ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto fin de Carrera - TFG

Ingeniería Informática - Ingeniería del Software

My Crossing

Realizado por
Moisés Pantión Loza
Adrián Gracia Barroso

Dirigido por
José Ramón Portillo Fernández

Departamento
Matemática Aplicada I

Sevilla, Junio de 2021

Resumen

En Marzo del pasado año 2020, salió a la venta la última entrega de la saga de videojuegos "Animal Crossing", publicado por Nintendo. Debido a la pandemia del COVID-19 que coincidió con la salida del videojuego, muchas personas fueron obligadas a quedarse en casa realizando cuarentena, y gracias a los servidores online que permiten la conexión entre jugadores, se creó una comunidad bastante amplia alrededor del videojuego que se vio potenciada debido a dicha pandemia.

De esta forma, no tardaron mucho en salir varias webs y páginas cuyo objetivo era informar y conectar a los jugadores que disfrutaban del juego. Del mismo modo, este trabajo al que hemos bautizado bajo el título de "My Crossing", no es sino un sistema de información que, como muchas otras, sirve de ayuda para el jugador proporcionándole herramientas e información sobre el juego.

No obstante, buscábamos proporcionarle al usuario un sistema de información lo más completo posible y que le ayudase tanto en el día a día como en tareas más esporádicas, por lo que como usuarios y parte de la comunidad que somos, decidimos implementar funcionalidades e ideas que hasta ahora no se habían visto en los demás sitios web y que son de gran utilidad.

La aplicación web está desarrollada principalmente con Spring Boot y Angular, ya que además de realizar el sistema también queríamos ampliar nuestro conocimiento sobre esta última tecnología ya que es bastante útil para el desarrollo web y está actualmente en auge.

Agradecimientos

TODO==========

Índice general

Índice de cuadros

Índice de figuras

Índice de código

CAPÍTULO 1

Introducción

En Internet podemos encontrar gran cantidad de herramientas para calcular diferentes aspectos y datos del videojuego "Animal Crossing: New Horizons". El problema recae en que muchas herramientas que podrían ser bastante útiles aún no han sido desarrolladas, así como la falta de un servicio web que las recoja a todas. Esto hace perder bastante tiempo a todos aquellos jugadores que deseen consultar ciertos datos, ya que tendrían que navegar por distintos sitios, en vez de disponer de todo lo que necesitan en un mismo lugar.

Debido a esto, llegamos a la decisión de crear una aplicación web que contenga una colección de herramientas, así como de desarrollar algunas nuevas por nuestra cuenta, todo esto con una interfaz simple que permita acceder a cada una sin ninguna dificultad.

Nuestra aplicación web mostrará ciertos datos dependiendo de si pertenecemos al hemisferio norte o sur, de la hora actual, de la fecha en la que nos encontremos etc, por lo que podemos concluir con que el servicio se podrá utilizar de forma internacional, siempre y cuando se sepa la lengua española.

CAPÍTULO 2

Definición de objetivos

Para que podamos dar este proyecto por finalizado, debemos cumplir una serie de objetivos:

- **Objetivos académicos:**
  - Centralizar las distintas herramientas que ya existen en una sola aplicación web, de forma que el usuario disponga de toda la información en un mismo lugar.
  - Crear como mínimo una herramienta nueva que aporte una funcionalidad útil de cara al videojuego.
  - Permitir el registro de usuarios y el almacenaje de datos del mismo
  - Evitar la reiterada introducción de datos de forma manual. Algunas herramientas necesitarán ciertos datos del jugador para realizar los cálculos, y dado que es una pesada tarea el introducirlos uno a uno varias veces, queremos intentar automatizarlo dentro de lo posible para que el usuario tenga que introducir los datos el mínimo numero de veces.
- **Objetivos personales:**
  - Realizar el apartado de front-end de este proyecto con la tecnología Angular, no solo debido a su utilidad para el desarrollo web de una sola página, sino también por su popularidad en el sector laboral, ya que pensamos que puede sernos útil de cara al futuro. Esta va a ser la primera vez que tratemos con Angular, por lo que pensamos que aprender a usar una tecnología desde cero y aplicarla de forma que se obtenga el mejor resultado posible es un gran reto que nos gustaría afrontar.
  - Usar PHP para el back-end, ya que es una tecnología que hemos visto poco en la carrera y que también tiene su popularidad en el sector laboral, por lo que pensamos que también podemos aprovechar esta oportunidad para ampliar nuestro conocimiento sobre la misma de forma que aumenten nuestras capacidades. Respecto a la base de datos, queremos usar MariaDatabase (derivado de MySQL) ya que durante la carrera hemos visto en profundidad las bases de datos relacionales, en especial MySQL, y pensamos que puede ser una buena oportunidad para aplicar los conocimientos que hemos adquirido durante la carrera.

CAPÍTULO 3

Análisis de antecedentes y aportación realizada

3.1– Análisis de antecedentes

A continuación se mostrarán las herramientas a las que uno debería de acceder para obtener toda la información que pudiera necesitar sobre el juego:

3.1.1. Servicios de cálculo de nabos

En el videojuego *Animal Crossing: New Horizons* hay implementado un mercado similar al de las acciones, pero con nabos. El precio de compra y el de venta, aunque limitados por un rango, varía cada semana. Esta funcionalidad es usada por los jugadores para amasar grandes fortunas comprando cientos de nabos y después, haciendo uso de servicios de cálculo y predicción de patrones de venta, venderlos cuando se pueda sacar el mayor beneficio.

Muchos servicios se centran exclusivamente en esta función del cálculo de futuros patrones de venta, y además precisan de bastante información, al igual que en nuestro servicio, para poder realizar los cálculos necesarios para dar una aproximación cercana a la realidad.

Figura 3.1: Turnip Prophet

Este tipo de servicio lo proveen varias páginas web, como Turnip Prophet (véase la figura 3.1), siendo esta la página mas conocida y con una interfaz más cercana al usuario, siguiendo un diseño similar a la estética del HUD de *Animal Crossing: New Horizons*; o Turnip Price Calculator, la cual es una web un poco más técnica que ofrece varias gráficas en la que obtener información adicional sobre las probabilidades de los patrones de venta.

Todos estos servicios web, incluido el nuestro, han podido hacer uso de dicha herramienta gracias al usuario *Ninji*, el cual analizó el código del videojuego para desarrollar un algoritmo que pudiera obtener la aproximación de los patrones de venta de la próxima semana.

3.1.2. Servicios de cálculo de probabilidades sobre mudanzas

En este videojuego existe la posibilidad de que tus vecinos decidan irse de tu isla. Esta funcionalidad, al igual que la de los nabos, sigue un algoritmo que se puede usar para calcular la probabilidad de que alguien se mude en un día en concreto, y si se da el caso, averiguar cuál es el vecino que tiene más probabilidades de mudarse.

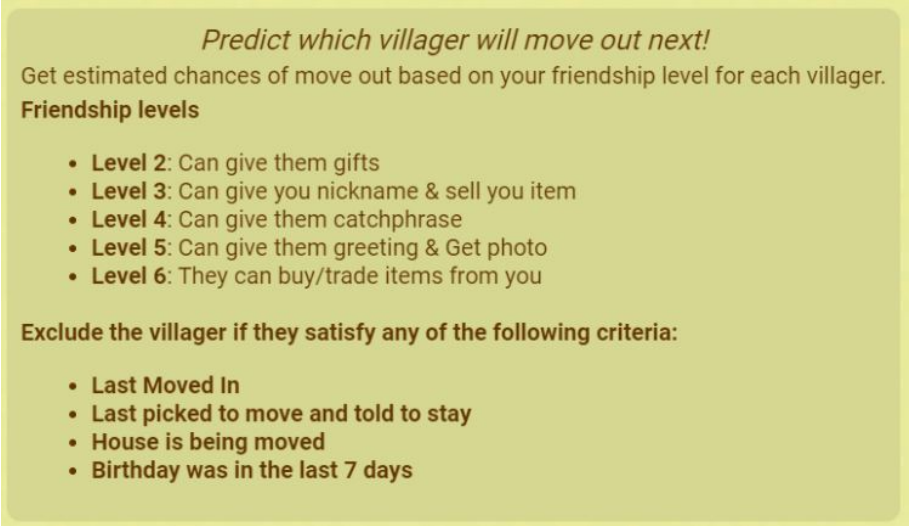


Figura 3.2: Mudanza

Para realizar esos cálculos existen varias páginas web (véase la figura 3.2), como puede ser NookPlaza, la cual no solo ofrece la funcionalidad de calcular las mudanzas de vecinos, sino que además ofrece algunas pequeñas funcionalidades más. Otro ejemplo de este servicio sería Yue's Move-Out Calculator, donde además se puede calcular la probabilidad de mudanza en caso de que más de un jugador viva en la misma isla, añadiendo datos sobre cada jugador para realizar una media y obtener una predicción fiable.

Al igual que con el algoritmo de los nabos, este servicio también ha sido desarrollado por el usuario *Ninji*.

3.1.3. Nook's Island

En esta página podemos encontrar varios servicios a destacar. Para empezar, dispone de un mercado donde cualquier jugador puede poner a la venta artículos que no necesite y desee intercambiar por bienes dentro del juego. Asimismo, también pueden comprar productos que se encuentren anunciados. Todo el servicio del mercado es informativo, ya que la página no lleva un sistema de venta como tal, sino que solo informa a los usuarios y los pone en contacto para que sean ellos los que se reúnan de forma online en el videojuego y realicen la transacción. Gracias a este servicio, se puede contar con una gran interacción de la comunidad mediante la página web, lo que le da mucha vida.

Figura 3.3: Nook's Island

Esta es una funcionalidad que hemos decidido no añadir ya que hemos considerado que con el conjunto de herramientas que vamos a centralizar, estábamos realizando bastante trabajo y añadir una funcionalidad tan extensa como esta puede resultar demasiado ambicioso, ya que aunque siendo meramente informativo, habría que realizar algún tipo de conexión entre usuarios de forma que se pudiesen comunicar. Además, el objetivo de nuestro sistema era centralizar todas las herramientas que un usuario pueda necesitar a la hora de jugar de forma individual, como si de una enciclopedia con funcionalidades extra se tratase, por lo que no se busca la interacción entre usuarios. Sin embargo, no se descarta completamente en caso de que haya que aumentar el alcance del proyecto.

Entre otras funcionalidades, Nook's Island cuenta también con un sistema para compartir Códigos de Sueño y Diseños personalizados, así como una enciclopedia de criaturas. Sin embargo, creemos que tanto el apartado visual como la accesibilidad de la web es bastante mejorable. Estas funciones estarán en nuestro sistema de una forma algo distinta, cambiando la información que se muestra de forma que lo más importante se encuentre a primera vista y los detalles menos importantes se queden en segundo plano. De esta forma se espera que el usuario pueda hacer uso de ellas de una forma más fácil e intuitiva.

3.2– Aportación realizada

Los servicios mostrados anteriormente no son los únicos que se pueden encontrar en Internet, hay una extensa cantidad de herramientas disponibles pero la mayoría se encuentran en sitios web distintos y algunas pueden llegar a ser algo confusas y poco intuitivas para el usuario.

Para hacerle más fácil la búsqueda de información al usuario, nuestro proyecto centraliza varias de las herramientas existentes en un mismo sitio web, buscando obtener una interfaz sencilla y simple que pueda ser apta para todos los públicos, incluidos los más pequeños, ya que no hay que olvidar que este videojuego abarca un público bastante extenso.

La centralización de las herramientas resulta en un ahorro de tiempo notable para los usuarios que busquen información de forma frecuente, además de dar a conocer aspectos del juego y funciones que puede que no conozcan del todo. De esta forma buscamos que les pueda ser útil para obtener ciertos objetos o hitos en el juego.

Por último, hemos añadido una herramienta original (que a día de hoy no hemos conseguido encontrar en Internet): El Vestidor. Esta herramienta simula un vestidor de cualquier tienda

de ropa, pero aplicada a las prendas del videojuego. Dispone de una interfaz que permite personalizar al personaje para que resulte lo más parecido al del usuario, o incluso probar nuevas combinaciones. De la misma forma, se puede vestir a dicho personaje con cualquier prenda que forme parte del videojuego.

Pensamos que esta herramienta es bastante útil ya que de este modo, un usuario puede probar distintos estilos y prendas para ver cual le convence más sin necesidad de disponer del objeto dentro del videojuego. De esta forma se puede centrar en conseguir aquella prenda que le interese en vez de realizar un proceso de prueba y error adquiriendo ropa que no utilizará y gastando tiempo y recursos. Además, cuenta con filtros para que no solo la búsqueda sea mas sencilla, sino que pueda idear conjuntos de ropa de manera más fácil.

CAPÍTULO 4

Análisis temporal y costes de desarrollo

4.1– Análisis temporal

4.1.1. Iteraciones

Dividiremos el trabajo en varias iteraciones, de forma que el proyecto se quede lo más planificado posible para que, a la hora de trabajar, solo haya que seguir las iteraciones:

Iteración 1

En esta iteración buscamos dejar empezado tanto el proyecto como la documentación, creando una base sobre la que trabajar más adelante. Además dejamos preparadas las herramientas a utilizar, tanto de trabajo (Microsoft Visual Studio Code, XAMPP), como de redacción (TeXStudio, Google Docs), almacenamiento en la nube (GitHub, Google Drive) y gestión del tiempo (Toggl). Realizamos el primer capítulo de la documentación y diseñamos los mockups.

Iteración 2

Planificamos el proyecto de forma que se quede dividido para las siguientes iteraciones y continuamos con la documentación. Redactamos los apartados que podemos (ya que algunos de ellos requieren que se haya avanzado más) de los capítulos 2, 3, 4 y 5. Además se realiza un curso de Angular para prepararnos para la siguiente iteración.

Iteración 3

Esta iteración está pensada como introducción a Angular. Se planea tener lista la página de inicio, el registro de usuario y login, el acceso a perfil y su respectiva información sobre el usuario, así como las diversas acciones que puede realizar desde el mismo (a excepción del álbum de fotos). Se busca establecer la estructura de la base de datos, realizar la vista (final a ser posible) de las páginas mencionadas así como los elementos back-end necesarios para permitir a los usuarios registrarse en la plataforma, loguearse y acceder a su perfil. ¿¿¿¿¿Se realizan las pruebas respectivas.????

Iteración 4

Se busca implementar API y los elementos relacionados con esta. Habría que hacer un re-

paso de lo hecho en la iteración anterior para aplicar las imagenes y valores que se necesiten de la API. Además se planea realizar los distintos catálogos (a excepción del de sueños y canciones) así como el listado de criaturas en tiempo real. Se realizan las pruebas (+ las del anterior?).

Iteración 5

Implementar las funciones de calculadora de nabos, calendario de eventos, álbum de fotos, catálogo de sueño, catálogo de canciones y vestuario, así como las pruebas respectivas.

Iteración 6

Terminar la documentación, realizar los apartados restantes de la documentación y cerrar el proyecto.

4.1.2. Estimación vs Realidad

Iteración	Horas estimadas	Horas reales	Diferencia (
1	15	12	-3
2	50		
3	85		
4	200		
5	85		
6	90		
7	80		
8	45		
Total	650		

//TODO cambiar tabla Estimacion - Reales - Diferencia 1: 15 - 12 2: 50 - 56 3: 170 - 4: 170 5: 200 6: 45 Total: 650

4.2– Costes de desarrollo

4.2.1. Costes Directos

Personal

Dado que aunque sepamos algo de PHP gracias a algunas asignaturas cursadas anteriormente, no tenemos ninguna experiencia con Angular, calcularemos nuestros sueldos clasificándonos como programadores junior. Dicho conjunto cobra una media de 19.000€ al año, lo cual sería 1.584€ al mes, que al dividirlos en un horario de trabajo de 40 horas semanales, 160 horas al mes, equivaldría a 10€ la hora.

Esto hay que multiplicarlo por dos ya que el equipo del proyecto está formado por dos personas, es decir, dos programadores junior con sus respectivos sueldos, por lo que sería un total de 20€ la hora.

4.2.2. Costes Indirectos

Realizaremos el proyecto en Microsoft Visual Studio Code, un entorno de desarrollo gratuito, en una base de TypeScript con Angular usando el back-end en PHP a modo de API encargado de realizar las peticiones a una base de datos de MariaDatabase (MySQL), cuyas licencias son gratuitas.

Para el almacenamiento de datos en la nube usaremos GitHub y Google Drive, los cuales disponen de licencia gratuita con aspectos limitados.

Para la redacción utilizaremos tanto Google Docs como TeXStudio, ambos con licencia gratuita.

Para la gestión de tiempo y comunicación usaremos Toggl y WhatsApp respectivamente, las cuales también disponen de licencia gratuita (con limitaciones en el caso de Toggl).

Los únicos costes indirectos que podríamos encontrar sería la electricidad y la conexión a Internet, lo cual supone de media 35€ por persona, 70€ en total al mes.

4.2.3. Amortizaciones

Para realizar el proyecto hemos usado los siguientes equipos:

	Equipo 1	Equipo 2
Procesador	Intel Core i-5 4460	Intel Core i-5 4460
Almacenamiento	1 Tb	250 Gb
Memoria	16 GB	8 GB
Gráfica	NVIDIA GeForce GTX 950	NVIDIA GeForce 750 GTX
Precio	800€	650€

Además de esto para comunicarnos hemos usado nuestros teléfonos móviles, por eso hemos decidido añadirlo al documento:

	Móvil 1	Móvil 2
Modelo	Samsung Galaxy J7	Xiaomi MiA4
Precio	200€	130€

4.2.4. Total

Finalmente, sumaremos todos los cálculos anteriores para dar una cifra al precio total:

TODO TABLA TOTAL

CAPÍTULO 5

Análisis de requisitos y diseño

5.1– Participantes del proyecto

Organización	Departamento de Matemática Aplicada I
Dirección	E.T.S. de Ingeniería Informática, Av. Reina Mercedes s/n, 41012 Sevilla
Teléfono	954 552 766
Email	secretarma1@us.es

Participante	Moisés Pantión Loza
Organización	TFG MyCrossing
Rol	Desarrollador

Participante	Adrián Gracia Barroso
Organización	TFG MyCrossing
Rol	Desarrollador

Participante	José Ramón Portillo Fernández
Organización	Departamento de Matemática Aplicada I
Rol	Tutor

5.2– Requisitos

5.2.1. Requisitos de Información

//TODO Hacer requisito de info sobre usuario

IRQ-001	Información sobre Insectos

El sistema deberá almacenar y mostrar los siguientes datos sobre los insectos:

- Nombre
- Disponibilidad
- Rareza
- Imagen
- Precio

IRQ-002	Información sobre Peces

El sistema deberá almacenar y mostrar los siguientes datos sobre los peces:

- Nombre
- Disponibilidad
- Sombra
- Rareza
- Imagen
- Precio

IRQ-003	Información sobre Criaturas de mar

El sistema deberá almacenar y mostrar los siguientes datos sobre las criaturas de mar:

- Nombre
- Disponibilidad
- Sombra
- Velocidad
- Imagen
- Precio

IRQ-004	Información sobre Fósiles

El sistema deberá almacenar y mostrar los siguientes datos sobre los fósiles:

- Nombre
- Imagen
- Precio

IRQ-005	Información sobre Arte

El sistema deberá almacenar y mostrar los siguientes datos sobre las obras de arte:

- Nombre
- Imagen
- Existencia de copia falsa

IRQ-006	Información sobre Aldeanos

El sistema deberá almacenar y mostrar los siguientes datos sobre los aldeanos:

- Nombre
- Imagen
- Icono
- Personalidad
- Cumpleaños
- Especie
- Género

IRQ-007	Información sobre Canciones

El sistema deberá almacenar y mostrar los siguientes datos sobre las canciones:

- Nombre
- Imagen
- Música

IRQ-008	Información sobre Ropa

El sistema deberá almacenar y mostrar los siguientes datos sobre las prendas de ropa:

- Nombre
- Tipo de prenda
- Fuente
- Imagen
- Variaciones
- Precio

IRQ-009	Información sobre Muebles

El sistema deberá almacenar y mostrar los siguientes datos sobre los muebles:

- Nombre
- Tipo de mueble
- Fuente
- Tamaño
- Si se puede pedir
- Imagen
- Variaciones
- Precio
- Precio Millas

IRQ-010	Información sobre Eventos
El sistema deberá almacenar y mostrar los siguientes datos sobre los eventos: • Nombre • Horas • Descripción • FechaHN • FechaHS	

5.2.2. Requisitos Funcionales

FRQ-001	Registro
El sistema deberá permitir darse de alta en la página	

FRQ-002	Catálogo Peces
El sistema deberá permitir el acceso a un catálogo de Peces	

FRQ-003	Catálogo Bichos
El sistema deberá permitir el acceso a un catálogo de Bichos	

FRQ-004	Catálogo Criaturas marinas
El sistema deberá permitir el acceso a un catálogo de Criaturas marinas	

FRQ-005	Catálogo Fósiles
El sistema deberá permitir el acceso a un catálogo de Fósiles	

FRQ-006	Catálogo Arte
El sistema deberá permitir el acceso a un catálogo de Obras de Arte	

FRQ-007	Catálogo Muebles
El sistema deberá permitir el acceso a un catálogo de Muebles	

FRQ-008	Catálogo Ropa
El sistema deberá permitir el acceso a un catálogo de Ropa	

FRQ-009	Catálogo Vecinos
El sistema deberá permitir el acceso a un catálogo de Vecinos	

FRQ-010	Calendario de Eventos
El sistema deberá permitir el acceso a un calendario de Eventos	

FRQ-011	Catálogo Canciones
El sistema deberá permitir el acceso a un catálogo de Canciones	

FRQ-012	Criaturas en tiempo real
El sistema deberá permitir ver las criaturas disponibles para atrapar en tiempo real	

FRQ-013	Calculador de nabos
El sistema deberá permitir el cálculo del patrón de venta de nabos	

FRQ-014	Diseños de la comunidad
El sistema deberá permitir la búsqueda de diseños hechos por la comunidad	

FRQ-015	Catálogo de códigos de sueño
El sistema deberá permitir el acceso a un catálogo de códigos de sueño que hayan sido subido por los usuarios	

FRQ-016	Acceso Perfil
El sistema deberá permitir el acceso a un perfil propio	

FRQ-017	Edición Perfil
El sistema deberá permitir la edición del perfil propio	

FRQ-018	Visitantes semanales
El sistema deberá permitir ver y editar los visitantes semanales de mi isla	

FRQ-019	Tareas
El sistema deberá permitir ver, añadir, editar y borrar tareas diarias a elección del usuario	

FRQ-020	Multimedia
El sistema deberá permitir subir y borrar fotos y vídeos de la isla del usuario	

FRQ-021	Avance colección Peces
El sistema deberá permitir ver el avance personal de la colección de Peces, así como los que faltan por obtener	

FRQ-022	Avance colección Bichos
El sistema deberá permitir ver el avance personal de la colección de Bichos, así como los que faltan por obtener	

FRQ-023	Avance colección Arte
El sistema deberá permitir ver el avance personal de la colección de Obras de Arte, así como las que faltan por obtener	

FRQ-024	Avance colección Canciones
El sistema deberá permitir ver el avance personal de la colección de Canciones, así como las que faltan por obtener	

FRQ-025	Avance colección Fósiles
El sistema deberá permitir ver el avance personal de la colección de Fósiles, así como los que faltan por obtener	

FRQ-026	Avance colección Criaturas marinas
El sistema deberá permitir ver el avance personal de la colección de Criaturas marinas, así como las que faltan por obtener	

FRQ-027	Avance colección personal
El sistema deberá permitir registrar el avance personal de las colecciones	

FRQ-028	Probabilidad mudanza
El sistema deberá permitir calcular la probabilidad de que mis vecinos se muden de la isla	

FRQ-029	Restricción avance colecciones
El sistema no deberá permitir que un usuario modifique el avance de las colecciones de otro usuario	

FRQ-030	Restricción perfil
El sistema no deberá permitir que un usuario modifique el perfil de otro usuario	

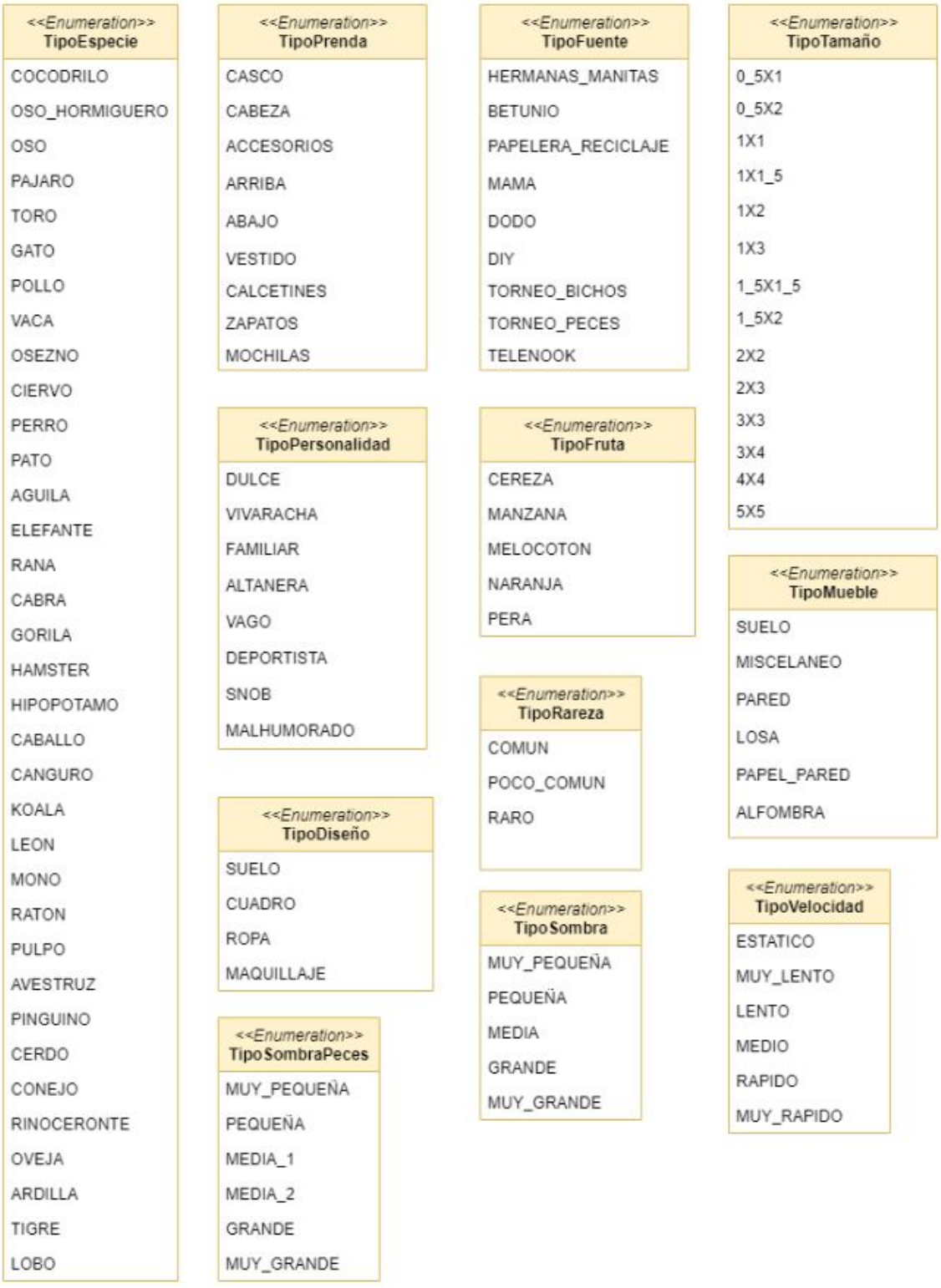
5.2.3. Requisitos no Funcionales

NFRQ-001	Tiempo de respuesta
El sistema deberá tener un tiempo medio de respuesta inferior a 1 segundo	

NFRQ-002	Disponibilidad
El sistema deberá estar disponible las 24 horas del día	

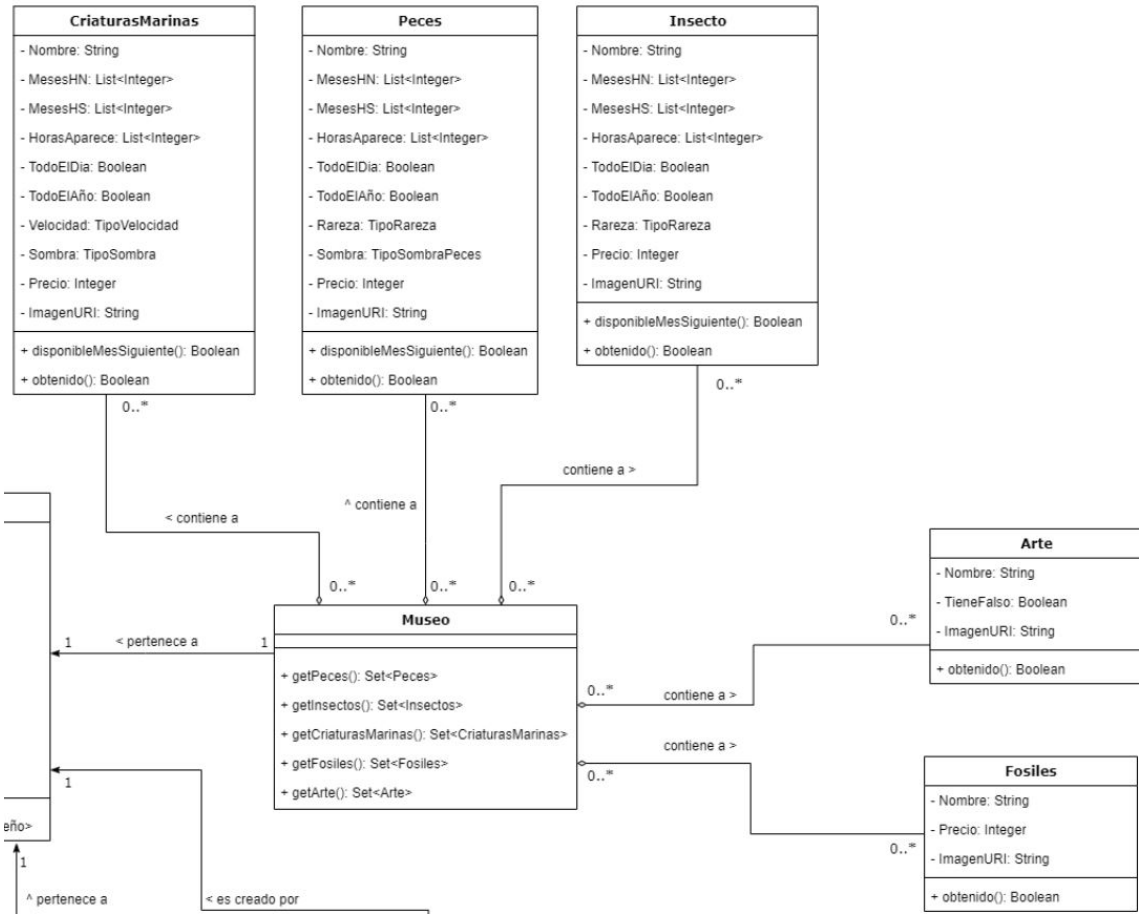
5.3– Diagrama de Clases

Dado que tratamos con grandes cantidades de información, gran parte de ella la tenemos que obtener a través de APIs. Así mismo, como sed trata de información de un videojuego, muchos datos han de aparecer en forma de enumeraciones.

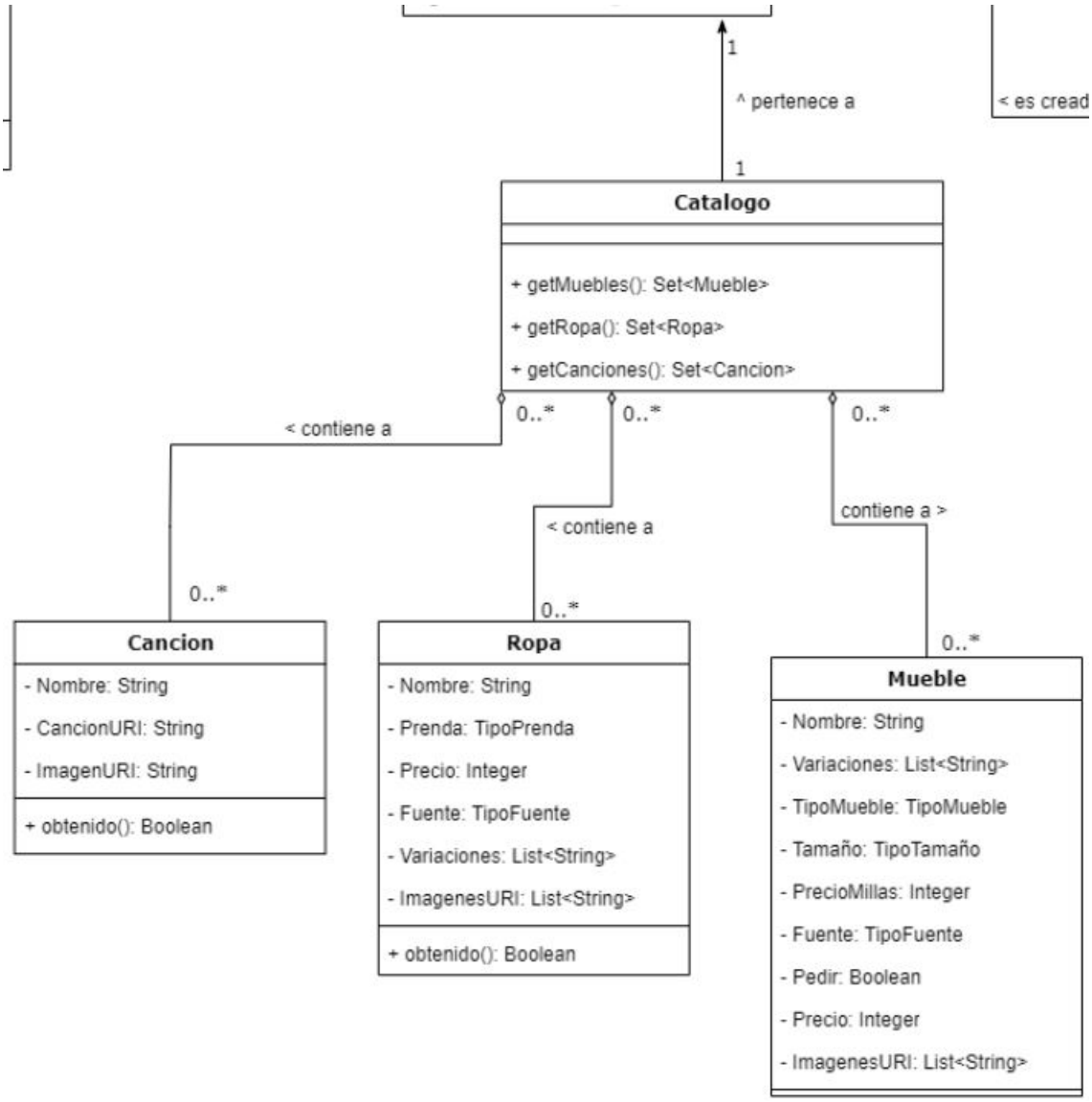


Debido al gran tamaño del diagrama de clases, vamos a verla dividida en partes más pequeñas de forma que muestren distintas relaciones y se pueda ver de forma más clara.

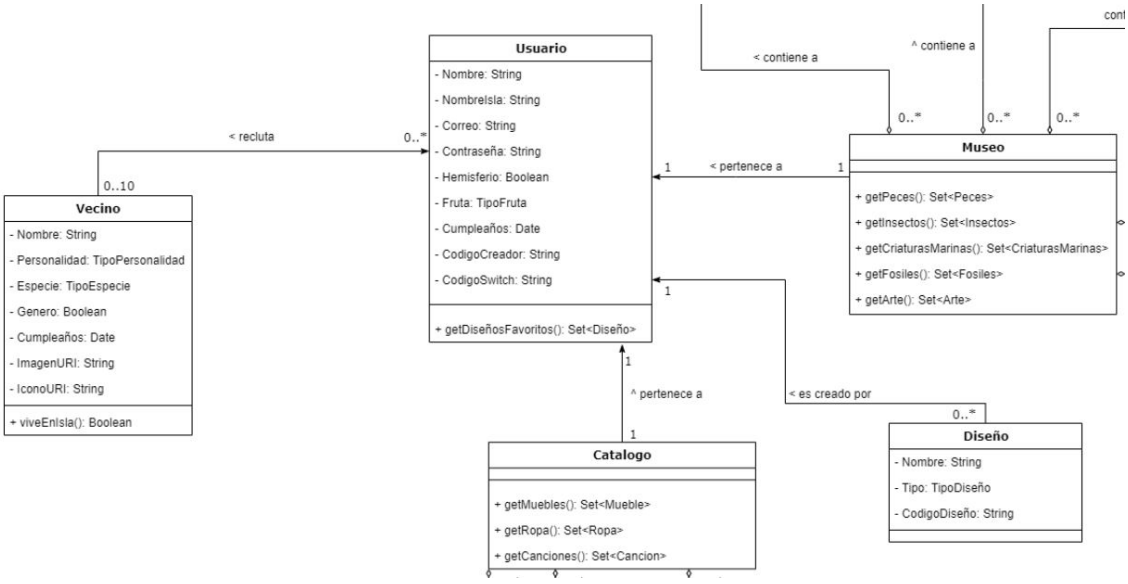
A continuación se puede ver la clase Museo, la cual recoge la colección personal de Insectos, Peces, Criaturas Marinas, Fósiles y Obras de Arte.



Luego tenemos la clase Catálogo que de manera análoga hace lo mismo con la Ropa, los Muebles y las Canciones.



Tanto Museo como Catálogo van asociados a Usuario, así como Vecinos y Diseños.



Por último está la clase Evento que no está relacionada con ninguna otra, ya que solo se

usarán los datos para el Calendario de Eventos el cual es meramente informativo.

5.4– Diseño

A continuación ofrecemos una primera visión de lo que sería la interfaz gráfica del sistema, acompañado de algunas explicaciones para entender a priori el funcionamiento del sistema, ya que entraremos en detalle más adelante en el manual de usuario. En las imágenes aparecen anotaciones realizadas en rojo para el mejor entendimiento del estilo que se busca obtener de cara a la hora de implementar las vistas.

Empezando por la página de inicio (véase la figura 5.1), tendríamos el navegador en la zona superior para acceder a las diferentes secciones del sistema que no necesitan que el usuario se haya registrado, además de un botón para iniciar sesión en caso de que se disponga de una cuenta en la página.

Bajando vemos diferentes secciones, disponiendo en primer plano de la portada con un botón para acceder al registro. Más abajo se encuentran las diferentes secciones de enciclopedia con las que contará el sistema: peces, bichos, criaturas marinas, fósiles y obras de arte (véase la figura 5.1), así como del catálogo de muebles, ropa, vecinos e incluso un calendario de eventos (véase la figura 5.2).

Empezando por las secciones que hemos comentado en el párrafo anterior, a continuación tenemos un listado de peces, aunque el formato es genérico para los demás listados, cambiado algunos pequeños detalles en cada uno como ciertos filtros.

Podemos observar que arriba se encuentra la sección de filtros para que el usuario pueda realizar una búsqueda más específica de manera simple. Debajo de los filtros se encuentra el listado en sí, que muestra una recopilación de los peces (en este caso) disponibles en el juego. Cada item cuenta con una imagen, su nombre y algunos de los datos más relevantes, como precio de venta o localización, y si se hace clic en la celda se abre un menú con información más detallada, de forma que lo importante esté siempre a la vista y no haya que entrar en los menús de forma repetitiva.

En rojo se muestran aquellas criaturas que no se encontrarán disponibles el mes que viene, y además, si el usuario está registrado en el sistema y ha iniciado sesión, puede llevar recuento

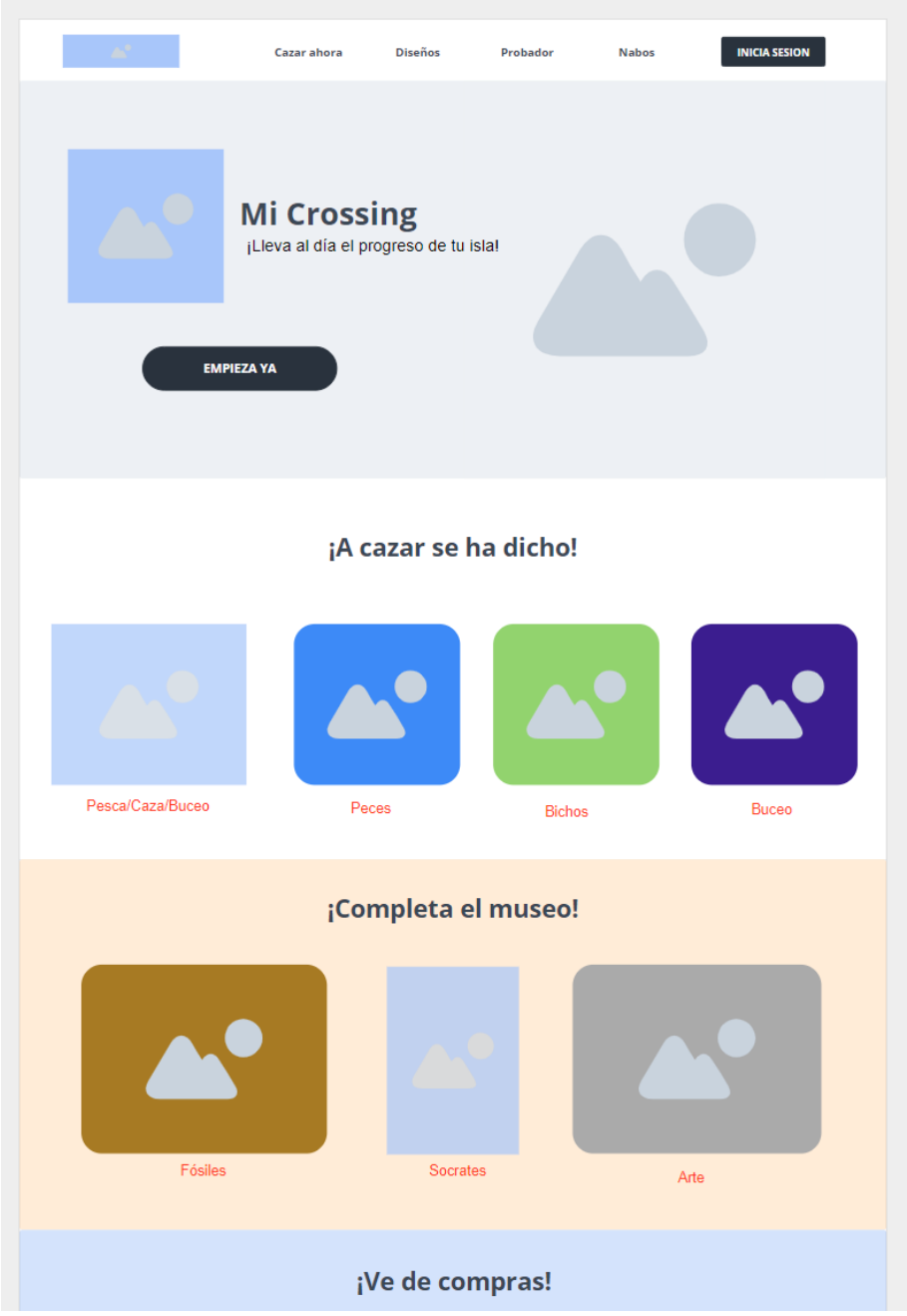


Figura 5.1: Inicio 1

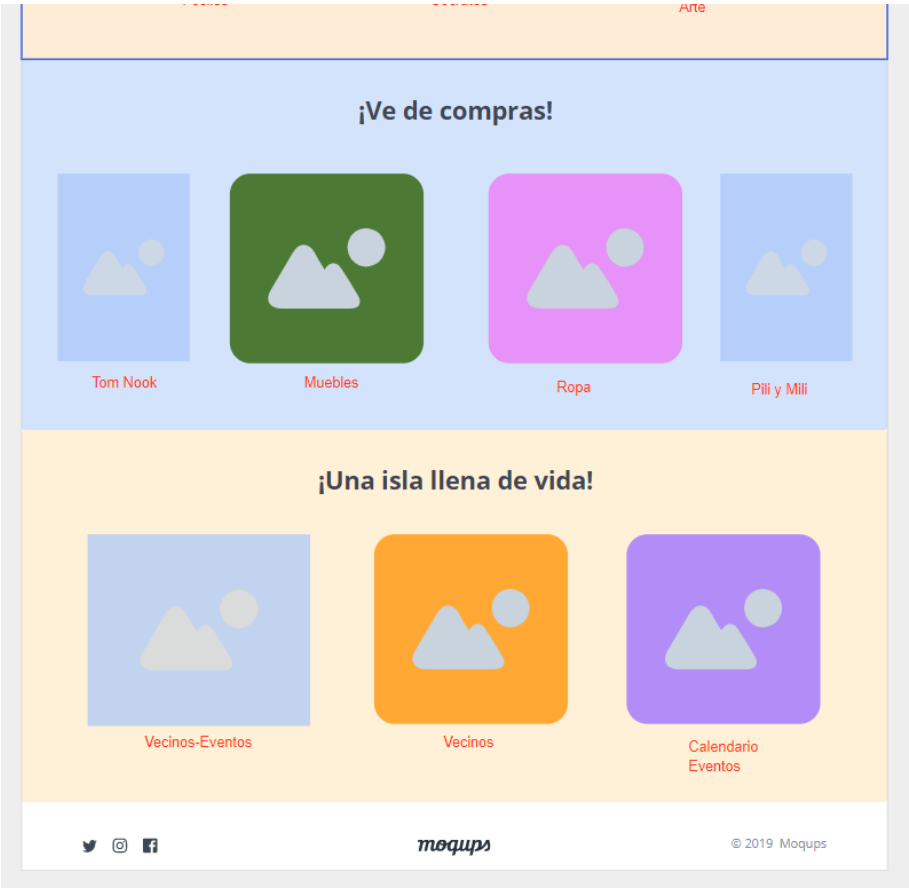


Figura 5.2: Inicio 2

de los que ya ha capturado marcándolos con el tick superior derecho en cada item, cambiándose así el fondo a un color distinto para diferenciarlos a simple vista (véase la figura 5.3).

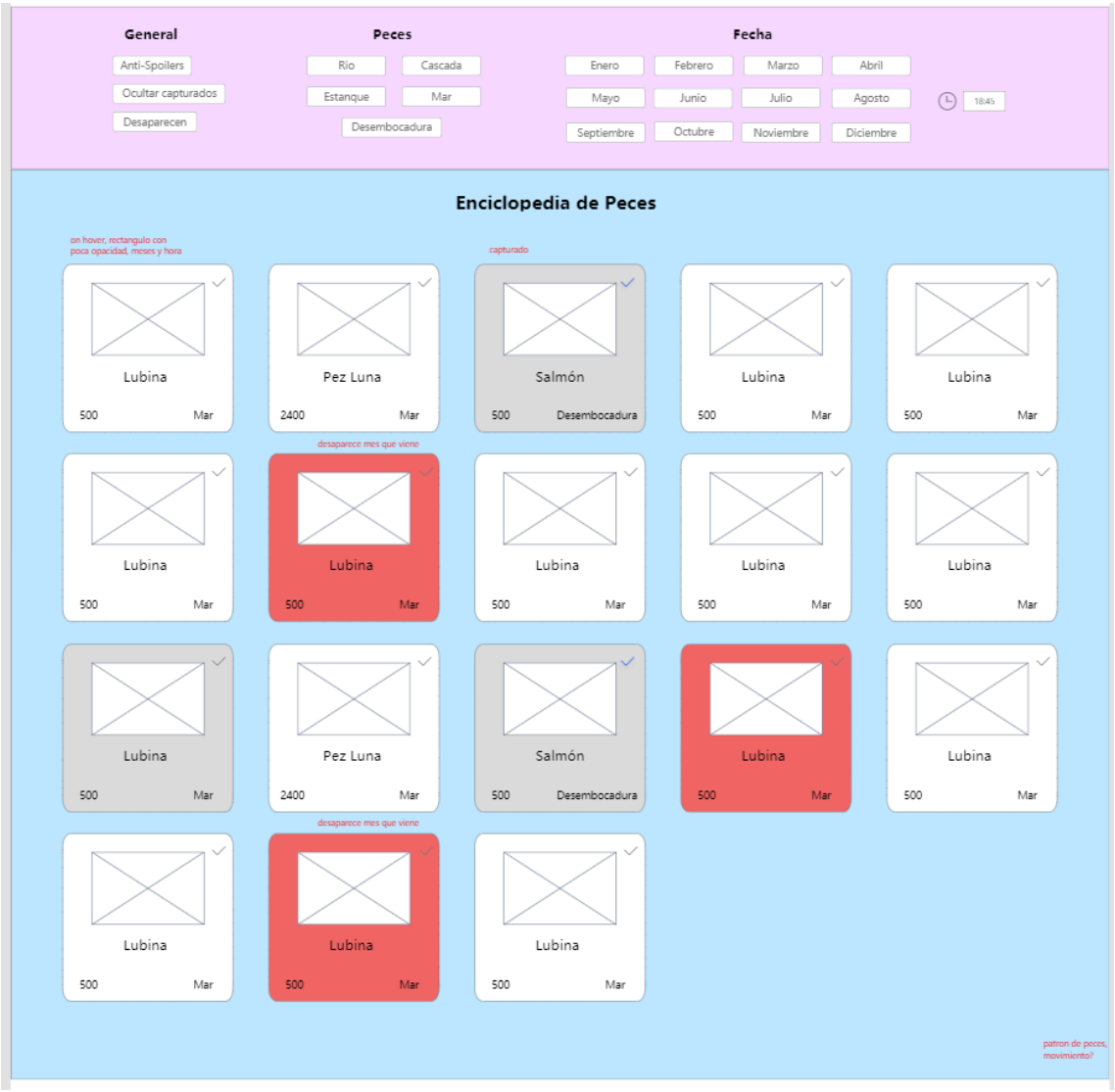


Figura 5.3: Listado genérico

Para terminar con el contenido de la página de inicio (ya que a excepción del calendario, lo demás son todo listados como el que acabamos de ver), tendríamos el calendario de eventos (véase la figura 5.4).

Esta sección no es mas que un gran calendario donde el usuario puede ver tanto las festividades del juego como los cumpleaños de los vecinos. Sin embargo, dispondrá de información adicional para los eventos al hacer clic en ellos, de forma que se desplegará un menú (véase la figura 5.5) en el que podremos ver información útil de dicho evento, como visitantes, objetos temáticos, actividades especiales, etc. De esta forma el usuario puede tener una idea general de todo lo que se puede hacer en ese evento y planificarse mejor su agenda.

Una vez acabado con el contenido de la página principal, si el usuario quisiera registrarse accedería a la siguiente vista (véase la figura 5.6), la cual es una simple página de registro con

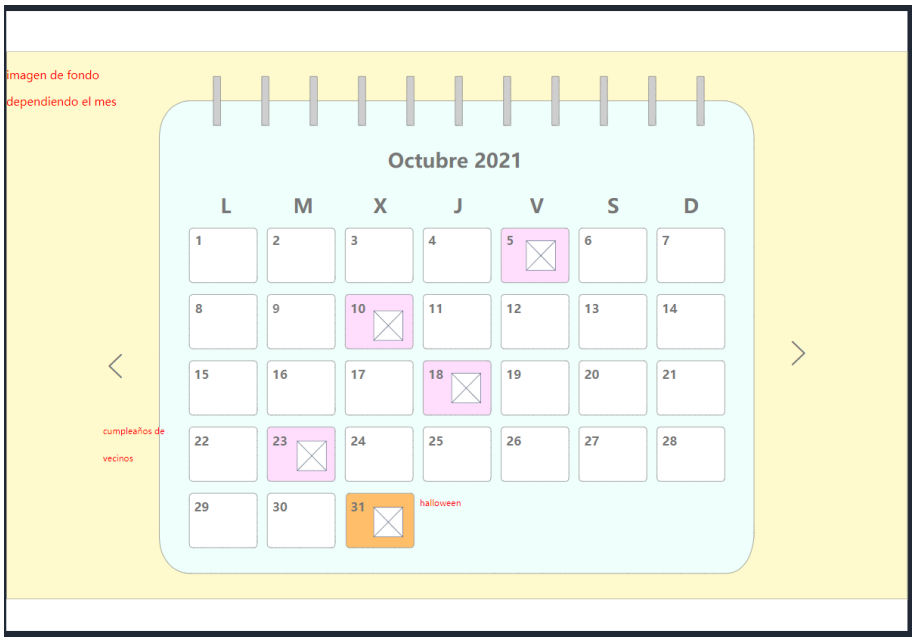


Figura 5.4: Eventos

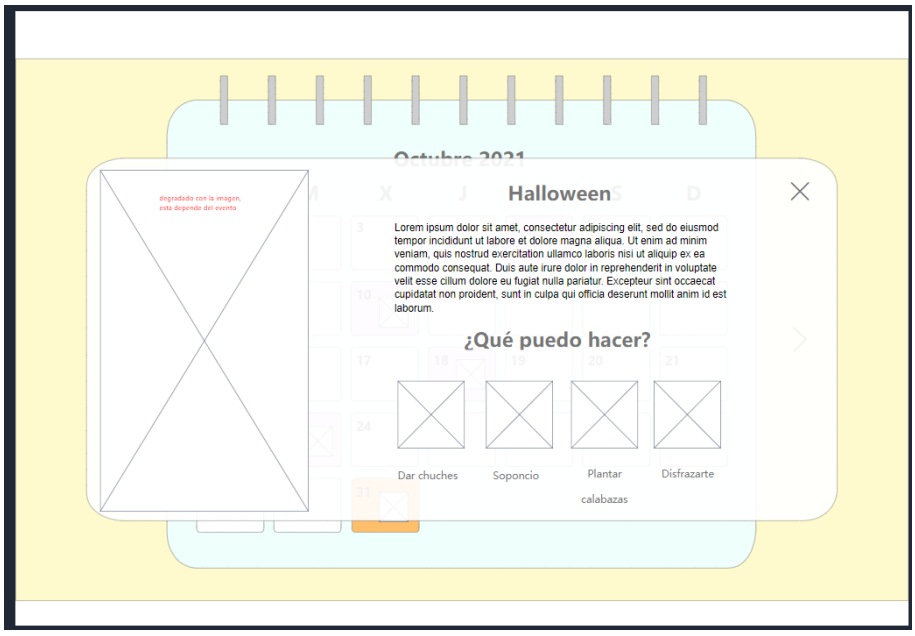


Figura 5.5: Eventos - Detallado

información relevante al juego, como el nombre de la Isla, el tipo de fruta o la localización. Este último es bastante importante ya que dependiendo de si se encuentra en el hemisferio norte o sur, el usuario visualizará unos datos u otros.

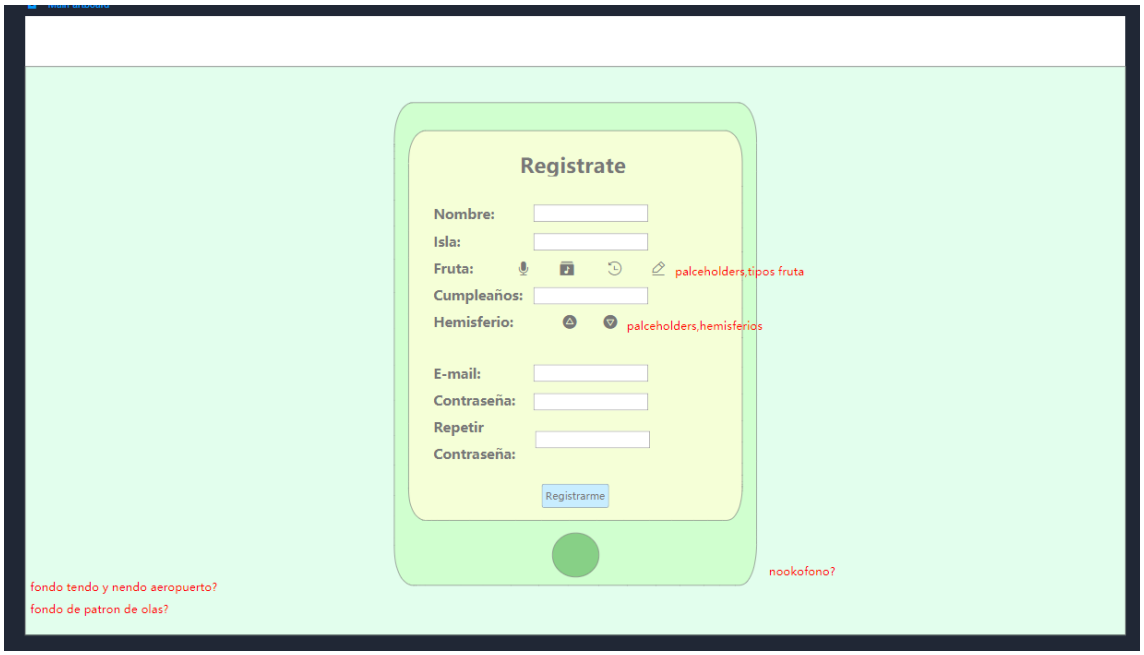


Figura 5.6: Registro

Si se completa el registro de forma satisfactoria, se puede acceder al perfil del usuario, el cual dispone de varias secciones (véase la figura 5.7).

A la izquierda se muestra la información del usuario a modo de resumen o biografía. A la derecha se encuentra la sección que se usará de forma diaria. Por una parte tenemos las tareas, las cuales se pueden editar con un icono dependiendo del propósito que tengan y serán marcadas por el usuario una vez hayan sido realizadas, reiniciándose cada día. Por otra parte tenemos la lista de visitantes semanales, donde podremos observar los visitantes de la semana anterior e ir rellenando los de esta semana en consecuencia.

Si bajamos, encontraremos un listado de los vecinos que actualmente residen en nuestra isla, con posibilidad de ir añadiendo hasta un máximo de diez vecinos. Al colocar el cursor sobre cualquiera de ellos, se mostrará información sobre el mismo.

Más abajo tenemos un pequeño listado de algunas colecciones de objetos especiales para que el usuario lleve un recuento de las que ya dispone y las pueda localizar de una manera más sencilla y organizada. Las colecciones irán variando dependiendo de que botón se encuentre activo, y dispondrá de un recuadro de texto para realizar una búsqueda específica.

Por último, tenemos una sección de galería que sirva a modo de portfolio para el usuario, para tener todas las capturas y/o vídeos recopilados en un mismo sitio.

Una vez acabado con el perfil, pasamos a las tres últimas vistas que se encuentran en el navegador principal de la página. Primero encontramos una página que es una recopilación de listados (como el mencionado anteriormente) de bichos, peces y criaturas marinas. La diferen-

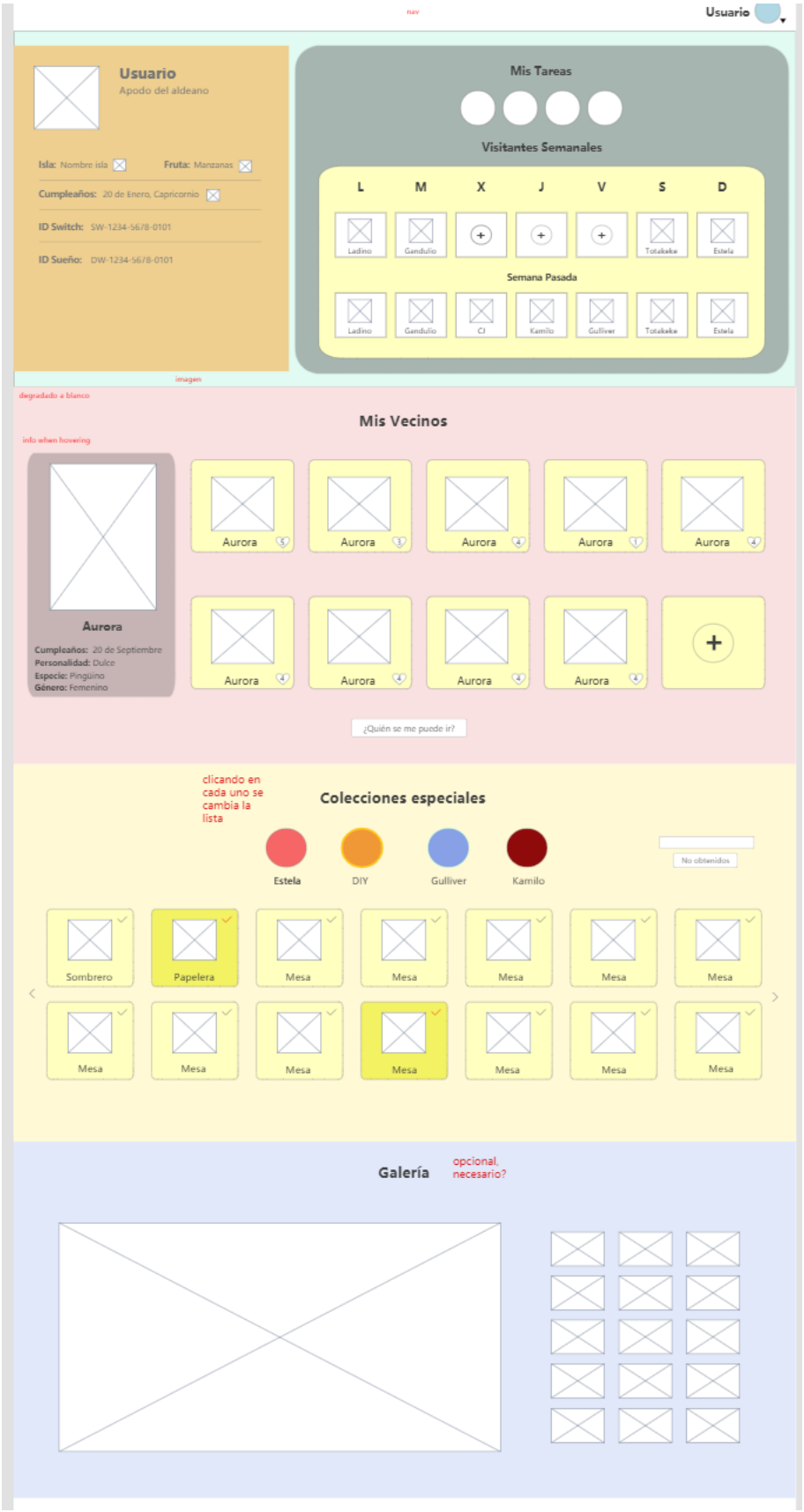


Figura 5.7: Perfil

cia, aparte de encontrarse los tres juntos, es que en este listado aparecen todas las criaturas que se pueden atrapar actualmente, actualizándose a medida que lo hace el tiempo (véase la figura 5.8). Esto es útil ya que si un usuario quiere saber que criaturas puede cazar ahora en su isla, tan solo tiene que acceder a esta página y ya dispone de dicha información, sin registro ni pérdida de tiempo, ya que dispone de las tres colecciones de criaturas en la misma página.

Además, también dispone de una serie de filtros para realizar una búsqueda algo más específica, incluso con opciones de ocultar cierta información para evitarse así los "spoilers", de forma que sepa donde puede encontrar la criatura pero no sepa ni cuál ni cómo es, de esta forma puede tener una cierta ayuda a la hora de capturarla pero seguir teniendo la emoción de no saber que es lo que le aguarda.

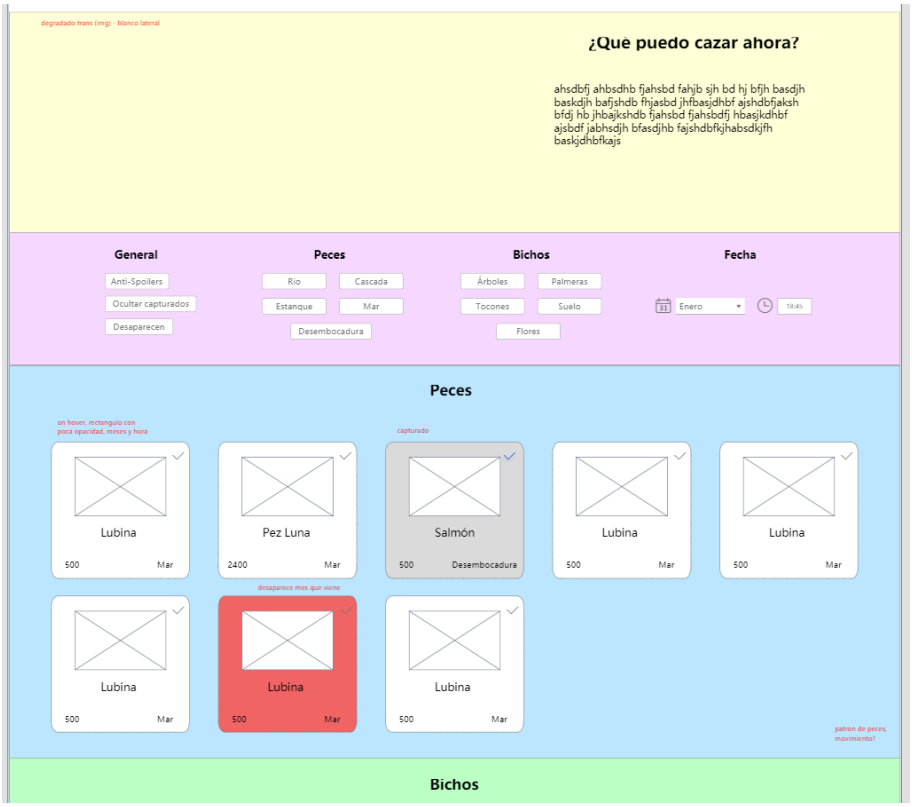


Figura 5.8: Cazar ahora 1

A continuación tendríamos la calculadora de nabos. En esta página se proporciona un poco de información acerca del mercado de nabos dentro del juego, de forma que el usuario entienda algo mejor como funciona y los distintos patrones de venta que existen (véase la figura 5.10).

En la parte inferior se encontraría la calculadora en sí, donde el usuario debe de escribir la información que haya ido recopilando para poder así realizar una predicción del patrón que hay actualmente en su isla. Además se acompañará de una gráfica y una tabla de precios por si quisiera información más detallada.

Para terminar con este apartado y con el capítulo, por último veremos la vista del probador. Esta es una de las funcionalidades que crearemos y que consiste, como su propio nombre indica, en un probador de ropa (véase la figura 5.11).

Figura 5.9: Cazar ahora 2

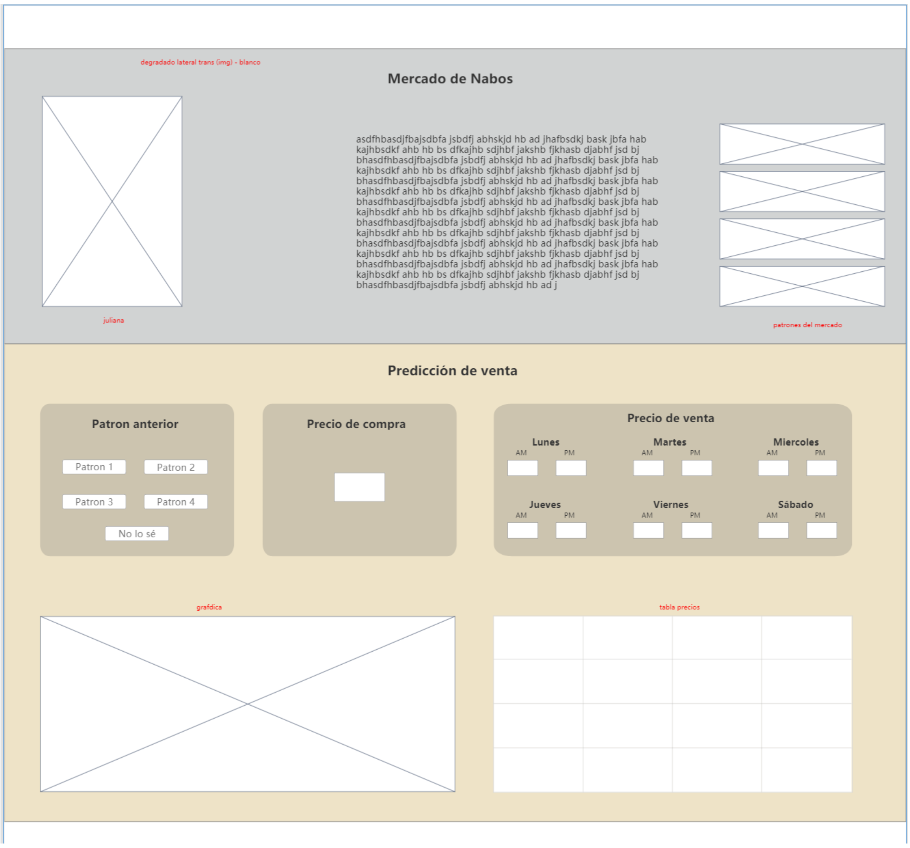


Figura 5.10: Calculadora de Nabos

En el centro tendremos una imagen de nuestro personaje, al cual podremos personalizar utilizando el menú superior, actualizándose a medida que seleccionemos las distintas opciones. Una vez elegido el personaje, la página dispondrá de un catálogo de ropa dividido en secciones (cabeza, torso, accesorios etc) para que el usuario pueda buscar la prenda que quiera probarse. Una vez la encuentre, con un clic sobre la prenda, esta aparecerá sobre el personaje central. Además se mostrarán en un desplegable todas las variaciones de color de las prendas para obtener así un catálogo completo. De esta forma el usuario puede realizar combinaciones de ropa para probar nuevos estilos y decidir que prendas necesita, en vez de ir comprándolas en el juego para luego cambiar de opinión.

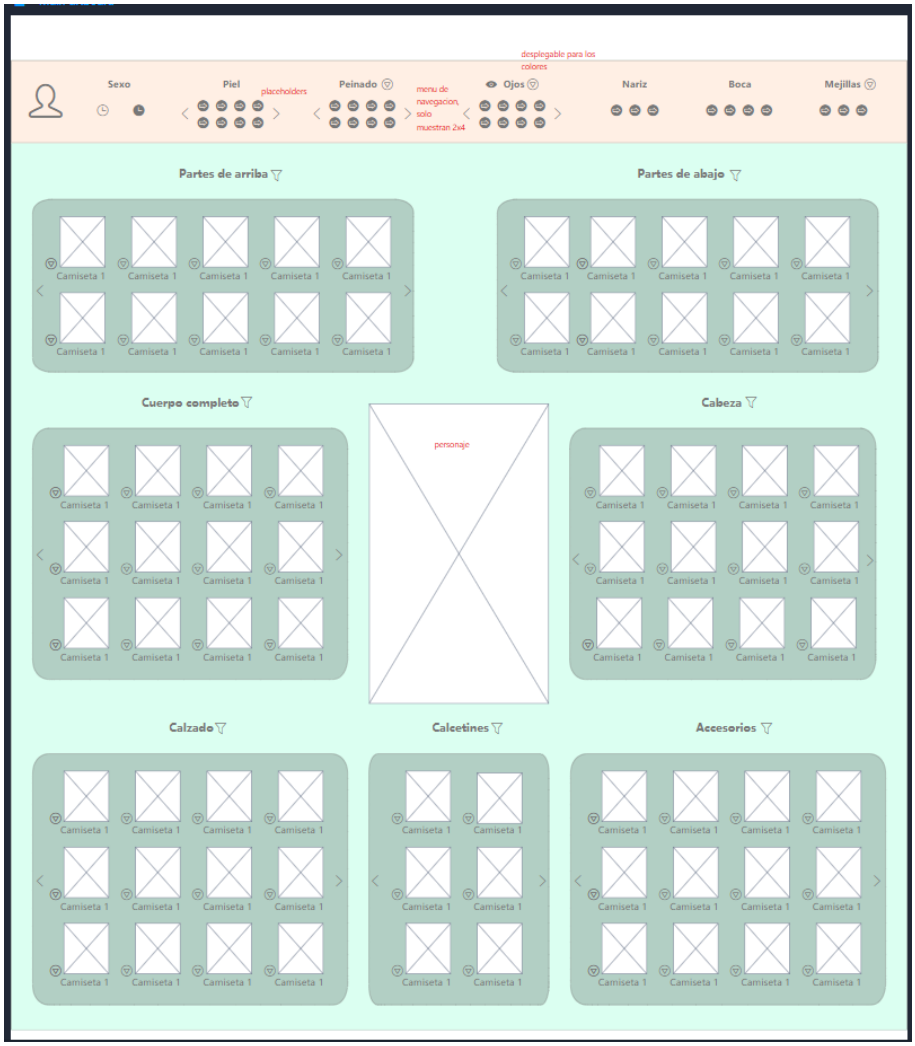


Figura 5.11: Probador

CAPÍTULO 6

Implementación

6.1– Entorno de desarrollo

Figura 6.1: Visual Studio Code

El entorno de desarrollo utilizado para estra proyecto ha sido Microsoft Visual Studio Code, ya que es un IDE bastante versátil que soporta una gran variedad de lenguajes y que, al ser de código abierto, presenta la posibilidad de instalar plugins creados por la comunidad para obtener funcionalidades útiles para cualquier tipo de lenguaje. Dado que vamos a trabajar con Angular, con la instalación de un paquete de plugins de Angular hemos podido trabajar de forma bastante cómoda y sin ningún problema.

Además, ofrece soporte para Git, de forma que se pueden realizar operaciones en el repositorio de forma bastante sencilla, así como gestionar lo posibles errores que puedan surgir mediante la misma interfaz, por lo que resulta bastante cómodo.

La principal ventaja de Visual Studio Code es que, al ser un IDE universal, contamos con varias funciones generales de los IDE (debug, consola de comandos) pero aplicado a una gran variedad de lenguajes, de forma que con el simple hecho de tenerlo instalado ya se puede comenzar a trabajar (incluso sin la necesidad de plugins, aunque no es recomendable). Es por esto que con el simple hecho de disponer del IDE y descargar un par de plugins, ya se puede trabajar de forma cómoda, y además se puede adaptar de manera sorprendente ya que, si en algún momento hay que trabajar con otro lenguaje, se puede realizar perfectamente sin tener que descargar otro IDE. Directamente desde Visual Studio Code y con algun plugin extra, ya se puede incorporar el nuevo lenguaje y trabajar de forma cómoda con ambos, lo cual centraliza el trabajo en una misma aplicación y resulta más eficiente.

6.2– Tecnologías utilizadas

La idea principal de este proyecto era usar Angular como tecnología para el front-end por las razones ya explicadas anteriormente. Las demás tecnologías que fueramos a usar no iban a tener tanto peso para nosotros como lo iba a tener esta, por lo que ibamos con la mente abierta dispuestos a usar la tecnología que, además de adaptarse bien a nuestro proyecto, fuera compatible con Angular. Al principio, pensamos en usar una base de Spring para el back-end, ya que es un framework que hemos visto en la carrera y además hemos adquirido experiencia en Java durante todo la carrera, pero debido a que no sabíamos nada sobre el funcionamiento de Angular (y todo lo que puede llegar a abarcar), pensamos que el peso de la aplicación se la llevaría el back-end (en Spring al principio). Sin embargo, una vez aprendimos y nos pusimos a dar nuestros primeros pasos en el framework, descubrimos que se puede realizar la gran mayoría de funcionalidades de la aplicación en Angular.

Además luego llegó la hora de conectar el proyecto en Angular con la base de datos. Estábamos acostumbrados a Spring, que realiza toda la gestión de base de datos de forma .ºculta.en el mismo framework y tan solo es necesario inicializar los repositorios con las querys, así como las beans para popular la base de datos. Sin embargo, al estar trabajando con Angular, tuvimos que buscar información sobre como realizar esta conexión y descubrimos que se podía conectar a una base de datos gestionada mediante PHP con una simple petición HTTP al archivo que realizase la conexión y las operaciones oportunas. Esta idea nos gustó bastante ya que, no solo tendríamos que usar otro lenguaje (por lo que aprendemos más), sino que además podíamos utilizar la conexión con la base de datos a modo de API, haciendo peticiones HTTP desde Angular al correspondiente archivo PHP, el cual se encarga de abrir la conexión con la base de datos, realizar la consulta oportuna, y devolver los datos, los cuales Angular se encarga de recoger, transformarlos en caso de ser necesario, y finalmente, mostrarlos.

Es por esto que decidimos entonces usar XAMPP, ya que ofrece un servidor de Apache (en el cual se ejecutarían los archivos PHP que se encargasen de conectarse con la base de datos) así como una base de datos de MariaDatabase. Esto nos vino particularmente bien ya que en otras asignaturas habíamos usado XAMPP, y no solo eso sino que la base de datos de MariaDatabase esta basada en MySQL, que es el lenguaje que hemos dado durante toda la carrera, por lo que al estar trabajando ya con dos lenguajes que no conocemos del todo, uno familiar se recibe bastante bien.

Una vez montada la estructura de la base de datos y habiendo probado ya el funcionamiento de Angular y todo su alcance, nos dimos cuenta de que el back-end de Spring que habíamos pensado no era para nada necesario, ya que toda la lógica se realiza a través de este framework y las operaciones para obtener datos de la base de datos las íbamos a realizar con PHP, por lo que al final se descartó la idea de usar Spring como framework para el back-end ya que contábamos con todo lo que necesitábamos con la estructura que disponíamos.

6.2.1. Angular

Figura 6.2: Angular

Angular es un framework para aplicaciones web desarrollado en TypeScript de código abierto que suele ser utilizado para la creación de aplicaciones web de una sola página (SPA), además de seguir el diseño Modelo-Vista-Controlador (MVC).

Angular funciona mediante componentes, los cuales disponen de sus respectivas variables y métodos que se encargan de realizar la lógica, así como de operar con su vista asociada. De esta forma, obtenemos pequeños paquetes de código, cada uno con su vista y su lógica. Podría explicarse como si de un puzzle se tratase, pero con muchas posibles soluciones, siendo cada pieza un componente con su vista, sus variables y métodos. Cada pieza es funcional por si misma, pero el objetivo es juntarlos con otros componentes para obtener la aplicación (o vista) completa.

Además de ser una forma bastante interesante de desarrollar una aplicación, es una muy buena forma de evitar la repetición de código, ya que si se necesita reusar un componente es tan fácil como importarlo y se encuentra listo. De la misma forma, se trabaja con servicios, directivas y etiquetas que pueden ser reutilizadas, reduciendo así la carga de trabajo considerablemente y aumentando la limpieza del código.

Además, una vez se esté ejecutando la aplicación, esta se actualiza con los nuevos cambios que se realicen, algo que resulta bastante cómodo a la hora de realizar algunas pruebas dado que no hay que estar ejecutando constantemente la aplicación.

6.2.2. XAMPP y Apache

Figura 6.3: XAMPP

XAMPP es un paquete de software libre que consiste en el sistema de gestión de bases de datos de MySQL, el servidor web Apache y los intérpretes para PHP y Perl.

Es una opción bastante buena para nuestro proyecto ya que resulta bastante sencillo de usar y ademas nos proporciona tanto el servidor web como la base de datos y el interprete para PHP, todo desde una misma aplicación.

Figura 6.4: Servidor Web Apache

El servidor web de Apache es un servidor HTTP de código abierto multiplataforma con una arquitectura basada en módulos. Tiene una sección base que actúa como núcleo, y a esta base se le adjuntan módulos que aportan funciones básicas para el servidor web. Además, existen módulos externos para ampliar su funcionalidad, ya sea ofreciendo soporte para distintos lenguajes (Perl, Python, PHP y Ruby entre otros) u ofreciendo otras funciones como llevar un control del tráfico web.

6.2.3. PHP

6.2.4. MariaDatabase

6.2.5. Git

CAPÍTULO 7

Pruebas

CAPÍTULO 8

Comparación con otras alternativas

CAPÍTULO 9

Manual

Conclusiones y desarrollos futuros

Apéndices

Referencias

AUTORES, Varios. «Escuela Técnica Superior de Ingeniería Informática.», 2014. Fecha de consulta: 24 de Noviembre de 2014, URL `http://www.informatica.us.es`.

BEZOS, Javier. «The titlesec and titletoc Packages.» *TexEmplares*, 8, (2007), 283–298. CervanTEX, 2007.

DE SOUSA, José Martínez. *Ortografía y ortotipografía del español actual*. Trea, 2004.