

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

Профиль подготовки: «Программное обеспечение средств вычислительной техники и автоматизированных систем»

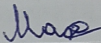
БАКАЛАВРСКАЯ РАБОТА

на тему

ПРИЛОЖЕНИЕ ДЛЯ КОНТРОЛЯ
СОСТОЯНИЯ

ЗДОРОВЬЯ ПОЛЬЗОВАТЕЛЯ

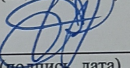
Студент


(подпись, дата)

23.06.2023

Махнаев Н.С.

Руководитель


(подпись, дата)

23.06.2023

Федюнин Р.Н.

Нормоконтролёр

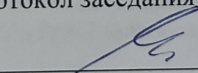

(подпись, дата)

23.06.2023

Бычков А.С.

Работа допущена к защите (протокол заседания кафедры от 2.06.23 г. № 14)

Заведующий кафедрой


(подпись)

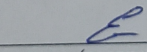
Митрохин М.А.

Работа защищена с отметкой удовлетворительно (протокол заседания ГЭК от

27.06.2023 г.

№ 30)

Секретарь ГЭК


(подпись)

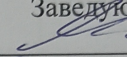
Гурин Е.И.

Пенза 2023

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
 М.А. Митрохин
«26» 04 2023г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Махнаев Никита Сергеевич гр. 19ВВП1 факультета вычислительной техники.
2. Тема работы «Приложение для контроля состояния здоровья пользователя»
Тема утверждена приказом ПГУ №458/о от "26" апреля 2023 г.
3. Руководитель бакалаврской работы к.т.н., доцент Федюнин Р.Н.

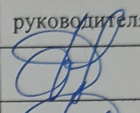
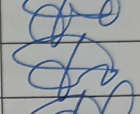
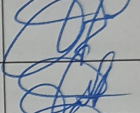
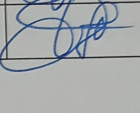
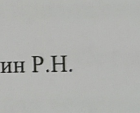
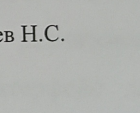
4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать приложение «Приложение для контроля состояния здоровья пользователя», выполняющее роль интегратора между пользователями: доктором, пациентом и администратором. Приложение должно быть с клиент-серверной архитектурой. Клиентская часть программы должна обладать графическим интерфейсом. Серверная часть должна включать функционал обработки информации о пациентах и лечебном персонале учреждения. Взаимодействие между частями приложения должно осуществляться с помощью *GET* и *POST* запросов. База данных должна работать по системе *CRUD*.

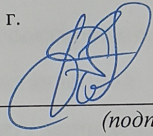
5. Перечень подлежащих разработке вопросов

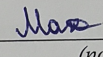
Требуется произвести анализ предметной области, изучить аналоги существующих приложений с подобным функционалом, изучить способы проектирования и разработки клиент-серверного приложения с применением *MVC* паттерна. Необходимо изучить язык программирования *PHP* и язык запросов *MySQL*.

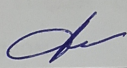
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Изучение предметной области	15%	26.04.2023-30.04.2023	
2	Анализ требований	4%	01.05.2023-02.05.2023	
3	Выбор технологий, применяемых при проектировании и разработке	15%	03.05.2023-07.05.2023	
4	Проектирование архитектуры и описание алгоритма функционирования программы	13%	08.05.2023-11.05.2023	
5	Разработка приложения для контроля состояния здоровья пользователя	45%	12.05.2023-29.05.2023	
6	Оформление пояснительной записки и презентации	8%	30.05.2023-31.05.2023	

Дата выдачи задания "26" апреля 2023 г.

Руководитель бакалаврской работы  Федюнин Р.Н.
(подпись, дата)

Задание к исполнению принял студент  Махнаев Н.С.
(подпись, дата)

Работу к защите допустить
Декан факультета  Фионова Л.Р.
(подпись, дата)
26.06.2023

Реферат

Бакалаврская работа содержит 91 лист, 38 рисунков, 3 таблицы, 5 источников и 3 приложения

ПАЦИЕНТ, КОНТРОЛЬ ЗДОРОВЬЯ, *МВС*, СУБД, *РНР*

Цель работы – разработать программу, позволяющую связать пациента и доктора с целью лечения и контроля здоровья пациента в режиме онлайн.

Объектами разработки являются приложение для сбора информации о пациенте и лечебном персонале в СУБД, с последующей возможностью записи пациента на прием, а также сущности и связи базы данных для хранения данных о пациентах, лечебном персонале и их взаимодействии с целью контроля здоровья пациентов.

В результате дипломного проектирования было разработано приложение, полностью соответствующее поставленной задаче, а именно: разработано приложение, позволяющее пациенту записаться на прием к врачу для лечения и контроля своего здоровья, разработана база данных, реализовано взаимодействие частей приложения, создан графический интерфейс для интерактивной работы с интернет ресурсом.

					ПГУ1.090301.14.001 ПЗ				
Изм	Лист	№ докум.	Подпись	Дата	Приложение для контроля состояния здоровья пользователя Пояснительная записка	Лит.	Лист	Листов	
Разраб.		Махнаев Н.С.					2	91	
Провер.		Федюнин Р.Н.				ФВТ, каф. ВТ гр.19ВВП1			
Н. Контр.		Бычков А.С.							
Утверд.		Митрохин М.А.							

Содержание

Перечень сокращений и обозначений.....	5
Введение.....	6
1 Приложение для контроля состояния здоровья пользователя	7
1.1 Анализ предметной области	Ошибка! Закладка не определена.
1.1.1 Актуальность.....	7
1.1.2 Пользователи и сценарии.....	7
1.1.3 Анализ аналогов.....	9
1.1.4 Выбор языка программирования	165
1.1.5 Выбор системы управления базой данных	19
1.1.6 Выбор СУБД	20
1.1.7 Выбор локальной платформы запуска проекта.....	20
1.1.8 Функциональные требования	231
1.1.9 Выводы	23
1.2 Проектирование и разработка	24
1.2.1 Разработка базы данных приложения.....	24
1.2.2 Разработка приложения	31
1.3 Технология размещения и запуска <i>WEB</i> приложения	34
1.3.1 Установка и среды языка <i>PHP</i>	34
1.3.2 Установка и проверка работоспособности <i>XAMPP</i>	38
1.3.3 Установка и проверка работоспособности <i>MySQL</i>	39
1.3.4 Установка и проверка работоспособности проекта в <i>IDE VSC</i>	443
2 Охрана труда.....	51
2.1 Санитарно-гигиенические стандарты.....	51
2.2 Классификация по опасности поражения электрическим током	511
2.3 Оборудование рабочего места.....	512
2.4 Освещенность рабочего места.....	513
2.5 Требования охраны труда в аварийных ситуациях.....	513
2.6 Контроль микроклимата.....	515

Закключение	598
Список используемых источников.....	609
Приложение А – <i>UML</i> -диаграммы	6160
Приложение А.1 – Диаграмма вариантов использования	6261
Приложение А.2 – Диаграмма развертывания.....	6362
Приложение А.3 – Диаграмма последовательности	6463
Приложение А.4 – Диаграмма данных	6564
Приложение Б – Листинг программы	6665
Приложение В – Презентация	80

Перечень сокращений и обозначений

В данной работе применяются следующие сокращения и обозначения

БД – база данных

Паттерн – шаблон проектирования

СУБД – система управления базами данных

ЭВМ – электронно-вычислительная машина

CRUD – акроним, обозначающий четыре базовые функции, используемые при работе с базами данных: создание (англ. *create*), чтение (*read*), модификация (*update*), удаление (*delete*)

HTML (Hyper Text Markup Language) – язык гипертекстовой разметки

MVC (Model-View-Controller) – «Модель-Представление-Контроллер» или «Модель-Вид-Контроллер»

UML (Unified Modeling Language) – унифицирован язык моделирования.

URL (Uniform Resource Locator) – система унифицированных адресов электронных ресурсов

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		5

Введение

Реализация программ для контроля состояния здоровья пользователя обширная и всегда актуальная тематика.

Часть таких программных продуктов реализуется как «умные» помощники или советники. Реализованные на основе мобильных устройств под управление *WatchOS* они работают в тандеме с датчиками и *API* мобильных устройств. Основная задача таких программ анализ суточной активности пациента с последующим анализом и советом ему и его врачам, что и как делать.

Другая часть программных продуктов данной тематики являются интерфейсами и интеграторами между врачом и пациентом, помогая каждой роли в поиске необходимой информации. Пациент благодаря такому интегратору находит специалиста медика для контроля и лечения своего здоровья. Врач находит и контролирует состояние пациента. Вплоть до того, что благодаря информации в системе доктор может вести ежегодную диспансеризацию пациента и отслеживать состояние его здоровья в течение длительного времени.

Целью данной дипломной работы является разработка второго типа программного продукта. Причина этого, программа интегратор может в своем функционале иметь *API* для взаимодействия с «умными» помощниками и обладает большим функционалом для контроля здоровья пациента в режиме онлайн и офлайн, так как ее пользователем является не один человек, а целая группа специалистов, готовых помочь пациенту сохранить и контролировать здоровье.

1 Приложение для контроля состояния здоровья пользователя

1.1 Анализ предметной области

1.1.1 Актуальность

Приложение для контроля состояния здоровья пользователя – это целая система, позволяющая врачам управлять информацией и данными, относящимися ко всем аспектам здравоохранения пациентов. Пациентам в свою очередь данное программное приложение, гарантирует быстрое и эффективное решение поставленных задач: записаться на прием, узнать информацию о враче или возможной консультации в поликлинике или больнице.

Системы управления базами данных «врач - пациент» были внедрены в середине 20 века году и с тех пор значительно эволюционировали благодаря возможности интеграции с существующими технологиями, программным обеспечением и персоналом лечебного заведения. Сегодня пациенты могут начать процесс получения медицинской помощи на первом этапе в приложении, и только на втором этапе процесс переходит к врачебному персоналу.

Учитывая большие объемы данных, количество вовлеченных в процесс контроля за пациентом людей, врачи имеют потребность в удобном программном обеспечении для управления данными и процессами лечения и контроля пациентов.

Программное обеспечение призвано уменьшить объем бумажной волокиты, сократить численность персонала на работах, которыми можно легко управлять в приложении, ускорить процессы лечения и диспансерного наблюдения, уменьшить количества ошибок, обеспечить конфиденциальность и безопасность данных.

1.1.2 Пользователи и сценарии

До проектирования и реализации приложения, требуется определить, какие конкретно пользователи будут пользоваться программным приложением

					ПГУ1.090301.14.001 ПЗ	
ИЗ	Лист	№ докум.	Подпись	Дата		7

и каким образом. В целях контроля здоровья основными пользователями могут

быть следующие категории людей: пациенты, врачи, администраторы (рисунок 1).



Рисунок 1 – Диаграмма деятельности

Определение пользователей поможет лучше сформулировать пользовательские сценарии.

Начнем с пользовательских сценариев для каждой группы.

Администратор:

- авторизация в программе;
- просмотр списка пользователей врачей и пациентов;
- получение сводки о записях на прием;
- добавление нового пользователя по требованию и выдача ему статуса:

врач или пациент.

Врачи:

- авторизация в приложении;

- посмотреть состояние здоровья ведомого пациента и контролировать его здоровье анализом показателей;

- посмотреть расписание возможных приемов пациентов;

- возможность принять или отклонить запись на прием.

Пациенты:

- авторизация в программе;

- посмотреть динамику своего здоровья на основании обезличенного отчета;

- записаться на прием к конкретному врачу на конкретную дату;

- возможность отменить свой прием к врачу.

На основании сценариев пользователей будут выдвинуты функциональные требования и реализована архитектура приложения.

1.1.3 Анализ аналогов

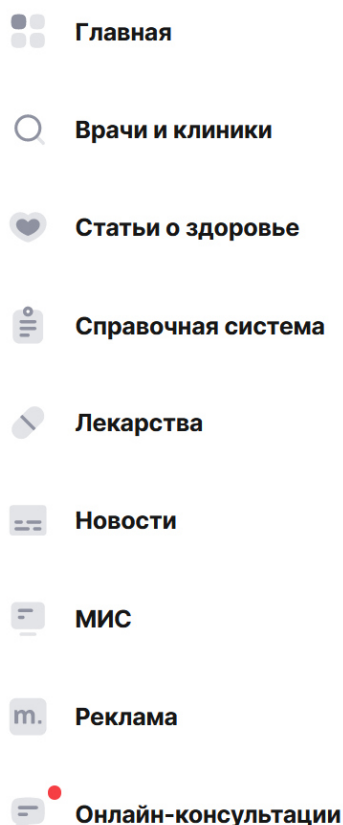
В рамках дипломной работы требуется провести сравнительный анализ уже существующих систем для определения их плюсов и минусов. Это позволит реализовать программное приложение с учетом опыта предыдущих разработок.

Таблица 1 Анализ аналогов

Анализатор	Тип реализации	Интерфейс	Контроль здоровья пациента	Стоимость
------------	----------------	-----------	----------------------------	-----------

<i>MedElement</i>	Облачный сервис с мощной справочной системой для слежения за здоровьем пациента	Громоздкий интерфейс	Автоматический, точный	От \$100000/год
<i>Medods</i>	<i>SaaS</i> платформа	Интуитивный интерфейс	Автоматический, точный	От \$10000/год
<i>Clinic365</i>	Анализ профиля ссылок	Интуитивный интерфейс	Полуавтоматический, точность проверяется вручную	От \$1000/год
Медучет <i>SQL</i>	Модульная программа	Простой интерфейс	Полуавтоматический, точность проверяется вручную	От \$10000 бессрочная лицензия

MedElement – это мощный инструмент для ведения всеобъемлющей работы и анализа здоровья пользователя системы. Пациент, становясь пользователем, получает все необходимые отчеты и карты прохождения диагностики и лечения (рисунок 2).



Медицинская информационная платфс
справочные системы для врачей и ме
о состоянии своего здоровья.

Цель проекта "МедЭлемент" - по

Рисунок 2 – Приложение для контроля состояния здоровья пользователя
MedElement

Функциональность *MedElement* дает возможность получить информацию, как врачу, так и пациенту из истории ведения пациента, а врачу и из медицинской базы знаний. Также имеется возможность подключения пациента к специальным датчикам. Однако, программа пока не поддерживает социальных сетей и нет интеграции с *API* почтовых сервисов.

Интерфейс *MedElement* достаточно громоздкий и понятен только опытным пользователем со стажем работы в программе. Он позволяет быстро

найти нужную информацию только благодаря системе умного помощника.

MedElement имеет множество аналитических инструментов, которые помогают не только контролировать здоровье, но и подсказывать дальнейший

ход профилактики или лечения. Также следует отметить, что анализаторы программы могут помочь выявить возможные проблемы здоровья. А врачу помочь с определением хода лечения.

Начальная цена подписки на *MedElement* составляет 100 000 долларов в год.

Преимущества:

- среди конкурентов обладает мощным функционалом работы с пациентами;
- имеет возможность рекомендации диагнозов;
- предоставляет рейтинг врачебного персонала;
- позволяет определить оптимальную карту лечения и контроля здоровья пациента.

Недостатки:

- стоимость программного обеспечения выше аналогов;
- нет интеграции с социальными сетями пациентов.

Вывод по анализу *MedElement*: программа является полноценным медицинским интегратором с возможностью контроля состояния здоровья пациента пользователя. Однако, цена на этот инструмент выше, что может не соответствовать бюджету многих учреждений.

Medods – это, программа для ведения пациента и анализа здоровья пользователя системы (рисунок 3).

Medods предоставляет высокую функциональность и гибкий инструментарий работы с пациентами. Он позволяет проводить анализ пациента несколькими врачами одновременно, формируя консилиум.

Медицинская информационная
система

MEDODS

Оставить заявку на
консультацию



Рисунок 3 – Приложение для контроля состояния здоровья пользователя
Medods

Основные функции *Medods* включают в себя: анализ состояния здоровья, рекомендации по профилактике, инструменты для периодической диспансеризации пользователей.

Интерфейс *Medods* интуитивно понятен и прост в использовании. Функции программы доступны благодаря наличию меню в верхней части экрана, что позволяет быстро находить нужные функции.

Приложение имеет инструменты аналитики, которые помогают контролировать здоровье пациентов, и имеет функционал для периодических диспансеризаций.

Цена подписки *Medods* составляет от 10000 долларов в месяц.

Преимущества:

- осуществляет мониторинг и анализ пациентов;
- обладает облачной базой данных;
- имеет встроенный поиск в БД по ключевым словам;
- способна осуществлять выгрузку информации о пациентах в ФГИС ФОМС.

Недостатки:

- стоимость услуг относительно высокая;
- не всегда обеспечивает корректную работу с ФГИС ФОМС.

Вывод по анализу *Medods* – это мощный инструмент ведения больных и пациентов, который может стать хорошим выбором для крупного учреждения. Программа обладает требуемой функциональностью и точностью выдачи результата о состоянии здоровья пациента.

Clinic365 – это еще один из иностранных аналогов приложения для контроля состояния здоровья пользователя и всестороннего обслуживания пациента (рисунок 4).



Рисунок 4 – Приложение для контроля состояния здоровья пользователя
Clinic365

Функциональность *Clinic365* также предоставляет инструменты для создания графика контроля пациентов. Основные функции *Clinic365* включают в себя: полная интеграция пациента в процесс анализа состояния и затем лечения.

Интерфейс *Clinic365* прост и удобен в использовании, и он легко понятен даже для новичков. Основной панелью управления встроенная в программу *CRM*.

Clinic365 является отличным инструментом контроля пациентов, так как содержит функции рассылки почтовых сообщений и интегрирована с *IP* телефонией для обзвона пациентов.

Начальная цена подписки на *Clinic365* составляет 100 000 долларов в год.

Преимущества:

- наличие интегрированной *CRM*-системы;
- настраиваемые опции работы с электронными медицинскими картами пациента;
- поддержка *IP*-телефонии.

Недостатки:

- пробный доступ учреждениям здравоохранения не предоставляется;
- отсутствует многофакторная авторизация.

В целом, *Clinic365* – это еще хороший аналог дорогим системам подобного уровня.

Медучет *SQL* – это еще один аналитический инструмент для анализа сайтов, который предлагает данные о трафике, привлекаемом сайтом, и его позициях в поисковой выдаче (рисунок 5).



Рисунок 5 – Приложение для контроля состояния здоровья пользователя
Медучет *SQL*

Медучет *SQL* состоит из базового блока, функционального ядра системы и набора дополнительных модулей. Особенностью является то, что информация и процессы выстраиваются вокруг пациента. Программа поддерживает электронные медицинские карты, историю болезней, инструменты ведения учёта пациентов и медицинских обследований.

Приложение удобно в использовании, программа имеет графический интерфейс приложения по ОС Windows 11.

Медучет *SQL* имеет встроенную карту ведения пациента, с возможностью фиксации состояния его здоровья.

Предоставляется бессрочная лицензия стоимостью 10000 долларов.

Преимущества:

- простой интерфейс;
- наличие модуля комплексного обследования пациента;
- имеет интеграцию с системами и устройствами слежения за состоянием пациента.

Недостатки:

- устаревший интерфейс;
- в качестве СУБД может выступать только *Microsoft SQL Server*.

В целом, Медучет *SQL* является полноценным комплексом для контроля состояния здоровья пользователя.

Анализ аналогов показал, что все существующие программные продукты — это дорогостоящие решения. Поэтому реализация простого и бесплатного аналога является актуальной темой дипломного проекта.

1.1.4 Выбор языка программирования

Существует множество языков программирования, которые могут использоваться для разработки веб-приложений (рисунок 6).

Рассмотрим некоторые из них и сравним определенным критериям.



Рисунок 6 – Языки программирования

Java.

Java является одним из самых популярных языков программирования для *WEB* разработки. *Java*-приложения запускаются на виртуальной машине, что обеспечивает высокую степень переносимости. Он также имеет множество библиотек для быстрого создавать *WEB* функционала.

Плюсы:

- имеет огромную базу экспертности;
- имеет большое количество библиотек и фреймворков;
- полностью объектно-ориентированный язык.

Минусы:

- длительная компиляция приложения;
- зависимость библиотек от версии *Java*;
- содержит много лишней информации и довольно громоздок.

Python.

Python имеет понятный синтаксис и подходит для быстрой разработки прототипов. *Python* поддерживает большое сообщество, библиотеки и фреймворки для *WEB* разработки.

Плюсы:

- высокий уровень абстракции;
- бесплатный инструмент и с открытым исходным кодом;
- имеет обширное комьюнити разработчиков;
- наличие библиотек для создания веб-приложений.

Минусы:

- интерпретируемый язык, не подходит в проектах, где важна скорость исполнения;
- недостаточное количество инструментов для многопоточности приложения.

Ruby.

Ruby это язык программирования для *WEB* разработки на основе фреймворка *Ruby on Rails*. Он имеет простой синтаксис и подходит для быстрой разработки прототипов.

Плюсы:

- простой и лаконичный синтаксис;
- большое количество библиотек от производителя;
- наличие поддержки многопоточности;
- фреймворки для создания веб-приложений.

Минусы:

- медленнее аналогов;
- проблемы при масштабировании проекта.

PHP.

PHP является одним из самых распространенных языков программирования для *WEB* разработки. Он хорошо интегрируется с *HTML* и

CSS, для создания динамических веб-сайтов.

Плюсы:

- простой в освоении язык программирования;
- наличие готовых библиотек и фреймворков;
- большое количество множество ресурсов для обучения.

Минусы:

- ограниченная поддержка многопоточности;
- проблемы с классическими вариантами отладки из-за интерпретатора.

Учитывая все выше сказанное, был выбран язык программирования *PHP* [1].

1.1.5 Выбор системы управления базой данных

Для организации структурированного хранения данных часто применяют БД. В случае выбора в качестве инструмента реализации логики приложения *PHP*, лучшим выбором будет использование *ORM* или прямых запросов *SQL*.

Приблизительная схема запроса к БД через *ORM* представлена на рисунке 7.



Рисунок 7 – Схема запроса к БД

В связи с тем, что требуется реализовать лаконичное и простое приложение независимое от изменений форм и библиотек *ORM*, было принято

решение в реализации использовать прямые запросы *SQL*.

1.1.6 Выбор СУБД

На сегодняшний день существуют различные виды баз данных и систем управления ими. Базы данных можно разделить на два основных класса: реляционные (*SQL*) и не реляционные (*NoSQL*). Основное отличие между ними заключается в представлении данных.

В не реляционных базах данных данные хранятся, как правило, в виде словарей, где ключом является первичный ключ сущности, а значением - все остальные поля таблицы. Такой подход позволяет абстрагироваться от строгой типизации и формата данных. Кроме того, *NoSQL*– базы данных могут хранить иерархические структуры данных, в то время как для реализации такой задачи в *SQL*-базах данных потребовалось бы создавать дополнительные таблицы. Самыми популярными *NoSQL* СУБД являются *MongoDB*, *HBase*, *Cassandra*.

Реляционные базы данных также известны как табличные. *SQL*-базы данных разделяются на два вида: строковые и колоночные. Основное отличие заключается в способе хранения данных.

В настоящий момент для реализации программ для контроля состояния здоровья пользователя реляционные БД эффективнее *NoSQL*.

Ниже перечислены некоторые из наиболее популярных СУБД и краткая информация о них.

MySQL.

MySQL является одной из самых популярных среди веб-приложений в связи с тем, что она открытая схема, бесплатная в использовании и легка в настройке [2].

Преимущества: открытая схема, бесплатная в использовании, легка в настройке, быстрая скорость работы.

Недостатки: могут возникнуть сложности при работе с большими объемами данных, ограниченный функционал в сравнении с другими СУБД, отсутствие устойчивости к сбоям.

PostgreSQL.

PostgreSQL – это СУБД с открытым исходным кодом, которая позиционируется как полнофункциональная альтернатива *Oracle* и *MySQL*. *PostgreSQL* поддерживает *ACID*-транзакции, доступ к объектам базы данных через язык программирования и хранилище данных.

Преимущества: открытый исходный код, большое количество функциональных особенностей, масштабируемость, хорошая поддержка языка *SQL*.

Недостатки: сложность в настройке, высокое потребление ресурсов, требовательность к аппаратным ресурсам.

SQL Server.

SQL Server является разработкой корпорации *Microsoft* и предназначен для работы на *Windows*-серверах. Она имеет широкие возможности, предназначенные для обработки и анализа данных, а также интеграции с другими продуктами *Microsoft*.

Преимущества: широкий функционал, интеграция с другими продуктами *Microsoft*, хорошая поддержка, отличное управление данными.

Недостатки: высокая стоимость, доступность только на *Windows*-серверах.

Учитывая плюсы и минусы рассмотренных аналогов, выбор был сделан в пользу *MySQL*.

1.1.7 Выбор локальной платформы запуска проекта

Для разработки и демонстрации *WEB* приложения требуется наличие локальной среды реализации и исполнения клиент-серверного приложения в формате *WEB*.

XAMPP, EasyPHP, Open Server, Winginx и *Denwer* – локальные серверные среды, используемые для разработки *WEB* приложений. Ниже приведено сравнение платформ.

XAMPP - популярный кроссплатформенный серверный дистрибутив, включающий *Apache, PHP, MySQL* и *Perl, PhpMyAdmin*. Его легко установить и настроить, и он предоставляет удобную панель управления для управления настройками сервера. *XAMPP* имеет большое сообщество пользователей и подходит как для начинающих, так и для опытных разработчиков.

EasyPHP — это быстрый и легкий серверный дистрибутив, включающий *Apache, PHP, MySQL* и *PhpMyAdmin*. Он прост в использовании и имеет дружелюбный пользовательский интерфейс. *EasyPHP* - вариант для тех, кому нужна простая и удобная локальная серверная среда без дополнительных настроек.

Open Server - это серверная среда на базе ОС *Windows*, включающая *Apache, NGINX, MariaDB, PHP* и другие компоненты. Он поставляется с удобной панелью управления и предлагает широкий спектр параметров настройки. *Open Server* - хороший вариант для разработчиков, которым нужна гибкая и настраиваемая среда.

Winginx — это серверная среда на базе ОС *Windows*, включающая *Apache, PHP, Nginx* и *MariaDB*. Он требует минимальной настройки и прост в использовании. *Winginx* - хороший вариант для разработчиков, которым нужна простая и быстрая серверная среда.

Denwer - русскоязычная серверная среда, включающая *Apache, PHP, MySQL, PhpMyAdmin* и другие компоненты. Имеет дружелюбный пользовательский интерфейс. *Denwer* специально разработан для российских пользователей и предоставляет документацию и поддержку на русском языке.

В связи с тем, что для реализации проекта требуется придерживаться принципа кроссплатформенности и наличия инструментов реализации, выбор остановлен на *XAMPP* [3].

1.1.8 Функциональные требования

Требования к приложению для контроля состояния здоровья пользователя на основе пользовательских сценариев, разработанных ранее:

- программа должна иметь мультипользовательский интерфейс и

поддерживать работу трех видов пользователей: администратор, врач, пациент;

- программа должна позволять пациенту войти в систему и осуществить весь перечень действий по карте здоровья: выбрать врача, записаться на прием, оставить сообщение в системе, иметь доступ к своим данным о здоровье;

- программа должна иметь функционал, который позволит врачу иметь возможность смотреть запись пациентов, возможность подтвердить или отменить запись, а также иметь карту ведения пациента;

- программа должна иметь функционал, который позволит администратору просматривать списки пользователей врачей и пациентов, добавлять, редактировать и удалять их.

1.1.9 Выводы

В ходе анализа предметной области были показаны и выявлены:

- актуальность выбранной темы;
- основные типы пользователей;
- функциональные требования к разрабатываемому приложению;
- инструменты для реализации проекта.

1.2 Проектирование и разработка

Приложение для контроля состояния здоровья пользователя состоит из серверного приложения и сервера базы данных, клиентом является браузер. Сервер получает от клиента запрос обрабатывает его с помощью роутера запросов и в зависимости от того *GET* или *POST* запрос, какие параметры запроса были использованы отправляет запрос в соответствующий контроллер обработки запросов. Далее в зависимости от запроса будет вызов базы данных, получение из нее результирующего ответа и формирование ответа в виде *HTML* страницы для браузера пользователя (рисунок 8).

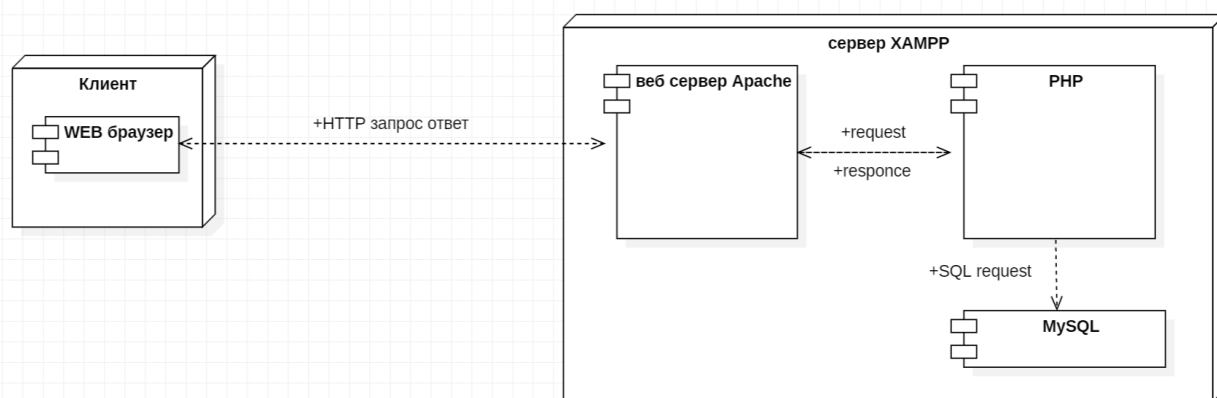


Рисунок 8 – UML диаграмма развертывания

Далее будут показаны шаги разработки базы данных и приложения.

1.2.1 Разработка базы данных приложения

Процесс разработки БД делится на проектирование (логическое и физическое) и реализация в СУБД *MySQL*.

Логический дизайн базы данных содержит описание определяемых объектов и того, как они связаны друг с другом и какие данные должны храниться в БД. Модель определяет, были ли собраны все функциональные требования к программе.

Физический дизайн базы данных определяет, как данные будут храниться в базе данных с использованием подходящей СУБД, и этот дизайн имеет прямое продолжение – схема БД в виде скриптов на языке *SQL*.

Реализуемая модель базы данных представляет собой схему, представленную на рисунке 9, и состоящую из четырех разделов: поликлиника и отделения поликлиники, врачи и график приема пациентов, пациенты и график приема врачей, отчеты и документы [4].

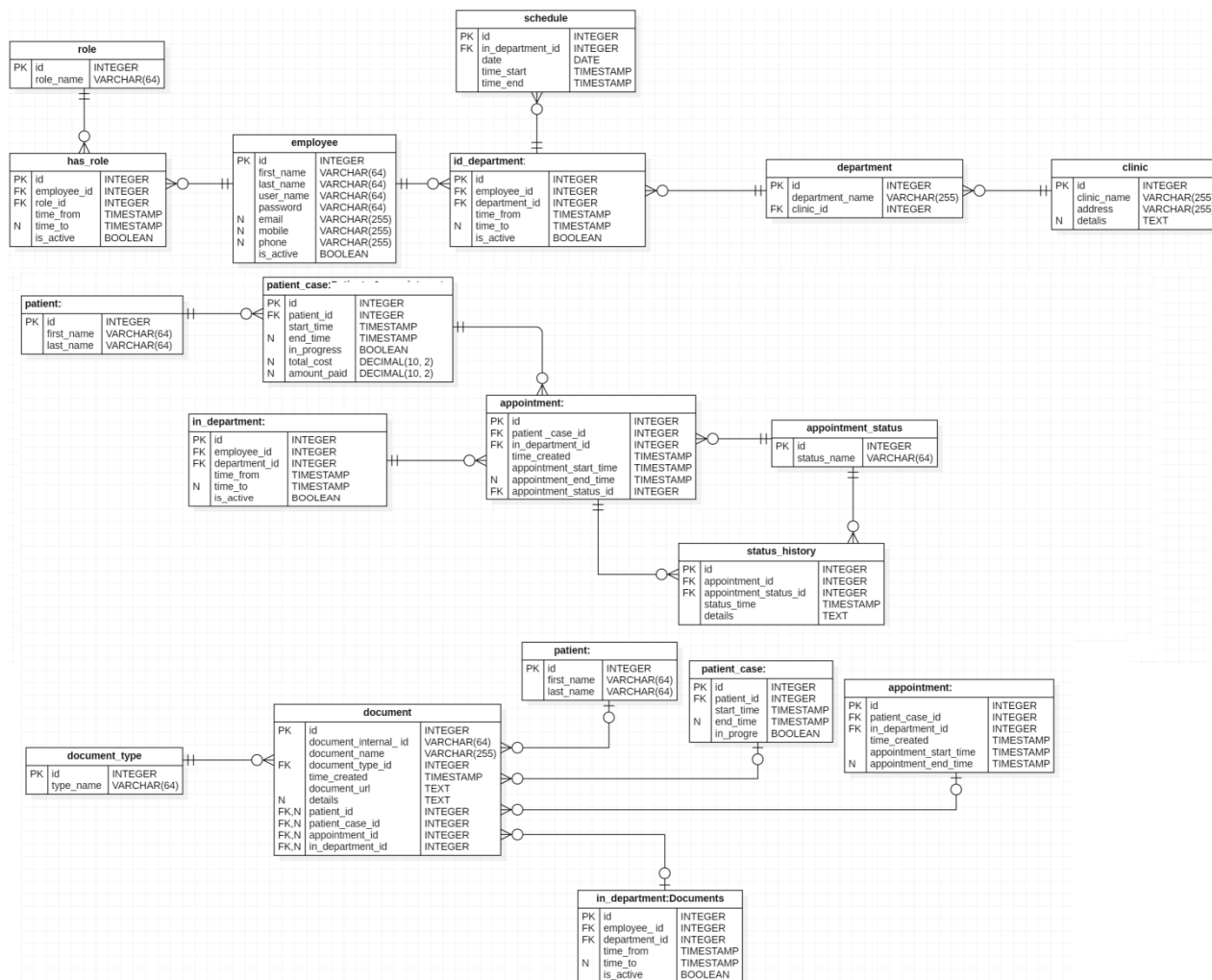


Рисунок 9 – Схема базы данных

Таблица 2.1 – Описание полей таблицы «Клиника»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>clinic_name</i>	<i>varchar(255)</i>	Обязательный
3	<i>adress</i>	<i>varchar(255)</i>	Обязательный

В таблице «Clinic» перечислены клиники. В данной таблице поле

clinic_name является уникальным атрибутом, в то время как атрибуты *address* и *details* используются для хранения местоположения клиники и иной информации в текстовом формате.

Таблица 2.2 – Описание полей таблицы «Отделения клиники»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>Int</i>	Первичный ключ, обязательный
2	<i>department_name</i>	<i>varchar(255)</i>	Обязательный
3	<i>clinic_id</i>	<i>Int</i>	Внешний ключ обязательный

Таблица «*Department*» необходима для хранения информации об отделениях клиники. Поле *department_name* и идентификатор клиники в данной таблице являются уникальными полями.

Таблица 2.3 – Описание полей таблицы «Врачи»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>first_name</i>	<i>varchar(64)</i>	Обязательный
3	<i>last_name</i>	<i>varchar(64)</i>	Обязательный
4	<i>user_name</i>	<i>varchar(64)</i>	Обязательный
5	<i>password</i>	<i>varchar(64)</i>	Обязательный
6	<i>email</i>	<i>varchar(255)</i>	Обязательный
7	<i>mobile</i>	<i>varchar(255)</i>	Необязательный
8	<i>phone</i>	<i>varchar(255)</i>	Обязательный
9	<i>active</i>	<i>boolean</i>	Обязательный

В таблице «*Employee*» содержится список всех сотрудников, работающих в клинике, независимо от роли. Для каждого сотрудника хранятся:

- *first_name* и *last_name* - имя и фамилия этого сотрудника;
- *user_name* - логин сотрудника для доступа к нашей системе;

- *password* - пароль, который пользователь будет использовать для доступа к системе;

- *email, mobile и phone* - контактные данные;

- *is_active* – признак активности пользователя в системе. Флаг имеет значение *true*, если сотрудник в данный момент работает в клинике.

Таблица 2.4 – Описание полей таблицы «Роль». Словарь

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>Int</i>	Первичный ключ, обязательный
2	<i>role_name</i>	<i>varchar(64)</i>	Обязательный

Таблица 2.5 – Описание полей таблицы «Отношения ролей»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>Int</i>	Первичный ключ, обязательный
2	<i>employee_id</i>	<i>varchar(64)</i>	Внешний ключ обязательный
3	<i>role_id</i>	<i>Int</i>	Внешний ключ обязательный
4	<i>time_from</i>	<i>TIMESTAMP</i>	Обязательный
5	<i>time_to</i>	<i>TIMESTAMP</i>	Обязательный
6	<i>is_active</i>	<i>boolean</i>	Обязательный

Таблица 2.6 – Описание полей таблицы «Расписание»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>Int</i>	Первичный ключ, обязательный
2	<i>in_department_id</i>	<i>varchar(64)</i>	Внешний ключ обязательный
3	<i>date</i>	<i>Int</i>	Обязательный
4	<i>time_start</i>	<i>TIMESTAMP</i>	Обязательный
5	<i>time_end</i>	<i>TIMESTAMP</i>	Обязательный

Таблица 2.7 – Описание полей таблицы «Пациенты»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>Int</i>	Первичный ключ, обязательный
2	<i>first_name</i>	<i>varchar(64)</i>	Обязательный
3	<i>last_name</i>	<i>varchar(64)</i>	Обязательный
4	<i>user_name</i>	<i>varchar(64)</i>	Обязательный
5	<i>password</i>	<i>varchar(64)</i>	Обязательный

Таблица 2.8 – Описание полей таблицы «Запись на прием к врачу»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>Int</i>	Первичный ключ, обязательный
2	<i>patient_case_id</i>	<i>Int</i>	Обязательный
3	<i>in_department_id</i>	<i>Int</i>	Обязательный
4	<i>time_created</i>	<i>TIMESTAMP</i>	Обязательный
5	<i>appointment_start_time</i>	<i>TIMESTAMP</i>	Обязательный
6	<i>appointment_end_time</i>	<i>TIMESTAMP</i>	Обязательный
7	<i>appointment_status</i>	<i>boolean</i>	Обязательный

Таблица 2.9 – Описание полей таблицы «Контроль здоровья пациента»

№	Имя поля	Тип	Примечание
1	<i>id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>patient_case_id</i>	<i>int</i>	Обязательный
3	<i>start_time</i>	<i>TIMESTAMP</i>	Обязательный
4	<i>end_time</i>	<i>TIMESTAMP</i>	Обязательный
5	<i>in_progress</i>	<i>boolean</i>	Обязательный

Таблица 2.10 – Описание полей таблицы «Статус записи на прием к врачу»

№	Имя поля	Тип	Примечание
1	<i>Id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>status_name</i>	<i>varchar(64)</i>	Обязательный

Таблица 2.11 – Описание полей таблицы «Истории записей на прием к врачу»

№	Имя поля	Тип	Примечание
1	<i>Id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>appointment_id</i>	<i>int</i>	Обязательный
3	<i>appointment_status_id</i>	<i>int</i>	Обязательный
4	<i>status_time</i>	<i>TIMESTAMP</i>	Обязательный
5	<i>details</i>	<i>TEXT</i>	Обязательный

Таблица 2.12 – Описание полей таблицы «Документ. Информация о здоровье»

№	Имя поля	Тип	Примечание
1	<i>Id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>document_internal_id</i>	<i>int</i>	Обязательный
3	<i>document_name</i>	<i>int</i>	Обязательный
4	<i>document_type_id</i>	<i>TIMESTAMP</i>	Обязательный
5	<i>time_created</i>	<i>TEXT</i>	Обязательный
6	<i>document_url</i>	<i>TEXT</i>	Обязательный
7	<i>details</i>	<i>TEXT</i>	Обязательный
8	<i>patient_id</i>	<i>int</i>	Обязательный
9	<i>patient_case_id</i>	<i>int</i>	Обязательный
10	<i>appointment_id</i>	<i>int</i>	Обязательный
11	<i>in_department_id</i>	<i>int</i>	Обязательный

Таблица 2.13 – Описание полей таблицы «Тип документа»

№	Имя поля	Тип	Примечание
1	<i>Id</i>	<i>int</i>	Первичный ключ, обязательный
2	<i>type_name</i>	<i>varchar(64)</i>	Обязательный

Таким образом, в процессе проектирования были получены следующие наборы сущностей.

Таблица 3 – Таблица сущностей

Название	Идентификатор	Описание
Лечебное заведение	<i>clinic</i>	Информация о лечебном учреждении
Отделение лечебного заведения	<i>department</i>	Информация о отделении в рамках лечебного учреждения
Врачи и сотрудники	<i>Employees</i>	Информация о врачах
Роли сотрудников	<i>role</i>	Информация о роли и должности врача
Словарь ролей	<i>has_role</i>	Таблица словарь ролей
Расписание приема врачей	<i>schedule</i>	Информация о приеме для врачей
Расписание приема пациентов	<i>appointment</i>	Информация о приеме для пациентов
Пациенты	<i>patient</i>	Информация о пациентах
Документы о	<i>documents</i>	Информация о документах врачей

Название	Идентификатор	Описание
состоянии здоровья		
Типы документов	<i>document_type</i>	Таблица словарь документов

1.2.2 Разработка приложения

Компоненты реализованного приложения (рисунок 10) были спроектированы на основании следующих принципов: независимость от фреймворка, тестируемость, независимость от *UI*, независимость от базы данных.

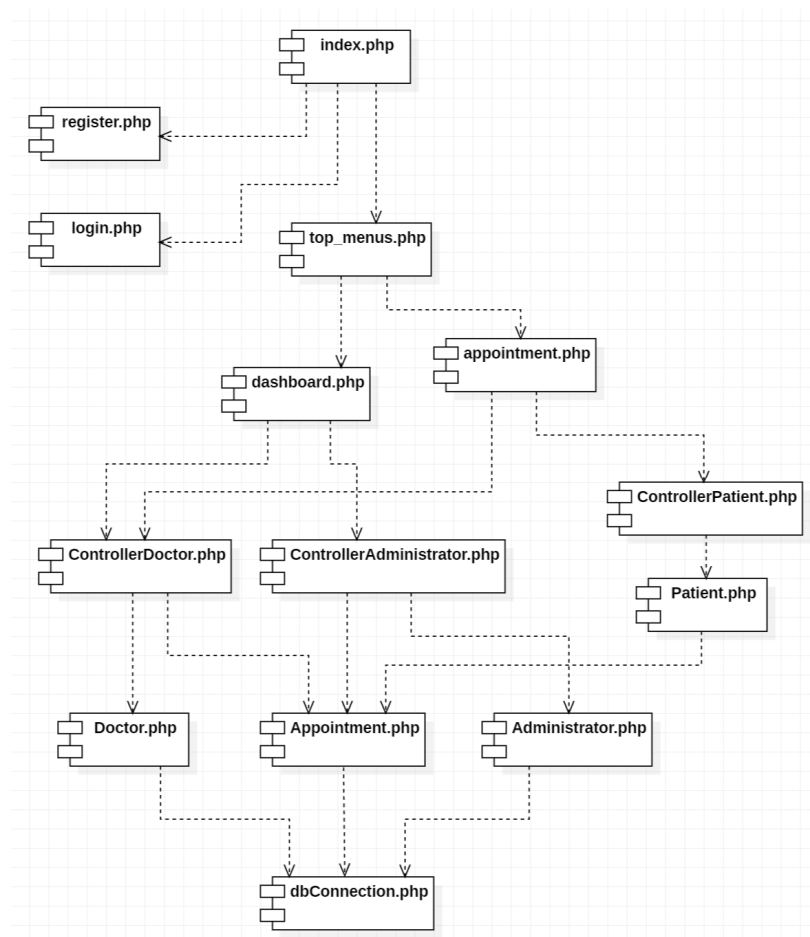


Рисунок 10 – Диаграмма компонентов системы

Достигается гибкость компонентной модели приложения за счет разделения приложения на слои *Service*, *Repository*, *Model*. В результате использования подхода *MVC* получаются следующие слои приложения:

- *View* - выводит данные клиенту, визуализирует состояние логики на интерфейс;
- *Controller* - отвечает за взаимодействие с пользователем посредством функций ввода-вывода;
- *Service* - отвечает за бизнес логику и ее использование между компонентами;
- *Repository* - отвечает за получение данных из источников: БД, *API*;
- *Models* - отвечает за перенос данных между слоями и за логику обработки этих данных.

Для отображения взаимодействия компонентов между собой на рисунке 11 представлена диаграмма последовательности (роль доктор) [5].

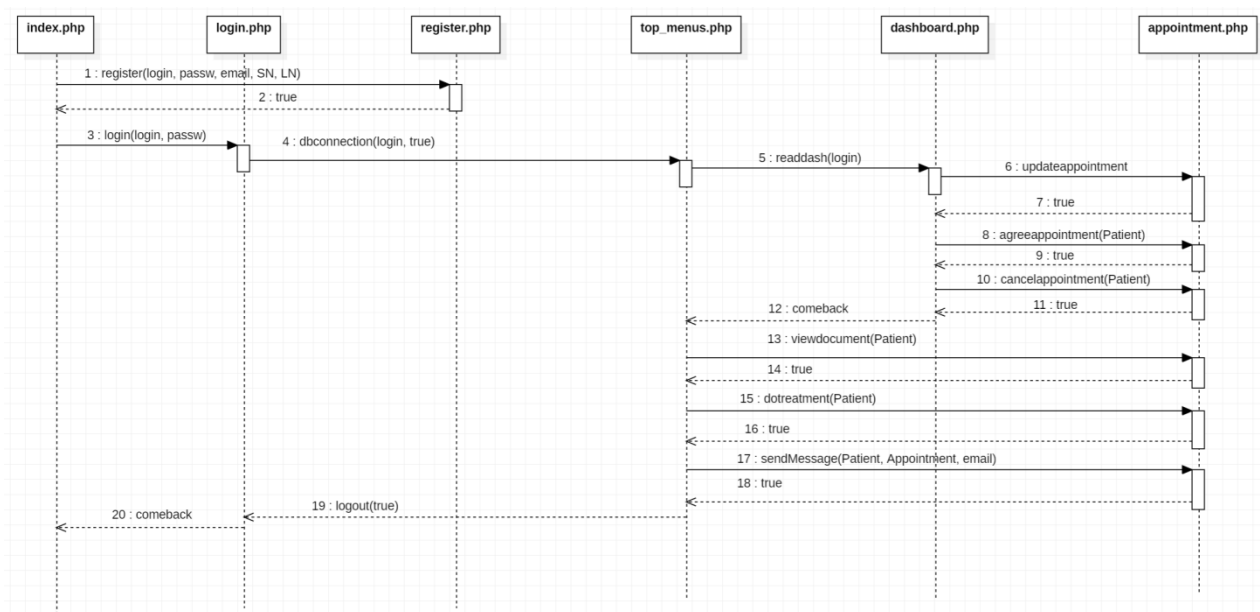


Рисунок 11 – Диаграмма последовательности

При входе в систему пользователь попадает на главную страницу *index.php*, где пользователю предлагают авторизоваться или зарегистрироваться. При выборе регистрации, пользователь попадает на

страницу регистрации *register.php*, осуществив данную функцию, пользователь возвращается на страницу *index.php*, где делает переход на страницу *login.php*. В данном компоненте отрабатывается код проверки логина и пароля:

```
$loginMessage = '';
if(!empty($_POST["login"]) && !empty($_POST["email"]) &&
!empty($_POST["password"]) && !empty($_POST["loginType"]) &&
$_POST["loginType"]) {
    $user->email = $_POST["email"];
    $user->password = $_POST["password"];
    $user->loginType = $_POST["loginType"];
    if($user->login()) {
        header("Location: dashboard.php");
    } else {
        $loginMessage = 'Пароль не верен';
    }
} else {
    $loginMessage = 'Поля не заполнены';
}
```

Если данные совпадают, пользователь переходит на страницу *top_menus.php* и работает с пунктами меню выбирая или панель приборов *dashboard.php*, где просматривает пациентов, документы и информацию контроля здоровья пациентов или переходит в компонент *appointment.php* для просмотра записанных к нему пациентов, для того чтобы принять или отклонить их.

Для этого сначала будет получен список записей:

```
$appointment = new Appointment($db);
if(!empty($_POST['action']) && $_POST['action'] ==
'listAppointment') {
    $appointment->listAppointment();
}
```

Затем если запись подтверждается, информация о ней сохраняется в БД с соответствующей пометкой активности

```
public function agreeappointment(){
    if($this->doctor_id && $this->specialization_id) {

        $stmt = $this->conn->prepare("
            INSERT INTO ".$this->appointment>("patient_id`,
            `specialization_id`, `doctor_id`, `consultancy_fee`,
            `appointment_date`, `appointment_time`, `status`)
            VALUES(?,?,?,?,?,?,?)");
        $this->doctor_id = htmlspecialchars(strip_tags($this-
            >doctor_id));
```

```

        $this->specialization_id =
htmlspecialchars(strip_tags($this->specialization_id));
        $this->fee = htmlspecialchars(strip_tags($this-
>fee));
        $this->appointment_date =
htmlspecialchars(strip_tags($this->appointment_date));
        $this->appointment_time =
htmlspecialchars(strip_tags($this->appointment_time));
        $this->status = htmlspecialchars(strip_tags($this-
>status));
        $stmt->bind_param("iiiisss", $_SESSION["userid"],
$this->specialization_id, $this->doctor_id, $this->fee, $this-
>appointment_date, $this->appointment_time, $this->status);
        if($stmt->execute()){
            return true;
        }
    }
}

```

В случае отклонения записи она больше не будет показана на странице *appointment.php*.

Для контроля здоровья пациентов имеется возможность фиксировать информацию и отсылать напоминания пациенту на *email*.

1.3 Технология размещения и запуска WEB приложения

1.3.1 Установка и среды языка *PHP*

Для инсталляции языка *PHP* на локальный ПК под управлением ОС *Windows 11* требуется скачать с официального сайта разработчиков инсталлятор.

Для того чтобы установить *PHP* в ОС *Windows 11* понадобится предустановленные инструменты: среда исполнения *Visual C++ Redistributable for Visual Studio 2021*, которую требуется скачать с официального сайта компании *Microsoft*. После установки *Visual C++ Redistributable* скачиваем архив *PHP* для ОС *Windows (Thread Safe x64)* с официального сайта *PHP*.

Содержимое полученного архива требуется распаковать в папку на жестком диске, по умолчанию рекомендуется установить программу в каталог *C:\PHP*.

После распаковки нужно добавить каталог *PHP* и исполнительные файлы *PHP* в системную переменную *PATH* для того, чтобы можно было запускать *PHP* из любого каталога, для этого:

- нажимаем *win+x* и переходим в настройки *System* ОС *Windows 11* (рисунок 12);
- переходим в *Advanced system settings* (рисунок 13);
- выбираем *Environment Variables* (рисунок 14);
- выбираем переменную *PATH* и нажимаем кнопку *Edit* (рисунок 15);
- нажимаем кнопку *New* и вводим адрес каталога с установленным *PHP*, например, *C:\PHP*;
- нажимаем кнопку *ОК*.

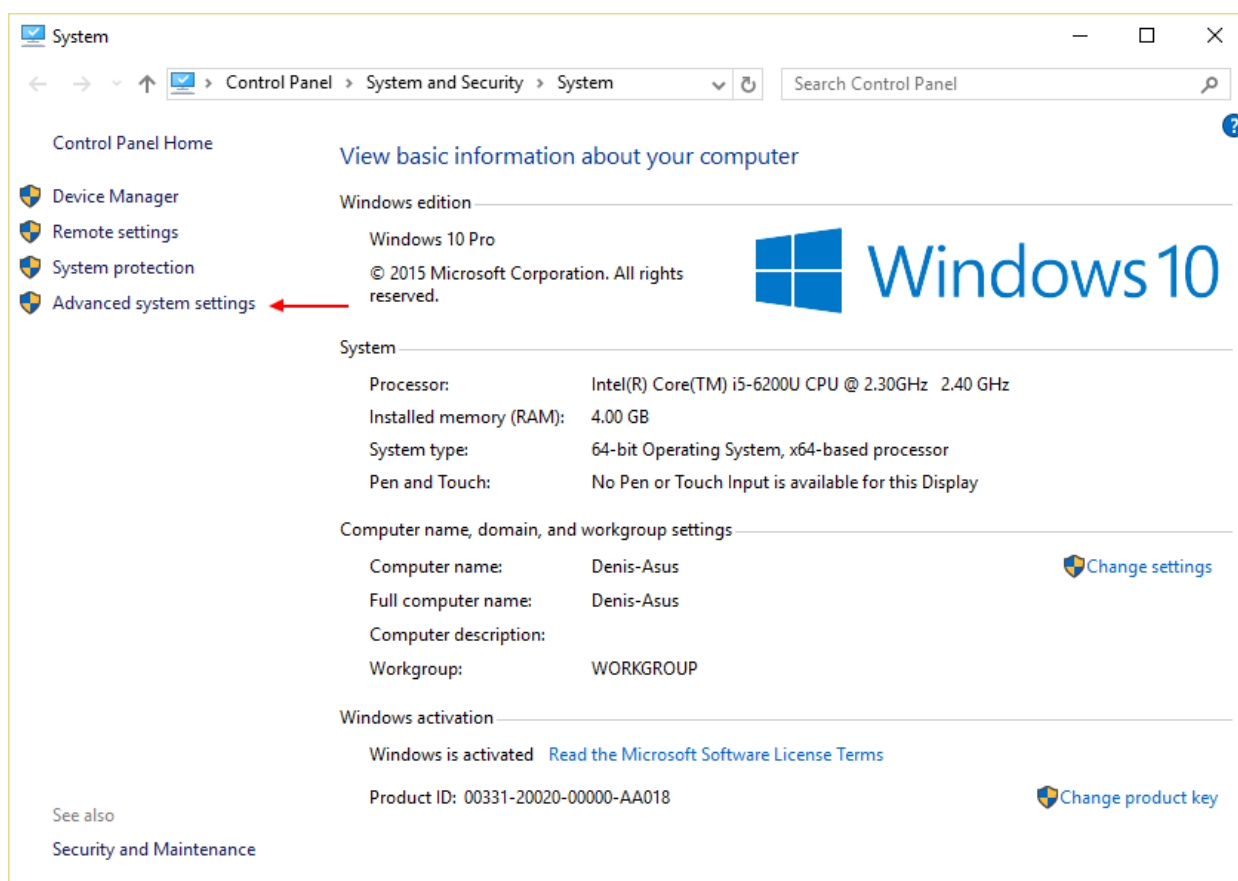


Рисунок 12 – Панель «Настройки сети»

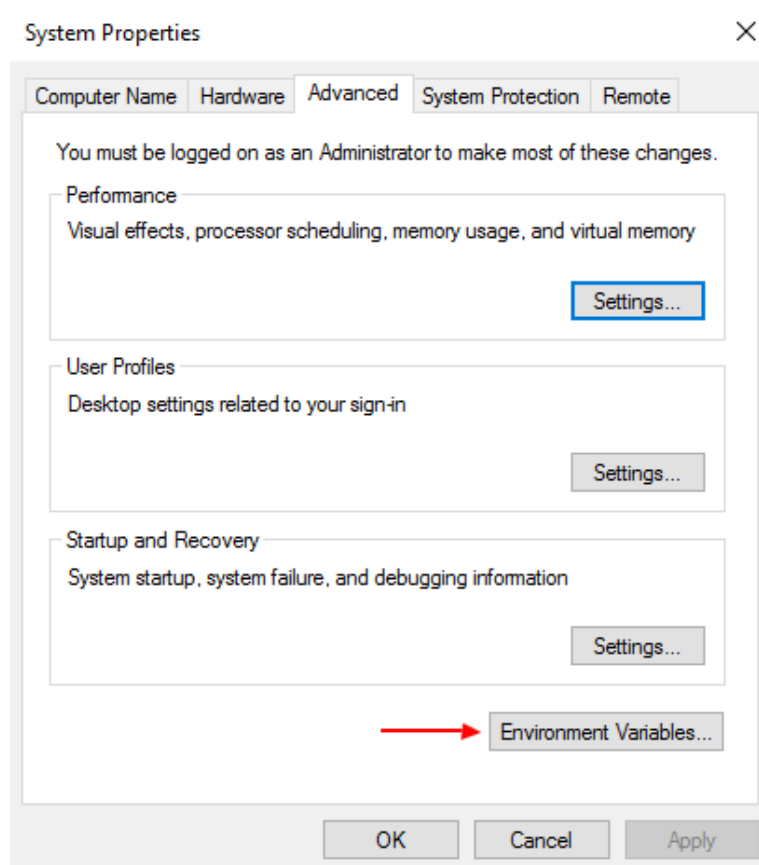


Рисунок 13 – Диалог Переменные окружения

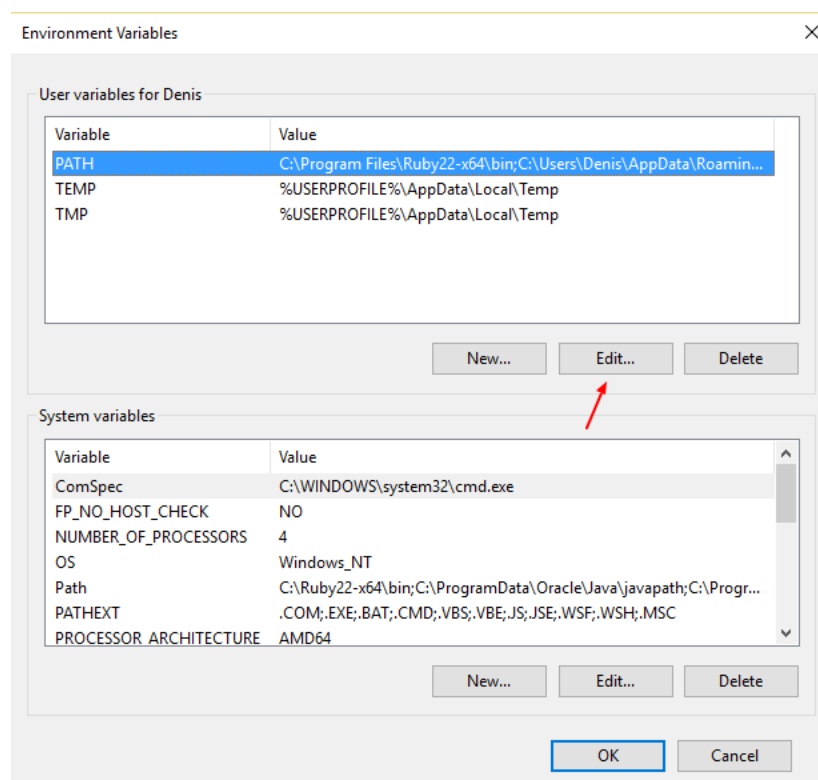


Рисунок 14 – Переменная окружения *PATH*

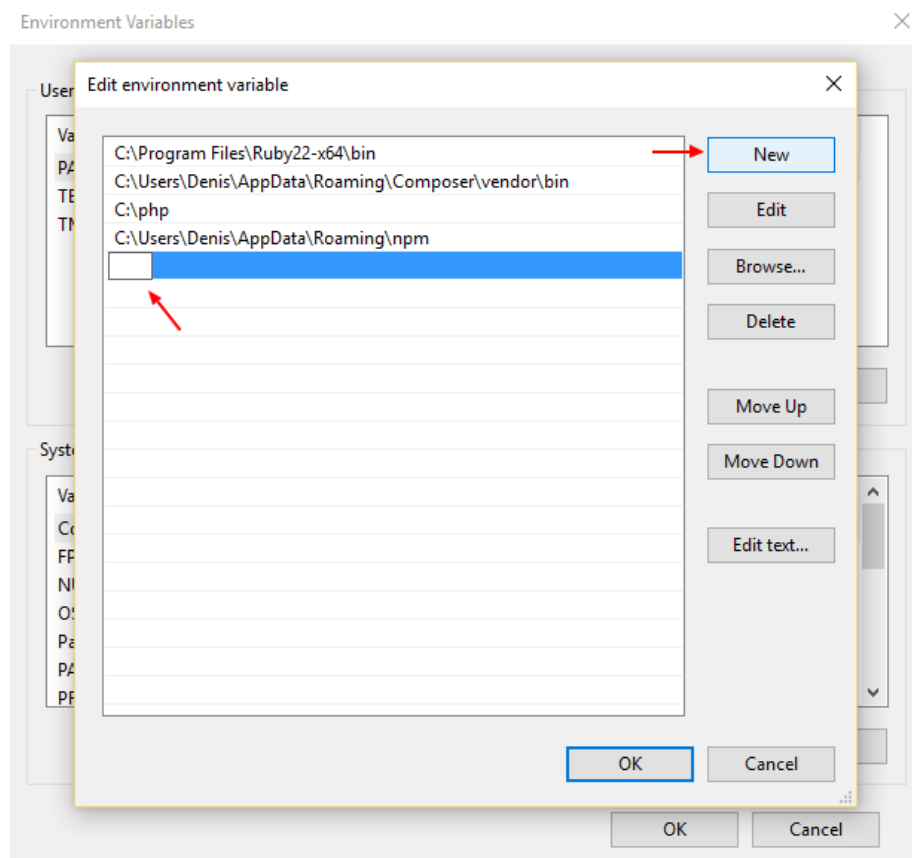


Рисунок 15 – Редактирование переменной окружения *PATH*

После настройки переменных среды нужно настроить *PHP* для работы с *WEB* окружением и СУБД *MySQL*, для чего требуется найти в каталоге *PHP* файл настроек *php.ini*, открыть данный файл и убрать комментарии с переменных исполнения:

- *extension=php_curl.dll;*
- *extension=php_gd2.dll;*
- *extension=php_mbstring.dll;*
- *extension=php_openssl.dll;*
- *extension=php_pdo_mysql.dll;*
- *extension=php_pdo_sqlite.dll;*
- *extension=php_sqlite3.dll.*

Проверка работоспособности языка *PHP* на локальном хосте под управлением ОС Windows 11 показана на рисунке 16.

```

C:\Windows\System32\cmd.e x + v
Microsoft Windows [Version 10.0.22621.1702]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

c:\php>npm -v
6.14.18

c:\php>php -v
PHP 7.4.0 (cli) (built: Nov 27 2019 10:14:18) ( ZTS Visual C++ 2017 x64 )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies

c:\php>

```

Рисунок 16 – Проверка работоспособности *PHP*

После успешной инсталляции инструментария, необходимо проверить работоспособность *XAMPP*.

1.3.2 Установка и проверка работоспособности *XAMPP*

Для установки среды исполнения *XAMPP* требуется скачать инсталлятор с официального сайта, пройти стандартные этапы инсталляции, после чего панель управления *WAMP* сервером автоматически будет запущена в ОС *Windows 11* (рисунок 17).

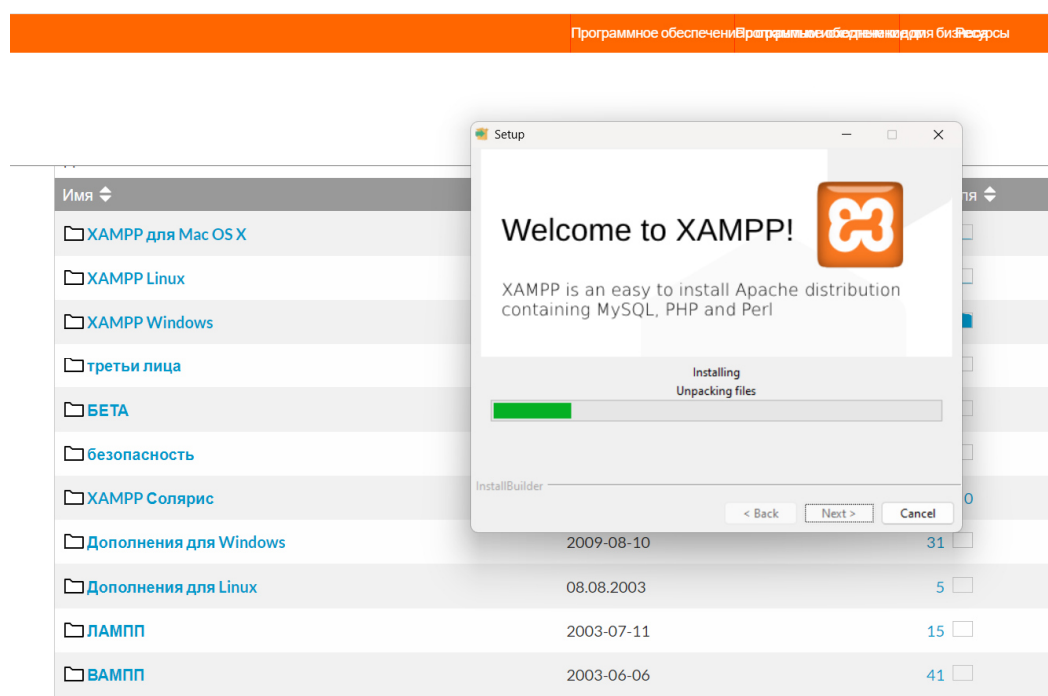


Рисунок 172 – Главная страница

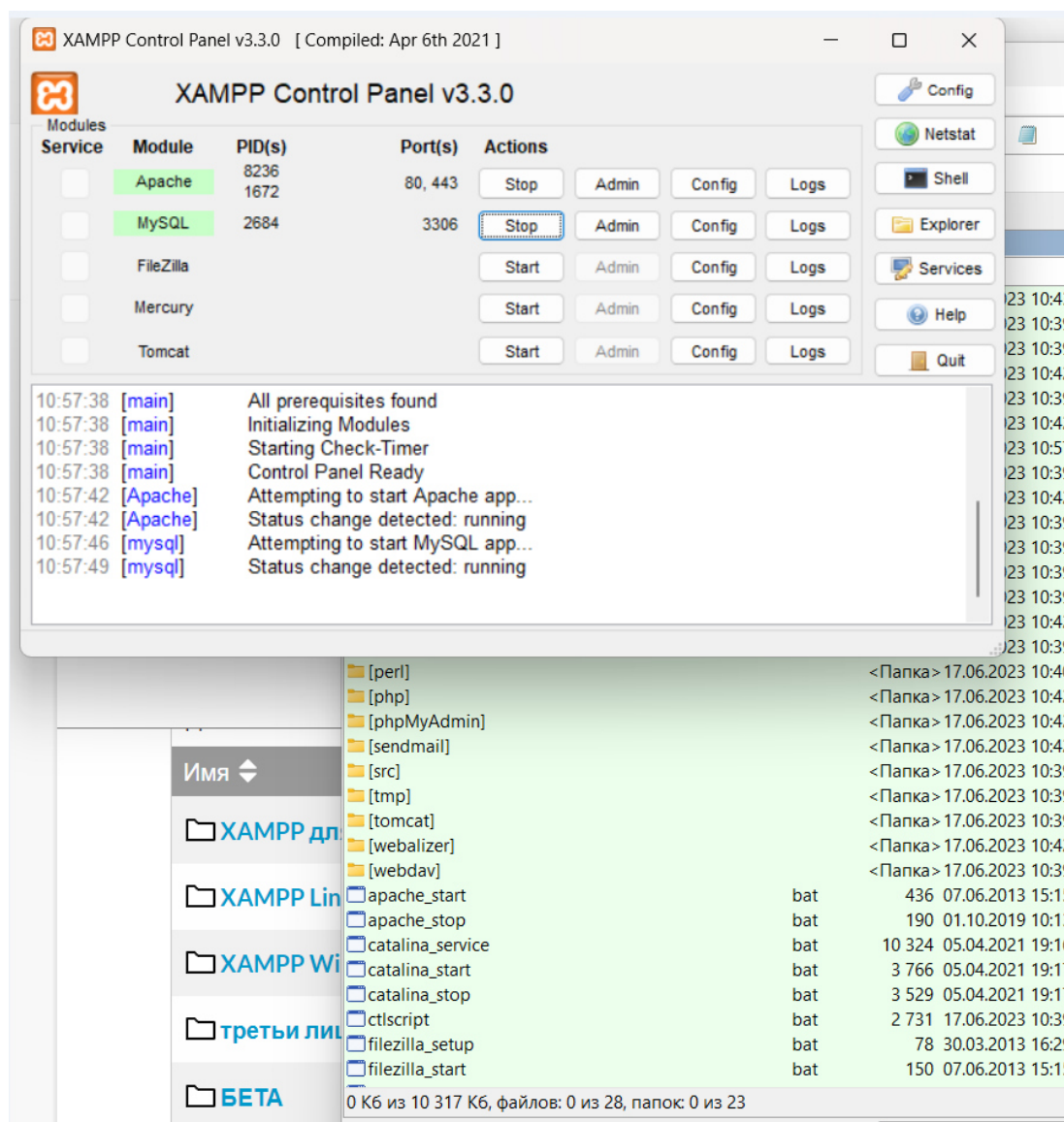


Рисунок 18 – Результаты сканирования

Для локальной работы достаточно чтобы модули *Apache* и *MySQL* были в запущенном состоянии (рисунок 18).

1.3.3 Установка и проверка работоспособности *MySQL*

Для наличия работоспособной версии *MySQL* на локальном хосте под управлением ОС *Windows 11* требуется скачать инсталлятор с официального сайта (рисунок 19).

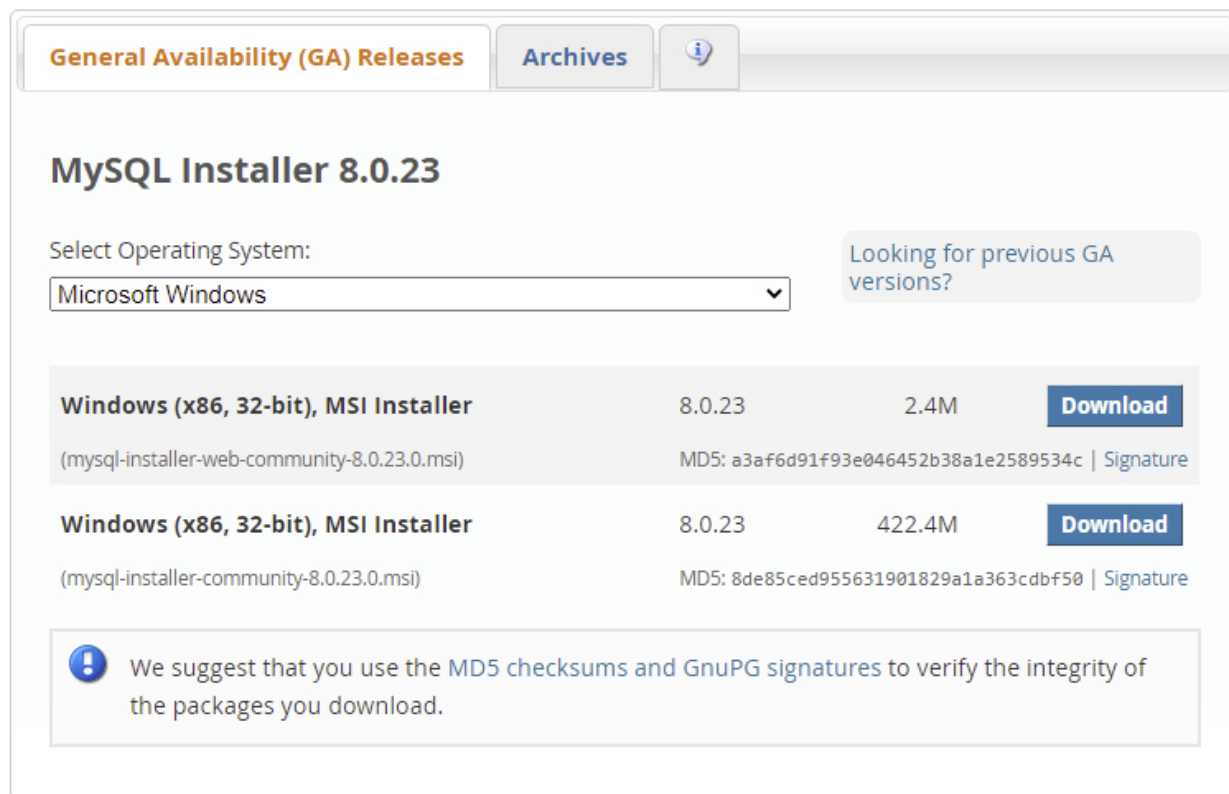


Рисунок 19 – Инсталляция MySQL

Далее требуется следовать строго инструкциям меню инсталляции и по шагам выполнять настройки устанавливаемого программного продукта *MySQL* (рисунки 20 - 21):

- выбор компонентов для инсталляции (рисунок 20);
- выбор компонентов настроек файрвола и рабочего порта для СУБД *MySQL* (рисунок 21);
- проверка доступности СУБД *MySQL* из командной строки ОС *Windows 11* (рисунок 22).

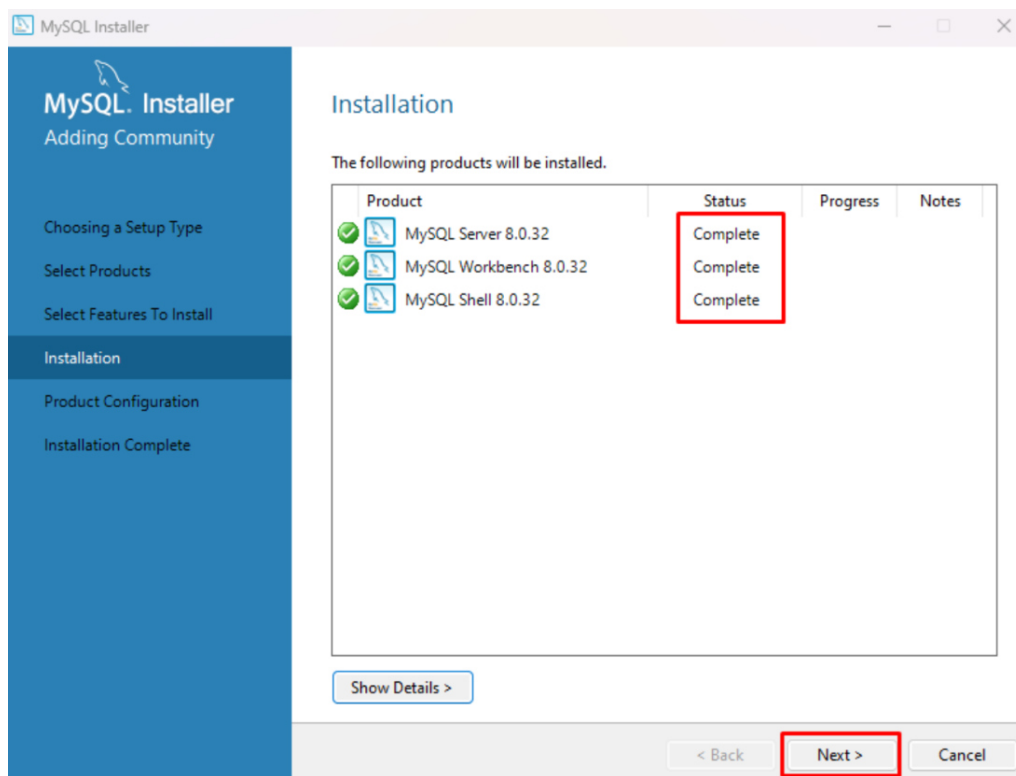


Рисунок 203 – Выбор компонентов для инсталляции *MySQL*

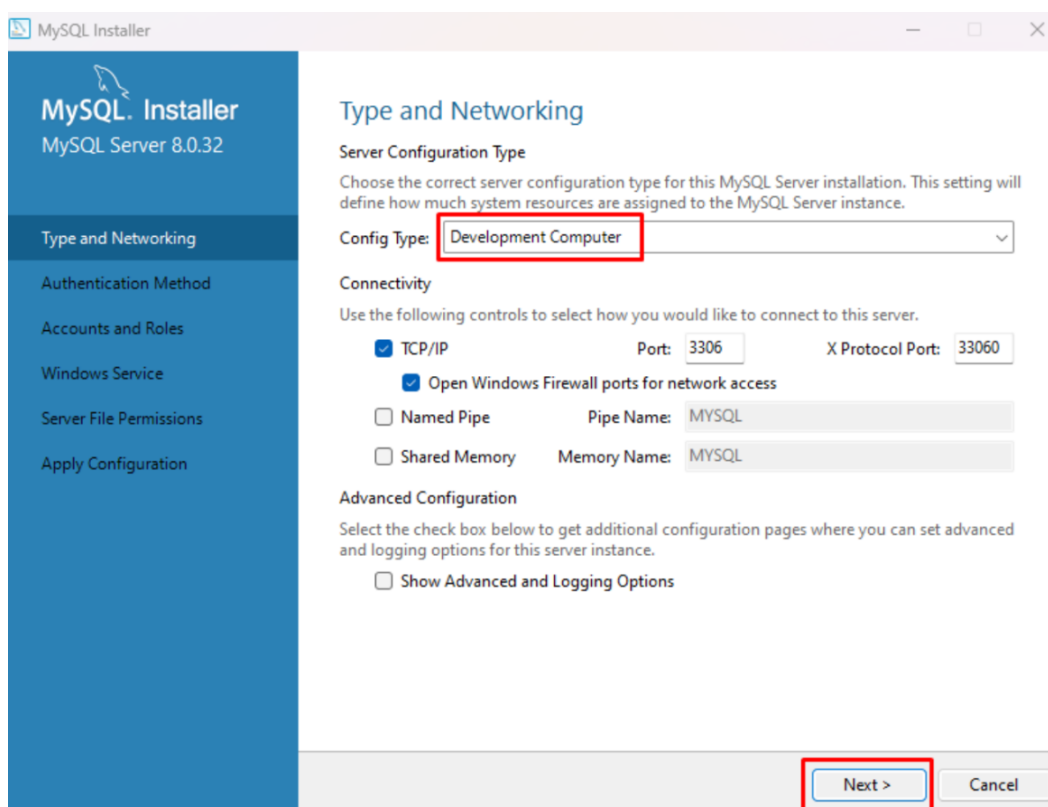
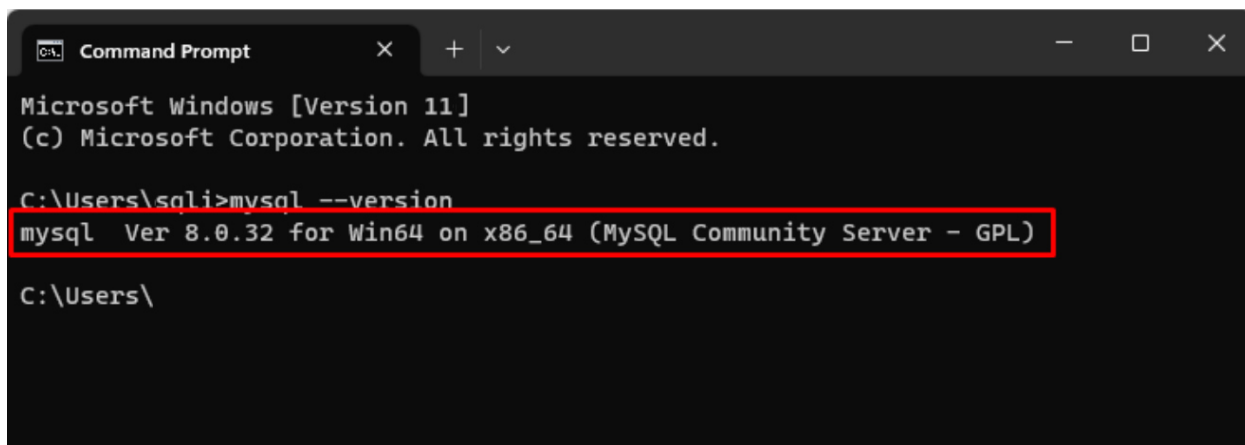


Рисунок 21 – Настройка среды исполнения во время инсталляции *MySQL*



```
Microsoft Windows [Version 11]
(c) Microsoft Corporation. All rights reserved.

C:\Users\sqli>mysql --version
mysql Ver 8.0.32 for Win64 on x86_64 (MySQL Community Server - GPL)

C:\Users\
```

Рисунок 22 – Проверка работоспособности инсталляции MySQL

Для создания базы данных в СУБД *MySQL* необходимо использовать инструмент *phpMyAdmin* (рисунок 23). Для этого в браузере необходимо ввести наименование хоста и инструмента.

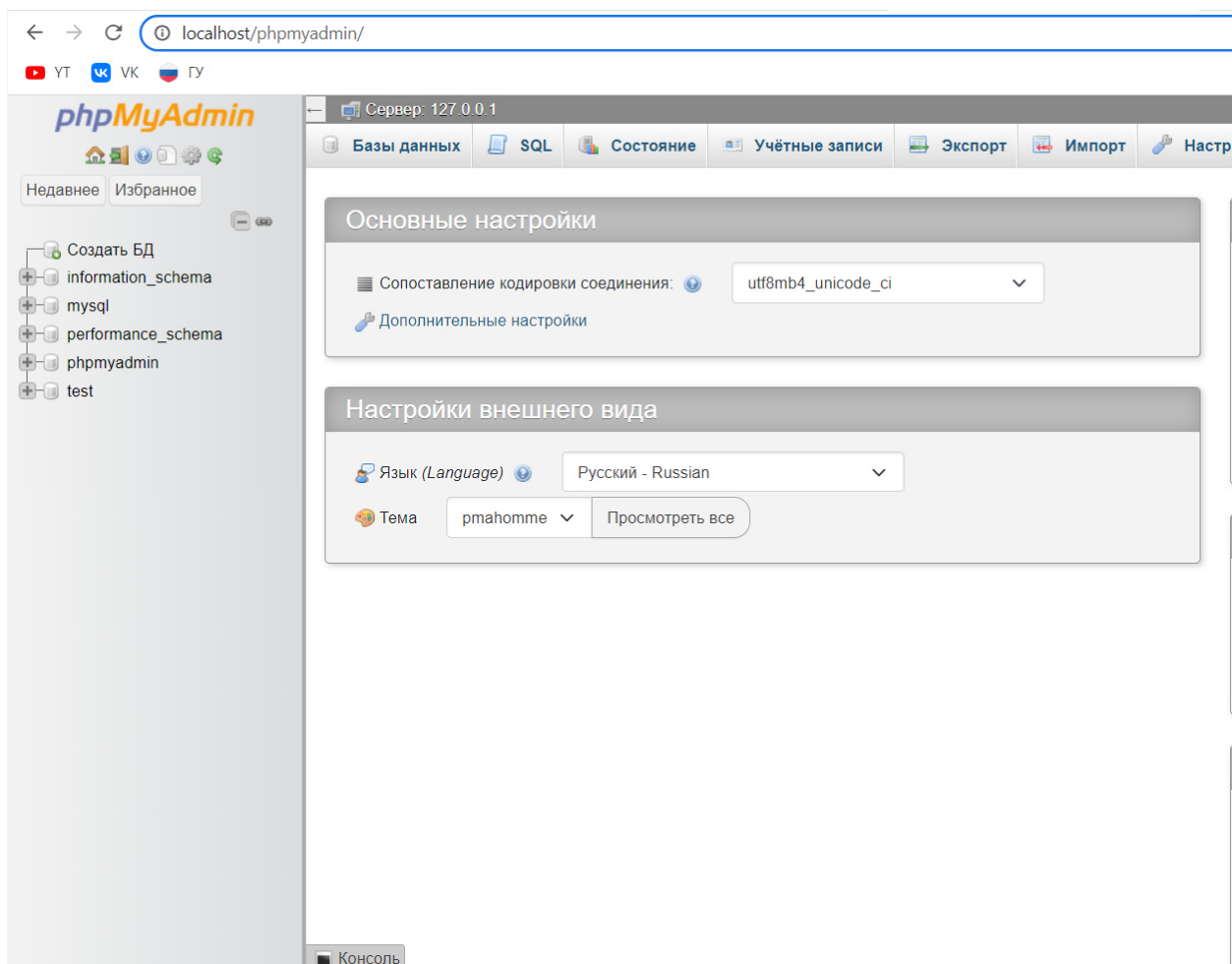


Рисунок 23 – Вызов *phpMyAdmin*

В программной среде *phpMyAdmin* требуется создать базу данных, с которой будет связан проект (рисунок 24, 25).

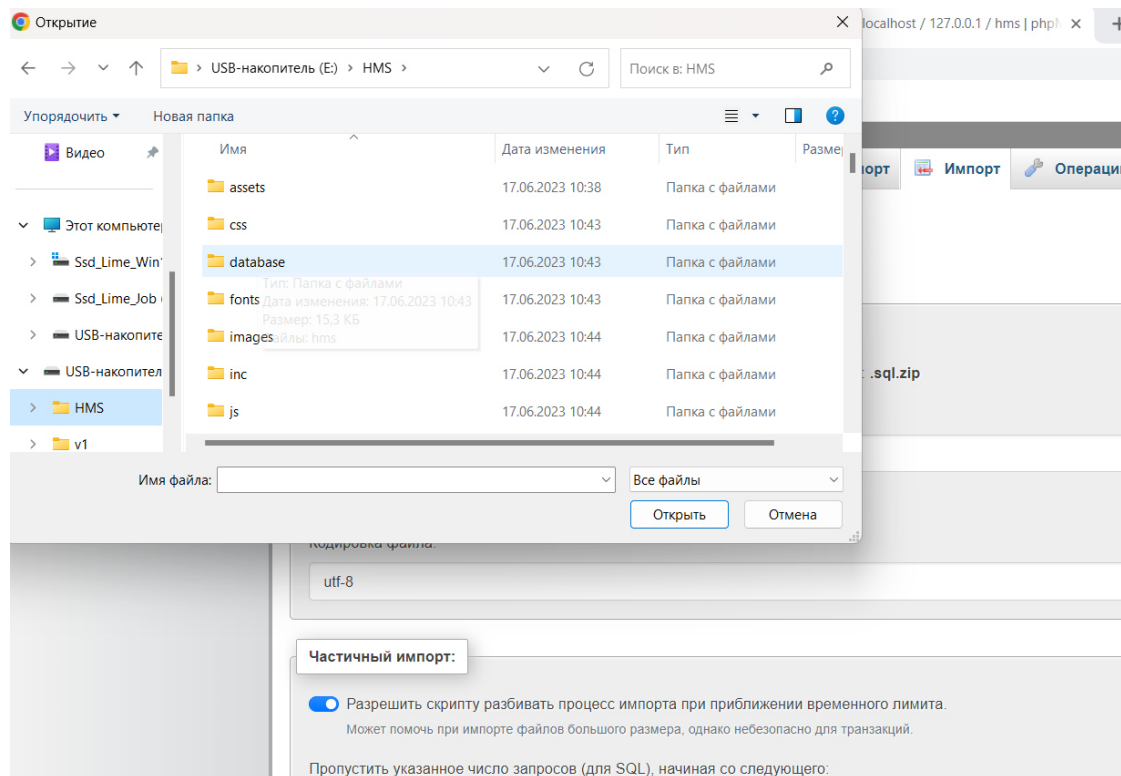


Рисунок 24 – Выбор схемы БД

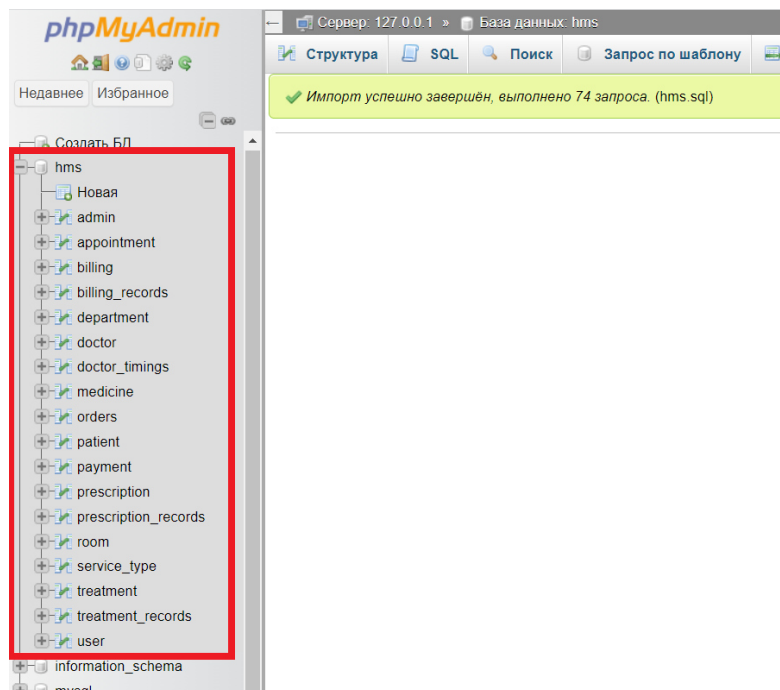


Рисунок 25 – Просмотр созданной БД

1.3.4 Установка и проверка работоспособности проекта в *IDE VSC*

Для реализации проекта необходимо использовать среду разработки *IDE Microsoft Visual Studio Code*. Для создания проекта требуется открыть рабочий каталог проекта в среде разработки (рисунок 26).

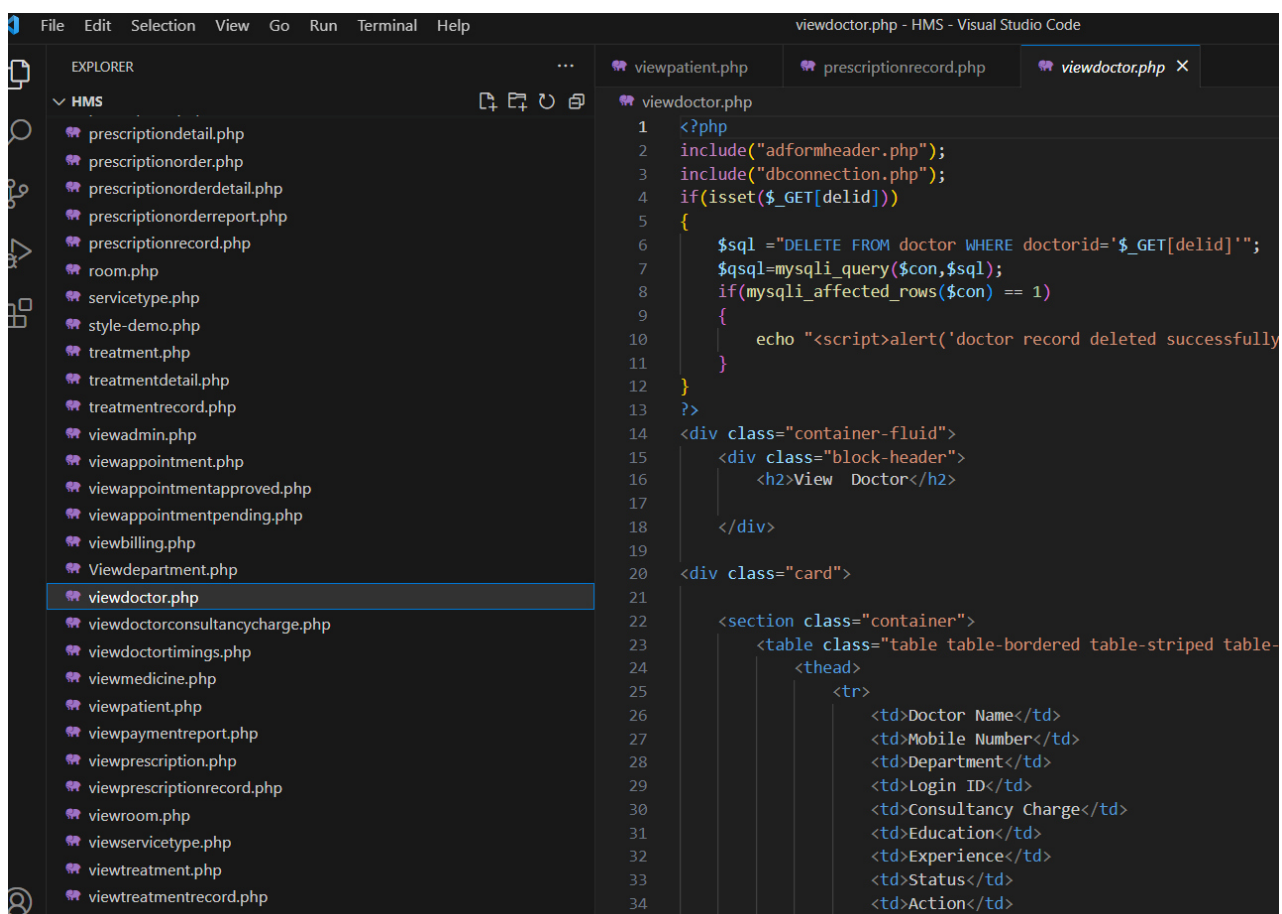


Рисунок 26 – Структура проекта в IDE Microsoft Visual Studio

Выделением файлов проекта просмотрим их содержимое (рисунок 27). Далее для запуска проекта воспользуемся консолью, и наберем команду *php -S 0.0.0.0:7777*, которая должна запустить наш проект на исполнение. Но в результате будет показана ошибка соединения с БД (рисунок 28). Для ее исправления изучим содержимое файла запуска и соединения с БД *php.ini* (рисунок 29). Как показал код ошибки для корректной работы приложения требуется снять комментарии с команды *execution= mysql*, для корректного соединения с СУБД из кода приложения.

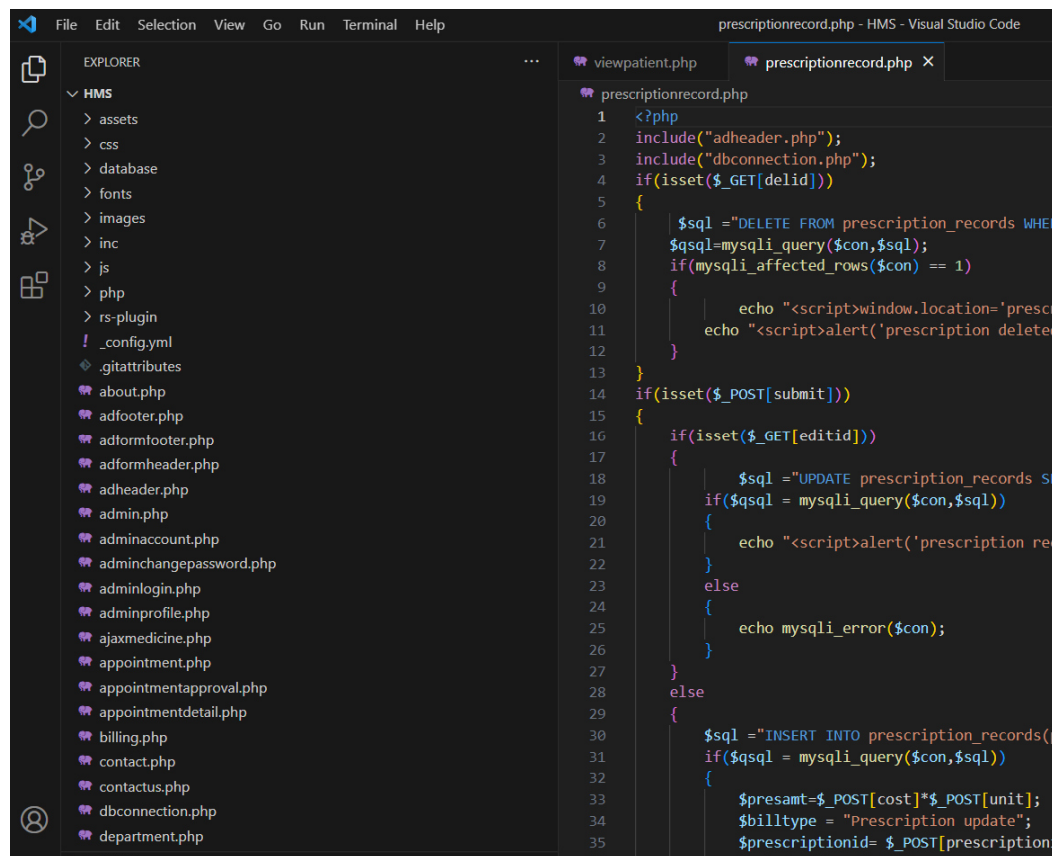


Рисунок 27 – Просмотр файла с кодом в IDE Microsoft Visual Studio

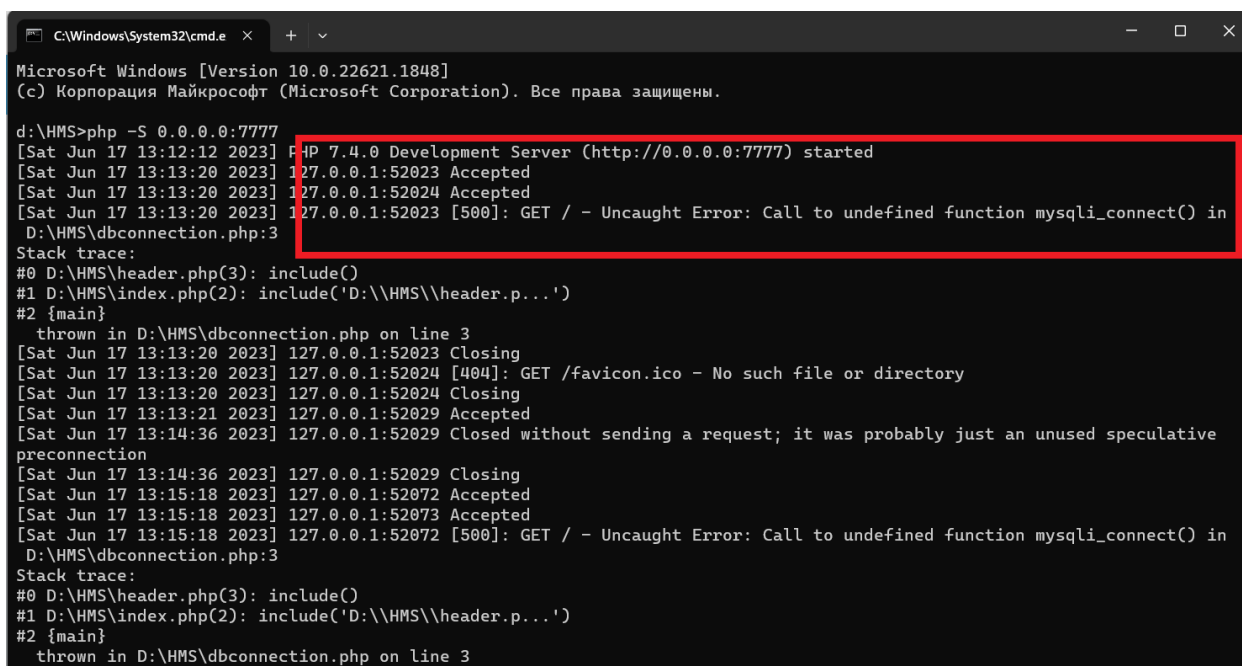


Рисунок 28 – Ошибка исполнения проекта при запуске

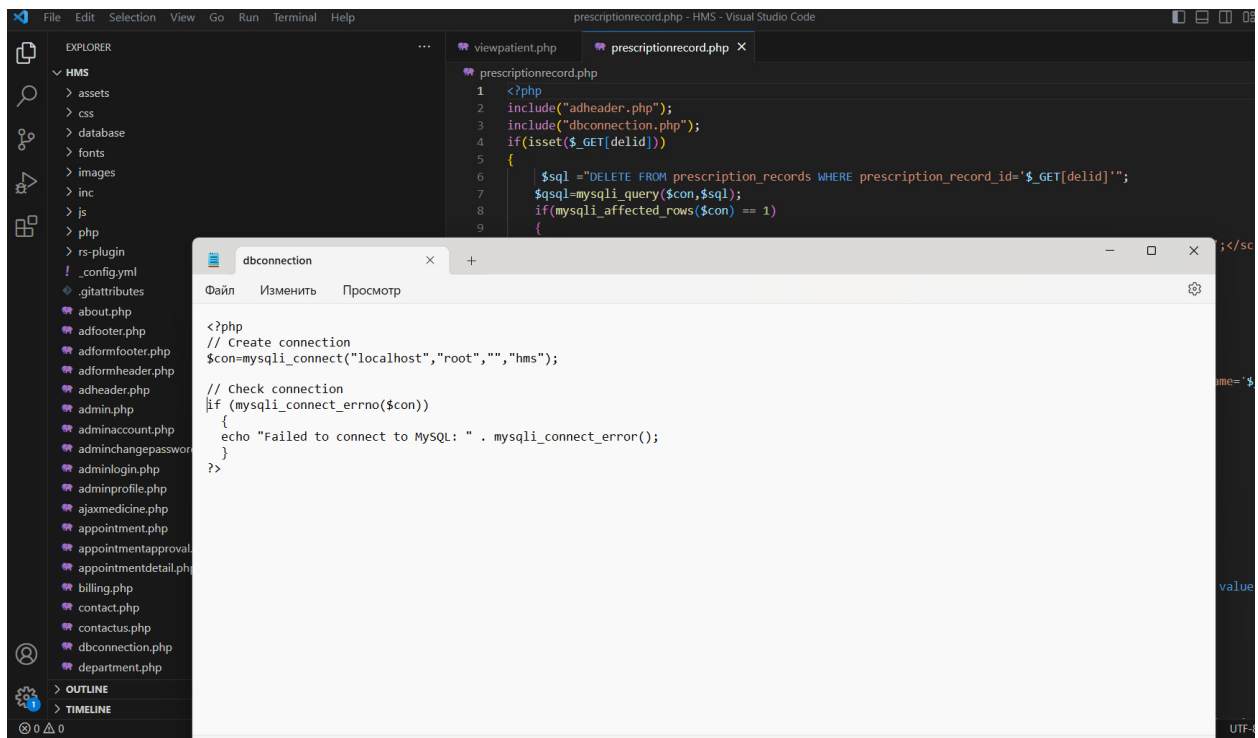


Рисунок 29 – Код инициализации работы с СУБД *MySQL*

После чего требуется повторить запуск приложения из командной строки (рисунок 30, 31).

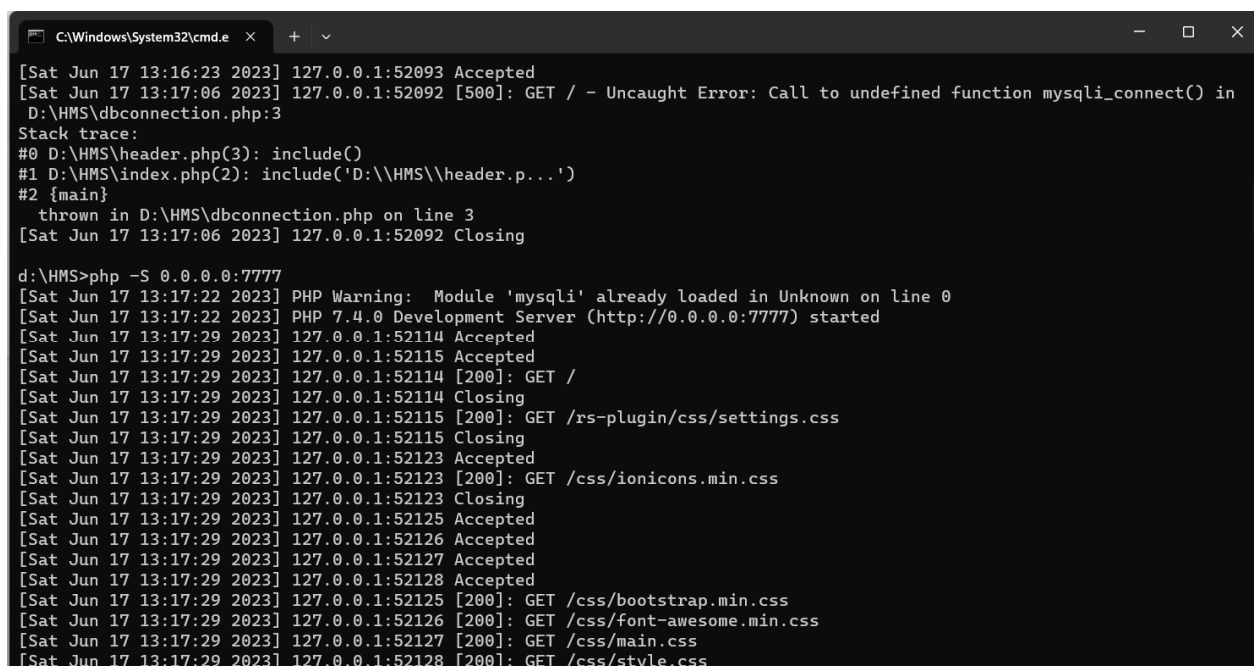


Рисунок 30 – Процесс успешного запуска приложения


```

C:\Windows\System32\cmd.e  x  +  v
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54323 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54325 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54325 [200]: GET /js/vendors/bootstrap.min.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54325 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54326 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54326 [200]: GET /js/vendors/own-menu.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54326 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54329 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54329 [200]: GET /js/vendors/jquery.sticky.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54329 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54330 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54330 [200]: GET /js/vendors/owl.carousel.min.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54330 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54331 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54331 [200]: GET /js/vendors/jquery.magnific-popup.min.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54331 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54332 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54332 [200]: GET /rs-plugin/js/jquery.tp.min.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54332 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54333 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54333 [200]: GET /rs-plugin/js/jquery.tp.min.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54333 Closing
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54334 Accepted
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54334 [200]: GET /js/main.js
[Sat Jun 17 13:19:45 2023] 127.0.0.1:54334 Closing
[Sat Jun 17 13:19:51 2023] 127.0.0.1:54337 Accepted
[Sat Jun 17 13:21:03 2023] 127.0.0.1:54337 Closed without sending a request; it was probably just an unused speculative preconnection
[Sat Jun 17 13:21:03 2023] 127.0.0.1:54337 Closing

```

Рисунок 4 – Приложение успешно запущено

После успешного запуска приложения, войдем по *URL localhost* на его главную страницу и попробуем зарегистрировать новый аккаунт (рисунок 32).

Рисунок 32 – Регистрация нового пользователя

Далее проверим работоспособность режимов перехода из одного пользователя в другой, для чего воспользуемся инструментом выбора роли пользователя (рисунок 33–35).

Вход

Логин

Пароль

Роль ▼

[Регистрация](#)

Рисунок 33 – Вход пациента

Вход

Логин

Пароль

Роль ▼

[Регистрация](#)

Рисунок 34 – Вход доктора

Вход

Логин

Пароль

Роль

Администратор
▼

Войти

Рисунок 35 – Вход администратора

После авторизации как пациент нам открывается доступ к главной странице (рисунок 36), на которой пользователь может просмотреть свои записи, создать напоминание о них или удалить, а также просмотреть динамику здоровья (рисунок 37) и записаться на новый прием.

Пациент: главная страница
Иванов И.И.

Мои записи

Динамика

Записаться на прием

Выйти

Врач	Дата приема	Время приема	Напоминание
Филатов И.И.	23.06.2023	12:00	☐ <div style="margin-left: 10px;"> <div style="background-color: #800000; color: white; padding: 2px 5px; border-radius: 3px;">Отменить</div> </div>
Герасимов А.М.	05.06.2023	11:00	☐ <div style="margin-left: 10px;"> <div style="background-color: #800000; color: white; padding: 2px 5px; border-radius: 3px;">Отменить</div> </div>

Рисунок 36 – Главная страница пользователя «Пациент»

Пациент: динамика здоровья		Иванов И.И.
Медосмотры 2022:		Назад Выйти
Посещения - 6 раз		
Состояние: здоров		
Медосмотры 2023:		
Посещения - 1 раз		
Состояние: здоров		
Напоминание: 01.09.2023		
e-mail: iva.ivanov@mail.ru		

Рисунок 37 – Просмотр динамики здоровья

При записи на прием пациент выбирает специальность врача, самого врача и назначает дату и время записи (рисунок 38).

Пациент: запись на прием к врачу		Иванов И.И.																																										
Запись к врачу		Назад Выйти																																										
Филатов Илья Игоревич: терапевт																																												
Выбор дня приема																																												
<div> <div> <div>« June 2023 »</div> <table border="1"> <tr> <td>Mon</td><td>Tue</td><td>Wed</td><td>Thu</td><td>Fri</td><td>Sat</td><td>Sun</td> </tr> <tr> <td></td><td></td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td> </tr> <tr> <td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td> </tr> <tr> <td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td> </tr> <tr> <td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td> </tr> <tr> <td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td></td><td></td> </tr> </table> </div> <div> <div>12:00</div> <div>12:30</div> <div>13:00</div> <div>13:30</div> <div>14:00</div> <div>14:30</div> <div>15:00</div> <div>15:30</div> <div>16:00</div> <div>16:30</div> </div> </div>	Mon	Tue	Wed	Thu	Fri	Sat	Sun			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31				
Mon	Tue	Wed	Thu	Fri	Sat	Sun																																						
		1	2	3	4	5																																						
6	7	8	9	10	11	12																																						
13	14	15	16	17	18	19																																						
20	21	22	23	24	25	26																																						
27	28	29	30	31																																								

Рисунок 38 – Запись к врачу

2 Охрана труда

В этом разделе излагаются условия, необходимые для обеспечения безопасности и здоровья пользователей ПК, которые работают с программным обеспечением. Кроме того, здесь будут описаны методы снижения вредных факторов, которые могут негативно сказаться на здоровье.

2.1 Санитарно-гигиенические стандарты

Необходимые санитарно-гигиенические стандарты включают в себя такие параметры, как сухость помещения, температура и количество света. Для обеспечения безопасных условий работы, существуют определенные стандарты, которые необходимо соблюдать, а именно:

- каждое рабочее место должно быть в объеме не менее 15 м³;
- высота помещения не менее 3,2 м;
- ширина дверей не менее 0,8 м;
- ширина коридоров не менее 1,4 м.

Окна должны быть расположены таким образом, чтобы максимально обеспечивать естественное освещение рабочих мест. Каждое окно должно открываться и регулярно очищаться от грязи и пыли. В случае недостаточного освещения естественным светом, помещение должно быть оснащено электрическими осветительными приборами. Фонари должны быть изготовлены из армированного стекла и защищены металлическими сетками. Оптимальный диапазон температур в помещении - от 18°C до 21°C, при влажности от 40% до 60%.

Руководители производства должны обеспечить своих работников чистой питьевой водой, и рекомендуется ежедневно менять воду. Помещения, в которых уровень шума превышает норму, должны быть оснащены специальными знаками. Каждый работник должен быть обучен и знать правила безопасности при работе в шумных условиях.

Помещения, где уровень шума превышает установленную норму, должны быть обязательно маркированы специальными знаками, чтобы предупредить работников о возможных рисках и предупредить их принимать необходимые меры предосторожности. Каждый работник, который работает в шумных условиях, должен быть проинструктирован и обязан знать правила безопасности при работе в таких условиях. Он должен быть осведомлен о необходимости использования средств индивидуальной защиты, таких как наушники или слуховые протекторы, чтобы защитить свой слух от повреждения. Работники также должны знать, как правильно использовать эти средства и следить за их состоянием, чтобы эффективно уменьшить воздействие шума на слух.

2.2 Классификация по опасности поражения электрическим током

Условия повышенной опасности поражения людей электрическим током:

- воздух с высокой влажностью (свыше 75%), где образуется конденсат или пар;
- наличие пыли, которая проводит электричество и оседает на проводах, внутри аппаратов и машин, ухудшая изоляцию и охлаждение, однако не является источником опасности для возникновения пожара или взрыва;
- использование металлических, земляных, железобетонных кирпичных оснований;
- высокая температура (в течение всего года более 35°C или кратковременно более 40°C).

Условия особой опасности поражения людей электрическим током:

- наличие сырости (дождь, снег, частое опрыскивание и покрытие влагой потолка, стен и т.д.);
- наличие химически активной среды;

					ПГУ1.090301.14.001 ПЗ	
ИЗ	Лист	№ докум.	Подпись	Дата		52

- наличие одновременно двух или более условий повышенной опасности.

Таким образом можно выделить три категории опасности помещений: безопасные, с повышенной опасностью и особо опасные.

Для минимизации рисков поражения электрическим током рекомендуется работать в помещениях, соответствующих следующим критериям:

- температура воздуха не ниже 30°C, рекомендованный диапазон 18-20 °C;
- полы должны быть выполнены из диэлектрического материала;
- помещение должно проветриваться;
- влажность воздуха не превышает 60%.

2.3 Оборудование рабочего места

Основные предъявляемые требования к рабочему месту:

- расстояние от глаз до экрана должно быть в пределах 600-700 мм;
- монитор необходимо расположить ниже уровня глаз под углом 10-15 градусов к направлению прямого взгляда;
- высота рабочего стола должна регулироваться в пределах 680-800 мм. В случае отсутствия такой возможности, высота стола должна составлять 725 мм;
- клавиатуру следует размещать на поверхности стола на расстоянии 100-300 мм от края, обращенного к пользователю, или на специальной регулируемой по высоте рабочей поверхности;
- рабочий стол должен иметь достаточное пространство для ног, включая высоту не менее 600 мм, глубину на уровне колен не менее 450 мм и на уровне вытянутых ног не менее 650 мм.

Важно, чтобы рабочее место с дисплеем обеспечивало удобство оператора в положении сидя и не создавало перегрузки костно-мышечной системы.

Организовав правильно рабочее место, также необходимо обеспечить правильную посадку пользователя ПК:

- запястья должны быть в прямом положении;
- плечи должны быть расслаблены;
- локти должны полностью лежать на столе, перпендикулярно полу (угол в локтевых суставах составляет 90-100 градусов).

2.4 Освещенность рабочего места

Стандарты освещенности определяются, учитывая характер работы, место и поверхность. Для пультов ЭВМ и дисплеев рекомендуется комбинированное освещение не менее 400 люкс и общее освещение не менее 200 люкс.

На рабочем месте необходимо обеспечивать равномерную яркость, исключая яркие и блестящие предметы. Для разнообразия в поле зрения рекомендуется использовать отдельные яркие поверхности.

В основном для освещения используются потолочные или встроенные светильники с люминесцентными лампами. Чтобы избежать засветки экранов, светильники общего освещения следует устанавливать сбоку от рабочего места, параллельно линии зрения оператора.

2.5 Требования охраны труда в аварийных ситуациях

При возникновении аварийной ситуации на рабочем месте необходимо:

- немедленно прекратить работу;
- сообщить о возникновении аварийной ситуации непосредственному руководителю или лицу, его замещающему;

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		54

- под руководством непосредственного руководителя принять участие в ликвидации создавшейся аварийной ситуации, если это не представляет угрозы для здоровья и жизни работников;

- при необходимости покинуть опасную зону.

При возникновении нарушений в работе персонального компьютера, а также при возникновении нарушений в работе электросети (запах гари, дым, посторонний шум при работе ПК, ощущение действия электрического тока при прикосновении к элементам ПК и т.п.) необходимо:

- прекратить работу;

- отключить ПК от электросети;

- сообщить о случившемся непосредственному руководителю или лицу его замещающему. Не приступать к работе до полного устранения повреждений или неисправностей.

При возникновении возгорания необходимо:

- прекратить работу;

- отключить ПК от электросети;

- сообщить о случившемся непосредственному руководителю или лицу его замещающему;

- приступить к тушению возгорания углекислотным или порошковым огнетушителем;

- при невозможности ликвидировать возгорание, покинуть опасную зону через эвакуационные выходы в соответствии со схемой эвакуации.

При резком ухудшении состояния здоровья или травмировании необходимо прекратить работу, сообщить о случившемся непосредственному руководителю или лицу его замещающему. При необходимости воспользоваться аптечкой.

При несчастном случае во время работы:

- оказать пострадавшему первую помощь в соответствии с инструкцией по оказанию первой (доврачебной) помощи при несчастном

					ПГУ1.090301.14.001 ПЗ	
ИЗ	Лист	№ докум.	Подпись	Дата		55

случае (ИОТ-ООЗ);

- при необходимости доставить пострадавшего в ближайшее медицинское учреждение или вызвать скорую медицинскую помощь;
- немедленно сообщить о случившемся своему непосредственному руководителю или лицу его замещающему;
- принять меры для сохранения обстановки несчастного случая, если это не сопряжено с опасностью для жизни и здоровья людей.

2.6 Контроль микроклимата

В России согласно СанПиН 2021 года работодатели должны соблюдать конкретные параметры микроклимата в рабочих помещениях. Допустимые параметры зависят от времени года и от уровня физических нагрузок сотрудника на рабочем месте.

Из текста СанПиН следует, что в летний период температура воздуха в помещении, в том числе в офисе, должна быть 23–25 °С. При этом температура поверхностей должна варьировать в пределах 22–26 °С. В зимний период в офисе должно быть 22–24 °С, а температура поверхностей около 21–25 °С.

В пределах таких температур работник испытывает «тепловой комфорт». Это означает, что человек, одетый в обычное для помещений количество одежды (например, не в верхней одежде), не испытывает неприятных ощущений. Восприятие температуры воздуха у людей может различаться. Однако даже при незначительных отклонениях от установленного стандарта температур в офисе работник может испытывать дискомфорт, чувство усталости и психоэмоциональное напряжение. Это может негативно повлиять на его производительность и безопасность.

При установленном стандарте температуры в офисе рекомендуется поддерживать относительную влажность воздуха в пределах 40–60%. Такой уровень относительной влажности должен сохраняться независимо от времени года.

					ПГУ1.090301.14.001 ПЗ	
ИЗ	Лист	№ докум.	Подпись	Дата		56

Относительная влажность ниже 20% в офисе может вызвать у сотрудника высыхание кожи, глаз, слизистых оболочек носа и рта. Также низкий уровень относительной влажности может вызвать накопление статического электричества и отрицательно повлиять на работу офисного оборудования, такого как принтеры и компьютеры. Если относительная влажность в офисе выше 70%, это может привести к образованию конденсата на поверхностях и внутри оборудования и строительных конструкций. При длительном воздействии влажности в помещении и в оборудовании может образоваться плесень. Помимо этого, из-за повышенной влажности в офисе становится душно.

Скорость движения воздуха в офисе не должна превышать 0,1 метра в секунду. Скорость воздуха может быть создана системой кондиционирования или вентиляции. Если скорость воздуха превышает установленный стандарт, в помещении могут возникнуть сквозняки. Они могут вызвать у работника дискомфорт.

Административные меры контроля микроклимата:

- обеспечение регулярного обслуживания вентиляции, вытяжки или кондиционирования воздуха;
- предоставление увлажнителей воздуха при высоких температурах в сочетании с низкой относительной влажностью на рабочем месте;
- обеспечение рабочих помещений вентиляторами (например, настольные, напольные или потолочные вентиляторы);
- предоставление переносных обогревательных приборов при холодных условиях на рабочем месте;
- установка жалюзи или светоотражающей плёнки на окна, чтобы контролировать прямой и непрямой солнечный свет;
- при наличии униформы — контроль за соблюдением формального дресс-кода, при этом предоставление возможности работникам самостоятельно адаптировать свою одежду под условия работы;

- предоставление возможности работникам регулировать термостаты или открывать окна или двери по мере необходимости при наличии соответствующего инструктажа по технике безопасности;

- учёт влияния теплового дискомфорта на беременных, людей с заболеваниями или инвалидностью, пожилых или принимающих лекарства работников;

- оценка состояния микроклимата в помещениях специалистами из надзорных организаций, например центров гигиены и эпидемиологии Роспотребнадзора.

Если работодатель не обеспечивает в офисе установленные стандарты температуры, влажности и проветриваемости, безопасность здоровья работников оказывается под угрозой и производительность труда падает.

Заключение

В ходе выполнения дипломного проекта была изучена предметная область, в результате чего было сформулировано техническое задание на разработку приложения для контроля состояния здоровья пользователя.

Был произведен выбор инструментария, выполнено проектирование, в результате которого были разработаны UML диаграммы разрабатываемого ПО.

Произведено проектирование и реализация частей программного обеспечения, в результате которого были разработаны интерфейс, структура базы данных и логика приложения. Приложение реализовано в соответствии с техническим заданием.

В разделе безопасности жизнедеятельности были описаны условия, необходимые для обеспечения безопасности и здоровья пользователей ПК, которые работают с программным обеспечением.

Задание на дипломное проектирование выполнено в полном объеме.

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		59

Список используемых источников

1. Шлемов М.В. «Профессиональное программирование на PHP». - Москва: Дом Пресс, 2018. - 511 с.
2. Багров Б.Ф. «Язык MySQL. Краткий справочник». Москва, Первый Бит, 2019. - 367 с.
3. Макаров С.А. «Реализация WEB приложений». Москва, Бином, 2021. - 431 с.
4. Колисниченко Д.Н. «PHP и MySQL. Разработка веб-приложений» (6-е издание), 2017. – 640 с.
5. Веллинг, Томсон. «Разработка веб-приложений с помощью PHP и MySQL», 2017. – 768 с.

Приложение А

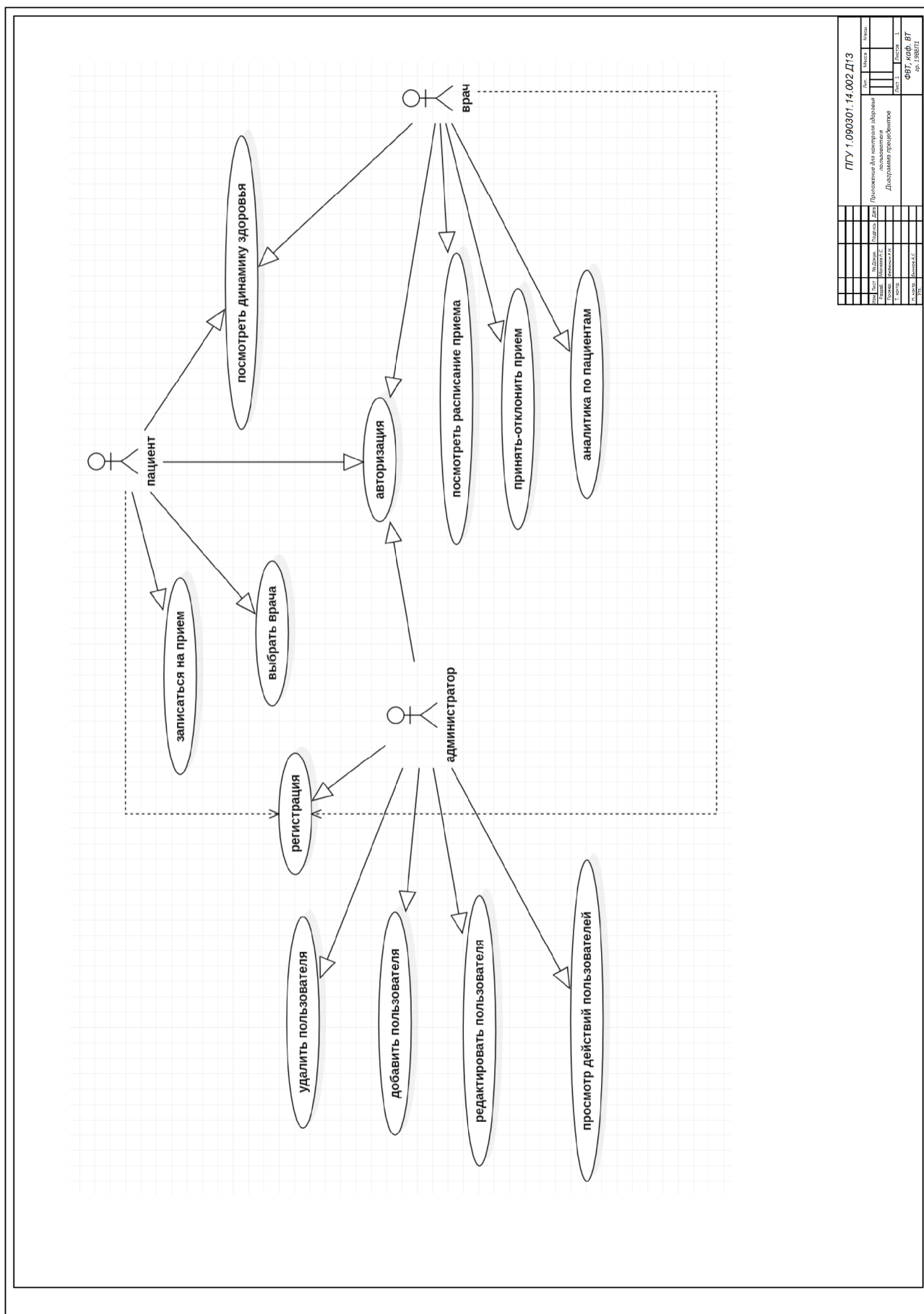
(обязательное)

Приложение для контроля состояния здоровья пользователя

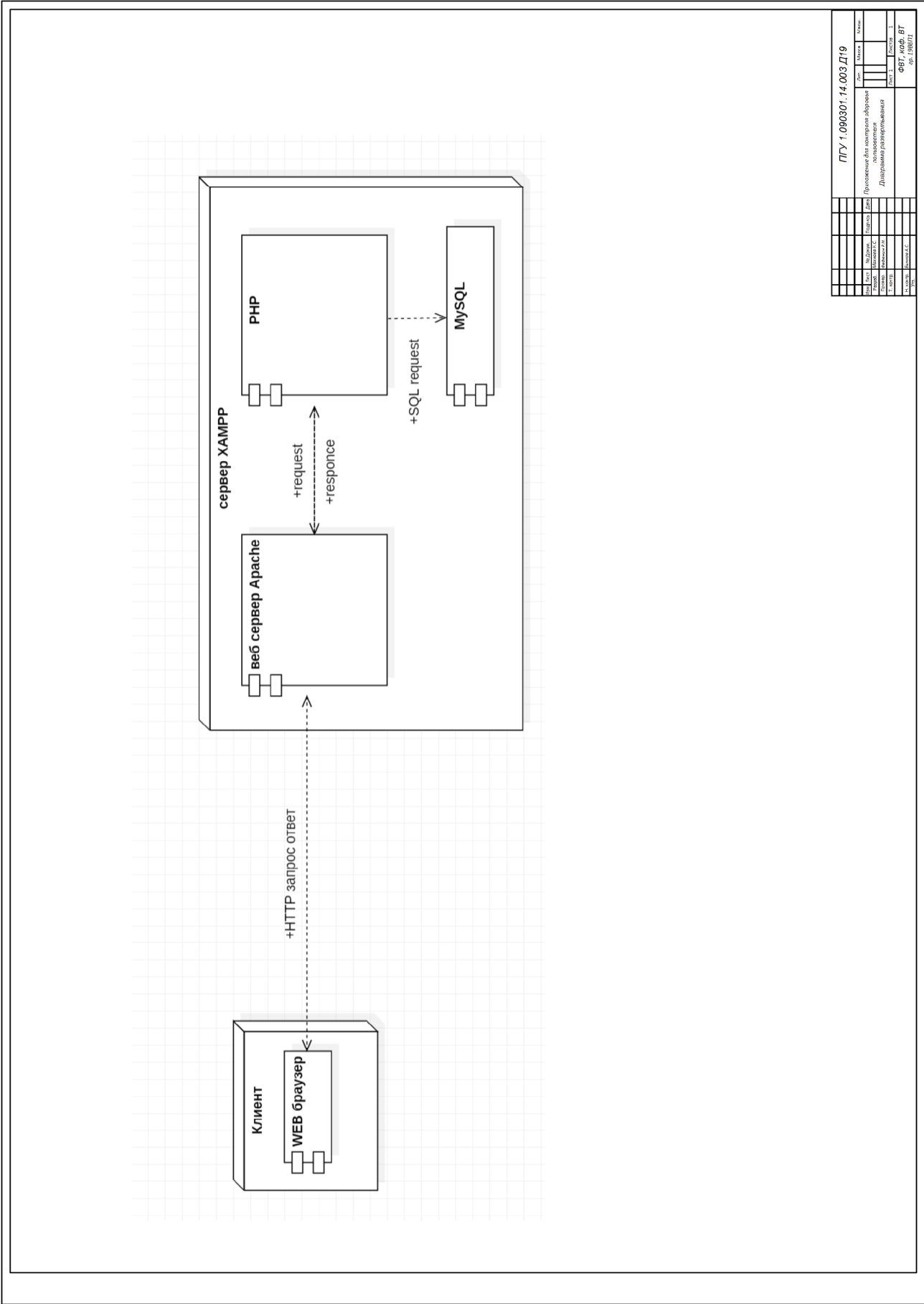
UML-диаграммы

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		61

Приложение А.1 – Диаграмма прецедентов



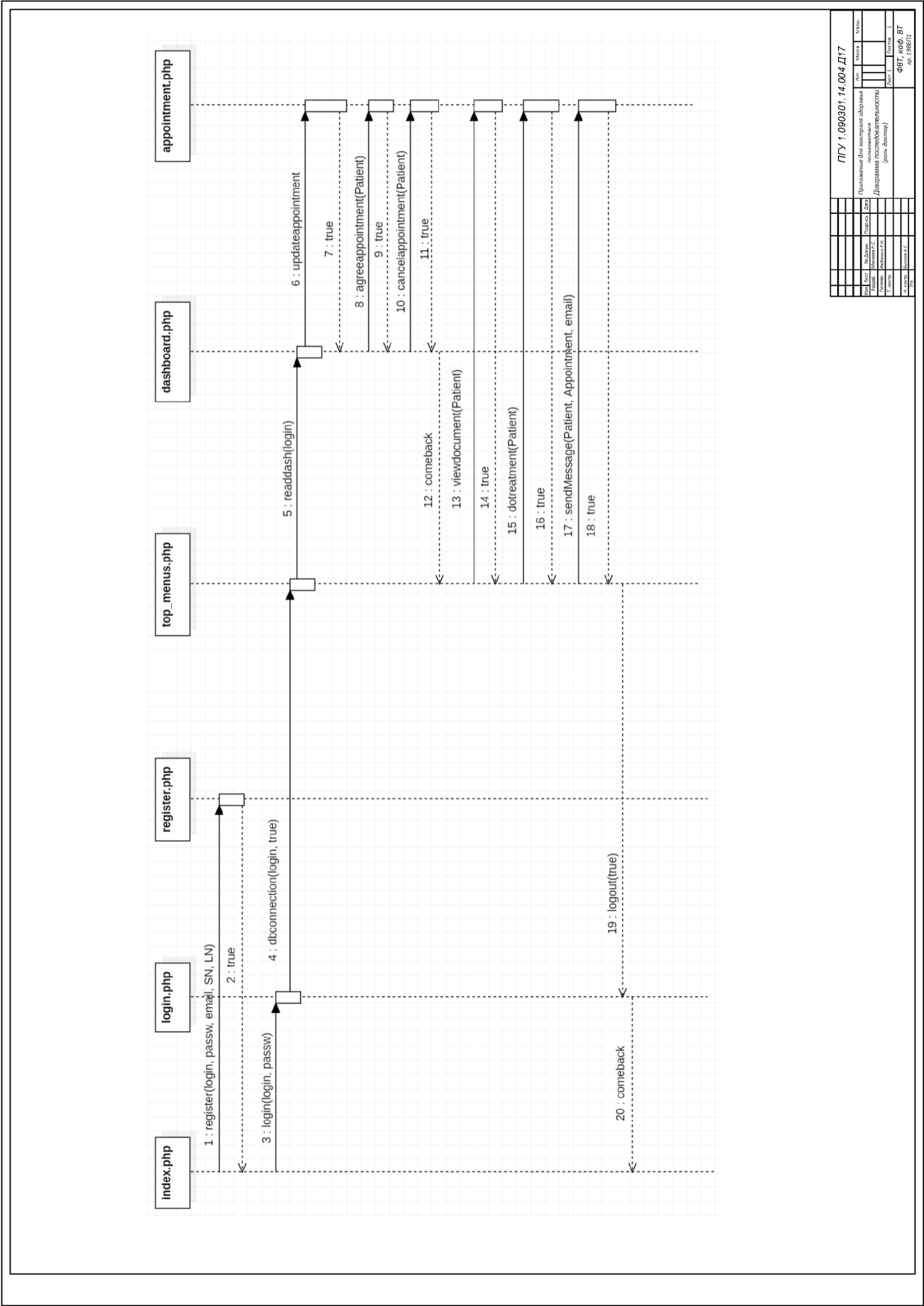
Приложение А.2 – Диаграмма развертывания



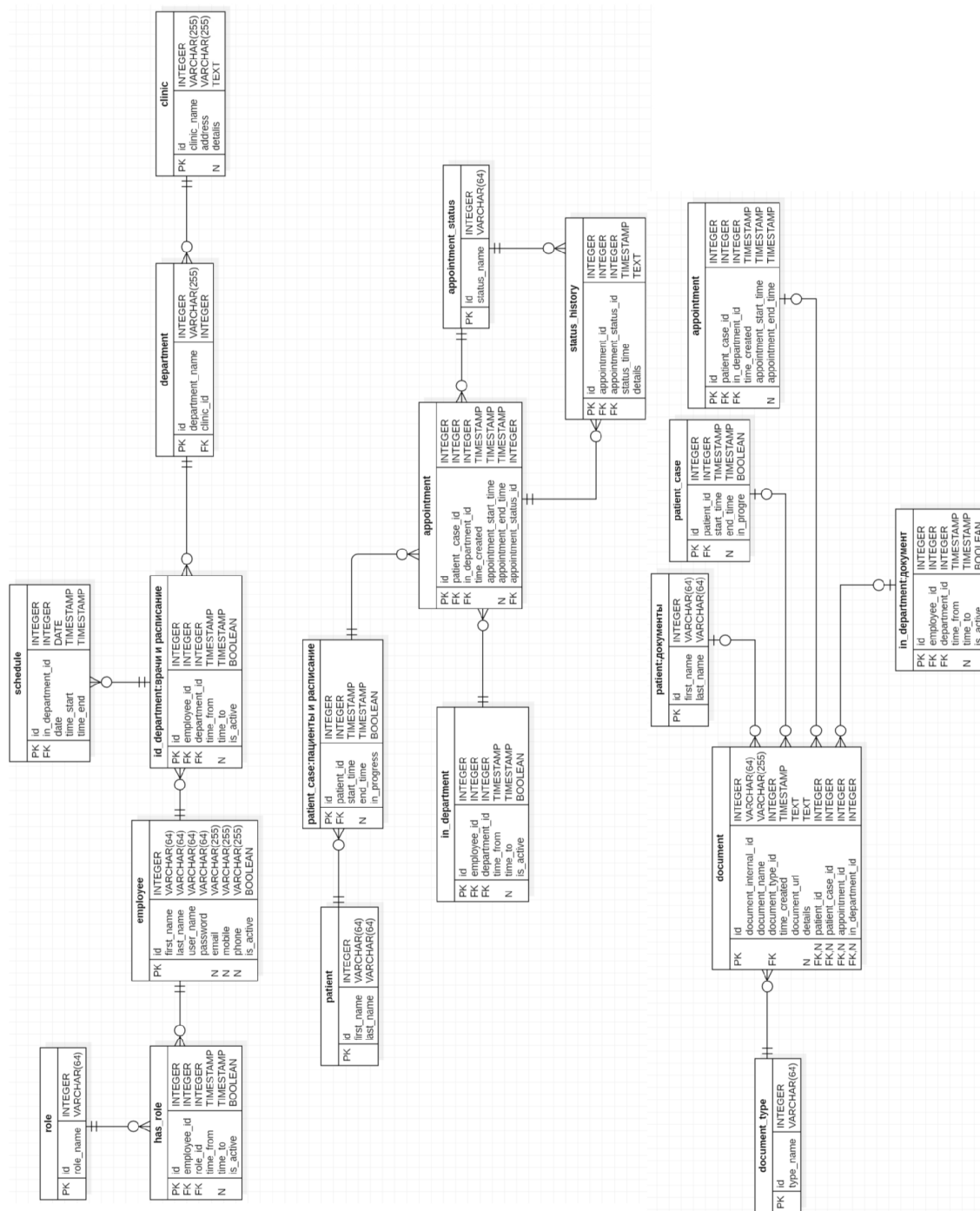
ПГУ 1.090301.14.003 Д19			
Изм.	№ докум.	Дата	Кто
1	1	1	1
Приложение для авторского оформления			
Диаграмма развертывания			
ФЭТ 1008 ВТ			
Фр. 1908/11			

Изм.	Лист	№ докум.	Подпись	Дата

Приложение А.3 – Диаграмма последовательности



ПГУ 1.090301.14.004 Д17									
Приложение для контроля достоверности получаемых результатов									
Диаграмма последовательности (разрешено)									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									
ФЭП, код, ВТ									

[illegible]

Приложение Б

(обязательное)

Приложение для контроля состояния здоровья пользователя

Листинг программы

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		66

Файл Index.php

```
<?php
include("adheader.php");
include("dbconnection.php");
if(isset($_POST[submit]))
{
    if(isset($_GET[editid]))
    {
        $sql = "UPDATE doctor SET
doctorname='$_POST[doctorname]',mobilenumber='$_POST[mobilenumber]',de
partmentid='$_POST[select3]',loginid='$_POST[loginid]',password='$_
_POST[password]',status='$_POST[select]',education='$_POST[educati
on]',experience='$_POST[experience]',consultancy_charge='$_POST[co
nsultancy_charge]' WHERE doctorid='$_GET[editid]'";
        if($qsql = mysqli_query($con,$sql))
        {
            echo "<script>alert('doctor record updated
successfully...');</script>";
        }
        else
        {
            echo mysqli_error($con);
        }
    }
    else
    {
        $sql = "INSERT INTO
doctor(doctorname,mobilenumber,departmentid,loginid,password,status,ed
ucation,experience,consultancy_charge)
values('$_POST[doctorname]', '$_POST[mobilenumber]', '$_POST[select3
]', '$_POST[loginid]', '$_POST[password]', 'Active', '$_POST[education
]', '$_POST[experience]', '$_POST[consultancy_charge]')";
        if($qsql = mysqli_query($con,$sql))
        {
            echo "<script>alert('Doctor record inserted
successfully...');</script>";
        }
        else
        {
            echo mysqli_error($con);
        }
    }
}
if(isset($_GET[editid]))
{
    $sql="SELECT * FROM doctor WHERE doctorid='$_GET[editid]'
";
    $qsql = mysqli_query($con,$sql);
    $rsedit = mysqli_fetch_array($qsql);
```

```

}
?>

<div class="container-fluid">
    <div class="block-header">
        <h2> Add New Doctor </h2>
    </div>
    <div class="row clearfix">
        <div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
            <div class="card">

                <form method="post" action=""
name="frmdoctr" onSubmit="return validateform()" style="padding:
10px">

                    <div class="form-
group"><label>Doctor Name</label>
                        <div class="form-line">
                            <input class="form-control"
type="text" name="doctorname" id="doctorname" value="<?php echo
$rsedit[doctorname]; ?>" />
                        </div>
                    </div>

                    <div class="form-
group"><label>Mobile Number</label>
                        <div class="form-line">
                            <input class="form-control"
type="text" name="mobilenumber" id="mobilenumber" value="<?php
echo $rsedit[mobilenumber]; ?>" />
                        </div>
                    </div>

                    <div class="form-
group"><label>Department</label>
                        <div class="form-line">
                            <select name="select3"
id="select3" class="form-control show-tick">
                                <option
value="">Select</option>
                                <?php
                                $sqldepartment= "SELECT *
FROM department WHERE status='Active'";
                                $qsqldepartment =
mysqli_query($con,$sqldepartment);

                                while($rsdepartment=mysqli_fetch_array($qsqldepartment)) {

```

```

        if($rsdepartment[departmentid] == $rsedit[departmentid])
        {
            echo "<option
value='$rsdepartment[departmentid] '
selected>$rsdepartment[departmentname]</option>";
        }
        else
        {
            echo "<option
value='$rsdepartment[departmentid] '>$rsdepartment[departmentname]<
/option>";
        }
    }
    ?>
</select>
</div>
</div>

<div class="form-
group"><label>Login ID</label>
<div class="form-line">
<input class="form-control"
type="text" name="loginid" id="loginid" value="<?php echo
$rsedit[loginid]; ?>"/>
</div>
</div>

<div class="form-
group"><label>Password</label>
<div class="form-line">
<input class="form-control"
type="password" name="password" id="password" value="<?php echo
$rsedit[password]; ?>"/>
</div>
</div>

<div class="form-
group"><label>Confirm Password</label>
<div class="form-line">
<input class="form-control"
type="password" name="cnfirmpassword" id="cnfirmpassword"
value="<?php echo $rsedit[password]; ?>"/>
</div>
</div>

<div class="form-
group"><label>Education</label>
<div class="form-line">

```



```

        <input class="form-control"
type="text" name="education" id="education" value="<?php echo
$rsedit[education]; ?>" />
    </div>
</div>

    <div class="form-
group"><label>Experience</label>
        <div class="form-line">
            <input class="form-control"
type="text" name="experience" id="experience" value="<?php echo
$rsedit[experience]; ?>" />
        </div>
    </div>

    <div class="form-
group"><label>Consultancy Charge</label>
        <div class="form-line">
            <input class="form-control"
type="text" name="consultancy_charge" id="consultancy_charge"
value="<?php echo $rsedit[experience]; ?>" />
        </div>
    </div>

    <div class="form-group">
        <label>Status</label>
        <div class="form-line">
            <select class="form-control show-
tick" name="select" id="select">
                <option value=""
selected="" hidden>Select</option>
                <?php
                    $arr=
                    array("Active","Inactive");
                    foreach($arr as $val)
                    {
                        if($val ==
$rsedit[status])
                        {
                            echo "<option
value='$val' selected>$val</option>";
                        }
                        else
                        {
                            echo "<option
value='$val'>$val</option>";
                        }
                    }
                ?>
            </select>
        </div>
    </div>

```

```

        </div>
    </div>

    <input class="btn btn-default"
type="submit" name="submit" id="submit" value="Submit" />

    </form>
</div>
</div>
</div>
</div>
<?php
include("adfooter.php");
?>
<script type="application/javascript">
var alphaExp = /^[a-zA-Z]+$/; //Variable to validate only
alphabets
var alphaspaceExp = /^[a-zA-Z\s]+$/; //Variable to validate only
alphabets and space
var numericExpression = /^[0-9]+$/; //Variable to validate only
numbers
var alphanumericExp = /^[0-9a-zA-Z]+$/; //Variable to validate
numbers and alphabets
var emailExp = /^[\\w\\-\\.\\+]+\\@[a-zA-Z0-9\\.\\-]+\\. [a-zA-z0-
9]{2,4}$/; //Variable to validate Email ID

function validateform()
{
    if(document.frmdoct.doctorname.value == "")
    {
        alert("Doctor name should not be empty..");
        document.frmdoct.doctorname.focus();
        return false;
    }
    else
if(!document.frmdoct.doctorname.value.match(alphaspaceExp))
    {
        alert("Doctor name not valid..");
        document.frmdoct.doctorname.focus();
        return false;
    }
    else if(document.frmdoct.mobilenumber.value == "")
    {
        alert("Mobile number should not be empty..");
        document.frmdoct.mobilenumber.focus();
        return false;
    }
    else
if(!document.frmdoct.mobilenumber.value.match(numericExpression))

```

```

{
    alert("Mobile number not valid..");
    document.frmdoct.mobilenumber.focus();
    return false;
}
else if(document.frmdoct.select3.value == "")
{
    alert("Department ID should not be empty..");
    document.frmdoct.select3.focus();
    return false;
}
else if(document.frmdoct.loginid.value == "")
{
    alert("loginid should not be empty..");
    document.frmdoct.loginid.focus();
    return false;
}
else
if(!document.frmdoct.loginid.value.match(alphanumericExp))
{
    alert("loginid not valid..");
    document.frmdoct.loginid.focus();
    return false;
}
else if(document.frmdoct.password.value == "")
{
    alert("Password should not be empty..");
    document.frmdoct.password.focus();
    return false;
}
else if(document.frmdoct.password.value.length < 8)
{
    alert("Password length should be more than 8
characters...");
    document.frmdoct.password.focus();
    return false;
}
else if(document.frmdoct.password.value !=
document.frmdoct.cnfirmpassword.value )
{
    alert("Password and confirm password should be
equal..");
    document.frmdoct.password.focus();
    return false;
}
else if(document.frmdoct.education.value == "")
{
    alert("Education should not be empty..");
    document.frmdoct.education.focus();
    return false;
}
else if(!document.frmdoct.education.value.match(alphaExp))

```

```

        {
            alert("Education not valid..");
            document.frmdoct.education.focus();
            return false;
        }
        else if(document.frmdoct.experience.value == "")
        {
            alert("Experience should not be empty..");
            document.frmdoct.experience.focus();
            return false;
        }
        else
        if(!document.frmdoct.experience.value.match(numericExpression))
        {
            alert("Experience not valid..");
            document.frmdoct.experience.focus();
            return false;
        }
        else if(document.frmdoct.select.value == "" )
        {
            alert("Kindly select the status..");
            document.frmdoct.select.focus();
            return false;
        }
        else
        {
            return true;
        }
    }
</script>

```

Файл Patient.php

```

<?php
include("adformheader.php");
include("dbconnection.php");
if(isset($_POST[submit]))
{
    if(isset($_GET[editid]))
    {
        $sql ="UPDATE patient SET
patientname='$_POST[patientname]',admissiondate='$_POST[admissiond
ate]',admissiontime='$_POST[admissiontme]',address='$_POST[address
]',mobilenumber='$_POST[mobileno]',city='$_POST[city]',pincode='$_
POST[pincode]',loginid='$_POST[loginid]',password='$_POST[password
]',bloodgroup='$_POST[select2]',gender='$_POST[select3]',dob='$_PO
ST[dateofbirth]',status='$_POST[select]' WHERE
patientid='$_GET[editid]';
        if($qsql = mysqli_query($con,$sql))
        {
            echo "<script>alert('patient record updated
successfully...');</script>";

```

```

    }
    else
    {
        echo mysqli_error($con);
    }
}
else
{
    $sql = "INSERT INTO
patient(patientname,admissiondate,admissiontime,address,mobileno,c
ity,pincode,loginid,password,bloodgroup,gender,dob,status)
values('$ _POST[patientname]','$dt','$tim','$ _POST[address]','$ _POS
T[mobilenummer]','$ _POST[city]','$ _POST[pincode]','$ _POST[loginid]
','$ _POST[password]','$ _POST[select2]','$ _POST[select3]','$ _POST[d
ateofbirth]','$Active')";
    if($qsql = mysqli_query($con,$sql))
    {
        echo "<script>alert('patients record inserted
successfully...');</script>";
        $insid= mysqli_insert_id($con);
        if(isset($_SESSION[adminid]))
        {
            echo
"<script>window.location='appointment.php?patid=$insid';</script>"
;
        }
        else
        {
            echo
"<script>window.location='patientlogin.php';</script>";
        }
    }
    else
    {
        echo mysqli_error($con);
    }
}
}
if(isset($_GET[editid]))
{
    $sql="SELECT * FROM patient WHERE patientid='".$_GET[editid]'"
";
    $qsql = mysqli_query($con,$sql);
    $rsedit = mysqli_fetch_array($qsql);
}
?>
<div class="container-fluid">
    <div class="block-header">
        <h2>Patient Registration Panel</h2>
    </div>
    <div class="card">
        <form method="post" action="" name="frmpatient"

```

```

onSubmit="return validateform()" style="padding: 10px">
    <div class="form-group"><label>Patient Name</label>
        <div class="form-line">
            <input class="form-control" type="text"
name="patientname" id="patientname"
                value="<?php echo $rsedit[patientname];
?>" />
        </div>
    </div>

    <?php
        if(isset($_GET[editid]))
        {
            ?>
            <div class="form-group"><label>Admission Date</label>
                <div class="form-line">
                    <input class="form-control" type="date"
name="admissiondate" id="admissiondate"
                        value="<?php echo $rsedit[admissiondate];
?>" readonly />
                </div>
            </div>
            <div class="form-group"><label>Admission Time</label>
                <div class="form-line">
                    <input class="form-control" type="time"
name="admissiontme" id="admissiontme"
                        value="<?php echo $rsedit[admissiontime];
?>" readonly />
                </div>
            </div>
            <?php
                }
            ?>
            <div class="form-group">
                <label>Address</label>
                <div class="form-line">
                    <input class="form-control " name="address"
id="tinymce" value="<?php echo $rsedit[address]; ?>">
                </div>
            </div>
            <div class="form-group"><label>Mobile Number</label>
                <div class="form-line">
                    <input class="form-control" type="text"
name="mobilenumber" id="mobilenumber"
                        value="<?php echo $rsedit[mobilenumber]; ?>"
/>
                </div>
            </div>
            <div class="form-group"><label>City</label>
                <div class="form-line">
                    <input class="form-control" type="text"
name="city" id="city"
                        value="<?php echo $rsedit[city]; ?>" />
                </div>
            </div>

```

```

        </div>
    </div>
    <div class="form-group"><label>PIN Code</label>
        <div class="form-line">
            <input class="form-control" type="text"
name="pincode" id="pincode"
                value="<?php echo $rsedit[pincode]; ?>" />
        </div>
    </div>
    <div class="form-group"><label>Login ID</label>
        <div class="form-line">
            <input class="form-control" type="text"
name="loginid" id="loginid"
                value="<?php echo $rsedit[loginid]; ?>" />
        </div>
    </div>

    <div class="form-group"><label>Password</label>
        <div class="form-line">
            <input class="form-control" type="password"
name="password" id="password"
                value="<?php echo $rsedit[password]; ?>"
/>
        </div>
    </div>
    <div class="form-group"><label>Confirm
Password</label>
        <div class="form-line">
            <input class="form-control" type="password"
name="confirmpassword" id="confirmpassword"
                value="<?php echo
$rsedit[confirmpassword]; ?>" />
        </div>
    </div>
    <div class="form-group"><label>Blood Group</label>
        <div class="form-line"><select class="form-control
show-tick" name="select2" id="select2">
            <option value="">Select</option>
            <?php
                $arr = array("A+", "A-", "B+", "B-
", "O+", "O-", "AB+", "AB-");
                foreach($arr as $val)
                {
                    if($val ==
$rsedit[bloodgroup])
                    {
                        echo "<option
value='$val' selected>$val</option>";
                    }
                    else
                    {
                        echo "<option

```

```

value='$val'>$val</option>";
                                }
                                }
                                ?>
                                </select>
                                </div>
                                </div>
                                <div class="form-group"><label>Gender</label>
                                <div class="form-line"><select class="form-control
show-tick" name="select3" id="select3">
                                <option value="">Select</option>
                                <?php
                                $arr = array("MALE","FEMALE");
                                foreach($arr as $val)
                                {
                                    if($val == $rsedit[gender])
                                    {
                                        echo "<option value='$val'
selected>$val</option>";
                                    }
                                    else
                                    {
                                        echo "<option
value='$val'>$val</option>";
                                    }
                                }
                                ?>
                                </select>
                                </div>
                                </div>
                                <div class="form-group"><label>Date Of Birth</label>
                                <div class="form-line">
                                <input class="form-control" type="date"
name="dateofbirth" max="<?php echo date("Y-m-d"); ?>"
                                id="dateofbirth" value="<?php echo
$rsedit[dob]; ?>" />
                                </div>
                                </div>
                                <input class="btn btn-default" type="submit"
name="submit" id="submit" value="Submit" />
                                </form>
                                <p>&nbsp;</p>
                                </div>
                                </div>
                                </div>
                                <div class="clear"></div>
                                </div>
                                </div>
                                <?php
                                include("adformfooter.php");
                                ?>
                                <script type="application/javascript">

```



```

var alphaExp = /^[a-zA-Z]+$/; //Variable to validate only
alphabets
var alphaspaceExp = /^[a-zA-Z\s]+$/; //Variable to validate only
alphabets and space
var numericExpression = /^[0-9]+$/; //Variable to validate only
numbers
var alphanumericExp = /^[0-9a-zA-Z]+$/; //Variable to validate
numbers and alphabets
var emailExp = /^[\\w\\-\\.\\+]+@[a-zA-Z0-9\\.\\-]+\\. [a-zA-z0-
9]{2,4}$/; //Variable to validate Email ID

function validateform() {
    if (document.frmpatient.patientname.value == "") {
        alert("Patient name should not be empty..");
        document.frmpatient.patientname.focus();
        return false;
    } else if
(!document.frmpatient.patientname.value.match(alphaspaceExp)) {
        alert("Patient name not valid..");
        document.frmpatient.patientname.focus();
        return false;
    } else if (document.frmpatient.admissiondate.value == "") {
        alert("Admission date should not be empty..");
        document.frmpatient.admissiondate.focus();
        return false;
    } else if (document.frmpatient.admissiontme.value == "") {
        alert("Admission time should not be empty..");
        document.frmpatient.admissiontme.focus();
        return false;
    } else if (document.frmpatient.address.value == "") {
        alert("Address should not be empty..");
        document.frmpatient.address.focus();
        return false;
    } else if (document.frmpatient.mobilenumber.value == "") {
        alert("Mobile number should not be empty..");
        document.frmpatient.mobilenumber.focus();
        return false;
    } else if
(!document.frmpatient.mobilenumber.value.match(numericExpression))
{
        alert("Mobile number not valid..");
        document.frmpatient.mobilenumber.focus();
        return false;
    } else if (document.frmpatient.city.value == "") {
        alert("City should not be empty..");
        document.frmpatient.city.focus();
        return false;
    } else if
(!document.frmpatient.city.value.match(alphaspaceExp)) {
        alert("City not valid..");
        document.frmpatient.city.focus();
        return false;
    }
}

```

```

    } else if (document.frmpatient.pincod.value == "") {
        alert("Pincode should not be empty..");
        document.frmpatient.pincod.focus();
        return false;
    } else if
(!document.frmpatient.pincod.value.match(numericExpression)) {
        alert("Pincode not valid..");
        document.frmpatient.pincod.focus();
        return false;
    } else if (document.frmpatient.loginid.value == "") {
        alert("Login ID should not be empty..");
        document.frmpatient.loginid.focus();
        return false;
    } else if
(!document.frmpatient.loginid.value.match(alphanumericExp)) {

        alert("Login ID not valid..");
        document.frmpatient.loginid.focus();
        return false;
    } else if (document.frmpatient.password.value == "") {
        alert("Password should not be empty..");
        document.frmpatient.password.focus();
        return false;
    } else if (document.frmpatient.password.value.length < 8) {
        alert("Password length should be more than 8
characters...");
        document.frmpatient.password.focus();
        return false;
    } else if (document.frmpatient.password.value !=
document.frmpatient.confirmpassword.value) {
        alert("Password and confirm password should be equal..");
        document.frmpatient.confirmpassword.focus();
        return false;
    } else if (document.frmpatient.select2.value == "") {
        alert("Blood Group should not be empty..");
        document.frmpatient.select2.focus();
        return false;
    } else if (document.frmpatient.select3.value == "") {
        alert("Gender should not be empty..");
        document.frmpatient.select3.focus();
        return false;
    } else if (document.frmpatient.dateofbirth.value == "") {
        alert("Date Of Birth should not be empty..");
        document.frmpatient.dateofbirth.focus();
        return false;
    } else if (document.frmpatient.select.value == "") {
        alert("Kindly select the status..");
        document.frmpatient.select.focus();
        return false;
    } else {
        return true;
    } } </script>

```

Приложение В

(обязательное)

Приложение для контроля состояния здоровья пользователя

Презентация

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		80

ПОЛОЖЕНИЕ ДЛЯ КОНТРОЛЯ **СОСТОЯНИЯ ЗДОРОВЬЯ** **ПОЛЬЗОВАТЕЛЯ**

работы:

студент гр. 19ВВП1 Махнаев Никита Сергеев
ый руководитель:

С.Т.Н., доцент Федюнин Роман Николаевич

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		81

аль работы и объект

разработать приложение для контроля за здоровьем наличия у приложения функций записи ведением пациента как пользователя

а является клиент-серверное приложение, и
ции:

иента иметь возможность записаться к врачу
ча иметь возможность принять пациента и
эм;

инистратора иметь возможность управлять

Реализованные задачи

Разработать приложение с доступом в виде тонкого клиента с функциями:

- авторизация пользователей;
- регистрация пользователей ;
- управление пользователями всех категорий ;
- управление процессом записи и ведения медицинской помощи и профилактики
- управление выбором требуемого врача.

Сравнение аналогов

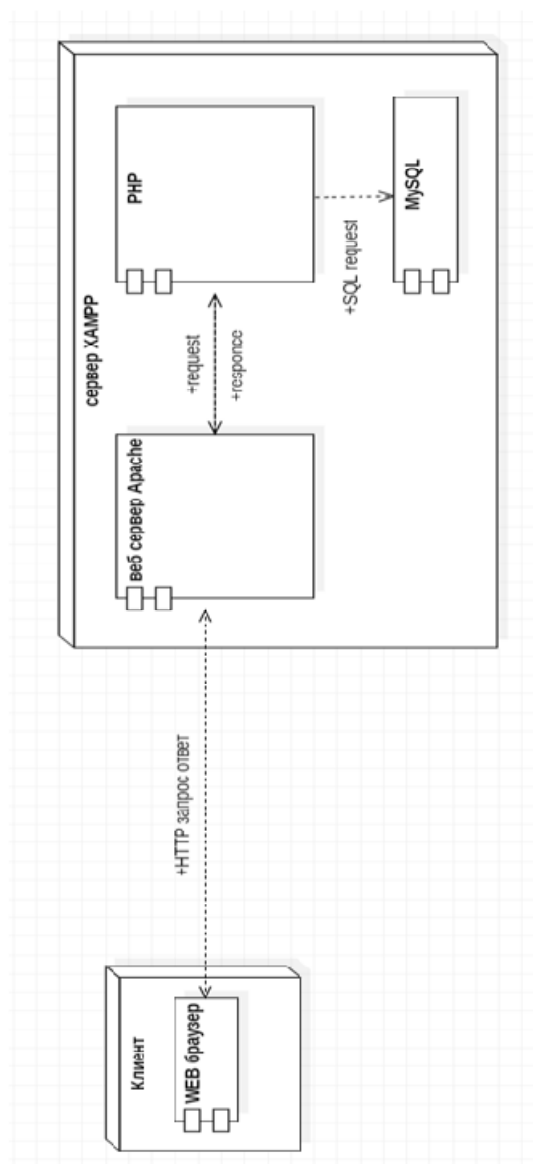
Изготовитель	Тип реализации	Интерфейс	Контроль здоровья пациента	Стоимость
Medioment	Облачный сервис с мощной справочной системой для слежения за здоровьем пациента	Громоздкий интерфейс	Автоматический, точный	От \$1000 в год
Medis	SaaS платформа	Интуитивный интерфейс	Автоматический, точный	От \$1000
Med365	Анализ профиля ссылок	Интуитивный интерфейс	Полуавтоматический, точность проверяется вручную	От \$1000
Medчет	Модульная программа	Простой интерфейс	Полуавтоматический, точность проверяется вручную	От \$1000

Используемые технологии

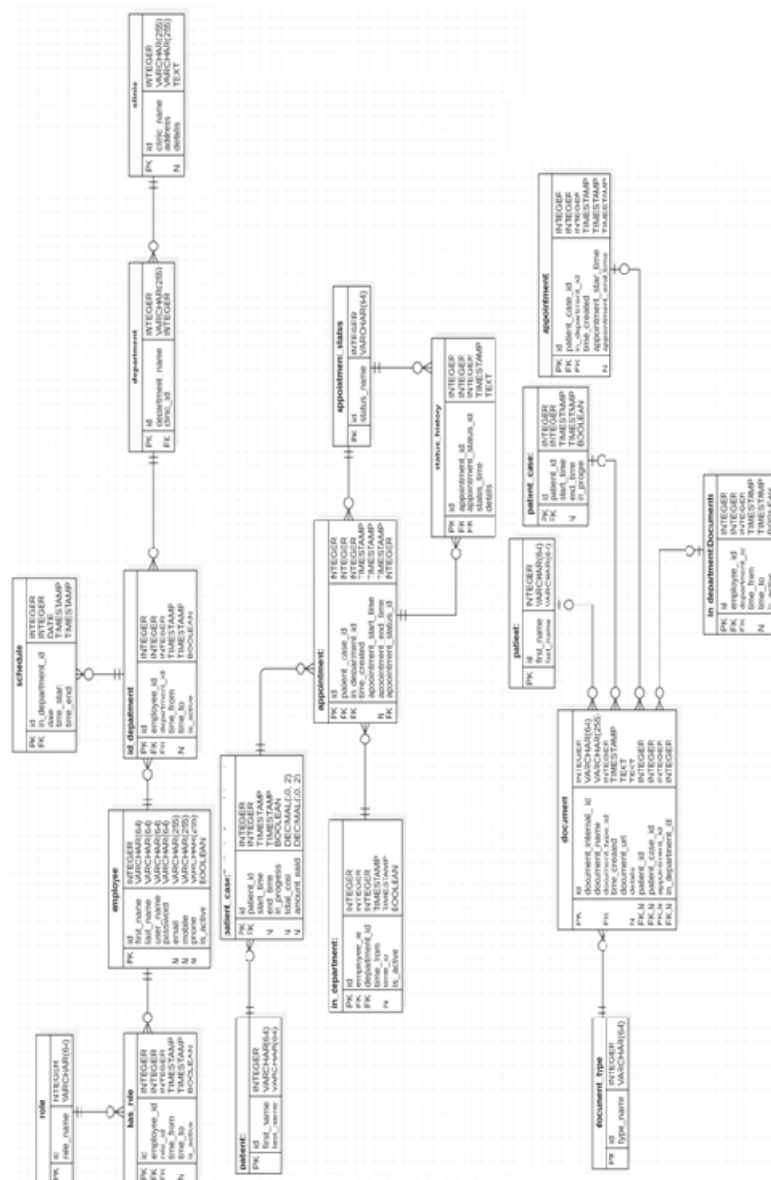


Из	Лист	№ докум.	Подпись	Дата

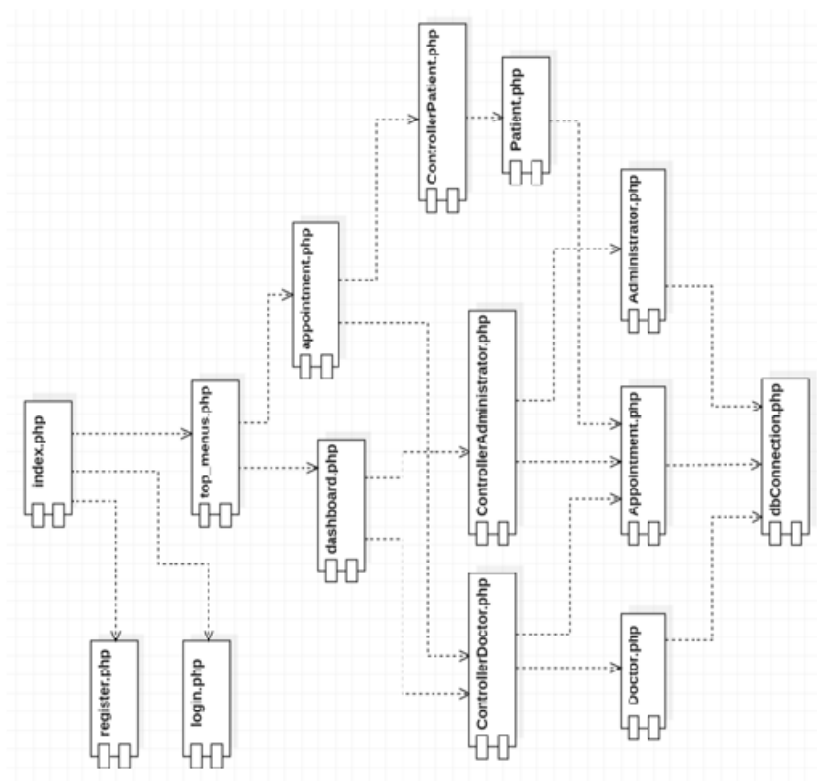
Архитектура приложения



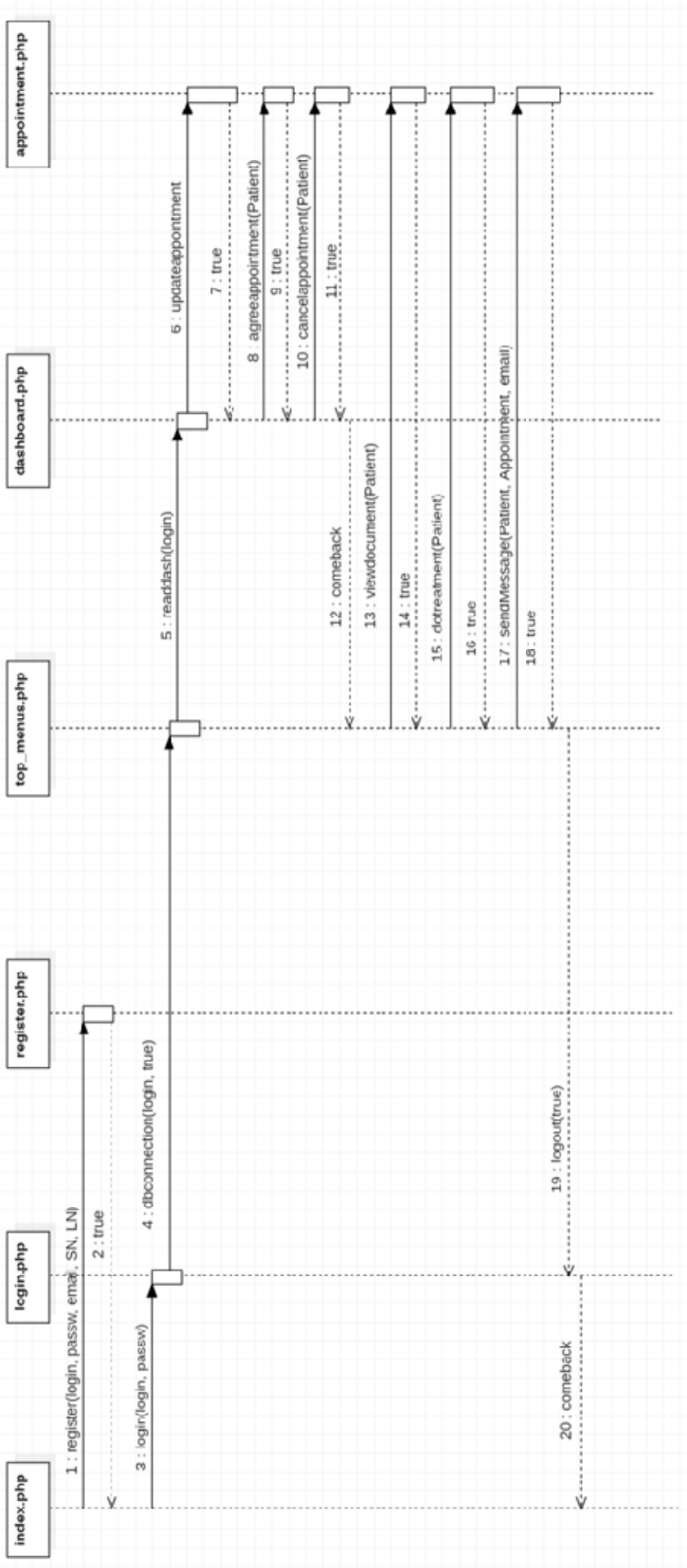
Структура базы данных



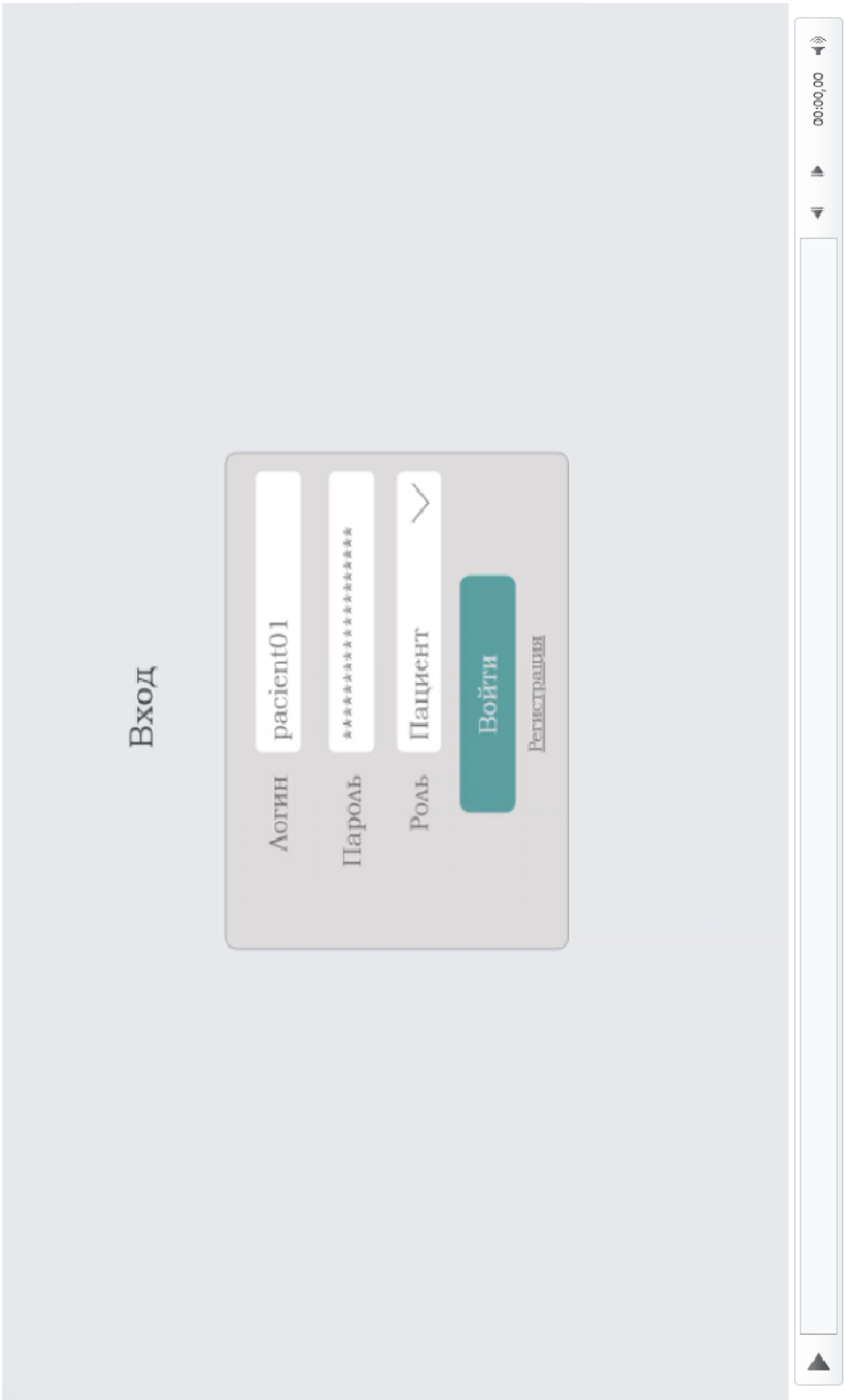
UML – диаграмма компонентов



UML – диаграмма последовательности



Обзор приложения



Результаты работы

- Разработано приложение полностью соответствующее техническому заданию;
- Проведено функциональное тестирование (успешно);
- Намечены задачи по развитию функциональных возможностей программы.

Спасибо за внимание!

					ПГУ1.090301.14.001 ПЗ	
Из	Лист	№ докум.	Подпись	Дата		92