

**GUIA PROYECTO FINAL LABORATORIO IV**

**“EL BUEN SABOR”**

**ALUMNOS:**

**JUAN ORTIZ.**

**FEDERICO SABATINI.**

**DOCENTE:**

**GERARDO MAGNI.**

**INTRODUCCION:**

**En la siguiente guía se detallará paso a paso el desarrollo del proyecto final para la materia Laboratorio IV titulado “El Buen Sabor”, donde aplicando los conocimientos adquiridos en el 2° semestre de la Tecnicatura los alumnos deben desarrollar una aplicación WEB para un negocio de comidas rápidas, respetando las consignas asignadas por el Docente de la catedra Gerardo Magni, se podrá seleccionar el lenguaje de Backend y Framework de FrontEnd, así como los IDE necesarios para la aplicación de los mismos.**

**Repositorio GitHub Proyecto Final =>** [**https://github.com/FedeConciencia/ProyectoFinal\_LaboratorioIV**](https://github.com/FedeConciencia/ProyectoFinal_LaboratorioIV)

**PRIMER PASO => CREACION DE LA BASE DE DATOS.**

**Seleccionamos MySQL como Gestor de Base de Datos para poder interactuar con la Base de Datos Relacional a través del lenguaje (SQL).**

**Gestionamos la normalización de las entidades => Se aplica una serie de Reglas al Diseño de la Base de Datos (Formas Normales) el objetivo del mismo es evitar la redundancia de datos innecesarios protegiendo la integridad de los mismos y evitando inconsistencias en la BD (Facilitando las Operaciones posteriores). 1FN – 2FN – 3FN**

**Primera Forma Normal (1FN) o Primera Regla:**

**Cada columna (Atributo) campo puede contener un único valor.**

**No pueden ser valores null (nulos).**

**No deben existir campos repetidos, grupos ordenados.**

**La tabla (Entidad) debe contener un único campo PK.**

**Segunda Forma Normal (2FN) o Segunda Regla:**

**Debe cumplir sin excepción la 1FN.**

**Cualquier campo (campo) no clave debe ser dependiente de la clave PK.**

**Si se utilizan en el diseño de la BD tablas (Entidad) con PK simples, se puede saltar esta regla o forma.**

**Tercera Forma Normal (3FN) o Tercera Regla:**

**Debe cumplir sin excepción la 2FN.**

**Ningún campo (columna) no clave, dependerá de ningún otro campo no clave.**

**Cuando el Diseño de la Base de Datos da cumplimiento de forma correcta a la 3FN (Tercer Forma Normal), podemos garantizar que estarán libres de anomalías, evitando repetición e inconsistencia de datos.**

**¿Qué es la Clave Primaria?**

**Es un campo único dentro de la tabla (Entidad) que no se puede repetir es único, permite identificar cada registro de la tabla como único, facilita la relación entre tablas, solo puede existir una sola llave primaria, se identifica dentro del modelo como PK.**

**¿Qué es la Clave Foránea?**

**Es un campo que debe llevar el mismo nombre de la clave Primaria PK de la tabla que establece la relación, se identifica como FK, debe ser del mismo tipo, permite establecer la relación entre las tablas. La llave FK de una tabla (Entidad) es la llave primaria PK en la otra tabla (Entidad) de la relación.**

**¿Qué es la Clave Candidata?**

**Es otro campo de la tabla (Entidad) que también permite identificar un registro (fila) como único ej (DNI-CUIT) una tabla puede tener + de una clave candidata. Se presenta como unique.**

**¿Qué Tipos de Relaciones Existen entre tablas en el Modelo Relacional?**

**Para que los datos de las tablas se puedan seleccionar, encontrar y procesar de forma rápida y eficaz, es necesario que las tablas se encuentren relacionadas unas con otras.**

**Relación Uno a Uno (1:1):**

**Se da cuando un registro o fila de una tabla (Entidad) esta relacionado con otro único registro de la otra tabla (Entidad) y a su vez este último se relaciona con un solo registro de la primera tabla (Entidad).**

**Ej: Un Producto solo puede tener un Código de Barra, como un Código de Barra solo puede tener un producto.**

**Importante => En la relación uno a uno la clave FK se puede ubicar en cualquiera de las 2 tablas (Entidades).**

**Relación Uno a Muchos (1: N):**

**A cada registro de la tabla (Entidad A) se le puede asociar varios registros de la tabla (Entidad B) y cada registro de la tabla (Entidad B) se le puede asociar con solo un registro de la tabla (Entidad A)**

**Ej: Una categoría puede estar relaciona con más de un Producto y este Producto puede solo tener una Categoría.**

**Importante => En las relaciones 1 a N la clave FK se debe ubicar en la tabla (Entidad) N muchos.**

**Relación Muchos a Muchos (N:M):**

**A cada registro de la tabla (Entidad A) se le puede asociar varios registros de la tabla (Entidad B) y cada registro de la tabla (Entidad B) se le puede asociar con varios registros de la tabla (Entidad A)**

**Ej: Un Producto puede ser traído por muchos Proveedores y un Proveedor puede traer muchos Productos.**

**Importante => Cuando existe una relación N:M se debe crear una tercera tabla UNION\_PRODUCTO\_PROVEEDOR que contenga el id\_Union (PK), id\_Producto (FK), id\_Proveedor (FK).**

**A continuación, Detallamos el Modelo Entidad Relación de la Base de Datos con sus Entidades (Tablas-Objetos), Campos (Atributos), Clave Primaria (PK), Clave Foránea (FK), Clave Candidata (Unique).**

**Modelo Entidad Relación Proyecto Final**

**Configuración del Modelo Entidades (Tablas) y Atributos (Campos):**

**Referencias:**

**COLOR: Entidad**

**COLOR: Clave Primaria PK**

**COLOR: Clave Foránea FK**

**COLOR: Clave Candidata**

**Cliente: idCliente (INT-PK Primary Key-not null- autoIncrement),** **nombre (varchar-not null), apellido (varchar-not null), dni (varchar-not null- unique-Clave Candidata), fechaNacimiento (date-not null), telefono (varchar-not null), email (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’).**

**Usuario: idUsuario (INT-PK Primary Key-not null- autoIncrement), usuario (varchar-not null- unique-Clave Candidata), contraseña (varchar-not null), rol (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), idCliente (INT – FK Foreing Key-not null).**

**Domicilio: idDomicilio (INT-PK Primary Key-not null- autoIncrement),**

**calle (varchar-not null), numero (varchar-not null), localidad (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), idCliente (INT – FK Foreing Key-not null).**

**Pedido: idPedido (INT-PK Primary Key-not null- autoIncrement), codigo (INT-not null- unique-Clave Candidata), horaEstimadaFin (date-not null), tipoEnvio (INT-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), idCliente (INT – FK Foreing Key-not null), idDomicilio (INT – FK Foreing Key-not null).**

**#SE CREAL LA TABLA UNION (DETALLE\_PEDIDO) DE LA RELACION N:M ENTRE ENTIDAD ARTICULO\_MANUFACTURADO Y ENTIDAD PEDIDO.**

**DetallePedido: idDetallePedido (INT-PK Primary Key-not null- autoIncrement), cantidad (INT- not null), subTotal(double-not null), idPedido (INT – FK Foreing Key-not null), idArticuloManufacturado (INT – FK Foreing Key-not null).**

**Factura: idFactura (INT-PK Primary Key-not null- autoIncrement), codigo (varchar-not null- unique-Clave Candidata), montoDescuento (double-not null), formaPago (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), idPedido (INT – FK Foreing Key-not null).**

**#SE CREAL LA TABLA UNION (DETALLE\_FACTURA) DE LA RELACION N:M ENTRE ENTIDAD ARTICULO\_MANUFACTURADO Y ENTIDAD FACTURA:**

**DetalleFactura: idDetalleFactura (INT-PK Primary Key-not null- autoIncrement), cantidad (INT- not null), subTotal(double-not null), idFactura (INT – FK Foreing Key-not null), idArticuloManufacturado (INT – FK Foreing Key-not null).**

**MercadoPago: idMercado (INT-PK Primary Key-not null, autoIncrement), codigo (varchar-not null- unique-Clave Candidata), fechaAprobacion (date-not null), nroTarjeta (varchar-not null), metodoPago(varchar, not null), estadoPago(varchar, not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), idPedido (INT – FK Foreing Key-not null).**

**ArticuloManufacturado: idArtManufacturado (INT-PK Primary Key-not null, autoIncrement) tiempoEstimado(INT – not null), denominacion (varchar-not null), precioVenta(double-not null), imagen (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’) idRubroGeneral (INT – FK Foreing Key-not null).**

**#SE CREAL LA TABLA UNION (ARTICULO\_MANUFACTURADO\_DETALLE) DE LA RELACION N:M ENTRE ENTIDAD ARTICULO\_MANUFACTURADO Y ENTIDAD ARTICULO\_INSUMO:**

**ArticuloManufacturadoDetalle: idArtManufacturadoDetalle (INT-PK Primary Key-not null), cantidad(double-not null), unidadMedida (varchar-not null), idArtManufacturado (INT – FK Foreing Key-not null), idArtInsumo (INT – FK Foreing Key-not null).**

**RubroGeneral: idRubroGeneral (INT-PK Primary Key-not null, autoIncrement), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), denominacion (varchar-not null).**

**ArticuloInsumo: idArtInsumo (INT-PK Primary Key-not null, autoIncrement), denominacion (varchar-not null), precioCompra(double-not null), precioVenta(double-not null), stockActual(double-not null), stockMinimo(double-not null), unidadMedida (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’), idRubroArticulo (INT – FK Foreing Key-not null).**

**RubroArticulo: idRubroArticulo (INT-PK Primary Key-not null, autoIncrement), denominacion (varchar-not null), fechaAlta (date-not null), fechaBaja (date-null- default ‘0000-00-00’), estado (varchar-null, default ‘activo’).**

**SEGUNDO PASO => SELECCIONAMOS EL LENGUAJE E IDE PARA EL BACKEND.**

**Trabajamos con lenguaje JAVA e IDE NetBeans 8.2 donde creamos un proyecto Web (New Proyect => Java Web => Web Application ) se debe tener configurado el servidor Apache TomCat.**

**Instalamos las librerías necesarias para el proyecto =>**

* **Mysql-connector-java.5.1.49.jar**
* **Gson.2.8.2.jar**
* **Apache-cammons.jar**
* **Etc**

**Creamos en el Source Packages => Paquetes y Clases Necesarias**

* **Conexión => Establece conexión con la BD**
* **Controlador => Métodos CRUD de las entidades**
* **Modelo => atributos y métodos de las entidades**
* **Servlet => establece conexión con el FrontEnd**
* **Principal (Test) => testeo de los métodos y funcionalidad**

**TERCER PASO => SELECCIONAMOS EL FRAMEWORK E IDE O EDITOR PARA EL FRONTEND.**

**Trabajamos con el FrameWork React.JS y el editor Open Source de Visual Code donde creamos un proyecto React.**

**Creación del Proyecto en React =>**

**Estos comandos se pueden ingresar desde la consola de windows o desde la consola de Visual Code (Recomendado desde la consola de Windows):**

**Desde la carpeta ProyectosReact =>create-react-app app-react-proyectoFinal**

**npm install --save react-router-dom => este paquete/libreria permite crear rutas, vinculos**

**npm install react-bootstrap bootstrap => este paquete/libreria corresponde a bootstrap**

**En la index.js => de esta forma podemos usar Bootstrap en la aplicación, gestionando la importación.**

**import 'bootstrap/dist/css/bootstrap.min.css';**

**npm install axios => librería que permite la comunicación cliente-servidor. "axios": "^0.21.1", se verifica desde el package.json**

**npm install fetch => librería que permite la comunicación cliente-servidor. "fetch": "^1.1.0",**

**Ejecutar Proyecto => npm start => http://localhost:3000/**

**En la carpeta src =>**

* **Creamos la carpeta css (archivos de css)**
* **Creamos la carpeta images (para las imágenes)**
* **Creamos la carpeta components (donde creamos los componentes del proyecto (vistas, menú, tarjetas, etc))**

**TERCER PASO => CREAMOS EL ADMINISTRADOR PARA TODAS LAS ENTIDADES (TABLAS) BD – MODELOS (BACKEND) DE NUESTRO PROYECTO.**

**ENTIDADES =>**

**ADMIN.CLIENTE**

**ADMIN.DOMICILIO**

**ADMIN.USUARIO**

**ADMIN.PEDIDO**

**ADMIN.FACTURA**

**ADMIN.MERCADO\_PAGO**

**ADMIN.DETALLE\_FACTURA**

**ADMIN.DETALLE\_PEDIDO**

**ADMIN.ART\_MANUFACTURADO**

**ADMIN.ART\_MANUF\_DETALLE**

**ADMIN.RUBRO\_GENERAL**

**ADMIN.RUBRO\_ARTICULO**

**ADMIN.ART\_INSUMO**

**ADMIN.CONFIGURACION**

**Se aplican las funciones principales CRUD para interactuar con las entidades (Tablas) en la Base de Datos:**

**CREATE => (INSERTAR)**

**READ => (LEER)**

**UPDATE => (ACTUALIZAR)**

**DELETE LOGIC => (ELIMINACION LOGICA DE DATOS, ACTUALIZACION).**

**CUARTO PASO => SE IMPLEMENTA LA ENCRIPTACION MD5 para la variable contraseña de la ENTIDAD Usuario.**

**Importamos las Librerías al Proyecto =>**

**commons-codec-1.6.jar**

**commons-codec-1.6-javadoc.jar**

**commons-codec-1.6-sources.jar**

**commons-codec-1.6-tests.jar**

**Se gestiona desde el BackEnd => Proyecto Web Java**

**Creamos la clase Encooder con los métodos =>**

**Encoode => gestiona la encriptación MD5 de la contraseña ingresada a través del Formulario React del FrontEnd y la clase ServletUsuario (Insertar).**

**Deecnode => desencripta la contraseña obtenida desde la consulta a la BD entidad Usuario en la clase ServletUsuario consultar oneUsuario o allUsuario.**

**QUINTO PASO => SE IMPLEMENTA EL LOGUIN Y REGISTRO DE USUARIO/CLIENTE EN NUESTRA APLICACIÓN.**

**Descargamos el paquete instalador de API Loguin Google.**

**Link =>** [**https://www.npmjs.com/package/react-google-login**](https://www.npmjs.com/package/react-google-login)

**Se gestiona desde la terminal (ruta proyecto) la instalación =>**

**npm i react-google-login**

**Ahora debemos ir al siguiente Link de Google para crear un Id de Cliente, previamente logueado con cuenta en Gmail =>**

[**https://console.cloud.google.com/projectselector2/apis/dashboard?hl=ES&pli=1&supportedpurview=project&project=&folder=&organizationId=**](https://console.cloud.google.com/projectselector2/apis/dashboard?hl=ES&pli=1&supportedpurview=project&project=&folder=&organizationId=)

**Crear Id de Cliente => Tutorial => Nuevo Proyecto => Nombramos y Creamos => Ver => Inicio => Credenciales => Crear Credenciales => Id de Cliente de Oauth => Externos => Crear**

**Genera el Id de Cliente =>**

**190721094702-ifpj8nj34rqig6799jeqjrjolpanmssa.apps.googleusercontent.com**

**Procedemos a incorporar el código de la Api Loguin de Google dentro de nuestro Proyecto React Componente Loguin.js =>**

**Importamos =>**

**import GoogleLogin from 'react-google-login';**

 //Creamos la funcion de respuesta del loguin de google:

    const responseGoogle = (response) => {

        console.log(response);

    }

 <GoogleLogin

clientId="190721094702-ifpj8nj34rqig6799jeqjrjolpanmssa.apps.googleusercontent.com"

render={renderProps => (

<Button onClick={renderProps.onClick} disabled={renderProps.disabled} variant="primary"  size="lg">LOGUIN GOOGLE ACCOUNT</Button>

)}

  onSuccess={responseGoogle}

  onFailure={responseGoogle}

  isSignedIn = { true }

  cookiePolicy={'single\_host\_origin'}

/>