

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

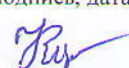
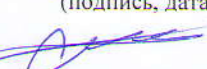
Специальность: 09.05.01 «Применение и эксплуатация
автоматизированных систем специального
назначения»

Специализация: «Эксплуатация вычислительных машин, комплексов,
систем и сетей специального назначения»

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
СПЕЦИАЛИСТА**

на тему

**Веб-приложение по персональному поиску
требуемых медицинских препаратов**

Студент	 (подпись, дата)	Борунов Вячеслав Игоревич (ФИО полностью)
Руководитель	 (подпись, дата)	Гурин Е.И. (фамилия, инициалы)
Нормоконтролёр	 (подпись, дата)	Кучин А.В. (фамилия, инициалы)
Рецензент	<i>Доцент Пенз ГТУ</i> (должность, место работы)	 (подпись, дата)
		Печерский С.В. (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 10.06.2020 г. № 13)

Заведующий кафедрой	 (подпись)	М.А. Митрохин (фамилия, инициалы)
---------------------	---	--------------------------------------

Работа защищена с отметкой отлично (протокол заседания ГЭК от
18.06.2020 № 3)

Секретарь ГЭК	 (подпись)	Е.И. Гурин
---------------	--	------------

Пенза 2020

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

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

«Утверждаю»

Заведующий кафедрой

 М.А. Митрохин

«30» апреля 2020г.

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
СПЕЦИАЛИСТА**

1. Студент Борунов В.И. гр. 15BO2 факультета ВТ

Борунов Вячеслав Игоревич

2. Тема работы Веб-приложение по персональному поиску
требуемых медицинских
препаратов

Тема утверждена приказом ПГУ №0335а/оот "30" апреля 2020г.

3. Руководитель работы Гурин Евгений Иванович

4. Задание на работу (назначение разработки, исходные данные и т.п.)
Целью выпускной квалификационной работы специалиста является разработка
программного обеспечения системы по персональному поиску требуемых
медицинских препаратов. Разрабатываемая система должна быть рассчитана на
работу разных типов пользователей. Система должна обеспечивать поиск
требуемых лекарственных препаратов несколькими способами, выводить
полную информацию о каждом препарате и обеспечивать создание, изменение
и удаление содержимого базы
лекарств;

5. Перечень подлежащих разработке вопросов Разработка базы данных онлайн сервиса, разработка логической модели базы данных, разработка физической модели базы данных, разработка программного обеспечения, разработка и анализ пользовательского интерфейса, разработка программного обеспечения клиентской части, разработка программного обеспечения серверной части.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Выбор темы и разработка индивидуального плана проведения работ	4%	11.02.20 - 12.02.20	
2	Подбор и изучение материала по теме работы	46%	12.02.20 - 17.04.20	
3	Получение и анализ результатов	37%	17.04.20 - 25.05.20	
4	Оформление отчета	13%	02.06.20	
	Общий объем	100%		

Дата выдачи задания "30" апреля 2020 г.

Руководитель работы
(подпись, дата)



Задание к исполнению принял студент
(подпись, дата)



Работу к защите допустить
Декан факультета
(подпись, дата)



РЕФЕРАТ

Пояснительная записка содержит 134 листа, 19 рисунков, 17 таблиц, 21 источник, 5 приложений.

ВЕБ-САЙТ, ООП, PHP, JS, CSS, UML, LARAVEL, MYSQL, ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ.

Объектом исследования является веб-приложение для поиска требуемых медицинских препаратов.

Цель работы - разработать веб-приложение для более простого и эффективного поиска требуемых медицинских препаратов.

В процессе работы проведен анализ предметной области, дано обоснование целесообразности разработки информационной системы, разработана объектно-ориентированная модель и структура приложения, написан и отлажен программный код.

В результате проделанной работы разработано и отлажено веб-приложение для более эффективного поиска лекарственных препаратов.

Работа находится на стадии тестирования с целью возможного дальнейшего внедрения в работу УНЦ КМЦ ПГУ.

Эффективность веб-приложения заключается в существенном повышении скорости нахождения нужной информации. Веб-приложения может применяться больницами, аптеками, и другими медицинскими учреждениями, для упрощения получения информации о лекарственных препаратах.

					ПГУ 1.090501.02.001 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Веб-приложение по персональному поиску требуемых медицинских препаратов Пояснительная записка	Лит.	Лист	Листов
Разраб.		Борунов В.И.					2	134
Провер.		Гурин Е.И.						
Реценз.		Печерский С.В.						
Н. Контр.		Кучин А.В.						
Утв.								
						Кафедра ВТ Гр.15ВО2		

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1 Обзор предметной области.....	7
1.1 Обзор известных методов решения задач	7
1.2 Постановка задачи на разработку приложения.....	11
2 Реализация веб-приложения в соответствии с требованиями.....	13
2.1 Описание веб-приложения и основных выполняемых функций	13
2.2 Описание технологии для создания веб-приложения	14
2.3 Описание функциональных требований.....	15
2.4 Построение исходной концептуальной модели базы данных	22
2.5 Описание требований к обобщённой архитектуре приложения	24
2.6 Выбор технологии и средств разработки	25
2.6.1 Выбор системы управления базами данных	26
2.6.2 Выбор фреймворков для разработки программного интерфейса ...	27
2.6.3 Средства визуализации.....	28
3 Разработка программного обеспечения веб-приложения для поиска лекарственных препаратов	30
3.1 Планирование бюджета проекта.....	30
3.2 Разработка архитектуры программного обеспечения	32
3.3 Разработка классов модели приложения	34
3.4 Разработка методов взаимодействия модели с базой данных.....	35
3.5 Разработка пользовательского интерфейса	41
4 Тестирование и анализ качества программного обеспечения.....	47
4.1 Объект испытаний.....	47
4.2 Цель испытания	47
4.3 Требования к программе.....	47
4.4 Средства и порядок испытаний.....	48
4.5 Методы испытаний.....	48
ЗАКЛЮЧЕНИЕ	57
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	58
ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА ПРОЕКТ	61
ЛИСТИНГ ИСХОДНОГО КОДА.....	64

ДИАГРАММЫ	101
ГРАФИЧЕСКИЕ РЕЗУЛЬТАТЫ ТЕСТОВЫХ ИСПЫТАНИЙ.....	106
ВНЕДРЕНИЕ	124
ПРЕЗЕНТАЦИЯ	126

ВВЕДЕНИЕ

Ни для кого не секрет, что для любого человека всегда самым главным в жизни является здоровье. Таким образом, мы подходим к идее веб-приложения по персональному поиску требуемых медицинских препаратов, места, где собрана информация обо всех известных лекарственных препаратах на сегодняшний день, все инструкции, внешний вид, ценовая политика, симптоматика и всё это в одном приложении. Общая информация о разработке приложений и теоретических основ [1-3].

Приложение по персональному поиску требуемых медицинских препаратов помогает избавиться от траты огромного количества времени, которое уходит: на запись к врачу, отсиживание больших очередей, ради получения консультации и списка требуемых для лечения препаратов. Однако у стандартной процедуры посещения врача всё таки остается несколько преимуществ, в отличие от веб-конкурента, врач, может выписать рецепт на лекарства которые без рецепта не купишь в аптеке, так же врач может точно определить заболевание, степень осложнения и прогнозировать его развитие. В будущем этот момент можно будет решить, если курировать данное приложение будет специализированный врач-консультант, с которым можно будет связаться, узнать дополнительную информацию, но рецепты в любом случае будут выписываться при личном осмотре.

Целью данной выпускной квалификационной работы (ВКР) является создание веб-приложения по персональному поиску требуемых медицинских препаратов, главной задачей которого является помощь в нахождении нужного лекарства прямо из дома, без траты времени на посещение врача. Для достижения поставленной цели следует провести

анализ предметной области и сформулировать задачи на разработку данного веб-приложения.

					ПГУ 1.090501.02.001 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

1 Обзор предметной области

1.1 Обзор известных методов решения задач

Рынок веб-приложений для поиска лекарств довольно обширный. В настоящее время насчитывается большое количество мобильных приложений различной сложности платной и бесплатной версии, которые отличаются своим функционалом.

Для анализа были выбраны следующие разработки, представленные в таблице 1.

Таблица 1 – Анализируемые приложения

Название	Разработчик	Цена
Артека.RU	АО НПК Катрен	Бесплатно
Яндекс Здоровье	Яндекс	Бесплатно, но есть платные услуги
Карманный доктор	Eugen Shaposhnikov	69 рублей

Для сравнения были выбраны следующие критерии: дизайн, доступные платформы, простота в использовании, информативность, стабильность работы.

Приложение Arтека.RU – проект компании «Катрен», ведущего фармацевтического дистрибьютора России. Отличное бесплатное приложение для Android и IOS, которое обладает довольно красивым дизайном (рисунок 1). В программе доступно большое количество встроенного в него функционала, среди которого: поиск лекарств по симптомам, поиск ближайших аптек в твоём городе, заказ и покупка лекарств, сортировка лекарств, создание профиля и т.д. Главные недостатки: приложение работает нестабильно, интерфейс слишком нагружен, функция заказа товара не проработана в полной мере.



Рисунок 1 – Приложение Apteka.RU

Приложение Яндекс Здоровье представлено на рисунке 2. В отличие от Apteka.RU в нём присутствуют следующие немало важные функции: консультация с врачом, разделение для детей и взрослых, но есть и отрицательные моменты. Самый главный недостаток приложения в том, что в нём очень много рекламы, которая нагружает интерфейс и мешает быстро ориентироваться на сайте. Также из недостатков стоит выделить отсутствие в приложении многих картинок с лекарствами, что не позволяет быстро и точно найти требуемый препарат. В приложении есть функция онлайн консультирования с врачом, но она является платной.



Яндекс Здоровье Врач онлайн Статьи Энциклопедия **Лекарства** Поиск Все сервисы

Лекарства > Противомикробные препараты для системного применения > Препараты, активные в отношении микобактерий Источник: vida.ru

Препараты, активные в отношении микобактерий

ОТПУСКАЕТСЯ ПО РЕЦЕПТУ	ОТПУСКАЕТСЯ ПО РЕЦЕПТУ	ОТПУСКАЕТСЯ ПО РЕЦЕПТУ
 <p>Изониазид таблетки 300 мг: 50 или 100 шт.</p>	 <p>Микобутин капсулы 150 мг: 30 шт.</p>	 <p>Протионамид таблетки, покрытые оболочкой, 250 мг: 50, 10...</p>
 <p>Рифакомб таблетки, покрытые оболочкой, 150 мг+100...</p>	 <p>Пиразинамид-акри® таблетки 500 мг: 100 шт.</p>	 <p>Фтизоэтам® таблетки 150 мг+400 мг: 100 шт.</p>

Реклама



Зарабатывай больше на фрилансе

В 4 раза больше заказов с PRO аккаунтом. Работодатель заметит тебя первым!

Я фрилансер
Я заказчик
Получи первое задание

[Узнать больше](#) 

Рисунок 2 – Приложение Яндекс Здоровье

Приложение Карманный доктор несколько не уступает по дизайну предыдущим приложениям (рисунок 3). Приложение отличается удобным и понятным интерфейсом, а индикатор, показывающий последовательность действий в виде сигналов светофора, очень помогает в принятии решений. Главное преимущество данного приложения – его встроенные инструкции для лечения заболеваний. Карманный доктор присутствует как на платформе IOS, так и Android. Но и данный продукт не обходится без недостатков – приложение является платным, так же в нём отсутствует информация о лекарствах.



Рисунок 3 – Приложение Карманный доктор

На основе анализа продуктов по определённым критериям была составлена следующая таблица 2.

Таблица 2 – Результаты анализа аналогичных разработок

Продукты	Дизайн	Платформы	Простота в использовании	Информативность	Стабильность работы
Аpteka.RU	8/10	Android, iOS	4/5	Отличная	Не стабильно
Яндекс Здоровье	7/10	Android, iOS	4/5	Удовлетворительная	Хорошая
Карманный доктор	9/10	Android, iOS	5/5	Не удовлетворительно	Хорошая

Рассмотренные выше программы – это лишь малая часть рынка приложений по поиску информации о медицинских препаратах. Исходя из проведённого анализа аналогичных разработок был сделан вывод, о том что в разрабатываемом веб-приложении должны обязательно присутствовать следующие функции:

- поиск лекарств по названию;

- поиск лекарств по симптомам;
- вывод полной информации о всех препаратах.

Благодаря описанным выше функциям, приложение будет актуально среди пользователей и составит конкуренцию на рынке приложений по поиску лекарств. Помимо функций, которые описаны выше, приложение должно содержать и базовые.

Исходя из проведённого анализа аналогичных разработок, была поставлена следующая задача на разработку приложения.

1.2 Постановка задачи на разработку приложения

В жизненном цикле программного продукта постановка задачи играет главную роль, если целью разработки является полноценно функционирующий продукт. Максимально подробно и четко проработанные задачи позволят получить стоящие программы.

Требуется разработать веб-приложение по персональному поиску требуемых медицинских препаратов, которое будет соответствовать определённым характеристикам. Проанализировав аналогичные разработки, были определены требования к интерфейсу:

- 1) интерфейс должен быть простым и дружелюбным;
- 2) все препараты должны иметь графическое представление;
- 3) интерфейс не должен быть захламлён рекламой.

Были определены требования к функционалу приложения:

- 1) приложение должно быть кроссплатформенным;
- 2) поиск лекарства по названию самого лекарства;
- 3) поиск лекарства по названию заболевания;
- 4) поиск лекарства по симптомам;
- 5) должен быть организован вход для администратора;
- 6) панель администратора должна позволять редактировать, удалять

и создавать новые записи.

Предложенный выше функционал есть у всех аналогичных разработок и, чтобы составить им конкуренцию на рынке, требуется реализовать все описанные функции. Кроме базовых функций должны быть реализованы уникальные функции:

- 1) вывод актуальных цен на все препараты;
- 2) вывод наиболее дешёвого аналога препарата;
- 3) сортировка препаратов по заданным критериям.

В данном разделе были выделены особенности приложения с точки зрения интерфейса и функциональности.

2 Реализация веб-приложения в соответствии с требованиями

2.1 Описание веб-приложения и основных выполняемых функций

Разрабатываемое веб-приложение должно помогать пользователю максимально быстро и просто находить нужное лекарство, не выходя из дома. Работа веб-приложения будет основана на таких функциях как: поиск в базе по названию лекарства, поиск лекарства по симптоматике, вывод лекарства по выбору заболевания, вывод полной информации о лекарстве. Интерфейс веб-приложения должен быть прост и удобен в использовании, для возможности использования приложения людей старшей возрастной группы.

Помимо вспомогательных функций, веб-приложение должно иметь следующие немало важные функции: добавление, редактирование, удаление препаратов, заболеваний и симптомов.

При добавлении нового препарата фиксируются: название лекарства, описание, инструкция, цена и фотография.

При добавлении нового заболевания фиксируются: название болезни, описание.

При добавлении нового симптома фиксируются: название симптома.

Помимо выше перечисленных задач, в веб-приложении должна присутствовать система, которая будет помогать пользователю найти наиболее бюджетный лекарственный препарат.

Веб-приложение разрабатывается с целью экономии времени человека, а также для предоставления требуемой информации о лекарствах, ведь инструкция может потеряться, а без неё принимать лекарства не зная дозировки и способа приёма опасно для здоровья. Зачастую многие сталкиваются с такой проблемой, что болит голова, а в аптечке столько таблеток, что ты уже и не помнишь, для чего нужна та или иная пилюля.

Данное приложение разрабатывается как некий помощник человека, который всегда находится под рукой. Так как в нашу эпоху высоких технологий общество не может обойтись без интернета, было решено с его помощью помочь человеку, ведь люди сами вольно-невольно периодически взаимодействуют с ним.

2.2 Описание технологии для создания веб-приложения

Выбор языка разработки приложения является очень важным этапом, от него зависит состав команды разработчиков, бюджет и условия поддержки проекта. Для того чтобы выбрать оптимальный для наших задач язык программирования, нам нужно рассмотреть все плюсы и минусы двух наиболее популярных языков PHP и Java.

- Тяжеловесность. Во всех отношениях. Практически любой проект на Java будет иметь более длинный код, чем его аналог на PHP;
- Java лучше справляется с проектами, для которых характерна высокая серверная нагрузка;
- В отличие от PHP, проекты на Java сперва должны пройти компиляцию. То есть, изменения на ходу попросту невозможны. В PHP же код можно редактировать напрямую, без промежуточных этапов;
- Стоимость разработки. Всё зависит от конкретного проекта, но в целом можно обозначить тенденцию — проекты на Java получаются заметно дороже, чем на PHP;
- В отличие от PHP, Java куда лучше справляется с большими нагрузками на сервер, несмотря на то, что имеет более высокие минимальные требования к “железу”;
- Универсальность. Java приложения могут одинаково работать на любом устройстве, которое поддерживает Java Virtual Machine. То есть, не только на смартфонах и планшетах, но и, например, в автомобилях, медиа-

плеерах, бытовой технике и т.п. Следует отметить, что PHP также себя замечательно чувствует при кроссплатформенной разработке;

- PHP — крайне гибкий язык. Практически любую задачу на PHP можно решить десятками разных способов;

- PHP обладает обширным сообществом программистов, поэтому в сети можно найти миллионы различных готовых заготовок кода для различных нужд [4].

По итогу, можно сделать вывод, что для разработки нашего приложения больше подходит язык PHP. На языке PHP код программы будет короче, а значит и стоимость разработки и обслуживания будет меньше. Также язык PHP является кроссплатформенным, поэтому ни каких сложностей при будущей разработке мобильного приложения не возникнет. Для подобного проекта будет легче найти обслуживающий персонал, так как программистов на языке PHP больше, а большое количество заготовок кода и гибкость языка, позволят нам крайне быстро обновлять функционал приложения.

2.3 Описание функциональных требований

Описание функциональных требований заключается в создании диаграммы вариантов использования, на которой отображается взаимодействие между вариантами использований, представляющими функции системы, и действующими лицами, представляющими людей или системы, получающие или передающие информацию в данную систему [5]. Диаграмма вариантов использования была разработана на основе пункта 1.3 данной выпускной квалификационной работы (рисунок 4).

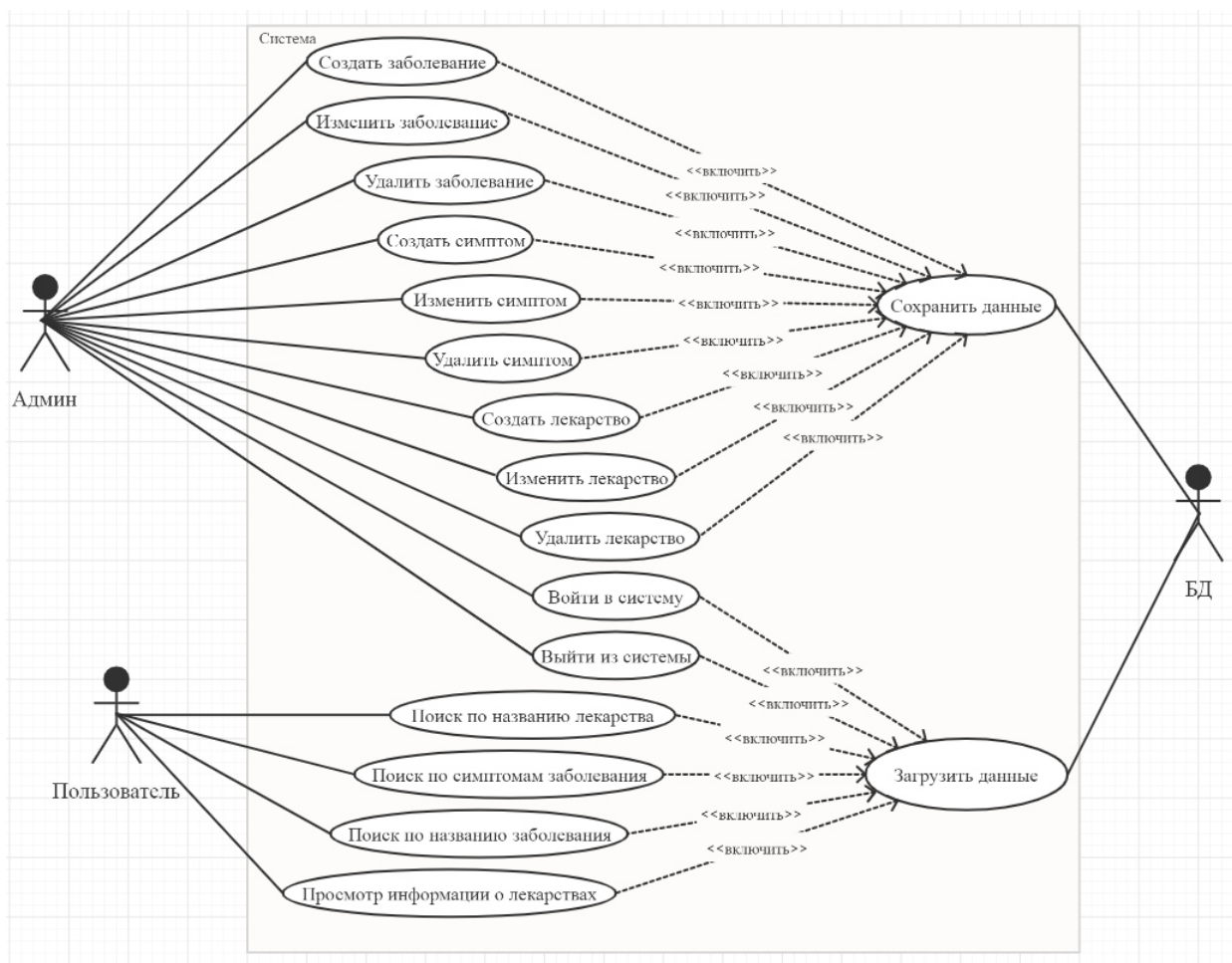


Рисунок 4 – Диаграмма вариантов использования разрабатываемого веб-приложения

Из представленной выше диаграммы можно выделить 3 актёра: «Админ», «Пользователь», «База данных (БД)». Актёр «Админ» включает следующие возможности использования программы:

- «Создать заболевание»;
- «Изменить заболевание»;
- «Удалить заболевание»;
- «Создать симптом»;
- «Изменить симптом»;
- «Удалить симптом»;
- «Создать лекарство»;
- «Изменить лекарство»;

- «Удалить лекарство»;
- «Войти в систему»;
- «Выйти из системы».

Актёр «Пользователь» включает следующие возможности использования программы:

- «Поиск по названию лекарства»;
- «Поиск по симптомам заболевания»;
- «Поиск по названию заболевания»;
- «Просмотр информации о лекарствах»;

Основные действия актёра «БД» – сохранение и загрузка данных в базу данных.

Для подробного рассмотрения предполагаемого поведения системы приведем спецификацию некоторых прецедентов, которые представлены на таблицах 3 – 5.

Таблица 3 – Сценарий выполнения прецедента «Войти в систему»

Прецедент: Войти в систему
ID: 1
Краткое описание: Админ авторизируется в системе
Главные действующие лица: Админ, БД
Предусловия: Введены e-mail и пароль
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. Админ нажимает на кнопку «Войти в систему»; 2. Система формирует запрос к серверу БД. «Include» Загрузить данные; 3. БД загружает данные с сервера; 4. Выполняется проверка; 5. Если админ прошёл проверку; <ol style="list-style-type: none"> 5.1. Система выдаёт сообщение об успешной авторизации; 5.2. Система перенаправляет пользователя на страницу профиля. 6. Иначе; <ol style="list-style-type: none"> 6.1. Система выдаёт ошибку.
Постусловия: Нет
Альтернативные потоки: Нет

Таблица 4 – Сценарий выполнения прецедента «Добавить лекарство»

Прецедент: Добавить лекарство
ID: 2
Краткое описание: Админ добавляет лекарство в БД
Главные действующие лица: Админ, БД
Предусловия: Заполнены данные формы
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. Админ нажимает на кнопку «Добавить лекарство»; 2. Система формирует запрос к серверу БД. «Include» Сохранить данные; 3. БД сохраняет данные с формы, которую отправил админ; 4. Выполняется проверка; 5. Если проверка на поля прошла успешно; <ol style="list-style-type: none"> 5.1. Система сохраняет форму; 6. Иначе; <ol style="list-style-type: none"> 6.1. Система выдаёт ошибку.
Постусловия: Нет
Альтернативные потоки: Нет

Таблица 5 – Сценарий выполнения прецедента «Поиск лекарства»

Прецедент: Поиск лекарства
ID: 3
Краткое описание: Пользователь производит поиск лекарства в БД
Главные действующие лица: Пользователь, БД
Предусловия: Заполнена БД
<p>Основной поток:</p> <ol style="list-style-type: none"> 1. Пользователь нажимает на кнопку «Поиск по названию лекарства»; 2. Система формирует запрос к серверу БД. «Include» Загрузить данные; 3. Приложение формирует; 4. Выполняется проверка; 5. Если проверка на поля прошла успешна; <ol style="list-style-type: none"> 5.1. Система сохраняет форму; 6. Иначе; <ol style="list-style-type: none"> 6.1. Система выдаёт ошибку.
Постусловия: Нет
Альтернативные потоки: Нет

Последний рассмотренный прецедент является одним из важнейших вариантов использования. Он отвечает за основную функцию приложения, поиск лекарственных препаратов тремя способами. Для наглядности была построена диаграмма деятельности данного прецедента (рисунок 5).

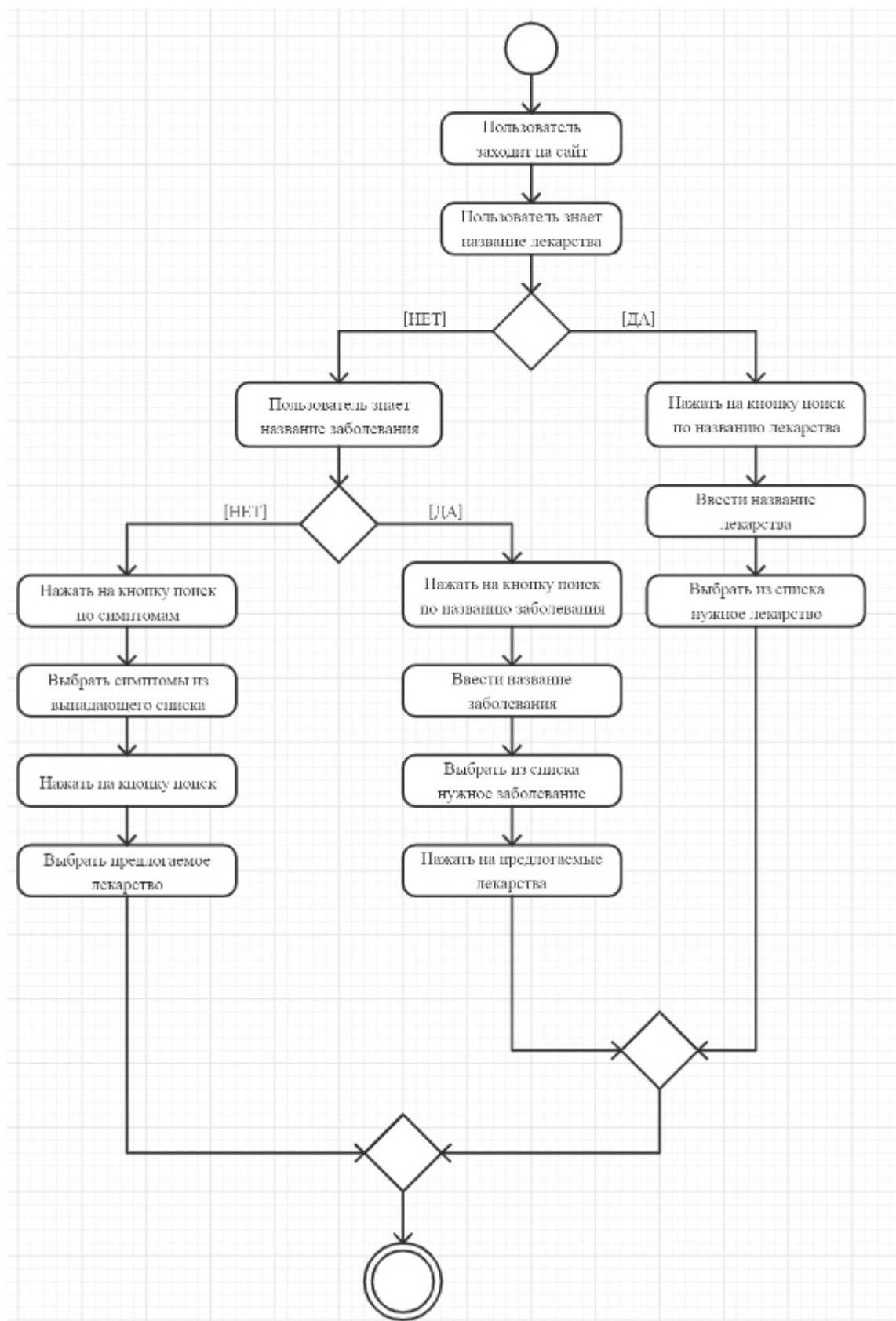


Рисунок 5 – Диаграмма деятельности для варианта использования «Поиск лекарства»

В данном разделе была построена диаграмма вариантов использования, диаграмма деятельности и были приведены спецификации некоторых требований. Также был составлен список требований на разработку программного продукта.

2.4 Построение исходной концептуальной модели базы данных

Концептуальная модель (англ. conceptual model) – это определённое множество понятий и связей между ними, являющихся смысловой структурой рассматриваемой предметной области.

Концептуальная модель – модель предметной области, состоящей из перечня взаимосвязанных понятий, используемых для описания этой области, вместе со свойствами и характеристиками, классификацией этих понятий, по типам, ситуациям, признакам в данной области и законов протекания процессов в ней.

Концептуальная модель представляет собой описание основных классов предметной области и отношения между ними без учёта принятой модели БД и синтаксиса целевой СУБД.

Структуры данных, проектируемые в этом разделе, обеспечивают хранение и доступ к данным. Требуется описать типы сущностей (таблица 6).

Таблица 6 – Типы сущностей предметной области

Наименование	Описание	Режим обновления
Заболевания	Список возможных заболеваний	1 раз в неделю
Симптомы	Список возможных симптомов заболеваний	1 раз в неделю
Лекарство	Список лекарственных препаратов	1 раз в неделю
Симптомы заболеваний	Список связей заболеваний и симптомов	1 раз в неделю
Лечение	Список связей заболеваний и лекарств	1 раз в неделю
Бюджетные аналоги	Список бюджетных аналогов лекарств	1 раз в неделю

У описанных выше сущностей имеются связи для взаимодействия данных между сущностями (таблица 7).

Таблица 7 – Типы связей предметной области

Тип сущности 1	Тип сущности 2	Описание	Мощность	Обязательность	Ограничения
Симптомы	Симптомы заболеваний	Реализует связь многие ко многим с сущностью заболевания через таблицу посредник симптомы заболевания	1:M	Обязательная, поскольку у каждого симптома должно обязательно быть заболевание	ID симптома должно существовать в таблице «Симптомы заболевания»
Симптомы заболеваний	Заболевания	Реализует связь многие ко многим с сущностью симптомы через таблицу посредник симптомы заболевания	1:M	Обязательная, поскольку у каждого заболевания должен обязательно быть симптом	ID заболевания должно существовать в таблице «Симптомы заболевания»
Заболевания	Лечение	Реализует связь многие ко многим с сущностью лекарство через таблицу посредник лечение	1:M	Обязательная, поскольку у каждого заболевания должно обязательно быть лекарство	ID заболевания должно существовать в таблице «Лечение»
Лечение	Лекарство	Реализует связь многие ко многим с сущностью заболевания через таблицу посредник лечение	1:M	Обязательная, поскольку у каждого лекарства должно обязательно быть заболевание	ID лекарства должно существовать в таблице «Лечение»
Лекарство	Бюджетные лекарства	Реализует связь один ко многим с сущностью лекарства и бюджетные аналоги	1:M	Не обязательная, поскольку не у каждого лекарства должен обязательно быть бюджетный аналог	ID лекарства должно существовать в таблице «Бюджетные аналоги»

На основе выше описанных таблиц были спроектированы атрибуты сущностей, которые представлены в приложении Б.

На основе концептуальной модели в дальнейшем будет проектироваться вся система приложения. Построенная исходная

концептуальная модель с описанием основных сущностей, связей между ними, а также с описанием атрибутов к каждой сущности, представлена на рисунке 6.

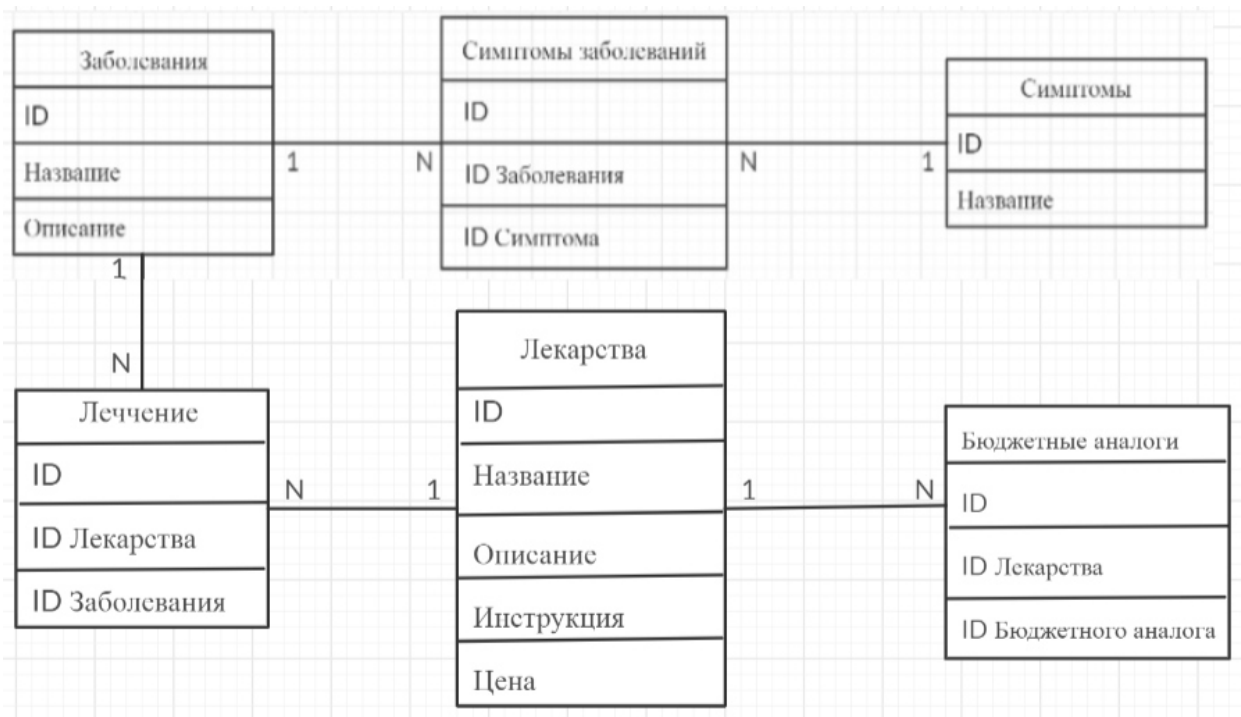


Рисунок 6 – Концептуальная модель предметной области (диаграмма сущность-связь)

В данном разделе были спроектированы структуры данных на основе предметной области, описаны все типы сущностей, связи между ними, а также их атрибуты.

2.5 Описание требований к обобщённой архитектуре приложения

Общая архитектура системы состоит из клиента, это веб-приложение, из веб-сервера и сервера базы данных. Данная архитектура называется – клиент-сервер с сервером приложений [6].

Архитектура «клиент-сервер с сервером приложений» – это вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами,

и заказчиками услуг, называемыми клиентами. Посредником между серверами и клиентами служит API или, в случае веб-приложения, сайт.

Таким образом, можно предложить следующую обобщённую архитектуру приложения информационной системы (рисунок 7).

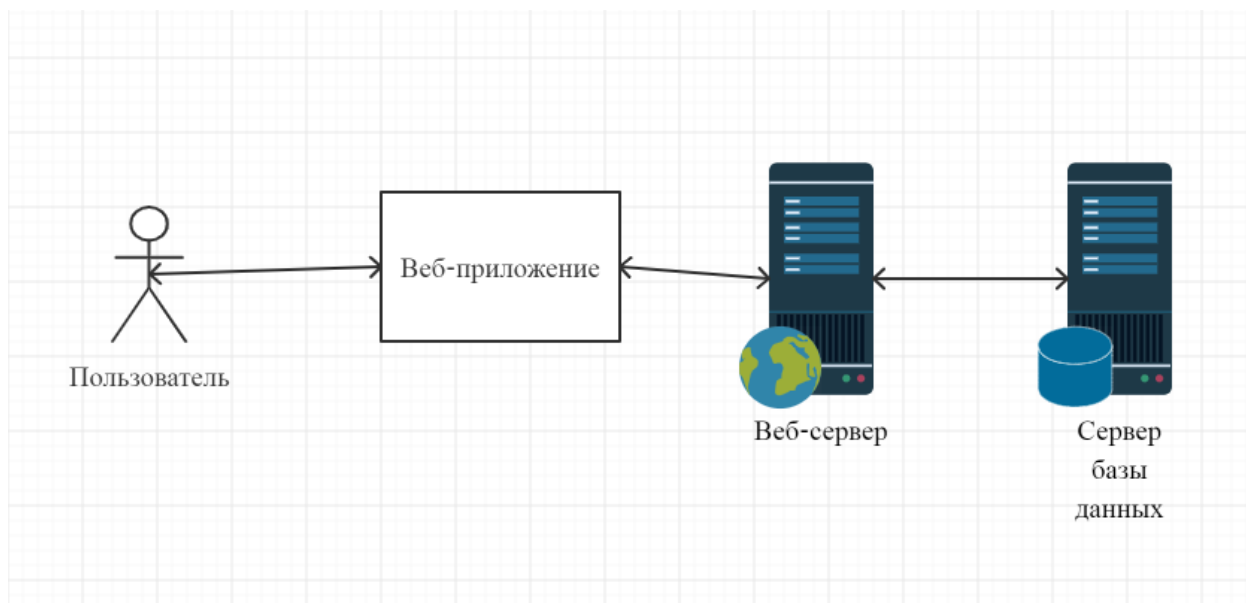


Рисунок 7 – Обобщенная архитектура приложения

Обобщённая архитектура приложения показывает, что веб-приложение пользователя, взаимодействует с веб-сервером. Достоинство данной системы в том, что ответы, которые будут получать клиенты, можно обработать на абсолютно разных устройствах, как мобильных, так и компьютерах, т.е. браузерах.

2.6 Выбор технологии и средств разработки

Выбор технологии и средства разработки всегда является одним из ключевых вопросов при разработке приложения. Необходимо выбрать технологии для следующих компонентов системы: СУБД, фреймворк для разработки.

2.6.1 Выбор системы управления базами данных

В настоящее время очень широкое распространение получили реляционные БД, которые представляют собой совокупность отношений, содержащих всю необходимую информацию и объединённых различными связями [7]. Поэтому в качестве СУБД была выбрана реляционная. Одной из лучших реляционных СУБД является MySQL. Простая в установке, работает отлично без особых настроек. Структура данной СУБД – таблицы. В любом языке программирования, фреймворке, CMS и так далее – есть интеграция с MySQL [8]. Данная СУБД работает успешно с различными сайтами и веб-приложениями. Из-за популярности MySQL, в интернете можно найти решение множества задач, связанных с данной СУБД. Разработку и поддержку MySQL осуществляет корпорация Oracle.

У выбранной СУБД множество достоинств.

1. Простота в работе – из-за большой популярности MySQL установить её очень просто. Дополнительные библиотеки, которые присутствуют во всех фреймворках, позволяют легко работать с БД.
2. Богатые функционал – MySQL поддерживает большинство функционала SQL.
3. Безопасность – множество функций, обеспечивающих безопасность, поддерживаются по умолчанию.
4. Масштабируемость – MySQL легко работает с большим количеством данных и легко масштабируется
5. Скорость – упрощение некоторых стандартов позволяет MySQL значительно увеличить производительность.

Благодаря выше перечисленным достоинствам, в качестве СУБД для мобильного приложения органайзера была выбрана MySQL.

2.6.2 Выбор фреймворков для разработки программного интерфейса

Для разработки серверной части приложения был выбран фреймворк Laravel. Веб-сервер отвечает за обработку запросов со стороны веб-приложения. Веб-сервер обращается к БД для получения информации, обрабатывает её и возвращает её запрашиваемому приложению.

У фреймворка Laravel немало достоинств.

1. Поддержка MVC (Model-View-Controller).
2. Фреймворк использует замечательный шаблонизатор Blade, который позволяет отделить вёрстку от PHP-кода.
3. Удобная маршрутизация и валидация данных. Маршрутизация позволяет с лёгкостью создавать API запросы и обычные запросы, которые в дальнейшем обрабатываются контроллером.
4. Поддержка работы с различными БД.
5. Огромный функционал Eloquent ORM включает в себя защиту от атак типа SQL Injection. Позволяет загружать данные из нескольких таблиц или же обрабатывать данные из БД частями.
6. Laravel Collections – как PHP массивы, но с продвинутыми возможностями, которые экономят массу времени.
7. Кэширование файлов маршрутизации, конфигурации, шаблонов. Ускоряет работу фреймворка.
8. Поддержка WebSockets для создания интерактивных приложений.
9. Огромные возможности для тестирования веб-проекта, включая заполнение базы данных тестовыми данными.
10. Поддержка полнотекстового поиска.

Данный фреймворк помогает и в разработке веб-приложения для браузера [9].

2.6.3 Средства визуализации

Для создания визуальной части веб-приложения был создан первоначальный макет внешнего вида в Photoshop. Далее с помощью языка HTML произвели верстку веб-страниц.

Основное преимущество HTML заключается в том, что документ может быть просмотрен на веб-браузерах различных типов и на различных платформах. Все что необходимо, чтобы прочитать HTML-документ - это веб-браузер, который интерпретирует элементы HTML и воспроизводит на экране документ в виде, который ему придает автор. HTML-документы могут быть созданы при помощи любого текстового редактора или специализированных HTML-редакторов и конвертеров. Как любой язык, HTML позволяет вставлять в тело документа комментарии, которые сохраняются при передаче документа по сети, но не отображаются браузером[21].

Следующим действием мы отредактировали наши веб-страницы с помощью языка таблиц стилей CSS. Отделяя стиль представления документов от содержимого документов, CSS упрощает создание веб-страниц и обслуживание сайта. CSS поддерживает таблицы стилей для конкретных носителей, поэтому мы можем адаптировать представление своих документов к визуальным браузерам, слуховым устройствам, принтерам, брайлевским устройствам, карманным устройствам и т.д.

Ну и конечно для того чтобы веб-приложение реагировало на действия пользователя был использован язык JavaScript. Сейчас JavaScript — единственный язык программирования для браузеров. Он работает под Windows, macOS, Linux и на мобильных платформах, то есть везде. JavaScript полностью интегрирован с HTML, он способен как угодно менять веб-страницу, программа на JavaScript — это простой текст. Писать на нём можно в любом текстовом редакторе.

JavaScript — можно назвать языком сценариев или скриптов. Скрипт представляет собой программный код — набор инструкций, который не требует предварительной обработки (например, компиляции) перед запуском. Код интерпретируется движком браузера во время загрузки веб-страницы. Интерпретатор браузера выполняет построчный анализ, обработку и выполнение исходной программы или запроса.

JavaScript — объектно-ориентированный язык с прототипным наследованием. Он поддерживает несколько встроенных объектов, а также позволяет создавать или удалять свои собственные (пользовательские) объекты. Объекты могут наследовать свойства непосредственно друг от друга, образуя цепочку объект-прототип.

3 Разработка программного обеспечения веб-приложения для поиска лекарственных препаратов

3.1 Планирование бюджета проекта

При проектировании проекта необходимо обосновать целесообразность разработки с точки зрения затрат на ресурсы. В расчёт был взят 7-ми месячный временной интервал, который в свою очередь состоит из 6-ти месяцев потраченных на разработку и 1-го месяца тестовой эксплуатации, все расходы учтены в табл. 8. В расчёте не учитывали затраты на приобретение компьютеров, комплектующих, сервера, программного обеспечения, оргтехники. В таблицу расходов были внесены такие пункты как:

- зарплата программиста;
- зарплата администратора;
- оплата электроэнергии;
- оплата интернет-соединения;
- единый социальный налог.

Таблица 8 – Расходы на разработку и тестовый запуск веб-приложения для поиска лекарственных препаратов

№	Статья расходов	Сумма
1	Зарплата программиста за 7 месяцев (разработка и тестовая эксплуатация) с окладом $18\,000 = 18\,000 * 7$	126 000
2	Зарплата администратора за 1 месяц (тестовой эксплуатации) с окладом $13\,000$ $= 13\,000 * 1$	13 000
3	Оплата электроэнергии за 7 месяцев	3 738
4	Оплата интернет-соединения	6 450

Продолжение таблицы 8

№	Статья расходов	Сумма
5	Единый социальный налог 26% (за 7 месяцев работы программиста и 1 месяц работы администратора) (18 000*26%)*7+(13 000*26%)	36 140
6	Всего вложений	185 328

Также была составлена таблица расходов при последующей эксплуатации и продвижения веб-приложения за месяц табл.9. Для популяризации веб-приложения нам потребуется реклама, а так же SMM-менеджер, поэтому в расходы мы включили следующие пункты:

- зарплата программиста за месяц;
- зарплата администратора за месяц;
- зарплата SMM-менеджера за месяц;
- единый социальный налог;
- оплата электроэнергии;
- оплата интернет-соединения;
- затраты на рекламу.

Таблица 9 – Расходы на эксплуатацию и продвижение веб-приложения для поиска лекарственных препаратов за месяц

№	Статья расходов	Сумма
1	Зарплата программиста за месяц	18 000
2	Зарплата администратора за месяц	13 000
3	Зарплата SMM-менеджера за месяц	23 000
4	Единый социальный налог 26% (за месяц работы программиста, администратора и SMM-менеджера) 18 000*26%+13 000*26%+20 000*26%	14 040

Продолжение таблицы 9

5	Оплата электроэнергии за месяц	534
6	Оплата интернет-соединения за месяц	922
7	Затраты на рекламу за месяц	25 000
8	Всего	94 496

3.2 Разработка архитектуры программного обеспечения

Для разработки данного ПО была использована MVC модель. В веб-разработке MVC является самым популярным паттерном. Сегодня понятие MVC-фреймворк достаточно обыденно, и мы привыкли считать, что окружающие знают об этих фреймворках достаточно, чтобы правильно разделять код на три и более составляющие (рисунок 8).

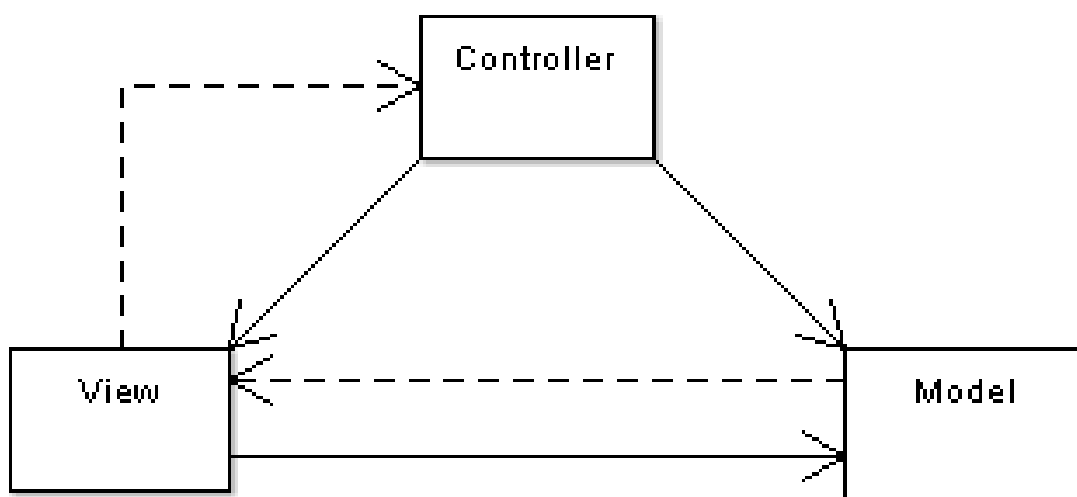


Рисунок 8 – Шаблон MVC

Model является «сутью» приложения и отвечает за непосредственные алгоритмы, расчёты и тому подобное внутреннее устройство приложения. Также предоставляет линк к хранилищу данных.

View предназначен для вывода данных, предоставленных Model. Это единственная часть MVC, которая непосредственно контактирует с пользователем.

Controller получает данные от пользователя и передаёт их в Model. Кроме того, он получает сообщения от Model и передаёт их в View.

Суть паттерна – подстроить MVC под http протокол и реалии Интернета. Данный паттерн рассматривал контроллер как объект, принимающий запрос от пользователя. Контроллер должен разобрать запрос, на его основе инстанцировать определенную модель и вывести определенное представление, осуществив проброс модели и других данных в представление. Модель использовалась для общения с базой данных и отображения строк таблицы в объекте (ORM).

MVC имеет ряд преимуществ.

1. MVC – единая концепция системы. Несомненным достоинством MVC является единая глобальная архитектура приложения. Даже в сложных системах разработчики (как те, которые разрабатывали систему, так и вновь присоединившиеся) могут легко ориентироваться в программных блоках. Например, если возникла ошибка в логике обработки данных, разработчик сразу отбрасывает 2 блока программы (controller и view) и занимается исследованием 3-го (model).

2. В MVC упрощен механизм отладки приложения. Так как весь механизм визуализации теперь сконцентрирован в одном программном блоке, упростились механизмы опционального вывода графических элементов.

MVC не обходится без недостатков.

1. В MVC усложнен механизм разделения программы на модули. В концепции MVC наличие трех блоков (Model, View, Controller) прописано жестко. Соответственно каждый функциональный модуль должен состоять из трех блоков, что, в свою очередь, несколько усложняет архитектуру функциональных модулей программы.

2. В MVC усложнен процесс расширения функционала. Проблема очень схожа с вышеописанной. Недостаточно просто написать

функциональный модуль и подключить его в одном месте программы. Каждый функциональный модуль должен состоять из трех частей, и каждая из этих частей должна быть подключена в соответствующем блоке [10-11].

Active Record – это модель в Laravel. Модель хранит данные и предоставляет базу для работы с данными. Кроме этого Active Record также является ORM фреймворком. ORM значит Object-relational mapping (Объектно-реляционная проекция).

ORM – технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных».

3.3 Разработка классов модели приложения

На основе диаграммы сущность-связь, пункт 2.2 данной работы, была составлена диаграмма классов (рисунок 9).

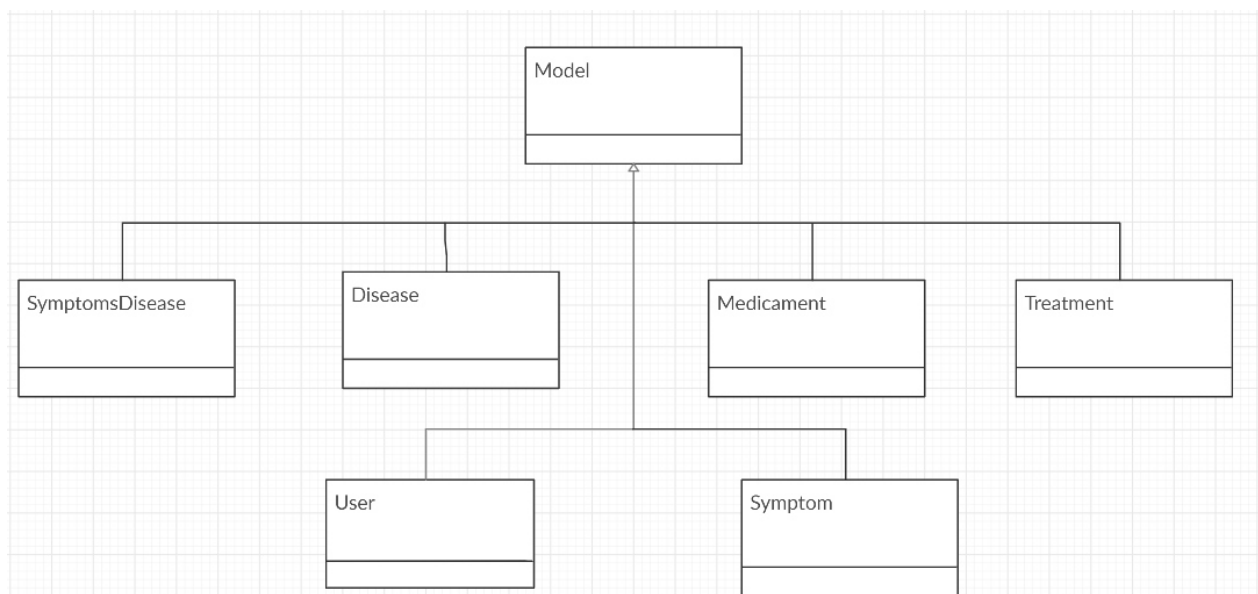


Рисунок 9 – Классы моделей (диаграмма классов)

Данная диаграмма описывает модели веб-приложения. Каждая модель – это класс, реализованный со своими настройками в файле. Набор классов представляет собой объекты, которые описывают сущность каждой таблицы базы данных.

Класс «Model» – базовый класс в котором описан функционал действий над базой данных [12].

Помимо базового класса «Model», были разработаны следующие классы:

- 1) «Disease» – отвечает за работу и хранение данных различных активностей человека;
- 2) «Medicament» – отвечает за работу над новостями мобильного приложения;
- 3) «Symptom» – содержит и работает с данными категорий для активностей;
- 4) «SymptomsDisease» – отвечает за работу с данными чатов пользователей;
- 5) «User» – работает со всеми пользователями, которые зарегистрировались в приложении;
- 6) «Treatment» – работает со всеми пользователями, которые зарегистрировались в приложении.

Более подробная спецификация разработанных классов представлена в приложении В.

В данном разделе работы был описан и реализован компонент Model шаблона MVC. Также были описаны разработанные классы и их назначение в веб-приложении для поиска лекарственных препаратов. Код реализации моделей представлен в приложении Г.

3.4 Разработка методов взаимодействия модели с базой данных

Взаимодействие с базой данных происходит с помощью моделей и контроллеров. В пункте 3.2 данной работы были описаны модели, которые представляют собой таблицы, находящиеся в базе данных. Для каждой из этих моделей должен быть реализован свой отдельный контроллер. Контроллер представляет собой набор методов, которые выполняют какую-

либо работу с данными. Основными методами контроллера являются: index, show, create, edit, store, update, delete. Естественно эти методы не являются необходимыми и могут дополняться или удаляться.

С реализованными моделями взаимодействуют контроллеры (рисунок 10).

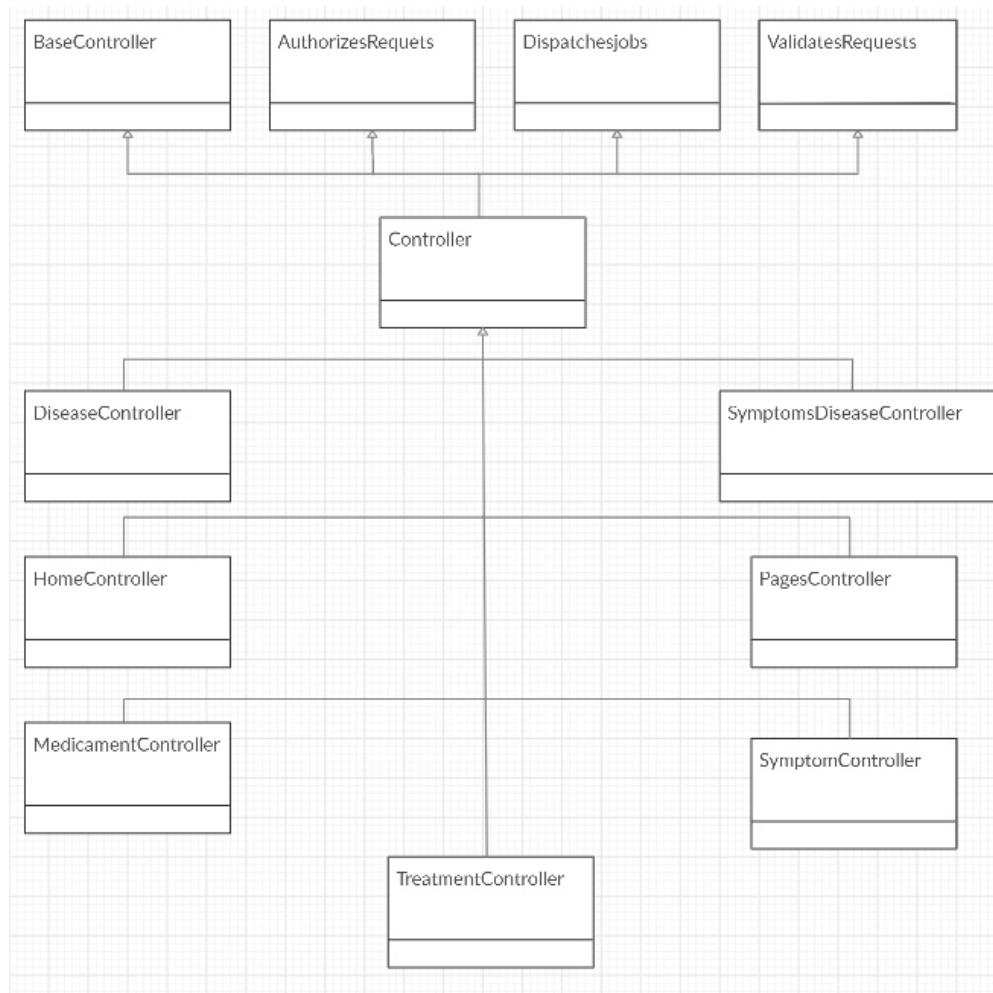


Рисунок 10 – Классы контроллеров (диаграмма классов)

Исходя из рисунка 10 можно сделать вывод, что разрабатывается достаточно обширная система. Каждый из проектированных классов отвечает за работу над определёнными данными.

Рассмотрев схему, можно выделить один общий класс – Controller, который наследует, в свою очередь, некоторые классы, которые необходимы

для полноценной работы всей системы. Controller наследует следующие классы:

- 1) класс BaseController – абстрактный базовый класс, в котором описан функционал наследуемых контроллеров [13-14];
- 2) класс AuthorizesRequests – класс проверки доступности контроллеров для текущего запроса [15];
- 3) класс ValidatesRequests – класс проверки правильности поступившего запроса [16];
- 4) класс DispatchesJobs – вспомогательный класс для преобразования данных [17].

Описание разработанных контроллеров мобильного приложения более подробно представлено в таблицах 10 – 16.

Таблица 10 – Спецификация класса DiseaseController

Класс	DiseaseController
Назначение	Работа с заболеваниями
Основные атрибуты	–
Методы	index – возвращает список заболеваний edit – возвращает страницу для редактирования заболевания show – возвращает запрашиваемое заболевание update – обновляет записи заболевания create – возвращает страницу с созданием заболевания store – сохраняет новое заболевание в БД delete – удаляет заболевание

Таблица 11– Спецификация класса HomeController

Класс	HomeController
Назначение	Возвращает главную страницу сайта
Основные атрибуты	–
Методы	–

Таблица 12 – Спецификация класса MedicamentController

Класс	MedicamentController
Назначение	Работа с лекарственными препаратами
Основные атрибуты	–
Методы	index – возвращает список лекарств edit – возвращает страницу для редактирования лекарств show – возвращает запрашиваемое лекарство update – обновляет записи лекарства create – возвращает страницу с созданием лекарства store – сохраняет новое заболевание в БД delete – удаляет заболевание

Таблица 13 – Спецификация класса TreatmentController

Класс	TreatmentController
Назначение	Присвоение лекарства определённому заболеванию
Основные атрибуты	–
Методы	index – возвращает список связей между таблицами заболевания и таблицей лекарства edit – открывает страницу с изменяемым связей update – обновляет связь create – открывает страницу для создания новой связи store – сохраняет новую связь delete – удаляет связь

Таблица 14 – Спецификация класса SymptomsDiseaseController

Класс	SymptomsDiseaseController
Назначение	Присвоение симптома определённому заболеванию
Основные атрибуты	–
Методы	index – возвращает список связей между таблицами заболевания и таблицей симптома edit – открывает страницу с изменяемым связей update – обновляет связь create – открывает страницу для создания новой связи store – сохраняет новую связь delete – удаляет связь

Таблица 15 – Спецификация класса PagesController

Класс	PagesController
Назначение	Работа с отображением страниц для пользователя
Основные атрибуты	–
Методы	<p>medicaments – возвращает страницу со списком медикаментов</p> <p>diseases – возвращает страницу со списком заболеваний</p> <p>medicament – возвращает страницу с медикаментом</p> <p>disease – возвращает страницу с заболеванием</p> <p>symptoms – возвращает страницу со списком симптомов</p> <p>findDiseases – поиск заболевания по симптомам</p> <p>findMedicament – поиск медикамента по названию</p>

Таблица 16 – Спецификация класса SymptomController

Класс	SymptomController
Назначение	Работа с симптомами
Основные атрибуты	–
Методы	<p>index – возвращает список симптомов</p> <p>edit – возвращает страницу для редактирования симптомов</p> <p>show – возвращает запрашиваемый симптом</p> <p>update – обновляет записи симптома</p> <p>create – возвращает страницу с созданием симптома</p> <p>store – сохраняет новый симптом в БД</p> <p>delete – удаляет симптом</p>

Одна из важных функций представлена в виде диаграммы последовательности на рисунке 11.

3.5 Разработка пользовательского интерфейса

Интерфейс веб-приложения – это веб-страница или совокупность веб-страниц, предоставляющая пользовательский интерфейс для взаимодействия с сервисом или устройством посредством протокола HTTP и веб-браузера. Одним из основных требований к веб-интерфейсам является их одинаковый внешний вид и одинаковая функциональность при работе в различных браузерах.

Разработанный интерфейс должен быть интуитивно понятен и прост. В то же время он должен содержать в себе весь функционал, который будет легко доступен [18]. В Laravel используются blade шаблоны, которые помогают серверу приложений быстро и не дублируя код собирать страницы для пользователя. Все эти страницы играют роль View из архитектуры MVC (рисунок 12).



Рисунок 12 – Иерархия переходов

Ниже представлено более подробное описание прототипа каждого экрана.

Элементы интерфейса формы «Экран главной страницы», представленные на рисунке 13:

- 1 – кнопка для авторизации администратора;
- 2 – кнопка для начала поиска лекарства по его названию;
- 3 – кнопка для начала поиска лекарства по названию болезни;
- 4 – кнопка для начала поиска лекарства по симптомам заболевания.

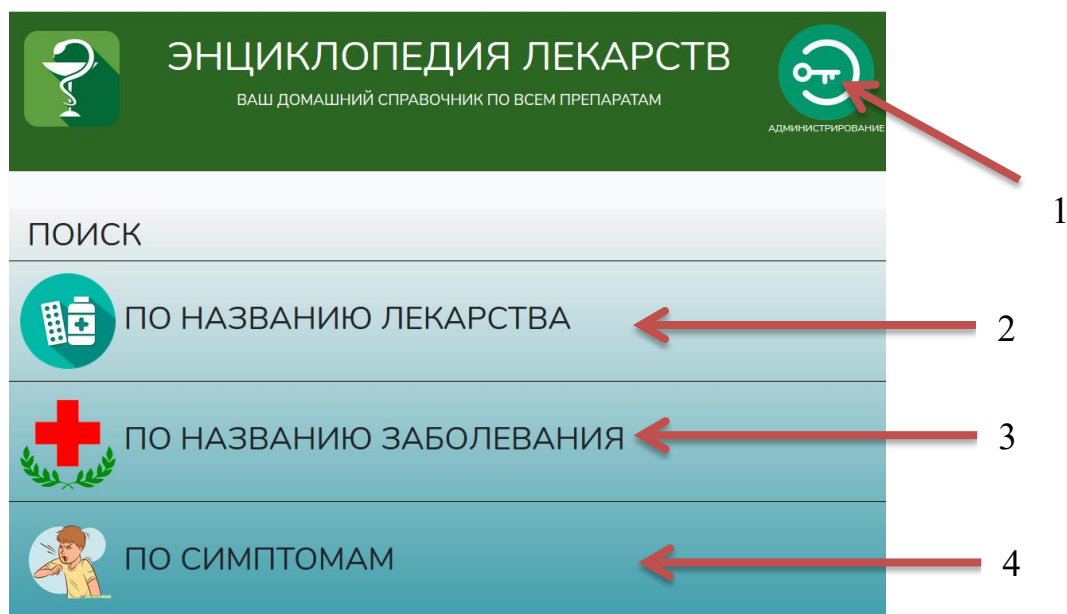


Рисунок 13 – Прототип главной страницы

Элементы интерфейса формы «Экран регистрации», представленные на рисунке 14:

- 1 – кнопка для регистрации нового администратора;
- 2 – кнопка для входа уже зарегистрированного администратора;
- 3 – поле для ввода имени;
- 4 – поле для ввода почты;
- 5 – поле для ввода пароля;
- 6 – поле для подтверждения введённого пароля;
- 7 – кнопка подтверждения регистрации.

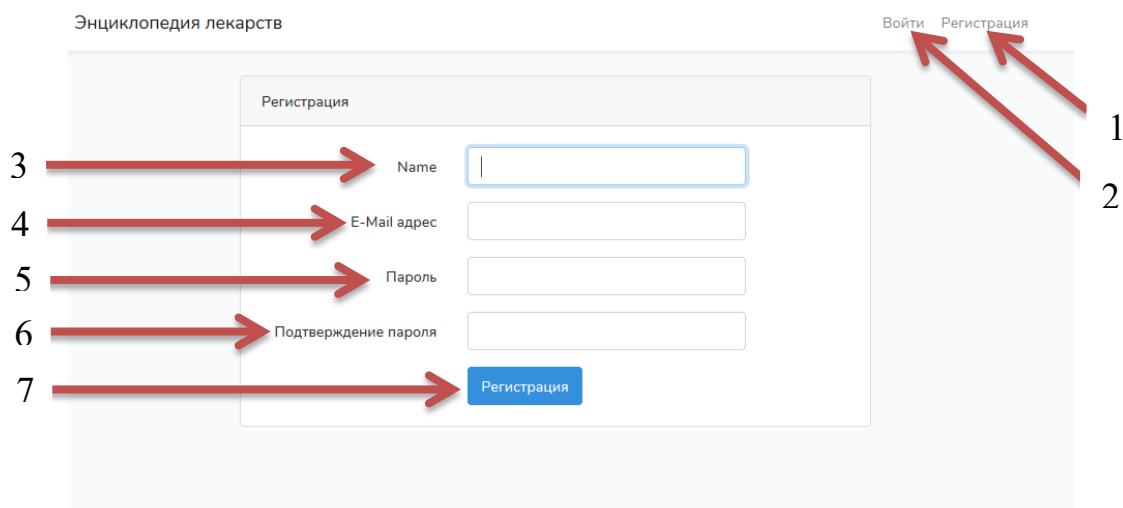


Рисунок 14 – Прототип экрана регистрации

Элементы интерфейса формы «Экран админ панели», представленные на рисунке 15:

- 1 – имя зашедшего администратора;
- 2 – кнопка для перехода к форме добавления нового лечения;
- 3 – кнопка для перехода к форме добавления новой связки симптома и заболевания;
- 4 – кнопка для перехода к форме добавления нового симптома болезни;
- 5 – кнопка для перехода к форме добавления нового заболевания;
- 6 – кнопка для перехода к форме добавления нового лекарства.

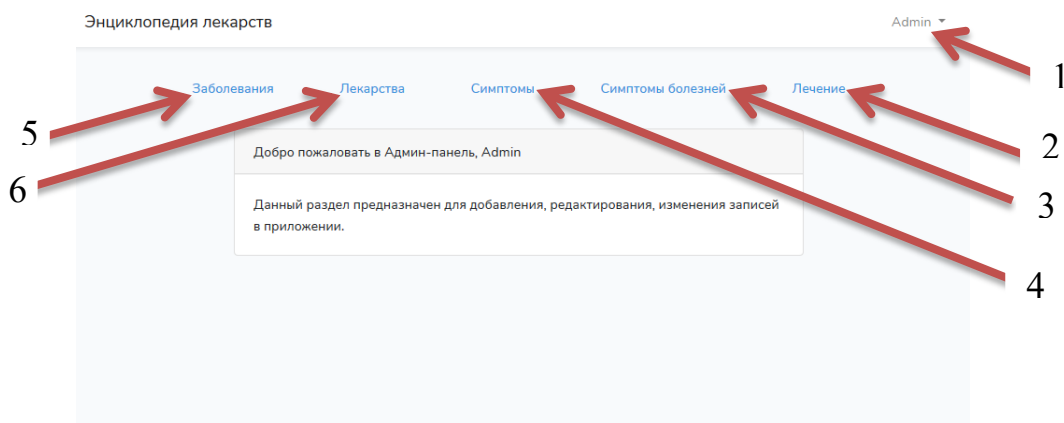


Рисунок 15 – Прототип экрана админ панели

Элементы интерфейса формы «Экран поиска лекарства по названию», представленные на рисунке 16:

- 1 – кнопка возврата на главную страницу;
- 2 – поле для ввода названия лекарства;
- 3 – кнопка для начала поиска;
- 4 – окно результатов поиска.

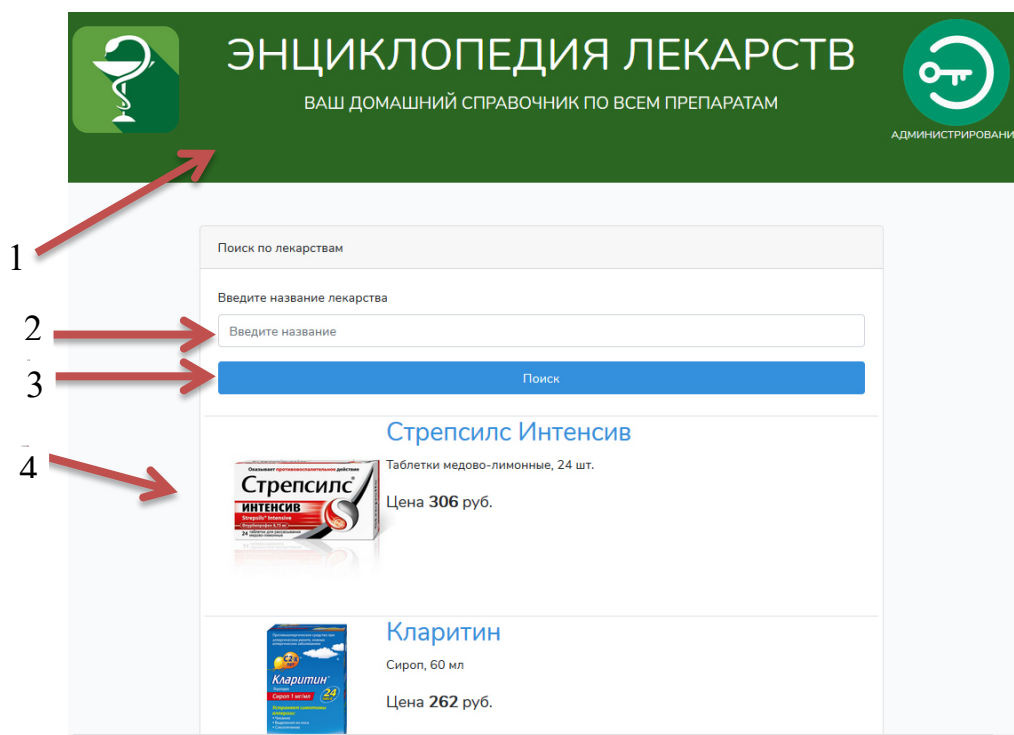


Рисунок 16 – Прототип экрана поиска лекарства по названию

Элементы интерфейса формы «Экран поиска лекарства по названию заболевания», представленные на рисунке 17:

- 1 – поле для ввода названия лекарства;
- 2 – кнопка для открытия подробной информации о заболевании;
- 3 – кнопка для перехода к лекарству от соответствующего заболевания;
- 4 – кнопка для выбора количество отображающихся заболеваний на одной странице;
- 5 – окно результата по запрашиваемым заболеваниям.

Поиск по заболеваниям

Количество элементов: 10

Поиск

Название ↑↓	Инструкция ↑↓	Лекарства ↑↓	Действия ↑↓
Аллергия	Аллергия (др.-греч. ἄλλος — другой, иной, чужой + ἔργον — воздействие) — типовой иммунопатологический процесс, выраженный сверхчувствительностью иммунной системы организма при повторных воздействиях аллергена на ранее сенсибилизированный этим аллергеном организм.	Кларитин	Посмотреть полностью
Ангина	Острый тонзиллит (от лат. tonsillae — миндалины), в обиходе — ангина (от лат. angio — «сжимаю, сдавливаю, душу»), — инфекционное заболевание с местными проявлениями в виде острого воспаления компонентов лимфатического глоточного кольца, чаще всего нёбных миндалин, вызываемое стрептококками или стафилококками, реже другими микроорганизмами, вирусами и грибами (см. подробнее Этиология =[3]). Ангиной также называют обострение хронического тонзиллита.	Стрепсилс Интенсив	Посмотреть полностью
Гайморит	Гайморит (максиллит) — воспаление верхнечелюстной (гайморовой) придаточной пазухи носа, возникает как осложнение при остром насморке, гриппе, кори, скарлатине и других инфекционных заболеваниях.	Отривин	Посмотреть полностью

Рисунок 17 – Прототип экрана поиска лекарства по названию заболевания

Элементы интерфейса формы «Экран поиска лекарства по симптомам», представленные на рисунке 18:

- 1 – поле для ввода симптомов или выбора их из выпадающего списка;
- 2 – кнопка начала поиска.

ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ

АДМИНИСТРИРОВАНИЕ

Поиск по симптомам

Выберите симптомы

Поиск

Рисунок 18 – Прототип экрана поиска лекарства по симптомам

Элементы интерфейса формы «Экран информации о лекарстве», представленные на рисунке 19:

- 1 – основная информация о лекарстве: название, фото, цена;
- 2 – полная инструкция лекарства.

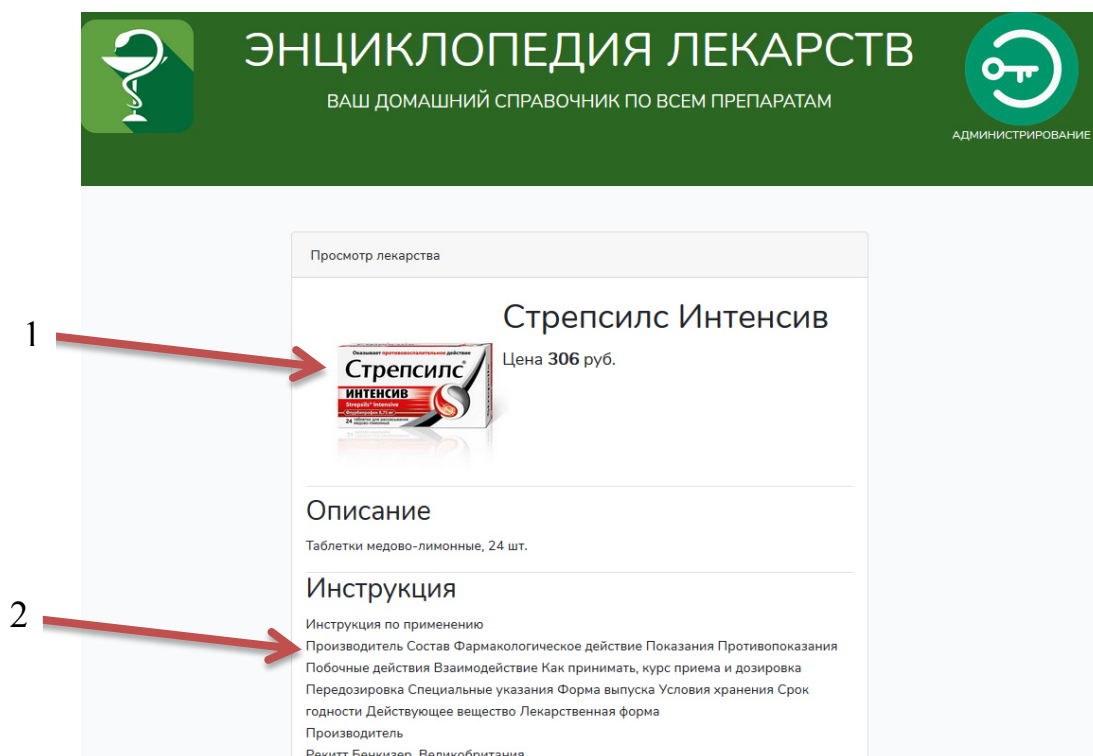


Рисунок 19 – Прототип экрана информации о лекарстве

В данном разделе был описан и реализован компонент View шаблона MVC [19]. Код реализации моделей представлен в приложении Е.

4 Тестирование и анализ качества программного обеспечения

4.1 Объект испытаний

Тестирование – это один из важных этапов разработки программного обеспечения. Оно проверяет надёжность и корректность программного проекта. Цель тестирования – сокращение к минимуму количества потенциальных ошибок программного обеспечения [20]. Предметом проверки является программное обеспечение, реализованное как веб-приложение и предназначенное для поиска лекарственных препаратов.

4.2 Цель испытания

Убедиться в правильности и адекватности выполнения требуемых от программного обеспечения задач. Испытать надёжность и отказоустойчивость системы, проверив все возможные варианты взаимодействия с ней.

Оценка работы системы будет рассчитываться исходя из результатов выполнения тестовых проверок.

4.3 Требования к программе

Веб-приложение должно позволить выполнение следующих задач:

- аутентифицировать и регистрировать нового администратора;
- ведение и учёт базы лекарственных препаратов, симптомов и заболеваний;
- возможность вести полную работу со связями препаратов и заболеваний, а также заболеваний и симптомов создавать/редактировать/удалять их;
- позволять осуществлять поиск требуемого лекарственного препарата тремя разными способами – по названию лекарства/по названию заболевания/по симптомам заболевания;
- задавать фильтры поиска лекарственных препаратов;

					ПГУ 1.090501.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		47

Выходными данными приложения должны быть:

- информация по найденному лекарственному препарату;
- вывод самого дешёвого аналогового лекарственного препарата.

4.4 Средства и порядок испытаний

Тестирование программного обеспечения должно проводиться на персональном компьютере с минимально допустимыми характеристиками под управлением OS Windows 7 и выше, на котором должен быть установлен любой доступный веб-браузер и подключение к интернету.

Оценка правильности и полноты работы программного обеспечения будет осуществляться путём визуального анализа результатов проверки всего функционала программы.

4.5 Методы испытаний

Испытание программного обеспечения должно проводиться на основе тестовых примеров, приведённых в таблице 17.

Таблица 17 – Сценарий пользовательского теста

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
1.Запуск веб-приложения	Запустить веб-приложение в браузере	Должна отобразиться главная страница веб-приложения	Успешно
2.Регистрация нового администратора	На главной странице есть три кнопки для поиска и кнопка «администрирование», Именно она нас и интересует, при нажатии на неё должен открыться экран входа в учётную запись	При нажатии на главной странице кнопки «администрирование», открывается форма для входа администратора, после нажатия на кнопку «регистрации», перед нами открывается окно для ввода: имени, почты, пароля и	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
	администратора, далее, должна отобразиться кнопка «регистрации», нажав на которую перед нами откроится форма, где нужно ввести: имя, почту, пароль и подтверждение пароля. И уже после нажатия кнопки «регистрация», данные нового администратора сохраняются в базе.	подтверждение пароля. После нажатия на кнопку «регистрация», данные нового администратора сохранились в базе.	
3. Ввод не правильного логина и пароля при входе администратора	При открытии экрана входа в систему, вводим неправильные данные, а именно: почта:admIn@mail.ru, пароль: qwert1234. В ответ на введенные данные программа должна отреагировать ошибкой и выводом окна с описанием проблемы, доступ к панели администратора не должен быть получен.	Открываем окно входа в систему, вводим данные. Система выдала ошибку, доступ к админ панели не получен, всплыло окно с описанием ошибки и просьбой проверить введенные данные и повторить попытку.	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
4. Ввод правильного логина и пароля при входе администратора	При открытии экрана входа в систему, вводим правильные данные, а именно: почта:admin@mail.ru, пароль: qwert12345. В ответ на введенные данные программа должна предоставить доступ к админ панели данному администратору.	Открываем окно входа в систему, вводим данные. Система приняла введенные данные и впустила нас под учетной записью администратора, открыв экран админ панели.	Успешно
5. Добавление нового лекарственного препарата	На экране админ панели, при нажатии кнопки «Лекарства», должна открыться форма со списком препаратов, при нажатии на кнопку «Создать», открывается форма с полями для добавления: названия, краткого описания, инструкции, фотографии и цены. После введения и нажатия на кнопку «Добавить» лекарство, записи должны добавиться в базу	На экране админ панели нажимаем кнопку «Лекарства», открывается форма с лекарствами, далее нажимаем на кнопку «Создать», открывается форма для создания нового препарата. Заносим все данные в форму. Нажимаем кнопку «Добавить». Приложение выдает сообщение о том, что лекарство успешно добавлено.	Успешно

	данных.		
--	---------	--	--

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
6.Добавление нового заболевания	На экране админ панели, при нажатии кнопки «Заболевания», далее на кнопку «Создать» должна открыться форма с полями для добавления: названия и описания заболевания. После введения и нажатия на кнопку «Добавить», записи должны добавиться в базу данных.	На экране админ панели нажимаем кнопку «Заболевания», открывается форма, для добавления нового заболевания нажимаем на кнопку «Создать». Заносим все данные в форму. Нажимаем кнопку «Добавить». Приложение выдаёт сообщение о том, что заболевание успешно добавлено.	Успешно
7.Добавление нового симптома	На экране админ панели, при нажатии кнопки «Симптомы», далее на кнопку «Создать» должна открыться форма с полями для добавления: названия симптома. После введения и нажатия на кнопку «Добавить», записи должны добавиться в базу данных.	На экране админ панели нажимаем кнопку «Симптомы», открывается форма, для добавления нового симптома нажимаем на кнопку «Создать». Заносим все данные в форму. Нажимаем кнопку «Добавить». Приложение выдаёт сообщение о том, что симптом успешно добавлен.	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
8.Добавление новой связи заболевания и симптома	На экране админ панели, при нажатии кнопки «Симптомы болезней», далее на кнопку «Создать» должна открыться форма с двумя выпадающими списками: заболевание и симптомы. Сопоставляем выбранное заболевание с симптомом и нажимаем на кнопку «Сохранить», записи должны добавиться в базу данных.	На экране админ панели нажимаем кнопку «Симптомы болезней», открывается форма, для добавления новой связи заболевания и симптома, нажимаем на кнопку «Создать».Выбираем нужное нам заболевание и соответствующие симптомы . Нажимаем кнопку «Сохранить». Приложение выдаёт сообщение о том, что симптомы болезни успешно добавлены.	Успешно
9. Добавление новой связи заболевания и лекарства	На экране админ панели, при нажатии кнопки «Лечение», далее на кнопку «Создать» должна открыться форма с двумя выпадающими списками: заболевание и лекарства. Сопоставляем выбранное заболевание с лекарством и нажимаем на	На экране админ панели нажимаем кнопку «Лечение», открывается форма, для добавления новой связи заболевания и лекарства, нажимаем на кнопку «Создать». Выбираем нужное нам заболевание и соответствующее лекарство. Нажимаем кнопку «Сохранить». Приложение выдаёт сообщение о том, что лечение	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
	кнопку «Сохранить», записи должны добавиться в базу данных.	успешно добавлено.	
9. Поиск лекарства по названию	На экране главной страницы при нажатии кнопки «По названию лекарства», должен открыться экран поиска по названию лекарства с полем для ввода названия, после ввода и нажатия на кнопку «Поиск», программа должна выдать результат. После этого по нажатию на поле с результатом должен открыться экран с полной информацией по данному лекарственному препарату.	На экране главной страницы нажимаем на кнопку «По названию лекарства», открывается экран поиска по названию лекарства с полем для ввода названия, вводим название лекарства: “Стрепсилс” и нажимаем на кнопку «Поиск», программа выдала результат: “Стрепсилс Интенсив”. После этого по нажатию на название лекарства, открылся экран с полной информацией по “Стрепсилс Интенсив”.	Успешно
10. Поиск лекарства по названию заболевания	На экране главной страницы при нажатии кнопки «По названию заболевания», должен открыться экран поиска по названию заболевания с полем для ввода названия, после	На экране главной страницы нажимаем на кнопку «По названию заболевания», открывается экран поиска по названию заболевания с полем для ввода названия, вводим название заболевания:	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
	ввода и нажатия на кнопку «Поиск», программа должна выдать результат. После этого по нажатию на поле с результатом должен открыться экран с полной информацией по данному лекарственному препарату.	“Ангина” и нажимаем на кнопку «Поиск», программа выдала результат: заболевание – ангина, лекарство: “Стрепсилс Интенсив”. После этого по нажатию на название лекарства, открыться экран с полной информацией по “Стрепсилс Интенсив”.	
11. Поиск лекарства по симптомам	На экране главной страницы при нажатии кнопки «По симптомам», должен открыться экран поиска по симптомам заболевания с полем для ввода, после ввода или выбора симптомов из выпадающего списка и нажатия на кнопку «Поиск», программа должна выдать результат. После этого по нажатию на поле с результатом должен открыться экран с полной информацией по данному лекарственному препарату.	На экране главной страницы нажимаем на кнопку «По симптомам», открывается экран поиска по симптомам с полем для ввода названия, вводим название симптома: “Боль в горле” и нажимаем на кнопку «Поиск», программа выдала результат: заболевание – ангина, лекарство: “Стрепсилс Интенсив”. После этого по нажатию на название лекарства, открыться экран с полной информацией по “Стрепсилс Интенсив”.	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
12. Поиск самого бюджетного аналога лекарства	На экране главной страницы при нажатии кнопки «По названию лекарства», должен открыться экран поиска по названию лекарства с полем для ввода названия, после ввода и нажатия на кнопку «Поиск», программа должна выдать результат. После этого по нажатию на поле с результатом должен открыться экран с полной информацией по данному лекарственному препарату, в кратком описании нажимаем на название бюджетного аналога, должна открыться информация по данному препарату.	На экране главной страницы нажимаем на кнопку «По названию лекарства», открывается экран поиска по названию лекарства с полем для ввода названия, вводим название лекарства: “Стрепсилс” и нажимаем на кнопку «Поиск», программа выдала результат: “Стрепсилс Интенсив”. После этого по нажатию на название лекарства, открылся экран с полной информацией по “Стрепсилс Интенсив”, рядом с кратким описанием нажимаем на название бюджетного аналога: “Аджисепт” открывается экран информации о данном лекарственном препарате.	Успешно
13. Использование фильтров поиска необходимой информации	На экране главной страницы при нажатии кнопки «По названию лекарства», должен открыться экран поиска по названию лекарства с полем	На экране главной страницы нажимаем на кнопку «По названию лекарства», открывается экран поиска по названию лекарства с полем для ввода названия,	Успешно

Продолжение таблицы 17

Вид тестирования	Алгоритм	Ожидаемый результат	Полученный результат
	для ввода названия. А так же есть возможность сортировки: 1 – по цене:возрастание, убывание. Для этого необходимо нажать на вкладку сортировка. 2- по количеству отображаемых на одной странице лекарственных препаратов: 10,25,100. Для этого необходимо нажать на вкладку количество элементов.	Нажимаем на вкладку сортировка: по цене - по возрастанию. Лекарственные препараты отсортировались и начинаются от самого дешёгого и до сомого дорогого. Нажимаем на вкладку сортировка по количеству отображаемых на одной странице препаратов и выбираем 10, теперь на одной странице отображается только 10 лекарственных препаратов.	
14.Выход из профиля администратора	Для выхода из профиля администратора нужно нажать на имя администратора в верхнем правом углу и в выпадающем списке нажать на кнопку «Выход», вас выкинет из учётной записи на главную страницу приложения	Нажимаем на имя администратора: Admin. Выпадает список с кнопкой «Выход», нажимаем на неё и нас выбрасывает из профиля, мы попадаем на главную страницу приложения.	Успешно

ЗАКЛЮЧЕНИЕ

В рамках выполнения выпускной квалификационной работы был разработан прототип веб-приложения для поиска лекарственных препаратов.

На этапе проведения анализа предметной области и требований были проанализированы существующие аналоги приложений для поиска лекарственных препаратов. На основе результатов анализа были определены преимущества и недостатки существующих приложений, а также разработаны функциональные требования к будущему веб-приложению.

Осуществлено планирование разработки веб-приложения, которое показало, что длительность реализации проекта составляет 7 месяцев и на его реализацию следует затратить 185 328 рублей, а на ежемесячную поддержку около 94 496 рублей.

На этапе реализации прототипа веб-приложения были разработаны 3 компонента архитектуры MVC. Полученные компоненты представляют собой основу будущего мобильного приложения.

На этапе контроля качества разработанного программного обеспечения было проведено тестирование. На основе полученных результатов можно сделать вывод о том, что веб-приложение для поиска требуемых медицинских препаратов работает корректно.

Разработка веб-приложения для поиска лекарственных препаратов проводилась на языке PHP, а так же языков HTML, CSS, JS. В качестве среды разработки был выбран продукт Microsoft, под названием Visual Studio Code.

В результате был получен прототип приложения «Энциклопедия лекарств», который является веб-приложением под любой браузер.

Дальнейшее развитие приложения имеет перспективы. Из-за наличия системы подбора дешёвых аналогов лекарственных препаратов и может составить должную конкуренцию на рынке веб-приложений.

					ПГУ 1.090501.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		57

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Трейси Б. Мастер времени / Б. Трейси – М.: Попурри, 2016. – 144 с.
2. Кюстенмахер В., Кюстенмахер М. Упростите свою жизнь. Мастер-класс по организации рабочего дня / Кюстенмахер В., Кюстенмахер М. – пер. с нем. – СПб.: Издательство Диля, 2006. – 128 с.
3. Глисон К. Работай меньше, успевай больше. Программа персональной эффективности / К. Глисон – М.: Манн, Иванов и Фербер (МИФ), 2012. – 352 с.
4. Выбор технологии для разработки приложения [Электронный ресурс] URL: <https://smartum.pro/ru/blog-ru/java-ili-php-kakoy-yazyik-vyibrat/> (дата обращения: 20.03.2020)
5. Буч Градди. Объектно-ориентированный анализ и проектирование с примерами приложений, 3-е изд. / Буч Градди, Максимчук Роберт А., Энгл Майкл У., Янг Бобби Дж., Коналлен Джим, Хьюстон Келли А. – Пер с англ. – М.: ООО «И.Д. Вильямс», 2010. – 720 с.
6. Шкляр, Леон. Архитектура веб-приложений: принципы, протоколы, практика / Леон Шкляр, Рич Розен. – пер. с англ. – М.: Эксмо, 2011. – 634 с.
7. Ульман Дж. и Уидом Дж. Реляционные базы данных / Дж.Ульман и Дж.Уидом – М.: Лори, 2014. – 384 с.
8. Тахагхогхи С., Вильямс Хью Е. Руководство по MySQL / Тахагхогхи С., Вильямс Хью Е – пер. с англ. – М.: «Русская редакция», 2007. – 544 с.
9. Bašić Mario. A Collection of Laravel Tutorials / Mario Bašić – МА.: «Leanpub», 2017. – 208 p.
10. Стефанов С. React.js. Быстрый старт / Стефанов С. – СПб.: Питер, 2017. – 304 с.

11. Сергей Рогачев. Обобщённый Model-View-Controller [Электронный ресурс] URL: <http://rsdn.ru/article/patterns/generic-mvc.xml> (дата обращения 17.04.2020)

12. Общие модели и классы фрейворка Laravel Illuminate\Database\Eloquent\Model [Электронный ресурс] URL: <https://laravel.com/api/5.6/Illuminate/Database/Eloquent/Model.html> (дата обращения: 12.03.2020)

13. Общие модели и классы фрейворка Laravel User [Электронный ресурс] URL: <https://laravel.com/api/5.6/Illuminate/Foundation/Auth/User.html> (дата обращения: 10.03.2019)

14. JSON [Электронный ресурс] URL: <https://ru.wikipedia.org/wiki/JSON> (дата обращения: 25.04.2019)

15. Общие модели и классы фрейворка Laravel Controller [Электронный ресурс] URL: <https://laravel.com/api/5.6/Illuminate/Routing/Controller.html> (дата обращения: 15.03.2019)

16. Общие модели и классы фрейворка Laravel Foundation/Auth/Access/AuthorizesRequests [Электронный ресурс] URL: <https://laravel.com/api/5.6/Illuminate/Foundation/Auth/Access/AuthorizesRequests.html> (дата обращения: 09.03.2019)

17. Общие модели и классы фрейворка Laravel Foundation/Validation/ValidatesRequests [Электронный ресурс] URL: <https://laravel.com/api/5.6/Illuminate/Foundation/Validation/ValidatesRequests.html> (дата обращения: 17.03.2019)

18. Общие модели и классы фрейворка Laravel Foundation/Bus/DispatchesJobs [Электронный ресурс] URL: <https://laravel.com/api/5.1/Illuminate/Foundation/Bus/DispatchesJobs.html> (дата обращения: 20.03.2019)

19. Раскин Д. Интерфейс: новые направления в проектировании компьютерных систем / Раскин Д. – пер. с англ. – СПб: «Символ-Плюс», 2007. – 272 с.

20. Куликов С. Тестирование программного обеспечения. Базовый курс / Куликов С. – Минск: «Четыре четверти», 2017. – 312 с.

21. Т.В. Зудилова, М.Л. Буркова Web-программирование HTML/редакционно-издательский отдел НИУ ИТМО/СПб 2012. – 5 с.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА ПРОЕКТ

ПРИЛОЖЕНИЕ А

(Обязательное)

					ПГУ 1.090501.02.001 ПЗ	Лист
						61
Изм.	Лист	№ докум.	Подпись	Дата		

1 Введение

Техническое задание распространяется на разработку программных и аппаратных средств входящих в состав системы по поиску требуемых медицинских препаратов.

2 Основание для разработки

Настоящий проект выполнен на основании приказа ректора университета № 0335а/о , от 30 апреля 2020 г.

3 Назначение для разработки

Система предназначена для быстрого и мультиплатформенного доступа к поиску требуемых лекарственных препаратов по поддерживаемым в актуальном состоянии базам данных.

4 Требование к функциональным характеристикам

Система должна обеспечивать выполнение следующих функций:

- регистрация новых администраторов в системе;
- авторизация администраторов по почте и паролю;
- управление и ведение БД, администраторами по средствам админ панели;
- добавление подробной информации о каждом лекарстве, включая: цену, графическое представление и инструкцию по применению;
- позволять осуществлять поиск требуемого лекарственного препарата тремя разными способами – по названию лекарства/по названию заболевания/по симптомам заболевания;
- вывод самого дешёвого аналогового лекарственного препарата;
- задавать фильтры поиска лекарственных препаратов.

5 Персонал для эксплуатации системы

Администраторы;

Программисты;

SMM-менеджеры;

Пользователи.

					ПГУ 1.090501.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		62

6 Требования к параметрам технических средств, информационная и программная совместимость

ОС система Windows 7, Ubuntu;

Apache + PHP 7;

СУБД сервера: MySQL;

					ПГУ 1.090501.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		63

ЛИСТИНГ ИСХОДНОГО КОДА
ПРИЛОЖЕНИЕ Б
(Обязательное)

					ПГУ 1.090501.02.001 ПЗ	Лист
						64
Изм.	Лист	№ докум.	Подпись	Дата		

Disease.php

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Disease extends Model

{

protected \$fillable = ['name', 'description'];

public \$timestamps = false;

public function treatments()

{

return \$this->hasMany('App\Treatment');

}

public function symptomsDiseases()

{

return \$this->hasMany('App\SymptomsDisease');

}

}

Medicament.php

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Medicament extends Model

{

protected \$fillable = ['name', 'description', 'instruction', 'amount', 'foto'];

public \$timestamps = false;

public function treatments()

{

return \$this->hasMany('App\Treatment');

}

}

Symptom.php

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Symptom extends Model

{

protected \$fillable = ['name'];

public \$timestamps = false;

public function symptomsDiseases()

{

return \$this->hasMany('App\SymptomsDisease');

}

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| | | | | | | 65 |
| Изм. | Лист | № докум. | Подпись | Дата | | |

```
}
```

SymptomsDisease.php

```
<?php
```

```
namespace App;
```

```
use Illuminate\Database\Eloquent\Model;
```

```
class SymptomsDisease extends Model
```

```
{  
    protected $fillable = ['disease_id', 'symptom_id'];  
    public $timestamps = false;
```

```
    public function disease()  
    {  
        return $this->belongsTo('App\Disease');  
    }
```

```
    public function symptom()  
    {  
        return $this->belongsTo('App\Symptom');  
    }  
}
```

Treatment.php

```
<?php
```

```
namespace App;
```

```
use Illuminate\Database\Eloquent\Model;
```

```
class Treatment extends Model
```

```
{  
    protected $fillable = ['medicament_id', 'disease_id'];  
    public $timestamps = false;
```

```
    public function medicament()  
    {  
        return $this->belongsTo('App\Medicament');  
    }
```

```
    public function disease()  
    {  
        return $this->belongsTo('App\Disease');  
    }  
}
```

User.php

```
<?php
```

```
namespace App;
```

```
use Illuminate\Contracts\Auth\MustVerifyEmail;  
use Illuminate\Foundation\Auth\User as Authenticatable;  
use Illuminate\Notifications\Notifiable;
```

```
class User extends Authenticatable
```

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 66 |

```

{
    use Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password',
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    /**
     * The attributes that should be cast to native types.
     *
     * @var array
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
}

```

Controller.php

<?php

```

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class Controller extends BaseController
{
    use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}

```

DiseaseController.php

<?php

```

namespace App\Http\Controllers;

use App\Disease;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;

class DiseaseController extends Controller
{
    private $rules;
}

```

```

public function __construct()
{
    $this->rules = [
        'name' => 'required|string|max:150',
        'description' => 'string',
    ];
}

/**
 * Display a listing of the resource.
 *
 * @return \Illuminate\Http\Response
 */
public function index()
{
    $page = 'diseases';
    $diseases = Disease::all();

    return view('admin.pages.diseases', compact('diseases', 'page'));
}

/**
 * Show the form for creating a new resource.
 *
 * @return \Illuminate\Http\Response
 */
public function create()
{
    return view('admin.forms.disease');
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validation = Validator::make($request->all(), $this->rules);

    if($validation->fails()){
        return back()->withErrors($validation);
    }else{
        $disease = new Disease();
        $disease->name = $request->name;
        $disease->description = nl2br($request->description);

        $disease->save();
    }

    return redirect()->route('diseases.index')->with('success', '');
}

/**
 * Display the specified resource.
 *
 * @param \App\Disease $disease
 * @return \Illuminate\Http\Response
 */
public function show(Disease $disease)

```

```

{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\Disease $disease
 * @return \Illuminate\Http\Response
 */
public function edit(Disease $disease)
{
    $disease->description = str_replace("<br />", " ", $disease->description);

    return view('admin.forms.disease', compact('disease'));
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param \App\Disease $disease
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, Disease $disease)
{
    $validation = Validator::make($request->all(), $this->rules);

    if($validation->fails()){
        return back()->withErrors($validation);
    }else{
        $disease->name = $request->name;
        $disease->description = nl2br($request->description);

        $disease->save();
    }

    return redirect()->route('diseases.index')->with('success', "");
}

/**
 * Remove the specified resource from storage.
 *
 * @param \App\Disease $disease
 * @return \Illuminate\Http\Response
 */
public function destroy(Disease $disease)
{
    $disease->delete();

    return redirect()->route('diseases.index')->with('success', "");
}
}

```

HomeController.php

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 69 |

```

class HomeController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('auth');
    }

    /**
     * Show the application dashboard.
     *
     * @return \Illuminate\Contracts\Support\Renderable
     */
    public function index()
    {
        return view('home', ['page' => '']);
    }
}

```

MedicamentController.php

```
<?php
```

```
namespace App\Http\Controllers;
```

```

use App\Medicament;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;
use Storage;

```

```
class MedicamentController extends Controller
```

```

{
    private $rules;

    public function __construct()
    {
        $this->rules = [
            'name' => 'required|string|max:150',
            'image' => 'required|file|mimes:jpg,png,jpeg',
            'instruction' => 'string',
            'amount' => 'required|numeric',
        ];
    }

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $page = 'medicaments';
        $medicaments = Medicament::all();

        return view('admin.pages.medicaments', compact('medicaments', 'page'));
    }
}

```

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 70 |

```

/**
 * Show the form for creating a new resource.
 *
 * @return \Illuminate\Http\Response
 */
public function create()
{
    return view('admin.forms.medicament');
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validation = Validator::make($request->all(), $this->rules);

    if($validation->fails()){
        return back()->withErrors($validation);
    }else{
        $medicament = new Medicament();
        $medicament->name = $request->name;
        $medicament->description = nl2br($request->description);
        $medicament->instruction = nl2br($request->instruction);
        $medicament->amount = $request->amount;

        $medicament->save();

        $request->image->storeAs('public/images/medicaments', $medicament->id.'.png');
    }

    return redirect()->route('medicaments.index')->with('success', "");
}

/**
 * Display the specified resource.
 *
 * @param \App\Medicament $medicament
 * @return \Illuminate\Http\Response
 */
public function show(Medicament $medicament)
{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\Medicament $medicament
 * @return \Illuminate\Http\Response
 */
public function edit(Medicament $medicament)
{
    $medicament->description = str_replace("<br />", "", $medicament->description);
    $medicament->instruction = str_replace("<br />", "", $medicament->instruction);

    return view('admin.forms.medicament', compact('medicament'));
}

```

```

    }

    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param \App\Medicament $medicament
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, Medicament $medicament)
    {
        $validation = Validator::make($request->all(), $this->rules);

        if($validation->fails()){
            return back()->withErrors($validation);
        } else {
            $medicament->name = $request->name;
            $medicament->description = nl2br($request->description);
            $medicament->instruction = nl2br($request->instruction);
            $medicament->amount = $request->amount;

            if(Storage::exists("public/images/medicaments/".$medicament->id.".png")){
                Storage::delete("public/images/medicaments/".$medicament->id.".png");
            }

            $request->image->storeAs('public/images/medicaments', $medicament->id.".png");

            $medicament->save();
        }

        return redirect()->route('medicaments.index')->with('success', "");
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Medicament $medicament
     * @return \Illuminate\Http\Response
     */
    public function destroy(Medicament $medicament)
    {
        $medicament->delete();

        if(Storage::exists("public/images/medicaments/".$medicament->id.".png")){
            Storage::delete("public/images/medicaments/".$medicament->id.".png");
        }

        return redirect()->route('medicaments.index')->with('success', "");
    }
}

```

PagesController.php

<?php

namespace App\Http\Controllers;

use App\Symptom;
use App\Disease;
use App\Medicament;
use Illuminate\Http\Request;

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 72 |

```

use Illuminate\Support\Facades\Validator;

class PagesController extends Controller
{
    public function medicaments()
    {
        $medicaments = Medicament::all();

        return view('pages.medicaments', compact('medicaments'));
    }

    public function diseases()
    {
        $diseases = Disease::all();

        return view('pages.diseases', compact('diseases'));
    }

    public function medicament($medicament)
    {
        $medicament = Medicament::find($medicament);

        return view('pages.medicament', compact('medicament'));
    }

    public function disease($disease)
    {
        $disease = Disease::find($disease);

        return view('pages.disease', compact('disease'));
    }

    public function symptoms()
    {
        $symptoms = Symptom::all();

        return view('pages.symptoms', compact('symptoms'));
    }

    public function findDiseases(Request $request)
    {
        $validation = Validator::make($request->all(), [
            'symptom_ids' => 'required',
        ]);

        if($validation->fails()){
            return back()->withErrors($validation);
        }

        $diseases = [];
        $symptom_ids = [];

        foreach ($request->symptom_ids as $key => $value) {
            $symptom_ids[] = $value;
            $symptomsDiseases = Symptom::find($value)->symptomsDiseases;

            foreach ($symptomsDiseases as $key => $value) {
                $disease = $value->disease;

                if(!in_array($disease, $diseases)) {
                    $diseases[] = $disease;
                }
            }
        }
    }
}

```

```

        }
    }
}

$symptoms = Symptom::all();

return view('pages.symptoms', compact('symptoms', 'diseases', 'symptom_ids'));
}

public function findMedicament(Request $request)
{
    $validation = Validator::make($request->all(), [
        'name' => 'required',
    ]);

    if($validation->fails()){
        return back()->withErrors($validation);
    }

    $name = $request->name;
    $medicaments = Medicament::where('name', 'LIKE', '%'. $name . '%')->get();

    return view('pages.medicaments', compact('medicaments', 'name'));
}
}

```

SymptomController.php

```

<?php

namespace App\Http\Controllers;

use App\Symptom;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;

class SymptomController extends Controller
{
    private $rules;

    public function __construct()
    {
        $this->rules = [
            'name' => 'required|string|max:150',
        ];
    }

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $page = 'symptoms';
        $symptoms = Symptom::all();

        return view('admin.pages.symptoms', compact('symptoms', 'page'));
    }

    /**

```

```

* Show the form for creating a new resource.
*
* @return \Illuminate\Http\Response
*/
public function create()
{
    return view('admin.forms.symptom');
}

/**
* Store a newly created resource in storage.
*
* @param \Illuminate\Http\Request $request
* @return \Illuminate\Http\Response
*/
public function store(Request $request)
{
    $validation = Validator::make($request->all(), $this->rules);

    if($validation->fails()){
        return back()->withErrors($validation);
    }else{
        Symptom::create($request->all());
    }

    return redirect()->route('symptoms.index')->with('success', "");
}

/**
* Display the specified resource.
*
* @param \App\Symptom $symptom
* @return \Illuminate\Http\Response
*/
public function show(Symptom $symptom)
{
    //
}

/**
* Show the form for editing the specified resource.
*
* @param \App\Symptom $symptom
* @return \Illuminate\Http\Response
*/
public function edit(Symptom $symptom)
{
    return view('admin.forms.symptom', compact('symptom'));
}

/**
* Update the specified resource in storage.
*
* @param \Illuminate\Http\Request $request
* @param \App\Symptom $symptom
* @return \Illuminate\Http\Response
*/
public function update(Request $request, Symptom $symptom)
{
    $validation = Validator::make($request->all(), $this->rules);

```

```

        if($validation->fails()){
            return back()->withErrors($validation);
        }else{
            $symptom->update($request->all());
        }

        return redirect()->route('symptoms.index')->with('success', "");
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param \App\Symptom $symptom
     * @return \Illuminate\Http\Response
     */
    public function destroy(Symptom $symptom)
    {
        $symptom->delete();

        return redirect()->route('symptoms.index')->with('success', "");
    }
}

```

SymptomsDiseaseController.php

<?php

namespace App\Http\Controllers;

use App\Disease;
 use App\Symptom;
 use App\SymptomsDisease;
 use Illuminate\Http\Request;
 use Illuminate\Support\Facades\Validator;

class SymptomsDiseaseController extends Controller
 {

private \$rules;

public function __construct()

```

    {
        $this->rules = [
            'disease_id' => 'integer|exists:diseases,id',
            'symptom_id' => 'integer|exists:symptoms,id',
        ];
    }

```

/**

* Display a listing of the resource.

*

* @return \Illuminate\Http\Response

*/

public function index()

```

    {
        $page = 'symptomsDiseases';
        $diseases = Disease::all();
        $symptoms = Symptom::all();
        $symptomsDiseases = SymptomsDisease::all();
    }

```

```

        return view('admin.pages.symptomsDiseases', compact('diseases', 'symptoms', 'symptomsDiseases', 'page'));
    }
}

```

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| Изм. | Лист | № докум. | Подпись | Дата | | 76 |

```

/**
 * Show the form for creating a new resource.
 *
 * @return \Illuminate\Http\Response
 */
public function create()
{
    $diseases = Disease::all();
    $symptoms = Symptom::all();

    return view('admin.forms.symptomsDisease', compact('diseases', 'symptoms'));
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validation = Validator::make($request->all(), $this->rules);

    if($validation->fails()){
        return back()->withErrors($validation);
    }else{
        SymptomsDisease::create($request->all());
    }

    return redirect()->route('symptomsDiseases.index')->with('success', '');
}

/**
 * Display the specified resource.
 *
 * @param \App\SymptomsDisease $symptomsDisease
 * @return \Illuminate\Http\Response
 */
public function show(SymptomsDisease $symptomsDisease)
{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param \App\SymptomsDisease $symptomsDisease
 * @return \Illuminate\Http\Response
 */
public function edit(SymptomsDisease $symptomsDisease)
{
    $diseases = Disease::all();
    $symptoms = Symptom::all();

    return view('admin.forms.symptomsDisease', compact('diseases', 'symptoms', 'symptomsDisease'));
}

/**
 * Update the specified resource in storage.
 *

```

```

* @param \Illuminate\Http\Request $request
* @param \App\SymptomsDisease $symptomsDisease
* @return \Illuminate\Http\Response
*/
public function update(Request $request, SymptomsDisease $symptomsDisease)
{
    $validation = Validator::make($request->all(), $this->rules);

    if($validation->fails()){
        return back()->withErrors($validation);
    }else{
        $symptomsDisease->update($request->all());
    }

    return redirect()->route('symptomsDiseases.index')->with('success', "");
}

/**
 * Remove the specified resource from storage.
 *
 * @param \App\SymptomsDisease $symptomsDisease
 * @return \Illuminate\Http\Response
 */
public function destroy(SymptomsDisease $symptomsDisease)
{
    $symptomsDisease->delete();

    return redirect()->route('symptomsDiseases.index')->with('success', "");
}
}

```

RegisterController.php

```
<?php
```

```

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use App\User;
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;

class RegisterController extends Controller
{
    /**
     *
     *-----
     * Register Controller
     *-----
     *
     * This controller handles the registration of new users as well as their
     * validation and creation. By default this controller uses a trait to
     * provide this functionality without requiring any additional code.
     *
     */

    use RegistersUsers;

    /**
     * Where to redirect users after registration.
     */

```

```

*
* @var string
*/
protected $redirectTo = RouteServiceProvider::HOME;

/**
 * Create a new controller instance.
 *
 * @return void
 */
public function __construct()
{
    $this->middleware('guest');
}

/**
 * Get a validator for an incoming registration request.
 *
 * @param array $data
 * @return \Illuminate\Contracts\Validation\Validator
 */
protected function validator(array $data)
{
    return Validator::make($data, [
        'name' => ['required', 'string', 'max:255'],
        'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
        'password' => ['required', 'string', 'min:8', 'confirmed'],
    ]);
}

/**
 * Create a new user instance after a valid registration.
 *
 * @param array $data
 * @return \App\User
 */
protected function create(array $data)
{
    return User::create([
        'name' => $data['name'],
        'email' => $data['email'],
        'password' => Hash::make($data['password']),
    ]);
}
}

```

disease.blade.php

```

-----
@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-12">
            <div class="card">
                @if (isset($disease))
                    <div class="card-header">Изменение заболевания</div>
                @else
                    <div class="card-header">Создание заболевания</div>
                @endif
            </div>
        </div>
    </div>
</div>

```

```

<div class="card-primary">
    @if (isset($disease))
        <form method="POST" action="{{ route('diseases.update',
['disease' => $disease->id]) }}" role="form">
            <input name="_method" type="hidden"
value="PUT">
        @else
            <form method="POST" action="{{ route('diseases.store') }}"
role="form">
        @endif

        @csrf

        <div class="card-body">
            <div class="form-group">
                <label>Название заболевания</label>
                <input type="text" class="form-control
@error('name') is-invalid @enderror" name="name" placeholder="Введите название" value="{{ isset($disease) ?
$disease->name : '' }}">

                @error('name')
                    <span class="invalid-feedback"
role="alert">
                        <strong>{{ $message
}}</strong>
                    </span>
                @enderror
            </div>

            <div class="form-group">
                <label>Описание</label>
                <textarea class="form-control
@error('description') is-invalid @enderror" name="description" placeholder="Введите текст описания"
rows="3">{{ isset($disease) ? $disease->description : '' }}</textarea>

                @error('description')
                    <span class="invalid-feedback"
role="alert">
                        <strong>{{ $message
}}</strong>
                    </span>
                @enderror
            </div>
        </div>

        <div class="card-footer">
            <button type="submit" class="btn btn-primary btn-
block">Сохранить</button>
        </div>
    </form>
</div>
</div>
</div>
</div>
</div>
</div>
@endsection

medicament.blade.php
-----
@extends('layouts.app')

```

```

@section('content')
<div class="container">
  <div class="row justify-content-center">
    <div class="col-md-12">
      <div class="card">
        @if (isset($medicament))
          <div class="card-header">Изменение лекарства</div>
        @else
          <div class="card-header">Создание лекарства</div>
        @endif

        <div class="card-primary">
          @if (isset($medicament))
            <form method="POST" action="{{ route('medicaments.update', ['medicament' => $medicament->id]) }}" role="form" enctype="multipart/form-data">
              <input name="_method" type="hidden" value="PUT">
              @else
                <form method="POST" action="{{ route('medicaments.store') }}" role="form" enctype="multipart/form-data">
              @endif

              @csrf

              <div class="card-body">
                <div class="form-group">
                  <label>Название лекарства</label>
                  <input type="text" class="form-control"
                    @error('name') is-invalid @enderror" name="name" placeholder="Введите название" value="{{
                    isset($medicament) ? $medicament->name : '' }}"

                    @error('name')
                      <span class="invalid-feedback"
                        role="alert">
                          <strong>{{ $message }}
                        </strong>
                      </span>
                    @enderror
                  </div>

                  <div class="form-group">
                    <label>Описание</label>
                    <textarea class="form-control"
                      @error('description') is-invalid @enderror" name="description" placeholder="Введите текст описания"
                      rows="3">{{ isset($medicament) ? $medicament->description : '' }}</textarea>

                      @error('description')
                        <span class="invalid-feedback"
                          role="alert">
                            <strong>{{ $message }}
                          </strong>
                        </span>
                      @enderror
                    </div>

                    <div class="form-group">
                      <label>Инструкция</label>
                      <textarea class="form-control"
                        @error('instruction') is-invalid @enderror" name="instruction" placeholder="Введите текст инструкции"
                        rows="3">{{ isset($medicament) ? $medicament->instruction : '' }}</textarea>

```

```

@error('instruction')
    <span class="invalid-feedback"
role="alert">
        <strong>{{ $message
    }}</strong>
    </span>
@enderror
</div>

<div class="form-group">
    <label>Цена</label>
    <input type="text" class="form-control
@error('amount') is-invalid @enderror" name="amount" placeholder="Введите название" value="{{
isset($medicament) ? $medicament->amount : "" }}">

@error('amount')
    <span class="invalid-feedback"
role="alert">
        <strong>{{ $message
    }}</strong>
    </span>
@enderror
</div>

<div class="form-group">
    <label>Загрузить фото лекарства</label>
    <input name="image" type="file"
class="form-control-file @error('image') is-invalid @enderror">

@error('image')
    <span class="invalid-feedback"
role="alert">
        <strong>{{ $message
    }}</strong>
    </span>
@enderror
</div>
</div>

<div class="card-footer">
    <button type="submit" class="btn btn-primary btn-
block">Сохранить</button>
</div>
</form>
</div>
</div>
</div>
</div>
@endsection

symptom.blade.php
-----
@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-12">
            <div class="card">
                @if (isset($symptom))

```



```

<div class="card-primary">
    @if (isset($symptomsDisease))
        <form method="POST" action="{{
route('symptomsDiseases.update', ['symptomsDisease' => $symptomsDisease->id]) }}" role="form">
            <input name="_method" type="hidden"
value="PUT">

            @else
                <form method="POST" action="{{
route('symptomsDiseases.store') }}" role="form">
            @endif

            @csrf

            <div class="card-body">
                <div class="form-group">
                    <label>Выберите заболевание</label>
                    <select name="disease_id" class="selectpicker form-control @error('disease_id') is-invalid
@enderror" data-show-subtext="true" data-live-search="true">
                        @foreach ($diseases as $disease)
                            <option {{ (isset($symptomsDisease) && ($symptomsDisease->disease_id == $disease-
>id)) ? 'selected' : '' }} value="{{ $disease->id }}">{{ $disease->name }}</option>
                        @endforeach
                    </select>

                    @error('disease_id')
                        <span class="invalid-feedback" role="alert">
                            <strong>{{ $message }}</strong>
                        </span>
                    @enderror
                </div>

                <div class="form-group">
                    <label>Выберите симптом</label>
                    <select name="symptom_id" class="selectpicker form-control @error('symptom_id') is-invalid
@enderror">
                        @foreach ($symptoms as $symptom)
                            <option {{ (isset($symptomsDisease) && ($symptomsDisease->symptom_id ==
$symptom->id)) ? 'selected' : '' }} value="{{ $symptom->id }}">{{ $symptom->name }}</option>
                        @endforeach
                    </select>

                    @error('symptom_id')
                        <span class="invalid-feedback" role="alert">
                            <strong>{{ $message }}</strong>
                        </span>
                    @enderror
                </div>
            </div>

            <div class="card-footer">
                <button type="submit" class="btn btn-primary btn-
block">Сохранить</button>
            </div>
        </form>
    </div>
</div>
</div>
</div>
@endsection

```

treatment.blade.php

```
@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-12">
            <div class="card">
                @if (isset($treatment))
                    <div class="card-header">Изменение лечения против заболевания</div>
                @else
                    <div class="card-header">Создание лечения против заболевания</div>
                @endif

                <div class="card-primary">
                    @if (isset($treatment))
                        <form method="POST" action="{{ route('treatments.update',
['treatment' => $treatment->id]) }}" role="form">
                            <input name="_method" type="hidden"
value="PUT">
                            @else
                                <form method="POST" action="{{ route('treatments.store')
}}" role="form">
                                    @csrf

                                    <div class="card-body">
                                        <div class="form-group">
                                            <label>Выберите заболевание</label>
                                            <select name="disease_id" class="selectpicker form-control @error('disease_id') is-invalid
@enderror" data-show-subtext="true" data-live-search="true">
                                                @foreach ($diseases as $disease)
                                                    <option {{ (isset($treatment) && ($treatment->disease_id == $disease->id)) ? 'selected' : "
}} value="{{ $disease->id }}">{{ $disease->name }}</option>
                                                @endforeach
                                            </select>

                                            @error('disease_id')
                                                <span class="invalid-feedback" role="alert">
                                                    <strong>{{ $message }}</strong>
                                                </span>
                                            @enderror
                                        </div>

                                        <div class="form-group">
                                            <label>Выберите симптом</label>
                                            <select name="medicament_id" class="selectpicker form-control @error('medicament_id') is-
invalid @enderror">
                                                @foreach ($medicaments as $medicament)
                                                    <option {{ (isset($treatment) && ($treatment->medicament_id == $medicament->id)) ?
'selected' : " }} value="{{ $medicament->id }}">{{ $medicament->name }}</option>
                                                @endforeach
                                            </select>

                                            @error('medicament_id')
                                                <span class="invalid-feedback" role="alert">
                                                    <strong>{{ $message }}</strong>
                                                </span>
                                            @enderror
                                        </div>
                                    </div>
                                </form>
                            @endif
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>

```



```

<div class="row justify-content-center">
  <div class="col-md-12">
    <div class="card">
      <div class="card-header">Симптомы</div>

      <div class="col-lg-2 mt-2">
        <a class="btn btn-primary" href="{{ route('symptoms.create') }}" role="button">Создать</a>
      </div>

      <div class="card-body">
        <div id="example2_wrapper" class="dataTables_wrapper dt-bootstrap4">
          <div class="row">
            <div class="col-sm-12 col-md-6"></div>
            <div class="col-sm-12 col-md-6"></div>
          </div>
          <div class="row">
            <div class="col-sm-12">
              <table id="data-table" class="table table-bordered table-hover dataTable" role="grid" aria-
describedby="example2_info">
                <thead>
                  <tr role="row">
                    <th>ID</th>
                    <th>Название</th>
                    <th>Действия</th>
                  </tr>
                </thead>
                <tbody>
                  @foreach ($symptoms as $symptom)
                    <tr>
                      <td>{{ $symptom->id }}</td>
                      <td>{{ $symptom->name }}</td>
                      <td>
                        <a class="btn btn-info btn-sm" href="{{ route('symptoms.edit', ['symptom' =>
$symptom->id]) }}">
                          <i class="fas fa-pencil-alt"></i>
                          Изменить
                        </a>
                        <a class="btn btn-danger btn-sm" href="{{ route('symptoms.destroy', ['symptom'
=> $symptom->id]) }}" onclick="event.preventDefault(); document.getElementById('symptom-{{ $symptom->id
}}-destroy-form').submit();">
                          <i class="fas fa-trash"></i>
                          Удалить
                        </a>

                        <form id="symptom-{{ $symptom->id }}-destroy-form" action="{{
route('symptoms.destroy', ['symptom' => $symptom->id]) }}" method="POST" style="display: none;">
                          @csrf
                          <input type="hidden" name="_method" value="DELETE">
                        </form>
                      </td>
                    </tr>
                  @endforeach
                </tbody>
              </table>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

```

```

</div>
</div>
@endsection
symptomsDiseases.blade.php
-----
@extends('layouts.app')

@section('content')
<div class="container">
    @include('partials.nav')
    @include('partials.message')

    <div class="row justify-content-center">
        <div class="col-md-12">
            <div class="card">
                <div class="card-header">Симптомы заболеваний</div>

                <div class="col-lg-2 mt-2">
                    <a class="btn btn-primary" href="{{ route('symptomsDiseases.create') }}" role="button">Создать</a>
                </div>

                <div class="card-body">
                    <div id="example2_wrapper" class="dataTables_wrapper dt-bootstrap4">
                        <div class="row">
                            <div class="col-sm-12 col-md-6"></div>
                            <div class="col-sm-12 col-md-6"></div>
                        </div>
                        <div class="row">
                            <div class="col-sm-12">
                                <table id="data-table" class="table table-bordered table-hover dataTable" role="grid" aria-
describedby="example2_info">
                                    <thead>
                                        <tr role="row">
                                            <th>ID</th>
                                            <th>Название заболевания</th>
                                            <th>Симптом</th>
                                            <th>Действия</th>
                                        </tr>
                                    </thead>
                                    <tbody>
                                        @foreach ($symptomsDiseases as $symptomsDisease)
                                            <tr>
                                                <td>{{ $symptomsDisease->id }}</td>
                                                <td>{{ $symptomsDisease->disease->name }}</td>
                                                <td>{{ $symptomsDisease->symptom->name }}</td>
                                                <td>
                                                    <a class="btn btn-info btn-sm" href="{{ route('symptomsDiseases.edit',
['symptomsDisease' => $symptomsDisease->id]) }}">
                                                        <i class="fas fa-pencil-alt"></i>
                                                        Изменить
                                                    </a>
                                                    <a class="btn btn-danger btn-sm" href="{{ route('symptomsDiseases.destroy',
['symptomsDisease' => $symptomsDisease->id]) }}" onclick="event.preventDefault();
document.getElementById('symptomsDisease-{{ $symptomsDisease->id }}-destroy-form').submit();">
                                                        <i class="fas fa-trash"></i>
                                                        Удалить
                                                    </a>

                                                    <form id="symptomsDisease-{{ $symptomsDisease->id }}-destroy-form"
action="{{ route('symptomsDiseases.destroy', ['symptomsDisease' => $symptomsDisease->id]) }}"
method="POST" style="display: none;">

```



```

</a>
@endforeach

</div>
<div class="row mt-2">
  <h2>Описание</h2>
</div>
<div class="row">
  <p>{!! $disease->description !!}</p>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
@endsection

diseases.blade.php
-----
@extends('layouts.pages')

@section('content')
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-md-12">
        <div class="card">
          <div class="card-header">Поиск по заболеваниям</div>

          <div class="card-body">
            <div id="example2_wrapper" class="dataTables_wrapper dt-bootstrap4">
              <div class="row">
                <div class="col-sm-12 col-md-6"></div>
                <div class="col-sm-12 col-md-6"></div>
              </div>
              <div class="row">
                <div class="col-sm-12">
                  <table id="data-table" class="table table-bordered table-hover dataTable" role="grid" aria-
describedby="example2_info">
                    <thead>
                      <tr role="row">
                        <th>Название</th>
                        <th>Инструкция</th>
                        <th>Лекарства</th>
                        <th>Действия</th>
                      </tr>
                    </thead>
                    <tbody>
                      @foreach ($diseases as $disease)
                        <tr>
                          <td>{{ $disease->name }}</td>
                          <td>{!! (strlen($disease->description) > 1500) ? substr($disease->description, 0,
1500) : $disease->description !!}</td>
                          <td>
                            @foreach ($disease->treatments as $treatment)

                              <a href="{{ { route('medicament', ['medicament' => $treatment->medicament->id]) } }}">
                                {{ $treatment->medicament->name }}
                              </a>
                            @endforeach
                        </td>
                      </tr>
                    </tbody>
                  </table>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>

```



```

</div>
</div>
</div>
</div>
</div>
</div>
@endsection

medicaments.blade.php
-----
@extends('layouts.pages')

@section('content')
    <div class="container" style="margin-bottom: 150px;">
        <div class="row justify-content-center">
            <div class="col-md-10">
                <div class="card">
                    <div class="card-header" style="cursor: pointer;" onclick="location.href = '{{ route('medicaments') }}';">Поиск по лекарствам</div>

                    <div class="card-body">

                        <form method="POST" action="{{ route('findMedicament') }}" role="form">
                            @csrf

                            <div class="form-group">
                                <label>Введите название лекарства</label>
                                <input type="text" class="form-control @error('name') is-invalid @enderror" name="name" placeholder="Введите название" value="{{ (isset($name)) ? $name : '' }}">

                                @error('name')
                                    <span class="invalid-feedback"
                                        role="alert">
                                            <strong>{{ $message }}</strong>
                                        </span>
                                @enderror

                            </div>

                            <button type="submit" class="btn btn-primary btn-block">Поиск</button>
                        </form>

                        @if(count($medicaments) == 0)
                            <div class="row justify-content-center mt-5">
                                <p>Данные по запросу не найдены</p>
                            </div>
                        @else
                            @foreach ($medicaments as $medicament)
                                <div class="row border-top mt-4">
                                    <div class="col-md-3">
                                        name }}" style="width: 200px; height: 200px;">
                                    </div>
                                    <div class="col-md-9">
                                        <a href="{{ route('medicament', ['medicament' => $medicament->id]) }}">
                                            <h2>{{ $medicament->name }}</h2>
                                        </a>
                                        <p>{{ str_replace("<br />", "", $medicament->description) }}</p>
                                    </div>
                                </div>
                            </foreach>
                        </if>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

        </p>
        <p style="font-size: 18px;">
            Цена <span style="font-weight: bold;">{{ $medicament->amount }}</span> руб.
        </p>
    </div>
</div>
@endforeach
@endif
</div>
</div>
</div>
</div>
</div>
</div>
@endsection

symptoms.blade.php
-----
@extends('layouts.pages')

@section('content')
    <div class="container" style="margin-bottom: 150px">
        <div class="row justify-content-center">
            <div class="col-md-12">
                <div class="card">
                    <div class="card-header">Поиск по симптомам</div>

                    <div class="card-body">
                        <form method="POST" action="{{ route('findDiseases') }}"
role="form">

                            @csrf

                            <div class="form-group">
                                <label>Выберите симптомы</label>
                                <select name="symptom_ids[]" class="selectpicker form-control @error('symptom_id') is-
invalid @enderror" multiple="multiple">
                                    @foreach ($symptoms as $symptom)
                                        <option {{ (isset($diseases) && (in_array($symptom->id, $symptom_ids))) ? 'selected' : '' }}
value="{{ $symptom->id }}">{{ $symptom->name }}</option>
                                    @endforeach
                                </select>

                                @error('symptom_id')
                                    <span class="invalid-feedback" role="alert">
                                        <strong>{{ $message }}</strong>
                                    </span>
                                @enderror
                            </div>

                            <button type="submit" class="btn btn-primary btn-block">Поиск</button>
                        </form>

                        @if (isset($diseases))
                            <div id="example2_wrapper" class="dataTables_wrapper dt-bootstrap4">
                                <div class="row mt-4">
                                    <div class="col-sm-12">
                                        <table id="data-table" class="table table-bordered table-hover dataTable" role="grid" aria-
describedby="example2_info">
                                            <thead>
                                                <tr role="row">
                                                    <th>Название</th>
                                                    <th>Описания</th>
                                                </tr>
                                            </thead>
                                        </table>
                                    </div>
                                </div>
                            </div>
                        </if>
                    </div>
                </div>
            </div>
        </div>
    </div>
</section>
</div>

```



```

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="{{
__('Toggle navigation') }}">
        <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <!-- Left Side Of Navbar -->
        <ul class="navbar-nav mr-auto">

            </ul>

        <!-- Right Side Of Navbar -->
        <ul class="navbar-nav ml-auto">
            <!-- Authentication Links -->
            @guest
                <li class="nav-item">
                    <a class="nav-link" href="{{ route('login') }}">{{ __('Login') }}</a>
                </li>
                @if(Route::has('register'))
                    <li class="nav-item">
                        <a class="nav-link" href="{{ route('register') }}">{{ __('Register') }}</a>
                    </li>
                @endif
            @else
                <li class="nav-item dropdown">
                    <a id="navbarDropdownRight" class="nav-link dropdown-toggle" href="#" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>
                        {{ Auth::user()->name }} <span class="caret"></span>
                    </a>

                    <div class="dropdown-menu dropdown-menu-right" aria-labelledby="navbarDropdownRight">
                        <a class="dropdown-item" href="{{ route('logout') }}"
                            onclick="event.preventDefault(); document.getElementById('logout-form').submit();">
                            {{ __('Logout') }}
                        </a>

                        <form id="logout-form" action="{{ route('logout') }}" method="POST" style="display:
none;">

                            @csrf
                        </form>
                    </div>
                </li>
            @endguest
        </ul>
    </div>
</div>
</nav>

<main class="py-4">
    @yield('content')
</main>
</div>

<script type="text/javascript" src="{{ asset('js/jquery-3.3.1.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/bootstrap.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/bootstrap.bundle.min.js') }}"></script>

<script type="text/javascript" src="{{ asset('js/select2.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/select2.ru.js') }}"></script>

```

```

<script type="text/javascript" src="{{ asset('js/jquery.dataTables.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/dataTables.bootstrap4.min.js') }}"></script>
<script>
    $(document).ready(function() {
        $('#data-table').DataTable({
            "dom":
                "<row'<col-sm-12 col-md-6'l><col-sm-12 col-md-6'f>>" +
                "<row'<col-sm-12'tr>>" +
                "<row'<col-md-12'p>>",
            "language": {
                "emptyTable": "Таблица пуста",
                "paginate": {
                    "previous": "<",
                    "next": ">",
                },
                "search": "{{ __('Search') }}",
                "zeroRecords": "Таблица пуста",
                "lengthMenu": "Количество элементов _MENU_",
            },
        });
    });
</script>
</body>
</html>

```

pages.blade.php

```

<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- CSRF Token -->
    <meta name="csrf-token" content="{{ csrf_token() }}">

    <title>{{ config('app.name', 'Laravel') }}</title>

    <!-- Fonts -->
    <link rel="dns-prefetch" href="//fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css?family=Nunito" rel="stylesheet">

    <!-- Styles -->
    <link rel="stylesheet" type="text/css" href="{{ asset('css/dataTables.bootstrap4.css') }}" />
    <link rel="stylesheet" type="text/css" href="{{ asset('css/select2.min.css') }}" />
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
</head>
<body>
    <div class="row mb-5" style="background-color: #2b6622; height: 200px;">
        <div class="col-md-2">
            <div class="row justify-content-center mt-4">
                
            </div>
        </div>
        <div class="col-md-8 mt-3">
            <div class="row justify-content-center" style="color: white; cursor: pointer;" onclick="location.href = '{{ route('main') }}'">
                <span style="font-size: 50px; text-transform: uppercase">Энциклопедия лекарств</span>
            </div>
        </div>
    </div>

```

```

    </div>
    <div class="row justify-content-center" style="color: white">
      <span style="font-size: 19px; text-transform: uppercase">Ваш домашний справочник по всем
препаратам</span>
    </div>
  </div>
  <div class="col-md-2 mt-3" onclick="location.href = '{{ route('login') }}';" style="cursor: pointer;">
    <div class="row justify-content-center" style="color: white">
      
    </div>
    <div class="row justify-content-center" style="color: white">
      <span style="font-size: 12px; text-transform: uppercase">Администрирование</span>
    </div>
  </div>
</div>

@yield('content')

<script type="text/javascript" src="{{ asset('js/jquery-3.3.1.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/bootstrap.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/bootstrap.bundle.min.js') }}"></script>

<script type="text/javascript" src="{{ asset('js/select2.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/select2.ru.js') }}"></script>

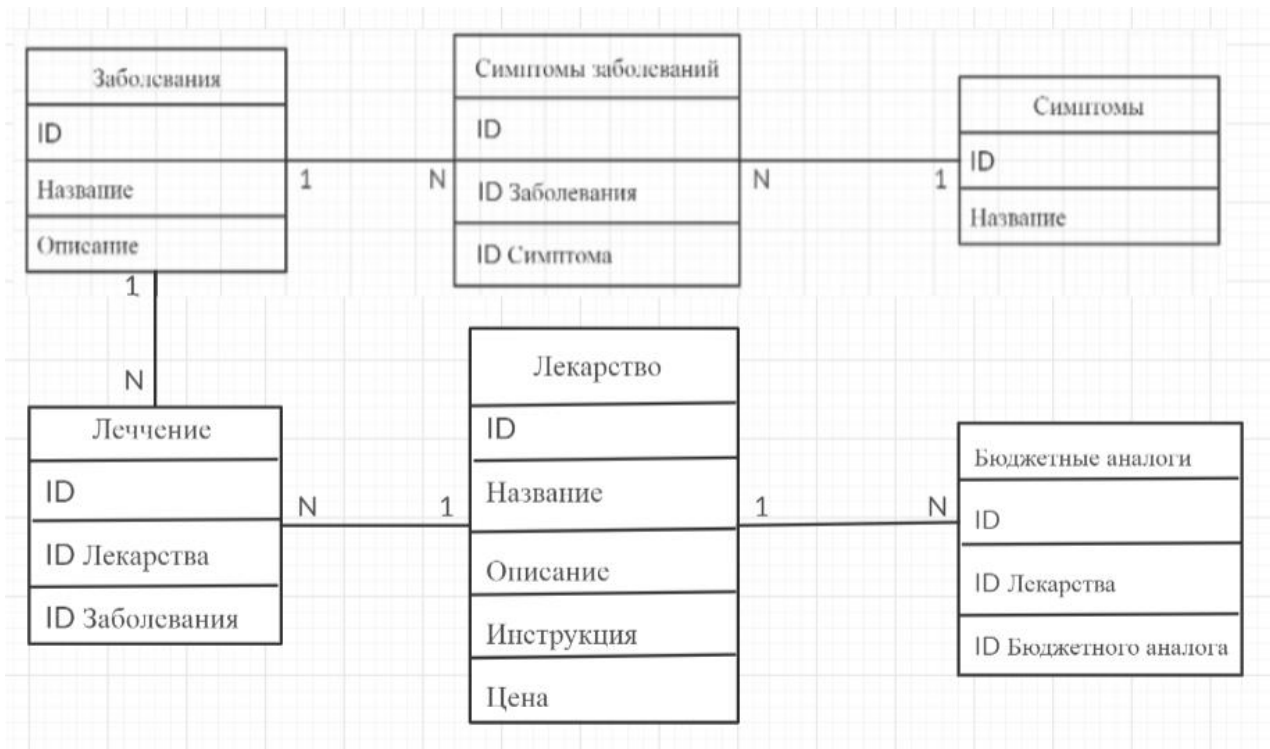
<script type="text/javascript" src="{{ asset('js/jquery.dataTables.min.js') }}"></script>
<script type="text/javascript" src="{{ asset('js/dataTables.bootstrap4.min.js') }}"></script>
<script>
$(document).ready(function() {
  $('#selectpicker').select2();

  $('#data-table').DataTable({
    "dom":
      "<row'<col-sm-12 col-md-6'l><col-sm-12 col-md-6'f>>" +
      "<row'<col-sm-12'tr>>" +
      "<row'<col-md-12'p>>",
    "language": {
      "emptyTable": "Таблица пуста",
      "paginate": {
        "previous": "<",
        "next": ">",
      },
    },
    "search": "{{ __('Search') }}",
    "zeroRecords": "Таблица пуста",
    "lengthMenu": "Количество элементов _MENU_",
  },
  });
});
</script>
</body>
</html>

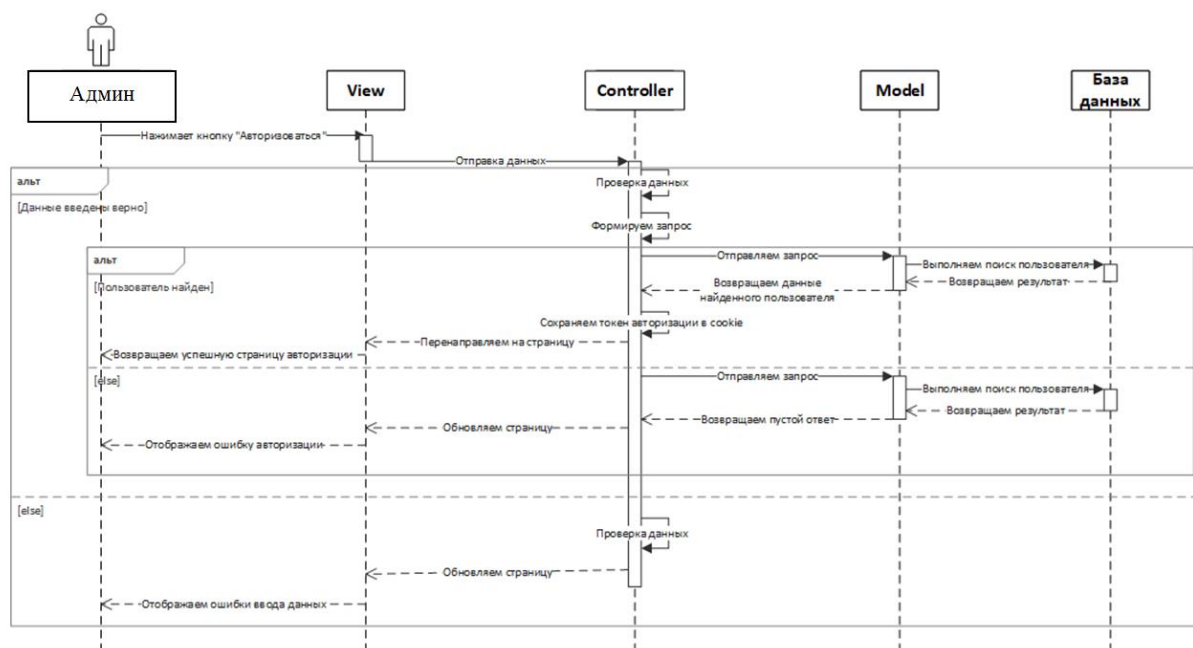
```

ДИАГРАММЫ
ПРИЛОЖЕНИЕ В

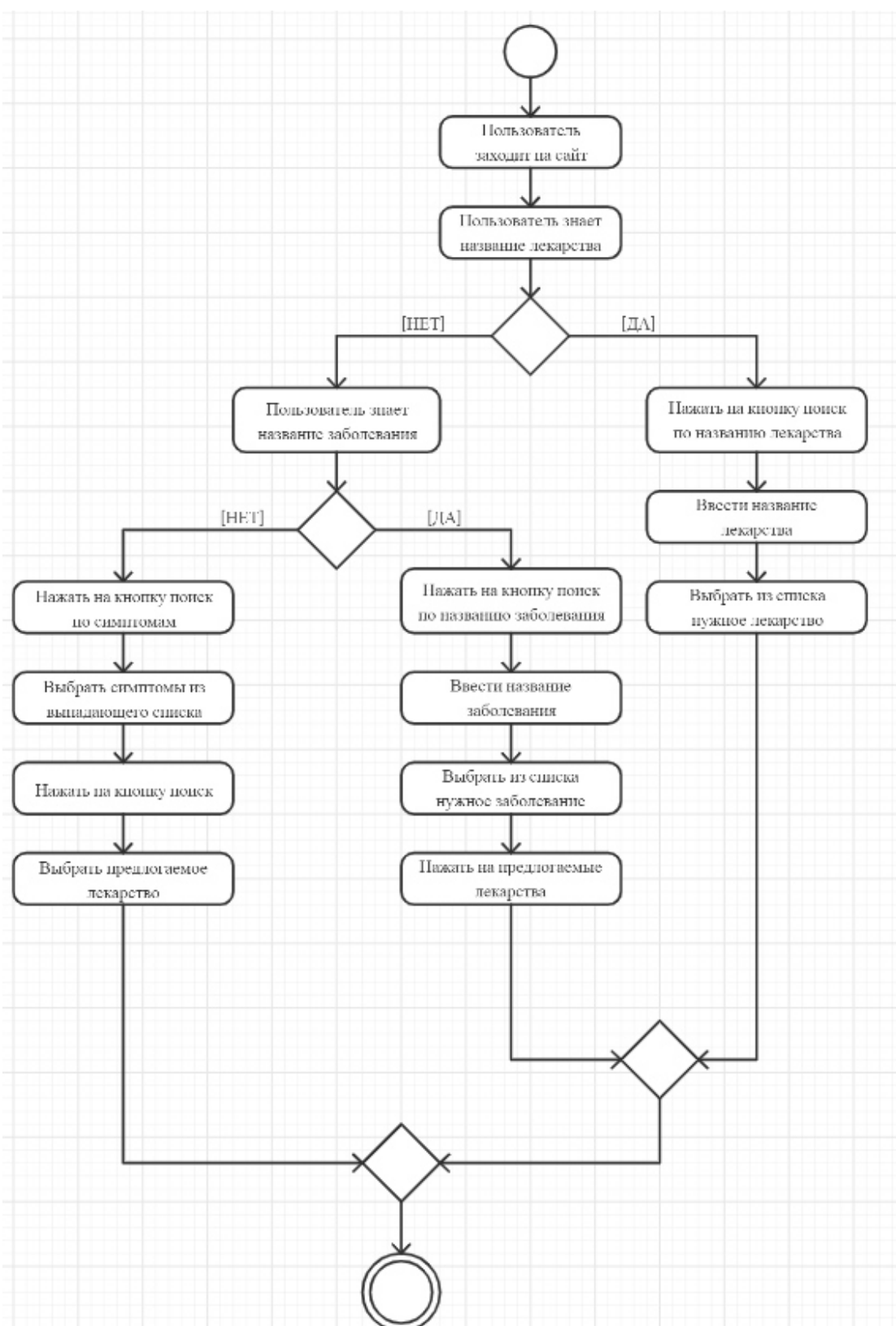
| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| | | | | | | 101 |
| Изм. | Лист | № докум. | Подпись | Дата | | |



| | | | | | | | |
|-----------|--------------|----------|---------|------|---|---------------------|------|
| | | | | | ПГУ 1.090501.02.002 ПЗ | | |
| Изм. | Лист | № докум. | Подпись | Дата | | | |
| Разраб. | Борунов В.И. | | | | Веб-приложение по
персональному поиску требуемых
медицинских препаратов
Диаграмма сущность-связь | Лит. | Лист |
| Провер. | Гурин Е.И. | | | | | | 102 |
| Реценз. | | | | | | Листов | 134 |
| Н. Контр. | Кучин А.В. | | | | | Кафедра ВТ Гр.15ВО2 | |
| Утверд. | | | | | | | |



| | | | | | | | |
|-----------|--------------|----------|---------|------|---|---------------------|--------|
| | | | | | ПГУ 1.090501.02.004 ПЗ | | |
| Изм. | Лист | № докум. | Подпись | Дата | | | |
| Разраб. | Борунов В.И. | | | | Веб-приложение по персональному
поиску требуемых
медицинских препаратов
Диаграмма последовательности | Лит. | Лист |
| Провер. | Гурин Е.И. | | | | | | Листов |
| Реценз. | | | | | | 104 | 134 |
| Н. Контр. | Кучин А.В. | | | | | Кафедра ВТ Гр.15ВО2 | |
| Утверд. | | | | | | | |



| | | | | | | | |
|-----------|--------------|----------|---------|------|---|--|--|
| | | | | | ПГУ 1.090501.02.005 ПЗ | | |
| Изм. | Лист | № докум. | Подпись | Дата | Веб-приложение по персональному
поиску требуемых
медицинских препаратов
Диаграмма деятельности | | |
| Разраб. | Борунов В.И. | | | | | | |
| Провер. | Гурин Е.И. | | | | | | |
| Реценз. | | | | | | | |
| Н. Контр. | Кучин А.В. | | | | | | |
| Утверд. | | | | | Лит. Лист Листов
105 134
Кафедра ВТ Гр.15В02 | | |

ГРАФИЧЕСКИЕ РЕЗУЛЬТАТЫ ТЕСТОВЫХ ИСПЫТАНИЙ
ПРИЛОЖЕНИЕ Г

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| | | | | | | 106 |
| Изм. | Лист | № докум. | Подпись | Дата | | |

1. Запуск веб-приложения



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

ПОИСК

 ПО НАЗВАНИЮ ЛЕКАРСТВА

 ПО НАЗВАНИЮ ЗАБОЛЕВАНИЯ

 ПО СИМПТОМАМ

2. Регистрация нового администратора

Энциклопедия лекарств

[Войти](#) [Регистрация](#)

Регистрация

Name

Е-Mail адрес

Пароль

Подтверждение пароля

Регистрация

Регистрация

Name

Vacheslav

E-Mail адрес

admin@mail.ru

Пароль

.....

Подтверждение пароля

.....

Регистрация

Заболевания

Лекарства

Симптомы

Симптомы болезней

Лечение

Бюджетные аналоги

Добро пожаловать в Админ-панель, Vacheslav

Данный раздел предназначен для добавления, редактирования, изменения записей в приложении.

3. Ввод не правильного логина и пороля при входе администратора

Войти

E-Mail адрес

admin@mail.ru

Пароль

.....

☐ Запомнить меня

Войти

Войти

Е-Mail адрес

admin@mail.ru

Неверно введен логин или пароль.

Пароль

☐ Запомнить меня

Войти

4. Ввод правильного логина и пароля при входе администратора

Войти

Е-Mail адрес

admin@mail.ru

Неверно введен логин или пароль.

Пароль

.....

☐ Запомнить меня

Войти

Заболевания

Лекарства

Симптомы

Симптомы болезней

Лечение

Бюджетные аналоги

Добро пожаловать в Админ-панель, Vacheslav

Данный раздел предназначен для добавления, редактирования, изменения записей в приложении.

5. Добавление нового лекарственного препарата

Энциклопедия лекарств

Vacheslav ▾

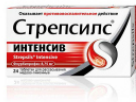
[Заболевания](#) [Лекарства](#) [Симптомы](#) [Симптомы болезней](#) [Лечение](#) [Бюджетные аналоги](#)

Лекарства

Создать

Количество элементов 10 ▾

Поиск

| ID ▴▾ | Картинка ▴▾ | Название ▴▾ | Описание ▴▾ | Инструкция ▴▾ | Цена ▴▾ | Действия ▴▾ |
|-------|--|-----------------------|---|--|---------|---|
| 1 |  | Стрепсилс
Интенсив | Таблетки
медово-
лимонные, 24
шт. | Инструкция по применению
Производитель Состав Фармакологическое действие
Показания Противопоказания Побочные действия
Взаимодействие Как принимать, курс приема и
дозировка Передозировка Специальные указания ⚡ | 306 | Изменить
Удалить |
| 2 |  | Кларитин | Сироп, 60 мл | Инструкция по применению
Производитель Состав Фармакологическое действие
Показания Применение при беременности и кормлении
грудью Противопоказания Побочные действия
Взаимодействие Как принимать, курс приема и | 262 | Изменить
Удалить |
| 3 |  | Отривин | Спрей
назальный
дозированный
0.1%, 10 мл | Инструкция по применению
Производитель Кратко о товаре Состав
Фармакологическое действие Клиническая
фармакология Показания Применение при
беременности и кормлении грудью Противопоказания
Побочные действия ⚡ | 177 | Изменить
Удалить |

Энциклопедия лекарств

Vacheslav ▾

Создание лекарства

Название лекарства

Описание

Инструкция

Цена

Загрузить фото лекарства

| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

110

Создание лекарства

Название лекарства

Келтикан комплекс

Описание

Капсулы 20 шт

Инструкция

для взрослых
Штрих-код и вес
Штрих-код: 5901234123457
Вес: 0.017 кг;

Цена

750

Загрузить фото лекарства

Выберите файл 3.jpg

Сохранить

| | | | | | | |
|---|---|----------------------|---------------------------------|---|-----|--|
| |  | | приема
внутри, 16 г, 6
шт | Производитель Состав Фармакологическое действие
Показания Противопоказания Побочные действия
Взаимодействие Как принимать, курс приема и
дозировка Передозировка Специальные указания | | Удалить |
| 6 |  | Келтикан
комплекс | Капсулы 20
шт | Инструкция по применению
Производитель Кратко о товаре Состав
Фармакологическое действие Показания
Противопоказания Как принимать, курс приема и
дозировка Специальные указания Назначение
Производитель | 750 | <div>Изменить</div> <div>Удалить</div> |

< 1 >

6.Добавление нового заболевания

Заболевания Лекарства Симптомы Симптомы болезней Лечение Бюджетные аналоги

Заболевания

Создать

Количество элементов 10

Поиск

| ID ↑↓ | Название ↑↓ | Описание ↑↓ | Действия ↑↓ |
|-------|-------------|---|--|
| 1 | Ангина | Острый тонзиллит (от лат. tonsillae — миндалины), в обиходе — анги́на (от лат. angio — «сжимаю, сдавливаю, душу»), — инфекционное заболевание с местными проявлениями в виде острого воспаления компонентов лимфатического глоточного кольца, чаще всего нёбных миндалин, вызываемое стрептококками или стафилококками, реже другими микроорганизмами, вирусами и грибами (см. подробнее Этиология[=])[3]. Ангиной также называют обострение хронического тонзиллита. | <div>Изменить</div> <div>Удалить</div> |
| 2 | Аллергия | Аллергия (др.-греч. ἀλλος — другой, иной, чужой + ἔργον — воздействие) — типовой иммунопатологический | <div>Изменить</div> |

Создание заболевания

Название заболевания

Диабет

Описание

Сахарный диабет — эндокринное заболевание, обусловленное недостатком в организме гормона инсулина или его низкой биологической активностью. Характеризуется нарушением всех видов обмена веществ, поражением крупных и мелких кровеносных сосудов и проявляется гипергликемией. Первым, кто дал название болезни – «диабет» был врач Аретиус, живший в Риме во втором веке н. э. Намного позже уже в 1776 году врач Добсон

Сохранить

| | | | |
|---|--------|---|--|
| 6 | Диабет | Сахарный диабет — эндокринное заболевание, обусловленное недостатком в организме гормона инсулина или его низкой биологической активностью. Характеризуется нарушением всех видов обмена веществ, поражением крупных и мелких кровеносных сосудов и проявляется гипергликемией. Первым, кто дал название болезни – «диабет» был врач Аретиус, живший в Риме во втором веке н. э. Намного позже уже в 1776 году врач Добсон (англичанин по происхождению), исследуя мочу больных диабетом обнаружил, что у нее сладковатый привкус, который говорил о наличии в ней сахара. Так, диабет стал называться «сахарным». При любом типе диабета контроль содержания сахара в крови становится одной из первостепенных задач пациента и его лечащего врача. Чем ближе уровень сахара к границам нормы, тем меньше проявляются симптомы диабета, и меньше риск возникновения осложнений | <div>Изменить</div> <div>Удалить</div> |
|---|--------|---|--|

< 1 >

7.Добавление нового симптома

Создание симптома

Название симптома

Чувство неутолимой жажды

Сохранить

| | | |
|---|--------------------------|--|
| 6 | Чувство неутолимой жажды | <div>Изменить</div> <div>Удалить</div> |
|---|--------------------------|--|

< 1 >

8.Добавление новой связки заболевания и симптома

Энциклопедия лекарств

Vacheslav ▾

Заболевания

Лекарства

Симптомы

Симптомы болезней

Лечение

Бюджетные аналоги

Симптомы заболеваний

Создать

Количество элементов: 10 ▾

Поиск

| ID ▴ ▾ | Название заболевания ▴ ▾ | Симптом ▴ ▾ | Действия ▴ ▾ |
|--------|--------------------------|------------------------------------|--|
| 1 | Ангина | Боль в горле | <div>Изменить</div> <div>Удалить</div> |
| 3 | Гайморит | Насморк | <div>Изменить</div> <div>Удалить</div> |
| 4 | Остеоартроз | Боль в суставах | <div>Изменить</div> <div>Удалить</div> |
| 5 | Изжога | желудочные и кишечные расстройства | <div>Изменить</div> <div>Удалить</div> |

<

1

>

Энциклопедия лекарств

Vacheslav ▾

Создание связи симптома и заболевания

Выберите заболевание

Диабет ▾

Выберите симптом

Чувство неутолимой жажды ▾

Сохранить

| | | | |
|---|--------|--------------------------|--|
| 6 | Диабет | Чувство неутолимой жажды | <div>Изменить</div> <div>Удалить</div> |
|---|--------|--------------------------|--|

<

1

>

9. Добавление новой связки заболевания и лекарства

Заболевания

Лекарства

Симптомы

Симптомы болезней

Лечение

Бюджетные аналоги

Лечение

Создать

Количество элементов 10 ▾

Поиск

| ID | Заболевание | Лекарство | Действия |
|----|-------------|--------------------|--|
| 1 | Ангина | Стрепсилс Интенсив | <div>Изменить</div> <div>Удалить</div> |
| 2 | Аллергия | Кларитин | <div>Изменить</div> <div>Удалить</div> |
| 3 | Гайморит | Отривин | <div>Изменить</div> <div>Удалить</div> |
| 4 | Остеоартроз | Найз | <div>Изменить</div> <div>Удалить</div> |
| 5 | Изжога | Фосфалюгель | <div>Изменить</div> <div>Удалить</div> |

<

1

>

Создание лечения против заболевания

Выберите заболевание

Диабет ▾

Выберите симптом

Келтикан комплекс ▾

Сохранить

| ID | Заболевание | Лекарство | Действия |
|----|-------------|-------------------|--|
| 6 | Диабет | Келтикан комплекс | <div>Изменить</div> <div>Удалить</div> |

<

1

>

10. Поиск лекарства по названию



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

| Поиск по лекарствам | | |
|--|--------------------|----------------------------|
| Количество элементов | 10 | Поиск <input type="text"/> |
| Картинка лекарства | Название лекарства | Цена |
|  | Стрепсилс Интенсив | 306 руб. |
|  | Кларитин | 262 руб. |



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ

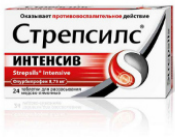


АДМИНИСТРИРОВАНИЕ

Поиск по лекарствам

Количество элементов: 10

Поиск: Стреп

| Картинка лекарства | Название лекарства | Цена |
|---|--------------------|----------|
|  | Стрепсилс Интенсив | 306 руб. |

< 1 >



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Просмотр лекарства



Стрепсилс Интенсив

Цена 306 руб.

Бюджетные аналоги

| Название | Стоимость |
|-------------------------------|-----------|
| Бюджетные аналоги отсутствуют | |

Описание

Таблетки медово-лимонные, 24 шт.

Инструкция

Инструкция по применению

| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

116

11. Поиск лекарства по названию заболевания



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по заболеваниям

Количество элементов 10

Поиск

| Название | Инструкция | Лекарства | Действия |
|----------|---|-----------------------|--------------------------------------|
| Аллергия | Аллергия (др.-греч. ἄλλος — другой, иной, чужой + ἔργον — воздействие) — типовой иммунопатологический процесс, выраженный сверхчувствительностью иммунной системы организма при повторных воздействиях аллергена на ранее сенсибилизированный этим аллергеном организм. | Кларитин | Посмотреть полностью |
| Ангина | Острый тонзиллит (от лат. tonsillae — миндалины), в обиходе — ангина (от лат. angio — «сжимаю, сдавливаю, душу»), — инфекционное заболевание с местными проявлениями в виде острого воспаления компонентов лимфатического глоточного кольца, чаще всего нёбных миндалин, вызываемое стрептококками или стафилококками, реже другими микроорганизмами, вирусами и грибами (см. подробнее Этиология[=]). Ангиной также называют обострение хронического тонзиллита. | Стрепсилс
Интенсив | Посмотреть полностью |
| Гайморит | Гайморит (максиллит) — воспаление верхнечелюстной (гайморовой) придаточной пазухи носа, возникает как осложнение при остром насморке, гриппе, кори, скарлатине и других инфекционных заболеваниях. | Отривин | Посмотреть полностью |
| Диабет | Сахарный диабет — эндокринное заболевание, обусловленное недостатком в организме гормона инсулина или его низкой биологической активностью. Характеризуется нарушением всех видов обмена веществ, поражением крупных и мелких кровеносных сосудов и проявляется гипергликемией. Первым из названных болезней — диабет был врач Аристотель, живший в Риме во втором веке н.э. | Келтикан
комплекс | Посмотреть полностью |



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по заболеваниям

Количество элементов 10

Поиск Ангина

| Название | Инструкция | Лекарства | Действия |
|----------|---|-----------------------|--------------------------------------|
| Ангина | Острый тонзиллит (от лат. tonsillae — миндалины), в обиходе — ангина (от лат. angio — «сжимаю, сдавливаю, душу»), — инфекционное заболевание с местными проявлениями в виде острого воспаления компонентов лимфатического глоточного кольца, чаще всего нёбных миндалин, вызываемое стрептококками или стафилококками, реже другими микроорганизмами, вирусами и грибами (см. подробнее Этиология[=]). Ангиной также называют обострение хронического тонзиллита. | Стрепсилс
Интенсив | Посмотреть полностью |

< 1 >

| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

117



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Просмотр лекарства



Стрепсилс Интенсив

Цена 306 руб.

Бюджетные аналоги

| Название | Стоимость |
|-------------------------------|-----------|
| Бюджетные аналоги отсутствуют | |

Описание

Таблетки медово-лимонные, 24 шт.

Инструкция

12. Поиск лекарства по симптомам



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по симптомам

Выберите симптомы

Поиск



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по симптомам

Выберите симптомы

Поиск



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по симптомам

Выберите симптомы

Боль в горле

Поиск

Количество элементов 10

Поиск

| Название ↑↓ | Описания ↑↓ | Лекарства ↑↓ |
|-------------|--|---|
| Ангина | Острый тонзиллит (от лат. tonsillae — миндалины), в обиходе — анги́на (от лат. angō — «сжимаю, сдавливаю, душу»), — инфекционное заболевание с местными проявлениями в виде острого воспаления компонентов лимфатического глоточного кольца, чаще всего нёбных миндалин, вызываемое стрептококками или стафилококками, реже другими микроорганизмами, вирусами и грибами (см. подробнее Этиология[=])[3]. Ангиной также называют обострение хронического тонзиллита. | Стрепсилс
Интенсив |

< 1 >



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Просмотр лекарства



Стрепсилс Интенсив

Цена 306 руб.

Бюджетные аналоги

| Название | Стоимость |
|-------------------------------|-----------|
| Бюджетные аналоги отсутствуют | |

Описание

Таблетки медово-лимонные, 24 шт.

Инструкция

Инструкция по применению

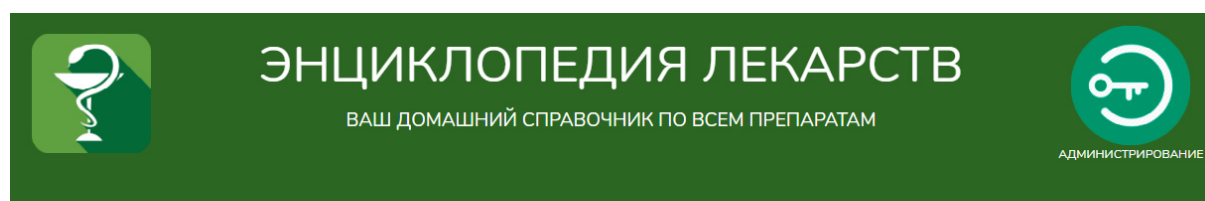
| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

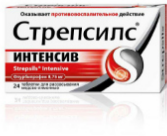
119

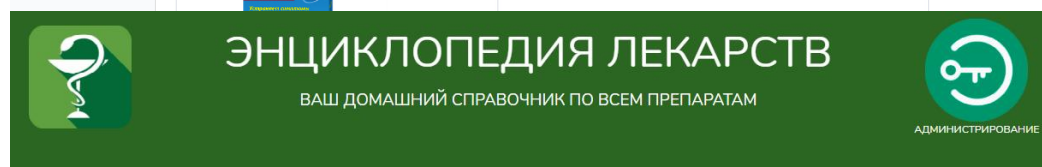
13. Поиск самого бюджетного аналога лекарства



Поиск по лекарствам

Количество элементов 10 Поиск

| Картинка лекарства | Название лекарства | Цена |
|--|--------------------|----------|
|  | Стрепсилс Интенсив | 306 руб. |
|  | Кларитин | 262 руб. |



Просмотр лекарства

|  | Стрепсилс Интенсив |
|---|--------------------|
| | Цена 306 руб. |
| Бюджетные аналоги | |
| Название | Стоимость |
| Аджисепт | 155 |

Просмотр лекарства



Аджисепт

Цена 155 руб.

Бюджетные аналоги

| Название | Стоимость |
|-------------------------------|-----------|
| Бюджетные аналоги отсутствуют | |

Описание

Таблетки для рассасывания с ананасом, 24 шт.

14.Использование фильтров поиска необходимой информации



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

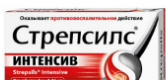
ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по лекарствам

Количество элементов
Поиск

| Картинка лекарства | Название лекарства | Цена |
|---|--------------------|----------|
|  | Келтикан комплекс | 750 руб. |
|  | Стрепсилс Интенсив | 306 руб. |



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

Поиск по лекарствам

Количество элементов 10

Поиск

| Картинка лекарства | Название лекарства | Цена |
|---|--------------------|----------|
|  | Аджисепт | 155 руб. |
|  | Отривин | 177 руб. |

15.Выход из профиля администратора

Энциклопедия лекарств

Vacheslav

Выйти

[Заболевания](#) [Лекарства](#) [Симптомы](#) [Симптомы болезней](#) [Лечение](#) [Бюджетные аналоги](#)

Добро пожаловать в Админ-панель, Vacheslav

Данный раздел предназначен для добавления, редактирования, изменения записей в приложении.



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



АДМИНИСТРИРОВАНИЕ

ПОИСК



ПО НАЗВАНИЮ ЛЕКАРСТВА



ПО НАЗВАНИЮ ЗАБОЛЕВАНИЯ



ПО СИМПТОМАМ

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| | | | | | | 123 |
| Изм. | Лист | № докум. | Подпись | Дата | | |

ВНЕДРЕНИЕ
ПРИЛОЖЕНИЕ Д

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| | | | | | | 124 |
| Изм. | Лист | № докум. | Подпись | Дата | | |



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Пензенский государственный университет»
(ФГБОУ ВО «ПГУ»)

Учебно-научный центр «Клинический медицинский центр ПГУ»

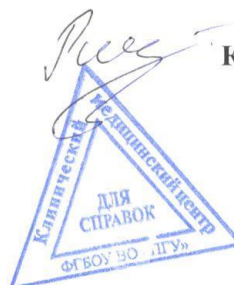
Красная ул., д.40, г. Пенза, Россия, 440026. Тел./факс: (841-2) 56-51-22

E-mail: cnit@pnzgu.ru; <http://www.pnzgu.ru>, kmc-pgu@mail.ru

ОКПО 02069042, ОГРН 1025801440620, ИНН 5837003736 КПП 583701001

Веб-приложение по персональному поиску требуемых медицинских препаратов, разработанное студентом Пензенского государственного университета Боруновым Вячеславом Игоревичем, находится на стадии тестирования с целью возможного дальнейшего внедрения в работу УНЦ КМЦ ПГУ.

Программист 2 кат. КМЦ ПГУ



Киреев Р.А.

| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

125

ПРЕЗЕНТАЦИЯ
ПРИЛОЖЕНИЕ Е

| | | | | | | |
|------|------|----------|---------|------|------------------------|------|
| | | | | | ПГУ 1.090501.02.001 ПЗ | Лист |
| | | | | | | 126 |
| Изм. | Лист | № докум. | Подпись | Дата | | |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ТЕМА РАБОТЫ: «ВЕБ-ПРИЛОЖЕНИЕ ПО ПЕРСОНАЛЬНОМУ ПОИСКУ ТРЕБУЕМЫХ МЕДИЦИНСКИХ ПРЕПАРАТОВ»

АВТОР РАБОТЫ: СТУДЕНТ ГРУППЫ 15В02 БОРУНОВ В.И.
НАУЧНЫЙ РУКОВОДИТЕЛЬ: Д.Т.Н., ПРОФЕССОР ГУРИН Е.И.

ПГУ 2020Г.



ЦЕЛЬ РАБОТЫ

РАЗРАБОТАТЬ ВЕБ-ПРИЛОЖЕНИЕ ДЛЯ ПОИСКА ТРЕБУЕМЫХ МЕДИЦИНСКИХ ПРЕПАРАТОВ, ГЛАВНОЙ ЦЕЛЮ КОТОРОГО ЯВЛЯЕТСЯ БОЛЕЕ ПРОСТОЙ И ЭФФЕКТИВНЫЙ СПОСОБ ДОСТУПА К ИНФОРМАЦИИ.

ОБЪЕКТ ИССЛЕДОВАНИЯ

ОБЪЕКТОМ ИССЛЕДОВАНИЯ ЯВЛЯЕТСЯ ОСУЩЕСТВЛЕНИЕ ПОИСКА ИНФОРМАЦИИ О ТРЕБУЕМЫХ ЛЕКАРСТВЕННЫХ ПРЕПАРАТАХ



| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

127

АКТУАЛЬНОСТЬ

АКТУАЛЬНОСТЬ РАЗРАБОТКИ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ПЕРСОНАЛЬНОГО ПОИСКА ТРЕБУЕМЫХ МЕДИЦИНСКИХ ПРЕПАРАТОВ ПОДТВЕРЖДАЕТСЯ СЛЕДУЮЩИМИ ФАКТОРАМИ:

- ПРИ ВНЕЗАПНЫХ НЕДОМОГАНИЯХ НАМ ТРЕБУЕТСЯ ВЫПИТЬ ЛЕКАРСТВО, НО ЕСЛИ ИНСТРУКЦИЯ ОТ НЕГО ПОТЕРЯНА ИЛИ ПРИШЛА В НЕГОДНОСТЬ, ТО СДЕЛАТЬ ЭТО НЕ ВОЗМОЖНО, ТАК КАК НЕ ЗНАЯ ПРАВИЛЬНОГО СПОСОБА ПРИМЕНЕНИЯ, ПРИЁМ ЛЕКАРСТВА ОПАСЕН ДЛЯ ЗДОРОВЬЯ! ДАННОЕ ВЕБ-ПРИЛОЖЕНИЕ ПОЗВОЛИТ ЗА СЧИТАНЫЕ СЕКУНДЫ НАЙТИ ВСЮ ИНФОРМАЦИЮ ЛЕКАРСТВЕ И СПОКОЙНО ПРИНЯТЬ ЕГО.
- ТАК ЖЕ ВОЗМОЖЕН ВАРИАНТ, ЕСЛИ У ВАС НЕТ ДОМА ТРЕБУЮЩИХСЯ ЛЕКАРСТВ, ДОПУСТИМ ОБЫЧНЫХ ТАБЛЕТОК ОТ БОЛИ В ЖИВОТЕ, ЗАПИСЬ И ПРИЁМ У ВРАЧА ЗАНИМАЕТ ОГРОМНОЕ КОЛИЧЕСТВО ВРЕМЕНИ, А В АПТЕКЕ НАМ МОГУТ ПОСОВЕТОВАТЬ КУПИТЬ ДОРОГОЙ ПРЕПАРАТ, БЛАГОДАря ПРИЛОЖЕНИЮ ВЫ САМИ СМОЖЕТЕ НАЙТИ ТРЕБУЮЩЕЕСЯ ВАМ ЛЕКАРСТВА И ПРОЧИТАТЬ ВСЮ ИНФОРМАЦИЮ ПО ЕГО ДЕЙСТВИЮ
- ЗНАЯ ЧТО ЛЕКАРСТВЕННЫЕ ПРЕПАРАТЫ СТОЯТ ДОВОЛЬНО ДОРОГО, И МНОГИМ МОЖЕТ БЫТЬ НЕ ПО КАРМАНУ ПОКУПАТЬ БРЕНДИРОВАННЫЕ ЛЕКАРСТВА, ТО ФУНКЦИЯ ПРИЛОЖЕНИЯ: "ПОИСК БЮДЖЕТНОГО АНАЛОГА", БУДЕТ КАК НЕЛЬЗЯ КСТАТИ!



РЕШАЕМЫЕ ЗАДАЧИ

- РЕГИСТРАЦИЯ НОВЫХ АДМИНИСТРАТОРОВ В СИСТЕМЕ;
- ПОЛНОЕ УПРАВЛЕНИЕ И ВЕДЕНИЕ БД, ПО СРЕДСТВАМ АДМИН ПАНЕЛИ;
- ОСУЩЕСТВЛЕНИЕ ПОИСКА ТРЕБУЕМОГО ЛЕКАРСТВЕННОГО ПРЕПАРАТА ТРЕМЯ РАЗНЫМИ СПОСОБАМИ:
 - 1) ПО НАЗВАНИЮ ЛЕКАРСТВА;
 - 2) ПО НАЗВАНИЮ ЗАБОЛЕВАНИЯ;
 - 3) ПО СИМПТОМАМ ЗАБОЛЕВАНИЯ.
- ЗАДАВАТЬ ФИЛЬТРЫ ПОИСКА ЛЕКАРСТВЕННЫХ ПРЕПАРАТОВ;
- ВЫВОД САМОГО ДЕШЁВОГО АНАЛОГОВОГО ЛЕКАРСТВЕННОГО ПРЕПАРАТА.



ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ

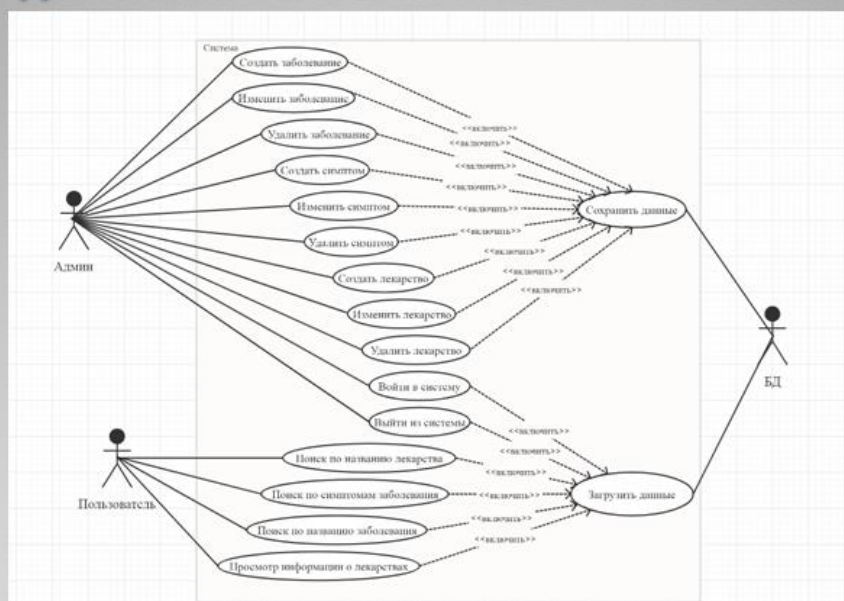


ДИАГРАММА СУЩНОСТЬ-СВЯЗЬ

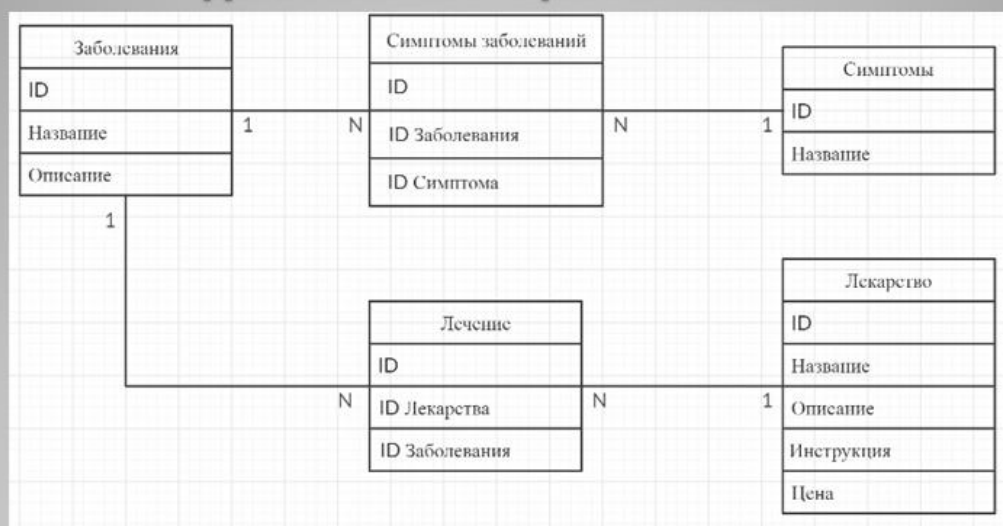


ДИАГРАММА ПОСЛЕДОВАТЕЛЬНОСТИ

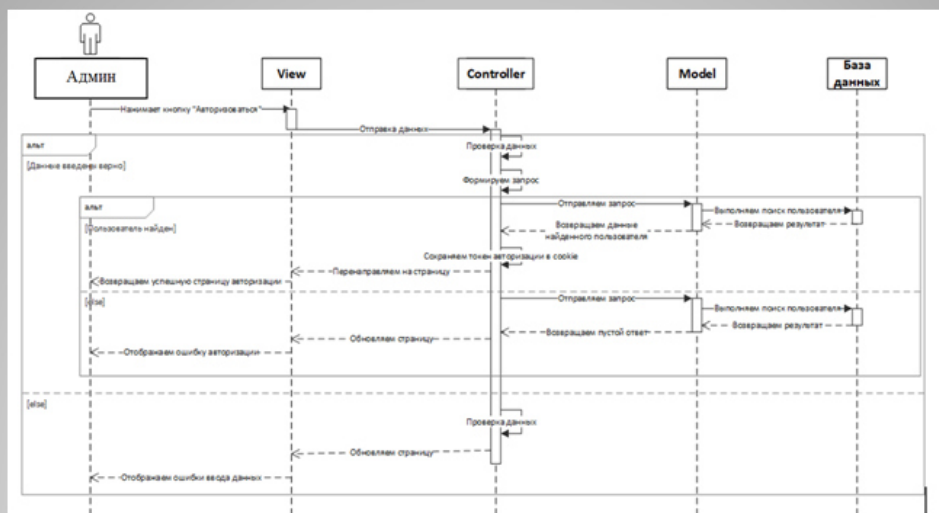
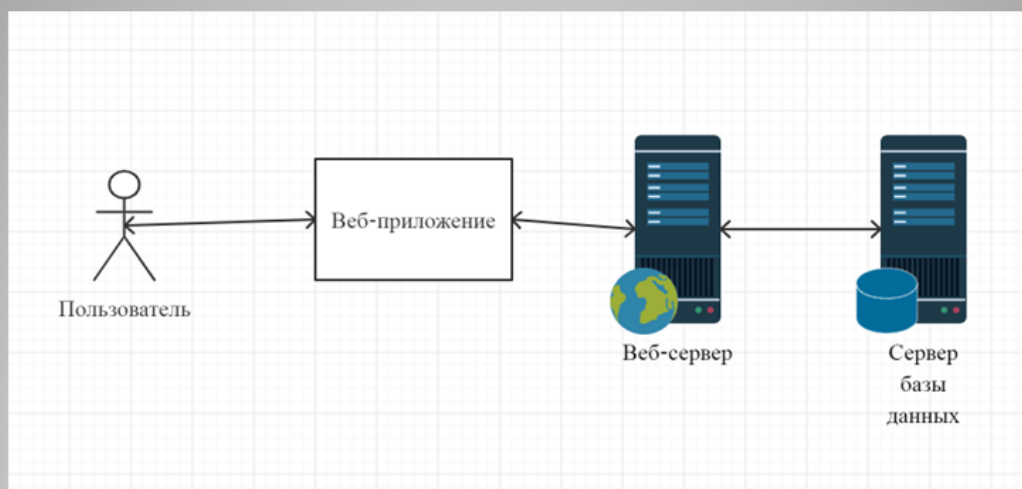


ДИАГРАММА ДЕЯТЕЛЬНОСТИ



АРХИТЕКТУРА ПРИЛОЖЕНИЯ



ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ



ADMINISTRATOR

ПОИСК



ПО НАЗВАНИЮ ЛЕКАРСТВА



ПО НАЗВАНИЮ ЗАБОЛЕВАНИЯ



ПО СИМПТОМАМ



РЕГИСТРАЦИЯ НОВОГО АДМИНИСТРАТОРА

Энциклопедия лекарств

[Войти](#) [Регистрация](#)

Регистрация

Name

E-Mail адрес

Пароль

Подтверждение пароля



| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

132

ИНФОРМАЦИЯ О ЛЕКАРСТВЕ

Аджисепт

155

Описание

Таблетки медово-лимонные, 24 шт.

Инструкция

Инструкция по применению

Производитель Состав Фармакологическое действие Показания Противопоказания

Побочные действия Взаимодействие Как принимать, курс приема и дозировка

Передозировка Специальные указания Форма выпуска Условия хранения Срок

годности Действующее вещество Лекарственная форма

Производитель

Рекитт Бенкизер, Великобритания

Состав

1 таблетка для рассасывания содержит:

Действующие вещества:

флурбипрофен 8,75 мг;



ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

Энциклопедия лекарств

Vacheslav ▾

Заболевания

Лекарства

Симптомы

Симптомы болезней

Лечение

Бюджетные аналоги

Добро пожаловать в Админ-панель, Vacheslav

Данный раздел предназначен для добавления, редактирования, изменения записей в приложении.



| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

133

ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС



ЭНЦИКЛОПЕДИЯ ЛЕКАРСТВ

ВАШ ДОМАШНИЙ СПРАВОЧНИК ПО ВСЕМ ПРЕПАРАТАМ

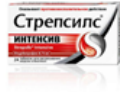


АДМИНИСТРИРОВАНИЕ

Поиск по лекарствам

Количество элементов 10

Поиск

| Картинка лекарства | Название лекарства | Цена |
|---|--------------------|----------|
|  | Стрепсилс Интенсив | 306 руб. |
|  | Кларитин | 262 руб. |



СПАСИБО ЗА ВНИМАНИЕ!



| | | | | |
|------|------|----------|---------|------|
| | | | | |
| Изм. | Лист | № докум. | Подпись | Дата |

ПГУ 1.090501.02.001 ПЗ

Лист

134