LAPORAN TUGAS BESAR

Implementasi Sistem Distribusi

Studi Kasus Antrean Registrasi Medis

Dipersiapkan oleh:

Kelas: IF-45-01

Kelompok Netflix

Muhammad Naufal Hawari	(1301213069)
Sayid Rayhan Mulachela	(1301213355)
Aldi Muhammad Farhan	(1301213053)
Dani Abizar Ahmad	(1301213128)
Adhitama Wichaksono	(1301210201)

Fakultas Informatika

Universitas Telkom

2023

DAFTAR ISI

BAB I.
PENDAHULUAN

1.1 Deskripsi Tugas Besar

Tugas Besar di kerjakan oleh kelompok kami yang berjumlah 5 orang . Tugas Besar yang kami pilih ialah “Antrean Registrasi Medis” . Tugas Besar ini bertujuan untuk membantu pasien agar tidak harus menunggu lama di rumah sakit saat antre ke klinik tertentu di rumah sakit. Pada Tugas besar ini Client dapat melakukan registrasi (nomor rekam medis, nama, dan tanggal lahir) ke klinik tertentu di rumah sakit, dan mendapatkan nomor antrean. Server akan mengirimkan informasi ke client berupa data antrean saat ini dan perkiraan waktu kapan antrean client mendapatkan giliran. Setelah itu, Client dapat melihat daftar klinik yang buka serta memilih salah satu klinik.

1.2 Peran Anggota Kelompok

Tugas Besar ini kami kerjakan dengan membagi tugas kepada masing-masing anggota kelompok yang ditunjukkan pada tabel dibawah ini.

Tabel Peran Anggota Kelompok

NIM	NAMA	TUGAS
1301213069	Muhammad Naufal H.	1. Pembuatan code client 2. Pembuatan code server
1301213355	Sayid Rayhan M.	1. Pembuatan code server 2. Menyusun laporan
1301213053	Aldi Muhammad F.	1. Pembuatan code server 2. Menyusun laporan
1301213128	Dani Abizar Ahmad	1. Pembuatan code client 2. Menyusun laporan
1301210201	Adhitama Wichaksono	1. Pembuatan code client 2. Menyusun laporan

BAB II.
ANALISIS

1.3 Alasan Pemilihan Solusi

Pemilihan solusi sistem Remote Procedure Call (RPC) untuk Antrean Registrasi Medis di rumah sakit didasarkan pada kebutuhan efisiensi, keamanan, dan skalabilitas. Pertama, RPC memberikan kemudahan dalam komunikasi antara client dan server dengan menyederhanakan pemanggilan fungsi atau prosedur di server melalui mekanisme pemanggilan jarak jauh. Dalam konteks Antrean Registrasi Medis, hal ini memungkinkan pasien untuk melakukan registrasi ke klinik tertentu dan mendapatkan nomor antrean tanpa harus menunggu lama.

Kemudian keamanan data medis menjadi prioritas utama dalam sistem ini. Implementasi RPC dapat diintegrasikan dengan protokol keamanan seperti HTTPS, yang menyediakan enkripsi data dan proteksi terhadap ancaman keamanan. Dengan demikian, pemilihan RPC memperhitungkan perlunya menjaga kerahasiaan dan integritas informasi medis pasien yang dikirimkan antara client dan server, memastikan bahwa data sensitif tetap aman.

Selain itu, skalabilitas sistem adalah aspek kritis yang dipertimbangkan. RPC memungkinkan perluasan sistem secara horizontal, memudahkan penambahan sumber daya atau server saat jumlah pengguna atau klinik meningkat. Kemampuan sistem untuk menangani pertumbuhan ini penting dalam mengoptimalkan pelayanan antrean pasien di rumah sakit, memastikan responsivitas sistem yang baik tanpa mengorbankan kualitas layanan. Dengan demikian, pemilihan RPC sebagai solusi untuk Antrean Registrasi Medis menggabungkan aspek-aspek tersebut untuk menciptakan sistem yang efisien, aman, dan dapat berkembang sesuai dengan kebutuhan rumah sakit.

1.4 Model Sistem

Model sistem RPC kami terdiri dari 2 proses, yaitu proses registrasi antrian dan proses cek status antrian. Proses registrasi antrian dimulai dengan antarmuka pengguna yang menyediakan daftar rumah sakit yang tersedia. Pengguna dapat memilih rumah sakit tertentu dengan melihat list yang disediakan. Setelah memilih rumah sakit, pengguna diminta untuk memasukkan kode rumah sakit yang valid. Setelah verifikasi kode rumah sakit, pengguna dapat menginputkan data pasien yang diperlukan untuk proses registrasi, seperti nomor rekam medis, nama, dan tanggal lahir. Setelah semua informasi terisi, pengguna dapat mengirimkan data registrasi ke server menggunakan Remote Procedure Call (RPC). Server akan mengelola proses registrasi, menetapkan kode antrean berdasarkan antrian saat ini, dan memberikan respons kepada pengguna berupa tiket antrian yang mencakup kode antrean dan perkiraan waktu pelayanan.

Pada proses cek status antrean, pengguna dapat untuk menginputkan kode antrian yang diterima saat proses registrasi. Melalui RPC, input kode antrian akan dikirimkan ke server untuk verifikasi. Server akan mengambil informasi terkait kode antrian dari basis data antrean dan memberikan respons kepada pengguna. Respons tersebut berisi informasi mengenai perkiraan waktu ketika pasien akan dipanggil. Dengan ini, pengguna dapat memantau status antrian mereka secara real-time dan membuat keputusan yang lebih baik terkait waktu kedatangan mereka ke rumah sakit.

Keseluruhan model sistem ini didesain untuk memberikan pengalaman registrasi antrian yang mudah dan efisien bagi pengguna. Dengan integrasi RPC, proses registrasi dan cek status antrian dapat dilakukan secara cepat dan akurat, memberikan kejelasan dan keteraturan dalam manajemen antrean di rumah sakit.

Setiap 10 detik server akan otomatis melakukan *dequeue*, yaitu mengurangi waktu antrian tiap kode antrian di seluruh klinik. Sistem *dequeue* ini dilakukan secara paralel dengan sistem untuk registrasi antrean medis menggunakan *threading*.

BAB III.
PERANCANGAN

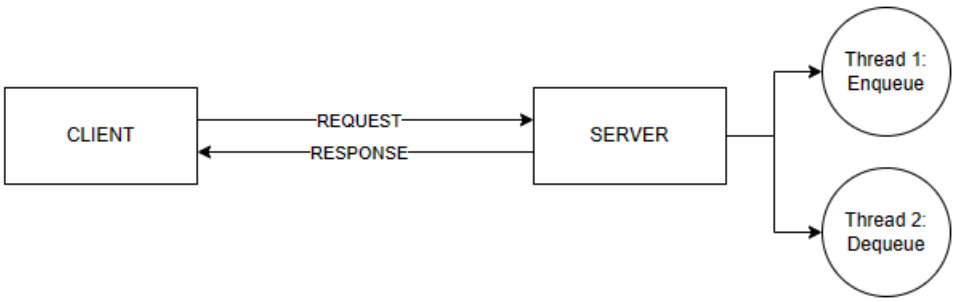
1.5 Arsitektur Sistem dan Jaringan

Sistem dan jaringan yang digunakan berbasis client-server dan penggambaran hubungan jaringan yang digunakan menggunakan topologi star. Topologi jaringan berbentuk star adalah jaringan dari beberapa komputer yang memiliki koneksi dengan node yang berada di tengah sistem jaringan.

- *Node* yang menghubungkan tiap komputer, atau dapat disebut Client, adalah IP Address dan digunakan sebagai perantara untuk mengakses *Server*.
- *Server* menyimpan data antrean dalam sebuah *dictionary*. Selain itu, terdapat Admin yang mengatur dan mengawasi *server* secara langsung.

Server juga menggunakan pemrograman paralel menggunakan *threading*. Threading ini dilakukan untuk melayani registrasi antrean medis sekaligus melayani proses dequeue (proses mengurangu waktu antrian di seluruh klinik). Visualisasi arsitektur sistem dapat dilihat di Gambar 1.

Gambar 1. Arsitektur Sistem Aplikasi Antrean Registrasi Medis



1.6 Alur Proses Aplikasi

Alur proses aplikasi dapat dirinci sebagai berikut:

1. User melakukan registrasi dengan memilih klinik yang sedang buka, jika klinik sedang tutup, proses registrasi akan gagal.
2. User dapat memeriksa lama waktu antrian terhadap registrasi yang sudah dilakukan pada menu “9. Lihat semua status antrian”.

BAB IV.
IMPLEMENTASI

1.7 Screencapture Program

→Kode Server

```python
from xmlrpc.server import SimpleXMLRPCServer  # Mengimpor SimpleXMLRPCServer dari modul xmlrpc.server
import threading  # Mengimpor modul threading untuk threading
import time  # Mengimpor modul time untuk manipulasi waktu

QUEUE_TIME = 10  # Konstanta untuk waktu antrian

class ClinicServer:  # Mendefinisikan kelas ClinicServer
    def __init__(self):  # Fungsi konstruktor untuk kelas ClinicServer
        # Mendefinisikan status klinik dengan beberapa data klinik
        self.klinik_status = {
            1: {"name": "Klinik A", "status": "Buka", "queue_wait_time": [], "queue_patients": []},
            2: {"name": "Klinik B", "status": "Tutup", "queue_wait_time": [], "queue_patients": []},
            3: {"name": "Klinik C", "status": "Buka", "queue_wait_time": [], "queue_patients": []},
        }

        self.thread_dequeue = threading.Thread(target=self.dequeue)  # Membuat thread baru untuk fungsi dequeue

    def dequeue(self):  # Mendefinisikan fungsi dequeue
        while True:  # Loop tak terbatas
            time.sleep(QUEUE_TIME)  # Tidur selama waktu antrian
            print("UPDATING QUEUES...")  # Mencetak pesan
            for clinic_id in self.klinik_status.keys():  # Iterasi melalui setiap klinik
                # Mengupdate waktu tunggu di setiap klinik
                for i in range(len(self.klinik_status[clinic_id]["queue_wait_time"])):
                    self.klinik_status[clinic_id]["queue_wait_time"][i] = max([0, self.klinik_status[clinic_id]["queue_wait_time"][i] - QUEUE_TIME])
                # Mencetak status waktu tunggu terakhir
                print(f'{self.klinik_status[clinic_id]["name"]}: last queue waiting time = {max([0] + self.klinik_status[clinic_id]["queue_wait_time"])}')

    def register_patient(self, clinic_id, patient_info):  # Fungsi untuk mendaftarkan pasien
        if (self.klinik_status[clinic_id]["status"] == "Tutup"):  # Jika klinik tutup
            return "none"  # Mengembalikan "none"

        # Menambahkan pasien ke antrian
        self.klinik_status[clinic_id]["queue_patients"].append(patient_info)

        # Menghitung dan memperbarui waktu tunggu antrian
        last_queue_wait_time = max(self.klinik_status[clinic_id]["queue_wait_time"] + [0])
        self.klinik_status[clinic_id]["queue_wait_time"].append(last_queue_wait_time + QUEUE_TIME)

        return f"{clinic_id}{len(self.klinik_status[clinic_id]['queue_patients']) - 1}"  # Mengembalikan ID antrian
```

```python
    def get_queue_status(self, queue_id):  # Fungsi untuk mendapatkan status antrian
        clinic_id = int(queue_id[0])  # Mengambil ID klinik dari ID antrian
        queue_wait_time_index = int(queue_id[1:])  # Mengambil index waktu tunggu dari ID antrian
        estimation_wait_time = self.klinik_status[clinic_id]["queue_wait_time"][queue_wait_time_index]  # Menghitung estimasi waktu tunggu
        return f"Estimasi waktu tunggu kode antrian {queue_id}: {estimation_wait_time}"  # Mengembalikan estimasi waktu tunggu

    def get_menu(self):  # Fungsi untuk mendapatkan menu
        menu_text = "Selamat Datang di Antredis\n"
        menu_text += "No.\tNama\t\tStatus\tEstimasi Waktu Tunggu\n"
        for id in self.klinik_status.keys():  # Iterasi melalui setiap klinik
            # Menambahkan informasi klinik ke teks menu
            menu_text += f"{id}.\t{self.klinik_status[id]['name']}\t{self.klinik_status[id]['status']}\t{max([0] + self.klinik_status[id]['queue_wait_time'])}\n"

        menu_text += "9. Lihat semua status antrian\n"
        menu_text += "0. Keluar"
        return menu_text  # Mengembalikan teks menu

def run_server():  # Fungsi untuk menjalankan server
    server = SimpleXMLRPCServer(("0.0.0.0", 8000))  # Membuat server XMLRPC di alamat tertentu
    clinic_server = ClinicServer()  # Membuat instance dari ClinicServer
    server.register_instance(clinic_server)  # Mendaftarkan instance ClinicServer ke server
    print("Server listening on port 8000...")  # Mencetak pesan

    clinic_server.thread_dequeue.start()  # Memulai thread dequeue
    server.serve_forever()  # Menjalankan server selamanya
    clinic_server.thread_dequeue.join()  # Menunggu thread dequeue selesai

if __name__ == "__main__":  # Jika file dijalankan sebagai skrip utama
    run_server()  # Menjalankan fungsi run_server
```

→Kode Client

```python
import xmlrpc.client  # Mengimpor modul xmlrpc.client

queue_ids = []  # Mendefinisikan variabel global untuk menyimpan ID antrian

class ClinicClient:  # Mendefinisikan kelas ClinicClient
    def __init__(self, server_address):  # Fungsi konstruktor untuk kelas ClinicClient
        # Menghubungkan ke server menggunakan alamat server
        self.server = xmlrpc.client.ServerProxy(server_address)

    def register_patient(self, clinic_id, patient_info):  # Fungsi untuk mendaftarkan pasien
        queue_id = self.server.register_patient(clinic_id, patient_info)  # Memanggil fungsi register_patient pada server
        if queue_id == "none":  # Jika pendaftaran gagal
            print(f"Pendaftaran gagal. Klinik sedang tutup")
            return None
        else:
            queue_ids.append(queue_id)  # Menambahkan ID antrian ke daftar
            print(f"Pendaftaran berhasil. Kode antrian: {queue_id}")
            return queue_id

    def get_queue_status(self, clinic_id):  # Fungsi untuk mendapatkan status antrian
        status = self.server.get_queue_status(clinic_id)  # Memanggil fungsi get_queue_status pada server
        print(f"{status}")  # Mencetak status antrian

    def get_menu(self):  # Fungsi untuk mendapatkan menu
        print(self.server.get_menu())  # Memanggil fungsi get_menu pada server dan mencetak hasilnya
```

```python
if __name__ == "__main__":  # Jika file dijalankan sebagai skrip utama
    server_address = "http://127.0.0.1:8000"  # Mendefinisikan alamat server
    client = ClinicClient(server_address)  # Membuat instance dari ClinicClient

    client.get_menu()  # Memanggil fungsi untuk menampilkan menu

    pilihan_menu = int(input("Pilihan menu: "))  # Meminta input dari pengguna untuk pilihan menu
    while (pilihan_menu ⩽ 3 and pilihan_menu ⩾ 1) or pilihan_menu == 9:  # Loop selama pilihan menu valid

        if pilihan_menu ≠ 9:  # Jika pilihan bukan 9
            name = input("Nama Lengkap: ")  # Meminta input nama lengkap
            dob = input("Tanggal Lahir (contoh: 30-01-2003): ")  # Meminta input tanggal lahir

            patient_info = {"name": name, "dob": dob}  # Membuat dictionary informasi pasien
            queue_id = client.register_patient(pilihan_menu, patient_info)  # Mendaftarkan pasien
            if queue_id ≠ None:  # Jika pendaftaran berhasil
                client.get_queue_status(queue_id)  # Mendapatkan status antrian
        elif pilihan_menu == 9:  # Jika pilihan adalah 9
            if len(queue_ids) > 0:  # Jika ada ID antrian

                for queue_id in queue_ids:  # Iterasi melalui setiap ID antrian
                    client.get_queue_status(queue_id)  # Mendapatkan status untuk setiap antrian
            else:
                print("Anda belum melakukan registrasi")  # Jika belum ada pendaftaran
        client.get_menu()  # Menampilkan menu lagi
        pilihan_menu = int(input("Pilihan menu: "))  # Meminta input pilihan menu lagi
```

1.8 Keterbatasan

Keterbatasan pada aplikasi yang kami buat belum terhubung dengan database, sehingga ketika server dimatikan, data antrean akan langsung terhapus. Maka dari itu aplikasi atau program kami harus terhubung dengan server untuk bisa berjalan. Pada aplikasi yang kami

buat hanya dapat menggunakan server address dari perangkat yang menjalankan program server pada aplikasi kami, jika tidak seperti itu akan muncul eror pada client.

1.9 Demo Program

Berikut adalah video link demo program antrean registrasi medis:

https://telkomuniversityofficial-my.sharepoint.com/:v:/g/personal/aldimuhfar_student_telkomuniversity_ac_id/EfiY_cuwcIhEiv29AfhvR8EBfLRFGayJnGZx1lV4L_WAzg?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJPbmVEcml2ZUZvckJ1c2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=Sef2nZ