predev. Academy

خريطة المسار

تطوير الـ Backend باستخدام .NET

تبني REST API جاهزة للإنتاج وتختبرها وتنشرها باستخدام C# 14 و ASP.NET Core و EF Core و PostgreSQL على سيرفر Linux خاص بك.

Web Backend

المدة	الجلسات	المواعيد	المستوى	لغة الشرح	طريقة الحضور
10 أسابيع	20 جلسة مباشرة، 60 ساعة	مرتين في الأسبوع، 3 ساعات لكل محاضرة، مع ساعة office hours أونلاين كل أسبوع	مبتدئ، بعد Tech Compass	عربي مصري مع المصطلحات التقنية بالإنجليزية	حضوري أو أونلاين، لايف

لمن هذا المسار

- طلاب علوم الحاسب ونظم المعلومات الذين يريدون وظيفة backend بـ .NET بعد التخرج
- حديثو التخرج الذين يعرفون أساسيات البرمجة لكنهم لم يسلموا API حقيقية من قبل
- من يغيرون مسارهم المهني وأنهوا Tech Compass واختاروا تطوير الـ backend
- مطورون مبتدئون في تقنيات أخرى يريدون الانتقال إلى C# و ASP.NET Core، وهي التقنية التي توظف عليها بنوك وشركات كبيرة كثيرة في مصر وعملاء الخليج

قبل اليوم الأول

- إنهاء Tech Compass، أو معرفة أساسيات البرمجة: المتغيرات والشروط والتكرار والدوال
- لابتوب بذاكرة 8 GB RAM على الأقل (يفضل 16 GB) وقرص SSD، بنظام Windows 11 أو macOS أو Linux
- تثبيت قبل اليوم الأول: .NET 10 SDK، و Visual Studio Code مع إضافة C# Dev Kit (أو JetBrains Rider)، و Git، و Docker Desktop، و Postman أو Bruno
- حسابات: GitHub (بصورة شخصية واسمك الحقيقي)، وحساب Claude أو Cursor للأسبوع 8
- القدرة على قراءة توثيق تقني بسيط باللغة الإنجليزية

في نهاية المسار تستطيع

- تشرح ما يحدث في HTTP request وتختار الـ method و status code وشكل JSON المناسب لكل endpoint
- تكتب C# 14 نظيفة باستخدام classes و interfaces و generics و LINQ و records و async/await و nullable reference types
- تصمم schema علائقية وتكتب استعلامات SQL فيها joins و grouping و indexes على PostgreSQL و SQL Server
- تبني REST API باستخدام ASP.NET Core بالـ controllers أو Minimal APIs مع validation ومعالجة الأخطاء وتوثيق OpenAPI
- تصمم البيانات باستخدام EF Core 10 وتشغل migrations وتكتب استعلامات فعالة وتستخدم transactions
- تؤمن الـ API باستخدام ASP.NET Core Identity و JWT access و refresh tokens والصلاحيات بالأدوار والـ policies
- تضيف background jobs و Redis caching و structured logging باستخدام Serilog
- تكتب unit tests باستخدام xUnit و integration tests باستخدام WebApplicationFactory
- تحول الـ API إلى container باستخدام Docker وتنشرها على Linux VPS عن طريق Coolify مع domain و HTTPS

- تستخدم أدوات الذكاء الاصطناعي لتسريع العمل مع مراجعة كل تعديل والحفاظ على نجاح الاختبارات

التقنيات المستخدمة

اللغة وبيئة التشغيل

.NET 10 LTS / C# 14	إصدار بدعم طويل المدى يستخدم في مشاريع الشركات الجديدة، مع مزايا C# الحديثة مثل records و nullable reference types
.NET CLI	إنشاء المشاريع وبناؤها وتشغيلها واختبارها من الـ terminal بنفس الطريقة التي يعمل بها الـ CI والسيرفرات

البيانات

Entity Framework Core 10	الـ ORM الأساسي في .NET: الـ models و migrations والعلاقات واستعلامات LINQ
PostgreSQL 17	مجانية وسريعة، وهي الاختيار الافتراضي لمعظم الـ APIs الجديدة والنشر على VPS
SQL Server 2022	ما زالت قاعدة البيانات الأكثر طلبا في إعلانات وظائف الشركات الكبيرة والبنوك في مصر
Redis 8	تخزين القراءات المتكررة في cache ومشاركة الحالة بين أكثر من instance

التشغيل

Serilog	structured logs تستطيع البحث فيها عندما يحدث خطأ في الإنتاج
Hangfire و Hosted services	تشغيل الإيميلات والتذكيرات وأعمال التنظيف خارج الـ request
Docker	تغليف الـ API والأدوات التي تعتمد عليها بنفس الشكل على أي جهاز
Coolify على Linux VPS	نشر الـ containers مع HTTPS و environment variables وقواعد البيانات على سيرفر تتحكم فيه

إطار عمل الويب

ASP.NET Core 10 Web API	Controllers و Minimal APIs و middleware و dependency injection و configuration جاهزة داخل الإطار
FluentValidation	قواعد validation واضحة ومفصولة عن الـ controllers
Scalar مع OpenAPI	توثيق API مدمج يستطيع فريق الـ frontend والموبايل الاختبار عليه

الأمان

ASP.NET Core Identity	المستخدمون و password hashing والأدوار و lockout دون كتابتها من الصفر
JWT bearer tokens	مصادقة stateless لتطبيقات الموبايل والـ single page frontends

الجودة والأدوات

xUnit و WebApplicationFactory	unit tests و integration tests تستدعي الـ API الحقيقية في الذاكرة
Git و GitHub	إدارة الإصدارات و pull requests والـ portfolio الذي ينظر إليه أصحاب العمل
Cursor أو Claude Code	أدوات برمجة بالذكاء الاصطناعي نستخدمها مع المراجعة والاختبارات كما تستخدمها الفرق اليوم

مشروع المسار: Mawid: API لحجز المواعيد

Mawid هو الـ backend لتطبيق حجز تنشر فيه العيادات والصالونات ومراكز التدريب خدماتها ومواعيد العاملين، ويحجز العملاء المواعيد ويعدلونها ويلغونها. المشروع يشبه أنظمة الحجز والتشغيل التي تبنيها predev. Solutions لعملائها، ويغطي المشاكل التي تواجه أي backend حقيقي: المستخدمين والأدوار، والبيانات المرتبطة، والحجز المزدوج، والتذكيرات، والـ caching، والنشر.

- التسجيل وتسجيل الدخول و JWT access tokens و refresh tokens باستخدام ASP.NET Core Identity

- الأدوار: عميل وصاحب نشاط وموظف ومدير نظام، مع policies مثل أن صاحب النشاط يعدل نشاطه فقط
- الأنشطة والفروع والخدمات والموظفون مع ساعات العمل
- endpoint للمواعيد المتاحة يحسب الأوقات الفارغة لكل موظف وخدمة
- حجوزات مع الإنشاء والتعديل والإلغاء وتغيير الحالة، ومحمية من الحجز المزدوج باستخدام transaction و unique constraint
- بحث وفلترة وترتيب و pagination على الأنشطة والحجوزات
- validation باستخدام FluentValidation وردود أخطاء موحدة بصيغة ProblemDetails
- إيميلات تذكير وتنظيف يومي كـ background jobs
- Redis caching لقوائم الأنشطة والمواعيد المتاحة، مع إلغاء الـ cache عند التغيير
- تسجيل الـ requests والأخطاء بشكل structured باستخدام Serilog
- مستند OpenAPI مع واجهة Scalar
- مجموعة unit و integration tests تعمل بأمر dotnet test
- Docker image وإعداد docker compose مع PostgreSQL و Redis
- نشر حي على Linux VPS عن طريق Coolify مع domain و HTTPS

مراحل المسار

الأسابيع 1-3
الأساسيات
نفهم كيف يعمل الويب ونكتب C# 14 و SQL بشكل سليم قبل استخدام أي framework.

الأسابيع 4-7
الـ Framework الأساسي
نبني Mawid booking API خطوة بخطوة باستخدام ASP.NET Core و EF Core و Identity والـ caching والـ jobs والـ logging والاختبارات و Docker.

الأسابيع 8
البناء بالذكاء الاصطناعي
نستخدم أدوات البرمجة بالذكاء الاصطناعي لتسريع العمل الحقيقي على Mawid مع الحفاظ على التحكم في الجودة والأمان.

الأسابيع 9
مشروع التخرج
ننشر Mawid على سيرفر حقيقي، ثم يبني كل طالب وينشر API خاصة به من brief.

الأسابيع 10
أسبوع المسار المهني
تعرض شغلك وتستعد لمقابلات .NET. وتبدأ التقديم على الوظائف والعمل الحر.

الخطة أسبوعًا بأسبوع

الأساسيات · الأسابيع 1-3

الأسبوع 1 كيف يعمل الويب وأول خطوات في C#

المحاضرة 1 HTTP و JSON ونموذج client server

- الـ clients والسيرفرات و DNS وعناوين IP والـ ports
- HTTP requests و responses: الـ methods والـ URLs والـ headers والـ body
- الـ status codes حسب العائلة: 2xx و 3xx و 4xx و 5xx ومتى نستخدم كل منها
- صيغة JSON وكيف تستخدمها الـ APIs
- أساسيات REST: الـ resources، والأسماء في الـ URLs، و statelessness
- اختبار APIs عامة حقيقية باستخدام Postman و curl

تطبيق عملي في المحاضرة: نستدعي ثلاث APIs عامة باستخدام Postman و curl ونقرأ الـ headers والـ status codes، ثم نكتب API contract من صفحة واحدة لقائمة مهام بسيطة: الـ endpoints والـ methods والـ request bodies والـ status codes المتوقعة.

المحاضرة 2 إعداد .NET 10. والـ CLI وأساسيات C#

- ما هي .NET.: الـ SDK والـ runtime و NuGet وإصدارات LTS
- أوامر dotnet new و build و run وهيكل مشروع console
- المتغيرات والأنواع الأساسية و string interpolation و var
- الشروط والتكرار و switch expressions
- الـ methods والـ parameters والقيم المرجعة
- أساسيات Git: init و add و commit و push إلى GitHub

تطبيق عملي في المحاضرة: نبني تطبيق console لتتبع المصروفات يقرأ المبالغ والتصنيفات من المستخدم ويطبع الإجمالي لكل تصنيف، ثم نرفعه إلى repository جديد على GitHub مع commits واضحة.

الواجب: حل 6 مسائل للمبتدئين على التكرار والشروط بلغة C# (من قائمة مرفقة)، وإضافة تنبيه ميزانية شهرية لتطبيق المصروفات، ورفع كل شيء على GitHub مع 5 commits واضحة على الأقل.

الأسبوع 2 البرمجة الكائنية في C# والـ collections و LINQ

المحاضرة 1 الـ classes والـ interfaces والـ generics

- الـ classes والـ objects والـ constructors والـ properties
- الـ encapsulation والـ access modifiers
- الوراثة ومتى يكون الـ composition اختيارا أفضل
- الـ interfaces والبرمجة على abstractions
- الـ generics: generic classes و methods والـ constraints
- reference types و value types والـ structs

تطبيق عملي في المحاضرة: نصمم نظام مكتبة صغير بالـ classes التالية: Book و Member و Loan، مع interface اسمها ILoanPolicy لها تطبيقان، ثم نكتب generic Repository<T> تخزن العناصر في الذاكرة.

المحاضرة 2 الـ collections و LINQ و Git branching

- List و Dictionary و HashSet ومتى نستخدم كل منها
- LINQ: Where و Select و OrderBy و GroupBy و Any و First و Sum
- الـ deferred execution ولماذا يهمنا لاحقا مع EF Core
- طريقة حل المسائل: افهم، مثال، خطة، كود، اختبار
- الـ branches في Git والدمج و pull requests على GitHub
- كتابة README مفيد

تطبيق عملي في المحاضرة: نحمل ملف JSON فيه 500 طلب ونجيب عن 8 أسئلة تخص البيزنس باستخدام LINQ (أكثر العملاء شراء، الإيراد لكل شهر، المنتج الأكثر مبيعا). نعمل على feature branch ونفتح pull request للمراجعة.

الواجب: إنهاء أسئلة LINQ وإضافة 4 أسئلة من عندك، وحل 5 مسائل على الـ arrays والـ strings على LeetCode أو HackerRank. مراجعة pull request لزميل وكتابة 3 تعليقات على الأقل.

الأسبوع 3 C# الحديثة وقواعد البيانات العلائقية

المحاضرة 1 async/await والـ records والـ nullability والـ exceptions

- الـ Tasks و async/await ولماذا يحتاج السيرفر async I/O
- أساسيات HttpClient واستدعاء API من C#
- الـ records و init properties والـ immutability
- nullable reference types وقراءة تحذيرات الـ compiler
- الـ exceptions: try/catch/finally و custom exceptions ومتى لا نرمي exception
- الـ pattern matching على الأنواع والـ properties

تطبيق عملي في المحاضرة: نكتب تطبيق console غير متزامن يجلب بيانات من API عامة باستخدام HttpClient ويحولها إلى records ويتعامل مع أخطاء الشبكة والحقول الفارغة ويطبع تقريرا ملخصا.

المحاضرة 2 SQL وتصميم قواعد البيانات العلائقية

- الجداول و primary keys و foreign keys والـ constraints
- علاقات one to many و many to many وأساسيات الـ normalization
- SELECT و WHERE و JOIN و GROUP BY و HAVING و ORDER BY
- INSERT و UPDATE و DELETE والـ transactions
- الـ indexes وقراءة query plan بسيط
- PostgreSQL و SQL Server جنبا إلى جنب داخل Docker

تطبيق عملي في المحاضرة: نصمم أول database schema لمشروع Mawid (المستخدمون والأنشطة والخدمات والموظفون والحجوزات) على الورق، ثم ننشئها على PostgreSQL داخل Docker ونضيف بيانات تجريبية ونكتب 10 استعلامات مثل عدد الحجوزات لكل نشاط في كل يوم.

الواجب: تشغيل نفس الـ schema على SQL Server وتسجيل الفروق، وكتابة 10 استعلامات SQL إضافية من ورقة التمارين، وإضافة صورة ERD مختصرة لـ schema مشروع Mawid في الـ README.

الأسبوع 4 أساسيات ASP.NET Core Web API

المحاضرة 1 أول API: الـ routing والـ controllers و Minimal APIs

- إنشاء مشروع Web API وقراءة Program.cs
- الـ Controllers و Minimal APIs: نفس الوظيفة بأسلوبين
- الـ routing والـ route parameters والـ query strings والـ model binding
- إرجاع النتائج: Ok و Created و NotFound و NoContent
- الـ middleware pipeline وترتيب تنفيذه
- مستند OpenAPI المدمج وواجهة Scalar

تطبيق عملي في المحاضرة: ننشئ solution مشروع Mawid ونبني CRUD endpoints للأنشطة والخدمات مع تخزين في الذاكرة، مرة بالـ controllers ومرة بـ Minimal APIs، ثم نختبرها من Scalar و Postman.

المحاضرة 2 الـ Dependency injection والـ configuration والـ validation

- الـ dependency injection: الـ services والـ lifetimes (singleton و scoped و transient)
- appsettings.json والـ environments و user secrets و environment variables
- الـ options pattern مع إعدادات strongly typed
- الـ DTOs وإبعاد الـ entities عن الـ responses
- الـ validation باستخدام FluentValidation
- معالجة الأخطاء بشكل عام وردود ProblemDetails

تطبيق عملي في المحاضرة: ننقل المنطق إلى services مسجلة في الـ DI، ونضيف request DTOs بقواعد FluentValidation، ونضيف global exception handler يرجع ProblemDetails، ونحمل إعدادات النشاط باستخدام الـ options pattern.

الواجب: إضافة endpoints للموظفين وساعات العمل مع validation كامل و status codes صحيحة. كتابة Postman collection تغطي حالات النجاح والفشل لكل endpoint.

مرحلة في المشروع: هيكل Mawid API: endpoints للأنشطة والخدمات والموظفين مع validation ومعالجة الأخطاء وتوثيق OpenAPI.

الأسبوع 5 Entity Framework Core 10

المحاضرة 1 الـ DbContext والـ entities والعلاقات والـ migrations

- DbContext و DbSet والاتصال بـ PostgreSQL
- إعداد الـ entities بالـ conventions و Fluent API
- علاقات one to many و many to many و owned types
- الـ migrations: الإضافة والتحديث والتراجع ومراجعة الـ SQL الناتج
- إضافة بيانات تجريبية لبيئة التطوير
- تغيير الـ provider إلى SQL Server

تطبيق عملي في المحاضرة: نستبدل التخزين في الذاكرة بـ EF Core و PostgreSQL. نعد كل entities مشروع Mawid وعلاقاتها، وننشئ أول migration ونراجع الـ SQL الخاص بها، ونضيف أنشطة وموظفين تجريبيين.

المحاضرة 2 الاستعلامات والـ tracking والـ transactions

- LINQ to SQL: Where و Include والـ projection باستخدام Select
- الـ change tracking و AsNoTracking لاستعلامات القراءة فقط
- مشكلة N+1 وكيف نكتشفها من الـ logs
- الـ pagination والفلترة والترتيب
- الـ transactions والـ unique constraints لمنع الحجز المزدوج
- الـ concurrency tokens والتعامل مع التعارضات

تطبيق عملي في المحاضرة: نبني endpoints الحجز و endpoint المواعيد المتاحة. نمنع الحجز المزدوج باستخدام transaction مع unique index، ثم نثبت ذلك بإرسال requests متوازية من script صغير.

الواجب: إضافة البحث والفلترة والـ pagination لقائمة الأنشطة، وإضافة endpoint لتعديل موعد الحجز، وإصلاح أي استعلامات N+1 تجدها باستخدام logs الخاصة بـ EF Core.

مرحلة في المشروع: Mawid يحفظ البيانات في PostgreSQL: migrations وعلاقات وحجوزات ومواعيد متاحة وحماية من الحجز المزدوج.

الأسبوع 6 المصادقة والصلاحيات وتنظيم المشروع

المحاضرة 1 ASP.NET Core Identity و JWT

- الـ authentication والـ authorization: السؤال الذي يجيب عنه كل منهما
- ASP.NET Core Identity مع EF Core: المستخدمون و password hashing و lockout
- إصدار JWT access tokens والتحقق منها
- الـ refresh tokens: التخزين والتدوير والإلغاء
- حماية الـ endpoints وقراءة المستخدم الحالي
- أخطاء شائعة: أسرار داخل الكود، tokens صالحة لفترة طويلة، غياب HTTPS

تطبيق عملي في المحاضرة: نضيف endpoints للتسجيل وتسجيل الدخول والـ refresh وتسجيل الخروج في Mawid باستخدام Identity و JWT، ثم نجعل endpoints الحجز تستخدم المستخدم المسجل دخوله بدلا من user id يأتي في الـ request.

المحاضرة 2 الأدوار والـ policies وهيكل طبقات عملي

- الصلاحيات بالأدوار: عميل وصاحب نشاط وموظف ومدير نظام
- الصلاحيات بالـ policies والتحقق من ملكية الـ resource (صاحب النشاط يعدل نشاطه فقط)
- تقسيم الـ solution إلى مشاريع Api و Application و Domain و Infrastructure
- أين توضع قواعد البيزنس وإبقاء الـ controllers خفيفة
- متى تضيف الطبقات والـ patterns الإضافية تكلفة دون فائدة
- قائمة مراجعة الكود لـ pull requests الخاصة بالـ API

تطبيق عملي في المحاضرة: نعيد تنظيم Mawid إلى solution مقسمة لطبقات، وننقل قواعد الحجز إلى الـ application layer، ونضيف قواعد الأدوار والـ policies بحيث لا يستطيع صاحب نشاط تعديل بيانات نشاط آخر. نتحقق من ذلك في Postman بأربعة مستخدمين مختلفين.

الواجب: إضافة endpoint للمدير لإيقاف نشاط، و endpoint للموظف لتعليم الحجز كمكتمل أو لم يحضر، كل منهما بالـ policy المناسبة. كتابة ملاحظة أمان قصيرة في الـ README توضح ما هو محمي وكيف.

مرحلة في المشروع: Mawid مؤمن: Identity و JWT مع refresh tokens وأدوار و policies و solution منظمة في طبقات واضحة.

الأسبوع 7 الـ Background jobs والـ caching والـ logging والاختبارات و Docker

المحاضرة 1 الـ Background jobs و Redis caching و Serilog

- الـ hosted services و BackgroundService للأعمال المتكررة البسيطة
- Hangfire لـ jobs المجدولة والقابلة لإعادة المحاولة مع dashboard
- الـ caching باستخدام IDistributedCache و Redis والـ expiry والـ invalidation
- الـ output caching لـ endpoints العامة
- الـ structured logging باستخدام Serilog: الـ sinks والـ enrichers و request logging
- الـ correlation ids لتتبع request واحد في الـ logs

تطبيق عملي في المحاضرة: نجدول إيميلات تذكير بالحجز باستخدام Hangfire، ونضيف hosted service للتنظيف الليلي، ونخزن قوائم الأنشطة والمواعيد المتاحة في Redis مع إلغاء الـ cache عند التغيير، ونضيف Serilog request logging.

المحاضرة 2 الاختبار باستخدام xUnit والتسليم باستخدام Docker

- unit tests باستخدام xUnit: arrange و act و assert والـ theories
- integration tests باستخدام WebApplicationFactory
- قواعد بيانات حقيقية في الاختبارات باستخدام Testcontainers لـ PostgreSQL
- اختبار الـ endpoints التي تحتاج تسجيل دخول
- كتابة Dockerfile لـ ASP.NET Core API multi stage و builds
- docker compose مع الـ API و PostgreSQL و Redis

تطبيق عملي في المحاضرة: نكتب unit tests لحساب المواعيد المتاحة و integration tests للتسجيل وتسجيل الدخول وإنشاء حجز والحجز المزدوج. ثم نحول Mawid إلى container ونشغل كل المكونات باستخدام docker compose.

الواجب: الوصول إلى 15 اختبارا ناجحا على الأقل تغطي قواعد الحجز الأساسية، وإضافة health check endpoint، وتوثيق طريقة تشغيل المشروع محليا باستخدام docker compose في الـ README.

مرحلة في المشروع: Mawid جاهز للإنتاج: background jobs و Redis caching و structured logs ومجموعة اختبارات وإعداد docker compose يعمل.

الأسبوع 8 البناء بالذكاء الاصطناعي بالطريقة الصحيحة

المحاضرة 1 أدوات البرمجة بالذكاء الاصطناعي في مشروع حقيقي

- كيف تقرأ أدوات مثل Claude Code و Cursor الكود وتعدل فيه
- إعطاء السياق: ملف تعليمات المشروع والـ conventions ومهام صغيرة محددة
- التخطيط لـ feature مع الذكاء الاصطناعي قبل كتابة الكود
- قراءة الـ diffs التي يولدها الذكاء الاصطناعي سطرا بسطر قبل قبولها
- أخطاء شائعة للذكاء الاصطناعي في .NET: إصدارات packages خاطئة، APIs قديمة، methods غير موجودة
- إبعاد الأسرار وبيانات العملاء عن الـ prompts

تطبيق عملي في المحاضرة: نكتب ملف تعليمات مشروع لـ Mawid، ثم نستخدم Claude Code أو Cursor لإضافة feature التقييمات (العملاء يقيمون الحجوزات المكتملة). نراجع كل diff ونرفض الخطأ ونسجل ما عدلناه بأيدينا.

المحاضرة 2 الاختبارات كشبكة أمان ومتى لا نثق في الذكاء الاصطناعي

- كتابة الاختبارات أولا وطلب جعلها تنجح من الذكاء الاصطناعي
- استخدام الذكاء الاصطناعي لكتابة الاختبارات ثم التأكد أنها يمكن أن تفشل
- مراجعة أمان كود الذكاء الاصطناعي: التحقق من الصلاحيات و SQL injection و mass assignment
- الـ debugging مع الذكاء الاصطناعي: إعطاء الـ logs والـ stack traces والتحقق من الإصلاح
- مهام تبقى في يدك: المصادقة، المال، data migrations، قرارات الـ architecture
- الصراحة في المقابلات حول طريقة استخدامك للذكاء الاصطناعي

تطبيق عملي في المحاضرة: نأخذ ثلاثة pull requests ولدها الذكاء الاصطناعي وجهزها المدرب، في كل منها bug أو ثغرة أمان مخفية. نجد المشكلة ونكتب اختبارا فاشلا يثبتها ثم نصلحها بالذكاء الاصطناعي أو بدونه.

الواجب: استخدام الذكاء الاصطناعي لإضافة feature أخرى لمشروع Mawid من اختيارك (مثل استثناءات مواعيد عمل النشاط أو تصدير الحجوزات إلى CSV) مع اختبارات. تسليم سجل مراجعة قصير: ما أحسن فيه الذكاء الاصطناعي، وما رفضته، ولماذا.

مرحلة في المشروع: Mawid يحصل على feature مبنية بمساعدة الذكاء الاصطناعي مع اختبارات وسجل مراجعة مكتوب.

مشروع التخرج · الأسابيع 9

الأسبوع 9 نشر حقيقي وبناء مشروع التخرج

المحاضرة 1 النشر على Linux VPS باستخدام Coolify

- أساسيات Linux VPS: مفاتيح SSH والمستخدمون والـ firewall والتحديثات
- Coolify: الـ projects والتطبيقات من GitHub repository والبناء من Dockerfile
- PostgreSQL و Redis كخدمات مدارة داخل Coolify
- الـ environment variables والأسرار وتشغيل الـ migrations عند النشر
- الـ domains وسجلات DNS و HTTPS تلقائي
- الـ health checks والـ logs والرجوع عن نشر فيه مشكلة

تطبيق عملي في المحاضرة: ننشر Mawid على VPS الخاص بالدفعة عن طريق Coolify مع PostgreSQL و Redis و subdomain و HTTPS. نشغل الـ migrations ونختبر الـ API الحية من Postman، ثم يستلم كل طالب الـ brief الخاص بمشروع تخرجه ويقسمه إلى مهام.

المحاضرة 2 استوديو مشروع التخرج: بناء ومراجعة ونشر

- تحويل الـ brief إلى endpoints و schema وقائمة مهام
- تحديد النطاق: ما الذي نحذفه ليتم التسليم يوم الجمعة
- جولات مراجعة كود قصيرة مع المدرب والزملاء
- كتابة README فيه خطوات التشغيل وملخص الـ API والرابط الحي
- التحضير لعرض مدته 5 دقائق

تطبيق عملي في المحاضرة: جلسة استوديو: يبني كل طالب API مشروع تخرجه مع متابعة كل ساعة، ويحصل على مراجعة كود واحدة من المدرب، وينتهي بأول نشر للمشروع على Coolify.

الواجب: إنهاء مشروع التخرج: كل المتطلبات الأساسية، والاختبارات ناجحة، ومنشور مع HTTPS، والـ README مكتمل. تسليم رابط GitHub والرابط الحي قبل يوم الجمعة من الأسبوع 10.

مرحلة في المشروع: Mawid يعمل على VPS، و API مشروع التخرج منشورة مع HTTPS.

أسبوع المسار المهني · الأسابيع 10

الأسبوع 10 أسبوع المسار المهني ويوم العرض

المحاضرة 1 الـ CV و LinkedIn و GitHub والعمل الحر

- CV لمطور backend من صفحة واحدة يمر من فلاتر ATS
- بروفايل LinkedIn: الـ headline وقسم about والمشاريع المميزة
- GitHub portfolio: repositories مثبتة و READMEs وروابط حية وتاريخ commits نظيف
- أين توجد وظائف .NET في مصر وعن بعد: LinkedIn و Wuzzuf وصفحات التوظيف في الشركات
- أول خطوات في العمل الحر: مستقل و Upwork، وتحديد نطاق وتسعير مشروع API صغير
- كتابة رسالة تقديم قصيرة ومحددة

تطبيق عملي في المحاضرة: نعيد كتابة الـ CV وبروفايل LinkedIn مع ملاحظات مباشرة، ونحسن بروفايل GitHub مع تثبيت Mawid ومشروع التخرج، ونرسل 3 طلبات تقديم حقيقية داخل الجلسة.

المحاضرة 2 مقابلات تجريبية ويوم العرض

- أسئلة مقابلات .NET الشائعة: الـ DI lifetimes و async و EF Core tracking والـ middleware
- live coding: مسألة LINQ صغيرة أو مسألة حل مشكلات
- أساسيات system design للمبتدئين: شرح الـ API الخاصة بك من البداية للنهاية
- الأسئلة السلوكية والحديث عن مشاريعك
- يوم العرض: تعرض مشروع تخرجك أمام ضيوف من predev. Solutions وشركات شريكة

تطبيق عملي في المحاضرة: مقابلة تقنية تجريبية مدتها 30 دقيقة في ثنائيات مع ملاحظات المدرب، ثم عرض مشروع التخرج مباشرة في يوم العرض: المشكلة، العرض الحي، الـ architecture، وشيء واحد كنت ستغيره.

الواجب: التقديم على 10 وظائف أو مشاريع عمل حر مناسبة على الأقل خلال الأسبوعين التاليين، مع جدول متابعة بسيط، ومشاركة التقدم في الـ office hours.

مشروع التخرج

يستلم كل طالب brief لمنتج صغير حقيقي ويبني الـ REST API الخاصة به بمفرده في الأسبوع 9: نموذج البيانات والـ endpoints والمصادقة والـ validation والاختبارات ونشر حي على Linux VPS عن طريق Coolify. الـ brief يصف المستخدمين واحتياجاتهم، والطالب يحدد الـ endpoints والـ schema.

المتطلبات

- ASP.NET Core 10 Web API على .NET 10 مع EF Core 10 و PostgreSQL
- 5 entities مرتبطة على الأقل مع migrations
- التسجيل وتسجيل الدخول باستخدام ASP.NET Core Identity و JWT، مع دورين على الأقل وتحقق بالـ policies
- validation باستخدام FluentValidation وردود أخطاء ProblemDetails
- pagination وفلترة على list endpoint واحدة على الأقل
- background job واحدة على الأقل أو Redis cache
- structured logging باستخدام Serilog
- 10 اختبارات على الأقل منها integration tests باستخدام WebApplicationFactory
- مستند OpenAPI مع واجهة عرض
- Dockerfile ونشر حي على VPS عن طريق Coolify مع HTTPS
- README فيه خطوات التشغيل وملخص الـ API والرابط الحي

أفكار مقترحة

- API لمخزون وطلبات صيدلية مع تنبيهات نقص المخزون لسلسلة صيدليات صغيرة
- API لاشتراكات جيم مع الاشتراكات وحجز الكلاسات وتسجيل الحضور
- API لسوق خدمات منزلية (سباكين وكهربائية) مع الطلبات وعروض الأسعار والتقييمات
- API لمركز تدريب للكورسات والمجموعات والحضور ودفع الأقساط
- API لطلبات مطعم مع المنيو والفروع والطلبات وحالة التوصيل
- API لإيجار العقارات للإعلانات وطلبات المعاينة ولوحة تحكم المالك

معايير تقييم مشروع التخرج

يعمل من البداية للنهاية ومنشور مع HTTPS	25%
تصميم الـ API: الـ endpoints والـ status codes والـ validation والأخطاء	15%
نموذج البيانات واستخدام EF Core	15%
المصادقة والصلاحيات وأساسيات الأمان	15%
الاختبارات	15%
جودة الكود والتنظيم والـ README	10%
العرض والشرح	5%

طريقة التقييم

الواجبات الأسبوعية تسلم على GitHub قبل الجمعة التالية، وتقيم على الصحة والـ commits ووضوح الكود	20%
مراحل مشروع Mawid تراجع في الأسابيع 4-8 حسب قائمة المراحل، مع مراجعة كود قصيرة كل أسبوع	30%
مشروع التخرج يقيم حسب معايير مشروع التخرج	35%
المقابلة التجريبية ويوم العرض الإجابات التقنية والـ live coding ووضوح شرحك لعملك	10%
الحضور والمشاركة حضور 18 جلسة من 20 على الأقل، ومراجعة pull requests الزملاء، والنشاط في الـ office hours	5%

قائمة المهارات

الويب و HTTP

- [ ] قراءة وكتابة HTTP requests و responses
- [ ] اختيار الـ methods والـ status codes الصحيحة
- [ ] تصميم REST endpoints و JSON payloads

#C

- [ ] الـ classes والـ interfaces والـ generics والـ records
- [ ] الـ collections و LINQ
- [ ] async/await و HttpClient
- [ ] nullable reference types ومعالجة الـ exceptions

قواعد البيانات

- [ ] تصميم schema علائقية بعد الـ normalization
- [ ] كتابة SQL فيها joins و grouping و indexes
- [ ] العمل على PostgreSQL و SQL Server

ASP.NET Core

- [ ] الـ Controllers و Minimal APIs والـ routing والـ model binding
- [ ] الـ middleware pipeline ومعالجة الأخطاء بشكل عام
- [ ] الـ dependency injection والـ options pattern
- [ ] الـ validation باستخدام FluentValidation
- [ ] توثيق OpenAPI

EF Core

- [ ] الـ entities والعلاقات والـ migrations
- [ ] استعلامات فعالة و projection وقراءات no tracking
- [ ] الـ transactions والتعامل مع الـ concurrency

الأمان

- [ ] ASP.NET Core Identity
- [ ] JWT access و refresh tokens
- [ ] الصلاحيات بالأدوار والـ policies
- [ ] إدارة الأسرار باستخدام environment variables

الإنتاج

- [ ] background jobs باستخدام hosted services و Hangfire
- [ ] Redis caching والـ invalidation
- [ ] structured logging باستخدام Serilog
- [ ] Docker و docker compose
- [ ] النشر على Linux VPS باستخدام Coolify

الجودة وطريقة العمل

- [ ] unit tests باستخدام xUnit و integration tests باستخدام WebApplicationFactory
- [ ] الـ branches في Git و pull requests ومراجعة الكود
- [ ] مراجعة واختبار الكود الذي يولده الذكاء الاصطناعي

بعد التخرج

الشهور 1-3 بعد التخرج

- التقديم على وظائف junior .NET backend والتدريب كل أسبوع
- إضافة API أخرى منشورة إلى الـ portfolio
- التدرب على حل المسائل 3 مرات أسبوعيا على LeetCode
- قراءة أقسام توثيق ASP.NET Core و EF Core التي استخدمناها في الكورس كاملة

الشهور 4-6

- تعلم CI/CD باستخدام GitHub Actions: build و test و deploy عند كل push
- تعلم الـ messaging باستخدام RabbitMQ أو Azure Service Bus
- تعلم المزايا اللحظية باستخدام SignalR
- المساهمة بإصلاحات صغيرة في مشروع .NET. مفتوح المصدر

الشهور 7-12: نحو مستوى mid-level

- دراسة الأداء: الـ profiling وتحسين الاستعلامات والـ load testing
- تعلم منصة cloud (Azure أو AWS) والاستعداد لشهادة fundamentals
- تعلم الـ observability باستخدام OpenTelemetry
- القراءة عن domain driven design و modular monoliths وتطبيق ما يحتاجه مشروعك فقط
- توجيه طالب أحدث منك وقيادة مراجعات الكود

الوظائف التي يقود إليها المسار

Junior .NET Backend Developer | ASP.NET Core Developer | C# Software Engineer | API Developer

Junior Full Stack Developer (.NET مع فريق frontend) | مطور API حر على مستقل و Upwork

مطور .NET. عن بعد لشركات في الخليج وأوروبا

مصادر

- توثيق C#: https://learn.microsoft.com/en-us/dotnet/csharp/
- توثيق .NET: https://learn.microsoft.com/en-us/dotnet/
- توثيق ASP.NET Core: https://learn.microsoft.com/en-us/aspnet/core/
- توثيق Entity Framework Core: https://learn.microsoft.com/en-us/ef/core/
- ASP.NET Core Identity: https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity
- دليل HTTP من MDN: https://developer.mozilla.org/en-US/docs/Web/HTTP
- توثيق PostgreSQL: https://www.postgresql.org/docs/
- توثيق SQL Server: https://learn.microsoft.com/en-us/sql/sql-server/
- توثيق FluentValidation: https://docs.fluentvalidation.net/
- توثيق Redis: https://redis.io/docs/
- Serilog: https://serilog.net/
- توثيق Hangfire: https://docs.hangfire.io/
- xUnit.net: https://xunit.net/
- Testcontainers لـ .NET: https://dotnet.testcontainers.org/
- توثيق Docker: https://docs.docker.com/

- توثيق Coolify: https://coolify.io/docs/
- توثيق Git: https://git-scm.com/doc
- GitHub Docs: https://docs.github.com/
- توثيق Claude Code: https://docs.claude.com/en/docs/claude-code/overview
- توثيق Cursor: https://docs.cursor.com/