Memoria del proyecto

Plataforma de apoyo en el estudio de algoritmia

Trabajo de Fin de Grado

INGENIERÍA INFORMÁTICA

VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

Julio de 2022

Autor

Fausto Sánchez Hoya

Tutor/a

Gabriel Villarrubia González

Juan Francisco De Paz Santana

Héctor Sánchez San Blas

Certificado de los tutores

D. Gabriel Villarrubia González, D. Juan Francisco de Paz Santana y D. Héctor Sánchez San Blas, profesores del Departamento de Informática y Automática de la Universidad de Salamanca.

Hacen constar:

Que el trabajo titulado “Plataforma de apoyo en el estudio de algoritmia” ha sido realizado por D. Fausto Sánchez Hoya con DNI 70937831N y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado de la Titulación Grado de Ingeniería Informática de esta esta Universidad.

Y para que así conste a todos los efectos oportunos.

En Salamanca a 4 de Julio de 2022

VILLARRUBIA GONZALEZ GABRIEL - 71932845T

Firmado digitalmente por VILLARRUBIA GONZALEZ GABRIEL - 71932845T
Fecha: 2022.07.07 12:14:43 +02'00'

D. Gabriel Villarrubia González

SANCHEZ SAN BLAS HECTOR - 70901148Z

Firmado digitalmente por SANCHEZ SAN BLAS HECTOR - 70901148Z
Fecha: 2022.07.04 21:02:00 +02'00'

D. Héctor Sánchez San Blas

DE PAZ SANTANA JUAN FRANCISCO - 76125754D

Firmado digitalmente por DE PAZ SANTANA JUAN FRANCISCO - 76125754D
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-76125754D, givenName=JUAN FRANCISCO, sn=DE PAZ SANTANA, cn=DE PAZ SANTANA JUAN FRANCISCO - 76125754D
Fecha: 2022.07.05 10:27:51 +02'00'

D. Juan Francisco de Paz Santana

Resumen

Actualmente muchas de las empresas top en el mundo de la informática incorporan test de programación en sus pruebas de evaluación de candidatos. Estos test de programación suelen consistir en encontrar soluciones a problemas de la forma más rápida y eficiente posible por lo que en los últimos años han surgido un gran número de páginas web que permiten al usuario resolver problemas de ese tipo a modo de entrenamiento. Por otro lado, hay que tener en cuenta que estas soluciones suelen involucrar la ordenación y búsqueda en arrays de elementos, haciendo así indispensable que cualquiera que quiera aplicar a estas empresas tenga un conocimiento elevado sobre algoritmos de búsqueda y ordenación. Sin embargo, estas páginas mencionadas antes no suelen ofrecer información teórica sobre estos algoritmos y mucho menos explicaciones que involucren el seguimiento paso a paso de la ejecución de estos.

En el ámbito académico vemos como se hace hincapié en la programación óptima y eficiente mediante la enseñanza de dichos algoritmos y técnicas para medir la eficiencia. Esta enseñanza se puede ver ayudada en gran medida con el apoyo de páginas que muestren la ejecución de estos algoritmos y donde puedan ponerlos en práctica.

El objetivo de este proyecto es la realización de una aplicación web en la que se junte estos tres aspectos fundamentales: explicaciones teóricas con ejemplos de código de los principales algoritmos de ordenación y búsqueda, una plataforma de resolución de problemas con un editor de código online para que el usuario pueda practicar independientemente de la capacidad hardware de su dispositivo y, finalmente, un simulador gráfico que represente la ejecución de estos algoritmos y cuya ejecución pueda ser 100% personalizable por el usuario.

De esta forma un usuario no tiene que estar cambiando entre 3 páginas diferentes para poder prepararse de cara a una entrevista técnica o para poder tener un punto de referencia a la hora de estudiar, sino que tiene todo agrupado en una misma web.

Palabras clave: programación, algoritmos, problemas, preparación de entrevistas, formación.

Summary

Currently many of the top companies in the IT world incorporate programming tests in their candidate evaluation tests. These programming tests usually consist of finding solutions to problems as quickly and efficiently as possible, so in recent years a large number of websites have emerged that allow the user to solve such problems as training. On the other hand, it should be noted that these solutions usually involve sorting and searching arrays of elements, thus making it indispensable for anyone who wants to apply to these enterprises to have a high level of knowledge about search and sorting algorithms. However, these pages cited above do not usually provide theoretical information about these algorithms, let alone explanations involving step-by-step follow up of the execution of these algorithms.

In academia, we see an emphasis on optimal and efficient scheduling through the teaching of such algorithms and techniques for measuring efficiency. This teaching can be greatly helped with the support of pages that show the execution of these algorithms and where they can put them into practice.

The objective of this project is the realization of a web application that brings together these three fundamental aspects: theoretical explanations with code examples of the main sorting and search algorithms, a problem solving platform with an online code editor so that the user can practice regardless of the hardware capacity of his device and, finally, a graphical simulator that represents the execution of these algorithms and whose execution can be 100% customizable by the user.

In this way a user does not have to be changing between 3 different pages to prepare for a technical interview or to have a point of reference when studying, but has everything grouped on the same website.

Keywords: programming, algorithms, problems, interview preparation, training.

Tabla de contenido

Índice de ilustraciones

Índice de tablas

1. Introducción

Actualmente la mayoría de las empresas del mundo de la informática involucran en su proceso de contratación entrevistas técnicas en las que el aplicante debe resolver un problema de la forma más rápida y eficiente posibles. Para poder realizar esto cualquier usuario deberá haber entrenado, resolviendo muchos problemas similares y tener un gran conocimiento sobre algoritmia de la programación.

El objetivo de este proyecto es juntar en una misma web los tres recursos principales para la preparación de estas entrevistas técnicas: la resolución de problemas de programación online, explicaciones de los principales algoritmos de ordenación y búsqueda y la simulación gráfica de las ejecuciones de estos algoritmos para mejorar su comprensión.

Este documento está dividido en las siguientes partes:

- Aplicaciones actuales: donde se presentan las webs que existen actualmente y suplen estos problemas.
- Objetivos: donde se desarrollan los objetivos que tiene el proyecto
- Conceptos teóricos: donde se explica la terminología necesaria para entender este documento.
- Técnicas y herramientas: donde se especifica las técnicas y herramientas utilizadas para el desarrollo
- Aspectos relevantes del desarrollo: donde se hace un resumen de los momentos claves del desarrollo.
- Conclusiones y líneas de trabajo futuras: donde se especifica aquello aprendido tras la realización de este proyecto y los planes de futuro que este tiene.

Este documento y especialmente la parte de aspectos relevantes del desarrollo viene cumplimentada con la información disponible en los siguientes anexos:

- Anexo I Planificación temporal: se desarrolla la planificación de las tareas y todo lo relacionado con la asignatura de Gestión de proyectos.
- Anexo II Especificación de requisitos: se desarrollan los objetivos, actores, requisitos y casos de uso del sistema.
- Anexo III Análisis: se desarrolla la fase de análisis de requisitos

- Anexo IV Diseño del sistema: se desarrolla el diseño del sistema, la especificación más cercana al desarrollo final del proyecto.
- Anexo V Manual del programador: se especifica la funcionalidad de cada fichero, cómo extender el sistema y la localización de la documentación.
- Anexo VI Manual de usuario: se desarrollan las tareas que puede realizar un usuario y los pasos que debe hacer para conseguirlas.

2. Aplicaciones actuales

Existe una gran cantidad de páginas que presenten alguna de las funciones que presenta nuestro proyecto, pero se van a mencionar sólo las 3 principales, una por cada función.

2.1 Resolución de problemas

Para la resolución de problemas online existe una página por excelencia: HackerRank.

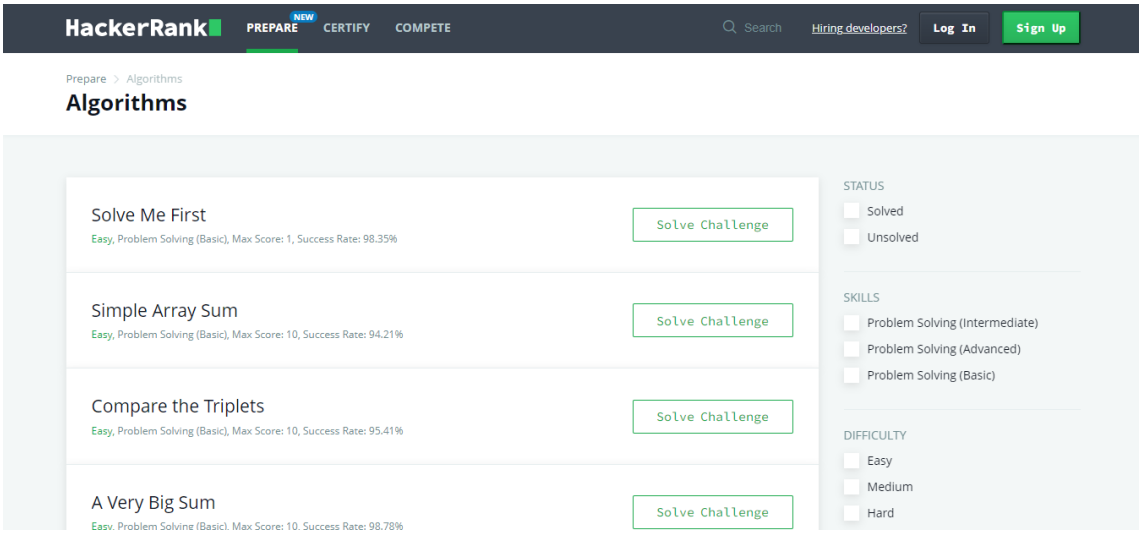


Ilustración 1 Página HackerRank

Esta página cuenta con una gran cantidad de problemas como se puede ver en la Ilustración 1. Estos problemas de programación son divididos por categorías, dificultad, etc. Además, ofrece también kits de preparación de entrevistas donde se plantean sesiones de entrenamiento mucho más comprimidas en el tiempo cuando se tiene una entrevista incipiente. Por otro lado, ofrece certificados para validar las aptitudes en cuanto a tecnologías como lenguajes de programación, frameworks, etc. Las empresas pueden también utilizar HackerRank para organizar concursos de programación que faciliten la selección de personal.

Sin embargo, HackerRank no ofrece ni explicaciones de algoritmos ni la visualización gráfica de estos.

Otro ejemplo que podemos destacar es LeetCode que, al igual que HackerRank, es una página enfocada en la resolución de problemas y organización de competiciones y entrevistas de trabajo enfocadas en la programación como podemos ver en la Ilustración 2.

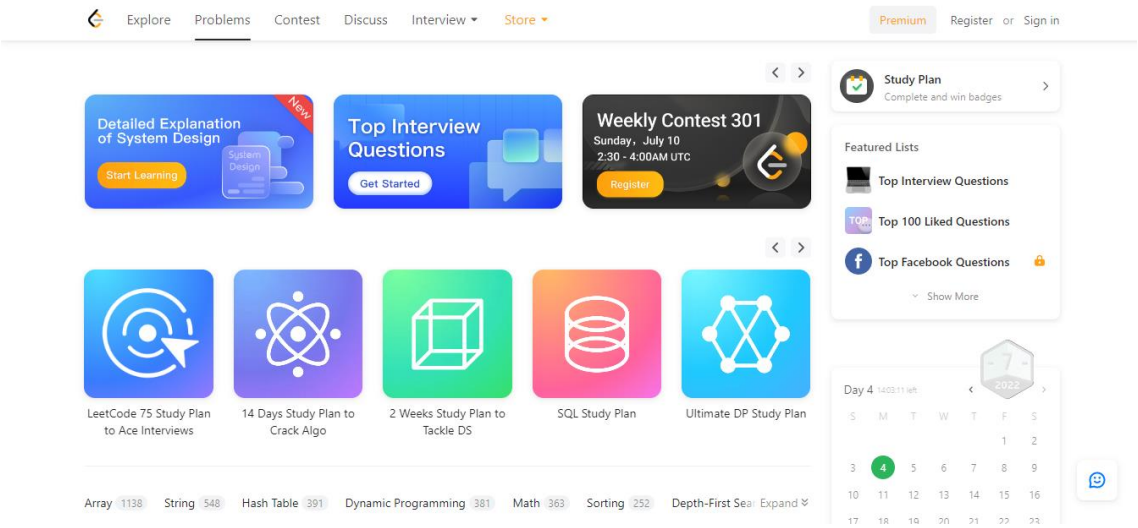


Ilustración 2 Página LeetCode

2.2 Explicación de algoritmos

La página por excelencia para aprender sobre cualquier lenguaje de programación, algoritmos, arquitecturas, etc. Es GeeksForGeeks.

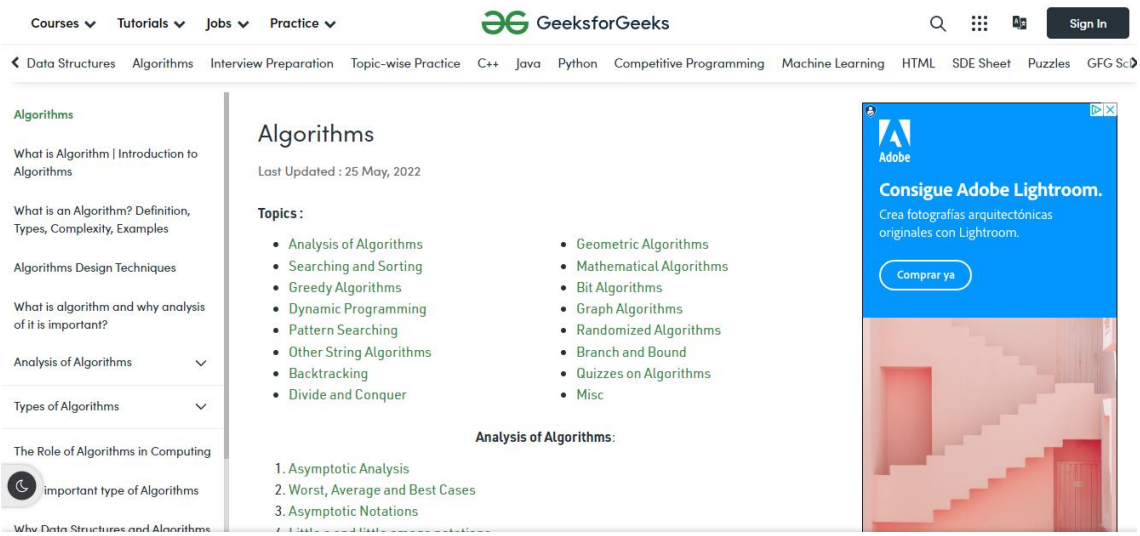


Ilustración 3 Página GeeksForGeeks

Como vemos en la Ilustración 3, esta página web es un portal con enfocado a la enseñanza de todo tipo de tecnologías, aunque también tiene secciones relacionadas con la contratación, certificaciones y resolución de problemas siendo esta última no tan desarrollada como en HackerRank.

En cuanto a la enseñanza de algoritmos presenta unas muy buenas explicaciones creadas principalmente por la comunidad en las que se suelen incluir ejemplos, código

en diferentes lenguajes y explicaciones sobre su eficiencia. Sin embargo, no presenta un simulador como tal entre sus funciones, haciendo que la parte del seguimiento de la ejecución del algoritmo caiga en el usuario.

2.3 Simulación de algoritmos

La página que más destaca en este ámbito es algorithm-visualizer.org.

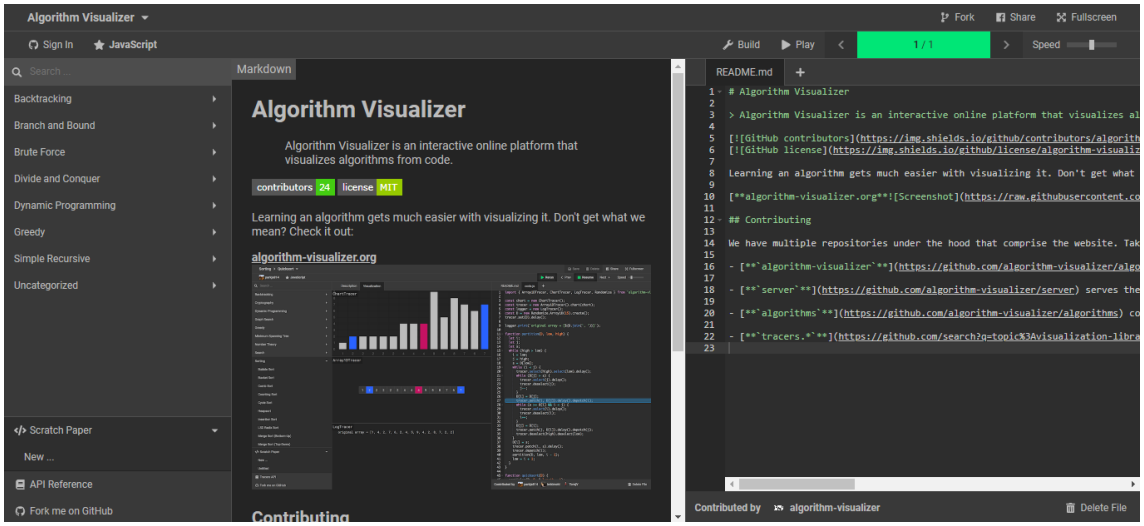


Ilustración 4 Página Algorithm-Visualizer

En ella se pueden encontrar simulaciones muy logradas de algoritmos de cualquier tipo, no sólo búsqueda y ordenación además de muy breves explicaciones sobre estos como se puede ver en la Ilustración 4. Estas simulaciones son contribuidas por una comunidad de usuarios a través de su repositorio en GitHub.

En cuanto a las simulaciones, una página que también destaca es visualgo.net . En ella podemos ver simulaciones de diferentes tipos de algoritmos, pero sin centrarse tanto en las explicaciones como en la página anterior. En la Ilustración 5 podemos ver un ejemplo de algunos de los tipos de algoritmos que simula:

Ilustración 5 Página visualgo.net

3. Objetivos

En este apartado se presentan los diferentes objetivos a cumplir en el desarrollo del proyecto.

- El sistema debe ser capaz de representar gráficamente la ejecución de un algoritmo calculando cada frame de la simulación en función de un input de elementos introducidos por el usuario. Además, este usuario podrá controlar la reproducción de esta simulación.
- El sistema debe presentar al usuario diferentes explicaciones de algoritmos de búsqueda y ordenación en la que se muestren ejemplos de su ejecución, ventajas y desventajas y su código en diferentes lenguajes de programación.
- El sistema debe presentar al usuario diferentes problemas de programación que podrá ir resolviendo y desbloqueando. Para su resolución se presenta un editor de código online que el usuario podrá utilizar sin descargar ninguna herramienta externa.
- El sistema debe ser capaz de permitir la creación de cuentas únicas para cada usuario que quiera registrarse permitiéndole acceder así a la resolución de problemas y almacenando datos sobre dicho usuario como sus estadísticas.
- El sistema debe estar optimizado para su carga en navegadores permitiendo así un mejor posicionamiento en buscadores.
- El sistema debe poder calcular rápidamente los frames de las simulaciones para evitar interrupciones en la experiencia de uso del usuario.
- La web debe contar con secciones legales, así como una política de tratamiento de cookies que el usuario puede personalizar.

4. Conceptos teóricos

En este apartado se procede a la explicación de los conceptos necesarios para el entendimiento de este documento.

4.1 UML

“UML es un lenguaje de modelado, un lenguaje visual en el que se trabaja con cajas, flechas y diagramas. Esto nos permite poder representar las ideas de cómo queremos estructurar nuestros programas de una forma mucho más visual.” [3]

Utilizamos UML para poder planificar cómo vamos a desarrollar un determinado software evitando así un código inestable.

4.2 Frontend y backend

El frontend y el backend son las dos partes principales de un determinado software.

- Frontend: es la capa visual con la que interactúa el usuario. En ella se encuentra la interfaz gráfica, así como el tratamiento de eventos como redirecciones a otras interfaces o el manejo de botones, etc.
- Backend: es la capa que no ve el usuario. En ella se lleva a cabo toda la lógica de datos y su almacenamiento.

5. Técnicas y herramientas

Para el desarrollo de este proyecto usan las siguientes técnicas y herramientas.

5.1 Desarrollo frontend

Al principio del desarrollo se ha priorizado el diseño de prototipos de la interfaz a través de recursos tan simples como el papel o más complejos como el prototipado digital usando la herramienta Adobe XD.

Más tarde en la programación se ha usado React junto con HTML y CSS junto con Typescript para el manejo de los eventos HTML. Además, también se han usado gran cantidad de imágenes que complementan la interfaz que han sido extraídas de la página https://www.flaticon.es/

Para la parte del editor online se ha usado el editor Ace en su versión para Typescript.

5.2 Desarrollo backend

Para la parte backend se ha usado también React junto con Typescript para la lógica de datos, Firebase con los servicios de autentificación para gestionar los usuarios, Firestore para el almacenamiento de datos, Hosting para la publicación, Extensions y Functions para la eliminación de los datos de los usuarios y Analytics para llevar las estadísticas de la página.

Además, también se ha utilizado la versión gratuita de Heroku junto con NodeJS para almacenar las claves privadas y redirigir las peticiones del compilador online JDoodle.

5.3 Firebase

Firebase es una plataforma de desarrollo de aplicaciones que ayuda a crear y hacer crecer aplicaciones, webs, juegos, etc. Esta plataforma está desarrollada por Google y ofrece determinados servicios evitando así el tener que desarrollarlos de 0. Algunos de ellos son:

- Authentication: este servicio se encarga de manejar los inicios de sesión, registros y verificaciones de identidad de los usuarios.
- Firestore Database: este servicio se trata de una base de datos NoSQL organizada en una estructura de colecciones-documentos.

- Hosting: este servicio permite la subida del proyecto a internet brindando además un dominio de forma gratuita con el certificado SSL ya instalado.
- Extensions: este servicio da funcionalidades ya creadas que funcionan automáticamente como, por ejemplo, la utilizada para este proyecto: la eliminación de los datos de los usuarios cuando borren su cuenta.
- Functions: complementa a extensions y se utiliza para ejecutar acciones de manera automática.
- Analytics: este servicio se encarga de conectar el software con la herramienta de estadísticas de Google Analytics.

5.4 React

"React es una biblioteca JavaScript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página." [1] El funcionamiento de React se basa en el uso de componentes que tienen una serie de estados y propiedades. Estos componentes se van renderizando a medida que se necesitan o bien cuando su estado o propiedades cambian.

5.5 Typescript

Typescript es una variante o "superconjunto" de JavaScript que le añade principalmente dos características: las variables tienen un tipo de dato y los valores sólo pueden asignarse a variables de ese tipo. De esta forma Typescript puede ofrecer otras funcionalidades como interfaces, casting, autocompletado, auto documentación, etc.

Sin embargo, los navegadores actualmente no pueden hacer uso de Typescript directamente por lo que hay que transformar el código a JavaScript antes de poder subirlo a la red. [2]

5.6 Heroku

Se trata de un servicio web que permite el alojamiento de servidores de manera gratuita hasta una cierta cantidad de recursos. Su paquete gratuito incluye 512 Mb de almacenamiento, un dominio gratuito, hasta 1000 horas de actividad y una vida del servidor de 30 min si se mantiene inactivo o hasta las 1000 horas si se mantiene activo.[5]

5.7 Otras herramientas del desarrollo

A parte de las mencionadas anteriormente se utilizan otras herramientas en el desarrollo:

- Create React App: [6] se trata de una herramienta que permite la preparación de un proyecto de React inicializando todos los archivos y añadiendo las dependencias necesarias. De esta forma el usuario no tiene que hacerlo a mano.
- Visual Studio Code: [12] editor de código utilizado para la programación del código.
- Geeks For Geeks: página web que se ha utilizado para obtener los códigos de los algoritmos de ordenación y búsqueda. Además, se ha utilizado como fuente principal para escribir las explicaciones de estos.
- https://github.com/MTrajK/coding-problems : se utiliza este repositorio para la redacción de los diferentes problemas que aparecen en la página.

Si se quiere saber más sobre los diferentes paquetes de NPM usados en el desarrollo de la aplicación se recomienda leer el Anexo V Manual del programador.

5.8 Planificación y documentación

Para la planificación y estructuración del proyecto se utilizan las siguientes herramientas:

- EZEstimate: [8] herramienta para el cálculo de una estimación del tiempo necesario para el desarrollo del proyecto.
- Microsoft Project: [7] herramienta para la planificación de tareas y organización en el tiempo del proyecto.
- Visual Paradigm: [10] herramienta utilizada para la creación de todos los diagramas UML
- Microsoft Word: [11] para la redacción de cualquier documento relacionado con el proyecto.
- TypeDoc: [9] paquete de NPM utilizado para la creación de la documentación del código.

6. Aspectos relevantes en el desarrollo

En este apartado se desarrollan los aspectos más relevantes en el desarrollo del proyecto.

6.1 Marco de trabajo

Para el desarrollo de este trabajo se ha utilizado el marco de trabajo del Proceso Unificado. [4] “El Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos”

El Proceso Unificado se caracteriza por estar basado en componentes y por usar el UML como lenguaje de representación. Además, también:

- Es un proceso conducido por casos de uso
- Está centrado en una arquitectura
- Es iterativo e incremental

El desarrollo usando el Proceso Unificado se basa en la repetición de ciclos tras los cuales se obtiene una versión entregable del producto (iterativo e incremental). Estos ciclos se dividen en 4 fases diferentes:

- Inicio: en esta fase se definen el alcance y desarrollo de casos de negocio
- Elaboración: fase en la que se realiza la planificación, detalle de los casos de uso y diseño de la arquitectura.
- Construcción: fase en la que se construye el producto
- Transición: es esta fase ya se tiene la versión entregable del producto. Además, también se produce la corrección de errores y se implementan las mejoras sugeridas en la revisión.

En la Ilustración 6 se puede ver el reparto de las diferentes cargas de trabajo divididas por disciplinas a lo largo de un ciclo de desarrollo:

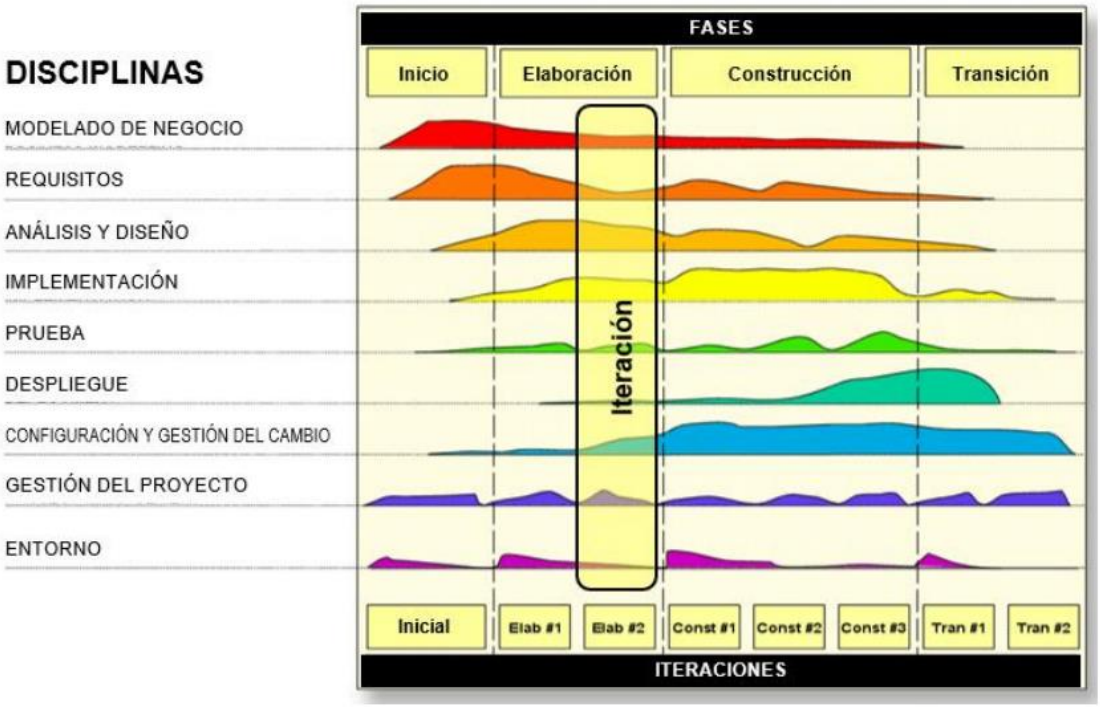


Ilustración 6 Diagrama de fases del ciclo de desarrollo

6.2 Planificación temporal

Una fase fundamental del proyecto es determinar el coste temporal de este y hacer un buen reparto y planificación de las tareas. Este apartado presenta un resumen de dicha fase, si se quiere conocer más acerca de ella se puede ver el Anexo I Planificación temporal.

Primeramente, se realiza la estimación del esfuerzo. Para ello se utiliza la estimación de costes en función de los casos de uso siguiendo la siguiente fórmula:

UCP = UUCP * TCF * ECF

Para el cálculo de los puntos de casos de uso sin ajustar se tienen que sumar el peso de los casos de uso sin ajustar y el peso de los actores sin ajustar, dando lugar a tablas de la siguiente forma (especificación de todos los casos de uso en el Anexo I Planificación temporal):

Actor	Complejidad
Usuario no autenticado	Complejo
Usuario autenticado	Complejo
Usuario administrador	Complejo
Sistema	Simple
Compilador online	Medio

Tabla 1 Complejidad de actores

Caso de uso	Transacciones	Complejidad
Registrar usuario	4	Medio
Borrar cuenta	2	Simple
Iniciar sesión	3	Simple
Cerrar sesión	1	Simple
Verificar correo	2	Simple
Pedir confirmación de correo	2	Simple
Editar mail	3	Simple
Editar contraseña	2	Simple
Actualizar datos estadísticos del usuario	1	Simple
Mostrar datos estadísticos del usuario	1	Simple
Inicializar cuenta	1	Simple
Recordar contraseña	3	Simple

Tabla 2 Complejidad de casos de uso

Seguidamente se calculan los factores de complejidad técnica y de entorno y todos estos valores son introducidos en la herramienta EZEstimate como podemos ver en las ilustraciones 7 y 8.

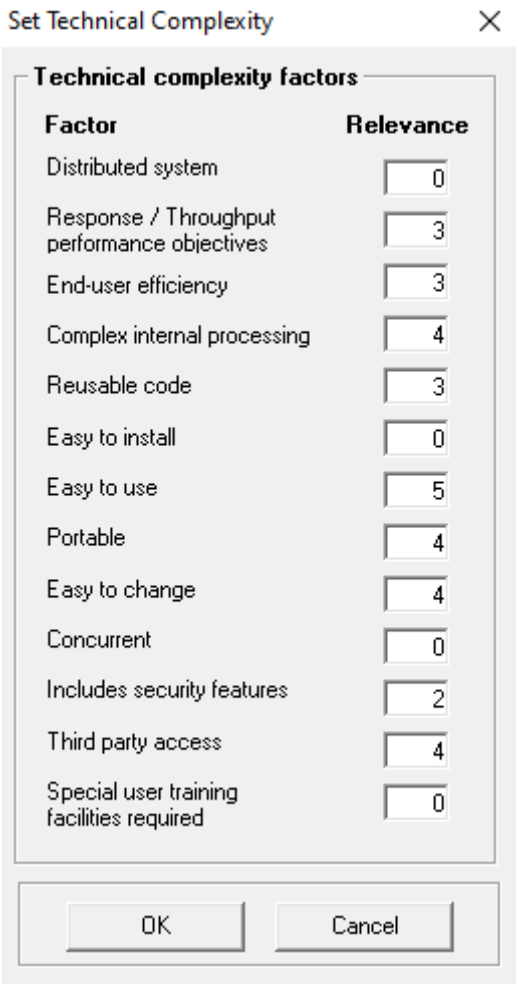


Ilustración 7 Factores de complejidad técnica

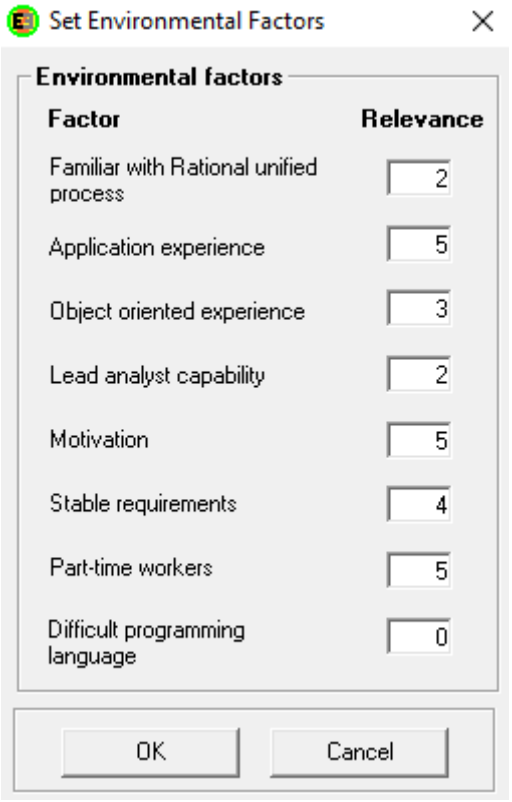


Ilustración 8 Factores de complejidad del entorno

Se puede ver la explicación detallada de por qué se dan esos valores a los factores de complejidad en el Anexo I Planificación temporal.

Una vez hecho esto, la herramienta EZEstimate ha arrojado los siguientes resultados presentados en la Ilustración 9.

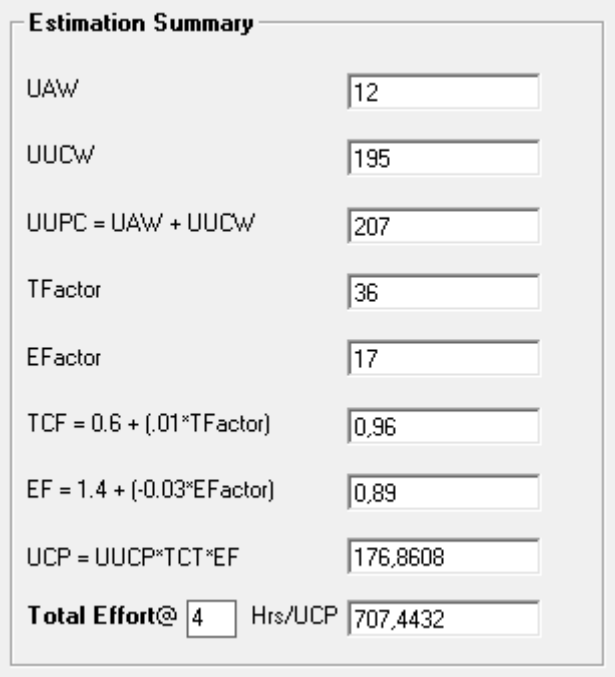
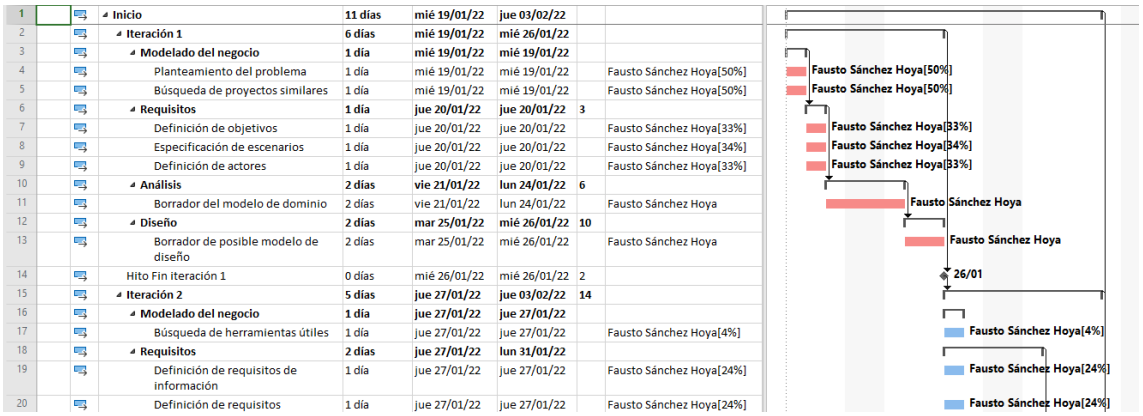


Ilustración 9 Resultados de la estimación de costes

En la que podemos ver que, teniendo en cuenta la cantidad de casos de uso simples, se ha cambiado el número de horas por caso de uso a 4, dando así un tiempo total de trabajo de unos 4 meses en una jornada laboral de 8 horas.

A continuación, usando la herramienta Microsoft Project se especifica un calendario laboral y se hace un reparto de tareas siguiendo la estructura del Proceso Unificado en el que agrupa las tareas en las siguientes divisiones: Modelado del trabajo, requisitos, análisis, diseño, implementación y pruebas. En la Ilustración 10 se puede ver un fragmento de esa división de tareas junto con su diagrama de Grantt correspondiente.



1		**Inicio**	**11 días**	**mié 19/01/22**	**jue 03/02/22**	
2		**Iteración 1**	**6 días**	**mié 19/01/22**	**mié 26/01/22**	
3		**Modelado del negocio**	**1 día**	**mié 19/01/22**	**mié 19/01/22**	
4		Planteamiento del problema	1 día	mié 19/01/22	mié 19/01/22	Fausto Sánchez Hoya[50%]
5		Búsqueda de proyectos similares	1 día	mié 19/01/22	mié 19/01/22	Fausto Sánchez Hoya[50%]
6		**Requisitos**	**1 día**	**jue 20/01/22**	**jue 20/01/22**	**3**
7		Definición de objetivos	1 día	jue 20/01/22	jue 20/01/22	Fausto Sánchez Hoya[33%]
8		Especificación de escenarios	1 día	jue 20/01/22	jue 20/01/22	Fausto Sánchez Hoya[34%]
9		Definición de actores	1 día	jue 20/01/22	jue 20/01/22	Fausto Sánchez Hoya[33%]
10		**Análisis**	**2 días**	**vie 21/01/22**	**lun 24/01/22**	**6**
11		Borrador del modelo de dominio	2 días	vie 21/01/22	lun 24/01/22	Fausto Sánchez Hoya
12		**Diseño**	**2 días**	**mar 25/01/22**	**mié 26/01/22**	**10**
13		Borrador de posible modelo de diseño	2 días	mar 25/01/22	mié 26/01/22	Fausto Sánchez Hoya
14		Hito Fin iteración 1	0 días	mié 26/01/22	mié 26/01/22	2
15		**Iteración 2**	**5 días**	**jue 27/01/22**	**jue 03/02/22**	**14**
16		**Modelado del negocio**	**1 día**	**jue 27/01/22**	**jue 27/01/22**	
17		Búsqueda de herramientas útiles	1 día	jue 27/01/22	jue 27/01/22	Fausto Sánchez Hoya[4%]
18		**Requisitos**	**2 días**	**jue 27/01/22**	**lun 31/01/22**	
19		Definición de requisitos de información	1 día	jue 27/01/22	jue 27/01/22	Fausto Sánchez Hoya[24%]
20		Definición de requisitos	1 día	jue 27/01/22	jue 27/01/22	Fausto Sánchez Hoya[24%]

Ilustración 10 Diagrama de tareas y diagrama de Grantt

Además, en cuanto al reparto de recursos a lo largo del desarrollo se consiguió un reparto de prácticamente el 100% todas las semanas como se puede apreciar en la Ilustración 11.

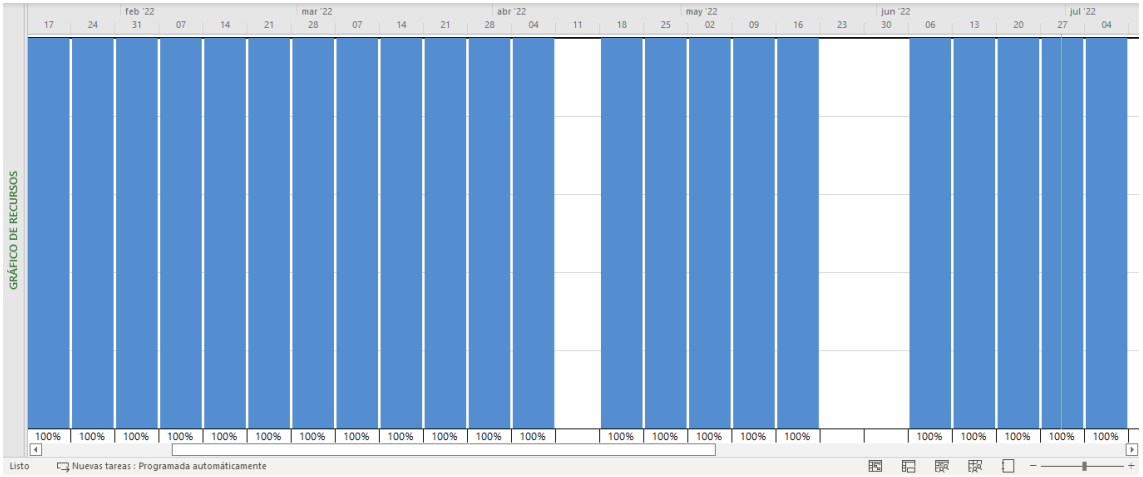


Ilustración 11 Reparto de recursos

Si se quiere más información acerca de la fase de planificación temporal se puede consultar el Anexo I Planificación temporal.

6.3 Especificación de requisitos de software

En la fase de especificación de requisitos se definen los participantes, actores, objetivos, requisitos de información, requisitos de restricción de información, requisitos no funcionales, requisitos funcionales y se llevan a cabo la especificación de los diferentes casos de uso. Todo ello acompañado de sus correspondientes tablas y diagramas.

Para saber más sobre esta fase se recomienda leer el Anexo II Especificación de requisitos de software.

6.3.1 Participantes

Los participantes en este proyecto son:

- Fausto Sánchez Hoya
- Juan Francisco de Paz Santana
- Héctor Sánchez San Blas
- Gabriel Villarrubia González

La especificación de los participantes se lleva a cabo a través del siguiente modelo de tabla:

Participante	**Fausto Sánchez Hoya**
Organización	**Universidad de Salamanca**
Rol	**Autor**
Es desarrollador	**Si**
Es cliente	**No**
Es usuario	**No**
Comentarios	**Ninguno**

Tabla 3 Participante Fausto Sánchez Hoya

6.3.2 Actores

Para la definición de actores se ha realizado el siguiente diagrama de la Ilustración 12 junto con las tablas correspondiente para cada actor.

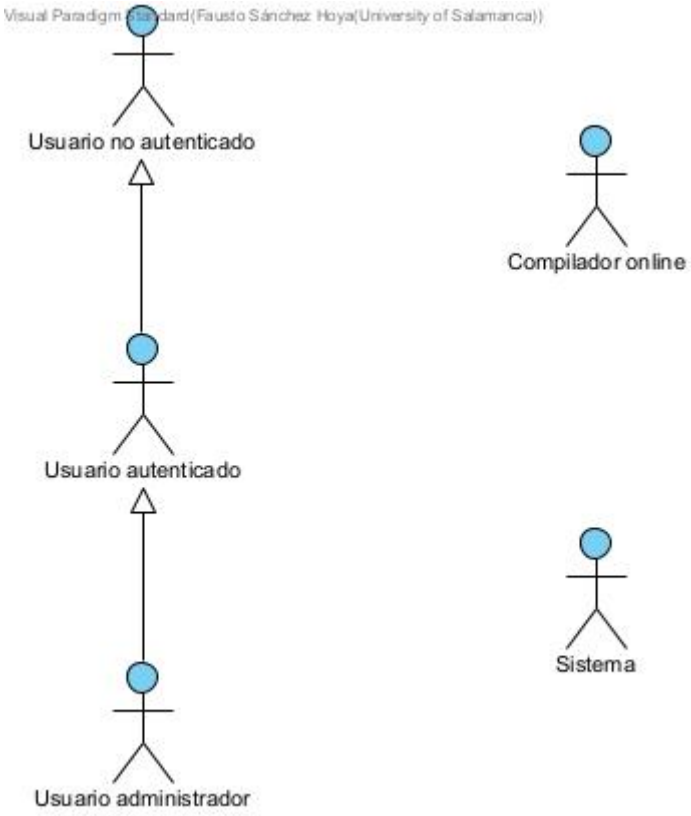


Ilustración 12 Diagrama de actores

ACT-0001	Usuario no autenticado
Versión	**1.0 (06/06/2022)**
Autores	**Fausto Sánchez Hoya**
Descripción	Este actor representa a un usuario que ha accedido al sistema, pero no está autenticado.
Comentarios	**Ninguno**

Tabla 4 Actor usuario no autenticado

6.3.3 Objetivos del sistema

Los objetivos del sistema son los siguientes:

- Gestión de usuarios
- Gestión de problemas
- Gestión de algoritmos
- Gestión de simulación

- Gestión de código
- Gestión de analytics

Su especificación se ha llevado a cabo con el uso de la siguiente tabla:

OBJ-001	Gestión de usuarios
Versión	**1.0 (06/06/2022)**
Autores	**Fausto Sánchez Hoya**
Descripción	El sistema deberá permitir el alta y baja de usuarios, la confirmación de su identidad a través del correo electrónico, la autentificación para iniciar sesión, así como la edición de sus datos una vez tengan sesión iniciada y la recopilación de estadísticas del usuario.
Subobjetivos	**Ninguno**
Importancia	**Vital**
Urgencia	**Inmediatamente**
Estado	**Validado**
Estabilidad	**Alta**
Comentarios	**Ninguno**

Tabla 5 Objetivo Gestión de usuarios

6.3.4 Requisitos de información

Los requisitos de información especificados son:

- Datos de usuario registrado
- Usuarios administradores
- Problema

Esta especificación se ha llevado a cabo usando tablas como la siguiente:

IRQ-001	Datos de usuario registrado
Versión	**1.0 (06/06/2022)**
Autores	**Fausto Sánchez Hoya**
Dependencias	*(OBJ-001) Gestión de usuarios*
Descripción	El sistema deberá almacenar la información de aquellos usuarios que estén registrados en el sistema. En concreto:

Datos específicos	ID Email Día que ha resuelto por última vez un problema Número de entrenamientos en cada día los últimos 365 días Lista de ID de problemas resueltos Tiempo total utilizado en resolver problemas
Importancia	**Vital**
Estado	**Validado**
Estabilidad	**Alta**
Comentarios	**Ninguno**

Tabla 6 Requisito de información

6.3.5 Requisitos de restricción de información

Vemos la especificación de dos requisitos de restricción de información:

- Datos de usuario registrado
- Problema

Esta especificación se ha llevado a cabo usando tablas como la siguiente:

CRQ-001	**Datos de usuario registrado**
Versión	**1.0 (06/06/2022)**
Autores	**Fausto Sánchez Hoya**
Dependencias	*(OBJ-001) Gestión de usuarios* *(IRQ-001) Datos de usuario registrado*
Descripción	La información almacenada por el sistema deberá satisfacer la siguiente restricción: No puede haber dos usuarios que tengan el mismo email o el mismo ID.
Importancia	**Vital**
Urgencia	**Inmediatamente**
Estado	**Validado**
Estabilidad	**Alta**
Comentarios	**Ninguno**

Tabla 7 Requisito de restricción de información

6.3.6 Requisitos no funcionales

Se presentan los siguientes requisitos no funcionales:

- Usabilidad
- Extensibilidad
- Rendimiento
- Portabilidad

Para ello se han usado tablas como la siguiente:

NFR-004	Portabilidad
Versión	**1.0 (08/06/2022)**
Autores	**Fausto Sánchez Hoya**
Dependencias	*(OBJ-004) Gestión de simulación*
Descripción	El sistema deberá asegurar que la presentación de la información sea la misma en cualquier navegador usado, así como asegurar la adaptación a otros dispositivos como móviles o tabletas.
Importancia	**Vital**
Urgencia	**Inmediatamente**
Estado	**Validado**
Estabilidad	**Alta**
Comentarios	**Ninguno**

Tabla 8 Requisito no funcional

6.3.7 Requisitos funcionales

En este apartado se realiza la separación del sistema en diferentes paquetes como podemos ver en la Ilustración 13.

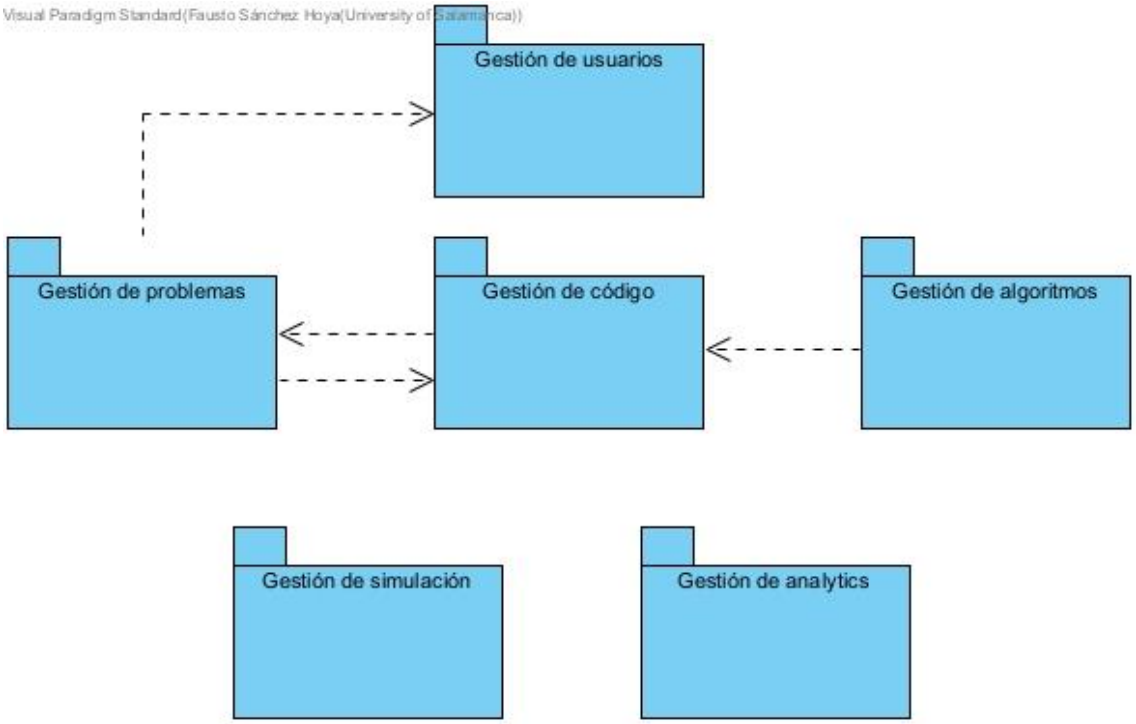


Ilustración 13 Diagrama de paquetes

Repartidos entre los paquetes vemos especificados un total de 37 de casos de uso. En la Ilustración 14 podemos ver el diagrama de casos de uso del paquete Gestión de usuarios.

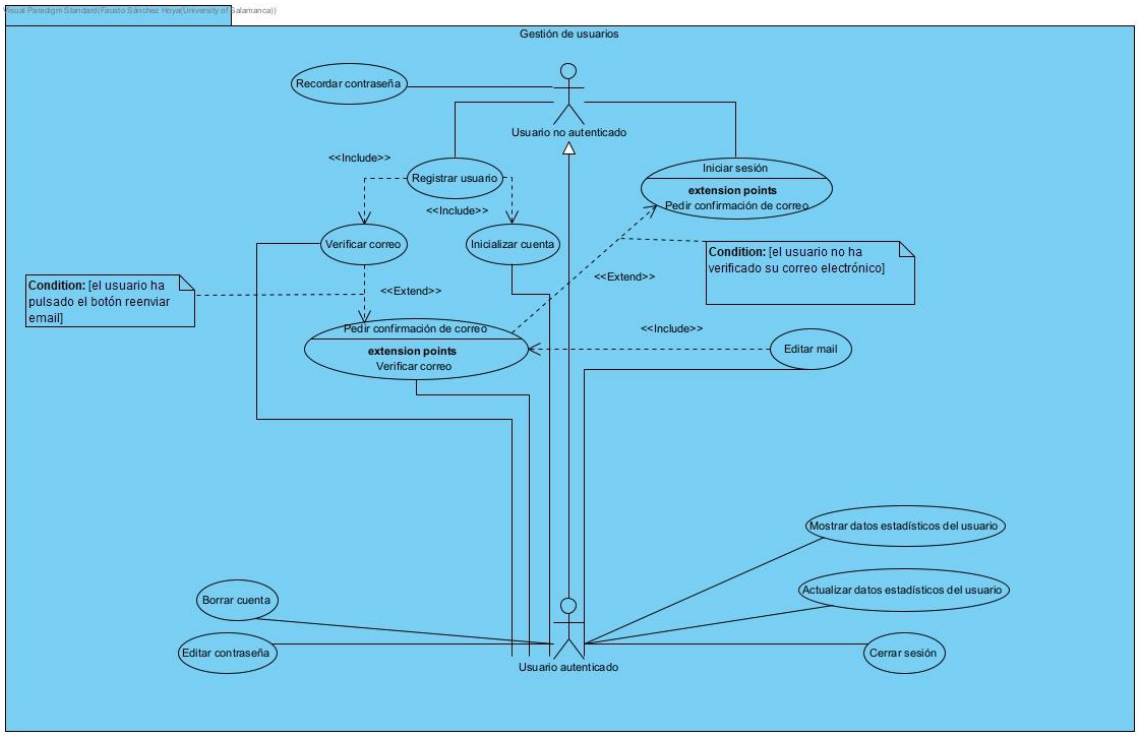


Ilustración 14 Diagrama de casos de uso

UC-001	Registrar usuario	
Versión	**1.0 (08/06/2022)**	
Autores	**Fausto Sánchez Hoya**	
Dependencias	*(OBJ-001) Gestión de usuarios* *(IRQ-001) Datos de usuario registrado* *(CRQ-001) Datos de usuario registrado* *(NFR-001) Usabilidad*	
Descripción	El sistema debe comportarse tal como se describe en el siguiente caso de uso cuando un *Usuario no autenticado (ACT-0001)* de al botón de Registrarse.	
Precondición	El usuario no debe estar autenticado.	
Secuencia normal	**Paso**	Acción
	1	El actor *Usuario no autenticado (ACT-0001)* inicia el caso de uso.
	2	El sistema presenta la pantalla de registro con el correspondiente formulario.
	3	El actor *Usuario no autenticado (ACT-0001)* rellena el formulario presentado.
	4	El sistema valida los datos proporcionados por el usuario.
	5	Se realiza el caso de uso *Verificar correo (UC-005)*
	6	Se realiza el caso de uso *Inicializar cuenta (UC-011)*
	7	El sistema finaliza el caso de uso.
Postcondición	El *Usuario no autenticado (ACT-0001)* pasa a ser un *Usuario autenticado (ACT-0002).*	
Excepciones	**Paso**	Acción
	4	Si la validación de los datos no es correcta, el sistema informa al usuario para que este los reintroduzca. A continuación, este caso de uso continua.
Importancia	**Vital**	
Urgencia	**Inmediatamente**	
Estado	**Validado**	
Estabilidad	**Alta**	
Comentarios	**Ninguno**	

Tabla 9 Definición de caso de uso

Si se quiere saber más sobre todo este proceso se puede consultar el Anexo II Especificación de requisitos software

6.4 Análisis del sistema

En la parte del análisis del sistema se baja un poco más el nivel de abstracción y se especifica lo mencionado en los siguientes apartados.

Si se quiere saber más sobre esta sección se recomienda la lectura del Anexo III Análisis del sistema.

6.4.1 Modelo del dominio

El modelo de dominio, representado en la Ilustración 15, especifica las propiedades internas de la aplicación y sus relaciones.

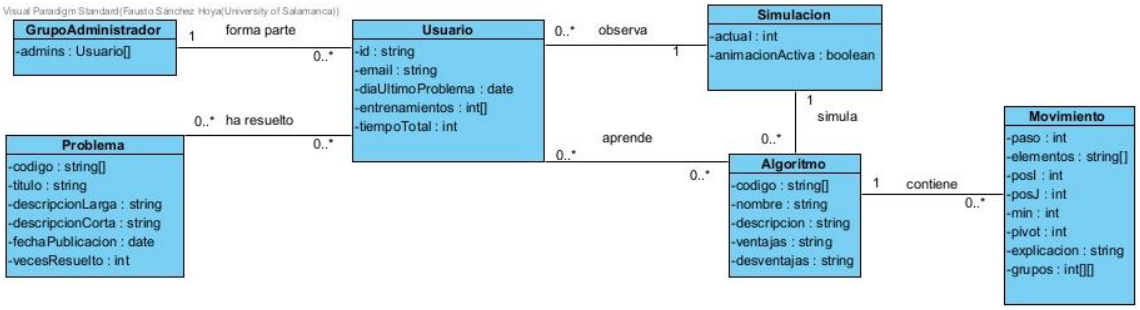


Ilustración 15 Diagrama del modelo de dominio

6.4.2 Paquetes de análisis

Se ha llevado a cabo con el mismo reparto que en los paquetes especificados en la sección anterior, pero, además, en cada paquete se especifican una serie de clases. En la Ilustración 16 podemos ver el diagrama de paquete de análisis de Gestión de usuarios.

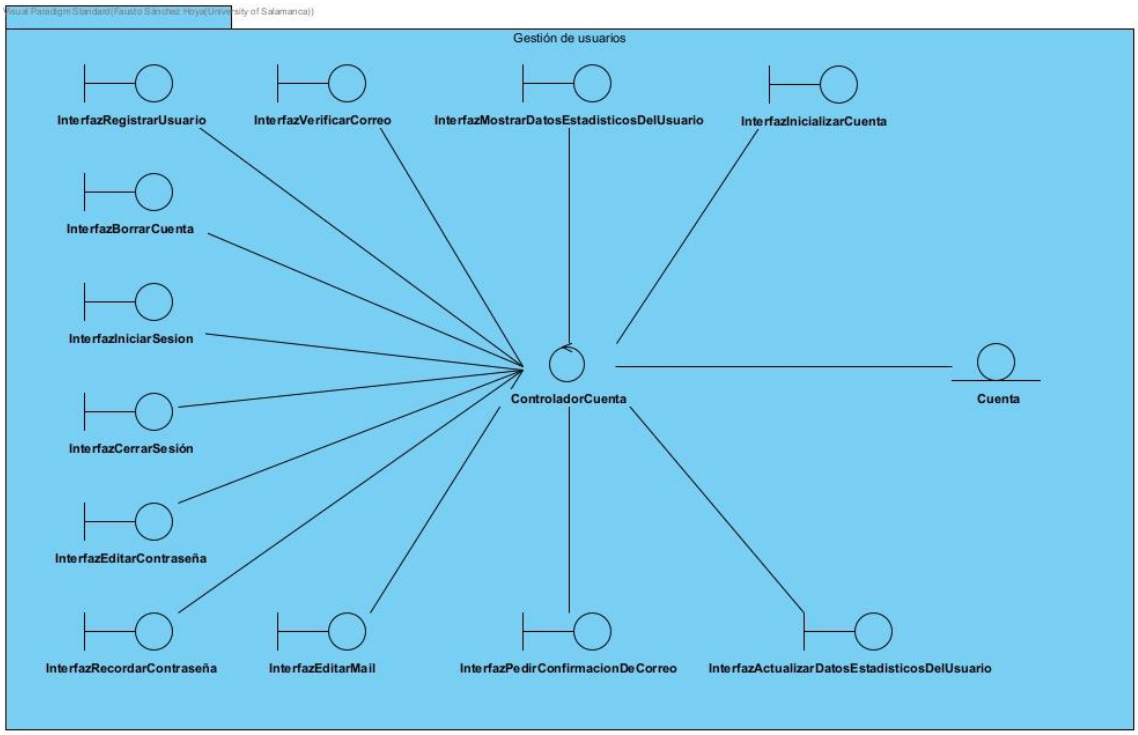


Ilustración 16 Diagrama de paquete de análisis

6.4.3 Vista de arquitectura

Gracias a todos los subsistemas de análisis se puede ver una imagen global de la arquitectura del sistema como se muestra en la Ilustración 17.

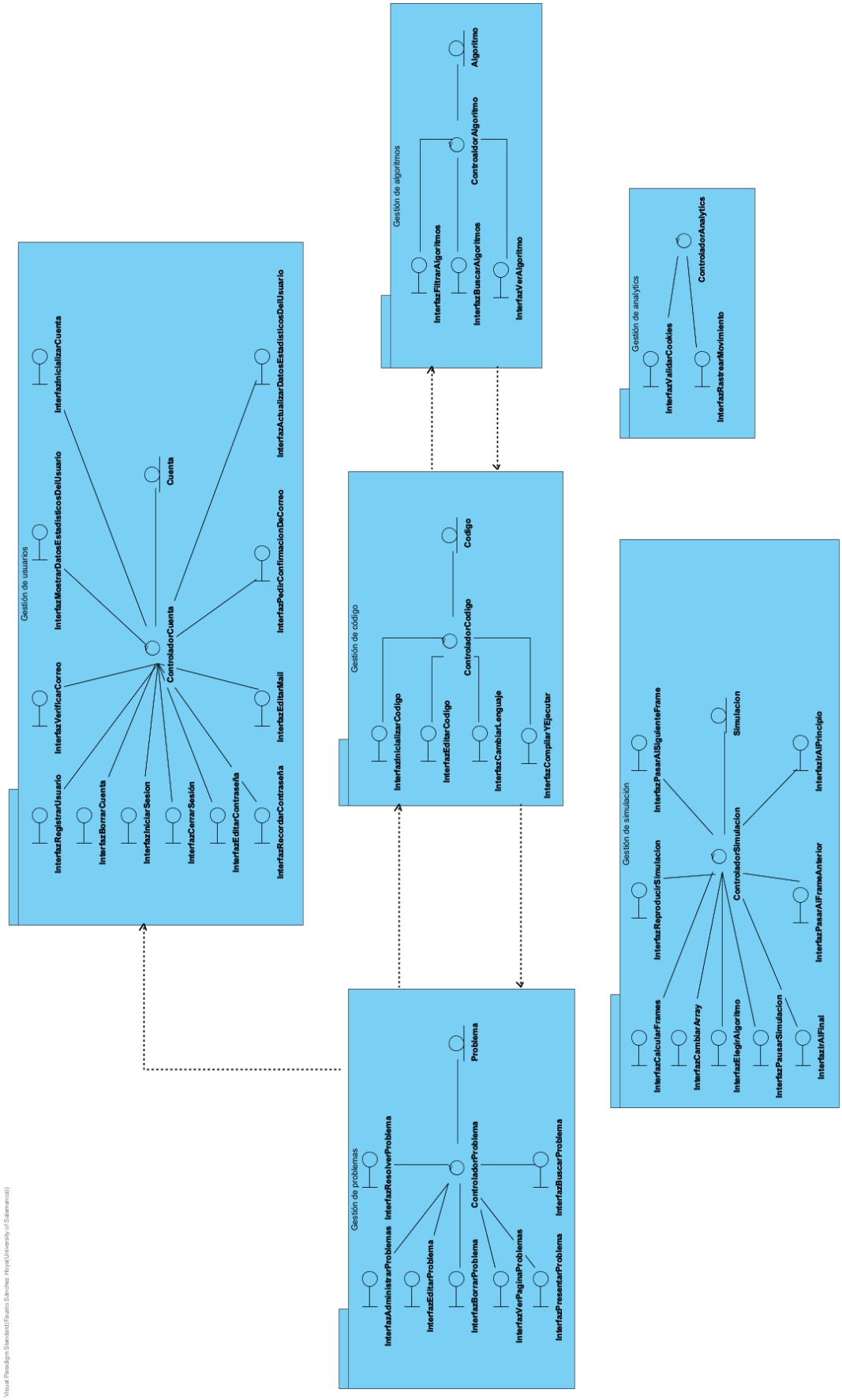


Ilustración 17 Vista de arquitectura

6.4.4 Vista de interacción de casos de uso

Finalmente, y para justificar esta vista de arquitectura, se procede a realizar los diagramas de secuencia de cada caso de uso. Un ejemplo de uno de ellos es el mostrado en la Ilustración 18.

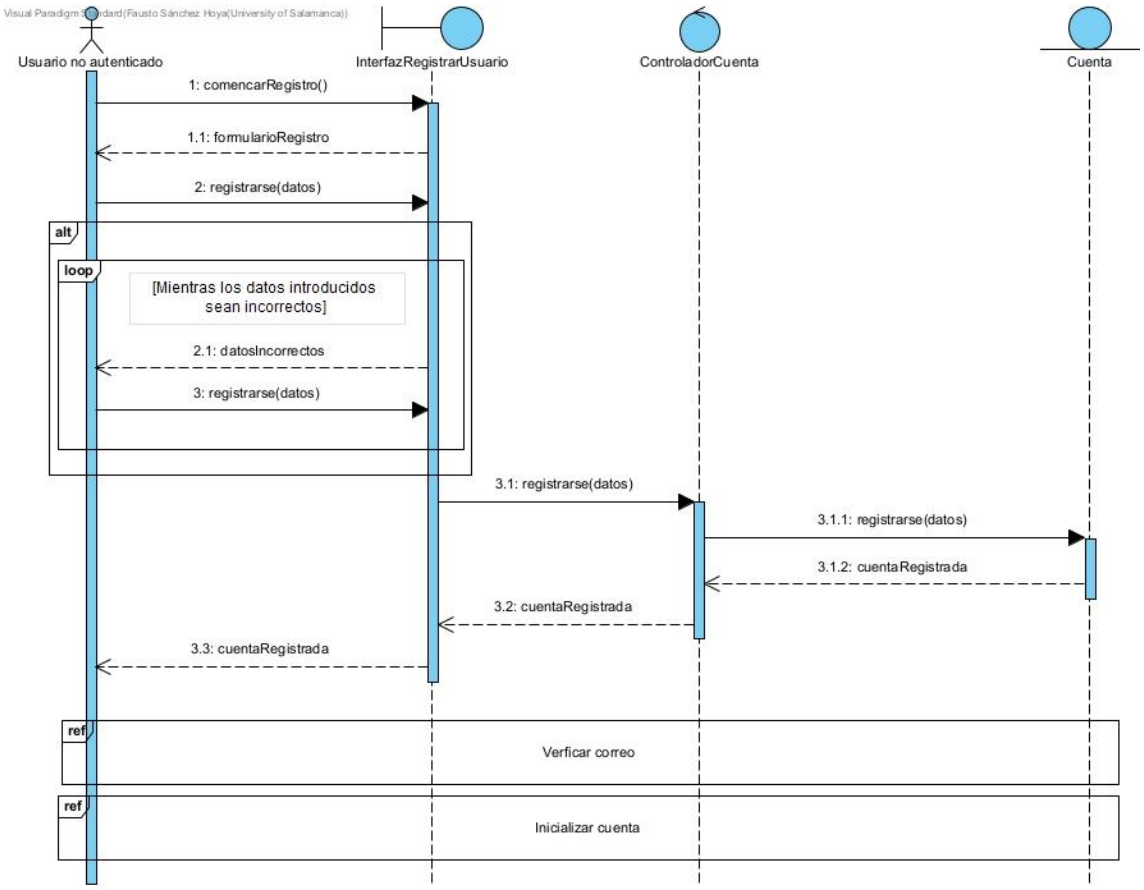


Ilustración 18 Diagrama de secuencia de casos de uso

6.5 Diseño del sistema

En esta fase se llega a la especificación más cercana antes de la implementación. Para saber más acerca de ella se recomienda leer el Anexo IV Diseño del sistema.

Debido al uso de React debemos tener en cuenta en el diseño que cada componente de React seguirá el patrón MVVM de manera individual. Este patrón trata de desacoplar al máximo la vista del modelo, pero, a diferencia de otros patrones parecidos como el MVC, habrá comunicación bidireccional entre la parte View y View-Model y entre View-Model y Model. Esto es así ya que se tienen en cuenta los estados del componente. Para más información acerca de este patrón se recomienda leer el Anexo IV Diseño del sistema.

6.5.1 Subsistemas de diseño

En esta parte se especifica la separación del sistema en varios subsistemas y sus dependencias entre ellos tal y como se muestra en la Ilustración 19.

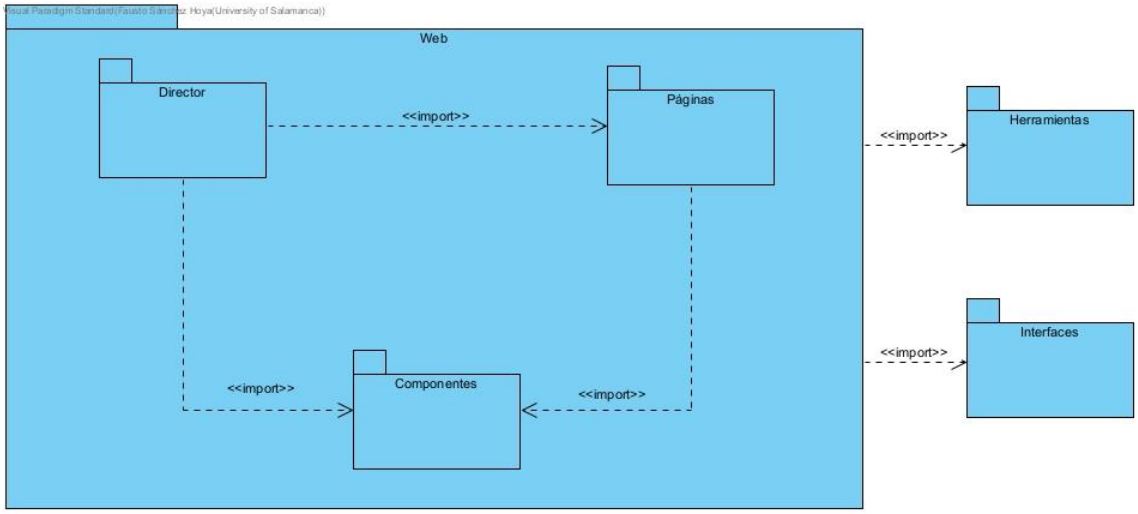


Ilustración 19 Diagrama de subsistemas de diseño

También se especifica, mediante diagramas, qué compone cada subsistema del proyecto. En este caso podemos ver un diagrama del subsistema “Componentes” en la Ilustración 20.

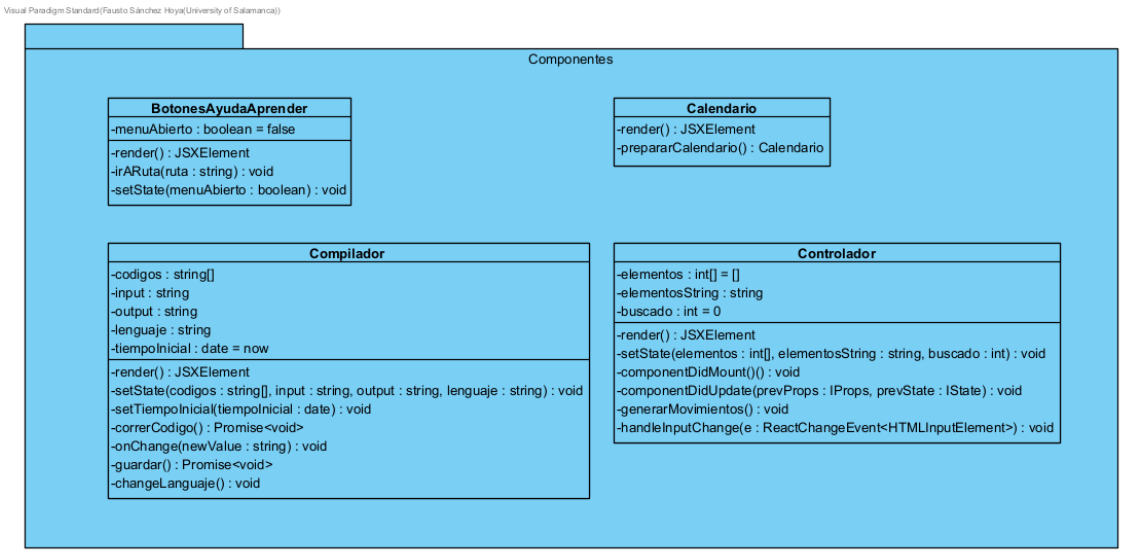


Ilustración 20 Diagrama de subsistema Componentes

6.5.2 Realización de casos de uso

A continuación, teniendo como referencia lo especificado anteriormente, se llevan a cabo los diferentes diagramas de secuencia como el mostrado en la Ilustración 21.

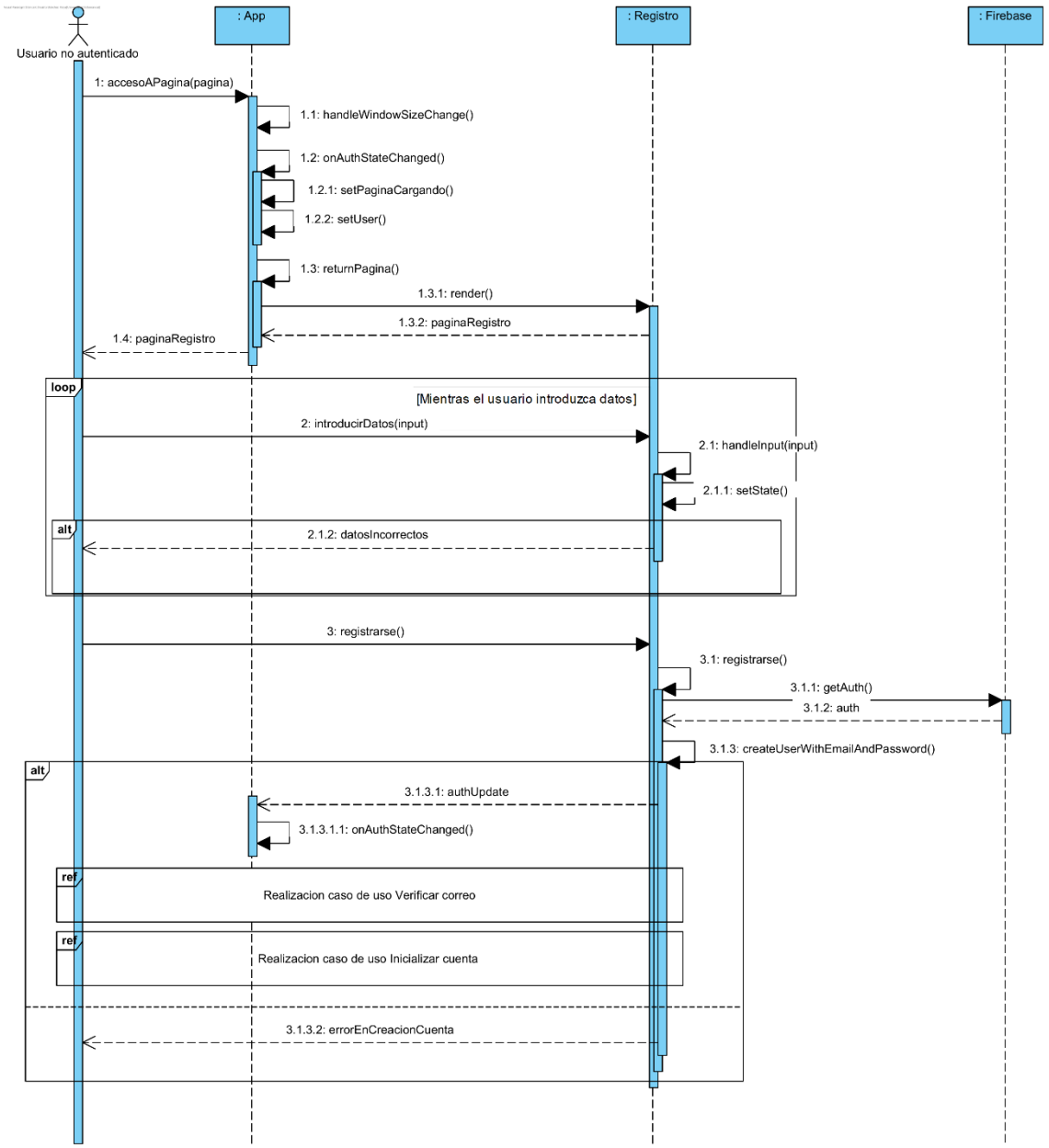


Ilustración 21 Diagrama de secuencia de Realización de casos de uso

6.5.3 Vista de arquitectura

Seguidamente, gracias a los diagramas de secuencia de la sección anterior, se presenta un diagrama que representa las relaciones existentes entre las diferentes clases. (Por temas de espacio y presentación se ruega verlo en el Anexo IV Diseño del sistema)

6.5.4 Modelo de despliegue

En este apartado se especifican todas las partes que conforman el despliegue del sistema y cómo se relacionan entre ellas como se refleja en la Ilustración 22:

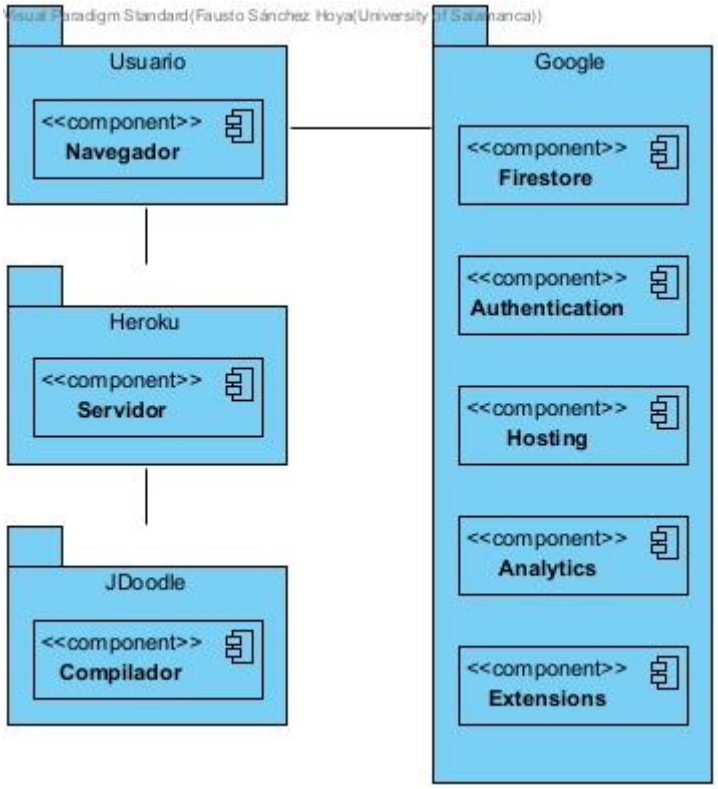


Ilustración 22 Modelo de despliegue

6.5.5 Diseño de la base de datos

Finalmente se especifica cómo está organizada la base de datos de Firebase teniendo en cuenta su modelo de colecciones y documentos tal y como se muestra en la Ilustración 23.

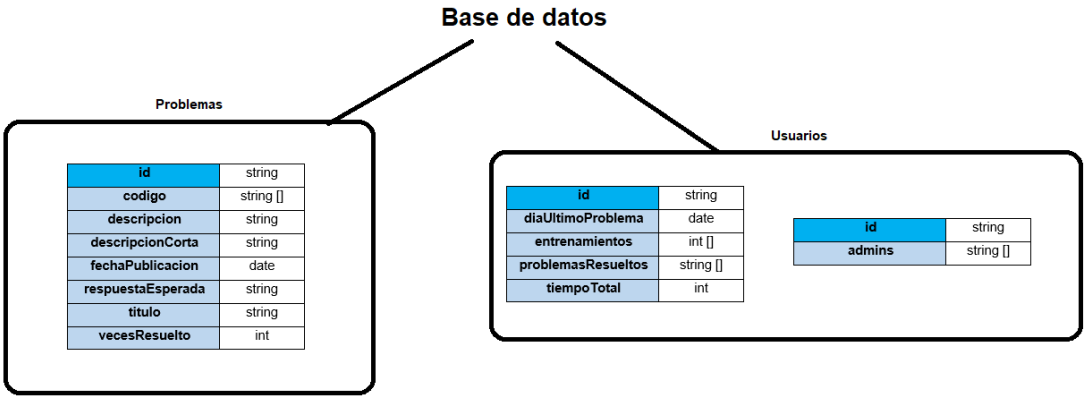


Ilustración 23 Diagrama de base de datos

6.6 Implementación

Después de la definición de los requisitos, el análisis y el diseño llega la fase de implementación. En esta fase vemos las siguientes etapas:

6.6.1 Etapa de prototipado

En esta etapa se llevan a cabo la implementación y pruebas tanto del editor de código como del simulador gráfico para comprobar su viabilidad.

En cuanto al simulador se prueba cómo realizar los cálculos de los frames de la simulación a partir del input introducido por un usuario, así como el control de la reproducción por parte de este. En la Ilustración 24 podemos ver dicho prototipo.

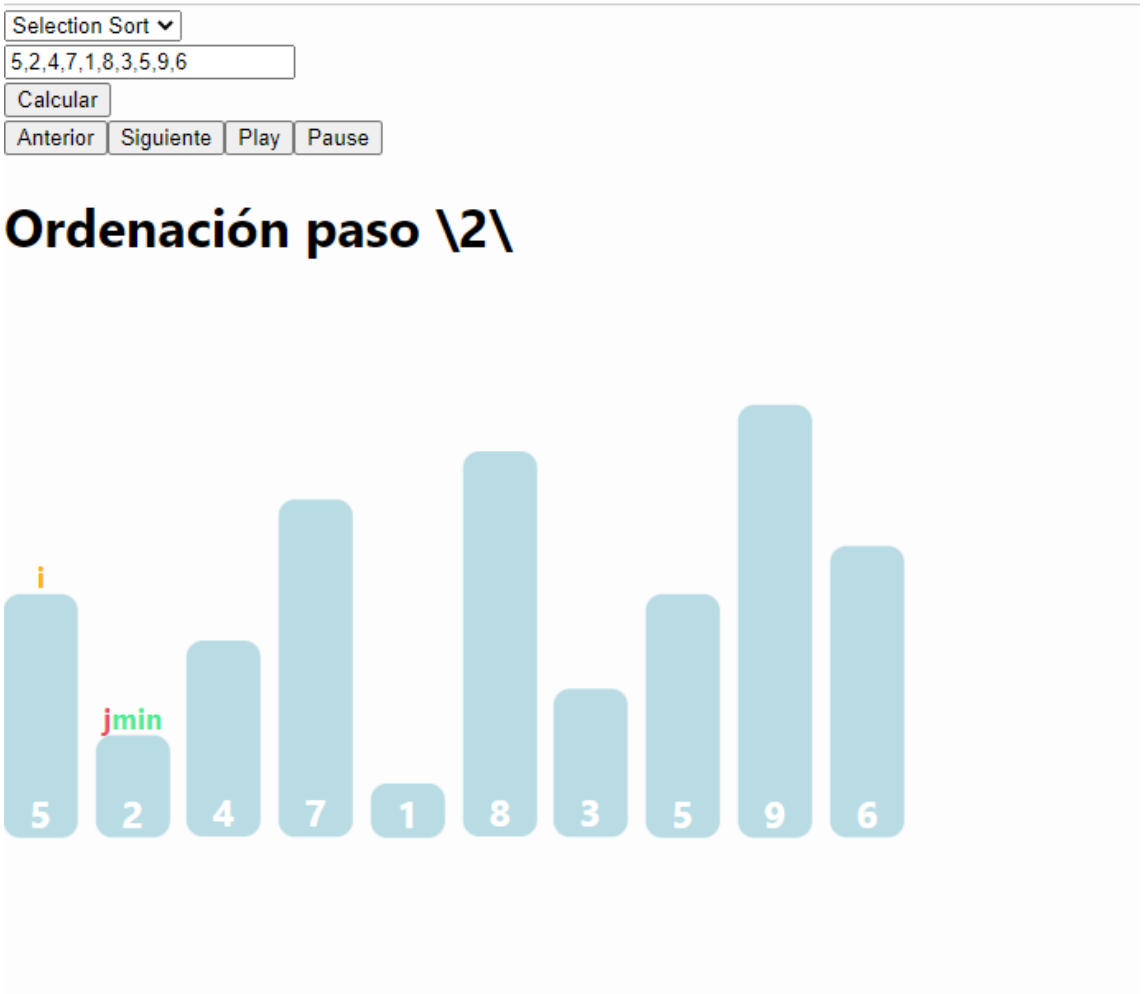


Ilustración 24 Prototipo de simulador

En cuanto al editor de código online se prueba la conexión con el compilador online JDoodle, así como la posibilidad de comparar el output de la ejecución con una

respuesta esperada para así completar los problemas. En la Ilustración 25 podemos ver dicho prototipo.

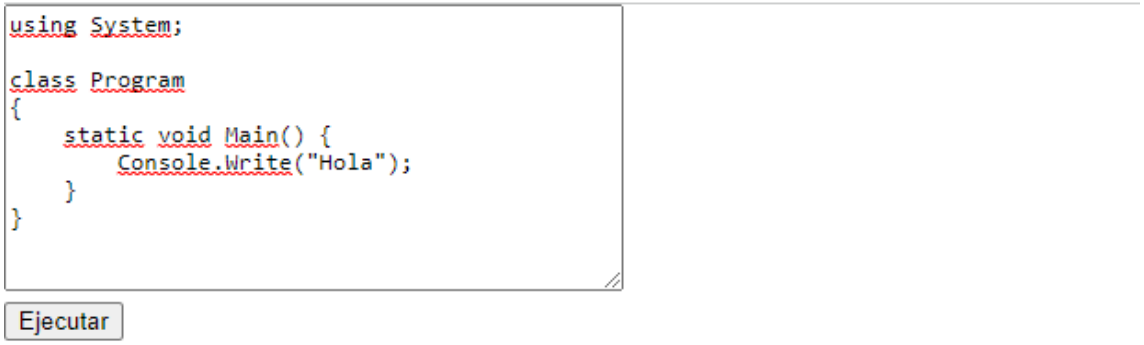


Ilustración 25 Prototipo de editor online

6.6.2 Etapa de diseño de la interfaz

Una vez hechos los prototipos y con la funcionalidad principal del sistema privados se pasa al diseño de la interfaz de usuario. Para ello, al principio se investigan diferentes páginas relacionadas con este proyecto y se observa la disposición de sus elementos, así como las fuentes utilizadas. Entre estas páginas se encuentran:

- https://www.topcoder.com/
- https://www.coderbyte.com/
- https://www.hackerrank.com/
- https://exercism.io/
- https://visualgo.net/en/sorting
- https://algorithm-visualizer.org/

De esta exploración de la competencia se obtiene que la mayoría de las páginas web tienen un fondo blanco y con el menú arriba. El menú hay páginas que lo tienen o bien blanco o negro y su paleta no suele variar mucho más. En cuanto a la localización de los elementos del menú tenemos las tres opciones, derecha, centro e izquierda siendo esta última la más utilizada.

En cuanto a fuentes se utiliza de forma general Open Sans para textos y títulos (diferenciando con colores y grosores) y Source Code Pro para la programación.

En cuanto al diseño general me gusta mucho la página https://exercism.org/ y se ha tomado como referencia a la hora de estructurar el proyecto.

A continuación, y en mi caso casi de manera anecdótica, se dibujan en papel las diferentes pantallas que se querrían tener en el sistema junto con anotaciones que explican más o menos cómo sería su implementación. En las ilustraciones 26 y 27 podemos ver algunos ejemplos.

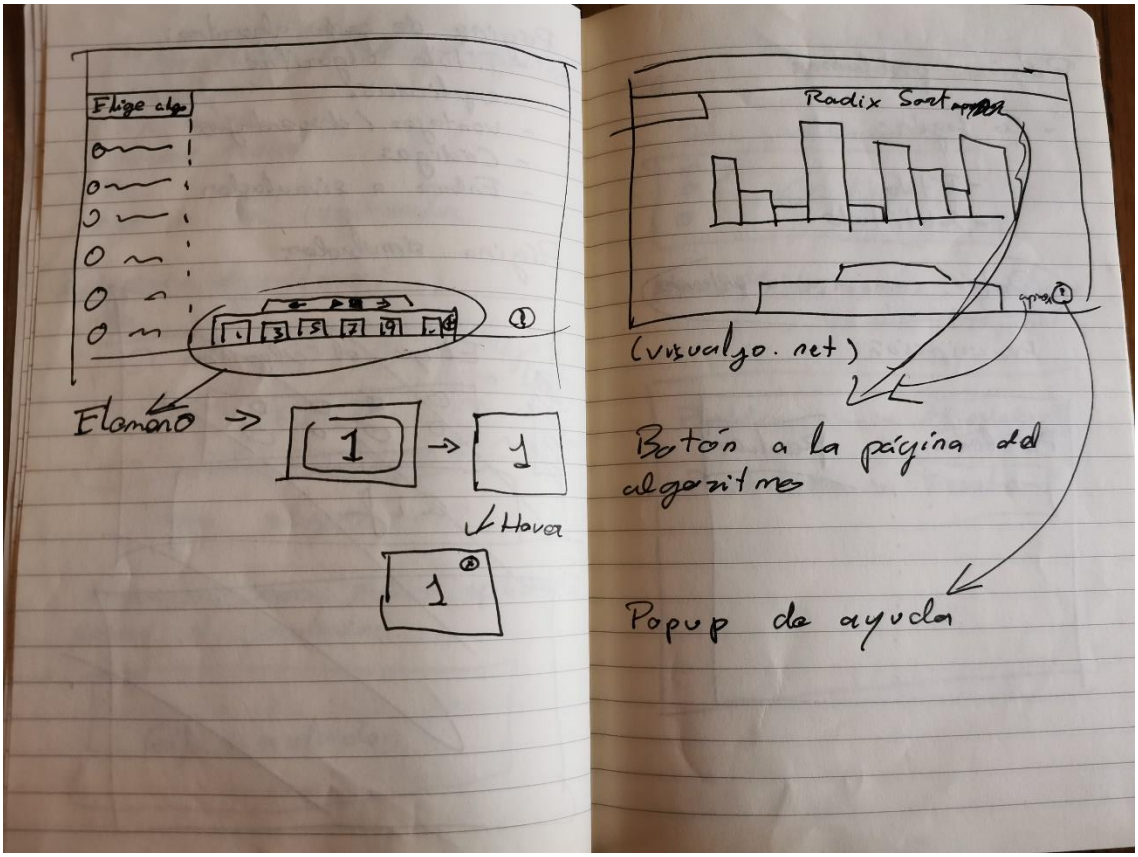


Ilustración 26 Prototipado a papel 1

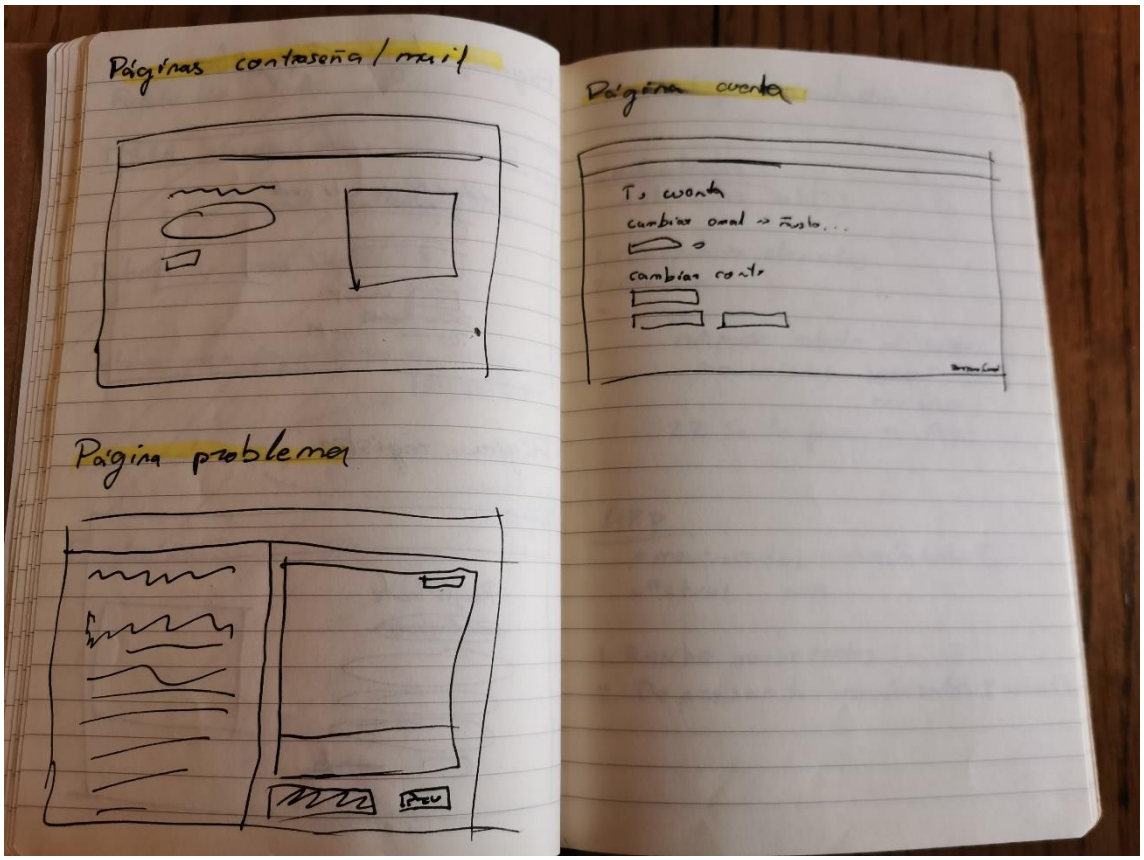


Ilustración 27 Prototipado a papel 2

Antes de pasar al prototipado digital se eligen las fuentes que se van a utilizar en la página, así como su paleta de colores. En cuanto a las fuentes elegidas se ha optado por Open Sans para tanto títulos y texto, y Source Code Pro para los fragmentos de código, estando así acorde con el resto de las páginas vistas. En cuanto a la paleta de colores se ha escogido una paleta que represente tanto la tecnología como el aprendizaje, por ello se utiliza una mezcla entre el azul y el negro, combinación usada frecuentemente en la tecnología, pero en tonos un poco pastel, para así representar el aprendizaje. En la Ilustración 28 podemos ver la paleta de colores principal.

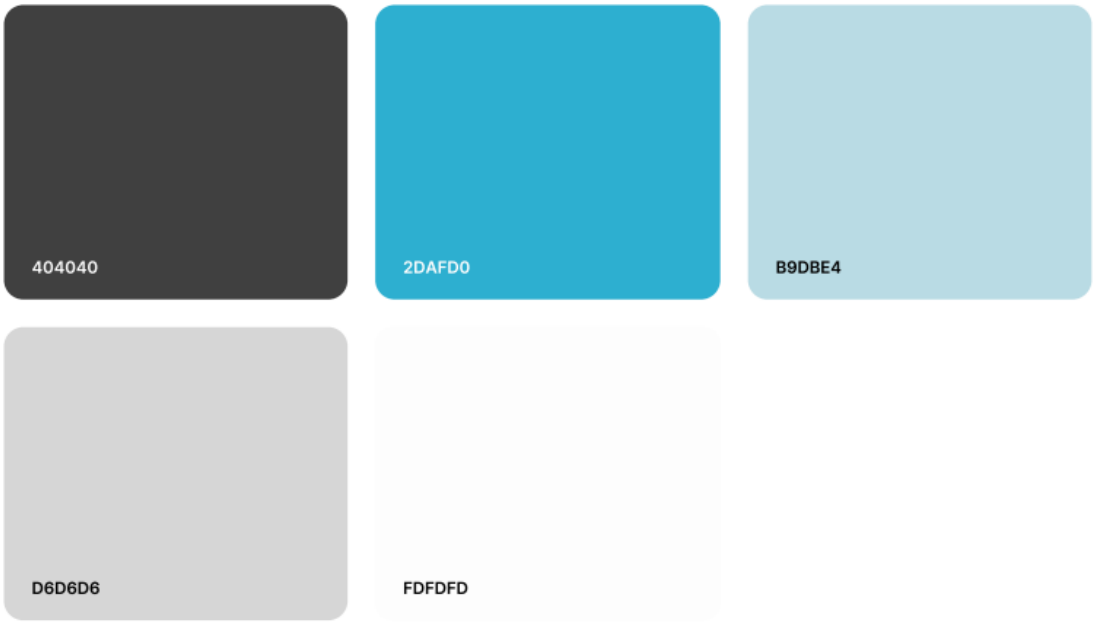


Ilustración 28 Paleta de colores

- #404040:se usa principalmente para los títulos, textos y la barra de navegación.
- #2DAFD0: se usa como color para destacar elementos como botones, enlaces, etc.
- #B9DBE4: este color es utilizado para elementos que no se quisieran destacar tanto como con el color anterior pero aun así llamaran la atención del usuario.
- #D6D6D6: este color es utilizado en el footer.
- #FDFDFD: se elige este color para el fondo en vez de un blanco puro para que no hubiera tanto contraste.

Además de estos colores principales, se utilizan otros en menor cantidad como por ejemplo el #EC4C5E para mensajes de alerta u otros como #794CEC, #4CEC94, #FAB400, #EC4C5E, #B54CEC, #ECA44C (morado, verde, amarillo, rojo, rosa, naranja) para los diferentes elementos secundarios de las simulaciones.

Seguidamente a esto se pasa directamente al prototipado digital usando el programa Adobe XD. Aquí podemos ver algunos ejemplos de las pantallas en las ilustraciones 29, 30 y 31.

Ilustración 29 Prototipado digital 1

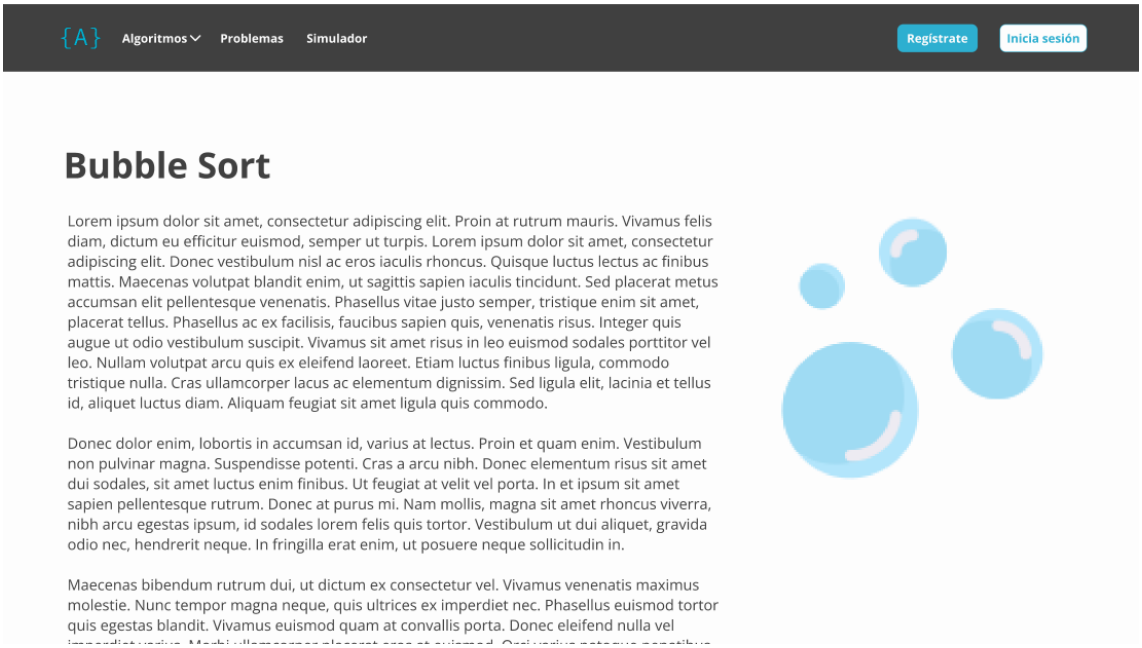


Ilustración 30 Prototipado digital 2

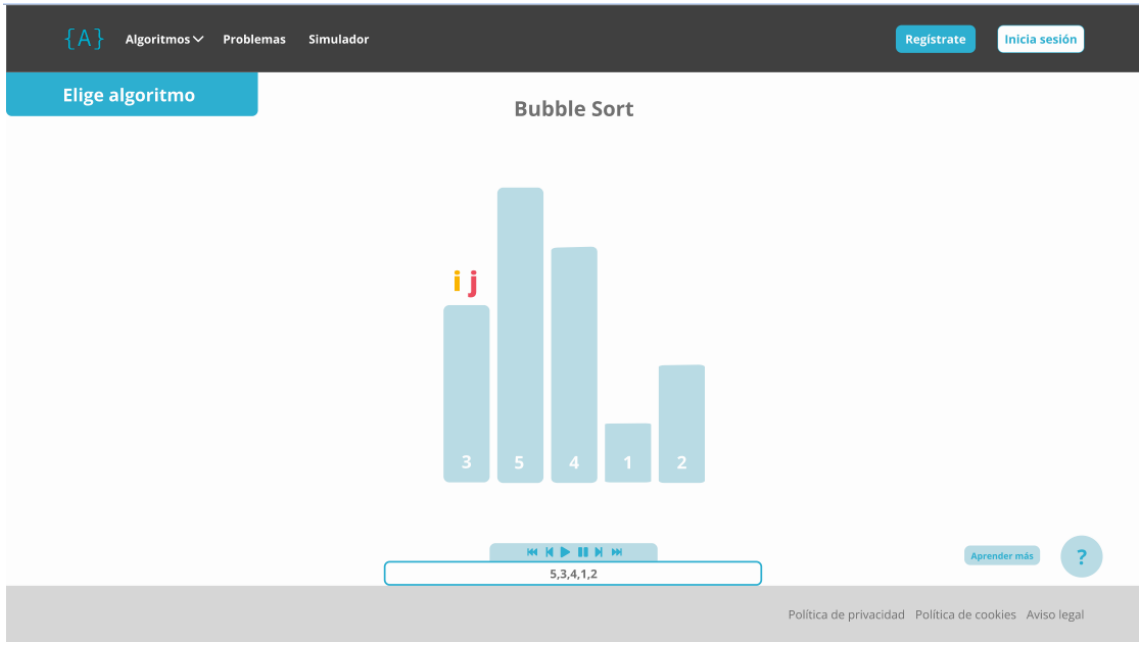


Ilustración 31 Prototipado digital 3

Utilizando este prototipado digital se realizan pruebas con diferentes usuarios de las que se obtienen los siguientes resultados:

- En la página de algoritmos, las pestañas no se diferencian lo suficiente y el usuario hay veces que no sabe exactamente dónde está. Para solucionar este problema, como podemos ver en la Ilustración 32, en el producto final se sigue el ejemplo de navegadores como Google Chrome en las que las pestañas no utilizadas se presentan con un color más oscurecido.

Ilustración 32 Solución a entrevistas de usuarios 1

- En cuanto al simulador se ve que hay algunos usuarios que no saben distinguir la funcionalidad de los botones de controlar la reproducción. Además, también existe el problema que los usuarios no saben si la reproducción de la simulación estaba activa. Para solucionar estos problemas, como podemos ver en la Ilustración 33, en el producto final se añaden unos recuadros con una breve descripción de la funcionalidad del botón cuando se pone el ratón encima y los botones de pausa y play se hace que se cambien de color en función del estado de la animación.

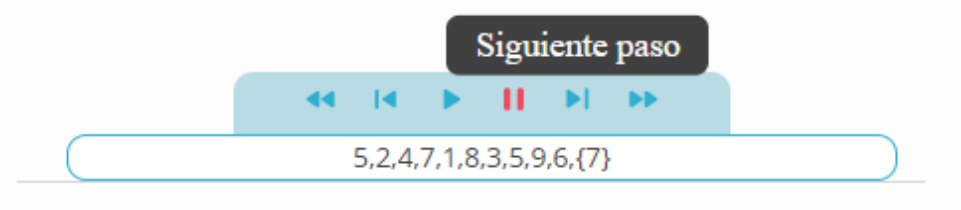


Ilustración 33 Solución a entrevistas de usuarios 2

- Por otro lado, varios usuarios informan que los textos largos como explicaciones de algoritmos o problemas no son del todo cómodos de leer con la fuente elegida por lo que, al final, se ha pasado a usar la fuente Roboto, la cual es mucho más legible y es también muy utilizada en otras webs por lo que el usuario está acostumbrado.

Una vez terminado el diseño de la interfaz se pasa directamente a la programación.

6.6.3 Etapa de programación

Durante esta etapa se procede a la programación del sistema teniendo en cuenta todo lo anterior especificado.

Para saber más información sobre este apartado se recomienda leer el Anexo V Manual del programador.

En la programación se tiene en cuenta el patrón MVVM que utiliza React explicado en el apartado de diseño y a parte se tiene en cuenta el siguiente esquema de funcionamiento principal:

1. React detecta un cambio en la URL
2. El archivo App.tsx comprueba el estado de autentificación del usuario y si este está accediendo desde un dispositivo móvil. Después de esto, presenta una serie de componentes programados también usando React y Typescrit que son de carácter general y aparecerán en todas las pantallas como la barra de navegación y el footer. Finalmente llama a renderizar a la página correspondiente en función de la URL.
3. El componente de React que representa dicha página calcula la funcionalidad correspondiente a esa página como podría ser obtener los datos del usuario y a continuación presenta por pantalla dicha página usando opcionalmente otros componentes.

En cuanto al funcionamiento del simulador sigue el siguiente esquema:

1. El usuario accede a la página del simulador como se ha mencionado anteriormente. Se carga Simulador.tsx que a su vez usa los componentes SelectorAlgoritmo.tsx, RepreMovimiento.tsx, Controlador.tsx, FuncionesAlgoritmos.tsx, BotonesAyudaAprender.tsx y la interfaz que representa un movimiento Movimiento.tsx.
2. El usuario selecciona un algoritmo usando la interfaz proporcionada por RepreMovimiento.tsx y este estado se le comunica a Simulador.tsx que llama a Controlador.tsx que se encarga de cargar el array de elementos predeterminado, llamar a FuncionesAlgoritmos.tsx para que generen el array de frames de la simulación utilizando la interfaz Movimiento.tsx y finalmente ese array se pasa de vuelta al estado de Simulador.tsx.
3. Simulador.tsx se encarga de pasarle cada movimiento por separado a RepreMovimiento.tsx para que lo presente por pantalla.

Para saber más información acerca del resto de ficheros y paquetes NPM utilizados para la implementación del proyecto, así como la forma correcta de extender la funcionalidad de la página se recomienda leer el Anexo V Manual del programador.

Una vez terminada la programación se pasó a la implementación de las diferentes simulaciones, problemas y explicaciones de algoritmos que se iban a poder visualizar en la página.

6.7 Fase de pruebas

La realización de pruebas es una parte fundamental del desarrollo del proyecto. Aprovechando la escasa o nula dependencia que se podía dar entre diferentes secciones de la web se han podido ir probando por separado a medida que se iban implementando. De esta forma se ha conseguido que todo lo subido a la rama de producción fuera estable.

Finalmente se desarrollan pruebas globales para comprobar que todo el sistema funcione de forma correcta y óptima en diferentes escenarios entre los que se encuentran:

- Pruebas con diferentes resoluciones de pantalla
- Pruebas con diferentes velocidades de internet aprovechando la herramienta inspeccionar de Google Chrome
- Pruebas en diferentes navegadores.

También, y teniendo en cuenta el requisito no funcional de rendimiento, se llevan a cabo pruebas con la herramienta PageSpeed Insights de Google que permite evaluar la capacidad de carga de una página web, arrojando los siguientes resultados que podemos observar en las ilustraciones 34 y 35.

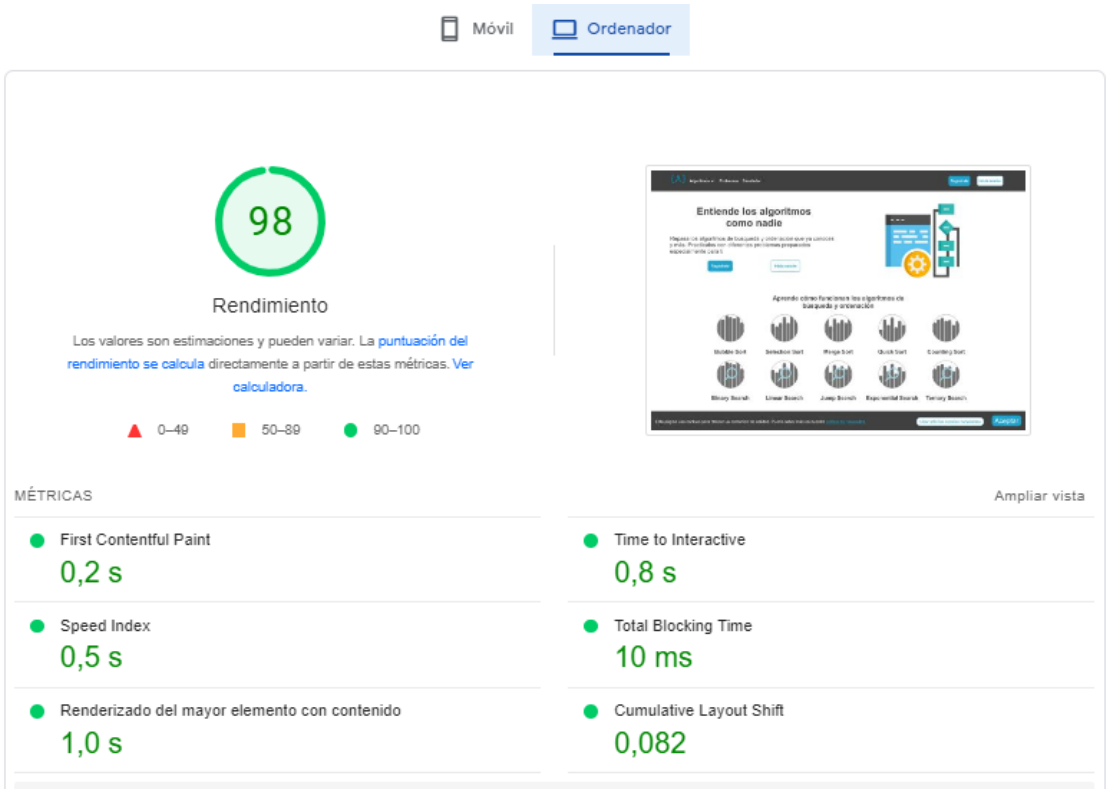


Ilustración 34 Test PageSeed Insights Ordenador

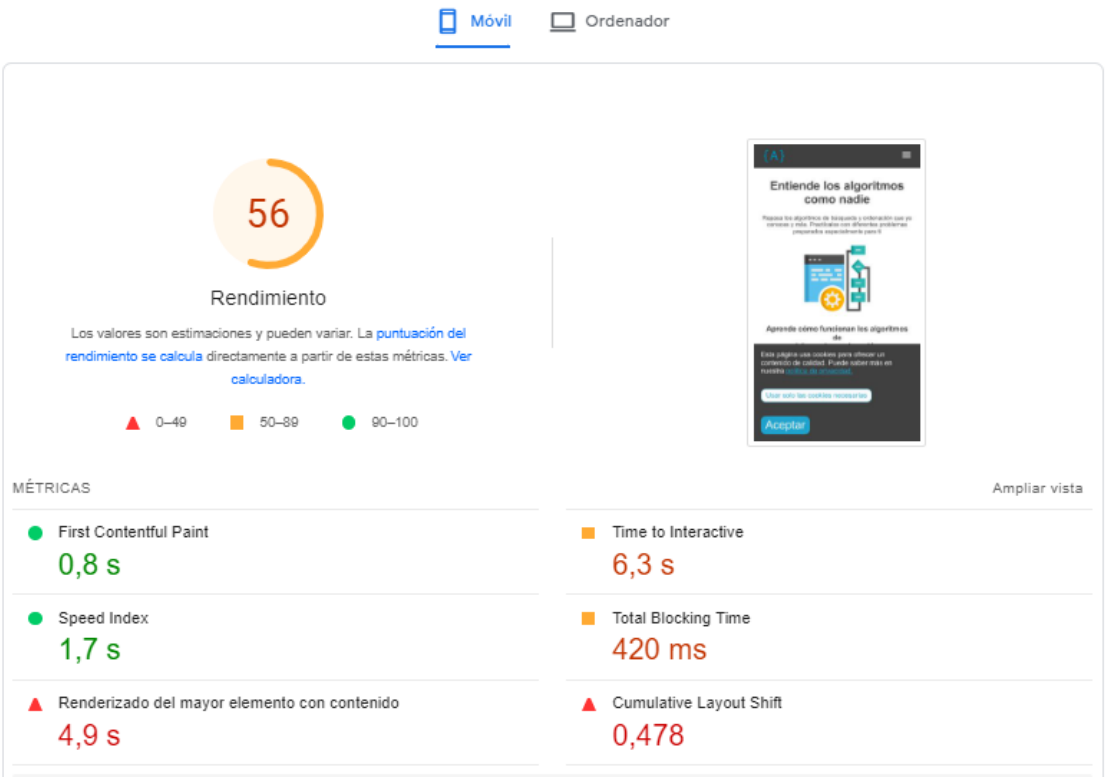


Ilustración 35 Test PageSeed Insights Móvil

Se puede ver un excelente resultado en las pruebas para ordenadores, pero un resultado mucho peor en las pruebas para móviles. Esto tampoco ha supuesto un problema muy grande ya que, como se mencionó al principio, esta aplicación está creada para un uso prioritario en ordenadores.

6.8 Funcionalidad de la página

En este apartado se explica paso por paso las diferentes acciones que puede llevar a cabo un usuario dentro de la página. Este apartado es sólo un resumen y se mencionará 1 funcionalidad de cada parte de la web, si se quiere conocer más se recomienda leer el Anexo VI Manual de usuario.

Para separar las diferentes funcionalidades se tienen en cuenta los diferentes paquetes en los que se divide el sistema a excepción del paquete de Gestión de Analytics que se trata de funcionalidad casi completamente automatizada.

6.8.1 Funciones relacionadas con la gestión de la cuenta

En ellas podemos destacar cómo registrarse. Para ello primero presiona sobre el botón “Regístrate” en la barra de navegación como se ve en la Ilustración 36.

Ilustración 36 Botón de registro

A continuación, se te redirigirá a la siguiente página en la que deberás rellenar el formulario como se te indica y acto seguido deberás pulsar sobre el botón “Regístrate” señalado en la Ilustración 37.

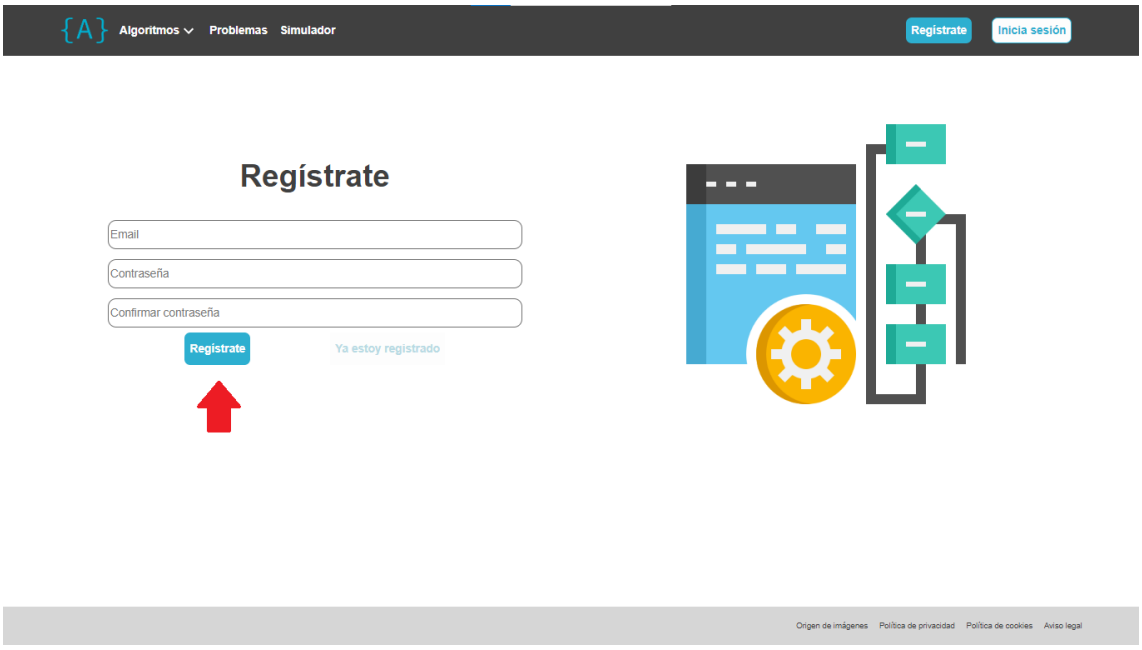


Ilustración 37 Página de registro

Acto seguido te llegará un correo a tu cuenta de correo electrónico en el que deberás pulsar en el enlace destacado en la Ilustración 38 para que la verificación de tu cuenta se haga efectiva.

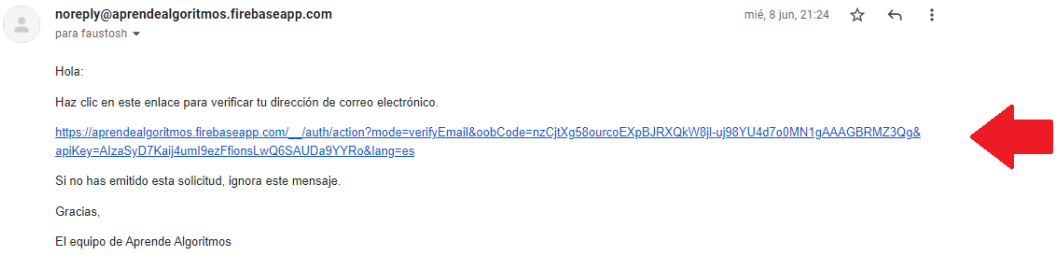


Ilustración 38 Enlace de verificación de correo

Y el proceso se habrá realizado correctamente. Puedes verificar que tu cuenta se ha creado iniciando sesión.

6.8.2 Funciones relacionadas con los algoritmos

En ellas podemos destacar el seleccionar la clase de algoritmo. Para ello se tienen dos formas.

6.8.2.1 Mediante el menú de la barra de navegación

Coloca tu ratón sobre la flecha a la derecha de la sección “Algoritmos” de la barra de navegación y se abrirá un menú desplegable como se ve en la Ilustración 39.

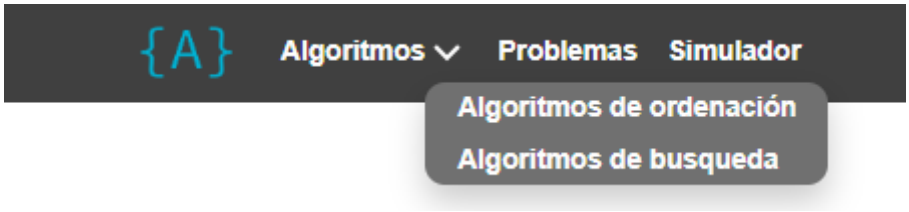


Ilustración 39 Menú desplegable de algoritmos

Si presionas en cualquiera de las opciones de este menú serás redirigido a la página de esa clase de algoritmos.

6.8.2.2 *Mediante las pestañas de la página de algoritmos*

En la página de algoritmos, independientemente de la clase, podrás encontrar dos pestañas que citan “Algoritmos de ordenación” y “Algoritmos de búsqueda”. Presiona sobre cualquiera de ellas y serás redireccionado a la página correspondiente.

Ten en cuenta que la pestaña resaltada en blanco corresponde con la clase de algoritmos en los que estás en ese momento como puedes ver en la Ilustración 40.

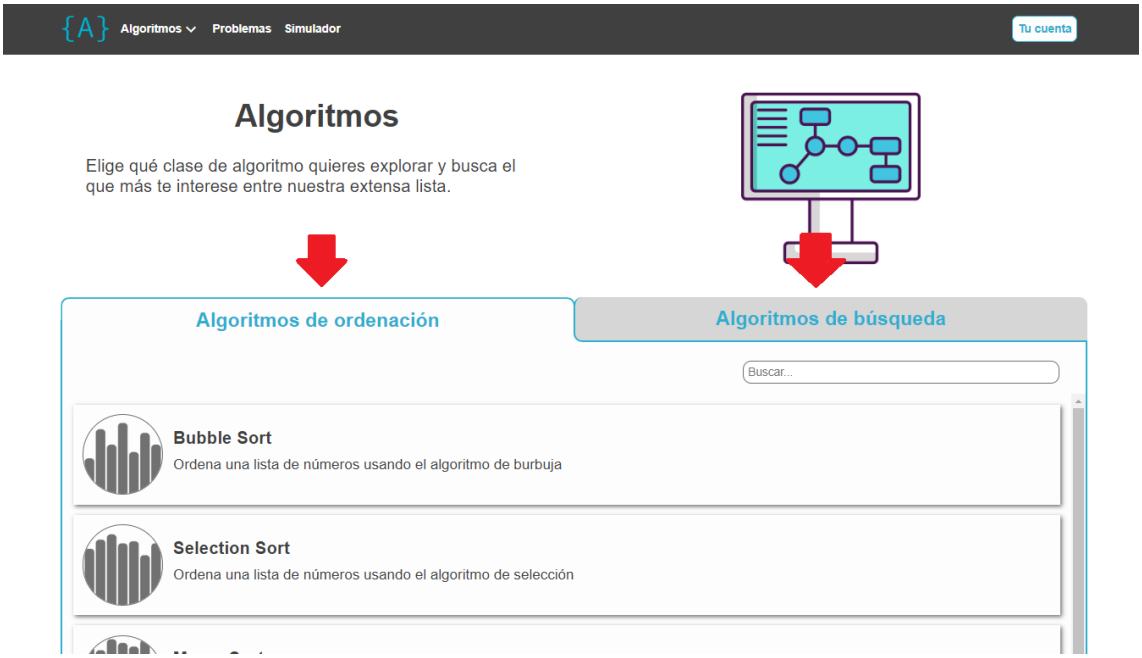


Ilustración 40 Página de algoritmos con pestañas

6.8.3 Funciones relacionadas con el código

En esta sección se puede destacar la función de cambiar el lenguaje de programación. Para ello, cuando veas una sección de código ya sea en la página de un algoritmo o en la página de un problema, puedes cambiar el lenguaje de programación haciendo clic sobre el desplegable de la parte superior derecha del editor de código señalado en la Ilustración 41.

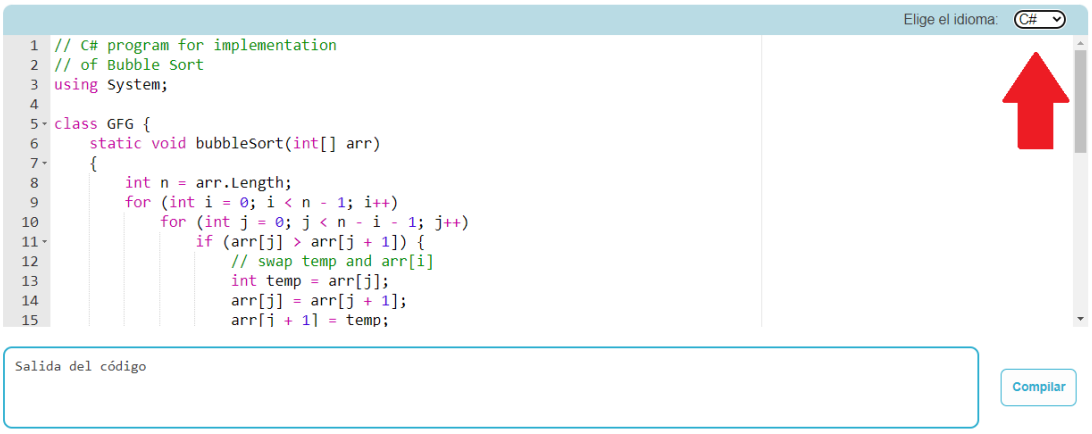


Ilustración 41 Selector de lenguaje de programación

A continuación, selecciona el lenguaje en el que quieras ver el código en el desplegable que se puede ver en la Ilustración 42. No te preocupes si has editado el código, tus cambios no se perderán a menos que actualices la página.

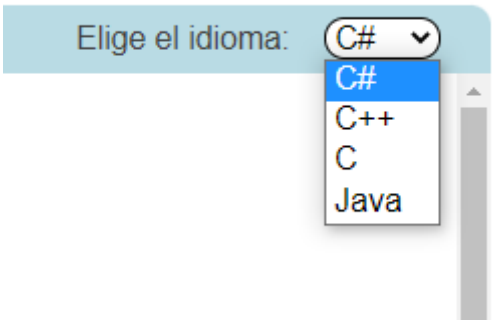


Ilustración 42 Desplegable del selector de lenguaje de programación

6.8.4 Funciones relacionadas con los problemas

Podemos destacar la funcionalidad de ver los problemas disponibles. Para poder ver los problemas disponibles primero asegúrate de tener la sesión iniciada. A continuación, presiona sobre el botón “Problemas” de la barra de navegación señalado en la Ilustración 43.

Ilustración 43 Botón de problemas

Seguidamente se te redirigirá a la página de problemas en la que, si haces scroll hacia abajo podrás ver los problemas disponibles listados como se puede ver en la Ilustración 44.

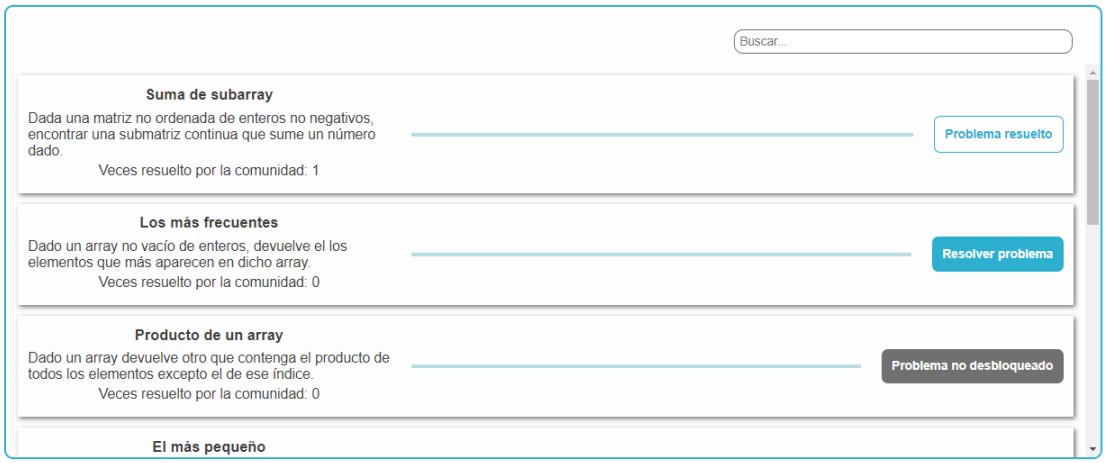


Ilustración 44 Listado de problemas disponibles

6.8.5 Funciones relacionadas con la simulación.

Podemos destacar la funcionalidad de personalizar la simulación. Para ello deberás primero haber un elegido un algoritmo para simular. Una vez hecho esto, podrás escribir los números a utilizar en la barra de input de la parte inferior señalada en la Ilustración 45.

Ilustración 45 Input para personalizar simulación

El formato en el que deberás escribir el input será **obligatoriamente** el siguiente:

número, número, número, … , número

Y en el caso de que se trate de un algoritmo de búsqueda, podrás introducir el número a buscar al final de la siguiente forma:

número, número, número, … , número, {búsqueda}

Por ejemplo, si queremos que el array de números a utilizar sea el 1,2,3,4 y el número a buscar el 5 entonces nuestro input sería:

1,2,3,4,{5}

La cantidad de números introducidos (sin contar el número a buscar) no puede exceder los 10.

Si se quiere saber más sobre el resto de las funcionalidades disponibles en la aplicación se recomienda leer el Anexo VI Manual de usuario.

7. Conclusiones y líneas de trabajo futuras

En este apartado se comenta todo aquello que se ha logrado conseguir durante la realización de este proyecto, así como los posibles cambios que se producirán en este en el futuro.

7.1 Conclusiones

En cuanto a los logros conseguidos durante la realización de este proyecto podemos destacar:

- La creación de un simulador gráfico de algoritmos completamente desde 0 que no se basa en el uso de animaciones anteriormente guardadas en el sistema, sino que se basa en el cálculo en tiempo real de todos los frames con sus correspondientes explicaciones para cada algoritmo.
- La implementación de un editor de código y su conexión a un compilador online permitiendo así la existencia de un IDE online dentro de la página.
- La implementación de un sistema de problemas desbloqueables con un estilo parecido al de otras páginas top de la actualidad.
- La redacción y presentación de explicaciones de diferentes algoritmos de ordenación y búsqueda con ejemplos tanto de ejecución como de código en diferentes lenguajes de programación.

A título personal también quiero destacar el hecho de que he alcanzado un conocimiento mucho más profundo de las herramientas utilizadas y he entendido partes teóricas en las que anteriormente tenía ciertas dudas.

7.2 Líneas de trabajo futuras

El proyecto todavía tiene partes que pulir y funcionalidad que se le podría añadir, entre las que podemos destacar:

- Extender la funcionalidad del simulador para que no trabaje sólo con arrays sino también permita simular algoritmos para árboles, listas enlazadas, etc.
- Perfeccionar el método de resolución de problemas para que no acepte sólo una respuesta posible, sino que haga varias comprobaciones al igual que hacen otras páginas.

- Aumentar la cantidad de categorías de algoritmos para que no se queden sólo en ordenación y búsqueda.
- Migrar las explicaciones de los algoritmos del código a la base de datos para mejorar así su extensibilidad y crear un editor online como el de los problemas.
- Crear contenido de pago con su correspondiente conexión a plataformas de pago como Stripe para así monetizar la página.
- Permitir la edición de la lista de usuarios administradores desde dentro de la web por un usuario “superAdministrador”

8. Referencias

[1] Wikipedia contributors. (s. f.). React. Wikipedia, The Free Encyclopedia. https://es.wikipedia.org/w/index.php?title=React&oldid=143695921

[2] Qué es TypeScript. (s. f.). CódigoFacilito. Recuperado 1 de julio de 2022, de https://codigofacilito.com/articulos/typescript

[3] Qué es UML: Unified Modeling Language. (2018, septiembre 11). Openwebinars.net. https://openwebinars.net/blog/que-es-uml-unified-modeling-language/

[4] Francisco José García Peñalvo, PhD. María N. Moreno García, PhD. Jesús F. Rodríguez-Aragón, PhD. Carolina Zato Domínguez. (2020). Tema 5: Proceso Unificado. https://moodle2.usal.es/pluginfile.php/523200/mod_resource/content/2/Transparencias/IS_I%20Tema%205%20-%20Proceso%20Unificado_2020.pdf

[5] Cloud application platform. (s. f.). Heroku.com. Recuperado 16 de junio de 2022, de https://www.heroku.com/

[6] Npm: Create-react-app. (s. f.). Npm. Recuperado 16 de junio de 2022, de https://www.npmjs.com/package/create-react-app

[7] (S. f.). Microsoft.com. Recuperado 29 de junio de 2022, de https://www.microsoft.com/es-es/microsoft-365/project/project-management-software

[8] ezestimation. (s. f.). Google.com. Recuperado 29 de junio de 2022, de https://sites.google.com/site/ezestimation/

[9] Home. (s. f.). TypeDoc Documentation Generator. Recuperado 30 de junio de 2022, de https://typedoc.org/

[10] Ideal modeling & diagramming tool for Agile team collaboration. (s. f.). Visual-paradigm.com. Recuperado 6 de junio de 2022, de https://www.visual-paradigm.com/

[11] (S. f.-b). Microsoft.com. Recuperado 4 de julio de 2022, de https://www.microsoft.com/es-es/microsoft-365/word

[12] Extensions, L. M. A. (s. f.). Visual Studio Code - code editing. Redefined. Visualstudio.com. Recuperado 4 de julio de 2022, de https://code.visualstudio.com/