Speaker Notes for CI/CD Presentation

## Slide 1: Title Slide - CI/CD Login and Signup Portal

Good [morning/afternoon], everyone. I’m Kyaw Htay, an intern DevOps engineer. Today, I’ll be presenting a project I built to apply DevOps practices and CI/CD automation in a real-world scenario. The project is a login/signup portal fully powered by a CI/CD pipeline and deployed on Kubernetes.

## Slide 2: Contents

Here’s the outline of today’s presentation. I’ll start with a project overview, followed by the vision, mission, and objectives. Then we’ll explore DevOps, GitOps, and CI/CD fundamentals. I’ll compare manual and automated deployments, show the architecture, tools used, implementation steps, challenges faced, benefits achieved, and finally share my learning outcomes and future plans.

## Slide 3: Project Overview

This project is a web application for user authentication, built with PHP and MySQL, and deployed using Kubernetes. What makes it special is the full CI/CD pipeline behind it, automating builds and deployments through GitHub Actions and Argo CD. It’s a hands-on implementation of DevOps practices like containerization and orchestration.

## Slide 4: Objectives

The main goal was to learn by doing. I wanted to understand how modern DevOps pipelines work and how automation improves the development lifecycle. Through this project, I gained real experience with tools like GitHub Actions, Docker, and Kubernetes.

## Slide 5: Vision

My vision was to reduce repetitive manual tasks by automating everything from build to deployment. I also wanted to prepare myself for industry-level DevOps workflows and improve my practical skills.

## Slide 6: Mission

My mission was clear: build a fully functional CI/CD pipeline and deploy an actual PHP application using GitOps principles. I focused on hands-on practice and aimed to demonstrate the reliability of automated software delivery.

## Slide 7: What is DevOps?

DevOps is about breaking down the wall between development and operations. It encourages collaboration, automation, and fast, reliable releases. In this project, DevOps meant automating every step from code commit to live deployment.

## Slide 8: What is GitOps?

GitOps uses Git as the single source of truth for deployments. With tools like Argo CD, you can deploy to Kubernetes by simply pushing changes to Git. It improves security, traceability, and speed. I used Argo CD in this project to implement GitOps.

## Slide 9: What is CI/CD?

CI stands for Continuous Integration, where code is automatically built and tested on every push. CD, or Continuous Deployment, means automatically delivering that code to production. This approach minimizes human errors and speeds up releases. In my case, GitHub Actions handled CI, and Argo CD handled CD.

## Slide 10: Manual vs Automated Delivery

Here’s a comparison of manual vs automated delivery. Manual deployments are error-prone and slow. Automated delivery, like what I built, ensures consistency, faster updates, and fewer issues in production. The pipeline triggers from Git commits and handles everything up to syncing the Kubernetes cluster.

## Slide 11: Architecture Diagram

This diagram shows the entire pipeline. It starts with code pushed to GitHub, which triggers GitHub Actions. It builds a Docker image and pushes it to Docker Hub. Argo CD then pulls the image and syncs the Kubernetes cluster to deploy the latest version.

## Slides 12–13: Tools & Technologies

I used a variety of tools:  
- PHP for backend logic.  
- MySQL as the database.  
- GitHub for version control.  
- GitHub Actions for automating the CI/CD pipeline.  
- Docker for containerization.  
- Docker Hub to store images.  
- Kubernetes for orchestration.  
- Argo CD to automate deployments via GitOps.  
- Ingress to expose services outside the cluster.

## Slides 14–20: Implementation

The implementation involved several steps: writing the PHP backend, containerizing with Docker, setting up GitHub Actions for CI, creating Kubernetes manifests, deploying with Argo CD, and configuring Ingress for external access. Each step helped build a deeper understanding of real-world DevOps processes.

## Slide 21: Benefits of CI/CD in This Project

The CI/CD pipeline made development faster and more reliable. Every change is automatically tested, built, and deployed. This reduced downtime and ensured the application always reflected the latest codebase.

## Slide 22: Challenges Faced

It wasn’t easy. I struggled with Kubernetes YAML files and debugging syntax errors. Learning Kubernetes and GitOps from scratch was challenging. But those challenges taught me the most.

## Slide 23: Learning Outcomes

I now understand how modern DevOps pipelines work. I’ve gained confidence using GitHub Actions, Docker, Kubernetes, and Argo CD. Most importantly, I learned how to troubleshoot and solve problems on my own—skills that are critical in DevOps.

## Slide 24: Future Plan

Moving forward, I plan to explore Flux CD as another GitOps tool and GitLab CI for more advanced pipelines. I also want to integrate the ELK Stack to add monitoring and logging capabilities for production-grade systems.

## Slide 25: Conclusion

In conclusion, this project helped bridge the gap between academic theory and real-world DevOps. I’m proud of what I’ve built and learned during this internship. With the help of my team lead and hands-on practice, I now feel more prepared for future DevOps roles.

## Slide 26: Thank You

Thank you all for your time and attention. I’m happy to take any questions or feedback you have.