Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Estructuras de Datos

Catedráticos: Ing. René Ornélis, Ing. Carlos Alonzo, Ing. Luis Espino

Tutores Académico: José Morejón, Juan Lemus,Hugo Boiton, Fernando Hernandez

Práctica de laboratorio #1

Guatemala 11 de febrero de 2019

GLOSARIO

HTTP verbs:

Conjunto de métodos de petición (get, post, put, entre otros) en el cual se puede definir una acción al recurso determinado.

GET:

Este método solicita una representación de un set de datos.

POST:

Este método se utiliza para poder enviar un set de datos a un recurso específico, el cual genera un cambio dentro de la información.

DELETE:

Este método borra un set de datos específicos.

PUT:

Este método cambia un recurso ya existente, la entidad o set adjunta debería de considerar una versión modificada.

REST:

Se basa en un modelo cliente – servidor. REST es un conjunto de principios que describen cómo se pueden usar los estándares para desarrollar aplicaciones web.

BACK-END:

Cuando el desarrollo se encuentra del lado del servidor aplicado a tecnologías o herramientas que estas interactúan con capas internas existentes.

FRONT-END:

Son todas aquellas herramientas disponibles para la ejecución del lado del cliente, por ejemplo la ejecución de una aplicación que se muestra del lado del cliente en el navegador web.

ENDPOINT

URLs de un servicio web que responden a una petición para poder cargar o consumir información.

RESTful

Referencia a un web service que implementa una arquitectura REST

CORS

Intercambio de Recurso de Origen Cruzado

Objetivos

Generales

- Que el estudiante aplique los conceptos vistos en clase respecto a estructuras dinámicas de datos lineales.
- Que el estudiante aprenda y aplique los conceptos de punteros y memoria dinámica en la solución de problemas.
- Que el estudiante pueda integrar estructuras de datos para la construcción de endpoints basados en framewors RESTful con el lenguaje c++.

Específicos

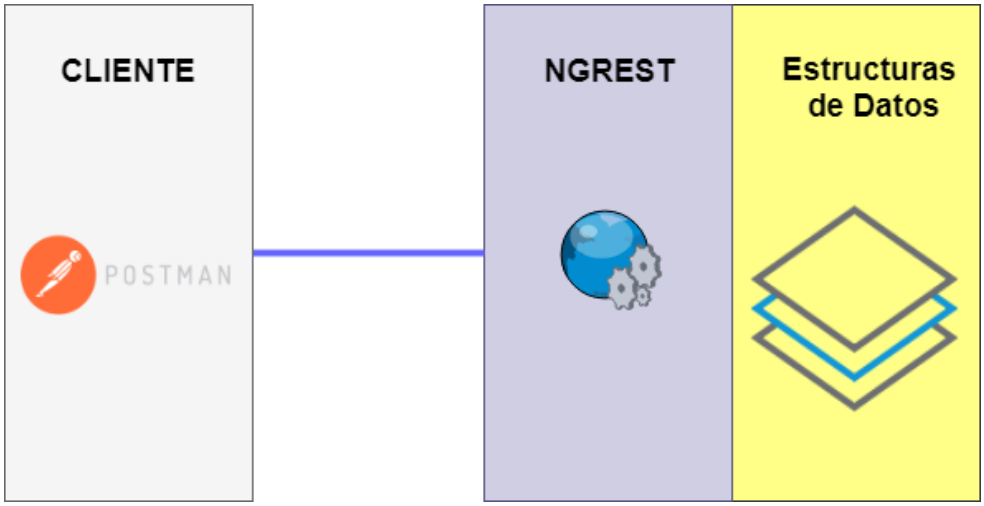
- Que el estudiante identifique e integre las estructuras de datos que puedan dar solución al problema de manera efectiva.
- Que el estudiante logre diferenciar y aplicar el paso de variables por valor y referencia en el lenguaje C/C++.
- Que el estudiante pueda definir tipos (typedef) y estructuras(struct) en el lenguaje de C.
- Que el estudiante pueda crear estructuras de datos como servicios web.

Descripción

Las estructuras de datos aplicadas como servicios web, es una etapa para que el estudiante pueda conocer los términos básicos basados en este tipo de arquitectura de software, con el agregado de poder integrar listas, pilas y colas como un servicio. De tal manera esta estrategia de aprendizaje incluye:

- Crear un entorno de desarrollo cliente - servidor
- Poder entender cómo integrar un servidor de aplicaciones
- Interactuar con el servidor y un cliente por peticiones http
- Utilizar herramientas que ayuden al desarrollo ágil

Definición de comunicación entre tecnologías



- El software POSTMAN se utilizara como cliente para las pruebas de los endpoints.
- NGREST es el framework que se utilizara para crear los endpoints.
- Estructuras de datos, es donde se desarrollará la lógica para poder disponibilizar en servicios.

Estructuras de Datos

Lista circular

Se debe desarrollar una lista simple ordenada por el nombre del usuario.

Datos que debe almacenar:

- Identificador (Alfanumérico)
- Nombre(texto)
- Correo(texto)
- Fecha de ingreso(date)

Pila de servicio de un enlace

Se debe desarrollar una pila con los siguientes datos

- Identificador sesión (Alfanumério)
- Identificador de usuario(Alfanumérico)

Cola circular de recursos

Se debe de desarrollar una cola de recursos con los siguientes datos

- dominio
- Contenido

NGREST:Endpoints

URL: /newuser

Desarrollar un servicio POST para la inserción de un nuevo usuario con el siguiente JSON

```
{
	“user”:{
		“id”:”AX100”,
		“nombre”:”María Debe”,
		“correo”:”uno@uno.com”,
		“fecha”:”01/12/18”
	}
}
```

Al crear los usuarios deben de asignarles el identificador, formándolo de la siguiente manera:
<texto>.dominio1

El texto indica cualquier cadena de texto y agregarle el dominio. El listado de dominios se debe de asignar con la estructura cola, la cual debe recorrer de manera circular asignando los dominios a los usuarios creados. Por ejemplo, si el primer dominio de la cola se encuentra

- ingenieria.usac.edu.gt

Debe de asignarlo al primer usuario creado, ejemplo:
AX100.ingenieria.usac.edu.gt

URL: /newrecurso

Desarrollar un servicio POST para la inserción de un nuevo recurso con el siguiente JSON

```
{
	“recurso”:{
		“dominio”:”dominio1.com”,
		“contenido”:”  <!DOCTYPE html>
				<html>
				  <head>
				    <title>Example</title>
				  </head>
				  <body>
				    <p>This is an example of a simple HTML page with one
			paragraph.</p>
				  </body>
				</html>”
	}
}
```

Al crear un nuevo recurso debe de almacenarlo en la cola circular de recursos.

URL: /getrecurso/id_usuario}

Desarrolla un servicio GET enviando el identificador del usuario. Al ingresar esta información, se debe de proceder a insertar la información (identificador de sesión, identificador de usuario) a la pila de servicio si y solo si el usuario es válido y la cola de recursos no está vacía.

Como respuesta debe de retornar el siguiente JSON

```
{
	“recurso”:{
		“id_sesion”:”crt100”,
		“usuario”:”María Debe”,
		“dominio”:”ingenieria.usac.edu.gt”,
		“contenido”: ”  <!DOCTYPE html>
				<html>
				  <head>
				    <title>Example</title>
				  </head>
				  <body>
				    <p>This is an example of a simple HTML page with one
			paragraph.</p>
				  </body>
				</html>”

	}
}
```

Observación: Para asignarle contenido al usuario deben de buscar la información según al dominio que pertenece el usuario en la cola de servicios.

URL: /getusuarios

Debe de retornar un array de los usuarios de forma ordenada.

URL: /getcolaservicios

Debe de retornar un array de los servicios encolados.

URL: /getpila

Debe de retornar toda la información almacenada en la pila con la observación de mostrar el nombre del usuario y no el identificador, al solicitar este servicio devuelve la información y como paso siguiente debe de vaciar la pila.

RECURSOS

Ngrest wiki

https://github.com/loentar/ngrest/wiki

Postman

https://www.getpostman.com/

Tutorial de cómo implementar una lista en ngrest

https://www.dropbox.com/s/8jl5smlgq674bik/Manual%20de%20ngrest.pdf?dl=0

Restricciones

- Todas las estructuras deben de ser realizadas por el estudiante. No se permite el uso de alguna estructura por librerías externas y el uso de vectores, de manera parcial o total dentro de la práctica.

Fecha y modo de entrega

- Lunes 18 de febrero del 2019 antes de las 11:59 (No se aceptarán practicas después de esta fecha y hora)
- Se habilitara un formulario para enviar el código antes de la fecha límite. Con el siguiente nombre, usted debe de enviar su práctica:
  <carné>.zip
  Ejemplo: 201600000.zip
  - Si no lo envía con este formato, el formulario no podrá almacenar su código.

  Solo se permite una carga.