# Introducción

## Propósito

Este documento se enfoca mejorar la comprensión del sistema “Vida Foránea” y contexto del producto para el equipo asignado al igual que a los stakeholders. En él, se encuentra información relacionada a los requisitos y el dominio del sistema; por mencionar algunos de los apartados se detallan: Alcance, referencias, contexto, requisitos funcionales y no funcionales, clases de usuario, prototipos de baja fidelidad, restricciones, entre otros aspectos que se añadirán en futuras entregas.

Es importante elaborar una especificación funcional, porque aquí es donde los diseñadores, desarrolladores obtienen la información para iniciar con el desarrollo del sistema, según las necesidades establecidas por el presente documento, asegurando la satisfacción de los interesados, y que exista una trazabilidad durante todas las etapas del ciclo de vida de un producto.

## Alcance

El sistema “Vida Foránea” tiene como finalidad conseguir que los estudiantes de la FEI consigan una orientación adecuada para la su entrada a la Universidad, ubicando los sitios cercanos a la Facultad relacionados con la comida, papelerías , lugares de ocio , descuentos de la UV; al igual que proporcionar un sitio donde ellos puedan comentar acerca de los lugares comentados anteriormente y expresarse sobre su carrera correspondiente en un foro , buscando mejorar la convivencia entre estudiantes y conocer una diversidad de opiniones.

El sistema incluirá un total de tres tipos de usuarios que serán: Estudiante, Moderador, Administrador. Hablando del primer tipo el poseerá la posibilidad del registro, consulta de nuevos lugares, agregación, eliminación y consulta de comentarios en el foro. El administrador es el único con la posibilidad de aceptar o descartar las solicitudes de nuevos lugares, eliminar y modificar un lugar ya existente. El moderador posee la posibilidad de manejar los comentarios del foro ya sea para eliminar los que no sean adecuados.

## Definiciones, acrónimos y abreviaciones

Tabla 1‑1 Acrónimos

|  |  |
| --- | --- |
| Acrónimo | Definición |
| FEI | Facultad de Estadística e Informática |
| UV | Universidad Veracruzana |
| SVF | Sistema “Vida Foránea” |

Tabla 1‑2 Abreviaciones

|  |  |
| --- | --- |
| Abreviación | Definición |
| CU-XX | Caso de uso |
| RQF-XX | Requisito funcional |
| RQNF-XX | Requisito no funcional |
| RES-XX | Restricción |
| SVF | Sistema de Vida Foranea |

## Referencias

# Requerimientos

## Contexto

Diagrama, Esquemático

Descripción generada automáticamente

Figura 2‑1 Diagrama de contexto

## Clases de usuario

Tabla 2‑1 Clases de usuario

|  |  |
| --- | --- |
| Clases de usuario | Descripción |
| Estudiante | El estudiante es un alumno matriculado a la Facultad de Estadística e informática. Tiene la posibilidad de consultar los lugares registrados en el sistema y registrar nuevos. También es capaz de agregar y eliminar comentarios en los foros y añadir opiniones a los lugares registrados. |
| Administrador | El administrador es parte de la comunidad educativa de la Facultad de Estadística e informática, es el encargado de registrar, modificar, agregar lugares, además de aprobar o rechazar las solicitudes de los estudiantes. El administrador puede eliminar comentarios de los estudiantes y generar reportes acerca de las puntuaciones de los lugares. |
| Moderador | El moderador es un alumno matriculado en la Facultad de Estadística e Informática, su función es revisar y eliminar los comentarios de los lugares y foro. |

## Casos de uso

### CU-01: Agregar lugar

El estudiante y el administrador tienen la posibilidad de crear una solicitud para agregar un lugar a la base de datos del sistema, incluyendo el nombre de lugar, su ubicación, foto del lugar para tener referencia de su ubicación, precio, servicios que ofrece y los horarios de atención.

### CU-02: Agregar comentario al foro

El estudiante tiene la posibilidad de agregar comentarios al foro de especifico de su carrera. Mientras que el administrador podrá agregar comentarios para todas las carreras. Los comentarios pueden incluir texto e imágenes.

### CU-03: Consulta de lugar

El administrador, el estudiante y el moderador pueden con consultar los lugares registrados hasta al momento. Con la posibilidad de añadir un comentario, una puntuación y eliminar comentarios. Los administradores y moderadores pueden eliminar los comentarios propios y ajenos, mientras que los estudiantes únicamente los propios.

### CU-04: Verificar solicitud de lugar

El administrador tiene la posibilidad de revisar las solicitudes ingresadas por los estudiantes para agregar un lugar; Puede aceptar o rechazar el registro del lugar.

### CU-05: Modificar lugar

El administrador tiene la posibilidad de modificar un lugar ya registrado en la base de datos.

### CU-06: Generar reporte

El administrador tiene la posibilidad de generar los reportes acerca de los lugares registrados, en el que se añadirá el nombre del lugar, su puntuación y cuantos comentarios han sido publicados.

### CU-07: Añadir opinión

El estudiante y el administrador pueden agregar una opinión al lugar que se encuentren consultando.

### CU-08: Consultar foro

El administrador, el estudiante y el moderador pueden consultar el foro, añadir y eliminar comentarios. Los administradores y moderadores pueden consultar los foros de todas las carreras y eliminar los comentarios propios y ajenos, mientras que los estudiantes solo pueden consultar el foro de la carrera a la que pertenecen y únicamente eliminar comentarios propios

### CU-09: Registrar estudiante

El estudiante puede registrarse a la aplicación usando de autenticación su matrícula junto con una contraseña que será cifrada en el servidor

## Prototipos

## 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.

## Restricciones

RES-01: La capa de datos de la API debe usar una base de datos relacional o NoSQL

RES-02: La aplicación de cliente debe ser de escritorio

RES-03: La capa de servicios de la API debe incluir servicio RESTful

RES-04: La capa de servicios de la API debe incluir servicio RPC

RES-05: La API debe estar implementada en un entorno de desarrollo GNU/Linux

RES-06: El lenguaje de programación para el cliente rico será C# y en el framework de .NET.

RES-07: El lenguaje de programación para la aplicación de servicios será Python

# Diseño

## Diseño arquitectónico

### Diagrama Descripción generada automáticamenteVista de casos de uso

### Texto Descripción generada automáticamente con confianza mediaVista lógica

### Vista de implementación

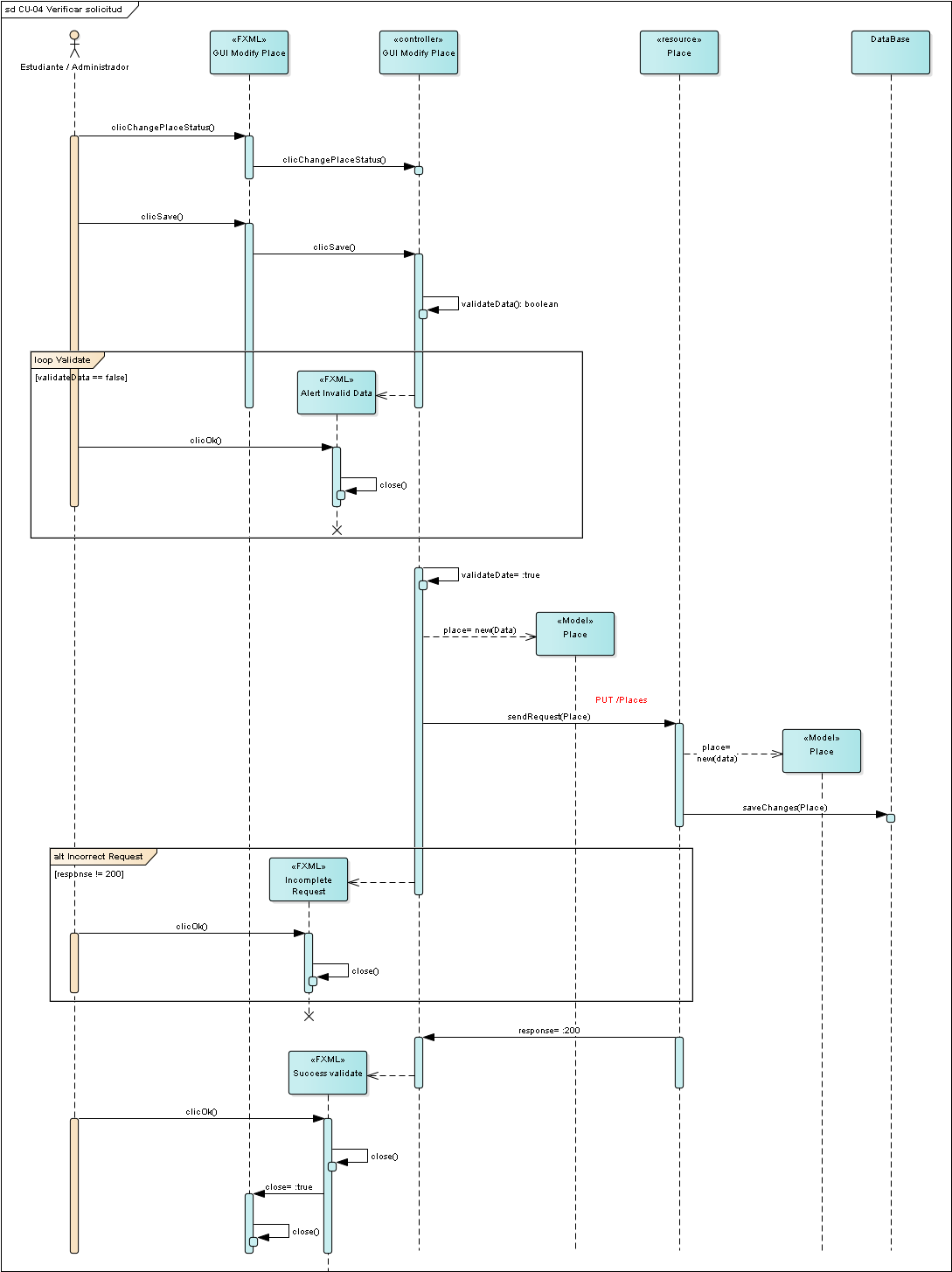
#### Diagrama Descripción generada automáticamenteCU-01 Agregar lugar

#### CU-03 Consultar lugar

Diagrama

Descripción generada automáticamente

#### CU-04 Verificar solicitud



### Vista de procesos

Diagrama, Esquemático

Descripción generada automáticamente

### Diagrama, Esquemático Descripción generada automáticamenteVista de despliegue

## Diagrama Descripción generada automáticamenteModelo de datos

## Descripciones de casos de uso

### 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 manda una petición a la **API** para recuperar el LUGAR seleccionado. (ver EX1). 2. La **API** recibe la petición y manda el LUGAR. 3. El **administrador** da clic en el botón “Aprobar” (ver FA-3.1) (ver FA-3.2) 4. El **Cliente** manda la solicitud de aprobar a la **API (**ver FA-4.1 y FA-4.2). 5. La **API** recibe los datos para la aprobación de un LUGAR actualizando el LUGAR y enviando un código de respuesta**.** 6. 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) 7. 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. La **API** recibe los datos para rechazar un LUGAR eliminado el LUGAR y enviando un código de respuesta**.** 3. El **Cliente** regresa a la ventana “Lista de solicitudes de lugar” 4. 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 **Servidor** para recuperar el LUGAR seleccionado junto con las OPINIÓN. (ver EX1). 2. La **API** recibe la petición y manda el LUGAR junto con sus OPINIÓN ligadas. 3. El **Cliente** 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). 4. El **Administrador** ingresa la información correspondiente de los campos y da clic en el botón “Guardar” (ver FA-2.1) 5. 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). 6. La **API** recibe los datos para la modificación de un lugar, actualizando en la base de datos y enviando un código de respuesta**.** 7. El **Cliente** recibe un código de respuesta exitoso, muestra la Opinión en la GUI-XX “Lugar” (ver FA-7.3) 8. El **Administrador** da clic en “Ok” de la ventana de diálogo 9. El **Cliente** cierra la ventana “Modificar lugar” y regresa a la ventana “Lista de lugares” 10. 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 a la **API** para recuperar la lista de lugares. (ver EX1). 2. La **API** recibe la petición y manda el LUGAR. 3. El **Cliente** muestra la GUI-XX “Reporte de lugares”. (ver EX1). 4. El **administrador** da clic en el botón “regresar”. 5. El **Cliente** cierra la GUI-XX “Reporte de lugares” y regresa a la GUI-XX “Listado de solicitudes de lugar” 6. 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:** |  |

# Construcción

## Selección justificada de pila de tecnología

### Aplicación de cliente rico

Se selecciono el lenguaje de programación C# con .NET Framework que es un entorno de ejecución administrado para Windows que proporciona diversos servicios a las aplicaciones en ejecución. Se decidió el uso de esta tecnología porque en la experiencia educativa de Tecnologías para la Construcción de Software se describieron sus principales ventajas que nos serán de apoyo para el desarrollo del cliente, las cuales son: Realización de un control automático del código enfocado en la seguridad junto con el manejo de los recursos del sistema, Interoperabilidad multilenguaje con cualquier lenguaje compatible con .NET ya que siempre se compilará en código intermedio, simplificación del despliegue de aplicaciones.

### Aplicación de servicios

Para la aplicación de servicios se seleccionó el lenguaje de programación Python, ajustándonos a la arquitectura de Rest, con el Framework de flask y utilizando SQLAlchemy para la base de datos. Se decidió el uso de esta tecnología debido a que el uso de flask es ideal para usuarios principiantes y funciona bien en conjunto con el lenguaje Python para realizar una API Rest.

Además, se hace uso de JSON Web Token (JWT) para la creación de tokens de acceso y asegurar los privilegios de los usuarios que accedan a la API, se decidió utilizarlo debido a que el formato es más sencillo a comparación de otras formas de codificación. Se utiliza el paquete de marshmallow para la serialización y deserialización en Python, permite validar los datos durante la deserialización.

## Estándares de codificación

PEP 257 – Docstring Conventions | peps.python.org. (s. f.). Python Enhancement Proposals. <https://peps.python.org/pep-0257/>

B. (2021a, octubre 12). C# Coding Conventions. Microsoft Docs. https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/coding-conventions

## Reportes de análisis estático de código

### Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamenteReporte del Cliente

### Interfaz de usuario gráfica, Aplicación Descripción generada automáticamenteReporte del Servidor

## Prácticas de construcción realizadas

### Convenciones de nombres

Se usará las convenciones UpperCamelCase en las funciones para C# mientras que LowelCamelCase para los métodos en Python

|  |  |
| --- | --- |
| void getId(Student student): | def get\_by\_id(cls, id): |

### Nombres consistentes

Las variables deben poseer nombres que permitan conocer la utilidad de variable

|  |  |
| --- | --- |
|  | place = Place(\*\*data) |

### Manejo de errores y excepciones

Es necesario especificar el error que se ha generado por la excepción y deben ser guardadas para una futura retroalimentación

|  |  |
| --- | --- |
|  | try:  data = place\_schema.load(data=json\_data)  except ValidationError as exc:  return {'message': "Validation errors", 'errors': exc.messages}, HTTPStatus.BAD\_REQUEST |

### Independencia funcional

Procura que las funciones hagan la menor cantidad de tareas posibles

|  |  |
| --- | --- |
|  | def delete(self, place\_id):  place = Place.get\_by\_id(place\_id)  if place is None:  return {'message': 'Lugar no encontrado'}, HTTPStatus.NOT\_FOUND  current\_user = get\_jwt\_identity()  current\_student = Student.get\_by\_enrollment(current\_user)  if current\_student.rol == 'estudiante' or current\_user=='moderador' :  response=jsonify({'message': 'Access is not allowed'})  response.status\_code=403  return response  place.delete()  response=jsonify({})  response.status\_code=204  return response |

### Usuarios autorizados

Para aumentar la seguridad utilicen la API con múltiples peticiones se implementa un token de acceso para solo permitir a usuarios autorizados

|  |
| --- |
| @jwt\_required()  def delete(self, place\_id): |

### Privilegios

Se poseen tres tipos de usuarios , por lo que para ciertas tareas se deberán especificar cuando uno de ellos tenga la autorización de realizar alguna de ellas en la API

|  |
| --- |
| @jwt\_required()  def delete(self, place\_id):  place = Place.get\_by\_id(place\_id)  if place is None:  return {'message': 'Lugar no encontrado'}, HTTPStatus.NOT\_FOUND  current\_user = get\_jwt\_identity()  current\_student = Student.get\_by\_enrollment(current\_user)  if current\_student.rol == 'estudiante' or current\_user=='moderador' :  response=jsonify({'message': 'Access is not allowed'})  response.status\_code=403  return response  place.delete()  response=jsonify({})  response.status\_code=204  return response |

### Minimizar el ámbito de las variables y accesibilidad de clases

# Pruebas

## Plan de pruebas para la Aplicación de servicios (API)

Las pruebas fueron realizadas en Postman con el propósito de automatizar dicho proceso , por lo tal se considero un plan de pruebas con los datos necesarios para las peticiones de http junto con una carpeta con las colecciones con las pruebas necesarias.

## Procedimiento de prueba

El procedimiento de pruebas se realizó a través de Postman que es una herramienta que ayuda en el proceso de desarrollo de APIS: Se utiliza principalmente para el testing y consumo de ella donde se puede monitorear, documentar , simular y escribir pruebas automatizadas.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente con confianza mediaPrimero se creó una nueva colección llamada Pruebas-VidaForanea donde se van a dividir las pruebas por paquete de cada uno de los métodos http

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteEn los paquetes de escriben las pruebas tanto para el caso de éxito junto con el de fracaso , se muestra un ejemplo con el paquete de pruebas GET

Postman nos da la posibilidad de correr las pruebas por paquete o de toda una colección en total, así que para seguir la estructuración de los casos de prueba se irán probando por cada uno de los métodos http.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamentePara correr un paquete o colección se debe dar clic en la carpeta principal y luego seleccionar el botón Run y así empezará a ejecutar el script de las pruebas.

## Casos de prueba

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Método HTTP | Modelo | Precondiciones | Datos de entrada | Resultado esperado |
| GET | Lugar | Exista al menos un lugar registrado | - | Código de respuesta 200 junto con el arreglo de lugares en formato JSON |
| Lugar | Exista al menos un lugar registrado | Id: 1 | Código de respuesta 200 junto con el lugar que coincida con la ID correspondiente |
| Estudiante | Exista al menos un estudiante registrado | - | Código de respuesta 200 junto con el arreglo de estudiantes en formato JSON |
| Estudiante | Exista al menos un estudiante registrado | Id: 1 | Código de respuesta 200 junto con el estudiante que coincida con la ID correspondiente |
| Foro | Exista al menos un foro registrado | - | Código de respuesta 200 junto con el arreglo de foros en formato JSON |
| Foro | Exista al menos un foro registrado | Id: 1 | Código de respuesta 200 junto con el foro que coincida con la ID correspondiente |
| Comentario | Exista al menos un comentario registrado | - | Código de respuesta 200 junto con el arreglo de comentarios en formato JSON |
| Comentario | Exista al menos un comentario registrado | Id: 1 | Código de respuesta 200 junto con el comentario que coincida con la ID correspondiente |
| Opinión | Exista al menos una opinión registrada | - | Código de respuesta 200 junto con el arreglo de opiniones en formato JSON |
| Opinión | Exista al menos una opinión registrada | Id: 1 | Código de respuesta 200 junto con la opinión que coincida con la ID correspondiente |
| POST | Lugar | - | {  "name": "ImprimePlus",  "address": "Av. Manuel Ávila Camacho 140, Centro, 91010",  "services":"Centro de copias",  "schedule":"Lunes 7:00-8:00, Martes 7:00-8:00",  "type\_place": "Herramientas",  "image": "00000010 01011010 00000111 01101001 01000010 10000101 10001000 00011111"  } | Código de respuesta 201 junto con el lugar creado |
| Lugar | - | {  "name": "Bola de Oro",  "address": "Av. Manuel Ávila Camacho 120, Centro, 91010",  "services":"Cafetería y restaurante",  "schedule":"Lunes 7:00-8:00, Martes 7:00-8:00",  "type\_place": "Comida",  "image": "00000010 01011010 00000111 01101001 01000010 10000101 10001000 00011111"  } | Código de respuesta 400 junto con el mensaje “Lugar ya registrado” |
| Estudiante | - | {  "name": "Edgar Valdes",  "enrollment": "s16013022",  "password":"12345",  "degree": "Redes",  "rol": "administrador"  } | Código de respuesta 201 junto con el estudiante creado |
| Estudiante | - | {  "name": "Karina Valdes Iglesias",  "enrollment": "s19014023",  "password":"Sanrio23Kuromi",  "degree": "Ingeniería de Software",  "rol": "administrador"  } | Código de respuesta 400 junto con el mensaje “Estudiante ya registrado” |
| Foro | - | {  "degree": "Estadística"  } | Código de respuesta 201 junto con el foro creado |
| Foro |  | {  "degree": "Redes"  } | Código de respuesta 400 junto con el mensaje “Foro ya registrado” |
| Opinión |  | {  "student":"s19014023",  "date":"12/06/2022",  "hour":"9:00",  "description":"Un gran lugar para relajarse",  "id\_place":1  } | Código de respuesta 201 junto con la opinión registrada |
| Opinión |  | {  "student":"s19014023",  "date":"13/06/2022",  "hour":"10:00",  "description":"Bonita cafeteria con platillos accesibles",  "score":5  "id\_place":1  } | Código de respuesta 400 junto con el mensaje “Opinión ya registrada” |
| Comentario |  | {  "student":"s19014023",  "date":"13/06/2022",  "hour":"9:00",  "description":"Organiza tus tiempos para conseguir un mejor control ",  "forum\_id":1  } | Código de respuesta 201 junto con el comentario registrado |
| Comentario |  | {  "student":"s19014022",  "date":"12/07/2022",  "hour":"9:10",  "description":"Organizar las tareas por prioridad de entrega",  "forum\_id":1  } | Código de respuesta 400 junto con el mensaje “Comentario ya registrado” |
| DELETE | Lugar |  | Id: 2 | Código de respuesta 204 |
| Comentario |  | Id:2 | Código de respuesta 204 |
| opinión |  | Id:2 | Código de respuesta 204 |
| PUT | Lugar | - | {  "Name": "Bola de Oro",  "address": "Av. Manuel Ávila Camacho 120, Centro, 91010",  "services":"Cafetería y restaurante",  "schedule":"Lunes 7:00-8:00, Martes 7:00-10:00",  "type\_place": "Comida",  "image": "00000010 01011010 00000111 01101001 01000010 10000101 10001000 00011111",  "status": "Aprobado"  } | Código de respuesta 200 junto con el lugar actualizado |

## Resultados

### Interfaz de usuario gráfica, Texto Descripción generada automáticamente con confianza mediaInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamenteGET

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

### Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamentePOST

### Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamentePUT

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza mediaInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

### Interfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamenteDELETE

# Estrategia de despliegue

|  |  |  |
| --- | --- | --- |
| Función | Herramienta | Aplicación |
| Despliegue continuo | Docker | Para mantener la heterogeneidad en la API RESTful se decidió utilizar el despliegue por contenedores los cuales ya poseen la configuración necesaria para que el servidor este en ejecución que proveen ventajas gracias a que son ligeros , pueden compartirse fácilmente al ser subidos en la nube de DockerHub , lo que permite la reutilización para otras aplicaciones similares. |
| Pipeline  Manifestación automatizada de procesos para llevar al software desde el control de versiones hasta los usuarios finales. | GitHub Actions | Fue implementada en la parte de la API RESTful donde se permite tener una integración y despliegues continuos ya que automatiza la construcción , pruebas y despliegue que cada vez que suceda un push , este pipeline se ejecute automáticamente. |
| Pruebas automatizadas | Postman | Con el fin de asegurarse que las peticiones HTTP devolvieran los resultados esperados se utilizo para probar tanto el caso normal como los alternos que fueron descritos en una sección anterior del presente documento ya que automatizar las pruebas permite agilizar esta etapa del desarrollo del servidor y aumentar la confiabilidad por el sistema. |
| Control de versiones | GitHub | Se mantuvo tanta documentación , artefactos y codigo almacenados en un mismo lugar para evitar el desfase de cambios entre los integrantes del equipo, asegurándose que cuando su cambio estuviera en funcionamiento hacer un pull request en el main, para poseer la versión más reciente y con el menor número de fallas en la rama principal. |

# Conclusiones