CURSO EBAC – IT START

**ÍNDICE ANALÍTICO**

1.-Introducción-----------------------------------------------------------------------------------------4

1.1-Meta-------------------------------------------------------------------------------------------------4

1.2-Alcance---------------------------------------------------------------------------------------------4

1.3-Definiciones, Acrónimos y Abreviaturas----------------------------------------------------4

1.4-Referencias---------------------------------------------------------------------------------------4

1.5-Visión general------------------------------------------------------------------------------------4

2.-Representación Arquitectónica ----------------------------------------------------------------4

3.-Objetivos y limitaciones de la arquitectura--------------------------------------------------4

4.-Vista de caso de uso------------------------------------------------------------------------------5

4.1-Realizaciones de casos de uso--------------------------------------------------------------5

5.-Visión lógica----------------------------------------------------------------------------------------5

5.1-Visión general------------------------------------------------------------------------------------5

5.2-Paquetes de diseño arquitectónicamente significativos--------------------------------5

6.-Visión de proceso----------------------------------------------------------------------------------5

7.-Visión de Implementación-----------------------------------------------------------------------5

8.-Visión de Implementación-----------------------------------------------------------------------5

8.1-Visión general------------------------------------------------------------------------------------5

8.2-Capas-----------------------------------------------------------------------------------------------6

9.-Vista de datos (opcional)------------------------------------------------------------------------6

10-Tamaño y rendimiento---------------------------------------------------------------------------6

11-Calidad----------------------------------------------------------------------------------------------6

**DOCUMENTO DE ARQUITECTURA DE SOFTWARE**

**1.-Introducción**

El presente software llamado *“nutrIA”* el cual es una aplicación donde el usuario vaciara sus datos como:

-Peso.

-Altura.

-Índice de masa corporal.

-Que le gustaría definir en su cuerpo (Reducir grasa, Aumentar masa corporal, Marcar).

-Si entrena en Gimnasio o en casa (y con que aparatos cuenta en casa)

-Comida en general que si puede consumir.

Estos datos proporcionados sirven para crear una dieta diaria / Semanal / Quincenal / Mensual, elaborada específicamente para el usuario y así mismo una rutina de ejercicio dependiendo de lo que el usuario selecciono (Las cuales las rutinas se pueden cambiar conforme pasa el tiempo, para modificarlas o cambiarlas).

Así mismo incluye videos donde se representan como realizar de manera adecuada y segura todos los ejercicios, así mismo cuantas series y repeticiones de los mismo.

**1.1-Meta**

El presente software esta programado mediante en lenguaje “JavaScript” (El cual se anexa también el archivo en formato .js para la verificación de la fuente de los códigos).

Dicho Software es para motivar a las personas a hacer ejercicio, a ayudarlas a que empiecen a tener una dieta saludable, esto para beneficiar a su salud en todos los sentidos.

Acorde a cada persona en específico con la información vaciada de cada uno, con la ayuda de más de 15 Nutriólogos y 20 fisiculturistas donde cada uno colaboro para implementar y nutrir la información presentada en el código, para poder personalizar tanto las rutinas, la comida y tener una rotación semanal de cada una de ellas, esto para que no se vuelva aburrido o monótono a los usuarios.

La arquitectura de la aplicación, se realizo para la optimización en dispositivos móviles, tanto en sistema Operativo IOS como en Android, por lo que los usuarios podrán descargar directamente desde sus dispositivos la aplicación, la cual es una aplicación muy amigable con el usuario, muy intuitiva y sobre todo le damos el poder al usuario de poder modificar y cambiar muchas cosas, para ajustarse y adaptarse a cada usuario.

**1.2-Alcance**

Las limitaciones que tiene el programa serían los siguientes dependiendo de que membrecía pague el usuario:

-Gratis: Se da 1 semana de prueba en membrecía Hércules.

-Membrecía Hércules: El pago es de $399.00 MNX Mensuales, el cual incluye 1 mes personalizado de rutinas (Estas dependerán de la meta del usuario) y de 1 mes de comidas personalizadas para el usuario, 1 sesión con nutriólogo y 1 sesión con Fisiculturista.

-Membrecía Zeus: El pago es de $599.00 MNX, el cual incluye lo descrito en la membrecía Hércules, más aparte semanalmente se tiene 1 sesión con nutriólogo, 1 sesión con Fisiculturista, “Lives” diarios con rutinas como zumba, aeróbicos, cardio, Yoga y 15% de descuento en los artículos en venta de la página.

La aplicación esta orientado a personas mayores de 18 años en adelante, que tengan la condición y fuerza física para que puedan desempeñar las rutinas de la mejor manera sin afectar o dañar su cuerpo.

**1.3-Definiciones, Acrónimos y Abreviaturas**

[Esta subsección contiene las definiciones de todos los términos, siglas y abreviaturas necesarias para interpretar correctamente elDocumento de arquitectura de software. Esta información se puede proporcionar consultando el Glosario del proyecto.]

**1.4-Visión general**

[Esta subsección describe lo que el resto de losDocumento de arquitectura de software contiene y explica cómo está organizado el documento.]

**2.-Representación Arquitectónica**

[Esta sección describe cuál es la arquitectura de software del sistema actual y cómo se representa. Desde elVista de caso de uso,Visión Lógica,Visión de proceso,Vista de Implantación EsVista de Implementación, enumera las vistas requeridas y, para cada vista, explica qué tipos de elementos de modelo contiene.]

**3.-Objetivos y limitaciones de la arquitectura**

[Esta sección describe los requisitos y objetivos del software que tienen algún impacto en la arquitectura; por ejemplo, seguridad, garantía, privacidad, uso de un producto listo para usar desarrollado internamente, portabilidad, distribución y reutilización. También captura las restricciones especiales que pueden aplicarse: estrategia de diseño e implementación, herramientas de desarrollo, estructura del equipo, cronograma, código fuente heredado, etc.]

**4.-Vista de caso de uso**

[Esta sección enumera casos de uso o escenarios de modelos de casos de uso cuando representan una funcionalidad central y significativa del sistema final o, cuando tienen una gran huella arquitectónica: experimentan muchos elementos arquitectónicos o cuando enfatizan o ilustran un punto complejo y específico de la arquitectura.]

**4.1-Realizaciones de casos de uso**

[Esta sección ilustra cómo funciona el software, presenta algunas realizaciones (o escenarios) de casos de uso seleccionados y explica cómo los diversos elementos del modelo de diseño contribuyen a su funcionalidad.]

**5.-Visión lógica**

[Esta sección describe las partes arquitectónicamente significativas del modelo de diseño, como su división en subsistemas y paquetes. Además, para cada paquete significativo, muestra su desglose en clases y utilidades de clase. Presente las clases arquitectónicamente significativas y describa sus responsabilidades, así como algunas relaciones, operaciones y atributos importantes.]

**5.1-Visión general**

[Esta subsección describe la descomposición completa del modelo de diseño en términos de capas y la jerarquía del paquete.]

**5.2-Paquetes de diseño arquitectónicamente significativos**

[Para cada paquete importante, incluya una subsección con su nombre, una breve descripción y un diagrama de todos los paquetes y clases importantes que contiene.

Para cada clase significativa en el paquete, incluya su nombre, una breve descripción y, opcionalmente, una descripción de algunas de sus responsabilidades, operaciones y atributos clave.]

**6.-Visión de proceso**

[Esta sección describe la descomposición del sistema en procesos livianos (hilos únicos de control) y procesos pesados (grupos de procesos livianos). Organice la sección en grupos de procesos que se comunican o interactúan. Describa los principales modos de comunicación entre procesos, como el paso de mensajes y las interrupciones.]

**7.-Visión de Implementación**

[Esta sección describe una o más configuraciones de la red física (hardware) en la que se implementa y ejecuta el software. Es una vista del modelo de implementación. Como mínimo, para cada configuración, debe indicar los nodos físicos (computadoras, CPU) que ejecutan el software y sus interconexiones (bus, LAN, punto a punto, etc.).Visión de proceso en los nodos físicos.]

**8.-Visión de Implementación**

[Esta sección describe la estructura general del modelo de implementación, la división del software en capas y subsistemas en el modelo de implementación y todos los componentes arquitectónicos significativos.]

**8.1-Visión general**

[Esta subsección nombra y define las diversas capas y su contenido, las reglas que determinan la inclusión en una capa en particular y los límites entre las capas. Incluya un diagrama de componentes que muestre las relaciones entre las capas. ]

**8.2-Capas**

[Para cada capa, incluya una subsección con su nombre, una lista de subsistemas ubicados en la capa y un diagrama de componentes.]

**9.-Vista de datos (opcional)**

[Una descripción de la perspectiva de almacenamiento de datos persistentes del sistema. Esta sección es opcional si los datos persistentes son escasos o inexistentes, o si la conversión entre el modelo de diseño y el modelo de datos es trivial.]

**10.-Tamaño y rendimiento**

[Una descripción de las características clave de dimensionamiento del software que tienen un impacto en la arquitectura, así como las limitaciones en el rendimiento deseado.]

**11.-Calidad**

[Una descripción de cómo la arquitectura del software contribuye a todas las funciones (excepto la funcionalidad) del sistema: extensibilidad, confiabilidad, portabilidad, etc. Si estas características tienen un significado especial, como implicaciones de seguridad, garantía o privacidad, deben estar claramente delineadas.]