

DOCUMENTO DE ANÁLISIS, DISEÑO E IMPLEMENTACIÓN “I-FOLDAMERIC: INTELLIGENT HP FOLDAMER DESIGN”

Pedro Pablo González Pérez*, García Martínez Javier Alexis*, Perez
Perez Abraham* Ponce Rodriguez Milton*

*Departamento de Matemáticas Aplicadas y Sistemas, Universidad
Autónoma Metropolitana, Unidad Cuajimalpa

I-FOLDAMERIC: IHPFD Login Register

I-FOLDAMERIC
INTELLIGENT HP FOLDAMER DESIGN

DISPLAY THE TYPES OF CORRELATED SPACES

Aminoacid or HP Sequence
P4H9P4H9P4H9P4

Space Types Homogeneous Lattice	Space Dimensions 2D Square	Optimization Algorithms Simple
Selection Operator	CrossOver Operator	Mutation Operator

<http://bioinformatics.cua.uam.mx/i-foldameric>

Índice

1. Fase de inicio	2
1.1. Visión	2
1.2. Objetivo	2
1.3. Modelo de contexto	3
1.4. Diagramas de caso de uso	3
1.5. Diagramas de Caso de Uso de Nivel Contextual	3
1.6. Diagramas de Caso de Uso por Macro-funcionalidad	4
1.7. Diagramas de Caso de Uso por Funcionalidad	5
1.8. Especificación de requerimientos	9
1.8.1. Requerimientos funcionales	9
1.8.2. Requerimientos no funcionales	11
1.9. Glosario	12
1.10. Modelo de Dominio	14
1.11. Plan de iteración	14
2. Fase de elaboración	16
2.1. Elaboración de bocetos	16
2.2. Diagramas de actividades	19
2.3. Diagrama de Clases	20
2.4. Diagramas de paquetes	21
2.5. Inicio de la arquitectura de software	25
2.5.1. Arquitectura lógica	25
2.5.2. Glosario para las vistas	26
2.5.3. Glosario para el controlador	27
2.6. Diagramas de secuencia	28
2.6.1. Diagramas de secuencia entre la vista y el controlador	28
2.7. Refinamiento de bocetos	34
3. Fase de construcción	38
3.1. Diseño detallado	38
3.1.1. Identificación de métodos y atributos	38
3.2. Administración de archivos	44
3.3. Modelo de implementación	63
4. Fase de transición	70
4.1. Modelo de pruebas	70

1. Fase de inicio

1.1. Visión

Se busca desarrollar una aplicación web tomando como base el sistema heredado “Bioinformatics Evolution”, para realizar una reingeniería al mismo, el cual se encuentra parcialmente obsoleto por diversas librerías del lenguaje de programación en el que se implementó (*Java*) que dependen fuertemente del sistema operativo (SO), el cual permite a los usuarios desarrollar el plegamiento de proteínas por medio de una secuencia de aminoácidos dada por el usuario, en dos tipos de espacios diferentes (homogéneo y correlacional), y en diversas dimensiones tales como: la 2D-Cuadrada, la 2D-Triangular, y la 3D cúbica.

1.2. Objetivo

Los objetivos generales del proyecto es desarrollar una SPA API-RESTFULL, la cual su funcionalidad más importante es modelar el plegamiento de proteínas en un espacio homogéneo o correlacional, implementando una técnica de optimización inspirada en los procesos evolutivos (Algoritmo Evolutivo). Dentro de los principales objetivos particulares del sistema se encuentran los siguientes:

1. **Reingeniería/Migración del sistema heredado:** Se deberán migrar todos los requerimientos Funcionales/No funcionales del sistema heredado al nuevo lenguaje de programación y tecnologías modernas que se van a utilizar.
2. **Optimización del Algoritmo Evolutivo:** Se deberá optimizar el Algoritmo Evolutivo como una optimización Multi-objetivo, en el cual las funciones objetivo correspondientes son:
 - a) Optimizar los contactos topológicos H-H.
 - b) Optimizar los contactos topológicos P-P, dependiendo de un valor α dado.
 - c) Encontrar el valor óptimo de α para mantener el mejor núcleo (*core*) Hidrófobo.
3. **Modelado 2D/3D del plegamiento de proteínas:** Utilizando tecnologías modernas, se deberá modelar el plegamiento de proteínas en un espacio homogéneo o correlacional, en las siguientes dimensiones:
 - a) 2D Cuadrada.
 - b) 2D Triangular.
 - c) 3D Cúbica.

1.3. Modelo de contexto

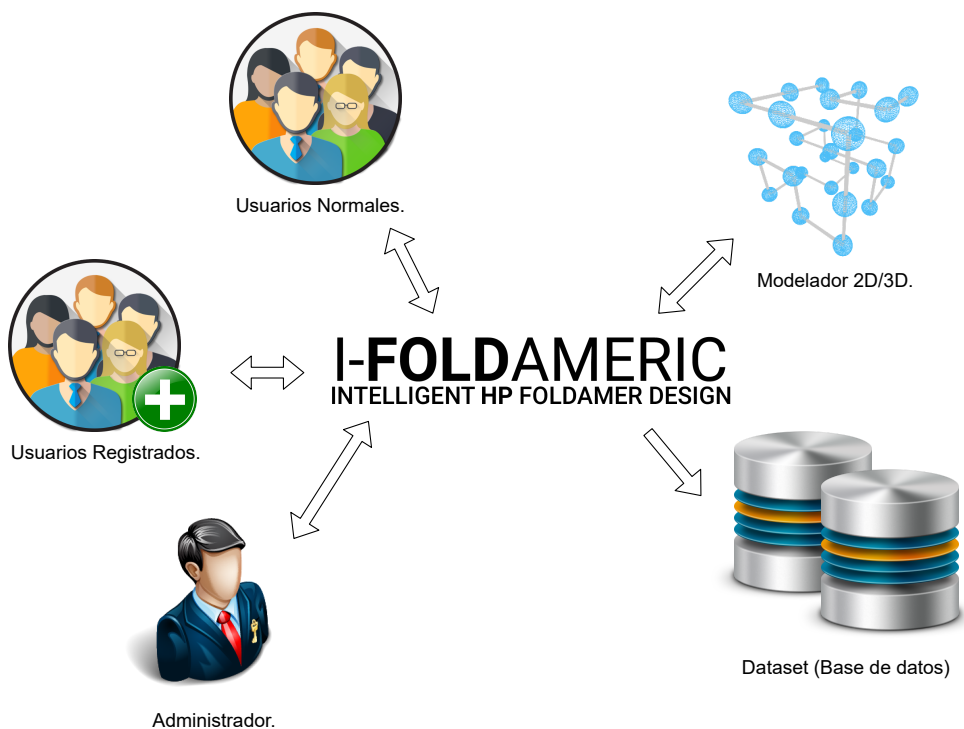


Figura 1: Diagrama de contexto del sistema I-Foldameric: IHPFD.

1.4. Diagramas de caso de uso

1.5. Diagramas de Caso de Uso de Nivel Contextual

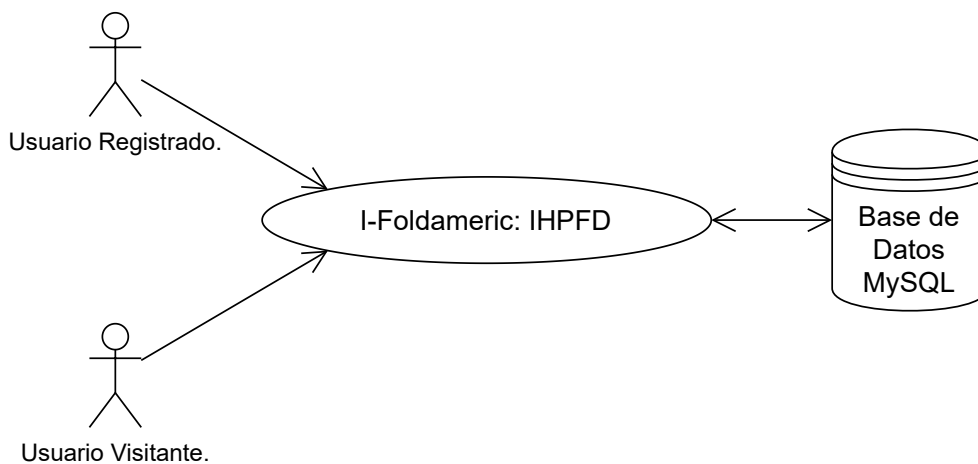


Figura 2: Diagrama de casos de uso: Nivel contextual.

1.6. Diagramas de Caso de Uso por Macro-funcionalidad

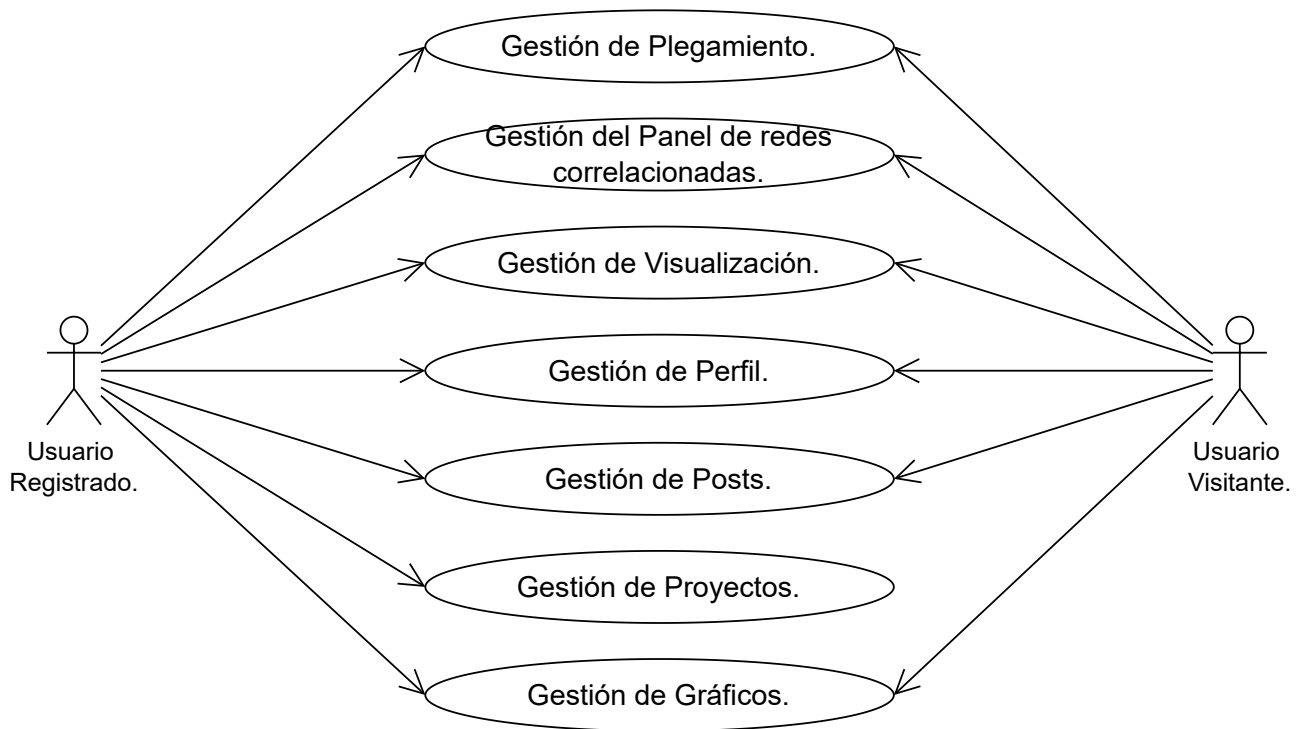


Figura 3: Diagrama de casos de uso: Nivel Macro-funcionalidad.

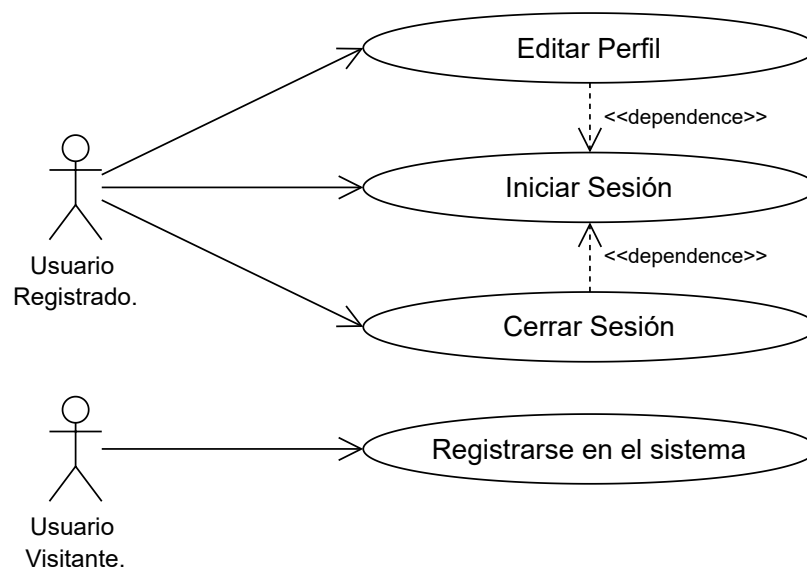


Figura 4: Diagrama de casos de uso: Gestión de Perfil.

1.7. Diagramas de Caso de Uso por Funcionalidad

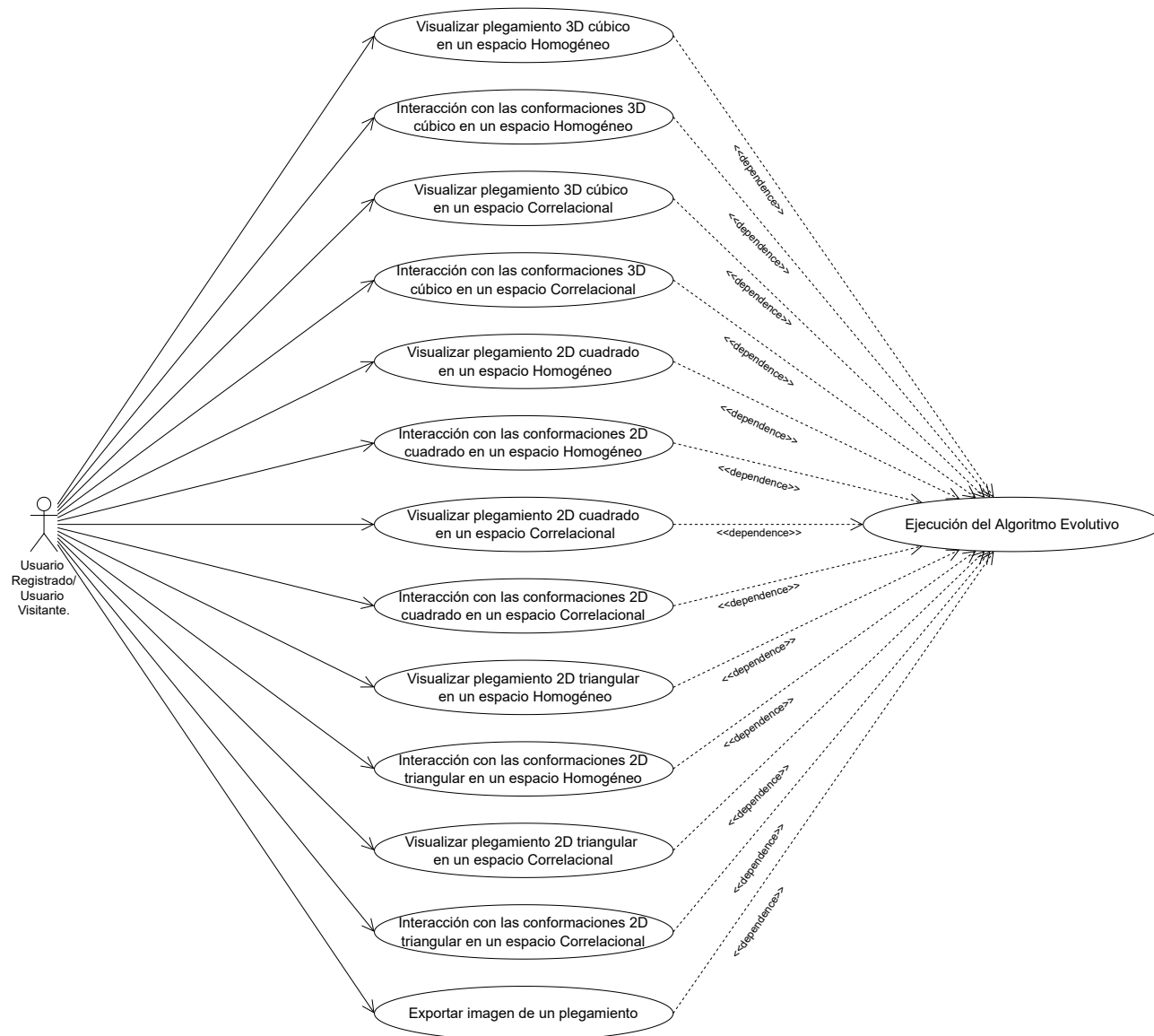


Figura 5: Diagrama de casos de uso: Gestión de Visualización.

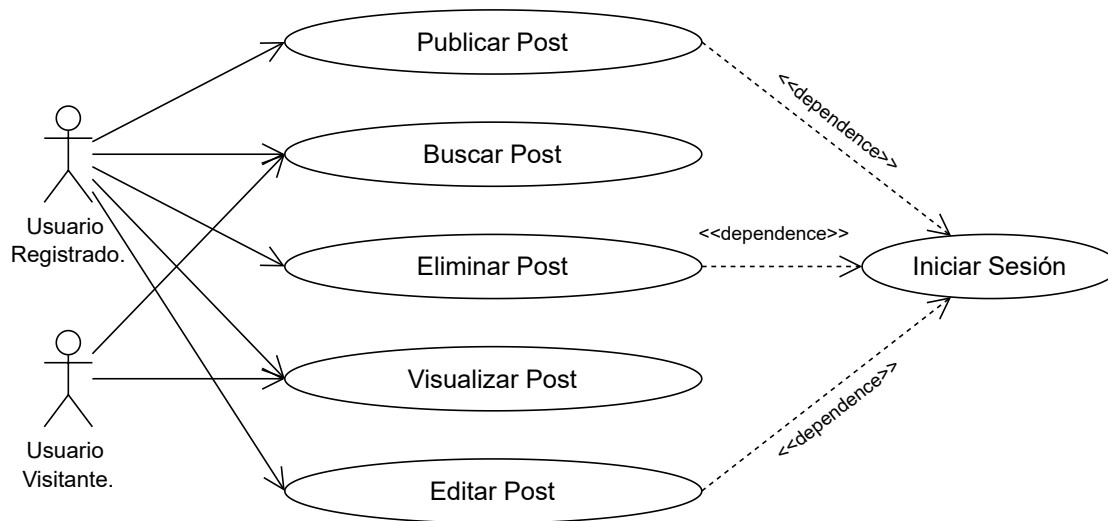


Figura 6: Diagrama de casos de uso: Gestión de Posts.

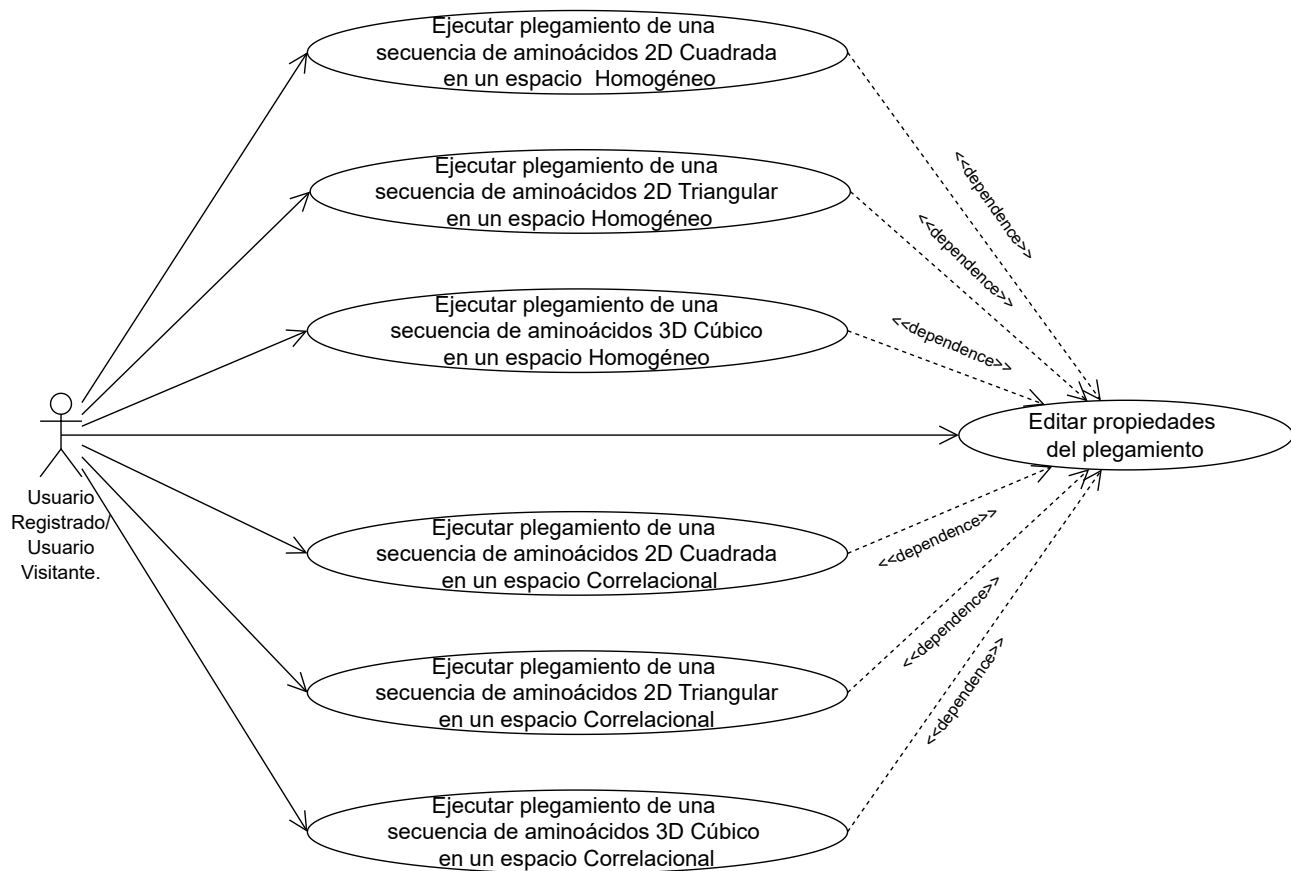


Figura 7: Diagrama de casos de uso: Gestión de Plegamientos.

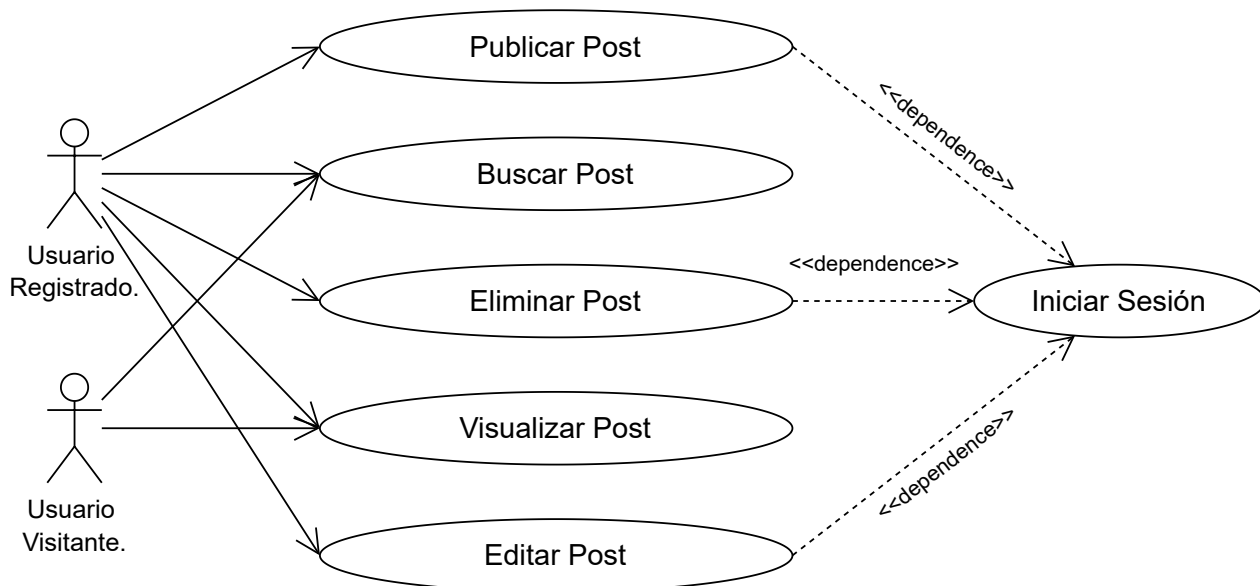


Figura 8: Diagrama de casos de uso: Gestión de Proyectos.

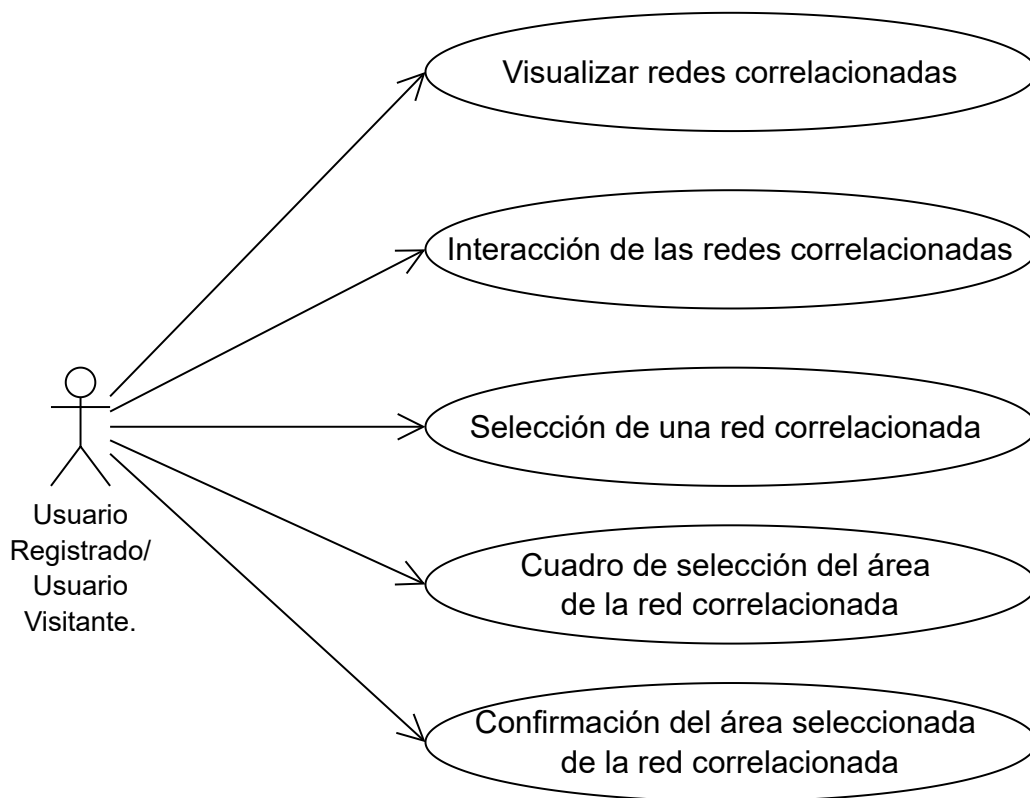


Figura 9: Diagrama de casos de uso: Gestión de Redes Correlacionadas.

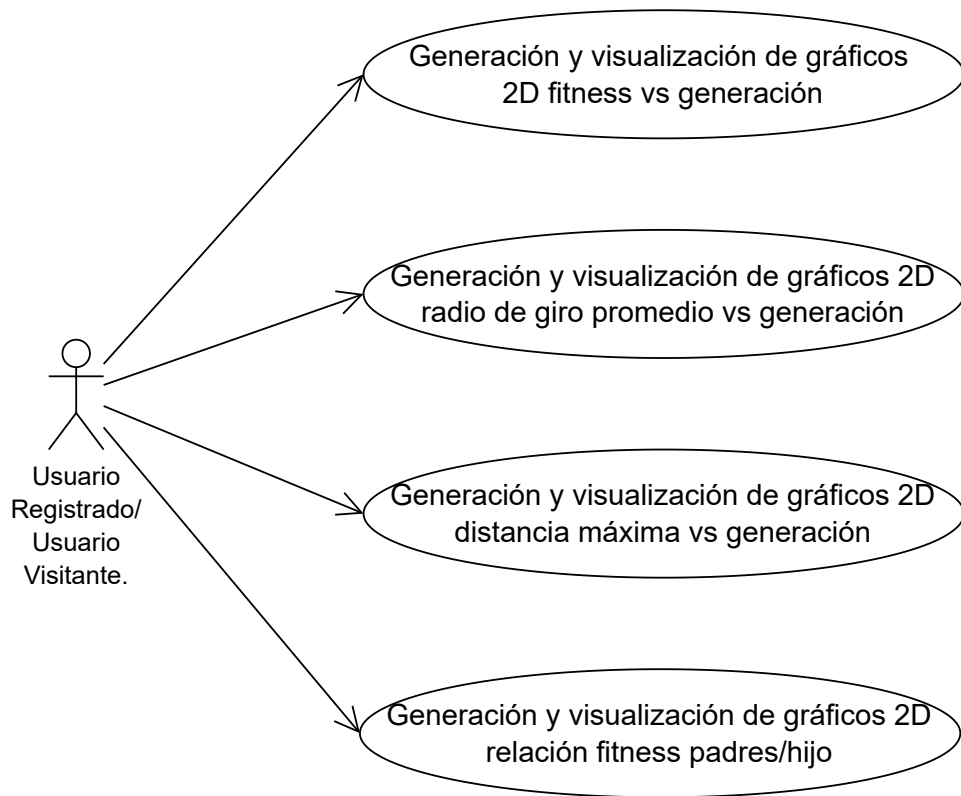


Figura 10: Diagrama de casos de uso: Gestión de Gráficos.

1.8. Especificación de requerimientos

1.8.1. Requerimientos funcionales

Actor	Funcionalidad a nivel superior	Funcionalidad detallada
Usuario Registrado / Usuario Visitante	Gestión de plegamiento	Ejecutar plegamiento de una secuencia de aminoácidos 2D Cuadrada en un espacio Homogéneo.
		Ejecutar plegamiento de una secuencia de aminoácidos 2D Triangular en un espacio Homogéneo.
		Ejecutar plegamiento de una secuencia de aminoácidos 3D Cúbico en un espacio Homogéneo.
		Ejecutar plegamiento de una secuencia de aminoácidos 2D Cuadrada en un espacio Correlacional.
		Ejecutar plegamiento de una secuencia de aminoácidos 2D Triangular en un espacio Correlacional.
		Ejecutar plegamiento de una secuencia de aminoácidos 3D Cúbico en un espacio Correlacional.
		Editar propiedades del plegamiento.
	Gestión de panel de redes correlacionadas	Visualización de las redes correlacionadas proporcionadas por la aplicación.
		Interacción de las redes correlacionadas proporcionadas por la aplicación.
		Selección de la Red Correlacionada donde se desea efectuar el plegamiento de la secuencia de aminoácidos.
		Cuadro de selección del área de la red correlacionada.
		Confirmación del área seleccionada de la red correlacionada.
	Gestión de visualización	Visualizar plegamiento en 3D cúbico en un espacio homogéneo.
		Visualizar plegamiento en 3D cúbico en un espacio correlacional.
		Interacción con las conformaciones 3D cúbico en un espacio homogéneo.

Table 1 continued from previous page

		Interacción con las conformaciones 3D cúbico en un espacio correlacional.
		Visualizar plegamiento 2D cuadrado en un espacio homogéneo.
		Visualizar plegamiento 2D cuadrado en un espacio correlacional.
		Interacción con las conformaciones 2D cuadrado en un espacio homogéneo.
		Interacción con las conformaciones 2D cuadrado en un espacio correlacional.
		Visualizar plegamiento 2D triangular en un espacio homogéneo.
		Visualizar plegamiento 2D triangular en un espacio correlacional.
		Interacción con las conformaciones 2D triangular en un espacio homogéneo.
		Interacción con las conformaciones 2D triangular en un espacio correlacional.
		Exportar imagen de un plegamiento.
	Gestión de Posts	Buscar post.
		Visualizar post.

Tabla 1: Requerimientos Funcionales, Usuario Registrado / Usuario Visitante.

Actor	Funcionalidad a nivel superior	Funcionalidad detallada
Usuario Registrado	Gestión de Proyectos	Guardar Proyecto.
		Modificar Proyecto.
		Eliminar Proyecto.
		Consultar Proyectos
		Consultar Proyecto en específico.
	Gestión de Posts	Publicar post.
		Eliminar post.
		Editar post.
	Gestión de Perfil	Editar perfil.
		Cerrar sesión.
		Iniciar sesión.
Usuario Visitante	Gestión de Perfil	Realizar registro.

Tabla 2: Requerimientos Funcionales, Usuario Registrado / Usuario Visitante.

1.8.2. Requerimientos no funcionales

Requerimientos no funcionales
La arquitectura lógica de cada sistema debe ser Model-View-Controller (MVC) Multicapa.
La arquitectura física del sistema será Cliente-Servidor Web.
Proporcionará un nivel de seguridad diferente para cada tipo de usuario.
El sistema se conectará a una base de datos
El sistema deberá ser altamente visual e interactivo

Tabla 3: Requerimientos No Funcionales.

1.9. Glosario

Entidad	Descripción	Clase
Sistema web	Aplicación de Software que puedes utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador.	No
Multiplataforma	Al sistema se pueda acceder desde cualquier navegador web en diferentes sistemas operativos.	No
Usuario	Persona registrada dentro del sistema web I-Foldameric con el beneficio de disfrutar todas las funcionalidades que le corresponde.	Si
Usuario visitante	Persona capaz de usar el simulador de plegamiento de proteínas que ofrece I-Foldameric, exceptuando funcionalidades de guardar resultados.	No
Post	Forma en que un usuario, hace públicos resultados (proyecto) obtenidos con la plataforma I-Foldameric para que otros usuarios puedan consultarlos.	Si
Proyecto	Forma en que un usuario, tiene la posibilidad de almacenar los resultados obtenidos después de la simulación del plegamiento de una proteína.	Si
Proteína	Secuencia de aminoácidos de diferentes tipos. Estos se pueden dividir en dos tipos, Hidrofóbico e Hidrofílicos.	No
Secuencia	Cadena alfabética de longitud L, escrita a partir de un alfabeto de 2 caracteres (H, P) o de un alfabeto de 20 caracteres (los principales 20 aminoácidos).	No
Aminoacido	Los aminoácidos son moléculas que se combinan para formar proteínas. Los aminoácidos y las proteínas son los pilares fundamentales de la vida.	No
Modelo HP	El Modelo HP de Dill, consiste en separar a las proteínas en dos tipos: Hidrofílicos e Hidrofóbicos. Asignando un color específico a cada una de estas.	No
Algoritmo Evolutivo	Los algoritmos evolutivos son métodos de optimización y búsqueda de soluciones basados en los postulados de la evolución biológica.	Si
Fitness	Valor asignado a cada conformación de una población. Este valor es obtenido a partir de su forma de plegamiento y que tan optimizado esté.	Si
Operador Genético	Un operador genético son las partes que conforman un algoritmo evolutivo. Y son esenciales para el funcionamiento de este.	No
Operador Genético de Cruce	El operador de cruce, se encarga de realizar el cruce y obtención de nuevas conformaciones. El cruce se realiza con dos padres, obtenidos de una población.	Si
Operador Genético de Selección	El operador de selección es el encargado de; mediante un proceso de selección especializado, realizar la selección de las conformaciones a cruzar.	Si
Elitismo	El elitismo consiste en, mediante el operador de selección, conservar un n porcentaje de los mejores individuos de una población.	Si

Table 4 continued from previous page

Entidad	Descripción	Clase
Operador Genético de Mutación	El operador genético de mutación es el encargado de realizar mutaciones luego de un cruce de dos padres. La mutación es realizada con técnicas especiales.	Si
Operador Genético de Formación de Parejas	El operador de formación de parejas, es el encargado de seleccionar las parejas a cruzar, luego de haber realizado la parte del operador de selección.	Si
Operador Genético de Generación de puntos	El operador genético encargado de la generación de puntos tiene la tarea de generar puntos que serán una conformación. La generación de estos puntos se realiza de forma específica, considerando la posición de cada n-1 punto.	Si
Experimento	Un experimento consiste en una n cantidad de generaciones obtenidas. Esta n es proporcionada por el usuario que quiere simular el plegamiento.	No
Generación	Una generación es un conjunto de k conformaciones. Las generaciones van cambiando conforme se ven afectadas por los operadores genéticos.	Si
Conformación	Una conformación es la representación computacional de un plegamiento de una proteína. Donde cada conformación está representada por los puntos de las posiciones de cada aminoácido en el plano 3D y 2D.	Si
Plegamiento	El plegamiento es el resultado obtenido luego de aplicar el algoritmo evolutivo, donde podemos ver en el simulador como se plegó la proteína.	No
Plegamiento homogéneo	El plegamiento homogéneo consiste en que la proteína que se está plegando, no tiene restricciones o barreras de en que posición x, y, z se puede plegar.	No
Plegamiento correlacional	El plegamiento correlacional tiene restricciones que afectan al plegamiento. Es decir, hay posiciones x, y, z que el plegamiento no puede tomar, provocando que los resultados pueden ser diferentes al plegamiento homogéneo.	No
Red correlacional	Una red correlacional es la encargada de indicar en que posiciones x, y, z un plegamiento se puede realizar su plegamiento.	Si
Resultados	Los resultados son los experimentos obtenidos luego de aplicar el algoritmo evolutivo. En estos los experimentos tienen generaciones y las generaciones conformaciones.	Si
Visualizador de resultados	El visualizador de resultados se encarga de pasar la información obtenida con el algoritmo evolutivo a una interfaz gráfica, donde el usuario puede interactuar con los resultados, ampliando totalmente la usabilidad de la plataforma I-Foldameric.	No

Table 4 continued from previous page

Entidad	Descripción	Clase
Gráficas de resultados	Las gráficas de resultados se realizan con base en el fitness y radio de giro. Estas sirven para dar una interpretación más completa a los resultados, y complementar el trabajo ya realizado por el algoritmo evolutivo.	No

Tabla 4: Glosario

1.10. Modelo de Dominio

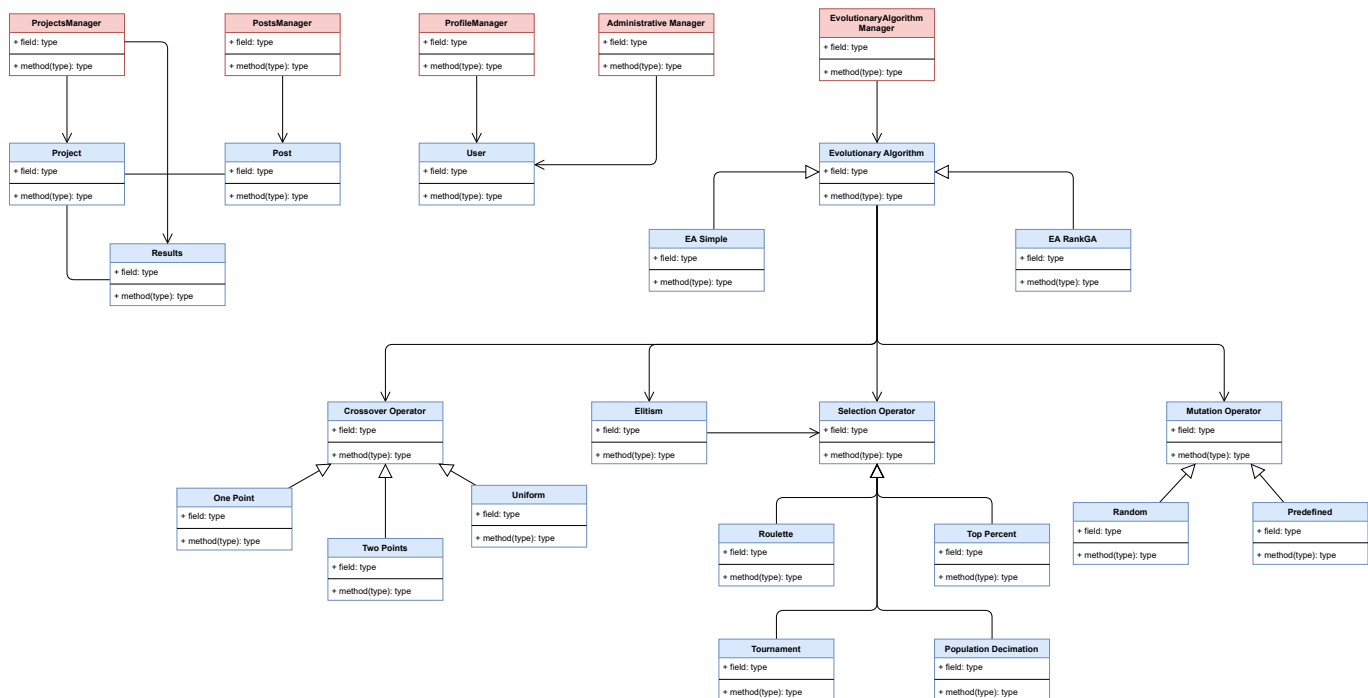


Figura 11: Modelo del Dominio.

1.11. Plan de iteración

Las actividades que se llevarán a cabo en la primera iteración son las siguientes:

- Refinamientos
 - Refinamiento del objetivo

- Refinamiento de visión (opcionalmente)
- Elaboración de bocetos
- Refinamiento de modelos de contexto
- Refinamiento diagrama de casos de uso
- Refinamiento especificación complementaria
- Refinamiento glosario
- Inicio y refinamiento del modelo de diseño basado en el modelo del dominio
- Validación de requerimientos
 - Validación con diagramas de actividades
 - Validación con diagramas de secuencia
- Diagrama de clases

2. Fase de elaboración

2.1. Elaboración de bocetos

The wireframe shows a web interface for 'I-FOLDAMERIC INTELLIGENT HP FOLDAMER DESIGN'. At the top, there is a dark header with 'Login' and 'Register' links. Below this, the main content area has a light gray background. It features a form with the following elements: a text input for 'Aminoacid or HP Sequence'; three columns of inputs for 'Space Types', 'Space Dimensions', and 'Optimization Algorithms'; and three columns of inputs for 'Selection Operator', 'CrossOver Operator', and 'Mutation Operator'. The title 'I-FOLDAMERIC' is prominently displayed in large, bold, black letters, with the subtitle 'INTELLIGENT HP FOLDAMER DESIGN' below it. At the bottom, a dark footer contains the text 'Universidad Autonoma Metropolitana.' followed by a line of placeholder text: 'Praesent sed lobortis mi. Suspendisse vel placerat ligula. Vivamus ac sem lacus. Ut vehicula rhoncus elementum. Etiam quis tristique lectus. Aliquam in arcu eget veli'.

Figura 12: Boceto de la pantalla de inicio de I-Foldameric.

This mockup shows a login interface. At the top, a black header contains the links "Login" and "Register". The main area has a gray background with a centered white form. The form includes an "Email" label above a white input field, a "Password" label above another white input field, and a blue "SIGN IN" button below them. At the bottom, a black footer contains the text "Universidad Autonoma Metropolitana." followed by a line of placeholder text: "Praesent sed lobortis mi. Suspendisse vel placerat ligula. Vivamus ac sem lacus. Ut vehicula rhoncus elementum. Etiam quis tristique lectus. Aliquam in arcu eget veli".

Figura 13: Boceto de la pantalla de login de I-Foldameric.

This mockup shows a registration interface. It features a black header with "Login" and "Register" links. The main gray area contains a centered white form with four input fields labeled "Name", "Surname", "Email", and "Password" stacked vertically. A blue "SIGN UP" button is positioned below the fields. The black footer at the bottom displays "Universidad Autonoma Metropolitana." and the same placeholder text as the login page: "Praesent sed lobortis mi. Suspendisse vel placerat ligula. Vivamus ac sem lacus. Ut vehicula rhoncus elementum. Etiam quis tristique lectus. Aliquam in arcu eget veli".

Figura 14: Boceto de la pantalla de registro de I-Foldameric.

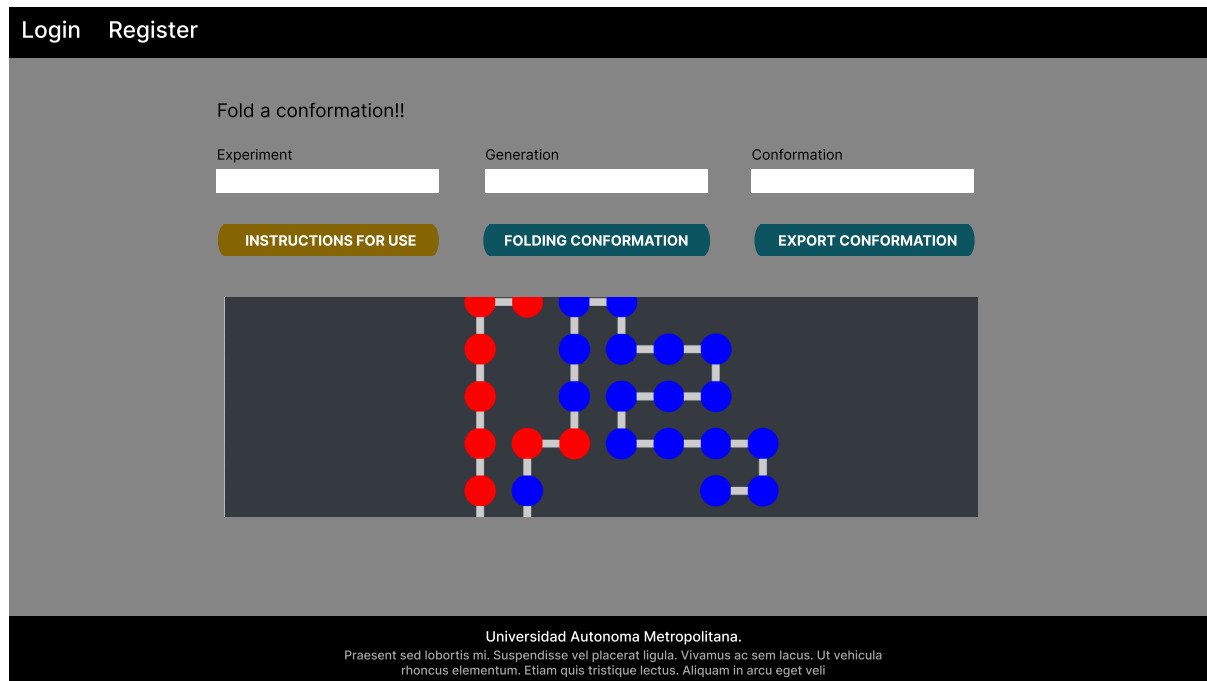


Figura 15: Boceto de la pantalla de resultados de I-Foldameric.

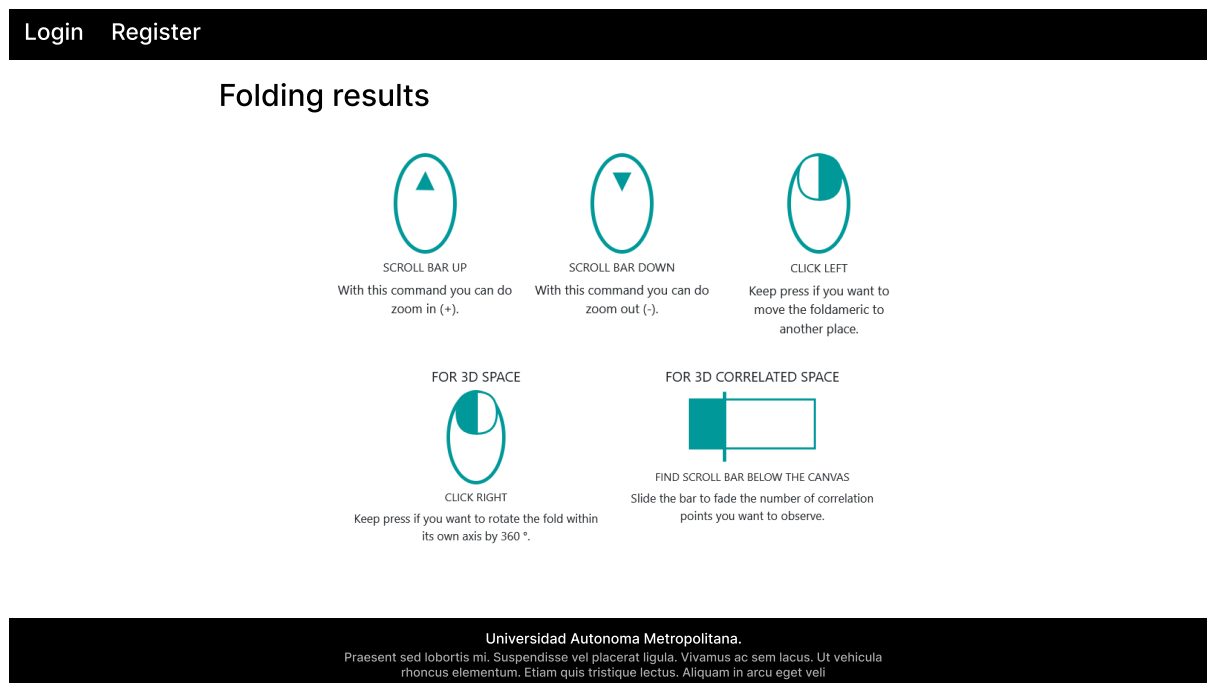


Figura 16: Boceto de la pantalla de instrucciones de I-Foldameric.

2.2. Diagramas de actividades

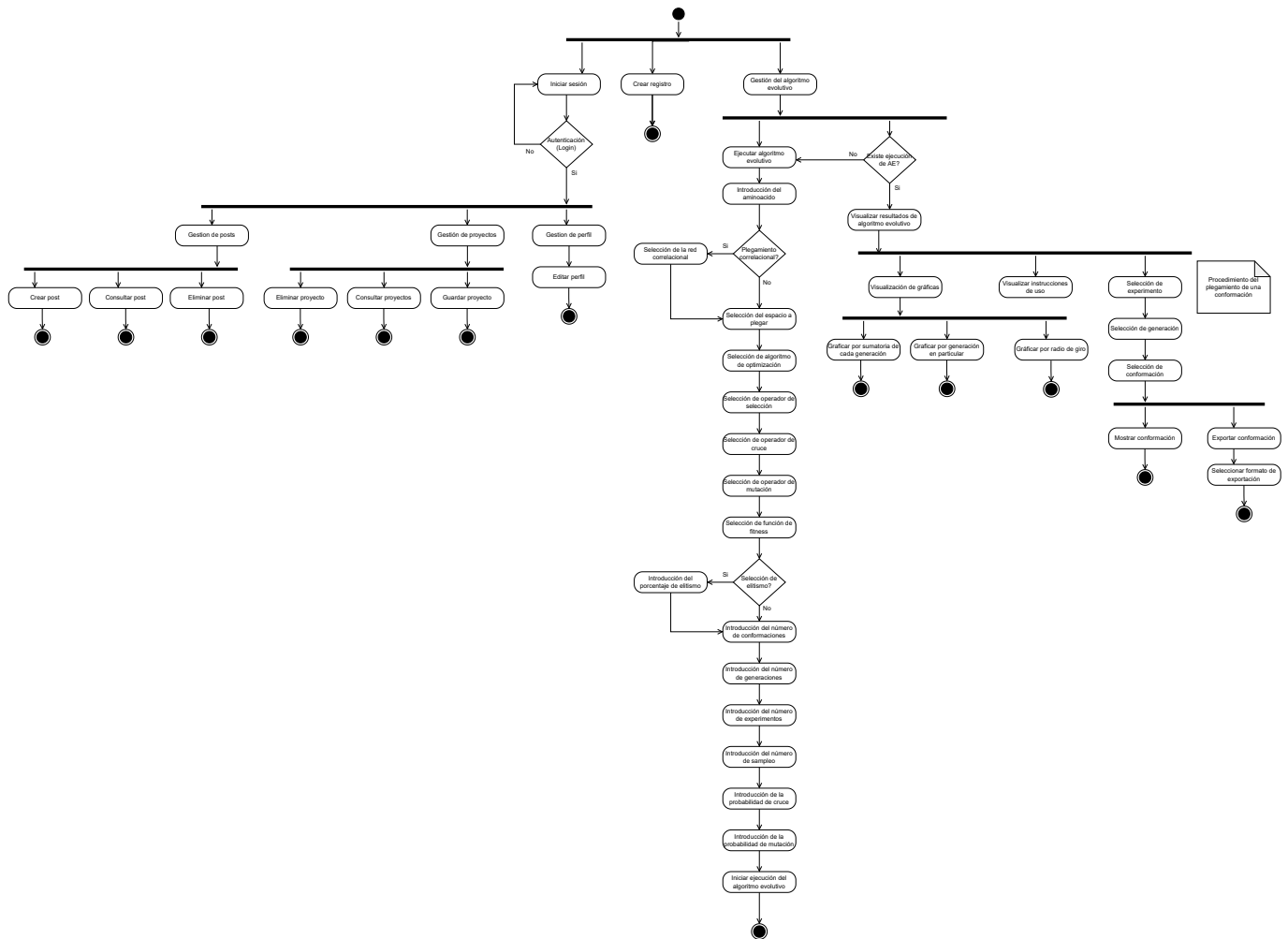


Figura 17: Diagrama de Actividades.

2.3. Diagrama de Clases

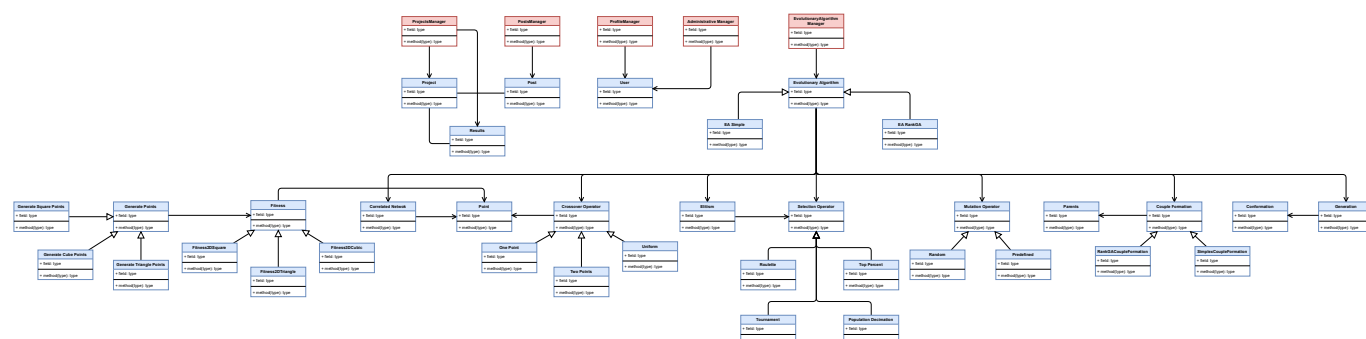


Figura 18: Diagrama de Clases del sistema I-Foldameric.

2.4. Diagramas de paquetes

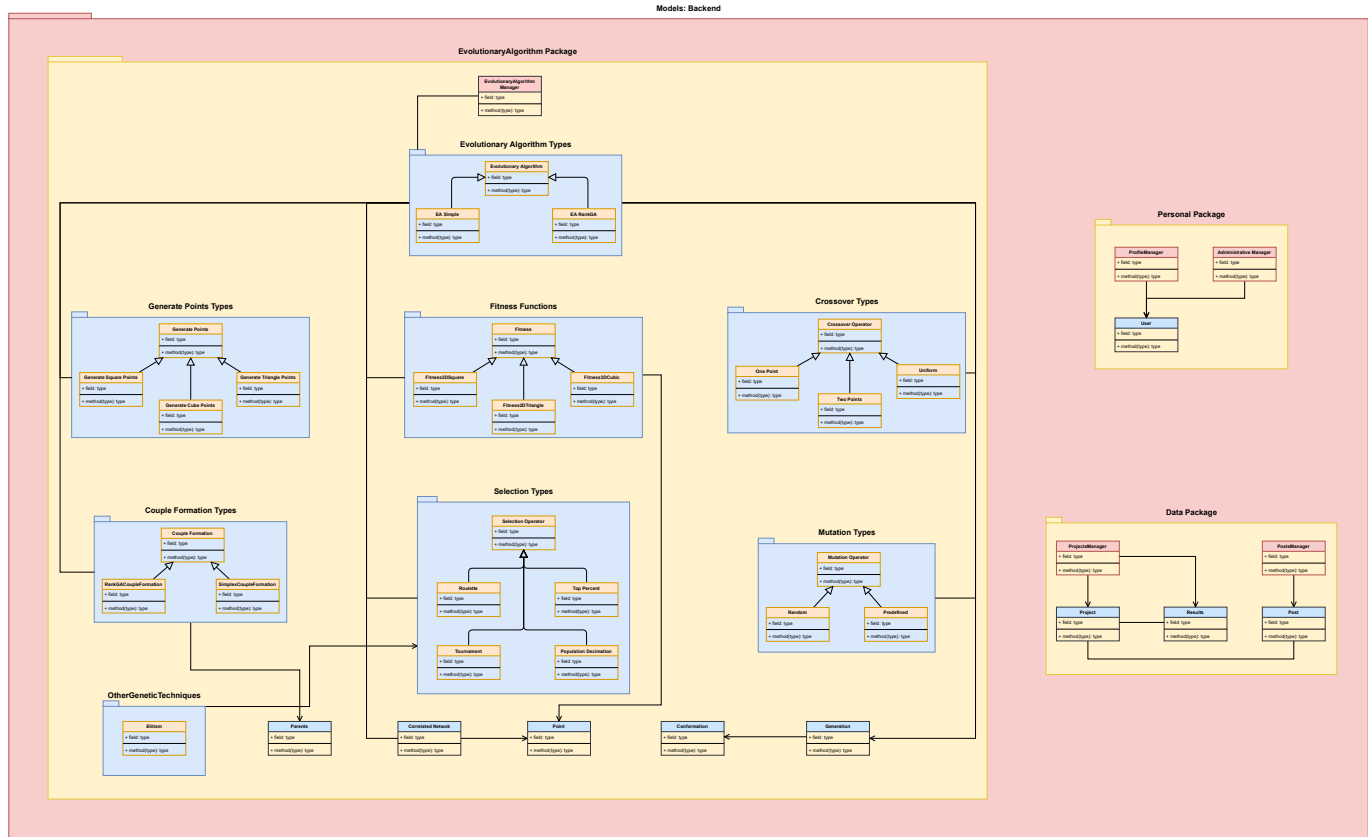


Figura 19: Diagrama de Paquetes Backend: capa Model

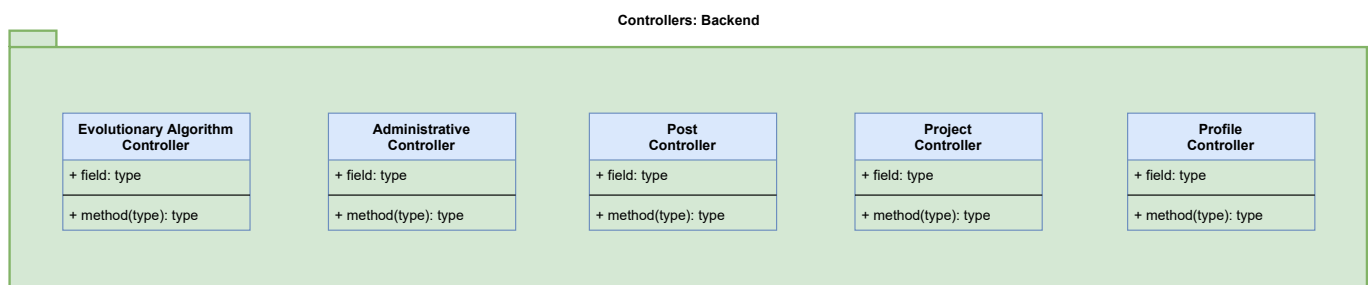


Figura 20: Diagrama de Paquetes Backend: capa Controller

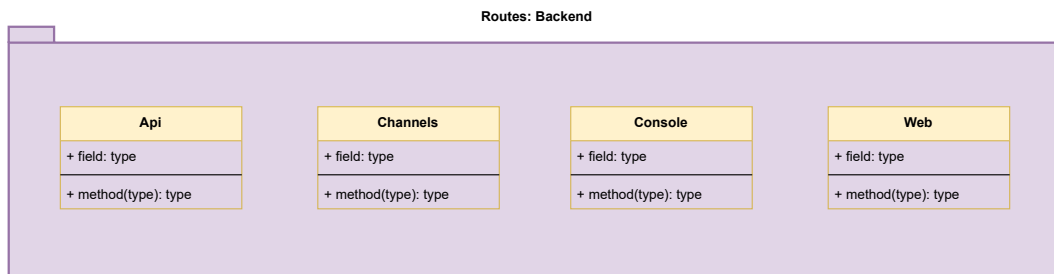


Figura 21: Diagrama de Paquetes Backend: capa Routes

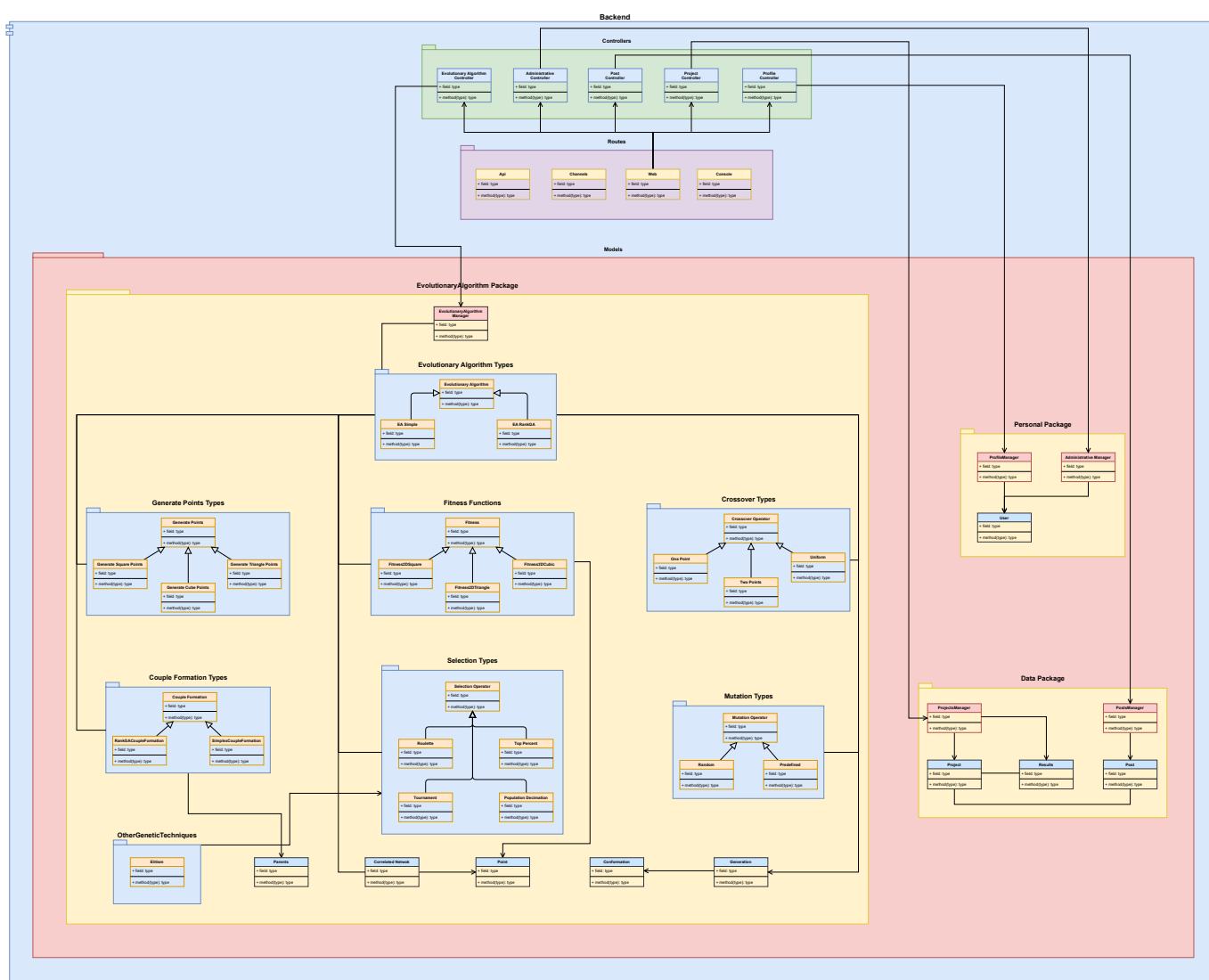


Figura 22: Diagrama de Paquetes Backend

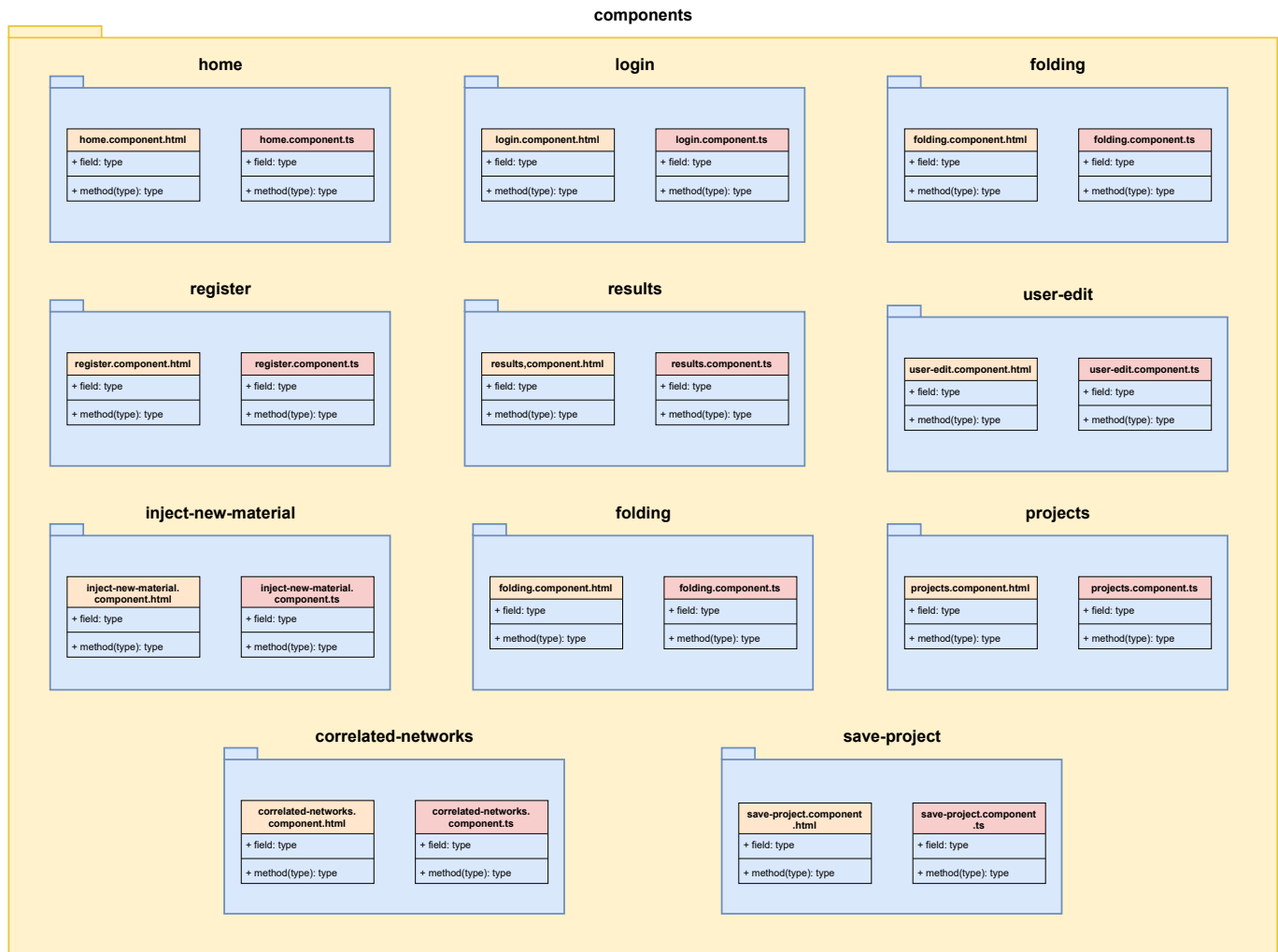


Figura 23: Diagrama de Paquetes Frontend: capa components

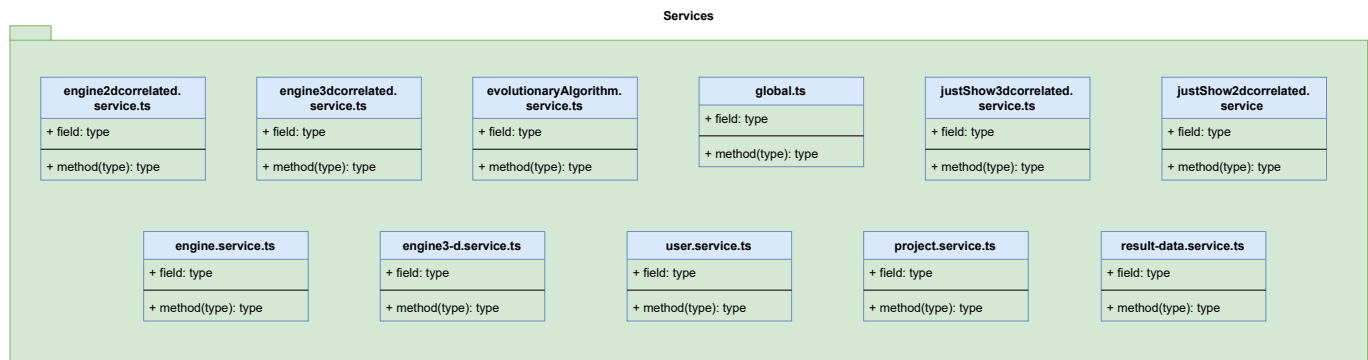


Figura 24: Diagrama de Paquetes Frontend: capa services

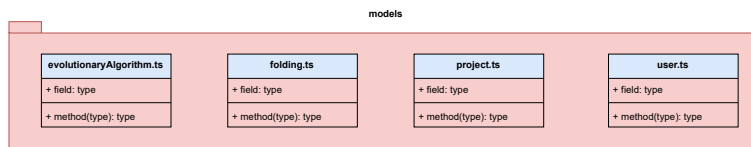


Figura 25: Diagrama de Paquetes Frontend: capa models

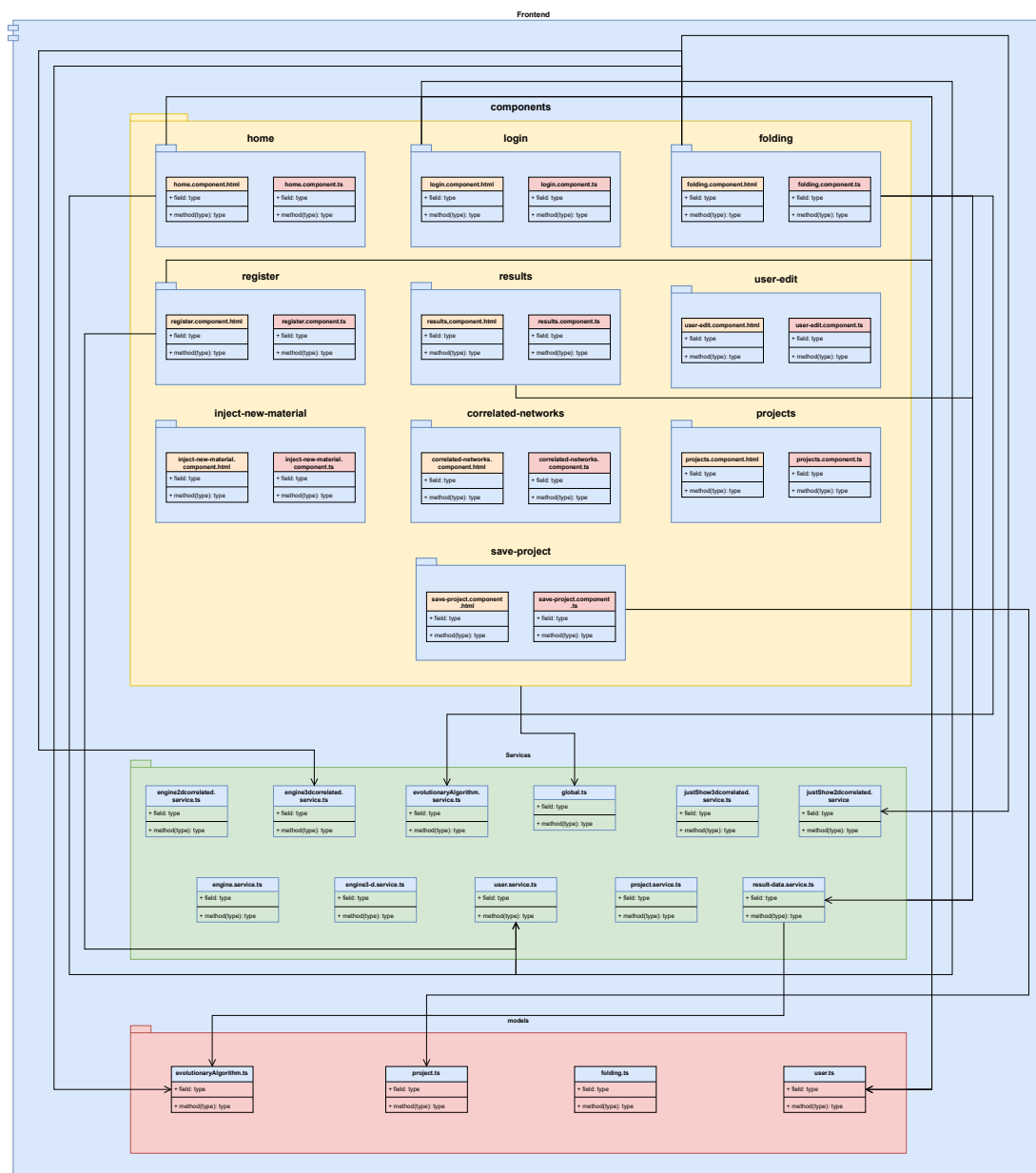


Figura 26: Diagrama de Paquetes Frontend

2.5. Inicio de la arquitectura de software

2.5.1. Arquitectura lógica

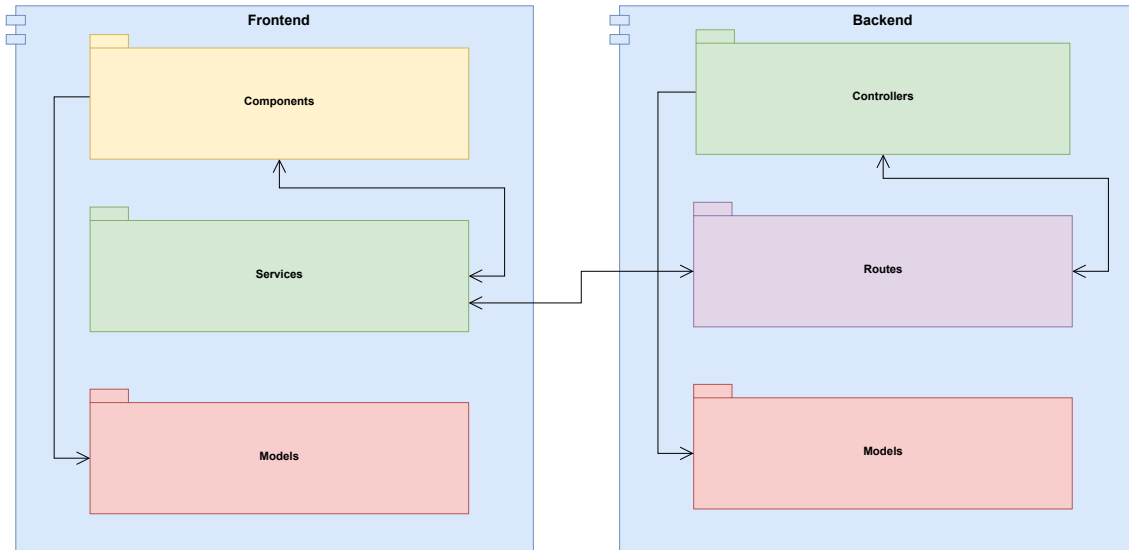


Figura 27: Núcleo Arquitectónico del sistema I-Foldameric.

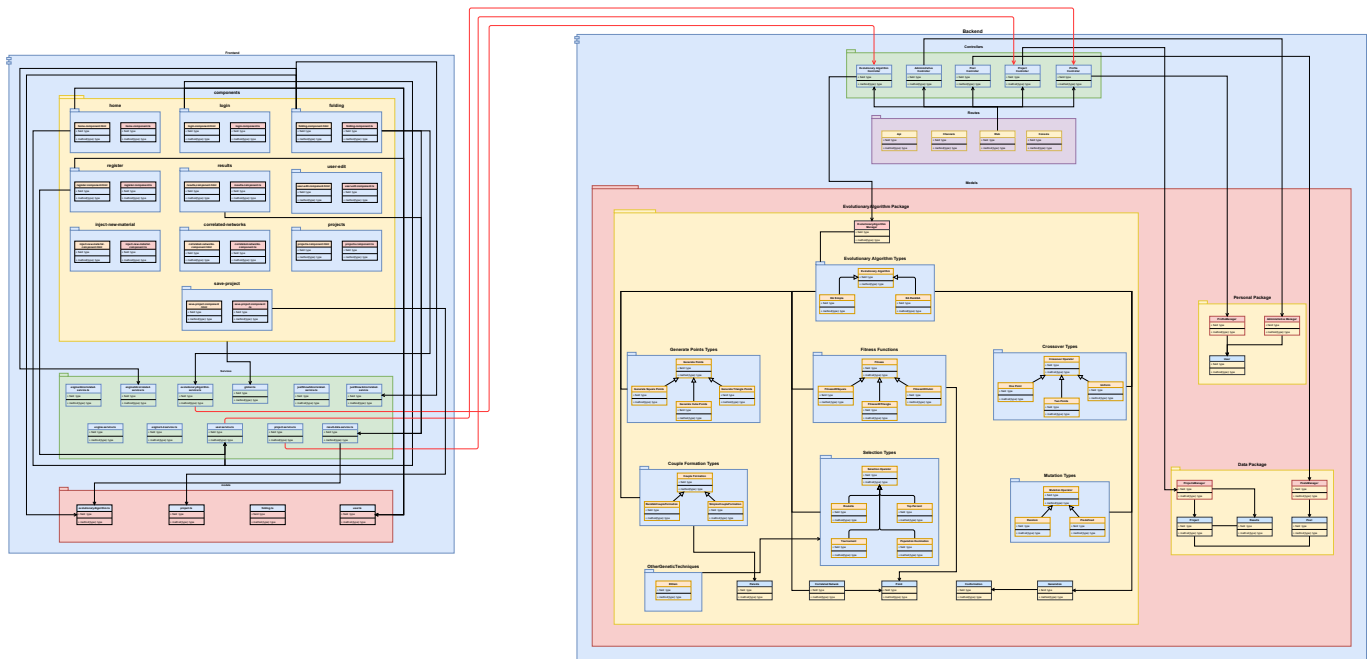


Figura 28: Núcleo Arquitectónico del sistema I-Foldameric: Paquetes y clases.

2.5.2. Glosario para las vistas

Clase	Descripción
home.component.html	Esta interfaz gráfica se encarga de presentar posts, o publicaciones propias y de otros usuarios, además de presentar noticias asociadas de la plataforma.
login.component.html	En esta interfaz gráfica se presenta el formulario de login de usuarios, permitiendo acceder mediante el uso de credenciales a más funcionalidades de la plataforma.
folding.component.html	Esta interfaz gráfica se encarga de presentar el formulario que permite al usuario introducir los parámetros necesarios del algoritmo evolutivo para el plegamiento de proteínas.
register.component.html	En esta interfaz gráfica permite al usuario realizar el registro de sus datos para acceder a más funcionalidades de la plataforma.
results.component.html	Esta interfaz gráfica se encarga de presentar los resultados del un plegamiento de proteína. Además, permite visualizar analíticas correspondientes al comportamiento del algoritmo evolutivo exportar plegamientos y visualizar las instrucciones para interactuar con la plataforma.
user-edit.component.html	Esta interfaz gráfica le permite los usuarios registrados modificar sus datos personales en la plataforma I-Foldameric.
inject-new-material.component.html	En esta interfaz gráfica se presenta al usuario el formulario, donde puede seleccionar el material genético para realizar volver a ejecutar el algoritmo evolutivo, con una generación especificada por el usuario.
projects.component.html	Esta interfaz gráfica permite visualizar e interactuar con los proyectos almacenados pertenecientes al usuario.
correlated-networks.component.html	En esta interfaz gráfica se presenta al usuario las redes correlacionadas que están disponibles para su uso en el plegamiento de proteínas en un espacio correlacional 2D cuadrado y correlacional 3D cúbico.
save-project.component.html	Esta interfaz gráfica se solicita al usuario los datos correspondientes de un proyecto para almacenarlo en el DataSet del sistema I-Foldameric.

Tabla 5: Glosario de las vistas.

2.5.3. Glosario para el controlador

Clase	Descripción
EvolutionaryAlgorithmController	Este controlador es el encargado de recibir los datos del AE desde el frontend, para su ejecución y posterior a ella, devolver el código de respuesta HTTP correspondiente y los datos de respuesta de la lógica del negocio. También se encarga de delegar los datos al modelo para la funcionalidad "inyección de material genético".
AdministrativeController	Este controlador cumple la tarea de otorgar o remover permisos administrativos a usuarios del sistema I-Foldameric, y envía como resultado el código de respuesta HTTP y la de la lógica de negocio.
PostController	El controlador PostController se encarga de administrar el contenido de los posts, envía estos datos a la lógica de negocio para su procesamiento y devuelve la respuesta HTTP y la de la lógica de negocio.
ProjectController	Este controlador se encarga de administrar los proyectos de los usuarios, recibiendo el contenido de los mismos, y delegándolos a la lógica de negocio. Devuelve la respuesta HTTP y la de la lógica de negocio.
ProfileController	El controlador ProfileController se encarga de recibir la información de los usuarios registrados en el sistema I-Foldameric, de esta manera, envía la información a la lógica de negocio. Como resultado devuelve la respuesta HTTP y de la lógica de negocio.

Tabla 6: Glosario del controlador.

2.6. Diagramas de secuencia

2.6.1. Diagramas de secuencia entre la vista y el controlador

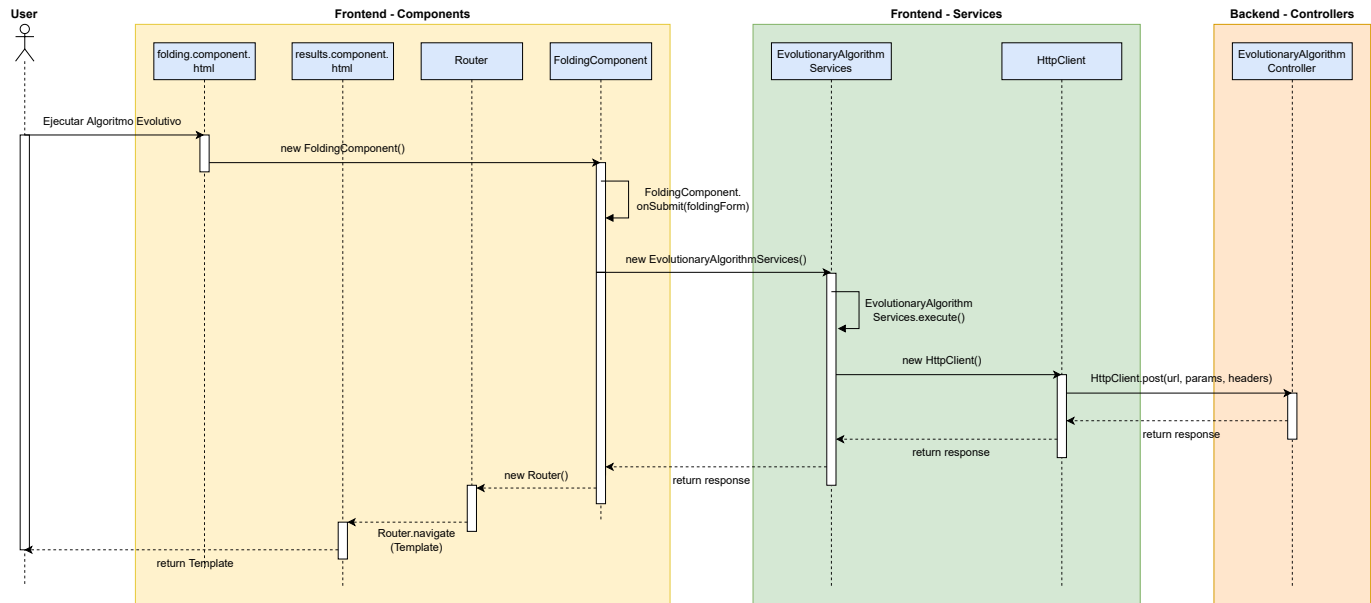


Figura 29: Diagrama de Secuencia: Ejecutar el Algoritmo Evolutivo.

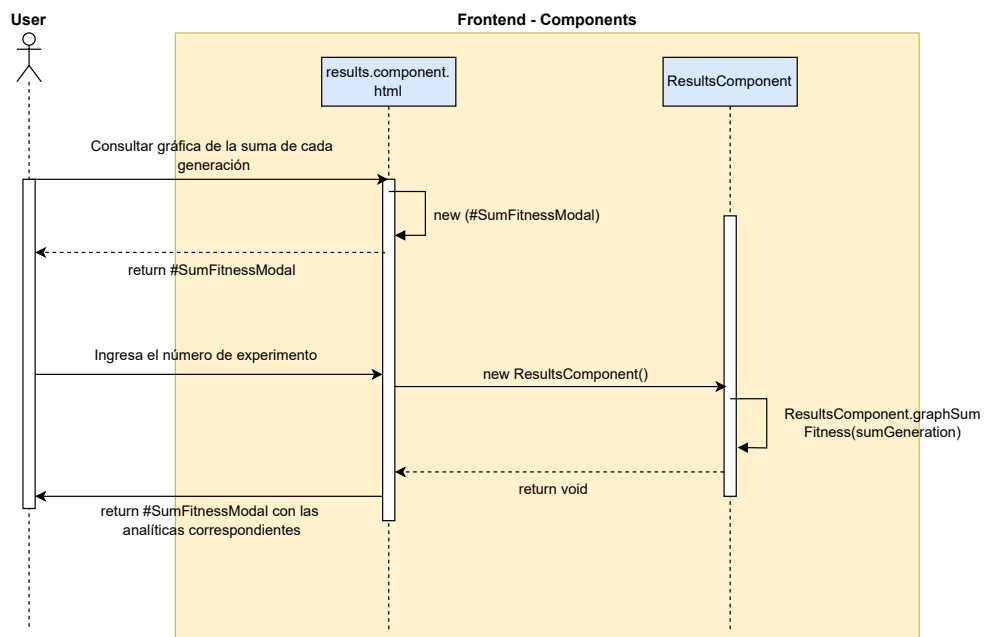


Figura 30: Diagrama de Secuencia: Graficar el comportamiento de la suma de cada Generación.

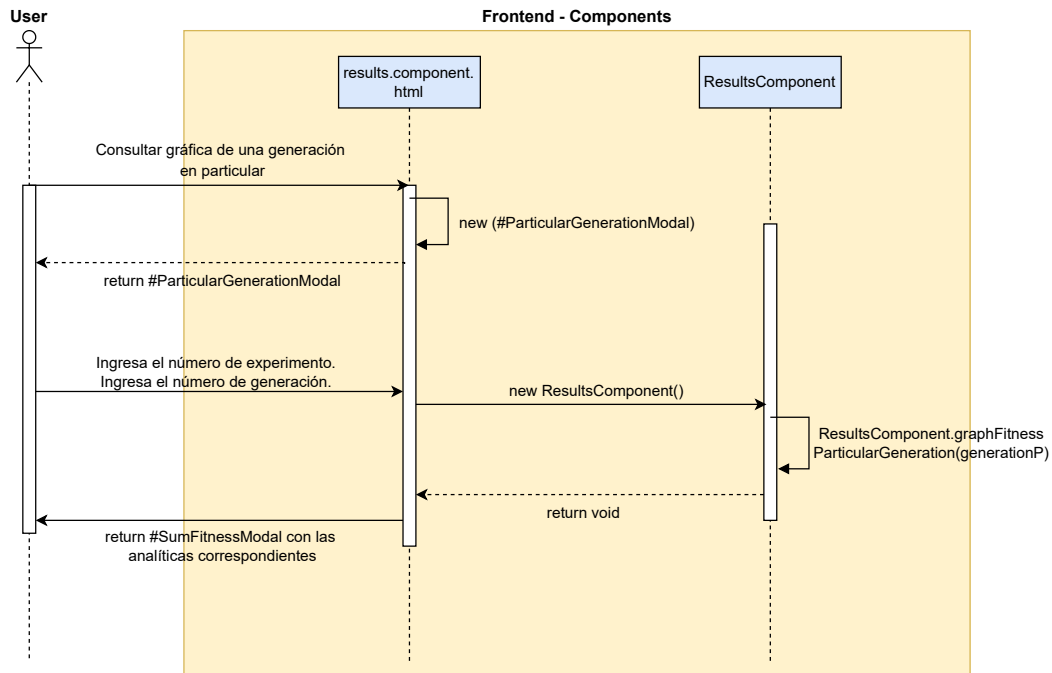


Figura 31: Diagrama de Secuencia: Graficar el comportamiento de una Generación en Particular.

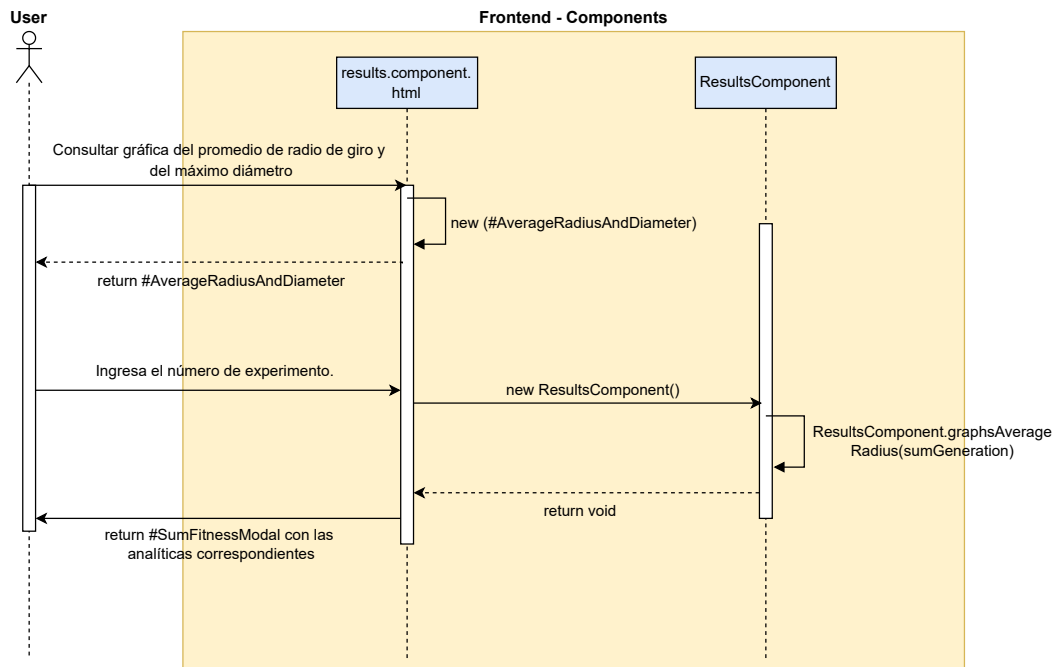


Figura 32: Diagrama de Secuencia: Graficar el comportamiento del promedio de Radio de Giro y promedio de la Distancia Máxima.

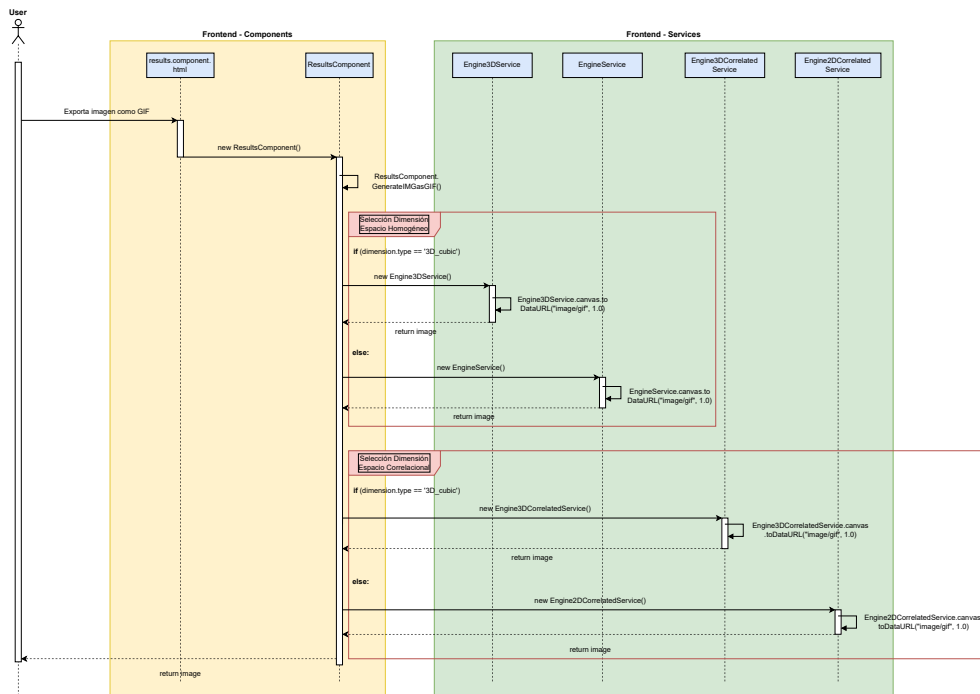


Figura 33: Diagrama de Secuencia: Exportar el Plegamiento de una Conformación como GIF.

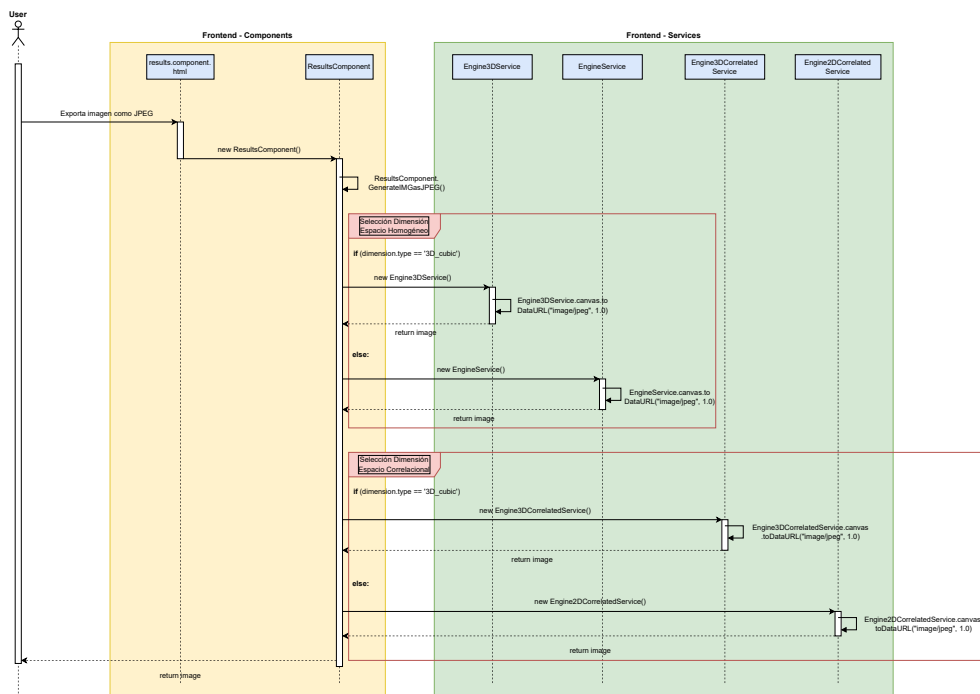


Figura 34: Diagrama de Secuencia: Exportar el Plegamiento de una Conformación como JPEG.

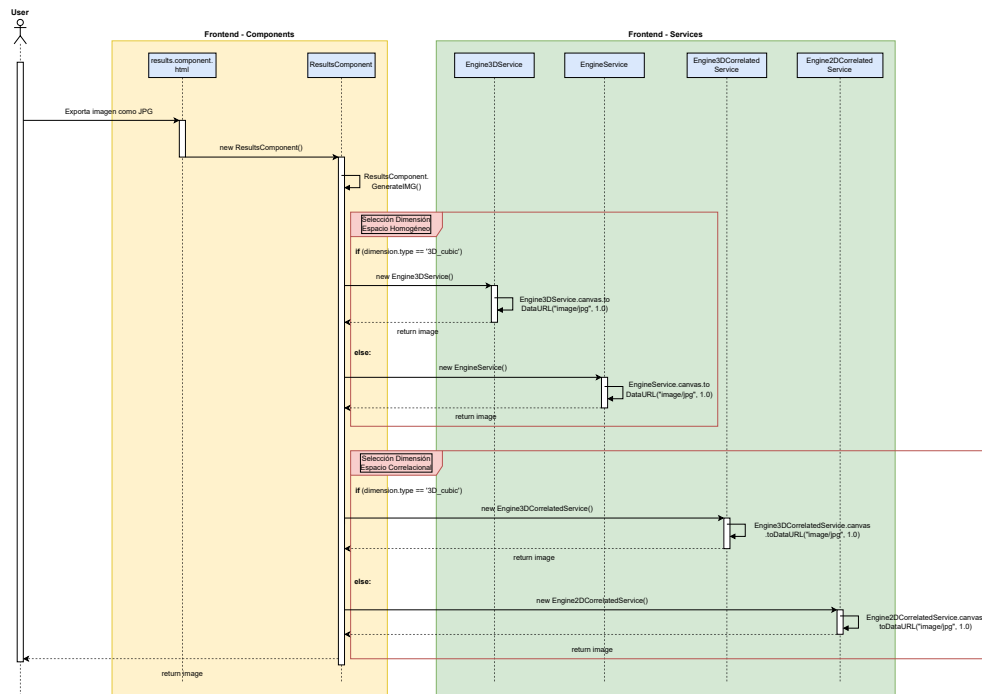


Figura 35: Diagrama de Secuencia: Exportar el Plegamiento de una Conformación como JPG.

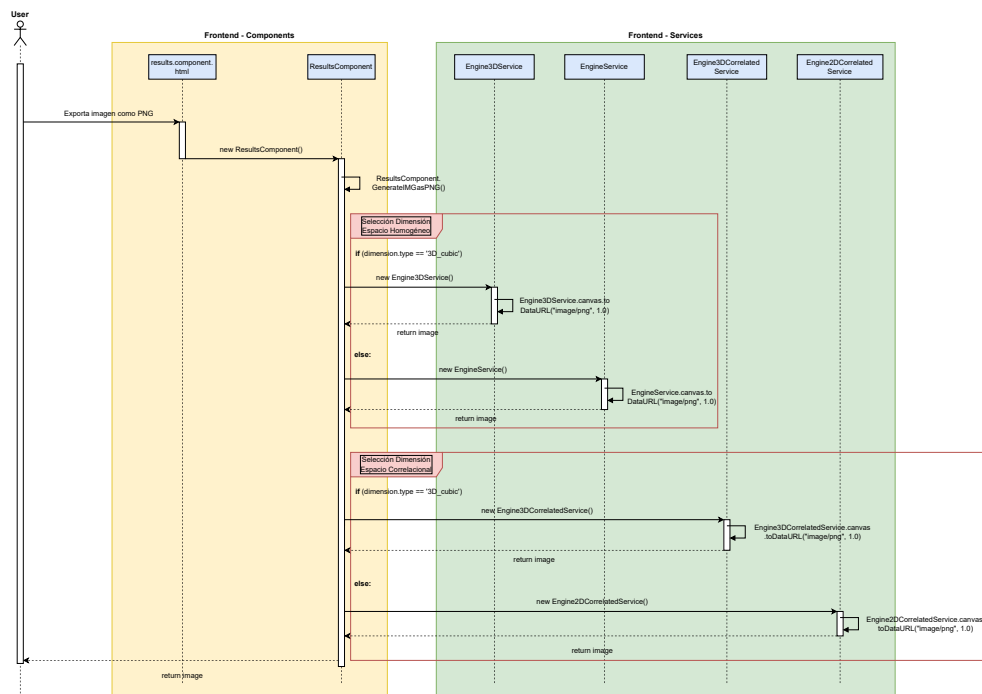


Figura 36: Diagrama de Secuencia: Exportar el Plegamiento de una Conformación como PNG.

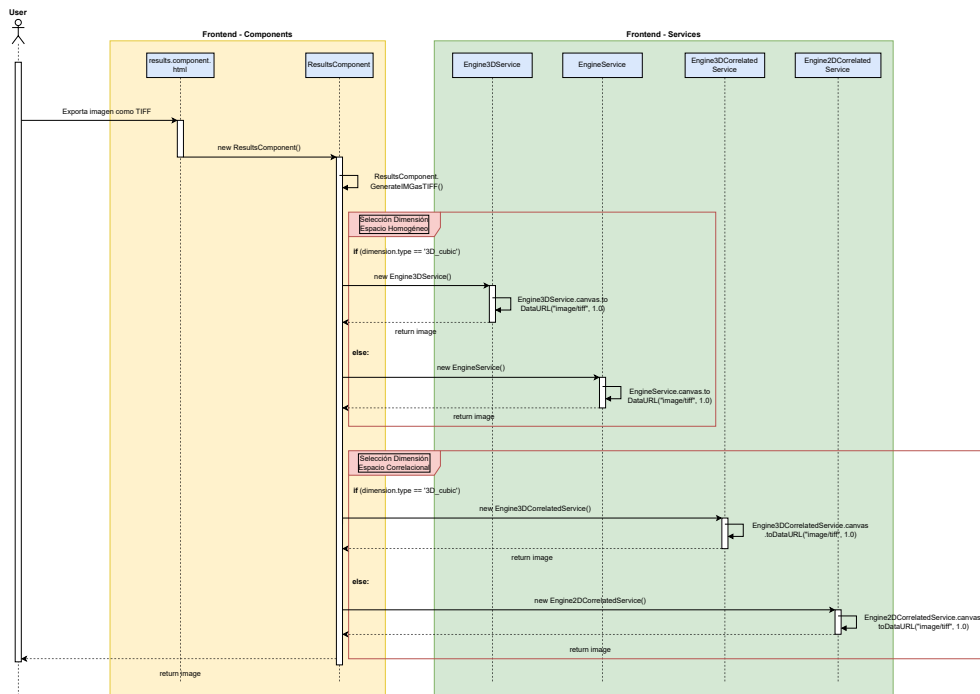


Figura 37: Diagrama de Secuencia: Exportar el Plegamiento de una Conformación como TIFF.

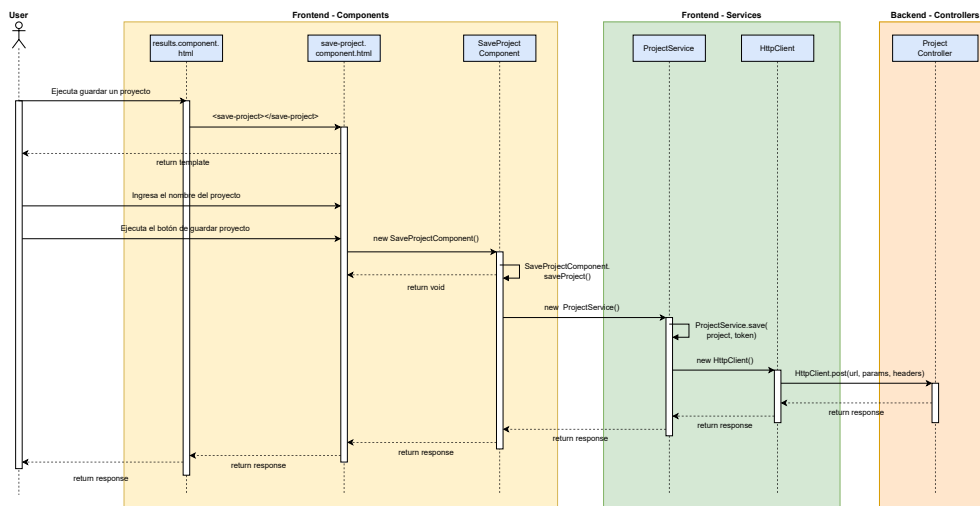


Figura 38: Diagrama de Secuencia: Guardar los resultados del Algoritmo Evolutivo como Proyecto.

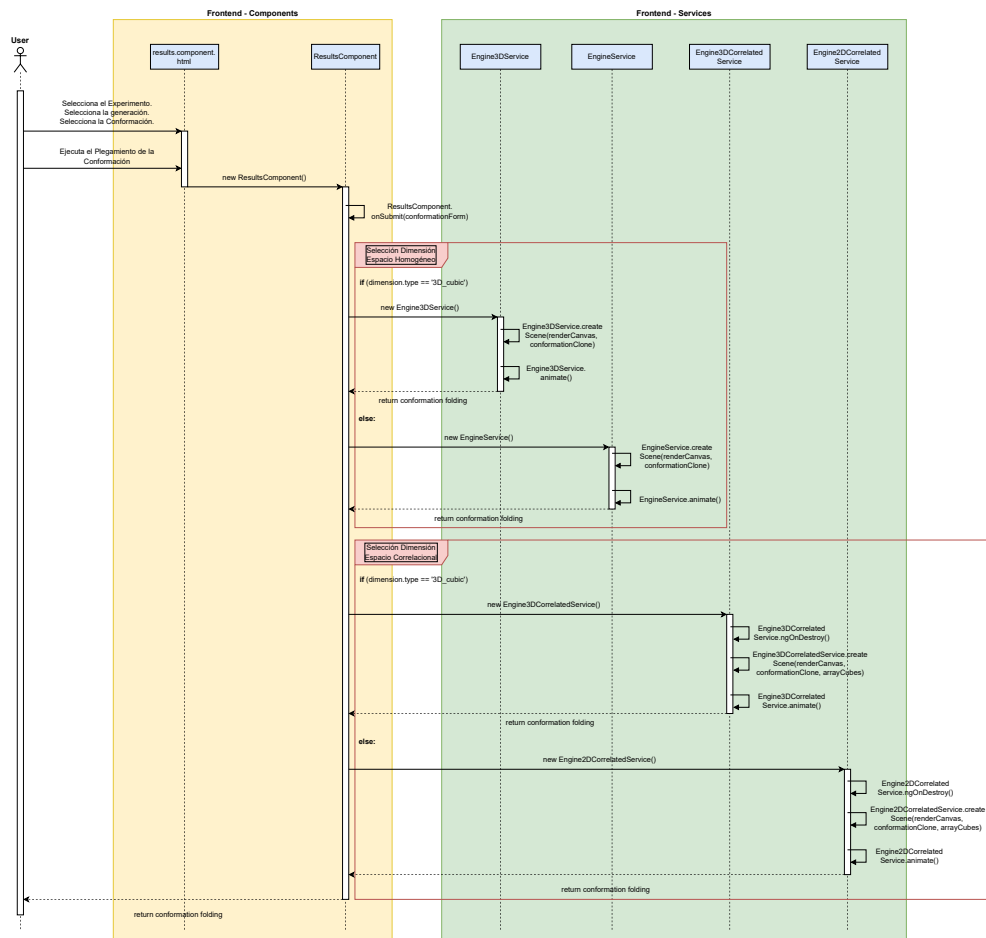


Figura 39: Diagrama de Secuencia: Plegar una Conformación de un Experimento y Generación en particular.

2.7. Refinamiento de bocetos

The refined sketch of the I-Foldameric homepage features a dark header with the site name and user links. Below this, three placeholder boxes for conformation images are arranged horizontally. The main title is prominently displayed in the center. The input section is organized into a grid with labels and corresponding text input fields for sequence, space types, dimensions, algorithms, and various operators.

I-Foldameric Login Register

conformation image conformation image conformation image

I-FOLDAMERIC
INTELLIGENT HP FOLDAMER DESIGN

Aminoacid or HP Sequence

Space Types Space Dimensions Optimization Algorithms

Selection Operator CrossOver Operator Mutation Operator

Figura 40: Refinamiento del boceto de la pantalla de inicio de I-Foldameric.

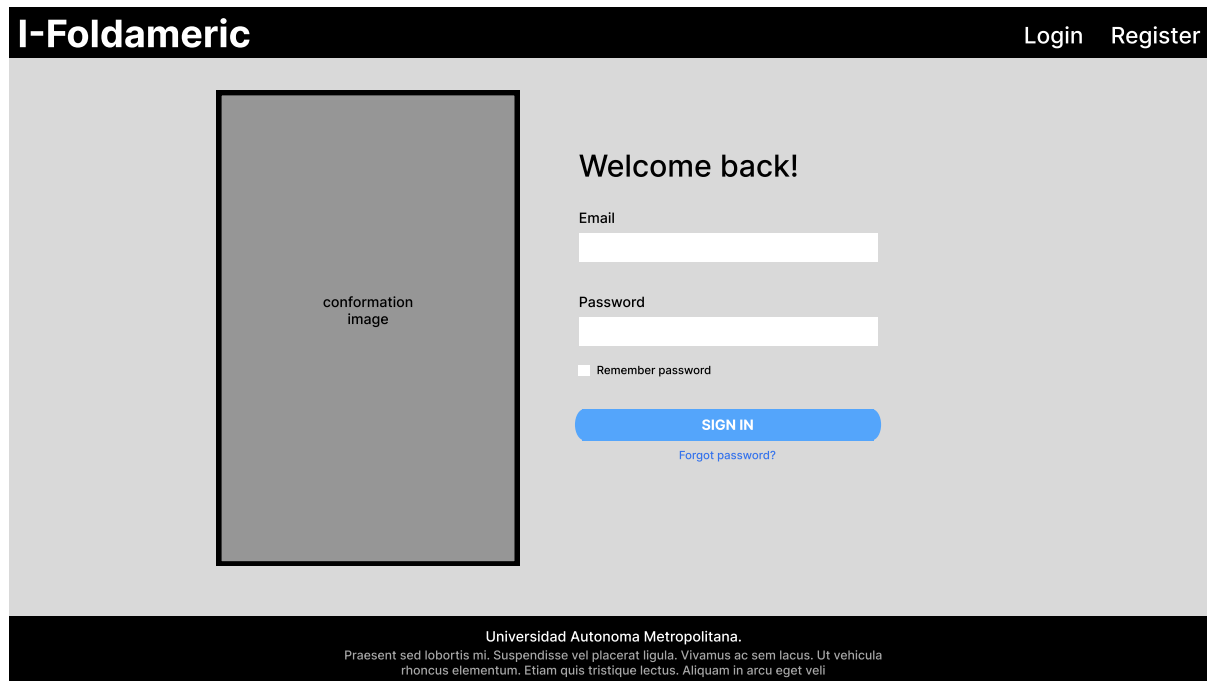


Figura 41: Boceto de la pantalla de login de I-Foldameric.

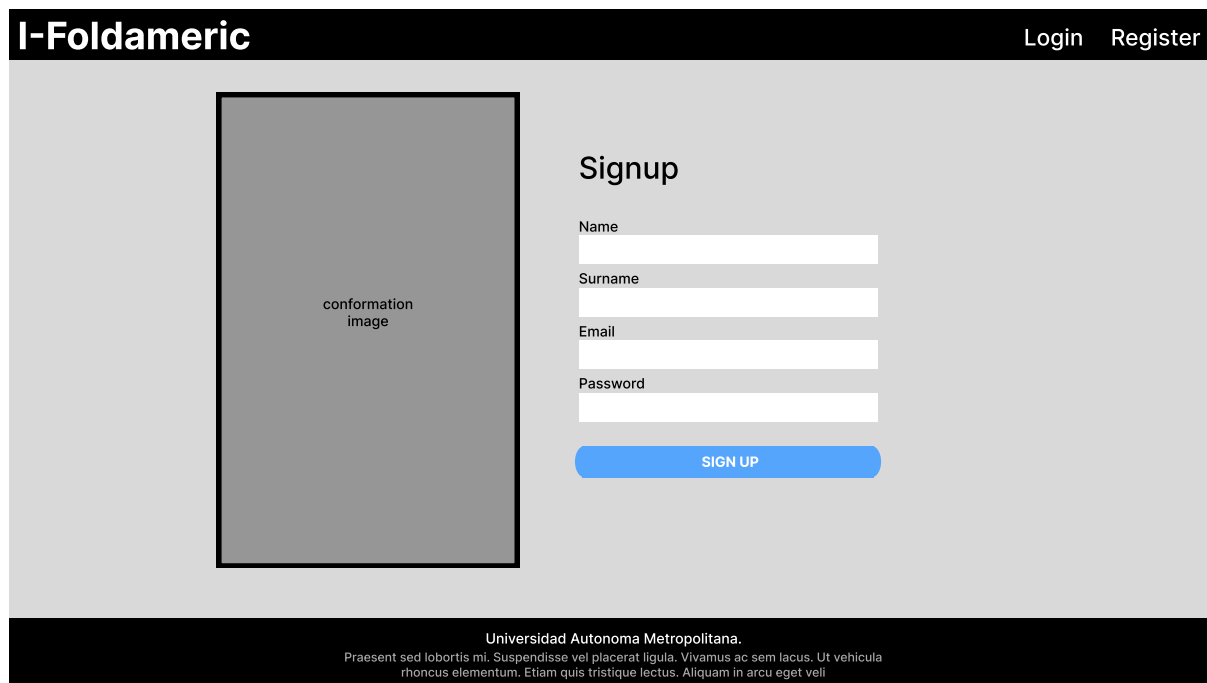


Figura 42: Refinamiento del boceto de la pantalla de registro de I-Foldameric.

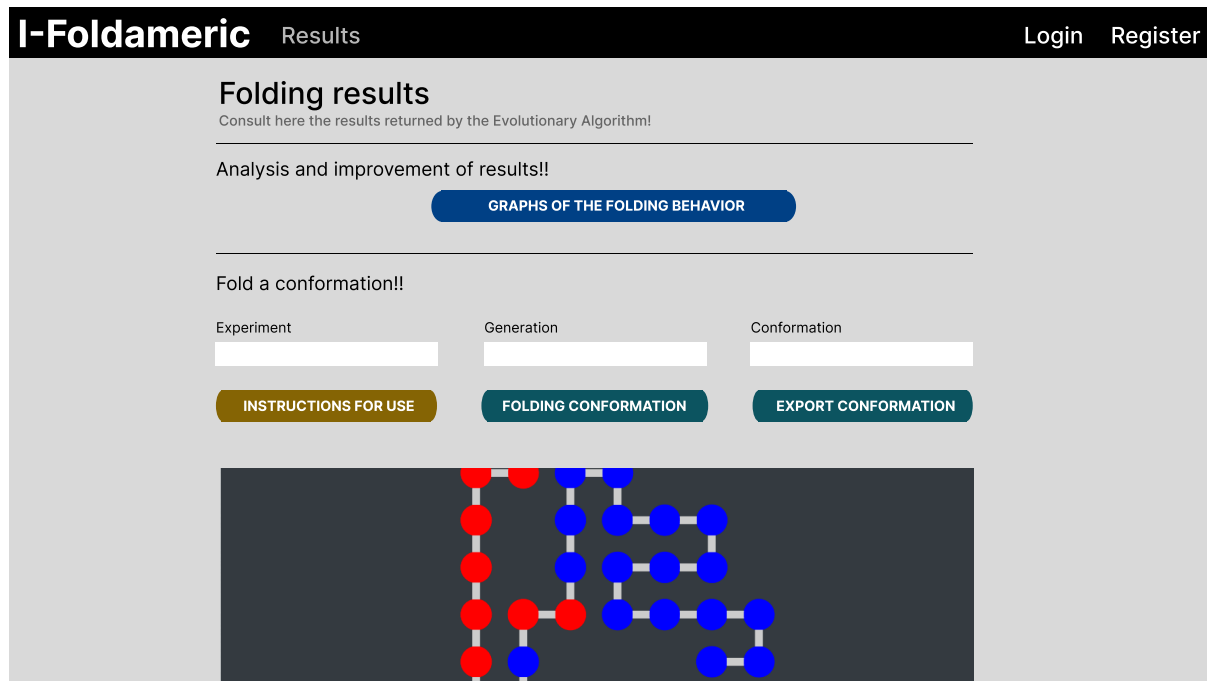


Figura 43: Refinamiento del boceto de la pantalla de resultados de I-Foldameric. (1)

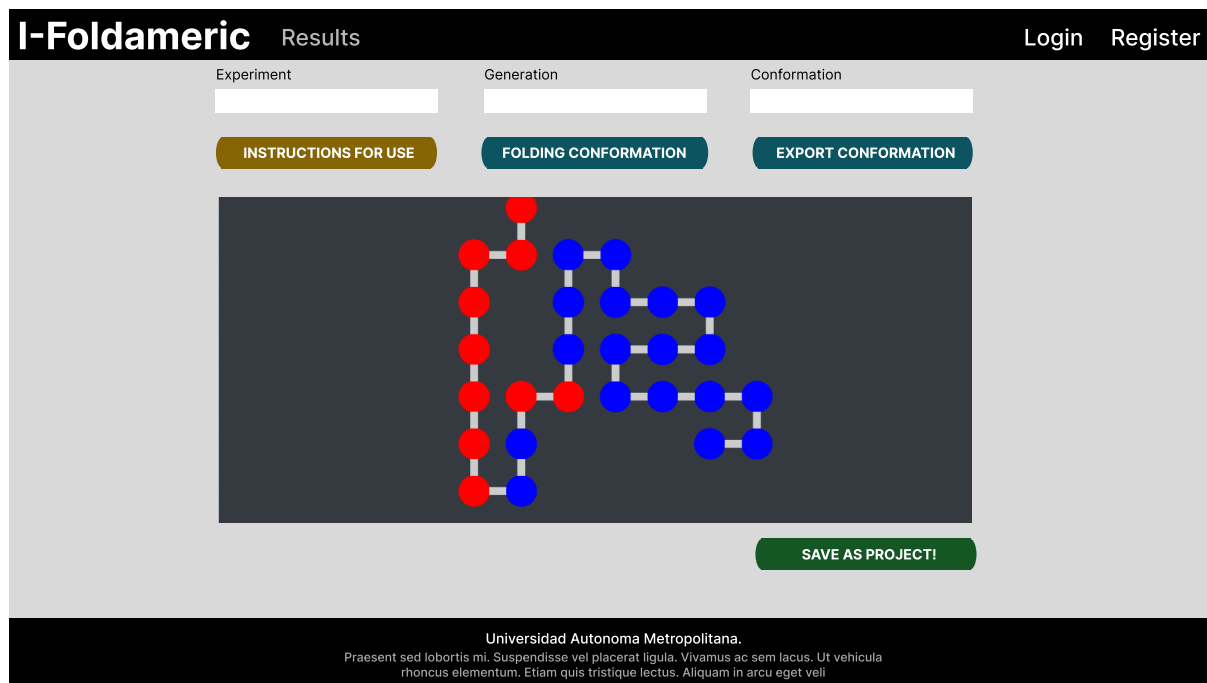


Figura 44: Refinamiento del boceto de la pantalla de resultados de I-Foldameric. (2)

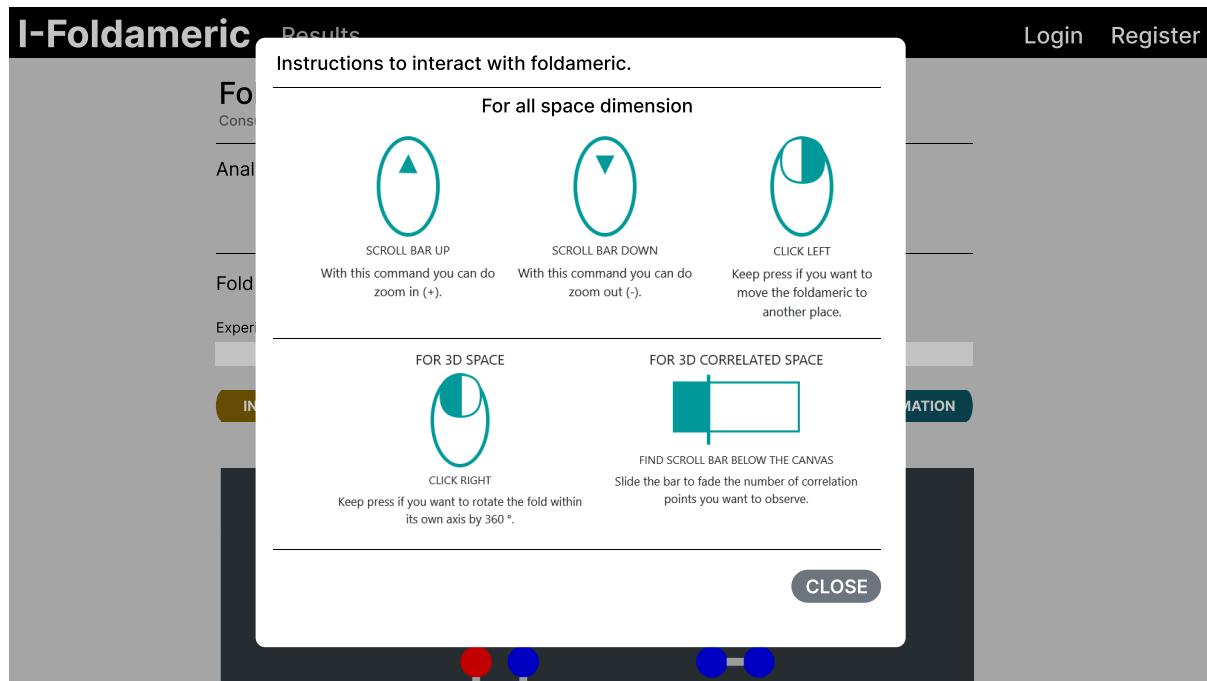


Figura 45: Refinamiento del boceto de la pantalla de instrucciones de I-Foldameric.

3. Fase de construcción

3.1. Diseño detallado

3.1.1. Identificación de métodos y atributos

EvolutionaryAlgorithm Manager	EA Simple
private \$params_array	private \$mutationProbability
construct(\$params_array) destruct() executeEvolutionaryAlgorithm() executeInjectNewMaterial()	construct(\$hpSequence, \$spaceType, \$dimensionType, \$fileNameCorrelatedNetwork, \$pointsCorrelatedNetworkSelected, \$selectionOperator, \$percentOfTournament, \$percentOfTopPercent, \$crossoverType, \$crossoverProbability, \$mutationType, \$mutationProbability, \$isKnowBestFitness, \$fitnessValue, \$conformationsNumber, \$generationsNumber, \$experimentsNumber, \$sampling, \$elitismSelected, \$percentOfElitism, \$functionType, \$alphaValue, \$injectNewMaterial = null, \$percentageToReplace = null, \$generationToInject = null, \$lengthMatrix = null) executeVersion1() executeVersion2()
Evolutionary Algorithm	Generate Points
protected \$hpSequence protected \$spaceType protected \$dimensionType protected \$fileNameCorrelatedNetwork protected \$pointsCorrelatedNetworkSelected protected \$correlatedMatrix protected \$startingPoint protected \$selectionOperator protected \$percentOfTournament protected \$percentOfTopPercent protected \$crossoverType protected \$crossoverProbability protected \$mutationType protected \$isKnowBestFitness protected \$fitnessValue protected \$conformationsNumber protected \$generationsNumber protected \$experimentsNumber protected \$sampling protected \$appliedSampling protected \$elitismSelected protected \$percentOfElitism protected \$functionType protected \$alphaValue protected \$currentExperiment = array() protected \$experiments = array() protected \$responseOfEA protected \$generationsJson = array() protected \$experimentsJson = array() protected \$injectNewMaterial protected \$percentageToReplace protected \$generationToInject protected \$lengthMatrix	protected \$hpSequence; protected \$hpLength; protected \$points; protected \$rand; protected \$typeSpace; protected \$correlatedMatrix; protected \$startingPoint; protected \$dimension_type; protected \$function_type; protected \$alphaValue; protected \$lengthMatrix;
getAppliedSampling() getValueSampling() getResponseOfEA() calculateSampling() getExperiments() saveGenerationsJson() saveExperimentsJson() getExperimentsJson() executeVersion1() executeVersion2() executeAlgorithm()	construct(\$hpString, \$typeSpace, \$correlatedMatrix, \$dimension_type, \$function_type, \$alphaValue, \$startingPoint = null, \$lengthMatrix=null) destruct() initializeGeneration(\$conformationsNumbers) generatePoints(\$pointsChildren) doPoints(\$childPoints_C, \$i)

Figura 46: Identificación de métodos y atributos. (1)

<table><tr><th>Generate Square Points</th></tr><tr><td>.</td></tr><tr><td>doPoints(\$pointsChildren_C, \$i)</td></tr></table>	Generate Square Points	.	doPoints(\$pointsChildren_C, \$i)	<table><tr><th>Fitness2DTriangle</th></tr><tr><td>private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0</td></tr><tr><td>construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)</td></tr></table>	Fitness2DTriangle	private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0	construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)
Generate Square Points							
.							
doPoints(\$pointsChildren_C, \$i)							
Fitness2DTriangle							
private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0							
construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)							
<table><tr><th>Generate Cube Points</th></tr><tr><td>.</td></tr><tr><td>doPoints(\$pointsChildren_C, \$i)</td></tr></table>	Generate Cube Points	.	doPoints(\$pointsChildren_C, \$i)				
Generate Cube Points							
.							
doPoints(\$pointsChildren_C, \$i)							
<table><tr><th>Generate Triangle Points</th></tr><tr><td>.</td></tr><tr><td>doPoints(\$pointsChildren_C, \$i)</td></tr></table>	Generate Triangle Points	.	doPoints(\$pointsChildren_C, \$i)	<table><tr><th>Fitness3DCubic</th></tr><tr><td>private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0</td></tr><tr><td>construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)</td></tr></table>	Fitness3DCubic	private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0	construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)
Generate Triangle Points							
.							
doPoints(\$pointsChildren_C, \$i)							
Fitness3DCubic							
private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0							
construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)							
<table><tr><th>Fitness</th></tr><tr><td>private \$points = array() private \$dimension_type private \$function_type private \$alphaValue</td></tr><tr><td>construct(\$points, \$dimension_type, \$function_type, \$alphaValue) destruct() getFitness()</td></tr></table>	Fitness	private \$points = array() private \$dimension_type private \$function_type private \$alphaValue	construct(\$points, \$dimension_type, \$function_type, \$alphaValue) destruct() getFitness()	<table><tr><th>Conformation</th></tr><tr><td>private \$fitness private \$positionIndex private \$points private \$parents private \$radiusGyrationCM = array() private \$radiusGyrationRCDM private \$Dmax</td></tr><tr><td>construct(\$points) destruct() getFitness() setFitness(\$fitness) setFitnessTo0() setPositionIndex(\$position) getPositionIndex() getPoints() getParents() setParents(\$parents) getPoints.Json() genRadGyrCM() genRadGyrRCDM() genDmax() getRadiusGyrationCM() getRadiusGyrationRCDM() getDmax()</td></tr></table>	Conformation	private \$fitness private \$positionIndex private \$points private \$parents private \$radiusGyrationCM = array() private \$radiusGyrationRCDM private \$Dmax	construct(\$points) destruct() getFitness() setFitness(\$fitness) setFitnessTo0() setPositionIndex(\$position) getPositionIndex() getPoints() getParents() setParents(\$parents) getPoints.Json() genRadGyrCM() genRadGyrRCDM() genDmax() getRadiusGyrationCM() getRadiusGyrationRCDM() getDmax()
Fitness							
private \$points = array() private \$dimension_type private \$function_type private \$alphaValue							
construct(\$points, \$dimension_type, \$function_type, \$alphaValue) destruct() getFitness()							
Conformation							
private \$fitness private \$positionIndex private \$points private \$parents private \$radiusGyrationCM = array() private \$radiusGyrationRCDM private \$Dmax							
construct(\$points) destruct() getFitness() setFitness(\$fitness) setFitnessTo0() setPositionIndex(\$position) getPositionIndex() getPoints() getParents() setParents(\$parents) getPoints.Json() genRadGyrCM() genRadGyrRCDM() genDmax() getRadiusGyrationCM() getRadiusGyrationRCDM() getDmax()							
<table><tr><th>Fitness2DSquare</th></tr><tr><td>private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0</td></tr><tr><td>construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)</td></tr></table>	Fitness2DSquare	private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0	construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)				
Fitness2DSquare							
private \$points = array() private \$actualPosition = 0 private \$nextPosition = 0 private \$alphaHH = 0.0 private \$alphaPP = 0.0							
construct(\$points) getFitnessDillModel() getFitnessConvexFunction(\$alphaValue) getFitnessConvexFunctionPolar(\$alphaValue)							

Figura 47: Identificación de métodos y atributos. (2)

Crossover Operator protected \$generation protected \$typeSpace protected \$typeDimension protected \$lengthHpString protected \$conformationsNumber protected \$crossoverProbability protected \$correlatedMatrix protected \$origenX protected \$origenY protected \$hpSecuence protected \$mutationType protected \$newGeneration protected \$functionType protected \$alphaValue protected \$lengthMatrix construct(\$generation, \$typeSpace, \$typeDimension, \$lengthHpString, \$crossoverProbability, \$correlatedMatrix, \$hpSecuence, \$mutationType, \$functionType, \$alphaValue, \$origenX = null, \$origenY = null, \$lengthMatrix = null) destruct() getNewGeneration() execute(\$pointsParentOne, \$pointsParentTwo, \$newChildrenOne, \$newChildrenTwo, \$pointsChildren_C) getChildrenOne() getChildrenTwo() make_seed() decimalRandom() checkSquareChildren(&\$pointsChildren_C, \$movVectorValue, &\$pointsChildren, \$j) checkTriangleChildren(&\$pointsChildren_C, \$movVectorValue, &\$pointsChildren, \$j) checkCubeChildren(&\$pointsChildren_C, \$movVectorValue, &\$pointsChildren, \$j)	One Point . execute(\$pointsParentOne, \$pointsParentTwo, \$newChildrenOne, \$newChildrenTwo, \$pointsChildren_C) execute23(\$lengthHpString)
	Two Points . execute(\$pointsParentOne, \$pointsParentTwo, \$newChildrenOne, \$newChildrenTwo, \$pointsChildren_C) execute23(\$lengthHpString)
	Uniform . execute(\$pointsParentOne, \$pointsParentTwo, \$newChildrenOne, \$newChildrenTwo, \$pointsChildren_C)
	Couple Formation private \$generation coupleFormation() make_seed()
RankGACoupleFormation . construct(\$generation) coupleFormation()	SimplexCoupleFormation . construct(\$generation) coupleFormation()

Figura 48: Identificación de métodos y atributos. (3)

<table><tr><th>Selection Operator</th></tr><tr><td>protected \$generation</td></tr><tr><td>execute() make_seed() decimalRandom()</td></tr></table>	Selection Operator	protected \$generation	execute() make_seed() decimalRandom()	<table><tr><th>Random</th></tr><tr><td>.</td></tr><tr><td>execute(\$generation, \$i) make_seed()</td></tr></table>	Random	.	execute(\$generation, \$i) make_seed()
Selection Operator							
protected \$generation							
execute() make_seed() decimalRandom()							
Random							
.							
execute(\$generation, \$i) make_seed()							
<table><tr><th>Roulette</th></tr><tr><td>.</td></tr><tr><td>construct(\$generation) execute(\$conformationsToSelectParemeter=null)</td></tr></table>	Roulette	.	construct(\$generation) execute(\$conformationsToSelectParemeter=null)	<table><tr><th>Predefined</th></tr><tr><td>.</td></tr><tr><td>execute(\$generation, \$i)</td></tr></table>	Predefined	.	execute(\$generation, \$i)
Roulette							
.							
construct(\$generation) execute(\$conformationsToSelectParemeter=null)							
Predefined							
.							
execute(\$generation, \$i)							
<table><tr><th>Top Percent</th></tr><tr><td>private \$percent</td></tr><tr><td>construct(\$generation, \$percent) execute(\$conformationsToSelectParemeter=null)</td></tr></table>	Top Percent	private \$percent	construct(\$generation, \$percent) execute(\$conformationsToSelectParemeter=null)	<table><tr><th>Elitism</th></tr><tr><td>public \$percentOfElitism public \$generation public \$selectionOperator public \$percentOfSelectionOperator public \$selectedConformations = array() construct(\$percentOfElitism, \$generation, \$selectionOperator, \$percentOfSelectionOperator) execute() caseRoulette(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) caseTournament(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) caseTopPercent(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) casePopulationDecimation(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations)</td></tr></table>	Elitism	public \$percentOfElitism public \$generation public \$selectionOperator public \$percentOfSelectionOperator public \$selectedConformations = array() construct(\$percentOfElitism, \$generation, \$selectionOperator, \$percentOfSelectionOperator) execute() caseRoulette(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) caseTournament(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) caseTopPercent(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) casePopulationDecimation(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations)	
Top Percent							
private \$percent							
construct(\$generation, \$percent) execute(\$conformationsToSelectParemeter=null)							
Elitism							
public \$percentOfElitism public \$generation public \$selectionOperator public \$percentOfSelectionOperator public \$selectedConformations = array() construct(\$percentOfElitism, \$generation, \$selectionOperator, \$percentOfSelectionOperator) execute() caseRoulette(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) caseTournament(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) caseTopPercent(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations) casePopulationDecimation(\$sizeGeneration, \$percentConformationsElitism, &\$indexSelectedConformations)							
<table><tr><th>Tournament</th></tr><tr><td>private \$percent</td></tr><tr><td>construct(\$generation, \$percent) execute(\$conformationsToSelectParemeter=null)</td></tr></table>	Tournament	private \$percent	construct(\$generation, \$percent) execute(\$conformationsToSelectParemeter=null)				
Tournament							
private \$percent							
construct(\$generation, \$percent) execute(\$conformationsToSelectParemeter=null)							
<table><tr><th>Population Decimation</th></tr><tr><td>.</td></tr><tr><td>construct(\$generation) execute(\$conformationsToSelectParemeter=null)</td></tr></table>	Population Decimation	.	construct(\$generation) execute(\$conformationsToSelectParemeter=null)				
Population Decimation							
.							
construct(\$generation) execute(\$conformationsToSelectParemeter=null)							
<table><tr><th>Mutation Operator</th></tr><tr><td>.</td></tr><tr><td>execute(\$generation, \$i)</td></tr></table>	Mutation Operator	.	execute(\$generation, \$i)	<table><tr><th>Parents</th></tr><tr><td>private \$parent1 private \$parent2 construct() destruct() getParent1() setParent1(\$parent1) getParent2() setParent2(\$parent2)</td></tr></table>	Parents	private \$parent1 private \$parent2 construct() destruct() getParent1() setParent1(\$parent1) getParent2() setParent2(\$parent2)	
Mutation Operator							
.							
execute(\$generation, \$i)							
Parents							
private \$parent1 private \$parent2 construct() destruct() getParent1() setParent1(\$parent1) getParent2() setParent2(\$parent2)							

Figura 49: Identificación de métodos y atributos. (4)

Correlated Network	Point
<pre> private \$fileName private \$correlatedMatrix = array() private \$upperLeft private \$upperRight private \$lowerLeft private \$lowerRight private \$startingPoint private \$width private \$height private \$amount = array() private \$density = array() private \$density2 = array() private \$x1 private \$y1 private \$higher = 0.0 private \$typeDimension private \$lengthMatrix construct(\$fileName, \$upperLeft, \$upperRight, \$lowerLeft, \$lowerRight, \$typeDimension = null, \$lengthMatrix=null) destruct() getCorrelatedMatrix() setCorrelatedMatrix(\$correlatedMatrix) getStartingPoint() readMatrix() generateStartingPoint() </pre>	<pre> private \$xValue private \$yValue private \$zValue private \$letter private \$movVectorValue private \$way0 = true private \$way1 = true private \$way2 = true private \$way3 = true private \$way4 = true private \$way5 = true construct(\$xValue, \$yValue, \$zValue, \$letter, \$movVectorValue) destruct() getValueX() setValueX(\$x) getValueY() setValueY(\$y) getValueZ() setValueZ(\$z) getLetter() getMovVectorValue() setMovVectorValue(\$movVectorValue) isWay0() setWay0(\$way0) isWay1() setWay1(\$way1) isWay2() setWay2(\$way2) isWay3() setWay3(\$way3) isWay4() setWay4(\$way4) isWay5() setWay5(\$way5) resetR() </pre>
Generation	
<pre> private \$conformations = array() private \$sizeGeneration private \$indexSelectedConformations = array() private \$parentsList = array() construct(\$conformations) destruct() calcPositionIndexAndSizeGeneration() getCloneGeneration() getCloneGeneration() setConformations(\$conformations) getTotalFitness() getSizeGeneration() getOrderedConformations(\$order) getIndexSelectedConformations() setIndexSelectedConformations(\$arrayIndex) getSelectedConformations() getParentsList() setParentsList(\$parentsList) convertToJson() </pre>	

Figura 50: Identificación de métodos y atributos. (5)

ProfileManager
private \$params private \$params_array
construct(\$params, \$params_array) register() login() updateUser(\$token, \$checkToken) upload(\$image) getImage(\$filename) detail(\$id) verifyCode(\$code)

User
protected \$fillable protected \$hidden protected \$casts
projects()

ProjectsManager
.
index() show(\$id) store(\$params, \$params_array, \$token) updateProject(\$id, \$params_array, \$token) destroyProject(\$id, \$token) uploadImage(\$params_array, \$image) getImage(\$filename) getProjectsByUser(\$id) getIdentity(\$token)

Project
protected \$table = 'projects' protected \$fillable
user()

Results
protected \$table = 'results' protected \$fillable
.

Evolutionary Algorithm Controller
.
execute(Request \$request) injectNewMaterial(Request \$request)

Project Controller
.
construct() index() show(\$id) store(Request \$request) update(\$id, Request \$request) destroy(\$id, Request \$request) upload(Request \$request) getImage(\$filename) getProjectsByUser(\$id)

Profile Controller
.
register(Request \$request) login(Request \$request) update(Request \$request) upload(Request \$request) getImage(\$filename) detail(\$id) verifyCode(\$code)

Figura 51: Identificación de métodos y atributos. (6)

3.2. Administración de archivos

En la **Figura 52** se presenta el directorio raíz del sistema I-Foldameric. En el directorio raíz, se presentan dos carpetas, donde se hace la separación del subsistema frontend y el subsistema backend.

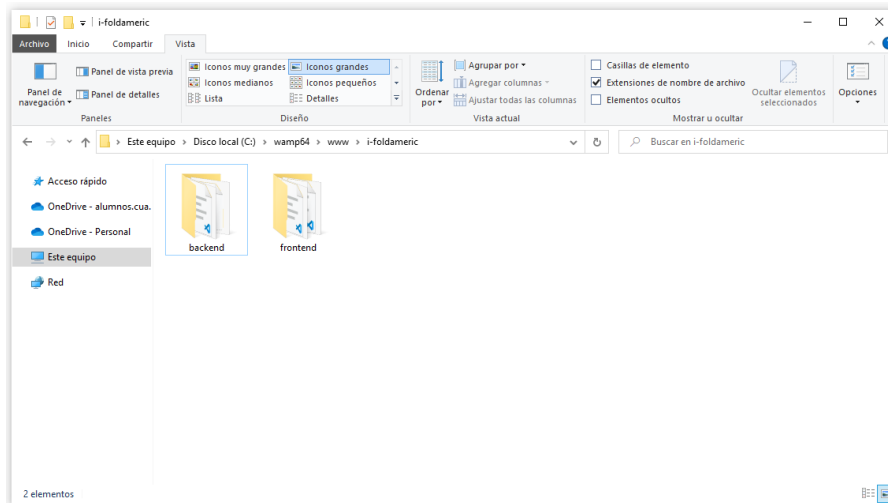


Figura 52: Directorio raíz del sistema I-Foldameric.

En la **Figura 53**, se presenta la raíz del directorio backend del sistema I-Foldameric.

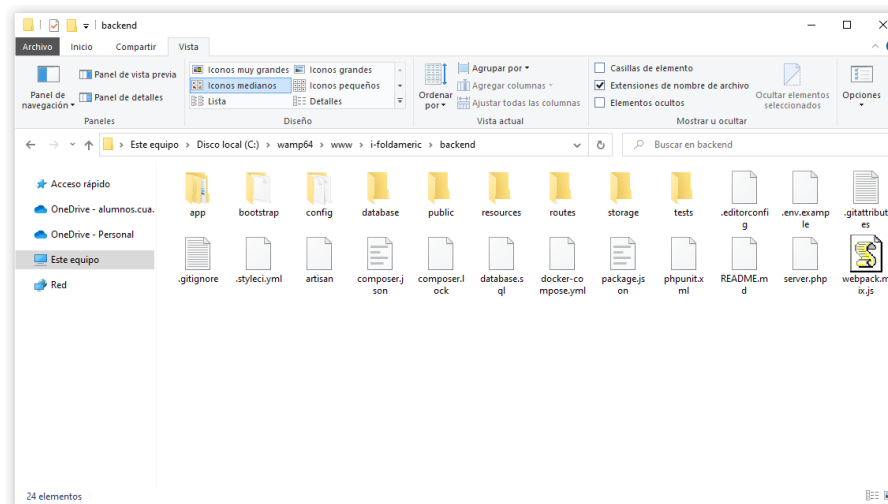


Figura 53: Directorio raíz del backend de I-Foldameric.

En la **Figura 54** podemos ver el contenido de la carpeta app, donde se resaltan las carpetas Http y Models.

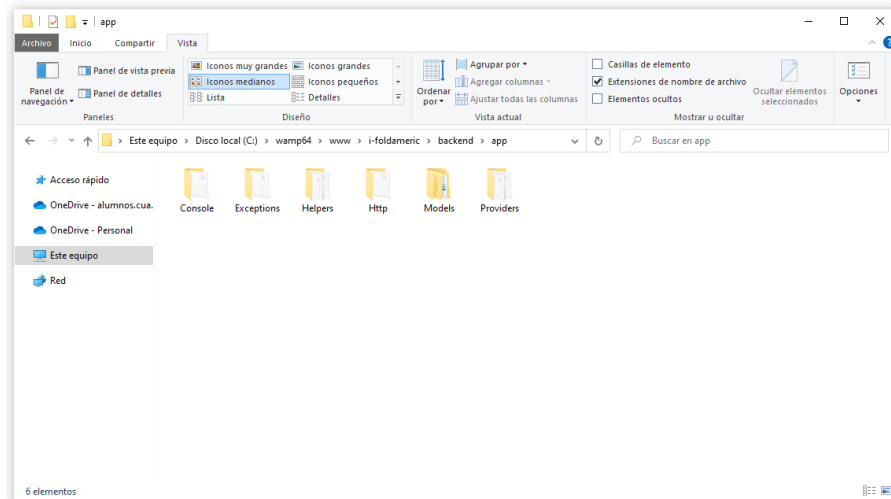


Figura 54: Directorio app de la carpeta backend.

En la **Figura 55** se observa la carpeta Controllers del sistema I-FoldamERIC.

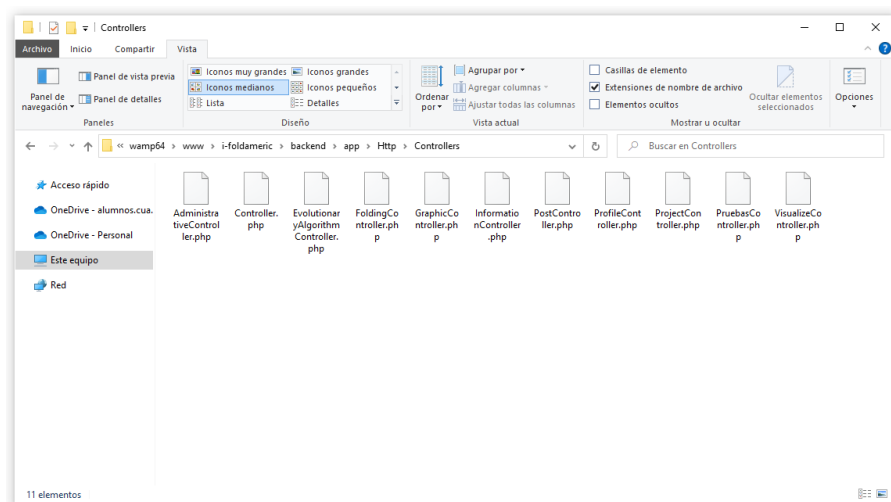


Figura 55: Directorio de los controladores.

En la **Figura 56** tenemos el directorio raíz Models, donde se encuentran toda la lógica de negocio de acuerdo a la presente documentación.

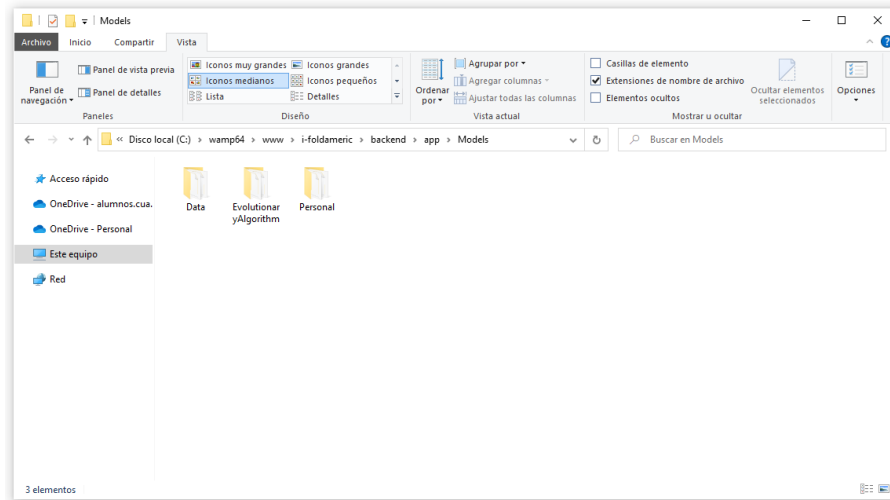


Figura 56: Directorio Models.

En las **Figuras 57, 58 y 67**, se presentan los paquetes del backend de acuerdo a la organización presentada en esta documentación.

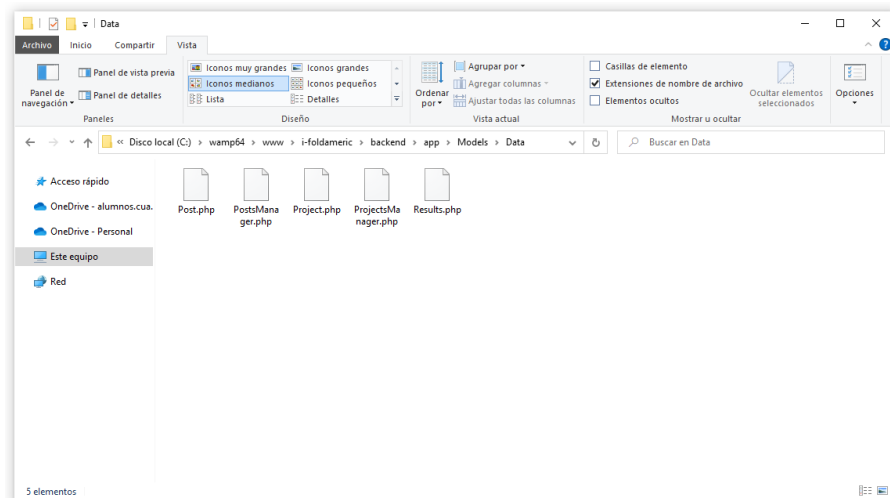


Figura 57: Directorio Data del modelo.

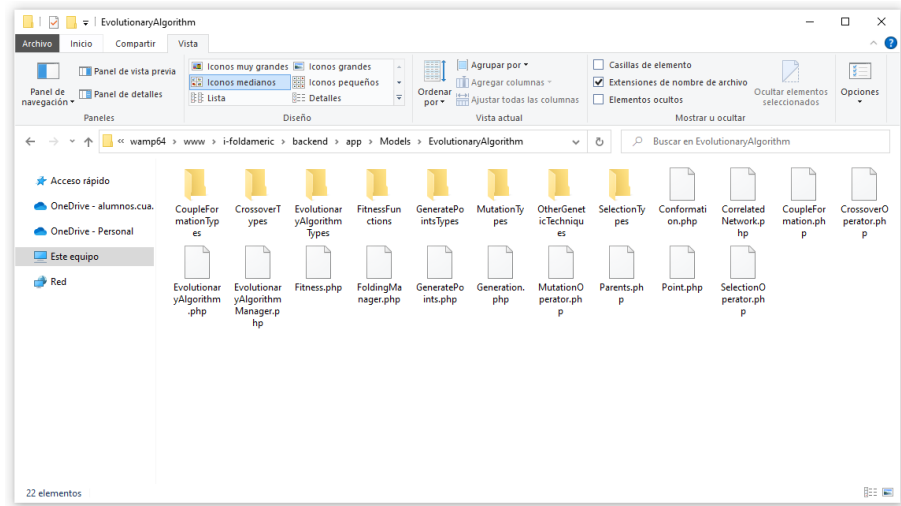


Figura 58: Directorio EvolutionaryAlgorithm del modelo.

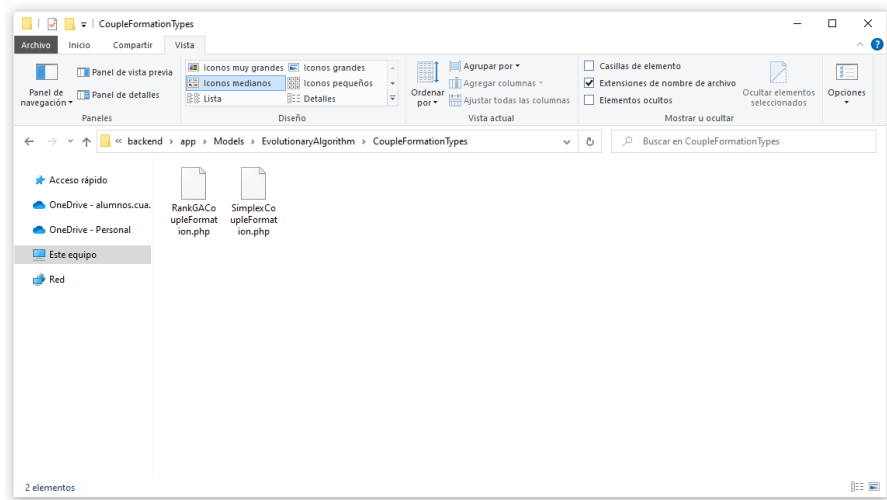


Figura 59: Directorio CoupleFormationTypes del paquete EvolutionaryAlgorithm.

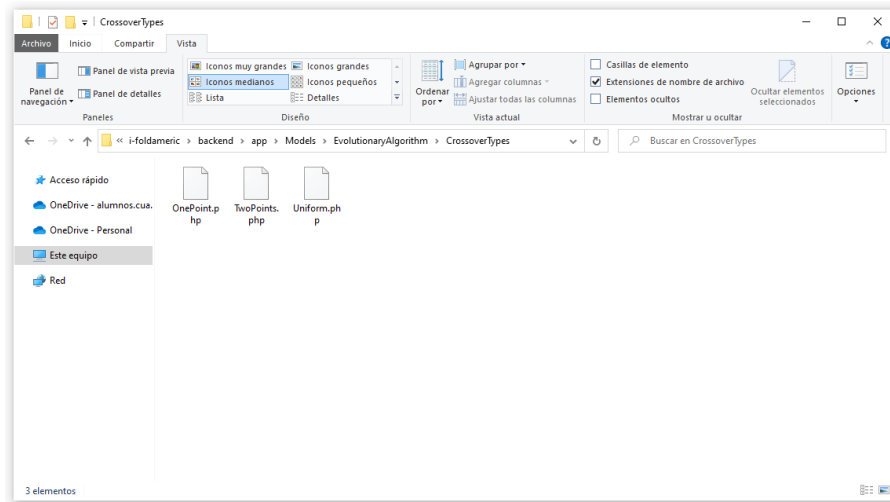


Figura 60: Directorio CrossoverTypes del paquete EvolutionaryAlgorithm.

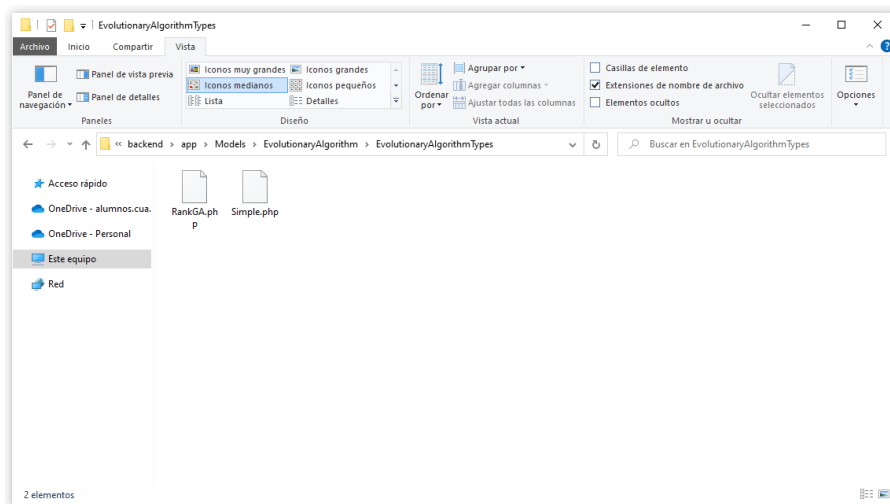


Figura 61: Directorio EvolutionaryAlgorithmTypes del paquete EvolutionaryAlgorithm.

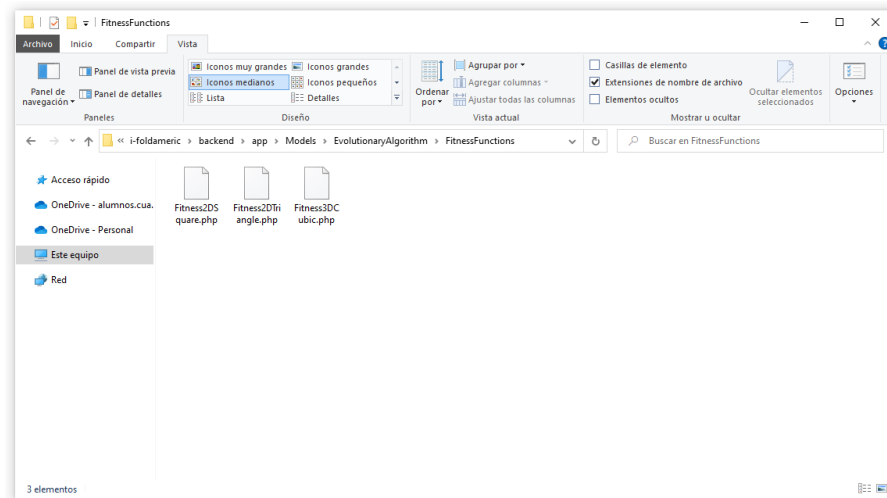


Figura 62: Directorio FitnessFunctions del paquete EvolutionaryAlgorithm.

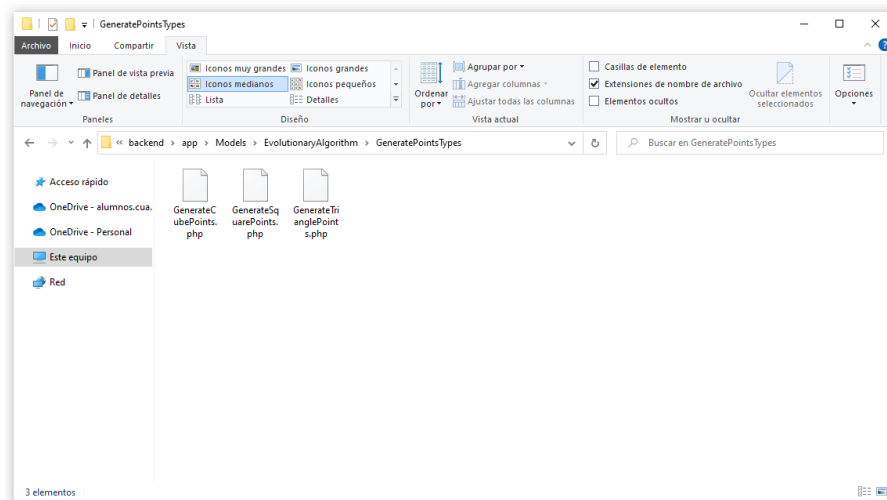


Figura 63: Directorio GeneratePointsTypes del paquete EvolutionaryAlgorithm.

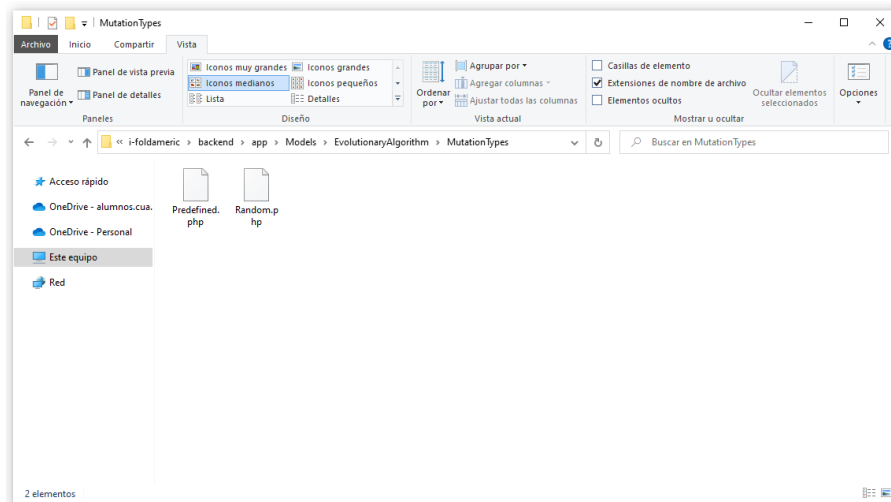


Figura 64: Directorio MutationTypes del paquete EvolutionaryAlgorithm.

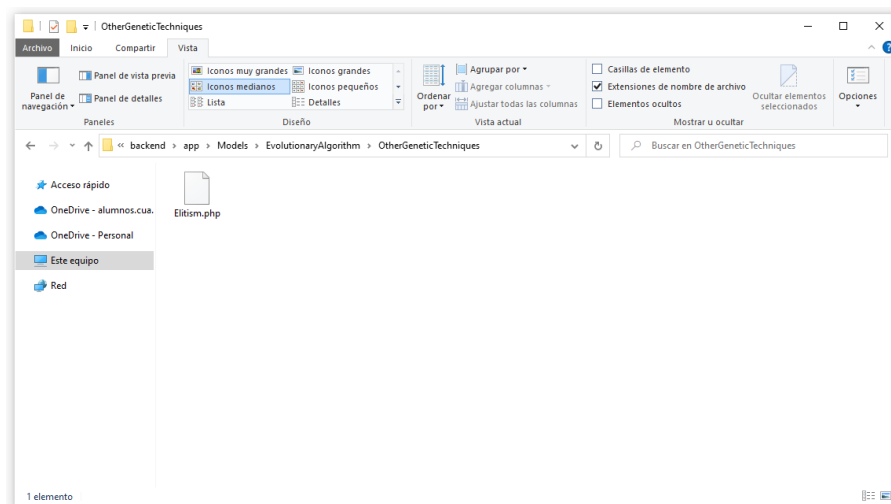


Figura 65: Directorio OtherGeneticTechniques del paquete EvolutionaryAlgorithm.

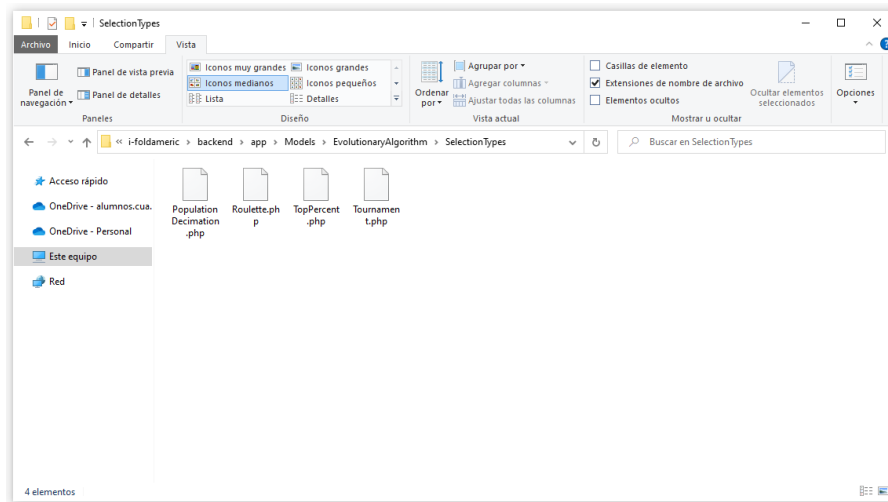


Figura 66: Directorio SelectionTypes del paquete EvolutionaryAlgorithm.

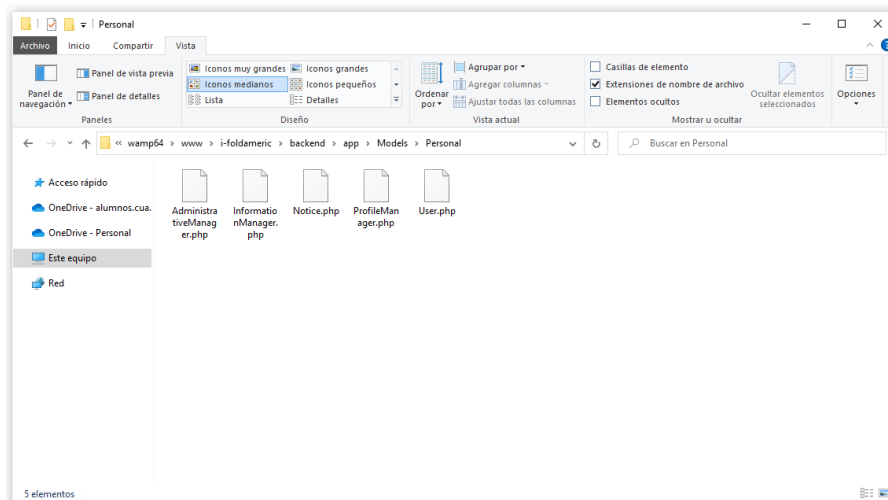


Figura 67: Directorio Personal del modelo.

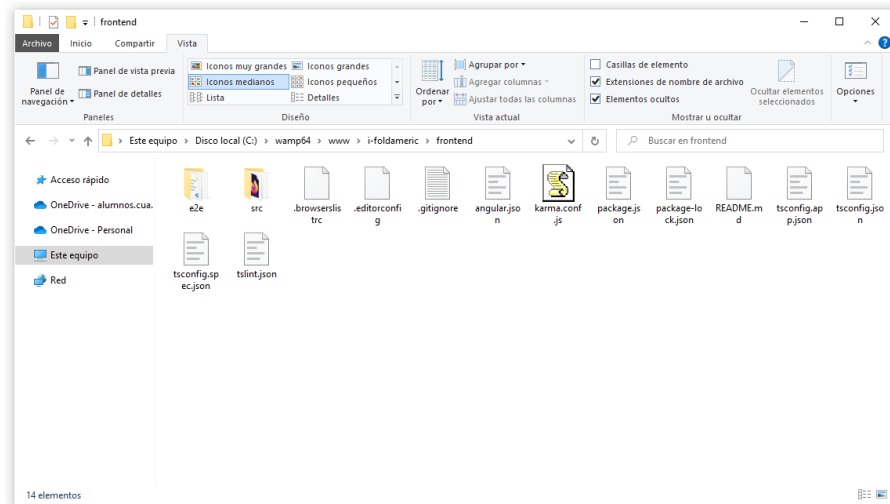


Figura 68: Directorio raíz del subsistema frontend.

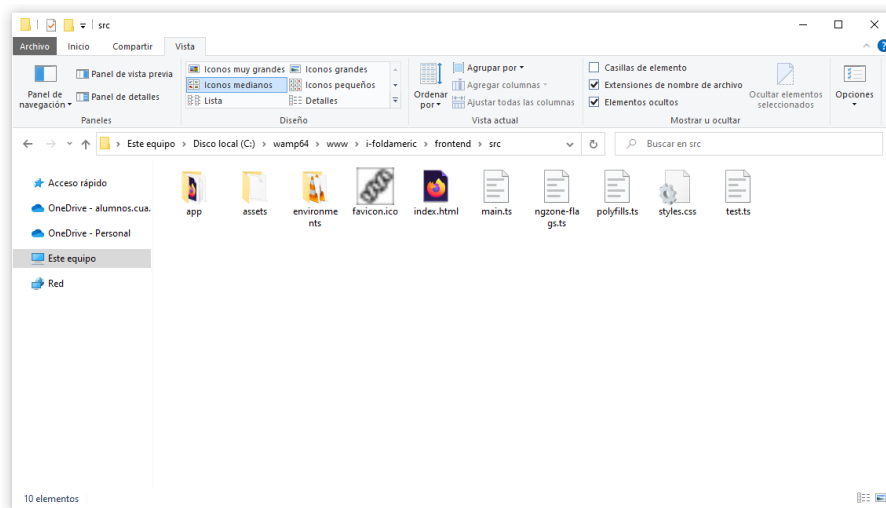


Figura 69: Directorio src del subsistema frontend.

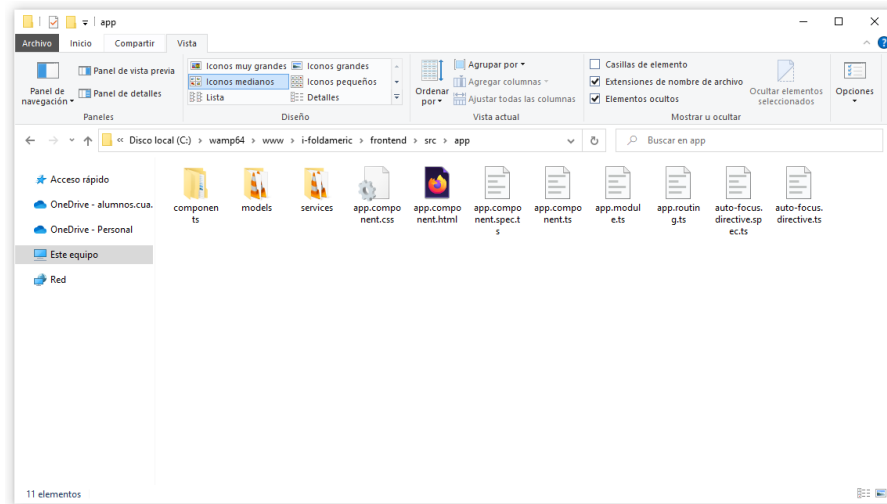


Figura 70: Directorio app.

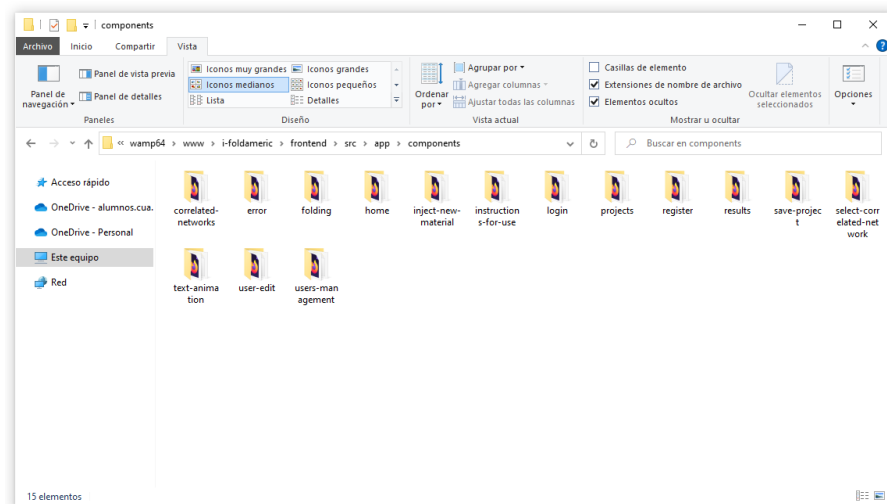


Figura 71: Directorio raíz de los componentes del frontend.

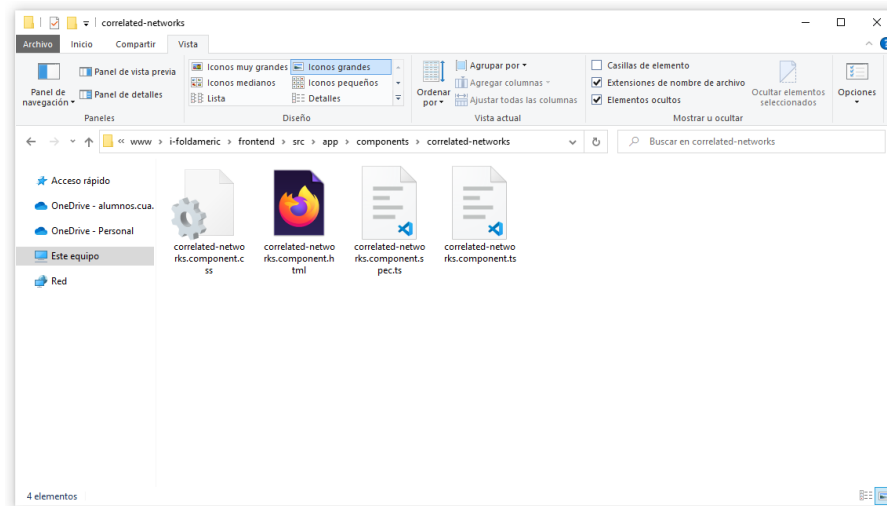


Figura 72: Directorio del componente correlated-networks.

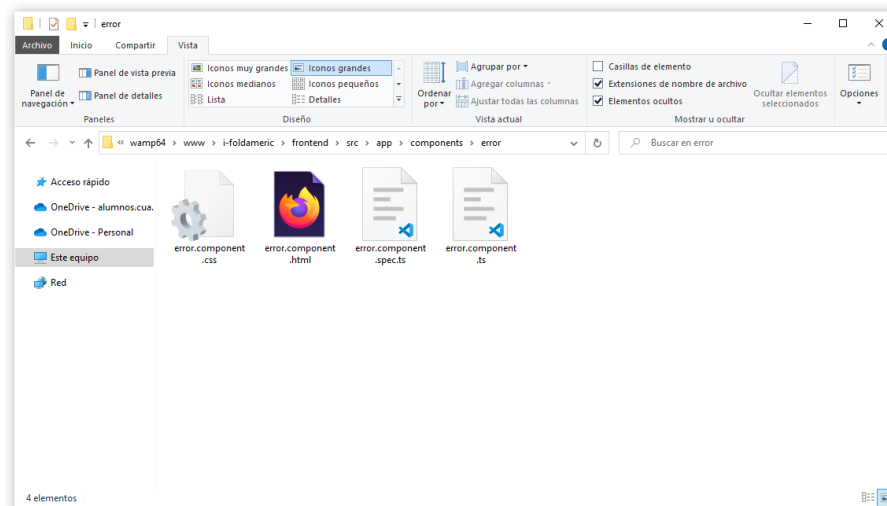


Figura 73: Directorio del componente error.

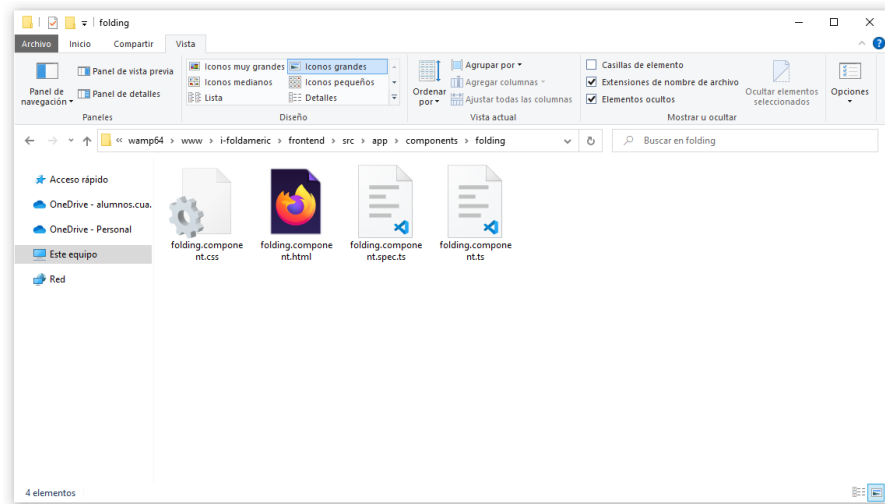


Figura 74: Directorio del componente folding.

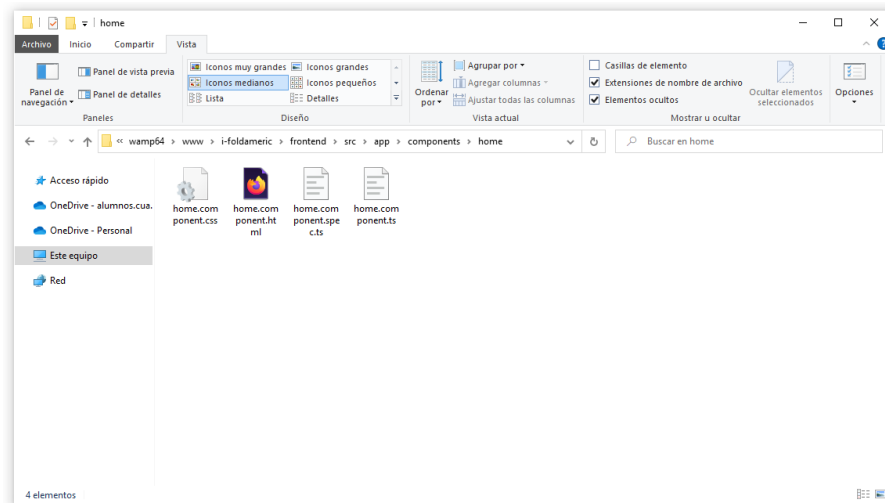


Figura 75: Directorio del componente home.

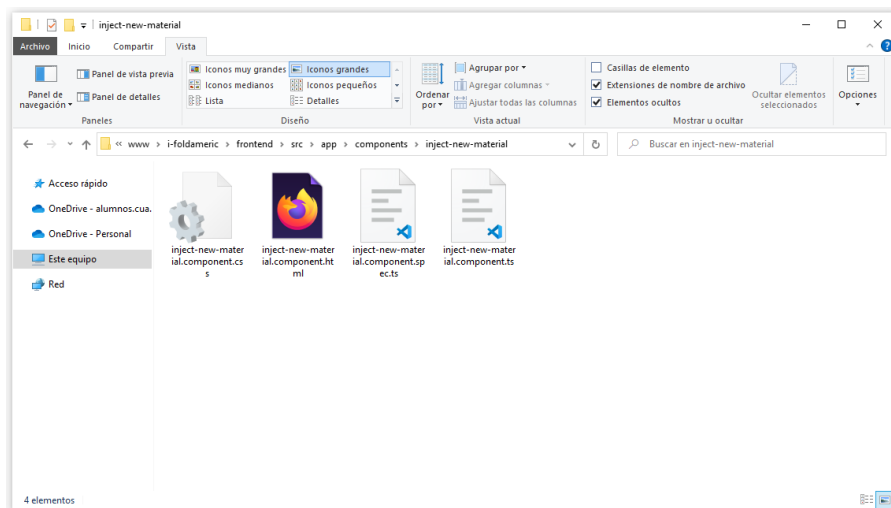


Figura 76: Directorio del componente inject-new-material.

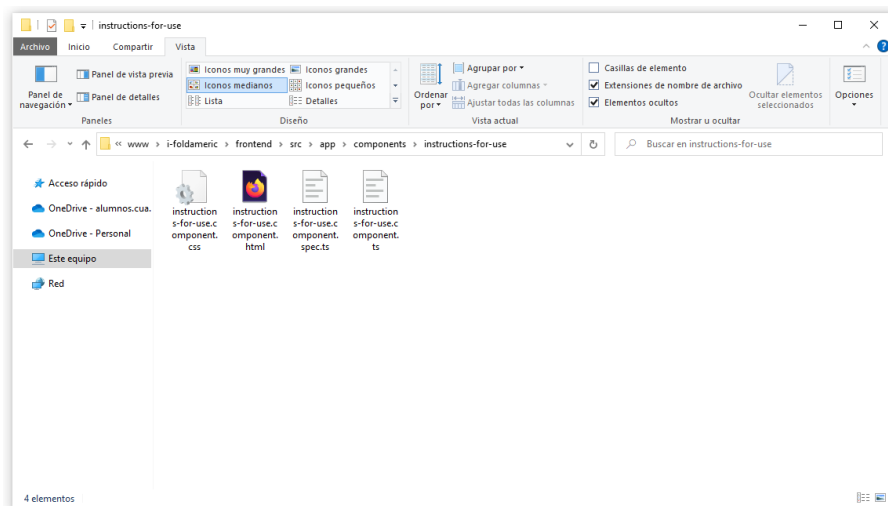


Figura 77: Directorio del componente instructions-for-use.

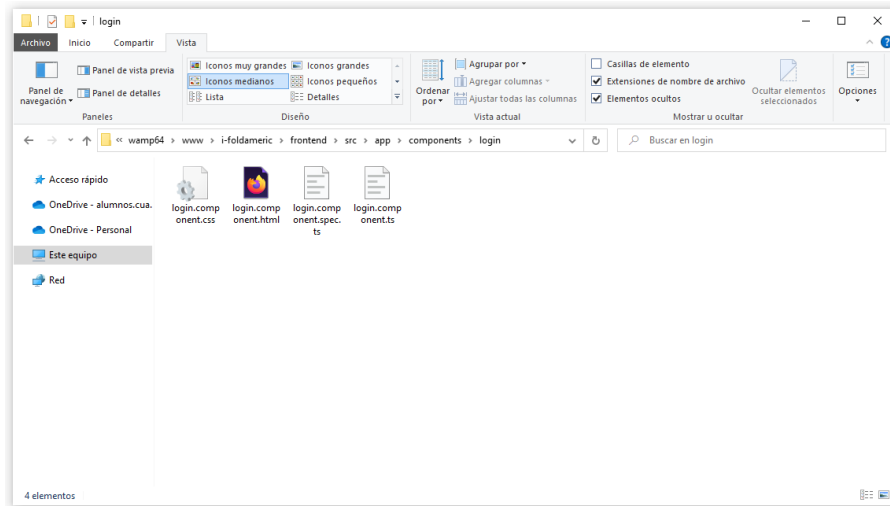


Figura 78: Directorio del componente login.

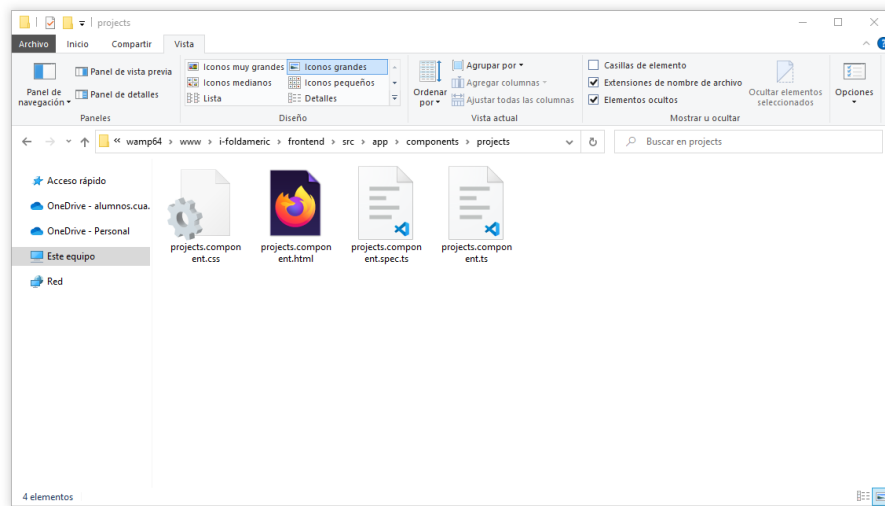


Figura 79: Directorio del componente projects.

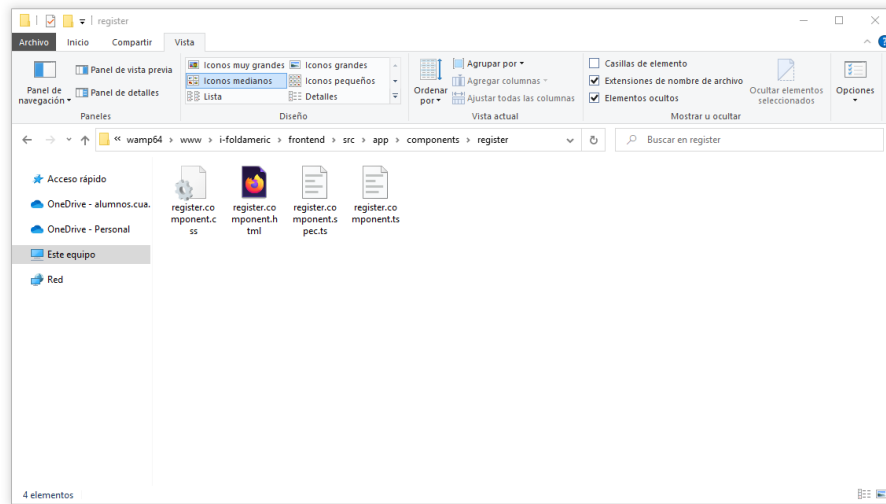


Figura 80: Directorio del componente register.

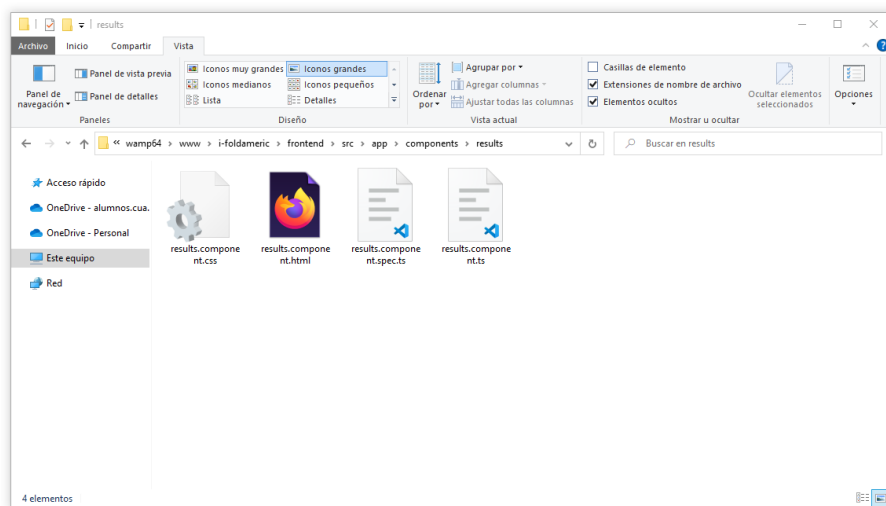


Figura 81: Directorio del componente results.

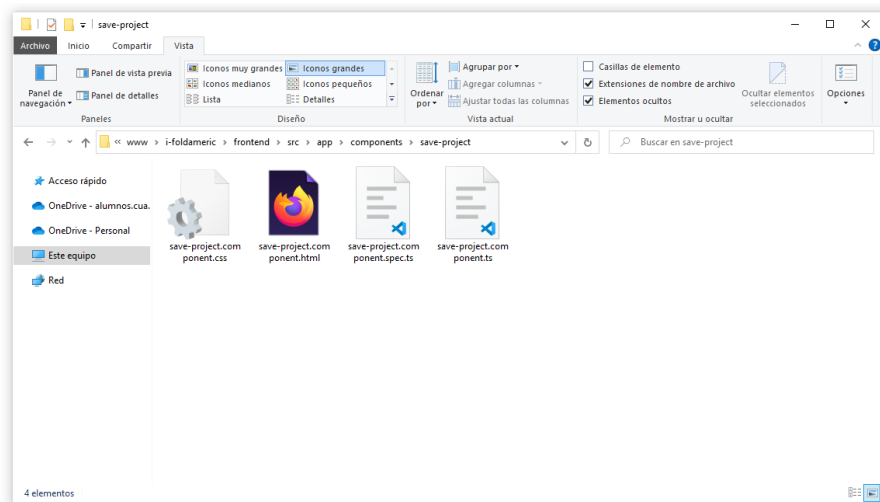


Figura 82: Directorio del componente save-project.

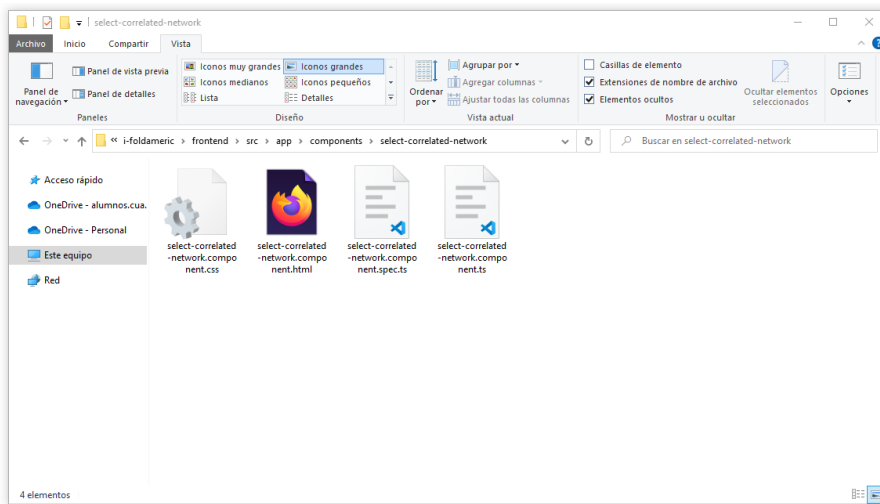


Figura 83: Directorio del componente select-correlated-network.

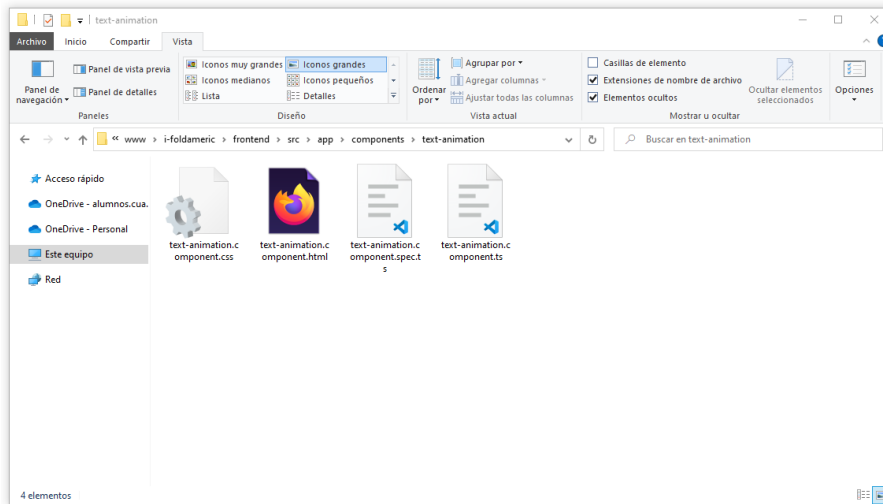


Figura 84: Directorio del componente text-animation.

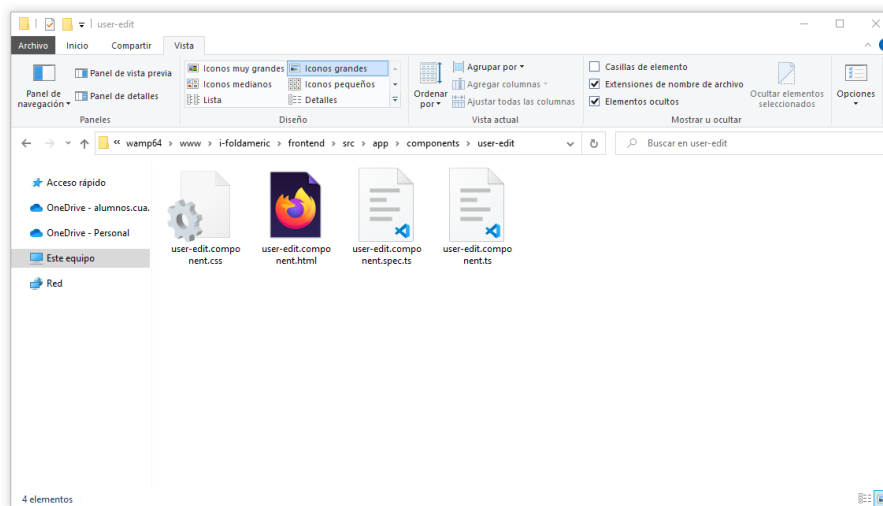


Figura 85: Directorio del componente user-edit.

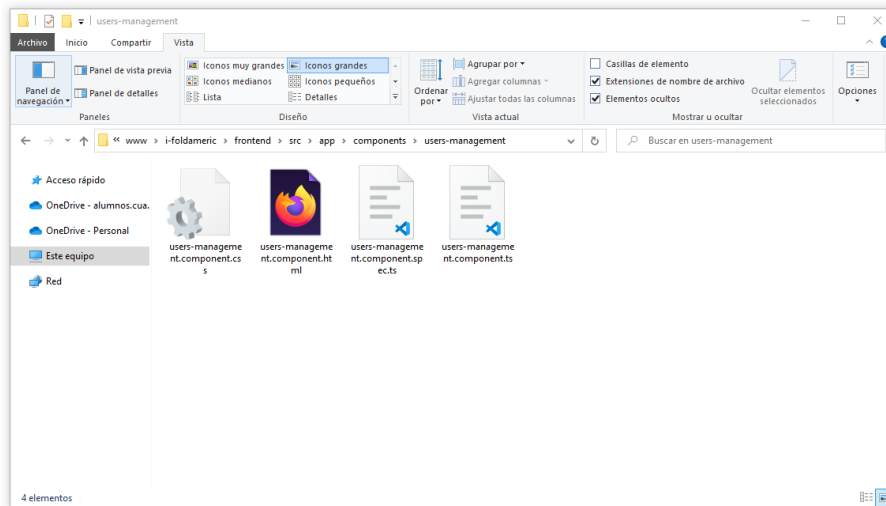


Figura 86: Directorio del componente users-management.

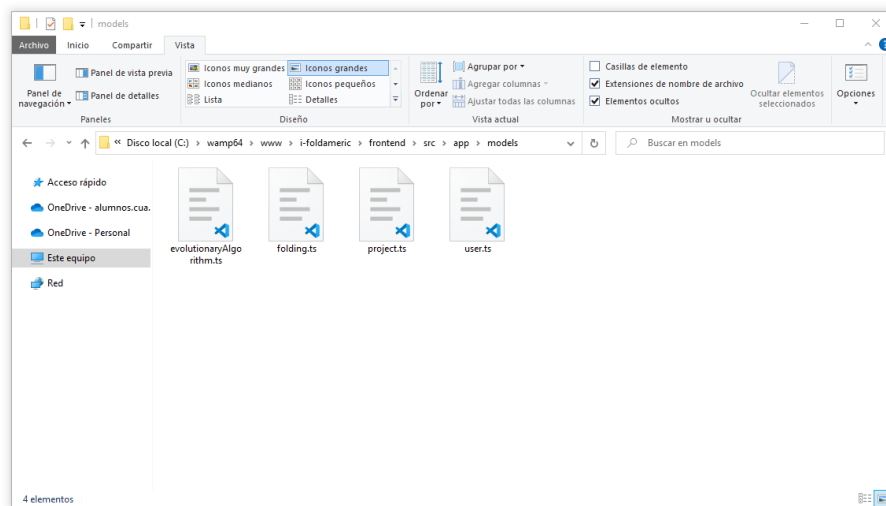


Figura 87: Directorio del modelo en el subsistema frontend.

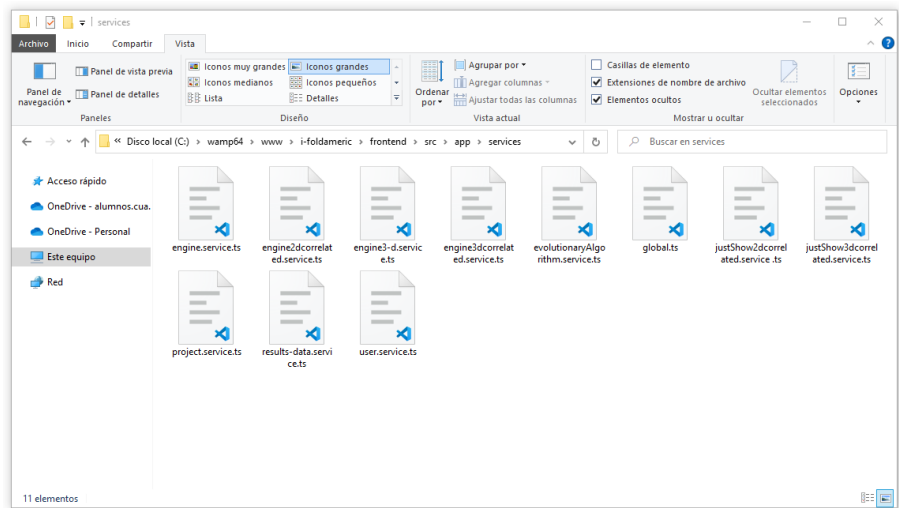


Figura 88: Directorio de los servicios del subsistema frontend.

3.3. Modelo de implementación

En la **Figura 89**, se observa el código HTML del formulario, donde el usuario introduce los parámetros para su posterior procesamiento por el algoritmo evolutivo en la lógica de negocio.

```
frontend > src > app > components > folding > folding.component.html > div.container-fluid.mt-3
1 <div class="container-fluid mt-3">
2
3   <div class="row">
4     <div class="col">
5       <div>
6         
7       </div>
8     </div>
9     <div class="col">
10      <div>
11        
12      </div>
13    </div>
14    <div class="col">
15      <div>
16        
17      </div>
18    </div>
19  </div>
20
21  <div class="text-folding">
22    <h1>
23      I- <b>FOLD</b> <b>AMERIC
24    </h1>
25    <h2>INTELLIGENT <b>HP</b> FOLDAMER DESIGN</h2>
26  </div>
27  <!--
28  <h2 class="title_page_compos mb-4">
29    {{page_title}}
30  </h2>
31  -->
32
33  <div class="container">
34    <div class="row col-sm-12 p-0 justify-content-center">
35      <div>
36        <correlated-networks></correlated-networks>
37      </div>
38
39      <div>
40        <button #openButtonSelectCorrelated type="button" hidden="false" class="btn btn-lg btn-info btn-folding text-uppercase font-weight-bold col-sm" data-toggle="modal"
41          <i class="far fa-object-ungroup"></i>
42          DISPLAY THE TYPES OF CORRELATED SPACES
43        </button>
44      </div>
45    </div>
46  </div>
47
48 </div>
49
```

Figura 89: Fragmento del archivo folding.component.html.

En la **Figura 90** se observa la validación de los datos introducidos por el usuario, para posteriormente hacer uso del servicio para comunicar el frontend con el backend y que la lógica de negocio realice todo el procesamiento necesario.

```
frontend > src > app > components > folding > folding.component.ts > ...
132 onSubait(form) {
133   this.loading=true;
134   $('#errorMessage').hide();
135
136   // Verificamos que el aminoácido sea válido
137   if(this.evolutionaryAlgorithm.validateAminoacid()) {
138     // Si es válido mandamos la petición
139     // Expandimos la cadena de aminoácidos
140     this.evolutionaryAlgorithm.expandAminoacid();
141
142     if(this.evolutionaryAlgorithm.space_type == "correlated") {
143       // Setear los puntos de la red correlacionada
144       this.evolutionaryAlgorithm.setPointsCorrelatedNetwork2D(this.resultsData.upperLeftPoint, this.resultsData.upperRightPoint, this.resultsData.lowerLeftPoint, this.resultsData.lowerRightPoint);
145       this.resultsData.fileNameCorrelatedNetwork = this.evolutionaryAlgorithm.fileNameCorrelatedNetwork;
146       // Parametro que nos dice cuantas veces se repite la red
147       if(this.evolutionaryAlgorithm.dimension_type == "3D_cubic"){
148         this.evolutionaryAlgorithm.calculateLengthMatrix3D();
149       }
150     }
151
152     this.resultsData.evolutionaryAlgorithm = this.evolutionaryAlgorithm;
153     this.evolutionaryAlgorithmService.execute(this.evolutionaryAlgorithm).subscribe(
154       response => {
155         //console.log(response);
156         if(response.status == "success") {
157           this.status = response.status;
158
159           // Persistir los resultados devueltos por el API
160           this.resultsData.resultsExperiments = response.experiments;
161           this.resultsData.dimensionType = response.dimension_type;
162           this.resultsData.spaceType = response.space_type;
163
164           // Persistir los parametros
165           localStorage.setItem('params', JSON.stringify(this.evolutionaryAlgorithm));
166
167           // Redireccion al componente para visualizar los resultados del EA.
168           this._router.navigate(['results']);
169         } else {
170           this.status = 'error';
171           this.loading=false;
172           $('#errorMessage').show();
173         }
174       },
175       error => {
176         // this.status = 'error';
177         this.loading=false;
178         this.status = error.statusText;
179         $('#errorMessage').show();
180       }
181     );
182   }
183 }
```

Figura 90: Fragmento del archivo folding.component.ts.

En la **Figura 91**, se presenta la funcionalidad de la comunicación entre el backend y el frontend, donde se preparan los datos que serán enviados y puedan ser interpretados de forma correcta por la lógica de negocio.

```
frontend > src > app > services > evolutionaryAlgorithm.service.ts > ...
1 import { Injectable } from '@angular/core';
2 import { HttpClient, HttpHeaders } from '@angular/common/http';
3 import { Observable } from 'rxjs';
4 import { global } from './global';
5
6 @Injectable()
7 export class EvolutionaryAlgorithmService {
8   public url: string;
9   public response;
10  public status;
11
12  constructor(
13    public _http: HttpClient
14  ) {
15    this.url = global.url;
16  }
17
18  test() {
19    return 'Hola mundo desde el servicio del AE !!!';
20  }
21
22  execute(EA): Observable<any> {
23    let json = JSON.stringify(EA);
24    let params = 'json='+json;
25
26    // Console Log para ver el json de la petición
27    // console.log(params);
28
29    let headers = new HttpHeaders().set('Content-Type', 'application/x-www-form-urlencoded');
30    return this._http.post(this.url+'EA/execute', params, {headers: headers});
31  }
32
33  injectNewMaterial(EA): Observable<any> {
34    let json = JSON.stringify(EA);
35    let params = 'json='+json;
36
37    // Console Log para ver el json de la petición
38    // console.log(params);
39
40    let headers = new HttpHeaders().set('Content-Type', 'application/x-www-form-urlencoded');
41    return this._http.post(this.url+'EA/injectNewMaterial', params, {headers: headers});
42  }
43 }
```

Figura 91: Fragmento del archivo evolutionaryAlgorithm.service.ts.

En la **Figura 92** podemos ver el controlador encargado de recibir los datos y delegar al modelo toda la ejecución del algoritmo evolutivo. Es el mismo controlador el encargado de recibir una respuesta del modelo, y de esta forma prepararla para regresarla al frontend y pueda ser interpretada de forma correcta.

```
backend > app > Http > Controllers > EvolutionaryAlgorithmController.php
4
5 use Illuminate\Http\Request;
6 use App\Helpers\Helpers;
7
8 use App\Models\EvolutionaryAlgorithm\EvolutionaryAlgorithmManager;
9
10 // para la inyección
11 use App\Models\EvolutionaryAlgorithm\Point;
12 use App\Models\EvolutionaryAlgorithm\Generation;
13 use App\Models\EvolutionaryAlgorithm\Conformation;
14 use App\Models\EvolutionaryAlgorithm\EvolutionaryAlgorithmTypes\Simple;
15
16
17 class EvolutionaryAlgorithmController extends Controller
18 {
19
20     public function execute(Request $request) {
21
22         $json = $request->input('json', null);
23         $params_array = json_decode($json, true);
24
25         $evolutionaryAlgorithmManager = new EvolutionaryAlgorithmManager($params_array);
26         $data = $evolutionaryAlgorithmManager->executeEvolutionaryAlgorithm();
27
28         return response()->json($data, $data['code']);
29     }
30
31     public function injectNewMaterial(Request $request) {
32
33         $json = $request->input('json', null);
34         $params_array = json_decode($json, true);
35
36         $evolutionaryAlgorithmManager = new EvolutionaryAlgorithmManager($params_array);
37         $data = $evolutionaryAlgorithmManager->executeInjectNewMaterial();
38
39         return response()->json($data, $data['code']);
40     }
41
42 }
43
44 }
45
```

Figura 92: Fragmento del archivo EvolutionaryAlgorithmController.php.

En la **Figura 93**, podemos observar el inicio de la funcionalidad del algoritmo evolutivo, donde se realizará su ejecución de acuerdo a los parámetros dados por el usuario y luego de la ejecución se devuelve la respuesta al controlador.

```

backen > app > Models > EvolutionaryAlgorithm > EvolutionaryAlgorithmManager.php
9  use App\Models\EvolutionaryAlgorithm\EvolutionaryAlgorithmTypes\Simple;
10
11 class EvolutionaryAlgorithmManager extends Model
12 {
13     use HasFactory;
14     private $params_array;
15
16     public function __construct($params_array) {
17         $this->params_array = $params_array;
18     }
19
20     function __destruct() {
21         $this->params_array = null;
22     }
23
24     public function executeEvolutionaryAlgorithm() {
25
26         $data = array(
27             'code' => 404,
28             'status' => 'error',
29             'message' => "Data dosen't sending"
30         );
31
32         if(!empty($this->params_array)) {
33
34             // Se precogen los datos
35             $optimizationAlgorithm = $this->params_array['optimization_algorithm'];
36             $sequence = $this->params_array['aminoacid'];
37             $spaceType = $this->params_array['space_type'];
38             $dimensionType = $this->params_array['dimension_type'];
39             $fileNameCorrelatedNetwork = $this->params_array['fileNameCorrelatedNetwork'];
40             if($spaceType == "correlated") {
41                 $pointUpperLeft = new Point($this->params_array['upperLeftPoint'][1] , $this->params_array['upperLeftPoint'][0], null, null, null);
42                 $pointUpperRight = new Point($this->params_array['upperRightPoint'][1] , $this->params_array['upperRightPoint'][0], null, null, null);
43                 $pointLowerLeft = new Point($this->params_array['lowerLeftPoint'][1] , $this->params_array['lowerLeftPoint'][0], null, null, null);
44                 $pointLowerRight = new Point($this->params_array['lowerRightPoint'][1] , $this->params_array['lowerRightPoint'][0], null, null, null);
45                 $pointsCorrelatedNetworkSelected = array($pointUpperLeft, $pointUpperRight, $pointLowerLeft, $pointLowerRight);
46             } else {
47                 $pointsCorrelatedNetworkSelected = null;
48             }
49             $selectionOperator = $this->params_array['selection_op'];
50             $percentOfTournament = $this->params_array['percent_tournament'];
51             $percentOfPopPercent = $this->params_array['percent_best'];
52             $crossoverType = $this->params_array['crossover_op'];
53             $crossoverProbability = $this->params_array['crossover_probability'];
54             $mutationType = $this->params_array['mutation_op'];
55             $mutationProbability = $this->params_array['mutation_probability'];
56             $isKnowBestFitness = $this->params_array['i_know_fitness'];
57         }
58     }
59 }

```

Figura 93: Fragmento del archivo EvolutionaryAlgorithmManager.php.

Una vez que él el folding.component.ts (**Figura 90**), recibe una respuesta por parte del servicio, se verifica el resultado de la ejecución. Con la verificación se obtiene la decisión de sí, se redirige al componente results.component o se muestra una alerta de error.

Cuando la respuesta por parte de la lógica de negocio es positiva, se redirige al results.component, mostrando el código HTML (**Figura 94**) en el que se puede interactuar con los resultados del algoritmo evolutivo.

```

67 </div>
68 </div>
69 </div>
70 <hr/>
71
72 <h4 class="h4-titles mb-3">Analysis and improvement of results!!</h4>
73
74 <!-- BOTONES DROPDOWN -->
75 <div class="col-md-12 ml-0 pl-0">
76   <div class="container">
77     <div class="row justify-content-center">
78       <!-- DROPDOWN PARA GRAFICAR LAS FUNCIONES DE APTITUD -->
79       <div class="btn-group dropup">
80         <button type="button" class="btn alert-primary border border-primary btn-result dropdown-toggle text-uppercase font-weight-bold mb-2 col-sm mt-1" data-toggle=
81           <i class="fas fa-chart-area"></i>
82           Graphs of the folding behavior
83         </button>
84         <div class="dropdown-menu">
85           <!-- BOTON PARA GRAFICAR LA SUMA TOTAL DE FITNESS PARA CADA GENERACION -->
86           <button class="dropdown-item" data-toggle="modal" data-target="#SumFitnessModal">
87             <i class="fas fa-chart-bar"></i>
88             Sum of each generation.
89           </button>
90
91           <!-- BOTON PARA GRAFICAR LOS FITNESS DE UNA GENERACION -->
92           <button class="dropdown-item" data-toggle="modal" data-target="#ParticularGenerationModal">
93             <i class="fas fa-chart-bar"></i>
94             Particular generation.
95           </button>
96
97           <!-- BOTON PARA GRAFICAR LOS FITNESS DE UNA GENERACION -->
98           <button class="dropdown-item" data-toggle="modal" data-target="#AverageRadiusAndDiameter">
99             <i class="fas fa-chart-bar"></i>
100             Average Radius of Gyration <br/> Average Maximum Diameter.
101           </button>
102         </div>
103       </div>
104     </div>
105   </div>
106 </div>
107 </div>
108
109 <!-- FIN BOTONES DROPDOWN -->
110
111 <div class="container mt-4">
112   <div class="row justify-content-center">
113     <div class="col-sm-4 mr-5">
114       <div class="modal fade bd-example-modal-xl id="SumFitnessModal" tabindex="-1" role="dialog" aria-labelledby="SumFitnessModalLabel" aria-hidden="true">
115         <div class="modal-dialog modal-xl" role="document">

```

Figura 94: Fragmento del archivo results.component.html.

Mientras que para la funcionalidad de la visualización de los resultados dependiendo de la dimensión deseada, el archivo results.component.ts (**Figura 95**) utiliza los servicios necesarios, dependiendo de si se quieren visualizar los resultados en 2D, 3D, 2D correlacional o 3D correlacional.

```

152
153 onSubmit(async () => {
154   let generation = this.experiments[this.folding.experiment][this.folding.generation];
155   this.conformationClone = generation[0][this.folding.conformation];
156
157   if(this.spaceType == 'correlated'){
158
159     if(this.counter_NET == 0){
160
161       this.resultsData.fileNameCorrelatedNetwork = this.resultsData.fileNameCorrelatedNetwork.replace('1','');
162       this.counter_NET++;
163       this.getFile();
164
165     }else{
166       if(this.dimensionType == '3D_cubic' ){
167
168         this.engServ3D_Correlated.ngOnDestroy();
169         this.gui.hide();
170         this.gui = new GUI({ autoPlace: false });
171         this.engServ3D_Correlated.createScene(this.gui,this.rendererCanvas,this.conformationClone ,this.arrayCubes);
172         this.engServ3D_Correlated.animate();
173       }else{
174
175         // 2D correlacional OK
176         this.engServ2D_Correlated.ngOnDestroy();
177         this.engServ2D_Correlated.createScene(this.rendererCanvas,this.conformationClone ,this.arrayCubes);
178         this.engServ2D_Correlated.animate();
179       }
180
181     }
182
183   }
184
185   } else if(this.dimensionType == '3D_cubic' ){
186     this.engServ3D.createScene(this.rendererCanvas, this.conformationClone);
187     this.engServ3D.animate();
188   } else {
189     this.engServ.createScene(this.rendererCanvas, this.conformationClone);
190     this.engServ.animate();
191   }
192 }
193
194
195
196 }
197
198
199 private getFile(){
200   var linea=Array();

```

Figura 95: Fragmento del archivo results.component.ts.

4. Fase de transición

4.1. Modelo de pruebas

Finalmente, una prueba de la ejecución de la simulación del plegamiento de una proteína, su interacción con los resultados y sus gráficas, por lo que se pueden considerar los siguientes detalles:

- Se realiza una prueba en 2D cuadrado homogéneo
- Se realiza una prueba en 2D cuadrado correlacional
- Se realiza una prueba en 2D triangular homogéneo
- Se realiza una prueba en 3D cúbico
- Se realiza una prueba en 3D cúbico correlacional
- Se realizan pruebas de interacción en cada uno de los espacios probados
- Se realizan pruebas de la funcionalidad de exportar el plegamiento en cada uno de los espacios probados
- Se realizan pruebas de las gráficas de acuerdo al fitness en cada una de las pruebas

A continuación se presentan las capturas de pantalla de cada una de las pruebas enlistadas en el párrafo anterior.

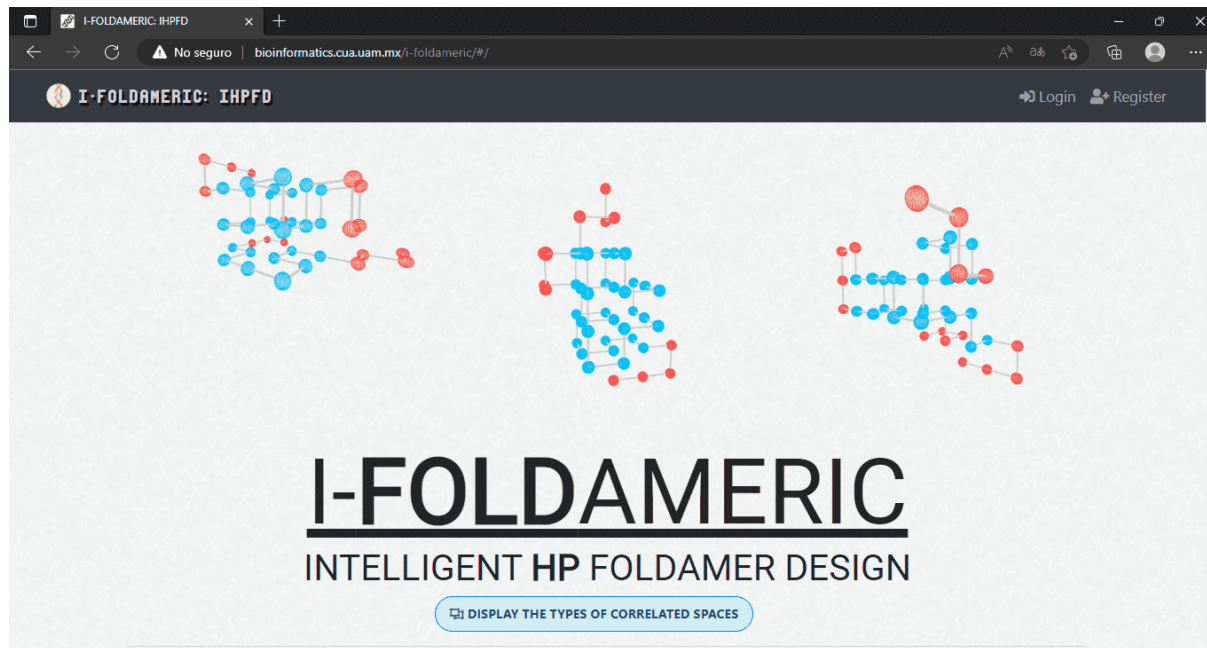


Figura 96: Página principal del sistema I-Foldameric.

Figura 97: Formulario con la secuencia P11H16P20 en 2D cuadrado homogéneo.

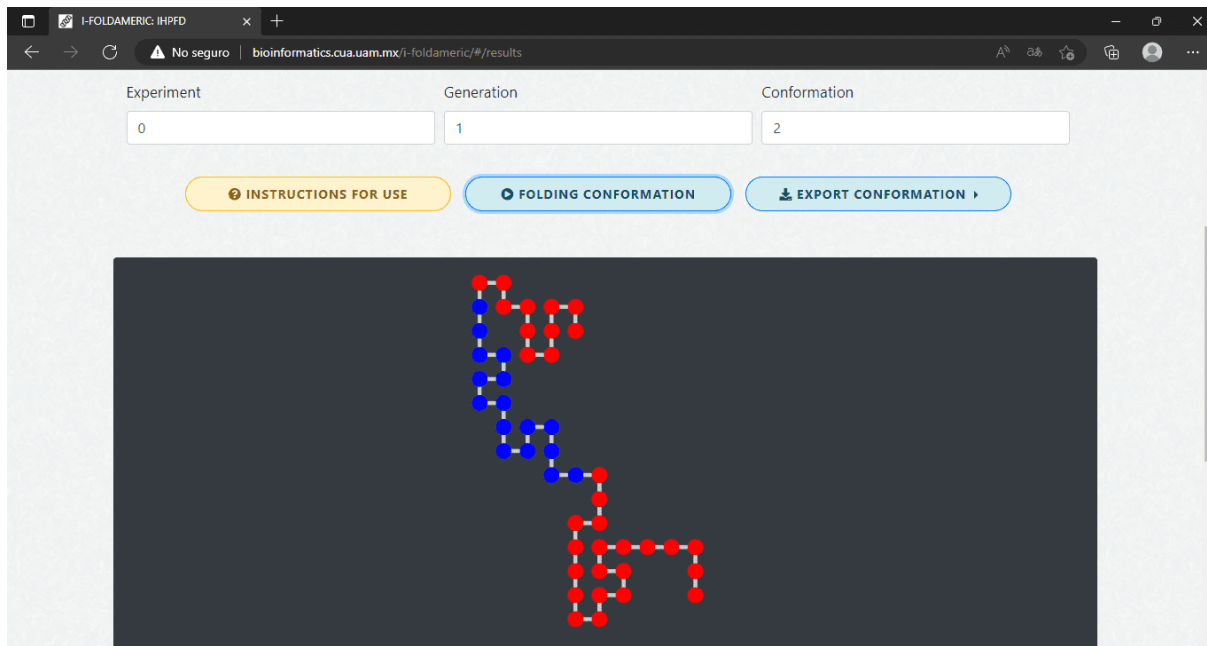


Figura 98: Resultado del plegamiento de la secuencia P11H16P20 en 2D cuadrado homogéneo.

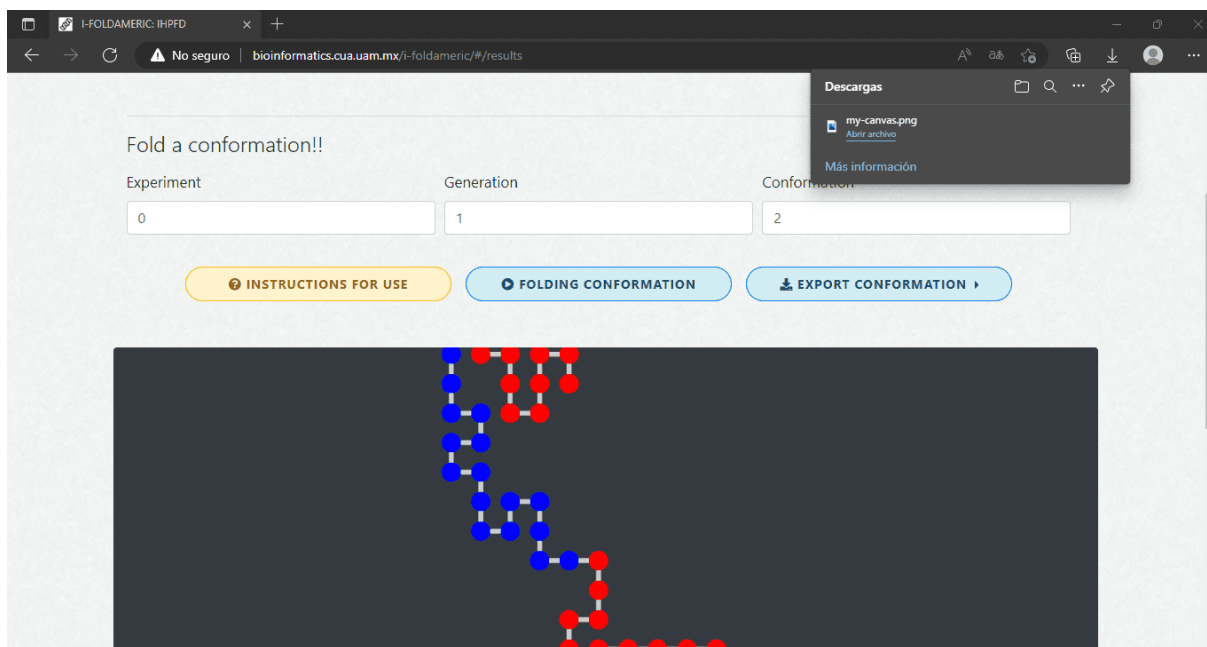


Figura 99: Exportando el plegamiento 2D cuadrado homogéneo en formato .png.

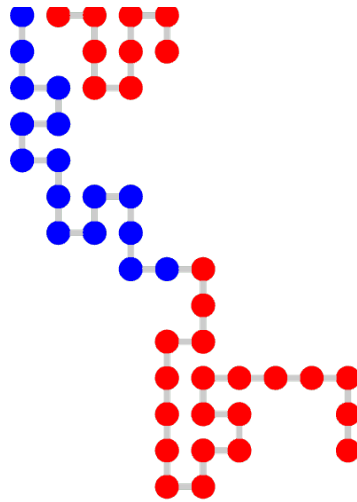


Figura 100: Plegamiento exportado en formato .png en la **Figura 99**.

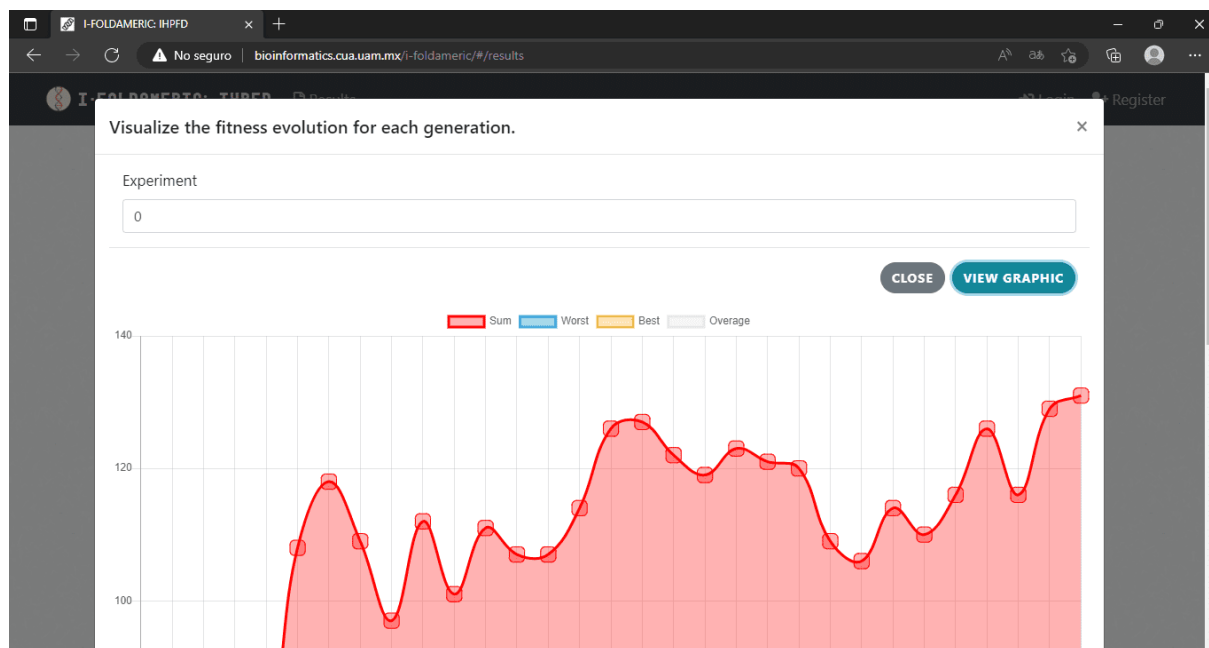


Figura 101: Visualizando la gráfica de la secuencia de aminoácidos en 2D cuadrado homogéneo.

DISPLAY THE TYPES OF CORRELATED SPACES

Aminoacid or HP Sequence
P11H16P20

Space Types: Correlated Lattice
Dimensions Types: 2D Square Correlated
Optimization Algorithms: Simple

Selection Operator: Roulette
CrossOver Operator: One-point
Mutation Operator: Predefined

Fitness Function: Dill's model
Other genetic techniques: ☐ Elitism ☐ Clamp mutation ☐ Caterpillar mutation

Parameters

Chose the number of conformations to generate: 20
How many times do you want to run the evolutionary algorithm?: 30

How many experiments do you want to do?: 1
Sampling value: 1
Percent of individuals in the tournament: 20

Figura 102: Formulario con la secuencia P11H16P20 en 2D cuadrado correlacional.

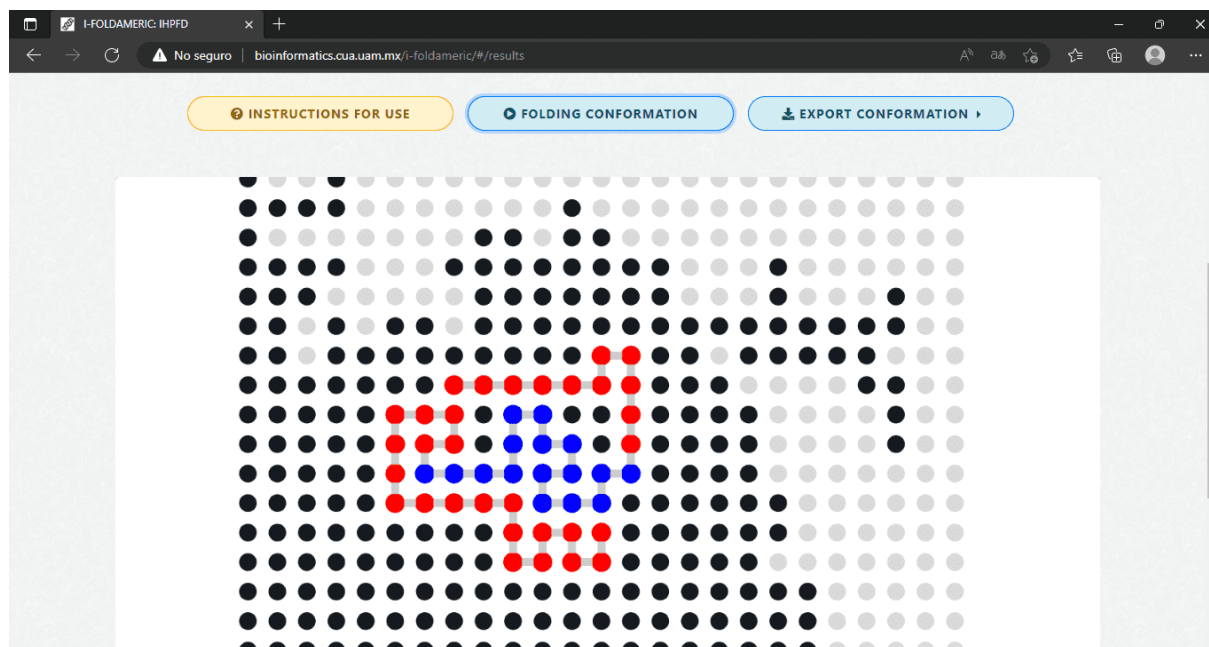


Figura 103: Resultado del plegamiento de la secuencia P11H16P20 en 2D cuadrado correlacional.

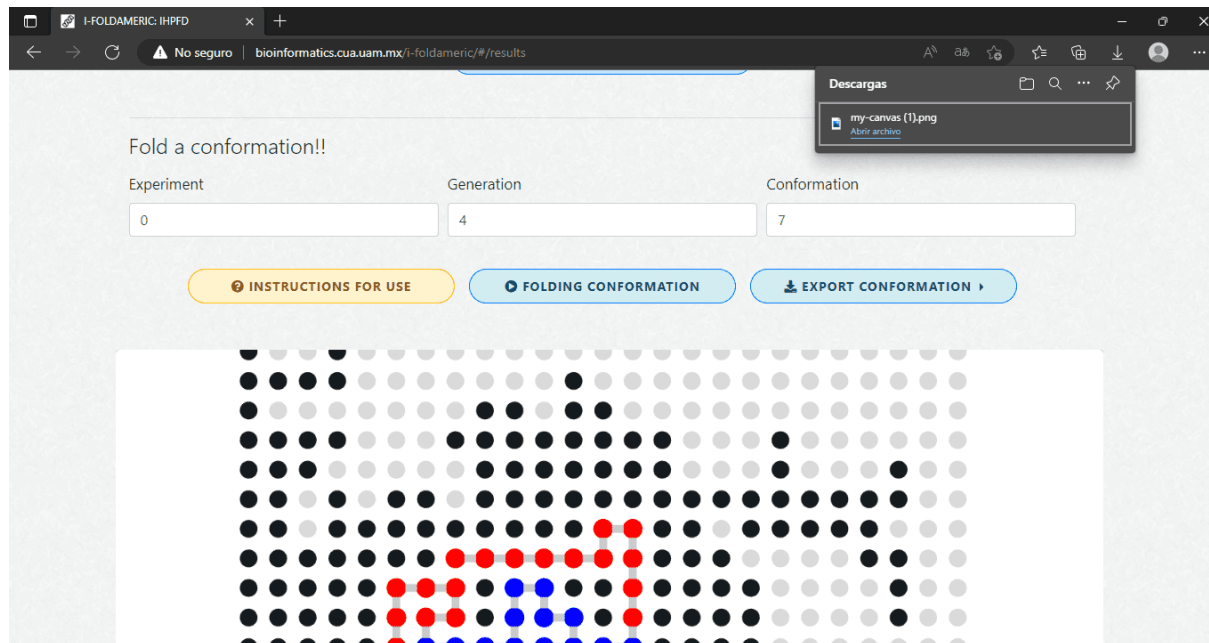


Figura 104: Exportando el plegamiento 2D cuadrado correlacional en formato .png.

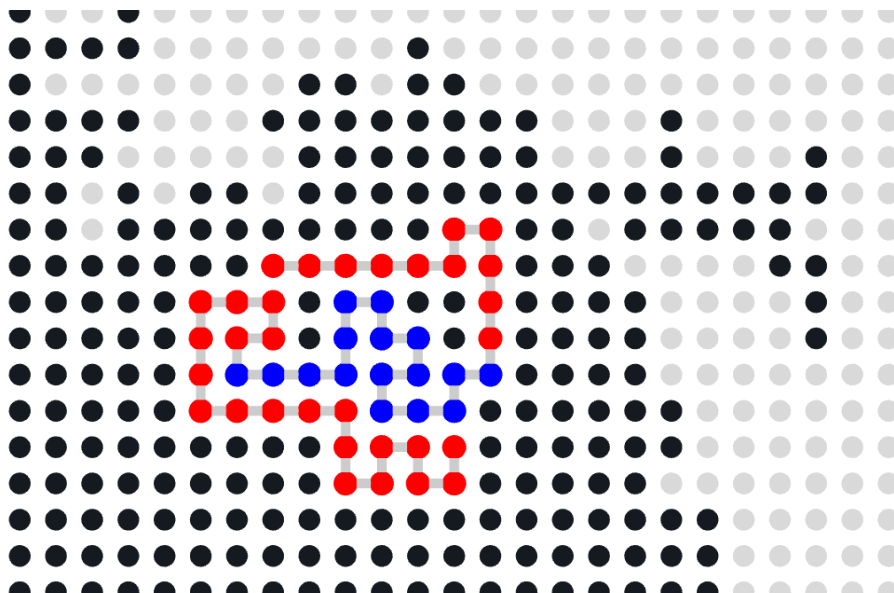


Figura 105: Plegamiento exportado en formato .png en la **Figura 104**.

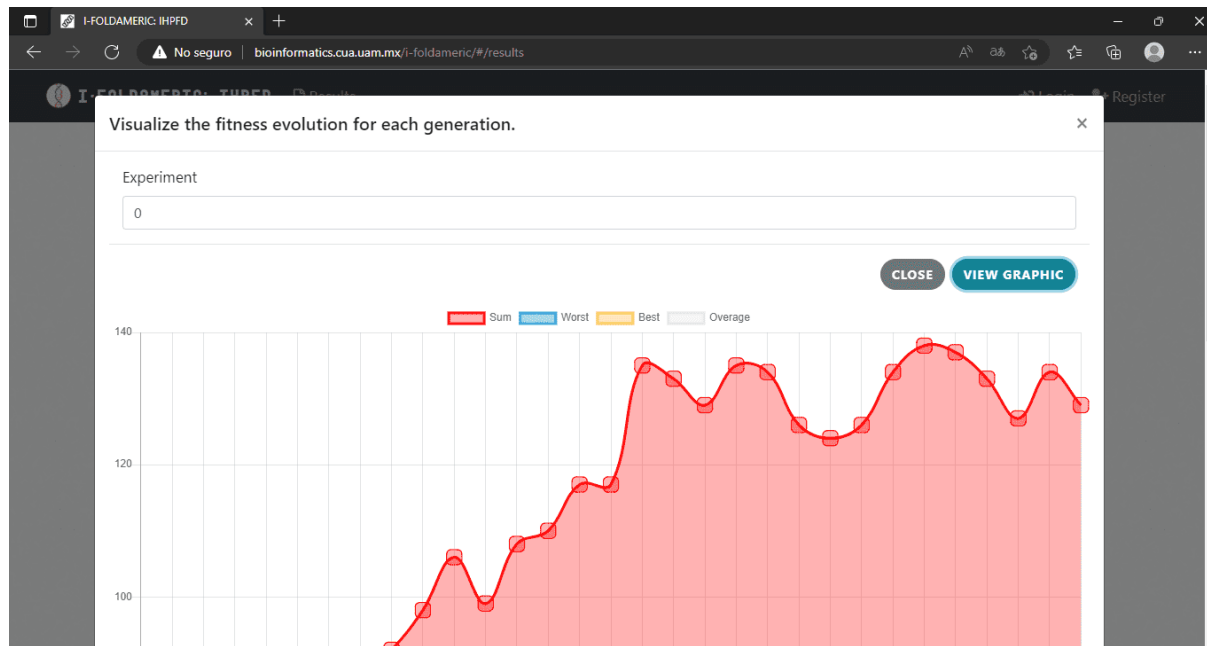


Figura 106: Visualizando la gráfica de la secuencia de aminoácidos en 2D cuadrado correlacional.

DISPLAY THE TYPES OF CORRELATED SPACES

Aminoacid or HP Sequence: P11H16P20

Space Types: Homogeneous Lattice

Space Dimensions: 2D Triangle

Optimization Algorithms: Simple

Selection Operator: Roulette

CrossOver Operator: One-point

Mutation Operator: Predefined

Fitness Function: Dill's model

Other genetic techniques: ☐ Elitism ☐ Clamp mutation ☐ Caterpillar mutation

Parameters:

Chose the number of conformations to generate: 20

How many times do you want to run the evolutionary algorithm?: 30

How many experiments do you want to do?: 1

Sampling value: 1

Percent of individuals in the tournament: 20

Figura 107: Formulario con la secuencia P11H16P20 en 2D triangular homogéneo.

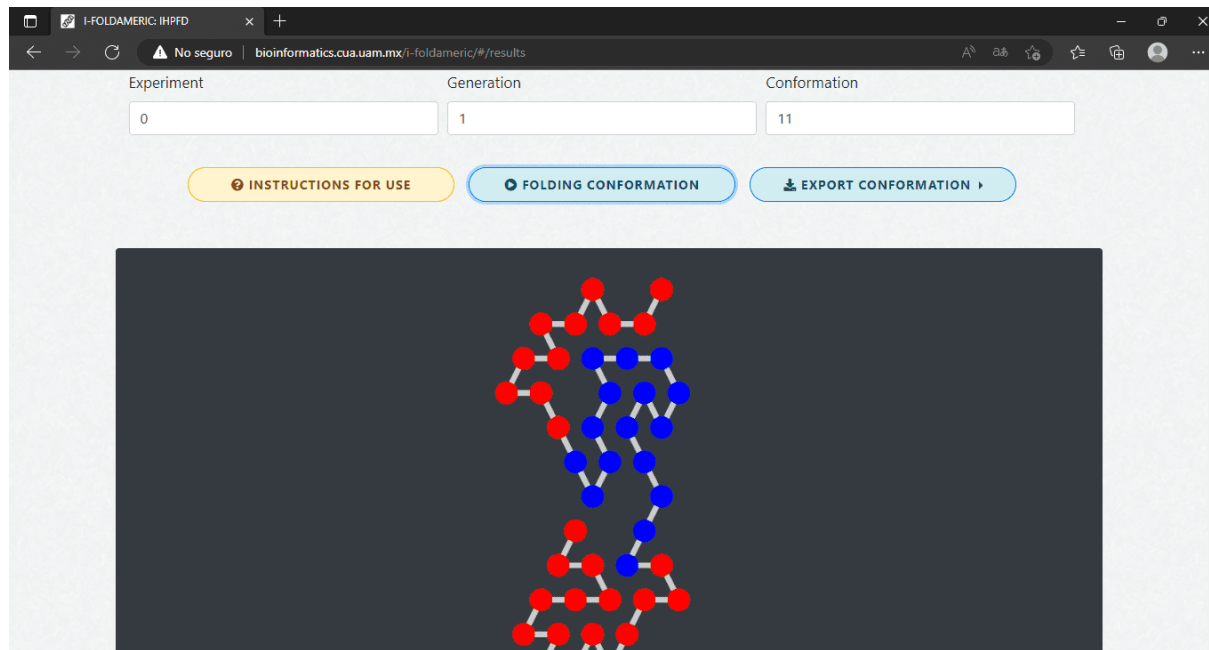


Figura 108: Resultado del plegamiento de la secuencia P11H16P20 en 2D triangular homogéneo.

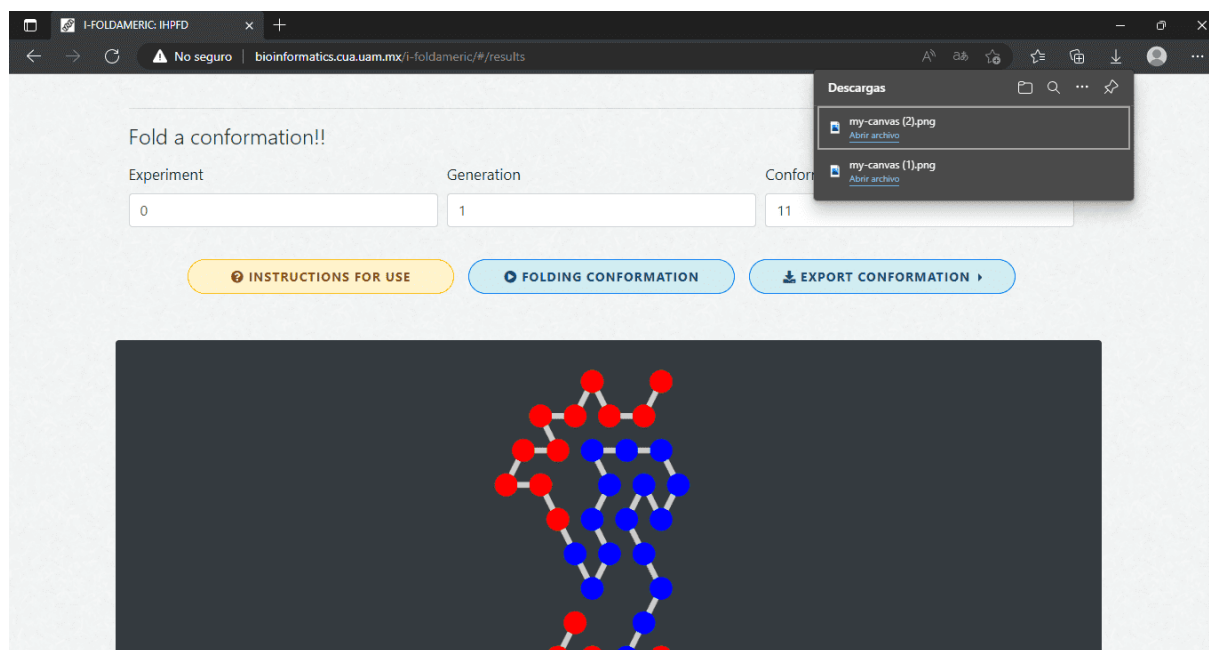


Figura 109: Exportando el plegamiento 2D triangular homogéneo en formato .png.

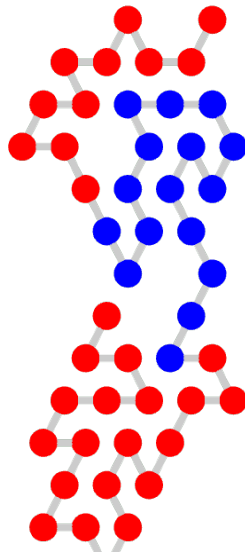


Figura 110: Plegamiento exportado en formato .png en la **Figura 109**.

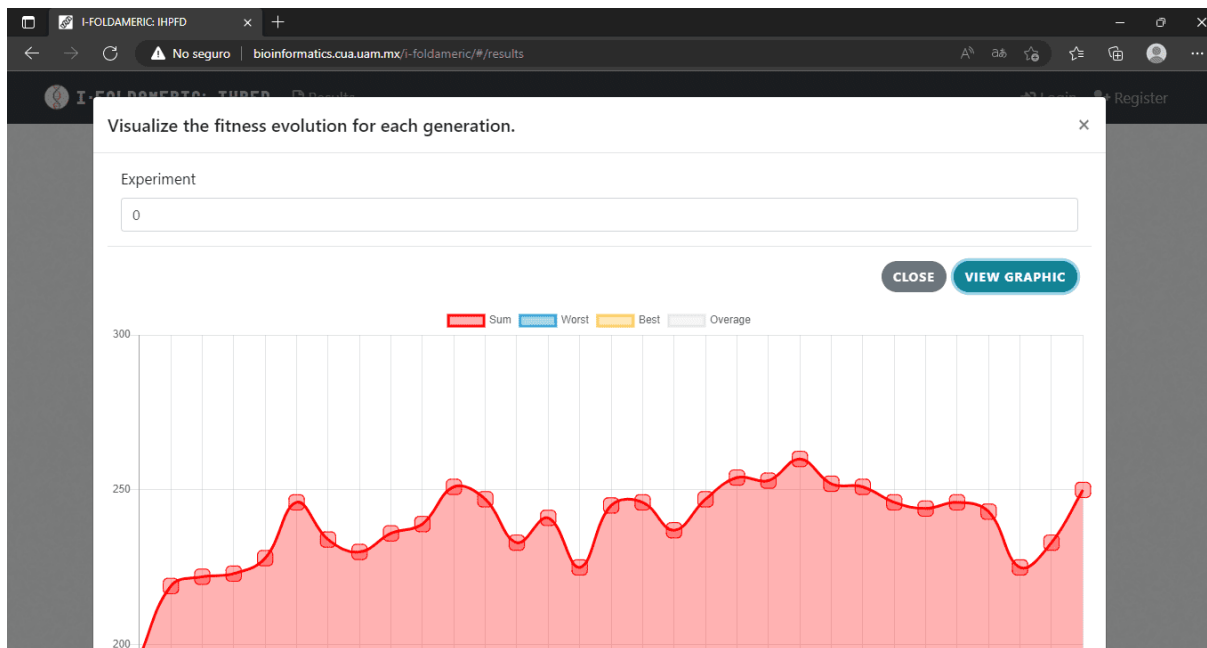


Figura 111: Visualizando la gráfica de la secuencia de aminoácidos en 2D triangular homogéneo.

The screenshot shows the I-FOLDAMERIC: IHPFD web interface. At the top, there is a button labeled "DISPLAY THE TYPES OF CORRELATED SPACES". Below this, the "Aminoacid or HP Sequence" is entered as "P11H16P20". The "Space Types" dropdown is set to "Homogeneous Lattice", "Space Dimensions" to "3D Cubic", and "Optimization Algorithms" to "Simple". The "Selection Operator" is "Roulette", "CrossOver Operator" is "One-point", and "Mutation Operator" is "Predefined". Under "Fitness Function", "Dill's model" is selected. Under "Other genetic techniques", there are checkboxes for "Elitism", "Clamp mutation", and "Caterpillar mutation", all of which are currently unchecked. In the "Parameters" section, "Chose the number of conformations to generate" is set to 20, "How many times do you want to run the evolutionary algorithm?" is set to 30, "How many experiments do you want to do?" is set to 1, "Sampling value" is set to 1, and "Percent of individuals in the tournament" is set to 20.

Figura 112: Formulario con la secuencia P11H16P20 en 3D cúbico.

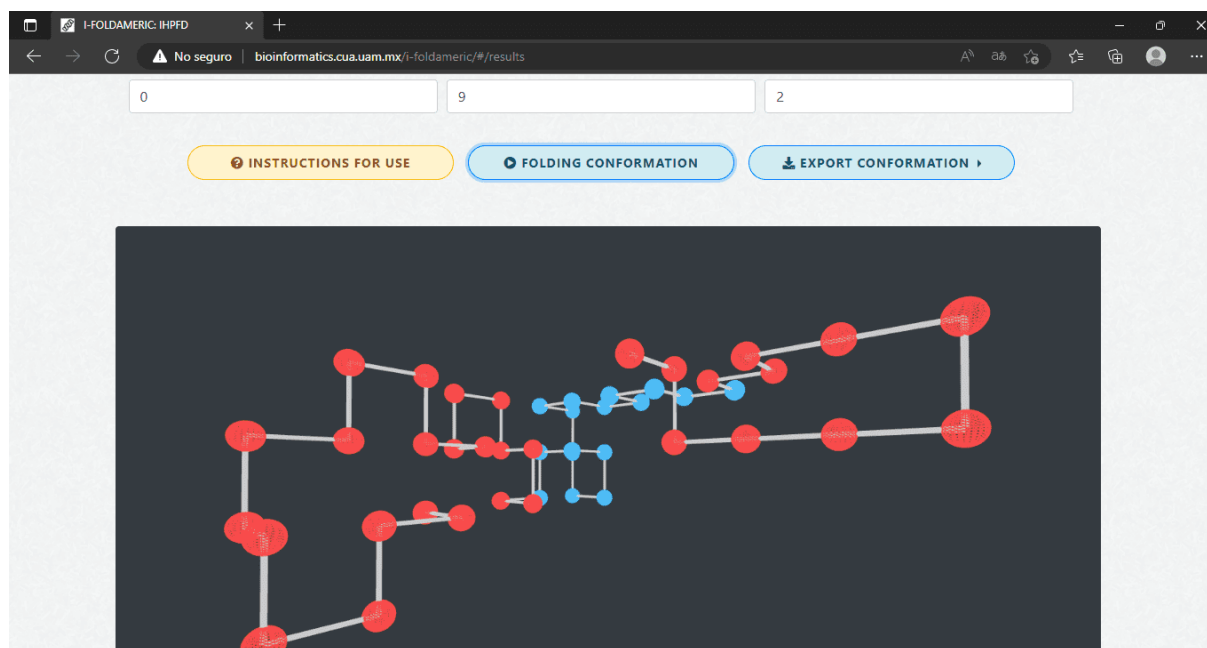


Figura 113: Resultado del plegamiento de la secuencia P11H16P20 en 3D cúbico.

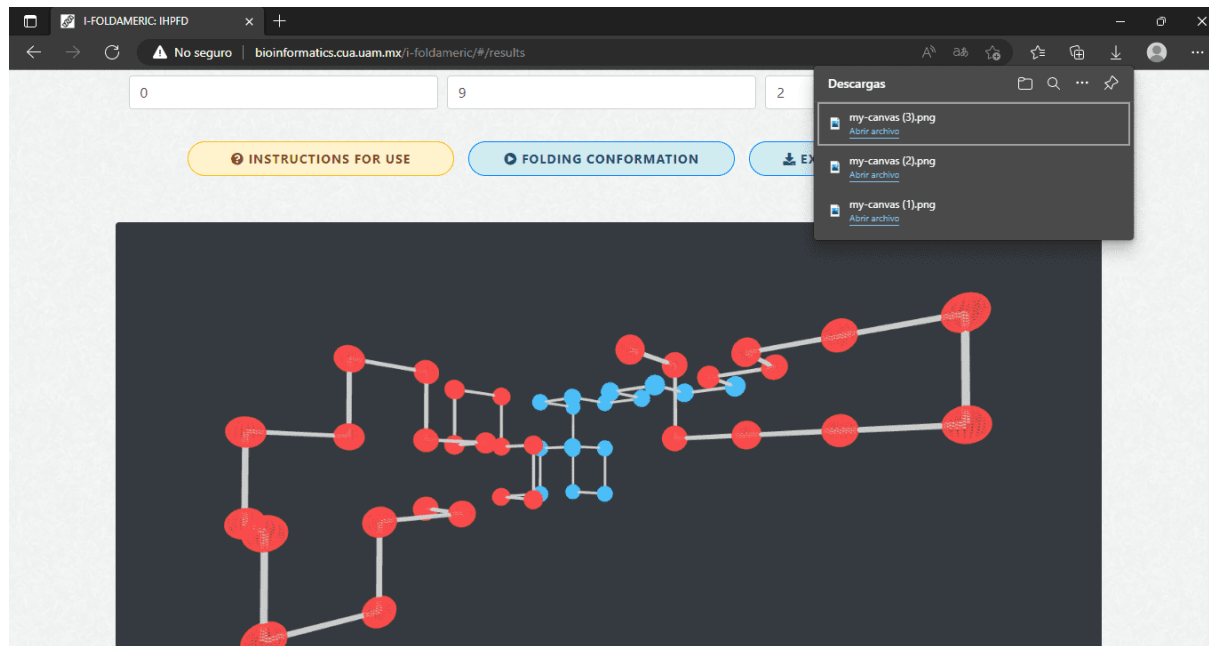


Figura 114: Exportando el plegamiento 3D cúbico en formato .png.

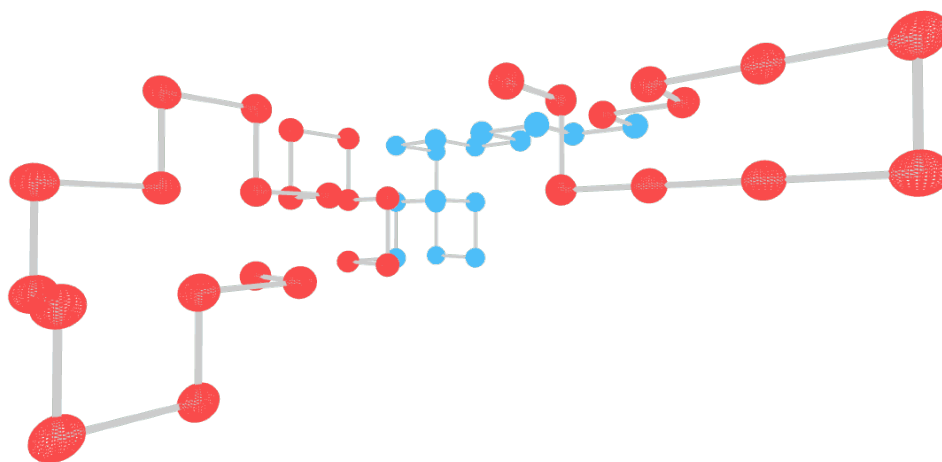


Figura 115: Plegamiento exportado en formato .png en la **Figura 114**.

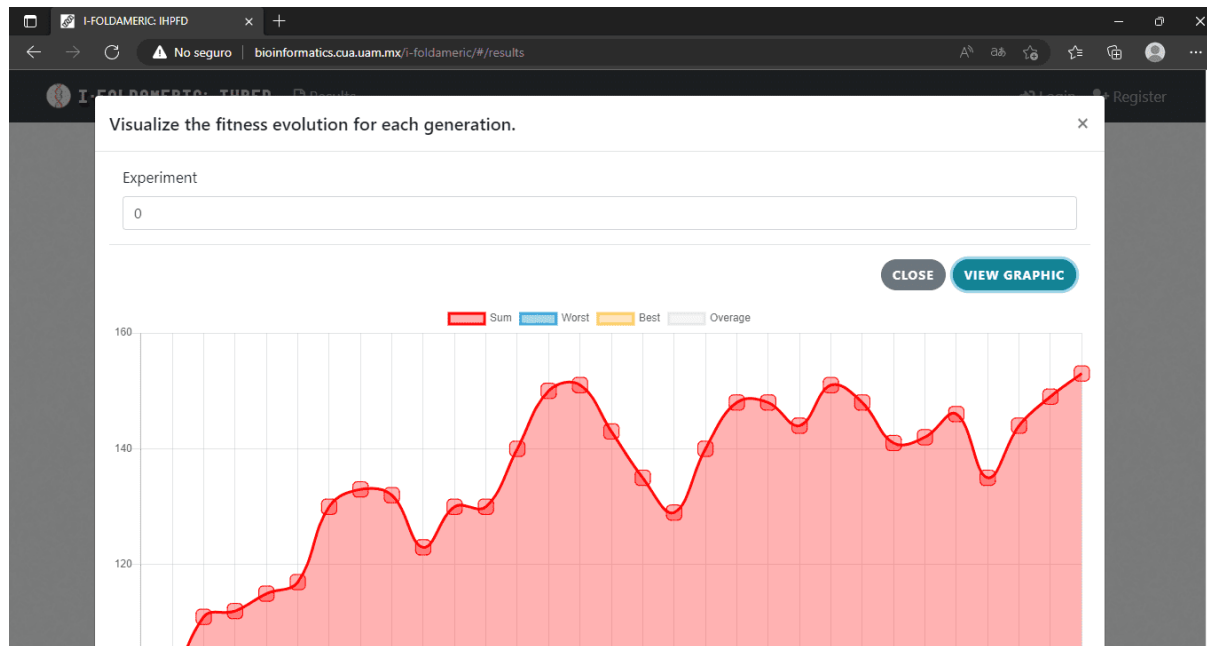


Figura 116: Visualizando la gráfica de la secuencia de aminoácidos en 3D cúbico.

DISPLAY THE TYPES OF CORRELATED SPACES

Aminoacid or HP Sequence: P11H16P20

Space Types: Correlated Lattice

Dimensions Types: 3D Cubic Correlated

Optimization Algorithms: Simple

Selection Operator: Roulette

CrossOver Operator: One-point

Mutation Operator: Predefined

Fitness Function: Dill's model

Other genetic techniques: ☐ Elitism ☐ Clamp mutation ☐ Caterpillar mutation

Parameters:

Chose the number of conformations to generate: 20

How many times do you want to run the evolutionary algorithm?: 30

How many experiments do you want to do?: 1

Sampling value: 1

Percent of individuals in the tournament: 20

Figura 117: Formulario con la secuencia P11H16P20 en 3D cúbico correlacional.

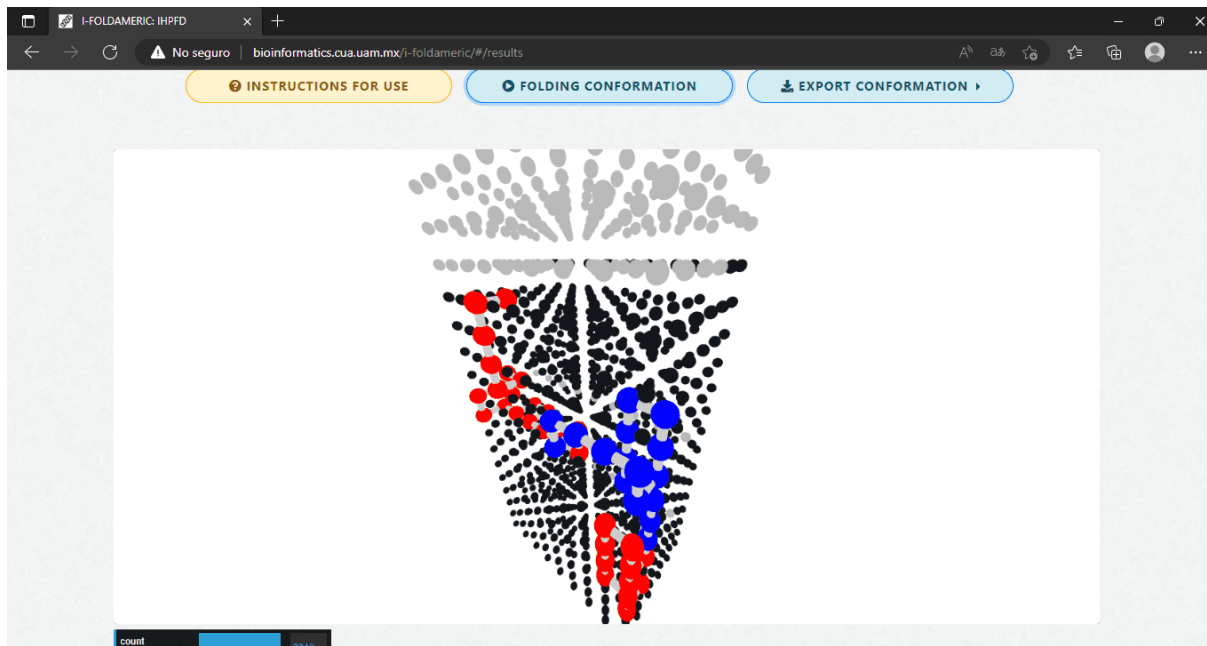


Figura 118: Resultado del plegamiento de la secuencia P11H16P20 en 3D cúbico correlacional.

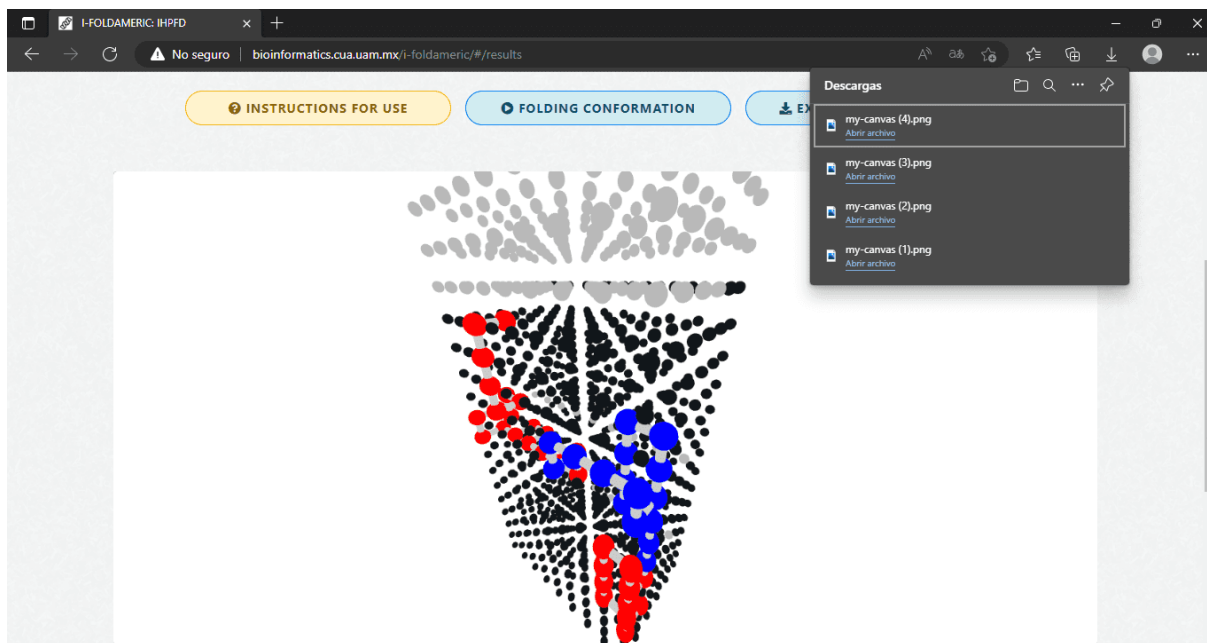


Figura 119: Exportando el plegamiento 3D cúbico correlacional en formato .png.



Figura 120: Plegamiento exportado en formato .png en la **Figura 119**.

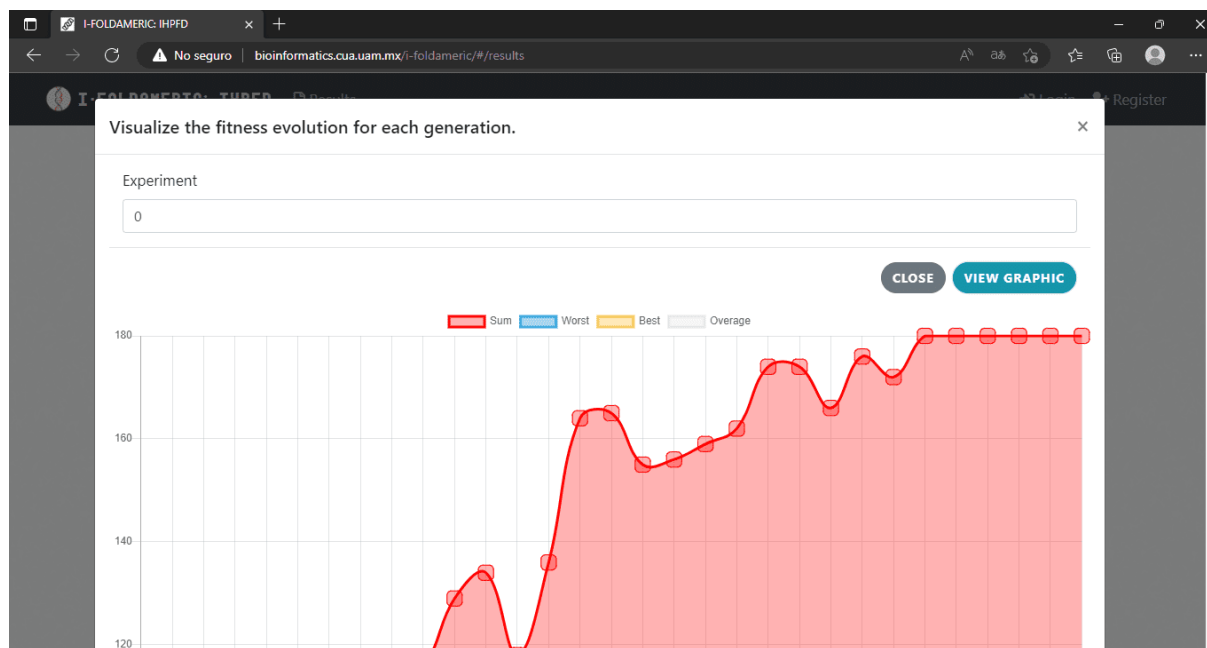


Figura 121: Visualizando la gráfica de la secuencia de aminoácidos en 3D cúbico correlacional.