STACK TECNOLOGICO

SENA – SOFT 2025 – DESARROLLO LIBRE

PROYECTO: APLICACIÓN WEB DE COMPRA DE TIKETES AHEREOS – SKY ROUTE

ALEJANDRO HERNANDEZ

DANIEL VELOZA

SENA



Centro de Gestión de Mercados, Logística y Tecnologías de la Información.

21/10/2025

**INTRODUCCION**

El presente documento tiene como propósito describir el Stack Tecnológico propuesto para el desarrollo del sistema web “SkyRoute”, un proyecto orientado a la gestión integral del proceso de compra, reserva y emisión de tiquetes aéreos.

El objetivo principal es definir las tecnologías, herramientas y entornos que se emplearán en las diferentes capas del sistema, asegurando la coherencia con la arquitectura Modelo–Vista–Controlador (MVC), que permite separar de forma estructurada la lógica de negocio, la presentación y la gestión de datos.

Este stack fue seleccionado con base en criterios de rendimiento, compatibilidad, seguridad, mantenibilidad y facilidad de implementación, garantizando así un desarrollo ágil, colaborativo y sostenible.

El documento detalla las herramientas que componen cada capa del sistema (frontend, backend y base de datos), los mecanismos de control de versiones y las utilidades de apoyo en diseño, pruebas y documentación técnica.  
De esta manera, se establece una base tecnológica sólida para el desarrollo y despliegue del proyecto, cumpliendo con los estándares de calidad exigidos en la competencia SENASoft 2025.

**FRONTEND**

Tecnologías:

* HTML5: estructura de las vistas del sistema.
* CSS3: estilos personalizados y diseño adaptable.
* Bootstrap 5: framework CSS para lograr interfaces limpias y responsivas.
* EJS (Embedded JavaScript Templates): motor de plantillas usado por Node.js para generar las vistas dinámicas desde el servidor.

Justificación:  
Las vistas serán renderizadas directamente desde los controladores en el servidor, siguiendo el flujo MVC.  
Esto permite mantener toda la aplicación en un solo entorno de ejecución y simplifica la integración entre las capas del sistema.

**BACKEND**

Tecnologías:

* Node.js: entorno de ejecución JavaScript orientado a eventos, ideal para aplicaciones web dinámicas.
* Express.js: framework para gestionar rutas, peticiones HTTP y controladores.
* MVC (Modelo–Vista–Controlador):
  + Modelos: definen la lógica de datos y conexión con MySQL.
  + Vistas: se generan con EJS y reciben datos desde los controladores.
  + Controladores: manejan la lógica del negocio y el flujo entre vistas y modelos.

Justificación:  
El patrón MVC permite una separación clara entre la lógica de negocio, la presentación y los datos, lo que facilita el mantenimiento, la escalabilidad y la reutilización del código.

**BASE DE DATOS**

Tecnología:

* MySQL 8.0 con motor InnoDB
* Administrado localmente mediante XAMPP

Justificación:  
MySQL ofrece integridad referencial, soporte de transacciones y facilidad de integración con Node.js a través de librerías como mysql2 o sequelize.  
Se emplea para gestionar usuarios, vuelos, asientos, reservas, pagos y relaciones entre entidades.

**CONTROL DE VERSIONES**

Tecnología:

* GitHub con estrategia GitFlow

Justificación:  
Se usa GitHub como repositorio central y GitFlow para gestionar ramas de desarrollo (develop, feature, release, main).  
Esto garantiza trabajo colaborativo ordenado y control de versiones durante la competencia.

**DISEÑO UML UX/UI**

Herramientas:

* Draw.io: para modelado UML (casos de uso, secuencia, clases).
* Figma: para prototipado visual y diseño de la interfaz.

Justificación:  
Estas herramientas permiten mantener coherencia entre el diseño visual y la arquitectura funcional del sistema.

**Entorno de Ejecución Local**

Herramientas:

* XAMPP: servidor local que provee MySQL y Apache para pruebas de conexión.
* Visual Studio Code: entorno de desarrollo principal con soporte para Node.js.

Justificación:  
Proporciona un entorno unificado para desarrollo, depuración y pruebas sin necesidad de infraestructura remota.

**SEGURIDAD Y AUTENTICACIÓN**

Tecnologías complementarias:

* bcrypt.js: para encriptar contraseñas.
* express-session: para manejar sesiones de usuario.
* dotenv: para proteger credenciales y configuraciones sensibles.

Justificación:  
Estas herramientas garantizan un manejo seguro de la autenticación de usuarios y la persistencia de sesiones entre vistas.

**RESUMEN DEL STACK**

|  |  |  |
| --- | --- | --- |
| Capa | Tecnología | Rol |
| Frontend (Vistas) | HTML5, CSS3, Bootstrap, Jquery, Javacript | Interfaz del usuario renderizada en servidor |
| Backend (Servidor) | PHP | Lógica de negocio y control de flujo MVC |
| Base de Datos | MySQL (InnoDB) | Almacenamiento relacional |
| Arquitectura | MVC | Separación lógica y mantenibilidad |
| Control de Versiones | GitHub + GitFlow | Flujo ordenado de desarrollo |
| Diseño / UML | Figma, Draw.io | Documentación visual y prototipado |
| Entorno Local | XAMPP, VS Code | Desarrollo y pruebas locales |
| Seguridad | bcrypt.js, express-session, dotenv | Autenticación y cifrado |

**CONCLUSIÓN**

El stack propuesto implementa un enfoque MVC puro de la mano de php puro, en el cual se gestiona tanto la lógica como la renderización de vistas dinámicas.  
Esta arquitectura asegura simplicidad, claridad y un control total del flujo de datos desde el modelo hasta la interfaz del usuario, lo que la hace ideal para un proyecto académico o competitivo como SENASoft.