Manual Técnico – Revistas Actuales

Realizado por: Javier Alejandro Mérida Gómez - 202230368

Descripción del Software:

¡Bienvenido al "Revistas Actuales"!

La solicitud del cliente es desarrollar un sitio web dedicado a la publicación de revistas digitales en formato PDF, donde los editores puedan publicar sus revistas en diversas categorías y los usuarios puedan acceder, leer o descargar dichas publicaciones.

Requisitos del Sistema

- Plataforma: El juego fue desarrollado para ejecutarse en sistemas operativos Windows y Linux.
- Entorno de Desarrollo: Se utilizó Apache NetBeans IDE 20 para el desarrollo del código y MYSQL para el desarrollo de la base de datos.
- Lenguaje de Programación: La aplicación web se implementó en Java (Programación Orientada a Objetos), versión 17 y JSP.
- Sistema de Desarrollo: El desarrollo se llevó a cabo en un entorno Ubuntu Linux.

Componentes de la aplicación web.

- **Web Pages:**
  - Encontramos todo el apartado de la interfaz del usuario, aquí se encuentran todos los JSP de la aplicación web, las cuales enumeramos a continuacion:
    - Login.jsp
    - Perfil.jsp
    - registrarCuenta.jsp
    - publicistaJsp.jsp
    - lectorJsp.jsp

        - autorJsp.jsp
        - RepoSuscripcion.jsp
        - RepoMeGustas.jsp
        - RepoComentario.jsp
        - adminJsp.jsp
    - A de mas encontramos clases y recursos que son dedicas al completo al apartado visual de la aplicación web.
        - Includes:
            - Footer.jsp
            - Headers.jsp
            - barraAD.jsp
            - barraAI.jsp
            - resources.jsp
        - Resources
            - Son las carpetas css, img y js que tienen los diseños y los estilos para la aplicación web
- **Source Packages:**
    - Este apartado prácticamente es nuestro backen, es la parte que se encarga de comunicar todas las partes y de realizar la lógica necesaria.
        - ConexionDB:
            - Constructor conexionDB(): Inicializa la conexión a la base de datos revistas_actuales y carga el controlador de MySQL. Maneja excepciones para errores de conexión.
            - Método getConnection(): Devuelve el objeto Connection para realizar consultas a la base de datos.

- Método estático cerrarConnection(Connection connection): Cierra la conexión a la base de datos si está activa, liberando recursos.

- Revistas:
  - obtenerTodasLasRevistasDelAutor(String user): Recupera todas las revistas publicadas por un autor específico de la base de datos. Devuelve una lista de objetos Revista.
  - ObtenerTodasLasRevistas(): Recupera todas las revistas disponibles en la base de datos. Devuelve una lista de objetos Revista.
  - obtenerRevistasSuscritas(String userLector): Recupera todas las revistas a las que un lector específico está suscrito. Devuelve una lista de objetos Revista.
  - actualizarEstadoComentario(int codRevista): Cambia el estado de los comentarios (activar/desactivar) de una revista específica. Utiliza el código de la revista como parámetro.
  - actualizarEstadoMeGustas(int codRevista): Cambia el estado de los "me gusta" (activar/desactivar) de una revista específica. Utiliza el código de la revista como parámetro.
  - actualizarEstadoSuscripciones(int codRevista): Cambia el estado de las suscripciones (activar/desactivar) de una revista específica. Utiliza el código de la revista como parámetro.

- ArchivoPdf:
  - guardarTomo(archivoPdf tomo): Inserta un nuevo archivo PDF (tomo) en la base de datos. Recibe un objeto archivoPdf como parámetro, que contiene el número de revista, el nombre del archivo y el contenido del archivo en formato de bytes.
  - obtenerTomos(int noRevista): Recupera todos los tomos (archivos PDF) asociados a una revista específica de la base de datos. Utiliza el número de revista como parámetro y devuelve una lista de objetos archivoPdf, cada uno representando un tomo.

- Comentario:
  - obtenerComentarios(): Recupera todos los comentarios de la base de datos, incluyendo el contenido del comentario, el usuario que lo hizo y el nombre de la revista asociada. Devuelve una lista de objetos comentario, donde cada objeto contiene la información de un comentario.
- Suscripcion:
  - obtenerSubs(): Recupera todas las suscripciones de la base de datos, incluyendo el usuario lector, la fecha de suscripción y el nombre de la revista asociada. Devuelve una lista de objetos suscripcion, donde cada objeto contiene la información de una suscripción.
- Anuncio:
  - Anuncio(String anuncio, String fechaPago, String estado, String fechaLimite): Inicializa un nuevo objeto Anuncio con los parámetros proporcionados.
  - getAnuncio(): Devuelve el contenido del anuncio.
  - getFechaPago(): Devuelve la fecha de pago del anuncio.
  - getEstado(): Devuelve el estado del anuncio (por ejemplo, activo o inactivo).
  - getFechaLimite(): Devuelve la fecha límite de publicación del anuncio.
- AnuncioPrecios:
  - AnuncioPrecios(): Inicializa un nuevo objeto `AnuncioPrecios` sin parámetros.
  - getTipoAnuncio(): Retorna el tipo de anuncio.
  - setTipoAnuncio(String tipoAnuncio): Establece el tipo de anuncio.
  - getPrecioUnDia(): Retorna el precio por un día.

- setPrecioUnDia(double precioUnDia): Establece el precio por un día.
- getPrecioTresDias(): Retorna el precio por tres días.
- setPrecioTresDias(double precioTresDias): Establece el precio por tres días.
- getPrecioUnaSemana(): Retorna el precio por una semana.
- setPrecioUnaSemana(double precioUnaSemana): Establece el precio por una semana.
- getPrecioDosSemanas(): Retorna el precio por dos semanas.
- setPrecioDosSemanas(double precioDosSemanas): Establece el precio por dos semanas.

- Usuario:
  - crearUsuario(String user, String password, String passwordConfirm, String tipo, double billetera): Crea un nuevo usuario en la base de datos.
  - existeUsuario(String user): Verifica si un usuario existe en la base de datos.
  - iniciarSesion(Usuario user): Verifica las credenciales de inicio de sesión del usuario.
  - getUserName(): Retorna el nombre de usuario.
  - setUserName(String userName): Establece el nombre de usuario.
  - getPassword(): Retorna la contraseña.
  - setPassword(String password): Establece la contraseña.
  - getPasswordConfirm(): Retorna la confirmación de la contraseña.
  - setPasswordConfirm(String passwordConfirm): Establece la confirmación de la contraseña.
  - getTipo(): Retorna el tipo de usuario.

- setTipo(tipoUser tipo): Establece el tipo de usuario.
- getBilletera(): Retorna el saldo de la billetera.
- setBilletera(double billetera): Establece el saldo de la billetera.

- TipoUser:
  - tipoUser (enum)
  - ADMINISTRADOR: Representa el tipo de usuario administrador.
  - LECTOR: Representa el tipo de usuario lector.
  - AUTOR: Representa el tipo de usuario autor.
  - PUBLICISTA: Representa el tipo de usuario publicista.
- Serverts:
  - Un Servlet es una clase Java que se ejecuta en un servidor web y responde a las solicitudes HTTP de los clientes, generalmente navegadores, permiten manejar la lógica de negocio y la interacción con bases de datos en aplicaciones web.

Link de Draw.io:

https://drive.google.com/file/d/1I2p0ziFfdbf90dEVKkTQCQaWFPcZmSNV/view?usp=sharing

Diagrama de clases:

Diagrama E/R:

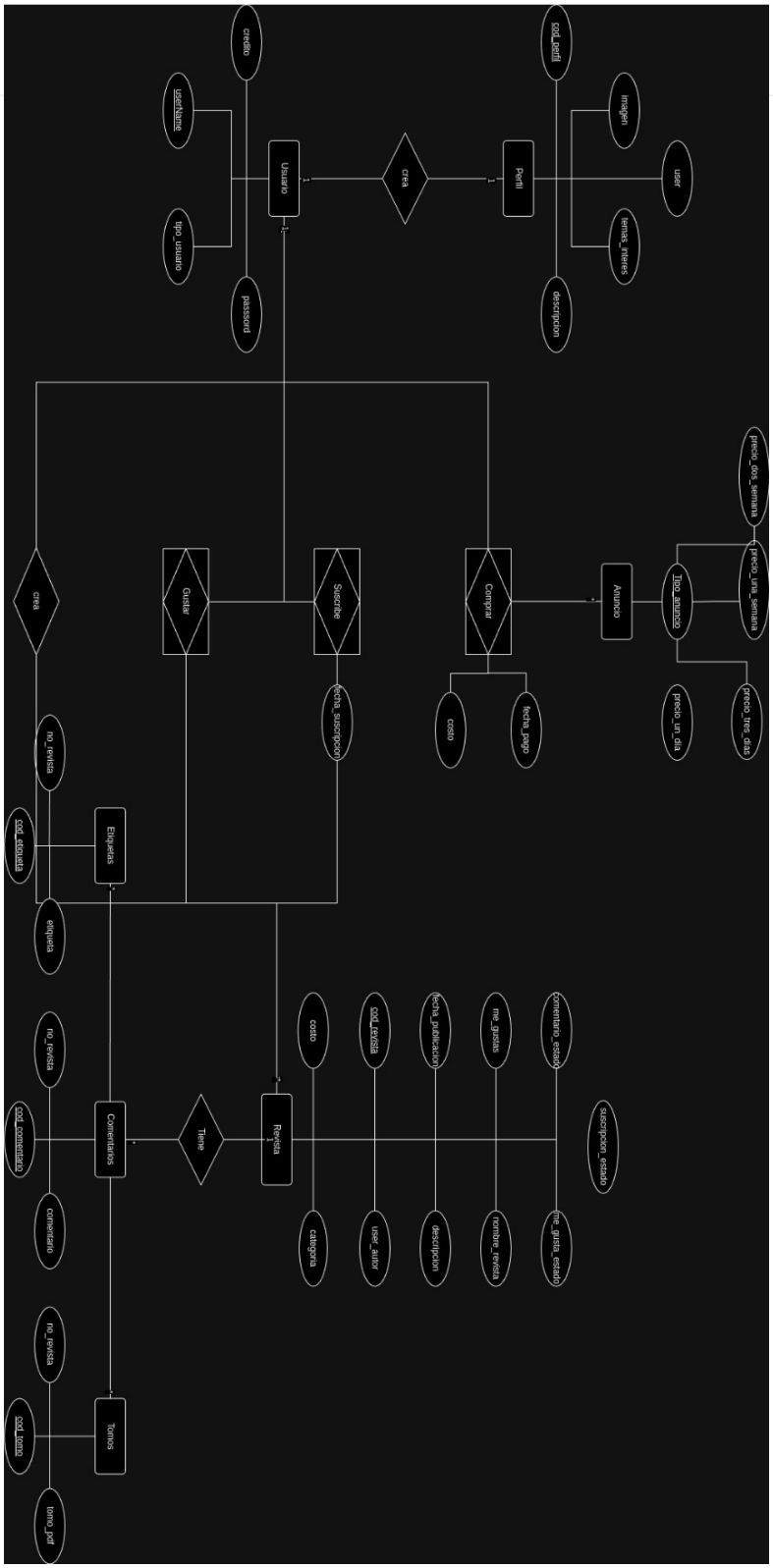
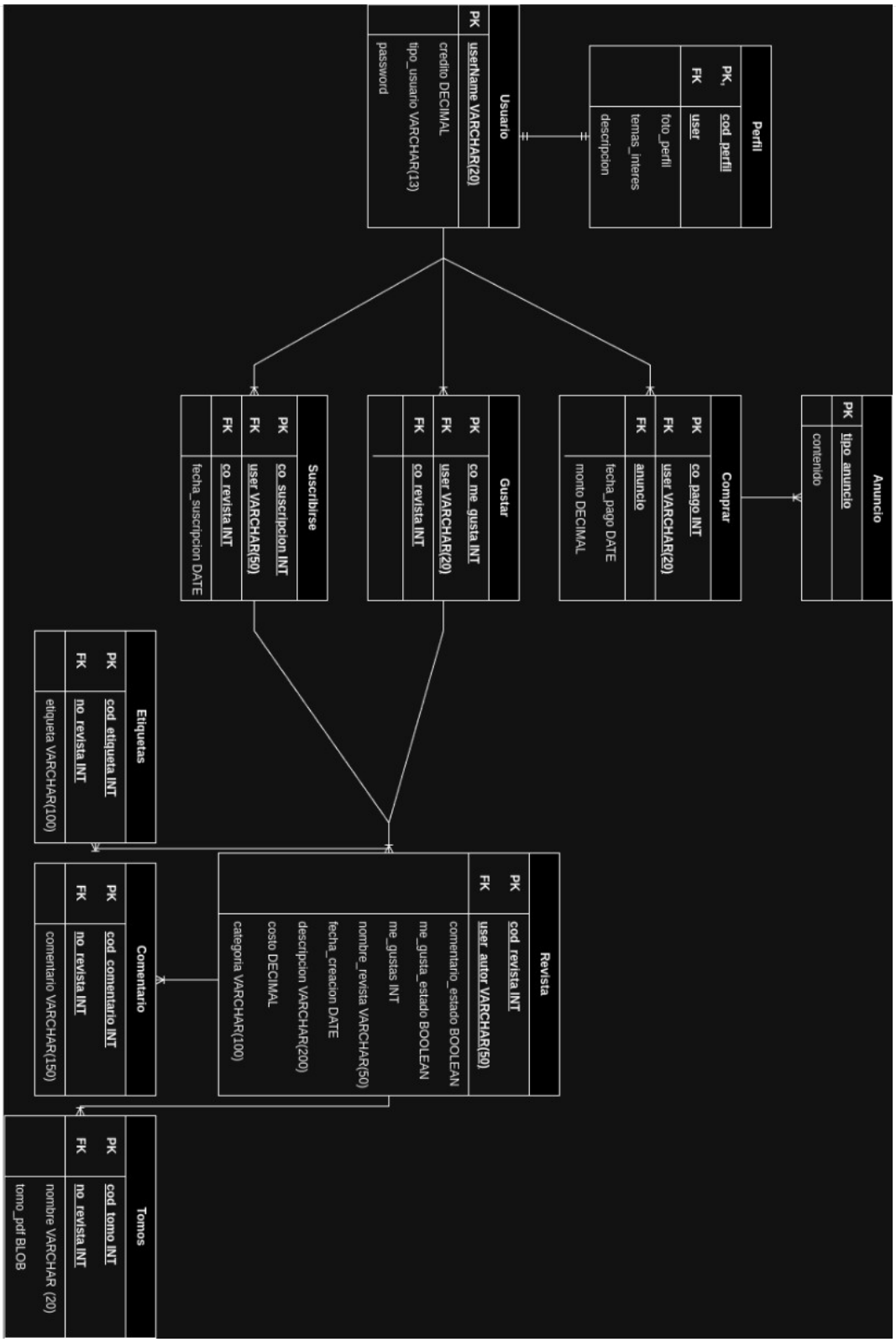

credito
username
tipo_usuario
password
Usuario
1:1
crea
Perfil
cod_perfil
imagen
user
semas_interes
descripcion
precio_dos_semana
precio_una_semana
Tipo_anuncio
precio_un_dia
precio_tres_dias
Anuncio
Comprar
costo
fecha_pago
Suscribe
fecha_suscripcion
Gustar
crea
no_revista
cod_etiqueta
Etiquetas
etiqueta
Tiene
Revista
costo
categoria
cod_revista
user_autor
fecha_publicacion
descripcion
no_gustas
nombre_revista
comentario_estado
no_gusta_estado
suscripcion_estado
no_revista
cod_comentario
Comentarios
comentario
no_revista
cod_tomo
Tomos
tomo_pdf

Diagrama de tablas:



Mapeo físico de la DB:

DDL

Crear esquema:

```
CREATE SCHEMA revistas_actuales;
USE revistas_actuales ;
```

Crear tabla USUARIO

```
CREATE TABLE USUARIO (
  user_name VARCHAR(20) NOT NULL,
  credito DECIMAL(15, 2) NOT NULL,
  tipo_usuario VARCHAR(13) NOT NULL,
  password VARCHAR(70) NOT NULL,
  CONSTRAINT PK_USUARIO PRIMARY KEY(user_name)
);
```

Crear tabla PERFIL

```
CREATE TABLE PERFIL (
  cod_perfil INT AUTO_INCREMENT NOT NULL,
  user VARCHAR(20) NOT NULL,
  temas_interes VARCHAR(150)  NULL,
  descripcion VARCHAR(200)  NULL,
  foto_perfil LONGBLOB NULL,
  CONSTRAINT PK_PERFIL PRIMARY KEY(cod_perfil),
  CONSTRAINT FK_USUARIO_IN_USER
        FOREIGN KEY (user) REFERENCES USUARIO(user_name)
);
```

Crear tabla ANUNCIO

```
CREATE TABLE ANUNCIO (
  tipo_anuncio VARCHAR(6) NOT NULL,
  precio_un_dia DECIMAL(15, 2) NULL,
  precio_tres_dias DECIMAL(15, 2) NULL,
  precio_una_semana DECIMAL(15, 2) NULL,
  precio_dos_semanas DECIMAL(15, 2) NULL,
  CONSTRAINT PK_ANUNCIO PRIMARY KEY(tipo_anuncio)
);
```

Crear tabla COMPRAR

```
CREATE TABLE COMPRAR (
  cod_compra INT AUTO_INCREMENT NOT NULL,
  usuario VARCHAR(20) NOT NULL,
  anuncio VARCHAR(6) NOT NULL,
  fecha_pago DATE NOT NULL,
  precio DECIMAL(15, 2) NOT NULL,
  estado BOOLEAN NOT NULL,
  fecha_limite DATE NOT NULL,
  CONSTRAINT PK_COMPRAR PRIMARY KEY(cod_compra),
  CONSTRAINT FK_USUARIO_IN_USUARIO
    FOREIGN KEY (usuario) REFERENCES USUARIO(user_name),
  CONSTRAINT FK_ANUNCIO_IN_ANUNCIO
    FOREIGN KEY (anuncio) REFERENCES ANUNCIO(tipo_anuncio)
);
```

Crear tabla REVISTA

```
CREATE TABLE REVISTA (
```

```
 cod_revista INT AUTO_INCREMENT NOT NULL,
 user_autor VARCHAR(20) NOT NULL,
 nombre_revista VARCHAR(20) NOT NULL,
 categoria VARCHAR(20) NOT NULL,
 descripcion VARCHAR(150) NOT NULL,
 fecha_publicacion DATE NOT NULL,
 costo DECIMAL(15, 2) NOT NULL,
 estado_comentario BOOLEAN NOT NULL,
 estado_me_gustas BOOLEAN NOT NULL,
 estado_suscripciones BOOLEAN NOT NULL,
 CONSTRAINT PK_REVISTA PRIMARY KEY(cod_revista),
 CONSTRAINT FK_USUARIO_IN_USER_AUTOR
   FOREIGN KEY (user_autor) REFERENCES USUARIO(user_name)
);
Crear tabla SUSCRIPCION
CREATE TABLE SUSCRIPCION (
 cod_suscripcion INT AUTO_INCREMENT NOT NULL,
 user_lector VARCHAR(20) NOT NULL,
 codigo_revista INT NOT NULL,
 fecha_suscripcion DATE NOT NULL,
 CONSTRAINT PK_SUSCRIPCION PRIMARY KEY(cod_suscripcion),
 CONSTRAINT FK_USUARIO_IN_USER_LECTOR
   FOREIGN KEY (user_lector) REFERENCES USUARIO(user_name),
 CONSTRAINT FK_REVISTA_IN_CODIGO_REVISTA
   FOREIGN KEY (codigo_revista) REFERENCES REVISTA(cod_revista)
```

```
);
Crear tabla TOMO
CREATE TABLE TOMO (
  cod_tomo INT AUTO_INCREMENT NOT NULL,
  no_revista INT NOT NULL,
  nombre_archivo VARCHAR(20) NOT NULL,
  archivo MEDIUMBLOB NOT NULL,
  CONSTRAINT PK_TOMO PRIMARY KEY(cod_tomo),
    CONSTRAINT FK_REVISTA_IN_NO_REVISTA
    FOREIGN KEY (no_revista) REFERENCES REVISTA(cod_revista)
);
Crear tabla MEGUSTA
CREATE TABLE MEGUSTA (
  cod_megusta INT AUTO_INCREMENT NOT NULL,
  user_like VARCHAR(20) NOT NULL,
  codigo_revi INT NOT NULL,
  CONSTRAINT PK_MEGUSTA PRIMARY KEY(cod_megusta),
  CONSTRAINT FK_USUARIO_IN_USER_LIKE
    FOREIGN KEY (user_like) REFERENCES USUARIO(user_name),
  CONSTRAINT FK_REVISTA_IN_CODIGO_REVI
    FOREIGN KEY (codigo_revi) REFERENCES REVISTA(cod_revista)
);
Crear tabla COMENTARIO
CREATE TABLE COMENTARIO (
  cod_com INT AUTO_INCREMENT NOT NULL,
```

```
  user_com VARCHAR(20) NOT NULL,
  no_revista_com INT NOT NULL,
  contenido VARCHAR(150) NOT NULL,
  CONSTRAINT PK_COMENTARIO PRIMARY KEY(cod_com),
  CONSTRAINT FK_USUARIO_IN_USER_COM
    FOREIGN KEY (user_com) REFERENCES USUARIO(user_name),
    CONSTRAINT FK_REVISTA_IN_NO_REVISTA_COM
    FOREIGN KEY (no_revista_com) REFERENCES REVISTA(cod_revista)
);
```

DML tabla ANUNCIO

```
INSERT INTO ANUNCIO (tipo_anuncio, precio_un_dia, precio_tres_dias,
precio_una_semana,precio_dos_semanas) VALUES ('TEXTO', '15.00', '25.00',
'100.00','175.00');
INSERT INTO ANUNCIO (tipo_anuncio, precio_un_dia, precio_tres_dias,
precio_una_semana,precio_dos_semanas) VALUES ('IMGTXT', '20.00', '35.00',
'110.00','190.00');
INSERT INTO ANUNCIO (tipo_anuncio, precio_un_dia, precio_tres_dias,
precio_una_semana,precio_dos_semanas) VALUES ('VIDEO', '25.00', '45.00',
'150.00','300.00');
```