Summary

General Introduction

Internify is a comprehensive web-based system designed to enhance and streamline the internship management process. In today's rapidly evolving job market, internships play a crucial role in bridging the gap between academic knowledge and professional experience. However, many institutions and companies still rely on outdated and inefficient methods for handling internships, leading to miscommunication, delays, and a lack of transparency. Our platform aims to address these issues by providing a centralized and intelligent system that facilitates smooth interactions between students, companies, and administrators. The system will ensure that every aspect of the internship process, from application submission to final report approval, is managed efficiently and effectively within a unified digital space.

Problem Statement

The internship process is often plagued by inefficiencies due to the lack of a centralized management system. Many students struggle to find relevant internship opportunities, while companies face challenges in handling and reviewing applications efficiently. Moreover, institutions and universities lack a streamlined method for tracking internship progress and ensuring compliance with necessary documentation. Communication between students, recruiters, and administrators is frequently scattered across multiple platforms, making it difficult to maintain a clear record of interactions. These challenges contribute to delays, lost opportunities, and unnecessary administrative burdens, highlighting the urgent need for an intelligent and automated internship management system.

Existing Solutions

JobTeaser

JobTeaser is a platform specialized in internship and job offers for students and recent graduates. It is used by several universities and schools to connect their students with partner companies.
Pros:
•Targeted internship offers based on the student's profile and academic background.
•Integrated collaborative platform with educational institutions.
•Additional features: coaching, career advice, webinars.
Cons:
•Limited access for non-partner universities.
•Few options for advanced administrative tracking of internships.
•Primarily oriented towards large companies, limiting opportunities for SMEs.

JOBTEASER

Internships.com

Internships.com is an international platform dedicated to students looking for internships and short-term jobs. It offers a wide catalog of job postings based on students' fields of study and skills.

Pros:

•Large database with thousands of internship offers.
•Companies can easily post job offers.
•Advanced filters for optimized search (location, salary, field, etc.).

Cons:

•No university integration, meaning no academic tracking of internships.
•Variable offer quality (fake listings or low-quality internships).
•Interface is not always well-suited for structured application tracking.

LinkedIn (Internship & Jobs Section)

LinkedIn offers a dedicated section for internships and job postings for students and recent graduates. This platform highlights professional networking to facilitate recruitment and recommendations.

Pros:

•Direct integration with a professional network and alumni recommendations.
•Messaging system to communicate directly with recruiters.
•Ability to follow companies and view job postings in real-time.

Cons:

•Not specifically designed for internships (mixes internships and full-time jobs).
•No administrative management for internships (contracts, academic tracking, etc.).
•Some advanced features require a LinkedIn Premium subscription.

Comparison Table

Criteria	JobTeaser	Internships.com	LinkedIn
Student Targeting	Yes, via university partnerships	Yes, but no university tracking	No, general platform
Job Offer Database	Large but focused on big companies	Very large	Very large
Offer Filtering	Yes, optimized search	Yes, advanced filters	Yes, but not very precise
Application Management	Yes, tracks applications	Partial	Unstructured
Internship Tracking	Basic, depends on universities	None	None
Company Interaction	Yes, direct contact	Yes	Yes, via messaging

Proposed Solution

Internify, an Intelligent Internship Platform designed to streamline the internship process by facilitating seamless interaction between students and companies. This platform will provide a structured environment where students can easily search and apply for internships, track their applications, and receive updates in real-time. Companies will be able to post opportunities, review applications, and communicate directly with candidates. To ensure better academic oversight, the platform will include features for monitoring internships, managing required documents, and validating reports. Additionally, a built-in messaging system will enhance collaboration by enabling direct communication between all stakeholders. By integrating these functionalities, the platform will offer a more efficient, transparent, and user-friendly approach to internship management, addressing the shortcomings of existing solutions and ensuring a smoother transition from education to the professional world.

Functional Requirements

Task	Description	Responsible
1. Implemented CRUD operations for contract entity	Developed full Create, Read, Update, and Delete operations for managing contract records in the system. This included setting up the controller, service, and repository layers in the Spring Boot application.	Mariem Jemaiel
2. Implemented CRUD operations for report entity	Built CRUD functionalities for managing reports, following the same layered architecture approach used for contracts. This ensures consistency and code reusability across the project.	
3. Added functionality to upload PDF files in contract	Enabled users to upload PDF documents related to each contract. The uploaded PDFs are stored as base64-encoded strings in the database. This approach simplifies data transmission, ensures compatibility, and avoids managing a separate file system.	
4. Added functionality to download PDF files in contract	Implemented the ability to download the stored base64-encoded PDFs by decoding them and sending them back as file responses. This allows users to retrieve and view the original contract files securely.	
5. Watermarking PDF files with enhanced security	Integrated logic to automatically apply multiple tilted watermarks to each uploaded PDF file. The watermarks are designed to cover the page diagonally,	

	making it harder to alter or misuse the documents. This feature boosts document security and traceability.	Mariem Jemaiel
6. Contract status management system	Introduced a status field (e.g., PENDING, SIGNED) for contracts and implemented the logic to update and track the current state of each contract. This enables better workflow control and monitoring.	
7. Statistical function for contract status	Developed a function that returns statistics (e.g., count of each status) to help monitor the distribution of contracts based on their current status. This can later be used for dashboard visualizations or reports.	
8. SMS notifications on contract status change	Integrated an SMS notification system that triggers every time the status of a contract changes. This ensures that concerned parties are informed in real-time of any updates to contract statuses.	
9. Dynamic search for contracts by ID and status	Implemented a flexible search feature for the contract entity that allows users to retrieve contract data dynamically by providing either an ID, a status, or both. This improves the usability and efficiency of the contract management module.	
10. Statistical function for validated company reports	Developed a stat function to calculate and return the number of reports that have been validated by the company. This helps in monitoring and evaluating	

	the report validation process and can support future reporting/dashboard needs.	Mariem Jemaiel
11. Integrated signature pad for report signing	Added a digital signature pad feature allowing users to sign reports directly within the application. Once a report is signed, it automatically updates the validated by company field to "yes", ensuring that validation status is always in sync with user actions.	
12. Email notification on report signature	Integrated an email notification system that sends an alert immediately after a report is signed. This keeps relevant stakeholders informed in real-time whenever a report has been validated via a signature.	
13. Displaying signature in downloaded report	Enhanced the report generation feature to embed the captured signature at the bottom right corner of the report when it's downloaded. This adds a layer of authenticity and verification to each report.	
14. Research function for reports by ID and validation status	Implemented a dynamic search feature for reports that allows querying by report ID or by validation status. This helps users quickly access specific reports or filter by those validated by the company.	
15. Summary functionality using text frequency	Implemented a feature that summarizes reports by analyzing the frequency of words within the text. The system identifies the most relevant and repeated terms to generate a concise version of the report. This improves	

	readability and allows users to grasp key insights quickly without reading the entire content.	Mariem Jemaiel
16. **Implemented an AI-based student final grade prediction system.**	Developed a machine learning model to predict students' final grades based on multiple academic and personal variables. Collected and preprocessed data including attendance records, continuous assessment scores, etc. Designed and trained predictive models using appropriate algorithms, evaluated model performance, and fine-tuned parameters to enhance accuracy. The system aimed to provide early performance insights, enabling academic staff to identify at-risk students and implement timely intervention strategies.	
1. CRUD operations for messages	Users can create, view, update, and delete messages within a conversation in real time. These actions reflect instantly in the UI and backend.	Samer Ghazouani
2. Read indicator (read/unread)	Each message has a read status that updates in real time when the recipient views it, allowing clear tracking of seen/unseen messages.	
3. Sending messages with attachments	Users can send messages with file attachments such as images, PDFs, or audio. The system handles format validation and size restrictions before sending.	
4. Support for voice messages	Users are able to record and send voice messages that are playable directly within the chat interface.	
5. Ability to pin a message	Specific messages can be pinned by users to highlight important content. Pinned messages stay accessible	

	across sessions and are always displayed on the top of the conversation.	Samer Ghazouani
6. Distribution of messages by type	The system classifies all messages by type (text, image, audio, PDF) for each conversation and user, and updates these stats in real time.	
7. Total messages sent/received per user	Displays a real-time count of how many messages each user has sent and received, updated dynamically as messages are exchanged.	
8. Creating a conversation using email	A user can start a new conversation by entering the email of another user. The conversation is created instantly and added to the list.	
9. Deleting a conversation	Users can delete any existing conversation, which removes it from their view in real time.	
10. Display of conversations sorted by favorites/date	Conversations are automatically sorted based on whether they are marked as favorites and the time of the last message sent or received.	
11. Display unread message count per conversation	Each conversation shows a real-time counter of unread messages, which decreases as messages are read.	
12. Automatic notification for new messages	The system can notify the user in real time when a new message is received in any active conversation.	
13. Filtering/searching messages by keyword or date	Users can search messages dynamically within a conversation by typing keywords or selecting a date range. Results update instantly.	
14. Marking a conversation as favorite	Users can mark conversations as favorites for easier access. This status is saved immediately and reflected in the sorting.	

15. Enabling/disabling notifications per conversation	Notifications can be turned on or off for each conversation, and changes take effect immediately.	Samer Ghazouani
16. Average number of messages per conversation	Calculates and displays the average number of messages exchanged per conversation, based on live data and their types.	
17. Time between exchanging messages	Analyzes the average delay between messages, providing a time in hours, minutes, and seconds.	
18. Toxicity detection in messages	Messages are automatically analyzed to detect inappropriate or toxic content. Toxic messages are blocked on the frontend before being sent.	
19. Friend recommendation system	The system automatically suggests the most engaging user profiles based on past interactions, using a trained Machine Learning model.	
1. Implemented CRUD operations for demands	Built Create, Read, Update, and Delete functionalities for internship demand records using Spring Boot, ensuring a clean separation of concerns across the controller, service, and repository layers.	Youssef Azzouz
2. Uploaded and parsed CVs for internship demands	Enabled CV upload in Base64 format and developed a parser that decodes, analyzes, and extracts skills from the uploaded CVs (PDFs), enhancing the demand's informational quality.	
3. Field-based search and date-based sorting	Implemented flexible search and sort functions that allow companies to filter demands by field of study and sort them by submission date.	
4. Geolocation-based demand matching	Created a feature to find internship demands within a specific geographical radius using latitude and longitude, useful for location-based filtering.	
5. Automatic skill-based demand matching	Added an automated matching system to link demands with offers based on skills, domain,	

	and availability, streamlining the selection process.	Youssef Azzouz
6. Demand status tracking	Introduced a status field (e.g., Pending, Reviewed, Accepted, Rejected) for internship demands and built logic to manage and update their statuses.	
7. Statistical functions for demands	Implemented endpoints to return demand statistics based on field of study and status for future analytics or dashboard integration.	
8. Implemented CRUD operations for responses	Developed full Create, Read, Update, and Delete functionality for responses associated with demands, following layered Spring Boot architecture.	
9. Added response comment moderation and filtering	Built a moderation mechanism that automatically flags or deletes responses containing inappropriate language using a scheduled job.	
10. Integrated notification and SMS alerts on response submission	Enabled real-time alerts through email, in-app notifications, and SMS when a response is created, ensuring that students are instantly informed.	
11. Statistical function for response tracking	Built statistical functions to provide insights on response counts by status or keyword frequency, supporting administrative oversight.	
12. Evaluation system for internship requests	Added a feature for companies to rate and comment on demands, providing students with feedback and enhancing profile credibility.	
13. Implemented an AI-powered CV analysis and clustering system for internship demand evaluation.	Developed a machine learning pipeline to analyze students' CVs by extracting technical and soft skill levels using natural language processing techniques. Designed a scoring mechanism to quantify skill proficiency from unstructured CV data. Integrated a Python-based parser with a Spring Boot microservice to process	

	Base64-encoded CV PDFs and extract structured data. Sent extracted skill features to a predictive clustering model to classify internship applicants into performance-based clusters. This system provides early insights into student profiles, enabling companies to identify strong candidates and tailor recruitment strategies accordingly.	
1. Implemented CRUD operations for offers	Developed Create, Read, Update, and Delete functionalities for internship offer records using Spring Boot, with clean separation across controller, service, and repository layers	Omar Ben Mahmoud
2. Implemented CRUD operations for applications	Built full CRUD functionality for application records. The Application entity contains fields such as status, CV (PDF), motivation letter, and post date, with proper validation and relationships to both Offer and User entities.	
3. Implemented CRUD operations for comments	Created Create, Read, Update, and Delete operations for managing comment records. Each Comment includes fields for content, creation date, sentiment label, and report count. It is linked to the User and Offer entities for traceability and moderation	
4. Automatic expiration and cleanup of outdated offers	Added a scheduled Spring Boot task that runs daily at midnight to automatically delete expired offers, ensuring the platform remains relevant and clutter-free	
5. PDF generation for offer details	Implemented functionality to export offer data as a downloadable PDF containing all key information, making it easy to share or archive offers	
6. Company information scraping from offer descriptions	Integrated a web scraper that extracts company name, location, phone, and description when an offer	

	includes a website link. The extracted information is stored in a .txt file for future reference	Omar Ben Mahmoud
7. Pagination for offers listing	Enabled paginated retrieval of offers to improve performance and usability. The frontend dynamically loads offer pages, with controls for easy navigation	
8. Dynamic keyword-based search for offers	Developed a live search feature to filter offers based on keywords such as title or category, offering a fast and seamless user experience	
9. Offer statistics and performance tracking	Introduced metrics to identify the best-performing offer in the last 30 days, based on engagement indicators like application count and user interactions	
10. Advanced comment moderation with bad word filtering	Implemented automatic detection of offensive language using a predefined bad words list, preventing toxic comments from being posted	
11. Sentiment analysis integration for comments	Integrated Apilayer's Sentiment Analysis API to classify each comment as positive, neutral, or negative, and stored sentiment labels in the database for moderation and analytics	
12. User restrictions based on negative sentiment	Blocked users from commenting for 3 days if their most recent comment is classified as negative, encouraging more constructive interactions	
13. Comment reporting and auto-hiding mechanism	Enabled users to report inappropriate comments. Once a comment receives 3 or more reports, it is automatically hidden, enhancing user safety and content quality.	Omar Ben Mahmoud
14. AI-powered application pertinence prediction and skill matching	Integrated a machine learning model to automatically evaluate the pertinence of each application by comparing required offer skills with student CV skills. The system predicts whether an	

	application is relevant and calculates a match percentage, providing recruiters with clear, data-driven insights into candidate compatibility	
1.Implemented CRUD operations for User and Reclamations	Developed Create, Read, Update, and Delete functionalities for User and Reclamation entities to manage user data and Reclamations efficiently.	Yassine Hmedi
2. implemented Geolocation-Based Authentication with 2FA	Used geolocation during login; if the user logs in from a new location, two-factor authentication is required.	
3.Reclamation Response Notification	Notifies users when their reclamation receives a response to keep them informed and engaged.	
4. Enhanced User Management with SMS and Email Verification	Implemented SMS for password recovery and email verification to improve account security and user management.	
5. Smart Reply Suggestions for Reclamations	Integrated AI to suggest three response options for admins when replying to reclamations, streamlining email communication with users.	
6. Keycloak Integration for Role-Based Access Control	Integrated Keycloak to secure sensitive pages and enable access based on user roles, ensuring that only authorized individuals can view specific content.	
7. Dashboard for Reclamation and User Insights	Created a static dashboard to track pending and waiting reclamations, and display the number of job seekers and companies for users.	
8. Dynamic User Search	Implemented a dynamic search feature to find users by email, username, and other fields in real-time, displaying results instantly as the user types.	Yassine Hmedi
9. CAPTCHA Verification	Added a CAPTCHA system to verify user authenticity and prevent automated submissions during form validation.	

10. AI-Based Anomaly Detection with 2FA Trigger	Implemented an AI-driven system to detect logins from unfamiliar User-Agents. When an anomaly is detected, it flags the session as Anomaly 1 and prompts the user for two-factor authentication to confirm identity.	

Non-Functional Requirements

1/Performance & Scalability :

*Real-Time Processing :
All interactions (document uploads, messaging…) should be processed instantly with WebSockets or event-driven architecture to provide real-time updates.

2/Security & Compliance :

*End-to-End Encryption for Messaging & Documents :
All communications and document transactions should be encrypted to protect sensitive data.
*Multi-Factor Authentication (MFA):
To prevent unauthorized access, the platform should enforce MFA
*Watermarking & Digital Signatures :

Documents must be automatically watermarked and digitally signed by the student and company to prevent forgery.

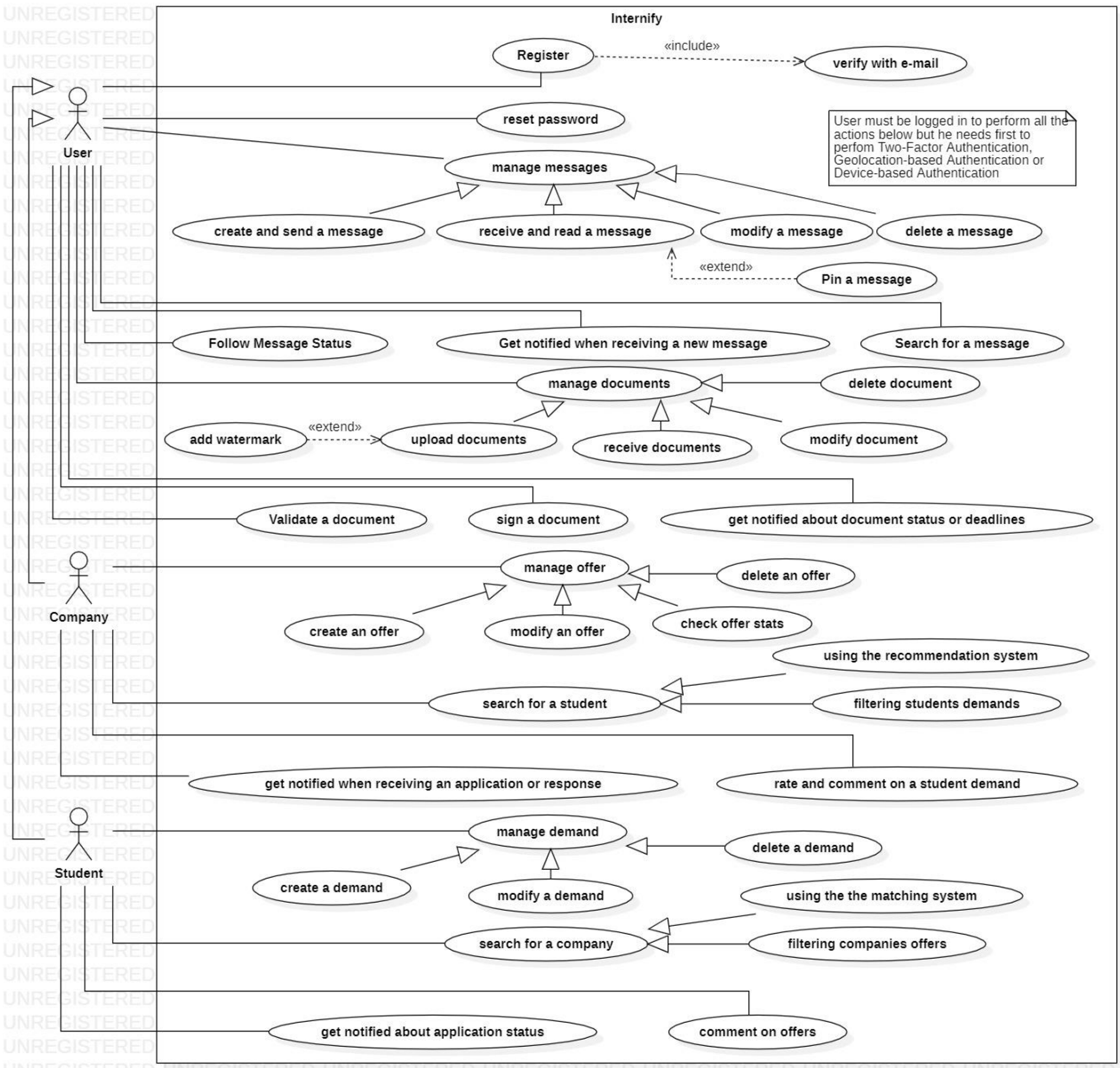
3/Sustainability & Eco-Friendly Tech :

*Paperless Internship Management :
With e-signatures and digital reports the platform should completely eliminate paper usage in internship processes.

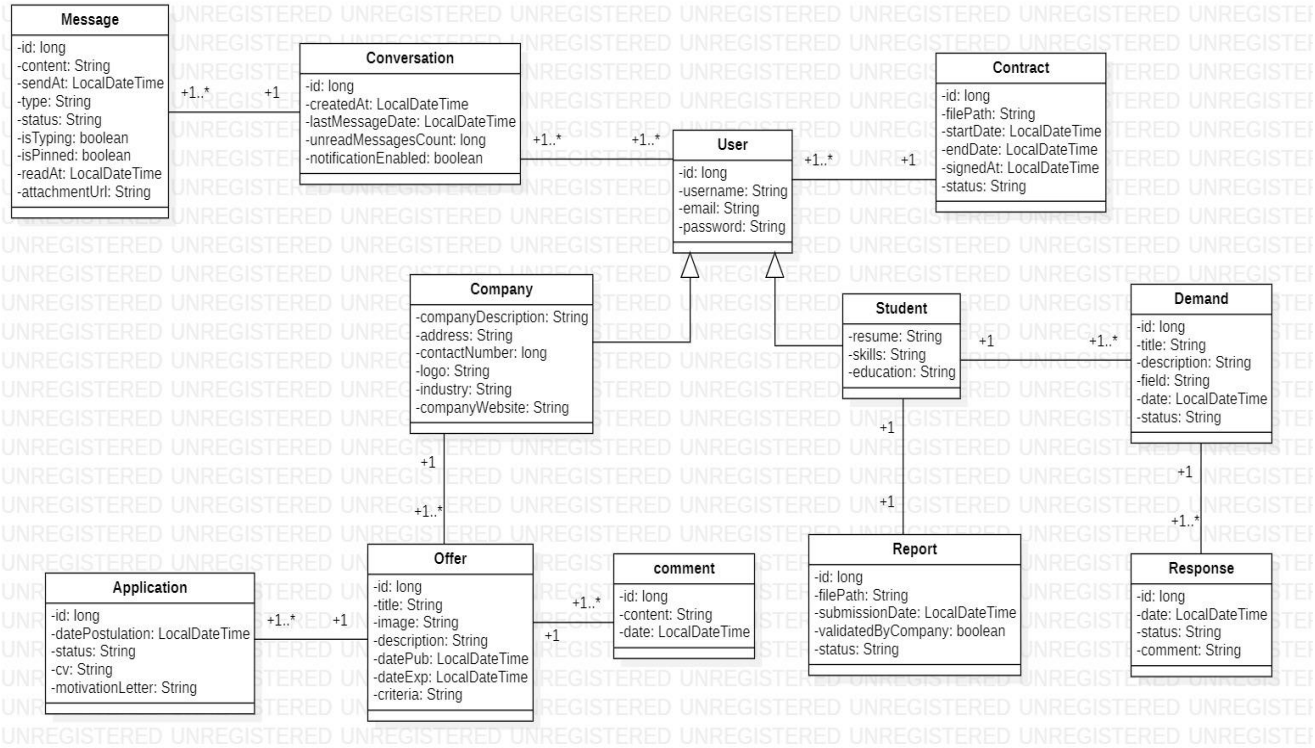
4/User Experience & Accessibility :

*Voice Messages Feature:
This feature allows users to record and send voice messages within the platform, making communication faster, more inclusive, and convenient.

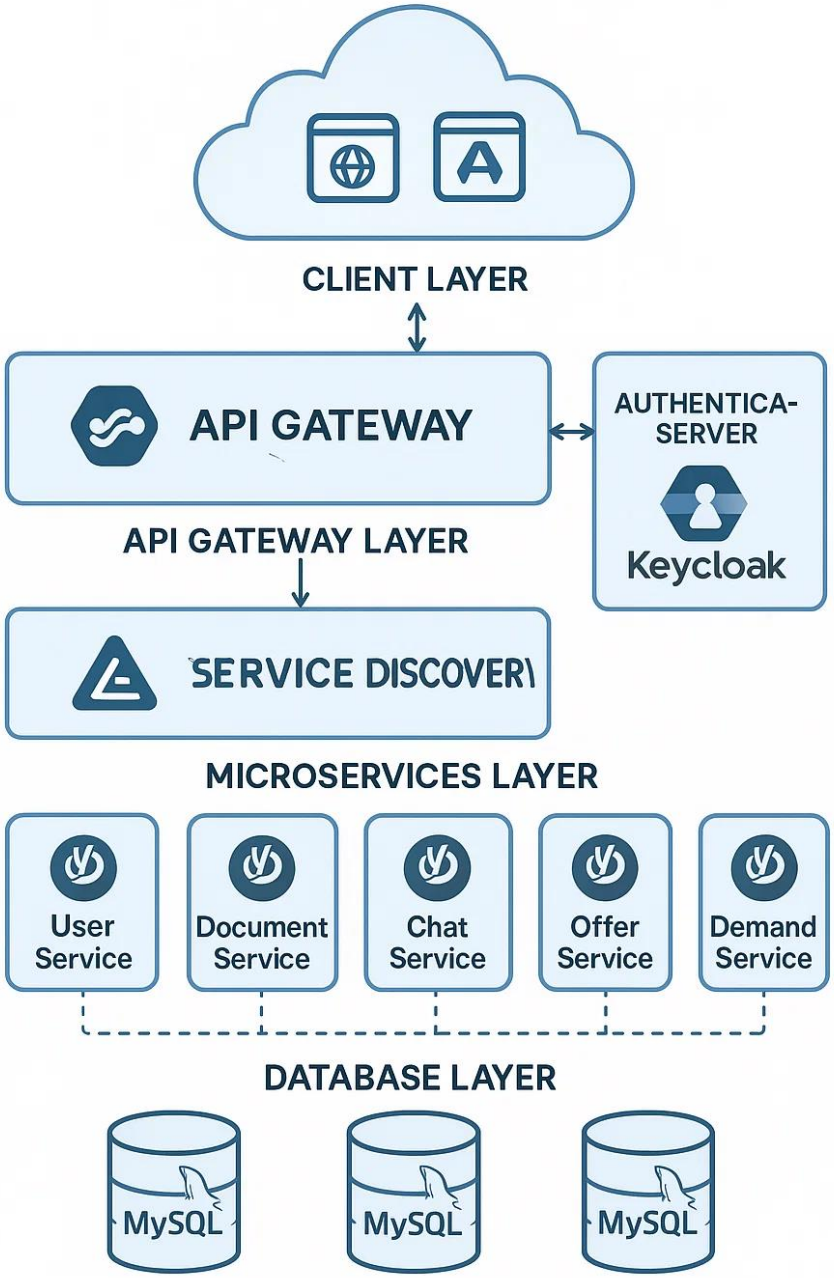
Global Use Case Diagram



Class Diagram



Global Application Architecture



Internify

Cloud-based microservices art