Digital House >
Coding School

Certified Tech Developer

The Ultimate Degree

Back End I

Objetivos de aprendizaje

Implementar un back end implica conocer un stack de tecnologías amplio, generalmente no es suficiente con conocer el lenguaje. Durante la cursada de esta materia, aprenderás los frameworks más usados del lenguaje mediante un stack de tecnologías para el desarrollo de aplicaciones web —desde el acceso a datos hasta la vista, a través de la implementación del patrón MVC y la exposición de servicios API REST—.

Metodología de enseñanza-aprendizaje

Desde Digital House proponemos un modelo educativo que incluye entornos de aprendizaje sincrónicos y asincrónicos con un enfoque que vincula la teoría y la práctica, mediante un aprendizaje activo y colaborativo.

Nuestra propuesta incluye clases en vivo con tu grupo de estudiantes y docentes, a los que podrás sumarte desde donde estés. Además, contamos con un campus virtual a medida, en el cual encontrarás las clases virtuales, con actividades, videos, presentaciones y recursos interactivos, para realizar a tu ritmo antes de cada clase en vivo.

A lo largo de tu experiencia de aprendizaje en Digital House lograrás desarrollar habilidades técnicas y blandas, como el trabajo en equipo, la creatividad, la responsabilidad, el compromiso, la comunicación efectiva y la autonomía.

En Digital House utilizamos la metodología de “Aula invertida”. ¿Qué quiere decir? Cada semana te vamos a pedir que te prepares para la que sigue, leyendo textos, viendo videos, realizando actividades, etcétera. De esta forma, cuando llegues al encuentro en vivo, estarás preparado para abordar el tema de manera más rica.

Utilizamos actividades y estrategias basadas en los métodos participativos y activos para ponerte en movimiento, ya que uno solo sabe lo que hace por sí mismo. Por ese motivo, organizamos las clases para que trabajes en ellas de verdad y puedas poner en práctica las distintas herramientas, lenguajes y competencias que hacen a la formación de un programador. Concebimos la clase como espacio de trabajo.

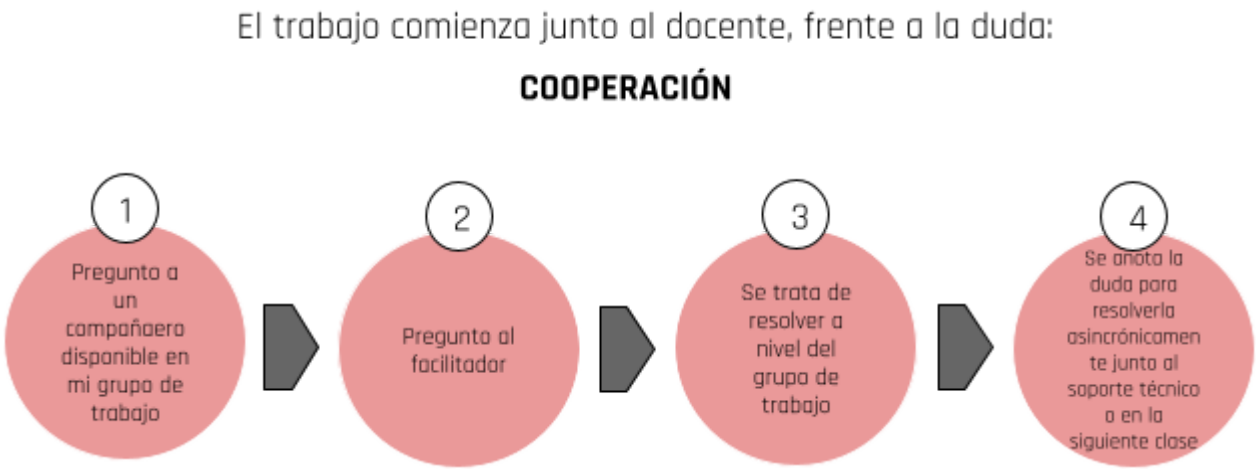
Una de las cuestiones centrales de nuestra metodología de enseñanza es el aprendizaje en la práctica. Por ese motivo, a lo largo de la cursada estarán muy presentes las ejercitaciones, es decir, la práctica de actividades de diversos tipos y niveles de complejidad que te permitirán afianzar el aprendizaje y comprobar que lo hayas asimilado correctamente. De esta forma, lograrás un aprendizaje más significativo y profundo, la asimilación de los conocimientos de manera más eficaz y duradera, relacionar lo aprendido con la realidad de los desarrolladores web, fomentar la autonomía y el autoconocimiento, mejorar el análisis, la relación y la comprensión de conceptos. Todas estas herramientas te ayudarán a ejercitar una multitud de competencias.

El aprendizaje entre pares es uno de los elementos centrales de nuestra metodología, por eso, en cada clase te propondremos que trabajes en mesas de trabajo junto a tus compañeros. A lo largo de la cursada iremos variando la composición de los grupos para potenciar la cooperación. Lo que proponemos es un cambio de mirada sobre el curso en cuestión: ya no se contempla al estudiante transitando su camino académico de manera individual, sino como parte de un equipo que resulta de la suma de las potencialidades de cada uno. La distribución en grupos de trabajo fomenta la diversidad y el aprovechamiento del potencial de cada integrante para mejorar el rendimiento del equipo.

La explicación recíproca como eje del trabajo cotidiano no solo facilita el aprendizaje de los compañeros, sino que sobre todo potencia la consolidación de conocimientos por parte de quien explica. Se promueve la responsabilidad, la autonomía, la proactividad, todo en el marco de la cooperación. Esto lleva a resignificar la experiencia de aprendizaje y a vincularla con emociones positivas.

El trabajo cooperativo permite entablar relaciones responsables y duraderas, aumenta la motivación y compromiso y promueve un buen desarrollo cognitivo y social. La cooperación surge frente a la duda. En caso de tener una pregunta, le consultarás a algún miembro de su grupo asignado que esté disponible. Si la duda continúa, se convocará al facilitador. Si no lo

resuelven, el facilitador pedirá a todos que se detengan para cooperar como equipo en la resolución del conflicto que ha despertado la duda. Así debatirán todos los integrantes de la mesa buscando la solución. Si aun así no pueden resolverlo, anotarán la duda, que será abordada asincrónicamente por el soporte técnico o de forma sincrónica en la siguiente clase por parte del profesor.



Todos los días, finalizada la jornada, los estudiantes reconocerán a uno de los integrantes del grupo con quienes compartió ese día. El criterio para ese reconocimiento es la cooperación.
Cada grupo tendrá un facilitador que será elegido a partir de los reconocimientos y generando un sistema de rotación donde cualquiera puede pasar por ese rol. El facilitador no es una figura estática, sino que cumple un rol dinámico y versátil. El facilitador es un estudiante que moviliza el alcance de los objetivos comunes del equipo poniendo en juego la cooperación. Es quien comparte con la mesa su potencial en favor del resto del equipo, y que por lo tanto promueve la cooperación.

Información de la materia

- Modalidad 100% a distancia.
- Cantidad de semanas totales: 9
- Cantidad de encuentros sincrónicos semanales: 6
- Clases virtuales en nuestro campus Playground: 54
- Cantidad de clases en vivo: 54

Requisitos y correlatividades

Para cursar Back End I se necesita tener aprobadas: Programación Orientada a Objetos y Base de Datos. Asimismo, para realizar las materias de especialización de back end es necesario haber realizado Back end I.

Modalidad de trabajo

Nuestra propuesta educativa está diseñada especialmente para esta modalidad 100% a distancia, mediante un aprendizaje activo y colaborativo siguiendo nuestro pilar de "aprender haciendo".

Los entornos de aprendizaje son tanto sincrónicos como asincrónicos, con un enfoque que vincula teoría y práctica, por lo que ambas están presentes en todo momento.

Contamos con un Campus virtual propio en el cual vamos a encontrar actividades, videos, presentaciones y recursos interactivos con instancias de trabajo individual y en equipo para profundizar en cada uno de los conceptos.

Además, realizaremos encuentros online y en vivo con el grupo de estudiantes y docentes, a los que podremos sumarnos desde donde estemos a través de una plataforma de videoconferencias con nuestra cámara y micrófono para generar una experiencia cercana.

Metodología de evaluación

La evaluación formativa es un proceso continuo que genera información sobre la formación de nuestros estudiantes y de nosotros como educadores.

A su vez, se genera conocimiento de carácter retroalimentador, es decir, tiene una función de conocimiento, ya que nos permite conocer acerca de los procesos de enseñanza y aprendizaje. También tiene una función de mejora continua porque nos permite saber en qué parte del proceso nos encontramos, validar si continuamos por el camino planificado o necesitamos tomar nuevas decisiones para cumplir los objetivos propuestos.

Por último, la evaluación desempeña un papel importante en términos de promover el desarrollo de competencias muy valiosas.

Nuestro objetivo es correrno de la evaluación tradicional, donde muchas veces resulta un momento difícil, aburrido y tenso. Para ello, vamos a utilizar la gamificación, una técnica donde se aplican elementos de juego para que el contenido sea más atractivo y te sientas más motivado e inmerso en el proceso, utilices los contenidos de aprendizaje como retos que realmente quieras superar y aprendas del error.

A su vez, para registrar dicha formación, utilizamos un conjunto de instrumentos, para los cuales es fundamental utilizar la mayor variedad posible y técnicas de análisis.

Criterios de aprobación

- Realizar las actividades de Playground (80% de completitud).
- Asistencia a los encuentros sincrónicos (90% de asistencia).
- Obtener un puntaje de 7 o más en la evaluación final.
- Obtener un puntaje de 7 o más en la nota final de la materia.

Contenidos

Módulo 1: Patrones de diseño

Abordaremos los patrones de diseño más utilizados en la capa de back end. Existe una numerosa variedad de librerías y frameworks que los implementan y su conocimiento es crucial para comprender y hacer un uso óptimo del stack de tecnologías necesarias para acelerar el desarrollo.

Clase 1: Bienvenida

- Introducción a la materia
- Pruebas unitarias con JUnit
- Introducción: Test unitarios vs. Test integración
- Beneficios y principio FIRST
- Arquitectura JUnit
- Configuración de la librería
- Anotaciones, assertions

- Testeo parametrizado
- Suite, Suite de Suites
- Ejemplo en Java

Clase 2: Patrón template method

- Template method
- Presentación en UML
- Ejemplo
- Implementación típica en Java

Clase 3: Integradora

Clase 4: Patrón cadena de responsabilidad

- Cadena de responsabilidad
- Presentación en UML
- Ejemplo
- Implementación típica en Java

Clase 5: Patrón proxy

- Patrón proxy
- Presentación en UML
- Ejemplo
- Implementación típica en Java

Clase 6: Integradora

Clase 7: Patrón flyweight

- Patrón flyweight
- Presentación en UML
- Ejemplo

- Implementación típica en Java

Clase 8: Patrón facade

- Patrón Facade
- Presentación en UML.
- Ejemplo
- Implementación típica en Java

Clase 9: Integradora

Módulo 2: Logging y acceso a datos

Estudiaremos las tres capacidades clave que debe poseer todo back end. Estas consisten en lograr una buena trazabilidad en el logueo para la rápida detección de errores (troubleshooting), el testeo de las piezas de software que vamos desarrollando para asegurar la calidad en etapas tempranas y el acceso a base de datos relacionales a través de buenas prácticas.

Clase 10: Logging (trazas y debug)

Presentar la importancia de contar con un framework de logueo. Concepto de trazabilidad y troubleshooting. Utilización de log4, configuraciones y opciones.

- Introducción al logueo (ventajas, desventajas, evaluación framework de logueo)
- Arquitectura Log4J
- Instalación de Log4J
- Crear archivo de configuración Log4J
- Level object, logger
- Appender objects (ConsoleAppender, FileAppender, RollingFileAppender, DailyRollingFileAppender, AsyncAppender)
- Ejemplo en Java

Clase 11: Acceso a base de datos

JDBC. Statement. Creación de conexiones. Resultset. Inserciones, borrado, actualizaciones y consultas. Manejo de transacciones.

- Puesta a punto de H2 Database
- Arquitectura JDBC
- Conexión a una base de datos
- Statement
- Insert y delete

Clase 12: Integradora

Clase 13: Consultas y transacciones sobre base de datos

- Consultar datos (execute, ResultSet)
- Modificar datos (executeUpdate)
- Utilización de PreparedStatements
- Transacciones (commit, auto commit, Rollback)
- Invocar StoredProcedures
- Loguear en base de datos con JDBCAppender

Clase 14: Patrón DAO (Data Access Object)

Armado de capas para el acceso a datos con JDBC. Importancia de contar con una capa de acceso a los datos. Utilización del patrón template method en el patrón DAO. Reutilización. Introducción al concepto de ORM.

- DAO
- Presentación en UML
- Implementación típica en Java
    - Clases de acceso a datos

    - Utilización de factory y template method
    - Clase de servicio
    - Generación de tipos de excepciones por capas
- Testing unitario de cada capa
- Introducción al concepto de ORM

Clase 15: Integradora

Clase 16: Taller de coding y repaso

Clase 17: Evaluación 1

Clase 18: Maven

Introducción a la gestión de proyectos en Java, su mecanismo y repositorio, gestión de dependencias, información de proyecto y plugins.

- Introducción a Maven, ¿qué es?
- Instalación de Maven
- Modelo conceptual de un proyecto en Maven
- POM (Project Object Model)
- Super POM
- Dependencias.
- Ciclo de vida.

Clase 19: Serialización de objetos y E/S archivos

Procesamiento de los flujos (streams), y utilizarlos adecuadamente. Serializar y deserializar objetos. Distinguir los lectores y escritores de los flujos, y seleccionar apropiadamente entre ellos. Manejo de E/S de archivos.

- Jerarquía de clases de flujo (inputStream, outputStream)

- Java NIO, Manejo de archivos, carpetas y recursos.
- Creación, escritura, lectura, eliminación, copiado y movimiento de ficheros.
- Serialización de objetos
- Manejo de carpetas

Clase 20: Integradora

Módulo 3: Framework para el desarrollo ágil de aplicaciones

Aprenderemos los conocimientos necesarios para utilizar uno de los frameworks más utilizados en las compañías para facilitar y acelerar el desarrollo de aplicaciones empresariales. A partir de este aprendizaje, contaremos con un set de herramientas para realizar un acceso a datos más simple, rápido, seguro y elegante —a través de un ORM—, la construcción de APIs y vistas web —mediante la utilización del patrón MVC—. Desarrollaremos un trabajo integrador para llevar a la práctica la correcta utilización del framework.

Clase 21: Introducción a Spring Boot

Introducción y primeros pasos con el framework para el desarrollo de aplicaciones Java.

- Arquitectura cliente-servidor
- Protocolo HTTP
- Framework
    - Definición
    - Framework vs. Librería
    - Introducción a Spring Framework
    - Spring vs. Spring Boot
- Crear un proyecto en Spring Boot
    - Initializr
    - Estructura del proyecto

- Aplication.java

Clase 22: Patrón MVC

- Modelo Vista Controlador
- Objetivo MVC
- Spring MVC
- Spring Boot
- Vista

Clase 23: Integradora

Clase 24: Taller de coding sobre el trabajo integrador

Construcción por parte de los alumnos del resto de las entidades con la capa de acceso a datos (DAO, Service) con testeos unitarios.

Clase 25: API REST

- ¿Qué es una API?
- Introducción, enfoque y diferencias con un monolito
- REST vs. SOAP
- API REST / RESTFul
- JSON
- Instalación Postman

Clase 26: Integradora

Clase 27: API REST II

- Anotaciones en el controller.
- DTO.

- ResponseEntity
- Consumir API

Clase 28: Consumir APIs

- Invocando una API desde la vista
- Invocando una API desde el backend

Clase 29: Integradora

Clase 30: Inyección de dependencias

- Inyección de dependencias
- IoC Inversion of control
- Formas de Inyectar dependencias

Clase 31: ORM

- Arquitectura JPA
- Hibernate
- Anotaciones
- Configurar Hibernate con Spring Boot

Clase 32: Integradora

Clase 33: Spring Data - Hibernate

- Relaciones (1 a 1, 1 a muchos, muchos a muchos)
- JoinColumn
- Cascada
- Fetch Type

Clase 34: HQL

- Características
- Sintaxis
- Ubicación de consultas en Spring

Clase 35: Integradora

Clase 36: Taller de coding aplicando lo aprendido hasta ahora

Repaso por parte del docente de la construcción de la capa de acceso a datos (Repository), capa de negocio (service), APIs (controller) y la capa de la vista (HTML con JavaScript) de una entidad y su mapeo con la base de datos, mostrando los cambios con respecto al DAO original.

Clase 37: Taller de coding sobre el trabajo integrador

Construcción por parte de los alumnos de la capa de acceso a datos (Repository), negocio (service), APIs (controller) y vista (HTML con JavaScript) de una entidad y su mapeo con la base de datos.

Clase 38: Base de datos no relacionales: MongoDB

- Introducción a MongoDB
- MongoDB vs. Base de datos relacionales
- Spring Data + MongoDB
- Creación de una base MongoDB
- Acceso a MongoDB

Clase 39: Integradora

Clase 40: Sistemas distribuidos

- ¿Qué es un microservicio?
- Patrones de comunicación entre microservicios
- Monolito vs. Microservicios

Clase 41: Manejo de excepciones

- ControllerAdvice
- ExceptionHandler
- Exception Handling para servicios RESTFul

Clase 42: Integradora

Clase 43: Seguridad

- Librería starter de Spring Security
- Autenticación y seguridad de URL
- Encriptar contraseña
- @PreAuthorize y @PostAuthrorize

Clase 44: Autenticación de APIs basada en tokens JWT

- Jason Web Token (JWT)
- Construcción de una API de Generación de Token

Clase 45: Integradora

Clase 46: Pruebas de integración

- Test de integración con Spring Boot
- Pruebas de integración para controllers con Mock

- Cobertura de código

Clase 47: Taller de coding sobre nuestro trabajo integrador

Añadir seguridad al trabajo integrador mediante un formulario de login de spring security

Clase 48: Taller de coding y entrega

Clase 49: Documentación

- Swagger OpenApi
- Hateoas

Clase 50: Despliegue de APIs en Docker

- Test de integración Cucumber
- Gherkin
- Maven Archetype
- Docker Compose

Clase 51: Integradora

Clase 52: Taller de coding sobre pruebas de integración

Clase 53: Estamos llegando al final...

Feedback general del trabajo integrador.

Clase 54: ¡Llegamos al final!

Repasamos los temas principales de la materia: clases de negocio, clases de acceso a datos, uso de ORM, clase de servicio, testeo unitario e integral, vista (templates) y controllers (MVC), exponer una API REST y llenar un combo con API REST.