GRADO EN INGENIERÍA TELEMÁTICA

Escuela Técnica Superior de Ingeniería de Telecomunicaciones

Curso académico 2021-2022

Trabajo fin de grado

Desarrollo de nuevas funcionalidades en juez online de programación competitiva

Tutor: Raúl Martín Santamaría
CoTutor: José Manuel Colmenar Verdugo
Autor: Rufino García Sánchez

Agradecimientos

Ante todo agradecer el apoyo de mi familia desde el primer momento en todas las decisiones que he tomado; siempre han sido un pilar fundamental durante mi vida en todos los sentidos y sin ellos no hubiera sido posible absolutamente nada, ni hubiera conseguido lo que ha día de hoy soy y me siento orgulloso de tenerlos y poder ver todo lo que he logrado.

Les agradezco a mis tutores en concreto a Raúl Martín Santamaría por su esfuerzo y dedicación guiándome y por ofrecerme la oportunidad de trabajar en este proyecto. Gracias al camino por el que me han llevado y guiado ya que han aguantado cuando he estado fuera por problemas y aún así a la volver me han seguido acompañando y levantándome en los baches que he ido teniendo.

También quiero agradecer a mis abuelos por haberme ayudado, haberme dado siempre sabios consejos y a mis compañeros de universidad los cuales siempre me han apoyado y hemos tenido una bonita y cariñosa relación, siendo esta como una simbiosis.

Muchas gracias por todo, me siento afortunado de tener gente tan buena a mi lado.

Mi madre y mi padre;
mis pilares fundamentales

Madrid, Viernes 15 de Julio de 2022
Rufino García Sánchez

Resumen

Este trabajo final de grado consiste en añadir nuevas funcionalidades y mejoras intuitivas, sencillas, atractivas y modernas, realizando tareas de desarrollo web y mantenimiento web en la aplicación llamada iudex.

En el documento, el cual nos encontramos, se explican los fundamentos y bases del proyecto en el cual se ha trabajado junto con las herramientas y tecnologías usadas. También los requisitos y motivaciones. Problemas y baches durante el camino y soluciones aplicadas a ellos. En este Trabajo de Fin de Grado (TFG) se continúa del desarrollo de un proyecto software existente, extendiendo su funcionalidad y añadiendo nuevas características.

Esta memoria está compuesta por cinco capítulos, en los que se presenta una forma diferente, moderna, eficaz y segura de evaluar en el ámbito educativo problemas de programación. En este proyecto nos hemos centrado en la ampliación de funcionalidades y mejora del código de un juez de programación desarrollado para el ámbito educativo, pero que su uso se puede extender a muchos más campos.

Además de esto, se explicará en una breve introducción de que se trata el juez, también trataremos los objetivos que se pretenden obtener, que metodología se ha llevado a cabo para obtenerlos y que herramientas y tecnologías se han usado.

Esta memoria finalizará con los diferentes resultados obtenidos, aplicando las nuevas implementaciones y herramientas, con las diferentes conclusiones que se habrán obtenido tras la finalización del proyecto y con una proyección a futuro de cara a seguir explorando el juez e introduciendo mejoras que lo hagan más intuitivo, seguro y eficaz.

Palabras Clave

Juez (the judge o iudex), juez automático, aplicación web, web, internet, programación competitiva, programación web, mejoras, pruebas, nuevas funcionalidades, correcciones, lenguajes de programación, funcionamiento, problemas encontrados, configuración del entorno, Docker, contenedores, imágenes, desarrollo web, mantenimiento, logs, bugs, test, test unitarios, test de integración, Spring Boot, Springdoc, SonarCloud, Java, JavaScript, Spring, Git/GitHub, Agile, BBDD, WebSockets, Relaciones de las entidades, OneToMany, ManyToMany, ManyToOne, etc.

Conceptos Básicos

Internet

«Internet es una gran red internacional de ordenadores,Permite, como todas las redes, compartir recursos. Es decir: mediante el ordenador, establecer una comunicación inmediata con cualquier parte del mundo para obtener información sobre un tema que nos interesa» [47].

Web

La World Wide Web o simplemente WWW o Web es uno de los métodos más importantes de comunicación que existe en Internet. Fue fundada y creada por Tim Berners Lee en año 1990 [2].

La forma más común para conectarse a Internet es a través de una línea telefónica. Los usuarios potenciales pueden solicitar el servicio a un proveedor de conexiones y con un simple navegador y una URL junto con la conexión anterior ya estarías dentro de la red [42].

Lenguajes de programación

Los lenguajes de programación son las herramientas básicas de los programadores. Hay una inmensa variedad de lenguajes; mas o menos tipados, orientados a objetos, en conclusión los más utilizados son C, Java y Python.

Hojas de estilos

Una hoja de estilo es un archivo de extensión *.CSS (CSS, Cascading Style Sheets = Hojas de estilo). Se caracterizan por presentar los siguientes contenidos: tipo, fuente y tamaño de la letra, la alineación y posicionamiento de las palabras y el texto, colores y fondos, etc [10].

Base de datos

Llamamos base de datos a una colección organizada de información estructurada, o datos, típicamente almacenados electrónicamente en un sistema de computadora.La mayoría de las bases de datos utilizan lenguaje de consulta estructurado (SQL) para escribir y consultar datos.

Existen diferentes tipos de bases de datos pero los mas usados a día de hoy son la bases de datos llamada SQL y las NO SQL.

API

El término API se puede definir como un conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción. Las siglas del termino API provienen del inglés, Application Programming Interface.

REST API

«REST API es una API creada por el informático Roy Fielding en el año 2000, la cual se ajusta a los límites de la arquitectura REST o Transferencia de Estado Representacional (en inglés Representational State Transfer) y permite la interacción con los servicios web de RESTful» [29].

Bibliotecas

Una biblioteca o librería agrupa muchas clases que su uso de cara a los programadores les facilita el trabajo y son muy útiles para agilizar el mismo. En java hay distintas bibliotecas útiles como JUnit y Mockito.

CI/CD

CI/CD es un método para entregar aplicaciones con frecuencia a los clientes mediante la introducción de la automatización en las etapas de desarrollo de aplicaciones. Los principales conceptos atribuidos a CI/CD son integración continua, entrega continua e implementación continua.

Desarrollo software

Podemos definir al desarrollo de software como a un conjunto de actividades informáticas dedicadas al proceso de creación, diseño, despliegue y compatibilidad de software.

Desarrollo web

El desarrollo web se refiere en general a las tareas asociadas con el desarrollo de sitios web para hospedaje a través de internet. El proceso de desarrollo web incluye diseño web, desarrollo de contenido web, secuencias de comandos del lado del cliente/del lado del servidor y configuración de seguridad de la red, entre otras tareas.

El desarrollo web se divide en: *frontend* y *backend*.

frontend

El desarrollo *frontend* es un estilo de programación informática que se centra en la codificación y creación de elementos y funciones de un sitio web que luego verá el usuario.

backend

El desarrollo de *backend* se enfoca en el lado oculto del sitio web que los usuarios no pueden ver. Es lo que hace que un sitio sea interactivo. También puede referirse al *backend* como el "lado del servidor" de un sitio web.

Página web

Una página web es un documento, comúnmente escrito en HTML, que se visualiza en un navegador de Internet. Se puede acceder a una página web ingresando una dirección URL en la barra de direcciones de un navegador. Una página web puede contener texto, gráficos e hipervínculos a otras páginas web y archivos.

Una página web a menudo se usa para proporcionar información a los espectadores, incluidas imágenes o vídeos. Una página web también se puede utilizar como método para vender productos o servicios.

Servicios web

«Se dice que un servicio web es un tipo de tecnología que, a través de ciertos protocolos y estándares, habilita la comunicación entre distintas computadoras y permite intercambiar datos entre ellas».

Existen distintos tipos de protocolos y estándares utilizados en los servicios web, destacando el más común y usado que es HTTP.

Aplicaciones web

Una aplicación es un programa diseñado e implementado para realizar operaciones o funciones específicas. Generalmente, son diseñadas para facilitar tareas complejas y/o rutinarias.

Plantillas web

Las plantillas son documentos HTML. Estas plantillas son diseños previamente diseñados que permiten organizar el contenido en una página web para crear un sitio web.

Motor de plantillas

Es un software diseñado para combinar plantillas. su funcionamiento es muy simple, leen un fichero de texto y obtienen los datos del mismo, e inserta en el fichero toda la información que obtenga del controlador modificando así la plantilla.

Índice de contenidos

Índice de figuras

1 Introducción

"No hay preguntas sin respuestas, solo preguntas mal formuladas"

Morfeo, *The Matrix*

A día de hoy nos encontramos en una situación bastante interesante, ruda, diferente, escalofriante, tenebrosa y a la vez apasionante y desafiante, en conclusión, ante un paradigma que nunca habíamos vivido antes y no lo hubiéramos pensado.

Después de una pandemia y de estar meses confinados. Podemos asegurar con conocimiento de causa que hemos realizado trabajos, asistencias a "clases online", cervezas con los compañeros, risas, juegos, el grandioso teletrabajo que parece que ha venido a quedarse desde casa. A mí en un primer momento nunca se me habría ocurrido ni se me habría pasado por la cabeza la situación que nuestra generación ha sobrellevado y vivido. A todo esto hay que sumarle aquel que por desgracia haya pasado la maldita enfermedad y maldita lo digo por que por desgracia se ha llevado por delante a innumerables personas.

Es un hecho que la generación en la cual nos encontramos y yo me incluyo; ha sufrido mucho y estas situaciones marcarán un antes y un después a nivel generacional.

Ya puestos un poco en contexto en el infierno vivido, vamos a explicar por que decidí empezar este trabajo de fin de grado junto con mis dos tutores; Raúl Martín Santamaría y José Manuel Colmenar Verdugo.

Debido a la situación surgió el primer contacto con la programación competitiva, en la cual siempre me había llamado la atención, pero nunca había dado el paso a probar y lanzarme. Un día un poco inspirado me lancé y decidí probarlo. Aquel día la programación competitiva cambio en cierto modo para mí. He de reconocer que alguna vez había escuchado y observado alguna pincelada sobre este término y como se llevaban a cabo los concurso gracias a algún compañero de clase, pero fue ahí en ese mismo instante cuando me empecé a introducir un poco en este apasionante mundo, muy desconocido, y a practicar y practicar ejercicios y problemas.

A todo esto le empecé a introducirme más en este mundo y es ahí cuando conocí gracias y mediante a un grupo de Telegram de la universidad a mis dos tutores que a día de hoy me han acompañado y están a mi lado presentando este Trabajo de fin de Grado.

Que ahora mismo diga que son mis tutores es debido a que un día en una simple conversación que no iba a ningún lado, surgió y salió el tema sobre Trabajos de fin de grados. Y si mal no recuerdo fue al día siguiente cuando les comenté que buscaba uno y así surgió.

Entonces, comenzamos a trabajar de manera conjunta ellos como mis tutores y yo como su alumno en una aplicación que llevaban ya un tiempo desarrollando y que estaba en proceso de desarrollo, llamada Iudex, y que en relación a la competición competitiva decidí centrar todos mis esfuerzos, conocimientos y pasión en desarrollar mi tesis en él.

A lo largo de este documento, el lector se va a encontrar principalmente con nuevas funcionalidades añadidas en la aplicación. Las mejoras y nuevas implementaciones. Por lo que observará como se mejora la implementación del código y la calidad del mismo.

Para ponernos en contexto, este proyecto está dirigido al sector educativo, es de software libre y día a día mejorando en todos los niveles de la aplicación. El desarrollo de esta aplicación es meramente de carácter educativo, de cara a usar la programación y la aplicación como un sistema único y en simbiosis con la idea de obtener beneficios para estudiantes y profesores.

Cabe destacar que más tarde explicaremos en detalle que esta aplicación también se puede usar de cara al ámbito laboral, aunque este no haya sido el principal objetivo, ni se haya desarrollado con ese fin. Se podría usar perfectamente para procesos de selección de personal en una empresa.

La utilización de la aplicación en el ámbito el cual ha sido principalmen-

te desarrollada es viable y eficaz, sobre todo cuando hablamos de concursos de programación competitiva, prácticas en clase, exámenes, validaciones de código, comprobación de su calidad, etc.

La estructura de la memoria consta de 5 capítulos. Los capítulos comenzarán con una introducción 1 seguido de los objetivos 3 que se pretenden conseguir con la realización del proyecto. A continuación el lector se encontrará con el marco teórico y estado del arte 2, para continuar con la Descripción del trabajo desarrollado 4, donde explicaremos detalladamente el proyecto. Se finalizará con las conclusiones 5.

2

Marco teórico y estado del arte

En este capítulo explicaremos en profundidad en que consiste la aplicación y los antecedentes de la misma. Explicaremos los entorno utilizados y las tecnologías planteadas inicialmente y más tarde derivadas. Esto sucedió con Visual Studio Code [5] la cual se dejó de utilizar migrando a IntelliJ IDEA [44], siendo una migración satisfactoria, ya que se aumentaron muchas funcionalidades y ayudas con este último programa que con Visual Studio Code.

Se ha desarrollado con Java, Spring Boot, Docker y Kubernetes. Estas últimas herramientas se han estado utilizando para levantar y abrir la aplicación en un entorno controlado y para comprobar que se estaba ejecutando de una manera correcta y segura.

2.1. ¿Qué es la Programación competitiva?

La programación competitiva se suele llevar a cabo y evaluar en Concursos de programación competitiva. Estos eventos suelen celebrase a través una red local o de internet. Los concursantes de los eventos se conocen como programadores deportivos de competición.

En estos concursos los concursantes resuelven preguntas de programación lógicas o matemáticas en el lenguaje de codificación de su elección en un tiempo limitado.

El número de preguntas varía en función del concurso. Los resultados se evalúan y se juzgan por varias directrices, las cuales destacan estas:

- El número de preguntas resueltas
- El tiempo empleado en resolver los problemas
- Otros factores, como el tiempo de ejecución, el tamaño del programa, en cuánto a líneas de código, etc.

2.2. ¿Qué es un juez de código en linea?

Un juez de programación en línea se trata de un software (web, habitualmente) que emite un veredicto sobre la corrección y eficiencia de la solución a problemas de programación realizados por unos participantes [43].

Esto los proporciona el creador de los problemas y administrador del sistema, mientras que las diferentes soluciones que se obtienen son proporcionadas por los participantes o usuarios que han resuelto el problema.

Por lo tanto, podemos afirmar que está compuesto por un corrector automático, así como por un repositorio con absolutamente todos los enunciados de los problemas disponibles y listos para la realización de los mismos.

La naturaleza de la mayoría de estos ejercicios es lógica y matemática.

Estos problemas se dividen en diferentes formatos a su vez, como pueden ser problemas de combinatoria, teoría de números, algoritmia o estructuras de datos [24].

2.2.1. Jueces de Programación competitiva

Se trata del uso más extendido de estos programas, donde equipos de participantes se enfrentan a los mismos problemas, el objetivo es ser capaz de resolverlos de una manera más eficaz, completa, correcta y rápida posible antes que el contrincante.

El veredicto final de dichos sistemas es habitualmente llevado a cabo en máquinas o "jueces de programación", tras someter el resultado a extensos casos de prueba.

La respuesta se suele basar en un sistema binario, 0 o 1, es decir, **todo o nada**. Solo considera correcta una solución, si supera todos las pruebas sin excepción alguna. Sin embargo, puesto que este comportamiento es algunas veces un poco extremista o radical. Ante esto, algunos recurren a distintos sistemas de puntuación que premian que en cierto modo sea correcto y funcione.

Estos concursos tienen un crecimiento cada vez mayor. A día de hoy no presenta límite, habiendo participado más de 75.000 personas en las fases clasificatorias de algunos de ellos en el pasado trimestre. Y con una exponencial proyección de crecimiento de aquí a poco tiempo.

Diferentes tipos de jueces

Existen dos tipos de jueces, los jueces de programación competitiva y los jueces educativos [36].

Los jueces educativos ofrecen a los profesores una herramienta con la que impartir a los alumnos conocimientos, les permiten la creación de ejercicios (normalmente de iniciación a la programación o de cara a exámenes) para que estos puedan resolver con mayor flexibilidad y rapidez [21].

El uso de estos sistemas proporciona ventajas, algunas de estas pueden ser:

1. Provee de una mayor objetividad y rigurosidad a la enseñanza, permitiendo a los profesores verificar con facilidad la exactitud de las soluciones, aliviando la carga de trabajo del profesor.
2. Los estudiantes reciben una retroalimentación casi inmediata de manera que puedan realizar más rápidamente las correcciones y, por tanto, los problemas [25].
3. La disponibilidad del juez es absoluta y al 100x100, esto implica que no disponen de nadie y que su tiempo no está limitado al horario lectivo. Esto permite a alumnos entregar los ejercicios realizados desde cualquier parte del mundo en cualquier momento y recibir un feedback instantáneo sin necesidad del profesor estar operativo para corregirlo.

Tanto en los jueces de programación competitiva como los jueces educativos ya existen diferentes plataformas que están operativas. Estas funcionan eficaz y correctamente. Estos tipos de jueces son totalmente automáticos y autónomos [8].

Dentro de todos los jueces que a día de hoy existen, los más usados y completos son **CodeForces 2.1 [32], DOMjudge [37] y Kattis 2.2 [13] en el ámbito empresarial y por parte de los jueces educativos estarían ¡Acepta el**

reto! 2.4 [20] y Online-judge-mean [18], HackerRank [30], CodeChef 2.3 [15], TopCoder [28].

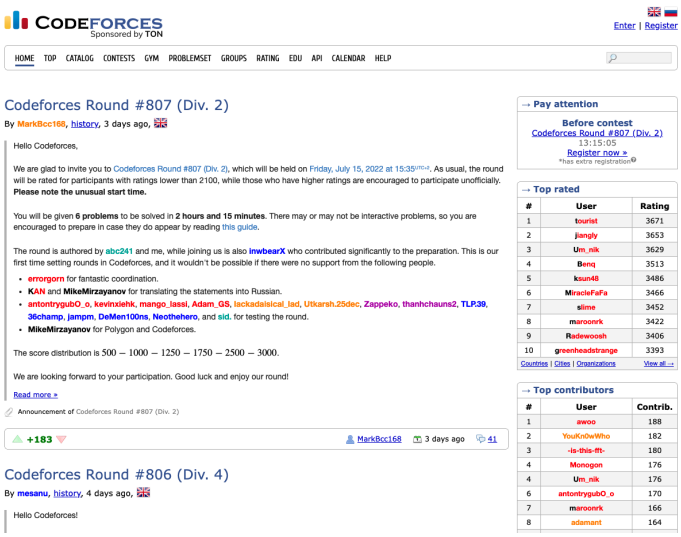

Figura 2.1: Plataforma de CodeForces

Figura 2.2: Plataforma de Kattis

En estas figuras 2.1, 2.2, 2.3, 2.4 podemos observa las representaciones del *frontend* por parte de los jueces online mas usados a nivel mundial. se puede observa las grandes diferencias entre aquellos que son de software libre y los que obtienen unos beneficios sobre ellos.

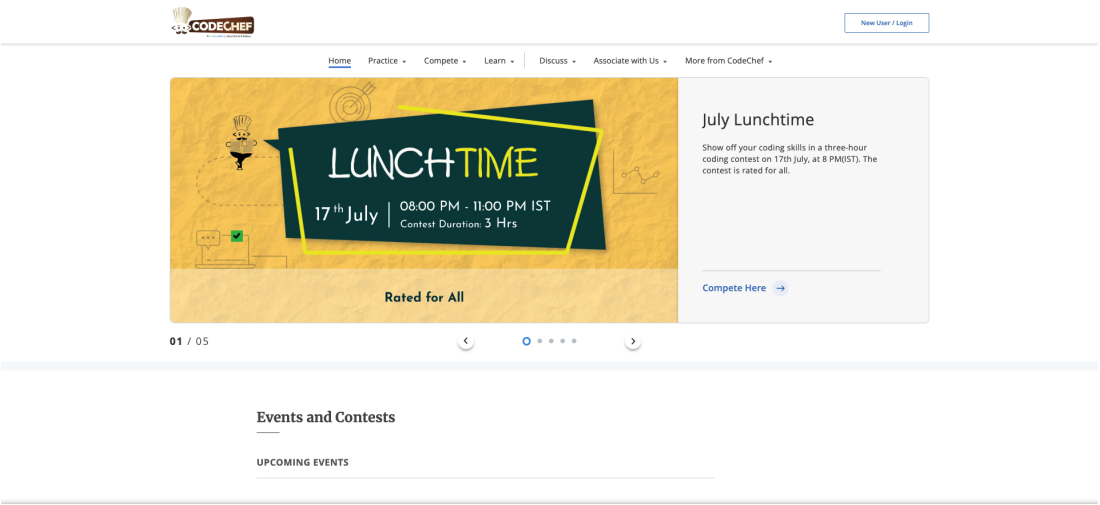

Figura 2.3: Plataforma de Codechef

Figura 2.4: Plataforma de ¡Acepta el reto!

2.3. Concursos, tipos y plataformas

En los concursos de programación competitiva los participantes compiten para resolver los problemas de programación. Por cada problema se proporciona un enunciado específico, este enunciado lo deben usar los participantes para diseñar e implementar un algoritmo eficiente que resuelva dicho problema. Dicho algoritmo a su vez se entrega al juez automático que dispone de una serie de casos de prueba con las soluciones esperadas [43].

El juez va a compilar, ejecutar y evaluar el código generando una salida que indica si el código es correcto o no, es decir, si pasa el algoritmo y la salida coincide o, por lo contrario, es erróneo y la salida no coincide. Existen muchos concursos de programación competitiva.

2.3.1. Ejemplos de Concursos

ACM - ICPC

El Concurso Internacional de Programación Universitaria [34], conocido como ICPC, es una competencia anual de programación competitiva de varios niveles entre las universidades del mundo. Con sede en la Universidad de Baylor, dirigida por el Director Ejecutivo del CIPC y Profesor de Baylor, el Dr. William B. Poucher, el CIPC organiza concursos regionales autónomos que cubren seis continentes y culminan en una Final mundial mundial cada año. En 2018, la participación del CIPC incluyó a 52.709 estudiantes de 3233 universidades en 110 países.

El ICPC se fundó en 1970 cuando la primera competencia fue organizada por pioneros del Capítulo Alfa de la Sociedad de Honor de Ciencias de la Computación de la UPE. La iniciativa se difundió rápidamente en los Estados Unidos y Canadá como un programa innovador para aumentar la ambición, la aptitud para resolver problemas y las oportunidades de los estudiantes más fuertes en el campo de la computación.

HashCode

Se trata de un concurso que se prepara todos los años de programación que es organizado por Google para personas mayores de edad en su país. Se participa en grupos de 2 a 4 personas.

Para cada ronda un problema es entregado y para tener la posibilidad de poder participar en el mundial de HashCode [4] es necesario superar la fase de cualificación y tener como mínimo 2 miembros en el equipo.

Kaggle

Kaggle [6] es una plataforma comunitaria en línea para científicos de datos y entusiastas del aprendizaje automático creada por Google.

Kaggle permite a los usuarios colaborar con otros usuarios, buscar y publicar conjuntos de datos competir con otros científicos de datos para resolver los desafíos de la ciencia de datos, también llamada programación.

Facebook Hacker Cup

Facebook Hacker Cup [17] es un concurso de anual de programación competitiva. Facebook comenzó a organizar concursos de programación desde 2011 para identificar a los mejores talentos de ingeniería para posibles empleos en Facebook.

Olimpiada Internacional de Informática (IOI)

La Olimpiada Internacional de Informática [12] es una de las Olimpiadas Internacionales de Ciencias, que tiene como objetivo potenciar el aprendizaje de la informática en alumnos de secundaria y preparatoria.

La La Olimpiada Internacional de Informática es una competencia anual organizada cada año por uno de los países participantes que normalmente envían una delegación de 4 alumnos y 2 adultos que los acompañan. En la IOI los problemas están orientados a los algoritmos. Para participar en la IOI se debe clasificar a través de las olimpiadas nacionales como la Olimpiada Informática Española (OIE).

La Olimpiada Informática Española es el evento anual de programación más importante de España.

2.4. Como se evalúan los concursos y la obtención de los resultados

Los concursos se evalúan en función de la respuesta del juez a la resolución de los problemas entregados por parte de los concursantes y se divide en dos, **aceptado o no aceptado**:

- **Aceptado**:

  Si la respuesta por parte del juez de la resolución del problema por parte del concursante es aceptado, indica que la respuesta y la solución es correcta y que la salida son las esperadas

- **No Aceptado**:

  No aceptado, en cambio, implica que la resolución del problema es incorrecta y que no supera el algoritmo establecido por el juez. Puede haber diferentes causas, entre las que se encuentran fallos de presentación, fallos de compilación, errores en ejecución, errores en compilación, etc.

2.5. Comparativa entre iudex y otros jueces de código

2.5.1. Alcance y uso

Los jueces mencionados en este documento están disponibles como plataformas web. [31].

Iudex se ha desarrollado con el objetivo de ayudar y ser útil en el ámbito educativo. Aunque el objetivo inicial solo sea eso, puede llegar a ser de utilidad en distintos ámbitos. Es una aplicación moderna creada con herramientas y tecnologías actuales, segura y eficaz.

Es una aplicación de código libre. Esto implica que cualquier persona podrá utilizarla, tanto particulares como empresas o instituciones educativas.

2.5.2. Entorno de ejecución

En los concursos de programación se ejecuta el código de los problemas realizado por los participantes para poder valorarlos. Uno de los problemas es la ejecución de código de una manera controlada y segura. Para llevar a cabo la ejecución del código de forma segura debemos seguir las siguientes directrices:

- **Filtrar el código**
- **Acotar el entorno de ejecución**

  Consiste en limitar el entorno en el que se encuentra el sistema y donde ejecuta el problema.

- **Utilizar la virtualización**

  Este es el proceso que consiste en ejecutar instancias virtuales de un entorno abstrayéndolo de la capa física del mismo. Es la técnica más moderna y eficaz que se conoce actualmente. Únicamente le es necesario un sistema que se encargue de construir los entornos de ejecución para cada entrega de los problemas.

La mayoría de los jueces se basan en la técnica de modificar y acotar el entorno de ejecución, pero están sujetos a que les envíen malware y eso implica que pueda haber algunos problemas y fallos [23].

2.5.3. Lenguajes soportados

Los jueces normalmente soportan varios tipos de lenguajes (los más famosos y comunes son C, C++, Java y Python). Pero, cada vez al aumentar la cantidad de gente involucrada en los concursos de programación, se está viendo aumentado también lenguajes disponibles dentro de los jueces, de cara a no excluir a ningún participante y exista un mayor abanico de posibilidades [11].

Iudex es un juez flexible y extensible. Extensible debido a que permite y es capaz de poder integrar más lenguajes de programación.

2.6. Mantenimiento, Logs y Pruebas

El mantenimiento [9] software es la modificación después de la entrega, para corregir errores, mejorar el rendimiento u otros atributos.

El mantenimiento es un proceso continuo y necesario para dar continuidad de servicio, dar soporte y ayuda a las peticiones de los usuarios y dar facilidades de futuros mantenimientos [7].

Tipos de mantenimiento

Mantenimiento correctivo

El mantenimiento correctivo consiste en modificar el software tras la detección de defectos, ambigüedades o errores [38].

Mantenimiento adaptativo o evolutivo

El mantenimiento adaptativo [19] consiste en modificar el sistema para acomodarlo a cambios físicos del entorno.

Mantenimiento perfectivo

El mantenimiento perfectivo [26] consiste en mejorar el software para cumplir con las nuevas necesidades o requerimientos.

Mantenimiento preventivo

El mantenimiento preventivo [35] es una técnica científica del trabajo industrial, que en especial está dirigida al soporte de las actividades de producción y

en general a todas las instalaciones empresarias.

Mantenimiento estructural

Este consiste en modificar la arquitectura interna del sistema para aumentar y mejorar la mantenibilidad del código [14]. Es muy importante este mantenimiento, ya que un código debe de ser mantenible en el tiempo.

2.6.1. Logs

En informática se utiliza el término registro, *log* o historial de *log* para referirse a la grabación secuencial en un archivo o en una base de datos de todos los acontecimientos que afectan a un proceso particular [46].

Niveles de log

Existen distintos niveles de *logs* que proporcionan información adicional sobre el suceso: [22].

- **TRACE:**

  Se utiliza durante el desarrollo para rastrear errores.

- **DEBUG:**

  Se utiliza durante la depuración para registrar acciones o eventos que se van produciendo en el sistema.

- **INFO:**

  Se utiliza para registrar las acciones de los usuarios o las acciones específicas del sistema.

- **NOTICE:**

  Se utiliza en producción para registrar todos los eventos notables que no se consideren error.

- **WARNING:**

  Se utiliza para registrar todos los eventos que potencialmente se pueden convertir en errores.

- **ERROR:**

  Se utiliza para registrar los errores o los fallos del sistema.

- **FATAL:**

  Se utiliza para registrar errores bloqueantes que pueden tener efectos secundarios en el sistema.

2.6.2. Pruebas

Las pruebas son actividades que se usan para comprobar el correcto funcionamiento del sistema.

Existen varias formas de clasificar las pruebas:

- **Según las características que prueban**

  Pruebas funcionales y pruebas no funcionales.

- **Según el SUT o sujeto bajo pruebas**

  Pruebas de unidad o pruebas unitarias, pruebas de componente, pruebas de integración y pruebas del sistema.

- **Según la ejecución**

  Pruebas manuales y pruebas automáticas.

- **Según los objetivos**

  Pruebas de aceptación, pruebas de humo, pruebas de sanidad.

- **Según el nivel de abstracción**

  Pruebas de caja blanca y pruebas de caja negra.

3
Objetivos

3.1. Objetivos

El objetivo principal de este TFG es desarrollar e implementar nuevas funcionalidades y mejoras, utilizando las tecnologías más livianas, eficientes, modernas, y garantizando el mayor nivel de seguridad. Estas mejoras se implementarán en la aplicación realizando tanto tareas de mantenimiento como de desarrollo web; en concreto, ha habido un mayor desarrollo en la parte *backend*

Gracias al objetivo principal, podemos definir varios objetivos secundarios que queremos conseguir con este proyecto.

- Mejoras de ciertas funcionalidades que ya existían.
- Corrección de fallos que han aparecido al implementar las nuevas funcionalidades.
- Corrección de fallos previos a ninguna actualización.
- Aprender el funcionamiento, gestión y configuración de la aplicación Docker. En concreto de sus contenedores.
- Aprender como funciona la librería Docker-Java.
- Aprender el funcionamiento de RabbitMQ.

- Aumentar los conocimientos sobre Spring Y su utilización e implementación.
- Ampliar los conocimientos interrelaciónales de las identidades.
- Ampliar los conocimientos sobre las bases de datos y sus relaciones en la ejecución del programa.
- Ampliar los conocimientos sobre API Rest, programación web e integración continua.

El objetivo final del proyecto desarrollar una aplicación final de cara a su uso en el ámbito educativo, empresarial y de negocio.

Otro de los principales objetivos cumplidos ha sido ampliar conocimientos principalmente sobre APIS, programación web y redes. No pueden faltar la parte de micro-servicios con Docker y Kubernetes.

4

Descripción del trabajo desarrollado

Este capítulo resume en detalle el proceso de mejora, se habla sobre el juez (El juez, the judge o iudex) y se explican las tareas realizadas, incluyendo la metodología utilizada para el desarrollo del proyecto, así como las nuevas interfaces, herramientas y diseño utilizado.

4.1. Metodología

Se ha llevado a cabo una metodología ágil o *Agile* [1]. Hemos usado el enfoque ágil para el desarrollo de software, debido a que este, busca distribuir de forma permanente sistemas de software en funcionamiento.

En concreto, esta metodología que hemos usado durante el desarrollo de software ha sido de cara a poder proporcionar y conseguir en poco tiempo pequeñas piezas de software con un correcto funcionamiento.

Hemos usado varios marcos ágiles para el desarrollo de software en concreto y la integración/implementación continua (CI/CD). En concreto, para la división del proyecto completo en pequeñas partes utilizamos pinceladas de la metodología *Scrum* [45].

Para el desarrollo de la aplicación se han llevado a cabo ciertas pautas, estas son:

- Plan de elaboración con los objetivos en mano de cara a desarrollar las tares a realizar.
- Creación de distintas ramas para cada tarea.

Siempre se ha querido separar cada tarea de una y otra para que no fuera un caos, por lo que siempre para cada diferente tarea se ha creado una rama independiente de máster o para más tarde corregirla y si era correcta realizar *merge* y fusionarla.

- Anotación de fallos y mejoras.

  La aplicación ya se encontraba en desarrollo, por lo que había una serie de implementaciones y mejoras sugeridas. No obstante, se han ido anotando fallos dentro de las diferentes *issues* en modo de comentarios dentro de GitHub.

- Utilización del tablero Kanban.

  Las ideas que ibas surgiendo se cerraban ahí en forma de *issue*.

  Es una opción que te da GitHub y nosotros hemos hecho para dividir las tareas realizadas en dos partes, un **Sprint 1 y un Sprint 2**, y ahí íbamos poniendo cuáles tareas estaban en proceso, cuáles hechas y terminadas y cuáles aún todavía empezar.

- Creación de un *pull request*.

  Para cada tarea se creó una *pull request*, ya que así comprobábamos y se podía revisar cada rama antes de hacer *merge* e introducirla con la rama principal.

- Contacto vía Teams/Telegram/Outlook de cara a cualquier duda o problema.

Se ha mantenido un contacto constante a través de estas plataformas antes cualquier problema o duda se ha utilizado estas plataformas.

4.2. Iudex

Iudex es un juez de código automático en línea (online) implementando las nuevas corrientes tecnológicas y herramientas actuales y en auge como Docker, Kubernetes, Spring booot, REST APIS, etc.

4.2.1. Implementación

El juez es una aplicación web implementada con HTML, CSS, SCSS, JavaScript, Java, Spring Boot, SQL y MySQL. A dia de hoy, continua en proceso de desarrollo y da soporte a los lenguajes de programación Java, C, C++, Python y SQL [3].

Al final se ha dividido en dos partes llamadas **primer sprint** y **segundo sprint**.

Primer Sprint

Este *sprint* se divide en 3 tareas:

- Dont download dependencies on container rebuild.
- Migrate Springfox to Springdocs.
- Eliminar ManyToManys.

Segundo Sprint

- API Websockets.
- Creación y ampliación de test unitarios y de integración en la aplicación.

Todo esto mientras ha sido llevado a cabo también se ha realizado:

- Corrección de logs y Errores.
- Mejoras de ciertas funcionalidades al implementar las tareas de los Sprints.

4.2.2. Arquitectura

El juez está formado por el *frontend*, el *backend*, el orquestador o el ejecutor, el servicio de mensajería Rabbit y la base de datos MySQL.

El juez funciona junto a Docker y a sus contenedores. Cuando ejecutamos la aplicación, se crean pequeños contenedores con las diferentes partes antes dichas y estas se comunican mediante las diferentes APIs Rest y el paso de mensajes asíncronos implementado con RabbitMQ.

Todas las operaciones entre el *frontend* y el *backend* se realizan a través del protocolo HTTP y el método API Rest. En el *backend* se encuentra el orquestador

una de las piezas clave del juez. En el *backend* se utiliza el servicio de mensajes de Rabbit únicamente para la comunicación con el orquestador y MySQL Connector para la conexión con la base de datos.

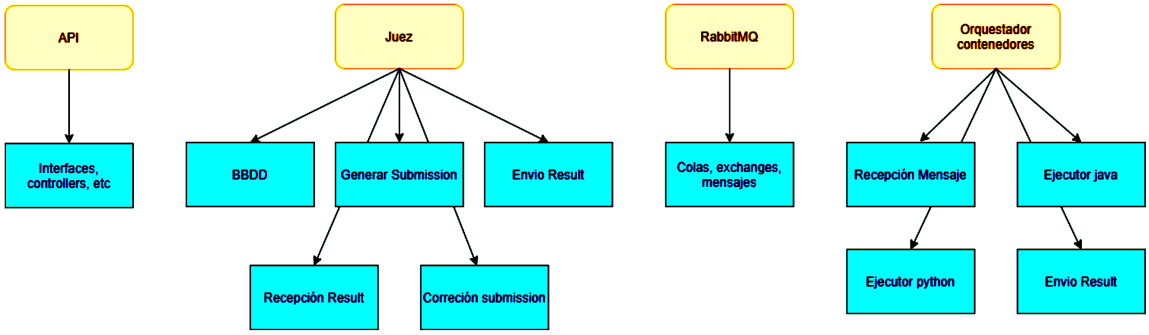

Figura 4.1: Esquema de arquitectura de módulos y servicios de la aplicación de un Result

4.3. Funcionamiento

El juez permite crear concursos, problemas, usuarios o participantes y equipos. Permite realizar entregas, consultar el marcador a todo momento y de manera indirecta gracias a los Websockets implementados. El juez evalúa la entrega y genera una salida que indica si el código subido (entrega) es correcto, por lo que será aceptable o no. Es capaz de evaluar entregas realizadas en los lenguajes de programación Java, C, C++, Python y SQL (habrá una mejora en el futuro).

Para crear un problema es necesario utilizar un archivo .zip, con los resultados que se esperan y los distintos casos de prueba.

El fichero .zip debe contener:

1. Data
2. Sample
3. Secret

Cada entrada tiene una salida. Las entradas deben tener la extensión .in y las salidas deben tener la extensión .ans.

Evaluación de las entregas

Cuando se realizan las entregas, el *frontend* se comunica con el *backend* mediante los métodos HTTP/IP y la API Rest.

En el *backend* se envían los ficheros para que los ejecute el orquestador. Para transferirlos se utilizan los servicios de mensajería de Rabbit (RabbitMQ).

Las entregas se ejecutan en el *backend* creando contenedores. Cuando se termina la evaluación, automáticamente se actualizan todas las bases de datos y se consultan los resultados de la ejecución en el *frontend* a través del marcador que de una manera automática y constante se está actualizando.

4.4. Estado inicial de la aplicación

Nos encontramos ante una aplicación bastante desarrollada en la cual vamos a trabajar implementando nuevas funcionalidades. En esta aplicación la API ya estaba creada y la parte *frontend* ya había sido desarrollada con plantillas HTML, un código de estilo y una interfaz gráfica. Todo ello fue elaborado por el anterior alumno que participo en el desarrollo del juez de cara también a realizar su TFG.

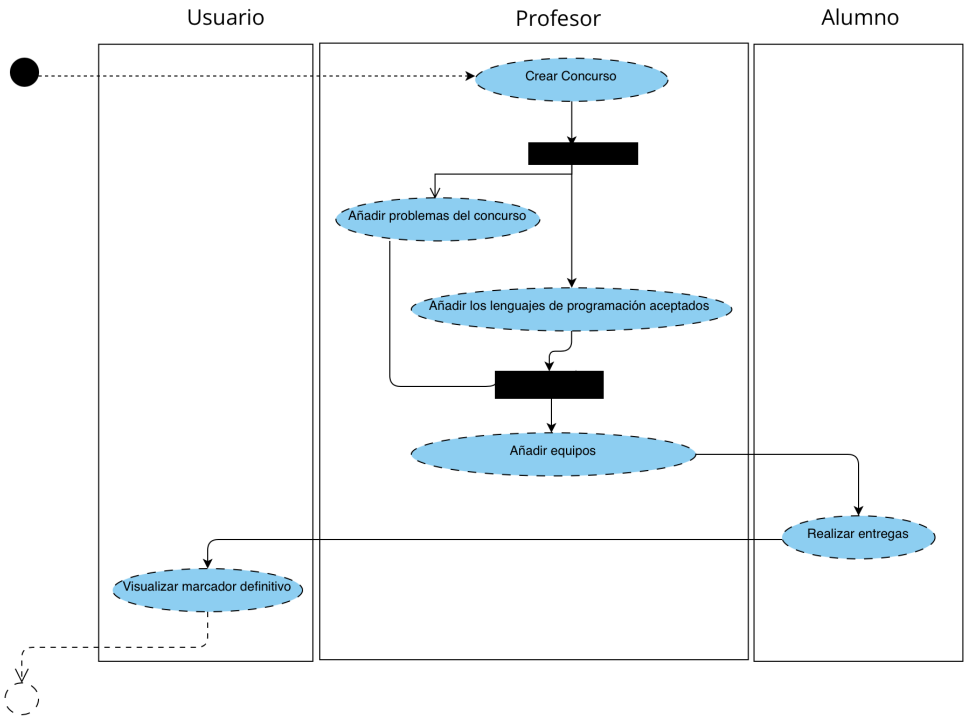

Figura 4.2: Estado inicial de la aplicación

4.5. Novedades

En este apartado vamos a comentar las novedades, junto con nuevas implementaciones y mejoras realizadas:

- Aumento de test. Tanto test unitarios como de integración.
- Implementación de Websockets en el programa para una carga automática de los resultados sin necesitar de ir manualmente a recargar el navegador.
- Corrección de las relaciones entre entidades del programa. cambiando estas relaciones establecidas de manera errónea para que este sin dependencias unas de otras.
- Mejora de la parte de Docker. Al hacer esto hemos mejorado el tiempo de descarga y la cantidad de archivos basura generados.
- Migración de Sprinfox a Springdocs de cara a una mejora en el futuro
- Corrección de logs y errores a la hora de implementar ciertas nuevas funcionalidades que han ido apareciendo.
- Mejora de la integración con SonarCloud.

4.6. Tecnologías y herramientas de desarrollo empleadas

Existen diferentes herramientas y tecnologías efectivas, útiles y eficaces que ayudan y facilitan el trabajo de los desarrolladores. A continuación, vamos a describir algunas de estas herramientas entre las que la mayoría han sido utilizadas en este trabajo de fin de grado. Estas son:

- Spring

  Spring Framework de código abierto o software libre que proporciona un modelo integral de programación y configuración para aplicaciones empresariales modernas basadas en Java, en cualquier tipo de plataforma de implementación.

  Todo esto trae consigo una gran ventaja, ya que permite que los equipos de desarrollo puedan enfocarse directamente en la lógica empresarial que requiere la aplicación, haciendo el proceso más corto, rápido y eficaz, ahorrando líneas de código, evitando tareas repetitivas.

  La aplicación ha sido desarrollada con Spring.

- Inyección de Dependencias

  La inyección de dependencias es un patrón de diseño en el que un objeto o función recibe otros objetos o funciones de los que depende. Una forma de inversión de control, la inyección de dependencia tiene como objetivo separar las preocupaciones de construir objetos y usarlos, lo que lleva a programas débilmente acoplados.

- Programación orientada a objetos (*Object Oriented Programming (OOP)*)

  La programación orientada a objetos es un tipo de programación basado en el concepto de un "objeto", que pueden contener datos y código; datos en forma de campos y código en forma de procedimientos. Una característica común de los objetos es que los procedimientos están adjuntos a ellos y pueden acceder y modificar los campos de datos del objeto.

- Spring Boot

  Spring Boot es un micro framework de código abierto mantenido por una empresa llamada Pivotal. Proporciona a los desarrolladores de Java una plataforma para comenzar con una aplicación Spring.

- JPA o API de Persistencia de Java

  JPA es una especificación que indica cómo se debe realizar la persistencia (almacenamiento) de los objetos en programas Java.

4.7. Docker

Docker es un servicio que utiliza la virtualización a nivel del sistema operativo para entregar software en paquetes denominados contenedores [33].

Docker es un sistema que usa contenedores. De manera similar a cómo una máquina virtual. Virtualiza el hardware del servidor, los contenedores virtualizan el sistema operativo de un servidor.

4.7.1. Dockerfile

Se trata de un archivo o documento de texto simple que incluye una serie de instrucciones o comandos necesarios para la construcción de una nueva imagen.

A este conjunto de instrucciones se le conoce como línea de comandos y serán los encargados de indicar los pasos a seguir para el ensamblaje de una imagen en Docker, es decir, los elementos necesarios para el desarrollo de un contenedor en Docker.

Opciones de DockerFile

Dentro de las opciones más relevantes para un Dockerfile se encuentran herramientas encargadas de labores, como el establecimiento de la imagen base, el cambio de usuario, etc.

Algunas de estas opciones son:

1. **FROM**:

   Es una opción de Dockerfile que debe presentarse como la primera instrucción. Cumple con la función de establecer la imagen sobre la que los pasos e imágenes siguientes se desarrollan en el sistema.

2. **ENV**:

   Hace referencia a la opción que indica las variables de entorno que se necesitan en el proceso de construcción de una imagen en Docker y permite la ejecución de los contenedores y sus labores en el sistema.

3. **USER**:

   Esta herramienta se utiliza en los archivos de instrucciones de Dockerfile con el objetivo de cambiar el usuario y su pertenencia a un grupo determinado. Una vez se ejecute esta opción, se aplicará a la totalidad de instrucciones siguientes.

4. **RUN**:

   Es una de las opciones de mayor importancia y popularidad en Dockerfile. Cumple la labor de ejecutar una instrucción incluida en la línea de comandos de la imagen durante su proceso de construcción. Dockerfile RUN puede escribirse en formato SHELL o bajo la opción de escritura EXEC.

5. **ADD**:

   Este elemento se encarga de las tareas relacionadas con la copia de ficheros, directorios y archivos de una imagen en Dockerfile. Se debe tener en cuenta que el uso de la instrucción ADD implica la creación de una nueva capa de imagen, por lo que debes ser cuidadoso al implementar esta opción.

6. **EXPOSE**:

   Es la opción que tiene como labor la definición de las asignaciones referentes a los puertos para los contenedores de la plataforma que se encuentren en su etapa de ejecución.

4.7.2. Docker compose

Docker Compose es una herramienta que se desarrolló para ayudar a definir y compartir aplicaciones de varios contenedores. Con Docker Compose, podemos crear un archivo YAML para definir los servicios y con un solo comando, podemos girar todo o romperlo todo..

Contenedores

Un contenedor Docker es un entorno de tiempo de ejecución virtualizado que se utiliza en el desarrollo de aplicaciones. Se utiliza para crear, ejecutar e implementar aplicaciones que están aisladas del hardware subyacente. Un contenedor Docker puede usar una máquina, compartir su kernel y virtualizar el sistema operativo para ejecutar procesos más aislados. [33].

4.7.3. Github / Git

GitHub es una empresa con fines de lucro que ofrece un servicio de alojamiento de repositorio Git basado en la nube. Esencialmente, hace que sea mucho más fácil para las personas y los equipos usar Git para el control de versiones y la colaboración.

La interfaz de GitHub es lo suficientemente fácil de usar para que incluso los programadores novatos puedan aprovechar Git. Sin GitHub, el uso de Git generalmente requiere un poco más de conocimientos técnicos y el uso de la línea de comandos.

Sin embargo, GitHub es tan fácil de usar que algunas personas incluso usan GitHub para administrar otros tipos de proyectos.

4.7.4. RabbitMQ

RabbitMQ es un software de cola de mensajes, también conocido como intermediario de mensajes o administrador de colas. Dicho simplemente; es un software donde se definen las colas a las que se conectan las aplicaciones para transferir un mensaje o mensajes. Un mensaje puede incluir cualquier tipo de información.

Sus características principales son:

- **Garantía de entrega**.
- **Enrutamiento flexible**.
- **Clusterización**.
- **Federación**.
- **Alta disponibilidad**.
- **Tolerancia a fallos**.

4.7.5. SonarCloud

SonarCloud es una plataforma de análisis de código online y continuo con la que puedes analizar tus proyectos, obtener y poder ver los resultados online en la nube [39].

Con SonarCloud se puede obtener métricas sobre el estado actual del código, detectar donde hay fallos, errores, bugs, vulnerabilidades, etc.

SonarCloud no solo muestra donde están los fallos, sino que también proporciona información sobre el fallo detectado.

4.7.6. Swagger

Swagger es una herramienta de software de código abierto para diseñar, construir, documentar, y utilizar servicios web RESTfull. Swagger se utiliza para documentar las API [41].

4.7.7. Postman

Es un software de colaboración para desarrollar APIs que permite crear APIs de forma sencilla y rápida.

4.7.8. Bases de datos o BBDD

Se ha usado la base de datos de MySQL. Es la herramienta que nos permite guardar todos los datos que hay en el sistema.

4.7.9. Bibliotecas de JAVA

Las bibliotecas de java más utilizadas con Spring son:

JUnit:

Permite realizar pruebas de código de forma controlada, ya sean generales o parciales.

http://junit.org/junit4/

Mockito:

Mockito es una librería Java para la creación de Mock Objects ampliamente usados en pruebas unitarias de Test Driven Development y basado en EasyMock [27].

https://github.com/mockito/mockito

Las Commons de Apache Software Foundation:

commons-io
Es parte de un conjunto de proyectos desarrollado por Apache Software Foundation. Están divididos en varias categorías:

- **Utility classes**

  Con métodos estáticos que realizan tareas comunes.

- **Input**

  Implementaciones bastante útiles de Stream y Reader.

- **Output**

  Implementaciones de Output Stream y Writer.

- **Filters**

  Diversas implementaciones de filtros de archivos.

- **Comparators**

  Diversas implementaciones de java.util.Comparator para archivos.

- **File Monitor**

  Componente para monitorizar eventos de archivos de sistema.

Spring-context

Spring-contex también se les llama como contenedores Spring IoC, son aquellos responsables de crear instancias, configurar y ensamblar beans mediante la lectura de metadatos de configuración de XML, anotaciones Java y/o código Java en los archivos de configuración.

https://projects.spring.io/spring-framework/

Httpclient

HttpClient busca llenar el vacío del paquete java.net aumentando la funcionalidad y flexibilidad al acceder a recursos via HTTP, de forma eficiente, actualizada y rica en características para implementarse del lado cliente con los estándares y recomendaciones de HTTP más recientes.

https://hc.apache.org/httpcomponents-client-ga/

Testing

Es un framework de pruebas inspirado en JUnit y NUnit, pero introduce algunas nuevas funcionalidades que lo hacen más potente y fácil de usar.

http://testng.org/doc/index.html
Docker-java

Para Docker y contenedores.
Spring-boot-starter-amqp

Para el servicio de paso de mensajes por cola RabbitMQ.
Mysql-connector

Para las bases de datos MySQL.
Snakeyaml

Se utiliza para serializar texto yaml o yml en objetos java y deserializar objetos java en textos yaml o yml.
Jackson

Para la conversión automática de objetos java a JSON y viceversa. Para Jackson se utilizan 3 bibliotecas:

- jackson-core.
- jackson-annotations.

- jackson-databind.

Spring-boot-starter
Para crear aplicaciones básicas con Spring.

Spring-boot-starter-web
Para crear aplicaciones web con Spring.

Spring-boot-devtools
Para utilizar herramientas de desarrollo internas de Spring.

Spring-boot-starter-test
Para realizar pruebas con Spring.

Spring-boot-starter-jpa
Para gestionar las bases de datos con Spring y JPA.

4.7.10. Otras herramientas también importantes

- Overleaf.
- Teams.
- Outlook.
- Telegram.

4.8. Tareas llevadas a cabo

De cara a conseguir el objetivo final podríamos decir que este se ha ido dividiendo en diferentes tareas y diferentes subtareas. Comentadas anteriormente el apartado 4.1

4.8.1. Explicación de las tareas

.

Dont Download dependencies on container rebuild

Durante esta tarea únicamente he modificado el Dockerfile cambiando el orden de ciertas acciones que hacía y copiándolo en el directorio /app para que cada vez que se ejecute el comando de build y se construya el contenedor no se tenga

que volver a descargarse todas las dependencias de nuevo como se hacía antes.

Al realizar esto y que la aplicación las coja de la caché, así es mucho más productivo y más eficiente el contenedor y el Dockerfile.

En esta *issue* aparte de modificar el Dockerfile para que no tuviera que reinstalarse todo, también tuve que modificar el pom.xml en el cual tuve que configurar un plugin, en concreto este:

```
<plugin>
    <groupId> org.apache.maven.plugins </groupId>
    <artifactId> maven-surefire-plugin</artifactId>
    <version> 2.12.4 </version>
    <configuration>
        <skipTests> false </skipTests>
    </configuration>
</plugin>
```

Este plugin lo que hace es que pasa por alto los test internos de Maven en los cuales no se podría ejecutar porque no hay ningún pom.xml para la instalación del mismo.

También me daba error porque tuve que ampliar la memoria del contenedor porque llegaba ya a 1 MiB (miliByte) y fallaba por la falta de recursos. Conseguí solucionarlo aumentando la capacidad del contenedor y levantándolo ahí.

Migrate Springfox to Springdocs

En esta *issue* vamos a migrar cierto código para tener el repositorio actualizado.

Para realizarlo, elimino las dependencias que se encuentran en el pom.xml del proyecto y sustituyo múltiples anotaciones que cambian de una versión a otra.

En un primer momento, pensé que lo había hecho correctamente y es que ahora para Springdocs los *paths* para que swagger-ui funcione debe de ser acabado en .html.

Solo y cuando acabe en .html es cuando te muestra y el enlace funciona.

Por lo que para no tener en cuneta esta regla nueva impuesta. si empezamos introduciendo y usando este comando es como conseguimos que no lo tenga en cuneta y funcione:

springdoc.api-docs.path=/api-docs

springdoc.packagesToScan = es.urjc.etsii.grafo.iudex.api.v1

springdoc.swagger-ui.path=/swagger-ui.html

Finalmente, he corregido el error del .html para que se pudiera acceder a la interfaz de swagger-ui sin necesidad de poner.html ha sido tan sencillo como modificar la anterior línea en negrita quitando el .html dejando el path solo como /swagger-ui/.

Eliminar ManyToManys

Las relaciones están y tienen mal configurado los atributos OneToMany, ya que no corresponden a los tipos creados como ContesTeams, ProblemTeams etc. Por lo que salta este problema el cual debemos de arreglar.

Se arregló añadiendo y creando un nuevo objeto del tipo de datos correspondiente, añadiéndole como parámetros el contests, el team, el problem, el language, etc. correspondiente y este mismo introduciéndolo en el atributo adecuado para que así las relaciones concordasen en cuanto a tipos corresponde.

En primer lugar, se crearon más clases que compartieras atributos como ContestTeams, ContestProblems, etc. Una vez completado, eliminamos las relaciones ManyToManys y las cambiamos poniéndolas como OneToManys y ManysToOnes.

API Websockets

Esta tarea ha resultado un desafío, ya que era la primera vez que tocaba este tipo de tecnologías. En un primer momento, no me ubicaba, pero ya buscando documentación e información sobre como implementar los Websockets, el puerto que usan, etc.

Al final fui capaz de llevarlo a cabo y ponerlo en práctica junto con un test llamado test.html para comprobar que indirectamente se iba recargando la página a sin necesidad del usuario tener que refrescar.

Esta parte me cree los métodos OnMessage, OnChat, etc. los cuales son clave para activar y actualizar los Websockets [16].

4.9. Mejoras llevadas a cabo

Al realizar todas las tareas anteriores comentadas hemos ido dándonos cuenta de ciertas mejoras a implementar y corrección de ciertos errores que salían y que indirectamente se podían solventar realizado con una mejora en el código y una mejorar en ciertas funcionalidades.

La siguiente mejora vino dada por un problema procedente de los repositorios. A la hora de realizar ciertas funciones como el buscar un concurso o el buscar un problema.

Esto se estaba haciendo mediante bucles for, lo cual era una opción bastante ruda y ineficiente. Para corregir esto y hacer el código más eficiente y mejor. Pensamos en realizar y llevar a cabo una mejora.

Por lo que he decidido crear una serie de nuevos repositorios que extienden de JPARepository los cuales inyectas en los servicios para así poder realizar esas funciones sin necesidad de bucles for haciendo así el código más legible, más eficiente y limpio.

Por poner un ejemplo: la creación de un repositorio llamado ContestProblemRepository en el cual puedes hacer un findByContest o findByProblem eliminando así los bucles for.

Otra de las mejores importantes es que mientras trabajábamos en la parte de modificar, salvar o añadir los concursos, problemas, lenguajes, etc.

Nos dimos cuenta de que en las diferentes entidades con las funciones de los repositorios observamos que directamente podíamos añadir la anotación de **@Transactional** para así poder eliminar los *saves*.

Nos dimos cuenta de que al añadir esta anotación ya no debería de depender los métodos de ello, es decir, aplicando el @Transactional hacemos que se ejecute el método al completo y si falla algo se descarte lo anterior y no se guarde en la BBDD.

Por el contrario, si se ejecuta y todo es correcto, se guarda automáticamente sin necesidad de los *saves*.

También cabe destacar que realizando esto me di cuenta de que únicamente se realiza solo cuando la entidad es nueva debido a que no lleva un registro pre-

vio de la misma. Una vez hecho, si no se modifica, no hace falta volver a guardarlo.

Dentro de esta parte con los @Transactional al añadir varios usuarios a un concurso no se utilizaba. A veces, si alguna de las operaciones falla, puede ser necesario deshacer las operaciones que no han fallado, como por ejemplo los posibles problemas que podrían dar si no se podía añadir a los equipos indicados.

Transactional es una opción que nos facilitan los SGBD para realizar este tipo de operaciones y así hicimos para corregirlo y mejorar esta implementación.

También se han llevado a cabo tareas de cara a la mejora de la aplicación que podríamos describir como reformatear ficheros y eliminar comentarios que sobran.

También aquí podríamos introducir el eliminar ciertas partes de las entidades que sobraban como el grade y el de la introducción de equals y hashCode de una manera correcta, ya que se había introducido con todas las partes de las entidades y no era correcto.

5 Conclusiones

En este capítulo, definiremos y desarrollaremos las conclusiones y trabajos futuros generados gracias a este Trabajo de Fin de Grado.

5.1. Conclusiones

Este proyecto nació con la necesidad de implementar nuevas herramientas en la aplicación de cara a hacerla más eficiente y segura. Para cumplir este objetivo se han llevado a cabo varias funciones. Estas funciones han sido la de desarrollo web y mantenimiento del mismo.

También se han llevado a cabo implementaciones de nuevas tecnológicas unificando otros conocimientos como conocimientos de programación en Java y en estructuras de datos. Como parte final se han llevado a cabo la mejora a nivel de test llevando a cabo test unitarios y de integración dentro de la propia aplicación.

Durante todo el proceso de desarrollo del trabajo de fin de grado se han empleado herramientas y tecnologías actuales, seguras e intuitivas. Se ha obtenido un sistema fiable y robusto, el cual ha cumplido los objetivos propuestos. Objetivos planteados en el capítulo 3.

Durante la realización de este proyecto he obtenido múltiples nuevos conoci-

mientos y mejorado otros.

Finalmente, este proyecto se ha desarrollado de tal manera que cualquier estudiante u organización que quiera adaptarlo a su entorno puedan hacerlo sin ningún problema e inconveniente. Este proyecto seguirá teniendo continuidad dentro de la Universidad Rey Juan Carlos, ya que faltan mejoras y tiene mucho futuro por delante.

5.2. Trabajos futuros

El juez es una aplicación web de código libre que se puede emplear en distintos campos y puede ser utilizado por distintos tipos de usuarios.

La aplicación puede ser mejorada de varias maneras, algunas de las cuales pueden ser estas:

- Mejora del juez. Este TFG es un proyecto para revisar y mejorar ciertas estructuras que han sido utilizadas en este, además de implementar más funcionalidades en la parte del orquestador, como puede ser añadir nuevos lenguajes.
- Mejora de las formas de puntuar las entregas. Actualmente se puntúa como se hace normalmente en los concursos de programación, el ganador (el equipo con más puntuación) es el que resuelve la mayor cantidad de problemas más rápidamente y con menos intentos fallidos.
- Mejora en la métrica de las soluciones para que los usuarios puedan observa como de bueno es su algoritmo en los problemas donde la eficiencia, el rendimiento, el tiempo de ejecución o alguna otra característica se considere más importante.
- Mejora de la gestión de roles y de la autenticación. Actualmente, el usuario es un parámetro, debería ser el usuario actual que ha iniciado sesión, pero no hay sistema de login-sesiones. Añadir sistema de gestión de sesiones basado en LTI.
- Mejora en la estructura de la aplicación dividiendo el *backend* en dos componentes.
    - El equivalente al Domserver:
      el "coordinador", se encarga de gestionar todo y dividir el trabajo. Es el único que tiene acceso a la base de datos.
    - El equivalente al Judgehost:
      Coge códigos de la cola, los evalúa, y envía el resultado de vuelta.

- Integración de la aplicación dentro del aula virtual. Mediante la utilización del proyecto Spring-LTI [40] de Raúl Martín Santamaría, podemos incluir la gestión de usuarios al aula virtual, centrándonos en la funcionalidad del juez.

Bibliografía

[1] Pekka Abrahamsson, Outi Salo, Jussi Ronkainen, and Juhani Warsta. Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439*, 2017.

[2] Alessandro Alessandrini. Red informática mundial: Los primeros objetivos de las redes son las aplicaciones científicas. *Mundo científico*, (238):52–57, 2002.

[3] Miguel Sierra Alonso. Desarrollo de una interfaz web para juez de codigo. Master's thesis, URJC, 2022.

[4] LA Bassalygo, M Burmester, A Dyachkov, and G Kabatianskii. Hash codes. In *IEEE INTERNATIONAL SYMPOSIUM ON INFORMATION THEORY*, pages 174–174. INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 1997.

[5] Sufyan bin Uzayr. Introduction to visual studio code. In *Optimizing Visual Studio Code for Python Development*, pages 1–46. Springer, 2021.

[6] Casper Solheim Bojer and Jens Peder Meldgaard. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 37(2):587–603, 2021.

[7] Gerardo Canfora and Aniello Cimitile. Software maintenance. In *Handbook of Software Engineering and Knowledge Engineering: Volume I: Fundamentals*, pages 91–120. World Scientific, 2001.

[8] Marta Caro Martínez. Sistemas de recomendación basados en técnicas de predicción de enlaces para jueces en línea. 2017.

[9] Ned Chapin, Joanne E Hale, Khaled Md Khan, Juan F Ramil, and Wui-Gee Tan. Types of software evolution and software maintenance. *Journal of software maintenance and evolution: Research and Practice*, 13(1):3–30, 2001.

[10] Lluís Codina. Hojas de estilo. *El profesional de la información*, 11(4):282–285, 2002.

[11] Andrés Felipe Pineda Corcho and Julián Moreno Cadavid. Trazabilidad de actividades de aprendizaje en cursos de programación de computadores usando un juez automático en línea dentro de un LMS (traceability of learning activities in computer programming courses using an automatic online judge within an LMS). In Eliana Scheihing, Julio Guerra, Valeria Henríquez, Cristian Olivares, and Pedro J. Muñoz-Merino, editors, *Proceedings of the 2nd Latin American Conference on Learning Analytics, Valdivia, Chile, March 18-19, 2019*, volume 2425 of *CEUR Workshop Proceedings*, pages 122–131. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2425/paper35.pdf.

[12] Gobierno de Córdoba. Ministerio de educación. *Secretaría de Educación. Subsecretaría de Promoción de Igualdad y Calidad Educativa (2011b). Diseño Curricular del Ciclo Básico de la Educación Secundaria*, 2015, 2011.

[13] Emma Enström, Gunnar Kreitz, Fredrik Niemelä, Pehr Söderman, and Viggo Kann. Five years with kattis—using an automated assessment system in teaching. In *2011 Frontiers in education conference (FIE)*, pages T3J–1. IEEE, 2011.

[14] Jennifer Erazo Martínez, Andrés Florez Gómez, and Francisco J Pino. Generando productos software mantenibles desde el proceso de desarrollo: El modelo de referencia mantus. *Ingeniare. Revista chilena de ingeniería*, 24(3):420–434, 2016.

[15] Ali Fadel, Husam Musleh, Ibraheem Tuffaha, Mahmoud Al-Ayyoub, Yaser Jararweh, Elhadj Benkhelifa, and Paolo Rosso. Overview of the pan@ fire 2020 task on the authorship identification of source code. In *Forum for Information Retrieval Evaluation*, pages 4–8, 2020.

[16] Ian Fette and Alexey Melnikov. The websocket protocol. Technical report, 2011.

[17] Michal Forišek. Pushing the boundary of programming contests. *Olympiads in Informatics*, 7, 2013.

[18] Guzmán Garrido Alique, César Garza Sánchez, and Jesús Alejandro Sánchez Couto. Juez en línea para introducir programación con blockly. 2021.

[19] Ana Gómez Codutti, Sonia Itatí Mariño, and Pedro Luis Alfonzo. Una propuesta integradora de mantenimiento correctivo aplicada al diseño web adaptativo y accesibilidad web. 2016.

[20] Pedro Pablo Gómez-Martín and Marco Antonio Gómez-Martín. ¡ acepta el reto!: juez online para docencia en español. *Actas de las Jornadas sobre Enseñanza Universitaria de la Informática*, 2:289–296, 2017.

[21] Jeisson Hidalgo-Céspedes, Gabriela Marín-Raventós, and Marta Eunice Calderón Campos. Online judge support for programming teaching. In *XLVI Latin American Computing Conference, CLEI 2020, Loja, Ecuador, October 19-23, 2020*, pages 522–530. IEEE, 2020. doi: 10.1109/CLEI52000.2020.00067.

[22] Richard Hull, Peter Thiemann, and Philip Wadler, editors. *Programming Paradigms for the Web: Web Programming and Web Services, 28.01. - 02.02.2007*, volume 07051 of *Dagstuhl Seminar Proceedings*, 2007. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany. URL http://drops.dagstuhl.de/portals/07051/.

[23] Yuya Inagaki, Hajime Iwata, Junko Shirogane, and Yoshiaki Fukazawa. Visualization method of important regions by combination of webpage structures and saliency maps, 2020.

[24] Jesús Insuasti. Problemas de enseñanza y aprendizaje de los fundamentos de programación. *Revista educación y desarrollo social*, 10(2):234–246, 2016.

[25] Vladimir Costas Jáuregui, Marcelo Flores Soliz, and Jorge Orellana Araoz. Laboratorio virtual basado en experiencias de concursos de programación. *Suplemento Signos EAD*, 2016.

[26] Gema Jerez Gaitán. Sistemas web de postgrado: mantenimiento correctivo, adaptativo y perfectivo y diseño para dispositivos móviles. 2020.

[27] Myeongsoo Kim, Qi Xin, Saurabh Sinha, and Alessandro Orso. Automated test generation for REST apis: No time to rest yet. *CoRR*, abs/2204.08348, 2022. doi: 10.48550/arXiv.2204.08348.

[28] Karim R Lakhani, David A Garvin, and Eric Lonstein. Topcoder (a): Developing software through crowdsourcing. *Harvard Business School General Management Unit Case*, (610-032), 2010.

[29] Kyrylo Malakhov, Oleksandr Kurgaev, and Vitalii Velychko. Modern restful API dls and frameworks for restful web services API schema modeling, documenting, visualizing. *CoRR*, abs/1811.04659, 2018. URL http://arxiv.org/abs/1811.04659.

[30] Praveen Mallela. Classifying difficulty levels of programming questions on hackerrank. In *Advances in Decision Sciences, Image Processing, Security and Computer Vision: International Conference on Emerging Trends in Engineering (ICETE), Vol. 1*, volume 3, page 301. Springer, 2019.

[31] P. P. G. Martín and M. A. G. Martín. !acepta el reto!: juez online para docencia en español, 2017. Master's thesis, Universidade de Vigo, , 2018, pp. 45–52.

[32] Mike MIRZAYANOV, Oksana PAVLOVA, Pavel MAVRIN, Roman MELNIKOV, Andrew PLOTNIKOV, Vladimir PARFENOV, and Andrew STANKEVICH. Codeforces as an educational platform for learning programming in digitalization. *Olympiads in Informatics*, 14:133–142, 2020.

[33] Orestis Rafail Nerantzis, Apostolos Tselios, and Alexandros Karakasidis. MI-OPJ: A microservices-based online programming judge. In Yixin Chen, Heiko Ludwig, Yicheng Tu, Usama M. Fayyad, Xingquan Zhu, Xiaohua Hu, Suren Byna, Xiong Liu, Jianping Zhang, Shirui Pan, Vagelis Papalexakis, Jianwu Wang, Alfredo Cuzzocrea, and Carlos Ordonez, editors, *2021 IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, December 15-18, 2021*, pages 5969–5971. IEEE, 2021. doi: 10.1109/BigData52589.2021.9671978.

[34] Aaron BloomfieldUniversity of Virginia aaron@virginia.eduBorja Sotomayor University of Chicago borja@cs.uchicago.edu. A programming contest strategy guide. *K.3.m [Computers and Education]: Miscellaneous*, 2021.

[35] Nelson Orozco Alzate. Conceptos básicos sobre mantenimiento preventivo y mantenimiento correctivo. *Facultad de Minas*, 2013.

[36] Alexandra Pérez Bermejo. Dashboard para jueces en línea. 2020.

[37] Minh Tuan Pham and Tan Bao Nguyen. The domjudge based online judge system with plagiarism detection. In *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–6. IEEE, 2019.

[38] DF Primero, JC Díaz, LF García, and A González-Vargas. Manual para la gestión del mantenimiento correctivo de equipos biomédicos en la fundación valle del lili. *Revista Ingeniería Biomédica*, 9(18):81–87, 2015.

[39] Daniel Rodrigo Quiñones Pinilla and Diego Sebastián Díez. Devops en el desarrollo de una api rest con. net core y arquitectura de microservicios. 2019.

[40] Raul Martín Santamaría. Spring lti. 2022. URL https://github.com/rmartinsanta/spring-lti.

[41] Jéssica Santos, Leonardo Azevedo, Elton Soares, Raphael Thiago, and Viviane Silva. Analysis of tools for rest contract specification in swagger/openapi, 2020.

[42] Arno Scharl. *Evolutionary web development*. Springer Science & Business Media, 2012.

[43] S. Skiena and M. Revilla. Programming challenges: The programming contest training manual. *Texts in Computer Science*, Springer, 2003.

[44] Oleg Smirnov, Ameya Ketkar, Timofey Bryksin, Nikolaos Tsantalis, and Danny Dig. Intellitc: Automating type changes in intellij idea. In *2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pages 115–119. IEEE, 2022.

[45] Manuel Trigás Gallego. Metodologia scrum. 2012.

[46] Daniel Vargas Azpitarte. Gestión de logs en un entorno de seguridad informática. 2019.

[47] Ayodhya Wathuge and Darshana Sedera. Saving the environment from the internet: A polynomial mitigation model of reducing individual internet consumption through internet pricing and environmental awareness. *Australas. J. Inf. Syst.*, 26, 2022. doi: 10.3127/ajis.v26i0.3239.