UNIVERSIDAD SAN CARLOS DE GUATEMALA -USAC

CENTRO UNIVERSITARIO DE OCCIDENTE -CUNOC-

DIVISIÓN DE CIENCIAS DE INGENIERÍA

INGENIERÍA EN CIENCIAS Y SISTEMAS

Portal Financiero (Tarjeta de crédito)- Manual técnico.

Seminario De Sistemas 1

Catedrático: Ing. Pedro Domingo

Alumnos :

Mario Moisés Ramírez Tobar 201830007

Luis Estuardo Bolaños Gonzalez 201731766

Celia Esmeralda Vargas Lopez 20173093

Guatemala, 05 de noviembre 2021

ÍNDICE

INTRODUCCIÓN

El sistema compras por internet está conformado por cuatro módulos, uno de ellos es el portal financiero en la sección tarjeta de crédito, este módulo se desplegará tanto para un usuario administrador como para un usuario clásico, a continuación se describirán las tecnologías usadas para el desarrollo de este subsistema , la base de datos y código fuente.

OBJETIVOS

Objetivo General

- El portal financiero de tarjeta de crédito es el encargado de mantener las cuentas de tarjeta de crédito de los clientes.

Objetivos Específicos

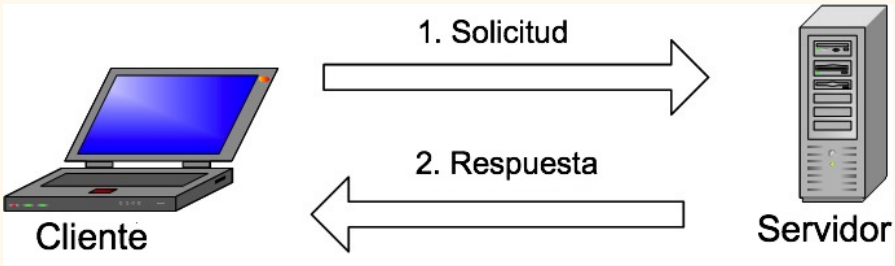
- Desplegar vista para usuario administrador y usuario clasico
- Facilidad de uso
- Despliegue de reportes

TECNOLOGÍAS USADAS

MYSQL:

En cuanto a la definición general, MySQL es un sistema de gestión de bases de datos relacionales de código abierto (RDBMS, por sus siglas en inglés) con un modelo cliente-servidor. RDBMS es un software o servicio utilizado para crear y administrar bases de datos basadas en un modelo relacional.

¿Cómo funciona MySQL?



La imagen explica la estructura básica cliente-servidor. Uno o más dispositivos (clientes) se conectan a un servidor a través de una red específica. Cada cliente puede realizar una solicitud desde la interfaz gráfica de usuario (GUI) en sus pantallas, y el servidor producirá el output deseado, siempre que ambas partes entiendan la instrucción. Sin meternos demasiado a fondo en temas técnicos, los procesos principales que tienen lugar en un entorno MySQL son los mismos, y son:

- MySQL crea una base de datos para almacenar y manipular datos, definiendo la relación de cada tabla.
- Los clientes pueden realizar solicitudes escribiendo instrucciones SQL específicas en MySQL.
- La aplicación del servidor responderá con la información solicitada y esta aparecerá frente a los clientes.

Versión:

NodeJs:

Es un entorno de tiempo de ejecución de JavaScript (de ahí su terminación en .js haciendo alusión al lenguaje JavaScript). Este entorno de tiempo de ejecución en tiempo real incluye todo lo que se necesita para ejecutar un programa escrito en JavaScript. También aporta muchos beneficios y soluciona muchísimos problemas.

Node.js utiliza un modelo de entrada y salida sin bloqueo controlado por eventos que lo hace ligero y eficiente (con entrada nos referimos a solicitudes y con salida a respuestas). Puede referirse a cualquier operación, desde leer o escribir archivos de cualquier tipo hasta hacer una solicitud HTTP.

La idea principal de Node.js es usar el modelo de entrada y salida sin bloqueo y controlado por eventos para seguir siendo liviano y eficiente frente a las aplicaciones en tiempo real de uso de datos que se ejecutan en los dispositivos. Es una plataforma que

no dominará el mundo del desarrollo web pero si que satisface las necesidades de una gran mayoría de programadores.

La finalidad de Node.js no tiene su objetivo en operaciones intensivas del procesador, de hecho, usarlo para programación de más peso eliminará casi todas sus ventajas. Donde Node.js realmente brilla es en la creación de aplicaciones de red rápidas, ya que es capaz de manejar una gran cantidad de conexiones simultáneas con un alto nivel de rendimiento, lo que equivale a una alta escalabilidad.

Github:

GitHub es una red social de desarrolladores. La primera parte del nombre, «Git», se debe al uso del sistema de control de versiones y la segunda parte, «Hub», tiene que ver con la conexión entre los profesionales de programación de cualquier parte del mundo.

Además, GitHub es una de las plataformas de trabajo colaborativo en línea más grandes del mundo.

Aquí los usuarios comparten sus proyectos, y las personas de cualquier parte del mundo pueden trabajar en paralelo en ellos.

HEROKU:

Heroku es una solución de PaaS (plataforma como servicio, por sus siglas en inglés) en la nube y basada en contenedores. Los desarrolladores utilizan Heroku para

implementar, administrar y escalar las aplicaciones modernas. La plataforma es elegante, flexible y fácil de usar, además de ofrecer el camino más simple para que las aplicaciones salgan al mercado rápidamente. Heroku es una solución totalmente administrada, lo que brinda a los desarrolladores la libertad de enfocarse en el producto principal sin la carga de tener que mantener la infraestructura de la aplicación. Las herramientas, los servicios y los flujos de trabajo integrados están diseñados meticulosamente para ayudar a maximizar la productividad individual y del equipo

CLEVER CLOUD:

Clever Cloud hace que sea realmente fácil de implementar, ejecutar y escalar aplicaciones MySQL en una plataforma de nube de alta disponibilidad como servicio.

Clever Cloud ofrece una plataforma de automatización de TI para desarrolladores con infraestructura a prueba de balas y soporte completo. La solución proporciona colaboración basada en roles, herramientas de línea de comandos, implementaciones y lanzamientos fáciles, reparación automática y monitoreo activo. Permite a los usuarios crear una base de datos en segundos y viene con copias de seguridad automáticas y un amplio mercado.

El servicio ofrece escalado de aplicaciones en función de su uso, ya sea por tamaño o instancias, y proporciona límites de escalado personalizados. Ofrece a los usuarios un aislamiento completo de la máquina virtual y mantiene las VM actualizadas en lo que respecta a la seguridad. La solución mantiene todo monitoreado y analizado automáticamente, desde el hardware hasta las aplicaciones, y ofrece métricas sobre las aplicaciones en tiempo real.

Clever Cloud tiene una característica sorprendente de recuperación automática que vuelve a implementar o reinicia la aplicación si es demasiado lenta o falla. Los desarrolladores pueden acceder fácilmente a sus registros para corregir sus errores y pueden compartirlos con otras personas. Viene con la integración de Slack para permitir a los usuarios obtener información sobre sus problemas de implementación.

BASE DE DATOS:

Como requisito se solicitó que la base de datos fuera relacional, para el módulo de tarjeta de crédito se ha utilizado como gestor de base de datos MYSQL.

A continuación se presenta el script para llenar los datos de la base de datos y también el diagrama de la base de datos.

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
-- -----------------------------------------------------
-- Schema tarjeta_credito
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema tarjeta_credito
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `tarjeta_credito` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci ;
USE `tarjeta_credito` ;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`comentario`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`comentario` (
  `comentario_id` INT NOT NULL,
  `comentario` VARCHAR(45) NULL DEFAULT NULL,
  `usuario_id` INT NULL DEFAULT NULL,
  PRIMARY KEY (`comentario_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`deshabilitacion`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`deshabilitacion` (
  `id` INT NOT NULL,
  `motivo` VARCHAR(45) NULL DEFAULT NULL,
  `tarjeta_id` INT NULL DEFAULT NULL,
  `fecha` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`eliminacion`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`eliminacion` (
  `id` INT NOT NULL,
```

```
  `motivo` VARCHAR(45) NULL DEFAULT NULL,
  `tarjeta_id` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`tipo_cuenta`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`tipo_cuenta` (
  `tipo_cuenta_id` INT NOT NULL,
  `nombre` VARCHAR(45) NULL DEFAULT NULL,
  `limite_default` DOUBLE NULL DEFAULT NULL,
  `moneda_id` INT NULL DEFAULT NULL,
  PRIMARY KEY (`tipo_cuenta_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`moneda`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`moneda` (
  `moneda_id` INT NOT NULL,
  `nombre` VARCHAR(45) NULL DEFAULT NULL,
  `cambio_quetzales` DOUBLE NULL DEFAULT NULL,
  `simbolo` VARCHAR(45) NULL DEFAULT NULL,
  INDEX `fk_moneda_tipo_cuenta1_idx` (`moneda_id` ASC) VISIBLE,
  PRIMARY KEY (`moneda_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`transaccion`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`transaccion` (
  `transaccion_id` INT NOT NULL,
  `monto` DOUBLE NULL DEFAULT NULL,
  `es_Aumento` TINYINT(1) NULL DEFAULT NULL,
  `tarjeta_id` INT NULL DEFAULT NULL,
  `fecha` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`transaccion_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;

-- -----------------------------------------------------
-- Table `tarjeta_credito`.`tarjeta`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`tarjeta` (
  `tarjeta_id` INT NOT NULL,
  `numero_tarjeta` INT NULL DEFAULT NULL,
  `cvv` INT NULL DEFAULT NULL,
  `fecha_vencimiento` DATE NULL DEFAULT NULL,
  `notifyme` TINYINT(1) NULL DEFAULT NULL,
  `limite` DOUBLE NULL DEFAULT NULL,
  `saldo` DOUBLE NULL DEFAULT NULL,
  `usuario_id` INT NULL DEFAULT NULL,
  `tipo_cuenta_id` INT NULL DEFAULT NULL,
  INDEX `fk_tarjeta_transaccion_idx` (`tarjeta_id` ASC) VISIBLE,
  INDEX `fk_tarjeta_tipo_cuenta1_idx` (`tipo_cuenta_id` ASC) VISIBLE,
  PRIMARY KEY (`tarjeta_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
```

```
-- -----------------------------------------------------
-- Table `tarjeta_credito`.`usuario`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `tarjeta_credito`.`usuario` (
  `usuario_id` INT NOT NULL,
  `dpi` INT NULL DEFAULT NULL,
  `nombre` VARCHAR(45) NULL DEFAULT NULL,
  `username` VARCHAR(45) NULL DEFAULT NULL,
  `correo` VARCHAR(100) NULL DEFAULT NULL,
  `password` VARCHAR(45) NULL DEFAULT NULL,
  `admin` TINYINT(1) NULL DEFAULT NULL,
  INDEX `fk_usuario_tarjeta1_idx` (`usuario_id` ASC) VISIBLE,
  PRIMARY KEY (`usuario_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

CÓDIGO FUENTE:

Se ha utilizado NodeJs para el desarrollo de la aplicación web.

Link del codigo fuente en github:

https://github.com/LouisGonzalez/Tarjeta-Credito.git