Logo, company name

Description automatically generated

Profesor: Alonso Ramirez Oscar

Fecha: 15/06/22

Integrantes:

* FLORES ROMERO GUSTAVO
* VALDES IGLESIAS KARINA
* VARGAS SEGURA MARIANA YAZMIN

Proyecto Final

Desarrollo de aplicaiones

Contenido

[1 Descripción de la aplicación 3](#_Toc106151285)

[2 Funcionalidad 3](#_Toc106151286)

[3 Aplicación del control de versiones 5](#_Toc106151287)

[4 implementación del pipeline 6](#_Toc106151288)

[5 Ventajas de las herramientas 10](#_Toc106151289)

[6 Conclusiones 10](#_Toc106151290)

[ANEXO A: Requerimientos funcionales y no funcionales 11](#_Toc106151291)

[6.1 Requisitos funcionales 11](#_Toc106151292)

[6.1.1 CU-01: Agregar lugar 11](#_Toc106151293)

[6.1.2 CU-02: Agregar comentario al foro 11](#_Toc106151294)

[6.1.3 CU-03: Consulta de lugar 11](#_Toc106151295)

[6.1.4 CU-04: Verificar solicitud de lugar 12](#_Toc106151296)

[6.1.5 CU-05: Modificar lugar 12](#_Toc106151297)

[6.1.6 CU-06: Generar reporte 12](#_Toc106151298)

[6.1.7 CU-07: Añadir opinión 12](#_Toc106151299)

[6.1.8 CU-08: Consultar foro 12](#_Toc106151300)

[6.1.9 CU-09 Registrar estudiante 12](#_Toc106151301)

[6.2 Requisitos no funcionales 13](#_Toc106151302)

[ANEXO B: Modelo y descripción de casos de uso 13](#_Toc106151303)

[1.1.1 CU-01 Agregar lugar 14](#_Toc106151304)

[1.1.2 CU-02 Agregar comentario al foro 16](#_Toc106151305)

[1.1.3 CU-03 Consulta de lugar 17](#_Toc106151306)

[1.1.4 CU-04 Verificar solicitud de lugar 18](#_Toc106151307)

[1.1.5 CU-05 Modificar lugar 19](#_Toc106151308)

[1.1.6 CU-07 Generar solicitud de lugar 21](#_Toc106151309)

[1.1.7 CU-07 Añadir opinión 22](#_Toc106151310)

[1.1.8 CU-08 Consultar foro 23](#_Toc106151311)

[1.1.9 CU-09 Registrar estudiante 26](#_Toc106151312)

[ANEXO C: Diagrama de despliegue 28](#_Toc106151313)

# Descripción de la aplicación

La aplicación de escritorio Vida Foránea su objetivo principal es apoyar a los estudiantes de la FEI en el descubrimiento de establecimientos relacionados con el consumo de alimentos, papelerías , centros de copias o tiendas de tecnología , al igual que lugares de ocio ; junto con un apartado por carrera correspondiente al alumno para que describa su experiencia en la licenciatura al igual que obtener una retroalimentación por miembros de semestres superiores , con el fin de mejorar la convivencia entre los estudiantes.

|  |  |
| --- | --- |
| Aplicación de Cliente de Escritorio | C# con apoyo de la IDE Visual Studio |
| Servidor | Python con Flask RESTful, flask SQLAlchemy, Marshmallow y JWT Manager |
| Pruebas automatizadas | Postman |
| Despliegue del servidor | Docker |

# Funcionalidad

Nuestra visión final de sistema que cuente con un amplio registro de lugares cercanos a la Facultad de las categorías antes mencionadas, y que cada una de estas secciones posea información relevante para su consulta, que en este caso sería una puntuación y comentarios dados por los usuarios, posibilidad de carga de fotos y videos de los sitios recomendados, foro de recomendaciones relevantes a cada una de las licenciaturas de la FEI, al igual que un apartado de consulta sobre los descuentos por parte de la UV nivel Xalapa.

Específicamente, estamos contemplando los siguientes puntos de información de cada lugar:

* Nombre del lugar
* Ubicación
* Foto
* Puntuación
* Comentarios de usuarios
* Servicios que ofrece
* Horarios

Se permitirá el registro de nuevos lugares, mediante un formulario implementado dentro de la misma aplicación. Estos registros serán verificados previamente por un administrador del sistema antes de su publicación con el propósito de asegurar que los datos sean correctos, no sean duplicados y que la información sea relevante.

Se contará con dos tipos de usuario, administrador y estudiante. Se tienen planteadas las siguientes funcionalidades, además del inicio de sesión y registro de estudiante:

* Administrador
  + Relacionado a los lugares
    - Modificar
    - Consultar
    - Aprobar
    - Agregar
  + Relacionado a las opiniones
    - Consultar
    - Eliminar
  + Relacionado a foros
    - Agregar
    - Consultar
  + Relacionado a los comentarios
    - Eliminar
    - Generar reportes
* Estudiante
  + Relacionado a los lugares
    - Agregar
    - Consultar
  + Relacionado a los comentarios
    - Agregar
    - Eliminar
  + Relacionado a las opiniones
    - Agregar
    - Consultar

En la siguiente tabla se representan las funcionalidades que fueron implementadas tanto en el cliente como las peticiones necesarias por el servidor para completar esas tareas:

|  |  |
| --- | --- |
| Funcionalidad | Completado |
| Modificar lugar | ✔ | |
| Consultar lugar | ✔ | |
| Aprobar lugar | ✔ | |
| Agregar lugar | ✔ | |
| Consultar lugares | ✔ | |
| Eliminar comentario | ✔ | |
| Agregar comentario | ✔ | |
| Consultar comentario | ✔ | |
| Consultar comentarios | ✔ | |
| Agregar opinión | ✔ | |
| Eliminar opinión | ✔ | |
| Consultar opinión | ✔ | |
| Consultar opiniones | ✔ | |
| Consultar foro | ✔ | |
| Agregar foro | ✔ | |
| Consultar foros | ✔ | |
| Generar reportes | ✘ | |

# Aplicación del control de versiones

Texto

Descripción generada automáticamenteSe utilizo GitHub para el control de versiones de la aplicación, el cual se dividió en el apartado del cliente y el servidor para mantener un orden cada que los cambios fueran subidos, la organización se muestra en la siguiente figura.

Captura de pantalla de un celular

Descripción generada automáticamentePara evitar conflictos con los commits de los tres miembros, se utilizó la plantilla definida de GitHub para el IDE de Visual Studio en el .gitignore.

Cada uno de los integrantes creo su rama para dedicarse a subir sus cambios periódicamente, donde al asegurarse que todo funcionará correctamente se avisaba a los otros compañeros que se realizaría un pull request a la rama principal, evitando que se borrarán actualización por otro y mantener a todos con la versión reciente del proyecto.

En los commits se escribían títulos significativos para que los miembros entendieran la razón por el cambio, especificando si se trataba de que corrección de errores, nuevas funcionalidades o modificación de ya existentes y si era necesario se terminaban de especificar en el apartado de mensajes.

Como ayuda al equipo se utilizó la herramienta de GitHub Desktop para facilitar los commits, usado también por la rapidez que proporciona al resolver conflictos entre los merge , ya que permite conocer cuáles son los archivos que no permiten continuar con la operación.

A continuación, se muestra un ejemplo de commit para un mayor entendimiento.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

A continuación, se presenta el flujo de trabajo de todo el equipo, representado por Commits.

Graphical user interface, application

Description automatically generated

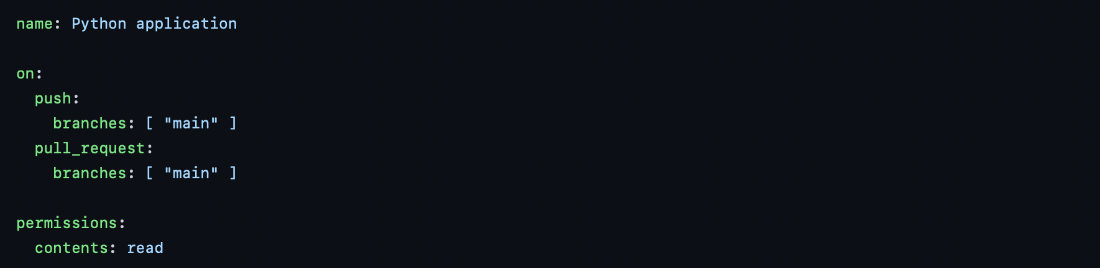
# implementación del pipeline

La Pipeline fue implementada en la parte de la API Restful (servidor) utilizando la herramienta de Github Actions. Esta herramienta permite tener una integración y despliegue continuos al automatizar la construcción, las pruebas y el despliegue. Cada vez que sucede un push, este pipeline se ejecuta automáticamente.

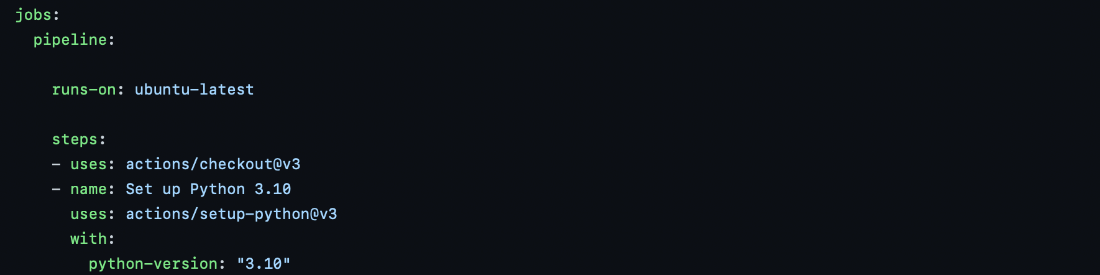
En nuestro caso, construimos el API Restful en Python, aplicamos pruebas utilizando pytest y desplegando en Docker.

A continuación, se explica cada parte del código que permite a Github Actions implementar la Pipeline.

En esta parte del código, se declara la Branch en la cual cada vez que se haga un push o un pull\_request se va a ejecutar la Pipeline. También, se le puso a la Pipeline el permiso de solo lectura.



En GitHub Actions se denomina “workflow” al flujo de trabajo automatizado que se ejecuta mediante un archivo .yml, cada workflow puede ejecutar uno o más “Jobs”, en este caso se le denomino “pipeline” al trabajo ejecutado por el workflow. El flujo de trabajo va a correr en un ambiente de Ubuntu, a continuación, se describen los pasos que se siguen cada que se realiza un push en el main.

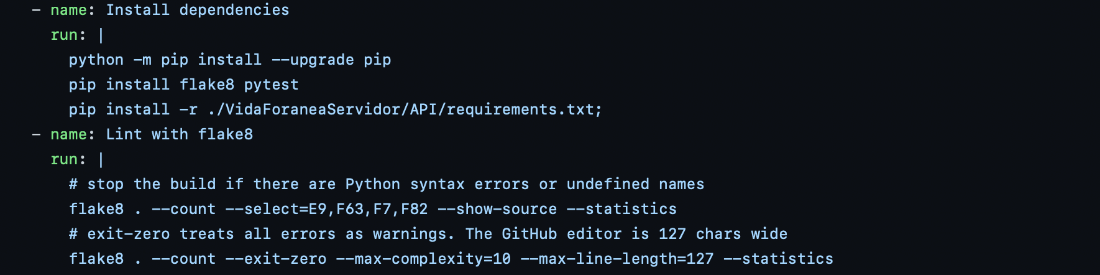


Los primero que se declara es que se usa “actions/checkout@v3”, una acción oficial de GitHub que permite verificar a repositorio para que el workflow pueda acceder a él.

**Set up Python 3.10:** En esta sección se prepara el ambiente para utilizar Python 3.10 para distintos propósitos durante el workflow. Utilizando la accion oficial “actions/setup-python@v3”.

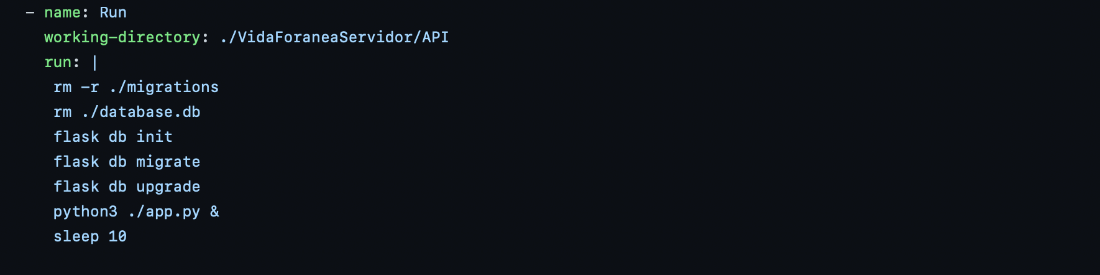
**Install dependecies:** Instala las dependencias que se utilizan en el workflow, entre ellas pip (Sistema de gestion de paquetes de Python) y pytest (Para las pruebas). Ademas, se instalan los requerimientos de la API, definidos en el archivo “requeriments.txt”.

**Lint winth flake8:** permite detener la construcción si existen errores de sintaxis en Python.



**Run:** En esta sección se definen los comandos a ejecutar para correr la API. Primero, se define el directorio en el que se va a trabajar. Después, se borra el directorio migration y el archivo database.db. Los siguientes comandos por ejecutar son flask db init, flask db migrate, flask db upgrate, que inicializan la base de datos desde cero.

Para correr la aplicación se utiliza el comando “python3 ./app.py &” (El simbolo & se utiliza para ejecutar este comando en segundo plano, de esta forma el servidor de la API sigue corriendo mientras posteriormente se ejecutan las pruebas)



**Test**: En esta sección se corre el comando pytest, busca los casos de prueba en los archivos del proyecto mediante el identificador “Test”. Si alguno de los casos falla, se detiene la ejecución del workflow.

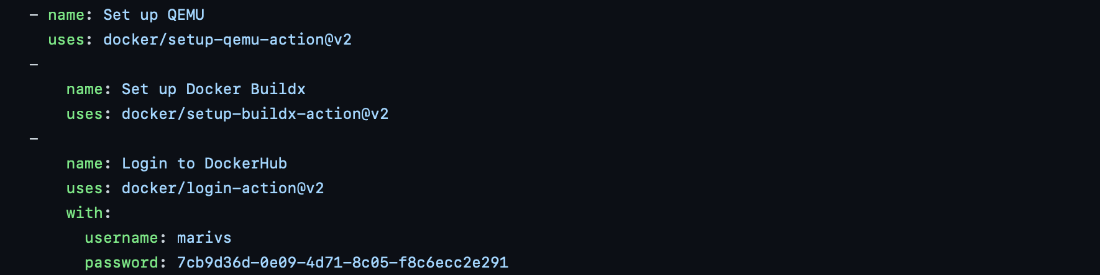


Cuando se tiene que la API funciona y los casos de prueba pasan, se empieza con la configuración para construir y desplegar con ayuda de docker.

**Set up QEMU:** Es una acción de github que permite inicializar QEMU, un emulador y virtualizador.

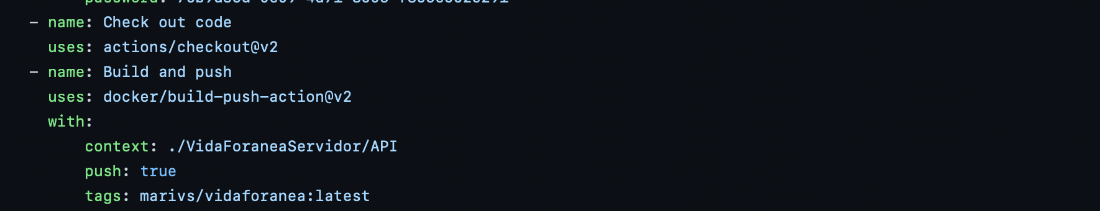
**Set up Docker Buildx:** Acción de github que permite inicializar un builder que se puede utilizar en el “workflow” , usualmente es utilizado para crear imágenes de docker.

**Login to DockerHub:** Acción de github que permite iniciar sesión con el usuario y la contraseña, que token de Docker.



**Check out code:** Se declara “actions/checkout@v2”, una acción oficial de GitHub que permite verificar a repositorio para que el workflow pueda acceder a él.

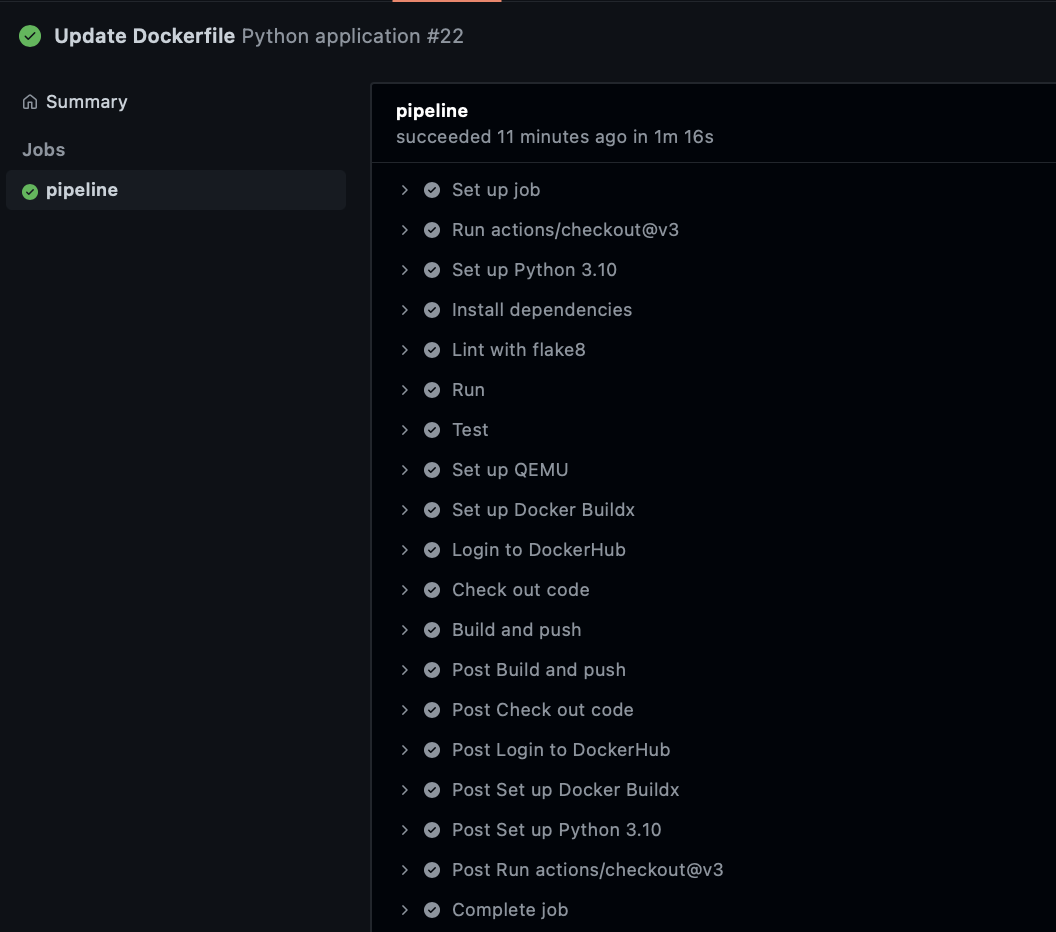
**Build and push:** Acción de github que permite construir la imagen de docker y posteriormente subirla al Hub de docker con el nombre de “vidaforanea”.



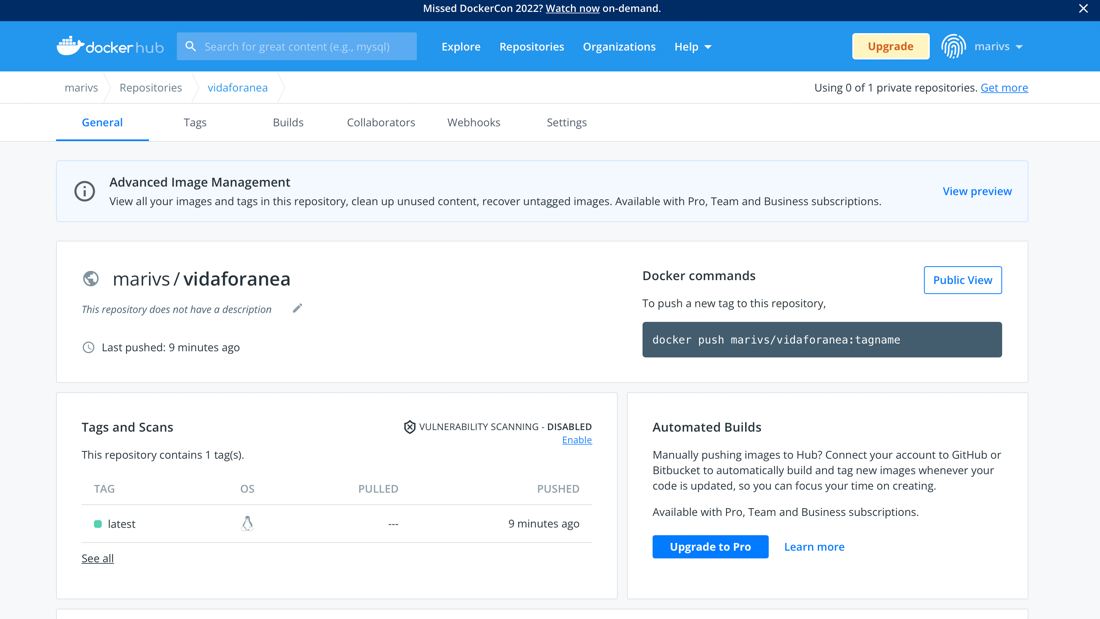
Por último, podemos ver que cuando se hizo un push, se ejecutó el pipeline y dio un resultado exitoso



Observamos que cada paso fue concluido con éxito



Y, por último, podemos ver que en DockerHub se subió la imagen del servidor que se construyó, probó y ejecutó en el pipeline de Github Actions.



# Ventajas de las herramientas

|  |  |
| --- | --- |
| Herramienta | Ventajas |
| GitHub | * Se guardan los cambios de código efectuados a un archivo * Permite contribuir entre ramas (hacer fork de dos ramas de trabajo) * Compatibilidad en distintas plataformas (en nuestro caso, Windows y Mac) |
| Github Actions | * Permite la integración y entrega continua * Pruebas automatizadas * Despliegue con Docker |
| GitHub Desktop | * Rapidez de contribución * Mejor visibilidad en la evolución de proyectos * Interfaz intuitiva * Curva de aprendizaje más rápida comparada con el uso de consola |
| Postman | * Interfaz bastante sencilla y fácil de aprender * Permite la colaboración del equipo * Permite trabajar con colecciones definidas en la API * Permite la creación y aplicación de pruebas |

# Conclusiones

La implementación de la pipeline fue exitosa; pudimos automizar los procesos de compilación, construcción, aplicación de pruebas y despliegue del servidor, lo que aumentó la eficiencia del equipo y el tiempo de trabajo que se le dedica al desarrollo o programación de este. En otras palabras, ya no tenemos que manualmente estar creando la imagen de Docker del servidor y subirla al DockerHub, sin antes haber probado de forma manual que los cambios realizados a los distintos métodos del servidor funcionen de forma adecuada; todo esto ya se realiza automáticamente al realizar un push sobre la rama main.

Nos parece que es bastante útil implementar la integración, entrega y despliegue continuo, porque, por ejemplo, cada vez que alguien hace un cambio en alguna parte del código y hace un push a la rama main, podemos consultar, gracias a la pipeline, si todo funciona correctamente al ver que la acción fue concluida; pero, si existe un error, se detiene la pipeline y no permite que este cambio se despliegue al DockerHub, evitando que se esté trabajando con una versión nueva pero que no funciona como debería. Así, se puede consultar qué fue lo que falló en las pruebas y modificar y volver a realizar el push a la rama main.

# ANEXO A: Requerimientos funcionales y no funcionales

## Requisitos funcionales

### CU-01: Agregar lugar

RQF-01: Cuando el estudiante seleccione la opción "Enviar”, si los campos son válidos, el SVF debe generar la solicitud para agregar el lugar a la base de datos.

RQF-02: Cuando el estudiante seleccione la opción "Enviar”, si los campos no son válidos, el SVF muestra un mensaje de campos inválidos

RQF-03: Cuando el administrador seleccione la opción "Enviar”, si los campos son válidos, el SVF debe guardar el lugar en la base de datos.

RQF-04: Cuando el administrador seleccione la opción "Enviar”, si los campos no son válidos, el SVF muestra un mensaje de campos inválidos.

RQF-05: Cuando el estudiante o el administrador selecciona la opción “Cancelar”, el SVF deberá cerrar la ventana.

### CU-02: Agregar comentario al foro

RQF-06: Cuando el estudiante, administrador o moderador seleccione la opción “Enviar”, si los campos son válidos, el SVF debe guardar la información de comentario.

RQF-07: Cuando el estudiante, administrador o moderador seleccione la opción “Enviar”, si los campos no son válidos, el SVF muestra un mensaje de campos inválidos.

RQF-08: Cuando el estudiante, administrador o moderador selecciona la opción “Cancelar”, el SVF deberá cerrar el apartado desplegado.

### CU-03: Consulta de lugar

RFQ-09: Cuando el estudiante, administrador o moderador selecciona un lugar de la vista, el SVF debe mostrar la información del lugar.

RQF-10: Cuando el administrador selecciona la opción “Modificar”, el SVF debe cerrar la ventana actual y extender al caso de uso “CU-06: Modificar lugar”

RQF-11: Cuando el estudiante, administrador o moderador selecciona la opción “Regresar”, el SVF debe cerrar la ventana actual y regresar a la anterior.

RQF-12: Cuando el administrador o estudiante selecciona la opción "Añadir comentario”, el SVF debe cerrar la ventana actual y extender al caso de uso “CU-07 añadir opinión”

RQF-13: Cuando el administrador o estudiante selecciona la opción "Eliminar comentario”, el SVF debe eliminar el comentario de la base de datos.

### CU-04: Verificar solicitud de lugar

RQF-14: Cuando el administrador seleccione una solicitud de la lista de solicitudes pendientes, el SVF deberá mostrar la información de esta solicitud.

RQF-15: Cuando el administrador seleccione la opción “Aprobar”, el SVF deberá registrar el lugar en la base de datos como aprobado y notificar al estudiante por correo electrónico.

RQF-16: Cuando el administrador seleccione la opción “Desaprobar”, el SVF deberá eliminar la solicitud de la base de datos y abrir una ventana para introducir un comentario para notificar al estudiante el por qué no fue aprobado.

### CU-05: Modificar lugar

RQF-17: Cuando el administrador selecciona la opción “Actualizar”, si los campos son válidos, el SVF debe guardar las modificaciones realizadas al lugar.

RQF-18: Cuando el administrador selecciona la opción “Actualizar”, si los campos no son válidos, el SVF debe mostrar un mensaje de campos inválidos.

### CU-06: Generar reporte

RQF-19: Cuando el administrador selecciona la opción “Regresar”, el SVF debe cerrar la ventana actual y regresar a la anterior.

### CU-07: Añadir opinión

RQF-20: Cuando el estudiante seleccione la opción “Enviar”, si los campos son válidos, el SVF deberá agregar la opinión al lugar.

RQF-21: Cuando el estudiante seleccione la opción “Enviar”, si los campos no son válidos, el SVF deberá mostrar un mensaje de campos inválidos

RQX-22: Cuando el estudiante seleccione la opción "Cancelar”, el SVF deberá cerrar la ventana.

### CU-08: Consultar foro

RQF-23: Cuando el estudiante, administrador o moderador selecciona la opción “Regresar”, el SVF debe cerrar la ventana actual y regresar a la anterior.

RQF-24: Cuando el administrador o estudiante selecciona la opción "Añadir comentario”, el SVF debe cerrar la ventana actual y extender al caso de uso “CU-02 agregar comentario al foro.”

### CU-09 Registrar estudiante

RQF-25: Cuando el estudiante, administrador o moderador selecciona la opción “Registrar”, el SVF debe guardar los datos escritos y mandar un mensaje de éxito.

RQF-26: Cuando el estudiante, administrador o moderador selecciona la opción “Cancelar”, el SVF debe cerrar la ventana actual y regresar a la anterior.

## Requisitos no funcionales

RQNF-01: El 100% de las contraseñas de los usuarios registrados se encuentran cifrados

RQNF-02: El sistema se encuentran funcionando al 80% durante horas y días hábiles

RQNF-03: El sistema no posee información duplicada en ninguno de sus apartados

RQNF-04: A los usuarios con rol de estudiante solo se les permitirá la eliminación de los comentarios propios.

RQNF-05: El sistema tarda menos de 5 segundos en realizar una consulta al lugar

RQNF-06: El usuario inexperto tarda menos de 20 minutos en entender el funcionamiento básico del sistema.

# ANEXO B: Modelo y descripción de casos de uso

Diagrama

Descripción generada automáticamente

### CU-01 Agregar lugar

|  |  |
| --- | --- |
| **ID:** | CU-01 |
| **Nombre:** | Agregar lugar |
| **Responsable:** | Gustavo Flores Romero |
| **Descripción:** | Se desea crear una solicitud para agregar un lugar |
| **Actor(es):** | **Estudiante o administrador** |
| **Disparador:** | El **responsable** da clic en el botón “Añadir lugar” de la ventana “MenuPlaces” |
| **Precondiciones:** | PRE-01: Se tiene iniciada una sesión de Estudiante o administrador |
| **Flujo Normal:** | 1. El **Cliente** muestra la ventana “AddPlace” con los campos Nombre, Ubicación, un ComboBox para seleccionar el tipo de lugar (Comida, Papelería u Ocio), Servicios que ofrece el lugar, CheckBox’es y seleccionadores para el horario de cada día de la semana, junto con botones para “Enviar”, “Cancelar”. (ver EX1). 2. El **Estudiante o Administrador** ingresa la información correspondiente de los campos y da clic en el botón “Enviar” (ver FA-2.1) 3. El **Cliente** valida los campos ingresados y, si son correctos, crea la solicitud del lugar y la manda al servidor y muestra una ventana de diálogo con el mensaje “Se ha registrado la solicitud” y el botón de “Ok” (ver FA-3.1). 4. El **Estudiante o Administrador** da clic en “Ok” de la ventana de diálogo 5. El **Cliente** cierra la ventana “AddPlace” y regresa a la ventana “MenuPlaces” 6. Fin del caso de uso |
| **Flujos Alternos:** | FA-2.1: Cancelar registro de solicitud   1. El **Estudiante o Administrador** selecciona el botón “Cancelar” de la ventana “AddPlace” 2. El **Cliente** regresa a la ventana “MenuPlaces”     FA-3.1: Campos inválidos   1. El **Cliente** valida los campos ingresados, detecta los campos inválidos y muestra la ventana de diálogo notificando que existen campos inválidos 2. El **Estudiante o Administrador** da clic en el botón “Ok” 3. El **Cliente** cierra la ventana de diálogo. 4. Regresar al paso 3 del flujo normal. |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: La solicitud queda registrada en el servidor, y, si es un estudiante el que manda la solicitud, se mantiene como pendiente de aprobación, si es un administrador, se aprueba automáticamente |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

### CU-02 Agregar comentario al foro

|  |  |
| --- | --- |
| **ID:** | CU-02 |
| **Nombre:** | Agregar comentario al foro |
| **Responsable:** | Gustavo Flores Romero |
| **Descripción:** | Se desea agregar un comentario al foro correspondiente a la licenciatura del estudiante |
| **Actor(es):** | **Estudiante** |
| **Disparador:** | El **responsable** da clic en el botón “Foros” de la “PrincipalMenu” |
| **Precondiciones:** | PRE-01: Se tiene iniciada una sesión de Estudiante |
| **Flujo Normal:** | 1. El **Cliente** recupera del servidor los comentarios correspondientes a la licenciatura del estudiante y los muestra en pantalla para consultarlos. También, muestra una barra hasta abajo para ingresar un nuevo comentario junto con los botones “Enviar” y “Regresar”. (ver EX1). 2. El **Estudiante o Administrador** ingresa su comentario y da clic en el botón “Enviar” (ver FA-2.1) 3. El **Cliente** valida los campos ingresados y, si son correctos y no existen palabras en el diccionario de palabras no permitidas, se crea el comentario, se manda al servidor y muestra una ventana de diálogo con el mensaje “Se ha registrado el comentario en el foro” y el botón de “Ok” (ver FA-3.1 y FA-3.2). 4. El **Estudiante o Administrador** da clic en “Ok” de la ventana de diálogo 5. Fin del caso de uso |
| **Flujos Alternos:** | FA-3.1: Filtro de palabras   1. El **Cliente** muestra una ventana de diálogo donde informa que se ingresaron palabras no permitidas en el foro 2. El **Estudiante o Administrador** da clic en el botón “Ok” de la ventana de diálogo 3. Regresa al paso 2 del flujo normal     FA-3.2: Campos inválidos   1. El **Cliente** valida los campos ingresados, detecta los campos inválidos y muestra la ventana de diálogo notificando que existen campos inválidos 2. El **Estudiante o Administrador** da clic en el botón “Ok” 3. El **Cliente** cierra la ventana de diálogo. 4. Regresar al paso 3 del flujo normal. |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: El comentario queda registrado en el foro correspondiente |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

### CU-03 Consulta de lugar

|  |  |
| --- | --- |
| **ID:** | CU-03 |
| **Nombre:** | Consulta de lugar |
| **Responsable:** | Gustavo Flores Romero |
| **Descripción:** | Se desea consultar la información de un lugar |
| **Actor(es):** | **Estudiante o administrador** |
| **Disparador:** | El **estudiante/administrador** da clic sobre un lugar de la lista de lugares en la ventana “PlaceList” |
| **Precondiciones:** | PRE-01: Se tiene iniciada una sesión de Estudiante |
| **Flujo Normal:** | 1. El **Cliente** recupera del servidor la información del lugar seleccionado, mostrando los campos del nombre del lugar, la imagen del lugar, ubicación, los servicios que ofrece, el horario y las opiniones asignadas al lugar. También, muestra una barra hasta abajo para ingresar una nueva opinión junto con los botones “Enviar” y “Regresar”. (ver EX1). 2. El **Estudiante o Administrador** consulta la información del lugar y da clic en el botón “Regresar” (ver FA-2.1) 3. El **Cliente** regresa a la ventana de “PrincipalMenu” 4. Fin del caso de uso |
| **Flujos Alternos:** | FA-2.1 Agregar opinión   1. El **Cliente** extiende al CU-07 Añadir Opinión 2. Regresa al paso 2 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: Si el **Estudiante** agrega una opinión, queda registrada y ligada al lugar |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** | CU-07 Añadir opinión |

### CU-04 Verificar solicitud de lugar

|  |  |
| --- | --- |
| **ID:** | CU-04 |
| **Nombre:** | Verificar solicitud de lugar |
| **Responsable:** | Mariana Yazmin Vargas Segura |
| **Descripción:** | Se desea aceptar o rechazar una solicitud de un lugar propuesta por un estudiante |
| **Actor(es):** | **Administrador** |
| **Disparador:** | El **administrador** da clic en un lugar específico de la GUI-XX “Listado de solicitudes de lugar” |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El **Cliente** despliega la GUI-XX “Lugar específico” y recupera el LUGAR seleccionado. (ver EX1). 2. El **administrador** da clic en el botón “Aprobar” (ver FA-3.1) (ver FA-3.2) 3. El **Cliente** manda la solicitud de aprobar **(**ver FA-4.1 y FA-4.2). 4. El **Cliente** recibe un código de respuesta exitoso, muestra un mensaje de confirmación, cierra la GUI-XX “Lugar específico” y regresa a la GUI-XX “Listado de solicitudes de lugar” (ver FA-6.1) 5. Fin del caso de uso |
| **Flujos Alternos:** | FA-2.1: Clic en el botón regresar   1. El **Administrador** selecciona el botón “regresar” de la ventana “Lugar específico” 2. El **Cliente** regresa a la ventana “Lista de solicitudes de lugar” 3. Termina caso de uso   FA-2.2: Clic en el botón rechazar   1. El **Administrador** selecciona el botón “rechazar” de la ventana “Lugar específico” 2. El **Cliente** regresa a la ventana “Lista de solicitudes de lugar” 3. Termina caso de uso   FA-6.1: Código de respuesta fallido   1. El **Cliente** recibe un código de respuesta fallido, despliega un cuadro de dialogo avisando la razón por la que no se completó la solicitud con el botón de “Ok” 2. El **administrador** da clic en “Ok” 3. Se regresa al paso 6 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: El LUGAR queda actualizado en la base de datos. |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

### CU-05 Modificar lugar

|  |  |
| --- | --- |
| **ID:** | CU-05 |
| **Nombre:** | Modificar lugar |
| **Responsable:** | Mariana Yazmin Vargas Segura |
| **Descripción:** | Se desea crear una solicitud para modificar un lugar |
| **Actor(es):** | **Administrador** |
| **Disparador:** | El **administrador** da clic en el botón “Modificar lugar” de la ventana “Lugar específico” |
| **Precondiciones:** | PRE-01: Se tiene iniciada una sesión de Estudiante o administrador |
| **Flujo Normal:** | 1. El **Cliente** manda una petición al para recuperar el LUGAR seleccionado junto con las OPINIÓN. (ver EX1). Muestra la GUI-XX “Modificar lugar” con los campos Nombre, Ubicación, un ComboBox para seleccionar el tipo de lugar (Comida, Papelería u Ocio), Servicios que ofrece el lugar, CheckBox’es y seleccionadores para el horario de cada día de la semana habilitados para su modificación, junto con botones para “Guardar”, “Cancelar”. (ver EX1). 2. El **Administrador** ingresa la información correspondiente de los campos y da clic en el botón “Guardar” (ver FA-2.1) 3. El **Cliente** valida los campos ingresados y, si son correctos, crea la solicitud del lugar y la manda al servidor y muestra una ventana de diálogo con el mensaje “Se ha modificado la solicitud” y el botón de “Ok” (ver FA-3.1). 4. El **Cliente** recibe un código de respuesta exitoso, muestra la Opinión en la GUI-XX “Lugar” (ver FA-7.3) 5. El **Administrador** da clic en “Ok” de la ventana de diálogo 6. El **Cliente** cierra la ventana “Modificar lugar” y regresa a la ventana “Lista de lugares” 7. Fin del caso de uso |
| **Flujos Alternos:** | FA-2.1: Cancelar modificación de solicitud   1. El **Administrador** selecciona el botón “Cancelar” de la ventana “Modificar lugar” 2. El **cliente** regresa a la ventana “Lista de lugares”     FA-3.1: Campos inválidos   1. El **Cliente** valida los campos ingresados, detecta los campos inválidos y muestra la ventana de diálogo notificando que existen campos inválidos 2. El **Estudiante o Administrador** da clic en el botón “Ok” 3. El **Cliente** cierra la ventana de diálogo. 4. Regresar al paso 3 del flujo normal.   FA-4.3: Código de respuesta fallido   1. El **Cliente** recibe un código de respuesta fallido, despliega un cuadro de dialogo avisando la razón por la que no se completó la solicitud con el botón de “Ok” 2. El **administrador** da clic en “Ok” 3. Se regresa al paso 7 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: La solicitud queda registrada en el servidor, y, si es un estudiante el que manda la solicitud, se mantiene como pendiente de aprobación, si es un administrador, se aprueba automáticamente |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

### CU-07 Generar solicitud de lugar

|  |  |
| --- | --- |
| **ID:** | CU-06 |
| **Nombre:** | Generar solicitud de lugar |
| **Responsable:** | Mariana Yazmin Vargas Segura |
| **Descripción:** | Se desea aceptar o rechazar una solicitud de un lugar propuesta por un estudiante |
| **Actor(es):** | **Administrador** |
| **Disparador:** | El **administrador** da clic en el botón “Generar Reporte” de la GUI-XX “Listado de lugares” |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El **Cliente** manda una petición para recuperar la lista de lugares. (ver EX1). Muestra la GUI-XX “Reporte de lugares”. (ver EX1). 2. El **administrador** da clic en el botón “regresar”. 3. El **Cliente** cierra la GUI-XX “Reporte de lugares” y regresa a la GUI-XX “Listado de solicitudes de lugar” 4. Fin del caso de uso |
| **Flujos Alternos:** |  |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: El LUGAR queda actualizado en la base de datos. |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

### CU-07 Añadir opinión

|  |  |
| --- | --- |
| **ID:** | CU-07 |
| **Nombre:** | Añadir opinión |
| **Responsable:** | Karina Valdés Iglesias |
| **Descripción:** | Se desea agregar una opinión a un lugar especifico |
| **Actor(es):** | **Estudiante /Administrador** |
| **Disparador:** | El **estudiante / administrador** da clic en un lugar específico de la GUI-XX “Listado de lugares” |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El **Cliente** recupera el LUGAR y despliega la GUI-XX “Lugar específico” y manda una petición al **Servidor** para recuperar el LUGAR seleccionado junto con las OPINIÓN. (ver EX1). 2. El **estudiante / administrador** ingresa su comentario y da clic en el botón “Enviar” (ver FA-2.1) 3. El **Cliente** valida los campos ingresados y, si son correctos, no posee palabras prohibidas del diccionario, se crea la OPINIÓN (ver FA-2.1 y FA-2.2). 4. El **Cliente** manda la OPINION para ser guardada en la base de datos y muestra la Opinión en la GUI-XX “Lugar” (ver FA-4.1) 5. El **Estudiante o Administrador** da clic en “Regresar” de la GUI-XX Lugar 6. El **Cliente** cierra la GUI-XX Lugar y regresa a la GUI-XX “Listado de lugares” 7. Fin del caso de uso |
| **Flujos Alternos:** | FA-3.1: Filtro de palabras   1. El **Cliente** muestra una ventana de diálogo donde informa que se ingresaron palabras no permitidas en el foro 2. El **Estudiante o Administrador** da clic en el botón “Ok” de la ventana de diálogo 3. Regresa al paso 3 del flujo normal     FA-3.2: Campos inválidos   1. El **Cliente** valida los campos ingresados, detecta los campos inválidos y muestra la ventana de diálogo notificando que existen campos inválidos 2. El **Estudiante o Administrador** da clic en el botón “Ok” 3. El **Cliente** cierra la ventana de diálogo. 4. Regresar al paso 3 del flujo normal.   FA-4.3: Código de respuesta fallido   1. El **Cliente** recibe un código de respuesta fallido, despliega un cuadro de dialogo avisando la razón por la que no se completó la solicitud con el botón de “Ok” 2. El **estudiante/administrador** da clic en “Ok” 3. Se regresa al paso 3 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: La OPINIÓN queda registrada en el LUGAR correspondiente |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

### CU-08 Consultar foro

|  |  |
| --- | --- |
| **ID:** | CU-08 |
| **Nombre:** | Consultar foro |
| **Responsable:** | Karina Valdés Iglesias |
| **Descripción:** | Se desea consultar los COMENTARIO de un FORO |
| **Actor(es):** | **Estudiante** |
| **Disparador:** | El **estudiante** da clic a la GUI-XX “Menú” en el botón de Foro |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El **Cliente** recupera los COMENTARIO de la licenciatura relacionada con el **estudiante** y despliega la GUI-XX “Foro” con los botones de “Regresar” y “Añadir”. (ver EX1) 2. El **estudiante** visualiza los COMENTARIO y da clic en el botón Regresar de la GUI-XX “Foro” (ver FA 4.1) 3. Termina el caso de uso |
| **Flujos Alternos:** | FA-3.1 Código de respuesta fallido   1. El **Cliente** recibe un código de respuesta fallido, despliega un cuadro de dialogo avisando la razón por la que no se completó la solicitud con el botón de “Ok” 2. El **estudiante/administrador** da clic en “Ok” 3. Se regresa al paso 3 del flujo normal   FA-4.1 Agregar comentario   1. El **Cliente** extiende al CU-02 Agregar comentario al foro 2. Regresa al paso 4 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: El **estudiante** visualizo los COMENTARIO de su licenciatura |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** | CU-02 Agregar comentario al foro |

|  |  |
| --- | --- |
| **ID:** | CU-08 |
| **Nombre:** | Consultar foro |
| **Responsable:** | Karina Valdés Iglesias |
| **Descripción:** | Se desea consultar los COMENTARIO de un FORO |
| **Actor(es):** | **Moderador** |
| **Disparador:** | El **moderador** da clic a la GUI-XX “Menú” en el botón de Foro |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El **Cliente** recupera los FORO con sus COMENTARIO y despliega la GUI-XX “Foros” con el botón de “Regresar”. (ver EX1) (ver FA 1.1) 2. El **moderador** da clic al FORO que desea consultar de la GUI-XX “Foros” 3. El **Cliente** despliega la GUI-XX “Foro” con los COMENTARIO correspondientes 4. El **moderador** visualiza los COMENTARIO y da clic en el botón Regresar de la GUI-XX “Foro” (ver FA-4.1) (ver FA-4.2) 5. Termina el caso de uso |
| **Flujos Alternos:** | FA-1.1 Código de respuesta fallido   1. El **Cliente** recibe un código de respuesta fallido, despliega un cuadro de dialogo avisando la razón por la que no se completó la solicitud con el botón de “Ok” 2. El **moderador** da clic en “Ok” 3. Se regresa al paso 3 del flujo normal   FA-4.1 Agregar comentario   1. El **Cliente** extiende al CU-02 Agregar comentario al foro 2. Regresa al paso 4 del flujo normal   FA-4.2 Agregar comentario   1. El **Cliente** extiende al CU-02 Agregar comentario al foro 2. Regresa al paso 4 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **moderador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: El **moderador** visualizo los COMENTARIO de un FORO |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** | CU-02 Agregar comentario al foro |

### CU-09 Registrar estudiante

|  |  |
| --- | --- |
| **ID:** | CU-07 |
| **Nombre:** | Añadir opinión |
| **Responsable:** | Karina Valdés Iglesias |
| **Descripción:** | Se desea agregar una opinión a un lugar especifico |
| **Actor(es):** | **Estudiante** |
| **Disparador:** | El **estudiante** da clic en el botón de Registrarse en la GUI-XX “Login” |
| **Precondiciones:** |  |
| **Flujo Normal:** | 1. El **Cliente** despliega la GUI-XX “Registro de estudiante” con los campos nombre, matricula, licenciatura, contraseña junto con los botones de Guardar y Regresar 2. El **estudiante** ingresa sus datos y da clic en el botón “Guardar” (ver FA-2.1) 3. El **Cliente** valida los campos ingresados y, si son correctos, se crea el ESTUDIANTE, se lo manda a la **API (**ver FA-3.1). 4. La **API** recibe los datos para la creación de un ESTUDIANTE, añadiéndola a la base de datos y enviando un código de respuesta**.** 5. El **Cliente** recibe un código de respuesta exitoso y muestra una ventana de texto emergente con el mensaje de “Estudiante registrado con éxito” con el botón de Ok (ver FA-5.1) 6. El **Estudiante o Administrador** da clic en “Ok” de la ventana emergente 7. El **Cliente** cierra la GUI-XX “Registrar Estudiante” y regresa a la GUI-XX “Login ” 8. Fin del caso de uso |
| **Flujos Alternos:** | FA – 2.1: Clic en Regresar   1. El **Cliente** cierra la ventana actual y regresa a la GUI-XX “Login ” 2. Termina el caso de uso   FA-3.1: Campos inválidos   1. El **Cliente** valida los campos ingresados, detecta los campos inválidos y muestra la ventana de diálogo notificando que existen campos inválidos 2. El **Estudiante o Administrador** da clic en el botón “Ok” 3. El **Cliente** cierra la ventana de diálogo. 4. Regresar al paso 3 del flujo normal.   FA-5.1: Código de respuesta fallido   1. El **Cliente** recibe un código de respuesta fallido, despliega un cuadro de dialogo avisando la razón por la que no se completó la solicitud con el botón de “Ok” 2. El **estudiante/administrador** da clic en “Ok” 3. Se regresa al paso 3 del flujo normal |
| **Excepciones:** | EX1. No hay conexión con el servidor   1. El **Cliente** muestra una ventana de diálogo notificando que no se puede conectar al servidor. 2. El **Estudiante o Administrador** da clic al botón “Ok” de la ventana de diálogo 3. El **Cliente** se regresa a la ventana “Login” 4. Termina caso de uso. |
| **Postcondiciones:** | POST-1: El ESTUDIANTE queda registrado en el sistema |
| **Reglas de negocio:** | Ninguna |
| **Incluye:** | Ninguno |
| **Extiende:** |  |

# ANEXO C: Diagrama de despliegue

Diagrama, Esquemático

Descripción generada automáticamente