Topic 7

1. **Title:** ***Development of Bhuvan Generic framework using micro services architecture.***
2. **Description**: The workloads on Bhuvan web applications hosted on live environment vary continuously and needs a flexible platform for hosting requirements. A container-based approach will highly improve the efficiency in terms of resource utilization and for ease of maintenance. Containerization and Microservices are the latest software architectural approach that decomposes a complex application into a set of small, independent services, each responsible for a specific function. Containerization involves packaging these microservices and their dependencies into lightweight, portable containers. When both are combined in which it enables the developers to build and deploy the applications that are highly flexible and easy to manage.
3. **Objectives**:
   a. Develop the given application as a microservice using the given programming language and technology in a way that can be independently maintained, deployed and scaled.
   b. Package the microservice within the container to ensure portability.
   c. Use Kubernetes, container orchestration platform to automate the deployment and scaling.
   d. Implement clear API endpoints for interaction with the microservice
   e. Ensure the microservice can be integrated into the existing application infrastructure
4. **Expected Outcomes:**
   a. A fully functional microservice that address the chosen functionality.
5. **Relevant data and steps to get the data from Bhuvan/ other sources:** Any sample Web Application that needs to scaled the resources dynamically along with the sample data of Bhuvan WMS/ Images.
6. **Steps to be followed for achieving the objectives:**
   a. Identify a specific, well-defined functionality that the microservice will provide within the application.
   b. Choose a programming language and framework that best suits the requirements of the microservice.
   c. Developing a microservice implementing the defined functionality.
   d. Design and implement clear API endpoints for microservice.
   e. Containerize microservice using Docker. Create a Docker file to specify how application should be packaged into a container. Use Kubernetes to automate the deployment.
   f. Configure auto-scaling for microservice using Kubernetes to handle the changes in demand automatically.
7. **Evaluation**
   a. **Functionality and Purpose**: Whether microservice effectively provides the intended functionality and how easily it can be reused across different parts of the application or in other projects.
   b. **Containerization and Deployment**: Assess the effectiveness of containerization and how easily it can be deployed in different environments.
   c. **Scalability and Performance**: How well the microservice scales horizontally or vertically to handle increased loads. Measure its response time, latency and throughput under different loads to ensure it meets performance requirements.
   d. **Integration:** How well the microservice integrates with other services or systems.