Universidad Latina De Costa Rica

Facultad de Ingenierías, Tecnologías de la Información y Comunicación

BIT 28 - Sistemas Operativos II

Proyecto Final

Portal de Tutorías y Soporte Académico

Elaborado Por:
Melissa Bermúdez Araya

Docente:
Carlos Andres Mendez Rodriguez

Fecha de Entrega:
Viernes 28 de Marzo de 2025

Tabla de Contenidos

Resumen

El presente artículo propone el desarrollo y creación de un portal de tutorías y ayuda escolar para la comunidad de estudiantes de la Universidad Latina; esta herramienta, con una infraestructura tecnológica que asegura alta disponibilidad y buen rendimiento mediante el uso de balanceo de carga con HAProxy. La solución está enfocada a responder a una demanda creciente de asistencia académica, y busca garantizar que siga funcionando sin problemas, incluso ante grandes volúmenes de tráfico o fallos en los servidores. Se añadirán métodos para distribuir como Round-robin y Least Connections, formas para comprobar el estado (chequeos de salud) como también protecciones contra ataques grandes (DDoS) y un límite en el número de conexiones.

Introducción

El ambiente educativo hoy pide soluciones tecnológicas que den el acceso rápido a los materiales escolares y la estabilidad sin parar de los servicios. En este lugar, las clases unipersonales y ayuda de estudio en internet se han hecho partes claves para mejorar el resultado de los alumnos y bajar la deserción. Pero, la mucha gente que usa estas páginas puede poner en juego su operación si no hay una base capaz de crecer duro.

Este trabajo trae la idea de crear un sitio web que dejará a los alumnos organizar ayuda extra, ver materiales para aprender y hablar de inmediato con los tutores. Para estar seguros de su buen funcionamiento y rapidez, se plantea usar HAProxy como herramienta principal para equilibrar eltrabajo, junto con métodos simples para evitar ataques y sobrecargar la plataforma.

Justificación del Proyecto

Durante los periodos académicos, especialmente en tiempos de exámenes o entrega de trabajos finales, se incrementa la demanda de tutorías.

Un sistema caído o lento impacta negativamente en la experiencia del estudiante y puede dificultar el acceso a recursos clave.

Mediante la incorporación de un balanceador de carga como HAProxy, se busca garantizar un servicio disponible, distribuido y protegido contra fallas y ataques básicos, permitiendo así una experiencia confiable y continua.

Objetivos

Objetivo General

- Diseñar e implementar un portal web de tutorías y soporte académico para la Universidad Latina, asegurando alta disponibilidad y rendimiento eficiente.

Objetivos Específicos

- Desarrollar un portal web funcional que permita la gestión de tutorías académicas, incluyendo reservas, chat y acceso a recursos.
- Evaluar el rendimiento, disponibilidad y seguridad del sistema mediante pruebas controladas de carga y recuperación ante fallos.
- Documentar la arquitectura y buenas prácticas para su implementación futura en el entorno institucional.

Script del Servicio Web

```
from flask import Flask, request, jsonify
import sqlite3

app = Flask(__name__)
DATABASE = 'tutorias.db'

def init_db():
    with sqlite3.connect(DATABASE) as conn:
        c = conn.cursor()
        c.execute('''CREATE TABLE IF NOT EXISTS estudiantes (
                     id INTEGER PRIMARY KEY AUTOINCREMENT,
                     nombre TEXT NOT NULL,
                     correo TEXT NOT NULL)''')
        c.execute('''CREATE TABLE IF NOT EXISTS tutores (
                     id INTEGER PRIMARY KEY AUTOINCREMENT,
                     nombre TEXT NOT NULL,
                     especialidad TEXT NOT NULL)''')
        c.execute('''CREATE TABLE IF NOT EXISTS tutorias (
                     id INTEGER PRIMARY KEY AUTOINCREMENT,
                     estudiante_id INTEGER,
                     tutor_id INTEGER,
                     fecha TEXT,
                     hora TEXT,
                     FOREIGN KEY(estudiante_id) REFERENCES estudiantes(id),
                     FOREIGN KEY(tutor_id) REFERENCES tutores(id))''')
        conn.commit()

@app.route('/estudiantes', methods=['POST'])
def registrar_estudiante():
    data = request.get_json()
    nombre = data['nombre']
    correo = data['correo']
    with sqlite3.connect(DATABASE) as conn:
        cursor = conn.cursor()
            cursor.execute("INSERT INTO estudiantes (nombre, correo) VALUES (?, ?)",
(nombre, correo))
        conn.commit()
    return jsonify({'mensaje': 'Estudiante registrado'}), 201
```

```
@app.route('/tutores', methods=['POST'])
def registrar_tutor():
    data = request.get_json()
    nombre = data['nombre']
    especialidad = data['especialidad']
    with sqlite3.connect(DATABASE) as conn:
        cursor = conn.cursor()
            cursor.execute("INSERT INTO tutores (nombre, especialidad) VALUES (?, ?)",
(nombre, especialidad))
        conn.commit()
    return jsonify({'mensaje': 'Tutor registrado'}), 201

@app.route('/tutorias', methods=['POST'])
def agendar_tutoria():
    data = request.get_json()
    estudiante_id = data['estudiante_id']
    tutor_id = data['tutor_id']
    fecha = data['fecha']
    hora = data['hora']
    with sqlite3.connect(DATABASE) as conn:
        cursor = conn.cursor()
              cursor.execute("INSERT INTO tutorias (estudiante_id, tutor_id, fecha, hora)
VALUES (?, ?, ?, ?)",
                     (estudiante_id, tutor_id, fecha, hora))
        conn.commit()
    return jsonify({'mensaje': 'Tutoría agendada'}), 201

@app.route('/tutorias', methods=['GET'])
def obtener_tutorias():
    with sqlite3.connect(DATABASE) as conn:
        cursor = conn.cursor()
        cursor.execute('''SELECT t.id, e.nombre, tu.nombre, tu.especialidad, t.fecha, t.hora
                       FROM tutorias t
                       JOIN estudiantes e ON t.estudiante_id = e.id
                       JOIN tutores tu ON t.tutor_id = tu.id''')
        resultados = cursor.fetchall()
        tutorias = []
        for fila in resultados:
            tutorias.append({
```

```
                'id': fila[0],
                'estudiante': fila[1],
                'tutor': fila[2],
                'especialidad': fila[3],
                'fecha': fila[4],
                'hora': fila[5]
            })
    return jsonify(tutorias)

if __name__ == '__main__':
    init_db()
    app.run(host='0.0.0.0', port=5000)
```