# Korisni linkovi

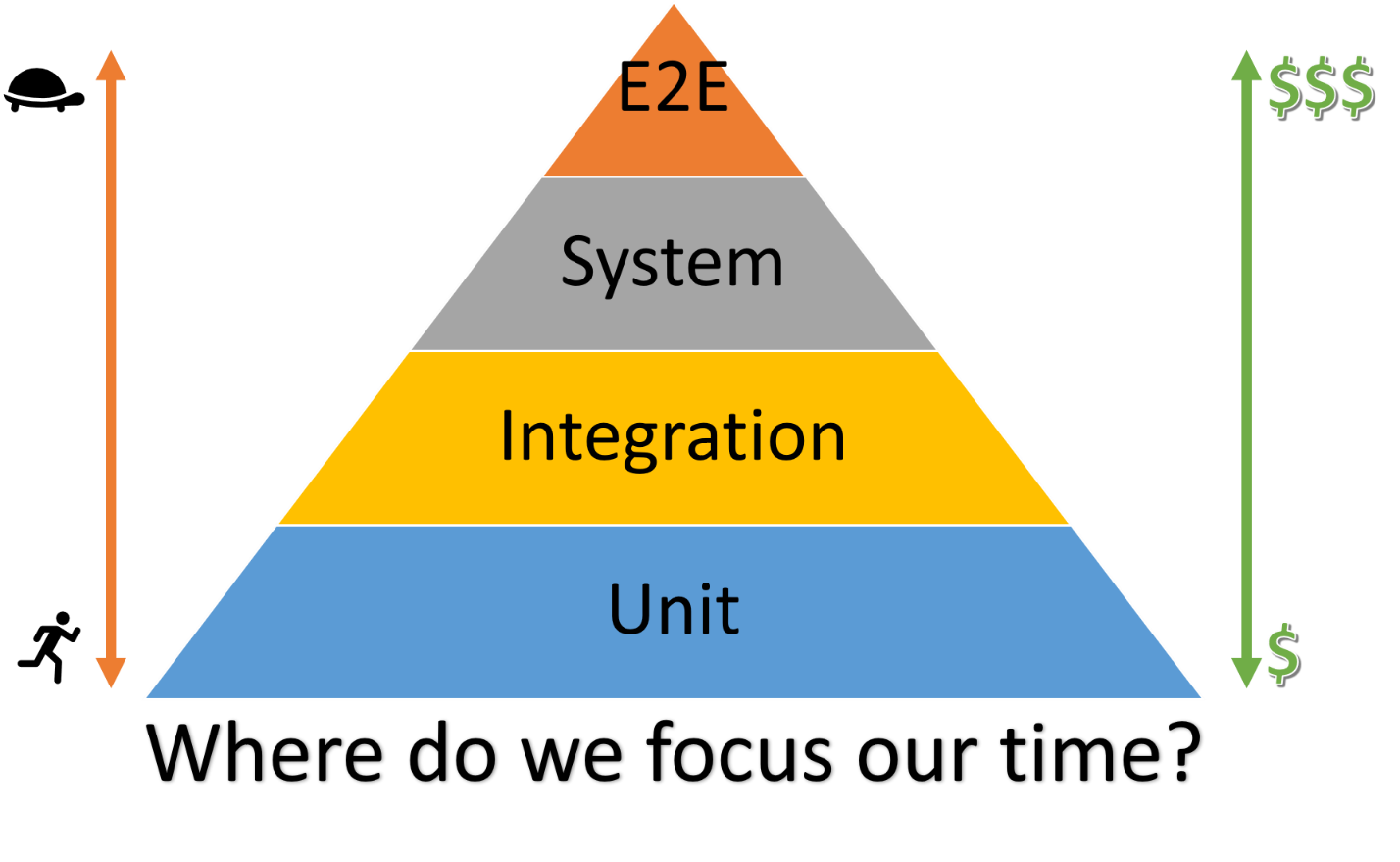
<https://www.nuget.org/packages/Moq/> - Framework za mogovanje

<https://xunit.net/> - Framework za testiranje .net aplikacija

<https://fluentassertions.com/> - Biblioteka za lakse deklarisanje assertion-a

<https://testcontainers.com/>

# Postoje nekoliko vrsta testiranja



# E2E(End to end) testiranje

## Implementacija factory-a za API

|  |
| --- |
| internal class BackendApiWebApplicationFactory: WebApplicationFactory<Program>  {  protected override void ConfigureWebHost(IWebHostBuilder builder)  {  base.ConfigureWebHost(builder);  }  }  internal class WebAppBroker  {  private static BackendApiWebApplicationFactory backendApiWebApplicationFactory;  private static HttpClient httpClient;  private static readonly object padlock = new object();  private WebAppBroker() { }  public static BackendApiWebApplicationFactory GetWebAppInstance()  {  lock (padlock)  {  if (backendApiWebApplicationFactory == null)  {  backendApiWebApplicationFactory = new BackendApiWebApplicationFactory();  }  return backendApiWebApplicationFactory;  }  }  public static HttpClient GetHttpClient()  {  lock (padlock)  {  if (httpClient == null)  {  var factory = GetWebAppInstance();  httpClient = factory.CreateClient();  }  return httpClient;  }  }  } |

## Implementacija testne klase

U xunit u svaka testna klasa se izvrasava u paraleli sa ostali klasama dok se unutrasnji test case-ovi izvrsavaju redno. Konstruktor klase se izvrsava za svaki test pojedinacno.

|  |
| --- |
| public class AuthControllerTests  {  public AuthControllerTests()  {  \_httpClient = WebAppBroker.GetHttpClient();  \_appConfig = WebAppBroker.GetWebAppInstance().Services.GetService<IOptions<AppConfigClass>>().Value;  \_validJWT = JWToken.CreateToken(adminUser, \_appConfig.JWTSettings);  \_testConfig = TestConfigProvider.GetTestConfig();  }  [Fact]  public async void RegistrationSuccess()  {  RegistrationDTO registrationDTO = new RegistrationDTO()  {  Email = adminUser.Email,  Lastname = adminUser.Lastname,  Name = adminUser.Name,  Password = "string",  PhoneNumber = adminUser.PhoneNumber,  WorkerType = adminUser.WorkerType  };  var request = new HttpRequestMessage(HttpMethod.Post, \_testConfig.BaseApiURL + "/Auth/Register")  {  Content = new StringContent(  JsonConvert.SerializeObject(registrationDTO),  Encoding.UTF8,  "application/json")  };  request.Headers.Authorization = new AuthenticationHeaderValue("Bearer", \_validJWT);  // Make the request  var response = await \_httpClient.SendAsync(request);  var responseData = response.Content.ReadFromJsonAsync<ServiceResponse<bool>>().Result;  response.StatusCode.Should().Be(System.Net.HttpStatusCode.OK);  responseData.Data.Should().BeTrue();  } |