Yes ✅ you can absolutely replace **Java Spring Boot** with **Python** for your e-learning platform backend. Many modern ed-tech platforms (like Udemy, Coursera) use Python-based stacks because of its flexibility, AI/ML integrations, and large ecosystem.

Here’s how the tech stack and step-by-step repo design would look if you go with **Python** instead of Java Spring Boot:

**🔹 Updated Tech Stack (with Python)**

**Frontend:**

* ReactJS (for web)
* React Native (for mobile app)
* TailwindCSS / Material UI (UI components)

**Backend (Python):**

* **FastAPI** (modern, async, super fast, better than Flask/Django for APIs)
* **GraphQL** (via Strawberry/FastAPI GraphQL or Ariadne) for flexible queries
* **REST API** for standard endpoints

**Databases:**

* MySQL → structured data (users, courses, payments, enrollments)
* MongoDB → semi-structured data (discussions, reviews, dynamic course metadata)
* AWS S3 → unstructured data (videos, notes, images, PDFs)

**AI/ML Add-ons (Python):**

* **NLP (Transformers like BERT/GPT)** → personalized recommendations, auto-tagging, QnA search
* **Computer Vision (OpenCV/TensorFlow/PyTorch)** → proctoring (face detection), image recognition
* **Recommendation System (Collaborative + Content-based Filtering)** → recommend courses

**DevOps:**

* Docker + Kubernetes for deployment
* GitHub Actions / GitLab CI for CI/CD
* AWS/GCP/Azure for hosting

**🔹 Repository Design (Monorepo or Polyrepo)**

You can go with **Polyrepo** (separate repos for each service) for scalability.

elearning-platform/

│

├── frontend-web/ # ReactJS (web portal)

│ ├── src/

│ ├── public/

│ └── package.json

│

├── frontend-mobile/ # React Native (mobile app)

│ ├── src/

│ └── package.json

│

├── backend-api/ # FastAPI backend

│ ├── app/

│ │ ├── main.py # entry point

│ │ ├── routes/ # REST & GraphQL routes

│ │ ├── models/ # SQLAlchemy (MySQL models)

│ │ ├── schemas/ # Pydantic schemas

│ │ ├── services/ # Business logic

│ │ └── utils/ # JWT, OAuth, payments

│ ├── tests/

│ └── requirements.txt

│

├── recommender-system/ # Python ML models

│ ├── notebooks/ # Jupyter experiments

│ ├── models/ # Trained ML/DL models

│ └── service/ # FastAPI microservice for recommendations

│

├── media-service/ # S3 file handling

│ └── app.py

│

├── discussion-forum/ # Forum (MongoDB + FastAPI)

│ ├── routes/

│ └── models/

│

├── devops/ # CI/CD, Docker, K8s

│ ├── Dockerfile

│ ├── docker-compose.yml

│ └── k8s/

│

└── docs/ # Documentation

**🔹 Step-by-Step Process**

**1. Setup Backend (FastAPI + GraphQL + MySQL + MongoDB)**

* Install FastAPI + Uvicorn
* Connect MySQL with SQLAlchemy ORM
* Connect MongoDB with Motor (async Mongo driver)
* Create models: Users, Courses, Enrollments, Payments (MySQL)
* Create forum/review collections in MongoDB
* Setup JWT auth for login/signup
* Add GraphQL for querying course catalog & reviews

**2. Media Service (AWS S3 Integration)**

* API for video/PDF/image upload & retrieval
* Secure signed URLs for streaming (like Udemy video system)

**3. Frontend Development**

* **ReactJS (Web):**
  + Pages: Home, Course Catalog, Course Player, Discussion Forum, Dashboard, Payment
  + Integrate REST/GraphQL APIs
* **React Native (Mobile App):**
  + Similar to web but optimized for mobile (offline downloads, push notifications)

**4. Recommendation & AI/ML Service**

* Build a recommendation engine (Collaborative Filtering + NLP on course tags)
* Use Python ML libraries (scikit-learn, TensorFlow, PyTorch)
* Deploy as a separate FastAPI service

**5. Discussion Forum (Semi-structured Data in MongoDB)**

* Threads, replies, upvotes stored in MongoDB
* Real-time updates with WebSockets

**6. Authentication + Payments**

* JWT/OAuth for user authentication
* Payment gateway integration (Stripe, Razorpay for India)

**7. DevOps & Deployment**

* Dockerize all services (FastAPI, frontend, ML service, forum service)
* Use Kubernetes for scaling
* CI/CD with GitHub Actions
* Host on AWS (EC2/EKS/RDS/S3/CloudFront)

⚡ **Result** → A complete **E-learning Platform** like Udemy/Scaler with:  
✅ Structured (MySQL), Semi-structured (MongoDB), Unstructured (S3) data  
✅ Python-powered backend with ML recommendation system  
✅ React + React Native frontends  
✅ Scalable microservice architecture

**📅 Roadmap (8 Weeks Plan)**

**Week 1 – Project Setup & Foundations**

* ✅ Create GitHub repositories (frontend-web, frontend-mobile, backend-api, recommender-system, devops, etc.)
* ✅ Setup **FastAPI backend** with Uvicorn/Gunicorn
* ✅ Initialize MySQL (RDS/Local) & MongoDB
* ✅ Setup project structure in backend-api:
  + models/ (SQLAlchemy ORM)
  + routes/ (Auth, User, Course)
  + schemas/ (Pydantic models)
* ✅ JWT authentication (Signup, Login, Refresh token)
* ✅ API test using Postman

Deliverable:

* Backend running locally with Auth & DB connection.
* GitHub repos initialized.

**Week 2 – Core Backend (Users, Courses, Enrollments, Payments)**

* ✅ MySQL Models:
  + Users (students, instructors, admins)
  + Courses (title, description, price, metadata)
  + Enrollments (user\_id, course\_id, progress)
  + Payments (transactions, receipts)
* ✅ REST APIs:
  + Create/Get Courses
  + Enroll in Course
  + Payment Integration (Stripe/Razorpay sandbox)
* ✅ GraphQL (optional, for course catalog queries)

Deliverable:

* Core backend APIs ready.
* Course + Enrollment flow working via API.

**Week 3 – Media Service & AWS S3 Integration (Unstructured Data)**

* ✅ Setup AWS S3 bucket
* ✅ FastAPI route for **file upload/download**
* ✅ Secure signed URLs for video streaming (like Udemy)
* ✅ Store uploaded file metadata in MySQL (course → video mapping)

Deliverable:

* Videos/PDFs uploaded securely to AWS S3
* Course content retrieval via API

**Week 4 – Discussion Forum & Reviews (Semi-structured Data)**

* ✅ MongoDB Collections:
  + ForumThreads (course\_id, user\_id, title, content)
  + ForumReplies (thread\_id, user\_id, content, timestamp)
  + Reviews (course\_id, rating, text, upvotes)
* ✅ FastAPI routes for:
  + Post new thread
  + Reply to thread
  + Add/Get reviews
* ✅ WebSocket integration (for real-time discussions)

Deliverable:

* Forum & Reviews working in backend.
* MongoDB connected.

**Week 5 – Frontend Web (ReactJS)**

* ✅ Setup React project with TailwindCSS/Material UI
* ✅ Pages:
  + Home (list courses)
  + Course Detail (course info + videos)
  + Enroll/Payment
  + Dashboard (enrolled courses, progress)
  + Forum & Reviews
* ✅ Integrate APIs (FastAPI REST/GraphQL)

Deliverable:

* Functional web portal (basic flow: browse → enroll → learn → discuss).

**Week 6 – Mobile App (React Native)**

* ✅ Setup React Native project
* ✅ Implement:
  + Login/Signup
  + Browse courses
  + Enroll & payment
  + Course player (video streaming from S3)
  + Forum access
* ✅ Push Notifications (Firebase or OneSignal)

Deliverable:

* Mobile app with main course flow running.

**Week 7 – AI/ML Features (Python Microservices)**

* ✅ Recommender System:
  + Collaborative Filtering (recommend based on other users)
  + Content-based (recommend similar courses using NLP on course titles/descriptions)
* ✅ NLP Search:
  + Use Transformers (BERT) for semantic search in forum/reviews
* ✅ Proctoring (Optional Advanced):
  + OpenCV face detection for test/exam monitoring

Deliverable:

* ML microservice deployed (FastAPI).
* Recommendations integrated in web + mobile.

**Week 8 – DevOps, Testing & Deployment**

* ✅ Dockerize all services (frontend, backend, recommender, forum)
* ✅ Docker Compose for local multi-service setup
* ✅ Setup CI/CD with GitHub Actions (test → build → deploy)
* ✅ Deploy on AWS/GCP/Azure (EKS, RDS, S3, CloudFront)
* ✅ Load Testing & Security (JWT expiration, role-based access)

Deliverable:

* Full platform deployed & accessible online.
* CI/CD pipeline ensures auto-deploy on push.

**🔹 Final Output**

At the end of 8 weeks you’ll have:  
✅ **E-learning Platform** like Newton School/Udemy with

* Structured (MySQL), Semi-structured (MongoDB), Unstructured (S3) data
* Web + Mobile frontends
* AI-powered recommendations & search
* Scalable backend with Python FastAPI
* Production-ready DevOps setup