МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра Програмної інженерії

ПОЯСНЮВАЛЬНА ЗАПИСКА З КУРСОВОЇ РОБОТИ

Дисципліна: «Бази даних»

Тема роботи: «Інформаційна система «Відділення міліції»»

|  |  |
| --- | --- |
| Виконав ст. гр. ПІ-15-5:  Керівник:  Комісія:  Доц. каф. ПІ:  Ст. вик. каф. ПІ:  Ст. вик. каф. ПІ: | Цомкалов О.А.  Широкопетлєва М.С.  Мазурова О.А.  Черепанова Ю.Ю.  Широкопетлєва М.С. |

Харків 2016

Харківський національний університет радіоелектроніки

Кафедра:\_\_\_Програмної інженерії

Дисципліна:        Бази даних

Спеціальність:\_Програмна інженерія\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Курс:\_\_\_2\_\_\_\_\_\_\_\_ Група \_\_\_ПІ-15-5\_\_\_\_\_ Семестр\_\_\_\_3\_\_\_\_\_\_\_\_\_\_

ЗАВДАННЯ

на курсову роботу студента

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Цомкалова Олега Андрійовича\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Тема роботи: Інформаційна система «Відділення міліції»

2. Строк здачі закінченої роботи:          30.12.2015

3. Вихідні дані для роботи: методичні вказівки до виконання курсової роботи, вимоги до інформаційної системи, предметна область, що пов’язана із відділенням міліції. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Зміст розрахунково - пояснювальної записки: вступ, аналіз предметної області; постановка задачі; проектування бази даних; опис програми; висновки; перелік посилань.

5. Перелік графічного матеріалу: загальна схема концептуальної моделі, ER-діаграма, схема моделі бази даних, копії екранів прикладної програми, приклади звітів прикладної програми.

6. Дата видачі завдання \_\_\_01.10.15 р.\_\_\_

РЕФЕРАТ

Пояснительная записка к курсовой работе: 42 с., 28 рис., 5 источников.

Цель работы: разработка программы под названием «Інформаційна система «Відділення міліції»», которая будет использоваться для принятия заявлений пользователей, на языке программирования JavaScript.

Метод разработки: JetBrains WebStorm 2016, JavaScript, Node.js.

В результате разработки была получена программа под названием «Відділення міліції», на котором можно оставить заявление в полицию. Пользователь имеет возможность добавлять, редактировать, удалять и получать для распечатки заявления.

КУРСОВАЯ РАБОТА, ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ, ЯЗЫК ПРОГРАММИРОВАНИЯ JAVASCRIPT, NODE.JS, БАЗЫ ДАННЫХ, ЗАЯВЛЕНИЯ, МИЛИЦИЯ, ОТДЕЛЕНИЕ МИЛИЦИИ, ПРИБУТОК.

СОДЕРЖАНИЕ

Введение 5

1 Анализ и концептуальное моделирование предметной области 6

2 Постановка задачи 15

3 Проектирование баз данных 16

3.1 Построение UML-диаграммы 16

3.2 Построение ER-диаграмм 17

3.3 Построение схемы реляционной базы данных в 3НФ……………..…...…18

4 Описание программы 27

4.1 Общие ведомости 27

4.2 Запуск программы 27

4.3 Предназначение и логическая структура 28

4.4 Описание физической модели базы данных 29

4.5 Описание программной реализации 35

4.5.1 Функции для незарегистрированного пользователя 35

4.5.2 Общие функции для зарегистрированных пользователей

и администратора 36

4.5.3 Функции для администратора 46

4.5.4 Задача автоматизации 47

4.6 Запросы SQL в данной работе 49

Выводы .51

Список литературы 52

ВВЕДЕНИЕ

В наше время развитых электронных технологий высока необходимость в возможности быстро и эффективно оперировать большими, сложно взаимосвязанными объемами данных. Эту возможность предоставляют базы данных. Также к преимуществам баз данных, а именно реляционных баз данных, в которых данные хранятся в виде таблиц, можно отнести компактность, легкую организацию работы с использованием единого языка SQL, возможность надежного совместного доступа.

Целью данной курсовой работы является создание информационной системы «Отделение милиции», которая должна оперировать с достаточно большим объемом данных о заявлениях и пользователях, причем должна делать это довольно быстро. Также программа должна поддерживать гибкий поиск по данным. Реляционные базы данных, как можно убедиться, предоставляют удобные инструменты для воплощения этой цели в жизнь, при этом при разумных затратах времени и ресурсов со стороны разработчика. Именно поэтому они и были выбраны для использования в проекте.

Данный продукт позволяет подать заявление в милицию, а также сохранить их в виде отчёта для дальнейшего использования и печати. Кроме этого, программа имеет большое количество функций.

Также при реализации данного программного обеспечения используется объектно-ориентированный подход, главными принципами которого является инкапсуляция, наследование и полиморфизм. Объектно-ориентированный подход позволяет вести независимую разработку отдельных частей программы, при этом код программы остается легко модифицируемым и наглядным.

1 АНАЛИЗ И КОНЦЕПТУАЛЬНОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ

ОБЛАСТИ

* 1. Анализ предметной области

В данной курсовой работе будет рассмотрена информационная система «Отделение милиции».

Данная система предназначена для подачи пользователями заявлений в полицейский участок. Система позволяет сотрудникам отделения вести базу данных заявлений. Администраторы данного сайта смогут добавлять и изменять информацию, а также получать статистику по работе сотрудников отдtления.

Предметная область «Информационная система отделения милиции» на сегодняшний день сталкивается с такими проблемами:

1. отсутствие возможности быстрого доступа к базе заявлений;
2. отсутствие возможности подачи заявления в милицию в электронном

виде;

1. отсутствие возможности видеть реальные показатели работы

сотрудников отделения;

1. большой объём информации до сих пор хранится в бумажном виде,

что создаёт трудности в работе.

Необходимость разработки данной информационной системы и базы данных обусловлена тем, что для поиска нужной информации по заявлениям требуется много времени, что замедляет процесс раскрытия преступления. Разработка данной системы облегчит жизнь работникам отделений милиции, т.к. позволит очень быстро регистрировать заявления в базе, а также искать данные о заявлениях.

Целью данной курсовой работы является проектирование и разработка информационной системы «Отделение милиции».

Разработанная информационная система предназначена для автоматизирования приведенных выше проблем и значительно упростит процесс работы с заявлениями пользователей.

Функциональная структура схематически представлена на рисунке 1.1. Данная структура описывает основные взаимодействия, которые происходят в процессе пользования информационной системой. Незарегистрированный пользователь должен иметь возможность зарегистрироваться или войти в систему. Зарегистрированный пользователь должен иметь возможность заполнить заявление. Сотрудник отделения должен иметь возможность просматривать и принимать заявления. Администратору данного сайта доступны функции добавления, редактирования и удаления сотрудников.

Организация работы данной системы является комплексной областью деятельности. В таком случае функциональными объектами системы являются администраторы, сотрудники, пользователи и сама программа (см.рис. 1.1).



Рисунок 1.1 – Функциональная структура информационной системы «Отделение милиции»

* 1. Описание информационных потребностей пользователей

Информационные потребности пользователей данной информационной следующие.

Для незарегистрированных пользователей:

1. регистрация на сайте;
2. вход на сайт.

Для зарегистрированных пользователей:

1. заполнение заявления в милиции;
2. просмотр своих заявлений;
3. изменение своих данных.

Для сотрудников:

1. просмотр заявлений;
2. приём заявлений.

Для администраторов:

1. просмотр данных сотрудников;
2. удаление сотрудников;
3. получение статистики.

Документооборот предметной области «Отделение милиции» состоит из:

1. заявлений пользователей. Состоит из самого заявления пользователя.
2. данных о преступниках. Состоит из полной информации о преступнике.

Далее приведено описание объектов данной информационной системы и описание связей между ними.

Обобщенная схема взаимодействия объектов ПО изображается на рисунке 1.2. Рассмотрим ее подробней.

1.3 Описание объектов и связей между ними

Описание объектов ПО и их атрибутов:

1. пользователь: id пользователя, фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, контактный номер телефона, email-адрес, логин, зашифрованный пароль;
2. сотрудник: id сотрудника, фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, номер удостоверения, контактный номер телефона, email-адрес, логин, зашифрованный пароль;
3. администратор: id администратора, фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, контактный номер телефона, логин, зашифрованный пароль;
4. заявление: id заявление, текст заявления, id пользователя, id работника, просмотрено, принято, дата.

1.4 Описание функциональной структуры системы

Изучая предметную область, я выделил, что функциональными элементами системы являются пользователь, работник отделения и заявление.

В системе будут реализованы следующие функции:

1. добавление, удаление, редактирование заявлений;
2. получение информации о пользователях;
3. составление отчёта за день/неделю/месяц/год, в котором будут указаны: номера работников, количество просмотренных заявлений, количество принятых;
4. формирование накладной на поставку продукта к продавцам, где будут указаны: имена заказчика и поставщика, название товара, его цена, количество и общая сумма сделки, расчетные счета заказчика и поставщика;
5. информирование пользователя о том, что поданное им заявление проверено или принято;
6. сортировка и фильтрация заявлений по дате;
7. поиск заявления по фамилии, имени и(или) отчеству автора;
8. получение статистики заявлений (общее количество заявлений, количество просмотренных, количество принятых);
9. получение статистики пользователей (информация о пользователей, количество заявлений всего, количество просмотренных, количество принятых)
10. получение статистики работников (информация о работнике, количество просмотренных заявлений, количество принятых заявлений).

Задача автоматизации:

а) в приложении реализована функция автоматического уведомления пользователя на e-mail адрес в случае изменения состояния заявления;



Рисунок 1.2 – Схема взаимосвязи объектов ПО

1.5 Описание алгоритмических зависимостей

1. у каждого заявления есть автор;
2. у каждого просмотренного заявления есть сотрудник, который его просмотрел;
3. у каждого принятого заявления есть сотрудник, который его просмотрел и принял;
4. MD5-шифрование паролей для пользователей системы.

1.6 Описание ограничений целостности

В данной предметной области присутствуют такие ограничения целостности для БД:

1. у каждого заявления есть пользователь, который его оставил;
2. у каждого заявления либо есть, либо нет id сотрудника который его смотрел;

1.7 Требования к ИС и существующие ограничения

Для выполнения поставленных задач и целей будет создана информационная система «Отделение милиции» средствами языка программирования Javascript в виде веб-приложения на основе технологии Node.js. Приложение будет разработано в среде WebStorm 2016 с использованием СУБД PostgreSQL 9.3.

2 ПОСТАНОВКА ЗАДАЧИ

Необходимо организовать процесс регистрации пользователя с внесением в базу данных информации о нём, а также с последующим входом в систему. Существует следующая информация о пользователе: фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, контактный номер телефона, email-адрес. Также пользователь после регистрации имеет свой логин и пароль. Так как целью рассматриваемого проекта является упрощение принятия и рассмотрения заявлений пользователей, разрабатываемая система должна иметь высокую надежность, интуитивно понятный интерфейс и выполнять функции, являющиеся задачами автоматизации в данной системе.

После заполнения регистрационной формы, она проходит проверку системой, и либо сохраняется в базу и создается соответствующий пользователь, либо сообщается об отказе в регистрации.

После регистрации пользователи имеют возможность заполнить заявление. Сотрудники после входа в систему имеют возможность работать с заявлениями пользователей и присваивать им разные статусы. Администраторы после входа в систему имеют возможность добавления и удаления сотрудника, получать статистику заявлений по датам. Система должна выполнять следующие задачи:

1. внесение информации о пользователе;
2. подтверждение регистрации пользователем;
3. редактирование информации о собственном профиле;
4. поиск заявлений по автору;
5. сортировка заявлений по дате и состоянию заявления;
6. фильтрация заявлений по дате и состоянию;
7. изменение множества сотрудников администратором;
8. создание заявления;
9. отмена заявления;
10. печать заявления;
11. возможность обратной связи с администратором;
12. шифрование паролей, используемых пользователями сайта;
13. отправка пользователю e-mail сообщения при изменении состояния его заявления;
14. отчёт в виде заявления пользователя;
15. отчёт о работе сотрудника;
16. статистика подачи заявлений пользователем;
17. статистика количества заявлений по дате;
18. статистика по пользователям системы.

3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

3.1 Построение UML-диаграммы

В данной системе присутствуют 4 вида актеров, каждый из которых играет отведенную ему роль. Актер незарегистрированный пользователь имеет возможность внести свои данные в систему для сохранения. Данный элемент USE CASE будет включать валидацию данных. Незарегистрированный пользователь также может просматривать информацию о преступниках.

Актер зарегистрированный пользователь имеет, помимо возможностей незарегистрированного пользователя ещё и возможность заполнять заявления. Также он может вносить дополнительные данные о себе или изменять персональные данные.

Актер сотрудник помимо возможностей зарегистрированного пользователя ещё имеет возможно работать с данными заявлений.

Актер администратор имеет кроме тех возможностей, которые имеет сотрудник, ещё и возможность добавления и удаления информации о сотрудниках, получения различным статистик.

На основе этого составим USE CASE диаграмму предложенной системы (см.рис. 3.1).

Данная информационная система будет иметь трехуровневую архитектуру. Это позволит улучшить структуру системы и повысить ее безопасность. Учитывая архитектуру системы, можно выделить три основных узла, которые присутствуют в ней: «Клиент», «Сервер», «СУБД».

Узел «Клиент» отвечает за вывод необходимой информации конечному пользователю и так же за предоставление пользователю удобного интерфейса для взаимодействий с системой в целом. В узле «Сервер» будет выполняться вся основная логика по работе системы. И последний узел отвечает за хранение внесенных данных.



Рисунок 3.1 – USE CASE диаграмма «Информационная система «Отделение милиции»»

«Клиент» и «Сервер» связаны между собой напрямую. А «Сервер» и «СУБД» связаны между собой через посредника. В качестве посредника выступает DAO Layer. Он позволяет серверу абстрагироваться от типа СУБД, сохранять и загружать объекты, хранящиеся в базе данных, без информации о том, как эти данные хранятся в базе данных, и вообще какая база данных используется. На основе всей информации, приведенной выше можно построить диаграмму развертывания для данной информационной системы (см.рис. 3.2).

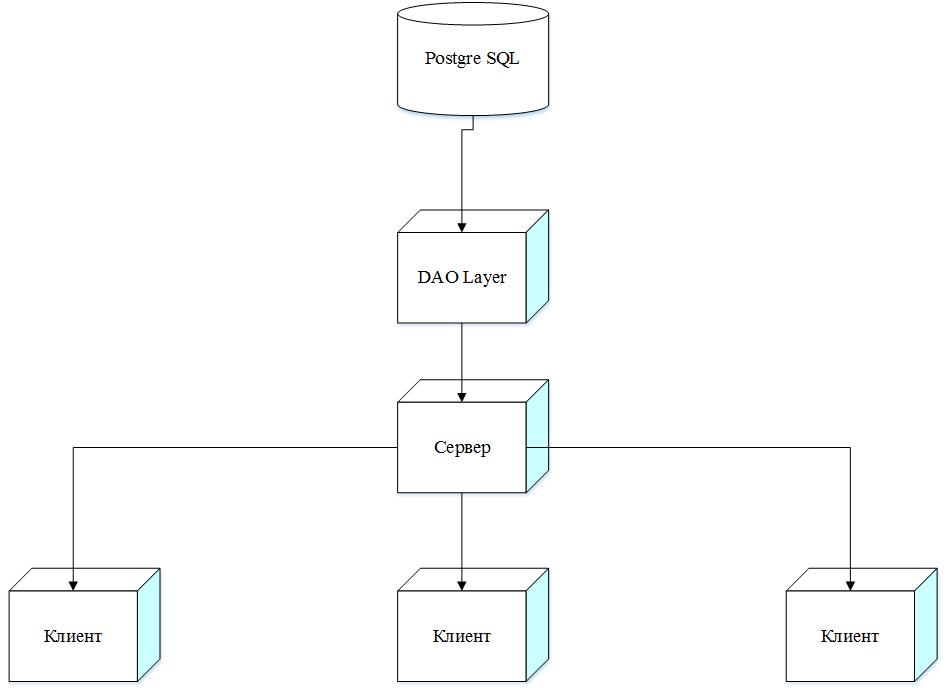


Рисунок 3.2 – Диаграмма развертывания «Системы бронирования билетов на соревнования»

3.2 Построение ER-диаграммы

1. пользователь: id пользователя, фамилия, имя, отчество, дата

рождения, домашний адрес, номер паспорта, контактный номер телефона, email-адрес, логин, зашифрованный пароль, id заявления;

1. сотрудник: id сотрудника, фамилия, имя, отчество, дата рождения,

домашний адрес, номер удостоверения, контактный номер телефона, email-адрес, логин, зашифрованный пароль, id преступника;

1. заявление: id заявления, id пользователя, id сотрудника, текст заявления, просмотрено, принято, дата создания.

На основании концептуальной модели предметной области, которая была получена в первом разделе можно получить такие сущности:

1. стержневая сущность «Пользователь» с атрибутами фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, контактный номер телефона, email-адрес, логин, зашифрованный пароль. Первичный ключ – id\_пользователя;
2. стержневая сущность «Сотрудник» с атрибутами фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, номер удостоверения, контактный номер телефона, email-адрес, логин, зашифрованный пароль. Первичный ключ – id\_сотрудника;
3. зависимая сущность «Заявление» с атрибутом текст заявления, id\_пользователя, id\_сотрудника, текст заявления, просмотрено, принято, дата создания заявления. Первичный ключ – id\_заявления;
4. стержневая сущность «Администратор» с атрибутами фамилия, имя, отчество, дата рождения, домашний адрес, номер паспорта, контактный номер телефона, email-адрес, логин, зашифрованный пароль. Первичный ключ – id\_администратора.

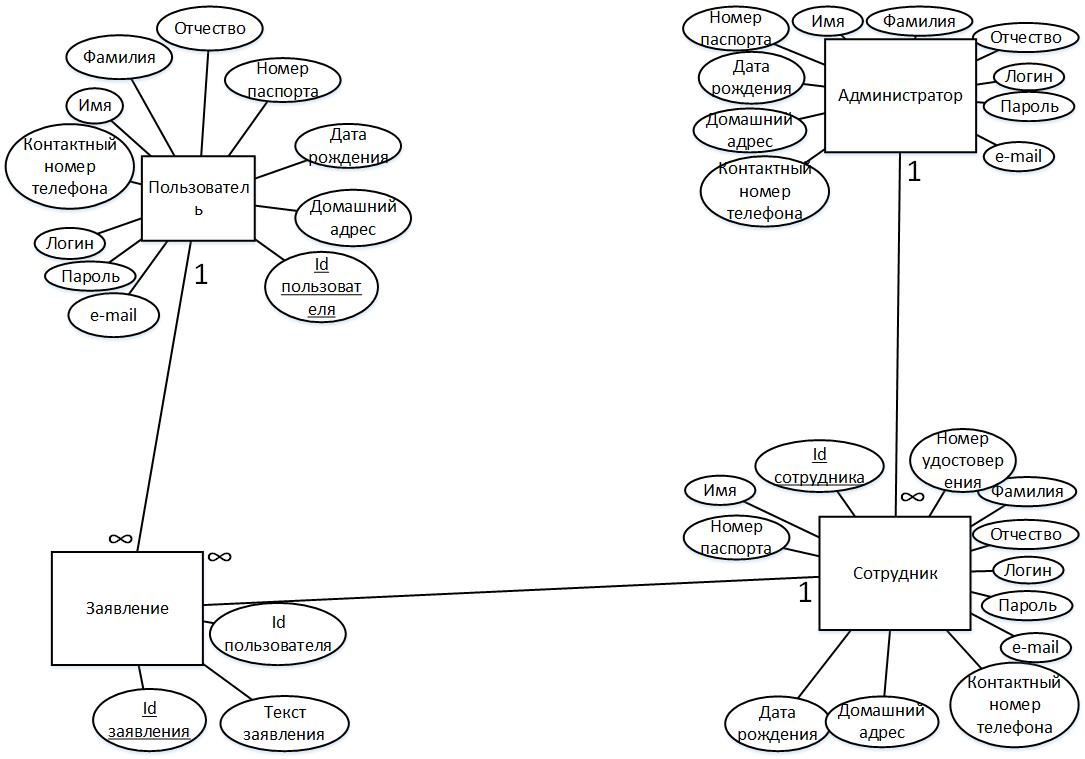


Рисунок 3.3 – ER-диаграмма предметной области «Отделение милиции»

3.3 Построение схемы реляционной базы данных в 3НФ

Нормализация - разделение таблицы на две или более с целью исключения избытка и избегания аномалий баз данных [3].

1НФ представляет собой отношения. Для него характерны атомарность значений на пересечении столбцов и строк, кортежи и атрибуты должны быть неупорядоченные и кортежи не дублированы.

Для 2НФ характерные особенности 1НФ и полная функциональная зависимость не ключевых атрибутов от первичного ключа отношения.

3НФ - состояние отношения, которое находится в 2НФ и в котором между атрибутами нет транзитивных зависимостей.

Переход к следующей нормальной форме возможен только тогда, когда удовлетворяются все условия предыдущей формы. Для предметной области построим универсальное отношение, включающее описание ранее выделенных объектов и их свойств (см. рис. 3.4).

На основании этого разобьем отношение в отношения, находящиеся в 3НФ и изображенные на рисунке 3.5, рисунке 3.6, рисунке 3.7, рисунке 3.8.

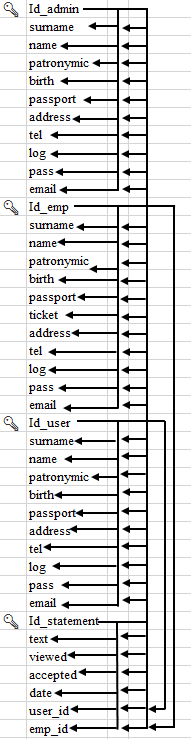


Рисунок 3.4- Универсальное отношение базы данных

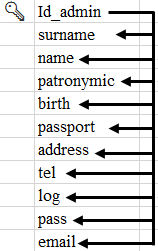


Рисунок 3.5-таблица admins

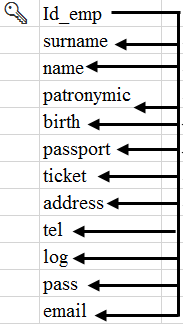
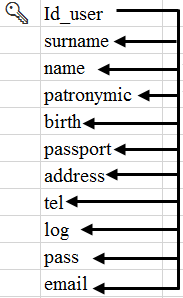


Рисунок 3.6-таблица employees



­

Рисунок 3.7-таблица users

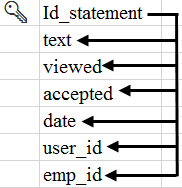


Рисунок 3.7-таблица statements

4 ОПИСАНИЕ ПРОГРАММЫ

4.1 Общие сведения

Данный продукт был реализован в виде веб-сайта, с серверной частью на языке JavaScript с помощью фреймворка Node.js и клиентской частью на языке разметки HTML5 с использованием генератора шаблонов Pug, генератора каскадных таблиц стилей Stylus и JavaScript. Используется СУБД PostgreSQL 9.3, доступ к которой осуществляется с помощью библиотеки pg. Также используются библиотеки Bootstrap, JQuery.

Итоговая программа занимает 35 мегабайт. Для запуска программы на персональном компьютере потребуется операционная система Windows, Linux, или Mac OS X, а также Node.js версии 7.3.0.

Для установки программного продукта потребуется сервер PostgreSQL.

Требования к операционной системе, памяти и процессору:

Память: не менее 4 ГБ по мере роста размера базы данных;

Процессор: процессор х86: 1,0 ГГц;

Тип процессора: совместимость с Pentium III или выше.

4.2 Вызов и загрузка

Приложение поставляется в папке PoliceStation.

Внутри папки находится папка bin, в ней находится главный файл-www, являющийся точкой входа. Для запуска необходимо находясь в главной папке выполнить команду «node bin/www». При первом запуске следует зарегистрировать администратора.

Создание в локальной директории пользователя и подключение базы данных происходит автоматически при старте программы.

4.3 Предназначение и логическая структура

В постановке задачи, описанной выше, указаны все пункты, которые являются обязательными для выполнения. Полученная программа соответствует этим требованиям целиком и полностью.

Вкладка браузера с адресом этого сайта является единой рабочей областью, на которой присутствуют различные элементы в зависимости от страницы сайта, на которой находится пользователь.

4.4 Описание физической модели базы данных

В качестве СУБД была выбрана PostgreSQL, поскольку что она является одной из самых удобных и надежных платформ баз данных. Кроме этого данная СУБД поддерживается Node.js, поэтому описание физической модели представлено со скриншотами данной программы и описанием.

Структура таблицы users (см. рис. 4.1). На данном рисунке показаны все атрибуты таблицы users, первичным ключом является id, все поля не могут принимать значение NULL. Внешним ключом является id, который соединяет данную таблицу с таблицей statements связью один ко многим, один пользователь может составить много заявлений.

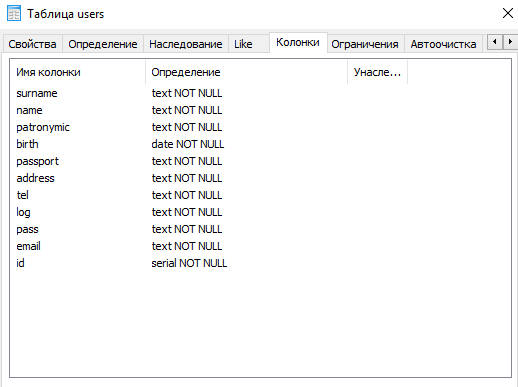


Рисунок 4.1 – Структура таблицы users

Структура таблицы employees (см. рис. 4.2). Первичным ключом является Id, все атрибуты не принимают значение NULL. Внешним ключом является id, который соединяет данную таблицу с таблицей statements связью один ко многим, один работник может проверить или принять много заявлений.

Структура таблицы admins (см. рис. 4.3). Первичным ключом является Id, все атрибуты не принимают значение NULL.

Структура таблицы statements (см. рис. 4.4). Первичным ключом является Id, все атрибуты не принимают значение NULL, кроме emp\_id.

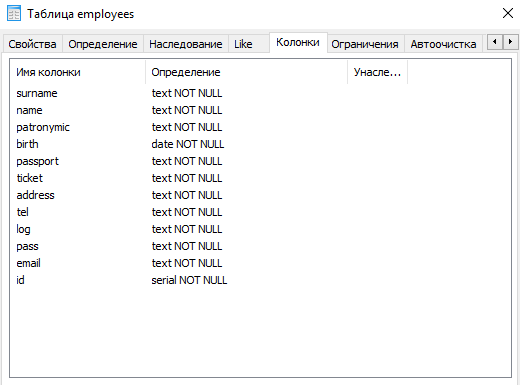


Рисунок 4.2 – Структура таблицы employees

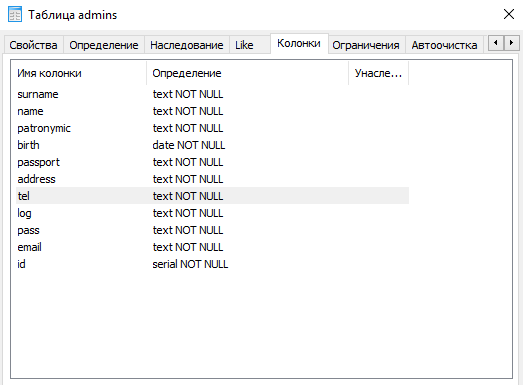


Рисунок 4.3 – Структура таблицы admins

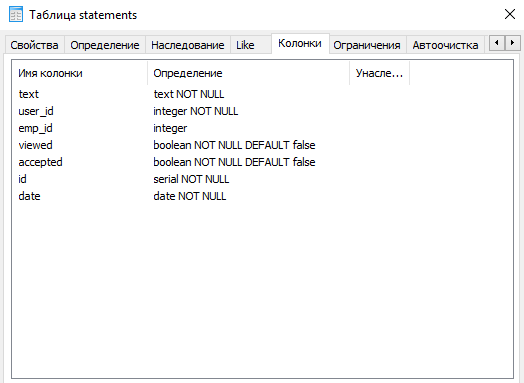


Рисунок 4.4– Структура таблицы statements

4.5 Описание программной реализации

4.5.1 Функции для незарегистрированного пользователя

При переходе на сайт пользователь увидит главную страницу сайта (см. рис. 4.8). Сверху справа, на панели навигации имеются кнопки «Вход» и «Регистрация». При нажатии на кнопку «Вход» откроется форма входа, предназначенная для всех видов пользователей системы (см. рис. 4.9). Если все данные были введены верно, то пользователь будет переадресован на страницу с заявлениями. При нажатии на кнопку «Регистрация» открывается форма регистрации (см. рис. 4.10) предназначенная исключительно для обычных пользователей сайта, где пользователь должен заполнить все поля, представленные на рисунке. Далее нажать кнопку «Зарегистрировать». После чего произойдёт переадресация пользователя на страницу где он может оставить заявление. В случае случайного попадания на форму регистрации, есть кнопка с названием проекта, которая так же вернет пользователя на главную страницу.

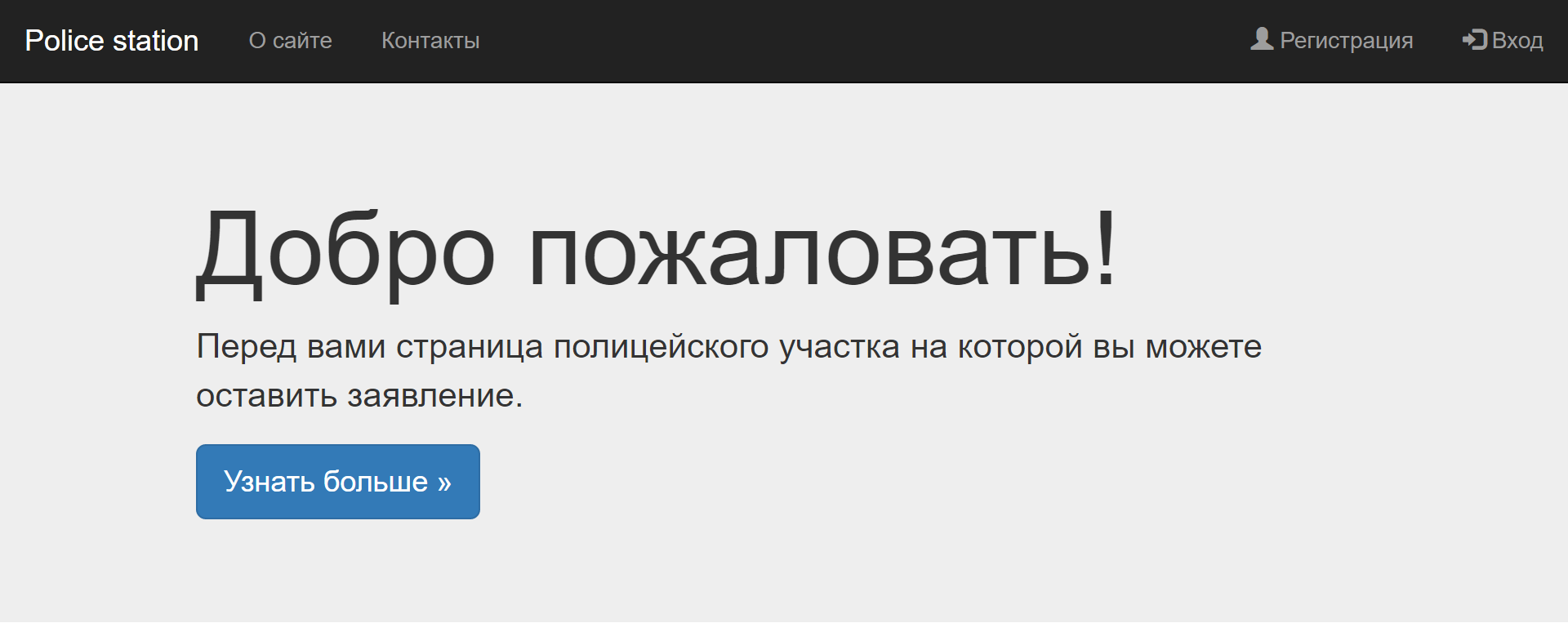


Рисунок 4.8 – Главная страница

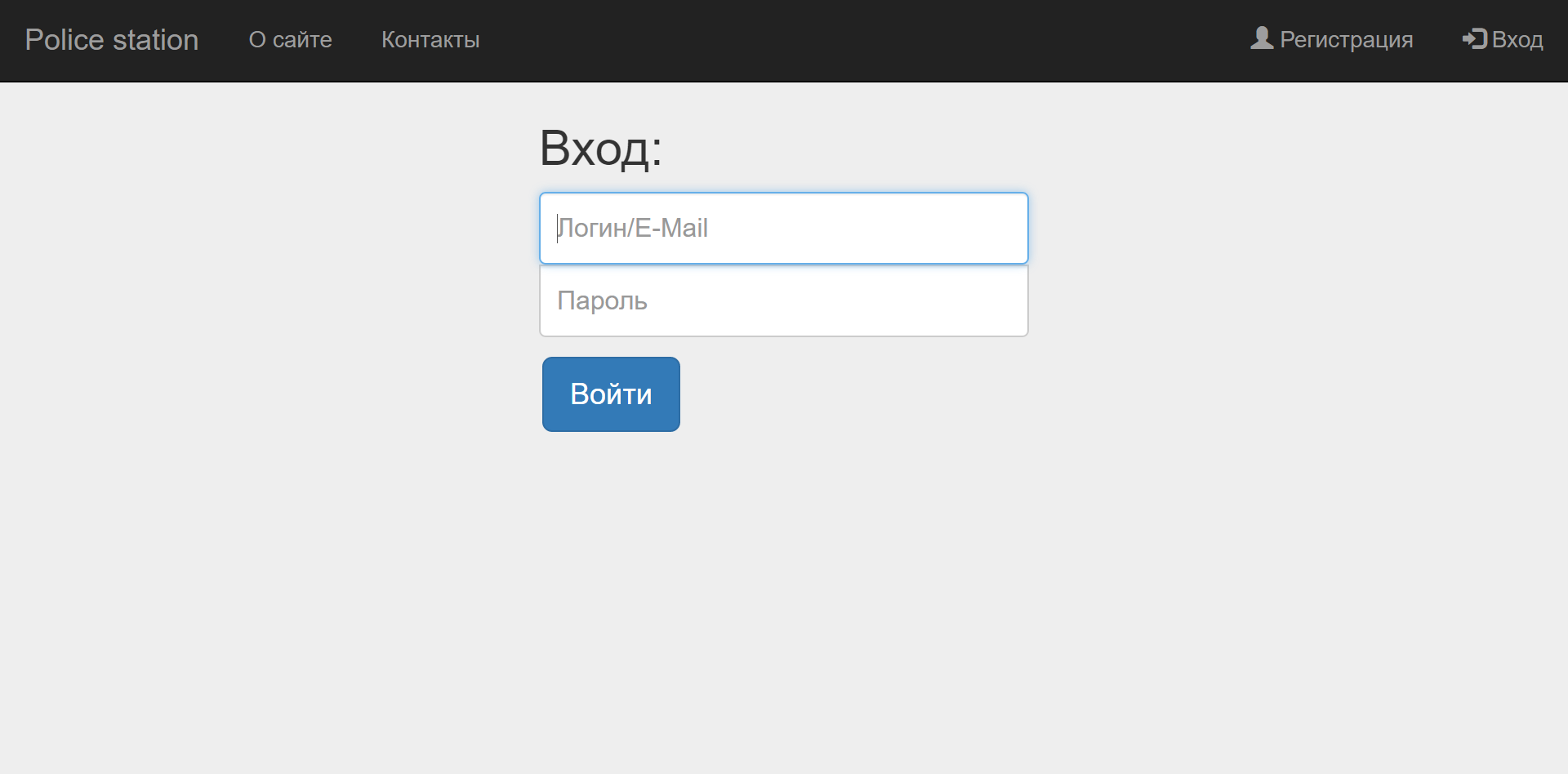


Рисунок 4.9 – Страница входа

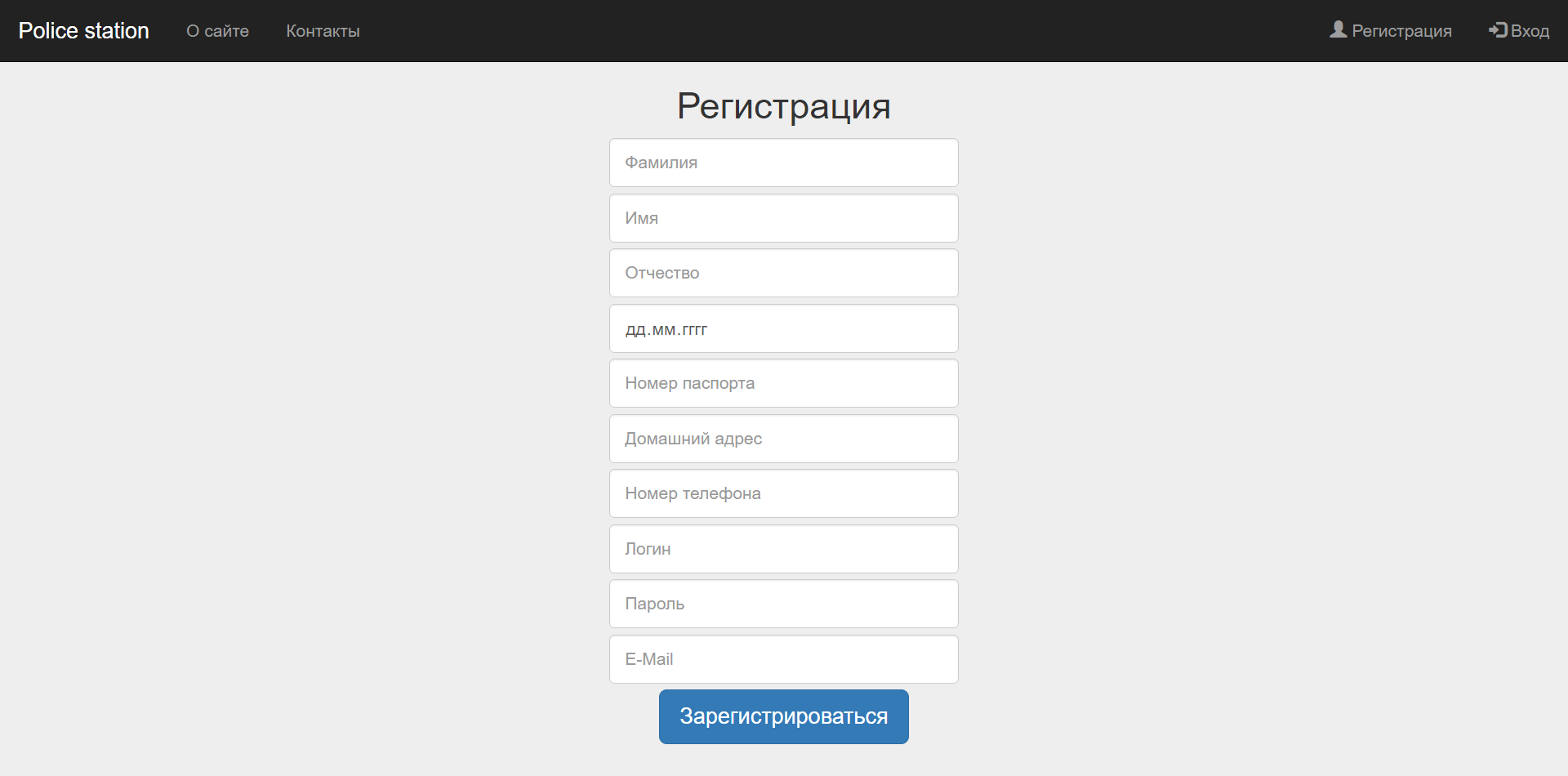


Рисунок 4.10 – Страница регистрации

4.5.2 Функции для зарегистрированных пользователей

При переходе на сайт пользователь увидит главную страницу сайта (см. рис. 4.8). При нажатии на кнопку «Вход» откроется форма входа, предназначенная для всех видов пользователей системы (см. рис. 4.9).

Если данные введены правильно пользователь будет переадресован на страницу с его заявлениями (см. рис. 4.11). Первая функция, которую видит пользователь — это различные действия с его заявлениями. Одно из возможных действий-получения версии заявления для печати. Сделать это можно нажав на значок принтера, находящийся под текстом заявления. После нажатия происходит переадресация и пользователь видит на экране версию заявления для печати (см. рис. 4.12).

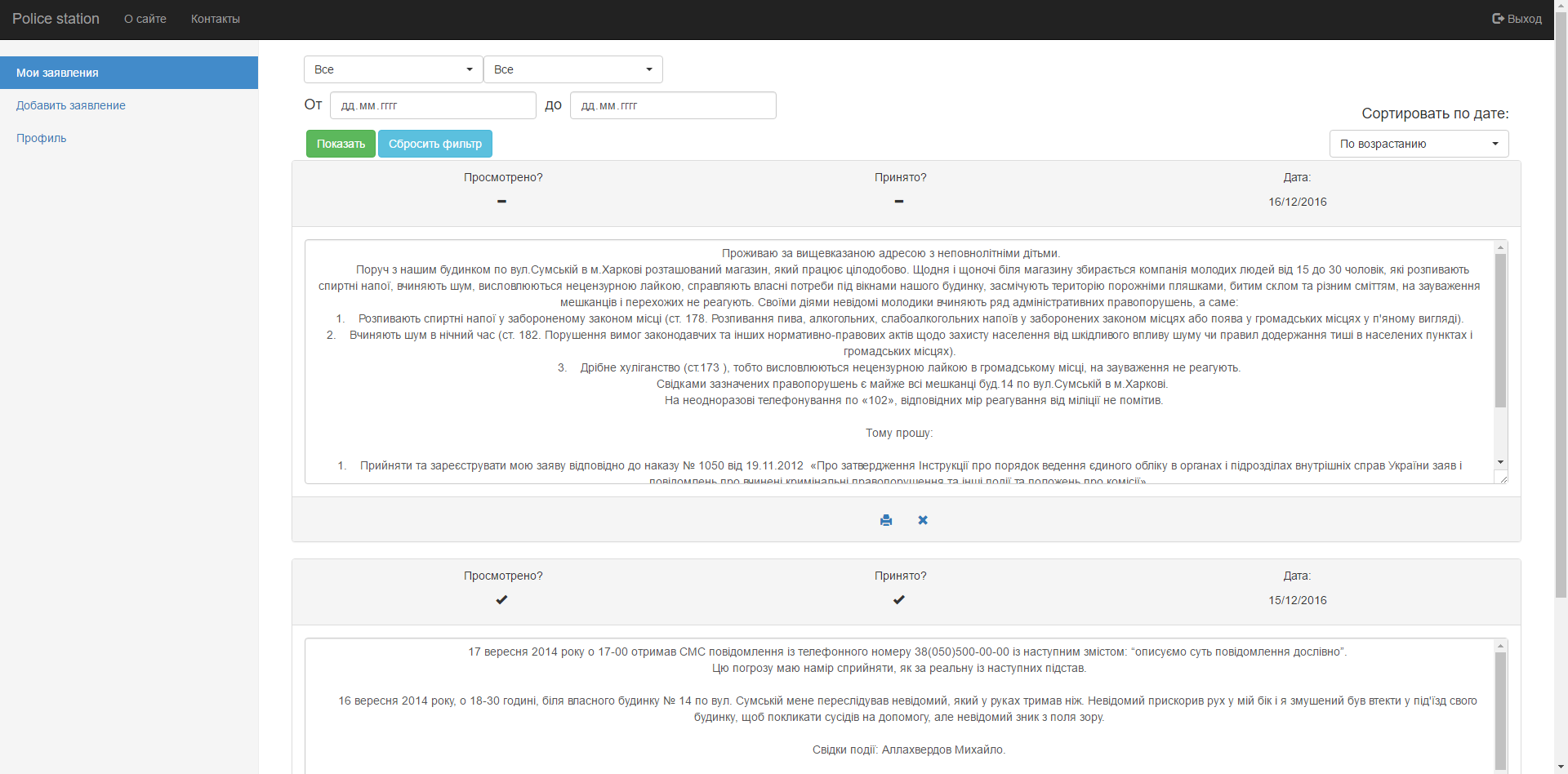


Рисунок 4.11 – Страница с заявлениями

Если пользователь хочет добавить новое заявление, ему нужно нажать на кнопку «Добавить заявление» в боковом меню. После нажатия на кнопку будет выполнена переадресация на страницу добавления заявления (см. рис. 4.13). После добавления заявления пользователь будет переадресован на страницу с его заявлениями, где будет отображено добавленное им заявление.

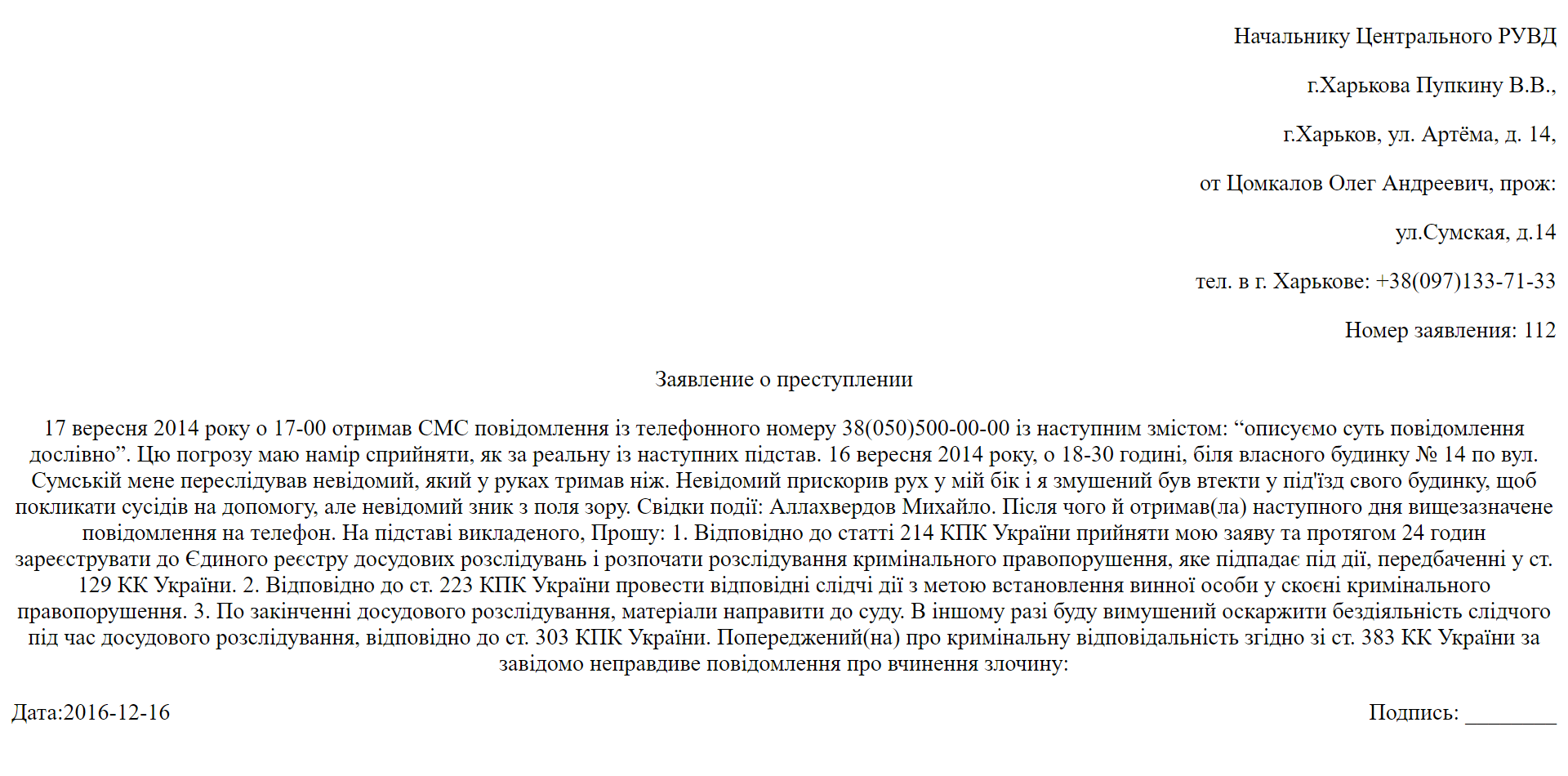


Рисунок 4.12 – Страница с версией заявления для печати

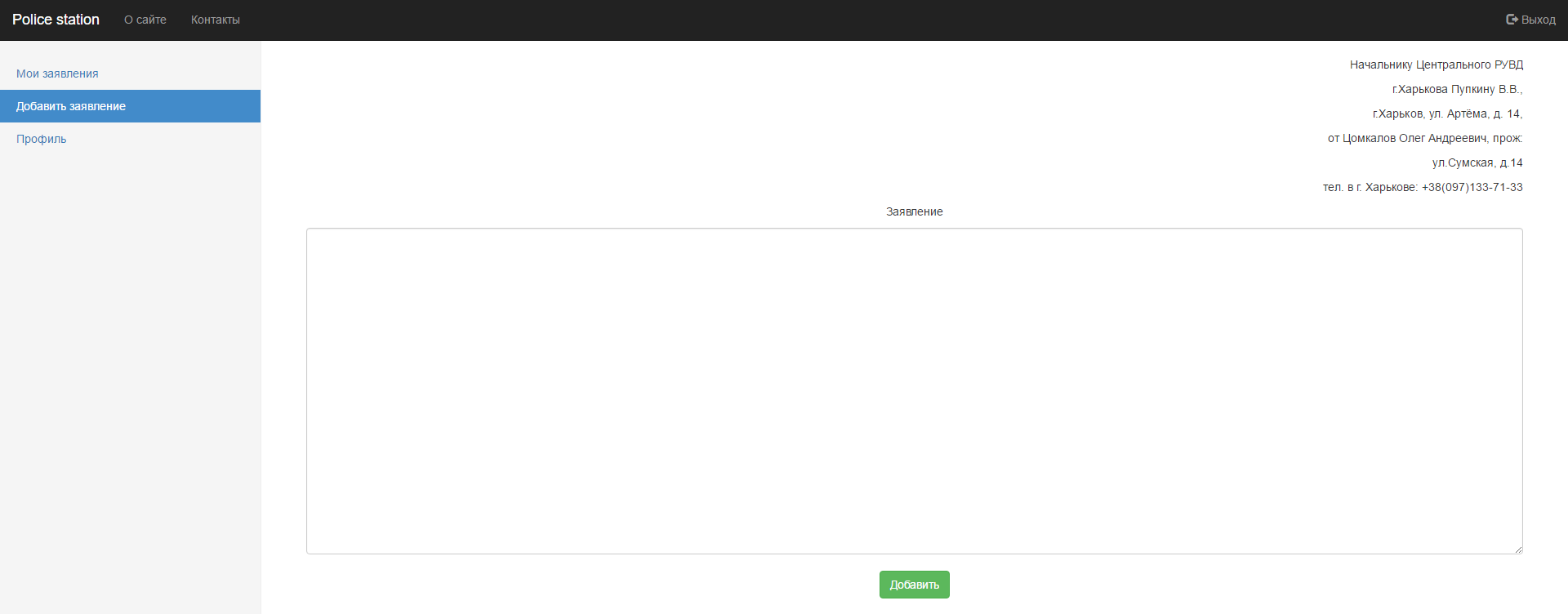


Рисунок 4.13 – Страница добавления заявления

Следующая функция-это изменение пользователем данных о себе. При нажатии в боковом меню на кнопку «Профиль» откроется вкладка (см. рис. 4.14), на которой пользователь может изменить информацию о себе.

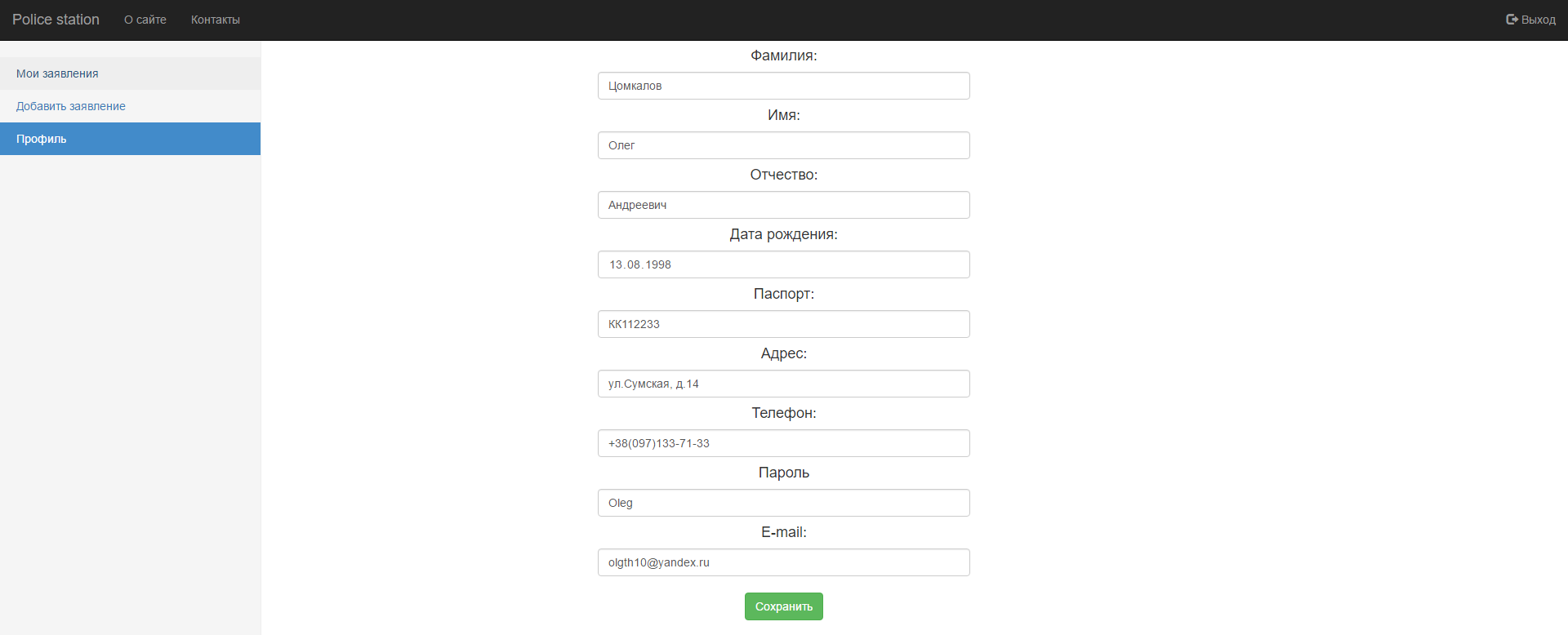


Рисунок 4.14 – Страница редактирования данных пользователя

4.5.3 Функции работника отделения

Функции просмотра и приёма заявлений доступны только работникам отделения. Вход в аккаунт работника осуществляет через страницу (см. рис. 4.9). После входа в аккаунт работник будет переадресован на страницу с заявлениями всех пользователей (см. рис. 4.15). Над заявлениями пользователей имеются элементы для поиск нужных заявлений, что упрощает нахождения нужных заявлений. Также под текстом каждого заявления имеются две кнопки: отметить заявление как просмотренное и отметить заявление как принятое. Если заявление отмечено только как принятое значит пользователю необходимо либо изменить заявление, либо прийти с его распечаткой в отделение. После изменения работником статуса заявления, пользователю на электронную почту будет выслано письмо с уведомлением об этом.

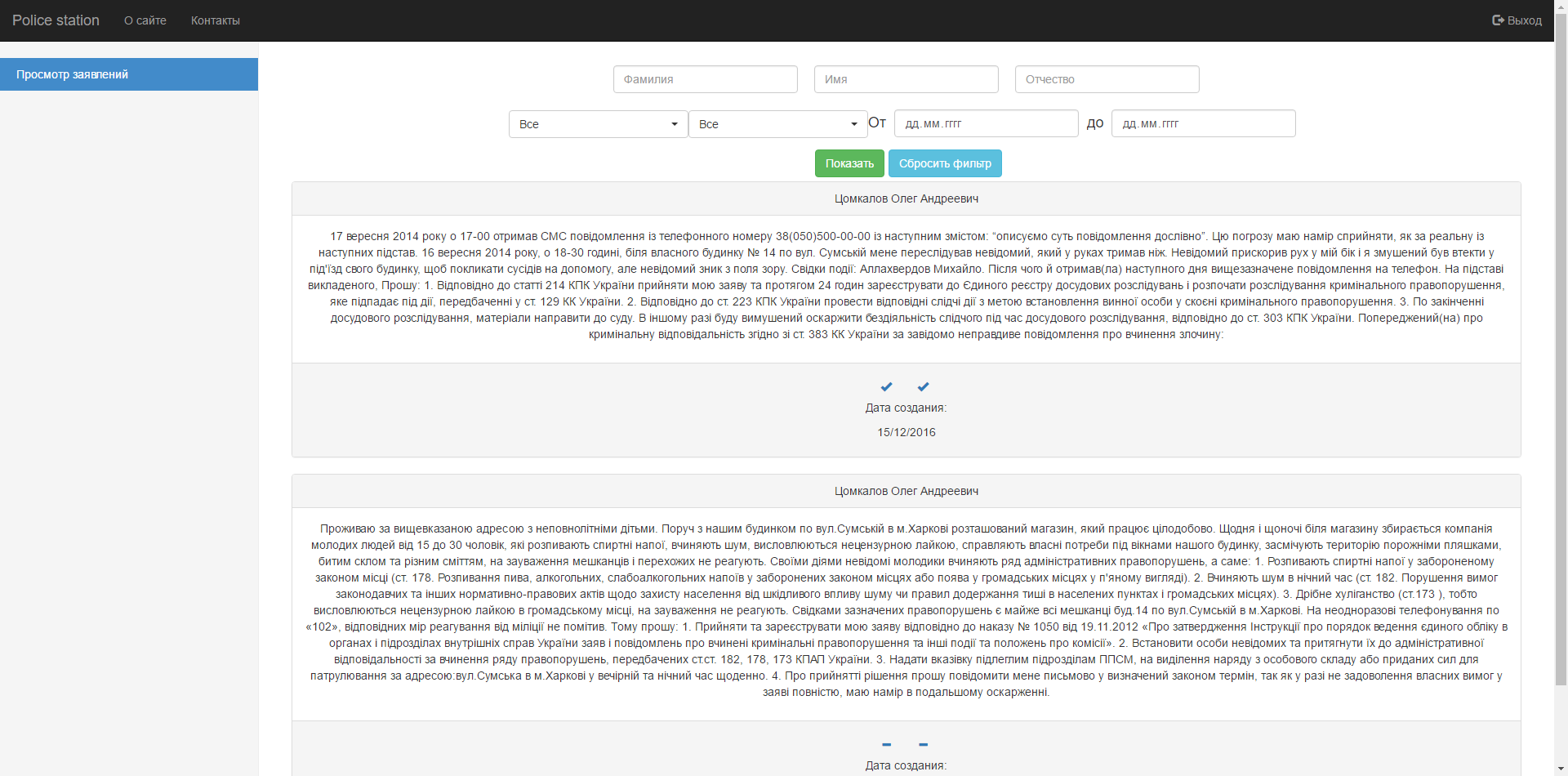


Рисунок 4.15 – Страница заявлений пользователей

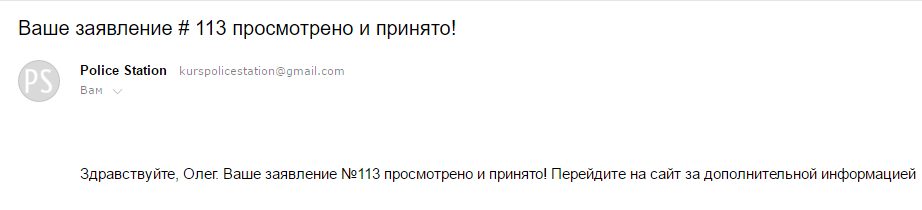


Рисунок 4.16 – Уведомление на e-mail пользователя об изменении статуса заявления

* + 1. Функции администратора

Функции добавления, редактирования и удаления сотрудников, а также получения статистики и запросы к БД доступны только адинистраторам отделения. Вход в аккаунт администратора осуществляет через страницу (см. рис. 4.9). После входа в аккаунт администратор будет переадресован на страницу со списком всех работников (см. рис. 4.17).

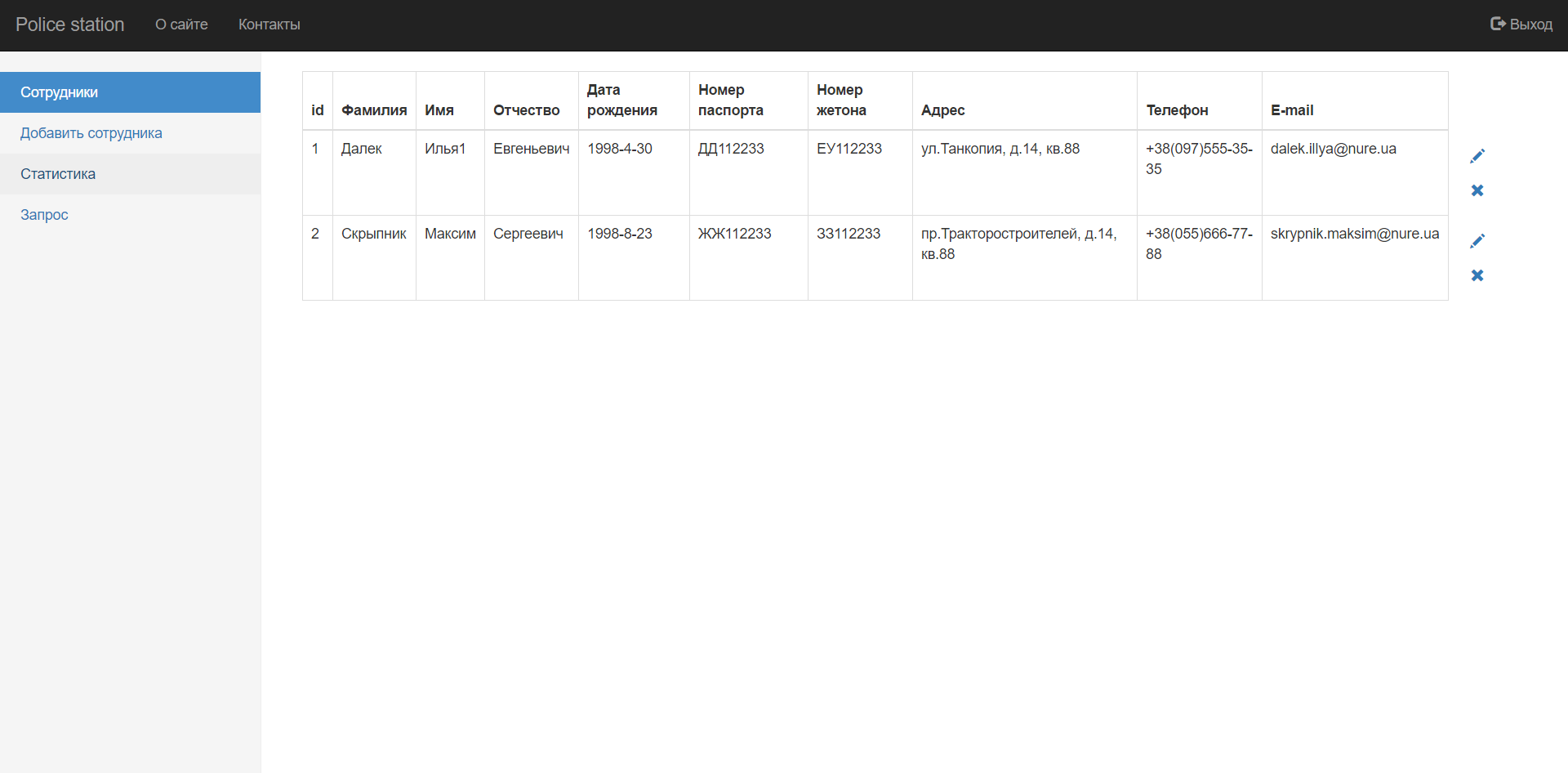


Рисунок 4.17 – Список всех работников отделения

Напротив каждого из работников имеется две кнопки: в виде карандаша-редактировать работника и в виде крестика-удалить работника. При нажатии на первую кнопку произойдёт переадресация на страницу для редактирования данных работника (см. рис. 4.18). При нажатии на вторую кнопку, если работник не связан ни с одним заявлением-произойдёт его удаление, в противном случае будет выведена ошибка.

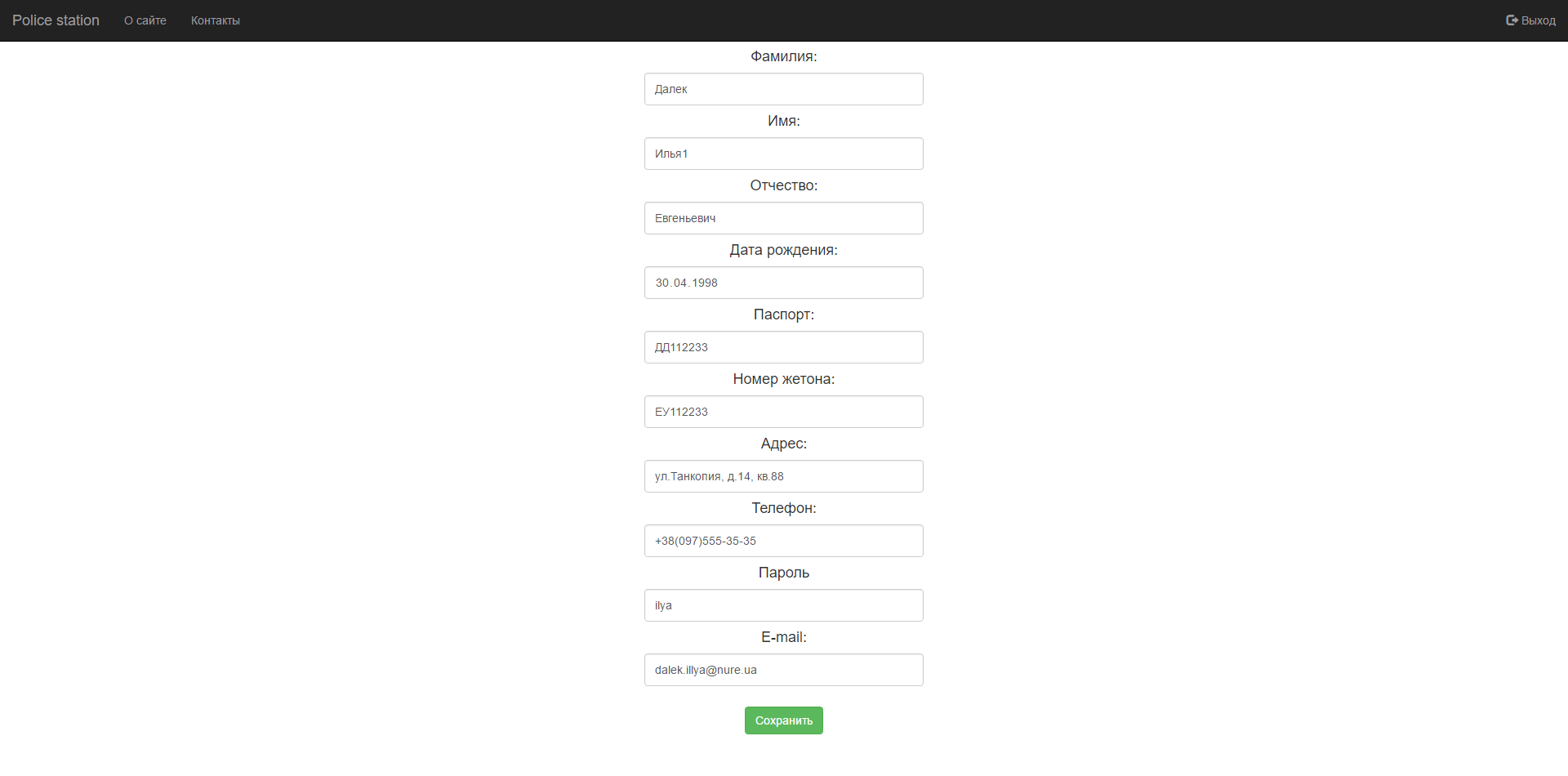


Рисунок 4.18 – Страница редактирования данных сотрудника

При нажатии в боковом меню на кнопку «Добавить сотрудника» произойдёт переадресация на страницу добавления сотрудника (см. рис. 4.19).

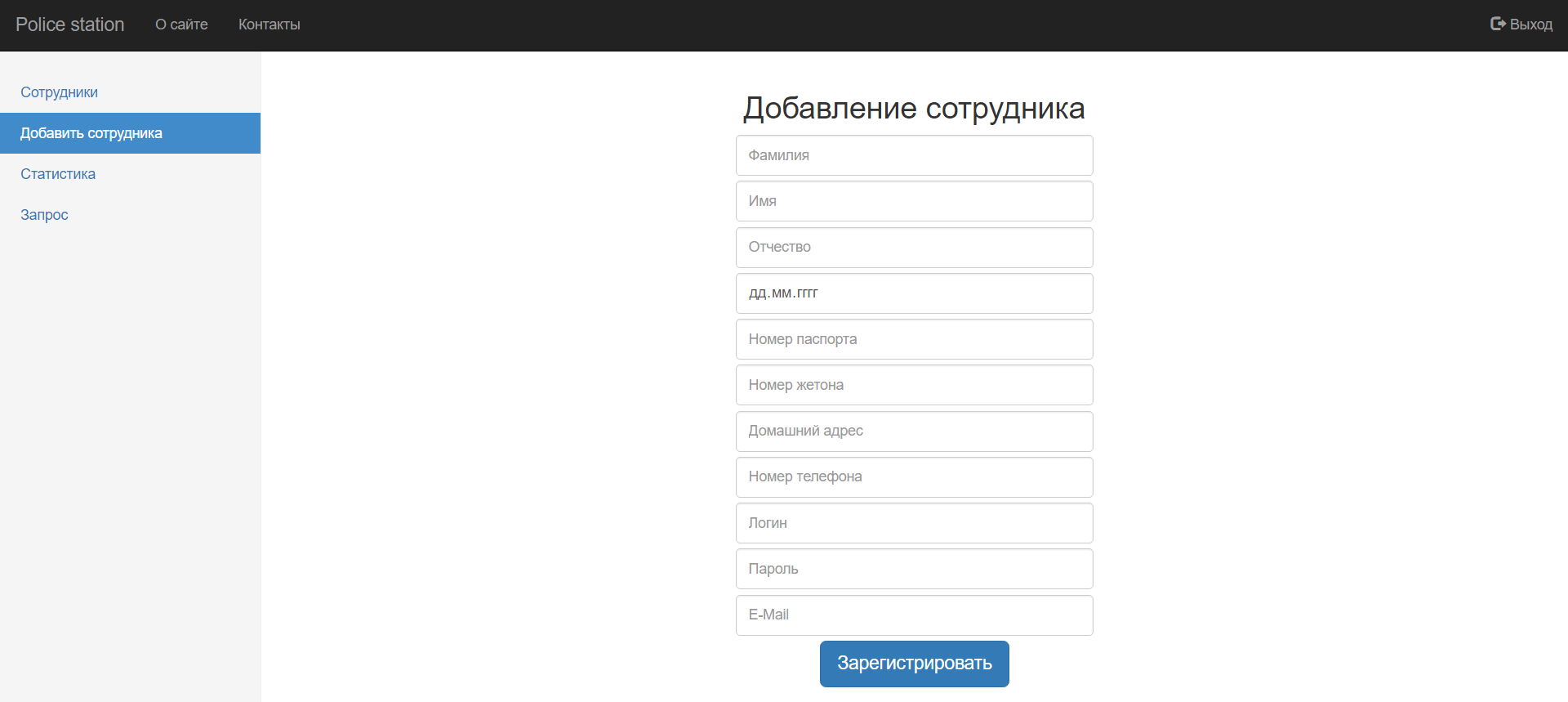


Рисунок 4.19 – Страница редактирования данных сотрудника

При нажатии в боковом меню на кнопку «Статистика» произойдёт переадресация на страницу со статистикой (см. рис. 4.20), где можно увидеть общую статистику и получить статистику по количеству заявлений пользователя за определённый промежуток времени.

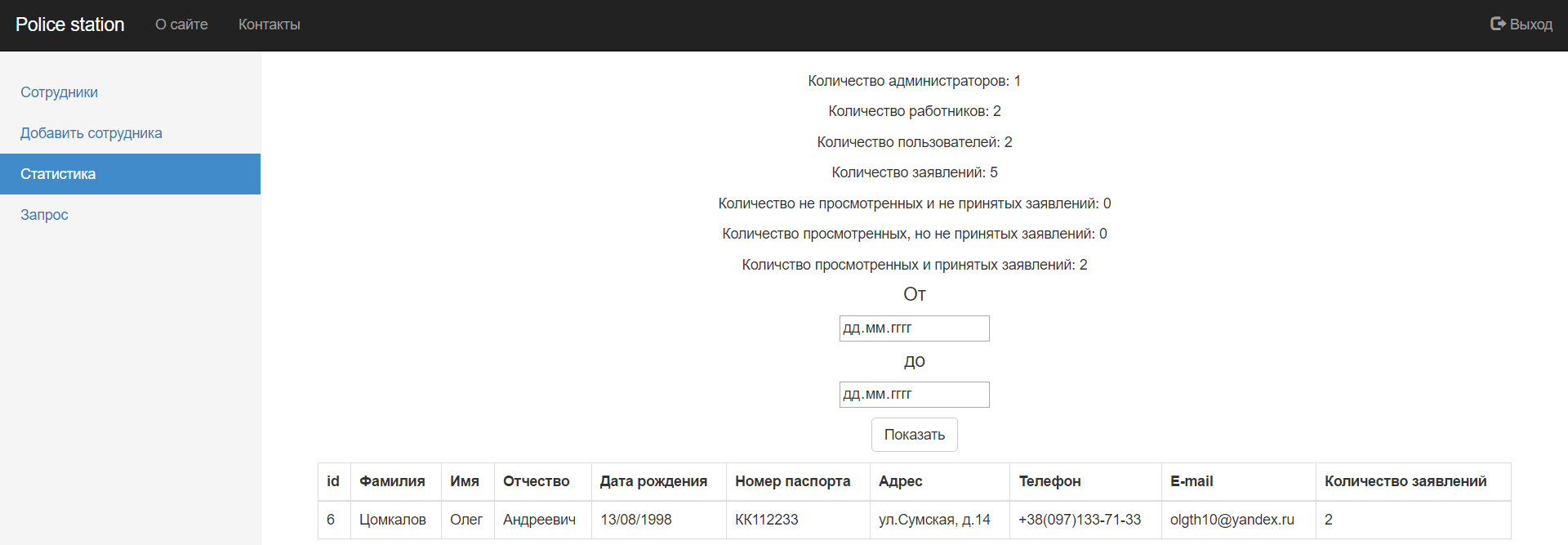


Рисунок 4.20 – Страница со статистикой

При нажатии в боковом меню на кнопку «Запрос» произойдёт переадресация на страницу с полем для запроса (см. рис. 4.20), где можно выполнить разные запросы к БД и увидеть их результат.

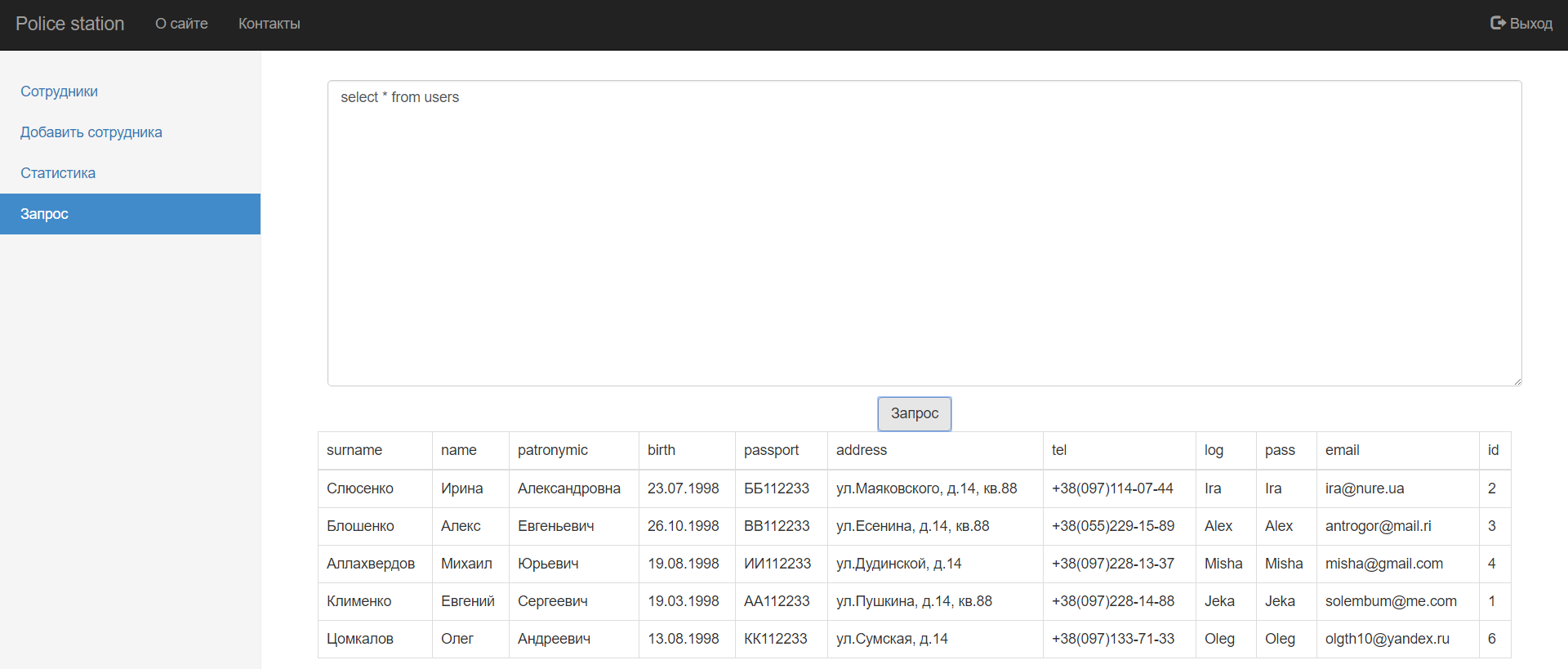


Рисунок 4.21 – Страница c полем для запроса к БД

4.5.4 Задача автоматизации

В качестве задачи автоматизации была выбрана функция отправки пользователю письма на e-mail указанный при регистрации с информацией об изменении состояние его заявления (см. рис. 4.16).

**const** nodemailer=*require*(**'nodemailer'**);

**let** client=*require*(**'../../db'**);  
**const** smtpConfig = {  
 **host**: **'smtp.gmail.com'**,  
 **port**: 465,  
 **secure**: **true**,  
 **auth**: {  
 **user**: **'kurspolicestation@gmail.com'**, *// Your email id* **pass**: **'we34n8xza'** *// Your password* }};  
**const** transporter=nodemailer.createTransport(smtpConfig);

client.query(**"SELECT name,email FROM users WHERE id=$1"**,[arg1],(err,result)=>{  
 **if** (err){  
 callback(500)  
 }  
 **else**{  
 **let** mailOptions = {  
 **from**: **'"Police Station" <kurspolicestation@gmail.com>'**, *// sender address* **to**: result.rows[0].email, *// list of receivers* **subject**: **'Ваше заявление # '** + arg2 + **' просмотрено и принято!'**, *// Subject line* **text**: **'Здравствуйте, '** + result.rows[0].name+**'. Ваше заявление №'** + arg2+**' просмотрено и принято! Перейдите на сайт за дополнительной информацией'**, *// plaintext body* };  
 transporter.sendMail(mailOptions, **function**(error, info){  
 **if**(error){  
 callback(500)  
 }  
 });  
 }  
})

4.6 Запросы SQL в данной работе

Выбор всех заявлений пользователя отсортированных по дате по убыванию:

client.query(**"SELECT text,viewed,accepted,date FROM statements WHERE user\_id=$1 ORDER BY date DESC"**,[req.**session**.**userId**],callback)

Обновление данных о пользователе:

client.query(**"UPDATE users SET surname=$1,name=$2,patronymic=$3,birth=$4,passport=$5,address=$6,tel=$7,pass=$8,email=$9 WHERE id=$10"**,  
 [body.**surname**,body.name,body.**patronymic**,body.**birth**,body.**passport**,body.**address**,body.**tel**,body.**pass**,body.email,req.**session**.**userId**],callback);

Удаление работника отделения:

client.query(**"DELETE FROM employees WHERE id=$1"**,[req.body.id],callback);

Добавление заявления:

client.query(**"INSERT INTO statements (text,user\_id,date) VALUES($1,$2,$3)"**,[req.body.**text**,req.**session**.**userId**,**new** Date()],callback);

Изменение заявления:

client.query(**"SELECT id FROM statements WHERE user\_id=$1 ORDER BY date DESC"**,[req.**session**.**userId**],callback);

client.query(**"UPDATE statements SET accepted=false,emp\_id=null,viewed=false,text=$1,date=$2 WHERE id=$3"**,  
[req.body.**text**,**new** Date().toLocaleString(),arg[req.body.id][**'id'**]],callback);

Получение статистики заявлений пользователей:

client.query(**`SELECT   
users.id,users.surname,users.name,users.patronymic,users.birth,users.passport,users.address,users.tel,  
users.email,COUNT(statements.id)  
FROM users,statements   
WHERE statements.user\_id=users.id   
GROUP BY users.id  
ORDER BY users.id`**,callback);

ВЫВОДЫ

В данной курсовой работе была разработана информационная система «Отделение милиции» для упрощения и оптимизации этой области.

Работа над информационной системой требовала детального разбора будущей базы данных, в ходе которого были изучены различные этапы оптимизации базы данных и проекта в целом. Среди них - ER-диаграмма, нормализация, концептуальный анализ и т. д.

В результате написания курсового проекта был разработан полноценный программный продукт на платформе Node.js [4], в среде разработки WebStorm 2016 средствами языка JavaScript. Программа была разработана в полном соответствии с составленным планом. Также были изучении отдельные фазы разработки программного обеспечения и компоновки их в единое целое. Главным является то, что я не только закрепил знания по дисциплине, а также получил опыт в разработке программного продукта, изучил стандарты оформления документации [5].

В дальнейшем возможно расширение функционала программы.

СПИСОК ЛИТЕРАТУРЫ

1.Ульман Дж.Д. Основы систем баз данных. - М.: Финансы и статистика, 1983, - 334 с.

2.Теория и практика построения баз данных. 8-е изд. / Д.Кренке. – СПб.: Питер, 2003. – 800 с.: ил.

3. Теория и практика построения баз данных. 8-е изд. [Текст] / Д.Кренке.

– СПб.: Питер, 2003. – 800 с.: ил.

# 4. Node.js. Docs | Node.js. Node.js v7.3.0 Documentation [Электронный ресурc] . – Режим доступа: www/URL: https://nodejs.org/dist/latest-v7.x/docs/api/– 25.05.2016 г. – Загл. с экрана.

5. Документація. Звіти у сфері науки і техніки : Структура і правила

оформлення: ДСТУ 3008-95. – [Чинний від 1996–01–01]. – К. : Держстандарт України, 1995. – 37 с.