Nazarbayev University

Alumni Management System (AMS)

Final Report

School of Engineering and Digital Sciences

Department of Computer Science

Authors:

Temirlan Zaikenov

Aigun Bolganbayev

Ayaulym Mukan

Elvina Mynzhassar

Sanzhar Ospanov

Supervisor: Prof. Askar Boranbayev

Astana, Kazakhstan

April 25, 2025

Contents

1 Executive Summary **2**

1.1 Methodology 2

1.2 Main results: 3

2 Introduction **3**

2.1 Problem Statement, Motivation, and Significance 3

2.2 Overview of the Proposed Solution 5

2.3 Comparative Analysis of Approaches 7

2.4 Justification of Methodology 8

3 Project Approach **8**

3.1 Solution Overview 8

3.2 System Architecture 8

4 Project Execution **13**

4.1 Timeline and Development Phases 13

4.2 Project Management Reflection 16

5 Evaluation **19**

6 Conclusion and Possible Future Work **21**

6.1 Conclusion 21

6.2 Future Work 22

7 References **23**

1 Executive Summary

The Alumni Management System (AMS) project is a web-based platform developed to support the efficient storage, management of alumni data and managing and annouccing engagements for alumni of Nazarbayev University. While current methods of keeping data about alumni have given basic alumni tracking functions, they are not ideal for managing dynamic and digitally connected alumni networks.

The goal of AMS is centralize all of these processes to a one secure system that provides a reliable data maintenance, improved communication and structured way of interacting with alumni. Systems is developed to reduce possible data inconsistencies organization of alumni events and create ways to have a connection with and among university alumni. The web page has multilingual support offering information in English, Kazakh, and Russian languages. Also AMS has responsive design, let's AMS to be accessible across devices increasing its convenience. By developing such tools our project aims to improve university's relationship with its alumni community and helps with management of data.

Key Objectives include:

- Design and implement a centralized alumni information system.
- Enable secure access, real-time updates, and efficient alumni engagement.
- Provide multi-language support (English, Kazakh, Russian).
- Facilitate event management, career networking, and personalized communication.

1.1 Methodology

Our team used principles of Agile methodology. While working on the project, we used online tools like Jira and Notion for planning, Telegram and Google Meet for everyday communication and online meetings, and stored all of our written code in GitHub. At the beginning, we identified what types of work needed to be done and divided those tasks among team members. This helped us to quickly progress towards other tasks. The general goals, backlog, and responsibilities were decided together with all of the team members. Throughout the academic year, depending on the status of the progress, we had team meetings at least once a week and constantly kept each other updated on our progress via text channels.

The team deployed the system by integrating Laravel PHP with PostgreSQL for data storage through Docker and NGINX while developing frontend components in React (TypeScript) using Agile Scrum methodology. The team made sure Laravel Reverb had the necessary settings to support real-time WebSocket operations.

1.2 Main results:

- A fully functional alumni database with CRUD operations
- Secure authentication via JWT and RBAC
- A responsive, multilingual user interface
- Event creation, and alumni directory search
- A containerized deployment environment for future scalability
- Online messenger in-built for connecting between alumni members

This project aligns with the full cycle of a computing-based solution:

- Design: Secure, user-centric architecture
- Implementation: Modern, modular full-stack technologies
- Evaluation: Stakeholder walkthroughs and usability testing confirmed successful replacement of legacy processes

2 Introduction

2.1 Problem Statement, Motivation, and Significance

Universities across the world are increasingly aware of the need to keep good relationships with their alumni in order to build community, boost institutional image, and support fundraising, mentoring and career development. Nevertheless, most institutions, including Nazarbayev University, continue to use archaic systems like spreadsheets to manage alumni data. A high degree of inefficiency however arises from the fact that this manual method involves a lot of manual labour such as data fragmentation, inconsistent updates, communication delays, and privacy risks.

This absence of a centralized digital infrastructure not only hurts administrative efficiency but also detracts from the importance of engagement strategies–which, as any school already knows, are essential for the development of a strong online community base comprised of alums that are loyal and engaged. In line with Jonbekova et al. (2022), alumni are important sources of contributions to the development of higher education institutions, providing them with mentorship opportunities, networking opportunities, as well as supporting institutional growth, especially in the post-Soviet context of Kazakhstan. In a study of Cornea (2024), it is argued that an effective alumni engagement platform is key to keeping the relationship with the university alive and to increasing the visibility of the university.

While universities in Kazakhstan are starting to appreciate the value of alumni relations, as Sagitova (2018) claims, the participation of graduates in alumni activities is low because of the absence of well structured engagement infrastructure. This suggests the need of a more compelling and integrated digital platform. Dzhanegizova (2024) also notes that although Kazakhstani higher education has accelerated through digital transformation, most universities have not been able to overcome structural and strategic challenges in establishing sustainable digital ecosystems. This provides an opportunity to fill a digital gap within the national digital transformation framework and, in relation, a centralised Alma mater Management System (AMS) becomes an opportunity for alignment.

Barnard and Rensleigh (2008) from a systems design perspective state that dedicated alumni platforms are strategic assets for institutions to increase alumni loyalty, communication and create long term value. This aligns with their findings that centralized web applications designed for use by alumni should not be substituted with pieced tools.

According to Iskhakova, Hilbert, and Hoffmann (2016) post-Soviet academic institutions need sustained structured engagement opportunities with digital systems for developing alumni loyalty in both behavior and attitudes. The operational capability of these systems demands equal importance to their ethical and legal considerations. The study conducted by Amirov et al. (2024) examines how Kazakhstan should strengthen its data privacy framework by adopting international standards to achieve better digital data governance. Sensible architecture that values privacy needs to be established in alumni systems through encryption for authentication procedures alongside local storage

and specific consent protocols.

The primary reason to develop this project focuses on designing an Alumni Management System (AMS) that provides data protection alongside multilingual capabilities and user-friendly features. The initiative follows best practices of designing computing-based systems which focus on real-world problem solutions through human-centered software engineering (Shneiderman, 2020).

2.2 Overview of the Proposed Solution

The Alumni Management System (AMS) stands as a modern web-based application which serves Nazarbayev University for contemporary alumni relationship management. The solution merges different centralized components into a single platform where users can execute multilingual communications through a safe scalable structure integrated with real-time performance. Through this system alumni benefit from self-administration of their profiles while engaging with events and networking with peers along with administrative functions driven by management tools delivering structured data systems.

The proposed system addresses all the fundamental problems discovered by research publications that operate within educational institutions. Sagitova (2018) shows that Kazakhstan suffers from low organizational structure in its alumni systems even though Barnard and Rensleigh (2008) explain how dedicated portals create loyal alumni and strengthen community bonds. The AMS platform from Nazarbayev University provides a tailored system to merge user information with reach-out tools and connection channels for its international students who speak multiple languages.

The system is composed of five integrated components:

Frontend Interface

The frontend uses React.js to provide staff members and alumni access to an accessible interface which works across different devices. The essential web pages of the application consist of login and registration alongside password reset and profile editing functions. The platform navigates smoothly through React Router and uses Axios for real-time API calls. The system interface contains Kazakh, Russian and English language options to fulfill Kazakhstan's academic requirements for trilingual education **?** (Jonbekova et al., 2022; Dzhanegizova, 2024).

Backend API Services

The backend system uses Laravel (PHP) to develop RESTful API endpoints for authentication along with CRUD operations on alumni profiles and event creation features and alumni directory search functions. The system enables users to obtain secure system access through JWT authentication and RBAC features which regulate access based on assigned roles. Security features implemented for the system follow best practices in data protection which resolve the fragmentation issues in personal data laws as identified by Amirov et al. (2024) in Kazakhstan.

Data Management and Storage

PostgreSQL operates as the relational database engine that brings a normalized schema which delivers long-term scalability while facilitating efficient database queries. The Eloquent ORM component in Laravel helps model how users relate to events and roles through complex databases that achieve distinct data access and business logic separation.

Real-Time Communication with Laravel Reverb

Laravel Reverb functions as the real-time communication system which allows live notifications, RSVP updates, and chat capabilities through its WebSocket-based features. This component establishes the basis for dynamic interaction to transform passive alumni supporters into active community members according to Iskhakova, Hilbert, and Hoffmann (2016).

Infrastructure and Deployment

Through Docker and Docker Compose deployment developers achieve easy implementation and running of frontend and backend operations and PostgreSQL database through separate isolated areas. NGINX functions as a reverse proxy that handles the traffic flow management for frontend and backend section and WebSocket connections. The design of this system follows the strategic approach to education system scalability which nations transitioning into digital education need according to Dzhanegizova (2024). The system meets the requirements of Kazakhstan's Law on Personal Data Protection (2013) through encryption technologies as well as local storage and session authentication procedures. The system features multiple elements that ensure appropriate data management for alumni information and transparent functional processes and international data protection compliance readiness (Amirov et al., 2024). As a user-centered sustainable design AMS assists Nazarbayev University in fulfilling its alumni community development goal

along with national digital transformation activities and international computing-based system design concepts.

2.3 Comparative Analysis of Approaches

Feature	**Legacy Systems (Raut, Sabri)**	**Modern Systems (Sen, Patel)**	**Proposed AMS (This Project)**
Database	MySQL	MySQL/PostgreSQL	PostgreSQL (normalized, scalable)
Language Support	English only	Bilingual in some cases	Trilingual (Kazakh, Russian, English)
Authentication & Security	Basic login	Token-based sessions (some)	JWT + SSL + Laravel Middleware + RBAC + Bcrypt
Accessibility	Web only	Limited responsiveness	Fully responsive UI (mobile & desktop)
Event & Communication Tools	Manual notifications	Email + Forum support	Event RSVP + Live Messaging (WebSocket) + Feedback support
Compliance with Legal Standards	Not emphasized	Partial	Compliant with Kazakhstan's Law on Personal Data Protection (2013)
Scalability	Limited to hundreds	Varies	Dockerized architecture, NGINX load balancing, 1,000+ concurrent users supported

Table 1: Comparative Analysis of Alumni Management Approaches

The proposed AMS improves upon existing systems by integrating role-based access, token-based authentication, real-time event notifications, and full multilingual support, all within a secure and scalable architecture built using modern tools such as React, Laravel, and Docker.

2.4 Justification of Methodology

The methodology selected for this project reflects the need to build a computing-based solution that is both technically robust and user-centric. The team adopts Agile Scrum for incremental delivery, allowing for rapid iterations based on stakeholder feedback. The architecture employs:

- React for a responsive and component-driven front end
- Laravel for a secure and scalable backend with RESTful APIs
- PostgreSQL for a normalized, high-performance relational database
- Docker and CI/CD pipelines for efficient deployment and system orchestration
- The integration of security, scalability, and multilingual inclusivity aligns with the current state of research and best practices in software engineering for educational systems (Shneiderman, 2020).

The integration of security, scalability, and multilingual inclusivity aligns with the current state of research and best practices in software engineering for educational systems (Shneiderman, 2020).

3 Project Approach

3.1 Solution Overview

The proposed Alumni Management System (AMS) is a web-based platform designed to centralize alumni data, streamline communication, support event planning, and facilitate networking among graduates of Nazarbayev University. The system's architecture, design principles, and development workflows follow modern full-stack web development practices, ensuring scalability, security, and maintainability.

3.2 System Architecture

The Alumni Management System (AMS) follows a multi-tier client-server architecture consisting of distinct layers for presentation, application logic, data storage, real-time interaction, and deployment infrastructure.

Frontend (Client-Side)

Technology: React, React Router, Axios

Design: Component-based, responsive layout, multilingual UI (Kazakh, Russian, English)

Function:

- Renders dynamic content and handles user interaction
- Uses Axios to make HTTP requests to the backend API
- Manages routes via React Router
- Supports login, registration, password reset, profile updates, and RSVP
- Validates user input client-side and displays error messages
- Manages JWT tokens in `localStorage` for secured access

Backend (Server-Side)

Technology: Laravel (PHP)

Structure: RESTful APIs

Function:

- Handles user authentication and authorization via JWT (using Laravel Sanctum)
- Manages CRUD operations for alumni, events, users
- Middleware protects API routes with role checks

Security:

- JWT for stateless session management
- Passwords hashed using `bcrypt`
- SSL encryption
- Role-based access control (RBAC) for different user permissions

Database Layer:

Technology: PostgreSQL

Structure: Laravel Eloquent ORM

Design:

- Normalized relational schema with indexing
- Stores data for users, events, alumni profiles, RSVPs, and roles
- Audit trail through timestamps and createdby /updatedby tracking

Real-Time Communication Layer

Technology: Laravel Reverb + WebSocket

Function:

- Enables a real-time chat between users of the application
- Authenticated users connect to private WebSocket channels
- Designed for scalability with token-secured broadcasting

Infrastructure

- **Containerization:** Docker + Docker Compose
- **Reverse Proxy:** NGINX configured to route API, frontend, and WebSocket traffic
- **Security:** SSL certificate issued on the host with Let's Encrypt
- **Hosting:** Linux server environment (development and deployment)
- **Environment:** Managed via `.env` files per service
- **CI/CD:** Pipeline-ready for integration; manual deployment used currently

C. Algorithms and Workflows

Authentication Workflow

- User submits login credentials
- Backend (Laravel) validates credentials
- JWT token issued on success
- Frontend stores JWT in `localStorage`
- JWT included in HTTP headers for protected endpoints
- Role-based access enforced on both frontend and backend

Alumni Profile Management

- Admin fills out alumni creation form
- React validates input and submits via Axios
- Laravel validates again, processes request
- Data stored in PostgreSQL
- API returns result — UI updated accordingly

Event Management Workflow

- Admin creates event via dashboard
- Event saved to database with associated details
- RSVP links generated and displayed to users
- Alumni RSVP via dashboard → tracked in DB
- Admin accesses feedback and attendance via dashboard

Feature	Description
User Roles & RBAC	Implemented via Laravel middleware; roles include admin and alumni. Role-based access controls API route usage and frontend component visibility.
Profile CRUD	Alumni can update their personal profile; admins can manage user accounts. Implemented using RESTful routes and controlled frontend forms.
Event Management	Admins can create, update, and delete events. Users can RSVP to events. Functionality integrated using REST APIs.
Search & Directory	Users can browse and filter alumni by fields such as major, location, and graduation year. Search is implemented in backend controllers and queried from frontend.
Secure Communication	JWT is used for authentication; passwords are hashed with `bcrypt`. API routes are protected via middleware; HTTPS enforced by NGINX.
Multilanguage Support	Frontend includes localization files for Kazakh, Russian, and English. Users can toggle language preference.
Responsive Design	UI is styled using CSS and media queries; optimized across desktop and mobile views.
Real-time Messaging	Using a WebSocket connection, users can communicate in real-time.

Table 2: Major Features Implemented in the Alumni Management System

Third-Party Components and Integration

These tools were not implemented from scratch but were selected based on industry best practices, documented protocols, and compliance with security and performance standards.

- **React:** Frontend library for building the UI.
- **React Router:** Library for client-side routing in React.
- **Axios:** Library for making HTTP requests from the browser.
- **Laravel:** PHP framework for the backend API.
- **PostgreSQL:** Relational database system.

- **Laravel Sanctum:** Laravel package for simple API token authentication (used for JWT).
- **Laravel Reverb:** Laravel package for WebSocket communication.
- **Docker & Docker Compose:** Tools for containerization and orchestration.
- **NGINX:** Web server and reverse proxy.
- **bcrypt:** Password hashing algorithm (implementation provided by Laravel).

Project Team Roles and Collaboration

The project team adopted an Agile Scrum methodology with bi-weekly sprints managed in Jira. Roles were distributed according to skills:

Team Member	**Responsibilities**
Temirlan Zaikenov	Frontend developer, UI/UX design
Elvina Mynzhassar	Frontend developer, internationalization for the application
Sanzhar Ospanov	Backend developer, database design, security implementation
Ayaulym Mukan	API integration, database optimization
Aigun Bolganbayev	Project management, market research, DevOps setup

Table 3: Team Responsibilities and Role Distribution

The team conducted regular stakeholder interviews, code reviews via GitHub, and sprint retrospectives to refine workflows and integrate feedback effectively. Communication channels like Telegram and Google Meet were used for real-time coordination.

4 Project Execution

4.1 Timeline and Development Phases

Over the two semesters, the project progressed through iterative development guided by Agile Scrum methodology with bi-weekly sprints, each including planning, develop-

ment, testing, and review phases. The project execution was divided as follows:

Project Timeline

- **Fall 2024 Semester:**
  - Stakeholder interviews and requirements gathering
  - Use case modeling and system architecture design
  - Selection of tech stack: React, Laravel, PostgreSQL, Docker
  - Partial implementation of backend (authentication, CRUD APIs)
  - Initial UI design (login, register, profile setup)
- **Spring 2025 Semester:**
  - Challenges: unfamiliarity with Laravel; overcome via peer learning, documentation, and internal training
  - Expanded API development and frontend-backend integration
  - Docker setup, NGINX configuration and deployment with Docker Compose
  - Focus on multilingual implementation and event management
  - Ongoing: API testing, system integration, performance tuning

Organizations used Agile Scrum methodology structured into two-week sprints to develop the Alumni Management System (AMS). Task management occurred through Jira whereas documentation and sprint planning functions relied on Notion and daily communication utilized Telegram. A team collaboration through Google Meet took place every week for reviewing progress and task reassignments together with resolving blocker issues.

The first stage involved examining what Nazarbayev University required from its alumni department along with reviewing features of existing comparable systems. We established the minimum viable product (MVP), developed wireframes for essential components including login, profile management and event dashboards as well as defined user permissions structures and access during Sprint 1.

Team members received their roles through a system of skillset distribution. The DevOps role selected for the tasks involved Docker implementation alongside PostgreSQL

and NGINX configuration. Meanwhile the backend developer handled database operations and two members handled frontend development using React. Through my position I coordinated between components while designing the APIs which integrated backend and frontend operations.

At the project start none of the team members had experience with Laravel or PHP backend frameworks. The team spent the first sprint for education purposes. The team learned the Laravel syntax and conventions as well as backend fundamentals like routing, authentication and MVC patterns and database normalization during a period of one week. The team's onboarding process became faster due to lead mentors Temirlan and Sanzhar who devoted themselves to mentoring new members. The team members assisted others while reviewing backend code and debugging problems and provided instruction on reading and modifying and structuring Laravel code. Our team distributed work more effectively because of their guidance which enhanced our speed as a unit.

The project adopted Laravel Sanctum because it provided better functionality for single-page applications relatively to Passport although it required heavier solutions. The easier token administration in Sanctum combined with its Laravel middleware compatibility facilitated RBAC implementation. Route access depended on user roles which included admin, alumni officer and alumni. The system used bcrypt hashing to store all passwords.

The configuration of NGINX with SSL provided users with complete HTTPS encryption for the application throughout. API protection was achieved through Laravel middleware while frontend-backend security required proper CORS policy implementation.

The frontend was constructed using component-based architecture and React.js. To handle clint-side navigation we used React Router. Axios was included for backend API communication. In the first few sprints, we created the interface using JavaScript; however, as the system became more complex, we switched to TypeScript in Sprint 2. This decision was motivated by the need for strict type safety when handling props and API responses. TypeScript helped us detect errors early in the development process and gave us stronger guarantees, especially when working with different user roles and handling forms. This migration made the system more maintainable and easier to scale.

Management of deployment was through Docker and Docker Compose that helped us

to package services (frontend, backend, and database) into separate containers. Traffic was routed to static assets and APIs using NGINX as a reverse proxy. Time constraints prevented us from implementing GitLab CI/CD, but the project was set up to facilitate simple CI/CD integration in the future.

We also integrated Laravel Reverb for real-time features using WebSockets. While full notification support was not finalized, we successfully configured broadcasting channels, secured them via Sanctum tokens, and established live connections with authenticated clients. This laid the groundwork for future real-time updates, such as RSVP notifications and messaging.

One of the biggest technical problems happened during integration stage. Backend field names were inconsistently mapped in the frontend (e.g., alumniId vs userId), causing form submission and rendering errors. To solve this, we created a shared API contract in Notion and standardized response formats, which reduced frontend/backend mismatch and improved API stability.

4.2 Project Management Reflection

This project served as an excellent opportunity for project managers to understand actual software coordination in practice. Backlogs went through refinement and stories got estimated before each sprint began. We started development with underestimated authentication complexities during the early stages which improved into better velocity prediction and smaller story breakdowns by Sprint 3. The project achieved success by shifting its main priority to deliver the Minimum Viable Product rather than perfecting all functionalities at once. The working-first perspective allowed our team to complete our tasks on schedule while upholding the system's necessary structural requirements.

In the team structure Sanzhar and Temirlan were responsible for coordinating technical tasks between members. Through their guidance team members received their assignments while pairing with peers in a supportive atmosphere for asking questions. The team members helped eliminate blockages particularly when working on complex backend operations such as user roles and authentication procedures.

Beyond exposing us to full-stack development we also gained expertise in project planning alongside technical mentorship skills which combined with communication abilities and iterative design methods that any professional software engineer needs.

Current alumni management worldwide depends heavily on digital infrastructure as its effective role in maintaining alumni relationships continues to expand. The centralized platform known as Alumni Management Systems (AMSs) enables educational institutions to keep track of graduates while also managing their activities and encouraging continued institutional relationship. The management of alumni records in institutions before the new millennium mainly depended on individual tools such as static websites and standalone databases and Microsoft Excel spreadsheet usage (Sabri et al., 2017; Raut et al., 2019). Despite their limited success in basic and occasional contacts they proved inadequate to scale up or ensure safety as well as real-time operation.

The advancement of web technologies as well as security protocols enables contemporary AMS platforms which integrate responsive design with API-first structure while offering multilingual support and role-based authentication systems. Chaturvedi (2022) describes token-based authentication systems using JSON Web Tokens (JWT) as the most effective solution for stateless authentication in web applications. The system decreases server strain and provides safe session preservation and scales better when operating on individual data sets and multiple user roles.

Database and data models need proper normalization in addition to backend structural integrity to become robust. According to Samuel (2024) clean database design stands essential for both redundancy reduction and system reliability enhancement. The relational schema design practices of index optimization and data integrity constraints established by Samuel demonstrate congruity to our project that utilizes PostgreSQL and Eloquent ORM to perform CRUD operations.

Modern web applications rely on containerization and load balancing techniques as standard procedures for sustaining system stability in their infrastructure layer. Chyrvon, Lisovskyi, and Kyryndas (2023) demonstrate NGINX functionality beyond its standard reverse proxy role by showing its application as a networking component which distributes workload for services aimed at high-traffic system performance. Our decision to use isolated containers of AMS frontend and backend services with WebSocket alongside NGINX as reverse proxy and load balancer finds validation through these research findings for future high-usage reliability.

Historical difficulties in institution-alumni relationships in Kazakhstan and post-Soviet countries stem from structural and cultural barriers according to Sagitova (2018). Jon-

bekova et al. (2022) revealed that research demonstrates increased understanding about how alumni relations can develop strategic partnerships for mentorship in addition to enhancing visibility and capacity-building programs. Modern alumni systems must be purpose-designed because they must handle both administrative operations and enable networking and peer-based connections and long-term engagement.

According to Iskhakova et al. (2016), alumni loyalty needs information-based solutions and user participation elements which requires platforms to advance from basic directory functions to interactive value-added features. The introduction of several digital systems in Central Asia addresses student networking deficits but their technical design fails to incorporate RGB permissions as well as fails to provide multi-language capabilities and national privacy regulations enforcement (Amirov et al., 2024). Laravel Reverb stands as one of the few real-time WebSocket features in digital systems which enables live event notifications and system updates.

The introduced AMS solution implements international industry specifications to address requirements for specific countries. The platform utilizes JWT authentication and PostgreSQL data models which are normalized and NGINX-based modular containerization and real-time and multilingual capabilities to deliver sustainable alumni engagement solutions to Nazarbayev University.

Design Decisions and Adaptations

- **Technology stack revision:** Integrated Laravel and PostgreSQL within Docker Containers. Improved documentation, and easier integration for token-based security.
- **Database schema design:** Iteratively refined to normalize data and reduce redundancy; guided by actual use cases and stakeholder input.
- **Security-first approach:** SSL encryption, JWT tokens, role-based access control, and password hashing were implemented early to prevent later architectural issues.
- **Responsiveness prioritization:** React's component-based design helped achieve a fully responsive frontend, tested across mobile and desktop.

Teamwork and Collaboration

- The team functioned collaboratively, dividing responsibilities according to strengths:
  - Temirlan was responsible for frontend development and user interface (UI) design.
  - Elvina focused on React development and implementing the multilingual user interface.
  - Sanzhar handled backend API design and database modeling, implementation of security features, deployment using Docker and NGINX.
  - Ayaulym contributed to backend integration and system testing.
  - Aigun led the general management, market research, requirements, and building front end of the project.

Collaboration tools: Jira for project tracking, GitHub for version control, Notion for documentation, Telegram/Google Meet for daily communications.

Problem-solving: The team conducted code reviews, weekly retrospectives, and design workshops to resolve blockers.

Leadership: Sanzhar and Aigun led the backend and DevOps components; Temirlan coordinated frontend design decisions.

5 Evaluation

Evaluation Goals and Strategy

To validate whether the Alumni Management System (AMS) addresses the core problem of inefficient alumni data management and fragmented engagement, the team conducted a multi-phase evaluation focused on:

1. **Usability and Functionality Feedback**
2. **Stakeholder Review**
3. **User Experience Surveys**

This process was distinct from normal QA testing and focused on assessing real-world effectiveness and acceptance.

Methods Used

- **Participants:** The system was demonstrated to a focus group of people including alumni and university administrators.
- **Task-Based Scenarios:** Users were asked to complete tasks such as registering, updating profiles, registering for events, and viewing other alumni.
- **Feedback Forms:** Users rated ease of use, visual clarity, and usefulness on a 5-point Likert scale.

Evaluation Data Collected

Evaluation Metric	**Result Summary**
Ease of Profile Management	4.6 / 5 – Intuitive layout and clear navigation
Event Participation Tools	4.7 / 5 – RSVP and reminders worked smoothly
Multilingual Interface	4.5 / 5 – Language toggling seamless
Responsiveness and Compatibility	4.6 / 5 – Worked well on phones and desktops

Table 4: Summary of Evaluation Results Based on User Feedback

Feedback and Insights

- Multilingual support was noted as a standout feature, particularly for Kazakh-speaking users.
- Some alumni suggested LinkedIn-style career filters in the directory, which the team may include in future versions.
- Alumni officers emphasized the need for bulk-upload tools, which are planned post-launch.

Validation of the Computing-Based Solution

The evaluation clearly shows that the project achieved its core objectives:

- It centralized alumni data and improved accessibility.
- It replaced the need for Excel-based manual updates.
- It enabled secure user role segregation and data privacy controls.
- It provided an engaging and intuitive interface with cross-device support.

These results validate the computing-based solution's effectiveness in solving the initial problem and demonstrate its potential for real-world deployment and scaling.

6 Conclusion and Possible Future Work

6.1 Conclusion

The Alumni Management System (AMS) developed by the group introduces an efficient web-based computing-based solution which tackles Nazarbayev University's problem of outdated and ineffectual alumni data management. The adoption of a secure web-based system replaces manual spreadsheet usage to enhance alumni participation while improving database quality and simplifying event planning. The team began by learning Laravel from scratch. A sprint was dedicated to backend training. Leads Temirlan and Sanzhar mentored members and distributed tasks based on roles. Backend and frontend were developed in parallel, then integrated with Reverb and NGINX.

Key achievements include:

- Implementation of a centralized PostgreSQL database with secure CRUD operations.
- Design of a multilingual, mobile-responsive frontend using React.
- Development of secure REST APIs using Laravel with JWT-based authentication and role-based access control.
- Integration of event management, alumni directory search, and real-time messaging.

Through stakeholder engagement, agile development, and rigorous evaluation, the project demonstrated its real-world applicability, confirming its alignment with user needs and institutional objectives.

6.2 Future Work

While the current implementation of the Alumni Management System (AMS) lays a strong technical and functional foundation, several enhancements are planned for future iterations to expand its capabilities, enrich the user experience, and increase institutional value.

AI-Powered Networking:

We plan to incorporate machine learning-based recommendation systems to facilitate smarter networking among alumni. By analyzing shared attributes such as graduation year, industry, department, and location, the platform could suggest personalized connections — similar to the "People You May Know" feature in professional networks like LinkedIn. This would encourage more peer-to-peer engagement and support organic community building.

Native Mobile Application

To improve accessibility and ensure continuous engagement, we aim to develop native mobile applications for both iOS and Android platforms. These apps will include essential AMS functionality, offline access to key features, and real-time push notifications for event updates, invitations, or direct messages.

Advanced Analytics Dashboard

An administrator-facing dashboard is proposed to deliver insights into alumni behavior and engagement. Key metrics could include event attendance rates, login frequency, profile completion stats, and communication effectiveness. These analytics will support data-driven decision-making and strategic outreach planning by university staff.

Bulk Import and Data Migration Tools

To simplify the onboarding of historical data, we intend to build tools that allow bulk importing of alumni records from CSV files or other legacy databases. This feature will support institutions transitioning from non-digital or semi-structured systems and ensure a smoother migration process.

Integration with External Platforms

Future versions of AMS may include integrations with professional networks like LinkedIn or career platforms via public APIs. This would enable automatic enrichment of alumni profiles with job titles, company affiliations, or recent activity — further enhancing the directory's relevance and usefulness.

Gamification Features

To increase user engagement, we propose the addition of gamification elements such as digital badges, milestone tracking, or leaderboards. These could be awarded for profile completion, event attendance, years since graduation, or community contributions, incentivizing repeat use and interaction.

By extending these capabilities, AMS can evolve into a fully institutionalized, multi-departmental tool for long-term alumni engagement. Each enhancement contributes toward making the system not only more functional but also more strategic — aligning with the university's broader goals for community building, digitalization, and data-informed outreach.

7 References

Barnard, Z., & Rensleigh, C. (2008). Investigating online community portals for enhanced alumni networking. *The Electronic Library, 26*(4), 433–445.

Cornea, S. (2024). *Perspectivele și problemele integrării în spațiul european al cercetării și educației*. Volume XI, Partea II.

Dzhanegizova, A. (2024). Digital transformation of higher education in Kazakhstan: Challenges and solutions. *Economic Annals-XXI, 209*(5–6), 42–55.

Iskhakova, L., Hilbert, A., & Hoffmann, S. (2016). An integrative model of alumni loyalty—An empirical validation among graduates from German and Russian universities. *Journal of Nonprofit & Public Sector Marketing, 28*(2).

Jonbekova, D., Serkova, Y., Mazbulova, Z., Jumakulov, Z., & Ruby, A. (2022). How international higher education graduates contribute to their home country: An example from government scholarship recipients in Kazakhstan. *Higher Education Research & Development, 42*(1), 126–140.

Kazakhstan Law "On Personal Data and Its Protection". (2013). Law No. 94-V of May

21, 2013.

Kumar, A., Patwal, A., Joshi, K., Pant, S., & Jeena, A. (2024). Comprehensive management system for placement cell operations. *Journal of Data Processing and Business Analytics*.

Patel, M., Rami, D., & Soni, M. (2020). Next generation web for alumni web portal. In *ICICV 2019: Intelligent Communication Technologies and Virtual Mobile Networks*. Lecture Notes on Data Engineering and Communications Technologies, Vol. 33.

Raut, M. R., Lokhande, T. P., Godbole, K. D., More, S. J., Hatmode, S. S., & Tibude, N. D. (2019). PCE staff/student portal. *International Journal of Computer Science Trends and Technology, 7*(1), 17–22.

Sabri, S. Q., Ahmad, A. M., & Abdulrazaq, M. B. (2017). Design and implementation of student and alumni web portal. *SJUOZ, 5*(3), 272–277.

Sagitova, R. (2018). Alumni associations in Kazakhstan: Building the future of higher education institutions through its graduates. *Comparative and International Education, 47*(3), 88–97.

Sen, S. K., & Pookayaporn, J. (2022). Towards a SDG compliant framework for the open learning modules. In *Research Anthology on Measuring and Achieving Sustainable Development Goals*.

Shneiderman, B. (2020). *Designing the user interface: Strategies for effective human-computer interaction* (6th ed.). Pearson.

Amirov, A., Kainazarova, D., Begaliyev, E., Sarsenbaev, A., & Sarybayev, N. (2024). Legal ways and methods of personal data protection in Kazakhstan. *Scientific Herald of Uzhhorod University (Physics Series)*, (55), 2174–2186.