1. **PORTADA**

**UNIVERSIDAD TÉCNICA DE AMBATO**

Facultad de Ingeniería en Sistemas, Electrónica e Industrial

“Proyecto Académico de Fin de Semestre: octubre 2021-febrero 2022”

**Título:** Sistema de Escritorio y móvil para la gestión de comida

**Carrera:** Software

**Unidad de Organización Curricular:** Nombre de la Unidad de Organización Curricular

**Línea de Investigación:**  Gestión de cambios.

**Nivel y Paralelo:** Cuarto A

**Alumnos participantes:**  Cuesta Tabares Christian Roberto

Flores Pusil Alexander Javier

Paredes Mangui Anthony Javier

**Módulo y Docente:** Manejo y Configuración del Software

Ing. Torres Valverde Leonardo David

1. **INFORME DEL PROYECTO**
   1. **Título**

Sistema de Escritorio y móvil para la gestión de comida.

* 1. **Objetivos**
     1. **Objetivo General**
* Desarrollar un sistema de escritorio y móvil para el agendamiento de pedidos de comida.
  + 1. **Objetivos Específicos.**
* Diseñar un software que use la arquitectura de capas para agregarle escalabilidad al proyecto en caso de que a futuro se lo necesite.
* Llevar una gestión del proyecto por medio de una herramienta de gestión que permita la aplicación de la metodología SCRUM.
* Usar un sistema de control de versiones que permite el trabajo en equipo para el desarrollo del proyecto.
  1. **Resumen**

Se ha decidido desarrollará una aplicación para la gestión de comida de un restaurante, en este documento se describirá los requisitos funcionales y no funcionales, las herramientas con las que se trabajó para el desarrollo, junto con las metodologías de trabajo y su flujo de trabajo dentro del desarrollo del software aplicando las herramientas para la gestión de cambios dentro de un proyecto definidos y aprendidos en el transcurso del ciclo académico.

* 1. **Palabras clave: (Producto, Metodología, Arquitectura, Entorno, Software)**
  2. **Introducción**

Este proyecto nace debido a la necesidad de mantener el control y la gestión de comidas dentro de un restaurante debido a los problemas ocurridos por el COVID-19 que han puesto en complicaciones la manera de llegar a los clientes a muchas industrias alimenticias del país, para ello se tratara gestionar los menús que ofrece un restaurante y llevarlos a los clientes por medio del internet evitando así las aglomeraciones que existen dentro de los locales comerciales, permitiendo asegurar la salud tanto de los clientes como de los empleados.

* 1. **Materiales y Metodología**

**Git**

Es un software utilizado para realizar un seguimiento de los cambios en cualquier conjunto de archivos, generalmente para coordinar el trabajo entre los desarrolladores que colaboran en el desarrollo del código fuente durante el desarrollo del software [1]. Sus objetivos incluyen velocidad, integridad de datos y soporte para flujos de trabajo distribuidos y múltiples ramas paralelas que se ejecutan en distintos sistemas [1].

**GitFlow**

Es un modelo de ramificación de Git que implica el uso de ramas secundarias y varias ramas principales con ramas más largas y compromisos más grandes [1]. En este modelo, los desarrolladores crean una rama de funcionalidades y retrasan la conexión a la rama principal hasta que se completa la función [1]. Estas ramas funcionales de larga duración requieren más cooperación para fusionarse y tienen un mayor riesgo de desviación que la rama principal, también pueden introducir actualizaciones contradictorias dentro de estas ramas [1].

Gitflow define las ramas de trabajo de la siguiente manera:

* **Main:** Esta rama tiene como fin la producción. Contiene el código del software que es presentado al cliente [1].
* **Develop:** Es la rama de desarrollo donde mayormente ocurren y se complementan todos los cambios presentes dentro del software [1].
* **Feature:** Se las usan usualmente para el desarrollo de nuevas características del software que pueden ser implementadas en una futura versión del software [1].
* **Release:** Aquí se desarrollalas correcciones de errores menores dentro del sistema, y sirven para preparar nuevas versiones del software [1].
* **Hotfix:** En esta rama se desarrolla las correcciones de bugs o errores críticos que pueden presentarse dentro del sistema [1].

**Gráfico, Diagrama, Esquemático, Gráfico de burbujas

Descripción generada automáticamente**

Imagen 1 Ejemplo de flujo de trabajo de GItFlow

**Jira**

Es un software desarrollado por Atlassian para el seguimiento de problemas, errores y la gestión de proyectos, con gran capacidad para mejorar y gestionar procesos, también para conllevar un seguimiento exhaustivo del desarrollo del software por medio de metodologías agiles y gestiones de cambios [2].

Nos brinda cuatro paquetes para el desarrollo del software que son:

**Jira Work Management:** Para la gestión genérica de proyectos [2].

**Jira Software:** Para la gestión genérica de proyectos por medio de las metodologías agiles [2].

**Jira Service Management** diseñada para el manejo por operaciones de TI o mesas de servicio empresariales [2].

**Jira Align**: Está diseñado para la gestión estratégica de productos y carteras [2].

**Arquitectura en capas**

El modelo de arquitectura en capas es un modelo de n capas en el que los componentes se apilan horizontalmente [3]. Esta es la forma tradicional de diseñar la mayoría del software autónomo [3]. Esto significa que todos los componentes están interconectados, pero no dependen unos de otros [3].

Hay cuatro capas en la arquitectura y cada capa tiene conexiones entre el módulo y los componentes que contiene [3]. De arriba a abajo son:

**Capa de presentación:** Contiene todas las categorías relacionadas con la capa de presentación y la manera en cómo se presentará el software al cliente [3].

**Capa negocio:** Contiene la lógica empresarial [3].

**Capa entidad:** Se utiliza para admitir funciones como el mapeo relacional de objetos [3].

**Capa de datos:** contienen los datos de la base de datos y como estos serán almacenados y modificados [3].

Interfaz de usuario gráfica

Descripción generada automáticamente

Imagen 2 estructura de la arquitectura en capas

**Problema**

En Ecuador debido al gran contagio por el COVID-19, muchas empresas se han visto afectadas económicamente ya que su producción y acogida en el mercado han sido reducidas drásticamente incluso a un punto de tener que parar su producción o cerrar las empresas por tiempos indefinidos, donde la industria más afectada ha sido la alimentaria por sus restricciones de aforo dentro de su contacto con el cliente, por ende, nace la necesidad de buscar una manera de que se mantenga financieramente estable y mantener la disponibilidad del producto de una manera segura a nivel de contagio tanto para los clientes como para los trabajadores y cumpliendo con las normativas de restricciones implantadas por el COE nacional.

**Alcance del Software**

El software estará disponible para dos sistemas operativos que son:

* Android
* Windows 10

En la parte de Windows contará con varios módulos para la creación y control de los pedidos de comida junto con los atributos que conforman a cada pedido de comida.

En el apartado de Android, incluirá una vista para los clientes, en el que se mostrará una visualización de los menús disponibles del día, así como la posibilidad de anunciar que quieren reservarlo en el restaurante, si es el caso.

En cuanto al software y la arquitectura estará conformado por un modelo de 5 capas que son:

* Datos
* Entidad
* Negocio
* Presentación
* Servicios

Todo esto desarrollado en C# para Windows 10, en Android por medio Kotlin y en cuanto a los servicios serán consumidos por medio de JSON.

**Entorno operativo**

|  |  |
| --- | --- |
| **Sistema operativo:** | Windows 10, Android |
| **Base de datos:** | SQL server |
| **Servidor:** | Somee |
| **Arquitectura:** | De 5 capas |
| **Lenguajes de programación:** | C#, JSON, Kotlin |
| **Tecnología de descarga y actualización:** | ClickOnce |
| **Tecnología de consulta:** | LINQ |
| **Sistema de control de versiones:** | Git |
| **Repositorio:** | GitHub |
| **Flujo de trabajo:** | GitFlow |
| **Sistema de gestión del proyecto:** | Jira |
| **Metodología de trabajo:** | SCRUM |
| **IDE de desarrollo:** | Visual Studio 2022 |
| **IDE para la gestión de la base de datos:** | Microsoft SQL Server Management Studio 18 |

**Funcionalidades del Software**

|  |  |
| --- | --- |
| **Id** | **Funcionalidad** |
| **F1** | Será desarrollado en una arquitectura de 5 capas. |
| **F2** | Se llevará una gestión del proyecto por medio de la herramienta JIRA usando la metodología SCRUM. |
| **F3** | se usará el sistema de control de versiones GIT. |
| **F4** | Para el desarrollo se aplicará el flujo de trabajo de GITFLOW. |
| **F5** | Para su base de datos se ocupará SQL server. |
| **F6** | El proyecto será almacenado en el servidor gratuito Somee. |
| **F7** | Se aplicará la tecnología ClikOnce para poder publicar y actualizar el software de escritorio. |
| **F8** | Se desarrollará un software de escritorio para el manejo de los pedidos. |
| **F9** | Se desarrollará un software móvil para la reservación de los pedidos que será usado por el cliente. |
| **F10** | Se creará una vista de escritorio para la gestión de pedidos. |
| **F11** | Se creará una vista de escritorio para la gestión de sopas. |
| **F12** | Se creará una vista de escritorio para la gestión de segundos. |
| **F13** | Se creará una vista de escritorio para la gestión de bebidas. |
| **F14** | Se creará una vista de escritorio para la gestión de postres. |
| **F15** | Se creará una vista de escritorio para la gestión de ingredientes. |
| **F16** | Se creará una vista de escritorio para la gestión de menús diarios. |
| **F18** | Se creará una vista de escritorio de bienvenida donde se muestra un registro de los menús. |
| **F18** | Se creará una vista móvil para visualizar los menús. |
| **F17** | Se creará una vista móvil para reservar pedidos. |
| **F18** | El sistema deberá tener una repuesta rápida. |
| **F19** | El sistema deberá llevar un control de los valores de entrada. |
| **F20** | El sistema deberá salvaguardar la información del cliente. |
| **F21** | El número de pedidos realizados por cada cliente no debe superar el stock |
| **F22** | Los platos y menús serán únicos y no podrán repetirse. |

**Requerimientos funcionales**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Nombre** | **Características** | **Descripción** | **Prioridad** |
| **RF01** | Vista de bienvenida | Vista para la empresa.  Vista de escritorio. | Se mostrará el registro de menús.  Se mostrarán los botones de redirección para crear platos o ver pedidos. | Baja |
| **RF02** | Agregar menú | Mostrar fechas.  Mostrar platos. | Agregar un nuevo menú con los platillos existentes dentro de la base de datos. | Alta |
| **RF03** | Eliminar menú | Eliminar menús aún vigentes. | Eliminará el menú seleccionado de la base de datos. | Alta |
| **RF04** | Mostrar secos | Registro de todos los secos. | Mostrará los segundos disponibles con sus respectivos detalles. | Alta |
| **RF05** | Crear segundo | Crear nuevo segundo con ingredientes. | Agregará un nuevo segundo con los ingredientes existentes dentro de la base de datos. | Alta |
| **RF06** | Editar segundo | Modificar el segundo existente. | Editará el segundo seleccionado y actualizar en la base de datos. | Alta |
| **RF07** | Eliminar segundo | Eliminar el segundo existente. | Eliminará el segundo seleccionado de la base de datos. | Alta |
| **RF08** | Mostrar sopa | Registro de las sopas. | Mostrará las sopas disponibles con sus respectivos detalles. | Alta |
| **RF09** | Crear sopa | Crear nueva sopa con ingredientes. | Agregará una nueva sopa con los ingredientes existentes dentro de la base de datos. | Alta |
| **RF10** | Editar sopa | Editar sopa existente. | Editará la sopa seleccionada y actualizar en la base de datos. | Alta |
| **RF11** | Eliminar sopa | Eliminar sopa existente. | Eliminará la sopa seleccionada de la base de datos. | Alta |
| **RF12** | Mostrar postre | Registro de los postres. | Mostrará los postres disponibles con sus respectivos detalles. | Alta |
| **RF13** | Crear postre | Crear nuevo postre. | Agregará un nuevo postre con los ingredientes existentes dentro de la base de datos. | Alta |
| **RF14** | Editar postre | Editar postre existente. | Editar el postre seleccionado y actualizar en la base de datos. | Alta |
| **RF15** | Eliminar postre | Eliminar postre o jugo existente. | Eliminar el postre seleccionado de la base de datos. | Alta |
| **RF16** | Mostrar bebida | Registro de la bebida. | Mostrará las bebidas disponibles con sus respectivos detalles. | Alta |
| **RF17** | Crear bebida | Crear nueva bebida con ingredientes. | Agregará una nueva bebida con los ingredientes existentes dentro de la base de datos. | Alta |
| **RF18** | Editar bebida | Editar bebida existente. | Editar la bebida seleccionado y actualizar en la base de datos. | Alta |
| **RF19** | Eliminar bebida | Eliminar bebida existente. | Eliminar la bebida seleccionado de la base de datos. | Alta |
| **RF20** | Nuevo Ingrediente | Registro de los ingredientes. | Mostrar los ingredientes disponibles con sus respectivos detalles. | Alta |
| **RF21** | Nuevo Ingrediente | Crear nuevo ingrediente con sus atributos. | Agregar un nuevo platillo con los ingredientes existentes dentro de la base de datos. | Alta |
| **RF22** | Ver pedidos | Registro de los pedidos existentes. | Mostrar la lista de pedidos realizados con sus respectivos detalles y el precio total por cada usuario. | Alta |
| **RF23** | Crear pedido | Crear nuevo pedido con sus atributos.  Vista móvil. | El cliente podrá crear un nuevo pedido de acuerdo con su necesidad. | Media |
| **RF24** | Ver menús disponibles | Mostrar menú del día.  Vista móvil. | Se visualiza el menú disponible con los platos que le conforma. | Media |

**Requerimientos no funcionales**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Nombre** | **Características** | **Descripción** | **Prioridad** |
| **RNF01** | Requerimientos de rendimiento | El sistema debe ser fluido y de carga rápida por medio del consumo de servicios, para que el usuario pueda acceder rápidamente a su cuenta y realizar sus solicitudes. El tiempo de carga debe estar entre 2 y 7 segundos. | Deberá estar optimizado, a la vez que no debe saturarlo y que existan problemas con la caché. | Alta |
| **RNF02** | Requerimientos de Protección | El sistema debe arrojar mensajes de error en caso de datos erróneos o transacciones no realizadas correctamente. | El software debe recuperar los datos y restablecerse a su estado anterior en caso de algún daño o problema externo.  Se controlará las entradas de datos. | Alta |
| **RNF03** | Requerimientos de Seguridad | La información de los clientes será manejada de manera adecuada para evitar filtrar la información. | Dentro de la parte administrativa se restringirá el acceso a la base de datos para evitar posibles ataques al software. | Alta |
| **RNF04** | Requerimientos de desarrollo | Se ocupará GIT para el control de versiones del código. | Git permitirá el trabajo en equipo y controlar las nuevas funcionalidades. | Media |
| **RNF05** | Requerimientos de desarrollo | Se ocupará JIRA para la gestión del proyecto | Permitirá llevar un control sobre cómo se va desarrollando el proyecto. | Media |
| **RNF06** | Requerimientos de desarrollo | Se ocupará el flujo de trabajo de GITFLOW para las actualizaciones de versiones del código. | Permitirá un mejor desempeño y gestión de las actualizaciones dentro del código. | Media |
| **RNF07** | Requerimientos de desarrollo | Se usará SQL server como base de datos | Permitirá un mejor funcionamiento con el lenguaje de desarrollo. | Alta |
| **RNF08** | Requerimientos de escalabilidad | Se ocupará una arquitectura de 5 capas en caso de a futuro nazca algún cambio dentro del software. | Permitirá agregar cambios al software sin afectar todo el desarrollo. | Alta |
| **RNF09** | Atributos de calidad del software | El sistema se debe adaptar a dispositivo con Windows 10 ya sea un usuario de escritorio o Android si es cliente móvil. | El sistema se construirá con el fin de que el código pueda ser reutilizable dentro del mismo sistema o para otros proyectos. | Alta |
| **RNF10** | Requerimientos de mantenibilidad. | El sistema será publicado por medio de OnceClick, para que este pueda ser lanzado por primera vez y actualizado en caso de requerir algún cambio. | Esta herramienta permite la actualización rápida del software en caso de ser necesario. | Media |
|  | Normas empresariales | Los pedidos solo se solicitan en la aplicación móvil. | los clientes están restringidos a solo hacer pedidos. | Baja |
|  | Normas empresariales | Somee será el servidor de alojamiento. | Se optará Somee por ser un servidor gratuito. | Baja |

**Módulos del software**

**Modulo bienvenida:** Este módulo contiene una vista de todos los menús existentes y los botones para redireccionar a los demás menús.

**Módulo menú:** Este módulo permite crear, editar, eliminar y visualizar menús que contenga un platillo.

**Módulo platos:** Este módulo permite crear, editar, eliminar y visualizar platillos los cuales pueden agregarse al menú del día.

**Módulo ingredientes:** Este módulo permite crear, editar, eliminar y visualizar ingredientes los cuales pueden agregarse al plato.

**Módulo pedido:** Este módulo permite registrar y visualizar los pedidos que genere el cliente.

|  |  |
| --- | --- |
| **Módulo** | **Necesidades** |
| **Bienvenida** | Conocimiento en base de datos.  Conocimiento en SQL.  Conocimiento C#  Conocimiento en Entity Framework.  Conocimiento en consumo de servicios WCF.  Conocimiento en diseño escritorio.  Conocimiento en creación de scripts. |
| **Menú** | Conocimiento en base de datos.  Conocimiento en SQL.  Conocimiento C#  Conocimiento en Entity Framework.  Conocimiento en consumo de servicios WCF.  Conocimiento en diseño escritorio.  Conocimiento en creación de scripts. |
| **Platos** | Conocimiento en base de datos.  Conocimiento en SQL.  Conocimiento C#  Conocimiento en Entity Framework.  Conocimiento en diseño escritorio.  Conocimiento en consumo de servicios WCF.  Conocimiento en creación de scripts. |
| **Ingredientes** | Conocimiento en base de datos.  Conocimiento en SQL.  Conocimiento C#  Conocimiento en Entity Framework.  Conocimiento en consumo de servicios WCF.  Conocimiento en diseño escritorio.  Conocimiento en creación de scripts. |
| **Pedidos** | Conocimiento en base de datos.  Conocimiento en SQL.  Conocimiento C#.  Conocimiento en consumo de servicios WCF.  Conocimiento en Entity Framework.  Conocimiento en diseño móvil y escritorio.  Conocimiento en creación de scripts. |

**Lista de tareas del proyecto del software**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Id** | **Requerido** | **Sprint** | **Tarea** | **Responsable** | **Días** |
| **A** | Ninguno | Diseño de los diagramas | Diseño del diagrama entidad-relación.  Diseño del diagrama de clases  Diseño de los casos de uso.  Diseño del diagrama de arquitectura.  Diseño del árbol web. | Flores Pusil Alexander Javier | 7 |
| **B** | **A** | Creación de la base de datos | implementar la base de datos en SQL. | Cuesta Tabares Christian Roberto | 7 |
| **C** | **B** | Creación de la capa entidades | Creación de la entidad segundo.  Creación de la entidad sopa.  Creación de la entidad ingrediente.  Creación de la entidad bebida.  Creación de la entidad postre.  Creación de la entidad menú.  Creación de la entidad cliente.  Creación de la entidad pedido. | Paredes Mangui Anthony Javier | 7 |
| **D** | **B, C** | Creación de la capa datos | Creación de los datos segundo.  Creación de los datos sopa.  Creación de los datos ingrediente  Creación de los datos bebida.  Creación de los datos postre.  Creación de los datos menú.  Creación de los datos cliente.  Creación de los datos pedido. | Flores Pusil Alexander Javier | 7 |
| **E** | **C, D** | Creación de la capa negocio | Creación del negocio segundo.  Creación del negocio sopa.  Creación del negocio ingrediente.  Creación del negocio bebida.  Creación del servicio postre.  Creación del negocio menú.  Creación del negocio cliente.  Creación del negocio pedido. | Cuesta Tabares Christian Roberto | 7 |
| **F** | **C, E** | Creación de la capa servicios | Creación del servicio segundo.  Creación del servicio sopa.  Creación del servicio ingrediente.  Creación del servicio bebida.  Creación del servicio postre.  Creación del servicio menú.  Creación del servicio cliente.  Creación del servicio pedido. | Paredes Mangui Anthony Javier | 7 |
| **G** | **F** | creación de la capa de presentación Windows | Creación de la vista ingredientes.  Creación de la vista platos.  Creación de la vista pedidos.  Creación de la vista menús.  Creación de la vista bienvenida. | Flores Pusil Alexander Javier | 7 |
| **H** | **F** | Creación de la capa de presentación Android | Creación de la vista de menú.  Creación de la vista de pedido. | Cuesta Tabares Christian Roberto | 7 |

* 1. **Resultados y Discusión**

El flujo de trabajo de GitFlow permite un desarrollo de software eficiente debido a su capacidad de conllevar adecuadamente los distintos cambios que pueden existir dentro del desarrollo de trabajo, nos permite evitar conflictos de código causados ​​por cada miembro del equipo de desarrollo, lo que nos ayuda a mejorar los tiempos de entrega y añadir calidad al programa.

Cuando se trata de administrar cambios a través de Jira, nos brinda un control de cambios efectivo, lo que ayuda a evitar cambios no deseados en nuestro software que pueden interrumpir el desarrollo y el propio sistema.

El desarrollo por medio de C# y la arquitectura multicapa añaden atributos de calidad al software como eficiencia y adaptabilidad a los nuevos requerimientos que puedan surgir en el futuro por parte de los clientes, permitiéndoles implementarlos de manera eficiente y sin tener que recrear el software desde cero.

* 1. **Conclusiones**
* Se pudo cumplir con los requerimientos que se han solicitado para el desarrollo del software, permitiendo la gestión de alimentos de un restaurante.
* La arquitectura podrá cumplir con una solicitud de adaptabilidad para el caso en que a futuro se requiera una mejora o implementar alguna nueva funcionalidad dentro del software.

El manejo del flujo de trabajo de GitFlow permitió llevar un desarrollo de la aplicación eficiente y omitiendo el riesgo de perdidas o conflictos entre el código desarrollado por cada miembro del equipo.

* La gestión de los cambios por medio de Jira pudo evitar que los cambios no deseados salieran a flote, evitando complicaciones en el desarrollo de software.
  1. **Referencias bibliográficas**

|  |  |
| --- | --- |
| [1] | S. Chacon y B. Straub, «Pro Git book,» 2014. [En línea]. Available: https://git-scm.com/book/en/v2. |
| [2] | J. Fisher, D. Koning y A. P. Ludwigsen, «Utilizing Atlassian Jira For Large-Scale Software Development Management,» 24 Septiembre 2013. [En línea]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1090.9400&rep=rep1&type=pdf. |
| [3] | M. Richards, «Software Architecture Patterns,» 11 Mayo 2015. [En línea]. Available: http://sddconf.com/brands/sdd/library/Architecture\_Patterns.pdf. |

* 1. **Fotografías y gráficos**

**Diagrama entidad-relación**

Una captura de pantalla de un celular con texto e imágenes

Descripción generada automáticamente con confianza media

**Diagrama de clases**

Interfaz de usuario gráfica

Descripción generada automáticamente

**Diagramas de casos de uso**

Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente

**Diagrama de arquitectura**

Diagrama

Descripción generada automáticamente

**Árbol web del sistema de escritorio.**

**Árbol Web del sistema móvil**

**Flujo de trabajo con cambios desde GitHub**

**Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente**