# Online Workplace Training Portal - Backend Architecture & Plan

## 1. Language and Framework Selection

Recommended Language: Go (Golang)  
  
Why Go?  
- Fast and lightweight  
- Great for REST APIs and concurrency  
- Easy to deploy (single binary)  
- Mature ecosystem for backend web services (e.g., Gin, Echo, Fiber)  
- Strong static typing with faster development cycle than Java or Rust  
  
Rust is excellent for performance but may be overkill for this use case due to slower development cycles. Java is robust but heavier to deploy compared to Go.

## 2. Project Directory Structure

training-portal/  
├── cmd/  
│ └── server/  
│ └── main.go  
├── internal/  
│ ├── domain/  
│ │ ├── user/  
│ │ │ └── model.go  
│ │ ├── course/  
│ │ │ └── model.go  
│ ├── usecase/  
│ │ ├── user/  
│ │ │ └── service.go  
│ │ ├── course/  
│ │ │ └── service.go  
│ ├── interface/  
│ │ ├── http/  
│ │ │ ├── middleware/  
│ │ │ ├── handler/  
│ │ │ │ ├── user.go  
│ │ │ │ └── course.go  
│ │ │ └── router.go  
│ │ ├── repository/  
│ │ │ ├── postgres/  
│ │ │ │ ├── user.go  
│ │ │ │ └── course.go  
├── migrations/  
├── configs/  
├── scripts/  
├── docs/  
├── web/  
├── .env  
├── go.mod  
└── go.sum

## 3. Architectural Plan

Architecture: Clean Architecture + Fiber (or Gin)  
  
Layers:  
1. Domain Layer - Interfaces and models  
2. Usecase Layer - Business logic implementations  
3. Interface Layer - HTTP handlers and DB adapters  
4. Infrastructure Layer - PostgreSQL and file storage

## 4. Core Dependencies

- Web Framework: Fiber or Gin  
- ORM/DB: GORM or pgx  
- JWT Auth: github.com/golang-jwt/jwt/v5  
- Validation: github.com/go-playground/validator/v10  
- Env Config: github.com/spf13/viper  
- Migrations: github.com/golang-migrate/migrate/v4

## 5. Testing and Frontend

Testing:  
- Unit tests per usecase  
- Use Go’s built-in testing  
  
Frontend:  
- React + TypeScript  
- TailwindCSS or Material UI  
- Connect to Go backend via REST

## 6. Suggested Sprint Plan (8 Weeks)

Week 1: Setup project, define models, DB migrations  
Week 2: Implement Auth (JWT), User CRUD  
Week 3: Course CRUD, File uploads  
Week 4: Module/Quiz engine, Enrollments  
Week 5: Progress tracking, scoring  
Week 6: Certificates and reporting  
Week 7: Notifications and gamification  
Week 8: Testing and deployment

## 7. DevOps / Deployment

- Docker for Go backend and React frontend  
- PostgreSQL (Docker or managed)  
- File storage (local or S3)  
- CI/CD: GitHub Actions or GitLab CI  
- Deployment: Fly.io, Railway, Render, AWS/GCP