UNIVERSIDAD DE TARAPACÁ
DEPARTAMENTO DE INGENIERÍA EN COMPUTACIÓN E INFORMÁTICA
FACULTAD DE INGENIERÍA

Sistema de Enseñanza y Aprendizaje en Programación

Alumno: **Pedro Araya A.**
Profesor: **Diego Aracena P.**
Asignatura: **Proyecto IV**

ARICA, 22 DE DICIEMBRE DE 2021

TABLA DE CONTENIDOS

1. INTRODUCCIÓN

Se sabe que actualmente las empresas buscan el trabajo en equipo y líderes que guíen a la empresa a cumplir sus objetivos. En el desarrollo de software es igual, es necesario en el ámbito laboral saber cooperar con los diferentes miembros de un equipo, saber expresar las ideas y tener una buena productividad. Actualmente en el Departamento de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá, existen múltiples factores que pueden impedir el buen aprendizaje de los alumnos en torno al desarrollo de software.

De aquí nace la iniciativa del proyecto “Sistema de Enseñanza y Aprendizaje en programación” o SAEP-UTA, que consiste en un sistema que va a permitir una mejora, a través de una plataforma web permite el desarrollo temprano de habilidades necesarias en el campo laboral de desarrollo de software a los estudiantes. Una plataforma que permita, aprender a programar, implementar proyectos, trabajar colaborativamente, sin la necesidad de tener que instalar software, como compiladores, editores de texto, extensiones, etc. Además de permitir a los docentes poder gestionar cursos y evaluaciones en esta misma. Y así tener un ambiente de enseñanza y aprendizaje centralizado.

En el presente informe, se detalla todo el proceso para el desarrollo del proyecto SAEP-UTA, realizado en conjunto a los requerimientos del cliente del proyecto, Ricardo Valdivia Pinto, actual jefe de carrera del departamento de Ingeniería Civil en Computación e Informática.

2. ANTECEDENTES DE LA ORGANIZACIÓN

El desarrollo del proyecto será desarrollado para la organización de la Universidad de Tarapacá, en específico, el departamento de Ingeniería Civil en Computación e Informática. El presente punto detalla los aspectos generales del funcionamiento de esta.

2.1. Descripción general

La carrera de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá, procura la formación de profesionales capaces de identificar y responder a las demandas presentes y futuras del contexto social, conjugando sistemáticamente los conocimientos científicos y técnicos propios del campo disciplinar con los requerimientos del mercado.

2.2. Misión

El Departamento de Ingeniería en Computación e Informática (DICI) tiene como misión crear, difundir y hacer uso de las tecnologías de información y comunicación en beneficio de la sociedad y la formación de ingenieros con sólidas habilidades técnicas y sociales. Esta misión involucra un compromiso y una responsabilidad con la región y el país.

2.3. Visión

El DICI tiene como visión ser un referente en el ámbito informático a través de la innovación a nivel universitario en la región, el país y el mundo.

2.4. Objetivos estratégicos

Formar Ingenieros Civiles en Computación e Informática que posean una sólida base científica, tecnológica y personal que les permita concebir, desarrollar, liderar y evaluar sistemas informáticos para mejorar procesos de negocios, todo ello con una actitud de responsabilidad social, respetando el escenario ético y normativo.

2.5. Historia

Desde aquella fecha se hace cargo de la carrera de Ingeniería de Ejecución en Computación e Informática (Decreto Exento Nº 00.37 del 13 de Abril de 1983).

El 15 de marzo de 1982 asumió como Director Interino el Señor Ricardo Durán Arriagada. Las principales preocupaciones del Sr. Durán fueron consolidar el inicio del departamento y gestionar la salida a perfeccionamiento en España de dos académicos.

En 1985 se crea oficialmente la carrera de Ingeniería Civil en Computación e Informática (Decreto Exento Nº 00.2571 del 20 de Agosto de 1985), y se inicia un plan de contratación de profesores jornada completa para estos efectos. Se utilizan las modernas instalaciones, de aquel entonces, Centro de Tecnología de la Información, que contaba con un microcomputador PDP 11/34 Digital y posteriormente con una VAX 11-750 multiusuario y una Estación Ultrix Digital para docencia.

2.6. Productos

La Universidad de Tarapacá entrega como producto al usuario certificados por sus experiencias, capacitaciones, seminarios, academias y al finalizar la carrera, el diploma de titulación de la carrera. Por otra parte, la Universidad ofrece los siguientes servicios a la comunidad estudiantil: Servicio de atención psicológica Clínica odontológica Estudios clínicos Maternidad Asesorías jurídicas.

2.7. Factor Humano

La siguiente tabla muestra el funcionamiento del factor humano dentro de DICI, denotando; cargo, responsable y una descripción de su función dentro de la organización.

Tabla 1. Factor Humano en la Universidad de Tarapacá.

Cargo	Responsable	Función
Director	Raúl Herrera Acuña	Tiene por función organizar la enseñanza y la investigación del departamento, dirigiendo todos los asuntos académicos, administrativos y financieros.
Jefe de carrera	Ricardo Valdivia Pinto	Es el académico responsable, que bajo la autoridad del Decano, administra y gestiona una o más carreras, desde el punto de vista académico
Secretaria	Claudia Carvajal Araya	Aportar con todos los registros físicos y electrónicos de toda la normativa vigente para todo acto administrativo con relación a la docencia y gestión universitaria.
Secretaria	Marianela Mamani Gutierrez	--
Ingeniero de sistemas	Humberto Urrutia Lopez	Encargado de diseñar, desarrollar, aplicar y mantener los sistemas informáticos del departamento.
Ingeniero de software y seguridad	Patricio Collao Caiconte	Encargado de diseñar, implementar y monitorear planes de seguridad y vulnerabilidad de los sistemas de información, entregando soluciones preventivas y correctivas para fortalecer la seguridad de la información y la continuidad operacional del departamento.
Docente de jornada completa	Diego Aracena Pizarro	Planear, organizar y controlar la docencia, investigación y extensión dentro del área de conocimiento. Además, transmitir los conocimientos de acuerdo a los programas académicos establecidos por la Facultad.
Docente de jornada completa	Mauricio Arriagada Benitez	--

Docente de jornada completa	Roberto Espinosa Oliva	--
Docente de jornada completa	Raúl Herrera Acuña	--
Docente de jornada completa	Héctor Ossandón Díaz	--
Docente de jornada completa	Ricardo Valdivia Pinto	--

2.8. Proveedores

La Universidad de Tarapacá al ser una entidad de carácter público debe realizar sus compras por medio de la plataforma "Mercado Público", el cual hace como su principal proveedor de productos para la universidad, otros de sus proveedores esta REUNA.

2.9. Ventas

El Ingeniero Civil en Computación e Informática de la Universidad de Tarapacá es un profesional competente en el desarrollo de productos de software y en incorporar TICs que mejoren los procesos de negocio de las organizaciones. Este profesional está capacitado para identificar y responder a las demandas presentes y futuras de su contexto laboral, además posee una formación ética integral basada en el respeto por los demás, el compromiso y la responsabilidad social.

2.9. Servicios post venta

Una vez concluida la formación del estudiante la universidad ofrece dos tipos de Postgrado: Doctorado y Magíster. El Doctorado será un programa académico de carácter científico que podrá tener dos modalidades: Programa Regular de Doctorado o Programa de Doctorado por Tesis. Y los programas de Magíster podrán ser de tipo Académico o Profesional. El Programa Académico podrá ser de carácter científico o tecnológico, y a su vez podrá tener las siguientes estructuras: Programa Regular de Magíster o Programa de Magíster por Tesis. El Programa de Magíster Profesional será de Especialización o Actualización de conocimientos.

2.10. Infraestructura y tecnología de información

DICI cuenta con un edificio principal que concentra los laboratorios de clases, oficinas de académicos y administración del departamento. Este cuenta con 6 laboratorios orientados para el trabajo en equipo y desarrollo de tecnologías, además cuenta con un auditorio con capacidad para 80 personas. Anexo al departamento, existen dos laboratorios en el edificio Verónica Rey Mas, los cuales cuentan con puestos de trabajo con computadoras All in One.

Por otra parte, en el departamento, existe un laboratorio de servidores, el cual cuenta con 3 puestos de trabajo, servidores Dell, UPS y Switches, conexión fibra óptica, aire acondicionado y punto de acceso inalámbrico.

Además, existe un laboratorio abierto para los alumnos del departamento todo el día, para sus trabajos y requerimientos especiales.

Los alumnos tienen acceso a:

- Computadores de escritorio y notebooks.
- Kit de robótica Lego.
- Kit de Arduino y Raspberry PI con sensores.
- 3 x Impresora 3D.
- Cortador Láser.
- Drones.

2.11. Unidades de producción

DICI se encuentra dentro de la Universidad de Tarapacá, en la casa Central ubicada en la Región de Arica y Parinacota, en la provincia y comuna de Arica, 18 de Septiembre #2222.

3. PANORAMA GENERAL

3.1. Descripción de la problemática

Actualmente en el Departamento de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá, existen múltiples deficiencias entre los alumnos y docentes a la hora de poner en práctica el aprendizaje y enseñanza de la programación, las principales son:

- Accesibilidad, tanto alumnos como profesores requieren de instalar Software en sus equipos personales o prestados por la universidad, tanto compiladores, editores de texto, dependencias requeridas para el uso de cierto lenguaje de programación, etc. Este aspecto toma varios factores, principalmente la falta de prestación hardware, inexperiencia instalando ambientes de programación, tiempo para la preparación del ambiente, factores económicos, factores temporales como problemas con el equipo, etc.

- Vías de comunicación, no existe una vía rápida de comunicación para poder discutir sobre código de programación, tanto entre alumnos en caso de dudas, y entre alumno profesor para realizar consultas. Todo esto se realiza enviando mensajes o correos con el archivo adjunto, o simplemente de forma sincrónica en un entorno específico.

- Entorno de desarrollo, al programar en algunos lenguajes de programación, hay veces que sucede que el código de programación funciona con ciertas dependencias o versiones que no necesariamente las cumplen sus pares o el docente, por lo cual habrá problemas de ejecución recurrentes en ambientes con diferentes parámetros de ejecución. Para un docente, el poder evaluar este tipo de problemas puede ser una gran pérdida de tiempo.

- Trabajo en equipo, considerando que no todos los alumnos y principalmente los de primeros años conocen como utilizar repositorios Git para el versionado de un proyecto, usualmente recurren a que solo algunos integrantes tengan el código principal y si uno de los alumnos del equipo desea arreglar algún error, este tiene que ponerse de acuerdo con el alumno correspondiente. Lo definido anteriormente, definitivamente no permite la programación en equipo, y esto puede causar problemas en las bases del conocimiento, al tener encargados del código del proyecto, los otros colaboradores no pueden participar activamente, siendo posible causarles deficiencias futuras en sus capacidades de programación.

3.2. Propuesta de solución

El proyecto consiste en un sistema distribuido que mediante una plataforma web orientada al aprendizaje de programación básica e intermedia se logre mejorar las deficiencias mencionadas, solución que será implementada dentro del Departamento de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá. Debido a que los usuarios principales son los alumnos y docentes, se explicarán las funcionalidades principales de cada uno.

La idea principal, es que los alumnos tengan un lugar en donde poder aprender a programar o experimentar con algún lenguaje de programación sin la necesidad de tener el hardware o software adecuado (compiladores, editores de texto, IDEs, etc) y en cualquier lugar con conexión a internet. Esto funcionará a través de repositorios personales de los alumnos en donde podrán compartir carpetas o archivos con sus compañeros y poder trabajar programando colaborativamente, además de poder probar el código a través de un compilador embebido en la aplicación. Otra funcionalidad será el poder realizar talleres o evaluaciones que serán definidas por el docente, en donde se trabajará en conjunto en tiempo real.

Los docentes que requieran utilizar esta aplicación web para la realizar la enseñanza a sus alumnos, podrán inscribir a los alumnos a un curso virtual correspondiente a la asignatura dentro de la aplicación. Dentro del curso virtual, se podrán crear recursos de aprendizaje en los cuales los alumnos podrán acceder en cualquier momento, además de poder crear evaluaciones en donde los alumnos puedan participar tanto individualmente como en grupos (definido por el docente), y trabajar con respecto a un enunciado que él defina pertinente. Luego, una vez finalizado el taller, el docente tendrá acceso al desarrollo de los alumnos de la actividad para realizar la respectiva evaluación.

3.2.1. Modelo BPMN de la propuesta

A continuación se muestra un modelo de BPMN de la propuesta de solución.

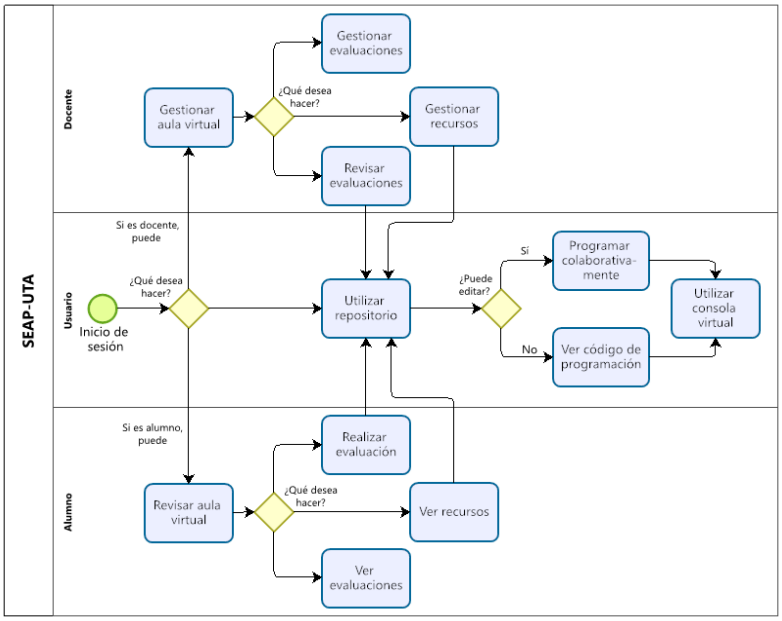

Figura 1. Modelo BPMN de la propuesta de solución.

3.3. Propósito y justificación del proyecto

El proyecto surge de la necesidad de mejorar el estado actual de la enseñanza y aprendizaje de programación y trabajo en equipo de los estudiantes del Departamento de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá, mediante un sistema que contenga las herramientas necesarias para este fin, permitiendo que sea accesible para todos los involucrados. Si consideramos un ambiente de desarrollo de software actual, se requieren de habilidades y hábitos, tales como; buen manejo al explicar código de programación, utilizar lenguaje técnico, realizar una buena documentación, conocimiento amplio en diversos lenguajes de programación y experiencia en el trabajo en equipo.

El propósito principal del proyecto es, hacer más accesible las condiciones de trabajo mencionadas a los estudiantes, de manera que estos tengan un aprendizaje temprano significativo de cómo funciona el desarrollo de software en la actualidad. Además, de que estos puedan desarrollarse como futuros profesionales, en cualquier lugar, sin importar las condiciones, simplemente con un navegador web y acceso a internet.

Otro punto importante, son las evaluaciones, actualmente se realizan de forma independiente por cada alumno que curse la asignatura, cada uno en un entorno de trabajo diferente, por lo cual el docente al finalizar la evaluación, tiene que recolectar y organizar el conjunto de evaluaciones realizadas para luego revisarlas una a una. Es por esto, que el sistema propuesto va a permitir mejorar el tiempo de revisión al ser centralizado, con las herramientas necesarias para llevarlo a cabo.

Como referencia a este proyecto, existen herramientas de trabajo colaborativo, tales como Google Drive con sus herramientas de ofimática o GitHub para alojar proyectos utilizando un sistema de control de versiones.

3.4. Alcance

A continuación se establecen qué puntos serán considerados parte del proyecto y qué no.

- Los lenguajes de programación considerados para la compilación en consola virtual de la plataforma, son; Python, C, C + +, Java y Javascript.

- La compilación online tendrá limitaciones funcionales, en cuanto a librerías utilizables, las entradas posibles, salidas que serán limitadas a resultados de consola y un tiempo límite de espera por una respuesta, además de un límite de uso de memoria RAM y CPU.

- El usuario va a tener la posibilidad de instalar librerías a través de comandos en la consola.

- El diseño de experiencia de usuario (UX) tendrá un nivel inferior al actual en el mercado, además no será orientado a móviles.

3.5. Objetivos del proyecto

3.5.1. Objetivo general

- Desarrollar e implementar el proyecto SEAP-UTA para el Departamento de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá.

3.5.2. Objetivos específicos

- Definir SEAP-UTA mediante técnicas de Ingeniería de Software y modelamiento de sistemas distribuidos.
- Definir y estudiar las herramientas necesarias para el proceso de implementación del proyecto.
- Implementar los múltiples servicios requeridos para SEAP-UTA.
- Realizar pruebas y analizar los resultados obtenidos del proyecto.
- Establecer un acuerdo de aprobación del cliente con respecto al producto realizado.

3.7. Requisitos de alto nivel

A continuación se tiene una lista con las funcionalidades que debe contener el sistema, con respecto a los usuarios, es decir, alumnos y docentes:

- Las cuentas serán creadas por los usuarios administradores del sistema, las cuales serán de alumnos o docentes.
- Solo los usuarios registrados tendrán acceso a la plataforma.
- Los usuarios podrán crear repositorios, los cuales permitirán gestionarlos a partir de carpetas y archivos.
- Los repositorios podrán ser compartidos a otros usuarios, los cuales podrán verlos o editarlos.
- Los usuarios podrán programar en un ambiente con un texto colaborativo, además podrán compilar o ejecutar código de programación en una consola embebida en la plataforma web del sistema.
- Todos los cambios ocurridos en el sistema tienen que ser guardados de forma automática y tiene que tener un registro de cambios realizados.
- Las cuentas de los docentes pueden crear repositorios de tipo aula virtual, los cuales vienen con funcionalidades extras, como carpeta para los recursos de las clases, gestión de estudiantes y realizar evaluaciones o talleres en tiempo real.
- El sistema debe ser montado en el servidor del departamento de la carrera.

3.8. Modelo de contexto

El siguiente diagrama muestra el modelo de contexto de la solución propuesta:

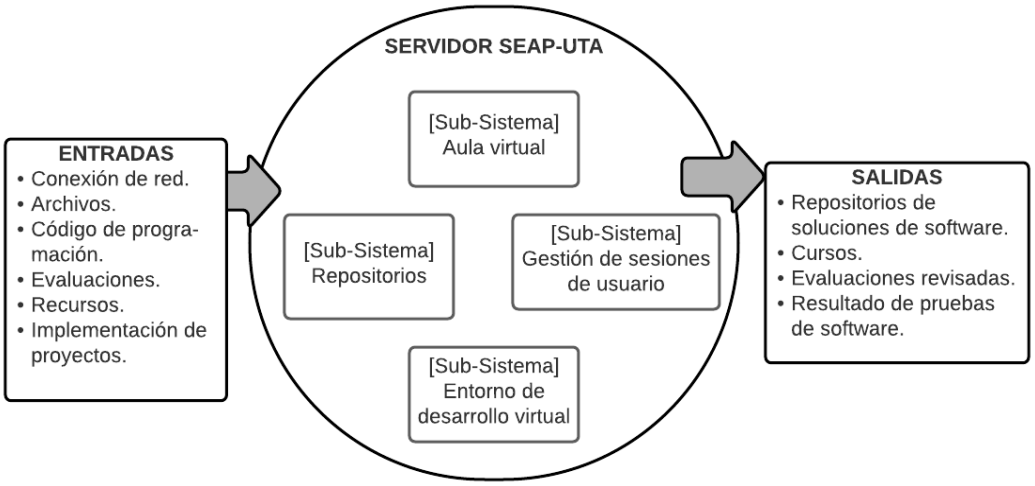

Figura 2. Modelo de contexto del sistema SEAP-UTA.

Como se observa en la figura del modelo contexto, el servidor de la solución de software SAEP-UTA, cuenta con 4 subsistemas para su funcionamiento, los cuales a través de las diversas entradas mencionadas, procesaran la información y lograrán realizar las salidas correspondientes.

4. PLANIFICACIÓN DEL PROYECTO

4.1. Hitos y entregables del proyecto

A continuación se muestra una tabla con los entregables e hitos del proyecto:

Tabla 2. Hitos y entregables del proyecto.

Hito o entregable	Fecha	Descripción
Inicio del proyecto	6 de septiembre	- Inicio y planteamiento del proyecto individual del ramo.
Parte 1	18 de octubre	- Descripción inicial del proyecto. - Diseño y modelado de software. - Presentación inicial del proyecto. - Diseño y modelado del sistema distribuido.
Parte 2	8 de noviembre	- Selección de herramientas para el desarrollo de software. - Diseño y desarrollo de la arquitectura de software. - Desarrollo de la implementación del proyecto.
Parte 3	6 de diciembre	- Desarrollo de la implementación del proyecto. - Manual de usuario. - Aprobación del cliente. - Presentación del producto.
Parte 4	15 de diciembre	- Fin del proyecto.

4.2. Planificación temporal mediante Carta Gantt

Para la planificación temporal del proyecto, se ha definido una carta Gantt con las tareas para cada etapa del proyecto definida por el encargado de la asignatura, como se muestra a continuación:

Análisis de la problemática y propuesta del proyecto:

Análisis de la problemática y propuesta de proyecto	Septiembre				Octubre				Noviembre					Diciembre			
	6	13	20	27	4	11	18	25	1	8	15	22	29	6	13	20	27
Planteamiento de la problemática																	
Planteamiento de la solución																	
Establecimiento de comunicación con el cliente																	
Definición de requisitos funcionales																	
Definición de casos de uso de sistema																	
Diseño de diagarmas de estado																	
Diseño de clases																	
Diseño de modelo estático del software																	
Diseño del diagrama de contexto																	
Diseño de BPM del proyecto																	
Desarrollo de informe y presentación																	

Figura 3. Carta Gantt. Etapa de análisis de la problemática y propuesta del proyecto.

Análisis y diseño de la solución:

Análisis y diseño de la solución	Septiembre				Octubre				Noviembre					Diciembre			
	6	13	20	27	4	11	18	25	1	8	15	22	29	6	13	20	27
Selección de herramientas para la implementación																	
Definición de alcance del producto																	
Diseño de diagramas de interacción																	
Diseño de base de datos																	
Definir modelo de análisis																	
Desarrollo de informe y presentación																	

Figura 4. Carta Gantt. Etapa de Análisis y diseño de la solución.

Implementación:

Implementación	Septiembre				Octubre				Noviembre					Diciembre			
	6	13	20	27	4	11	18	25	1	8	15	22	29	6	13	20	27
Levantamiento del ambiente de desarrollo																	
Desarrollo Back-End																	
* Levantamiento de la base de datos																	
* Comunicación con la base de datos																	
* Definición de consultas SQL																	
* Definición de la API REST																	
* Implementación de consola SSH con compilador																	
* Implementación de editor de colaborativo																	
* Implementación de manejo de archivos																	
Desarrollo Front-End																	
* Desarrollo de máqueta de páginas web																	
* Definición de rutas de direccionamiento																	
* Definición de utilización de datos																	
* Implementación de editor colaborativo																	
* Implementación de consola virtual																	
* Comunicación del Front-End y Back-End																	
Documentación en GitHub																	
Documentación de implementación del proyecto																	
Documentación de análisis y pruebas de resultados																	

Figura 5. Carta Gantt. Etapa de implementación.

Cierre del proyecto:

Cierre del proyecto	Septiembre				Octubre				Noviembre					Diciembre			
	6	13	20	27	4	11	18	25	1	8	15	22	29	6	13	20	27
Desarrollo de informe y presentación final																	
Documentación del manual de usuario																	
Aprobación del cliente																	
Presentación del producto																	
Cierre del proyecto																	

Figura 6. Carta Gantt. Etapa de cierre del proyecto.

Comunicación con el cliente:

Comunicación con el cliente	Septiembre				Octubre				Noviembre					Diciembre			
	6	13	20	27	4	11	18	25	1	8	15	22	29	6	13	20	27
Reuniones y comunicación vía correo eléctronico																	
Aprobación de requerimientos																	

Figura 7. Carta Gantt. Etapa de comunicación con el cliente.

4.4. Metodología de trabajo

Para la ejecución del proyecto se utilizará el modelo de desarrollo de software basado en Scrum, además de Kanban como soporte de control de actividades.

El modelo de proceso de Scrum en conjunto a Kanban consta del siguiente sprint:

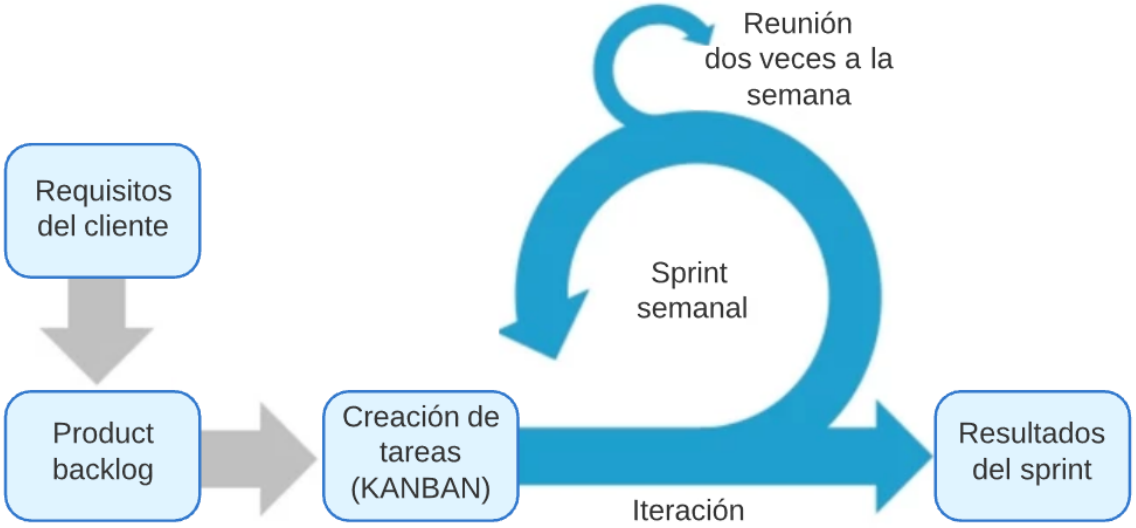

Figura 8. Modelo de proceso de scrum implementado en el proyecto.

El modelo presentado en la figura 8 utiliza Kanban para la creación y distribución de tareas, como se muestra en la siguiente figura:

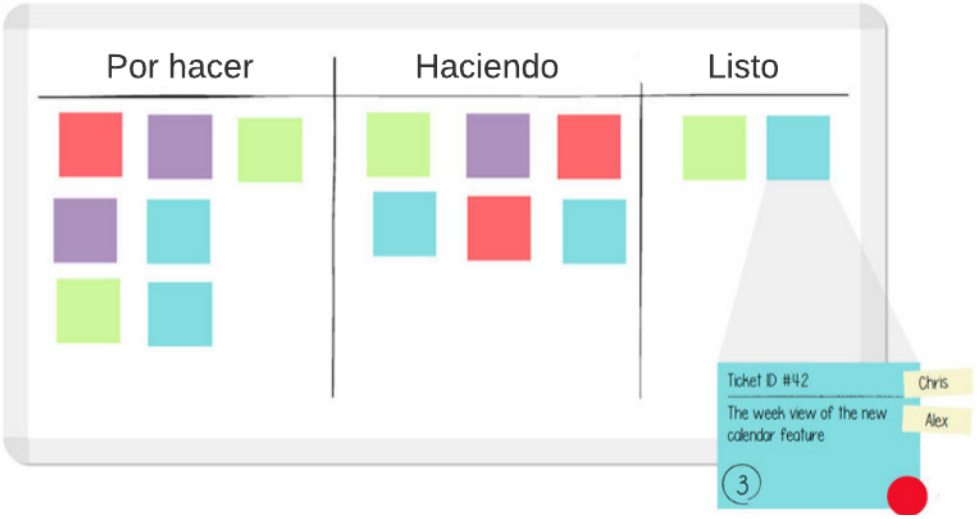

Figura 9. Referencia de utilización de Kanban.

El control y gestión de actividades mencionadas en el modelo de proceso se realiza mediante la plataforma online Redmine, actualmente en funcionamiento dentro del departamento de Ingeniería en Computación e Informática de la Universidad de Tarapacá.

4.5. Herramientas de gestión del proyecto

Para la gestión completa del proyecto, se utilizaron las siguientes herramientas:

Google Drive: Servicio de Google para el almacenamiento y gestión de archivos, este cuenta con diversas herramientas de elaboración de documentos, tales como; informes, presentaciones, diagramas, hojas de cálculo, etc. En el proyecto se utilizará para la elaboración de la documentación y el control del tiempo mediante la carta Gantt.

LucidChart: Herramienta web para la elaboración de múltiples tipos de diagramas, tales como, UML, prototipos de software, arquitecturas distribuidas, etc. LucidChart será la principal herramienta de modelado utilizada en el diseño del proyecto.

Trello: Es una solución fácil, gratuita, flexible y visual para gestionar proyectos y organizar tareas.

Redmine: Herramienta para la gestión de proyectos, que con sus diversas funcionalidades permite a los usuarios de diferentes proyectos realizar el seguimiento y organización de los mismos. En el proyecto es utilizado para dar a conocer el progreso a los interesados.

5. ANÁLISIS Y DISEÑO DE SOFTWARE

5.1. Definición de requisitos del proyecto

5.1.1. Requerimientos funcionales

La siguiente tabla muestra los requerimientos funcionales que se deben tener en cuenta en el diseño del proyecto, estos fueron extraídos de los requisitos y fueron validados por el cliente. La nomenclatura RFN significa Requisito Funcional Número, la cual será referenciada para las dependencias en el diseño de casos de uso.

Tabla 3. Numeración y descripción de los requerimientos funcionales del proyecto.

RFN	Descripción del requerimiento funcional
RF1	El administrador de la plataforma podrá crear cuentas de usuario para los alumnos y los docentes, a través de un nombre de usuario, contraseña y un correo de contacto.
RF2	El usuario podrá iniciar sesión en la plataforma a través de un usuario y contraseña.
RF3	El usuario puede crear repositorios en su cuenta, para ello se requerirá el nombre del repositorio, una descripción (de manera opcional) y la visibilidad (privada o pública). A este usuario creador, se le asigna el rol de “Creador de repositorio”.
RF4	Los repositorios privados, solo podrán ser vistos por los miembros del repositorio, al contrario de los repositorios públicos, en donde cualquier usuario registrado puede ver el repositorio con el respectivo enlace.
RF5	El usuario creador o administrador de un repositorio podrá agregar a otros usuarios al repositorio, definiendo roles, como; administrador (solo el creador puede definirlos), editor y lector (en caso de un repositorio privado).
RF6	Los roles para la gestión de un repositorio son los siguientes: **Creador del repositorio**, es el usuario que ha creado el repositorio, el cual le permite poder eliminar el repositorio de la plataforma, además este posee las capacidades de los otros roles. **Administrador**, va a ser el encargado de gestionar los miembros y roles del equipo (excepto el rol de creador), además de tener las funciones de los roles editor y lector. **Editor**, este puede gestionar los elementos del repositorio, es decir, carpetas y archivos, además de tener las capacidades del rol lector. **Lector**, tiene la posibilidad de ver la información de todos los aspectos y elementos del repositorio.
RF7	Un usuario con permisos de lector, puede descargar los archivos o carpetas del repositorio.
RF8	Los usuarios editores podrán gestionar los repositorios a través de elementos, es decir, archivos y carpetas. En general un elemento podrá ser creado, modificado, y eliminado.
RF9	Los usuarios que tengan permisos de editor, al acceder a algún archivo editable dentro de un repositorio que tenga una extensión reconocida por la plataforma, podrán entrar a una sesión colaborativa, en donde podrán editar el documento respectivo y realizar pruebas. En caso de

	tener permisos de lector, entrará a un editor de texto estático de lectura.
RF10	El sistema deberá mostrar los cambios realizados en la hoja de trabajo en tiempo real para todos los usuarios que estén colaborando.
RF11	Los archivos y sus cambios quedarán guardados en los respectivos repositorios a los que pertenecen.
RF12	Dentro del editor de texto colaborativo o el estático, si la plataforma tiene la característica de poder compilar y ejecutar el código de programación correspondiente, esta debe mostrar la característica de una consola virtual, en donde el usuario a través de un botón, podrá mandar el código y ejecutarlo en la misma.
RF13	Los usuarios que sean docentes tendrán la posibilidad de crear repositorios con funcionalidades especiales para la gestión de cursos, llamados “aula virtual”, para esto se requerirá el nombre del curso, el año y el semestre correspondiente.
RF14	Un docente creador de una aula virtual, podrá agregar a los alumnos del curso que por defecto, serán lectores del repositorio.
RF15	En una aula virtual, el docente podrá crear evaluaciones, para ello, se requiere, un enunciado mediante una descripción o documento, la fecha/hora de inicio y de fin, cantidad de alumnos por evaluación.
RF16	El usuario alumno al abrir una evaluación, verá el enunciado y las condiciones de la evaluación. En caso de que sea en equipo, se generará un código de equipo, al cual el o los otros estudiantes tendrán que ingresar para formar el equipo. Luego al iniciar, se crea un repositorio privado para el alumno o alumnos participantes donde tendrán que desarrollar la solución.
RF17	Al finalizar el tiempo de evaluación, los estudiantes no podrán editar más el repositorio.
RF18	Luego de haber finalizado la evaluación, el docente podrá ver las soluciones de los alumnos en una carpeta privada del curso, que solo él tendrá acceso.
RF19	Los alumnos podrán ver sus soluciones realizadas en evaluaciones anteriores.
RF20	El docente puede modificar las evaluaciones que ha creado, incluso si se encuentra en curso.

5.1.2. Requerimientos no funcionales

A continuación se listan los requerimientos no funcionales del proyecto, esto quiere decir que, no aportan una funcionalidad como tal al proyecto.

- Si la extensión del lenguaje de programación utilizado en el editor de texto colaborativo es reconocido por la plataforma, este debe tener su sintaxis reflejada (colores en variables, funciones, control, etc).
- El sistema debe estar en el servidor del Departamento de Ingeniería Civil en Computación e Informática de la Universidad de Tarapacá.

5.2. Definición de casos de uso

5.2.1. Diagrama general de casos

Para la definición del modelo de casos de uso se ha dividido la solución del sistema SEAP-UTA en cuatro paquetes de casos de uso, los cuales buscan separar cada una de las funcionalidades generales de la plataforma.

5.2.1.1. Paquete de casos de uso: Gestión de usuario

El siguiente paquete de casos de uso, permite el manejo de sesiones de usuario además de la creación de cuentas de usuario, los cuales pueden ser docentes o alumnos del departamento de la carrera de la organización.

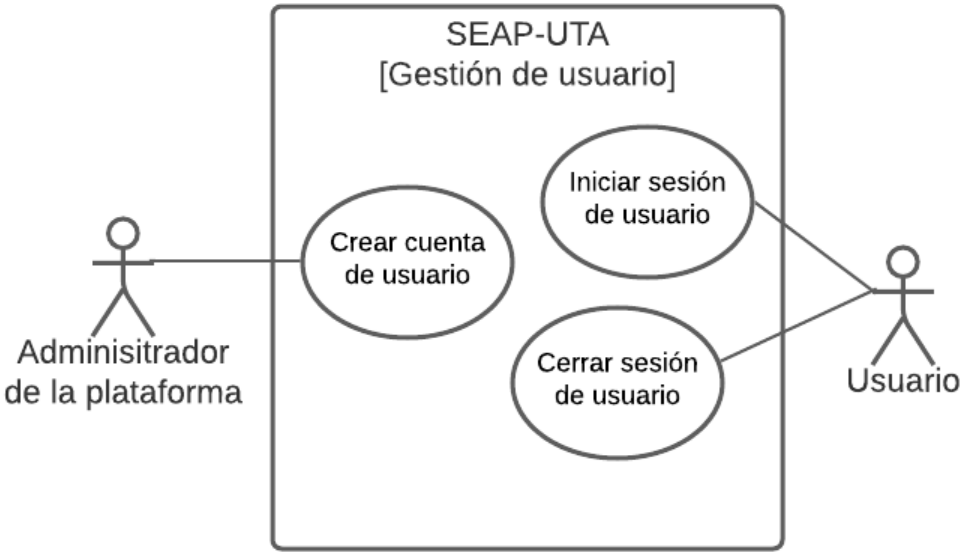

Figura 10. Diagrama general de casos de uso, Paquete de Gestión de usuario.

Definición de actores:

Administrador de la plataforma: Consiste en la persona encargada de la plataforma, ya sea de forma física o virtual.

Usuario: Un usuario de la plataforma puede ser tanto un alumno o docente de la carrera.

5.2.1.2. Paquete de casos de uso: Gestión de repositorio

Este paquete de casos de uso contempla todos los aspectos necesarios para la gestión de repositorios del sistema SEAP-UTA, contemplando las funcionalidades de todos los roles asociados definidos en los requerimientos.

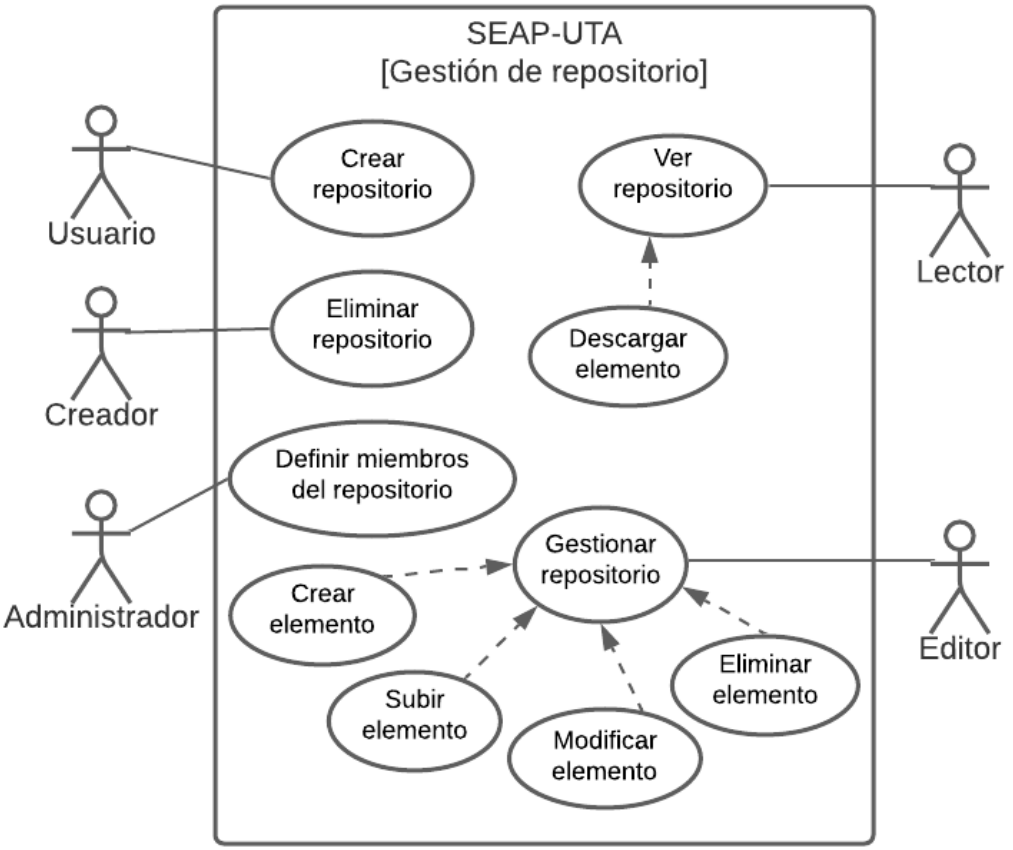

Figura 11. Diagrama general de casos de uso, paquete de casos de uso Gestión de repositorio.

Definición de actores

Administrador: Usuario, que utiliza el rol de administrador dentro de un repositorio, que le permite gestionar los miembros y roles del equipo (excepto el rol de creador), además de tener las capacidades de los roles editor y lector.

Editor: Usuario, que utiliza el rol de editor dentro de un repositorio, el cual puede gestionar los elementos del repositorio, es decir, carpetas y archivos, además de tener las capacidades del rol lector.

Lector: Usuario, que utiliza el rol de lector dentro de un repositorio, el cual tiene la posibilidad de ver la información de todos los aspectos y elementos del repositorio.

5.2.1.3. Paquete de casos de uso: Gestión de aula virtual

El paquete de casos de uso contempla la utilidad principal de ayuda a la docencia, la cual contempla un ambiente virtual llamado aula virtual, el cual permitirá al alumno ver sus recursos y realizar evaluaciones, y al docente poder gestionar el curso y evaluar a los estudiantes.

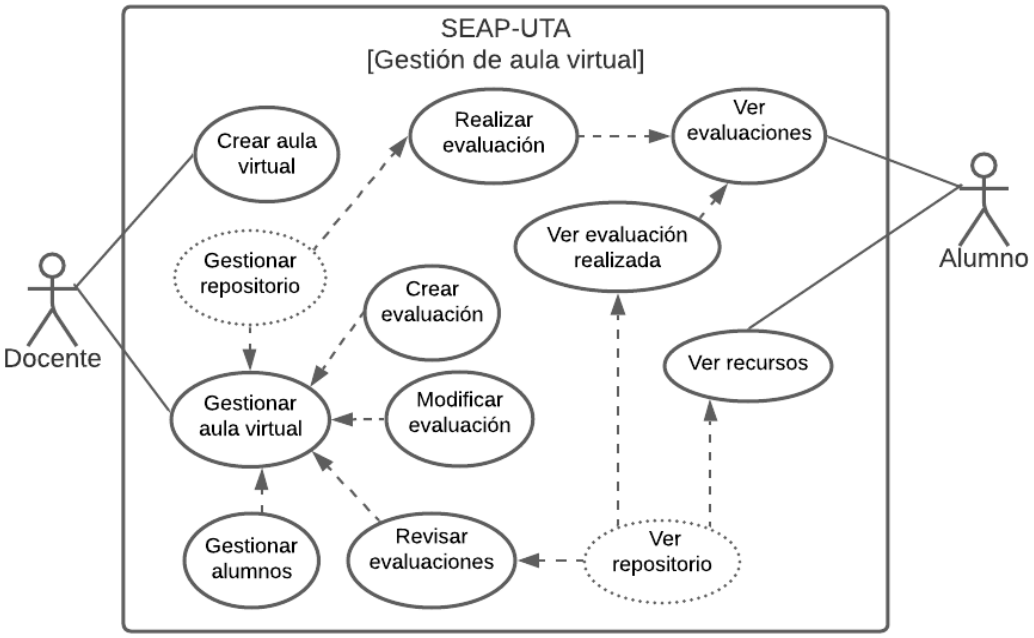

Figura 12. Diagrama de casos de uso general, paquete de casos de uso de Gestión de aula virtual.

Definición de actores

La identificación de este tipo de usuario es definida por el administrador de la plataforma al crear la cuenta de usuario del docente o estudiante.

Docente: Usuario definido como docente, el cual puede crear y gestionar aulas virtuales.

Alumno: Usuario calificado como alumno, este puede ver los recursos de las aulas virtuales a las que pertenezca y poder realizar las evaluaciones pertinentes del curso.

5.2.1.4. Paquete de casos de uso: Programación colaborativa

El siguiente paquete de programación colaborativa, muestra donde un usuario dentro de un archivo de código de programación en un repositorio, pueda programar colaborativamente y probar el código del repositorio mediante un compilador embebido. Una de las utilidades del sistema más importante para los alumnos.

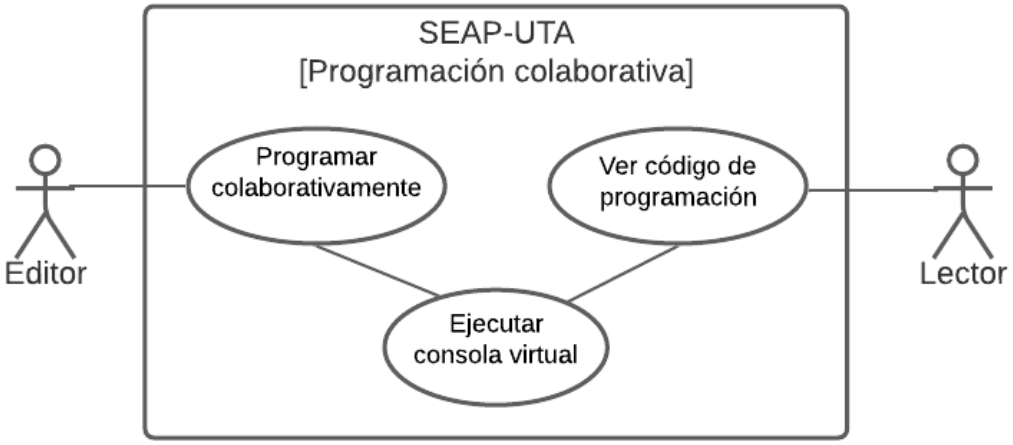

Figura 13. Diagrama de casos de uso general, paquete de casos de uso de Programación colaborativa.

Definición de actores

Editor: Usuario, que utiliza el rol de editor dentro de un repositorio, el cual puede gestionar los elementos del repositorio, es decir, carpetas y archivos, en donde podrá editar los archivos de código de programación utilizando un editor colaborativo y probar el código del repositorio a través de la consola virtual.

Lector: Usuario que utiliza el rol de lector dentro de un repositorio, el cual puede ver todos los aspectos de este, además podrá acceder al editor colaborativo, sin poder editar, y podrá utilizar la consola virtual para probar el código del repositorio.

5.3. Descripción de casos de uso

En el siguiente punto se muestran las tablas de descripciones de cada uno de los casos de uso mencionados en el punto anterior.

Tabla 4. Descripción de caso de uso. Paquete de casos de uso - Gestión de usuario: Crear cuenta de usuario.

Nombre	Crear cuenta de usuario	
Dependencias	RF1.	
Precondición	Entrar al sitio web de SEAP-UTA y solicitar crear una nueva cuenta de usuario.	
Actor	Administrador de la plataforma.	
Descripción	La plataforma web solicita al usuario los datos necesarios para crear una cuenta en la plataforma, sea de docente o alumno.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la página web de registro de cuenta.
	2	La página web le entrega un formulario, solicitando; tipo de usuario (docente o alumno), correo electrónico, nombre, apellido, nombre de usuario y contraseña de la cuenta.
	3	El usuario ingresa los datos solicitados y confirma crear una nueva cuenta.
	4	La aplicación muestra que se ha creado la cuenta correctamente.
Postcondición	Se ha creado la cuenta de usuario.	
Excepciones	4	La aplicación web le indica al usuario que hay un error en alguna entrada.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 5. Descripción de caso de uso. Paquete de casos de uso - Gestión de usuario: Iniciar sesión de usuario.

Nombre	Iniciar sesión de usuario.	
Dependencias	RF2.	
Precondición	Entrar al sitio web de la plataforma SEAP-UTA, sin haber iniciado sesión anteriormente. Se ha creado la cuenta de usuario.	
Actor	Usuario.	
Descripción	La plataforma web solicita al usuario iniciar sesión para utilizar las funcionalidades respectivas, a través de un nombre de usuario y contraseña.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la plataforma web de SEAP-UTA.
	2	La página web le entrega un formulario, solicitando el nombre de usuario y contraseña de la cuenta.
	3	El usuario ingresa los datos solicitados y confirma el iniciar sesión.
	4	La aplicación web redirige al usuario a la página principal de SEAP-UTA.
Postcondición	El usuario ha iniciado sesión.	
Excepciones	4	La aplicación web le indica al usuario que hay un error en alguna entrada.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 6. Descripción de caso de uso. Paquete de casos de uso - Gestión de usuario: Cerrar sesión de usuario.

Nombre	Cerrar sesión de usuario.	
Dependencias	RF2.	
Precondición	Entrar al sitio web de la plataforma SEAP-UTA, y haber iniciado sesión anteriormente.	
Actor	Usuario.	
Descripción	El usuario solicita a la plataforma web cerrar la sesión de su cuenta.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la plataforma web de SEAP-UTA y escoge cerrar sesión.
	2	El sistema SEAP-UTA cierra la sesión del usuario y le muestra un mensaje en la página web. Y la aplicación web redirige al usuario a la página principal del sitio.
Postcondición	El usuario ha cerrado sesión.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 7. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Crear repositorio.

Nombre	Crear repositorio.	
Dependencias	RF3 y RF4.	
Precondición	El usuario ha iniciado sesión.	
Actor	Usuario.	
Descripción	El usuario crea un repositorio, aquí se define el nombre, una descripción (opcional) y la visibilidad de este (público o privado), además al usuario se le asigna el rol de creador.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la dirección de “Crear repositorio”, y selecciona crear un nuevo repositorio.
	2	La plataforma le solicita, el nombre del repositorio, una descripción opcional y si la visibilidad es pública o privada.
	3	El usuario indica el nombre respectivo, la descripción y la privacidad respectiva.
	4	El sistema crea el repositorio con el nombre y descripción respectiva y le otorga la propiedad de visibilidad establecida, pública o privada. Además, se define como miembro creador al usuario.
Postcondición	Repositorio creado por un usuario. Usuario con rol de creador. Privacidad de repositorio establecida.	
Excepción	4	La aplicación web le indica al usuario que hay un error en alguna entrada.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 8. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Definir miembros del repositorio.

Nombre	Definir miembros del repositorio.	
Dependencias	RF5, RF6.	
Precondición	Repositorio creado o administrado por un usuario. El usuario ha iniciado sesión.	
Actor	Administrador.	
Descripción	Se definen o modifican los miembros y roles para el repositorio respectivo. Tales roles definen las funcionalidades disponibles para los usuarios.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ha ingresado a definir miembros del repositorio.
	2	La plataforma web le lista los miembros actuales del repositorio, y le solicita modificar la lista (agregar, eliminar o cambiar rol de un miembro), en donde se muestra el nombre de usuario y rol correspondiente. Además, le pide la confirmación de sus acciones.
	3	El usuario realiza las modificaciones correspondientes y confirma sus acciones.
	4	El sistema SEAP-UTA actualiza los cambios a los miembros del repositorio.
Postcondición	Miembros y roles de un repositorio definidos.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 9. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Gestionar repositorio.

Nombre	Gestionar repositorio.		
Dependencias	RF8.		
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Gestionar repositorio”. El usuario tiene permisos de editor en el repositorio. El usuario se encuentra dentro de una carpeta de un repositorio.		
Actor	Usuario.		
Descripción	El usuario que tenga el rol de editor, gestiona el repositorio, ya sea creando, eliminando o editando archivos o carpetas llamadas elementos del repositorio.		
Secuencia normal	**Paso**	**Acción**	
	1	El usuario ingresa al repositorio y la dirección que desea gestionar, ya sea en la carpeta raíz o dentro de una carpeta.	
	2	La plataforma espera a que el usuario realice una acción para gestionar algún elemento.	
	3	Si el usuario desea crear un elemento.	
		3.1	Entonces, <<Se incluye el C.U.S. Crear elemento>>.
	4	Si el usuario desea subir un elemento.	
		4.1	Entonces, <<Se incluye el C.U.S. Subir elemento>>.
	5	Si el usuario desea modificar un elemento.	
		5.1	Entonces, <<Se incluye el C.U.S. Modificar elemento>>.
	6	Si el usuario desea eliminar un elemento.	
		6.1	Entonces, <<Se incluye el C.U.S. Eliminar elemento>>.
Postcondición	Modificación al repositorio realizada.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 10. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Crear elemento.

Nombre	Crear elemento.		
Dependencias	RF8.		
Precondición	El usuario ha iniciado sesión. El usuario se encuentra dentro de una carpeta de un repositorio. El usuario ha solicitado crear un elemento. El usuario tiene permisos de editor en el repositorio.		
Actor	Editor.		
Descripción	Permite al usuario editor de un repositorio, crear un elemento, es decir, un archivo o carpeta según lo requiera.		
Secuencia normal	**Paso**	**Acción**	
	1	La plataforma le solicita el tipo de elemento que desea crear.	
	2	Si el usuario desea crear una carpeta	
		2.1	El usuario solicita crear una carpeta.
		2.2	La plataforma le solicita el nombre de la carpeta y confirmar su acción.
		2.3	El usuario ingresa lo solicitado y confirma la acción.
		2.4	El sistema SEAP-UTA agrega la nueva carpeta y actualiza la vista del usuario.
	3	Si el usuario desea crear un archivo	
		3.1	El usuario solicita crear un archivo.
		3.2	La plataforma le solicita el tipo de extensión del archivo, el nombre y la confirmación de su acción.
		3.3	El usuario ingresa lo solicitado y confirma la acción.
		3.4	El sistema SEAP-UTA agrega el nuevo archivo y actualiza la vista del usuario.
Postcondición	Modificación al repositorio realizada.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 11. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Subir elemento.

Nombre	Subir elemento.	
Dependencias	RF8.	
Precondición	El usuario ha iniciado sesión. El usuario se encuentra dentro de una carpeta de un repositorio. El usuario ha solicitado subir un elemento. El usuario tiene permisos de editor en el repositorio.	
Actor	Editor.	
Descripción	Permite al usuario editor de un repositorio, subir un elemento, es decir, un archivo o carpeta según lo requiera.	
Secuencia normal	**Paso**	**Acción**
	1	La plataforma le solicita seleccionar el elemento que desea subir.
	2	El usuario selecciona lo que desea subir desde su entorno de trabajo.
	3	La plataforma recibe el elemento que desea subir el usuario y lo agrega al repositorio.
Postcondición	Modificación al repositorio realizada.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 12. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Modificar elemento.

Nombre	Modificar elemento.		
Dependencias	RF8.		
Precondición	El usuario ha iniciado sesión. El usuario se encuentra dentro de una carpeta de un repositorio. El usuario ha solicitado subir un elemento. El usuario tiene permisos de editor en el repositorio.		
Actor	Editor.		
Descripción	Permite al usuario editor de un repositorio, subir un elemento, es decir, un archivo o carpeta según lo requiera.		
Secuencia normal	**Paso**	**Acción**	
	1	La plataforma le solicita seleccionar el elemento que desea modificar.	
	2	El usuario selecciona el elemento que desea modificar.	
	3	La plataforma le muestra el tipo de modificación disponible, cambiar nombre o mover el elemento dentro del repositorio.	
	4	Si el usuario desea cambiar el nombre del elemento.	
		4.1	La plataforma le solicita el nuevo nombre del elemento y la confirmación de la acción.
		4.2	El usuario escribe el nuevo nombre del elemento y confirma la acción.
		4.3	El sistema actualiza el nombre del elemento correspondiente.
	5	Si el usuario desea mover el elemento dentro del repositorio	
		5.1	La plataforma le solicita la dirección a la cual desea mover el elemento dentro del repositorio y la confirmación correspondiente.
		5.2	El usuario selecciona la nueva dirección del elemento y confirma lo hecho.
		5.3	El sistema modifica la dirección del elemento correspondiente.
Postcondición	Modificación al repositorio realizada.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 13. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Eliminar elemento.

Nombre	Eliminar elemento.	
Dependencias	RF8.	
Precondición	El usuario ha iniciado sesión. El usuario se encuentra dentro de una carpeta de un repositorio. El usuario ha solicitado eliminar un elemento. El usuario tiene permisos de editor en el repositorio.	
Actor	Editor.	
Descripción	Permite al usuario editor de un repositorio, eliminar un elemento, es decir, un archivo o carpeta según lo requiera.	
Secuencia normal	**Paso**	**Acción**
	1	La plataforma le solicita la confirmación de la eliminación del elemento seleccionado.
	2	El usuario confirma la acción.
	3	El sistema realiza la eliminación del elemento correspondiente y actualiza el repositorio.
Postcondición	Modificación al repositorio realizada.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 14. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Ver repositorio.

Nombre	Ver repositorio.	
Dependencias	RF6.	
Precondición	Ser miembro de un repositorio privado o acceder a un repositorio público.	
Actor	Lector.	
Descripción	Ver un repositorio consiste en poder ver el contenido de este. Para ello es necesario ser un miembro del repositorio con las funcionalidades de lector, en caso de que el repositorio sea privado. Por otra parte, si el repositorio es público, todos los usuarios registrados en la plataforma pueden acceder a este.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a un repositorio que es miembro o uno público.
	2	La plataforma muestra el repositorio mostrando los elementos correspondientes de este, además le permite navegar dentro de este al usuario.
Postcondición	El usuario se encuentra dentro de una carpeta de un repositorio.	
Importancia	Importante.	
Prioridad	Alta.	

Tabla 15. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Descargar elemento.

Nombre	Descargar elemento.	
Dependencias	RF7.	
Precondición	El usuario ha iniciado sesión. El usuario se encuentra dentro de una carpeta de un repositorio. El usuario tiene permisos de lector en el repositorio.	
Actor	Editor.	
Descripción	Permite al usuario editor de un repositorio, descargar un elemento, es decir, un archivo o carpeta según lo requiera.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario escoge un elemento dentro del repositorio.
	2	La plataforma muestra las opciones disponibles para el usuario, contemplando la descarga del elemento.
	3	El usuario escoge descargar el elemento.
	4	El sistema le entrega el elemento correspondiente a través de una descarga.
Postcondición	Descarga de un archivo o carpeta del repositorio.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 16. Descripción de caso de uso. Paquete de casos de uso - Gestión de repositorio: Eliminar repositorio.

Nombre	Eliminar repositorio.	
Dependencias	RF6.	
Precondición	El usuario ha iniciado sesión. El usuario se encuentra dentro de un repositorio. El usuario tiene permisos de creador en el repositorio.	
Actor	Creador.	
Descripción	Permite al usuario creador de un repositorio, eliminarlo permanentemente de la plataforma.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario selecciona la opción de eliminar repositorio.
	2	La plataforma le muestra al usuario si está seguro de querer borrar el repositorio permanentemente.
	3	El usuario confirma su acción.
	4	El sistema borra el repositorio de la plataforma y redirige al usuario.
Postcondición	Eliminación de un repositorio.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 17. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Crear aula virtual.

Nombre	Crear aula virtual.	
Dependencias	RF3 y RF4.	
Precondición	El usuario ha iniciado sesión.	
Actor	Docente.	
Descripción	El usuario docente crea una aula virtual, aquí se define el nombre del curso, el año y semestre correspondiente. En sí el ambiente de aula virtual, es un repositorio con características para ayudar a la gestión de cursos.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la dirección de “Crear aula virtual”.
	2	La plataforma le solicita, el nombre del curso, el año, y el semestre correspondiente.
	3	El usuario ingresa los campos solicitados.
	4	El sistema crea el aula virtual con el nombre y descripción respectiva (semestre y año) y le otorga la propiedad de visibilidad privada al repositorio del aula virtual. Además se le asigna al usuario como rol administrador del repositorio del curso.
Postcondición	Aula virtual creada por un docente. Usuario con rol de creador. Privacidad de repositorio establecida.	
Excepción	4	La aplicación web le indica al usuario que hay un error en alguna entrada.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 18. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Gestionar aula virtual.

Nombre	Gestionar aula virtual.		
Dependencias	RF14, RF15, RF18 y RF20.		
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de "Gestionar aula virtual". El usuario es el administrador de una aula virtual.		
Actor	Docente.		
Descripción	El usuario docente, gestiona su aula virtual, ya sea creando, gestionando recursos, alumnos y evaluaciones.		
Secuencia normal	**Paso**	**Acción**	
	1	El usuario ingresa al aula virtual y escoge una de las opciones de gestión.	
	2	La plataforma espera a que el usuario realice una acción para gestionar algún elemento.	
	3	Si el usuario desea gestionar recursos.	
		3.1	Entonces, <<Se incluye el C.U.S. Gestionar repositorio>>.
	4	Si el usuario desea crear una evaluación.	
		4.1	Entonces, <<Se incluye el C.U.S. Crear evaluación>>.
	5	Si el usuario desea revisar evaluaciones realizadas.	
		5.1	Entonces, <<Se incluye el C.U.S. Revisar evaluaciones>>.
	6	Si el usuario desea modificar una evaluación.	
		6.1	Entonces, <<Se incluye el C.U.S. Modificar evaluación>>.
Postcondición	Modificación al aula virtual realizada.		
Importancia	Muy importante.		
Prioridad	Alta.		

Nota: El C.U.S. Gestionar repositorio es el mismo que el que se encuentra en el paquete de C.U.S. Gestión de repositorio.

Tabla 19. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Gestionar alumnos.

Nombre	Gestionar alumnos.	
Dependencias	RF14.	
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Gestionar aula virtual”. El usuario es el administrador de una aula virtual.	
Actor	Administrador.	
Descripción	Se definen o modifican los alumnos del curso respectivo. En definición de un repositorio, los alumnos serán lectores que podrán acceder a los recursos del aula virtual.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ha ingresado a definir alumnos del aula virtual.
	2	La plataforma web le lista los alumnos actuales del aula virtual, y le solicita modificar la lista (agregar o eliminar), en donde se muestra el nombre de usuario del alumno. Además, le pide la confirmación de sus acciones.
	3	El usuario realiza las modificaciones correspondientes y confirma sus acciones.
	4	El sistema DAEP-UTA actualiza los cambios a los alumnos del aula virtual.
Postcondición	Alumnos del aula virtual modificados.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 20. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Crear evaluación.

Nombre	Crear evaluación.	
Dependencias	RF15.	
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Gestionar aula virtual”. El usuario es el administrador de una aula virtual.	
Actor	Docente.	
Descripción	El usuario docente crea una evaluación, la cual se ejecutará según lo programado, para esto es necesario; un enunciado, ya sea mediante una descripción o archivo, fecha/hora de inicio y de fin, además de la cantidad de alumnos de la evaluación.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la dirección de “Crear evaluación”.
	2	La plataforma le solicita, nombre de la evaluación, el enunciado, ya sea con una descripción o archivo, fecha/hora de inicio y de fin.
	3	El usuario ingresa los campos solicitados.
	4	El sistema crea la evaluación en el aula virtual del curso.
Postcondición	Evaluación definida.	
Excepción	4	La aplicación web le indica al usuario que hay un error en alguna entrada.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 21. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Modificar evaluación.

Nombre	Modificar evaluación.	
Dependencias	RF15.	
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Gestionar aula virtual”. El usuario es el administrador de una aula virtual.	
Actor	Docente.	
Descripción	El usuario docente modifica una evaluación, ya sea; el enunciado, fecha/hora de inicio y de fin o la cantidad de alumnos por evaluación.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la dirección de “Modificar evaluación”.
	2	La plataforma le solicita si desea modificar; nombre de la evaluación, el enunciado, ya sea con una descripción o archivo, fecha/hora de inicio y de fin.
	3	El usuario ingresa los campos solicitados.
	4	El sistema modifica la evaluación en el aula virtual del curso.
Postcondición	Evaluación modificada.	
Excepción	4	La aplicación web le indica al usuario que hay un error en alguna entrada.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 22. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Revisar evaluaciones.

Nombre	Revisar evaluaciones.	
Dependencias	RF18.	
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Revisar evaluaciones”. El usuario es el administrador de una aula virtual.	
Actor	Docente.	
Descripción	El usuario desea revisar las evaluaciones realizadas por los alumnos, aquí podrá observar y probar todos los códigos de programación realizados en sus respectivos repositorios.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario ingresa a la dirección de “Revisar evaluaciones”.
	2	La plataforma le muestra todas las evaluaciones pasadas ya realizadas, espera a que el usuario seleccione alguna.
	3	El usuario escoge una evaluación.
	4	El sistema le muestra una lista de las soluciones de los alumnos realizadas, mostrando los integrantes de esta, y espera a que se seleccione alguna.
	5	El usuario escoge una solución de evaluación.
	6	Entonces, <<se incluye el C.U.S. Ver repositorio>>.
Postcondición	Evaluación modificada.	
Importancia	Muy importante.	
Prioridad	Alta.	

Nota: El C.U.S. Ver repositorio, es el mismo que el que se encuentra en el paquete de C.U.S. Gestión de repositorio.

Tabla 23. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Ver evaluaciones.

Nombre	Ver evaluaciones.		
Dependencias	RF16, RF17 y RF19.		
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Revisar evaluaciones”. El usuario es alumno de una aula virtual.		
Actor	Alumno.		
Descripción	El alumno ingresa a una aula virtual de algún curso para ver las evaluaciones correspondientes del curso.		
Secuencia normal	**Paso**	**Acción**	
	1	El usuario ingresa al aula virtual correspondiente y selecciona ver evaluaciones.	
	2	La plataforma le muestra todas las evaluaciones futuras, pendientes por hacer y las ya realizadas, espera a que el usuario seleccione alguna.	
	3	Si el usuario escoge una evaluación futura.	
		3.1	Entonces, la plataforma le muestra la fecha correspondiente de dicha evaluación.
	4	Si el usuario escoge una evaluación pendiente que tenga que realizar.	
		4.1	Entonces, <<Se incluye el C.U.S. Realizar evaluación>>.
	5	Si el usuario escoge una evaluación ya realizada.	
		5.1	Entonces, <<Se incluye el C.U.S. Ver evaluación realizada>>.
Postcondición	Evaluación modificada.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 24. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Realizar evaluación.

Nombre	Realizar evaluación.		
Dependencias	RF16 y RF17.		
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Revisar evaluaciones”. El usuario es alumno de una aula virtual.		
Actor	Alumno.		
Descripción	El alumno ingresa a una aula virtual de algún curso para realizar alguna evaluación correspondiente.		
Secuencia normal	**Paso**	**Acción**	
	1	El sistema le muestra la evaluación pendiente, mediante el nombre y fecha correspondiente y si se encuentra dentro del rango de fecha para su realización, le pregunta si desea comenzar la evaluación.	
	2	El usuario selecciona el comenzar la evaluación.	
	3	La plataforma le muestra la información de la evaluación, tales como nombre, fecha de inicio y de fin, tipo de evaluación (individual o en equipo), descripción del enunciado, archivos adjuntos.	
	4	Si la evaluación es en equipo.	
		4.1	La plataforma le muestra un código de equipo o un campo para ingresar un código de equipo al cual entrar.
	5	Si el usuario se une a un código de equipo.	
		5.1	El sistema agrega al estudiante al repositorio en el cual trabaja el equipo como rol de editor, se incluye el C.U.S. <<Gestión de repositorio>> .
	6	Si el usuario inicia la evaluación.	
		6.1	El sistema genera un repositorio en el cual va a trabajar el alumno el cual será editor de este, se incluye el C.U.S. <<Gestión de repositorio>> .
	7	Si se acaba el tiempo de la evaluación.	
		7.1	El sistema asigna a todos los usuarios que pertenecen al repositorio de la evaluación como lectores, es decir, les quita el permiso de poder seguir editando el repositorio.
Postcondición	Evaluación realizada.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 25. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Ver evaluación realizada.

Nombre	Ver evaluación realizada.	
Dependencias	RF19.	
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de “Revisar evaluaciones”. El usuario es alumno de una aula virtual.	
Actor	Alumno.	
Descripción	El alumno ingresa a una aula virtual de algún curso para ver una evaluación que ya ha realizado anteriormente.	
Secuencia normal	**Paso**	**Acción**
	1	El sistema le muestra la evaluación ya realizada, mediante le muestra la información de la evaluación, tales como nombre, fecha de inicio y de fin, tipo de evaluación (individual o en equipo), descripción del enunciado, archivos adjuntos, además, le solicita si desea ver la solución realizada.
	2	El usuario selecciona el ver solución realizada.
	3	Entonces, <<Se incluye el C.U.S. Ver repositorio>>.
Postcondición	Evaluación realizada.	
Excepción	3	Si el usuario no realizó la evaluación, la plataforma le mostrará un mensaje respectivo dando el aviso.
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 26. Descripción de caso de uso. Paquete de casos de uso - Gestión de aula virtual: Ver recursos.

Nombre	Ver recursos.	
Dependencias	RF13.	
Precondición	El usuario ha iniciado sesión. El usuario ha entrado al sitio de "Ver recursos" dentro del aula virtual. El usuario es alumno de una aula virtual.	
Actor	Alumno.	
Descripción	El alumno ingresa a una aula virtual de algún curso para ver los recursos.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario accede a ver los recursos del aula virtual.
	3	Entonces, <<Se incluye el C.U.S. Ver repositorio>>.
Postcondición	El alumno accede a ver los recursos del curso.	
Importancia	Muy importante.	
Prioridad	Alta.	

Tabla 27. Descripción de caso de uso. Paquete de casos de uso - Programación colaborativa: Programar colaborativamente.

Nombre	Programar colaborativamente.		
Dependencias	RF9, RF10 y RF11.		
Precondición	El usuario ha iniciado sesión. El usuario tiene permisos de editor en el repositorio del archivo.		
Actor	Usuario.		
Descripción	Uno o varios usuarios pueden programar colaborativamente en un código de programación mediante un editor de texto colaborativo de la plataforma.		
Secuencia normal	**Paso**	**Acción**	
	1	El usuario ingresa a un archivo de código de programación dentro de un repositorio.	
	2	La plataforma lo redirecciona a un editor de texto colaborativo, mostrando los detalles del documento visto. Está a la espera de la edición de alguno de los miembros que hayan accedido al documento.	
	3	Si el usuario ingresa un cambio en el código de programación.	
		3.1	La plataforma actualiza el estado del documento a los otros miembros, y la base de datos del sistema.
	4	Si otro miembro editor del repositorio realiza un cambio en el código de programación.	
		4.1	La plataforma actualiza el estado del documento del usuario.
Postcondición	Editor de texto colaborativo activo. Cambio realizado en el código de programación.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 28. Descripción de caso de uso. Paquete de casos de uso - Programación colaborativa: Ver código de programación.

Nombre	Ver código de programación.		
Dependencias	RF9 y RF10.		
Precondición	El usuario ha iniciado sesión. El usuario tiene permisos de lector en el repositorio del archivo.		
Actor	Usuario.		
Descripción	El usuario puede ver el código de programación realizado en un archivo de un repositorio.		
Secuencia normal	**Paso**		
	1	El usuario ingresa a un archivo de código de programación dentro de un repositorio.	
	2	La plataforma lo redirecciona a un editor de texto estático, mostrando los detalles del documento visto. Además, está a la espera de la edición de alguno de los miembros editores que hayan accedido al documento.	
	3	Si otro miembro editor del repositorio realiza un cambio en el código de programación.	
		3.1	La plataforma actualiza el estado del documento del usuario.
Postcondición	Editor de texto estático activo.		
Importancia	Muy importante.		
Prioridad	Alta.		

Tabla 29. Descripción de caso de uso. Paquete de casos de uso - Programación colaborativa: Ejecutar consola virtual.

Nombre	Ejecutar consola virtual	
Dependencias	RF12.	
Precondición	Editor de texto colaborativo o estático activo. El lenguaje de programación debe ser soportado por la plataforma.	
Actor	Usuario.	
Descripción	El usuario ejecuta el repositorio en una consola virtual con el lenguaje de programación pre-instalado, para que finalmente este pueda probar su código ejecutándose.	
Secuencia normal	**Paso**	**Acción**
	1	El usuario desea probar el código de programación en el estado actual de la edición del código.
	2	La plataforma muestra una opción de “Ejecutar consola virtual”.
	3	El usuario selecciona la opción de ejecutar la consola virtual.
	4	El sistema carga el repositorio dentro de un contenedor virtual con el lenguaje de programación respectivo y al usuario le muestra y permite la interacción con éste mediante comandos de consola.
Postcondición	Se ejecuta la consola virtual dentro de la plataforma.	
Importancia	Muy importante.	
Prioridad	Alta.	

5.4. Modelamiento estático del sistema

5.4.1. Diseño de diagramas de estado

A continuación se muestran todos los diagramas de estado de los respectivos casos de uso descritos en el punto anterior.

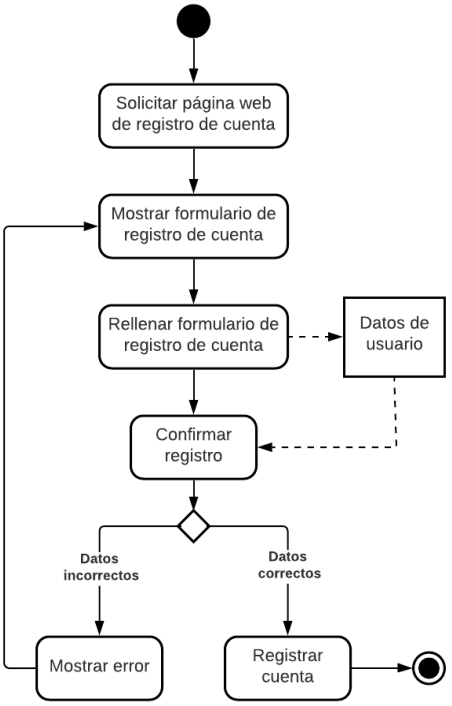

Figura 14. Diagrama de estado. Paquete de casos de uso - Gestión de usuario: Crear cuenta de usuario.

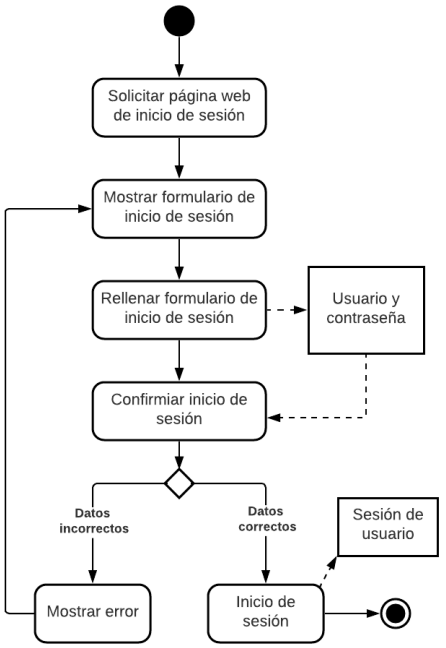

Figura 15. Diagrama de estado. Paquete de casos de uso - Gestión de usuario: Iniciar sesión de usuario.

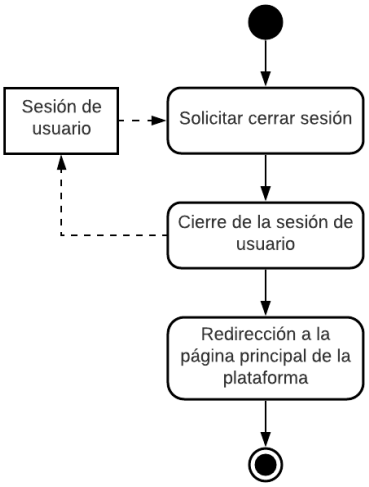

Figura 16. Diagrama de estado. Paquete de casos de uso - Gestión de usuario: Cerrar sesión de usuario.

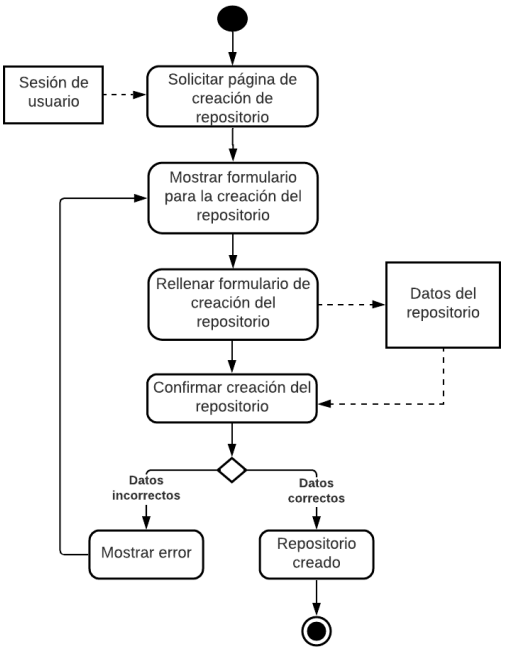

Figura 17. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Crear repositorio.

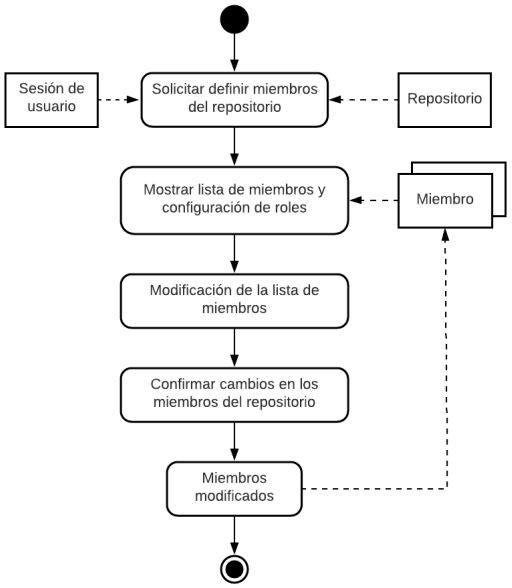

Figura 18. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Definir miembros del repositorio.

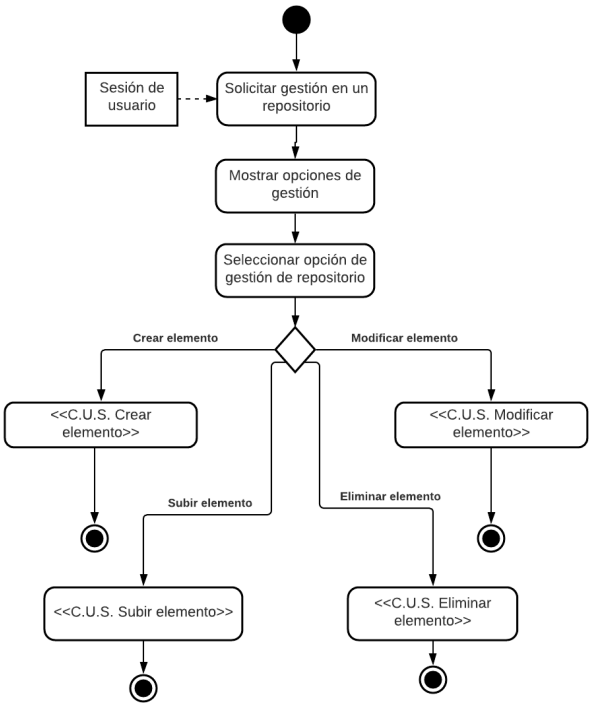

Figura 19. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Gestionar repositorio.

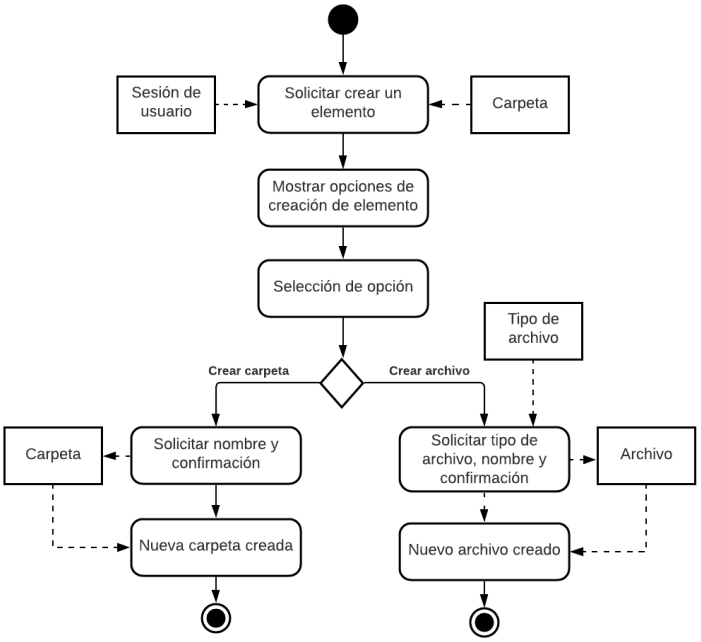

Figura 20. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Crear elemento.

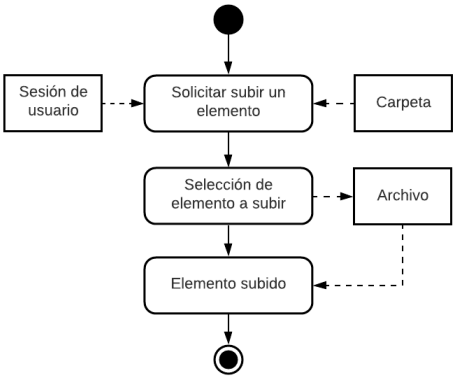

Figura 21. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Subir elemento.

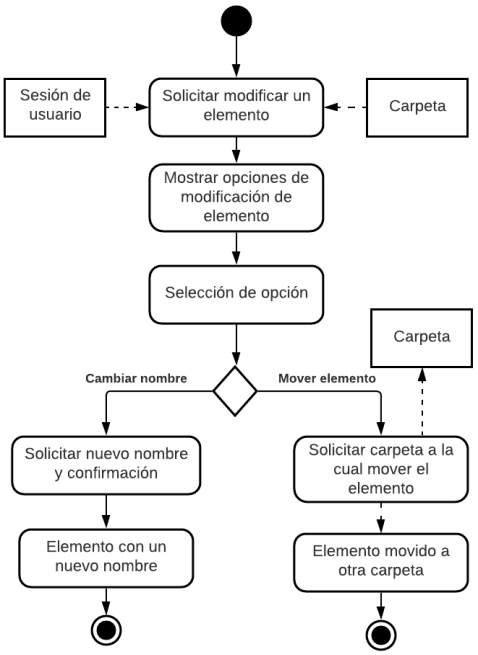

Figura 22. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Modificar elemento.

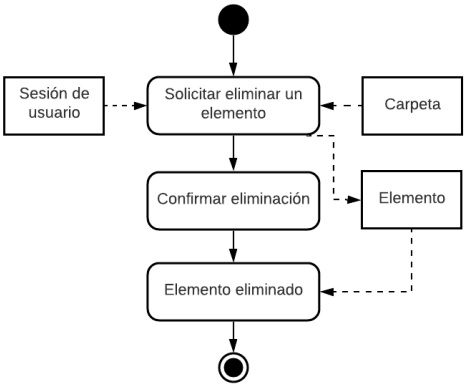

Figura 23. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Eliminar elemento.

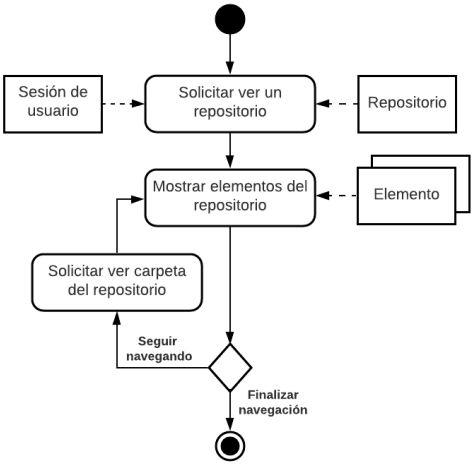

Figura 24. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Ver repositorio.

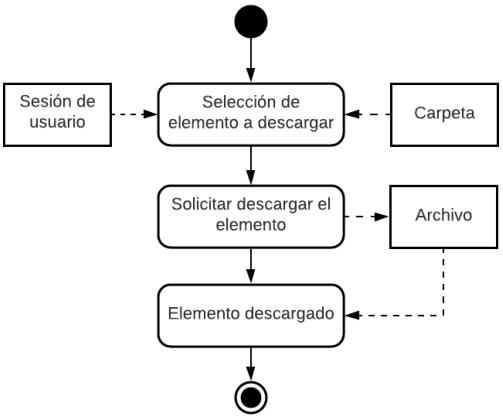

Figura 25. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Descargar elemento.

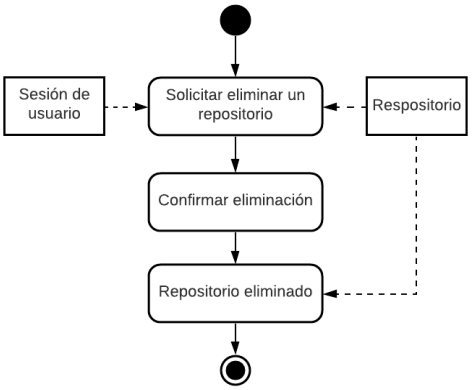

Figura 26. Diagrama de estado. Paquete de casos de uso - Gestión de repositorio: Eliminar repositorio.

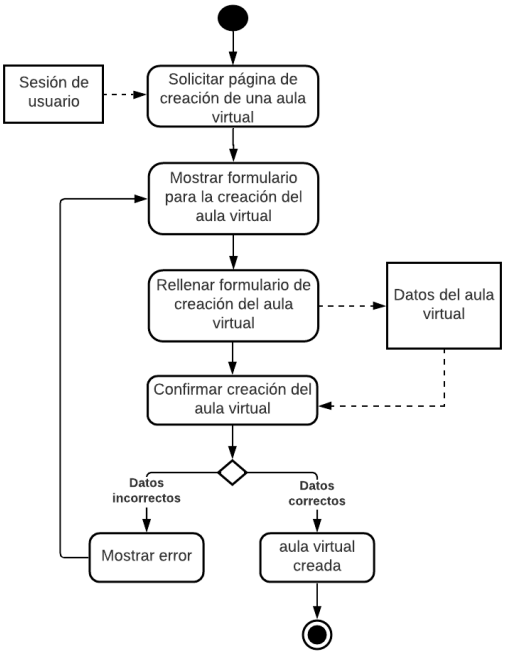

Figura 27. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Crear aula virtual.

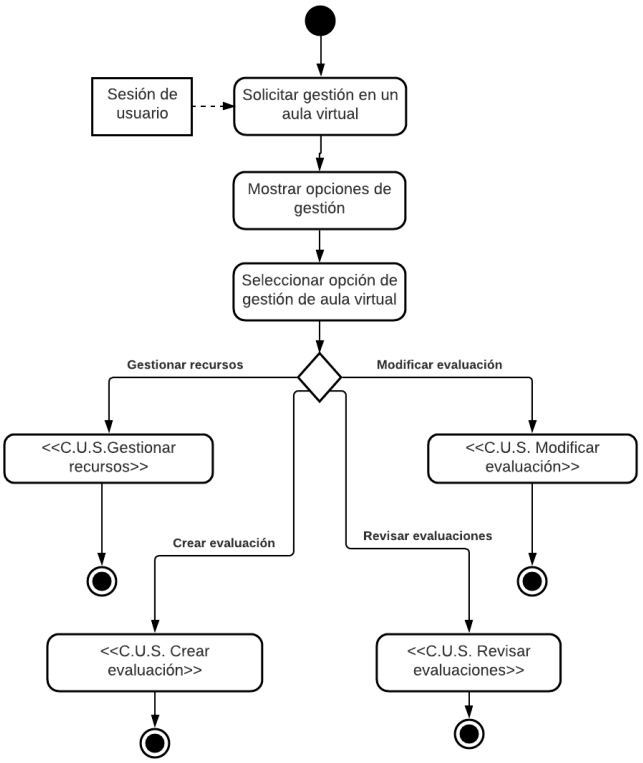

Figura 28. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Gestionar aula virtual.

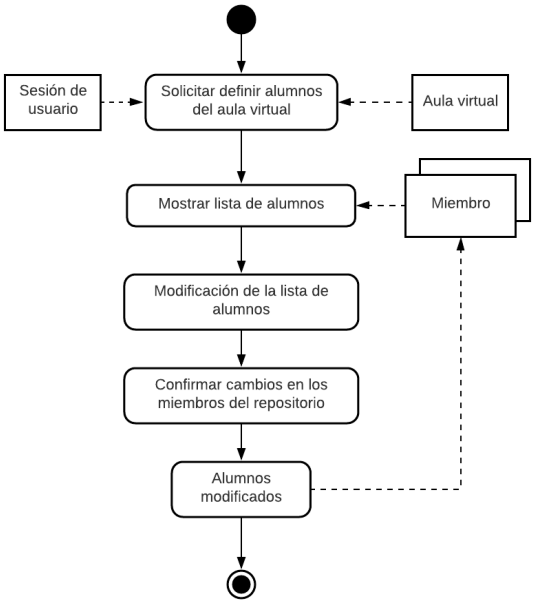

Figura 29. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Gestionar alumnos.

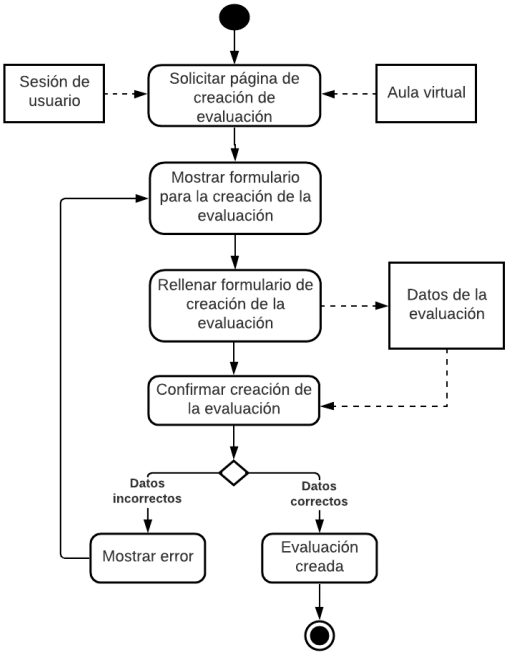

Figura 30. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Crear evaluación.

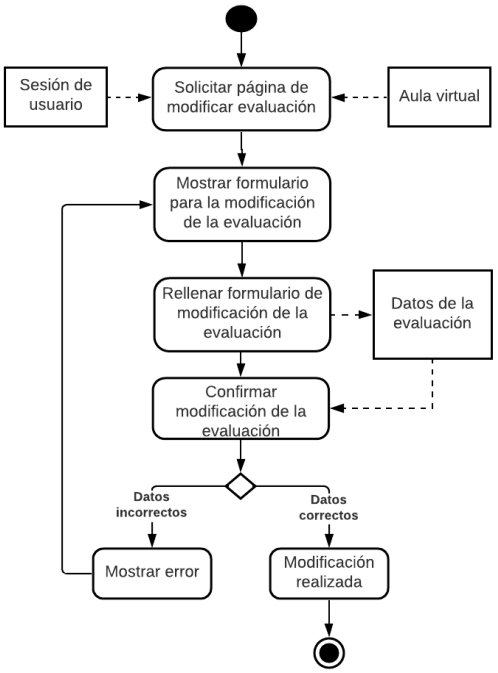

Figura 31. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Modificar evaluación.

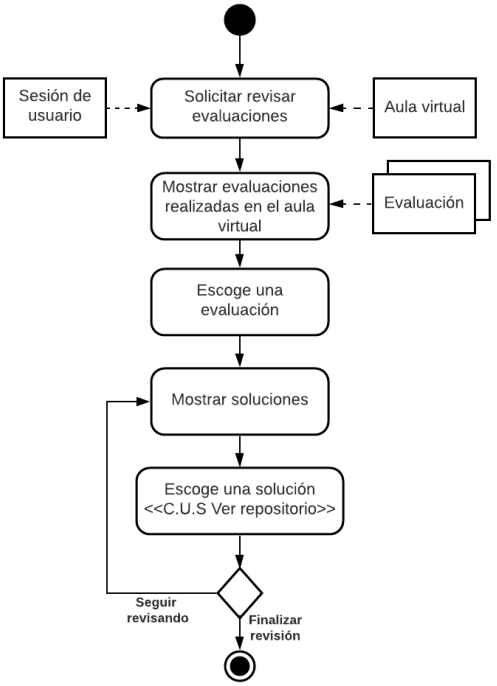

Figura 32. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Revisar evaluaciones.

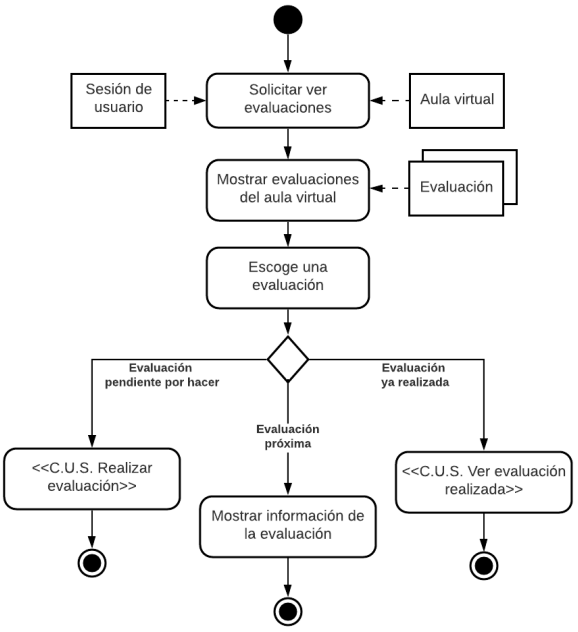

Figura 33. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Ver evaluaciones.

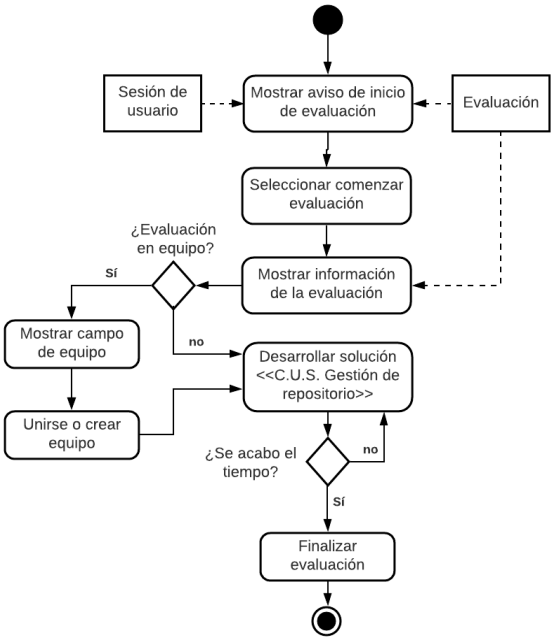

Figura 34. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Realizar evaluación.

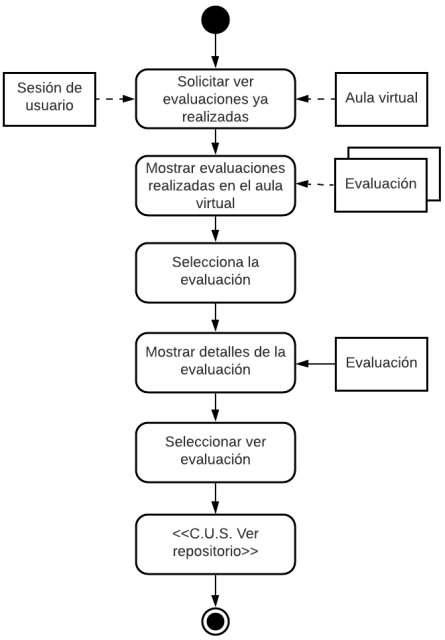

Figura 35. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Ver evaluación realizada.

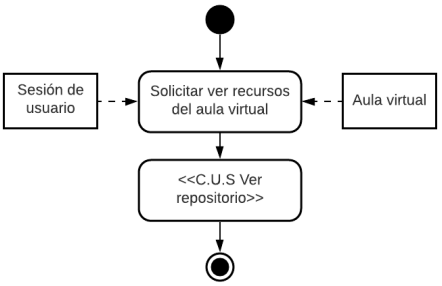

Figura 36. Diagrama de estado. Paquete de casos de uso - Gestión de aula virtual: Ver recursos.

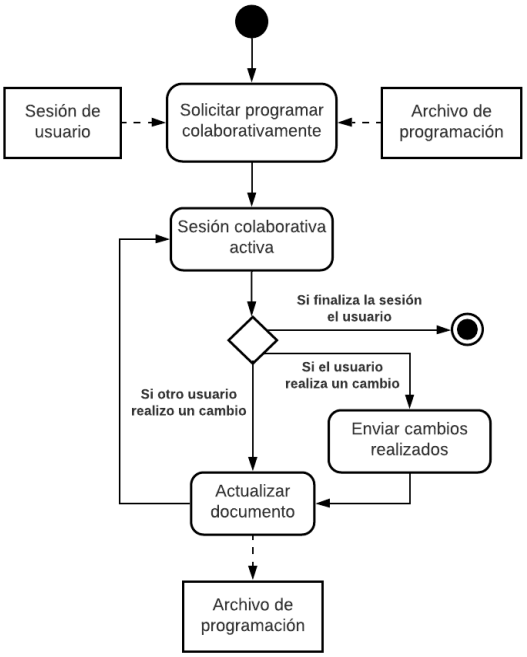

Figura 37. Diagrama de estado. Paquete de casos de uso - Programación colaborativa: Programar colaborativamente.

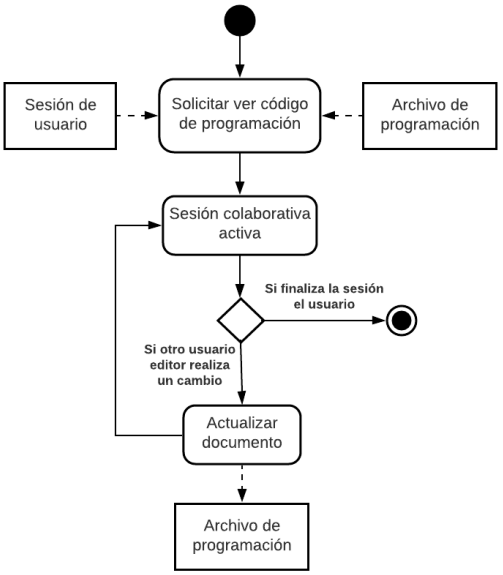

Figura 38. Diagrama de estado. Paquete de casos de uso - Programación colaborativa: Ver código de programación.

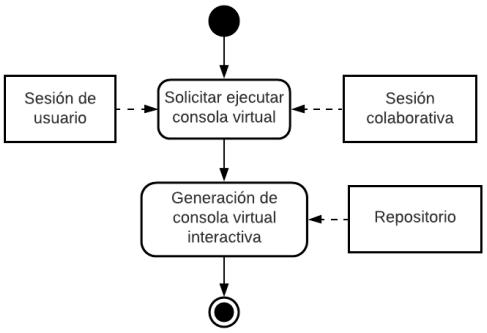

Figura 39. Diagrama de estado. Paquete de casos de uso - Programación colaborativa: Ejecutar consola virtual.

5.4.2 Diseño de modelo conceptual estático

Mediante el análisis y diseño realizado mediante los casos de uso, se realiza el siguiente modelo conceptual estático:

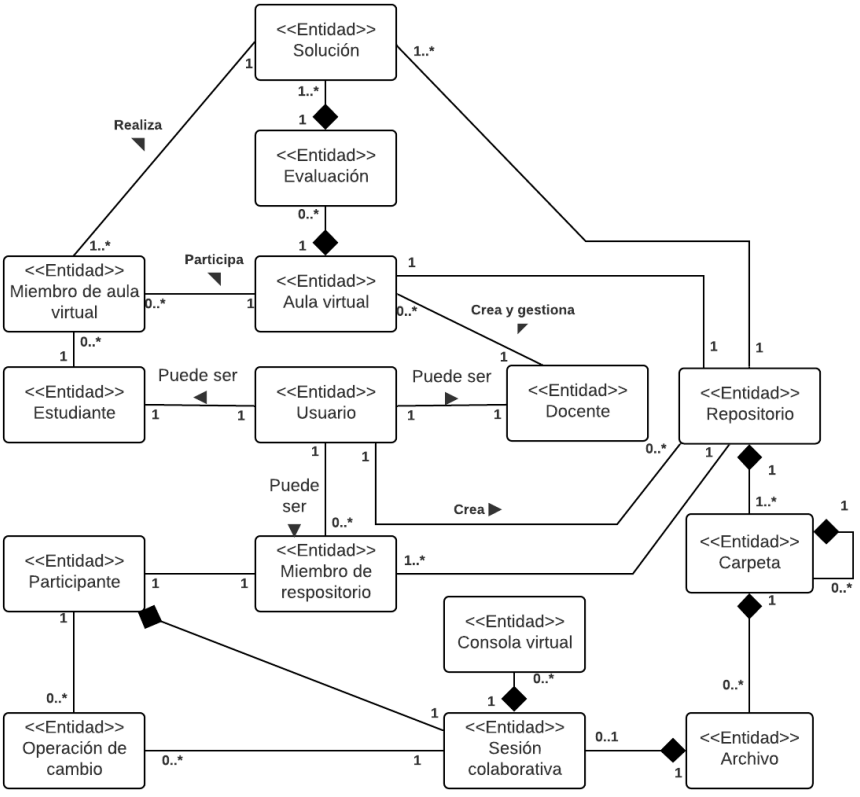

Figura 40. Modelo conceptual estático de SEAP-UTA.

5.4.3. Diseño de clases

A continuación se definen las clases a través de las entidades del modelo conceptual.

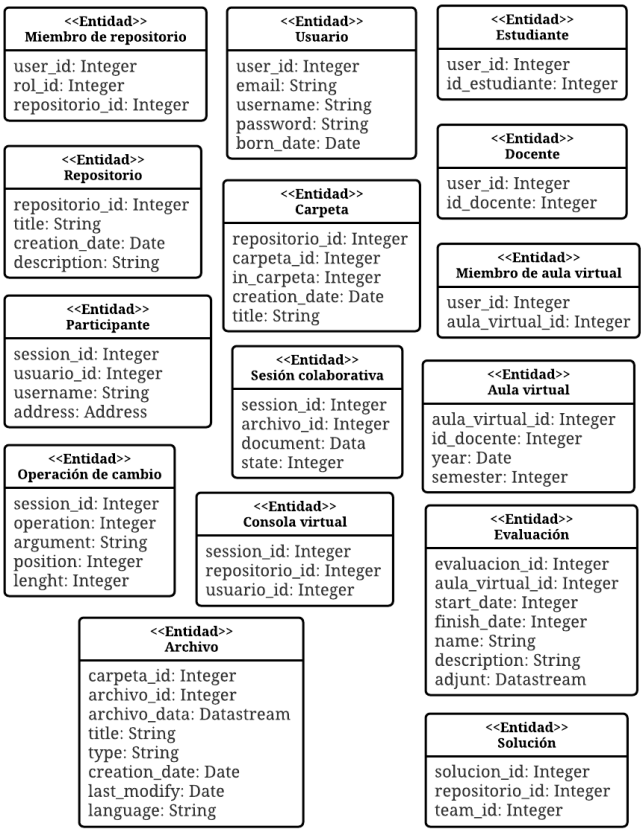

Figura 41. Modelado de las clases de tipo entidad.

5.5. Estructuración de objetos y clases

A continuación se define la estructuración de las clases como servicios o subsistemas para el software, en donde se agrupan las clases definidas anteriormente.

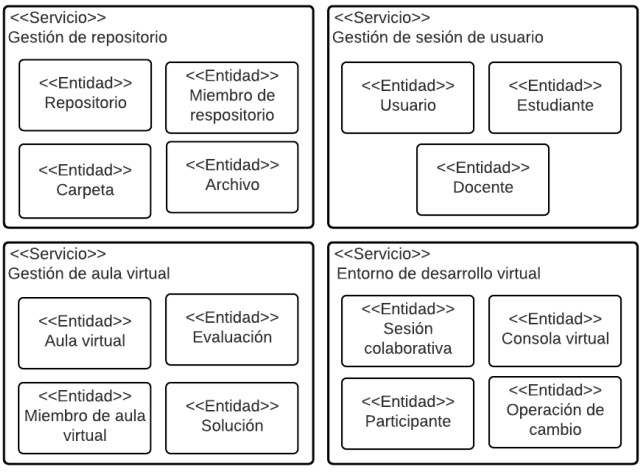

Figura 42. Estructuración de clases como servicios del sistema.

5.6. Modelamiento dinámico

Para la realización del modelamiento dinámico se definen el sistema de software como:

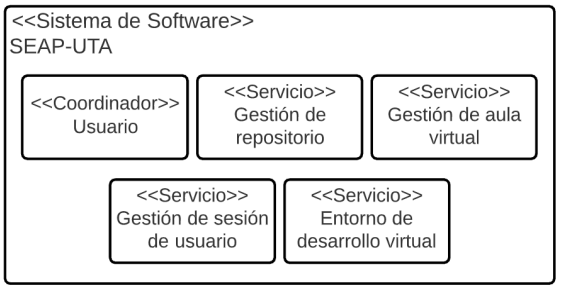

Figura 43. Sistema de software SAEP-UTA, incluyendo cada uno de los subsistemas de este.

Explicación de cada subsistema:

Subsistemas de tipo <<Coordinador>>:

- Usuario: Encargado de coordinar el usuario con los servicios requeridos y la coordinación necesaria entre los diferentes servicios.

Subsistemas de tipo <<Servicio>>:

- Entorno de desarrollo virtual: Servicio encargado de las sesiones de programación colaborativa de los usuarios dentro de un archivo de programación contenido dentro de un repositorio, además se encarga de la gestión de consolas virtuales.
- Gestión de repositorio: Servicio que permite todos los aspectos de gestión del repositorio, incluyendo roles, archivos y carpetas.
- Gestión de sesión de usuario: Encargado del manejo de cuentas de usuario, tanto la creación, inicio de sesión y manejo de sesiones de cada tipo de usuario.
- Gestión de aula virtual: Servicio que permite todos los aspectos de gestión del aula virtual, incluyendo alumnos, recursos y evaluaciones.

5.6.1. Diagrama colaborativo consolidado del sistema

A continuación se muestra un avance de lo que podría ser el diagrama de comunicación consolidado del sistema SEAP-UTA.

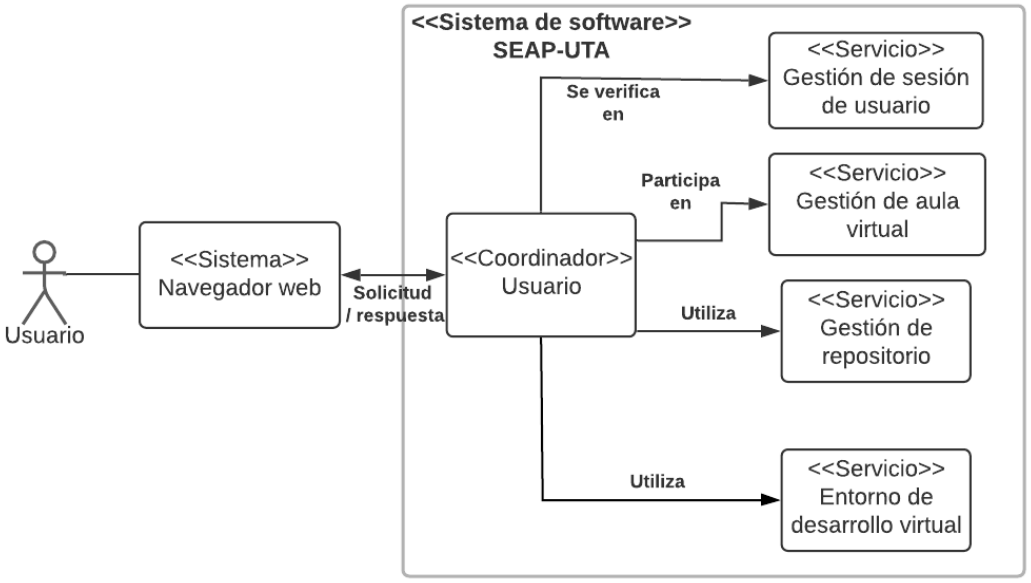

Figura 44. Diagrama colaborativo del sistema.

6. IMPLEMENTACIÓN

En este punto se verán los detalles de la implementación del proyecto SEAP-UTA, considerando las herramientas, supuestos y resultados parciales obtenidos.

6.1. Herramientas de desarrollo

A continuación se verán las herramientas utilizadas para la implementación del proyecto, considerando una definición general de la herramienta.

Visual Studio Code
Editor de código fuente desarrollado por Microsoft. Entre las características más importantes, incluye soporte para la depuración, control integrado de Git, gratuito y de código abierto. Este será el editor y depurador principal para el código fuente del proyecto.

Node JS
Para el desarrollo backend del proyecto, se utiliza el entorno de desarrollo Node JS, un framework basado en javascript. Este entorno será el motor principal del backend del proyecto. En específico, los servicios realizados en node, son:

- **Consola virtual**: Permite la visualización de la consola virtual en el navegador, y mediante el servicio, se realiza una conexión SSH a la máquina virtual correspondiente del repositorio, para ejecutar los comandos correspondientes. Todo esto posible mediante un proyecto de código abierto, llamado “webssh2” [5], para más detalles ver las referencias.
- **Gestión de máquinas virtuales**: Servicio que se encarga de gestionar y coordinar las máquinas virtuales de los repositorios, mediante el frontend (React) y el motor de modelos en tiempo real (Convergence). Este servicio es una elaboración propia, debido a que no existe un sistema de este tipo en código abierto.
- **Gestión de cuentas de usuario y repositorios**: Encargado de la gestión de usuarios, mediante la creación y validación de sesiones. Además, se encarga de gestionar los repositorios de los usuarios, en conjunto trabaja con el motor Convergence para la definición de permisos. Creación propia, debido a la particularidad del sistema.
- **Gestión de aula virtual**: Permite la gestión de aulas virtuales, servicio principal para los usuarios profesores y alumnos.

Las librerías principales utilizadas en Node JS, son las siguientes:

- **Express**: Proporciona un conjunto sólido de características para la implementación de aplicaciones web.
- **JWT**: Conocido como JSON Web Token, es un estándar para la creación de tokens de acceso.
- Convergence y extensiones:
- **SSH2**: Módulo que permite la implementación de cliente y servidor SSH.
- **Mysql2**: Cliente Mysql que permite la conexión con un servidor de base de datos MySQL.
- **WebSocket**: Implementación de protocolo WebSocket, tanto cliente como servidor, utilizado para la conexión con Convergence.

- **Convergence**: Módulo que integra un cliente de Convergence, el cual permite utilizar la API para comunicarse con el servidor Convergence.

Postman

Aplicación que permite realizar pruebas API. Es un cliente HTTP que da la posibilidad de probar solicitudes "HTTP Request", a través de una interfaz gráfica. Está es utilizada para probar el código en desarrollo del backend.

MySQL Community Server

Sistema administrador de bases de datos (Database Management System, DBMS) para bases de datos relacionales, esta aplicación permite montar un servidor de base de datos MySQL con una licencia gratuita. Es la base de datos utilizada para la implementación del proyecto, en específico, la gestión de usuarios, aulas virtuales y parte de los repositorios (Debido que gran parte de la información se guarda en la base de datos de Convergence).

MySQL Workbench

Herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, gestión y mantenimiento para el sistema de base de datos MySQL. En el contexto de la implementación, se utiliza para la creación, gestión y realización de pruebas de la base de datos.

Convergence

Motor para el desarrollo rápido de aplicaciones colaborativas de tiempo real. Para ello utiliza un almacén de datos colaborativo en tiempo real, a diferencia de otros frameworks que se detienen al intentar proporcionar datos en tiempo real. Proporciona las funcionalidades que un desarrollador necesita para crear una aplicación colaborativa. Algunas características principales son:

- **Datos en tiempo real**: Permite a los usuarios editar el mismo dato al mismo tiempo, manteniendo todas las vistas de la información sincronizadas.
- **Usuarios e identificación**: Permite a los desarrolladores gestionar a los usuarios e identificar mediante el sistema en donde están colaborando y validar permisos correspondientes.
- **Presencia**: Permite a los usuarios conocer quién está trabajando en el modelo de tiempo real y quién está disponible, para poder trabajar de forma más efectiva con los usuarios colaboradores.
- **Conciencia de colaboración (Awareness)**: La mejor manera de resolver conflictos de edición colaborativa, es evitarlos en primer lugar. Convergence proporciona sólidas capacidades de conocimiento de la colaboración que ayudan a los usuarios trabajar juntos sin conflictos.
- **Chat y mensajería**: Permite a los usuarios comunicarse con un chat embebido directamente en la aplicación.
- **Código abierto y gratuito**.

Convergence provee todas estas características y más mediante una API, reduciendo el tiempo y complejidad de implementar para diferentes frameworks.

En el proyecto, este será el núcleo principal para el manejo de sesiones colaborativas en el entorno virtual de los usuarios del sistema SEAP-UTA. En específico, se guardarán los datos del repositorio en tiempo real, tanto elementos (carpetas y archivos) y su contenido (archivos o código de programación).

React

Utilizado para el desarrollo del frontend del proyecto. React es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Entre sus características principales contiene:

- **Componentes**: Los componentes permiten separar la interfaz de usuario en piezas independientes, reutilizables y pensar en cada pieza de forma aislada.
- **Propiedades:** Las propiedades (también conocidas como 'props') pueden definirse como los atributos de configuración para dicho componente. Éstas son recibidas desde un nivel superior, normalmente al realizar la instancia del componente y por definición son inmutables.
- **DOM (Document Object Model) virtual**: React mantiene un virtual DOM propio, en lugar de confiar solamente en el DOM del navegador. Este comportamiento es clave en React y causante de su alto rendimiento, ya que cuando cambia un dato en la aplicación no necesita actualizar la vista entera.
- **Elementos y JSX**: React no retorna HTML. El código embebido dentro de Javascript, similar a HTML pero realmente es JSX. Similares a las funciones de Javascript, pero expresadas mediante una sintaxis propia de React llamada JSX. Lo que produce son elementos en memoria y no elementos del DOM tradicional, con lo cual las funciones no ocupan tiempo en producir pesados objetos del navegador sino simplemente elementos de un DOM virtual.
- **Isomorfismo**: Permite ejecutar el código tanto en el cliente como en el servidor.

En específico, en el proyecto es utilizado para la generación de vistas web del usuario. Cabe destacar que para el desarrollo del editor de código, este está basado en un ejemplo de Convergence llamado “code-editor-demo” [11], el cual en principio da una base para el desarrollo del editor de código del entorno virtual del proyecto.

Nginx

Servidor web/proxy inverso ligero de alto rendimiento y un proxy para protocolos de correo electrónico. Es un software libre y de código abierto, contemplando una versión de pago llamada Nginx Plus. Utilizando la versión gratuita, en el proyecto se implementa como coordinador de los diferentes servicios del proyecto, permitiendo tener un solo dominio.

Docker

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos.

En el proyecto SEAP-UTA, Docker es una pieza fundamental para el despliegue de la aplicación tanto para los servicios y el coordinador. Pero no solo para despliegue, esta herramienta es fundamental para el desarrollo de entornos virtuales, en específico, para la consola virtual. Esto se aborda mediante el uso de contenedores virtuales con una imagen “linuxserver/openssh-server” [14], modificada para el propósito del repositorio en particular, en conjunto al servicio encargado de gestionar estos contenedores y derivar las sesiones a las máquinas respectivas.

El contenedor mencionado anteriormente posee un sistema operativo Linux con la distribución Alpine, el cual puede ser configurado para ejecutar múltiples lenguajes de programación y realizar diversas funciones para el desarrollo de software.

GIT

Git es un sistema de control de versiones distribuido, rápido y escalable, el cual funciona con un conjunto de comandos que proporciona operaciones de alto y completo acceso a los componentes internos de un proyecto.

Su propósito es llevar un registro de los cambios que se realicen en el proyecto, incluyendo la coordinación de trabajo que está ofrece para que varios personas sobre un mismo proyecto puedan trabajar sobre un repositorio.

Github

GitHub es una plataforma para alojar el código de las aplicaciones y herramientas de cualquier desarrollador, y como usuario, puedes conocer a los desarrolladores y colaborar en sus proyectos. Esta trabaja en conjunto a GIT.

Esta herramienta será utilizada para general el control de versiones del proyecto, de forma que el cliente pueda monitorizar el estado actual del proyecto.

6.2. Definición de la implementación

Los requerimientos fueron acotados debido al tiempo requerido para realizar un sistema de esta complejidad, es por esto que se realizaron los siguientes servicios, los cuales fueron mencionados en las herramientas:

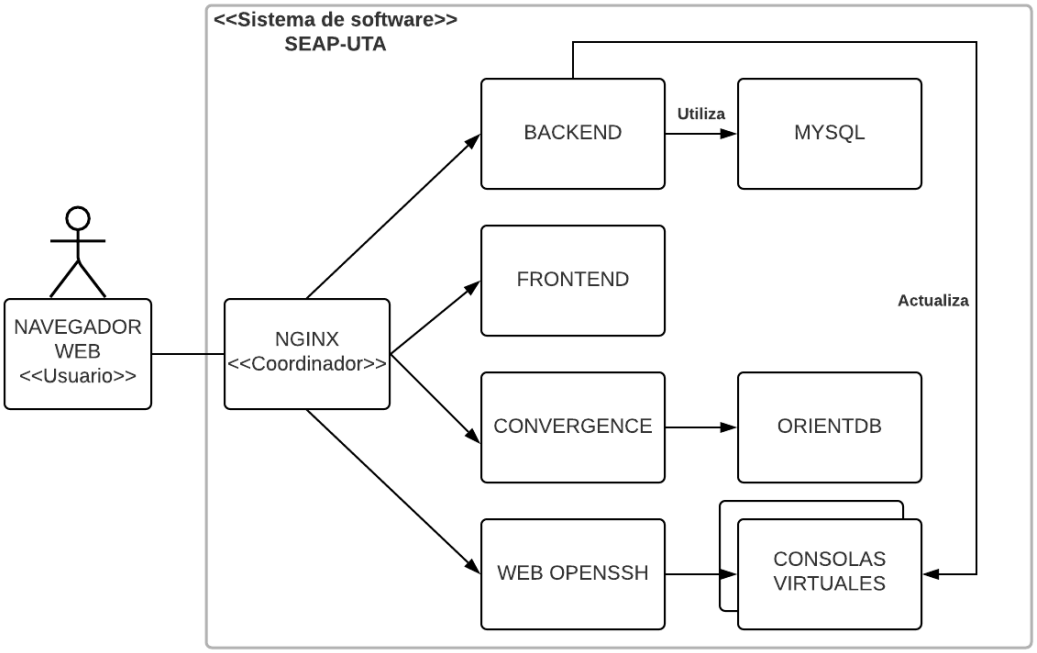

Figura 45. Diagrama colaborativo del sistema implementado.

7. RESULTADOS DEL PROYECTO

En el siguiente punto, se muestran todas las funcionalidades implementadas al sistema SEAP-UTA. Parte del funcionamiento del usuario común y el administrador.

7.1. Usuario común en SEAP-UTA

7.1.1. Inicio de sesión

Al entrar en la página web de SEAP-UTA, se le solicitan sus credenciales de usuario, para el inicio de sesión.

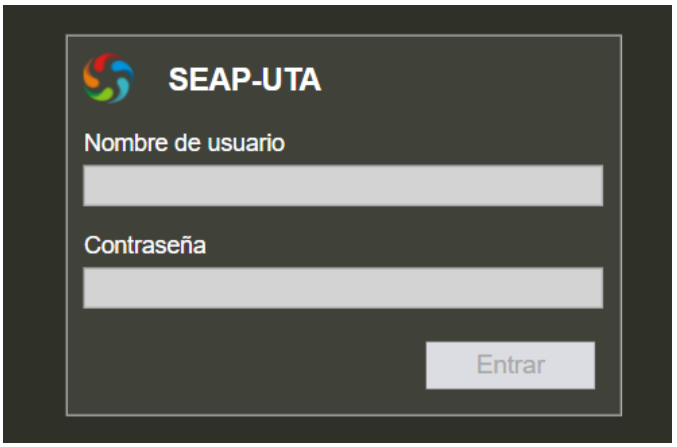

Figura 46. SEAP-UTA. Inicio de sesión del usuario.

Al iniciar sesión correctamente, si el navegador web utilizado puede guardar cookies, la sesión del usuario queda iniciada hasta que se elimine la cookie o el usuario cierre sesión.

7.1.2. Menú principal

Cuando el usuario haya iniciado sesión, se le muestra una vista del menú principal de SEAP-UTA, el cual cuenta con una vista de los repositorios disponibles para el usuario y el rol que desempeña, los cuales pueden ser:

- Mis repositorios: Repositorios creados por el usuario en cuestión.
- Repositorios en los que soy miembro: Donde el usuario participa con un rol definido por otro usuario.
- Repositorios públicos: Vista de todos los repositorios que se hayan definido públicos en la plataforma.

Figura 47. SEAP-UTA. Menú principal. Listas de repositorios.

Además se cuenta con las siguientes opciones:

- **Abrir**: Al seleccionar un repositorio de alguna de las listas, se puede seleccionar este botón que dará acceso al entorno de desarrollo virtual del repositorio.

- **Crear nuevo repositorio**: Permite al usuario crear un repositorio, se requiere de un nombre, y si el repositorio es público o privado. Al generar el proyecto de forma pública, todos los usuarios que no sean miembros del repositorio, serán considerados con el rol de lector. Si se selecciona crear un repositorio, se entrará automáticamente su entorno de desarrollo virtual respectivo.

- **Borrar**: Al seleccionar un repositorio en la lista de repositorios que haya sido creado por el usuario en específico, podrá eliminar el repositorio respectivo de la plataforma.

- **Actualizar**: Permite actualizar las listas de repositorios.

- **Botón de ON/OFF**: Permite cerrar la sesión del usuario.

Figura 48. SEAP-UTA. Menú principal. Botones de opciones disponibles.

Figura 49. SEAP-UTA. Menú principal. Creación de un nuevo repositorio.

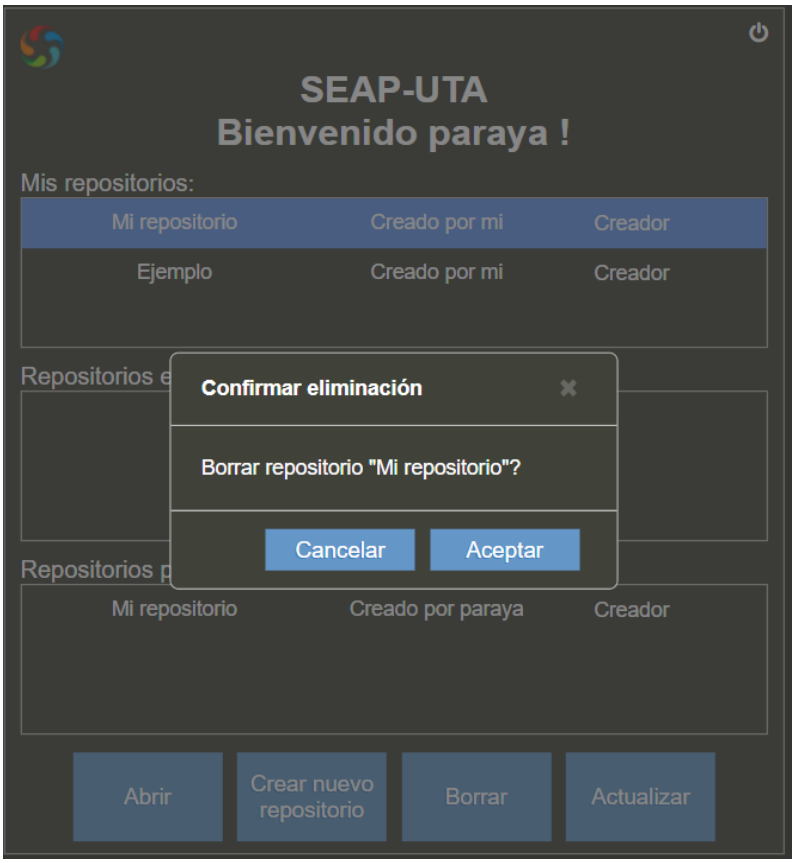

Figura 50. SEAP-UTA. Menú principal. Eliminación de un repositorio.

7.1.3. Entorno de desarrollo virtual

El entorno de desarrollo virtual cuenta con múltiples funcionalidades, las cuales se dividen en 6 partes:

- **Encabezado**: Muestra una descripción del sitio, además, contempla las opciones para salir del ambiente de desarrollo virtual y cerrar sesión.
- **Manejador del repositorio**: Contempla la navegación, administración y gestión de miembros del repositorio.
- **Tablero de edición**: Al abrir un archivo desde el manejador del repositorio, el tablero muestra el contenido del mismo, permitiendo la edición o lectura correspondiente.
- **Participantes**: Muestra una lista de los participantes activos en la sesión del ambiente de desarrollo virtual.
- **Chat de grupo**: Permite la comunicación vía texto entre los participantes de la sesión.
- **Consola del ambiente virtual**: Permite la interacción de un ambiente de ejecución con el contenido del repositorio en uso.

Figura 51. SEAP-UTA. Entorno de desarrollo virtual.

En los siguientes puntos se entra en detalle del funcionamiento de cada una de las funcionalidades del entorno de desarrollo virtual.

7.1.3.1. Encabezado

Muestra una descripción del sitio, además, contempla el nombre de usuario conectado y las opciones para salir del ambiente de desarrollo virtual y cerrar sesión.

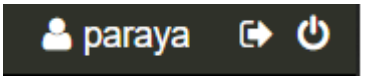

Figura 52. SEAP-UTA. Encabezado del entorno de desarrollo virtual. Nombre de usuario, botón de salir del entorno y botón de cerrar sesión respectivamente.

7.1.3.2. Manejador del repositorio

Este menú presenta las funcionalidades de administración del repositorio, las cuales cambian dependiendo del rol del repositorio, los cuales son:

- **Creador**: El usuario creador es el que ha creado el repositorio, este tiene permisos absolutos sobre este, puede administrar los elementos del repositorio, agregar miembros y modificarlos, y puede definir administradores. Además, puede eliminar el repositorio de la plataforma.

- **Administrador**: Un administrador es definido por el creador del repositorio, tiene permisos para administrar los elementos del repositorio, agregar y definir miembros de tipo editor y lector.

- **Editor**: El rol de editor tiene permitido administrar los elementos del repositorio, además de ver los miembros de este.

- **Lector**: El rol de lector, simplemente puede ver los elementos del repositorio, además de ver los miembros de este.

7.1.3.4. Ver o modificar miembros del repositorio

En general, todos los tipos de roles cuentan con la siguiente opción de ver o modificar miembros del repositorio, y dependiendo del rol, varía la funcionalidad de este.

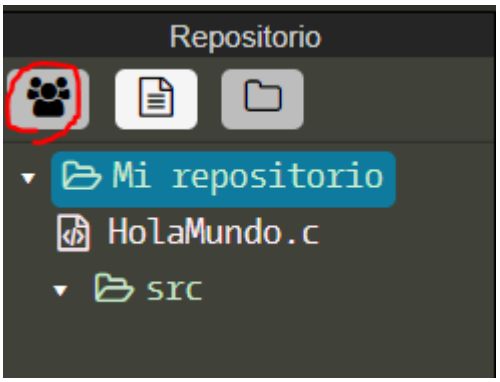

Figura 53. SEAP-UTA. Manejador del repositorio. Opción para ver o modificar miembros.

Como miembro Creador: La siguiente figura muestra la funcionalidad de este rol, la cual contempla una vista con todos los usuarios miembros del repositorio con las opciones de modificación respectivas, y la opción de agregar un nuevo miembro a través de un nombre.

Interfaz de usuario para el rol creador:

Figura 54. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol creador.

Al agregar un miembro:

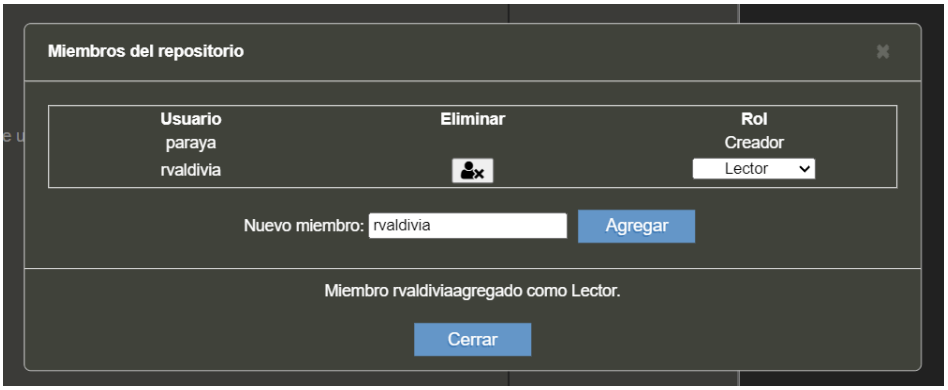

Figura 55. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol creador. Agregación de un nuevo miembro.

Como opciones de cambio de rol, el creador puede definir un miembro del repositorio como, administrador, editor o lector:

Figura 56. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol creador. Modificación de rol de un miembro del repositorio.

Además, existe la posibilidad de eliminar miembros del repositorio, a través del siguiente botón:

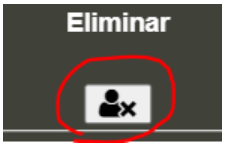

Figura 57. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol creador o administrador. Botón para eliminar un miembro del repositorio.

Resultado de una eliminación de un miembro del repositorio:

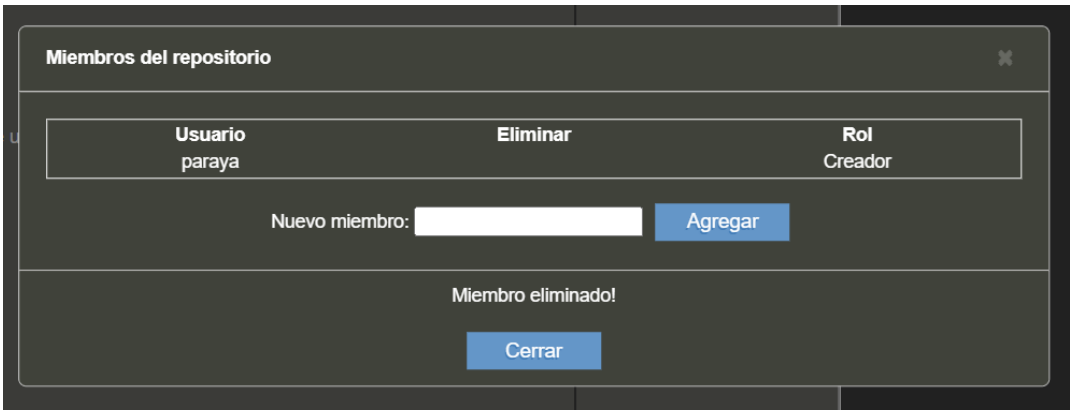

Figura 58. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol creador o administrador. Resultado de la eliminación de un miembro.

Como miembro administrador: Al igual que el creador, el administrador tiene una general permitiéndole modificar aspectos de los miembros del repositorio, sin embargo, este rol, no tiene permitido eliminar al creador del repositorio, como también definir otros miembros del repositorio como administradores.

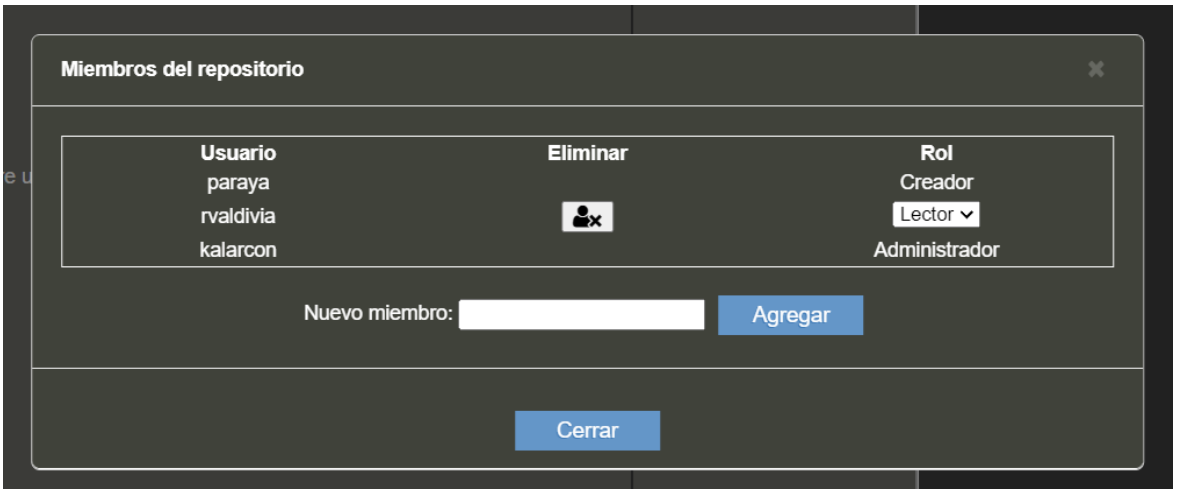

Figura 59. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol administrador.

Figura 60. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol administrador. Modificación de rol de un miembro del repositorio.

Como miembro editor y lector: Ambos tienen la misma funcionalidad al ver los miembros del repositorio, que simplemente es una vista actual de los miembros.

Figura 61. SEAP-UTA. Manejador del repositorio. Miembros del repositorio, vista desde el rol editor o lector.

7.1.3.5. Gestión de archivos y carpetas del repositorio

La gestión de archivos y carpetas del repositorio consiste en poder crear, modificar y eliminar archivos o carpetas, a través de la vista del árbol de elementos (carpetas y archivos) y las funcionalidades correspondientes. Para poder realizar estas acciones se requiere tener un rol de tipo creador, administrador o editor y seleccionar el elemento en donde se realizará la acción. En caso del lector, simplemente tendrá acceso de solo lectura a todos los elementos del repositorio.

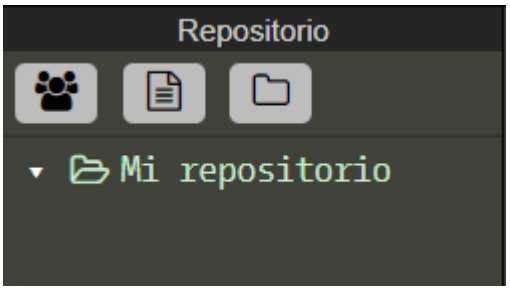

Figura 62. SEAP-UTA. Manejador del repositorio. Gestión del repositorio.

Para crear un archivo en la carpeta raíz “Mi repositorio”:

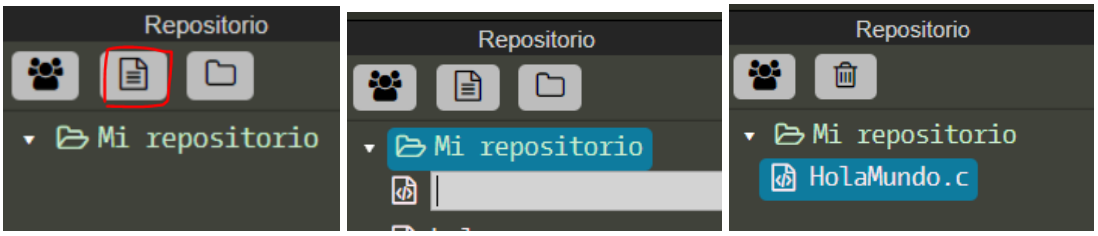

Figura 63. SEAP-UTA. Manejador del repositorio. Gestión del repositorio. Creación de un archivo.

Para crear una carpeta en la carpeta raíz “Mi repositorio”:

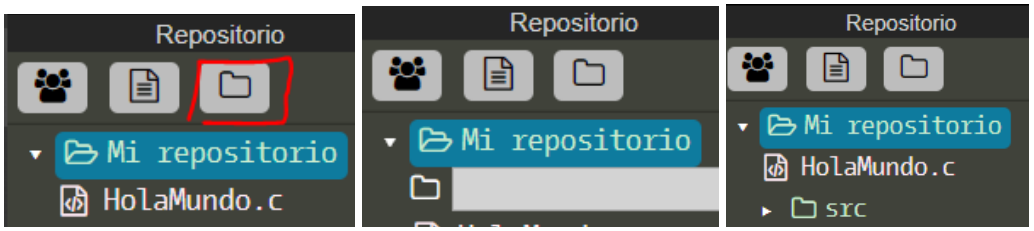

Figura x. SEAP-UTA. Manejador del repositorio. Gestión del repositorio. Creación de una carpeta.

Eliminar un archivo o carpeta:

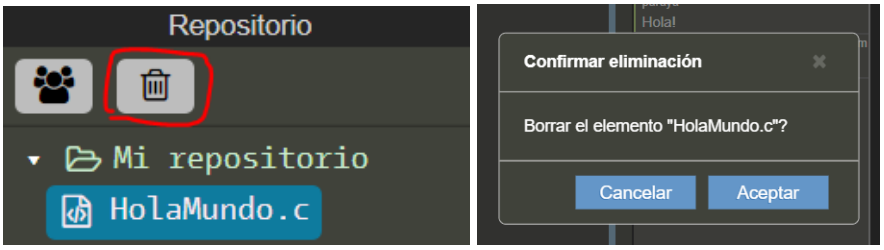

Figura 64. SEAP-UTA. Manejador del repositorio. Gestión del repositorio. Eliminación de un archivo y confirmación.

Existen más opciones al seleccionar un elemento y utilizar el click derecho del mouse:

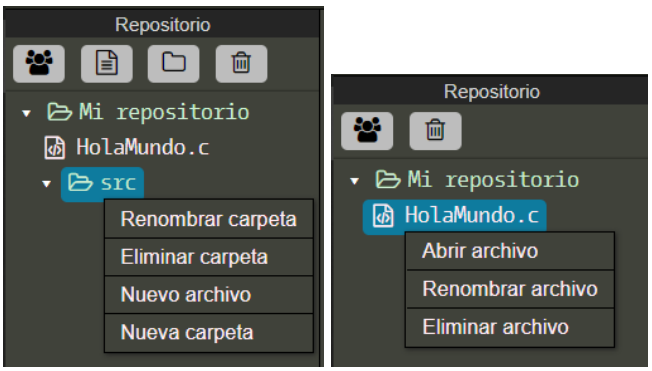

Figura 65. SEAP-UTA. Manejador del repositorio. Gestión del repositorio. Opciones adicionales.

La vista de un miembro de lector es la siguiente:

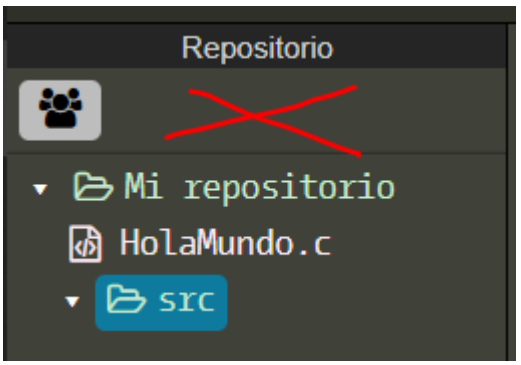

Figura 66. SEAP-UTA. Manejador del repositorio. Gestión de archivos. Opciones adicionales.

7.1.3.6. Tablero de edición

Al abrir un archivo desde el manejador del repositorio, el tablero muestra el contenido del mismo, permitiendo la edición o lectura correspondiente, desplegando cada uno de los archivos abiertos como ventanas.

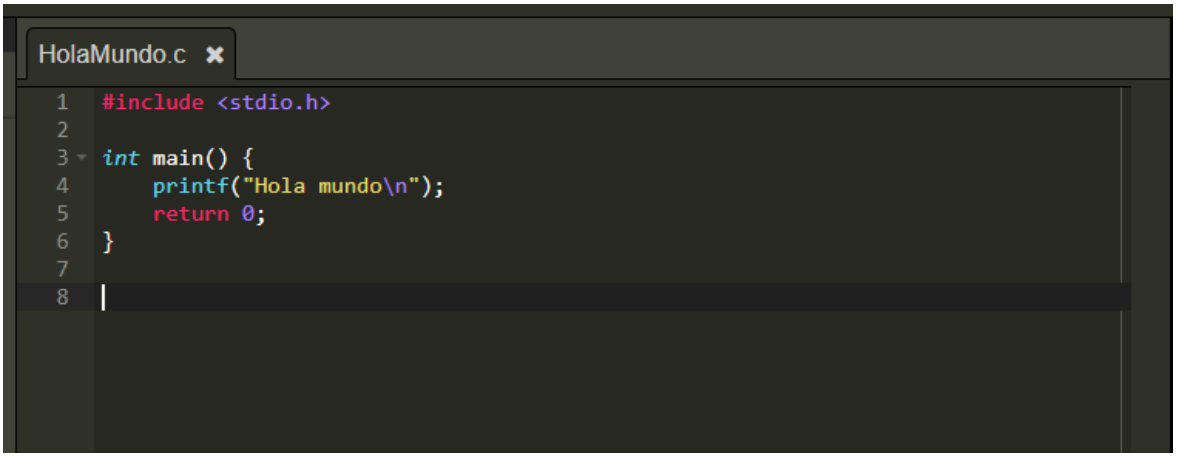

Figura 67. SEAP-UTA. Tablero de edición. Ventana con el archivo "HolaMundo.c".

Para cerrar una ventana de edición, es necesario seleccionar el botón en forma de "x".

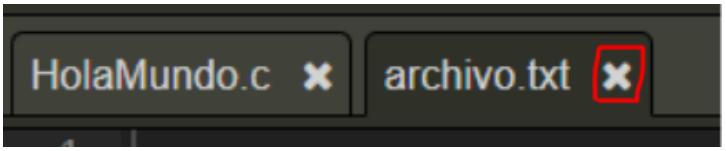

Figura 68. SEAP-UTA. Tablero de edición. Funcionalidad para cerrar una ventana.

Vista de los cursores de los participantes participando en tiempo real:

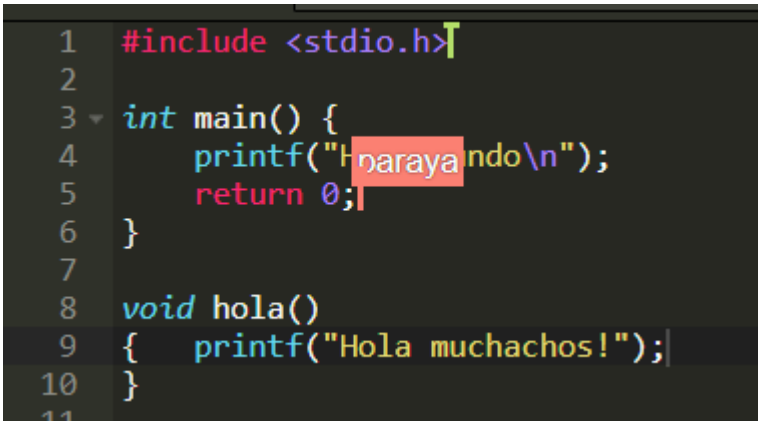

Figura 69. Tablero de edición. Cursores de los participantes colaborando en tiempo real.

7.1.3.6. Participantes

Permite listar los usuarios que están participando dentro del repositorio.

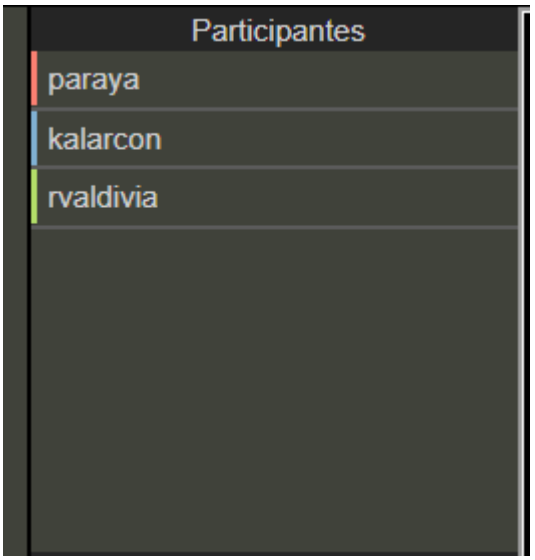

Figura 70. Participantes. Lista de participantes que participan en el repositorio actual.

7.1.3.7. Participantes

Permite la comunicación vía texto entre los participantes de la sesión, para enviar un mensaje, se requiere de escribir el mensaje y luego presionar la tecla “Enter”.

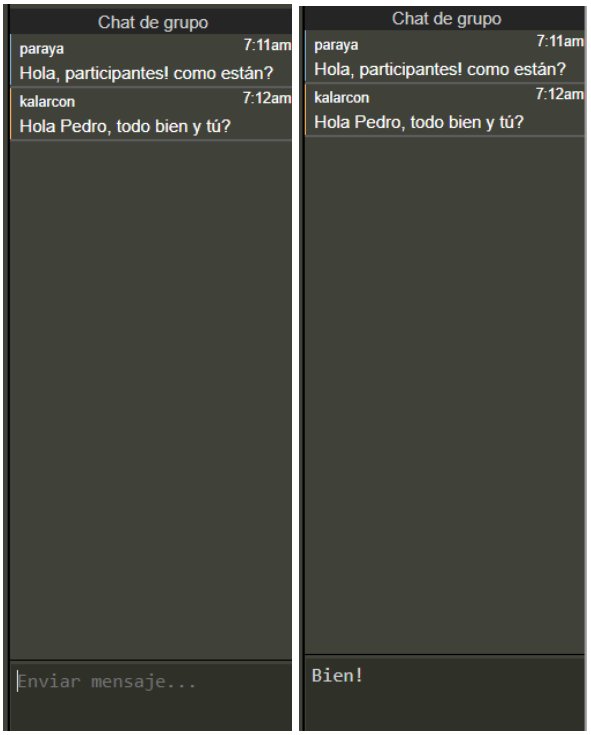

Figura 71. SEAP-UTA. Chat de grupo.

7.1.3.8. Consola del ambiente virtual

Permite la interacción de un ambiente de ejecución con el contenido del repositorio en uso, el contenido de la consola virtual se actualiza por cada cambio en el repositorio. Es importante mencionar que la actualización es unidireccional, desde el repositorio a la consola y no desde la consola al repositorio.

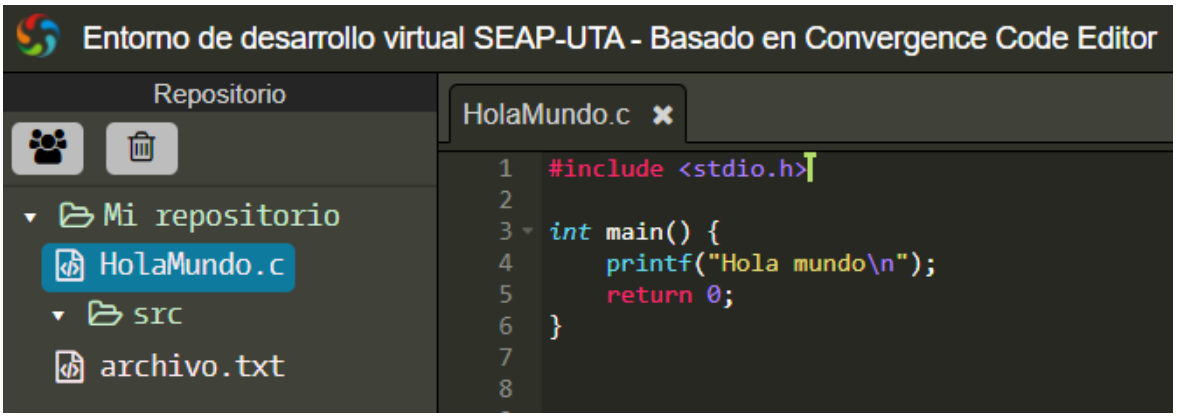

Figura 72. SEAP-UTA. Contexto de ejecución de la consola del ambiente virtual.

Figura 73. SEAP-UTA. Ejecución del código de programación dentro del repositorio.

La consola de comandos, funciona igual que cualquier otra, es por ello que para la utilización se recomienda ver algún tutorial en internet, es importante mencionar que los intérpretes o compiladores integrados en la consola virtual son; gcc para compilar C, g++ para compilar C++, python 3 con pip para interpretar Python3, Node para interpretar Javascript.

7.1. Usuario administrador de SEAP-UTA

La funcionalidad principal del administrador. es crear cuentas, el cual requiere acceso directo al servidor backend, se debe realizar una solicitud directa a través de una interfaz de la API REST del servidor backend, como en el siguiente ejemplo, utilizando la herramienta POSTMAN:

Los requerimientos de la solicitud, son los siguientes:

- Solicitud tipo POST.
- Enviar información en el Body de la solicitud, en formato Raw, utilizando una estructura JSON:

```
{
  "username": "nuevo_usuario",
  "password": "contraseña",
  "correo": "correo@alumnos.uta.cl"
}
```

Al enviar la solicitud, el servidor responderá si la cuenta de usuario ha sido creada correctamente.

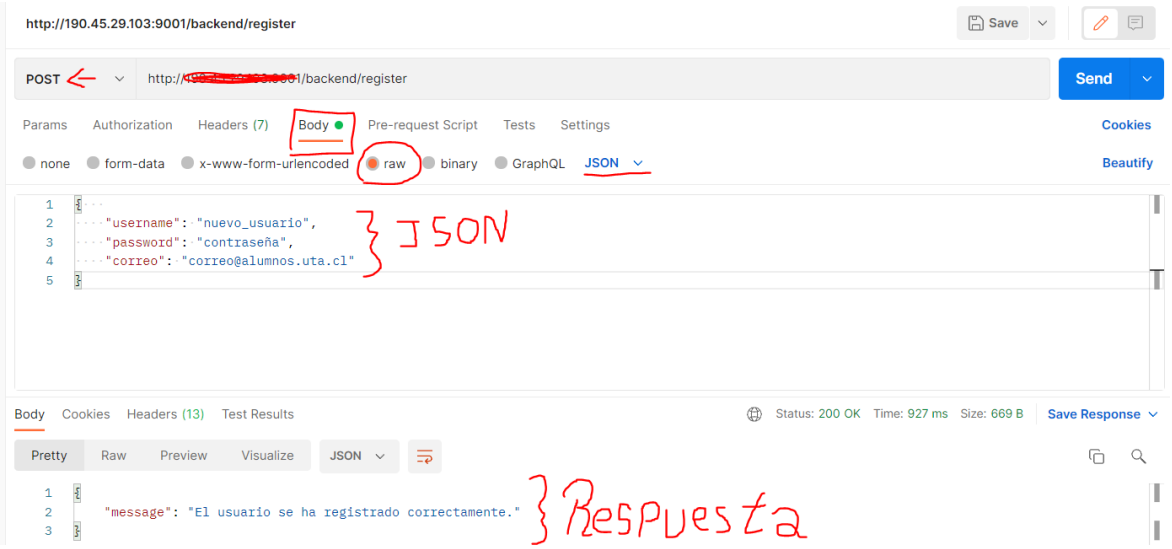

Figura 74. POSTMAN. Enviando una solicitud para crear una cuenta de usuario al servidor backend de SEAP-UTA.

7. CONCLUSIÓN

Al definir la propuesta de solución para este proyecto, fue un desafío a nivel personal y académico, ya que, se requirió de una determinación para realizar un sistema de gran complejidad sin tener la experiencia previa para desarrollar uno.

La acotación de requerimientos para el producto final, fue debido a la falta de estimación por la falta de experiencia en el desarrollo de este tipo de sistemas, además, de ser el primer proyecto desarrollado de forma individual. Y si bien, no se alcanzaron todos los requerimientos iniciales, la parte que corresponde a aulas virtuales, se puede decir que la implementación del núcleo del proyecto SEAP-UTA fue un éxito, se obtuvo un producto completamente funcional y desplegable en un sistema de producción.

Los conocimientos adquiridos luego de haber finalizado el proyecto, son de gran importancia para un buen perfil profesional, ya que, se requirió de un desarrollo full stack, es decir, tanto backend, como frontend.

Como trabajo a futuro, se espera desplegar el sistema en el departamento de ingeniería en computación e informática, de forma que sea una herramienta para que los alumnos aprendan a programar de forma colaborativa y participativa, y los profesores puedan enseñar a como programar de forma más interactiva.

8. REFERENCIAS

[1] Software Modeling & Design. 2011. Hassan Gomaa.

[2] Pattern-Oriented Software Architecture Vol. 4. 2007. Frank Buschmann.

[3] Apuntes del curso de aplicaciones distribuidas avanzadas - 2021. Diego Aracena.

[4] Visual Studio Code. 2021. Microsoft.
Enlace: https://code.visualstudio.com

[5] Proyecto en Github - "billchurch/webssh2". 2021. Bill Church.
Enlace: https://github.com/billchurch/webssh2

[6] Productos de MySQL. Oracle. 2021.
Enlace: https://www.mysql.com

[7] Node JS. Open JS Foundation. 2021.
Enlace: https://nodejs.org/es/

[8] Express para NodeJS. Open JS Foundation. 2021.
Enlace: https://expressjs.com/es/

[9] Documentación de Convergence. Convergence Labs. 2021.
Enlace: https://convergence.io/documentation/

[10] React. Facebook Open Source. 2021.
Enlace: https://es.reactjs.org

[11] Proyecto en Github - "convergencelabs/code-editor-demo". 2021.
Enlace: https://github.com/convergencelabs/code-editor-demo

[12] Nginx. F5 Company. 2021.
Enlace: https://www.nginx.com

[13] Docker Engine. Docker. 2021.
Enlace: https://www.docker.com

[14] Documentación de contenedor "linuxserver/openssh-server". linuxserver.io Community. 2021
Enlace: https://docs.linuxserver.io