Project Title: Electronic eCommerce website

About Project: An e-commerce platform for an electronics retail store involves deploying the retail application to a virtual sandbox on AWS/GCP using a centralization technique. The application deployment is facilitated through Kubernetes and Docker technologies.

Project Overview
In this project, we aim to develop and deploy a robust e-commerce platform tailored for an electronic retail store. The primary focus is on ensuring efficient deployment using cloud services, centralization techniques, and containerization technologies like Kubernetes and Docker.

Problem Statement
The existing e-commerce website offers product suggestions primarily based on product price and features. In our project, we will enhance this by sharing version specifications of a product, including its relationship with predecessor or successor products. This added information will provide consumers with valuable insights, helping them make informed decisions when selecting the right product version.

Objectives
1. Develop and deploy an e-commerce website on a cloud sandbox environment (AWS/GCP) using containerization and Kubernetes for easy scalability, high availability, and efficient resource management.
2. Utilize a microservices architecture for the e-commerce application to ensure flexibility, maintainability, and to facilitate the integration of new features and services.
3. Ensure security and compliance by utilizing best practices for securing data, user authentication, and access control.
4. Integrate popular payment gateways and shipping APIs to facilitate seamless transactions and deliveries.
5. Set up an efficient database management system to store and manage product information, user data, and order details.

Scope
1. Cloud Environment Setup: Provision and configure the required cloud resources on AWS/GCP, including virtual machines, storage, and networking components in a sandbox environment.
2. Containerization: Containerize the e-commerce application components using Docker, allowing for easy deployment, scaling, and management of the application.
3. Kubernetes Deployment: Deploy, manage, and scale the containerized application components using Kubernetes, ensuring high availability and efficient resource management.
4. Microservices Implementation: Break down the e-commerce application into multiple, independent microservices, allowing for flexibility and maintainability.
5. Security and Compliance: Implement best practices for securing data, user authentication, and access control, such as encryption, role-based access control, and secure API gateways.

Methodology/Approach
For this project, we will adopt an agile methodology, leveraging AWS as our chosen cloud provider. We will construct the website pages using PHP and employ MongoDB as the database system.

Expected deliverables
- Fully Functional E-commerce Website
- Shopping Cart
- Product Listings and Details
- Order Tracking
- Search Functionality
- Admin Dashboard
- Documentation

Project Timeline
The project will be executed over a specific timeline, with milestones and deliverables defined for each phase
Week 1-2 (October):
- Define project requirements and scope.
- Set up the cloud environment on AWS/GCP and configure required resources.
- Begin containerizing e-commerce application components using Docker.

Week 3-4 (October):
- Complete containerization of application components.
- Set up the Kubernetes environment and deploy containerized components.
- Implement microservices architecture for the e-commerce application.
- Start developing CI/CD pipeline.

Week 5-6 (November):
- Complete CI/CD pipeline implementation and testing.
- Implement security and compliance measures.
- Begin frontend development and database management system setup.
- Start integrating payment gateways and shipping APIs.

Week 7-8 (November):
- Complete frontend development and database management system setup.
- Finish integrating payment gateways and shipping APIs.
- Set up performance optimization techniques, such as caching and load balancing.

Week 9-10 (December):
- Implement monitoring and logging systems.
- Conduct extensive testing of the e-commerce site, including performance, security, and user experience.
- Address any issues or bugs found during testing and optimize the application.

Week 11-12 (December):
- Finalize documentation, including architecture diagrams, infrastructure details, and management instructions.

- Prepare for deployment, including any final testing and adjustments.
- Deploy the e-commerce site to AWS/GCP Sandbox.

Benefit/Impact

1. Microservice hosting on cloud (AWS/GCP) using Containerization ,Kubernetes and Docker enable us to Scalability, High Availability, Flexibility, Cost Efficiency of the e-commerce application to accommodate increased traffic and demand. This ensures that the website can handle traffic surges during peak times, such as sales events or holidays.
2. Faster Time-to-Market: The use of CI/CD pipelines and containerization simplifies and accelerates the deployment process, enabling developers to quickly push updates and new features to the e-commerce site.
3. Improved Security: By following best practices for securing data and implementing role-based access control, the e-commerce site can protect sensitive customer data and maintain compliance with industry regulations.
4. Enhanced Performance: Performance optimization techniques like caching, load balancing, and auto-scaling ensure that the e-commerce site delivers a fast and seamless user experience, even during periods of high traffic.
5. Better User Experience: A user-friendly and responsive web interface ensures that customers can easily navigate and shop on the e-commerce site across various devices and platforms.
6. Seamless Payment and Shipping Integrations: By integrating popular payment gateways and shipping APIs, the e-commerce site can provide a streamlined and convenient shopping experience for customers.

Team- Team 21

Name	**Role and responsibilities**
SANJIB Saha	Project management and Product owner
Shaurya Pratap Singh	Development(Fullstack)
Shiv Kumar Sah	Provisioning and Deployment
Shivam Srivastava	Database management, Testing and Documentation

Risks and Mitigations

1.Project delivery risk
Risk: failure to meet project delivery timelines
Mitigation: adopt the Scrum methodology and break down the project deliverables into multiple sprint cycles, aiming to prevent any Product Backlog Item (PBI) overflow or spillage.

2.Technical Risks
Risk: Technical glitches, downtime, or slow loading times can frustrate customers and result in revenue loss.

Mitigation: Employ a reliable hosting service (like AWS) with high availability and scalability features. Regularly monitor website performance, and have a disaster recovery plan in place. Implement content delivery networks (CDNs) for faster loading times.

3.Scalability Risks
Risk: Inadequate server capacity during peak traffic can lead to slow performance or downtime.
Mitigation: Use scalable cloud infrastructure like AWS, and regularly monitor website traffic patterns to anticipate scaling needs

References: eCommerce portals like Amazon and flipkart