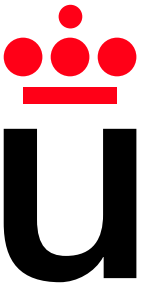

Universidad
Rey Juan Carlos

Escuela Técnica Superior
de Ingeniería Informática

Grado en Ingeniería del Software

Curso 2022-2023

Trabajo Fin de Grado

INTERFAZ WEB PARA GRÁFICOS DE SERIES TEMPORALES ASOCIADAS A DATOS OCEANOGRÁFICOS

Autor: Óscar Ballesteros Izquierdo
Tutor: Manuel Rubio Sánchez

Agradecimientos

A las primeras personas que quiero agradecer este trabajo es a mi familia, que son quienes más me han apoyado para que pueda estudiar y dedicarme a lo que me gusta desde que era pequeño, la informática.

También tengo que agradecer a todos mis compañeros de Medio Físico de Puertos del Estado, he estado cuatro años trabajando con ellos y desde un primer momento siempre me han dato total confianza en los proyectos que he planteado, gracias a ellos he crecido como profesional y siete años después sigo trabajando en aplicaciones relacionadas con la oceanografía, en concreto me gustaría mencionar a las personas que más me han ayudado explicándome los conceptos que necesito para desarrollar este trabajo: Enrique Álvarez, Susana Pérez, Begoña Pérez y José María García-Valdecasas.

Por último, agradecer a mi profesor del TFG, Manuel Rubio Sánchez, que me ha ayudado mucho a sacar unas memorias con la calidad necesaria, y a la Universidad Rey Juan Carlos por todos los conocimientos que he aprendido a lo largo de estos años, también, gracias al programa de prácticas encontré el puesto en Puertos del Estado que tanto me ha ayudado a encauzar mi carrera profesional.

Resumen

El fin del proyecto es desarrollar una interfaz web completamente interactiva y adaptable a cualquier dispositivo, a través de la cual los usuarios puedan ver varios tipos de representaciones de datos oceanográficos, tales como series de tiempo, mapas de calor o tablas especiales.

El primer paso es la recolección de los casos de uso y objetivos que el cliente espera de la aplicación, como pueden ser los distintos elementos interactivos, el tipo de interfaz esperada o los requisitos de carga y rendimiento. Después se estudian todas las posibles tecnologías a implementar y se justifican las elecciones tomadas. Para el frontend se buscan las librerías más importantes de representación gráfica. En la parte backend se investigan los frameworks para el desarrollo de servidores web que se ajustan mejor a las características de este proyecto. Por último, se aborda la forma de implementar estas tecnologías y la puesta en producción para los usuarios finales.

Este proyecto es público y los gráficos son accesibles para todo el mundo a través de la aplicación Portus [1]. Para facilitar el acceso a éstos, al final de este documento se añadirá un apéndice con varios enlaces a distintos gráficos, en los que se podrán ver de forma rápida todas las características más importantes de la aplicación desarrollada.

La descripción del desarrollo de la aplicación se ha dividido en dos trabajos de fin de grado (Ingeniería del Software e Ingeniería Informática). Esta memoria, relacionada con el Grado de Ingeniería del Software, se centrará en la parte frontend de PortusData.

Índice de contenidos

1 Introducción

Este TFG es fruto del trabajo realizado para Puertos del Estado [2], donde además realicé las prácticas externas, en concreto, llevé a cabo el mantenimiento, desarrollo y evolución de aplicaciones web enfocadas a predicciones oceanográficas, la mayoría de ellas se integran en el sistema Portus de acceso público.

El sistema Portus está compuesto de varios servicios desarrollados por distintas personas y empresas. Este trabajo va a tratar uno de los más importantes, PortusData, que ha sido desarrollado completamente por mí.

Tengo el permiso de Puertos del Estado para compartir el código y poder mostrarlo como trabajo final de carrera.

1.1. Alcance

El alcance de este proyecto es poder presentar los datos que tenemos de predicciones y tiempo real de forma gráfica, sencilla e interactiva para el público general, y que a su vez estos gráficos sean precisos, funcionales y tengan algunas herramientas para usuarios más avanzados, hay mucha gente que trabaja en los puertos de España y parte su trabajo depende de estos datos.

Para ello, partimos de una base de datos muy grande y un servidor REST ya existente que sirve los datos de esta, el objetivo es crear una aplicación web que sea capaz de representarlos de varias formas específicas y un pequeño backend que organice de manera eficiente los metadatos de estas y se encargue del routing de la aplicación.

Estos datos se pueden representar de varias formas: series temporales, tablas especiales, gráficos de área, mapas de calor, etc. Cada variable puede tener algunas características específicas para lograr la representación más óptima posible.

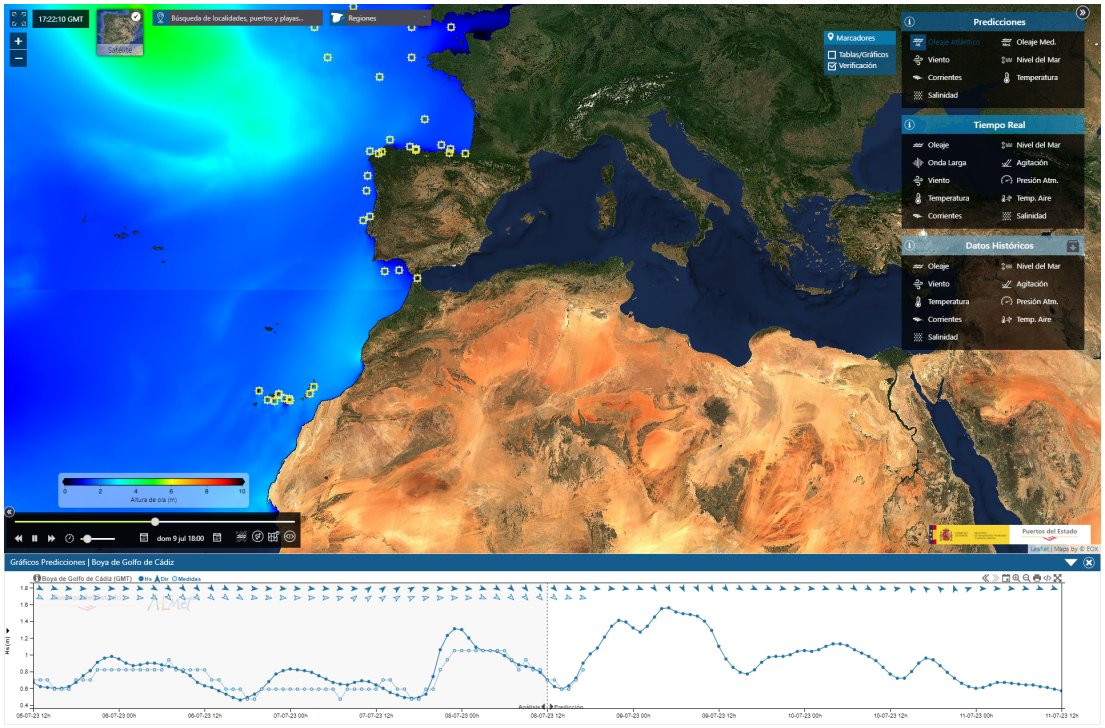

Figura 1.1: Chart de PortusData embebido en Portus.

1.2. Contexto

La idea de este proyecto surge de una necesidad de modernización de los servicios web de Puertos del Estado, partimos de un sistema Portus que se trataba de una sola aplicación que integraba todos los servicios en ella, tales como charts, animaciones, mapas, tablas, reportes en PDF y más. Este sistema tenía dos grandes problemas:

- **Tecnología anticuada:**
  Los dos frameworks principales eran una versión antigua de Spring MVC [3] (con Java 7) y GWT [4], que son perfectamente usables, pero cada vez más en desuso ya que la comunidad ha ido migrando a otras tecnologías más modernas. Esto afectaba mucho a los futuros evolutivos que se querían hacer ya que limitaba bastante las opciones en cuanto a librerías, actualizaciones y seguridad.
- **Complejidad del proyecto:**
  El tener tantos servicios distintos integrados en la misma aplicación generaba muchos problemas, por ejemplo, el generador de reportes en PDF a veces llenaba la memoria del servidor y esto hacía que tampoco se pudiera acceder ni a los gráficos ni a los mapas. También, por la aplicación han pasado distintas empresas y desarrolladores, y a lo largo del tiempo la estructura del proyecto se ha ido haciendo innecesariamente compleja, con mucho código antiguo o duplicado y con partes del proyecto que generaban problemas de compilación.

Figura 1.2: Portus antiguo (Tecnología anticuada, todos los servicios en una app).

Debido a estos problemas, se decidió que para los futuros evolutivos del pro-

yecto la mejor opción era reescribir todo el sistema Portus, pero esta vez usando librerías modernas y con buena proyección y siguiendo un esquema de microservicios en el que se separen las aplicaciones con distinto uso, con un web service central que se encargue de nutrirlas de los datos necesarios.

La integración de este nuevo esquema ha traído bastantes cosas beneficiosas para los servicios web:

- **Código con más funcionalidad y menos complejidad:**
  El reescribir de cero todo y la modernización de los frameworks nos ha permitido aplicar los nuevos estándares web que han ido surgiendo, y esto se refleja mucho en la calidad y menor complejidad del código final.
- **Los servicios se mantienen en pie:**
  Antes era un gran problema ya que los servicios se pasaban caídos una parte muy importante del tiempo, la modernización de librerías ha ayudado, pero lo que más ha influido en esto es la separación en microaplicaciones, ahora si surge un problema en una de poca importancia, esto no se propaga hacia las principales.
- **Mejora en el mantenimiento:**
  Al ser aplicaciones separadas, ahora distintas empresas o desarrolladores pueden trabajar en una de ellas solo y especializarse en esta, lo que ha hecho mucho más eficiente la forma de interactuar entre terceros.
- **Mayores posibilidades de evolución:**
  En los sistemas de información geográfica se trabaja con muchos estándares que evolucionan de forma muy rápida, el tener frameworks con una gran comunidad nos ayuda a poder implementarlos de forma mucho más rápida y sencilla que antes.

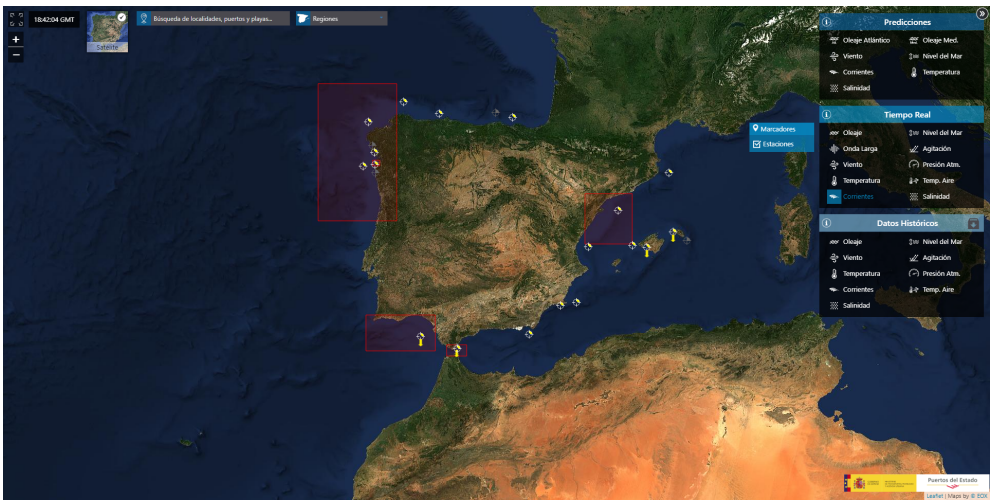

Figura 1.3: Portus nuevo (Tecnología moderna, arquitectura de microservicios).

2

Objetivos

Este proyecto trata de uno de los nuevos servicios más importantes: los charts. Partimos de una versión que estaba integrada con todo el sistema Portus antiguo, usaban una librería para GWT bastante antigua y con pocas capacidades y posibilidades de interacción, solo sirve para representar series temporales en formatos muy concretos.

Los nuevos charts queremos que sean un proyecto individual mucho más potente que las gráficas originales, que represente los datos de varios formatos distintos y no solo series de tiempo, que sean mucho más interactivos y fáciles de manejar para los usuarios y que puedan ser usados no solo en Portus, sino en otras aplicaciones que ofrecen e incluso el público pueda usarlos como widgets en su página web.

Tras la aprobación del nuevo proyecto, paso a recoger los datos del cliente sobre los charts actuales, qué cosas se quieren mejorar de ellos y qué nuevas características esperan de PortusData.

Las características más importantes que se recogen del proyecto para este frontend son:

- Varios tipos de representaciones gráficas (series de tiempo, mapas de calor, bandas de confianza, etc)
- Algunas representaciones con formatos muy específicos.
- Interactividad en todos los elementos y capacidad de configuración por parte del usuario.
- Que se puedan servir tanto en formato web interactivo como en imagen.

- Adaptables a cualquier tamaño de interfaz.
- Integración de controles táctiles para su funcionamiento en dispositivos móviles.

3

Descripción Informática

3.1. Análisis

3.1.1. Requisitos funcionales

Los requisitos funcionales son declaraciones detalladas que describen las acciones y funciones que la aplicación debe ser capaz de realizar para satisfacer las necesidades del usuario o del cliente.

Los relativos al frontend de PortusData son los siguientes:

- **Interacción total:**
  Todos los elementos deben ser interactivos en la medida de lo posible.
- **Personalizar vista:**
  Se permitirán mostrar u ocultar los elementos del gráfico.
- **Acceso a metainformación:**
  El usuario a parte de la serie temporal de datos, podrá ver otro tipo de información relacionada con el código de modelo o estación.
- **Cambiar fechas:**
  El chart cargará las fechas más recientes, pero el usuario podrá navegar por todos los datos disponibles.
- **Compartir:**
  Cualquier persona podrá mostrar el gráfico en su página web.
- **Logos:**
  Se mostrarán los logos para que sean reconocibles los proveedores de datos.

- **Cambios de referencia o unidades:**
  El usuario podrá cambiar algunas propiedades del gráfico interactuando con los elementos.
- **Imprimir:**
  Se permitirá imprimir el gráfico de una manera personalizada para que cuadre mejor dentro de un folio.

3.1.2. Requisitos no funcionales

Los requisitos no funcionales son las restricciones no impuestas al sistema, se centran en cómo se deben hacer las cosas y están relacionados con la calidad del software final. Será necesario cumplirlos todos para garantizar la satisfacción del cliente y los usuarios.

Para esta parte del proyecto son los siguientes:

- **Visualización óptima:**
  Cuando el usuario se desplaza por el gráfico, es el chart el que calcula los nuevos dominios de manera automática.
- **Fiabilidad:**
  Es importante que la representación gráfica sea del todo fiable y no cometa errores pintando las series temporales ya que estos gráficos sirven para validar otro tipo de modelos y herramientas.
- **Lenguaje de desarrollo:**
  Para el frontend se usará Javascript y HTML, incluyendo D3.js [5] como librería principal para gestionar los datos.
- **Acceso a datos:**
  La petición de los datos se hará a través de un web service existente llamado SIMO.
- **Compatibilidad:**
  Entre los posibles usuarios todavía hay gente usando sistemas operativos antiguos, por lo que la mayor parte de la funcionalidad debe ser compatible con estos.
- **Rendimiento:**
  El gráfico debe de tener un tiempo de carga lo más rápido posible y debe de limitarse la cantidad de datos a pintar para evitar problemas de memoria.

3.1.3. Casos de uso

Los casos de uso sirven para capturar y describir las interacciones entre el sistema y los actores, en este caso el usuario.

Representan una forma de entender y documentar toda la funcionalidad que tendrá la aplicación, centrándose en las tareas específicas que serán llevadas a cabo por los actores.

El diagrama de casos de uso es una representación gráfica de todas estas interacciones y funcionalidades, de forma esquemática, sin entrar en los detalles internos de su representación:

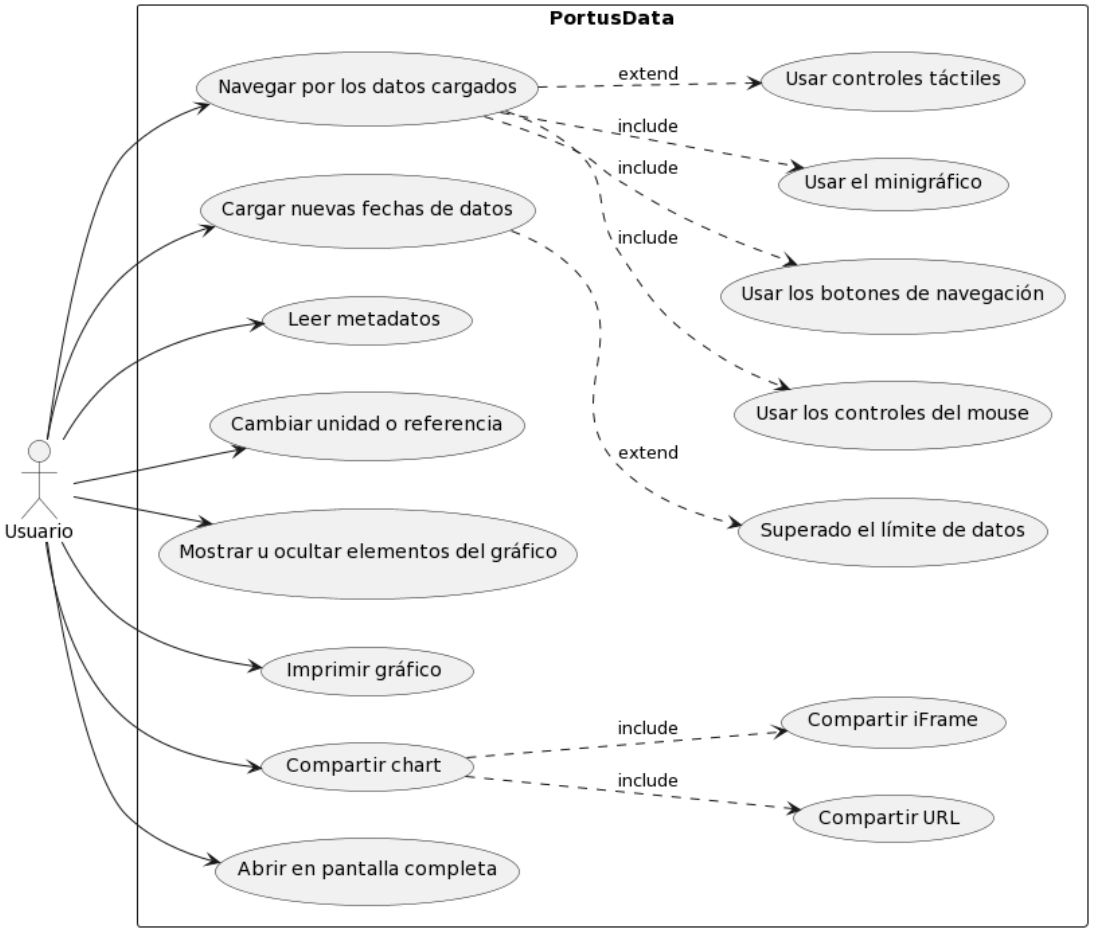

Figura 3.1: Diagrama de casos de uso en el frontend de la aplicación.

A continuación, se muestran las tablas de casos de uso, que es otra forma de representarlos en la que las interacciones se describen con más detalle:

Caso de uso	Navegar por los datos cargados
Actores	Usuario
Descripción	El usuario podrá navegar de forma interactiva por el gráfico y ajustar la vista según desee
Precondición	Esperar carga del chart
Escenario	El usuario podrá moverse por los datos de varias formas
Postcondición	El gráfico se desplazará usando animaciones para que el usuario no se pierda en la transición
Extensión	▪ Usar controles táctiles
Inclusión	▪ Usar el minigráfico ▪ Usar los controles del mouse ▪ Usar los botones de navegación

Caso de uso	Usar controles táctiles
Actores	Usuario
Descripción	Si se accede al gráfico desde un móvil, los controles táctiles se activarán automáticamente
Precondición	Cargar el chart desde un dispositivo táctil
Escenario	Se podrán usar tanto gestos multitáctiles como pulsando los botones con el dedo
Postcondición	El gráfico se desplazará usando animaciones para que el usuario no se pierda en la transición
Extensión	-
Inclusión	-

Caso de uso	Usar los controles del mouse
Actores	Usuario
Descripción	Los charts serán manejables interactuando con el propio gráfico
Precondición	Esperar carga del chart
Escenario	▪ Click izquierdo y arrastrar para dar zoom en el eje X. ▪ Click derecho y arrastrar para moverse atrás o adelante en el tiempo. ▪ Doble click para mostrar la serie completa.
Postcondición	El gráfico se desplazará usando animaciones para que el usuario no se pierda en la transición
Extensión	-
Inclusión	-

Caso de uso	Usar los botones de navegación
Actores	Usuario
Descripción	El usuario se podrá desplazar por el gráfico usando los botones de la interfaz
Precondición	Esperar carga del chart
Escenario	El usuario hará click en los botones de izquierda/derecha y aumentar/-disminuir zoom para desplazarse por la serie de tiempo
Postcondición	El gráfico se desplazará usando animaciones para que el usuario no se pierda en la transición
Extensión	-
Inclusión	-

Caso de uso	Usar el minigráfico
Actores	Usuario
Descripción	Habrá un gráfico auxiliar pequeño en el que se podrá seleccionar la fecha inicial y final a visualizar.
Precondición	Pulsar el botón para abrirlo
Escenario	El usuario verá la serie de tiempo completa en miniatura, podrá seleccionar de forma rápida el rango temporal nuevo que desee
Postcondición	El gráfico se desplazará usando animaciones para que el usuario no se pierda en la transición
Extensión	-
Inclusión	-

Caso de uso	Cargar nuevas fechas de datos
Actores	Usuario
Descripción	Se podrán cargar datos más antiguo a los iniciales
Precondición	Abrir los calendarios
Escenario	El usuario podrá desplazarse por los calendarios en todas las fechas en las que haya datos, tendrá que fijar una fecha inicial y otra final
Postcondición	Volverá a salir el spinner de carga y se pintará la nueva serie temporal.
Extensión	Superado el límite de datos
Inclusión	-

Caso de uso	Superado el límite de datos
Actores	Usuario
Descripción	Hay un límite máximo en la cantidad de días que se pueden cargar en el chart
Precondición	Abrir los calendarios
Escenario	Si el usuario elige unas fechas que superan el límite, se le informará al usuario de este y no se cargarán nuevos datos.
Postcondición	-
Extensión	-
Inclusión	-

Caso de uso	Leer metadatos
Actores	Usuario
Descripción	Al usuario se le presentará información extra sobre el punto de modelo o estación.
Precondición	Pultar el botón de información
Escenario	Se abre un popup con estos metadatos en forma de tabla y recursos de imagen si hay disponibles.
Postcondición	-
Extensión	-
Inclusión	-

Caso de uso	Cambiar unidad o referencia
Actores	Usuario
Descripción	El usuario podrá cambiar ciertos parámetros que modifican los valores del gráfico
Precondición	Esperar carga del chart
Escenario	Al pulsar sobre la unidad o sobre la referencia se abre un seleccionable con las nuevas posibles opciones. El usuario deberá hacer otro click sobre el ítem deseado
Postcondición	Se actualizan los valores del gráfico
Extensión	-
Inclusión	-

Caso de uso	Mostrar u ocultar elementos del gráfico
Actores	Usuario
Descripción	Las distintas variables se podrán mostrar u ocultar de forma interactiva
Precondición	Esperar carga del chart
Escenario	El usuario hará click sobre los elementos de la leyenda, si este es visible se ocultará, y si ya estaba ocultado se mostrará de nuevo
Postcondición	La variable dejará de ser visible en el gráfico. Es posible que algún eje cambie su dominio para adaptarse a la nueva vista.
Extensión	-
Inclusión	-

Caso de uso	Imprimir gráfico
Actores	Usuario
Descripción	El gráfico podrá imprimirse en formato físico
Precondición	Pulsar el botón de la impresora
Escenario	Algunos elementos del gráfico se ocultan o mueven para conseguir una impresión óptima
Postcondición	Una vez se sale del menú de impresión, todos los elementos vuelven a su estado original
Extensión	-
Inclusión	-

Caso de uso	Compartir chart
Actores	Usuario
Descripción	Los charts pueden ser exportados para que cualquier persona los integre en su página web
Precondición	Pulsar el botón de compartir
Escenario	Se abrirá un popup con diversas opciones para compartir el chart
Postcondición	-
Extensión	-
Inclusión	▪ Compartir URL ▪ Compartir iFrame

Caso de uso	Compartir URL
Actores	Usuario
Descripción	Se ofrece la url del gráfico con todos los parámetros con los que se ha llamado.
Precondición	Pulsar el botón de compartir
Escenario	El usuario podrá copiar la url del chart y compartirla donde desee
Postcondición	-
Extensión	-
Inclusión	-

Caso de uso	Compartir iFrame
Actores	Usuario
Descripción	Se ofrece el código html necesario para incrustar el gráfico en una web
Precondición	Pulsar el botón de compartir
Escenario	El usuario podrá copiar el código para insertarlo en su web de forma rápida
Postcondición	-
Extensión	-
Inclusión	-

Caso de uso	Abrir en pantalla completa
Actores	Usuario
Descripción	Los charts podrán abrirse en una ventana nueva que utilice el máximo posible de la pantalla
Precondición	Pulsar el botón de pantalla completa
Escenario	Se abrirá una nueva ventana con el mismo gráfico, esta ventana aprovechará todo el ancho de la pantalla y ajustará el alto para que tenga unas proporciones óptimas para su visualización
Postcondición	-
Extensión	-
Inclusión	-

3.2. Especificación

Partimos de una versión de los charts muy básica, consistía en una librería que solamente pintaba una nube de puntos a partir de unos datos de series temporales, previamente formateados para el uso con esta librería.

Problemas o carencias de esta versión:

- En niveles de zoom bajos, los puntos se amontonan y hace difícil la comprensión del gráfico.
- La interactividad es mínima, se reduce a las acciones de los botones y el uso del ratón para seleccionar un área al que dar zoom.
- El uso de los puntos sin una línea que una toda la serie complica entender la escala temporal cuando los valores son dispersos.
- Al seleccionar un área con el ratón y moverse por el gráfico, se permite hacer tanto en el eje X como el Y, más adelante se explicará por qué es más eficiente restringirlo a solo la dimensión temporal.
- La interfaz desaprovecha mucho espacio e impide que se puedan visualizar de manera eficiente cuando se insertan en contenedores pequeños.
- El título del gráfico se puede configurar por un parámetro en la URL y estos están preparados para que la gente los comparta en su web, cualquiera podría cambiar el título por un mensaje ofensivo y esto saldría con el logo del cliente.

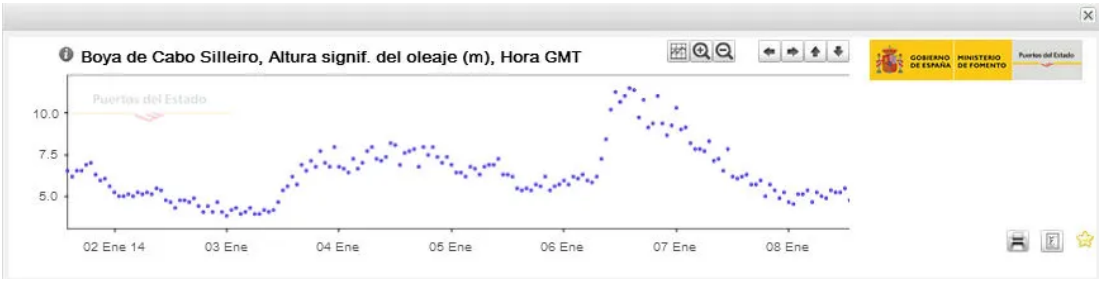

Figura 3.2: Charts antiguos.

La mayoría de estos problemas son causa de las limitaciones de la librería que se usa para representar los datos, dadas las características del nuevo proyecto comentadas anteriormente, se concluye que es inviable cumplirlas siguiendo usando la misma, por lo que el nuevo proyecto no puede ser una evolución del anterior y se decide reescribir todo el sistema desde cero, usando herramientas más modernas, separándolo de la aplicación principal en la que estaba integrado y definiendo desde el principio todas las características actuales y futuras evoluciones que se plantean en el proyecto.

Una vez analizados todos los casos de uso con el cliente, el siguiente paso del

proyecto es elegir las nuevas librerías con las que se va a trabajar, tras analizar todas las opciones para representar series temporales, destacan dos tipos de librerías:

- **Librerías de charting:**
  Están preparadas para funcionar de forma rápida y sencilla, se les pasan los datos y estas se encargan de representarlos en varios formatos predeterminados (series de tiempo, gráficos de barras...), suelen permitir opciones para customizar los estilos y las más completas cuentan con sistemas de plugins para aumentar la funcionalidad de estos o añadir nuevos tipos de gráficos. Ejemplos: Charts.js, Highcharts.

- **Librerías de representación gráfica:**
  Están preparadas para representar cualquier tipo de gráficos como series temporales, mapas, 3d. Tienen funciones que ayudan tanto a gestionar los datos como a imprimirlos por pantalla usando estándares web. Ejemplo: D3.js, webGL.

El proyecto se podría cumplir perfectamente con cualquiera de estos dos tipos de librerías, pero dado que queremos que sea un proyecto con muchas posibilidades de evolución en el futuro, es importante elegir la tecnología más adecuada, para ello hago una tabla comparativa con los elementos más importantes:

Elemento	Librerías charting	Librerías gráficas
Datos	Se les proporciona en un formato predeterminado y la librería los gestiona.	El formato de los datos es libre y cuentan con funciones que ayudan a la gestión de estos.
Tipos de gráficos	Limitados, algunas librerías incluyen más que otras. Extensible por medio de plugins.	Ilimitados. Cualquier representación es posible con este tipo de librerías.
Complejidad	Sencillas de usar, suelen seguir estándares para el tratamiento y representación de los datos.	Más complejas de usar ya que no solo hay que gestionar los datos sino que hay que crear toda la interfaz del usuario.
Interactividad	Definida por la librería. Ampliable mediante plugins, pero con limitaciones.	Ilimitada, se tiene el control de todos los elementos en pantalla y por tanto se puede hacer cualquier tipo de conexión entre ellos.
Controles táctiles	Las librerías más completas lo gestionan ellas mismas.	El desarrollador tiene que implementarlos por su cuenta.
Evolutivos	Aumentar la funcionalidad mediante plugins te hace depender de una librería gestionada por un tercero.	Al tener más control sobre el código de la aplicación hay pocas limitaciones a la hora de incorporar nuevas funcionalidades.

Una vez hecha esta comparación se deduce que las librerías de charting son más sencillas y rápidas para cumplir con el proyecto, pero las librerías gráficas ofrecen muchas más posibilidades y libertad para programar. En este caso, como ya existe una aplicación que da un servicio parecido, el tiempo no es un problema, por lo que se decide elegir la segunda opción para tener todas las posibilidades que nos ofrecen las librerías gráficas y las garantías de tener una aplicación propia de representación gráfica.

Ahora, dentro de este tipo de librerías toca escoger la más correcta para empezar a escribir el código de la nueva aplicación, entre todas las opciones posibles, hay tres frameworks que destacan claramente sobre el resto:

- **Canvas:**
  Es una API de HTML5 que permite dibujar gráficos y animaciones en la página web mediante JavaScript. Se usa principalmente para representaciones sencillas.
- **D3.js:**
  Es una biblioteca de JavaScript que permite crear visualizaciones de datos interactivas y dinámicas usando el estándar SVG. Permite altas opciones de personalización y se usa tanto para visualizaciones sencillas como para las más complejas.
- **WebGL:**
  WebGL es una API de gráficos para navegadores web que utiliza la GPU

del equipo. Está diseñada para representaciones en 3D pero también puede utilizarse para gráficos en 2D. Es la librería más potente de las tres pero a su vez la más compleja.

Una vez analizadas las opciones, la primera que se descarta es Canvas debido a sus limitaciones y que queremos hacer representaciones tanto sencillas como complejas. Entre las otras dos que quedan, WebGL es la más potente, pero requiere un alto nivel de conocimiento y eso limitaría las posibilidades a la hora de encontrar futuros desarrolladores para evolucionar el proyecto, además, ni está pensado en los futuros evolutivos el tener representaciones tridimensionales.

Finalmente, se elige D3.js ya que es la librería perfecta para hacer esta aplicación: se basa en SVG que está aceptado por todos los navegadores modernos, tiene una curva de aprendizaje media, nos permite todas las posibilidades de interacción entre elementos que necesitamos y tiene una gran comunidad de desarrollo detrás de ella, lo que nos garantiza que tendrá mejoras y mantenimiento durante muchos años.

3.3. Diseño

Para el diseño de esta aplicación, hay unos requisitos fundamentales pedidos por el cliente:

- **Integración con Portus:**
  La mayoría de las visualizaciones de estos gráficos vendrán de la aplicación principal, por lo que el diseño tendrá que ser amigable con el de esta app y a su vez sencillo para que se integre bien con el resto de páginas web y aplicaciones.

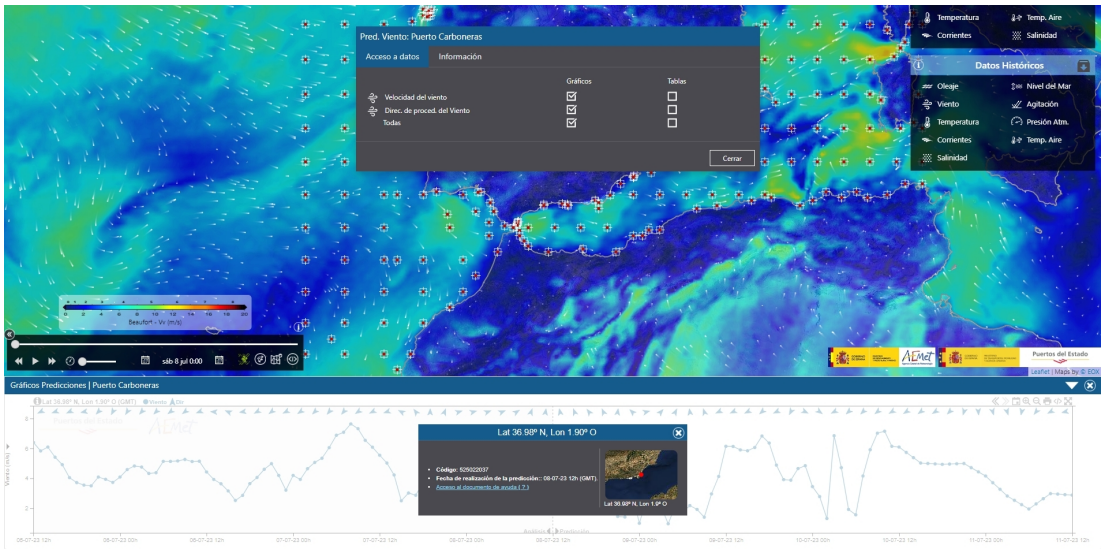

Figura 3.3: Los charts y Portus son aplicaciones independientes, pero comparten estilos para conseguir una buena integración.

- **Interfaz única y reconocible:**
  Dado que estos charts son públicos y cualquiera puede insertar nuestra información en su web, se requiere una interfaz en la cual se reconozca fácilmente la procedencia de estos datos, para ello, se diseñan los gráficos desde cero y se asegura que los logos sean siempre visibles.

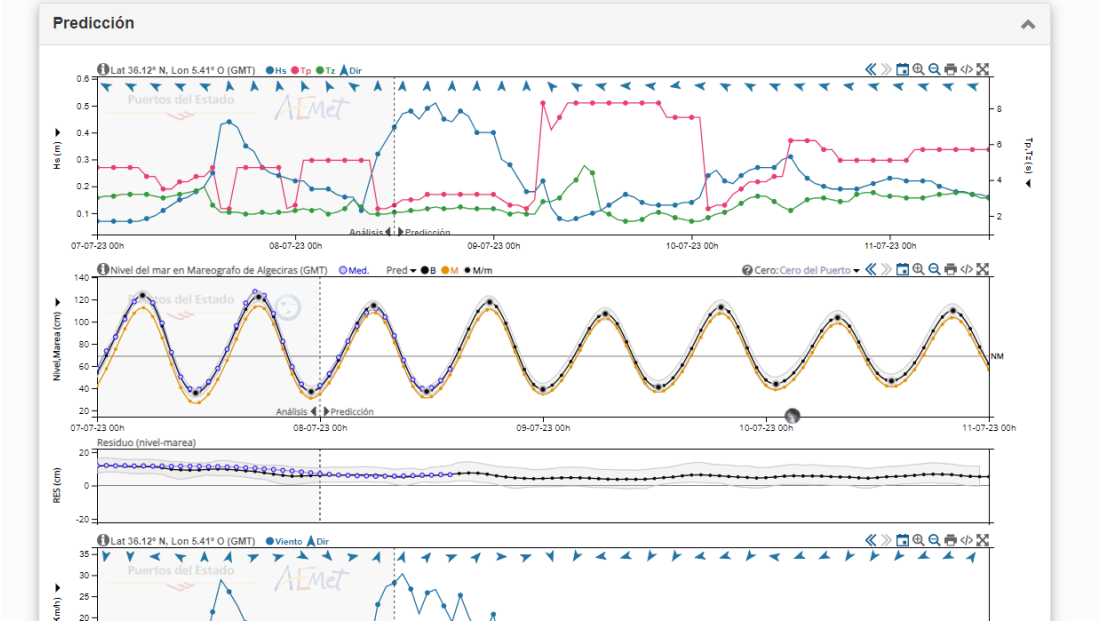

Figura 3.4: Varios tipos de charts embebidos juntos en otra aplicación.

- **Aprovechamiento del espacio:**
  En muchos casos, los usuarios pueden ver varios gráficos a la vez, por lo que es muy importante que la visualización sea siempre la óptima, para ello se han hecho varias mejoras para que el gráfico tenga el mayor tamaño posible sin afectar al resto de elementos como los botones y los logos.

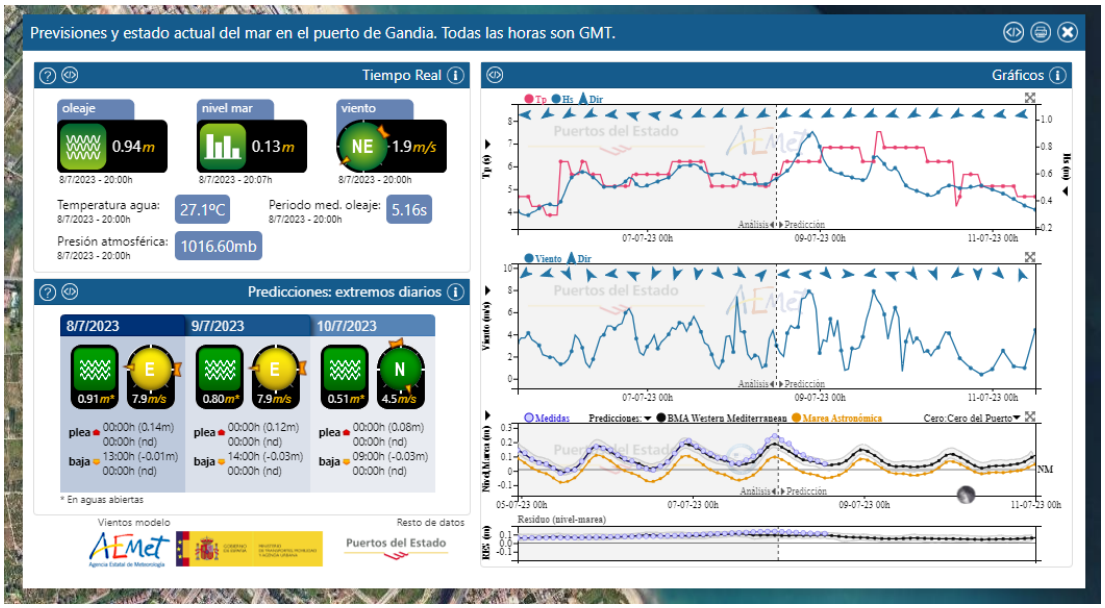

Figura 3.5: Varios charts en un espacio pequeño de pantalla.

- **Visualización en cualquier tamaño:**
  Estos gráficos están pensados para poder verse en cualquier tipo de pantalla o contenedor, para ello se han diseñado tres tipos de interfaces comunes a todos los tipos de gráficos:
  - **Default:** Se usa cuando el contenedor tiene un ancho mayor a 1000px, tiene márgenes definidos y un tamaño de botones y leyenda agradables a la vista cuando hay espacio suficiente en la pantalla.

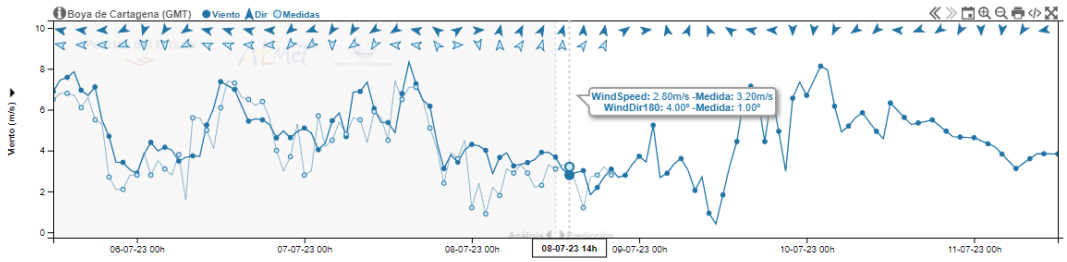

Figura 3.6: Interfaz default.

- **Mínima:** Cuando el tamaño del contenedor es menor a 1000px, se le reducen los márgenes y el tamaño del resto de elementos para darle más importancia al gráfico.

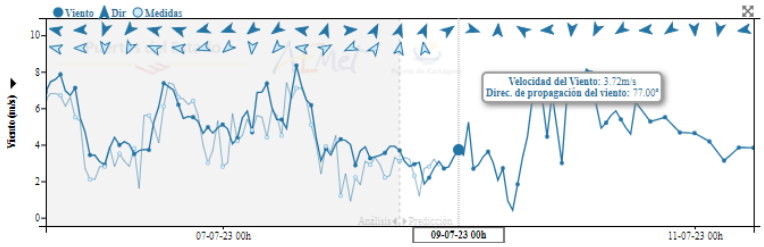

Figura 3.7: Interfaz mínima.

- **Previsualización:** Es un tipo de interfaz para un tamaño ya muy pequeño, como puede ser en una aplicación móvil o un popup pequeño de información rápida, carece de interactividad y botones ya que su función es hacer de enlace para ver el gráfico en una ventana nueva con la interfaz default.

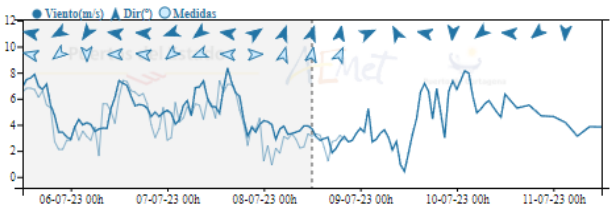

Figura 3.8: Interfaz de previsualización.

- **Uso en móviles:**
  Se requiere que los gráficos puedan ser usados en cualquier tipo de dispositivo, para ello, una vez definidos los controles con el ratón, se han adaptado a controles táctiles/multitáctiles para que puedan ser usados tanto en navegadores como en aplicaciones de móvil y tablet.

Figura 3.9: Charts integrados en aplicación móvil iMar.

- **Imprimibles:**
  A muchos usuarios todavía les gusta tener los datos para analizarlos en formato físico, para ello, se añade un botón de imprimir a la interfaz. Antes de la ventana para configurar la impresión, se esconden los botones del gráfico y se mueven los logos fuera para que se vean mejor. Cuando se

cierra el diálogo de impresión, todos los elementos de la interfaz vuelven a su estado original.

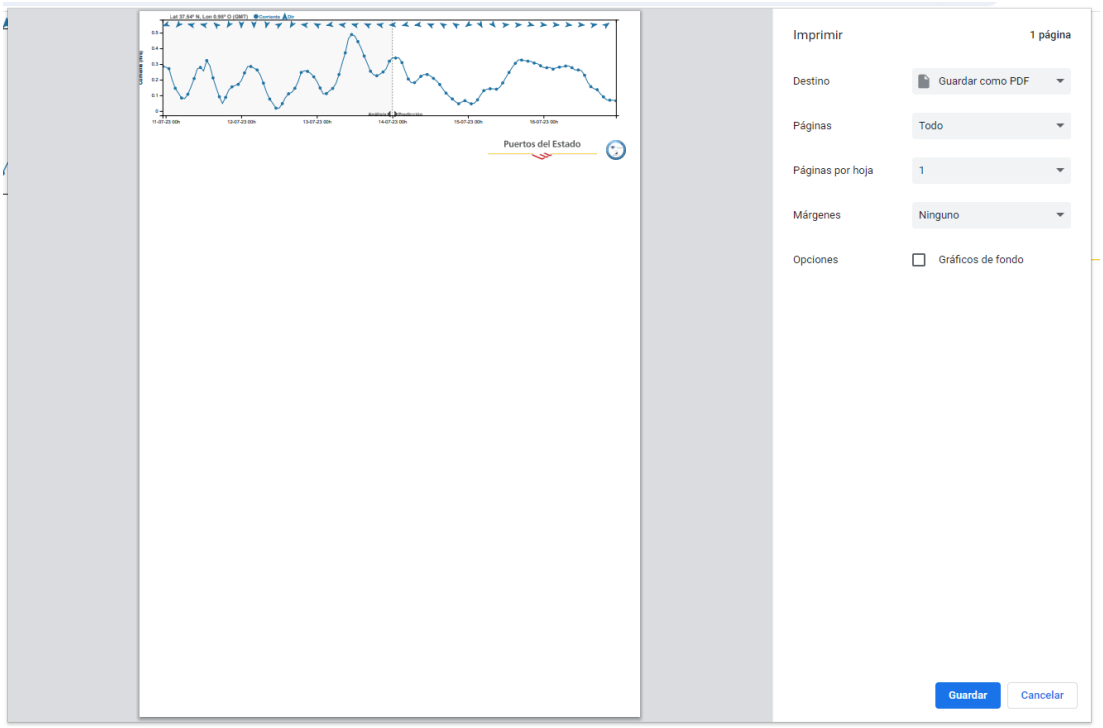

Figura 3.10: Los charts se ajustan para una impresión óptima.

3.3.1. Colores

Una parte muy importante de la representación gráfica es la correcta elección de los colores y escalas, una mala elección de estos podría romper la visualización del gráfico e incluso hacer imposible la comprensión de este. Para este proyecto es muy importante que los colores tengan las siguientes características:

- Cada serie temporal tiene que estar coloreada de tal forma que se distinga bien de los colores grises y blancos del fondo.
- Los distintos colores que se usen en un mismo gráfico tienen que ser distinguibles entre sí sin forzar la vista.
- Tienen que ser amigables con la paleta de colores de Portus.
- Cada variable deberá de tener siempre el mismo color, con esto se consigue que los usuarios se acostumbren a ellos y reconozcan la información del gráfico lo más rápido posible.

En este caso, es el cliente quien ha elegido la tonalidad exacta final de los colores ya que existen más aplicaciones que dan los datos de las mismas variables, y es importante para ellos que estos sean coherentes en todos sus sistemas de representación de datos.

A continuación, se explica cómo se han integrado estas características en cada tipo de chart:

- **Charts de predicción y tiempo real:**
  En estos dos tipos de gráficos el algoritmo para elegir los colores es muy parecido, ambos muestran las mismas variables y por ello los colores tienen que ser los mismos en ambos.
  En total hay más de 20 variables distintas que se pueden representar, pero la mayoría de ellas no se van a visualizar nunca juntas por la forma de agruparse según al modelo que correspondan. Si usaramos un color por cada variable, sería muy difícil distinguirlos a simple vista, por ello se ha ideado un sistema por el cual con solo 4 colores se pueden representar todas las variables de estos gráficos:
  - Por cada modelo, se separan las variables de dirección y las variables que irán en forma de serie temporal.
  - Se ordenan en cada uno de los grupos estas variables por orden de importancia, de más consultadas a menos consultadas.
  - En cada grupo de variables, a la principal se le da un tono azul, la segunda será un rosa intenso, la tercera de color verde y la última será morada.
  - De momento no hay ningún modelo que pueda agrupar más de 4 variables en cada grupo, por lo que no se necesitan más, si en un futuro se quieren añadir otras, habría que buscar un quinto color que quede bien junto al resto.
  - En el caso de las medidas, los puntos y flechas tendrán un borde y

un relleno, el relleno será una tonalidad más clara que el color de la variable.

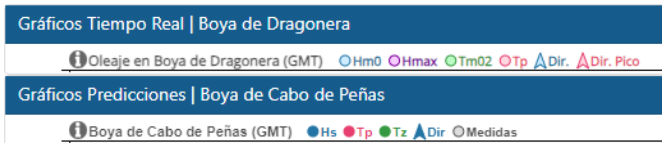

Figura 3.11: Posibles colores en los gráficos de tiempo real y predicción.

Existe una excepción en los gráficos de tiempo real, cuando se está comparando la misma variable en distintas estaciones no quedaría bien darles a todas las series el mismo color, y por ello, en este caso se eligen colores al azar, aunque para evitar visualizaciones incorrectas, el algoritmo intenta que estos siempre estén lo más alejado posible en la escala cromática.

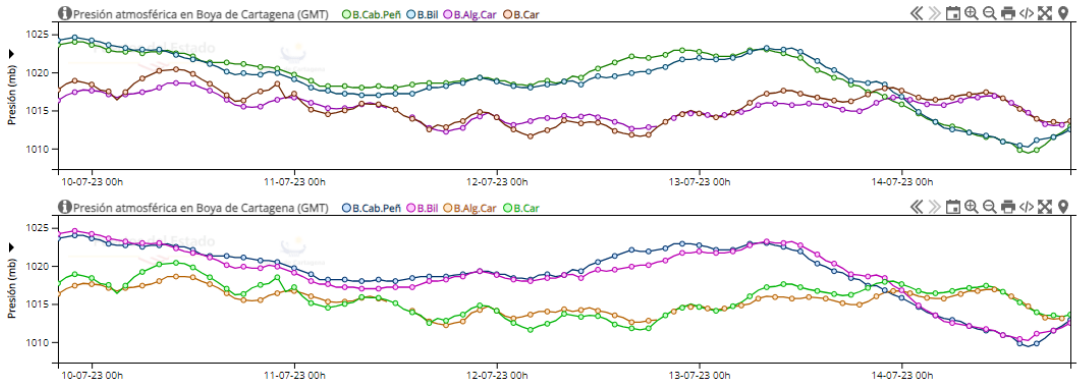

Figura 3.12: Distintas comparaciones de estaciones darán distintos colores al azar.

- **Charts de nivel del mar:**
  En este gráfico hay variables que tienen predicciones de distintos modelos, por lo que no podemos darle a todos el mismo color:
  - La marea astronómica siempre va en naranja.
  - Solo hay medidas de nivel, que irán en azul.
  - Cada modelo de predicción tendrá siempre el mismo color para ser reconocible.
  - Solo puede haber un modelo con banda de confianza disponible, siempre irán en negro.
  - Solo se muestran los máximos y mínimos de la serie más importante, irán en el mismo color, pero para distinguirse de las medidas, el borde será el que tenga la tonalidad clara.
  - El gráfico anidado muestra también distintos modelos, y los colores coincidirán con los del principal.

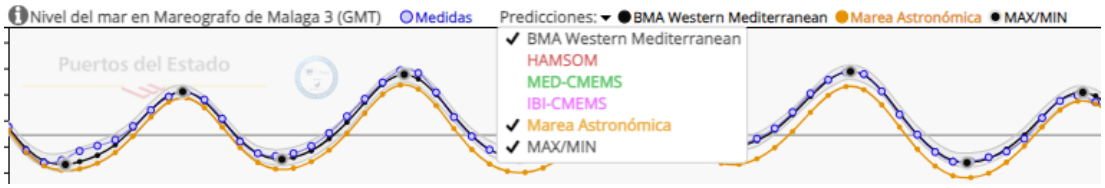

Figura 3.13: Cada modelo que mide el nivel tiene un color reconocible.

- **Charts de marea astronómica:**
  Este es un tipo de gráfico muy simple y no tiene muchos colores, sus características son:
  - La marea astronómica siempre va en naranja para ser coherente con las de los anteriores gráficos.
  - Los máximos y los mínimos se pintan en verde.
  - Para que el gráfico no quede demasiado simple, se pinta el área de este en una tonalidad azul.

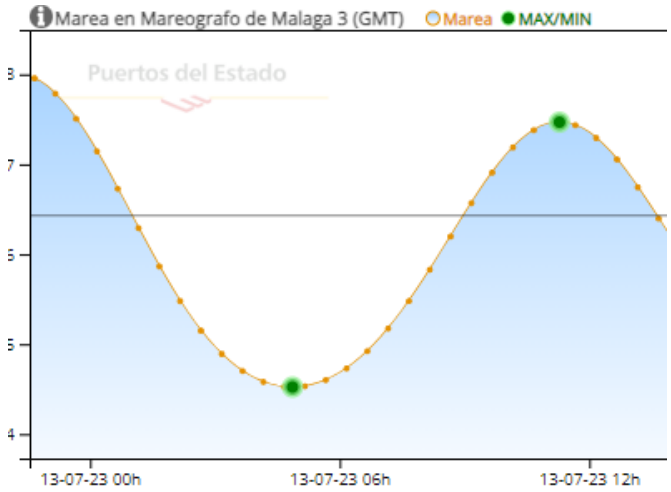

Figura 3.14: Colores de marea astronómica.

- **Tablas de mareas**
  En este caso no se pinta ningun dato de forma gráfica, son simplemente tablas y no necesitamos distintas tonalidades de color, por eso, los colores finales son los mismos que la aplicación que muestra estas tablas, para así conseugir una mayor integración con esta.

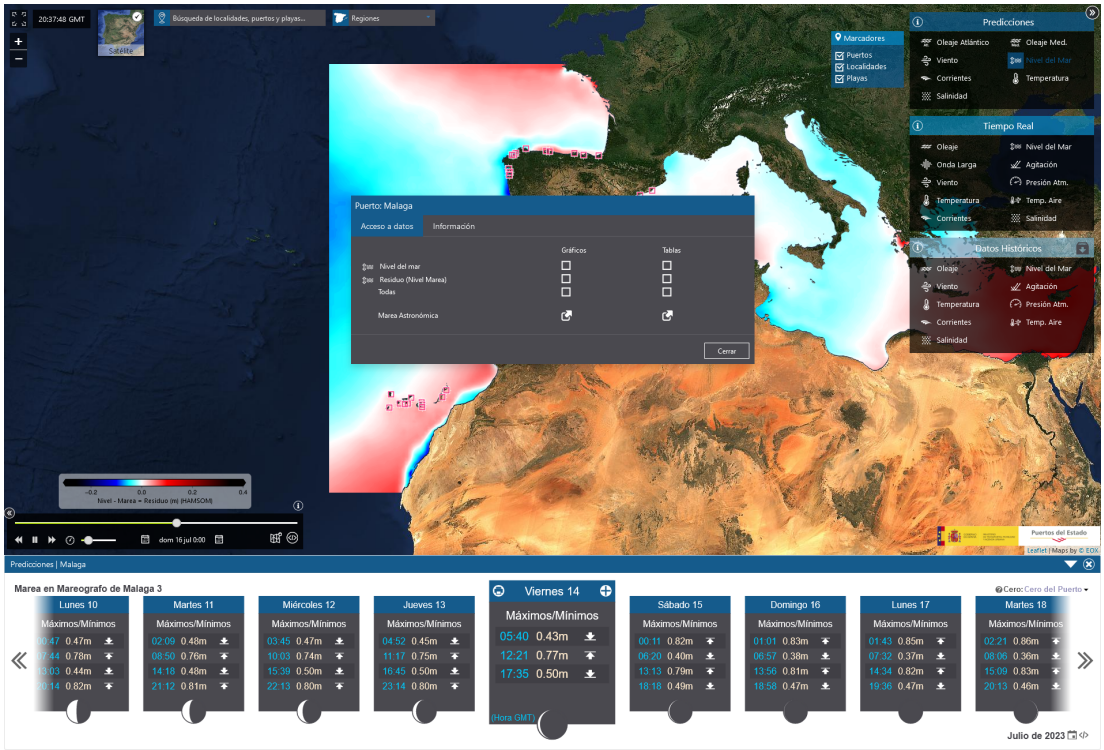

Figura 3.15: Las tablas de mareas usan los mismos colores que Portus.

- **Charts de perfiladores:**
  En este tipo de gráfico los colores son todavía más importantes, en este caso no se eligen tonalidades individuales, sino una escala continua que pueda representar todos los datos del mapa de calor que se genera.
  La serie de tiempo anidada siempre irá en el mismo color azul principal que los otros charts.
  Los perfiladores miden dos variables muy distintas entre sí y cada una de ellas tiene una escala diferente para su óptima visualización:
  - **Velocidad de la corriente:** En esta variable el mínimo valor posible es 0 y el máximo es teóricamente infinito, pero es muy raro encontrar ya un valor por encima de los 150cm/s, y por ello, si se encuentra algún valor mayor, se pintará del mismo color.
    Para conseguir el efecto del mapa de calor y lograr unos valores que se distingan de la forma más cómoda posible hay que jugar con las propiedades de los colores. Si se elige una escala de grises o de un solo color, los valores cercanos son muy difíciles de distinguir entre sí, ya que lo único que los diferencia es la luminancia del color. Para mejorar esto, se hace una escala que tenga en cuenta tanto la luminancia como la tonalidad de los colores.
    El valor 0 será un color con luminosidad baja y un tono frío, lo que da como resultado un azul oscuro, progresivamente, la escala irá cambian-

do la tonalidad a colores cálidos y de luminosidad alta, en la mitad de esta nos encontramos un color amarillo muy intenso, finalmente, la tonalidad sigue cambiando a colores más cálidios pero vuelve a tener una luminosidad baja, dando como resultado un color rojo oscuro para los valores más altos. Gracias a esto, se pueden identificar los valores a los que corresponde cada color de una forma bastante fácil sin tener que usar el tooltip para conocer el valor exacto.
Finalmente, esta escala se desplaza hacia los valores más comunes, aunque el límite de la escala es 150, lo más normal es que las medidas tengan valores entre 0 y 30 cm/s, por este motivo, en este rango los cambios de tonalidad y luminosidad son más pronunciados, permitiendo distinguir todavía mejor estos valores más comunes.

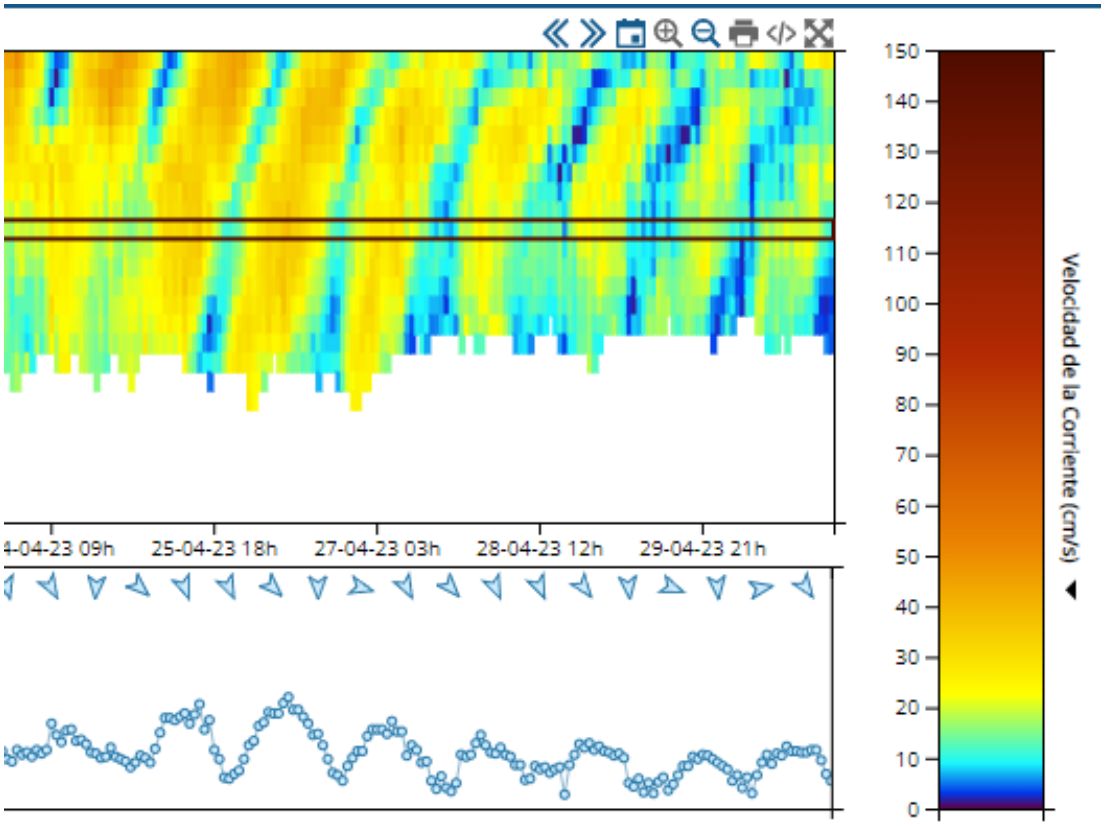

Figura 3.16: La escala de velocidad de la corriente tiene mayor información de colores en los valores más comunes.

- **Dirección de la corriente:** La dirección es una variable muy distinta al resto de las que se miden, usa una escala circular en la que los valores están entre el 0 y el 359, siendo ambos límites un valor muy cercano entre ellos por la naturaleza de estas escalas. Para representar eso, se puede usar el círculo cromático RGB, que consiste en una escala

circular en la que se muestran todas las posibles tonalidades de los colores, comenzando por el rojo, progresivamente irá cambiando a verde, luego al azul y terminará en el mismo rojo con el que ha empezado.

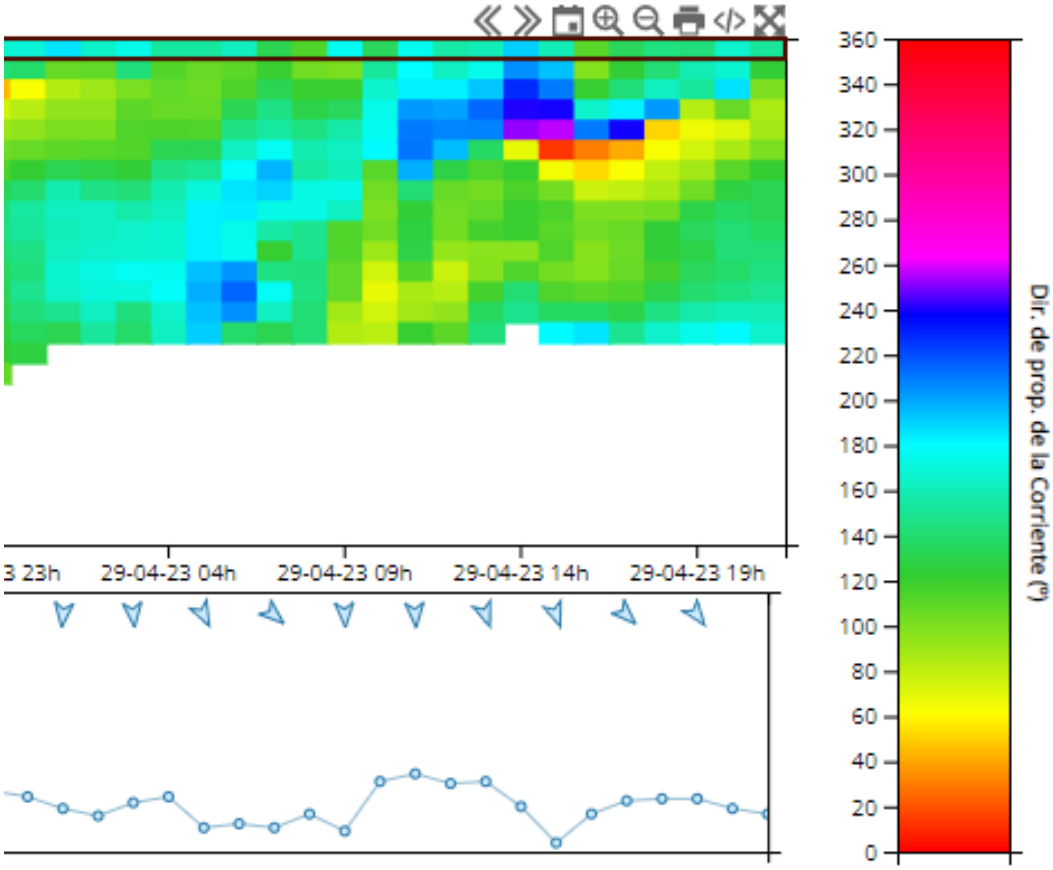

Figura 3.17: El círculo cromático permite una representación óptima de las variables de dirección.

3.4. Implementación

Para comenzar la implementación del frontend del proyecto, hay que tener claro todos los tipos de gráficos distintos que se van a representar, para ello, entre todas las representaciones que hay que hacer, se dividen en grupos en función de las características comunes que tienen, como el formato de los datos de entrada, las variables comunes, la distinta funcionalidad que se necesita... Al final se decide agrupar los charts en 6 tipos distintos:

- **Series temporales:**
  Charts de predicción (predChart), charts de tiempo real: (rtChart), charts de nivel del mar (nivmarChart) y charts de marea astronómica (astroCharts)
- **Mapas de calor:**
  Mapa de perfiladores (ADCPChart)
- **Tablas especiales:**
  Tablas de mareas (tablasMareas)

Cada tipo de chart tendrá una funcionalidad distinta al resto y su propio archivo con código JavaScript. Más adelante se explicarán las diferencias entre cada uno de estos.

Aunque cada chart funciona de una forma distinta, para optimizar el proyecto y la futura creación de nuevos tipos, se ha hecho que tengan muchas características básicas en común:

- **Interfaz:**
  Todos los charts deberán tener una interfaz parecida y reconocible, estos archivos gestionan elementos como los márgenes, el tamaño del texto, los botones... (InterfazDefault, InterfazMin, InterfazPrev).
- **Controles:**
  La forma de manejar los distintos gráficos deberá ser lo más parecida posible para evitar confusiones del usuario, estos archivos manejan los eventos del ratón y sus equivalentes para dispositivos táctiles. (ControlesDefault, ControlesMovil)
- **Funciones auxiliares:**
  Resto de funciones que son comunes entre cualquier tipo de gráfico, como pueden ser los formateos de fecha, la gestión de umbrales, los cálculos de dominio, tooltip, unidades, etc. (CommonChart).

Cada gráfico tendrá que incorporar la misma interfaz y controles, también podrá hacer uso de las funciones auxiliares que son comunes entre ellos, pero tendrá la libertad de sobrescribir cualquier funcionalidad o elemento de la interfaz de estos archivos para adaptarla a sus características individuales.

3.4.1. Charts de Predicción

Son el tipo de chart más simple, pero el más usado. Se trata de series de tiempo de datos horarios de muchos modelos: oleaje, viento, nivel del mar, corrientes, temperatura y salinidad.

Cada uno de estos modelos tiene varios parámetros a representar en formato de serie temporal. En cada modelo se pueden agrupar los distintos parámetros en el mismo gráfico, los que comparten misma unidad de medida irán al mismo eje, con la posibilidad de representar dos ejes distintos: uno en el lado izquierdo y otro en el derecho.

Los parámetros de dirección se pueden representar tanto como una serie temporal como con flechas en la parte superior del gráfico, por lo que realmente se pueden mostrar hasta tres unidades distintas en un mismo gráfico.

Por último, existe la posibilidad de comparar los datos de los modelos de predicción con estaciones cercanas que midan los mismos parámetros, esto sirve para validar la calidad de estas predicciones.

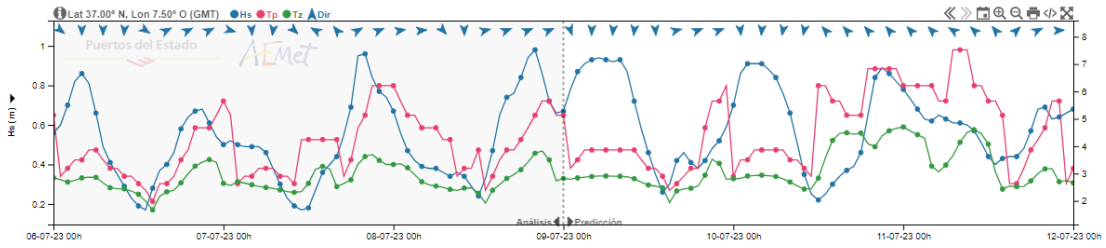

Figura 3.18: Chart de predicción que combina varios parámetros de oleaje en dos ejes distintos con flechas de dirección.

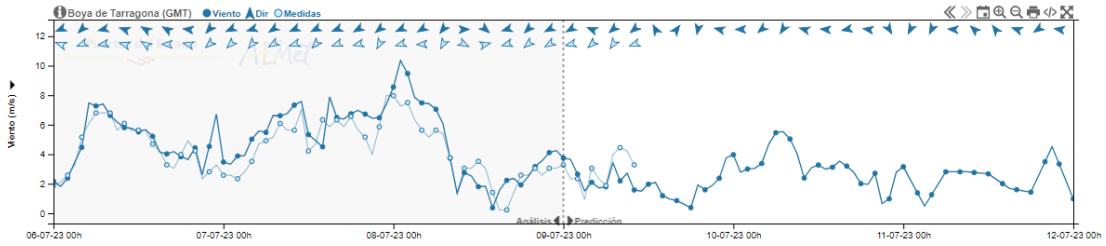

Figura 3.19: Chart de predicción que compara los modelos con los datos de observaciones de una estación cercana.

3.4.2. Charts de Tiempo Real

Son también series de tiempo que muestran los datos de distintos tipos de medidas, cada una con sus variables que pueden combinarse si son del mismo tipo.

En este caso los datos pueden tener distinta cadencia, desde medio segundo hasta una hora. Debido a esto, la cantidad de días iniciales que se cargan al principio no es siempre la misma, las estaciones con cadencias más bajas toman tantos datos al día que hay que limitar la cantidad máxima de estos, si no, las llamadas al backend serían demasiado lentas y el rendimiento del chart muy pobre.

Si solo se está viendo una variable, esta podrá compararse con los datos de otras estaciones cercanas que también la midan.

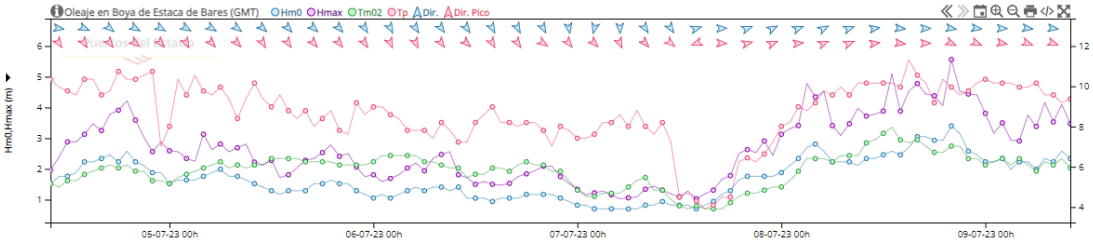

Figura 3.20: Chart de tiempo real mostrando seis variables en un mismo gráfico.

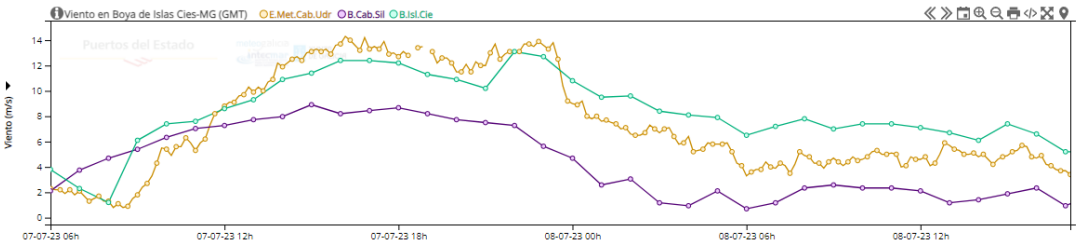

Figura 3.21: Chart de tiempo real que compara la velocidad del viento en tres estaciones cercanas.

3.4.3. Charts especiales de Nivel del Mar

Aunque los otros charts anteriores pueden dar también información del nivel del mar, se ha creado un gráfico propio ya que esta variable requiere de funcionalidades muy especiales.

Entre las series temporales, es la más compleja de todas, las características que tiene son:

- Una segunda serie de tiempo anidada al eje temporal de la principal con datos extra.
- Se añaden Máximos y Mínimos de toda la serie principal.
- Se pueden comparar varios modelos a la vez.
- Algunos modelos disponen de banda de confianza.
- Se pueden usar distintas referencias para la altura del nivel del mar.
- Aparecen las fases de la luna en el eje temporal.

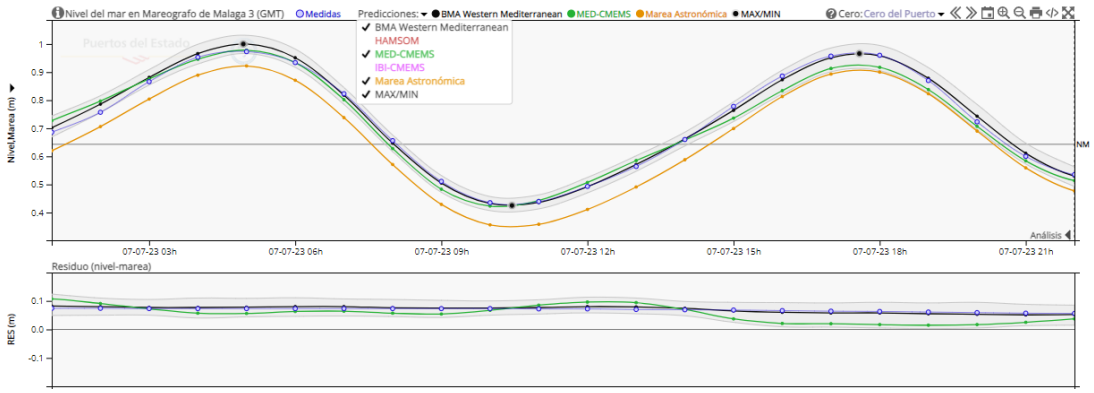

Figura 3.22: Chart de nivel del mar comparando varios modelos de predicción.

3.4.4. Charts de Marea Astronómica

Es una serie temporal bastante sencilla. La diferencia principal con el resto es que los datos no provienen de modelos de predicción o medidas, sino de un algoritmo que calcula cuánto afecta la posición de la luna a las mareas, por este motivo, la serie temporal no tiene límite ni en el pasado ni en el futuro, puede calcular cualquier día.

Consta solo de una serie de tiempo, junto a máximos y mínimos que también son calculados por el mismo algoritmo.

Para cada dato pintará la fase de la luna exacta en ese momento, también se puede cambiar la referencia entre algunas preestablecidas.

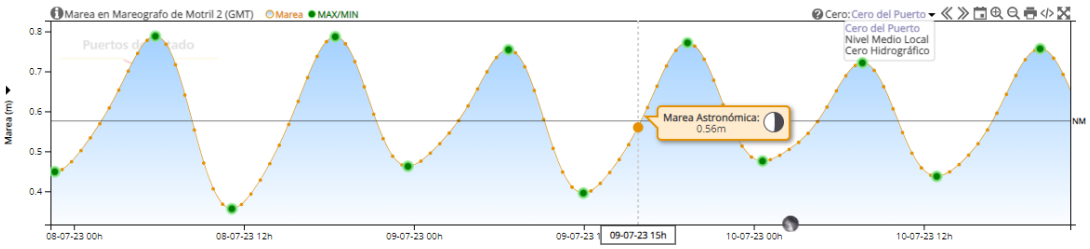

Figura 3.23: Chart de marea astronómica mostrando las diferentes referencias disponibles.

3.4.5. Tablas de Mareas

Estos gráficos se originan en unos antiguos reportes en PDF que se mandaban a los usuarios con unas tablas, como los usuarios estaban ya acostumbrados a este formato, se decide crear una versión mejorada e interactiva de estas tablas para que puedan seguir usandolo como siempre.

Estas tablas dan los mismos datos que los charts de marea astronómica pero presentados de una forma distinta. Se crea una tabla por cada día, en ella se representan todos los máximos y mínimos que ocurrirán.

Esta tabla se puede expandir para acceder a los datos horarios, y cada dato horario se puede a su vez expandir para dar los datos cada 10 minutos.

También se representa la fase de la luna en cada tabla diaria.

Figura 3.24: Tabla de Mareas mostrando máximos y mínimos diarios junto a la fase lunar.

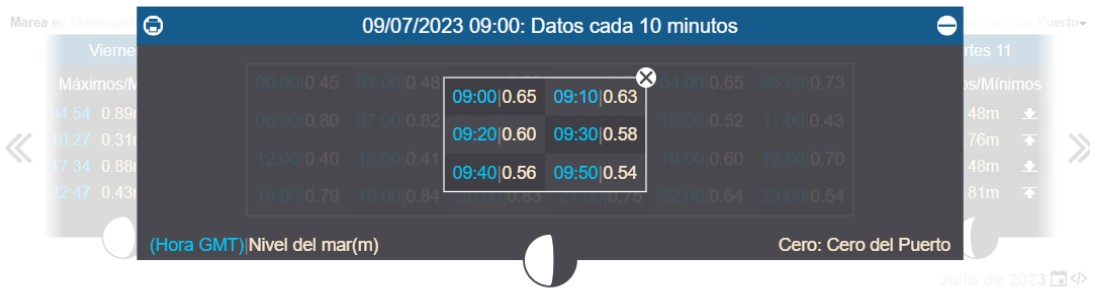

Figura 3.25: Tabla de Mareas mostrando datos cada 10 minutos.

3.4.6. Charts de Perfiladores

Los perfiladores son sistemas que miden las corrientes a diferentes metros de profundidad.

Si queremos pintar una serie temporal con varios datos distintos en cada punto, el tipo de gráfico más eficiente es un mapa de calor. Este es el tipo de gráfico más complejo de todo el proyecto.

En el eje Y se ponen las profundidades disponibles y el eje X será el del tiempo como siempre, en este caso, se divide el área de la gráfica en una matriz de rectángulos y cada uno de estos se pintará en función del valor que le corresponda a su posición en la matriz.

Puede representar tanto velocidad como dirección de la corriente, se le añade una pequeña serie temporal anidada al eje X del mapa de calor, esta representa los datos de solo la profundidad que está seleccionada.

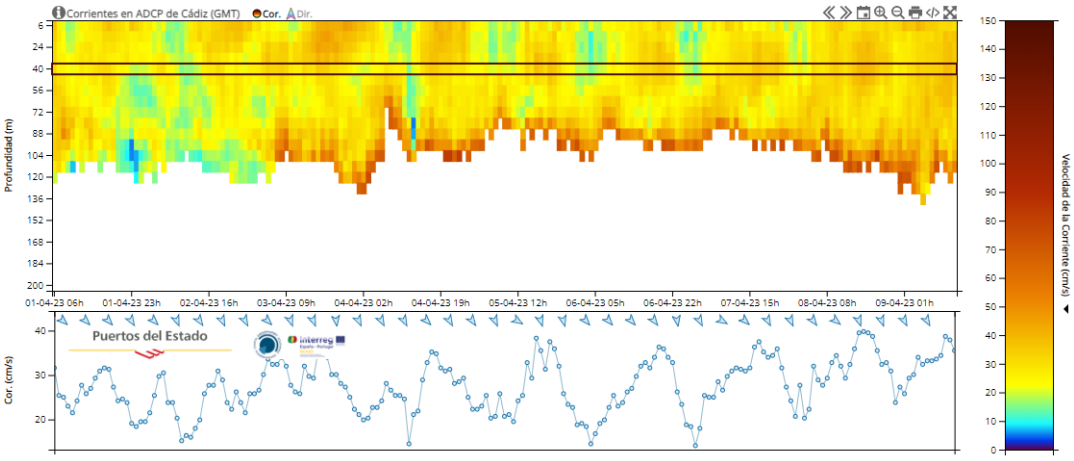

Figura 3.26: Chart de un perfilador mostrando la velocidad de la corriente a distintas profundidades.

3.5. Gestión de los datos

Hasta ahora solo hemos visto cómo representar gráficamente los datos, pero también es importante hacer una buena gestión de estos desde la aplicación para que los tiempos de carga y pintado sean los más óptimos posibles. En este caso, se reciben dos tipos distintos de datos:

- **Metadatos:**
  Son los datos relativos a la información del código de modelo o estación seleccionado, tales como latitud, longitud, nombre, fecha de alta y otro tipo de datos que se quieren mostrar al usuario. El peso de estos datos es muy pequeño y el encargado de servirlos es el propio backend de PortusData.
- **Datos de series temporales:**
  Es la información que se representa en los gráficos, el encargado de servirlos es el web service de acceso público de Puertos del Estado llamado SIMO, para ello se hacen llamadas con parámetros, tales como el código, modelo, fecha inicial y fecha final.

Figura 3.27: Metadatos de una estación de medida.

Como los metadatos son servidos por el backend, la gestión de estos se va a explicar en la otra memoria. Este trabajo se centra en los datos de las series temporales.

El web service devuelve la información en formato array, con todas las variables que se solicitan juntas. Esto nos permite recibir estos datos de forma rápida y sencilla, pero a la hora de representarlos y acceder a ellos no es lo más óptimo. Para ello, se separan todas las variables y se almacenan en memoria en un objeto JSON. Cada tipo de chart tiene una endpoint y formato de salida distinto, por lo que según que tipo de chart sea, parseará estos datos de una forma distinta y los convertirá a un formato generalizado por la aplicación.

Un aspecto importante sobre la gestión de estos datos es qué hacer cuando fallan o falta algunos de estos. En el caso del web service, normalmente no manda estas fechas en blanco, y esto genera un problema a la hora de representarlos ya

que la librería considera que la serie temporal continúa y pinta la línea entre estos, falseando el resultado final. Para solucionar esto, antes de parsear las distintas variables, hay que generar una serie temporal solo con la información de todas las fechas posibles, para ello se añade un elemento al objeto con la fecha inicial del gráfico, después se añaden más elementos sumando a la fecha anterior la cadencia que hay entre dos datos, hasta llegar a la fecha final. Una vez hecho esto, podemos recoger los datos de SIMO y añadirlos según coincidan con las fechas creadas anteriormente. Esto nos garantiza que ante la ausencia de algunos datos, en nuestra aplicación siempre van a existir esas fechas y por tanto se pinta un hueco entre ellos en vez de unirlos.

Cada punto de predicción o estación puede llegar a tener varios años de datos, por lo que hay que restringir la cantidad que pedimos de estos al web service por dos motivos:

- La carga inicial y posteriores peticiones serían demasiado lentas, una de las características que buscamos es que tarden en cargar menos de un segundo.
- Pintar demasiados puntos en una serie temporal consume mucha memoria del navegador y esto hace que tanto el gráfico como la aplicación que lo contiene tengan problemas de rendimiento.

Estos charts están pensados para dar la información más reciente posible, por lo que siempre cargará al principio los últimos datos disponibles. La primera llamada al web service suele pedir 21 días de datos, pero solo muestra en la primera visualización los últimos 7, esto permite que el usuario vea primero la información que más le interesa y pueda irse un poco más atrás en el tiempo sin tener que cargar de peticiones a SIMO. En los casos en los que tenemos cadencias con muchos datos por hora, la cantidad de carga inicial se reduce para evitar los problemas descritos anteriormente.

Figura 3.28: Minigráfico auxiliar donde se pueden ver todos los datos descargados y los que se están visualizando actualmente.

Si el usuario intenta navegar más a la izquierda del dato más antiguo cargado o si hace click manualmente en el icono del calendario, le aparecerán dos calendarios

que le permitirán seleccionar cualquier otra fecha con datos disponibles, el de la izquierda seleccionará la primera fecha y el de la derecha la última que se desea cargar, las características de estos son:

- Ambos calendarios están sincronizados para que el usuario no pueda cometer errores, como seleccionar una fecha final más antigua que el primer dato a cargar.
- Permiten navegar por día, mes, o año, gracias a ello se pueden seleccionar con comodidad los datos más antiguos.
- El de la derecha no te permitirá seleccionar más adelante en el tiempo que la última fecha disponible, en el caso de medidas en tiempo real, ese límite es el mismo día en el que se está cargando el chart, en el caso de predicciones pueden ser hasta los cinco siguientes, dependiendo del modelo.
- El de la izquierda tampoco te permitirá ir más al pasado si no hay datos, para ello el backend de PortusData manda una variable que contiene la información de la fecha de alta de una estación o de cuándo se lanzó por primera vez un modelo de predicción.

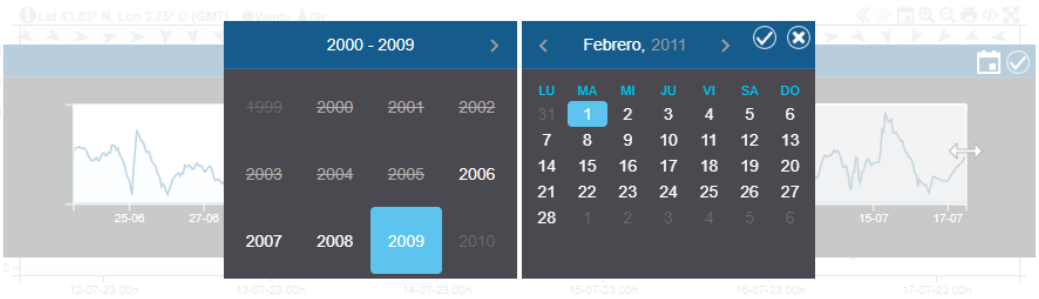

Figura 3.29: Calendarios de selección de fechas disponibles.

Por último, estos calendarios permiten ver hasta más de diez años atrás en el tiempo en algunos casos, como ya se ha explicado, intentar cargar todos estos datos sería ineficiente, por lo que también hay que limitar la cantidad máxima de estos que se pueden pedir al web service. Estos límites son los siguientes:

- Los charts de predicciones pueden descargar hasta seis meses de datos.
- Los de tiempo real pueden descargar también hasta seis meses, pero en las cadencias más pequeñas solo deja uno.
- En los de nivel del mar el límite son 100 días ya que descargan varios modelos de datos a la vez.

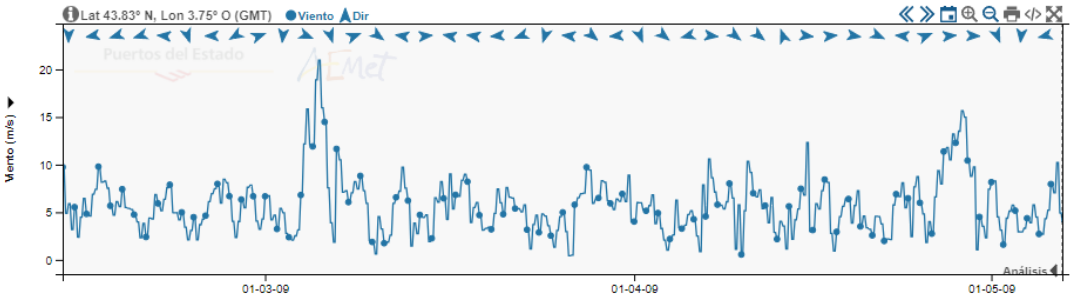

Figura 3.30: Chart de predicción mostrando datos de hace más de diez años, aunque no sea su uso habitual.

El caso de las tablas de mareas es especial, solo tiene un calendario y los datos se descargan por meses. Como se muestran varias tablas diarias una tras otra y los datos no tienen límite pasado ni futuro, se cargan inicialmente tres meses de datos: el actual, el mes anterior y el siguiente, si solo se cargara uno, el primer o último día de cada mes no podría mostrar las tarjetas adyacentes. Según se navega por las tablas y se pasa a un mes nuevo, se hace la descarga de datos de este y se borran los datos del mes que ya no es necesario, por lo que siempre las tablas tendrán tres meses de datos en memoria.

Figura 3.31: Los datos de mareas, al ser calculados por un algoritmo, no tienen límite de fechas.

4

Validación

En proyectos como este es muy difícil y poco práctico ejecutar baterías de pruebas automáticas, ya que el resultado de la aplicación es puramente visual y para validar un nuevo gráfico o un cambio en uno ya existente es necesario comprobar todo en su conjunto, el desarrollador hará sus pruebas, pero en este caso, para que se pueda dar el visto bueno para salir a producción con cosas nuevas, tiene que pasar la revisión por alguien experto en oceanografía, que en este caso es el propio cliente.

Como desarrollador la mejor herramienta que he tenido para ir validando los resultados según iba haciendo el código de la aplicación han sido unos widgets existentes de tablas con los mismos datos que tengo que representar, pero esto solo me vale para comprobar la integridad de estos datos.

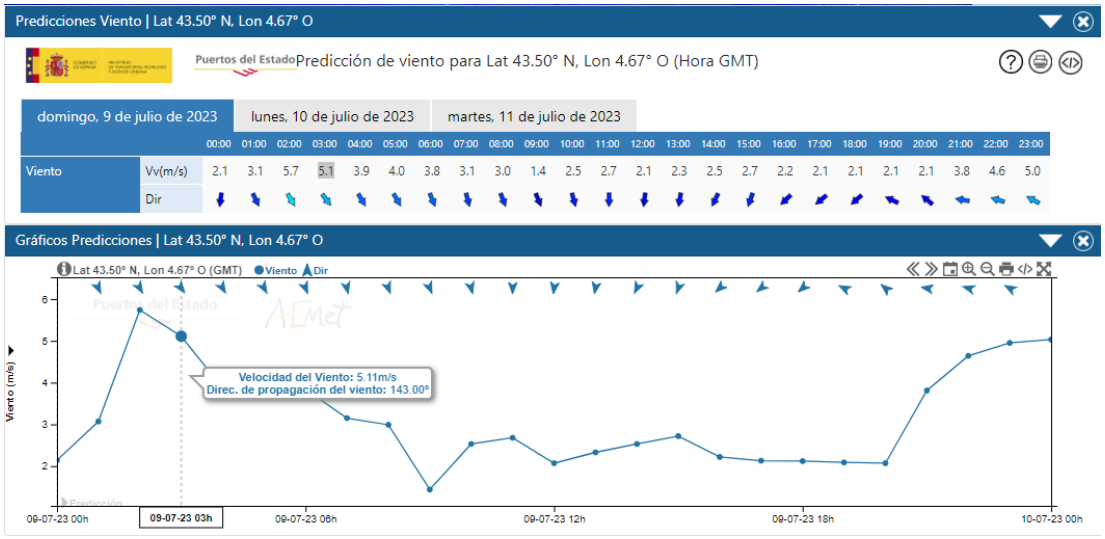

Figura 4.1: Comprobación de la integridad de los datos.

También me he encargado de validar la funcionalidad y la interactividad de los elementos, para ello he usado como herramienta las heurísticas de Nielsen.

Las heurísticas de Nielsen son diez reglas generales que se aplican para evaluar y mejorar la usabilidad en las interfaces de usuario:

- **Visibilidad del estado del sistema:**
  Para dar mayor visibilidad a los estados del sistema, se aportan herramientas como un custom loader con el logo del cliente, gracias a esto el usuario sabrá que la aplicación está descargando datos y tiene que esperar, las animaciones en este loader ayudan a que si se da un tiempo de carga más alto de lo normal, no parezca se ha quedado bloqueado.
  La leyenda interactiva nos permite saber en qué estado están las variables, haciendo un efecto de sombreado a las que se han ocultado.
- **Relación entre el sistema y el mundo real:**
  Para cumplir esta heurística se ha necesitado ayuda del cliente, que es quien me ha proporcionado los textos a mostrar. En oceanografía se usa un tipo de lenguaje técnico que es muy importante cumplir para que los usuarios tengan claros todos los conceptos de la representación.
  Como estos gráficos corresponden a medidas o predicciones del mundo real, los usuarios que vivan cerca de esos sitios pueden comparar que estos datos sean correctos, por ejemplo que los máximos y mínimos de mareas se correspondan con la realidad.
  Otro elemento de los gráficos muy fácil de comprobar son las fases de la luna que se pintan en alguno de ellos, solo bastará con mirar por la noche en la ventana y comprobar que los dibujos se ajustan a lo que se ve.
- **Control y libertad del usuario:**

En estos gráficos es muy difícil que el usuario cometa errores ya que están preparados para que las visualizaciones se calculen automáticamente.
El ejemplo más claro de error que puede cometer el usuario sería intentar cargar más datos que el máximo permitido, en estos casos, el usuario siempre será avisado de que ha cometido un error y se le informará de este límite para que pueda corregirlo.

- **Consistencia y estándares:**
  Para cumplir la consistencia, se ha buscado que todos los iconos y botones que se encuentran en el gráfico representen correctamente la acción que se va a realizar, por ejemplo, una impresora para imprimir.
  Respecto a los controles del ratón y táctiles, están programados de tal forma que se usen los gestos típicos para navegar por este tipo de aplicaciones.
- **Prevención de errores:**
  Darle libertad al usuario es una prioridad en este proyecto, pero a veces hay que prohibir algunas acciones para evitar que se cometan errores comunes. Un claro ejemplo de esto, es la restricción a solo poder controlar y moverse por el eje temporal de la gráfica, evita que el usuario pueda cargar visualizaciones incorrectas y la función de optimizar la vista recae sobre la aplicación.
  Los calendarios también están preparados para evitar errores, el usuario nunca podrá seleccionar una fecha final que sea anterior a la fecha inicial, evitando fallos en la posterior llamada al web service.
- **Reconocer antes que recordar:**
  Esta aplicación tiene usuarios que suelen ser bastante frecuentes, hay personas que comprueban todos los días varias de estas predicciones.
  Los códigos de colores que usamos para las variables y la estandarización de nombres hacen que el usuario no tenga que estar mirando la leyenda o el tooltip constantemente para recordar los elementos que se ven en el gráfico.
- **Flexibilidad y eficiencia de uso:**
  Esta aplicación la usan tanto personas expertas en oceanografía como usuarios más casuales que solo quieren saber una predicción para un fin en concreto.
  PortusData está diseñado para que cualquiera pueda usarlo, para ello siempre empezará desde un estado básico, y el usuario con sus acciones podrá aumentar la complejidad del sistema si desea, por ejemplo, añadiendo más modelos para comprarlos juntos en los gráficos de nivel del mar.
- **Diseño estético y minimalista:**
  Para cumplir la heurística de diseño, se han fabricado distintos tipos de interfaces, desde la más sencilla hasta la más compleja, gracias a esto pueden usarse en prácticamente cualquier tipo de aplicación o elemento multimedia.
  Otro elemento importante es que el diseño se ha hecho usando unos estilos parecidos a la aplicación Portus, que es de donde vendrá la mayor parte del tráfico de estos charts.
- **Ayudar a los usuarios a reconocer, diagnosticar y corregir los**

errores:
A la hora de cargar el gráfico pueden ocurrir varios tipos de errores, como una url mal formada, o la ausencia de datos, en estos casos el usuario no puede hacer nada, pero es importante que se informe debidamente de esto. Para ayudar a la comprensión de estos errores, se intenta que cada uno de estos tenga su mensaje personalizado, ayudando a saber qué es lo que está pasando.

- **Ayuda y documentación:**
  Estos gráficos son muy sencillos de usar y la mayoría de usuarios no necesitan ayuda para aprender a manejarlos, aún así, se aporta un breve manual [6] que explica la mayor parte de la funcionalidad y controles de los charts.

Una vez se hace la primera versión del gráfico, pasa la revisión del experto, ya que hay algunos elementos que son bastante complejos o no se ha tenido tiempo para traspasar el conocimiento necesario, cosas como:

- Las conversiones de unidades y cambios de referencia se hacen de forma correcta.
- La escala de los datos es la más óptima posible.
- Las flechas de dirección apuntan al lugar correcto (en oceanografía se usan referencias distintas).
- Las fases de la luna están correctamente pintadas.

Una vez pasa la validación, se puede incluir a producción el nuevo gráfico para que los usuarios accedan a él.

Como curiosidad, estos gráficos sirven como una propia herramienta de validación de los modelos y datos de observaciones del cliente, por ejemplo, los gráficos que comparan los datos de predicciones con los datos de las estaciones de medida sirven para evaluar la calidad de estos modelos, también, son la forma más eficiente de acceder a los datos y gracias a ellos se puede detectar de forma rápida errores en los modelos o estaciones que no están funcionando de forma de correcta.

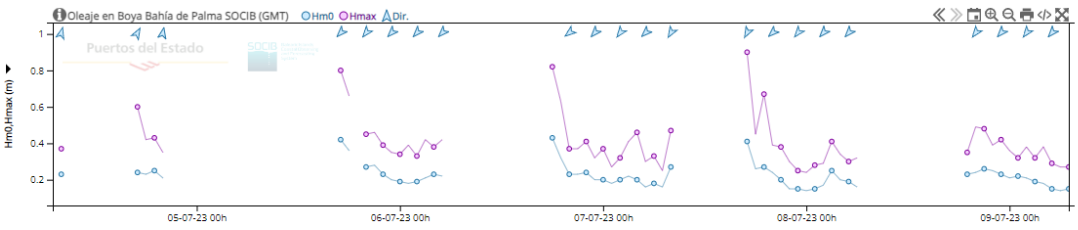

Figura 4.2: Detectada una estación con cortes de transmisión.

5

Conclusiones

Aunque es un proyecto para mostrar gráficos a todo el público, el primer usuario de estos charts es el propio cliente, por lo que el feedback siempre ha sido muy rápido y los errores se han podido detectar y corregir en fases muy tempranas del desarrollo.

Gracias al uso de las heurísticas de Nielsen se puede concluir de forma objetiva que la interfaz y la interacción con el usuario son las que se esperan de una aplicación de este tipo.

Dado que esta aplicación está ya en producción y lleva funcionando varios años, ya se sabe que ha cumplido todos sus objetivos.

La acogida por los usuarios más comunes siempre ha sido muy buena, nunca nadie ha echado de menos el anterior sistema de series temporales.

Por último hay que destacar que también se han recibido felicitaciones de gente más experta que depende de estos gráficos para su trabajo, dando un feedback muy positivo de la simplicidad de uso y de la fiabilidad de las representaciones gráficas.

5.1. Alcance final

La estructura de código común del proyecto hace que sea bastante fácil crear nuevos tipos de gráficos y por eso el proyecto ha ido creciendo hasta su estado actual. Gracias a todos los tipos de interfaz y modos de control que se han creado, han permitido que tenga un uso mucho más allá de la aplicación para la que fueron creados, podemos encontrarlos en varios proyectos de Puertos del Estado, tanto públicos como privados:

- **Portus** [1]**:** Es la aplicación principal para la que fueron pensados estos charts, sirven toda la información que se encuentra en los apartados de "Predicciones" y "Tiempo real". Cuenta con decenas de parámetros distintos, cientos de estaciones y miles de puntos de modelo, por lo que las distintas series temporales que pueden mostrar estos gráficos son prácticamente infinitas.
- **iMar** [7]**:** Es una app para Android e iOS que muestra los datos de Portus en una forma más manejable para dispositivos táctiles. En esta aplicación primero se presentan los charts en pequeño usando la interfaz de previsualización, pero permite expandirlo a pantalla completa, donde el usuario ya podrá interactuar con el chart de manera táctil.
- **CMA** [8]**:** En este caso es una aplicación de acceso privado a usuarios registrados. Se muestran prácticamente los mismos gráficos que en Portus, pero esta aplicación hace uso de los umbrales. También, gracias a la interfaz "img", permite insertar estos gráficos en los correos de alertas que se mandan.

Figura 5.1: Charts en CMA mostrando umbrales.

- **Reportes en PDF:** Gracias a poder devolver el gráfico en formato de imagen se pueden insertar en cualquier elemento multimedia, otro uso que se le da es para mostrar datos en reportes PDF que se generan diariamente de forma automática y se mandan a usuarios seleccionados.

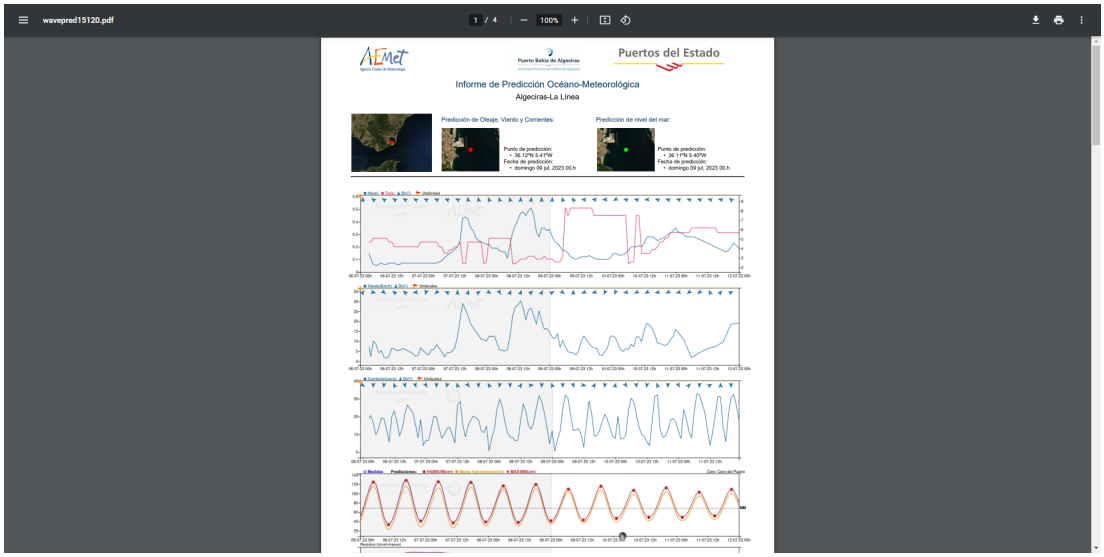

Figura 5.2: Charts integrados en reportes en PDF.

También se han usado en algunas aplicaciones o proyectos de menor importancia, gracias a que el cliente permite que los gráficos puedan ser compartidos como widgets por cualquier persona, se pueden encontrar estos charts en muchas páginas externas a Puertos del Estado, la mayoría de estas webs que comparten los gráficos están dedicadas al surf o tratan temáticas de meteorología.

Bibliografía

[1] P. del Estado, "Portus." [Online]. Available: https://portus.puertos.es/

[2] "Puertos del estado." [Online]. Available: https://www.puertos.es/

[3] "Spring." [Online]. Available: https://spring.io/

[4] "Google web toolkit." [Online]. Available: https://www.gwtproject.org/

[5] "Data driven documents (d3.js)." [Online]. Available: https://d3js.org/

[6] Óscar Ballesteros Izquierdo, "Manual portusdata." [Online]. Available: https://portus.puertos.es/Portus/docs/documCharts.pdf

[7] P. del Estado, "imar." [Online]. Available: https://play.google.com/store/apps/details?id=es.nologin.imar.android

[8] ——, "Cuadro de mando ambiental (cma)." [Online]. Available: https://cma.puertos.es/

Apéndices

A

Prueba PortusData

Como ya se ha indicado, PortusData es un proyecto público al que cualquier usuario puede tener acceso, la forma más fácil de ver el proyecto es accediendo a la aplicación Portus y navegar por las secciones de Predicción y Tiempo Real para seleccionar las distintas variables y puntos de modelo que pueden mostrar estos gráficos.

Para facilitar aún más el acceso, voy a recopilar una lista de URLs de acceso directo a cada uno de los tipos distintos de charts que existen y que muestren las características más importantes del proyecto.

Los gráficos están configurados para ocupar el máximo ancho y alto de su contenedor, la aplicación que los incluye es la que maneja el tamaño y relación de aspecto de estos contenedores para lograr la visualización más óptima posible, en este caso, como accedemos directamente por URL, el contenedor es el propio navegador, por lo que se recomienda cambiar el tamaño de la ventana a uno en el que el ancho sea el doble del alto.

Tanto los datos de predicciones como los de tiempo real se renuevan cada día, por lo que es posible que alguno de los modelos falle o una estación se estropee y alguna de las URLs que se presentarán a continuación falle debido a que no tenga datos disponibles en el momento de acceso.

Charts de Predicción

Todas las variables posibles de oleaje combinadas en un solo gráfico: `https://portus.puertos.es/PortusData/predChart?code=3120048&var=Hm0,Tp,Tm02&dirVar=MeanDir180`

Comparación del modelo de predicción de viento con los datos medidos de la estación más cercana: `https://portus.puertos.es/PortusData/predChart?code=2056079&station=2548&var=WindSpeed&dirVar=WindDir180`

Charts de Tiempo Real

Medidas de oleaje de una estación: `https://portus.puertos.es/PortusData/rtChart?`

station=2630&params=Hm0,Hmax&dirParams=MeanDir,MeanDirPeak

Observaciones de Nivel del Mar con cadencia 1 minuto: https://portus.puertos.es/PortusData/rtChart?station=3210&params=SeaLevel

Charts de Nivel del Mar

Los distintos modelos se van añadiendo a lo largo del día según se ejecutan, los modelos con bandas de confianza suelen estar disponibles a partir de las 14:00 cada día.

Nivel del Mar en el Mediterráneo: https://portus.puertos.es/PortusData/nivmarChart?code=16210&station=3656&var=SeaLevel,SeaSea,Residual

Nivel del Mar en el Atlántico: https://portus.puertos.es/PortusData/nivmarChart?code=11210&station=3108&var=SeaLevel,SeaSea,Residual

Charts y tablas de Mareas

Chart de mareas: https://portus.puertos.es/PortusData/astroChart?code=3219

Tablas de mareas: https://portus.puertos.es/PortusData/tablasMareas?code=3219

Charts de Perfiladores

Actualmente todos los perfiladores que hay están sin transmisión o tienen algún fallo en los datos, solo se puede acceder a uno de ellos y para ver datos buenos hay que seleccionar el calendario y cambiar las fechas a alguna pasada, por ejemplo, en Abril de 2023 los datos son correctos.

Perfilador de Cádiz: https://portus.puertos.es/PortusData/ADCPChart.html?station=5342&params=CurrentSpeed

Interfaces

Para una visualización óptima de las interfaces mínima y de previsualización, se recomienda un tamaño de ventana pequeño.

Interfaz default: https://portus.puertos.es/PortusData/predChart?code=3043043&var=Hm0&dirVar=MeanDir180&int=default

Interfaz mínima: https://portus.puertos.es/PortusData/predChart?code=3043043&var=Hm0&dirVar=MeanDir180&int=min

Interdaz de previsualización:https://portus.puertos.es/PortusData/predChart?code=3043043&var=Hm0&dirVar=MeanDir180&int=prev

Interfaz imagen: https://portus.puertos.es/PortusData/predChart?code=3043043&var=Hm0&dirVar=MeanDir180&int=img

B

Instalar proyecto en local

Para poder arrancar el proyecto en local lo primero que se necesita es tener Java 8 instalado en el sistema. También será necesario tener NodeJS instalado y añadir la librería Babel de forma global:

"npm i -g @babel/cli @babel/core"

Por último, si se quiere probar el exportar gráficos a imagen, se tiene que tener instalado phantomjs y asegurarse de que los binarios están incluidos en el PATH.

Si se está haciendo esto desde un Linux es necesario modificar la línea 95 del archivo "pom.xml" para cambiar la extensión del script a ".sh"

El entorno de desarrollo usado es Eclipse, pero no debería haber problema en usar otros.

El primer paso es importar el directorio entero como proyecto, detectará automáticamente que es un proyecto Maven y comenzará a instalar las librerías necesarias.

Para que funcione el frontend es necesario compilar primero, para ello se hace click derecho en el proyecto - Run As.. - Maven Build.. Se abrirá un popup con varias opciones, en el campo Goals se pone "compile" y se pulsa en "Run"

Si todo ha salido bien ya se puede arrancar, para ello simplemente hay que hacerlo como cualquier aplicación Java.

Click derecho en el proyecto - Run As.. - Java Application

Se abrirá un nuevo popup para seleccionar la clase de entrada, hay que buscar "PortusDataApplication" y darle al OK, con esto ya debería inicializarse.

Para empaquetar la aplicación y generar el WAR hay que volver a ejecutar un comando Maven:

Click derecho en el proyecto - Run As.. - Maven Build.. Y en este caso se pone "package"

como goal.

Si todo ha salido bien, el .war se encontrará en la carpeta “target“.