**Proyecto 3 Diseño**

Natali Andrea Bohórquez Jiménez – 202214826 - n.bohorquezj

Mateo Calderón Rincón - 202213589 - m.calderonr

Juan David García Rojas - 202213434- jd.garciar12

**Ventana Administrador**

Para crear la ventana del administrador necesitábamos primero que todo entender cuales eran las actividades que este realizaba y que cambios debíamos hacer en comparación al proyecto anterior, para ello era necesario que el administrador pueda desempeñar todas sus funciones de la forma más práctica posible. Pero además de los cambios mencionados en el enunciado de este nuevo proyecto, realizamos otras modificaciones que eran necesarias para el correcto funcionamiento del programa y para que este cumpliera con todo lo necesario. En el documento de diseño entregado anteriormente se mencionaba que por temas de practicidad solo era posible escoger un adicional al momento de crear las habitaciones, pero con las modificaciones necesarias para este proyecto, esa idea quedaba obsoleta, por lo que decidimos cambiar la forma en como estos se presentaban, ya que anteriormente los adicionales estaban en una lista plegable, en cambio como ahora los adicionales son muchos más, decidimos que estos se le mostraran al administrador en formato de CheckBox, que hacia que se le pudieran agregar cualquiera sin ningún tipo de límite, lo que cumple a cabalidad lo que menciona el enunciado del proyecto. Otro cambio importante que se realizo es sobre el tema de las camas, ya que en el proyecto anterior también se presentaban con listas plegables, en cambio ahora el administrador puede agregar las camas a su elección, aunque allí se tomó una decisión para que fuera lo mas realista, ya que una habitación no puede tener camas un numero exorbitante de camas, ya que no cabrían, por lo que decidimos que cada cuarto tenga un máximo de tres camas sencillas y dos dobles. Adicionalmente, se agrego un contador para la capacidad de las habitaciones teniendo en cuenta el número de camas, por lo que en cada cama sencilla cabe una persona y cada cama doble, dos personas; entonces dependiendo del número y del tipo de camas que se agreguen será la capacidad de la habitación.

Como estos cambios a la hora de crear las habitaciones son significativos, decidimos cambiar el panel que se encargaba de esto, ya que antiguamente se encontraba todo, junto al botón de cargar un archivo para la creación de habitaciones; ahora, se tiene un botón para cargar el archivo (que ya estaba) y un botón para conducir a una nueva ventana que permite crear las habitaciones con todos los elementos e información necesaria para este proceso.

Una decisión que tomamos fue que cada persona sin importar su edad ocupa el mismo espacio, por lo que una cama sencilla puede ser ocupada por un niño de 7 años o un adulto de 59 años, es el mismo caso para las camas dobles. También decidimos que el precio que se le da a cada habitación, es un precio total, es decir el valor que se le asigna al momento de crear la habitación, es un precio para la habitación por día, no importa su capacidad o cuantas personas la ocupen, aunque si es cierto que una habitación con una cama doble, sin adicionales, será mas económica, que una habitación con capacidad para 5 personas con balcón y televisión, pero el precio que se le asigna será responsabilidad y los criterios manejados por los del hotel.

De igual forma, decidimos que las medidas de las habitaciones y de las camas serán estándar, es decir, todas las habitaciones dependiendo el tipo tiene una medida distinta (pero igual con las de su mismo tipo) y lo mismo ocurre con las camas.

Se decidió que el formato que siga el csv de las habitaciones creadas sea: id habitación, tipo habitación, # de camas sencillas, # de camas dobles, capacidad, precio por noche de la habitación, adicionales. Que se ve así:



Por su parte, el panel de creación de habitaciones luce así:



En donde allí se pueden apreciar las medidas definidas de las habitaciones según su tipo, al igual que de las camas.

Cabe mencionar que al igual que con el proyecto pasado la clase principal del Administrador es VentanaAdministrador y que desde ella se corren las diferentes funcionalidades.

Algo importante de mencionar también, es que, al momento de ejecutar el programa en otra máquina, se pueden presentar errores para guardar los datos en los csv, ya que la ubicación de los archivos cambia, pero arreglando esa línea de código debería funcionar.

**Ventana Empleado**

Pagos

Para crear la ventana del empleado necesitábamos primero que todo entender cuales eran las actividades que este realizaba y que cambios debíamos hacer en comparación al proyecto anterior, para ello era necesario que el empleado pueda generar una factura de acuerdo a lo que el cliente consumiera o el servicio que utilizara, generar el cobro y factura de una forma adecuada. Pero además de los cambios mencionados en el enunciado de este nuevo proyecto, realizamos otras modificaciones que eran necesarias para el correcto funcionamiento del programa y para que este cumpliera con todo lo necesario. Nos centramos en desarrollar un sistema de pagos que permitiera a los usuarios realizar transacciones con tarjetas de crédito. A continuación, se presentan algunas de las decisiones clave que tomamos:

Clase CreditCard: Decidimos crear la clase CreditCard para encapsular los datos de una tarjeta de crédito y manejar las validaciones necesarias. Esta decisión resultó acertada, ya que nos permitió tener un código más limpio y modular, evitando la dispersión de la lógica de validación en diferentes partes del sistema.

Clase PaymentGateway: Implementamos la clase PaymentGateway como una pasarela de pago que encapsula la lógica de procesamiento de pagos. Esta decisión nos permitió tener un código más estructurado y reutilizable, ya que centralizamos la lógica de pagos en un solo lugar. Además, facilitó la realización de pruebas y el mantenimiento del código relacionado con los pagos.

Clase PaymentGUI: Diseñamos la clase PaymentGUI para la interfaz gráfica de usuario que permite ingresar los datos de la tarjeta y realizar el pago. Esta decisión mejoró la usabilidad de la aplicación, ya que proporcionó una experiencia más intuitiva al usuario. Separar la interfaz de usuario del código de procesamiento de pagos también facilitó los cambios y el mantenimiento en el futuro.

Generar Reportes

Nos enfocamos en desarrollar un sistema de generación de reportes que mostrara información sobre las ventas del restaurante del hotel. A continuación, se presentan algunas de las decisiones clave que tomamos:

Clase ReportGenerator: Implementamos la clase ReportGenerator para generar reportes con información sobre las ventas del restaurante del hotel. Esta decisión nos permitió separar la lógica de generación de reportes de otras partes del sistema, lo que mejoró la modularidad y facilitó la generación de diferentes tipos de reportes en el futuro.

Clase ReportGUI: Diseñamos la clase ReportGUI para la interfaz gráfica de usuario que muestra los reportes y las gráficas generadas. Esta decisión mejoró la presentación visual de los datos y facilitó la comprensión de la información por parte del usuario. También promovió la reutilización de código y mejoró la legibilidad y mantenibilidad del código.