ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN (DCC0)

CARRERA EN INGENIRÍA EN TIC'S

INGENIERÍA DE SOFTWARE II

NRC: *8517*

PROFESOR: *EFRAIN RODRIGO FONSECA CARRERA*

INTEGRANTES DEL GRUPO 3: *MARLON CEVALLOS, DYLAN JIMENEZ, VALERY NARANJO, ROBERTO PALLO Y RÓMULO PARDO.*

FECHA: *11/12/2022*

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

INTRODUCCIÓN

Un problema social de los últimos años ha sido la pandemia por COVID 19, razón por la cual las empresas de cadenas alimenticias tuvieron que cerrar sus negocios de manera parcial o permanente, perdiendo clientes y proveedores importantes. Según el censo realizado por el INEC, 2019; en el Ecuador se consideraron 12087 locales de comida y actualmente esta cifra disminuyó un 40% de los locales de comida registrados. [12]

Para ello los negocios o empresas tomaron alternativas como entregas de alimentos a domicilio, preparación de alimentos ya no en los locales sino en los hogares y finalmente entregas de pedidos a través de plataformas como páginas web y aplicaciones móviles.

El propósito de este trabajo es presentar la arquitectura de una aplicación web, la cual permita desarrollar una página web que ayude a gestionar los platillos y menús que ofrecen los restaurantes, generar pedidos en línea y administrar clientes registrados.

1. Objetivos

Objetivo General

Plantear una arquitectura de software analizando un caso de estudio práctico considerando la evolución del software, para el desarrollo de una aplicación web de gestión de pedidos.

Objetivos Específicos

- Proponer la problemática basado en un caso de estudio práctico.
- Identificar los requerimientos del sistema que producen impacto en la estructura del software.
- Plantear y diseñar la arquitectura del sistema utilizando el patrón de diseño MVC.
- Detallar la metodología, métodos y técnicas usadas en la arquitectura propuesta.

2. Desarrollo

2.1 Problemática

El restaurante "Mis Delicias" requiere una aplicación web que le permita visualizar los platillos y menús que ofrece a sus clientes, automatizar el proceso de pedidos y la gestión de clientes en el cual se registrará los datos personales y ubicación del cliente para su entrega a domicilio.

2.1.1 Características

- Es responsive: lo que significa hacer que un sitio web sea accesible y adaptable en todos los devices: tablets, smartphones, etc.
- Atractivo visual (código de colores): se utiliza la paleta de colores, la cual empieza con los colores primarios y llena el espectro para crear una útil y completa paleta para el sitio web. Se sugiere usar los colores 500 como los colores primarios en el sitio web y los otros colores como colores de acento.
- Es dinámica: el sitio web permite la interacción con el usuario para crear experiencias personalizadas y únicas.
- Es simplificada: permite una interacción más fácil con el sitio web.

2.1.2 Requerimientos generales del sistema

Tabla 1: Requerimiento general, registro de cliente

HISTORIAS DE USUARIO	
Número: REQ 001	Usuario: Administrador/Personal
Nombre Historia: Registro de Cliente	
Prioridad en Negocio: Alta	Riesgo en desarrollo: Alto
Iteración asignada: 1	Tiempo de desarrollo: 2 horas
Programador Responsable: Marlon Cevallos	

Descripción: Registrarse con un correo, nombre y contraseña. Independientemente se valida que la contraseña ingresada sea correcta.
Validación: Primera posibilidad: el usuario ya existe y muestre un mensaje de error, Segunda posibilidad: no se valida la contraseña ingresada. Esto se logra repitiendo la contraseña, y si coinciden, se registrará un nuevo cliente, caso contrario, se mostrará un mensaje de error.

Tabla 2: Requerimiento general, inicio de sesión

HISTORIAS DE USUARIO	
Número: REQ 002	Usuario: Administrador/Cliente
Nombre Historia: Inicio de Sesión	
Prioridad en Negocio: Alta	Riesgo en desarrollo: Alto
Iteración asignada: 1	Tiempo de desarrollo: 2 horas
Programador Responsable: Roberto Pallo y Valery Naranjo	
Descripción: Se debe ingresar el correo y la contraseña	
Validación: Mensaje con mensaje de error en caso de estar mal ingresado el usuario y contraseña. Ingreso exitoso a la página.	

Tabla 3: Requerimiento general, mostrar Menú

HISTORIAS DE USUARIO	
Número: REQ 003	Usuario: Administrador
Nombre Historia: Mostrar Menú	
Prioridad en Negocio: Alta	Riesgo en desarrollo: Alto
Iteración asignada: 1	Tiempo de desarrollo: 2 horas
Programador Responsable: Rómulo Pardo y Dylan Jiménez	
Descripción: El usuario podrá seleccionar el menú que desee	
Validación: Se debe dar click en elegir y seleccionar cantidad	

2.2 Diseño de la arquitectura del sistema utilizando el patrón MVC

2.2.1 Arquitectura cliente-servidor

Para el desarrollo del presente trabajo se ha tomado como base una arquitectura Cliente-Servidor, el cual se basa en un cliente que realiza peticiones o solicita los servicios, y un servidor que proporciona los servicios solicitados [3].

La arquitectura cliente-servidor al ser un modelo informático, posee otros sistemas conectados a través de una red donde los recursos se comparten entre las distintas computadoras. Por lo tanto, en esta arquitectura existe un servidor y múltiples clientes, en el que cada cliente viene a ser únicamente la representación de los datos, mientras que el servidor es el que realiza todo el trabajo [4].

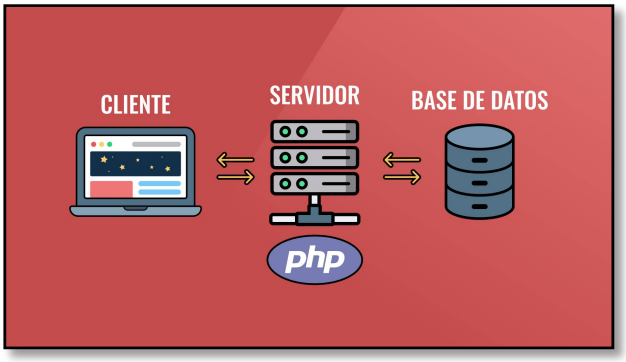

Ilustración 1.Arquitectura cliente servidor.

Tomado de Fuente: https://ed.team/blog/que-es-backend-y-frontend-guia-completa:

2.1.2 Técnicas

Basándonos en un patrón MVC, la arquitectura propuesta consta de tres capas, y tal y como se mencionó con anterioridad, el cliente será el que solicite los recursos mediante una interfaz de usuario o mediante un navegador web. Por otro lado, el servidor, el cual se comunicará directamente con el cliente, y por último una capa de datos, en la que se almacena cada uno de los registros y solicitudes del cliente, de manera corta, esta capa se encarga de los datos desde una base de datos [5].

Si se analiza de una manera más minuciosa, se pude notar que la arquitectura sigue un patrón MVC, donde el modelo es el encargado de los datos, la vista es la parte de la interfaz de usuario y el controlador, es el servidor el cual tiene comunicación tanto con la vista como el modelo.

2.2.3 Métodos

Los métodos para evaluar este tipo de arquitectura son los siguientes:

- **ALMA:** Método que se basa en el uso de escenarios de cambio, en pocas palabras describen posibles cambios que puede tener el sistema. Además, permite predecir costos de mantenimiento y posibles riesgos de la arquitectura [8].
- **PASA:** Este método de análisis de arquitectura evalúa el desempeño, por lo tanto, puede aplicarse una vez que la arquitectura ya haya sido implementada [8].
- **ATAM:** Método que permite evaluar arquitecturas de software en relación con los objetivos de atributos de calidad. [7].

- **SAAM:** Básicamente es un método que se centra en los casos de uso significativos de la arquitectura [7].

2.3.4 Metodologías

Se aplica la metodología SCRUM debido al conjunto de tareas que se debe aplicar para trabajar colaborativamente, y de forma remota permitiendo entregar resultados o prototipos de forma regular al futuro cliente o dueño de la aplicación.
Tomando en consideración la metodología prevista para el proyecto de servicio de entrega de comidas a domicilio, se ha desarrollado un cronograma con las siguientes etapas:

Tabla 4:Metodología SCRUM

Noviembre Diciembre	Inicio	Planificación	Implementación	Revisión	Lanzamiento
Semana 1					
Semana 2					
Semana 3					
Semana 4					
Semana 5					

2.4 Estándares

2.4.1. Inicio

Generar un sprint (Mini-proyecto)

⇒ Levantamiento de requisitos, requerimientos funcionales o lista de requerimientos, presupuesto a utilizar, miembros del equipo, objetivos, cronograma de entregas entre algunos procesos esenciales.

⇒ Para esta etapa nos ayudamos con las preguntas:

- ¿Qué quiero?
- ¿Cómo lo quiero?
- ¿Cuándo lo quiero

2.4.2 Planificación y Estimación

Delegar las tareas correspondientes a cada individuo, grupo o equipo de trabajo sobre qué se debe hacer, cómo se va a hacer en qué plazo y qué objetivo se va a cumplir; para esto se utilizará de un Sprint Backlog o iterador de tareas.

Algunos de los pasos a desarrollar en esta fase son:

- Crear, estimar y comprometer historias de usuario.
- Identificar y estimar tareas.
- Crear el sprint backlog o iteración de tareas.

2.4.3 Elementos, Función Individual y en Conjunto

Esta fase corresponde a realizar entregables (prototipos) los cuales van a ser revisados y en caso de ser necesario repotenciados o reformulados de acuerdo con el sprint realizado. Esta revisión se realiza con todos los miembros del equipo evitando dudas o malentendidos.

2.4.4 Revisión y retrospectiva

Esta fase corresponde a la revisión o autoevaluación del prototipo, demostración y validación del sprint y su comparación con el anterior entregable, el cual debe ser presentado con su respectivo informe de mejora o corrección. En esta fase también se mide el progreso del proyecto y el Scrum Máster (Líder) se encarga de actualizar los datos de cada Sprint (Burn Down).

2.4.5 Lanzamiento

En esta fase cumple la única tarea de enviar los entregables aprobados y poner en marcha la versión del sistema. Es preciso aclarar que en esta fase el producto ya habrá cumplido todos sus objetivos en un plazo, por lo general no superior a un mes, cumpliendo con todas las exigencias y necesidades planteadas en el caso de entregas de comidas a domicilio.

2.5. Patrón de Diseño

¿Qué es un patrón de diseño?

Un patrón de diseño es una forma reutilizable de resolver un problema común, cuya efectividad se ha comprobado al resolver problemas similares en ocasiones anteriores. Una característica fundamental es que son reusables, es decir, deben ser aplicables a diferentes problemas de diseño en diversas circunstancias. Los patrones de diseño son útiles por los siguientes motivos: [1]

- Proporcionan listas de elementos reusables en el diseño de sistemas software.
- Evitar reiteraciones en la búsqueda de soluciones a problemas conocidos y solucionados anteriormente.
- Formalizan un lenguaje común entre diseñadores.
- Estandarizar la forma en que se realiza el diseño. Además, los patrones de diseño facilitan el aprendizaje a las nuevas generaciones de diseñadores resumiendo el conocimiento ya existente.

Patrón de diseño Modelo Vista controlador (MVC)

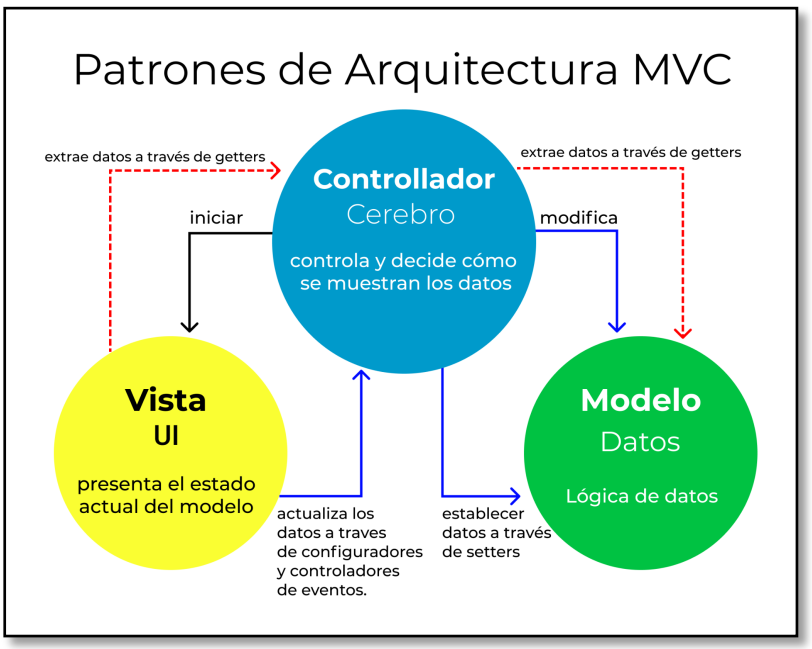

Ilustración 2. Patrón de diseño MVC

Fuente: https://www.freecodecamp.org/espanol/news/content/images/2021/06/MVC3.png

Para el caso práctico en mención, se opta por la arquitectura de diseño MVC (Modelo, Vista, Controlador) por las siguientes razones: es mucho más fácil entender y aplicar el patrón a cualquier aplicación web y la mayoría de los frameworks modernos utilizan MVC. Además, es una arquitectura muy importante, puesto que se utiliza tanto en componentes gráficos básicos hasta en grandes sistemas empresariales.

A continuación, se presenta un resumen de lo que se trata esta arquitectura de diseño:

MVC es un patrón de diseño en el cual se divide una aplicación en tres módulos claramente identificados y con funcionalidad definida: El Modelo, las Vistas y el Controlador. [2]

- El Modelo

  Es un conjunto de clases que representan información del mundo real que el sistema debe procesar, sin tomar en cuenta la forma ni los mecanismos en la que la información va a ser mostrada.

- Las Vistas

  Son el conjunto de clases encargadas de mostrar al usuario la información contenida en el modelo. Una vista está asociada a un modelo, así como también pueden existir varias vistas asociadas al mismo modelo.

- El Controlador

  Se encarga de dirigir el flujo del control de la aplicación mediante mensajes externos. Por ejemplo, datos introducidos por el usuario o también opciones del menú seleccionadas. Con base en estos mensajes, el controlador se encarga de modificar el modelo o de abrir y cerrar vistas. El controlador tiene acceso al modelo y a las vistas, sin embargo, las vistas y el modelo no conocen de la existencia de dicho controlador. [6]

Link al video de YouTube

https://youtu.be/5s6qy6aH5v4

3. Conclusiones

a) En el presente documento se abordó la temática de arquitectura de software cumpliendo con el objetivo de analizar y conocer su importancia en el desarrollo y posteriores mejoras en la infraestructura tecnológica del sistema. Algo a destacar es que es fácil de entender y lo más atractivo es la separación de componentes, permitiendo modificar sin afectar a las demás partes del sistema.
b) Se utiliza la metodología SCRUM la cual nos permite trabajar en equipo y de forma remota cada sprint (mini-proyecto) funciona como un prototipo el cual debemos entregar y mejorar en cada fase hasta llegar a concluir con una entrega final de acuerdo con las exigencias y características solicitadas por el cliente o empresa.
c) El uso de un patrón de diseño MVC, puede tomar muchas formas dependiendo el tipo de arquitectura que use la aplicación, por lo tanto, podría decirse que MVC es una arquitectura que divide su software en componentes más pequeños, permitiendo con ello tener una mayor legibilidad, además de facilitar la parte de prueba.
d) Para el diseño del sitio web a realizar, se debe tener en cuenta el código de colores, mismo que ayuda a una mejor visualización del usuario, al igual que debe ser responsive, dinámica y simplificada.

4 Bibliografía

[1] Leiva, A. (2016, marzo 8). *Patrones de diseño de software - DevExperto, por Antonio Leiva*. DevExperto, por Antonio Leiva; DevExperto. https://devexperto.com/patrones-de-diseno-software/

[2] Bascón Pantoja, E. (2004). El patrón de diseño Modelo-Vista-Controlador (MVC) y su implementación en Java Swing. *Acta Nova*, *2*(4), 493–507. http://www.scielo.org.bo/scielo.php?script=sci_arttext&pid=S1683-07892004000100005

[3] *What is Client Server Architecture?* (2022b, marzo 30). Intellipaat Blog. https://intellipaat.com/blog/what-is-client-server-architecture/

[4] *Arquitectura Cliente-Servidor*. (s. f.). https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/cliente-servidor

[5] Schiaffarino, A. (2019, 7 agosto). *Modelo cliente servidor*. Infranetworking. https://blog.infranetworking.com/modelo-cliente-servidor/

[6]Steve Burbeck. Applications Programming in Smalltalk-80(TM): How to use Model-View-Controller (MVC). http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html.

[7] *Definiciones y Conceptos sobre EvaluaciÃ3n de Arquitecturas - ` - Arquitectura de Gobierno*. (s. f.). https://centroderecursos.agesic.gub.uy/web/arquitectura-de-gobierno/arquitectura-para-tramites/-/wiki/Arquitectura+para+Tr%C3%A1mites/Definiciones+y+Conceptos+sobre+Evaluaci%C3%B3n+de+Arquitecturas/pop_up

[8] *Evaluando la arquitectura de software: Parte 2. metodos de Evaluación*. (s. f.). SG Buzz. https://sg.com.mx/content/view/217

[9] (2013). *IMPLEMENTACION DEL PATRON DE MVC PARA EL PROCESO DE SELECCIÓN DE PERSONAL.* Recuperado de https://repository.unilibre.edu.co/bitstream/handle/10901/8881/Trabajo%20de%20Grado%20Gelen%20Guzman%20y%20Natalia%20Tovar.pdf?sequence=1

[10] (2022). *Cómo aplicar la metodología Scrum y qué es el método Scrum.* Recuperado de https://www.apd.es/metodologia-scrum-que-es/

[11] Mancuso. G. (2020). *Fases de la Metodología Scrum*. Recuperado de https://blog.comparasoftware.com/fases-metodologia-scrum/

[12] *INEC*(2019). Gob.ec. Recuperado el 10 de diciembre de 2022, de https://www.ecuadorencifras.gob.ec/estadisticas/