МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»

(БГТУ им. В.Г. Шухова)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ

ОТЧЁТ

о прохождении производственной технологической (проектно-технологической) практики

Руководитель практики от организаци: руководитель направления разработки Дьяков Александр Михайлович

Руководитель практики от кафедры: доцент Твердохлебов Виталий Викторович

Студент группы ПВ-223 Пахомов Владислав Андреевич

Белгород 2025 г.

Оглавление

1 Общая характеристика организации

Компания «Антара» является экспертом в области тестирования ПО и заказной разработки. Она предлагает широкий спектр услуг, включая тестирование программного обеспечения, разработку банковских приложений и внедрение инновационных технологий в области тестирования ПО.

«Антара» была основана в 2019 году и с тех пор стала одной из самых динамически развивающихся компаний на рынке IT-услуг в России и уже успела завоевать доверие своих клиентов благодаря высокому качеству предоставляемых услуг.

В своей работе компания использует современные методы и технологии, что позволяет ей создавать высококачественное программное обеспечение, которое успешно применяется в различных сферах бизнеса.

Партнёры и клиенты компании включают: Сбербанк, ВТБ, ХКФ Банк, Банк Открытие, X5 Retail Group, ГлобалЛаб, Мосбиржа, ЛАНИТ, Синимекс, ОТП Банк, МКБ.

Главный офис компании находится в Москве, однако есть и множество филиалов в других городах, например, в Нижнем Новгороде.

2 Анализ

В качестве индивидуального задания была выбрана разработка программного обеспечения для обеспечения внутренней работы внутри компании.

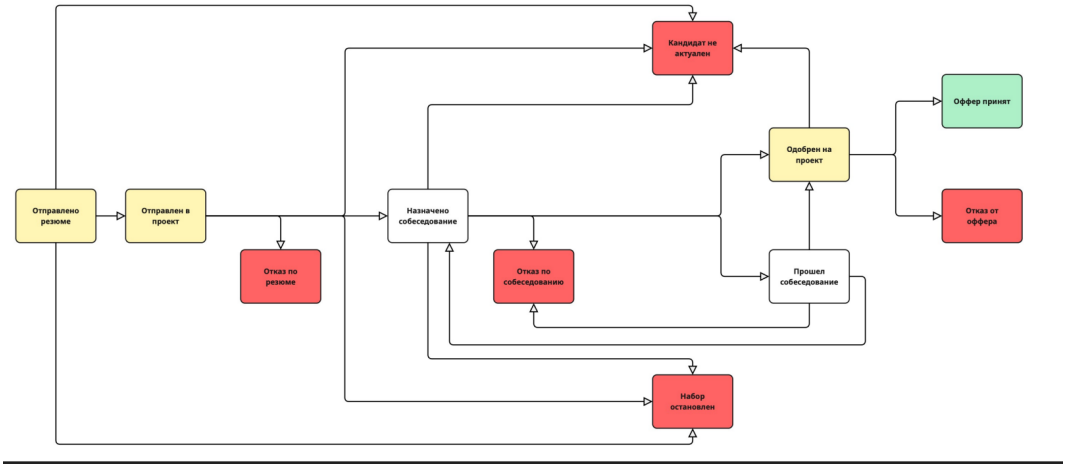
2.1 Предметная область

HRM (Human Resource Management) - специализированное программное обеспечение, которое помогает в управлении персоналом HR-специалистам. В задачи такого ПО входят отслеживание ”пути”потенциального кадра от получения отклика на сервисах подбора до его принятия на место работы, составлении инфографики и автоматизации процессов, что позволяет HR-специалистам более эффективно организовывать свою работу.

Интеграции с другими сервисами позволяет HRM стать по-настоящему гибким инструментом, агрегируя в одном месте информацию из разрозненных источников, что позволяет специалисту иметь более полную и целостную картину.

В текущей информационной системе рассмотрим несколько сущностей. Ресурс ”кандидат”описывает самого рассматриваемого кандидата. Содержит общую информацию о кандидате, его файлы, ФИО, информация о работе, желаемой ЗП и т. д. Ресурс ”вакансия”описывает в себе вакансию, которую предоставляет проект. Она содержит в себе описание, ЗП, тип кандидата и его квалификация (например, нужен синьор-бекендер), ответственного и статус. Связующим звеном между кандидатом и вакансией является ”отправка”. Отправка содержит в себе информацию о кандидате, текущий статус отправки, комментарий, дату начала, вакансию и другие поля. Изменить статус заявки можно при помощи ”событий”. События описывают жизненный цикл отправки. Они могут включать в себя отправку резюме, приглашение на проект и т. д. В целом эта сущность нужна для отслеживания присвоенного статуса отправки и даты присвоения. Между статусами настроены переходы. Так, отправленный на созвон кандидат не может внезапно откатиться назад и быть на этапе рассмотрения резюме. Есть и терминальные статусы - это статусы, в которые

никуда нельзя перейти. Например, кандидат самовольно решил отказаться от позиции.



2.2 Информационная система и её особенности

Программное обеспечение представляет из себя клиент-серверное приложение.

Серверная часть написана с использованием FastAPI, позволяющим быстро развернуть WEB-сервер. В качестве ORM была выбрана SQLAlchemy, предоставляющая гибкий и одновременно лёгкий подход к формированию запросов в базу данных. Система контроля миграций базы данных - Alembic. Для валидации входных и выходных данных используется Pydantic. Приложение контейнеризируется при помощи Docker Compose, в котором находится база данных и, собственно, сам сервер. Сервер содержит в себе фейкер - небольшой скрипт, который позволяет добавить демонстрационные данные для локального тестирования. **Очень** удобно.

В качестве клиентской части за основу была взята библиотека React Admin. Эта библиотека позволяет удобно контролировать ресурсы и содержит очень много готовых компонент, готовых к использованию. Она же может задавать и стили. Для использования более широких возможностей доступна подписка Enterprise. Библиотека позволяет быстро и эффективно

развернуть работу с API, однако имеет ограничения. Так каждый ответ сервера должен иметь строгий формат, например полученный ресурс обязан включать в себя идентификатор - поле ID.

CI/CD включает в себя Pre-Commit-Hooks - действия, выполняемые перед тем, как отправить код в репозиторий Git. Они позволяют держать код в едином стиле, а также позволяют избежать простых ошибок. Система автоматической отправки собранных образов на сервер включает с помощью Github Actions и выделенного раннера позволяет собрать продукт, создать образ докер, отправит его в репозиторий образов, после чего можно будет запустить полученный образ на удалённом сервере. Github Actions настроены на автоматический деплой при назначении тега версии для коммита.

До недавних пор приложение являлось монолитным, то есть клиентская и серверная часть находились в одном репозитории. Однако из-за растущего функционала было принято решение разделить монолитный репозиторий на два отдельных - для клиента и сервера. Разделение кодовой базы позволило команде из нескольких человек работать над своими задачами более эффективно.

3 Задачи практики и результаты их выполнения

Отслеживание статуса задач выполнялись при помощи PM, развёрнутого на собственном сервере компании. В нём руководитель разработки и участники могут создавать задачи, назначать ответственного за задачи и отслеживать их прогресс при помощи статуса.

3.1 Создание сквозных статусов для отправок

Данная задача нужна была для введения автоматического опциональнгого присвоения всем привязанным отправкам соответствующего статуса при закрытии вакансии.

Для реализации задачи необходимо также было ввести понятие свкозного статуса. Если из терминального статуса попасть никуда нельзя, то из сквозного статуса можно попасть в любой статус, а из любого статуса можно попасть в сквозной.

На стороне сервера был ранее реализован модуль, который определяет список статусов, какой из статусов доступен для перехода и куда и так далее. Именно поэтому этот модуль и было решено дополнить сквозными статусами:

```
...
PASS_THROUGH_TRANSITIONS = {
    SubmitStatus.VACANCY_CLOSED,
}

@classmethod
def transitions(cls, for_status: SubmitStatus) -> list[SubmitStatus]:
    if for_status in cls.PASS_THROUGH_TRANSITIONS:
        available_transitions = SubmitStatus
    else:
        available_transitions = cls.TRANSITIONS.get(for_status, set())
        available_transitions = chain(
            available_transitions, cls.PASS_THROUGH_TRANSITIONS
        )

    return list(available_transitions)

@classmethod
def confirm(cls, new_status: SubmitStatus, old_status: SubmitStatus) -> bool:
    return new_status in cls.transitions(for_status=old_status)
...
```

После чего в методе для изменения вакансии мы добавили проверку. Если статус сменяется на закрытый и пользователь дополнительно утверждает, что хочет сменить статус, то мы будем добавлять всем связанным отправкам новый статус.

На фронтенде было принято решение сделать этот функционал в виде модального окна, всплывающего при редактировании вакансии.