FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

`

AI-Powered Career Opportunity Distribution System

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

1. Project Specification:

The goal of this project is to create the architecture (Features AI agent behavior) system that connects corporate opportunities with students through Discord using advanced AI-driven conversational intelligence. This system will enhance the EXPERTS.AI platform by providing intelligent opportunity matching and seamless distribution via Discord.

Problem Domain

- Companies post opportunities (jobs, internships, thesis topics) but lack effective engagement with students.
- Current methods (faculty websites, newsletters, and Discord) have low engagement rates.
- Manual opportunity matching is inefficient.
- Students struggle to find relevant options tailored to their skills and preferences.
- A lack of automation in opportunity distribution leads to lower engagement and slower matching.

Agenda

- Enhance the EXPERTS.AI platform with AI capabilities for better matching and real-time engagement.
- Automate the opportunity distribution process using an AI-powered Discord bot.
- Improve student engagement through personalized recommendations and real-time interactions.
- The solution should improve engagement, personalization, and automation using AI.

Tools and Technologies:

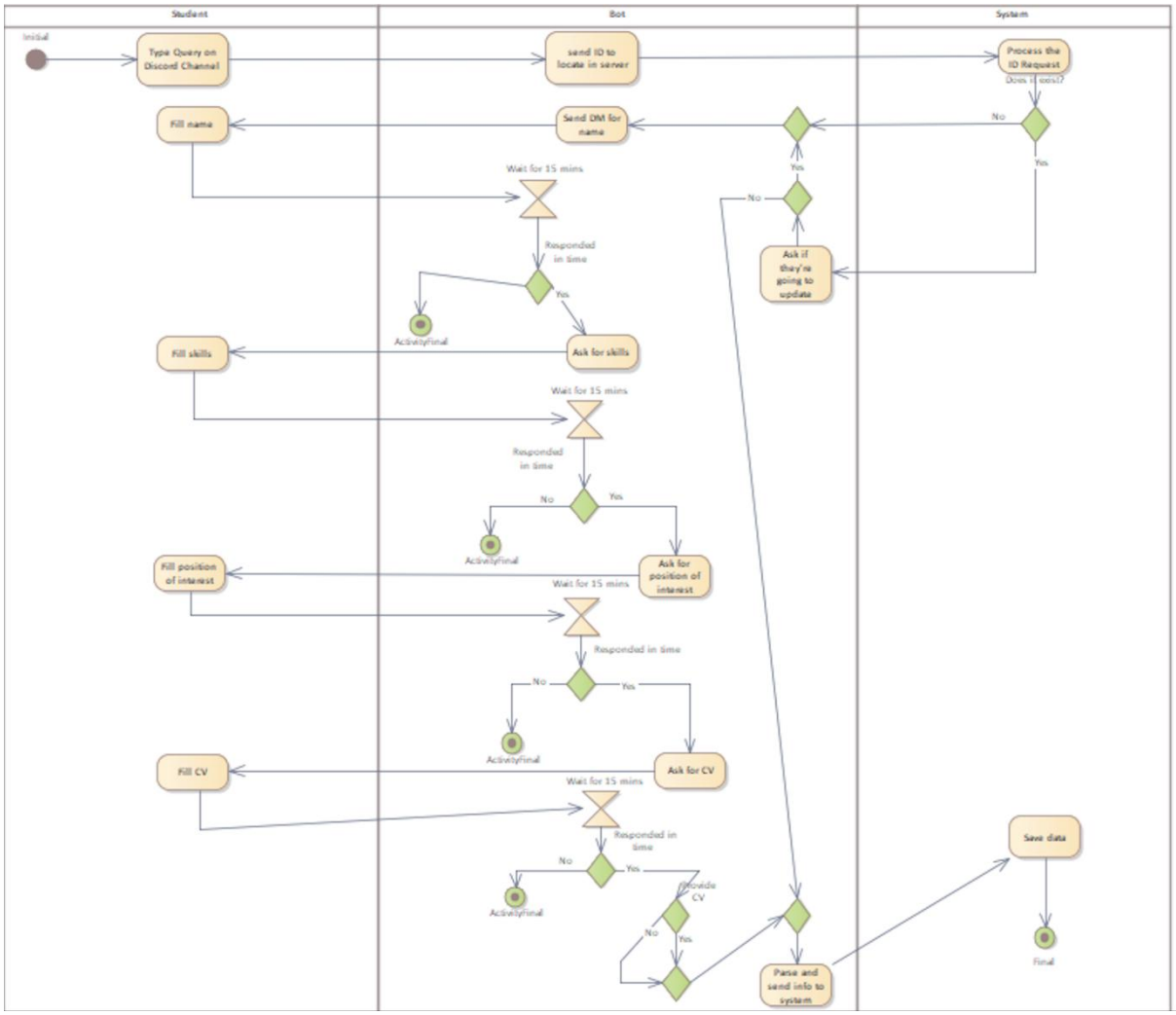
In our project, we utilize a robust tech stack that includes Java with Spring and Spring Boot for backend development, ensuring scalability and efficiency. For the frontend, we leverage Angular 16/17, providing a dynamic and responsive user interface. Our database management is powered by PostgreSQL, offering reliable data storage and retrieval. Additionally, we integrate Elasticsearch for efficient full-text search capabilities, enhancing data accessibility. To optimize recommendations, we

employ Recombed, a powerful recommendation engine that personalizes user experiences based on advanced algorithms. This combination of technologies allows us to build a high-performance and intelligent system.

Project Analysis

AI-Powered Career Opportunity Distribution System

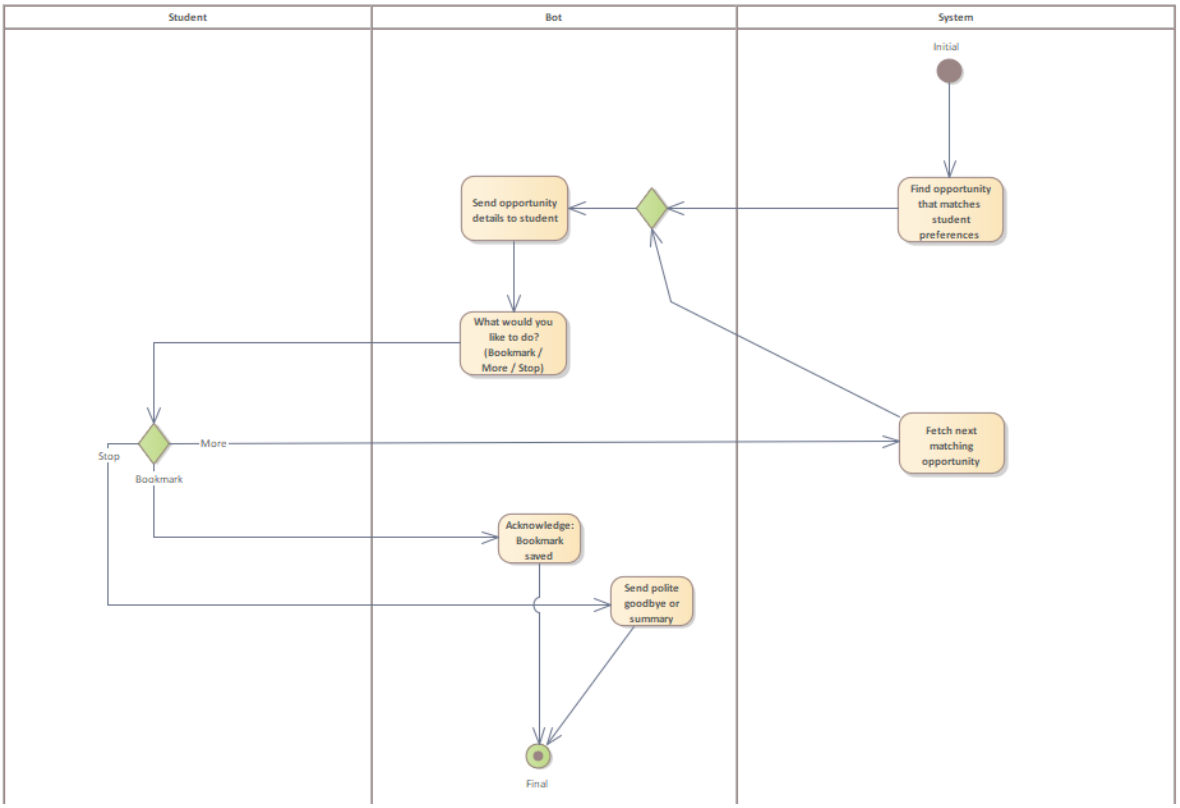
Student Information Retrieval This process shows how a Discord bot systematically collects a student's profile information through timed direct messages and stores it in a database for future automated interactions.



The student profile information retrieval process begins when a student initiates an interaction by submitting a query prompt on a designated Discord channel. This prompt is received by a bot, which acts as the primary agent responsible for collecting the student's data. Upon receiving the query, the bot checks whether the student's profile already exists in the database. If the profile is found, the process is terminated, as the required information is assumed to be available. If no existing profile is found—indicating it is the student's first interaction—the bot immediately initiates a direct message (DM) conversation to collect the necessary details. This includes sequentially requesting the student's name, skills, position of interest, and CV, with a five-minute timeout set for each step. If the student fails to respond within the allotted time for any request, the process is terminated. However, for the final step involving the CV, whether the student uploads a CV or opts to proceed without one, the bot considers the collected information sufficient and does not terminate the process. It proceeds to parse the available data and sends it to the backend system. The system then saves the structured profile information in the database. This streamlined and systematic process ensures that sufficient data is reliably collected and stored, enabling the bot to generate appropriate responses in subsequent interactions.

Bot response:

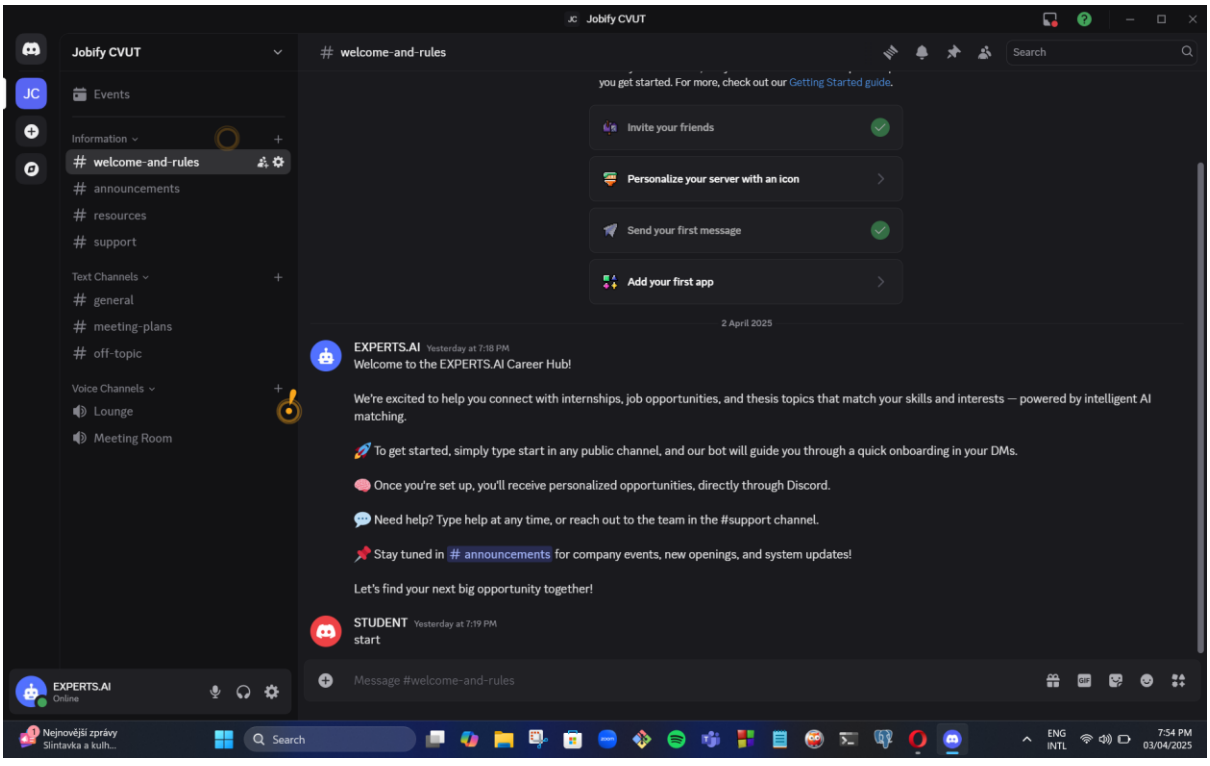
This UML activity diagram illustrates the interaction process between a student, a Discord-based bot, and the backend system, specifically focusing on the stage after a job opportunity is found that matches the student's preferences. The diagram is divided into three swim lanes: Student, Bot, and System, each representing a distinct actor in the flow.



FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

Stage 1: Public Initiation

- **Action**: Student types start in a public Discord channel.
- **Bot Response:** Sends a private message (DM) to the student and provides instructions.
- **Purpose:** Triggers a private session while maintaining a clean public interface.
- **Proof:** public initiation with start and bot redirection to private chat.



Private Onboarding

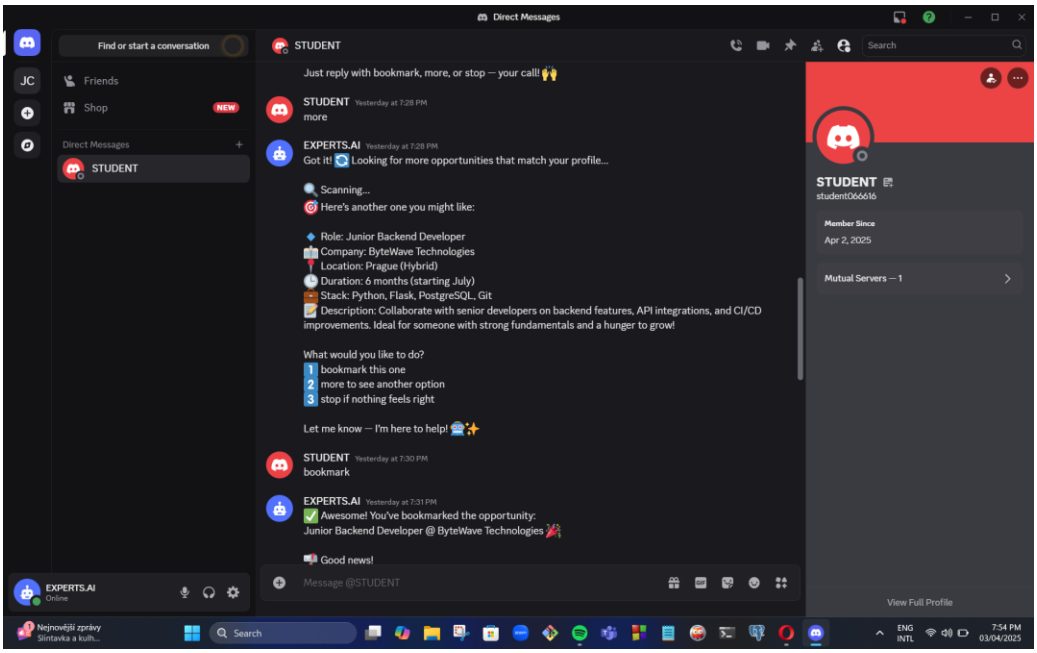
- **Action:** Bot opens a private DM and begins collecting data:
    - Name
    - Skills
    - Career interests
    - Resume upload or link
- **Purpose**: Gather key data to personalize job recommendations.
- **Proof:** bot asking structured, step-by-step onboarding questions.

Stage 3: Student Profile Submission

- **Action**: Students provide their name, skills, interests, and resume.
- **Bot Response:** Acknowledges receipt and begins scanning for relevant opportunities.
- **Proof**: both student response and bot processing message.

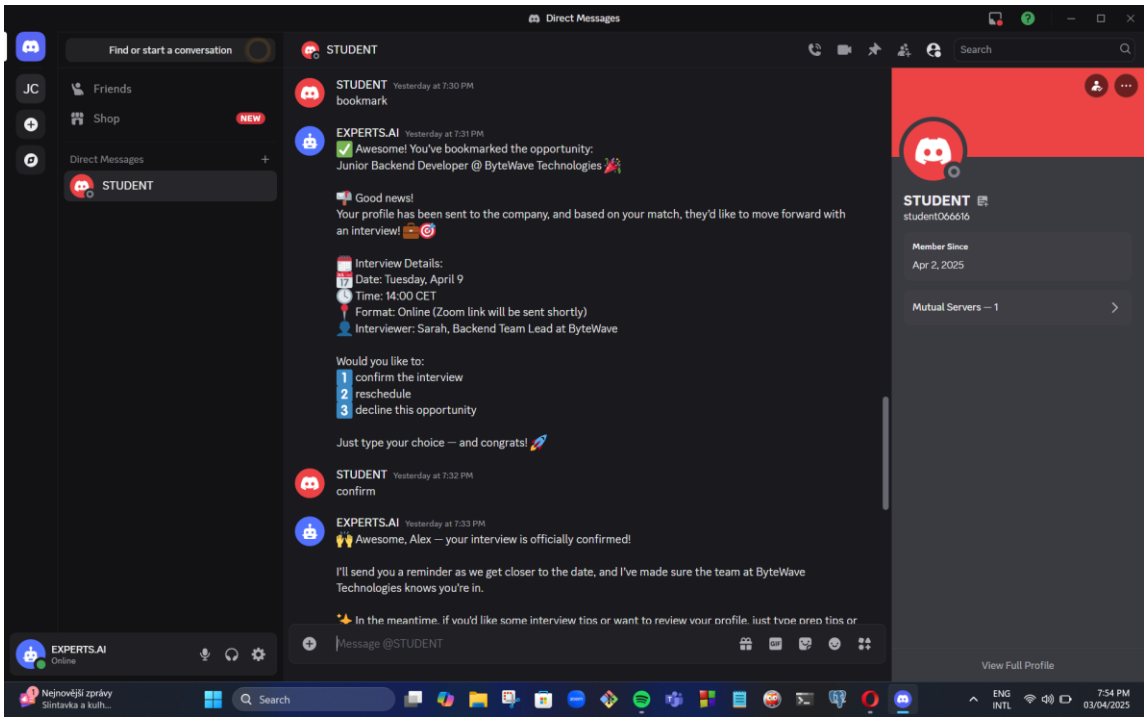
Stage 4: First Opportunity Match

- **Action**: Bot sends the first personalized opportunity to the student.
- **Details Include:**
    - Role title
    - Company
    - Location
    - Duration
    - Tech stack
    - Description
- **User Choices**: Bookmark, see more, or decline.
- **Proof:** full opportunity card and response options.



Stage 5: Additional Recommendation

- **Action**: Student chooses “more” to receive another opportunity.
- **Bot Response**: Sends another matching opportunity with new details.
- **Proof**: next opportunity based on student’s profile and preferences.

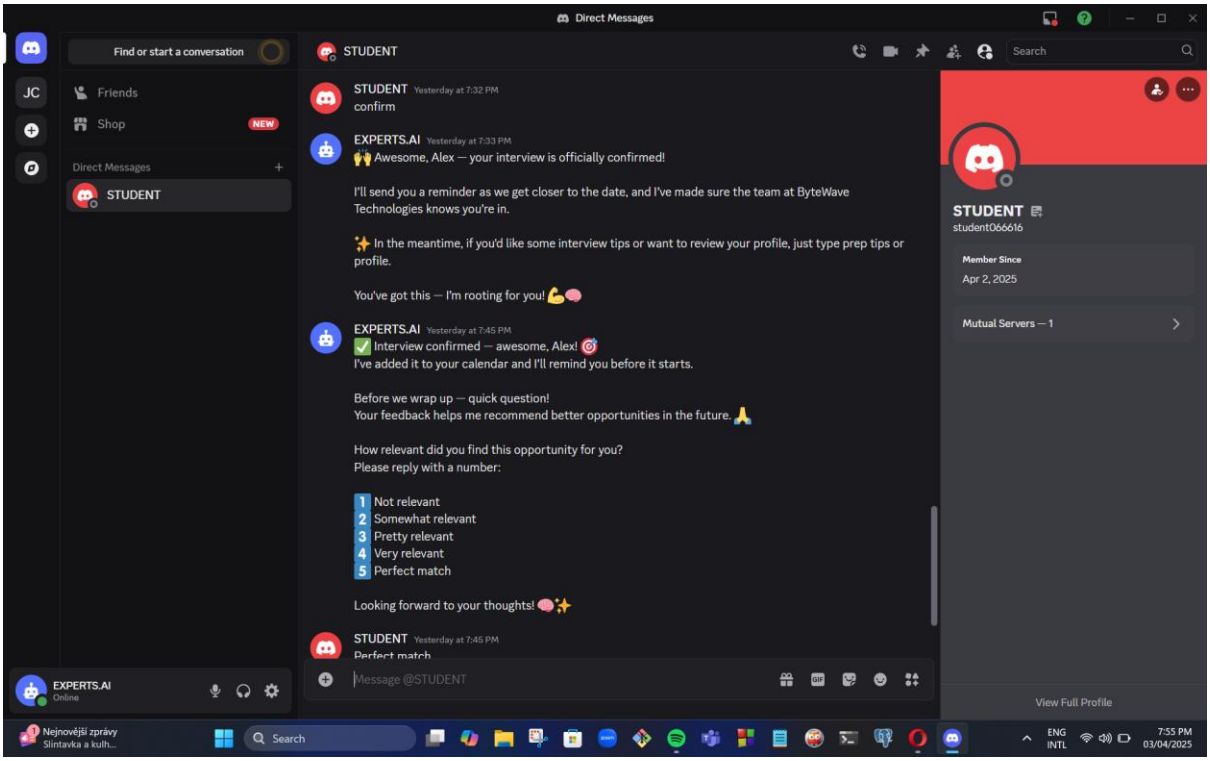


Stage 6: Bookmark and Interview Offer

- **Action**: Student bookmarks an opportunity.
- **Bot Response:**

- Confirms bookmark
- Informs the student that their profile was sent to the company
- Sends interview invitation with full details

- **Proof:** clear progression from bookmark to confirmed interview offer.

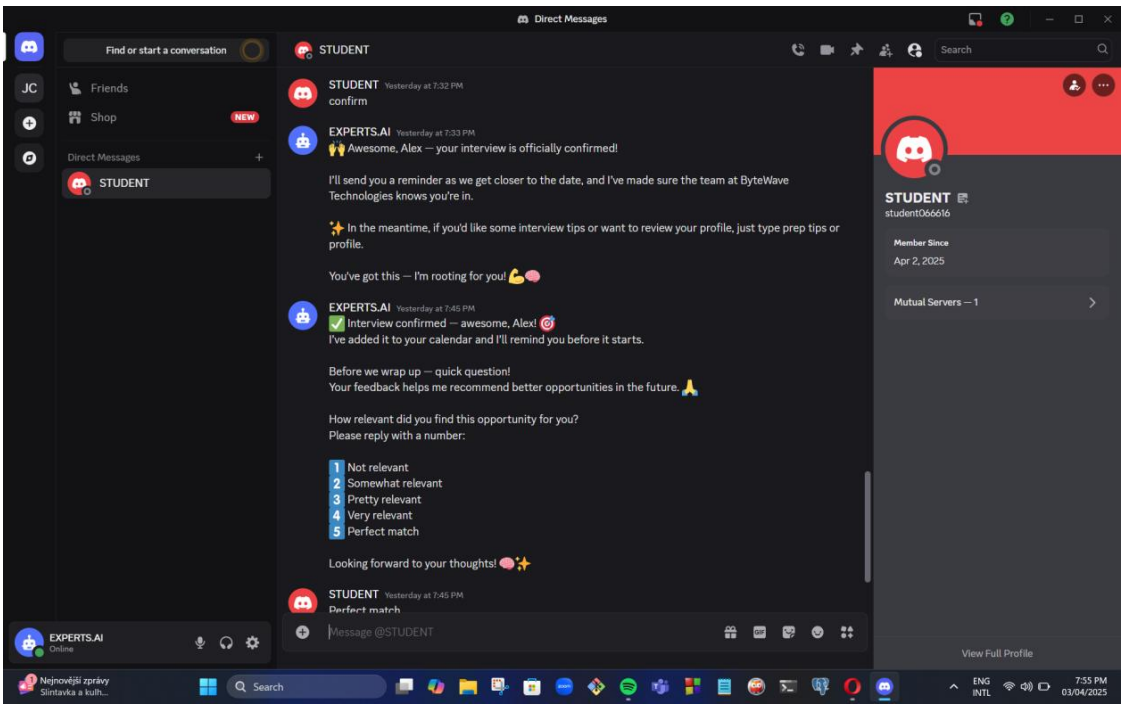


Stage 7: Interview Confirmation

Screenshot Reference: Screenshot (5) and (6)

- **Action**: Student confirms the interview.

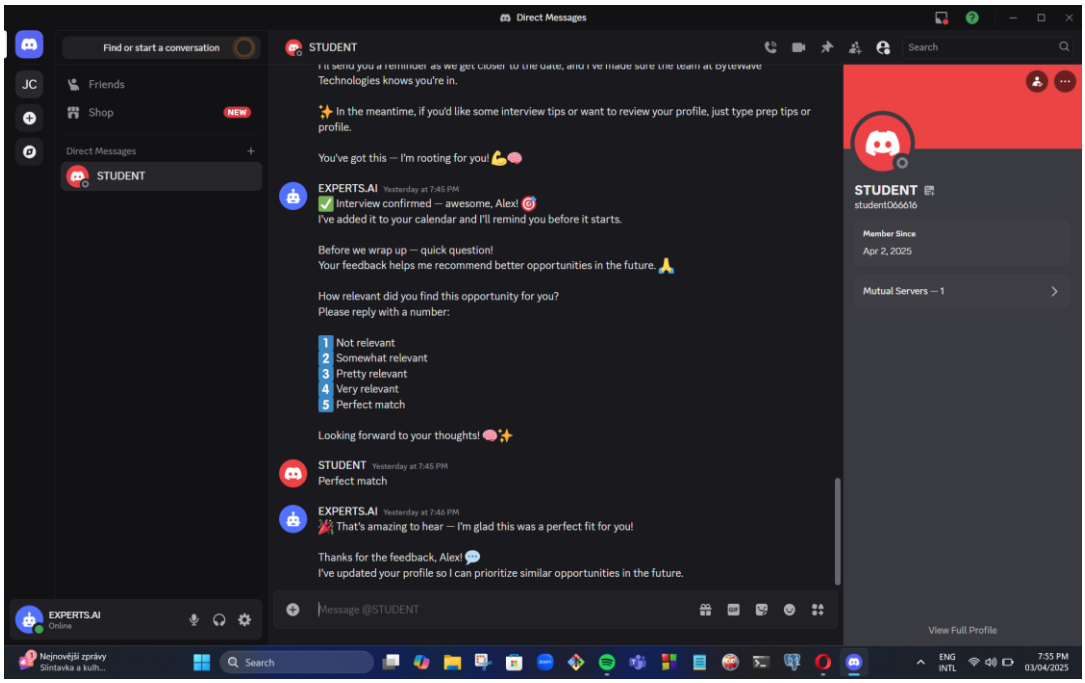
- **Bot Response:**
    - Acknowledges confirmation
    - Schedules reminders
    - Offers additional help (e.g., prep tips)
- **Proof**: confirmation process and supportive messaging from the bot.



Stage 8: Feedback Collection

- **Action**: Bot asks the student for feedback on the opportunity.
- **Student Response**: "Perfect match"

- **Bot Response**: Thanks the student, logs the feedback, and updates profile preferences.
- **Proof**: feedback prompt and final acknowledgment.



2.1 Opportunity Matching - AS IS

This section describes the current process of how students discover and apply for opportunities.

Current Process (AS-IS)

1. Companies post opportunities on the EXPERTS.AI platform and faculty websites.
2. Students manually browse opportunities via:
    - Faculty websites
    - Newsletters
    - Discord posts
3. Students must actively search for relevant opportunities.
4. Many relevant opportunities go unnoticed due to lack of personalized recommendations.
5. Faculty members and admins manually manage opportunity postings.

2.2 Opportunity Matching - TO BE

This section describes the proposed AI-driven process.

Optimized Process (TO-BE)

1. Companies post opportunities on EXPERTS.AI.
2. AI analyzes and categorizes opportunities.
3. AI matches opportunities with student profiles.
4. AI automatically interacts with students via Discord.
5. AI parses student CV.
6. AI sends personalized opportunity recommendations based on CV and interaction.
7. Student automatically receives when they have their interview
8. AI learns from student responses to improve recommendations.
9. The system ensures seamless integration between EXPERTS.AI, Discord, and faculty websites.

Key Benefits

- **Automated AI-driven matching** → Increased efficiency.
- **Personalized recommendations** → Higher engagement.
- **Real-time AI interactions** → Faster response times.
- **Reduced administrative workload** → Faculty and admins focus on strategic tasks.

3. Market analysis

3.1 Introduction

This report analyzes existing solutions that connect students with corporate opportunities through AI and digital platforms. The goal is to identify key competitors, evaluate their features, and highlight gaps that our solution can address.

3.2 Competitor Identification

Several platforms currently provide job and opportunity matching services for students. The key competitors analyzed are:

3.2.1 LinkedIn

Description:
LinkedIn is the world's largest professional networking platform, offering a comprehensive ecosystem for job searching, networking, and career development. It provides a space where users can showcase their skills, connect with professionals, follow companies, and apply for job opportunities directly.

Key Features:

- AI-based job recommendations tailored to user profiles and browsing behavior.
- Professional networking through connections, messaging, and LinkedIn Groups.
- Career insights and learning recommendations based on interests and job history.
- Integration with Applicant Tracking Systems (ATS) and company career portals.
- Mobile-friendly job application experience with "Easy Apply" functionality.

Relevant APIs:

LinkedIn provides several APIs that power integrations with its platform. However, these APIs are primarily available to approved partners and come with strict usage limitations. Key APIs include:

- **Jobs API (Apply Connect / Easy Apply)** – Allows companies to retrieve applicants for job postings and collect resumes.
- **Job Posting API** – Enables the creation and management of job listings on LinkedIn.
- **People API** – Provides access to limited profile information of authenticated users (e.g., name, headline, skills), useful for building matching systems.
- **Organization API** – Retrieves data about companies on LinkedIn.
- **Learning API** – For LinkedIn Learning integration, enabling tracking of learning progress and recommended courses.
- **Share API** – For publishing updates or promoting job opportunities directly on LinkedIn.

Limitations of the API Ecosystem:

- **Partner-only access**: Most advanced APIs require being an official LinkedIn Talent Solutions Partner.
- **Data restrictions**: Access to user data is limited and heavily regulated.
- **No support for real-time or conversational integrations**, such as chatbots or Discord agents.
- **APIs are not optimized for student environments or academic ecosystems**.

Opportunity for Differentiation:

Unlike LinkedIn's rigid and formal integration environment, our system is designed to be **student-first**, **real-time**, and **embedded in familiar digital spaces like Discord**. By enabling autonomous, conversational agents and contextual academic integration (e.g., thesis matching, coursework relevance), we address a niche that LinkedIn's API ecosystem does not serve.

3.2.2 Handshake

Description:
Handshake is a career platform specifically designed to connect college students and recent graduates with internship and entry-level job opportunities. It partners directly with universities and employers to

create a tailored and secure environment for student career development, offering a centralized space for job discovery, employer outreach, and career event participation.

Key Features:

- University-verified student accounts and job postings aligned with academic credentials.
- AI-based job recommendations based on student profiles, career interests, and academic background.
- Employer outreach tools, allowing companies to directly message students that match job criteria.
- Integrated career event management and virtual job fairs hosted in collaboration with universities.
- Mobile-friendly platform with real-time notifications and personalized opportunity feeds.

Relevant APIs:

Handshake offers APIs primarily for partner universities and employers to streamline data flow and integration into institutional systems. Key APIs include:

- **Employer API** – Allows registered employers to manage job postings, interviews, and events.
- **Job Posting API** – Enables the creation, updating, and retrieval of job postings.
- **Student API** – Lets universities access and manage student data, including profiles and activity.
- **Event API** – For creating and managing career fairs, info sessions, and workshops.
- **Reporting API** – Provides analytics on engagement, applications, and platform activity.

Limitations of the API Ecosystem:

- Restricted Access: APIs are available only to approved institutions or employers within the Handshake network.
- Closed Ecosystem: Handshake is designed to operate as a centralized platform with limited extensibility into third-party tools.

- No support for conversational or real-time integrations, such as Discord, chatbots, or autonomous AI agents.
- Heavily reliant on university partnerships, limiting accessibility for independent platforms or non-affiliated students.

Opportunity for Differentiation:
While Handshake is highly effective within its partnered university networks, it lacks flexibility in communication and integration. Our solution is positioned to complement and go beyond this model by delivering:

- Real-time, conversational engagement through Discord integration.
- Autonomous, proactive opportunity delivery without relying on manual student searches.
- Open access to students outside predefined institutional networks, expanding reach and inclusivity.
- Smart, context-aware matching powered by advanced LLMs and personalized AI agents.

By building on modern communication habits and student-friendly digital spaces, our system offers a more dynamic and inclusive alternative to Handshake's institution-bound infrastructure.

3.2.3 LoopCV

Description:
LoopCV is an AI-powered job automation platform designed to streamline the job search process by automating job discovery, resume analysis, and application submission. It empowers users to find and apply to hundreds of relevant jobs effortlessly by leveraging AI-driven matching and workflow automation.

Key Features:

- AI-based job matching system that scores and selects jobs tailored to user resumes and preferences.
- Automatic job application engine that sends CVs or emails to recruiters on behalf of the user.
- Resume parsing and keyword optimization for improved job-to-resume fit.
- Dashboard with analytics to track application activity, job matches, and response rates.

- Integration with job boards like LinkedIn, indeed, and Glassdoor to widen opportunity scope.

Relevant APIs:
LoopCV offers a set of APIs aimed at automating the job search lifecycle. These APIs are available through a developer portal, with access provided upon request.

- **Job Aggregation API** – Retrieves job listings from 30+ sources based on filters like location, title, keywords, and experience.
- **Resume Parsing API** – Extracts structured data (education, experience, skills) from uploaded CVs in multiple formats.
- **Job Matching API** – Matches parsed resumes to jobs using AI and returns relevance scores.
- **Auto-Apply API** – Enables programmatic submission of job applications or follow-up emails.
- **Analytics API** – Provides metrics such as application count, open rates, and response success.

Limitations of the API Ecosystem:

- **Access upon request:** Developer access and API key require reaching out to LoopCV's team via email.
- **Limited public documentation:** Currently, there is no full open developer portal or sandbox for testing.
- **No real-time conversational integration:** APIs are designed for automation workflows, not Discord or chatbot interaction out of the box.
- **Limited customization on UI level:** LoopCV APIs focus on backend automation, requiring external UI support for interactive use cases.

Opportunity for Differentiation:

Although LoopCV excels at automating job applications at scale, it lacks interactivity, user engagement, and real-time feedback, especially for students seeking internships or curated job discovery.

- **Community-Driven Discovery via Discord:**
  Leverage a Discord-based experience where users can explore opportunities, ask questions, and engage with peers—something LoopCV does not offer.

- **Internship-Focused Filters:**
  Build specialized filters and smart match logic specifically for internships, part-time roles, and junior-level positions—an underserved niche in LoopCV.

- **Conversational AI Matching:**
  Replace rigid forms with a natural language assistant that asks users about their interests, availability, skills, and preferences to surface tailored job matches.

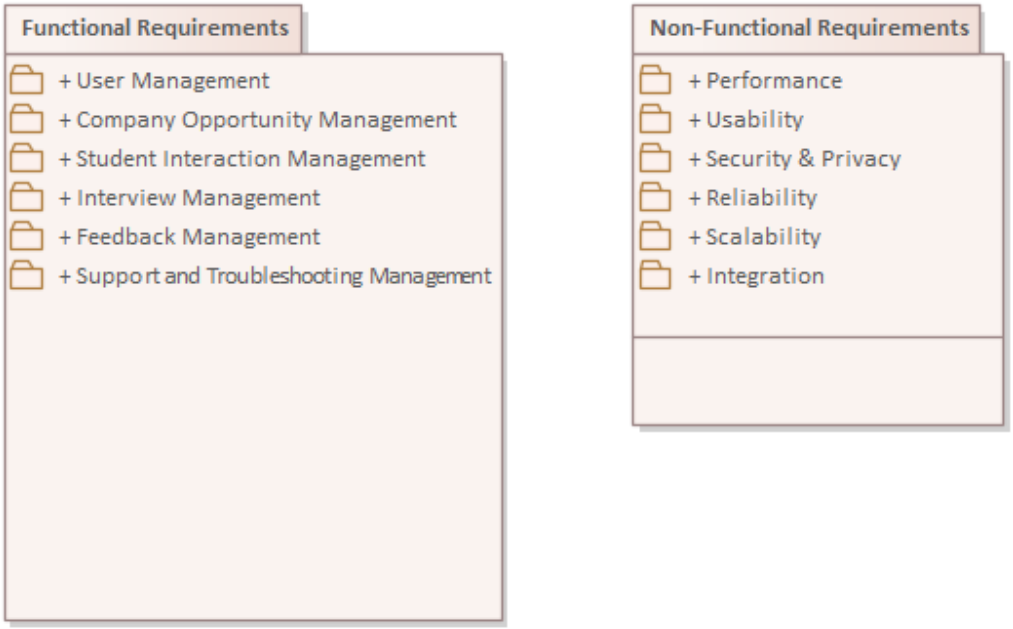
- **Live Notifications and Role Updates:**
  Offer real-time updates and alerts for new relevant job openings, directly in Discord, to keep users engaged and responsive.

- **Personalized Match Ranking:**
  Score and rank job opportunities for each user based on profile fit, relevance, and opportunity quality—prioritizing quality over quantity, unlike LoopCV's high-volume approach.

- **User-Controlled Applications:**
  Rather than automatically applying to jobs, provide users with smart, personalized job recommendations—letting them stay in control of what they pursue.

4. User Requirements

This section outlines the core functional and experiential needs of the target users, primarily students and recent graduates, based on identified limitations in existing platforms and observed user behavior patterns. These requirements will guide the design and development of our AI-powered opportunity matching system.



Identified User Requirements:

- **Personalized Opportunity Matching:**
  Users require intelligent job and internship suggestions that align with their education, interests, skills, and location preferences, without the need to manually filter through generic listings.

- **Conversational Interface & Real-Time Interaction:**
  Students prefer interactive and responsive environments. A conversational AI embedded in a familiar platform (such as Discord) offers a more engaging and intuitive experience compared to static web forms.

- **Internship and Entry-Level Focus:**
  Users seek platforms that prioritize early-career roles, especially internships, part-time positions, and graduate opportunities—roles often underserved or hard to locate on mainstream job boards.

- **Community and Peer Engagement:**
  Access to a peer network or community for sharing opportunities, asking questions, and receiving support is a valuable feature, especially for first-time job seekers navigating the career space.

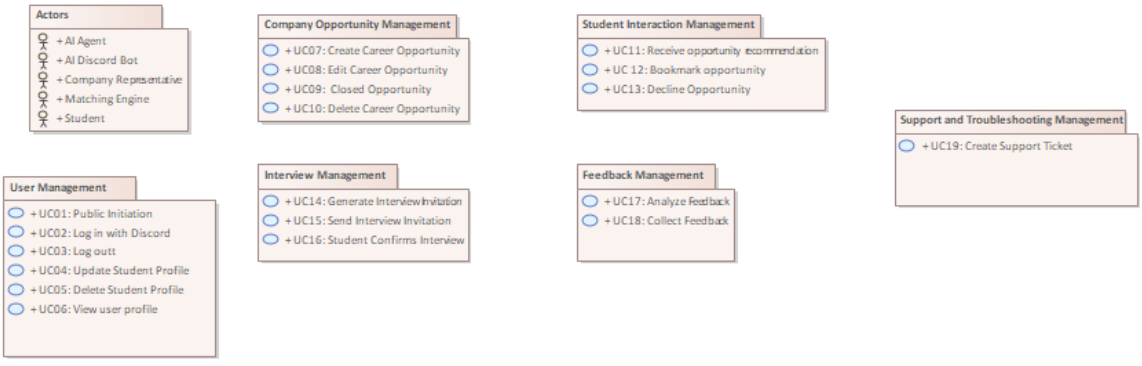
- **User-Controlled Application Workflow:**
  Unlike fully automated systems, users desire visibility and control over where and how they apply. They expect the platform to suggest high-quality roles but allow them to make the final decision.

- **Live Notifications and Alerts:**
  Users want to be notified immediately when new opportunities that match their profile become available. Timely alerts reduce the chance of missing deadlines and improve responsiveness.

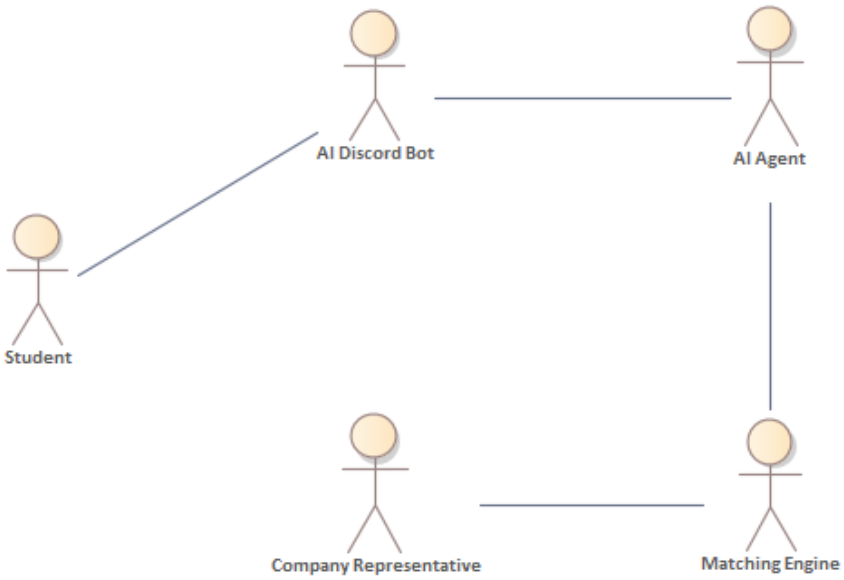
- **Simple, Accessible Interface:**
  The platform should be mobile-friendly, quick to set up, and seamlessly integrate with platforms students already use (such as Discord or email), requiring no technical expertise.

3.1 Functional Requirements

In this section, the functional requirements of the AI-Powered Career Opportunity Distribution System are defined in the form of use cases, reflecting the exact interactions between actors and the system.



3.1.1 Actors



Student:
A user registered in Discord who manages their profile, receives career opportunities, and interacts with opportunities.

AI Discord Bot:
The intelligent, automated agent that manages interactions and matches students to job opportunities, integrated with Discord.

AI Agent:
Service that reformats conversation data and orchestrates the Matching Engine.

Matching Engine:

Component that evaluates a student profile against the full opportunity index and returns a ranked list of matches.

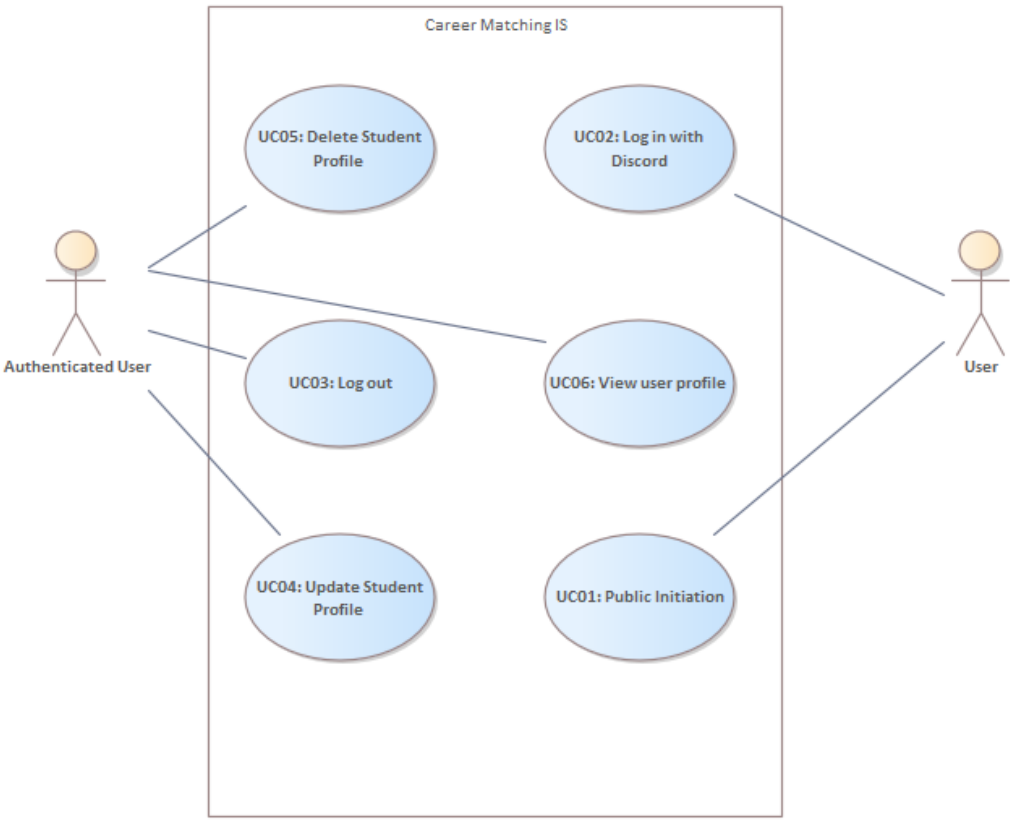
Company Representative:

Posts, edits or withdraws opportunities on EXPERTS.AI; changes are picked-up asynchronously by the Matching Engine.

3.1.2 User Management

This package defines the use cases related to basic user management.

The user management use cases cover signing in to the system through Discord (or signing out), creating an account, viewing personal details, and editing or deleting the user profile.



3.1.2.1 UC01: Public Initiation

The user triggers interaction with the bot in any public server channel by typing “/start” (or a recognised greeting such as “Hi bot”). The command must be visible to the bot and to other users in the channel, yet no subsequent personal data should appear in public. The bot opens a private DM thread with the student and sends a welcome banner plus a short explanation of the service.

3.1.2.2 UC02: Log in with Discord

After collecting the necessary data (name, skills, career interest, resume) a short welcome message confirms sign-in.

Basic path: logging in

1. Bot asks for the student's name; validates that it is not empty.
2. Prompts for skills → student selects tags from a predefined list or types custom ones.
3. Prompts for preferred opportunity type; validates choice.
4. Requests a short free-text statement on interests; length limited to 500 chars.
5. Offers file-upload; accepts PDF, DOCX, or a URL.
6. Displays a summary of the captured profile and asks for confirmation (Save / Edit / Cancel).
7. On Save, the profile is persisted; the student receives a "Profile complete" confirmation and the bot queues the profile for matching.

Alternative path: Missing data

At any step the student might forget to or choose not to add a data, the bot records "unknown" and continues, or asks the user to fill the data.

Alternative path: Cancelling

Student types /cancel; bot abandons onboarding and deletes un-saved data.

3.1.2.3 UC03: Log out

The authenticated user can end the session at any time by sending the /logout command (or by removing the bot's OAuth permission inside Discord's *Authorized Apps* panel)

3.1.2.4 UC04: Update Student Profile

Users can refine their profile at any time. The bot opens an interactive DM wizard where they can adjust:

- Their skills,
- desired opportunity types (internship, thesis, part-time, full-time),
- preferred locations or remote-only flag,
- CV or portfolio URL,
- notification preferences (real-time, daily digest, mute).

3.1.2.5 UC05: Delete Student Profile

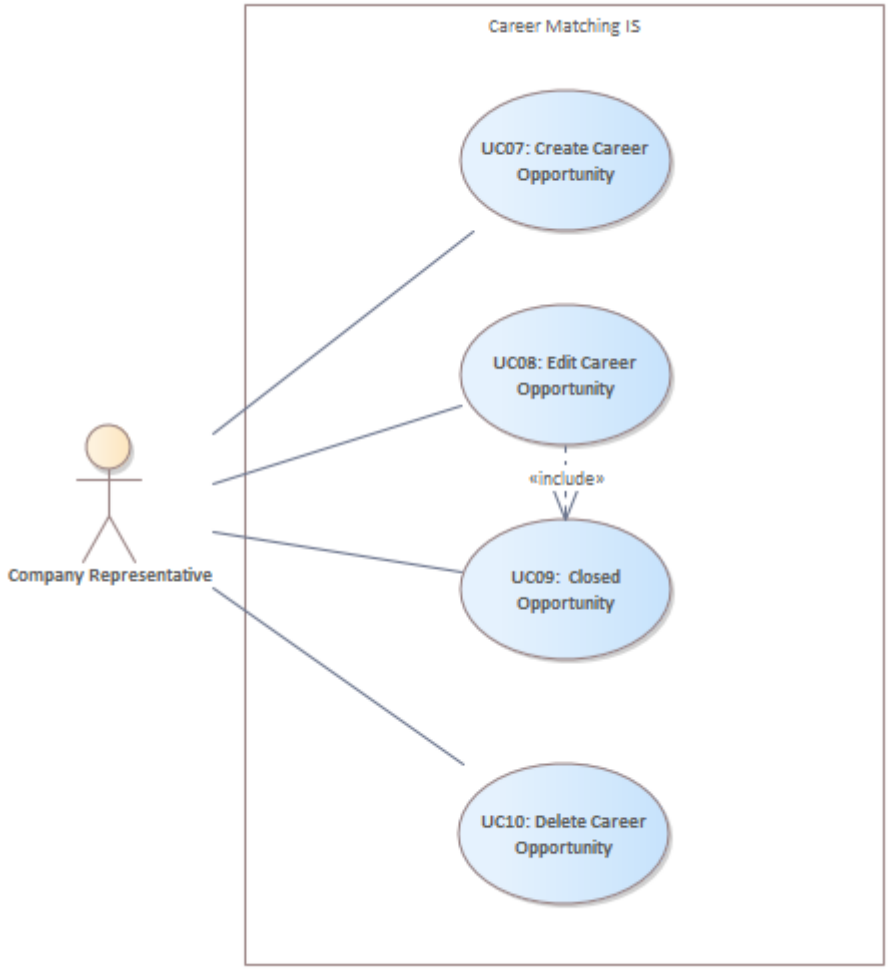
A user who no longer wishes to use the service can delete their profiles. The bot presents a two-step confirmation to prevent accidental loss of data.

3.1.2.6 UC06: View user profile:

By issuing /profile view, the user receives a compact, embed-style snapshot that lists:
• Registered e-mail
• Skill summary
• Current opportunity preferences
• Number of bookmarked roles and interviews in progress

3.1.3 Company Opportunity Management

This package groups all use-cases that enable a company representative (acting through the EXPERTS.AI and Discord bot) to create, publish, maintain and ultimately close an opportunity. The package also covers interactions with matched students, viewing bookmarked candidates, issuing interview invitations and recording hiring outcomes.



3.1.3.1 UC07: Create Career Opportunity

Company representatives add new opportunities detailing required skills, roles, and deadlines to EXPERTS.AI

3.1.3.2 UC08: Edit Career Opportunity

Company representatives update existing opportunities' descriptions, requirements, or application deadlines.

3.1.3.3 UC09: Closed Opportunity

When the position is filled or withdrawn, the bot flips the status to *closed*, stops new

recommendations, and sends a courtesy DM to every student who bookmarked the opening, thanking them and indicating the outcome (“filled”, “cancelled”, etc.).

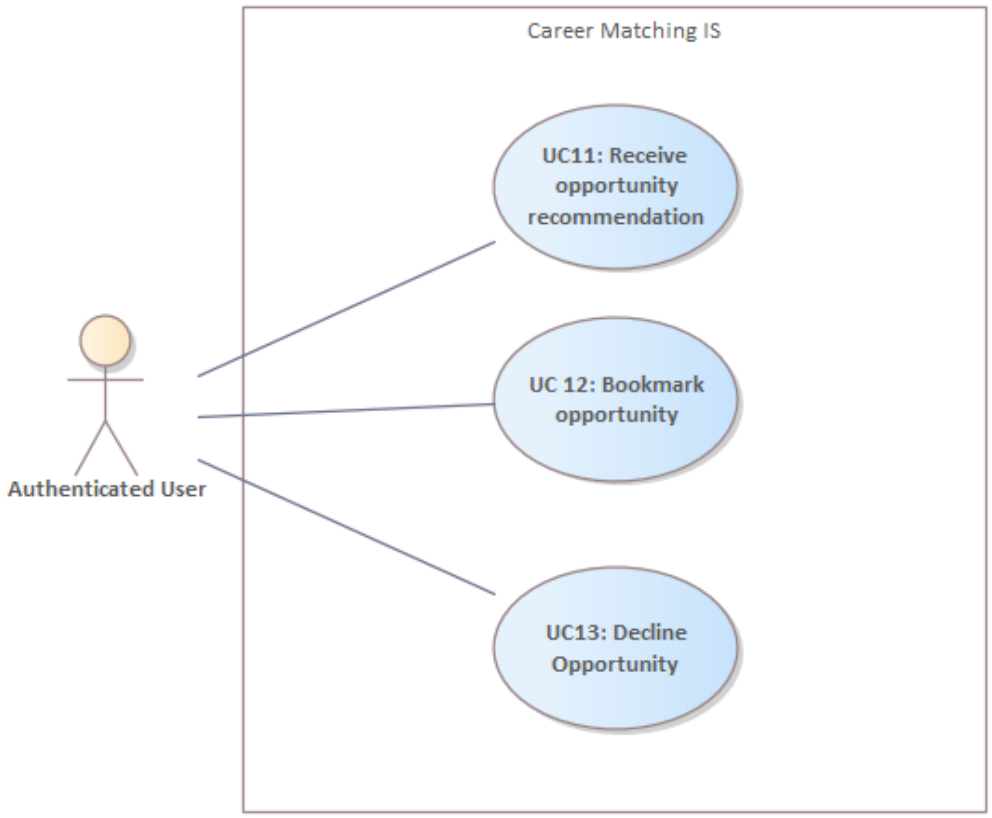
3.1.3.4 UC10: Delete Career Opportunity

Representatives remove opportunities that are no longer active or needed. The action requires confirmation; once executed, the record is purged, and no history is kept except an audit log stub for compliance.

If critical fields change by the company representative (title, location, tech stack), the system re-runs the matcher and, when appropriate, notifies new students while retracting the card from students who no longer match.

3.1.4 Student Interaction Management

This package defines the use cases that let a user receive, review and manage career opportunities delivered by the Discord bot.



3.1.4.1 UC11: Receive opportunity recommendation

The system sends a personalized opportunity card to the student via Discord DM as soon as a student's skills and career interest meets a job offer.

Basic Path: Matching Opportunity with Student

1. Discord Bot acknowledges the request and loads the student's stored profile
2. Bot sends the profile to the AI Agent.
3. AI Agent forwards profile vector to the Matching Engine
4. Matching Engine evaluates the request and matches with the job offers.
5. Job offer has been found.

6. Bot composes a rich message containing title, company, location, duration, tech stack and brief description, plus buttons: Bookmark, see more, decline.

7. System delivers the message to the student's DM.

8. A student reads the message and chooses one of the available actions. (Bookmark, See more, Decline.)

Alternative Path: More opportunities

1. Student reacts "see more."

2. Bot requests the next item in the ranked list (repeat basic steps 6 – 9).

3. Control returns to Basic Path step 8 for the new card.

Exception: No suitable opportunity

1. Matching Engine returns an empty list.

2. AI Agent notifies Bot; Bot apologies, offers to broaden criteria, and closes the card.

3.1.4.2 UC 12: Bookmark opportunity

After an opportunity card is presented, the student can bookmark it to express interest. The system must persist the decision, notify the company, and prepare the ground for a later interview.

Basic Path: Bookmark Opportunity

1. Student bookmarks an opportunity

2. Discord Bot confirms and removes any reaction-based menus from the card.

3. Bot forwards the student's profile and the bookmarked opportunity ID to the AI Agent for persistence and analytics.

4. AI Agent sets the student-opportunity relation to "interested", and creates an Interview Draft (status = pending) that will be processed later by the *Interview Management* package.

5. Bot informs the student that their profile was sent to the company.

Exception: Already Bookmarked

Occurs at step 1 if a duplicate bookmark is detected.

1. Bot responds, "You've already bookmarked this role."

2. No further action; return to waiting for other reactions. If no response is met, bot cancel the bookmarked job offer.

3.1.4.3 UC13: Decline Opportunity

The student explicitly rejects a delivered opportunity card. The system must acknowledge the choice, log it for learning metrics and offer next-step options, while ensuring the same card is not re-sent unless the student later revokes the decision.

Basic Path: Rejecting the Opportunity

1. Students decline the opportunity.

2. Discord Bot records the rejection, then sends a reject request to the AI Agent.

3. AI Agent forwards the request to the Matching Engine along with the current profile vector.

4. Matching Engine marks the opportunity as declined for this student and lowers its ranking weight in future matching.

5. Bot sends a confirmation DM for rejection.

6. Bot asks whether the student wants to see another match or give feedback to redo the matching based on the feedback.

7. Students either agree to see more opportunities, skip or undo.

8. Flow ends:

    - If yes, show next → control returns to *UC11 Receive Opportunity Recommendation* (step 6) for the next opportunity card.

    - If no → DM thread closes; bot waits for the next matching trigger.

Alternative Path A: Change of Mind (Undo Decline)

1. At step 7 the student types "/undo" (or presses a contextual "Undo" button.)

2. Bot cancels the decline flag in the Matching Engine, restores the card.

Alternative Path B: Provide Detailed Feedback

1. Students want to give feedback on the current opportunity card.

2. Bot opens a short free-text prompt.

3. Student submits a comment (e.g., "Salary range not shown").

4. Bot stores the comment, thanks the student, then continues to do the matching again considering the given feedback.

Exception Path: Feedback Timeout

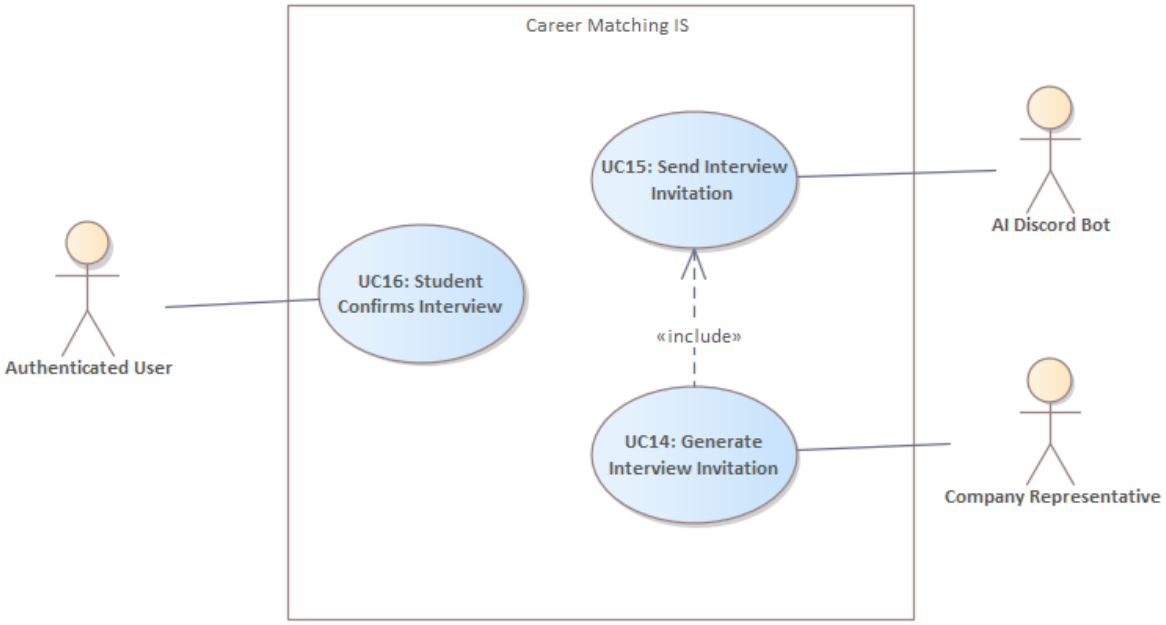
If the student does not respond within 60 seconds

1. Bot sends a reminder.

2. After an additional 60 seconds with no interaction, the bot records no-feedback and proceeds to give other job opportunities.

3.1.5 Interview Management

This package covers the flow that begins when a company decides to interview a book-marked student and ends when the interview slot is either confirmed, rescheduled, or cancelled.



3.1.5.1 UC14: Generate Interview Invitation

Company Representative enters the date, time, format, and interviewer; system creates a draft invitation.

3.1.5.2 UC15: Send Interview Invitation

Discord Bot delivers the invitation to the student, stores a pending status, and starts a response timer.

3.1.5.3 UC16: Student Confirms Interview

Student accepts; bot schedules reminders and updates status to confirmed.

Basic Path: Confirmation of Interview

1. Student clicks Confirm in the Discord interview card.
2. Discord Bot sets interview status to confirmed.
3. AI Agent records the event in the student timeline for analytics.
4. System emits a real-time webhook to the Company Representative: “Candidate confirmed.”
5. Company representative arranges an optimal date and time for the interview.

6. Matching Engine stores and sends to AI Engine.
7. Discord Bot acknowledges the optimal dates and times and sends a message to the user.
8. User selects a time for the interview and types “confirm”.
9. Bot confirms the interview and sends an optional “Interview Prep” message (tips, company link, sample questions).

Exception: Double-booking
If a slot has been taken by another candidate before confirmation.

1. Bot detects clashes, reverts status to pending, apologies.

2. Bot starts to check for other optimal dates.

Exception: No suitable interview

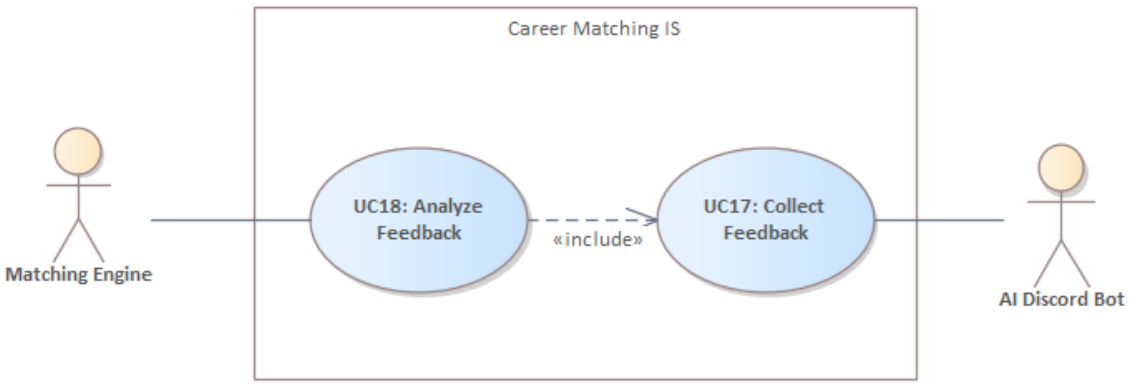
1. Matching Engine returns to an empty list.
2. AI Agent notifies Bot; Bot apologies, offers other job opportunities.

Exception: No response in a 3-day response time

1. Company representative cancels the interview.
2. Matching Engine closes that job offer for the user.
3. AI Agent notifies the bot and sends a message to the user saying that the desired interview can no longer be taken.

3.1.6 Feedback Management

Allow the student to provide feedback about recommended opportunities and interview experiences, enabling the matching algorithm to continuously learn and improve recommendations.



3.1.6.1 UC17: Collect Feedback

post-interview or after reviewing an opportunity, the student provides structured feedback about the relevance of the opportunity or the quality of the interview process. This helps the system better understand how accurate the match was and how helpful the opportunity turned out to be.

Basic Path: Collecting Feedback

1. After a student bookmarks or completes an interview, the Discord Bot automatically sends a message asking:

    - "How relevant was this opportunity for you?" (if bookmarked or declined)

2. The student responds by selecting a rating on a 5-point scale:
   1 = Not relevant / Poor
   2 = Somewhat relevant / Below average
   3 = Neutral
   4 = Relevant / Good
   5 = Perfect match / Excellent

3. The Bot confirms stores the rating in the feedback database and updates profile preferences.

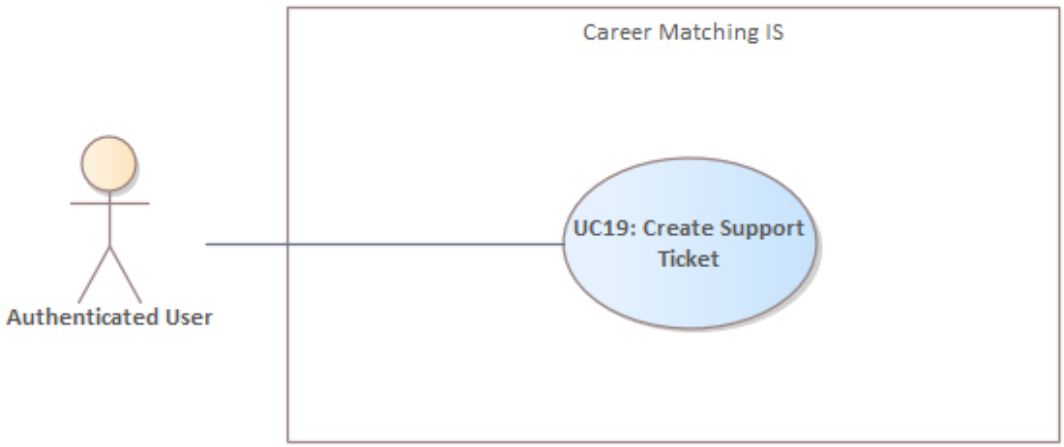
3.1.6.2 UC18: Analyze Feedback

All collected feedback data is automatically analyzed and used to improve the relevance and accuracy of future recommendations. The Matching Engine updates its scoring weight and

matching threshold based on trends and patterns from recent feedback. Next time the student requests a match, the Discord Bot uses the refined scoring from the updated Matching Engine.

3.1.7 Support and Troubleshooting Management

This package defines the use cases related to reporting, managing, and resolving support issues encountered by students while interacting with the Discord Bot or any connected services.



It includes ticket creation, automated first-line troubleshooting, and escalation paths for unresolved or complex problems.

3.1.7.1 UC19: Create Support Ticket

Students report technical issues or unexpected behavior directly through the Discord Bot, which generates a structured support ticket for follow-up.

Basic Path:

1. Student types of the command /support or selects the “Need Help” button in the bot interface.

2. Discord Bot opens an interactive form asking the student to describe the problem, optionally attach a screenshot or paste an error code.

3. Upon submission, the bot confirms and stores a new ticket record.
4. If the issue matches a known pattern or FAQ, the bot sends an automatic response with a possible solution.
5. If no automatic solution is found, ticket data is logged in the system and added to the troubleshooting queue.
6. A system administrator reviews the ticket, investigates the issue, and provides a solution or further instructions.
7. Once the resolution is submitted, the bot notifies the student

Alternative Path: Anonymous Request

If the student is not authenticated when using "/support".

1. Bot responds with "You must sign in first to report an issue."
2. Redirects the student to the login flow and resumes ticket creation after successful sign-in.

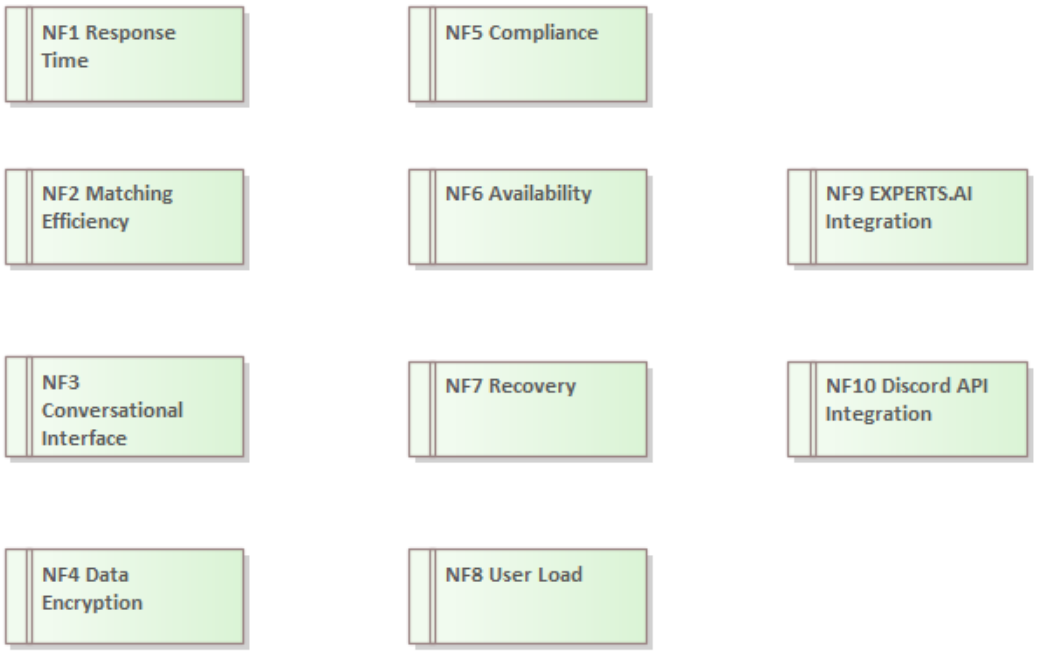
Exception Path: Invalid Input

Student submits an incomplete form (e.g., blank description):

1. Bot displays inline validation error.
2. Students are prompted to revise and resubmit.

3.2 Non-Functional Requirements

This section specifies measurable qualities and constraints on the system's behavior, which ensure usability, reliability, and performance of the AI-powered Career Opportunity Distribution System.



3.2.1 NF1 Response Time:

The system shall respond to user interactions within 3-5 seconds at most.

3.2.2 NF2 Matching Efficiency:

Opportunity-to-student matching must be completed within 10 seconds from the moment a new opportunity is posted.

3.2.3 NF3 Conversational Interface:

The system interactions shall be structured clearly and understandably, such that any primary interaction (e.g., profile creation, expressing interest, viewing details) requires no more than 3 interaction steps from start to finish. The clarity will be evaluated based on usability testing and feedback questionnaires:

1. At least 80% of test users must successfully complete given interaction tasks without additional help.
2. User satisfaction surveys must indicate a minimum average rating of 4 out of 5 regarding clarity and ease of use.

3.2.4 NF4 Data Encryption:

All user CVs and personal data shall be encrypted using AES-256 encryption standards at rest and during transmission.

3.2.5 NF5 Compliance:

The system must adhere strictly to GDPR standards for data privacy, ensuring explicit user consent, transparency in data use, and secure data management practices. Compliance shall be verified by regular internal audits conducted quarterly.

3.2.6 NF6 Availability:

The system shall ensure 99% availability. To achieve this, the proposed architecture includes:

1. Redundant cloud-based infrastructure (AWS, Azure, or similar).
2. Automatic scaling to handle variable loads and ensure stability during peak usage.
3. Failover mechanisms that redirect traffic automatically in case of partial system failure.
4. Regular backups to enable rapid recovery.

3.2.7 NF7 Recovery:

In case of unexpected downtime, the system shall automatically detect issues and restore functionality within 30 minutes using automated scripts and predefined recovery protocols.

3.2.8 NF8 User Load:

The system must comfortably support concurrent users actively interacting with the Discord bot and the web application without noticeable degradation in performance

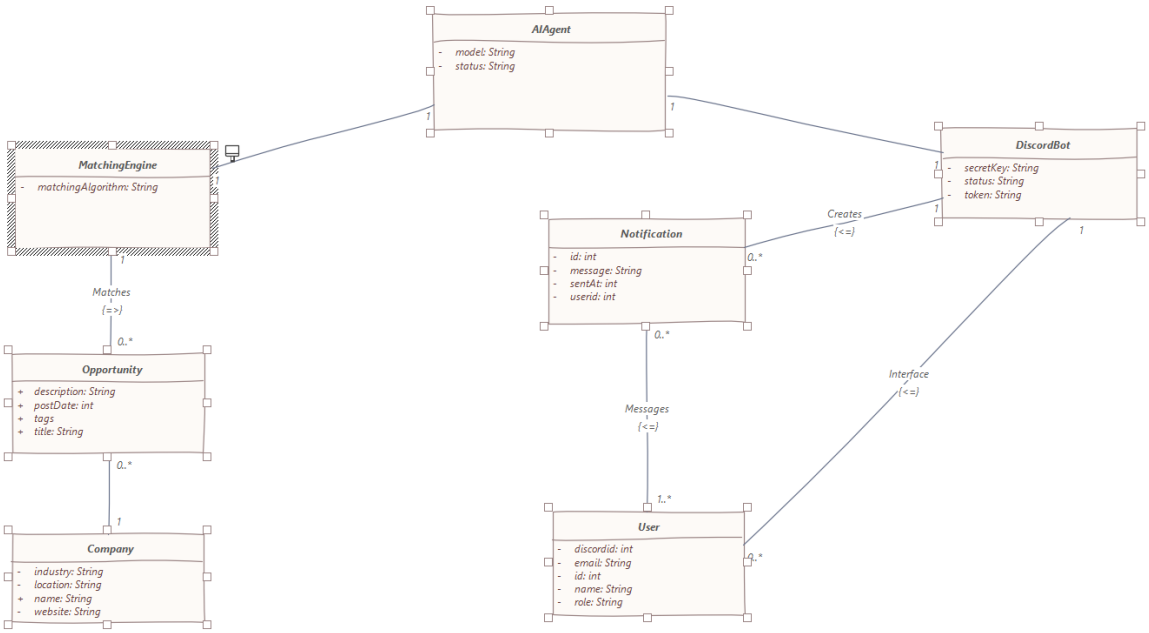
3.2.9 NF9 EXPERTS.AI Integration:

The integration between the system and EXPERTS.AI must guarantee reliable data synchronization. This will be validated through regular synchronization tests conducted monthly, ensuring no loss of information or delay in data transfer.

3.2.10 NF10 Discord API Integration:

The integration with Discord’s API must be stable. The reliability will be monitored continuously through system logs and monthly reporting

5. Domain Model



5.4 User

The User class represents a general user of the system, which can be either a student or an administrator. This class holds shared data like name, email, role, and Discord identity, and serves as a base entity for both **Student** and **Admin** specializations. Each user must be assigned one role and may receive multiple notifications sent through the Discord bot.

5.5 Notification

The **Notification** class represents a message sent to a user, typically via Discord, to inform them of important events such as job recommendations, updates, or system alerts. Each notification is associated with one user and includes content and a timestamp indicating when it was sent.

5.6 Discord Bot

The **Discord Bot** class represents the system's integration interface with Discord. It handles communication with users, sending messages such as job recommendations or alerts directly to their Discord accounts. The bot has credentials and a status that indicates whether it's online and functioning.

5.7 AI Agent

The **AI Agent** class represents the core artificial intelligence component responsible for recommending job opportunities to students. It works alongside the **Matching Engine** to evaluate student profiles and deliver tailored suggestions based on learned patterns and preferences.

5.8 Matching Engine

The **Matching Engine** class handles the logic for matching user profiles with job opportunities. It executes the algorithm defined in the system and provides data to the AI agent to support the decision-making process. This component is independent and reusable, allowing for algorithm upgrades or different matching strategies.

5.9 Opportunity

The **Opportunity** class represents a job offer or internship available to students. Each opportunity is posted by a company and includes details such as title, description, tags, and the date it was posted.An opportunity can be suggested to one or more students by the AI agent.

5.10 Company

The **Company** class represents an organization that offers job opportunities to students. Each company can publish multiple opportunities, and its data helps students understand the context and credibility of the offers.