**Описание компонентной структуры веб-приложения «ЛК Руководителя»**

**Пояснение**

Разработанное приложение состоит из множества React-компонентов, взаимодействующих друг с другом. В данном документе изложена вся необходимая для модификации информация касательно каждого компонента.

Компоненты перечислены в алфавитном порядке. В конце файла представлено текстовое дерево вложенности компонентов.

* **App.js**

Является корневым компонентом приложения, безусловно проходящим рендеринг в файле *index.js*. Включает в свой рендеринг один из двух компонентов: *LoginPage.js* (страница с авторизацией) или *WorkspacePage* (основной экран авторизованного пользователя), в зависимости от того, активна ли текущая сессия или нет (посылается асинхронный запрос серверу).

Используемые состояния:

- *page* (string) (название текущей открытой страницы «login»/«workspace»)

* **Content.js**

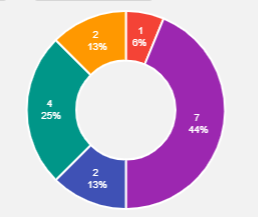
Проходит безусловный рендеринг в компоненте *WorkspacePage.js.* В зависимости от полученного *subpage* компонент выводит один из следующих компонентов: *EmployeesSubpage.js*, *StudentsSubpage.js*, *SubjectsSubpage.js*.

Принимаемые параметры:

- *subpage* (string) (наименование текущего открытого подраздела авторизованного пользователя: список сотрудников, обучающихся или дисциплин, «employees»/«students»/«subjects»)

- *editsubpage* (function) (редактирование текущего открытого подраздела)

* **DoughnutChart.js**



Отображается в модальном окне с информацией о выбранном(-ой) сотруднике/обучающемся/дисциплине (*Modal.js)* и представляет собой кольцевую диаграмму с отображением распределения выставляемых или получаемых оценок по объекту аналитики.

Применяет в себе компонент из библиотеки Chart.js. В зависимости от получаемого параметра *about* посылает определенный асинхронный запрос к серверу (запрос либо по сотруднику, либо по студенту, либо по дисциплине) в функции useEffect() и обновляет состояние *data,* по которому и строится диаграмма.

Принимаемые параметры:

- *key* (number) (случайное число, требуемое для оперативного обновления рендеринга в случае редактирования параметров фильтрации по датам пользователем)

- *about* (string) (название объекта аналитики, по которому и требуется получить информацию «employee»/«student»/«subject»)

- *id* (number)(уникальный номер сотрудника, студента или дисциплины, по которой нужно получить информацию)

- *startDate* (date, ‘yyyy-mm-dd’) (параметр фильтрации, настраиваемый пользователем, начальная дата для фильтрации оценок)

- *endDate* (date, ‘yyyy-mm-dd’) (параметр фильтрации, настраиваемый пользователем, конечная дата для фильтрации оценок)

Используемые состояния:

- *data* (JSON-object/string) (данные, полученные от сервера и применяющиеся как источник данных для диаграммы, либо простая строка с указанием ошибки)

- *studentAvgMark* (number) (число, показывающее средний балл студента за весь период обучения)

- *subjectTotalMarks* (number) (число, показывающее общее количество выставленных оценок по дисциплине)

* **EmployeesSubpage.js**

Изображение выглядит как текст

Автоматически созданное описание

Один из подразделов авторизованного пользователя, является дочерним компонентом для *Content.js.* Включает в себя компонент *Modal.js*, отображающий детальную информацию о выбранном пользователем сотруднике.

Принимаемые параметры:

- *editSubpage* (function) (изменение текущего открытого подраздела: список сотрудников, обучающихся, дисциплин «employees»/«students»/«subjects»)

Используемые состояния:

- *employees* (array) (массив со списком объектов – сотрудников, получаемый от сервера)

- *sortMode* (string) (название столбца, по которому следует производить сортировку при рендеринге таблицы)

- *filterMode* (string) (название активного параметра из панели фильтрации над таблицей)

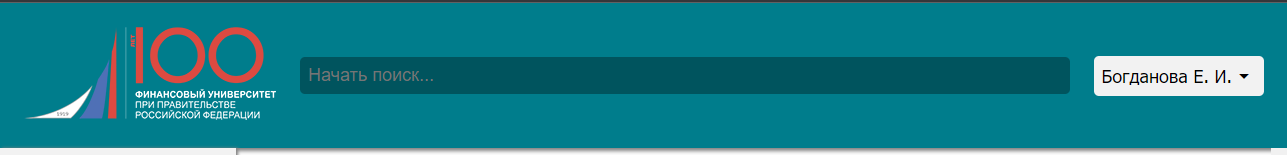
- *userFilterText* (string) (значение текстового поля фильтрации над таблицей)

- *filterByBirthdate* (boolean) (осуществлять ли фильтрацию списка сотрудников по наличию дня рождения на текущую дату)

- *modalVisible* (boolean) (отображать ли модальное окно с подробной информацией о конкретном сотруднике)

- *empIdClicked* (number) (номер сотрудника из списка (id из базы данных), по которому следует показать модальное окно)

* **Header.js**



Представляет собой «шапку» веб-портала. Включает в себя компоненты *Modal.js* (модальное окно с информацией об активном пользователе ИС) и *SearchResultsBlock.js* (показ результатов глобального поиска). В рамках заданного компонента осуществляется отображение имени активного пользователя и выход из аккаунта, здесь же срабатывает рендеринг выпадающего меню (не представлен в виде отдельного компонента). Является дочерним для *WorkspacePage.js.*

Принимаемые параметры:

- *editPage* (function) (изменение открытой страницы: экран входа «login» или страница авторизованного пользователя «workspace»)

Используемые состояния:

- *name* (string) (имя активного пользователя ИС)

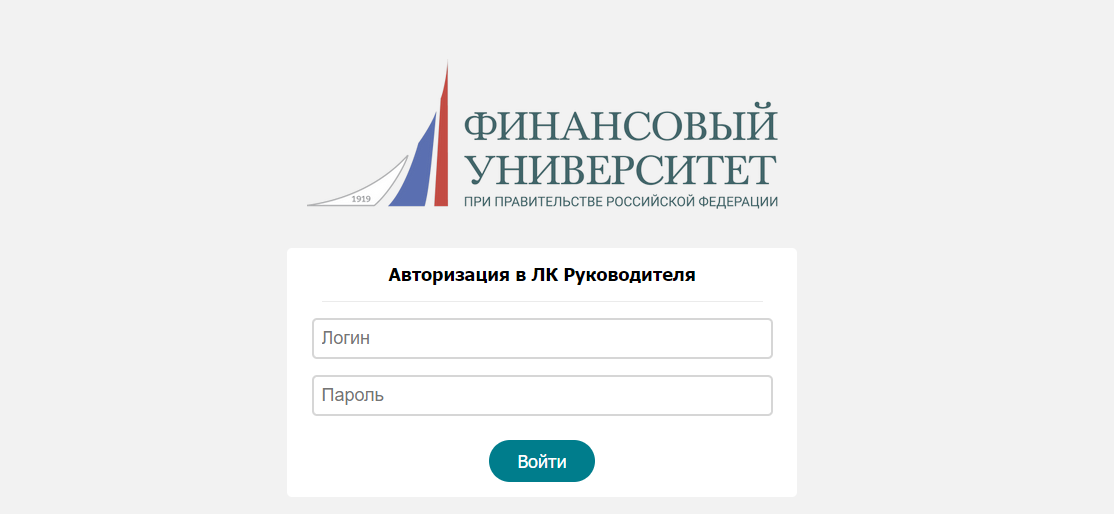
- *menuOpened* (boolean) (открыто ли выпадающее меню)

- *showUserCard* (boolean) (отображать ли модальное окно с информацией о пользователе ИС)

- *userQueryText* (string) (текстовый запрос пользователя в поле глобального поиска)

- *showSearchResults* (boolean) (отображать ли список результатов поиска под текстовым полем ввода)

* **LoginPage.js**



Страница с авторизацией. Не имеет дочерних компонентов. Является дочерним для *App.js.* Редактирует поле с токеном (*session*) в LocalStorage (успех или ошибка при попытке аутентификации).

Принимаемые параметры:

- *editPage* (function) (изменение текущей открытой страницы, «workspace» или «login»)

Используемые состояния:

- *login* (string) (введенный в поле логин пользователя)

- *password* (string) (введенный в поле пароль пользователя)

* **Modal.js**

Изображение выглядит как текст

Автоматически созданное описание

Компонент представляет собой модальное окно с затемняющим фоном на всю область веб-страницы. Отображает в зависимости от переданного параметра сведения об активном пользователе, сотруднике, обучающемся или дисциплине.

Является дочерним компонентом для: *Header.js, EmployeesSubpage.js, StudentsSubpage.js, SubjectsSubpage.js.*

Включает в себя компоненты: *DoughtnutChars.js, TableGroupByComponent.js, VerticalBarChart.js.*

Принимаемые параметры:

- *about* (string) (объект аналитики какого типа применяется, «user»/ «employee»/ «student»/ «subject»)

- *id* (number) (уникальный номер из БД объекта, по которому требуется информация)

- *editVisible* (function) (изменить видимость модального окна)

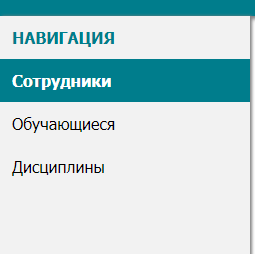
Используемые состояния:

- *modalContent* (JSON/string) (сведения об объекте аналитики, полученные от сервера, или текст ошибки при получении данных)

- *filterStartDate* (date, ‘yyyy-mm-dd’) (параметр фильтрации оценок по дате от пользователя, начальная дата)

- *filterEndDate* (date, ‘yyyy-mm-dd’) (параметр фильтрации оценок по дате от пользователя, конечная дата)

* **NavMenu.js**



Компонент меню навигации между подразделами (списками сотрудников, обучающихся и дисциплин). Является дочерним для *WorkspacePage.js.*

Принимаемые параметры:

- *subpage* (string) (активный подраздел на странице авторизованного пользователя, «employees»/«students»/«subjects»)

- *editsubpage* (function) (изменение активного подраздела)

Используемые состояния:

- *mobMenuRevealed* (boolean) (раскрыто ли меню навигации в мобильной версии сайта)

* **SearchResultsBlock.js**

Изображение выглядит как текст

Автоматически созданное описание

Компонент, отображающий в выпадающем списке результаты глобального поиска. Является дочерним компонентом для *Header.js*.

Принимаемые параметры:

- *query* (string) (текстовый запрос пользователя из поля ввода)

Используемые состояния:

- *searchResult* (array/string) (массив с JSON-объектами, полученный от сервера, либо уведомление в виде текста о нулевом результате)

* **StudentsSubpage.js**

Изображение выглядит как текст

Автоматически созданное описание

Один из подразделов авторизованного пользователя, является дочерним компонентом для *Content.js.* Включает в себя компонент *Modal.js*, отображающий детальную информацию о выбранном пользователем обучающемся.

Принимаемые параметры:

- *editSubpage* (function) (изменение текущего открытого подраздела: список сотрудников, обучающихся, дисциплин «employees»/«students»/«subjects»)

Используемые состояния:

- *students* (array) (массив со списком объектов – обучающихся, получаемый от сервера)

- *sortMode* (string) (название столбца, по которому следует производить сортировку при рендеринге таблицы)

- *filterMode* (string) (название активного параметра из видимой (первой) панели фильтрации над таблицей)

- *userFilterText* (string) (значение первого (видимого) текстового поля фильтрации над таблицей)

- *showExtendedFilterForm* (boolean) (отображать ли раскрытой дополнительную форму с фильтрацией)

- *extraUserFilterText* (string) (значение второго (дополнительного) текстового поля фильтрации над таблицей)

- *extraFilterMode* (string) (название активного параметра из второй (дополнительной) панели фильтрации над таблицей)

- *modalVisible* (boolean) (отображать ли модальное окно с подробной информацией о конкретном обучающемся)

- *studIdClicked* (number) (номер обучающегося из списка (id из базы данных), по которому следует показать модальное окно)

* **SubjectsSubpage.js**

Изображение выглядит как текст

Автоматически созданное описание

Один из подразделов авторизованного пользователя, является дочерним компонентом для *Content.js.* Включает в себя компонент *Modal.js*, отображающий детальную информацию о выбранной пользователем дисциплине.

Принимаемые параметры:

- *editSubpage* (function) (изменение текущего открытого подраздела: список сотрудников, обучающихся, дисциплин «employees»/«students»/«subjects»)

Используемые состояния:

- *subjects* (array) (массив со списком объектов – дисциплин, получаемый от сервера)

- *userFilterText* (string) (значение текстового поля фильтрации над таблицей)

- *modalVisible* (boolean) (отображать ли модальное окно с подробной информацией о конкретной дисциплине)

- *subjIdClicked* (number) (номер дисциплины из списка (id из базы данных), по которой следует показать модальное окно)

* **TableGroupByComponent.js**

Изображение выглядит как стол

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Компонент, отвечающий за формирование запроса к серверу, отображение распределения всех оценок за определенный период по дисциплинам или учебных планов по дисциплине.

Является дочерним для *Modal.js.*

Включает в себя компонент *TableGroupByComponentDetails.js* (отображение распределения оценок по типу (5, 4, 3, 2, зачтено, не зачтено) в рамках одной дисциплины, по которой кликнул пользователь)*.*

Принимаемые параметры:

- *about* (string) (какой тип у запрашиваемого объекта аналитики, «employee» вызывает отображение распределения оценок, «subject» вызывает показ учебных планов по дисциплине)

- *id* (number) (уникальный номер из БД объекта аналитики)

- *startDate* (date, ‘yyyy-mm-dd’) (начальная дата в качестве параметра фильтрации)

- *endDate* (date, ‘yyyy-mm-dd’) (конечная дата в качестве параметра фильтрации)

- *key* (number) (случайное число для оперативного обновления рендеринга компонента)

Используемые состояния:

- *data* (array) (массив с объектами, полученный от сервера (дисциплины с количеством оценок либо учебные планы))

- *marksTotal* (number) (количество всех оценок, отображаемых в таблице, число подсчитывается на клиенте)

- *avgMark* (number) (средний балл по всем оценкам в таблице, подсчитывается на клиенте)

- *detailsBlockVisible* (boolean) (отображать ли компонент *TableGroupByComponentDetails.js*)

- *detailsBlockSubjectId* (number) (уникальный номер из БД дисциплины, по которой требуется детализация оценок по типу, на какую дисциплину из таблицы кликнул пользователь)

- *detailsBlockSubjectName* (string) (наименование дисциплины, по которой требуется детализация оценок по типу, на какую дисциплину из таблицы кликнул пользователь)

* **TableGroupByComponentDetails.js**

Изображение выглядит как стол

Автоматически созданное описание

Компонент, отвечающий за показ распределения выставленных сотрудником оценок по типу в рамках одной дисциплины. Является дочерним компонентом для *TableGroupByComponent.js.*

Принимаемые параметры:

- *userId* (number) (уникальный номер сотрудника из БД, по которому нужно получить информацию)

- *subjectId* (number) (уникальный номер дисциплины из БД, по которой и требуется детализация)

- *startDate* (date, ‘yyyy-mm-dd’) (начальная дата как параметр фильтрации)

- *endDate* (date, ‘yyyy-mm-dd’) (конечная дата в качестве параметра фильтрации)

- *subjectName* (string) (название дисциплины для отображения в заголовке)

Используемые состояния:

- *data* (array/string) (массив с результатами от сервера либо текст с ошибкой)

* **VerticalBarChart.js**



Компонент, демонстрирующий средний балл обучающегося по семестрам. Является дочерним компонентом для *Modal.js.*

За основу берется библиотечный компонент Bar из Chart.js.

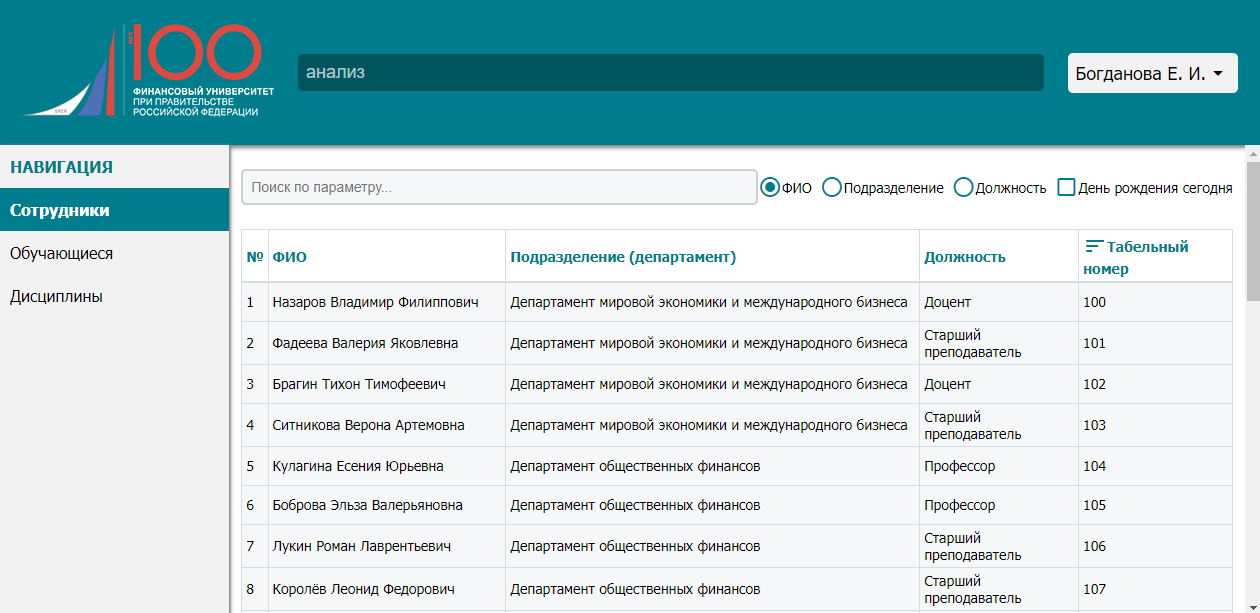
Принимаемые параметры:

- *id* (number) (уникальный номер обучающегося из БД)

Используемые состояния:

- *data* (JSON-object/string) (данные, получаемые от сервера и используемые для отображения диаграммы, либо текст с ошибкой)

* **WorkspacePage.js**



Компонент, являющийся дочерним для *App.js,* который отображается только в случае успешной авторизации пользователя в системе. Включает в себя дочерние компоненты *Header.js, NavMenu,js, Content.js.*

Принимаемые параметры:

- *editPage* (function) (изменение активной страницы, «login» или «workspace»)

Применяемые состояния:

- *subpage* (string) (текущий активный раздел авторизованного пользователя, «employees»/«students»/«subjects»)

**Текстовое дерево вложенности компонентов**

* App.js
  + LoginPage.js
  + WorkspacePage.js
    - Header.js
      * Modal.js
        + DoughnutChart.js
        + TableGroupByComponent.js

TableGroupByComponentDetails.js

* + - * + VerticalBarChart.js
      * SearchResultsBlock.js
    - NavMenu.js
    - Content.js
      * EmployeesSubpage.js
        + Modal.js

DoughnutChart.js

TableGroupByComponent.js

TableGroupByComponentDetails.js

VerticalBarChart.js

* + - * StudentsSubpage.js
        + Modal.js

DoughnutChart.js

TableGroupByComponent.js

TableGroupByComponentDetails.js

VerticalBarChart.js

* + - * SubjectsSubpage.js
        + Modal.js

DoughnutChart.js

TableGroupByComponent.js

TableGroupByComponentDetails.js

VerticalBarChart.js