Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Imagen que contiene Logotipo

Descripción generada automáticamente

TFG del Grado en Ingeniería Informática

**Sibling Rewiring 2.0**

Documentación Técnica

Presentado por Rubén Arasti Blanco

en la Universidad de Burgos — 9 de julio de 2024

Tutores: Dr. José Manuel Galán Ordax

y Dra. Virginia Ahedo García

# Índice General

[Índice General 1](#_Toc170407707)

[Índice de Ilustraciones 2](#_Toc170407708)

[Índice de Tablas 3](#_Toc170407709)

[Apéndice A. Plan de Proyecto Software 4](#_Toc170407710)

[A.1 Introducción 4](#_Toc170407711)

[A.2 Planificación temporal 4](#_Toc170407712)

[Pasos iniciales 4](#_Toc170407713)

[Sprint 1 (24/04/2024 – 1/05/2024) 4](#_Toc170407714)

[Sprint 2 (1/05/2024 – 8/05/2024) 5](#_Toc170407715)

[Sprint 3 (8/05/2024 – 15/05/2024) 6](#_Toc170407716)

[Sprint 4 (15/05/2024 – 22/05/2024) 6](#_Toc170407717)

[Sprint 5 (22/05/2024 – 05/06/2024) 7](#_Toc170407718)

[Sprint 6 (05/06/2024 – 19/06/2024) 7](#_Toc170407719)

[Sprint 7 (05/06/2024 – 19/06/2024) 8](#_Toc170407720)

[Sprint 8 (27/06/2024 – 08/07/2024) 9](#_Toc170407721)

[A.3 Estudio de viabilidad 9](#_Toc170407722)

[Viabilidad económica 9](#_Toc170407723)

[Viabilidad legal 9](#_Toc170407724)

[Apéndice B. Especificación de Requisitos 10](#_Toc170407725)

[Apéndice C. Especificación de diseño 11](#_Toc170407726)

[Apéndice D. Documentación técnica de programación 12](#_Toc170407727)

[Apéndice E. Doc umentación de usuario 13](#_Toc170407728)

[Apéndice F. Anexo de sostenibilización curricular 14](#_Toc170407729)

[Bibliografía 15](#_Toc170407730)

# Índice de Ilustraciones

[Ilustración 1: Diagrama de casos de uso 13](#_Toc171095150)

# Índice de Tablas

[Tabla 1: Caso de uso 1 14](#_Toc171095197)

[Tabla 2: Caso de uso 2 14](#_Toc171095198)

[Tabla 3: Caso de uso 3 15](#_Toc171095199)

[Tabla 4: Caso de uso 4 15](#_Toc171095200)

[Tabla 5: Caso de uso 5 16](#_Toc171095201)

# Apéndice A. Plan de Proyecto Software

## A.1 Introducción

El plan se divide en dos secciones: planificación temporal y estudio de viabilidad económica y legal.

## A.2 Planificación temporal

Con el objetivo de asegurar una ejecución ordenada y coherente de las diversas fases y tareas del proyecto, se ha decidido adoptar SCRUM como metodología ágil de trabajo basada en sprints. En este contexto, el proyecto se ha estructurado en sprints de aproximadamente una semana de duración cada uno.

### Pasos iniciales

Durante los primeros meses del año, se avanzó muy lentamente en el proyecto y no se utilizó una metodología de trabajo. No obstante, se realizaron tareas relevantes para el desarrollo del proyecto que serán mencionadas en este apartado.

En la primera reunión se habló de los aspectos más relevantes acerca del proyecto. Estos son la importancia de desplegar la aplicación web, las metodologías que debía usar y la elección de las herramientas para llevar a cabo el proyecto.

En la segunda reunión, se informó a los tutores de qué herramientas iba a utilizar.

Tras varios intentos de ejecutar la anterior aplicación sin éxito, se acordó otra reunión en la que se proporcionó una máquina virtual de la anterior aplicación. Gracias a esto, se pudieron entender algunos pasos que no se especificaban en el anterior trabajo sobre las operaciones necesarias con MySQL y se consiguió ejecutar la anterior aplicación.

### Sprint 1 (24/04/2024 – 1/05/2024)

En este sprint se han realizado tareas iniciales de investigación e instalación de algunos programas.

Las tareas programadas fueron las siguientes:

#1 Instalar Zotero.

#2 Entender la anterior aplicación. Esta tarea se dividía en leer la documentación de la anterior aplicación, entender el código de la anterior aplicación y documentar el problema a resolver.

#3 Investigación sobre los algoritmos genéticos multiobjetivo. Esta tarea se dividía en leer un artículo que me proporcionaron los tutores y documentar acerca del algoritmo.

La instalación de Zotero, su extensión para Chrome y un plugin para MS Word llevó aproximadamente media hora. Sumando otra media hora para entender su funcionamiento básico, esta tarea se completó en un total de 1 hora.

La lectura de la documentación de la anterior aplicación tomó 3 horas.

La comprensión y documentación del código de la anterior aplicación requirió 30 horas. En esta tarea, no sólo documenté el problema a resolver sino también los archivos principales del código y la resolución del problema.

No se logró iniciar la última tarea, por lo que se trasladó al siguiente sprint.

Retrospectiva del sprint. Se concluye que se sobreestimó el tiempo necesario para comprender el trabajo anterior. Además, no se registraron las estimaciones de tiempo ni los puntos de historia de cada tarea

### Sprint 2 (1/05/2024 – 8/05/2024)

Este sprint se ha centrado en crear una primera implementación del algoritmo genético.

Las tareas programadas fueron las siguientes:

#5 Documentar el sprint 1. Se estimó un tiempo de 1 hora y se terminó en 45 minutos.

#3 Investigación sobre los algoritmos genéticos multiobjetivo. Se dividió en 2 subtareas: leer un artículo acerca de los algoritmos genéticos multiobjetivo y documentar acerca de este. Se estimó un tiempo de 12 horas en total. La lectura detallada del artículo tomó 6 horas. No se logró iniciar la documentación, por lo que esta tarea se trasladó al siguiente sprint.

#6 Primera implementación del algoritmo genético. Se dividió en 2 subtareas: implementar el código y crear la interfaz en la web. Se estimó un tiempo de 25 horas en total. Completar el código llevó 15 horas y crear la interfaz 3 horas. No obstante, tras una reunión con los tutores, se observó que el algoritmo implementado presentaba numerosos fallos, por lo que se decidió implementar uno nuevo para la siguiente semana. Aun así, el esqueleto principal del código está montado y la representación del genotipo es bueno, por lo que se puede reutilizar para la siguiente implementación.

#4 Corregir la función descargar archivo. Se estimó un tiempo de 3 horas. Esta tarea no se empezó por falta de tiempo invertido. Pasa al siguiente sprint.

Retrospectiva del sprint. Se sobreestimó el tiempo requerido para las tareas, ya que se completaron en menos tiempo del previsto. No se invirtió el tiempo necesario en el proyecto, lo que se reflejó en la cantidad de tareas sin completar.

### Sprint 3 (8/05/2024 – 15/05/2024)

Este sprint se ha centrado en implementar una segunda versión del algoritmo genético.

Las tareas programadas fueron las siguientes:

#7 Documentar el sprint 2. Se estimó un tiempo de 30 minutos y se terminó en 40 minutos.

#3 Documentar acerca de los algoritmos multiobjetivo. Se estimó un tiempo de 6 horas. Tomó 3 horas documentar el frente de Pareto y la definición del problema a resolver.

#8 Segunda implementación al algoritmo genético. Se debía mostrar en una gráfica el frente de Pareto e implementar una interfaz para la web que permitiese escoger entre las distintas soluciones del frente. Se estimó un tiempo de 18 horas y se terminó en 20 horas.

#9 Implementar todos los fitness encontrados a la gráfica. Se estimó un tiempo de 1 hora y se completó en 50 minutos.

También se programaron para este sprint la función de creación de un archivo final y la de descarga del archivo pero no dio tiempo a empezarlas.

Retrospectiva del sprint. Los problemas ocasionados por el funcionamiento de la librería DEAP han retrasado otras tareas menos prioritarias.

### Sprint 4 (15/05/2024 – 22/05/2024)

Este sprint se ha centrado en migrar el algoritmo genético de la librería deap a pymoo.

Las tareas programadas fueron las siguientes:

#12 Documentar el sprint 3. Se estimó un tiempo de 30 minutos y se completó en 40 minutos.

#13 Cambio de biblioteca de deap a pymoo. Se estimo un tiempo de 6 horas y se completó en 10 horas. No se logró replicar completamente el funcionamiento del algoritmo con la biblioteca DEAP.

También se programaron para este sprint la función de creación de un archivo final y la de descarga del archivo pero no se llegaron a comenzar.

Retrospectiva del sprint. Se invirtió poco tiempo en el proyecto durante este sprint, lo cual se reflejó en la cantidad de tareas completadas. Las tareas necesarias para el archivo final y para la descarga no son tan prioritarias como el buen funcionamiento del algoritmo, pero no lo he previsto al principio del sprint y he tenido que crear tareas a mitad de este.

### Sprint 5 (22/05/2024 – 05/06/2024)

Este sprint se ha centrado en realizar correcciones sobre el algoritmo genético y añadir la funcionalidad de descarga de varios archivos a partir de una solución.

Las tareas programadas fueron las siguientes:

#14 Documentar el sprint 4. Se estimó un tiempo de 30 minutos y se completó en 10 minutos.

#16 Selección de parámetros para el algoritmo genético. Se estimó un tiempo de 20 horas y se completó en 15 horas. En esta tarea no sólo se seleccionaron los parámetros por defecto del algoritmo genético, sino que también se cambió el algoritmo para que no devolviese la misma solución siempre.

#11 Implementar una función para crear el archivo final. Se estimó un tiempo de 2 horas y se completó en 5 horas. Surgió un problema con la representación en imagen de NetworkX y los auto enlaces.

#18 Implementar descargas para el algoritmo genético. Se estimó un tiempo de 2 horas y se completó en el tiempo previsto. Se codificaron los archivos para su descarga a través de la página web.

También se programaron las tareas de documentar el algoritmo genético y la creación de test pero no llegué a empezarlas.

Retrospectiva del sprint. Este sprint se ha extendido una semana más porque al acabar la primera semana no se realizó ninguna tarea. Los nombres dados a las tareas no han sido del todo adecuados y eso ha provocado confusión sobre cómo abordarlas, retrasando su inicio.

### Sprint 6 (05/06/2024 – 19/06/2024)

Este sprint se centró en mejorar la solución obtenida y los archivos resultado.

Las tareas programadas fueron las siguientes:

#Documentar sprint 5. Se estimó un tiempo de 1 hora y se completó en 30 minutos.

#23 Añadir la decisión de parámetros para el usuario. Se estimó un tiempo de 1 hora y se completó en 4 horas.

#24 Bug creación red aleatoria. Se estimó un tiempo de 1 hora y 30 minutos y se completó en 2 horas.

#20 Mejorar descargas para el algoritmo genético. Se estimó un tiempo de 7 horas y se invirtieron 4 horas pero no se completó. Se realizaron algunas mejoras como colorear los compontes en la imagen del grafo.

#21 Heurística para soluciones no factibles. Se estimó un tiempo de 5 horas y se completó en 11 horas. Además de crear la heurística para cambiar una solución no factible a una factible, se corrigió un problema en la creación de la solución. Esto hizo que llevara más tiempo del esperado.

También se programaron tareas de documentación, corrección de bugs e implementación de tests, pero no se llegaron a empezar.

Retrospectiva del sprint. En la mayoría de los sprints realizados hasta el momento, se han dejado tareas sin completar debido a una planificación inadecuada. Para mejorar este aspecto, es fundamental priorizar y programar las tareas más importantes, asegurándose de que puedan ser completadas dentro del período del sprint. Las tareas secundarias deben mantenerse en el producto backlog. Una vez que las tareas principales se hayan completado, las tareas secundarias pueden añadirse al sprint actual.

### Sprint 7 (05/06/2024 – 19/06/2024)

Este sprint se centró sobre todo en la documentación del proyecto y en acabar el desarrollo de la aplicación.

Las tareas programadas fueron las siguientes:

#25 Documentar sprint 6. Se estimó un tiempo de 30 minutos y se completó en 45 minutos.

#26 Documentar introducción. Se estimó un tiempo de 3 horas y se completó en 6 horas.

#22 Documentar objetivos principales. Se estimó un tiempo de 3 horas y se completó en 3 horas.

#15 Documentación del algoritmo genético. Se estimó un tiempo de 4 horas y se completó en 40 horas aproximadamente. Esta tarea ha sido considerada como la documentación de los conceptos teóricos y por eso el error entre el tiempo estimado e invertido.

#20 Mejorar descargas para el algoritmo genético. Se estimó un tiempo de 3 horas y se completó en 12 horas. Se añadieron las siguientes mejoras, la leyenda en la imagen del grafo, la asignación de nombres a las soluciones y su corrección cuando son modificadas y la descarga de todos los archivos.

#27 Arreglar un bug con los pares de hermanos en la creación de la red aleatoria. Se estimó un tiempo de 1 hora y se completó en 8 horas. En esta tarea también se crearon archivos de ejemplo nuevos y se añadieron las revisiones a la memoria.

Retrospectiva del sprint. La tarea de documentar los objetivos teóricos fue mal definida y desglosada. Es importante que ninguna tarea individual exceda un total de 24 horas. Además, se observó una planificación deficiente en las tareas de diseño, ya que se introdujeron nuevas funcionalidades no planificadas durante la ejecución de las tareas, lo que complicó su gestión y desarrollo adecuado.

### Sprint 8 (27/06/2024 – 08/07/2024)

El último sprint, se centró sobre todo en tareas de documentación, desplegar la aplicación web y la revisión de la calidad del código.

Las tareas programadas fueron las siguientes:

#29 Documentar sprint 7. Se estimó un tiempo de 1 hora y se completó en 1 hora.

#30 Desplegar la web. Se estimó un tiempo de 24 horas y se completó en 30 horas aproximadamente. Durante esta tarea se generaron archivos de configuración necesarios para la creación de un Docker, pero el tiempo invertido en estos se añadirá a la tarea #36.

#38 Revisión de código mediante SonarCloud. Se estimó un tiempo de 24 horas y se completó en 12 horas. Durante esta tarea se invirtieron 5 horas en migrar el algoritmo a la librería pymoo para corregir un bug encontrado con la librería DEAP.

#36 Creación de máquina virtual y Docker. Se estimó un tiempo de 24 horas y se completó en 15 horas. La creación del Docker tomo 10 horas teniendo en cuenta la corrección de sus fallos. Y la creación de la máquina virtual tomó 5 horas.

#33 Documentar apéndice B especificación de requisitos. Se estimó un tiempo de 12 horas y se completó en 8 horas.

#35 Documentar apéndice F anexo de sostenibilización curricular. Se estimó un tiempo de 5 horas y tomó 3 horas y media completarlo.

#40 Documentar técnicas y herramientas. Se estimó un tiempo de 8 horas y se completó en 6 horas.

#41 Documentar aspectos relevantes en el desarrollo del proyecto. Se estimó un tiempo de 8 horas y

#42 Documentar trabajos relacionados. Se estimó un tiempo de 5 horas y se completó en 2 horas.

#39 Documentar conclusiones y líneas de trabajo futuras. Se estimó un tiempo de 8 horas y se completó en

#34 Documentar apéndice C especificación de diseño. Se estimó un tiempo de 24 horas y se completó en

#31 Documentar apéndice D documentación técnica de programación. Se estimó un tiempo de 12 horas y se completó en

#32 Documentar apéndice A plan de proyecto software. Se estimó un tiempo de 8 horas y se completó en

#28 Documentar resumen. Se estimó un tiempo de 1 hora y se completó en

## A.3 Estudio de viabilidad

### Viabilidad económica

Heroku despliegue 7 dólares al mes

### Viabilidad legal

# Apéndice B. Especificación de Requisitos

## B.1. Introducción

Este apéndice se enfoca en la definición detallada de los requisitos de la aplicación, los cuales guiarán su diseño y desarrollo para que el producto final cumpla con los objetivos establecidos.

Debido a que este proyecto es una continuación de un proyecto anterior, se ampliará el diagrama de casos de uso del anterior proyecto, pero no se explicarán dichos casos ni se tendrán en cuenta los requisitos que no hayan sido añadidos en este proyecto, utilizando códigos de referencia nuevos.

## B.2. Objetivos generales

* Desarrollar una aplicación web para optimizar la organización de los alumnos en un colegio y reducir el riesgo de contagio.
* Implementar un algoritmo genético como método de optimización multiobjetivo para aportar diversidad de soluciones al problema.
* Mostrar gráficamente la evaluación de distintas soluciones exploradas y resaltar el frente de Pareto.
* Aportar una solución intuitiva para el usuario y con información relevante para la toma de decisiones.

## B.3. Catálogo de requisitos

En este apartado se mostrarán los requisitos funcionales y no funcionales.

### Requisitos funcionales

**RF-1. Selección de algoritmo**. La aplicación debe permitir al usuario seleccionar entre el algoritmo genético y el recocido simulado.

**RF-2. Configuración de Parámetros del Algoritmo Genético**. La aplicación debe permitir al usuario configurar los parámetros del algoritmo genético , como el número de generaciones, el tamaño de la población, la probabilidad de cruce y mutación, y el operador de cruce. Además, debe ofrecer la opción de utilizar parámetros por defecto predefinidos.

**RF-3. Mostrar soluciones**. La aplicación debe mostrar las soluciones generadas por el algoritmo genético.

**RF-3.1 Visualizar soluciones exploradas**. La aplicación debe mostrar tres gráficas con la evaluación de las soluciones exploradas, resaltando las soluciones óptimas en otro color para facilitar su identificación.

**RF-3.2 Mostrar botones de descarga**. La aplicación debe mostrar botones de descarga para cada solución en una lista.

**RF-4. Creación de archivos solución**. La aplicación debe crear y gestionar archivos para las soluciones generadas.

**RF-4.1 Corrección para soluciones no factibles**. La aplicación debe convertir las soluciones no factibles en soluciones factibles antes de su almacenamiento o descarga.

**RF-4.2 Creación de nombre para la solución**. La aplicación debe asignar un nombre único y descriptivo a cada archivo solución.

**RF-4.3 Creación de los archivos**. La aplicación debe generar los archivos correspondientes a cada solución.

**RF-5. Descarga de los archivos solución**. La aplicación debe permitir al usuario descargar los archivos de solución generados.

**RF-5.1 Descargar una sola solución**. La aplicación debe permitir al usuario descargar una sola solución específica en un formato adecuado.

**RF-5.2 Descargar todas las soluciones**. La aplicación debe permitir al usuario descargar todas las soluciones generadas en un único archivo comprimido.

### Requisitos no funcionales

**RNF-1. Accesibilidad**. Las soluciones de la aplicación deben estar diseñadas de manera que sean comprensibles y fáciles de usar para los encargados de administración del colegio.

**RNF-2. Mantenibilidad**. El código de la aplicación debe estar estructurado de forma modular y desacoplada, permitiendo que las modificaciones, mejoras y correcciones puedan realizarse de manera eficiente y con un mínimo de impacto en el resto del sistema. El código también debe tener una correcta documentación interna.

**RNF-3. Escalabilidad**. La arquitectura de la aplicación debe permitir el manejo de una mayor carga de trabajo sin una reestructuración significativa, garantizando que el sistema pueda evolucionar conforme a las necesidades futuras.

**RNF-4. Usabilidad**. La aplicación debe proporcionar una experiencia de usuario intuitiva, facilitando la navegación y el uso de sus funcionalidades mediante una interfaz clara y amigable. Además, la interfaz debe ser adaptable y responsive, garantizando una visualización óptima en dispositivos con distintos tamaños de pantalla.

**RNF-5. Estabilidad**. La aplicación debe ser capaz de ejecutar y gestionar múltiples hilos de procesamiento de manera continua durante un período máximo de 24 horas. Esto es esencial para que el algoritmo pueda realizar una exploración más amplia y profunda de las soluciones.

## B.4. Especificación de requisitos

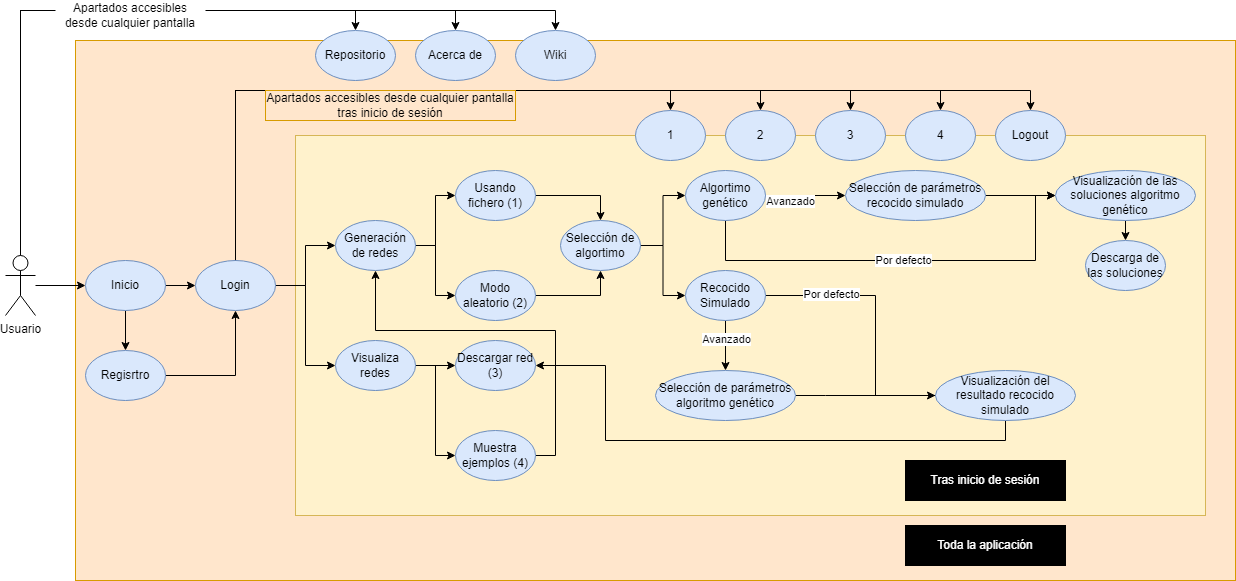
 En este apartado se definirán los casos de uso junto al diagrama de casos de uso. Debido a que sólo hay un tipo de usuario, el actor en todos los casos será el usuario.

Ilustración 1: Diagrama de casos de uso

Tabla 1: Caso de uso 1

|  |  |  |
| --- | --- | --- |
| CU-1 | Seleccionar Algoritmo | |
| Requisitos asociados | **RF-1** | |
| Descripción | Permitir al usuario elegir entre el algoritmo genético y el recocido simulado como método de Optimización. | |
| Precondición | Se deben haber introducido los datos de entrada del problema | |
| Acciones | Paso | Acción |
| 1 | El usuario accede a la interfaz de selección de algoritmos. |
| 2 | El usuario debe elegir una opción entre algoritmo genético y recocido simulado |
| 3 | La aplicación registra la selección del usuario. |
| Postcondición | Se redirige a la pantalla de selección de parámetros dependiendo del algoritmo escogido | |
| Excepciones | Ninguna | |

Tabla 2: Caso de uso 2

|  |  |  |
| --- | --- | --- |
| CU-2 | Seleccionar parámetros genéticos | |
| Requisitos asociados | **RF-2** | |
| Descripción | Permitir al usuario elegir los parámetros del algoritmo genético | |
| Precondición | Se debe haber seleccionado el tipo de algoritmo | |
| Acciones | Paso | Acción |
| 1 | Introducir el valor para el número de generaciones |
| 2 | Introducir el valor para el tamaño de la población |
| 3 | Introducir el valor para la probabilidad de cruce |
| 4 | Introducir el valor para la probabilidad de mutación |
| 5 | Elegir el operador de cruce |
| Postcondición | El algoritmo genético se ejecuta | |
| Excepciones | Número | Excepción |
| 1 | El número de generaciones y el tamaño de la población deben ser enteros positivos |
| 2 | Los valores de probabilidad deben estar entre 0 y 1 |

Tabla 3: Caso de uso 3

|  |  |  |
| --- | --- | --- |
| CU-3 | Mostrar soluciones | |
| Requisitos asociados | **RF-3** | |
| Descripción | Se muestran 3 gráficas de las soluciones exploradas al usuario resaltando el frente de Pareto. Debajo de las gráficas se muestran las opciones de descarga para cada solución. | |
| Precondición | La ejecución del algoritmo genético debe haber finalizado. | |
| Acciones | Paso | Acción |
| 1 | Se muestran las gráficas |
| 2 | Se muestran los botones de descarga para las soluciones |
| Postcondición | Ninguna | |
| Excepciones | Ninguna | |

Tabla 4: Caso de uso 4

|  |  |  |
| --- | --- | --- |
| CU-4 | Descargar solución | |
| Requisitos asociados | **RF-3.2, RF-4, RF-5.1** | |
| Descripción | Permitir al usuario descargar una solución individual | |
| Precondición | La ejecución del algoritmo genético debe haber finalizado. | |
| Acciones | Paso | Acción |
| 1 | Pulsar en el botón de descarga de una solución |
| 2 | La aplicación convierte la solución a factible si es necesario |
| 4 | Se pone nombre a la solución |
| 3 | Los archivos solución son creados y comprimidos |
| Postcondición | Los archivos solución son enviados al usuario | |
| Excepciones | Número | Excepción |
| 1 | El número de iteraciones para convertir la solución a factible supera el número de clases |

Tabla 5: Caso de uso 5

|  |  |  |
| --- | --- | --- |
| CU-5 | Descargar todas las soluciones | |
| Requisitos asociados | **RF-3.2, RF-4, RF-5** | |
| Descripción | Permitir al usuario descargar todas las soluciones | |
| Precondición | La ejecución del algoritmo genético debe haber finalizado. | |
| Acciones | Paso | Acción |
| 1 | Pulsar en el botón de descargar todas las soluciones |
| 2 | La aplicación convierte la siguiente solución a factible si es necesario |
| 3 | Se pone nombre a la solución actual |
| 4 | Los archivos de la solución actual son creados y guardados en una carpeta |
| 5 | Se repite desde la acción 2 con todas las soluciones y se comprimen las carpetas |
| Postcondición | El conjunto de soluciones es enviado al usuario | |
| Excepciones | Número | Excepción |
| 1 | El número de iteraciones para convertir una solución a factible supera el número de clases |

# Apéndice C. Especificación de diseño

## C.1. Introducción

## C.2. Diseño de datos

## C.3. Diseño procedimental

## C.4. Diseño arquitectónico

Patrones de diseño, patrón modular

# Apéndice D. Documentación técnica de programación

## D.1. Introducción

## D.2. Estructura de directorios

## D.3. Manual del programador

## D.4. Compilación, instalación y ejecución del proyecto

## D.5. Pruebas del sistema

Las pruebas del sistema han sido manuales y se han ejecutado durante el desarrollo.

# Apéndice E. Documentación de usuario

## E.1. Introducción

## E.2. Requisitos de usuarios

## E.3. Instalación

## E.4. Manual del usuario

# Apéndice F. Anexo de sostenibilización curricular

## F.1 Introducción

La Agenda 2030 de las Naciones Unidas establece 17 Objetivos de Desarrollo Sostenible (ODS) que buscan abordar los principales desafíos globales, incluyendo la salud y la educación. [1] Este proyecto, centrado en la organización de los alumnos para reducir el riesgo de contagio de enfermedades en entornos escolares, contribuye directamente a los ODS relacionados con la salud y el bienestar (ODS 3) y la educación de calidad (ODS 4).

## F.2 Objetivo número 3: Salud y Bienestar

La organización de los alumnos en el colegio se realiza de manera estratégica para minimizar el riesgo de contagio de enfermedades infecciosas, cumpliendo así con el ODS 3 de la Agenda 2030, que se centra en garantizar una vida sana y promover el bienestar para todos en todas las edades. Este enfoque no solo se limita a una mera disposición física, sino que integra análisis avanzados de datos para una gestión proactiva y eficaz del riesgo.

En caso de un brote de enfermedad, las soluciones propuestas por la aplicación permiten identificar rápidamente las clases que están interconectadas y que, por lo tanto, pueden representar un riesgo de transmisión. Esto facilita la implementación de medidas de control específicas, tales como el confinamiento de ciertas clases o grupos de alumnos, evitando la necesidad de cerrar todo el colegio y permitiendo que el mayor número posible de estudiantes continúe su educación en un entorno seguro. Este enfoque selectivo y dirigido contribuye a la prevención de la propagación de enfermedades, minimizando el impacto en la comunidad educativa y en la salud pública en general.

Además, la aplicación proporciona una evaluación detallada del riesgo de contagio para cada conjunto de clases, facilitando la toma de decisiones informadas sobre la asignación de docentes, especialmente aquellos considerados pacientes de riesgo. Al asignar estos docentes a grupos con menor riesgo de contagio, se protege su salud y se asegura su participación continua en la educación, promoviendo así un entorno educativo seguro y saludable para todos.

Los datos permiten también a la administración educativa la coordinación efectiva con las autoridades de salud pública al proporcionar datos precisos y actualizados sobre la situación sanitaria del colegio. Esto facilita una respuesta más rápida y eficaz ante emergencias, ayudando a contener la propagación de enfermedades y a mantener un entorno seguro para todos los miembros de la comunidad escolar.

## F.2 Objetivo número 4: Educación de calidad

Este proyecto también contribuye significativamente al ODS 4, que busca garantizar una educación inclusiva, equitativa y de calidad. Al gestionar de manera efectiva las conexiones entre clases, la aplicación minimiza las interrupciones en la educación causadas por enfermedades, asegurando que los estudiantes puedan continuar su aprendizaje sin pausas significativas. Esto es especialmente importante en contextos de emergencia sanitaria, como lo fue la pandemia de COVID-19, que ha tenido un impacto profundo en la educación a nivel global.

La pandemia de COVID-19 ha destacado la importancia de la educación presencial, que ofrece un entorno de aprendizaje más enriquecedor que la educación a distancia. Las clases presenciales facilitan la interacción social y el aprendizaje práctico, aspectos fundamentales para el desarrollo académico y personal de los estudiantes. Al permitir que las clases presenciales se lleven a cabo de manera segura, la aplicación contribuye a una educación de calidad, asegurando que los estudiantes puedan beneficiarse de una experiencia educativa completa y enriquecedora.

Al considerar la salud y el bienestar de todos los profesores, incluidos aquellos que son pacientes de riesgo, la aplicación garantiza que todos los docentes puedan seguir participando activamente en la educación. Esto fomenta un entorno educativo inclusivo, donde se valora y protege la diversidad de experiencias y enfoques pedagógicos, ya que cada docente, dependiendo de su edad y experiencia, aporta una perspectiva única y valiosa al proceso educativo.

Además, la aplicación puede servir como un recurso educativo en sí mismo, enseñando a los estudiantes sobre la importancia de la salud pública y la gestión del riesgo, proporcionando conocimientos valiosos que pueden aplicar a lo largo de sus vidas. Al involucrar a los estudiantes en la comprensión de cómo la aplicación organiza las clases para minimizar el riesgo de contagio, se les introduce a conceptos clave en epidemiología y salud pública.

# Bibliografía

[1] M. J. Gamez, «Portada», Desarrollo Sostenible. Accedido: 4 de julio de 2024. [En línea]. Disponible en: https://www.un.org/sustainabledevelopment/es/