Contractors: Ospan Ramazan, Saparbek Ulpan, Abdigali Nurasyl

EXERCISE – 1 SOFTWARE REQUIREMENTS SPECIFICATION

System: Job Search Engine (JSE)

Requested By: Ramazan Director of “NoHedHunter”

Purpose:

JSE is an online platform designed to facilitate the connection between employers seeking candidates and individuals searching for employment opportunities. It provides a comprehensive set of tools for both job seekers and employers, ensuring a seamless and efficient job search and recruitment process.

Functional Requirements:

1. User Authentication:

• The system should have a secure user authentication mechanism.

• Users can register as either job seekers or employers.

• Account verification is mandatory and achieved through a confirmation email.

2. Job Search and Posting:

• Job Seekers can search for job vacancies based on criteria such as job title, location, and salary range.

• Employers can post job vacancies, specifying details like job title, description, qualifications, and application deadline.

3. Application Submission and Review:

• Job seekers can submit applications for job positions directly through the platform.

• Employers receive notifications of new job applications and can review applicant profiles.

4. Communication Facilities:

• The platform facilitates communication between employers and job seekers.

• Messaging functionalities and support for interview scheduling should be provided.

5. Logging and Monitoring:

• The system logs each job application, job posting, and communication interaction for future reference.

• Authorized administrators can remotely access these logs for system monitoring.

6. Error Handling:

• In case of application or system errors, the platform should display clear explanations to guide users through potential solutions.

• Users should be given the option to attempt another action after a failed transaction.

EXERCISE – 2 IDENTIFYING USE CASES

Actors:

1. Job Seeker (соискатель)

i. Размещение заявления на работу <<include>> Добавить заявление

ii. Помощь соискателю и обратная связь <<include>> Добавить обращение

iii. Просмотр профилей компаний

iv. Просмотр вакансий

v. Обмен сообщениями <<extend>> Указать тему

vi. Авторизация <<include>> (если нету аккаунта) Регистрация

<<extend>> обязательное подтверждение почты

2. Employer (работодатель)

i. Размещение вакансий

ii. Обработка заявок от соискателей

iii. Просмотр профиля соискателя

iv. Обмен сообщениями <<extend>> Указать тему

v. Авторизация <<include>> (если нету аккаунта) Регистрация

<<extend>> обязательное подтверждение почты

3. Platform Admin (администратор платформы)

i. Просмотр данных <<extend>> (если пользователь дал согласие)

личные данные пользователя

ii. Проверка пользователей на подлинность

iii. Управление пользователями <<extend>> блокировка/разблокировка

iv. Обмен сообщениями <<extend>> Указать тему

4. Support Service (служба технической поддержки)

i. Обработка обращений в службу поддержки <<include>> решение проблем

ii. История обращений

iii. Потомок  Platform Admin

EXERCISE – 3

ID  Description  Actors  Assumptions  Steps  Variations  Quality Concerns  Issues 

1 Размещение заявления на работу Job Seeker имеет аккаунт на платформе.

успешно авторизован в системе. входит в свой аккаунт на платформе.

выбирает опцию "Разместить заявление на работу".

Система запрашивает необходимую информацию, такую как опыт работы, образование и навыки.

вводит требуемую информацию.

Система сохраняет заявление и генерирует уведомление об успешном размещении. Если Job Seeker вводит недостаточную информацию, система отображает сообщение об ошибке и запрашивает дополнительные данные. размещение заявления должно быть выполнено в течение определенного времени. Как обрабатывать случай, если система временно недоступна в момент размещения заявления?

2 Просмотр вакансий Job Seeker успешно авторизован в системе. 1. входит в свой аккаунт на платформе.

2. выбирает опцию "Просмотр вакансий".

3. Система отображает список доступных вакансий с информацией о них, такой как название, местоположение и зарплата.

4. может использовать фильтры для уточнения результатов поиска

5. выбирает интересующую вакансию для получения дополнительной информации. Возможность сортировки вакансий по различным критериям (например, по дате публикации или релевантности). Точность поиска: Система должна предоставлять соответствующие списки вакансий Как обрабатывать ситуацию, когда у вакансии нет дополнительной информации?

3 Размещение вакансии Employer имеет аккаунт на платформе.

успешно авторизован в системе. 1. входит в свой аккаунт на платформе.

2. выбирает опцию "Разместить вакансию".

3. Система запрашивает необходимую информацию, такую как название вакансии, требования, местоположение и срок подачи заявок.

4. вводит требуемую информацию.

5. Система сохраняет вакансию и генерирует уведомление об успешном размещении. Возможность добавления дополнительных деталей, таких как тип занятости (полная занятость, частичная занятость) или вид оплаты. Обеспечение актуальности вакансий: система должна регулярно обновлять список вакансий. Как обрабатывать ситуацию, когда Employer забыл указать местоположение вакансии?

4 Обработка заявок от соискателей Employer успешно авторизован в системе.

разместил вакансию. 1. входит в свой аккаунт на платформе.

2. входит в свой аккаунт на платформе.

3. Система отображает список заявок с информацией о соискателях.

4. может рассматривать профили соискателей и принимать решение о приглашении на собеседование или отказе.

5. Система генерирует уведомление для соискателей о решении работодателя. Возможность фильтрации заявок по определенным критериям (например, по уровню опыта или образования). страница с вакансиями должна загружаться быстро даже при большом объеме данных. Как обрабатывать ситуацию, когда Employer хочет связаться с соискателем для дополнительной информации?

5 Уведомление об отказе Job Seeker, Employer 1. Job Seeker отправил заявление на вакансию.

2. Employer рассмотрел заявление и принял решение об отказе. 1.System получает уведомление от Employer об отказе в заявлении соискателя.

2. System проверяет, что заявление соискателя было рассмотрено.

3. System отправляет уведомление Job Seeker о том, что его заявление было отклонено.

4. Job Seeker получает уведомление и может просмотреть причины отказа. Возможность Employer прикрепить комментарии или обоснование к отказу. Быстрота доставки уведомлений: система должна оперативно отправлять уведомления об отказе. Как обрабатывать случай, когда у Job Seeker есть вопросы по отказу и он хочет связаться с работодателем для получения обратной связи?

Laboratory Work – 2

1. Risk: Lack of clarity of requirements:

• Description: Lack of a clear mechanism for determining the requirements for user accounts (jobseekers, employers), which can lead to problems in the registration and authentication process.

• Importance: 4 (important to prevent problems in the initial stage of use)

• Difficulty: 3 (requires more clarity in documentation)

• Mitigation strategy methods:

1) Develop detailed registration scenarios for each type of user.

2) Conduct testing of enrollment scenarios with users to identify potential issues.

• Considered: Yes.

2. Risk: Improper dependency management.

• Description: Lack of integration between different components of the system, such as between the job search system and the application process, which can lead to unexpected errors.

• Importance: 4 (important to ensure the consistent and effective functioning of the system)

• Difficulty: 3 (requires more thorough definition of dependencies)

• Mitigation strategy methods:

1) Development of clear interfaces between the different components of the system.

2) Regular testing of processes that depend on each other.

• Considered: Yes.

3. Risk: Ineffective technology management.

• Description: Potential problems with the operation of payment systems or errors in the mobile application may cause failures that affect the usability of the system.

• Importance: 4 (Critical for Uptime)

• Difficulty: 3 (more thorough testing and updating is needed)

• Mitigation strategy methods:

1) Implementation of a monitoring system and periodic updating of technological elements.

2) Conducting training of personnel on safe and effective methods of using technology.

• Considered: Yes.

4. Risk: Poor security.

• Description: The lack of reliable protection makes the system vulnerable to web attacks such as SQL injections, XSS, and others, which can lead to the leakage of confidential information and loss of user trust.

• Importance: 5 (critical as unsafe practices can lead to leakage of confidential information and loss of user trust)

• Difficulty: 4 (requires high technical expertise and implementation of additional security tools)

• Mitigation strategy methods:

1) Implement a robust Web Application Firewall (WAF) to filter malicious traffic and prevent attacks.

2) Regularly update and audit the application code to identify and fix potential vulnerabilities.

3) Train staff on safe development practices and ensure compliance with security standards.

• Considered: Yes.

5. Risk: Failure to send a confirmation email when a user registers.

• Description: In case of a trivial server glitch during registration, the user may not receive a confirmation of mail, which creates a risk of missing a potential user and losing trust.

• Severity: 5 (critical, missed users, and loss of trust).

• Difficulty: 2 (technical problem).

• Mitigation strategy methods:

1) Replacement of servers with more productive and stable ones.

2) Ability to resubmit a request for written confirmation.

• Considered: Yes.

6. Risk: Lack of platform administrators and help desk.

• Description: The project will certainly someday serve a lot of traffic, to control it, maintaining a large staff can cost a pretty penny for a young company.

• Importance: 4 (very important from the point of view of the perception of the stakeholders of the parties).

• Difficulty: 3 (finding a reliable CRM service provider).

• Mitigation strategy methods:

1) Traffic monitoring to predict staffing needs.

2) Establishing a long-term partnership with a reliable CRM service provider.

3) Automation of routine tasks through an artificial intelligence bot.

• Considered: Yes.

7. Risk: Users misuse their personal profile and post phishing links.

• Description: Users can abuse their profile by posting phishing links, which directly affects the safety of users' data and devices.

• Severity: 5 (extremely important, affects the security of data and devices).

• Difficulty: 2 (solution from the technical part).

• Mitigation strategy methods:

1) Every time you log in to the service, remind the user that they need to keep their eyes peeled and remind them of possible ill-wishers.

2) Ability to check links before publication through the CRM system.

• Considered: Yes.

8. Risk: Mess among messages/requests to the platform administrator.

• Summary: Confusion in various requests can slow down processing and responses, affecting the responsiveness of the service.

• Importance: 4 (affects the responsiveness of the service).

• Difficulty: 1 (easy to solve).

• Mitigation strategy methods:

1) Adding template themes for competent sorting of messages.

2) Redirect to the current administrator in case of urgent need.

• Considered: Yes.

9. Risk: Error management.

• Summary: Confusion when errors occur can slow down the troubleshooting process.

• Importance: 4 (affects the speed of error resolution)

• Difficulty: 2 (difficulty in identifying problems in time)

• Mitigation strategy methods:

1) It's important to create informative error messages in plain language.

2) Implement an error log that will keep a record of the origin of the error and the cause, for more efficient troubleshooting.

3) Appoint individuals who will periodically test different aspects of the system and collect data.

• Considered: Yes.

LABORATORY WORK 3

EXERCISE 01 – SOFTWARE LIFE CYCLE MODELING

Model Description Advantages Disadvantages

Waterfall В модели Каскада разработки процесс линейно прогрессирует через последовательные фазы: определение требований, проектирование, реализация, тестирование, внедрение и обслуживание. Простота и понятность, подходит для проектов с устойчивыми и четко определенными требованиями. Отсутствие гибкости, сложности с внесением изменений после начала разработки, высокий риск появления дефектов на поздних этапах.

Iterative В модели Итерации процесс разработки разбивается на меньшие циклы, каждый из которых состоит из планирования, реализации, тестирования и оценки. Каждая итерация создает поэтапное увеличение функциональности программного обеспечения. Позволяет получать обратную связь и разрабатывать продукт поэтапно, снижает риски, обеспечивая приоритетное выполнение важных требований, легче вносить изменения по сравнению с моделью Каскада. Требует более активного участия заказчика и обратной связи, может повысить сложность, если итерации не управляются хорошо, возможен перерост объема работ, если требования недостаточно контролируются.

Scrum Scrum - это гибкая методология,которая подчеркивает коллективную работу, ответственность и итеративное движение к четко определенной цели. Она включает разбиение проекта на короткие циклы, называемые спринтами, с частыми инспекциями и адаптациями. Высокая гибкость и адаптивность к изменяющимся требованиям, способствует сотрудничеству и коммуникации внутри команды, позволяет непрерывно совершенствоваться через регулярные ретроспективы. Требует опытных и самоорганизующихся команд, может быть сложно внедрить в организациях, не знакомых с гибкими практиками, сильно зависит от эффективности коммуникации и координации.

Summary: В выборе моделей Waterfall, Iterative и Scrum мы учли специфику проекта "job search engine". Waterfall будет опорой для четкого определения требований, особенно в контексте точного поиска вакансий. Итеративный подход обеспечит гибкость внесения изменений, что критично для постоянно меняющегося рынка труда. С использованием Scrum мы сможем лучше координировать команду, поддерживая регулярные обзоры прогресса и максимизируя эффективность работы в условиях быстрого развития сферы трудоустройства. Такой комбинированный подход оптимизирует разработку "job search engine", делая её более адаптивной и результативной.

SOFTWARE DEVELOPMENT TOOLS

LABORATORY WORK 4

EXERCISE 1 – ANALYZE ARCHITECTURE DIAGRAMS

The purpose of this exercise is to give you concrete practice in evaluating software abstractions.

In this exercise, you will study several architectural diagrams to identify how they could be used. You will also practice identifying and classifying ambiguities.

ACTIVITIES

Perform each of the following activities. If you have questions, issues, or doubts, please ask for help and do not just guess.

1. Individually take 5 minutes to review the purpose of each of the models in the canonical model structure. Think about them in terms of what kinds of elements each depicts and what they are used for – what kinds of questions they can help answer.

2. For each of the cartoons on the following pages, do the following:

a. Discuss what kinds of questions the diagram could help answer then create and record one question that the cartoon could help answer.

b. Take 5 minutes as individuals to study the cartoons for ambiguities. Write them down.

c. As a team, compile the identified ambiguities into a single list.

d. For each ambiguity, decide whether or not your team feels the ambiguity is likely an error and should be referred back to the architect or whether it is clearly something that can be left open to downstream design and coding. On your paper record whether it should be Referred or Left Open. If your decision that it should be referred then describe why your team things so.

VIEW DIAGRAMS:

A. Какие способы оплаты есть?

B. Компания Бредли размещает вакансию на какую должность?

C. От чего зависит веб порт участник ?

D. Лицо трудоустроено или работает на работе в компании?

E. Какую операционную систему использует веб браузер?

Какой тип канала связи используется для связи между роутером, системой и базой данных?

F. Какой из компонентов может отправить электронное письмо ?

EXERCISE 2: IDENTIFYING VIEW TYPES

PURPOSE

The purpose of this exercise is to give you concrete practice in identifying different kinds of views by thinking about how they would be used.

In this exercise you will study several architectural diagrams and for each, identify with which of the three models from the canonical model structure it is associated.

ACTIVITIES

Perform each of the following activities. If you have questions, issues, or doubts, please ask for help and do not just guess.

1. Individually take 5 minutes to review the purposes of each of the models in the canonical model structure, thinking about them in terms of what kinds of elements are depicted in each and what they are used for – what kinds of questions they can help answer.

2. For each cartoon above, using the results from Exercise 1, discuss what kind of structure the cartoon represents, and then identify and record with which of the canonical models it is associated. Also describe the basis for your decision.

A.Этот диаграмма классов является логической моделью. Она показывает структуру системы в терминах классов, атрибутов и отношений между ними. Логическая модель обычно не углубляется в детали физической реализации, такие как схемы баз данных или технологический стек. Она описывает систему на уровне абстракции, который помогает понять ее логику и взаимосвязи между компонентами.

B.Диаграмма, описывающая процесс найма Оуэна в компании Widgetron, фокусируется на ключевых концепциях и взаимосвязях между компонентами системы на высоком уровне абстракции. Она описывает последовательность действий, начиная с встречи Оуэна и Брэдли на профессиональной конференции и заканчивая наймом Оуэна в компании Widgetron. На этом уровне абстракции важно понимать, какие события происходят и как они связаны друг с другом, но не вдаваться в детали реализации, такие как структуры данных или конкретные технологии, что соответствует концептуальной модели архитектуры.

C.Диаграмма связана с конструктивной моделью канонической модели архитектуры. Она показывает структуру и взаимосвязь различных компонентов системы. На диаграмме изображены модули, такие как Job / Ads Web Port, Member Web Port,Non-Member Web Port, SMTP Client Port,HTML / HTTP Classes / Interfaces и SMTP Classes / Interfaces, с центральным положением "Yinzer System". Стрелки "depends" указывают на зависимости между "Yinzer System"и другими модулями.

D.Данная диаграмма связана с концептуальной моделью (Conceptual Model) архитектуры.

Концептуальная модель описывает основные сущности, их атрибуты и отношения между ними в предметной области, не углубляясь в детали реализации. В данной диаграмме представлены основные понятия и связи между ними: Реклама (Ad), Компания (Company), Контакт (Contact), Трудоустройство (Employment), Должность (Job), Соответствие должности (Job Match), и Человек (Person). Не указано, как эти сущности хранятся или реализуются в базе данных или программном коде, поэтому это описание остается на уровне концепций и связей между ними.

E.Диаграмма связана с \*\*физической моделью\*\* канонической модели архитектуры. Она описывает физическую структуру и развертывание системы. В диаграмме представлены пользовательский ПК с операционной системой Linux и веб-браузером, маршрутизатор, компания Yinzer с основным и резервным центрами обработки данных, а также система Yinzer. Легенда объясняет, что прямоугольники представляют элементы окружения (узлы), пунктирные линии - экземпляры компонентов, а сплошные линии - каналы связи.

F.Диаграмма связана с функциональной моделью архитектуры. Она описывает процессы и взаимодействия между компонентами системы, такими как "Non-Member Contacts", "Contacts", "Users", "Advertisements" и "Emails".

EXERCISE 3 – EXPLORING ATM DOMAIN VIA THE INFORMATION MODEL

PURPOSE

The purpose of this exercise is to give your team an opportunity to refine your understanding of the ATM domain (your project domain) through creation of a textual and graphical information models.

ACTIVITIES

Perform each of the following activities. If you have questions, issues, or doubts, please ask for help and do not just guess.

3. Get into teams.

4. Since we assume you have never worked in the ATM domain (your project domain) we assume you have limited understanding of that domain. But, we do assume you have interacted with an ATM (your project).

5. As a team, revisit the roles you identified when developing your use cases. For each role, using the example in Fairbanks section 8.2 create definitions for each of the roles. The definitions should include statements that define the relationships among the roles and between the roles and the ATM (your project). Capture these definitions in a textual information model.

6. Discuss the possibility that there are other roles you did not consider earlier in the course and add definitions for those.

• Исполнитель услуг:

Определение: Фрилансер или физическое лицо, предоставляющее свои услуги. Ищет вакансии или проекты, соответствующие его навыкам.

• Рекрутер:

Определение: Пользователь или внешний консультант, специализирующийся на поиске и привлечении кандидатов для компаний-работодателей. Размещает вакансии, анализирует резюме, связывается с потенциальными соискателями.

• Эксперт по рынку труда:

Определение: Следит за изменениями на рынке труда, анализирует тенденции и предоставляет информацию о перспективах в различных отраслях.

7. Consider the things, other than roles, that are in the ATM domain (your project domain) such as the physical parts of the ATM machine, accounts, etc. Define these in your textual information model.

8. Using the Fairbanks figure 8.2 as a guide create a graphical information model that depicts the relationships between the roles.

9. When your team has completed the previous steps of the laboratory work, upload it to DL.

1) Job Seeker

• Пользователь, ищущий работу, обладающий функциональностью размещения заявлений на работу, получения помощи и обратной связи, просмотра профилей компаний и вакансий, а также обмена сообщениями с другими пользователями. Для входа в систему требуется авторизация, и, если у пользователя нет аккаунта, необходима регистрация. Подтверждение почты обязательно для завершения регистрации.

Отношения:

С Работодателем: Подает заявки на вакансии и взаимодействует через сообщения.

С Службой Поддержки: Запрашивает помощь и предоставляет обратную связь.

2) Emoloyer

• Представляет работодателя, управляющего вакансиями. Может размещать вакансии, обрабатывать заявки от соискателей, просматривать профили соискателей. Также может обмениваться сообщениями с другими пользователями. Для входа в систему требуется авторизация, и, если у пользователя нет аккаунта, необходима регистрация. Подтверждение почты обязательно для завершения регистрации.

Отношения:

С Соискателем: Просматривает профили соискателей и взаимодействует через сообщения.

С Службой Поддержки: Общается по вопросам управления процессом подбора персонала.

3) Platform Admin

• Администратор платформы, обладающий функциональностью просмотра данных пользователей с их согласия, проверки подлинности пользователей, управления пользователями (включая блокировку/разблокировку) и обмена сообщениями. Для входа в систему требуется авторизация.

Отношения:

С Job Search Engine: Управляет платформой, обеспечивает целостность данных и контролирует действия пользователей.

С Соискателем/Работодателем: Проверяет подлинность пользователей и решает вопросы, связанные с пользователями.

С Службой Поддержки: Взаимодействует в административных вопросах.

4) Support Service

• Представляет службу технической поддержки. Обрабатывает обращения от пользователей, включая решение проблем, поддерживает историю обращений. Является потомком роли "Platform Admin".

Отношения:

С Соискателем/Работодателем: Помогает в решении проблем и ведет историю взаимодействия.

С Администратором Платформы: Общается по техническим вопросам.

5) Database Admin

• Администратор баз данных, управляющий структурой данных, оптимизацией производительности и созданием резервных копий данных.

Отношения:

С Job Search Engine: Обеспечивает работу базы данных в поддержку функционала платформы.

С Ботом: Предоставляет данные для статистического анализа.

6) Bot

• Искусственный интеллект, выполняющий функции ведения статистики запросов пользователей и популярных вакансий, предоставляющий помощь и советы через искусственный интеллект, а также оповещающий о новых вакансиях по желанию соискателя. Является потомком роли "Database" для получения базы запросов и вакансий для статистики.

Отношения:

С Соискателем: Помогает с запросами и предоставляет советы.

С Базой Данных: Собирает данные для статистического анализа.

7) Database

• Система, ответственная за автоматическое сохранение данных пользователей и информации о запросах.

Отношения:

С Job Search Engine: Сохраняет и извлекает данные пользователей и запросы.

С Администратором Базы Данных: Управляется и оптимизируется Администратором Базы Данных.

С Ботом: Предоставляет данные для статистического анализа.

EXERCISE – 1 Actors, Individuals, and the Job Search Domain

Акторы, связанные с системой поиска работы:

• Соискатель (JobSeeker)

• Работодатель (Employer)

• Администратор системы (Platform Admin)

Для каждой из этих ролей рассмотрим следующие взаимодействия с системой поиска работы:

• Соискатель работы ищет вакансии.

• Соискатель работы подает заявление на работу.

• Работодатель размещает вакансии.

• Работодатель просматривает заявления на работу.

• Администратор платформы управляет учетными записями.

• Администратор платформы контролирует безопасность системы.

Потенциальные конфликты и их отражение в требованиях к ПО:

1) Пол как работодатель и соискатель:

• Конфликт: Пол может получить доступ к конфиденциальной информации о других соискателях или неправомерно повлиять на процесс рассмотрения заявлений.

• Решение: Ввести разделение ролей в системе с четкими правилами доступа и действий, доступных для каждой роли. Также необходим механизм аудита действий пользователей для обеспечения прозрачности процессов.

2) Работодатель использует систему для поиска информации о конкурентах:

• Конфликт: Работодатель может использовать JSE не только для поиска кандидатов на вакансии, но и для сбора информации о вакансиях конкурентов, что может привести к неэтичному использованию платформы.

• Решение: В требованиях к системе необходимо установить политику использования, которая запрещает использование информации о вакансиях для не предназначенных целей. Можно также ограничить видимость некоторой информации о вакансиях для работодателей, не давая доступ к деталям вакансий конкурентов.

3) Соискатель использует систему для массовой рассылки нецелевых заявлений:

• Конфликт: Соискатель может использовать JSE для массовой отправки заявлений на вакансии, не соответствуя критериям этих вакансий, что приводит к перегрузке работодателей неактуальными кандидатурами.

• Решение: Разработать механизм фильтрации и верификации заявлений перед их отправкой работодателю, чтобы убедиться, что соискатели соответствуют ключевым требованиям вакансии. Ввести систему предупреждений для соискателей, злоупотребляющих функционалом массовой рассылки, и, в крайних случаях, ограничить им возможность подачи заявлений.

4) Соискатель получает предложения от нескольких работодателей:

• Конфликт: Соискатель может чувствовать себя перегруженным при получении множества предложений от работодателей, особенно если эти предложения не полностью соответствуют его интересам или квалификации.

• Решение: Разработать систему рекомендаций, которая помогает соискателям фильтровать предложения на основе их предпочтений, квалификации и карьерных целей. Предоставить инструменты для управления предложениями, позволяющие легко отклонять неинтересующие вакансии.

EXERCISE – 2 Context Diagram

In short

Decomposition (подробнее о процессах)

EXERCISE – 3 Major JSE decisions

1. Пересмотрите свои высокоприоритетные риски

• Недостаточная защита данных пользователей.

• Недостаточная масштабируемость системы при увеличении числа пользователей.

• Сложность интеграции с внешними сервисами для проверки резюме и профилей.

2. Архитектурные решения

a) Решение: Внедрение слоев безопасности и шифрования.

• Риск: Несанкционированный доступ к данным пользователей и их утечка.

• Отражение в архитектуре: Реализация многоуровневой архитектуры с защищенными каналами связи, включая использование HTTPS и шифрование баз данных.

• Отклоненные альтернативы: Использование только базового парольного доступа без двухфакторной аутентификации.

• Причина отклонения: Недостаточное уровень безопасности и уязвимость для атак.

b) Решение: Применение микро сервисной архитектуры.

• Риск: Проблемы с масштабируемостью при увеличение числа пользователей.

• Отражение в архитектуре: Разработка системы с использованием набора микросервисов, каждый из которых отвечает за определенную функцию, обеспечивая тем самым легкость масштабирования и обслуживания.

• Отклоненные альтернативы: Монолитная архитектура.

• Причина отклонения: Монолитные системы сложнее масштабировать и обновлять, особенно при быстром росте пользовательской базы.

c) Решение: API-центрическая интеграция с внешними сервисами.

• Риск: Ограниченная функциональность и проблемы интеграции с внешними платформами.

• Отражение в архитектуре: Создание мощного API для интеграции с внешними сервисами, что позволяет легко добавлять новые сервисы и функции.

• Отклоненные альтернативы: Разовые, жестко закодированные интеграции.

• Причина отклонения: Негибкий подход, который ограничивает будущее расширение и обновление системы.

3. Требования качества:

• Безопасность: Реализация комплексных мер безопасности, включая регулярные аудиты безопасности и обновления системы.

• Надежность: Внедрение отказоустойчивых механизмов и резервного копирования данных.

• Производительность: Оптимизация процессов и инфраструктуры для обеспечения высокой производительности при большом числе запросов.

4. Документирование решений:

a. Решение: Внедрение механизмов обеспечения конфиденциальности и безопасности персональных данных.

b. Адресованные качественные аспекты: Защита персональных данных пользователей, соответствие требованиям GDPR и обеспечение высокого уровня доверия со стороны пользователей.

c. Как решение отражено в архитектуре: Внедрены методы шифрования данных, усилены механизмы аутентификации и авторизации, введены процессы для управления доступом и контроля за использованием персональных данных.

d. Отклоненные альтернативы: Не обеспечивать соответствие GDPR для обеспечения конфиденциальности и безопасности персональных данных.

e. Причина отклонения: Несоблюдение GDPR могло бы создать риск нарушения законодательства и ущерба для пользователей и системы JSE.

JSE Team

Activity Diagram

Инициация действия

• Ввод данных: Пользователь начинает взаимодействие с системой, вводя свои личные данные.

Выбор роли

• Пользователь выбирает роль, под которой будет продолжать работу в системе: Работодатель или Соискатель.

Работодатель

1. Размещение вакансий: Работодатель размещает информацию о вакансии в системе.

2. Просмотр заявлений: Работодатель просматривает заявления от соискателей, поступившие на размещенные вакансии.

Соискатель

1. Поиск вакансий: Соискатель ищет вакансии, используя различные критерии поиска.

2. Подача заявления на работу: Соискатель подает заявление на выбранные вакансии.

3.

Общие процессы для Работодателя и Соискателя

1. Направление приглашения/отказа: В зависимости от решения Работодателя, Соискатель получает приглашение на интервью или уведомление об отказе.

2. Выбор кандидата и отправка оферта на себе: Работодатель выбирает кандидатов и отправляет им предложения о работе.

3. Отправка запроса работодателю: Соискатель может инициировать обратную связь с Работодателем.

4. Рассмотрение заявлений соискателей: Работодатель рассматривает заявления от Соискателей.

Система

• Авторизация в аккаунт/Регистрация: Пользователь авторизуется в системе или проходит процесс регистрации, если у него нет аккаунта.

• Подтверждение почты: Является обязательным шагом для завершения регистрации и возможности дальнейшей работы в системе.

• Список вакансий: Система предоставляет Работодателям список вакансий, который может быть использован для управления текущими вакансиями.

• База заявлений: Система хранит все заявления от Соискателей в базе данных для доступа Работодателями.

Знак завершения узла: Этот символ означает завершение определенного потока процесса в диаграмме.

Sequence Diagram

Элементы диаграммы:

1. Пользователи:

• Соискатель

• Работодатель

2. Компоненты системы:

• platform frontend

• platform backend

• Database

Процесс регистрации:

1. Пользователи начинают регистрацию, выбирая свою роль в системе.

2. Данные пользователя отправляются на фронтенд.

Обработка фронтендом:

1. Фронтенд валидирует введенные данные и передает их на бэкенд.

2. Бэкенд передает запрос в базу данных на добавление нового пользователя.

Авторизация:

1. Пользователи вводят данные для входа в систему.

2. Фронтенд валидирует данные и передает их бэкенду.

3. Бэкенд запрашивает данные пользователя из базы данных.

4. База данных возвращает результаты бэкенду.

5. Пользователи получают сообщение о результатах операции (например, об успешной авторизации).

Создание заявления/вакансии:

1. После авторизации пользователи могут создать заявление на работу или вакансию.

2. Данные отправляются на фронтенд и валидируются.

3. Валидированные данные отправляются бэкендом в базу данных.

4. Пользователи получают уведомление о результатах операции, такие как успешное создание заявления или вакансии.

Диаграмма классов

Классы и Отношения:

Класс: Userы

Описание:

Базовый класс для всех пользователей системы.

Атрибуты:

UserID: Уникальный идентификатор пользователя.

Username: Имя пользователя, используемое для входа.

Email: Адрес электронной почты пользователя.

Password: Зашифрованный пароль пользователя.

Role: Роль пользователя в системе (например, JobSeeker, Employer).

Методы:

Register(): Регистрирует нового пользователя.

Login(): Проверяет учетные данные и осуществляет вход в систему.

Logout(): Выход пользователя из системы.

Класс: Admin (наследует User)

Описание:

Администратор системы, имеет расширенные привилегии для управления системой.

Атрибуты:

AdminID: Уникальный идентификатор администратора.

Name: Имя администратора.

Email: Адрес электронной почты администратора.

AccessLevel: Уровень доступа администратора.

Методы:

ViewLogs(): Просмотр логов системы.

ManageUsers(UserID): Управление пользователями.

Класс: JobSeeker (наследует User)

Описание:

Пользователь, ищущий работу в системе.

Атрибуты:

Resume: Резюме соискателя.

SearchHistory: История поиска вакансий.

AppliedJobs: Список вакансий, на которые пользователь подал заявление.

Методы:

SearchJob(): Поиск вакансий.

ApplyForJob(JobID): Подача заявления на вакансию.

UpdateResume(): Обновление резюме.

Класс: Employer (наследует User)

Описание:

Работодатель или представитель компании, размещающий вакансии.

Атрибуты:

CompanyProfile: Профиль компании работодателя.

PostedJobs: Список размещенных вакансий.

Методы:

PostJob(): Размещение новой вакансии.

ReviewApplications(JobID): Просмотр заявлений на вакансии.

SendInvitation(JobSeekerID): Отправка приглашения соискателю.

Класс: JobPosting

Описание:

Информация о вакансии, размещенной работодателем.

Атрибуты:

JobID: Уникальный идентификатор вакансии.

EmployerID: Идентификатор работодателя, разместившего вакансию.

Title: Название вакансии.

Description: Описание вакансии.

Requirements: Требования к соискателям.

ApplicationDeadline: Срок подачи заявлений на вакансию.

Методы:

Create(): Создание новой вакансии.

Update(): Обновление информации о вакансии.

Delete(): Удаление вакансии.

Класс: Application

Описание:

Заявление на вакансию от соискателя, содержащее информацию о его желании работать на определённой должности.

Атрибуты:

ApplicationID: - Уникальный идентификатор заявления.

JobID: - Идентификатор вакансии, на которую подается заявление.

JobSeekerID: - Идентификатор соискателя, подавшего заявление.

SubmissionDate: - Дата подачи заявления.

Status: - Статус заявления (например, 'принято', 'рассматривается', 'отклонено').

Методы:

Submit(): - Отправка заявления на рассмотрение работодателем.

Withdraw(): - Отзыв заявления соискателем.

UpdateStatus(Status: String): - Обновление статуса заявления.

Диаграмма компонентов