**Annotation**

This diploma project is dedicated to the development of a web-based platform designed to support students in forming teams and finding academic supervisors for their diploma theses. The primary goal of the project is to provide a centralized, intuitive, and scalable solution to address the inefficiencies of traditional team formation and supervisor selection processes in academic settings.

To realize this vision, the project follows a user-centered development approach backed by modern technologies. The frontend is implemented using Vue.js for its dynamic and modular capabilities, while the backend relies on Django REST Framework to deliver reliable and maintainable APIs. Figma is used for building a clean and user-friendly interface, ensuring accessibility and responsiveness across devices. Real-time features such as notifications are supported through WebSockets, providing an enhanced communication experience for students and faculty.

Beyond the technical aspects, the project incorporates strategic planning elements including requirement analysis, cost estimation, and a comprehensive SWOT analysis. The proposed solution considers integration opportunities with university systems, potential scalability, and compliance with data protection standards. The result is a flexible and robust academic collaboration platform that empowers students to organize their thesis work more effectively and fosters better interaction between students and supervisors.

# 1 INTRODUCTION

The history of web systems began in the early 1990s with the advent of the World Wide Web (WWW), pioneered by Sir Tim Berners-Lee and his team at CERN, who have been responsible for the discovery of the WWW which got here at the side of different technologies with it, for instance, the Hypertext Transfer Protocol (HTTP), Hypertext Markup Language (HTML), and a system termed Identifiers referred to as Uniform Resource Locators (URLs). The advent of those technologies made it possible for the Web to act as a worldwide, or customary, community for the dissemination of information [1].

Marc Andreessen’s contribution to the invention of Mosaic in 1992, the first widely used web browser, was instrumental in establishing Web culture. Mosaic’s graphical interface made the Web much easier to navigate, leading to an exponential increase in its user base and cementing Web 1.0 as the “internet of cognition” [2]. During this period, web systems were mostly informational, offering limited interactivity and serving primarily as a digital library where users could access content but not contribute to it. As browsers with the use of protocols like HTTP and HTML improved, the Web became more accessible, transforming users into participants and paving the way for the growth of the Internet from Web 1.0 to Web 2.0.

The late 1990s and early 2000s saw the rise of Web 2.0, the second one era of the Web, referring to a set of net-primarily based totally technologies, which include blogs, wikis, audio-podcasting, video-podcasting, RSS feeds, social bookmarking and tagging, social networking, multimedia sharing, and so on [3]. While the specific technologies associated with Web 2.0 differ, key examples include AJAX, XML, and RSS, which enhance interactivity and data sharing.

Despite its advantages, Web 2.0 encounters challenges like data privacy issues, content moderation complications, and the management of misinformation. Still, its focus on user involvement and collective intelligence has significantly altered how individuals and organizations engage online, paving the way for future web advancements.

The 2010s introduced Web 3.0, characterized by the integration of artificial intelligence, machine learning, and semantic search technologies, allowing web systems to become more intelligent and responsive to user needs. Advanced frameworks and tools, such as React, Angular, and Vue.js, further enhanced the capabilities of web systems, enabling more complex and user-friendly interfaces. According to Gan et al., Web 3.0’s decentralized structure promotes new business models, disrupts platform monopolies, and fosters cultural co-creation. They also highlight its potential to power the Metaverse and AI-generated content, which could redefine digital interactions and content creation [4]. These technologies underpin the shift towards a user-centric internet, enabling applications that integrate data ownership, privacy, and scalability.

Academic web systems evolved alongside these trends. From simple portals providing course materials to comprehensive platforms like Moodle [5] and Platonus [6], educational institutions began integrating web systems to support course management, assessments, and collaboration. This progression highlights how web systems have evolved to support complex, interactive environments where users can work, learn, and connect across distances. Today, web systems are integral to almost every facet of professional, social, and academic life, and they continue to evolve to meet the growing demand for accessibility, collaboration, and innovation.

The goal of this diploma project is to develop a user-centered web platform called “Diplomatch”, that enables students to find compatible team members and supervisors for their diploma projects. By providing filtering options, profiles, and a collaborative interface, the platform aims to address the gap in existing academic systems, promoting a structured and accessible environment for project collaboration. Novelty of this project lies in its targeted approach to address specific needs in academic collaboration that are often overlooked by existing educational platforms. This platform not only supports students in achieving their academic goals but also enhances the overall experience of diploma projects, making them more engaging and productive for both students and mentors.

# 2 TECHNOLOGICAL SECTION

## 2.1. Research and Planning

Before starting development, a comprehensive study of existing direct and indirect competitors and Customer Development was conducted. The goal of research was to identify weaknesses of the existing products and define clear requirements and tasks for platform development.

### 2.1.1. State of the Problem

Currently, there is no centralized digital platform specifically designed to facilitate connections between potential teammates and supervisors for diploma thesis projects. Existing academic portals primarily focus on administrative functions, such as course enrollment and academic records management, but they lack interactive features that enable students to search for available supervisors, find team members with complementary skills, and collaborate effectively on research projects. Additionally, the absence of structured communication tools and recommendation systems often leads to inefficiencies, making the process time-consuming and challenging for students and faculty alike.

Two different surveys of academic supervisors and students were carried out to further support the difficulties that were found. The goal was to learn more about their individual experiences and difficulties in choosing supervisors and forming thesis project teams.

There are 48 responses from students. Several fundamental inefficiencies were identified by the student survey. Only a small percentage of students used official channels, including announcements or professor recommendations, to create teams; the majority (60%) depended on personal interactions. Furthermore, referring to the Figure 2.1.1.1 below, according to 73% of respondents, they had trouble assembling teams for a variety of reasons, including a lack of visibility into open positions, time restrictions, or difficulty finding qualified applicants. Another 64% relied on invites from outside sources such as group chats or cooperation with friends, whilst 36% actively sought out their teams. There is a discrepancy between expectations and results, as evidenced by the fact that only 47% of students were completely content with the teams they were assigned and 55% were totally satisfied with the project topic.



Figure 2.1.1.1 - “Did you experience any difficulties in finding a team? If so, what kind? (multiple options possible)”

Overall, only 8 supervisors passed the survey. There were obvious difficulties in the supervisor search process. Figure 2.1.1.2 may show understanding of 27% of students faced challenges such restricted availability of selected faculty, lack of knowledge regarding specialty, or mismatch of interests, despite 43% of students finding a supervisor within a week. Additionally, 64% of students said they were indifferent or unsatisfied with the existing supervisor and team selection process.



Figure 2.1.1.2 - “Did you face any challenges in choosing a supervisor? If so, what kind? (multiple options possible)”

Findings from the supervisor survey corroborate this opinion. Supervisors noted that students generally chose them based on broad interest areas or past teaching contacts, while 22% acknowledged a lack of student comprehension of academic specialty. Poor alignment of research interests (11%), and overworked professors (19%) are common challenges, as may be seen on Figure 2.1.1.3. When asked how the process might be made better, 35% of supervisors preferred structured consultations prior to selection, and 41% supported the use of preliminary interviews. Furthermore, everyone agreed that thesis topics ought to be the outcome of striking a balance between student initiative and supervisor approval.



Figure 2.1.1.3 - “What do you think are the main difficulties in choosing a supervisor? (multiple options possible)”

All these observations highlight the necessity of a centralized digital solution that guarantees openness, promotes collaboration, and expedites the hiring process. By including features like comprehensive supervisor profiles, automated matching based on academic interests, and secure communication tools, a well-designed platform could solve present inefficiencies. (See Appendices A–E for comprehensive survey questions and results.)

### 2.1.2. Planning

The flexible Agile (Scrum) project management methodology was taken as a basis for the LMS development process.This approach helps in adaptation to changing requirements, having a precise action plan. Therefore, the first step in building the web platform was to gather and analyse the requirements:

* The need for an intuitive search and filtering system for supervisors and team members;
* A structured application process for joining a project or team;
* Secure and efficient communication tools, including real-time messaging and notifications.

## 2.2. Platform Architecture

Table 1.2.1. – Architecture Overview

|  |  |  |
| --- | --- | --- |
| **Component** | **Description** | **Used Technology** |
| Frontend | Frameworks and libraries for creating user interfaces with modern UI/UX. | Vue.js (Vite), Figma |
| Backend | Server technologies for query processing, business logic and database operations. | Django |
| Database | Data storage and caching for better performance. | PostgreSQL |
| API | Ways of interaction between client and server. | RESTful API |
| Authentification | User authorisation and authentication methods, including security tokens. | JWT |
| CI/CD | Automate the deployment, testing and delivery of code to production. | Docker, GitHub |

This platform architecture is built for scalability, performance, and security, integrating modern technologies across different layers. The frontend utilizes Vue.js, with prototype created on Figma to ensure a dynamic and responsive UI. The backend is powered by Django to handle business logic and API requests efficiently. PostgreSQL serves as the relational database. RESTful API facilitates structured client-server communication. Authentication is managed using JWT for secure access control. CI/CD pipelines leverage GitHub and Docker to streamline deployment, ensuring smooth updates and system reliability. This architecture provides a robust foundation for modern web applications.

## 2.3. Database Modeling

### 2.3.1. SQL

A standard query language for accessing and manipulating data in databases, is called Sequence Query Language (SQL) [7]. Its compatibility with many different technologies makes SQL essential for organizing and accessing data in databases. From traditional Relational Databases Management Systems (RDBMS) to modern technologies including blockchain, artificial intelligence, and machine learning, SQL is absolutely vital. Whether data is stored in structured RDBMSs or other database kinds, SQL integration with DBMSs (Database Management Systems) allows user interaction with data.

The main way to communicate with a database is through SQL commands, which let users create, edit, and manage data in addition to running different types of analytical queries. These commands fall into four major categories: TCL (Transaction Control Language) is in charge of managing transactions, DCL (Data Control Language) is in charge of managing access permissions, DML (Data Manipulation Language) is in charge of handling data operations, and DDL (Data Definition Language) is used to define and modify the database structure [8]. In relational databases like PostgreSQL, SQL acts as a standard language that facilitates effective data searching, filtering, aggregation, and updating. All database management systems, including MySQL and SQL Server, rely on their underlying SQL engines to process and run queries, even though they offer user-friendly interfaces and APIs.

The SQL Engine, the main part in charge of query processing, converts these commands into actions. In order to communicate with the stored data, the SQL Engine [8] optimizes and runs SQL queries after parsing and compiling them. Additionally, the SQL Engine guarantees reliable and effective data retrieval and change.

Some of the most prominent advantages or benefits of Structured Query Language are Simple Approach, Efficient Execution, Standardization and lastly Scalability. SQL is easy to learn since the majority of its instructions and syntax are similar to those of regular English. Additionally, SQL adheres to a logical framework that facilitates comprehension and readability. Fast query execution is guaranteed by SQL's RDBMS optimization. A commonly used standard, SQL guarantees interoperability among various database systems. Large databases can be effectively managed with SQL, allowing for expansion without sacrificing efficiency.

### 2.3.2. Database Management System

Databases are used to store and manipulate data because they make it easy to save, organize, and process information later on. It was crucial to select trustworthy tools for database management, development, and design as part of this project.

A vital step in the architecture of any information system is selecting a database management system (DBMS). The DBMS is in charge of managing, processing, and storing data while guaranteeing its security, accessibility, and integrity. System performance, scalability, and data usefulness are all impacted by the DBMS selection. Every one of the several DBMSs has unique features, benefits, and drawbacks. A number of criteria need to be considered when selecting a database, including the kind of data, volume of information, performance needs, system integration capabilities, and cost.

Because they make it convenient to save, organize, and process information later on, databases are used to work with and store data. Selecting trustworthy technologies for database design, development, and management was crucial to this project.

For the project PostgreSQL was selected for use. PostgreSQL is an object-relational database management system (ORDBMS) based on POSTGRES, Version 4.2, developed at the University of California at Berkeley Computer Science

Department in 1986, with Professor Michael Stonebreaker acting as the supervisor [9]. Originally created as a research database with cutting-edge capabilities including extensibility and support for complicated data types, the project was funded by DARPA, ARO, and NSF.

POSTGRES pioneered many concepts that only became available in some commercial database systems much later. This choice was made for a few important reasons. One of the most widely used open-source relational databases is PostgreSQL, which is free to use and unrestricted by licensing. Transaction security and dependability are ensured by its complete adherence to the ACID standards (atomicity, consistency, isolation, and durability) since 2001 [10]. Working with a range of formats, different types of indexes, stored procedures, and triggers are just a few of the many functionalities that PostgreSQL enables. It is the best option for this project due to its great performance, scalability, and large user base. Many of the ideas introduced by POSTGRES were not accessible until much later in certain commercial database systems.

With speed enhancement and the addition of a SQL interpreter, POSTGRES changed to Postgres95 in 1994 [11]. The project was given its present name, PostgreSQL, in 1996, with an emphasis on SQL compatibility and the advancement of early concepts. Since then, PostgreSQL has continued to evolve and is still among the most sophisticated open-source databases.

An open-source variant of this original Berkeley code is called PostgreSQL. It provides many contemporary features and supports a significant portion of the SQL standard: complex queries, foreign keys, triggers, updatable views, etc. Additionally, users may expand PostgreSQL in a variety of ways, such as by adding supplementary data types, functions, operators, aggregate functions, index methods, and procedural languages [9].

The extensibility of PostgreSQL is one of its key benefits. You may modify the database to fit certain project needs by using its support for creating custom data types, functions, and operators. Additionally, it has defined APIs and supports a wide range of features, like indexes, which allow you to expand PostgreSQL to meet your needs [10]. Furthermore, in a multi-user setting, the system's provision for concurrent query execution greatly accelerates data processing. High availability and fault tolerance are essential for high-load systems, and PostgreSQL offers both with built-in replication capability.

### 2.3.3. Integrated Development Environment

For further data storage and control having an Integrated Development Environment (IDE) to create and maintain a database is crucial. Selecting trustworthy technologies for database design, development, and management was essential to this project. An important tool for database development is IDE. The IDE offers easy-to use tools for controlling the database structure and creating, running, and debugging SQL queries. Data analysis is made easier, repetitive tasks are automated, and developer productivity is raised in a good development environment.

When selecting an IDE, there are a few important factors to take into account. Supporting the appropriate database and having built-in tools for dealing with it, such as query debugging, SQL code auto-completion, and database structure visualization, are crucial. It's also important to consider the interface's ease of use, compatibility with version control systems, performance when handling big data sets, and extension or plug-in support.

DataGrip is a multi-engine database environment [12]. It provides database introspection and various instruments for creating and modifying objects for the supported engines. This software will enable you to work efficiently with SQL queries, analyze data, and administer the database. Working with many databases, including PostgreSQL, is supported by this robust tool. The application has an intuitive user interface, auto-completion of SQL queries, code analysis and refactoring capabilities, and version control system integration.

Code auto-completion, syntax highlighting, refactoring, query analysis, built-in debugging tools, SQL query performance analysis and connection monitoring, database structure visualization, results presented in an easy-to-read tabular format, flexible environment configuration, and plugin support are all features of this smart and handy SQL editor. Key features highlighted for Datagrip are Intelligent query console, AI Assistant, Code completion, Import/export in many formats, On-the-fly analysis and quick-fixes, Query history, Version control integration, UI and themes [13]. Datagrid makes database management easier, analyzes query performance, and automates repetitive tasks.

### 2.3.4. Database

Databases and data are the fundamental building blocks of new technology. Data is the building blocks of information, like numbers, words, pictures, and more, that computers use and process. Databases, on the other hand, are like organized libraries, making sure this data is stored, retrieved, and managed effectively [14]. A database is a structured collection of data that is organized in a way to facilitate efficient storage, retrieval, and manipulation of information. It acts as a centralized and organized repository where data can be stored, managed, and accessed by various applications or users.

This diplomas project database is made to handle information on professors, theses, students, and their teams. Because it was created with the relational model in mind, information can be processed, stored, and analyzed effectively. Students, supervisors, teams, diploma projects, and skills are the main entities in this database. Table relationships provide logical representation and data integrity.

This work uses Relational Database Management System (RDBMS), that organizes data into tables, where each table has rows and columns [15]. This database uses SQL for defining and manipulating data.

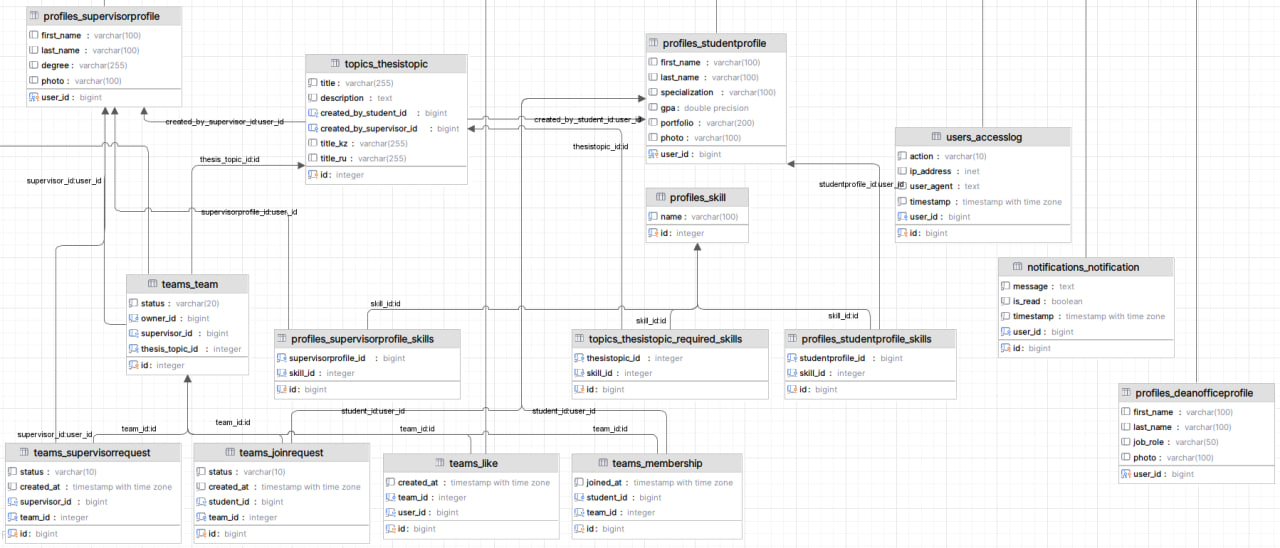


Figure 1.3.4. Database Diagram

Here, Figure 1.3.4.1. shows all existing tables, their attributes and relation between the tables through Primary and Foreign Keys.

#### profiles\_studentprofile table

**Purpose:** Stores key information about each student including personal details, academic performance, and identification for system-level operations. **Attributes:**

* user\_id (bigint, PRIMARY KEY, FOREIGN KEY) – Uniquely identifies the student and connects with the users table for authentication and user actions.
* first\_name / last\_name (varchar(100)) – Personal identification of the student, used in interfaces and reporting.
* specialization (varchar(255)) – The academic track of the student, useful for filtering candidates during team/topic matching.
* gpa (double precision) – The student’s academic performance metric, used for evaluation, ranking, or selection.
* portfolio (varchar(200)) – Link to personal academic or professional portfolio (e.g., GitHub, PDF), used for supervisor review.
* photo (varchar(100)) – Stores the path to the student’s photo for profile display.

#### profiles\_supervisorprofile table

**Purpose:** Captures basic profile data about supervisors for identification, filtering, and display in the system.  
**Attributes:**

* user\_id (bigint, PRIMARY KEY, FOREIGN KEY) – Unique identifier linking supervisors to system-wide user accounts.
* first\_name / last\_name (varchar(100)) – Supervisor identity used in all user-facing interactions.
* degree (varchar(255)) – Academic or professional degree held by the supervisor, useful for profile presentation.
* photo (varchar(100)) – Profile image path for visual interface.

#### profiles\_supervisorprofile table

**Purpose:** Stores thesis project topics proposed either by supervisors or students. **Attributes:**

* id (integer, PRIMARY KEY) – Unique identifier for the topic.
* title (varchar(255)) – General title of the thesis project.
* title\_kk / title\_ru (varchar) – Localized versions of the title for multilingual support.
* description (text) – Full abstract or idea of the thesis project.
* created\_by\_supervisor\_id / created\_by\_student\_id (bigint, FOREIGN KEY) – Indicates who proposed the topic.
* supervisor\_id (bigint, FOREIGN KEY) – Links the topic to the assigned supervisor.

#### profiles\_skills table

**Purpose:** Stores all possible skills that students or supervisors can associate with. **Attributes:**

* id (integer, PRIMARY KEY) – Unique identifier for each skill.
* name (varchar(100)) – The name or label of the skill (e.g., Python, Research Methods).

#### profiles\_studentprofile\_skills table

**Purpose:** Links students to their declared skills using a many-to-many relationship. **Attributes:**

* studentprofile\_id (bigint, FOREIGN KEY) – Links to a student profile.
* skill\_id (integer, FOREIGN KEY) – Links to a specific skill.

#### profiles\_supervisorprofile\_skills table

**Purpose:** Links supervisors to the skills or domains they specialize in.  
**Attributes:**

* supervisorprofile\_id (bigint, FOREIGN KEY) – Refers to the supervisor.
* skill\_id (integer, FOREIGN KEY) – Points to the corresponding skill.

#### topics\_thesistopic\_required\_skills table

**Purpose:** Stores which skills are required for a given thesis topic. **Attributes:**

* thesistopic\_id (integer, FOREIGN KEY) – The related thesis project.
* skill\_id (integer, FOREIGN KEY) – The required skill for successful completion.

#### teams\_team table

**Purpose:** Represents a team composed of students working together on a diploma project. **Attributes:**

* id (integer, PRIMARY KEY) – Unique identifier of the team.
* status (varchar(20)) – Current team status, e.g., ‘active’, ‘archived’.
* owner\_id (bigint) – Student who initiated or leads the team.
* supervisor\_id (bigint, FOREIGN KEY) – Supervisor assigned to the team.
* thesis\_topic\_id (integer, FOREIGN KEY) – Associated topic selected by or assigned to the team.

#### teams\_membership table

**Purpose:** Manages membership records for students in teams.  
Attributes:

* team\_id (integer, FOREIGN KEY) – ID of the team.
* student\_id (bigint, FOREIGN KEY) – ID of the student.
* joined\_at (timestamp with time zone) – Records when the student joined the team.

#### teams\_supervisorrequest table

**Purpose:** Logs supervisor responses to team collaboration requests. **Attributes:**

* team\_id (integer, FOREIGN KEY) – The team making the request.
* supervisor\_id (bigint, FOREIGN KEY) – The supervisor to whom the request was sent.
* status (varchar(10)) – Indicates if the request was accepted, rejected, or pending.
* created\_at (timestamp) – Timestamp of the request creation.

#### teams\_joinrequest table

**Purpose:** Handles student requests to join an existing team. **Attributes:**

* team\_id (integer, FOREIGN KEY) – Target team.
* student\_id (bigint, FOREIGN KEY) – Requesting student.
* status (varchar(10)) – Status of the request (approved, pending, etc.).
* created\_at (timestamp) – Date the request was made.

#### teams\_like table

**Purpose:** Allows users to express interest in teams, possibly used for visibility or ranking. **Attributes:**

* team\_id (integer, FOREIGN KEY) – Team that was liked.
* student\_id (bigint, FOREIGN KEY) – Student who liked the team.
* created\_at (timestamp) – When the like was made.

#### users\_accesslog table

**Purpose:** Tracks login and action history for auditing and behavioral insights. **Attributes:**

* user\_id (bigint, FOREIGN KEY) – Linked user.
* action (varchar(100)) – The action performed (e.g., login, update).
* ip\_address (inet) – Origin IP of the session.
* user\_agent (text) – Browser/device metadata.
* timestamp (timestamp with time zone) – When the action occurred.

#### notifications\_notification table

**Purpose:** Stores notification messages sent to users. **Attributes:**

* user\_id (bigint, FOREIGN KEY) – Recipient user.
* message (text) – Content of the notification.
* is\_read (boolean) – Indicates if the message has been read.
* timestamp (timestamp with time zone) – Time the message wassent.

#### profiles\_deanofficeprofile table

**Purpose:** Contains identity and role information for dean’s office staff. **Attributes:**

* user\_id (bigint, PRIMARY KEY, FOREIGN KEY) – Links to the user account.
* first\_name / last\_name (varchar(100)) – Dean office staff names.
* job\_role (varchar(50)) – Title or position in the administration.
* photo (varchar(100)) – Path to staff photo.

This database structure ensures data integrity, logical interconnection between students, professors, teams, and theses. Each property contributes significantly to the information transmission across tables, making data processing and administration easier. Data types are chosen based on accurate data processing, speedy access, and the effectiveness of information storage. Integer IDs facilitate data indexing and linking, varchar and text offer practical text information storage, while char[] readability and simplicity of use when working with established values.

### 2.3.5. Entity Relationship Diagram

Entity Relationship Diagram (ERD), also known as entity–relationship model, describes interrelated things of interest in a specific domain of knowledge. A basic ERD is composed of entity types and specifies relationships that can exist between entities [16].

The basic elements of ERD include entities, attributes, and relationships. Entities are objects whose data is stored in the database (for example, students, professors, teams). Attributes are the characteristics of entities (for example, full\_name, gpa, email). Relationships – relationships between entities (for example, "a student belongs to a team").

There are also 3 types of relations [17] being considered:

One-to-one (1:1) – there may only be one supervisor per diploma project.

One-to-many (1:N) – multiple theses can be supervised by a single professor.

Many-to-many (M:N) – if students could work on multiple diploma projects.

Apart from connectivity, a Primary Key (PC) and Foreign Key (FK) are essential for a good connection. Each record in the table is uniquely identified by PK characteristic or collection of attributes. Each row's uniqueness is ensured by the PK, which is also immune to NULL values. An attribute that establishes a connection between two tables is called an FK. It guarantees data integrity and makes reference to another table's primary key.

The PK and FK that guarantee data integrity are displayed in the ERD diagrams along with the relationships between the database's elements. For this project, the ERD reflects how students, professors, teams, theses, and skills are connected. Student\_id in the Students table, professor\_id in the Professors table, and team\_id in the Student\_team table are a few instances of primary keys in this database. These keys are used to avoid data duplication and guarantee each entity's unique identity.

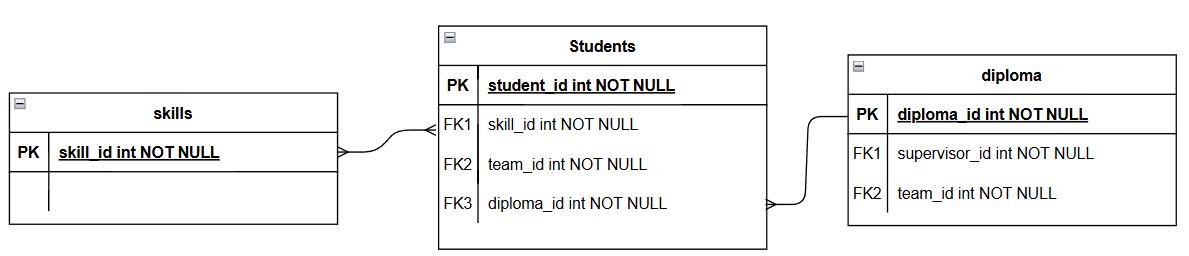


Figure 1.3.5.1. Skill - Student - Diploma ERD

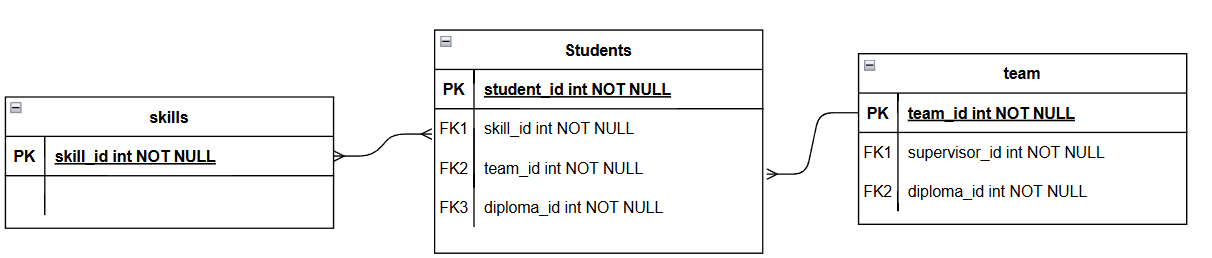


Figure 1.3.5.2. Skill - Student - Student\_team ERD

The relationship between Students and Diplomas, depicted in Figure 1.3.4.1, is designated as "Many-to-One" (N:1), which means that each student can only be associated with one diploma project, but one project can include several students. The direct relation suggests that each student has a personal attachment to projects. But the Students to Student\_team to Diplomas approach makes more sense. Because the diploma project is being completed by a group of students, the diploma is displayed in the diagram at the bottom. Furthermore, the link between students and skills is described as "Many-to-Many" (M:N), as a student may possess several skills and a single skill may belong to multiple students. Students can also be associated with teams using team\_id, which allows them to be grouped to participate in graduation projects.

The entities Skill, Students, and Student\_team are shown in the Figure 1.3.5.2. Similar to Figure 1.3.5.1, the relationship between Skill and Students shows that multiple students can possess the same skill. Student and Student\_team have a “Many-to-One” (N:1) connection, meaning that each student is a member of only one team, yet each team may include more than one student. Additionally, the relationship between Student\_team and Diplomats is depicted in this diagram, where each team is associated with a single diploma project via the foreign key diploma\_id. This indicates that a group, not a single student, is given the project.

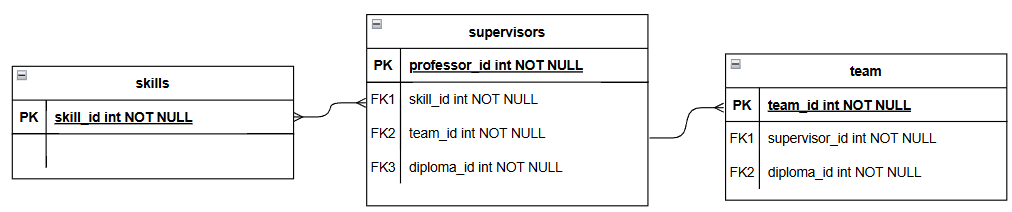


Figure 1.3.5.3. Skill – Professor – Student\_team ERD

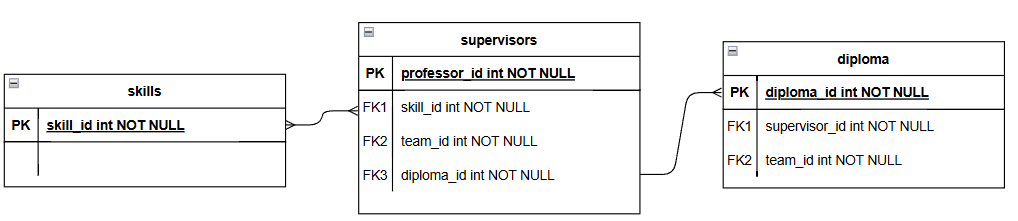


Figure 1.3.5.4. Skill – Professor – Diploma ERD

Skill, Professors and Student Teams are included in Figure 1.3.5.3 of the new pair of diagrams: Student\_team, Professors, and Skill. The link between professors and skills is described as "Many-to-Many" (M:N), as a professor may obtain several skills and a single skill may belong to multiple professors. Professors and Student\_team have a "Many-to-One" (N:1) connection, meaning that while each student team is under the supervision of a single professor, a single professor can oversee several teams. Additionally, Professors have a FK called diploma\_id that links them to the diploma project they are working on.

As seen by the link between Professors and Skills, as inFigure 1.3.5.3, more than one professor may have the same skill. The relation described in Figure 1.3.5.4 between Professors and Diploma also notes that the supervisors are in charge of various projects. Professors and Diplomats have a “Many-to-One” (N:1) connection, meaning that while a single professor can oversee several diplomas, each project is overseen by a single professor. Additionally, Diplomats has a team\_id foreign key that identifies the student team working on the thesis.

## 2.4. Cybersecurity

To ensure the protection and security of server and user data, many security mechanisms have been used. To ensure the protection and security of server and user data, many security mechanisms have been used. User security is ensured through registration, authentication and password recovery.

Security in registration is the process of creating a new user and adding to the database. The user specifies a corporate email, password and confirms his password. After registration, looking at the user's email, the role of either Student or Supervisor or Dean's Office is automatically given. All passwords are stored in hashed form. For example, "password123" becomes “pbkdf2\_sha256$870000$tnBAxQQQrQhh4Q433J0f7yX$mH5VC/rT7EiPvKQv....” This is done using the built-in function in Django [18].

The authentication system verifies users through the information provided in their input. You must provide your email address together with your password on the login page. The system generates JWT(JSON Web Token)[19] access-token and refresh-token automatically when the entered data is valid to ensure system security. The system will lock users out for 5 minutes after three consecutive incorrect password attempts to prevent hacker access. Users who forget their passwords can restore access through the “Forgot Password” button. The system directs users to a different page where they must provide their email address to receive a password reset link which remains active for 15 minutes. Users must click the link before they can enter their new password which requires confirmation. Users must re-authenticate after password changes because the previous token becomes invalid.

JWT presents itself as a JSON object which serves as a secure method to transmit information between servers and clients. The server generates tokens which receive a secret key signature before the client uses them to authenticate their identity.

The JWT structure includes three essential elements: Header(Algorithm & Token Type), Payload(data), Signature(Verification). The Header part of the token specifies which algorithm should generate the signature. The useful information which the token contains is stored inside the Payload section. The token contains three applications with “sub” specifying the topic and “name” representing the user's identity and “admin” verifying the user's admin role. These fields will be useful when creating the token. The encoded header and payload get folded together to generate the signature. The final string gets hashed through the algorithm listed in the header.

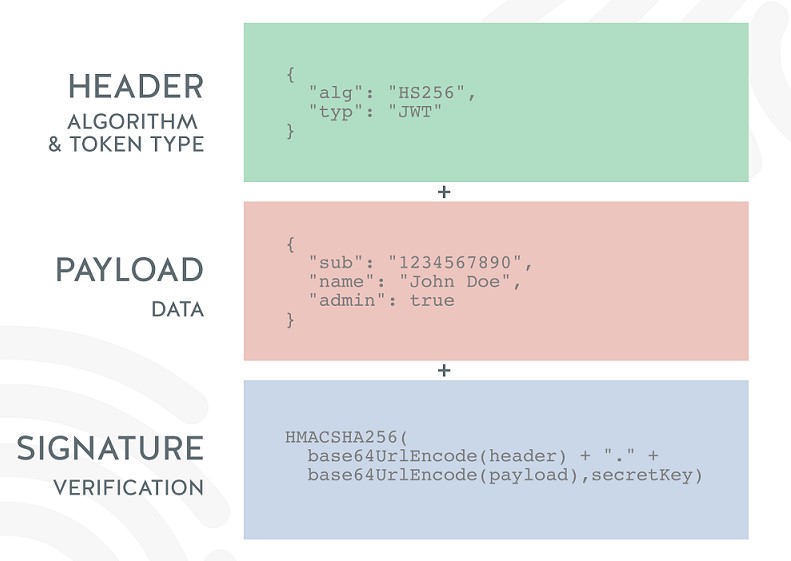


Figure 2.4.1. Structure of JWT

The final token combines all parts into “eyJhbGciOiJIUUzI1NiIsInInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiw ibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.DpeX5TOuCOWQbZ6M0lGr SFeLd6S-2aENJoQv”. After authentication success the server delivers the JWT token to the client. Two tokens are typically transmitted during this process. The access token functions as a request authentication mechanism for protected server resources [20]. It is reusable and short lived. The refresh token functions to obtain new access tokens and refresh tokens. The token exists as a single instance and maintains a prolonged duration [21]. The algorithm of applying access and refresh token is shown in Figure 2.4.2.

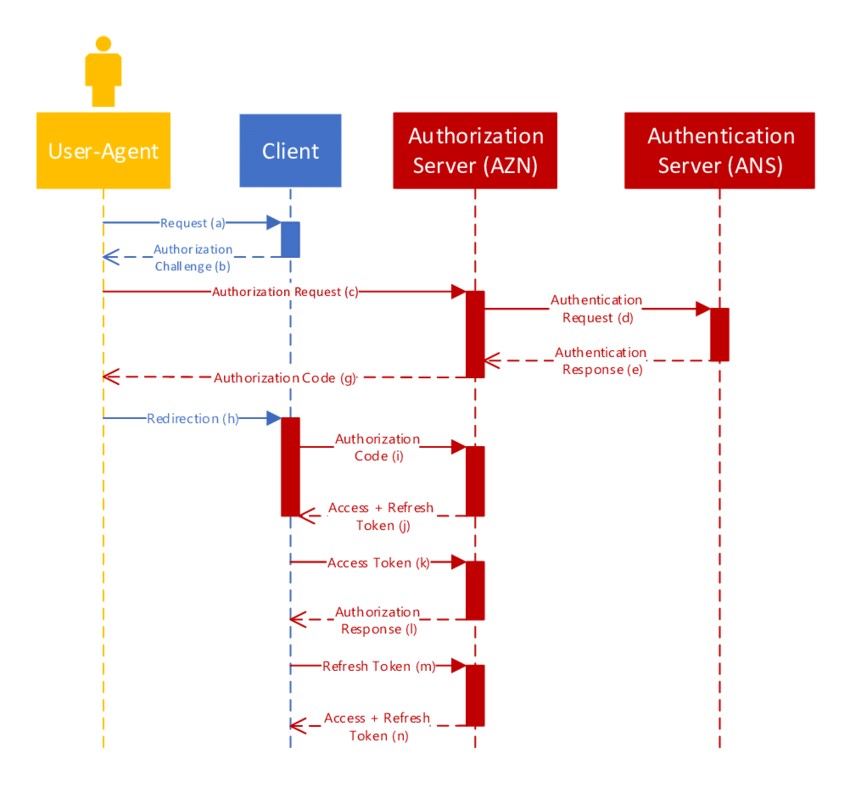


Figure 2.4.2. Access and Refresh Token

The authorization security system grants users access to specific actions based on their role. The system allows users to register as student, supervisor or dean office.

## 2.5. Backend Development

The backend of the website acts as the primary motor of the project. It oversees managing data storage, executing business logic, and real-time communication while enforcing secure relations with the frontend. To implement a modern, robust, and scalable control system, a composite of multi technologies was used such as Django, PostgreSQL, Docker, Redis, Django REST Framework, WebSocket (via Django Channels), and JWT security tokens.

Since speed was of essence for development, post construction procedure testing complex systems, Django was chosen to be the backbone web framework because of its evident architecture, usefulness, and incorporated functions. PostgreSQL was chosen to be the primary relational database management system considering its reliability, provides optimized performance, and supports advanced query capabilities over data. For precision and performance needs, Docker was employed for containerization for consistency cross developed environments and deployment directory while improving system maintenance and control.

For effortless communication between the front and backend, DRF was integrated into the project for establishing a simple flexible sustainable API to enable proper communication through RESTful interfaces: Framework for Django Representational State Transition enables batched request processing to reduce response times. Django Channel were employed for real time features like performing live updates or instant messaging using WebSocket. Redis served as an in-memory data store and message broker, significantly enhancing the performance of real-time operations and caching mechanisms.

### 2.5.1. Django and Rest API

The website's backend was coded utilizing Django — a well-known Python framework known for fast development, clean code organization, and having good security mechanisms. Django also allowed for simplicity in maintaining the project organized and adhering to best practices, especially when it came to interacting with the database and building user-related functionality [22].

A specialized user model was utilized to fit different user roles — Student, Supervisor, and Dean Office. Users were given a corresponding profile with personal information, avatar images, and a list of selected skills automatically based on their role. Special validation rules were applied to prevent users from selecting more skills than they could manage, and a profile completion flag was utilized to track whether users had finished filling in their data.

To connect the backend to the frontend, Django REST Framework (DRF) was used to develop a RESTful API [23]. Views and serializers were separated, which helped maintain the code clean and reusable. An arrangement like that helped make data validation, request processing, and returning standardized responses to the client easier.

API endpoints were formatted using Django routing mechanism, while DRF was responsible for the data serialization as well as request working logic. The API was used to allow core functionality including user registration, login, update profile, as well as role-specific actions. Everything was arranged to work fluidly with the frontend side of the application.

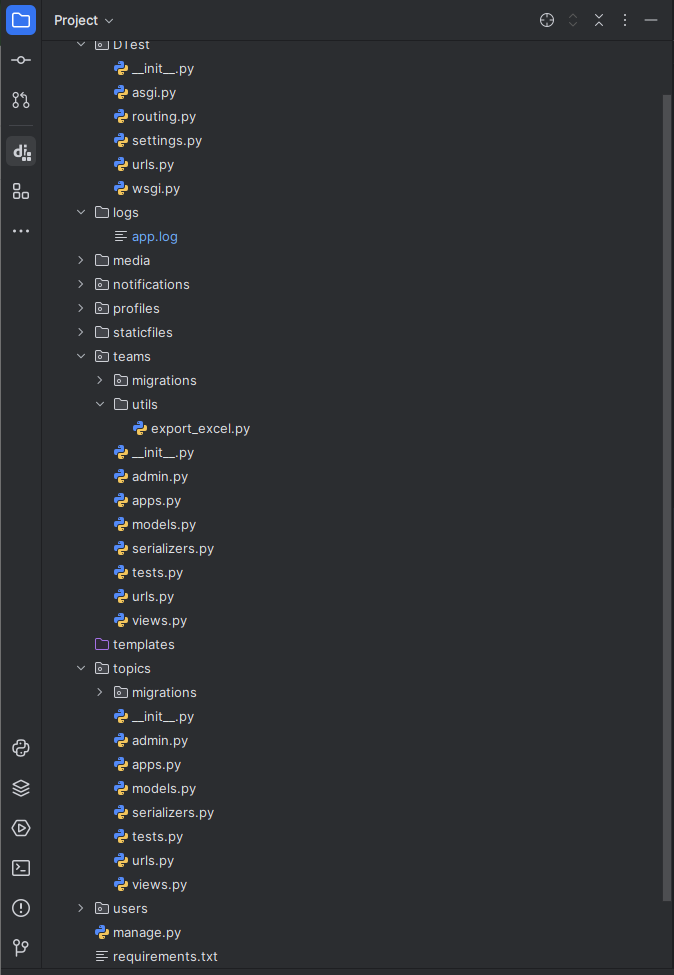


Figure 2.5.1. Folder structure of the backend using DRF

### 2.5.2 Real-time Notifications with Channels & Redis

Django Channels was implemented to allow real-time functionality including system alerts (such new join requests or supervisor answers). Asynchronous WebSocket communication was made possible by configuring Redis as a channel layer [24]. Every authorized user is associated with a distinct WebSocket group (user\_<id>), and in the case of an event, a notice is generated and, if the user is online, may be pushed in real time. Every notification has a time stamp, is associated with a particular user, and may be marked as read. All parties involved are kept updated on any changes or activities that need their attention through these alerts. By enabling the backend to transmit notifications without requiring a page refresh, this design greatly enhances user experience.

### 2.5.3 Team and Supervisor Matching Logic

A key functionality of the platform is the dynamic formation of teams around thesis topics, submitted either by students or supervisors. Each topic defines a list of required skills. Teams are linked to a thesis topic and can include multiple students. They can send requests to supervisors, and supervisors can approve or reject based on the team's qualifications. The system checks whether the team collectively meets the minimum required skills. Supervisors are limited to a maximum of 10 active teams.

### 2.5.4. Logging and Error Tracking

To identify backend operation and fix errors, the project has its own registration based on the built-in Django system. Logs go through two handlers - console and files, while all messages are specified in a separate log file logs/app.log in an extended format: with timestamps, importance levels and a detailed description of the events that occur [25]. The system records errors, user actions such as logging in and out of the system, changing IP addresses, password reset requests and other key events. Failed authorization errors are also recorded with the access code, IP address and user agent, which allows for prompt identification of suspicious activity. This approach allows for effective tracking of user behavior, finding weaknesses in the API logic and quickly preventing errors before they begin to affect the user experience. A particular rationale arose during the testing and deployment phase, when it was important to have a transparent picture of what was happening without the need for additional add-on services.

### 2.5.5. Working with E-mail

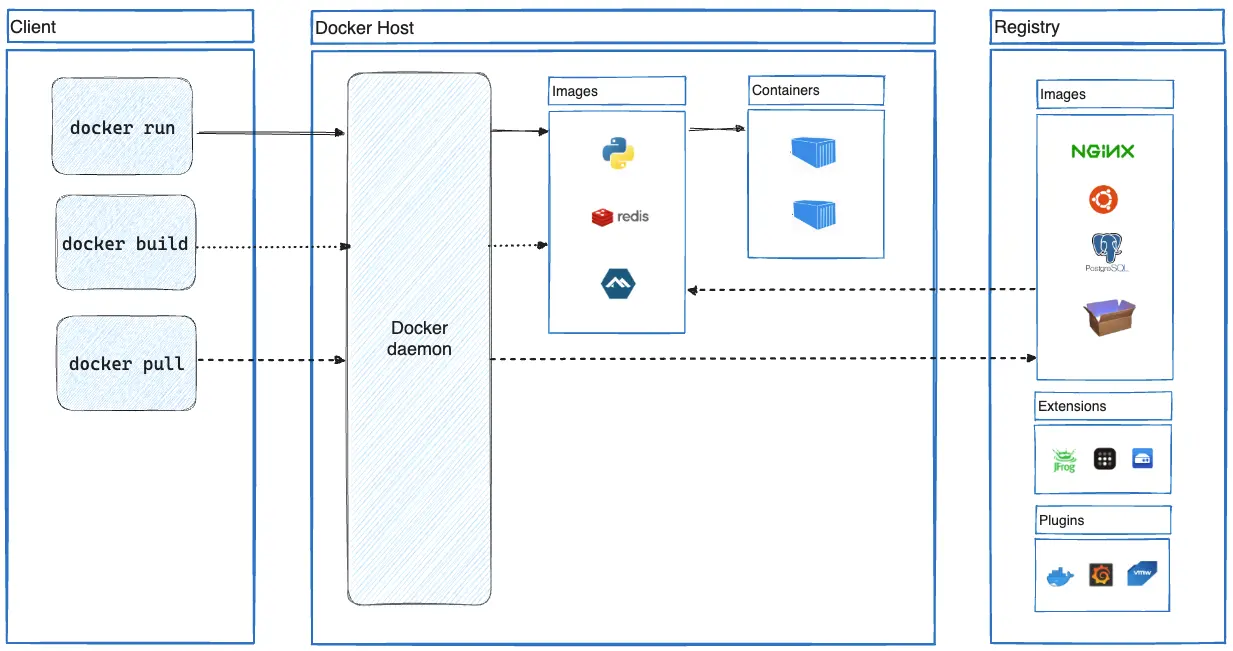
The project implements a system for sending emails via the Gmail SMTP server, which is used to restore access to the account [26]. When a user requests a password reset, the system automatically generates a unique token and forms a link leading to the corresponding page in the frontend. This link is sent to the user by email. Emails are sent using the built-in send\_mail function, and all events are logged: successful sending of an email, errors during generation or transmission, as well as attempts to reset to non-existent email addresses. This allows you to track the performance of the email functionality and quickly respond in case of failures. Security tokens are generated via the standard Django mechanism (default\_token\_generator), and the link includes an encrypted user ID and the token itself. After clicking the link and entering a new password, the data is checked, and if validated successfully, the password is updated. Thus, email integration plays an important role in the password recovery process and is built on proven and secure tools. The rejection of third-party solutions allowed us to maintain control over the sending logic without complicating the project architecture.

### 2.5.6. API Documentation

To simplify interaction with the API, detailed documentation was created using Swagger [27]. This allowed developers and testers to easily understand available endpoints, request methods (GET, POST) and data structure. The documentation provided examples of requests and responses, as well as describing possible errors and how to handle them. This approach greatly simplified the integration between backend and frontend and improved the quality of application testing.

### 2.5.7. Docker

Docker was used to make the development and deployment process straightforward [28]. Django, PostgreSQL database and ancillary services were packaged into containers, and all components of the application were also packaged into separate containers. This would guarantee their isolation and make their deployment more straightforward across platforms. Using Docker Compose enabled the automation of everything from startup to configuration of all services with just a single configuration, reducing the manual component and chance for human error. As a result, the entire team managed to establish a uniform development environment while also maintaining a flexible scaling of the application. Render served as a host while the deployment of containers to production required minimal effort.

 Figure 2.5.7. Docker Architecture

Docker works by running applications inside containers, which are created from something called images. An image acts like a template — it includes everything the application needs to run, such as code, libraries, and system tools. One of Docker’s key features is its layered file system: each change to the environment (like installing a package or modifying a file) is stored as a separate layer. This makes the system more efficient, since Docker can reuse existing layers instead of rebuilding everything from scratch. As a result, even small updates don’t require recreating the entire environment, which saves both time and resources.

## 2.6. Frontend Development

### 2.6.1. Requirements for UX/UI design

The first stage of development is UX/UI user interface design, as the usability of the platform determines its effectiveness. Requirements for the platform were composed according to the worldwide standards of UX/UI design. The prototype of the web platform was carried out according to Jakob Nielsen's 10 Heuristics for User Design Interface [29]:

##### 1. Visibility of system status

The design should always keep users informed about what is going on, through appropriate feedback within a reasonable amount of time.

2. **Match between the system and the Real World**   
The design should speak the users' language.

##### 3. User Control and Freedom

Users often perform actions by mistake. They need a clearly marked "emergency exit" to leave the unwanted action without having to go through an extended process.

##### 4. Consistency and Standard

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform and industry conventions.

##### 5. Error Prevention

Good error messages are important, but the best designs carefully prevent problems from occurring in the first place.

##### 6. Recognition Rather than Recall

The user should not have to remember information from one part of the interface to another.

##### 7. Flexibility and Efficiency of Use

Shortcuts — hidden from novice users — may speed up the interaction for the expert user so that the design can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

##### 8. Aesthetic and Minimalist Design

Interfaces should not contain information that is irrelevant or rarely needed. Every extra unit of information in an interface competes with the relevant units of information and diminishes their relative visibility.

##### 9. Help Users recognize, diagnose, and recover from Errors

Error messages should be expressed in plain language (no error codes), precisely indicate the problem, and constructively suggest a solution.

##### 10. Help and Documentation

It’s best if the system doesn’t need any additional explanation. However, it may be necessary to provide documentation to help users understand how to complete their tasks.

Also, in the design process was applied the psychology of color, according to “Web UI Design for the Human Eye” by J. Cao, K. Zieba, K. Stryjewski, M. Ellis [30]. Depending on the shades chosen, the user's perception of the content also varies. Shades of blue were chosen as the main color scheme for the overall style, which reflects friendliness and evokes a sense of trust, as well as being associated with technology and education. Based on global practices (Facebook, VK), it can be argued that the use of blue in the interface has a positive impact on the user experience.

For the action buttons on the platform were chosen different shades of green, yellow, red and gray. These shadows trigger users to act depending on the use-case.

### 2.6.2. Figma Prototype

This real-time, all-in-one environment is a boon for design professionals, who have long relied on cobbled-together workflows involving multiple programs, plug-ins, and cloud services [34]. Cloud-based Figma tool ensured flexible and convenient teamwork on the platform design, having real-time collaboration features, that allow multiple members work simultaneously.

The user interface (UI) of the platform was designed and a prototype of the platform was created using Figma which provided a clear visual representation of the structure and key features of the platform. The design process began with low-fidelity wireframes that defined the basic layout and functionality of the platform for future iterations. The wireframes were later developed into high-fidelity prototypes that included buttons, navigation menus, and progress indicators. The platform’s collaborative capabilities allowed for the efficient development of an aesthetically pleasing and user-friendly interface.

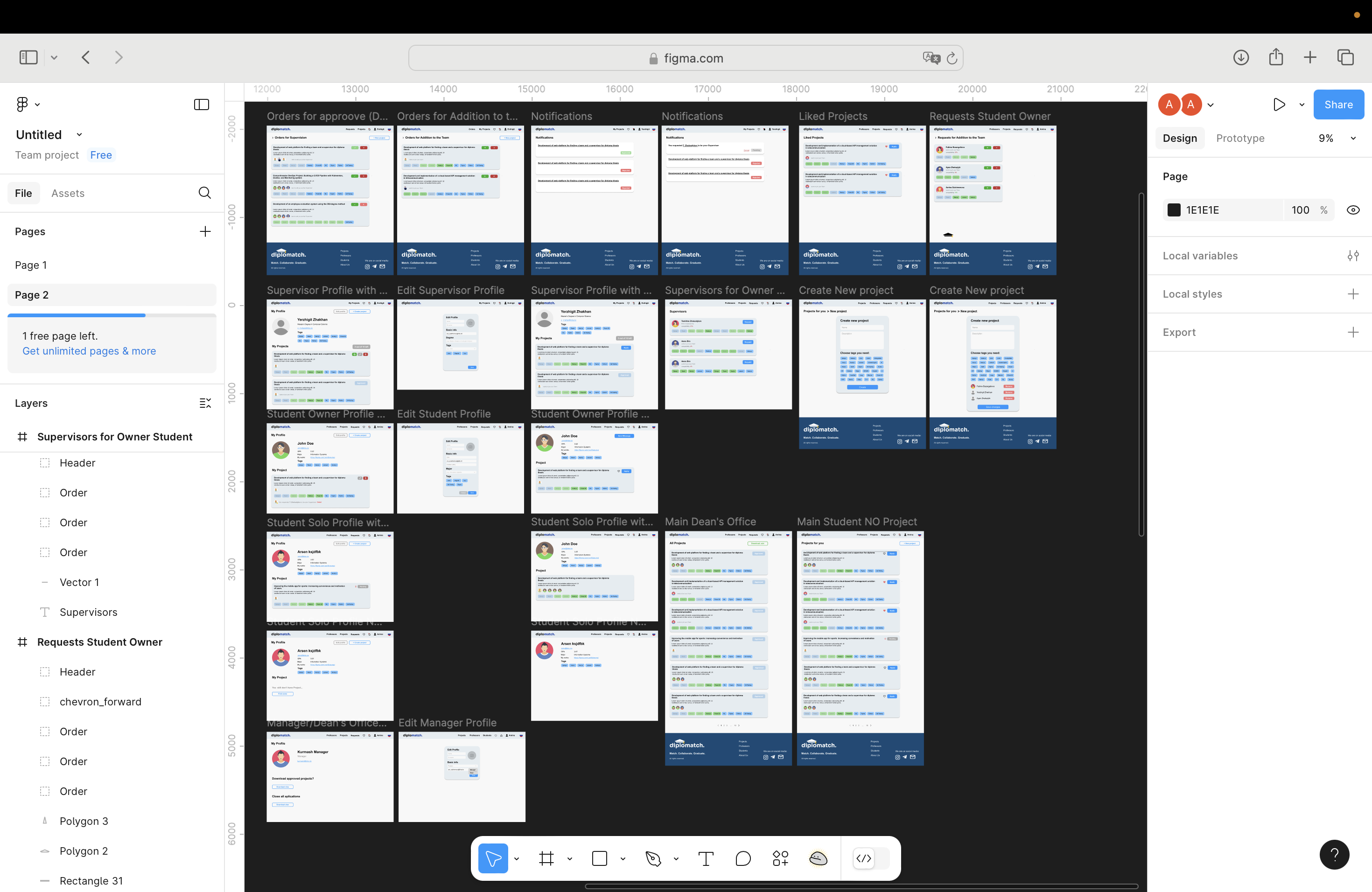
****

Figure 1.6.1.1. UX/UI design on Figma

The Web-platform consists of:

* **Authorization and registration pages for Student and Supervisor users.**

To work with the platform the user needs to authorize as each account has different authority.

Figure 1.6.1.2. Registration page

* **Main page**

From the Main page user can see the available Projects. Clicking on the logo refers to the Projects page

Figure 1.6.1.3. Main page (Projects Page) for Student

* **Student's Profile**

Includes the information necessary to analyze the competence of the Student. The user can edit their profile (excluding the skills).

Figure 2.6.1.4. Student Profile View

* **Supervisor’s Profile**

Viewing a Supervisor's Profile helps to understand what skills they are focused on and allows them to view active Projects. The user can edit their profile.

Figure 1.6.1.5. Supervisor’s Profile View

* **Supervisors page**

Contains a list of teachers available to supervise projects. Matching skills marked with green color. Compatibility labelled in percentage, calculated as less than 33% - low, from 33% to 67% - medium and more than 67% - high compatibility.

Figure 1.6.1.6. Supervisors Page

Compatibility labelled in percentage, calculated as less than 33% - low, from 33% to 67% - medium and more than 67% - high compatibility.

* **Liked Projects**  
  Contains a list of liked projects, with the built “Apply” feature.

Figure 1.6.1.7. Liked Projects

* **Notifications**

The notifications page shows the user which actions have been taken by other users towards him.

Figure 1.6.1.8. Notifications Page

### 2.6.3. Vite + Vue.js

Vite functions as a contemporary frontend build tool and development server which delivers optimized development efficiency. The tool delivers fast startup times and real-time updates which produce development speed improvements above traditional tools such as Webpack. During development Vite functions through native ES modules which modern browsers support to serve individual modules on demand. The application no longer requires upfront bundling of its entire codebase which cuts down page reload times to enable faster development cycles. The production phase of Vite relies on Rollup to create optimized and minified builds that result in a lightweight final application.

The development of Diplomatch platform benefited from Vite because it provided instant coding feedback and interface updates which led to faster design iterations. The hot module replacement (HMR) feature of Vite allowed Vue components to update in real time which resulted in fast interface development without interrupting the workflow. The platform-maintained performance at all stages of development and deployment because Vite supported modern JavaScript frameworks and handled static assets effectively. Furthermore, Vite's low configuration requirements simplified setup, freeing up more time to concentrate on implementing essential features rather than maintaining the build environment.

In keeping with the project's objectives of creating an effective, scalable, and user-friendly academic tool, Vite's use greatly improved the platform's responsiveness, development speed, and technical quality.

3 SPECIAL SECTION

## 3.1. Survey

Two structured questionnaires (survey questions can be reached in Appendix A and Appendix B) form the basis of the analysis were conducted: one was given to faculty members who supervise diploma projects, and the other was given to graduating students who had just completed the team and supervisor search procedure. This analysis's goals are to assess the present system's efficacy and pinpoint trends, roadblocks, and variables affecting both groups' levels of satisfaction.

The method starts by looking at supervisors' and students' general satisfaction ratings. Gaining an understanding of these attitudes provides a starting point for assessing whether the system now enables efficient team and supervisor matching. Figure 3.1 and Figure 3.2 below show that supervisors indicate neutrality, with few expressing strong sentiments, whereas students show a nearly equal split between satisfaction and displeasure. This implies that while supervisors may be passively limited by the system's constraints, students are more immediately impacted by inefficiencies.

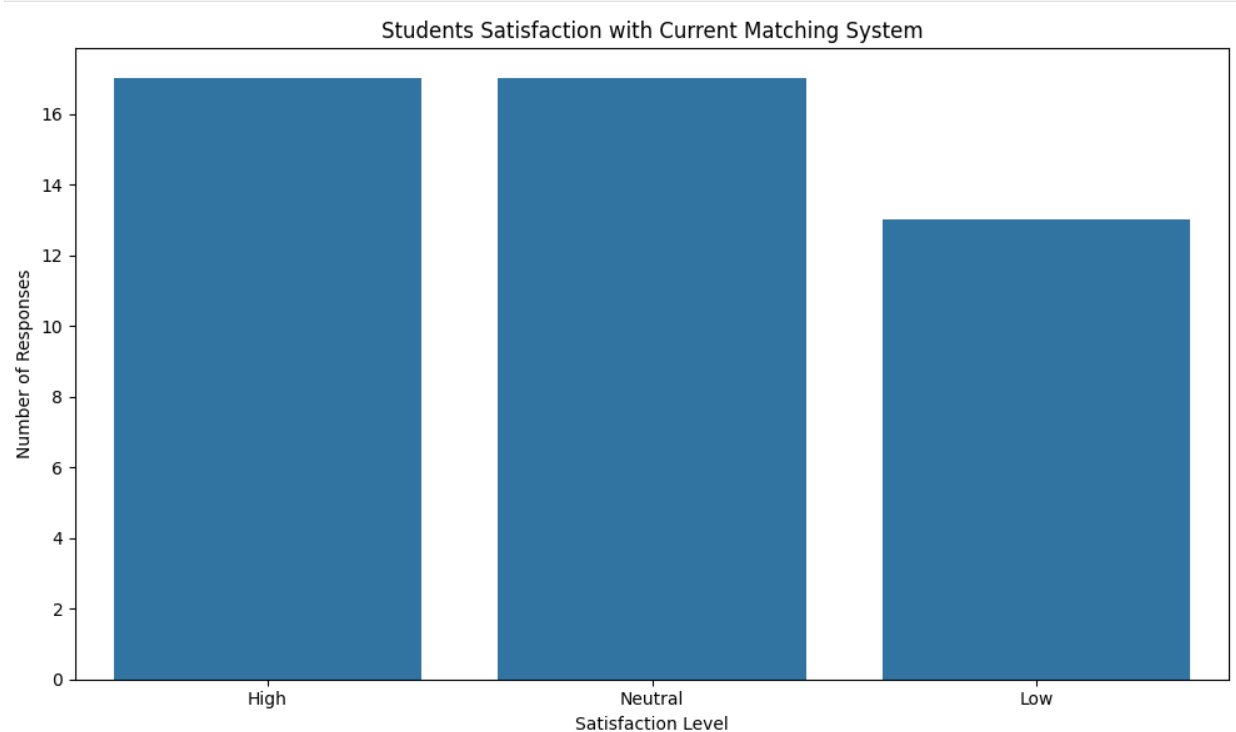


Figure 3.1.1 - Student satisfaction with current system

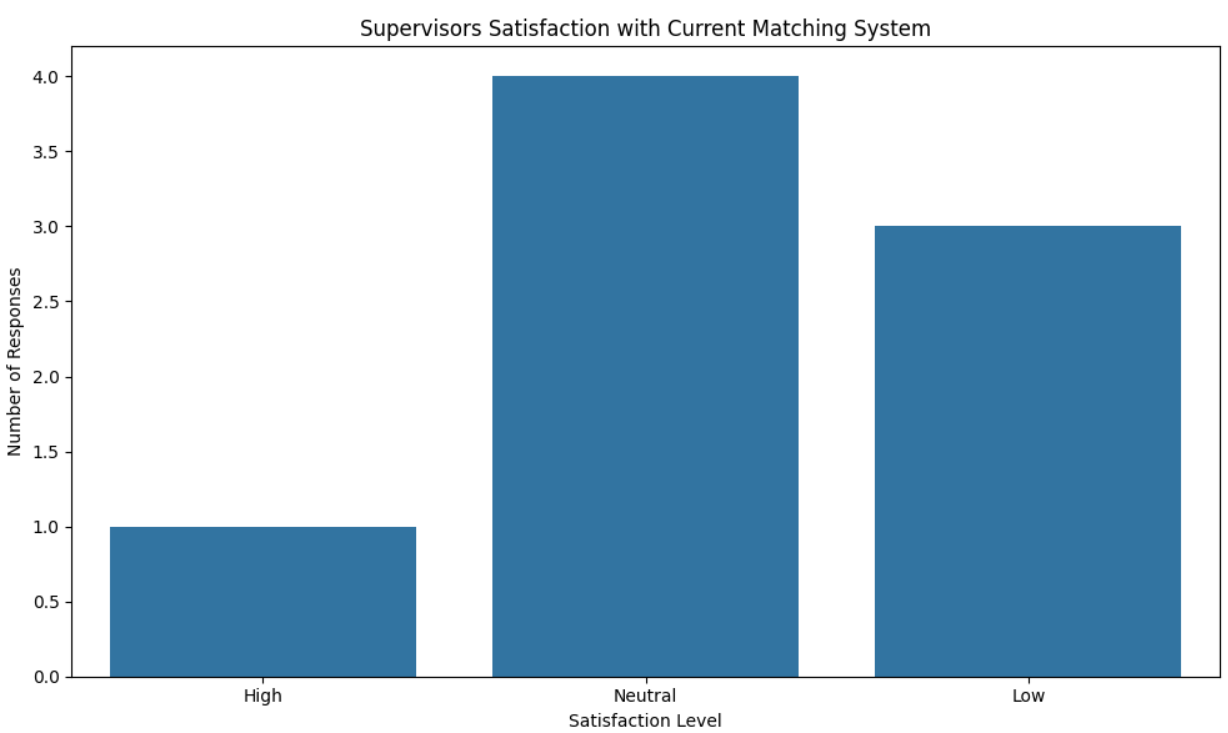


Figure 3.1.2 - Supervisors satisfaction with current system

As satisfaction is the main object in this project, it was observed with other three parameters that cover whole process experience. Three blocks were conducted during the survey, that cover major aspects in project development. There are Team, Supervisor, Topic and Overall satisfaction that collects all questions and answers into related blocks. To enhance understanding of the dataset and validate the scoring methodology, two bar charts were created that display the average satisfaction indices separately for students and supervisors. Figure 3.3 and Figure 3.4 consolidate the mean values for four satisfaction dimensions—Team, Supervisor, Topic, and Overall—based on the standardized scale previously discussed for both students and supervisors.

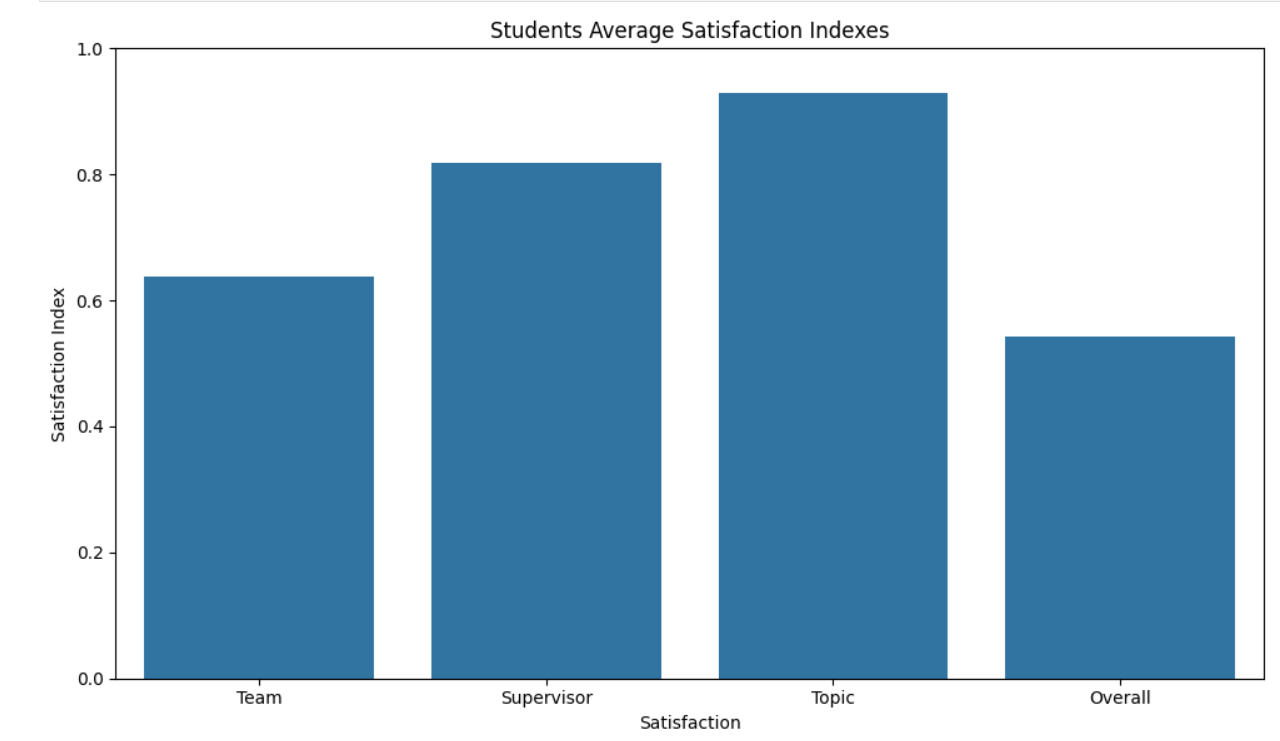


Figure 3.1.3 – Students Average Satisfaction indexes

The Figure 3.3 illustrates the students' average satisfaction levels across the main blocks of coordination. It is observed that students reported the highest satisfaction with the selected topic (0.93), followed by supervisor interaction (0.82). The team satisfaction score is comparatively lower (0.65), and the overall satisfaction—which likely captures a combination of all interaction experiences—drops further to 0.55. This discrepancy highlights that while certain aspects of the coordination process are functioning well (notably, topic selection), team dynamics and overall orchestration may require closer attention.

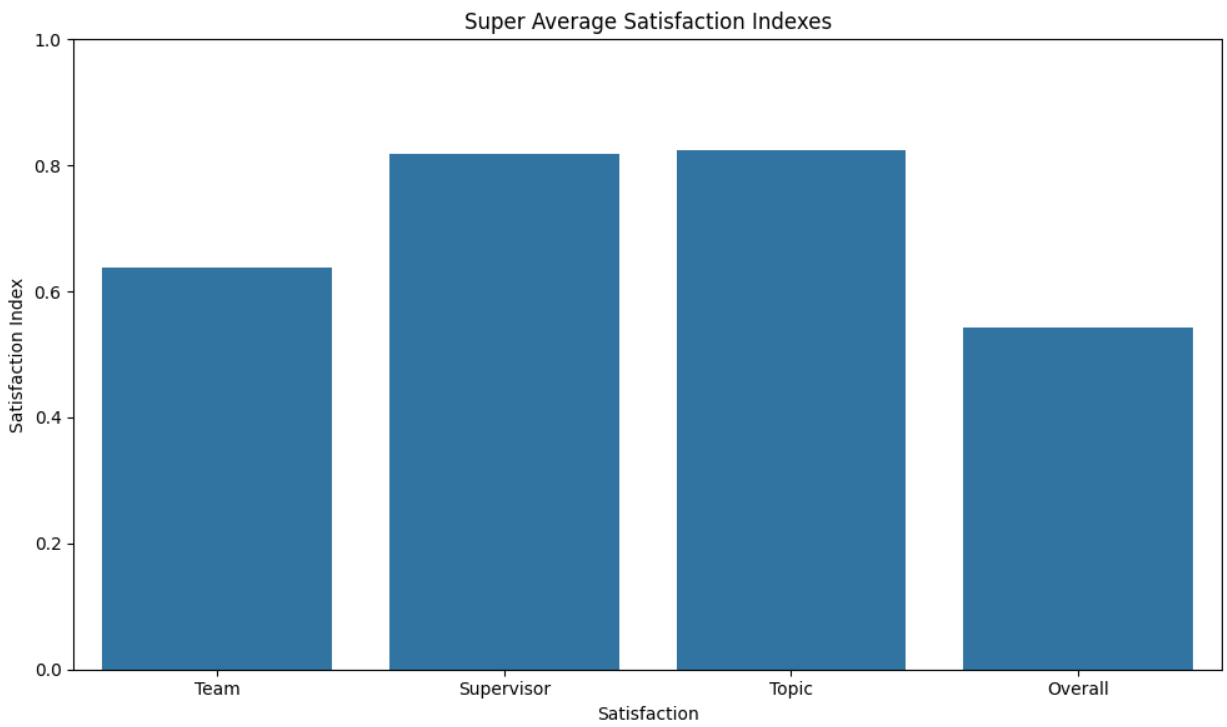


Figure 3.1.4 – Supervisors Average Satisfaction indexes

The supervisors' perspective on the same dimensions is shown in Figure 3.4. A similar trend is observed: topic and supervisor satisfaction remain high (approximately 0.83 each), indicating alignment between supervisors and the chosen academic themes or their role in the process. However, team-related satisfaction is again lower (around 0.64), and overall satisfaction stands at 0.54, matching the students' feedback. The parallel between both groups confirms that team formation and coordination pose a common challenge, negatively affecting holistic satisfaction.

These visualizations serve two critical purposes. First, they validate the consistency of data transformation from qualitative to quantitative formats. Second, they highlight priority areas for process improvement, guiding further statistical analysis and administrative decision-making.

Going deeper into analysis, the research examined the coordination process between students and supervisors in the context of diploma project formation, focusing on satisfaction levels, challenges encountered, and factors influencing key decision-making stages. A comprehensive analysis was conducted using various statistical techniques including correlation analysis, linear and logistic regression, and clustering methods. Data was collected through structured surveys distributed among students and supervisors.

## 3.1. Students

Starting on correlation analysis, between parameters and block’s satisfaction and overall satisfaction, this section presents a detailed examination of student-related survey data with the objective of uncovering patterns, predictors, and segmentation of student satisfaction in the diploma project coordination process. All analytical procedures—logistic regression, clustering, and supporting visualizations—are applied specifically to student responses. The aim is to determine how student experiences across various dimensions, such as team formation, supervisor interaction, and topic selection, contribute to overall satisfaction.

### 3.1.1 Correlation Analysis

To explore the internal dynamics of student satisfaction, a detailed correlation analysis was conducted within thematic blocks—Team, Supervisor, and Topic. The goal was to measure the linear relationship between each block’s input parameters and their corresponding satisfaction index, as well as to identify the global impact of these factors on Overall Satisfaction.

The first set of visualizations presents the correlations within each individual block. As seen in the three subplots below, each horizontal bar represents the strength and direction of correlation between a parameter and its related satisfaction measure.

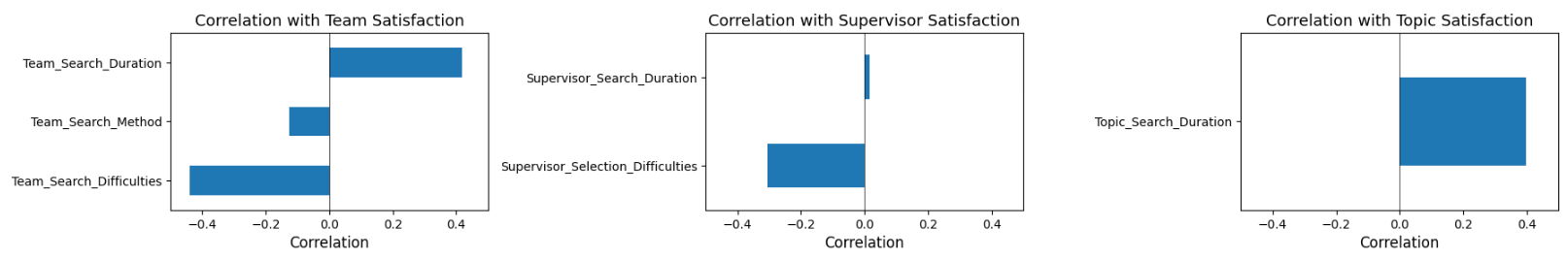


Figure 3.1.1.1 – Students Correlation with blocks

In the Team Satisfaction block, “Team\_Search\_Duration” shows a strong positive correlation with satisfaction, suggesting that students who spent more time finding a team reported better alignment or outcomes. Conversely, “Team\_Search\_Difficulties” has a significant negative correlation, indicating that complications in the team formation process directly undermine satisfaction.

The Supervisor Satisfaction block displays a similar trend. “Supervisor\_Selection\_Difficulties” negatively correlates with satisfaction, confirming that students facing more obstacles in supervisor selection are generally less content. “Supervisor\_Search\_Duration,” while present, demonstrates minimal impact.

The Topic Satisfaction block includes only one key input, “Topic\_Search\_Duration,” which positively correlates with topic satisfaction. This pattern might reflect a deeper engagement process, where more deliberate search efforts result in more appropriate or satisfying topic selections.

To expand the analysis beyond localized satisfaction, the Figure 3.6 evaluates the correlation between all parameters and Overall Satisfaction.

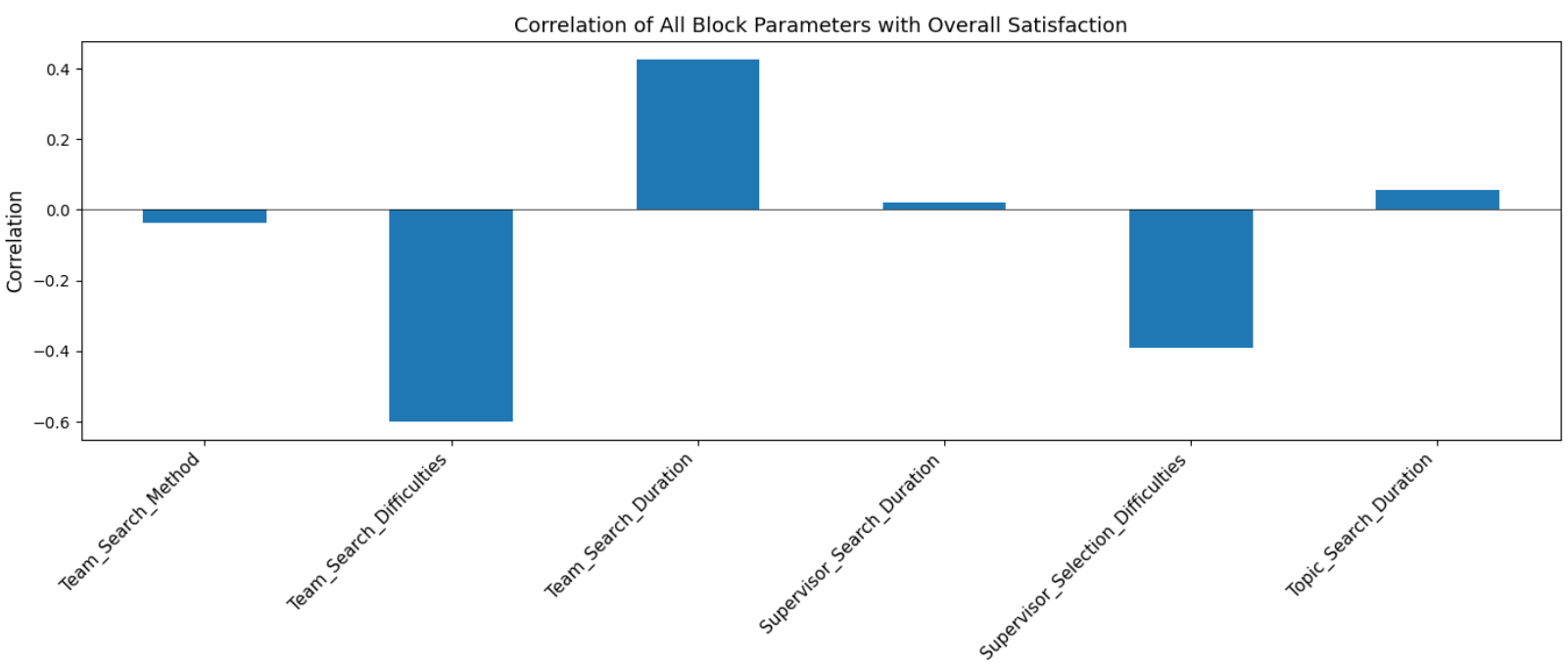


Figure 3.1.1.2 – Students Correlation with Overall Satisfaction

This aggregated correlation graph reveals broader system-wide patterns. Notably, “Team\_Search\_Difficulties” and “Supervisor\_Selection\_Difficulties” exhibit the most negative influence, both scoring below -0.4. These findings underline how logistical and interpersonal challenges during search and selection phases impact the general perception of the process. On the other hand, “Team\_Search\_Duration” displays a positive correlation of approximately 0.42 with Overall Satisfaction, reinforcing earlier insights from the team-specific block.

These visualizations play a crucial explanatory role, highlighting areas of improvement. Specifically, minimizing search-related difficulties and optimizing matching strategies may lead to measurable increases in satisfaction. The distinct patterns between different blocks also suggest the need for customized solutions depending on whether the issue lies within team dynamics, supervisor interactions, or topic selection processes.

### 3.1.2 Linear Regression

To further investigate the influence of operational parameters on student satisfaction, linear regression analysis was conducted for each thematic block: team formation, supervisor interaction, and topic selection. This approach aimed to quantify how strongly each factor contributes to the corresponding satisfaction index.

The first model targeted Team Satisfaction using five predictors: Team Search Method, Team Search Difficulties, Team Initiative, Team Criteria, and Team Search Duration. The results, visualized in the graph below, show that Team Search Duration and Team Initiative are the strongest positive predictors of satisfaction, while Search Difficulties and Search Method exhibit negative coefficients, suggesting that increased difficulties and passive methods of team formation reduce satisfaction.

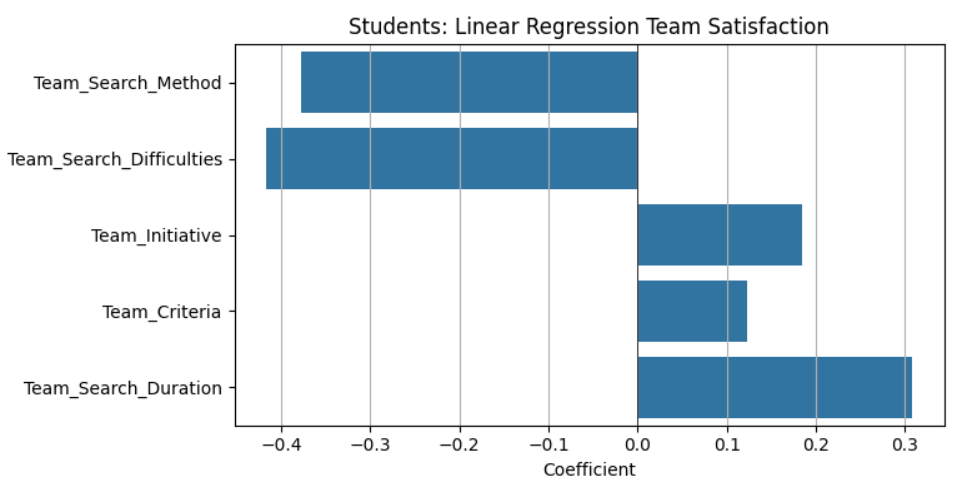


Figure 3.1.2.1 – Students Linear Regression on Team Satisfactions

For the Supervisor Satisfaction block, three variables were included: Supervisor Search Method, Supervisor Selection Difficulties, and Supervisor Search Duration. The graph illustrates that the use of proactive search methods (e.g., recommendations or department announcements) correlates positively with satisfaction. Conversely, selection difficulties and longer search durations contribute negatively.

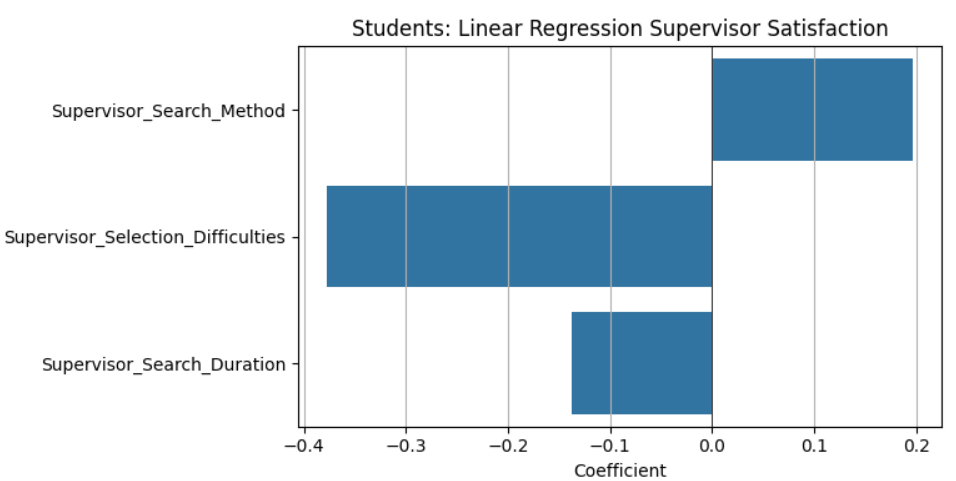


Figure 3.1.2.2 – Students Linear Regression on Supervisor Satisfactions

The Topic Satisfaction regression model included only one significant predictor: Topic Search Duration. As seen in the graph below, longer and presumably more deliberate topic selection processes are associated with higher satisfaction, as indicated by the positive regression coefficient.

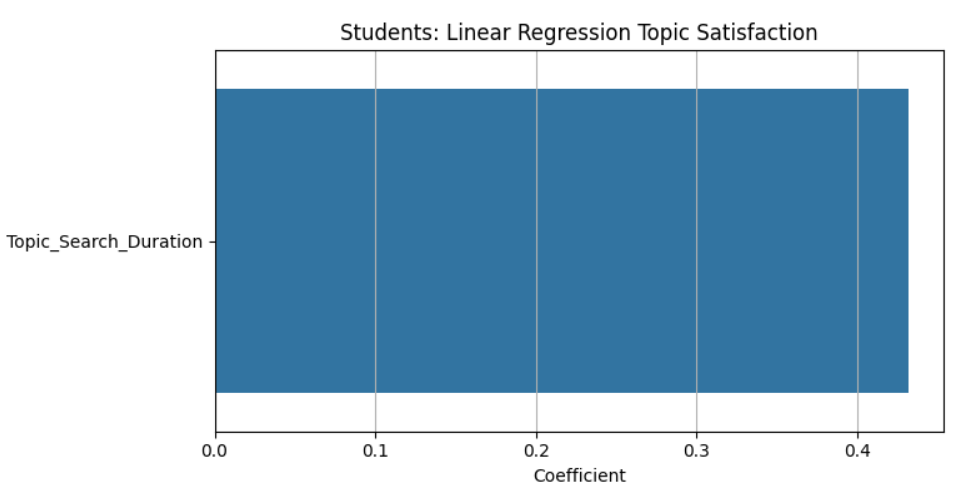


Figure 3.1.2.3 – Students Linear Regression on Topic Satisfactions

Finally, a combined model was constructed to assess the predictive power of each block's satisfaction score—Team, Supervisor, and Topic—on the Overall Satisfaction index. This model confirmed that all three components positively influence general satisfaction. Among them, Team Satisfaction held the strongest coefficient, indicating its central role in the student's overall experience.

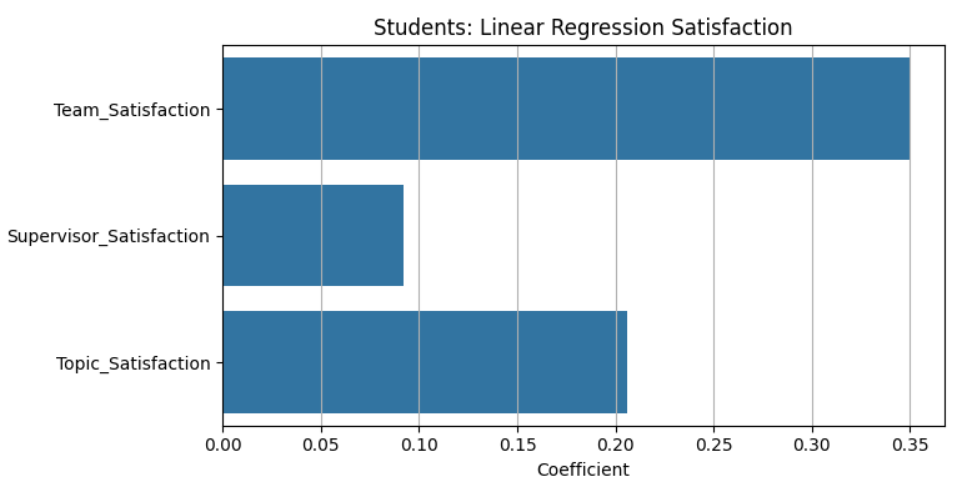


Figure 3.1.2.4 – Students Linear Regression on OverallSatisfactions

These findings reinforce the earlier correlation analysis and provide statistically grounded evidence for which aspects of the coordination process most significantly affect student satisfaction. They underscore the importance of minimizing procedural difficulties and encouraging proactive participation in team and topic selection processes.

Through the integration of binarized satisfaction indicators, scaled parameter scores, and machine learning techniques such as KMeans clustering, the analysis reveals both statistical relationships and behavioral profiles among student respondents. The following subsections demonstrate how these tools are used to quantify satisfaction dynamics, visualize group structures, and identify practical areas for system improvement from the student perspective.

### 3.1.3 Logistic Regression

In addition to linear models, logistic regression was utilized by binarizing satisfaction outcomes (satisfied vs. not satisfied). This approach provided a probabilistic view of how specific conditions affect the likelihood of satisfaction. Logistic regression results echoed linear trends, with clearer binary separations indicating where interventions could be most effective.

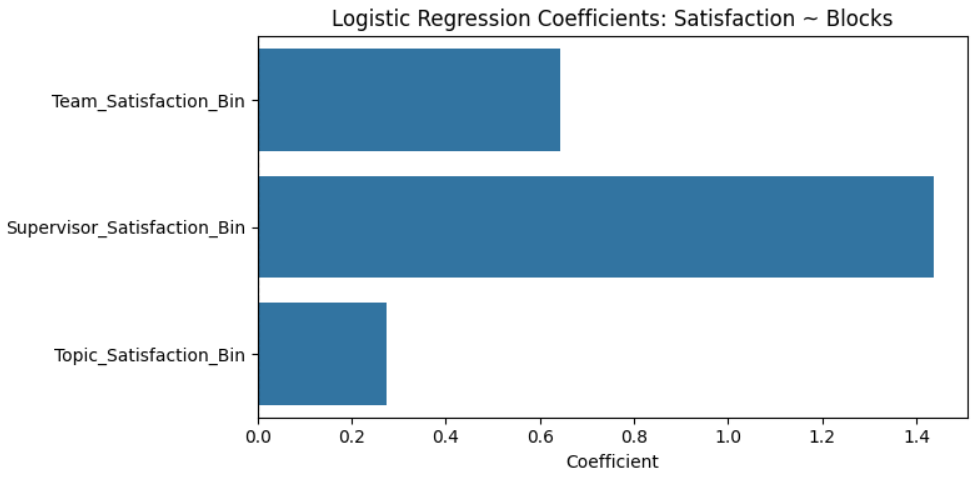


Figure 3.1.3.1 – Students Logistic Regression on Overall Satisfactions

### 3.1.4 Clustersation

The student clustering analysis aimed to identify patterns in how students' experiences and challenges during the coordination process align with their overall satisfaction levels. A set of key features was selected based on previous regression and correlation results, including variables related to search difficulty, duration, and perceived alignment with the team, supervisor, and topic.

Before clustering, the selected variables were standardized using StandardScaler to ensure uniformity of scale. Dimensionality reduction was applied using Principal Component Analysis (PCA), which allowed the complex multidimensional data to be visualized in two main components. These components represent abstract dimensions summarizing the primary variance across all selected indicators.

The following visualization illustrates the distribution of student respondents across four clusters, defined using the KMeans algorithm:

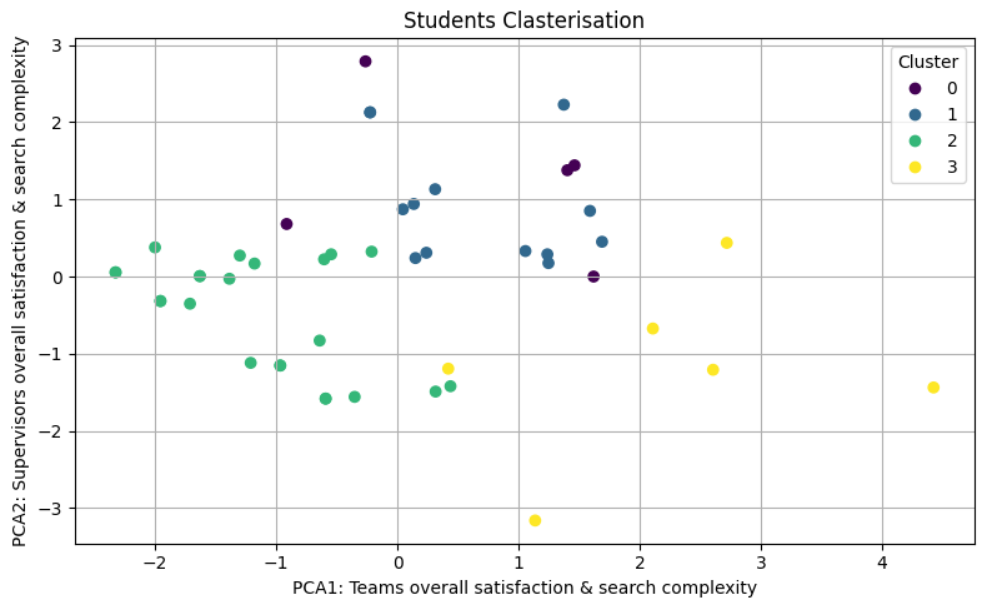


Figure 3.1.4.1 – Students Clasterisation

The X-axis (PCA1) is interpreted as an aggregate representation of team satisfaction and search complexity, while the Y-axis (PCA2) corresponds to supervisor satisfaction and associated difficulties. Each color indicates a different cluster, and a legend identifies cluster membership numerically.

The clusters demonstrate clear separation, suggesting that students’ experiences during coordination naturally segment into distinct profiles. To better understand these clusters, the following summary table presents the number of students in each cluster and their average overall satisfaction index:

|  |  |  |
| --- | --- | --- |
| Cluster | Number of Students | Average Satisfaction |
| 0 | 5 | 0.50 |
| 1 | 13 | 0.31 |
| 2 | 23 | 0.72 |
| 3 | 6 | 0.42 |

Cluster 2 stands out with the highest average satisfaction (0.72), indicating this group encountered fewer challenges and reported smoother coordination with supervisors and teams. In contrast, Cluster 1, with the lowest satisfaction (0.31), likely faced compounded difficulties throughout the process. Cluster 0 shows moderate satisfaction (0.50), while Cluster 3 reflects a mixed experience with an average of 0.42.

This segmentation allows for precise targeting of support and interventions. For example, students in Cluster 1 may benefit from improved access to supervisors or clearer communication during topic selection. Meanwhile, practices observed in Cluster 2 could inform best practices for guiding new students through the coordination process.

## 3.2. Supervisors

### 3.2.1 Correlation Analysis

To understand the key drivers of satisfaction from the supervisors’ perspective, a block-based correlation analysis was conducted. This method assessed how various operational parameters influenced satisfaction within three core domains: Team Satisfaction, Supervisor Satisfaction, and Topic Satisfaction. Additionally, a cross-block correlation analysis was performed to identify the overall impact of all variables on Overall Satisfaction.

The first set of visualizations illustrates the internal correlations within each block. As shown below, each horizontal bar represents the strength and direction of correlation between a parameter and its associated satisfaction indicator.

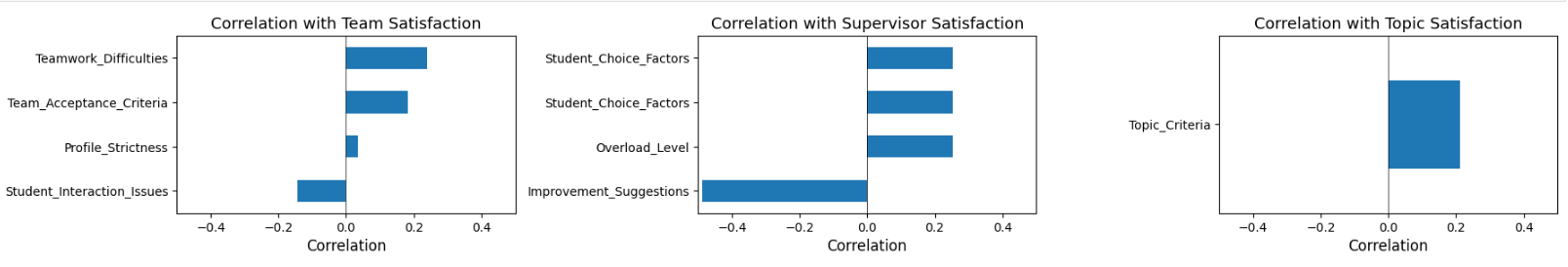


Figure 3.2.1.1 – Supervisors Correlation with blocks

In the Team Satisfaction block, Teamwork\_Difficulties and Team\_Acceptance\_Criteria both show moderate positive correlations with satisfaction. These results imply that well-defined team selection processes and fewer team-related issues are associated with higher supervisor contentment. Student\_Interaction\_Issues, by contrast, exhibits a slight negative correlation, indicating that relational challenges with students may reduce satisfaction.

In the Supervisor Satisfaction block, Student\_Choice\_Factors and Overload\_Level both demonstrate positive correlations, suggesting that supervisors feel more satisfied when chosen for meaningful reasons and when their workload is manageable. A notably negative correlation is observed with Improvement\_Suggestions, likely because dissatisfaction motivates constructive criticism.

The Topic Satisfaction block features Topic\_Criteria as the sole analyzed variable, with a small but positive correlation, reflecting that satisfaction rises when topics align with clearly defined academic criteria.

To assess the broader implications of all process variables, the analysis extended to their correlation with Overall Satisfaction.

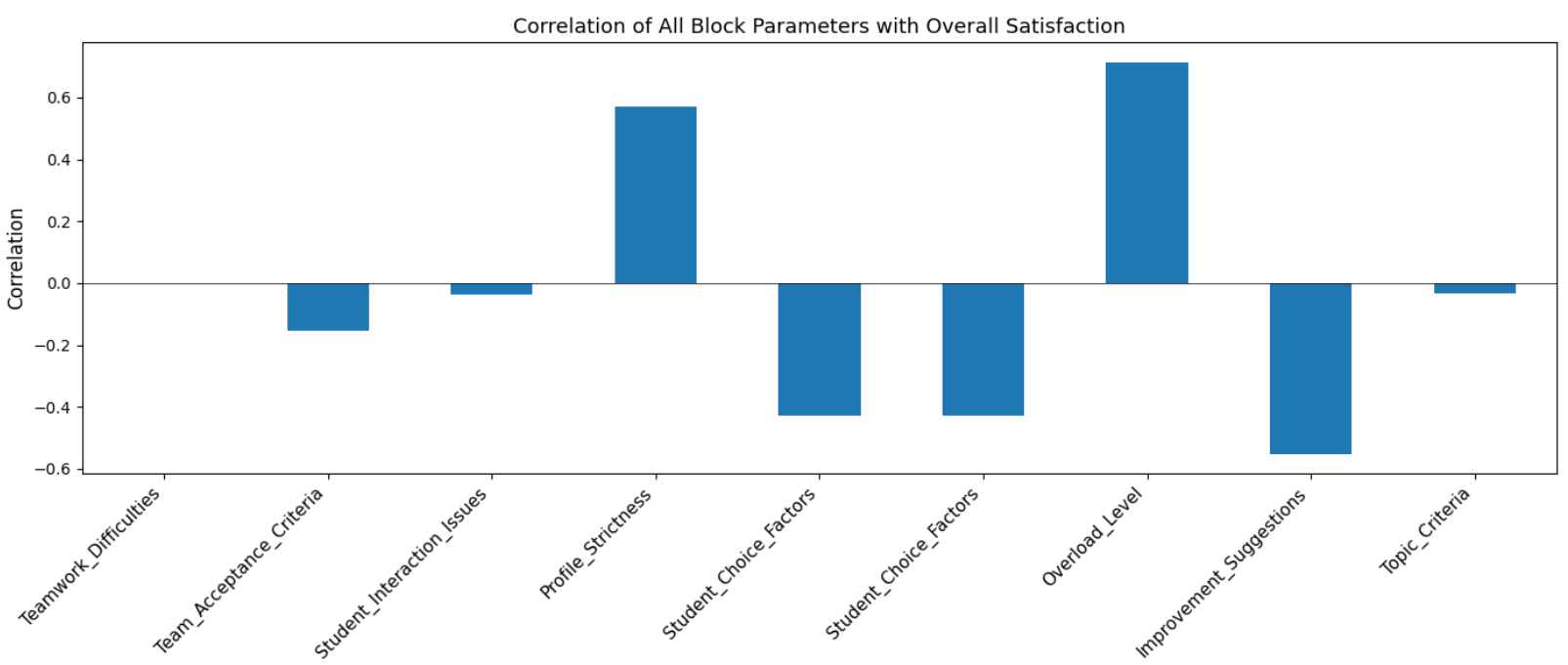


Figure 3.2.1.2 – Supervisors Correlation with Overall Satisfaction

This combined visualization highlights that Overload\_Level and Profile\_Strictness are the most positively correlated with Overall Satisfaction. These results emphasize that supervisors value both a balanced workload and the opportunity to supervise within their field of expertise. On the other end, parameters such as Improvement\_Suggestions and both instances of Student\_Choice\_Factors show negative correlations, underlining areas where alignment or transparency may need improvement.

Taken together, these graphs offer meaningful diagnostic value. They suggest that enhancing clarity in student selection, managing supervisory load, and supporting effective team integration may significantly elevate supervisor satisfaction across all project coordination stages.

The correlation analysis for supervisors establishes the groundwork for further modeling. The following section explores linear regression to quantify the predictive power of these parameters.

### 3.2.2 Linear Regression

To assess which factors most significantly affect satisfaction among supervisors, linear regression models were applied to each satisfaction block: Team Satisfaction, Supervisor Satisfaction, Topic Satisfaction, and Overall Satisfaction. Each model included logically related predictors, previously normalized and scored according to the importance of their impact on decision-making and coordination.

The graph below presents the regression coefficients for predictors of Supervisor Satisfaction. This block focused on factors such as supervisor selection difficulties, student choice factors, workload (overload level), and suggestions for improvement. The analysis shows that difficulties during supervisor selection, perceived student motivations, and overload levels had a strong positive contribution to satisfaction, while the number of improvement suggestions had a mild negative effect.

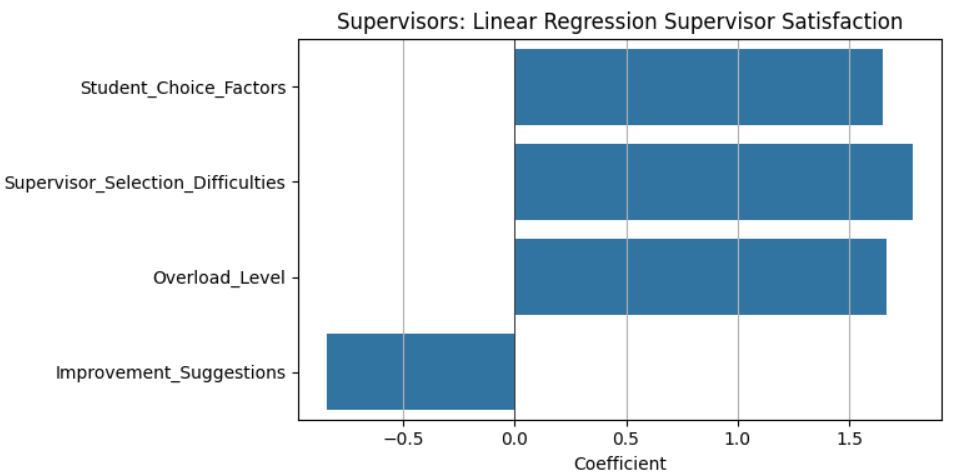


Figure 3.2.2.1 – Supervisors Linear Regression on Supervisor Satisfactions

In the Topic Satisfaction block, predictors included the criteria used to select topics, preferences in topic initiation (student vs. supervisor), and how strictly the supervisor adheres to their research profile. All predictors were positively associated with satisfaction. In particular, the desire for compromise in topic selection had the strongest positive effect.

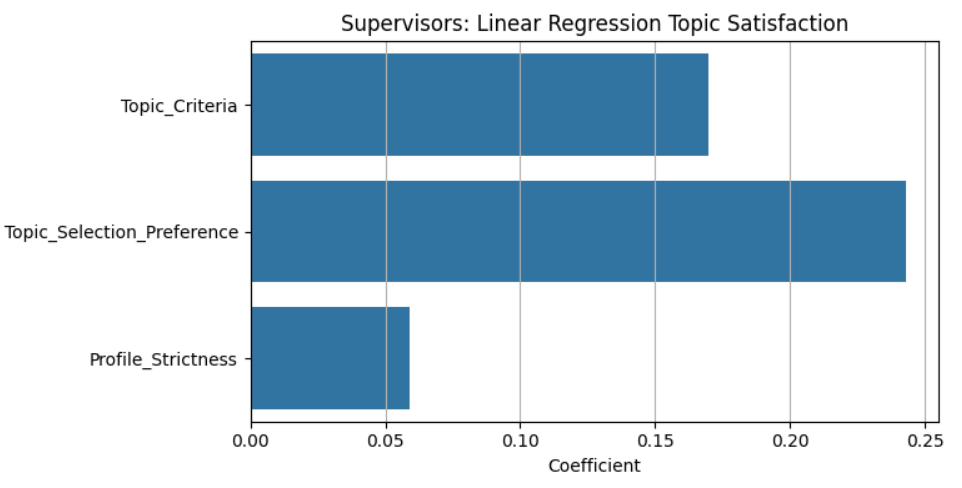


Figure 3.2.2.2 – Supervisors Linear Regression on Topic Satisfactions

The Team Satisfaction model included teamwork difficulties, criteria for accepting teams, and issues in interaction with students. The results indicate that teamwork difficulties had the strongest positive correlation with satisfaction, followed by acceptance criteria. Interestingly, the presence of student interaction issues appeared to slightly decrease satisfaction.

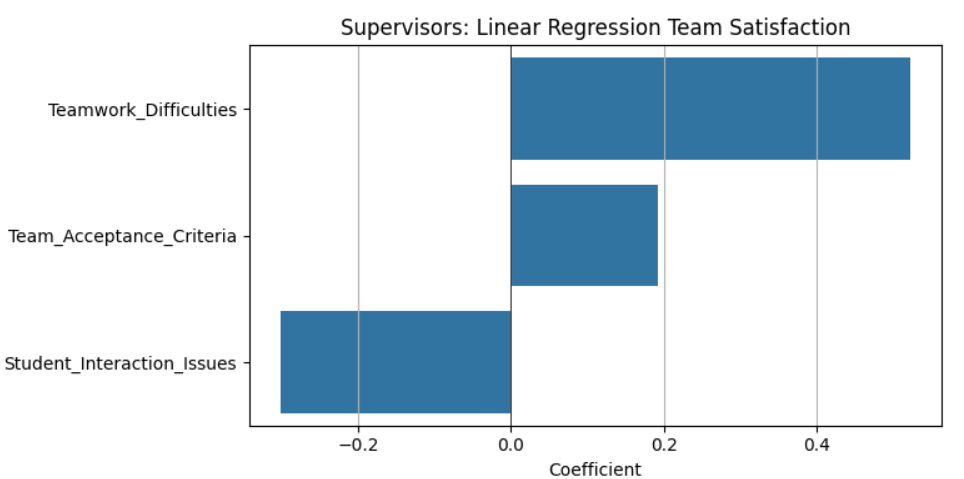


Figure 3.2.2.3 – Supervisors Linear Regression on Team Satisfactions

Finally, the model for Overall Satisfaction regressed general satisfaction on the three core satisfaction indicators from the previous blocks. Topic satisfaction showed the strongest positive contribution to the overall outcome. Surprisingly, supervisor satisfaction was slightly negatively associated, suggesting that factors outside of this block may dominate the perception of the overall system.

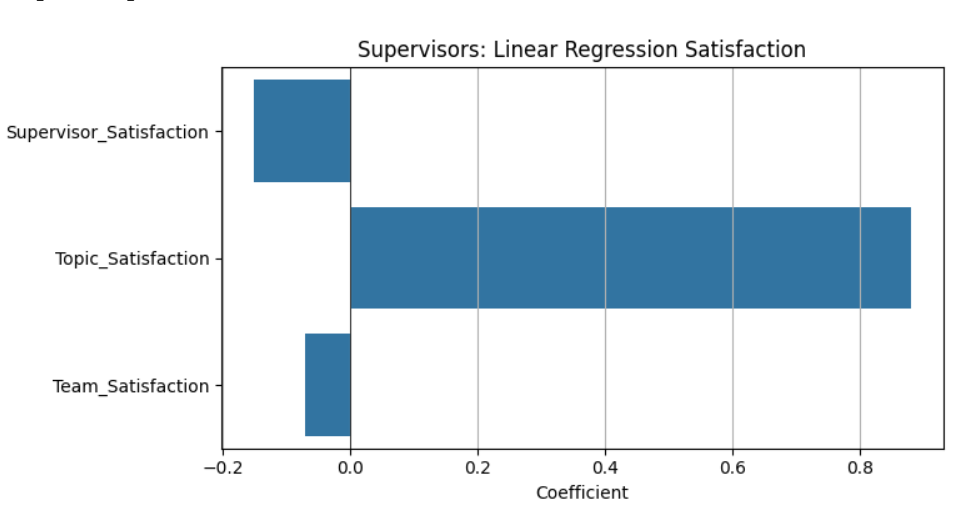


Figure 3.2.2.4 – Supervisors Linear Regression on Overall Satisfactions

These linear models provide a clear picture of how each factor relates to different aspects of supervisor satisfaction. The positive coefficients in most blocks indicate that improving these conditions could enhance satisfaction, while negative contributions (e.g., student interaction issues, overload) highlight areas for targeted intervention.

### 3.2.3 Logistic Regression

To complement the linear models, logistic regression was applied to evaluate how satisfaction in specific blocks influences the probability of overall satisfaction among supervisors. In this model, the outcome variable was binarized — representing whether the supervisor expressed general satisfaction with the system (1) or not (0). Predictors included the binary representations of satisfaction in the Team, Supervisor, and Topic blocks.

The graph below presents the logistic regression coefficients for these predictors:

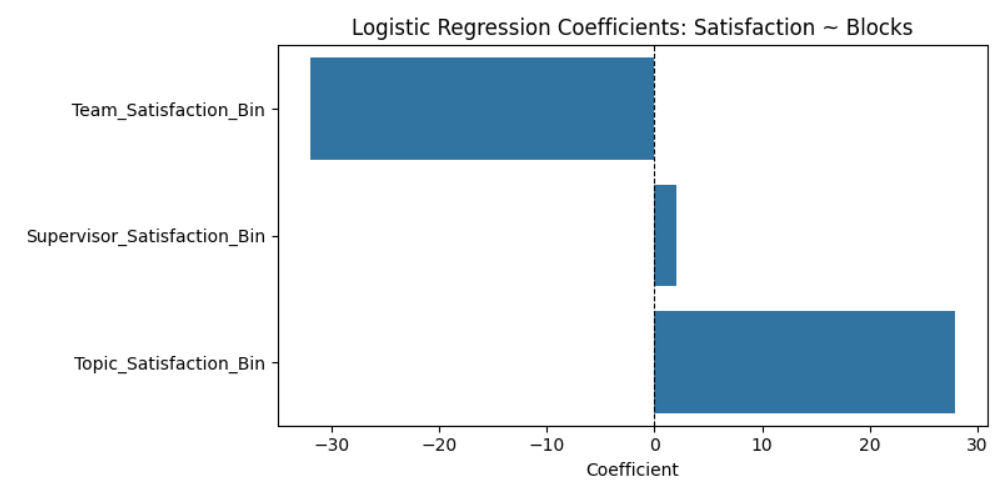


Figure 3.2.3.1 – Supervisors Logistic Regression on Overall Satisfactions

The coefficients reflect the log-odds of overall satisfaction given a change in each block-specific satisfaction indicator. Among the predictors, Topic Satisfaction demonstrated the strongest positive effect, suggesting that when supervisors are satisfied with the topic selection process, they are significantly more likely to express overall satisfaction. Team Satisfaction showed a surprisingly strong negative coefficient, potentially indicating a statistical artifact or a complex inverse relationship, possibly due to multicollinearity or limited variation within team-related satisfaction scores. Supervisor Satisfaction contributed positively but with less magnitude.

The magnitude and direction of these coefficients offer insight into the conditions under which supervisors perceive the coordination system as effective. High dependence on topic satisfaction highlights the importance of aligning project themes with supervisor expectations and preferences. These results reinforce the findings from linear regression and further prioritize areas for improvement.

### 3.2.4 Clusterisation

To capture diversity in supervisor experiences and satisfaction profiles, a clustering analysis was conducted using KMeans. The clustering process began with the transformation of supervisor-related variables into normalized numerical scores reflecting key behavioral and perceptual dimensions. Principal Component Analysis (PCA) was then applied to reduce the feature space into two components for effective visualization.

The scatter plot below illustrates the resulting clusters in a two-dimensional PCA space, where each point represents a supervisor. The x-axis (PCA1) captures the combined dimension of Satisfaction & Student Fit, while the y-axis (PCA2) reflects Search Effort & Team Experience. Supervisors are color-coded by cluster membership.

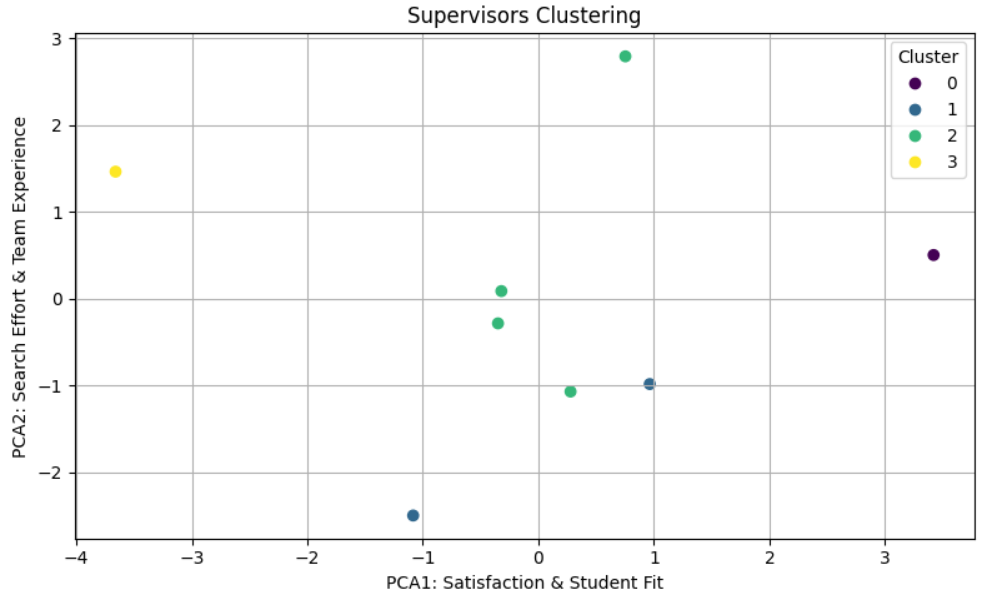


Figure 3.2.4.1 – Supervisors Clusterisation

|  |  |  |
| --- | --- | --- |
| Cluster | Number of Students | Average Satisfaction |
| 0 | 1 | 0.50 |
| 1 | 2 | 0.75 |
| 2 | 4 | 0.25 |
| 3 | 1 | 0.00 |

The clustering analysis revealed four distinct groups of supervisors, each representing a unique combination of satisfaction levels and supervision experiences:

* Cluster 0 comprises a single supervisor with a moderate satisfaction score of 0.50, reflecting an average experience in terms of workload, team coordination, and procedural alignment.
* Cluster 1 includes two supervisors with a comparatively high satisfaction level (0.75). These individuals appear to have encountered fewer difficulties during the supervision process, likely benefiting from strong student alignment and efficient collaboration.
* Cluster 2 is the largest group, containing four supervisors with a notably low average satisfaction of 0.25. This suggests recurring issues across this cluster, such as excessive workload, challenges in topic or team coordination, or mismatched student expectations.
* Cluster 3 consists of one supervisor who reported zero satisfaction, indicating a highly problematic experience, potentially characterized by systemic barriers or persistent dissatisfaction across multiple stages of the process.

The visual distribution across the two principal components highlights how satisfaction is shaped by both procedural complexity and the perceived compatibility between supervisors and students. These clusters provide actionable insights into the segmentation of supervisor experiences, enabling targeted improvements for those groups experiencing lower satisfaction.

## 3.3. Relevance

This project addresses a critical dimension of academic organization—the process of coordinating diploma projects between students and supervisors, which serves as a cornerstone of higher education quality and efficiency. The effectiveness of this process directly influences not only the academic success and motivation of students, but also the workload management and professional fulfillment of faculty members. By systematically analyzing satisfaction levels, search difficulties, and interaction patterns, this research provides an empirical foundation for enhancing the structure and outcomes of the diploma project system.

The relevance of the work extends beyond individual experiences. It identifies systemic inefficiencies, such as mismatches in topic selection processes, communication breakdowns during team formation, and inconsistent supervisor engagement. Through statistical modeling and visual diagnostics, the project translates subjective feedback into measurable indicators, revealing which aspects most significantly affect satisfaction. This allows educational institutions to implement targeted improvements, ensuring better alignment between student expectations and faculty capabilities. As universities increasingly seek to personalize academic trajectories and strengthen mentorship, this work offers a data-driven blueprint for optimizing academic project coordination, reinforcing both academic integrity and institutional performance.

# 4 ECONOMIC SECTION

## 4.1. Cost Analysis of Development and Maintenance

Recreation and maintaining of a web-platform for finding a team and a supervisor for diploma thesis requires full-stack web-development cycle, which should be meticulously analyzed for associated costs. These include both one-time development costs and ongoing operational expenses, considering 5-month development and annual maintenance.

### 4.1.1. Hardware and Equipment

Taking the average costs for laptops, that can ensure comfortable development without any hitches, and adding average monthly cost for unlimited Internet, we have following calculations:

|  |  |  |  |
| --- | --- | --- | --- |
| **Equipment** | **Units** | **Unit Cost (USD)** | **Total Cost (USD)** |
| Laptops | 4 | 1 000 | 4 000 |
| Internet | - | - | 100 |
| **Total** | **-** | **-** | **4 100** |

### 4.1.2. Human Resource Costs

The estimated number of hours required for different roles in the project lifecycle, considering the data from “Structure and distribution of wages of employees in the Republic of Kazakhstan (by the end of 2024)” publication [32], as for professional specialists, whose average monthly wage is 402 226 tenge, and work schedule is 5/2, 8 hours per day, hourly rate will be equal:

Average monthly wage / (Work hours per day \* Work days in a month) = Hourly Rate

402 226 / (8 \* 21) = 2 394,2

2 394,2 / 500 = 4,79 (USD)

Table 4.1.2. - Human Resource Costs

|  |  |  |  |
| --- | --- | --- | --- |
| **Specialist** | **Hours** | **Hourly Rate (USD)** | **Total Cost (USD)** |
| Frontend Developer | 300 | 4,79 | 1 437 |
| Backend Developer | 300 | 1 437 |
| UX/UI Designer | 120 | 574,8 |
| Data Analyst | 150 | 718,5 |
| Project Manager | 100 | 479 |
| **Total** | **-** | **-** | **4 646,3** |

This way, combined work of all developers recruited will cost 4 646,3 USD.

### 4.1.3. Software Licenses and Subscriptions

For better and optimized performance in development and deploying

Table 4.1.3. - Software Licences and Subscriptions

|  |  |
| --- | --- |
| **Item** | **Cost (USD/Year)** |
| DataGrip | 229 |
| Figma Professional | 144 |
| Docker for Teams | 288 |
| **Total** | **661** |

### 4.1.4. Maintenance and Support

For web-platform maintenance and support were taken approximate costs, all calculations are as follow in Table 4.1.4.:

Table 4.1.4. Estimated Costs of Maintenance and Support

|  |  |
| --- | --- |
| **Item** | **Estimated Cost (USD)** |
| Bug Fixes & Minor Updates | 3,000 |
| Server Maintenance & Scaling | 1,000 |
| Customer Support | 1,200 |
| Performance Monitoring Tools | 300 |
| **Total (Annual)** | **5,500** |

Total cost of such platform forfinding a team and a supervisor for diploma thesis is 14 907,3 USD.

## 4.2. Revenue Potential

Although this platform is designed primarily for academic use, it also offers revenue-generation opportunities if extended or scaled beyond initial institutional deployment. Below are possible monetization strategies:

**1. Freemium Model:**

- Basic platform features remain free for all users (e.g., messaging, profile creation, basic search).

- Premium features such as advanced filtering, team analytics, progress tracking, and AI-based recommendations are available via paid subscription.

**2. Institutional Licensing:**

- Offer the platform as a Software-as-a-Service (SaaS) product to universities.

- Annual licensing fees per institution, scaled based on the number of active students/faculty.

**3. Sponsored Projects and Partnerships:**

- Allow third-party research organizations and academic partners to list sponsored diploma topics for a fee.

- Enable verified companies to post project-based internships or thesis topics, generating partnership revenue.

**4. Certification and Value-Added Services:**

- Provide premium certificates of participation, supervisor evaluations, or progress portfolios for a fee.

- Offer value-added features like storage upgrades, team video conferencing, or integration with professional platforms (e.g., LinkedIn).

**5. Advertising (Optional):**

- Carefully curated, non-intrusive academic or career-related ads could be displayed in the free version to generate ad revenue.

These revenue streams ensure long-term sustainability and scalability of the platform while keeping core functionality accessible to all users.

Developing and maintaining a web platform for finding a team and supervisor for diploma thesis projects requires a comprehensive analysis of associated costs. These include both one-time development costs and ongoing operational expenses.

## 4.3. SWOT Analysis

### 4.3.1 Strengths​​​​​​​​

Starting from non-technological advantages, one of the primary strengths of the platform it is in focusing on centralizing thesis-related interactions in one digital environment, which is currently being an urgent problem for the academic systems in Kazakhstani universities.

Another strong side of the platform lies on using of open-source technologies. Vite (Vue.js) and Figma for frontend, Django for backend, WebSockets for real-time interactions, Docker as the tool for deployment - all these technologies ensure simplicity and speed, while providing a high level of functionality and scalability.

Platform’s modular architecture allows for varied future expansions, adaptations for different needs and maintenance.

Additionally, real-time messaging and notification systems enhance communication efficiency among users, while secure, role-based access improves the overall safety of the data shared within the platform.

### ​​4.3.2 Weaknesses​​​​​​​​

While having significant strengths, platform also has several disadvantages. Initial development and deployment costs can be significant, especially considering the complexity of features like integration with third-party services and secure authentication.

The use of multiple technologies and frameworks increases the complexity of the system, potentially leading to technical debt if not properly maintained. Moreover, for non-technical users, the interface might present a learning curve if the UX design is not adequately optimized.

### ​​4.3.3 Opportunities​​​​​​​

There are numerous opportunities for the platform’s growth and development in the academic environment. Integration with existing university systems as WSP [31] (locally in Kazakh-British Technical University or spreading the platform to all Kazakhstani universities) can streamline Students, Professors and Dean’s Offices workflows.

The implementation of Artificial Intelligence and Machine Learning algorithms can further enhance the matching process between Students, Teams, and Supervisors, and help in choosing relevant and interesting topics for thesis.

Additionally, the platform can be scaled to support broader academic initiatives, including internships, group coursework, and research mentoring. With proper localization and internationalization, the platform could be adapted for use across different countries and institutions.

### ​​4.3.4 Threats​​​​​​​

The primary threats include the emergence of competing tools—either specialized academic platforms or general-purpose professional networking sites that incorporate similar features. Adhering to strict data protection regulations such as GDPR can pose legal and operational challenges.

The reliance on third-party services for hosting and real-time messaging can result in vendor lock-in and cost fluctuations. Finally, as with any digital platform, there are cybersecurity risks, including data breaches, unauthorized access, and potential downtime, all of which must be proactively managed.

## ​4.4 Competitive Analysis

While developing the LMS platform for diploma thesis collaboration, it is important to understand the competitive landscape. Notably, the platform does not face direct competition from existing academic systems, as few offer dedicated solutions for team formation and supervisor discovery. However, several indirect competitors provide overlapping functionalities, particularly in communication and coordination.

Table 4.4. - Competitors Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Platform** | **Type** | **Purpose** | **Strengths** | **Weaknesses** |
| Microsoft Teams | Indirect competitor | Team collaboration in education | Integration with Office 365, video conferencing | No features for academic team formation or supervisor search |
| Telegram | Indirect competitor | Messaging & file sharing | Fast, lightweight, supports large groups | No academic context or structured collaboration features |
| WhatsApp | Indirect competitor | Messaging | Ubiquitous, mobile-friendly, encrypted chats | Poor scalability for large teams, lacks academic features |
| University LMS (wsp.kbtu.kz) | Partial overlap | Course and assignment management | Integrated with university records | No features for team matching or real-time collaboration |

Our platform fills a unique niche by combining features of messaging apps, collaborative tools, and academic databases. It is specifically tailored for the academic environment and supports structured team building, supervisor discovery, and project tracking - functionality not offered by current solutions.

# CONCLUSION

The development and research of the web platform "Diplomatch" have successfully achieved the initial objectives: to streamline the process of forming student teams and finding academic supervisors for diploma theses. Through the use of modern technologies such as Vue.js, Django, Docker, and PostgreSQL, we have created a scalable, intuitive, and user-centric solution tailored to the realities of academic collaboration.

From the outset, this project was grounded in a clear understanding of existing challenges. By analyzing the current academic environment — both globally and within Kazakhstan — we identified a clear gap in the availability of structured, digital tools to support thesis preparation. Unlike general-purpose platforms such as messaging apps or LMS portals, Diplomatch was designed from the ground up to meet the specific needs of students navigating the diploma project process.

The implementation of cutting-edge web technologies enabled us to deliver a robust and flexible platform. Vue.js provided a responsive and modular frontend, while Django ensured the development of secure and reliable backend services. With Docker simplifying deployment and Figma accelerating the design process, the team was able to iterate quickly and respond to user feedback in real time. The inclusion of user profile systems with skillsets, real-time messaging, and intelligent matching features exemplifies our platform’s thoughtful, data-driven design.

Diplomatch also reflects behavioral and academic insights. By making student skills and interests visible, the platform fosters more effective collaboration and targeted team formation. The integration of notifications, progress tracking, and supervisor listings further reduces ambiguity and enhances productivity across student-supervisor relationships.

Economically, the project demonstrates potential for financial sustainability. Licensing the platform to universities, offering premium academic tools, and forming partnerships with educational institutions represent realistic revenue streams. A freemium model ensures accessibility for all students while providing pathways to long-term scalability.

From a safety and ethical standpoint, the project incorporates essential privacy measures to safeguard user data and support responsible digital behavior within academic communities.

Overall, Diplomatch has met its functional goals while remaining adaptable to the evolving needs of its audience. Nonetheless, future improvements could include deeper analytics for team performance, integration with university LMS systems, and multilingual support. We also see potential in expanding to internships, research projects, and academic mentoring.

Based on our experience, we recommend continued user-driven development, university partnerships to increase adoption, and long-term investments into platform modularity and personalization.

In conclusion, Diplomatch is a thoughtfully designed platform that solves a real problem in higher education. It not only meets its technical and functional targets but also offers a vision of how digital tools can enhance collaboration, mentorship, and academic success. This project serves as a strong foundation for future innovations in the field of educational technology.