INSTITUTO TECNOLÓGICO SUPERIOR DE SAN PEDRO DE LAS COLONIAS

INGENIERÍA EN SISTEMAS COMPUTACIONALES

INGENIERÍA DE SOFTWARE

GRUPO: 6 A

SPRINT: 1

DEFINICIÓN DEL PROYECTO

PERIODO:

9 a 19 DE FEBRERO

DOCENTE:

RUTH AIVI CHÁVEZ RODRÍGUEZ

NOMBRE DE LOS INTEGRANTES:

MADDOX KALEF REGALADO SANCHEZ

KAROL DAVID KALDERON BORDALLO

JESUS JAVIER PECINA RODRÍGUEZ

BARUK OMAR GARZA PINEDA

ALEXIS ALVARADO GOMEZ

San Pedro, Coahuila de Zaragoza, a 12 de diciembre de 2026

Problema

Existe una brecha de conocimiento en programación entre los estudiantes de nuevo ingreso. Muchos inician la carrera sin bases previas en conceptos fundamentales, herramientas o contexto laboral.

Esto provoca:

- Bajo rendimiento en materias iniciales.
- Confusión en conceptos básicos.
- Desmotivación.
- Posible abandono temprano.

Problema central definido correctamente:

No existe un recurso introductorio estructurado, accesible y organizado que prepare al estudiante antes o durante sus primeras materias de programación.

Objetivo General

Desarrollar una plataforma digital introductoria a la programación que proporcione conocimientos básicos estructurados, ejercicios prácticos y orientación general sobre el campo laboral.

Usuarios del Sistema

Usuario principal:

Estudiantes de nuevo ingreso con poco o ningún conocimiento en programación.

Usuario secundario:

Personas de la comunidad interesadas en aprender desde cero.

Alcance del Proyecto

Incluye:

- Contenido teórico básico.
- Ejemplos prácticos.
- Ejercicios introductorios.
- Orientación general sobre campo laboral.
- Registro simple para acceso al contenido.

No incluye:

- Certificaciones oficiales.
- Sustituir materias formales.
- Clases en tiempo real.
- Tutorías personalizadas.
- Sistema de pagos.
- Desarrollo de app móvil nativa (solo versión web).

Viabilidad del Proyecto

Viabilidad técnica

El proyecto es técnicamente viable debido a que puede desarrollarse utilizando tecnologías accesibles y ampliamente conocidas en el desarrollo web.

- Puede desarrollarse con HTML, CSS y JavaScript.
- Puede utilizar frameworks básicos.
- Puede alojarse en plataformas gratuitas de hosting.
- El sistema de registro es sencillo (usuario y contraseña).
- No requiere infraestructura compleja.
- Puede funcionar en computadoras y dispositivos móviles.
- No necesita servidores avanzados ni equipos costosos.

2. Viabilidad Económica

El proyecto es económicamente viable debido a que no requiere grandes inversiones iniciales ni costos operativos elevados.

- Puede desarrollarse con software libre.
- Puede alojarse en servicios gratuitos o de bajo costo.
- No requiere personal especializado permanente.
- Puede realizarse con presupuesto reducido.

3. Viabilidad Operativa

El proyecto es operativamente viable ya que responde a una necesidad real y puede implementarse dentro del contexto académico actual.

- Atiende la falta de orientación inicial.
- Tiene un público claramente definido.

- Puede integrarse como apoyo académico.
- Es fácil de usar.
- Los estudiantes pueden aprovecharlo sin capacitación previa.

4. Viabilidad Académica

El proyecto tiene viabilidad académica debido a su contribución directa al proceso de aprendizaje.

- Apoya el aprendizaje autónomo.
- Refuerza contenidos básicos.
- Contribuye a reducir la deserción.
- Mejora el rendimiento académico.
- Posee valor formativo.

¿Por qué se afirma que disminuye la deserción?

La plataforma contribuye a disminuir la deserción debido a que brinda orientación inicial, reduciendo la incertidumbre y ansiedad que muchos estudiantes experimentan al comenzar la carrera. Además, refuerza contenidos básicos, evitando rezagos académicos que suelen provocar bajo rendimiento. Facilita el aprendizaje autónomo, permite el acceso constante al material y mejora la adaptación académica, uno de los factores más relacionados con el abandono escolar. Cuando el estudiante comprende mejor los contenidos y se siente acompañado en su proceso, aumenta su motivación y permanencia.

¿Cómo se comprueba que tiene valor formativo?

El valor formativo del proyecto puede comprobarse mediante la aplicación de evaluaciones diagnósticas y finales, comparando el nivel de conocimientos antes y después de utilizar la plataforma. Asimismo, pueden analizarse indicadores de rendimiento académico, tasas de permanencia, encuestas de satisfacción estudiantil y métricas de uso del sistema, como participación, actividades completadas y progreso alcanzado.

Requisitos Funcionales

RF-1 Registro de usuario

El sistema deberá permitir el registro mediante nombre, nombre de usuario y contraseña.

RF-2 Inicio y cierre de sesión

El sistema deberá permitir a los usuarios iniciar y cerrar sesión mediante usuario y contraseña.

RF-3 Acceso a contenido por módulos

El sistema deberá mostrar contenido estructurado en módulos (conceptos básicos, algoritmos, variables, tipos de datos y estructuras de control).

RF-4 Ejercicios prácticos

El sistema deberá incluir ejercicios básicos de práctica y mostrar retroalimentación básica.

RF-5 Persistencia de progreso

El sistema deberá guardar el avance del usuario en cada módulo y ejercicio.

Requisitos No Funcionales

RNF-1 Usabilidad

El sistema deberá tener una interfaz clara y sencilla para principiantes.

RNF-2 Comprensibilidad

El contenido deberá utilizar lenguaje simple y accesible para usuarios sin experiencia previa.

RNF-3 Plataforma

El sistema será una aplicación web accesible desde navegador.

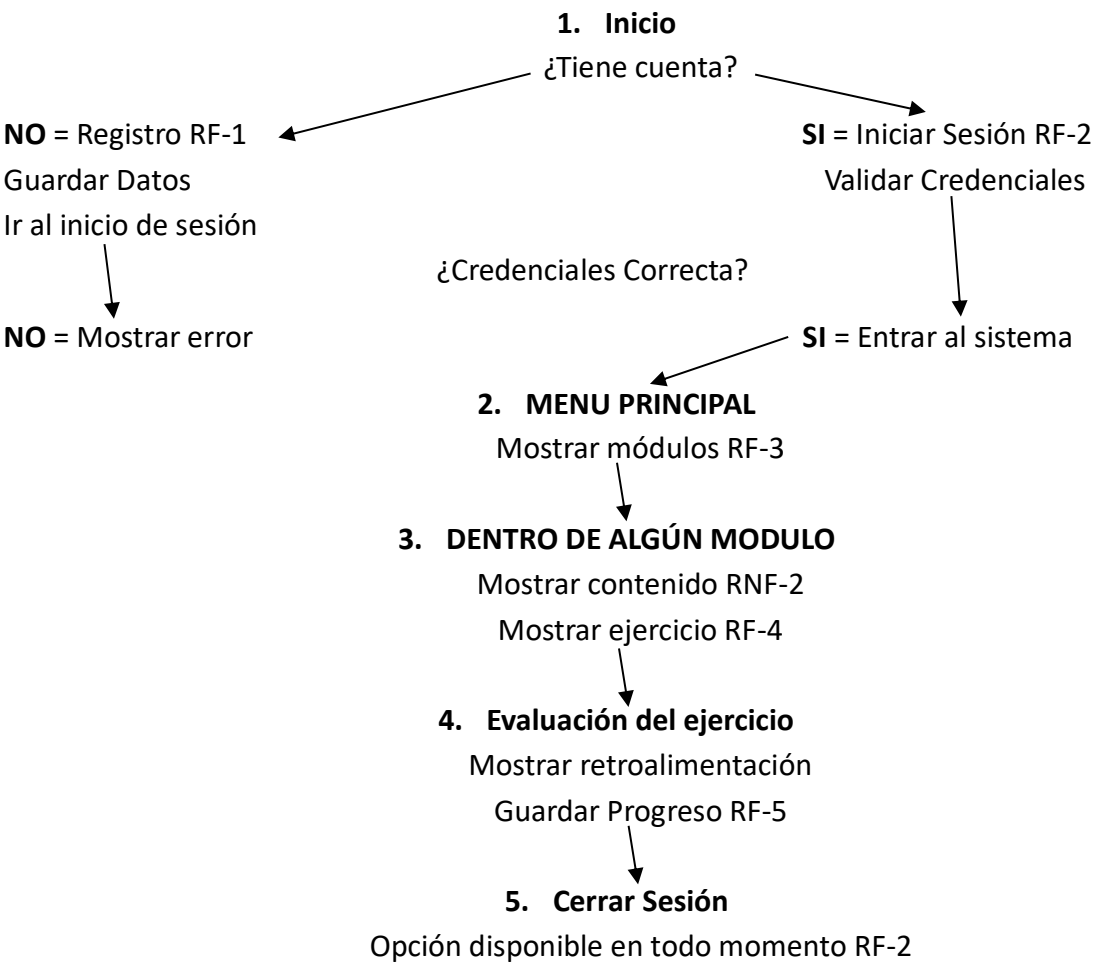
RNF-4 Seguridad básica

El sistema deberá proteger los datos del usuario mediante autenticación y almacenamiento seguro de contraseñas.

RNF-5 Rendimiento

El sistema deberá cargar cada página en un tiempo máximo de 3 segundos bajo condiciones normales.

FLUJO DEL SISTEMA SIMPLE



Cambios

Cambio 1: Definir claramente el problema central

Antes:
El problema estaba descrito de forma general y emocional.

Ahora:
Se definió una causa concreta: ausencia de un recurso estructurado.

¿Por qué?

Si el problema no es claro, la solución puede desviarse.

¿Qué evita?

Evita desarrollar algo que no ataque el problema real.

Cambio 2: Separar Usuarios de Requisitos

Antes:
Se mezclaban funciones del sistema con perfiles de usuario.

Ahora:
Usuarios = quién usa.
Requisitos = qué hace el sistema.

¿Por qué?

Mezclar conceptos genera confusión en desarrollo.

¿Qué evita?

Errores en diseño de base de datos y lógica del sistema.

Cambio 3: Delimitar el alcance (lo que NO incluye)

Antes:
El proyecto podía interpretarse como una plataforma educativa completa.

Ahora:
Se especifica lo que no incluye.

¿Por qué?

Un proyecto sin límites crece sin control.

¿Qué evita?

- Sobrecarga de trabajo.
- Cambios constantes.
- Retrasos en entrega.
- Conflictos en el equipo.

Cambio 4: Agregar contraseña en registro

Antes:
Solo nombre y correo.

Problema:
No hay autenticación real.

Ahora:
Se agrega contraseña.

¿Por qué?

Sin contraseña no hay control de acceso.

¿Qué evita?

Problemas de seguridad y errores estructurales en el sistema.

Cambio 5: Especificar que es plataforma web

Antes:
No estaba definido si era app, web o PDF.

Ahora:
Se define como aplicación web.

¿Por qué?

Cada tipo requiere tecnologías diferentes.

¿Qué evita?

Replantear todo el proyecto a mitad del desarrollo.

Por qué estas decisiones son importantes

En ingeniería de software:

Decisiones mal definidas al inicio →
Errores en diseño →
Cambios costosos →
Retrasos →
Confusión →
Proyecto mal evaluado.

Un análisis claro:

- Reduce riesgos.
- Facilita programación.
- Mejora organización.

- Evita contradicciones.
- Hace que el sistema sea coherente.

Conclusión

Las mejoras realizadas permiten que el proyecto:

- Sea claro.
- Sea técnicamente viable.
- Tenga límites definidos.
- Evite ambigüedades.
- Pueda programarse sin reinterpretaciones constantes.