**Documento de requerimientos del proyecto “ByteMart”**

***ByteMart***

***Fecha: 24/10/2024***

**Tabla de contenido**

[**Historial de Versiones 2**](#_heading=h.b6ouobjkmfoj)

[**Información del Proyecto 2**](#_heading=h.30j0zll)

[**Aprobaciones 3**](#_heading=h.1fob9te)

[**● Propósito 3**](#_heading=h.7rgk9vn94zub)

[**● Alcance del producto / Software 3**](#_heading=h.h1dg34aru6yt)

[**● Referencias 4**](#_heading=h.py59n0iu08k9)

[**● Funciones del producto 4**](#_heading=h.l7ab9b1sbyp6)

[**● Características de usuarios 4**](#_heading=h.y8tcjt3ex5q9)

[**Requerimientos Funcionales 5**](#_heading=h.ly7g3cje2djg)

[**● Reglas de negocio 8**](#_heading=h.3iuranur4jg1)

[**● Requerimientos de interfaces externas 8**](#_heading=h.o9kllbropsda)

[**9.1. Interfaces de usuario 8**](#_heading=h.1ksv4uv)

[**9.2. Interfaces de hardware 9**](#_heading=h.h4qxka5m0wm5)

[**9.3. Interfaces de software 10**](#_heading=h.ypoyjn9alfb5)

[**9.4. Interfaces de comunicación 11**](#_heading=h.h8tm5k32udh7)

[**● Requerimientos no funcionales 11**](#_heading=h.fcgbcx28emf6)

[**● Otros requerimientos 15**](#_heading=h.wxdnydkf6b29)

[**● Glosario 15**](#_heading=h.xtj4exyo2m0n)

# Historial de Versiones

| **Fecha** | **Versión** | **Autor** | **Organización** | **Descripción** |
| --- | --- | --- | --- | --- |
| 24/10/2024 | 1.0 | Cristian Ojeda |  | Desarrollo de puntos. |
|  |  |  |  |  |

# Información del Proyecto

| Empresa / Organización | - |
| --- | --- |
| Proyecto | ByteMart |
| Fecha de preparación | 2024 |
| Cliente | Duoc UC |
| Patrocinador principal | Duoc UC |
| Líder de Proyecto | Vicente Laguna |
| Gerente | Julio Tapia |

# Aprobaciones

| **Nombre y Apellido** | **Cargo** | **Departamento u Organización** | **Fecha** | **Firma** |
| --- | --- | --- | --- | --- |
| Vicente Laguna | Lider de Proyecto | Equipo desarrollo | 24-10-2024 |  |
| Julio Tapia | Gerente | Gerencia | 24-10-2024 |  |

# Propósito

El presente documento, tiene como propósito detallar los requerimientos generales y específicos, necesarios para el desarrollo del proyecto Byte Mart.

# Alcance del producto / Software

El sistema desarrollado se trata de un e-commerce de venta de productos tecnológicos, por lo cual, el sistema está diseñado principalmente para automatizar procesos mediante web scraping, además de realizar extracción e ingesta de datos para permitir la comparativa de precios con las competencias de mercado, además integrar distintas APIS de sistemas de pagos, productos recomendados, consulta de categorías y notificación de stock bajo para el manejo de datos, análisis y publicación de estos. También, el sistema estará diseñado para el usuario, con una interfaz intuitiva y amigable que permite una gran experiencia de usuario.

Objetivos:

* Desarrollar una plataforma e-commerce totalmente funcional y segura.
* Poder integrar sistemas externos APIS.
* Lograr implementar análisis de datos que permita monitorear el comportamiento de ventas y otras métricas claves del negocio.
* Implementar tecnologías novedosas, que permitan que la plataforma sea llamativa e innovadora.
* Lograr que el sistema cumpla con la automatización y optimización de procesos relevantes para el manejo de información del negocio.

# Referencias

* Acta de Constitución.
* Documento ERS.
* Matriz de Requerimientos.

# Funciones del producto

* Una de las principales funciones del proyecto Byte Mart es la automatización para realizar un web scraping, se realiza una extracción e ingesta de datos con el objetivo de estudiar la competencia y analizar el estado del mercado, esto optimiza la toma de decisiones. Además de integrar API para el sistema de pago y desarrollar APIs con distintos propósitos, para la gestión de inventario, productos recomendados, notificación de stock bajo y consultar categorías. Nuestra plataforma web ofrece una interfaz amigable para el usuario, que permita la visualización de productos y realizar compras. Por otro lado, la plataforma de escritorio permite asegurar continuidad del trabajo sin necesidad de tener conectividad a internet. Finalmente, permite gestionar información empresarial de forma efectiva, estructurando los datos para facilitar el análisis.

# Características de usuarios

* Clientes: Usuarios que necesitan adquirir o comparar productos tecnológicos.
* Administradores: Usuarios mantenedores del sistema, con credenciales necesarias para agregar/eliminar/modificar productos, usuarios, etc.
* **Entorno operativo**

Para alojar este sistema está diseñado para operar de forma local.

**Sistema operativo:** Windows.

**Web:** Python Flask.

**App de Escritorio:** Java//Netbeans.

**Base de datos:** MySQL Workbench.

**Entorno de desarrollo:** Visual Studio Code.

**Navegador web:** Google Chrome y Microsoft Edge.

# Requerimientos Funcionales

* **Definición de requerimientos específicos:**
* 3.2.1 Login de usuario: El sistema permite a los usuarios iniciar sesión mediante credenciales válidas.
* 3.2.2 Registro de usuarios: El sistema permite crear nuevas cuentas a usuarios con su información.
* 3.2.3 Eliminar usuario: El administrador puede eliminar usuarios del sistema
* 3.2.4 Editar usuario: El administrador y los usuarios pueden modificar y visualizar la información de sus cuentas.
* 3.2.5 Recuperación de contraseña: El sistema permite a los usuarios restablecer la contraseña de sus cuentas mediante un formulario de recuperación.
* **Requerimientos para la gestión de inventario y carrito de compras:**
* 3.2.6 Agregar productos al carrito de compras: Los usuarios y clientes pueden añadir productos al carrito de compras.
* 3.2.7 Eliminación de productos del carrito de compras: Los usuarios y clientes pueden eliminar productos agregados al carrito de compras.
* 3.2.8 Añadir elementos al inventario: El administrador, los jefes de tienda y los vendedores pueden añadir elementos al inventario.
* 3.2.9 Agregar productos al inventario: Los usuario autorizados pueden ingresar productos dentro del inventario.
* 3.2.10 Editar productos en el inventario: Los usuarios autorizados pueden editar los productos que se encuentran en el inventario.
* 3.2.11 Eliminar productos del inventario: Los usuarios autorizados podrán editar los productos que se encuentran en el inventario.
* 3.2.12 Eliminación de elementos añadidos al inventario: El administrador, los jefes de tienda y los vendedores pueden eliminar elementos del inventario.
* 3.2.13 Filtrado de elementos del inventario: El administrador, los jefes de tienda y los vendedores pueden filtrar los elementos del inventario.
* 3.2.14 Visualización de elementos del inventario: El administrador, los jefes de tienda y los vendedores pueden visualizar los elementos del inventario.
* 3.2.15 Visualización de stock de los artículos: Los usuarios podrán visualizar los stock de los artículos que se encuentran en el inventario.
* 3.2.16 Entregar precio de productos: Los usuarios podrán visualizar los stock de los artículos que se encuentran en el inventario.
* **Requerimientos para la gestión de categorías en el portal web:**
* 3.2.17 Agregar categorías de productos: Los usuarios autorizados pueden agregar categorías al sistema.
* 3.2.18 Editar categorías de productos: Los usuarios autorizados pueden editar categorías en el sistema.
* 3.2.19 Eliminar categoría de productos: Los usuarios autorizados , pueden eliminar categorías en el sistema.
* 3.2.20 Visualización de categorías por productos: Los usuarios pueden visualizar las categorías que se encuentran en el inventario.
* 3.2.21 Buscador de productos por categoría: Los usuarios pueden buscar productos según su categoría por palabras claves.
* **Requerimientos para la gestión de facturas y ordenes de compras:**
* 3.2.22 Los usuarios autorizados pueden eliminar órdenes de compra registradas en el sistema
* 3.2.23 Visualizar facturas de venta: En el sistema permite a los usuarios acceder y visualizar las facturas de ventas correspondientes.
* **Requerimientos para la gestión de usuarios en el portal de escritorio:**
* 3.2.24 Login de usuario: El sistema permite a los usuarios iniciar sesión mediante credenciales válidas.
* 3.2.25 Registro de usuarios: El sistema permite crear nuevas cuentas a usuarios con su información.
* 3.2.26 Editar usuario: El administrador y los usuarios pueden modificar y visualizar la información de sus cuentas.
* 3.2.27 Eliminar usuario: El administrador puede eliminar usuarios del sistema.
* **Requerimientos para la gestión de inventario en el portal de escritorio:**
* 3.2.28 Agregar productos al inventario: Los usuario autorizados pueden ingresar productos dentro del inventario.
* 3.2.29 Editar productos en el inventario: Los usuarios autorizados pueden editar los productos que se encuentren en el inventario.
* 3.2.30 Eliminar productos del inventario: Los usuarios autorizados pueden editar los productos que se encuentren en el inventario.
* 3.2.31 Visualización de artículos: Usuarios pueden visualizar los artículos que se encuentran en el inventario.
* 3.2.32 Visualización de stock de productos: Usuarios pueden visualizar el stock de los productos registrados en el inventario.
* 3.2.33 Buscador de productos: Los usuarios pueden buscar productos por palabras claves.
* **Requerimientos para la gestión de categorías en el portal de escritorio:**
* 3.2.34 Agregar categorías de productos: Los usuarios autorizados pueden agregar categorías al sistema.
* 3.2.35 Editar categorías de productos: Los usuarios autorizados pueden editar categorías en el sistema.
* 3.2.36 Eliminar categoría de productos: Los usuarios autorizados , pueden eliminar categorías en el sistema.
* 3.2.37 Buscador de productos por categoría: Los usuarios pueden buscar productos según su categoría por palabras claves.
* **Requerimientos para la gestión de facturas y ordenes de compras:**
* 3.2.38 Eliminar órdenes de compra: Los usuarios autorizados pueden eliminar órdenes de compra registradas en el sistema
* 3.2.39 Visualizar facturas de venta: En el sistema permite a los usuarios acceder y visualizar las facturas de ventas correspondientes.

# Reglas de negocio

* **Funcionalidad:** El sistema, debe cumplir con las normativas, disponibilidad, usabilidad y seguridad, permitiendo una correcta funcionalidad de las APIs de pago , Notificación de stock bajo, consulta de categorías y productos recomendados..
* **Mantenimiento Programado**: Mantenimientos programados en horarios de baja demanda para las actualizaciones y mejoras sin afectar la experiencia.
* **Contenido interfaz:** El contenido mostrado en la página debe ser claro y preciso para los usuarios.

# Requerimientos de interfaces externas

# Interfaces de usuario

Para el sistema que se ha compuesto, se realiza una interfaz gráfica la cual permite realizar distintas interacciones de usuarios finales y a los administradores. Esta misma es intuitiva, atractiva visualmente y que todas sus funcionalidades cuentan con un fácil acceso.

Como entradas se consideran las siguientes:

1. Formulario de registro de usuarios: Los usuarios completan distintos campos como correo electrónico, nombre de usuario, contraseña y dirección.
2. Formulario de Login: Los usuarios ingresan mediante los campos de nombre de usuario y contraseña.
3. Formulario para la recuperación de la contraseña: El usuario registrado ingresa el correo electrónico, para hacer el envío de código de recuperación.
4. Búsqueda de productos: El sistema debe operar con un campo de texto el cual permite ingresar distintos términos de búsqueda.
5. Formulario para agregar productos al inventario: Los usuarios completan los campos requeridos incluyendo fotografías o imágenes de los productos.
6. Eliminar productos del inventario: Los usuario autorizados pueden seleccionar los productos a eliminar del inventario.
7. Formulario para ingresar categorías de productos: Los usuarios autorizados ingresan nombre de la categoría a ingresar.
8. Eliminar órdenes de compra: Los usuarios autorizados seleccionan la orden de compra a eliminar.

Como salidas se consideran las siguientes:

1. Confirmación de registro de usuarios: El sistema valida los datos entregados por el usuarios, genera un mensaje si la cuenta se ha registrado exitosamente
2. Respuesta del login: El sistema al validar los datos entregados por el usuario muestra un mensaje cuando las credenciales son correctas o incorrectas.
3. Respuesta de recuperación de contraseña: El sistema solicita al usuario que ingrese el código que se envía por correo electrónico, para solicitar que el usuario ingrese una contraseña nueva.
4. Resultado búsqueda de productos: El sistema muestra mediante un listado, estos resultados coinciden con las palabras claves que se ingresan en el campo de búsqueda.
5. Resultados del ingreso de producto: El sistema valida los datos e imagen ingresada por el usuario.
6. Confirmación para eliminar el producto: El sistema muestra un mensaje confirmando que el producto fue eliminado.
7. Resultado del ingreso de categoría de productos: El sistema muestra un mensaje cuando la categoría se registró exitosamente o un mensaje de error si ya existe la categoría en la base de datos.
8. Confirmación para eliminar orden de compra: El sistema debe mostrar un mensaje de que la orden de compra se eliminó correctamente.

# Interfaces de hardware

Se señalan los requisitos de las características lógicas de las diferentes interfaces del producto y cada componente asociado, para entregar la trazabilidad entre los distintos sistemas.

El sistema web utiliza la tecnología de flask levantada en python, para la cual se utiliza un entorno local para poder realizar conexión con el aplicativo, con la tecnología de flask podremos realizar la creación directa de clases para interactuar con la base de datos, como también manejar las distintas peticiones que pueden existir en el portal, ya sean formularios de creación de distintos tipos de datos, como pueden ser usuarios, productos, categorías, entre otros.

Para el manejo y almacenamiento de los datos generados, se utiliza el motor de base de datos de MySQL, dónde hacemos uso de distintas herramientas como secuencias para la autocreación de ID en algunas clases, creación de tablas y la conexión entre ellas para entrelazar el modelo de negocio dentro de sus procesos.

Para el sistema de escritorio, al igual que el web, se hace uso de un entorno local, levantado en netbeans y construido en lenguaje java, en el cual se crean distintas clases para la interacción con la base de datos que en este caso compartirá con el sistema web en MySQL, este sistema contará con las funcionalidades más importantes para la administración del modelo de negocio.

Para el resguardo y seguridad de los datos se hará uso de una base de datos espejo la cual periódicamente se va cargando con la información generada por el portal, el proceso de carga de datos de respaldo será por lotes después de un tiempo determinado por el cliente, de esta manera minimizamos los riesgos ante incidentes perjudiciales para la organización y/o el cliente.

Las interacciones con las distintas Apis, como las de pago, gestión de inventario, productos recomendados y consultar categorías, se usa el formato JSON para el intercambio de información,de esta manera administramos efectivamente el detalle de cada concepto relevante para el cliente.

# 

# Interfaces de software

* Utilizamos flask, una herramienta basada en python para el aplicativo web en conjunto con el motor de base de datos MySQL para la gestión de la data generada. Por otro lado, utilizamos Java para el desarrollo del aplicativo de escritorio, el cual cuenta con el mismo motor de base de datos.
* Para el manejo del contenido y el formato, utilizamos el formato json para el intercambio de datos entre los sistemas y las APIS, también, el uso de consultas básicas en el portal ecommerce para la creación, eliminación, actualización y listado de datos. Finalmente, el sistema de escritorio a través del driver jdbc realizará la conexión a la base de datos.

# Interfaces de comunicación

* El sistema diseñado para Byte Mart se comunica e interactúa a través de diversos componentes, tanto web como escritorio, utilizando un entorno local para una gestión de inventarios, productos, simulaciones de pago, e intercambio de datos compartidos entre los sistemas.
* El protocolo definido para la comunicación, en este caso es local, HTTP para el aplicativo web en el localhost 8080, para el cliente del sistema de escritorio, este funciona directamente con la gestión y administración de data con una base de datos.

# Requerimientos no funcionales

**Rendimiento:**

El tiempo de respuesta que tiene el sistema al momento de realizar las transacciones es de un 85% de las transacciones en menos de 5 segundos.

El sistema de Byte Mart soporta un máximo de 200 usuarios conectados simultáneamente, también se considera que el tiempo de respuesta de las diversas operaciones no excede los 5 segundos.

**Requerimientos para desarrollo de automatización:**

3.2.40 Abrir navegador: El bot debe abrir un navegador.

3.2.41 Conectarse a la base de datos: El bot debe realizar la conexión con la base de datos.

3.2.42 Leer JSON: El bot debe leer un archivo JSON de configuración.

3.2.43 Convertir Texto: El bot debe ser capaz de estandarizar el texto extraído.

3.2.44 Extraer información: El bot debe ser capaz de extraer nombre, categoría, marca, stock y precio.

3.2.45 Insertar datos en la tabla: El bot debe ser capaz de tomar la información extraída de la web, para insertarla en la base de datos.

3.2.46 Cerrar conexión con la base de datos: El bot debe ser capaz de cerrar conexión con la base de datos

3.2.47 Cerrar navegador: El bot debe ser capaz de cerrar el navegador.

**Requerimientos para la integración de apis en el portal web:**

3.2.48 Integración de API de pago: La plataforma web está conectada con la API de pago desarrollada, permitiendo que el proceso de pago sea efectivo, el tiempo de espera para validar el proceso de pago es de al menos 5 segundos para el 85% de las transacciones.

3.2.49 Integración de API DPA: La plataforma web está conectada con la API para Distribución Política Administrativa, donde se visualizan las regiones, provincias y comunas del país.

3.2.50 Integración de API para consulta de categorías: La plataforma web está conectada con una API para la consulta de categorías, esta permite buscar productos según la categoría a la cual pertenecen, esto permite que el usuario tenga una mejor experiencia al encontrar el producto deseado.

3.2.51 Integración de API para Notificación de stock bajo: La plataforma web está conectada con una API para la notificación de stock bajo, la cual mediante correo electrónico envía una notificación que señala el producto y el total de stock disponible en inventario.

**Requerimientos para las Interfaces de usuarios:**

3.2.52 Sistema debe ser responsivo: La interfaz de la plataforma web se adapta automáticamente a las distintas resoluciones, desde dispositivos móviles hasta computadores, sin afectar el rendimiento, continuidad y usabilidad de la página web

3.2.53 UI web: El aplicativo web cuenta con una interfaz web que es intuitiva para los usuarios y de fácil uso, el 90% de las funcionalidades principales, se puede acceder en menos de 3 clicks

3.2.54 UI escritorio: El aplicativo de escritorio cuenta con una interfaz de escritorio que permite realizar la gestión de inventario, categorías, ordenes y usuarios mediante una interfaz intuitiva y de fácil uso, el tiempo de respuesta promedio para realizar las distintas tareas de carácter administrativo no supera los 15 segundos.

**Requerimientos para la sincronización de datos entre los sistemas Web y Escritorio:**

3.2.55 Sistema debe ser confiable: El Sistema de Byte Mart cuenta con una disponibilidad del sistema de un mínimo del 80% asegurando una operación continua durante las horas de más demanda.

3.2.56 Sistema debe ser eficiente: El sistema garantiza que las operaciones para la gestión de productos, categorías y usuarios debe realizarse en menos de 5 segundos, esto garantiza que la respuesta sea rápida y fluida.

3.2.57 Sistema debe ser escalable: El sistema puede manejar un aumento del 30% en los usuarios, además se considera un aumento del 50% del inventario sin realizar un reestructuración de la infraestructura

3.2.58 Sistema debe ser tolerable a los fallos: El sistema funciona parcialmente en caso de fallos, minimizando el impacto en los usuarios.

3.2.59 Sincronización de datos: El sistema web y el sistema de escritorio mantiene sus datos sincronizados con el fin de tener un respaldo de los datos en ambos aplicativos.

3.2.60 Respaldo de información: Ambos sistemas cuentan con una base de datos espejo la cual es almacenada y realiza un respaldo de la misma información generada en los aplicativos de producción.

**Usabilidad:**

El sistema debe ser fácil de usar y comprender.

**Fiabilidad:**

Los factores de fiabilidad para este sistema cuentan con un 85% bajo condiciones normales de funcionamiento. Además, considerar que ante fallos críticos, el sistema tiene un tiempo de recuperación no mayor a dos horas. Los incidentes permisibles dependen de su gravedad y serán considerados semanalmente para un mayor monitoreo y control de estos. Lo más importante de este punto y del punto de Seguridad, son los datos de los usuarios, los cuales deben estar protegidos ante posibles pérdidas, en caso de pérdidas, el sistema es capaz de restaurar la información en el menor tiempo posible.

**Portabilidad:**

El sistema Byte Mart no presenta especificaciones de portabilidad, por lo cual el software no contará con atributos para facilitar el traslado a otras plataformas y entornos.

Seguridad:

Considerando que el sistema de Byte Mart trata sobre un ecommerce, que maneje datos sensibles de los clientes, gestión de inventario y transacciones financieras. Tenemos que contar con un control para proteger al sistema de accesos no autorizados o modificaciones accidentales de productos o usuarios.

Se incorpora una base de datos espejo, que sirve como respaldo para la información, en caso de pérdidas o fallos en el sistema. También se adopta una política para garantizar la protección contra acceso no autorizados al sistema, un cambio periodico de contraseñas cada 90 días.

Interoperabilidad: El sistema debe interactuar con otros sistemas de manera exitosa, como por ejemplo la API de pago.

**Mantenibilidad:**

La mantenibilidad del sistema, ocurre por las siguientes opciones: errores, mejoras o prevención ante problemas del sistema.

* Errores: Errores que puedan surgir en la plataforma web, base de datos o app de escritorio, estos son detectados por usuarios o por sistemas de monitoreo.
* Mejoras: Implementación de nuevas tecnologías, mejoras de los procesos o interfaz del usuario. Esto depende de las necesidades del negocio o solicitudes de los usuarios.
* Prevención: Dependiendo de los estándares de rendimiento del sistema, se debe realizar el proceso de mantenibilidad de la infraestructura, arquitectura, equipos, para que no surja un problema mayor.

Todos estos puntos, deben ir con su correspondiente reporte y se realizarán dependiendo de su tipo en momentos y horarios de bajo flujo de usuarios.

Los encargados de realizar el proceso de mantenibilidad son principalmente:

* Usuarios finales: Principales notificadores de cualquier inconveniente que pueda presentar el sistema.
* Equipo de Desarrollo: Encargados de realizar este proceso, principalmente en la codificación general del sistema, ajustes de la base de datos y app de escritorio.
* Administrador del Sistema: Responsable de mantenimiento de la infraestructura de funcionamiento del sistema, así como también de los equipos, actualizaciones, manejo de respaldos, etc.

El proceso de mantenimiento en cuanto a frecuencia y tiempo:

Errores: tan pronto como se notifique o detecten errores, se debe aplicar las correcciones necesarias en un plazo máximo de **48 horas.**

Mejoras y prevención: Cada **3 meses**, el equipo de desarrollo realizará una revisión del sistema para identificar posibles problemas futuros o desarrollar e implementar nuevos requerimientos solicitados por el cliente.

Reportes: Los reportes de funcionamiento y rendimiento del sistema se realizan **semanalmente** para ser revisados por el equipo técnico.

**Disponibilidad:**

Se garantiza la disponibilidad total del sistema en un 85%, lo que conlleva a tener el sistema operativo en casi la totalidad de días en el año, el porcentaje restante corresponde a días en los que el sistema se pueda ver sometido a mantención, lo cual ocurrirá fuera de los horarios críticos y en caso de inactividad no planificada del sistema, se debe resolver en un tiempo máximo de dos horas.

El sistema es monitoreado constantemente para prevenir y mitigar caídas, además de tener la capacidad de soportar una alta demanda.

# Otros requerimientos

De momento, no se especifican otros requerimientos fuera de los mencionados anteriormente.

# Glosario

**Python Flask:** Flask es un framework ligero para crear aplicaciones web en Python. Es conocido por su simplicidad y flexibilidad, permitiendo a los desarrolladores crear aplicaciones web y APIs rápidamente. Flask sigue una arquitectura modular que permite usar solo los componentes necesarios, lo que lo hace ideal para proyectos pequeños a medianos.

**MySQL:** MySQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto. Utiliza SQL (Structured Query Language) como lenguaje para gestionar y consultar bases de datos. Es ampliamente utilizado en aplicaciones web y es conocido por su rendimiento, estabilidad y facilidad de uso. MySQL permite almacenar, modificar y gestionar grandes cantidades de datos de manera estructurada.

**GitHub:** GitHub es una plataforma de desarrollo colaborativo que permite almacenar código fuente utilizando Git como sistema de control de versiones. Ofrece herramientas para el control de versiones, colaboración entre equipos, revisión de código, seguimiento de errores y más. Es una herramienta clave en el desarrollo de software, ya que permite a los desarrolladores trabajar en equipo de manera eficiente y gestionar sus proyectos de forma organizada.

**APIs:** Una API es un conjunto de reglas y definiciones que permite que diferentes aplicaciones o sistemas se comuniquen entre sí. A través de las APIs, los desarrolladores pueden acceder a funcionalidades o datos de otras aplicaciones o servicios sin necesidad de conocer el código interno.

**HTTP:** HTTP es el protocolo base de la web, utilizado para la transmisión de datos entre un cliente (como un navegador) y un servidor web. Es un protocolo basado en solicitudes y respuestas: el cliente envía una solicitud HTTP al servidor, y este responde con los datos solicitados (como una página web, un archivo, etc.).

**Web Scraping:** El web scraping es la técnica de extraer información de sitios web de manera automatizada. Utilizando scripts o herramientas especializadas, se accede a páginas web y se recopilan datos estructurados (como texto, imágenes o tablas). Aunque puede ser útil para análisis de datos, vigilancia de precios, entre otros, es importante respetar las políticas de los sitios web y los aspectos legales del scraping.

**RPA:** Es la tecnología que permite automatizar tareas repetitivas mediante la creación de robots de software o scripts que imitan las acciones humanas en interfaces digitales. Se utiliza para realizar tareas como el procesamiento de datos, la generación de reportes, la extracción de información y otras actividades que consumen tiempo. Los robots de RPA pueden interactuar con aplicaciones y sistemas de la misma manera que lo haría una persona, mejorando la eficiencia y reduciendo el error humano.