

# Table of Contents

1. Project Description .....	3
1.1 Purpose .....	3
1.2 Scope .....	3
2. Needs and Goals .....	3
2.1 Business Needs .....	3
2.2 Goals and Objectives .....	4
2.3 Long-term Goals .....	4
3. Technological Requirements .....	4
4. Stakeholders and User Overview .....	4
5. User Personas .....	5
6. Constraints .....	5
7. Functional Requirements .....	6
7.1 Registration and Login Module .....	6
7.2 Input Module .....	6
7.3 Recommendation Module .....	6
7.4 Course and Instructor Search Module .....	6
7.5 User Interaction & Customization .....	6
7.6 Material Exchange Module .....	7
7.7 Reports and Analytics .....	7
8. Non-functional Requirements .....	7
8.1 Performance Requirements .....	7
8.2 Reliability Requirements .....	7
8.3 Security and Privacy Requirements .....	7
8.4 Usability & Accessibility Requirements .....	8
8.5 Legal & Compliance Requirements .....	8

<b>9. Use Cases .....</b>	<b>8</b>
9.1 Use Case 1: Student Registration and Course Data Entry .....	8
9.2 Use Case 2: AI-Based Course Recommendation .....	8
9.3 Use Case 3: Searching for Courses and Reading Reviews .....	9
9.4 Use Case 4: Searching for Study Materials .....	9
9.5 Use Case 5: Searching for a Professor and Viewing Course Assignments .....	10
 <b>10. Mockups .....</b>	 <b>11</b>
 <b>11. Hosting .....</b>	 <b>13</b>
11.1 Hosting Provider .....	13
11.2 Hosting Setup .....	14
11.3 Deployment Strategy .....	14
11.4 Security Measures .....	14

## **Project Description**

Many students struggle with selecting courses that align with their academic strengths and future career goals. This challenge is further complicated by the overwhelming array of elective courses available, the varying prerequisites, and the pressure to make informed decisions. As a result, students often miss opportunities to tailor their educational experiences, and academic advisors face an increased workload.

### **1.1 Purpose**

The purpose of this document is to define the requirements, scope, and objectives for developing an AI-driven academic subject recommendation system. The system will help university students select elective courses based on their academic history, career goals, and personal interests, leveraging machine learning (ML).

This solution aims to:

- Automate course selection by providing AI-generated recommendations.
- Personalize student academic planning using AI-driven insights.
- Reduce the workload of academic advisors by offering preliminary suggestions.
- Improve decision-making by presenting well-structured course insights.

### **1.2 Scope**

The system will provide:

- A web-based platform where students can log in and receive course recommendations.
- A recommendation engine trained on past student course selections, grades, and trends.
- A feedback loop that continuously improves AI predictions based on student inputs.
- Integration with university course databases to fetch subject prerequisites, descriptions, and credits.
- An interactive dashboard for students to explore course trends and career mappings.

Out of Scope:

- The system does not replace academic advisors but supplements their recommendations.
- The system does not guarantee perfect subject selection but aids decision-making.
- Initially, the system will only support one university; multi-university support will be added in later phases.

## **2. Needs and Goals**

### **2.1 Business Needs**

- Simplified elective selection to reduce confusion among students.
- Enhanced academic planning by aligning electives with career aspirations.
- Data-driven recommendations to improve student satisfaction and retention rates.

2.2 Goals and Objectives

- Develop an AI-powered recommendation engine that predicts the best elective subjects.
- Design an intuitive and engaging UI/UX for easy student interaction.
- Ensure system scalability and reliability for a growing student population.
- Ensure compliance with GDPR and institutional data privacy regulations.

2.3 Long-term Goals

- Expand the system to multiple universities, creating a universal platform that provides insight into course trends across various institutions.
- Develop a recommendation system capable of offering suggestions based on both academic performance and extracurricular activities or skill endorsements.

3. Technological Requirements

The system will require various technological components to function effectively. The following outlines the key technological requirements for developing and maintaining the platform:

- Backend: Java Spring Boot
- Frontend: React
- UI/UX: Figma
- Database: PostgreSQL
- AI/ML Frameworks: OpenAI’s Models
- Web Scraping: Python Script
- Hosting: Microsoft Azure

4. Stakeholders and User Overview

Stakeholder	Role & Responsibilities
Students	Primary users who receive AI-generated recommendations.
Academic Advisors	University faculty members who can verify and refine AI recommendations.

<b>University IT Department</b>	<b>Manages system integration with student databases.</b>
<b>Project Development Team</b>	<b>Engineers, data scientists, and UI/UX designers developing the system.</b>
<b>University Administration</b>	<b>Evaluates system effectiveness and adoption.</b>

## 5. User Personas

To better understand system users, here are two main personas:

Persona 1: The Undergraduate Student

- Name: Sarah (21 years old, 3rd-year student)
- Major: Computer Science
- Goal: Wants to specialize in AI but is unsure which electives to take.
- Pain Points: Finds the university's course catalog confusing and lacks clear guidance.

Persona 2: The Academic Advisor

- Name: Professor James (45 years old, faculty member)
- Department: Engineering
- Goal: Wants to help students select courses efficiently.
- Pain Points: Too many students to advise individually, making it difficult to provide personalized guidance.

## 6. Constraints

- Must comply with university data privacy policies and GDPR regulations.
- Limited access to real-time academic records due to institutional policies.
- The recommendation system should not provide absolute decisions but rather suggestions.
- The recommendation engine's effectiveness will initially be limited by the amount of historical data available, particularly in the early stages of the system's deployment. As more student data is collected, the system's ability to provide accurate and personalized recommendations will improve.

## Functional requirements:

1. Registration and Login Module
  - 1.1. User Profiles
    - 1.1.1. The system should allow students to register by entering basic information.
    - 1.1.2. The system should allow students to log in using email and password.
    - 1.1.3. The system should allow password reset
    - 1.1.4. The system should allow users to edit their profile
2. Input Module
  - 2.1. Data Entry
    - 2.1.1. The system should allow manual input of previously completed courses and grades.
    - 2.1.2. The system should allow input of previously completed courses and grades through an imported file
    - 2.1.3. The system should allow students to enter their academic interests.
3. Recommendation Module
  - 3.1. Generating Recommendations
    - 3.1.1. The system should provide AI-based analysis of the student's academic history.
    - 3.1.2. The system should generate a list of recommended courses based on grades.
    - 3.1.3. The system should generate a list of recommended courses based on interests
    - 3.1.4. The system should generate a list of recommended courses based on past subjects.
    - 3.1.5. The system should provide a short explanation of why a course is recommended.
    - 3.1.6. The system should consider prerequisite requirements before suggesting
4. Course and Instructor Search Module
  - 4.1. Search and Information Review
    - 4.1.1. The system should allow searching for courses by name, code, or keyword.
    - 4.1.2. The system should display basic course information (description, professor, prerequisites, other students experiences)
    - 4.1.3. The system should allow students to share their experiences about a course.
    - 4.1.4. The system should allow searching for professors by name or the course they teach.
5. User interaction & customization
  - 5.1. Filtering recommendations
    - 5.1.1. The system should allow saving recommended subject lists
    - 5.1.2. The system should allow filtering recommended courses

6. Material Exchange Module
  - 6.1. Sharing and Downloading Materials
    - 6.1.1. The system should allow downloading materials.
    - 6.1.2. The system should enable organizing materials by course.
7. Reports and Analytics
  - 7.1. Reports for a subject
    - 7.1.1. The system should generate a report on a subject with information about how many students have taken the subject and their experiences on the subject
  - 7.2. Selection trends
    - 7.2.1. The system should generate a report about selection trends, subjects that have been recommended to most students
8. General Features
  - 8.1. Administrative Control
    - 8.1.1. The system should allow administrators to manage the database of courses and instructors.
    - 8.1.2. The system should allow user account management (blocking, activation).

## Non-functional requirements

1. Performance requirements
  - 1.1. The system should support a minimum of 100 concurrent users, without noticeable performance issues.
  - 1.2. The system should provide a user interface with a maximum response time of 2 seconds for user interactions, excl. AI features.
  - 1.3. AI-based recommendations should be processed within 10 seconds for an optimal user experience.
  - 1.4. The system should have an uptime of at least 99.9% under normal operating conditions.
  - 1.5. File uploads/downloads should complete within 10 seconds for files up to 50 MB under normal network conditions.
2. Reliability requirements
  - 2.1. The system should perform automated backups at least once a week.
  - 2.2. Backups should be stored in at least one physically separate location.
  - 2.3. The system should have automated failover mechanisms to ensure minimal downtime during failures.
  - 2.4. A rollback mechanism should be implemented for reverting to the last stable version in case of deployment issues.
3. Security and privacy requirements
  - 3.1. The system should use the HTTPS protocol for communication.
  - 3.2. Sensitive user data should be stored using AES-256 encryption.

- 3.3. The system should allow the creation of accounts using verified faculty email domains only.
- 3.4. The system should log failed login attempts and enforce account lockout after five failed attempts.
- 4. Usability & Accessibility Requirements
  - 4.1. The system should have an intuitive and user-friendly UI that allows users to perform key actions within three clicks.
  - 4.2. The recommendation feature should present suggestions in a clear, visually appealing format.
  - 4.3. The system should be accessible via desktop, tablet, and smartphone devices using modern web browsers (Chrome, Edge, Firefox, Safari).
  - 4.4. Error messages should be clear and informative, guiding users on how to resolve issues.
  - 4.5. The system should support common file formats for study materials (e.g., PDF, DOCX, PPTX, JPG)
- 5. Legal & Compliance Requirements
  - 5.1. The system should comply with local data protection laws.
  - 5.2. Users must agree to Terms of Service and Privacy Policy before using the platform.

## Use Cases

### Use Case 1: Student Registration and Course Data Entry

**Actors:** Student

**Preconditions:** The student is not registered in the system.

**Flow:**

1. The student accesses the registration page.
2. The student enters personal information (name, email, student ID, etc.).
3. The student selects past courses and enters grades.
4. The student optionally uploads a grade sheet for automatic data extraction.
5. The student specifies their current semester and academic interests.
6. The system stores the entered data.
7. The student completes the registration process.

**Postconditions:** The student is registered, and their academic history is stored in the system.

---

### Use Case 2: AI-Based Course Recommendation



**Actors:** Student, AI Model

**Preconditions:** The student has completed the registration process and entered past academic data.

**Flow:**

1. The student logs into the system.
2. The student requests course recommendations.
3. The AI model analyzes past courses, grades, interests, and elective prerequisites.
4. The system generates a list of recommended courses.
5. The student reviews and selects preferred courses.

**Postconditions:** The student receives personalized course recommendations.

---

### **Use Case 3: Searching for Courses and Reading Reviews**

**Actors:** Student

**Preconditions:** The student is logged into the system.

**Flow:**

1. The student searches for a specific course.
2. The system displays course details, including prerequisites, syllabus, and instructor information.
3. The student views aggregated reviews and summarized feedback from past students (AI-based summary).
4. The student can add their own review and rating for a course they have taken (optional use-case).

**Postconditions:** The student gains insights into the course and can contribute feedback.

---

### **Use Case 4: Searching for Study Materials**

**Actors:** Student

**Preconditions:** The student is logged into the system.

**Flow:**

1. The student searches for a specific course.
2. The system displays available study materials.
3. The student can upload additional study materials.
4. The student can download materials for study purposes.

**Postconditions:** The student can access or contribute to study materials for the course.

---

## **Use Case 5: Searching for a Professor and Viewing Course Assignments**

**Actors:** Student

**Preconditions:** The student is logged into the system.

**Flow:**

1. The student searches for a professor by name.
2. The system retrieves and displays the professor's details, including courses taught and student reviews.
3. The student can view information on the professor's teaching style, course difficulty, and expectations.

**Postconditions:** The student gains insights into the professor and their courses.

# Mockups

## LOG IN

Welcome back! Please enter your details.


**Email**

**Password**

☐ Remember me [Forgot password](#)

Log in

Don't have an account? [Sign up for free!](#)




## FORGOTTEN PASSWORD

Enter your email and you will receive a code

**Email**

Next

Don't have an account? [Sign up for free!](#)




## FORGOTTEN PASSWORD

Enter the code you received

**Code**


Next

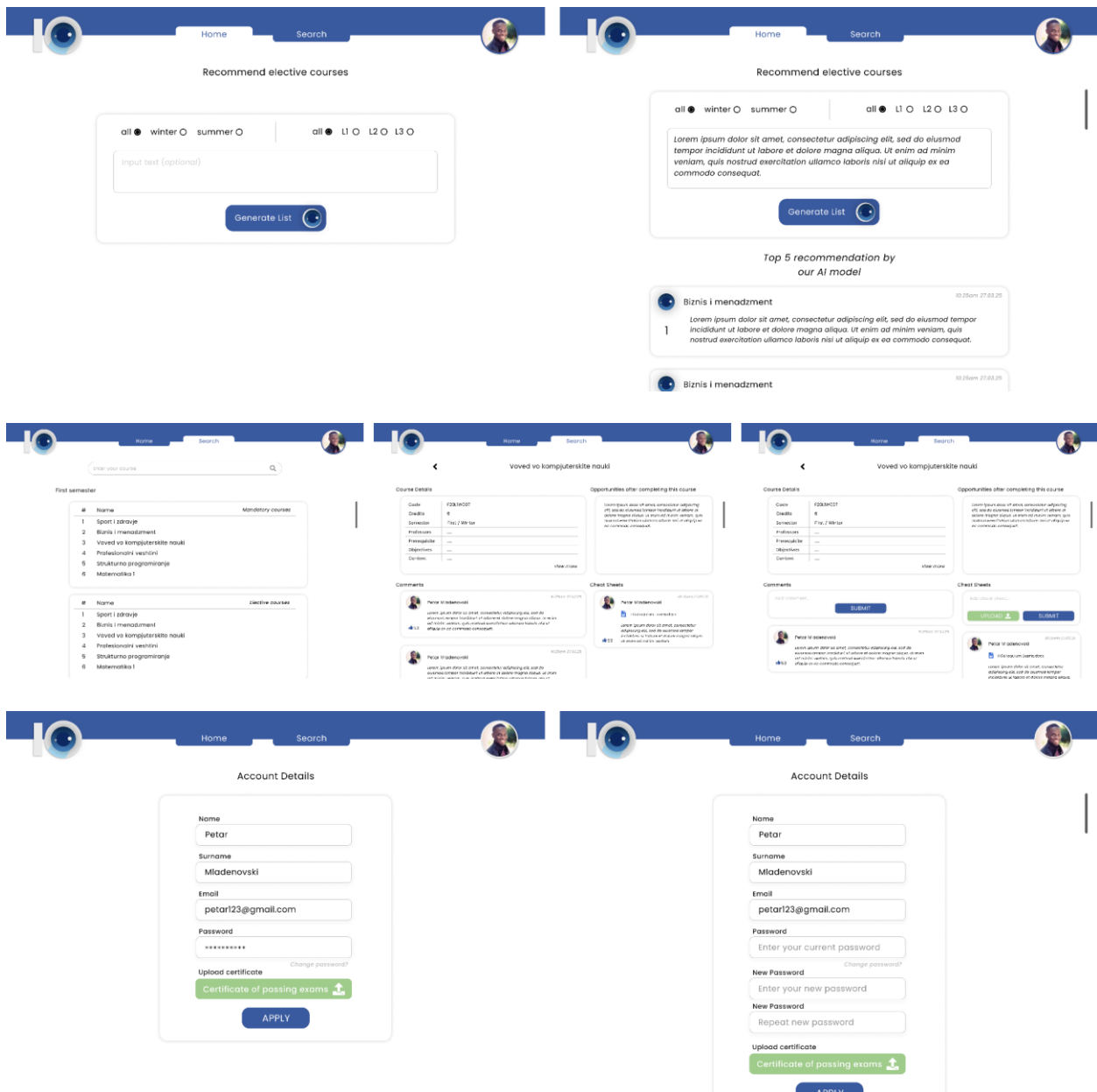
Don't have an account? [Sign up for free!](#)





#	Name	Type	Grade
1	Sport i zdravje	M	10
2	Biznis i menadžment	M	10
3	Voved vo kompjuterskite nauki	M	10

Recommend elective courses




# Hosting

## 1. Hosting Provider

The project will be hosted on **Microsoft Azure**, utilizing its cloud services for simplicity.

## 2. Hosting Setup

- 2.1. Frontend: Deployed on **Azure App Service** for easy hosting.
- 2.2. Backend: Hosted on **Azure App Service** or Azure Kubernetes Services if containerization is required.
- 2.3. Database: Stored in **Azure Database for PostgreSQL** for a fully managed relational database solution.
- 2.4. Storage: **Azure Blob Storage** for static assets if needed.

## 3. Deployment Strategy

- 3.1. Manual Deployment: The application will be deployed manually via Azure Portal.
- 3.2. Version Control: The codebase will be stored in GitHub, with updates manually deployed to Azure.

## 4. Security Measures

- 4.1. Basic Authentication: Simple authentication methods will be used.
- 4.2. HTTPS: Free SSL via Azure App Service Managed Certificates.
- 4.3. Database Security: PostgreSQL database secured with Azure's built-in firewall.
- 4.4. DDoS Protection: Azure DDoS Protection enabled for mitigating attacks.