# **Revisión Técnica del Proyecto Final 2025**

## **Definición de la Fase de Análisis**

En esta fase se realiza una recolección y procesamiento de datos alimenticios proporcionados por los usuarios, con el objetivo de ofrecer recomendaciones personalizadas y detectar hábitos no saludables. El sistema utiliza información ingresada como edad, peso, estatura, nivel de actividad física y alimentos consumidos diariamente para realizar los siguientes análisis:

* **Cálculo del IMC (Índice de Masa Corporal):** Se calcula utilizando la fórmula clásica (peso/estatura²). Este valor permite clasificar al usuario según rangos establecidos por la OMS: bajo peso, normal, sobrepeso u obesidad.
* **Requerimientos calóricos diarios:** Se calcula el requerimiento energético diario utilizando la fórmula de Harris-Benedict adaptada según sexo, edad y nivel de actividad física. Este cálculo se contextualiza con los hábitos y disponibilidad alimenticia de Guatemala.
* **Comparación de calorías ingeridas vs. recomendadas:** El sistema compara las calorías ingresadas por el usuario (según los alimentos consumidos) con las calorías recomendadas según su perfil.
* **Detección de desequilibrios nutricionales:** Se identifican carencias o excesos en macronutrientes (grasas, proteínas, carbohidratos), y en micronutrientes esenciales si los datos lo permiten.
* **Alertas o recomendaciones preliminares:** Si se detecta un patrón repetido de desequilibrio, el sistema genera mensajes como “Consumo excesivo de azúcares” o “Bajo consumo de vegetales”, recomendando una mejora.

## **Determinación de Variables y Estándares de Programación**

### **Variables Principales**

* peso, estatura, edad, sexo, actividadFisica, caloriasIngeridas, macronutrientes.
* imc, caloriasRecomendadas, desequilibrios, alertas, usuarioID.

### **Estructuras de Datos**

* Estructuras tipo struct para representar usuarios, alimentos, menús diarios.
* Vectores y listas enlazadas para almacenar registros temporales.
* Arreglos multidimensionales para análisis de tendencias.

### **Estándares Aplicados**

* Nomenclatura en español usando camelCase.
* Comentarios descriptivos para cada función.
* Código modular y reutilizable.
* Convenciones de codificación recomendadas por el estilo de Visual Studio 2022.

### **Prácticas de Colaboración**

* Repositorio GitHub para trabajo en equipo.
* Control de versiones con ramas por módulo.
* Documentación y revisión por pares.

## **Definición y Progreso de la Interfaz de Usuario**

### **Tipo de Interfaz**

* Consola (basada en WinAPI) con diseño textual interactivo.

### **Pantallas Implementadas**

* Registro de usuario
* Ingreso de datos personales
* Registro de alimentos consumidos
* Reporte diario de análisis nutricional

### **Diseño de Experiencia de Usuario**

* Menús claros y navegación paso a paso.
* Validación de entradas del usuario.
* Mensajes y sugerencias personalizadas en base a resultados.

### **Plan de Mejoras**

* Incorporar una versión gráfica simple.
* Agregar más colores y símbolos para representar gráficamente el estado nutricional.

## **Estructura de Base de Datos**

### **Descripción de Tablas Principales**

* Usuarios: ID, nombre, edad, sexo, peso, estatura, actividad física.
* Alimentos: ID, nombre, calorías, macronutrientes.
* Registros: ID, usuarioID, fecha, alimentos consumidos, calorías totales.
* Alertas: ID, usuarioID, tipo de alerta, recomendación.

### **Relaciones**

* Uno a muchos entre Usuarios y Registros.
* Muchos a muchos entre Registros y Alimentos mediante tabla intermedia.

### **Justificación del Modelo**

* Modelo relacional adaptado a seguimiento nutricional personalizado.
* Pensado para mantener coherencia y escalabilidad de datos.

### **Herramienta**

* Microsoft SQL Server Express.

### **Pruebas**

* Se realizaron pruebas de inserción, actualización y consultas JOIN entre tablas con éxito.

## **Gestión del Proyecto con GitHub**

Con estas herramientas de GitHub logré llevar un seguimiento claro y ordenado de cada etapa del proyecto.  
Esto no solo me ayudó a **organizar mejor el trabajo**, sino también a **priorizar tareas**, **detectar posibles retrasos** y mantener un flujo de trabajo constante.

Además, el uso de estas herramientas fomenta el trabajo en equipo. Por ejemplo, si varias personas están colaborando en un mismo repositorio, cada uno puede ver qué está haciendo el otro, tomar tareas pendientes, dejar comentarios o sugerencias, y evitar que se repitan trabajos. Todo queda registrado de forma transparente.

Gracias a la gestión con GitHub, pude mantener la **trazabilidad del proyecto**, es decir, saber quién hizo qué, cuándo y por qué. Esto es especialmente útil si más adelante hay que revisar decisiones o corregir errores.

En conclusión, GitHub no es solo una herramienta para programar, sino también una excelente plataforma para **planificar, coordinar y ejecutar proyectos de forma profesional**. Usarlo como herramienta de gestión mejora tanto la organización como la calidad del producto final.

## **Análisis de Eficiencia del Código**

* Se detectó que los ciclos anidados en el análisis de tendencias eran los menos eficientes.
* Se mejoró el uso de funciones mediante modularización.
* Uso eficiente de punteros para manejo dinámico de memoria.
* Lectura y escritura a base de datos optimizada con consultas parametrizadas.

## **Análisis Crítico del Porcentaje de Avance**

### **Porcentaje Completado**

* Aproximadamente 75% del desarrollo.

### **Obstáculos Enfrentados**

* Dificultades al integrar la base de datos al backend.
* Ajustes a las fórmulas nutricionales para el contexto guatemalteco.

### **Lecciones Aprendidas**

* La planificación detallada inicial es clave.
* Importancia de testear desde fases tempranas.

### **Plan para Fase 3**

* Finalizar interfaz completa.
* Agregar módulo de gráficos nutricionales.
* Pruebas con usuarios reales para validar recomendaciones.

## **Evidencia del Progreso**

### **Fragmentos de Código Comentado**

float calcularIMC(float peso, float estatura) {  
 return peso / (estatura \* estatura);  
}  
  
void generarAlerta(float imc) {  
 if (imc >= 25.0) cout << "Riesgo de sobrepeso" << endl;  
 else if (imc < 18.5) cout << "Bajo peso detectado" << endl;  
}

### **Reportes o Resultados de Análisis**

Usuario: Juan Pérez  
IMC: 28.4 (Sobrepeso)  
Calorías recomendadas: 2400 kcal  
Calorías ingeridas: 3100 kcal  
Recomendación: Reducir consumo de grasas y azúcares.