**3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

Этот раздел посвящен описанию работы и состава разрабатываемого программного продукта.

В дальнейшем представлены взаимосвязи между различными классами программного обеспечения в виде диаграммы классов ГУИР.400201.000 РР.1.

**3.1 Описание структуры приложения**

Для серверной части программного продукта были выбраны несколько технологий, включая:

- Spring Boot, которая позволяет создавать веб-сервер и настраивать взаимодействие между различными классами приложения;

- Maven, используется для настройки процесса сборки, упаковки и запуска приложения;

- Spring Web, для веб-сервисов;

- Spring Data JPA, для работы с базой данных;

- Spring Security, для обеспечения защиты, а также создания авторизации.

В данном приложении серверная часть приложения реализовала на Java. Исходя из гексагональной архитектуры, в приложении классы делятся по исполняемому функционалу. Структура пакетов приведена ниже:

1. authentication - этот модуль предназначен для тех классов, которые помогают при аутентификации, иначе говоря, реализуют авторизацию пользователя, создание аккаунта и тому подобное. В этот модуль входит еще несколько пакетов. Разделенных по функционалу:

1.1 auth – пакет с классами, реализующими авторизацию\аутентификацию пользователя. В него входят классы:

-

-

-

-

1.2 config – пакет с классами, реализующими конфигурацию Spring Secure. В него входят классы:

-

-

-

1.3 controlers

1.4 domainsService

2. models

2.1

2.2

2.3

Отдельно от всех этих моделей и пакетов находится класс EpaWebApplication в общей папке со всем вышеперечисленным с названием epadiplom.

В качестве архитектурного подхода для написания сервера приложения выбрана гексагональная архитектура. В связи с этим в приложении не выделяются привычные пакеты по общей деятельности классов, например, пакет для сервис-классов. Гексагональный архитектурный подход предлагает выделять в приложении модули, которые определяются по функционалу приложения. В данном приложении модули реализованы с помощью одноименной технологии Maven. Все модули приложения и их описание представлены в списке ниже:

1. Модуль service – содержит классы бизнес-логики архитектуры. В связи со спецификой гексагональной архитектуры классы в данном модуле лишены реализации. Реализацию должны предоставить модули «верхнего» уровня приложения. Интерфейсы данного модуля покрывают функционал поиска устройств, а также сбора информации о них.

2. Модуль domain – содержит классы приложения, которые являются основным представлением модели данных. Используются модулем service, следовательно и всеми модулями, которые используют service.

3. Модуль persistence – содержит классы данных, для представления моделей данных БД, а также набор репозиториев для работы с различными документами БД.

4. Модуль main – основной модуль приложения, в котором реализуются интерфейсы модуля service, а также необходимые для него адаптеры для модуля persistence и для веб-части приложения. Также в этом модуле представлены файлы конфигурации классов приложения, сервисы для интернационализации, файлы констант и функция main.

Клиентская часть приложения разделена на пакеты в соответствии со стилевым соглашением разделения на пакеты React и имеет следующую структуру:

1. Пакет api содержит функции для обращения к Web API сервера приложения.

2. Пакет components содержит React компоненты.

3. Пакет hooks содержит дополнительные хуки.

4. Пакет pages содержит отдельные компоненты страниц приложения.

5. Пакет store содержит файлы конфигурации Redux-хранилица приложения.

6. Пакет styles содержит стилевые конфигурации используемой в приложении библиотеки пользовательского интерфейса.

7. Пакет utils содержит вспомогательные функции приложения.

8. Файл App.js содержит главный компонент приложения, в котором отображаются все остальные компоненты приложения.

9. Файл index.js является главным скриптовым файлом приложения, который исполняется при старте приложения.

Такая организация пакетов позволяет легко добавлять новые компоненты и изменять старые. При добавлении нового функционала возможно добавить новый пакет и, таким образом, изолировать его логику от других, существующих функций.

Взаимодействие внутри описанных пакетов и вообще в приложении приведено на диаграмме классов, приведенной на чертежах ГУИР.400201.000 PP1.

**3.2 Описание модели данных**

В данном разделе рассматривается база данных, которая работает с помощью СУБД PostrgeSQL. Базу данных можно условно разбить на несколько логических блоков (см. рисунок 3.1):

- Данные сотрудника;

- Данные по должностям;

- Данные по отделам;

- Данные по таскам;

- Данные по событиям;

- Данные по заявлениям;

- Представления.

Рисунок 3.1 – Условное логическое разделение блоков БД

Сотрудник

Отделы

Таски

События

Заявления

Должности

Представления

**3.2.1 Таблица Employee**

Данная таблица предназначена для хранения основной информации о пользователе, которая можно указать в общем доступе, и которая будет отображаться в глобальном поиске сотрудников.

Поля таблицы:

- id – первичный ключ, bigint;

- first\_name – имя сотрудника, varchar (128);

- middle\_name – отчество сотрудника, varchar (128);

- last\_name – фамилия сотрудника, varchar (128);

- work\_number – рабочий номер сотрудника, numeric (16);

- location\_street – зашехированный пароль, который был выслан пользователю для восстановления аккаунта, varchar (128);

- cabinet\_office – время и дата отправления пароля для восстановления аккаунта, varchar(8);

- id\_dep – внешний ключ для связи с таблицей department, bigint.

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. С полем id используется специальный атрибут «null: false», это означает, что поле не может быть нулевым. Для остальных колонок это значение не выставлено, чтобы при создании аккаунта для сотрудника данные заполнялись отделом кадров и не было ошибок в системе и документах.

**3.2.2 Таблица Login**

Данная таблица предназначена для хранения информации о пользователе, которая связана с аккаунтом сотрудника, а именно содержит данные необходимые для авторизации и создания аккаунта.

Поля таблицы:

- id\_login – первичный ключ, а также внешний ключ для таблицы employee, bigint;

- login\_user – логин сотрудника, varchar (319);

- password\_user – захешированный пароль сотрудника, varchar (128);

- mail\_user – мейл сотрудника, varchar (319);

- role – роль пользователя (касаемо аккаунта, а не работы), varchar (6).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. С полем id\_login, login\_user, password\_user, mail\_user, role используется специальный атрибут «null: false», это означает, что поле не может быть нулевым. Все эти поля заполняются при создании профиля сотрудника.

**3.2.3 Таблица Personal**

Данная таблица предназначена для хранения личной информации о пользователе, которую можно будет увидеть только при наличии определенных прав, и которая требуется, в основном, отделу кадров.

Поля таблицы:

- id\_personal – первичный ключ, а также внешний ключ для таблицы employee, bigint;

- birth\_d – дата рождения сотрудника, timestamp without time zone;

- entry\_d – дата устройства на работу (в этот же день должен быть и создан аккаунт), timestamp without time zone;

- personal\_number – личный номер сотрудника, numeric (16).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. С полем id\_personal используется специальный атрибут «null: false», это означает, что поле не может быть нулевым. Для остальных колонок это значение не выставлено, чтобы при создании аккаунта для сотрудника данные заполнялись отделом кадров и не было ошибок в системе и документах. Поле entry\_d будет заполняться автоматически при создании аккаунта. Формат данных timestamp without time zone означает, что нет привязки к часовому поясу. Это сделано, чтобы не было разногласий во времени.

**3.2.4 Таблица Department**

Данная таблица нужна для содержания списка отделом университета, чтобы было удобнее распределять и сортировать сотрудников.

Поля таблицы:

- id – первичный ключ, bigint;

- name\_dep – название отдела университета, varchar (128).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. С полями id и name\_dep используется специальный атрибут «null: false». Это означает, что поля не могут быть нулевыми.

**3.2.5 Таблица Log\_statement**

Данная таблица предназначена для информации о заявлениях. Они могут быть разных типов: от отпусков, до увольнения за свой счет и тому подобные.

Поля таблицы:

- id – первичный ключ, bigint;

- days\_sum – сумма дней, integer;

- date\_leave – дата, когда работник уходит по заявлению (дата действия заявления), timestamp without time zone;

- date\_of\_ls – дата, когда работник составляет заявление, timestamp without time zone;

- id\_approver – номер сотрудника, который должен подтвердить заявление, bigint;

- comment\_ls – комментарий сотрудника к заявлению, varchar (300);

- type\_leave – тип заявления (типы будут прописаны в логике), smallint;

- approve – статус подтверждения, numeric;

- id\_employee – внешний ключ для таблицы employee (того работника, что составляет заявление), bigint.

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля за исключением comment\_ls используется специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении и сотрудник не может оставить их пустыми, за исключением комментария и самого скана документа. Поле date\_of\_ls будет заполняться автоматически при создании аккаунта. Формат данных timestamp without time zone означает, что нет привязки к часовому поясу. Это сделано, чтобы не было разногласий во времени.

**3.2.6 Таблица Document**

Данная таблица предназначена для хранения сканов оригинальных завлений. Они не являются обязательными, поэтому таблица log\_statement может существовать без привязки к данной таблице.

Поля таблицы:

- id – первичный ключ, bigint;

- id\_ls – внешний ключ для таблицы log\_statement, bigint;

- body\_doc – ссылка на скан оригинального заявления, varchar (300).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении.

**3.2.7 Таблица Job\_title**

Данная таблица предназначена для хранения списка должностей сотрудников БГУИР.

Поля таблицы:

- id – первичный ключ, bigint;

- job\_title\_name – название должности, varchar (128).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении.

**3.2.8 Таблица Job\_employee**

Данная таблица является реализацией связи многие-ко-многим между таблицами job\_title и employee.

Поля таблицы:

- id – первичный ключ, bigint;

- id\_job\_title – внешний ключ для таблицы job\_title, bigint;

- id\_employee – внешний ключ для таблицы employee, bigint.

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении.

**3.2.9 Таблица Task**

Данная таблица предназначена для хранения информации о заданиях или же действиях, которые нужно сделать работнику.

Поля таблицы:

- id – первичный ключ, bigint;

- date\_task – дата, когда работник составляет заявление, timestamp without time zone;

- name\_of\_task – название таски или ее суть, varchar (128);

- id\_executor – номер сотрудника, который будет исполнять задание, bigint;

- comment\_te – комментарий сотрудника к таске, по сути, описание, если таковое требуется, varchar (300).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля, кроме comment\_te, используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении. Поле comment\_te можно пропустить, потому что некоторые задания могут быть ясны без уточнений. Формат данных timestamp without time zone означает, что нет привязки к часовому поясу. Это сделано, чтобы не было разногласий во времени.

**3.2.10 Таблица Emp\_task**

Данная таблица является реализацией связи многие-ко-многим между таблицами task и employee.

Поля таблицы:

- id – первичный ключ, bigint;

- id\_job\_title – внешний ключ для таблицы job\_title, bigint;

- id\_employee – внешний ключ для таблицы employee, bigint.

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении.

**3.2.11 Таблица Event**

Данная таблица предназначена для хранения основной информации о пользователе, которая можно указать в общем доступе, и которая будет отображаться в глобальном поиске сотрудников.

Поля таблицы:

- id – первичный ключ, bigint;

- date\_of\_event – дата события, timestamp without time zone;

- type\_of\_event – название события, varchar (40);

- comment\_fe – комментарий сотрудника к событию, по сути, описание, если таковое требуется, varchar (300).

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля, кроме comment\_fe, используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении. Поле comment\_fe можно пропустить, потому что некоторые задания могут быть ясны без уточнений. Формат данных timestamp without time zone означает, что нет привязки к часовому поясу. Это сделано, чтобы не было разногласий во времени.

**3.2.12 Таблица Notice\_event**

Данная таблица является реализацией связи многие-ко-многим между таблицами event и employee.

Поля таблицы:

- id – первичный ключ, bigint;

- id\_event – внешний ключ для таблицы event, bigint;

- id\_recipient – номер сотрудника, которому предназначается отправить событие, bigint;

- id\_employee – внешний ключ для таблицы employee, bigint.

В таблице для колонок существуют определенные атрибуты и условия, которые задавались при их создании. Все поля используют специальный атрибут «null: false», это означает, что поле не может быть нулевым, потому что все эти поля имеют важное значение при заполнении.

**3.2.13 Представление Employee\_full\_info\_view**

Данная таблица является представлением, виртуальной или же логической таблицей, которая представляет собой поименованный запрос. Это представление нужно для облегченной работы с блоком данных, касаемых общей информации о сотруднике, которая представляет собой данные по аккаунту, персональные данные и общие сведения, которые будут в общем доступе.

Таблицы, которые объединены в данном представлении: personal, employee, department, job\_employee, job\_title, login.

Поля представления:

- job\_employee.id;

- login.id\_login;

- employee.first\_name;

- employee.middle\_name;

- employee.last\_name;

- personal.birth\_d;

- personal.entry\_d;

- login.login\_user;

- login.password\_user;

- login.mail\_user;

- login.role;

- employee.work\_number;

- personal.personal\_number;

- employee.location\_street;

- employee.cabinet\_office;

- department.name\_dep;

- job\_title.job\_title\_name.

**3.2.14 Представление Employees\_view**

Данная таблица является представлением, виртуальной или же логической таблицей, которая представляет собой поименованный запрос. Это представление нужно для облегченной работы с блоком данных, касаемых общей информации о сотруднике, которые могут быть в общем доступе и видимы для других сотрудников.

Таблицы, которые объединены в данном представлении: employee, department, job\_employee, job\_title.

Поля представления:

- job\_employee.id;

- job\_employee.id\_employee;

- employee.first\_name;

- employee.middle\_name;

- employee.last\_name;

- employee.work\_number;

- employee.location\_street;

- employee.cabinet\_office;

- department.name\_dep;

- job\_title.job\_title\_name.

**3.2.15 Представление Ls\_view**

Данная таблица является представлением, виртуальной или же логической таблицей, которая представляет собой поименованный запрос. Это представление нужно для облегченной работы с блоком данных, касаемых заявлений и их заполнения.

Таблицы, которые объединены в данном представлении: employee, log\_statement, document, login.

Поля представления:

- document.id;

- log\_statement.type\_leave;

- log\_statement.date\_leave;

- log\_statement.date\_of\_ls;

- log\_statement.days\_sum;

- log\_statement.id\_approver;

- log\_statement.approve;

- log\_statement.comment\_ls;

- log\_statement.id\_employee;

- document.body\_doc;

- login.role.

**3.2.16 Представление Events\_view**

Данная таблица является представлением, виртуальной или же логической таблицей, которая представляет собой поименованный запрос. Это представление нужно для облегченной работы с блоком данных, касаемых должности сотрудников.

Таблицы, которые объединены в данном представлении: event, notice\_event.

Поля представления:

- notice\_event.id;

- event.type\_of\_event;

- event.date\_of\_event;

- event.comment\_fe;

- notice\_event.id\_recipient;

- notice\_event.id\_employee.

**3.2.17 Представление Job\_title\_view**

Данная таблица является представлением, виртуальной или же логической таблицей, которая представляет собой поименованный запрос. Это представление нужно для облегченной работы с блоком данных, касаемых должности сотрудников.

Таблицы, которые объединены в данном представлении: job\_title, job\_employee.

Поля представления:

- job\_employee.id;

- job\_title.job\_title\_name;

- job\_employee.id\_employee;

**3.3 Описание структуры и взаимодействия между классами**

При разработке приложения использовался паттерн MVC, поэтому оно имеет структуру, состоящую из контроллера, сервиса и репозитория. Также стоит отметить, что все проектируемые сервисы разработаны с соблюдением всех правил и норм REST архитектуры. Рассмотрим классы каждого из данных слоев приложения.

**3.3.1 Класс ApplicationController**

**3.3.2 Класс ApplicationController**

**3.3.3 Класс ApplicationController**

**3.3.4 Класс ApplicationController**

**3.3.5 Класс ApplicationController**

**3.3.6 Класс ApplicationController**

**3.3.7 Класс ApplicationController**

**3.3.8 Класс ApplicationController**

**3.3.9 Класс ApplicationController**

**3.3.10 Класс ApplicationController**

**3.3.11 Класс ApplicationController**