Universidade Vigo

ESCOLA **S**UPERIOR DE **E**NXEÑARÍA **I**NFORMÁTICA

Memoria do Traballo de Fin de Grao que presenta

D. Collazo Sorribas Sergio
para a obtención do Título de Graduado en Enxeñaría Informática

Xerador aleatorio de exercicios de matemáticas para Bacharelato

Setembro, 2020

Traballo de Fin de Grao Nº: EI 19/20-105

Titor/a: Tomás R. Cotos Yáñez
Área de coñecemento: Estatística e Investigación Operativa
Departamento: Estatística e Investigación Operativa

Contenido

1. Introducción 3
2. Objetivos 4
   2.1. Objetivo principal 4
   2.2. Objetivos específicos 4
3. Resumen de la solución propuesta 4
   3.1 Metodología Ágil 4
      3.1.1. Scrum 5
      3.1.2. Kanban 7
   3.2. Metodología aplicada 7
4. Planificación y seguimiento 9
   4.1 Planificación general 9
   4.2 Planificación de Sprints 10
5. Arquitectura 16
   5.1. JavaEE 16
      5.1.1. Servlet 16
   5.2. Maven 17
6. Tecnologías e integración de productos de terceros 18
   6.1. Interfaz de usuario 18
      6.1.1. HTML 18
      6.1.2. Bootstrap 18
      6.1.3. JavaScript 18
      6.1.4. MathJax 19
   6.2. Servidor 19
      6.2.1. Java 19
      6.2.2. GlassFish 20
   6.3. Entorno de desarrollo 20
      6.3.1. NetBeans 20
   6.4. Aplicaciones externas 20
      6.4.1. Maxima 20
      6.4.2. MiKTeX 21
   6.5. Function Plot 21
      6.5.1. D3.js 21
7. Especificación y análisis de requisitos 22
   7.1. Primera recopilación de requisitos 22
      7.1.1. Requisitos funcionales 22
      7.1.2. Requisitos no funcionales 26
   7.2. Segunda recopilación de requisitos 28
      7.2.1. Requisitos funcionales 28
8. Diseño del Software 31
   8.1. Java-Maxima 31

8.2. Generación de documentos descargables 32
9. Gestión de datos e Información 33
10. Pruebas llevadas a cabo 34
10.1. Pruebas de caja negra 34
10.2. Pruebas de caja blanca 35
10.3. Múltiples pruebas simultaneas 35
11. Manual de usuario 36
11.1. Instalación 36
11.1.1. Recursos necesarios 36
11.1.2. Instalar Maxima 5.43.2 36
11.1.3. Instalar MiKTeX 37
11.1.4. Apache NetBeans 11.2 38
11.2. Ejecución 44
11.3. Manual de uso 44
12. Principales aportaciones 47
13. Conclusiones 47
14. Vías de trabajo futuro 48
15. Índice de Ilustraciones 49
16. Índice de tablas 50
17. Referencias 51

1. Introducción

Con el progresivo crecimiento de Internet y las plataformas online, se puede ver cómo los ámbitos a los que se puede aplicar estas tecnologías son muy diversos y no dejan de aumentar en número.

Los entornos de educación y la adquisición de conocimientos y aprendizaje de los mismos no se quedan atrás. La diversidad de herramientas y plataformas que aparecen en la red en estos ámbitos, facilitan y fomentan un mayor aprendizaje autónomo, sirviendo tanto a estudiantes como educadores de todas las ramas. Respecto a estos últimos además de existir herramientas para seguir formándose como profesionales y actualizar sus conocimientos, también existen varias utilidades y herramientas para que puedan liberarse de una carga de trabajo en tareas repetitivas.

Actualmente existen diversas plataformas que permiten resolver distintos ejercicios del campo de las matemáticas. Es decir, funcionan cuando un alumno quiere, por ejemplo, comprobar la solución a un problema, sin embargo, no existen muchas herramientas que permitan generar de manera aleatoria múltiples ejercicios de distintos tipos.

En el conjunto de webs que si permiten esta generación aleatoria se encuentran aquellas que proporcionan la posibilidad de crear múltiples ejercicios aleatorios de manera simple, pero la diversidad de ejercicios posibles es escasa y de un nivel de dificultad baja, ya que solo pueden generar ejercicios de operaciones básicas como la suma, la resta, la multiplicación, o división. También se encuentran entre estas webs o aplicaciones aquellas que ofrecen ejercicios para un nivel educativo superior como bachillerato, el problema de estas aplicaciones es que solo pueden ofrecer una lista de ejercicios ya construidos en un documento tipo PDF, es decir, tienen cierta variedad, pero les falta la capacidad de generar estos ejercicios de manera aleatoria, por lo cual solo tienen un número limitado de ellos y se hace insuficiente a largo plazo.

Teniendo en cuenta la información que conforma el marco general de webs y aplicaciones enfocadas al aprendizaje autónomo y a facilitar el trabajo repetitivo a educadores, se puede ver que no existen aplicaciones o webs que permitan una generación múltiple de ejercicios variados a nivel de bachillerato, de manera aleatoria. Por tanto, este trabajo se centrará en el desarrollo de una aplicación web en la que se podrá generar este tipo de ejercicios, pudiendo elegir entre el tipo deseado, la dificultad del mismo y el número de ejercicios a generar de manera aleatoria. Además, es interesante dar opción a descargar estos ejercicios de manera automática en formato PDF, de manera que sean fácilmente imprimibles para su uso posterior en caso de así desearlo el usuario.

2. Objetivos

2.1. Objetivo principal

El objetivo principal de este trabajo es el de ofrecer una herramienta que dé la posibilidad de generar un numero de ejercicios aleatorios entre varios tipos a escoger a nivel de bachillerato. Para lo cual el trabajo se centrará en el desarrollo de una aplicación web.

2.2. Objetivos específicos

La aplicación web desarrollada deberá tener una interfaz sencilla e intuitiva de usar, de manera que no requiera ningún aprendizaje extra para ser utilizada.

La aplicación deberá tener variedad de ejercicios a nivel de bachillerato, como, por ejemplo, ecuaciones, limites, derivadas, integrales, logaritmos, matrices… Para la generación de los mismos se podrá escoger entre diversas dificultades y número de ejercicios. Los ejercicios que se generen deben ser aleatorios, pero el usuario puede tener la opción de modificarlos y editarlos a su antojo.

Incorporar gráficos a los ejercicios y su solución también es un objetivo a tener en cuenta ya que ofrece otro enfoque al usuario, de manera que tener graficas interactivas construidas dinámicamente para cada ejercicio es una aportación valiosa para la aplicación y el usuario.

La posibilidad de descargar los ejercicios en un documento una vez generados con las condiciones propuestas por el usuario, es de gran utilidad favoreciendo y facilitando su uso posterior por parte del mismo. Los formatos de archivo para los que se dará soporte serán PDF por su facilidad de uso para el usuario, tanto en lectura, como en la facilidad que da para ser impreso y el otro formato será TeX que da la posibilidad de integración con otras aplicaciones y es fácilmente editable.

3. Resumen de la solución propuesta

3.1 Metodología Ágil

La metodología utilizada ha sido una metodología de filosofía ágil (Manifesto for Agile Software Development, 2001), al ser un proyecto individual las metodologías ágiles como Scrum o Kanban que son utilizadas de manera general por equipos de desarrollo no encajaban del todo con las necesidades de este proyecto, es por esto que se han adaptado varias de sus propuestas y sistemas, siguiendo y manteniendo en todo lo posible los 12 principios del Manifiesto Ágil (Principles behind the Agile Manifesto, 2001). Siendo estos los siguientes:

> Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
>
> Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
>
> Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
>
> Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.

El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.

El software funcionando es la medida principal de progreso.

Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.

La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.

La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.

Las mejores arquitecturas, requisitos y diseños emergen de equipos autoorganizados.

A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

3.1.1. Scrum

Scrum (Kniberg & Skarin, 2009) es un modelo de desarrollo que sigue los principios ágiles, se originó en empresas de software cuyos proyectos tenían requisitos cambiantes que necesitaban ser integrados de manera rápida y flexible. Se caracteriza principalmente por solapar las diversas etapas de desarrollo en vez de seguir los modelos más clásicos de cascada o secuencial.

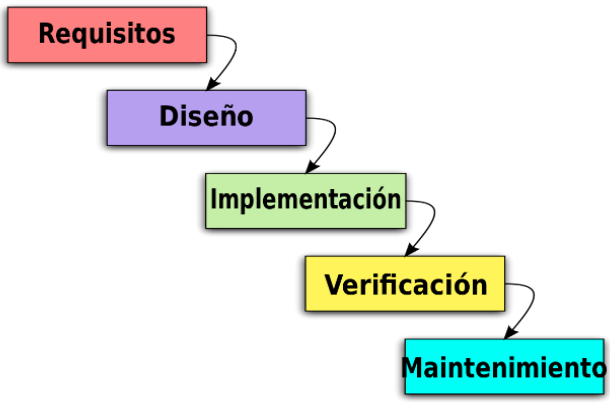


Ilustración 1: Modelo Cascada. Paulsmith99, Recuperado (2020): https://es.wikipedia.org/wiki/Desarrollo_en_cascada

También se caracteriza por tener una estrategia de desarrollo incremental en vez de dar prioridad a la planificación y ejecución completa del software. Finalmente, también se puede destacar su uso de roles dentro del desarrollo en los que cada miembro del equipo de desarrollo tiene un papel bien definido. Hay un total de 3 roles diferentes, el Product Owner, Equipo de desarrollo y el Scrum Master.

De esta metodología se utiliza su filosofía general y el concepto de Sprints. Un Sprint es el ciclo o iteración que se usa en esta metodología, puede haber uno o varios dentro del desarrollo, aunque lo más común es tener varios Sprints, estos suelen tener una duración corta, de entre varias semanas o como máximo dos

meses. Cada Sprint tiene como objetivo obtener software ya funcional con los requisitos o tareas designadas para cada Sprint, en caso de no cumplir todas las tareas establecidas las sobrantes se añadirán al siguiente Sprint y se estimará de nuevo su tiempo de desarrollo.

Los Sprints contienen y consisten de la Reunión de Planificación del Sprint (Sprint Planning Meeting), los Scrums Diarios (Daily Scrums), el trabajo de desarrollo, la Revisión del Sprint (Sprint Review), y la Retrospectiva del Sprint (Sprint Retrospective). (Schwaber & Sutherland, 2013/2013)

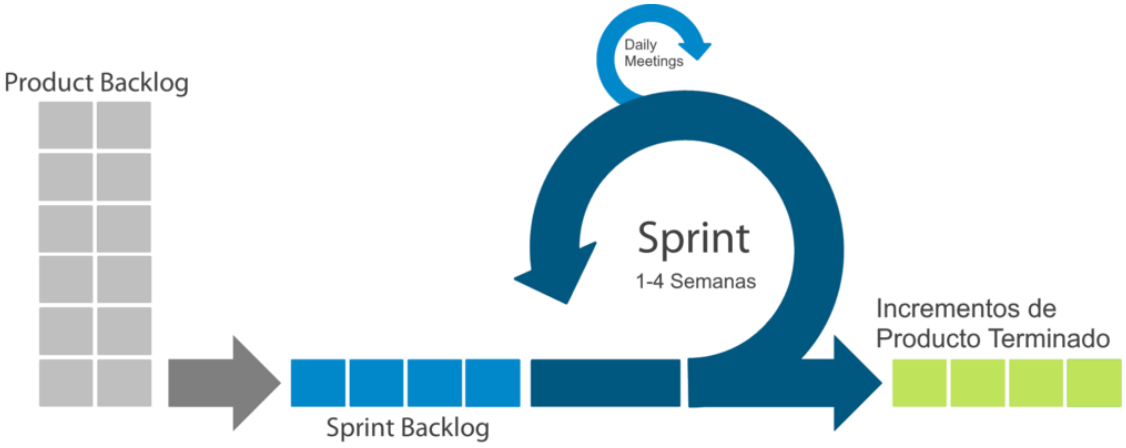


Ilustración 2: Metodología Scrum. Gracia, Richard. Recuperado (2020). https://richardgracia.com/scrum-para-startups/

En la reunión de planificación del Sprint se seleccionan los requisitos o tareas que se harán en cada sprint que habrá que elegirlos del Product Backlog en donde ya se encontrarán las tareas que previamente se hayan localizado, definido y estimado su duración.

El Product Backlog es un artefacto de Scrum que está formado por una lista ordenada en la que se encuentran todas las tareas y requisitos para que el producto software final esté completo. Estas tareas tienen que estar bien definidas, en su totalidad de características, requisitos, mejoras, correcciones y funcionalidades. Una vez definidas correctamente se procede a su estimación temporal. Es decir, estimar cuánto tiempo de trabajo se tardará en completar dichas tareas. Esta lista de tareas no es fija. Esto es, que puede alterarse añadiendo o eliminando tareas si los requisitos del cliente cambian y se debe estar abierto a este tipo de cambios, ya que es algo que forma parte de los principios ágiles.

Es importante definir ahora qué significa que una tarea esté completa, ya que puede variar dependiendo del equipo de desarrollo en el que nos encontremos, en el caso de este proyecto se hablará de tarea completa cuando esta haya sido definida, estimada, programada, implementada, probada y se haya puesto de manifiesto en un documento de tareas finalizadas, que se han cumplido los anteriores requisitos. Es decir, que si una tarea no ha sido probada o no se ha incluido en el documento a pesar de funcionar correctamente y estar implementada, no se considerará una tarea completa.

3.1.2. Kanban

Kanban es un método para definir, gestionar y mejorar servicios o productos intelectuales, como el diseño de software entre otros (Anderson & Carmichael, 2016). El método está basado en hacer visible lo que de otra manera es trabajo intelectual intangible, usando el sistema Kanban que es un sistema de flujo de trabajo que limita la carga de trabajo en progreso usando señales visuales. Estas señales se disponen en lo que se llama Kanban board, desde donde se puede controlar el flujo de trabajo que hay en cada momento, el trabajo se introduce en el sistema cuando se tiene la capacidad para ser desarrollado, no cuando es demandado.

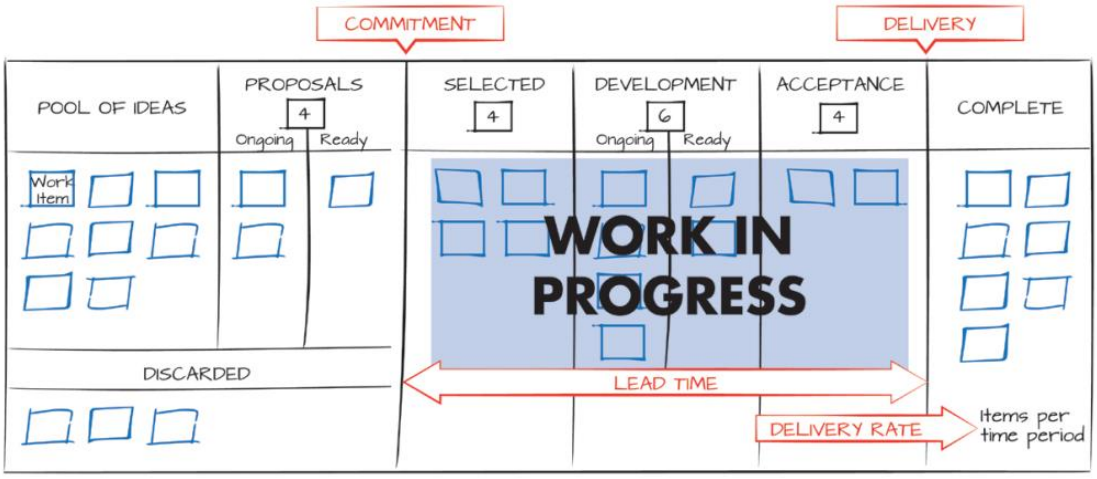


Ilustración 3:Kanban board. Anderson, David J., Carmichael, A., Essential Kanban Condensed, pag.13

Kanban se rige por 6 prácticas:

1. Visualizar.
2. Limitar la carga de trabajo.
3. Gestionar el flujo.
4. Hacer políticas explicitas.
5. Implementar bucles de retroalimentación.
6. Mejorar colaborativamente y experimentalmente.

3.2. Metodología aplicada

Para este proyecto se utiliza una metodología híbrida entre Scrum y Kanban, en la que se escogen los principios y prácticas que mejor funcionan para el desarrollo de la aplicación web de manera individual, por tanto, un gran número de las prácticas o filosofías que consisten en colaborar o trabajar en equipo no son necesarias para este desarrollo de software. Sin embargo, existen varias prácticas de ambas metodologías que sí se utilizan.

En primer lugar, se hace una toma de requisitos de la aplicación, en la que, tras una reunión o entrevista con el cliente, se recogen en un documento los requisitos y funcionalidades que tendrá la aplicación. Estos requisitos forman un acuerdo entre ambas partes y el desarrollador se compromete a cumplirlos, pero siguiendo la metodología ágil, estos pueden estar sujetos a cambios si se considera que otros requisitos o la modificación de los mismos puede proporcionar cierta ventaja al cliente, siempre ante petición del mismo.

El segundo paso de la metodología aplicada es crear tareas a partir de las funcionalidades a desarrollar. Una vez creadas las tareas, se deben describir y después estudiar su posible fragmentación en subtareas, con su

propia descripción y estimación. La forma de describir y explicar estas tareas será en forma de tarjetas. Que incluirán el nombre de la tarea, su descripción, sus restricciones y su estimación en horas de trabajo. El conjunto de estas tareas conformará el Product backlog.

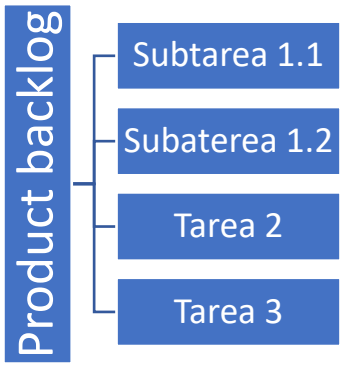


Ilustración 4: Esquema Product backlog

El tercer paso es empezar un Sprint que en este caso será de duración variable pero no superior a 1 mes. El Sprint comienza con la reunión de planificación del Sprint, en la que con las tareas recogidas en el Product backlog creado en el paso anterior se procede a escoger las tareas que formarán parte del primer Sprint, el objetivo es cubrir un 90% del tiempo del Sprint con el tiempo ya estimado de las tareas, las que no entren en este Sprint lo harán en los siguientes. Cada día de trabajo se realizará un Scrum diario que consiste en anotar qué se ha hecho el día anterior, qué problemas hubo, cómo se va a trabajar para solucionarlos y en qué tarea se trabajará ese día. El Scrum diario siempre es lo primero que se hace en el día de trabajo antes de empezar con el desarrollo del software, es útil para tener una perspectiva del trabajo realizado y centrarse en comó continuar con el mismo sin entrar directamente al código. Tras este Scrum diario se realiza el desarrollo. Antes de dar el Sprint por finalizado al acabar el tiempo marcado para el mismo, se procede a la revisión del Sprint que consiste principalmente en inspeccionar el Product backlog, revisar las tareas, y visualizar el avance del proyecto. Se realizan Sprints hasta terminar las tareas del Product backlog. Cada tarea terminada dentro del Sprint debe estar completa para ser aceptada.

Para la visualización del Product backlog, el avance del proyecto y revisar las tareas en producción se utilizará el Kanban board, ya que es de gran ayuda y se complementa a la perfección con las metodologías usadas de Scrum.

El cuarto paso es comprobar que todos los Sprints han terminado. Una vez comprobado se dará por finalizado el desarrollo.

Ilustración 5: Metodología aplicada al proyecto

4. Planificación y seguimiento

4.1 Planificación general

De manera general cada día de trabajo dentro del Sprint equivalen a 4 horas de trabajo real.
Un día de trabajo normal en un Sprint, con la metodología usada mencionada en el apartado *3.2. Metodología* aplicada de este documento es el siguiente:

Primero realizar el Scrum diario, que ha tenido una duración aproximada de unos 10 o 15 minutos cada día.

Segundo desarrollar o realizar pruebas de las funcionalidades y requisitos que estén en desarrollo. Tiene una duración aproximada de 3 horas y 30 minutos.

Por último, si en ese día de trabajo se ha completado alguna tarea, incluirla en el documento de tareas finalizadas y desplazar la tarjeta del Kanban board al apartado de tareas completas. Esta última fase tiene una duración aproximada de 10 o 15 minutos.

Nombre Fases Realizadas	Fecha Inicio de Fase	Fecha Fin de Fase	Duración Estimada	Duración Real
Toma de requisitos y funcionalidades	04/02/2020	04/02/2020	1 hora 30 minutos	1 hora
Redacción de requisitos y funcionalidades	05/02/2020	07/02/2020	12 horas	14 horas
Sprint 1	17/02/2020	9/03/2020	80 horas	71 horas
Sprint 2	23/03/2020	13/04/2020	80 horas	80 horas
Sprint 3[1]	14/04/2020	24/04/2020	44 horas*	30 horas*
Sprint 4	18/05/20220	29/05/2020	40 horas	40 horas
Reunión 1 Comprobación de Requisitos y trabajo realizado	16/07/2020	16/07/2020	1 hora 30 minutos	50 minutos
Sprint 5	20/07/2020	14/08/2020	80 horas	80 horas
Reunión 2 Comprobación de Requisitos y trabajo realizado	20/08/2020	20/08/2020	1 hora	1 hora
Elaboración Manual de Usuario	20/08/2020	20/08/2020	3 horas	1 hora 50 minutos

Tabla 1: Planificación general

Tiempo estimado final	Tiempo real final
299 horas	**319 horas y 40 minutos**

Tabla 2: Tiempos estimado y real finales

[1] El Sprint 3 no se ha tenido en cuenta a la hora de calcular la duración estimada del proyecto, pero si se ha tenido en cuenta al calcular la duración total ya que ha sido un Sprint no planificado inicialmente.

4.2 Planificación de Sprints

Sprint 1		
Tarea	**Duración estimada**	**Duración real**
T1.1 Estudio estructuras software	8 horas	8 horas
T1.2 Diseño estructura software	4 horas	4 horas
T1.3 Creación estructura software	8 horas	8 horas
T2.1 Estudio conector Java-Maxima	8 horas	10 horas
T2.2 Diseño conector Java-Maxima	8 horas	8 horas
T2.3 Creación conector Java-Maxima	24 horas	16 horas
T3 Creación StreamGobbler	4 horas	4 horas
T4 Creación Controlador de Configuración	8 horas	7 horas
T5 Creación Controlador de Ficheros	8 horas	6 horas
	Total	**Total**
	80 horas	71 horas

Tabla 3: Planificación Tareas Sprint 1

Recopilación Scrum Diario Sprint 1				
Fecha	**¿Qué se ha realizado el día anterior?**	**¿Qué problemas hubo?**	**¿Cómo se va a solucionar?**	**¿Qué se va a hacer este día?**
17/02/2020				**T1.1**
18/02/2020	T1.1			**T1.1**
19/02/2020	T1.1			**T1.2**
20/02/2020	T1.2			**T1.3**
21/02/2020	T1.3	Necesario cambiar parte de la estructura donde se encuentra el HTML principal y los scripts	Centralizando los scripts en una sola carpeta, junto con las librerías JS correspondientes	**T1.3**
24/02/2020	T1.3			**T2.1**
25/02/2020	T2.1			**T2.1**
26/02/2020	T2.1	No se ha encontrado una forma sencilla de implementar en java	Buscar formas alternativas que puedan adaptarse a la aplicación	**T2.1 y T2.2**
27/02/2020	T2.1 y T2.2			**T2.2**
28/02/2020	T2.2			**T2.2 y T2.3**
29/02/2020	T2.2 y T2.3	Problemas en T2.3 para ejecutar comandos de manera correcta desde java	Ejecutar el entorno en modo administrador, o cambiar permisos para las carpetas usadas	**T2.3**
02/03/2020	T2.3			**T2.3**
03/03/2020	T2.3			**T2.3**
04/03/2020	T2.3			**T2.3 y T3**
05/03/2020	T2.3 y T3			**T3 y T5**
06/03/2020	T3 y T5			**T5**
07/03/2020	T5			**T4**
09/03/2020	T4			**T4**

Tabla 4: Scrum diario Sprint1

Revisión del Sprint 1	
Tareas del Sprint	**Tareas Completadas**
T1.1, T1.2, T1.3, T2.1, T2.2, T2.3, T3, T4, T5	T1.1, T1.2, T1.3, T2.1, T2.2, T2.3, T3, T4, T5
	Porcentaje completo
	100%

Tabla 5: Revisión Sprint 1

Sprint 2		
Tarea	**Duración estimada**	**Duración real**
T6 Crear Servlet	4 horas	4 horas
T7.1 Diseñar Interfaz Usuario	4 horas	4 horas
T7.2 Integrar Interfaz Usuario	8 horas	8 horas
T8 Crear interfaz Operación	2 horas	2 horas
T9.1 Crear Operación Ecuación 2º Grado	8 horas	8 horas
T9.2 Crear Formulario Ecuación 2º Grado	10 horas	10 horas
T9.3 Crear Scripts Formulario Ecuación 2ºGrado	8 horas	8 horas
T9.4 Tratamiento del Formulario Ecuación 2ºGrado en Servlet	20 horas	16 horas
T9.5 Interpretación Salida Maxima para Ecuación 2º Grado	10 horas	20 horas
T9.6 Mostrar Resultados Ecuación 2º Grado en HTML	6 horas	-
	Total	**Total**
	80 horas	80 horas

Tabla 6: Planificación Tareas Sprint 2

Recopilación Scrum Diario Sprint 2				
Fecha	**¿Qué se ha realizado el día anterior?**	**¿Qué problemas hubo?**	**¿Cómo se va a solucionar?**	**¿Qué se va a hacer este día?**
23/03/2020				**T6**
24/03/2020	T6			**T7.1**
25/03/2020	T7.1			**T7.2**
26/03/2020	T7.2			**T7.2**
27/03/2020	T7.2			**T8 y T9.1**
28/03/2020	T8 y T9.1			**T9.1**
29/03/2020	T9.1			**T9.1 y T9.2**
30/03/2020	T9.1 y T9.2			**T9.2**
31/03/2020	T9.2			**T9.2**
01/04/2020	T9.2			**T9.3**
02/04/2020	T9.3	Script falla para comprobar todas las ecuaciones cuando se selecciona el botón correspondiente	Revisar como está interactuando el código JS con el HTML para esta función	**T9.3**
03/04/2020	T9.3			**T9.4**
06/04/2020	T9.4			**T9.4**
07/04/2020	T9.4	Conseguir que se trate a todas las ecuaciones que se generen dinámicamente y no solo a las estáticas	Generalizar la función “ProcesarFormulario-Ecuacion2Grado”	**T9.4**
08/04/2020	T9.4			**T9.4**
09/04/2020	T9.4			**T9.5**
10/04/2020	T9.5	Problemas para seleccionar exactamente las líneas deseadas para cada ecuación	Mejorar las expresiones regulares y condiciones ya creadas para este problema	**T9.5**
11/04/2020	T9.5	Cuando hay más de una ecuación la ejecución se detiene	Revisar bucles, o hacer debug línea a línea de la función “procesarSalidaMaxima”	**T9.5**
12/03/2020	T9.5			**T9.5**
13/04/2020	T9.5			**T9.5**

Tabla 7: Scrum diario Sprint 2

Revisión del Sprint 2	
Tareas del Sprint	**Tareas Completadas**
T6, T7.1, T7.2, T8, T9.1, T9.2, T9.3, T9.4, T9.5, T9.6	T6, T7.1, T7.2, T8, T9.1, T9.2, T9.3, T9.4, T9.5
	Porcentaje completo
	90%

Tabla 8: Revisión Sprint 2

Sprint 3		
Tarea	**Duración estimada**	**Duración real**
T9.6 Mostrar Resultados Ecuación 2º Grado en HTML	8 horas	8 horas
T10.1 Crear Operación Logaritmo	8 horas	2 horas
T10.2 Crear Formulario Logaritmo	8 horas	8 horas
T10.3 Crear Scripts Logaritmo	4 horas	4 horas
T10.4 Tratamiento del Formulario Logaritmo en Servlet	4 horas	4 horas
T10.5 Interpretación Salida Maxima para Logaritmo	8 horas	3 horas
T10.6 Mostrar Resultados Logaritmo	4 horas	1 horas
	Total	**Total**
	44 horas	33 horas

Tabla 9: Planificación Tareas Sprint 3

Recopilación Scrum Diario Sprint 3				
Fecha	**¿Qué se ha realizado el día anterior?**	**¿Qué problemas hubo?**	**¿Cómo se va a solucionar?**	**¿Qué se va a hacer este día?**
14/04/2020				**T9.6**
15/04/2020	T9.6			**T9.6**
16/04/2020	**T9.6**			**T10.1 T10.2**
17/04/2020	**T10.1 T10.2**			**T10.2**
20/04/2020	**T10.2**			**T10.2 T10.3**
21/04/2020	**T10.2 T10.3**			**T10.3 T10.4**
22/04/2020	**T10.3 T10.4**			**T10.4 T10.5**
23/04/2020	**T10.4 T10.5**			**T10.5 T10.6**
24/04/2020	**T10.5 T10.6**			

Tabla 10: Scrum diario Sprint 3

Revisión del Sprint 3	
Tareas del Sprint	**Tareas Completadas**
T9.6, T10.1, T10.2, T10.3, T10.4, T10.5, T10.6	T9.6, T10.1, T10.2, T10.3, T10.4, T10.5, T10.6
	Porcentaje completo
	100%

Tabla 11: Revisión Sprint 3

Sprint 4		
Tarea	**Duración estimada**	**Duración real**
T11 Diseñar Controlador Latex2PDF	4 horas	4 horas
T12 Crear Controlador Latex2PDF	4 horas	4 horas
T13 Generar PDF	8 horas	8 horas
T14.1 Crear Servlet descarga PDF completo	8 horas	8 horas
T14.2 Crear Servlet descarga PDF solo preguntas	4 horas	4 horas
T14.3 Crear Servlet descarga TeX	4 horas	4 horas
T15 Crear vista de descarga archivos	8 horas	8 horas
	Total	**Total**
	40 horas	40 horas

Tabla 12: Planificación Tareas Sprint 4

Recopilación Scrum Diario Sprint 4				
Fecha	**¿Qué se ha realizado el día anterior?**	**¿Qué problemas hubo?**	**¿Cómo se va a solucionar?**	**¿Qué se va a hacer este día?**
18/05/2020				**T11**
19/05/2020	T11			**T12**
20/05/2020	T12			**T13**
21/05/2020	T13	Problema comando latex2PDF, no se generan los archivos auxiliares en las carpetas correspondientes	Revisar documentación latex2PDF, buscar un comando diferente o cambiar de librería.	**T13**
22/05/2020	T13			**T14.2**
23/05/2020	T14.2			**T14.3**
24/05/2020	T14.3			**T14.1**
27/05/2020	T14.1			**T14.1**
28/05/2020	T14.1			**T15**
29/05/2020	T15	El Servlet debería descargar un archivo y en cambio se abre un HTML en blanco	Debug Servlet, modificar Servlets PDF completo, TeX y PDF solo preguntas	**T15**

Tabla 13: Scrum diario Sprint 4

Revisión del Sprint 4	
Tareas del Sprint	**Tareas Completadas**
T11, T12, T13, T13, T14.1, T14.2, T14.3, T15	T11, T12, T13, T13, T14.1, T14.2, T14.3, T15
	Porcentaje completo
	100%

Tabla 14: Revisión Sprint 4

Sprint 5		
Tarea	**Duración estimada**	**Duración real**
T16.1 Crear Operación Limite	2 horas	2 horas
T16.2 Crear Formulario Limite	2 horas	2 horas
T16.3 Crear Scripts Limite	4 horas	4 horas
T16.4 Tratamiento del Formulario Limite en Servlet	4 horas	4 horas
T16.5 Interpretación Salida Maxima para Limite	2 horas	2 horas
T17.1 Crear Operación Derivada	2 horas	2 horas
T17.2 Crear Formulario Derivada	2 horas	2 horas
T17.3 Crear Scripts Derivada	4 horas	4 horas
T17.4 Tratamiento del Formulario Derivada en Servlet	4 horas	4 horas
T17.5 Interpretación Salida Maxima para Derivada	2 horas	2 horas
T18.1 Crear Operación Integral	2 horas	2 horas
T18.2 Crear Formulario Integral	2 horas	2 horas
T18.3 Crear Scripts Integral	4 horas	4 horas
T18.4 Tratamiento del Formulario Integral en Servlet	4 horas	4 horas
T18.5 Interpretación Salida Maxima para Integral	2 horas	2 horas
T19.1 Crear Operación Matriz	2 horas	2 horas
T19.2 Crear Formulario Matriz	2 horas	2 horas
T19.3 Crear Scripts Matriz	4 horas	4 horas
T19.4 Tratamiento del Formulario Matriz en Servlet	4 horas	4 horas
T19.5 Interpretación Salida Maxima para Matriz	2 horas	2 horas
T20.1 Crear Operación Polinomio	2 horas	2 horas
T20.2 Crear Formulario Polinomio	2 horas	2 horas
T20.3 Crear Scripts Polinomio	4 horas	4 horas
T20.4 Tratamiento del Formulario Polinomio en Servlet	4 horas	4 horas
T20.5 Interpretación Salida Maxima para Polinomio	2 horas	3 horas
T21 Generar graficas	10 horas	9 horas

Tabla 15: Planificación Tareas Sprint 5

Recopilación Scrum Diario Sprint 5				
Fecha	**¿Qué se ha realizado el día anterior?**	**¿Qué problemas hubo?**	**¿Cómo se va a solucionar?**	**¿Qué se va a hacer este día?**
20/07/2020				**T16.1 716.2**
21/07/200	**T16.1 716.2**			**T16.3**
22/07/2020	**T16.3**			**T16.4**
23/07/2020	**T16.4**			**T16.5 T17.1**
24/07/2020	**T16.5 T17.1**			**T17.2 T17.3**
27/07/2020	**T17.2 T17.3**	Problemas con la generación de algunas opciones aleatorias	Revisar expresiones, revisar ids del formulario Derivadas	**T17.3 T17.4**
28/07/2020	**T17.3 T17.4**			**T17.4 T17.5**
29/07/2020	**T17.4 T17.5**			**T18.1 T18.2**
30/07/2020	**T18.1 T18.2**			**T18.3**
31/07/2020	**T18.3**			**T18.4**
03/08/2020	**T18.4**			**T18.5 T19.1**
04/08/2020	**T18.5 T19.1**			**T19.2 T19.3**
05/08/2020	**T19.2 T19.3**			**T19.3 T19.4**
06/08/2020	**T19.3 T19.4**			**T19.4 T19.5**
07/08/2020	**T19.4 T19.5**			**T20.1 T20.2**
10/08/2020	**T20.1 T20.2**			**T20.3**
11/08/2020	**T20.3**			**T20.4**
12/08/2020	**T20.4**			**T20.5 T21**
13/08/2020	**T20.5 T21**	Problema para detectar de manera correcta y corregir la salida de cada matriz	Mejorar las expr. regulares que hacen de filtro y cambiar el tratamiento de los String de solución en Matriz	**T21 T20.5**
14/08/2020	**T21 T20.5**	Problemas de visualización en las expresiones logarítmicas o potencias	Encontrar la configuración adecuada en la implementación de gráficas o buscar librerías alternativas	**T21**

Tabla 16 : Scrum diario Sprint 5

Revisión del Sprint 5	
Tareas del Sprint	**Tareas Completadas**
T16.1, T16.2, T16.3, T16.4, T16.5, T17.1, T17.2, T17.3, T17.4, T17.5, T18.1, T18.2, T18.3, T18.4, T18.5, T19.1, T19.2, T19.3, T19.4, T19.5, T20.1, T20.2, T20.3, T20.4, T20.5, T21	T16.1, T16.2, T16.3, T16.4, T16.5, T17.1, T17.2, T17.3, T17.4, T17.5, T18.1, T18.2, T18.3, T18.4, T18.5, T19.1, T19.2, T19.3, T19.4, T19.5, T20.1, T20.2, T20.3, T20.4, T20.5, T21
	Porcentaje completo
	100%

Tabla 17: Revisión Sprint 5

5. Arquitectura

Un sitio web se puede definir como una colección o un conjunto de documentos y archivos estáticos, como documentos HTML, imágenes y otros archivos.

Este proyecto es una aplicación web, por tanto, es un sitio web, pero además incluye contenido dinámico, y funcionalidad del lado del servidor. Es decir, esta aplicación web ejecuta programas en el servidor, el usuario realiza una petición y el servidor produce una respuesta tras la ejecución de un programa. En este caso la petición por parte del usuario será una serie de formularios que en conjunto dan la información necesaria al servidor para ejecutar la generación de ejercicios aleatorios y su solución, de manera que después la respuesta sean estos ejercicios.

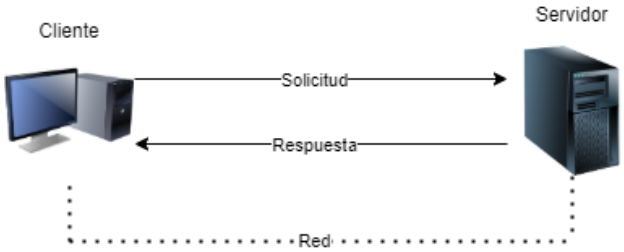


Ilustración 6: Modelo Cliente-Servidor

El proyecto se ha realizado siguiendo una arquitectura y estructura Maven para aplicaciones web, utilizando las herramientas que provee JavaEE.

5.1. JavaEE

JavaEE o Java Enterprise Edition es un conjunto de APIs que reducen la complejidad, mejoran el rendimiento y reducen el tiempo de desarrollo de las aplicaciones. El objetivo principal de la plataforma JavaEE es simplificar el desarrollo ofreciendo bases y arquitecturas de distintos componentes. Uno de ellos son los Servlets. (Jendrock, Cervera-Navarro, Evans, Haase, & Markito, 2014)

5.1.1. Servlet

Una de las partes más importantes de la aplicación web, es el Servlet. Un Servlet es una clase Java que se ejecuta en el servidor, su objetivo es el de procesar peticiones HTTP y generar dinámicamente una respuesta, en este caso genera una respuesta en forma de HTML generado dinámicamente tras procesar los formularios enviados por el usuario desde la vista principal.

Una de las razones para usar este tipo de objetos Java como los Servlet, es que también cada tipo de Operación que se ha implementado es una clase objeto, que se trata con Java de manera muy sencilla al ser un robusto lenguaje orientado a objetos.

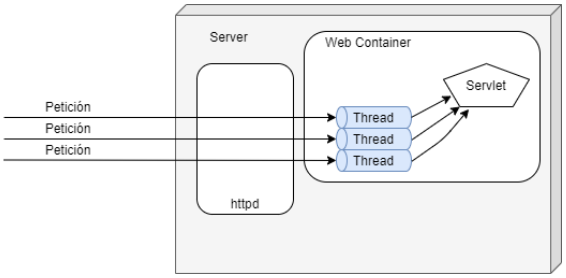


Ilustración 7: Funcionamiento Servidor

En el Servlet se encuentra toda la funcionalidad de la aplicación, en él se procesa la petición del usuario. De manera que se obtienen todos los datos relativos a las operaciones y se preparan los comandos que después ejecutará Maxima, el Servlet procesa la salida de Maxima y da una salida en forma de HTML generado dinámicamente y de documentos tipo PDF y TeX, de manera que contiene las respuestas a los ejercicios generados aleatoriamente.

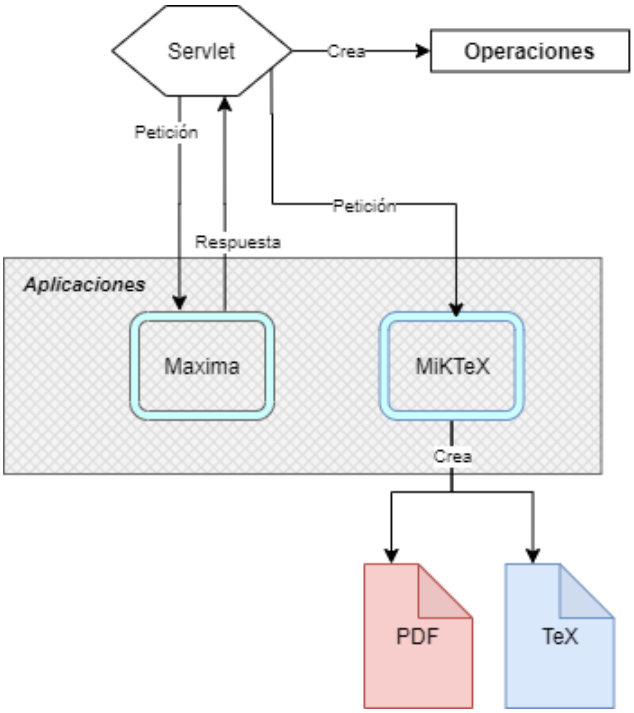


Ilustración 8: Arquitectura Servlet Back-end

5.2. Maven

"*Maven is a tool that can now be used for building and managing any Java-based project".* (Apache Software Foundation, 2020)

La estructura del proyecto se basa en la sugerida para aplicaciones web por Maven en Java, esta estructura de archivos es como la que se puede ver en la Ilustración 9. Es útil para tener una estructura estándar en la organización del proyecto.

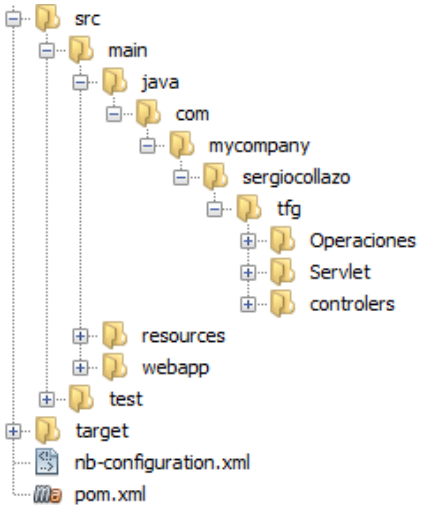


Ilustración 9: Estructura proyecto

Las distintas clases de ejercicios se implementan siguiendo la interfaz de Operación.java y se encuentran en la carpeta Operaciones, como se puede ver en la Ilustración 10.

Los Servlet necesarios se encuentran en la carpeta Servlet, el principal es MaximaResult.java, los otros 3 Servlet se ejecutan solamente si el usuario quiere descargar los archivos que generan.

Los controladores se encuentran en la carpeta controlers.

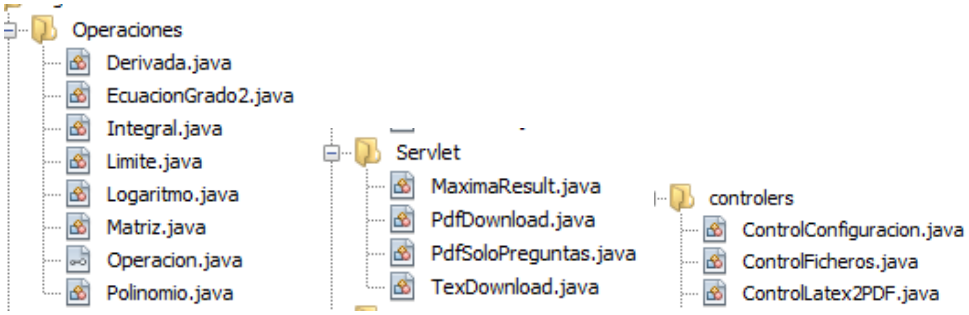


Ilustración 10: Estructura de Operaciones, Servlets y Controladores

6. Tecnologías e integración de productos de terceros

6.1. Interfaz de usuario

6.1.1. HTML

HTML es un lenguaje de marcas que se emplea para dar formato a los documentos que se quieren publicar en la web. Los navegadores son capaces de interpretar las etiquetas y mostrar los documentos con el formato deseado. (Luján Mora, 2002)

Es el lenguaje utilizado para estructurar toda la interfaz de usuario de la aplicación, tanto los documentos HTML estáticos como los generados dinámicamente.

6.1.2. Bootstrap

Bootstrap se creó en Twitter desarrollado por Mark Otto y Jacob Thornton, antes de ser conocido por ser un framework de código abierto se conocía como Twitter Blueprint. Tras unos meses de desarrollo se presentó en la primera Hackweek de Twitter y desarrolladores de todos los niveles comenzaron a usarlo sin guías externas. Ha servido como una guía de estilo para herramientas de desarrollo internas para la compañía durante más de un año hasta que se lanzó al público como código abierto en el año 2011, desde entonces no ha dejado de crecer y actualizarse. (TeamBootstrap, 2020)

En este proyecto se ha utilizado su versión 4.5 para otorgar un estilo estandarizado y coherente a la aplicación web, de manera que sea sencilla e intuitiva de manipular para el usuario.

6.1.3. JavaScript

Es un lenguaje de programación interpretado, basado en prototipos, imperativo y orientado a objetos. Se utiliza en varios ámbitos sin embargo es conocido principalmente por utilizarse como lenguaje de scripting para páginas web. Sigue el estándar ECMAScript. En la actualidad todos los navegadores modernos pueden interpretar este lenguaje al estar integrado en las páginas web. (Mozilla, 2020)

Es el lenguaje más usado en Internet y se puede considerar el estándar de la programación web. La parte del cliente se ejecuta en un navegador y permite intercomunicar el código JavaScript con código Java. Su principal aplicación es validar la entrada introducida por el usuario en formularios y proporcionar dinamismo a las paginas HTML. (Luján Mora, 2002)

En este proyecto se utiliza para dar funcionalidad a la interfaz de usuario, gestionando animaciones e interacciones del usuario con la aplicación, como todo lo relacionado con la gestión de formularios, se ocupa de generar los números aleatorios de entre los escogidos por el usuario, de ocultar y mostrar formularios, así como de enviar los datos al servidor una vez confirmada la configuración de los ejercicios a generar por parte del usuario.

6.1.4. MathJax

MathJax es una librería desarrollada en lenguaje JavaScript que permite visualizar expresiones matemáticas en todos los buscadores de manera simple y efectiva.

MathJax nació del proyecto jsMath, un temprano sistema de renderizado de matemáticas basado en Ajax desarrollado por Davide Cervone en 2004. En 2009 AMS, Desing Science y SIAM formaron el MathJax Consortium para permitir a Cervone continuar con el desarrollo para las plataformas de la siguiente generación. Desde su salida al público en 2010, MathJax se ha convertido en un estándar para las matemáticas en la web. (Cervone & Sorge, 2020)

En este proyecto se emplea precisamente para esa visualización de expresiones matemáticas, en especial en la pantalla de resultados ya que su compatibilidad con el formato TeX es de gran utilidad para la integración con el resto de herramientas de la aplicación.

6.2. Servidor

6.2.1. Java

Java es un lenguaje de programación concurrente, basado en clases, fuertemente tipado y orientado a objetos. Está diseñado de manera suficientemente simple para que muchos programadores puedan adquirir rápidamente fluidez para desarrollar en él. (Gosling, y otros, 2020)

En este proyecto se utiliza para dar funcionalidad a la aplicación web. Los Servlets están plenamente desarrollados como una clase Java y son los encargados de intercomunicar las distintas aplicaciones necesarias como Maxima o MikTeX. Además, es multihilo por lo que permite tener varios Servlets activos en distintos hilos al mismo tiempo, por lo que es un lenguaje ideal para este proyecto.

La especificación de los distintos problemas o ejercicios también están definidos como clases Java, es por esta estructura de clases que se emplea este lenguaje en el proyecto, permite una diferenciación y estructuración muy sencilla.

6.2.2. GlassFish

GlassFish Server Open Source Edition es una aplicación de código libre construida dentro de la comunidad GlassFish, forma parte de las APIs de Java EE y además provee con más herramientas externas a Java EE, pero de utilidad para el desarrollador. (Oracle, 2013)

El proyecto GlassFish provee de un ligero y modular servidor para el desarrollo de Java EE y de un proceso estructurado para desarrollar usando esta tecnología, pero manteniendo una de las características más importantes de Java EE que es la compatibilidad. (Oracle, 2017)

En este proyecto GlassFish Server permite la comunicación entre cliente-servidor utilizando protocolos como HTTP/HTTPs, en general ofrece varios servicios como: Servicio de nombres, gestión de transacciones, clustering, consola de administración y seguridad.

6.3. Entorno de desarrollo

6.3.1. NetBeans

Apache NetBeans Platform es un extenso entorno de desarrollo libre en el que se pueden basar un gran número de aplicaciones. Simplifica el desarrollo de software a varios niveles. Cada característica distintiva de NetBeans se obtiene de módulos, que son similares a los plugin. Un grupo de módulos en NetBeans es un grupo de clases Java que dan soporte o funcionalidad a una característica determinada. (Apache Software Foundation, 2020)

En este proyecto se utiliza como entorno de desarrollo, facilita el desarrollo de la aplicación web ya que centraliza toda la programación en un solo entorno, que además integra las herramientas de servidor y estructura que son necesarias. Se utiliza como editor de todos los lenguajes del proyecto como HTML, JavaSript y Java. Además de usar Maven y Glassfish Server como módulos integrados.

6.4. Aplicaciones externas

6.4.1. Maxima

Maxima es un sistema para la manipulación de expresiones simbólicas y numéricas, incluyendo diferenciación, integración, expansión en series de Taylor, transformadas de Laplace, ecuaciones diferenciales ordinarias, sistemas de ecuaciones lineales, vectores, matrices y tensores. Maxima produce resultados de alta precisión usando fracciones exactas, números enteros de precisión arbitraria y números de coma flotante con precisión variable. Adicionalmente puede graficar funciones y datos en dos y tres dimensiones. (Maxima, 2020)

Maxima es un descendiente de Macsyma, el legendario sistema de álgebra computacional desarrollado a finales de 1960 en el Instituto Tecnológico de Massachusetts (MIT). Este es el único sistema basado en ese programa que está todavía disponible públicamente y con una comunidad activa de usuarios, gracias a la naturaleza del software abierto. (Maxima, 2020)

En este proyecto se utiliza como la aplicación que resuelve los ejercicios generados por el servidor, este se comunica con el sistema Maxima y provee una solución en forma de texto de consola o en formato TeX, que es de gran utilidad a la hora de generar PDFs que incluyan ecuaciones o expresiones matemáticas al mismo tiempo que también es de ayuda para visualizar estas expresiones en HTML gracias a otras herramientas y librerías como MathJax que se integran perfectamente con este tipo de salida.

6.4.2. MiKTeX

MiKTeX es una implementación actualizada de TeX/LaTeX y programas relacionados. Ofrece una serie de comandos y herramientas para integrar de manera sencilla el formato o sistema TeX. (Schenk, 2020)

En este proyecto se utiliza para generar archivos PDF con un formato matemático que permita la rápida interpretación de la expresión matemática y tenga presencia y estética.

6.4.2.1. TeX

TeX es un compositor del lenguaje. En vez de formatear visualmente el texto lo que se hace es escribir directamente el texto manuscrito entrelazado con comandos TeX en un archivo de texto plano. (TUG, 2020) Después ejecutando un comando de la herramienta se genera un PDF con el formato correcto.

Este sistema fue diseñado en su mayoría por Donald Knuth y publicado en 1978. (Per, 2020)

6.5. Function Plot

Function Plot es una librería de graficas construida sobre D3.js utilizada para renderizar funciones con poca configuración previa. Además, soporta gráficos interactivos en los que se puede hacer zoom, moverlos y cambiar la escala. (Mauricio, 2020)

En la aplicación web se utiliza esta librería para agilizar la creación de gráficas, de manera que evalúan las funciones como una ecuación de segundo grado sin mucho nivel de complicación y pudiendo ejecutarse de manera dinámica en el Servlet.

6.5.1. D3.js

D3.js es una librería para manipular elementos basados en datos. D3.js ayuda a trasladar a HTML, SVG y CSS de manera gráfica datos de manera libre.

D3 permite enlazar datos arbitrarios a un Document Object Model (DOM) y después aplicar transformaciones impulsadas por datos al documento para generar gráficos con transiciones ligeras y pulidas. (Bostock, 2020)

7. Especificación y análisis de requisitos

En este apartado de la memoria se describen los requisitos necesarios para el desarrollo de la aplicación web, al aplicarse una metodología ágil es necesario recordar que estos requisitos no son finales y pueden estar sujetos a cambios durante el desarrollo, por ello en la memoria se divide este apartado en dos partes, primero los requisitos después de la primera toma de requisitos y segundo los requisitos después de la segunda toma.

7.1. Primera recopilación de requisitos

7.1.1. Requisitos funcionales

La aplicación debe:

1- Generar ejercicios aleatorios para las operaciones: Ecuación de segundo grado, Logaritmo, Limite, Derivada, Integral, Matriz.
2- Resolver los ejercicios aleatorios con un software dedicado a matemáticas.
3- Generar un documento PDF que muestre una lista de ejercicios y resultados.
4- Permitir que el usuario configure el número de ejercicios que quiere de cada tipo y los valores aleatorios que pueden contener.
5- Gestionar la configuración de la aplicación por parte del administrador.

El Product backlog antes de subdividir las tareas y especificar sus requisitos queda de la siguiente manera:

Nº	Nombre tarea
1	Estructurar aplicación web
2	Resolver los ejercicios aleatorios con un software dedicado a matemáticas.
3	Gestionar la configuración de la aplicación por parte del administrador.
4	Crear Interfaz de Usuario
5	Generar ejercicios aleatorios para la operación Ecuación segundo grado
6	Generar ejercicios aleatorios para la operación Logaritmo
7	Generar ejercicios aleatorios para la operación Limite
8	Generar ejercicios aleatorios para la operación Derivada
9	Generar ejercicios aleatorios para la operación Integral
10	Generar ejercicios aleatorios para la operación Matriz
11	Generar PDF

Tabla 18: Product backlog inicial

La Tabla 19: Product backlog subdividido es la representación del Product backlog una vez subdivididas las tareas y requisitos recogidos en un primer momento, además se han añadido tareas nuevas para tener mejor estructurado el desarrollo del proyecto y especificar mejor que se realizará en cada Sprint. Las tareas que se añaden en este momento no aumentan o amplían las funcionalidades si no que se añaden porque son necesarias para que se cumplan los requisitos recogidos.

Nº	Nombre tarea
1.1	Estudio estructuras software
1.2	Diseño estructura software
1.3	Creación estructura software
2.1	Estudio conector Java-Maxima
2.2	Diseño conector Java-Maxima
2.3	Creación conector Java-Maxima
3	Creación StreamGobbler
4	Creación Controlador de Configuración
5	Creación Controlador de Ficheros
6	Crear Servlet
7.1	Diseñar Interfaz Usuario
7.2	Integrar Interfaz Usuario
8	Crear interfaz Operación
9.1	Crear Operación Ecuación 2º Grado
9.2	Crear Formulario Ecuación 2º Grado
9.3	Crear Scripts Formulario Ecuación 2ºGrado
9.4	Tratamiento del Formulario Ecuación 2ºGrado en Servlet
9.5	Interpretación Salida Maxima para Ecuación 2º Grado
9.6	Mostrar Resultados Ecuación 2º Grado en HTML
10.1	Crear Operación Logaritmo
10.2	Crear Formulario Logaritmo
10.3	Crear Scripts Logaritmo
10.4	Tratamiento del Formulario Logaritmo en Servlet
10.5	Interpretación Salida Maxima para Logaritmo
10.6	Mostrar Resultados Logaritmo
11	Diseñar Controlador Latex2PDF
12	Crear Controlador Latex2PDF
13	Generar PDF
14	Crear Servlet descarga PDF completo
15	Crear vista de descarga archivos
16.1	T16.1 Crear Operación Limite
16.2	T16.2 Crear Formulario Limite
16.3	T16.3 Crear Scripts Limite
16.4	T16.4 Tratamiento del Formulario Limite en Servlet
16.5	T16.5 Interpretación Salida Maxima para Limite
17.1	T17.1 Crear Operación Derivada
17.2	T17.2 Crear Formulario Derivada
17.3	T17.3 Crear Scripts Derivada
17.4	T17.4 Tratamiento del Formulario Derivada en Servlet
17.5	T17.5 Interpretación Salida Maxima para Derivada
18.1	T18.1 Crear Operación Integral
18.2	T18.2 Crear Formulario Integral
18.3	T18.3 Crear Scripts Integral
18.4	T18.4 Tratamiento del Formulario Integral en Servlet
18.5	T18.5 Interpretación Salida Maxima para Integral
19.1	T19.1 Crear Operación Matriz
19.2	T19.2 Crear Formulario Matriz
19.3	T19.3 Crear Scripts Matriz
19.4	T19.4 Tratamiento del Formulario Matriz en Servlet
19.5	T19.5 Interpretación Salida Maxima para Matriz

Tabla 19: Product backlog subdividido

En la Ilustración 11 se ve cómo se subdivide la tarea 9 en 6 subtareas, además a continuación en las ilustraciones Ilustración 12, Ilustración 13, Ilustración 14, Ilustración 15, Ilustración 16 e Ilustración 17 se puede ver cómo es una tarjeta en Scrum con la tarea o historia de usuario definida y estimada en horas de trabajo.

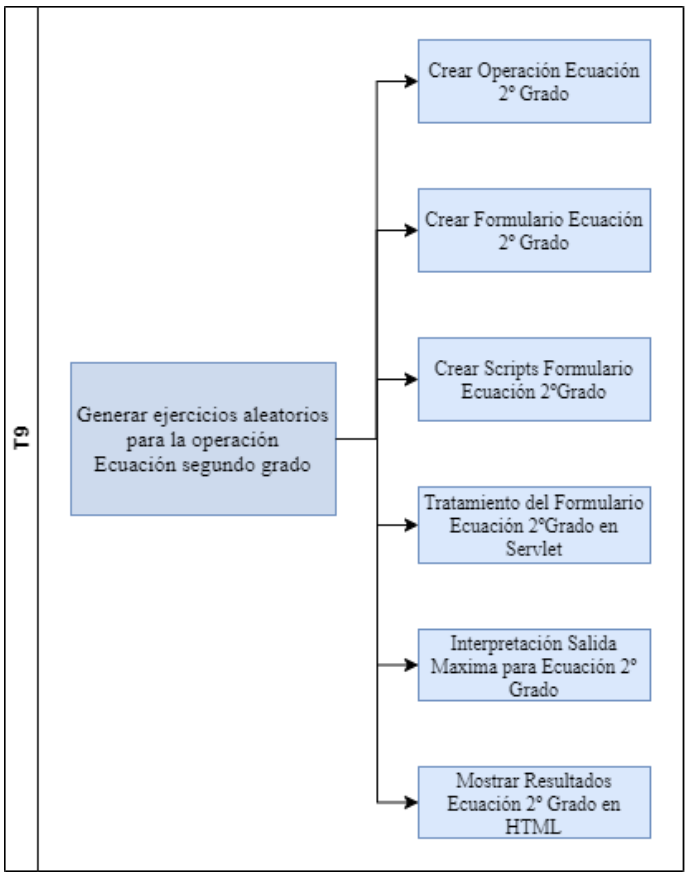


Ilustración 11: División de la tarea 9

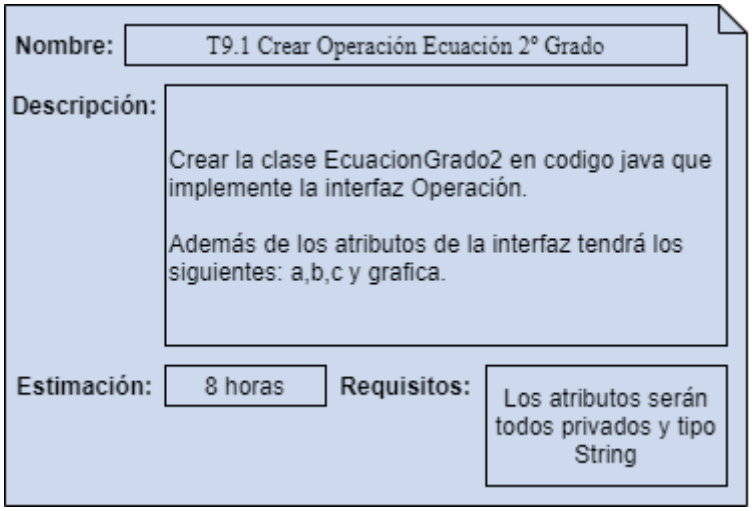


Ilustración 12: Especificación T9.1

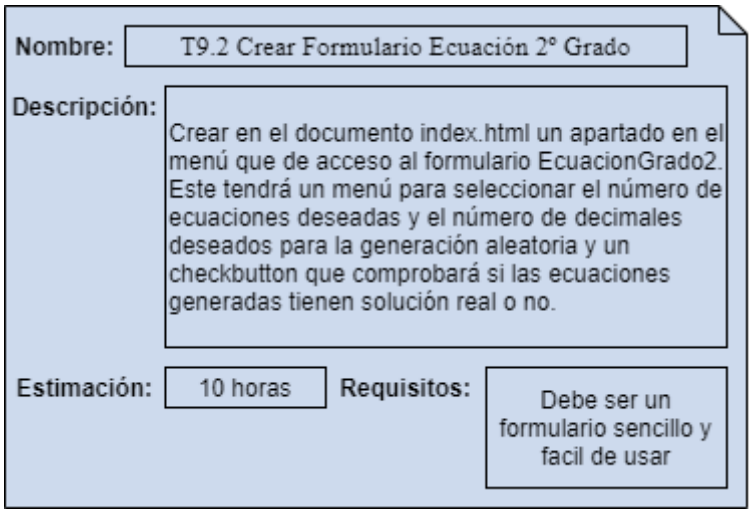


Ilustración 13: Especificación T9.2

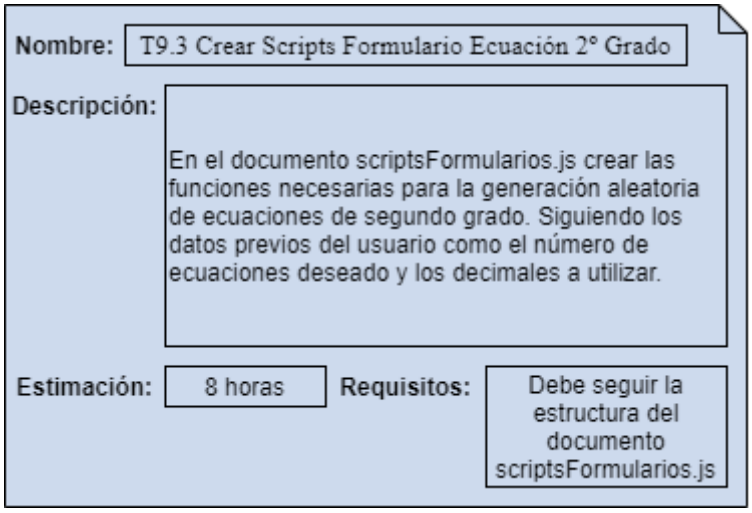


Ilustración 14: Especificación T9.3

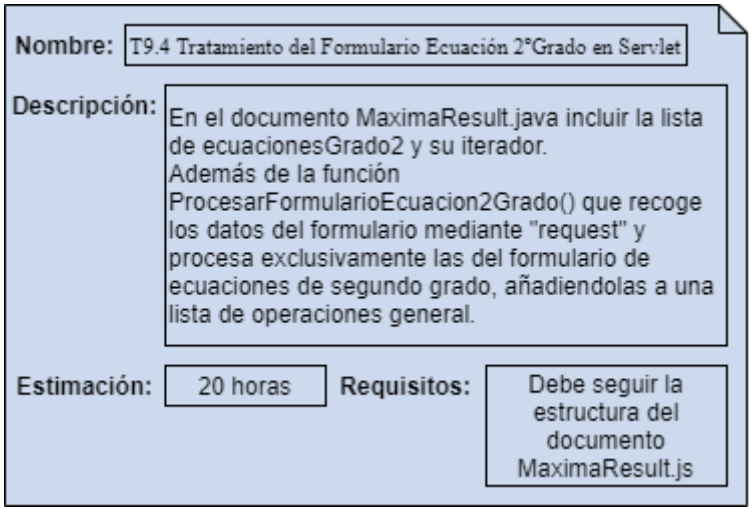


Ilustración 15: Especificación T9.4

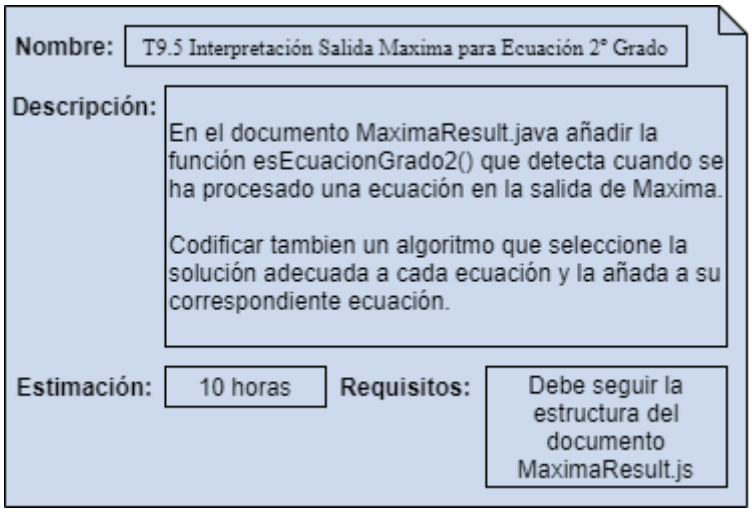


Ilustración 16: Especificación T9.5

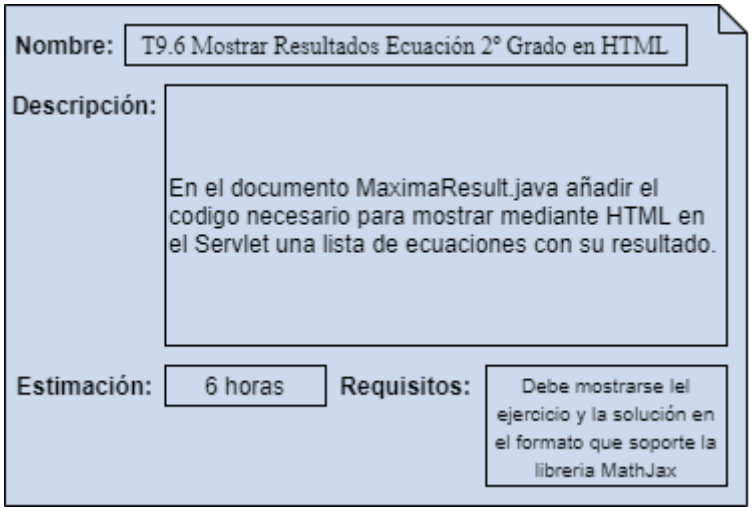


Ilustración 17: Especificación T9.6

7.1.2. Requisitos no funcionales

La aplicación web debe ser:

1- Intuitiva y fácil de usar de manera que no requiera ningún aprendizaje previo para ser utilizada por los usuarios.
2- Seguir un diseño responsive, de manera que se adapte a diversas resoluciones o tamaños de ventana.
3- Funcionar en los buscadores más actuales de manera adecuada.

En las ilustraciones Ilustración 18, Ilustración 19 e Ilustración 20, se puede ver la definición de las tareas o historias de usuario que son necesarias para cumplir con los requisitos no funcionales número 1 y 2.

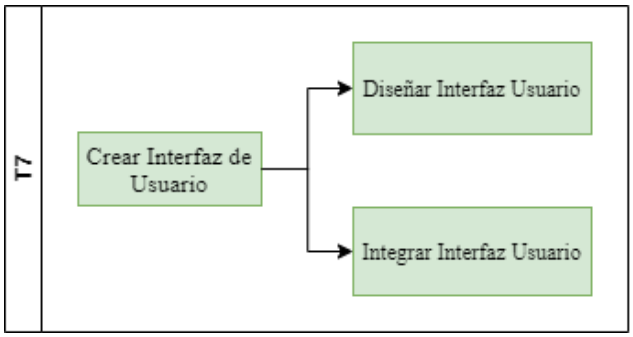


Ilustración 18: División de la tarea 7

Nombre:	T7.1 Diseñar Interfaz Usuario
Descripción:	Realizar un boceto de como se quiere estructurar la interfaz, decidir que estandares o frameworks usar en cuanto a aplicar estilo al documento index.html
Estimación:	4 horas
Requisitos:	Debe ser una interfaz responsive y simple

Ilustración 19: Especificación T7.1

Nombre:	T7.2 Integrar Interfaz Usuario
Descripción:	Codificar la interfaz en el documento index.html de manera que siga el boceto o estructura definida en la T7.1 aplicando el estilo de la libreria bootstrap. Los elementos con id deben tener un atributo name igual al id.
Estimación:	8 horas
Requisitos:	Debe ser una interfaz responsive y simple

Ilustración 20: Especificación T7.2

7.2. Segunda recopilación de requisitos

En esta segunda recopilación se añadieron algunos requisitos funcionales a mayores de los ya existentes en el Product backlog. Sin embargo, no se modifica ni añade ningún requisito no funcional, se mantienen los de la primera toma de requisitos.

7.2.1. Requisitos funcionales

A los requisitos ya existentes en el Product backlog se añaden los siguientes:

1- Generar ejercicios aleatorios para las operaciones: Polinomios de cualquier grado.
2- Generar un documento PDF que muestre solamente los resultados.
3- Generar un documento TeX que muestre los ejercicios y lo resultados.
4- Generar graficas adecuadas para las operaciones: Ecuación de segundo grado, Límite, Derivada e Integral.

Los requisitos que se modifican de los ya existentes en el Product Backlog es el siguiente:

1- Permitir que el usuario configure el número de ejercicios que quiere de cada tipo y los valores aleatorios que pueden contener

Nº	Nombre tarea
1.1	Estudio estructuras software
1.2	Diseño estructura software
1.3	Creación estructura software
2.1	Estudio conector Java-Maxima
2.2	Diseño conector Java-Maxima
2.3	Creación conector Java-Maxima
3	Creación StreamGobbler
4	Creación Controlador de Configuración
5	Creación Controlador de Ficheros
6	Crear Servlet
7.1	Diseñar Interfaz Usuario
7.2	Integrar Interfaz Usuario
8	Crear interfaz Operación
9.1	Crear Operación Ecuación 2º Grado
9.2	Crear Formulario Ecuación 2º Grado
9.3	Crear Scripts Formulario Ecuación 2ºGrado
9.4	Tratamiento del Formulario Ecuación 2ºGrado en Servlet
9.5	Interpretación Salida Maxima para Ecuación 2º Grado
9.6	Mostrar Resultados Ecuación 2º Grado en HTML
10.1	Crear Operación Logaritmo
10.2	Crear Formulario Logaritmo
10.3	Crear Scripts Logaritmo
10.4	Tratamiento del Formulario Logaritmo en Servlet
10.5	Interpretación Salida Maxima para Logaritmo
10.6	Mostrar Resultados Logaritmo
11	Diseñar Controlador Latex2PDF
12	Crear Controlador Latex2PDF
13	Generar PDF
14.1	Crear Servlet descarga PDF completo
14.2	Crear Servlet descarga PDF solo preguntas
14.3	Crear Servlet descarga TeX
15	Crear vista de descarga archivos

16.1	T16.1 Crear Operación Limite
16.2	T16.2 Crear Formulario Limite
16.3	T16.3 Crear Scripts Limite
16.4	T16.4 Tratamiento del Formulario Limite en Servlet
16.5	T16.5 Interpretación Salida Maxima para Limite
17.1	T17.1 Crear Operación Derivada
17.2	T17.2 Crear Formulario Derivada
17.3	T17.3 Crear Scripts Derivada
17.4	T17.4 Tratamiento del Formulario Derivada en Servlet
17.5	T17.5 Interpretación Salida Maxima para Derivada
18.1	T18.1 Crear Operación Integral
18.2	T18.2 Crear Formulario Integral
18.3	T18.3 Crear Scripts Integral
18.4	T18.4 Tratamiento del Formulario Integral en Servlet
18.5	T18.5 Interpretación Salida Maxima para Integral
19.1	T19.1 Crear Operación Matriz
19.2	T19.2 Crear Formulario Matriz
19.3	T19.3 Crear Scripts Matriz
19.4	T19.4 Tratamiento del Formulario Matriz en Servlet
19.5	T19.5 Interpretación Salida Maxima para Matriz
20.1	T20.1 Crear Operación Polinomio
20.2	T20.2 Crear Formulario Polinomio
20.3	T20.3 Crear Scripts Polinomio
20.4	T20.4 Tratamiento del Formulario Polinomio en Servlet
20.5	T20.5 Interpretación Salida Maxima para Polinomio
21	T21 Generar graficas

Tabla 20: Product backlog final

En la Ilustración 21 se puede ver la división en subtareas de la tarea 14 “Descargar Documentos”.

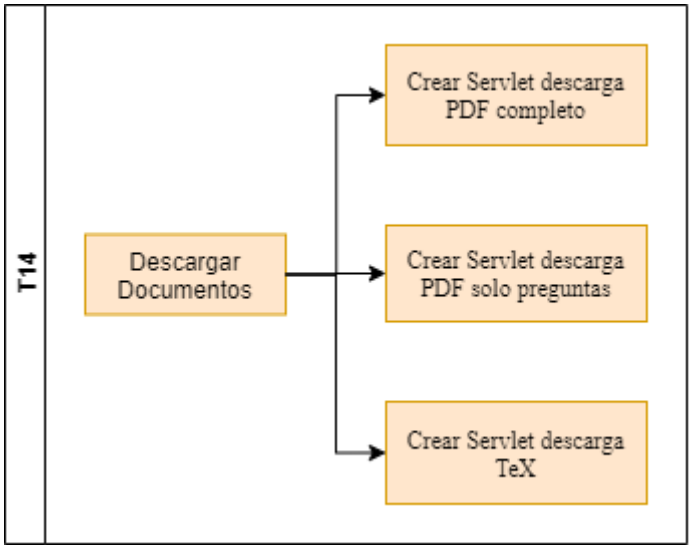


Ilustración 21: División de la tarea 14

En las ilustraciones Ilustración 22, Ilustración 23 e Ilustración 24 se puede ver la definición y estimación de las tareas o historias de usuario que hacen cumplir los requisitos funcionales “Generar un documento PDF que muestre solamente los resultados”, “Generar un documento TeX que muestre los ejercicios y lo resultados” y “Generar un documento PDF que muestre una lista de ejercicios y resultados”. De los cuales los dos primeros son nuevos requisitos que no se encontraban en la primera toma de requisitos

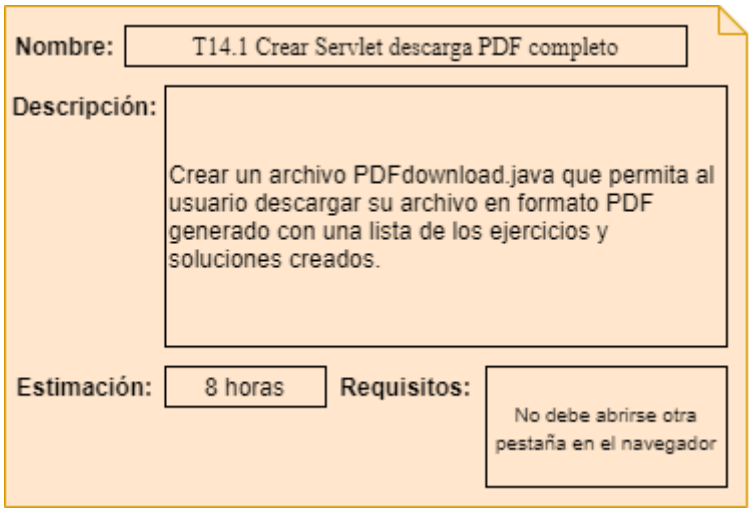


Ilustración 22: Especificación T14.1

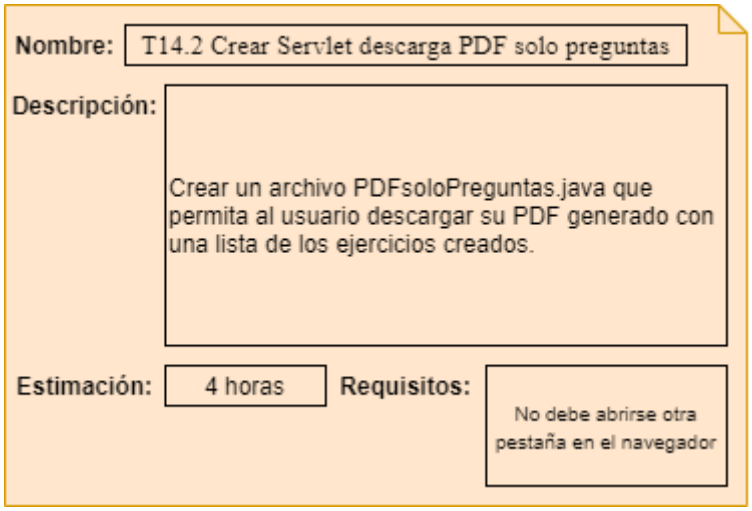


Ilustración 23: Especificación T14.2

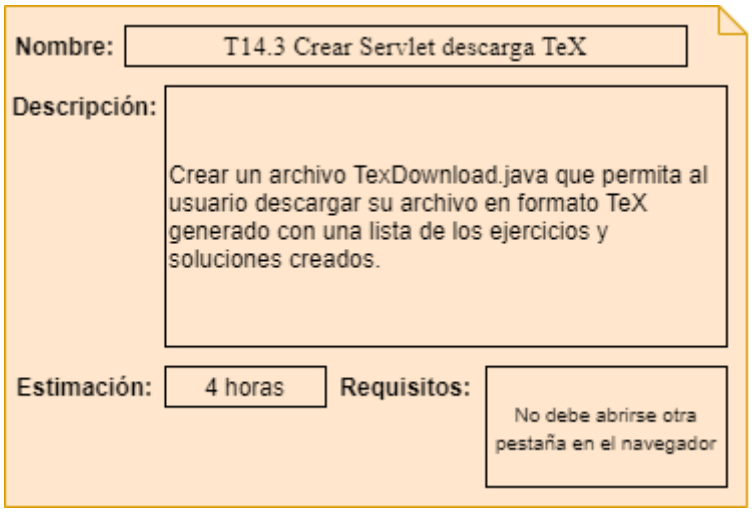


Ilustración 24: Especificación T14.3

8. Diseño del Software

En cuanto al diseño del software lo más destacable es cómo se realiza la conexión entre el Servlet en Java y la aplicación Maxima y cómo se generan los documentos descargables.

Existen conectores u otras aplicaciones que realizaban esta acción, pero enfocada a objetivos distintos y utilizando otros lenguajes como PHP. En el caso de este proyecto la aplicación web se conecta con Maxima mediante el Servlet.

8.1. Java-Maxima

Solo se puede interactuar con la aplicación Maxima mediante comandos o mediante la interfaz gráfica que incorpora WXMaxima, por lo tanto, para que podamos usar esta aplicación en el servidor hay que utilizar el método de comandos. Esto es posible en Java mediante la función *exec*, que nos permite ejecutar un comando del sistema desde el código Java, en este caso ejecutamos un comando en la consola de Windows 10.

El comando de Maxima que se utiliza es: 'Ruta absoluta Maxima' -b 'ruta archivo batch';

Como se puede apreciar es necesario un archivo batch, que no es más que un archivo .mac que contiene las instrucciones que después ejecuta Maxima. La salida de la ejecución por consola de las instrucciones de Maxima hay que tratarla, aquí existen varias opciones en el caso de esta aplicación se crea un fichero de texto con el contenido de la salida para poder trabajar con ella posteriormente, con una función llamada procesarSalidaMaxima(File f), siendo f el archivo creado con la salida de las instrucciones procesadas por Maxima.

```
private void ejecutarComandoConsola() throws IOException, InterruptedException {
    String comando = diretorioMaxima +
            " -b " +
            controlconf.getNombreArchivoBatch();
    Process process;
    process = Runtime.getRuntime().exec(String.format("cmd.exe /c %s", comando));
    StreamGobbler streamGobbler =
            new StreamGobbler(process.getInputStream(),
                    CF.creaFichero(direcorioArchivosBatch, controlconf.getNombreSalidaMaxima()));
    Executors.newSingleThreadExecutor().submit(streamGobbler);
    int exitCode = process.waitFor();
    assert exitCode == 0;
    procesarSalidaMaxima(CF.creaFichero(direcorioArchivosBatch, controlconf.getNombreSalidaMaxima()));
}
```

Ilustración 25: Función ejecutarComandoConsola().

El archivo generado se procesa línea a línea detectando las operaciones correspondientes y asignando sus respuestas una a una. La parte de más complejidad, es este proceso en el que se trabaja con un gran número de expresiones regulares complejas para discernir lo que es un enunciado, respuesta, error, etc.

Una vez asignadas las respuestas a cada operación simplemente se recorre una lista con los ejercicios y se procede a generar el código HTML de manera dinámica, además de generar todos los documentos que puede descargar el usuario.

8.2. Generación de documentos descargables

Para la generación de documentos descargables[2] se utiliza la clase ControlLatex2PDF que se encarga de crear la plantilla vacía en la que se introducirán los ejercicios como se puede ver en la Ilustración 26.

```
this.cabecera = "\\documentclass[a4paper,11pt]{article}\n"
        + "\n"
        + "\\usepackage[latin1]{inputenc}\n"
        + "\\usepackage{color}\n"
        + "\\usepackage{array}\n"
        + "\\usepackage{amsmath,amssymb}\n"
        + "\n"
        + "\\addtolength{\\textwidth}{2cm}\n"
        + "\\addtolength{\\hoffset}{-1cm}\n"
        + "\n"
        + "\\title{Ejercicios}\n"
        + "\n"
        + "\\begin{document}\n"
        + "\n"
        + "\\maketitle\n"
        + "\n"
        + "";
this.footer = "\n"
        + "\n"
        + "\n"
        + "\\end{document}";
```

Ilustración 26: Código plantilla TeX en Java

Esta primera plantilla se rellena con los ejercicios generados y sus respuestas (en caso de ser necesarias) con la función que se puede ver en la Ilustración 27. Esta función tiene como parámetro una lista de cadenas de texto, estas cadenas son las líneas a añadir a la plantilla y forman los enunciados de los ejercicios y sus respuestas. Para ello simplemente recorre esta lista y una por una (ya están ordenadas) escribe las líneas en otra lista de cadenas.

```
public List<String> construirTemplateLatex(List<String> contenido) {
    contenidoDocumentoLatex.add(this.cabecera);
    Iterator it = contenido.iterator();
    while (it.hasNext()) {
        String c = (String) it.next();
        contenidoDocumentoLatex.add(c);
    }
    contenidoDocumentoLatex.add(this.footer);
    return contenidoDocumentoLatex;
}
```

Ilustración 27: Función construirTemplateLatex

Esta lista de cadenas se usará finalmente para crear un archivo de texto con formato TeX, que después de ejecutar un comando de la herramienta MiKTeX en este caso el comando 'pdflatex' como se ve en la Ilustración 28, creará los documentos descargables. Este comando se ejecuta de igual manera que el de Maxima, pero sus parámetros cambian, necesita la ruta absoluta donde se generan los archivos de salida y la ruta absoluta del archivo TeX creado en el paso anterior.

```
String comando = "pdflatex -output-directory="+dirOutput+" " + rutaPdflatex+"\\" + nombreArchivoTex;
Process process;
process = Runtime.getRuntime().exec(String.format("cmd.exe /c %s", comando));
StreamGobbler streamGobbler = new StreamGobbler(process.getInputStream());
```

Ilustración 28: Código ejecutar comando MiKTeX

[2] El documento PDF con ejercicios y sus respuestas, el documento PDF con solamente los ejercicios y el documento TeX con ejercicios y su respuesta.

9. Gestión de datos e Información

Durante la ejecución de la aplicación web se generan diversos archivos, algunos de funcionamiento interno y otros para que el usuario pueda descargarlos si lo desea. Por ello se debe usar un método de generación de nombres lo más seguro posible.

Para asegurar que el nombre de los archivos es único se emplea la librería *java.util.UUID*. UUID o Universally Unique IDentifier es un número de 128 bits de largo y que asegura unicidad a través de espacio y tiempo. Los UUIDs se utilizaron originalmente en el sistema Apollo Network Computing y posteriormente en la OSF DCE. (Leach, Mealling, & Salz, 2005)

El formato de los UUID son 16 octetos que se representan como 32 dígitos hexadecimales dispuestos en 5 grupos separados por un guion, de la forma 8-4-4-4-12 para formar un total de 36 caracteres, es decir, 32 cifras hexadecimales y 4 guiones.

Un ejemplo de formato UUID es el siguiente: 5e10940e-1ff0-4a06-b617-1087050e26c3

También se utiliza la librería *java.security.SecureRandom*. Esta librería ofrece una generación de un número criptográficamente fuerte. (Oracle, 2020)

En el proyecto se implementa una solución para la generación de nombres únicos para los archivos utilizando ambas librerías tal como se ve en la Ilustración 29.

```
public void generarID(){
    UUID id = UUID.randomUUID();
    UUID id2 = UUID.randomUUID();
    UUID id3 = UUID.randomUUID();
    UUID id4 = UUID.randomUUID();

    String time = id.toString().replace("-", "");
    String time2 = id2.toString().replace("-", "");
    String time3 = id3.toString().replace("-", "");
    String time4 = id4.toString().replace("-", "");

    StringBuffer data = new StringBuffer();
    data.append(time);
    data.append(time2);
    data.append(time3);
    data.append(time4);

    SecureRandom random = new SecureRandom();
    int beginIndex = random.nextInt(100);
    int endIndex = beginIndex + 10;            //bengi

    String ID = data.substring(beginIndex, endIndex);

    this.nombreArchivoBatch=ID+".mac";
    this.nombreArchivoTex=  ID+".tex";
    this.nombreArchivoTexSoloPreguntas=  ID+"p.tex";
    this.nombreArchivoPDF=  ID+".pdf";
    this.nombreArchivoPDFSoloPreguntas=ID+"p.pdf";
    this.nombreSalidaMaxima=ID+".txt";
    this.nombreAuxPDFlatex=ID+".aux";
    this.nombreAuxPDFlatexSoloPreguntas=ID+"p.aux";
    this.id=ID;
}
```

Ilustración 29: Función generarID

Se generan 4 UUIDs diferentes, y se eliminan los guiones, por lo que se generan 4 UUIDs de 32 caracteres, todos ellos se concatenan en un String llamado 'data' dejando como resultado una cadena de 128 caracteres. Como un archivo con 128 caracteres como nombre es demasiado largo, se utiliza la función SecureRandom() para de manera criptográficamente segura y aleatoria generar un número que marcará el comienzo de la subcadena de la cadena 'data', después se escoge el número de caracteres que se desean para el nombre de los archivos. Al ser el número de comienzo de la subcadena que formará el ID final un número entre 0-100 en caso de ser ese número 100 el número máximo no podrá ser superior a 27 ya que existiría la posibilidad de error al ejecutar la función substring(begin,end).

10. Pruebas llevadas a cabo

10.1. Pruebas de caja negra

Las pruebas de caja negra son aquellas que se centran en analizar las salidas de un programa o parte de código, ante unos parámetros de entrada específicos.

Antes de la finalización de cada tarea para que se pueda considerar una tarea completa hay que probarla, para ello se pueden emplear test Junit, que ayudan a comprobar las distintas casuísticas que se pueden dar y sirven para cumplir las restricciones marcadas y el correcto funcionamiento de la funcionalidad que se añaden.

El framework Junit para Java está enfocado a realizar pruebas unitarias, estas pruebas están enfocadas en verificar una sección del código. En el caso de este proyecto existen pruebas para cada clase, en concreto han sido de mucha utilidad en las clases de tipo Operación.

Por ejemplo, en la Ilustración 30 se puede ver una prueba de un método de la clase Límite, en la cual sabemos que la librería que se utiliza para crear graficas tiene problemas para aceptar funciones como $\sec(x)$, pero como $\sec(x) = 1/\cos(x)$ entonces para los casos especiales la función GetGrafica() de la clase Limite debería devolver la función $1/\cos(x)$, para no tener que comprobarlo para todos los casos y cada vez que se realiza un cambio en la clase y para asegurar de que se ha construido sin errores se crean test para cada función, comprobando los puntos críticos, que son aquellos más propensos a contener errores.

En el caso de la Ilustración 30 se puede ver que se comprueban otros casos además de la devolución de la función $1/\cos(x)$ si se crea el límite con una expresión $\sec(x)$.

```
/**
 * Test of getGrafica method, of class Limite.
 */
@Test
public void testGetGrafica() {
    System.out.println("getGrafica");
    Limite instance = new Limite("sec(x)","x","1","cod");
    String expResult = "1/cos(x)";
    String result = instance.getGrafica();
    assertEquals(expResult, result);
    instance = new Limite("cot(x)","x","1","cod");
    expResult = "1/tan(x)";
    result = instance.getGrafica();
    assertEquals(expResult, result);
    instance = new Limite("csc(x)","x","1","cod");
    expResult = "1/sin(x)";
    result = instance.getGrafica();
    assertEquals(expResult, result);
}
```

Ilustración 30: Limite Test GetGrafica()

En la Ilustración 31 se puede ver otro test sencillo para en este caso, la clase Matriz.

```
/**
 * Test of crearSentenciaMaxima method, of class Matriz.
 */
@Test
public void testCrearSentenciaMaxima() {
    System.out.println("crearSentenciaMaxima");
    Matriz instance = new Matriz("[1,2,3],[2,3,4]","[4,4,3],[1,3,4]","+");
    String expResult = "s_1 : \"matriz.\";tex(x:matrix([1,2,3],[2,3,4]),false);tex(y:matrix([4,4,3],[1,3,4]),false);tex(x+y,false);";
    String result = instance.crearSentenciaMaxima();
    assertEquals(expResult, result);
    instance = new Matriz("[1,2,3],[2,3,4]","[1,2,3],[2,3,4]","invert");
    expResult = "s_1 : \"matrizI.\";tex(x:matrix([1,2,3],[2,3,4]),false);tex(invert(x),false);";
    result = instance.crearSentenciaMaxima();
    assertEquals(expResult, result);
}
```

Ilustración 31: Matriz Test CrearSentenciaMaxima()

10.2. Pruebas de caja blanca

Las pruebas de caja blanca son aquellas que se centran en analizar la estructura lógica o el flujo de ejecución de un programa, ante unos parámetros de entrada específicos.

En este proyecto se han realizado pruebas en aquellas partes del código que involucran bucles. Una de las secciones de código que más bucles tiene y por tanto de mayor complejidad es la de procesar la salida de Maxima, ya que pasa por varios bucles anidados para discernir entre que ejercicios son, que es enunciado y que es solución, mediante múltiples expresiones regulares.

Otra parte con bucles en la que es importante comprobar su correcto funcionamiento, aunque sean bucles simples es en las funciones de procesar formularios, para comprobar que se han leído todos los datos existentes, hay que comprobar los extremos.

10.3. Múltiples pruebas simultaneas

Para poder realizar los distintos test de cada clase simultáneamente sin tener que pasar uno por uno se crea lo que se conoce como Test Suit que no es más que una clase que agrupa diversos test de manera que están centralizados.

Con Junit solamente hace falta crear una clase con el contenido que se puede ver en la Ilustración 32. Y crear un Test Runner que ejecutará esta clase, de manera que con la etiqueta @Suite.SuiteClasses ya sabe a qué clases llamar para ejecutar sus test propios. En el caso de la Ilustración 32, estamos ejecutando los test de las clases Limite y Matriz simultáneamente sin tener que hacerlo de manera individual clase por clase.

```
@RunWith(Suite.class)
@Suite.SuiteClasses({LimiteTest.class,MatrizTest.class})

public class NewTestSuite {

}
```

Ilustración 32: Test Suit

11. Manual de usuario

11.1. Instalación

11.1.1. Recursos necesarios

Para la instalación de la aplicación web es necesario: un sistema operativo Windows 10 y tener instalado Java 8. Además de tener descargados los jdk1.8.0_231 y jre1.8.0_231 y moverlos a la carpeta Java que se creará automáticamente en el sistema una vez instalado Java 8.

Es necesario un navegador web, recomendado Google Chrome, Internet Explorer o Microsoft Edge.

11.1.2. Instalar Maxima 5.43.2

Es necesario instalar la aplicación Maxima 5.43.2 para ello primero hay que descargar el archivo resaltado en la Ilustración 33 y realizar una instalación completa con las opciones marcadas por defecto.

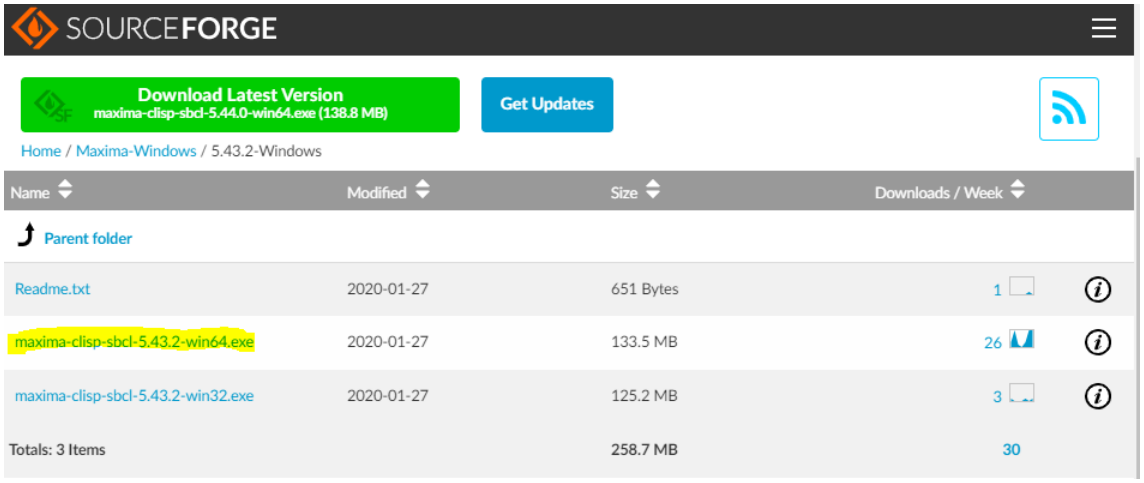


Ilustración 33: Maxima 5.43.2

Mover el archivo llamado 'maxima-init.mac' que se encuentra en la carpeta 'Recursos' del proyecto al directorio creado con la instalación por defecto de Maxima, llamado 'maxima-5.43.2/share/máxima/5.43.2'.

Crear una carpeta llamada 'tmp' en ese mismo directorio. El resultado de estos dos últimos pasos debe ser el que se ve en la Ilustración 34.

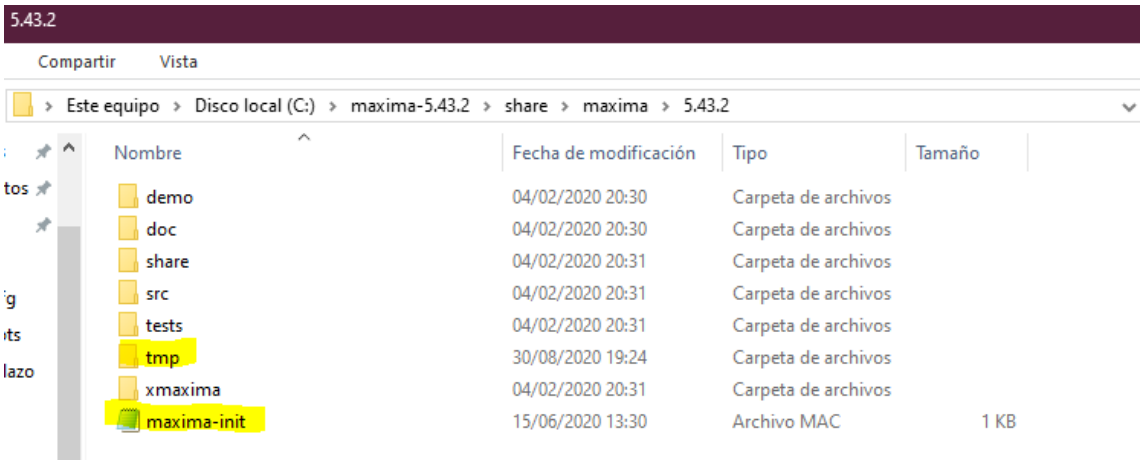


Ilustración 34: Carpeta Maxima

11.1.3. Instalar MiKTeX

Es necesario instalar la aplicación MikTeX para hay que pulsar el botón 'Download' resaltado que se ve en la Ilustración 35Ilustración 35.

Ilustración 35: MiKTeX

Hay que realizar una instalación completa con todas las opciones por defecto excepto la que se ve marcada en la Ilustración 36.

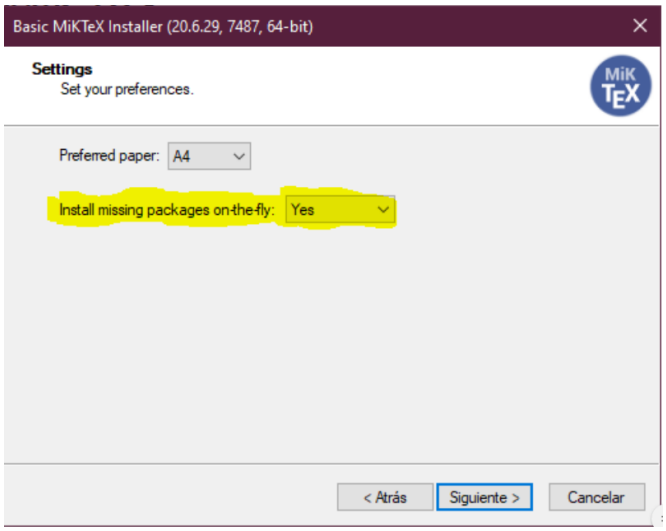


Ilustración 36: Instalación MiKTeX

11.1.4. Apache NetBeans 11.2

Hay que descargar el entorno Apache NetBeans 11.2, para ello descargar el instalador marcado de la Ilustración 37.

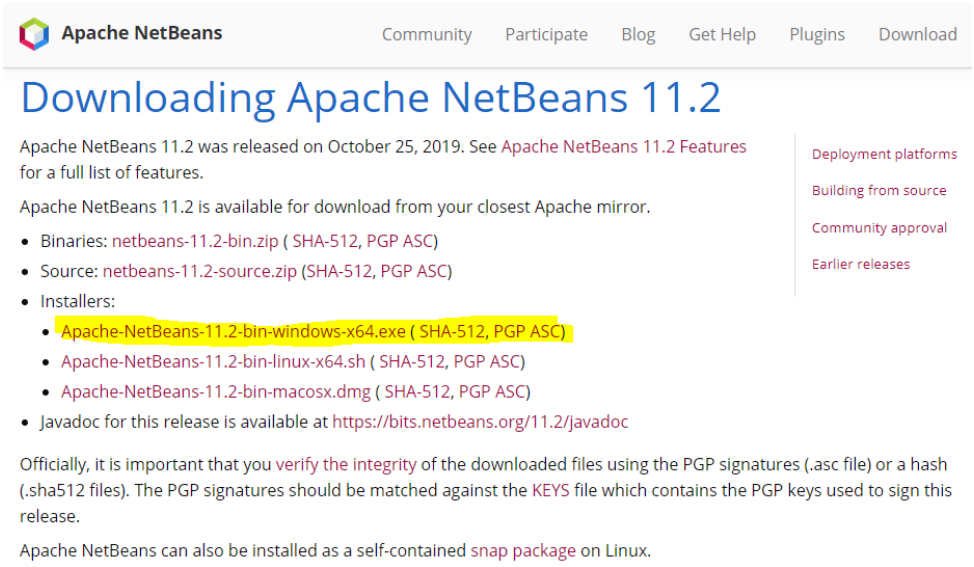


Ilustración 37: NetBeans 11.2

Instalar el entorno con todos los paquetes recomendados. Abrir el entorno Apache NetBeans 11.2 y en la barra de herramientas seleccionar File-New Project, se abrirá la ventana que se puede ver en la Ilustración 38, desde ahí seleccionar 'Java with Maven' y crear una 'Web Application', el nombre y la configuración debe dejarse por defecto.

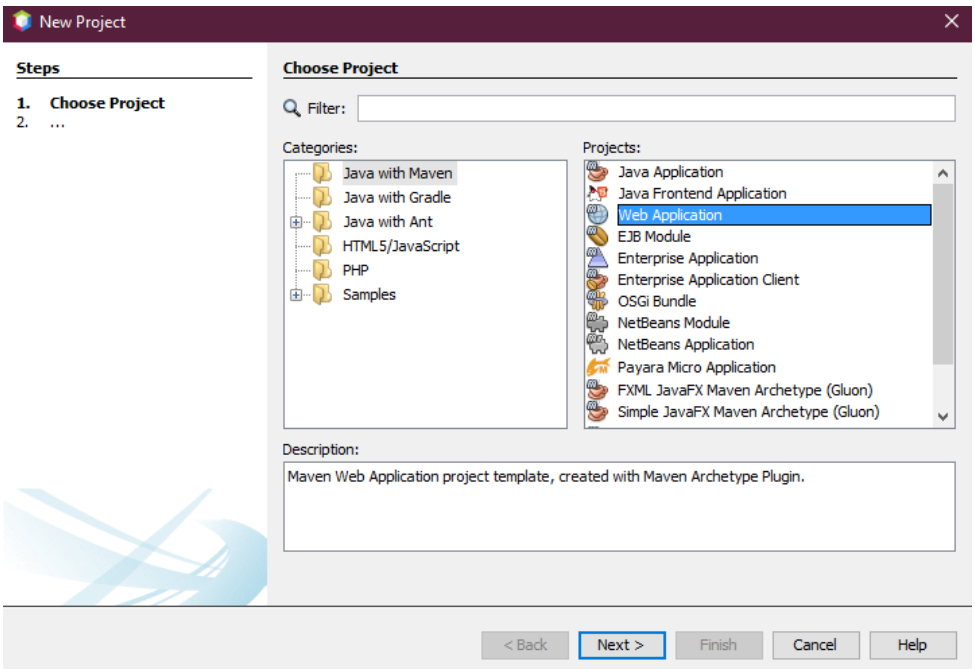


Ilustración 38: Ventana New Project

Con esto se habrá creado una carpeta que representa un espacio de trabajo por defecto llamado 'NetBeansProjects'. Además, se habrá descargado e instalado Maven y otras dependencias necesarias. Ahora es posible borrar el proyecto creado en el paso anterior de ser deseado.

Introducir en el espacio de trabajo la carpeta que contiene al proyecto.

Desde la barra de herramientas seleccionar ‘File-Open Project’, buscar y seleccionar el proyecto movido al espacio de trabajo en el paso anterior.

Ahora debería aparecer el proyecto en el apartado ‘Projects’ como en la Ilustración 39.

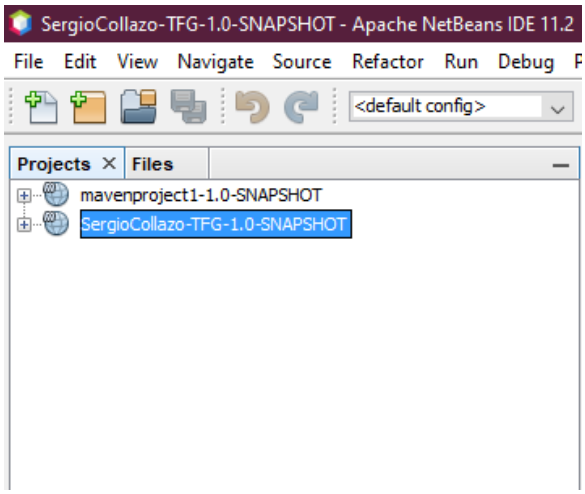


Ilustración 39: Seccion Projects

Hacer clic derecho del ratón sobre el proyecto y seleccionar En el menú desplegable ‘Properties’. En el apartado ‘Build’, seleccionar ‘Compile’ y hacer clic en ‘Manage Java Platforms’ como se ve en la Ilustración 40.

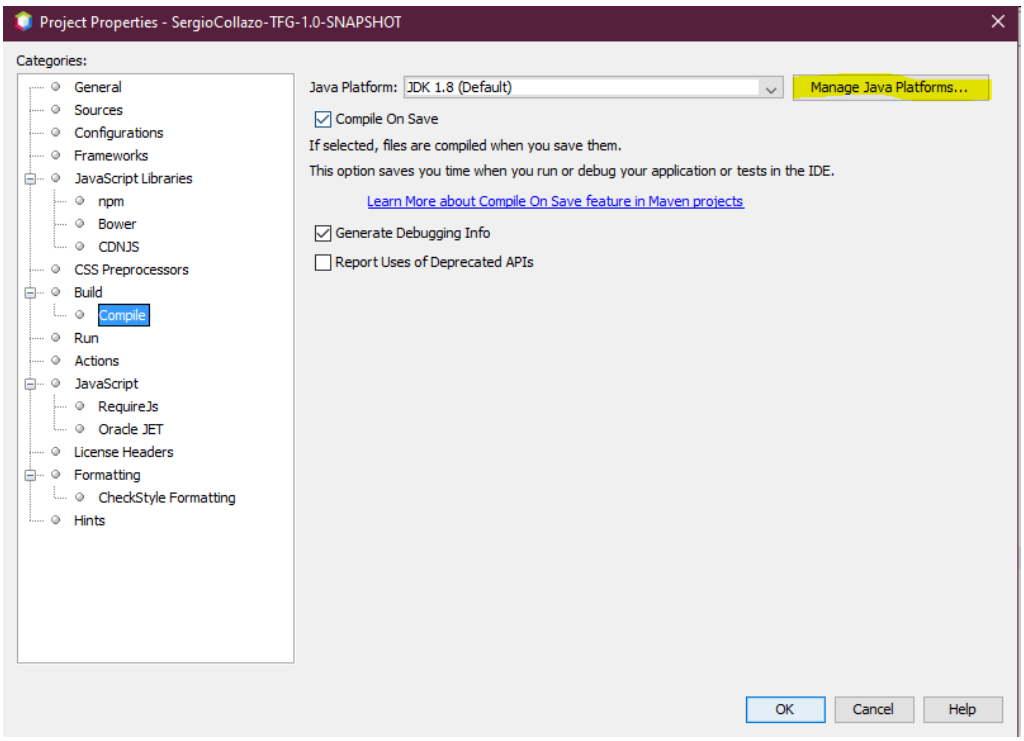


Ilustración 40: Ventana Properties/Build/Compile

Hacer clic en ‘Add Platform’ y a continuación, seleccionar la opción ‘Java Standard Edition’, pulsar ‘Next’, después navegar hasta la carpeta ‘Java’ y seleccionar en ella el jdk adecuado, pulsar ‘Next’ y por último ‘Finish’.

Este último proceso se puede ver en las ilustraciones Ilustración 41, Ilustración 42, Ilustración 43 e Ilustración 44.

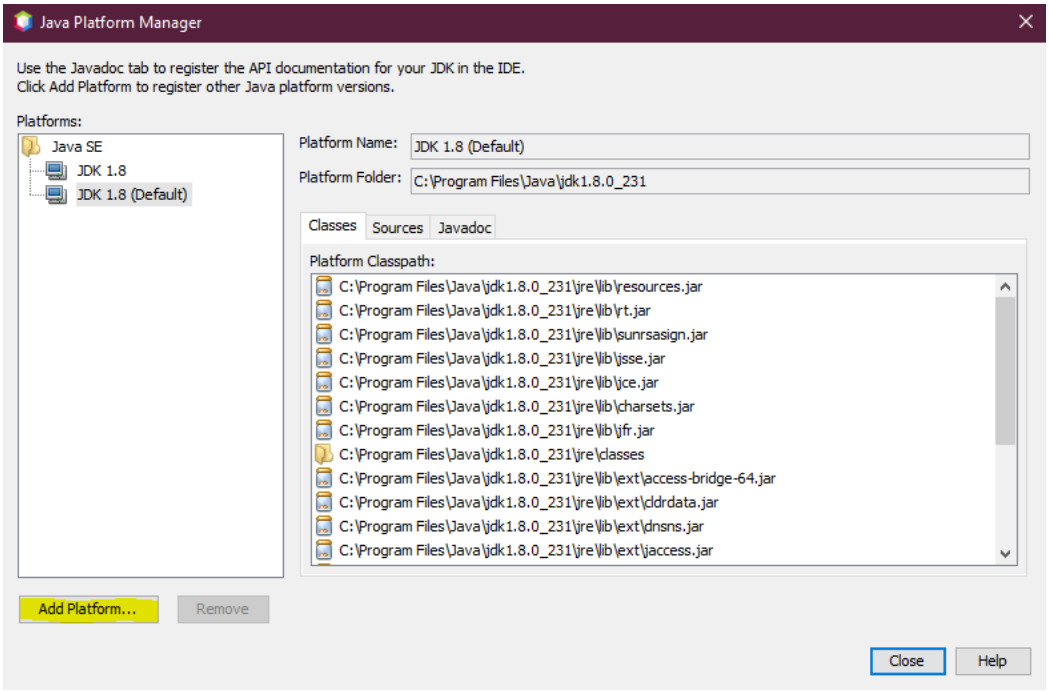


Ilustración 41: Ventana Java Platform Manager

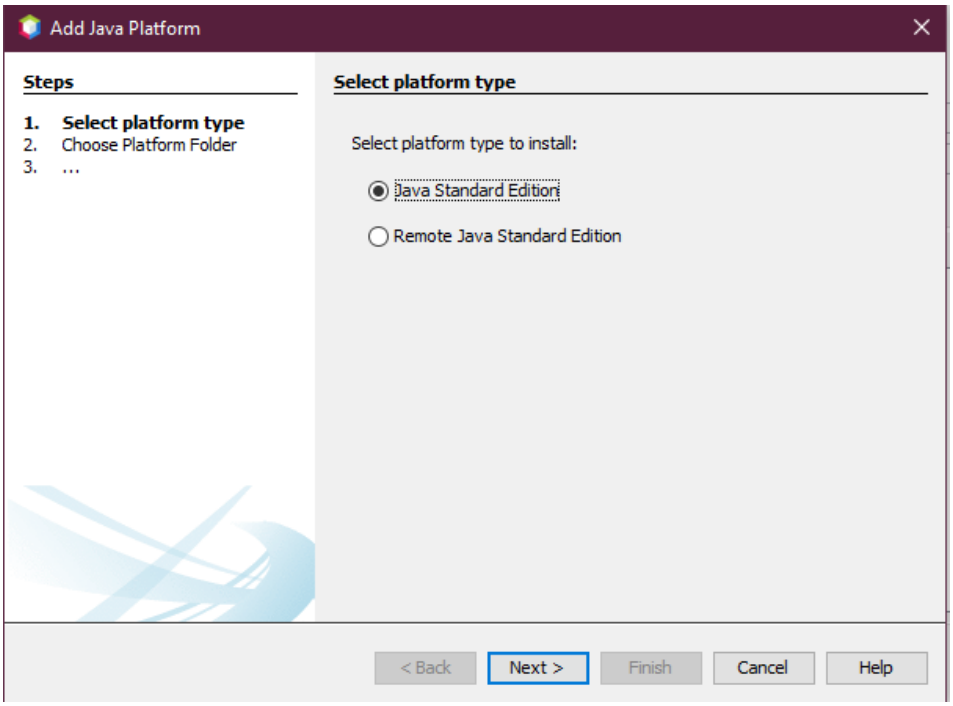


Ilustración 42: Ventana Add Java Platform/Select

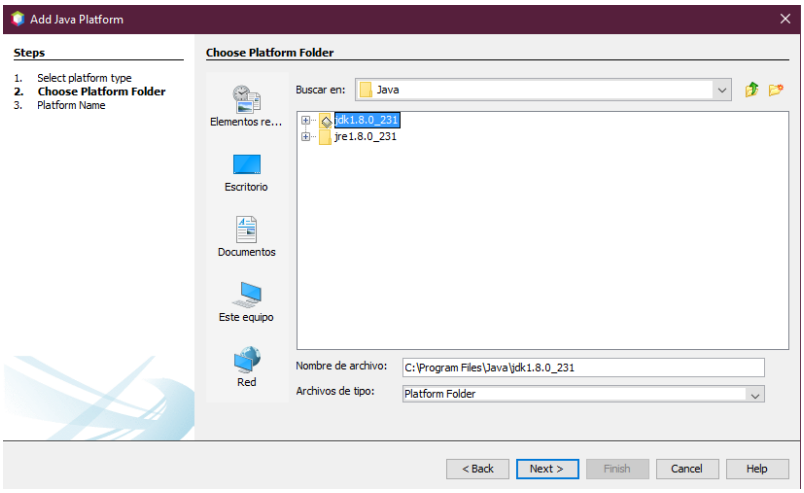


Ilustración 43: Ventana Add Java Platform/Choose

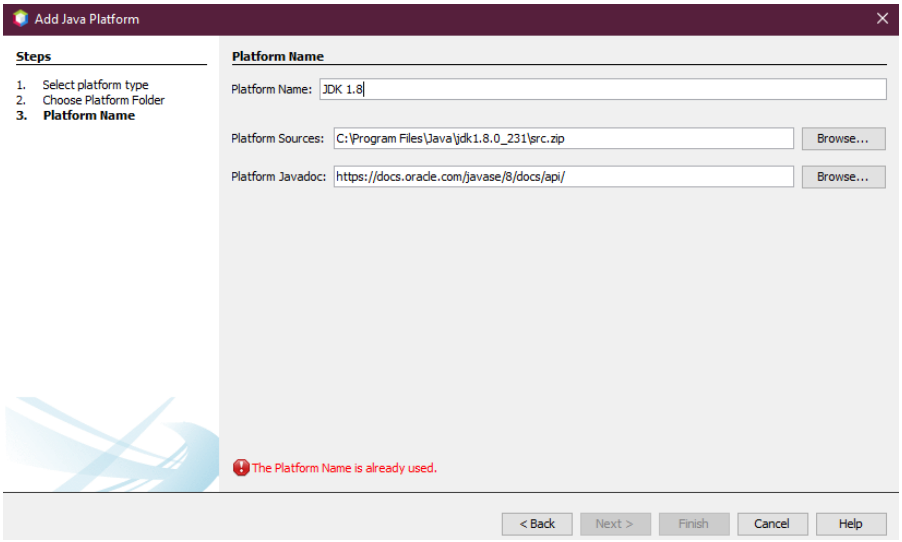


Ilustración 44: Ventana Add Java Platform/Name

En la barra de herramientas del entorno seleccionar 'Tools', 'Servers', hacer clic en 'Add Server', seleccionar 'GlassFish Server' y hacer clic en 'Next' como se ve en la Ilustración 45.

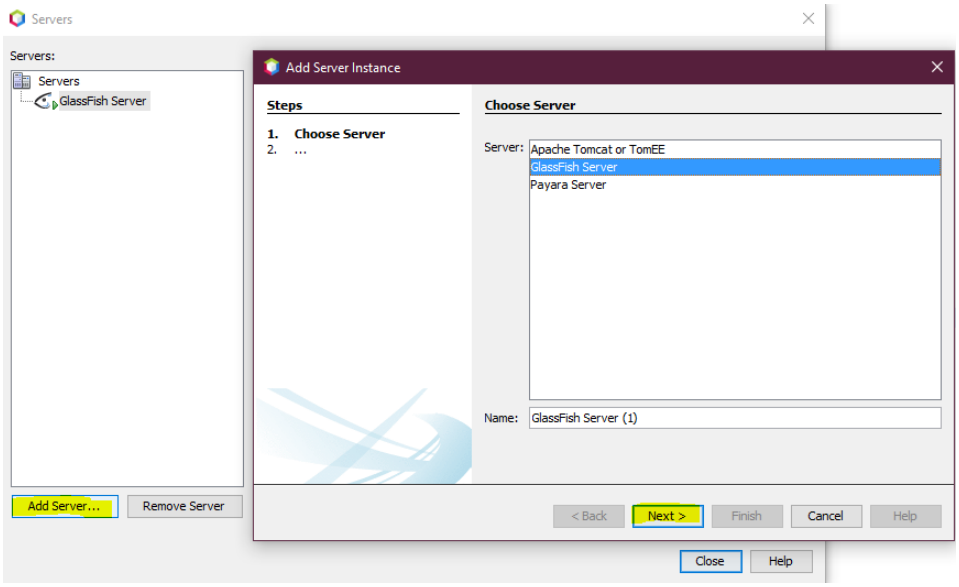


Ilustración 45: Ventana Servers

Aceptar la licencia y seleccionar 'Download Now' como se ve en la Ilustración 46, después proseguir con la configuración dejando todas las opciones por defecto.

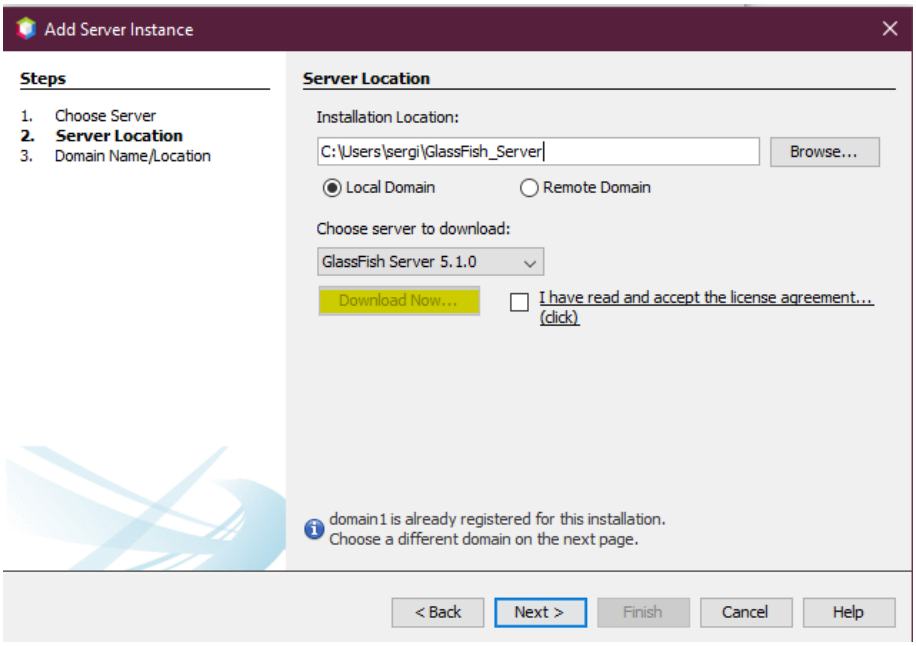


Ilustración 46: Ventana Add Server Instance

Hacer clic derecho del ratón sobre el proyecto y seleccionar 'Properties'. Después seleccionar 'Run' y comprobar que las opciones para el Server son las mismas que las de la Ilustración 47.

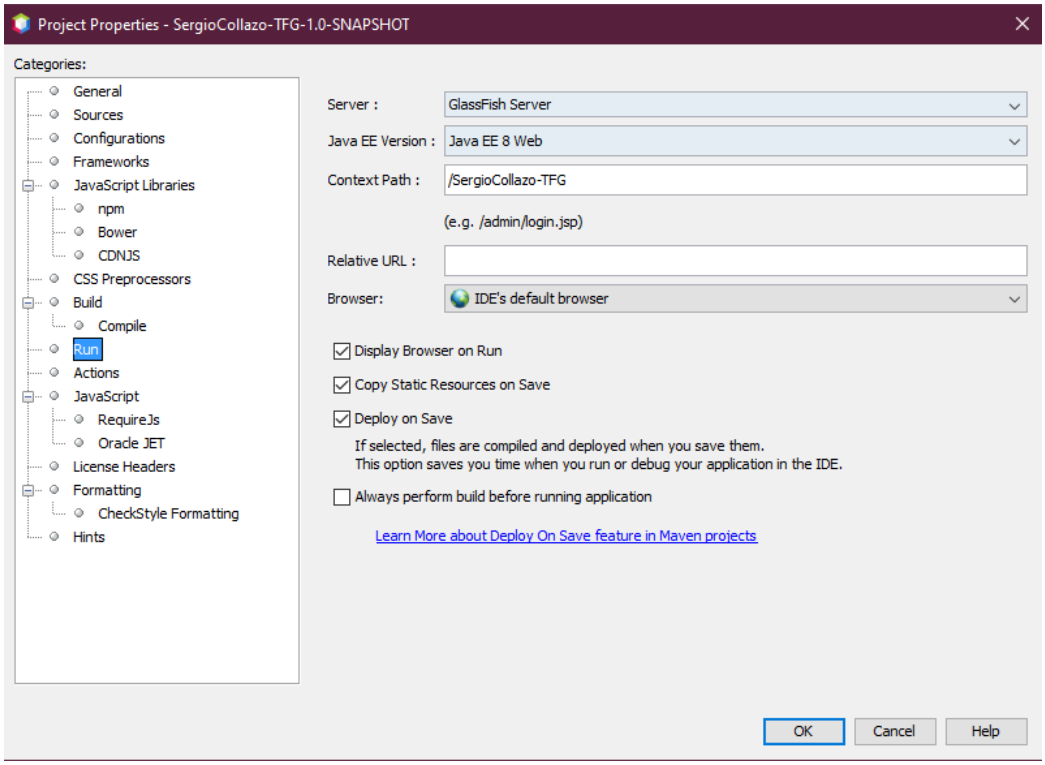


Ilustración 47: Project Properties Run

Para finalizar, desde el apartado Files, hay que editar el archivo ‘ControlConfiguracion.java’ que se encuentra en:

“SergioCollazo-TFG\src\main\java\com\mycompany\sergiocollazo\tfg\controlers\ControlConfiguracion.java “

Crear una carpeta en cualquier dirección con un nombre a elegir, por ejemplo ‘pdfs’.

Como se ve en las ilustraciones Ilustración 48 e Ilustración 49 hay que cambiar las variables ‘dirPDFlatex’, ‘dirAuxPDFlatex’ y ‘dirOutputPDFlatex’, se pueden tener diferentes rutas, pero recomiendo usar la misma para las 3. La ruta debe ser la ruta absoluta de la carpeta creada en el paso anterior, llamada ‘pdfs’.

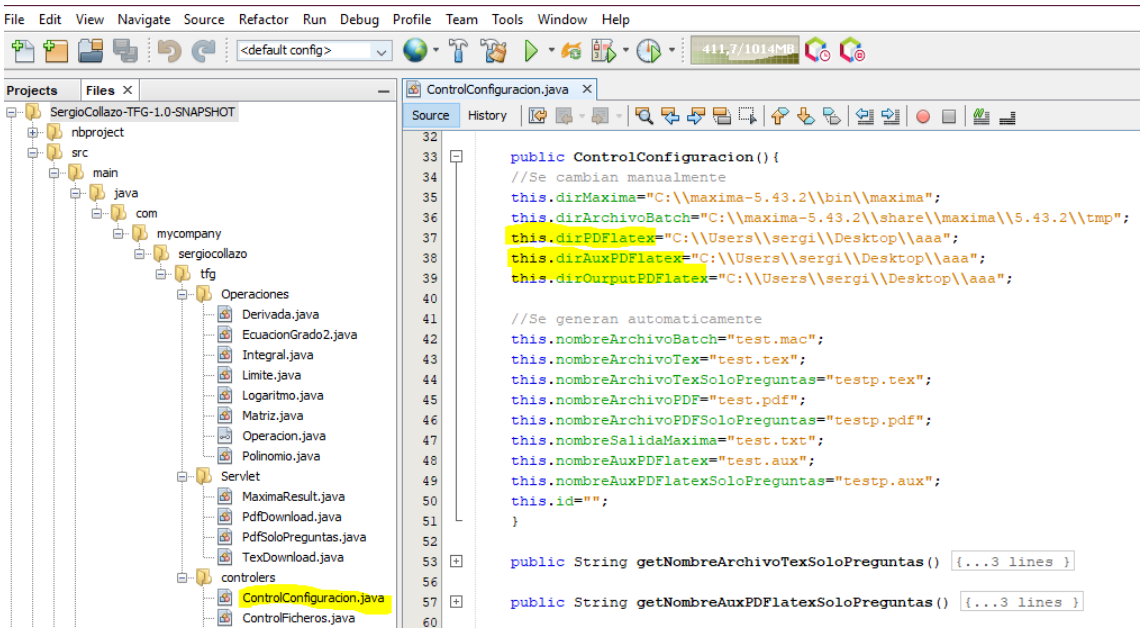


Ilustración 48: Ubicación ControlConfiguración.java

```
public ControlConfiguracion(){
//Se cambian manualmente
this.dirMaxima="C:\\maxima-5.43.2\\bin\\maxima";
this.dirArchivoBatch="C:\\maxima-5.43.2\\share\\maxima\\5.43.2\\tmp";
this.dirPDFlatex="C:\\Users\\sergi\\Desktop\\aaa";
this.dirAuxPDFlatex="C:\\Users\\sergi\\Desktop\\aaa";
this.dirOurputPDFlatex="C:\\Users\\sergi\\Desktop\\aaa";
```

Ilustración 49: Detalle variables a cambiar

Guardar los cambios realizados después de editar este archivo y hacer clic derecho del ratón sobre el proyecto y usar la opción llamada ‘Clean and Build’.

11.2. Ejecución

Una vez instalados todos los recursos y configurado el proyecto y el entorno, ya es posible ejecutar la aplicación web, para ello desde el entorno haciendo clic derecho y seleccionando 'Run' (se recomienda hacer clic en 'Clean and Build' antes de cada ejecución). De esta manera ya se abrirá de manera automática un navegador con la aplicación desplegada y lista para usar.

Hay que tener en cuenta que la primera ejecución del proyecto será la más lenta, ya que NetBeans se encargará de instalar las dependencias que faltan.

En caso de que no se despliegue de forma automática el buscador siempre es posible hacer una búsqueda manual desde el buscador deseado a la dirección: 'http://localhost:8080/SergioCollazo-TFG/'.

11.3. Manual de uso

En este apartado de la memoria vamos a mostrar un ejemplo en el que se generarán 2 ejercicios de derivadas y se descargará el PDF con las soluciones.

Primero en el buscador escribir la ruta: 'http://localhost:8080/SergioCollazo-TFG/'.

Se abrirá una web como se puede ver en la Ilustración 50Ilustración 50, desde ahí seleccionar el menú desplegable de derivadas y elegir por ejemplo la opción combinación.

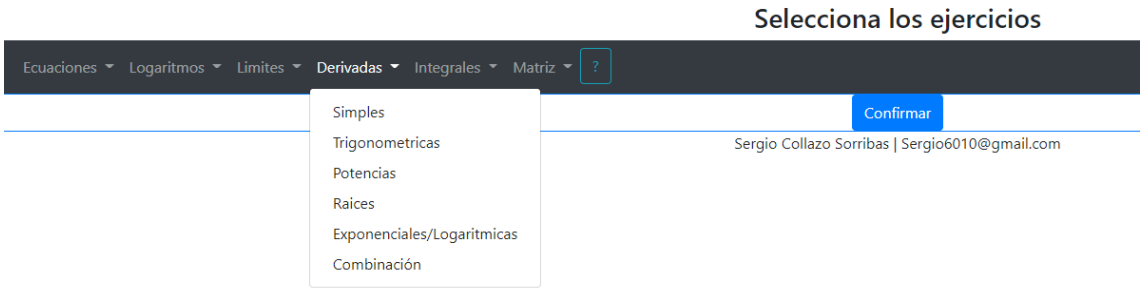


Ilustración 50: Menú Principal

Se desplegará un menú formulario como el que se ve en la Ilustración 51, este menú de combinación de derivadas nos permite seleccionar el número de ejercicios que deseamos en nuestro caso 2, además de los tipos de Derivadas que queremos combinar y el tipo de combinación posible. Una vez seleccionadas las combinaciones podemos pulsar el botón de 'Aleatorio'.

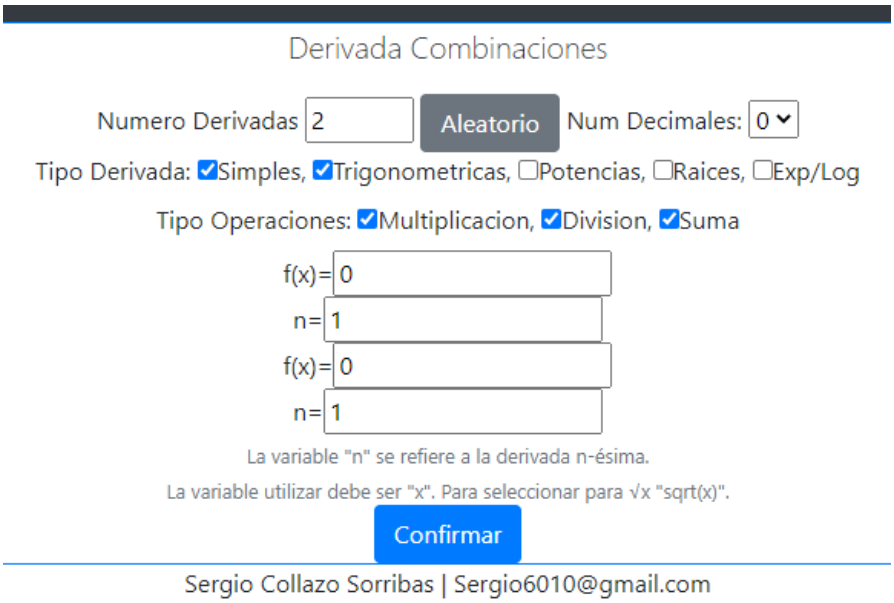


Ilustración 51: Menú Derivada Combinaciones

Como vemos en la Ilustración 52 se han generado dos ejercicios de manera aleatoria que se pueden editar en caso de así desearlo.

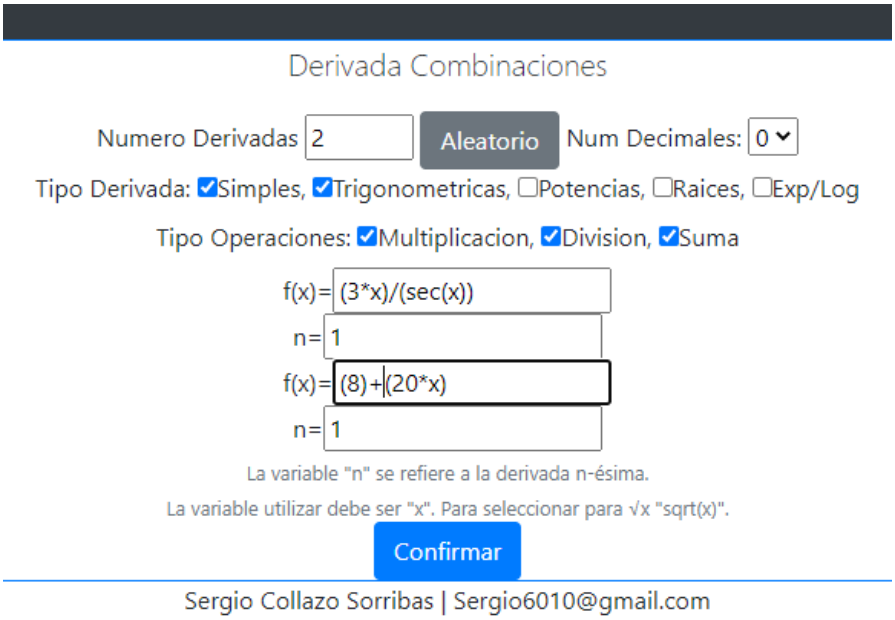


Ilustración 52: Menú Derivada Combinaciones 2

Si pulsamos el botón de 'Confirmar' y la aplicación procesará el formulario entregado y devolverá una pantalla de resultados como se ve en la Ilustración 53, donde tenemos 3 botones diferentes para descargar el archivo que queremos, además de los resultados de los ejercicios ya en pantalla con sus graficas interactivas correspondientes. (En la Ilustración 53 solamente se ven los resultados de 1 ejercicio, haciendo scroll en la pantalla se puede ver el otro ejercicio generado).

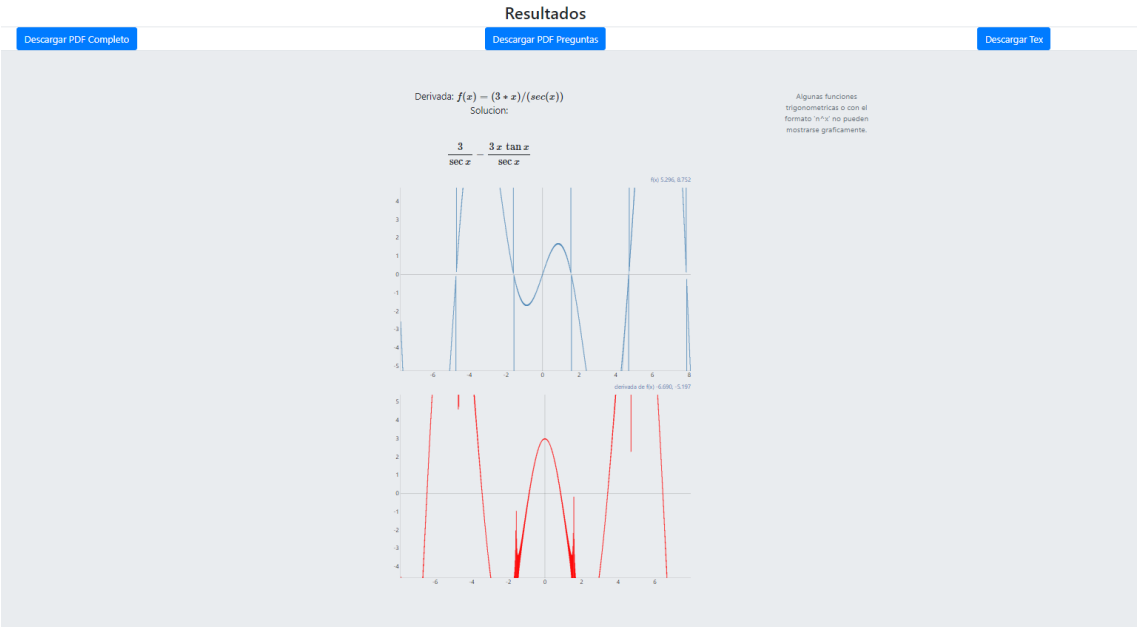
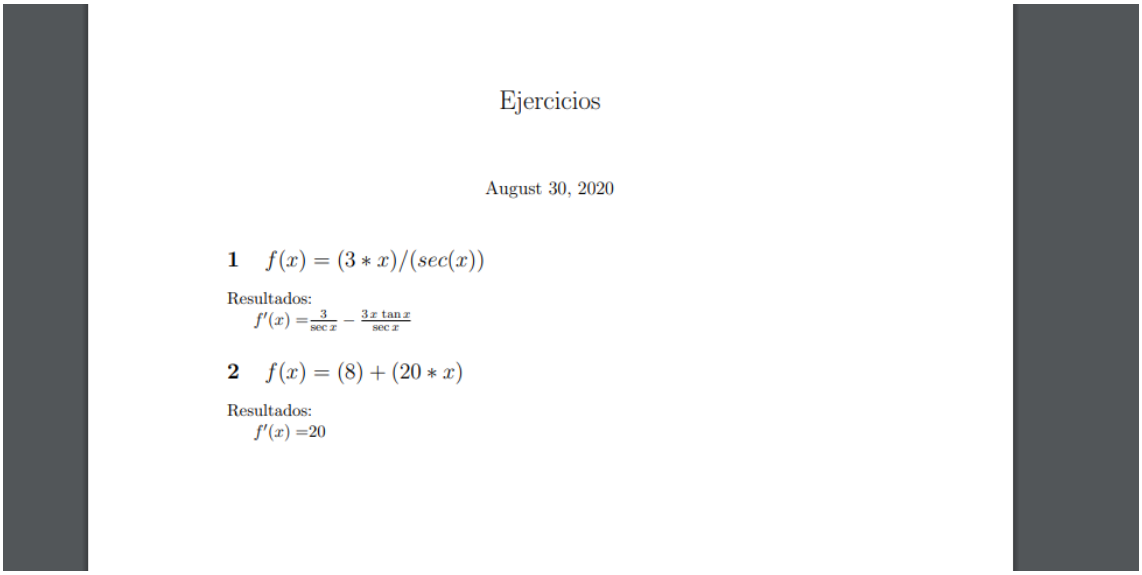


Ilustración 53: Menú resultados

Si se pulsa el botón 'Descargar PDF Completo' se descargará un documento PDF con los ejercicios y resultados ordenados tal y como se ve en la Ilustración 54.



Ejercicios

August 30, 2020

1 $f(x) = (3 * x)/(sec(x))$

Resultados:

$f'(x) = \frac{3}{\sec x} - \frac{3x \tan x}{\sec x}$

2 $f(x) = (8) + (20 * x)$

Resultados:

$f'(x) = 20$

Ilustración 54: Documento PDF descargado

12. Principales aportaciones

En este proyecto se ha desarrollado una aplicación web accesible desde cualquier navegador que ofrece una solución al problema de crear múltiples ejercicios aleatorios de entre varios tipos a un nivel de bachillerato el cual hasta el momento no contaba con soluciones prácticas.

La aplicación ha cumplido con las expectativas y objetivos marcados, es capaz de generar un número ilimitado de ejercicios para el nivel de dificultad escogido, además ofrece la posibilidad de modificar a placer los ejercicios generados y crear nuevos ejercicios de manera manual por parte del usuario con los datos que desee. Por tanto, se ha creado una herramienta muy versátil a la hora de generar y solucionar ejercicios que beneficia tanto a un usuario alumno como a un usuario profesor.

La aplicación también ha sido capaz de generar graficas para los ejercicios que se benefician de esta opción, como una ecuación de segundo grado, que puede ser interesante ver visualmente los puntos de corte con el eje X. También se benefician las derivadas entre otras, cuyas graficas muestran la gráfica de la función a derivar, y su derivada, además de ser graficas interactivas en las que se muestra de manera dinámica para cada punto de del eje X como es la pendiente de la función.

Además, la aplicación ha cumplido con el objetivo de ser intuitiva y fácil de usar, ya que su interfaz es clara y simple.

Los documentos que se pueden descargar una vez terminado el procesamiento de los ejercicios son completos y claros, ha cumplido con los objetivos marcados, los de tipo PDF son de utilidad para usar como boletín de ejercicios o como hoja de soluciones, y los de tipo TeX ofrecen una flexibilidad de uso que no se encuentra en otras aplicaciones actuales.

13. Conclusiones

Realizar este proyecto me ha permitido profundizar en las técnicas y tecnologías que he aprendido a lo largo del grado de Ingeniería Informática. Siendo estas tecnologías todas las relativas a la interfaz de usuario y al lenguaje de programación Java. Además de profundizar en los frameworks Junit, Maven, Java EE y Bootstrap.

Además, me ha dado la oportunidad de adquirir conocimientos de tecnologías y aplicaciones que desconocía como para integrar y utilizar correctamente Maxima, no existían conectores o aplicaciones que juntaran el código Java con esta aplicación y fue un reto buscar alternativas y formas de realizar esta unión de manera correcta, además de aprender mucho sobre el manejo y funcionamiento de la aplicación Maxima en sí misma. Otra aplicación que desconocía era MikTex que ha resultado de gran utilidad para generar los documentos de manera adecuada, ya que requerían una correcta y fácil visualización de expresiones matemáticas.

Esta visualización placentera de expresiones matemáticas ha sido posible gracias al formato TeX, que también desconocía, pero tras estudiarlo y trabajarlo en este proyecto ha resultado tan interesante como útil.

Por otra parte, realizando este proyecto he descubierto otras formas de hacer aplicaciones web con contenido generado de manera dinámica, la tecnología he utilizado para esto ha sido una de las múltiples que me ofrecía el framework JavaEE siendo el Servlet la escogida.

He descubierto varias librerías para integrar con el lenguaje JavaScript que desconocía, siendo una de estas la librería Function Plot, la cual resultó ser muy simple de integrar y configurar. Sin embargo, tiene una gran potencia de cálculo dinámico de gráficos.

A nivel más personal tenía interés en aprender más sobre metodologías agiles, este proyecto me ha dado la oportunidad de hacerlo y adaptar esta metodología a mis preferencias ya que no se cierra a modificaciones en su estructura. En concreto he basado la metodología usada en Scrum que es con la que estaba un poco familiarizado, y aunque he tenido que adaptarla ya que ha sido un proyecto realizado de manera individual creo que ha resultado bueno para mi aprendizaje utilizar este tipo de metodología que apenas he podido utilizar durante la realización del grado.

14. Vías de trabajo futuro

Las posibles ampliaciones a la aplicación desarrollada en el proyecto son varias. Una de ellas bastante clara es la posibilidad de añadir nuevos tipos de ejercicios, cualquier tipo de ejercicio que se pueda resolver utilizando Maxima sería posible, algunos ejemplos son: conjuntos de ecuaciones, vectores, la transformada de Fourier, análisis grafico de una función, etc.

Otra ampliación que puede resultar de utilidad sería la de añadir a los documentos descargables las gráficas generadas para algunos ejercicios, que ahora mismo se puede realizar de manera manual descargando la imagen directamente del HTML e insertándola después, pero no se realiza de manera automática.

Una ampliación dentro de las opciones de generación aleatoria de valores para los ejercicios también puede resultar de utilidad, o añadir más niveles de configuración para los tipos de ejercicios existentes.

Para usuarios más avanzados podría añadirse una interfaz para interactuar con Maxima de manera visual e integrar los resultados que obtengan con la visualización de expresiones matemáticas en TeX. Ya que se pueden realizar problemas o ejercicios más complejos de esta manera, aunque requiere un conocimiento amplio del funcionamiento de Maxima que con el estado actual de la aplicación no es necesario.

Aunque la aplicación web funciona perfectamente en cualquier SO, para mantenerla en un servidor actualmente solo es posible realizarlo si este es un SO Windows 10, por lo tanto, una vía de trabajo futuro puede ser hacer una migración a otros SO para que la aplicación se pueda mantener o lanzar en un servidor correspondiente a un SO Linux o Mac.

15. Índice de Ilustraciones

Ilustración 1: Modelo Cascada. Paulsmith99, Recuperado (2020): https://es.wikipedia.org/wiki/Desarrollo_en_cascada 5
Ilustración 2: Metodología Scrum. Gracia, Richard. Recuperado (2020). https://richardgracia.com/scrum-para-startups/ 6
Ilustración 3:Kanban board. Anderson, David J., Carmichael, A., Essential Kanban Condensed, pag.13 7
Ilustración 4: Esquema Product backlog 8
Ilustración 5: Metodología aplicada al proyecto 8
Ilustración 6: Modelo Cliente-Servidor 16
Ilustración 7: Funcionamiento Servidor 16
Ilustración 8: Arquitectura Servlet Back-end 17
Ilustración 9: Estructura proyecto 17
Ilustración 10: Estructura de Operaciones, Servlets y Controladores 18
Ilustración 11: División de la tarea 9 24
Ilustración 12: Especificación T9.1 24
Ilustración 13: Especificación T9.2 25
Ilustración 14: Especificación T9.3 25
Ilustración 15: Especificación T9.4 25
Ilustración 16: Especificación T9.5 26
Ilustración 17: Especificación T9.6 26
Ilustración 18: División de la tarea 7 27
Ilustración 19: Especificación T7.1 27
Ilustración 20: Especificación T7.2 27
Ilustración 21: División de la tarea 14 29
Ilustración 22: Especificación T14.1 30
Ilustración 23: Especificación T14.2 30
Ilustración 24: Especificación T14.3 30
Ilustración 25: Función ejecutarComandoConsola(). 31
Ilustración 26: Código plantilla TeX en Java 32
Ilustración 27: Función construirTemplateLatex 32
Ilustración 28: Código ejecutar comando MiKTeX 32
Ilustración 29: Función generarID 33
Ilustración 30: Limite Test GetGrafica() 34
Ilustración 31: Matriz Test CrearSentenciaMaxima() 35
Ilustración 32: Test Suit 35
Ilustración 33: Maxima 5.43.2 36
Ilustración 34: Carpeta Maxima 36
Ilustración 35: MiKTeX 37
Ilustración 36: Instalación MiKTeX 37
Ilustración 37: NetBeans 11.2 38
Ilustración 38: Ventana New Project 38
Ilustración 39: Seccion Projects 39
Ilustración 40: Ventana Properties/Build/Compile 39
Ilustración 41: Ventana Java Platform Manager 40
Ilustración 42: Ventana Add Java Platform/Select 40
Ilustración 43: Ventana Add Java Platform/Choose 41
Ilustración 44: Ventana Add Java Platform/Name 41
Ilustración 45: Ventana Servers 41
Ilustración 46: Ventana Add Server Instance 42
Ilustración 47: Project Properties Run 42
Ilustración 48: Ubicación ControlConfiguración.java 43
Ilustración 49: Detalle variables a cambiar 43
Ilustración 50: Menú Principal 44

Ilustración 51: Menú Derivada Combinaciones *45*
Ilustración 52: Menú Derivada Combinaciones 2 ..... *45*
Ilustración 53: Menú resultados *46*
Ilustración 54: Documento PDF descargado *46*

16. Índice de tablas

Tabla 1: Planificación general *9*
Tabla 2: Tiempos estimado y real finales *9*
Tabla 3: Planificación Tareas Sprint 1 ..... *10*
Tabla 4: Scrum diario Sprint1 ..... *10*
Tabla 5: Revisión Sprint 1 ..... *10*
Tabla 6: Planificación Tareas Sprint 2 ..... *11*
Tabla 7: Scrum diario Sprint 2 ..... *11*
Tabla 8: Revisión Sprint 2 ..... *12*
Tabla 9: Planificación Tareas Sprint 3 ..... *12*
Tabla 10: Scrum diario Sprint 3 ..... *12*
Tabla 11: Revisión Sprint 3 ..... *12*
Tabla 12: Planificación Tareas Sprint 4 ..... *13*
Tabla 13: Scrum diario Sprint 4 ..... *13*
Tabla 14: Revisión Sprint 4 ..... *13*
Tabla 15: Planificación Tareas Sprint 5 ..... *14*
Tabla 16 : Scrum diario Sprint 5 ..... *15*
Tabla 17: Revisión Sprint 5 ..... *15*
Tabla 18: Product backlog inicial *22*
Tabla 19: Product backlog subdividido *23*
Tabla 20: Product backlog final *29*

17. Referencias

Anderson, D. J., & Carmichael, A. (2016). *Essential Kanban Condensed* (First ed.). Seattle, Washington: Lean Kanban University.

Apache Software Foundation. (2020). *Apache Maven Project*. Obtenido de https://maven.apache.org/what-is-maven.html

Apache Software Foundation. (2020). *Apache NetBeans*. Obtenido de https://netbeans.apache.org/kb/docs/platform/index.html

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). *Manifesto for Agile Software Development*. Obtenido de http://agilemanifesto.org/

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). *Principles behind the Agile Manifesto*. Obtenido de http://agilemanifesto.org/principles.html

Bostock, M. (2020). *D3 Data-Driven Documents*. Obtenido de https://d3js.org/

Cervone, D., & Sorge, V. (2020). *MathJax*. Obtenido de https://www.mathjax.org/

Gosling, J., Joy, B., Steele, G., Bracha, G., Buckley, A., Smith, D., & Bierman, G. (2020). *The Java Language Specification Java SE 14 Edition.* Obtenido de https://docs.oracle.com/javase/specs/jls/se14/jls14.pdf

Jendrock, E., Cervera-Navarro, R., Evans, I., Haase, K., & Markito, W. (2014). *Oracle.* Obtenido de https://docs.oracle.com/javaee/7/JEETT.pdf

Kniberg, H., & Skarin, M. (2009). *Kanban and Scrum, making the most of both.* United States of America : C4Media.

Leach, P., Mealling, M., & Salz, R. (2005). *A Universally Unique IDentifier (UUID) URN Namespace*. Obtenido de https://tools.ietf.org/html/rfc4122

Luján Mora, S. (2002). *Programación de aplicaciones web: historia, principios básicosy clientes web.* Alicante: Editorial Club Universitario.

Mauricio, P. (2020). *Function Plot*. Obtenido de https://mauriciopoppe.github.io/function-plot/

Maxima. (2020). *Maxima, un sistema de álgebra computacional*. Obtenido de http://maxima.sourceforge.net/es/index.html

Mozilla. (2020). *MDN web docs*. Obtenido de https://developer.mozilla.org/es/docs/Web/JavaScript

Oracle. (2013). *The Java EE 6 Tutorial.* Obtenido de https://docs.oracle.com/javaee/6/tutorial/doc/bnadf.html

Oracle. (2017). *GlassFish Server Open Source Edition.* Obtenido de https://javaee.github.io/glassfish/doc/5.0/release-notes.pdf

Oracle. (2020). *Class SecureRandom*. Obtenido de https://docs.oracle.com/javase/7/docs/api/java/security/SecureRandom.html

Per, B. (2020). Bothner, P. (2020). Per Bothner makes the big time [Correo electrónico]. Obtenido de https://gcc.gnu.org/legacy-ml/java/1999-q2/msg00419.html

Schenk, C. (2020). *MiKTeX*. Obtenido de https://miktex.org/about

Schwaber, K., & Sutherland, J. (2013/2013). *La Guía de Scrum.* (J. L. Soria, J. Francia, A. Salazar Carballo, & M. López, Trads.)

TeamBootstrap. (2020). *Getbootstrap*. Obtenido de https://getbootstrap.com/

TUG. (2020). *TeX Users Group*. Obtenido de http://www.tug.org/