1 ОБЗОР ЛИТЕРАТУРЫ

1.1 Обзор аналогов

В данном разделе рассматриваются исследования предметных областей, которые затрагиваются в разрабатываемом проекте. Под предметными областями подразумевается используемые методы для создания, а также инструменты и подходы к проектированию.

В качестве вводной части дается рассмотреть используемые методы и технологии, которые применяются в дипломном проекте. В данном подразделе рассматривается и проводится анализ существующих аналогов, для разработки плана действий и облегчения поиска необходимых материалов. Необходимость состоит в предотвращении ошибок в проектировании и поиске наиболее оптимальных решений.

1.1.1 Sage HR

Sage HR [1] (см. рисунок 1.1) – это компания по разработке программного обеспечения (ПО) для управления персоналом. Приложение предназначено как для малых, так и для средний компаний. Управление происходит через веб-сайт и мобильное приложение, каждый пользователь имеет доступ к своему расписанию.

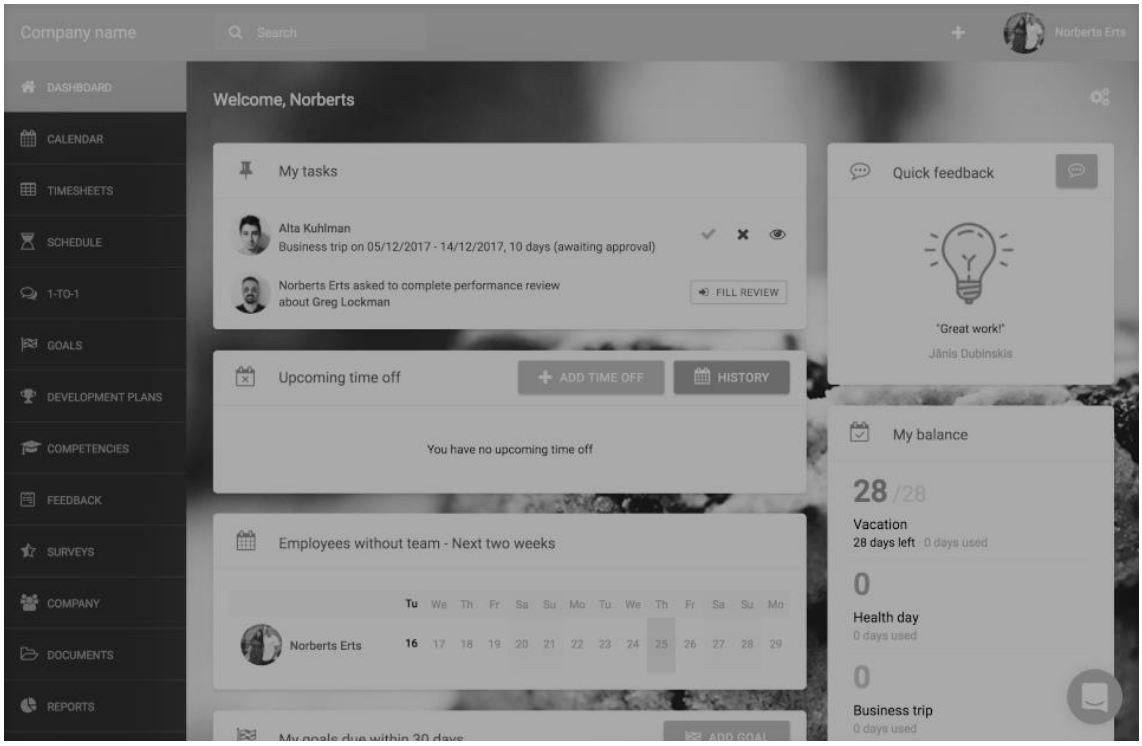

Рисунок 1.1 – Главная страница Sage HR [1]

На главной странице приложения представлены основные функциональные элементы, необходимые для работы пользователя. Дизайн

интерфейса привлекателен и выдержан в ярких тонах, однако, избыточное использование насыщенных цветов может вызвать утомление глаз при продолжительном использовании приложения.

Отсутствие русской локализации хоть и не является минусом, но все же это небольшой недостаток в силу того, что в университете много людей, которым предпочтительнее использовать русский.

При подключении данного приложения есть возможность выбирать только необходимые функции. Это значит, что программное средство модульное и каждая из его функциональных частей работает независимо от другой. Это достоинство, которое стоит иметь ввиду, потому что не весь функционал необходим сразу и возможность подключать по надобности достаточно удобная и практичная.

В данном приложении сотрудник должен самостоятельно регистрироваться. С одной стороны это плюс, так как может сэкономить время отделу кадров, но с другой, если возникнут ошибки при заполнении, возникнут разногласия в данных. В случае данного дипломного проекта эта функция возлагается на отдел кадров и администраторов, которые создают учетную запись работнику при его устройстве.

1.1.2 WebHR

WebHR [2] (см. рисунок 1.2) – это онлайн-инструмент для управления персоналом, который охватывает все аспекты работы отдела кадров, от приема на работу до выхода на пенсию. WebHR упрощает HR-процессы за счет шаблонизации, автоматизации и интеграции HR-процессов, чтобы к ним можно было получить доступ и обработать их с единой панели управления.

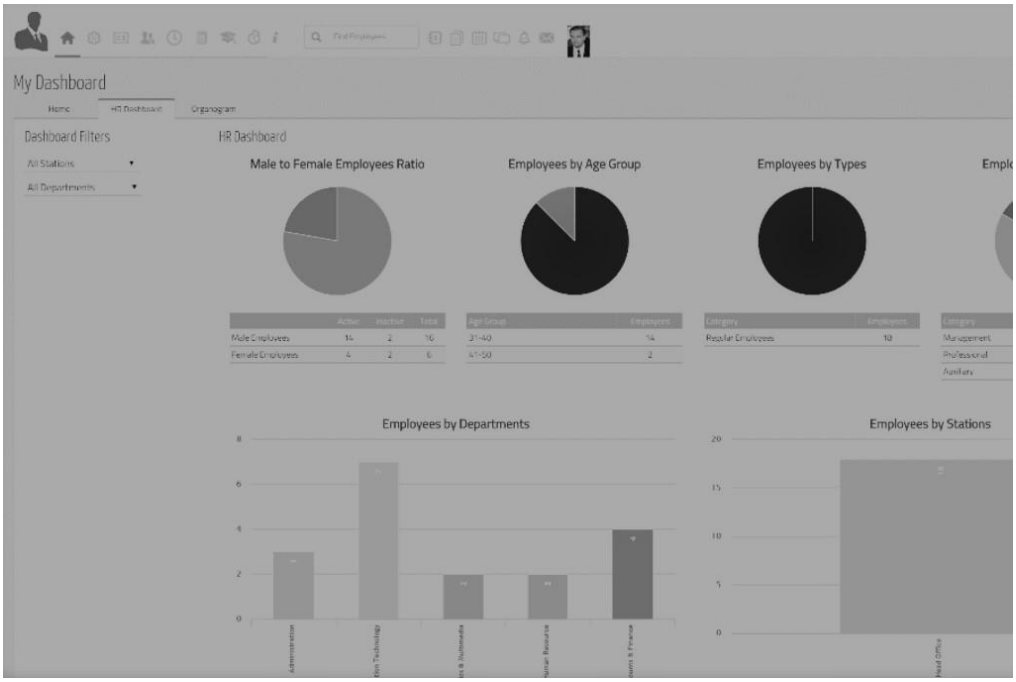

Рисунок 1.2 – Страница WebHR [2]

Одной из возможностей WebHR является интеграция с сайтом поиска работы Indeed.com, которая позволяет автоматически загружать объявления о вакансиях на сайт. Подобная функция удобная, но стоит учесть, что в нашем регионе используется другой сайт для поиска кадров. Стоит принять ее во внимание и учесть в дальнейшей разработке и развитии приложения.

За расчет заработной платы отвечает внешний подрядчик, однако все функции для расчета заработной платы также присутствуют в приложении. Тем не менее, следует отметить, что это приложение ориентировано преимущественно на работу с отделом кадров, поэтому это может стать недостатком в контексте необходимости обеспечения эффективного обмена информацией между разными подразделениями и, соответственно, самом использовании разными отделами. Одна из целей данного дипломного проекта заключается в обеспечении коммуникации между всеми отделами, так как разные отделы находятся в разных корпусах, поэтому ориентирование вокруг одного специализированного отдела не соответствует запросам данного веб-приложения.

Следует отметить, что наличие только англоязычной версии является недостатком приложения. Внедрение русской локализации в приложении может значительно улучшить его удобство и доступность для пользователей.

С точки зрения визуального оформления, это приложение выглядит более привлекательно, чем первый вариант, благодаря использованию простых и неярких цветов, которые не отвлекают внимание и позволяют ясно видеть все необходимые разделы и использования минималистичного дизайна.

Значительным же недостатком данного приложения является ограниченная поддержка. В БГУИР находится множество сотрудников и в случае возникновения сбоев в работе приложения, это может стать источником коллизий в рабочем процессе.

1.2 Обзор технологий

Дипломный проект предполагает разработку системы, которая является веб-приложением, это значит, что это ПО запускается в веб-браузере и оно имеет клиент-серверную архитектуру. Логика такого приложения разделена между сервером и клиентом, а данные, как правило, хранятся на сервере. Обмен информацией между сервером и клиентом происходит через сеть Интернет. Одним из основных преимуществ такого подхода является возможность запуска приложения на разных операционных системах, поскольку клиенты не зависят от конкретной ОС пользователя. В результате веб-приложения могут быть использованы на разных платформах, что делает их универсальными [3].

Система является небольшой и создается одним человеком, поэтому она имеет монолитную архитектуру. Это означает, что различные компоненты, а именно бизнес-логика, слой доступа к данным, он же подключение к базе

данных, интерфейс пользователя и так далее, находятся внутри одного процесса. Даная архитектура проста в развертывании, масштабировании и, соответственно, тестировании.

Паттерн проектирования, использованный в веб-приложении – Model, View, Controller (MVC). Приложение разделено на три условные части:

- модель, то есть данные, которые будут передаваться между представлениями и контроллерами;
- представления, которые будут визуализировать данные модели для пользовательского интерфейса;
- контроллеры, которые будут обрабатывать запросы и выбирать соответствующие им представления для визуализации [6].

Язык программирования для серверной части – Java, библиотека Spring. Этот фреймворк содержит много разных модулей. Для клиентской части – Java Script, библиотека React. База данных, для содержания необходимых данных – PostgreSQL.

1.2.1 Клиент-серверная архитектура

Клиент-серверная архитектура (см. рисунок 1.3) – это модель взаимодействия между компьютерами в сети, при которой один компьютер (сервер) предоставляет определенные ресурсы или услуги, а другие компьютеры (клиенты) запрашивают эти ресурсы или услуги через сеть.

В клиент-серверной архитектуре обычно выделяют два типа компонентов: клиентские и серверные.

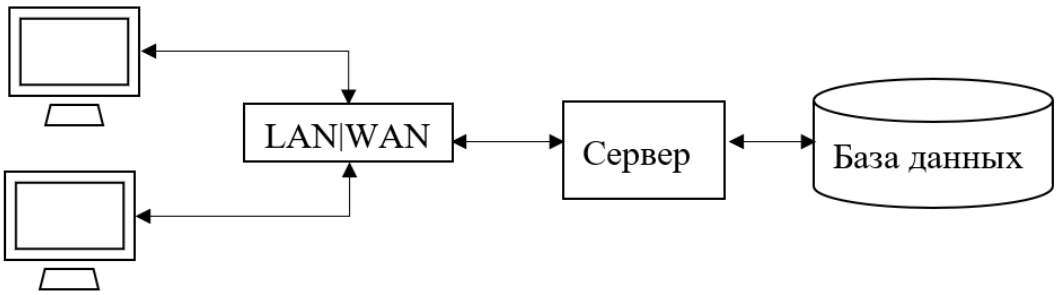

Рисунок 1.3 – Клиент-серверная архитектура

Клиенты – это устройства, на которых запускается пользовательский интерфейс, которые запрашивают ресурсы или функции у серверов. Конечный пользователь, который использует приложение, и обычно оно работает на его компьютере или мобильном устройстве.

Серверы – это высокопроизводительные вычислительные устройства, на которых размещаются ресурсы и функции, которые клиенты запрашивают. Серверы могут выполнять различные задачи, например, обрабатывать запросы

клиентов, хранить и обрабатывать данные, осуществлять авторизацию и аутентификацию пользователей и так далее.

Взаимодействие между клиентами и серверами происходит через сеть, обычно с использованием протоколов передачи данных, таких как HTTP, FTP, SMTP и тому подобные. Клиентские компоненты отправляют запросы на сервер, и серверные компоненты отвечают на эти запросы, предоставляя запрошенные данные или услуги.

Клиент-серверная архитектура является широко распространенной и используется во многих областях, таких как веб-приложения, базы данных, игровые системы и т.д. Архитектура позволяет создавать масштабируемые и гибкие системы, которые могут обрабатывать большие объемы запросов и предоставлять доступ к данным и услугам из любой точки сети.

В разрабатываемой системе присутствует еще один компонент – база данных, программа, в которой хранятся все данные приложения. Благодаря ей данная архитектура является трехуровневой, потому что она состоит из трех компонентов.

В клиент-серверной архитектуре сервер играет роль не только компьютера, на котором работает приложение или сайт, но также является хранилищем всех данных. Клиенты не имеют прямого доступа к базе данных, что защищает их личную информацию и обеспечивает приватность других пользователей, например, в социальных сетях.

Клиенты обращаются к серверу за информацией, и если сервер считает, что пользователь имеет права на получение этой информации, то он ее предоставляет. Подобный подход гарантирует защиту личных данных других пользователей [4].

Трехуровневая архитектура информационных систем может быть улучшена путем добавления дополнительных серверов и преобразована в многоуровневую архитектуру. Такая виртуальная архитектура позволяет значительно увеличить эффективность работы информационных систем и оптимизировать использование программно-аппаратных ресурсов [5].

1.2.2 MVC

Шаблон MVC (см. рисунок 1.4) [6] – это шаблон проектирования веб-приложений, который включает в себя три отдельных компонента: модель данных приложения, пользовательский интерфейс и логику взаимодействия пользователя с системой. Подобный подход позволяет минимизировать воздействие на другие компоненты при модификации одного из них. Кроме того, MVC включает несколько мелких шаблонов, что делает его более гибким и удобным для использования.

Основной целью использования шаблона проектирования MVC является разделение бизнес-логики и данных от визуализации. Такое разделение упрощает повторное использование кода и облегчает сопровождение. Например, если необходимо добавить новое представление

данных, такое как JSON, XML, PDF или XLSX, к уже существующему маршруту, это можно сделать без изменений в бизнес-логике маршрута. А изменения визуализации не затрагивают бизнес-логику, а изменения бизнес-логики не влияют на визуализацию.

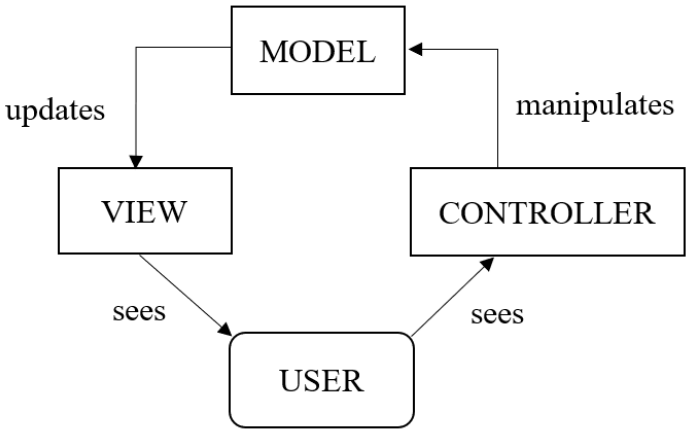

Рисунок 1.4 – Шаблон MVC [6]

Изменение каждого из модулей происходит независимо. Все модули имеют разные обязанности и ответственности и все поведение будет направлено на выполнение одной задачи. Подобный подход соответствует (single responsibility, open–closed, Liskov substitution, interface segregation и dependency inversion) SOLID -принципу единой ответственности [7].

1.2.3 База данных PostgreSQL

Для проекта была выбрана система управления базами данных PostrgreSQL.

PostgreSQL – это популярная свободная объектно-реляционная система управления базами данных (СУБД). Система базируется на языке SQL и поддерживает многочисленные возможности [8].

Преимущества данной СУБД:

- поддержка баз данных неограниченного размера;
- мощные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования и поддержка загрузки С-совместимых модулей;
- наследование;
- легкая расширяемость.

СУБД обеспечивает высокую надежность и эффективность работы, включая поддержку транзакций (ACID) и встроенные механизмы репликации. Кроме того, система расширяема, что позволяет создавать собственные типы данных и индексы, а также расширять ее функциональность с помощью языков программирования.

Строительство системы на основе PostgreSQL обеспечивает более гладкую интеграцию с другими внутренними системами БГУИР, что является преимуществом для разработки данной системы.

1.2.4 Spring Framework Java

Spring Framework [9]– это один из самых популярных фреймворков для разработки приложений на языке Java, он универсальный, с открытым кодом. Предоставляет комплексный набор инструментов и библиотек для упрощения создания сложных приложений. В какой-то степени его можно назвать фреймворком фреймворков, потому что он состоит из отдельный модулей. В данном веб-приложении используются такие модули как Spring Data, Spring Security, Hibernate, Spring MVC, JPA.

Spring позволяет создавать RESRful веб-сервисы. Ниже эти модули рассмотрены подробнее.

REST (от англ. Representational State Transfer — «передача состояния представления») – это общие принципы организации взаимодействия приложения/сайта с сервером посредством протокола (HyperText Transfer Protocol) HTTP. По сути, взаимодействие с сервером происходит в четыре операции: 1) получение данных с сервера, 2) добавление на сервер новых данных, 3) изменение уже существующих на сервере данных и 4) удаление данных.

Основные принципы такой организации являются:

1. Ресурсы (Resources), каждый из которых должен обладать своим уникальным идентификатором URI (Uniform Resource Identifier), который клиент может использовать для доступа к этому ресурсу. Ресурсы могут представлять собой данные, которые могут быть получены или как-то изменены с помощью HTTP-методов.

2. Представления (Representation), когда каждый ресурс может иметь несколько представлений, которые определяют формат и содержание данных, возвращаемых сервером в ответ на запрос клиента. Представления могут быть в таких форматах как HTML, XML, JSON или любом другом формате, который может быть прочитан клиентом.

3. Методы HTTP (HTTP Methods), для работы с ресурсами, доступны четыре основных метода HTTP: GET для получения данных, POST для создания новых ресурсов, PUT для изменения уже существующих ресурсов и DELETE для удаления ресурсов.

4. Без состояния (Stateless), каждый запрос, отправленный клиентом на сервер, содержит все необходимые данные для его обработки. Сервер не запоминает информацию о предыдущих запросах клиента, что делает реализацию и масштабирование сервера более простым.

Spring Data – это набор инструментов и библиотек, предоставляющих единую модель программирования для доступа к данным. Набор использует

Spring-подход для упрощения работы с различными типами баз данных, включая реляционные, нереляционные и облачные базы данных.

Основной концепцией является репозиторий, который представляет собой набор интерфейсов для взаимодействия с JPA Entity. Например, интерфейс `CrudRepository<T, ID extends Serializable>`, который расширяет `Repository<T, ID>`, обеспечивает базовую функциональность для создания, чтения, обновления и удаления данных, иначе говоря, обеспечивают CRUD (create, read, update, delete).

Есть абстракции типа `PagingAndSortingRepository`, которая предоставляет методы для разбиения на страницы и сортировки записей. И еще одна `JpaRepository` предоставляет некоторые связанные с JPA методы, такие как очистка контекста постоянства и удаление записей в пакете.

Spring Security – это фреймворк безопасности, который предназначен для использования в приложениях, построенных на платформе Spring. Фреймворк предоставляет мощные функции аутентификации и авторизации, которые можно использовать для обеспечения безопасности веб-приложений, микросервисов, REST-сервисов и других приложений на основе Spring.

Основным компонентом Spring Security является фильтр безопасности, который работает на уровне HTTP запросов и ответов. Фильтр может быть настроен для выполнения различных задач, таких как проверка учетных данных пользователя, проверка разрешений доступа к ресурсам, фильтрация запросов и другое. Фильтр обеспечивает безопасность, путем применения цепочки фильтров к каждому запросу, что позволяет выполнять настройку защиты для каждого запроса.

Hibernate – это фреймворк для объектно-реляционного отображения в Java, который упрощает доступ к базам данных, позволяя разработчикам работать с объектами, а не с SQL-запросами. Hibernate позволяет сопоставлять объекты Java с таблицами в базе данных и автоматически генерировать соответствующий SQL-код, обеспечивает механизмы для управления отношениями между объектами, транзакциями, кэшированием данных и другими функциями, необходимыми при работе с базами данных.

Преимущества Hibernate включают улучшенную производительность, упрощенное программирование и снижение количества ошибок, связанных с неправильным использованием SQL-запросов. обеспечивает переносимость кода между различными СУБД и улучшенную безопасность при работе с базами данных.

Java Persistence API (JPA) – это спецификация Java EE для управления объектно-реляционным отображением в приложениях Java. Спецификация предоставляет стандартный способ описания сущностей, которые могут быть сохранены в базе данных, и управления их жизненным циклом в рамках приложения. JPA облегчает работу с базами данных и ORM-фреймворками, такими как Hibernate, EclipseLink, OpenJPA и другими.

JPA определяет аннотации, которые можно добавлять к классам и полям, чтобы указать отображение на реляционную базу данных. Эти аннотации включают аннотации, такие как `@Entity`, `@Table`, `@Column` и `@Id`, которые используются для описания сущностей и их свойств.

JPA также определяет набор методов для управления жизненным циклом объектов, таких как `persist()`, `merge()`, `remove()`, `refresh()` и `find()`. Поддерживает JPQL (Java Persistence Query Language), что позволяет создавать запросы к базе данных, используя объектную модель данных вместо SQL.

Обеспечивает абстракцию от конкретной базы данных, что позволяет легко переносить приложения между различными СУБД без изменения кода приложения. Это делает JPA мощным инструментом для разработки приложений, которые должны работать с различными базами данных и управлять большим объемом данных.

Spring MVC – это фреймворк для разработки веб-приложений на языке программирования Java, основанным на паттерне проектирования Model-View-Controller (MVC), описанном ранее в подразделе выше.

Фреймворк обеспечивает обработку ошибок, валидацию данных, аутентификацию и авторизацию, обработку AJAX-запросов и множество других функций, что делает его одним из наиболее популярных фреймворков для разработки веб-приложений на Java.

1.2.5 React

React [10] является JavaScript-библиотекой для создания пользовательских интерфейсов. Позволяет создавать компоненты, которые отвечают за отображение данных на странице. Компоненты могут быть многоразовыми и взаимодействовать между собой.

React использует DOM (Document Object Model или виртуальное дерево объектов), что позволяет обновлять только те элементы, которые действительно изменились, а не обновлять всю страницу целиком. Такой подход повышает производительность и скорость работы приложения.

DOM – это объектная модель документа, которая представляет собой иерархическую структуру документа в виде дерева объектов. Виртуальное DOM дерево состоит из узлов, которые могут быть элементами, атрибутами, текстом, комментариями и др. Узлы связаны друг с другом иерархически в родительские (parentNode) и дочерние (childNodes) отношения, а также соседние узлы (previousSibling и nextSibling). Позволяет программно создавать, изменять и удалять элементы и их атрибуты, а также реагировать на события, такие как щелчки мыши, изменения размеров и другие. DOM API может быть использован в JavaScript, чтобы создавать интерактивные и динамические веб-страницы.

Это не браузерная технология, это стандарт, определяемый World Wide Web Consortium (W3C). Браузеры предоставляют DOM API, который можно

использовать для манипулирования содержимым документа. DOM API позволяет получать доступ к элементам документа, изменять их содержимое, атрибуты и стили, добавлять и удалять элементы, а также реагировать на события.

Одним из преимуществ React является большое количество готовых компонентов и библиотек, которые можно использовать в своих проектах. React также хорошо подходит для создания больших приложений, которые могут быть разбиты на множество многоразовых компонентов.

1.3 Постановка задачи

Исходя из анализа аналогов, можно сделать вывод, что для современных программ важна простота, незагруженный и интуитивно понятный интерфейс. Программа должна быть масштабируемая, платформенно независимая, чтобы ее можно было запускать на разных системах. Также следует отметить, что должен быть соответствующий нуждам пользователей функционал.

Применение указанных технологий и подходов поможет создать качественную систему для дипломного проекта, а также обеспечит выполнение требований к проекту.

Следует добавить, что данный дипломный проект создавался для учреждения, в котором уже существуют свои внутренние ресурсы и данная система должна не только соответствовать требованиям современного приложения, а также иметь возможность легко интегрироваться в имеющуюся среду. Соответственно, некоторыми из пунктов выполнения данного требования являлось использование СУБД PostgreSQL и инструментов Spring.

Основные принципы, лежащие в основе данного программного комплекса – это интуитивно понятный и легкий в использовании интерфейс, расширяемый функционал, а также охват нужд сотрудников и облегчение их коммуникаций.

Для дипломного проекта были определены следующие задачи:

- разработка системы таблиц в базе данных и их взаимодействие;
- разработка бизнес-логики для серверной части;
- разработка и реализация пользовательского интерфейса.

Данный программный комплекс будет реализован в виде веб-сайта, доступного через браузер, и предоставлять следующий набор функций:

- регистрация и авторизация пользователя;
- наличие трех уровней доступа – block, user и admin;
- просмотр всех сотрудников;
- администрирование аккаунтов;
- заполнение документов;
- создание событий и заданий, назначение их на других пользователей;
- система фильтрации списка пользователей для удобного поиска;
- смена логина.