Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

|  |
| --- |
| *К защите допустить*: |
| Заведующий кафедрой ИТАС |
| \_\_\_\_\_\_\_\_\_\_\_ А.А. Навроцкий |

Пояснительная записка

к дипломному проекту

на тему

**АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ ПОЛОЧНЫМ ПРОСТРАНСТВОМ СУПЕРМАРКЕТА**

БГУИР ДП 1-53 01 02 06 013 ПЗ

|  |  |  |
| --- | --- | --- |
| Студент |  | В. Г. Колесников |
| Руководитель |  | В. Л. Арановский |
| Консультанты: |  |  |
| *от кафедры ИТАС* |  | Т. С. Боброва |
| *по экономической части* |  | Т. Л. Слюсарь |
| Нормоконтролер |  | В. И. Ярмолик |
| Рецензент |  |  |

Минск 2020

**РЕФЕРАТ**

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ ПОЛОЧНЫМ ПРОСТРАНСТВОМ СУПЕРМАРКЕТА: дипломный проект / В. Г. Колесников. – Минск : БГУИР, 2020, – п.з. – 64 с., чертежей (плакатов) – 6 л. формата А1.

Цель работы: разработка автоматизированной системы управления полочным пространством супермаркета.

Пояснительная записка к дипломному проекту состоит из введения, 4 разделов, включающих характеристику объекта разработки, разработку автоматизированной системы управления полочным пространством супермаркета, программную реализацию автоматизированной системы, а также технико-экономическое обоснование разработки системы, заключение, список использованных источников.

Для разработки программного комплекса был выбран язык программирования Python. Система представляет собой клиентское приложение, написанное с использованием библиотеки React, и серверное приложение, написанное на фреймворке Django. Доступ к приложению пользователи будут получать посредством веб-браузера.

Результатом работы стала автоматизированная система управления полочным пространством супермаркета, предоставляющая возможность собирать статистику об изменениях артериального давления, отслеживать их динамику, получать оповещения и рекомендации о возможных изменениях артериального давления. Программный комплекс также включает в себя базу данных, которая будет формироваться в ходе работы приложения, и может иметь большое значение в сфере здравоохранения, как агрегатор показателей артериального давления различных категорий людей при различных условиях.

Министерство образования Республики Беларусь

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Учреждение образования БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| Факультет | | | ИТиУ | | | | Кафедра | | | | ИТАС | | | | | | | | | | | |
| Специальность | | | 1-53 01 02 | | | | Специализация | | | | | | | 06 | | | | | | | | |
| УТВЕРЖДАЮ | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | |  | | | | | Зав. кафедрой | | | | |
| « | | | | | | | | | | | | | 2 | | » | | апреля | | | 2020 | г. | |
| **ЗАДАНИЕ** | | | | | | | | | | | | | | | | | | | | | | |
| **по дипломному проекту (работе) студента** | | | | | | | | | | | | | | | | | | | | | | |
| Колесникова Владислава Гавриловича | | | | | | | | | | | | | | | | | | | | | | |
| (фамилия, имя, отчество) | | | | | | | | | | | | | | | | | | | | | | |
| 1 Тема проекта (работы): | | | | Автоматизированная система управления полочным | | | | | | | | | | | | | | | | | | |
| пространством супермаркета | | | | | | | | | | | | | | | | | | | | | | |
| утверждена приказом по университету от | | | | | | | « | 03 | | » | | апреля | | | | 2020 г. | | | № | 750-с | | |
| 2 Срок сдачи студентом законченной работы | | | | | | | | | 1 июня 2020 г. | | | | | | | | | | | | | |
| 3 Исходные данные к проекту: | | | | | версия операционной системы – Windows 7 и выше; | | | | | | | | | | | | | | | | | |
| СУБД – *mongodb*; | | | | | | | | | | | | | | | | | | | | | | |
| Язык программирования – *Python*; Библиотека – *React*; Фреймворк – *Django*; | | | | | | | | | | | | | | | | | | | | | | |
| Библиотека - Express; | | | | | | | | | | | | | | | | | | | | | | |
| Назначение разработки: | | | | | | | | | | | | | | | | | | | | | | |
| разработать автоматизированную систему управления полочным пространством | | | | | | | | | | | | | | | | | | | | | | |
| супермаркета | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| 4 Содержание пояснительной записки (перечень подлежащих разработке вопросов): | | | | | | | | | | | | | | | | | | | | | | |
| Введение | | | | | | | | | | | | | | | | | | | | | | |
| 1 Характеристика объекта разработки | | | | | | | | | | | | | | | | | | | | | | |
| 2 Проектирование структуры автоматизированной системы управления полочным | | | | | | | | | | | | | | | | | | | | | | |
| пространством супермаркета | | | | | | | | | | | | | | | | | | | | | | |
| 3 Программная реализация автоматизированной системы | | | | | | | | | | | | | | | | | | | | | | |
| 4 Технико-экономическое обоснование разработки автоматизированной системы | | | | | | | | | | | | | | | | | | | | | | |
| Заключение | | | | | | | | | | | | | | | | | | | | | | |
| 5. Перечень графического материала (с точным указанием обязательных чертежей): | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | |
| 6 Содержание задания по технико-экономическому обоснованию. | | | | | | | | | | | | | | | | | | | | |
| Технико-экономическое обоснование разработки проекта «Автоматизированная | | | | | | | | | | | | | | | | | | | | |
| система управления полочным пространством супермаркета» | | | | | | | | | | | | | | | | | | | | |
| Задание выдал |  | | | | Т. Л. Слюсарь | | | | | | | | | | | | | | | |

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов дипломного проекта (работы) | Объём этапа, % | Срок выполнения этапа | Примечание |
| Сбор и изучение материалов по теме дипломного | 40 | 21.04.2020 |  |
| проектирования. Написание раздела 1 пояснительной записки, расчет технико-экономического обоснования |  |  |  |
|  |  |  |  |
| Проектирование программного комплекса. Написание раздела 2 пояснительной записки и соответствующего графического материала | 60 | 03.05.2020 |  |
|  |  |  |  |
| Реализация программного комплекса. Написание раздела 3 пояснительной записки и соответствующего графического материала | 80 | 17.05.2020 |  |
|  |  |  |  |
| Оформление пояснительной записки и подготовка презентации | 100 | 01.06.2020 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Дата выдачи задания | 02.04.2020 | | Руководитель | |  | Т.С. Боброва |
| Задание принял к исполнению | |  | | В.Г. Колесников | |

**СОДЕРЖАНИЕ**

[Перечень условных обозначений, символов и терминов 5](#_Toc9905439)

[Введение 6](#_Toc9905440)

[1 Характеристика объекта разработки 8](#_Toc9905441)

[1.1 Система кровообращения человека 8](#_Toc9905442)

[1.2 Анализ значений артериального давления 11](#_Toc9905444)

[1.3 Анализ существующих аналогов разрабатываемого программного комплекса 12](#_Toc9905445)

[1.4 Постановка задачи 17](#_Toc9905449)

[2 Проектирование структуры программного комплекса анализа артериального давления 18](#_Toc9905450)

[2.1 Структура разрабатываемого программного комплекса 18](#_Toc9905451)

[2.2 Проектирование программного комплекса с использованием инструментальных средств 19](#_Toc9905452)

[2.3 Информационное обеспечение программного комплекса 25](#_Toc9905455)

[2.4 Алгоритмическое обеспечение 28](#_Toc9905459)

[2.5 Техническое и системное программное обеспечение 33](#_Toc9905460)

[2.6 Эргономическое обеспечение 33](#_Toc9905461)

[3 Программная реализация программного комплекса анализа артериального давления 35](#_Toc9905462)

[3.1 Обоснование выбора средств разработки 35](#_Toc9905463)

[3.2 Структура программного продукта 38](#_Toc9905464)

[3.3 Проектирование программной реализации комплекса 38](#_Toc9905465)

[3.4 Описание программной реализации отдельных частей комплекса 39](#_Toc9905466)

[3.5 Руководство пользователя 54](#_Toc9905473)

[4 Технико-экономическое обснование разработки программного комплекса анализа артериального давления 55](#_Toc9905474)

[4.1 Характеристика программного продукта 55](#_Toc9905475)

[4.2 Оценка трудоемкости и сроков разработки 55](#_Toc9905476)

[4.3 Расчет затрат на разработку программного комплекса 58](#_Toc9905477)

[4.4 Расчёт экономической эффективности реализации программного комплекса аналиха артериального давления 60](#_Toc9905478)

[4.5 Результат технико-экономического обоснования разработки программного продукта 61](#_Toc9905479)

[Заключение 62](#_Toc9905480)

[Список использованных источников 63](#_Toc9905481)

[Приложение А (обязательное) Листинг моделей соединения с базой данных 64](#_Toc9905482)

[Ведомость документов дипломного проекта 6](#_Toc9905483)5

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ, символов и терминов

Торговый объект - имущественный комплекс, а также иное имущество, используемое для осуществления торговли, принадлежащие на праве собственности, праве хозяйственного ведения, праве оперативного управления или на ином законном основании торговым организациям и (или) индивидуальным предпринимателям.

Магазин – специально оборудованное стационарное здание или его часть, предназначенное для продажи товаров и оказания услуг покупателям и обеспеченное торговыми, подсобными, административно-бытовыми помещениями, а также помещениями для приема, хранения и подготовки товаров к продаже.

Киоск – оснащенное торговым оборудованием строение, не имеющее торгового зала и помещений для хранения товаров, рассчитанное на одно рабочее место продавца, на площади которого хранится товарный запас.

Универсальный магазин – предприятие розничной торговли, реализующее универсальный ассортимент продовольственных и непродовольственных товаров.

Супермаркет – магазин с торговой площадью от 650 до 4000 квадратных метров, в котором реализуется универсальный ассортимент продовольственных товаров и ограниченный ассортимент непродовольственных товаров методами самообслуживания, традиционного обслуживания, продажи товаров по предварительным заказам.

Гипермаркет – предприятие торговли, реализущее продовольственные и непродовольственные товары универсального ассортимента преимущественно по форме самообслуживания.

Планограмма – документ, в котором детально изображается выкладка товаров с точным указанием мест размещения на торговом оборудовании торгового предприятия ассортиментных позиций.

Мерчендайзинг – оптимизация системы торговли, связанная с подготовкой товаров, их рекламой, а также стимулированием торговой деятельности.

*UML (Unified Modeling Language) –*унифицированный язык объектно-ориентированного моделирования.

*IDEF*0 – методология, которая представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области.

***IDEF*3** – методология моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов.

СУБД – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

# ВВЕДЕНИЕ

В настоящее время люди слабо себя представляют без посещения магазина. Даже если они делают это не каждый день, рано или поздно все равно приходится сходить или съездить за продуктами, которые им необходимы. Обычная процедура покупки мало чем отличается раз за разом. Однако, не всегда получается приобрести нужный товар определенной фирмы. Это может происходить по разнообразным причинам, начиная от того, что данный товар все еще не готов, и заканчивая тем, что фирма перестала его производить. Существует и обратная проблема, когда фирма сама хочет найти как можно больше покупателей, но не может из-за ряда обстоятельств.

Оставив без внимания случаи, когда отсутствие товара является проблемой фирмы, рассматривается посредник между покупателем и продавцом – торговый объект.

Существуют ситуации, в которых основная проблема того, что покупатель не купил нужный товар, исходит от самого магазина: товар не поступил вовремя на склад, либо его просто не выставили вовремя. Конечно, можно решить эту проблему радикальным способом: все время заполнять полки и склад до отказа, чтобы дефицит данного товара просто не случалось. Но в данном случае есть две существенные проблемы. Во-первых, у некоторых товаров есть определенный срок годности, после которого его придется просто утилизировать. Во-вторых, излишки товаров сами магазины не всегда могут себе позволить, т.к. тогда, скорее всего, очень сильно упадет рентабельность магазина в связи с избытком закупленного товага, который магазин не в силах продать.

С другой стороны, проблему можно решить, наняв большое количество персонала, который бы следил за состоянием магазина, вовремя пополнял склад и полки, а также, в случае необходимости, консультировал покупателя в процессе выбора товара. Однако, при такой ситуации магазин может нести большие финансовые потери, т.е. вся прибыль будет уходить на зарплаты. В данном случае могут даже возникнуть сомнения по поводу того, что магазин сможет обеспечивать свой персонал конкурентной оплатой труда. Поэтому становится неизвестно, насколько целесообразны данные меры.

Это лишь некоторые варианты развития событий, которые руководство супермаркета может выбрать.

К счастью, современные решения в области информационных технологий позволяют автоматизировать большой пласт операций, совершаемых персоналом магазина. При внедрении в торговый процесс разработок, направленных на облегчение выполнения повседневных и многоразовых операций, позволяют многократно сократить затраты на те или иные действия при разумных вложениях.

Одним из вариантов частичной автоматизации работы супермаркета может являться автоматизированная система управления полочным пространством. Прежде всего, данная система позволяет следить за общим состоянием магазина в целом и каждой полки по отдельности в режиме реального времени на основе производимых операций купли-продажи и выкладки товаров на полки. Кроме того, данная система позволяет собирать и анализировать данные, поступающие в процессе работы супермаркета, на основе чего, вырабатывается дальнейшая тактика работы с выдачей определенных рекомендаций.

Для разработки данной системы необходимо решить следующие задачи:

* исследовать преимущества и недостатки существующих автоматизированных систем такого же или похожего типа;
* определить требования к автоматизированной системе;
* спроектировать автоматизированную систему;
* экономически либо социально обосновать разработку и реализовать автоматизированную систему.

Преимуществом данной автоматизированной системы является база данных, которая будет заполняться данными о товарах, их количестве на полках и складе, а также точным или приблизительным сроком годности. Кроме того, система обладает возможностью анализировать ежедневные операции с товарами, на основе чего строится статистика работы супермаркета и настраиваются показатели на очередной день его работы.

Тема дипломного проекта: «Автоматизированная система управления полочным пространством супермаркета». Целью данного дипломного проекта является разработка и реализация автоматизированной системы управления полочным пространством супермаркета. Система позволит следить за общим состоянием магазина и каждой полки по отдельности, а также выдавать статистические данные о работе магазина с последующей настройкой параметров, что позволит сократить некоторые расходы на персонал.

Дипломный проект выполнен самостоятельно, проверен в системе «Атиплагиат». Процент оригинальности соответствует норме, установленной кафедрой (68%). Цитирования обозначены ссылками на публикации, указанные в «Списке использованных источников».

# АНАЛИЗ ОБЪЕКТА АВТОМАТИЗАЦИИ

## Описание объекта автоматизации

Принято различать торговые объекты по их видам и типам, которые, в свою очередь, зависят от различных факторов. Существует большое количество разновидностей магазинов. В основе их классификации лежат следующие признаки:

* тип ассортимента;
* уровень цен;
* форма обслуживания потребителей;
* местонахождение.

Каждый из них оказывает существенное влияние на прибыль бизнеса, но, в зависимости, например, от региона, экономической ситуации в данный период времени и прочих субъективных факторов, содержит те или иные признаки [1].

Большое влияние на тип торгового объекта оказывает его местоположение. Например, в таких местах, как остановки общественного транспорта, наиболее распространены киоски. Бóльшим разнообразием отличаются места, где чаще всего можно заметить массовое скопление людей. В таких местах обычно располагаются торговые центры, которые могут иметь как гигантские универсальные магазины, так и маленькие продуктовые магазины. Также существует такой класс гипермаркетов, которые могут располагаться за пределами города или на его границе. Обычно, такая форма торгового объекта предлагает широкий ассортимент товаров разнообразных классов и видов. Предназначены такие гипермаркеты преимущественно для людей, которые посещают магазин раз в несколько недель, а иногда и реже.

На данный момент размер торгового объекта и его ассортимент определяется, прежде всего, наличием спроса на тот или иной товар. Так, если спрос на строительные материалы преобладает в определенной местности, вероятнее всего, там может появиться специализированный магазин, который будет удовлетворять нужды покупателей.

Исходя из спроса, чаще всего происходит регулирование цен на ту или иную продукцию. Так, например, при достаточном количестве товара в разных торговых объектах и одинаковом спросе на него, при условии, что эти торговые объекты находятся неподалеку друг от друга, выиграет тот магазин, у которого цена на товар будет более привлекательной, нежели в другом. К сожалению, несмотря на регулирование цен, существуют ситуации, в которых цена на товар может варьироваться в зависимости от предложения данного товара. Так, например, при дефиците товара определенного вида его цена будет большей, чем он мог бы стоить на самом деле. Чаще всего такие ситуации происходят в малых городах и деревнях, где товары сами по себе являются дефицитными в таких местах.

При достаточном количестве торговых объектов в больших городах иногда выигрывает тот, который является более дружелюбным и подходящим для клиента. То есть если магазины предлагают определенный товар по примерно одинаковой цене, может выиграть тот, в котором, по мнению покупателя, персонал более снисходителен и внимателен к покупателю, даже если цена в этом торговом объекте немного больше, чем в другом. Кроме того, если в магазине товары расположены таким образом, что найти нужную позицию бывает довольно сложно, а иногда и невозможно, либо торговое пространство распределено крайне неграммотно, такой магазин может потерять клиента в связи с данными обстоятельствами.

Учитывая выделенные факторы, необходимо проанализировать каждый вид торгового объекта, чтобы выделить тот вид, которому действительно будет необходима проектируемая автоматизированная система.

## Киоск

Киоски являются наиболее распространенной формой торговых объектов, расположенных около остановочных пунктов общественных транспортных средств. Чаще всего, киоски связаны с индивидуальной предпринимательской деятельностью, ограничены в разнообразии предоставляемых услуг и вмещают в себя относительно небольшое количество товаров. Обычно, в киоске есть небольшое торговое пространство, в котором может находиться только один продавец, окруженный ассортиментом товаров. Внешний вид киоска представлен на рисунке 1.1.

Данный вид торгового объекта на ранних этапах своего существования предоставляет весьма небольшой ассортимент, чаще всего товаров, не связанных между собой. В начале каждый товар располагается на точном месте, где он должен быть по определению. Со временем спрос на определенный вид товаров начинает опережать предложение магазина, из-за чего, в случае недостатка конкретного места для конкретного товара, продавец либо несет потенциальные убытки из-за отсутствия данного товара, либо пытается укомплектовать товар как можно плотнее.

Сами по себе киоски не нуждаются в правильно выверенном расположении товара, то есть определенная позиция может находиться в любом месте. Это происходит по причине того, что в киосках большая часть товаров находится сразу в поле зрения покупателя.



Рисунок 1.1 – Киоск

Беря во внимание данные факторы, становится очевидно, что данный вид торгового объекта не нуждается в автоматизированной системе управления полочным пространством из-за своих особенностей и малого пространства.

## Магазин

В данном случае магазин рассматривается как торговый объект, имеющий достаточно пространства для нахождения в нем от одного до нескольких продавцов и покупателей. Этот вид торгового объекта наиболее распространен в небольших городах, агрогородках и деревнях, но также магазины могут быть и в больших и средних городах, преимущественно в тех районах, где нет поблизости супермаркетов или гипермаркетов, которые забирают на себя основную часть покупателей в силу большого ассортимента товаров разного типа.

Небольшие магазины имеют свои особенности по сравнению с киосками и универсальными магазинами. Так, например, в данном виде торгового объекта, по сравнению с киоском, есть пространство не только для продавцов и товаров, но и для покупателей, что делает его немного более значимым, относительно киоска в силу разности их размеров. Как и в киосках, в небольших магазинах зачастую покупатель сразу же видит весь ассортимент товаров. Кроме того, покупателю предоставляется возможность выбора из большего списка товаров, нежели в киоске. Внутренний вид продовольственного магазина представлен на рисунке 1.2.



Рисунок 1.2 – Продовольственный магазин

Также, кроме своих размеров, небольшой магазин преобладает и в специализации предоставляемых услуг. Например, существуют специализированные магазины, предоставляющие товары определенного вида или класса, такие как магазины стройматериалов, ювелирные магазины, магазины бытовой техники, магазины косметики и другие. Внутренний вид магазина стройматериалов представлен на рисунке 1.3.

Несмотря на свое разнообразие, данный вид торгового объекта не всегда предоставляет большой ассортимент товаров разного вида, в результате чего он проигрывает универсальным магазинам.



Рисунок 1.3 – Магазин строительных материалов

Так как в магазине могут присутствовать лишь несколько продавцов и покупателей, а ассотримент товаров зачастую представляет собой небольшую группу, относительно универсальных магазинов, данный вид торговых объектов является довольно простым во взаимодействии между покупателями и продавцами. Для продавцов с большим опытом работы в определенном магазине становится намного проще ориентироваться в списке товаров, необходимых для покупателя, что также помогает рекомендовать покупателю те товары, которые нужны покупателю и выгодны при продаже для магазина.

Проанализировав выделенные преимущества и недостатки данного вида торговых объектов, можно прийти к выводу, что небольшие магазины могут нуждаться только в автоматизации составления планограмм. Глубокий и расширенный анализ продаж и оптимизации расположения и количества товаров на полках магазина не требуются, так как расстановкой товаров на полках и их правильным расположением занимается продавец в течение рабочего дня.

## Супермаркет

Кеееек.

## Анализ современных методов автоматизации работы супермаркетов

Сами по себе отклонения значений артериального давления хоть и являются серьезным, но редко могут являться единственным аргументом при вынесении диагноза о заболевании. Чаще всего необходимо учитывать множество различных факторов, начиная с жалоб, образа жизни, колличества стрессов, количества потребляемого алкоголя и заканчивая ростом и весом. Все эти данные, в совокупности с отклонениями значений артериального давления от нормы, могут способствовать принятию верного решения.

На самом деле, у каждого человека имеется свое рабочее давление. Это такое давление, когда человеку комфортно. Оно не обязательно является нормотоническим, т.е. общепринятым нормальным 120 на 80. Людей с общепринятой нормой давления в 120 на 80 называют *нормотониками*, людей с частым повышенным артериальным давлением 140 на 90 называют *гипертониками*, людей с пониженным 90 на 60 называют *гипотониками*. К сожалению это не значит, что они здоровы, т.к. не смотря на хорошее самочувствие, они по прежнему и более других подвержены тем заболеваниям, которые характерны для их вида отклонений от нормы.

Учитывая такую индивидуальность, в поисках патологий рассматривают пульсовую разницу, которая в норме не должна выходить за рамки от 30 до 50 единиц, учитывая фактор возраста. Если исправить ситуацию с показателями АД можно используя капли для поднятия или препараты для понижения давления, то с пульсовой разницей дело обстоит гораздо сложнее. Эта величина очень информативна и указывает на заболевания, требующие лечения.

Маленькая пульсовая разница проявляется у человека в виде слабости, апатии, раздражительности, сонливости или нарушения внимания. Низкая пульсовая разница всегда должна вызывать опасения. Она свидетельствует о таких патологических процессах, как сердечная недостаточность, недостаточность внутренних органов, инсульт левого желудочка, аортальном стенозе, тахикардии, кардиосклерозе и многих других.

Большая же пульсовая разница, в свою очередь, является еще более опасным фактором. Она может указывать на угрозу инсульта. Если произошло увеличение пульсовой разницы, это подсказывает, что сердце теряет свою активность и имеет место риск инфаркта миокарда.

В случае значительного отклонения значения пульсовой разницы от нормативного значения, необходимо обратиться к кардиологу.

## Анализ существующих систем и инструментов

В рамках анализа существующих аналогов и прототипов разрабатываемого программного комплекса были изучены имеющиеся на рынке системы схожего предназначения. Многие из них специализируются такой функции, как построение оптимального плана выкладки товаров на полке для повышения продаж и увеличения прибыли. Кроме того, они предоставляют некоторый анализ уровня продаж. Однако, не удалось выявить систему, которая настраивала бы свои параметры в процессе работы и выдавала информацию по текущему состоянию продуктов и товаров на полках, а также рекомендации к действиям в определеннный момент времени. Относительно данных факторов, можно сделать вывод, что разрабатываемая автоматизированная система предоставляет уникальные возможности в своей сфере.

## ABM Shelf

Данная система представляет собой облачное решение, разработанное компанией *ABM Cloud*. *ABM Shelf* предоставляет возможность составлять планограммы, моделировать залы и схемы выкладки в зависимости от разнообразных факторов. Кроме того, предоставляется возможность анализировать продажи магазина, в зависимости от чего выдаются статистические данные.

Основные функции системы *ABM Shelf*:

* конструктор оборудования с возможностями объемного моделирования различных видов торгового оборудования с учетом габаритных характеристик (длина, глубина, высота и др.) и создания произвольного количества шаблонов для торгового оборудования с фиксированными размерами;
* эффективное управление торговым пространством;
* управление торговым залом;
* работа с базой данных магазинов (готовый перечень), удобный поиск и навигация;
* создание моделей торгового зала для магазинов (неограниченное количество);
* аналитика продаж торгового зала (*ABC* анализ, анализ продаж);
* автоматизация управления выкладкой и управление торговым оборудованием;
* разработка планограммы выкладки товара;
* работа с несколькими стеллажами одновременно;
* удобная визуальная аналитика по данным продаж (суммы продаж) и выкладки (по производителям, по полкам, по габаритам, стоимости, остаткам) с возможностью экспорта в *Excel* [2]. Стартовая страница сайта системы *ABM Shelf* представлена на рисунке 1.3.

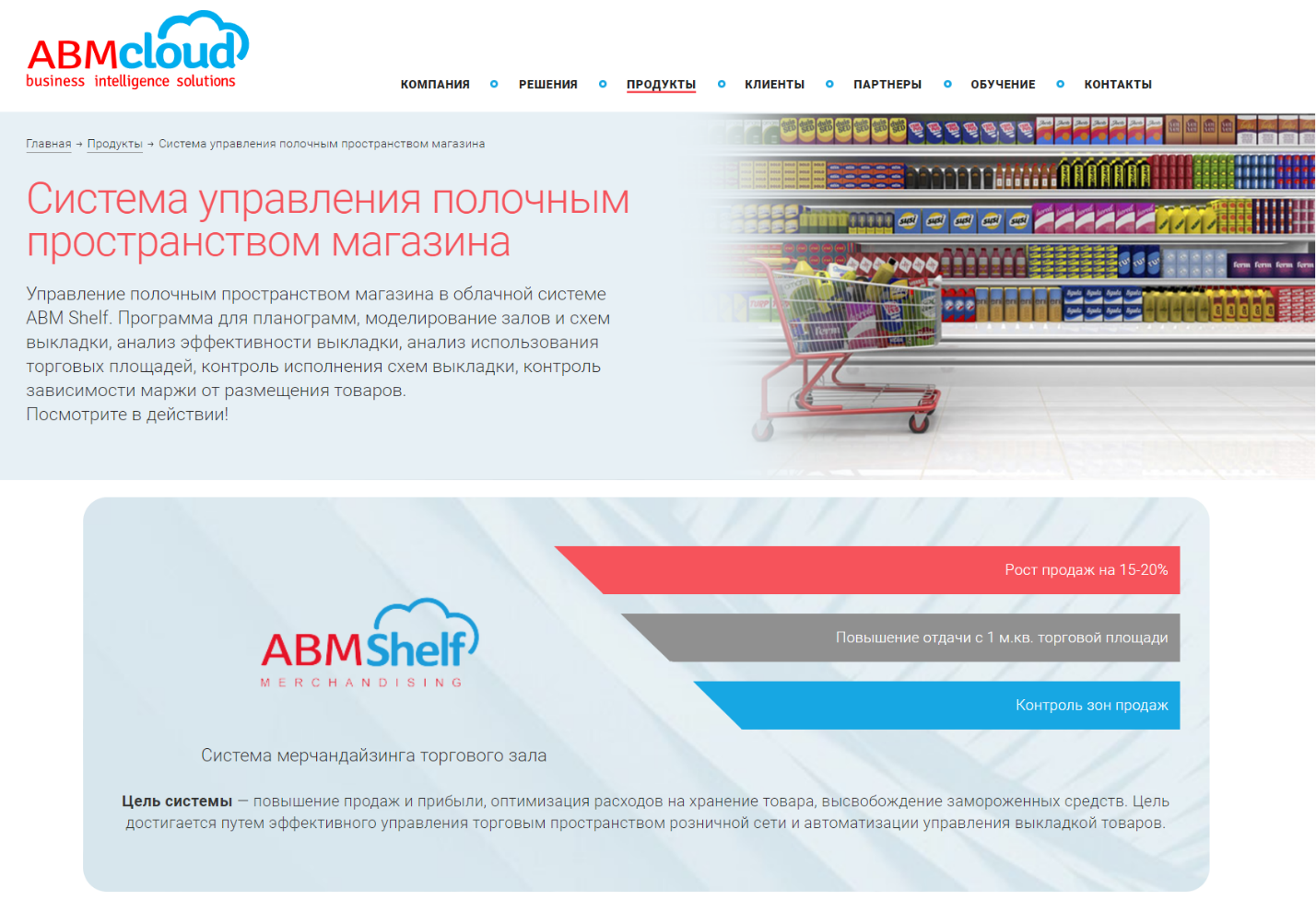


Рисунок 1.3 – Стартовая страница сайта системы *ABM Shelf*

Система предоставляет много функционала по управлению полочным пространством. Данные о расположении товаров предоставляются в виде изображений торгового зала и каждого стеллажа по отдельности. Расположение товаров просчитывается автоматическими алгоритмами, благодаря чему выдаются оптимальные варианты расстановки товаров.

Также система предоставляет управление торговым пространством розничной сети с помощью анализа данных по мерчендайзингу товара, который предполагает анализ таких параметров, как текущий остаток товара, выкладка товара и оптимальное количество товара.

К плюсам данной системы можно отнести:

* составление планограмм;
* моделирование торговых залов;
* аналитика продаж.

К минусам можно отнести:

* не всегда интуитивно понятный интерфейс;
* отсутствие взаимодействия со складом магазина.

## RS.ShelfSpace

*Retail Suite Shelfspace* — это система управления полочным пространством магазина. Она позволяет автоматизировать и настроить эффективное управление торговым пространством розничной сети. Реализован сценарный подход, который позволяет одновременно управлять несколькими задачами, запущенными параллельно. Торговое пространство розничной сети должно быть эффективным и управляемым. Для того, чтобы этого добиться, *RS.ShelfSpace* позволяет контролировать целый набор показателей визуального мерчандайзинга, отражающих качество использования торговых площадей.

При моделировании розничного пространства, *RS.ShelfSpace* обрабатывает сотни и тысячи объектов. Управление полочным пространством магазина выполняется централизованно. Выкладка товаров осуществляется по рассчитанным системой планограммам, построение которых, в свою очередь, ведется по настроенным правилам в автоматическом режиме. Реализована всеобъемлющая поддержка как регулярных продаж, так и промо-акций [3].

Данная система обладает рядом преимуществ и недостатков. Прежде всего, она, возможно, могла бы предоставить больший функционал, однако по исходным данным видно, что система реализует лишь несколько общих вариантов автоматизации, без привнесения новизны в уже существующие методы. Так, например, данная система может составлять планограммы и возможные варианты расположения товаров только по заранее настроенным правилам, то есть это позволяет выставить определенный набор правил перед началом работы, относительно которого система выдаст возможные варианты без последующей возможности перенастройки системы в процессе работы. Кроме того, так как данная система анализирует данные только в конце рабочего дня, она не может сигнализировать об отсутствии определенного товара на полке, поэтому в данном случае остается человеческий фактор отслеживания заполненности полок.

В дополнение можно сказать, что данная система имеет довольно громоздкий интерфейс, предоставляющий информацию о стеллажах в виде списка типа «дерево». Данное решение может сказаться на интуитивном понимании интерфейса в случае разрастания списка. Пример конфигурации магазина представлен на рисунке 1.4. Стартовая страница сайта системы *RS.ShelfSpace* представлена на рисунке 1.5.



Рисунок 1.4 – Пример конфигурации магазина

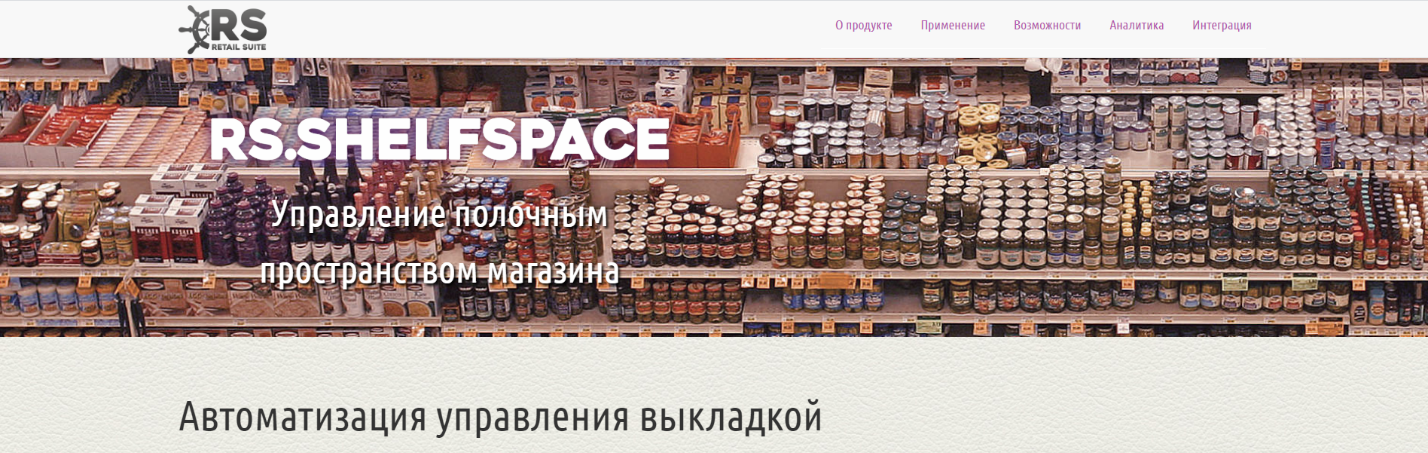


Рисунок 1.5 – Стартовая страница сайта системы *RS.ShelfSpace*

Преимущества системы *RS.ShelfSpace*:

* автоматизация мерчендайзинга;
* создание планограмм;
* анализ продаж.

Недостатки данной системы:

* анализ данных происходит только после загрузки всех транзакций, т.е. в конце рабочего дня;
* громоздкий интерфейс.

## PlanoManager

Инструмент *PlanoManager* предназначен для создания планограмм и анализа их эффективности. Данное решение позволяет визуализировать торговое оборудование и размещение продукции, а также определить необходимый уровень товарного запаса для оценки и оптимизации выкладки. *PlanoManager* автоматически обновляет базу данных продуктов и автоматически строит планограммы на основе поступивших продуктов.

Данный инструмент поддерживает не так много функций, как предыдущие системы, однако, базируясь на одном пункте, он делает себя максимально эффективным в сфере планограмм. Также, *PlanoManager* проигрывает в плане автоматизации, так как этот инструмент не предоставляет расширенной автоматизации на основе анализа продаж. Стартовая страница сайта инструмента PlanoManager представлена на рисунке 1.6.

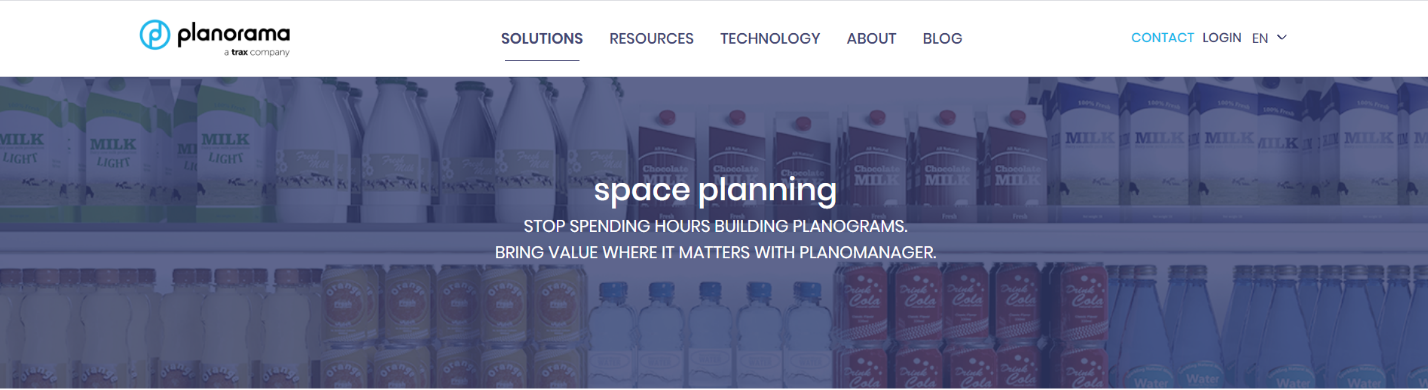


Рисунок 1.7 – Стартовая страница сайта инструмента *PlanoManager*

К преимуществам данного инструмента можно отнести:

* приятный, простой и интуитивно понятный интерфейс;
* расширенный функционал для построения планограмм;
* автоматическое обновление базы данных продуктов (на основе данных, предоставляемых пользователем);
* визуализация планограмм.

К недостаткам можно отнести:

* отсутствие расширенного анализа данных;
* отсутствие автоматизации контроля полочного пространства.

## Постановка задачи

Разработать автоматизированную систему управления полочным пространством супермаркета, которая позволит автоматизировать ряд задач по работе с товарами, сократит расходы на персонал и улучшит качество обслуживания клиентов.

Система должна обладать возможностью визуализации текущего состояния супермаркета, анализировать поступающие данные и настраивать параметры своей работы в соответствии с полученной после обработки данных статистикой.

Необходимо, чтобы данная система была доступна для персонала супермаркета и обладала возможностью разграничения прав доступа к отдельным ее частям: управляющий персонал должны иметь доступ ко всем компонентам системы, авторизованные пользователи и рабочий персонал – к страницам бла-бла-бла, неавторизованные пользователи – только к страницам «Домашняя» и «Авторизация». Доступ к системе должен предоставляться через веб-интерфейс.

1. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ПОЛОЧНЫМ ПРОСТРАНСТВОМ СУПЕРМАРКЕТА

## Структура разрабатываемой автоматизированной системы

Исходя из того, что программный комплекс должен быть доступен пользователям в сети интернет, однако для большинства операций необходим доступ к базе данных, для реализации проекта была выбрана модель клиент-серверного приложения.

Клиент-серверная архитектура приложения подразумевает под собой разделение приложения на различные модули, решающие различные задачи и выполняющиеся в разных местах. Клиентская часть решает задачи отображения контента, серверная же часть, в свою очередь, содержит бизнес логику, основные алгоритмы и берет на себя большую часть вычислительных нагрузок. Крмое того, серверная часть осуществляет связь с базой данных приложения.

К преимуществам клиент-серверной архитектуры можно отнести:

* серверная часть комплекса является куда более защищенной, благодаря более качественному контролю доступа различных уровней клиентов к данным;
* минимальные системные требования к компьютерам пользователей, т.к. вся аналитическая нагрузка вынесена на сервер;
* отсутствие дублирования кода программы-сервера программами-клиентами.

Однако такая архитектура имеет и крайне серьезный недостаток, который заключается в том, что недоступность сервера по любой причине парализует работу всего программного комплекса. Кроме того, серверное оборудование чаще всего имеет высокую стоимость.

Пользователь взаимодействует с клиентской частью приложения, которая при необходимости формирует запросы и обращается на сервер за данными, после чего принимает ответ и отображает данные.

Сервер, при поступлении на него запроса, совершает обработку введенных данных, в большинстве случаев обращается в базу данных за данными, из которых, после анализа, формирует ответ и отправляет на клиент.

Архитектура приложения представлена на рисунке 2.1.

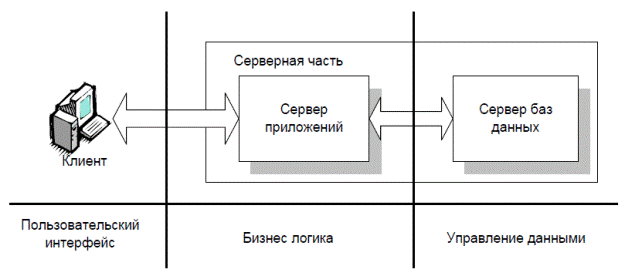


Рисунок 2.1 – Архитектура клиент-сервеного приложения

## Проектирование программного комплекса с использованием инструментальных средств

## Проектирование программного комплекса с использованием языка UML.

*UML (Unified Modeling Language)* – это унифицированный язык объектно-ориентированного моделирования. Он служит для наглядного представления функциональности разрабатываемого программного продукта. Диаграмму на этом языке называют просто диаграммой вариантов использования. Она описывает взаимодействия объектов программного комплекса между собой в процессе использования программного продукта по его назначению. Диаграмма вариантов использования программного комплекса представлена на рисунке 2.2.

В проектируемом программном комплексе можно выделить двух актеров: авторизированного и неавторизированного пользователя.

У каждого из актеров имеются свои прецеденты использования. Прецедент – это типичное взаимодействие пользователя с системой, которое при этом:

* описывает видимую пользователем функцию;
* может представлять различные уровни детализации;
* обеспечивает достижение конкретной цели, важной для пользователя.

Рассмотрим прецеденты каждого актера. Возможности программного комплекса, доступные незарегистрированному пользователю:

* Получение нормального значения артериального давления для введенного сочетания возраста и пола;
* Регистрация в системе;
* Вход в учетную запись.

Возможности авторизированного пользователя:

* Получение нормального значения артериального давления для введенного сочетания возраста и пола, аналогично неавторизованному пользователю;
* Получение рекомендаций по улучшению состояния на основе анализа уточненных и расширенных данных;
* Получение уведомлений о возможном ухудшении самочувствия на основе погодных условий.
* Получение визуализации отклонения проанализированных данных от нормы с течением времени, представленное ввиде графика;
* Выход из учетной записи.

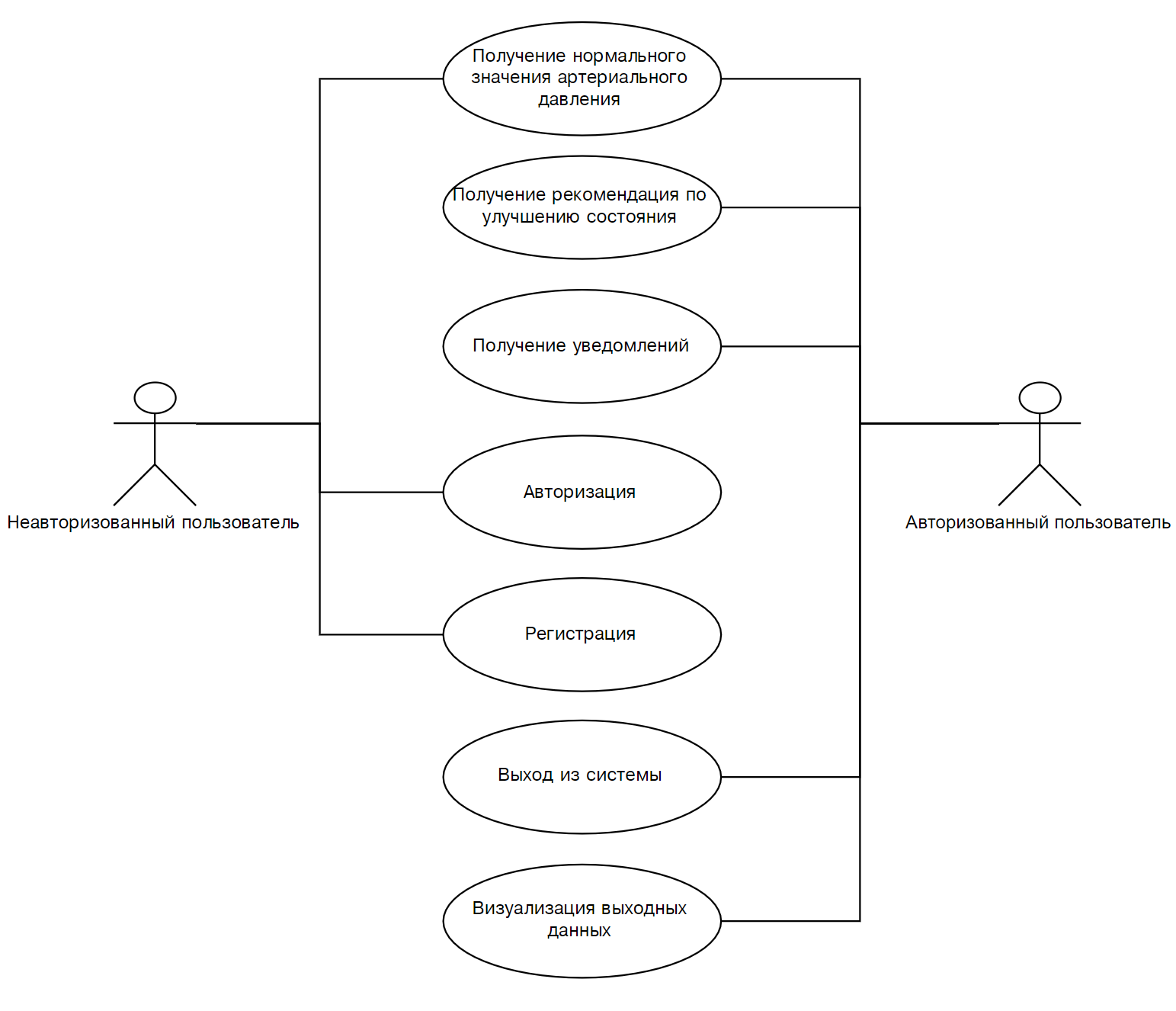


Рисунок 2.2 – Диаграмма вариантов использования программного комплекса

При открытии сайта пользователь попадает на главную страницу, с которой он может либо зарегистрироваться, либо войти в систему, либо на основе примитивного анализа получить данные о нормальном артериальном давлении для введенного сочетания возраста и пола.

При входе в систему пользователь попадает на страничку личного кабинета, на которой он может на основе проведенного анализа введенных данных получить рекомендации по улучшению своего состояния, включить рассылку уведомлений, а также выйти из системы.

Кроме того есть возможность получить визуализацию отклонения значений своего артериального давления от нормы в виде графика.

## Функциональная структура программного комплекса

*IDEF0* – методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов. В *IDEF*0 рассматривается функциональная структура объекта, т.е. выполняемые обьектами действия и логические отношения между ними, а не их временная последовательность.

**Методология** *IDEF0* предписывает построение иерархической системы диаграмм: сначала проводится описание комплекса в целом и его взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функциональная деком­позиция, т.е. происходит разбиение комплекса на подкомплексы и так далее, вплоть до необходимого уровня декомпозиции.

Функциональная модель *IDEF0* представляет собой набор блоков, каждый из которых имеет входы и выходы, управление и механизмы. Наиболее важная функция расположена в верхнем левом углу. Соединяются функции между собой при помощи стрелок и описаний функциональных блоков. При этом каждый вид стрелки или активности имеет собственное значение. Данная модель позволяет описать все основные виды процессов.

Контекстная диаграмма разрабатываемого комплекса представлена на рисунке 2.3.

На вход системы подается информация о пользователе. Эти данные будут отличаться в зависимости от типа пользователя.

Допустимые значения параметров – это допустимые значения таких характеристик, как вес, рост, возраст, образ жизни, степень потребления алкоголя и сигарет, на основе которых, анализируя входные данные, программный комплекс будет рассчитывать выходной результат.

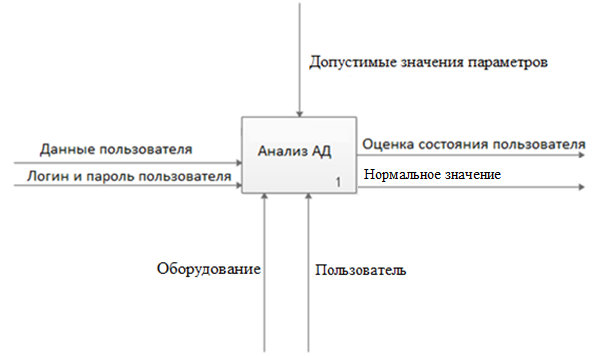


Рисунок 2.3 – Контекстная диаграмма программного комплекса анализа артериального давления

В качестве механизмов управления выступают пользователь, которым является человек. Человек, перейдя на сайт в сети интернет, вводит входные данные, а также оборудование, к которому относится сервер, на котором хранится информация о зарегистрированных пользователях, а также их статистика.

Оценка состояния пользователя – это данные с рекомендациями и проанализированными значениями артериального давления, которые поступают на выходе программного комплекса при прохождении пути авторизированного пользователя.

Нормально значение – это данные с нормальным значением артериального давления для введенных возраста и пола. Эти выходные данные поступают на выходе при прохождении пути неавторизированного пользователя.

Чтобы более подробно описать процессы, происходящие внутри программного комплекса используется диаграмма декомпозиции, представленная на рисунке 2.4.



Рисунок 2.4 – Диаграмма декомпозиции программного комплекса анализа артериального давления

Но диаграмме изображены 7 работ, которые будут выполняться в процессе функционирования системы.

Авторизация необходима для идентификации пользователя и определения его прав доступа. Регистрация необходима для создания новых пользователей.

Поступление данных в систему необходимо для считывания введенных данных из форм, формирования и отправки запросов на сервер.

Загрузка данных авторизированного пользователя в базу и выборка его истории служит для занесения полученных от клиента данных в базу данных, а также для выборки из базы данных всей доступной информации о пользователе, о его предыдущих замерах показателей.

На следующем шаге происходит анализ данных пользователя. На этом этапе, на основе выборки данных пользователя из базы данных и анализирующего алгоритма, происходит формирование массива рекомендаций и диагнозов, если таковые имеются, и отправка сформированного массива на клиент.

Формирование выходных данных авторизированного пользователя визуализирует полученные с сервера данные в читабельную структуру.

Поступление данные необходимо для считывания из формы введенных пользователем значений, формировании запроса и отправки его на сервер.

Анализ введенных данных включает в себя примитивный анализ введенных данных и определение нормального значения артериального давления для введенной совокупности значений.

Формирование выходных данных формирует в результате анализа результирующее значение и отправляет его на клиент, который в свою очередь отображает это значение.

Исходя из того, что в процессе работы программного комплекса подразумевается работа с информацией, имеет смысл построение диаграммы на основе *DFD* методологии для этапа работы с базой данных. Такая диаграмма представлена на рисунке 2.5.



Рисунок 2.5 – Диаграмма декомпозиции процесса работы с базой данных

На диаграмме декомпозиции процесса работы с базой данных изображен процесс работы с базой данных значений. Поступающие на входе данные, представляющие из себя набор характеристик пользователя, необходимых для анализа, заносятся в базу данных, после чего происходит извлечение всех записей для данного пользовательского токена, формируется массив записей для текущего пользователя и подается на выход.

## Информационное обеспечение программного комплекса

## Входные данные

Входные данные прогрмаммного комплекса могут быть двух видов. Они отличаются в зависимости от вида пользователя.

Входными данными для неавторизированного пользователя являются:

* Возраст пользователя;
* Пол пользователя;

Входными данными для авторизированного пользователя являются:

* Возраст пользователя;
* Пол пользователя;
* Систолическое давление;
* Дистолическое давление;
* Рост пользователя;
* Вес пользователя;
* Образ жизни, под которым подразумевается степень физической активности пользователя.
* Степень употребления алкоголя;
* Отношение к курению.

Источником всех входных данных, за исключение даты, которая генерируется автоматически при отправке запроса на сервер, является пользователь. Периодичность поступления данных отсутствует. Это обусловлено спецификой программного комплекса.

## Разработка БД

Разрабатываемый программный комплекс взаимодействует с базой данных. База данных представляет из себя совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняется в соответствии с правилами средств моделирования данных. В базе данных разрабатываемого программного комплекса содержится 2 сущности: пользователи и давление.

В данном дипломном проекте была использована документоориентированная база данных, которая является *NoSQL* типом базы данных и работает с *JSON*-подобными документами.

Первичным ключом в обоих таблицах является *id* пользователя, которое служит для связи пользователя с его записями в таблице данных и выступает в роли токена пользователя. Оно генерируется уникальным для каждого пользвателя при помощи библиотеки *passport-local-mongoose*, которая используется для функциональности авторизации и для шифрования паролей.

Для работы с базами данных в *mongodb* необходима также модель сущности, описанная в коде, которая служит для связи с соответствующей базой данных. Эти модели позволяют в любой момент расширить структуру таблицы без причинения вреда хранящимся там данным. Листинг данных моделей приведен в приложении.

Сущность *«Users»* содержит информацию о пользователях. Её структура приведена в таблице 2.1.

Сущность *«Pressure»* содержит статистическую информацию, введенную пользователем. Её структура приведена в таблице 2.2.

Сущности базы данных связаны между собой при помощи ключевого поля, которым является первичный ключ, которым является уникальный идентификатор пользователя. На рисунке 2.5 представленая структурная схема используемой базы данных.



Рисунок 2.5 – Структурна схема базы данных

Таблица 2.1 – Структура таблицы *«Users»*

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| *Id* | *String* | Идентификатор пользователя |
| *Email* | *String* | Почта пользователя |
| *Password* | *String* | Пароль пользователя |
| *City* | *String* | Город пользователя |

Таблица 2.2 – Структура таблицы *«Pressure»*

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Описание |
| *Token* | *String* | Идентификатор пользователя |
| *Value* | *String* | Значение давления |
| *Age* | *Number* | Введенный возраст |
| *Gender* | *String* | Пол пользователя |
| *Date* | *Number* | Время ввода данных |
| *Height* | *Number* | Рост пользователя |
| *Weight* | *Number* | Вес пользователя |
| *PhysicalActivity* | *String* | Степень физической активности пользователя |
| *Alcohol* | *String* | Степень потребления алкоголя пользователя |
| *Smoke* | *Boolean* | Отношение к курению |

## Выходные данные

Выходными данными программного комплекса являются два типа информации: значение нормального артериального давления для неавторизированного пользователя, рекомендации и история для авторизованного пользователя.

Для неавторизованного пользователя выходная информация будет состоять из значения нормального артериального давления

Для авторизованного пользователя выходные данные будут состоять из следующего:

* История значений артериального давления (графическое отображение);
* История значений артериального давления (числовое отображение);
* Список рекомендаций и диагнозов пользователю.

## Алгоритмическое обеспечение

Исходя из того, что одной из основных целей работы разрабатываемого программного комплекса является анализ входных данных, имеет смысл описать алгоритм, который используется для анализа и формирования выходных данных.

Ниже представлены блок-схемы функционирования программного комплекса для всех видов пользователей.

Блок-схема функционирования программного комплекса для неавторизованного пользователя представлена на рисунке 2.6.

Если говорить о неавторизованном пользователе, то после ввода данных в форму и ее подтверждения на сервер отправляется запрос с введенными данными, сервер на основе полученных данных проверяет пол, затем для выбранного пола по введенному возрасту определяет нормативное значение и отправляет его в ответе клиенту, который в свою очередь отображает полученную информацию.

Первый этап для авторизованного пользователя аналогичен, это отправка введенных значений на сервер. Сервер извлекает данные из запроса и записывает их в базу данных, используя как *id* записи токен пользователя. Затем происходит выборка из базы данных всех значений с текущим токеном, таким образом мы получаем историю значений конкретного пользователя. После этого для каждой записи истории мы находим нормативное значение артериального давления на основе данных этой записи и добавляем это значение в эту запись. Затем для каждой записи истории мы запускаем модуль анализа, который пройдется по всем данным текущей записи и на основе их значений добавит в эту запись список рекомендаций и возможных причин отклонения значений артериального давления от нормы, после чего сервер отправит эти данные на клиент. Клиент, получив эти данные, визуализирует их, построив график отклонения значений артериального давления от нормы, так и просто в текстовом формате. Кроме того он отобразит список выявленных рекомендаций, причин и потенциальных диагнозов.

Блок-схема функционирования программного комплекса для авторизованного пользователя представлена на рисунках 2.7 и 2.8.



Рисунок 2.6 – Блок-схема функционирования программного комплекса для неавторизованного пользователя



Рисунок 2.7 – Блок-схема функционирования программного комплекса для авторизованного пользователя



Рисунок 2.8 – Блок-схема функционирования программного комплекса для авторизованного пользователя

## Техническое и системное программное обеспечение

Так как программный комплекс имеет архитектуру клиент-сервер, а клиентская часть запускается в браузере и написана с использованием современных решений в сфере веб-разработки, таких как *React* и *Redux*, минимальными системными требованиями со стороны клиента можно назвать минимальные системные требования браузера, поддерживающего эти технологии.

Серверная часть приложения также не производит сколь-либо тяжелых вычислительных операций, поэтому изначально не требуется больших вычислительных мощностей для ее функционирования. Однако с учетом ожидаемого ежедневного роста базы данных со значениями, а соответственно ежедневного увеличения сложности выборки данных из базы с последующей обработкой и анализом каждой записи, потенциально необходимость в вычислительных мощностях может увеличиться.

Ниже приведены предположительные минимальные системные требования.

Системные требования для пользователя:

* процессор *Intel® Core™ i*3*–*2350*M* 2300 *МHz* или более мощный;
* ОЗУ емкостью от 256 Мб;
* обязательно необходим выход в Интернет;
* встроенная видеокарта 256 – 1024 Мб или больше;
* монитор желательно с разрешением 1024х768 или больше;
* клавиатура, мышь.

Системные требования для реализации сервера:

* процессор *Intel® Core™ i*5*–*3230*M* 2.60 *GHz* или более мощный;
* ОЗУ емкостью от 512 Мб;
* обязательно необходим выход в Интернет;
* видеокарта встроенная;
* клавиатура, мышь.

## Эргономическое обеспечение

Эргономика программного обеспечения – это степень, в которой продукт может быть использован пользователем для достижения поставленных целей эффективно и с удовольствием в заданном контексте использования.

Под эргономикой программного обеспечения в большинстве случаев понимают такие вещи, как:

* удобство интерфейса;
* доступность и логичность размещения элеметнов упраления;
* оптимальные цветовые решения.

Для решения вопросов эргономичности сквозь весь программный комплекса прослеживается одна структура размещения окон. Все формы, сменяющие друг друга, расположены на одном месте, в центре, куда и направлен взгляд пользователя. Кнопки навигации по страницам расположены сразу под формами и сменяют друг друга на одном и том же месте, что исключает необходимость отыскивать навигационные ссылки по углам сайта. Кроме того, такой выбор структуры сайта позволил сохранить десктопный и мобильный дизайны максимально схожими, что также облегчает использование сайта пользователям, использующим и мобильную, и настольную версию. Благодаря адаптивности реализована мобильная версия, что позволяет значительно увеличить удобство использования сайта с мобильных устройств.

Для оформления страниц выбран единый стиль материального дизайна, привычный современным пользователям, с узкими и элегантными, но легко различаемыми и читаемыми белыми шрифтами на темном фоне. В качестве шрифтов для шрифта заголовка был выбран *Dosis*, а для основного текста сквозь все страницы используется *Raleway*. Также шрифт *Dosis* был выбран для текста на кнопках. Для сообщений об ошибках выбран темно-красный цвет, привлекающий внимание, однако не давящий на глаза.

Фон для форм выбран мягкий темно синий, чтобы не резать глаза пользователю и улучшить читаемость белых надписей. С точки зрения восприятия глазом, было принято решение слегка зажелтить остальной фон, так как желтый свет гораздо легче воспринимается глазами человека.

Фон для графика был выбран прозрачным с еле видимой сеткой, и темными значениями, так как акцент должен идти не на сами значения, а на динамику их изменений и отклонений с течением времени. Сами же значения, для большей наглядности, продублированы в цифровом виде.

Ключевым аспектом эргономики является отсутствие каких-либо раздражителей либо вещей, которые могут вызвать недовольство либо дискомфорт у пользователя. Чтобы достичь этого, были убраны любые моргающие эффекты, которые могут отвлекать пользователя, отсутствует банерная реклама.

## Организационное обеспечение

Организационное обеспечение – это совокупность методов и средств, регламентирующих взаимодействие работников с техническими средствами и между собой в процессе разработки и эксплуатации программного комплекса.

В данном дипломном проекте к средствам, регламентирующим взаимодействие пользователей с ситемой будут являться права пользователя.

Предполагается наличие двух видов пользователей, имеющих различные права:

* авторизованные пользователи;
* неавторизованные пользователи.

Тип пользователя будет определяющим фактором при определении доступной ему функциональности.

Так неавторизованный пользователь будет иметь доступ только к начальному экрану с первичной формой анализа, а также функциям регистрации и авторизации.

Авторизованный же пользователь, помимо начального экрана с первичной формой анализа, будет иметь доступ к своему личному кабинету, где и распологается более интересная функциональность, а также истории своих введенных данных.

1. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ПОЛОЧНЫМ ПРОСТРАНСТВОМ СУПЕРМАРКЕТА

## Обоснование выбора средств разработки

Процесс создания веб-сайта называется веб-разработкой. Он включает в себя верстку веб-страниц, веб-программирование, а также конфигурирование веб-сервера. Для каждой из перечисленных задач используются различные средства разработки.

Типичный веб-сайт представляет из себя текстовый файл в формате *HTML*, который может содержать ссылки на файлы в других форматах, а также ссылки на другие веб-сайты. По факту для саздания статичного веб-сайта достаточно только *HTML*, однако выглядят статические веб-сайты довольно скудно, к тому же они не обладают какой-либо приемлимой функциональностью, и разумеется не могут предоставить какой-либо возможности обратной связи с сервером.

Слегка оживить статические сайты, наполнить их какой-либо простой анимацией, расположить элементы а также украсить их внешне позволяет *CSS*. *CSS* – это формальный язык описания внешнего вида документа. Использоюется для задания цветов, изменения шрифтов, расположения элементов, создания простых анимаций.

Однако, чтобы сделать сайт хоть сколько-нибудь полезным и функциональным, придется воспользоваться одним из множества языков программирования.

Языки веб-программирования делятся на две группы: клиентские и серверные.

Клиентские языки программирования используются при *front-end* разработке, они обрабатываются на стороне клиента пользователя, т.е. программы на клиентском языке обрабатывает браузер. Отсюда следует недостаток: обработка скрипта зависит от браузера пользователя, и он в свою очередь имеет полномочия настроить свой браузер так, чтобы он игнорировал написанные программистом скрипты. При этом, если клиент использует старую версию браузера, то он может не поддерживать тот или иной язык или версию языка. С современными браузерами такие проблемы возникают редко, к тому же языки программирования обновляются сравнительно не часто и не кардинально, а лучшие из них давно известны.

Преимуществом же клиентского языка является то, что обработка скриптов на таком языке может выполняться без отправки документа на

сервер. Отсюда же вытекает и то ограничение, что с помощью клиентского языка программирования ничто не может быть записано на сервер, то есть, например, с его помощью нельзя сделать гостевую книгу, потому что тогда надо записывать сообщения в какой-либо файл на сервере.

Самым распространенным из клиентских языков является *JavaScript*, он так же, как и *HTML*, лежит в основе многих веб-технологий. На сегодняшний день этот язык является одним из самых популярных языков программирования, получил широчайшее распространение и его по умолчанию поддерживают практически все браузеры, операционные системы и мобильные устройства.

*Back-end* разработка подразумевает под собой создание серверной части приложения при использовании серверных языков программирования, которые соответственно работают на стороне сервера. Серверная часть приложения несет ответственность за взаимодействие с БД, архитектуру и программную логику.

К серверным языкам программирования можно отнести такие языки, как *PHP*, *JAVA*, *Perl*, *Python*, *Ruby*, а также любой .*NET* язык программирования. Все они имеют свои особенности и преимущества, однако в данном дипломном проекте будет использоваться *JavaScript* и на сервеной стороне. В этом на помощь приходит *Node.js*, который представляет из себя программную платформу, основанную на движке *v8*, превращающем *JavaScript* из узкоспециализированного языка в язык общего назначения, и выполняет роль веб-сервера. В качестве библиотеки будет использоваться *EXPRESS* – гибкая библиотека для *Node.js* приложений, предоставляющая большую функциональность для создания веб-приложений и *API*.

Среди систем управления базами данных, которые являются совокупность проммных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных, по модели данных выделяют следующие классификации: иерархические, сетевые, реляционные, объектно-ориентированные и объектно-реляционные.  
 Ярким представителем реляционных БД является *MySql*. Она подходит для огромного спектра задач, это проверенная и старая технология, которая используется крупными компниями вот уже 15 лет. Она обладает хорошей производительность, достойной расширяемостью, и проста в администрировании. Данные она хранит в виде таблиц, со строками и столбцами. Эти таблицы могут иметь связи между собой, что значительно упрощает работу с ними в некоторых случаях.

Если в *MySql* данные содержатся в таблицах, то в нереляционной MongoDB база данных состоит из коллекций, которые не имеют связей между собой. Это документоориентированная БД, в которой каждый документ можно представить как объект. Значительно упрощает работу с ней также формат данных, с которым она работает, а конкретно *JSON*.

Для создания веб-сайта были выбраны *HTML* и *CSS*, как не имеющие достойных аналогов средства написания веб-страниц. В качестве языка программирования для данного дипломного проекты был выбран язык *JavaScript*. Данный выбор был обусловлен огромной популярность *JavaScript*, его обширной поддержкой со стороны как ит-компаний, так и программистов, наличием необходимой функциональности как для написания *front-end* части, так и для *back-end* разработки. Кроме того, данный язык распростроняется на бесплатной основе и имеет открытый исходный код.

*JavaScript* – прототипный язык программирования. Он является реализацией языка *ECMAScript*(стандарт *ECMA*-262).

Обычно *JavaScript* используется в браузерах, как язык сценариев для приданий веб-сайтам интерактивности.

В связке с *Javascript* будут использоваться библиотеки *React* и *Redux*. *React* – это *JavaScript*-библиотека с открытым исходным кодом для разработки пользовательских интерфейсов. *React* может использоваться для разработки одностраничных и мобильных приложений. Он позволяет разбивать страницу на мелкие части, представляющие из себя независимые компоненты. *React* использует *JSX* синтаксис, который является некоей интерпретацией *HTML* внутри *JavaScript*. В качестве библиотеки для разработки пользовательских интерфейсов *React* часто используется с другими библиотеками, такими как *Redux*.

В качестве серверного языка приложения был выбран все тот же *JavaScript*, который будет использоваться в сочетании с такими библиотеками как *Node.js* и *express*.

В качестве СУБД будет использоваться *MongoDB*. Такой выбор обусловлен опытом работы автора с данной системой. Кроме того, она отлично подходит для работы с JavaScript, имеет гибкий формат документов *JSON*, работа с которым автор находит гораздо более приятным, чем с таблицами, колонками и столбцами. Кроме того проблема производительности при написании крупных запросов в данном случае легко решается на стороне преложения, что также является хорошим решением.

## Структура программного продукта

Перед проектированием сложного программного продукта необходимо определиться с его структурой. Для этого нужно определить структурные компоненты программного продукта, а также связи между ними, а после этого составить структурную схему.

Структурной схемой называют схему отражающую состав и взаимодействие частей разрабатываемого программного обеспечения.

Структурная схема программного комплекса анализа артериального давления представлена на рисунке 3.1.



Рисунок 3.1 – Структурная схема программного комплекса

## Проектирование программной реализации комплекса

В качестве архитектурного подхода клиентской части программного комплекса была выбрана *Flux*-архитектура. *Flux*-архитектура – это архитектурный подход для построения пользовательского интерфейса веб-приложения, сочетающийся с реактивным программированием и построенный на однонаправленных потоках данных.

Основной отличительной особенностью *Flux* явлется односторонняя направленность передачи данных между компонентами *Flux*-архитектуры. Архитектура накладывает ограничения на поток данных, исключая возможность обновления состояния компонентов самими собой. Такой подход позволяет отслеживать причины изменений и возможных ошибок, значительно облегчая их обнаружение.

В реализуемом программном комплексе *Flux*-архитектура содержит четыре слоя, взаимодействующие между собой по порядку:

* *Actions* (действия);
* *Dispatcher* (диспетчер);
* *Stores* (хранилища);
* *Views* (представления);

Действия – это выражения событий. По факту пресдавляют из себя объект, содержащий тип действия и данные, с которыми в процессе этого действия необходимо будет манипулировать.

Диспетчер – это функция, которая содержит список зависимостей вызова определенных функций от типа действия и в зависимости от типа переданного действия определяет, какая функция будет вызван для манипуляции с переданными данными.

Хранилище – это место сосредоточения данных приложение, где хранится его состояние. Его обновление происходит строго на основе его предыдущего состояния и данных действия.

Представление – это компонент, главное предназначение которого в отражении данных пользователю.

Для *Flux*-архитектуры важно, чтобы данные с представлений попадали в хранилища только через действия.

ДОПИШИ !!!!!!!

## Описание программной реализации отдельных частей комплекса

## Модуль регистрации нового пользователя

Этот модуль служит для добавления в систему новых пользоветелей. После введения почты, повторения нового пароля и введения города, если пользователь желает получать уведомления, происходит запрос на сервер. На сервере происходит проверка на наличие пользователей с аналогичной почтой, после чего, при успешном прохождении проверки, сервер заносит эту запись в базу данных пользователей. В ответ же он возвращает уникальный идентификатор пользователя, который будет использоваться при последующем использовании приложения пользователем.

Форма регистрации новых пользователей представлена на рисунке 3.2.

На рисунке 3.3 представлена форма регистрации новых пользователей с сообщением о том, что пользователь с такой почтой уже имеется в базе.

По нажатии кнопки «*Submit*» происходит отправка данных на сервер:

*formSubmit(e) {*

*e.preventDefault();*

*let form = e.target;*

*if (form[1].value === form[2].value) {*

*let data = {*

*username: form[0].value,*

*password: form[1].value,*

*city: form[2].value,*

*};*

*fetch(constants.register,{*

*method: 'POST',*

*headers:{'content-type': 'application/json'},*

*body: JSON.stringify(data)*

*})*

*.then(function(resp) {*

*return resp.json();*

*})*

*.then((resp) => {*

*that.props.updateToken(resp.token);*

*that.props.history.push('/personalcab');*

*})*

*.catch((error) => {*

*this.setState({error: 'User with such mail is already registered'});*

*});*

*} else {*

*this.setState({error: 'entered passwords are different'});*

*}*

*}*

На сервере происходит следующая обработка:

*apiRouter.post('/register', (req, res) => {*

*usersModel.findOne({'username': req.body.username}, (error, user) => {*

*if (!user) {*

*usersModel.register(new usersModel({*

*username: req.body.username,*

*city: req.body.city*

*}), req.body.password, function(err, user){*

*if (err) {return res.send({ message: err });}*

*});};*

*});*

*});*

## Модуль авторизации пользователей

Модуль авторизации пользователей служит для предоставления зарегистрированным пользователям идентификационного токена, который необходим для доступа к личному кабинету пользователя.

Он представляет из себя форму с полями для ввода почты и пароля. Форма авторизации представлена на рисунке 3.4. При подтверждении формы происходит формирование запроса и отправка данных на сервер:

*formSubmit(e) {*

*e.preventDefault();*

*let that = this;*

*let form = e.target;*

*let data = {*

*username: form[0].value,*

*password: form[1].value*

*};*

*fetch(constants.login,{*

*headers:{'content-type': 'application/json'},*

*body: JSON.stringify(data)*

*})*

*.then(function(resp) {*

*return resp.json();*

*})*

*.then((resp) => {*

*that.props.updateToken(resp.token);*

*that.props.history.push('/personalcab');*

*});*

*}*

Сервер проверяет наличие соответствующего пользователя в базе данных, и в случае успеха возвращает его токен:

*apiRouter.post('/login', passport.authenticate('local'), function(req, res) {*

*usersModel.findOne({ 'username': req.body.username }, (error, user) => {*

*res.send({token: user.\_id});*

*});*

*});*

## Модуль определения нормативного артериального давления

Этот модуль предназначен для неавторизованных пользователей и представляет из себя форму с полями для ввода возраста и выбора пола. Форма определения нормативного давления представлена на рисунке 3.5. При потверждении формы происходит отправка данных на сервер:

*formSubmit(e) {*

*e.preventDefault();*

*let that = this;*

*let form = e.target;*

*let data = {*

*age: form[0].value,*

*gender: form[1].value*

*};*

*fetch(constants.calculatePressureURL,{*

*method: 'POST',*

*headers:{'content-type': 'application/json'},*

*body: JSON.stringify(data)*

*})*

*.then(function(resp) {*

*return resp.json();*

*})*

*.then(function(resp) {*

*that.setState({pressureValue: resp.normalValue});*

*})*

*}*

Сервер на основе поступивших данных определяет нормативное для такого сочетания данных значение и отправляет его на клиент:

*getNormalPressure: (req, res) => {*

*let data = {*

*age: req.body.age,*

*gender: req.body.gender,*

*systolicBloodPressure: req.body.systolicBloodPressure,*

*diastolicBloodPressure: req.body.diastolicBloodPressure*

*};*

*let bpOnGender = bpMocks[data.gender];*

*let normalValue;*

*for (key in bpOnGender) {*

*if (data.age > key) {*

*normalValue = bpOnGender[key];*

*}*

*}*

*data.normalValue = normalValue;*

*res.send(data);*

*},*

## Модуль анализа значений артериального давления

Этот модуль представляет из себя форму и представлен на рисунке 3.7. На вход в данный модуль поступает информация о пользователе. После подтверждения формы эта информация отправляется на сервер для дальнейшего анализа:

*formSubmit(e) {*

*e.preventDefault();*

*let that = this;*

*let form = e.target;*

*let value = form[2].value + '/' + form[3].value;*

*let data = {*

*token: this.state.token,*

*age: form[0].value,*

*gender: form[1].value,*

*value: value,*

*date: +new Date(),*

*height: form[4].value,*

*weight: form[5].value,*

*physicalActivity: form[6].value,*

*alcohol: form[7].value,*

*smoke: form[8].value*

*};*

*let xhr = new XMLHttpRequest();*

*xhr.open('PUT', constants.addvalue);*

*xhr.setRequestHeader('Content-Type', 'application/json');*

*xhr.onreadystatechange = () => {*

*if (xhr.readyState === 4) {*

*that.getUserHistory();*

*}*

*};*

*xhr.send(JSON.stringify(data));*

*}*

На сервере происходит сохранение полученного значения в базу данных:

*apiRouter.put('/addvalue', function(req, res, next) {*

*let data = new pressureModel({*

*token: req.body.token,*

*value: req.body.value,*

*age: req.body.age,*

*gender: req.body.gender,*

*date: req.body.date,*

*height: req.body.height,*

*weight: req.body.weight,*

*physicalActivity: req.body.physicalActivity,*

*alcohol: req.body.alcohol,*

*smoke: req.body.smoke*

*});*

*data.save();*

*res.send();*

*});*

Затем, после того как сервер сообщил, что данные сохранены, клиент делает запрос за всей историей пользователя:

*getUserHistory() {*

*let that = this;*

*fetch(constants.gethistory + '/' + this.state.token,{*

*method: 'GET',*

*headers:{'content-type': 'application/json'}*

*})*

*.then(function(resp) {*

*return resp.json();*

*})*

*.then(function(resp) {*

*that.setState({history: resp});*

*})*

*}*

Сервер делает выборку по идентификатору пользователя и добавляет к этим значениям значение нормативного артериального давления для каждой записи, после чего передает историю пользователя для анализа:

*apiRouter.get('/gethistory/:token', function(req, res, next) {*

*let token = req.params.token;*

*pressureModel.find({'token': token}, (error, items) => {*

*if (items) {*

*let array = [];*

Ниже происходит определение нормативного значения артериального давления в зависимости от пола и возраста пользователя:

*items.forEach(function(item) {*

*let bpOnGender = bpMocks[item.gender];*

*let normalValue;*

*let obj;*

*for (key in bpOnGender) {*

*if (item.age > key) {*

*normalValue = bpOnGender[key];*

*}*

*}*

Затем формируется массив объектов, который и будет отправлен на клиент:

*obj = {*

*token: item.token,*

*value: item.value,*

*age: item.age,*

*gender: item.gender,*

*date: item.date,*

*height: item.height,*

*weight: item.weight,*

*physicalActivity: item.physicalActivity,*

*alcohol: item.alcohol,*

*smoke: item.smoke,*

*normalValue: normalValue,*

*};*

*obj.rejectionReasons = apiController.analyzeRejectionReasons(obj);*

*array.push(obj);*

*});*

*res.send(array);*

*} else {next();};*

*});*

*});*

Ниже представлена функция, анализирующая записи истории пользователя и формирующая массив рекомендаций и диагнозов пользователя:

*let reasonsArray = [];*

*let systolicPressureValue = obj.value.split('/')[0];*

*let dyastolicPressureValue = obj.value.split('/')[1];*

*let systolicNormalPressureValue = obj.normalValue.split('/')[0];*

*let dyastolicNormalPressureValue = obj.normalValue.split('/')[1];*

*if (systolicPressureValue && dyastolicPressureValue) {*

Ниже представлена часть функции, анализирующая введенные пользователеи значения веса и роста:

*if (obj.height && obj.weight) {*

*let weight = obj.weight / (obj.height / 100 \* obj.height / 100);*

*if (weight < 18.5) {*

*reasonsArray.push('Recommend to gain weight');*

*} else if (weight > 25 && weight < 30) {*

*reasonsArray.push('Recommend to lose weight');*

*} else if (weight > 30) {*

*reasonsArray.push('Extremely recommend to lose weight');*

*}*

*}*

Следующий код отвечает за анализ введенного типа активности пользователя:

*if (obj.physicalActivity) {*

*if (obj.physicalActivity === 'notactive') {*

*reasonsArray.push('Try to increase your physical activity');*

*}}*

Код ниже отвечает за анализ введенного пользователем типа употребления алкоголя:

*if (obj.alcohol) {*

*if (obj.alcohol === 'normal') {*

*reasonsArray.push('Reduction in the amount of alcohol consumed can contribute to better health');*

*} else if (obj.alcohol === 'active') {*

*reasonsArray.push('You need to reduce the amount of alcohol consumed.');*

*}}*

Ниже происходит анализ введенного отношения пользователя к курению:

*if (obj.smoke) {*

*if (obj.smoke === true) {*

*reasonsArray.push('Smoking can cause pressure deterioration');*

*}}*

Ниже происходит анализ введенных пользователем значений систолического и дистолического давления:

*if (systolicNormalPressureValue && dyastolicNormalPressureValue) {*

*if (*

*(systolicNormalPressureValue - systolicPressureValue) > 20 &&*

*(dyastolicNormalPressureValue - dyastolicPressureValue) > 20*

*) {*

*reasonsArray.push('Try to avoid stress conditions');*

*reasonsArray.push('Try to avoid taking various stimulating substances');*

*}}*

*if (systolicPressureValue > 140) {*

*reasonsArray.push('There is a risk of cardiovascular disease.');*

*}*

Следующий код анализирует разницу между систолическим и дистолическим значениями давления пользователя:

*if ((systolicPressureValue - dyastolicPressureValue) > 55) {*

*reasonsArray.push('You have a significant difference between the upper and lower pressure, which may indicate a malfunction of the digestive system, lesions of the gallbladder, ducts, and tuberculosis.');*

*} else if ((systolicPressureValue - dyastolicPressureValue) < 30) {*

*reasonsArray.push('You have a lower difference in upper and lower pressure, which can lead to hypoxia, atrophic changes in the brain, impaired vision, respiratory paralysis, cardiac arrest. Seek medical attention immediately.');*

*}}*

Результатом работы модуля служит массив рекомендаций, сформированный в соответствии с входными значениями:

*return reasonsArray;*

После чего проанализированные данные отправляются на клиент, который в свою очередь приступает к их отображению. Для визуализации графика отклонений артериального давления от нормативных значений с течением времени использовалась библиотека *react-chartjs-2*, которая имеет широкий спектр геометрических возможностей.

## Модуль отображения выходных данных

Данный модуль не имеет какой-либо смысловой нагрузки с точки зрения логики, за исключением функции формирования графика:

*prepareDataForSchedule() {*

*let labelsArray = [];*

*let systolicPressureValues = [];*

*let dyastolicPressureValues = [];*

*let systolicNormalPressureValues = [];*

*let dyastolicNormalPressureValues = [];*

*this.state.history.forEach(function(item) {*

*labelsArray.push(item.date);*

*systolicPressureValues.push(item.value.split('/')[0]);*

*dyastolicPressureValues.push(item.value.split('/')[1]);*

*systolicNormalPressureValues.push(item.normalValue.split('/')[0]);*

*dyastolicNormalPressureValues.push(item.normalValue.split('/')[1]);*

*});*

*return {*

*labels: labelsArray,*

*datasets: [*

*{*

*label: 'systolic blood pressure',*

*data: systolicPressureValues*

*},{*

*label: 'diastolic blood pressure',*

*data: dyastolicPressureValues*

*},{*

*label: 'systolic normal blood pressure',*

*data: systolicNormalPressureValues*

*},{*

*label: 'diastolic normal blood pressure',*

*data: dyastolicNormalPressureValues*

*}*

*]};*

*}*

Он также разбирает пришедшие от сервера данные:

*if (this.state.history) {*

*reasons = this.state.history[this.state.history.length-1].rejectionReasons;*

*data = this.prepareDataForSchedule();*

*items = this.state.history.map(item => <li>{item.value} (Normal value:{item.normalValue})</li>);*

*rejectionReasons = <ul>{reasons.map(item => <li>{item}</li>)}</ul>*

*}*

Результатом работы модуля служит текстовое и визуальное отображение выходных данных.

Данный модуль изображен на рисунке 3.8.

## Модуль рассылки уведомлений

Данный модуль предназначен для рассылки уведомлений о изменениях атмосферного давления, которые могут повлиять на самочувствие пользователей. Вся логика данного модуля размещена на сервере, так как необходимо постоянное функционирование данного модуля и постоянная связь с базой данных. Для формирования и рассылки писем используется библиотка *Nodemailer.* Для начала происходит установка таймера, который будет вызывать функцию колбэк раз в сутки в определенное время:

*setInterval(function () {*

*var date = new Date();*

*if (date.getHours() === 7 && date.getMinutes === 0) {*

*apiController.sendNotifications()*

*};*

*}, 3600000)*

Функция колбэк содержит следующий код:

*sendNotifications() {*

*usersModel.find((error, users) => {*

*users.forEach(user => {*

*if(user.username.match( /@/ig ) && user.city) {*

*let testAccount = await nodemailer.createTestAccount();*

*let xhr = new XMLHttpRequest();*

*xhr.open('GET', 'http://api.apixu.com/v1/current.json?key=d9fe708ea76e4ad6977111820191705&q=' + user.city);*

*xhr.onreadystatechange = (resp) => {*

*let pressureValue = resp.condition.pressure\_mb / 133.466;*

*let text;*

*if (pressureValue > 770 || pressureValue < 740) {*

*text = 'Be careful, there is increased pressure, with high atmospheric pressure, weather-sensitive people should behave more passively, avoid physical exertion, measure blood pressure more often and consult a doctor if necessary. Additional medications should not be taken, however, with an increase in blood pressure, you can resort to short-acting drugs.';*

*} else if (pressureValue < 740) {*

*text = 'Be careful, there is a reduced atmospheric pressure, with low atmospheric pressure, weather-sensitive people should behave more passively, avoid physical exertion, measure blood pressure more often and, if necessary, consult a doctor. Additional medications should not be taken, however, with an increase in blood pressure, you can resort to short-acting drugs.';*

*}*

*if (text) {*

*let transporter = nodemailer.createTransport({*

*host: "smtp.ethereal.email",*

*port: 587,*

*secure: false,*

*auth: {*

*user: testAccount.user,*

*pass: testAccount.pass*

*}*

*});*

*await transporter.sendMail({*

*from: 'mimohojij@mail-finder.net',*

*to: user.username,*

*subject: "weather",*

*text: text,*

*});*

*}*

*};*

*xhr.send(JSON.stringify(data));*

*}*

*});*

*});*

*}*

## Руководство пользователя

Рассмотрим работу программного комплекса на рабочем примере. Рассмотрим пример работы для авторизованного пользователя. Первоначально пользователь попадает на главный экран неавторизованного пользователя, что отражено на рисунках 3.5 и 3.6.



Рисунок 3.5 – Форма определения нормативного давления



Рисунок 3.6 – Форма определения нормативного давления с выходными данными

Затем пользователю необходимо авторизоваться, либо зарегистрироваться если аккаунт отсутствует и затем авторизоваться, что изображено на рисунке 3.4. Данные шаги отражены на рисунках 3.2 и 3.3.



Рисунок 3.3 – Форма регистрации новых пользователей с сообщением о ошибке



Рисунок 3.2 – Форма регистрации новых пользователей



Рисунок 3.4 – Форма авторизации пользователей

Если авторизация прошла успешно, то пользователь попадает в личный кабинет, где имеется форма для введения данных. Данная форма представлена на рисунке 3.7.



Рисунок 3.7 – Модуль анализа значений артериального давления

После введения данных пользователь отправляет данные на сервер, затем полученный от сервера ответи визуализируется и пользователь может просмотреть полученные данные. Визуализация результирующих данных представлена на рисунке 3.8.



Рисунок 3.8 – Модуль отображения выходных данных

Затем чтобы выйти из системы пользователю необходимо вернуться на начальную страницу приложения и нажать кнопку *logout*.

# ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБСНОВАНИЕ РАЗРАБОТКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ПОЛОЧНЫМ ПРОСТРАНСТВОМ СУПЕРМАРКЕТА

## Характеристика программного продукта

Программный комплекс анализа артериального давления является сервисом, позволяющим пользователям собирать статистику об изменениях артериального давления, отслеживать их динамику, получать оповещения и рекомендации о возможных изменениях артериального давления. Данный программный комплекс собирает данные людей различных категорий, с различным образом жизни, анализирует их, пытается найти корреляцию между различными показателями, наподобие погоды и атмосферного давления.

Разработка и внедрение данного комплекса позволит:

* Производить мониторинг состояния артериального давления на протяжении длительного времени;
* Создать удобный инструмент для людей, заботящихся о своем артериальном давлении;
* Собрать базу данных с показателями артериального давления людей различных категорий и в различных условиях, что может быть полезно с точки зрения медицины и здравоохранения.

Целесообразность инвестиций в разработку и использование программного продукта осуществляется на основе социальной значимости программного комплекса, которая заключается в необходимости наличия инструмента анализа и мониторинга показателей артериального давления, создания базы данных показателей артериального давления различных людей, а также возможной перспективы размещения рекламных банеров на сайте.

В результате разработки и внедрения данного программного комплекса будет разработан и внедрен инструмент для сбора статистики и мониторинга изменений показателей состояния артериального давления при влиянии различных внешних факторов.

## 4.2 Оценка трудоемкости и сроков разработки

Объем ПК определяется на основе нормативных данных, приведённых в таблице 4.1.

Таблица 4.1. – Характеристики функций и их ПК

|  |  |  |
| --- | --- | --- |
| Номер функции | Содержание функции | Объем функций (условных машинных команд) |
| 101 | Организация ввода информации | 150 |
| 203 | Формирование баз данных | 2180 |
| 204 | Обработка наборов и записей базы данных | 2670 |
| 707 | Графический вывод результатов | 480 |
| Всего | | 5480 |

Общий объем ПК рассчитывается по формуле:

|  |  |
| --- | --- |
|  | (4.1) |

где *VO* – общий объем ПК;

*Vi* – объем функций ПК;

*n* – общее число функций.

Таким образом, получаем объем данного программного средства:

ПК имеет 2 группу сложности, и, следовательно, нормативная трудоемкость составит:

 человеко-дней.

На основании нормативной трудоемкости с учетом дополнительного коэффициента сложности КСЛ рассчитываем общую трудоемкость программного комплекса:

|  |  |
| --- | --- |
| , | (4.2) |

где То – общая трудоемкость ПК;

Тн – нормативная трудоемкость ПК;

Ксл – дополнительный коэффициент сложности ПК.

 человеко-дней.

Определим уточнённую трудоёмкость. Она определяется следующим образом:

|  |  |
| --- | --- |
| , | (4.3) |

где Ту – трудоемкость разработки ПК на *i*-й стадии (технического задания, эскизного проекта, технического проекта, рабочего проекта и внедрения);

Кн – поправочный коэффициент, учитывающий степень новизны программного комплекса;

Кт – поправочный коэффициент, учитывающий степень использования в разработке типовых программ и ПК.

Разрабатываемое ПК является принципиально новым ПК ВТ, не имеющим доступных аналогов, при этом не используется нового типа ЭВМ или ОС, поэтому отнесем его к группе «А» с коэффициентом новизны 1,0. Степень охвата реализуемых функций разрабатываемого ПК ВТ типовыми программами и ПК ВТ составляет КТ = 0,9.

Исходя из степени новизны, определим коэффициенты удельных весов трудоемкости на каждой стадии:

Техническое задание:

Технический проект:

Рабочий проект:

Внедрение:

Рассчитаем трудоемкость по стадиям:

 человеко-дней,

человеко-дней,

 человеко-дней,

 человеко-дней,

Уточненная трудоемкость будет равна:

 человеко-дней.

Численность исполнителей проекта() рассчитывается по формуле

где − годовой эффективный фонд времени работы одного работника, (дн.);

− общая трудоемкость разработки проекта, (чел./дн.);

−срок разработки проекта, (лет).

Эффективный фонд времени работы одного работника составит:

дн.

Численность разработчиков программного комплекса составит:

Разработкой программного комплекса анализа артериального давления занимаются 2 человека.

## 4.3 Расчет затрат на разработку программного комплекса

Таблица 4.2 – Расчет основной зарплаты научно-технического персонала

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исполнители | Коли-чество | Трудоем-кость, дн. | Среднедневная заработная плата, руб. | Заработная плата, руб. |
| Инженер-программист | 2 | 87 | 50 | 4350 |
| Всего | | | | 4350 |

Дополнительная зарплата (Зд) определяется следующим образом:

|  |  |
| --- | --- |
|  | (4.4) |
| где HД –  норматив дополнительной заработной платы основных производственных рабочих, %. |  |

Отчисления на социальные нужды (Рсоц) определяются следующим образом:

|  |  |
| --- | --- |
|  | (4.5) |

где Hсоц – страховые взносы на обязательное страхование наймных работников (34%) и обязательное страхование от несчастных случаев на производстве (0,6%).

Расходы по статье «Машинное время» (Рм) определяются по формуле:

|  |  |
| --- | --- |
|  | (4.6) |

где Цм – цена одного часа машинного времени (0,7руб);

*Vо* –  общий объём программного средства (строк исходного   
кода) ();

Нмв – норматив расхода машинного времени на откладку 100 строк исходного кода, 12%.

Накладные расходы (Рнакл) определяются следующим образом:

|  |  |
| --- | --- |
|  | (4.7) |

где Ннакл – норматив накладных расходов (80 %)

Общая сумма затрат на разработку программного комплекса составит:

руб

## Расчёт экономической эффективности реализации программного комплекса анализа артериального давления

Программный комплекс планируется разместить в сети Интернет со свободным доступом. Исходя из социального значения программного комплекса в сфере здравоохранения, которое заключается в необходимости инструмента анализа и мониторинга артериального давления. В современном обществе огромное количество людей страдает от гипертонии или гипотонии, при том, что многие из них даже не догадываются об этом. По словам главного кардиолога РНПЦ кардиологии, около 40% белорусов страдают от повышенного давления, и это огромные цифры. Кроме того, остро стоит не только вопрос самого давления, а также возможных последствий его отклонения, так в связи с ухудшающейся экологической обстановкой а также повсеместным снижением иммунитета младших поколений крайне распространены заболевания, прямо или косвенно связанные с отклонениями артериального давления организма. Соответственно, не предполагается извлечения какого-либо дохода из реализации программного комплекса, за исключением возможного размещения рекламных банеров по бокам от основного контента сайта, так, чтобы это не припятствовало использованию сайта.

Согласно исследованиям по аналитике баннерной рекламы сайта, наибольшую эффективность будет иметь реклама по левой стороне от контента и ее эффективность будет падать по направлению взгляда сверху вниз.

## Результат технико-экономического обоснования разработки программного продукта

В результате технико-экономического обоснования разработки программного комплекса были получены следующие значения показателей их эффективности: затраты на разработку ПК составят 10380,93 рублей.

Однако, исходя из большого социального значения программного комплекса, разработка ПК является эффективной.

**ЗАКЛЮЧЕНИЕ**

В процессе работы над дипломным проектом был разработан программный комплекс, позволяющий анализировать артериальное давление, предназначенный для помощи людям, имеющим проблемы с артериальным давлением, либо просто следящи за состоянием собственного здоровья.

В процессе работы над дипломным проектом были найдены аналоги в сфере анализа артериального давления и проведен их анализ. Были определены их сильные и слабые стороны.

В ходе выполнения дипломной работы были выполнены следующие задачи:

* + - разработать структуру программного комплекса анализа артериального давления;
    - спроектировать базу данных;
    - разработать алгоритмы функционирования программного комплекса;
    - разработать интерфейс в соответствии с требованиями эргономики;
    - реализовать программный продукт;
    - провести технико-экономическое обоснование разработки проекта.

Также были реализованы следующие функции автоматизированной системы:

* регистрация и авторизация пользователей;
* ввод данных;
* анализ введенных данных о артериальном давлении;
* вывод результатов анализа.

Программный комплекс представляет из себя клиент-серверное приложение, анализирующее входные данные артериального давления и сохранящее данные о пользователях и введенными ими данными в базе данных.

Было проведено технико-экономическое обоснование разработки программного комплекса, которое показало целесообразность его разработки с экономической и социальной точки зрения.

Цели, поставленные в дипломном проекте, были достигнуты в полном размере.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Виды и типы магазинов розничной торговли [Электронный ресурс]. – Режим доступа : https://litstile.ru/index.php?ht=1674.

[2] Система управления полочным пространством магазина, автоматизация управления выкладкой [Электронный ресурс] – Режим доступа : https://abmcloud.com/abm-soft/shelf/

[3] RS.ShelfSpace: система управления полочным пространством магазина, автоматизация мерчандайзинга, контроль выкладки товаров [Электронный ресурс] – Режим доступа : https://shelfspace.ru/.

[4] Документация Nodemailer [Электронный ресурс] – Электронные данные. – Режим доступа : https://nodemailer.com/

[5] Документация React [Электронный ресурс] – Электронные данные. – Режим доступа : https://ru.reactjs.org/docs/

[6] Независимая библиотека википедия [Электронный ресурс] – Электронные данные. – Режим чдоступа : https://ru.wikipedia.org

[7] Документация Redux [Электронный ресурс] – Электронные данные. – Режим доступа : https://redux.js.org/

[8] Экономическое обоснование проекта по разработке программного обеспечения: методическое пособие. В. Г. Шоровой. Минск : БГУИР, 2018.

[9] Батин, Н. В. Дипломное проектирование. Методическое пособие / Н. В. Батин, А. А. Навроцкий – Минск : БГУИР, 2018. – 65 с.

[10] Документация Node.js [Электронный ресурс] – Электронные данные. – Режим доступа : https://nodejs.org/

[11] Документация Express.js [Электронный ресурс] – Электронные данные. – Режим доступа : https://expressjs.com/

[12] 25-я центральная районная поликилиника города Минска [Электронный ресурс] – Электронные данные. – Режим доступа : http://www.25crp.by

[13] Правила измерения артериального давления [Электронный ресурс] – Электронные данные. – Режим доступа : belmapo.by

[14] Ломако, А. В. Электронный учебно-методический комплекс по дисциплине «Проектирование автоматизированных систем» / А. В. Ломако – Минск : БГУИР, 2016.

[15] Когаловский М.Р. Энциклопедия технологий баз данных/ М.Р. Когаловский – Москва : Финансы и статистика, 2002. – 800 с.

Приложение А  
(обязательное)  
Листинг моделей соединения с базой данных

*let UserSchema = new mongoose.Schema({*

*email: {*

*type: String,*

*unique: true,*

*index: true,*

*required: true*

*},*

*password: {*

*type: String,*

*required: true*

*},*

*city: {*

*type: String,*

*}*

*});*

*let pressureSchema = new mongoose.Schema({*

*token: {*

*type: String*

*},*

*value: {*

*type: String*

*},*

*age: {*

*type: Number*

*},*

*gender: {*

*type: String*

*},*

*date: {*

*type: Number*

*},*

*height: {*

*type: Number*

*},*

*weight: {*

*type: Number*

*},*

*physicalActivity: {*

*type: String*

*},*

*alcohol: {*

*type: String*

*},*

*smoke: {*

*type: Boolean*

*}*

*});*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Обозначение* | | | | | *Наименование* | *Дополнительные сведения* | | | | |
|  | | | | | *Текстовые документы* |  | | | | |
|  | | | | |  |  | | | | |
| *БГУИР ДП 1-53 01 02 01 010 ПЗ* | | | | | *Пояснительная записка* | *65 с.* | | | | |
|  | | | | | *Отзыв руководителя* |  | | | | |
|  | | | | | *Рецензия* |  | | | | |
|  | | | | |  |  | | | | |
|  | | | | | *Графические документы* |  | | | | |
|  | | | | |  |  | | | | |
| *ГУИР.000000.001 ПД* | | | | | *Организационная структура* |  | | | | |
|  | | | | | *программного комплекса* |  | | | | |
|  | | | | | *артериального давления* | *Формат А1* | | | | |
|  | | | | |  |  | | | | |
| *ГУИР.000000.002 ПЛ* | | | | | *Алгоритм неавторизованнго* |  | | | | |
|  | | | | | *пользователя* | *Формат А1* | | | | |
|  | | | | |  |  | | | | |
| *ГУИР.000000.003 ПД* | | | | | *Алгоритм авторизованного* |  | | | | |
|  | | | | | *пользователя* | *Формат А1* | | | | |
|  | | | | |  |  | | | | |
| *ГУИР.000000.004 ПЛ* | | | | | *Диаграмма декомпозиции* | *Формат А1* | | | | |
|  | | | | |  |  | | | | |
| *ГУИР.000000.005 ПЛ* | | | | | *Диаграмма вариантов* |  | | | | |
|  | | | | | *использования* | *Формат А1* | | | | |
|  | | | | |  |  | | | | |
| *ГУИР.000000.006 ПЛ* | | | | | *Скриншоты работы* |  | | | | |
|  | | | | | *программы* | *Формат А1* | | | | |
|  | | | | |  |  | | | | |
|  | | | | |  |  | | | | |
|  | | | | |  |  | | | | |
|  | | | | |  |  | | | | |
|  | | | | |  |  | | | | |
|  | | | | |  |  | | | | |
|  |  |  |  |  | *БГУИР ДП 1-53 01 02 06 013 Д1* | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| *Изм.* | *Л.* | *№ докум.* | *Подп.* | *Дата* | *Автоматизированная система управления полочным пространством супермаркета*  *Ведомость дипломного  проекта* | *Лит* | | | *Лист* | *Листов* |
| *Разраб.* | | *Колесников В.Г* |  |  |  | *Т* |  | *65* | *65* |
| *Провер.* | | *Арановский В.Л.* |  |  | *Кафедра ИТАС*  *гр. 620603* | | | | |
| *Т.контр.* | | *Боброва Т.С.* |  |  |
| *Н.контр.* | | *Ярмолик В.И.* |  |  |
| *Утв.* | | *Навроцкий А.А.* |  |  |