Codigo de tajeta de credito

Concepto general

TigerCard permite:

- Registrar usuarios con correo y contraseña.
- Asignarles automáticamente una tarjeta de crédito.
- Iniciar sesión y ver su perfil.
- Registrar compras que afectan el saldo de su tarjeta.

Componentes clave

1. **Modelos (Models)**

Definen la estructura de datos:

Modelo	Propósito
Usuario	Representa al usuario con nombre, email, contraseña, intentos fallidos...
TarjetaCredito	Tarjeta asociada al usuario con número, CVV, límite, saldo, fechas...
Transaccion	Registro de compras o pagos con monto, fecha, categoría, descripción...
LoginViewModel	Datos para iniciar sesión (email y contraseña).
RegisterViewModel	Datos para registrarse (nombre, email, contraseña, confirmación).

2. **Controladores (Controllers)**

AccountController

- **Registro (Register)**: Crea un usuario y le asigna una tarjeta con datos generados automáticamente.

- **Inicio de sesión (Login)**: Verifica credenciales, controla intentos fallidos y bloquea si hay 3 errores.
- **Cerrar sesión (Logout)**: Elimina la sesión del usuario.

UsuarioController

- Muestra el perfil del usuario, sus tarjetas y transacciones recientes.

TransaccionController

- Permite registrar una compra si hay saldo disponible en la tarjeta.

3. **Base de datos (ApplicationDbContext)**

Usa Entity Framework Core para definir las tablas:

- Usuarios
- Tarjetas
- Transacciones

Incluye relaciones:

- Un usuario puede tener varias tarjetas.
- Una tarjeta puede tener varias transacciones.

4. **Vistas (Views)**

Son las páginas HTML que el usuario ve:

- Login.cshtml: formulario de inicio de sesión.
- Register.cshtml: formulario de registro.
- Usuario/Index.cshtml: perfil del usuario con tarjetas y transacciones.
- Transaccion/Create.cshtml: formulario para registrar una compra.

5. **Program.cs**

Configura el servidor:

- Usa sesiones para mantener al usuario conectado.
- Define la ruta inicial: Account/Login.

Seguridad y lógica importante

- Contraseñas se almacenan con PasswordHasher (no en texto plano).
- Se bloquea la cuenta tras 3 intentos fallidos.
- Se valida que el correo no esté duplicado.
- Se genera automáticamente:
  - Número de tarjeta (16 dígitos aleatorios).
  - CVV (3 dígitos).
  - Fechas de corte y pago

.

1. **Modelos (Models):** Son las clases que definen cómo se estructura la información en el sistema. Por ejemplo, el modelo Usuario guarda datos del usuario como nombre, correo y contraseña; TarjetaCredito guarda los datos de la tarjeta; y Transaccionregistra las compras o pagos.
2. **Controladores (Controllers):** Son los que manejan la lógica de la aplicación y responden a las acciones del usuario. Por ejemplo, AccountController gestiona el registro, inicio y cierre de sesión; UsuarioController muestra el perfil y datos del usuario; y TransaccionController permite registrar compras.
3. **Base de datos (ApplicationDbContext):** Aquí se definen las tablas y relaciones entre usuarios, tarjetas y transacciones usando Entity Framework Core, que facilita trabajar con datos en la base.
4. **Vistas (Views):** Son las páginas web que el usuario ve y con las que interactúa, como formularios para iniciar sesión o registrar compras.
5. **Seguridad:** Las contraseñas no se guardan en texto plano, sino que se encriptan con PasswordHasher. Además, se bloquea la cuenta tras 3 intentos fallidos para proteger al usuario.
6. **Program.cs:** Configura el servidor web, las sesiones para mantener al usuario conectado y la ruta inicial para que el usuario empiece en la página de login.