**DevLearn**

**Jamiyanmyadag Baatar (12325947)**

**Urshita Parmar (12303424)**

**Anudari Ganbaatar (12327086)**

**Lovely Professional University, Punjab**

**Bachelor of Computer Science Engineering (DevOps Specialization)**

**Contents**

**1 Introduction**

**1.1 Project Definition. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**1.2 Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**1.3 Business Impact . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**2 Problem Statement**

**2.1 Current Challenges . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**2.2 Need for Automation. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**2.3 Industry Relevance . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**3 Project Scope**

**3.1 Key Deliverables . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**3.2 Technologies Used . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**3.3 Limitations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**4 System Architecture**

**4.1 High-Level Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**4.2 Workflow Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**4.3 Component Interactions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**5 Technology Stack**

**6 Implementation Details**

**6.1 CI/CD Pipeline . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**6.2 Containerization . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**6.3 Orchestration (EKS & Ingress) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**6.4 Monitoring & Logging . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**6.5 Performance Optimization . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .**

**7 Security & Compliance**

**8 Plan**

**9 Conclusion**

**1. Introduction**

1.1 Project Definition

DevLearn is an AI-powered platform that allows users to upload PDFs or PPTs and automatically generates summaries, MCQs, and flashcards, providing an interactive and efficient learning experience. The platform combines intelligent content extraction with secure, automated, and accelerated software delivery pipelines, giving DevOps students hands-on exposure to real-world workflows.

Key highlights:

* AI-driven content summarization and quiz generation.
* Automated CI/CD pipelines for fast and reliable updates.
* Integrated security and compliance checks for safe deployments.

1.2 Objectives

1. Automate CI/CD pipelines using Jenkins and GitHub Actions.
2. Enable zero-downtime deployments for uninterrupted access.
3. Ensure system security via SonarQube scans.
4. Monitor performance using Prometheus and Grafana.
5. Optimize resource usage for efficiency and cost-effectiveness.

1.3 Business Impact

* Provides instant learning resources, saving time for students and educators.
* Promotes hands-on DevOps experience through a live, automated system.
* Enhances learning quality with structured summaries and assessments.
* Demonstrates modern software delivery practices, including CI/CD and security automation.

**2. Problem Statement**

2.1 Current Challenges

* Manual content creation is time-consuming and inconsistent.
* Frequent updates or deployments often cause service disruption.
* Lack of integrated security checks and monitoring in existing platforms.

2.2 Need for Automation

* Automating content generation ensures fast, reliable learning resources.
* CI/CD pipelines allow frequent, zero-downtime updates.
* Integrated security tools maintain safe and compliant deployments.

2.3 Industry Relevance

* Aligns with modern DevOps practices in the software industry.
* Prepares students for real-world CI/CD, containerization, and monitoring workflows.
* Supports the growing demand for AI-driven educational tools.

**3. Project Scope**

3.1 Key Deliverables

* AI engine for PDF/PPT summarization.
* Automatic generation of MCQs and flashcards.
* Fully automated CI/CD pipeline with security and compliance checks.
* Monitoring dashboard for performance and usage metrics.

3.2 Technologies Used

* Frontend: React.js, Tailwind CSS
* Backend: Node.js, Express
* AI/ML: Python, NLP libraries
* CI/CD: GitHub Actions, Jenkins
* Containerization & Orchestration: Docker, Kubernetes/EKS
* Monitoring & Logging: Prometheus, Grafana, DataDog
* Security: SonarQube, Snyk, Trivy

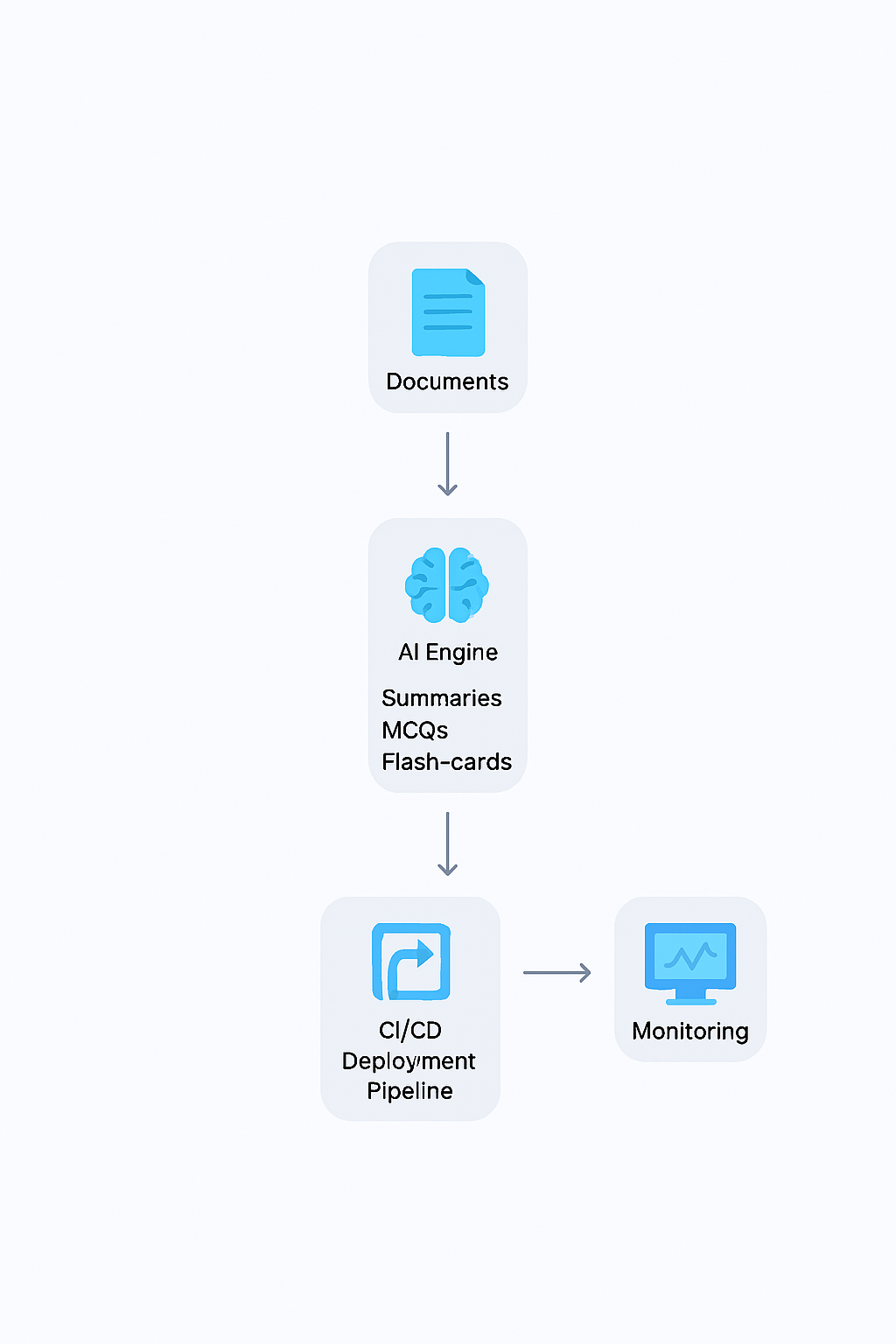
3.3 Limitations

* AI summarization may not capture all nuances of complex content.
* Resource-intensive tasks may require scalable cloud infrastructure.
* Real-time collaboration features are out of scope in the current version.

**4. System Architecture**

4.1 High-Level Design

Users upload documents → AI engine processes content → Generates summaries, MCQs, and flashcards → CI/CD pipeline deploys updates → Monitoring ensures system health.



4.2 Workflow Diagram

A diagram of a software development process

AI-generated content may be incorrect.

4.3 Component Interactions

* Frontend communicates with backend API.
* Backend interacts with AI modules and database.
* CI/CD pipeline triggers automated builds, tests, and deployments.
* Monitoring tools track performance, uptime, and security metrics.

**5. Technology Stack**

| Layer | Tools/Technologies |
| --- | --- |
| Frontend | React.js, Tailwind CSS |
| Backend | Node.js, Express |
| AI/ML | Python, NLP Libraries |
| CI/CD | Jenkins, GitHub Actions |
| Containerization | Docker |
| Orchestration | Kubernetes/EKS |
| Monitoring | Prometheus, Grafana |
| Security | SonarQube |

**6. Implementation Details**

6.1 CI/CD Pipeline

* Automated build, test, and deployment using GitHub Actions and Jenkins.

6.2 Containerization

* Docker containers for consistent development and production environments.

6.3 Orchestration (EKS & Ingress)

* Kubernetes EKS cluster with Ingress controllers for load balancing and routing.

6.4 Monitoring & Logging

* Prometheus collects metrics, Grafana visualizes dashboards6.5 Performance Optimization
* Auto-scaling, caching, and efficient resource allocation to enhance speed and cost efficiency.

**7. Security & Compliance**

* Automated scans using SonarQube.
* Secrets management and role-based access control.
* Compliance checks ensure alignment with industry standards and data privacy regulations.

**8. Plan**

Week 1: Setup & Planning

* Frontend: Initialize React project, create wireframes, set up routing.
* Backend: Initialize Maven project, configure Spring Boot, set up database, research Java NLP libraries.
* CI/CD: Set up GitHub repo, branch strategy, Docker skeleton, draft pipeline.
* Deliverables: Project skeletons ready, repo setup, Docker basic structure.

Week 2: Core Development

* Frontend: Implement file upload, placeholder UI for summaries, MCQs, flashcards.
* Backend: Implement upload endpoint, AI module for summary generation, store summaries in database.
* CI/CD: Set up build and test pipelines, create Docker images.
* Deliverables: Upload working, summary generation working, CI/CD pipeline skeleton operational.

Week 3: Integration

* Frontend: Connect to backend API for summaries, MCQs, flashcards; test with real data.
* Backend: Implement MCQ and flashcard generation, REST endpoints fully functional, unit/integration tests.
* CI/CD: Integrate Maven build & test, deploy staging environment with Docker/Kubernetes.
* Deliverables: Full frontend-backend integration, content generation complete, staging environment ready.

Week 4: Optimization & Security

* Frontend: Improve UI/UX, loading states, error handling, performance optimization.
* Backend: Optimize AI module, secure REST APIs (validation, JWT/Spring Security), improve database queries.
* CI/CD: Add security scans (SonarQube), monitoring setup (Prometheus, Grafana).
* Deliverables: Secure, optimized, monitored system.

Week 5: Final Testing & Deployment

* Frontend: Final UI polish, responsive design, cross-browser testing.
* Backend: End-to-end integration testing, stress testing, API documentation.
* CI/CD: Deploy full system, verify zero-downtime deployment, performance & security validation.
* Deliverables: Fully functional DevLearn platform, ready for demo and submission.

**9. Conclusion**

DevLearn is a secure, automated, and AI-driven learning platform that transforms uploaded PDFs and PPTs into summaries, MCQs, and flashcards. By integrating CI/CD, containerization, and monitoring, it ensures fast, reliability, and zero-downtime deployments while providing students with hands-on DevOps experience. The platform bridges the gap between theory and practice, enhances learning efficiency, and lays the foundation for future enhancements in personalized, adaptive education technology.