

Заведующему кафедрой
инструментального и прикладного
программного обеспечения (ИиППО)
Института информационных технологий (ИТ)
Болбакову Роману Геннадьевичу
От студента Федорова Дмитрия Андреевича

ФИО

группа
3 курс
курс

Контакт: +79162926917

Заявление

Прошу утвердить мне тему *курсовой работы* по дисциплине «Разработка серверных частей интернет-ресурсов» образовательной программы бакалавриата 09.03.04 (Программная инженерия) «Серверная часть веб-приложения «Страховая компания»

Приложение: лист задания на КР/КП в 2-ух экземплярах на двухстороннем листе (проект)

Подпись студента

/ Федоров

Д.А.

подпись

ФИО

Дата

Подпись руководителя

/

Стариковская Н.А.

подпись

Должность, ФИО

Дата



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ЗАДАНИЕ
на выполнение курсовой работы

по дисциплине: Разработка серверных частей интернет-ресурсов
по профилю: Разработка программных продуктов и проектирование информационных систем
направления профессиональной подготовки: Программная инженерия (09.03.04)

Студент: Федоров Дмитрий Андреевич

Группа: ИКБО-03-20

Срок представления к защите: 01.12.2022

Руководитель: Стариковская Надежда Анатольевна, к.т.н., доцент

Тема: Серверная часть веб-приложения «Страховая компания».

Исходные данные: используемые технологии: HTML5, CSS3, PHP, Visual Studio Code, SQL СУБД, наличие: межстраничной навигации, внешнего вида страниц. Использование паттерна проектирования MVC. Нормативный документ: инструкция по организации и проведению курсового проектирования СМКО МИРЭА 7.5.1/04.И.05-18.

Перечень вопросов, подлежащих разработке, и обязательного графического материала: 1. Провести анализ предметной области разрабатываемого веб-приложения. 2. Обосновать выбор технологий разработки веб-приложения. 3. Разработать архитектуру веб-приложения на основе выбранного паттерна проектирования. 4. Реализовать слой серверной логики веб-приложения с применением выбранной технологии. 5. Реализовать слой логики базы данных. 6. Разработать слой клиентского представления веб-приложения. 7. Создать презентацию по выполненной курсовой работе.

Руководителем произведён инструктаж по технике безопасности, противопожарной технике и правилам внутреннего распорядка.

Зав. кафедрой ИиППО: [подпись] /Р. Г. Болбаков/, «16» сентября 2022 г.

Задание на КР выдал: [подпись] /Н.А. Стариковская /, «16» сентября 2022 г.

Задание на КР получил: [подпись] /Д.А. Федоров/, «16» сентября 2022 г.

Федоров Д.А. Серверная часть веб-приложения «Страховая компания» / Курсовая работа по дисциплине «Разработка серверных частей интернет-ресурсов» профиля «Разработка и дизайн компьютерных игр и мультимедийных приложений» направления профессиональной подготовки бакалавриата 09.03.04 «Программная инженерия» (5-ий семестр) / руководитель доцент каф. ИиППО Стариковская Н.А. / кафедра ИиППО Института ИТ РТУ МИРЭА.

Целью курсовой работы является проведение анализа предметной области, формирование требований к разрабатываемой серверной части веб-приложения, определение набора инструментов, разработка архитектуры, а также оформление сопутствующей документации.

Fedorov D.A. The server part of the web application "Insurance Company" / Course work on the discipline "Development of server parts of Internet resources" profile "Development and design of computer games and multimedia applications" of the directions of professional training for bachelor's degree 09.03.04 "Software engineering" (5th semester) / Head, Associate Professor of the Department. IiPPO Starikovskaya N.A. /Department of the IiPPO, Institute of IT RTU MIREA.

The aim of the course work is to analyze the subject area, to formulate requirements for the server part of the web application being developed, to define a set of tools, to develop an architecture, as well as to design related documentation.

ГЛОССАРИЙ

SOLID-принципы – принципы единственной ответственности, открытости/закрытости, подстановки Барбары Лисков, разделения интерфейса и инверсии зависимостей.

MVC - схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер.

БД (база данных) - совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.

LLVM – (Low Level Virtual Machine) проект программной инфраструктуры для создания компиляторов и сопутствующих им утилит.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	6
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ	7
1.1 Описание предметной области	7
1.2 Выводы к разделу 1	9
2 ВЫБОР И ОБОСНОВАНИЕ ТЕХНОЛОГИЙ	10
2.1 Выбор языка программирования для серверной части	10
2.2 Выбор архитектуры приложения	11
2.3 Вывод к разделу 2	14
3 РАЗРАБОТКА АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ, НА ОСНОВЕ ВЫБРАННОГО ПАТТЕРНА	15
3.1 Паттерн MVC	15
3.2 Схема архитектуры MVC	16
3.3 Вывод к разделу 3	16
4 РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ	16
4.1 Структура серверной части	16
4.2 Процесс создания серверной части веб-приложения	17
4.2.1 Создание проекта и добавление зависимостей	17
4.2.2 Создание модели, сущностей и заполнение базы данных. Model, Entity	18
4.2.3 Создание контроллеров. Controller	23
4.2.4 Создание представлений. View	24
4.2.5 Итоговый вид	26
4.3 Вывод к разделу 4	30
ЗАКЛЮЧЕНИЕ	31
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	32
ПРИЛОЖЕНИЕ	34

ВВЕДЕНИЕ

В качестве темы курсовой работы была выбрана «Серверная часть веб-приложения “Страховая компания”».

Целью курсовой работы является создание рабочей архитектуры веб-приложения на выбранную тематику.

Актуальность темы «Страховая компания» обосновывается потребностью современного общества в быстрых способах получения информации.

Результатом проделанной курсовой работы станет разработанная серверная часть веб-приложения, соответствующая изучаемому материалу в данном семестре.

Для выполнения целей и задач курсовой работы будут выполнены следующие шаги:

1. Формирование требований посредством анализа предметной области и технологии разработки.
2. Создание архитектуры веб-приложения.
3. Разработка серверной части.

Для получения информации о продуктах будет использован сайт страховой компании straxitut.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Описание предметной области

Предметная область данной курсовой работы – исследование веб-ресурсов, которые представляют деятельность страховой компании. В данной части будут рассмотрены примеры сайтов страховой компании.

Примером сайта с хорошей реализацией является «Альфа страхование» [1] (Рисунок 1.1.1). На сайте представлены два раздела на тех кому нужны услуги “Частные лица” и “Юридические лица”. В каждом из этих разделов представлено множества разных услуг по страхованию.

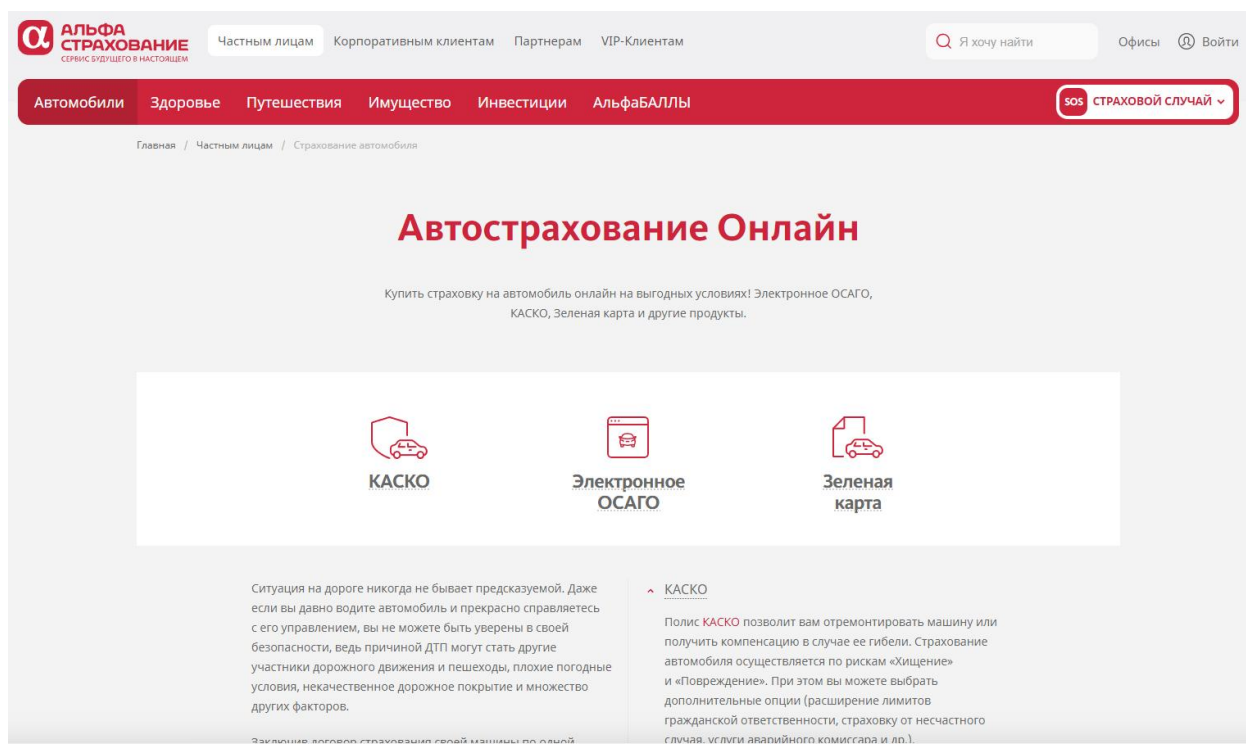


Рисунок 1.1.1 – Раздел "Главное меню" "Альфа страхование"

Также сайт страхования «Согласие страхование» [2] имеет приятный интерфейс и хорошую серверную часть, где, как и на сайте «Альфа страхование», реализованы меню и личный кабинет и несколько видов лиц: Физические и юридические лица. (Рисунок 1.1.2).

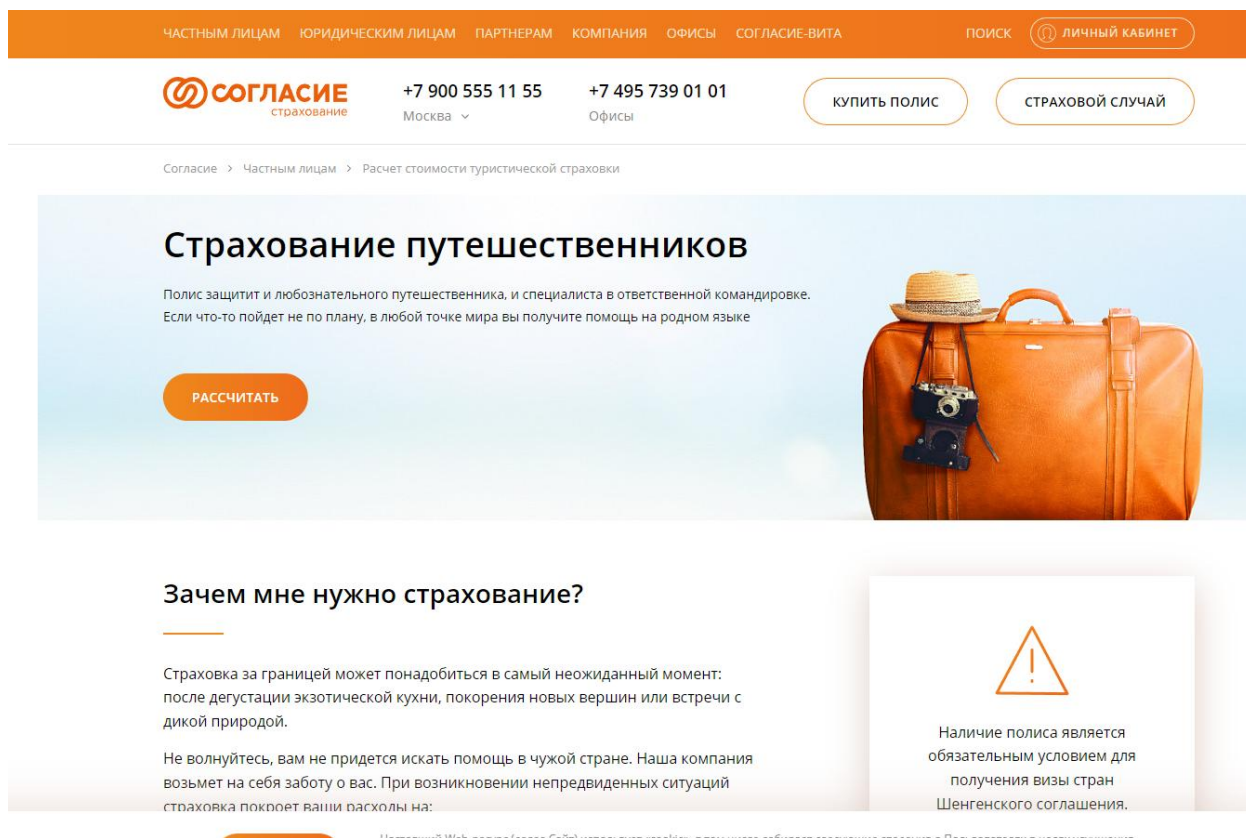


Рисунок 1.1.2 – Сайт «Согласие страхование»

Сайт страхования «Ренессанс» [3] не отличается по функционалу от других сайтов, описанных выше: есть личный кабинет, меню, уведомления об акциях (Рисунок 1.1.3).

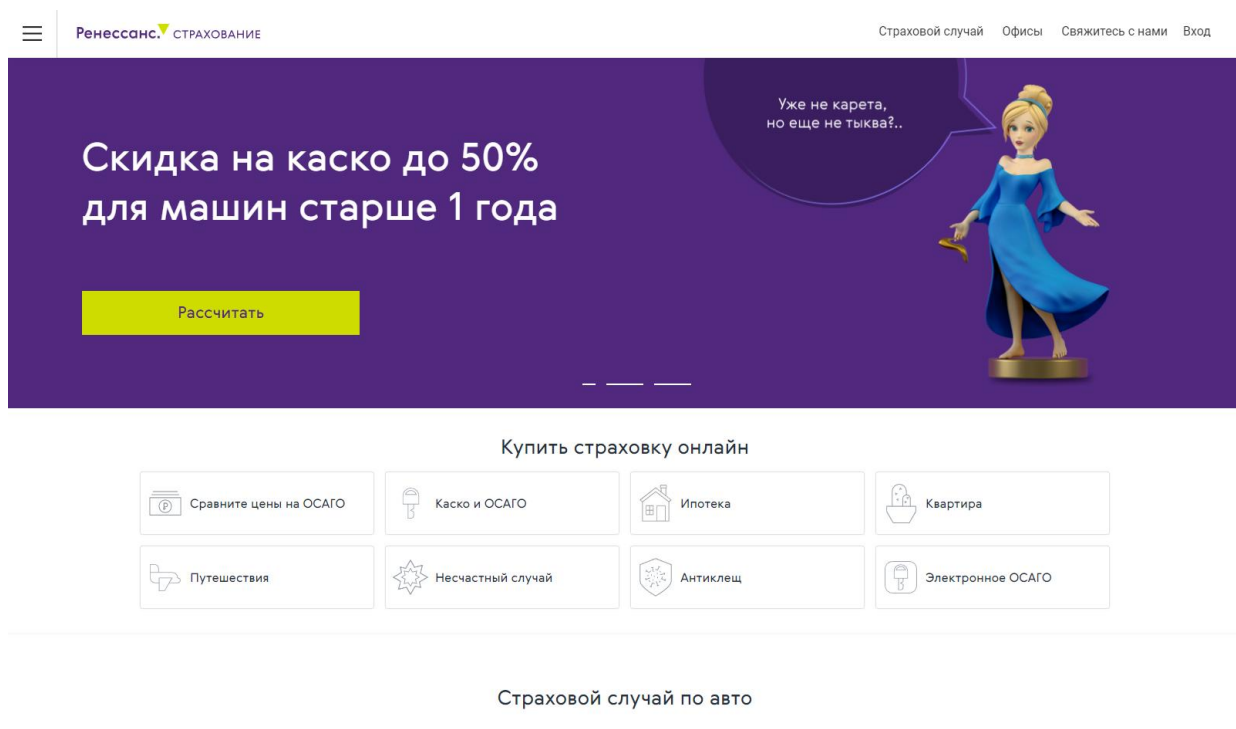


Рисунок 1.1.3 – Сайт "Ренессанс"

1.2 Выводы к разделу 1

В ходе анализа предметной области были выявлены плюсы и минусы сайтов одних из самых популярных сайтов страхования, в связи с этим были сформированы требования к будущей серверной части веб-приложения «Страховая компания»:

1. Наличие выбора на физические и юридические лица.
2. Наличие данных о заказе, в том числе его статуса.
3. Запись персональных данных клиента.

Следовательно, для реализации нам потребуется:

1. Создать несколько сущностей в базе данных: заказ, клиент, менеджер, услуги страхования.
2. Применение технологии для авторизации клиентов по отдельности (разделяя на юридических и физических лиц) и менеджеров.

2 ВЫБОР И ОБОСНОВАНИЕ ТЕХНОЛОГИЙ

2.1 Выбор языка программирования для серверной части

Для серверной части было рекомендовано выбрать один из трех языков программирования: PHP [4], Kotlin [5] или Java [6].

1. PHP

Это распространенный язык программирования общего назначения с открытым исходным кодом. PHP специально сконструирован для веб-разработок и его код может внедряться непосредственно в HTML.

2. Kotlin

Статически типизированный, объектно-ориентированный язык программирования, работающий поверх Java Virtual Machine и разрабатываемый компанией JetBrains. Также компилируется в JavaScript и в исполняемый код ряда платформ через инфраструктуру LLVM.

3. Java

Строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems. Разработка ведётся сообществом, организованным через Java Community Process; язык и основные реализующие его технологии распространяются по лицензии GPL.

Было решено использовать Java и ее известный фреймворк Spring [7], который упрощает разработку благодаря своего большому функционалу.

Помимо этого, стоит указать список технологий, которые также будут использованы в работе:

1. IntelliJ IDEA [8] – интегрированная среда разработки программного обеспечения для многих языков программирования от компании JetBrains.
2. Spring Data JDBC – модуль Spring, используется для того, чтобы предоставить доступ к реляционным базам данных без использования всей сложности JPA.
3. MySQL [9] – свободная система управления базами данных, в данной

курсовой работе нет необходимости работать с большими данными, но при этом нужна упорядоченность, поэтому реляционная база данных отлично подходит для этого.

4. Maven [10] – фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM.
5. Github [11] – репозиторий работ, позволяющий хранить проект удаленно в целях его мобильности, а также безопасности в случае удаления всех данных с локальной машины.
6. Visual Studio Code – интегрированная среда разработки программного обеспечения для многих языков программирования.

2.2 Выбор архитектуры приложения

В методических указаниях предлагается выбрать архитектуру веб-приложения из данных паттернов: DDD [12], Clean architecture [13] и MVC [14].

1. DDD

DDD (Рисунок 2.2.1) – это подход, который нацелен на изучение предметной области предприятия в целом или каких-то отдельных процессов. Этот подход подходит к проектам, сложность/запутанность достаточна велика. Бизнес-логика в данной курсовой работе не является запутанной, поэтому применение этого паттерна необоснованно.

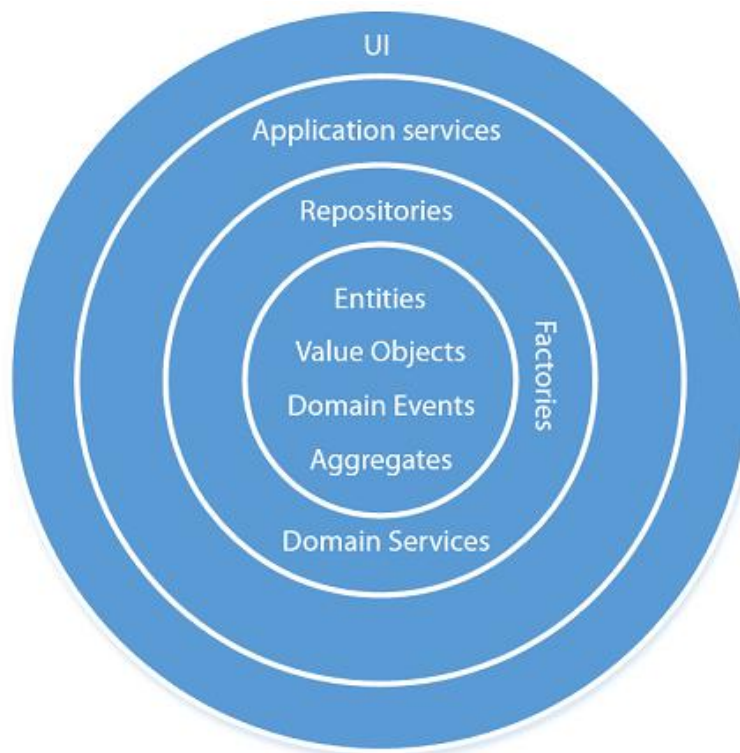


Рисунок 2.2.1 – Паттерн DDD

2. Clean architecture

Clean architecture (Рисунок 2.2.2) – это философия разработки программного обеспечения, которая разделяет элементы проекта на кольцевые уровни. Важная цель – предоставить разработчикам способ организовать код таким образом, чтобы он инкапсулировал бизнес-логику, но сохранял ее отдельно от механизма доставки. В этом подходе выделяется 4 слоя: Сущности, Логика приложения (Use Cases), Адаптеры и Фреймворки. Это лучшее решение для приложений с большим количеством функций и SOLID-принципами. Поэтому было выбрано решение использовать в курсовой работе данный паттерн проектирования.

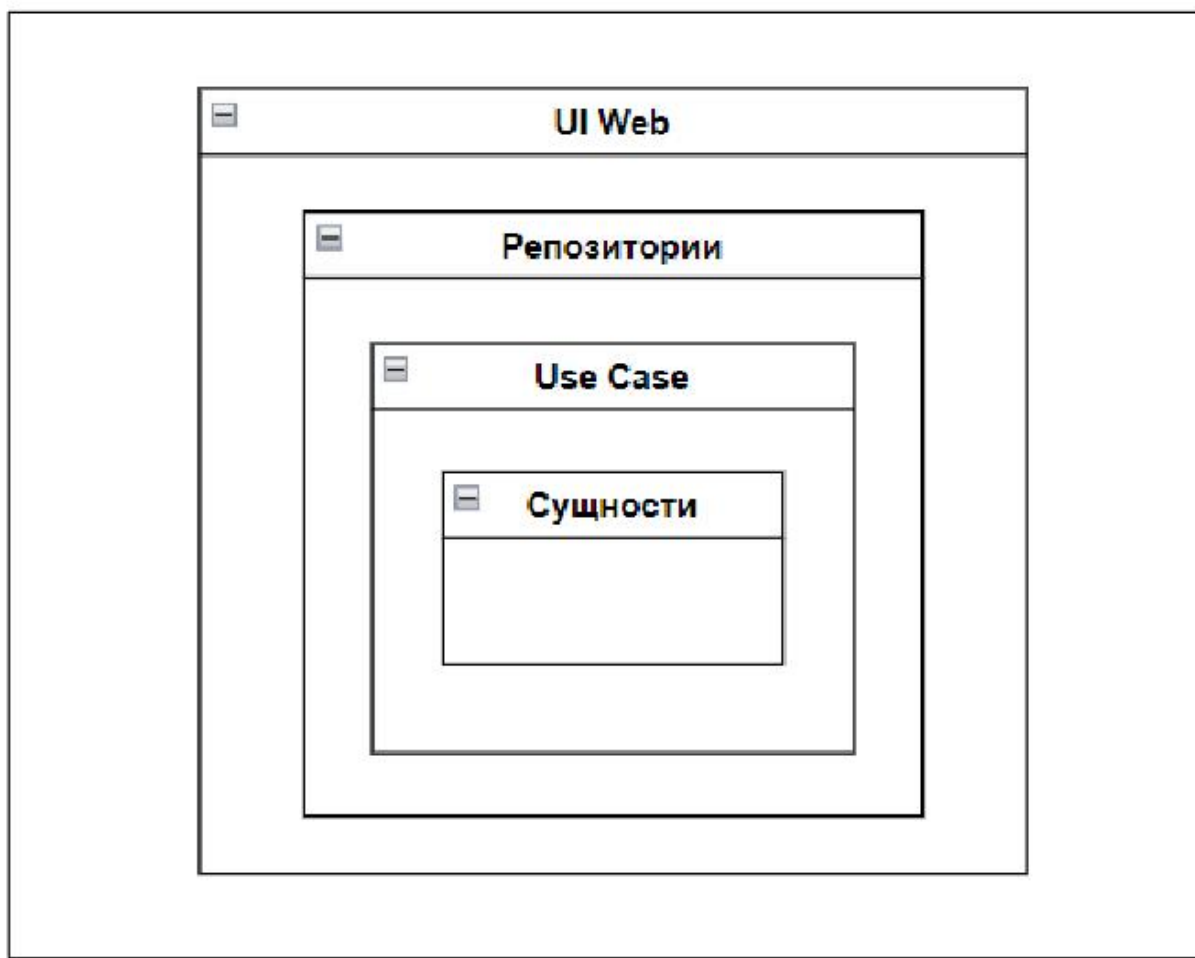


Рисунок 2.2.2 – Паттерн Clean architecture

3. MVC

MVC (Рисунок 2.2.3) – подход к проектированию приложения, который предполагает выделение кода в блоки модель, представление и контроллер. Контроллер обрабатывает входящие запросы. Модель достает из базы данных информацию, нужную для выполнения конкретных вопросов. Представление определяет результат запроса, который получает пользователь. Данный подход является простым, а также подходит для маленьких и простых работ. Однако данный паттерн предполагает обязательную клиентскую часть, поэтому он нам не подходит.

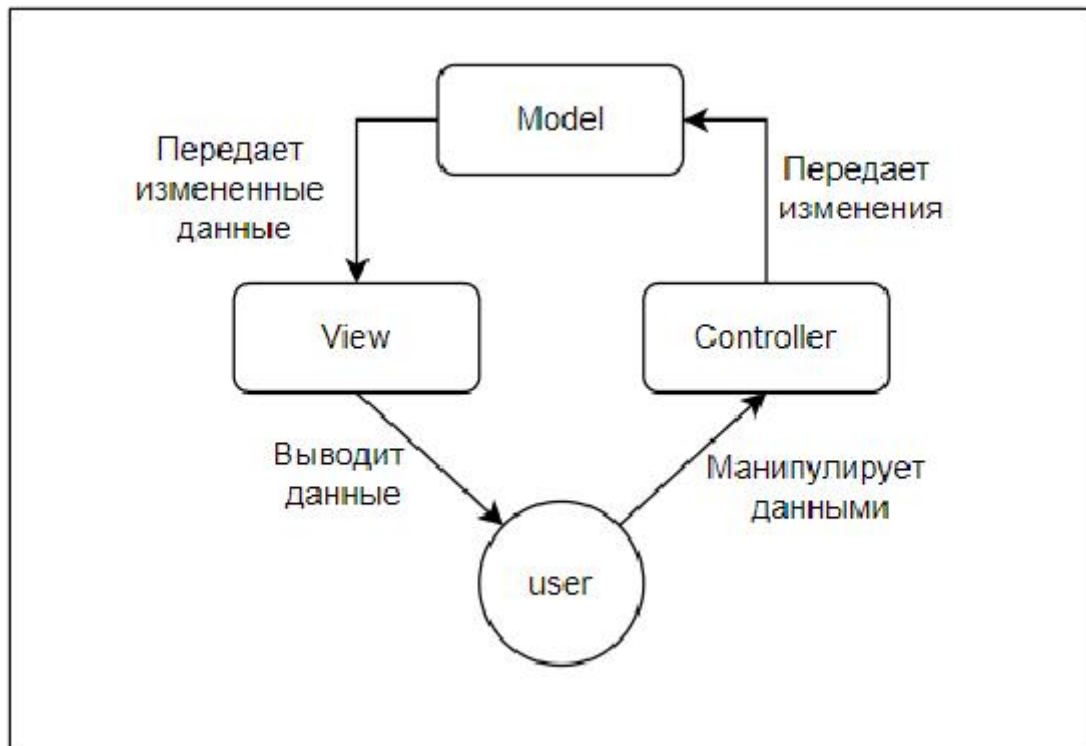


Рисунок 2.2.3 – Паттерн MVC

2.3 Вывод к разделу 2

Посредством анализа предложенных технологий и требований к курсовой работе, было принято решение выбрать следующие технологии:

- 1) PHP для серверной части.
- 2) MySQL для базы данных.
- 3) Паттерн проектирования MVC
- 4) Visual Studio Code для разработки

3 РАЗРАБОТКА АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ, НА ОСНОВЕ ВЫБРАННОГО ПАТТЕРНА

3.1 Паттерн MVC

MVC основывается на сепарации компонентов приложения по принципу Model-View-Controller, где Model предоставляет данные и реагирует на команды контроллера, изменяя своё состояние, View отвечает за отображение данных модели пользователю, реагируя на изменения модели, Controller интерпретирует действия пользователя, оповещая модель о необходимости изменений.

Моделью в курсовой работе будет база данных, состоящая из аккаунтов клиентов, включая разделение на физических и юридических лиц, доступных услуг, выбранных услуг и промежуточных таблиц для более легкой разработки.

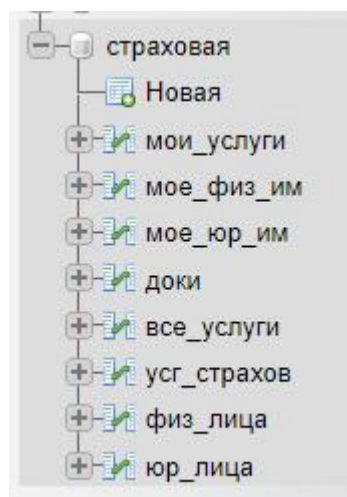


Рисунок 3.1.1 – Модель

Контроллерами будут выступать функции, изменяющие базу данных, а представлениями – HTML и CSS-код, представляющий интерфейс сайта.

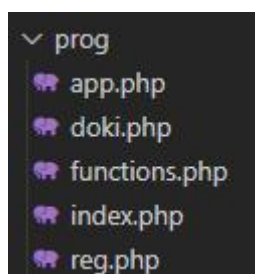


Рисунок 3.1.2 – Контроллеры и представления

3.2 Схема архитектуры MVC

Была составлена схема архитектуры MVC (Рисунок 3.2.1).

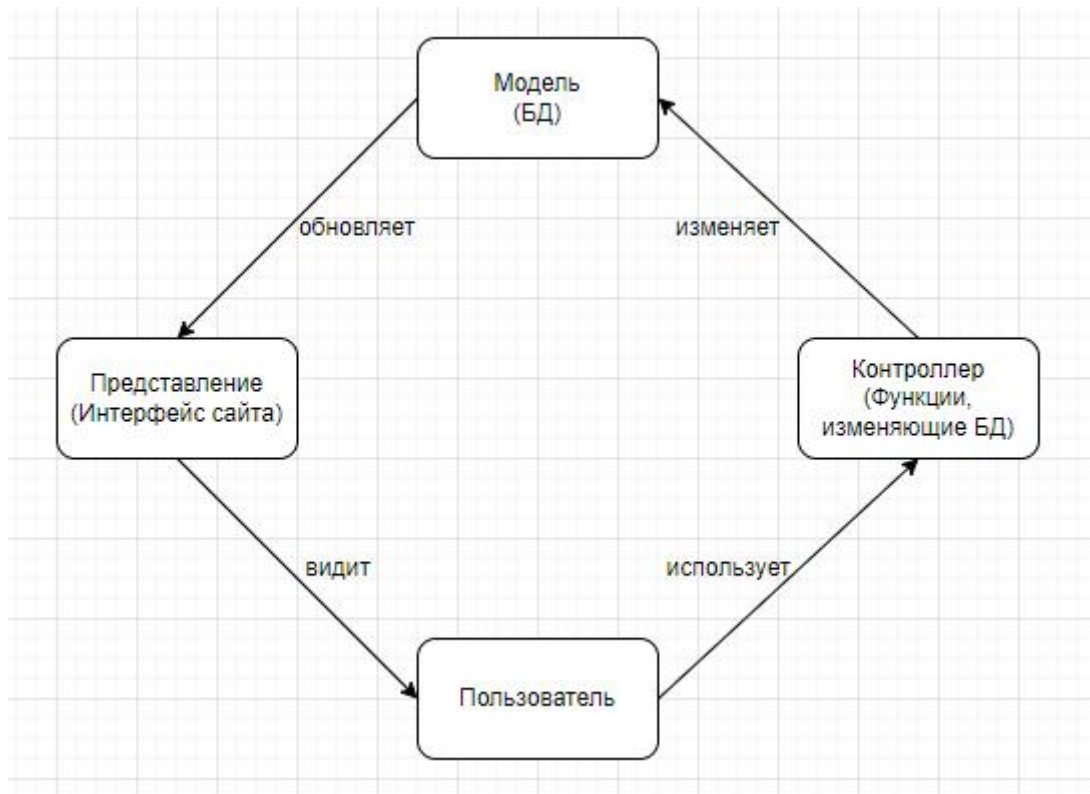


Рисунок 3.2.1 – Архитектура MVC

3.3 Вывод к разделу 3

Благодаря паттерну MVC, мы смогли добиться разделения проекта на части, что приведет к облегчению разработки.

4 РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ

4.1 Структура серверной части

Файлы серверной части, написанные с использованием PHP, CSS, MySQL.

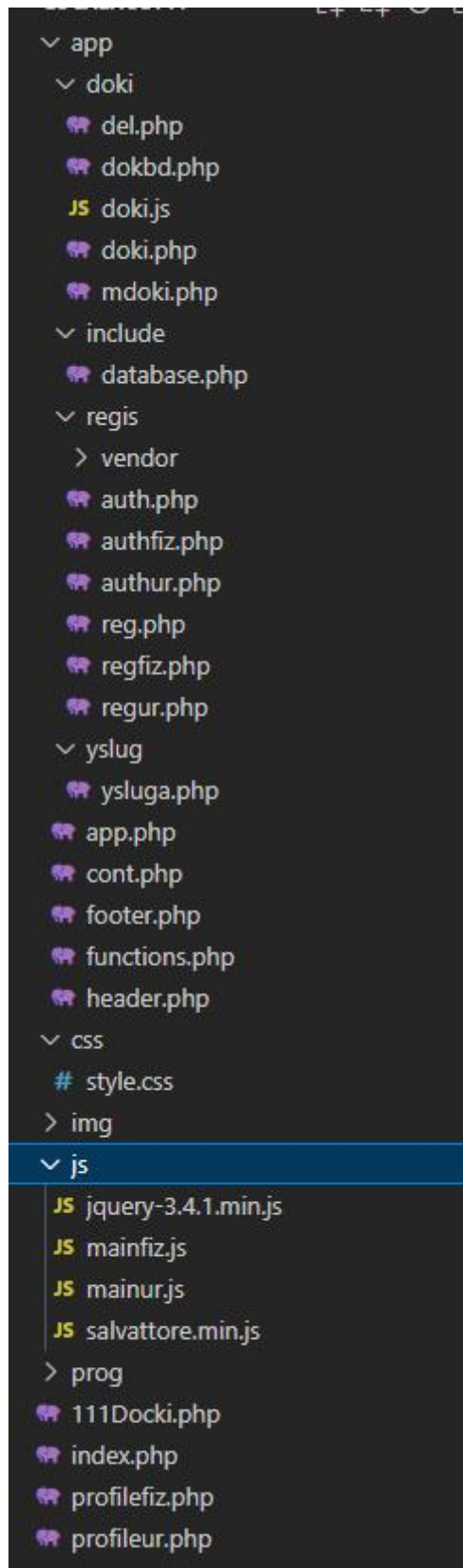


Рисунок 4.1.1 – Структура проекта

4.2 Процесс создания серверной части веб-приложения

4.2.1 Создание проекта и добавление зависимостей

Создаем новый проект, index.php (Листинг 1), включающий в себя:

```
<?php
    session_start();
    ini_set('error_reporting',E_ALL);
    ini_set('display_errors',1);
    ini_set('display_startup_errors',1);

    require 'app/header.php';
    ?>

    <div class="container">
        <div class="row">
            <div class="col-md-9">
                <div class="page-header">
                    <h1 class="mb-5">Все услуги:</h1>
                </div>
                <?php
                    $yslugi = get_yslugi();
                    ?>
                <?php foreach($yslugi as $yslug): ?>
                    <div class="col">
                        <div class="card mb-4 rounded-3 shadow-sm">
                            <div class="card-header py-3">
                                <h4 class="my-0 fw-
normal"><?=$yslug['Название']?></h4>
                            </div>
                            <div class="card-body">
                                <h1 class="card-title pricing-card-
title"><?=$yslug['Стоимость']?><small class="text-muted fw-light">
руб/год</small></h1>
                                <ul class="list-unstyled mt-3 mb-4">
                                    <li><?=$yslug['Описание']?></li>
                                    <p></p>
                                    <li>Условия к документам:
<?=$yslug['тип_документа']?></li>
                                </ul>
                                <button class="brat w-100 btn btn-lg btn-primary mb-
3" type="submit">Взять</button>
                            </div>
                        </div>
                    </div>
                <?php endforeach; ?>
            </div>
        </div>

    </div>
</div>

<?php require 'app/footer.php'; ?>
```

4.2.2 Создание модели, сущностей и заполнение базы данных. Model, Entity

После добавления зависимостей в проект можно начинать основную работу.

В начале работы осуществлялось параллельное создание и базы данных, и задание основных сущностей, их всего 5.

Сущность Юр_лица содержит следующие поля (Листинг 2):

- Id – номер пользователя.
- Название – название организации.
- ИНН – ИНН организации.
- email – email организации.
- пароль – пароль организации.

Листинг 2 – Юр_лица

```
CREATE TABLE `юр_лица` (  
  `id` int NOT NULL,  
  `название` varchar(65) NOT NULL,  
  `инн` bigint NOT NULL,  
  `юр_адресс` varchar(65) NOT NULL,  
  `телефон` bigint NOT NULL,  
  `email` varchar(65) NOT NULL,  
  `пароль` varchar(65) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;  
  
INSERT INTO `юр_лица` (`id`, `название`, `инн`, `юр_адресс`, `телефон`,  
`email`, `пароль`) VALUES  
  (1, 'ООО \"Иван\"', 7736050004, 'ул. 1999 дом 4 стр. 1', 89132826755,  
'MMM@MMM.ru', '202cb962ac59075b964b07152d234b70'),  
  (2, 'АНО \"Нужно построить ЗИККУРАТ\"', 7736055003, 'улица нужно больше  
золота дом 5', 89992422418, 'nerzul@yandex.ru',  
'202cb962ac59075b964b07152d234b70'),  
  (3, 'mmm', 123, 'ваыавы', 123145, 'mmm@mmm.mm',  
'202cb962ac59075b964b07152d234b70');
```

Сущность физ_лица содержит следующие поля (Листинг 3):

- Id – номер пользователя.
- фамилия – фамилия пользователя.
- имя – имя пользователя.
- отчество – отчество пользователя.
- инн – инн пользователя.
- почта – почта пользователя.
- телефон – телефон пользователя.
- адрес – адрес пользователя.
- пароль – пароль пользователя.

```

CREATE TABLE `физ_лица` (
  `id` int NOT NULL,
  `фамилия` varchar(65) NOT NULL,
  `имя` varchar(65) NOT NULL,
  `отчество` varchar(65) NOT NULL,
  `инн` bigint NOT NULL,
  `почта` varchar(65) NOT NULL,
  `телефон` bigint NOT NULL,
  `адрес` varchar(65) NOT NULL,
  `пароль` varchar(65) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

INSERT INTO `физ_лица` (`id`, `фамилия`, `имя`, `отчество`, `инн`,
`почта`, `телефон`, `адрес`, `пароль`) VALUES
(1, 'Икоб', 'Ика', 'Икобович', 7736050003, 'kakasha@yandex.ru',
89132826719, 'малая большая улица 65', '202cb962ac59075b964b07152d234b70'),
(2, 'Иванов', 'Иван', 'Иванович', 7736057703, 'ivan@yandex.ru',
89992926418, 'юго-западная или нет', '202cb962ac59075b964b07152d234b70'),
(3, 'Иванов', 'Ика', 'Икобович', 7736050003, 'fafd@fdf.rr', 89132826719,
'малая юго-западная', '202cb962ac59075b964b07152d234b70');

```

Сущность `усл_страхов` содержит следующие поля (Листинг 4):

- `strax_ID` – id записи.
- `individual` – возможный id физического лица.
- `lega` – возможный id юридического лица.
- `итог` – итоговая стоимость.
- `дата_начала` – дата начала.
- `дата_конца` – дата конца.

```

CREATE TABLE `усл_страхов` (
  `strax_ID` int NOT NULL,
  `individual` int DEFAULT NULL,
  `lega` int DEFAULT NULL,
  `итог` int NOT NULL,
  `дата_начала` date NOT NULL,
  `дата_конца` date NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

INSERT INTO `усл_страхов` (`strax_ID`, `individual`, `lega`, `итог`,
`дата_начала`, `дата_конца`) VALUES
(1, 1, NULL, 82500, '2022-12-05', '2023-12-05'),
(2, 2, NULL, 32500, '2022-12-06', '2023-12-06'),
(3, NULL, 1, 100000, '2022-12-06', '2023-12-06'),
(4, NULL, 2, 31000, '2022-12-04', '2023-12-04');

```

Сущность все_услуги содержит следующие поля (Листинг 5):

- Services_ID – номер услуги.
- Название – Название услуги.
- Описание – Описание услуги.
- Стоимость – Стоимость услуги.
- тип_документа – тип требуемого документа.

Листинг 5 – все_услуги

```
CREATE TABLE `все_услуги` (  
  `Services_ID` int NOT NULL,  
  `Название` varchar(64) NOT NULL,  
  `Описание` text NOT NULL,  
  `Стоимость` int NOT NULL,  
  `тип_документа` varchar(65) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;  
  
INSERT INTO `все_услуги` (`Services_ID`, `Название`, `Описание`,  
`Стоимость`, `тип_документа`) VALUES  
  (1, 'Страхование жизни', 'Страхование жизни и здоровья – это поддержание  
финансового благополучия семьи при наступлении несчастного случая с любым  
членом семьи и забота о здоровье взрослых, детей, спортсменов.', 32500,  
'паспорт'),  
  (2, 'Страхование недвижимого имущества физических лиц', 'Страхование  
имущества физических лиц – возможность обезопасить себя и своих близких от  
финансовых потерь в случае непредвиденных обстоятельств с вашей квартирой,  
коттеджем, дачей и ценными вещами в них.', 50000, 'кадастровый номер'),  
  (3, 'Страхование недвижимого имущества юридических лиц', 'Каждый день  
России в результате возгорания получают повреждения 270 зданий и сооружений.  
Порядка 120 строений гибнут от огня. Только в 2016 году сумма материального  
ущерба, причиненного пожарами на производственных площадках, складах и в  
торговых помещениях, составила свыше 10 млрд. рублей. Страхование позволит  
собственникам и арендаторам помещений защитить бизнес от непредвиденных  
расходов. Вы можете заключить договор, включающий комплекс рисков наряду с  
повреждением имущества в результате пожара.', 100000, 'кадастровый номер'),  
  (4, 'Страхование транспорта', 'Каждый день в России регистрируется свыше  
800 транспортных происшествий. Страхование защитит Ваше предприятие от ущерба,  
причиненного в результате аварий на автомобильном, водном, воздушном и ж/д  
транспорте.\n\nНаши страховые программы позволят не только избежать  
незапланированных расходов на ремонт транспорта, но и компенсировать вред,  
причиненный водителям, пассажирам, третьим лицам, пострадавшим в аварии.',  
21000, 'номер машины'),  
  (5, 'Страхование здоровья', 'Чтобы повысить свою финансовую защищенность,  
оформите обязательное и добровольное медицинское страхование. Будьте  
здоровы!', 10000, 'паспорт'),  
  (6, 'Страхование грузов', 'Программа страхования позволит защитить Ваш  
бюджет от расходов в случае потери или повреждения груза при транспортировке  
автомобильным, водным, воздушным или ж/д транспортом. При наступлении  
страхового случая мы возместим убытки и возьмем на себя урегулирование  
претензий. Страховая защита действует на территории России и за рубежом. Вы  
можете застраховать весь объем грузоперевозок или разовую перевозку грузов.',  
50000, 'номер документа'),  
  (7, 'Страхование ответственности юридических лиц', 'Страхование  
ответственности позволит Вам уверенно вести бизнес, если деятельность  
предприятия сопряжена с риском причинения вреда организациям или гражданам.
```

Мы разработали программы обязательного и добровольного страхования ответственности, позволяющие выбрать оптимальное страховое покрытие при возникновении экстренных ситуаций. Страховая компания «Согласие» обладает всеми необходимыми лицензиями и соответствует требованиям Национального союза страховщиков ответственности.', 24000, 'номер документа');

Сущность доки содержит следующие поля (Листинг 6):

- id_dok – номер документа.
- тип_документа – тип документа.
- номер_документа – номер документа.
- legal – возможный id юридического лица.
- individuals – возможный id физического лица.

Листинг 6 – доки

```
CREATE TABLE `доки` (  
  `id_dok` int NOT NULL,  
  `тип_документа` varchar(65) CHARACTER SET utf8mb4 COLLATE  
utf8mb4_0900_ai_ci NOT NULL,  
  `номер_документа` varchar(65) CHARACTER SET utf8mb4 COLLATE  
utf8mb4_0900_ai_ci NOT NULL,  
  `legal` int DEFAULT NULL,  
  `individuals` int DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;  
  
INSERT INTO `физ_лица` (`id`, `фамилия`, `имя`, `отчество`, `инн`,  
`почта`, `телефон`, `адрес`, `пароль`) VALUES  
(1, 'Икоб', 'Ика', 'Икобович', 7736050003, 'kakasha@yandex.ru',  
89132826719, 'малая большая улица 65', '202cb962ac59075b964b07152d234b70'),  
(2, 'Иванов', 'Иван', 'Иванович', 7736057703, 'ivan@yandex.ru',  
89992926418, 'юго-западная или нет', '202cb962ac59075b964b07152d234b70'),  
(3, 'Иванов', 'Ика', 'Икобович', 7736050003, 'fafd@fdf.rr', 89132826719,  
'малая юго-западная', '202cb962ac59075b964b07152d234b70');
```

Так же имеются связующие сущности таик как мои_услуги, которые содержат id на две таблицы, те которые нужно связать (листининг 7)

```
CREATE TABLE `мои_услуги` (  
  `Services` int NOT NULL,  
  `стоимость` int NOT NULL,  
  `Ind` int DEFAULT NULL,  
  `leg` int DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;  
  
INSERT INTO `мои_услуги` (`Services`, `стоимость`, `Ind`, `leg`) VALUES  
(1, 0, 1, 0),  
(2, 0, 1, 0),  
(1, 0, 1, 0),  
(3, 0, 0, 2),  
(4, 0, 0, 3),
```

```
(5, 0, 0, 4);  
COMMIT;
```

4.2.3 Создание контроллеров. Controller

После задания базы данных и создания сущностей для ее связи с прописываемым интерфейсом, следующим шагом является создание контроллеров. Примеры контроллеров представлены в листинге 7. Здесь представлены запросы POST, GET и INSERT.

Листинг 7 – Create Controller

```
<?php  
session_start();  
require_once '../include/database.php';  
require_once '../functions.php';  
mysqli_report(MYSQLI_REPORT_ERROR | MYSQLI_REPORT_STRICT);  
$doki = $_POST['doki'];  
$gos_n = $_POST['gos_n'];  
$idd=$_SESSION['user_fiz']['id'];  
$id=$_SESSION['user_ur']['id'];  
$check_doki=0;  
if ($idd>0)  
{  
    $sql = "SELECT * FROM доки WHERE individuals = $idd";  
    $check_doki = mysqli_query($link,$sql);  
    settype($idd, 'integer');  
    if (mysqli_num_rows($check_doki) > 0) {  
        $check_dok=$check_doki;  
        $check_doki = mysqli_query($link, "SELECT * FROM доки WHERE  
individuals = $idd and '$doki' = 'паспорт'");  
        if(mysqli_num_rows($check_dok)==mysqli_num_rows($check_doki))  
        {  
            $check_doki = 0;  
        }  
    }  
    else  
    {  
        $check_doki = 0;  
    }  
}  
if (mysqli_num_rows($check_doki)) {  
    $response = [  
        "status" => false,  
        "type" => 1,  
        "id" => $_SESSION['user_fiz']['id'],  
        "message" => "У вас уже есть этот документ",  
        "fields" => ['dok']  
    ];  
    echo json_encode($response);  
    die();  
}  
$error_fields = [];  
if ($gos_n === '') {  
    $error_fields[] = 'gos_n';  
}  
if (!empty($error_fields)) {  
    $response = [  

```

```

        "status" => false,
        "type" => 1,
        "message" => "Проверьте правильность полей",
        "fields" => $error_fields
    ];
    echo json_encode($response);
    die();
}
if ($_SESSION['user_fiz']['id']){
    mysqli_query($link, "INSERT INTO `доки`(`id_dok`, `тип_документа`,
`номер_документа`, `legal`, `individuals`) VALUES
(NULL, '$doki', '$gos_n', NULL, '$idd')");
    $response = [
        "status" => true,
        "message" => "Регистрация прошла успешно!",
    ];
    echo json_encode($response);
}
if ($_SESSION['user_ur']['id']){
    mysqli_query($link, "INSERT INTO `доки`(`id_dok`, `тип_документа`,
`номер_документа`, `legal`, `individuals`) VALUES
(NULL, '$doki', '$gos_n', '$id', NULL)");
    $response = [
        "status" => true,
        "message" => "Регистрация прошла успешно!",
    ];
    echo json_encode($response);
}
?>

```

4.2.4 Создание представлений. View

Далее были созданы представления. Данная часть кода является интерфейсом сайта и именно она видна пользователю. Примеры представлений в листинге 8. Для представления был использован язык разметки HTML вместе с php и css.

Листинг 8 – Index View

```

<?php
    session_start();
    require_once '../include/database.php';
    require_once '../functions.php';
?>
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <meta http-equiv="X-US-Compatible" content="ie=edge">
        <link rel="stylesheet" href="/css/style.css">
        <link rel="stylesheet"
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css
">
        <title>Strax</title>

```



```

        </head>
        <body>
        <div class="container">
            <header class="d-flex flex-wrap align-items-center justify-content-
center justify-content-md-between py-3 mb-4 border-bottom">
                <a href="/index.php" class="d-flex align-items-center col-md-3 mb-
2 mb-md-0 text-dark text-decoration-none">

                    
                    </a>

                <ul class="nav col-12 col-md-auto mb-2 justify-content-center mb-
md-0">
                    <li><a href="/index.php" class="nav-link px-2 link-
secondary">Главная</a></li>
                    <li><a href="/app/cont.php" class="nav-link px-2 link-
dark">Контакты</a></li>
                </ul>

                <div class="col-md-3 text-end">
                    <?php if ($_SESSION['user_fiz']['fam']): ?>
                        <button type="button" class="btn btn-primary"
onclick="document.location='/profilefiz.php'">Профиль <?php echo
$_SESSION['user_fiz']['fam']?> <?php echo $_SESSION['user_fiz']['nam'] ?>
<?php echo $_SESSION['user_fiz']['och'] ?>
                        </button>
                    <?php elseif ($_SESSION['user_ur']['ema'] != ''): ?>
                        <button type="button" class="btn btn-primary"
onclick="document.location='/profileur.php'">Профиль <?php echo
$_SESSION['user_ur']['organiz']?> <?php echo $_SESSION['user_ur']['ema'] ?>
                        </button>
                    <?php else: ?>
                        <button type="button" class="btn btn-outline-primary me-2"
onclick="document.location='/app/regis/auth.php'">Войти</button>
                        <button type="button" class="btn btn-primary"
onclick="document.location='/app/regis/reg.php'">Регистрация</button>
                    <?php endif ?>
                </div>
            </header>
        </div>

        <div class="container">
            <div class="row">
                <div class="col-md-9">
                    <div class="page-header">
                        <h1 class="mb-5">Все услуги:</h1>
                    </div>
                    <?php
                        $yslugi = get_yslugi();
                    ?>
                    <?php foreach($yslugi as $yslug): ?>
                        <div class="col">
                            <div class="card mb-4 rounded-3 shadow-sm">
                                <div class="card-header py-3">
                                    <h4 class="my-0 fw-
normal"><?=$yslug['Название']?></h4>
                                </div>
                                <div class="card-body">
                                    <h1 class="card-title pricing-card-
title"><?=$yslug['Стоимость']?><small class="text-muted fw-light">
руб/род</small></h1>

```

```

        <ul class="list-unstyled mt-3 mb-4">
        <li><?=$yslug['Описание']?></li>
        <p></p>
        <li>Условия к документам:
<?=$yslug['тип_документа']?></li>
        </ul>
        <button class="brat w-100 btn btn-lg btn-primary
mb-3" type="submit">Взять</button>
        </div>
        </div>
        <?php endforeach; ?>
    </div>
</div>

<script src="/js/jquery-3.4.1.min.js"></script>
<script src="/js/salvattore.min.js"></script>
<script src="/app/doki/doki.js"></script>

<footer class="pt-4 my-md-5 pt-md-5 border-top">
    <div class="row">
        <div class="col-12 col-md">
            
            <small class="d-block mb-3 text-muted">©2022</small>
        </div>
    </div>
</footer>

</body>
</html>

```

4.2.5 Итоговый вид

В результате разработки мы получили сайт, выполняющий функцию сайта страховой компании (Рисунки 4.2.5.1-4.2.5.8).

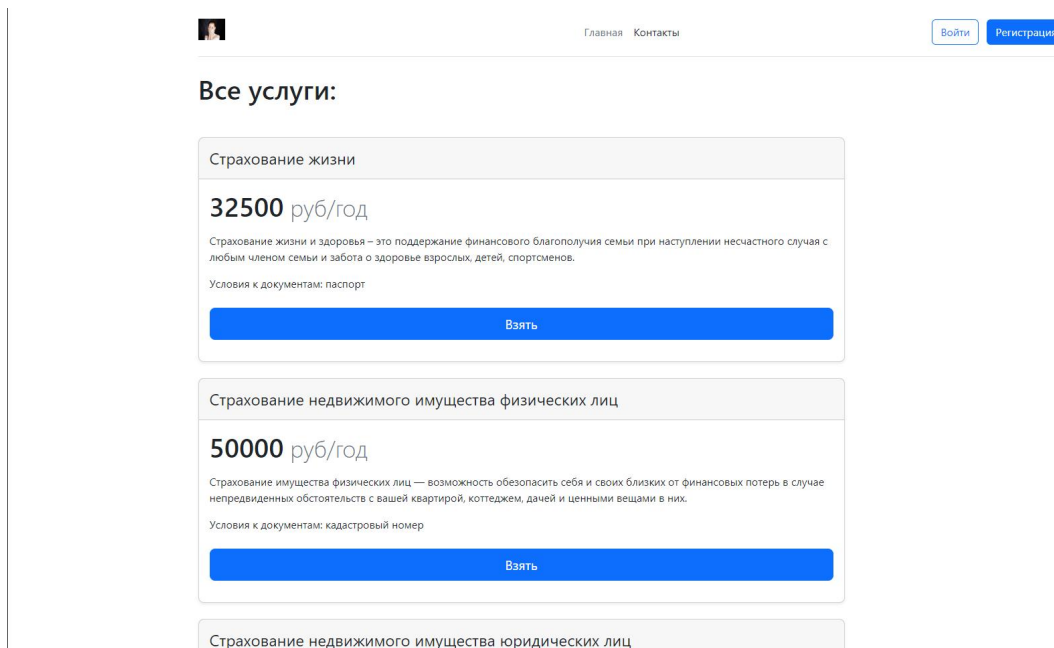


Рисунок 4.2.5.1 - Главная страница

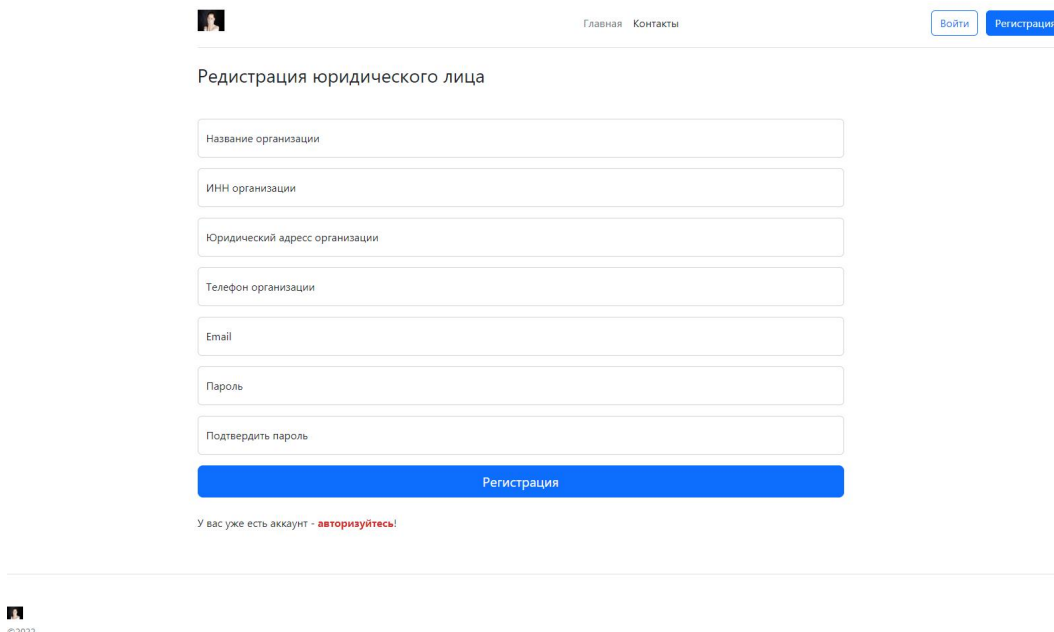



Рисунок 4.2.5.2 – Страница регистрации юридического лица



ГлавнаяКонтакты


ВойтиРегистрация

Редистрация физического лица

Регистрация

У вас уже есть аккаунт - [авторизуйтесь!](#)

Рисунок 4.2.5.3 – Страница регистрации физического лица



ГлавнаяКонтакты

ВойтиРегистрация

Вход юридического лица

Вход

У вас нет аккаунта? - [зарегистрируйтесь!](#)

Рисунок 4.2.5.4 – Страница входа

Профиль физ лица:

ФИО: Федоров Дмитрий Андреевич

ID: 4

Электронная почта: korasad@yandex.ru

ИНН: 233502835860

Телефон: 89162926917

Адресс жительства: Москва, просп. Вернадского, 78, стр. 4, Москва


[Выход](#)

Добавить документы

Мои документы

Мои услуги

Рисунок 4.2.5.5 – Профиль



Главная Контакты

Профиль Федоров Дмитрий Андреевич

Добавление документа

Тип документа

Выбрать тип документа... ▾

Гос номер

Добавить

Рисунок 4.2.5.6 – Добавить документы



Все документы:

Паспорт

Номер документа: 4516620063

Удалить

Рисунок 4.2.5.7 – Все добавленные документы



Ваши услуги:

31000 руб/год

Дата наччала действия: 2022-12-04

Дата конца: 2023-12-04

Рисунок 4.2.5.8 – Отчет по добавленным услугам

4.3 Вывод к разделу 4

В ходе выполнения данного раздела была разработана серверная и пользовательская части курсовой работы и база данных.

ЗАКЛЮЧЕНИЕ

В процессе выполнения данной курсовой работы был проведен анализ предметной области, в ходе которого были рассмотрены различные интернет-ресурсы, реализовавшие функцию «Страховая компания». Исходя из анализа, были составлены требования к серверной части веб-приложения. Был проведен анализ технологий в ходе которого были сформированы требования и выбраны инструменты для разработки серверной части веб-приложения. Была выбрана и разработана архитектура веб-приложения MVC.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Альфа страхование: официальный сайт – URL: <https://www.alfastrah.ru/>, (дата обращения: 11.11.2022).
2. Согласие страхование: официальный сайт – URL: <https://www.soglasie.ru/>, (дата обращения: 11.11.2022).
3. Ренесанс страхование: официальный сайт – URL: <https://www.renins.ru/>, (дата обращения: 11.11.2022).
4. Что такое PHP?: официальный сайт – URL: <https://www.php.net/manual/ru/intro-what-is.php> (дата последнего 21.11.2022).
5. Kotlin: официальный сайт – URL: <https://kotlinlang.org> (дата обращения 21.11.2022).
6. Java | Oracle: официальный сайт – URL: <https://www.java.com/ru/>, (дата обращения 21.11.2022).
7. Spring | Home: официальный сайт – URL: <https://spring.io/>, (дата обращения 21.11.2022).
8. IntelliJ IDEA: официальный сайт – URL: <https://www.jetbrains.com/ru-ru/idea/>, (дата обращения 21.11.2022).
9. MySQL: официальный сайт – URL: <https://www.mysql.com/>, (дата обращения 21.11.2022).
10. Apache maven: официальный сайт – URL: <https://maven.apache.org/>, (дата обращения 21.11.2022).
11. Github: официальный сайт – URL: <https://github.com/>, (дата обращения 21.11.2022).
12. Domain driven design, Особенность подхода и составляющие: сайт – URL: <https://simpleone.ru/glossary/domain-driven-design/> (дата обращения 22.11.2022).
13. Clean Coder Blog, Clean Architecture: сайт – URL: <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html/> (дата обращения 22.11.2022).

14. Design Patterns – MVC Pattern: сайт – URL: https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm/ (дата обращения 22.11.2022).
15. PHP 7 / Д.В. Котеров, И. В. Симдянов. - СПб.: БХВ-Петербург, 2021. 1088 с.: ил.
16. Хоффман Эндрю X85 Безопасность веб-приложений. — СПб.: Питер, 2021. — 336 с.: ил. — (Серия «Бестселлеры O'Reilly»)
17. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. — СПб. : Питер, 2021. — 352 с.
18. Персиваль, Г. Паттерны разработки на Python: TDD, DDD и событийно-ориентированная архитектура / Г. Персиваль, Б. Грегори. СПб. : Питер, 2022. — 336 с.
19. Раджпут Д. Spring. Все паттерны проектирования. - СПб.: Питер, 2019.
20. Меджуи М., Уайлд Э., Митра Р., Амундсен М. Непрерывное развитие API. Правильные решения в изменчивом технологическом ландшафте. СПб.: Питер, 2020.
21. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. 5-е изд.. - СПб.: Питер, 2021.
22. Бэнкс А., Порселло Е. GraphQL: язык запросов для современных веб-приложений. - СПб.: Питер, 2019.
23. Антонова И. И., Кашкин Е. В. Разработка web-сервисов с использованием HTML, CSS, PHP и MySQL [Электронный ресурс]: учебно -методическое пособие. - М.: РТУ МИРЭА, 2019. - – Режим доступа: <http://library.mirea.ru/secret/15052019/2022.iso>

ПРИЛОЖЕНИЕ

В связи с объемом кода курсовой работы, он доступен в данном гит-репозитории: <https://github.com/korasad/straxPhp>