МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

по направлению подготовки
09.03.04 Программная инженерия
Направленность (профиль) Индустриальное производство программного обеспечения

Студента Василения Ивана Валерьевича шифр 220887

Институт приборостроения, автоматизации и информационных технологий

Кафедра информационных систем и цифровых технологий

Тема выпускной квалификационной работы
«Разработка программного обеспечения подсистемы «Контроль и учет посещаемости и работы студентов на занятиях» для сайта ОГУ им. И.С. Тургенева»

Студент	И.В. Василения
Руководитель	А.Ю. Ужаринский
Нормоконтроль	О.В. Захарова
И.о. зав. кафедрой информационных систем и цифровых технологий	Д.В. Рыженков

Орёл 2026

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»

Институт приборостроения, автоматизации и информационных технологий
Кафедра информационных систем и цифровых технологий
Направление 09.03.04 Программная инженерия
Направленность (профиль) Индустриальное производство программного обеспечения

УТВЕРЖДАЮ:
И.о. зав. кафедрой

_____ Д.В. Рыженков
«___»_____ 20__ г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

студента Василения Ивана Валерьевича шифр 220887

1 Тема ВКР «Разработка программного обеспечения подсистемы «Контроль и учет посещаемости и работы студентов на занятиях» для сайта ОГУ им. И.С. Тургенева»
Утверждена приказом по университету от «26» ноября 2025г. № 8-305

2 Срок сдачи студентом законченной работы «22» мая 2026г.

3 Исходные данные к работе
Теоретический материал; информация о предметной области

4 Содержание пояснительной записки (перечень подлежащих разработке вопросов)
Анализ решаемой проблемы, методов и средств ее решения, формулировка требований к разработке
Определение спецификаций программного обеспечения
Проектирование программного обеспечения

Разработка программного обеспечения подсистемы «Контроль и учет посещаемости и работы студентов на занятиях» для сайта ОГУ им. И.С. Тургенева»

5 Перечень демонстрационного материала

Презентация, отображающая основные этапы и результаты выполнения ВКР

Дата выдачи задания « 26 » ноября 2025 г.

Руководитель _____ А.Ю. Ужаринский
(подпись)

Задание принял к исполнению _____ И.В. Василения
(подпись)

КАЛЕНДАРНЫЙ ПЛАН

Наименование этапов работы	Сроки выполнения этапов работы	Примечание
Анализ задачи	26.11.2025 – 28.11.2025	
Проведение теоретических исследований	01.12.2025 – 18.12.2025	
Определение спецификаций	19.12.2025 – 10.01.2026	
Проектирование системы	12.01.2026 – 31.03.2026	
Реализация системы	01.04.2026 – 30.04.2026	
Оформление ВКР	04.05.2026 - 22.05.2026	

Студент _____ И.В. Василения
(подпись)

Руководитель _____ А.Ю. Ужаринский
(подпись)

АННОТАЦИЯ

ВКР 91 с., 43 рис., 1 табл., 10 источников, 1 прил.

ИНФОРМАЦИОННАЯ СИСТЕМА, ПОСЕЩАЕМОСТЬ, ЭЛЕКТРОННЫЙ ЖУРНАЛ, САЙТ.

Выпускная квалификационная работа посвящена разработке программного обеспечения подсистемы «Контроль и учет посещаемости и работы студентов на занятиях» для сайта ОГУ им. И.С. Тургенева».

В первой главе сформулирована проблема и требования к разрабатываемой подсистеме.

Во второй главе определяются спецификации разрабатываемого программного обеспечения.

В третьей главе описывается проектирование программного обеспечения.

В четвертой главе описывается реализация компонентов и интерфейсов подсистемы.

СОДЕРЖАНИЕ

СОДЕРЖАНИЕ 4
ВВЕДЕНИЕ 5
1 АНАЛИЗ РЕШАЕМОЙ ПРОБЛЕМЫ, МЕТОДОВ И СРЕДСТВ ЕЕ РЕШЕНИЯ, ФОРМУЛИРОВКА ТРЕБОВАНИЙ К РАЗРАБОТКЕ 6
1.1 Анализ предметной области и формализация решаемой задачи 6
1.2 Анализ существующих методов решения поставленной задачи 7
1.3 Обзор аналогичных решений 12
1.4 Формирование функциональных и нефункциональных требований к системе 17
1.5 Обоснование инструментария разработки 18
1.6 Разработка обобщенной архитектуры 20
2 ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 22
2.1 Определение функциональных спецификаций 22
2.2 Определение поведенческих спецификаций 23
2.3 Определение информационных спецификаций 28
3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 31
3.1 Проектирование структуры программного обеспечения 31
3.2 Проектирование компонентов 32
3.3 Проектирование структур данных 36
3.4 Проектирование алгоритмов 39
3.5 Проектирование интерфейсов 44
4 РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПОДСИСТЕМЫ «КОНТРОЛЬ И УЧЕТ ПОСЕЩАЕМОСТИ И РАБОТЫ СТУДЕНТОВ НА ЗАНЯТИЯХ» ДЛЯ САЙТА ОГУ ИМ. И.С. ТУРГЕНЕВА 51
4.1 Реализация компонентов 51
4.2 Реализация интерфейсов 57
ЗАКЛЮЧЕНИЕ 63
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 65
Приложение – Фрагменты исходного кода 71

ВВЕДЕНИЕ

Развитие информационных технологий с каждым днем все больше влияет на повседневную жизнь людей. Возможности к автоматизации и аналитике данных упрощают множество сфер человеческой жизни. Образовательная сфера не стала исключением.

Данная тенденция в организации учебного процесса стимулирует студентов активнее применять различные инструменты, доступные в информационной среде ВУЗа. Для профессорско-преподавательского состава учебного заведения развитие информационных технологий представляет собой способ ускорить и упростить многие образовательные процедуры, а также открывает возможности по сбору и аналитике больших массивов данных.

Темой выпускной квалификационной работы является Разработка программного обеспечения подсистемы «Контроль и учет посещаемости и работы студентов на занятиях» для сайта ОГУ им. И.С. Тургенева.

Целью выпускной квалификационной работы является реинжениринг процесса и расширение функционала учёта посещаемости и работы студентов на занятиях для упрощения контроля за заполнением и повышения надёжности работы сервиса.

Для достижения поставленной цели необходимо выполнить следующие задачи:

а) провести анализ решаемой проблемы, методов и средств ее решения, сформулировать задачи исследования и требования к разработке;

б) провести теоретические исследования и определить спецификаций программного обеспечения;

в) спроектировать программное обеспечение;

г) разработать макет программных компонентов или экспериментального образца программного обеспечения.

1 АНАЛИЗ РЕШАЕМОЙ ПРОБЛЕМЫ, МЕТОДОВ И СРЕДСТВ ЕЕ РЕШЕНИЯ, ФОРМУЛИРОВКА ТРЕБОВАНИЙ К РАЗРАБОТКЕ

1.1 Анализ предметной области и формализация решаемой задачи

В процессе обучения преподаватель заполняет журнал учёта посещаемости (далее журнал). Журнал представляет собой книгу с расчерченными таблицами, в которые вносится информация о группе, преподавателе, проводимых датах занятий, темах занятий, присутствии и отсутствии отдельных студентов на занятии и т.д. Под заполнением журнала понимается заполнение вышеописанных таблиц.

Журнал есть у каждой студенческой группы. В каждой группе назначается староста, ответственный за предоставление журнала группы преподавателю. Староста в начале каждого учебного семестра получает журнал в деканате, а в конце возвращает его обратно. На занятиях староста предоставляет журнал преподавателю, все остальное время журнал находится у старосты. Это порождает множество рисков, связанных с порчей и утерей журнала группы.

На каждом занятии преподаватель должен отмечать посещаемость студентов. Прежде всего отметка о посещаемости является для деканата университета инструментом контроля за выполнением студентом учебного плана. Также данные о посещаемости используются для составления отчетов учредителям или органам управления образованием, так как отражают реальный ход учебного процесса.

Одновременно с учетом посещаемости преподаватель фиксирует сдачу студентами практических и лабораторных работ, что является неотъемлемой частью контроля успеваемости. Эта информация накапливается в течение семестра и служит основой для допуска студента к промежуточной аттестации.

Эта информация может запрашиваться различными проверяющими инстанциями, такими как Рособрнадзор, запрашивающий журналы учета

посещаемости и успеваемости в ходе аккредитационной экспертизы, чтобы убедиться, что образовательная деятельность ведется в соответствии с законодательством.

Таким образом, разрозненное ведение бумажных журналов создает риски для всех участников процесса. Преподавателю приходится тратить время на двойную работу, а проверяющие органы могут выявить несоответствия в отчетности. Автоматизация этого процесса позволяет не только упростить ежедневную работу преподавателя, но и создать единую, прозрачную и достоверную базу данных. В такой системе все отметки о посещаемости и сданных работах мгновенно становятся доступными для формирования любых отчетов. Как для деканата, так и для официальных органов вроде Рособрнадзора или Росстата. Это исключает дублирование информации и ошибки «человеческого фактора», гарантируя, что данные, на основании которых принимаются решения об аттестации студентов или о выделении бюджетных мест, являются актуальными и точными.

1.2 Анализ существующих методов решения поставленной задачи

Заполнение журнала является трудоемким длительным процессом для преподавателей. Каждый преподаватель обязан иметь напечатанные списки групп, у которых ведет занятия. За выдачу списков групп преподавателю отвечают старосты групп. Журнал заполняется в соответствии с этими списками. Необходимость иметь при себе данные списки создает для преподавателя дополнительные трудности.

В свою очередь, для студентов использование «бумажного» журнала также ведет к неудобствам. Студенты не могут в любой момент ознакомиться с актуальной информацией о своей посещаемости и оценках за лабораторные и практические работы. Для этого им приходится обращаться либо к старосте группы, либо к преподавателю на соответствующем занятии.

Состав группы в процессе обучения может меняться, студенты могут отчисляться, восстанавливаться и переводиться из других учебных заведений. Изменение состава группы приводит к утере актуальности списков групп,

находящихся у преподавателей, то есть нарушается согласованность данных. Впоследствии это приводит к необходимости заново создать список группы и выдать всем преподавателям, что доставляет дополнительные неудобства для старост групп и преподавателей.

Случаются ситуации, в которых необходимо собрать и проанализировать крупный массив данных из журнала. В качестве таких ситуаций могут выступать, например, пересдачи с комиссией у студентов. В этом случае комиссии необходимо собрать данные о посещаемости и успеваемости студента за весь период обучения, чтобы учесть их при оценивании. При использовании «бумажного» журнала подобный анализ больших массивов данных является крайне трудоемкой задачей.

Пример журнала представлен на рисунке 1.2.1.

Рисунок 1.2.1 – Пример журнала посещаемости

На сайте ОГУ им. И.С. Тургенева уже существует система учета посещаемости и успеваемости. Использование этой системы не является обязательным, преподаватели могут использовать ее в качестве дополнительного контроля, но это не избавляет их от необходимости заполнения «бумажного» журнала. Для заполнения посещаемости необходимо выбрать нужную группу и дату занятия, секция выбора группы и даты занятия представлена на рисунке 1.2.2. Затем в сформировавшуюся таблицу

необходимо внести информацию о посетивших занятие студентах и о сданных ими работах, вид сформированной таблицы на рисунке 1.2.3.

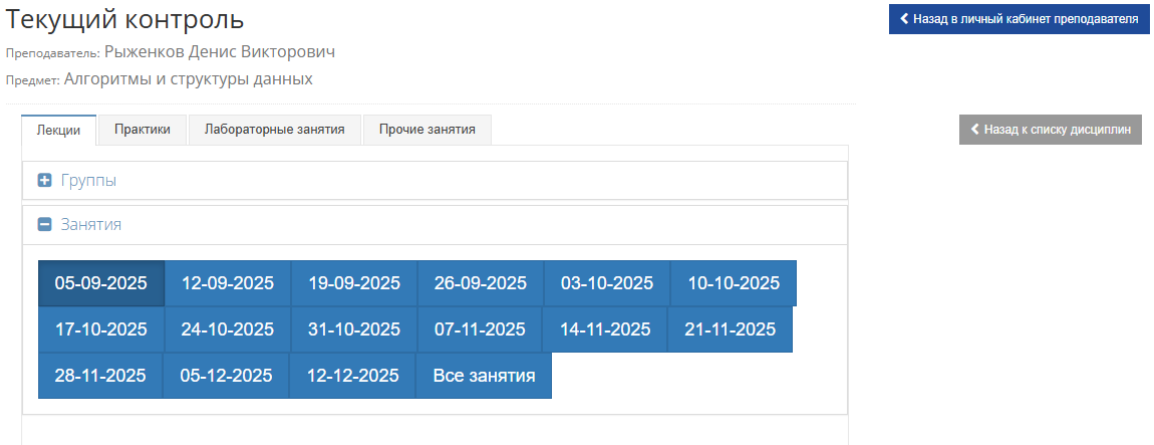

Рисунок 1.2.2 – Блок выбора занятия в существующей системе

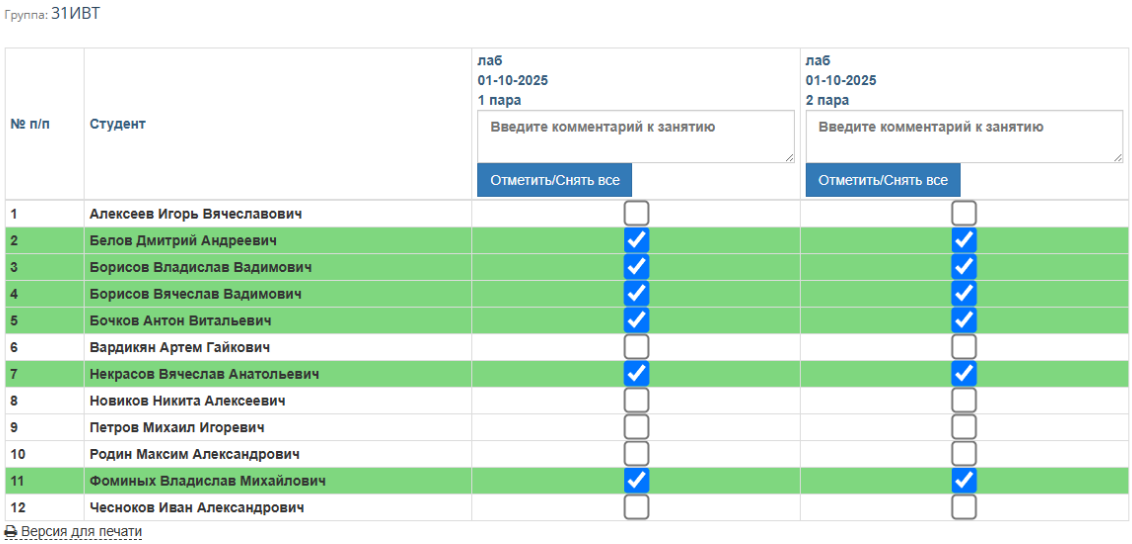

Рисунок 1.2.3 – Сформированная таблица посещаемости

в существующей системе

Существующая система позволяет в рамках двоичной логики (был или не был) ставить отметки о посещении. Отметки о сдаче лабораторных и практических работ имеют три варианта, которым соответствуют символы на рисунке 1.2.6. Данная система уже решает многие недостатки «бумажного» журнала. Наглядно увидеть упрощение процессов ведение журнала можно увидеть на диаграммах 1.2.4 и 1.2.5.

В связи с распоряжением и.о. ректора ОГУ им И.С. Тургенева от 10 февраля 2025 г. деканы обязаны контролировать своевременность заполнения электронного журнала профессорско-преподавательским составом институтов.

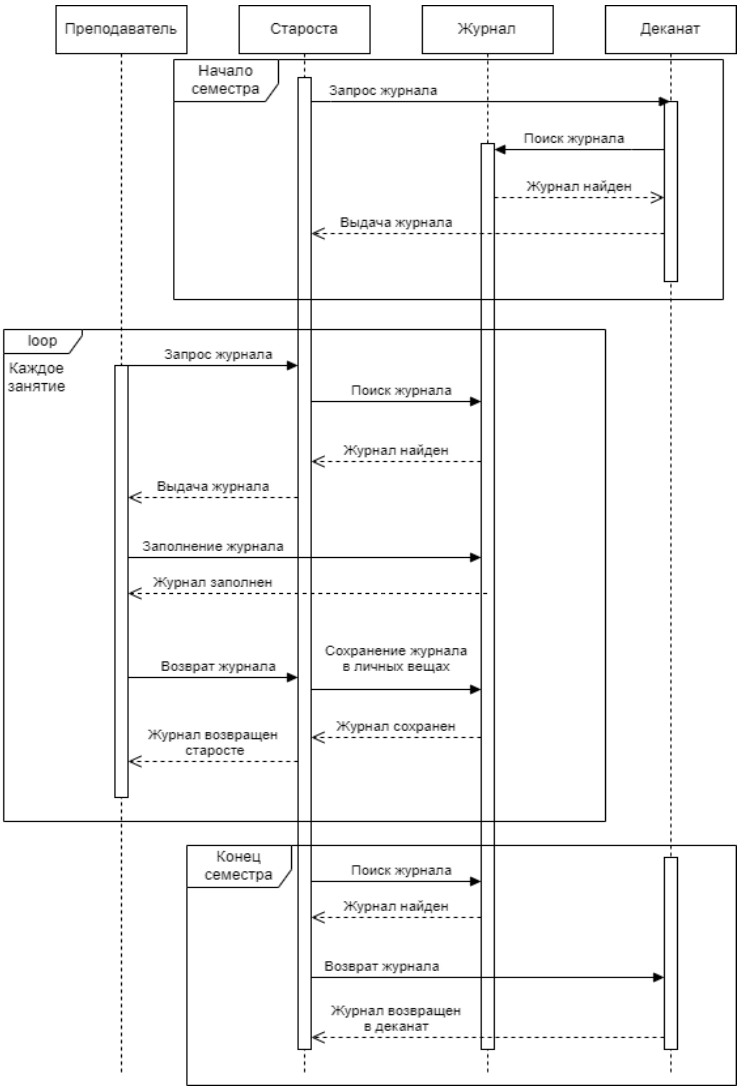

Рисунок 1.2.4 – Процесс работы с бумажным журналом

Существующая система не позволяет должным образом осуществить вышеописанный контроль. Для осуществления данного контроля необходимо вести учет деятельности преподавателей в подсистеме сайта. В существующей системе у отметки о посещаемости может быть только 2 состояния. Соответственно, если никто из группы не придет на занятие, преподаватель оставит отметки нетронутыми, и никакой информации о деятельности преподавателя в подсистеме не сохранится. Для решения этой проблемы необходимо ввести третье состояние для отметки посещаемости. Оно позволит

преподавателю понимать, каких студентов он отметил отсутствующими, а каких не отмечал вовсе.

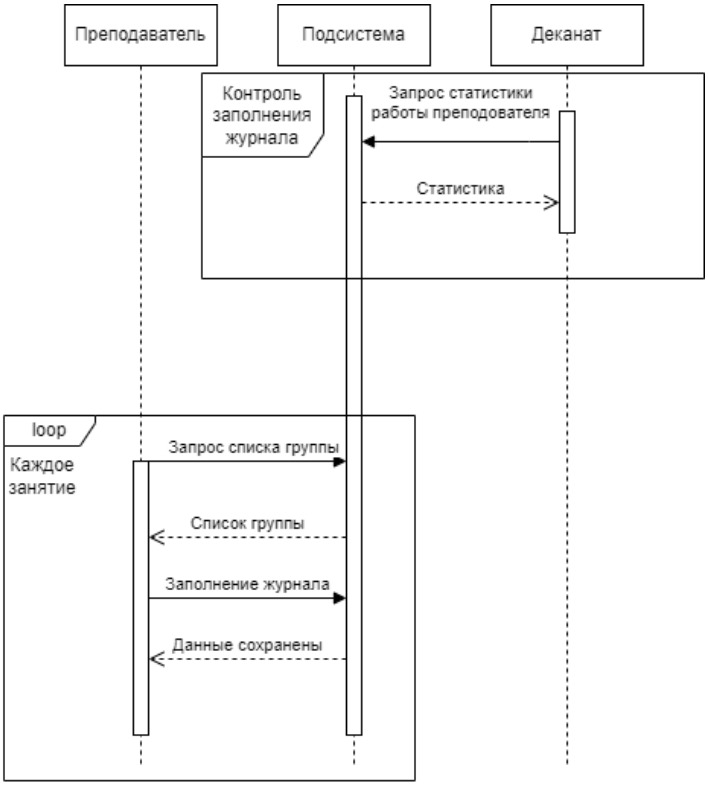

Рисунок 1.2.5 – Процесс работы с подсистемой сайта

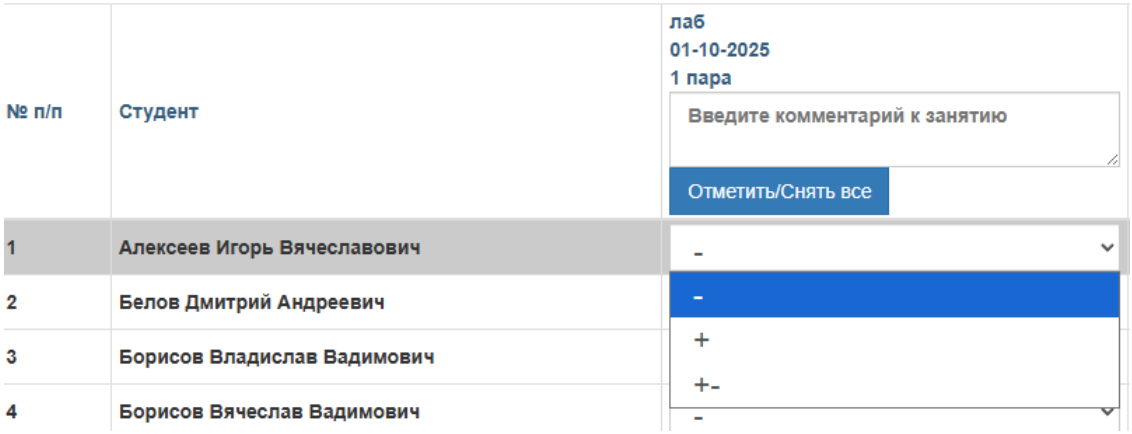

Рисунок 1.2.6 – Варианты отметки о сдаче работы

в существующей системе

Для того, чтобы сформировать список группы и отметить посещаемость преподавателю необходимо совершить ряд действий. Он должен выбрать тип занятия, группу и дату. Каждый выбор является отдельным запросом, что в условиях низкой скорости интернета может оказаться чрезмерно длительным.

Зачастую преподаватели стремятся отметить посещаемость во время занятия, в его начале или окончании. Соответственно, в большинстве случаев известно без всякого выбора, какую таблицу нужно сформировать. В разрабатываемой системе необходимо реализовать формирование списка группы для актуального на момент времени занятия.

Некоторые дисциплины преподаются не всей группе, а отдельно двум подгруппам, содержащим примерно половину из состава группы. Деление на подгруппы неофициально, но расписание составляется с учетом разных подгрупп. Из этого следует, что необходимо реализовать разделение на подгруппы в разрабатываемой подсистеме. Так как при использовании бумажного журнала каждый преподаватель на своих бумажных списках групп самостоятельно распределял студентов по подгруппам, будет рационально оставить право разделения на подгруппы за преподавателями. Чтобы преподаватели не перезаписывали данные о разделении, также будет правильно сохранять разделения каждого преподавателя отдельно.

Таким образом, необходимо разработать новую подсистему учета посещаемости и сдачи работ студентами, которая сохранит функционал существующей системы, и добавить в разрабатываемую подсистему логирование событий, троичную логику отметок посещаемости, разделение на подгруппы и формирование списка группы на основе проходящего в момент времени занятия.

1.3 Обзор аналогичных решений

Одним из наиболее перспективных инструментов в данной сфере является программный продукт «1С:Электронный журнал колледжа», разработанный на базе платформы «1С:Предприятие 8.3». Данное решение представляет собой комплексное средство для организации работы преподавателей и администрации в едином информационном пространстве [1].

Платформа распределяет пользователей по ролям (администратор, преподаватель, студент, куратор). Роль определяет, какой функционал будет

доступен пользователю. Доступом к наибольшему функционалу обладают пользователи с ролью «Администратор». Перечень доступных им разделов представлен на рисунке 1.3.1.

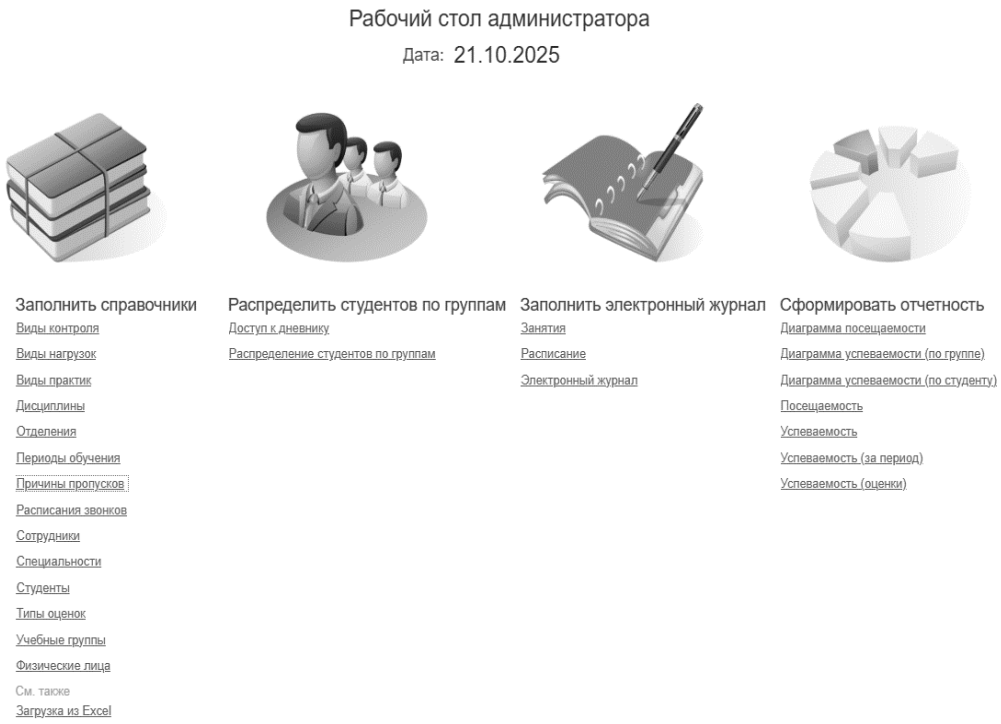

Рисунок 1.3.1 – Главная страница Администратора в «1С: Электронный журнал колледжа»

Пользователи с ролью «Преподаватель» работают главным образом с разделом «Электронный журнал», в котором отмечают посещаемость студентов и сдачу ими работ. Его вид представлен на рисунке 1.3.2. В качестве отметки можно поставить любой текст, однако сохранится оценка только в том случае, если представляет из себя число в диапазоне от 1 до 5.

В электронном журнале поддерживается учет до 3 подгрупп в одной группе. В нём преподаватели могут вести учёт посещений и выставлять оценки по всем видам контроля: от лекционных и практических занятий до промежуточной и итоговой аттестации. Параллельно ведётся учёт календарно-тематического плана, что позволяет соотносить учебный процесс с программой. Все эти данные доступны студентам через персональный

электронный дневник, где они в реальном времени могут отслеживать свою успеваемость и посещаемость.

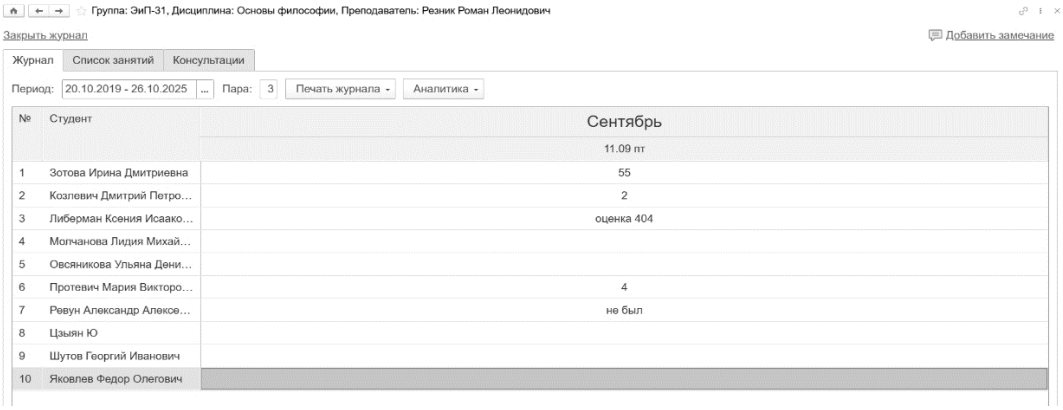

Рисунок 1.3.2 – Вид раздела «Электронный журнал»

Другим вариантом является платформа «Moodle». «Free Dean's Office» (Электронный деканат) – это модуль для среды дистанционного обучения «Moodle», который добавляет возможность управления процессом обучения, типичным для российских школ, колледжей и ВУЗов.

В данном случае используется решительно другой подход к решению задач автоматизации учебного процесса. В отличии от создания закрытой монолитной системы, как в «1С», этот проект является модулем для широко распространенной и бесплатной системы обучения «Moodle». Этот модуль расширяет функционал более крупной, зачастую уже внедренной системы. Это снижает порог входа для пользователей, которые уже знакомы с «Moodle».

Администратор учебного заведения или деканата предварительно формирует в модуле академическую структуру: создает учебные планы, закрепляет за ними дисциплины, формирует группы студентов и назначает преподавателей. Именно на этом уровне устанавливаются формальные рамки для последующего ведения журнала.

Преподаватель, в свою очередь, осуществляет оперативную работу в своих курсах Moodle. Он создает элементы курса — задания, тесты, лекции — и выставляет за них оценки. Важно отметить, что на этом этапе преподаватель пользуется стандартным, привычным для любого пользователя Moodle

инструментарием журнала оценок, который привязан к его конкретному курсу.

Функционал учета посещаемости учебных занятий реализован в рамках системного модуля «Посещаемость». Данный инструмент предоставляет преподавателям средства для фиксации посещений, а обучающиеся получают доступ к актуальным данным. Для оценки присутствия предусмотрен набор статусов: «Присутствовал», «Отсутствовал», «Опоздал» и «Уважительная причина». Использование этих статусов делает процедуру заполнения электронного журнала более гибкой и наглядной. Интерфейс модуля показан на Рисунке 1.3.3.

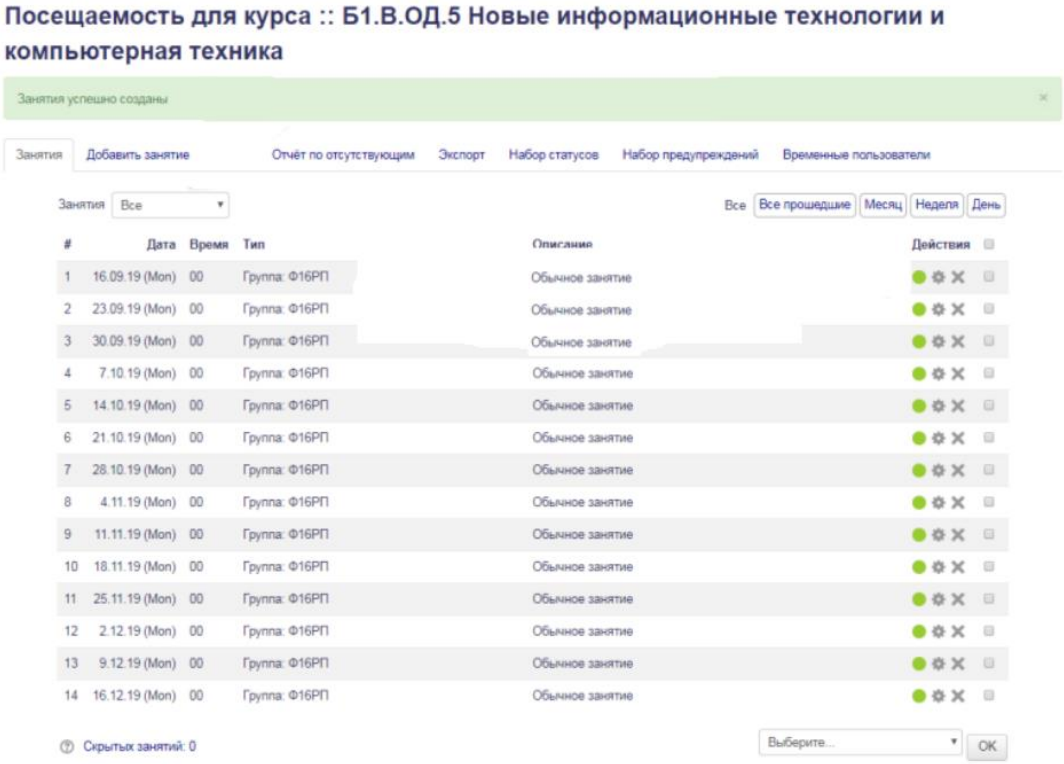

Рисунок 1.3.3 – Модуль «Посещаемость» системы «Moodle»

Лучшие Российские ВУЗы используют системы автоматизации учета успеваемости и посещаемости. Например, в МГУ преподаватели имеют возможность проставлять оценки через мобильное приложение. В НИУ ВШЭ преподаватели видят статистику по успеваемости студентов, что помогает регулировать учебный процесс. Также во многих вузах распространена балльно-рейтинговая система, в рамках которой итоговая оценка по предмету

за семестр напрямую связана с работой студента в течение семестра. В РГГУ система учитывает активность на занятиях и выполнение домашних заданий, формируя итоговую оценку на основе накопленных баллов. Также электронные журналы введены в НГУ и КФУ [4].

В таблице 1.3 приведена сравнительная характеристика рассматриваемых систем, а также разрабатываемой системы. В ней содержится функционал каждой системы, а также достоинства и недостатки каждой из них.

Таблица 1.3 – Сравнительная характеристика аналогичных решений

Критерии сравнения	Moodle	1С: Электронный журнал колледжа	Существующая система	Разрабатываемое решение
Проставление отметок о посещении	+	+	+	+
Просмотр отметок студентами	+	+	+	+
Проставление отметок о сдаче работ	-	+	+	+
Добавление комментария к занятию	-	-	+	+
Групповое проставление отметок	-	+	+	+
Автоматический вывод текущего занятия преподавателя	-	-	-	+
Автоматическое формирование списков групп	-	-	+	+
Троичная логика	-	-	-	+
Разделение на подгруппы	-	+	-	+

В пункте 1.2 было установлено, что для достижения цели целесообразно будет использовать существующие в информационной среде учебного

заведения инструменты. Рассмотреть аналоги, разработанные другими учебными заведениями для личного пользования, не имея личного доступа к их информационной среде, не представляется возможным. Однако, несмотря на разницу в подходах к интегрированию рассмотренных систем в образовательный процесс, вышеупомянутые аналоги, наряду с существующей системой отметки посещаемости, обзор которой приводился в пункте 1.1 данной работы, позволяют сформировать набор требований к разрабатываемой системе.

1.4 Формирование функциональных и нефункциональных требований к системе

На основе вышесказанного можно составить следующие требования к разрабатываемой системе. Функциональные требования:

а) Возможность отмечать посещение/непосещение студентов на занятиях;
б) Возможность поддержки троичной логики для отметок посещаемости: присутствовал, отсутствовал, не отмечен (по умолчанию);
в) Возможность отмечать сдачу лабораторных и практических работ;
г) Возможность редактирования комментария к занятию;
д) Возможность формирования актуального списка студентов для занятия, которое проходит в текущий момент времени;
е) Возможность разделения групп на подгруппы преподавателями;
ж) Возможность руководителями подразделений просмотра статистики заполнения отметок;
з) Возможность фиксации действий преподавателей;
и) Возможность просмотра истории изменений для контроля со стороны деканата;
к) Возможность просмотра статистики посещаемости и активности преподавателей.

Нефункциональные требования:

а) Минимизация сетевых запросов;

б) Корректная синхронизация данных между клиентской и серверной частями;
в) Разграничение прав доступа;
г) Интуитивно понятный интерфейс, схожий с существующей системой;
д) Совместимость с устаревшими web-браузерами;
е) Минимизация числа действий для выполнения типовых операций (поиск нужной группы, отметка посещаемости на текущем занятии);
ж) Возможность добавления новых функций в будущем без изменения архитектуры.

Нефункциональные требования вытекают главным образом из базовых требований к качественному программному обеспечению, таких как расширяемость, производительность, надежность и т.д. Учитывая специфику использования разрабатываемой системы, можно предположить, что она будет использоваться преподавателями на старых компьютерах, стоящих в аудиториях, имеющих низкую скорость интернета. Низкой в современных реалиях принято считать скорость 1 Мбит в секунду с учетом сетевой задержки в 200мс. Значит необходимо учесть это в разрабатываемой системе и по возможности минимизировать количество сетевых запросов и их наполнение. Также следует учесть, что на старых компьютерах могут быть устаревшие версии web-браузеров, так что при разработке необходимо свести к минимуму использование нововведений в язык клиентской части системы.

1.5 Обоснование инструментария разработки

Программные инженеры на протяжении всей истории разработки программного обеспечения создавали инструменты для упрощения своей деятельности и пользовались ими. Система будет представлять собой раздел сайта университета. На данный момент в разработке сайтов, помимо развитых языков программирования для серверной и клиентской частей, существуют готовые программные решения. Подобные программные решения называются «фреймворками», и, в отличии от библиотек, расширяющих программу

отдельными модулями, они диктуют свою собственную парадигму разработки и предоставляют все необходимые инструменты.

Фреймворки в области web-разработки делятся на решения для клиентской и серверной частей. В качестве примеров фреймворков можно выделить такие популярные из них как «Vue.js», «React», «Django», «Laravel» и т.д. Первые два из названных являются решениями для клиентской части сайта, они позволяют создавать пользовательский интерфейс и логику клиентской части и располагать на отдельном сервере. Если в стандартной клиент-серверной архитектуре сервер отвечает и за логику, и за формирование клиентского представления (HTML страницы), то с использованием фреймворков клиентской части можно расположить модуль формирования HTML страниц на отдельном сервере. Такой подход улучшает модульность системы. Фреймворки серверной позволяют настроить ответы сервера на определенные сетевые запросы.

Вопрос о выборе платформы в данном случае заключается в том, целесообразно ли использовать готовые программные решения для разрабатываемой системы и выделять для нее отдельный сервер, или стоит использовать уже интегрированные инструменты. Не стоит забывать, что сайт университета тоже использует некоторые готовые программные решения.

Сайт университета использует фреймворк «Zend Framework», с использованием инструментов этого фреймворка написан весь существующий функционал сайта. Выделение отдельного сервера для разрабатываемой подсистемы будет сильно контрастировать с архитектурой остальных модулей. Этот подход приведет к усложнению архитектуры, так как помимо сервера самого сайта придется еще обслуживать сервер подсистемы. Кроме того, это усложнит поддержку самой разрабатываемой системы, так как она будет использовать технологии, отличающиеся от тех, которые были необходимы для обслуживания сайта ранее.

Таким образом, в результате анализа было установлено, что целесообразно использовать инструменты, которые уже интегрированы в цифровую среду университета, а именно «Zend Framework».

Клиентская часть проекта будет реализована с использованием стандартного для веб-разработки стека технологий. Для вёрстки пользовательского интерфейса применяются HTML и CSS, что обеспечивает согласованность с существующими модулями сайта.

Генерация HTML-страниц осуществляется на стороне сервера с использованием языка PHP. Данный подход позволяет предварительно формировать контент страницы, включая в него необходимые данные из бизнес-логики, перед отправкой конечному пользователю.

Интерактивность и динамическое поведение элементов интерфейса реализуются на языке JavaScript, который является основным и наиболее распространенным языком программирования для выполнения кода на стороне клиента в браузере.

Для администрирования серверной части и развертывания файлов проекта будет применяться FTP-клиент FileZilla. Администрирование и отладка работы с базой данных будет осуществляться с помощью универсального клиента DBeaver, обеспечивающего полную поддержку используемой в информационной среде СУБД (MSSQL) и удобные средства для написания и анализа запросов.

Для написания и редактирования исходного кода выбрана среда разработки Visual Studio Code, обеспечивающая полноценную поддержку всех используемых языков программирования и технологий.

1.6 Разработка обобщенной архитектуры

Разрабатываемая система будет являться подсистемой сайта университета, а значит будет наследовать архитектуру существующих модулей.

Сайт университета реализует клиент-серверную архитектуру. Zend Framework основывается на принципах архитектурного шаблона MVC. Все

модули сайта придерживаются, согласно шаблону MVC, разделения на модель (model), представление (view) и контроллер (controller).

Модель представляет из себя программный модуль, отвечающий за обработку данных. В модели происходит обращение к базе данных, форматирование ответов, работа с кодировкой и т.д.

Представление в каждом модуле отвечает за отображение данных из модели для пользователя и реакцию на изменения модели и действия пользователя.

Контроллер связывает модель и представление между собой, интерпретируя действия пользователя и изменяя модель.

Описанная выше архитектура представлена на рисунке 1.6.1.

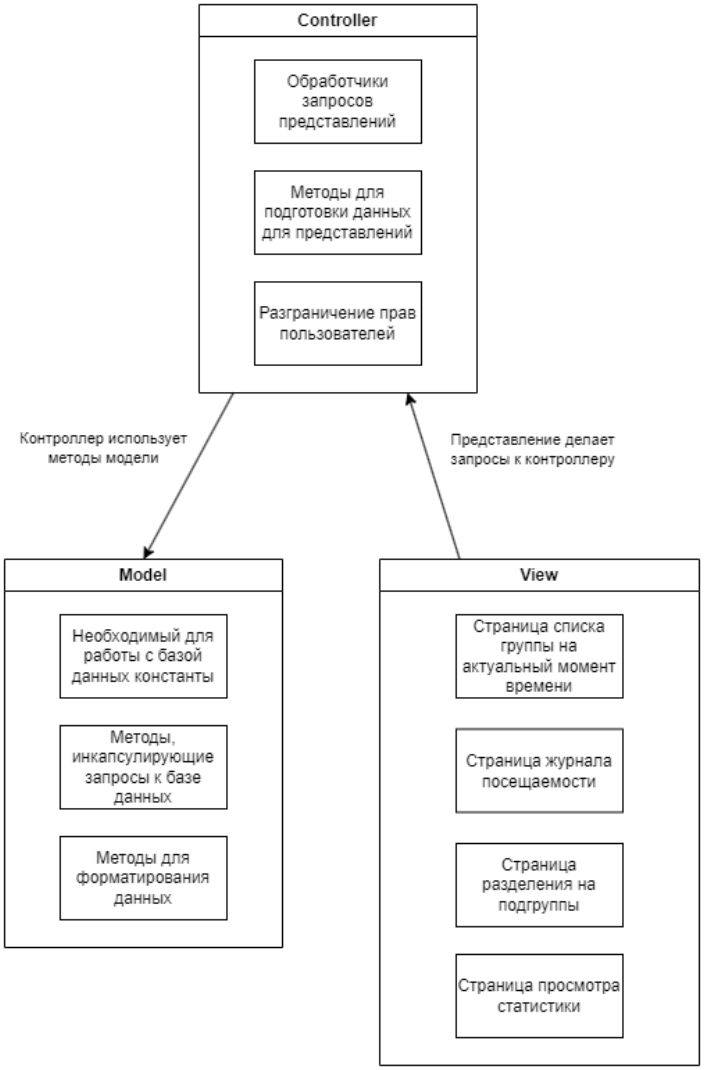

Рисунок 1.6.1 – Обобщенная архитектура разрабатываемой системы

2 ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

2.1 Определение функциональных спецификаций

Необходимо выделить конкретные возможности пользователей системы. В системе будут существовать 3 типа пользователей – Преподаватель, Студент и Руководитель подразделения.

Исходя из функциональных требований к системе из п. 1.4 настоящей работы пользователи с ролью Преподаватель должны иметь возможности по редактированию отметок посещаемости и отметок о сдаче лабораторных и практических работ, возможности по редактированию комментариев к занятиям, возможность получить список группы на текущий момент времени, а также возможность разделять студентов в группах на подгруппы.

Руководители подразделений, в связи с необходимостью контролировать своевременность заполнения электронного журнала профессорско-преподавательским составом, должны иметь возможность по просмотру отчетной статистики по конкретному преподавателю или студенту в определенных временных рамках.

Студент в разрабатываемой системе не вносит и не изменяет никаких данных. Ему должны быть предоставлены возможности по просмотру собственных отметок в электронном журнале, после авторизации в информационной среде университета через личный кабинет.

Рассмотренные возможности и их распределение между ролями пользователей в системе можно наглядно увидеть на Use-Case диаграмме (рисунок 2.1.1).

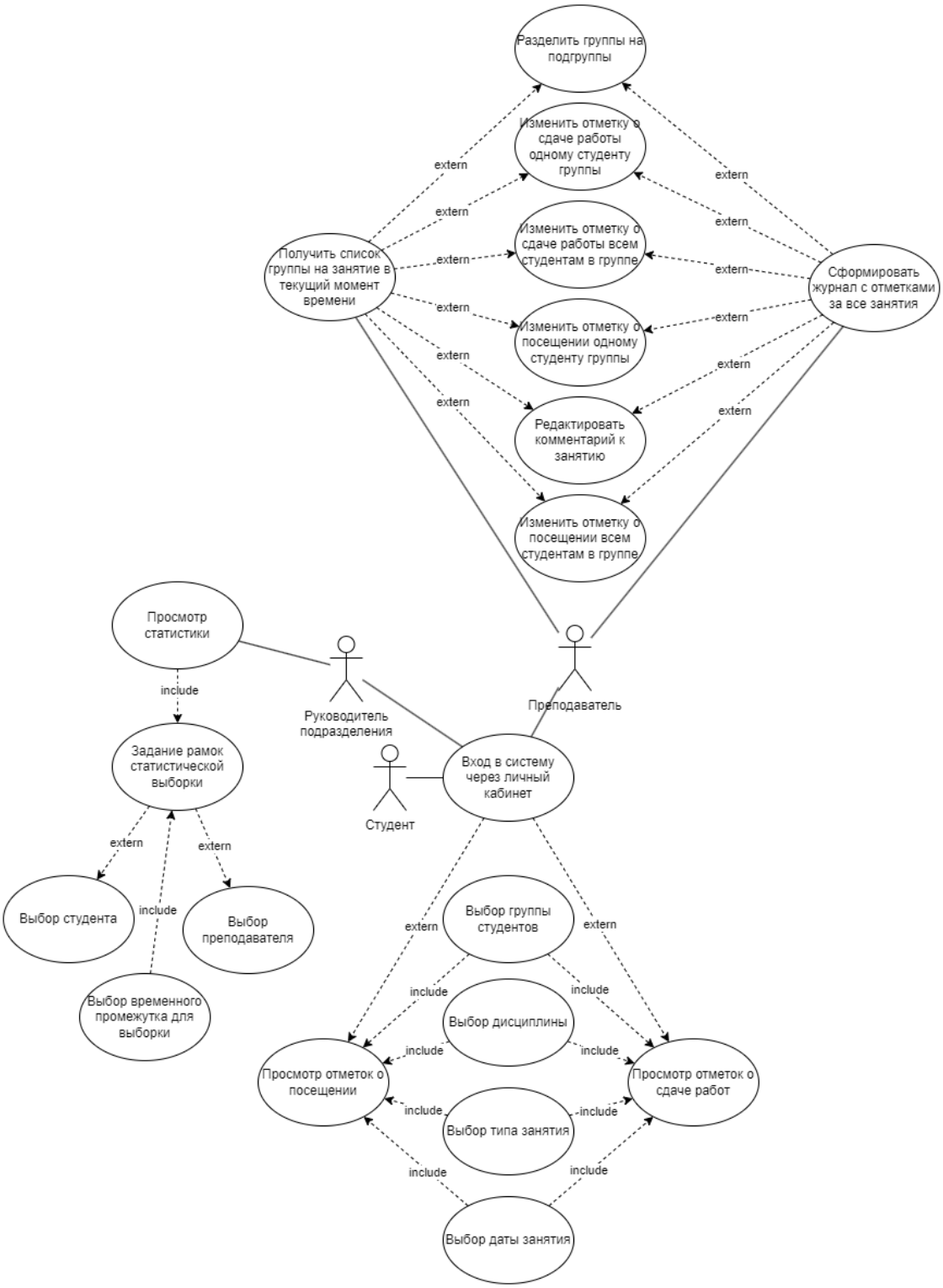

Рисунок 2.1.1 – Функциональные спецификации разрабатываемой системы

2.2 Определение поведенческих спецификаций

Для отображения изменения поведения модуля в различные моменты времени можно построить диаграмму состояний.

Диаграмма состояний призвана отразить множество устойчивых состояний, в которых может находиться система, и переходы между этими состояниями. Переходы осуществляются после определенного события, или по определенному условию.

Разрабатываемая подсистема будет включать в себя несколько обособленных web-страниц, таких как страница с актуальным на текущий момент времени занятием, страница журнала посещаемости за весь семестр, страница просмотра статистики и страница разделения учебных групп на подгруппы. Среди страниц подсистемы нет иерархии, каждой из них пользователь может пользоваться независимо от остальных. Это означает, что нельзя выделить в рамках разрабатываемой подсистемы стартовое состояние, из которого совершается переход в каждую из этих страниц. В таком случае имеет смысл рассматривать каждую страницу как отдельную подсистему, с набором состояний и переходов между ними.

Основным состоянием любой страницы является «Ожидание действий пользователя». Большую часть времени система ожидает действий пользователя, и в зависимости от них выполняются переходы в другие состояния. Система не должна заканчивать свою работу до тех пор, пока пользователь не выйдет из нее самостоятельно, соответственно после перехода из состояния «Ожидание действий пользователя» система однажды обязательно вернется в него. Выход из системы так же происходит из вышеупомянутого состояния.

На странице журнала посещаемости за семестр преподавателю необходимо выбрать характеристики занятия, по которому он собирается просматривать посещаемость. Выбор каждой из характеристик сопровождается двумя типовыми состояниями. Первое из них представляет из себя загрузку списка доступных вариантов характеристики, таких как список преподаваемых дисциплин, список учебных групп и список типов занятий. После загрузки данных с сервера пользователю показываются

соответствующие варианты для выбора. После совершения выбора система возвращается в состояние «Ожидание действий пользователя».

После того как выбраны все характеристики занятия осуществляется загрузка данных по посещаемости и отметкам. После окончания загрузки данные формируют читабельную таблицу и отображаются в таком виде пользователю.

Изменение отметки о посещаемости и изменение оценки за сданную работу очень похожи между собой, поэтому можно объединить их. После формирования таблицы с посещаемостью и нажатия на кнопку изменения отметки или оценки система переходит в состояние ожидание успешного изменения отметки или оценки. После успешного изменения и ответа сервера система возвращается в состояние «Ожидание действий пользователя» и изменяет внешний вид таблицы посещаемости.

Изменение комментариев к занятию схоже с проставлением отметок и оценок. После изменения текста комментария происходит переход в состояние ожидания ответа сервера, а после получения ответа происходит возврат в «Ожидание действий пользователя» с сохранением изменений как на сервере, так и на странице пользователя.

Визуализация вышеперечисленных состояний и переходов между ними представлена в виде диаграммы состояний и переходов (рисунок 2.2.1).

Страница с актуальным на текущий момент времени занятием очень похожа структурой своих состояний и переходов на страницу журнала посещаемости за семестр. Это обусловлено тем, что в случае страницы журнала за семестр, преподаватель самостоятельно выбирает дисциплину, группу и тип занятия, чтобы увидеть таблицу посещаемости, тогда как в случае страницы с актуальным на текущий момент времени занятием эта работа возложена на сервер, который выберет необходимые характеристики в зависимости от времени запроса.

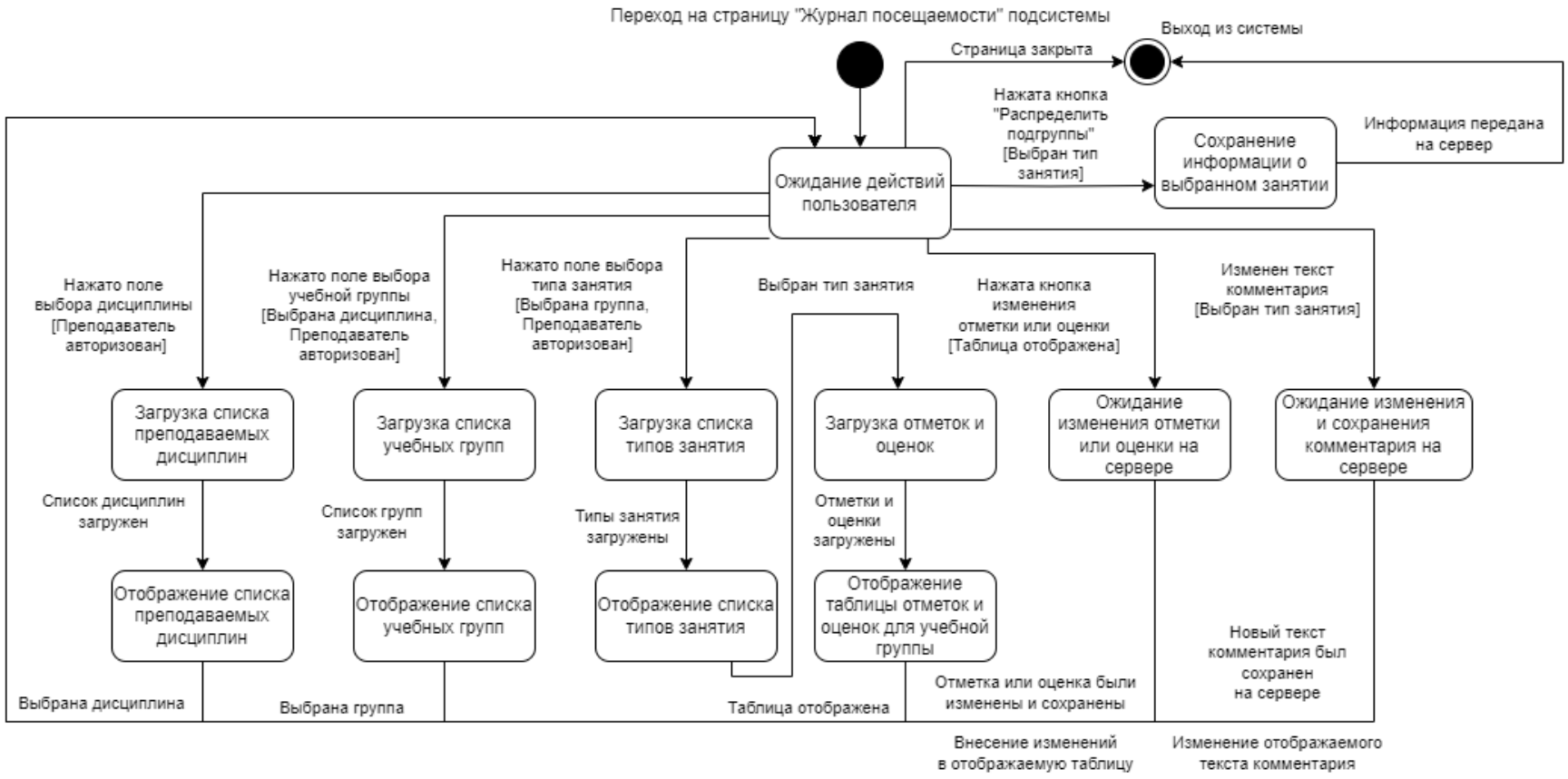

Рисунок 2.2.1 – Множество состояний и переходов страницы «Журнал посещаемости за семестр»

Также со страницы журнала после выбора характеристик занятия можно будет перейти к разбиению на подгруппы. Это позволит сохранить выбранные характеристики занятий и перейти сразу к разделению.

На странице разделения на подгруппы по аналогии с вышеупомянутыми страницами происходит выбор дисциплины и группы, либо они передаются с другой страницы. После того как определяется нужные группа и дисциплина отображается список студентов, которых можно разбивать на подгруппы (рисунок 2.2.2).

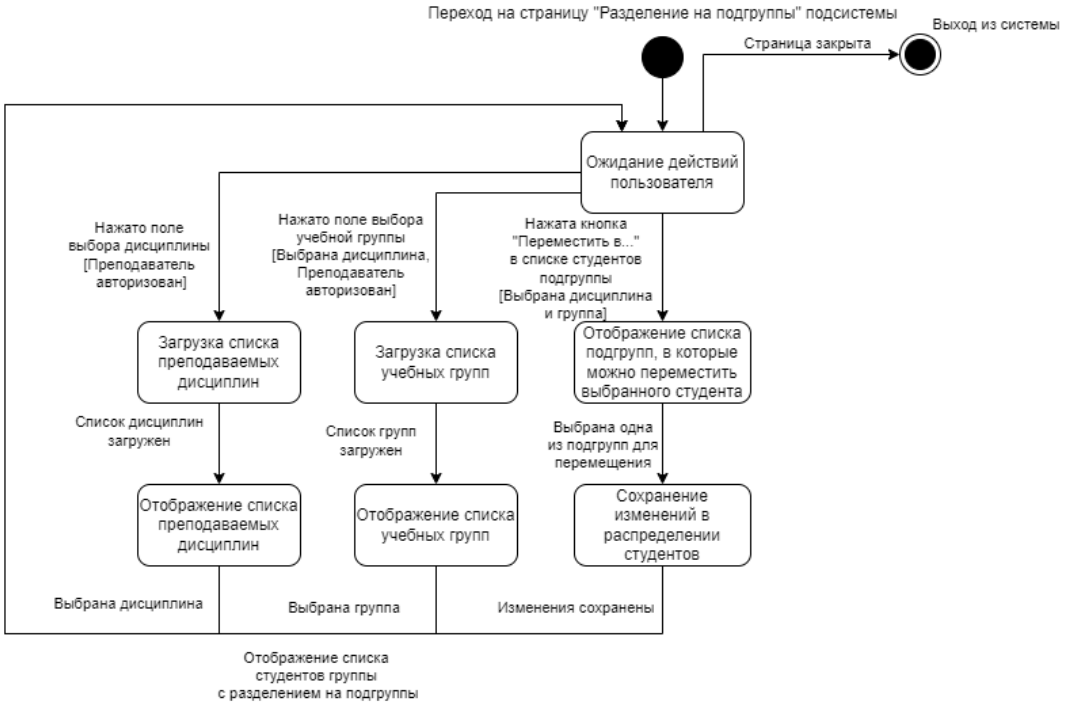

Рисунок 2.2.2 – Множество состояний и переходов страницы «Разделение на подгруппы»

Страница отображения статистики использует аналогичный метод выбора преподавателя. Для сбора данных по конкретному студенту используется номер его зачетки. Статистику нельзя получить без задания временных рамок, поэтому задание временных рамок является обязательным условием для перехода к состоянию загрузки данных с сервера (рисунок 2.2.3).

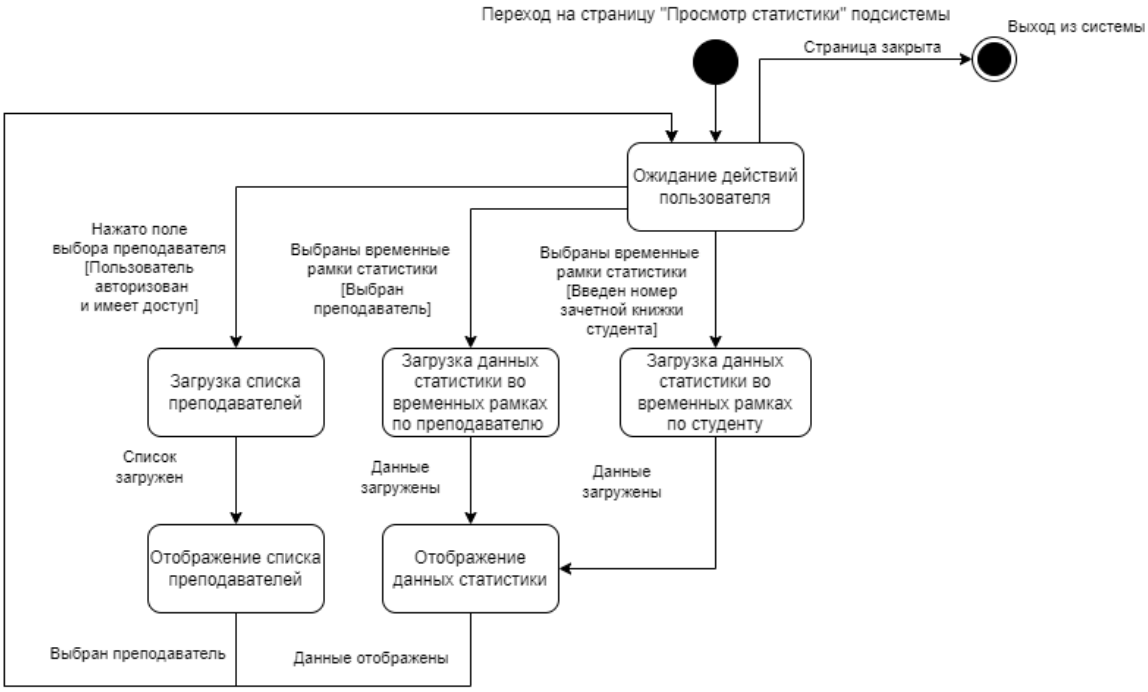

Рисунок 2.2.3 - Множество состояний и переходов страницы «Статистика»

2.3 Определение информационных спецификаций

Информационные спецификации можно описать с помощью сущностей и их взаимосвязей.

К ключевым сущностям системы можно отнести: «Дисциплина», «Сотрудник», «Группа», «Студент». Все эти сущности можно назвать «сущностями-справочниками». Они хранят данные, которые будут относительно нечасто меняться. Они связаны между собой процессом проведения занятия.

Сущность "Расписание" является центральной операционной сущностью системы. Она связывает между собой сущности "Дисциплина", "Сотрудник" и "Группа", определяя конкретное проводимое занятие. Каждая запись в "Расписании" содержит атрибуты места, времени и типа занятия, а также включает ссылки на соответствующие экземпляры справочников.

Сущность «Отметка» содержит данные о посещениях студентов и о сдачах ими практических и лабораторных работ. Каждая запись в этой сущности связана с конкретным студентом, группой, сотрудником и занятием.

Можно заметить, что все эти данные можно получить из сущности «Расписание», а такое содержание сущности «Отметка» приведет к избыточным связям и дублированию данных. Здесь важно отметить, что сущность «Отметка» должна сохранять информацию, актуальную на момент проставления отметок. Расписание может измениться, одно занятие могут вести несколько преподавателей, соответственно ссылаться на сущность «Расписание» не является надежным вариантом. Не смотря на отступления от общепринятых норм проектирования баз данных, в данном случае такой компромисс является приемлемым и эффективным решением, так как помимо более надежного сохранения информации обеспечит снижение сложности и времени выполнения запросов, за счет агрегации всех нужных для запросов данных в одной сущности.

Сущность «Журнал изменений» содержит информацию о времени создания и последнего изменения каждой отметки.

Сущность «Подгруппа» моделирует гибкое разделение академических групп для практических занятий. Она позволяет каждому преподавателю независимо формировать состав подгрупп по своей дисциплине. Сущность хранит связи между студентами, дисциплиной и преподавателем, определяя принадлежность студента к конкретной подгруппе.

Сущности «Дисциплина», «Сотрудник», «Группа», «Студент» и «Расписание» являются внешними по отношению к разрабатываемой системе. Они уже существуют в информационной среде университета независимо от работы разрабатываемой системы, и доступны разрабатываемой системе только для чтения. С остальными из перечисленных сущностей разрабатываемая система работает непосредственно, добавляет новые данные и изменяет существующие.

Представить сущности системы и связи между ними можно с помощью ER-диаграммы (рисунок 2.3.1).

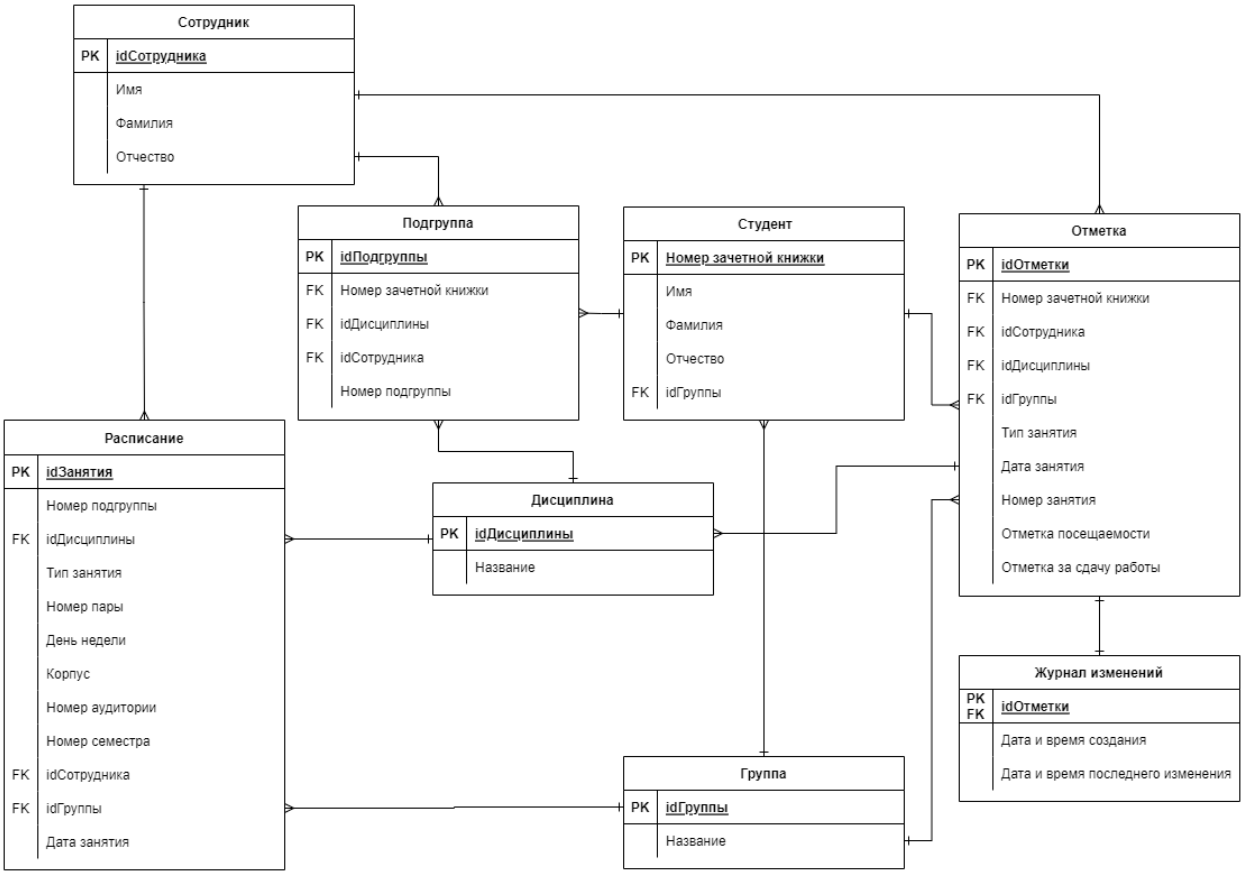

Рисунок 2.3.1 – Сущности системы и связи между ними

3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1 Проектирование структуры программного обеспечения

Структура разрабатываемой системы, как упоминалось выше, построена с использованием шаблона проектирования MVC, что продиктовано обстоятельствами, описанными в пункте 1.6 настоящей работы.

Структура разрабатываемой системы, внешне продолжающая идею архитектуры из пункта 1.6, изображена на рисунке 3.1.1.

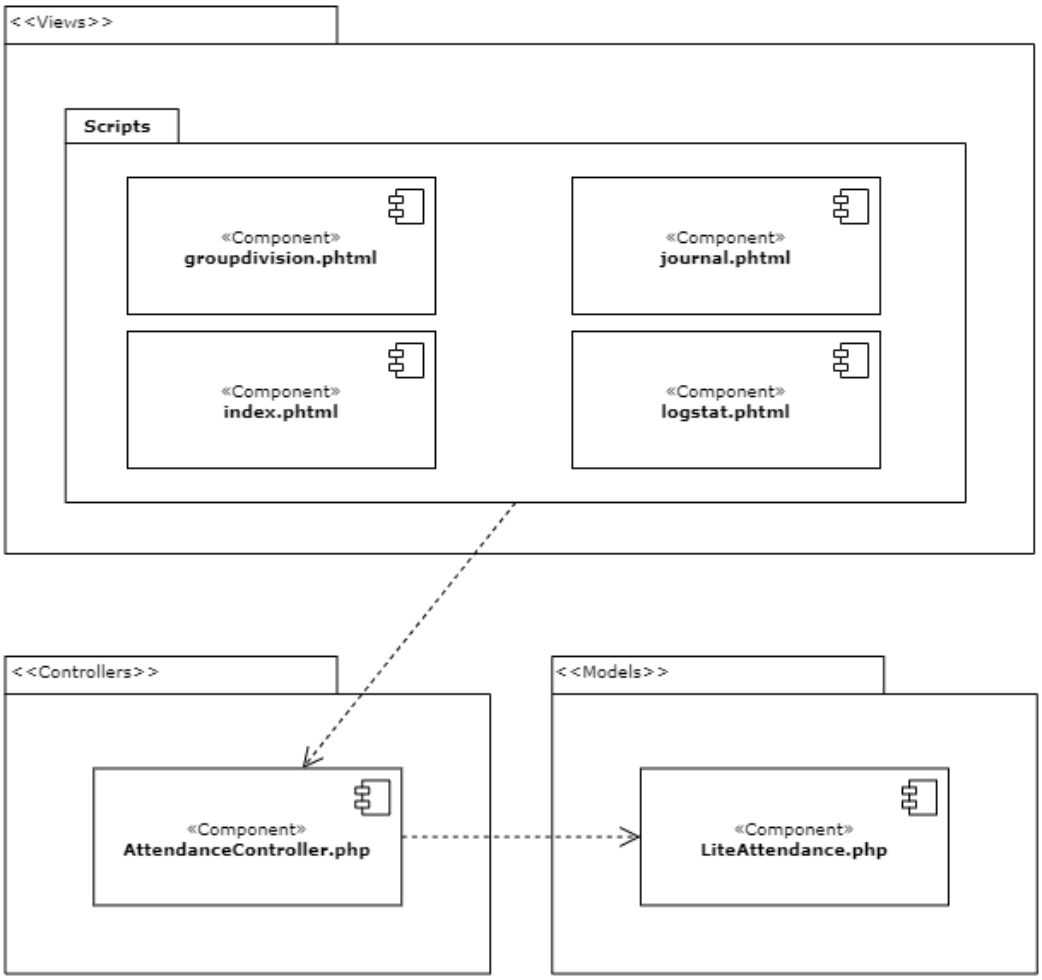

Рисунок 3.1.1 – Структура разрабатываемой системы

Модель «LiteAttendance.php» занимается всем, что связано с обработкой данных из базы данных в системе. Она инкапсулирует в себе функционал по доступу к данным, формированию запросов к базе данных, форматированию данных, такому как изменение кодировки и переупаковка в другие структуры данных.

Контроллер «AttendanceController.php» отвечает за связь внешних представлений с моделью. Контроллер определяет права доступа пользователя, использует модель для формирования данных, подготавливает данные для представлений в синхронных и асинхронных запросах.

Представления отвечают за отображение каждой конкретной страницы системы. В представлениях прописана логика клиентской части, запросы к API, предоставляемому контроллером, задан внешний вид страниц.

3.2 Проектирование компонентов

Код компонентов будет содержаться в функциях и методах классов.

Составляющие компонента «LiteAttendance» можно видеть на рисунке 3.2.1.

Рисунок 3.2.1 – Компонент LiteAttendance

Составляющие компонента «AttendanceController» представлены на рисунке 3.2.2.

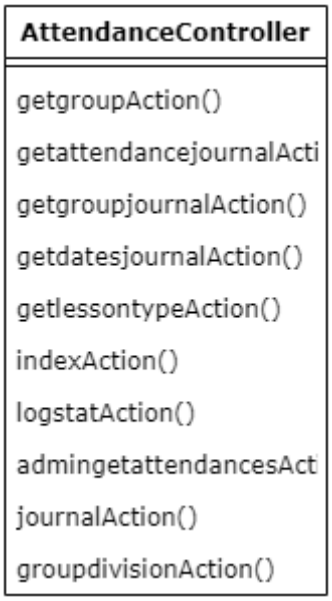

Рисунок 3.2.2 – Компонент AttendanceController

Составляющие компонентов представлений представлены на рисунке 3.2.3.

Рисунок 3.2.3 – Компоненты представлений системы

Ниже приведены пояснения к основным составляющим компонентов:

1) Модель «LiteAttendance»:
   а) Метод «get_attendances_encoded» – получает из базы данных информацию о посещаемости группы на указанное занятие по параметрам. Возвращает данные в массиве в кодировке UTF-8;
   б) Метод «getgroupjournal_encoded» – получает из базы данных набор групп, которым преподается указанная дисциплина указанным преподавателем. Возвращает данные в массиве в кодировке UTF-8;
   в) Метод «get_lesson_type_encoded» – получает из базы данных набор типов занятий, которые есть по указанной дисциплине у указанной группы. Возвращает данные в массиве в кодировке UTF-8;
   г) Метод «getdatesjournal_encoded» – получает из базы данных даты занятий на семестр вплоть до текущего момента времени и комментарии к занятиям. Возвращает данные в массиве в кодировке UTF-8;
   д) Метод «getattendancejournal_encoded» – получает из базы данных данные для формирования таблицы журнала. Возвращает данные в массиве в кодировке UTF-8;
   е) Метод «addAttendanceMark» – добавляет отметку посещаемости в базу данных, либо меняет существующую.
   ж) Метод «addDoneMark» – добавляет отметку сдаче работы в базу данных, либо меняет существующую;
   з) Метод «getLessonsList» – получает из базы данных набор занятий преподавателя на указанную дату. Если дата не указана, то получает набор занятий на сегодня;
   и) Метод «getEmployeeAttendanceInfo» – получает из базы данных количество отметок, сделанных преподавателем на занятиях в указанный период времени;

к) Метод «logAttendance» – добавляет в таблицу логирования информацию о созданной или измененной отметке;

л) Метод «searchEmployeeByFIO» – ищет в базе данных сотрудника по ФИО;

м) Метод «getStudentStatistics» – ищет в базе данных информацию о посещаемости студента в указанный период времени;

н) Метод «getMultigroupAmount» – ищет в базе данных количество отметок на потоковой лекции;

о) Метод «getSubjectListForEmployee» – получает из базы данных набор дисциплин, которые ведет указанный перподаватель;

п) Метод «mergeStudentsToSubGroups» – присваивает всем студентам группы, не определенным в подгруппы ранее, первый номер подгруппы;

р) Метод «updateSubGroup» – обновляет распределение по подгруппам;

с) Метод «getStudentListForSubGroupDisivion» – получает из базы данных набор студентов для разбиения на подгруппы. При необходимости осуществляет вызов «mergeStudentsToSubGroups».

2) Компонент AttendanceController:

а) Метод «getgroupAction» – обращается к модели для запроса посещаемости группы по заданным параметрам с использованием метода модели «get_attendances_encoded»;

б) Метод «getattendancejournalAction» – аналог метода «getgroupAction» для страницы журнала посещаемости;

в) Метод «getgroupjournalAction» – получает список групп для преподавателя и дисциплины используя метод модели «getgroupjournal_encoded»;

г) Метод «getdatesjournalAction» – получает даты занятий на семестр используя метод модели «getdatesjournal_encoded»;

д) Метод «getlessontypeAction» – получает тип занятия используя метод модели «get_lesson_type_encoded»;
е) Метод «indexAction» – обработчик главной страницы системы;
ж) Метод «logstatAction» – обработчик страницы просмотра статистики;
з) Метод «admingetattendancesAction» – получает информацию о заполнении журнала посещаемости;
и) Метод «journalAction» – обработчик страницы журнала посещаемости;
к) Метод «groupdivisionAction» – обработчик страницы разделения на подгруппы.

3.3 Проектирование структур данных

В проектируемой системе используется реляционная база данных. База данных сайта университета поддерживается системой управления базами данных Microsoft SQL Server.

Как описывалось в п. 2.3 настоящей работы, для разрабатываемой системы большинство сущностей являются внешними и используются только для чтения. Изменять и добавлять данные система будет для сущностей «Подгруппа», «Журнал изменений» и «Отметка». Таблица «Отметка» уже реализована и использовалась в предшествующей системе отметок посещаемости, описанной в п. 1.2. Соответственно, необходимо спроектировать структуры данных для таблиц «Журнал изменений» и «Подгруппа».

«Журнал изменений» хранит в себе информацию об изменении преподавателем отметки о посещении или сдаче работы студентов. Для этого необходимо сохранять ссылку на конкретную отметку, что возможно сделать по идентификатору записи из таблицы «Отметка». В базе будет храниться информация о времени создания отметки. Так же необходимо сохранять время последнего изменения отметки.

В реляционных базах данных свойства записей в таблицах определяются типом данных для каждой колонки таблицы. Для сохранения временной отметки в Microsoft SQL Server могут использоваться типы «date», который сохраняет лишь дату, а также «datetime», сохраняющий помимо даты еще и время суток с точностью до миллисекунд. Для целей разрабатываемой системы необходимо использовать тип «datetime». Тип ссылки на отметку будет таким же, как и у идентификатора в таблице «Отметка», а именно «int». Для каждой созданной отметки будет храниться ровно 1 запись в таблице «Журнал изменений», таким образом нет необходимости выделять суррогатный первичный ключ для этой таблицы. Внешний ключ к таблице «Отметка» будет первичным для таблицы «Журнал изменений».

Таблица «Подгруппа» представляет собой набор записей, характеризующих распределение на подгруппы, созданное каждым преподавателем. Для того, чтобы один и тот же студент мог содержаться в разных подгруппах для разных дисциплин и преподавателей, необходимо в каждой записи таблицы «Подгруппа» хранить ссылки на таблицы «Студент», «Сотрудник» и «Дисциплина». Помимо этого, также необходимо хранить номер присваиваемой студенту подгруппы.

Таким образом, необходимо хранить три ссылки на другие таблицы типа «int», номер подгруппы, который не может быть больше, чем студентов в группе, для чего однозначно хватит диапазона значений типа «smallint», а также необходимо выделить идентификатор записи. Для этого будет выделен суррогатный ключ типа «int».

Основой взаимодействия с пользователем в разрабатываемой системе будут асинхронные запросы к серверу. Такие запросы позволят обновлять содержимое web-страницы без ее перезагрузки, что улучшит пользовательский опыт и эффективность работы с системой.

Технология AJAX, поддерживаемая всеми современными браузерами, позволяет обмениваться JSON-объектами между сервером и клиентом. JSON (JavaScript Object Notation) – это нотация записи информационных объектов в

формате «ключ – значение». Большинство современных языков программирования имеют встроенную поддержку JSON.

Клиент будет отправлять на сервер запрос при действиях пользователя. Например, при попытке отметить студента будет отправляться запрос на сервер, содержащий идентификатор занятия, номер группы, номер зачетки студента. Сервер отправит клиенту ответ, в котором обозначит успешность операции. Клиент по содержанию ответа сервера сможет изменить интерфейс, например, поставить отметку студента за посещение как «присутствовал».

В качестве примера можно рассмотреть объект студента, который используется сервером для формирования списка группы в ответ на запрос посещаемости конкретного занятия. Структуру такого объекта можно увидеть в таблице 3.3.1.

Таблица 3.3.1 – Структура JSON-объекта студента

Название поля	Тип	Пояснение
fio	string	ФИО студента
idA	int	Номер записи об отметке в базе данных
mark	bool	Отметка присутствия
markDone	int	Оценка за работу
numberStudBook	string	Номер зачетной книжки студента
numberSubGroup	int	Номер подгруппы

Пример объекта студента представлен на рисунке 3.3.2.

```
▼ Array(37) i
  ▼ 0:
      fio: "Абдумажитов Шахзоджон Рахимжанович"
      idA: "7732370"
      mark: "1"
      markDone: ""
      numberStudBook: "222369"
      numberSubGroup: "2"
    ▶ [[Prototype]]: Object
  ▼ 1:
      fio: "Архипин Кирилл Сергеевич"
      idA: "7732361"
      mark: "0"
      markDone: ""
      numberStudBook: "223118"
      numberSubGroup: "2"
    ▶ [[Prototype]]: Object
  ▼ 2:
      fio: "Афанасьев Федор Николаевич"
      idA: "7732362"
      mark: "1"
      markDone: ""
      numberStudBook: "220884"
      numberSubGroup: "2"
    ▶ [[Prototype]]: Object
```


Рисунок 3.3.2 – Пример JSON-объектов студента

3.4 Проектирование алгоритмов

Основным алгоритмом в системе будет выступать запрос списка группы на определенный момент времени (рисунок 3.4.1). Алгоритм начинается с получения идентификатора пользователя и параметров запроса, переданных от клиента. Эти параметры включают в себя данные о занятии, дате, на которую запрашивается список группы и т.д.

Алгоритм начинается с проверки прав доступа пользователя. Если пользователь авторизован и имеет права доступа, то алгоритм продолжает работу, иначе пользователь будет перенаправлен на главную страницу сайта. В системе возможна ситуация, в которой пользователь будет запрашивать список группы не для своих занятий. Например, руководитель подразделения захочет посмотреть, как преподаватель отметил студентов на конкретном занятии. В таком случае идентификатор пользователя, делающего запрос, и идентификатор пользователя, отмечавшего студентов на занятии, будут отличаться. Этот момент важно учитывать в проверке прав доступа.

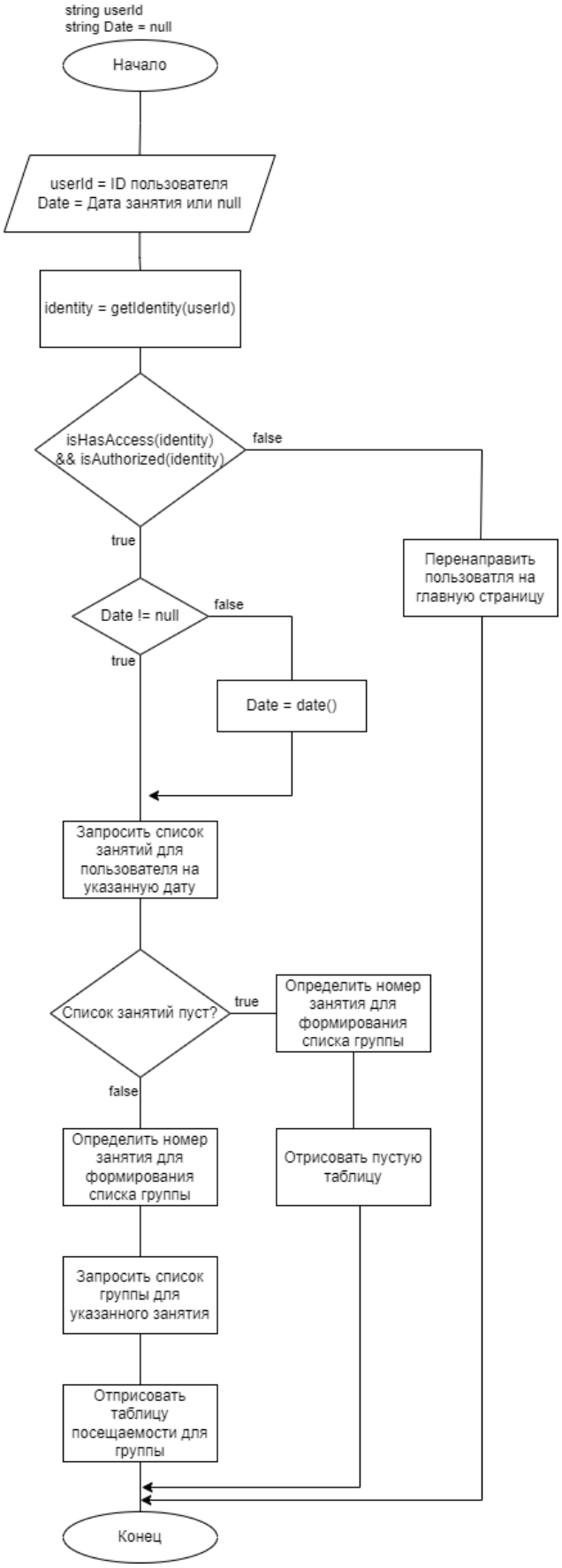

Рисунок 3.4.1 – Алгоритм формирования страницы журнала посещаемости для группы на текущую дату

После проверки прав доступа необходимо определить, на какую дату сформировать список группы. Это определяется параметрами запроса. Исходя из них становится ясно, пытается пользователь получить список занятий на сегодня, или на другую предшествующую дату.

Затем запрашивается из базы данных список занятий для пользователя на дату, определенную в предыдущем абзаце. Список занятий отсортирован по времени начала занятия.

Затем определяется, на какое именно занятие из списка необходимо получить список группы. Это может быть занятие, которое происходит в данный момент времени, либо это может быть любое другое занятие в произвольную дату. Это определяется параметрами запроса.

Конкретизированный алгоритм выбора нужного занятия из списка представлен на рисунке 3.4.2. Получив список занятий на указанную дату необходимо определить, искать нужное занятие исходя из текущего серверного времени суток, либо по номеру из параметров запроса.

После того как это определено, посредством цикла в обоих случаях находится нужное занятие. Важно отметить, что для снижения количества проверок при поиске последнего начавшегося занятия лучше идти с конца списка занятий. Если алгоритм будет просматривать список занятий с начала, то на каждой итерации цикла придется проверять не только началось ли текущее занятие, но и является ли оно последним начавшимся. То есть существуют ли занятия, которые уже начались, при этом находясь дальше в списке занятий. Если же идти с конца списка, то первое же занятие, которое подходит под условие «началось раньше текущего времени суток» будет искомым. Этот цикл представлен левой веткой исполнения на блок-схеме на рисунке 3.4.2.

В случае же, когда искомое занятие нужно просто идентифицировать по номеру (подразумевается номер занятия в ежедневном расписании, например, под номером 1 будут занятия, начавшиеся в 8:30, под номером 2 начавшиеся в 10:10 и т.д.), подойдет обычный перебор элементов с начала списка. Несмотря

на то, что список занятий отсортирован по номерам, не имеет смысла осуществлять оптимизированный поиск в этом списке. Количество элементов в списке занятий не может превышать восьми элементов. В самом худшем случае алгоритм совершит 8 итераций, что ничтожно мало в современных реалиях и не нуждается в оптимизации.

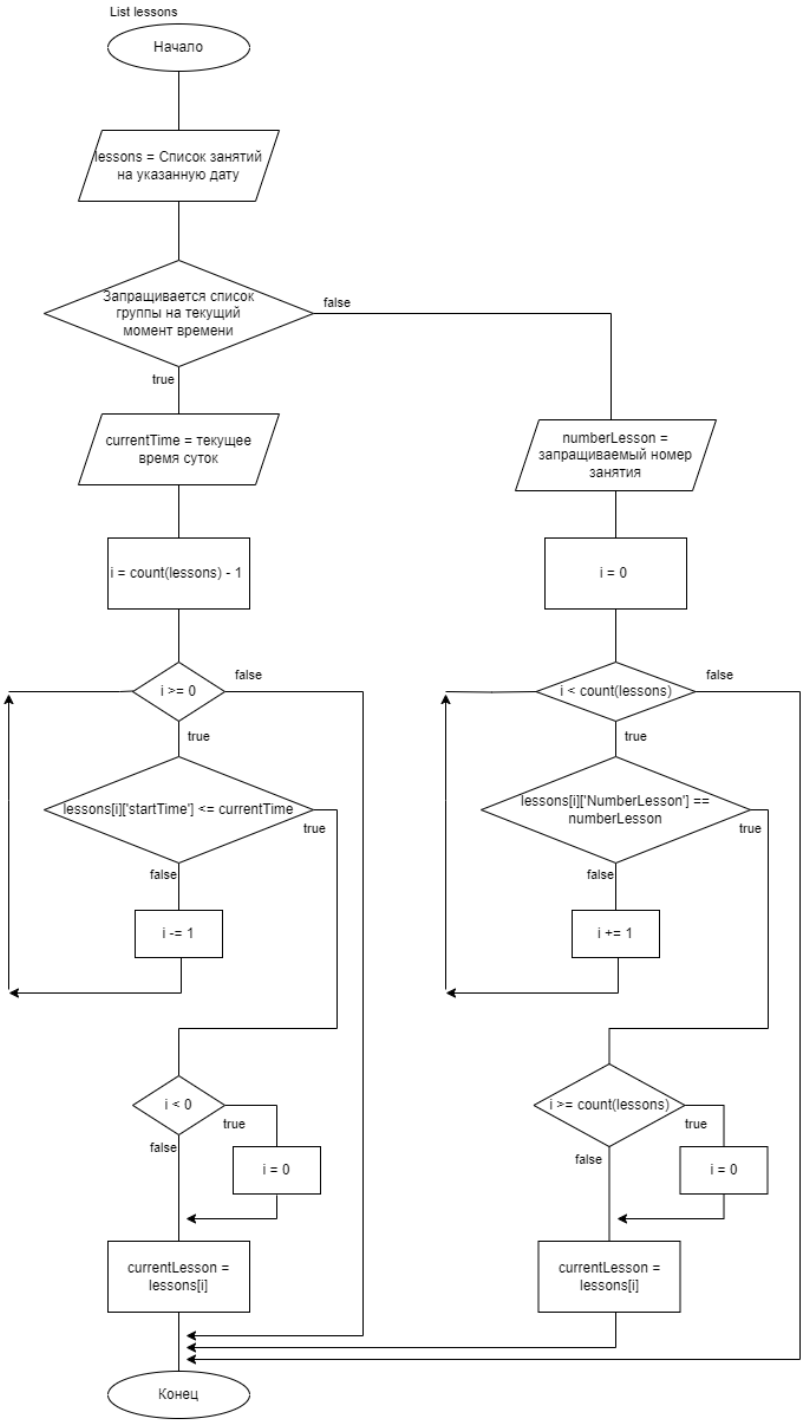

Рисунок 3.4.2 – Алгоритм поиска нужного занятия на текущий момент времени, либо по указанному номеру

Алгоритм работы кнопки «Отметить всех» представлен на рисунке 3.4.3. Этот алгоритм должен отправлять запрос на сервер для отметки всей группы как присутствовавших или отсутствовавших, в зависимости от заданной изначально переменной. Если переменная не задана, алгоритм должен вычислить, как именно отметить студентов.

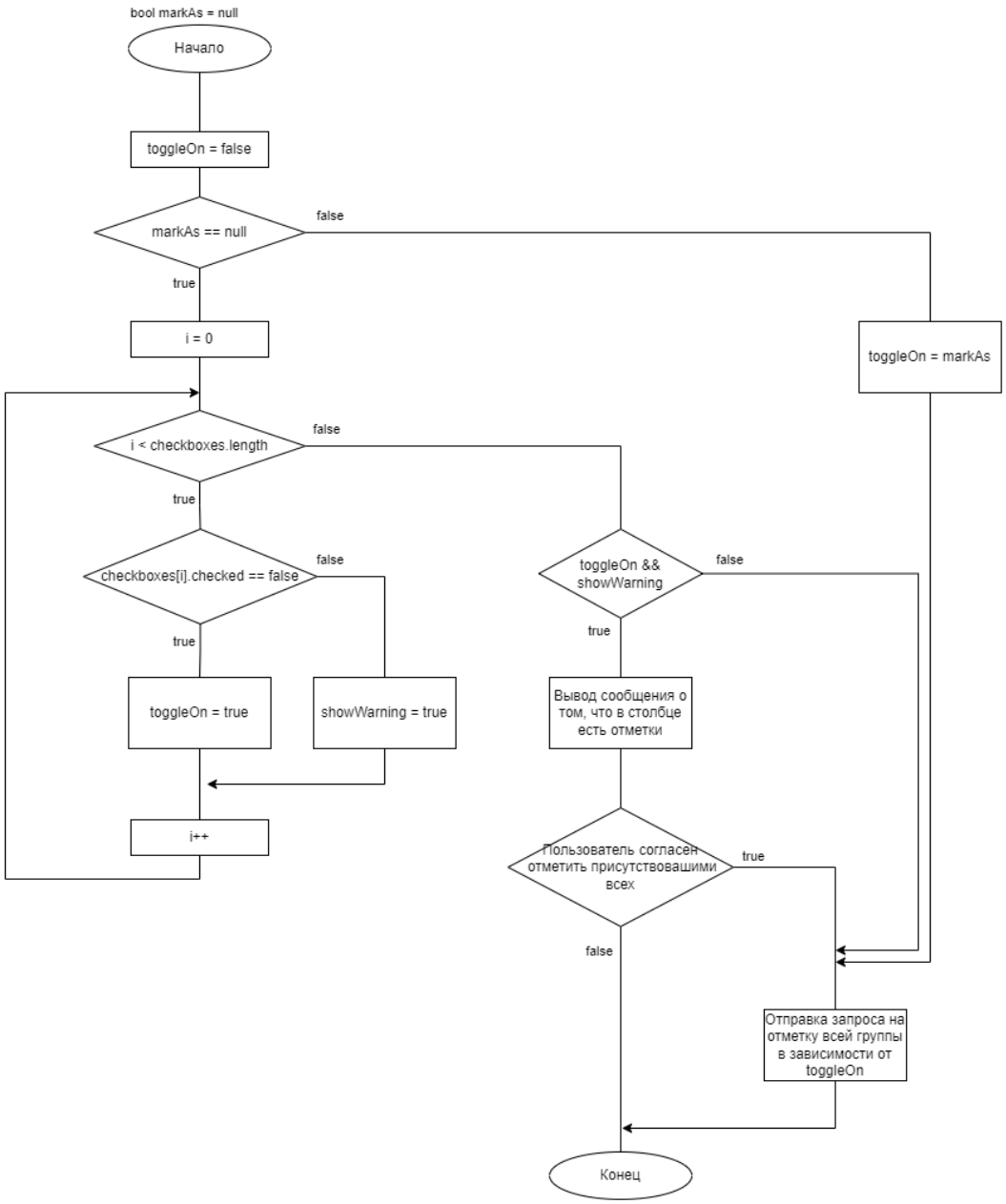

Рисунок 3.4.3 – Алгоритм работы кнопки «Отметить всех»

Алгоритм начинается с ввода переменной «MarkAs», которая является логической переменной и принимает значения «true» или «false», однако

может быть не задана и иметь значение «null». Эта переменная указывает на то, как стоит отметить всех студентов группы.

Далее инициализируется переменная «toggleOn» значением «false». Эта логическая переменная содержит в себе итоговое значение отметки группы (присутствовали или нет). По умолчанию студенты считаются отсутствовавшими.

Затем идет проверка на заданность переменной «markAs». Если переменная задана, значит вычислять как именно отметить студентов не нужно, можно сразу перейти к выполнению запроса и окончанию алгоритма.

В ином случае необходимо перебрать в цикле всех студентов группы и изучить их отметки. Если хотя бы один студент отмечен как «Отсутствовал», то пользователю будет предложено отметить всех студентов группы как присутствовавших. Если в ходе перебора будет обнаружено, что не вся группа отмечена одинаково (есть как присутствовавшие, так и отсутствовавшие студенты), то переменная-флаг «showWarning» принимает значение «true».

После окончания перебора осуществляется проверка на необходимость вывода информационного сообщения пользователю. Так как кнопка «Отметить всех» проставит одинаковые отметки для всех студентов группы, необходимо защитить пользователя от случайного нажатия и потери осознанно сделанных отметок. Это значит, что в случае, когда «toggleOn» и «showWarning» равны «true», пользователю будет предложено проставить всем студентам группы отметки «Присутствовал», затерев существующие отметки, с возможностью отказаться и отменить операцию. Если же пользователь соглашается, то в зависимости от значения «toggleOn» вся группа отмечается как присутствовавшие (если «toggleOn» равен «true») либо отсутствовавшие (в ином случае).

3.5 Проектирование интерфейсов

Главной страницей системы будет выступать страница с формированием списка группы на текущий момент времени (рисунок 3.5.1). В верхней части страницы будут располагаться навигационные кнопки. По

нажатию кнопки «Открыть журнал посещаемости» пользователь будет перенесён на страницу журнала посещаемости. Кнопки «Следующий день» и «Предыдущий день» предназначены для навигации по датам занятий.

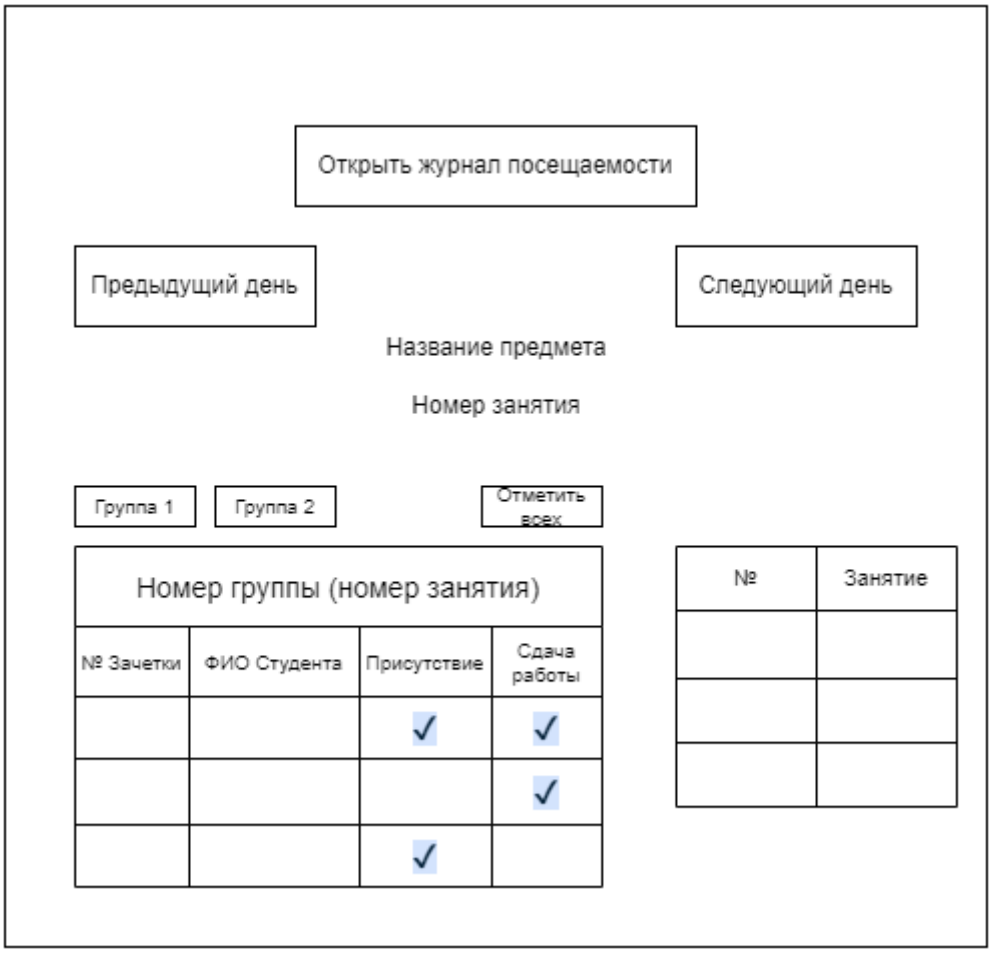

Рисунок 3.5.1 – Макет главной страницы системы

В нижней части страницы представлена таблица посещаемости и сдачи работ. Колонки таблицы содержат номер зачетной книжки студента, его фамилию, имя и отчество, его присутствие либо отсутствие, а также сдачу им работы. Присутствие либо отсутствие на занятии будет обозначаться путем отметки в соответствующем интерактивном поле выбора (checkbox-элемент). На лекциях студенты не сдают никаких работ, соответственно столбец «Сдача работ» вышеуказанной таблицы будет добавляться только для типов занятий «Лабораторная» и «Практика». Столбец «Сдача работ» будет также в своих ячейках содержать интерактивные поля выбора для отметки сдачи работы. Однако помимо обычного checkbox-элемента к нему будет приставлен

интерактивный выпадающий список с отметками за сдачу работ. Это необходимо для тех случаев, когда преподавателю недостаточно знать о самом факте сдачи работы, но еще важно сохранять оценку за нее.

Над таблицей посещаемости располагаются вспомогательные кнопки. С помощью этих кнопок можно будет сменить отображаемую группу, так как бывают потоковые лекции, на которых присутствует сразу несколько групп студентов. Кнопка «Отметить всех» пометит всех студентов выбранной группы как «Присутствующий». Эта кнопка помимо снижения времени на отметку каждого студента по одному снижает нагрузку на сеть и базу данных, так как отметить всю группу возможно всего одним запросом.

В правой части главной страницы будет расположена таблица навигации по занятиям за указанную дату. Формат отображения занятий в этой таблице будет схож с отображением на странице расписания занятий, что позволит интуитивно быстрее ориентироваться новым пользователям. По кнопкам в ячейках этой таблицы будет возможно перейти к отметке посещаемости за другое занятие на указанную дату.

Страница журнала посещаемости (рисунок 3.5.2) позволит отметить посещаемость и сдачу работ группой за все занятия по указанной дисциплине.

В верхней части страницы журнала посещаемости находятся интерактивные выпадающие списки. Списки позволяют выбрать соответственно дисциплину, группу и тип занятия, для которых необходимо сформировать таблицу журнала.

Сформированная таблица содержит ФИО студентов и колонки, содержащие интерактивные элементы выбора. Эти колонки схожи с колонками «Присутствие» и «Сдача работы» из таблицы посещаемости с главной страницы системы.

Обе вышерассмотренные страницы работают на основе асинхронных запросов, соответственно изменение отметки о посещении студентов, выбор

другой дисциплины, группы и типа занятий происходят без перезагрузки страницы.

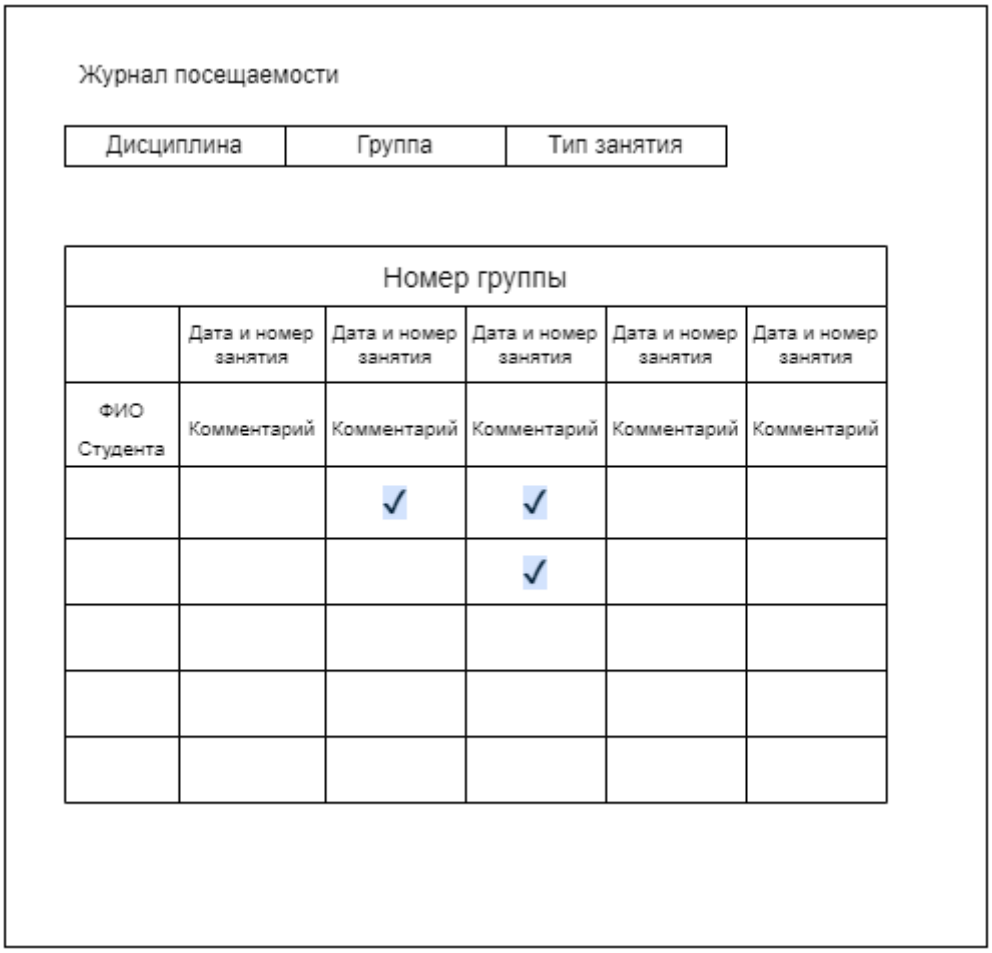

Рисунок 3.5.2 – Макет страницы журнала посещаемости

Страница разделения на подгруппы (рисунок 3.5.3) служит для деления преподавателями студенческих групп на подгруппы по своему усмотрению для каждой преподаваемой ими дисциплины.

В верхней части страницы аналогично странице журнала посещаемости располагаются интерактивные выпадающие списки для выбора дисциплины и группы.

Ниже, после выбора дисциплины и студенческой группы, отображаются таблицы со списками каждой из подгрупп. Они содержат колонки с номером зачетной книжки, ФИО студента и колонку с интерактивным элементом выбора под названием «Выделение». Между таблицами по центру располагаются три кнопки для управления составом подгрупп. Кнопка

перемещения из левой таблицы в правую, из правой таблицы в левую и кнопка сохранения. Подобный интерфейс для управления содержимым таблиц использовался в офисном программном обеспечении, таком как Microsoft Excel, Microsoft Word и т.п.

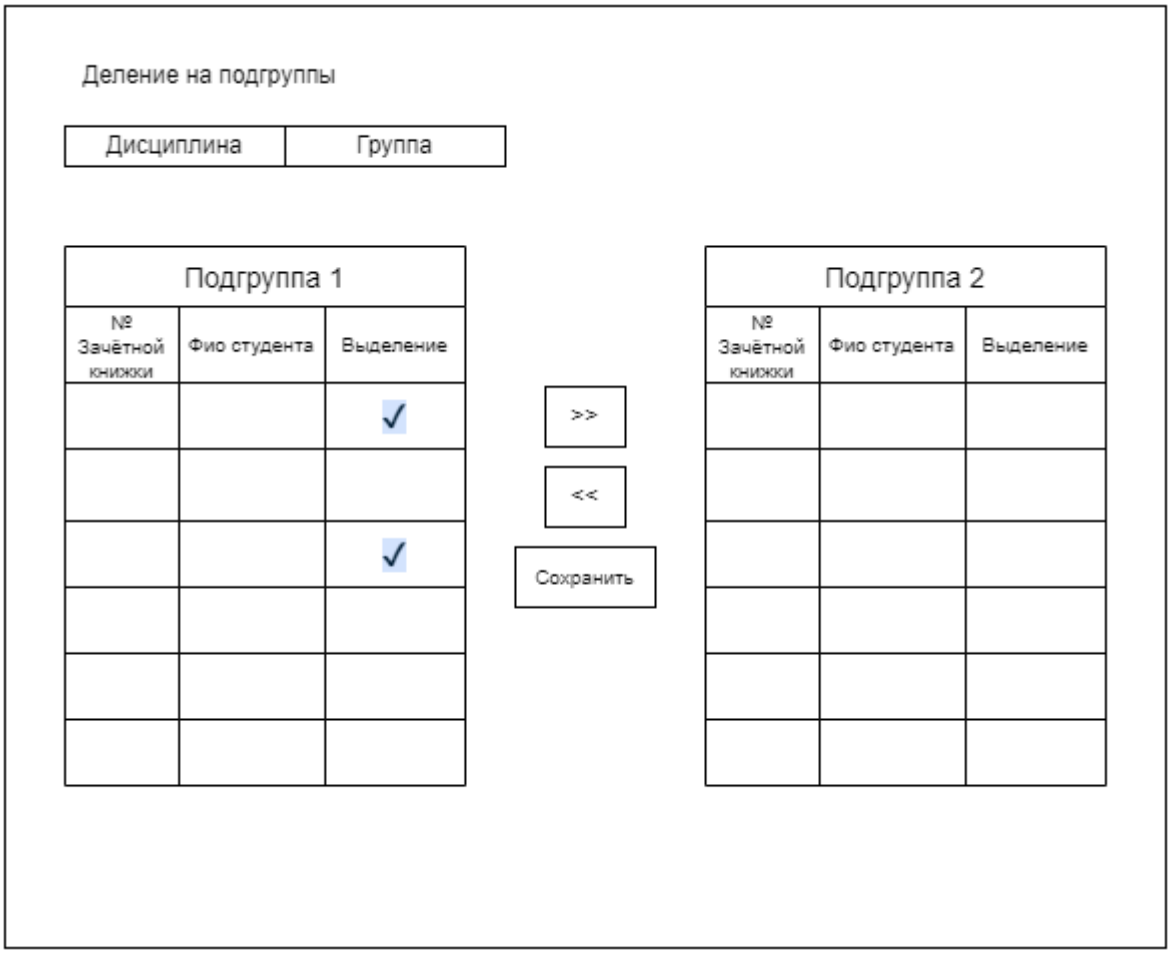

Рисунок 3.5.3 – Макет страницы разделения на подгруппы

Выделив нужных студентов одной из подгрупп для перемещения в другую, пользователь нажимает соответствующую кнопку, после чего состав подгрупп меняется. Соответствующие ячейки, выбранные пользователем, переносятся в другую таблицу, соблюдая порядок сортировки студентов по алфавиту. После всех необходимых перестановок пользователь нажимает кнопку сохранения, и новый состав подгрупп сохраняется.

Страница просмотра статистики (рисунок 3.5.4) предназначена для просмотра статистики проставления отметок преподавателем. Страница предназначена для руководителей подразделений.

В верхней части страницы расположены различные элементы выбора для формирования среза данных по нужным параметрам. Первым идет набор интерактивных полей для поиска нужного преподавателя по ФИО. Далее поле для поиска преподавателя по идентификатору в системе. Последним идет поле для ввода зачетной книжки для поиска информации о посещениях студента.

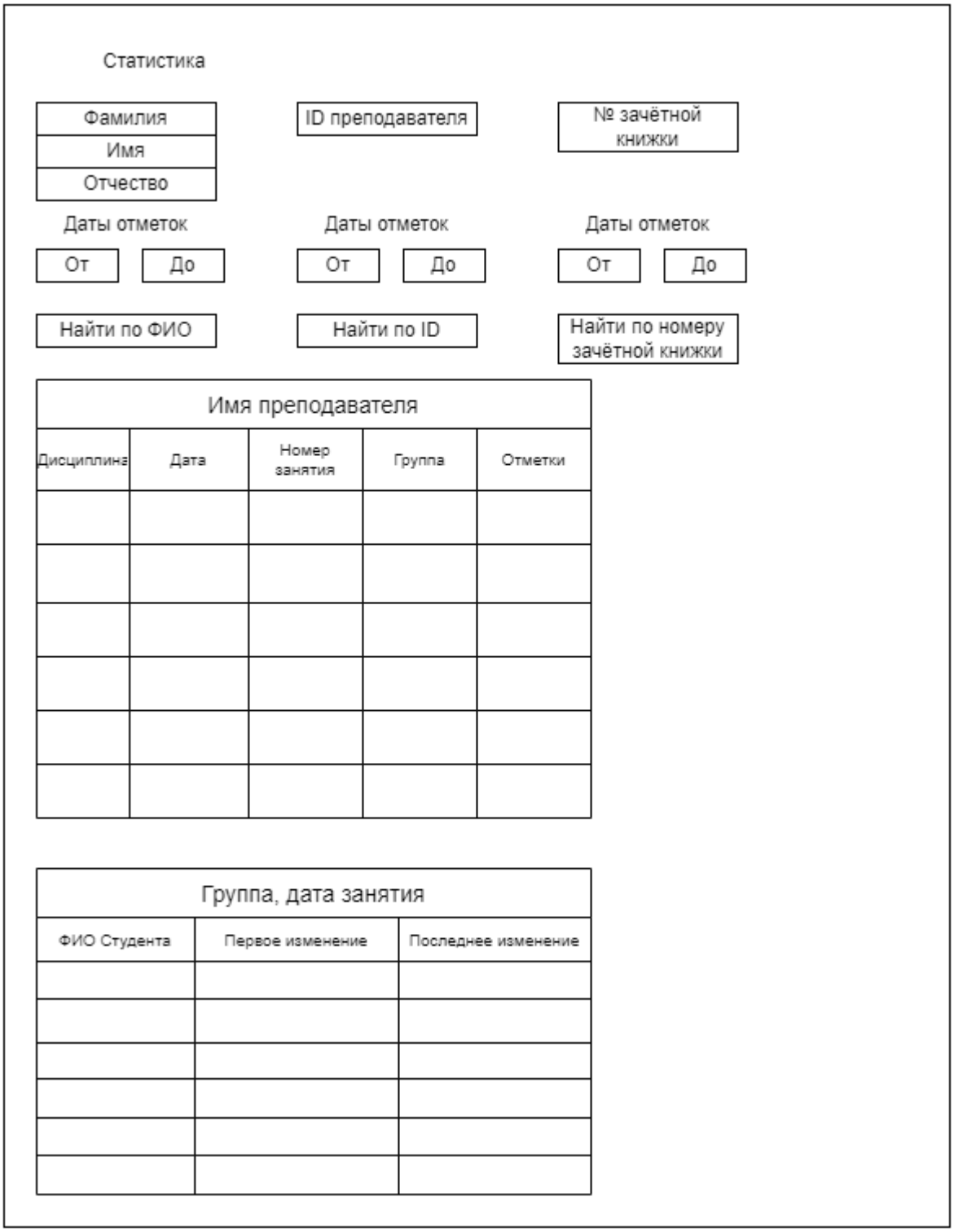

Рисунок 3.5.4 – Макет страницы просмотра статистики

После выбора периода времени, за который собирается информация, формируется первая таблица. Она содержит обобщенную статистику по отметкам преподавателя. В ней сгруппированы занятия по дисциплинам и отсортированы внутри групп по датам. Колонки таблицы содержат дисциплину, дату занятия, номер занятия, группу и количество отметок на этом занятии. В каждой ячейке колонки «Отметки» находится кнопка, нажатие на которую формирует еще одну таблицу.

Последняя таблица содержит информацию по конкретному занятию. На ней отображено каких студентов преподаватель отметил, в какое время появилась отметка и в какое время была в последний раз изменена. В случае поиска информации по студенту формируется только последняя таблица, в которой содержится только информация по искомому студенту.

4 РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПОДСИСТЕМЫ «КОНТРОЛЬ И УЧЕТ ПОСЕЩАЕМОСТИ И РАБОТЫ СТУДЕНТОВ НА ЗАНЯТИЯХ» ДЛЯ САЙТА ОГУ ИМ. И.С. ТУРГЕНЕВА

4.1 Реализация компонентов

Реализация контроллера в используемом фреймворке представляет из себя создание класса, наследующего функционал существующих в фреймворке класса контроллера. Это позволяет скрывать от разработчика детали реализации, позволяя сосредоточиться исключительно на функционале разрабатываемого модуля.

Реализация обработки определенного URL-адреса на сайте в контроллере заключается в создании метода обработчика с соответствующим названием. Для представления «index.phtml» создается метод «indexAction()» и т.д.

Создание класса контроллера представлен на рисунке 4.1.1. Класс контроллера наследуется от «Zend_Controller_Action». Приватным полем класса является информация об идентификации пользователя, полученного методом «getIdentity» из библиотеки сайта.

```
class User_AttendanceController extends Zend_Controller_Action
{
    private $identity;
    public function init()
    {
        $auth = Site_Auth::getInstance();
        $this->identity = $auth->getIdentity();
    }
}
```

Рисунок 4.1.1 – Создание класса контроллера

При реализации обработчика (рисунок 4.1.2) представления какой-либо страницы контроллеру необходимо получить параметры запроса и определить права доступа.

```
public function indexAction()
    {
        $params = $this->_getAllParams();
        $employee =
User_Model_Employee::getEmployee($params['iduser']);
        $idEmployee = $employee->id;
        $isIdSame = $idEmployee == $this->identity->id;
        $isUserHasAccess = $this->identity->usertype ==
Site_Auth::ADMIN || $this->identity->usertype ==
Site_Auth::EMPLOYEE;
        $this->view->usertype = $this->identity->usertype;
        if(false == $isUserHasAccess){
            $this->_helper->redirector->goToUrl('/');
        }
    }
```

Рисунок 4.1.2 – Создание метода обработчика представления главной страницы

В приведенном выше рисунке пользователи, не имеющие доступа, будут перенаправлены на главную страницу.

Контроллер должен обращаться за данными к модели. Для демонстрации обращения к модели можно рассмотреть алгоритм поиска нужного занятия на текущий момент времени, либо по указанному номеру (рисунок 4.1.3).

В реализации этого алгоритма контроллер обращается к модели через метод «getLessonsList» для формирования набора занятий на определенную дату для преподавателя. Затем в представление модели передаются полученные данные, в поле «todayLessonsDates» объекта представления «view». Далее реализуется сам алгоритм поиска нужного занятия. Параметр «numberlesson» указывает на то, что в запросе к контроллеру был указан конкретный номер занятия, значит нужно лишь отыскать его в массиве полученных занятий. Иначе, необходимо найти последнее начавшееся занятия, для чего массив занятий перебирается с конца. Этот алгоритм был представлен в виде блок-схемы на рисунке 3.4.2.

```
$todayLessonsDates =
User_Model_LiteAttendance::getLessonsList($idEmployee, $date);
        $this->view->todayLessonsDates = $todayLessonsDates;
        $currentLesson = [];
        if(isset($params['numberlesson'])){
            foreach($todayLessonsDates as $lesson){
                if ($lesson['NumberLesson'] ==
$params['numberlesson']){
                    $currentLesson = $lesson;
                    break;
                }
            }
        }
        elseif($date == Date('Y-m-d')){
            $current_time = date('H:i');
            for($i = count($todayLessonsDates)-1; $i>=0; $i--){
                if
(strtotime($todayLessonsDates[$i]['startTime']) <=
strtotime($current_time)) {
                    $currentLesson = $todayLessonsDates[$i];
                    break;
                }
            }
        }
```

Рисунок 4.1.3 – Обращение к модели и поиск занятия для отображения

Создание публичного API в контроллере так же происходит через создание методов обработчиков. Отличие лишь в том, что такие методы не подготавливают представления, а возвращают ответ в формате JSON. Пример такого обработчика можно увидеть на рисунок 4.1.4.

Из параметров запроса контроллер получает необходимые данные и отправляет в метод модели, ответ которой упаковывает в формат JSON и отправляет обратно клиенту.

```
    public function getlessontypeAction(){
        if(!$this->_request->isPost()){
            return;
        }
        $post = $this->_request->getPost();
        $isuup_subject_id = $post['isuup_subject_id'];
        $idEmployee = $post['idEmployee'];
        $idGroup = $post['idGroup'];
        $result =
User_Model_LiteAttendance::get_lesson_type_encoded($isuup_subjec
t_id, $idEmployee, $idGroup);
        $this->_helper->json($result);
        return;
    }
```

Рисунок 4.1.4 – Метод обработчик для запроса типов занятия

Модель реализуется отдельным классом с множеством статических методов. Пример реализации модели представлен на рисунке 4.1.5.

```
class User_Model_LiteAttendance
{
    public static function
get_lesson_type_encoded($isuup_subject_id, $idEmployee,
$idGroup)
    {
        $sql = "SELECT DISTINCT cos2.TypeLesson FROM
dbo.CellOfSchedule cos2
            JOIN gruop grp ON grp.idgruop = cos2.idGruop
            WHERE cos2.employee_id = :idEmployee
            AND cos2.isuup_subject_id = :isuup_subject_id
            AND cos2.Semester = :semester
            AND grp.idgruop = :idGroup";
        $semester =
User_Model_CurrentControl::getCurrentSemester();
        $queryParams = array(
            'idEmployee' => $idEmployee,
            'isuup_subject_id' => $isuup_subject_id,
            'idGroup' => $idGroup,
            'semester' => $semester);
        $query_result = Application_Model_Db::queryPDO($sql,
$queryParams)->fetchAll(PDO::FETCH_ASSOC);
        $lesson_types = array();
        foreach ($query_result as $result) {
            $row = [iconv('WINDOWS-1251', 'UTF-8', 'TypeLesson')
=> iconv('WINDOWS-1251','UTF-8',$result['TypeLesson'])];
            array_push($lesson_types, $row);
        }
        return $lesson_types;
    }
}
```

Рисунок 4.1.5 – Пример модели с методом получения типов занятий

Внешний вид страниц модуля создается при помощи разметки на языке HTML с вставками PHP кода, в котором возможно использование данных, переданных от контроллера (рисунок 4.1.6).

```
    <h1>Журнал посещаемости</h1>
    <div class="lists_of_groups">
        <div class="lesson_selector">
            <select class="lesson_selector_button"
id="choice_of_discipline">
                <option selected>Предмет</option>
                <?php if (!empty($this->subjects_list)):
                    foreach ($this->subjects_list as $subject):
?>
                    <option data-idEmployee="<?= $this-
>employee->id?>" data-id="<?= $subject['isuup_subject_id']; ?>"
value="<?= $subject['TitleSubject']; ?>"><?=
$subject['TitleSubject']; ?></option>
                <?php endforeach;
                endif; ?>
            </select>
```

Рисунок 4.1.6 – Фрагмент реализации разметки страницы журнала посещаемости

Действия пользователя обрабатываются кодом на языке JavaScript. События генерируются действиями пользователя и обрабатываются соответствующими функциями (рисунок 4.1.7).

```
document.getElementById('select_group').addEventListener('change
', () => {
        var selectedGroup = this.value;
        var choiceOfDiscipline =
document.getElementById('choice_of_discipline');
        var selectedOptionSubject =
choiceOfDiscipline.options[choiceOfDiscipline.selectedIndex];
        var isuup_subject_id =
selectedOptionSubject.getAttribute('data-id');

        clearSubgroupTables();
        getSubgroupsRequest(isuup_subject_id, selectedGroup);
    });
```

Рисунок 4.1.7 – Функция обработки выбора группы на странице разделения на подгруппы

В приведенном выше рисунке происходит обработка события выбора группы, а точнее события «change» на интерактивном элементе выбора в виде выпадающего списка групп. После подготовки данных вызывается функция «getSubgroupsRequest» (рисунок 4.1.8), инкапсулирующая асинхронный запрос на сервер.

```
    function getSubgroupsRequest(idSubject, idGroup) {
        $.ajax({
            url: '/attendance/groupdivision/<?= $this->employee-
>id ?>',
            type: 'POST',
            data: {
                action: 'getSubgroups',
                idEmployee: <?= $this->employee->id ?>,
                idSubject: idSubject,
                idGroup: idGroup
            },
            success: (responce) => {
                if (DEBUG) console.table(responce);
                var buttonContainer =
document.querySelector('.buttons-container');
                currentStudentsData = responce;
                createSubgroupTables();
            },
            error: function(xhr) {
                console.error('Error:', xhr.responseText);
            }
        });
    }
```

Рисунок 4.1.8 – Реализация асинхронного запроса на сервер в функции «getSubgroupsRequest»

4.2 Реализация интерфейсов

В реализации интерфейсов разрабатываемой системы важную роль играют удобство и восприятие информации пользователем. Это неизмеримые в количественном отношении характеристики, которые, тем не менее, являются важной частью пользовательского опыта в использовании любого web-сайта.

При реализации интерфейсов главной страницы и страницы журнала посещаемости были приняты меры, способствующие улучшению

пользовательского опыта. Например, интерактивные элементы выбора (чекбоксы) для отметки присутствия студентов были намеренно увеличены относительно размера шрифта.

Заголовки таблиц окрашены в темно синий цвет, что не только способствует более быстрому ориентированию на странице пользователем, но и вписывается в стилистику всего остального сайта.

Текущее занятие в таблице со списком всех занятий на сегодня подсвечено зеленым цветом, чтобы помочь пользователю сориентироваться в расписании (рисунок 4.2.1).

Рисунок 4.2.1 – Интерфейс главной страницы

Лабораторные и практические занятия зачастую идут парами. Это означает что два занятия с одной и той же группой у одного преподавателя будут идти подряд, соответственно состав группы, посетившей первое из занятий, редко будет отличаться от тех, кто посетил и второе. Для удобства в

этих случаях для главной страницы введена вспомогательная подсветка (рисунок 4.2.2) для студентов с прошлого занятия. Студенты, которых не было на прошлом занятии, отмечены красным, иначе зеленым. Такая подсветка становится видна только на втором занятии, в случае если два практических либо лабораторных занятия идут подряд.

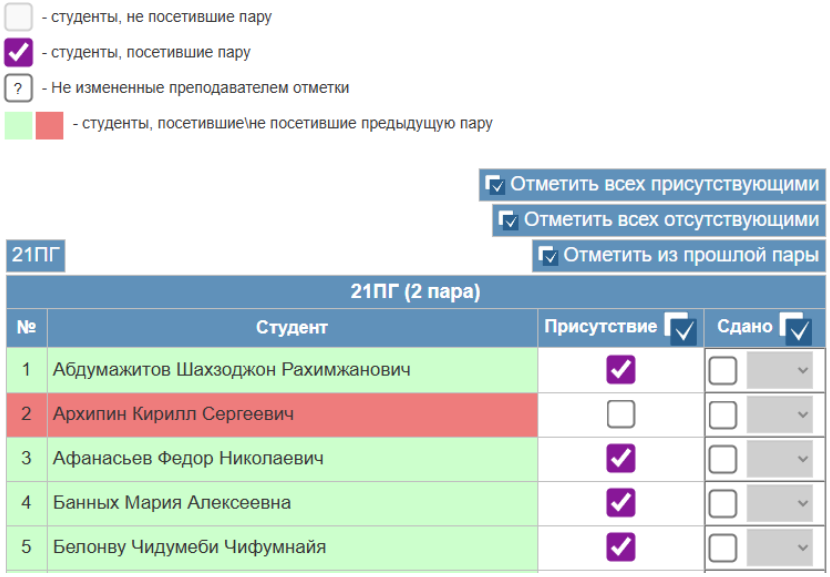

Рисунок 4.2.2 – Вспомогательная подсветка студентов, посетивших предыдущее занятие на главной странице

На странице журнала посещаемости необходимо ориентироваться в объемной таблице с большим количеством элементов. Для удобства пользователей при наведении мыши на кнопки для отметки всех студентов подсвечивается весь столбец отметок, которые будут затронуты нажатием кнопки. Это делает действия пользователя более предсказуемыми, а как следствие и осознанными (рисунок 4.2.3).

Отметить посещаемость на сегодня

Журнал посещаемости

Проектная деятельность | 21ПГ | лаб | Показывать отметки за прошлый семестр

№	Студент	10.02.2026 1 пара	10.02.2026 2 пара	17.02.2026 1 пара	17.02.2026 2 пара	24.02.2026 1 пара	24.02.2026 2 пара	03.03.2026 1 пара	03.03.2026 2 пара	06.03.2026 1 пара	06.03.2026 2 пара	10.03. 1 па
		21ПГ										
		Тема заня	Тема заня	Тема заня	Тема заня	Тема заня	Тема заня	Тема заня	Тема заня	Тема заня	Тема заня	Тема заня
1	Абдумажитов Шахзоджон Рахимжанович	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑
2	Архипин Кирилл Сергеевич	☐	☐	☐	☐	☑	☑	☐	☐	☐	☐	☑
3	Афанасьев Федор Николаевич	☐	☐	☐	☐	☑	☑	☐	☐	☐	☐	☑
4	Банных Мария Алексеевна	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐	☑
5	Белонву Чидумеби Чифумнайя	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☑
6	Василения Иван Валерьевич	☑	☑	☐	☐	☐	☐	☐	☐	☐	☐	☑

Рисунок 4.2.3 – Интерфейс страницы журнала посещаемости

На странице просмотра статистики (рисунок 4.2.4) представлены элементы для задания рамок статистики и просмотра занятий, на которых преподаватель отмечал студентов.

Назад в отметки о посещаемости

ФИО преподавателя: Конюхова Оксана Владимировна

Даты отметок: 22.03.2026 - 22.03.2026

Найти по ФИО

ID преподавателя: 121

Даты отметок: 22.03.2026 - 22.03.2026

Найти по ID

№ зачётки: № зачётки

Даты отметок: 22.03.2026 - 22.03.2026

Найти

Преподаватель: Конюхова Оксана Владимировна

Предмет	Дата	Номер пары	Группа	Отметки	
Проектирование вычислительных систем и сетей	04.03.2026	3, лек	31ИВТ	10	Посмотреть
	06.03.2026	3, лаб	31ИВТ	10	Посмотреть
	06.03.2026	4, лаб	31ИВТ	10	Посмотреть
	11.03.2026	3, лек	31ИВТ	10	Посмотреть
	16.03.2026	5, лек	31ИВТ	10	Посмотреть
	20.03.2026	3, лаб	31ИВТ	10	Посмотреть
	20.03.2026	4, лаб	31ИВТ	10	Посмотреть
Проектирование операционных систем	27.02.2026	5, лек	51ПГ-м	10	Посмотреть
	05.03.2026	5, лаб	51ПГ-м	10	Посмотреть
	05.03.2026	6, лаб	51ПГ-м	10	Посмотреть
	06.03.2026	5, лек	51ПГ-м	10	Посмотреть
	17.03.2026	6, лек	51ПГ-м	10	Посмотреть
	17.03.2026	7, лек	51ПГ-м	10	Посмотреть
	19.03.2026	5, лаб	51ПГ-м	10	Посмотреть
	19.03.2026	6, лаб	51ПГ-м	10	Посмотреть

Рисунок 4.2.4 – Интерфейс страницы просмотра статистики

При нажатии кнопки «Посмотреть» формируется таблица для просмотра отметок на конкретное занятие (рисунок 4.2.5).

Группа: 51ПГ-м
Дата занятия: 2026-03-05

Студент	Первое изменение	Последнее изменение
Гордеев Денис Алексеевич	Мар 6 2026 05:10:32:530AM	
Горлов Дмитрий Александрович	Мар 6 2026 05:10:32:530AM	
Дрожжаков Иван Игоревич	Мар 6 2026 05:10:32:530AM	Мар 20 2026 07:18:54:673AM
Ефанов Никита Павлович	Мар 6 2026 05:10:32:530AM	
Ильичев Артём Александрович	Мар 6 2026 05:10:32:530AM	
Новикова Ангелина Александровна	Мар 6 2026 05:10:26:667AM	Мар 20 2026 07:19:00:713AM
Пителин Дмитрий Алексеевич	Мар 6 2026 05:10:32:530AM	
Руднев Олег Владимирович	Мар 6 2026 05:10:32:530AM	
Шелыганов Дмитрий Сергеевич	Мар 6 2026 05:10:32:530AM	
Карасев Денис Игоревич	Мар 6 2026 05:10:32:530AM	

Рисунок 4.2.5 – Вывод статистики за конкретное занятие

Реализованный интерфейс для страницы разделения на подгруппы представлен на рисунке 4.2.5.

Деление на подгруппы

Преддипломная практика | 41ПГ-м

Выберите предмет, группу и тип пары, чтобы просмотреть и отредактировать разделение на подгруппы

Подгруппа 1

№	Студент	
245110	Волкова Ирина Сергеевна	☐
245525	Дударев Илья Владиславович	☐
235119	Колесников Артем Георгиевич	☐
245106	Коновалов Вячеслав Александрович	☐
245109	Лопатин Артем Денисович	☐
245526	Мальцев Евгений Максимович	☐
235537	Полянин Никита Денисович	☐
245111	Сосунов Лев Владимирович	☐
235538	Сурова Евгения Павловна	☐
245112	Сырцев Вадим Игоревич	☐

>>

<<

Сохранить

Подгруппа 2

№	Студент	
245101	Абрашин Сергей Александрович	☐
245102	Алёшин Сергей Юрьевич	☐
245103	Амелин Павел Павлович	☐
245567	Архебамен Шалом Мофогофолува	☐
245156	Кутаков Михаил Андреевич	☐

Рисунок 4.2.5 – Интерфейс страницы деления на подгруппы

Чаще всего современные пользователи используют телефоны для доступа к web-сайтам. Соответственно, необходимо уделять должное внимание адаптации интерфейсов под мобильные устройства. Для этого существуют стандартные инструменты языка формального описания внешнего вида страницы CSS. Адаптация под мобильные устройства представляет из себя изменение разметки и расположения некоторых элементов для удобства доступа с экрана телефона. Примеры адаптации под мобильные устройства на рисунках 4.2.6 и 4.2.7.

Рисунок 4.2.6 – Адаптированный под мобильные устройства интерфейс страницы журнала посещаемости

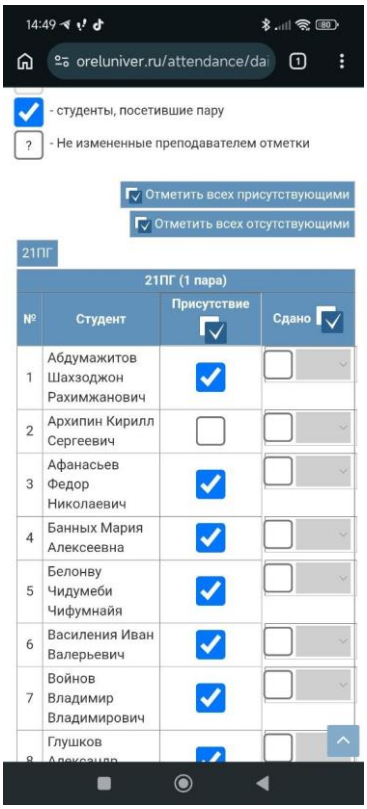

Рисунок 4.2.7 – Адаптированный под мобильные устройства интерфейс главной страницы

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы была достигнута поставленная цель. Произведен реинжениринг процесса и расширение функционала учёта посещаемости и работы студентов на занятиях для упрощения контроля за заполнением и повышения надёжности работы сервиса.

В рамках работы выполнены следующие задачи:

а) Произведен анализ предметной области, выявлены недостатки «бумажного» журнала по сравнению с разрабатываемой системой. Сформулированы задачи исследования и требования к разработке.

б) проведены теоретические исследования и определить спецификаций программного обеспечения;

в) спроектировано программное обеспечение;

г) разработан макет программных компонентов и пользовательских интерфейсов.

Внедрение разработанной подсистемы позволило получить следующие улучшения по сравнению с существующей системой учета посещаемости:

а) Благодаря реализации автоматического выбора занятия по текущему времени преподаватель избавляется от необходимости делать три последовательных запроса в большинстве случаев, что позволяет сократить время получения актуального списка группы с 8-12 секунд до 1-2 секунд, протестировано с задержкой 200 мс и скоростью 1 Мбит/с;

б) Введение троичной логики и логирования событий позволяет вести наблюдение за деятельностью преподавателей и формировать соответствующие отчеты. Время формирования деятельности преподавателя по всем дисциплинам за семестр займет не более 5 секунд, вместо 15-30 минут ручного анализа бумажных носителей;

в) Разделение на подгруппы позволяет преподавателям сохранять собственные разбиения на подгруппы и сократить время отметки студентов на занятии для одной из подгрупп. В разработанной системе это займет не более

минуты, как и при любом другом занятии, вместо 5-7 минут поиска необходимого списка и повторного уточнения состава подгрупп;

г) Изменения и создание отметок сохраняется в базе данных, что повышает надежность системы, предоставляя доступ к информации об отметках без риска утери и порчи бумажных носителей, а также без риска утери данных о занятиях из расписания.

Таким образом, разработанная система соответствует выдвигаемым функциональным и нефункциональным требованиям и позволяет существенно упростить процесс учета посещаемости и сдачи работ студентами.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Отраслевые и специализированные решения 1С:Предприятие : [сайт]. – Москва, 2025. – URL: https://solutions.1c.ru/ (дата обращения: 21.10.2025);
2. Электронный деканат (Free Dean's Office) : [сайт]. – URL: https://www.deansoffice.ru/ (дата обращения: 21.10.2025);
3. Zend Framework : материалы Википедии – свободной энциклопедии / Википедия. – Обновлена 21.10.2025. – URL: https://ru.wikipedia.org/wiki/Zend_Framework (дата обращения: 21.10.2025);
4. Электронные системы учета успеваемости в вузе: все, что нужно знать : [сайт]. – Обновлено 15.09.2025. – URL: https://wowprofi.ru/blog/elektronnye-sistemy-ucheta-uspevaemosti-v-vuze-vse-chto-nuzhno-znat (дата обращения: 21.10.2025);
5. Model-View-Controller : материалы Википедии – свободной энциклопедии / Википедия. – Обновлена 23.10.2025. – URL: https://ru.wikipedia.org/wiki/Model-View-Controller (дата обращения: 22.10.2025);
6. Allen, S. Zend Framework in Action / S. Allen, F. Wierzbicki, J. MacLean. – Shelter Island : Manning Publications, 2009. – 425 p. – ISBN 978-1-933988-32-0;
7. Приказ Министерства образования и науки РФ от 06.04.2021 № 245 «Об утверждении Порядка организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры» (с изм. от 27.02.2023). – URL: https://www.consultant.ru/document/cons_doc_LAW_384366/ (дата обращения: 15.03.2026);

8. Кузнецов, С. Д. Основы баз данных : учебное пособие / С. Д. Кузнецов. – 2-е изд., перераб. и доп. – Москва : Интернет-Университет Информационных Технологий, 2021. – 488 с. – ISBN 978-5-4497-0116-1;
9. Федоров, А. В. Разработка веб-приложений с использованием архитектурного шаблона MVC / А. В. Федоров, Д. А. Петров // Программные продукты и системы. – 2024. – № 2. – С. 214–221. – ISSN 0236-235X. – DOI: 10.15827/0236-235X.138.214-221;
10. Сайт ФГБОУ ВО «Орловский государственный университет имени И.С. Тургенева» : [сайт]. – Орел, 2025. – URL: https://oreluniver.ru/ (дата обращения: 20.03.25).

Приложение

Фрагменты исходного кода

Фрагмент файла AttendanceController.php

```
<?php

class User_AttendanceController extends Zend_Controller_Action
{
    private $identity;

    public function init()
    {
        $auth = Site_Auth::getInstance();
        $this->identity = $auth->getIdentity();
    }

    public function getmultigroupamountAction(){
        if(!$this->_request->isPost()){
            return;
        }
        if($this->identity->usertype != Site_Auth::ADMIN &&
$this->identity->usertype != Site_Auth::EMPLOYEE){
        }
        elseif($this->identity->usertype == Site_Auth::EMPLOYEE
&& $post['idEmployee'] != $this->identity->id){
            return;
        }

        $post = $this->_request->getPost();
        $res =
User_Model_LiteAttendance::getMultigroupAmount($post);
        $this->_helper->json($res);
    }
```

```
    public function indexAction()
    {
        $params = $this->_getAllParams();

        $employee =
User_Model_Employee::getEmployee($params['iduser']);
        $idEmployee = $employee->id;

        $isIdSame = $idEmployee == $this->identity->id;
        $isUserHasAccess = $this->identity->usertype ==
Site_Auth::ADMIN ||
                           $this->identity->usertype ==
Site_Auth::EMPLOYEE;

        if($this->_request->isPost()){
            $post = $this->_request->getPost();
            if (false == $isUserHasAccess){
                $this->getResponse()->setHttpResponseCode(402);
                return;
            }
            if ($this->identity->usertype == Site_Auth::EMPLOYEE
&& false == $isIdSame){
                $this->getResponse()->setHttpResponseCode(500);
                return;
            }

            $today = date('Y-m-d');


            $beginTimes = [
                1 => '8:30',
                2 => '10:10',
                3 => '12:00',
                4 => '13:40',
                5 => '15:20',
```

```
                6 => '17:00',
                7 => '18:40',
                8 => '20:15'
            ];
            $begintime = $beginTimes[$post['numberLesson']];


            if($post['numberLesson'] > 1) {
                if(isset($params['date']))
                    $date = $params['date'];
                else{
                    $date = $today;
                }

                $todayLessonsDates =
User_Model_LiteAttendance::getLessonsList($idEmployee, $date);

                foreach($todayLessonsDates as $i => $lesson){
                    if($lesson['NumberLesson'] ==
$post['numberLesson']){
                        break;
                    }
                }
                if(
                    $i != 0 &&
                    $todayLessonsDates[$i-1]['TypeLesson'] ==
iconv('UTF-8', 'WINDOWS-1251', $post['typeLesson']) &&
                    $todayLessonsDates[$i-1]['idSubject'] ==
$post['idSubject'] &&
                    $todayLessonsDates[$i]['GroupsID'] ==
$todayLessonsDates[$i-1]['GroupsID']
                )
                {
                    if($todayLessonsDates[$i-1] !== null){
```

```
                        $begintime =
$beginTimes[$post['numberLesson']-1];
                    }
                }
            }

            $current_HM_time = date('H:i');


            if(isset($post['markDone'])) {
                $params = [
                    'idEmployee' => $idEmployee,
                    'idSubject' => $post['idSubject'],
                    'idGroup' => $post['idGroup'],
                    'typeLesson' => iconv('UTF-8', 'WINDOWS-
1251', $post['typeLesson']),
                    'dateLesson' => $post['dateLesson'],
                    'numberLesson' => $post['numberLesson'],
                    'markDone' => $post['markDone'],
                ];
                if(isset($post['studBook'])) {
                    $params['NumberStudBook'] =
$post['studBook'];

User_Model_LiteAttendance::addDoneMark($params);
                }
                else {
                    $params['doneStudArray'] =
$post['doneStudArray'];

User_Model_LiteAttendance::addDoneMark($params);
                }
                $params['typeLesson'] = $post['typeLesson'];
                $this->_helper->json($post);
            }
```

```
            elseif(isset($post['mark'])){
                $params = [
                    'idEmployee' => $idEmployee,
                    'idSubject' => $post['idSubject'],
                    'idGroup' => $post['idGroup'],
                    'typeLesson' => iconv('UTF-8', 'WINDOWS-
1251', $post['typeLesson']),
                    'dateLesson' => $post['dateLesson'],
                    'numberLesson' => $post['numberLesson'],
                    'value' => $post['mark'] == 'true',
                ];


                if(isset($post['studBook'])){
                    $params['NumberStudBook'] =
Site_Ajax::dataUtf8ToWin1251($post['studBook']);

User_Model_LiteAttendance::addAttendanceMark($params);
                }
                elseif (isset($post['markArray'])) {
                    $params['markArray'] =
Site_Ajax::dataUtf8ToWin1251($post['markArray']);;

User_Model_LiteAttendance::addAttendanceMarkAll($params);
                }
                else{

User_Model_LiteAttendance::addAttendanceMarkAll($params);
                }

                $params['typeLesson'] = $post['typeLesson'];
                $this->_helper->json($post);

            }
            throw new Zend_Exception('Corrupted post-request');
```

```
        }

        $this->view->usertype = $this->identity->usertype;

        if(false == $isUserHasAccess){
            $this->_helper->redirector->goToUrl('/');
        }
        elseif($this->identity->usertype == Site_Auth::EMPLOYEE
&& $idEmployee != $this->identity->id){
            $this->_helper->redirector-
>goToUrl('/employee/'.$this->identity->id);
        }
        $this->view->employee = $employee;

        if(isset($params['date']))
            $date = $params['date'];
        else{
            $date = Date('Y-m-d');
        }

        $todayLessonsDates =
User_Model_LiteAttendance::getLessonsList($idEmployee, $date);
        $this->view->todayLessonsDates = $todayLessonsDates;

        $currentLesson = [];
        if(isset($params['numberlesson'])){
            foreach($todayLessonsDates as $lesson){
                if ($lesson['NumberLesson'] ==
$params['numberlesson']){
                    $currentLesson = $lesson;
                    break;
                }
            }
        }
        elseif($date == Date('Y-m-d')){
```

```
            $current_time = date('H:i');
            for($i = count($todayLessonsDates) - 1; $i >= 0; $i-
-){
                if
(strtotime($todayLessonsDates[$i]['startTime']) <=
strtotime($current_time)) {
                    $currentLesson = $todayLessonsDates[$i];
                    break;
                }
            }
        }
        elseif(count($todayLessonsDates) > 0){
            $currentLesson = $todayLessonsDates[0];
        }

        if($currentLesson['DateLesson'] == [])
$currentLesson['DateLesson'] = $date;
        $currentLesson['ConnectionDateTime'] = date('Y-m-d
H:i:s');
        $this->view->currentLesson = $currentLesson;
        $ind = null;
        foreach($todayLessonsDates as $i => $lesson){
            if($lesson['NumberLesson'] ==
$currentLesson['NumberLesson']){
                $ind = $i;
                break;
            }
        }
        if(
            $i != 0 &&
            $todayLessonsDates[$i-1]['TypeLesson'] ==
$currentLesson['TypeLesson'] &&
            $todayLessonsDates[$i-1]['idSubject'] ==
$currentLesson['idSubject'] &&
```

```
            $todayLessonsDates[$i-1]['GroupsID'] ==
$currentLesson['GroupsID']
        ){
            $this->view->prevLesson = $todayLessonsDates[$i-1];
        }
        if($currentLesson['TypeLesson'] == 'лек' &&
count($currentLesson['GroupsID']) > 1){
            $this->view->isLectureMultigroup = true;
        }
    }
}
```

Фрагмент файла LiteAttendance.php

```
class User_Model_LiteAttendance
{
    public static function get_attendances_encoded($idEmployee,
$idSubject, $idGroup, $dateLesson, $NumberLesson){
        $sql = "SELECT DISTINCT
            CONCAT(ms.Surname, ' ', ms.Name, ' ', ms.Patronymic)
AS fio,
            ms.numberStudBook,
            cca.Mark,
            cca.markDone,
            cca.idEmployee,
            cca.idA,
            sg.numberSubGroup
        FROM
            (select * from attest.dbo.ManStudent AS mms where
mms.idGruop = :idGroup) ms
        LEFT JOIN
            dbo.CurrentControlAttendance AS cca
        ON cca.NumberStudBook = ms.NumberStudBook
            AND cca.idSubject = :idSubject
            AND cca.NumberLesson = :NumberLesson
            AND cca.dateLesson = :dateLesson
```

```
            AND cca.idGroup = ms.idGruop
        LEFT JOIN dbo.CurrentControlSubGroup as sg
            ON sg.studBook = ms.NumberStudBook
            AND sg.idSubject = :idSubject
            AND sg.idEmployee = :idEmployee
            ORDER BY fio, cca.idA
        ";

        $queryParams = array(
            'idSubject' => $idSubject,
            'idGroup' => $idGroup,
            'idEmployee' => $idEmployee,
            'dateLesson' => $dateLesson,
            'NumberLesson' => $NumberLesson
        );
        $query_result = Application_Model_Db::queryPDO($sql,
$queryParams)->fetchAll(PDO::FETCH_ASSOC);

        $attendances = array();
        $prevRow = [];
        foreach($query_result as $result){
            if($result['numberStudBook'] ==
$prevRow['numberStudBook']){
                $right = $prevRow;
                if(self::is_mark_default($prevRow)){
                    if(self::is_mark_default($result)){
                        $right['idEmployee'] = $idEmployee;
                    }
                    else{

                        $right = $result;
                    }
                }
                $attendances[count($attendances) - 1]['mark'] =
$right['Mark'];
```

```
                $attendances[count($attendances) -
1]['markDone'] = $right['markDone'];
                $attendances[count($attendances) -
1]['rowFromEmployee'] = $right['idEmployee'];
                continue;
            }

            $row = [
                iconv('WINDOWS-1251', 'UTF-8', 'numberStudBook')
=> iconv('WINDOWS-1251', 'UTF-8', $result['numberStudBook']),
                iconv('WINDOWS-1251', 'UTF-8', 'fio') =>
iconv('WINDOWS-1251', 'UTF-8', $result['fio']),
                iconv('WINDOWS-1251', 'UTF-8', 'mark') =>
iconv('WINDOWS-1251', 'UTF-8', $result['Mark']),
                iconv('WINDOWS-1251', 'UTF-8', 'markDone') =>
iconv('WINDOWS-1251', 'UTF-8', $result['markDone']),
                iconv('WINDOWS-1251', 'UTF-8', 'numberSubGroup')
=> iconv('WINDOWS-1251', 'UTF-8', $result['numberSubGroup']),
                'idA' => $result['idA']
            ];
            if($result['idA'] != null && $result['idEmployee']
!= $idEmployee){


                if(!self::is_mark_default($result)){
                    $row['rowFromEmployee'] =
$result['idEmployee'];
                }
            }
            array_push($attendances, $row);
            $prevRow = $result;
        }
        return $attendances;
    }
```

Фрагмент файла index.phtml

```
     function createTable(responce) {
          if (DEBUG) {
               console.log('Успешный ajax (данные к созданию
таблицы студентов)');
               console.table(responce);
          }
          const currentLesson =
document.getElementById('currentLesson');

          const table = document.getElementById('student_table');
          table.innerHTML = '';
          const tbody = document.createElement('tbody');
          const thead = table.createTHead();

          let headerRow = thead.insertRow();
          const groupTagHeader = document.createElement('th');
          groupTagHeader.style.color = 'white';
          groupTagHeader.colSpan = 3;
          groupTagHeader.id = 'table-group-header';
          groupTagHeader.innerHTML = groupButton.innerHTML + " ("
+ document.getElementById('currentLesson').dataset.numberlesson
+ " пара)";
          headerRow.appendChild(groupTagHeader);

          let headerCell = document.createElement('th');

          headerRow = thead.insertRow();

          headerCell = document.createElement('th');
          headerCell.textContent = "№";
          headerCell.style.color = 'white';
          headerRow.appendChild(headerCell);

          headerCell = document.createElement('th');
          headerCell.textContent = "Студент";
```

```
        headerCell.style.color = 'white';
        headerRow.appendChild(headerCell);

        headerCell = document.createElement('th');
        headerCell.textContent = 'Присутствие';
        headerCell.style.color = "white";
        headerCell.classList.add("attendance_header_cell");
        const attendanceButton =
document.createElement('button');
        attendanceButton.className = 'image-button';
        attendanceButton.title = 'отметить/снять всех';
        attendanceButton.addEventListener('click',
toggleMarkAll);
        const attendanceImage = document.createElement('img');
        attendanceImage.src = 'http:
        attendanceButton.appendChild(attendanceImage);
        headerCell.appendChild(attendanceButton);
        headerRow.appendChild(headerCell);

        attendanceButton.addEventListener('mouseover',
function() {

            var marks = document.querySelectorAll(".mark");
            for(let i = 0; i < marks.length; i++){
                marks[i].parentElement.style.backgroundColor
= "#F4FFA5";
            }
        });

        attendanceButton.addEventListener('mouseout',
function() {

            var marks = document.querySelectorAll(".mark");
            for(let i = 0; i < marks.length; i++){
```

```
                if (isMarkDisabled()) {

marks[i].parentElement.style.backgroundColor = '#D6D6D6';
                } else {

marks[i].parentElement.style.backgroundColor = "";
                }
            }
        });

        var disable = isMarkDisabled();
        if (isMarkDisabled()) {
            let disablemark =
currentLesson.dataset.disablemark;
            let message = disablemark == 'too_late' ?
'<b>Редактирование пар более чем 3 дня спустя недоступно.</b>'
                                                                :
'<b>Пожалуйста, дождитесь начала пары, чтобы приступить к
редактированию.</b>';
            console.log('message: ' + message)

            document.getElementById('not-started-
warning').innerHTML = message;
        }

        let counter = 1;
        let attendanceCounter = 0;

        if(DEBUG && responce.some(stud =>
stud['numberSubGroup'] == null)){


            console.log('кто то из группы не распределен в
подгруппы');
            isSubGroupDivision = false;
```

```
        }
        else if (DEBUG &&
document.getElementById('isSubGroupDivisionCheckbox').checked){
            isSubGroupDivision = true;
            addSubgroupRow(tbody, 'Подгруппа 1');
            responce.sort(function(a, b) {
                return Number(a['numberSubGroup']) -
Number(b['numberSubGroup']);
            });
        }
        else{
            isSubGroupDivision = false;
        }

        var lastSubGroup = 1;

        for (let rowData of responce) {

            if(isSubGroupDivision && rowData['numberSubGroup']
!= lastSubGroup){
                lastSubGroup = rowData['numberSubGroup'];
                addSubgroupRow(tbody, 'Подгруппа ' +
rowData['numberSubGroup']);
            }

            const row = tbody.insertRow();

            let cell = row.insertCell();
            cell.width = '5%';
            cell.align = 'center';
            cell.textContent = counter++;
            cell.classList.add('tdata');

            cell = row.insertCell();
            cell.align = 'center';
```

```
                cell.textContent = rowData['fio'];
                cell.classList.add('tdata');

                cell = row.insertCell();
                cell.align = 'center';

                let checkboxContainer =
document.createElement('span');
                let checkbox = document.createElement('input');
                checkboxContainer.appendChild(checkbox);

                checkbox.setAttribute('data-studBook',
rowData['numberStudBook']);

                if(rowData['rowFromEmployee'] != undefined &&
rowData['rowFromEmployee'] != currentLesson.dataset.idemployee){

                     cell.style.backgroundColor = '#D6D6D6';
                     checkbox.disabled = true;

                }
                else{
                     checkbox.addEventListener('click', () => {
changeAttendanceMark(checkbox) });

                     if(rowData['idA'] === null){

     checkboxContainer.classList.add('centered-question-mark');
                          checkbox.addEventListener('change',
(event) => {

     checkboxContainer.classList.remove('centered-question-
mark');
                          });
                     }
```

```
                }
                if(isMarkDisabled()) {

     checkbox.classList.add('timer_disabled_mark');
                }

                checkbox.classList.add('mark');
            checkbox.type = 'checkbox';
                checkbox.checked = rowData['mark'] == '1' ? true :
false;
                cell.appendChild(checkboxContainer);
                cell.classList.add('tdata');
                checkbox.addEventListener('mouseover', () => {
highlight_student_row(row, true) });
                checkbox.addEventListener('mouseout', () => {
highlight_student_row(row, false) });
                attendanceCounter += rowData['mark'] == 1 ? 1 : 0;
            }
            let row = tbody.insertRow();
            let cell = row.insertCell();
            cell.colSpan = 2;
            cell.innerHTML = "Всего";
            cell = row.insertCell();
            cell.id = 'totalVisited';
            cell.innerHTML = attendanceCounter;
            cell.style.textAlign = "center";

            table.appendChild(tbody);
            updateCount('mark', 'totalVisited');
     if(document.getElementById('currentLesson').dataset.type !=
'лек'){
                addMarkDone(responce);
            }
     if(document.getElementById('currentLesson').dataset.isprevle
sson == 'true'){
```

```
                addPrevLesson();
            }
            if (isMarkDisabled()) {
                const attendanceCells =
tbody.querySelectorAll('td:nth-child(3)');
                attendanceCells.forEach(cell => {
                  if (cell.parentElement !== tbody.lastElementChild)
{
                        cell.style.backgroundColor = '#D6D6D6';
                    }
                });
            }

            let disablemark =
document.getElementById('currentLesson').dataset.disablemark;
            if (disablemark != '0' && disablemark != '1') {
                let currenttime =
document.getElementById('currentLesson').dataset.connectiontime;

                const milliseconds = (h, m, s) =>
((h*60*60+m*60+s)*1000);
                const timeParts = disablemark.split(":");
                const currenttimeParts = currenttime.split(":");
                var time = milliseconds(timeParts[0]-
currenttimeParts[0], timeParts[1]-currenttimeParts[1], -
currenttimeParts[2]);

                if(time < 0) time = 0;
                setTimeout(() => {
                    let checkboxes =
document.querySelectorAll('.mark');
                    checkboxes.forEach(checkbox => {

checkbox.classList.remove('timer_disabled_mark');
                    });
```

```
                    checkboxes =
document.querySelectorAll('.mark_done');
                    checkboxes.forEach(checkbox => {

checkbox.classList.remove('timer_disabled_mark');
                    });
               }, time);
          }
     }
```

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»

УДОСТОВЕРЯЮЩИЙ ЛИСТ № 1

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

на демонстрационный материал, представленный в электронном виде

Студента Василения Ивана Валерьевича шифр 220887
Институт приборостроения, автоматизации и информационных технологий
Кафедра информационных систем и цифровых технологий
Направление 09.03.04 Программная инженерия
Направленность (профиль) Индустриальное производство программного обеспечения

Наименование документа: Демонстрационные плакаты к выпускной квалификационной работе

Документ разработал:

Студент Василения И.В. _____

Документ согласован:

Руководитель Ужаринский А.Ю. _____

Нормоконтроль Захарова О.В. _____

Документ утвержден:

И.о. зав. кафедрой Рыженков Д.В. _____

Орел 2026

ИНФОРМАЦИОННО-ПОИСКОВАЯ ХАРАКТЕРИСТИКА ДОКУМЕНТА НА ЭЛЕКТРОННОМ НОСИТЕЛЕ

Наименование		Характеристики документа на электронном носителе
группы атрибутов	атрибута	
1. Описание документа	Обозначение документа (идентификатор(ы) файла(ов))	Презентация.pptx
	Наименование документа	Демонстрационные плакаты к выпускной квалификационной работе
	Класс документа	ЕСКД
	Вид документа	Оригинал документа на электронном носителе
	Аннотация	Демонстрационный материал, отображающий основные этапы выполнения выпускной квалификационной работы
	Использование документа	Операционная система Windows 10, Microsoft PowerPoint 2013
2. Даты и время	Дата и время копирования документа	28.06.2026
	Дата создания документа	26.05.2026
	Дата утверждения документа	27.06.2026
3. Создатели	Автор	Василения И.В.
	Изготовитель	Василения И.В.
4. Внешние ссылки	Ссылки на другие документы	Удостоверяющий лист № 1
5. Защита	Санкционирование	ОГУ имени И.С. Тургенева
	Классификация защиты	По законодательству РФ
6. Характеристики содержания	Объем информации документа	1 584 919 Б

7. Структура документа(ов)	Наименование плаката (слайда) №1	Титульный лист
	Наименование плаката (слайда) №2	Цели и задачи работы
	Наименование плаката (слайда) №3	Анализ существующих методов решения поставленной задачи
	Наименование плаката (слайда) №4	Сравнение аналогичных решений
	Наименование плаката (слайда) №5	Функциональные требования
	Наименование плаката (слайда) №6	Функциональная модель подсистемы
	Наименование плаката (слайда) №7	Информационная спецификация подсистемы
	Наименование плаката (слайда) №8	Структура подсистемы
	Наименование плаката (слайда) №9	Поведенческая спецификация подсистемы
	Наименование плаката (слайда) №10	Схема алгоритма поиска нужного занятия на текущий момент времени, либо по указанному номеру
	Наименование плаката (слайда) №11	Интерфейсы разрабатываемой системы
	Наименование плаката (слайда) №12	Анализ результатов