diagrama de caso de uso

```
@startuml
title Diagrama de caso de uso CU-02 - Sistema de autenticación
left to right direction
skinparam packageStyle rectangle

actor Cliente as C

package "Sistema de Autenticación" {
   usecase "CU-02: Iniciar Sesión" as UC02
   usecase "Validar credenciales" as UC_Validar
   usecase "Mostrar formulario de login" as UC_Form
   usecase "Crear sesión segura / Token" as UC_Sesion
   usecase "Mostrar mensaje de error" as UC_Error
   usecase "Bloquear cuenta" as UC_Bloqueo
}

C --> UC02
UC02 --> UC_Form
UC02 --> UC_Validar
UC_Validar --> UC_Sesion : Si las credenciales son correctas
UC_Validar --> UC_Error : Si las credenciales son incorrectas
UC_Validar --> UC_Bloqueo : Tras 5 intentos fallidos

@enduml
```

Diagrama de clase rf-02

```
@startuml
title diagrama de clase HU-02 - sistema de autenticación
skinparam classAttributeIconSize 0

class Cliente {
   - id: int
   - nombre: string
   - correo: string
   - contraseñaHash: string
   - estadoCuenta: string
   - intentosFallidos: int

   +validarCredenciales(correo: string, contraseña: string): boolean
   +bloquearCuenta(): void
   +reiniciarIntentos(): void
}

class SistemaAutenticacion {
   - servicioHash: ServicioHash
   - baseDatos: BaseDatosUsuarios

   +iniciarSesion(correo: string, contraseña: string): ResultadoAutenticacion
}

class ServicioHash {
   +verificarHash(contraseñaIngresada: string, hashAlmacenado: string): boolean
   +generarHash(contraseña: string): string
}

class ResultadoAutenticacion {
   - exito: boolean
   - mensaje: string
   - redireccion: string
}

class BaseDatosUsuarios {
   +obtenerClientePorCorreo(correo: string): Cliente
   +actualizarCliente(cliente: Cliente): void
}

Cliente --> SistemaAutenticacion : utiliza
SistemaAutenticacion --> ServicioHash : usa
SistemaAutenticacion --> BaseDatosUsuarios : consulta/actualiza
SistemaAutenticacion --> ResultadoAutenticacion : retorna
@enduml
```

diagrama de secuencia Historia de Usuario HU-02: Iniciar sesión en el sistema (RF-02)

```
@startuml
title Diagrama de Secuencia Historia de Usuario HU-02: Iniciar sesión en el sistema (RF-02)
actor Cliente
participant "Pantalla de Login" as UI
participant "SistemaAutenticacion" as SA
participant "BaseDatosUsuarios" as BD
participant "ServicioHash" as Hash

Cliente -> UI : Ingresa correo y contraseña
UI -> SA : iniciarSesion(correo, contraseña)

SA -> BD : obtenerClientePorCorreo(correo)
BD --> SA : Cliente

alt Cliente existe
   SA -> Hash : verificarHash(contraseñaIngresada, contraseñaHash)
   Hash --> SA : true/false

   alt Contraseña válida
      SA -> Cliente : reiniciarIntentos()
      SA --> UI : Autenticación exitosa\nRedirigir a dashboard
   else Contraseña inválida
      SA -> Cliente : incrementarIntentos()
      alt intentos < 5
         SA --> UI : Mostrar "Correo o contraseña incorrectos"
      else intentos >= 5
         SA -> Cliente : bloquearCuenta()
         SA --> UI : Mostrar "Cuenta bloqueada temporalmente"
      end
   end
else Cliente no existe
   SA --> UI : Mostrar "Correo o contraseña incorrectos"
end

@enduml
```

Diagrama de actividades - HU-02: Iniciar sesión en el sistema (RF-02)

```
@startuml
title Diagrama de Actividades - HU-02: Iniciar sesión en el sistema (RF-02)

start

:Cliente accede a la pantalla de login;
:Ingresa correo y contraseña;
:Sistema recibe credenciales;

if (Correo existe en la base de datos?) then (Sí)
  :Obtener datos del cliente;
  :Comparar contraseña con el hash;

  if (¿Contraseña válida?) then (Sí)
    :Reiniciar intentos fallidos;
    :Crear sesión o token;
    :Redirigir al dashboard;
  else (No)
    :Incrementar contador de intentos;

    if (¿Intentos >= 5?) then (Sí)
      :Bloquear cuenta temporalmente;
      :Mostrar mensaje "Cuenta bloqueada temporalmente";
    else (No)
      :Mostrar mensaje "Correo o contraseña incorrectos";
    endif
  endif

else (No)
  :Mostrar mensaje "Correo o contraseña incorrectos";
endif

stop
@enduml
```

diagrama de estado rf-02 **HU-02: Iniciar sesión en el sistema (RF-02)**

```
@startuml
title Diagrama de Estados - HU-02: Iniciar sesión en el sistema (RF-02)

[*] --> EsperandoCredenciales

EsperandoCredenciales --> ValidandoCredenciales : Cliente ingresa correo y contraseña

ValidandoCredenciales --> Autenticado : Credenciales correctas
ValidandoCredenciales --> ErrorCredenciales : Credenciales incorrectas
ErrorCredenciales --> ValidandoCredenciales : Cliente reintenta

ErrorCredenciales --> CuentaBloqueada : 5 intentos fallidos

Autenticado --> [*] : Acceso concedido

CuentaBloqueada --> [*] : Bloqueo temporal\n(Requiere desbloqueo o espera)

@enduml
```

Diagrama de componente RF-02 **HU-02: Iniciar sesión en el sistema (RF-02)**

```
@startuml
title Diagrama de Componentes - HU-02: Iniciar sesión en el sistema (RF-02)

package "Frontend" {
  [Pantalla de Login] --> [API de Autenticación]
}

package "Backend" {
  [API de Autenticación] --> [Servicio de Autenticación]
  [Servicio de Autenticación] --> [Gestor de Usuarios]
  [Servicio de Autenticación] --> [Servicio de Seguridad]
}

package "Persistencia" {
  [Gestor de Usuarios] --> [Base de Datos de Usuarios]
}

[Servicio de Seguridad] --> [Hash de Contraseña]

@enduml
```

Diagrama de despliegue RF-02 **HU-02: Iniciar sesión en el sistema (RF-02)**

```
@startuml
title Diagrama de Despliegue - HU-02: Iniciar sesión en el sistema (RF-02)

node "Cliente Web\n(Navegador)" {
  component "Pantalla de Login"
}

node "Servidor Web\n(Nginx / Apache)" {
  component "API REST de Autenticación"
}

node "Servidor de Aplicaciones\n(Node.js / Django / Spring Boot)" {
  component "Servicio de Autenticación"
  component "Gestor de Usuarios"
  component "Servicio de Seguridad (Hash)"
}

node "Servidor de Base de Datos\n(MySQL / PostgreSQL)" {
  database "Base de Datos de Usuarios"
}

"Pantalla de Login" --> "API REST de Autenticación" : Solicitud de login (HTTP/HTTPS)
"API REST de Autenticación" --> "Servicio de Autenticación"
"Servicio de Autenticación" --> "Gestor de Usuarios"
"Gestor de Usuarios" --> "Base de Datos de Usuarios" : Consulta/verificación
"Servicio de Autenticación" --> "Servicio de Seguridad (Hash)"

@enduml
```

Diagrama de Objeto RF-02 **HU-02: Iniciar sesión en el sistema (RF-02)**

```
@startuml
title Diagrama de Objetos - HU-02: Iniciar sesión en el sistema (RF-02)

object cliente1 {
  id = 102
  correo = "usuario@ejemplo.com"
  contraseñaHash = "a1b2c3..."
  intentosFallidos = 2
  estadoCuenta = "Activa"
}

object formularioLogin {
  correo = "usuario@ejemplo.com"
  contraseña = "123456"
}

object servicioAutenticacion

object servicioHash

object resultadoAutenticacion {
  exito = true
  mensaje = "Autenticación exitosa"
  redireccion = "/dashboard"
}

formularioLogin --> servicioAutenticacion : enviar credenciales
servicioAutenticacion --> cliente1 : buscar por correo
servicioAutenticacion --> servicioHash : verificarHash(contraseña, contraseñaHash)
servicioAutenticacion --> resultadoAutenticacion : retornar resultado

@enduml
```

Diagrama de Flujo RF-02 **HU-02: Iniciar sesión en el sistema (RF-02)**

```
@startuml
title Diagrama de Flujo - HU-02: Iniciar sesión en el sistema (RF-02)

start

:Cliente accede al formulario de login;
:Introduce correo y contraseña;
:Enviar credenciales al servidor;

if (¿Correo existe?) then (Sí)
  :Obtener datos del cliente;

  if (¿Contraseña es correcta?) then (Sí)
    :Reiniciar contador de intentos fallidos;
    :Crear sesión o token;
    :Redirigir al dashboard;
  else (No)
    :Incrementar intentos fallidos;

    if (¿Intentos >= 5?) then (Sí)
      :Bloquear cuenta;
      :Mostrar mensaje: "Cuenta bloqueada temporalmente";
    else (No)
      :Mostrar mensaje: "Correo o contraseña incorrectos";
    endif
  endif

else (No)
  :Mostrar mensaje: "Correo o contraseña incorrectos";
endif

stop
@enduml
```