.NET, desarrollo con patrones y marcos de diseño

Curso práctico de 5 días - 35h
Ref.: TAA - Precio 2025: 2 120€ sin IVA

Crea aplicaciones empresariales sólidas y fáciles de mantener utilizando los patrones de diseño más homologados en ingeniería de software. Conoce los diferentes frameworks del ecosistema .NET y utiliza sus recursos/clases para acelerar su desarrollo, hacerlo más fiable y estandarizarlo.

OBJETIVOS PEDAGÓGICOS

Al término de la formación, el alumno podrá:

Dominar la inyección de dependencias y el ciclo de vida de los objetos

Instalar el patrón de comandos en una arquitectura CQRS

Dominar el acceso a datos y la aplicación de transacciones

Construir una API REST y una interfaz gráfica de usuario con AspNet Core

Se realizará un ejercicio como «hilo conductor» y se validará cada etapa mediante pruebas unitarias.

PROGRAMA

última actualización: 02/2024

1) Cuestiones de desarrollo de aplicaciones empresariales

- Objetivos: productividad, escalabilidad, adaptabilidad, comprobabilidad.
- Separación de responsabilidades.
- La aplicación monolítica.
- Arquitectura de microservicios.

Trabajo práctico : Familiarización con el entorno de desarrollo (Visual Studio).

2) Introducción al ecosistema .NET

- Varios lenguajes: C#, VB.NET, F#.
- La aparición de .NET Core, la unificación de .NET y de .NET Framework desde .NET 5.
- Frameworks: DependencyInjection, EntityFramework, ASP.NET, etc.

Trabajo práctico : Familiarización con el entorno de desarrollo.

3) Buenas prácticas de diseño y patrones de diseño

- Separación de responsabilidades con fachada (facade).
- Inyección de dependencias con strategy.
- Intercepciones con proxy.
- Gestión del ciclo de vida de los componentes con singleton y prototype.
- Instanciación de componentes de aplicación con factory.
- Implementación de una arquitectura orientada a mensajes con command y observer.

Trabajo práctico : Implementación de estos patrones con Microsoft.Extensions.DependencyInjection.

4) Acceso a datos y transacciones

- Introducción a los diferentes tipos de bases de datos (SQL, NoSQL).

PARTICIPANTES

Este curso está dirigido a desarrolladores.

REQUISITOS PREVIOS

Buenos conocimientos de programación C# y del framework .NET. Se requiere experiencia en el desarrollo de aplicaciones .NET.

COMPETENCIAS DEL FORMADOR

Los expertos que imparten la formación son especialistas en las materias tratadas. Han sido validados por nuestros equipos pedagógicos, tanto en el plano de los conocimientos profesionales como en el de la pedagogía, para cada curso que imparten. Cuentan al menos con entre cinco y diez años de experiencia en su área y ocupan o han ocupado puestos de responsabilidad en empresas.

MODALIDADES DE EVALUACIÓN

El formador evalúa los progresos pedagógicos del participante a lo largo de toda la formación mediante preguntas de opción múltiple, escenificaciones de situaciones, trabajos prácticos, etc.
El participante también completará una prueba de posicionamiento previo y posterior para validar las competencias adquiridas.

MEDIOS PEDAGÓGICOS Y TÉCNICOS

• Los medios pedagógicos y los métodos de enseñanza utilizados son principalmente: ayudas audiovisuales, documentación y soporte de cursos, ejercicios prácticos de aplicación y ejercicios corregidos para los cursillos prácticos, estudios de casos o presentación de casos reales para los seminarios de formación.
• Al final de cada cursillo o seminario, ORSYS facilita a los participantes un cuestionario de evaluación del curso que analizarán luego nuestros equipos pedagógicos.
• Al final de la formación se entrega una hoja de presencia por cada media jornada de presencia, así como un certificado de fin de formación si el alumno ha asistido a la totalidad de la sesión.

MODALIDADES Y PLAZOS DE ACCESO

La inscripción debe estar finalizada 24 horas antes del inicio de la formación.

ACCESIBILIDAD DE LAS PERSONAS CON DISCAPACIDAD

¿Tiene alguna necesidad específica de accesibilidad? Póngase en contacto con la Sra. FOSSE, interlocutora sobre discapacidad, en la siguiente dirección psh-accueil@orsys.fr para estudiar de la mejor forma posible su solicitud y su viabilidad.

- Principios ACID y gestión de transacciones.
- Frameworks de persistencia (EntityFrameworkCore, NHibernate). mini-ORM (Dapper): ventajas/desventajas Mejores prácticas.
- El patrón unidad de trabajo (unit of work).

Trabajo práctico : Acceso a datos de bases de datos relacionales desde una aplicación C#, aplicando transacciones.

5) API REST con AspNet Core

- Principios de diseño de las API REST (URI, mediatype, HATEOAS).
- Bases del protocolo HTTP.
- Creación de una API REST con ASP.NET MVC.
- Seguridad: autenticación mediante token con OpenID Connect.

Trabajo práctico : Desarrollo de una API REST para exponer la aplicación desarrollada anteriormente.

6) Interfaz gráfica de usuario con AspNet Core

- Recordatorio del patrón MVC.
- Vistas de Razor: acceso al modelo, internacionalización, gestión de excepciones.
- Autenticación por formulario, protección de rutas y vistas, protección contra ataques CSRF.

Trabajo práctico : Desarrollo de una interfaz gráfica de usuario para visualizar la aplicación desarrollada anteriormente.

7) Industrialización de los desarrollos

- Integración continua.
- Entrega continua.
- Creación de una imagen OCI con Docker.
- Despliegue en el orquestador Kubernetes.

Trabajo práctico : Creación de una imagen Docker y estudios de los descriptores de despliegue de Kubernetes.

FECHAS

Contacto