Software Architecture Document

Labor exchange platform

Illia Medvediev, Vladyslav Zraiko, Ihor Kubrysh

Version 1.0

Table of contents

1. Introduction

1.1 Purpose

This document provides the architectural overview of an online platform designed to connect job seekers with employers in Ukraine, focusing on employment, internships, and student practices.

1.2 Scope

The platform encompasses features such as account creation, job posting, application tracking, and in-platform communication, primarily catering to students, job seekers, and employers in Ukraine.

2. Architectural Representation

The architectural representation serves as a visual guide to the overall structure and organization of the system. It provides stakeholders with a clear and concise view of the system's components, their interactions, and the flow of data. The representation aids in understanding the high-level architecture and serves as a foundation for detailed discussions and decision-making.

In the context of this project, the architectural representation will include diagrams and charts that illustrate the key components of the labor exchange platform. These visual aids will highlight the relationships between different modules, the flow of information, and the integration points crucial for the successful functioning of the application. Additionally, the representation will help communicate the system architecture to various stakeholders, fostering a shared understanding of the project's structure and design principles.

3. Architectural Goals and Constraints

The architectural goals and constraints articulate the fundamental objectives and limitations that shape the software architecture for the online labor exchange platform.

3.1 Architectural Goals

1. **Scalability:**
    - *Objective:* Design the architecture to seamlessly scale with increased user demand, ensuring optimal performance as the user base and data volume grow.
    - *Rationale:* Accommodating scalability is crucial for handling varying loads, especially during peak usage times or periods of rapid platform adoption.
2. **User Experience (UX):**
    - *Objective:* Prioritize an intuitive and engaging user experience in the web application, emphasizing ease of use and efficient interaction with sport location functionalities.
    - *Rationale:* Enhancing UX contributes to user satisfaction, encouraging continued platform use and positive feedback.
3. **Security:**
    - *Objective:* Implement robust security measures to protect user data, authentication processes, and sensitive transactions within the mobile platform.
    - *Rationale:* Safeguarding user information and ensuring secure interactions are paramount for building trust and complying with privacy standards.
4. **Flexibility and Adaptability:**
    - *Objective:* Design the architecture to be flexible and adaptable to future technological advancements and changes in sport location management requirements.

- *Rationale:* Embracing flexibility ensures the platform's longevity and the ability to integrate new features or technologies seamlessly.

5. **Performance Optimization:**

    - *Objective:* Optimize system performance to minimize latency, enhance responsiveness, and provide a smooth user experience for both sport enthusiasts and venue administrators.

    - *Rationale:* Improved performance contributes to user satisfaction and retention, critical for the success of a web platform.

4. Use-Case View

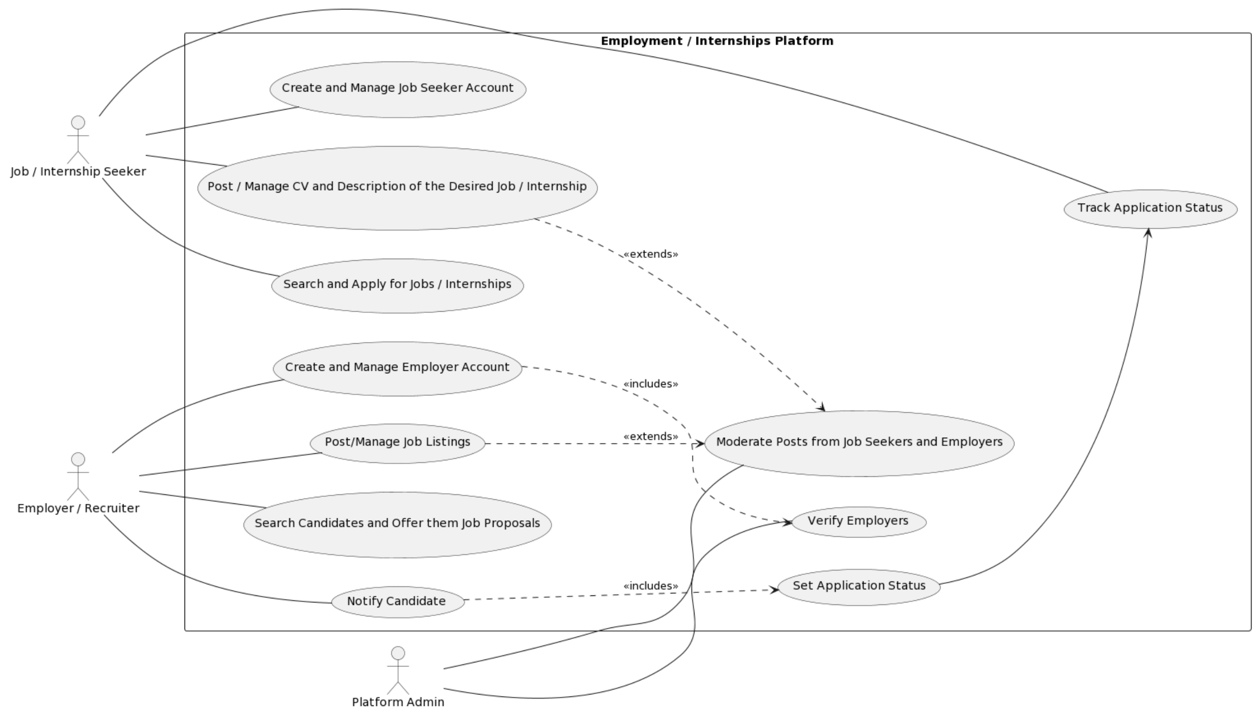
The Use-Case View provides a detailed analysis of the architecturally significant use cases that play a pivotal role in shaping the system's design and functionality. These use cases capture the major interactions between actors and the system, highlighting critical scenarios related to the online labor exchange platform implementation.

The online labor exchange platform use cases are:

- Receive Notifications
- Search for Sports Locations
- Advanced Search
- Book Sports Locations
- Secure Payment Process
- Register/Create Profile
- Profile Verification
- Rate and Review Sports Locations
- Respond to Reviews
- Receive Alerts
- Update Availability
- Manage Bookings
- Reservation Modification
- Dynamic Pricing
- Register/List Sports Location
- Profile Verification

4.1 Architecturally-Significant Use Cases

The architecturally-significant use cases, integral to shaping the system's design, encompass a diverse range of functionalities pivotal for the successful operation of the online labor exchange platform:



Employment / Internships Platform
Job / Internship Seeker
Employer / Recruiter
Platform Admin
Create and Manage Job Seeker Account
Post / Manage CV and Description of the Desired Job / Internship
Search and Apply for Jobs / Internships
Create and Manage Employer Account
Post/Manage Job Listings
Search Candidates and Offer them Job Proposals
Notify Candidate
Track Application Status
Moderate Posts from Job Seekers and Employers
Verify Employers
Set Application Status
«extends»
«includes»
«extends»
«includes»

Figure 4.1: Use-case Diagram

1. **Create and Manage Job Seeker Account**

Description:
Job seekers can register on the platform and manage their accounts, setting up profiles that include personal information, qualifications, and job preferences. This process is the entry point into the platform for job seekers and lays the groundwork for their interaction with the system.

Architectural Significance:
This use case is central to the User Account Management Module, requiring secure authentication services, reliable data storage, and integration with profile and CV management features. It also demands a user-friendly interface to encourage engagement.

2. **Post/Manage CV and Description of the Desired Job/Internship**

Description:
Job seekers can post their CVs and specify the types of jobs or internships they are interested in. This involves not only uploading documents but also entering data that can be parsed and searched by employers.

Architectural Significance:
Influences the CV Management and Search Modules, necessitating a flexible data model to store CVs and job preferences. It also requires a parsing engine to extract and index data for efficient searching and matching.

3. **Search and Apply for Jobs/Internships**

Description:
This use case allows job seekers to search for available positions based on various criteria and apply to them directly through the platform. It is critical for the platform's core functionality of matching job seekers with opportunities.

Architectural Significance:
It impacts the Search Engine and Application Management Module, requiring sophisticated search algorithms and a robust application processing workflow. This use case is vital for ensuring high user engagement and satisfaction.

4. **Create and Manage Employer Account**

Description:
Employers and recruiters register and manage their accounts, which include company information, job posting capabilities, and candidate search tools. They must be able to efficiently manage their presence and opportunities on the platform.

Architectural Significance:
This use case shapes the Employer Account Management Module. It requires validation mechanisms to ensure the authenticity of employers, role-based access control, and an interface for job management.

5. **Post/Manage Job Listings**

Description:
Employers can post new job listings and manage existing ones. This includes specifying job requirements, managing applications received, and updating the status of job postings.

Architectural Significance:
Affects the Job Listing Module, demanding capabilities for content management, real-time updates, and notification systems. It also requires integration with the application tracking system to provide employers with up-to-date information on their listings.

6. **Search Candidates and Offer them Job Proposals**

Description:
Employers and recruiters search the database of job seeker profiles to find suitable candidates and extend job offers directly through the platform.

Architectural Significance:
Critical for the Search and Filtering Module and the Communication Module. It requires a powerful search engine to filter candidates based on various criteria and a secure and reliable messaging system for proposal communication.

7. **Moderate Posts from Job Seekers and Employers**

Description:
The platform admin oversees the moderation of posts and listings from both job seekers and employers to ensure the quality and reliability of the content on the platform.

Architectural Significance:
This use case is significant for the Content Moderation Module. It requires administrative tools to review, approve, or reject user-generated content and mechanisms to automate the detection of inappropriate or fraudulent posts.

8. Track Application Status

Description:
Job seekers can track the status of their applications for jobs and internships, receiving updates as their applications are reviewed and processed by employers.

Architectural Significance:
Impacts the Application Tracking Module, necessitating a state management system that keeps track of the application lifecycle and timely notification services to inform job seekers of status changes.

Each of these use cases contributes to the platform's overall user experience and operational efficiency. The architectural design must ensure that these functions are seamlessly integrated, secure, and performant to support the dynamic nature of the employment and internship matching process.

5. Logical View

The Logical View of the mobile platform is presented through a UML Class Diagram, emphasizing the key layers and their interrelationships. This diagram delineates the logical organization of the system, showcasing classes within the Database, User Interface, Notification Service, Payment Service, Booking Service, Review Service, Search Service, and Authentication Service layers.

5.1 Architecture Overview- Package and Subsystem Layering

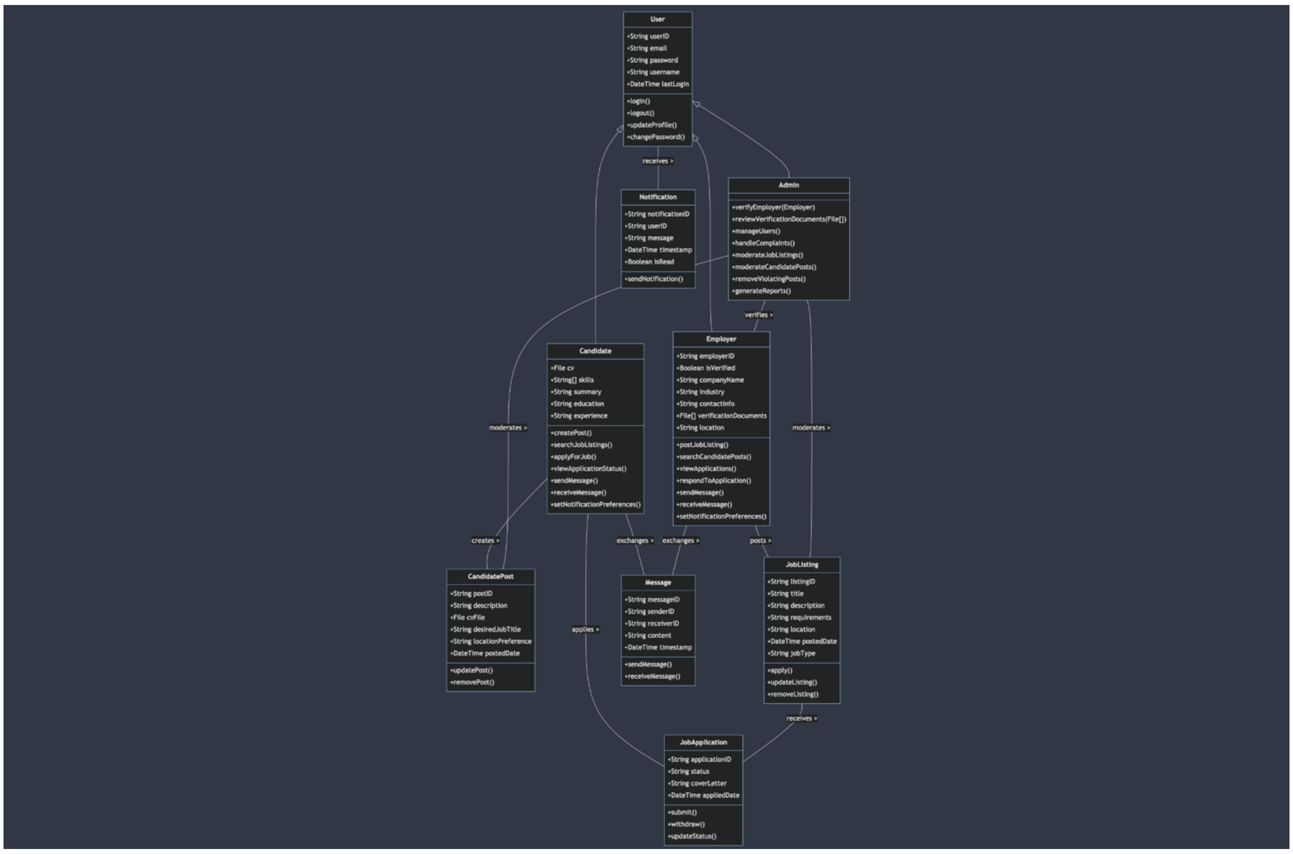


Figure 5.1: UML Class Diagram - Logical View

Diagram Description:

- **User layer:**
  - *Classes:* User

- *Description:* This class represents the base user with common attributes such as userID, email, username, password, and methods for user login management.

- **Admin layer:**
  - *Classes:* Admin
  - *Description:* The Admin class extends from the User class, inheriting its attributes and adding administrative functions like verifying employers, managing content, and generating reports.
- **Candidate layer:**
  - *Classes:* Candidate
  - *Description:* Represents job-seeking users, with additional attributes for skills, education, and experience, and methods for job posting and preference settings.
- **Employer layer:**
  - *Classes:* Employer
  - *Description:* Describes users offering employment, with verification status, company details, and functionality to manage job listings and candidate searches.
- **CandidatePost layer:**
  - *Classes:* CandidatePost
  - *Description:* This entity represents job posts created by Candidates, including details like post ID, job description, and methods to manage these posts.
- **JobListing layer:**
  - *Classes:* JobListing
  - *Description:* Corresponds to the job advertisements posted by Employers, featuring details about the job and methods to manage the listings.
- **Message layer:**
  - *Classes:* Message
  - *Description:* Facilitates communication between Candidates and Employers, containing message content and sender/receiver information.
- **JobApplication layer:**
  - *Classes:* JobApplication
  - *Description:* Connects Candidates to JobListings, tracking the application status and managing the application lifecycle.

Diagram Justification: The UML Class Diagram is chosen to visually convey the static structure of the logical architecture, showcasing the classes and their relationships within each layer. This diagram provides a comprehensive overview of how different layers collaborate and contribute to the overall functionality of the mobile platform. The logical view with the UML Class Diagram serves as a reference for developers, aiding in the comprehension and maintenance of the system's architecture.

This logical view lays the foundation for subsequent sections, where each class and its relationships will be explored in greater detail, offering insights into the functionality and interactions within the mobile platform.

6. Process View

The Process View is detailed through a UML Class Diagram, offering insights into the system's processes and their corresponding classes. This diagram highlights the classes related to User, Notification, Booking, Payment, Admin, FacilityOwner, Facility, and Review, showcasing their functions and interactions.

6.1 Processes

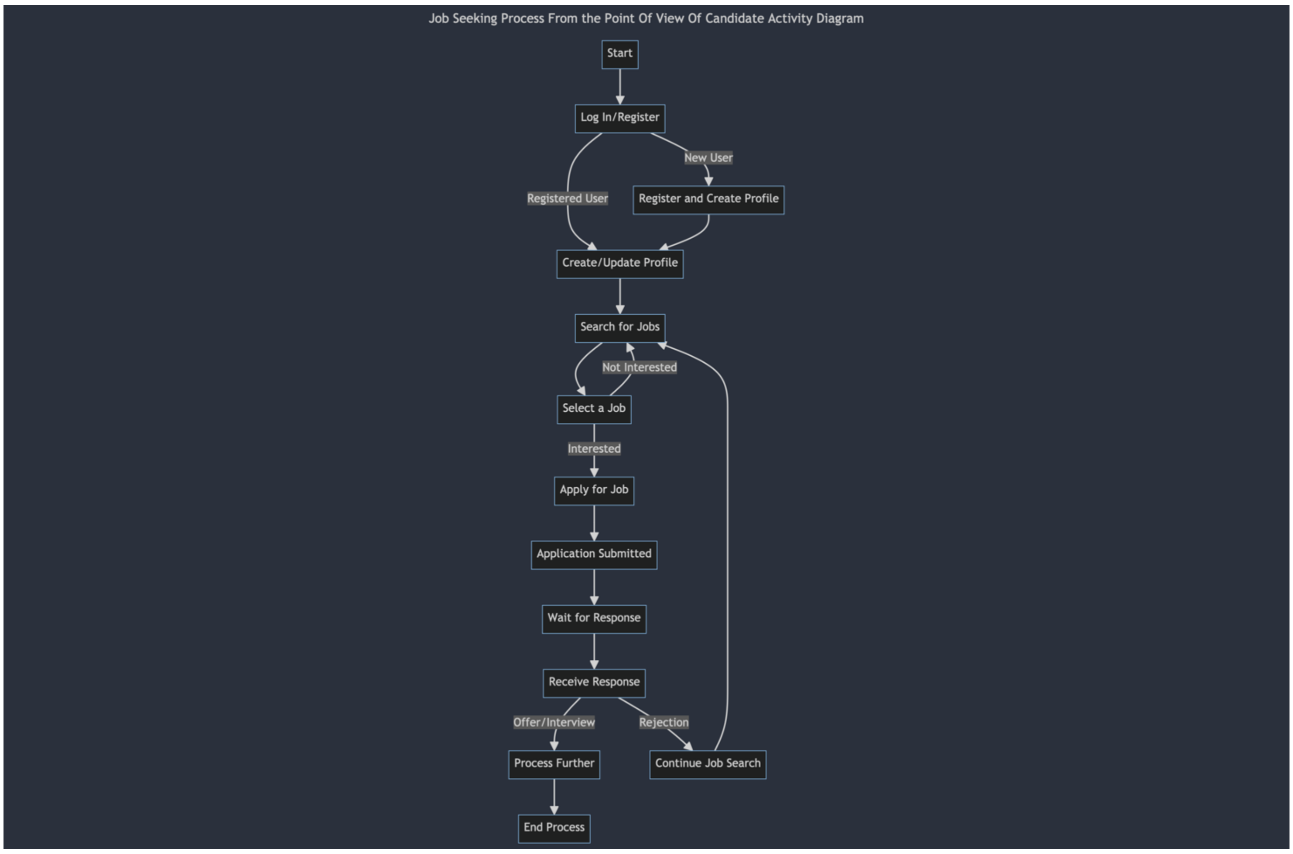


Figure 6.1: UML Class Diagram - Process View

- **Start:**

The process begins when a candidate interacts with the system.

- **Log In/Register:**

The candidate logs in to their existing account or registers as a new user.

- **Register and Create Profile (for new users):**

New users go through the registration process and create their profile.

- **Create/Update Profile:**

Both new and registered users have the ability to create or update their profiles.

- **Search for Jobs:**

Candidates search for jobs that align with their skills and interests.

- **Select a Job (if interested):**

Upon finding a suitable job, the candidate selects it to view more details or to apply.

- **Apply for Job:**

The candidate applies for the job of their choice.

- **Application Submitted:**

The job application is submitted to the potential employer.

- **Wait for Response:**

After submitting the application, the candidate waits for a response from the employer.

- **Receive Response:**

The candidate receives a response, which could be an offer, an invitation for an interview, or a rejection.

- **Process Further (if offer/interview):**

If the response is positive, such as an offer or an interview invitation, the candidate proceeds with further interactions.

- **Continue Job Search (if rejection):**

If the candidate is rejected, they continue the job search process.

- **End Process:**

The process ends, which could mean the candidate has accepted a job offer or decides to stop the job search.

Diagram Justification: The UML Class Diagram is selected to visually represent the static structure of the system's processes, showcasing the functions and interactions among key classes. This diagram provides a clear overview of how different classes collaborate to execute various processes within the mobile platform. The process view with the UML Class Diagram serves as a reference for developers, aiding in the understanding and development of the system's processes.

This process view establishes a foundation for subsequent sections, where each class and its functions will be explored in greater detail, offering insights into the processes and interactions within the mobile platform.

6.2 Process Sequence

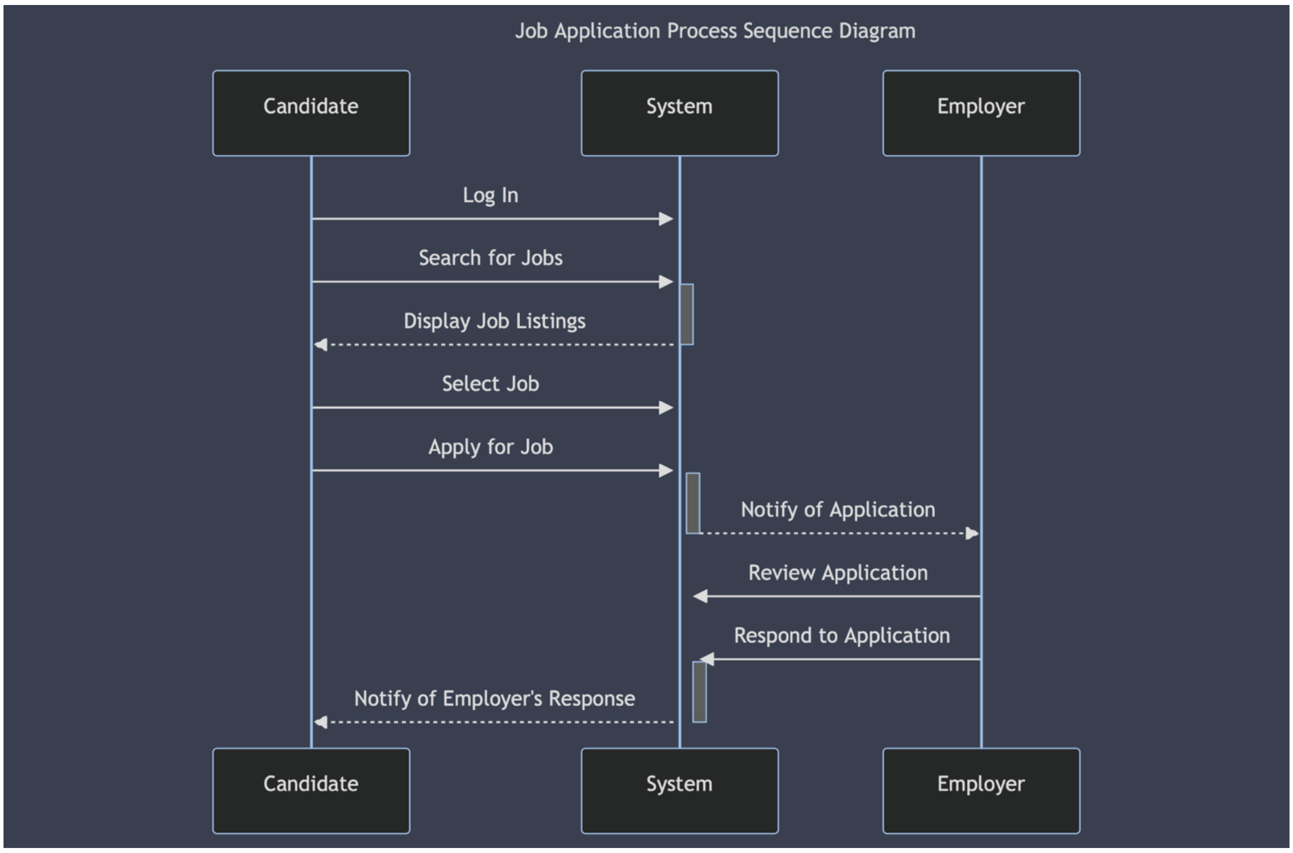


Figure 6.2: UML Diagram - Process Sequence

1. **Candidate**: The individual seeking employment who initiates the process.
2. **System**: The online job application platform that facilitates the interaction between the Candidate and the Employer.
3. **Employer**: The entity that has job vacancies and is looking to hire suitable candidates.

- The process begins with the **Candidate** logging into the system.
- Once logged in, the **Candidate** searches for jobs within the system.
- The **System** then displays the job listings to the **Candidate**.
- The **Candidate** selects a job from the list of displayed jobs.
- After selecting a job, the **Candidate** applies for the job through the system.
- The **System** sends a notification to the **Employer** to inform them of the new job application.
- The **Employer** reviews the job application submitted by the **Candidate**.
- Once the review is complete, the **Employer** responds to the job application.
- The **System** then notifies the **Candidate** of the **Employer's** response.
- The process concludes with the **Candidate** being notified of the outcome (response) of their job application.

Diagram Justification: The UML process sequence diagram describes the interactions between a Candidate, the System, and an Employer during the job application process.

6.3 Processes to Implementation

The Processes to Implementation are detailed through a UML Component Diagram, offering a comprehensive overview of the processes and their corresponding components within the mobile platform dedicated to integrating and managing sport locations in Kyiv.

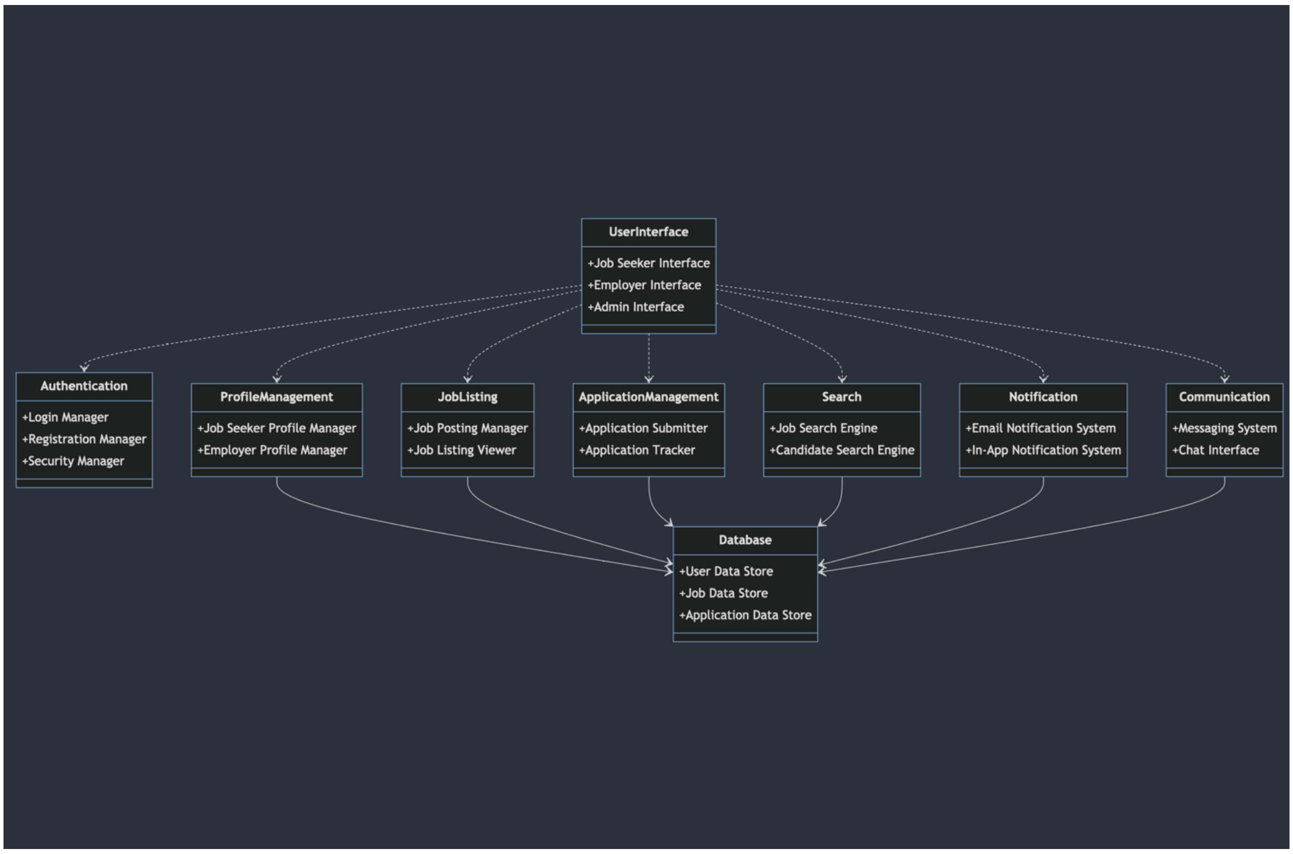


Figure 6.3: UML Component Diagram - Processes to Implementation

Components:

1. **Authentication Component:**

- Login Manager
- Registration Manager
- Security Manager

2. **Profile Management Component:**

- Job Seeker Profile Manager
- Employer Profile Manager

3. **Job Listing Component:**

- Job Posting Manager
- Job Listing Viewer

4. **Application Management Component:**

- Application Submitter
- Application Tracker

5. **Search Component:**

- Job Search Engine
- Candidate Search Engine

6. **Notification Component:**

- Email Notification System
- In-App Notification System

7. **Communication Component:**

- Messaging System
- Chat Interface

8. **User Interface Component:**

- Job Seeker Interface
- Employer Interface
- Admin Interface

9. **Database Component:**

- User Data Store
- Job Data Store
- Application Data Store

Creating a UML Component Diagram with this level of granularity will provide a visual representation of how various processes align with different components in the system, aiding in understanding the system's architectural structure and the relationships between different functionalities.

7. Deployment View

The Deployment View provides an insight into the physical architecture of the mobile platform, illustrating the distribution of components across various servers. The UML Deployment Diagram is utilized to depict the deployment configuration, showcasing the interaction between the web server, application server, client server, and database server.

Deployment Configuration Overview:

- User interactions are facilitated through the Web Server, which hosts the User Interface, Authentication Service, and Review System components.
- The Application Server manages the core business logic, handling processes related to booking, notifications, facility management, and compliance.
- The Client Server hosts client-side components responsible for user interactions, including the User Interface, Authentication Service, and Review System.
- The Database Server stores and manages essential data entities, such as Sport Location Data, User Data, Booking Data, and Review Data.

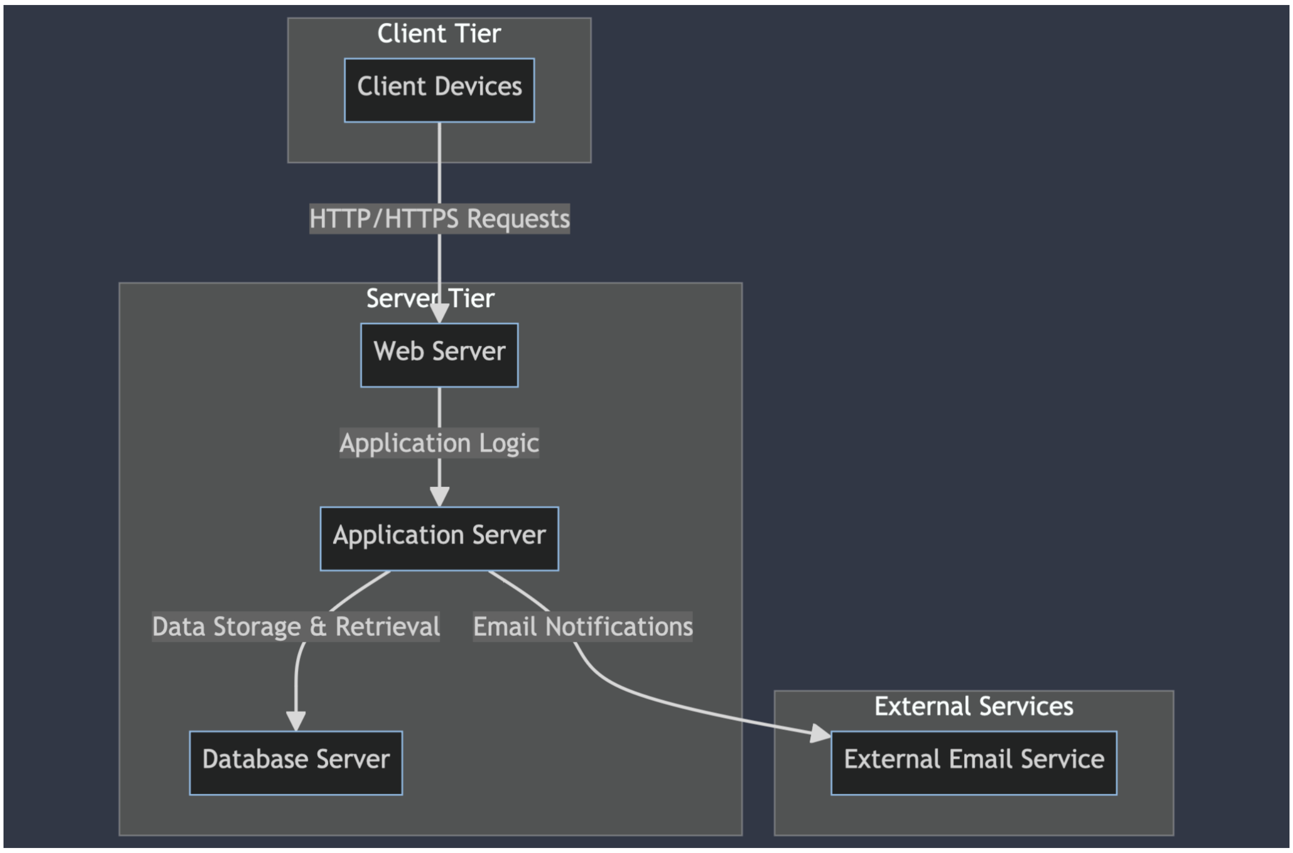


Figure 7: UML Deployment Diagram - Deployment View

7.1 Client Tier

- Description: The Client Tier represents the layer where user interaction occurs. It typically consists of client devices that make HTTP/HTTPS requests to a web server.
- Components: Client devices

7.2 Server Tier

- Description: The Server Tier is responsible for handling user requests from the Client Tier. It processes these requests through a Web Server, which then communicates with an Application Server to execute application logic.
- Components:
  - Web Server
  - Application Server

7.3 External Email Service

- Description: The External Email Service is an external system that the Application Server interacts with to send email notifications to users as part of the application's functionality.

7.4 Database Server

- Description: The Database Server is dedicated to data storage and retrieval. It is the persistent storage layer that the Application Server interacts with to store and retrieve data.

8. Size and Performance

The chosen software architecture aligns with the specified sizing and timing requirements outlined in the Supplementary Specification for the mobile platform dedicated to integrating and managing sport locations in Kyiv:

- The system is designed to efficiently support up to 2000 simultaneous users accessing the central database concurrently, with an additional capacity of handling up to 500 simultaneous users interacting with local servers concurrently.

- The architecture prioritizes responsiveness by ensuring that access to the platform's central database maintains a latency of no more than 10 seconds, optimizing user experience in retrieving critical information.
- A critical performance metric dictates that 80% of all transactions must be successfully completed within 2 minutes, emphasizing the importance of timely and efficient user interactions.

This architecture, exemplified by its client-server model, effectively addresses these sizing and timing requirements. The client-side implementation accommodates diverse user scenarios, whether on local devices or remote connections, emphasizing a design approach that minimizes disk and memory requirements on the client side.

9. Quality

The software architecture strongly upholds the quality requirements stipulated in the Supplementary Specification for the online platform:

- The user interface is designed for optimal user experience. The design aims to be intuitive, reducing the need for additional training.
- Comprehensive online help is integrated into each feature of the mobile platform, offering step-by-step instructions and clear definitions for terms and acronyms, enhancing user understanding and promoting a user-friendly experience.
- The web platform guarantees 24/7 availability, with downtime restricted to no more than 4%, ensuring continuous access for users and reliability in service provision.
- The Mean Time Between Failures exceeds specified requirements, signifying the robustness and reliability of the system architecture in handling potential issues.
- Upgrades to the client portion of the mobile platform can be seamlessly downloaded, providing users with convenient access to system updates and enhancements. This feature facilitates system evolution while ensuring user accessibility and satisfaction.