|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fecha | Versión | Autor | Organización | Descripción |
| 2024-11-18 | **1.0** | **Luis Adrian Méndez**  **Kevin Saeteros** | **Espoch** | **Elección de las tecnologías para el desarrollo del e-comerce** |
| 2024-11-21 | **1.1** | **Luis Adrian Méndez**  **Kevin Saeteros** | **Espoch** | **Actualización de tecnologías** |
| 2024-12-18 | **1.2** | **Luis Adrian Méndez**  **Kevin Saeteros** | **Espoch** | **Inclusión de nuevas tecnologías implementadas** |
|  |  |  |  |  |

**Lenguajes de Programación**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lenguaje | Ventajas | Desventajas | Recomendación | Fuente |
| Python | Fácil de aprender, amplia comunidad, excelente para desarrollo web con Django/Flask y para IA/ML. | Menor rendimiento que lenguajes como Java o C++. | ✅ Ideal para Backend y AI. | [Documentación Python](https://www.python.org/doc/) |
| JavaScript | Lenguaje universal para frontend y backend (Node.js), gran comunidad, soporte de frameworks modernos como React y Vue. | Manejo complejo de asincronía y gran cantidad de bibliotecas desactualizadas. | ✅ Ideal para Frontend. | [Documentación JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript) |
| PHP | Popular para e-commerce con herramientas como WordPress o Laravel. | Considerado más antiguo, menos eficiente en grandes escalas frente a Python o Node.js. | ❌ No recomendado. | [Documentación PHP](https://www.php.net/docs.php) |
| Ruby | Conciso, utilizado en plataformas como Shopify. | Menor comunidad que Python y JS, menor rendimiento en procesos intensivos. | ❌ No recomendado. | Documentación Ruby  <https://www.ruby-lang.org/es/documentation/> |

**Elección**:

* **Backend**: JavaScript con Node.js, por su capacidad de unificar tecnologías entre frontend y backend, reduciendo la curva de aprendizaje y optimizando el desarrollo.
* **Frontend**: JavaScript con React.js para interfaces dinámicas y fluidas.

**IDEs (Entornos de Desarrollo Integrados)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| IDE | Ventajas | Desventajas | Recomendación | Fuentes |
| Visual Studio Code | Gratuito, ligero, extensiones para todos los lenguajes y frameworks, excelente para JS y Python. | No tiene funciones avanzadas de debugging integradas (sin extensiones). | ✅ Ideal para todo el stack. | <https://code.visualstudio.com/docs> |
| PyCharm | Excelente para Python, con herramientas de debugging avanzadas y gestión de entornos virtuales. | Pesado en sistemas con pocos recursos. | ✅ Para backend en Python. | <https://www.jetbrains.com/pycharm/documentation/> |

**Elección**:

* **Principal**: **Visual Studio Code** por su flexibilidad, soporte para múltiples lenguajes y extensiones.

**Frameworks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Framework | Lenguaje | Ventajas | Desventajas | Recomendación | Fuentes |
| Node.js con Express | JavaScript | Unifica frontend y backend con JavaScript, rápido, soporte para APIs REST. | Curva de aprendizaje para manejo de asincronía. | ✅ Ideal para Backend | Documentación Node.js y [Express](https://expressjs.com/) |
| React.js | JavaScript | Framework para interfaces de usuario, gran comunidad y rendimiento. | Dificultad en el manejo de componentes complejos. | ✅ Ideal para Frontend | [Documentación React](https://react.dev/) |
| Django | Python | Full-stack, incluye autenticación, administración y escalabilidad. | Menos flexible que Flask si se requiere personalización. | ✅ Ideal para Backend | [Documentación Django](https://docs.djangoproject.com/) |
| Flask | Python | Microframework, ligero y flexible, ideal para APIs personalizadas. | No incluye funcionalidades preconstruidas como Django. | ✅ Alternativa flexible | Documentación Flask |

**Elección**:

* **Backend**: **Node.js con express**, ya que ofrece todas las herramientas necesarias para realizar un e-commerce.
* **Frontend**: **React.js**, por su capacidad para crear interfaces rápidas y dinámicas.

**Sistemas de Gestión de Bases de Datos (SGBD)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SGBD | Tipo | Ventajas | Desventajas | Recomendación | Fuentes |
| PostgreSQL | Relacional | Alto rendimiento, escalabilidad, excelente manejo de datos complejos, código abierto. | Requiere más configuración inicial comparado con MySQL. | ✅ Ideal para el proyecto | Documentación PostgreSQL  <https://www.postgresql.org/docs/> |
| MySQL | Relacional | Popularidad, fácil de usar, amplia documentación. | Menos eficiente en manejo de datos complejos que PostgreSQL. | ✅ Alternativa válida | [Documentación MySQL](https://dev.mysql.com/doc/)  <https://dev.mysql.com/doc/> |
| SQLite | Relacional | Ligero, ideal para pruebas locales. | No recomendado para sistemas con alto tráfico. | ❌ Solo para pruebas | Documentación SQLite  <https://www.sqlite.org/docs.html> |
| Supabase | Plataforma de base de datos basada en PostgreSQL | Fácil de usar, ofrece autenticación y API REST integradas, ideal para proyectos ágiles con tiempos ajustados. | Dependencia de un proveedor externo, menos opciones de personalización avanzada comparado con PostgreSQL puro. | ✅ Ideal para el proyecto | Documentación Supabase  <https://supabase.com/docs> |

**Elección:**

* **Principal:** PostgreSQL para robustez y escalabilidad.
* **Alternativa:** Supabase, para integraciones rápidas en proyectos con tiempos ajustados.

**Object-Relational Mapping (ORM)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ORM | Lenguaje | Ventajas | Desventajas | Recomendación | Fuentes |
| Prisma | JavaScript | Intuitivo, generación automática de migraciones y tipos, integración con TypeScript. | Necesita configuración inicial. | ✅ Ideal para el proyecto | Documentación Prisma  <https://www.prisma.io/docs> |
| SQLAlchemy | Python | Madurez, flexibilidad, integración con varios motores de bases de datos. | Mayor complejidad para proyectos pequeños. | ✅Ideal para Python | Documentación SQLAlchemy  <https://docs.sqlalchemy.org/en/20/> |
| Sequelize | JavaScript | Compatible con varios motores, buen soporte para Node.js y bases relacionales. | Configuración inicial puede ser engorrosa. | ✅ Alternativa válida | Documentación Sequelize  <https://sequelize.org/> |

**Elección:**

**Principal:** Prisma es la opción ideal como ORM para el proyecto debido a su perfecta integración con JavaScript, facilidad de uso con generación automática de migraciones.