Community LMS Web App Full Development Plan

Project Objective

To build a **web-based Learning Management System (LMS)** for a campus innovation club, where members of different **communities** (e.g., Web Dev, AI/ML, IoT) can:

- Join learning communities
- Access structured learning **roadmaps** from beginner to advanced
- Mark progress as they complete resources
- Receive **ML-powered personalized recommendations** for what to learn next
- Enable admins to add or manage resources per community

Phase-by-Phase Breakdown (with Features)

PHASE 1: Project Initialization & Setup

Objective:

Set up the project architecture for Flutter Web and the ML backend.

Flutter Web Tasks

- Create Flutter Web project
- Enable web support: `flutter config --enable-web`
- Add dependencies:
  - `firebase_core`

  - `firebase_auth`

  - `cloud_firestore`

  - `provider` or `riverpod`

  - `http` (to call ML API)

ML Backend Tasks

- Set up Python project (FastAPI or Flask)

- Install required packages:

```bash
pip install fastapi uvicorn scikit-learn pandas sentence-transformers joblib
```

- Create project folder and structure

PHASE 2: Authentication & User Onboarding

Objective:

Allow users to sign up, log in, and join communities.

Flutter Web Features

- Firebase Auth: Signup, Login, Logout

- Join Community screen:

  - Display available communities

  - Save selection to Firestore

- Save user profile: `uid`, name, email, joined communities

ML Tasks

- Prepare dummy resource dataset (JSON)

- Set up user progress simulation (for testing recommendations)

PHASE 3: Roadmap Navigation & Resource Tracking

Objective:

Users can navigate a communitys roadmap and track their learning.

Flutter Web Features

- Your Communities dashboard

- View learning roadmap by level (Beginner Intermediate Advanced)

- Resource card view:

  - Title, type, description

  - Mark as Completed button

- Store user progress in Firestore (`completed_resources`)

ML Tasks

- Prepare embedding model (`sentence-transformers`)

- Build resource vector database using titles + descriptions

- Store as `embeddings.pkl`

PHASE 4: ML Recommendation Engine

Objective:

Build a smart engine that recommends the next best resources to learn.

ML Features

- Input: users completed resource IDs
- Logic: compute average vector cosine similarity with all resources
- Output: top N recommended resources
- Create `/recommend` API in FastAPI

Flutter Web Features

- Add HTTP POST to ML backend
- Display list of Recommended For You resources on dashboard

PHASE 5: Admin Panel & Resource Management

Objective:

Enable admins to manage resources for their communities.

Admin Features

- Admin login detection
- Resource uploader form:
  - Title, Description
  - Tags (ai, web, etc.)
  - Type (video, article, task)

  - Level (Beginner, etc.)

  - Link (YouTube, article, etc.)

- Save to Firestore under correct community and level

ML Tasks

- Recompute embeddings when new resources are added

- (Optional) Build auto-tagging feature using NLP (topic extraction)

PHASE 6: Testing, Polish & Deployment

Objective:

Test app across devices, finalize design, and deploy both frontend & backend.

Flutter Web

- Responsive layout for desktop & mobile browsers

- Add progress indicators, success messages

- Build app: `flutter build web`

- Deploy to Firebase Hosting

ML Backend

- Test endpoint with sample users

- Deploy API using Render or Railway

- Share public endpoint with frontend

OPTIONAL: Future Features

Feature	ML or Flutter?
Personalized learning paths	ML (Clustering or Classification)
Auto-evaluation of short answers	ML (NLP: BERT/Semantic search)
Forum/Discussion channels	Flutter
Dropout prediction	ML
In-app quizzes	Flutter + ML

Final Checklist

Task	Owner	Status
Flutter Web UI setup	Frontend Dev	
Firebase Auth integration	Frontend Dev	
Community & Roadmap screens	Frontend Dev	
ML backend setup (FastAPI)	You	
Recommender system logic	You	
`/recommend` API + Deployment	You	
ML-Flutter integration	Frontend Dev	
Admin resource panel	Frontend Dev	
Firebase hosting for Flutter	Frontend Dev	
Render/Railway deployment for ML API	You	