Names:

Emmanuel Mwamba 2420928 – Team Member A

Elijah Nonde 2420934 – Team Member B

Leonard Nyirenda 2300797 – Team Member C

Group: 3

Course: Software Design

Course Code: BSE 2210

Assignment: 1

Lecturer: Mr. Chikwanda J

Design & Architecture Report for the University Unified Student Experience Platform (USEP)

Member A — Design & Principles Lead:

1. Software Design in 2025 — Process and Artifact

Software design in 2025 is both **a process** and **an artifact**.

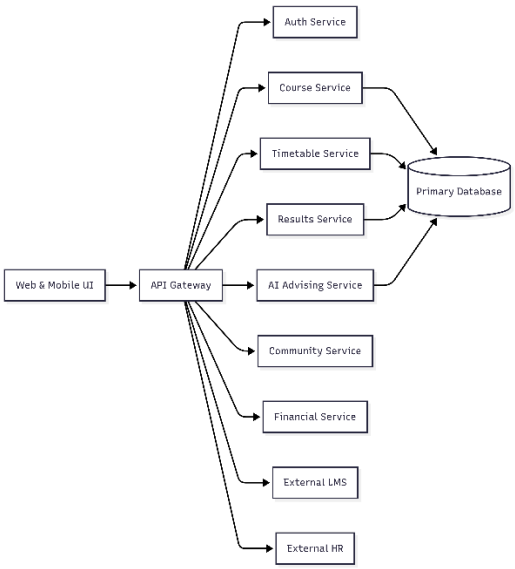
- **As a process**, it is collaborative and iterative, involving students, lecturers, and administrators. Prototypes are built, tested, and refined continuously, not just once at the beginning.
- **As an artifact**, it produces tangible outputs like diagrams, Architecture Decision Records (ADRs), and contracts. This helps record decisions, onboard new members, and keep the platform evolving smoothly.

This dual view ensures that design is both practical and durable for the University Unified Student Experience Platform (USEP).

2. System Overview of USEP

USEP integrates academic, financial, and community services into one unified platform. The high-level component diagram below shows how its parts interact.

System Overview / Component Diagram



Students access the system via web or mobile. All requests pass through an API Gateway, which routes them to services such as Course, Timetable, Results, Advising, Community, and Payments. Each service connects to a shared database or external systems such as LMS and HR. This setup makes the platform flexible and scalable.

3. Key Architecture Decision — ADR

One artifact we prepared is an **Architecture Decision Record (ADR)**. This document explains why we chose microservices over a monolith. The **ADR** can be found in the **docs** folder of our repository as ***ADR.md.***

4. Trends in Modern Software Design

Three trends most relevant to USEP are:

- **Microservices** → independent services that scale on demand.
- **AI integration** → for personal advising, chatbots, and fraud detection.
- **Sustainable architecture** → cloud-native and serverless designs that reduce cost and energy.

These trends make USEP future-ready and student-focused.

5. Principles-First vs Application-First

Design can begin either with principles or with quick applications.

- **Principles-first** ensures values like privacy, accessibility, and modularity are built in from the start.
- **Application-first** moves faster initially but risks technical debt.

For USEP, a **principles-first** approach is the best because protecting student data and ensuring accessibility are non-negotiable.

Member B — Business case and outsourcing analysis:

Problem

Currently, student services at the university are **fragmented** across multiple platforms:

- Academic services (registration, results) are separate from support services (financial aid, advising).

- Community services (events, clubs, forums) lack digital integration.

- Students face delays, redundant logins, and inconsistent user experiences.

This fragmentation negatively affects student satisfaction, retention, and operational efficiency.

Proposed Solution

Develop the Unified Student Experience Platform (USEP):

- A single digital hub integrating academic, support, and community services.

- Seamless integration with LMS and HR systems.

- Cloud-native, modular design (microservices + AI assistance).

- Long-term scalability and sustainability beyond 2025.

Expected Value

For Students

- Personalized experience through AI-powered academic advising.
- Convenience with one-stop access to academic, financial, and community services.
- Increased engagement and inclusivity with multilingual and accessible features.

For the University

- Improved retention rates by offering stronger support systems.
- Operational efficiency by reducing redundancies across multiple systems.
- Enhanced global competitiveness, making the university attractive to international students.
- Cost-effectiveness through sustainable architecture and smart outsourcing.

4. Outsourcing Analysis

Options

- Onshore (local outsourcing):

+ Pros: cultural alignment, easier communication.
+ Cons: expensive, limited scalability within budget.

- Offshore (far-away outsourcing, e.g., Asia):

+ Pros: very cost-effective, large talent pool.
+ Cons: time-zone issues, cultural barriers, and possible quality concerns.

- Nearshore (regional outsourcing, e.g., within Africa or Europe):

+ Pros: lower costs than onshore, easier collaboration than offshore, compatible time zones.
+ Cons: may still require cultural alignment efforts.

Recommendation

We recommend Nearshore Outsourcing.

- It balances cost-efficiency with effective collaboration.

- Developers within similar time zones can provide faster response times.

- Regional alignment makes it easier to meet cultural and legal standards (e.g., data privacy).

Member C — Culture & Ops Lead: