MainLoadPro: Consolidación digital de estimaciones en entorno empresarial.

Raimundo Prieto Alonso
Grado de Ingeniería Informática *(Ingeniería del Software)*
Desarrollo Web

Profesor consultor y tutor: **Gregorio Robles Martínez**
Profesor responsable asignatura: **Santi Caballé Llobet**

Junio 2023

FICHA DEL TRABAJO FINAL

Título del trabajo:	*MainLoadPro*
Nombre del autor:	*Raimundo Prieto Alonso*
Nombre del consultor/a:	*Gregorio Robles Martínez*
Nombre del PRA:	*Santi Caballé Llobet*
Fecha de entrega (mm/aaaa):	06/2023
Titulación:	*Grado de Ingeniería Informática – Ingeniería del Software*
Área del Trabajo Final:	*Desarrollo web*
Idioma del trabajo:	*Castellano*
Palabras clave	*Planificación, Validación, Consolidación*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

MainLoadPro es una herramienta web que pone en práctica las competencias adquiridas durante la realización del Grado de Ingeniería Informática en la especialización de Ingeniería del Software, para la cual se han desarrollado los conocimientos adquiridos en las diferentes asignaturas realizadas.

La implementación de esta aplicación proviene de la necesidad de una herramienta capaz de digitalizar y validar los procesos de consolidación de las planificaciones de carga de trabajo asociadas a los proyectos desarrollados en un entorno empresarial.

Siendo el objetivo principal el almacenado de la información relativa a planificaciones validadas por los departamentos que sirvan para la consolidación de tiempos estimados de trabajo y evaluar procesos de carga capacidad asociados al desarrollo de los proyectos existentes o futuros de la compañía, así como la futura explotación de datos almacenados.

Adicionalmente a la implementación de una interfaz amigable al usuario, el caso de éxito corresponderá a la codificación de una base de datos con altas capacidades de potencial para almacenar grandes cantidades de datos, e igual agilidad para la recopilación y explotación de datos.

Abstract (in English, 250 words or less):

MainLoadPro is a web tool that take into practice the skills acquired during the Computer Engineering Degree in the Software Engineering specialization, for which the knowledge acquired in the different areas taken has been developed.

The implementation of this application born from the need for a tool capable of digitizing and validating the consolidation processes of the workload schedules associated with the projects developed in a business environment.

The main objective being the storage of information related to planning validated by the departments that belongs to consolidate estimated workload and evaluate workforce processes associated with the development of existing or future project developments of the company, as well as future exploitation of stored information.

In addition to the implementation of a user-friendly interface, the success story will correspond to the creating a database with high potential capacities to store large amounts of information also agility for data gathering.

Palabras clave (entre 4 y 8):

Planificación, Carga de Trabajo, Validación, Consolidación, Capacidad

A mis canijas, lo conseguimos.
"No hay nada imposible".

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

El proyecto toma nombre de la propia solución de software a desarrollar mainLoadPro, y quiere ser primer paso de una transformación digital en la gestión de los procesos asociados al balance de carga de trabajo. Estos procesos pueden ser adoptados por una empresa de tamaño grande que se embarca en el proceso de digitalización y transformación de las herramientas que ayudan a mantener una trazabilidad en tiempo pasado y futuro de las cargas y estimaciones de los departamentos que componen la empresa, siendo este el visor de la consolidación de horas y costes de los trabajos en curso o futuros.

Las motivaciones para la elección de este desarrollo frente a otros posibles es la propuesta sobre una real capacidad de mejora de los procesos de estimación y costes que se deben mantener de manera anual por los diferentes proyectos que una empresa lleve a cabo. Durante mi trayectoria profesional, he podido experimentar cómo empresas que tiene varias áreas de negocio y una gran división departamental usan herramientas como Excel para realizar una trazabilidad de los trabajos que han realizado o tienen pendientes de realizar; es decir, el plan anual de horas y costes, o plan operativo OP, el uso de esta herramienta (Excel) de manera compartida da infinidad de problemas de integridad en los datos, lo que hace que las labores de convergencia de los datos aportados entre departamentos, sea una labor costosa y tediosa para el departamento final de costes o planificaciones, sin poner en manifiesto posibles problemas en la corrupción de las versiones, y nulo aseguramiento de los datos ya validados. Todo esto se traduce en una estimación de horas y costes muy alejada de la naturaleza de los procesos de la empresa. Lo que se transforma en desviaciones de miles de horas, es decir, millones de euros.

1.2 Objetivos del Trabajo

Para este proyecto se destacan 2 principales objetivos subdivididos en múltiples tareas para ser alcanzados.

Almacenar de información relativos a la consolidación de un ejercicio de carga de horas de trabajo asociadas a un entorno empresarial

- Librerías de negocio y de acceso a datos que permitan la comunicación, almacenamiento y representación de los datos con una interfaz de cliente amigable.
- Base de datos relacional capacitada para gestión y almacenamiento de grandes volúmenes de información asociada a las planificaciones estratégicas del sector empresarial.

Representar de la información.

- Interfaz de representación de datos ágil y amigable, que permita la visualización, edición y validación de los datos introducidos mediante perfiles de usuario autenticados, elaborando un ciclo de trabajo evolutivo de las

planificaciones que conforman la consolidación de la carga de proyectos de una compañía.

Alcance

Considerando como alcance del desarrollo la realización de una interfaz que mediante acceso autenticado e identificado según perfil de acceso, sea capaz de mostrar, por anualidades y a través de paneles desplegables según criterio de navegación del usuario, las horas registradas por cada uno de los departamentos a la estructura de proyectos previstos para ese ejercicio, y desde esta "parrilla" de datos se pueda tanto realizar la introducción de datos, a nivel de la unidad atómica identificada, que es el departamento de trabajo. Así mismo se requerirá la validación, a nivel departamental o por proyecto, según la fase definida, donde intervendrá cada rol de actor en su versión autorizada, para esta manera lograr el asentamiento y consolidación de los datos de la versión para futuras revisiones conllevando una aceptación digital de los presupuestos planificados.

1.3 Enfoque y método seguido

La estrategia implementada para este proyecto consiste en la construcción de un software nuevo con la base aportada por librerías y frameworks más adelante indicados que permiten un desarrollo más acelerado y una mayor eficacia en la elaboración de un amigable entorno de trabajo para el usuario final de la herramienta.

Dado que se trata de un proyecto de digitalización de datos y herramientas que ofrezcan un valor añadido a la estructuración de información planificada, el desarrollo de una aplicación ágil que se adapte a las necesidades del cliente es fundamental para lograr el éxito y evolución del aplicativo. Por ello, la realización de un software a medida que permita un gran grado de personalización es la clave en la selección de un producto de nuevo desarrollo.

Para conseguir este enfoque identificado, necesitaremos la implementación de una herramienta web de estimación de horas/costes de trabajo que recoja los diferentes departamentos de la compañía y para cada uno de los proyectos que conformen el ejercicio de negocio de la compañía, permitiendo la creación de diferentes versiones, en la que cada perfil correspondiente para esa versión pueda rectificar y validar los datos existentes.

Adicionalmente, estas estimaciones de horas reflejadas, al estar realizada por los propios departamentos (contará con un número de personas en dicho departamento), puede detectar de manera prematura posibles carencias o excesos de personal para la realización de las tareas.; es decir, una detección precoz de falta de personal o de la ausencia de suficiente carga de trabajo.

Sin duda, la apuesta por el inicio en la creación de este un software a medida, puede ser de gran utilidad para departamentos encargados de la planificación general de costes de una compañía y permitirá el crecimiento y evolución dicho software acorde los procesos empresariales que identifican al departamento y la empresa en cuestión.

1.4 Planificación del Trabajo

A continuación, se describen los elementos relativos a la planificación del trabajo, seccionada en varios subapartados que contienen información sobre elementos del proyecto a desarrollar, tecnologías aplicadas en la implementación, planificación temporal de hitos, así como evaluación de posibles riesgos.

1.4.1 Elementos a desarrollar

Áreas del producto identificadas:

- Acceso y credenciales de usuario.

- Selector de anualidades y perfiles.

- Panel visualización "parrilla" de identificación de proyectos y departamentos.
  - Agrupador de proyectos
  - Agrupador de departamentos
  - Componentes detalle a nivel atómico de los proyectos
  - Filtros de búsqueda de entidades proyecto y departamento.
  - Filtro de visualización de versiones/validaciones.
  - Acción de validación y versionado.
  - Detalle de edición e introducción de datos
  - Detalle informativo de los comentarios asociados a cada dato y valor en versiones previas.

- Administración (Áreas, proyectos y departamentos)

1.4.2 Tecnologías utilizadas

A continuación, se detallan las tecnologías y lenguajes utilizados en el desarrollo del proyecto MainLoadPro:

- Aplicación Web desarrolladas bajo *ASP.Net MVC* [1] y lenguaje *C#*.[2]
  Mediante la versión licenciada para estudiantes, denominada *Visual Studio Community*[3], la cual permite la integración de cualquier componente y desarrollo que permitiría en su versión profesional, pero orientada a un entorno académico como en el que nos encontramos y para el cual está orientada la licencia del software.
  Desarrollo de un proyecto distribuido en las N-Capas clásicas del desarrollo web, donde se diferencia las capas de acceso a base de datos, de la lógica de negocio y representación, pero orientado a un entorno *Modelo Vista Controlador (MVC)*[4], desde el que mediante un modelo contextualizado de la base de datos se representa, a través mediante la acción de un controlador de operaciones compilado, las vistas que se usarán como interfaz de comunicación con el aplicativo y sus operaciones.

- Implementación de librerías *API web C#* [5] de consulta por los elementos del aplicativo.

Basado en el mismo concepto anterior, se desarrolla servicio REST[6] de operaciones que bien puede ser distribuido en otra máquina mediante orquestador de servicios o formar parte de una capa librería de lógica de procesos y datos (DLL)

- Comunicación de acceso ágil a las librerías y funcionalidades con JQuery / Ajax[7].
  Mediante el framework que JQuery dispone para un acceso rápido y ágil a las operaciones ofrecidas por el servicio REST mediante protocolo de comunicaciones asíncronas mediante tecnología Ajax.

- Capa de presentación implementada con *Bootstrap*[8] y framework cliente JQuery.
  Mediante el aprovechamiento de comunicación entre funcionalidades licencia MIT de terceros basadas en diseño Bootstrap que permite una representación enmarcada o "responsive" de los contenidos, integrada con los procesos de consulta y representación de JQuery.

- Capa de interconexión y mapeo con BBDD (ORM) bajo *EntityFramework 6* [9].
  Si bien parte de los procesos de conexión a BBDD estarán implementados mediante procedimientos almacenados por la rapidez de ejecución de consultas (pesadas o no) de estos frente al uso de entidades *ORM*[10], del cual aprovechamos las ventajas de creación de un modelo fuertemente tipado de conexión, procesamiento y devolución de los datos. De esta manera, pueden ser interpretados de manera más optima y manejable por vistas y controladores del aplicativo.

- Capa de base de datos bajo el motor de base de datos *MySQL*[11]
  Aprovechando las capacidades de gestión adquiridas por la última versión MySQL8, adquirida por Oracle[12], en la cual aparecen mejoras de rendimiento de procesos sobre el motor *innodb*[13]. Mediante la realización de indexados de búsqueda y particionado de tablas para el aprovechamiento del rendimiento adquirido en esta última versión 8.0.20 de MySQL.

- Gantt Project[14] y herramientas Kanban[15] (Miro) para la gestión de tareas, actividades y tiempos.

1.4.3 Planificación temporal

En base a las áreas a desarrollar identificadas, se establecen las siguientes grandes fases en el desarrollo.

Fases principales:

- Análisis de procesos, actores e interesados de la solución del producto.
- Identificación de roles y versiones que realizarán cada uno de los perfiles identificados. (Modelo de negocio).
- Identificación y confección del modelo de datos necesario para las actividades a implementar. (MySQL o MariaDB[16])

- Desarrollo de esqueleto de la aplicación (ASP.Net MVC), interfaces y clases bases que conformarán la herramienta. Acompañado por un Login que identifique usuario y perfil. (C#)
- Desarrollo de las APIs de operaciones que ofrecerán obtención y almacenamiento de la información de procesos. (C#)
- Desarrollo de interfaz de usuario que permita la usabilidad de la herramienta. (JQuery, Bootstrap, MVC)
- Pruebas de testeo unitario y sobre el aplicativo.
- Recodificación de partes necesarias.

Fases secundarias (opcionales, según posibilidades temporales del proyecto)

- Implementación de proyecto de Tests de las APIs desarrolladas.
- Despliegue de la herramienta y complementos en servicios de la nube (Azure[17]).

Futuribles deseados

- Desarrollo de pequeño módulo de interpretación inteligente de datos, que sea capaz de aproximar estimaciones futuras en base a los datos almacenados de anualidades anteriores y parámetros externos por definir, por ejemplo, índice de inflación anual, índice de absentismo laboral, índices de bajas, etc

Planificación y Gantt

En la planificación elaborada para este proyecto se han tenido en cuenta además de los parámetros temporales, circunstancias externas al proyecto como la consecución de otras asignaturas como parte de proyectos interferentes en la evolución discrecional del proyecto. Es por ello por lo que se han interpretado jornadas tanto lectivas como festivas, a sabiendas que el tiempo para la realización del proyecto será un recurso necesario y carente, determinado por la naturaleza intensiva del producto a desarrollar.

Adicionalmente, en la interpretación Gantt[18] de la estimación temporal, se puede observar diferentes colores de grupos de tareas, estos equivalen a la representación de los Sprint a realizar para la consecución del proyecto, de esta manera, podemos hacer seguimiento de los 4 Sprints[19] principales dedicados a desarrollo más las correspondientes fases de pruebas asociadas al propio proyecto.

Debido a la naturaleza cambiante en los inicios del proyecto y la aparición de factores externos durante el inicio de las tareas de desarrollo, parte de las tareas se han visto afectadas en su fecha de inicio o T0, tomando por tanto la determinación de acortar las fases de desarrollo en las tareas indicadas sobre el diagrama, lo que implica el aporte de un esfuerzo extra para el cumplimiento de fechas estimadas.

A continuación, se muestra una imagen que detalla la planificación Gantt generada y adicionalmente se incluye exportación PDF anexada desde

GanttProject[20]. Para más detalle de las tareas identificadas, visitar anexo al documento del informe de la planificación.

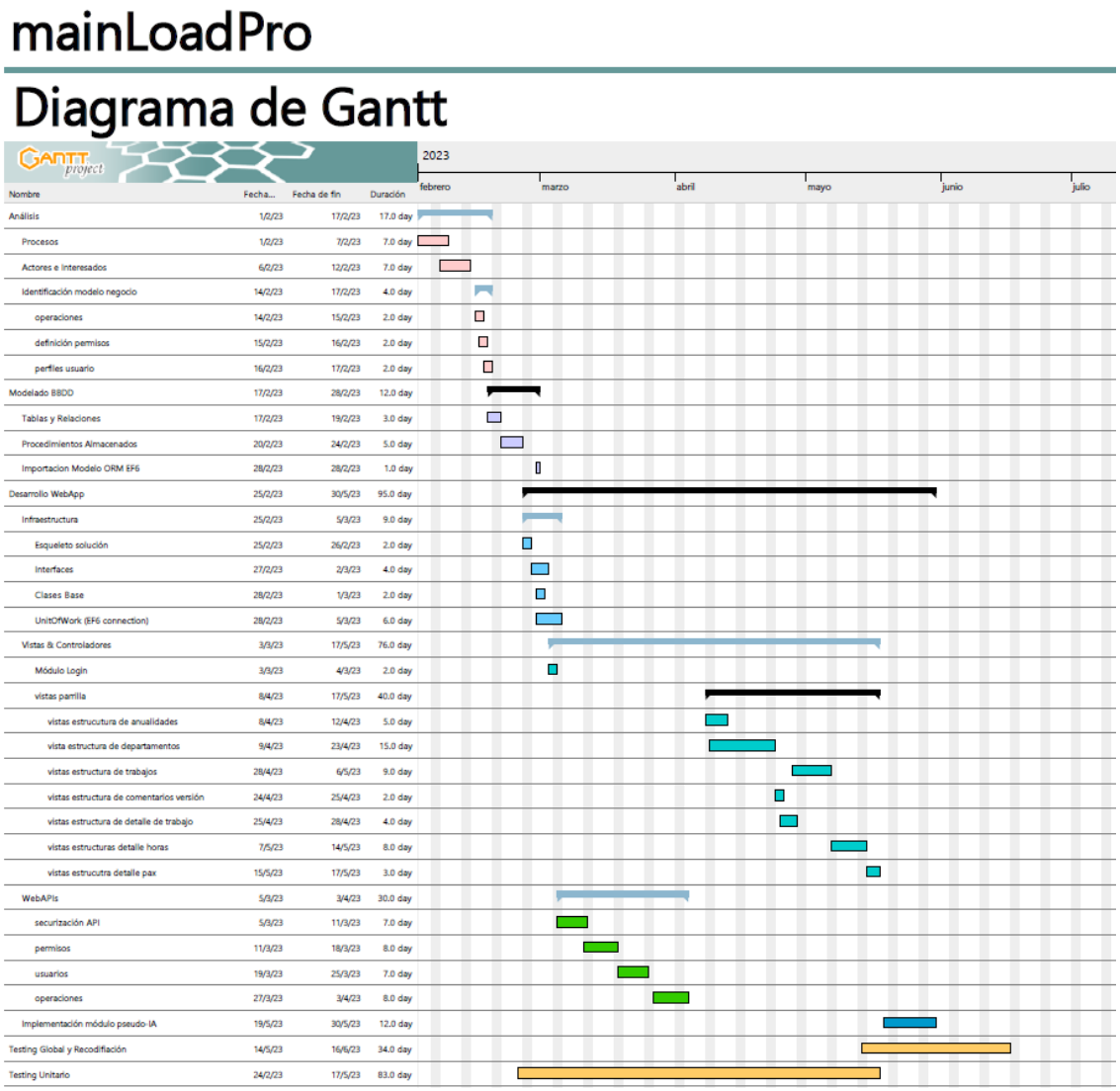


Ilustración 1. Detalle de planificación inicial al T0 del proyecto.

1.4.4 Evaluación de riesgos

Los riesgos que a priori fueron identificados corresponde con la sobrecarga de actividades que pueda tener el proyecto, y posibles desviaciones temporales que impidan que se realice alguna de las fases identificadas, siendo quizás la de más alto riesgo negativo la consecución de las fases secundarias del proyecto, en las que se encuentra la implantación de todos los Test de desarrollo, despliegue de la herramienta en plataforma de la nube, y fase futurible de los desarrollos asociados a la implantación de un módulo de aproximación inteligente de planificaciones, tal y como anteriormente fue indicado futuribles deseados.

La mitigación interpretada pasaba o por la ampliación de recursos (descartado) o acotamiento de las funcionalidades a metas más tangibles y realistas posibles,

para realizar en el tiempo previsto. Cumpliendo con la entrega de un desarrollo cerrado y funcional en todas las funcionalidades presentadas.

1.5 Breve sumario de productos obtenidos

La aplicación mainLoadPro representa una herramienta web de consolidación de las estimaciones horarias en un plan operativo sobre un ejercicio anual determinado, que corresponden a la realización de los proyectos a realizar por una empresa, seccionados en diferentes departamentos que componen la empresa, mostrando de esta manera, mediante el workflow entre diferentes actores/perfiles de la herramienta, la consolidación de las diferentes versiones del trabajo a realizar, obteniendo como resultado una estimación general de las horas que representa cada uno de los proyectos, presentes o futuros, pudiendo evaluar de esta manera costes, carga de trabajo y capacidad de trabajo necesaria para acometer los proyectos establecidos, etc.

El proyecto principal realizado en ASP.Net MVC se comunica con otras capas del proyecto relativas al negocio y el acceso a datos, adicionalmente las acciones de mantenimiento CRUD[21] usarán métodos API expuestos por el correspondiente proyecto.

1.6 Breve descripción de los otros capítulos de la memoria

- Análisis, diseño y principales funcionalidades.
  Apartado en el que se muestra en detalle de las necesidades de alcance del proyecto y requisitos para la cumplimentación de las funcionalidades asociadas al producto.

- Arquitectura[22] implementada.
  Apartado en que se describen las características de las metodologías, tecnologías y artefactos empleados para la completa implementación del proyecto.

- Preparación de entorno de desarrollo e implementación.
  Sección en la que se muestran los principales elementos empleados para crear un entorno de desarrollo en base a las necesidades generales del proyecto, así como aplicaciones de software y características establecidas en cada uno de ellos.

- Elementos del proyecto.
  Apartado con la descripción general de los elementos empleados para crear la solución de proyecto, clases, interfaces, modelos, vistas, etc.

- FrontEnd[23].
  Relación de artefactos y tecnología empleada en la realización de las vistas de cliente que explotarán los datos ofrecidos por el motor de elementos del servidor para una amigable y cercana interfaz de usuario.

- BackEnd[24].
  Apartado en el que se destacan las principales implementaciones realizadas en el proyecto, que permiten la seguridad de contenidos, visualización y, obtención y catalogación de datos.

- Vistas e Interfaces de usuario.
  Apartado en el que se muestran las principales vistas, interfaces o pantallas que permiten la comunicación del cliente final con la aplicación.

- WebAPI[25].
  Sección en la que se describen los métodos WebApi creados para explotación propia del producto o acceso de terceros, y expuestas mediante el uso de swagger[26].

- Ejecución y despliegue en Azure.
  Apartado en el que se destacan los principales características y configuraciones empleadas para la realización del despliegue de la aplicación y base de datos en la nube de Azure.

2. Análisis, diseño y funcionalidades

2.1 Análisis del alcance y objetivos del proyecto.

El aplicativo debe contar con:

- Estructura de Proyectos (**WBS**), que permita jerarquizar cada uno de los proyectos en sus líneas de negocio y áreas. Estos proyectos, deben constar con una división por sedes, que la empresa puede tener repartidas por territorio nacional e internacional, cada una de estas sedes, puede acometer ciertos trabajos relacionados con el mismo proyecto, pero las casuísticas económicas de cada sede deben de ser interpretadas de manera individual. Debido a la naturaleza de incertidumbre de los proyectos, estos estarán catalogados como proyectos asegurados (A) o como proyectos futuros (F), ambos cuales contarán con un valor porcentual de la probabilidad de formación de este, siendo este valor tratado como peso porcentual asociado a las horas de trabajo estimadas, solo para determinados campos de los informes.

- Estructura de Departamentos (**OBS**), que permita jerarquizar los departamentos que componen la empresa, cada uno de estos debe estar asociado a una sede de trabajo, permitiendo que dentro de un departamento existan diferentes unidades de negocio.

- **Sedes** (Sites), información transversal al aplicativo no indexado por ejercicio, y que representa las diferentes ciudades nacionales o internacionales en las que la empresa tiene negocio.

- **Usuarios:** identificados por perfil y ejercicio en que realizar las consultas y modificaciones. Cada usuario debe tener mediante los correspondientes permisos, accesos a los diferentes proyectos y departamentos que se considere oportuno según sus funciones.
  Existiendo los perfiles de Administrador, Planificador (Lectura y Escritura y Validación), jefe Departamental y jefe de Proyectos, cada uno de ellos, a excepción del administrador, estará asociado a la responsabilidad de inclusión y verificación de datos introducidos en cada versión. Desde la parrilla principal de datos se debe permitir el cambio entre los diferentes ejercicios y perfiles que contenga el usuario logado en la aplicación.

- **Versiones** (Baselines): cada una de las existentes, permite la congelación de los datos asociadas a las validaciones realizadas por el perfil específico que le corresponda, de esta manera consolidando datos horarios relativos a la carga de trabajo estimada. Siendo estas tuplas versión-perfil las siguientes:

  Baseline 0 – Planificador Write
  Baseline 1 – Jefe Departamental
  Baseline 2 – Jefe de Proyectos
  Baseline 3 – Planificador Valid

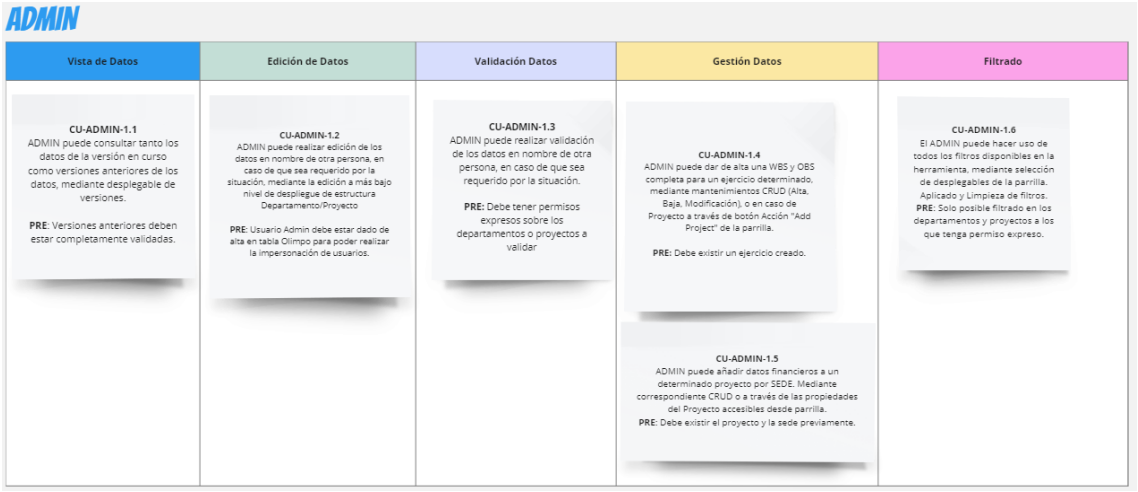
- **Estimaciones** horarias: almacenado de las horas estimadas, siendo la mínima granularidad representada por los datos ejercicio, versión, departamento, proyecto y año. Cada estimación horaria tendrá asociadas uno o varios comentarios relativos a la edición y validación de los datos.

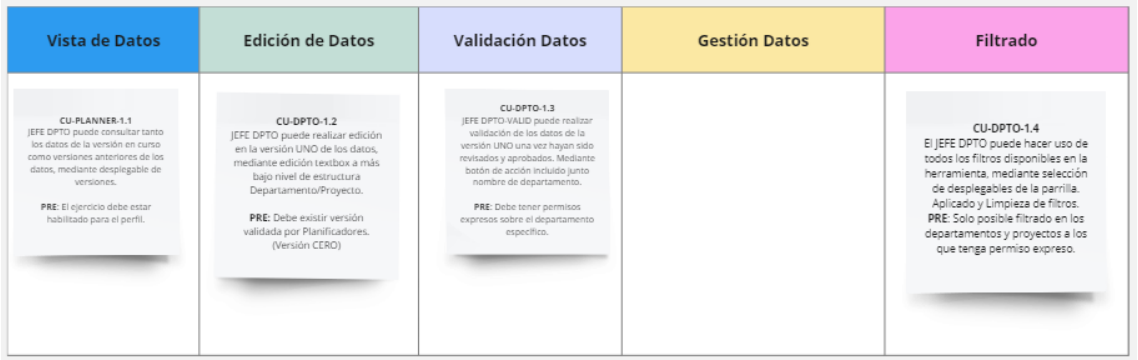
- **Filtrado**: los datos mostrados en la parrilla principal deben de poder ser filtrados en base a los criterios de granularidad anteriormente indicados, y por ello cada filtro debe estar asociado a los permisos del usuario que está realizando la operativa.

- **Administración:** la aplicación debe contener una zona administrativa donde usuarios con privilegios puedan dar de alta los usuarios, sedes, ejercicios, departamentos, proyectos, de manera que sea totalmente autónoma para los usuarios del sistema.

- **Workflow-Validación:** el proceso de validaciones entre los diferentes perfiles debe realizarse de manera secuencial y con las comprobaciones pertinentes antes de pasar a formar una nueva versión de los datos. De esta manera, cuando se realice la validación completa de cada versión debe comprobarse que el número de departamentos/proyectos según casuística debe corresponderse con el número total de elementos registrados, y así se podrá conformar la integridad de los datos asociados al proceso, en el momento que el número de validaciones parciales sea el mismo que el dato general de elementos, se dará por validada la versión. Las validaciones parciales dispondrán validación por actores requeridos y des validación por actores de permisos privilegiados, permitiendo la reevaluación de las estimaciones en caso de que así sea determinado. Tras la validación de los datos, será necesaria la generación de nueva versión que permita a siguiente perfil continuar con el flujo de trabajo para la consolidación de las estimaciones.

2.2 Modelo de casos de uso





Versión detallada en Miro[27]:
https://miro.com/welcomeonboard/SHA2Y1ZDZUJzdzlIdjF5aHBrMExHYkQ1YUFDdTdZUnhsSVlqTmtQbTV0bzd0dDRkQjlSMldwVDVaN2NmTTJNUXwzNDU4NzY0NTM1Njg4ODQzNzYxfDI=?share_link_id=773923718530

2.3 Actores

La identificación de los actores existentes en base a los perfiles que tendrán usabilidad en la aplicación, siendo éstos y el detalle de estos:

- Perfil Administrador: Actor encargado de labores de gestión y mantenimiento de los datos introducidos en la herramienta, con acceso privilegiado a los diferentes accesos y funcionalidades expuestas por la aplicación e identificadas en las historias de usuario.

- Perfil Planificador: Actor principal y orquestador de las actividades definidas por la herramienta, que se pone de acuerdo con varios perfiles para la introducción de datos en función de la versión validada, y de esta manera logar una consolidación de los datos para todo el ejercicio en curso. Adicionalmente, compartirá las funcionalidades de gestión de las tablas maestras, usuarios y accesos. Es el encargado de la realización de la versión cero del producto, para establecimiento de las estimaciones en horas por departamento y proyecto, en base a conocimiento ejercicios anteriores o expectativas de carga de trabajo estimado.

- Perfil responsable de Departamento: Actor encargado de la validación parcial de las estimaciones relativas a su Departamento, pudiendo rectificar o confirmar las estimaciones iniciales realizadas por el perfil Planificador. La validación será considerada como firma digital de las estimaciones plasmadas en la herramienta. Auditoria de los comentarios y valores indicados para dicha estimación. Este perfil puede ser encargado de uno o varios departamentos, tan solo tendrá constancia de los datos locales a los que tiene acceso. Relativo al versionado de la primera de las versiones disponibles.

- Perfil responsable de Proyecto: Actor encargado de la confirmación y validación de las estimaciones aseguradas por el perfil responsable Departamental, pero en este caso a nivel de Proyectos o Áreas de producto elaboradas por la organización. Es el encargado de la validación

a nivel de Producto, diferentes usuarios de la validación de la versión 2 de las estimaciones del ejercicio.

2.4 Historias de usuario

HU-ADMIN-1.1: Como ADMIN quiero dar de alta Departamentos por Sede transversales al ejercicio.
HU-ADMIN-1.2: Como ADMIN quiero dar de alta Proyectos transversales al ejercicio.
HU-ADMIN-1.3: Como ADMIN quiero acceder como otro usuario al sistema.
HU-ADMIN-1.4: Como ADMIN quiero modificar cualquier dato en cualquier versión, no validada.
HU-ADMIN-1.5: Como ADMIN quiero validar completa o parcialmente cualquier versión.
HU-ADMIN-1.6: Como ADMIN quiero visitar cualquier versión ya validada.
HU-ADMIN-1.7: Como ADMIN quiero dar de alta información financiera de un proyecto por sede.
HU-ADMIN-1.8: Como ADMIN quiero acceder a la administración de datos de la herramienta. CRUD usuarios, ejercicios, sedes, departamentos, proyectos, áreas…

HU-PLANW-1.1: Como PLANNER-WRITE quiero editar horas en la versión 0 y 3 del ejercicio.
HU-PLANW-1.2: Como PLANNER-WRITE quiero añadir comentarios a las estimaciones realizadas.
HU-PLANW-1.3: Como PLANNER-WRITE quiero visualizar los datos de anteriores versiones.
HU-PLANW-1.4: Como PLANNER-WRITE quiero descargar informes relativos a las estimaciones de horas.
HU-PLANW-1.5: Como PLANNER-WRITE quiero adjuntar ficheros a la estimación versión 0 a nivel de proyecto.

HU-PLANV-1.1: Como PLANNER-VALID quiero editar horas en la versión 0 y 3 del ejercicio.
HU-PLANV-1.2: Como PLANNER-VALID quiero añadir comentarios a las estimaciones realizadas.
HU-PLANV-1.3: Como PLANNER-VALID quiero visualizar los datos de anteriores versiones.
HU-PLANV-1.4: Como PLANNER-VALID quiero descargar todos los informes existentes.
HU-PLANV-1.5: Como PLANNER-VALID quiero adjuntar ficheros a la estimación versión 0.
HU-PLANV-1.6: Como PLANNER-VALID quiero validar la versión 0 al completo.
HU-PLANV-1.7: Como PLANNER-VALID quiero validar la versión 1 parcial o completamente.
HU-PLANV-1.8: Como PLANNER-VALID quiero validar la versión 2 parcial o completamente.
HU-PLANV-1.9: Como PLANNER-VALID quiero crear nuevas versiones de las estimaciones.

HU-PLANV-1.10: Como PLANNER-VALID quiero comprobar el estado de las validaciones en la versión 1 y 2.
HU-PLANV-1.11: Como PLANNER-VALID quiero acceder a la administración de datos de la herramienta. CRUD usuarios, departamentos, proyectos

HU-DPTO-1.1: Como JEFE-DPTO quiero editar horas en la versión 1.
HU-DPTO-1.2: Como JEFE-DPTO quiero añadir comentarios a las estimaciones realizadas.
HU-DPTO-1.3: Como JEFE-DPTO quiero visualizar los datos de anteriores versiones
HU-DPTO-1.4: Como JEFE-DPTO quiero descargar informes relativos a las estimaciones de horas.
HU-DPTO-1.5: Como JEFE-DPTO quiero validar la versión 1 de manera parcial (DEPARTAMENTOS)

HU-PROY-1.1: Como JEFE-PROY quiero editar horas en la versión 2.
HU-PROY-1.2: Como JEFE-PROY quiero añadir comentarios a las estimaciones realizadas.
HU-PROY-1.3: Como JEFE-PROY quiero visualizar los datos de anteriores versiones.
HU-PROY-1.4: Como JEFE-PROY quiero descargar informes relativos a las estimaciones de horas.
HU-PROY-1.5: Como JEFE-PROY quiero validar la versión 2 de manera parcial (PROYS).

HU-GRAL-1.1: Como USUARIO puedo logarme en la aplicación
HU-GRAL-1.2: Como USUARIO puedo iniciar sesión con diferentes perfiles/ejercicios
HU-GRAL-1.3: Como USUARIO puedo filtrar datos según mis permisos
HU-GRAL-1.4: Como USUARIO puedo navegar por Dpto/Proy hasta las horas.
HU-GRAL-1.5: Como NO USUARIO puedo solicitar acceso a la aplicación.

Versión detallada en Miro[28]
(https://miro.com/welcomeonboard/SHA2Y1ZDZUJzdzlldjF5aHBrMExHYkQ1YUFDdTdZUnhsSVlqTmtQbTV0bzd0dDRkQjlSMldwVDVaN2NmTTJNUXwzNDU4NzY0NTM1Njg4ODQzNzYxfDI=?share_link_id=773923718530)

2.5 Diseño del proyecto

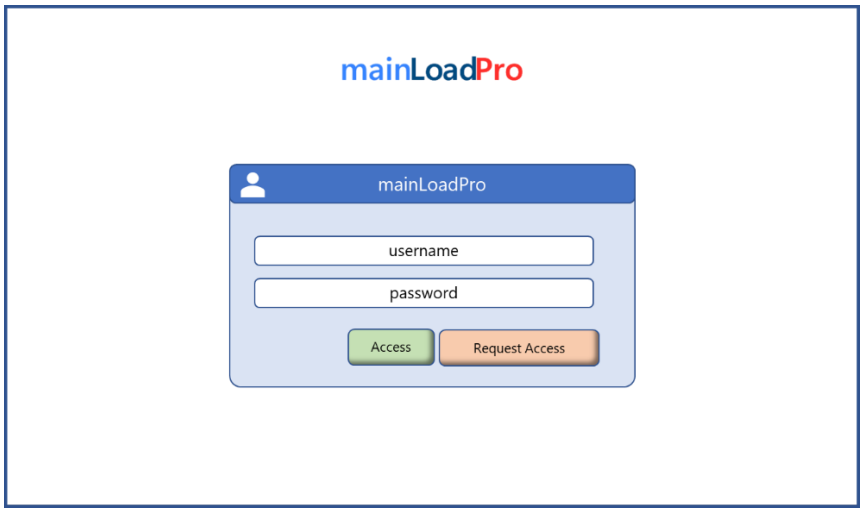
2.5.1 Prototipado de pantallas

A continuación, se muestra detalle de las principales pantallas que conformarán la aplicación, las cuales serán:

- Acceso (Login)
- Parrilla (vista principal)
- Alta/Edición Proyectos & Proyectos Finanza
- CRUD datos maestros (ejercicios, departamentos, secciones, proyectos, áreas de proyecto, líneas de negocio, usuarios)

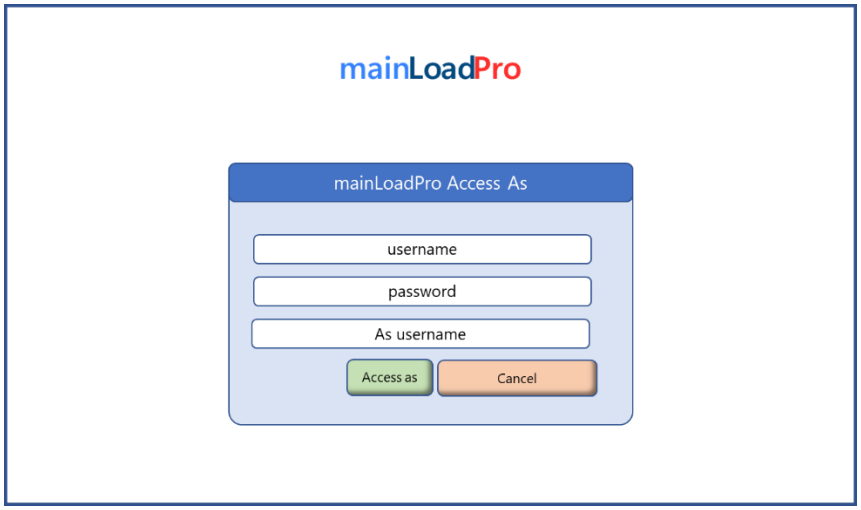
Vistas e historias de usuario relacionadas:

Acceso Login: Vista que muestra la vía de acceso a la aplicación, mediante un nombre de usuario (username) y clave de acceso (password), mediante el clicado en icono usuario accedemos al login con impersonalización de usuario.



HU-GRAL-1.1
HU-GRAL-1.5

Ilustración 2. Prototipo de pantalla de login.



HU-ADMIN-1.3

Ilustración 3. Prototipo de pantalla de login impersonalización de usuario

Vista Parrilla (Vista principal): Vista compuesta de múltiples vistas que mediante la carga Ajax muestra para un usuario y perfil indicado las horas correspondientes a un ejercicio (*cycle*), versión, pudiendo navegar, mediante desplegado de componentes cargados mediante nuevas vistas, las horas de estimación relativas a un proyecto y departamento para un determinado año. Al llegar al más bajo nivel de granularidad, proyecto y departamento, al clicar sobre las horas, nos muestra información de auditoría, entre versiones y autores, de las horas representadas

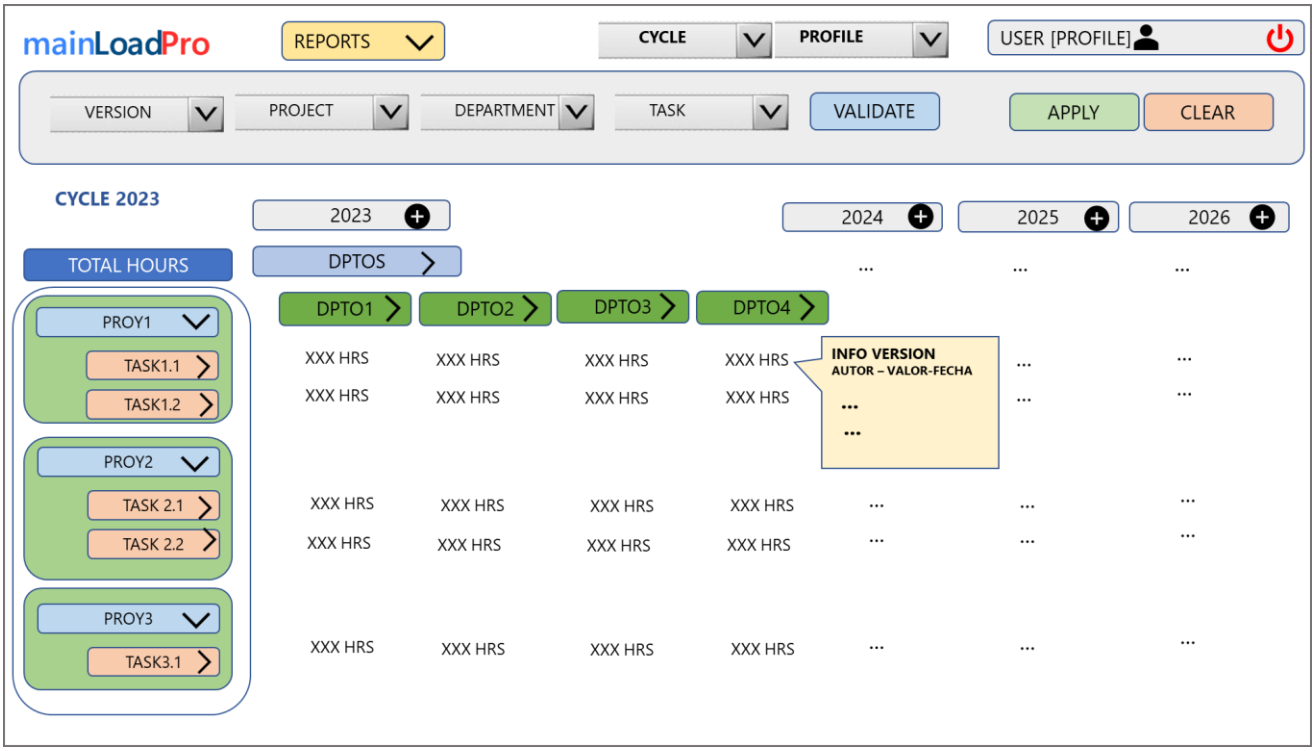
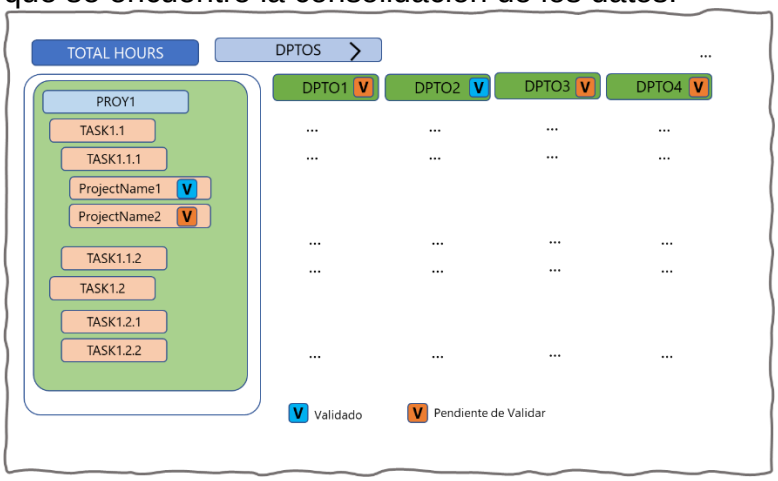


Ilustración 4. Prototipo de pantalla principal "parrilla" de datos

HU-ADMIN-1.4	HU-PLANW-1.1	HU-PLANV-1.1	HU-DPTO-1.1	HU-PROY-1.1	HU-GRAL-1.2
HU-ADMIN-1.5	HU-PLANW-1.2	HU-PLANV-1.2	HU-DPTO-1.2	HU-PROY-1.2	HU-GRAL-1.3
HU-ADMIN-1.6	HU-PLANW-1.3	HU-PLANV-1.3	HU-DPTO-1.3	HU-PROY-1.3	HU-GRAL-1.4
	HU-PLANW-1.4	HU-PLANV-1.4	HU-DPTO-1.4	HU-PROY-1.4	
	HU-PLANW-1.5	HU-PLANV-1.6			
		HU-PLANV-1.8			
		HU-PLANV-1.9			
		HU-PLANV-1.10			

Vista seccionada de parrilla, en la que se muestra cómo se realizarían las validaciones a nivel de departamento o proyecto, en función de la versión en la que se encuentre la consolidación de los datos.



HU-ADMIN-1.5
HU-PLANV-1.6
HU-PLANV-1.7
HU-PLANV-1.8
HU-DPTO-1.5
HU-PROY-1.5

Ilustración 5. Prototipo de validaciones en pantalla principal "parrilla"

Alta/Edición Proyectos y Proyectos finanza: Subvista enlazada desde vista principal parrilla que permite la actualización de los proyectos (misma vista usada para alta de proyectos) permite la inclusión de la información financiera por sede (ciudad).

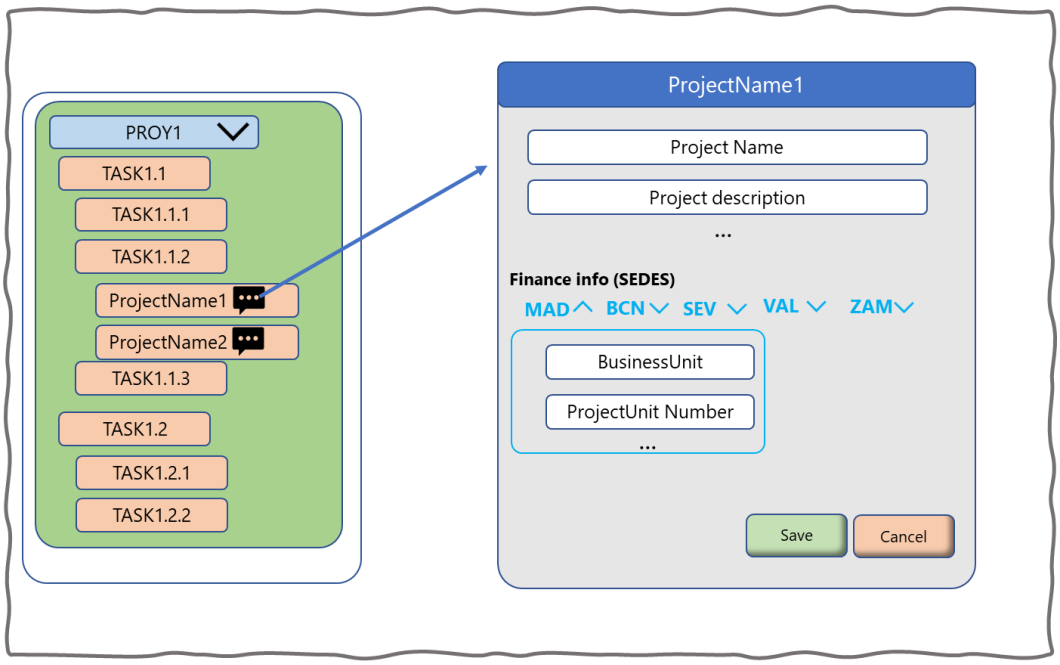


Ilustración 6. Prototipo de Información de proyecto definido

HU-ADMIN-1.7

Vistas **CRUD** relacionadas con los **mantenimientos** de los datos maestros de la aplicación, este mismo tipo de vista se aplicará a los mantenimientos de ejercicios, sedes, departamentos, secciones, proyectos, áreas de proyecto, líneas de negocio y usuarios.

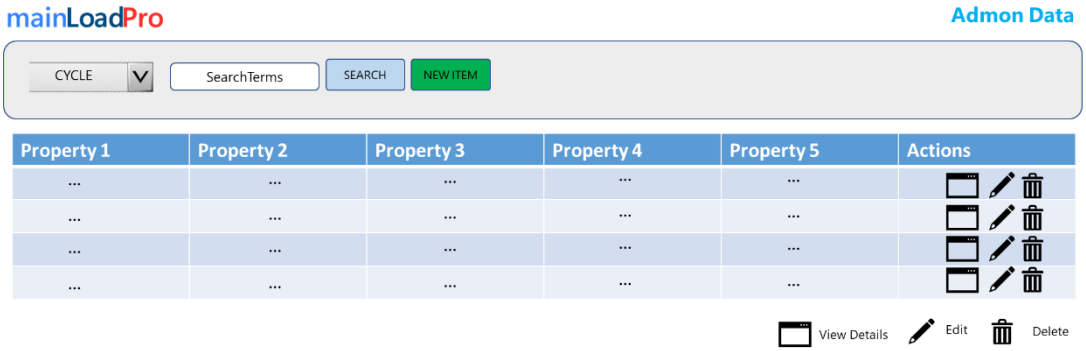


Ilustración 7. Prototipo de administración CRUD datos maestros aplicación

HU-ADMIN-1.1
HU-ADMIN-1.2
HU-ADMIN-1.8
HU-PLANV-1.11

2.6 Diseño relacional de base de datos

A continuación, se muestra detalle del diagrama de la base de datos relacional asociado a la aplicación mainLoadPro.

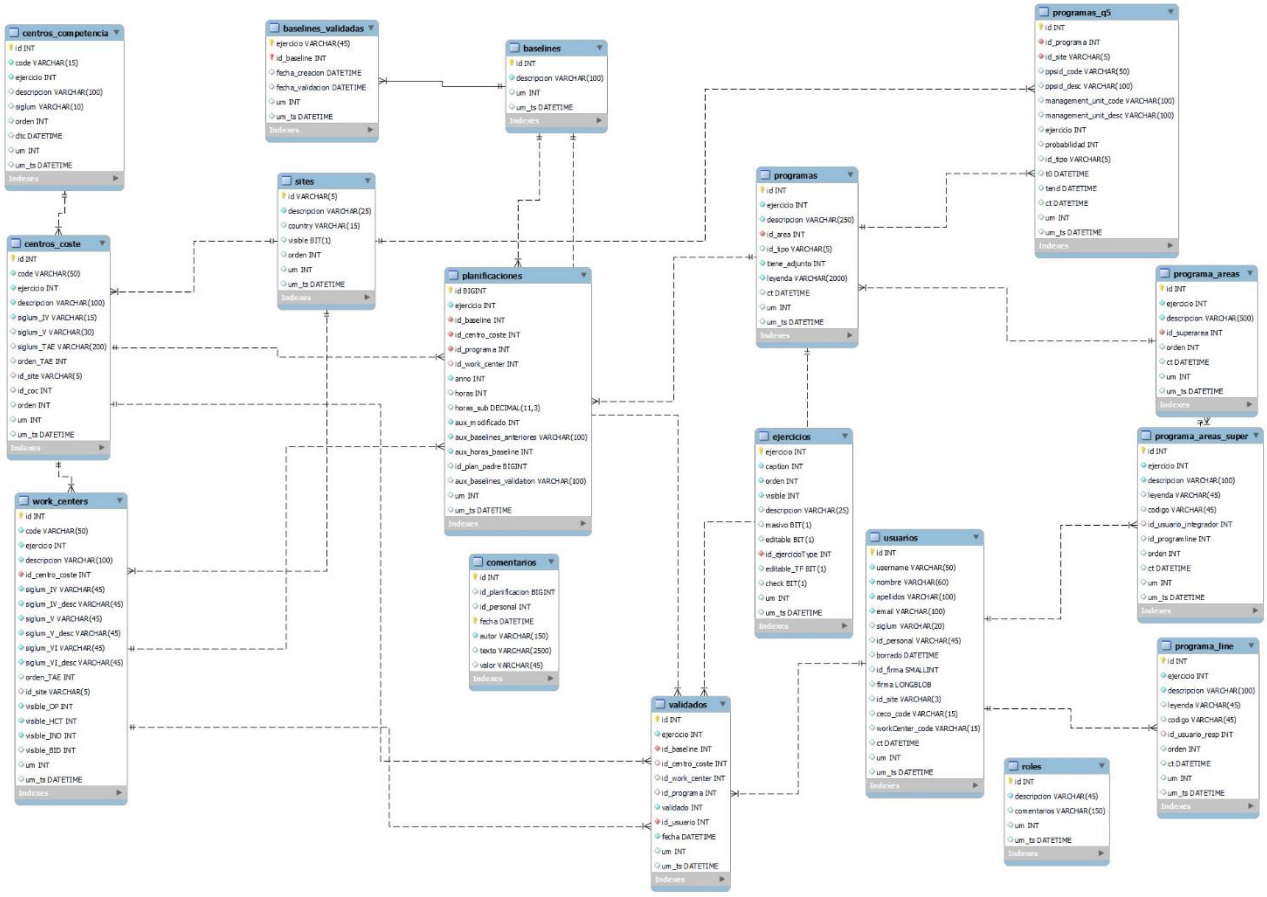


Ilustración 8. Vista del modelo de BBDD implementado para la aplicación

Base de datos, modelada con gestor MySQL8, con motor *innodb*, en la que se incluye además de los campos funcionales de la aplicación, propios campos de auditoría de datos para facilitar a los administradores de BBDD las gestiones y consultas relativas a la información, mediante los campos *um* (usuario modificador) y *um_ts* (usuario modificador time stampt).

Todas las tablas involucradas en el modelo de BDR además de contar con sus claves primarias y claves foráneas de relación con otras tablas, si las tuviese, tienen asociados propios índices de búsqueda que optimizan las consultas que se realizan, facilitando al motor de base de datos *innodb* el cacheo de las consultas más frecuentemente realizadas y que permita que la consulta de información se realice de manera fluida. Asimismo, cabe indicar que tablas potencialmente pesadas, como por ejemplo comentarios, ha sido creada como *tabla particionada*[29], de manera que la indexación de los datos para el conjunto de las búsquedas se realiza en un perímetro seccionado de tabla más acotado,

aligerando de manera sustancial los tiempos de acceso y consulta a la base de datos.

Las consultas más pesadas o en las que están involucradas más tablas, se realizan mediante el uso de *procedimientos almacenados*[30], y aunque en carácter general el uso de *EntityFramework* es mayoritario, la búsqueda de no penalización de tiempos en las consultas ha llevado a la realización de este modelo híbrido de trabajo.

Adicionalmente, incluida base de datos para la gestión del log de la aplicación, usando para ello librería de terceros bajo licencia GNU no comercialización, *Log4Net* [31], almacenando en base de datos las trazas, errores y excepciones tratadas en la aplicación. De esta manera, se permite la explotación de las trazas de error por otros futuros usuarios de gestión de la información o control de errores.

2.7 Diagrama de clases principales

Debido al fuerte componente de tipado de datos que posee el proyecto basado en *EntityFramework*, las principales clases se componen de tipados directos de datos autogenerados mediante método "*DataBase First*[32]" y obtención del modelo de datos mediante conexión a BBDD, con la consiguiente autogeneración de clases de control de entidades.
No obstante, a continuación, se muestra representación en diagrama de las principales clases código existentes en mainLoadPro.
mainLoadPro.mvc: Clases controladoras asociadas a las entidades de BBDD e interacción con usuario que toman herencia de *BaseController*, la cual efectúa mecanismos de control sobre las peticiones realizadas, gestionando autenticación y gestión centralizada de los errores de navegación.

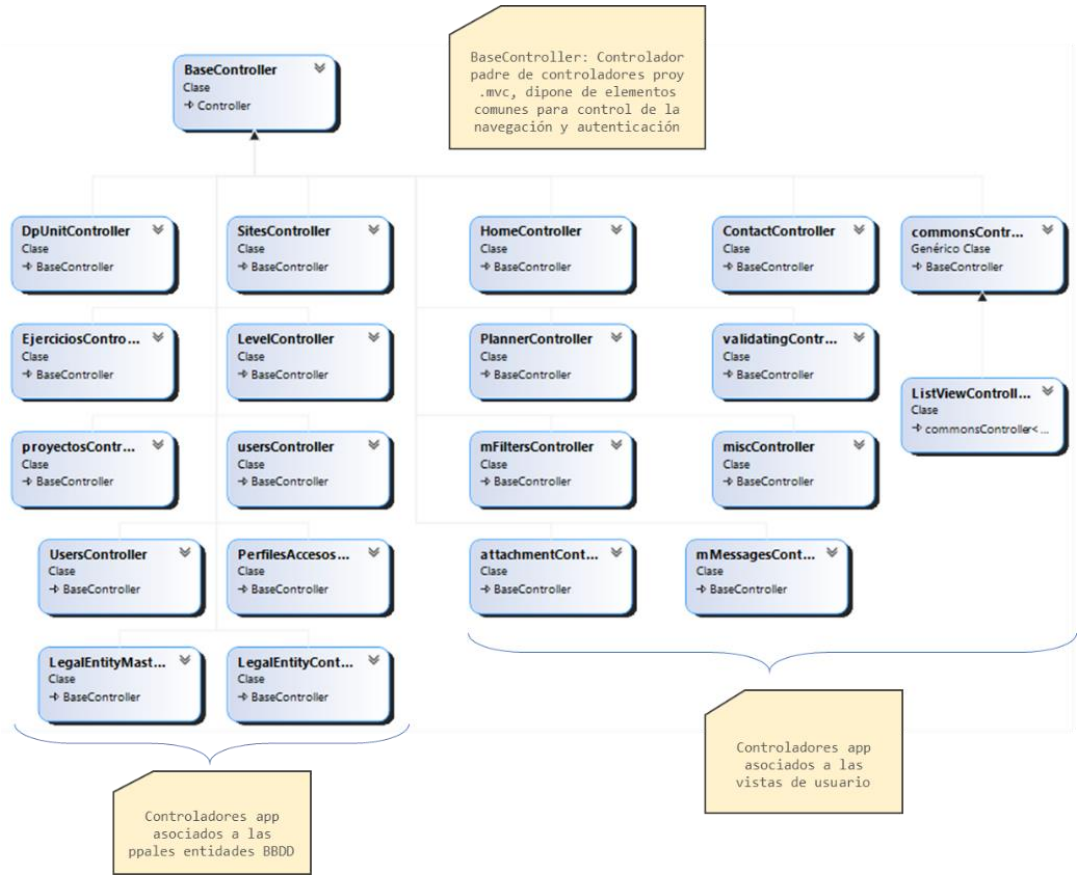


Ilustración 9. Vista del diagrama de principales clases usadas en el desarrollo de la aplicación.

mainLoadPro.mvc.Models.admon: Clases de modelado de entidades para los mantenimientos y gestiones administrativas, como base heredera principal, toma la clase *BaseEntityYear*, como patrón de guía común parar todas entidades regidas por un ejercicio. Adicionalmente, se toma, en clases dependientes de listados paginados, la sub-herencia de *BaseEntityPagination*.

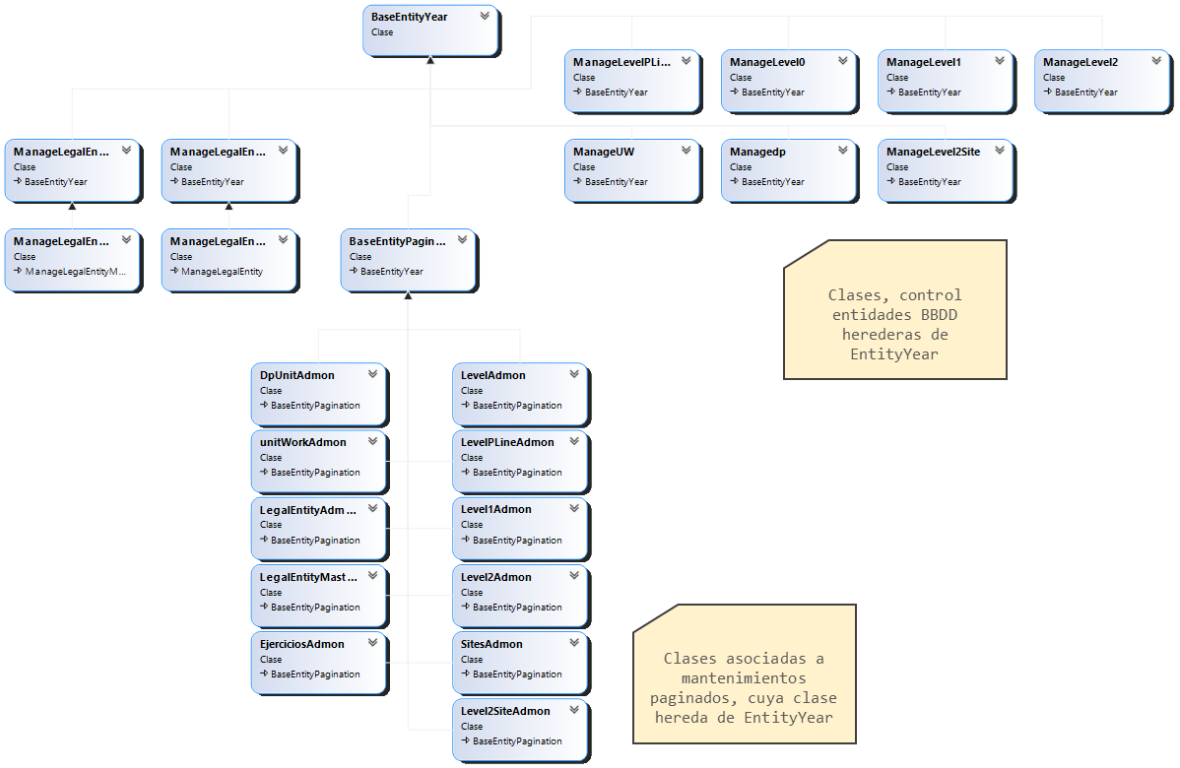


Ilustración 10. Vista del diagrama de principales clases basadas en modelo EntityYear, para módulo de administración

mainLoadPro.mvc.Models.planner: Clases de modelado de entidades para visualización de los listados de “parrilla” principal de datos, toman como herencia principal la clase modelo caracterizadora del departamento.

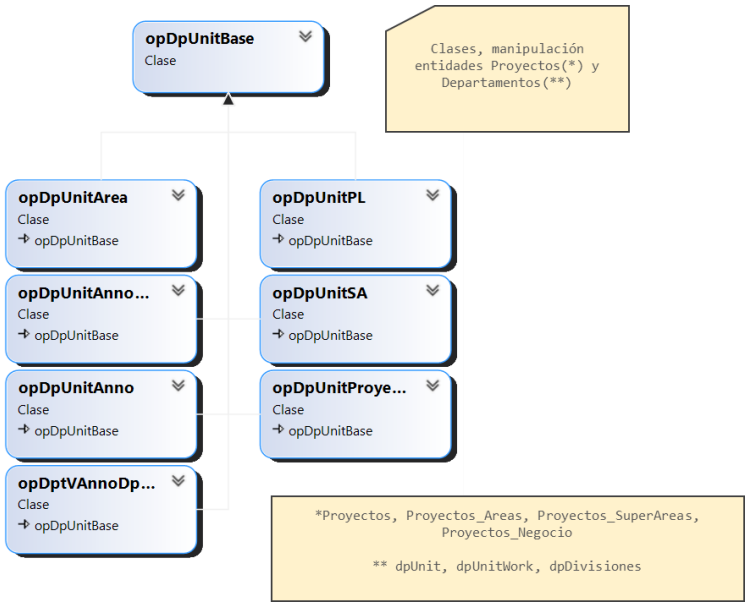


Ilustración 11. Vista de las clases del modelado para las vistas de página principal "parrilla".

2.8 Diagrama de Arquitectura

La arquitectura planteada para la aplicación *mainLoadPro* está basada en una arquitectura bajo servicios IIS, conformante de un módulo que puede replicarse en varias y simultáneos servicios web que apunten a diferentes bases de datos, de esta manera podemos realizar un escalado horizontal del aplicativo según la demanda de despliegue para diferentes entidades de una empresa, o diferentes empresas. Basándose, en que es la aplicación, puede estar en constante crecimiento, ofreciendo desde un mismo motor ampliable, funcionalidades a diferentes negocios, adaptando para ello los datos residentes en la base de datos externalizada.

Para ello, presento una arquitectura en *n-capas*, en la que podemos diferenciar a grandes pinceladas un motor web de vistas y modelos, *mainLoadPro.mvc*, que se apoya, por un lado, para toda la capa de negocio con la librería *mainLoadPro.bll*, y ésta accediendo a los servicios datos mediante *mainLoadPro.data* con correspondientes *ORM* dedicados de *EntityFramework* a un servicio externalizado y *dockerizado* de base de datos en *MySQL*.

Adicionalmente se incluye servicio API, *mainLoadPro.api,* para consultas a BBDD, modelo de exponer datos ofrecidos por el servicio a otras entidades.
Tal y como ya se ha indicado, además de una escalabilidad horizontal para el despliegue de nuevos entornos o entidades del aplicativo, en caso de producirse por una de las entidades que necesiten de alta disponibilidad, pueden balancearse la aplicación dentro de un enrutador de servidor apache, análogamente se puede replicar balanceo[33] de datos entre entidades de datos dockerizadas *MySQL*.

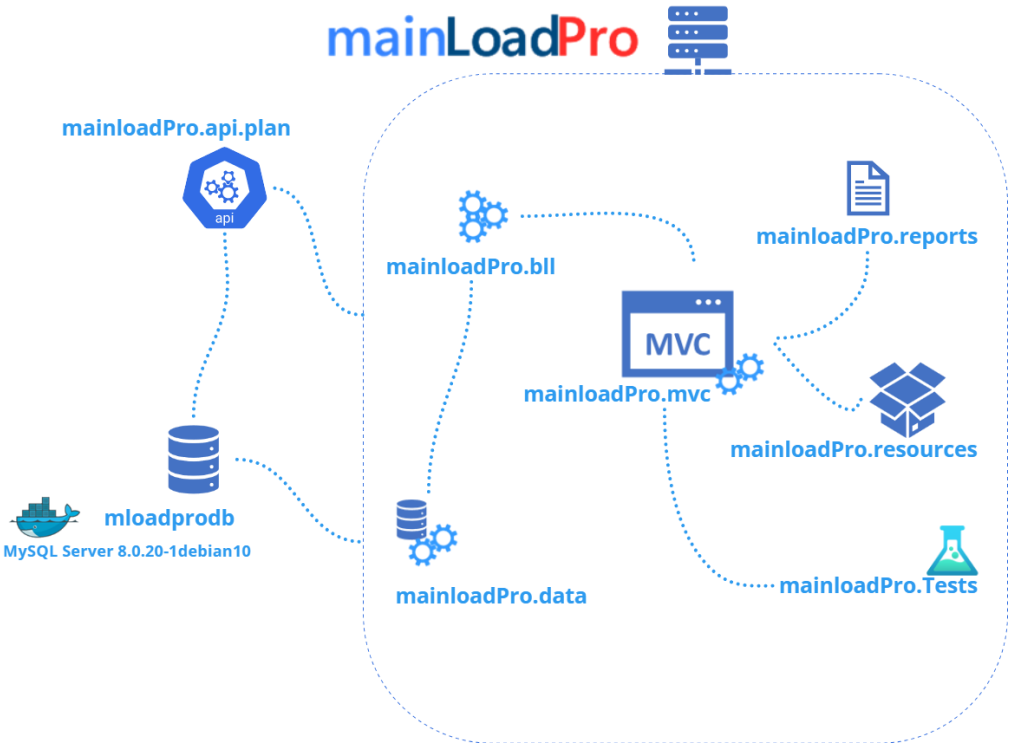


Ilustración 12. Detalle del diagrama de arquitectura mainLoadPro

3. Preparación de entorno de desarrollo

3.1 BBDD

Docker (MySQL Server)

La preparación de un entorno de base de datos estable para la implementación de correspondiente repositorio y conexión de pruebas al mismo ha sido realizada bajo un contenedor *debian*[34] que incluye el motor de base de datos *MySQL* Server en su versión *8.0.20-1*, la rápida instalación y configuración del entorno ha sido clave para su elección, instalación realizada mediante el siguiente comando Docker.

```
docker run -d --name mysql -e MYSQL_ROOT_PASSWORD=root -p 3306:3306 mysql:8.0
```

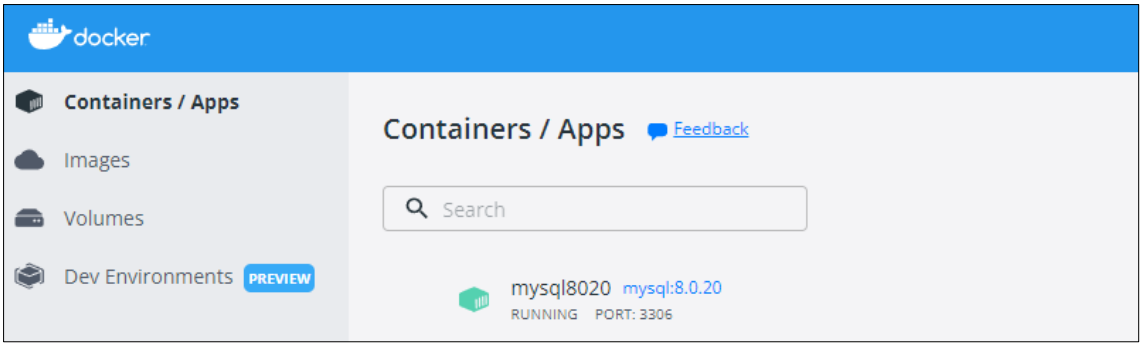


Ilustración 13. Detalle de contenedor con repositorio de datos en funcionamiento.

MySQL WorkBench

Estando ya en funcionamiento el servidor de base de datos que utilizaremos para los desarrollos, conectamos el software específico para la gestión y control de las BBDD, en nuestro caso, seleccionado *MySQL WorkBench*[35], en el cual realizando conexionado al *dockerizado* de MySQL, inicialmente con usuario administrador, podemos acceder a los servicios ofrecidos por la BBDD, estableciendo inicialmente usuario que se conectará desde nuestra aplicación web al servicio de datos.

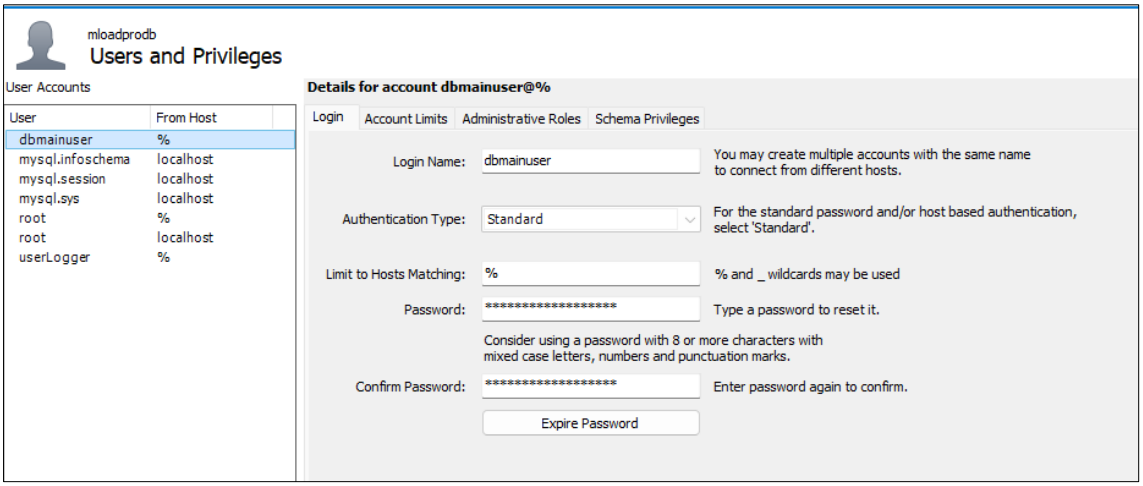


Ilustración 14. Detalle de vista de usuarios creados para las diferentes conexiones del aplicativo.

Estructura Procedimientos / Tablas MySQL

Los procedimientos y funciones creados para las operaciones sobre BBDD están estructurados de manera que los que comienzan por guion son elementos dedicados para la administración del sistema, y por otro lado comenzando con prefijo de la operación que realizan sobre los datos encontramos el resto de los elementos, véase imagen adjunta de la estructuración de los elementos incluidos en la BBDD.

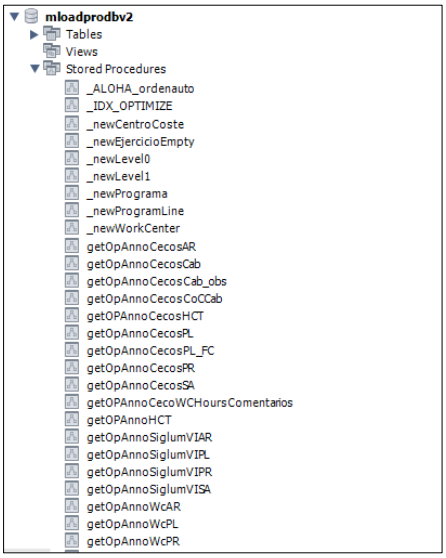


Ilustración 15. Detalle de la estructuración de elementos BBDD

Detalle procedimiento, definición, captura de errores y asentamiento

Los procedimientos están internamente estructurados con las siguientes clausulas

Declaración:

```
CREATE DEFINER=`dbmainuser`@`%` PROCEDURE `_newWorkCenter`
```

Declaración excepciones y rollback:

```
DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        ROLLBACK;
        SELECT @ERROR;
    END;
START TRANSACTION;
...
```

Posible control de errores y persistencia (commit) de los datos:

```
...
  SET @ERROR = -8;

END IF;
COMMIT;
SELECT 0;
END
```

Determinación para el uso de procedimiento almacenados en entorno EntityFramework

De manera adicional al proceso de obtención de datos que será realizado desde el *ORM* hacia nuestro servidor *MySQL*, el uso de procedimientos garantiza que se producirá un acceso más eficiente para grandes volúmenes de datos, y permite que se trabaje de manera asíncrona en la obtención continuada de datos, como el proyecto en implementación, adicionalmente podemos trazar errores propios de la BBDD de una manera más clara y concisa. Por tanto, la mayor de las ventajas del uso de procedimientos frente a un uso intensivo ORM es una cuestión de rendimiento de acceso a datos frente a una escalabilidad importante en volumen de información alimentada para y por la aplicación.

Adicionalmente, el uso de procedimientos almacenados proporciona a los administradores de la aplicación realizar consultas y posibles modificaciones sobre los datos de manera rápida, permitiendo la ejecución en lote de información. Generación de informes de primera mano, así como control y seguimiento de las trazas que puedan provocar cada operación de consulta a BBDD.

Configuración de la base de datos en entorno Azure Cloud

El propósito de esta propuesta de aplicación es el despliegue en un entorno de producción de manera que necesitaremos llevar nuestros datos o a un servidor o a un servicio en la nube de base de datos. Adicionalmente a las tareas implementadas sobre la BBDD en local, se ha preparado versión cloud en *Azure*, para la cual ha sido imprescindible la configuración personalizada de algunas parametrizaciones del servidor.

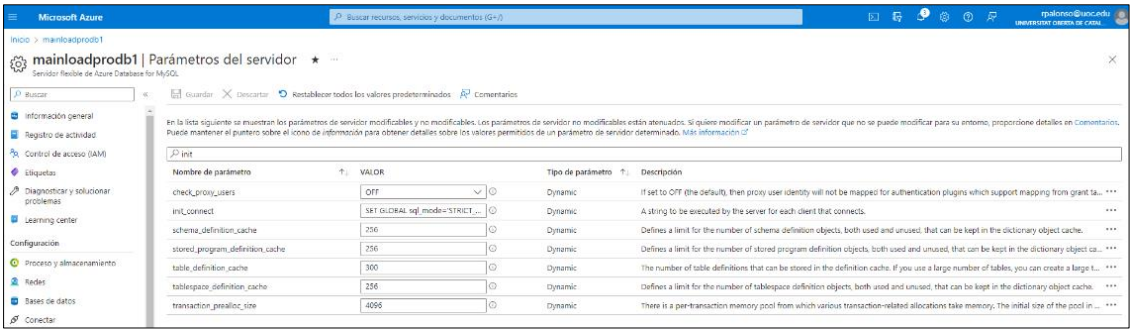


Ilustración 16. Detalle de parámetros del Servidor de servicio de BBDD creado en cloud para futuro despliegue en Producción

3.2 Microsoft Visual Studio

3.2.1 Versión VS Instalada

La versión elegida para la realización del proyecto web es *Microsoft Visual Studio Community 2019*[36], que permite de una manera sencilla la integración de múltiples proyectos de diferente naturaleza en una misma solución, siendo la versión más adecuada para el .Net Framework 4.7.2[37]

3.2.2 Versión Framework .Net

Esta versión del framework .Net elegida, es un paso previo al siguiente producto evolutivo de la familia .Net, el *.Net Framework Core*[38], debido a la alta curva de aprendizaje de este nuevo Framework, por lo que el 4.7.2 se considera una elección satisfactoria en cuanto a robustez y eficiencia, además de una completa integración del conexionado a BBDD *EntityFramework* bajo un entorno de *MySQL*.

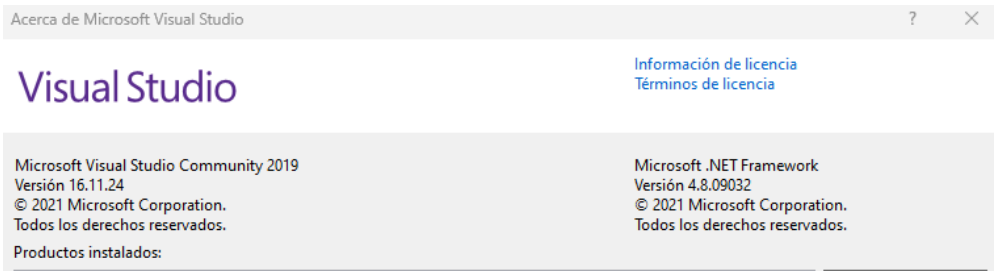


Ilustración 17. Detalle de versión IDE VS Community 2019 en uso para desarrollo e implementación.

3.2.3 Librerías nuGet – MySQL, EF, .Net, MVC

Las dependencias propias a la naturaleza de la solución en implementación corresponden con las siguientes características y sus respectivas versiones.

- Conector de BBDD[39]: Versión 8.2
- Plugin MySQL for VisualStudio[40]
- Paquetes NuGet incluidos en la implementación:
  - MySQL.Data | MySQL.Data.EntityFramework 8.0.18
  - BootStrap 3.3.7
  - JQuery 3.6.0
  - *Log4Net* 2.0.15
  - Versiones alineadas con paquete *Microsoft.Net.Http 2.2.29*[41]
  - *Swagger-Net*[42], en proyecto API

3.2.4 Estructura de la solución de proyectos

La solución en implementación consta de diferentes tipos de proyectos compilables por sí mismos y que referencian el resultado de su compilación hacia el compilado de binario de otros proyectos dependientes. A continuación, se muestra detalle del árbol de estructura de proyecto en implementación.

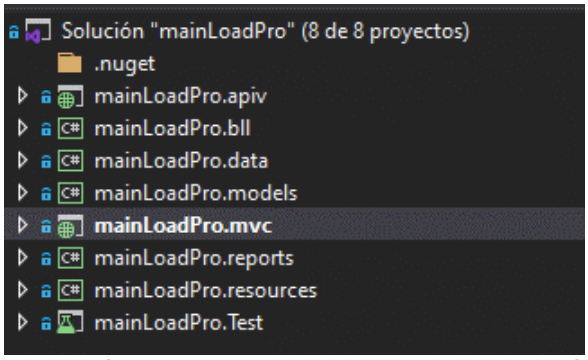


Ilustración 18. Detalle de estructura de la solución mainLoadPro

3.2.5 Proyectos definidos

El pilar central del aplicativo es sin duda el proyecto que conforma toda la unidad de negocio e interfaz para el usuario, *mainLoadPro.mvc*, el cual referencia en su justa medida para una separación de las capas de la aplicación con el resto de los proyectos de la siguiente manera.

mainLoadPro.api → Proporciona métodos webApi para el acceso desde librerías o terceros.

mainLoadPro.bll → proporciona acceso a los métodos de negocio y posterior acceso a datos del aplicativo.

mainLoadPro.data → Aunque no de una manera directa, ciertas vistas y entramados de EntityFramework reciben directamente información desde esta librería de acceso a datos.

mainLoadPro.models → correspondiente con los modelos que se usan en el aplicativo para poder transformar la información proveniente de BBDD a las vistas de cliente, adicionalmente permiten la obtención y persistencia de datos.

mainLoadPro.reports → Aunque fuera del alcance para esta entrega es previsiblemente el proyecto que permitirá la exportación de reportes e información almacenada en nuestra BBDD.

mainLoadPro.resources → Mediante este proyecto personalizamos textos y configuraciones para diferentes motivos de negocio, localización y personalizaciones de cultura bajo demanda. Permite la adaptación a varios idiomas de nuestro aplicativo con tan solo la detección de las opciones de cultura del navegador web.

3.2.6 Proyección Alcance solución implementada

Como parte inicial de un aplicativo en crecimiento esta implementación debe permitir un escalado arquitectónico de la solución que permita que cualquiera que sea el desarrollador que se ponga al frente de la labor pueda ampliar el perímetro y funcionalidades de la aplicación sin un alto esfuerzo para el entendimiento y sin interferencias y problemas de carácter estructural. Por ello se basa la solución en una implementación por capas diferenciadas que permita la posterior adición de más áreas, funcionalidades y proyectos.

Incluso con la inclusión de API de servicio en caso de consulta de datos por otras entidades fuera de ámbito, así pues, en la estructura puede visualizarse como modularmente se permite crecimiento de la herramienta, mediante áreas del MVC, aisladas y funcionales.

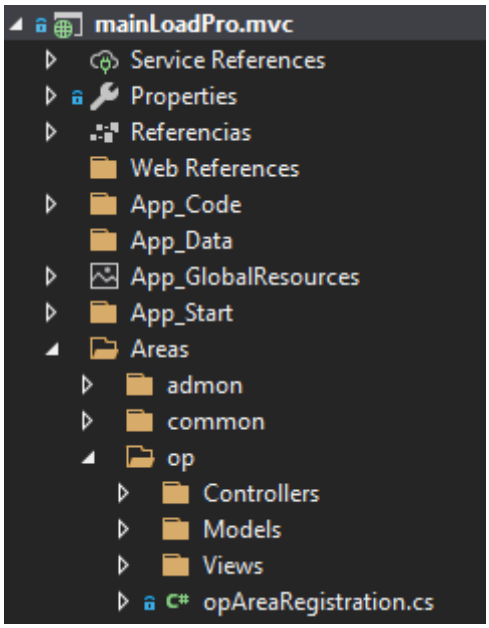


Ilustración 19. Detalle del uso de áreas MVC como referencia modular del aplicativo.

Cada área consta de su unidad funcional del modelo de implementación *MVC*, con Controladores, Modelos y Vistas asociadas a ese contexto de la aplicación, de esta manera la escalabilidad y crecimiento del aplicativo puede realizarse sin deuda técnica de conflicto entre componentes.

Para esta implementación, se ha propuesto la definición de pequeños componentes funcionales y modulares que pueden conformar diferentes vistas, de manera que sean fácilmente acoplables y reutilizados en otras funcionalidades similares presentes o futuras, en hilo a esta disposición encontraremos una representación basada en decenas de componentes que forman una pantalla global. Tal y como podemos ver en las siguientes imágenes extraídas de la estructura de proyecto del IDE de desarrollo y del uso de las vistas para conformación de vista principal.

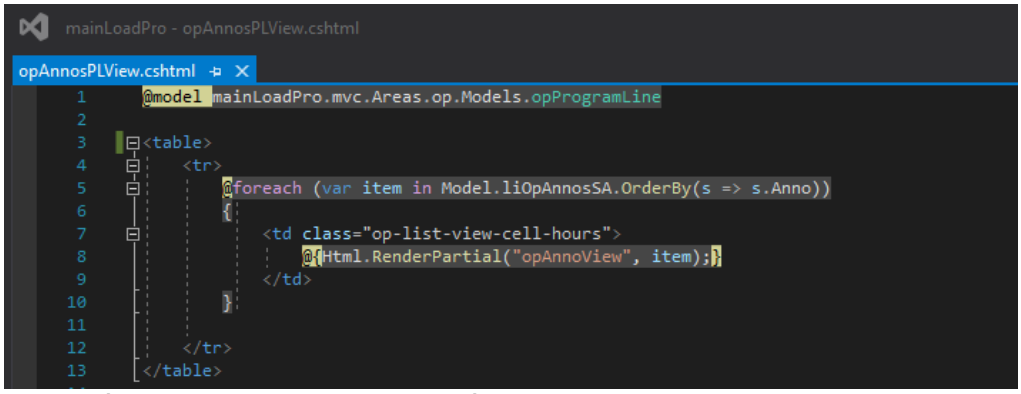


Ilustración 20. Detalle de implementación y uso de la vista de vistas

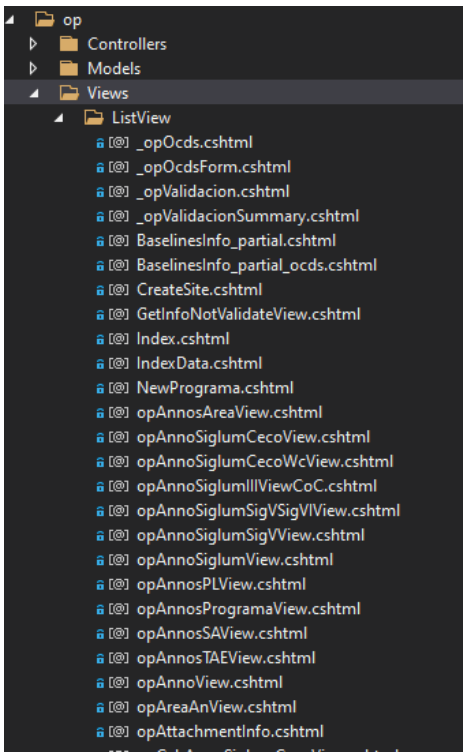


Ilustración 21. Detalle de la estructura con múltiples vistas para conformación de pantalla principal

3.2.7 Frameworks de Cliente (JS / Jquery / Bootstrap)

La selección de interfaz usuario seleccionada está basada en frameworks fácilmente implantables en soluciones web y extensamente popularizados, por lo que el soporte que puede aportar la comunidad en caso de nuevos desarrollos o localización de incidencias es de un alto nivel. Estos frameworks seleccionados son, por un lado para la parte de interacción con el usuario Bootstrap 4[43], con componentes modulares basados en marcas de estilos CSS, altamente personalizables y que ofrecen a la navegación un dinamismo particular y muy intuitivo, adicionalmente estos compontes son fácilmente adaptables a todos los navegadores y formatos de pantalla existentes, siendo de esta manera un “responsive framework“. Mientras que para la operatividad con cliente y comunicación de operaciones con servidor se ha implementado el uso de JQuery 3.6, que proporciona optimizados accesos mediante comunicación Ajax con operativas de servidor.

A continuación, se muestra detalle de la operatividad general de comunicación Ajax entre cliente y servidor y disposición de elementos mediante métodos JQuery:

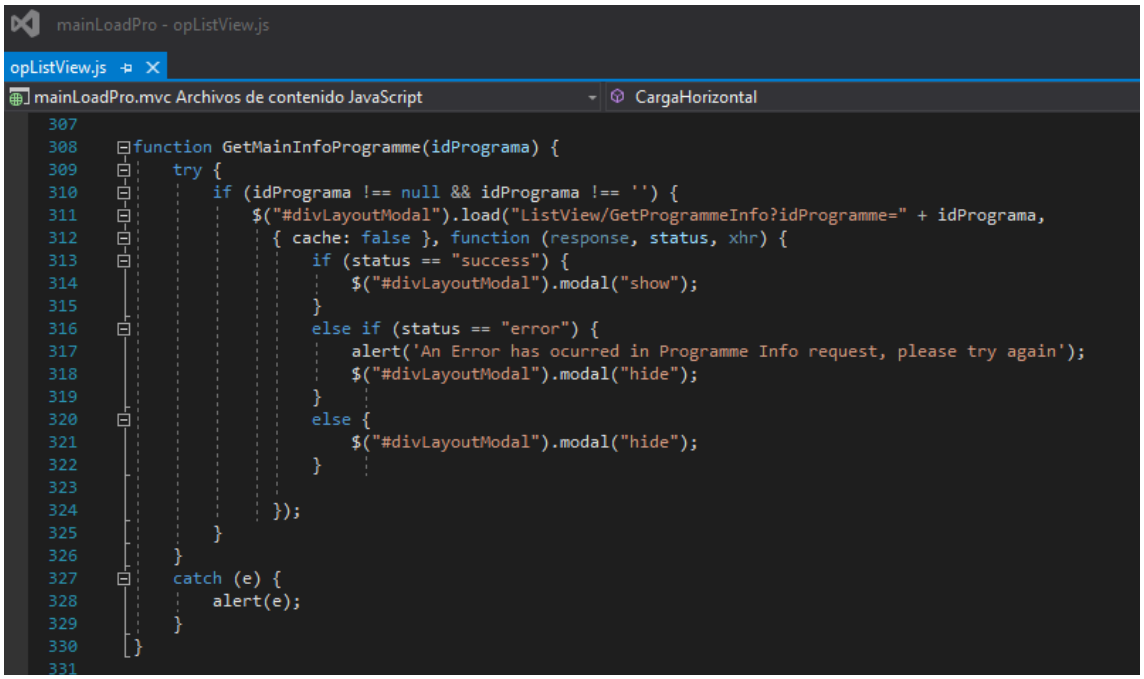


Ilustración 22. Detalle de uso en carga modal, llamada cliente Ajax y control de visibilidades mediante JQuery.

En la siguiente figura, encontramos detalle del funcionamiento y uso de vista cliente con denominación a clases Bootstrap:

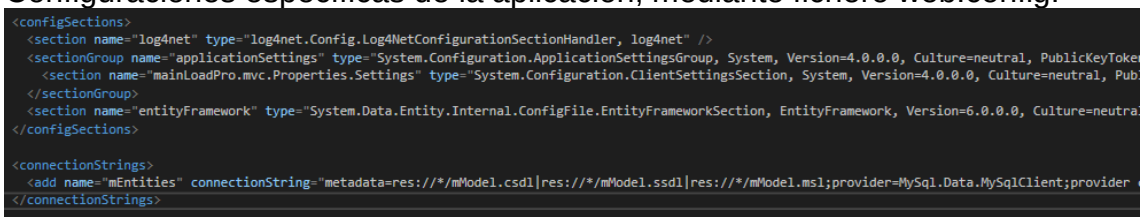
```
if (Model.liBaselineList.FirstOrDefault().Text.Contains("PENDING"))
{

    if (Model.op_add_programa)
    {
        <button id="btnAddProgram" type="button" class="btn btn-sm btn-success" onclick="AddProgram(0)">
            <span class="glyphicon glyphicon-plus icon-2x"></span> Add Programme</button>
    }
    if (Model.op_validate_all)
    {
        <button id="btnValidateAll" type="button" class="btn btn-sm btn-danger" onclick="ValidateAll(@(Int32.Parse(Model.liBaselineList.FirstOrDefault().Value)))">
            <span class="glyphicon glyphicon-ok icon-2x"></span> Validate ALL</button>
    }
    if (Model.op_info_not_validate)
    {
        <button id="btnInfoNotValidate" type="button" class="btn btn-sm btn-info" onclick="GetInfoNotValidate()">
            <span class="glyphicon glyphicon-info-sign"></span> Validation Info</button>
    }
    if (Model.permissionValidatedCecos)
    {
        <button id="btnValidateCecos" type="button" class="btn btn-sm btn-warning" onclick="ValidateCecos()">
           <span class="glyphicon glyphicon-ok icon-2x"></span> Validate Cost Centers</button>
    }
```

Ilustración 23. Detalle de uso clases de estilos en vistas cliente con Bootstrap.

3.2.8 Configuración aplicación

La configuración posee varios mecanismos de configuración, parte de ellos asociados en el propio fichero web.config, los generales a funcionamiento de la herramienta, y otros definidos en BBDD, generalmente para la definición de acciones por cada uno de los perfiles y configuración general por ejercicio en curso.

Configuraciones específicas de la aplicación, mediante fichero web.config:



```
<configSections>
  <section name="log4net" type="log4net.Config.Log4NetConfigurationSectionHandler, log4net" />
  <sectionGroup name="applicationSettings" type="System.Configuration.ApplicationSettingsGroup, System, Version=4.0.0.0, Culture=neutral, PublicKeyToke
    <section name="mainLoadPro.mvc.Properties.Settings" type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0, Culture=neutral, Pub
  </sectionGroup>
  <section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutra
</configSections>

<connectionStrings>
  <add name="mEntities" connectionString="metadata=res://*/mModel.csdl|res://*/mModel.ssdl|res://*/mModel.msl;provider=MySql.Data.MySqlClient;provider
</connectionStrings>
```

Ilustración 24. Detalle de las claves de configuración de aplicación definidas en web.config

```
<appSettings>
  <!-- CURRENT KEYS APP -->
  <add key="webpages:Version" value="3.0.0.0" />
  <add key="webpages:Enabled" value="false" />
  <add key="ClientValidationEnabled" value="true" />
  <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  <!-- APP PATHS -->
  <add key="APP_ABSOLUTE_PATH" value="http://localhost:52888/" />
  <add key="LOGIN_ABSOLUTE_PATH"
  value="http://localhost:52888/Index/LoginSelf" />
  <add key="LOGINOFF_ABSOLUTE_PATH"
  value="http://localhost:52888/Index/LoginOff" />
  <add key="MENU_ABSOLUTE_PATH" value="http://localhost:52888/Index/" />
  <add key="K_SECONDS_TO_REFRESH_SESSION" value="600001" />
  <!-- SYSTEM APP SETTINGS -->
  <add key="AppVirtualDir" value="" />
  <add key="webpages:Version" value="3.0.0.0" />
  <add key="webpages:Enabled" value="false" />
  <add key="ClientValidationEnabled" value="true" />
  <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  <!-- NOTIFICACIONES EMAILS -->
  <add key="MAIL_FROM" value="mainLoadPro@uoc.edu" />
  <add key="EMAIL_PLAN_SUPPORT" value="rpalonso@uoc.edu" />
  <add key="EMAIL_TECH_SUPPORT" value="rpalonso@uoc.edu" />
  <add key="PATH_MAIL_TEMPLATES" value="/Content/mailTemplates/" />
  <!-- ATTACHMENT KEYS  -->
  <add key="ATTACHMENTS_PATH" value="attachments/" />
  <add key="ATTACHMENTS_PATH_TEMP" value="attachments_temp/" />
  <add key="ATTACHMENTS_PATH_ERASED" value="attachments/erased/" />
  <!-- APP TRAZAS -->
  <add key="ACTIVA_TRAZA" value="false" />
  <add key="ACTIVE_LAG_REQUEST" value="true" />
</appSettings>
```

Ilustración 25. Detalle de las claves de aplicación definidas en web.config

Configuraciones de perfil de usuario especificadas en base de datos, aun siendo configuraciones que pueden ser consideradas como constantes, pueden sufrir

cambios en el ciclo de vida del producto, y que los valores que afectan a cada perfil de usuario cambien por condiciones del negocio o funcionales de la herramienta, siendo una manera fácil e inmediata de que se vean reflejados los cambios en un entorno de producción. A continuación, se incluye extracto de consulta realizado sobre las tablas relacionadas, *accesos* y *perfiles_accesos* que contienen la información relevante de los perfiles y las acciones que pueden hacer cada uno.

En la captura se muestran los campos importantes de la estructura en los que se destaca: id (clave del acceso), descripción (detalle del control de acceso), *id_perfil* (perfil de la aplicación afectado por el registro), modo (*baselines* del producto a la que afectan, en caso de ser clave binaria, puede tomar valores `*true* | *false*´, el motivo de que este campo se trate como texto es la multitud de valores que puede albergar así como el rápido tratamiento desde la codificación mediante clausulas “contiene” para localizar las versiones en las que el perfil puede realizar acción, limitación de datos o acceso a determinada área.

```
29 •  Select a.id,a.descripcion,pa.id_perfil,pa.modo
30    from accesos a left join perfiles_accesos pa on a.id = pa.id_acceso;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	descripcion	id_perfil	modo
op_add_programa	Add Program	PLAN-VALID	0123
op_borrar_programa	Remove Program	PLAN-VALID	0123
op_borrar_programa	Remove Program	ADMIN	0123
op_desvalidar	Unvalidate OP Programs/Cost Centers	ADMIN	true
op_desvalidar	Unvalidate OP Programs/Cost Centers	PLAN-VALID	true
op_editar	Edit OP Hours by Profile/Baseline	ADMIN	01234
op_editar	Edit OP Hours by Profile/Baseline	PLAN-VALID	0234
op_editar	Edit OP Hours by Profile/Baseline	PLAN-WRITE	02
op_editar	Edit OP Hours by Profile/Baseline	CHIEF-VALID	2
op_editar	Edit OP Hours by Profile/Baseline	CHIEF-WRITE	2
op_editar	Edit OP Hours by Profile/Baseline	COSTC-VALID	1
op_editar	Edit OP Hours by Profile/Baseline	COSTC-WR...	1
op_editar	Edit OP Hours by Profile/Baseline	INTEGRATOR	1
op_info_not_validate	Info Button Not Validate	PLAN-VALID	012345
op_info_not_validate	Info Button Not Validate	ADMIN	012345

Ilustración 26. Detalle de las configuraciones de perfil en BBDD para la aplicación, pueden incluirse claves globales de uso.

4. Funcionalidades asociadas a BackEnd

4.1 Selección modelos y creación EDMX – EF – MySQL

El modelado de los datos mediante capa ofrecida por EntityFramework, además de permitir una mayor celeridad en el desarrollo de los aplicativos, el control integrado de errores tipográficos que pueden demorar fases de pruebas, así como una interacción integrada con la base de datos. Para este caso, el proceso de modelado seguido está basado en ´Database First´. De esta manera se logran seguir las recomendaciones y requerimientos para el funcionamientos y explotación de los datos, y en base a estos requerimientos se construye la

aplicación. Para este caso, la orientación a los datos es fundamental para la evolución y desarrollo de este, por lo que una correcta implementación de la estructura de los datos asegura el correcto funcionamiento en el tiempo del aplicativo. Al tratarse de una aplicación que requiere el tipado funcional de los datos, la interacción con modelos ORM es muy clara y permite el control de las funcionalidades requeridas entorno al modelo de datos implementado. El apoyo de procedimientos almacenados a los procesos de gestión de datos de EntityFramework, permite una agilidad necesaria para la manipulación de gran cantidad de datos para los procesos en marcha.

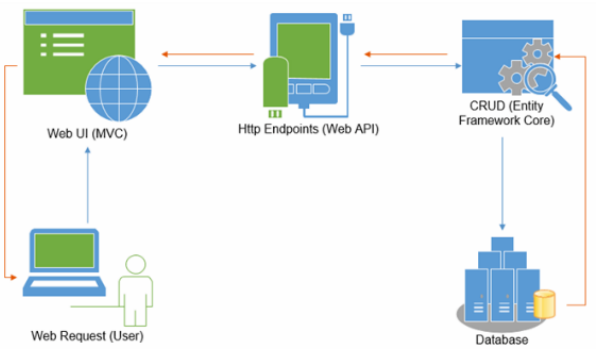


Ilustración 27. Diagrama funcional de EntityFramework dataFirst por https://www.codeproject.com/Articles/1218427/Getting-Started-with-Entity-Framework-Core-Buildin

Para el contexto general de BBDD debemos tener en cuenta que por defecto EF realiza una obtención “perezosa” de los datos, es decir, si no le indicamos lo contrario cargará en memoria todo el modelo que se encuentre relacionado con la entidad que queremos recuperar. Se trata de un mecanismo muy cómodo para trabajar desde vistas, que nos permiten un acceso total a los datos relacionados del dato obtenido/visualizado, pero que en casos de una alta carga de entidades o datos puede generarnos problemas piramidales en nuestra implementación. Para deshabilitarlo, debemos establecer en la configuración de la conexión el parámetro Lazy a falso.

4.1.1 Mecanismo llamadas a procedimientos y captura en modelo EntityFramework

Como anteriormente se ha comentado la recuperación de datos a volúmenes extensos es más eficiente realizando llamadas a procedimientos, siendo el motor de BBDD encargado de gestionar sus recursos para ofrecer un mayor rendimiento, ayudado por optimas configuraciones de nuestro servidor DB. Sin embargo, híbridamente a esta implementación, podemos derivar el resultado tabulado del procedimiento hacia el modelo de datos mapeado por EntityFramework, de esta manera aprovechamos la velocidad en el mapeo ofrecido por EF, y la velocidad del motor de BBDD. Para ello se realizan transformaciones en el retorno de datos del procedimiento para que coincidan en forma con el tipado de datos (modelo) que usaremos en vistas o capas de negocio. A continuación, se muestra un ejemplo de uno de las llamadas y conversiones al modelo tipado que conforma la aplicación.

```
//OBTENER PLANIFICACION PROGRAMA
//GET list opCecoPrograma: obtengo todas las planificaciones para mi Programa
queryParams = new MySqlParameter[] {
      new MySqlParameter("P_ejercicio", iEjercicio),
      new MySqlParameter("P_idBaseline", iBaseline),
      new MySqlParameter("P_idPrograma", programa.idPrograma),
      new MySqlParameter("P_tipoHoras",(int) oFilters.sTipoHoras),
      new MySqlParameter("P_tipoView",(int) oFilters.sVistaSelectedEurWL)
      };

List<Models.opCecosPrograma> liPlanPrograma = ctx.Database.SqlQuery<Models.opCecosPrograma>
    ("CALL getOpCecosPrograma(@P_ejercicio, @P_idBaseline, @P_idPrograma, @P_tipoHoras,@P_tipoView)", queryParams).ToList();

//FILTRAR PERMISOS
//Dejamos sólo las líneas de planificaciones de los cecos permitidos
liPlanPrograma = (from sa in liPlanPrograma
                  where sa is Models.opCecosPrograma &&
                  idsPermCecos.Contains(sa.idCeco)
                  select sa).ToList<Models.opCecosPrograma>();
```

Ilustración 28. Detalle de llamada y recuperación en modelo tipado por EntityFramework.

4.2 Implementación de la aplicación

En este apartado se documentan actividades destacables del desarrollo, ámbito y fases de este, con ilustraciones de las pantallas de aplicación en funcionamiento.

4.2.1 Multiplicidad y escalabilidad de conexiones a entidades de datos

Como parte de conformación de una aplicación escalable y aplicable a entornos diferenciados, se han establecido un gestor de conexiones que permitirá, mediante la adición de un nuevo entorno enumerado más las correspondientes claves en *web.config* de una nueva entidad de conexionado a BBDD, También se podría realizar conexión para implementar conexiones diferenciadas de lectura y escritura, no implementado en este desarrollo, por lo tanto queda fuera de alcance del proyecto, aunque se mantiene la posibilidad de realizarlo, y evaluar peticiones de conexión según sean obtención o persistencia de datos.

```
namespace mainLoadPro.bll.utils
{
    /// <summary>
    /// Contexto común para las funciones. Contiene operaciones comunes.
    /// </summary>
    5 referencias
    public abstract class FunctionContext : DbContext
    {
        1 referencia
        internal protected FunctionContext(string name) : base(name) { }

        /// <summary>
        /// Nombres de las cadenas de conexión para los diferentes entornos mainLoadPro (deben estar definidas en el web.config)
        /// </summary>
        3 referencias
        public enum eFunctionsConnections : short
        {
            mwlEntities = 1,
            mwlEntities_read = 2,
            mwlEntities_write = 3
        }

        0 referencias
        public static string GetNameConnection(int idFuncion) => GetNameConnection((eFunctionsConnections)idFuncion);
        2 referencias
        public static string GetNameConnection(eFunctionsConnections idFuncion) => $"name={(idFuncion).ToString()}";
```

Ilustración 29. Configuración de conexiones, para nuevos entornos y procesos de obtención o persistencia de datos

4.2.2 Autenticación de usuarios en la aplicación (Auth + TokenID)

Autenticación ideada bajo la clase *serializable* perteneciente al *System.Web.Security.FormAuthenticationTicket*[44] para realizar la autenticación de usuario y almacenarla, tanto en la correspondiente clave de sesión que será consultada por diferentes partes de la aplicación, como en caso de ser seleccionado, almacenar la información de usuario en cookie de sesión que se liberará pasadas 22 horas, de manera encriptada.

```
model.sTokenKey = repoUsuarios.usuarios_perfiles.FirstOrDefault().tokenID;
model.bPermitirAccesoAs = mwlSecurity.ComputumDeusEst(utilities.Resources.GetConnectioEntity(), model.userName.ToLower());

SessionHelper.SetSessionVariable(this.GetGuid(), "MW_CURRENT_USER_LOGGED", oCurrentUserLogged);

ResetSessionFilters();

Session["MW_CURRENT_USER_LOGGED"] = oCurrentUserLogged;

string userData = JsonConvert.SerializeObject(oCurrentUserLogged);
FormsAuthenticationTicket authTicket = new FormsAuthenticationTicket
        (1, repoUsuarios.userName, DateTime.Now, DateTime.Now.AddHours(22), true, userData);

string encTicket = FormsAuthentication.Encrypt(authTicket);
Response.Cookies.Add(new HttpCookie(FormsAuthentication.FormsCookieName, encTicket));
```

Ilustración 30. Detalle de la autenticación de un usuario recién logado en Sesión y FormAuthenticationTicket

Previo a este proceso de autenticación, tras la introducción por el usuario de la contraseña de acceso, esta se guarda en BBDD asociada al usuario de manera encriptada según el siguiente procedimiento SHA256.

```
mainLoadPro.bll\common\mwlSecurity.cs (1)
    35 : sSecureKey = getSecuredKey(sUserName.ToUpper().Trim() + "##" + userPass.Trim());
Contraer todo

1 referencia
private static string getSecuredKey(string sPreKey)
{
    using (var sha = new System.Security.Cryptography.SHA256Managed())
    {
        byte[] textData = System.Text.Encoding.UTF8.GetBytes(sPreKey);
        byte[] hash = sha.ComputeHash(textData);
        string sSecured = BitConverter.ToString(hash).Replace("-", String.Empty);

        return sSecured;
    }
}
```

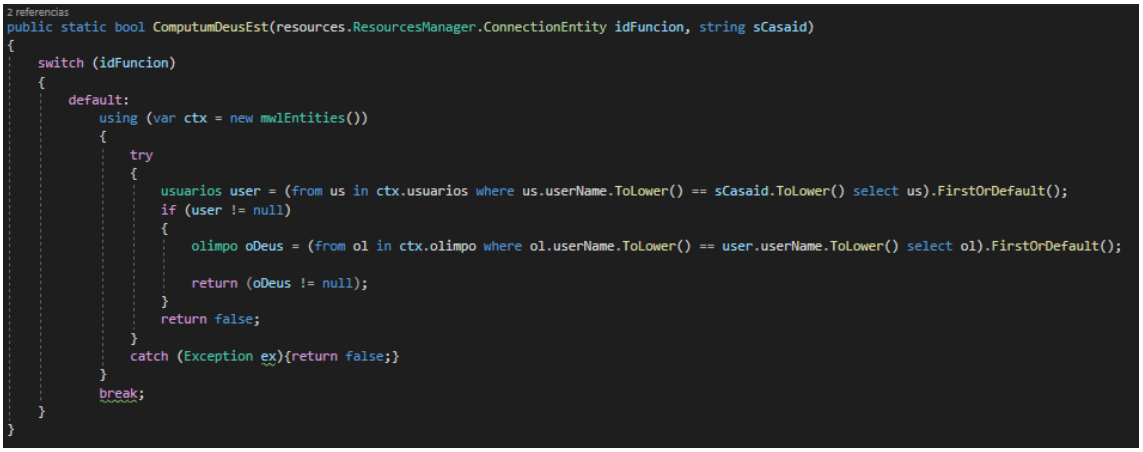
Ilustración 31. Obtención de clave segura de usuario basada en hash de encriptación SHA256.

4.2.3 Acceso mediante "impersonalización" de usuario.

Debido la naturaleza expansiva del negocio que engloba este desarrollo, las acciones están orientadas hacia un mantenimiento administrativo de las operaciones y datos que puedan consultar los usuarios finales, por ello, se ha habilitado un método de acceso no invasivo que permite acceder en sesión como si se tratase de otro usuario, funcionalidad solo permitida para usuarios que existan en la tabla de BBDD `Olimpo´, de esta manera mediante las propias credenciales del usuario, el Administrador podrá ver que es lo que ve el usuario, y contrastar así ciertas funcionalidades que puedan requerir una modificación o

resolver de manera más eficaz una incidencia que un usuario pueda tener. A continuación, se muestra proceso implementado para permitir el inicio impersonalizado del usuario.

En apartado dedicado al muestreo de pantallas en ejecución, se detalla pantalla para este fin.



```
2 referencias
public static bool ComputumDeusEst(resources.ResourcesManager.ConnectionEntity idFuncion, string sCasaid)
{
    switch (idFuncion)
    {
        default:
            using (var ctx = new mwlEntities())
            {
                try
                {
                    usuarios user = (from us in ctx.usuarios where us.userName.ToLower() == sCasaid.ToLower() select us).FirstOrDefault();
                    if (user != null)
                    {
                        olimpo oDeus = (from ol in ctx.olimpo where ol.userName.ToLower() == user.userName.ToLower() select ol).FirstOrDefault();

                        return (oDeus != null);
                    }
                    return false;
                }
                catch (Exception ex){return false;}
            }
            break;
    }
}
```

Ilustración 32. Detalle de comprobación de que existe usuario como admitido en Olimpo, adicionalmente se realiza comprobación de contraseña.

4.2.4 Almacenamiento y explotación permisos usuario

El procedimiento de almacenamiento de los permisos del usuario se encuentra dividido en 2 partes (perfiles y permisos). Por un lado, necesitamos un usuario dado de alta en la BBDD. Este usuario, puede contener varios perfiles, no solo por tipo, puesto que puede tener un tipo de perfil denominado para cada ejercicio, de manera que en cada uno de estos perfiles puede albergar información independiente, por tanto.

1ª parte, lo primero que necesita nuestro usuario es un perfil que le habilite a navegar por los ejercicios que le sean necesarios. Para ello debemos elegir entre los perfiles.

ADMIN → Perfil reservado para administradores de la herramienta, puede hacer cualquier tipo de acción sobre los datos y operaciones.

PLAN- * → Perfil encargado de realizar la primera versión de horas necesarias para la consolidación del plan operativo a nivel empresarial de ese ejercicio. Está subdividido en Lectura, Escritura y Validador, este último cercano al papel de administrador a la hora de manipular datos.

COSTC-* → Perfil encargado de realizar las comprobaciones, realizar sus planificaciones y rectificaciones de horas a nivel transversal de la compañía mediante la información aportada por los departamentos. Está igualmente subdividido en Lectura, Escritura y Validador, siendo este último el encargado de la validación de los datos introducidos por su departamento.

CHIEF-* → Perfil encargado de realizar las comprobaciones, realizar sus planificaciones y rectificaciones de horas a nivel transversal de la información de los proyectos existentes. Esta subdividido en Escritura y Validador, siendo este último el encargado de validar la información a nivel del su proyecto.

2ª parte: lo que necesitamos es nutrir de información de acceso el perfil seleccionado. Este perfil asociado a Role+Ejercicio, contendrá información de los departamentos y proyectos que el usuario del perfil puede visitar, escribir o validar. De esta manera, un PLAN* tendrá perfil a toda la estructura de departamentos/proyectos, un COSTC* tendrá acceso a todos los proyectos para un determinado departamento, y CHIEF* tendrá acceso a todos los departamentos de un determinado Proyecto. En base a estas directrices encontramos las tablas *usuarios_permisos_cecos* y *usuarios_permsisos_programas*, que junto a los perfiles determinan la matriz cúbica de permisos de un usuario que posicionará que puede hacer y como cada usuario.

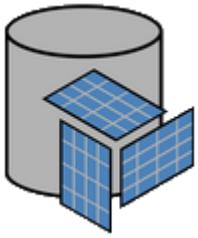


Ilustración 33. Representación de cúbica con modelo relacional de base de datos tradicional.

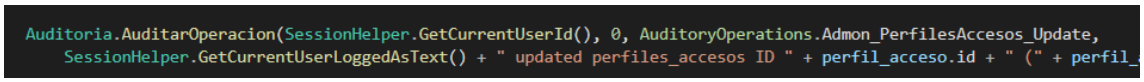
4.2.5 Mantenimiento de la información en la navegación (Session+Cookies+TokenID)

Durante la navegación, debemos conservar algunos de los datos que identifican al usuario en la aplicación, así como otro tipo de información que al mantener en "memoria" será de más rápido acceso que si la recuperamos de nuevo de la BBDD. Es, por tanto, que este tipo de información, particular a cada usuario podemos mantenerla en la sesión de usuario que los navegadores ofrecen para una mejor experiencia manejo de los datos.

En lo referente a la propia información de usuario mantenemos las siguientes claves de usuario y *helpers*[45] que nos permiten tratar más cómodamente este tipo de información.

```
43 referencias
public static int GetCurrentUserId(string sSessionId = "")
{
    return ((mainLoadPro.data.usuarios)HttpContext.Current.Session[sSessionId + "_" + "CurrentUser"]).id;
}

46 referencias
public static string GetCurrentUserLoggedAsText(string sSessionId = "")
{

    return ((mainLoadPro.data.usuarios)SessionHelper.GetCurrentUser(sSessionId)).nombre + " " +
        ((mainLoadPro.data.usuarios)SessionHelper.GetCurrentUser(sSessionId)).apellidos +
        " (" + ((mainLoadPro.data.usuarios)SessionHelper.GetCurrentUser(sSessionId)).userName + ")";
}

6 referencias
public static int GetCurrentEjercicio(string sSessionId = ""){
    return Int32.Parse(GetSessionVariable(sSessionId, "MW_USER_EJERCICIO").ToString());
}
```

Ilustración 34. Ejemplo de métodos incluidos en clase SessionHelper para recuperación y gestión de variables de sesión de usuario.

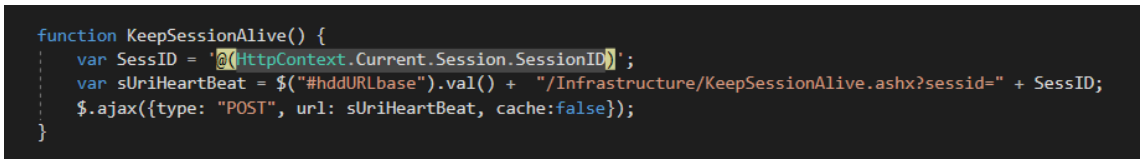


```
Auditoria.AuditarOperacion(SessionHelper.GetCurrentUserId(), 0, AuditoryOperations.Admon_PerfilesAccesos_Update,
    SessionHelper.GetCurrentUserLoggedAsText() + " updated perfiles_accesos ID " + perfil_acceso.id + " (" + perfil_
```

Ilustración 35. Detalle de uso clase SessionHelper en operaciones de auditoría de usuario.

4.2.6 Mantenimiento de la sesión de usuario activa (keepSessionAlive)

Para asegurar que la navegación de usuario no sufre cortes ni redirecciones no deseadas a la página de login, se ha implementado funcionalidad de servidor, llamada desde cliente, mediante llamada recursiva cada cierto número a determinar de segundos, que evita que se pierda la sesión de usuario, escribiendo en clave de sesión de usuario, el valor de fecha completa de la operación, refrescando de esta manera la sesión del navegador. De esta manera el usuario pueda navegar donde lo dejo en pausa, evitando pérdidas de datos y tiempo en la visita a la aplicación.



```
function KeepSessionAlive() {
    var SessID = '@(HttpContext.Current.Session.SessionID)';
    var sUriHeartBeat = $("#hddURLbase").val() +  "/Infrastructure/KeepSessionAlive.ashx?sessid=" + SessID;
    $.ajax({type: "POST", url: sUriHeartBeat, cache:false});
}
```

Ilustración 36. Detalle de la llamada desde cliente a método de servidor encargado de refrescar la sesión de usuario.

```
public class KeepSessionAlive : IHttpHandler, IRequiresSessionState
{
    0 referencias
    public bool IsReusable { get { return false; } }

    0 referencias
    public void ProcessRequest(HttpContext mContext)
    {
        try
        {
            //Recibe también por query el SessionID por si fuese necesario (sessid)
            if (mContext.Session != null)
            {
                mContext.Session["APP_KEEP_SESSION_ALIVE"] = DateTime.Now;
                string sSessionID = HttpContext.Current.Request.QueryString["sessid"].ToString();
               mainLoadPro.mvc.utilities.SessionHelper.registerUserAlive(sSessionID);
            }
        }
        catch(Exception ex)
        {
            string sEx = ex.ToString();
        }
    }


}
```

Ilustración 37. Método de servidor encargado de realizar el refresco de la sesión. Página de servicio ashx

4.2.7 Aseguramiento de zonas de acceso restringido

Con motivo de la implementación de acciones de seguridad en la navegación del usuario y que a su perfil no se le permitan acciones o navegaciones restringidas, se ha implementado la funcionalidad *mainLoadProAuthorize*, la cual se asigna como directiva en los controladores de las áreas existentes, indicando el acceso permitido en cada una de estas, en base a correspondiente fichero de recursos contenedor de áreas.

```
/// <summary>
/// Permite el acceso a los diferentes controladores de la app en base a la instancia
/// que se está ejecutando.
/// </summary>
10 referencias
public class mainLoadProAuthorize : AuthorizeAttribute
{
    private resources.ResourcesManager.Areas _area;
    private List<resources.ResourcesManager.ConnectionEntity> _modulos;

    /// <summary>
    /// Acceso al mismo controlador/acción para un área determinado.
    /// </summary>
    /// <param name="area"></param>
    6 referencias
    public mainLoadProAuthorize(resources.ResourcesManager.Areas area)
    {
        _modulos = new List<resources.ResourcesManager.ConnectionEntity>();
        _area = area;
    }
```

Ilustración 38. Detalle de la clase funcional de Autorización (Authorize)

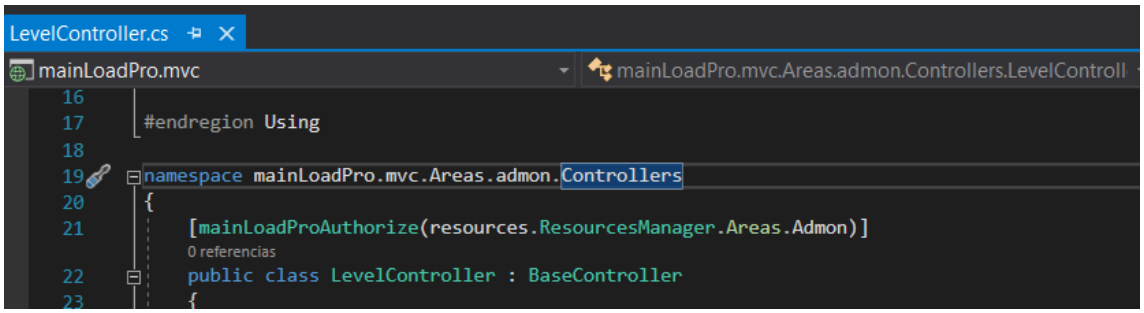


Ilustración 39. Detalle del uso de la clase mainLoadProAuthorize en área de administración de niveles WBS.

4.2.8 Sistema de Log y trazas

Como parte fundamental del control de la salud del aplicativo, el análisis que un Log puede aportar al correcto funcionamiento y detección de problemas es crucial para la evolución como software. Por ello, el sistema de log seleccionado, log4Net, ha sido basado en un modelo de BBDD, donde las operaciones problemáticas capturadas mediante instrucciones `catch´ y trazas de control y seguimiento son incluidas en base de datos, configurada según los parámetros indicados a continuación. El uso de este sistema permite un rápido análisis de lo que está pasando en tiempo real, y ofrece la posibilidad de ser fácilmente explotado o cedido a otros analistas, no desarrolladores, para el seguimiento y control de incidencias, el hecho de encontrase de manera tipada y clasificada ofrece consultas de la información con consultas SQL.

```
<log4net>
   <appender name="AdoNetAppender" type="log4net.Appender.AdoNetAppender">
     <bufferSize value="1" />
     <connectionType value="MySql.Data.MySqlClient.MySqlConnection, MySql.Data, Version=8.0.20.0, Culture=n
     <connectionString value="Server=localhost;Database=**; Uid=**;Pwd=**;" />
     <commandText value="INSERT INTO Log4Net_Error(Date,Thread,Level,Logger,Message,Exception) VALUES (?log
     <parameter>

 ...
```

Ilustración 40. Detalle de la configuración web.config necesaria para log4Net en BBDD

id	Date	Thread	Level	Logger	Message
1	2023-05-20 18:51:32	24	ERROR	mainLoadPro.mvc.Services.Logging.Log4Net.Lo...	Error in Path :/op/ListView/ApplyFilters Raw Url...
2	2023-05-26 19:33:37	6	ERROR	mainLoadPro.mvc.Services.Logging.Log4Net.Lo...	Error in Path :/op/ListView Raw Url :/op/ListVie...
3	2023-05-26 19:35:54	7	ERROR	mainLoadPro.mvc.Services.Logging.Log4Net.Lo...	Error in Path :/op/ListView Raw Url :/op/ListVie...
4	2023-05-26 19:49:18	7	FATAL	mainLoadPro.mvc.Services.Logging.Log4Net.Lo...	Error in Path :/op/ListView Raw Url :/op/ListVie...

Ilustración 41. detalle de la información almacenada en BBDD por log4Net

Adicionalmente, al este sistema de log mediante base de datos, se ha implementado un proceso de auditoría de acciones para registro de cualquier acción que realice el usuario, de esta manera, los administradores de la plataforma puede obtener de manera detallada que ocurre en el aplicativo y que acciones han sido realizadas por cada uno de los usuarios registrados, a continuación, se muestra cómo se realizaría el registro de un acceso y también detalle de alguno de los registros de auditoría realizado.

Al añadir este log de auditoria adicional, podemos trazar acciones de usuario, controlar errores en base a las acciones realizadas o seguimiento de incidentes de la aplicación.

```
[OutputCacheAttribute(VaryByParam = "*", Duration = 0, NoStore = true)]
0 referencias
public ActionResult Index()
{

    Auditoria.TracertOperationLag(SessionHelper.GetCurrentUserId(), Auditoria.LagOperation.BeginRequest);

    Auditoria.AuditarOperacion(SessionHelper.GetCurrentUserId(), SessionHelper.GetCurrentEjercicio(),
                               AuditoryOperations.OP_Enter, SessionHelper.GetCurrentUserLoggedAsText());

    Models.opListView oModel = new Models.opListView();
    RellenarModelo(ref oModel);

    Auditoria.TracertOperationLag(SessionHelper.GetCurrentUserId(), Auditoria.LagOperation.EndRequest);
    return View(oModel);

}
```

Ilustración 42. Detalle de uso de auditoría de acciones de usuario.

id_usuario	start_session	operation	mparam	info	reg	um	um_ts
1	2023-05-29 07:42:32	LAG BeginRequest	/op/ListView[GET]		NULL	NULL	2023-05-29 05:42:32
1	2023-05-29 05:42:35	op_enter	2023	RAIMUNDO PRIET…	OP_Enter	NULL	2023-05-29 05:42:35
1	2023-05-29 07:42:36	LAG EndRequest	/op/ListView[GET]		NULL	NULL	2023-05-29 05:42:36
0	2023-05-29 07:42:38	LAG BeginRequest	/op/ListView/GetAnnosTAE?_=1685338957983[GET]		NULL	NULL	2023-05-29 05:42:38
0	2023-05-29 07:42:38	LAG BeginRequest	/op/ListView/GetAnnosPL?idProgramLine=2&_=16853389…		NULL	NULL	2023-05-29 05:42:38
0	2023-05-29 07:42:38	LAG BeginRequest	/Index/GetUserProfile?_=1685338957988[GET]		NULL	NULL	2023-05-29 05:42:38
0	2023-05-29 07:42:38	LAG BeginRequest	/op/ListView/GetAnnosPL?idProgramLine=4&_=16853389…		NULL	NULL	2023-05-29 05:42:38
0	2023-05-29 07:42:38	LAG BeginRequest	/op/ListView/GetAnnosPL?idProgramLine=1&_=16853389…		NULL	NULL	2023-05-29 05:42:38
0	2023-05-29 07:42:38	LAG BeginRequest	/op/ListView/GetAnnosPL?idProgramLine=3&_=16853389…		NULL	NULL	2023-05-29 05:42:38
0	2023-05-29 07:42:39	LAG BeginRequest	/Index/GetPermissionsList?year=2023&yeardesc=2023&…		NULL	NULL	2023-05-29 05:42:39
0	2023-05-29 07:43:39	LAG BeginRequest	/Index/Home[GET]		NULL	NULL	2023-05-29 05:43:39
1	2023-05-29 05:43:39	login	0	RAIMUNDO PRIET…	Login	NULL	2023-05-29 05:43:39
0	2023-05-29 07:43:39	LAG BeginRequest	/Index/GetPermissionsList?year=2023&yeardesc=2023[G…		NULL	NULL	2023-05-29 05:43:39
0	2023-05-29 07:43:39	LAG BeginRequest	/Index/SetYearAndProfile_newWay?year=2023&yeardes…		NULL	NULL	2023-05-29 05:43:39

Ilustración 43. Detalle de registros recuperados `access_current´, auditoria de acciones usuario

5. Funcionalidades asociadas a FrontEnd

5.1 Pantallas de mantenimiento tablas maestras

A través del siguiente enlace, mostrado como opción del menú mainLoadPro Modules, accedemos a la administración de las tablas maestras y datos principales de la aplicación.

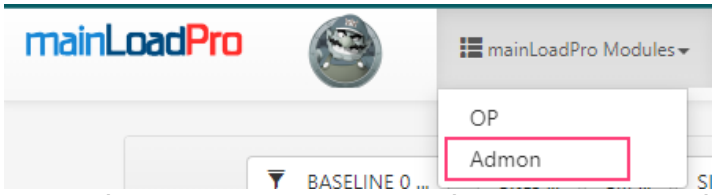


Ilustración 44. Detalle de acceso a módulo de administración.

Mantenimiento de WBS

El mantenimiento de la *WBS* implica la implementación de varias vistas asociadas a los niveles que representa la entidad programas de la aplicación, es decir *ProgramaLine, ProgramaAreasSuper, ProgramaAreas y Programas*.

Mediante un acceso escalado a los datos, podemos navegar por las diferentes entidades que conforman la WBS del proyecto, tal y como se representa en las capturas mostradas a continuación.
Para este proyecto tan solo se ha implementado el *CRUD* de las entidades a continuación mostradas, quedando por tanto fuera de alcance, aunque implementable según lo hecho para WBS el resto de las entidades maestras del modelo *mainLoadPro*.

Mantenimiento entidad Programa Line (PL)

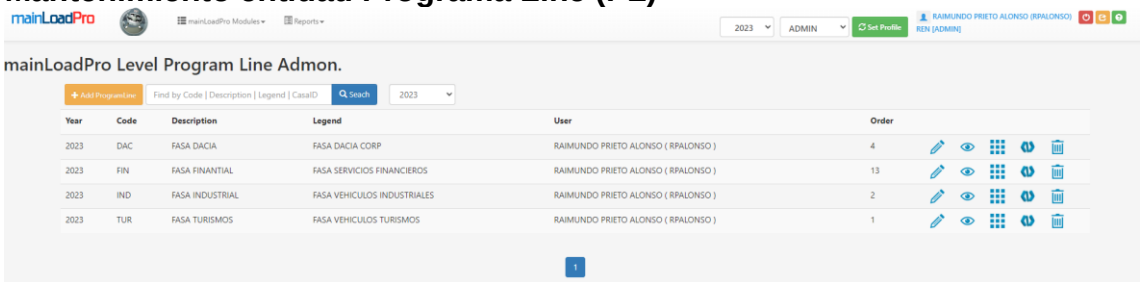


Ilustración 45. Mantenimiento entidad Programa Line

Mantenimiento entidad Programa Áreas Super (Level 0)

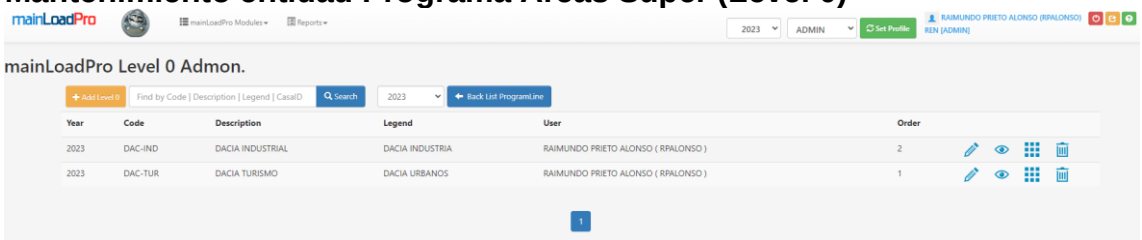


Ilustración 46.Mantenimiento entidad Programa Áreas Super

Mantenimiento entidad Programa Áreas (Level 1)

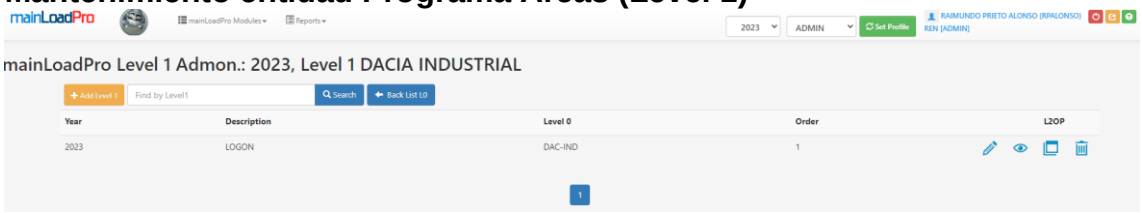


Ilustración 47. Mantenimiento entidad Programa Áreas

Mantenimiento entidad Programas (Level 2)

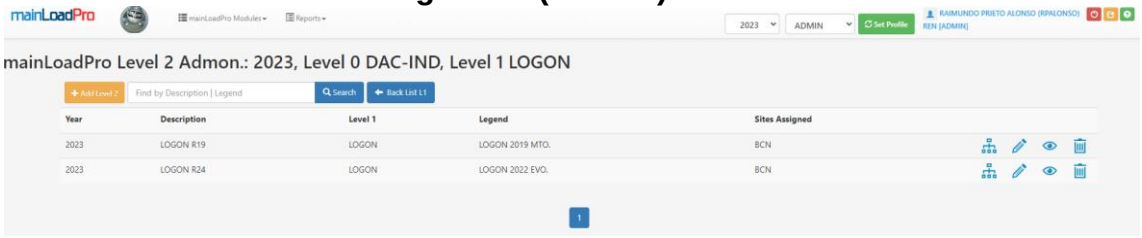


Ilustración 48. Mantenimiento entidad Programas

Mantenimiento entidad Sites (relacionado desde Level2)

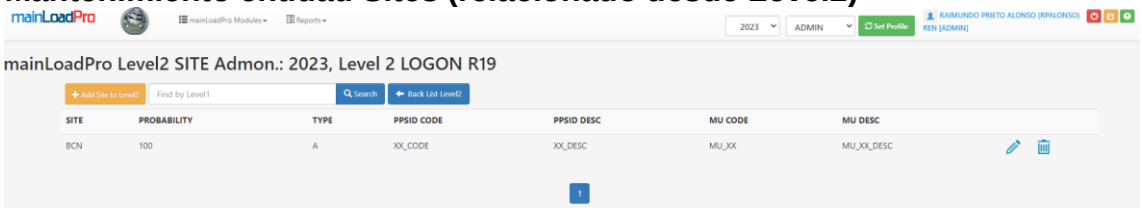


Ilustración 49. Mantenimiento entidad Sites

5.2 Pantallas navegación

Todas las vistas de pantalla principal, es decir, que no sean parte de una vista en modo pop-up, tienen asociada una vista madre Layout que ofrece características de script común, capacidad de muestra de ventanas pop-up, y en general la carga principal de los estilos permitidos por la propia hoja definida como por los inherentes al uso de Bootstrap. Adicionalmente, este Layout contiene información del usuario que está logado en sesión en ese momento, así como opciones propias de navegación por el sistema. A continuación, se muestran pantallas principales de la navegación del aplicativo.

Vista Login

Ilustración 50. Vista inicial de la aplicación en la que debemos introducir usuario y contraseña asociada a la cuenta.

Vista Login como otro usuario

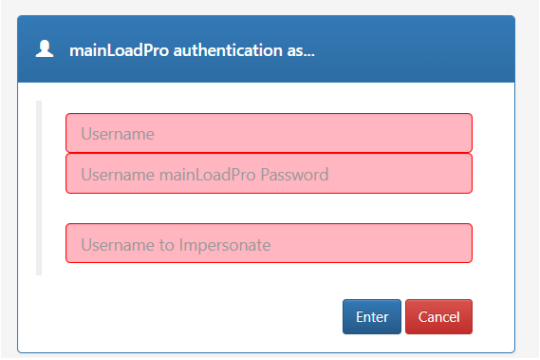


Ilustración 51. Vista de impersonalización como otro usuario, mediante credenciales propias de usuario, y nombre de usuario a impersonar. Vista Layout de Login

Vista de solicitud de recuperación de contraseña

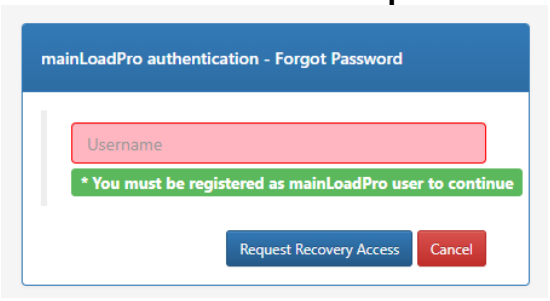


Ilustración 52. Vista de recuperación de contraseña, mediante introducción de usuario, y envío de email al correo electrónico asociado con enlace de recuperación.

Vista de solicitud de alta de usuario en el sistema

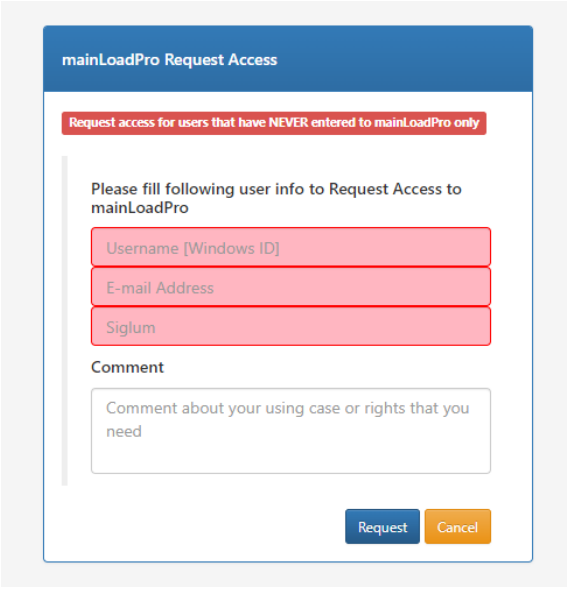


Ilustración 53. En caso de tratarse de usuario que no esté dado de alta y quiera hacerlo, se envía comunicación a los administradores con detalles del usuario

Vista Filtros Parrilla

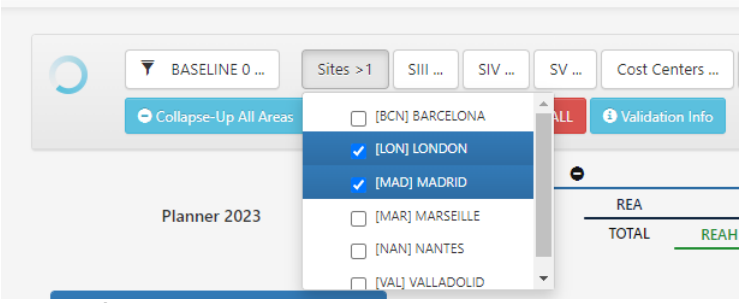


Ilustración 54. Detalle de filtros de vista de parrilla ppal. habilitada multi-selección de filtro.

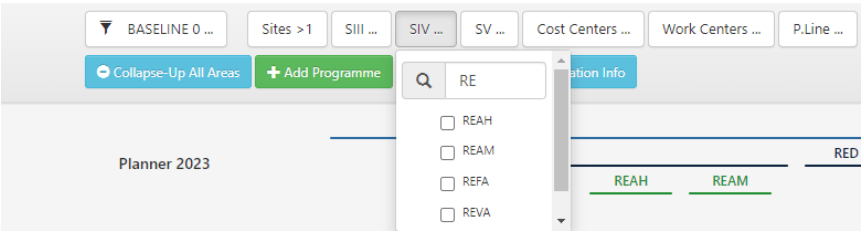


Ilustración 55. Más filtros asociados a la vista parrilla.

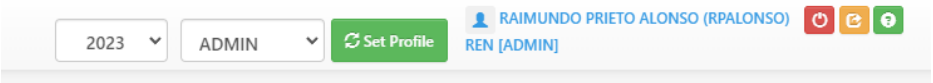


Ilustración 56. Detalle de componentes de cambio de perfil de usuario e información del usuario en sesión, integrado en Layout

Vista Parrilla Plegada

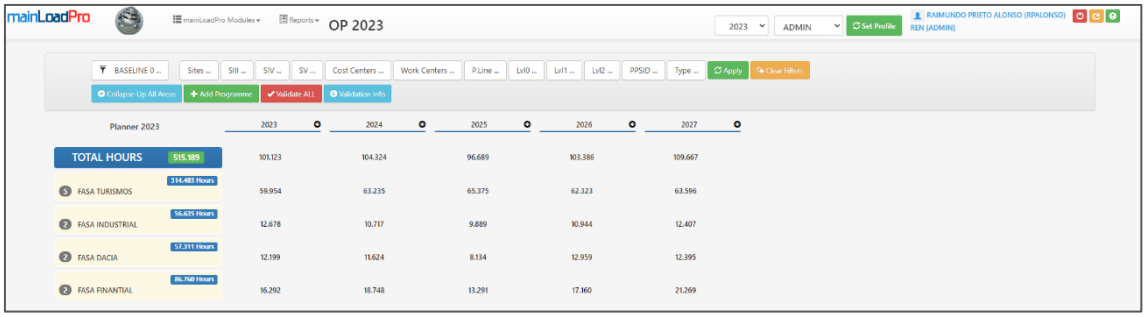


Ilustración 57. Vista parrilla al completo, sin apertura de agrupadores de información.

Vista Parrilla Desplegada

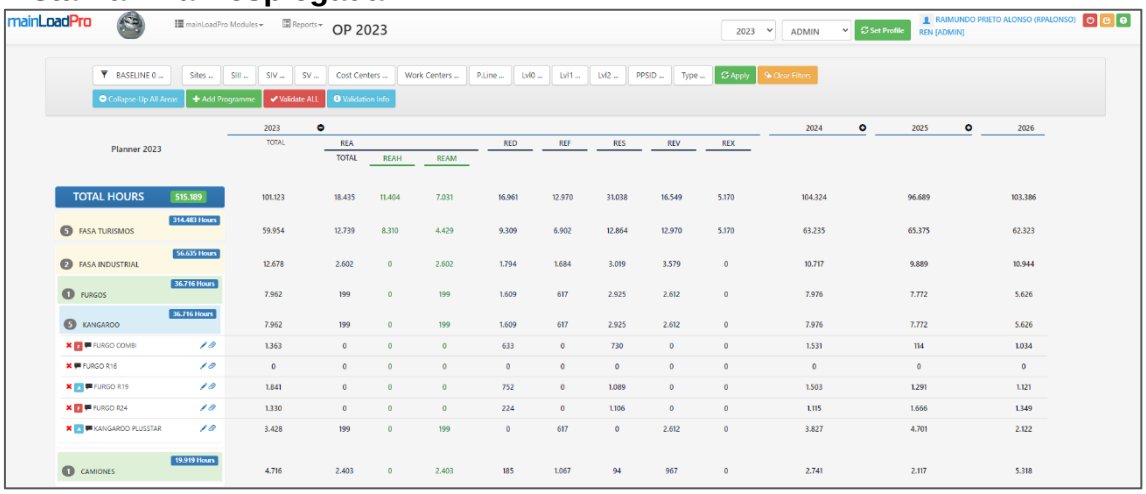


Ilustración 58. Vista de parrilla al completo, con uno de los niveles de agrupación de programas desplegado en su máxima escala.

Vista detalle de información de Programa

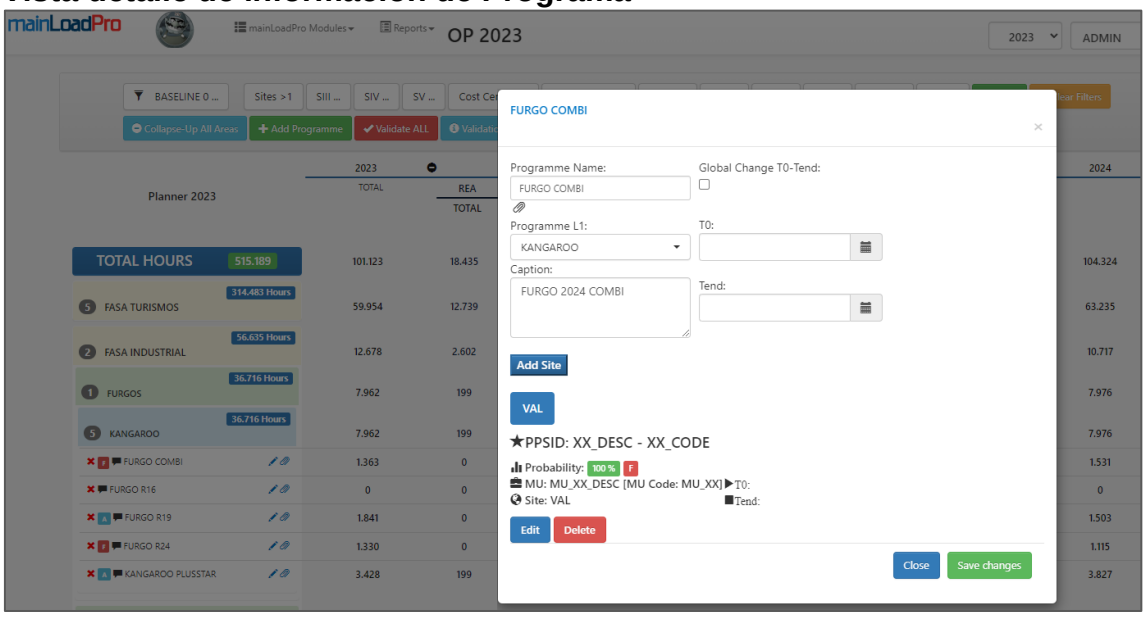


Ilustración 59. Vista de la información mostrada al editar la información del nivel más bajo del agrupador de programas.

Vista detalle de Adjuntos de Programa

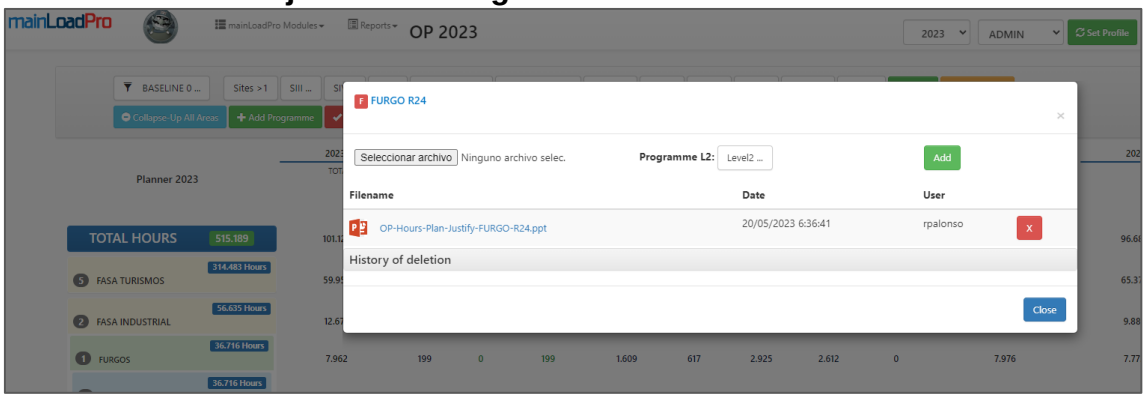


Ilustración 60. Vista popup de listado de adjuntos asociado a un programa

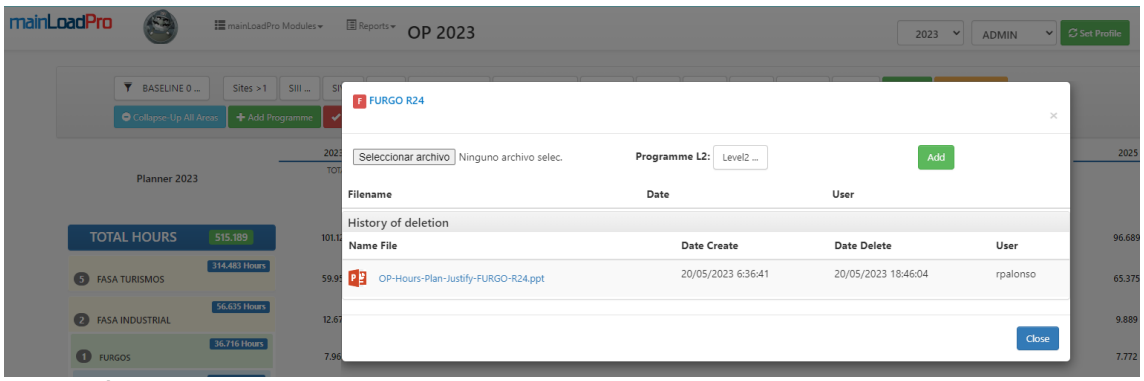


Ilustración 61. Vista detalle de adjunto de programa eliminado

Vista edición de horas a nivel de Unidad de Trabajo – Programa

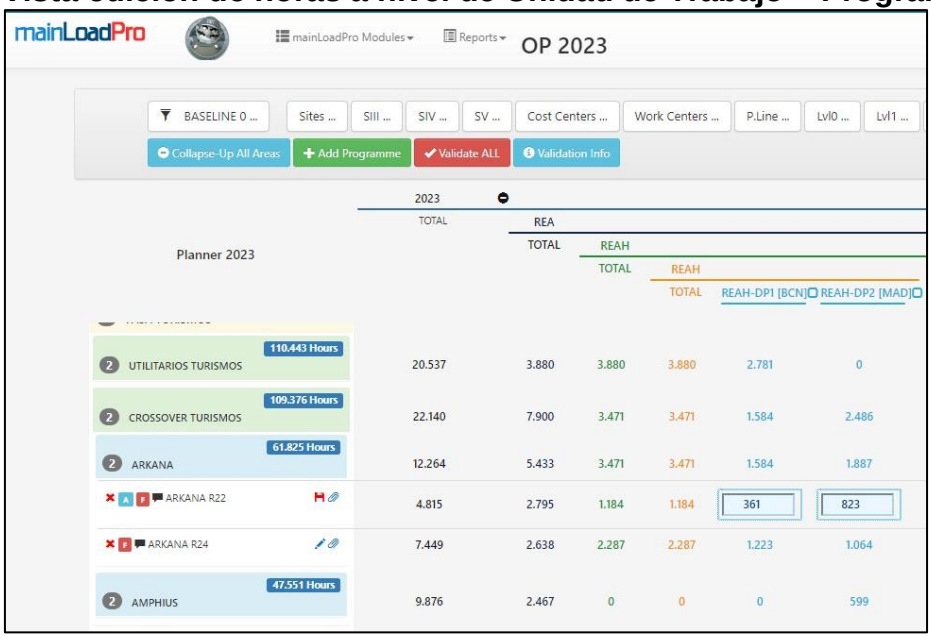


Ilustración 62. Vista detalle de edición de horas

Vista conformación datos modificados

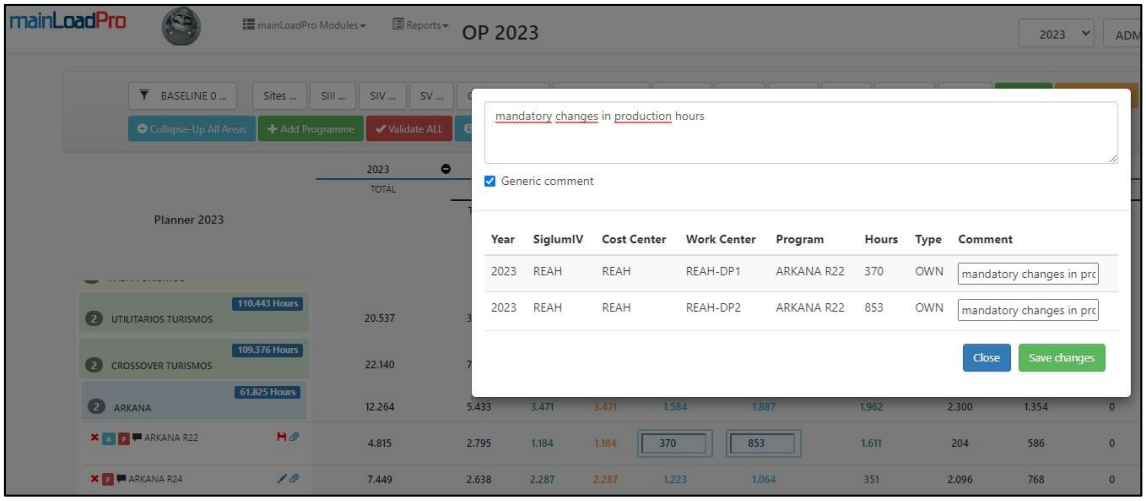


Ilustración 63. Vista detalle de confirmación de todas las horas modificadas en parrilla

Vista Información validaciones pendientes

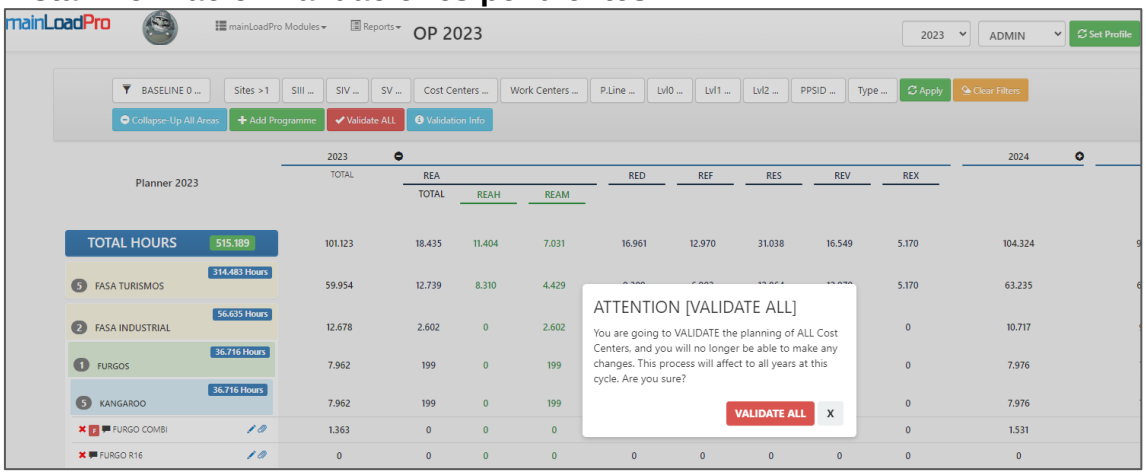


Ilustración 64. Vista detalle de Validación completa de la versión Cero de datos.

5.2.1 Praxis de carga de información (Parrilla)

El mecanismo de carga de datos de la página principal de información se ha basado en la obtención de la información a demanda según detalle necesario. Es decir, mediante carga de Ajax de los métodos del controlador de servidor en los que se solicita información de las horas establecidas como planificaciones para un determinado nivel (Programa Line, SuperÁrea, Área o Programa), correspondiente agrupación departamental (Centros Coste y WorkCenters), un determinado año de visualización (2023-2027) y la versión correspondiente en proceso de actualización (baseline). De este modo se agrupan sumatorios de horas que representan el amonto subtotal de horas para cada una de las vistas disponibles.
Mediante la carga dinámica en cada una de las vistas padre podemos hacer una navegación a demanda del usuario, sin precarga de datos que puede presentar algún cuello de botella. El método de carga asíncrona realizado desde Ajax Jquery, permite que se ejecuten varias descargas de información al mismo tiempo, convirtiendo la carga total del aplicativo en una serie de llamadas asíncronas que permiten la visualización fluida de los contenidos (planificaciones previstas)

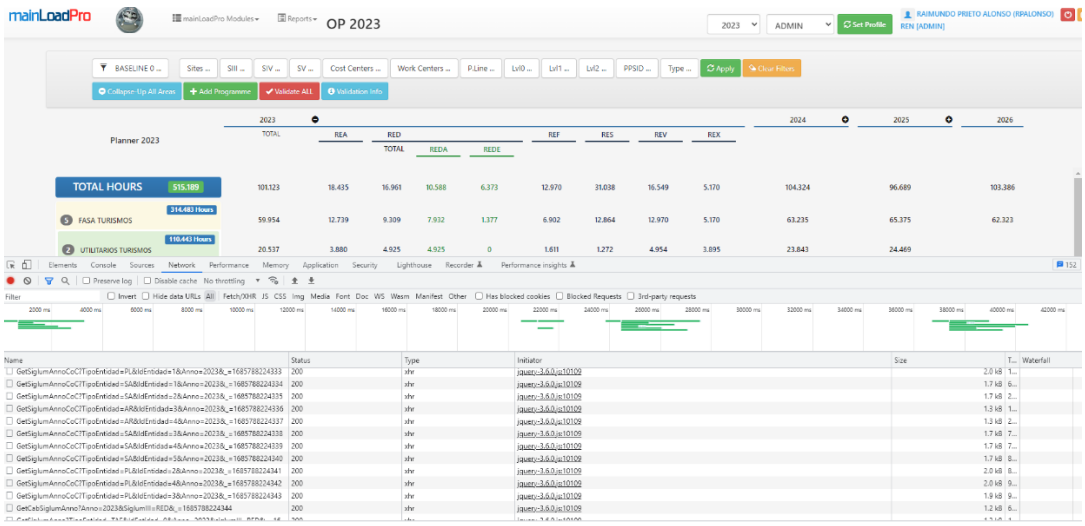
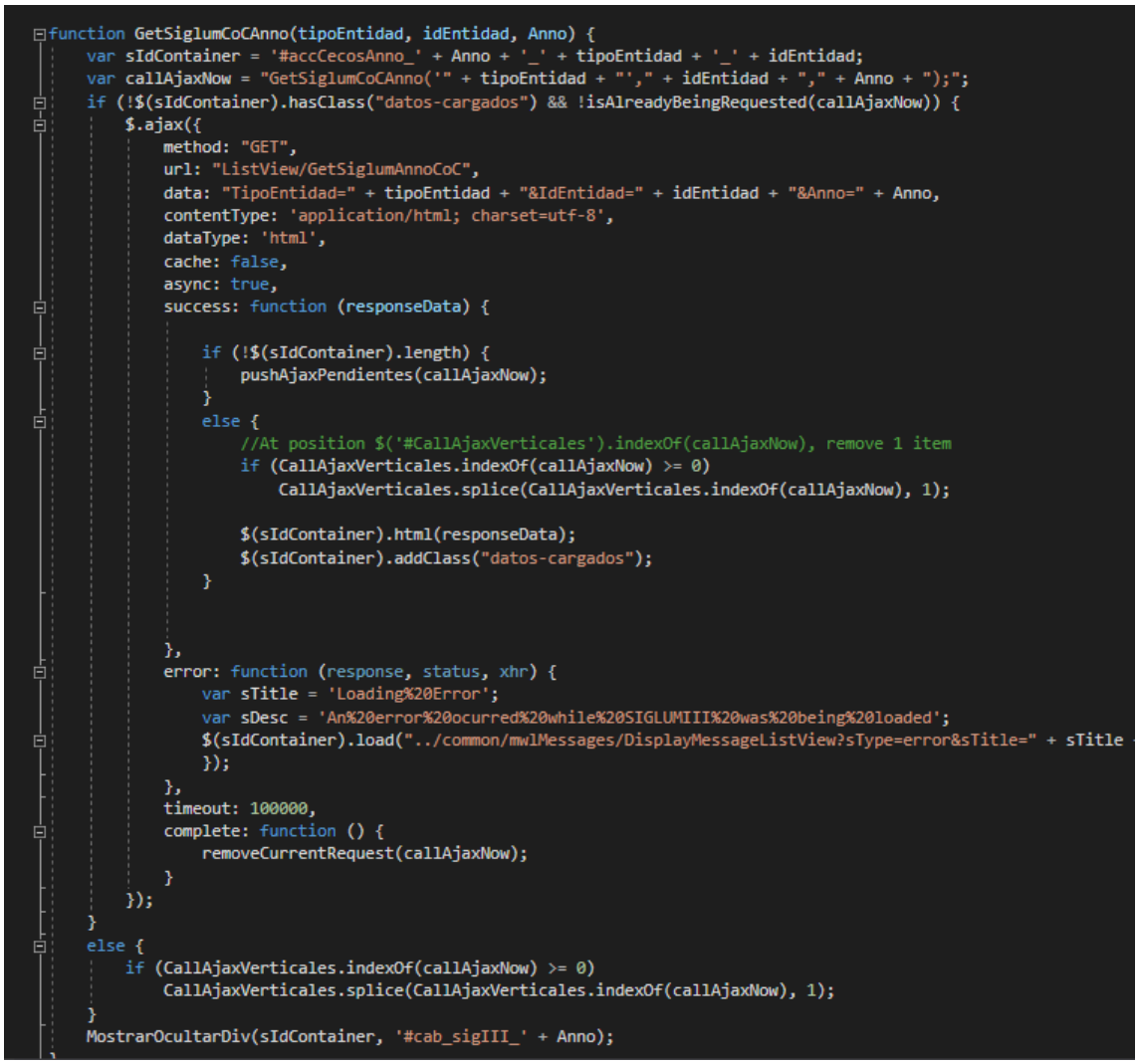


Ilustración 65. Detalle de pestaña `Network´ desde herramientas de desarrollador de Chrome, control de carga de datos

Las llamadas anteriormente citadas son almacenadas en un apilador de llamadas en código cliente JS que se va vaciando según las llamadas hayan sido completadas, es decir, renderizadas por pantalla. De esta manera, en caso de producirse algún error puntual de perdida de paquetes, al no haberse liberado esta llamada del apilador, volverá a ser llamada para su renderizado, en el momento que el apilador se vacíe, la carga de la parrilla se dará por completada, identificada por el fin de las peticiones Ajax.
A continuación, se muestra detalle de un método que usa el apilador de llamadas para la carga de datos, en la misma se puede ver como se evalúa si ya se ha cargado completamente los elementos que estaban almacenados con repetición recursiva de llamada en caso de que no se encuentre vacía, por el contrario, si se encontrase vacío el apilador, se libera la petición de llamada Ajax que contenía todas las llamadas mencionadas en el apilador.



```
function GetSiglumCoCAnno(tipoEntidad, idEntidad, Anno) {
    var sIdContainer = '#accCecosAnno_' + Anno + '_' + tipoEntidad + '_' + idEntidad;
    var callAjaxNow = "GetSiglumCoCAnno('" + tipoEntidad + "'," + idEntidad + "," + Anno + ");";
    if (!$(sIdContainer).hasClass("datos-cargados") && !isAlreadyBeingRequested(callAjaxNow)) {
        $.ajax({
            method: "GET",
            url: "ListView/GetSiglumAnnoCoC",
            data: "TipoEntidad=" + tipoEntidad + "&IdEntidad=" + idEntidad + "&Anno=" + Anno,
            contentType: 'application/html; charset=utf-8',
            dataType: 'html',
            cache: false,
            async: true,
            success: function (responseData) {

                if (!$(sIdContainer).length) {
                    pushAjaxPendientes(callAjaxNow);
                }
                else {
                    //At position $('#CallAjaxVerticales').indexOf(callAjaxNow), remove 1 item
                    if (CallAjaxVerticales.indexOf(callAjaxNow) >= 0)
                        CallAjaxVerticales.splice(CallAjaxVerticales.indexOf(callAjaxNow), 1);

                    $(sIdContainer).html(responseData);
                    $(sIdContainer).addClass("datos-cargados");
                }


            },
            error: function (response, status, xhr) {
                var sTitle = 'Loading%20Error';
                var sDesc = 'An%20error%20ocurred%20while%20SIGLUMIII%20was%20being%20loaded';
                $(sIdContainer).load("../common/mwlMessages/DisplayMessageListView?sType=error&sTitle=" + sTitle
                });
            },
            timeout: 100000,
            complete: function () {
                removeCurrentRequest(callAjaxNow);
            }
        });
    }
    else {
        if (CallAjaxVerticales.indexOf(callAjaxNow) >= 0)
            CallAjaxVerticales.splice(CallAjaxVerticales.indexOf(callAjaxNow), 1);
    }
    MostrarOcultarDiv(sIdContainer, '#cab_sigIII_' + Anno);
}
```

Ilustración 66. Detalle de funcionamiento del apilador de llamadas asíncrono de planificaciones a demanda de usuario.

5.3 Librería métodos API principales entidades

Tal y como previamente se ha indicado a través del proyecto *mainLoadPro.api*, se implementa la librería de métodos API necesarios para la edición de las principales entidades de la aplicación. Aunque no es específicamente el modelo de funcionamiento de la aplicación, esta implementación API permite que sean explotadas desde la administración de entidades o para el uso de terceros, adicionalmente se dispone para el uso mediante la realización de Test unitarios del aplicativo. Así podemos encontrar el siguiente paquete de métodos API mostrados mediante la interfaz integrada de *Swagger*, paquete *Nuget* → *Swagger-Net*.

Los métodos API contienen los principales accesos requeridos para la construcción de una librería API, obtención elementos [*httpGet*], obtención elemento identificado [*httpGet*], nuevo elemento [*httpPost*], edición de elemento [*httpPut*] y borrado de elemento [*httpDelete*].

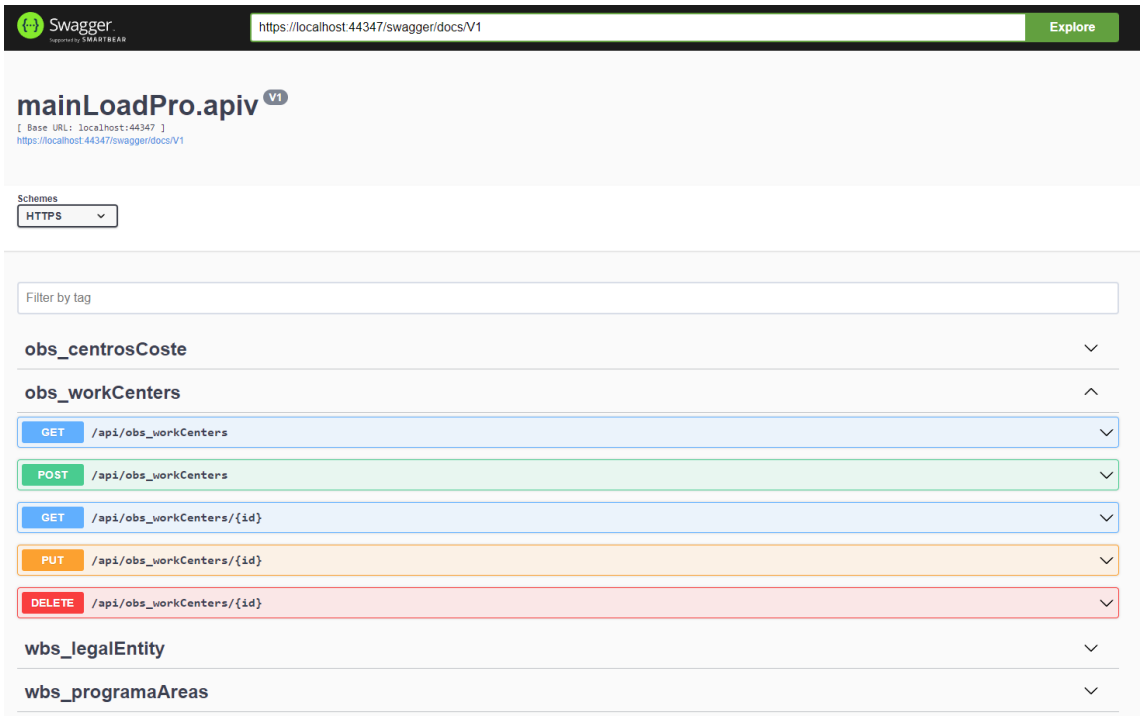
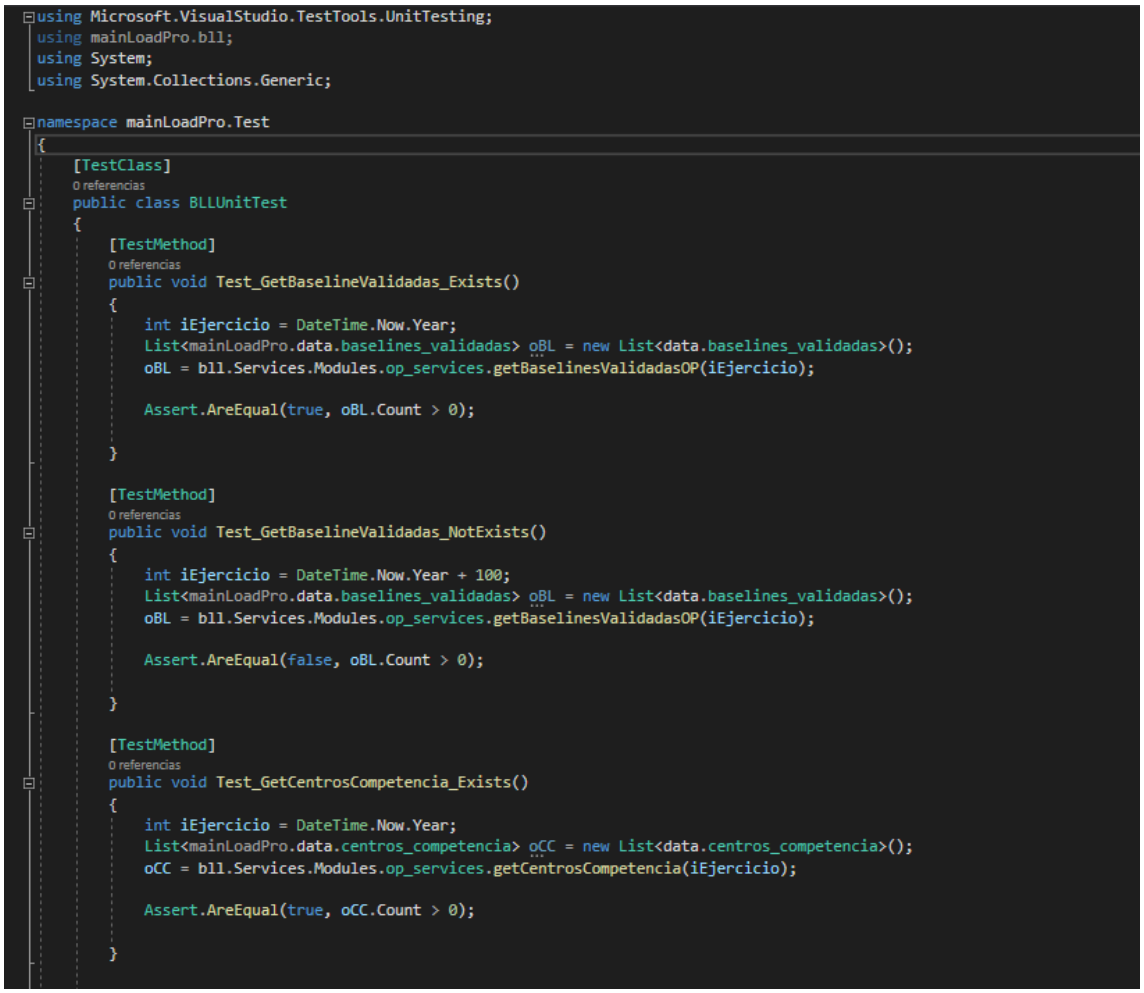


Ilustración 67. Detalle de ejecución librería API mainLoadPro.api con principales webMethods mostrados.

5.3.1 Librería casos de Test unitarios.

Para la realización de los test unitarios que determinen que se están realizando correctamente las operaciones de la aplicación, se ha implementado un proyecto que realiza test sobre las diferentes operaciones realizadas en el aplicativo. A pesar de ser un desarrollo no basado en la práctica *Test-driven development (TDD)*, las pruebas aseguran el funcionamiento general del aplicativo antes del despliegue en un entorno deslocalizado del desarrollo.

Dada la naturaleza del proyecto hacia la recuperación de información desde BBDD y la no estabilidad de test basados en procesos de recuperación desde una base de datos, los test implementados deben ir orientados a funcionalidades propias de la lógica de negocio y no al muestreo de datos, es por ello que los casos de test no son extensos en el aplicativo.
Es por ello que los test realizados sobre entidades se basan en la existencia de información y en el no fallo general del proceso a la hora de obtener resultados sobre las entidades concretas. Tal y como puede observarse en la ilustración mostrada a continuación.



```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using mainLoadPro.bll;
using System;
using System.Collections.Generic;

namespace mainLoadPro.Test
{
    [TestClass]
    0 referencias
    public class BLLUnitTest
    {
        [TestMethod]
        0 referencias
        public void Test_GetBaselineValidadas_Exists()
        {
            int iEjercicio = DateTime.Now.Year;
            List<mainLoadPro.data.baselines_validadas> oBL = new List<data.baselines_validadas>();
            oBL = bll.Services.Modules.op_services.getBaselinesValidadasOP(iEjercicio);

            Assert.AreEqual(true, oBL.Count > 0);

        }

        [TestMethod]
        0 referencias
        public void Test_GetBaselineValidadas_NotExists()
        {
            int iEjercicio = DateTime.Now.Year + 100;
            List<mainLoadPro.data.baselines_validadas> oBL = new List<data.baselines_validadas>();
            oBL = bll.Services.Modules.op_services.getBaselinesValidadasOP(iEjercicio);

            Assert.AreEqual(false, oBL.Count > 0);

        }

        [TestMethod]
        0 referencias
        public void Test_GetCentrosCompetencia_Exists()
        {
            int iEjercicio = DateTime.Now.Year;
            List<mainLoadPro.data.centros_competencia> oCC = new List<data.centros_competencia>();
            oCC = bll.Services.Modules.op_services.getCentrosCompetencia(iEjercicio);

            Assert.AreEqual(true, oCC.Count > 0);

        }
```

Ilustración 68. Detalle de implementación de un caso de Test orientado a las entidades del aplicativo.

5.4 Ejecución y Despliegue en Azure

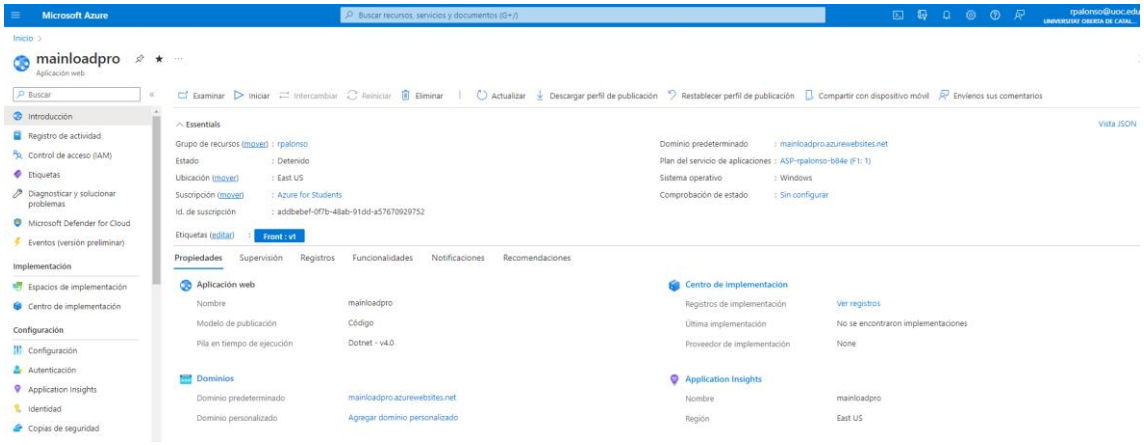
Una vez implementados los desarrollos, probados y chequeados, se ha determinado un despliegue de la aplicación en un entorno en la nube, la selección de la plataforma *cloud* ha sido mayormente determinada por la compatibilidad del entorno de desarrollo con dicha plataforma, de manera que este proceso, se pudiera agilizar lo mayormente posible. Es por ello que la selección ha sido de la plataforma Azure, de Microsoft, la cual posee capacidades de integración inherentes desde el IDE Visual Studio, si no tanto a nivel de BBDD si a nivel de aplicativo.

Inicialmente se determinó el despliegue automático mediante la creación de un pipeline en *github*, donde una vez confirmados los cambios que se realizan sobre la rama principal del desarrollo, se implementara la compilación automática del mismo y publicación en el *AppService* creado para este particular (mainloadpro.azurewebsites.net).

A pesar de este modelo, la no separación de los proyectos de desarrollo en varios proyectos de G*ithub*, ha complicado el que se ejecute de manera correcta el pipeline indicado, por lo que se ha tomado la determinación de usar las

capacidades de publicación del Visual Studio en entornos cloud. Si bien no queda representada tal publicación por la ejecución automática del pipeline, la necesidad de tomar acciones que dieran un empujón al proyecto fueron prioritarias para realizar este tipo de publicación que a continuación, detallo.

- Aplicación web: mainLoadPro
- Pila en tiempo de ejecución: Dotnet - v4.0
- Dominio predeterminado: mainloadpro.azurewebsites.net
- Tipo de plan: Plan de App Service
- Nombre: ASP-rpalonso-b84e
- Recuento de instancias: 1
- Región: East US



5.4.1 Publicación desde Visual Studio en Azure:

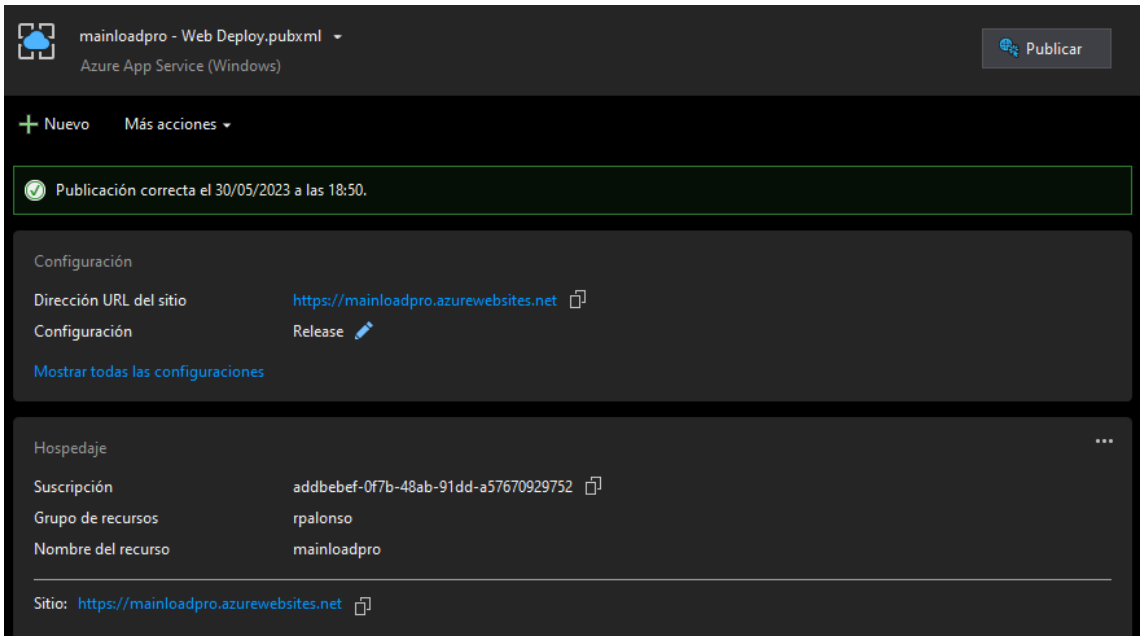


Ilustración 69. Detalle de ventana de publicación proyecto mainLoadPro.mvc en Azure

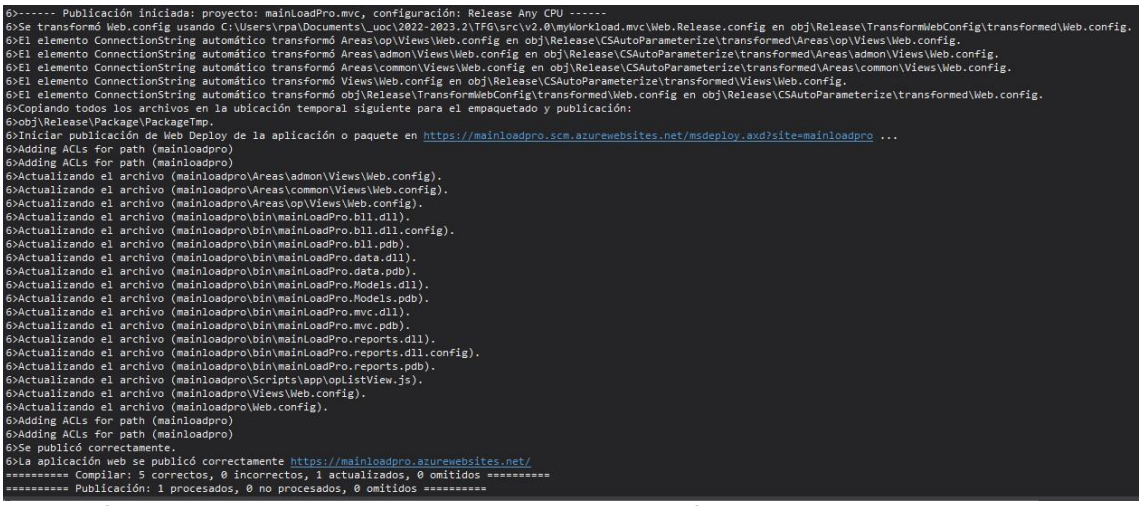


Ilustración 70. Detalle del reporte de salida de la publicación

La publicación de los entornos BBDD y aplicación se encuentra en las siguientes direcciones de la nube de Azure:

App Service mainLoadPro: https://mainloadpro.azurewebsites.net/
Servidor BBDD MySQL: mainloadprodb1.mysql.database.azure.com

6. Conclusiones

6.1 Principales conclusiones

En líneas generales considero que los resultados aportados en el proyecto son altamente satisfactorios en sentido de que cumplen con las expectativas inicialmente marcadas para conseguir una aplicación ágil capaz de mostrar grandes volúmenes de información de manera visual e intuitiva para el usuario final. Tal y como detallo en posteriores puntos, la complejidad del desarrollo ha impedido que pueda dedicar más tiempo a otros aspectos técnicos no directamente relacionados con la línea principal del desarrollo, pero en líneas generales, estoy satisfecho del resultado que muestro en este TFG.

6.2 Lecciones aprendidas

No cabe duda, que la primera de las lecciones aprendidas es que una mayor sencillez en la elección del proyecto hubiera agilizado el desarrollo, y por tanto, hubiera encontrado menos dificultades, convirtiéndose en un desarrollo más ágil y liviano.

En cuanto a las determinaciones técnicas, mis capacidades profesionales en el uso de motores de datos MySQL, así como .Net han ayudado a que pueda sacar este proyecto con relativa fluidez en lo que al desarrollo (codificación) se refiere, evitando grandes curvas de aprendizaje.

6.3 Reflexiones y objetivos

En cuanto a los logros de proyecto obtenidos, me satisface pensar que este puede ser un desarrollo útil en un entorno empresarial, pues se ajusta a la naturaleza y casuística departamental y de proyectos en la que se componen empresas ligadas a la industria. A pesar de ello, considero que un avance más ligero orientado a desarrollos completos en WebApi hubiesen aportado más valor al proyecto.

Como he comentado, creo que el mantra planteado para este proyecto puede ser aprovechado para realizar un sistema de validación de las estimaciones de proyectos en las que se elabore un ciclo de vida abierto a cada modelo planteado.

El proceso metodológico seleccionado, Scrum, es eficaz cuando estás rodeado de un equipo que te sugestiona implícitamente a la realización de las ceremonias, y seguimiento de actividades. Pues de muchas de ellas sale la resolución a bloqueos que el equipo de desarrollo puede tener en el periplo de la implementación.

6.5 Análisis póstumo

En el proceso de elaboración del este proyecto, he ido entendiendo que efectivamente como algunos otros que han pasado por este camino me habían comentado el TFG equivale a más de las 2 asignaturas que representan los 12 créditos. La complejidad a la que me he enfrentado, así como la interferencia de elementos externos en la elaboración del TFG, cual otras asignaturas pendientes, han hecho que la primera aproximación planificada para el proyecto haya sufrido ligeros cambios, en los cuales, además de contar con un sobre esfuerzo de la mano de obra, ha sido traducido como eliminación de una de las partes que a priori consideraba más atractivas del proyecto. La generación de un módulo IA capaz de determinar en base a las planificaciones establecidas, cuál era la tendencia con determinados elementos circunstanciales y evaluables.

Por otro lado, la elección de un framework .Net, como el 4.7.2, ha sido determinante para poder dar un avance ligero al proyecto, pues, aunque en un principio pensé hacerlo con .Net Core, pero la elevada curva de aprendizaje ponía en riesgo el desarrollo del proyecto.

Y aunque creo que en líneas generales el proyecto ha alcanzado un hito satisfactorio, podría haber evolucionado más como herramienta de no haber contado con problemáticas externas al mismo.

El proyecto podría ser mejorado mediante
- Mayor uso de servicios API REST.
- Una implementación 100% basada en TDD.
- Ampliación de la documentación de desarrollador y usuario.
- Una completa implementación de un entorno CI/CD

6.6 Elementos fuera de ámbito

Tal y como anteriormente he mencionado, para este proyecto, han quedado fuera de ámbito las siguientes implementaciones que en un principio quería acometer.

Por un lado, la realización de un aplicativo 100% basado en comunicación API, bien para la propia explotación o para la posterior explotación por terceros. Y, por otro lado, el desarrollo de un módulo IA capaz de orientar al planificador en las estimaciones a realizar basándose además de los datos ya existentes en el repositorio de datos, de elementos externos, definidos por la propia herramienta o el planificador.

7. Glosario

- Carga/Capacidad: Relación entre la carga de horas a realizar por una departamento o entidad laboral y los medios humanos de los que dispone para realizarlo.
- Consolidación (Datos): Aseguramiento de una versión de horas estimadas dentro de un ciclo de trabajo, de manera que varios actores puedas validar, evaluar, rectificar o ratificar los datos establecidos en una versión anterior por otro actor.
- Database First: Modelado de los datos de una aplicación a partir de la experiencia del usuario en las BDR, se basa en la realización de una base de datos robusta y equilibrada y posteriormente se procede a la generación de las entidades de acceso a esta BBDD como modelos de la aplicación.
- Departament Unit: entidad relativa a un departamento de trabajo, agrupadora de una o varias entidades “Department Unit Work”.
- Departament Unit Work: entidad atómica de unidad de trabajo, grupo de personas que realizan unas actividades.
- Department Divisiones: entidad a más alto nivel de departamentos, anida uno o varios “Departament Unit”, representa un grupo de departamentos (Generalmente con mismo objetivo o función relacionada).
- Jefe Departamental: perfil de usuario, coordinador de uno o varios departamentos, responsable de las actividades realizadas, encargado de realizar estimaciones a nivel departamental.
- Jefe Proyectos: perfil de usuario, coordinador de un determinado Proyecto (Producto), encargado de ratificar las estimaciones realizadas por un jefe de departamento para un determinado Proyecto (Producto).
- Log4Net: librería licenciada bajo la Apache Software Licence, que nos permite incluir en la aplicación trazas o log en nuestra base de datos.
- mainLoadPro: nombre elegido para la aplicación en contexto, deferente a una carga principal profesional de trabajo.
- mainLoadPro.api: proyecto de librería C# API para realización de consultas externas de los datos expuestos por la herramienta. Futura fuente de implementación de tareas IA sobre las cargas y estimaciones de horas.
- mainLoadPro.resources: proyecto C# librería de recursos usables desde distintos proyectos y que identifican un entorno concreto de trabajo, idioma o parámetros específicos de uso.
- mainLoadPro.bll: proyecto C# de librería perteneciente a la capa de negocio, ofrece funcionalidades y acceso a datos a la capa de diseño, en este caso mainLoadPro.mvc
- mainLoadPro.data: proyecto accesos a datos C#, implementado con ORM de conexión a bases de datos MySQL con EntityFramework6, contiene las entidades generadas mediante “Data First” como modelo de datos para uso por el resto de aplicación.
- mainLoadPro.mvc: proyecto principal ASP.Net MVC, conforma la capa de diseño y presentación al usuario, se conecta con librerías de negocio y datos para ofrecer contenido navegable al usuario, alta capacidad de escalabilidad y reusabilidad debido a los componentes HTML diseñados, permite el uso de lenguaje RazorC# en su contenido para ofrecer funcionalidades de servidor

en su proceso de visualización dinámico de datos, implementando el uso de framework como JQuery y Bootstrap para mejorar la experiencia de usuario.

- Miro: plataforma en línea de pizarras de colaboración que permite a equipos distribuidos trabajar juntos de manera eficaz.
- OBS: (Organization Breakdown Structure) conjunto de entidades que conforman la jerarquía de departamentos de una organización.
- Parrilla: Componente principal de la aplicación, conformado por varias vistas que trabajan de manera asíncrona para obtener una vista interfaz común de usuario. En esta se muestras, las anualidades, los departamentos, los proyectos y las opciones de validación y editado de los datos por parte de los usuarios, prácticamente conformado por un oneViewPage que permite la interacción con los datos desde una única pantalla de usuario.
- Planificador: perfil de usuario, responsable del producto Aplicación, encargado de realizar una estimación inicial de las horas de los proyectos por departamento y responsable de coordinar las actividades de estimaciones entre jefes de Departamento y Jefes de Proyecto, en una versión avanzada de los datos, consolidará la información en base a una congelación general de datos, para posterior explotación.
- Proyectos: unidad atómica de un trabajo o tarea a ser realizar por un departamento, puede corresponder con un producto o con una parte de este. (Estudio de cargas en la suspensión del amortiguador, evolutivo de motores, ensayos de aislamiento de chasis)
- Proyectos Área: agrupador de proyectos, en base a su área de trabajo o naturaleza cohesionadora de varias partes de producto para un único producto. (Suspensión, Motor, Chasis)
- Proyectos Área Super: agrupador de áreas de producto, consolidadora de un área específica que agrupa los esfuerzos de realización de varios productos en un proyecto. (automóvil).
- Proyectos Finanza: entidad de gestión sobre los aspectos financieros del producto, generalmente asociados a una cierta legalidad correspondiente de la región o país en el que residen los departamentos que realizan los trabajos específicos.
- Proyectos Negocio: agrupador de Proyectos áreas super en una entidad suprema que rige la línea de negocio de la compañía y en la que se enmarcan los desarrollos. (Renault Motor, Renault Sistemas Financieros, Renault Renting).
- Sedes: entidad relativa a la localización geográfica de la que dependen los departamentos, en organización multinacionales, muy importante la identificación, por las diferentes casuísticas financieras y fiscales de cada región.
- Versión Congelada: correspondiente con un grupo de datos consolidado que permite el bloqueo de los datos para posterior explotación de estos para generación de informes y estadísticas.
- WBS: (Work Breakdown Structure) conjunto de entidades que conforman la estructura de proyectos existente en una compañía, dividida por líneas de negocio o productos.
- WorkFlow: Flujo de trabajo redirigido entre los actores de la aplicación para conformar una versión congelada del producto, basada en las validaciones y ratificaciones de datos.

8. Bibliografía y referencias técnicas

[1] **ASP.Net MVC**: https://learn.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/getting-started
[2] **lenguaje C#**.: https://dotnet.microsoft.com/es-es/languages/csharp
[3] **Visual Studio Community**: https://apps.microsoft.com/store/detail/visual-studio-community-2019/XP8CDJNZKFM06W
[4] ***Modelo Vista Controlador (MVC)***: *https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/*
[5] **API web C#**: https://learn.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api
[6] **REST**: https://learn.microsoft.com/en-us/aspnet/web-api/overview/older-versions/build-restful-apis-with-aspnet-web-api
[7] **JQuery / Ajax**: https://api.jquery.com/category/ajax/
[8] **Bootstrap**: https://getbootstrap.com/docs/4.6/getting-started/introduction/
[9] **EntityFramework 6**: https://learn.microsoft.com/en-us/ef/ef6/
[10] **ORM EF6**: https://www.entityframeworktutorial.net/what-is-entityframework.aspx
[11] **MySQL 8.0.20**: https://dev.mysql.com/doc/relnotes/mysql/8.0/en/news-8-0-20.html
[12] **Oracle MySQL**: https://www.mysql.com/
[13] **Innodb**: https://dev.mysql.com/doc/refman/8.0/en/innodb-storage-engine.html
[14] **Gantt Project**: https://www.ganttproject.biz/
[15] **Kanban**: https://miro.com/agile/kanban/
[16] **MariaDB**: https://blog.devart.com/mysql-vs-mariadb.html
[17] **Azure**: https://portal.azure.com/#home
[18] **Gantt**: https://www.teamgantt.com/what-is-a-gantt-chart
[19] **Sprint**: https://www.scrum.org/resources/what-is-a-sprint-in-scrum
[20] **GanttProject**: https://www.ganttproject.biz/
[21] **CRUD**: https://learn.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/implementing-basic-crud-functionality-with-the-entity-framework-in-asp-net-mvc-application
[22] **Arquitectura**: https://learn.microsoft.com/es-es/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures
[23] **FrontEnd**: https://www.sitepoint.com/premium/books/front-end-development-with-asp-net-core-angular-and-bootstrap/read/1/k36msnfm/
[24] BackEnd
[25] **WebAPI**: https://learn.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api
[26] **Swagger-Net**: https://www.nuget.org/packages/Swagger-Net
[27] Miro: https://miro.com/welcomeonboard/SHA2Y1ZDZUJzdzlIdjF5aHBrMExHYkQ1YUFDdTdZUnhsSVlqTmtQbTV0bzd0dDRkQjlSMldwVDVaN2NmTTJNUXwzNDU4NzY0NTM1Njg4ODQzNzYxfDI=?share_link_id=773923718530
[28] Miro: https://miro.com/welcomeonboard/SHA2Y1ZDZUJzdzlIdjF5aHBrMExHYkQ1YU

FDdTdZUnhsSVlqTmtQbTV0bzd0dDRkQjlSMldwVDVaN2NmTTJNUXwzNDU4NzY0NTM1Njg4ODQzNzYxfDI=?share_link_id=773923718530

[29] Particionado de tablas MySQL: http://mundogeek.net/archivos/2012/03/09/particiones-en-mysql/

[30] **Procedimientos almacenados**: https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html

[31] **Log4Net**: https://stackify.com/log4net-guide-dotnet-logging/

[32] "Database First": https://learn.microsoft.com/en-us/ef/ef6/modeling/designer/workflows/database-first

[33] **Balanceo carga MySQL**: https://blogofsysadmins.com/replicacion-y-ditribucion-de-carga-de-mysql

[34] **Debian**: https://hub.docker.com/_/mysql

[35] ***MySQL WorkBench**: https://www.mysql.com/products/workbench/*

[36] ***Microsoft Visual Studio Community 2019**:* https://apps.microsoft.com/store/detail/visual-studio-community-2019/XP8CDJNZKFM06W

[37] **.Net Framework 4.7.2**: https://dotnet.microsoft.com/en-us/download/dotnet-framework/net472

[38] ***.Net Framework Core**: https://dotnet.microsoft.com/en-us/download*

[39] **MySQL Conector de BBDD**: https://www.mysql.com/products/connector/

[40] **Pluggin MySQL for VisualStudio**: https://downloads.mysql.com/archives/visualstudio/

[41] **Microsoft.Net.Http 2.2.29**: https://learn.microsoft.com/en-us/answers/questions/1147105/microsoft-bcl-package-dependency-issue

[42] ***Swagger-Net**:* https://www.nuget.org/packages/Swagger-Net

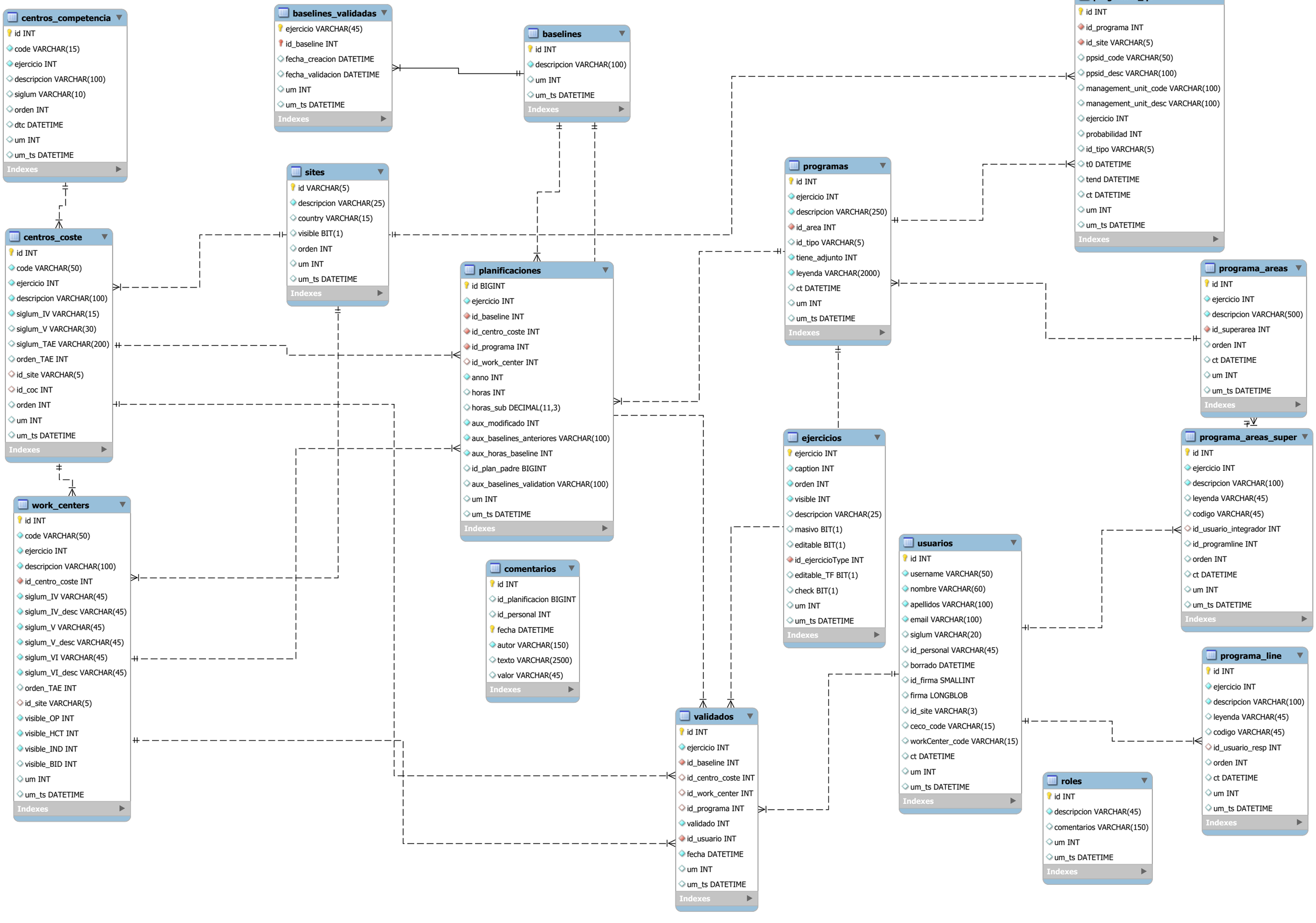
[44] ***System.Web.Security FormAuthenticationTicket:*** *https://learn.microsoft.com/en-us/dotnet/api/system.web.security.formsauthentication?view=netframework-4.8.1*

[45] ***Helpers**: https://learn.microsoft.com/en-us/aspnet/web-pages/overview/ui-layouts-and-themes/creating-and-using-a-helper-in-an-aspnet-web-pages-site*

Nota: todos los accesos web han tenido lugar entre febrero 2023 y Junio 2023, pueden contener posibles cambios o extinciones de uso en fechas futuras.

9. Anexos

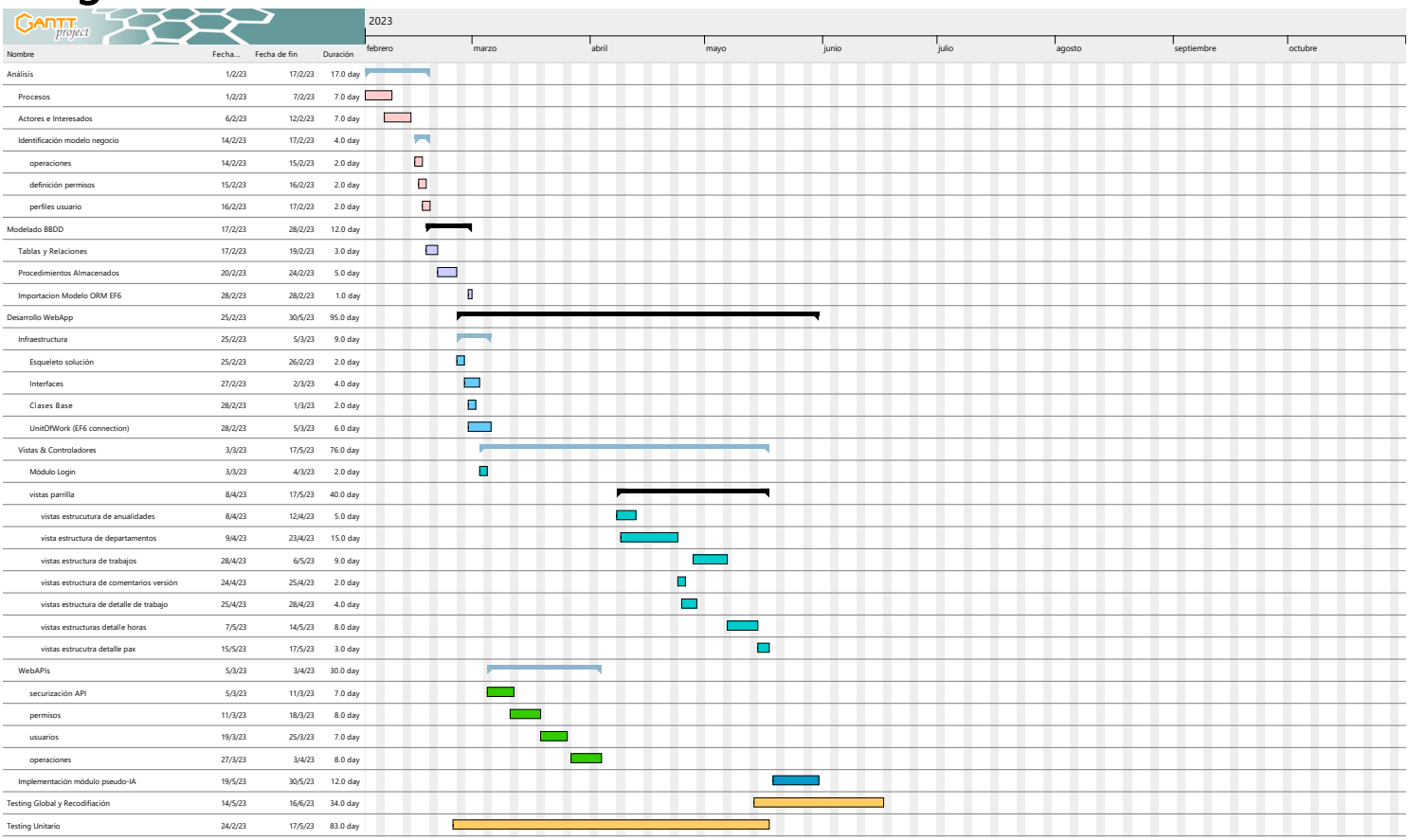
En las siguientes páginas del documento se encuentran anexadas documentaciones relativa al modelo de BBDD, y al informe de proyecto realizado con GanttProject, han sido incluidos como anexos al proyecto para poder evaluar con más detalle los mismos debido a la pérdida de calidad de estos en sus correspondientes apartados del presente documento.


centros_competencia
id INT
code VARCHAR(15)
ejercicio INT
descripcion VARCHAR(100)
siglum VARCHAR(10)
orden INT
dtc DATETIME
um INT
um_ts DATETIME
Indexes
baselines_validadas
ejercicio VARCHAR(45)
id_baseline INT
fecha_creacion DATETIME
fecha_validacion DATETIME
um INT
um_ts DATETIME
Indexes
baselines
id INT
descripcion VARCHAR(100)
um INT
um_ts DATETIME
Indexes
programas_q5
id INT
id_programa INT
id_site VARCHAR(5)
ppsid_code VARCHAR(50)
ppsid_desc VARCHAR(100)
management_unit_code VARCHAR(100)
management_unit_desc VARCHAR(100)
ejercicio INT
probabilidad INT
id_tipo VARCHAR(5)
t0 DATETIME
tend DATETIME
ct DATETIME
um INT
um_ts DATETIME
Indexes
sites
id VARCHAR(5)
descripcion VARCHAR(25)
country VARCHAR(15)
visible BIT(1)
orden INT
um INT
um_ts DATETIME
Indexes
programas
id INT
ejercicio INT
descripcion VARCHAR(250)
id_area INT
id_tipo VARCHAR(5)
tiene_adjunto INT
leyenda VARCHAR(2000)
ct DATETIME
um INT
um_ts DATETIME
Indexes
programa_areas
id INT
ejercicio INT
descripcion VARCHAR(500)
id_superarea INT
orden INT
ct DATETIME
um INT
um_ts DATETIME
Indexes
centros_coste
id INT
code VARCHAR(50)
ejercicio INT
descripcion VARCHAR(100)
siglum_IV VARCHAR(15)
siglum_V VARCHAR(30)
siglum_TAE VARCHAR(200)
orden_TAE INT
id_site VARCHAR(5)
id_coc INT
orden INT
um INT
um_ts DATETIME
Indexes
planificaciones
id BIGINT
ejercicio INT
id_baseline INT
id_centro_coste INT
id_programa INT
id_work_center INT
anno INT
horas INT
horas_sub DECIMAL(11,3)
aux_modificado INT
aux_baselines_anteriores VARCHAR(100)
aux_horas_baseline INT
id_plan_padre BIGINT
aux_baselines_validation VARCHAR(100)
um INT
um_ts DATETIME
Indexes
ejercicios
ejercicio INT
caption INT
orden INT
visible INT
descripcion VARCHAR(25)
masivo BIT(1)
editable BIT(1)
id_ejercicioType INT
editable_TF BIT(1)
check BIT(1)
um INT
um_ts DATETIME
Indexes
programa_areas_super
id INT
ejercicio INT
descripcion VARCHAR(100)
leyenda VARCHAR(45)
codigo VARCHAR(45)
id_usuario_integrador INT
id_programline INT
orden INT
ct DATETIME
um INT
um_ts DATETIME
Indexes
work_centers
id INT
code VARCHAR(50)
ejercicio INT
descripcion VARCHAR(100)
id_centro_coste INT
siglum_IV VARCHAR(45)
siglum_IV_desc VARCHAR(45)
siglum_V VARCHAR(45)
siglum_V_desc VARCHAR(45)
siglum_VI VARCHAR(45)
siglum_VI_desc VARCHAR(45)
orden_TAE INT
id_site VARCHAR(5)
visible_OP INT
visible_HCT INT
visible_IND INT
visible_BID INT
um INT
um_ts DATETIME
Indexes
comentarios
id INT
id_planificacion BIGINT
id_personal INT
fecha DATETIME
autor VARCHAR(150)
texto VARCHAR(2500)
valor VARCHAR(45)
Indexes
validados
id INT
ejercicio INT
id_baseline INT
id_centro_coste INT
id_work_center INT
id_programa INT
validado INT
id_usuario INT
fecha DATETIME
um INT
um_ts DATETIME
Indexes
usuarios
id INT
username VARCHAR(50)
nombre VARCHAR(60)
apellidos VARCHAR(100)
email VARCHAR(100)
siglum VARCHAR(20)
id_personal VARCHAR(45)
borrado DATETIME
id_firma SMALLINT
firma LONGBLOB
id_site VARCHAR(3)
ceco_code VARCHAR(15)
workCenter_code VARCHAR(15)
ct DATETIME
um INT
um_ts DATETIME
Indexes
programa_line
id INT
ejercicio INT
descripcion VARCHAR(100)
leyenda VARCHAR(45)
codigo VARCHAR(45)
id_usuario_resp INT
orden INT
ct DATETIME
um INT
um_ts DATETIME
Indexes
roles
id INT
descripcion VARCHAR(45)
comentarios VARCHAR(150)
um INT
um_ts DATETIME
Indexes

Tarea

Nombre	Fecha de inicio	Fecha de fin	Duración
Análisis	1/2/23	17/2/23	17
Procesos	1/2/23	7/2/23	7
Actores e Interesados	6/2/23	12/2/23	7
Identificación modelo negocio	14/2/23	17/2/23	4
operaciones	14/2/23	15/2/23	2
definición permisos	15/2/23	16/2/23	2
perfiles usuario	16/2/23	17/2/23	2
Modelado BBDD	17/2/23	28/2/23	12
Tablas y Relaciones	17/2/23	19/2/23	3
Procedimientos Almacenados	20/2/23	24/2/23	5
Importacion Modelo ORM EF6	28/2/23	28/2/23	1
Desarrollo WebApp	25/2/23	30/5/23	95
Infraestructura	25/2/23	5/3/23	9
Esqueleto solución	25/2/23	26/2/23	2
Interfaces	27/2/23	2/3/23	4
Clases Base	28/2/23	1/3/23	2
UnitOfWork (EF6 connection)	28/2/23	5/3/23	6
Vistas & Controladores	3/3/23	17/5/23	76
Módulo Login	3/3/23	4/3/23	2
vistas parrilla	8/4/23	17/5/23	40
vistas estrucutura de anualidades	8/4/23	12/4/23	5
vista estructura de departamentos	9/4/23	23/4/23	15
vistas estructura de trabajos	28/4/23	6/5/23	9
vistas estructura de comentarios versión	24/4/23	25/4/23	2
vistas estructura de detalle de trabajo	25/4/23	28/4/23	4
vistas estructuras detalle horas	7/5/23	14/5/23	8
vistas estrucutra detalle pax	15/5/23	17/5/23	3
WebAPIs	5/3/23	3/4/23	30
securización API	5/3/23	11/3/23	7
permisos	11/3/23	18/3/23	8
usuarios	19/3/23	25/3/23	7
operaciones	27/3/23	3/4/23	8
Implementación módulo pseudo-IA	19/5/23	30/5/23	12
Testing Global y Recodifiación	14/5/23	16/6/23	34
Testing Unitario	24/2/23	17/5/23	83

Diagrama de Gantt



Nombre	Fecha...	Fecha de fin	Duración
Análisis	1/2/23	17/2/23	17.0 day
Procesos	1/2/23	7/2/23	7.0 day
Actores e Interesados	6/2/23	12/2/23	7.0 day
Identificación modelo negocio	14/2/23	17/2/23	4.0 day
operaciones	14/2/23	15/2/23	2.0 day
definición permisos	15/2/23	16/2/23	2.0 day
perfiles usuario	16/2/23	17/2/23	2.0 day
Modelado BBDD	17/2/23	28/2/23	12.0 day
Tablas y Relaciones	17/2/23	19/2/23	3.0 day
Procedimientos Almacenados	20/2/23	24/2/23	5.0 day
Importacion Modelo ORM EF6	28/2/23	28/2/23	1.0 day
Desarrollo WebApp	25/2/23	30/5/23	95.0 day
Infraestructura	25/2/23	5/3/23	9.0 day
Esqueleto solución	25/2/23	26/2/23	2.0 day
Interfaces	27/2/23	2/3/23	4.0 day
Clases Base	28/2/23	1/3/23	2.0 day
UnitOfWork (EF6 connection)	28/2/23	5/3/23	6.0 day
Vistas & Controladores	3/3/23	17/5/23	76.0 day
Módulo Login	3/3/23	4/3/23	2.0 day
vistas parrilla	8/4/23	17/5/23	40.0 day
vistas estrucutura de anualidades	8/4/23	12/4/23	5.0 day
vista estructura de departamentos	9/4/23	23/4/23	15.0 day
vistas estructura de trabajos	28/4/23	6/5/23	9.0 day
vistas estructura de comentarios versión	24/4/23	25/4/23	2.0 day
vistas estructura de detalle de trabajo	25/4/23	28/4/23	4.0 day
vistas estructuras detalle horas	7/5/23	14/5/23	8.0 day
vistas estrucutra detalle pax	15/5/23	17/5/23	3.0 day
WebAPIs	5/3/23	3/4/23	30.0 day
securización API	5/3/23	11/3/23	7.0 day
permisos	11/3/23	18/3/23	8.0 day
usuarios	19/3/23	25/3/23	7.0 day
operaciones	27/3/23	3/4/23	8.0 day
Implementación módulo pseudo-IA	19/5/23	30/5/23	12.0 day
Testing Global y Recodifiación	14/5/23	16/6/23	34.0 day
Testing Unitario	24/2/23	17/5/23	83.0 day