Министерство науки и высшего образования РФ

Федеральное государственное автономное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ

УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

ПРОТОТИП ПРИЛОЖЕНИЯ ПО АГРЕГАЦИИ ПОИСКА АВТОЗАПЧАСТЕЙ

Пояснительная записка к курсовому проекту по дисциплине «Основы разработки программного обеспечения»

Студент гр. 431-3

\_\_\_\_\_\_\_ Д.П. Андреев

«\_\_» \_\_\_\_\_\_\_\_\_ 2024

Руководитель доцент кафедры АСУ, к.т.н.

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ А.К. Лукьянов (оценка) «\_\_» \_\_\_\_\_\_\_\_\_ 2024

Томск 2024

Министерство науки и высшего образования РФ

Федеральное государственное автономное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ

УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

Задание к курсовому проекту

по дисциплине «Основы разработки программного обеспечения»

Студенту группы \_\_431-3\_\_ факультета систем управления

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Андрееву Дмитрию Павловичу\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1. Тема курсовой работы: Прототип приложения по агрегации поиска автозапчастей.

1. Срок сдачи: 28 декабря 2024 г.
2. Содержание работы (перечень вопросов, подлежащих разработке):
   1. Описание среды разработки.
   2. Описание языка программирования.
   3. Описание проекта.
3. Требования:
   1. Программа должна обеспечить возможность поиска автозапчастей.
4. Требования к оформлению работы: в соответствии с ОС ТУСУР 01-2021

Дата выдачи задания: 10 сентября 2024 г.

Руководитель:

доцент кафедры АСУ, к.т.н., Лукьянов А.К. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

Задание принял к исполнению: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 10.09.2024

**Оглавление**

[Введение 4](#_Toc189314411)

[1 АНАЛИЗ ТРЕБОВАНИЙ 5](#_Toc189314412)

[1.1 Модель бизнес - процесса 5](#_Toc189314413)

[1.2 Пользователь вводит личные данные 5](#_Toc189314414)

[1.3 Вводит параметры поиска автозапчасти 5](#_Toc189314415)

[1.4 Система формирует запрос, обращается к сайту объявлений 5](#_Toc189314416)

[1.5 Формирование выходных данных по запросу 5](#_Toc189314417)

[1.6 Формулировка требований к системе 6](#_Toc189314418)

[1.7 Технологические требования 7](#_Toc189314419)

[1.8 Сравнение системы с аналогами 7](#_Toc189314420)

[2 ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ 9](#_Toc189314421)

[2.1 Модель проектирования и разработки системы 9](#_Toc189314422)

[2.2 Инструменты разработки 10](#_Toc189314423)

[2.3 Входные и выходные данные 11](#_Toc189314424)

[3 ПРОЕКТИРОВАНИЕ 12](#_Toc189314425)

[3.1 Пользователь вводит личные данные 12](#_Toc189314426)

[3.2 Вводит параметры поиска автозапчасти 12](#_Toc189314427)

[3.3 Система формирует запрос, обращается к сайту объявлений 12](#_Toc189314428)

[3.4 Формирование выходных данных по запросу 13](#_Toc189314429)

[4 КОДИРОВАНИЕ 14](#_Toc189314430)

[5 ТЕСТИРОВАНИЕ 18](#_Toc189314431)

[5.1 Поле для поиска запчастей 18](#_Toc189314432)

[5.2 Регистрация и авторизация пользователя 19](#_Toc189314433)

[Заключение 20](#_Toc189314434)

[Список использованных источников 21](#_Toc189314435)

# Введение

В современном мире автомобильный транспорт играет ключевую роль в жизни людей. Одной из актуальных проблем для автовладельцев является поиск и приобретение автозапчастей. Создание приложения-агрегатора для поиска автозапчастей позволит упростить этот процесс, обеспечивая пользователей удобным инструментом для быстрого нахождения необходимых деталей.

Цель проекта — разработать прототип приложения, позволяющий пользователю искать запчасти по введённым им данным.

Задачи проекта на этапах реализации:

* сформировать требования к приложению;
* сформировать спецификации к приложению;
* спроектировать приложение;
* реализовать прототип приложения;
* провести тестирование прототипа приложения.

# АНАЛИЗ ТРЕБОВАНИЙ

## Модель бизнес - процесса

Бизнес-процесс работы агрегатора включает следующие этапы:

1. Пользователь вводит личные данные.
2. Вводит параметры поиска автозапчасти.
3. Система формирует запрос, обращается к сайту объявлений.
4. Формирование выходных данных по запросу.

## Пользователь вводит личные данные

На данном этапе пользователь вводит личные данные для входа в систему, указывая имя, адрес электронной почты и пароль. Система проверяет данные на корректность и выводит сообщение.

## Вводит параметры поиска автозапчасти

Пользователь указывает город, марку, модель, поколение автомобиля, а также наименование или артикул нужной запчасти.

## Система формирует запрос, обращается к сайту объявлений

Для парсинга объявлений используются библиотеки AngleSharp и Leaf.xNet. Запросы отправляются к сайту avito.ru через HTTP-запросы. Данные извлекаются путем анализа HTML-структуры страниц, на которых размещены объявления. Для обхода защиты от парсинга используются прокси-серверы и задержки между запросами.

## Формирование выходных данных по запросу

Пользователь получает список найденных предложений с подробной информацией: название детали, цена, продавец, состояние, фотографии и контактные данные. При необходимости можно использовать фильтр сортировки по цене и наличию в вашем городе.

## Формулировка требований к системе

**Функциональные требования:**

1. Поиск автозапчастей: пользователь должен иметь возможность вводить параметры поиска, такие как город, марка автомобиля, модель и поколение.

2. Фильтрация результатов: реализовать возможность фильтрации по цене и наличию в вашем городе.

На рисунке 1.1 изображена диаграмма вариантов деятельности.



Рисунок 1.1 – Диаграмма вариантов деятельности.

**Нефункциональные требования:**

1. Удобство использования: прототип должен иметь интуитивно понятный интерфейс, обеспечивающий простоту навигации.

2. Адаптивность: интерфейс прототипа должен быть адаптирован для различных устройств (Планшеты, мобильные телефоны).

3. Визуальная привлекательность: дизайн должен быть современным и привлекательным для пользователей, учитывая тенденции в области UX/UI.

## Технологические требования

1. Инструменты разработки: должен быть создан с использованием инструментов для проектирования интерфейсов, таких как Figma, Adobe XD или аналогичных.

2. Платформа: прототип должен быть доступен как мобильное приложение, совместимое с современными смарт устройствами.

## Сравнение системы с аналогами

На данный момент не существует прямых зарубежных или отечественных аналогов систем, которые бы обладали полноценным функционалом, покрывающим все потребности пользователей.

Существуют различные онлайн-платформы для поиска автозапчастей, такие как Яндекс Маркет [1] и Enginex [2]. Однако эти ресурсы в большей степени ориентированы на поиск наиболее востребованных и расходных автозапчастей. Они не охватывают рынок б/у и контрактных запчастей, что значительно ограничивает возможности поиска для пользователей, заинтересованных в приобретении альтернативных вариантов.

Таким образом, несмотря на наличие отдельных решений, на данный момент отсутствует комплексная система, объединяющая все аспекты поиска автозапчастей, включая новые, б/у и контрактные детали.

# ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ

## Модель проектирования и разработки системы

Для разработки приложения была предпочтена эволюционная модель [3].

Эволюционная модель разработки приложения основана на следующей идее: разрабатывается первоначальная версия продукта, которая передаётся на испытание пользователям, затем она дорабатывается с учётом мнения пользователей, получается промежуточная версия продукта, которая также проходит «испытание пользователем», снова дорабатывается и так несколько раз, пока не будет получен необходимый программный продукт.

Различают два подхода к реализации эволюционного метода разработки:

1. Подход пробных разработок. Здесь большую роль играет постоянная работа с заказчиком (или пользователями) для того, чтобы определить полную систему требований к приложению, необходимую для разработки конечной версии продукта. В рамках этого подхода вначале разрабатываются те части системы, которые очевидны или хорошо специфицированы. Система эволюционирует (дорабатывается) путём добавления новых средств по мере их предложения заказчиком.
2. Прототипирование. Здесь целью процесса эволюционной разработки приложения является поэтапное уточнение требований заказчика и, следовательно, получение законченной спецификации, определяющей разрабатываемую систему. Прототип обычно строится для экспериментирования с той частью требований заказчика, которые сформированы нечётко или с внутренними противоречиями.

При разработке использовался подход пробных разработок

## Инструменты разработки

Для разработки приложения выбран ряд инструментов:

**Среда разработки Visual Studio** [4]. Интегрированная среда разработки Visual Studio является творческой стартовой площадкой, которую можно использовать для редактирования, отладки и сборки кода, а также для публикации приложения. В дополнение к стандартному редактору и отладчику, предоставляемых большинством интегрированных сред разработки, Visual Studio включает компиляторы, средства завершения кода, графические конструкторы и многие другие функции для улучшения процесса разработки программного обеспечения. Также имеет интеграцию с git [5] и поддерживает режим отладки.

**Язык программирования C#** [6]. Современный, инновационный, с открытым исходным кодом, кроссплатформенный объектно-ориентированный язык программирования. Язык ориентирован разработку различного программного обеспечения (ПО). На нём создают продвинутые бизнес-приложения, видеоигры, функциональные веб-приложения, приложения для Windows, macOS, мобильных программ для iOS и Android.

Последний актуальный фреймворк для C# на текущий момент – это .Net 8 [7].

**Платформа для кроссплатформенной разработки Xamarin** [8]**.** Xamarin.Forms представляет платформу, которая нацелена на создание кроссплатформенных приложений под Android, iOS и Windows 10.

**Библиотеки AngleSharp** [9] **и Leaf.xNet** [10]**.** AngleSharp — это библиотека .NET, которая позволяет анализировать гипертексты на основе угловых скобок, такие как HTML, SVG и MathML. Библиотека также поддерживает XML без проверки. Важным аспектом AngleSharp является то, что можно анализировать CSS. Входящий в комплект синтаксический анализатор основан на официальной спецификации W3C. Это позволяет создавать идеально переносимое HTML5 DOM-представление заданного исходного кода и обеспечивает совместимость с результатами в современных браузерах.

Leaf.xNet — библиотека для работы с HTTP-запросами, обеспечивающая удобный интерфейс для отправки GET и POST-запросов. Она поддерживает работу с прокси, управление cookies и заголовками, что делает её удобной для парсинга веб-страниц и сбора данных с сайтов объявлений.

## Входные и выходные данные

Используются следующие входные и выходные данные.

Входными данными для программы являются параметры, введенные пользователем, а также данные, полученные от магазинов. К основным входным данным относятся:

• Параметры автомобиля:

* марка автомобиля;
* модель;
* поколение.

• Параметры поиска запчастей:

* тип запчасти (например, тормозные колодки, фильтр масла и т.д.);
* поиск по коду запчасти или по названию;
* город, в котором интересует запчасть.

• Данные от магазинов:

* информация о наличии запчастей;
* цены на товары;
* условия доставки.

Программа генерирует следующие выходные данные:

• Результаты поиска — список найденных запчастей, соответствующих введённым параметрам. Для каждой запчасти выводятся:

* наименование запчасти;
* производитель;
* цена;
* наличие в выбранном городе;
* ссылка на магазин для оформления покупки;
* дополнительная информация (например, характеристики, фотографии).

# ПРОЕКТИРОВАНИЕ

Данные в системе проходят через следующие этапы:

1. Пользователь вводит личные данные.
2. Вводит параметры поиска автозапчасти.
3. Система формирует запрос, обращается к сайту объявлений.
4. Формирование выходных данных по запросу.

## Пользователь вводит личные данные

На данном этапе пользователь вводит личные данные для входа в систему, указывая имя, адрес электронной почты и пароль. Система проверяет данные на корректность и выводит сообщение.

## Вводит параметры поиска автозапчасти

Пользователь указывает город, марку, модель, поколение автомобиля, а также наименование или артикул нужной запчасти.

## Система формирует запрос, обращается к сайту объявлений

Для парсинга объявлений используются библиотеки AngleSharp и Leaf.xNet. Запросы отправляются к сайту avito.ru через HTTP-запросы. Данные извлекаются путем анализа HTML-структуры страниц, на которых размещены объявления. Для обхода защиты от парсинга используются прокси-серверы и задержки между запросами.

## Формирование выходных данных по запросу

Пользователь получает список найденных предложений с подробной информацией: название детали, цена, продавец, состояние, фотографии и контактные данные. При необходимости можно использовать фильтр сортировки по цене и наличию в вашем городе.

На рисунке 3.1 изображена диаграмма деятельности.

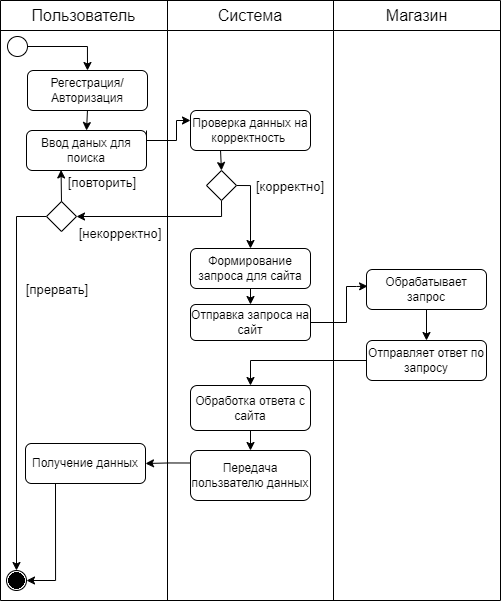


Рисунок 3.1 – Диаграмма деятельности.

# КОДИРОВАНИЕ

Разработка приложения проходит в несколько этапов. Сначала проектируется интерфейс, создаётся макет и прототип основных экранов. Разрабатывается интерактивный прототип с функциональными элементами. Производим соединение двух частей. Проводим пользовательское тестирование. Проверяются:

* корректность отображения данных;
* скорость получения результатов;
* корректность работы фильтров;
* удобство и информативность интерфейса.

По результатам тестирования вносим изменения и улучшения на основе полученной обратной связи. Изменения:

* улучшенная информативность интерфейса;
* добавление проверки корректности введённых данных;
* улучшение скорости получения результатов;
* добавление модуля расшифровки vin номера.

Результат изображен на рисунках 4.1-4.3.

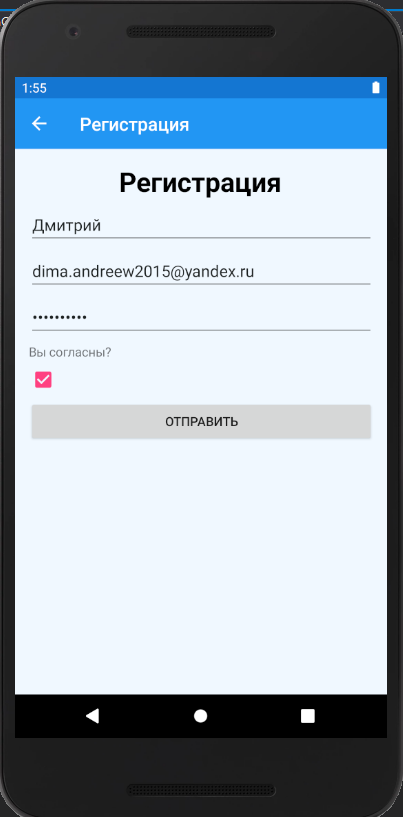


Рисунок 4.1 – Демонстрация экрана регистрации.

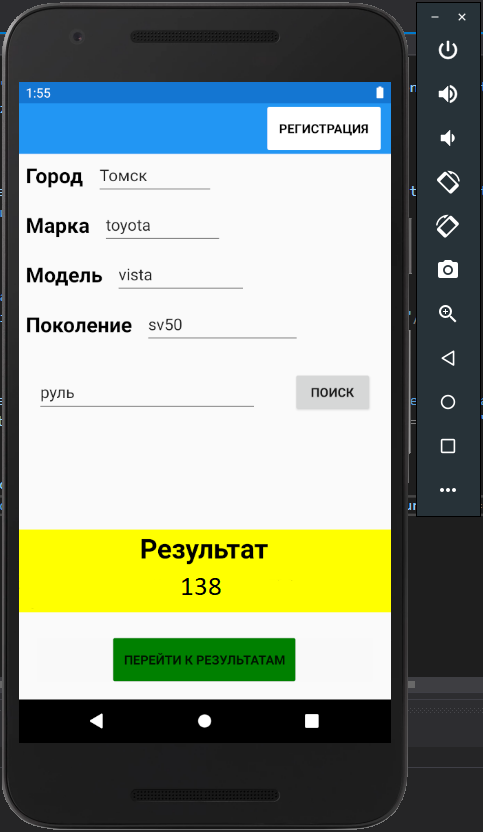


Рисунок 4.2 – Демонстрация экрана поиска.

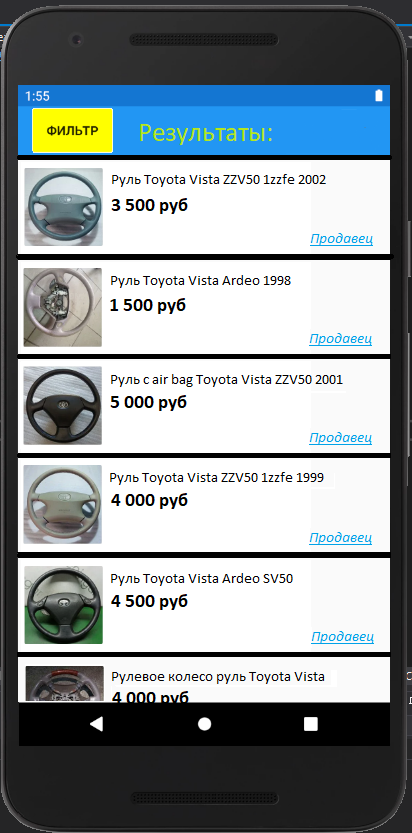


Рисунок 4.3 – Демонстрация результатов поиска.

# ТЕСТИРОВАНИЕ

В данной работе предусмотрено тестирование по методологии эквивалентного разбиения. Метод эквивалентного разбиения предполагает разделение всех возможных входных данных на классы, где все данные в одном классе считаются эквивалентными с точки зрения поведения программы.

Тестирование будет затрагивать проверки корректности входных данных, чтобы обеспечить корректную работу ПО.

## Поле для поиска запчастей

* **Классы эквивалентности:**
  + Валидные данные:
    - Название города (например, "Томск").
    - Название марки авто (например, "Toyota").
    - Название модели авто (например, "Mark II").
    - Название поколения авто (например, "X100").
    - Название детали (например, "Фильтр масляный").
  + Невалидные данные:
    - Пустой запрос в любом поле (кроме поля поколения авто).
    - Неразборчивый текст (например, "!!!@@@").
    - Слишком длинный текст (например, 500+ символов).
* **Тесты:**
  + Ввести корректный название детали, результат: деталь найдена.
  + Ввести название детали, результат: Количество деталей, найденных по параметрам.
  + Ввести пустой запрос, результат: предупреждение "Не удалось определить количество объявлений".
  + Ввести неразборчивый текст, результат: предупреждение " Не удалось определить количество объявлений ".

## Регистрация и авторизация пользователя

* **Классы эквивалентности:**
  + Валидные данные:
    - Корректное имя.
    - Корректный email.
    - Корректный пароль.
    - Присутствует галочка соглашения.
  + Невалидные данные:
    - Неправильный формат email (например, "user@com").
    - Слишком короткий пароль (например, менее 8 символов).
    - Пустое поле имя, email или пароль.
    - Не присутствует галочка соглашения.
* **Тесты:**
  + Ввести корректное имя, email и пароль, результат: успешная авторизация.
  + Ввести email без символа "@" и домена, результат: ошибка "Некорректный email".
  + Ввести пароль менее 8 символов, результат: ошибка "Некорректный пароль".
  + Оставить поля пустыми, результат: предупреждение "Заполните пустые поля".

# Заключение

В процессе курсового проекта были сформированы требования к приложению, сформированы спецификации к приложению, спроектирован и реализован прототип приложения.

Приложение в текущем состоянии позволяет пользователю искать автозапчасти, сравнивать их по различным критериям (цене, состоянию, наличию и т.д.), а также переходить на аккаунты продавцов для оформления заказа. Прототип приложения реализует базовый функционал для сайта Авито, который будет расширен в дальнейшем с учётом требований пользователей и поставленных задач.

# Список использованных источников

1. Официальный сайт ЯндексМаркет [Электронный ресурс]: Официальный сайт ЯндексМаркет. URL: <https://market.yandex.ru/> (дата обращения 7.11.2024).
2. Официальная страница Enginex [Электронный ресурс]: Официальный сайт Enginex. URL: https://enginex.pro/ (дата обращения 8.11.2024).
3. Официальный сайт StudFiles [Электронный ресурс]: Официальный сайт StudFiles. URL: <https://studfile.net/preview/4599943/page:8/> (дата обращения 5.09.2024).
4. Официальная сайт Microsoft [Электронный ресурс]: Официальный сайт Microsoft. URL: <https://visualstudio.microsoft.com/ru/> (дата обращения 29.09.2024).
5. Официальная сайт Git [Электронный ресурс]: Официальный сайт Git. URL: <https://git-scm.com/> (дата обращения 5.09.2024).
6. Официальная сайт Microsoft [Электронный ресурс]: Официальный сайт Microsoft. URL: <https://dotnet.microsoft.com/ru-ru/languages/csharp> (дата обращения 5.09.2024).
7. Официальная сайт Microsoft [Электронный ресурс]: Официальный сайт Microsoft. URL: <https://dotnet.microsoft.com/en-us/download/dotnet/8.0> (дата обращения 5.09.2024).
8. Официальная документация Xamarin [Электронный ресурс]: Официальный сайт Xamarin. URL: <https://learn.microsoft.com/ru-ru/previous-versions/xamarin/?link=man%3Amtouch(1)> (дата обращения 5.09.2024).
9. Официальная сайт GitHub [Электронный ресурс]: Официальный сайт GitHub. URL: <https://github.com/AngleSharp/AngleSharp> (дата обращения

13.10.2024).

1. Официальная сайт GitHub [Электронный ресурс]: Официальный сайт GitHub. URL: <https://github.com/csharp-leaf/Leaf.xNet> (дата обращения 13.10.2024)