FINAL PROJECT PROPOSAL

Project Name: SkillUp Academy

Submission Deadline: 21/11/2025

---

1. Project Description

Our project is a **Specialized Educational E-Platform** focused on **Skills Development**, covering both Hard Skills (technical) and Soft Skills (personal/professional). The platform aims to provide practical, interactive content centered around the actual application of in-demand skills in the job market.

Core Focus: To offer curated learning paths designed to help users master specific skills through micro-learning modules, practical challenges, and a continuous assessment system to track student progress toward mastery.

Architecture: The project is entirely built using the **N-Tier Architecture** pattern to ensure strict **Separation of Concerns**, which guarantees scalability, maintainability, and testability.

2. Group Members & Roles

Team Leader: Mohamed Haitham Afify

Team Member	Core Role	Completed & Current Tasks	Pending & Specific Tasks
Mohamed Haitham Afifi	Content Configuration & Business Rules Lead	CRUD for Course Category, SubCategory, and Course Review.	**Identity:** Setup Admin classes and user management. **API:** Build Endpoints for Categories and Reviews.
Mohamed Sabry Ouda	Financial & Assessment Systems Lead	CRUD for Subscription Plan, Subscription, and Exam Attempt.	**Payment:** Build Instructor Payouts functionality. **API:** Build Endpoints for Subscriptions and Plans.
Abdullah Mohamed Goudah	Course Structure & Content Flow Lead	CRUD for Course, Course Lesson, and Lesson.	**Identity:** Setup Student classes and Registration/Login functionality. **API:** Build Endpoints for Course and Lesson.
Mahmoud Mohamed Ahmed Hussein	Exam & Question Logic Lead	CRUD for Exam, Question, and Answer.	**Identity:** Setup Teacher classes and profile management. **API:** Build Endpoints for Exam and Question.

General Note: All members collaborated on the initial database schema design and standardization during Phase 1.

3. Objectives

Objective	Description	Metrics
1. Launch Minimum Viable Product (MVP)	Build a functional platform containing all core features and security functions (Identity) to enable the basic student and instructor life cycle.	Achieve **100%** on KPI #1 (System Functionality Completion Rate) for defined core features.
2. Ensure N-Tier Architecture Compliance	Strictly apply the N-Tier Architecture pattern to separate the Presentation/API, Business/Service, and Data Access layers.	Achieve **100%** adherence to the rule: no direct DbContext calls outside the **DAL** layer (KPI #2).
3. Provide Secure & Role-Based Access (Identity)	Fully integrate ASP.NET Identity Framework to distinguish the three roles (Student, Teacher, Admin) and provide secure, restricted access for each.	Achieve **100%** success in security tests for registration, login, and role-based authorization.
4. Ensure Code Quality & Scalability	Adhere to engineering standards, including layered architecture and Dependency Injection, to ensure clean, maintainable, and scalable code.	Achieve a minimum of **60%** Unit Test Coverage for the Business Logic layer.

4. Tools & Technologies

Category	Technology or Tool	Purpose
Primary Language	C# (Latest Version)	Full server-side development and business logic implementation.
Primary Framework (Backend)	ASP.NET Core MVC	Building the Presentation Layer, handling HTTP requests, and providing MVC/REST Endpoints.
Architecture Pattern	**N-Tier Architecture**	Strict separation between Application Layers (Presentation, Business Logic, Data Access).
Database	SQL Server	Primary Relational Database Management System.
Data Management (ORM)	Entity Framework Core (EF Core)	Database interaction, querying, and schema management via Migrations.
Identity & Security	ASP.NET Core Identity	User management, roles (Student, Teacher, Admin), and authentication.
Frontend (Current)	HTML5, CSS3, Bootstrap,JavaScript	Building the user interfaces (Views) using the traditional MVC pattern.
Coding Principle	Dependency Injection (DI)	Ensuring code testability, maintainability, and quality.
Version Control	Git & GitHub	Code management, change tracking, and team collaboration.

5. Milestones & Deadlines

Milestone	Covered Tasks	Status	Deadline
Phase 1: Foundation & Design	Schema design, Initial Migrations, and N-Tier structure definition.	**Completed**	**7/10/2025**
Phase 2: Core CRUD Functionality	Development of all Entities, Controllers, and Views for basic CRUD operations.	**Completed**	**17/10/2025**
Phase 3: Security & Architectural Integration (Identity & DI)	Implement ASP.NET Identity, Registration/Login, strict Dependency Injection, and basic Dashboards.	**In Progress**	**7/11/2025**
Phase 4: Financial Systems & Final Review & Testing	Develop Instructor Payment logic & Test coverage, performance optimization, bug fixing, and final documentation.	**Plan**	**21/11/2025**

6. Key Performance Indicators (KPIs)

KPI	What it Measures	Customization for the Educational Platform Project
1. System Functionality Completion Rate	Percentage of planned MVP features successfully implemented and tested.	**Customization:** We define **14 Core Features** for the system. The KPI is calculated as: . **Goal:** Achieve **100%** completion rate for all core MVP functionalities.
2. Code Quality & Architecture Compliance Score	Adherence to N-Tier structure, clean coding, and Dependency Injection usage.	**Customization:** Measured by three factors: 1. **Layer Separation:** Ensuring no direct DbContext calls occur outside the **DAL** layer. 2. **Architecture:** Strict usage of Dependency Injection for all services. 3. **Unit Test Coverage:** Achieving a minimum of **60%** Unit Test Coverage for the Business Logic layer.
3. Testing & Reliability Rate	Percentage of test cases passed successfully, reflecting system stability.	**Customization: 30 main test cases** are defined (including student registration, course creation, subscription payment). The KPI is calculated as: . **Goal:** Achieve a success rate of at least **95%** in integration testing before final submission.
4. Performance & Efficiency Rate	Measurement of system responsiveness, particularly for critical functions and the API.	**1**. **Database Queries:** No single query in the Repositories should take longer than **150ms** to execute.
5. Error or Defect Rate	Tracking the number of critical issues discovered during testing relative to the testing volume.	**Customization:** Measured by the number of **Critical/High Priority** bugs discovered. **Goal:** The critical defect rate must not exceed **1 bug per 50 test cases** executed, with all critical defects resolved before the final deadline.