

Full-stack

TechGuide - Alura, FIAP e PM3

Nivel 1

HTML - Fundamentos:

- HTML es un lenguaje de marcado que define la estructura de su contenido. HTML consta de una serie de elementos que se utilizan para que se vea o actúe de cierta manera. Las etiquetas de archivos adjuntos pueden vincular una palabra o imagen a otro lugar, pueden poner palabras en cursiva, pueden hacer que la fuente sea más grande o más pequeña, etc.
- Aprender qué etiquetas son necesarias para HTML básico
- Crear de un párrafo de texto
- Mostrar una imagen
- Conocer la diferencia entre 'h1', 'h2', 'h3', etc.
- Crear de un texto con hipervínculo
- Crear un formulario con campos relevantes
- Crear de una lista ordenada o desordenada de elementos
- Crear de una lista de elementos en una lista desplegable
- Vincular a un archivo CSS
- Crear una tabla
- Adicionar ID y clases

Contenidos

- **Web** HTML: Lenguaje de etiquetas de hipertexto (<https://developer.mozilla.org/es/docs/Web/HTML>)
- **Web** W3Schools: Formato de texto HTML (https://www.w3schools.com/html/html_formatting.asp)
- **Web** W3Schools: Bloque HTML y elementos en línea (https://www.w3schools.com/html/html_blocks.asp)
- **Web** W3Schools: Elementos y técnicas de diseño HTML (https://www.w3schools.com/html/html_layout.asp)
- **Web** W3Schools: Listas HTML (https://www.w3schools.com/html/html_lists.asp)
- **Web** W3Schools: Tablas HTML (https://www.w3schools.com/html/html_tables.asp)
- **Web** W3Schools: Formularios HTML (https://www.w3schools.com/html/html_forms.asp)
- **Web** MDN Web Docs: Conceptos básicos de HTML (https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics)
- **YouTube** ¿Qué es HTML? - 10 cosas que debes saber (<https://www.youtube.com/watch?v=tPzq8IufGxE>)
- **YouTube** Aprende HTML en 15 Minutos (<https://www.youtube.com/watch?v=mNbnV3aN3KA>)
- **YouTube** HTML - Ejemplos Prácticos básicos (<https://www.youtube.com/watch?v=NMh94GBcCQ>)

Contenidos Alura:

- **Artículo** Por una web más rápida: 26 técnicas de optimización de Sitios Web (<https://www.aluracursos.com/blog/por-una-web-mas-rapida-26-tecnias-de-optimizacion-de-paginas-web>)
- **Artículo** La semántica en HTML5 (<https://www.aluracursos.com/blog/la-semantica-en-html5>)
- **Artículo** HTML, CSS y Javascript, ¿cuáles son las diferencias? (<https://www.aluracursos.com/blog/html-css-javascript-cuales-son-las-diferencias>)
- **Artículo** Formulario con form validation de HTML5 (<https://www.aluracursos.com/blog/formulario-con-form-validation-de-html5>)
- **Artículo** Empezando con el desarrollo web Front-end (<https://www.aluracursos.com/blog/empezando-con-el-desarrollo-front-end>)
- **Artículo** Desde cero hasta programador front-end (<https://www.aluracursos.com/blog/desde-cero-hasta-programador-front-end>)
- **Artículo** VisualStudio Code: instalación, teclas de acceso directo, plugins e integraciones (<https://www.aluracursos.com/blog/visualstudio-code-instalacion-teclas-de-acceso-directo-plugins-e-integraciones>)
- **Artículo** Llenando un formulario HTML automáticamente con AJAX (<https://www.aluracursos.com/blog/llenando-un-formulario-html-automaticamente-con-ajax>)
- **Artículo** Vinculando elementos con HTML5 (<https://www.aluracursos.com/blog/vinculando-elementos-con-html5>)
- **Artículo** ¿Qué es HTML y sus tags? Estructura básica (<https://www.aluracursos.com/blog/html-y-sus-etiquetas>)
- **Artículo** ¿Qué es HTML y sus tags? Elementos inline (<https://www.aluracursos.com/blog/que-es-html-y-sus-etiquetas-parte2-elementos-inline>)
- **Artículo** ¿Qué es HTML y sus tags? Elementos a nivel de bloque (<https://www.aluracursos.com/blog/que-es-html-y-sus-tags-parte-3>)
- **Artículo** ¿Qué es HTML y sus tags? Elementos de un formulario (<https://www.aluracursos.com/blog/que-es-html-parte-4>)
- **Artículo** ¿Qué es HTML y sus tags? Atributos de los elementos. (<https://www.aluracursos.com/blog/ques-es-html-parte-5-atributos>)
- **Curso** HTML & CSS: Crea páginas increíbles con tecnologías web (<https://app.aluracursos.com/formacion-html-css>)

Javascript - Fundamentos:

- Javascript es el lenguaje de programación más popular del mundo y es una de las tecnologías centrales de la World Wide Web, junto con HTML y CSS. Tiene escritura dinámica, orientación a objetos basada en prototipos y funciones de primera clase. Es un paradigma múltiple que admite estilos de programación imperativos, funcionales e impulsados por eventos.
- Conocer los tipos primitivos
- Declarar variables, considerando la diferencia entre 'var', 'let' y 'const'
- Uso de estructuras condicionales ('if', 'else')
- Conocer los operadores de asignación y comparación ('=', '==', '===')
- Uso de estructuras de repetición y bucles ('while', 'for')
- Usar funciones, pasar parámetros y argumentos
- Manipulación de arreglos y listas
- Aprender el concepto de Orientación a Objetos

- Realizar un CRUD
- Obtener datos de una API
- Hacer llamadas asincrónicas usando 'Async/Await', 'Promise', etc.

Contenidos

- **Web** MDN Web Docs: JavaScript (<https://developer.mozilla.org/es/docs/Web/JavaScript>)
- **Web** MDN Web Docs: Guía de JS (<https://developer.mozilla.org/es/docs/Web/JavaScript/Guide>)
- **Web** MDN Web Docs: Gramática y tipos (https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Grammar_and_types)
- **Web** W3Schools: Comparación de JavaScript y operadores lógicos (https://www.w3schools.com/js/js_comparisons.asp)
- **Web** MDN Web Docs: Expresiones y operadores (https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Expressions_and_operators)
- **Web** MDN Web Docs: Array (https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Array)
- **Web** MDN Web Docs: IF y Else (<https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/if...else>)
- **Web** MDN Web Docs: While y For (<https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/while>)
- **Web** MDN Web Docs: Primeros pasos en JavaScript (https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/A_first_splash)

Contenidos Alura:

- **Artículo** Roadmap para principiantes en JavaScript (<https://www.aluracursos.com/blog/roadmap-js>)
- **Artículo** Guía de JavaScript: qué es y cómo aprender el lenguaje más popular del mundo (<https://www.aluracursos.com/blog/guia-de-javascript>)
- **Artículo** Funciones en JavaScript (<https://www.aluracursos.com/blog/articulo-funciones-en-javascript->)
- **Artículo** Empezar a programar es con JavaScript (<https://www.aluracursos.com/blog/empezar-a-programar-es-con-javascript>)
- **Artículo** Comenzando con Front-end (<https://www.aluracursos.com/blog/comenzando-con-front-end>)
- **Artículo** Organiza tu código Javascript de una manera fácil (<https://www.aluracursos.com/blog/organiza-tu-codigo-javascript-de-una-manera-facil>)
- **Artículo** Escopo en JavaScript (<https://www.aluracursos.com/blog/escopo-en-javascript>)
- **Artículo** Conociendo Arrow Functions (<https://www.aluracursos.com/blog/conociendo-arrow-functions>)
- **Artículo** Trabajando con fechas en JavaScript (<https://www.aluracursos.com/blog/trabajando-con-fechas-en-javascript>)
- **Artículo** Empezando con fetch en Javascript (<https://www.aluracursos.com/blog/empezando-con-fetch-en-javascript>)
- **Artículo** JavaScript replace: manipulando Strings y regex (<https://www.aluracursos.com/blog/javascript-replace-manipulando-strings-regex>)
- **Artículo** This, Getters y Setters en clases de Javascript (<https://www.aluracursos.com/blog/this-getters-y-setters-clases-de-javascript>)

- **Artículo** Empezando con fetch en Javascript (<https://www.aluracursos.com/blog/empezando-con-fetch-en-javascript>)
- **Artículo** JavaScript: ¿Cuándo debo usar forEach y map? (<https://www.aluracursos.com/blog/javascript-cuando-debo-usar-foreach-y-map>)
- **Artículo** ¿Cómo funciona el import y export de JavaScript? (<https://www.aluracursos.com/blog/como funciona-el-import-y-export-de-javascript>)
- **Artículo** Comprenda la diferencia entre var, let y const en JavaScript (<https://www.aluracursos.com/blog/comprenda-diferencia-entre-var-let-y-const-en-javascript>)
- **Artículo** Cómo usar operadores de comparación en Javascript (<https://www.aluracursos.com/blog/como-utilizar-operadores-de-comparacion-en-javascript>)
- **Artículo** Namespaces: cómo evitar conflictos en código JavaScript (<https://www.aluracursos.com/blog/namespaces-como-evitar-conflictos>)
- **Artículo** Una guía para importar y exportar módulos con JavaScript (<https://www.aluracursos.com/blog/una-guia-para-importar-exportar-modulos-con-java-script>)
- **Artículo** Javascript: ¿para qué sirve un array? (<https://www.aluracursos.com/blog/javascript-para-que-sirve-un-array>)
- **YouTube** ¿Qué es JavaScript? (https://www.youtube.com/watch?v=GJfOSoaXk4s&ab_channel=AluraLatam)
- **YouTube** El mejor lenguaje para empezar (<https://www.youtube.com/watch?v=Nrp3c6kNyAw>)
- **YouTube** Frameworks de Front End (https://www.youtube.com/watch?v=UNeKzI2WHgQ&ab_channel=AluraLatam)
- **YouTube** Bucles y Bucles Anidados en Javascript (<https://youtu.be/FEdqGdtZuwc>)
- **YouTube** De la Creación al Borrado: Descubre el Poder del CRUD en el Desarrollo de Aplicaciones (https://youtu.be/jr_98HCSTZc)
- **Curso** JavaScript: primeros pasos con el lenguaje (<https://app.aluracursos.com/course/javascript-primeros-pasos-lenguaje>)
- **Curso** JavaScript para Front-end (<https://aluracursos.com/formacion-javascript-para-front-end>)

Node.js - Fundamentos:

- Node.js es un entorno de ejecución de JavaScript que permite ejecutar aplicaciones desarrolladas en este lenguaje de manera autónoma, sin depender de un navegador.
- Aprender sobre operaciones bloqueantes y no bloqueantes
- Comprender el concepto de bucle de eventos (event loop)
- Aprender a utilizar las bibliotecas de Node.js, como 'net', 'fs', 'http', 'path', entre otras
- Entender cómo funcionan los temporizadores (Timers)

Contenidos

- **Web** Introduction to Node.js (<https://nodejs.dev/en/learn/>)
- **Artículo** Qué es Node.js: Casos de uso comunes y cómo instalarlo (<https://www.hostinger.es/tutoriales/que-es-node-js>)
- **Artículo** Qué es Node.js y por qué debería usarlo (<https://kinsta.com/es/base-de-conocimiento/que-es-node-js/>)
- **Artículo** Cómo escribir código asíncrono en Node.js (<https://www.digitalocean.com/community/tutorials/how-to-write-asynchronous-code-in-node-js-es>)

- **Artículo** Lectura de archivos con NodeJS y FS (<https://desarrolloweb.com/articulos/lectura-archivos-nodejs.html>)
- **Artículo** Crear servidor web HTTP en Node.js: Tutorial completo (<https://guru99.es/node-js-create-server-get-data/>)
- **Artículo** Módulo path en NodeJS (<https://desarrolloweb.com/articulos/modulo-path-nodejs>)
- **Artículo** Objetos en Node.js y Javascript. Lo realmente importante (<https://www.pensemoweb.com/objetos-node-js-javascript-lo-realmente-importante/>)
- **YouTube** ¿Qué es Node.js? Breve explicación animada (<https://www.youtube.com/watch?v=xJzzu7MVZXw>)
- **YouTube** ¿Qué se puede hacer con Node.JS? - Analizando Tecnologías (<https://www.youtube.com/watch?v=xL4oil0gQo>)
- **YouTube** Curso de Node.JS Completo desde cero (https://www.youtube.com/watch?v=yB4n_K7dZV8&list=PLUofhDIg_38qm2oPOV-IRTTEKyrVBaU7&index=1)
- **Podcast** Pedro: Un mes con Node.JS (https://www.ivoox.com/173-pedro-un-mes-node-js-audios-mp3_rf_669383371.html)

Contenidos Alura:

- **Curso** Formación Aprende a programar en JavaScript con enfoque en el Back-end (<https://www.aluracursos.com/formacion-js-backend>)

CSS - Fundamentos:

- Las hojas de estilo en cascada (CSS) son un lenguaje de hoja de estilo usado para descubrir una presentación de un documento escrito en un lenguaje de marca, como HTML o XML. O CSS puede ser usado para estilizar textos de documentos muy básicos — por ejemplo, para alterar a cor e o tamanho de cabeçalhos e links. Ele pode ser usado para criar um layout — por exemplo, transformando uma única columna de texto en um layout com uma área de conteúdo principal e uma barra lateral para información relacionada. Pode até ser usado para efectos como animação.
- Aprender a estructura visual de uma página, con 'margem' e 'preenchimento'
- Estabelecer o tamanho com 'larga' e 'altura'
- Aprender sobre la posición de un elemento ('estático', 'relativo' o 'absoluto')
- Aprender sobre la exposición de un elemento ('block', 'inline', 'inline-block')
- Aprender a posicionar imágenes en relación con el texto
- Aprender sobre alinhamento
- Aprender sobre estilo de fuente
- Aprender las diferencias y ventajas de usar las diferentes unidades de medida en CSS (% , relativo, etc.)
- Conectar-se a los elementos (IDs, clases) de un archivo HTML
- Cambiar las características de un elemento cuando se pasa el ratón sobre él
- Aprender a dimensionar cajas
- Aprender Flexbox
- Grado de aprendizaje

Contenidos

- **Web** CSS (<https://developer.mozilla.org/es/docs/Web/CSS>)
- **Web** W3Schools: Introducción a CSS (https://www.w3schools.com/css/css_intro.asp)

- **Web** W3Schools: Sintaxis CSS (https://www.w3schools.com/css/css_syntax.asp)
- **Web** W3Schools: CSS Selectores (https://www.w3schools.com/css/css_selectors.asp)
- **Web** W3Schools: CSS Box Model (https://www.w3schools.com/css/css_boxmodel.asp)
- **Web** W3Schools: Unidades CSS (https://www.w3schools.com/css/css_units.asp)
- **Web** MDN Web Docs: Comenzando con CSS (https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/Getting_started)
- **Web** MDN Web Docs: Cascada, especificidad y herencia. (https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Cascade_and_inheritance)
- **Web** MDN Web Docs: Propiedades de visualización (<https://developer.mozilla.org/en-US/docs/Web/CSS/display>)
- **Web** MDN Web Docs: Conceptos básicos de flexbox (https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox)
- **Web** MDN Web Docs: Conceptos básicos de Grid Layout (https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout/Basic_Concepts_of_Grid_Layout)
- **YouTube** Que rayos son las variables en CSS y como se utilizan. (https://www.youtube.com/watch?v=nJO1jjKUrol&ab_channel=FalconMasters)

Contenidos Alura:

- **Artículo** Bootstrap ¿Qué es, cómo y cuándo utilizar? (<https://www.aluracursos.com/blog/bootstrap-que-es-como-y-cuando-utilizar>)
- **Artículo** Comprenda la propiedad Position CSS (<https://www.aluracursos.com/blog/comprenda-la-propiedad-position-css>)
- **Artículo** CSS: Grids y tablas con responsividad en la Web (<https://www.aluracursos.com/blog/CSS-Grids-y-tablas-con-responsividad-en-la-Web>)
- **Artículo** Puntos de ruptura en el diseño responsivo (<https://www.aluracursos.com/blog/puntos-de-ruptura-en-el-diseno-responsivo>)
- **Artículo** Flexbox CSS: Guía Completo, Elementos y Ejemplos (<https://www.aluracursos.com/blog/flexbox-css-guia-completo-elementos-y-ejemplos>)
- **Artículo** ¿Por qué usar 'em' en su CSS hoy? (<https://www.aluracursos.com/blog/por-que-usar-em-en-tu-css-hoy>)
- **Artículo** Tu código CSS puede ser más limpio, flexible y reutilizable. (<https://www.aluracursos.com/blog/tu-codigo-css-puede-ser-mas-limpio-flexible-y-reutilizable>)
- **Artículo** Creando Layouts con Css Grid Layout (<https://www.aluracursos.com/blog/creando-layouts-con-css-grid-layout>)
- **Artículo** Creación de componentes CSS con el estándar BEM (<https://www.aluracursos.com/blog/creacion-de-componentes-css-con-el-estandar-bem>)
- **Artículo** Empezando a organizar tu CSS (<https://www.aluracursos.com/blog/empezando-a-organizar-tu-css>)
- **Artículo** Reset CSS: Qué es, Ejemplos, Cómo Crear y Utilizar (<https://www.aluracursos.com/blog/reset-css-que-es-ejemplos-como-crear-y-utilizar>)
- **Artículo** ¿Cómo lidiar con los límites de resolución en sitios responsivos? (<https://www.aluracursos.com/blog/como-lidiar-con-los-limites-de-resolucion-en-sitios-responsivos>)
- **Artículo** Cómo organizar el CSS en tu proyecto (<https://www.aluracursos.com/blog/como-organizar-el-css-en-tu-proyecto>)
- **Artículo** CSS: animaciones en Transition y Animation (<https://www.aluracursos.com/blog/css-animaciones-de-transition-y-animation>)

- **Artículo** Guía de Unidades en el CSS (<https://www.aluracursos.com/blog/guia-de-unidades-en-css>)
- **Artículo** Cambiando CSS con JavaScript (<https://www.aluracursos.com/blog/cambiando-css-con-javascript>)
- **YouTube** Animando un texto en HTML y CSS (https://www.youtube.com/watch?v=jZ1g5TiMA-M&ab_channel=AluraLatam)
- **YouTube** Box Model y Box sizing #AluraMás (<https://youtu.be/ts9qfCKamKg>)
- **YouTube** Consejos de CSS FlexBox para comenzar #aluramás (<https://youtu.be/EB4vWLzfVcl>)
- **YouTube** Como utilizar el Display block, inline, inline-block #aluramás (<https://youtu.be/AG2QssLpQzI>)
- **YouTube** ¡Domina CSS Flexbox con FlexboxFroggy! Aprende jugando en este desafío divertido (<https://youtu.be/MXPhyN5t0uQ>)
- **Curso** Flexbox: Posicione elementos en la pantalla (<https://www.aluracursos.com/curso-online-flexbox-posicione-elementos-pantalla>)
- **Curso** CSS Grid: Simplificando layouts (<https://www.aluracursos.com/curso-online-css-grid-simplificando-layouts>)
- **Curso** Arquitectura CSS: descomplicando los problemas (<https://www.aluracursos.com/curso-online-arquitectura-css-descomplicando-los-problemas>)

DOM - Fundamentos:

- El Document Object Model (DOM) es una interfaz de programación para documentos de la web. Representa la página para que los programas puedan cambiar la estructura, el estilo y el contenido del documento. El DOM representa el documento como nosotros y objetos; de esa forma, los lenguajes de programación pueden interactuar con la página.
- Entender cómo funciona el árbol DOM
- Accesando y manipulando elementos HTML y CSS
- Acceder a los padres e hijos de un elemento
- Insertar un nuevo elemento en el árbol
- Quitar un elemento del árbol
- Esperando un evento en un elemento determinado de la página usando

Contenidos

- **Web** W3Schools: JavaScript HTML DOM Methods (https://www.w3schools.com/js/js_htmldom_methods.asp)
- **Web** W3Schools: JavaScript HTML DOM Elements (https://www.w3schools.com/js/js_htmldom_elements.asp)
- **Web** W3Schools: JavaScript HTML DOM - Changing HTML (https://www.w3schools.com/js/js_htmldom_html.asp)
- **Web** W3Schools: JavaScript HTML DOM - Changing CSS (https://www.w3schools.com/js/js_htmldom_css.asp)
- **Web** W3Schools: JavaScript HTML DOM Events (https://www.w3schools.com/js/js_htmldom_events.asp)
- **Web** W3Schools: JavaScript HTML DOM EventListener (https://www.w3schools.com/js/js_htmldom_eventlistener.asp)
- **Web** W3Schools: JavaScript HTML DOM Elements (Nodes) (https://www.w3schools.com/js/js_htmldom_nodes.asp)

- **Web** MDN Web Docs: Introducción ao DOM (https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction)
- **Web** MDN Web Docs: querySelectorAll() (<https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelectorAll>)
- **Web** MDN Web Docs: addEventListener() (<https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener>)
- **Artículo** An introduction to the JavaScript DOM (<https://medium.com/free-code-camp/an-introduction-to-the-javascript-dom-512463dd62ec>)
- **Artículo** The DOM of Javascript (<https://javascript.plainenglish.io/the-dom-of-javascript-848506ebf386>)
- **YouTube** DOM JavaScript (https://www.youtube.com/watch?v=8u4Xef5YtFE&ab_channel=Bluuweb)
- **YouTube** Javascript manipulacion del DOM (https://www.youtube.com/watch?v=iaLiOXXR8eI&list=PLg9145ptuAijiWx4ixGBCZB0kh6YEeNqf&ab_channel=yacklyon)

Contenidos Alura:

- **Artículo** Qué es DOM? (<https://www.aluracursos.com/blog/que-es-dom>)
- **Curso** JS en la Web: Manipulación del DOM con JavaScript (<https://app.aluracursos.com/course/js-web-manipulacion-dom-javascript>)

Crear una aplicación React:

- Create React App es una forma oficialmente admitida de crear aplicaciones React de una sola página. Ofrece una configuración de construcción moderna sin configuración.
- Estructuración de un nuevo proyecto React
- Crear una aplicación funcional desde cero

Contenidos

- **Web** Empezando con Create React App (<https://create-react-app.dev/docs/getting-started>)
- **Web** MDN Web Docs: Primeros pasos con Reaccionar (https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started#setting_up_your_first_react_app)
- **Artículo** ¿Qué hace realmente Create React App por ti? (<https://medium.com/rewrite-tech/what-does-create-react-app-actually-do-for-you-c7a9354acdc4>)
- **Artículo** Cómo configurar un nuevo proyecto Create React App (<https://medium.com/@dimterion/how-to-set-up-a-new-create-react-app-project-2bfc2992aa16>)

Contenidos Alura:

- **Web** Actualización de React y uso de Vite (<https://app.aluracursos.com/extra/alura-mais/actualizacion-de-react-y-uso-de-vite-c235>)
- **Artículo** React: componentes con Styled Components (<https://www.aluracursos.com/blog/react-componentes-con-styled-components>)
- **Artículo** React: ¿Biblioteca o Framework? (<https://www.aluracursos.com/blog/react-framework-biblioteca>)

Conceptos de Orientación a Objetos:

- La Programación Orientada a Objetos es un paradigma de programación de software basado en la composición e interacción entre diversas unidades llamadas 'objetos' y las clases, que

contienen una identidad, propiedades y métodos. Se basa en cuatro componentes de la programación - abstracción digital, encapsulación, herencia y polimorfismo.

- Cómo funcionan los objetos
- Crear y utilizar constructores
- Qué son las clases
- Crear y utilizar métodos
- Cómo funciona la encapsulación
- Qué es la herencia
- Qué es el polimorfismo
- Cómo funcionan las interfaces
- Qué son las abstracciones

Contenidos

- **Web** Programación orientada a objetos (C#) (<https://learn.microsoft.com/es-es/dotnet/csharp/fundamentals/tutorials/oop>)
- **Web** Programación orientada a objetos - IBM (<https://www.ibm.com/docs/es/spss-modeler/saas?topic=language-object-oriented-programming>)
- **Web** Introducción a los objetos JavaScript (<https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects>)
- **Artículo** Introducción a POO con JavaScript ES6 (<https://medium.com/academia-hack/introducci%C3%B3n-a-poo-con-javascript-es6-80074fde0cdf>)
- **Artículo** Programación Orientada a Objetos (<https://ellibrodepython.com/programacion-orientada-a-objetos-python>)
- **YouTube** 4 Principios de la Programación Orientada a Objetos (<https://www.youtube.com/watch?v=Uk1C1NARZjU>)
- **YouTube** Mini Curso: POO con PHP (básico) (https://youtu.be/Ben_VC2rm10)
- **YouTube** 🟡 ¿Qué es la Programación Orientada a Objetos en PYTHON? - [Con EJEMPLOS] | Python desde CERO #13 (<https://www.youtube.com/watch?v=KwT1F7uL5rA>)

Contenidos Alura:

- **YouTube** Alura Latam: ¿Qué es la Programación Orientada a Objetos? (<https://www.youtube.com/watch?v=Oigen2sjagk>)
- **Artículo** POO: ¿Qué es la programación orientada a objetos? (<https://www.aluracursos.com/blog/poo-que-es-la-programacion-orientada-a-objetos>)
- **Artículo** Revisando la Orientación a Objetos: encapsulación de Java (<https://www.aluracursos.com/blog/revisando-la-orientacion-a-objetos-encapsulacion-de-java>)
- **Artículo** ¿Qué es encapsulamiento? (<https://www.aluracursos.com/blog/Que-es-encapsulamiento>)
- **Artículo** Herencia en JavaScript (<https://www.aluracursos.com/blog/herencia-en-javascript>)
- **Artículo** Interfaces Gráficas con Eclipse WindowBuilder (<https://www.aluracursos.com/blog/interfaces-graficas-con-eclipse-windowbuilder>)
- **Artículo** Cómo no aprender Java y Orientación a Objetos: getters y setters (<https://www.aluracursos.com/blog/como-no-aprender-java-y-orientacion-a-objetos-getters-y-setters>)
- **Artículo** Ordenar una lista de objetos en Java (<https://www.aluracursos.com/blog/ordenar-una-lista-de-objetos-en-java>)

- **Curso** Formación Java Orientado a Objetos (<https://app.aluracursos.com/formacion-javaoo>)
- **Curso** JavaScript: Introducción a la Orientación a Objetos (<https://app.aluracursos.com/course/javascript-introduccion-orientacion-objetos>)
- **Curso** JavaScript: Herencia e Interfaces en Orientación a Objetos (<https://app.aluracursos.com/course/javascript-herencia-interfaces-orientacion-objetos>)
- **Curso** C# y Orientación a Objetos (<https://app.aluracursos.com/formacion-c-sharpe-orientacion-a-objetos>)
- **Curso** Python: comprensión de la Orientación a Objetos (<https://app.aluracursos.com/course/python-comprension-orientacion-objetos>)
- **Curso** Python: avanzando en la orientación a objetos (<https://app.aluracursos.com/course/python-avanzando-orientacion-objetos>)
- **Curso** Java Polimorfismo: Entendiendo herencia e interfaces (<https://app.aluracursos.com/course/java-parte-3-entendiendo-herencia-interfaces>)
- **Curso** Curso de JavaScript: Objetos (https://app.aluracursos.com/course/javascript-objetos?utm_source=gnarus&utm_medium=timeline)
- **Curso** Formación: Aprenda a programar en Python con orientación a objetos (<https://app.aluracursos.com/formacion-lenguaje-python>)

Estructura de Datos:

- En el contexto de los ordenadores, una estructura de datos es una forma específica de almacenar y organizar los datos en la memoria del ordenador para que esos datos puedan ser fácilmente recuperados y utilizados de forma eficiente cuando sea necesario posteriormente.
- Conocer las principales estructuras de datos
- Implementar las principales estructuras de datos

Contenidos

- **Artículo** ¿Qué es una estructura de datos en programación y para qué se utiliza? (<https://blog.soyhenry.com/que-es-una-estructura-de-datos-en-programacion/>)
- **Web** Estructuras de datos - Documentación Python (<https://docs.python.org/es/3/tutorial/datastructures.html>)
- **Artículo** Estructuras de Datos y Algoritmos en Java (https://programacion.net/articulo/estructuras_de_datos_y_algoritmos_en_java_309/4)
- **Artículo** Estructuras de datos con Java: un enfoque práctico (<http://hp.fcencias.unam.mx/~alg/estructurasDeDatos/>)
- **Artículo** Colecciones: ARRAY, SET y DICTIONARY en Swift (<https://www.swiftbeta.com/colecciones-array-set-y-dictionary-en-swift/>)
- **Artículo** Colecciones en Swift y su manejo de memoria (<https://medium.com/@grago/colecciones-en-swift-y-su-manejo-de-memoria-61a03e236dd>)
- **Artículo** Estructuras de datos en .NET con C# (<https://www.genesisrrios.com/es/blog/estructuras-de-datos-en-csharp/>)
- **YouTube** Estructuras de Datos | Primeros Pasos (<https://youtu.be/Df-sgxGzyTg>)
- **YouTube** Qué son las estructuras de datos (<https://youtu.be/oQ0Wkldr73E?list=PLTd5ehJ0goMTSK7RRAPBF4wP-Nj5DRvT>)
- **YouTube** ¿Qué son y cómo funcionan los árboles? | Ejemplo de implementación (<https://youtu.be/tBaOQeyXYqg>)
- **YouTube** Aprende: Estructura de Datos con Java (https://www.youtube.com/watch?v=_9ScDWpqhFE)

- **YouTube** Introducción - 1 - Estructuras de Datos en C# (https://www.youtube.com/watch?v=rqagJXbauyA&ab_channel=nicosiored)
- **YouTube** Estructuras de datos con Python en 8 minutos: Listas, Tuplas, Conjuntos y Diccionarios (<https://www.youtube.com/watch?v=v25-m1LOUiU>)
- **YouTube** Programación en Python | Colecciones | Diccionarios (<https://www.youtube.com/watch?v=vAy4IM7NLIQ>)

Contenidos Alura:

- **Artículo** Estructura de datos: introducción (<https://www.aluracursos.com/blog/estructura-de-datos-introduccion>)
- **Artículo** Conociendo tuplas en Python (<https://www.aluracursos.com/blog/conociendo-las-tuplas-en-python>)
- **Artículo** Listas en Python: operaciones básicas (<https://www.aluracursos.com/blog/listas-de-python-operaciones-basicas>)
- **Artículo** Estructura de datos: computación práctica con Java (<https://www.aluracursos.com/blog/estructura-de-datos-computacion-practica-con-java>)
- **Artículo** Python: trabajando con diccionarios (<https://www.aluracursos.com/blog/python-trabajando-con-diccionarios>)
- **Artículo** Iterando una lista en Java (<https://www.aluracursos.com/blog/iterando-una-lista-en-java>)
- **Curso** Python: avanzando en el lenguaje (<https://app.aluracursos.com/course/python-avanzando-lenguaje>)
- **Curso** C#: array y tipos genéricos (<https://app.aluracursos.com/course/csharp-array-tipos-genericos>)
- **Curso** Python Collections: listas y tuplas (https://app.aluracursos.com/course/python-collections-listas-tuplas?utm_source=gnarus&utm_medium=timeline)
- **Curso** Curso de Java: trabajar con listas y colecciones de datos (<https://www.aluracursos.com/curso-online-java-trabajar-listas-colecciones-datos>)
- **Curso** Java y java.util: Colecciones, Wrappers y Lambda expressions (<https://www.aluracursos.com/curso-online-java-util-colecciones-wrappers-lambda-expressions>)

Java - Fundamentos:

- Java es un lenguaje de programación ampliamente utilizado para codificar aplicaciones web. Java es un lenguaje multiplataforma, orientado a objetos y centrado en red que se puede utilizar como una plataforma en sí. Es un lenguaje de programación rápido, seguro y confiable para codificar todo, desde aplicaciones móviles y software empresarial hasta aplicaciones de big data y tecnologías de servidor.
- Conocer los tipos primitivos.
- Declarar variables, considerando los diferentes tipos.
- Usar estructuras condicionales ('if', 'Else').
- Conocer a los operadores de comparación.
- Utilizar estructuras de repetición y bucles ('while', 'for').
- Usar funciones, pasar parámetros y argumentos.
- Manipular métodos.
- Manipular arrays y listas.
- Obtener datos de una API.

- Hacer llamadas asíncronas 'Future', etc.
- Crear constructores.

Contenidos

- **Podcast** Qué se necesita para ser programador junior en java (<https://open.spotify.com/episode/0sBiBS6TpdHjRUzm1oQm2O?si=b2a759c65ca44dd5>)
- **Artículo** ¿Qué es la tecnología Java y por qué la necesito? (https://www.java.com/es/download/help/whatis_java.html)


Contenidos Alura:

- **Artículo** Java: una guía para iniciar en esta tecnología (<https://www.aluracursos.com/blog/java-una-guia-para-iniciar-tecnologia>)
- **YouTube** ¿Por qué utilizar Java actualmente? (<https://www.youtube.com/watch?v=3kNuK-XAHEY&t=1s>)
- **YouTube** La magia detrás de Java (<https://www.youtube.com/watch?v=GrEO8nZzyZM&t=34s>)
- **Artículo** ¿Cómo empezar a desarrollar en java? (<https://www.aluracursos.com/blog/como-empezar-a-desarrollar-en-java>)
- **Artículo** Las características más destacables de Java 8 en adelante (<https://www.aluracursos.com/blog/caracteristica-destacables-java8-delante>)
- **Artículo** Java: Conozca el método main (<https://www.aluracursos.com/blog/java-conozca-el-metodo-main>)
- **Artículo** Importando clases en Java (<https://www.aluracursos.com/blog/Importando-clases-en-java>)
- **Artículo** Como hacer un import static en java (<https://www.aluracursos.com/blog/como-hacer-un-import-static-en-java>)
- **Artículo** Recibiendo datos en Java (<https://www.aluracursos.com/blog/recibiendo-datos-en-java>)
- **Curso** Java: creando tu primera aplicación (https://app.aluracursos.com/course/java-creando-primer-a-aplicacion?utm_source=gnarus&utm_medium=timeline)

Python - Fundamentos:

- Python es un lenguaje de programación de alto nivel de uso general, ampliamente utilizado en aplicaciones web, desarrollo de software, ciencia de datos y aprendizaje automático. Su filosofía de diseño se centra en la legibilidad del código mediante el uso de sangría significativa. Python es de tipado dinámico y cuenta con un recolector de basura.
- Los tipos de datos primitivos.
- Declarar variables, teniendo en cuenta los diferentes tipos.
- Utilizar estructuras condicionales ('if', 'else').
- Conocer los operadores de asignación y comparación.
- Usar estructuras de repetición y bucles ('while', 'for').
- Utilizar funciones, pasando parámetros y argumentos.
- Manipular métodos.
- Manipular arrays y listas.
- Obtener datos de una API.
- Crear constructores.
- Utilizar funciones anónimas.

Contenidos

- **Web** Documentación Python (<https://docs.python.org/es/3/tutorial/>)
- **Web** ¿Qué es Python? - AWS (<https://aws.amazon.com/es/what-is/python/>)
- **Artículo** El Manual de Python (<https://www.freecodecamp.org/espanol/news/el-manual-de-python/>)
- **Artículo** Variables y tipos de datos básicos en Python (<https://soka.gitlab.io/blog/post/2022-07-19-python-vars-intro-numbers/>)
- **Artículo** Sentencia If Else de Python: Explicación de las sentencias condicionales (<https://www.freecodecamp.org/espanol/news/sentencia-if-else-de-python-explicacion-de-las-sentencias-condiciones/>)
- **Artículo** Pasos iniciales para utilizar la biblioteca Requests de Python (<https://www.digitalocean.com/community/tutorials/how-to-get-started-with-the-requests-library-in-python-es>)
- **YouTube** Manejo de entrada y salida de datos | Curso de Python desde cero 🐍 (<https://www.youtube.com/watch?v=Sei1sltfocw>)
- **YouTube** CURSO de PYTHON 2020 🐍 CONSTRUCTORES (<https://www.youtube.com/watch?v=ElObT5rOx2M>)
- **YouTube** Curso Python 🐍 | Tipos de datos en Python, Enteros, Float, String, Booleanos  (https://www.youtube.com/watch?v=R3eQs-AsB_w)

Contenidos Alura:

- **Artículo** Python - Una introducción al Lenguaje (<https://www.aluracursos.com/blog/python-una-introduccion-al-lenguaje>)
- **Artículo** ¿Qué es Python? Historia, sintaxis y una guía para iniciarse en el lenguaje (<https://www.aluracursos.com/blog/que-es-python-historia-guia-para-iniciar>)
- **Artículo** Trabajando con precisión en números decimales en Python (<https://www.aluracursos.com/blog/precision-numeros-decimales-python>)
- **Artículo** Python datetime: trabajando con fechas (<https://www.aluracursos.com/blog/python-datetime-trabajando-con-fechas>)
- **Artículo** Listas en Python: operaciones básicas (<https://www.aluracursos.com/blog/listas-de-python-operaciones-basicas>)
- **Artículo** ¿Cómo comparar objetos en Python? (<https://www.aluracursos.com/blog/como-comparar-objetos-en-python>)
- **YouTube** Descubre el poder de Python: El lenguaje de programación que revoluciona la industria (<https://www.youtube.com/watch?v=BxcMMgmLKTU>)
- **Curso** Curso de Python: comenzando con el lenguaje (<https://www.aluracursos.com/curso-online-python-comenzando-con-lenguaje>)
- **Curso** Curso de Python: funciones incorporadas (<https://app.aluracursos.com/course/python-funciones-incorporadas>)
- **Curso** Formación Aprenda a programar en Python con orientación a objetos (<https://www.aluracursos.com/formacion-lenguaje-python>)

Accesibilidad en Javascript:

- La Accesibilidad Digital es la eliminación de barreras en la Web. El concepto presupone que los sitios y portales sean diseñados de modo que todas las personas puedan percibir, entender, navegar e interactuar de manera efectiva con las páginas.
- Escribir código teniendo en cuenta la accesibilidad.

Contenidos

- **Web** MDN Web Docs: Buenas prácticas de accesibilidad CSS y JavaScript (https://developer.mozilla.org/es/docs/Learn/Accessibility/CSS_and_JavaScript)
- **YouTube** Accesibilidad web: qué es y cómo aplicarla correctamente (<https://youtu.be/OkBlttAqCuo>)
- **Curso** Web.dev: Aprende sobre accesibilidad! (<https://web.dev/learn/accessibility/welcome?hl=es-419>)

Contenidos Alura:

- **Artículo** Inclusión y accesibilidad en América Latina (<https://www.aluracursos.com/blog/inclusion-y-accesibilidad-en-america-latina>)
- **Artículo** Directrices de Accesibilidad web (<https://www.aluracursos.com/blog/directrices-de-accesibilidad-web>)
- **Artículo** Accesibilidad web y el posicionamiento en motores de búsqueda (<https://www.aluracursos.com/blog/accesibilidad-web-y-el-posicionamiento-en-motores-de-busqueda>)
- **Artículo** 6 malas prácticas que perjudican usuarios disléxicos (<https://www.aluracursos.com/blog/6-malas-practicas-que-perjudican-usuarios-dislexicos>)
- **Artículo** Unobtrusive JavaScript (<https://www.aluracursos.com/blog/unobtrusive-javascript>)
- **YouTube** Accesibilidad Web | #Aluramás (<https://youtu.be/ngMOsuZL-XE>)
- **Curso** Formación Accesibilidad Web (<https://app.aluracursos.com/formacion-accesibilidad-web>)

Angular - Fundamentos:

- Angular es un framework para construir aplicaciones y una plataforma de desarrollo construida en TypeScript para crear aplicaciones eficientes y sofisticadas de página única (SPA).
- Construir interfaces utilizando HTML, CSS y TypeScript
- Crear aplicaciones SPA
- Construir aplicaciones web, mobile o desktop
- Integrar datos con API REST
- Utilizar la composición para crear componentes reutilizables
- Utilizar servicios de tipo Resolver
- Manipular solicitudes creando servicios de tipo Interceptor

Contenidos

- **Web** Introduction to the Angular docs (<https://angular.io/docs>)
- **Web** Introducción a la Documentación de Angular (<https://docs.angular.lat/docs>)
- **Artículo** Todo sobre Angular, la clave para acelerar tu desarrollo web (<https://www.thepowermba.com/es/blog/todo-sobre-angular>)
- **Artículo** Componentes en Angular (<https://medium.com/notasdeangular/componentes-en-angular-f25138b00c83>)
- **YouTube** Curso Angular en Español - Tutorial de Angular desde cero 📺 🍷 (<https://www.youtube.com/watch?v=i-oYrcNtc2s>)
- **YouTube** Diferencias entre Angular y AngularJs 2020 | Son lo mismo? (<https://www.youtube.com/watch?v=fMDiF6cXD28>)
- **YouTube** Programación Reactiva 🤖 4 conceptos para empezar (<https://www.youtube.com/watch?v=kkqn2Z2tl1k>)

Contenidos Alura:

- **Artículo** Creando aplicaciones Angular con Angular CLI (<https://www.aluracursos.com/blog/creando-aplicaciones-con-angular-cli>)
- **YouTube** Frameworks de Front End - Edición especial (https://www.youtube.com/watch?v=UNeKzI2WHgQ&ab_channel=AluraLatam)
- **Curso** Formación Angular (<https://app.aluracursos.com/formacion-angular>)

Nivel 2

Node.js - Express:

- Express es un framework flexible de aplicaciones web para Node.js que proporciona un conjunto robusto de recursos para aplicaciones web y móviles.
- Utiliza el framework Express para la creación de APIs REST con Node.js
- Administra peticiones de diferentes verbos HTTP en distintas URLs
- Define el puerto a utilizar para la conexión y la ubicación de los modelos que se utilizan para renderizar la respuesta
- Crea manejadores de rutas utilizando el método 'router'
- Conoce la biblioteca 'Router' y sus verbos HTTP, como 'get', 'post', 'put', etc.
- Define endpoints con route paths

Contenidos

- **Web** Express.js - Documentación Oficial (<https://expressjs.com/es/>)
- **Web** Express.js- Direccionamiento (<https://expressjs.com/es/guide/routing.html>)
- **Web** P+F de ExpressRoute (<https://learn.microsoft.com/es-es/azure/expressroute/expressroute-faqs>)
- **Artículo** Introducción a Express/Node (https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction)
- **Artículo** Introducción a Node Js y Express Js (<https://diego000.com/express-js-y-node-js-breve-introduccion/>)
- **Artículo** Qué es Express.JS y primeros pasos (<https://ifgeekthen.nttdata.com/es/que-es-expressjs-y-primeros-pasos>)
- **Artículo** Rutas en ExpressJS (<https://ull-esit-dsi-1617.github.io/estudiar-las-rutas-en-expressjs-alejandro-raul-3512/rutasexpressjs.html>)
- **Artículo** Construyendo aplicaciones seguras con Node.js y Express (<https://medium.com/@diego.coder/construyendo-aplicaciones-seguras-con-node-js-y-express-61a941085ead>)
- **Artículo** Integra otros frameworks en Express.js (<https://firebase.google.com/docs/hosting/frameworks/express?hl=es-419>)
- **Artículo** Renderizando React.js en el server con Express.js y react-engine (<https://medium.com/@sergiodxa/renderizando-react-js-en-el-server-con-express-js-y-react-engine-903de08c3df6>)
- **Artículo** Middleware en Express JS (<https://medium.com/@aarnlpezsosa/middleware-en-express-js-5ef947d668b>)
- **Artículo** API REST de datos geográficos con Node.js y Express (<https://medium.com/@pasoriano/api-rest-de-datos-geográficos-con-node-js-y-express-5242cfd400ee>)

- **Artículo** Desarrollando una REST API usando NodeJS, Express y Typescript (<https://medium.com/@ismaelbautista/developing-a-rest-api-using-nodejs-express-and-typescript-37391345b90e>)
- **Artículo** Estructura de una API Rest con NodeJS, Express y MongoDB (<https://medium.com/williambastidasblog/estructura-de-una-api-rest-con-nodejs-express-y-mongodb-cdd97637b18b>)
- **YouTube** Introducción a Express.js (<https://www.youtube.com/watch?v=VmH4tPbbDsM>)
- **YouTube** Por Qué Usar Express en Node.JS 🤔 Express vs HTTP Module (<https://www.youtube.com/watch?v=ccsm8vGFMLA>)
- **YouTube** Desarrollando una API con Express desde cero (<https://www.youtube.com/watch?v=YmZE1HXjpd4>)
- **YouTube** Qué es una Ruta en Express 🤓 Router y Middleware (<https://www.youtube.com/watch?v=j5UBvJhLA-k>)

React - Componentes:

- React te permite definir componentes como clases o funciones. Los componentes definidos como clases proporcionan más capacidades. Aceptan entradas arbitrarias (llamadas "accesorios") y devuelven elementos React que describen lo que debería aparecer en la pantalla.
- Comprender cómo funcionan los componentes
- Conociendo la biblioteca de componentes con estilo
- Comprender la diferencia entre clase y componentes funcionales

Contenidos

- **Web** Tu primer componente (<https://es.react.dev/learn/your-first-component>)
- **Web** Creando componentes en nuestra app de React (https://developer.mozilla.org/es/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_components)
- **Web** MDN Web Docs: Primeros pasos con React (https://developer.mozilla.org/es/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started)
- **Web** Importar y exportar componentes (<https://es.react.dev/learn/importing-and-exporting-components>)
- **YouTube** Tutorial de React Js - Mi primer componente (<https://www.youtube.com/watch?v=0LwhhotHyll>)
- **YouTube** Mejores Extensiones de Visual Studio Code (<https://www.youtube.com/watch?v=MNaMJ9bhwFI>)
- **YouTube** Estilos CSS para los componentes y la aplicación (<https://www.youtube.com/watch?v=ckiyt-JJCRk>)

Contenidos Alura:

- **Artículo** React: componentes con Styled Components (<https://www.aluracursos.com/blog/react-componentes-con-styled-components>)
- **Curso** React: ¡Crea aplicaciones web modernas con React! (<https://aluracursos.com/formacion-react>)

Node.js - ORM:

- Object-Relational Mapping (ORM), en español, mapeo objeto-relacional, es una técnica utilizada para mapear sistemas orientados a objetos y bases de datos relacionales, donde las tablas de la base de datos se representan como clases y los registros de las tablas serían instancias de esas clases.
- Entender qué son los ORMs y para qué se utilizan
- Conocer el SQL y sus sistemas de gestión de bases de datos
- Trabajar con Sequelize, un ORM para usar con Node.js
- Conocer otros ORMs de Node.js, como Prisma

Contenidos

- **Web** Sequelize - documentación oficial (<https://sequelize.org>)
- **Web** Prisma - documentación oficial (<https://www.prisma.io>)
- **Artículo** ¿Qué es un ORM? (<https://www2.deloitte.com/es/es/pages/technology/articles/que-es-orm.html>)
- **Artículo** Academia Hack: ORM (<https://medium.com/academia-hack/orm-3c26f4c34dba>)
- **Artículo** ¿Cuándo usar un ORM? (<https://www2.deloitte.com/es/es/pages/technology/articles/cuando-usar-orm.html>)
- **Artículo** Sequelize 101 (<https://medium.com/@khriztianmoreno/sequelize-101-5810bfa1332f>)
- **Artículo** Conexión a una base de datos PostgreSQL con Node.js y Sequelize (<https://medium.com/@diego.coder/conexión-a-una-base-de-datos-postgresql-con-node-js-y-sequelize-d93b0546e4cc>)
- **Artículo** Deploy, Node.js y base de datos con Sequelize (<https://community.listopro.com/deploy-node-js-y-base-de-datos-con-sequelize/>)
- **Artículo** Prisma, un toolkit para bases de datos (¿ORM?) para TypeScript y Node.js (<https://dev.to/denispixi/prisma-un-toolkit-para-bases-de-datos-orm-para-typescript-y-node-js-3g9>)
- **YouTube** ORM vs SQL ¿Cuándo usar cada uno? (<https://www.youtube.com/watch?v=xUWxLUDf29k>)
- **YouTube** Usando ORM SEQUELIZE para crear un API con MYSQL en NODE.JS (<https://www.youtube.com/watch?v=3wzMVj7nxtI>)
- **YouTube** Cómo usar el mejor ORM para JavaScript / TypeScript - Prisma (<https://www.youtube.com/watch?v=5iqM9V-1a4c>)

JavaScript - Pruebas:

- La prueba de software es el proceso de evaluación y verificación de que un software realmente hace lo que debería hacer. Los beneficios de las pruebas incluyen la prevención de errores, la reducción de los costos de desarrollo y la mejora del rendimiento.
- Usar pruebas unitarias
- Usar pruebas de integración
- Usar pruebas de comportamiento (behavior)
- Usar mocks

Contenidos

- **Web** Testing (Pruebas) en Javascript (<https://john-florez.gitbook.io/reactjs-book/testing-pruebas-en-javascript>)

- **Artículo** ¿Qué es la prueba unitaria? (<https://www.nimblework.com/es/agile/pruebas-unitarias/>)
- **Artículo** ¿Qué son las pruebas unitarias y cómo llevar una a cabo? (<https://www.yeeply.com/blog/que-son-pruebas-unitarias/>)
- **Artículo** Los 4 pilares de una buena prueba unitaria (<http://code4apps.com/code/4-pilares-de-una-buena-prueba-unitaria/>)
- **Artículo** Angular Unit Testing (<https://medium.com/@vito1986/angular-unit-testing-8a1479079f84>)
- **Artículo** Cómo usar Testing en Angular con Jasmine y Karma (<https://digital55.com/blog/como-usar-testing-angular-jasmine-karma/>)
- **YouTube** Introducción a TDD (<https://youtu.be/eFED4Vljwwk>)
- **YouTube** Cómo hacer Unit Testing en Angular de forma sencilla (<https://youtu.be/0ucTEskLvso>)
- **YouTube** Pruebas unitarias con Angular 12 (<https://youtu.be/tlqfNVvsHc>)
- **YouTube** Introducción a la Automatización de Pruebas de Software (https://youtu.be/R_hh3jAqn8M)
- **YouTube** Tipos y Niveles de Pruebas de Software. (<https://youtu.be/hSxXuRx9mo>)
- **YouTube** Objetivos de la Automatización de pruebas | ISTQB TAE (https://youtu.be/_nwkhYhbN6Y)

Contenidos Alura:

- **Artículo** Tipos de pruebas: ¿cuáles son las principales y por qué utilizarlas? (<https://www.aluracursos.com/blog/tipos-de-pruebas-cual-utilizar>)

JavaScript- Callbacks y Promises:

- Una Promesa (Promises) es un proxy de un valor que no necesariamente se conoce cuando se crea la promesa. Esto permite que los métodos asíncronos devuelvan valores como los métodos síncronos- en lugar de devolver inmediatamente el valor final, el método asíncrono devuelve la promesa de proporcionar el valor en algún momento en el futuro.
- Una función de devolución de llamada (Callback) es una función que se pasa a otra función como argumento, que luego se invoca dentro de la función externa para completar algún tipo de rutina o acción.
- Una función asíncrona es una función declarada con la palabra clave asíncrona y la palabra clave espera está permitida dentro de ella. Las palabras clave async y await permiten que el comportamiento asíncrono basado en promesas se escriba en un estilo más claro, lo que evita la necesidad de configurar explícitamente cadenas de promesas.
- Comprender el concepto de programación asíncrona
- Escribir código asíncrono entendiendo el concepto de promesas en JavaScript
- Usar métodos, palabras clave y objetos de JavaScript para manejar promesas como 'Async/Await', '.then()', 'Promise', etc.
- Aprender en qué situaciones necesitas usar la programación asíncrona
- Llamar a las API con 'fetch()'

Contenidos

- **Web** Funciones callbacks (<https://lenguajejs.com/javascript/asincronia/callbacks/>)
- **Web** ¿Qué son las promesas? (<https://lenguajejs.com/javascript/asincronia/promesas/>)
- **Web** ¿Qué es la asincronía? (<https://lenguajejs.com/javascript/asincronia/que-es/>)
- **Web** MDN Web Docs: Promises (https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Promise)

- **Web** MDN Web Docs: Función Callback (https://developer.mozilla.org/es/docs/Glossary/Callback_function)
- **Web** MDN Web Docs: Introducción a JavaScript asíncrono (<https://developer.mozilla.org/es/docs/Learn/JavaScript/Asynchronous/Introducing>)
- **Web** MDN Web Docs: Función async (https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/async_function)
- **Artículo** Formas de manejar la asincronía en JavaScript (<https://carlosazaustre.es/manejando-la-asincronia-en-javascript>)
- **Artículo** Cómo usar la API Fetch de JavaScript para obtener datos (<https://www.digitalocean.com/community/tutorials/how-to-use-the-javascript-fetch-api-to-get-data-es>)
- **Artículo** Tutorial de Fetch API en JavaScript con ejemplos de JS Fetch, Post y Header (<https://www.freecodecamp.org/espanol/news/tutorial-de-fetch-api-en-javascript-con-ejemplos-de-js-fetch-post-y-header/>)
- **YouTube** Callbacks vs Promises en JavaScript. ¡Entiende las diferencias y la importancia de cada una (<https://www.youtube.com/watch?v=frm0CHyeSbE>)
- **YouTube** Carlos Azaustre: ¿Cómo funcionan las Promises y Async/Await en JavaScript? (<https://www.youtube.com/watch?v=rKK1q7nFt7M>)
- **YouTube** Cómo CONSUMIR una API REST con JAVASCRIPT y Fetch + Promises (https://www.youtube.com/watch?v=FJ-w0tf3d_w)

Contenidos Alura:

- **Curso** JS en la Web: CRUD con JavaScript asíncrono (<https://aluracursos.com/course/js-web-crud-javascript-asincrono>)

JavaScript - Manejo de errores:

- El manejo de errores se refiere a los procedimientos de respuesta y recuperación de las condiciones de error presentes en una aplicación de software. En otras palabras, es el proceso compuesto por la anticipación, detección y resolución de errores de aplicación, programación o comunicación.
- Conocer y manejar las excepciones más comunes
- Conocer los tipos de errores y en qué situaciones se pueden producir
- Comprender cómo Node.js maneja los errores
- Usar 'try' y 'catch' para el manejo de errores
- Aprender en qué ocasiones y cómo usar **throw**
- Creación de excepciones específicas según las necesidades de su aplicación

Contenidos

- **Web** MDN Web Docs: Flujo de control y manejo de errores (https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Control_flow_and_error_handling)
- **Artículo** Guía Definitiva para el Manejo de Errores en JavaScript (<https://kinsta.com/es/blog/errores-en-javascript/>)
- **Artículo** Errores personalizados, extendiendo Error (<https://es.javascript.info/custom-errors>)
- **YouTube** Depurar JavaScript con el Navegador y con Visual Studio Code. (<https://www.youtube.com/watch?v=CRXMi2ZkS8>)
- **YouTube** Manejar errores en JavaScript con Try/Catch (<https://www.youtube.com/watch?v=h9YunSGIBQ8>)

Contenidos Alura:

- **Artículo** Depurando aplicaciones en el navegador (<https://www.aluracursos.com/blog/articulo-depurando-errores-en-el-navegador-formatado>)

JavaScript - Modularización:

- Dividir partes del código en módulos
- Usar import y export

Contenidos

- **Web** Módulos JavaScript (<https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Modules>)
- **Artículo** Modularización en Javascript (<https://medium.com/@sebastianpaduano/modularizaci%C3%B3n-en-javascript-538bd6c75fa>)
- **Artículo** Programación Orientada a Objetos: Patrón Módulo y Clases de JS (<https://anamartinezaguilar.medium.com/programaci%C3%B3n-orientada-a-objetos-patr%C3%B3n-m%C3%B3dulo-y-clases-de-js-8fc2dab1a085>)
- **Artículo** Programación Orientada a Objetos en JavaScript - Explicado con ejemplos (<https://www.freecodecamp.org/espanol/news/programacion-orientada-a-objetos-en-javascript-explicado-con-ejemplos/>)
- **YouTube** Organiza el código JavaScript con módulos {ES6} (https://youtu.be/_Fmsf6U4Gtg)
- **YouTube** Javascript: Entendiendo los módulos (https://youtu.be/bHziHEUrA_w)
- **YouTube** Curso Javascript - #07 Módulos, import & export, export default (https://youtu.be/_X7mQyat63Y)

Contenidos Alura:

- **Artículo** ¿Cómo funciona el import y export de JavaScript? (<https://www.aluracursos.com/blog/como-funciona-el-import-y-export-de-javascript>)

Jest:

- Jest es un corredor de pruebas de JavaScript que le permite acceder al DOM a través de jsdom. Proporciona una gran velocidad de iteración combinada con funciones potentes como módulos de simulación y temporizadores para que pueda tener más control sobre cómo se ejecuta el código.
- Componentes de prueba

Contenidos

- **Web** Descripción general de las pruebas (<https://reactjs.org/docs/testing.html#gatsby-focus-wrapper>)
- **Web** Jest: Documentación (<https://jestjs.io/es-ES/docs/getting-started>)
- **Artículo** Probando Componentes en React Usando Jest: Lo Básico (<https://code.tutsplus.com/es/articles/testing-components-in-react-using-jest-the-basics--cms-28934>)
- **Artículo** Cómo testear los componentes de React (<https://www.freecodecamp.org/espanol/news/como-probar-los-componentes-de-react-la-guia-completa/>)
- **YouTube** Introducción a Unit Testing en React con Jest (<https://www.youtube.com/watch?v=n5qbzhZUMsY>)
- **YouTube** Testing Components en React con Jest paso a paso (<https://www.youtube.com/watch?v=EjJu3hcPSCY>)

JavaScript - Almacenamiento:

- Almacenar datos en el front-end con localStorage
- Manipular datos almacenados
- Persistir datos almacenados

Contenidos

- **Web** Almacenamiento del lado cliente (https://developer.mozilla.org/es/docs/Learn/JavaScript/Client-side_web_APIs/Client-side_storage)
- **Web** Usando la API de almacenamiento web (https://developer.mozilla.org/es/docs/Web/API/Web_Storage_API/Using_the_Web_Storage_API)
- **Artículo** Almacenamiento en el navegador con JavaScript y localStorage (<https://parzibyte.me/blog/2019/03/16/almacenamiento-en-navegador-javascript-localstorage/>)
- **Artículo** Almacenamiento local en JavaScript (https://manuais.iessanclemente.net/index.php/Almacenamiento_local_en_JavaScript)
- **Artículo** Almacenamiento local (https://ng-girls.gitbook.io/todo-list-tutorial-spanish/local_storage)
- **Artículo** LocalStorage, sessionStorage (<https://es.javascript.info/localstorage>)
- **Artículo** Persistencia de Datos en Javascript (LocalStorage) (<https://dev.to/maxwellnewage/persistencia-de-datos-en-javascript-localstorage-419m>)
- **YouTube** Qué es LocalStorage y cómo utilizarlo en ejemplo práctico (<https://youtu.be/xzNqwjvHsYg>)
- **YouTube** Guardar y obtener la lista del local storage (<https://youtu.be/XUHYQEJhVzs>)

Contenidos Alura:

- **Curso** JS en la Web: Almacenando datos en el navegador (<https://app.aluracursos.com/course/js-web-almacenando-datos-navegador>)

NextJS - Fundamentos:

- Next.js es un marco de desarrollo web de código abierto creado por Vercel que permite aplicaciones web basadas en React con representación del lado del servidor y generación de sitios web estáticos.
- Creación de interfaces web
- Disminución de los tiempos de carga de la página
- Representación de páginas del lado del servidor
- Mejorando el rendimiento en React
- Creación de rutas API con funciones sin servicio

Contenidos

- **Web** Documentación: NextJS (<https://nextjs.org/docs>)
- **Artículo** Next.js: ¿qué es y por qué debes usarlo? (<https://todoxampp.com/next-js-que-es-y-por-que-debes-usarlo/>)
- **Artículo** NextJS: ¿el futuro de la web? (<https://www.aplyca.com/blog/nextjs-el-futuro-web-que-es-nextjs>)
- **Artículo** Crear una aplicación Next.js para chat: Añadir Sendbird UIKit a una aplicación Next.js (<https://sendbird.com/es/blog/build-nextjs-app-for-chat-with-sendbird-uikit>)

- **Artículo** Cómo habilitar la renderización del lado del servidor para una aplicación React (<https://www.digitalocean.com/community/tutorials/react-server-side-rendering-es>)
- **YouTube** Aprendiendo NextJS, el framework de React con Server Side Rendering (<https://www.youtube.com/watch?v=2jxc8DMzt0I>)
- **YouTube** Como cambiar tu proyecto de React Js a Next Js. (<https://www.youtube.com/watch?v=ERW30xdsLMs>)
- **YouTube** NextJS en 5 minutos (<https://www.youtube.com/watch?v=kRV23b1hYzw>)

Python - Comunicación con APIs:

- Una API es una interfaz que los desarrolladores de software utilizan para programar la interacción con componentes o recursos de software fuera de su propio código. Una definición aún más simple es que una API es la parte de un componente de software que es accesible para otros componentes.
- Comprender qué es una API REST
- Conocer los comandos básicos de comunicación HTTP
- Entender qué es una API REST
- Saber cómo hacer solicitudes autenticadas
- Convertir objetos a JSON y viceversa
- Saber cómo utilizar las herramientas del paquete Requests.

Contenidos

- **Web** ¿Qué es una API? - AWS (<https://aws.amazon.com/es/what-is/api/>)
- **Web** ¿Cuál es la diferencia entre SDK y API? - AWS (<https://aws.amazon.com/es/compare/the-difference-between-sdk-and-api/>)
- **Web** API de correo electrónico para Python 3 (<https://cloud.google.com/appengine/docs/standard/python3/services/mail?hl=es-419>)
- **Web** ¿Qué es una API REST? (<https://www.ibm.com/es-es/topics/rest-apis>)
- **YouTube** qué son los APIs? (SOAP, REST, GraphQL) (<https://www.youtube.com/watch?v=rAylamS1Hco>)
- **YouTube** APRENDE A CONSUMIR LA API DE SPOTIFY EN 15 MINUTOS!! (<https://www.youtube.com/watch?app=desktop&v=9cz8Gvh0J7g>)
- **YouTube** Implementación de una API en Python para enviar email con Angular (https://www.youtube.com/watch?v=RSGQuH_gWNw)
- **YouTube** Automatizando API con Python Capítulo 1: Introducción (<https://www.youtube.com/watch?v=6r2M1FEMauw>)

Contenidos Alura:

- **Artículo** Buscando tweets con Python (<https://www.aluracursos.com/blog/buscando-tweets-con-python>)

Spring Framework:

- Spring es un framework de código abierto para la plataforma Java. Se trata de un marco no intrusivo, basado en los estándares de diseño (design patterns) de inversión de control (ioc) e inyección de dependencia. En Spring el contenedor se encarga de "instanciar" clases de una aplicación Java y definir las dependencias entre ellas a través de un archivo de configuración en formato XML, inferencias del framework, lo que es llamado de auto-wiring o incluso anotaciones en las clases, métodos y propiedades. De esta forma, Spring permite el bajo acoplamiento entre clases de una aplicación orientada a objetos.

- Entender el concepto de inyección de dependencias
- Entender el patrón MVC
- Usar Spring Data para manipular datos

Contenidos

- **Artículo** ¿Qué son Spring framework y Spring Boot? Tu primer programa Java con este framework (https://dev.to/campusmvp_es/que-son-spring-framework-y-spring-boot-tu-primer-programa-java-con-este-framework-4ao6)
- **Artículo** Que es la inyección de dependencias en Spring (<https://gustavopeiretti.com/spring-inyeccion-dependencias/>)
- **YouTube** Aprende Spring Boot de una vez por todas! #17 Inyección de Dependencias (https://www.youtube.com/watch?v=SEaDMUy0xKo&ab_channel=CodigoMorsa)
- **YouTube** Aprende Spring Boot de una vez por todas! #19 Beans (https://www.youtube.com/watch?v=ZCy8JzmZi8Q&ab_channel=CodigoMorsa)
- **Artículo** Que es el Modelo Vista Controlador (MVC) (<https://peznuuss.medium.com/que-es-el-modelo-vista-controlador-mvc-5b7ff4dbf85e>)
- **Artículo** Conversores personalizados para @RequestBody (<https://dev.to/gekyzo/conversores-personalizados-para-requestbody-37hp>)
- **Artículo** Tips para usar Spring JdbcTemplate (<https://medium.com/@matedeilo/tips-para-usar-spring-jdbcTemplate-486d250dc5a>)
- **YouTube** JPARepository en SPRING DATA - Tutorial Completo Fácil (https://www.youtube.com/watch?v=6Vy0rS9Obog&ab_channel=ProgramandoenJAVA)

Contenidos Alura:

- **YouTube** Spring Framework. ¿Qué es? #AluraMás (<https://www.youtube.com/watch?v=t-igt1b2qgk&t=60s>)
- **Artículo** Spring: Conozca este framework de Java (<https://www.aluracursos.com/blog/spring-conozca-framework-java>)
- **Artículo** Empezando con Spring Framework (<https://www.aluracursos.com/blog/empezando-con-spring-framework>)
- **Curso** Curso de Spring MVC: crea un web app con Thymeleaf y Bootstrap (<https://app.aluracursos.com/course/spring-mvc-crea-web-app-thymeleaf-bootstrap>)
- **Artículo** Programando tareas con Scheduled de Spring (<https://www.aluracursos.com/blog/programando-tareas-con-scheduled-de-spring>)

Cypress:

- Cypress es una herramienta de prueba Front-end que permite configurar, escribir, ejecutar y depurar pruebas.

Contenidos

- **Web** Cypress: Documentación (<https://docs.cypress.io/guides/overview/why-cypress>)
- **Web** Runebook - Cypress: Documentación (<https://runebook.dev/es/docs/cypress/>)
- **Artículo** Cypress: ¿Qué es? ¿Por qué y cómo lo usamos en Get on Board? (<https://medium.com/get-on-board-dev/cypress-qué-es-por-qué-y-cómo-lo-usamos-en-get-on-board-17688b453fdc>)
- **Artículo** Una guía práctica para las pruebas de extremo a extremo con Cypress (<https://mx.devoteam.com/expert-view/una-guia-practica-para-las-pruebas-de-extremo-a-extremo-con-cypress/>)

- **Artículo** Explorando Cypress: La guía definitiva basada en su documentación oficial (<https://www.acontracorrientech.com/explorando-cypress-la-guia-definitiva-basada-en-la-documentacion-oficial/>)
- **YouTube** Aprende Testing en Cypress como lo hace un Senior en la vida real 🍷 | FullStack Javascript Bootcamp (<https://www.youtube.com/watch?v=HDFNjDKKO6A>)
- **YouTube** PRUEBAS de API con CYPRESS IO | Tutorial de CYPRESS | Curso de CYPRESS (<https://www.youtube.com/watch?v=lppTzTFY0I8>)
- **YouTube** 🍷 Tutorial de Cypress | Primer Test en Cypress io | Cypress Español (<https://www.youtube.com/watch?v=puyyzaZuIKI>)

Nivel 3

Arquitectura de Microservicios:

- Los microservicios son un enfoque de arquitectura en el que el software consiste en pequeños servicios independientes que se comunican entre sí y se organizan de acuerdo con sus dominios de negocio.
- Aprender el concepto de arquitectura diseñada para microservicios
- Realizar la comunicación mediante API
- Mejorar la escalabilidad de un sistema

Contenidos

- **Web** Diseño de una aplicación orientada a microservicios (<https://learn.microsoft.com/es-es/dotnet/architecture/microservices/multi-container-microservice-net-applications/microservice-application-design>)
- **Artículo** Microservicios y arquitectura de microservicios (<https://www.intel.la/content/www/xl/es/cloud-computing/microservices.html>)
- **Artículo** Microservicios vs API: Entendiendo la diferencia (<https://kinsta.com/es/blog/microservicios-vs-api/>)
- **Artículo** Microservicios Ejemplo de Flujo (<https://medium.com/mycodebad/microservicios-ejemplo-de-flujo-f45720a9b278>)
- **Artículo** Microservicios Conceptos (<https://medium.com/mycodebad/microservicios-conceptos-55d19636873e>)
- **YouTube** Patrones fundamentales de la arquitectura microservicios (<https://www.youtube.com/watch?v=A7y23uU1NHk>)
- **YouTube** Un ejemplo de microservicios #CafeConRivas (<https://www.youtube.com/watch?v=qAcUGw7HhxM>)

Node.js - Autenticación y Tokens:

- JWT (JSON Web Token) es un método creado según el estándar RFC 7519 que representa una comunicación segura entre dos partes. Este token consta de tres partes: encabezado, carga útil y firma.
- Construir un sistema de autenticación utilizando tokens
- Comprender el funcionamiento del JSON Web Token (JWT)
- Crear una lista de permisos para almacenar tokens opacos
- Implementar métodos de actualización de tokens

Contenidos

- **Web** JWT - Documentación oficial (<https://jwt.io>)
- **Web** Usa tokens de OAuth de JWT (<https://cloud.google.com/apigee/docs/api-platform/security/oauth/using-jwt-oauth?hl=es-419>)
- **Web** JSON Web Token (JWT) for OAuth Client Authorization Grants (<https://www.ibm.com/docs/es/was-liberty/base?topic=uocpao2as-json-web-token-jwt-oauth-client-authorization-grants>)
- **Artículo** JSON Web Token (JWT) (<https://medium.com/@mendezj374/json-web-token-jwt-dc2cecbec7df>)
- **Artículo** Json Web Tokens (JWT), Consejos para usarlos (<https://medium.com/@chrramirez/json-web-tokens-jwt-consejos-para-usarlo-92eabb3d4320>)
- **Artículo** JSON Web tokens (JWT): claves para usarlos de manera segura (<https://www.bbva.com/es/innovacion/json-web-tokens-jwt-claves-para-usarlos-de-manera-segura/>)
- **Artículo** Protegiendo tu aplicación web con Spring Security y autenticación basada en Tokens JWT (<https://medium.com/@espinozajge/protegiendo-tu-aplicación-web-con-spring-security-y-autenticación-basada-en-tokens-jwt-1321cbe4c4c3>)
- **Artículo** Atacando JSON Web Token (JWT) (<https://ironhackers.es/tutoriales/atacando-json-web-token-jwt/>)
- **Artículo** Compartir datos entre sistemas de forma segura con JWT y encriptación simétrica (<https://medium.com/@albclvjmn/compartir-datos-entre-sistemas-de-forma-segura-con-jwt-y-encriptación-simétrica-c05e34db5b69>)
- **Artículo** Login + JWT (<https://bluuweb.github.io/node/07-jwt/#jwt>)
- **YouTube** ¿Qué es JWT? ¿Para que sirve? ¿Cuándo usarlo? ¿Cómo se usa? (<https://www.youtube.com/watch?v=tWQobKFQLG0>)
- **YouTube** ¿Qué es JWT? - Diferencias entre sesiones y tokens (<https://www.youtube.com/watch?v=k3pbXdwlodw>)
- **YouTube** JWT para Programar Backend Seguro (<https://www.youtube.com/watch?v=JKa22Z44ap8>)
- **YouTube** Autenticación con JWT y Cookies (<https://www.youtube.com/watch?v=DxYAcXiy-ak>)

Contenidos Alura:

- **Artículo** ¿Qué es JSON Web Token? (<https://www.aluracursos.com/blog/que-es-json-web-token>)



Contenedores:

- Los contenedores son paquetes de software que contienen todos los elementos necesarios para ejecutarse en cualquier entorno. La gestión de contenedores es un área crucial en la computación en nube y DevOps, que implica el uso de tecnologías para automatizar el proceso de creación, implementación, escalado y monitoreo de contenedores. Los contenedores son unidades de software estandarizadas que permiten a los desarrolladores empaquetar todas las dependencias de una aplicación (código, bibliotecas, configuraciones, etc.) en un solo paquete. Esto permite que la aplicación se ejecute de forma consistente en cualquier entorno de infraestructura.
- La tecnología de contenedores, como ejemplifica Docker, proporciona un entorno coherente y portátil para el desarrollo, las pruebas y la implementación de aplicaciones, lo que es vital para el trabajo eficiente de la ingeniería de datos. Además, Kubernetes, un sistema de organización de contenedores, permite la gestión, automatización y escalabilidad de aplicaciones basadas en

contenedores en entornos de producción. Dominar estos conceptos y tecnologías permite a los ingenieros de datos construir y mantener canalizaciones de datos eficientes y confiables.

- Kubernetes (también conocido como k8s o Kube) es una plataforma de orquestación de contenedores de código abierto que automatiza gran parte de los procesos manuales necesarios para implementar, gestionar y escalar aplicaciones en contenedores.
- Aislar el software para que funcione independientemente
- Implementación de software en clústeres
- Modularizar su sistema en paquetes más pequeños
- Conocer la plataforma Docker
- Conocer Kubernetes

Contenidos

- **Web** IBM - ¿Qué son los contenedores? (<https://www.ibm.com/mx-es/topics/containers>)
- **Web** Microsoft - ¿Qué es un contenedor? (<https://azure.microsoft.com/es-mx/resources/cloud-computing-dictionary/what-is-a-container>)
- **Artículo** Contenedores y la nube (<https://dev.to/roremdev/contenedores-y-la-nube-3je9>)
- **Artículo** Contenedores: cómo es el ciclo de vida de una aplicación en Kubernetes (<https://dev.to/campusmvp/contenedores-como-es-el-ciclo-de-vida-de-una-aplicacion-en-kubernetes-1ief>)
- **Artículo** ¿Qué diferencia hay entre Docker (Contenedores) y Máquinas virtuales (VMWare, VirtualBox...)? (<https://dev.to/campusmvp/qu-diferencia-hay-entre-docker-contenedores-y-maquinas-virtuales-vmware-virtualbox-4ji3>)
- **Artículo** El potencial de Kubernetes para las empresas (<https://jlcasal.medium.com/el-potencial-de-kubernetes-para-las-empresas-d36f9eece999>)
- **Artículo** ¿Qué es Docker y para que sirve? Explicación (https://dev.to/prox_sea/que-es-docker-y-para-que-sirve-explicacion-5h2n)
- **YouTube** ¿Que es un contenedor? (<https://www.youtube.com/watch?v=x5zGoICZLnU>)
- **YouTube** ¿Qué es Docker y los contenedores? (<https://www.youtube.com/watch?v=kkfZs0vJFyU>)
- **YouTube** Volúmenes y redes en Docker (https://www.youtube.com/watch?v=DIdeI70dFI&ab_channel=Programaci%C3%B3nB3nespa%C3%B1ol)
- **YouTube** ¿Qué es KUBERNETES?  Relación con DOCKER y CONTENEDORES  (<https://www.youtube.com/watch?v=V86eTbswdQo>)

Contenidos Alura:

- **Artículo** Empezando con Docker (<https://www.aluracursos.com/blog/empezando-con-docker>)
- **Artículo** Creando volúmenes con Docker (<https://www.aluracursos.com/blog/creando-volumenes-con-docker>)

TypeScript - Fundamentos:

- TypeScript es un lenguaje de programación fuertemente tipado que se basa en JavaScript.
- Comprender en profundidad qué son los tipos y la importancia de la programación tipificada
- Aprender qué es TypeScript, por qué se creó, cómo funciona y su relación con JavaScript
- Conocer las herramientas de TypeScript (integración con el editor de código, verificador estático y compilador)
- Escribir código en TypeScript usando sus herramientas (interfaces, enumeración, decoradores, etc.)

Contenidos

- **Web** W3Schools: Introducción a TypeScript (https://www.w3schools.com/typescript/typescript_intro.php)
- **Artículo** TypeScript, ¿qué es y cómo se diferencia de JavaScript? (<https://immune.institute/blog/typescript-que-es-como-se-diferencia-javascript>)
- **Artículo** Qué es TypeScript? (<https://codigofacilito.com/articulos/typescript>)
- **Artículo** ¿Qué es TypeScript y por qué debes aprenderlo? (<https://ed.team/blog/que-es-typescript-y-por-que-debes-aprenderlo>)
- **Artículo** Introducción a Typescript (<https://medium.com/@diego.coder/introducción-a-typescript-7add491e256>)
- **Artículo** TypeScript: qué es, diferencias con JavaScript y por qué aprenderlo (<https://profile.es/blog/que-es-typescript-vs-javascript/#>)
- **YouTube** TypeScript - Tutorial (https://www.youtube.com/watch?v=xtp_DuPxo9Q)

Contenidos Alura:

- **YouTube** Ventajas de trabajar con TypeScript #AluraMás (https://www.youtube.com/watch?v=bFrhUOhbo00&ab_channel=AluraLatam)
- **Curso** TypeScript parte 1: evolucionando tu JavaScript (<https://app.aluracursos.com/course/typescript-evolucionando-javascript>)
- **Curso** TypeScript parte 2: avanzando en el lenguaje (<https://app.aluracursos.com/course/typescript-parte-2-avanzando-lenguaje>)
- **Curso** Typescript parte 3: técnicas y buenas prácticas (<https://app.aluracursos.com/course/typescript-parte-3-tecnicas-buenas-practicas>)

WebSockets:

- WebSocket es una tecnología que permite la comunicación bidireccional a través de canales full-duplex sobre un único socket del Protocolo de Control de Transmisión (TCP). Está diseñado para ser ejecutado en navegadores y servidores web que admitan HTML5, pero puede ser utilizado por cualquier cliente o servidor de aplicaciones.
- Conocer el protocolo WebSocket y su uso en la comunicación cliente-servidor
- Aprender sobre los diversos usos de WebSocket en la web
- Crear aplicaciones que utilicen WebSockets con las APIs y bibliotecas de Node.js

Contenidos

- **Web** Documentación: WebSocket (<https://developer.mozilla.org/es/docs/Web/API/WebSocket>)
- **Web** Documentación: Escribir servidores WebSocket (https://developer.mozilla.org/es/docs/Web/API/WebSockets_API/Writing_WebSocket_servers)
- **Web** Documentación: Escribiendo aplicaciones con WebSockets (https://developer.mozilla.org/es/docs/Web/API/WebSockets_API/Writing_WebSocket_client_applications)
- **Web** Documentación: Crea conexiones persistentes con WebSockets (<https://cloud.google.com/appengine/docs/flexible/python/using-websockets-and-session-affinity?hl=es-419>)
- **Artículo** ¿Qué es WebSocket? (<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/que-es-websocket/>)
- **Artículo** ¿Qué son los WebSockets y cómo crearlos? (<https://appmaster.io/es/blog/que-son-los-websockets-y-como-crearlos>)

- **Artículo** ¿Cuándo debemos usar WebSockets? (<https://styde.net/cuando-debemos-usar-websockets/>)
- **Artículo** HTTP vs Websockets (<https://iagolast.github.io/blog/2017/06/18/http-vs-websockets.html>)
- **YouTube** Servidor websocket dual con node js (https://www.youtube.com/watch?v=rS1XnZ_CP2s)
- **YouTube** Diferencias entre AJAX, WebSockets y Server-sent Events (<https://www.youtube.com/watch?v=czp6WcCat8Q>)
- **YouTube** Long polling vs Websockets vs Server-Sent Events | Enviar datos desde un servidor web (<https://www.youtube.com/watch?v=QgMLSspb0ls>)
- **YouTube** Websockets con Node.js (<https://www.youtube.com/watch?v=nK418x0ymaw>)

JavaScript - Concurrencia:

- La programación concurrente es un paradigma de programación para construir programas que utilizan la ejecución simultánea de varias tareas computacionales interactivas, que pueden ser implementadas como programas separados o como un conjunto de hilos creados por un único programa
- Ejecutar tareas en paralelo

Contenidos

- **Web** Hilo principal (https://developer.mozilla.org/es/docs/Glossary/Main_thread)
- **Artículo** Programación multihilos (threads) en Javascript (<https://purojavascript.com/programacion-multihilos-threads-en-javascript/>)
- **Artículo** Multithreading con Javascript: Web Workers. (<https://blog.apside.cl/multithreading-con-javascript-web-workers/>)
- **Artículo** Introducción de Paralelo y Concurrente a la Programación en Python (<https://code.tutsplus.com/es/introduction-to-parallel-and-concurrent-programming-in-python--cms-28612a>)
- **YouTube** Hilos (Threads) en Java (<https://youtu.be/filrTdSKVcE>)
- **YouTube** Paralelismo con JavaScript | WebWorkers, multi-hilo. (https://youtu.be/lVd8K_ntKKk)
- **YouTube** Concurrencia vs. Paralelismo (<https://youtu.be/kMr3mF71Kp4>)

GraphQL:

- GraphQL es un nuevo estándar API que proporciona una alternativa más eficiente, potente y flexible a REST. Fue desarrollado y de código abierto por Facebook y ahora lo mantiene una gran comunidad de empresas e individuos de todo el mundo.
- Comprender cómo se utiliza GraphQL en el desarrollo de API
- Creación de API utilizando bibliotecas y marcos GraphQL

Contenidos

- **Web** GraphQL: Documentación (<https://graphql.org/>)
- **Web** Cómo GraphQL: el tutorial Fullstack para GraphQL (<https://www.howtographql.com/>)
- **Web** RedHat: ¿Qué es GraphQL? (<https://www.redhat.com/en/topics/api/what-is-graphql>)
- **Artículo** ¿Cómo construir una API GraphQL? (<https://blog.back4app.com/es/como-construir-una-api-graphql/>)
- **Artículo** Por qué GraphQL es el futuro de las APIs (<http://developinginspanish.com/2019/11/16/por-que-graphql-es-el-futuro-de-las-apis/>)

- **Artículo** GraphQL, el futuro de las API (<https://blog.ida.cl/experiencia-de-usuario/graphql-el-futuro-de-las-api/>)
- **Artículo** GraphQL vs REST: Todo lo que Necesitas Saber (<https://kinsta.com/es/blog/graphql-vs-rest/>)
- **YouTube** ¿Qué es GraphQL? (<https://www.youtube.com/watch?v=RreRD41qIpw>)
- **YouTube** miduver: GraphQL desde de cero (<https://www.youtube.com/watch?v=QG-qbmW-wes>)

Apollo Client:

- Apollo Client es una biblioteca integral de administración de estado para JavaScript que le permite administrar datos locales y remotos con GraphQL.
- Utilizar Apollo para crear un servidor GraphQL
- Conectar a una API

Contenidos

- **Web** Documentación: Cliente Apollo (<https://www.apollographql.com/docs/react/>)
- **Artículo** Apollo, GraphQL, y cómo Redux me arruga la ropa (<https://medium.com/@p4bloch/apollo-graphql-y-cómo-redux-me-arruga-la-ropa-12bd071fda9>)
- **Artículo** Apollo Client, ahora con React Hooks (<https://www.apollographql.com/blog/announcement/frontend/apollo-client-now-with-react-hooks/>)
- **YouTube** Cómo usar GraphQL en React 🧩 con Apollo Client (<https://www.youtube.com/watch?v=sVFocedf-iU>)
- **YouTube** Creando el frontend con React, React-Router, GraphQL y Apollo Client (<https://www.youtube.com/watch?v=xxUfQROMbzc>)
- **YouTube** Crea tu servidor GraphQL con Apollo Server (<https://www.youtube.com/watch?v=zDrKmi9ph2Q>)
- **YouTube** APRENDE a usar GRAPHQL en tu APP de REACT con APOLLO CLIENT 🚀 (<https://www.youtube.com/watch?v=sVFocedf-iU>)

Nest.js - Fundamentos:

- NestJS es un framework de Node con soporte total para TypeScript que se ejecuta sobre frameworks HTTP como expressJS o Fastify. Utiliza varios elementos de programación orientada a objetos y una serie de funcionalidades de TypeScript.
- Aprender qué es NestJS y por qué se utiliza
- Utilizar características específicas de NestJS, como proveedores (providers), módulos y controladores (controllers)
- Desarrollar APIs utilizando NestJS
- Utilizar la Interfaz de Línea de Comandos (CLI) de Nest.js

Contenidos

- **Web** NestJS - Documentación Oficial (en inglés) (<https://docs.nestjs.com>)
- **Artículo** Introducción a Nestjs (<https://medium.com/@diego.coder/introducción-a-nestjs-88f1ca90df35>)
- **Artículo** NestJS: qué es y por qué empezar a usarlo (<https://gfourmis.co/nestjs-que-es-y-por-que-empezar-a-usarlo/>)
- **Artículo** Ventajas de NestJS que están revolucionando el desarrollo web (<https://www.moveapps.cl/blog/ventajas-de-nestjs-2023/>)

- **Artículo** NestJS Tips (<https://dev.to/fjbatresv/nestjs-tips-35lf>)
- **Artículo** NestJS — Aplicando SOLID (<https://mugan86.medium.com/nestjs-aplicando-solid-357b80d4245c>)
- **Artículo** Creación de una API REST para MongoDB con NestJS (<https://ualmtorres.github.io/nestjs-mongodb-tutorial/>)
- **Artículo** NestJS y React: Tutorial de aplicación video stream (<https://todoxampp.com/nestjs-y-react-tutorial-de-aplicacion-video-stream/>)
- **YouTube** Introducción al framework de NodeJS (<https://www.youtube.com/watch?v=KQBibtZFqF8>)
- **YouTube** Nest.js de 0 a 100 (https://www.youtube.com/playlist?list=PLzHaXzj_WAym4WR3gBYuy1iew5T3NgL0v)
- **YouTube** Curso de Login y Registro Completo con React (<https://www.youtube.com/watch?v=q4ywr3eZmk0>)

Java - Kafka:

- Apache Kafka es una plataforma de transmisión de datos distribuida que es capaz de publicar, suscribir, almacenar y procesar flujos de registro en tiempo real. Esta plataforma está diseñada para procesar flujos de datos provenientes de diversas fuentes y entregarlos a varios clientes.
- Utilizar Kafka para la comunicación asíncrona
- Crear microservicios con Kafka
- Crear productores y consumidores
- Entender cómo usar Kafka para paralelismo y ejecución serializada
- Obtener garantías relativas al envío o entrega de los mensajes

Contenidos

- **YouTube** Apache KAFKA: Qué es en 1 minuto (<https://www.youtube.com/watch?v=bxIsNY8Au9I>)
- **Artículo** Qué es Apache Kafka y cómo dar los primeros pasos (<https://profile.es/blog/que-es-apache-kafka-primeros-pasos/>)
- **Podcast** ¿Qué es y cómo funciona Apache Kafka? Podcast Apasionados por la tecnología. (<https://open.spotify.com/episode/52RjBgXyclVSd2FHkkENQV?si=130fac7df95b4ec6>)
- **Web** Introducción a Apache Kafka (<https://oscarfmdc.medium.com/introducci%C3%B3n-a-apache-kafka-aprende-big-data-en-espa%C3%B1ol-ce7281fdf0cc>)
- **YouTube** Apache Kafka - Ejemplo de configuración de intermediario de Apache Kafka (<https://www.ibm.com/docs/es/oala/1.3.7?topic=collection-apache-kafka-broker-configuration-example>)
- **YouTube** Como comunicar microservicios y ejemplo con Apache Kafka y Javascript (https://www.youtube.com/watch?v=rHRSOWa0cqI&ab_channel=NullSafeArchitect)
- **YouTube** Instalación de un cluster de Apache Kafka paso a paso (https://www.youtube.com/watch?v=MA-nxL14fr4&ab_channel=NullSafeArchitect)

Habilidad Auxiliar: Infraestructura y buenas prácticas

Git y GitHub - Fundamentos:

- Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes, con rapidez y eficiencia.

- GitHub es un servicio de hosting para el desarrollo de software y el control de versiones mediante Git.
- Crear un repositorio
- Clonar un repositorio
- Comprometerse, empujar y tirar hacia y desde el repositorio
- Revertir un commit
- Crear de ramas y Pull requests
- Manejar fusiones y conflictos

Contenidos

- **Web** Git: Libro de Consulta (<https://git-scm.com/book/es/v2>)
- **Web** GitHub Documentación (<https://docs.github.com/es>)
- **Web** Github Pages Documentación (<https://docs.github.com/es/pages/getting-started-with-github-pages/about-github-pages>)
- **Web** W3Schools: Git Tutorial (<https://www.w3schools.com/git/default.asp?remote=github>)
- **Web** Git School - Visualizing Git (<https://git-school.github.io/visualizing-git/>)
- **Web** Dangit, Git!?! (<https://dangitgit.com/es>)
- **Artículo** Git and Github Quickstart Tutorial (<https://medium.com/@prashantramnyc/git-and-github-quickstart-tutorial-654a71594dca>)
- **YouTube** ¿Qué es Git y cómo funciona? (<https://www.youtube.com/watch?v=jGehuhFhtnE>)
- **YouTube** Git y Github | Guía Práctico de Git y Github Desde Cero (<https://www.youtube.com/watch?v=HiXLkL42tMU>)

Contenidos Alura:

- **Artículo** Git y Github: que son y primeros pasos (<https://www.aluracursos.com/blog/git-y-github-que-son-y-primeros-pasos>)
- **Artículo** Guía sobre cómo instalar Git en diferentes sistemas operativos (<https://www.aluracursos.com/blog/guia-sobre-como-instalar-git-en-diferentes-sistemas-operativos>)
- **Artículo** Iniciando un repositorio con Git (<https://www.aluracursos.com/blog/iniciando-repositorio-con-git>)
- **Artículo** Comenzando con Git: aprendiendo a versionar (<https://www.aluracursos.com/blog/comenzando-con-git>)
- **Artículo** Creando un repositorio remoto en GitHub (<https://www.aluracursos.com/blog/creando-repositorio-remoto-en-github>)
- **Artículo** Clonando un repositorio con Git y GitHub (<https://www.aluracursos.com/blog/clonando-un-repositorio-remoto>)
- **Artículo** Paso a Paso para activar tu proyecto en GitHub Pages. (<https://www.aluracursos.com/blog/github-pages>)
- **Artículo** Cómo escribir un README increíble en tu Github (<https://www.aluracursos.com/blog/como-escribir-un-readme-increible-en-tu-github>)
- **Artículo** Buenas practicas en git: evitando errores (<https://www.aluracursos.com/blog/como-evitar-errores-en-git>)
- **Artículo** GIT: Errores de comandos y directorios (<https://www.aluracursos.com/blog/git-errores-de-comandos-y-directorios>)
- **Artículo** GIT: errores de commits (<https://www.aluracursos.com/blog/git-errores-de-commits>)






- **Artículo** GIT: Errores de fusión (<https://www.aluracursos.com/blog/errores-de-fusion>)
- **Artículo** GIT: Errores con el remoto (<https://www.aluracursos.com/blog/errores-con-el-remoto>)
- **YouTube** Git y GitHub para Principiantes #AluraMás (https://www.youtube.com/watch?v=-LmFK6skG7s&ab_channel=AluraLatam)
- **YouTube** Git y GitHub: Herramientas Esenciales para el Control de Versiones y Colaboración en Desarrollo (<https://youtu.be/dw04N616Abw>)
- **YouTube** ¿Puedo subir mi proyecto a Github sin líneas de comando? - Git y Github para principiantes (https://www.youtube.com/watch?v=Yfm16Tlpcwk&ab_channel=AluraLatam)
- **YouTube** Git y GitHub: Herramientas Esenciales para el Control de Versiones y Colaboración en Desarrollo (<https://www.youtube.com/watch?v=dw04N616Abw>)
- **YouTube** ¿Puedo subir mi proyecto a Github sin líneas de comando? - Git y Github para principiantes (<https://www.youtube.com/watch?v=Yfm16Tlpcwk>)
- **Curso** Git y GitHub: repositorio, commit y versiones (<https://app.aluracursos.com/course/git-github-repositorio-commit-versiones>)
- **Curso** Git y Github: estrategias de ramificación, conflictos y Pull Requests (<https://www.aluracursos.com/curso-online-git-github-estrategias-ramificacion-conflictos-pull-requests>)

HTTP - Fundamentos:

- HTTP significa Protocolo de transferencia de hipertexto. La comunicación entre las computadoras cliente y los servidores web se realiza mediante el envío de solicitudes HTTP y la recepción de respuestas HTTP.
- Comprender la diferencia entre los verbos HTTP
- Probar solicitudes y verificar los códigos de estado en el navegador
- Aprendiendo a hacer una solicitud HTTP en la línea de comando con WGET
- Descargar una imagen con WGET
- Realización de una POST

Contenidos

- **Web** W3Schools: ¿Qué es HTTP? (https://www.w3schools.com/whatis/whatis_http.asp)
- **Web** MDN Web Docs: Una descripción general de HTTP (<https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>)
- **Web** MDN Web Docs: Métodos de solicitud HTTP (<https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>)
- **Web** MDN Web Docs: HTTP (<https://developer.mozilla.org/es/docs/Web/HTTP>)
- **Web** MDN Web Docs: Métodos de petición HTTP (<https://developer.mozilla.org/es/docs/Web/HTTP/Methods>)
- **Web** MDN Web Docs: Mensajes HTTP (<https://developer.mozilla.org/es/docs/Web/HTTP/Messages>)
- **Web** HTTP Cats (<https://http.cat/>)
- **Web** HTTP Dogs (<https://http.dog/>)
- **YouTube** Peticiones, Métodos Http y Códigos de estado. (<https://www.youtube.com/watch?v=gBK-Mfa0lw8>)
- **YouTube** SSL, TLS, HTTPS, HTTP - Explicado Fácilmente (https://www.youtube.com/watch?v=6HJAWFenYx8&ab_channel=ProfeSang)

-   Protocolo HTTP  Requests y Responses con: GET, POST, PUT, PATCH y DELETE | Desarrollo web  (<https://www.youtube.com/watch?v=l2MihYAj0lw>)
-  REST y los Verbos de HTTP (<https://www.youtube.com/watch?v=OHBHeAPoZ8E>)








Contenidos Alura:

-  Artículo HTTP: Desmitificando el protocolo Web (<https://www.aluracursos.com/blog/http-desmitificando-el-protocolo>)
-  Artículo ¿Cual es la diferencia entre HTTP y HTTPS? (<https://www.aluracursos.com/blog/cual-es-la-diferencia-entre-http-y-https>)
-  Artículo HTTP: Diferencias entre GET y POST (<https://www.aluracursos.com/blog/diferencias-entre-get-y-post>)
-  Artículo Métodos de petición HTTP (<https://www.aluracursos.com/blog/metodos-de-peticion-http>)
-  Curso HTTP: La base de internet (<https://app.aluracursos.com/course/http-base-internet>)






JSON:

- JSON significa Notación de objetos de JavaScript. Es un formato de texto para almacenar y transportar datos.
- Crear un objeto
- Transformar un objeto en una cadena
- Transformar una cadena en un objeto
- Manipular un objeto

Contenidos

-  Web MDN Web Docs: JSON (https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/JSON)
-  Web MDN Web Docs: Trabajando con JSON (<https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON>)
-  ¿Que és JSON y cómo funciona? (https://www.youtube.com/watch?v=z8qk7T_2sWg&ab_channel=SoyDalto)
-   ¿Qué es JSON? ¿Cuál es su SINTAXIS?  ¿Cómo crear un archivo JSON? | DESARROLLO WEB  (https://www.youtube.com/watch?v=RhxOTqFbl5Q&ab_channel=TodoCode)

Contenidos Alura:

-  Artículo ¿Que es Json? (<https://www.aluracursos.com/blog/que-es-json>)
-  Artículo ¿JSON y Objeto JavaScript son lo mismo? (<https://www.aluracursos.com/blog/json-y-objeto-javascript-son-lo-mismo>)
-  Artículo Simulando una API REST con json-server (<https://www.aluracursos.com/blog/simulando-una-api-rest-con-json-server>)
-  Artículo ¿Qué es JSON Web Token? (<https://www.aluracursos.com/blog/que-es-json-web-token>)
-  Curso JS en la Web: CRUD con JavaScript asíncrono (<https://www.aluracursos.com/course/online-js-web-crud-javascript-asincrono>)

Entrega e Integración Continuas (CI/CD):

- CI/CD es la abreviación de Continuous Integration/Continuous Delivery, traducido al español como "entrega e integración continuas". Se trata de una práctica de desarrollo de software que

tiene como objetivo hacer más eficiente la integración de código a través de compilaciones (builds) y pruebas automatizadas.

- Automatizar la integración de código entre diversas partes del equipo se ha vuelto cada vez más importante, ya que de esta manera es posible acelerar el desarrollo y reducir el tiempo de entrega del software
- Ejecutar pruebas automatizadas de la aplicación para verificar su funcionamiento
- Realizar la entrega de actualizaciones de manera automática y segura
- Realizar pruebas de conexión y pruebas de carga para evitar que la aplicación presente problemas al ser actualizada

Contenidos

- **Web** ¿Qué es la entrega continua? (<https://learn.microsoft.com/es-es/devops/deliver/what-is-continuous-delivery>)
- **Web** ¿Qué es la integración continua? (<https://aws.amazon.com/es/devops/continuous-integration/>)
- **Web** Acerca de la integración continua (<https://docs.github.com/es/actions/automating-builds-and-tests/about-continuous-integration>)
- **Artículo** ¿En qué consiste la integración continua? (<https://www.atlassian.com/es/continuous-delivery/continuous-integration>)
- **Artículo** La integración y la distribución continuas (CI/CD) (<https://www.redhat.com/es/topics/devops/what-is-ci-cd>)
- **Artículo** Integración continua y entrega continua (CI/CD) (<https://www.aplyca.com/blog/integracion-continua-y-entrega-continua-cicd>)
- **Artículo** ¿Qué es la entrega continua? (<https://www.servicenow.com/es/products/it-operations-management/what-is-continuous-delivery.html>)
- **Artículo** Continuous Delivery – Pipelines en el desarrollo de software (<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/continuous-delivery/>)
- **Artículo** Integración, Entrega y Despliegue continuo con Github Actions (<https://medium.com/contraslashsas/integración-entrega-y-despliegue-continuo-con-github-actions-f49001e49541>)
- **Artículo** Drone.io como motor de CI/CD (<https://medium.com/ingeniería-en-tranqui-finanzas/drone-io-como-motor-de-ci-cd-32a8d714320d>)
- **YouTube** ¿Qué es CI/CD? Integración y despliegue continuos (https://www.youtube.com/watch?v=88bnb9eTRNo&ab_channel=AminEspinoza)
- **YouTube** Taller de Integración Continua (CI) y Despliegue Continuo (CD) (https://www.youtube.com/watch?v=MeW_rSQb9k)
- **YouTube** Proceso de integración/despliegue continuo con Gitlab (<https://www.youtube.com/watch?v=QDUKU-5UgNM>)
- **YouTube** Despliegues de aplicacion CI/CD para principiantes 'procesadores' (<https://www.youtube.com/watch?v=ThlpD-Rys00>)
- **YouTube** Entrega Continua CI/CD - AWS + EKS + Github Actions (<https://www.youtube.com/watch?v=XZyBb5BE-4>)

Contenidos Alura:

- **Artículo** ¿Que es GitOps? (<https://www.aluracursos.com/blog/que-es-gitops>)
- **Artículo** Tipos de pruebas: ¿cuáles son las principales y por qué utilizarlas? (<https://www.aluracursos.com/blog/tipos-de-pruebas-cual-utilizar>)

SQL - Fundamentos:

- Conocer los comandos más comunes de SQL
- Usar SELECT para consultar una tabla
- Usar INSERT para insertar datos en una tabla
- Usar UPDATE para actualizar una tabla
- Usar DELETE para eliminar datos de una tabla
- Usar JOIN para conectar los datos de múltiples tablas
- Conocer las cláusulas (FROM, ORDER BY, etc.)

Contenidos

- **Artículo** Amazon: ¿Qué es SQL? (https://aws.amazon.com/es/what-is/sql/?nc1=h_ls)

Contenidos Alura:

- **Artículo** MySQL: desde la descarga e instalación hasta su primera tabla (<https://www.aluracursos.com/blog/mysql-desde-la-descarga-e-instalacion-hasta-su-primera-tabla>)
- **Artículo** Bases de datos relacionales (<https://www.aluracursos.com/blog/base-de-datos-relacional>)
- **Artículo** ¿Qué es SQL? (<https://www.aluracursos.com/blog/que-es-sql>)
- **Artículo** Normalización en base de datos - Estructura (<https://www.aluracursos.com/blog/normalizacion-en-base-de-datos>)
- **Artículo** En SQL, null es null, vacío está vacío (<https://www.aluracursos.com/blog/en-sql-null-es-null-vacio-es-vacio>)
- **Artículo** SELECT, INSERT, UPDATE y DELETE en SQL: aprende a utilizar cada uno (<https://www.aluracursos.com/blog/select-insert-update-delete-sql>)
- **Artículo** Funciones de agregación con GROUP BY en SQL, ¿cómo utilizarlas? (<https://www.aluracursos.com/blog/funciones-de-agregacion-con-group-by-en-sql-como-utilizarlas>)
- **Artículo** SQL JOIN: Aprenda INNER, LEFT, RIGHT, FULL e CROSS (<https://www.aluracursos.com/blog/sql-join-aprenda-inner-left-right-full-e-cross>)
- **Artículo** select count(*), count(1) y count(nombre): batalla de los counts de SQL (<https://www.aluracursos.com/blog/select-count-count1-e-countnome-la-batalla-de-los-counts-de-sql>)
- **YouTube** Descomplicando Base de Datos | #Aluramás (https://www.youtube.com/watch?v=G1cDRqKuxpg&t=6s&ab_channel=AluraLatam)
- **YouTube** ¿Qué es SQL y NoSQL? (<https://www.youtube.com/watch?v=cLLKVd5CNLC>)
- **YouTube** Banco de Datos MySQL (<https://www.youtube.com/watch?v=8J0AoPZMVxA>)
- **Curso** SQL con MySQL (<https://app.aluracursos.com/formacion-sql-con-mysql>)
- **Curso** Curso Introducción a SQL con MySQL: Manipule y consulte datos (<https://app.aluracursos.com/course/introduccion-sql-mysql-manipule-consulte-datos>)
- **Curso** SQL Server: consultas avanzadas con Microsoft SQL Server 2019 (<https://app.aluracursos.com/course/sql-server-consultas-microsoft-sql-server-2019>)
- **Curso** Formación Modelado de datos (<https://www.aluracursos.com/formacion-modelado-de-datos>)
- **Curso** Formación SQL con Microsoft SQL Server (<https://www.aluracursos.com/formacion-SQL-con-Microsoft-SQL-Server-2019>)

SOLID:

- Solid tiene cinco principios considerados como buenas prácticas en el desarrollo de software que ayudan a los programadores a escribir los códigos más limpios, dividiendo las responsabilidades, disminuyendo los acoplamientos, facilitando la refactorización y estimulando el reaprovechamiento del código. Propuesto por Robert C. Martin, SOLID propicia el desarrollo de un código limpio, legible y comprobable.

Contenidos

- **Web** Los principios SOLID de programación orientada a objetos (<https://www.freecodecamp.org/espanol/news/los-principios-solid-explicados-en-espanol/>)
- **Web** Principios SOLID: Qué son, cuáles, y qué beneficios aporta usarlos (<https://devexperto.com/principios-solid/>)
- **Podcast** SOLID - los androides (<https://open.spotify.com/episode/47eElzhTPr1RluVcxeVal4?si=4df70e899c034283>)
- **YouTube** Los principios SOLID, ¡explicados! (<https://www.youtube.com/watch?v=2X50sKeBAcQ>)

Design Patterns:

- En ingeniería de software, un "patrón de diseño" (Design Pattern en inglés) es una solución general y reutilizable para un problema que ocurre normalmente dentro de un determinado contexto de diseño de software.
- Conocer y aplicar los principales patrones de diseño.

Contenidos

- **Web** Patrones de Diseño (<https://refactoring.guru/es/design-patterns>)
- **Web** Designer Patterns (<https://github.com/FernandoCalmet/design-patterns>)
- **Web** ¿Qué son los patrones de diseño de software? (<https://profile.es/blog/patrones-de-diseno-de-software/>)
- **Web** Design Patterns: conoce los diferentes tipos que existen y sus beneficios (<https://www.hostgator.mx/blog/design-patterns-que-debes-saber/>)
- **Podcast** Patrones Diseño, ventajas y desventajas (<https://open.spotify.com/episode/3VjQHnPVusU6zz5PyIMVFu?si=a613cb4c157e4055>)
- **Podcast** Patrones Diseño (<https://open.spotify.com/episode/6QO1HYdAgzrMGLVpz2kn0C?si=c95296f387c4490a>)
- **YouTube** ♦ Patrones de diseño software: Repaso completo en 10 minutos (<https://www.youtube.com/watch?v=6BHOeDL8vIs>)

Contenidos Alura:

- **Artículo** Design Patterns: Breve introducción a los patrones de diseño (<https://www.aluracursos.com/blog/design-patterns-introduccion-a-los-patrones-de-diseno>)

Clean Architecture:

- Clean Architecture (Arquitectura Limpia) es una forma de desarrollar software, de tal forma que solo mirando el código fuente de un programa, debes ser capaz de decir lo que el programa hace.

Contenidos

- **Web** ¿Qué es Clean Architecture? (<https://clean-architecture-python.readthedocs.io/en/latest/introduccion/index.html>)

- **Artículo** ¿Por qué utilizo Clean Architecture? (<https://xurxodev.com/por-que-utilizo-clean-architecture-en-mis-proyectos/>)
- **YouTube** Revisando Clean code, vale la pena leerlo? | review clean code (<https://www.youtube.com/watch?v=uQfm6YaJTJI>)
- **YouTube** Hexagonal architecture, qué es y qué diferencias tiene contra Clean Architecture? - PT 1 (<https://www.youtube.com/watch?v=NOWU4K6piwo>)
- **YouTube** Desarrollo ágil con Arquitectura limpia Hexa3 (<https://medium.com/@dariopalminio/desarrollo-%C3%A1gil-de-ecosistemas-de-aplicaciones-hexagonales-3-capas-hexa3l-d6370bf11db0>)
- **Podcast** DevTalles - 125: Arquitectura Limpia (<https://open.spotify.com/episode/3ftIJfucj7Jx4ZO2cDSghx?si=e6b2c5607ead43b5>)

Contenidos Alura:

- **Artículo** Design Patterns: Breve introducción a los patrones de diseño (<https://www.aluracursos.com/blog/design-patterns-introduccion-a-los-patrones-de-diseno>)

Línea de Comando - Fundamentos:

- CLI es un programa de línea de comandos que acepta la entrada de texto para ejecutar funciones del sistema operativo.
- Conocer los comandos más importantes

Contenidos

- **Web** W3Schools: What is Command Line Interface (CLI)? (https://www.w3schools.com/whatis/whatis_cli.asp)
- **Web** Uso de argumentos de la línea de comandos para Terminal Windows (<https://learn.microsoft.com/es-es/windows/terminal/command-line-arguments?tabs=windows>)
- **Artículo** Interfaz de línea de comandos o CLI (<https://www.computerweekly.com/es/definicion/Interfaz-de-linea-de-comandos-o-CLI>)
- **Artículo** El Manual de Comandos de Linux (<https://www.freecodecamp.org/espanol/news/comandos-de-linux/>)
- **YouTube** Consola vs Terminal vs Shell vs CLI 🖥️ ¿Qué es la terminal? (https://www.youtube.com/watch?v=1YxHXBsVNGQ&ab_channel=ProgramadorX)
- **YouTube** Aprende la línea de comandos en un mac - bash scripting (https://www.youtube.com/watch?v=vfwA3pUnVOg&ab_channel=Datademia)
- **YouTube** freeCodeCamp.org: Command Line Crash Course (<https://www.youtube.com/watch?v=yz7nYlnXLfE>)
- **YouTube** Traversy Media: Command Line Crash Course For Beginners - Terminal Commands (<https://www.youtube.com/watch?v=uwAqEzhyjtw>)
- **YouTube** Comandos Básicos e Intermedios CMD (<https://youtu.be/erKosEQaaFc>)
- **YouTube** Terminal MAC tutorial en Español - Cómo usar la terminal en MAC (<https://www.youtube.com/watch?v=TmP3y7Z2kk4>)

Contenidos Alura:

- **Artículo** CMD: Sugerencias para trabajar en el prompt de Windows (<https://www.aluracursos.com/blog/consejos-para-trabajar-en-el-indicador-de-windows>)
- **Artículo** Como usar el terminal integrado de Visual Studio Code (<https://www.aluracursos.com/blog/como-usar-el-terminal-integrado-de-visual-studio-code>)

- **Curso** Linux 1: conociendo y utilizando la terminal (<https://app.aluracursos.com/course/linux-1-conociendo-utilizando-terminal>)

Cloud - Fundamentos:

- La computación en nube, o cloud computing, es la distribución de servicios informáticos a través de Internet mediante un modelo de tarificación de pago por uso. Una nube se compone de varios recursos informatizados, desde los propios ordenadores (o instancias, en terminología de nube) hasta las redes, el almacenamiento, las bases de datos y todo lo que les rodea. En otras palabras, todo lo que normalmente se necesita para montar el equivalente a una sala de servidores, o incluso un centro de datos completo, estará listo para usar, configurar y ejecutar.
- Conocer la diferencia entre IaaS, PaaS y SaaS
- Conocer los mayores proveedores de nube
- Especializarse en un proveedor específico de su preferencia

Contenidos

- **Web** ¿Qué es la informática en la nube? | Microsoft Azure (<https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-cloud-computing/>)
- **Web** Amazon AWS: ¿Qué es la computación en nube? (<https://aws.amazon.com/en/what-is-cloud-computing/>)
- **Web** Tipos de computación en la nube (<https://aws.amazon.com/es/types-of-cloud-computing/>)
- **Web** ¿Cómo funciona Azure? (<https://learn.microsoft.com/es-es/azure/cloud-adoption-framework/get-started/what-is-azure>)
- **Web** ¿Qué es el almacenamiento en la nube? (<https://aws.amazon.com/es/what-is/cloud-storage/>)
- **Web** ¿Qué es la seguridad en la nube? (<https://cloud.google.com/learn/what-is-cloud-security?hl=es-419>)
- **Web** Arquitectura sin servidor (<https://learn.microsoft.com/es-es/dotnet/architecture/serverless/serverless-architecture>)
- **Web** ¿Qué es Docker? (<https://aws.amazon.com/es/docker/>)
- **Artículo** Guía para principiantes sobre los fundamentos de la computación en nube (<https://scientya.com/a-beginners-guide-to-the-basics-of-what-cloud-computing-is-about-e8b3b7f25a30/>)
- **Artículo** Cloud Computing para principiantes (<https://medium.com/hackernoon/cloud-computing-for-beginners-85d168959afb/>)
- **Artículo** ¿Qué es Google Cloud y para qué sirve? (<https://www.incentro.com/es-ES/blog/que-es-google-cloud-platform>)
- **YouTube** ¿Qué es computación en la nube? | ¿Qué es cloud computing? | Explicado en 4 minutos (<https://youtu.be/MCKdahh2ISo>)
- **YouTube** 🚀 CLOUD COMPUTING ¿Qué es IaaS, PaaS y SaaS? | Modelos de Servicio Cloud (<https://youtu.be/VR8aXePkQ5M>)
- **YouTube** ¿Qué es AWS? (<https://www.youtube.com/watch?v=x2vrg7HuM6g>)
- **YouTube** ¿Cómo empiezo con Google Cloud? (Hablemos en Cloud) (<https://www.youtube.com/watch?v=OiDWqu0oQfo>)
- **YouTube** Introducción a la infraestructura de Google Cloud (<https://www.youtube.com/watch?v=209DGQCism4>)
- **YouTube** ¿Qué es la Computación en la Nube? | AWS desde cero - Parte 1: Introducción (<https://www.youtube.com/watch?v=IciVhWQ8npw>)

Contenidos Alura:

- **Artículo** ¿Qué es Cloud y sus principales servicios? (<https://www.aluracursos.com/blog/que-es-cloud-y-sus-principales-servicios>)
- **Artículo** Conociendo Terraform (<https://www.aluracursos.com/blog/conociendo-terraform>)
- **Artículo** Empezando con Docker (<https://www.aluracursos.com/blog/empezando-con-docker>)
- **Artículo** Heroku, Vercel y otras opciones de cloud como plataforma (<https://www.aluracursos.com/blog/heroku-vercel-y-otras-opciones-de-cloud-como-plataforma>)
- **YouTube** Fundamentos del OCI | Contenidos ONE (<https://youtu.be/rEgSc0UqX-g>)
- **Curso** Curso Oracle Cloud Infrastructure: implementación de una aplicación en la nube (<https://app.aluracursos.com/course/oracle-cloud-infrastructure-implementacion-aplicacion-nube>)
- **Curso** Curso Oracle Cloud Infrastructure: base de datos e infraestructura como código (<https://app.aluracursos.com/course/oracle-cloud-infrastructure-base-datos-infraestructura-codigo>)
- **Curso** Curso Deploy en Amazon EC2: Alta disponibilidad y escalabilidad de una aplicación (<https://app.aluracursos.com/course/deploy-amazon-ec2-alta-disponibilidad-escalabilidad>)
- **Curso** Curso Amazon Lightsail: Simplificando la nube (<https://app.aluracursos.com/course/amazon-lightsail-simplificando-nube>)

Diseño Orientado a Dominio - DDD:

- El Diseño Orientado a Dominio (DDD) es un enfoque de diseño y desarrollo de software que se informa principalmente por los requisitos de negocio. Los componentes del programa (objetos, clases, matrices, etc.) indican la industria, sector o dominio empresarial en que opera el negocio.
- Modelar dominios de manera efectiva.
- Basar proyectos complejos en modelos de dominio.
- Conocer los bloques de construcción de DDD.

Contenidos

- **Web** Domain Driven Design: principios, beneficios y elementos — Primera Parte (<https://medium.com/@jonathanloscalzo/domain-driven-design-principios-beneficios-y-elementos-primera-parte-aad90f30aa35>)
- **Web** Domain Driven Design: principios, beneficios y elementos — Segunda Parte (<https://medium.com/@jonathanloscalzo/domain-driven-design-principios-beneficios-y-elementos-segunda-parte-337d77dc8566>)
- **Web** Introducción a Domain Drive Design - DDD (<https://refactorizando.com/introduccion-domain-drive-design/>)
- **Podcast** TDD y DDD, fortalezas y cuando usarlo (<https://open.spotify.com/episode/7iioBOv7ySQZttPyw2GshP?si=a114f642ecc94682>)
- **YouTube** Aprendiendo el Dominio (DDD en Español) (<https://www.youtube.com/watch?v=BilfTrk954A>)
- **YouTube** Domain Driven Design - Píldoras de conocimiento - Autentia (<https://www.youtube.com/watch?v=vccTIFGSiHA>)

Habilidad Auxiliar: UX y Design

Figma - Fundamentos:

- Figma es una aplicación web colaborativa para el diseño de interfaces. El conjunto de funciones de Figma se centra en la interfaz de usuario y el diseño de la experiencia del usuario, con énfasis en la colaboración en tiempo real, utilizando una variedad de herramientas de creación de prototipos y editor de gráficos vectoriales.
- Crear diseños de página y componentes

Contenidos

- **Artículo** Figma Basics- Primeros pasos en Figma (<https://www.figma.com/community/file/923140611594993345>)
- **Artículo** Introducing Figma to React (<https://medium.com/figma-design/introducing-figma-to-react-d2d545cba3cc>)
- **YouTube** Figma para principiantes (<https://www.youtube.com/watch?v=GoNzQHc7-qq>)

Contenidos Alura:

- **YouTube** ¿Cómo un desarrollador Front End utiliza el Figma? (https://www.youtube.com/watch?v=UuAX5azcvDQ&ab_channel=AluraLatam)

Diseño Responsivo:

- Ajustar tus páginas al tamaño de pantalla del usuario
- Media queries
- Conocer el concepto de Mobile first

Contenidos

- **Artículo** ¿Qué es el diseño responsivo? (<https://www.ionos.es/digitalguide/paginas-web/diseño-web/que-es-el-diseño-responsivo/#:~:text=El%20dise%C3%B1o%20responsivo%2C%20tambi%C3%A9n%20dise%C3%B1o,genera>)
- **Artículo** Qué es el diseño responsivo y por qué tu web lo necesita? (<https://trazada.com/que-es-el-diseño-responsivo/>)
- **Artículo** Mobile First vs Diseño Responsivo, diferencias y ventajas de cada uno (<https://www.marketerosagencia.com/blog/diseño-web/mobile-first-vs-diseño-responsivo/>)
- **Artículo** Diseño web responsive vs. adaptable ¿Cuál elegir? (<https://es.wix.com/blog/2022/07/diseño-web-responsive-y-adaptable>)
- **YouTube** Qué es el Responsive Web Design - Bien explicado (<https://youtu.be/2zypRIzIcHc>)
- **YouTube** Breakpoints Responsive | Qué es responsive design | Eduardo Fierro Pro (https://youtu.be/_ESkNVyXjdY)

Contenidos Alura:

- **Artículo** Puntos de ruptura en el diseño responsivo (<https://www.aluracursos.com/blog/puntos-de-ruptura-en-el-diseño-responsivo>)
- **Curso** Layouts Responsivos: Trabajando con layouts mobile (<https://app.aluracursos.com/course/layouts-responsivos-layouts-mobile>)

Sistemas de Diseño:

- Un sistema de diseño es una colección de componentes reutilizables, guiados por estándares claros, que se pueden ensamblar para crear aplicaciones.

- Creación y mantenimiento de bibliotecas que se consumirán y utilizarán como estándar para construir un proyecto.

Contenidos

- **Artículo** Por qué necesitas un sistema de diseño y cómo tus clientes te lo van a agradecer (<https://www.plainconcepts.com/es/sistema-diseno-guia/#:~:text=Un%20sistema%20de%20dise%C3%B1o%20es%20un%20conjunto%20de%20reglas%20y,tra>)
- **Artículo** Guía paso a paso para crear un sistema de diseño (<https://www.ranktracker.com/es/blog/a-step-by-step-guide-to-creating-a-design-system/>)
- **Artículo** A comprehensive guide to design systems (<https://www.invisionapp.com/inside-design/guide-to-design-systems/>)
- **Artículo** Design Tokens en tu Design System (<https://gonzalo-vasquez-correa.medium.com/design-tokens-en-tu-design-system-90498eeafd49>)
- **YouTube** Sistemas de Diseño (Team Library) (https://www.youtube.com/watch?v=WS67rHFTVzc&ab_channel=jonmircha)
- **YouTube** Crear un design System en Figma (https://www.youtube.com/watch?v=hrr_-3Y8j7M&ab_channel=CreatividadenBlanco)

Sistemas de color:

- Definición de una paleta de colores que tenga sentido para una interfaz dada

Contenidos


- **Artículo** Teoría del color. Guía definitiva para comprenderla (<https://graffica.info/teoria-del-color-guia-definitiva/>)
- **Artículo** La ciencia de los colores (<https://medium.com/100-days-of-product-design/the-science-of-color-bd0f057c08ea>)
- **Artículo** RGB frente a CMYK: una guía de sistemas de color para diseñadores (<https://medium.com/envato/rgb-vs-cmyk-a-guide-to-color-systems-for-designers-6be8c1ed8554>)
- **Artículo** Color en el diseño de interfaz de usuario: un marco (práctico) (<https://medium.com/@erikdkennedy/color-in-ui-design-a-practical-framework-e18cacd97f9e>)
- **Artículo** Guía básica de colores de la interfaz de usuario (<https://blog.prototypr.io/basic-ui-color-guide-7612075cc71a>)
- **YouTube** Teoría del Color | Diseño Web (<https://www.youtube.com/watch?v=g0JTvvalf8s>)
- **YouTube** ¿Qué es el color? Explicación de la Teoría del color (<https://www.youtube.com/watch?v=CFn-wPKxRR4>)

Cómo usar fuentes:


- Elegir la fuente más adecuada para un proyecto determinado

Contenidos

- **Web** Fuentes Web (https://developer.mozilla.org/es/docs/Learn/CSS/Styling_text/Web_fonts)
- **Artículo** Cómo utilizar cualquier fuente/tipografía en nuestra página web (<https://www.imaginanet.com/blog/como-utilizar-cualquier-fuente-tipografia-en-nuestra-pagina-web.html>)
- **YouTube** ¿Cómo Cambiar la Fuente a mi Página Web? (<https://www.youtube.com/watch?v=JQ6b-7geFc4>)

-  Descarga y utiliza Google Fonts de forma local (<https://www.youtube.com/watch?v=cTaFBV1Z0dw>)

Contenidos Alura:

-  6 malas prácticas que perjudican usuarios disléxicos
(<https://www.aluracursos.com/blog/6-malas-practicas-que-perjudican-usuarios-dislexicos>)

TechGuide - Alura
Alura, PM3 e FIAP
O Techguide.sh é um projeto open source