Project Proposal: Healthcare Appointment Booking System

1. Project Description

For this project, I propose to develop and deploy a Healthcare Appointment Booking System on Kubernetes, designed to allow patients to schedule appointments with healthcare providers efficiently. The application domain is healthcare management, a critical area where reliable, scalable, and secure systems can significantly improve patient experience and operational efficiency. The system will enable patients to view available doctor schedules, book appointments, and receive confirmations, while administrators can manage schedules and patient data. This project interests me because it addresses real-world challenges in healthcare delivery, such as handling peak booking demands and ensuring data privacy, which align well with cloud automation technologies.

To facilitate completion, I plan to leverage open-source tools such as a lightweight web framework (e.g., Flask or Django) for the frontend and backend, paired with a database like PostgreSQL for persistent storage. I may also explore existing healthcare-related APIs or libraries to streamline development, ensuring any custom components require only minor adjustments rather than extensive new coding. A significant challenge will be implementing automatic scaling to manage fluctuating demand—such as morning rushes—while maintaining system reliability and security for sensitive patient data. Another challenge is ensuring seamless integration of stateful components, like the appointment scheduler, with Kubernetes.

For testing and demonstration, I propose deploying the system initially on a local Kubernetes cluster using Minikube, which offers a low-cost, controlled environment to refine the setup. For the final demonstration, I aim to deploy it on a public cloud platform like Google Kubernetes Engine (GKE), which provides robust scalability and monitoring features suitable for showcasing advanced Kubernetes capabilities. This project's complexity—balancing scalability, stateful management, and security—makes it an ideal candidate to demonstrate the unit's concepts at a high level.

2. Demonstration of Unit Learning Outcomes (ULOs)

This project provides a strong platform to demonstrate the five ULOs of S1T226 at an advanced level, with the final presentation highlighting these achievements. Below, I outline my initial expectations for each:

- **ULO1: Explain how computing resources are represented within cloud systems and how they are managed within both public and private cloud systems to a**

range of audiences.
The project will involve managing resources like pods and nodes in Kubernetes, such as scaling them to handle appointment booking loads. I expect to demonstrate this by explaining resource allocation in my presentation, contrasting local (Minikube) and public (GKE) deployments to show advanced understanding of cloud resource management.

- **ULO2: Install and configure cloud orchestration technologies and underlying systems for deploying, monitoring, and managing network services and applications.**
  Configuring Kubernetes to deploy the booking system—using features like StatefulSets for the scheduler and monitoring tools for performance—will be central. I'll showcase this through a live demo of the deployed system, highlighting my ability to set up and manage a complex application at an advanced level.

- **ULO3: Evaluate the application of public and/or private cloud services for the deployment of network services and applications, including consideration of business impacts.**
  Choosing between Minikube and GKE involves evaluating cost, scalability, and reliability trade-offs. I'll demonstrate this by discussing my deployment choice in the presentation, linking it to business impacts like cost-efficiency and patient satisfaction, showing advanced decision-making skills.

- **ULO4: Collaborate with software architects, developers, and dev-ops teams on developing and maintaining cloud-deployed network services and applications across the lifecycle.**
  While working solo, I'll design the system with modularity (e.g., separate frontend, backend, and database components) to simulate team collaboration. The presentation will reflect this by explaining how the architecture supports lifecycle management, demonstrating advanced collaboration principles.

- **ULO5: Apply cloud orchestration technologies to optimise the deployment and management of network services and applications.**
  Using Kubernetes features like automatic scaling and Secrets for secure data management will optimize the system. I'll demonstrate this through metrics or logs in the presentation, showing how these optimizations enhance performance and security at an advanced level.