I.J. Modern Education and Computer Science, **2025, 4, 26-44**
Published Online on August 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2025.04.02

Fuzzy Intelligent System for Student Software Project Evaluation

Anna Ogorodova
Kazakh-British Technical University, School of Information Technology and Engineering, Almaty, 050000, Kazakhstan
E-mail: an_ogorodova@kbtu.kz
ORCID ID: https://orcid.org/0009-0000-7907-9695

Pakizar Shamoi*
Kazakh-British Technical University, School of Information Technology and Engineering, Almaty, 050000, Kazakhstan
E-mail: p.shamoi@kbtu.kz
ORCID iD: https://orcid.org/0000-0001-9682-0203
*Corresponding Author

Aron Karatayev
Kazakh-British Technical University, School of Information Technology and Engineering, Almaty, 050000, Kazakhstan
E-mail: ar_karatayev@kbtu.kz
ORCID iD: https://orcid.org/0009-0000-3824-3065

Received: 01 May, 2024; Revised: 22 June, 2024; Accepted: 16 March, 2025; Published: 08 August, 2025

Abstract: Developing software projects allows students to put knowledge into practice and gain teamwork skills. However, assessing student performance in project-oriented courses poses significant challenges, particularly as class sizes increase. This paper introduces a fuzzy intelligent system designed to evaluate academic software projects using an object-oriented programming and design course as an example. Our methodology involved conducting a survey of student project teams (n=31) and faculty (n=3) to identify key evaluation parameters and their applicable ranges. The critical criteria—clean code, use of inheritance, and functionality—were represented as fuzzy variables with corresponding fuzzy sets. We collaborated with three experts, including one professor and two course instructors, to define a set of fuzzy rules for a fuzzy inference system. This system processes the input criteria to produce a quantifiable measure of project success. Our fuzzy intelligent system demonstrated promising results in automating project evaluation, standardizing assessments, and reducing subjective bias in manual grading. The key findings show that the system effectively manages the increasing instructor workload, provides consistent and transparent evaluations, and offers timely and accurate feedback to students.

Index Terms: Fuzzy Sets and Logic, Software Project Evaluation, Student Performance, Automated Grading, Object-Oriented Programming

1. Introduction

Academic software projects are essential for information technology students to gain hands-on experience, real-world applications, teamwork, and portfolio building. Educational institutions, especially technical universities, often offer courses that teach programming. These courses require students to undertake projects like developing code to solve specific problems. While courses like Algorithms and Data Structures or Fundamentals of Programming might use automated assessments through input and output files, evaluating student performance in project-oriented courses like Object-oriented programming is more complex due to the diverse nature of project work. Therefore, given the importance of software projects in an academic environment, evaluation and feedback from teachers are increasingly taking a key position in education [1].

The integration of AI in education is a rapidly evolving field, with a focus on personalized learning and educational technology [2, 3]. The current trends in AI-based educational processes include adaptive personalization systems, intelligent tutoring systems, assessment and evaluation of students' outcomes, and learning analytics [4]. The key technologies of Educational AI include knowledge representation, machine learning, deep learning, natural language processing, intelligent agents, and affective computing [5]. In general, AI has a lot of potential applications in education, particularly in tutoring, assessment, and personalization [6]. Another important feature of AI in education is the ability to grade students automatically [6]. Providing students with timely and accurate feedback through qualitative assessment improves their learning in a higher education setting [7].

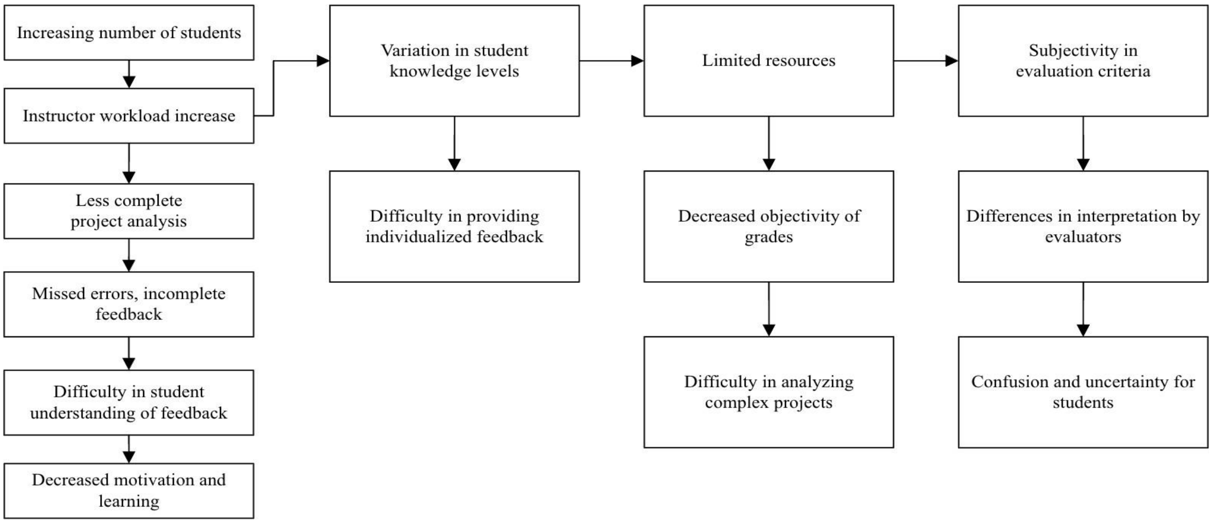


Fig. 1. Flowchart depicting the challenges of evaluating academic software projects

Assessing student performance in project-oriented courses is a complex task that demands careful attention. As the number of students in technical programs increases, the instructor's workload in checking student software projects also increases. This is due to time constraints, varying levels of student knowledge, limited resources, and the complexity of the projects (see Fig. 1). The instructor's overload can lead to less complete project analysis, missed errors, subjective grading, or incomplete feedback.

The next problem that may arise when evaluating academic projects is limited resources. As the number of students increases, the instructor may need more resources to validate software projects, such as grading software or more teaching assistants. Assuming each project takes 30 minutes, an instructor would need about 150 hours to review 300 projects. The teacher may also need additional time to provide feedback, assign grades, and communicate with students about their projects. In addition, with an increased number of students, the instructor will likely encounter a broader range of comprehension and skill levels [8], which makes it difficult to provide individualized feedback appropriate to each student's level of understanding and skill. Delays in feedback can lead to decreased engagement, motivation, and increased anxiety among students, significantly hindering their learning experience [9].

The problem of software project estimation is a complex issue requiring careful consideration of many factors [10]. Software projects may sometimes need clear objectives that can be easily measured. In an academic context, success may be defined differently, depending on the project's goals. For example, a program project may be evaluated on its technical merit, the student group's work, communication skills, and knowledge of theoretical material. Each of these factors requires its own set of evaluation criteria, and they can be challenging to measure objectively. Evaluating a software project can be subjective, as instructors may have different opinions about what constitutes success or failure [11]. For example, students may consider a project a success if it meets technical requirements. In contrast, faculty may consider it a success if all team members are equally involved in its development. Particular attention should be paid to another significant problem, evaluation uncertainty, which arises when more than one expert or evaluator is involved in the evaluation process. Differences in understanding assessment criteria and subjective views can lead to differences in final grades, creating confusion and uncertainty for students. It is crucial to develop evaluation criteria consistent with the project's goals and consider the context of the academic environment.

In this context, developing a fuzzy intelligent system for evaluating student software projects can address these problems. Such a system can automate the evaluation process, making it more objective, consistent, and transparent, a kind of evaluation assistant system (see Fig. 2). Developing software is by its nature imprecise [12]. The motivation for using Fuzzy Logic in this problem is that it can handle ambiguity and uncertainty, incorporate expert knowledge, and combine multiple criteria into an assessment [13]. In addition, fuzzy logic will allow the system to better adapt to the diversity of student projects.

Developing a fuzzy intelligent system is highly relevant to current trends in educational technology and AI applications. With increasing class sizes and the growing complexity of project-oriented courses, traditional student evaluation methods are becoming increasingly impractical. The integration of AI in education has the potential to revolutionize the learning experience, making it more personalized. Using fuzzy logic in educational assessments addresses several critical issues in the field. For instance, it allows for handling ambiguity and uncertainty in project evaluations, which are common challenges in subjective assessments. This aligns with the broader trend of utilizing AI to improve decision-making processes and quality measurement in education.

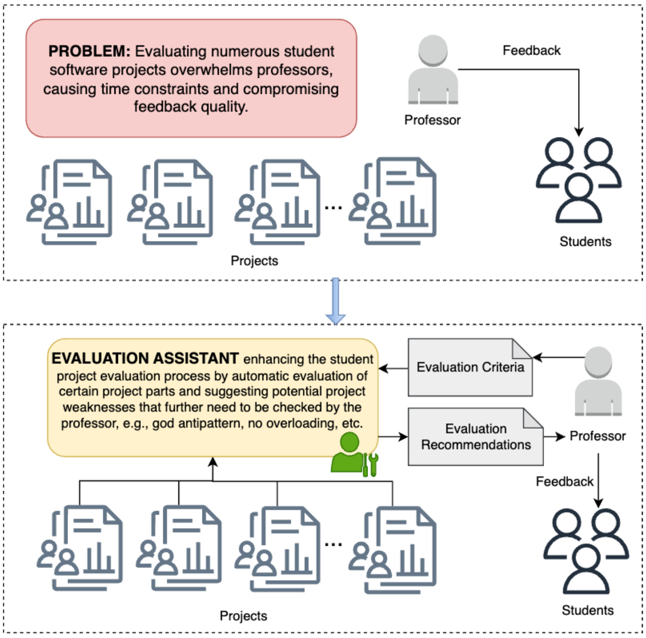


Fig. 2. Idea of evaluation assistant

This paper introduces an evaluation model for software development projects that utilize fuzzy logic to address the uncertainty resulting from human subjective perception during decision-making. The main contributions of this study are:

- Identifying the critical criteria for evaluating academic software projects based on surveys and real academic projects. Surveys were conducted with student project teams (n=31) and faculty (n=3) to identify key evaluation parameters and their applicable ranges. We analyzed survey data to determine essential criteria such as clean code, use of inheritance, and functionality, which were crucial for assessing the quality of academic software projects.
- Development of an intelligent system for evaluating software projects that process input criteria to produce a quantifiable measure of project success; its assessment is done by experts. We collaborated with three experts, including one professor and two instructors, to define fuzzy rules for the fuzzy inference system.
- Providing practical solutions for common challenges in academic project evaluation, such as instructor workload, resource limitations, and variations in student knowledge levels.
- Contributing to the field of AI in education by demonstrating the application of fuzzy logic to evaluate student performance.

The structure of the paper is as follows. Section I is this Introduction. Section II presents a thorough analysis of previous studies of academic project evaluation. Section III describes research methods, including an explanation of the fuzzy sets and logic that serve as the basis system. The section also covers the data collection procedure and the surveys used to identify the evaluation criteria. Results are presented in section IV. The study's conclusions and recommendations for future improvements are presented in Section V.

2. Related Work

This section reviews the existing literature on AI methods (specifically, fuzzy sets and logic) for evaluating software projects.

Developing assessment strategies and techniques that facilitate learning and teaching effectively has been the subject of extensive research [7]. Specifically, AI has been widely applied in education [14].

AI in education has the potential to revolutionize the learning experience, making it more personalized, engaging, and efficient. It can enhance student outcomes through personalized learning, intelligent tutoring systems, and automated grading and assessment [15–17]. However, challenges such as privacy, security, bias, and ethical considerations need to be addressed. Despite these challenges, AI can provide better data analysis, enabling educators to make data-driven decisions [15]. It can also automate administrative tasks, allowing educators to focus on personalized instruction [18]. Using AI in the classroom can tailor lessons, improve communication, and streamline routine work [19]. AI has been used to enhance personalized learning, feedback, and student evaluation, but challenges such as data quality and the need for training and skill development for educators and students must be addressed [20]. AI can potentially improve the efficiency and effectiveness of the teaching-learning process in education [19, 20]. Using AI in the classroom can improve communication, tailor lessons, and provide immediate feedback [21].

Fuzzy logic has been widely applied in education, particularly in decision-making processes and quality measurement [22, 23]. It has been applied in higher education to improve the quality of education and student learning experiences [24]. Fuzzy logic has been used in developing a student knowledge evaluation system to enhance the accuracy and fairness of student knowledge assessment [25]. It has also been applied in evaluating students' projects in engineering education, promoting unbiased evaluation [26]. Furthermore, it has been used in educational games to offer adaptation and personalized learning experiences [27]. Lastly, fuzzy logic has been proposed to measure educational achievement, particularly in assessing portfolio evidence [28].

Several works propose using fuzzy logic theory to evaluate students' performance [29, 30]. The authors used a criterion-based approach to evaluate student projects based on experiments. Students' work was graded according to a list of pre-agreed grading criteria developed by instructors in collaboration with students [31]. The authors allow users to modify the main and sub-criteria and their weights in decision-making systems according to their evaluation priorities. An objective multi-criteria decision-making system for evaluating the effectiveness and problem-oriented concepts in education has been proposed [32]. A survey questionnaire consisting of open-ended questions was also conducted in some studies to see the effectiveness of the study and get feedback.

Another study used fuzzy sets to determine the evaluation criteria and their corresponding weights. The matched criteria are then used to assess student learning outcomes [33]. The authors propose various criteria, such as acceptability, number of program classes, test coverage, and effectiveness, to help instructors evaluate program projects according to the criteria, given the strengths and limitations of the preferred project evaluation model, and to help project evaluators understand the logic behind different approaches to project evaluation [34]. The problem of assessing students' academic performance using the fuzzy logic model has been considered in [35]. Their research was based on assessments such as grades in lectures, practical classes, students' independent work, and laboratory work as criteria for academic performance. The other study introduced the fuzzy assessment system for distance learning that analyzes student performance, behavior, and exams [36]. Specification of teaching activity using fuzzy logic was introduced in [37].

Several studies considered the idea of automatic grading of students' projects. Evaluating academic software can be complex and multifaceted. Intelligent systems may need help to handle such a complex evaluation, especially if multiple criteria must be considered. Some aspects of academic software evaluation, such as user experience and interface design, are subjective. It can be difficult for an intelligent system to handle subjective evaluations because they vary from user to user. The fuzzy approach has also been used in an assessment model that builds upon the VIKOR compromise ranking method and uses the fuzzy multi-criteria decision-making (MCDM) approach to gauge the success of software development projects [12]. Another study focused on applying the association rules for project evaluation [38].

A recent work [7] provides a comparative analysis of how AI can improve student learning outcomes through assessment and feedback procedures. The study overviews the most popular AI and ML algorithms for student success. According to the results, I-FCN outperformed other methods (ANN, XG Boost, SVM, Random Forest, and Decision Trees). Fuzzy Logic was not used in this analysis. More recent work by [39] used a hybrid approach (ML models and fuzzy sets) to evaluate students' readiness for post-graduation challenges using surveys.

As we see, neural networks, deep learning, random forest, logistic regression, multilayer perceptron, naive Bayes, support vector machines, decision trees, and fuzzy methods have all been used in studies for the assessment and evaluation of student performance evaluation in the literature. However, most studies use subjectively defined criteria, and limited works provide ways to customize these methods to specific courses and experts. Additionally, despite the numerous research studies on student performance evaluation, only a few works focused on engineering project evaluation.

3. Methods

The creation of an intelligent system is divided into several stages, including data collection, definition of evaluation criteria, system design, and development. Data collection includes gathering relevant information from academic software projects via surveys of students and experts (teachers). Questionnaire responses will be collected and analyzed to determine key evaluation factors and their importance. The system design phase focuses on defining the architecture and functionality of the intelligent system. The development phase involves writing code and building the intelligent system.

3.1. Fuzzy Sets and Logic

Fuzzy set theory will be used as the basis for the evaluation model. Lotfi Zadeh introduced fuzzy sets in the 1960s to represent uncertainty and fuzziness in natural language expressions [40]. Fuzzy sets are used in various applications, such as decision-making, control systems, pattern recognition, and artificial intelligence. Fuzzy logic allows the representation of imprecise and uncertain information often found in software project evaluation. Fuzzy sets will be used to define evaluation criteria and linguistic variables.

3.1.1 Membership Functions and Fuzzy Sets

Fuzzy sets, first introduced by Zadeh [40], allow degrees of membership, which are indicated with a number between 0 and 1. So, in contrast to the pair of numbers {0,1} in Boolean logic, we move to all the numbers in a range [0,1]. This is called a *membership function* (MF) and is denoted as $\mu_A(x)$ and, in this way, can denote fuzzy sets. MFs are mathematical techniques for modeling the meaning of symbols by indicating flexible membership to a set. We can use it to represent uncertain concepts like age, performance, building height, etc. Therefore, MF's essential function is to convert a crisp value to a membership level in a fuzzy set.

The shape of the membership function reflects the degree of fuzziness or uncertainty of the set. In this study, we use triangular and trapezoidal MFs, illustrated in Fig. 3 and Fig. 4. The triangular membership function is defined by three parameters a, b, and c, where $a \leq b \leq c$. It is described by the piecewise function (1).

$$\mu_{\text{triangular}}(x; a, b, c) = \begin{cases} \frac{x-a}{b-a} & if\ a \leq x < b, \\ \frac{c-x}{c-b} & if\ b \leq x < c, \\ 0 & otherwise. \end{cases} \tag{1}$$

This function increases linearly from 0 at $x = a$ to 1 at $x = b$ and decreases back to 0 at $x = c$.

The trapezoidal membership function is defined by four parameters a, b, c and d, where $a \leq b \leq c \leq d$. It is described by the piecewise function (2).

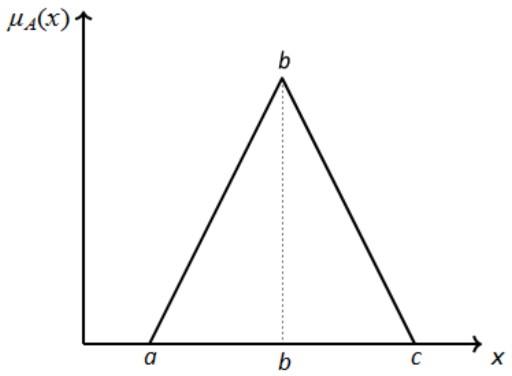


Fig. 3. Triangular Membership Function

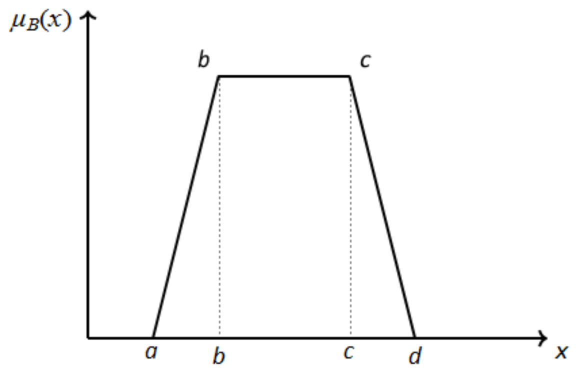


Fig. 4. Trapezoidal Membership Function

$$\mu_{\text{trapezoidal}}(x; a, b, c, d) = \begin{cases} \frac{x-a}{b-a} & if\ a \leq x < b, \\ 1 & if\ b \leq x \leq c, \\ \frac{d-x}{d-c} & if\ c < x \leq d, \\ 0 & otherwise./ \end{cases} \tag{2}$$

This function increases linearly from 0 at $x = a$ to 1 at $x = b$, stays constant at 1 between $x = b$ and $x = c$, and decreases back to 0 at $x = d$.

Representing linguistic terms and hedges, or linguistic expressions that modify other expressions, is a significant component of the fuzzy set theory framework [41]. A fuzzy set typically represents a linguistic term, and an operation that changes one fuzzy set into another represents a linguistic modifier or hedge.

3.1.2 Linguistic Variables

According to Zadeh [42], "By a linguistic variable we mean a variable whose values are not numbers but words or sentences in a natural or artificial language". For example, following that logic, the label *high* is considered a linguistic value of the variable *Student Performance*. It plays almost the same role as a number but needs to be more precise. The collection of all linguistic values of a linguistic variable is referred to as a *term set*.

3.1.3 Fuzzy Hedges

There are two families of modifiers, or hedges reinforcing and weakening modifiers. The hedge "very" represents the reinforcing modifier (3).

$$t_{\mathrm{very}}(u) = u^2 \tag{3}$$

The second family of modifiers is weakening modifiers. For instance, "more-or-less" hedge (4).

$$t_{\text{more-or-less}}(u) = \sqrt{u} \tag{4}$$

Furthermore, the "not" hedge is represented in (5).

$$t_{\mathrm{not}}(u) = 1 - u \tag{5}$$

Hedges can be applied several times. For example, *not very good performance* is the example of a combined hedge consisting of two atomic hedges *not* and *very*.

3.1.4 Fuzzy Operations

The α-cut (Alpha cut) is a crisp set that includes all the members of the given fuzzy subset f whose values are not less than α for $0 < \alpha \leq 1$ (6).

$$f_\alpha = \{x : \mu_f(x) \geq \alpha\} \tag{6}$$

Equations (7) and (8) are used to connect α-cuts and set operations (A and B are fuzzy sets).

$$(A \cup B)_\alpha = A_\alpha \cup B_\alpha \tag{7}$$

$$(A \cap B)_\alpha = A_\alpha \cap B_\alpha \tag{8}$$

3.1.5 Fuzzy Rules

Fuzzy rules control the output variable. A fuzzy rule is a usual if-then rule containing a condition and conclusion. It has the following form: For example, Rule 15: *If Clean code is Low AND Functionality level is High AND Use of inheritance is Medium THEN Project success is Good*.

Fuzzy sets have advantages over classical sets when dealing with complex, uncertain, or subjective data (see Fig. 5). However, they also have some limitations, such as difficulty defining membership functions and the lack of clear criteria for set membership.

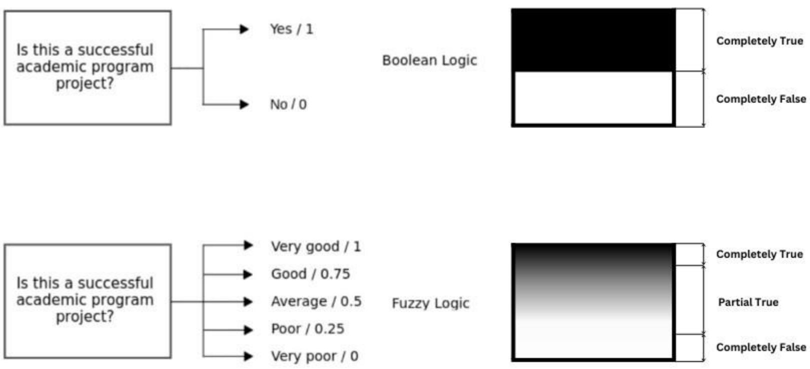


Fig. 5. Academic performance evaluation using Classical and Fuzzy sets.

In traditional grading systems, grades are given based on a fixed set of criteria, such as running a project with no errors and solution independence. Instead of giving a student a letter grade based on a fixed percentage, the fuzzy sets approach can give a grade based on how well the student's performance meets specific criteria. The instructor must first define the assessment criteria to use fuzzy sets to assess student performance. These criteria can be defined using linguistic variables such as "good," "average," and "poor."

To apply these rules to a particular student's work, it is necessary to determine the extent to which the project falls into each category. It can be done through various methods, such as self-assessment through questioning, teacher evaluation, or assessment by specific software analysis tools.

3.2. Data Collection

Our study's methodology involved two primary datasets: students' projects and data obtained from a detailed survey of students and instructors.

3.2.1 Projects of students

The first dataset comprised the source codes of 64 projects completed by teams of second-year students from the Information Systems major at the Kazakh-British Technical University, specifically from the School of Information Technology and Engineering (SITE). Each project was implemented by a team of four students. These projects were part of an Object-Oriented Programming (OOP) course, where students were tasked with developing an Information System for a research-oriented university. Students were required to create a system with various components, including classes (superclasses, subclasses, abstract classes), interfaces, enumerations, custom exceptions, and design patterns. The project also involved system design, where students had to create architecture using Use Case and UML class diagrams before coding. The project aimed to integrate multiple techniques studied throughout the course. Fig. 6 presents some code samples from the collected dataset.

3.2.2 Students survey data

The second dataset was obtained from a survey of students who had completed the OOP course and their instructors. The survey aimed to identify the key performance indicators required for evaluating OOP projects (discussed later in the subsection *Survey*).

The goal was to identify the key performance indicators required for evaluating OOP projects. The survey was completed by 32 teams, each consisting of four second-year SITE students majoring in Information Systems. Three course instructors also participated in the survey to provide expert insights. The survey included 21 project-related questions. These questions were designed to capture both quantitative and qualitative data. Quantitative data included numerical values for metrics such as the number of classes and lines of code. Qualitative data included subjective evaluations of clean code practices and the use of design patterns. Fig. 7 shows the distribution of the number of classes in the project, Fig. 8 presents the distribution of the number of lines of code used in the project (based on survey results), and Fig. 9 shows the distribution of final marks students got for the project.

```java
public class Student {
    private String name;
    private String studentId;
    private List<Course> courses;

    public Student(String name, String studentId) {
        this.name = name;
        this.studentId = studentId;
        this.courses = new ArrayList<>();
    }

    public boolean enrollInCourse(Course course) {
        if (course.addStudent(this)) {
            courses.add(course);
            System.out.println(name + " has enrolled in " + course.getName() + "
            return true;
        } else {
            System.out.println("Enrollment failed: " + course.getName() + " is f
            return false;
        }
    }
```

(a) Example of clean code

```java
public class UniversitySystem {
    public static void main(String[] args) {
        String[] studentNames = {"John Doe", "Jane Doe"};
        String[] studentIds = {"001", "002"};
        String[] courseNames = {"Computer Science", "Mathematics"};
        int[] courseCapacities = {2, 3};
        List<String>[] enrolledStudents = new ArrayList[2];
        enrolledStudents[0] = new ArrayList<>();
        enrolledStudents[1] = new ArrayList<>();

        // Enroll John Doe in Computer Science
        enrollStudent(studentNames[0], studentIds[0], courseNames[0], courseCapac
    }

    public static void enrollStudent(String studentName, String studentId, String
        if (enrolledStudents.size() < courseCapacity) {
            enrolledStudents.add(studentName);
            System.out.println(studentName + " has enrolled in " + courseName + "
        } else {
            System.out.println("Enrollment failed: " + courseName + " is full.");
        }
```

(b) Example of poorly written code

Fig. 6. Sample project code from the dataset of projects.

3.3. Survey

Various methods can be employed to identify the key fuzzy variables and their corresponding sets, such as surveys, direct rating methods, or consulting experts. We used a data-driven approach; by analyzing the distribution of data points (the mean, the median, the standard deviation), we decided on the parameters of membership functions, considering expert opinions as well. It is a common practice to identify key success factors for a project from a survey (see Fig. 10) [43]. So, in our case, we engaged three experts and conducted a survey among students who had completed the Object-Oriented Programming and Design course. The objective was to pinpoint the key performance indicators for evaluating OOP projects. Working collaboratively with these experts, we identified the necessary fuzzy variables, sets, and partitions. 32 teams, each consisting of four 2-year SITE (school of information technology and engineering) students, participated in the survey. The survey contained 21 project-related questions.

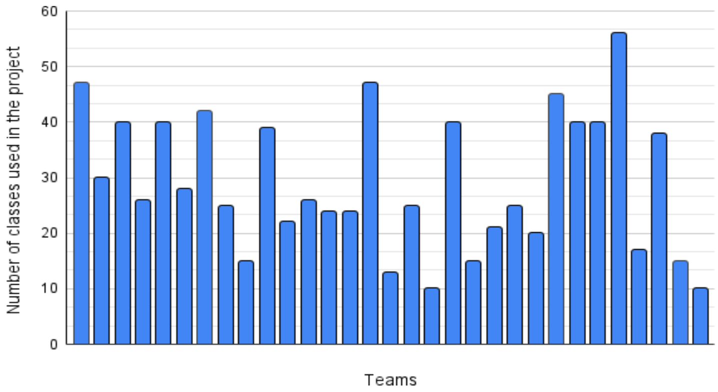


Fig. 7. Distribution of the number of classes used in the project

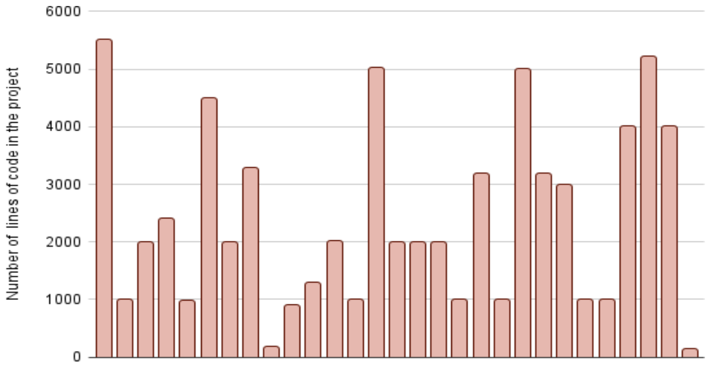


Fig. 8. Distribution of the number of lines of code used in the project

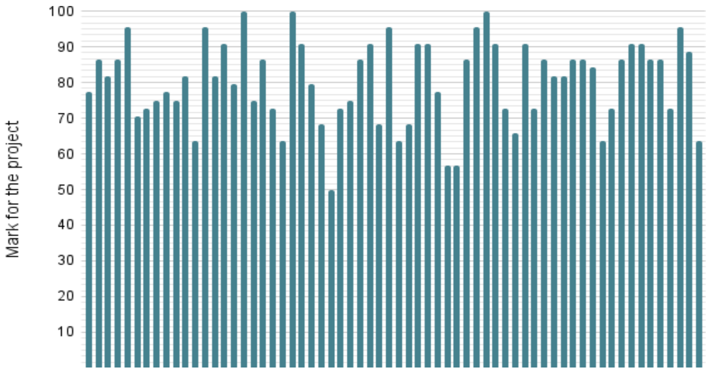


Fig. 9. Distribution of the final marks for the project.

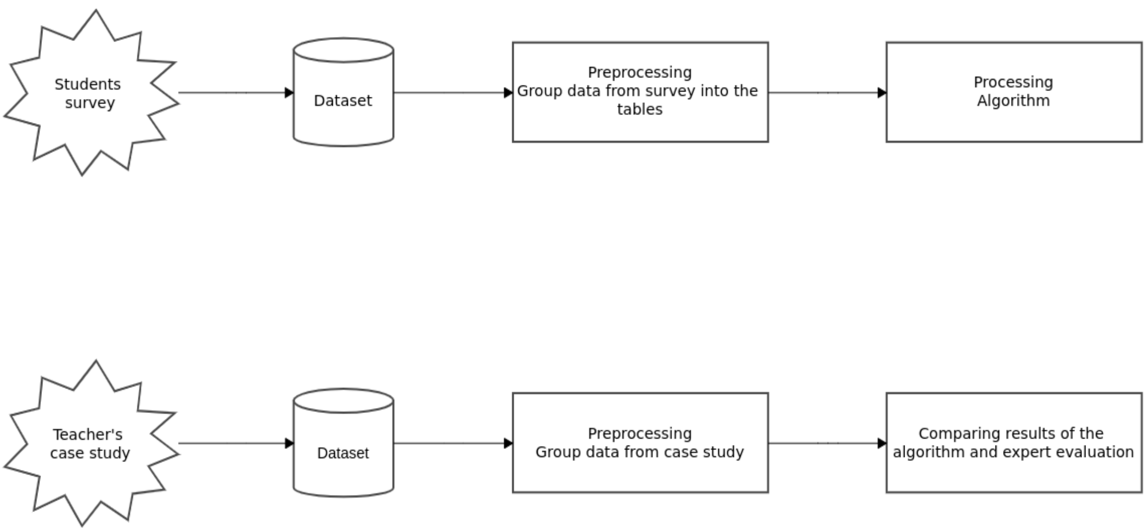


Fig. 10. Development of evaluation criteria. Authors use a survey and case analysis with the instructor, and a data set will be compiled to gather information about the criteria for analysing software projects. The selected criteria will be used to analyze the success of the projects.

Fig. 11 and Fig. 12 illustrate distributions of equal contribution of team members to the project and frequent ways to communicate with the team while working on the project, respectively.

The questionnaire was designed by course experts. Some of the questions were adapted from the book [44]. Surveys were distributed and collected electronically to ensure easy access and completion. The survey form is presented in Fig.

13. The survey contained the following questions:

- Team Leader's Name: First and last name of the team leader
- Number of Classes Used: Number of classes used in the project (e.g., 37)
- Number of Meetings: Number of online/offline meetings held during the project (e.g., 6-10).
- Communication Method: Main methods of communication used with the team (e.g., Offline, Telegram, Discord, etc.)

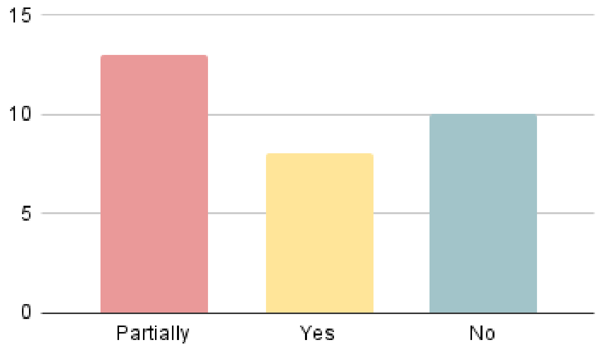


Fig. 11. Distribution of equal contribution of team members to the project

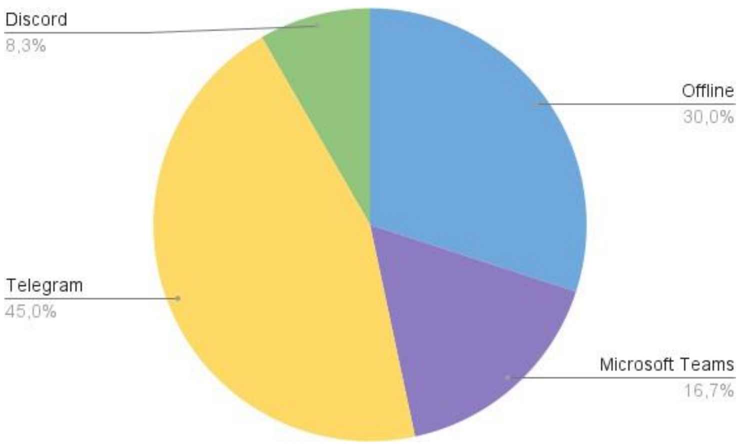
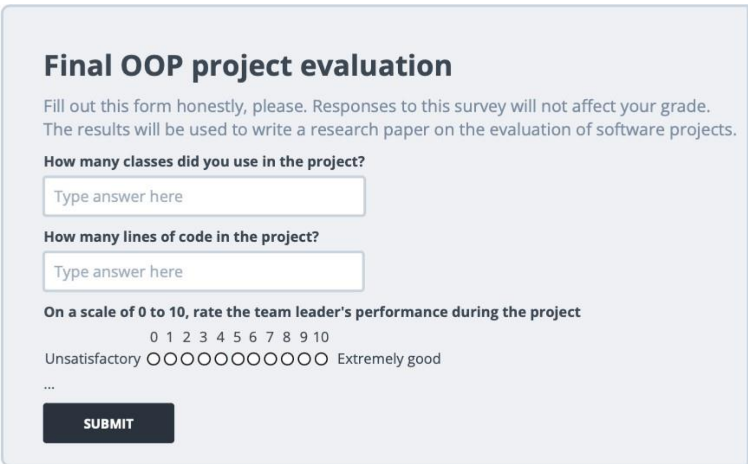


Fig. 12. Distribution of frequent ways to communicate with team while working on the project



Final OOP project evaluation

Fill out this form honestly, please. Responses to this survey will not affect your grade. The results will be used to write a research paper on the evaluation of software projects.

How many classes did you use in the project?

Type answer here

How many lines of code in the project?

Type answer here

On a scale of 0 to 10, rate the team leader's performance during the project

0 1 2 3 4 5 6 7 8 9 10

Unsatisfactory ○○○○○○○○○○○ Extremely good

...

SUBMIT

Fig. 13. Survey form for students

- Equal Contribution: Student's opinion on whether all team members contributed equally (yes/no)
- Did you consult with the lecturer or assistants during the project? (yes/no).
- How many lines of code are in the project? (e.g., 2000)
- How many uncommented lines of code in the project? (e.g., 1800)
- How many methods per class on average in the project? (e.g., 10)
- How many public methods per class on average in the project? (e.g., 3)
- How many public instance variables per class on average in the project? (e.g., 4)
- How many parameters per method on average in the project? (e.g., 3)
- How many lines of code per method on average in the project? (e.g., 30)
- Choose an appropriate distribution of tasks during the project (e.g., The leader took most of the work himself, gave minor tasks to the team)
- On a scale of 0 to 10, rate the team leader's performance during the project
- If you used patterns, what patterns did you use to design the project? (e.g., decorator)
- Documentation Creation: Whether certain types of documentation were created (e.g., software requirements specification document).
- Group Communication: Whether a group was created for communication.
- Importance to Career: How important the student thinks the project is to their future career.

As a result, we obtained a dataset containing information about a final Object-Oriented Programming (OOP) project evaluation, with each row representing a student's responses. We also extended the dataset with the real marks students obtained for their project. Data from the surveys were then compiled and analyzed to determine the critical performance indicators for the OOP projects. This analysis included calculating statistical measures such as mean, median, and standard deviation for quantitative metrics and thematic analysis for qualitative responses.

Table 1 shows the statistics of certain project features explored in the survey.

Table 1. Academic software project characteristics (data from students survey)

Feature	Average	Maximum	Minimum
Number of classes	29.2	56	10
Number of meetings	-	More than 15	0-5
Number of methods per class	-	50-60	3
Number of lines of code	2406.2	5500	130
Number of lines of code per method	-	90	3-7
Mark	82.5	100	56.8

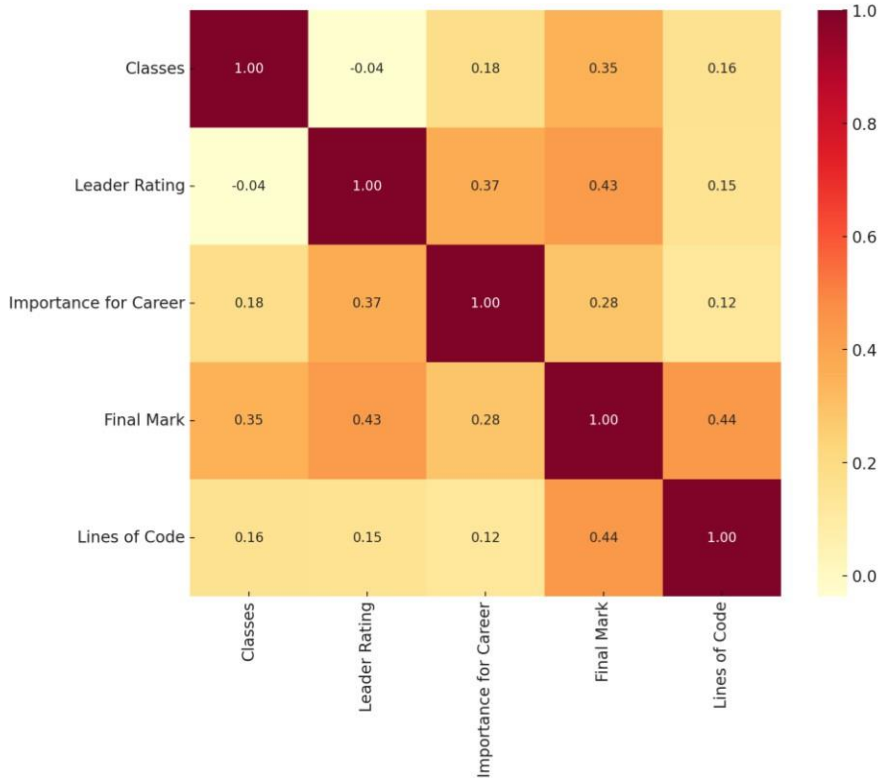


Fig. 14. Correlation heatmap of project evaluation metrics

Fig. 14 presents the correlation heatmap of project evaluation metrics built using numerical data extracted from a survey. The heatmap palette ranges from light yellow (indicating lower correlation) to deep orange (indicating higher correlation). The following observations can be made:

- **Final Mark and Classes.** Moderate positive correlation (0.35). So, having more classes in the project might be associated with slightly higher final marks, potentially reflecting a more complex project structure with bigger functionality.
- **Final Mark and Leader Rating.** Moderate positive correlation (0.43). Effective leadership likely contributes to better project outcomes.

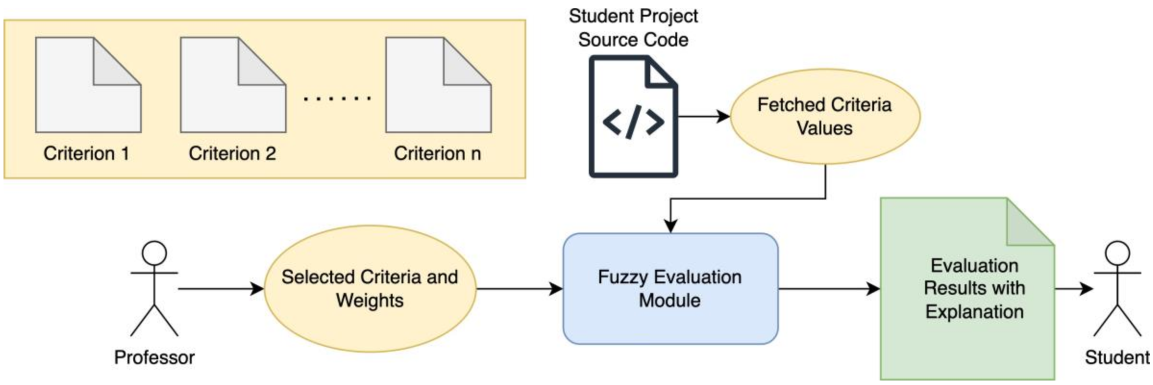


Fig. 15. The Proposed Evaluation Methodology.

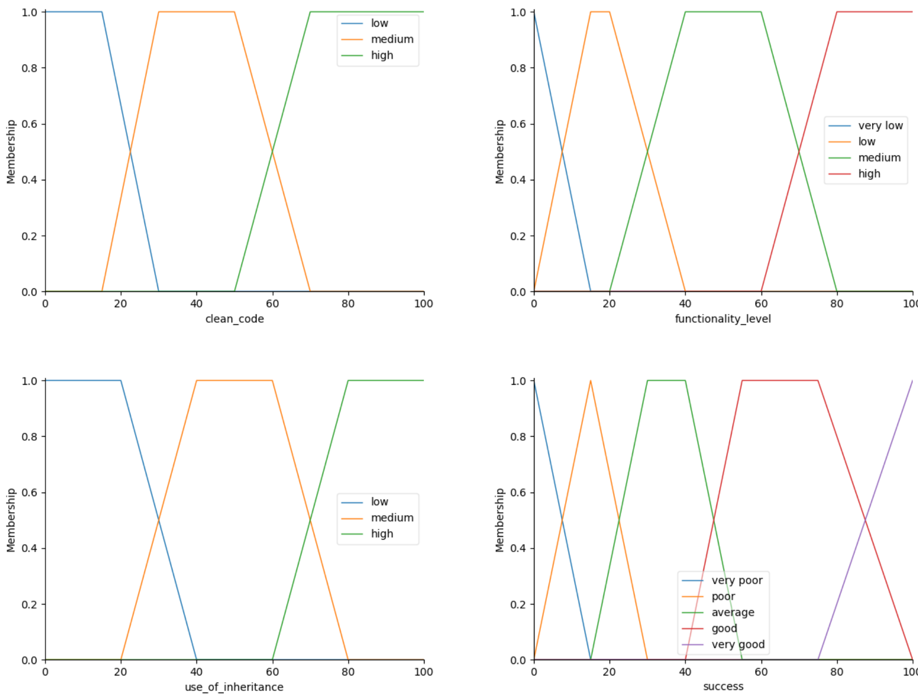


Fig. 16. Input fuzzy sets for Clean code , Functionality level, Use of inheritance and Output fuzzy sets for Success

- **Final Mark and Lines of Code.** A moderate positive correlation (0.44) suggests that projects with more lines of code tend to receive higher marks. This might indicate that larger or more complex projects, which require more code, are viewed favorably in evaluations, assuming the quality of the code is also high.

These correlations reveal how various factors related to project management and execution can influence a project's overall evaluation.

3.4. Proposed Methodology

The proposed intelligent system for evaluating academic software projects using a fuzzy inference system is presented in Fig. 15. Table 2 shows the information about term sets of the input and output variables and their domains. Table 3 provides information about project features selected to assess each fuzzy variable (evaluation criterion). Fig. 16

presents the membership functions for all fuzzy variables.

We partition the spectrum of possible assessments corresponding to linguistic tags [45]. We have three input variables describing the project *Clean Code*, *Functionality Level*, *Use of Inheritance*. The output variable is *Project Success* (see Fig. 16). As can be seen, we have *'Low'*, *'Medium'*, and *'High'* fuzzy sets for *Clean Code* and *Use of Inheritance* input variables, *'Very Low', 'Low', 'Medium',* and *'High'* fuzzy sets for *Functionality Level* and *'Very Poor', 'Poor', 'Average', 'Good',* and *'Very Good'* for the output variable. The linguistic expressions for the fuzzy model's output variable were partly adapted from [46].

Table 2. Fuzzy attributes of the fuzzy inference system.

Fuzzy Variable	**Term Set**	**Domain**
Clean Code	T = {Low, Medium, High}	X=[0,100]
Functionality Level	T = {Very Low, Low, Medium, High}	X=[0,100]
Use of Inheritance	T = {Low, Medium, High}	X=[0,100]
Project Success	T = {Very Poor, Poor, Average, Good, Very Good}	X=[0,100]

Table 3. The table demonstrates parameters to evaluate the code of software projects

clean code	functionality	use of inheritance
patterns presence	use of collections	use of overriding/overloading
number of fields	use of own interfaces / build-in	inherited classes
number of parameters in methods	use of serialization	use of polymorphism
use of comments	use of comparators	
number of own exceptions / build-in	number of methods	
	number of classes	
	lines of code	

We use fuzzy rules to build fuzzy relationships between input and output variables. Our fuzzy inference system has 36 fuzzy rules, as shown in Table 4. Fuzzy if-then rules are descriptive and usually created by experts or human knowledge. Based on the survey responses and instructors' opinions, fuzzy rules were created to determine the relationship between the evaluation criteria and the output linguistic variables. We identified these fuzzy evaluation criteria and fuzzy rules by working collaboratively with one professor and two instructors, paying attention to survey results. These rules form the basis for evaluating academic software projects (see Table 4).

In a more general case, when it is considered time-consuming to survey experts and students, collecting criteria and their importance can be done via the system. For example, let C = {C_1, C_2, ..., C_n}be the evaluation criteria that an instructor or professor chooses, e.g., the code clarity, documentation, etc. Then w = {w_1, w_2, ...w_n} are the weights (importance) of the corresponding criteria chosen by the course instructor. Next P = {P_1, P_2, ..., P_n} are student projects. L = {L_1, L_2, ..., L_n} represents linguistic variables with the corresponding term set representing its assessment, a vector of linguistic terms on L_i *{High, Average, Low}*.

Then, a student project can be evaluated based on weight-based fuzzy rules generation [47, 48]. If the weight is high, a mark for this criterion is essential. After a case study with professors about their evaluation methods, we concluded that the most essential criterion greatly influences the final result. For example, we have three criteria *Clean code, The use of Inheritance, Functionality* with respective weights selected by the Professor as, for example, *Medium, Low, High*. The code quality score was obtained by summing up all related scores automatically extracted from the source code, including the number of methods, following naming conventions, etc.

4. Results

4.1. Simulation and Performance Evaluation

We can now simulate our fuzzy inference system by specifying the inputs and using defuzzification. For example, let us consider the following input data and determine the overall project success: The *Clean code, Functionality*, and *Use of inheritance* are 61%, 74%, and 68% respectively. The output membership functions are then mixed with the maximum operator (fuzzy aggregation). Next, to obtain a clear answer, we must do defuzzification, which we accomplish using the centroid approach. Fuzzy rule-based aggregation yields 63.27 % as the total project success. Fig. 18 shows the visualized result.

Three instructors from an Object-Oriented Programming (OOP) course were enlisted to assess the proposed fuzzy intelligence system's effectiveness. Each instructor evaluated three student projects manually, using predefined evaluation criteria, and the mean of their evaluation was taken. These evaluations were then compared with assessments conducted by the fuzzy intelligent system. The results of this comparative analysis are presented in Table 5 and Fig. 17. As can be seen, similar scores were obtained from traditional evaluation methods and those predicted by the proposed system. The findings indicate that the fuzzy intelligent system performed robustly, demonstrating promising results that aligned with the manual evaluations conducted by the course instructors.

4.2. Application Prototype

Fig. 19 shows the layout of a professor evaluation form specifically designed for evaluating software projects. The professor has to enter data such as course name, personal information, and student data and then upload the code of the student's software project into the form. After entering the preliminary data about the software project, the instructor can select different evaluation criteria with appropriate weights reflecting the importance of each criterion. Ultimately, the form generates recommendations for evaluating student work based on weighted criteria, simplifying the grading process and ensuring a fair and objective evaluation of the student's project that meets educational standards and expectations.

Table 4. Fuzzy rules used in the fuzzy inference system.

Rule	Clean Code	Functionality Level	Use of Inheritance	Project Success
1	High	High	High	Very Good
2	Medium	High	High	Very Good
3	Low	High	High	Good
4	High	Medium	Medium	Good
5	Medium	Medium	Medium	Average
6	Low	Medium	Medium	Average
7	High	Low	Low	Poor
8	Medium	Low	Low	Very Poor
9	Low	Low	Low	Very Poor
10	High	Very Low	High	Average
11	Medium	Very Low	High	Poor
12	Low	Very Low	High	Poor
13	High	High	Medium	Very Good
14	Medium	High	Medium	Good
15	Low	High	Medium	Good
16	High	Medium	Low	Average
17	Medium	Medium	Low	Average
18	Low	Medium	Low	Poor
19	High	Low	High	Average
20	Medium	Low	High	Average
21	Low	Low	High	Poor
22	High	Very Low	Medium	Poor
23	Medium	Very Low	Medium	Poor
24	Low	Very Low	Medium	Very Poor
25	High	High	Low	Good
26	Medium	High	Low	Average
27	Low	High	Low	Poor
28	High	Medium	High	Very Good
29	Medium	Medium	High	Good
30	Low	Medium	High	Average
31	High	Low	Medium	Average
32	Medium	Low	Medium	Average
33	Low	Low	Medium	Poor
34	High	Very Low	Low	Poor
35	Medium	Very Low	Low	Poor
36	Low	Very Low	Low	Very Poor

Table 5. Comparing real marks (traditional evaluation methods) and predicted marks provided by the proposed fuzzy intelligent system

	Clean Code	Functionality	Use of Inheritance	Proposed method	Real mark
Project 1	100	82	84	**92**	**95**
Project 2	67	34	100	**64**	**57**
Project 3	100	63	100	**91**	**86**

Such an intelligent system can be integrated with the SonarLint code analyzer as an alternative to manually evaluating the criteria or taking them from students' surveys. The system includes a user-friendly interface for entering project data and evaluating it based on predefined fuzzy rules. It can analyze project source code and provide evaluation results in an understandable format. Fig. 20 shows the project evaluation final report page prototype.

The proposed system cannot replace the teacher. However, it helps evaluate the project by analyzing the student's work based on the given criteria. The user needs to specify the criteria and their weight for evaluation, indicate the documentation on the project, and upload the project's source code to get the algorithm's result. The system analyses the number of classes in the project, the average length of code in classes, the number of fields and methods, and the use of access modifiers (private/package/public). Also, with the help of the connected anti-plagiarism service, the system checks the project's uniqueness based on the uploaded works and information on the Internet and gives the plagiarism percentage.

5. Discussion

5.1. Comparison with related studies

Several recent studies have utilized a fuzzy approach to evaluate students' academic performance [49–52]. The comparative analysis of the current study with similar papers reveals similarities and distinctions in applying fuzzy logic for educational evaluations. In our study, we developed a fuzzy intelligent system for evaluating academic software projects, emphasizing technical criteria such as clean code, use of inheritance, and functionality. This approach aligns with the findings of [50], highlighting the benefits of fuzzy logic in improving accuracy and fairness in course-specific evaluations. However, our study distinguishes itself by integrating automated tools like SonarLint for code quality analysis, providing a flexible solution that addresses the subjective nature of project assessments.

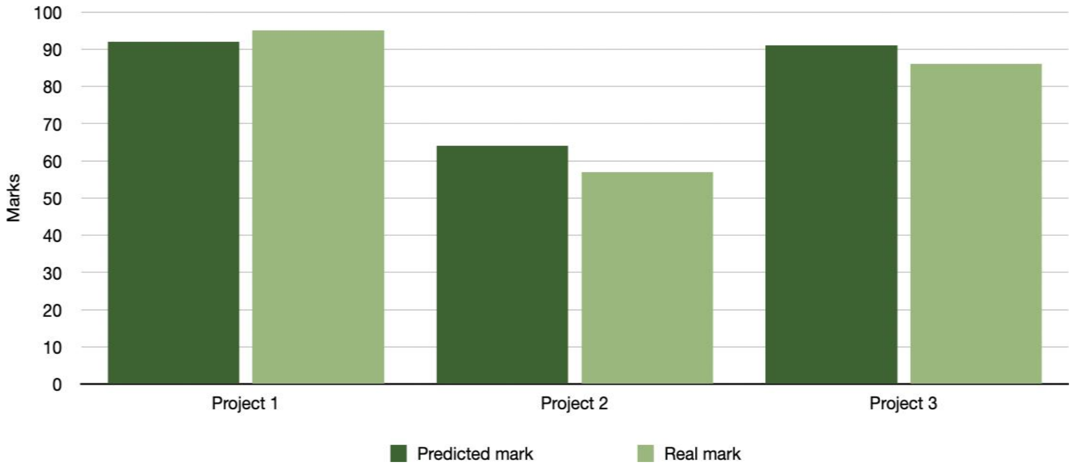
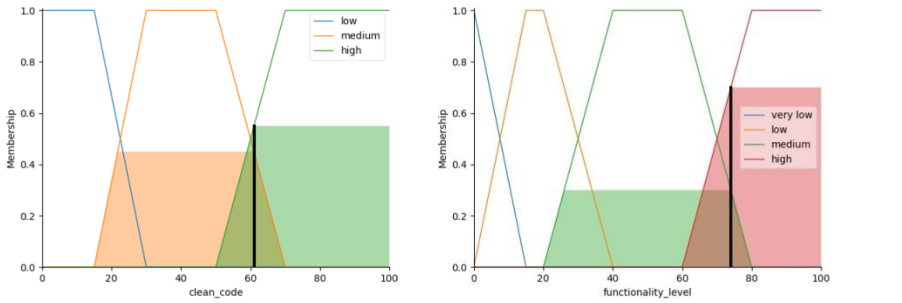
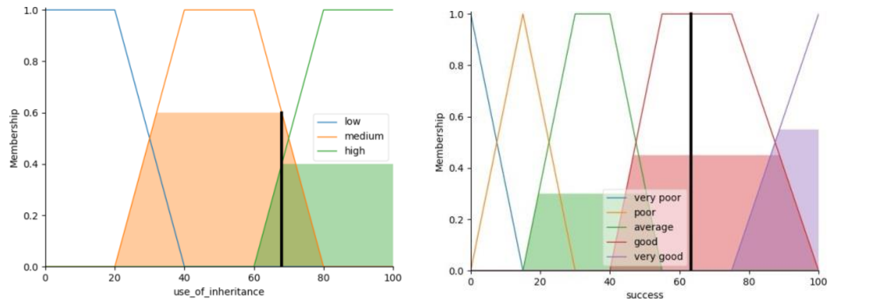


Fig. 17. Project evaluation scores obtained using traditional evaluation method and predicted by the proposed system



(a) Applying input 61% on *Clean code* fuzzy set (b) Applying input 74% on *Functionality level* fuzzy set



(c) Applying input 68% on *Use of inheritance* fuzzy set (d) Aggregated Membership and Result, 63.27%

Fig. 18. Simulation Results.

Similarly, [51] emphasizes the effectiveness of fuzzy logic in handling subjectivity and enhancing consistency in technical project evaluations. Both studies focus on reducing instructor workload and improving the reliability of assessments. In contrast, [52] extends the application of fuzzy logic to a broader range of educational assessments, demonstrating its adaptability across various domains, including non-technical skills. This broader approach contrasts with our targeted focus on software project evaluations. Another approach to this problem represents neuro-fuzzy systems: these systems combine neural networks with fuzzy logic to classify and predict student performance. They utilize previous exam results and other factors to label students based on expected performance [49]. These recent studies highlight the potential for future research to expand our system by integrating broader contextual factors to further enhance our evaluation framework's applicability.

All studies utilized expert knowledge to define fuzzy rules and membership functions (triangular and trapezoidal in most cases), ensuring the evaluation criteria are grounded in domain expertise.

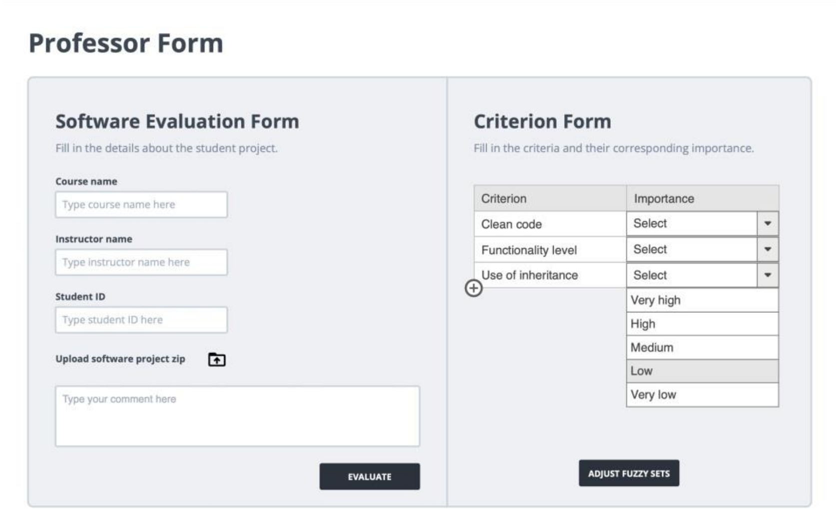


Fig. 19. Prototype: Project Evaluation Professor Form.

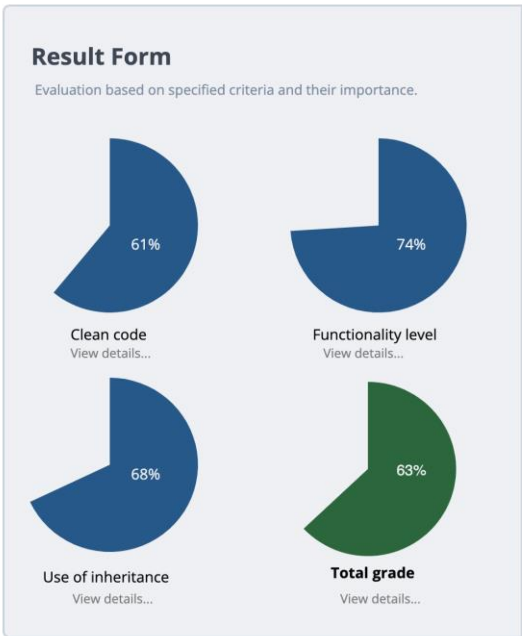


Fig. 20. Evaluation Report

5.2. Feasibility of the Experimental Design and Implementation

Several challenges were encountered during the system development. One significant challenge was defining the evaluation criteria due to the subjective nature of the criteria and the diversity of academic software projects. To address this, we conducted a survey involving student project teams (n=31) and faculty (n=3) to identify key evaluation parameters and their applicable ranges. By involving multiple experts in the survey, we ensured that the selected criteria—clean code, use of inheritance, and functionality—were representative and widely accepted.

Another challenge was developing appropriate fuzzy sets and rules, which was complex due to the need to accurately represent the identified criteria and their interactions. These rules were based on expert knowledge and survey data but may need to be regularly monitored and updated to remain effective and relevant.

Additionally, integrating the fuzzy intelligent system with existing tools and testing it in real-world educational scenarios posed significant challenges. We developed an application prototype with a user-friendly interface for instructors to enter project data, select evaluation criteria, and obtain evaluation results. The system was integrated with the SonarLint code analyzer to enhance its functionality. Rigorous testing was conducted by comparing the system's evaluations with manual evaluations performed by course instructors. The results showed that the system's evaluations aligned closely with those of the instructors, demonstrating the system's reliability and accuracy.

6. Limitations and Future Improvements

As for the limitations, the system may not easily adapt to course content or evaluation standards changes without significantly reconfiguring the fuzzy sets and rules. Another limitation is subjectivity in the criteria definition. The fuzzy set partitions depend on experts' subjective judgments. So, the possible improvement for future works can involve a more objective method for defining evaluation criteria using data analytics and machine learning to analyze historical project data. Another improvement we plan is the incorporation of teamwork as an evaluation parameter.

Based on the analysis of the system, here is the summary of potential limitations, along with suggested ways to mitigate them:

1. *Adaptability to Different Course Content.* The system may not easily adapt to changes in course content or evaluation standards without significant reconfiguration of the fuzzy sets and rules. The solution is to implement a dynamic and modular system design that allows for easy updates and customization of evaluation criteria and rules. Regular feedback from instructors should be incorporated to keep the system up-to-date with the latest educational standards.
2. *Handling of Diverse Project Types.* The system might face challenges evaluating various project types, especially if the projects have very different characteristics and requirements. Expanding the system to include a broader set of criteria and rules covering a wider range of project types can be a potential solution. Allowing instructors to customize the criteria for specific projects is also important.
3. *Limited Consideration of Teamwork and Collaboration.* The current system focuses primarily on the technical aspects of the projects and may not fully account for the teamwork and collaboration aspects. We plan to include teamwork and collaboration as evaluation parameters in future work.

7. Conclusion

This paper proposes a novel approach for evaluating software projects in an academic environment. By implementing a fuzzy intelligent system, we aim to automate the evaluation process, reduce subjective biases, and manage the increasing instructor workload effectively. We surveyed students and faculty, and the responses helped to identify key evaluation factors in evaluating academic software projects. The fuzzy system uses predefined criteria clean code, use of inheritance, and functionality transformed into fuzzy sets and employs a fuzzy inference mechanism defined in collaboration with educational experts. Fuzzy set theory served as the basis for our evaluation model, allowing us to represent imprecise and subjective information related to project evaluation.

The proposed fuzzy intelligent system for evaluating academic software projects has several unique features that distinguish it from existing solutions:

- Integration of fuzzy logic for handling subjectivity and uncertainty in assessments, unlike traditional grading systems that rely on fixed criteria.
- Expert and survey defined evaluation criteria and fuzzy rules. Key evaluation parameters are identified based on surveys involving both students and faculty.
- Validation through real-world testing in a university setting. The system has been tested by comparing its evaluations with manual evaluations performed by course instructors.

The study results can help academic instructors save time, reduce costs, and improve the quality and efficiency of evaluating student software projects. In turn, students can get faster feedback on their work and analyze the code of their software projects.

The proposed system has numerous potential applications in various educational settings. For example, it can be widely implemented in university-level software engineering courses to evaluate student projects. In addition, such systems can be used in online and distance learning programs to evaluate student work remotely. Finally, it can be applied in professional training and certification programs to assess the competencies of participants, assisting in maintaining the integrity of certification processes.

Acknowledgment

This research was funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP22786412). The authors thank all students and instructors from the School of Information Technology and Engineering of Kazakh-British Technical University for contributing to this study (survey participation, opinions, and recommendations).

References

[1] Shahid Rafiq, Ayesha Afzal, and Farrukh Kamran. Exploring the problems in teacher evaluation process and its perceived impact on teacher performance. *Gomal University Journal of Research*, 38:482–500, 12 2022.

[2] Mohd Yousuf and Abdul Wahid. The role of Artificial Intelligence in Education: Current Trends and Future Prospects. In *2021 International Conference on Information Science and Communications Technologies (ICISCT)*. IEEE, nov 3 2021.

[3] Chih-Pu Dai and Fengfeng Ke. Educational applications of artificial intelligence in simulation-based learning: A systematic mapping review. *Computers and Education: Artificial Intelligence*, 2022.

[4] M. Ivanovic´, Aleksandra Klašnja-Milic´evic´, M. Paprzycki, M. Ganzha, C. Baˇdicaˇ, A. Baˇdicaˇ, and L. Jain. Current Trends in AI-Based Educational Processes An Overview. *Intelligent Techniques in the Educational Process*, 2022.

[5] Xi Ganga. Educational Artificial Intelligence (EAI) Connotation, Key Technology and Application Trend Interpretation and analysis of the two reports entitled "Preparing for the Future of Artificial Intelligence" and "The National Artificial Intelligence Research and Development Strategic Plan". *International Conference on Applied Algorithms*, 2021.

[6] Víctor González-Calatayud, Paz Prendes-Espinosa, and Rosabel Roig-Vila. Artificial intelligence for student assessment: A systematic review. *Applied Sciences*, 11(12), 2021.

[7] Monika Hooda, Chhavi Rana, Omdev Dahiya, Ali Rizwan, and Md Shamim Hossain. Artificial intelligence for assessment and feedback to enhance student success in higher education. *Mathematical Problems in Engineering*, 2022:1–19, May 2022.

[8] Ramazan Yilmaz. Problems experienced in evaluating success and performance in distance education: A case study, 2017.

[9] Natasha Angeloska Galevska. Challenges of teachers in the process of evaluation and grading. *The Eurasia Proceedings of Educational & Social Sciences (EPESS) The Eurasia Proceedings of Educational& Social Sciences (EPESS)*, 15, 2019.

[10] Duanning Zhou, Ron C W Kwok, Quan Zhang, and Jian Ma. A new method for student project assessment using fuzzy sets, 2001.

[11] Nia Amelia, Ade Gafar Abdullah, and Yadi Mulyadi. Meta-analysis of student performance assessment using fuzzy logic. *Indonesian Journal of Science and Technology*, 4:74–88, 2019.

[12] Gülçin Büyüközkan and Da Ruan. Evaluation of software development projects using a fuzzy multi-criteria decision approach. *Mathematics and Computers in Simulation*, 77:464–475, 05 2008.

[13] Aron Karatayev, Anna Ogorodova, and Pakizar Shamoi. Fuzzy inference system for test case prioritization in software testing, 2024.

[14] Lijia Chen, Pingping Chen, and Zhijian Lin. Artificial intelligence in education: A review. *IEEE Access*, 8:75264– 75278, 2020.

[15] Alexandara Harry. Role of AI in Education. *Interdiciplinary Journal and Hummanity (INJURITY)*, 2023.

[16] E. Grace, Lecturer P. Vidhyavathi, and Lecturer P. Malathi. A STUDY ON "AI IN EDUCATION: Opportunities AND CHALLENGES FOR PERSONALIZED LEARNING. *Industrial Engineering Journal*, 2023.

[17] Oyebola Olusola Ayeni, Nancy Mohd Al Hamad, Onyebuchi Nneamaka Chisom, Blessing Osawaru, and Elizabeth Adewusi. Ai in education: A review of personalized learning and educational technology.

[18] Budee U Zaman. Transforming Education Through AI, Benefits, Risks, and Ethical Considerations. 2023.

[19] Mayank Chhatwal, Vikas Garg, and Nimit Rajput. Role of AI in the Education Sector. *Lloyd Business Review*, 2023.

[20] Nixon Duarte, Yury Montoya Pérez, A. Beltran, and Matilde Bolaño García. Use of Artificial Intelligence in Education: A Systematic Review. *Proceedings of the International Conference on Industrial Engineering and Operations Management*, 2023.

[21] Lijia Chen, Pingping Chen, and Zhijian Lin. Artificial intelligence in education: A review. *IEEE Access*, 8:75264–75278, 2020.

[22] R. Ilahi, I. Widiaty, and A. G. Abdullah. Fuzzy system application in education. *IOP Conference Series: Materials Science and Engineering*, 2018.

[23] Sergio Valdés-Pasarón, Bogart Yail Márquez, and Juan Manuel Ocegueda-Hernández. Methodology for Measuring the Quality of Education Using Fuzzy Logic. *International Conference on Software Engineering and Computer Systems*, 2011.

[24] Yuniana Cahyaningrum Cahyaningrum. Penerapan Artificial Intelligence Menggunakan Fuzzy Logic dalam Dunia Pendidikan. *JURNAL AMPLIFIER : JURNAL ILMIAH BIDANG TEKNIK ELEKTRO DAN KOMPUTER*, 13(2):62– 68, nov 22 2023.

[25] Mounira Chkiwa, Moez Chkiwa, and Fatma Achour. Student Knowledge Evaluation System: a case of application of Fuzzy logic in Intelligent Education. *International Symposium on INnovations in Intelligent SysTems and Applications*, 2023.

[26] A. Baba, F. M. Cin, and Didem Bakanay. A fuzzy system for evaluating students' project in engineering education. *Computer Applications in Engineering Education*, 2012.

[27] Spyros Papadimitriou, Konstantina Chrysafiadi, and Maria Virvou. Evaluating the use of fuzzy logic in an educational game for offering adaptation. In *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*. IEEE, 8 2019.

[28] C. Fourali. Using Fuzzy Logic in Educational Measurement: The Case of Portfolio Assessment. 1997.

[29] Nihan Arslan Namli and Ozan Senkal. Using the fuzzy logic in assessing the programming performance of students.*International Journal of Assessment Tools in Education*, pages 701–712, 10 2018.

[30] Nne R Saturday and Friday E Onuodu. Evaluation of students' performance using fuzzy logic, 2019.

[31] Tracy Adeline Ajol, Shirley Sinatra Gran, Agnes Kanyan, and Siti Farah Lajim. An enhanced systematic student performance evaluation based on fuzzy logic approach for selection of best student award. *Asian Journal of University Education*, 16:10–20, 12 2020.

[32] A. Fevzi Baba, F. Melis Cin, and Didem Bakanay. A fuzzy system for evaluating students' project in engineering education. *Computer Applications in Engineering Education*, 20:287–294, 6 2012.

[33] Jian Ma and Duanning Zhou. Fuzzy set approach to the assessment of student-centered learning. *Education, IEEE Transactions on*, 43:237 – 241, 06 2000.

[34] Omid Haass and Gustavo Guzman. Understanding project evaluation – a review and reconceptualization, 5 2020.

[35] Alibek Barlybayev, Altynbek Sharipbay, Gulden Ulyukova, Talgat Sabyrov, and Batyrkhan Kuzenbayev. Student's performance evaluation by fuzzy logic. volume 102, pages 98–105. Elsevier B.V., 2016.

[36] Beyza Esin Ozseven and Naim Cagman. A novel student performance evaluation model based on fuzzy logic for distance learning. *International Journal of Multidisciplinary Studies and Innovative Technologies*, 6:29, 2022.
[37] Andysah Putera and Utama Siahaan. Determination of thesis preceptor and examiner based on specification of teaching using fuzzy logic, 2015.
[38] Tadeusz A. Grzeszczyk. Developing a new project evaluation systems based on knowledge. *Foundations of Management*, 5:59–68, 12 2013.
[39] Assylzhan Izbassar, Muragul Muratbekova, Daniyar Amangeldi, Nazzere Oryngozha, Anna Ogorodova, and Pakizar Shamoi. Intelligent system for assessing university student personality development and career readiness. *Procedia Computer Science*, 231:779–785, 2024. 14th International Conference on Emerging Ubiquitous Systems and Pervasive Networks / 13th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (EUSPN/ICTH 2023).
[40] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.
[41] Pakizar Shamoi, Atsushi Inoue, and Hiroharu Kawanaka. Modeling aesthetic preferences: Color coordination and fuzzy sets. *Fuzzy Sets and Systems*, 395:217–234, 2020. Aggregation Operations.
[42] L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning: i, ii, iii. *Inform. Sci.*, 8:199–251, 1975.
[43] Nitin Agarwal and Urvashi Rathod. Defining 'success' for software projects: An exploratory revelation. *International journal of project management*, 24(4):358–370, 2006.
[44] T.C. Lethbridge and R. Laganière. *Object-oriented Software Engineering: Practical Software Development Using UML and Java*. McGraw-Hill, 2001.
[45] Pakizar Shamoi, Atsushi Inoue, , and Hiroharu Kawanaka. Fhsi: Toward more human-consistent color representation. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 20(3), 2016.
[46] Shyi-Ming Chen and Chia-Hoang Lee. New methods for students' evaluation using fuzzy sets. *Fuzzy Sets and Systems*, 104(2):209–218, 1999.
[47] Liviu-Cristian Duţu, Gilles Mauris, and Philippe Bolon. A fast and accurate rule-base generation method for mamdani fuzzy systems. *IEEE Transactions on Fuzzy Systems*, 26(2):715–733, 2017.
[48] Meenakshi Bansal, Dinesh Grover, and Dhiraj Sharma. Sensitivity association rule mining using weight based fuzzy logic. *Global Journal of Enterprise Information System*, 9(2):1–9, 2017.
[49] Quang Hung Do and Jeng-Fung Chen. A neuro-fuzzy approach in the classification of students' academic perfor-mance. Computational Intelligence and Neuroscience, 2013, 2013.
[50] Antonio Cervero, Adrián Castro-Lopez, Luc á Álvarez-Blanco, Mar á Esteban, and Ana Bernardo. Evaluation of educational quality performance on virtual campuses using fuzzy inference systems. *Plos one*, 15(5):e0232802, 2020.
[51] Michael Gr. Voskoglou. Fuzzy logic as a tool for assessing students' knowledge and skills. *Education Sciences*, 3(2):208–221, 2013.
[52] Daniel Doz, Darjo Felda, and Mara Coticˇ. Assessing students' mathematical knowledge with fuzzy logic. *Education Sciences*, 12(4):266, April 2022.

Authors' Profiles

Anna Ogorodova earned a B.S. degree in information systems from Kazakh-British Technical University (KBTU) in Almaty, Kazakhstan. She completed her M.S. degree in software engineering at the same institution. Her academic contributions include participation in notable conferences such as KBTU AGSRW 2023, EUSPN 2023, and IEEE SIST 2024. She has also served as a teaching assistant at KBTU in 2023. Professionally, she holds a position as a Senior Software Engineer at a prominent state bank in Kazakhstan, and she mentors Java programming courses. Her research interests are artificial intelligence and machine learning, focusing on fuzzy sets and logic.

Pakizar Shamoi received the B.S. and M.S. degrees in information systems from the Kazakh-British Technical University, Almaty, Kazakhstan, in 2011 and 2013, and the Ph.D. degree in engineering from Mie University, Tsu, Japan, in 2019. She has 13 years of experience in teaching technical subjects to university students. In her academic journey, she has held various teaching and research positions at Kazakh-British Technical University, where she has been serving as a professor in the School of Information Technology and Engineering since August 2020. She is the author of 1 book and more than 33 scientific publications. Awards for the best paper at conferences were received six times. Her research interests include artificial intelligence and machine learning in general, focusing on fuzzy sets and logic, soft computing, representing and processing colors in computer systems, natural language processing, computational aesthetics, and human-friendly computing and systems. She took part in the organization and worked in the org. committee of several international conferences IFSA-SCIS 2017, Otsu, Japan; SCIS-ISIS 2022, Mie, Japan; EUSPN 2023, Almaty, Kazakhstan. She served as a reviewer at several international conferences, including IEEE: SIST 2023/2024, SMC 2022, SCIS-ISIS 2022, SMC 2020, ICIEV-IVPR 2019, ICIEV-IVPR 2018. Dr. Shamoi is an IEEE member and member of the presidium of the Council of Young Scientists of the Academy of Sciences of Kazakhstan.

Aron Karatayev received a B.S. degree in information systems from the Kazakh-British Technical University, Almaty, Kazakhstan. He completed his M.S. degree in software engineering at the same institution. He participated in conferences like KBTU AGSRW 2023 and IEEE SIST 2024. He is currently a PhD student. Professionally, he is a Senior Quality Assurance Engineer at a leading outsourcing company in Kazakhstan. His research interests include fuzzy logic and sets, software testing, and finance.

How to cite this paper: Anna Ogorodova, Pakizar Shamoi, Aron Karatayev, "Fuzzy Intelligent System for Student Software Project Evaluation", International Journal of Modern Education and Computer Science(IJMECS), Vol.17, No.4, pp. 26-44, 2025. DOI:10.5815/ijmecs.2025.04.02