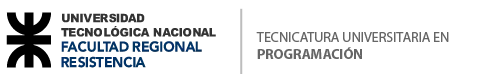
**Universidad Tecnológica Nacional**

**INFORME FINAL**

## Programación Web

## Comision: Tarde

## Integrantes:

## Ignacio Bianchi Berman

## Jose Oscar Thorlet

## Luciano Ivan Cirilo Rodriguez

## Marco Antonio Jesus Godoy

## Lucas Rabinovich

**Sistema de Gestión de Peluquería**

**Proyecto Integrador - Desarrollo Web**

**Link Github:** <https://github.com/IgnacioBianchi-cmd/peluqueria-web.git>  
**Versión:** 1.0  
**Fecha:** 10 de Julio de 2025

**1. INTRODUCCIÓN**

**1.1 Descripción del Proyecto**

El presente proyecto consiste en el desarrollo de una solución completa para la gestión de una peluquería, implementando tres aplicaciones complementarias que trabajan de manera integrada para satisfacer tanto las necesidades de los clientes como del personal administrativo.

**1.2 Escenario Elegido**

Se eligió el dominio de servicios de peluquería por su relevancia práctica y la oportunidad de implementar funcionalidades diversas que permiten demostrar el uso de las tecnologías requeridas. El sistema permite a los clientes solicitar servicios, agendar turnos y realizar pagos, mientras que el personal administrativo puede gestionar eficientemente la operación del negocio.

**2. ARQUITECTURA DEL SISTEMA**

**2.1 Componentes Desarrollados**

**A) Aplicación Web ASP.NET MVC con Razor Pages**

**Propósito:** Interfaz principal para clientes  
**Tecnologías:** ASP.NET Core MVC, Razor Pages, Entity Framework Core

**B) API RESTful ASP.NET Web API**

**Propósito:** Servicios backend para la aplicación SPA  
**Tecnologías:** ASP.NET Core Web API, JWT Authentication

**C) Aplicación SPA (Single Page Application)**

**Propósito:** Panel administrativo para gestión de turnos  
**Tecnologías:** React y CSS Tailwind

**3. FUNCIONALIDADES IMPLEMENTADAS**

**3.1 Aplicación Cliente (Razor Pages)**

**Gestión de Usuarios**

* **Registro de nuevos usuarios:** Sistema completo de registro con validaciones
* **Inicio de sesión:** Autenticación segura con usuario y contraseña
* **Confirmación de email:** Proceso de verificación de correo electrónico
* **Recuperación de contraseña:** Sistema de recuperación mediante enlace único

**Gestión de Servicios y Turnos**

* **Selección de servicios:** Catálogo completo de servicios disponibles
* **Agendamiento de turnos:** Sistema de reservas con disponibilidad en tiempo real
* **Historial de pedidos:** Visualización completa del historial de servicios

**Sistema de Pagos**

* **Múltiples métodos de pago:** Efectivo y tarjeta de crédito/débito
* **Guardar tarjetas:** Funcionalidad para almacenar métodos de pago de forma segura
* **Procesamiento de pagos:** Integración con sistemas de pago

**Confirmación con QR**

* **Generación de QR dinámico:** Códigos QR únicos para cada turno
* **Tiempo limitado:** QR válido hasta 1 hora después del inicio del turno
* **Confirmación automática:** Sistema de confirmación mediante escaneo

**3.2 Aplicación Administrativa (SPA)**

**Gestión Administrativa**

* **Registro de administradores:** Sistema con clave secreta administrativa
* **Panel de control:** Interfaz completa para gestión de turnos
* **Gestión de turnos:** Visualización, edición y eliminación de turnos
* **Control de QR:** Capacidad de desactivar códigos QR expirados

**4. CUMPLIMIENTO DE REQUERIMIENTOS TÉCNICOS**

**4.1 Arquitectura MVC**

✅ **Cumplido:** Implementación completa del patrón MVC con Razor Pages para la separación clara de responsabilidades entre modelo, vista y controlador.

**4.2 Gestión de Usuarios**

✅ **Cumplido:**

* Sistema de registro de nuevos usuarios implementado
* Inicio de sesión con usuario y contraseña funcional
* Contraseñas almacenadas con hash y salt siguiendo buenas prácticas de seguridad

**4.3 Recuperación de Contraseña**

✅ **Cumplido:** Sistema de recuperación mediante correo electrónico con enlace único y seguro por usuario.

**4.4 Gestión de Datos**

✅ **Cumplido:**

* **Listado paginado:** Implementado en el historial de turnos y servicios
* **Vista de detalle:** Cada turno y servicio tiene su vista detallada específica
* **Crear elementos:** Sistema de creación de turnos con múltiples campos y selectores
* **Editar elementos:** Funcionalidad completa de edición de turnos existentes

**4.5 Sistema QR**

✅ **Cumplido:**

* **Generación en backend:** Los códigos QR se generan desde .NET
* **Enlace con contenido:** Cada QR contiene un enlace único al turno
* **Funcionalidad exclusiva:** Acceso a confirmación de turno solo mediante QR
* **Caducidad temporal:** Enlaces válidos por tiempo limitado (1 hora)

**4.6 API RESTful**

✅ **Cumplido:**

* Endpoints RESTful para la aplicación SPA
* Autenticación mediante JWT para endpoints protegidos
* Manejo adecuado de códigos de estado HTTP

**4.7 Aplicación SPA**

✅ **Cumplido:**

* Desarrollada con framework JavaScript moderno
* Ejecutión completa en el cliente
* Comunicación con servidor solo mediante API

**5. CARACTERÍSTICAS DE SEGURIDAD**

**5.1 Autenticación y Autorización**

* **Hash y Salt:** Contraseñas almacenadas con algoritmos seguros
* **JWT Tokens:** Autenticación stateless para la API
* **Roles de usuario:** Diferenciación entre clientes y administradores
* **Clave administrativa:** Acceso restringido al panel administrativo

**5.2 Validaciones**

* **Validaciones del lado cliente:** JavaScript para mejor UX
* **Validaciones del lado servidor:** ASP.NET Core para seguridad
* **Sanitización de datos:** Prevención de inyecciones

**5.3 Gestión de Sesiones**

* **Tiempo de vida de tokens:** Configuración apropiada de expiración
* **Invalidación de sesiones:** Capacidad de cerrar sesiones remotamente

**6. TECNOLOGÍAS UTILIZADAS**

**6.1 Backend**

* **ASP.NET Core MVC:** Framework principal para la aplicación web
* **ASP.NET Core Web API:** Para servicios RESTful
* **Entity Framework Core:** ORM para acceso a datos
* **SQL Server:** Base de datos relacional
* **JWT:** Para autenticación en la API

**6.2 Frontend**

* **Razor Pages:** Para la aplicación cliente
* **[Framework SPA]:** Para el panel administrativo
* **HTML5, CSS3, JavaScript:** Tecnologías web estándar
* **Bootstrap:** Para diseño responsive

**6.3 Herramientas de Desarrollo**

* **Visual Studio/Visual Studio Code:** IDEs principales
* **Git:** Control de versiones
* **GitHub:** Repositorio remoto
* **SQL Server Management Studio:** Gestión de base de datos

**7. BASE DE DATOS**

**7.1 Diseño de la Base de Datos**

El diseño incluye las siguientes entidades principales:

**Usuarios**

* Información personal y credenciales
* Roles y permisos
* Historial de actividad

**Servicios**

* Catálogo de servicios disponibles
* Precios y duración
* Descripción y características

**Turnos**

* Información del agendamiento
* Relación con usuario y servicio
* Estado y confirmación
* Códigos QR asociados

**Pagos**

* Métodos de pago utilizados
* Información de transacciones
* Tarjetas guardadas (encriptadas)

**7.2 Migraciones**

Se incluyen scripts SQL y migraciones de Entity Framework para la creación y mantenimiento de la base de datos.

**8. ASPECTOS DESTACADOS**

**8.1 Características Especiales**

* **Sistema QR dinámico:** Implementación única de códigos QR con tiempo limitado
* **Doble interfaz:** Aplicación cliente y administrativa separadas
* **Seguridad robusta:** Múltiples capas de seguridad implementadas
* **Experiencia de usuario:** Interfaces intuitivas y responsive

**8.2 Buenas Prácticas Implementadas**

* **Separación de responsabilidades:** Arquitectura limpia y mantenible
* **Manejo de errores:** Gestión comprehensiva de excepciones
* **Validaciones:** Validación tanto del lado cliente como servidor
* **Diseño responsive:** Adaptación a múltiples dispositivos

**9. INSTALACIÓN Y CONFIGURACIÓN**

**9.1 Requisitos del Sistema**

* .NET 6.0 o superior
* SQL Server 2019 o superior
* Visual Studio 2022 o Visual Studio Code
* Node.js (para la aplicación SPA)

**9.2 Pasos de Instalación**

1. Clonar el repositorio desde GitHub
2. Configurar la cadena de conexión en appsettings.json
3. Ejecutar migraciones de Entity Framework
4. Configurar las claves secretas y tokens JWT
5. Instalar dependencias de la aplicación SPA
6. Ejecutar los proyectos en el orden especificado

**9.3 Configuración Inicial**

* Crear usuario administrador inicial
* Configurar servicios de email
* Establecer configuración de pagos
* Configurar parámetros de QR

**10. TESTING Y VALIDACIÓN**

**10.1 Pruebas Realizadas**

* **Pruebas unitarias:** Validación de lógica de negocio
* **Pruebas de integración:** Verificación de comunicación entre componentes
* **Pruebas de usuario:** Validación de flujos completos
* **Pruebas de seguridad:** Verificación de autenticación y autorización

**10.2 Escenarios de Prueba**

* Registro y login de usuarios
* Creación y gestión de turnos
* Procesamiento de pagos
* Generación y validación de códigos QR
* Funcionalidades administrativas

**11. CONCLUSIONES**

**11.1 Objetivos Cumplidos**

El proyecto ha cumplido satisfactoriamente todos los requerimientos técnicos establecidos, implementando una solución completa y funcional para la gestión de una peluquería. Se ha logrado:

* Integración exitosa de las tres aplicaciones componentes
* Implementación de todas las funcionalidades requeridas
* Cumplimiento de estándares de seguridad
* Desarrollo de una experiencia de usuario intuitiva

**11.2 Aprendizajes Obtenidos**

* Comprensión profunda del patrón MVC y su implementación
* Experiencia práctica con APIs RESTful y autenticación JWT
* Desarrollo de aplicaciones SPA modernas
* Implementación de sistemas de seguridad robustos

**11.3 Mejoras Futuras**

* Integración con sistemas de pago externos
* Implementación de notificaciones push
* Desarrollo de aplicación móvil nativa
* Análisis de datos y reportes avanzados

**Fin del Informe**