E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Aplicación Web en ASP.NET con arquitectura desacoplada para la gestión de peticiones de la cédula de habitabilidad.

Grado en Ingeniería Informática

Trabajo Fin de Grado

Eduard Vlad Ilie

José Javier Astrain Escola

Pamplona, 27/10/2020

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Resumen:

El presente Trabajo de fin de Grado ha sido desarrollado en el Servicio de Sistemas de Información Departamentales (SSID), del Gobierno de Navarra, en la Sección de Proyectos en las Áreas de Ordenación del Territorio, Cohesión Territorial, Desarrollo Rural y Medio Ambiente. Mi trabajo ha sido clasificado como un desarrollador frontend.

Este proyecto consiste en trabajar sobre una aplicación web que se encarga de gestionar el trámite de una cédula de habitabilidad. Crear nuevas vistas y mejorar las existentes acorde a las reglas de negocios y los requisitos.

Fue desarrollada a partir de una arquitectura propia orientada al dominio en ASP.NET con lenguaje C#. Para la capa de presentación emplea Bootstrap 3 y la aplicación se conecta con variedad de servicios internos. La aplicación se conecta a una base de datos SQL Server y todas las consultas se realizan mediante procedimientos almacenados.

La gestión de la cédula de habitabilidad se realizaba dentro de una aplicación antigua mucho más grande. Los cambios y las nuevas necesidades del departamento de vivienda, la complejidad de la aplicación existente y el deseo del grupo de desarrollo de que todos los productos sean originados a partir de la misma arquitectura de desarrollo, han llevado a crear un producto desde cero utilizando la arquitectura por defecto.

Palabras Clave:

Arquitectura orientada al dominio, ASP.NET, Bootstrap 3, Cédula Habitabilidad, Arquitectura de N capas

Contenido

Resumen:....2
Palabras Clave:....2
Tabla de Ilustraciones:....4
1. Introducción:....6
1.1. Antecedentes:....6
1.2. Justificación y Objetivos:....6
Objetivos:....7
1.3. Estado del arte:....7
1.4. Solución propuesta:....8
1.4.1. Framework:....8
1.4.2. Tecnologías actuales:....10
1.5. Metodología empleada:....13
2. Análisis:....14
2.1. Visión general:....14
2.2. Requisitos funcionales:....14
2.3. Requisitos no funcionales:....15
2.4. Requisitos de la interfaz de usuario....16
2.4.1. Requisitos:....16
2.4.2. Identificación de las restricciones técnicas:....16
2.4.3. Perfiles de usuario:....16
2.5. Casos de uso:....17
3. Diseño:....22
3.1. ¿Por qué definir una arquitectura?....22
3.2. Diseño guiado por el dominio:....22
3.3. Descripción de la arquitectura:....23
3.4. Diseño de la implementación de los casos de uso:....24
Validaciones:....25
4. Desarrollo:....52
5. Validación:....55
6. Conclusiones y líneas futuras....58
6.1. Conclusiones:....58
6.2. Líneas futuras:....58
Glosario:....59
Referencias....60

Tabla de Ilustraciones:

Ilustración 1:Patrón de Diseño Modelo Vista Controlador 8
Ilustración 2:Logo Java Enterprise Edition 8
Ilustración 3:Logo ASP:NET MVC de Microsoft 9
Ilustración 4:Volumen de búsqueda de ASP.NET MVC vs Java Web (Google, 2020).....9
Ilustración 5:Número total de Páginas Web (Stats, 2020) 10
Ilustración 6:Logotipo de HTML 5 11
Ilustración 7:Logotipo de JavaScript 11
Ilustración 8:Logotipo jQuery 11
Ilustración 9: Logotipo Bootstrap 3 12
Ilustración 10.Logotipo Foundation 12
Ilustración 11:Logotipo SQL Server 2012 13
Ilustración 12:Diagrama parcial de los casos de uso 17
Ilustración 13:Vista de arquitectura lógica simplificada de un sistema de N-capas DDD (Iberica, Microsoft, s.f.) 23
Ilustración 14:Fragmento pestaña Solicitante, datos obligatorios 26
Ilustración 15:Validaciones de campo Teléfono 26
Ilustración 16: Mascaras de validación de campos en el cliente, Se valida el formato del campo Teléfono y Código postal 26
Ilustración 17:Regla de validación del campo Teléfono en el Servidor 26
Ilustración 18: Segunda parte del JavaScript 27
Ilustración 19:Validación de los campos NIF, Nombre y Apellido 27
Ilustración 20: Reglas de validación del campo NIF en el Servidor 27
Ilustración 21:Evitar que el NIF del representante y del solicitante no sea el mismo 28
Ilustración 22:Mensaje Validación NIF 28
Ilustración 23:Poblar los campos del solicitante si su NIF se encuentra en la BD 29
Ilustración 24:Llamada encargada de obtener los datos y devolver respuesta a la llamada Ajax 30
Ilustración 25:Reglas de comprobación del check según caso 31
Ilustración 26:Implementación del mailto 31
Ilustración 27:Definición de la ayuda contextual 31
Ilustración 28:Captura de en la vista de la ayuda contextual 32
Ilustración 29:Diagrama de secuencia Vista Solicitudes Ciudadano 32
Ilustración 30:Prototipado pencil de la vista Solicitudes Ciudadano 33
Ilustración 31:Estructura que siguen todos los procedimientos almacenados 34
Ilustración 32:Procedimiento almacenado para obtener el número de Solicitudes que tiene un Cliente 36
Ilustración 33:Vista de como quedaría el menú inicio con el nuevo botón 36
Ilustración 34:Poblar los campos de la vista según los datos del Objeto Solicitud 37
Ilustración 35:Solución que deshabilita cualquier campo según el valor de la variable isReadOnly 38
Ilustración 36:Tabla de strings 38
Ilustración 37:Tabla Strings en formato XML 39
Ilustración 38: Getter del String llmado Actualizar 39
Ilustración 39:Definición de una etiqueta de un campo 39
Ilustración 40:Strings estáticos junto con strings dependiendo de la solicitud en una misma línea 40

Ilustración 41: Captura del editor Reporta dentro de Visual Studio.....41
Ilustración 42: PDF Cédula de Habitabilidad generado a partir de una solicitud 42
Ilustración 43:Prototipado Pencil de la vista Búsqueda Solicitudes 44
Ilustración 44: Ejemplo diseño fila del grid 44
Ilustración 45: Ejemplo de un campo y su etiqueta.....45
Ilustración 46: Organización de los distintos fragmentos que componen las vistas de Filtro Expediente.....45
Ilustración 47: Definición de la tabla en la que se presentan los resultados 46
Ilustración 48: Captura de pantalla de la vista implementada.....46
Ilustración 49: Filtro parcial por defecto.....47
Ilustración 50: Filtro parcial sin tener en cuenta mayúsculas o acentos 47
Ilustración 51: Definición de la variable en Web.config 48
Ilustración 52: Variable filas que guarda el resultado de la llamada al procedimiento almacenado 48
Ilustración 53: Fragmento procedimiento almacenado GET_ExpedienteNumero_count.....48
Ilustración 54:Logica dentro de ExpedienteController 49
Ilustración 55:Logica para asignar valor a los campos año si no lo tienen seleccionado.....49
Ilustración 56: Captura de los campos adicionales que se quieren añadir a la vista.....50
Ilustración 57: Logotipo grayLog.....50
Ilustración 58: Tipos de logs 50
Ilustración 59: Esquema básico de una iteración de un Método Ágil.....52
Ilustración 60: Arquitectura de la plataforma de desarrollo de software (Foto tomada de la presentación de la PDS) 53
Ilustración 61: Trabajar desde la empresa en la Intranet 54
Ilustración 62: Trabajar desde casa a través de VPN 55

1. Introducción:

1.1. Antecedentes:

El trabajo de fin de Grado lo desarrollé en la empresa Hiberus. Esta empresa esta subcontratada por el Gobierno de Navarra para la gestión y el mantenimiento de los sistemas de información y aplicaciones de carácter vertical o departamental de la Administración de la Comunidad Foral y sus organismos autónomos.

Concretamente el Servicio de Sistemas de Información Departamentales (en adelante SSID) tiene como clientes a todos los Departamentos del Gobierno de Navarra menos el departamento de salud que está atendido por SSIAS (Servicio de Sistemas de información del Área Sanitaria) y el área de economía y hacienda atendido por SSIC (Servicio de Sistemas de Información Corporativos).

He formado parte del grupo de desarrollo dentro de la SSID durante 4 meses. Las primeras semanas ha sido presencial, pero debido a la pandemia de COVID 19 el resto de las practicas curriculares las he desarrollado desde casa. Concretamente, he formado parte de la sección "*Sección de Proyectos en las Áreas de Ordenación del Territorio, Cohesión Territorial, Desarrollo Rural y Medio Ambiente*".

Mi tutor en la empresa fue Sergio Ibáñez Escarda, encargado de gestionar el grupo de desarrollo y la organización. Sin embargo, la coordinación y asignación de tareas durante mis prácticas han sido realizadas por Enrique Sierra Fernández, un desarrollador backend que ha llevado la mayor parte del desarrollo del proyecto hasta ese momento.

El equipo de desarrollo estaba formado por 5 personas. La gestora del proyecto, dos desarrolladores backend y dos desarrolladores frontend. He realizado actividades propias de un desarrollador frontend.

El proyecto comenzó a desarrollarse a finales de 2019 y me incorporé en la fase dos de la aplicación. La fase dos de la aplicación consiste en finalizar el backoffice y comenzar el desarrollo del frontend para la parte del cliente y llevar a la vez el desarrollo tanto de la parte del cliente ciudadano como el de los usuarios internos manteniendo la paridad en las funcionalidades donde sea necesario.

1.2. Justificación y Objetivos:

Se ha escogido esta aplicación como tema del trabajo final de carrera por la interesante oportunidad que nos ofrece para conocer las posibilidades de la plataforma .NET de Microsoft. Hay que aclarar que la obtención de la aplicación dista mucho de ser el objetivo principal del trabajo. Dicho objetivo es aprender a trabajar con el entorno orientado a objetos de Microsoft y, más concretamente, explorar el potencial que presenta esta plataforma a la hora de desarrollar páginas Web interactivas y aplicaciones de servidor.

También hacer efectivo el derecho de los ciudadanos navarros a relacionarse por medios electrónicos con la Administración Foral garantizando el cumplimiento efectivo de las obligaciones de la Ley Foral 11/2007.

Dar un impulso definitivo a las actuaciones de modernización realizadas anteriormente en la Administración Electrónica para garantizar una innovadora y profunda transformación del modelo de atención al ciudadano basado en la prestación de servicios públicos electrónicos, accesibles y multicanal. Es el eje impulsor para el desarrollo de actuaciones de transformación

y mejora interna de la Administración Foral de Navarra para el desarrollo de las competencias y procesos de forma más eficaz y eficiente. (Telecomunicaciones, 2019)

La consecuencia de esta misión y visión se aborda en este caso en modernizar el servicio de solicitud de una cédula de habitabilidad.

El proceso de tramitación de una solicitud para obtener una cédula de habitabilidad hasta este momento formaba parte de una aplicación heredada que combinaba varios trámites a la vez.

Actualmente el trámite de una cédula de habitabilidad es uno de los servicios más solicitados y consultados de forma telemática. De los 98 servicios más consultados de la página www.navarra.es la Consulta para la Expedición de la cédula de habitabilidad está en el puesto 74. Estamos ante un servicio muy demandado por la ciudadanía. (Navarra, 2020)

A todo esto, es necesario tener en consideración el número creciente de ciudadanos que tienen acceso a internet, muchos de ellos mediante dispositivos personales como pueden ser móviles (92,7%) (Noticias de Navarra, 2019). Es decir, el ciudadano debe tener la opción de realizar el mayor número de trámites posibles desde cualquier dispositivo con acceso a internet, ordenador, tableta o móvil.

La principal novedad aportada por esta aplicación es la posibilidad que ofrece de comenzar el desarrollo a partir de una arquitectura por defecto ya definida.

Objetivos:

- El objetivo principal del trabajo es alcanzar un buen conocimiento de la plataforma .NET de Microsoft, concretamente utilizando C#.
- Efectuar un desarrollo de una aplicación web partiendo de su arquitectura por defecto, utilizando estándares web modernos.
- Aprender la forma de trabajar de una empresa y utilizar sus herramientas.
- Explorar las distintas tecnologías que forman parte de la arquitectura.
- Aplicar los conocimientos adquiridos durante la carrera a un producto comercial y real.

1.3. Estado del arte:

Existen múltiples alternativas que pueden simplificar el desarrollo de una aplicación web, como pueden ser Node.js, React.js y alojar dichas aplicaciones en servicios de alojamiento o servicios de computación en la nube. Corporaciones como Amazon, Google y Microsoft dan servicio a ese mercado, ofreciendo un servicio flexible, versátil con una disponibilidad casi completa. En cambio, en las instituciones públicas desde siempre se han utilizado tecnologías probadas que tienen detrás a grandes compañías debido al carácter esencial de los servicios públicos. Además, necesitan de una mayor exigencia en la protección y seguridad de sus servicios. Por este motivo no confían en alojar sus datos en manos de terceros con el riesgo de quedar comprometidos. Por lo que éstas optan por el desarrollo de sus propias infraestructuras, compuestas por múltiples capas.

Aunque no se pueda innovar en las infraestructuras de una Administración Pública, sí que se puede hacer un cambio de paradigma a la hora de desarrollar nuevas aplicaciones para las administraciones. Cambiar las antiguas aplicaciones, gigantescas y muy complicadas, por un conjunto de aplicaciones más simples y fáciles de mantener ya que todas heredan de la misma arquitectura por defecto. Este es el concepto principal que se pretende desarrollar.

1.4. Solución propuesta:

1.4.1. Framework:

Para organizar la estructura del proyecto vamos a seguir el patrón de diseño Modelo Vista Controlador (MVC) se ha decidido utilizar un framework que implemente esta arquitectura.

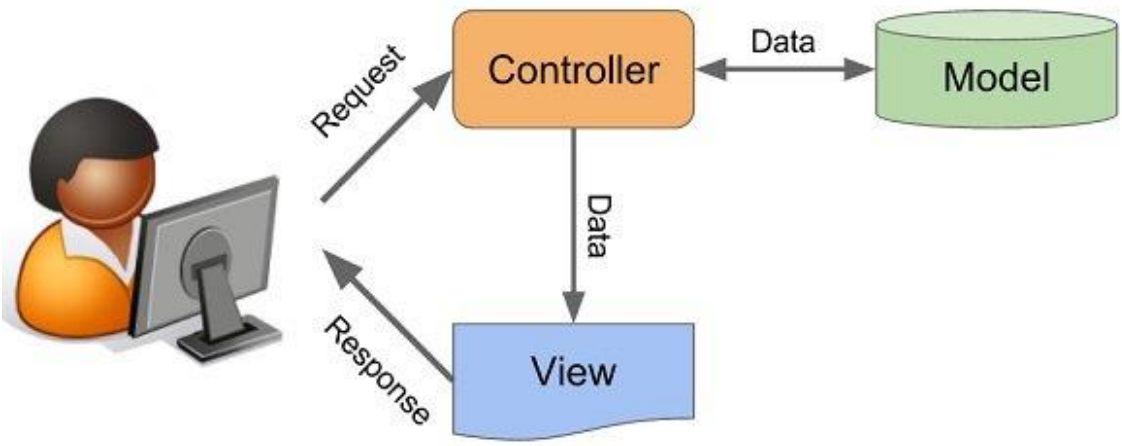


Ilustración 1:Patrón de Diseño Modelo Vista Controlador

MVC permite seguir una metodología de trabajo organizada, característica muy necesaria tanto del punto de vista del desarrollo, pero también facilita el mantenimiento y las mejoras. Se han investigado dos frameworks que cumplen este requisito que son: ASP.NET MVC5 y Java EE MVC 1.0.

1.4.1.1. Java EE MVC 1.0:

Ilustración 2:Logo Java Enterprise Edition

MVC 1.0 o JSR 371 es una framework de Java que utiliza la arquitectura MVC. Ha sido construido con componentes independientes que han sido creados por el Programa Java Community Process.

Se utiliza bajo la licencia de JCP de Oracle, es un software de libre uso con el que se pueden crear aplicaciones web.

Existe una gran comunidad internacional de programadores de todo el mundo que participan activamente en el Programa JCP. Esto garantiza la mejora continua y la continuidad del proyecto.

Se ha utilizado en variedad de productos del SSID como puede ser el catálogo de servicios y aplicaciones.

1.4.1.2. ASP.NET MVC

Ilustración 3:Logo ASP:NET MVC de Microsoft

ASP.NET MVC es un framework de aplicaciones web que implementa el patrón MVC desarrollado por Microsoft.

Posee gran cantidad de documentación y tutoriales online.

Es software de código abierto bajo licencia Apache 2.0 y aunque hoy ha sido reemplazado por ASP.NET Core, se sigue utilizando en las instituciones públicas.

1.4.1.3. Comparación:

A continuación, se puede observar en la siguiente imagen una comparación de volumen de búsqueda que hay en internet.

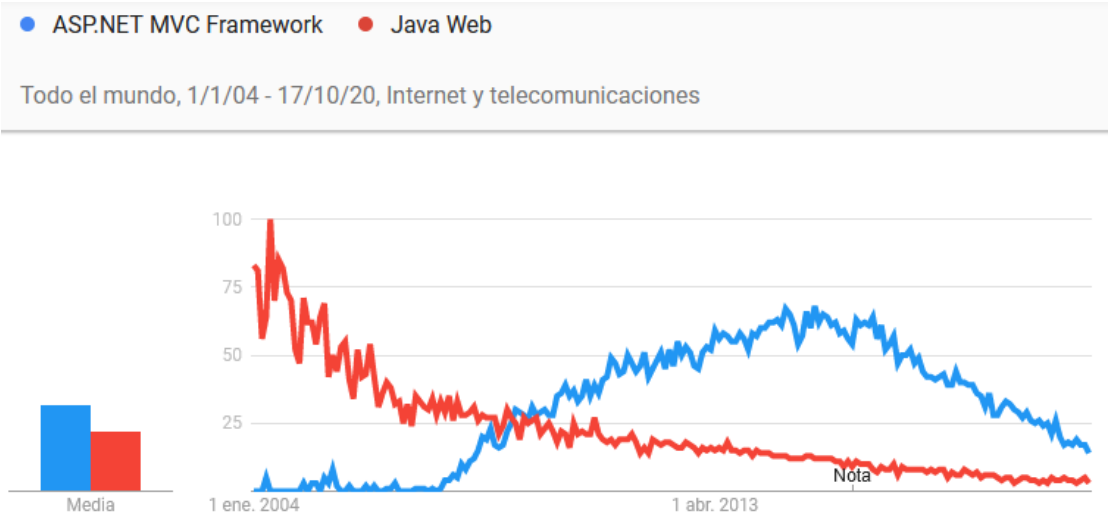


Ilustración 4:Volumen de búsqueda de ASP.NET MVC vs Java Web (Google, 2020)

Como se puede observar, en los últimos 10 años el interés por ASP.NET es mucho mayor que para Java aplicado a Web. Esto se debe básicamente a la popularidad y al boom de las páginas web, tal y como se puede apreciar en la siguiente gráfica.

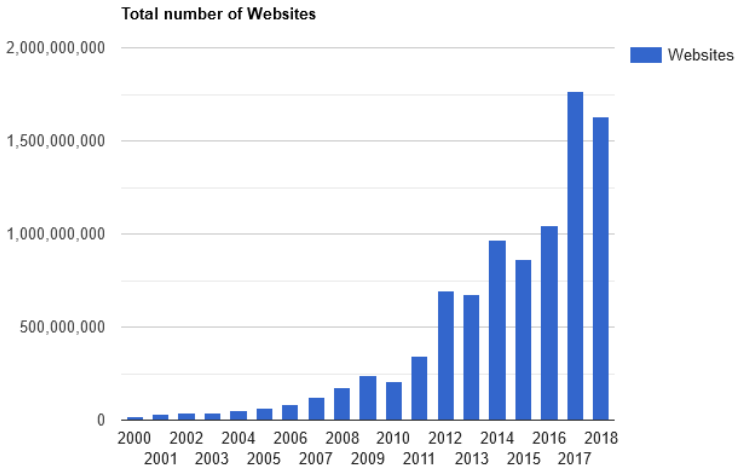


Ilustración 5:Número total de Páginas Web (Stats, 2020)

El tiempo de desarrollo de una aplicación web utilizando Java, según la experiencia de la empresa en la que se ha desarrollado este trabajo, es un orden de magnitud más grande que una aplicación web desarrollada con ASP.NET MVC. Para desarrollos de aplicaciones de tamaño medio (la mayoría) es más rápido y eficiente utilizar ASP.NET.

Los IDE de Java pueden ser Eclipse, NetBeans, JDeveloper y IntelliJ Idea, todos ellos de uso gratuito. En cambio, para ASP.NET básicamente hay que emplear Visual Studio.

1.4.1.4. Decisión Final

El SSID quiere centrarse solamente en un framework y definir en torno a él una arquitectura propia. Esta arquitectura servirá para agilizar la implementación de nuevas soluciones o realizar evoluciones. Si todos los productos parten de la misma arquitectura única, los desarrolladores tendrán una cierta familiaridad con todos los productos. El tiempo de incorporación será mínimo y podrán contribuir en varios proyectos a la vez.

Después de analizar los resultados y dado que esta aplicación web va a tener un tamaño como mucho medio se ha decidido utilizar ASP.NET como framewrok.

1.4.2. Tecnologías actuales:

Este apartado consiste en una búsqueda y un análisis de las tecnologías que mejor se adaptan a las necesidades del proyecto.

1.4.2.1. HTML5

Ilustración 6:Logotipo de HTML 5

HTML 5 es un lenguaje de programación web, es una evolución lógica del leguaje básico del WWW[2]. Surge debido a la necesidad de conseguir nuevos objetivos como, por ejemplo, la reproducción de forma nativa de vídeo o drag and drop. Esto hace que sea más fácil de usar y sea compatible con todos los dispositivos actuales.

Se ha decidido utilizar la última versión de HTML 5 respecto a su versión anterior por tener un código más sencillo, nuevos elementos y atributos que hacen mucho más rápida la carga y la navegación de las páginas web.

Hay que destacar que HTML 5 tiene compatibilidad con todos los navegadores web modernos y es una versión avanzada de HTML 4.

1.4.2.2. JavaScript

Ilustración 7:Logotipo de JavaScript

Es un lenguaje de programación interpretado, débilmente tipado, orientado a objetos que se puede modificar en cualquier editor de texto y se ejecuta en un navegador web en el lado del cliente. Su sintaxis es similar al lenguaje C con matices de Java, pero no están relacionados.

JavaScript y HTML5 hacen posible todo tipo de acciones e interacciones en un sitio web. Puede realizar la validación de campos o comunicar con el servidor a través de Ajax.

Ilustración 8:Logotipo jQuery

[2] WWW, en inglés “World Wide Web” es un sistema de distribución de documentos de hipertexto o hipermedia interconectados y accesibles a través de Internet

La librería más popular que se utiliza junto con JavaScript en una página web es JQuery. JQuery se utiliza en un 76,5% de las páginas web (w3Techs, 2020) y sirve para reducir el código de JavaScript, realizar validaciones y llevar el control de sistemas de plantillas, etc.

1.4.2.3. Framework CSS:

Se realizó una búsqueda de un framework CSS para la creación del diseño del front-end de la aplicación.

Los dos frameworks más populares son Bootstrap 3 y Foundation.

Ilustración 9: Logotipo Bootstrap 3

Bootstrap 3 es un framework de código abierto creado por Twitter que introdujo el diseño receptivo a gran escala. Es decir, la página web se adapta al tamaño de cualquier pantalla de forma automática. Permite crear interfaces web con CSS y JavaScript.

Ilustración 10.Logotipo Foundation

Foundation también es un framework que es de código abierto desde 2011 creado por ZURB y dispone de una gran cantidad de herramientas y opciones de personalización.

Ambos frameworks disponen de una documentación amplia y numerosas plantillas con ejemplos de diseño responsive. El diseño responsive o adaptativo es una técnica de diseño que busca la correcta visualización de una misma página web en distintos dispositivos. Ambos frameworks cumplen con los requisitos del proyecto.

Sin embargo, se ha decidido utilizar BootStrap porque posee componentes que simplemente se utilizan ya que no nos interesa personalizar demasiado nuestra aplicación.

1.4.2.4. Base de datos:

Para la base de datos es necesario utilizar la base de datos heredada que se encuentra en SQL Server 2012.

Ilustración 11:Logotipo SQL Server 2012

La elección de todas estas tecnologías se puede unir en un solo proyecto básico y crear la arquitectura por defecto.

1.5. Metodología empleada:

El método empleado para la realización del trabajo ha venido marcado por las limitaciones del tiempo, la realización de un plan de acogida para incorporarme cuanto antes al equipo y la necesidad de cumplir los plazos de finalización de tareas.

Este hecho ha provocado la decisión de dividir el proyecto en tres grandes bloques o tareas:

- Análisis y diseño de la aplicación.
- Implementación de la interfaz de usuario.
- Implementación de las funcionalidades de la aplicación.

Dentro de estos tres bloques se ha seguido un proceso paralelo en su realización debido a la existencia de siete tipos de usuarios muy diferenciados por las tareas que realizan en la aplicación: Los Gestores de Vivienda, Los Técnicos de Vivienda, Los Gestores ORVE, Los Técnicos ORVE, Administrador, el Director de Servicio y el Ciudadano.

Los Ciudadanos a grandes rasgos solo pueden realizar solicitudes y ver sus solicitudes. El director de servicio da el visto bueno final de las solicitudes y gestiona los usuarios del back-office. Los Gestores de ORVE (Oficina de Rehabilitación de Viviendas y Edificios) se encargan de comprobar las reformas de las rehabilitaciones realizadas. Los Gestores Vivienda comprueban los datos de las solicitudes, rellenar nuevas y validar los datos. Los técnicos se encargan de redactar los informes y subirlos a sus correspondientes pestañas de la solicitud. El Administrador tiene acceso a casi toda la aplicación.

Por ello el orden seguido de cada uno de los bloques definidos anteriormente ha sido:

I. Tareas relacionadas con los usuarios del back-office
II. Tareas relacionadas con los Ciudadanos

Esto ha facilitado la etapa de pruebas realizadas en cada una de las implementaciones. Ya que básicamente todos los usuarios Administrativos, estaban trabajando sobre las mismas vistas, solo que se deshabilitaban según el perfil aquellas secciones que no tenían permiso para visualizar. También se podían insertar datos desde la aplicación sin la necesidad de trabajar desde el servidor de la base de datos.

2. Análisis:

2.1. Visión general:

En primer lugar, hay que tener en cuenta que los usuarios finales de la parte pública son un grupo totalmente heterogéneo por lo que la usabilidad de la interfaz debe ser lo más clara y fácil de usar posible. Aprovecharemos las vistas comunes de los Gestores con los Ciudadanos para crear una vista paralela para los Gestores que contengan todas las opciones de los Ciudadanos más algunas opciones solo relevantes para los Gestores. Por lo tanto, no hay dos aplicaciones separadas, una para la parte publica y otra para la parte administrativa, sino que es un único proyecto de frontal que se dividirá en dos.

La fase completa de análisis se ha llevado a cabo por los desarrolladores Enrique Sierra y Oscar Baquedano. La fase de análisis comienza tras realizar el análisis de viabilidad y fijar un alcance. Las historias de usuario y los requisitos se han obtenido a través de las reuniones iniciales con el equipo de Vivienda y representantes de las ORVE. El desglose de las historias de usuario en tareas se lleva a cabo en un documento EDT (Estructura de descomposición de trabajos) y un DT (Diseño técnico).

2.2. Requisitos funcionales:

A continuación, se indican los requisitos funcionales que ha de cumplir la aplicación agrupados según el tipo de usuario que necesitará dichas funcionalidades:

Ciudadano:

- Completar una solicitud de cédula: permite completar los datos necesarios tanto como propietario, pero también como representante de alguien y subir los documentos que considere oportunos. Al final puede descargar la solicitud de cédula.
- Consultar todas las solicitudes que ha realizado: permite que el ciudadano pueda ver todas las solicitudes sin importar el estado en el que se encuentren.
- Modificar los datos de Solicitante de una solicitud ya realizada: permite al ciudadano modificar la forma de notificación o añadir más documentos si así lo considera oportuno el Gestor de Vivienda u ORVE.
- Descargar la cédula de habitabilidad una vez que el trámite haya finalizado exitosamente.

Gestor Vivienda:

- Completar una solicitud de cédula: permite completar los datos necesarios y subir los documentos que considere oportunos. Al final puede descargar la solicitud de cédula.
- Modificar los datos de una solicitud en cualquier momento salvo si la cédula ya ha sido emitida.
- Descargar la cédula de habitabilidad una vez que el trámite haya finalizado exitosamente.
- Acceder a su bandeja de entrada: permite al Gestor de Vivienda, acceder a las nuevas solicitudes generadas por los ciudadanos. Bandeja común a todos los Gestores de Vivienda.
- Dar el visto bueno y pasar el expediente al Gestor ORVE, al Técnico de Vivienda o al Director de Servicio o no estar conforme y devolverla al ciudadano para más cambios.

- Buscar expedientes: permite realizar la búsqueda por una serie de campos como pueden ser número de solicitud o número de cédula de todas las solicitudes de la aplicación.

Gestor ORVE:

- Acceder a su bandeja de entrada: permite al Gestor ORVE acceder a las nuevas solicitudes que el Gestor Vivienda le está añadiendo. Los Gestores ORVE tienen una bandeja común al área territorial a la que pertenecen.
- Dar el visto bueno y pasar el expediente al Técnico ORVE o al Director de Servicio o no estar conforme y devolver el expediente al Gestor de Vivienda.
- Ver todo el expediente, pero solo poder modificar la pestaña ORVE.

Técnico Vivienda:

- Acceder a su bandeja de entrada: permite al Técnico Vivienda, acceder a las nuevas solicitudes generadas por los ciudadanos. Bandeja común a todos los Técnico Vivienda.
- Ver todo el expediente, pero solo poder modificar la pestaña informe Técnico Vivienda y añadir el informe.
- Dar el visto bueno y pasar el expediente al Gestor Vivienda o no estar conforme y devolver el expediente al Gestor de Vivienda.

Técnico ORVE:

- Acceder a su bandeja de entrada: permite al Técnico ORVE, acceder a las nuevas solicitudes generadas por los ciudadanos. Bandeja común a todos los Técnicos de ORVE de la misma zona territorial.
- Ver todo el expediente, pero solo poder modificar la pestaña informe Técnico ORVE y añadir el informe.
- Dar el visto bueno y pasar el expediente al Gestor ORVE o no estar conforme y devolver el expediente al Gestor de ORVE.

Administrador:

- Puede ver todas las vistas y pestañas
- Tiene el mismo nivel de acceso que el Gestor de Vivienda
- Gestiona en una vista propia los usuarios administrativos.

Director de servicio:

- Acceder a su bandeja de entrada: permite al director de servicio, acceder a las solicitudes propuestas para emitir la cédula.
- Ver todo el expediente y puede modificar cualquier campo.
- Gestiona en una vista propia los usuarios administrativos.
- Dar el visto bueno para Emitir la Cédula o la devuelve a uno de los administrativos o directamente la deniega.

2.3. Requisitos no funcionales:

A continuación, se van a enumerar los requisitos no funcionales de la aplicación:

- El ingreso al sistema estará restringido bajo el servicio de clave

    - El servicio clave junto a CAR son los métodos de acceso propios dentro de la administración pública.
- La página web deberá funcionar en los navegadores Firefox 70 e Internet Explorer 11 puesto que son los únicos navegadores web con los que trabaja la Administración Pública.
- La página web se debe ajustar dinámicamente al tamaño de pantalla del usuario, independientemente del dispositivo.
    - La página web necesita tener un comportamiento responsive.
- El sistema no presentará problemas para su manejo e implementación.
- La aplicación debe utilizar la misma base de datos que la aplicación antigua, pero si es necesario, se pueden añadir nuevas tablas o columnas.

2.4. Requisitos de la interfaz de usuario

2.4.1. Requisitos:

- La aplicación debe estar disponible tanto en castellano como en euskera.
    - Para cumplir con el DECRETO FORAL 103/2017 donde se regula el uso del euskera en las administraciones públicas de Navarra.
- La aplicación debe seguir el manual de estilo definido por el Gobierno de Navarra.
    - Todas las aplicaciones web de la administración pública deben tener una apariencia homogénea.

2.4.2. Identificación de las restricciones técnicas:

Es necesario tener en cuenta que esta aplicación web ha de ser usada por todo tipo de usuarios utilizando conexiones de velocidad totalmente variable. Por lo tanto, es de vital importancia optimizar el apartado gráfico, así como la información de ida y vuelta al servidor para que no resulte excesivamente lenta la navegación de la aplicación.

2.4.3. Perfiles de usuario:

Es necesario hacer una distinción entre los siete tipos de usuarios que usarán la aplicación.

Ciudadano: Es el único usuario de la parte pública de la aplicación. Son usuarios básicamente pasivos, en el sentido que después de rellenar la solicitud, solo pueden estar consultando el estado en el que se encuentran las solicitudes y descargar la documentación.

Gestor Vivienda: Son los usuarios mayoritarios de la parte administrativa de la aplicación, son los encargados de gestionar la entrada de solicitudes en la aplicación, comprobar la veracidad de los datos y realizar un seguimiento de estas.

Técnico Vivienda: Son usuarios que tienen un acceso limitado a la aplicación, solo se encargan de subir los informes realizados tras la inspección del inmueble.

Gestor ORVE: Usuario encargado de comprobar si se dispone de suficiente información para afirmar que se han realizado obras en la vivienda. Tiene un acceso limitado a la aplicación. Solo tiene acceso a la pestaña ORVE de la solicitud.

Técnico ORVE: Es el profesional encargado de comprobar que las obras de mejora que se han realizado en la vivienda coinciden con la declaración del solicitante. Se encarga de subir su informe al expediente solamente.

Director Servicio: Es un único usuario que tiene el control de la aplicación y es el responsable final de conceder una cédula. También es el encargado de gestionar las cuentas de los demás administrativos.

Administrador: Es la cuenta que van a utilizar los desarrolladores cuando necesitan comprobar el funcionamiento correcto en la aplicación dentro del servidor en producción. A nivel de permisos este posee los mismos que el director de servicio. Menos el acceso a datos sensibles como pueden ser nóminas.

2.5. Casos de uso:

A partir de los requisitos se van a analizar solo los casos de uso que he implementado como desarrollador frontend y los necesarios para dar un contexto mínimo.

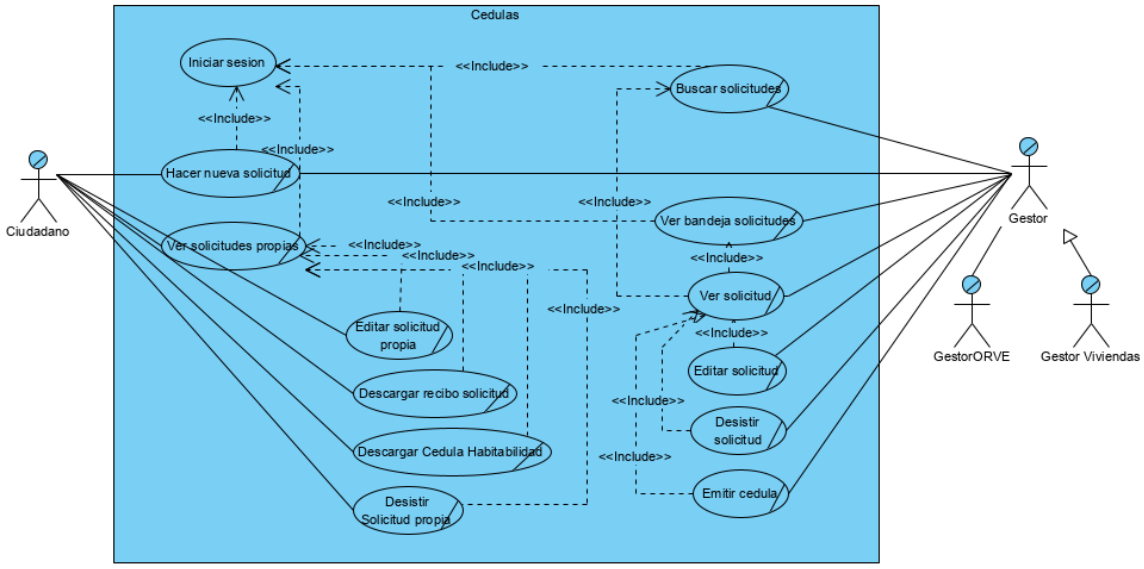


Ilustración 12:Diagrama parcial de los casos de uso

Caso de uso número 1: "Iniciar Sesión"

Resumen de la funcionalidad: Identifica al usuario mediante un nombre de usuario y una contraseña

Actores: **Ciudadano, Gestor Vivienda, Gestor ORVE, Técnico Vivienda, Técnico ORVE, Administrador, Director Servicio.**

Casos de uso relacionados: Servicio de identificación externo.

Precondiciones: El usuario debe estar dado de alta en clave y CAR

Postcondiciones: Se crea un objeto usuario cargando los datos de la base de datos. Se autentifica al usuario.

El usuario introduce su nombre de usuario y su contraseña, ambos son validados. Se carga la pantalla principal con el menú correspondiente según el tipo de usuario.

Caso de uso número 2: "Hacer nueva Solicitud"

Resumen de la funcionalidad: Se crea una nueva solicitud para la cédula de habitabilidad y se almacena en la base de datos.

Actores: **Ciudadano, Gestor Vivienda.**

Casos de uso relacionados: **Iniciar Sesión**

Precondiciones: El usuario debe estar registrado y que no exista un trámite en curso para esa vivienda o que la cédula de habitabilidad existente tiene una validez mayor de 2 meses.

Postcondiciones: Se crea un objeto Solicitud, se completan los datos de vivienda y catastro mediante los servicios externos y se guarda en la base de datos.

Caso de uso número 3: "Ver solicitudes propias"

Resumen de la funcionalidad: Se presenta al usuario un listado de las solicitudes que ha realizado

Actores: **Ciudadano**

Casos de uso relacionados: **Iniciar Sesión**

Precondiciones: Ninguna

Postcondiciones: Se presenta al usuario el listado de las solicitudes que ha realizado, en caso de no haber realizado ninguna, la página aparece vacía.

Caso de uso número 4: "Editar Solicitud propia"

Resumen de la funcionalidad: El usuario cliente realiza una modificación a una solicitud ya presentada.

Actores: **Ciudadano**

Casos de uso relacionados: **Ver solicitudes propias**

Precondiciones: El usuario debe tener al menos una solicitud realizada, sin importar el estado en el que se encuentre y que haya seleccionado una de ellas del listado de solicitudes.

Postcondiciones: El ciudadano realiza los cambios permitidos y pincha en el botón guardar Cambios. Si las modificaciones están correctas se guardan los cambios y se muestra un mensaje de confirmación.

Caso de uso número 5: "Descargar recibo solicitud"

Resumen de la funcionalidad: El usuario descarga el recibo que atesta que la solicitud se ha realizado.

Actores: **Ciudadano**

Casos de uso relacionados: **Ver solicitudes propias, Hacer nueva Solicitud**

Precondiciones: El usuario debe tener al menos una solicitud realizada, sin importar el estado en el que se encuentre.

Postcondiciones: En la pestaña resumen Solicitud el usuario puede descargar el PDF con los detalles de su solicitud.

Caso de uso número 6: "Descargar Cédula Habitabilidad"

Resumen de la funcionalidad: El usuario descarga la cédula de habitabilidad.

Actores: **Ciudadano**

Casos de uso relacionados: **Ver solicitudes propias**

Precondiciones: El usuario debe tener al menos una solicitud con el estado Emitida.

Postcondiciones: En la pestaña resumen Cédula el usuario puede descargar la cédula de habitabilidad en formato PDF.

Caso de uso número 7: "Desistir solicitud"

Resumen de la funcionalidad: El usuario no quiere seguir con el tramite

Actores: **Ciudadano**

Casos de uso relacionados: **Ver solicitudes propias**

Precondiciones: El usuario debe tener al menos una solicitud que no tenga el estado Emitida o Desistida o que haya sido revisada por el Gestor de Vivienda.

Postcondiciones: El usuario puedes desistir la solicitud en cualquier momento pulsando un botón.

La solicitud no se borra, se le cambia el estado a Desistida y desaparece para los Administrativos.

Caso de uso número 8: "Buscar solicitudes"

Resumen de la funcionalidad: El administrativo completa una serie de campos para filtrar los expedientes.

Actores: **Administrativos**

Casos de uso relacionados: **Iniciar Sesión**

Precondiciones: ninguna

Postcondiciones: El Administrativo obtiene una lista de expedientes de la base de datos según los filtros aplicados.

Los Administradores ORVE solo pueden buscar expedientes dentro de su ORVE.

Caso de uso número 9: “Ver bandeja solicitudes”

Resumen de la funcionalidad: El administrativo puede consultar su bandeja propia de entrada.

Actores: **Administrativos**

Casos de uso relacionados: **Iniciar Sesión**

Precondiciones: ninguna

Postcondiciones: El Administrativo obtiene una lista de expedientes pendientes de ser examinados por dicho usuario. Los usuarios del mismo tipo comparten la misma bandeja de entrada.

Los Administradores ORVE solo pueden ver los expedientes propios de su región territorial y también comparten bandeja con los administradores de este ORVE.

Caso de uso número 10: “Ver solicitud”

Resumen de la funcionalidad: El administrativo puede consultar una solicitud

Actores: **Administrativos**

Casos de uso relacionados: **Ver bandeja solicitudes, Buscar solicitudes**

Precondiciones: Obtener una lista de solicitudes, bien de la bandeja de entrada o bien como resultado de una búsqueda de solicitudes.

Postcondiciones: El Administrativo obtiene la visualización de la solicitud que quiere acceder, pero solo puede ver aquellos campos y pestañas a las que tiene permiso.

Caso de uso número 11: “Editar Solicitud”

Resumen de la funcionalidad: El administrativo puede modificar una solicitud.

Actores: **Administrativos**

Casos de uso relacionados: **Ver bandeja solicitudes, Buscar solicitudes**

Precondiciones: El Administrativo ha accedido a una solicitud en curso que no tenga el estado Emitido o Desistido.

Postcondiciones: El Administrativo realiza los cambios permitidos y pincha en el botón guardar Cambios. Si las modificaciones están correctas se guardan los cambios y se muestra un mensaje de confirmación.

Caso de uso número 12: “Desistir Solicitud”

Resumen de la funcionalidad: El Gestor Vivienda o El director de servicio puede desistir una solicitud si encuentra una anomalía no resuelta.

Actores: **El Gestor Vivienda, El director de servicio**

Casos de uso relacionados: **Ver bandeja solicitudes, Buscar solicitudes**

Precondiciones: El usuario ha accedido a una solicitud en curso que no tenga el estado Emitido o Desistido.

Postcondiciones: El usuario pincha en el botón de Desistir solicitud. Se le cambia el estado de la solicitud a Desistido y se guardan los cambios en la base de datos.

Caso de uso número 13: "Emitir Cédula"

Resumen de la funcionalidad: El Gestor Vivienda o el director de servicio puede emitir la cédula de habitabilidad si esta cumple los requisitos establecidos en la normativa.

Actores: **El Gestor Vivienda, El director de servicio**

Casos de uso relacionados: **Ver bandeja solicitudes, Buscar solicitudes**

Precondiciones: La vivienda y la solicitud cumple con los criterios necesarios para poder obtener la cédula de habitabilidad. El usuario accede a la solicitud de dicha cédula.

Postcondiciones: El usuario pincha en el botón de Emitir Cédula y se genera la nueva cédula, se guarda el documento en el servidor y se cambia el estado de la solicitud a Emitida.

3. Diseño:

A continuación, voy a explicar la motivación detrás de la necesidad de crear una arquitectura por defecto.

3.1. ¿Por qué definir una arquitectura?

La necesidad de definir una arquitectura por defecto surge a partir de la detección de una serie de problemas que han ido ocurriendo tras numerosos proyectos desarrollados en la empresa.

Se ha detectado que la mayoría de los proyectos que se estaban implementando reunían gran cantidad de criterios comunes y además utilizaban en general los mismos componentes. Sin embargo, como no había ningún estándar establecido, cada equipo de desarrollo organizaba la aplicación de forma diferente. Esto hacia muy difícil el cambio de los desarrolladores de un proyecto a otro ya que necesitaban conocer las particularidades de cada proyecto en el que estaban participando y también el mantenimiento de dicha aplicación si el equipo original ya no se encontraba en la empresa.

La arquitectura por defecto permite a los proyectos que tienen una misma estructura tener un mejor mantenimiento ya que los proyectos resultan más familiares a los otros desarrolladores. Ya no existe la dependencia del desarrollador.

Utilizando una arquitectura, el desarrollo es mucho más rápido ya que los componentes más frecuentes ya se encuentran presentes. Los componentes más frecuentes suelen ser el acceso a la base de datos, logging y servicios.

La estructura modular y desacoplada de los componentes añaden una serie de facilidades que hacen que el cambio y la extensibilidad de módulos y clases sea muy rápido y sencillo.

3.2. Diseño guiado por el dominio:

Arquitectura diseñada para aplicaciones complejas con un volumen importante de lógica de negocio. Aplicaciones con una vida larga y mantenimiento donde se quiere aislar y proteger la Capa de Dominio de las tecnologías concretas[3].

Define pautas importantes en el desarrollo de una aplicación como pueden ser:

- **Arquitectura de N-capas:** Se separan las responsabilidades y se pueden administrar las dependencias. Cada capa tiene una responsabilidad específica. Una capa superior puede utilizar los servicios de una capa inferior, pero no al revés. Facilita el intercambio de porciones de la aplicación sin modificar el resto de la aplicación.
- Patrones de diseño:
    - **Repository:** patrón que separa la lógica de negocio de la capa de origen de los datos. actúa como una colección de objetos de dominio en memoria.
    - **Entity:** Permite trabajar con los datos utilizando los objetos de dominio abstrayéndose de como se guarden esos datos en la base de datos.
    - **Services:** Utiliza estándares de presentación la información en formato texto (ejemplo XML) y permite la comunicación entre componentes escritos en diferentes lenguajes y entornos.
    - …

[3]Explicación tomada del libro “Architecting Modern Web Applications with ASP.NET Core and Microsoft Azure” https://dotnet.microsoft.com/download/e-book/aspnet/pdf

- **Desacoplamiento entre componentes:** Un bajo acoplamiento entre las unidades de software mejora la mantenibilidad de las unidades de software, aumenta la reutilización de dichas unidades y disminuye el riesgo de cambiar múltiples unidades de software cuando se quiere alterar una.

3.3. Descripción de la arquitectura:

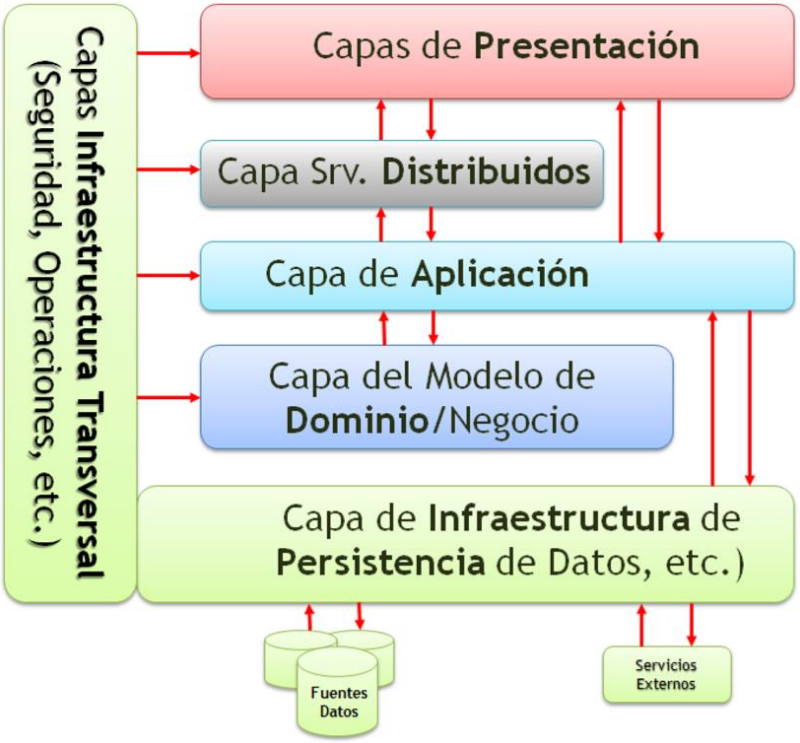


Ilustración 13:Vista de arquitectura lógica simplificada de un sistema de N-capas DDD (Iberica, Microsoft, s.f.)

La arquitectura está organizada en torno a una jerarquía de carpetas numeradas:

- Los números más bajos contienen la arquitectura básica de referencia que durante el desarrollo normal no se cambia.
- Los demás definen el lugar donde se debe guardar los diferentes ficheros que se han desarrollado para dicho proyecto.

Breve descripción de la arquitectura básica de referencia:

- Carpeta 0.1 – Solution items:
  - Repositorio de archivos comunes (Librerias, SQL, GesIntegracion,etc)
- Carpeta 0.2 – Infrastructure.Ssid:
  - Clase para configurar el acceso a la base de datos.
  - Repositorio de controles comunes para la capa de presentación.
  - Contiene el núcleo base de la arquitectura de referencia.

- Carpeta 0.3 – Infrastructure.Ssid.<Nombre Proyecto>:
    - Configura el contenedor de dependencias para poder utilizarlo desde cualquier composición root, siendo este un frontal web o aplicación de consola.
    - También agrupa los recursos compartidos por diferentes proyectos, como puede ser un mensaje de cierre de sesión.

  Breve descripción de los ficheros propios del proyecto en sí:

- Carpeta 1.0 – Tests
    - Contiene las pruebas unitarias y de integración.
- Carpeta 2.0 – Database
    - Proyecto Database
- Carpeta 3.0 – Dal
    - Proyecto de acceso a datos
    - Interfaz:Proyecto de contrato de acceso a base de datos
    - Repositorios: proyecto de implementación de las interfaces
- Carpeta 4.0 – Bll
    - Capa que contiene la lógica de negocio
    - Entidades, servicios externos, servicios…
- Carpeta 5.0 – External Service
    - Capa para exponer servicios a clientes externos
    - En este proyecto no se va a emplear
- Carpeta 6.0 – Web
    - Contiene los proyectos de presentación.

3.4. Diseño de la implementación de los casos de uso:

Caso de uso número 2: “Hacer nueva Solicitud”

Tarea: Pestaña Solicitante.

La pestaña Solicitante como su nombre indica, contiene todos los datos necesarios del ciudadano que está pidiendo la cédula.

La pestaña solicitante contiene todos los campos relacionados con los datos personales de la entidad que quiere realizar el trámite siendo bien una entidad Física o Jurídica y si actúa como Propietario o como Representante de uno.

Campos:

- Campos comunes:
    - Desplegable para el tipo de solicitante: Propietario/ Representante
    - Desplegable para la Entidad: Físico/ Jurídico
    - Check de la Autorización para notificar Telemáticamente: Es un checkbox que simula la autorización que otorga el cliente para ser notificado a la dirección de correo que ha proporcionado.
    - Campo NIF: Número de identificación Fiscal
    - Campo Teléfono
    - Campo Email: Correo Electrónico

- Campos que aparecen si el solicitante es Representante:
    - Aparece un fragmento abajo del todo con el título: Dato del propietario al que representa
    - Campo NIF
    - Campo Nombre
    - Campo Primer Apellido
    - Campo Segundo Apellido

- Campos que aparecen si la entidad es Físico:
    - Campo Nombre
    - Campo Primer Apellido
    - Campo Segundo Apellido
    - Aparece un fragmento donde se puede elegir el tipo de notificación:
        - Radio Button Notificación a la dirección postal de la vivienda
            - Campo Dirección solo en modo lectura.
        - Radio Button Notificación a otra dirección postal
            - Campo Dirección
            - Campo CP: Código Postal
            - Campo Provincia

- Campos que aparecen si la entidad es Jurídico:
    - Campo Razón Social

Comportamiento: El cliente puede modificar todos los campos de la pestaña

- A nivel de Campo:
    - El check Autorizo Notificación Telemática controla el campo Email.
        - Si está marcado, el campo Email puede ser cambiado
        - Si no está marcado, se limpia el campo Email y se deshabilita
    - El check también controla el fragmento Notificaciones
        - Si este marcado se ocultan otras formas de notificación
        - Si no está marcado se muestra dicho fragmento.
- El campo NIF.
    - En cuanto el valor de este campo cambia se hace una petición a la base de datos para que nos devuelva (si existe) los datos de solicitante más recientes de los que se tiene constancia para dicho NIF
    - Esta acción sobrescribe los campos:
        - Teléfono
        - Email (Solo si está habilitado)
        - Nombre/Razón Social
        - Primer Apellido
        - Segundo Apellido

Validaciones:

- A nivel de campo:

- Campo teléfono solo puede contener hasta 9 cifras
- Campo código postal solo puede contener hasta 5 cifras

En este caso, la vista estaba desarrollada solo con la funcionalidad básica de los desplegables y los campos de texto. Esta tarea ha consistido en añadir las reglas de negocio.

La primera subtarea ha sido sacar de las opciones de notificación el Teléfono y añadirlo como campo común.

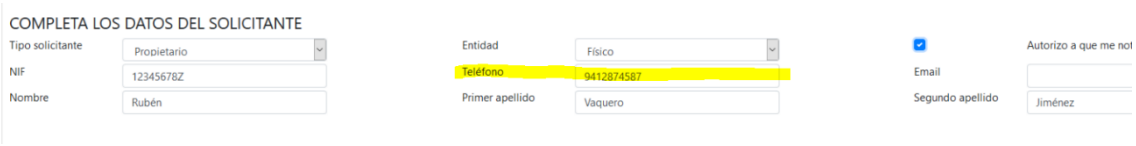


Ilustración 14:Fragmento pestaña Solicitante, datos obligatorios

A continuación, hacer que se rellene obligatoriamente y que solo admitan números y tenga una longitud máxima de 9 dígitos.

```
<div class="col-md-1">
    <label id="lblTelefonoNotificaciones" for="txtTelefonoNotificaciones">@Html.Raw(Resource.Telefono)</label>
</div>
<div class="col-md-2">
    @Html.TextBoxFor(_ => Model.Solicitante.Telefono, Model.Solicitante.IsReadOnly ? (object)new { @type = "tel", @maxlength = "9",
    @Html.ValidationMessageFor(_ => Model.Solicitante.Telefono)
```

Ilustración 15:Validaciones de campo Teléfono

Además, existe un fragmento de javascript que comprueba la longitud y el tipo de datos de la celda telefono.

```
/*** Mascaras de campos ***/
$(function () {
    $('.mascaraTelefono').mask("999999999");
    $('.mascaraCP').mask("99999");
});
```

Ilustración 16: Mascaras de validación de campos en el cliente, Se valida el formato del campo Teléfono y Código postal

Cuando se da a enviar solicitud, se hace una llamada a SolicitanteViewModelValidator donde hemos creado la regla que obliga a tener el telefono rellenado, en caso contario añade un mensaje de error a la lista de errores.

```
this.RuleFor(model => model.Telefono).NotEmpty().WithMessage(Resource.MsgValidarTelefonoVacio);
```

Ilustración 17:Regla de validación del campo Teléfono en el Servidor

Se realiza el mismo procedimiento para todos los campos obligatorios que son de texto, como pueden ser los campos Nombre, Código Postal, NIF y Primer Apellido (Solo cuando es Entidad Física).

La segunda subtarea: Ocultar los campos Primer Apellido y Segundo Apellido y cambiar el nombre del campo Nombre a Razón Social cuando el tipo de Entidad es Jurídico.

```
//Nombre o razon social(Si es Juridico)
var solicitanteNombre = $("#Solicitante_Nombre").val();
var idRazonSocial = $("#AdminAjaxRazonSocialVacio");
var idnombre = $("#AdminAjaxNombreVacio");
if ($("#cmbTipoRepresentante").val() == "Juridica") {
    idRazonSocial[0].hidden = (solicitanteNombre == "") ? false : true;
} else {
    idnombre[0].hidden = (solicitanteNombre == "") ? false : true;
}
```

Ilustración 18: JavaScript que controla el comportamiento de los componentes a mostrar en base al tipo Representante

```
//Persona Juridica vinculado con las notificaciones
//Tipo representante
var tipoRepresentante = $("#cmbTipoRepresentante").val()

//Apellido1
var apellido1 = $("#Solicitante_Apellido1").val();
//Mascara
var idapellido1 = $('#AdminAjaxApellido1');
//Control de Mascara
if (tipoRepresentante == "Fisica") {//Persona Fisica
    idapellido1[0].hidden = (apellido1 == "") ? false : true;
} else {
    //Ocultamos mensajes el apellido1 y notificaciones
    idapellido1[0].hidden = true;
}
```

Ilustración 18: Segunda parte del JavaScript

Tercera subtarea: Si la persona es Representante entonces validar los campos NIF, Nombre y Primer Apellido del representado.

```
//Si es Representante debemos validar que ha puesto Nif, Nombre y Primer apellido del representante
var tipoSolicitante = $("#cmbTipoSolicitante").val()
if (tipoSolicitante == "Representante") {

    //Mascaras
    var idNifRepresentado = $('#AdminAjaxNifRepresentado');
    var idNombreRepresentado = $('#AdminAjaxNombreRepresentado');
    var idApellido1Representado = $('#AdminAjaxApellido1Representado');

    idNifRepresentado[0].hidden = ($("#Solicitante_NIFRepresentado").val() == "") ? false : true;
    idNombreRepresentado[0].hidden = ($("#Solicitante_NombreRepresentado").val() == "") ? false : true;
    idApellido1Representado[0].hidden = ($("#Solicitante_Apellido1Representado").val() == "") ? false : true;

}
```

Ilustración 19:Validación de los campos NIF, Nombre y Apellido

Comprobar que el NIF del solicitante y del representado no es el mismo.

```
this.RuleFor(model => model.NIFRepresentado).NotEmpty().When(model => model.IdTipoSolicitante == SolicitanteViewModel.TipoSolicitante.Representante).WithMes
//TODO: validar formato NIF?
this.RuleFor(model => model.NombreRepresentado).NotEmpty().When(model => model.IdTipoSolicitante == SolicitanteViewModel.TipoSolicitante.Representante).Withi
this.RuleFor(model => model.Apellido1Representado).NotEmpty().When(model => model.IdTipoSolicitante == SolicitanteViewModel.TipoSolicitante.Representante).W
```

Ilustración 20: Reglas de validación del campo NIF en el Servidor

```
//Comprobamos si el nif del solicitante y del representado son iguales en tal caso informamos que no pueden ser iguales
//Mascaras
var idNifRepresentadoIgualNifSolicitante = $('#AdminAjaxNifRepresentadoIgualNifSolicitante');
var idNifRepresentateFormatError = $('#AdminAjaxNifRepresentadoNoEsNifValido');
var nifRepresentado = $("#Solicitante_NIFRepresentado").val();

idNifRepresentadoIgualNifSolicitante[0].hidden = (nifSolicitante.trim() == nifRepresentado.trim()) ? false : true;
idNifRepresentateFormatError[0].hidden = (nifRepresentado.trim() != "" && resultado.ErrorMessages.includes('NIF Represent
```

Ilustración 21:Evitar que el NIF del representante y del solicitante no sea el mismo

Se realiza la misma comprobacion una vez enviada la solicitud.

Una ejecucion para comprobar que el mensaje se recibe y ademas que las validaciones funcionan.

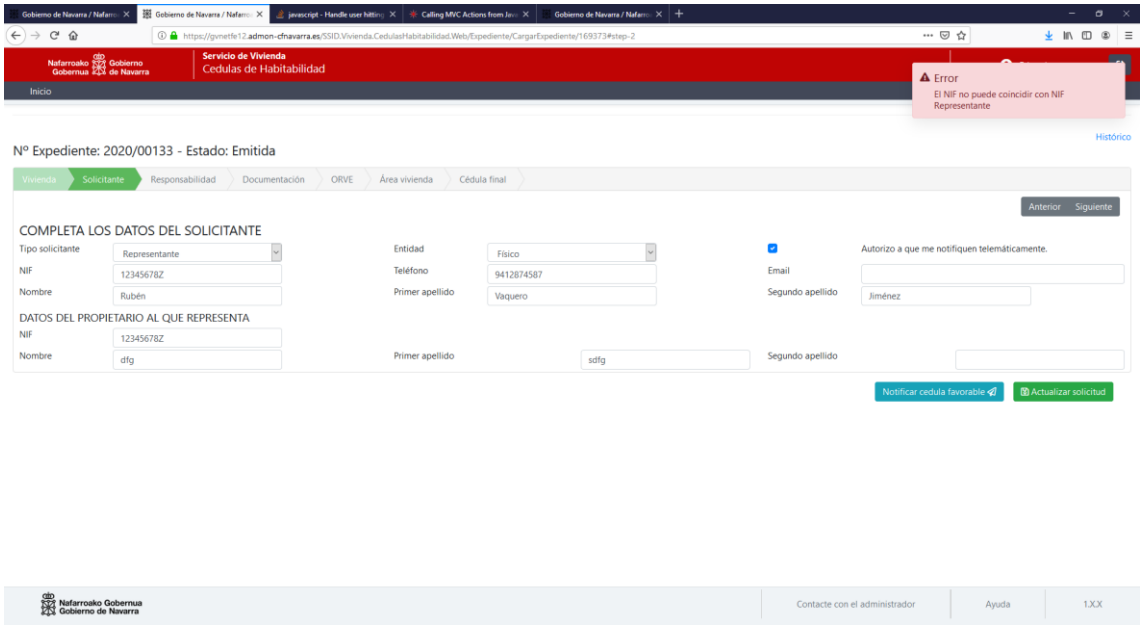


Ilustración 22:Mensaje Validación NIF

Cuarta subtarea: Validar NIF y si se encuentra el solicitante en la base de datos poblar los campos correspondientes.

```
$("input[name='Solicitante.NIF']").change(function () {
    try {
        //llamada a procedimiento para obtener los datos
        var accionURL = '@Url.Action("ObtenerDatosSolicitantePorNif", "Expediente", new { nif = "param-nif" })';
        var nif = $("input[name='Solicitante.NIF']").val();
        accionURL = accionURL.replace("param-nif", nif);
        if (nif) {
            $.post(accionURL).done(function (data) {
                if (data.Success == true) {
                    //si todo bien
                    //Oculto mensaje nif, email y nombre o razon social
                    var idValidoFormatNif = $("#AdminAjaxFormatoNif");
                    idValidoFormatNif[0].hidden = true;
                    //Email
                    var idAdminVacioMail = $("#AdminAjaxEmailVacio");
                    idAdminVacioMail[0].hidden = true;
                    //Nombre o razon social
                    var idNombre = $("#AdminAjaxNombreVacio");
                    idNombre[0].hidden = true;
                    var idRazonSocial = $("#AdminAjaxRazonSocialVacio");
                    idRazonSocial[0].hidden = true;


                    //Parte que rellena los campos correspondiente
                    $("input[name='Solicitante.Telefono']").val(data.Telefono);
                    $("input[name='Solicitante.Telefono']").focus();
                    $("input[name='Solicitante.Telefono']").blur();

                    if (document.getElementById('Solicitante_Email').disabled == false) {
                        $("input[name='Solicitante.Email']").val(data.Email);
                        $("input[name='Solicitante.Email']").focus();
                        $("input[name='Solicitante.Email']").blur();
                    }

                    $("input[name='Solicitante.Nombre']").val(data.Nombre);
                    $("input[name='Solicitante.Nombre']").focus();
                    $("input[name='Solicitante.Nombre']").blur();

                    if ($('#cmbTipoRepresentante').val() == "Fisica") {

                        $("input[name='Solicitante.Apellido1']").val(data.Apellido1);
                        $("input[name='Solicitante.Apellido1']").focus();
                        $("input[name='Solicitante.Apellido1']").blur();

                        $("input[name='Solicitante.Apellido2']").val(data.Apellido2);
                        $("input[name='Solicitante.Apellido2']").focus();
                        $("input[name='Solicitante.Apellido2']").blur();
                    }
                }
            });
        }
    }
```

Ilustración 23:Poblar los campos del solicitante si su NIF se encuentra en la BD

Cuando el NIF cambia, se hace la llamada correspondiente para comprobar si el NIF introducido tiene una solicitud anterior asociada. Si existe, se devuelve la más reciente mediante esta llamada AJAX

```
[HttpPost]
[AjaxOnly]
0 referencias
public JsonResult ObtenerDatosSolicitantePorNif(string nif)
{
    _logProvider.AddInfo("ExpedienteController | ObtenerDatosSolicitantePorNif | Entrada");
    if (string.IsNullOrEmpty(nif))
    {
        //devolvemos Json vacio = fallo
        _logProvider.AddInfo("ExpedienteController | ObtenerDatosSolicitantePorNif | Error: NIF vacío");
        return Json(new { }, JsonRequestBehavior.AllowGet);
    }
    else
    {
        var solicitante = _solicitanteService.GetLatestSolicitanteByNIF(nif);

        if (solicitante != null)
        {
            var data = new
            {
                Success = true,
                NIF = solicitante.NIF,
                Telefono = solicitante.Telefono,
                Email = solicitante.Email,
                Nombre = solicitante.Nombre,
                Apellido1 = solicitante.Apellido1,
                Apellido2 = solicitante.Apellido2
            };
            _logProvider.AddInfo("ExpedienteController | ObtenerDatosSolicitantePorNif | Salida");
            return Json(data, JsonRequestBehavior.AllowGet);
        }
        else
        {
            //devolvemos Json vacio = fallo
            _logProvider.AddInfo("ExpedienteController | ObtenerDatosSolicitantePorNif | Salida: No existe Solicitante");
            return Json(new { }, JsonRequestBehavior.AllowGet);
        }
    }
}
```

Ilustración 24:Llamada encargada de obtener los datos y devolver respuesta a la llamada Ajax

Si la llamada no devuelve un objeto, se devuelve una respuesta vacía que se interpreta como un fallo.

Si la llamada devuelve un resultado se devuelve un objeto con todos los datos necesarios para poblar los campos.

La repuesta vuelve a la Ilustración 28 donde después de comprobar que hemos recibido una respuesta, vamos ocultando y mostrando los campos de acuerdo con los datos proporcionados.

Y como paso final, vamos rellenando aquellos campos habilitados. No podemos rellenar los campos Primer Apellido y Segundo Apellido si la entidad es Jurídica. Tampoco podemos rellenar el campo email si este no está habilitado (Check Autorizo Notificación Telemática marcado).

Quinta subtarea: El check de Notificación Telemática

El check es necesario marcar cuando el solicitante es un Ciudadano.

```
this.When(model => !model.IdTipoPersona.Equals(SSID.Vivienda.CedulasHabitabilidad.Web.ViewModels.Solicitantes.SolicitanteViewModel.TipoPersona.Juridica), () =>
{
    this.RuleFor(model => model.Apellido1).NotEmpty().WithMessage(Resource.MsgValidarApellido1);
    this.RuleFor(model => model.IdTipoNotificacion).NotNull().When(model => !model.AutorizoNotificacionTelematica).WithMessage(Resource.MsgValidarTipoNotificacion);
    if (SessionManager.User.Esciudadano)
    {
        this.RuleFor(model => model.Email).NotEmpty().WithMessage(Resource.MsgValidarEmailVacio).EmailAddress().WithMessage(Resource.MsgValidarEmailFormato);
    }
    else
    {
        this.RuleFor(model => model.Email).EmailAddress().WithMessage(Resource.MsgValidarEmailFormato);
    }
    this.RuleFor(model => model.AutorizoNotificacionTelematica).Must(x => x == true).When(model => !string.IsNullOrEmpty(model.Email)).WithMessage(Resource.MsgValidarAutorizacionTelematica);
    this.RuleFor(model => model.AutorizoNotificacionTelematica).Must(x => x == true).When(model => model.IdTipoNotificacion == null).WithMessage(Resource.MsgValidarAutorizacionTelematica);

});
if (SessionManager.User.Esciudadano)
{
    this.When(model => !model.IdTipoPersona.Equals(SSID.Vivienda.CedulasHabitabilidad.Web.ViewModels.Solicitantes.SolicitanteViewModel.TipoPersona.Juridica), () =>
    {
        this.RuleFor(model => model.AutorizoNotificacionTelematica).Must(x => x == true).WithMessage(Resource.MsgValidarAutorizacionTelematica);
    });
}
```

Ilustración 25:Reglas de comprobación del check según caso

Tambien es necesario marcar el check por defecto si la nueva solicitud la realiza un ciudadano.

Tarea: Pestaña Dirección.

Primera subtarea: Enlace mailto si no se encuentra la dirección.

Se desea añadir un enlace mailto para que en caso de que un ciudadano no pueda encontrar la dirección de su vivienda en la aplicación, se pueda poner en contacto con el departamento de vivienda. Solo necesita hacer click sobre el enlace y se va a abrir la aplicación de correo que tiene por defecto con la Dirección, Asunto y Texto del correo ya rellenados.

```
<div class="row justify-content-end">
    <div class="col-md-3">
        <p>@Resource.TextoCorreoVivienda <a id="EnlaceNoEncuentroVivienda"
                                            href='mailto:@Resource.CorreoVivienda&subject=@Resource.AsuntoCorreoVivienda&body=@Resource.CuerpoCorreoVivienda'
                                            target="_blank" role='button' title='@Resource.TitleCorreoVivienda'>@Resource.CorreoVivienda</a></p>
    </div>
</div>
```

Ilustración 26:Implementación del mailto

Segunda subtarea: Implementar una ayuda contextual para proporcionar información adicional de un campo al usuario.

He optado por una implementación directa de Bootstrap 3

```
<a tabindex="0" class="button" role="button" data-toggle="popover" data-trigger="hover"
   title="@Html.Raw(Resource.AyudaTituloSuperficieUtil)" data-content="@Html.Raw(Resource.AyudaCuerpoSuperficieUtil)"><i class="fas fa-info-circle"></i></a>
```

Ilustración 27:Definición de la ayuda contextual

Quedaría en la vista de la siguiente manera:

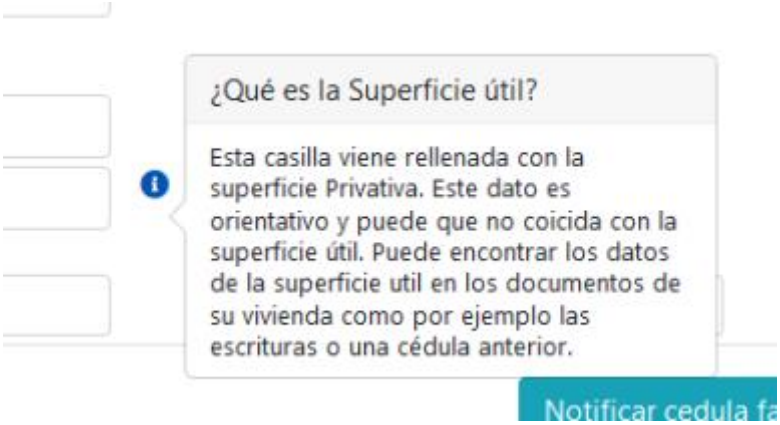


Ilustración 28:Captura de en la vista de la ayuda contextual

Basta solamente con pasar con el ratón por encima para activar la ayuda. En cuanto el ratón sale del circulito azul la ayuda desaparece.

Caso de uso número 3: "Ver solicitudes propias"

El ciudadano tendrá en su menú inicial un botón que le permitirá acceder a la vista de solicitudes de dicho ciudadano. Dicho botón también informara del número de expedientes que dicho ciudadano ha realizado.

La aplicación debe proporcionar a todos los ciudadanos la opción de consultar las solicitudes realizadas sin importar el estado en el que se encuentren. Se propone crear una vista específica para el ciudadano ya que este puede tener varias viviendas. Se deben presentar todas sus solicitudes en formato de tabla y que estén ordenadas cronológicamente desde la solicitud más reciente a la más antigua.

El siguiente diagrama de secuencia describe el comportamiento dinámico de esta vista.

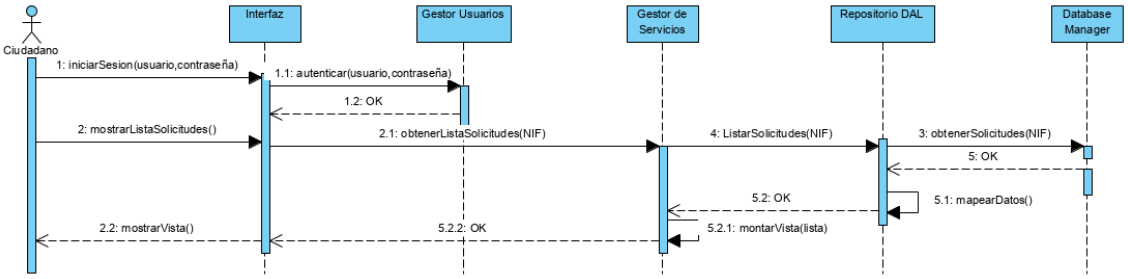


Ilustración 29:Diagrama de secuencia Vista Solicitudes Ciudadano

Se pide que los datos se presenten en formato de una tabla con 7 columnas y estas columnas van a presentar la siguiente información manteniendo el orden:

- Id del Expediente
- El número del expediente
- El número de la cédula (si está disponible)
- La dirección completa de la vivienda
- El estado en el que se encuentra la solicitud
- El ORVE al que pertenece dicha solicitud
- El botón para editar la solicitud

Como todos los campos descritos ya se encuentran en la vista de bandeja de entrada de los gestores de Vivienda, se ha decidido adaptar una vista ya implementada, y de cara a presentarla al usuario, no poblar aquellos campos que no interesa presentar, tales como puede ser el nombre completo del solicitante.

El sistema a la hora de montar la vista no presentará columnas vacías.

Esta forma de implementación tiene una desventaja obvia, ya que solo se pueden mostrar los campos que ya existen en la bandeja de entrada de los usuarios administrativos. En caso de querer mostrar un campo adicional solo para el ciudadano, es necesario implementar una vista desde cero para dicho usuario.

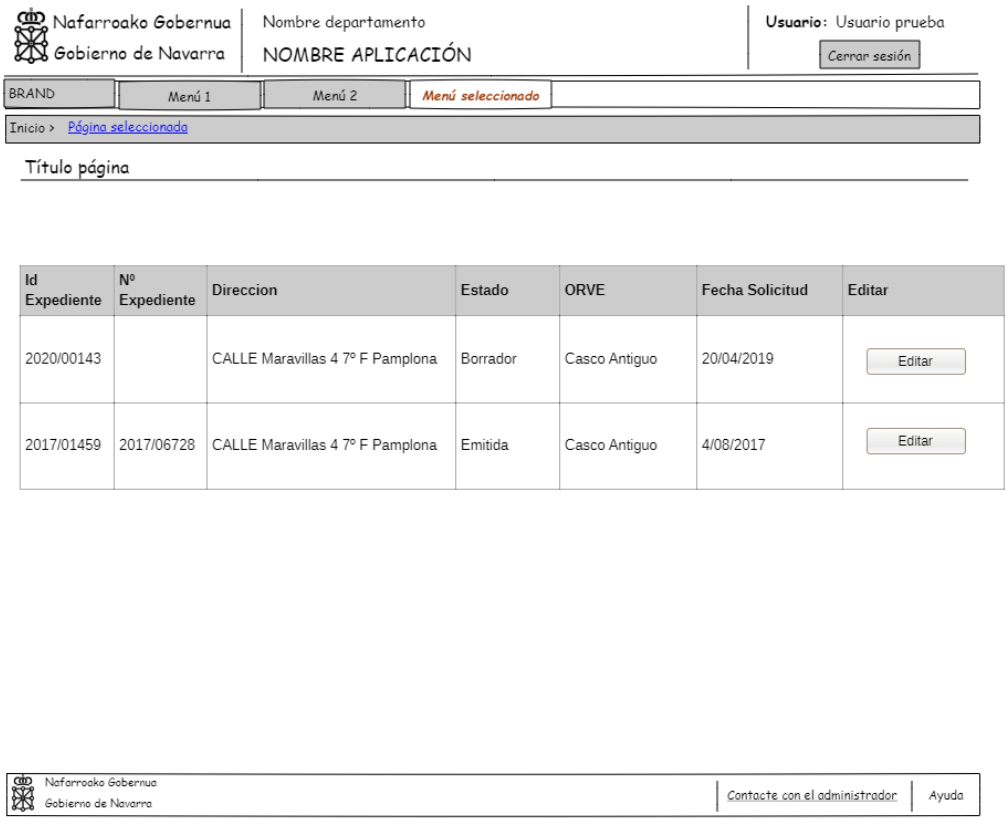


Ilustración 30:Prototipado pencil de la vista Solicitudes Ciudadano

A pesar de ahorrar el tiempo empleado en desarrollar una vista desde cero, aún es necesario crear todos los elementos necesarios para obtener los datos de la base de datos, procesarlos y al final conectarlos con la vista.

El proceso de llamadas se produce en el siguiente orden:

- Index:
  - El usuario pulsa el botón Consulta de Solicitudes
    - Llamada al metodo IndexBandejaEntradaUsuarioPublico
- ExpedienteController
  - Llamada al metodo IndexBandejaEntradaUsuarioPublico
    - Llamada al metodo GetExpedientesBandejaUsuarioPublico
- ExpedienteService
  - Llamada al metodo GetExpedientesBandejaUsuarioPublico de la interfaz IExpedienteRepository
- IExpedienteRepository
  - Llamada al metodo GetExpedientesUsuario
- ExpedienteRepository
  - Llamada al procedimiento almacenado CH_Get_Expedientes_By_Nif

Para ello, he comenzado con definir el procedimiento almacenado necesario para obtener los datos de la BBDD.

Todos los procedimientos almacenados tanto de consulta como de modificación suelen tener una misma estructura:

```
CREATE PROCEDURE [dbo].[Nombre_Procedimiento_Almacenado]
    @Variable_Entrada                   varchar(20)
AS
BEGIN
    SET NOCOUNT ON;

    SELECT
        T1.Columna1 as [NombreSalida]
    FROM dbo.Base_De_Datos AS CED
    INNER JOIN dbo.Tabla1 AS T1 ON T1.Columna1 = T2.ID
    INNER JOIN dbo.Tabla2 AS T2 ON T2.Columna2 = CED.ID
    WHERE
        Tabla3.Columna3=@Variable_Entrada
    ORDER BY CED.ID DESC
END
GO
GRANT EXECUTE
    ON OBJECT::[dbo].[Nombre_Procedimiento_Almacenado] TO [PerfilEjecucion]
    AS [dbo];
```

Ilustración 31:Estructura que siguen todos los procedimientos almacenados

Se pueden destacar una serie de características:

- El uso de procedimientos almacenados ayuda a:
  - Garantizar la seguridad: (Tartas, 2018):
    - Los usuarios/programas no necesitan tener permiso para acceder a las tablas
    - Solo necesitan permiso de ejecución de los procedimientos almacenados
  - Impiden operaciones incorrectas asegurando las reglas de negocio
  - Mejora de rendimiento:
    - Se ahorra la comunicación de grandes cantidades de datos

Uso de NOCOUNT e INNER JOIN evitan procesamiento innecesario y un uso de recursos menor.

Debido a que la aplicación pertenece al Gobierno de Navarra no puedo insertar código o divulgar información acerca de las tablas contenidas dentro de la base de datos. El procedimiento almacenado toma como variable de entrada el NIF del ciudadano y tras hacer un INNER JOIN de 6 tablas se devuelven más campos de los necesarios, en concreto 19.

Aunque vayamos a utilizar solo 7, los datos se procesan y se mapea cada expediente en un objeto llamado ExpedienteResumen utilizado por todas las vistas. Esta generalización hace que tenga campos que nunca se van a mostrar en nuestra vista, pero en cambio ganamos simplicidad en el código y empleamos un objeto de una clase ya existente.

El siguiente paso es definir el metodo GetExpedientesUsuario dentro de ExpedienteRepository.

Este método se encarga de hacer la llamada al procedimiento almacenado y pasar el NIF como parámetro y al obtener los resultados de la consulta hace el mapeo a una lista de ExpedienteResumen y devuelve dicha lista.

Esta Repositorio tiene una interfaz ya que hemos especificado que esta arquitectura tiene como objetivo el desacoplamiento de las reglas de negocio con las librerías y frameworks.

El ExpedienteService realiza todas las llamadas a la interfaz de ExpedienteRepository.

Se encarga de comprobar primero si el usuario tiene permiso para realizar la llamada.

ExpedienteController es el encargado de atender las llamadas resultantes de la interfaz.

Monta la vista con los datos obtenidos del ExpedienteService y devuelve la vista.

Por último, queda añadir un botón en el menú inicio que será mostrado al ciudadano nada más iniciar sesión en la aplicación.

Este botón debe tener como título “CONSULTA SOLICITUDES” y una descripción acerca de cuantas solicitudes tiene dicho ciudadano en curso “Nº de solicitudes en curso:”

Esto supone la creación de un nuevo procedimiento almacenado que devuelva el número de solicitudes que dicho cliente ha realizado. Este procedimiento se debe ejecutar siempre que el ciudadano acceda al Inicio.

La clase encargada de poblar la pantalla de inicio se llama HomeController. Esta clase contiene un método llamado CargarBandejasUsuario que lo único que hace es iterar sobre todos los perfiles de la aplicación y rellenar todos aquellos strings de la clase HomeViewModel que se corresponden al perfil.

En este caso concreto realiza la llamada al método del controlador ExpedienteService que a su vez hace las correspondientes llamadas hasta llegar al procedimiento almacenado.

```
CREATE PROCEDURE [dbo].[                                    ]
    @Nif                    varchar(20)
AS
BEGIN
    SET NOCOUNT ON;

    SELECT
        count(1) as [NumeroExpedientes]
    FROM dbo.                as SOLICITANTE
    INNER JOIN dbo.                as SOLICITUD on SOLICITANTE.Id                = SOLICITUD.Id
    WHERE SOLICITANTE.NIF    =@Nif
END
GO
GRANT EXECUTE
    ON OBJECT::[dbo].[                                    ] TO [UsuariosNormales]
    AS [dbo];
```

Ilustración 32:Procedimiento almacenado para obtener el número de Solicitudes que tiene un Cliente

Es de destacar el empleo de count(1) ya que no nos interesa acceder a los datos, solo contar el número de filas correspondientes a los solicitudes siendo estas NULL o no. Nos estamos ahorrando unos cuantos accesos, pero si lo extrapolamos al número de ciudadanos que pueden acceder a la vez es un ahorro de recursos importante.

La respuesta en forma de int recorre todo el camino de vuelta y se almacena como entero en el objeto HomeViewModel. A partir de este objeto se puebla la vista de inicio en base a que botones puede ver dicho perfil. El resultado de la implementación en el menú inicio quedaría del siguiente

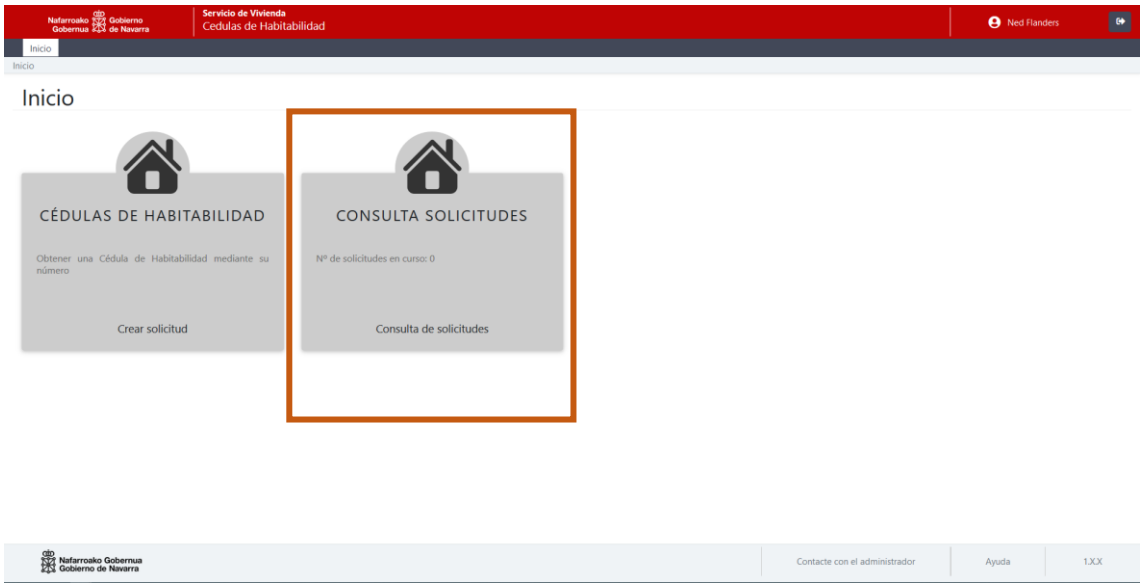


Ilustración 33:Vista de como quedaría el menú inicio con el nuevo botón

Caso de uso número 4: “Editar Solicitud propia”

Partiendo de la implementación realizada para la solicitud de una nueva cédula, se debe añadir la lógica necesaria para cargar el expediente utilizando la misma vista y que se muestre la misma informacion que cuando se generó la solicitud.

```
this.MostrarOcultarSecciones = function () {
    if ($('#cmbTipoSolicitante').val() == "Propietario") {
        $('#panelPropietarioAlQueRepresenta').hide();
    }

    if ($('#cmbTipoRepresentante').val() == "Juridica") {
        $('#lblRazonSocial').removeClass("d-none");

        $('#lblNombreSolicitante').addClass("d-none");

        $('#lblApellido1Solicitante').addClass("d-none");
        $('#Solicitante_Apellido1').addClass("d-none");
        $('#Solicitante_Apellido1').val(null);

        $('#lblApellido2Solicitante').addClass("d-none");
        $('#Solicitante_Apellido2').addClass("d-none");
        $('#Solicitante_Apellido2').val(null);

        $("#panelNotificaciones").hide();
        $('#panelNotificacionesDireccionVivienda').hide();
        $('#panelNotificacionesOtraDireccion').hide();

        $('#Solicitante_AutorizoNotificacionTelematica').on("click.readonly", function (event) { event.preventDefault(); })
        var checkAutorizacionTelematica = $('#Solicitante_AutorizoNotificacionTelematica');
        checkAutorizacionTelematica[0].checked = true;
        var radioButonNotTelematica = $("#optNotificacionesNotificacionTelematica");
        radioButonNotTelematica[0].defaultValue = "NotificacionTelematica";
        radioButonNotTelematica[0].checked = true;

        //deshabilitamos el check y la etiqueta
        //$('#lblAutorizacionNotificacionTelematica').addClass("d-none");
        //$('#Solicitante_AutorizoNotificacionTelematica').parent().hide();

        //habilitamos el campo Email
        document.getElementById('Solicitante_Email').disabled = false;
    } else {

        $('#lblNombreSolicitante').removeClass("d-none");
        $('#lblRazonSocial').addClass("d-none");

        $('#lblApellido1Solicitante').removeClass("d-none");
        $('#Solicitante_Apellido1').removeClass("d-none");

        $('#lblApellido2Solicitante').removeClass("d-none");
        $('#Solicitante_Apellido2').removeClass("d-none");

        //$('#lblAutorizacionNotificacionTelematica').removeClass("d-none");
        ////$('#Solicitante_AutorizoNotificacionTelematica').removeClass("d-none");
        //$('#Solicitante_AutorizoNotificacionTelematica').parent().show();

        $('#Solicitante_AutorizoNotificacionTelematica').off('.readonly').removeAttr("readonly").css("opacity", "1")
        //habilitamos el check y su etiqueta
        $('#lblAutorizacionNotificacionTelematica').removeClass("d-none");
        $('#Solicitante_AutorizoNotificacionTelematica').parent().show();

        if ($('#Solicitante_AutorizoNotificacionTelematica').is(':checked') == true) {
            $('#Solicitante_AutorizoNotificacionTelematica')[0].checked = true;
            var radioButonNotTelematica = $("#optNotificacionesNotificacionTelematica");
            radioButonNotTelematica[0].defaultValue = "NotificacionTelematica";
            radioButonNotTelematica[0].checked = true;
            $("#panelNotificaciones").hide();
            $('#panelNotificacionesDireccionVivienda').hide();
            $('#panelNotificacionesOtraDireccion').hide();
        } else {
            $("#panelNotificaciones").show();
            $('#panelNotificacionesDireccionVivienda').show();
            $('#panelNotificacionesOtraDireccion').show();
        }
    }

    //if (document.getElementById('Solicitante_Email').value) {
    //    $("#panelNotificaciones").hide();
    //    $('#panelNotificacionesDireccionVivienda').hide();
    //    $('#panelNotificacionesOtraDireccion').hide();
    //}
}
```

Ilustración 34:Poblar los campos de la vista según los datos del Objeto Solicitud

Gestion de permisos

El ciudadano puede ver en modo lectura todos los campos de la pestaña Solicitante, pero solo puede cambiar el tipo de notificación.

Para esto he introducido una variable booleana IsReadOnly a la cual se le asigna el valor a true siempre que se este cargando una solicitud ya enviada y que sea el perfil de ciudadano el que quiera verla.

```
<div class="col-md-2">
    @Html.DropDownListFor(_ => Model.Solicitante.IdTipoSolicitante,
    new List<SelectListItem>
    {
        new SelectListItem { Value = SolicitanteViewModel.TipoSolicitante.Propietario.ToString(), Text = @Html.Raw(Resource.TipoSolicitantePropietario).ToString() },
        new SelectListItem { Value = SolicitanteViewModel.TipoSolicitante.Representante.ToString(), Text = @Html.Raw(Resource.TipoSolicitanteRepresentante).ToString() }
    }, Model.Solicitante.IsReadOnly ? (object)new { @id = "cmbTipoSolicitante", @class = "form-control", @disabled = "disabled" } : new { @id = "cmbTipoSolicitante", @class = "form-control" })
</div>
```

Ilustración 35:Solución que deshabilita cualquier campo según el valor de la variable isReadOnly

Cada campo editable se deshabilita, además, aunque se cambie el valor de dicho campo, los cambios no se guardarán.

Las demás pestañas no se pueden editar, así que se añaden los permisos oportunos para que sea posible solamente ver los campos, pero no modificar dichas pestañas. La gestión de los permisos se da en función de las secciones de las vistas. Es un componente que no he desarrollado y por lo tanto no puedo entrar en detalles.

Mostrar el estado y el número de expediente cuando esta se edita

Debe seguir el formato “Nº Expediente: XXX – Estado: YYY”

Se pueden detectar dos tipos de Strings:

- Los strings estáticos: “Nº Expediente:”, “Estado:”, “-“
- Los strings dinámicos que cambiaran de valor según el caso: “XXX” y “YYY”

Para cumplir con los requisitos que exigen en el futuro traducir la aplicación en euskera además del castellano se debe declarar el String de la siguiente forma:

Los string estáticos se declaran en un fichero .resx en este caso se llama Resource.resx .

EL fichero se puede editar de dos formas:

- La primera es una tabla de dos columnas:
    - Primera columna es el nombre del String siguiendo la regla de nombramiento primera letra de cada palabra con mayúscula, el resto con minúscula, se escriben las palabras de forma seguida sin separación ni símbolos y las palabras deben ser cortas y describir brevemente el mensaje que contienen.
    - Segunda columna debe incluir el texto exactamente como se quiere mostrar incluido espacios y caracteres especiales:

	Nombre	Valor
▶	A	a
	Actualizar	Actualizar
	AnioCDesde	Año construcción desde
	AnioCHasta	Año construcción hasta

Ilustración 36:Tabla de strings

- La segunda forma se resume en editar el fichero en formato xml que lo compone:
    - Dentro de la sección <root> se añade una nueva etiqueta que comienza con <data seguido de un espacio.
    - Se añade un espacio y después otra etiqueta name que sirve para nombrar a la nueva variable.
    - Una última etiqueta que se añade a todos los nuevos string es xml:space que es siempre “preserve”
    - Se cierra la >
        - Entre las etiquetas se declara la <value> que es el valor que va a contener el string estático y se cierra al final con </value>
    - Se cierra con </data>

```
<data name="A" xml:space="preserve">
  <value>a</value>
</data>
<data name="Cancelado" xml:space="preserve">
  <value>Cancelado</value>
</data>
```

Ilustración 37:Tabla Strings en formato XML

Como se puede observar, es mucho más intuitivo y más rápida la primera opción, pero además tiene un añadido. No se puede acceder directamente al fichero de strings. Para proteger el fichero se crea de forma automática un getter personalizado para cada string nuevo declarado de la forma:

```
/// <summary>
///   Looks up a localized string similar to Actualizar.
/// </summary>
0 referencias
public static string Actualizar {
    get {
        return ResourceManager.GetString("Actualizar", resourceCulture);
    }
}
```

Ilustración 38: Getter del String llmado Actualizar

Este getter se declara en un fichero .cs nombrado (para facilitar las cosas) Resource1.Designer.cs

Esta forma automatizada de declarar un nuevo string estático solo se puede dar con el primer método presentado y de allí su ventaja y por qué se prefiere utilizar en favor del segundo método.

Para emplear el string solo hace falta llamar la función con el mismo nombre que la variable de la forma:

```
<label id="lblTipoSolicitante" for="cmbTipoSolicitante">@Html.Raw(Resource.TipoSolicitante)</label>
```

Ilustración 39:Definición de una etiqueta de un campo

Para los string dinámicos se utiliza una forma de trabajar que se encuentra en los lenguajes orientados a objetos, es decir, en este caso al menos, los datos del expediente se encuentran en un objeto de la clase Expediente. Como lo que nos interesa mostrar es el número de Expediente (por ejemplo), basta solamente con acceder al Modelo: Model.Solicitud.Expediente. Esto devuelve el número de Expediente.

Para finalizar el mensaje final se compone de una multitud de variables de la forma:

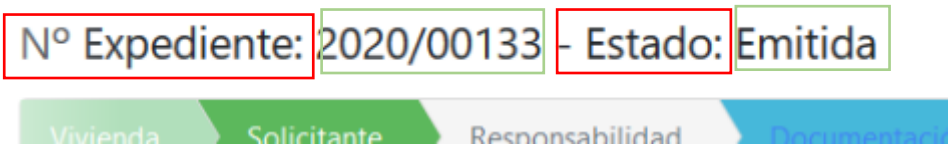


Ilustración 40:Strings estáticos junto con strings dependiendo de la solicitud en una misma línea

Donde el color rojo marca el string estático y el verde el string obtenido de forma dinámica.

Caso de uso número 6: "Descargar Cédula Habitabilidad"

Cuando los Administrativos consideran que todos los requisitos para emitir la cédula de habitabilidad se cumplen, presionan el botón de "Emitir cédula". El usuario cliente quedará notificado a través de su método elegido de notificación que ya es posible descargar su cédula de habitabilidad.

Por lo tanto, el usuario debe volver a ingresar a la aplicación, buscar en su bandeja de solicitudes, aquella solicitud Emitida y en la pestaña Resumen Cédula, se presentará al usuario con un botón "Descargar Cédula" que, al presionar, se descarga una copia guardada del PDF de cédula de habitabilidad generado utilizando un servicio propio del Gobierno de Navarra para descargar dicho documento.

La plantilla para generar la Cédula de Habitabilidad se ha realizado utilizando la herramienta propia de GAT, Reporta. Se ha generado y alineado el documento correspondiente a la cédula de Habitabilidad que el ciudadano descargaría una vez haya finalizado la tramitación de forma favorable.

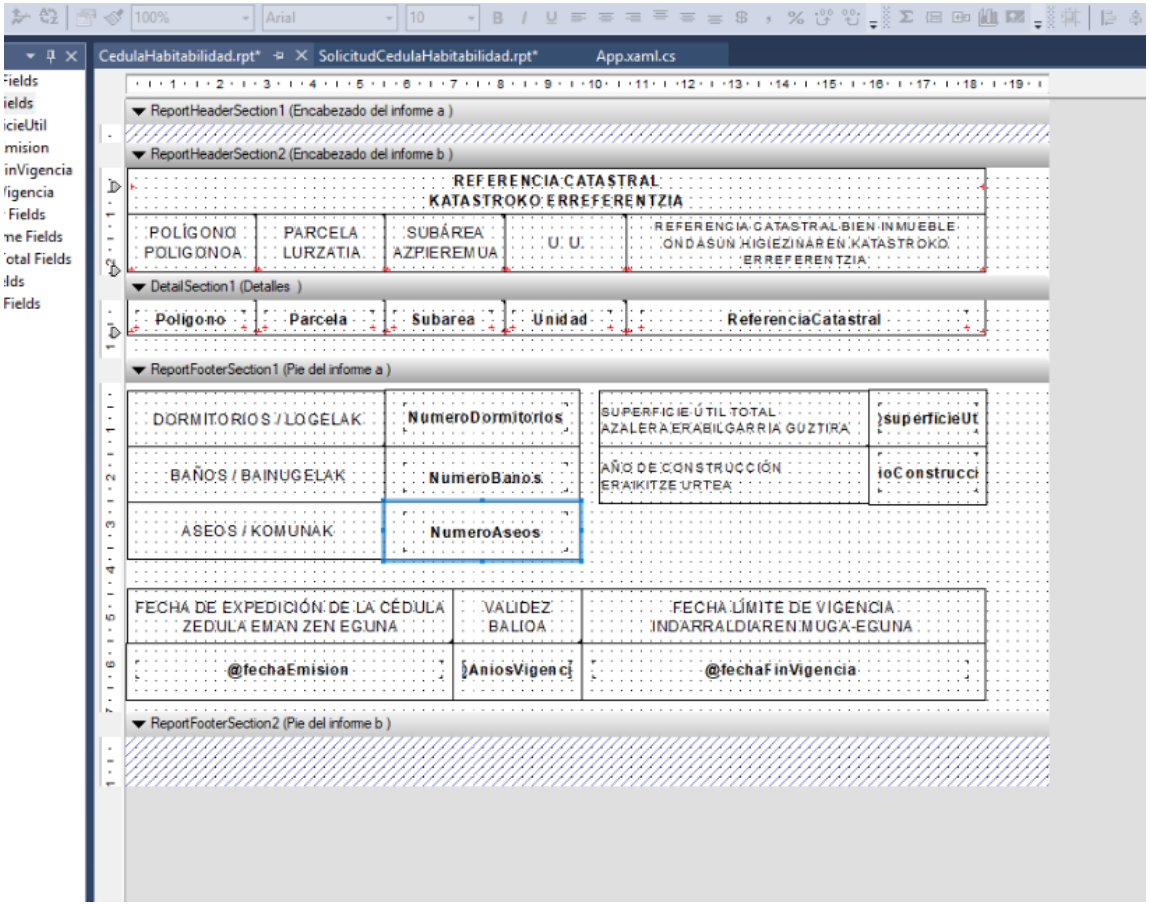
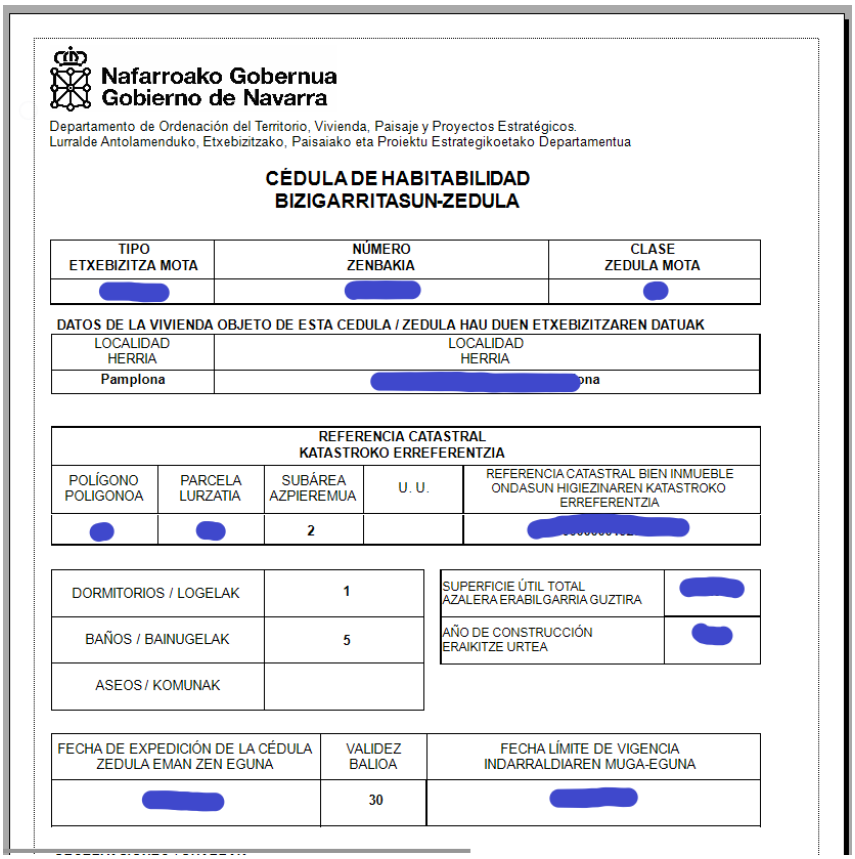


Ilustración 41: Captura del editor Reporta dentro de Visual Studio

A continuación, una renderización del documento PDF acabado utilizando la herramienta propia Prevista.



Nafarroako Gobernua
Gobierno de Navarra

Departamento de Ordenación del Territorio, Vivienda, Paisaje y Proyectos Estratégicos.
Lurralde Antolamenduko, Etxebizitzako, Paisaiako eta Proiektu Estrategikoetako Departamentua

CÉDULA DE HABITABILIDAD
BIZIGARRITASUN-ZEDULA

TIPO ETXEBIZITZA MOTA	NÚMERO ZENBAKIA	CLASE ZEDULA MOTA
[redacted]	[redacted]	[redacted]

DATOS DE LA VIVIENDA OBJETO DE ESTA CEDULA / ZEDULA HAU DUEN ETXEBIZITZAREN DATUAK

LOCALIDAD HERRIA	LOCALIDAD HERRIA
Pamplona	[redacted]ona

REFERENCIA CATASTRAL KATASTROKO ERREFERENTZIA				
POLÍGONO POLIGONOA	PARCELA LURZATIA	SUBÁREA AZPIEREMUA	U. U.	REFERENCIA CATASTRAL BIEN INMUEBLE ONDASUN HIGIEZINAREN KATASTROKO ERREFERENTZIA
[redacted]	[redacted]	2		[redacted]

DORMITORIOS / LOGELAK	1
BAÑOS / BAINUGELAK	5
ASEOS / KOMUNAK	

SUPERFICIE ÚTIL TOTAL AZALERA ERABILGARRIA GUZTIRA	[redacted]
AÑO DE CONSTRUCCIÓN ERAIKITZE URTEA	[redacted]

FECHA DE EXPEDICIÓN DE LA CÉDULA ZEDULA EMAN ZEN EGUNA	VALIDEZ BALIOA	FECHA LÍMITE DE VIGENCIA INDARRALDIAREN MUGA-EGUNA
[redacted]	30	[redacted]

Ilustración 42: PDF Cédula de Habitabilidad generado a partir de una solicitud

Caso de uso número 8: "Buscar solicitudes"

Los requisitos para esta vista son los siguientes:

- Solo los usuarios Administrativos pueden acceder esta vista.
- Administrativos pueden realizar una búsqueda de expedientes utilizando los siguientes campos:
    - Nº Expediente
    - Desplegable para seleccionar Estado
    - En caso de que el usuario no sea ORVE, desplegable para seleccionar el ORVE
    - Nº Cédula
    - Desplegable con la Clase de Cédula
    - Desplegable con los nombres de los Técnicos
    - NIF
    - Nombre
    - Primer Apellido
    - Segundo Apellido
    - Localidad
    - Calle
    - Portal

    - Escalera
    - Piso
    - Puerta
    - Polígono
    - Parcela
    - Subárea
    - Unidad
    - Año construcción desde:
    - Año construcción hasta:
- Después de realizar la búsqueda se puede filtrar el resultado por columnas
- Se pueden ocultar los campos de búsqueda cuando sea necesario
- La tabla que va a contener los resultados de la búsqueda debe presentar los siguientes campos:
    - Nº Expediente
    - Nº Cédula
    - Solicitante (Apellido1, Apellido2, Nombre)
    - Direccion (Direccion, Numero, Piso, Puerta, Población)
    - Estado
    - ORVE
    - Fecha Solicitud
    - Columna Editar

Antes de comenzar la implementación y dado que esta vista tiene una cantidad grande de campos, he optado por primero realizar un prototipo de como quedaría la interfaz de la vista utilizando la herramienta de prototipado Pencil. La tarea no especifica el orden y tampoco la distribución de los campos. El diseño de la vista mediante el prototipado ha permitido al equipo de desarrollo evaluar la idea inicial en muy poco tiempo y mejorarla hasta conseguir la distribución adecuada de los campos en la vista.

Esta opción ha sido un acierto en esta implementación ya que he podido desarrollar una representación de la vista en poco tiempo para poder ser evaluada por el equipo antes de comenzar la implementación.

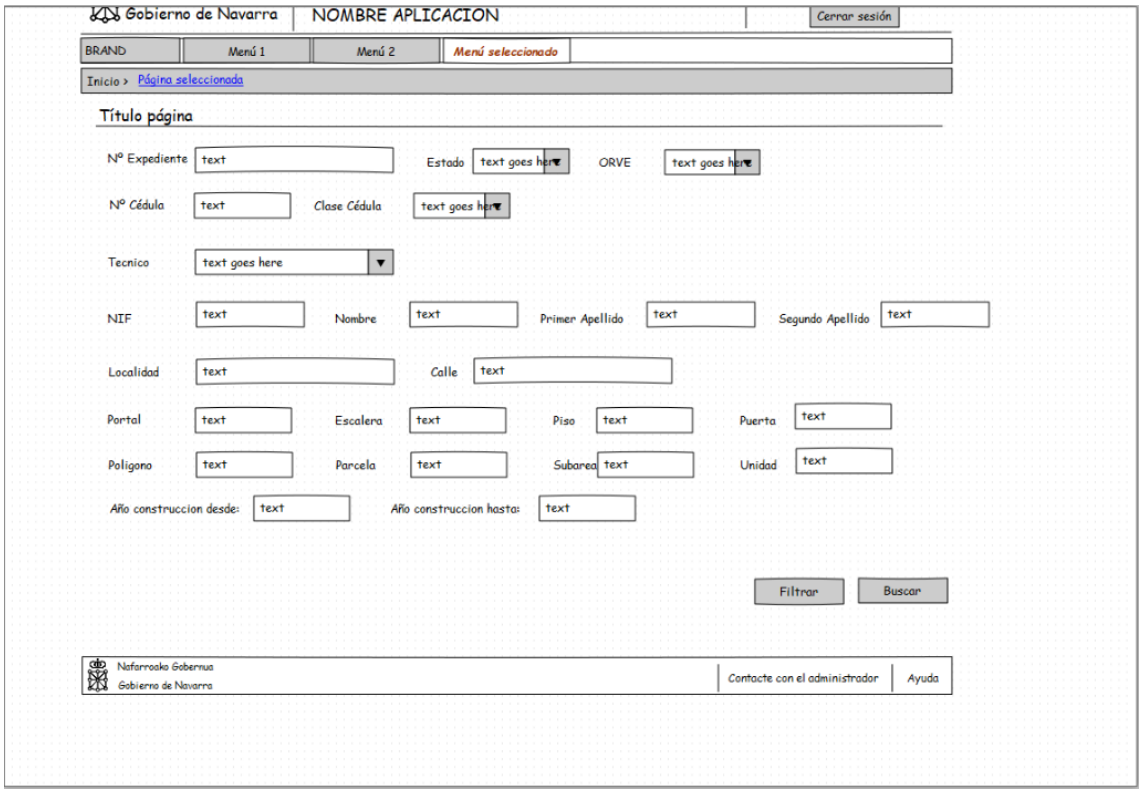


Ilustración 43:Prototipado Pencil de la vista Búsqueda Solicitudes

A partir de esta distribución ha sido mucho más fácil la implementación ya que el diseño ya estaba definido.

El diseño de la vista es bastante sencillo ya que utiliza un sistema de posicionamiento tipo grid de 11 columnas.

```
<div class="row mb-1">
    @Html.LabelFor(_ => Model.SolicitanteApellido1, Resource.Apellido1, new { @class = "col-form-label col-md-2 text-right" })
    <div class="col-md-2">
        @Html.TextBoxFor(_ => Model.SolicitanteApellido1, new { @class = "form-control" })
    </div>
    @Html.LabelFor(_ => Model.SolicitanteApellido2, Resource.Apellido2, new { @class = "col-form-label col-md-2 text-right" })
    <div class="col-md-2">
        @Html.TextBoxFor(_ => Model.SolicitanteApellido2, new { @class = "form-control" })
    </div>
    <div class="col-md-3"></div>
</div>
```

Ilustración 44: Ejemplo diseño fila del grid

Cada fila “row” está formada por un div con la etiqueta Bootstrap “row” con margen 1. Dentro del div, pueden estar presentes tantas divisiones como columnas (11). Cada campo editable de texto está rodeado por un div y la etiqueta se encuentra en el exterior.

Los espacios en blanco, hasta 11, se deben rellenar como esta en este ejemplo. Los campos ocupan 8 divisiones pero quedan 3 sin especificar así que se añade “class=”col-md-3” para que

se pueda mantener la forma y el comportamiento deseado al cambiar dinámicamente la resolución de pantalla.

También se puede comprobar que en el archivo se está definiendo un objeto Model que va a recoger los valores introducidos en los campos. Este modelo es enviado a la aplicación para las comprobaciones y las acciones pertinentes, bastante transparente al programador.

Hay una colección de componentes HTML ya implementadas que solo es necesario instanciar y utilizar. Ejemplos: “TextBoxFor”, “Select2For”

```
@Html.LabelFor(_ => Model.IdOrve, Resource.Orve, new { @class = "col-form-label col-md-1 text-right" })
<div class="col-md-2">
    @Html.Select2For(_ => Model.IdOrve, string.Empty, new SelectList(SessionManager.Orves, "Id", "Nombre"), new { @class = "form-control" })
</div>
```

Ilustración 45: Ejemplo de un campo y su etiqueta

A la hora de poblar la vista y generar el HTML se cambiarán dichas etiquetas con código estático. Esta forma de componer las vistas ya se ha visto durante la carrera en la asignatura de Sistemas de Información Web. Solo que en este proyecto y esta arquitectura consiguen abstraer al programador de las peculiaridades para enfocarse en realizar una buena implementación de la solución.

A pesar de estas facilidades, nos encontramos con un problema generado por la especificación de los requisitos. Nos dicen específicamente que los usuarios ORVE, no pueden filtrar por ORVE dentro de la vista de Búsqueda Expedientes.

La solución se traslada a generar dos ficheros cshtml. Uno para los usuarios ORVE y otro para los usuarios Administrativos que no son ORVE.

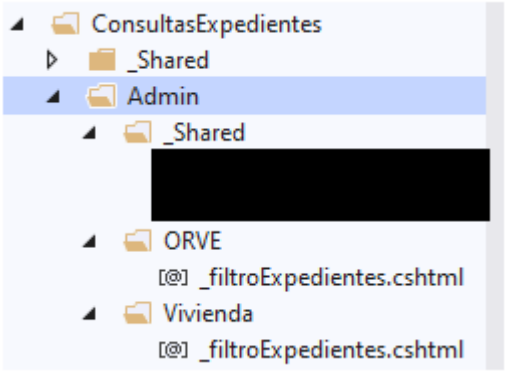


Ilustración 46: Organización de los distintos fragmentos que componen las vistas de Filtro Expediente

La única diferencia de los ficheros es el campo ORVE que solo los Administrativos no ORVE poseen. (figura X)

La tabla que va a contener los resultados de la búsqueda se encuentra en el fichero _listadoExpedientes.cshtml.

```
@Html.GridView("gridListadoExpedientes", Model,
    new
    {
        primaryKey = "IdExpediente",
        headerFilter = new { type = "onchange" },
        columns = new List<object> {
            new {field = "IdExpediente", hidden = true},
            new {field = "NumeroExpediente", title = Resource.ExpedienteNumero,  sortable = true, filterable = true, align = "center"},
            new {field = "NumeroCedula", title = Resource.CedulaNumero,  sortable = true, filterable = true, align = "center"},
            new {field = "NombreSolicitante", title = Resource.Solicitante,  sortable = true, filterable = true, align = "center"},
            new {field = "DireccionCompleta", title = Resource.Direccion, sortable = true, filterable = true},
            new {field = "DescripcionEstado", title = Resource.Estado, sortable = true, filterable = true},
            new {field = "NombreOrve", title = Resource.Orve.ToUpper() , sortable = true},
            new {field = "FechaSolicitud", title = Resource.FechaSolicitud, sorteable = true, filterable = true},
            new {tmpl = "<button id='boton-editargrid' class='btn btn-sm btn-light tabla-editar'><i class='far fa-edit'></i> " + Gl
        }
    }, new { @class = "table-hover" }
)
```

Ilustración 47: Definición de la tabla en la que se presentan los resultados

Una tabla grid estándar de la arquitectura que con pocas etiquetas “sorteable” y “filterable” obtenemos opciones de filtrado potentes una vez realizada la búsqueda.

Después de generar el procedimiento almacenado y crear las vistas, es necesario implementar la lógica. Para ello, en ExpedienteController, primero vamos a generar un método llamado IndexExpedientes que devuelve la vista de búsqueda.

Después vamos a añadir un comportamiento al botón buscar que realizara la llamada al método GetExpedientes de ExpedienteController. Este método se encarga de realizar la petición a ExpedienteService de la búsqueda en base al filtro obtenido, El ExpedienteService a su vez realizara la llamada al ExpedienteRepository, ejecutar el procedimiento almacenado y subir los resultados mapeados al nivel más alto, montar la vista y mostrar los resultados.

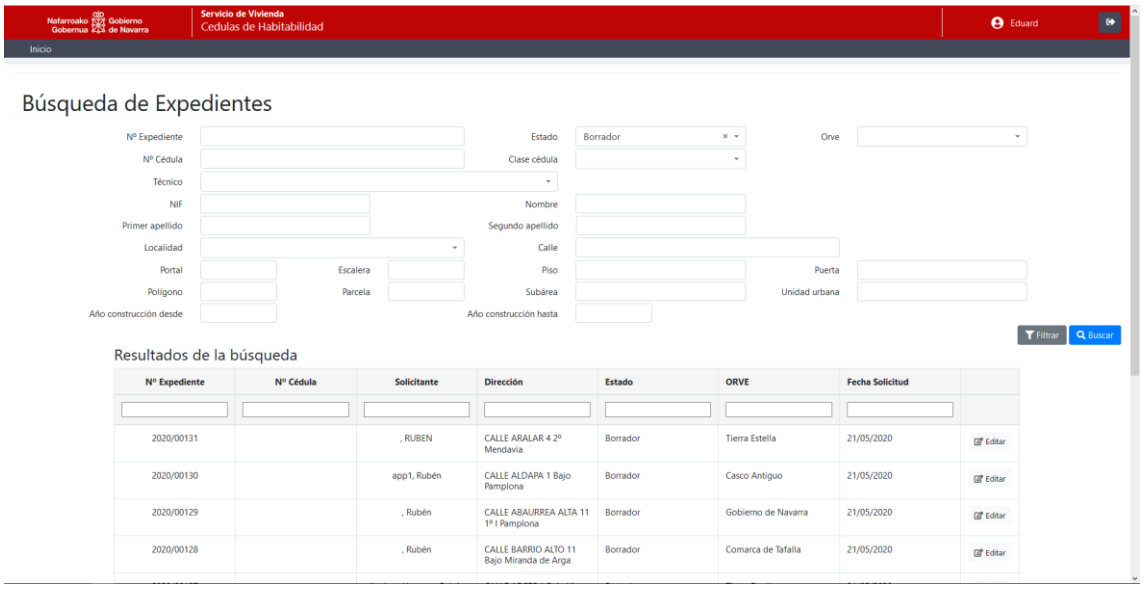


Ilustración 48: Captura de pantalla de la vista implementada

Sin embargo, la solución que se ha implementado hasta ahora ha generado tres problemas que a continuación explicaré y también las soluciones que he implementado para resolverlos.

Búsquedas parciales e incongruencia en la BBDD:

Se pide que los campos Vía, Apellido1, Apellido2 y Nombre tengan la posibilidad de poder realizar en ellas una búsqueda parcial. Esto significa que es necesario realizar una búsqueda “like ‘%%’” pero además un saneamiento de las entradas quitando todos los acentos y hacer que no se distingan mayúsculas o minúsculas.

Se puede realizar un tratamiento de estos campos en la aplicación.

Sin embargo, nos podemos encontrar con ejemplos de la base de datos donde el mismo nombre puede aparecer escrito de diferente forma: “Luis”, “Luís”,” luis”, “LUIS” ya que los registros más antiguos presentes se han añadido a mano en un momento dado y se han producido cambios en el formato en el que se guardan los datos. Estos no darían un buen resultado a la hora de realizar una búsqueda parcial “Lui” porque en este caso solo devolvería “Luis” como resultado obviando los demás registros.

Tampoco se puede plantear reescribir todos los registros de la base de datos ya que tenemos como requisito especial no cambiar las tablas ni los registros que se emplean en la aplicación actual que se está empleando.

Por lo tanto, es necesario investigar un poco las características que ofrece SQL Server y cómo podemos solucionar este problema de la forma más elegante y sencilla posible.

Mirando la documentación en línea de SQL Server COLLATE (Microsoft SQL Documentacion, s.f.) podemos observar que las bases de datos pueden tener especificado el idioma en el que se van a guardar los datos, para procesar letras de un determinado alfabeto por ejemplo y que existen las llamadas intercalaciones, que pueden ser transformaciones que se pueden aplicar a una tabla entera o solo a una columna siempre que sea de tipo nchar, nvarchar y ntext . Hay una variedad grande de configuraciones (Documentacion, s.f.)

En nuestro caso vamos a elegir el idioma SQL_Latin1_General con las configuraciones por defecto, pero queremos que no pueda distinguir entre mayúsculas y minúsculas. Tampoco que tenga en cuenta los acentos.

La repuesta la encontramos en la documentación de Microsoft (SQL, s.f.) donde a SQL_Latin1_General es necesario añadir las terminaciones CI o case insensitive, es decir que no distinga entre minúsculas y mayúsculas y una segunda terminación AI o accent insensitive que no distingue los acentos.

Por lo tanto, la opción con la que vamos a llamar a COLLATE dentro del procedimiento almacenado va a ser SQL_Latin1_General_CI_AI. Justo el comportamiento que estábamos buscando.

Los campos que precisen de una búsqueda parcial pasan de estar declarados de esta forma:

```
(@Via is null or VIV.Via like '%' + @Via + '%') and
```

Ilustración 49: Filtro parcial por defecto

a esta nueva forma especificando la transformacion tanto al elemento con el que comparar como a la columna con la que vamos a comparar.

```
(@Via is null or VIV.Via COLLATE Latin1_General_CI_AI like '%' + @Via + '%' COLLATE Latin1_General_CI_AI)
```

Ilustración 50: Filtro parcial sin tener en cuenta mayúsculas o acentos

Analicemos esta opción desde el punto de vista del rendimiento. Esta nueva especificación no requiere de procesamiento adicional ya que no distingue mayúsculas ni minúsculas y tampoco acentos. Se puede realizar a la vez que se ejecuta el procedimiento almacenado sin necesidad de un procesamiento anterior o posterior y se puede aplicar a cualquier campo de tipo nchar, nvchar o ntext.

El segundo problema al que nos estamos enfrentando es:

Filtro de búsqueda que devuelve todos los expedientes:

La búsqueda tal y como está implementada, si no se rellena ningún campo y se busca con este filtro vacío, se van a devolver todos los expedientes de la base de datos.

Este es un problema por varios motivos.

- Si un administrativo quiere buscar los últimos expedientes y le da a buscar, está gastando recursos en obtener todos los expedientes de la base de datos para ver solo los primeros resultados.
- Es un procesamiento de datos muy grande para la aplicación ya que todo el proceso de filtrar por estados se está llevando a cabo fuera de la base de datos. Un número muy grande de datos a procesar hace que la aplicación falle.

Por lo tanto, se propone forzar a los administrativos a rellenar los campos del filtro de búsqueda del siguiente modo.

Se especifica un parámetro global dentro de la configuración del proyecto (Web.config), que especificará el límite de resultados que una búsqueda puede devolver.

```
<add key="LimiteBusquedaExpedientes" value="100" />
```

Ilustración 51: Definición de la variable en Web.config

Se añade un nuevo procedimiento almacenado que se ejecuta antes de la búsqueda de la siguiente forma.

En el método BuscarExpedientes de ExpedienteController se realiza el mapeo del modelo ExpedienteFiltroViewModel al ExpedienteFiltro. Seguidamente se realiza la llamada a la función `GetExpedientesPorFiltroCount` que devuelve el número de expedientes según dicho filtro y el límite de expedientes permitidos más uno.

```
var filas = _expedienteRepository.GetNumeroExpedientesFiltroCount(filtro, Int32.Parse(ConfigurationManager.AppSettings["LimiteBusquedaExpedientes"])+1);
```

Ilustración 52: Variable filas que guarda el resultado de la llamada al procedimiento almacenado

El ExpedienteRepository hace la llamada al dicho procedimiento almacenado. Anteriormente, sin embargo, hemos dicho que le estamos pasando el límite máximo de filas más uno. He tomado esta decisión por cómo funciona el count dentro de SQL Server.

```
    @LimiteBusqueda         int

AS
BEGIN
    SET NOCOUNT ON;

    WITH Expedientes as
    (
        SELECT TOP (@LimiteBusqueda) CED.IdCedula
```

Ilustración 53: Fragmento procedimiento almacenado GET_ExpedienteNumero_count

Pasando el límite al procedimiento almacenado asegura que independientemente del número de resultados que dicha consulta devuelve, como mucho va a recorrer limite+1 filas antes de

salir. No nos sirve de nada saber cuántas filas totales devuelve esa búsqueda si sobrepasa el límite.

El +1 al límite sirve para saber si el número de filas es exactamente el límite o nos hemos pasado. Se puede dar el caso que la búsqueda devuelva justo el límite, entonces es necesario saber si la búsqueda ha terminado porque el número de filas es igual o menor al límite, en cuyo caso es un resultado satisfactorio, o por el contrario nos hemos pasado y necesitamos restringir más la búsqueda.

El resultado del número de filas sube hasta el ExpedienteController donde se comprueba si se ha sobrepasado el límite.

```
if (limite <= Int32.Parse(ConfigurationManager.AppSettings["LimiteBusquedaExpedientes"]))
{
    var list = _expedienteService.GetExpedientesPorFiltro(filtro);
    model.Lista = _mapperService.Map<IList<Entities.ExpedienteResumen>, IList<ExpedienteListadoViewModel>>(list).ToList();
    model.BusquedaRealizada = true;
    model.BusquedaPorEncimaLimite = false;
}
else
{
    model.Lista = _mapperService.Map<IList<Entities.ExpedienteResumen>, IList<ExpedienteListadoViewModel>>(null).ToList();
    model.BusquedaRealizada = false;
    model.BusquedaPorEncimaLimite = true;
}
_logProvider.AddInfo("ExpedienteController | BuscarExpedientes | Salida");
return View(Vista.ConsultasExpedientes.Global, model);
```

Ilustración 54:Logica dentro de ExpedienteController

Si no lo ha hecho se realiza la búsqueda de los expedientes, se mapean y se devuelven a la vista, en caso contrario se devuelve un mapeo vacío y un booleano que nos marca si nos hemos pasado del límite.

El valor booleano se comprueba en la vista para que en caso de que la búsqueda pasado el límite de resultados, ocultar la tabla y mostrar un mensaje que pide al usuario que restrinja más su búsqueda añadiendo más campos a su filtro.

Y por último el tercer problema:

Que valores asignar a los campos años desde/ hasta cuando no se rellenan.

Para que la búsqueda sea satisfactoria sin tener los campos años construcción desde /hasta rellenados, o solo uno de ellos, simplemente tenemos que comprobar si obtenemos los valores null cuando nos encontramos en el ExpedienteRepository.

```
_dataBaseManager.AddInInteger32Parameter(command, "@AnioCDesde", (filtro.AnioCDesde.HasValue ? filtro.AnioCDesde : Int32.MinValue));
_dataBaseManager.AddInInteger32Parameter(command, "@AnioCHasta", (filtro.AnioCHasta.HasValue ? filtro.AnioCHasta : Int32.MaxValue));
```

Ilustración 55:Logica para asignar valor a los campos año si no lo tienen seleccionado.

Aplicamos dos condiciones donde si el año construcción desde no está poblado, le asignamos el valor int32 más pequeño y lo mismo vamos a hacer con el campo año construcción hasta, pero en este caso le vamos a asignar el máximo valor de int32.

Caso de uso número 10: “Ver solicitud” junto con Caso de uso número 11: “Editar Solicitud”

Para la vista de los Administrativos en la pestaña dirección se quiere mostrar detalles adicionales de las direcciones como pueden ser el número de aseos, Tipo de vivienda, Año de construcción, Parcela, Polígono, Superficie Útil y Subárea.

Estos nuevos cambios se añaden solamente a la vista de los admirativos y quedaría de la siguiente manera.

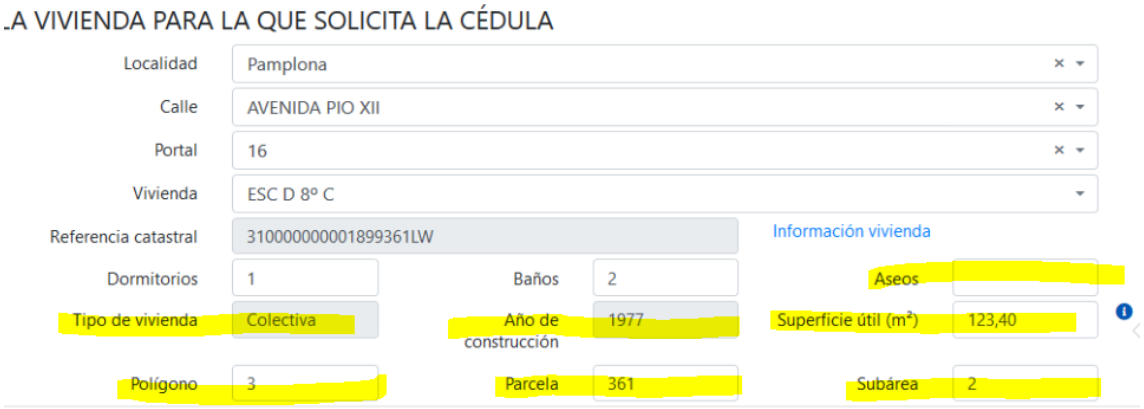


Ilustración 56: Captura de los campos adicionales que se quieren añadir a la vista

Implementación del servicio de logging Log4Net:

En la arquitectura por defecto se definen dos tipos de logging, uno local, donde se guarda en ficheros de log y otro online mediante grayLog.

Ilustración 57: Logotipo grayLog

Para guardar los logs es necesario configurar el servicio. La configuración consiste en realizar un pedido al catálogo de servicios para que puedan configurar grayLog para la aplicación. Se recibe un correo con la dirección a la que debe acceder la aplicación para conectarse con grayLog. Ese parámetro se guarda en Web.config

Todos los métodos internos deben poder ser auditados. Por lo tanto, cada llamada a un método puede escribir los siguientes logs:

```
_logProvider.AddInfo("ExpedienteService | GetExpedientesBandejaUsuarioPublico | Entrada");
/*** Realizar busqueda ***/
var lista = _expedienteRepository.GetExpedientesUsuario(nif);
_logProvider.AddInfo("ExpedienteService | GetExpedientesBandejaUsuarioPublico | Número de Expedientes es: " + lista.Count);

/*** Salida ***/
_logProvider.AddInfo("ExpedienteService | GetExpedientesBandejaUsuarioPublico | Salida");
```

Ilustración 58: Tipos de logs

Todos los logs tienen la misma construcción:

- Servicio en el que se realiza la llamada
- El método al cual se está llamando
- Por último, si es la entrada, la salida o si es una información importante obtenida de la llamada a otro método.

Los logs han sido de mucha utilidad durante el desarrollo de la aplicación ya que en caso de un error se podía localizar rápidamente donde se producía mirando los logs.

4. Desarrollo:

La metodología de desarrollo de software que se ha empleado en este proyecto ha sido uno Iterativo e Incremental.

Se considera iterativo ya que el proyecto comenzó a partir de la arquitectura definida por defecto, se han añadido los servicios que el proyecto necesita y se han configurado los ficheros de configuración y despliegue desde el primer momento y se ha subido una versión básica al servidor de desarrollo.

A partir de ese momento inicial se han priorizado las tareas que tienen que ver con la lógica de negocio. Es necesario obtener el mínimo producto viable para que el cliente pueda evaluarlo de forma temprana y comprobar que se están cumpliendo las expectativas y las funcionalidades acordadas. Esto crea una base sólida a partir de la cual se pueden construir las vistas y los flujos necesarios para satisfacer los requisitos del cliente.

Es también una metodología Incremental ya que el objetivo es obtener un producto básico con las funcionalidades mínimas. Debido a la larga vida de estos tipos de productos, es muy probable que se beneficie de una evolución en el futuro.

Desde el punto de vista de la gestión del proyecto se estaban utilizando matices de desarrollo ágil pero no tienen implementado completamente esta forma de trabajar.

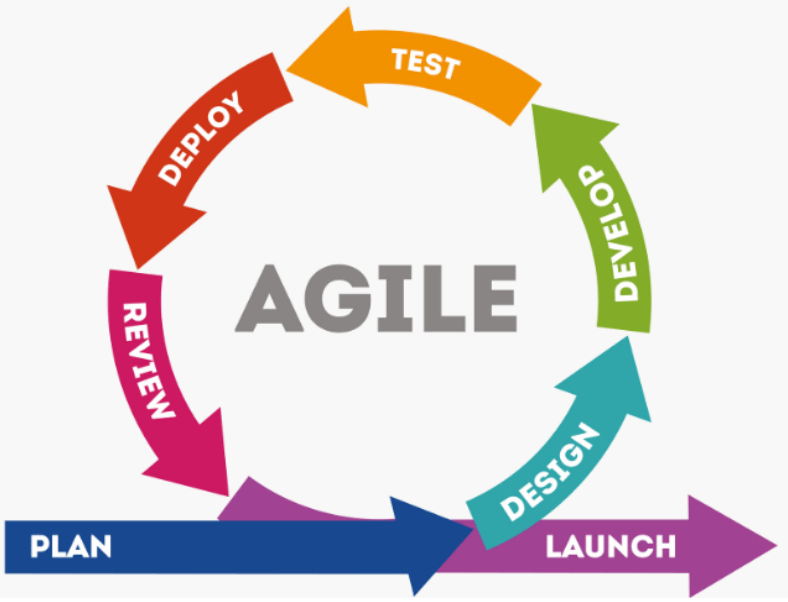


Ilustración 59: Esquema básico de una iteración de un Método Ágil

La planificación de las tareas que debe realizar cada miembro del equipo se decide al comenzar la semana después de una reunión de una hora donde se va poniendo en común en qué situación se encuentra cada uno. En esta hora se discuten multitud de proyectos que llevan en común como equipo, así que como mucho tienen 20 minutos por cada uno.

Es imposible realizar una implementación completa del método ágil ya que aparte de ser desarrolladores también están atendiendo incidencias que la mayoría de las veces son urgentes. Este hecho hace que sea imposible comprometerse con un número de horas a la hora de planificar cada iteración del ciclo de vida.

En mi caso, como era el único proyecto en el que estaba participando, sí que han podido asignarme a la semana un numero especifico de tareas que tenía que acabar.

Las iteraciones son reducidas prácticamente a diseñar, desarrollar, desplegar y pasar las pruebas. La única forma de comprobar y validar que todas las implementaciones son correctas es cuando se realiza la reunión con los clientes para presentar una nueva versión. La reunión se producía cada dos meses, más o menos.

Tampoco se realizaban retrospectivas ya que tuvieron malas experiencias con ellas en el pasado, pero sí que estaban considerando retomarlas.

Herramientas empleadas:

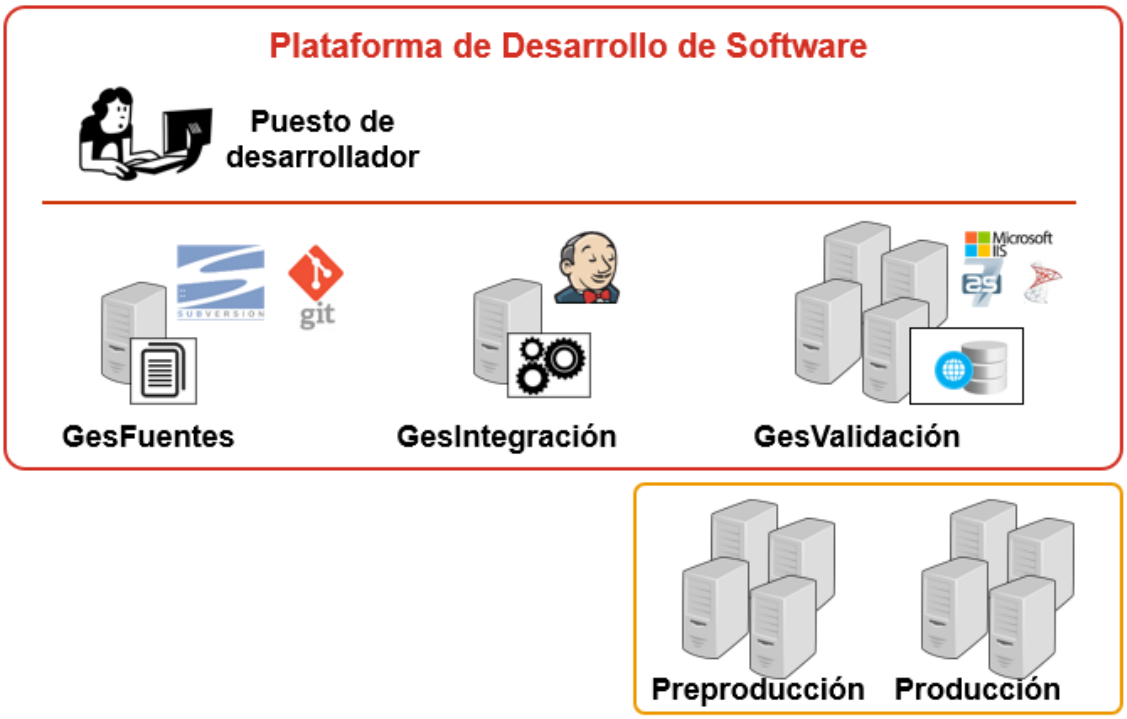


Ilustración 60: Arquitectura de la plataforma de desarrollo de software (Foto tomada de la presentación de la PDS)

Puesto de desarrollador:

- IDE: Visual Studio 2012
- Administrar BD: SQL Server Management Studio
- Navegadores Web: Firefox 70 y Internet Explorer 11
- Aplicación GIT: TortoiseGIT
- Servidor web local IIS para desplegar en local
- Editor de texto: Word
- Editor de hojas de cálculo: Excel
- Visualizador de presentaciones: PowerPoint
- Herramienta de prototipado: Pencil
- Servicio Puesto de Desarrollador
    - Servicio necesario para instalación de herramientas

GesFuentes:

- Sistema de control de versiones con Git

GesIntegracion:

- Interfaz Jenkins

- Los builds se almacenan en los servidores de preproducción.

GesValidacion:

- Infraestructura sobre la cual realizar pruebas
- Mecanismo para desplegar productos sobre los servidores de pruebas

Proceso de una implementación:

- Descargar la versión más actualizada de la aplicación usando GIT
- Leer la tarea a implementar desde el EDT
- En caso necesario realizar un prototipado y comentarlo con los demás desarrolladores
- Implementar de forma incremental la solución.
- Desplegar la solución con la nueva implementación en local
- Realizar las pruebas en local
- Subir los cambios a GIT
- Desplegar el proyecto en el servidor remoto mediante Jenkins
- Realizar las pruebas en el servidor remoto.
- Dar la tarea por finalizada

Formas de trabajar:

Durante el desarrollo he podido conocer dos formas de trabajar en la aplicación.

La primera forma de trabajar consiste en realizar las tareas en el lugar de trabajo, dentro de la intranet de la empresa. Esto significa que tengo el acceso a todos los servicios que la aplicación utiliza, la base de datos en el servidor remoto y a los sistemas de comunicación establecidos dentro de la empresa. Las ejecuciones de la aplicación en el servidor local IIS del ordenador funcionaba sin problemas.

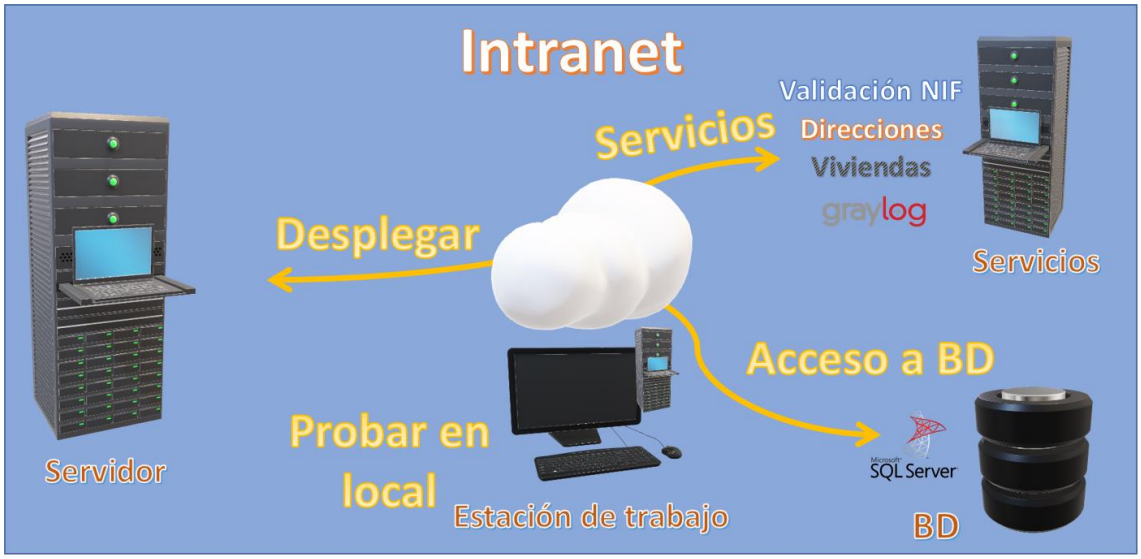


Ilustración 61: Trabajar desde la empresa en la Intranet

La segunda forma de trabajar, a la cual tuve que adaptarme debido a la pandemia mundial de 2020 es trabajar desde mi casa, en el ordenador de la empresa.

Esta forma de trabajo cambia radicalmente ya que la estación de trabajo se encuentra fuera de la intranet. Se deja de tener acceso a los servicios que la aplicación utiliza y se deja de tener acceso a la base de datos en remoto. Por lo tanto, la forma de trabajar se reduce en tener una aplicación que solo puede abrir solicitudes ya creadas, no puede crear nuevas solicitudes, ni eliminar. No es una forma buena de trabajar ya que la mayoría de las tareas de implementación estaban enfocadas justamente en el momento de creación de solicitudes y modificación de estas.

Este hecho imposibilita poder probar la aplicación correctamente en la maquina local y esto se traduce rápidamente en subir errores que solo se pueden descubrir al desplegar los cambios en el servidor de desarrollo.

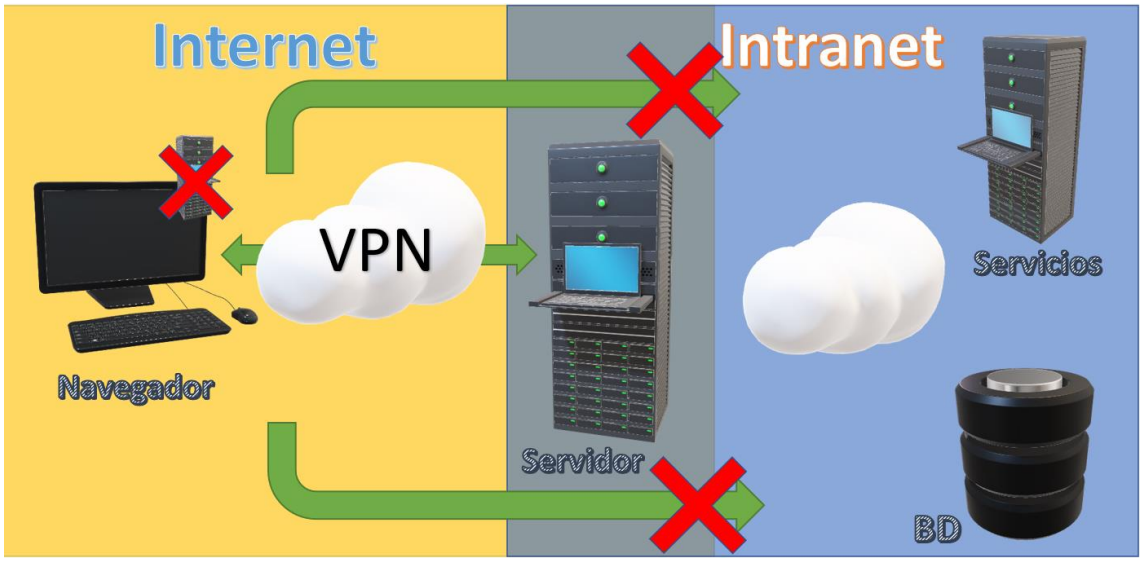


Ilustración 62: Trabajar desde casa a través de VPN

5. Validación:

Todas las pruebas realizadas durante el desarrollo se han anotado en un Excel llamado “Plan de pruebas Cédula Habitabilidad”.

Las pruebas se han realizado a mano sobre la aplicación desplegada en el Servidor remoto. Al final de las pruebas se anotan los resultados. Se corrigen aquellos fallos que han hecho fallar la prueba y se vuelven a ejecutar todas las pruebas.

El formato del plan de pruebas ha sido definido por la empresa. A continuación, voy a enumerar todos los campos que es necesario definir para cada prueba:

- Tarea GesHuman: Nombre de la tarea a la que hace referencia la prueba
- Prueba: Pequeño resumen de lo que se intenta probar
- Condiciones previas: El estado en el que se debe encontrar la aplicación para poder hacer la prueba
- Como establecer las condiciones previas: Que acciones realizar para llegar a las condiciones previas
- Pasos para realizar: Que se debe hacer para ejecutar la prueba

- Resultado esperado: Que comportamiento esperamos de la aplicación tras ejecutar la prueba
- Sentencia SQL: Añadirla en caso de que se utilice durante la prueba
- Datos de prueba: Por cada campo que se puebla u opción que se ejecuta hay que especificar exactamente los datos y las opciones para volver a probar con los mismos datos en caso de que falle la prueba.
- Fichero: en caso de que se esté trabajando con un fichero
- Fecha prueba
- Entorno: Si se realiza Enel servidor de producción PROD o desarrollo DES
- Resultado: OK si pasa la prueba o KO si no
- Observaciones: Pequeña descripción del error o por que se cree que falla

Tarea GesHuman	Prueba	Condiciones previas	Cómo establecer las condiciones previas	Pasos a realizar	Resultado esperado	Sentencia SQL	Datos de prueba	Fichero	Fecha prueba	Entorno	Tester	Resultado	Observaciones
Consulta de expedientes	Realizar una búsqueda con una combinacion de filtros que supere el limite de resultados permitidos	- Se debe tener acceso a la aplicación con un usuario válido - Se debe disponer de suficientes resultados en base a un filtro en la BD para que sobrepase el limite establecido	- Solicitar que el usuario se de de alta en la aplicación y se le asigne una ORVE - Añadir expedientes hasta tener mas que los permitidos en base a un filtro	-Acceder a la aplicacion - Pulsar en "Consulta de expedientes" - Introducir un filtro o combinacion de filtros que devuelva mas resultados de los permitidos - Pulsar en "Buscar"	* Se realiza la búsqueda y como el numero de resultados supera el limite salta el error que pide restringir la busqueda.								
							Usuario: Ilie Eduard Vlad Todos los campos de la busqueda vacios		08/04/2020	DES	Eduard	KO	El limite esta establecido en 100 pero no aparece el mensaje de error
Consulta de expedientes	Realizar una búsqueda con una combinacion de filtros lo suficientemente restrictiva que no supere el limite de resultados permitidos	- Se debe tener acceso a la aplicación con un usuario válido - Se debe disponer de suficientes resultados en base a un filtro en la BD para que sobrepase el limite establecido	- Solicitar que el usuario se de de alta en la aplicación y se le asigne una ORVE	-Acceder a la aplicacion - Pulsar en "Consulta de expedientes" - Introducir un filtro cuyos resultados no sobrepasen el limite establecido - Pulsar en "Buscar"	* Se realiza la búsqueda y como el numero de resultados no supera el limite se muestran los resultados poblados en una tabla..								
							Usuario: Ilie Eduard Vlad Desplegable Orve: Baztan-Bidasoa(1 resultado)		08/04/2020	DES	Eduard	OK	El limite esta establecido a 100
Consulta de expedientes	Realizar una búsqueda con un Via, Apellido1, Apellido2, Nombre que contenga un exceso de acentos y letras en mayusculas y menusculas con el objetivo de obtener un expediente concreto	- Se debe tener acceso a la aplicación con un usuario válido - Previamente se debe comprobar que para la combinacion de filtros que se quiere probar existe un expediente.	- Solicitar que el usuario se de de alta en la aplicación y se le asigne una ORVE	-Acceder a la aplicacion - Pulsar en "Consulta de expedientes" - Introducir un filtro con exceso de acentos - Pulsar en "Buscar"	* Se realiza la búsqueda y como debido al COLLATE todas las letras con acentos y/o mayusculas se reducen a la letra sin acento y obtenemos expedientes que existen en la BD								
							Usuario: Ilie Eduard Vlad Nombre: Nëd Primer Apellido: flândèrs Segundo Apellido: Flän Calle: ÄvÊnÌdÄ PÏÖ XÏÍ Portal: 1 Escalera: 1 Piso: 7º Puerta: b		08/04/2020	DES	Eduard	OK	El limite esta establecido a 100
Consulta de expedientes	Realizar una búsqueda utilizando el campo Año construccion hasta y que Año construccion desde quede vacio	- Se debe tener acceso a la aplicación con un usuario válido - Previamente se debe comprobar que para la combinacion de filtros que se quiere probar existe un expediente.	- Solicitar que el usuario se de de alta en la aplicación y se le asigne una ORVE	-Acceder a la aplicacion - Pulsar en "Consulta de expedientes" - Introducir un filtro solo para el campo Año construccion hasta - Pulsar en "Buscar"	* Se realiza la búsqueda satisfactoriamente sin necesariamente rellenar los dos limites del año de construccion								
							Usuario: Ilie Eduard Vlad Año construccion hasta: 1950		08/04/2020	DES	Eduard	OK	El limite esta establecido a 100
Consulta de expedientes	Realizar una búsqueda utilizando campos antiguos y nuevos y comprobar que se filtran los resultados teniendo en cuenta todos los campos	- Se debe tener acceso a la aplicación con un usuario válido - Previamente se debe comprobar que para la combinacion de filtros que se quiere probar existe un expediente.	- Solicitar que el usuario se de de alta en la aplicación y se le asigne una ORVE	-Acceder a la aplicacion - Pulsar en "Consulta de expedientes" - Introducir los filtros por los que queremos buscar - Pulsar en "Buscar"	* Se realiza la búsqueda satisfactoriamente teniendo en cuenta todos los campos rellenados para el filtrado								
							Usuario: Ilie Eduard Vlad Nº Expediente: 2020/00 Estado: Emitida Orve: Gobierno de Navarra Parcela: 13		08/04/2020	DES	Eduard	OK	El limite esta establecido a 100
Consulta de expedientes	Realizar una búsqueda utilizando solo campos nuevos y comprobar que se filtran los resultados teniendo en cuenta todos los campos	- Se debe tener acceso a la aplicación con un usuario válido - Previamente se debe comprobar que para la combinacion de filtros que se quiere probar existe un expediente.	- Solicitar que el usuario se de de alta en la aplicación y se le asigne una ORVE	-Acceder a la aplicacion - Pulsar en "Consulta de expedientes" - Introducir los filtros por los que queremos buscar - Pulsar en "Buscar"	* Se realiza la búsqueda satisfactoriamente teniendo en cuenta todos los campos rellenados para el filtrado								
							Usuario: Ilie Eduard Vlad Clase cédula: A (30 años) Portal: 2 Piso: 3 Puerta: d Poligono: 3 Parcela: 400 Subárea: 1 Unidad Urbana: 17 Año c. desde: 1970 Añoc. Hasta: 1980		08/04/2020	DES	Eduard	OK	El limite esta establecido a 100
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Autocompletado datos al introducir NIF	-Se debe tener un cetificado digital o Clave -Que el NIF que se quiere introducir este vinculado a una solicitud anterior	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar NIF - Completar NIF - Quitar la selección del campo NIF	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se carga el campo NIF * Se cargan los datos del solicitante en la pestaña.								
							Usuario: Ilie Eduard Vlad NIF : 12345		08/04/2020	DES	Eduard	OK	Como hay una multitud de solicitudes con ese NIF, el procediemiento devuelve correctamente los datos de la ultima solicitud que era de tipo Juridico, con telefono: 98765, Email: prueba.juridico@algo.com y Nombre: NombreJuridico
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Cambios de campos en formulario para Entidad Juridica	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar Entidad - Cambiar Entidad a Juridica	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se cambia el nombre del campo Nombre a Razon Solcial * Se ocultan los campos Apellidos * Se oculta la vista Notificaciones								
							Usuario: Ilie Eduard Vlad		08/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Creacion de una nueva solicitud para una Entidad Juridica	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar Entidad - Cambiar Entidad a Juridica - Completar los datos de la solicitud - Enviar solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se carga la pestaña "Responsabilidad" * Se cambia el nombre del campo Nombre a Razon Solcial * Se ocultan los campos Apellidos * Se oculta la vista Notificaciones * Se crea la nueva solicitud								
							Usuario: Ilie Eduard Vlad Localidad: Pamplona calle: CALLE ABEJERAS Portal: 1 Vivienda: 2ºA Dormitorios: 3 Baños: 2 Tipo Solicitante: Propietario Entidad: Juridico Check autorizo notificación telemática sin marcar NIF: 123456 Telefono: 123456 Razón Social: Pedro Y en declaración responsabilidad: No se han efectuado obras		14/04/2020	DES	Eduard	KO	Error inesperado al completar los datos y pulsar el boton a Enviar solicitud. Si se cambia la entidad a Fisica y se completa el campo apellido 1 si que se genera la nueva solicitud. (Nota: Este error no ocurre cuando se actualizan los campos)
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Cambios de campos en formulario para Entidad Fisica	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar Entidad - Cambiar Entidad a Fisica	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se cambia el nombre del campo Razon Solcial a Nombre * Se muestran los campos Apellidos * Se valida el campo Apellido1								
							Usuario: Ilie Eduard Vlad		08/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Validacion del campo Email	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar campo Email - Introducir una direccion no valida	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El cuadro Email con el borde rojo								
							Usuario: Ilie Eduard Vlad Email: estonoesunemail		08/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Validacion del campo Email	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar campo Email - Introducir una direccion valida	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El cuadro Email con la direccion introducida								
							Usuario: Ilie Eduard Vlad Email: correo@prueba.com		08/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Validacion del campo Notificaciones	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - NO Seleccionar tipo Notificacion - Enviar Solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se muestra un mensaje informando al usuario que debe seleccionar una forma de notificacion								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Validacion del campo Notificaciones	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Seleccionar tipo Notificacion a otra direccion postal - Enviar Solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se muestra tres mensajes de error informando al usuario que debe introducir una direccion, una provincia y un codigo postal								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Check Autorizo a que me notifiquen telematicamente marcado por defecto al crear nueva solicitud	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete esta marcado								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (Ciudadano) - Pestaña Solicitante	Check Autorizo a que me notifiquen telematicamente marcado por defecto al crear nueva solicitud	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete esta marcado								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	Se puede crear una nueva solicitud con el check Autorizo a que me notifiquen telematicamente desmarcando	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Desmarcar el check Autorizo a que me notifiquen telematicamente - Rellenar el resto de campos - Enviar la solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete esta marcado * La solicitud se ha creado satisfactoriamente								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (Ciudadano) - Pestaña Solicitante	No se puede crear una nueva solicitud con el check Autorizo a que me notifiquen telematicamente desmarcando	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Desmarcar el check Autorizo a que me notifiquen telematicamente - Rellenar el resto de campos - Enviar la solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete no esta marcado * Se muestra un error al ciudadano indicando que debe marcar el check Autorizo a que me notifiquen telematicamente								
									14/04/2020	DES	Eduard	KO	Un Error desconocido hace que no funcione el boton Enviar Solicitud
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	El campo Email se deshabilita si se desmarca el check Autorizo a que me notifiquen telematicamente	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Desmarcar el check Autorizo a que me notifiquen telematicamente	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete esta desmarcado * El campo Email esta deshabilitado								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (Ciudadano) - Pestaña Solicitante	El campo Email se deshabilita si se desmarca el check Autorizo a que me notifiquen telematicamente	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Desmarcar el check Autorizo a que me notifiquen telematicamente	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete esta desmarcado * El campo Email esta deshabilitado								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Solicitante	El campo Email se guarda vacio o nulo si se crea una solicitud sin marcar el check Autorizo que me notifiquen telematicamente	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Solicitante - Desmarcar el check Autorizo a que me notifiquen telematicamente - Completar los demas datos - Crear una nueva solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * El check Autorizo a que me notifiquen telematicamnete esta desmarcado * El campo Email esta deshabilitado y vacio * En la base de datos el campo Email de la tabla SolicitanteCH esta a NULL								
						SELECT * FROM [HabitaMigracion].[dbo].[SolicitanteCH] Where [NIFCIF] = '44431146e' and [IdSolicitanteCH] =306769	Se ha editado la solicitud con IdSolicitante 306769 y NIFCIF 306769		14/04/2020	DES	Eduard	OK	
(*)Visualizar la solicitud en curso (Ciudadano)	Los unicos cambios permitidos son el cambio de la forma de notificacion.	Se debe tener una solicitud en curso	Realizar la peticion de una nueva solicitud	- Acceder a la aplicación - Pulsar en "Consulta de solicitudes" - Seleccionar solicitud en curso - Cambiar cualquier campo menos forma de notificacion - Guardar los cambios	* Se carga la página de solicitud * No se ha modificado ningun campo que no sea forma de Notificacion								
(*)Visualizar la solicitud en curso (Interno)	Puede cambiar datos en cualquier momento	Se debe tener una solicitud en curso	Realizar la peticion de una nueva solicitud	- Acceder a la aplicación - Pulsar en "Consulta de expedientes" - Buscar por un filtro - Seleccionar expediente - Cambiar cualquier campo menos forma de notificacion - Guardar los cambios	* Se carga la página de solicitud * Se han guardado los cambios realizados								
						SELECT * FROM [HabitaMigracion].[dbo].[SolicitanteCH] Where [NIFCIF] = '44431146e' and [IdSolicitanteCH] =306769	Se ha editado la solicitud con IdSolicitante 306769 y NIFCIF 306769 Cambios: - Numero aseos: 3 - Nombre: Cambio		14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (Interno) - Pestaña Direccion	Enlace InformacionVivienda con campo Referencia Catastral vacio	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña direccion - Seleccionar Informacion vivienda	* Se carga la página de solicitud * Se carga la pestaña "Direccion" * Se abre una nueva pestaña direccion al servicio de vivienda.								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (Interno) - Pestaña Direccion	Enlace InformacionVivienda con campo Referencia Catastral completado	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña direccion - Completar Direccion para obtener referencia catastral - Seleccionar Informacion vivienda	* Se carga la página de solicitud * Se carga la pestaña "Direccion" * Se abre una nueva pestaña direccion a la informacion de la vivienda según la referencia catastral.								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Probar si se guardan los datos de los nuevos campos Aseos, Superficie, Poligono, Parcela, Subarea, Unidad Urbana	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Vivienda - Completar todos los campos de la solicitud - Completar los campos Aseos, Superficie, Poligono, Parcela, Subarea, Unidad Urbana	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se carga la pestaña "Responsabilidad" * Los datos que se han introducido a mano se guardan								
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Validacion del campo superficie	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Vivienda - Completar todos los campos de la solicitud - Completar el campo solicitud con un valor muy grande por debajo de la mascara	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se carga la pestaña "Solicitante" * Se carga la pestaña "Responsabilidad" * Se crea la nueva solicitud								
							Superficie: 999999999		14/04/2020	DES	Eduard	KO	Error Inesperado. En cambio si se introduce un valor por debajo de 100000000 si se crea la solicitud.
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Comprobacion de la mascara del campo superficie	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Navegar a la pestaña Vivienda - Introducir mas numeros de los permitidos en la mascara	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * La mascara no permite introducir mas numeros de los permitidos								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Los campos Aseos y Superficie se guardan en la BD	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Crear Solicitud" - Completar campos de la solicitud - Rellenar los campos - Completar los campos Aseos y Superficie - Enviar solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se genera una nueva solicitud * Los datos de los campos Superficie y Aseos se guarda en BD								
							Aseos: 99 Superficie: 12345		14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Los campos Aseos y Superficie se recuperan de la BD	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Acceder" seccion Bandeja de Entrada - Seleccionar Expediente 2020/00068 - Pulsar boton editar	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se cargan los datos del expediente guardados en la BD								
									14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Los campos Aseos y Superficie se actualizan cuando se modifican los datos de la solicitud	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Acceder" seccion Bandeja de Entrada - Seleccionar Expediente 2020/00068 - Pulsar boton editar - Cambiar Aseos a 11 - Cambiar Superficie a 300 - Actualizar Solicitud - Volver a cargar Solicitud	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Se cargan los datos del expediente actualizado.								
							Aseos: 11 Superficie: 300		14/04/2020	DES	Eduard	OK	
Solicitar una nueva cédula de habitabilidad para una vivienda (INTERNO) - Pestaña Vivenda	Probar si se cargan los datos de los nuevos campos Aseos, Superficie, Poligono, Parcela, Subarea, Unidad Urbana despues de crear una nueva Solicitud	Se debe tener un cetificado digital o Clave	Solicitar el certificado a la entidad correspondiente.	- Acceder a la aplicación - Pulsar en "Acceder" seccion Bandeja de Entrada - Seleccionar Expediente 2020/00068 - Navegar a la pestaña Vivienda	* Se carga la página de solicitud * Se carga la pestaña "Vivienda" * Los campos Poligono, Parcela, Subarea y Unidad urbana se pueblan con los datos de BD								
							Expediente: 2020/00068		14/04/2020	DES	Eduard	OK	

6. Conclusiones y líneas futuras

6.1. Conclusiones:

El resultado final del trabajo ha sido muy enriquecedor ya que me ha permitido tener una primera aproximación a la tecnología .NET de Microsoft. He adquirido los suficientes conocimientos como para poder desarrollar una aplicación Web de complejidad media.

También ha servido como repaso y puesta en práctica de los numerosos conocimientos adquiridos durante la carrera, especialmente las asignaturas de Programación Avanzada, Bases de Datos e Sistemas de Información Web.

Se puede decir que he alcanzado todos los objetivos propuestos inicialmente, tanto por el desarrollo de la aplicación como por los conocimientos adquiridos.

6.2. Líneas futuras:

Se ha conseguido una aplicación que implementa la mayoría de las funcionalidades totales planificadas ya que el proyecto aún se sigue desarrollando. Sin embargo, existe una serie de mejoras o añadidos que se pueden realizar para mejorar el producto final.

Por ejemplo, sería interesante que cuando un ciudadano quiera solicitar una nueva cédula, que los datos personales se puedan poblar solos en la pestaña Solicitante basándose en la cuenta de usuario con la que está accediendo.

Por otro lado, sería interesante añadir una opción para el ciudadano en forma de un check que permita enviar de forma telemática un correo de notificación para renovar la cédula cuando la cédula de habitabilidad está a dos meses de caducar, aún no hay una solicitud en curso para esa vivienda y el propietario no ha cambiado.

Glosario:

- **Entidad**: clase que tiene una identidad más allá de sus atributos, esta identidad se le asigna con un identificador. Dos objetos de tipo usuario con el mismo nombre no significa que sea el mismo usuario. Los valores de loa tributos pueden cambiar.
  Arquitectura orientada al dominio: Tipo de arquitectura orientada a objetos que permite construir aplicaciones empresariales complejas con una vida relativamente larga y con un volumen de cambios evolutivos considerable. En estas aplicaciones es muy importante todo lo relativo al mantenimiento, facilidad de actualización, sustitución de tecnologías o frameworks. El objetivo es poder realizar todos estos cambios con el mínimo impacto posible en el resto de la aplicación concretamente a la capa del dominio de la aplicación. (Iberica, Microsoft, s.f.)

- **ASP.NET:** Es una framework open source, multiplataforma, creada por Microsoft para construir aplicaciones web modernas y servicios con .NET. Está constituido por herramientas, lenguajes de programación, librerías para construir todo tipo de aplicaciones. (Microsoft, s.f.)

- **Bootstrap 3:** Es una biblioteca multiplataforma de código abierto basado en HTML, CSS y Javascript para el desarrollo front-end de una web. (Wikipedia, s.f.)

- **Cédula Habitabilidad:** Documento extendido por el Departamento de Vivienda del Gobierno de Navarra que garantiza al comprador de una vivienda que el inmueble cumple con la normativa sobre habitabilidad para que la construcción pueda ser considerada vivienda y por tanto apta para uso residencial. Es obligatoria para la contratación de servicios como la luz y el gas. (Vivienda, s.f.)

- **Arquitectura de N capas:** Arquitectura que divide una aplicación en capas lógicas y niveles físicos. De esta forma se separan responsabilidades y se pueden administrar las dependencias. Cada capa tiene una responsabilidad específica. Una capa superior puede utilizar los servicios de una capa inferior, pero no al revés.

Referencias

Documentacion, M. S. (s.f.). *https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms144250(v=sql.105)?redirectedfrom=MSDN*.

Google. (17 de 10 de 2020). *Google Trends*. Obtenido de https://trends.google.es/trends/explore?cat=13&date=2004-01-01%202020-10-17&q=%2Fm%2F03clphw,Java%20Web

Iberica, Microsoft. (s.f.). *Guia Arquitectura N capas orientada al dominio con .NET 4.0* . Obtenido de https://onedrive.live.com/view.aspx?cid=FA91E0517B4176A3&authKey=%21ANuqfYfQ6oYKWT0&resid=FA91E0517B4176A3%214439&ithint=%2Epdf&open=true&app=WordPdf

Microsoft SQL Documentacion. (s.f.). *https://docs.microsoft.com/es-es/sql/t-sql/statements/collations?view=sql-server-ver15*.

Microsoft. (s.f.). *What is ASP.NET?* Obtenido de https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet

Navarra, G. A. (13 de 10 de 2020). *Servicios mas consultados*. Obtenido de https://gobiernoabierto.navarra.es/es/open-data/datos/servicios-mas-consultados

Noticias de Navarra. (18 de 01 de 2019). *Tres de cada cuatro navarros se conecta diariamente a internet*. Obtenido de https://www.noticiasdenavarra.com/actualidad/sociedad/2019/01/18/tres-cuatro-navarros-conecta-diariamente/802106.html

SQL, M. D. (s.f.). *https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms180175(v=sql.105)*.

Stats, I. L. (2020). *Internet Live Stats*. Obtenido de https://www.internetlivestats.com/total-number-of-websites/

Tartas, E. B. (2018). Bases de datos II Tema 4. Pamplona, Navarra, España.

Telecomunicaciones, D. G. (2019). *Estandates tecnologicos del Gobierno de Navarra.* Pamplona. Obtenido de http://internet.gccpublica.navarra.es/DGIT/SistemasInformacionGN/Documentacion/Escenario%20Tecnologico%20Gobierno%20de%20Navarra%20Febrero%202019%20v1.pdf

Vivienda, D. d. (s.f.). *Cédula de Habitabilidad*. Obtenido de http://www.navarra.es/home_es/Temas/Vivienda/Ciudadanos/Compraventa/Antes+de+comprar+una+vivienda/Cedula+de+habitabilidad/

w3Techs. (17 de 10 de 2020). *Usage statistics ana market share of jQuery for websites*. Obtenido de https://w3techs.com/technologies/details/js-jquery#:~:text=jQuery%20is%20used%20by%2097.1,is%2076.5%25%20of%20all%20websites.

Wikipedia. (s.f.). *Bootstrap(front-end framework)*. Obtenido de https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)