4. System Architecture

4.1 High-Level Design

The High-Level Design of the Learning Platform with End-to-End CI/CD and GitOps on AWS defines the
overall system structure, components, and data flow. The platform integrates continuous integration
and continuous deployment pipelines with GitOps principles to automate code delivery.
The architecture uses GitHub for version control, GitHub Actions for automation, Docker for containerization,
and AWS ECS for deployment.

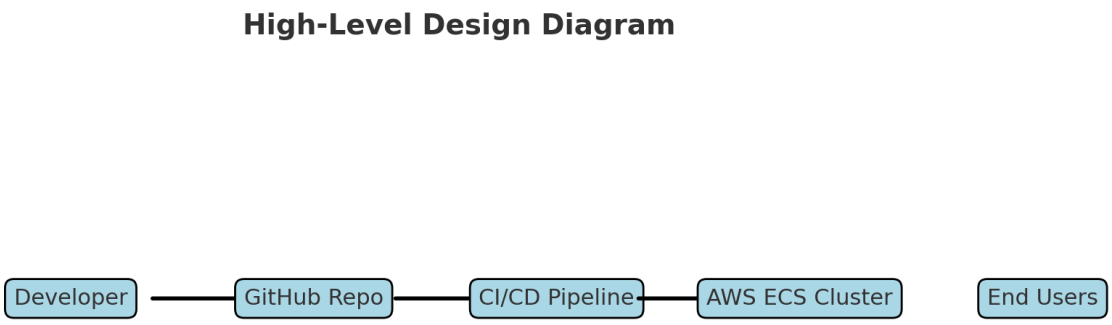

Figure 4.1: High-Level Design Diagram

4.2 Workflow Diagram

The workflow begins when developers commit code to the GitHub repository.
The CI/CD pipeline, triggered through GitHub Actions, performs automated testing, builds Docker images,
and pushes them to AWS Elastic Container Registry (ECR).
After a successful build, the deployment stage automatically updates AWS ECS (Elastic Container Service)
to deliver the latest version of the learning platform. GitOps ensures that the deployment configuration

remains consistent with the Git repository state.

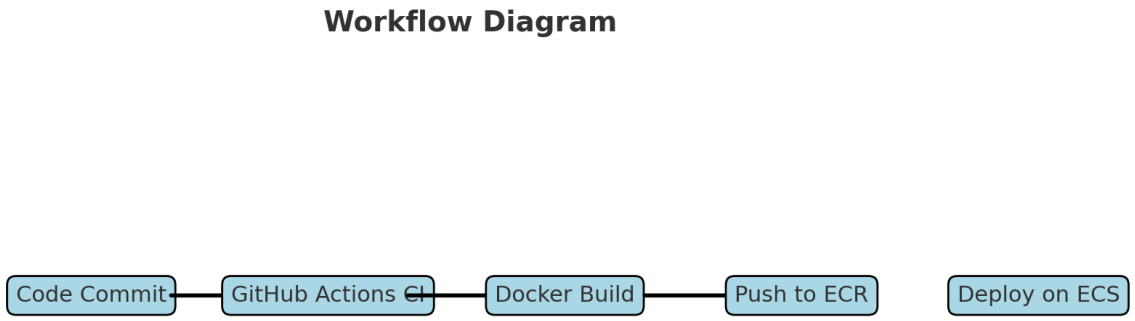

Figure 4.2: Workflow Diagram

4.3 Component Interactions

The component interactions describe how the system's modules and tools communicate to maintain smooth
DevOps operations. The frontend interacts with backend APIs hosted on AWS. The CI/CD system continuously
monitors the Git repository for changes and triggers automated workflows. AWS infrastructure handles
deployment, scaling, and network management, while the database layer stores and retrieves platform data.

When a developer commits changes, the CI/CD pipeline runs build and test stages.
If successful, the updated container images are deployed on AWS ECS. The frontend communicates with
backend services through REST APIs, and both layers access shared cloud resources securely.
This modular interaction ensures maintainability, scalability, and fault tolerance.

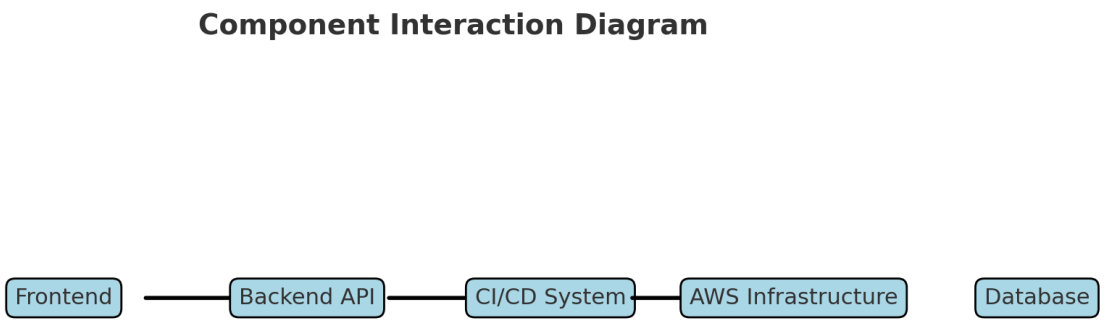

Figure 4.3: Component Interaction Diagram