* 1. **Эскизный проект**
     1. **Презентация идеи проекта**
        1. Название проекта

Программная система по привлечению внимания к проблемам города, а также учёту инцидентов различных областей – «NoProblems» (Название временное).

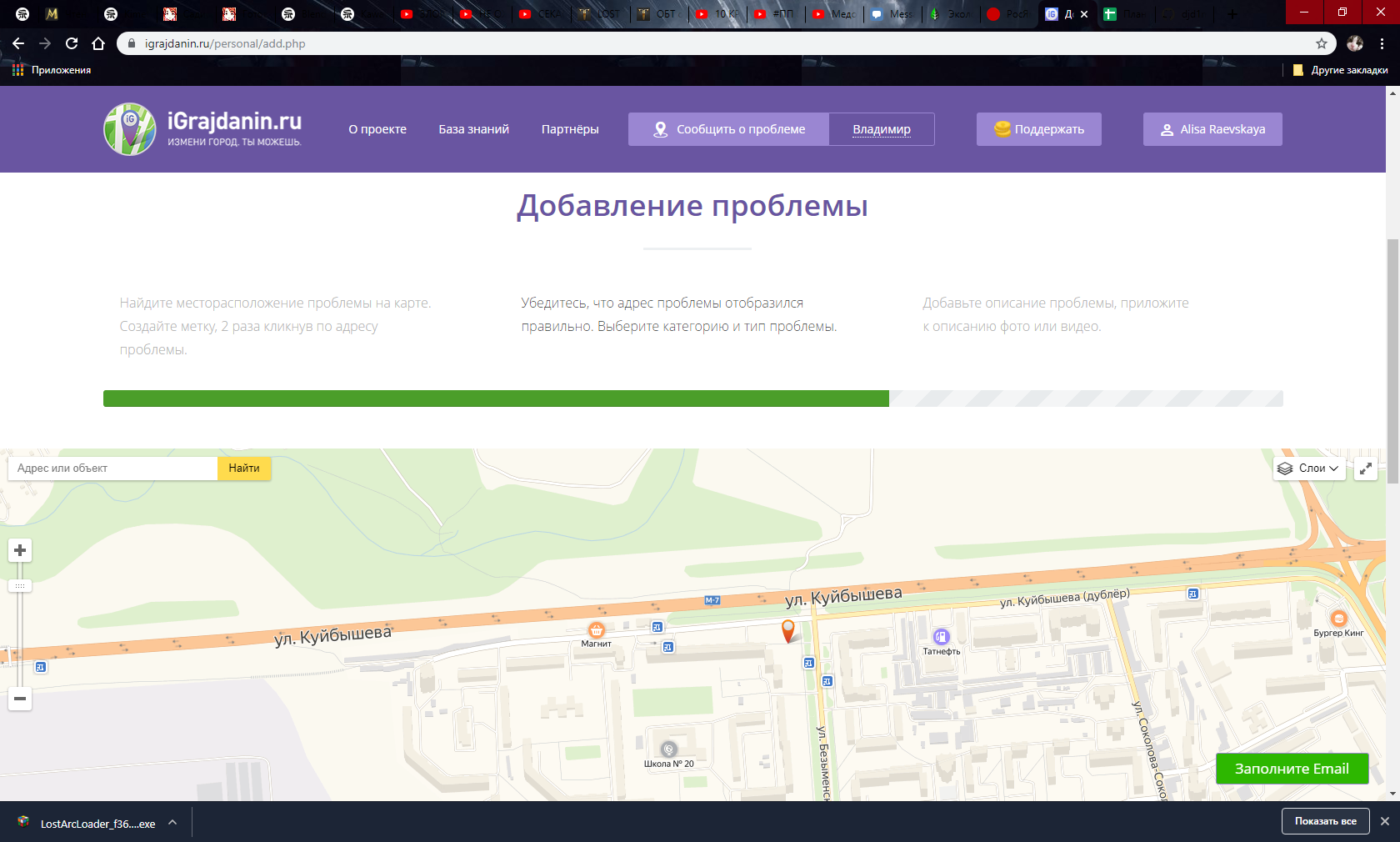
* + - 1. Цель и задачи проекта

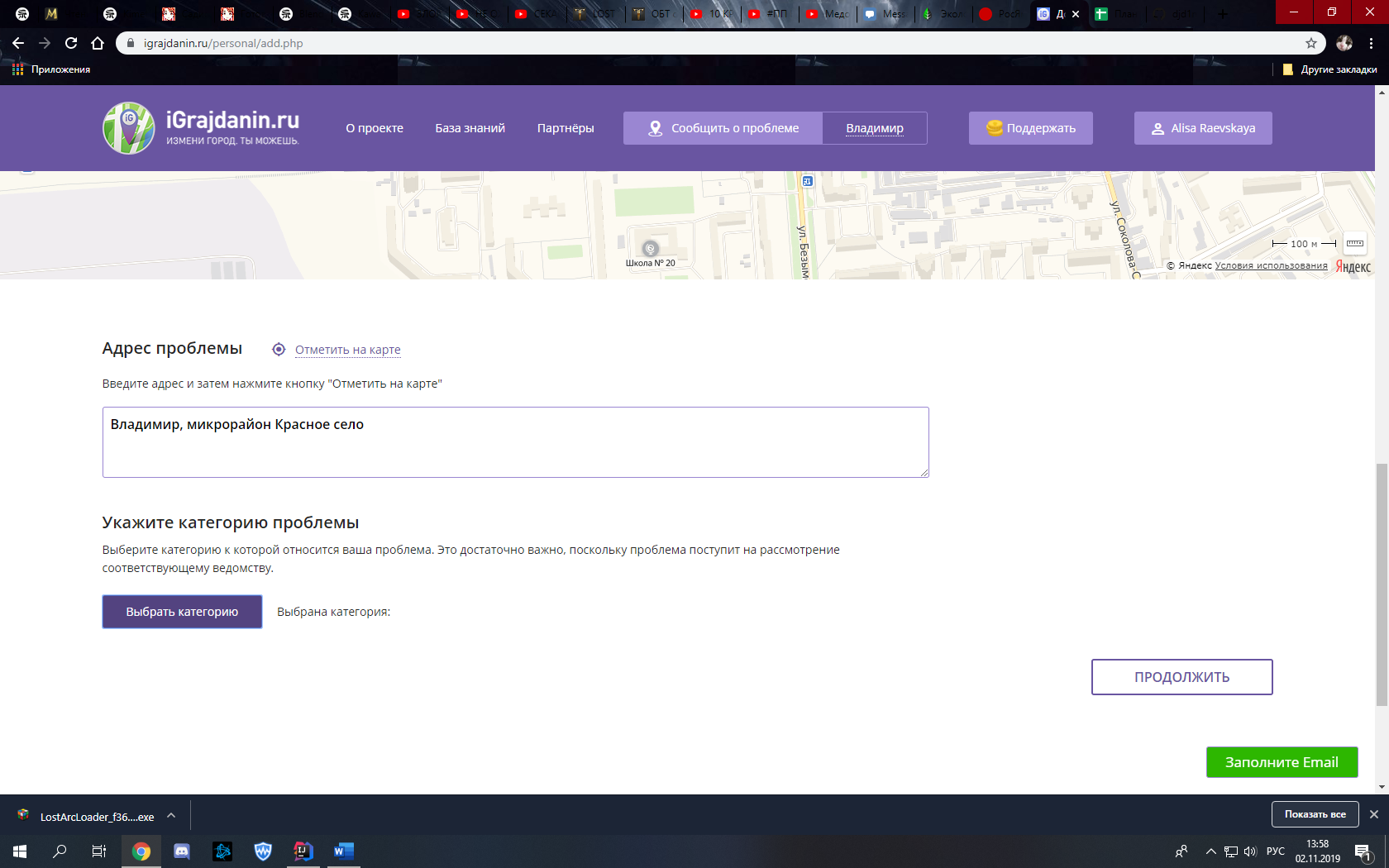
Цель – привлечение внимания к проблемам, волнующих граждан, с целью их решения и улучшение жизни, а также информирование о всех инцидентах, произошедших в области

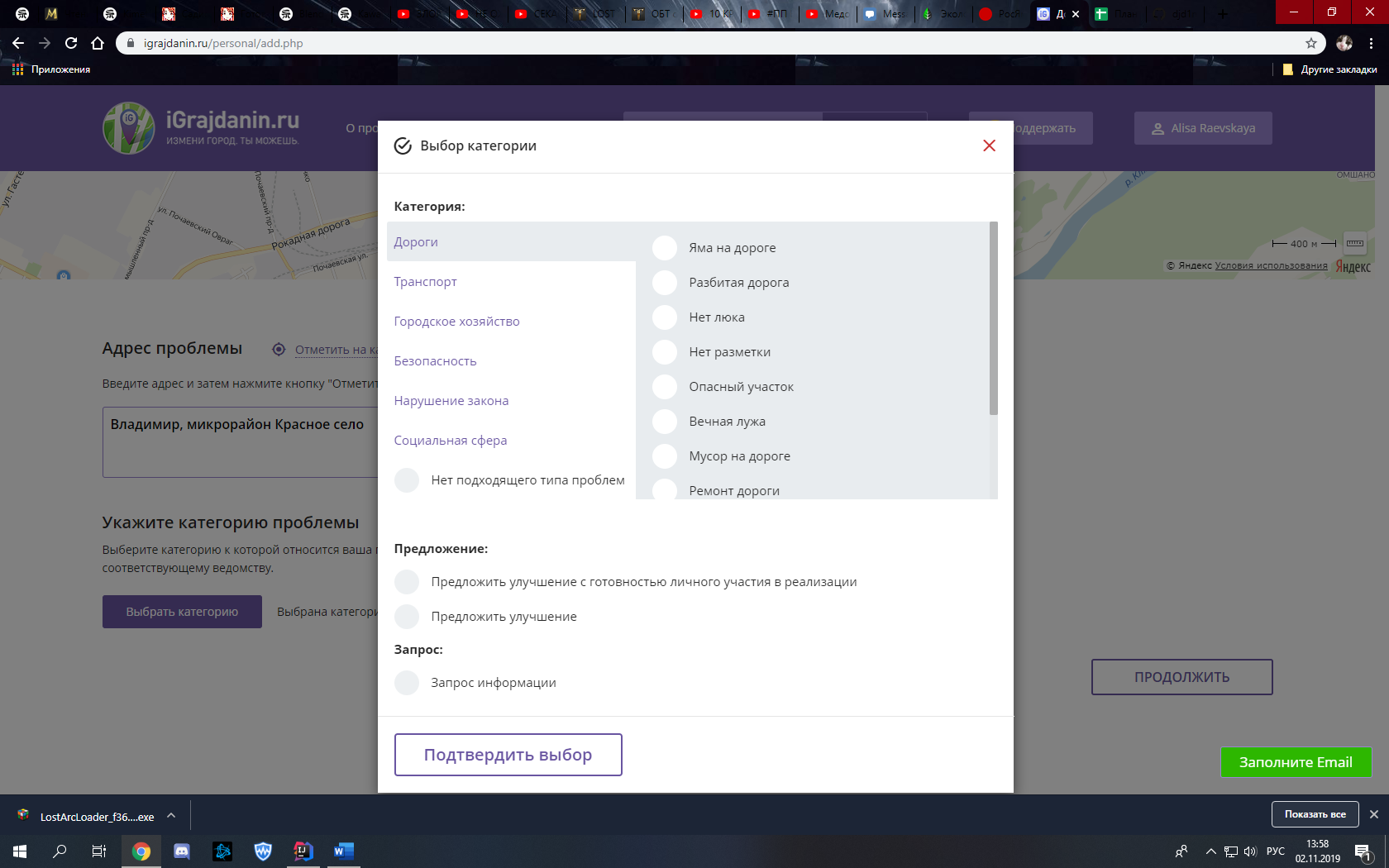
Задачи – изучить предметную область, изучить технологии для разработки приложения, спроектировать архитектуру приложения, разработать прототип приложения, а в будущем довести его до «релизного» состояния, разработать современный дизайн приложения

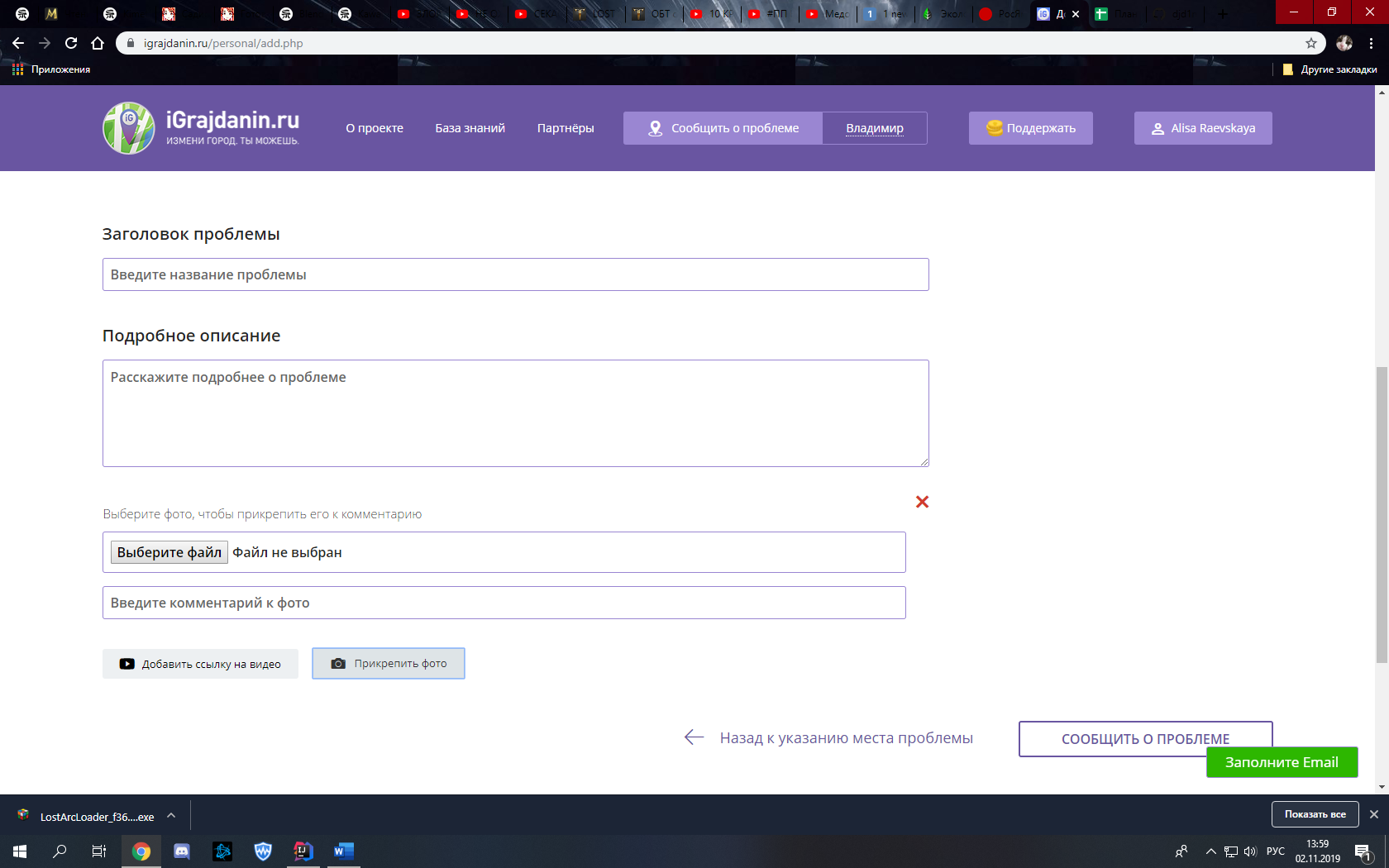
* + - 1. Список аналогов

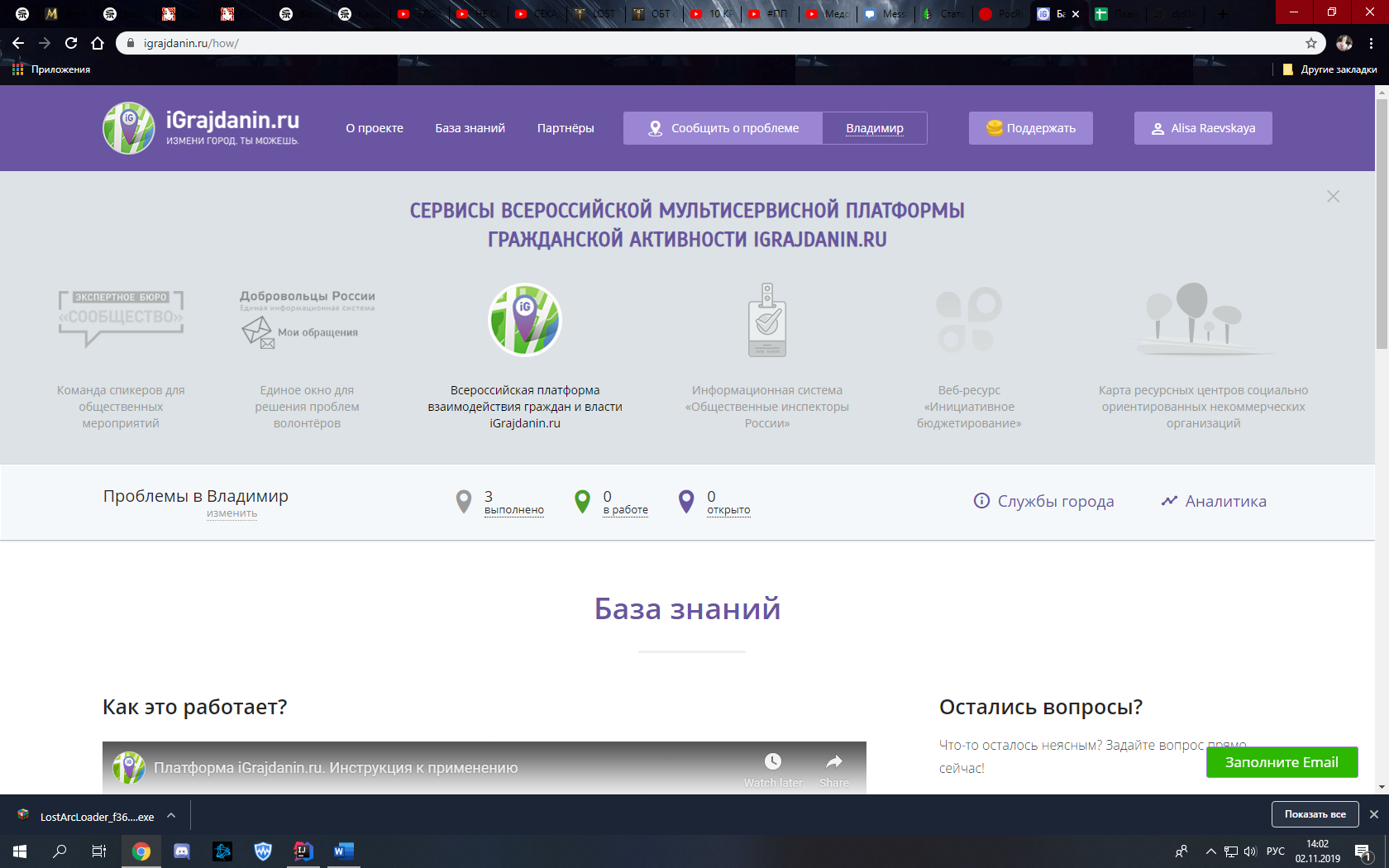
<https://igrajdanin.ru/> - заявление на проблему будет похоже на этот сайт



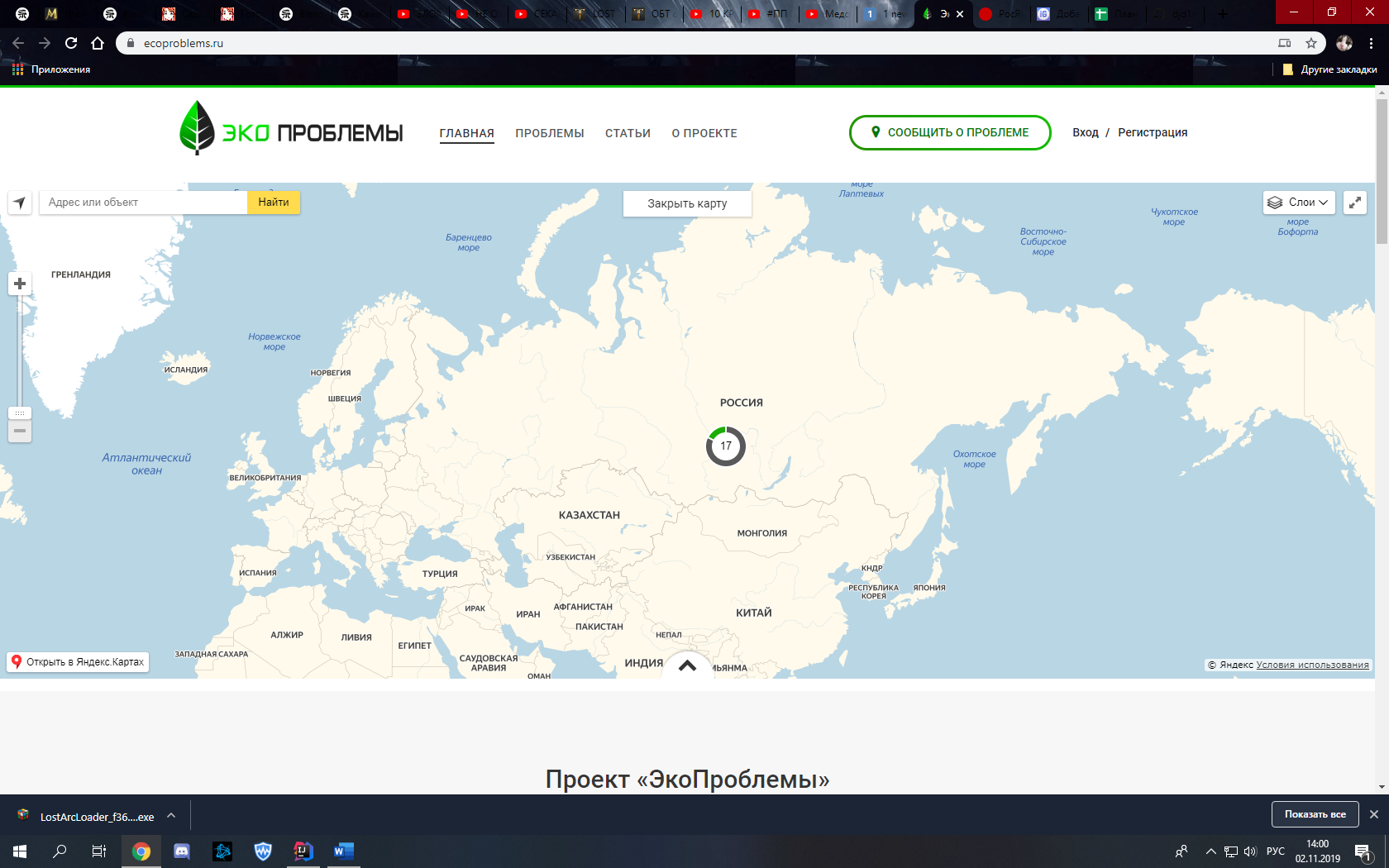


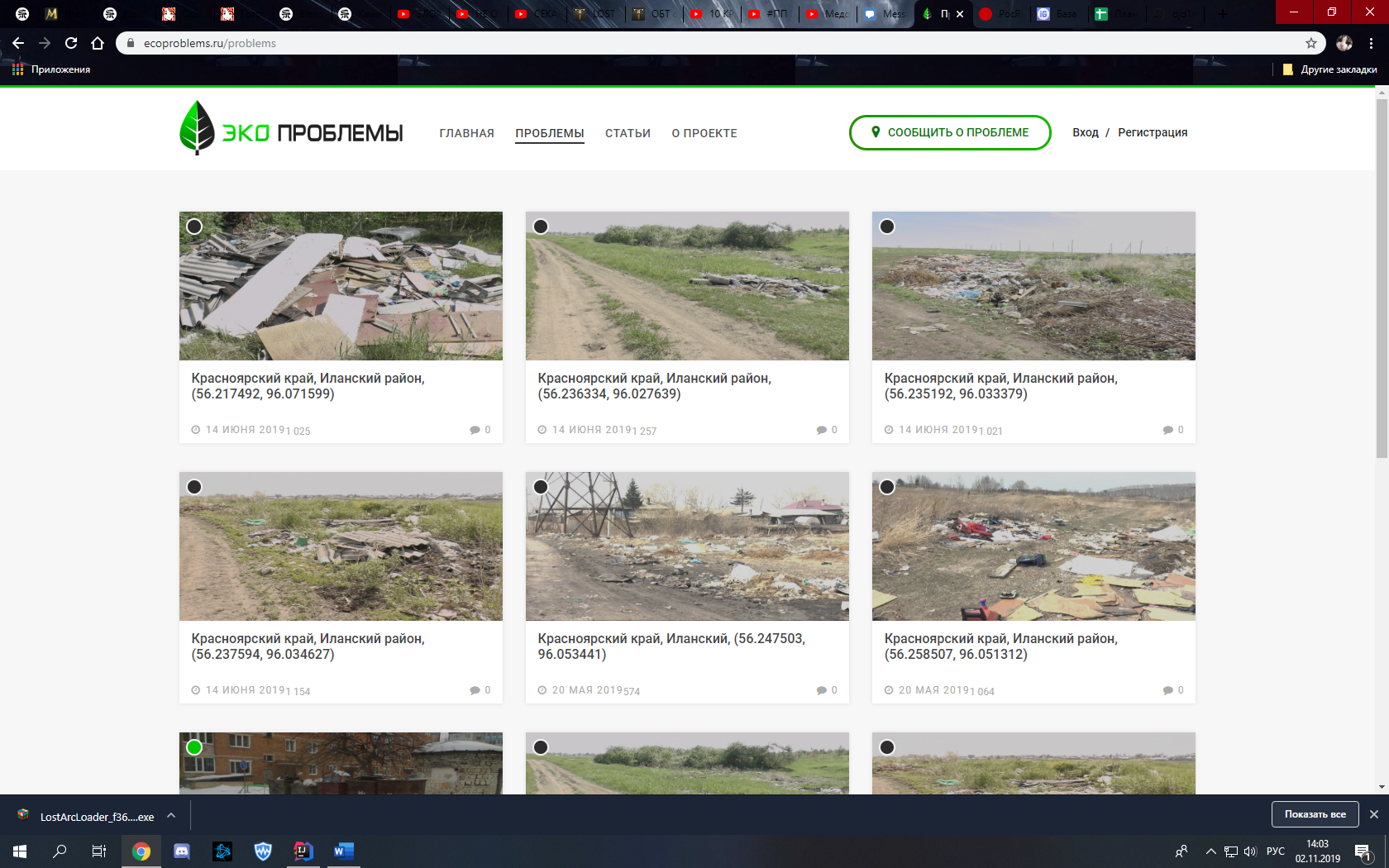




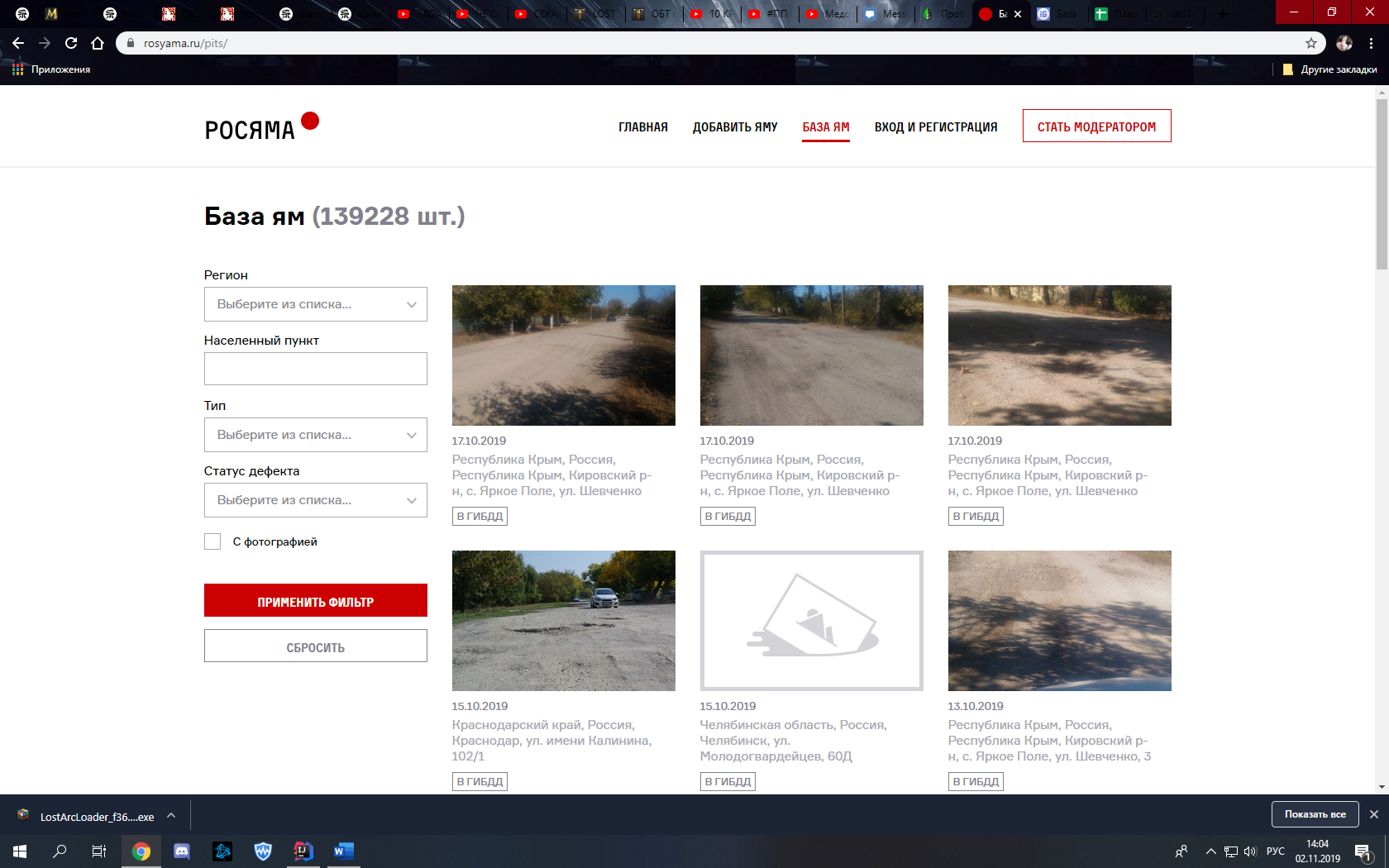


<https://ecoproblems.ru/> - стиль оформления сайта и карта будут похожи на этот сайт





<https://rosyama.ru/> - сортировка в проблемах будет как на этом сайте



<https://lenta.ru/> - дизайн новостей будет как на этом сайте



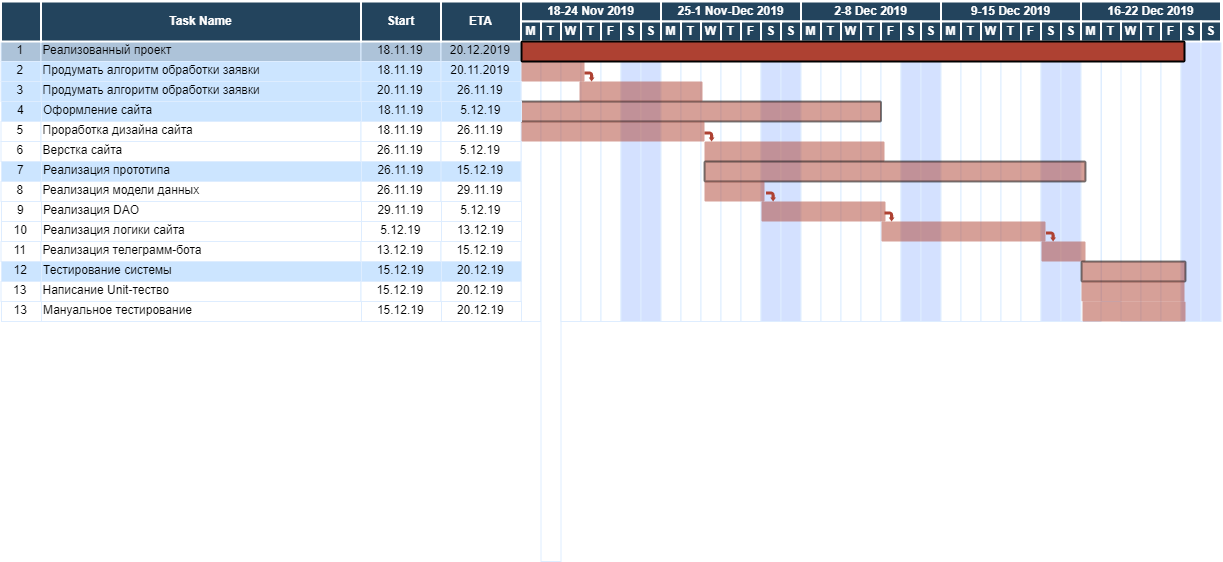
* + - 1. Команда, роли в команде

Команда: Марчёнок Владислав, Емельянов Дмитрий, Подолец Анастасия.

Роли в команде:

* Марчёнок – разработчик, аналитик
* Емельянов – разработчик, аналитик
* Подолец –дизайнер, frontend разработчик
  + 1. **Календарный план проекта**
       1. Календарный план проекта и распределение задач по исполнителям

|  |  |  |
| --- | --- | --- |
| Цель | Исполнитель | Дата |
| **Продумать алгоритм обработки заявки** | Марчёнок, Емельянов, Подолец | до 20.11.19 |
| **Спроектировать БД** | Марчёнок, Емельянов, Подолец | до 26.11.19 |
| **Оформление сайта** |  | до 5.12.2019 |
| Проработка дизайна сайта | Марчёнок, Подолец | до 26.11.2019 |
| Верстка сайта | Подолец | до 5.12.2019 |
| **Реализация прототипа** |  | до 15.12.2019 |
| Написание модели данных | Марчёнок | до 29.11.2019 |
| DAO | Емельянов, Подолец | до 5.12.2019 |
| Реализация логики сайта | Марчёнок, Емельянов | до 13.12.2019 |
| Реализация телеграм-бота | Емельянов | до 15.12.2019 |
| **Тестирование прототипа** |  | до 20.12.2019 |
| Unit-тесты | Марчёнок, Емельянов, Подолец | до 20.12.2019 |
| Мануальное тестирование | Подолец | до 20.12.2019 |



* + 1. **State of Art**
       1. Обзор актуальных технологий

Фрейморки для веб-разработки:

* PHP:

1. Yii - популярный фреймворк для php-разработки, основанный на парадигме MVC. Основное преимущество — очень высокая скорость работы и, как следствие, производительность.
2. Symphony - php-фреймворк, на базе которого можно разрабатывать сложные веб-приложения. У фреймворка Symfony весьма мощная функциональность, продуманная архитектура, а также развитое сообщество.

* Java:

1. Spring MVC – Spring популярная платформа приложений с открытым кодом, предназначенная для упрощения разработки для J2EE. Она состоит из контейнера, платформы управления элементами и набора интегрируемых служб для веб-интерфейсов пользователя, транзакций и сохранения состояния. В состав платформы Spring входит Spring Web MVC – расширяемая платформа MVC для создания веб-приложений.

* C# (И не только):

1. ASP.NET - .NET Framework - программная платформа, выпущенная компанией Microsoft, которая подходит для разных языков программирования. ASP.NET — технология создания веб-приложений и веб-сервисов от компании Microsoft. Она является составной частью платформы Microsoft .NET и развитием более старой технологии Microsoft ASP.
2. .NET Core обладает следующими характеристиками:
3. Кроссплатформенность. Поддержка операционных систем Windows, macOS и Linux.
4. Согласованность между архитектурами. Одинаковое выполнение кода в различных архитектурах, включая x64, x86 и ARM.
5. Программы командной строки. Удобные инструменты для локальной разработки и сценариев непрерывной интеграции.
6. Гибкая разработка. Может включаться в приложение или устанавливаться параллельно (на уровне пользователя или системы). Возможность использования с контейнерами Docker.
7. Совместимость. Платформа .NET Core совместима с .NET Framework, Xamarin и Mono благодаря .NET Standard.
8. Открытый код. Платформа .NET Core имеет открытый код и распространяется по лицензиям MIT и Apache 2. .NET Core является проектом .NET Foundation.
9. Поддержка от Майкрософт. Корпорация Майкрософт предоставляет поддержку .NET Core.

Выводы: Выбор был сделан в пользу Spring MVC, потому что нам заказчик принудительно порекомендовал использовать его.

* + - 1. Анализ аналогов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Сайт\Критерии | Удобство | Функционал | Дизайн | Актуальность | Оценка |
| [ЯГражданин](https://igrajdanin.ru/) | 6 | 9 | 8 | 8 | 7,75 |
| [ЭкоПроблемы](https://ecoproblems.ru/) | 9 | 8 | 9 | 7 | 8,25 |
| [РосЯма](https://rosyama.ru/) | 7 | 7 | 7 | 8 | 7,25 |

Выводы: ЭкоПроблемы является наиболее привлекательным и удобным аналогом.

* + 1. **Анализ предметной области**
       1. Описание предметной области
       2. Словарь предметной области

Заявка – это заявление, заполненное в указанной форме, с просьбой опубликовать волнующую актера проблему.

Проблема – это проблема или инцидент, волнующие граждан и требующие решения

Новость – это значимое событие, которое произошло в городе за последнее время.

* + - 1. Пользователи системы и роли

Гость может оставлять заявку, просматривать выложенные проблемы и новости, соглашаться с актуальностью проблемы.

Зарегистрированный пользователь может оставлять комментарии к заявкам, оставлять заявку, просматривать выложенные проблемы и новости, соглашаться с актуальностью проблемы.

Модератор рассматривает отправленные заявки, может выложить заявку или отказать.

Редактор выкладывает новости

* + - 1. Прецеденты

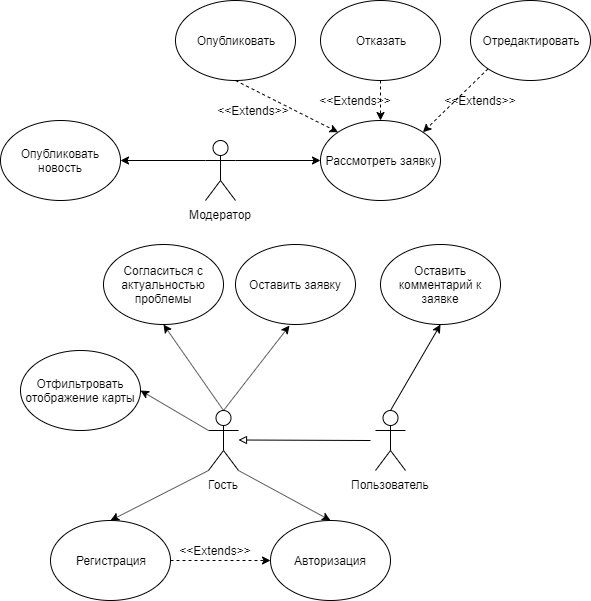


Рисунок 1. Диаграмма прецедентов

Название: «Оставить заявку»  
Предусловие: -.  
Действующее лицо: Гость, зарегистрированный пользователь.  
Основной поток: Актер нажимает на кнопку «Сообщить о проблеме».  
Открывается форма для заполнения заявки.  
Актер заполняет поля «название», «адрес», «категория», «описание», «фотография».   
Актер отправляет заявку.   
Альтернативный поток: -.  
Постусловие: модератору приходит уведомление об отправленной заявке.

Название: «Оставить комментарий к заявке»  
Предусловие: -.  
Действующее лицо: Зарегистрированный пользователь.  
Основной поток: Актер выбирает проблему, выложенную на сайте.  
Открывается форма проблемы.   
В поле «Оставить комментарий» актер пишет комментарий.  
Актер нажимает кнопку «Отправить комментарий»  
Альтернативный поток: -.  
Постусловие: -.

Название: «Согласиться с актуальностью проблемы»  
Предусловие: -.  
Действующее лицо: Гость, зарегистрированный пользователь.  
Основной поток: Актер выбирает проблему, выложенную на сайте.  
Открывается форма проблемы.  
Актер нажимает кнопку «Согласиться с актуальностью проблемы».  
Альтернативный поток: -.  
Постусловие: -.

* + - 1. Описание документов / сущностей предметной области

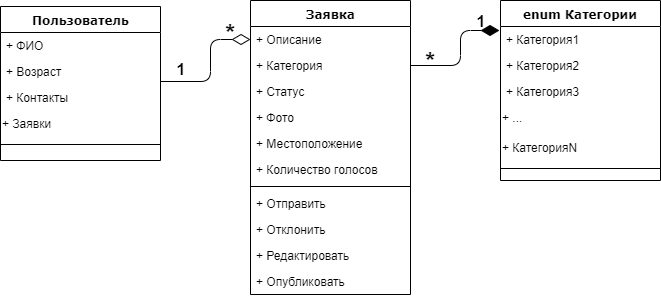
****

Рисунок 2. Концептуальная диаграмма классов.

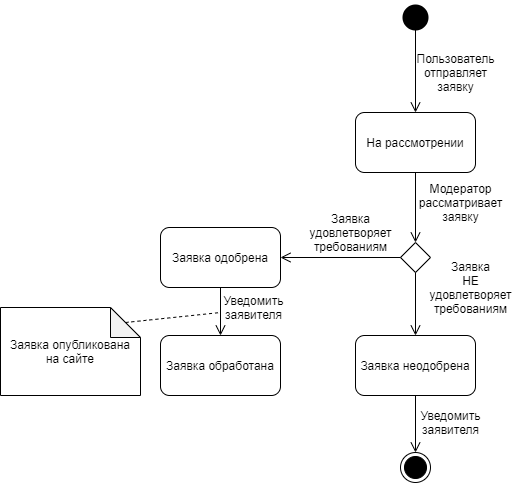


Рисунок 3. Диаграмма состояний сущности «Заявка»

* + - 1. Анализ бизнес-процессов
      2. Функциональные требования к системе
  + Система должна создавать заявку, отправлять на рассмотрение, опубликовывать заявку.
  + Система должна отправить пользователю уведомление о том, что его заявка на рассмотрении/опубликована/отклонена.
  + Система должна создавать нового пользователя, редактировать или **удалять (?)** пользователя.
  + Система должна фильтровать проблемы по категориям.
  + Система должна выводить топ актуальных проблем.
  + Система должна отображать новости указанного города.
  + Система должна поддерживать комментарии к проблемам.
  + Система должна поддерживать отправку заявок через приложение *«Telegram»*.
  + Система должна поддерживать отображение местонахождения проблем на карте.
  + Система должна поддерживать различные категорий проблем.
  + Система должна поддерживать рейтинговую таблицу проблем по количеству голосов пользователей.
  + Система должна возможность комментирования проблем только от зарегистрированных пользователей.
    - 1. Нефункциональные требования к системе
* Пользовательский интерфейс должен быть удобным, чтобы каждый пользователь мог интуитивно понять на какую кнопку нажать, чтобы совершить нужное действие.
* Поддержка клиент-серверной архитектуры.
  + 1. **Макет программной системы**

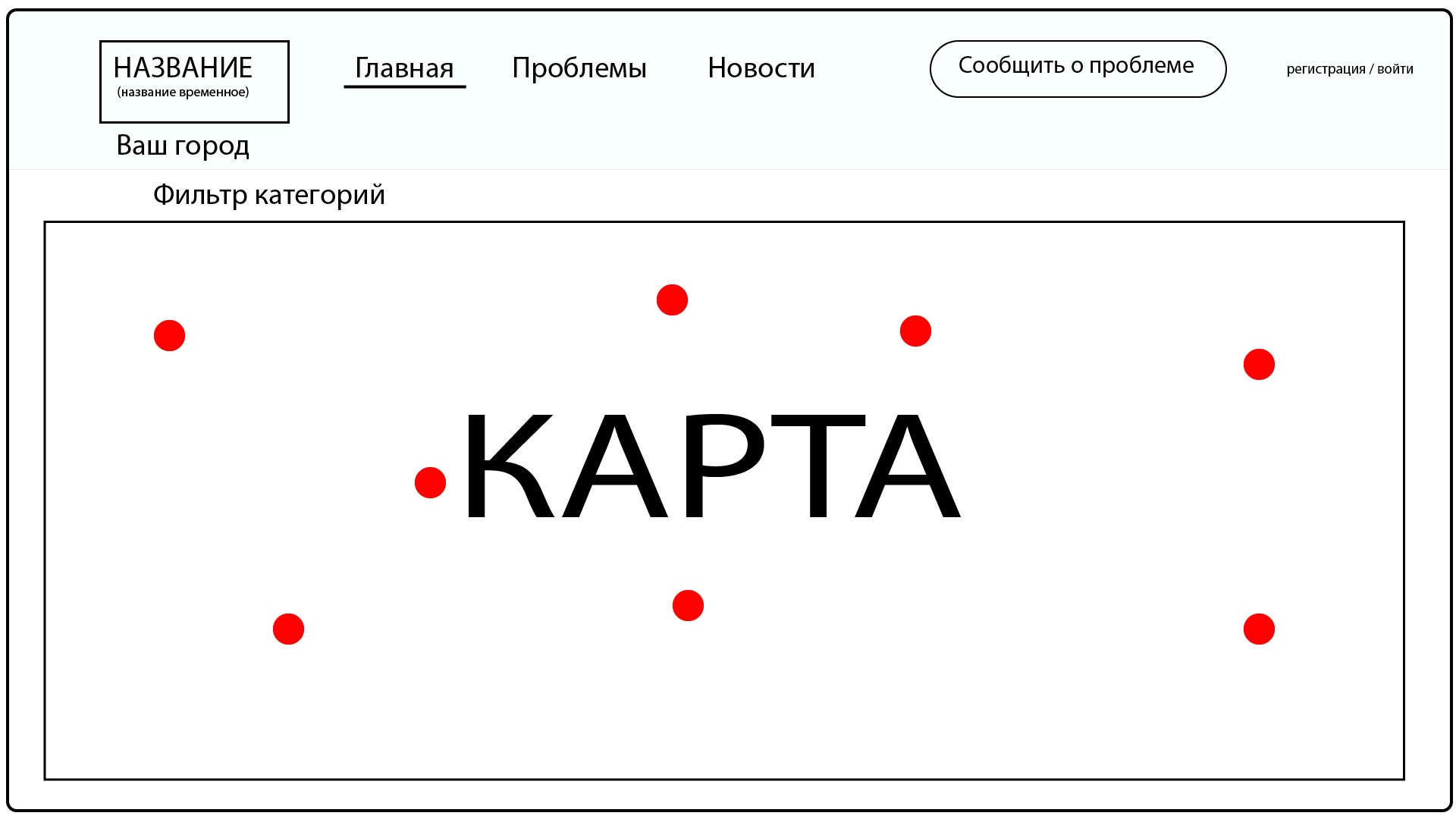
****

Рисунок 4. Главная

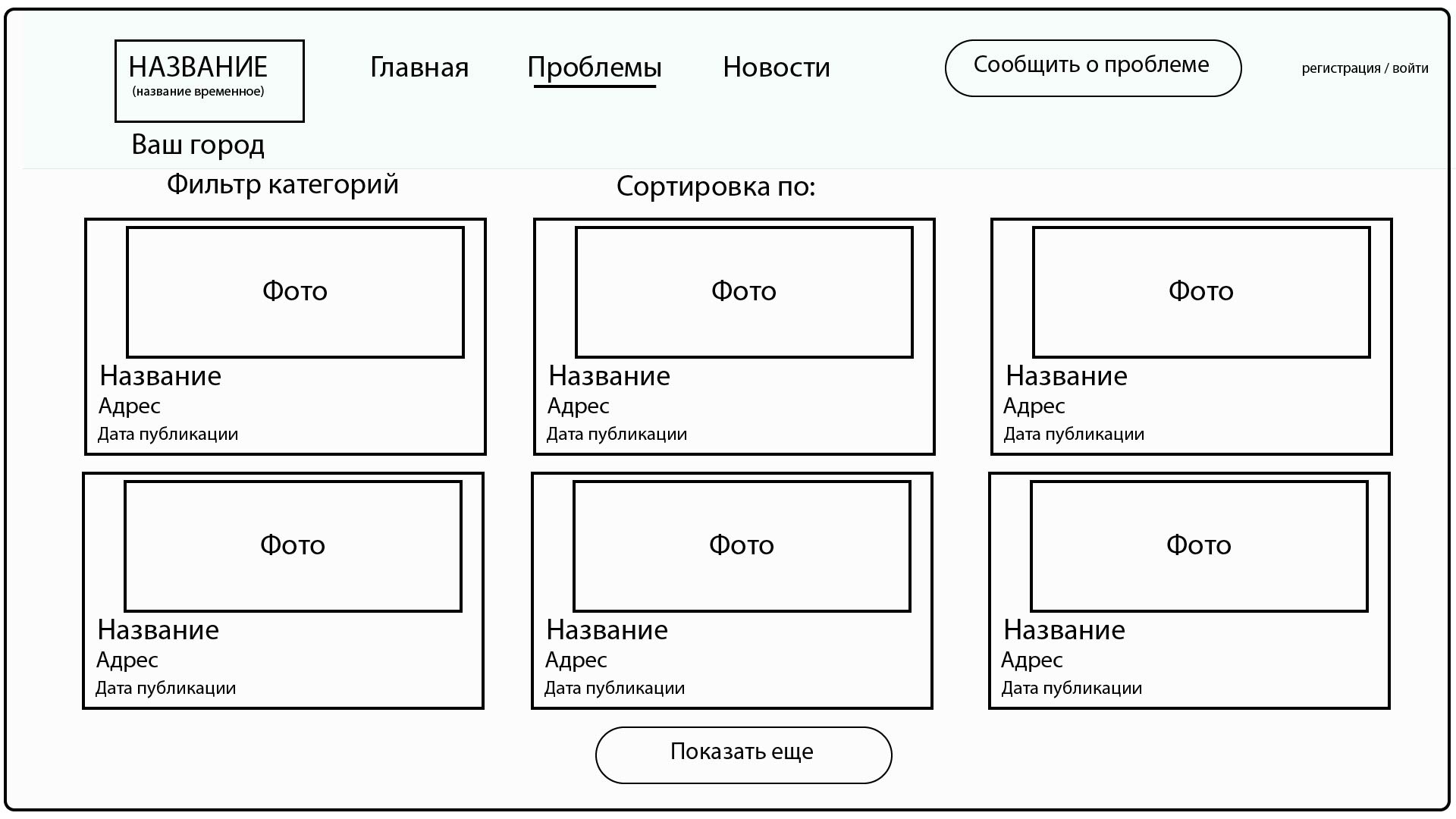
****

Рисунок 5. Проблеммы

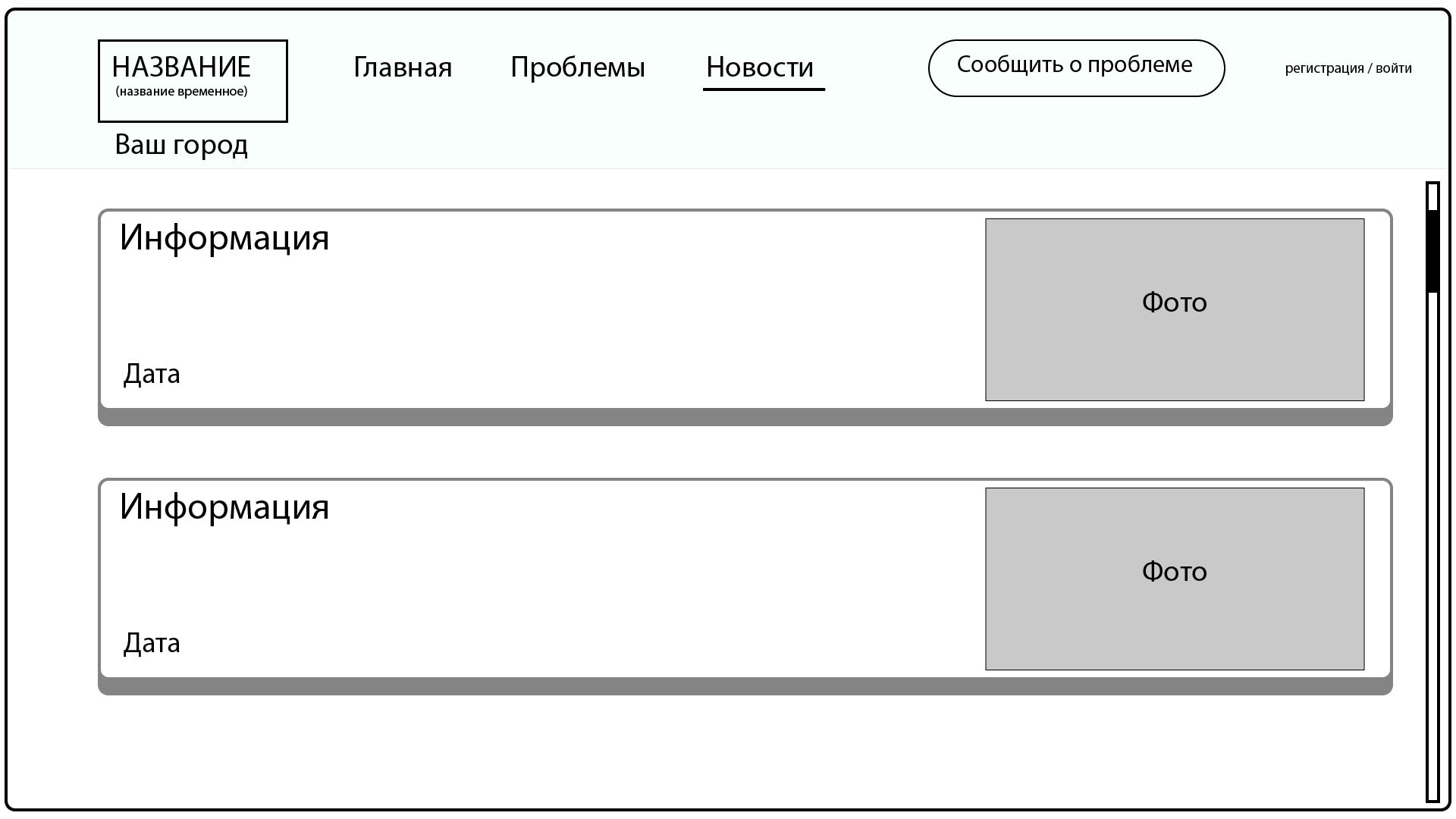
****

Рисунок 6. Новости

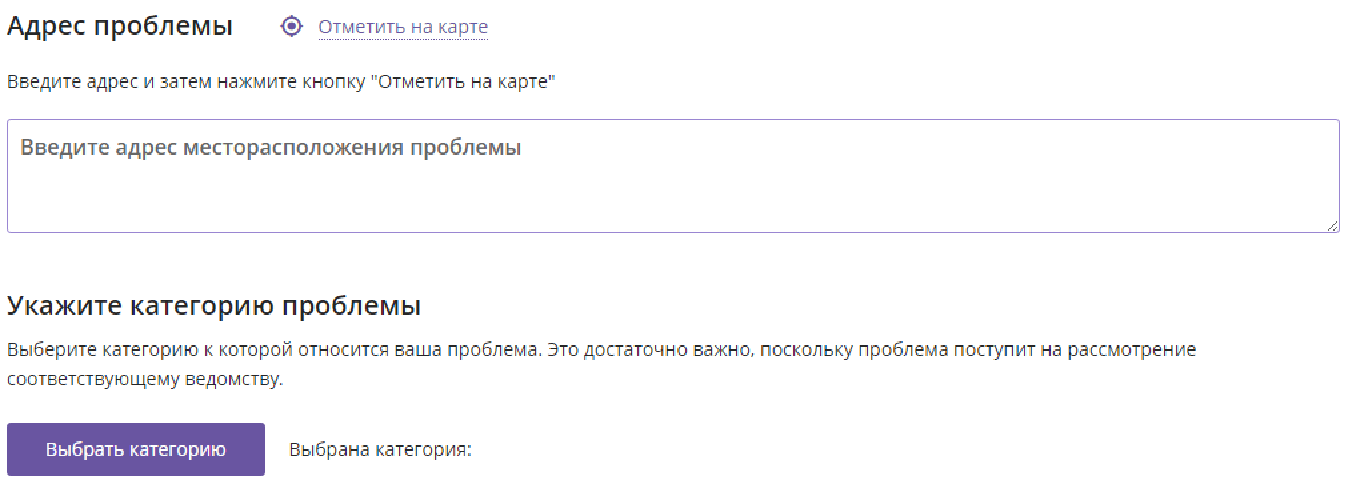
****

Рисунок 7. Заполнение заявки

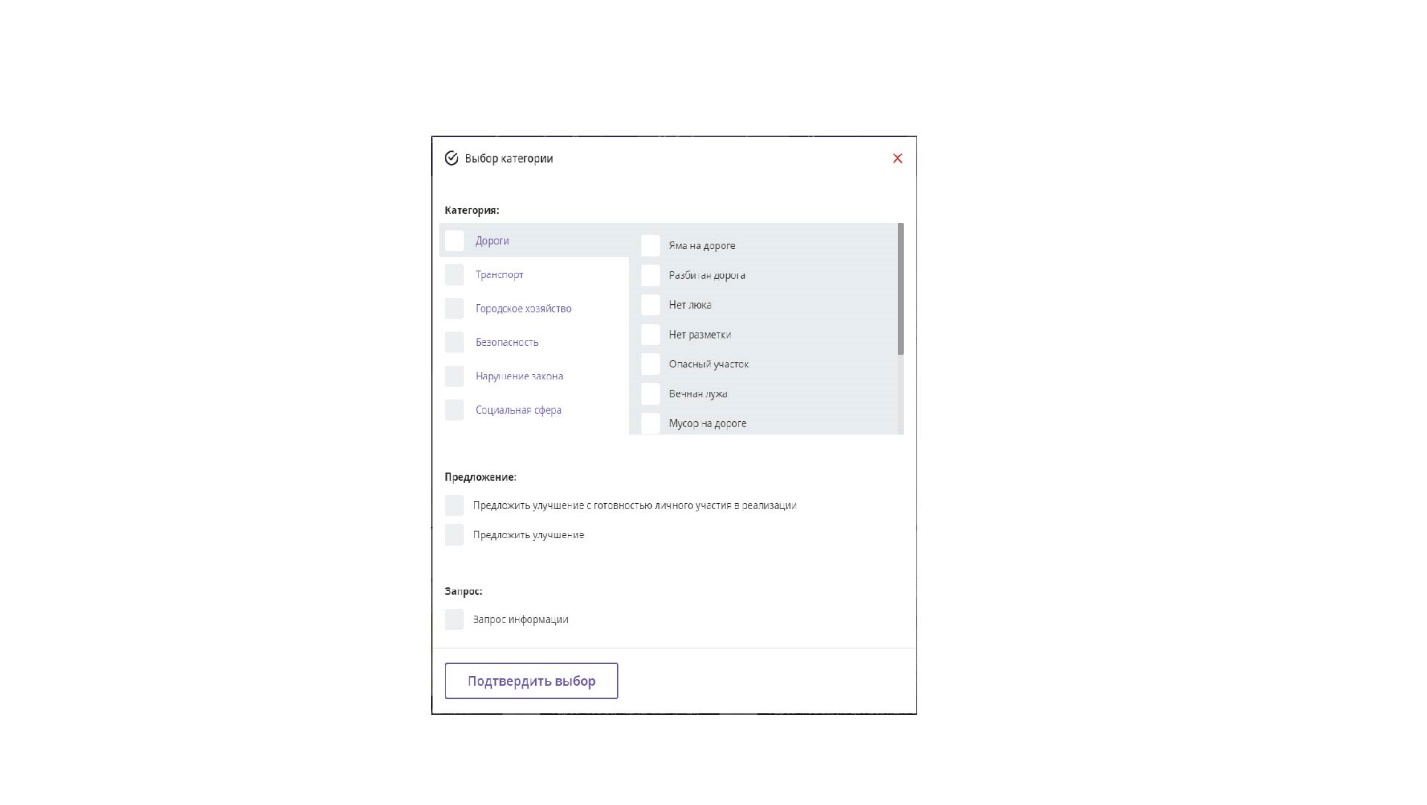
****

Рисунок 8. Фильтр категорий