {  return GetStandardEmployeeList();  }  [HttpPut("{id}")]  public ActionResult<Employee> Put(int id, [FromBody] Employee updatedEmployee)  {    if (id <= 0)  {  return BadRequest("Invalid employee id");  }  var employeeList = GetStandardEmployeeList();  var employeeToUpdate = employeeList.FirstOrDefault(e => e.Id == id);  if (employeeToUpdate == null)  {  return BadRequest("Invalid employee id");  }  employeeToUpdate.Name = updatedEmployee.Name;  employeeToUpdate.Salary = updatedEmployee.Salary;  employeeToUpdate.Permanent = updatedEmployee.Permanent;  return Ok(employeeToUpdate);  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.