**به نام خدا**

**جلسه اول**

به بهانه یک دوره جدید، می‌خواهم کمی تیراندازی کنم! پیشاپیش عذرخواهی می‌کنم!

عنوان دوره جدید:

ASP.NET Core Web API + Blazor Web Assembly **Custom Security**

تفاوت زبان‌های برنامه‌نویسی #C و Python و معرفی استراتژی یادگیری آن‌ها

معرفی Road Map برای NET. کارها

**یک سری مباحث جامعه‌شناسی!**

محدودیت LinkedIn!

اگر نمونه‌گیری بیش از ۱۰۰۰ واحد باشد، یک نمونه‌گیری نرمال بوده و تا حد زیادی، قابل اتکاء و اعتماد می‌باشد.

در طی ۳۳ سال، بیش از شانزده هزار نفر شاگرد داشته‌ام.

**ویژگی مردم ایران به صورت نرم یا نرمال**

**نکته:** ما ایرانی‌های بی‌اندازه! ویژگی‌های مثبت و قابل تحسین داریم! ولی اگر، کمی نقاط ضعف داریم، شایسته است که در صورت امکان این نقاط ضعف را اصلاح کنیم!

ما ایرانی‌ها، **کم و بیش**، مردمانی با **ضریب هوشی** متوسط بوده، که معمولاً بدون اندیشه و تحلیل (**دیدگاه فلسفی**)، **متعصبانه** و غیر منصفانه، نسبت به عقاید خود پافشاری می‌کنیم، و البته می‌توان با کار **Social Engineering**، ما را جوگیر کرده و نظرات / ایدئولوژی‌مان را تغییر دهیم!

**ضریب هوشی!**

متوسط ضریب هوشی ایرانی‌ها ۸۵ بوده و در بین ۲۰۰ و اندی کشور، رتبه ۸۳ را داراست.

**فلسفه!**

فلسفه هر چیزی از خود آن چیز مهم‌تر است!

* فلسفه ازدواج (ازدواج دو ارمنی)
* فلسفه فرزندآوری (برای بهبود زندگی و اصلاح طرف مقابل!)
* فلسفه مذهب (اهمیت حق الناس! در همه ادیان)
* فلسفه مهاجرت (هجرت!)
* فلسفه انتخاب بانک اطلاعاتی! مثلاً اوراکل و SQL Server
* فلسفه انتخاب زبان برنامه‌نویسی! مثلاً #C و Python (من جزء خوارج نشده‌ام!)

**تعصب!**

تعصب در هر موضوعی مجاز نیست! و در جایگاهی هم که الزامی است! تا فلسفه آن چیز را نمی‌دانیم، نباید پرخاشگرانه و غیر منصفانه اظهارنظر کنیم!

* طرفداری از یک تیم فوتبال
* طرفداری از فلسطین و اسرائیل

**جوگیر شدن! و تغییر رویه!**

* دو تا جلسه هیات / دو ساعت مشاهده فیلم‌ها و سخنرانی‌های Instagram و ماهواره
* انتخابات!

**به نام خدا**

**جلسه دوم**

**یک سؤال عجیب و غیر علمی!؟**

**زبان سی‌شارپ بهتر است یا زبان پایتون و یا… ؟**

می‌خواهم از تهران به کرج بروم! از چه وسیله‌ای استفاده کنم؟

* بنز
* خاور

نمونه‌ای از یک پژوهشگاه در سال ۱۳۸۰ در انتخاب بانک اطلاعاتی

هنر یک معمار نرم‌افزار؟

مدیریت مثلث طلائی! **زمان هزینه کیفیت / Feature**

یک نمونه از یک مناقصه در اوایل ریاست جمهوری آقای احمدی نژاد

انتخاب زبان:

تقریباً با هر زبانی، می‌توان همه کار انجام داد!

خلاصه: زبان سی‌شارپ یک بستر قدرتمند برای ساخت یک مجمع یا حتی یک برج را به ما می‌دهد! ولی زبان پایتون، بستر قدرتمندی نداشته، شکننده است و می‌توان در حد ایجاد یک آلاچیق یا حداکثر یک ویلای یک طبقه از آن استفاده کرد!

سایت‌های معروف پایتون:

اولین سایت معروف: سایت DropBox

**زبان پایتون**

بسیار ساده!

برای هوش مصنوعی **فوق‌العاده** است! ولی چرا؟

برای Web Scraping مناسب است! ولی چرا؟

برای اسکریپت‌نویسی **فوق‌العاده** است!

به تازگی در زبان برنامه‌نویسی سی‌شارپ نیز امکان اسکریپت‌نویسی وجود ندارد!

برای کارهای محاسبات عددی و ریاضی **فوق‌العاده** است!

برای Ethical Hacking **فوق‌العاده** است!

برای IoT مناسب است

Raspberry Pi

زبان مایکروپایتون MicroPython

Raspberry Pi Pico – ESP

مایکروسافت به دلیل انتقادات زیاد، تا حدی ساده‌سازی کرده است!

Do not use top-level statements

ولی چیزی که هم پایتون را جذاب می‌کند و هم **فاجعه است** در چیست؟

تعریف متغیرها!؟

اگر هر متغیری را یک لیوان (ظرف) در نظر بگیریم! دنیای پایتون به دنیای واقعی ما نزدیک و شبیه می‌باشد! ولی در سی‌شارپ / جاوا / سی و غیره اینگونه نیست!

**In C#:**

int x;

x = 10;

x = “Ali”; // خطا

int x = 10;

x = “Ali”; // خطا

var x; // خطا

var x = 10;

x = “Ali”; // خطا

**Strongly Typed**

**In Python:**

x = 10

print(x)

x = “Ali”

print(x)

چه استراتژی برای زبان پایتون توصیه می‌کنم؟

اگر یک زبان برنامه‌نویسی قوی بلد هستید، اگر می‌خواهید موضوعاتی که به طور تخصصی، تخصص و قدرت پایتون می‌باشد را پیگیری کنید

اگر می‌خواهید در ابتدا یک زبان برنامه‌نویسی قوی را شروع کنید، و هیچ‌گونه آشنایی با برنامه‌نویسی ندارید، برای شروع، و حتی قبل از دوره C# Fundamental شاید بد نباشد، یک دوره مقدماتی پایتون، برای درک بهتر برنامه‌نویسی بگذرانید!

کتابخانه‌های سی‌شارپ

کتابخانه‌هایی برای پروژه‌های Desktop

Windows Forms Application

WPF

کتابخانه‌هایی برای پروژه‌های وب

ASP.NET Core

MVC (Full Stack)

Razor Pages (Full Stack)

Web API

کتابخانه‌های پایتون **(این همه تنوع هم خیلی خوب نیست!)**

کتابخانه‌هایی برای پروژه‌های Desktop

PyQt

**Tkinter**

wxPython

pySide

Kivy

Libavg

PySimpleGUI

PyForms

Wax

PyGUI

…

کتابخانه‌هایی برای پروژه‌های وب

Django (Full Stack)

**Flask**

FastAPI

…

**به نام خدا**

**جلسه سوم**

C# Road Map

مساله‌ای به نام کارآموزی!

اگر هیچ زبان برنامه‌نویسی / الگوریتم / فلوچارت نمی‌دانید

Python Fundamental

C# Fundamental (24)

C# (Console Application)

Clean Code

Windows Forms Application

Programming in C# (1) (40)

Clean Code

OOP → Object Oriented Programming (Console Application)

Windows Forms Application

WPF → Windows Presentation Foundation (Direct X)

**Database**

LINQ To Entity Framework **Core** Code First (40)

**More Courses after 3 or 5 years:**

Design Patterns → تفکر

Programming in C# (2) → تکنولوژی

DDD → Domain Driven Design → متدلوژی

Micro-Service → معماری

**Web**

Full Stack / Frontend : HTML (5)

Full Stack / Frontend : CSS (4)

Full Stack / Frontend : Bootstrap (5.3.x)

Full Stack / Frontend : Less

Full Stack / Frontend : Sass

Full Stack / Frontend : JavaScript (ES6)

Full Stack / Frontend : Jquery

**ASP.NET Core**

Full Stack : MVC Core (Tiny / Small / Medium) Scale Projects

Full Stack : Razor Pages (Tiny / Small / Medium) Scale Projects

Backend : Web API (Large / Extra Large) Scale Projects

Onion Architecture (Old) → Clean Architecture

Frontend :

React (Library) (Normal / **SPA** → Single Page Application → PWA)

**JavaScript**

TypeScript

Vue (Library) (**Normal** / SPA → Single Page Application → PWA)

**JavaScript**

TypeScript

Angular (Framework) (**SPA** → Single Page Application → PWA)

**TypeScript**

**Blazor Web Assembly (Framework) (Not Blazor Server!)**

C#

You can use JavaScript!

Faster

Strongly Typed

More Security

**Learn 100**

[[Client]] → Create File : libman.json

[wwwroot] → Update File : index.html

**Learn 200**

[Properties] → Update File : launchSettings.json

[Shared] → Delete File : SurveyPrompt.razor

[Pages] → Delete File : Counter.razor

[Pages] → Delete File : FetchData.razor

[wwwroot] → Delete Folder : sample-data

[Pages] → Update File : Index.razor

[Pages] → Create File : Page1.razor

[Pages] → Create File : Page2.razor

[Pages] → Create File : Page3.razor

[Pages] → Create File : Page4.razor

[Pages] → Create File : Page5.razor

[Shared] → Update File : NavMenu.razor

Link to Pages…

[Pages] → Create Folder : Authentication

[Authentication] → Create File : Login.razor

[Authentication] → Create File : Logout.razor

[Shared] → Update File : MainLayout.razor

Link to Login and Logout

**Learn 300**

[Properties] → Update File : launchSettings.json

**Resources:**

[ASP.NET Core Blazor authentication and authorization](https://learn.microsoft.com/en-us/aspnet/core/blazor/security/webassembly/?view=aspnetcore-7.0)

<https://learn.microsoft.com/en-us/aspnet/core/blazor/security>

Secure ASP.NET Core Blazor WebAssembly

https://learn.microsoft.com/en-us/aspnet/core/blazor/security/WebAssembly

ASP.NET Core Blazor state management

https://learn.microsoft.com/en-us/aspnet/core/blazor/state-management

ASP.NET Core Blazor Hybrid authentication and authorization

https://learn.microsoft.com/en-us/aspnet/core/blazor/hybrid/security

ASP.NET Core Blazor WebAssembly additional security scenarios

https://learn.microsoft.com/en-us/aspnet/core/blazor/security/webassembly/additional-scenarios

**NuGets:**

https://www.nuget.org/packages/Microsoft.AspNetCore.Components.Authorization

[[Client]] → Update File : Client.csproj

Add NuGets…

[[Client]] → Create Folder : Providers

[Infrastructure] → Create File : CustomAuthenticationStateProvider.cs

Anonymous

[[Client]] → Update File : Program.cs

builder.Services.AddOptions();

// If we do not have Policy!

//builder.Services.AddAuthorizationCore();

// If we have Policy!

builder.Services.AddAuthorizationCore(options =>

{

options.AddPolicy("CanBuy", policy =>

policy.RequireClaim("Over21"));

options.AddPolicy("CanDelete", policy =>

policy.RequireRole("Administrator"));

});

builder.Services.AddScoped

<AuthenticationStateProvider,

CustomAuthenticationStateProvider>();

[Shared] → Create File : RedirectToLogin.razor

[[Client]] → Update File : App.razor

[[Client]] → Update File : \_Imports.razor

@using Microsoft.AspNetCore.Authorization; → For [Authorize]

@using Microsoft.AspNetCore.Components.Authorization → For App.razor

Breakpoints:

[Infrastructure] → Create File: CustomAuthenticationStateProvider.cs

[Shared] → Create File: RedirectToLogin.razor

[Pages] → Update File : Page2.razor

@attribute [Authorize]

[Pages] → Update File : Page3.razor

@attribute [Authorize(Roles = "Administrator")]

[Pages] → Update File : Page4.razor

<AuthorizeView></AuthorizeView>

**Learn 400**

References:

Implementing Authentication

https://blazorschool.com/tutorial/blazor-wasm/dotnet6/authentication-and-authorization-724479

https://blazorschool.com/tutorial/blazor-wasm/dotnet6/implementing-authentication-829424

https://blazorschool.com/tutorial/blazor-wasm/dotnet6/implementing-authorization-980267

AuthenticationStateProvider in Blazor WebAssembly

https://code-maze.com/authenticationstateprovider-blazor-webassembly

Custom Authentication in Blazor WebAssembly – Step-By-Step Detailed Guide

https://codewithmukesh.com/blog/authentication-in-blazor-webassembly

[Properties] → Update File : launchSettings.json

[Infrastructure] → Create File : CustomAuthenticationStateProvider.cs

Not Anonymous!

[Authentication] → Update File : Logout.razor

**Learn 500**