Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра интеллектуальных информационных технологий

**Отчет по лабораторной работе №2**

**По курсу “Проектирование баз знанмй”**

Выполнил студент группы 821703: Коктышев И.Д.

Проверил: Синельников П.М.

**МИНСК**

**2020**

**Вариант 11.** «Учет приказов и распоряжений»

Предприятие, имеющее в своей структуре ряд иерархически подчиняющихся подразделений.

Словесное описание предметной области: Необходимо хранить информацию о внутренних приказах, распоряжениях или других подобных документах, сроках их выполнения и исполнителях. Например: название документа - "Указание о введении в действие плана по совершенствованию охраны труда", мероприятие, описываемое в документе – "Установить порядок выпуска приказов по случаю нарушения охраны труда", - исходящий корреспондент (человек, готовивший приказ) – гл. инженер.

Информация о исходящем корреспонденте: название подразделения, должность корреспондента, ФИО. Приказ, распоряжение характеризуются номером, датой, содержанием, мероприятием, ответственным за выполнение (название подразделения, должность, ФИО), датой выполнения мероприятия, пометкой о выполнении мероприятия. В одном документе может быть перечислено несколько мероприятий. Фиксируется дата выхода документа. Распоряжение подписывает сотрудник ранга руководителя предприятия, необходимо фиксировать должность и ФИО.

Необходимо реализовать выполнения следующих функций:

- Добавление/редактирование/удаление информации о внутренних приказах, распоряжениях или других подобных документах.

- Добавление/редактирование/удаление информации о корреспондентам.

- Просмотр списка исходящих корреспондентов, мероприятия по которым на заданную дату не были выполнены – текущая дата, дата выхода документа, название мероприятия, дата выполнения, ФИО, должность, название подразделения ответственного.

- Просмотр списка мероприятий за заданный период – дата начала периода, дата окончания периода, название мероприятия, выполнено или нет;

- Просмотр списка всех зафиксированных документов заданного типа (например, приказов) на текущую дату, отсортированных по дате документа – название предприятия, текущая дата, название вида документа, список документов – номер, дата документа, пометка о выполнении.

**Реализация:**

Программа была реализована на языке Ruby и СУБД MySQL.

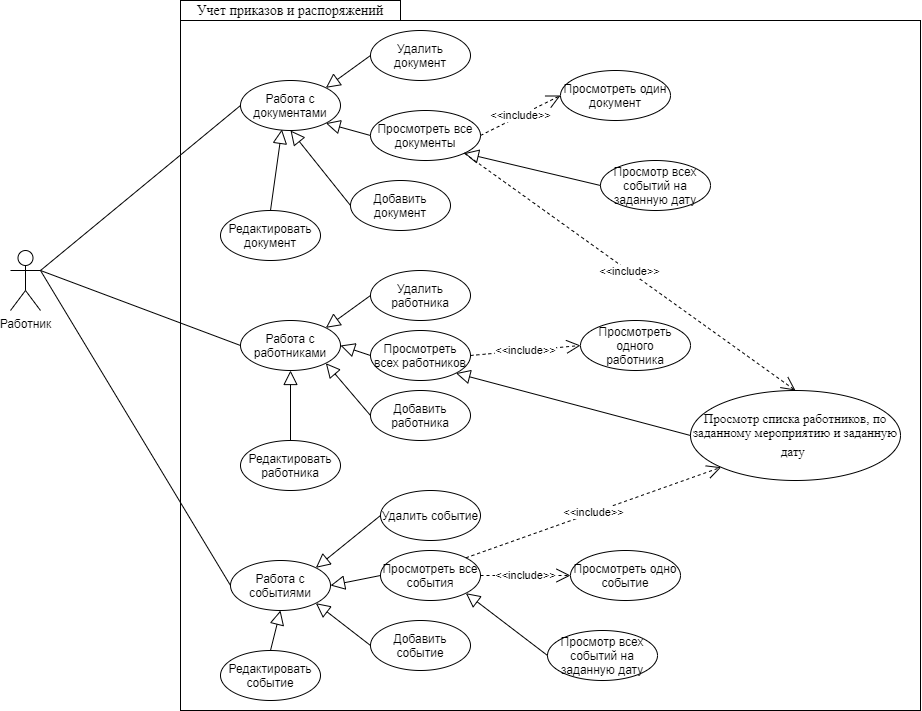


Рис.1 Диаграмма вариантов использования

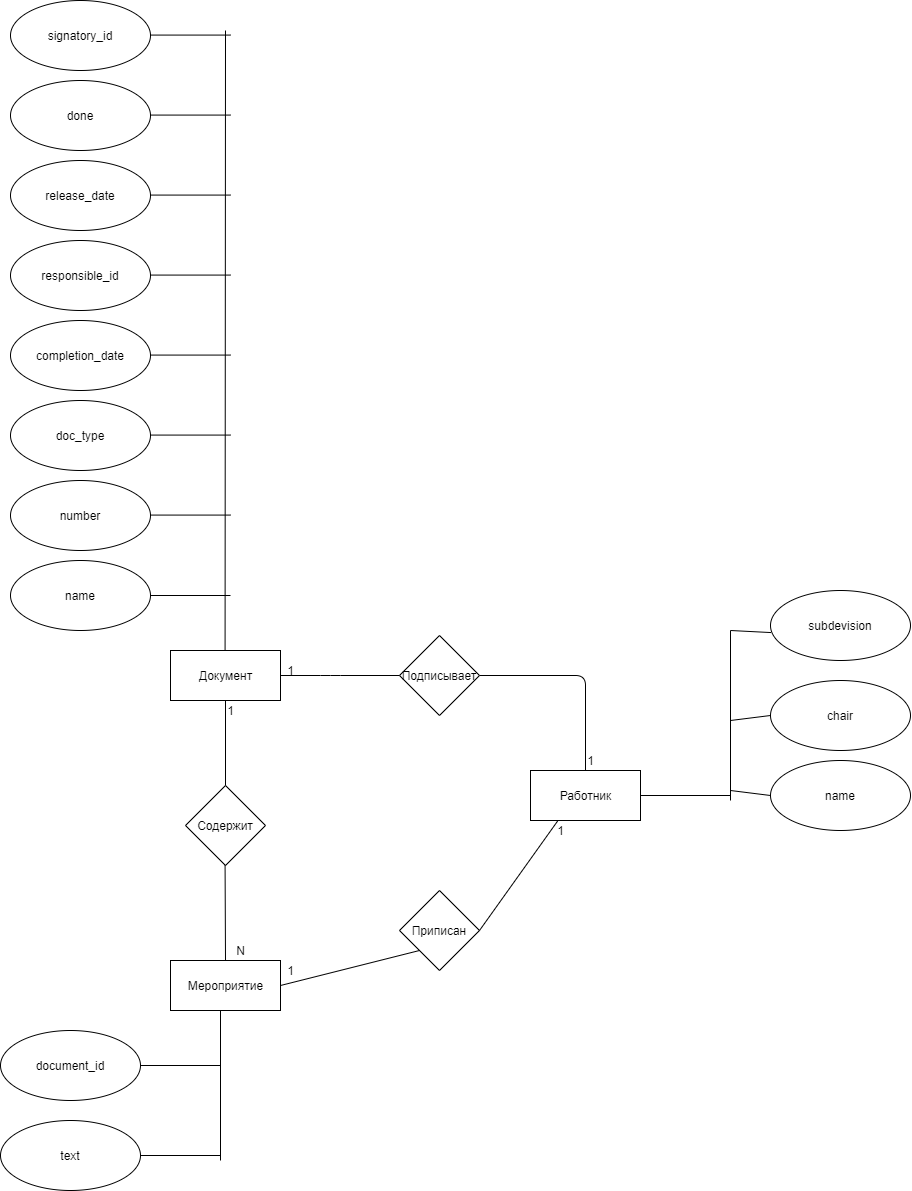


Рис.2 Диаграмма сущность-связь

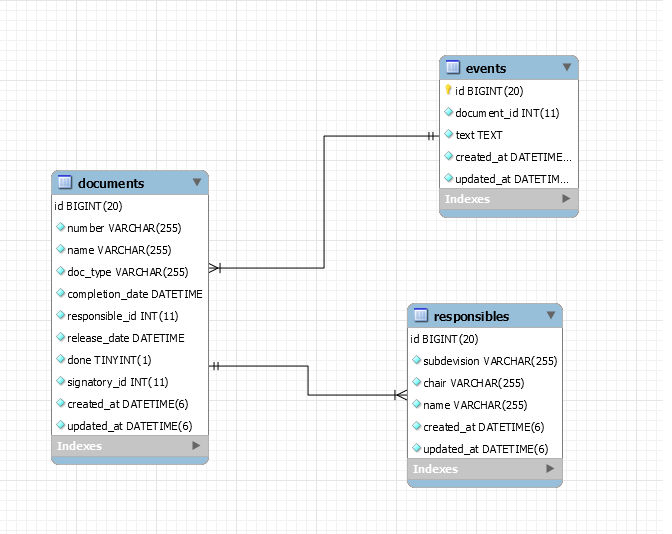


Рис.3 Схема БД

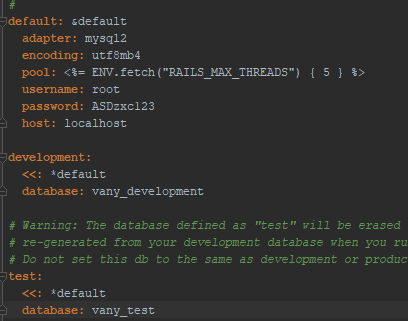
.

Рис.4 Конфигурация базы данных

Код указанный выше содержит в себе данные для входа в БД. Он создаётся при создании нового проекта и в нем указывается данные для входа в базу данных.

При установлении соединения с сервером пользователю предоставляется следующий графический интерфейс:

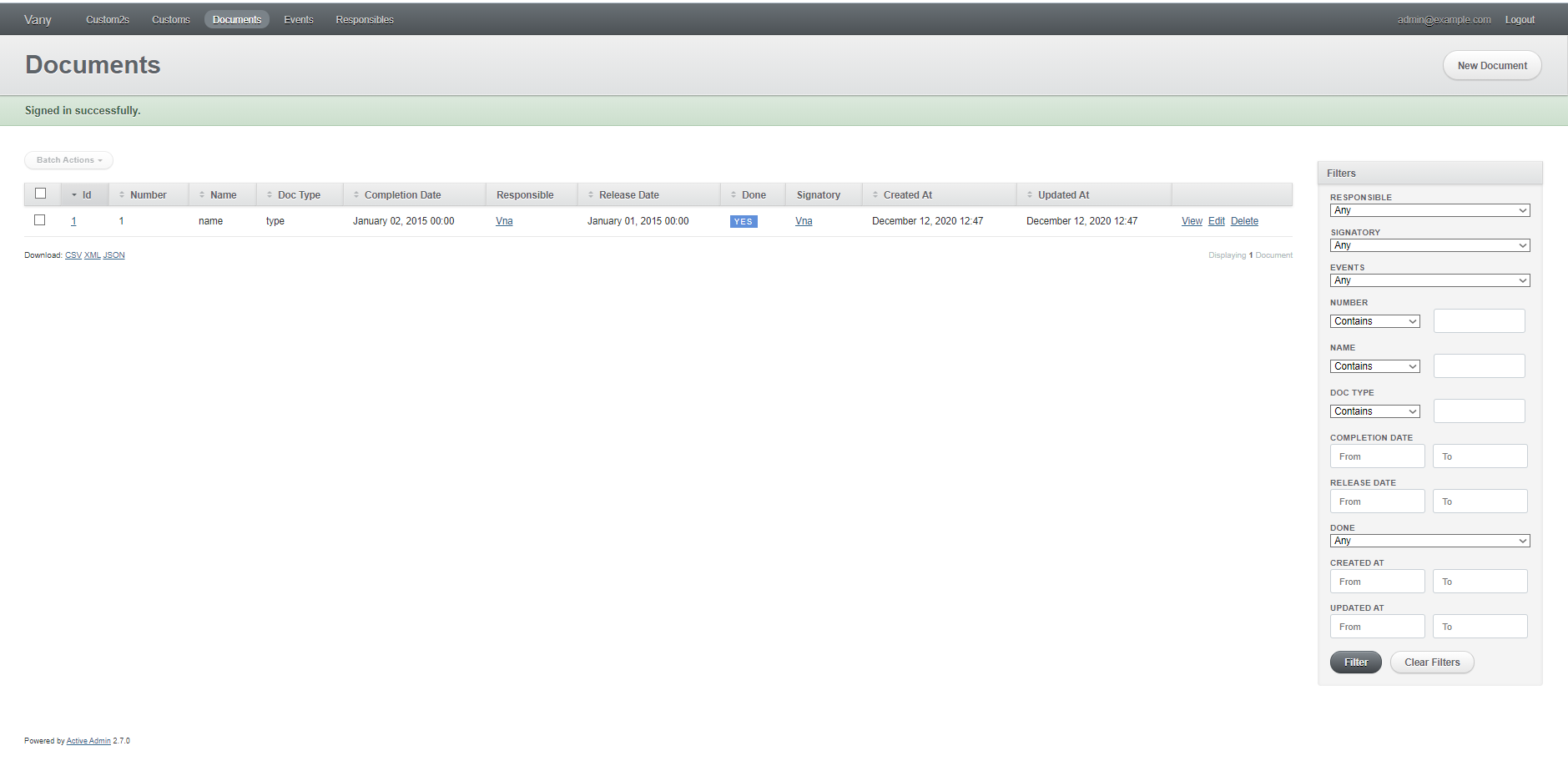


Рис.5 Графический интерфейс программы

Кнопки в header’е отвечают за переход между таблицами. Кнопки справа от таблицы отвечают за фильтрацию выходных данных. Кнопки Customs и Customs2s отвечают за выполнение 1 и 3 дополнительного задания соответственно.

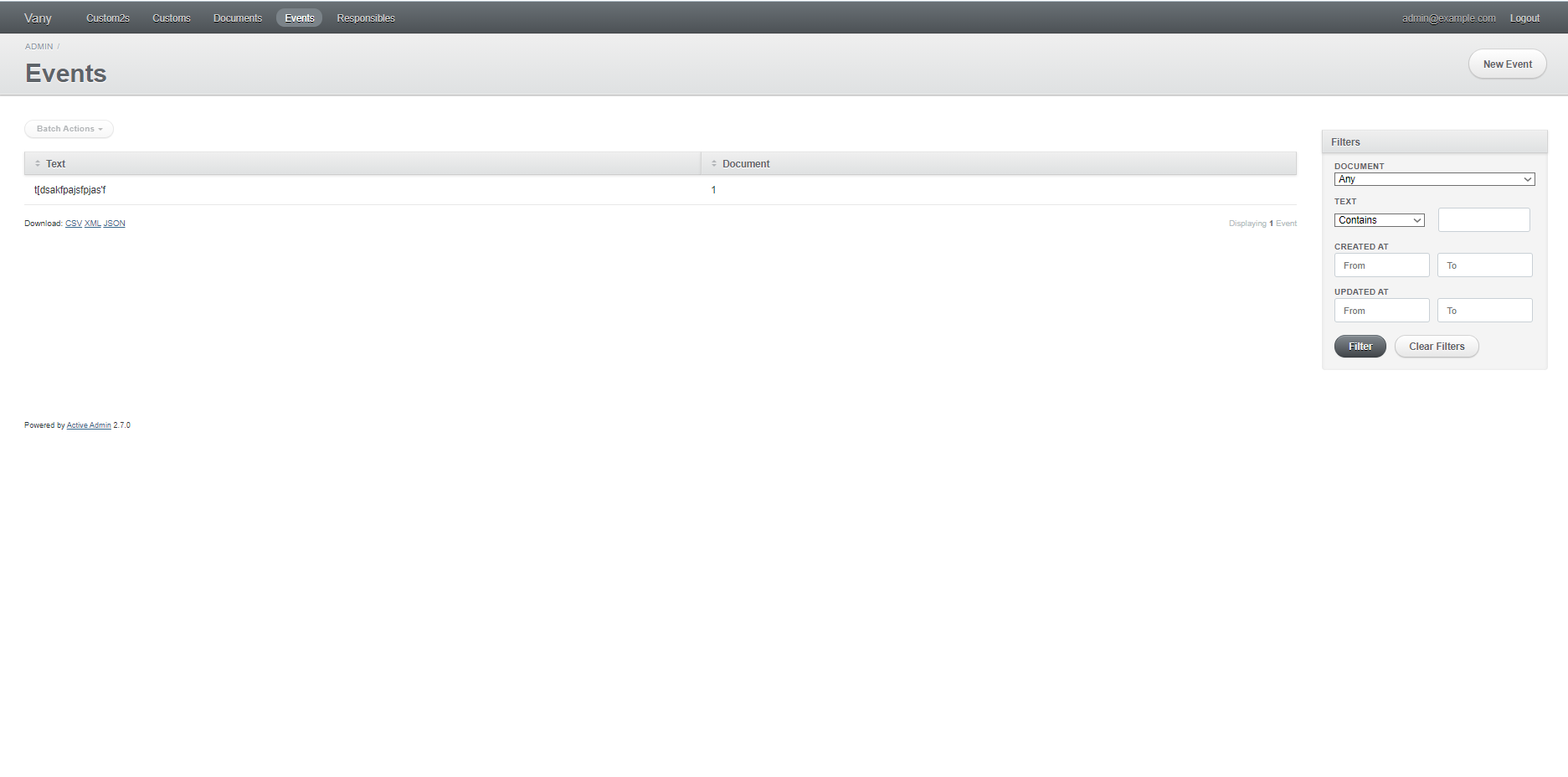


Рис.6 Графический интерфейс программы

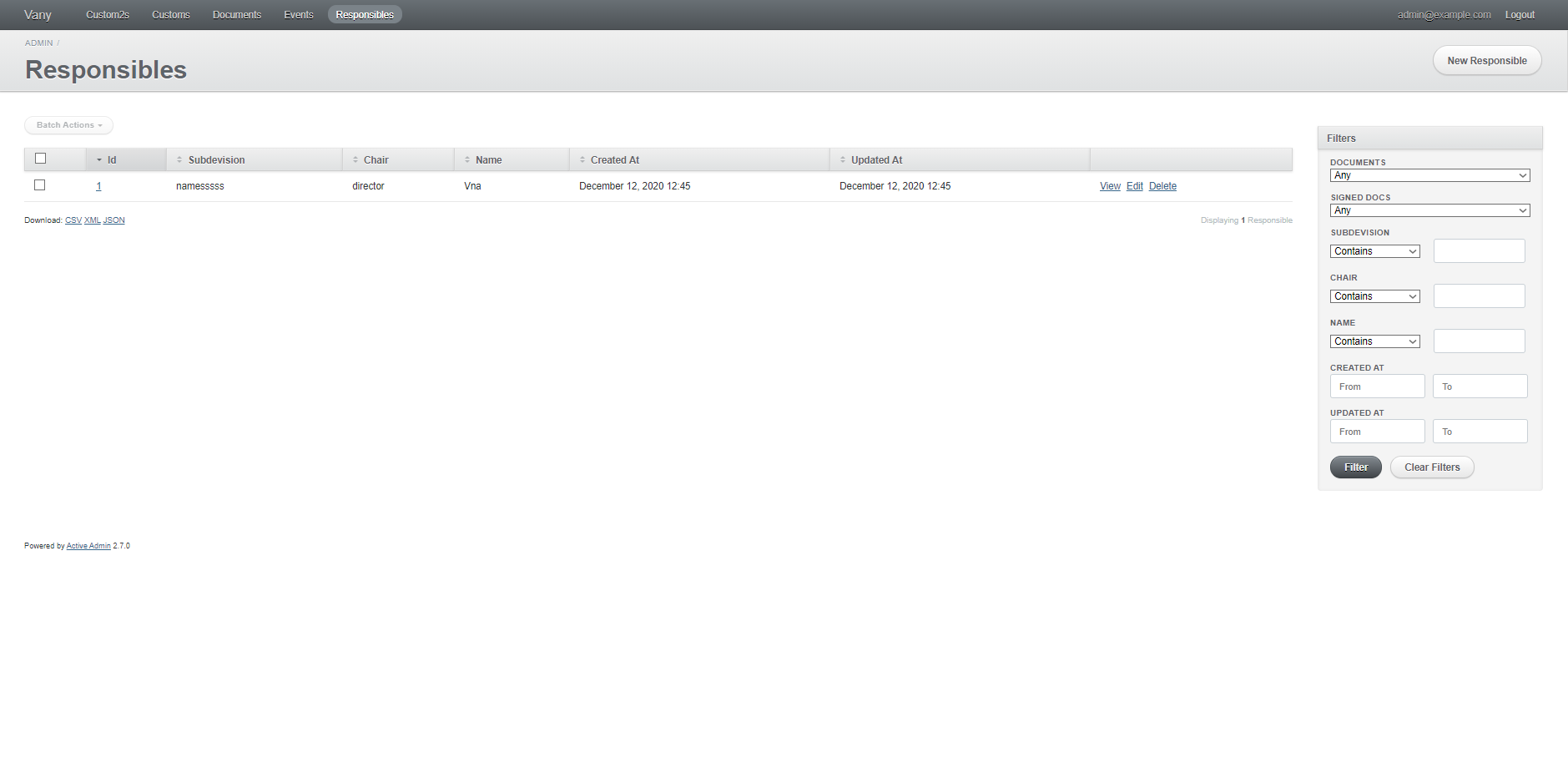


Рис.7 Графический интерфейс программы

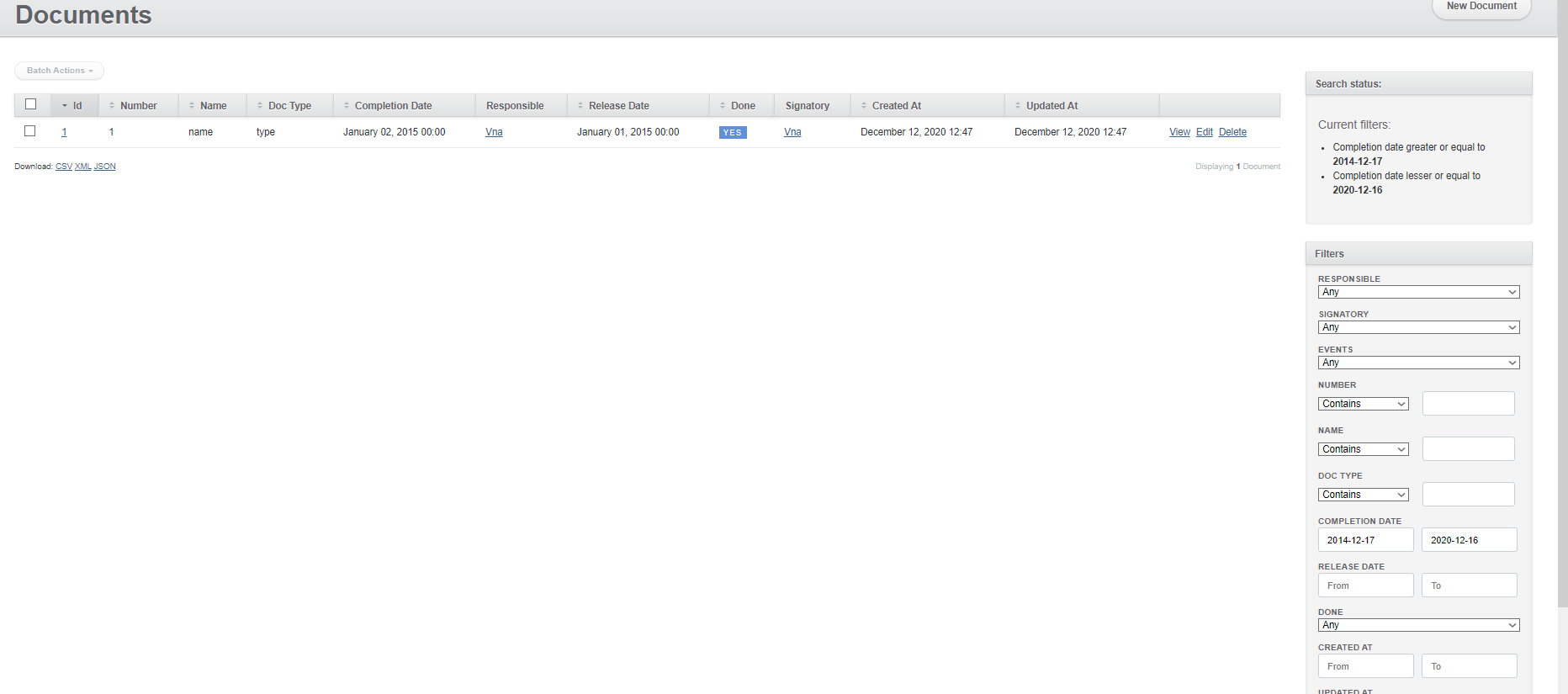


Рис.8 Графический интерфейс программы. Фильтрация (второе задание)

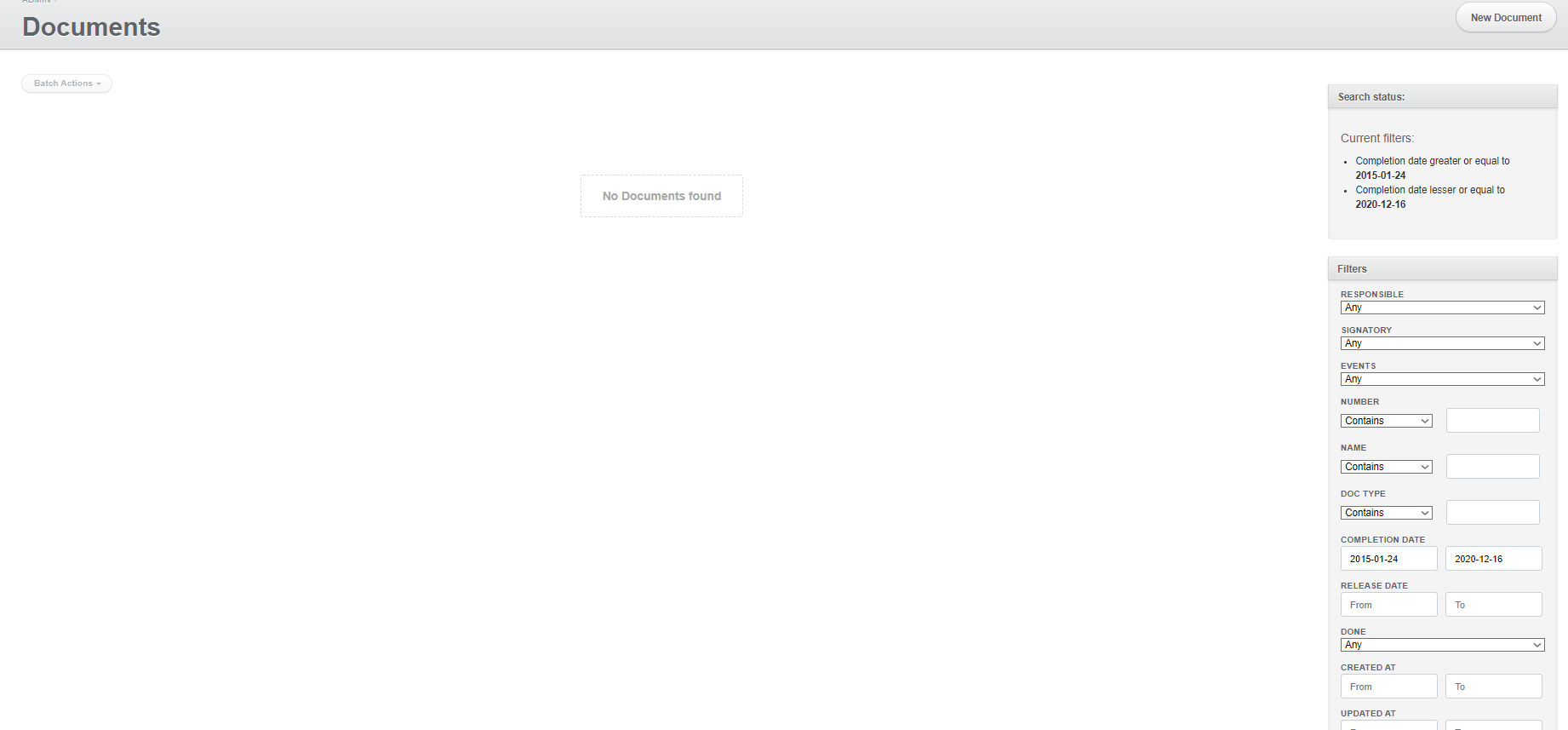


Рис.9 Графический интерфейс программы. Фильтрация (второе задание)

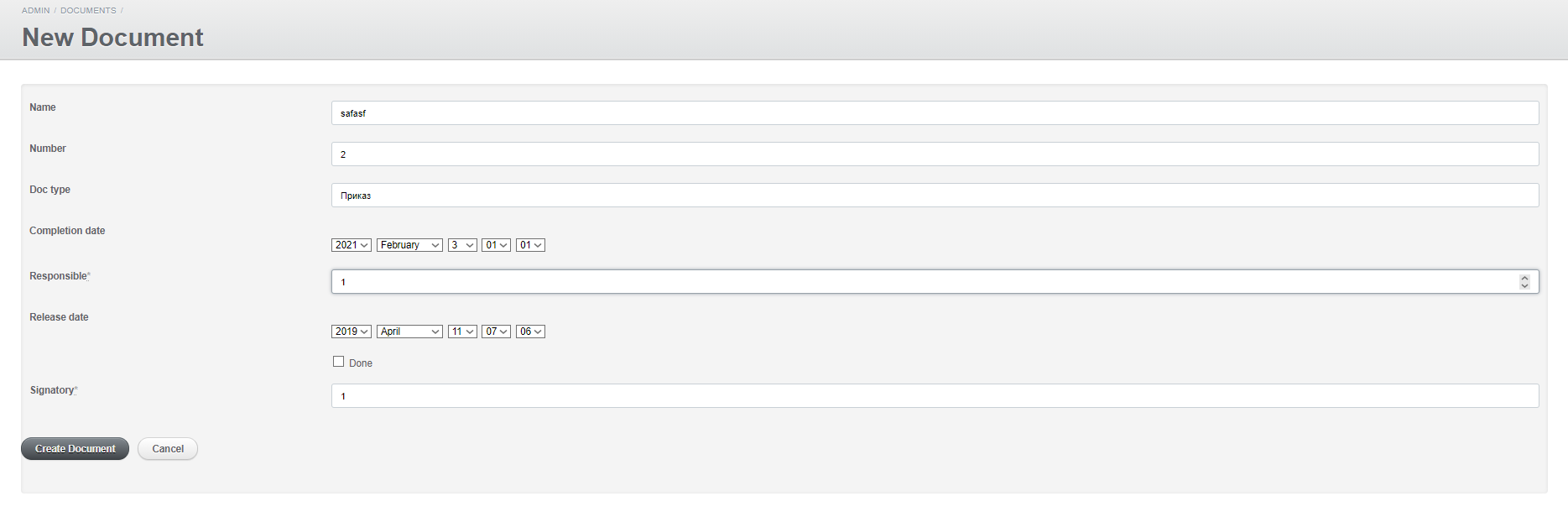


Рис.10 Графический интерфейс программы. Создание нового документа

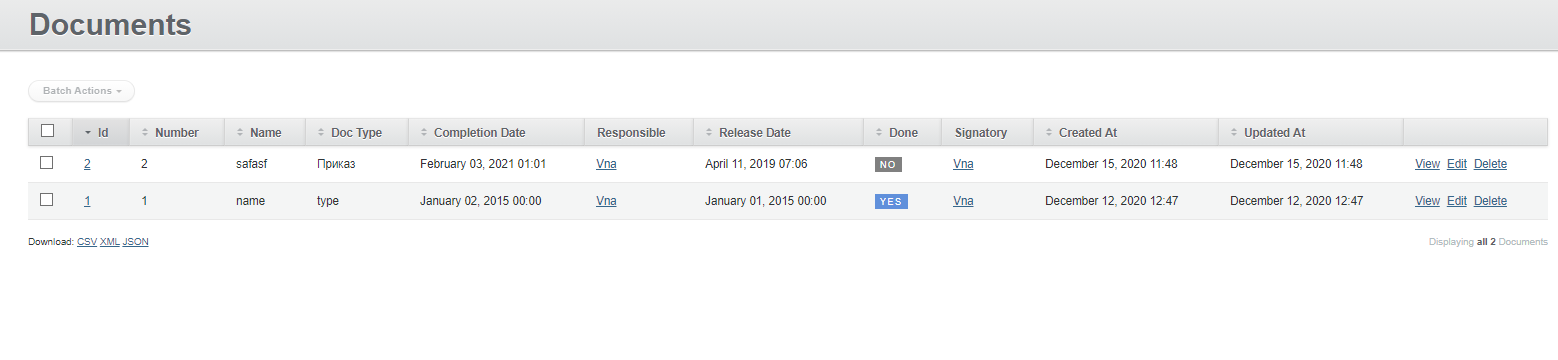


Рис.11 Графический интерфейс программы. Создан новый документ

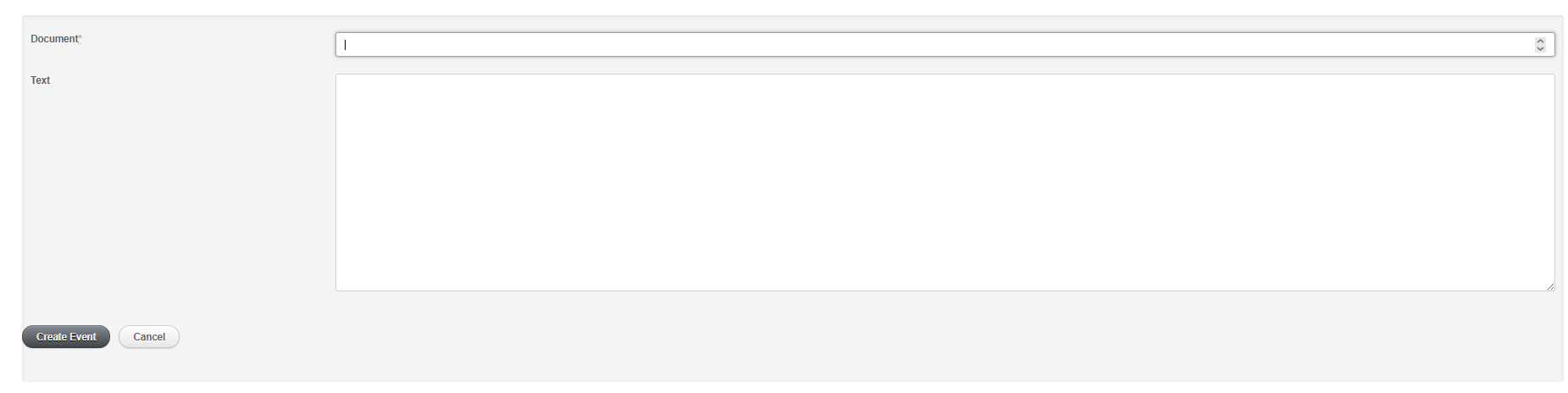


Рис.12 Графический интерфейс программы. Создание нового события

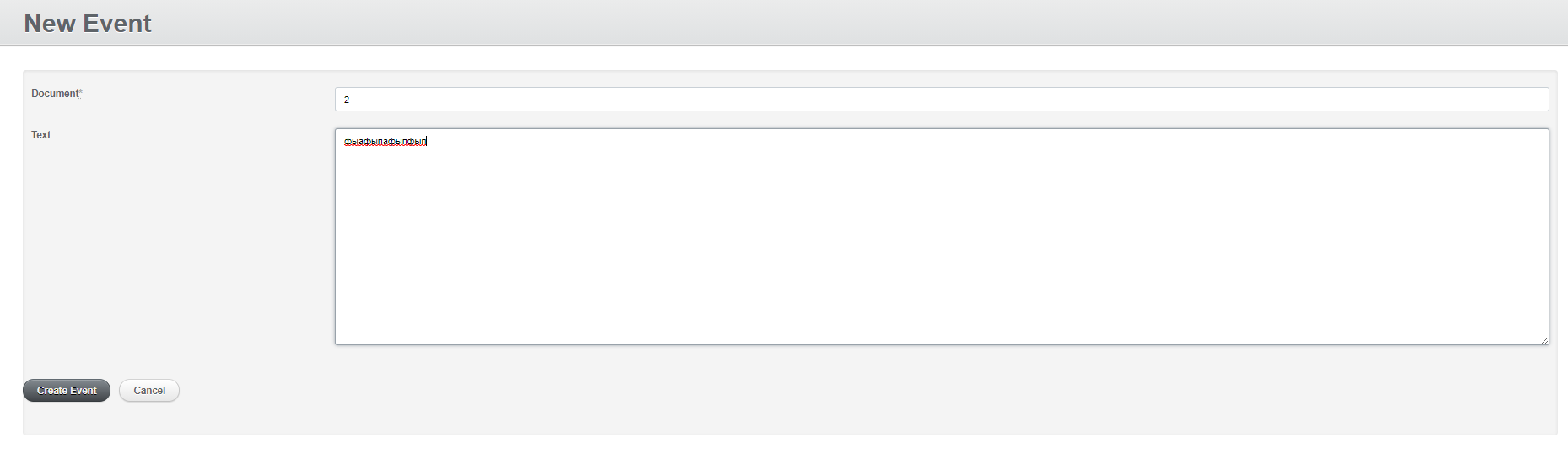


Рис.13 Графический интерфейс программы. Создание нового события



Рис.14 Графический интерфейс программы. Новое событие

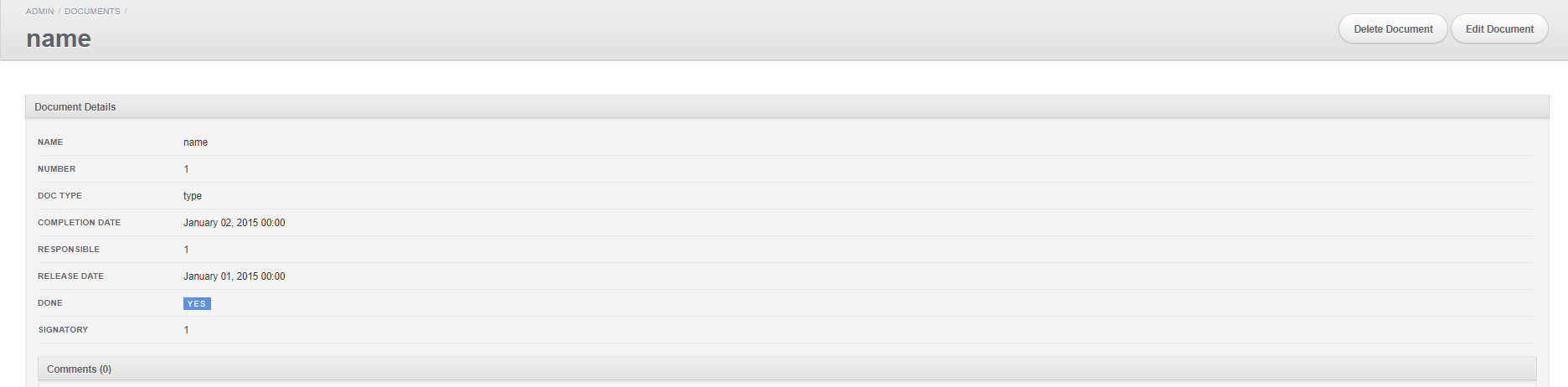


Рис.15 Графический интерфейс программы. Информация о документе



Рис.16 Графический интерфейс программы. Редактирование документа

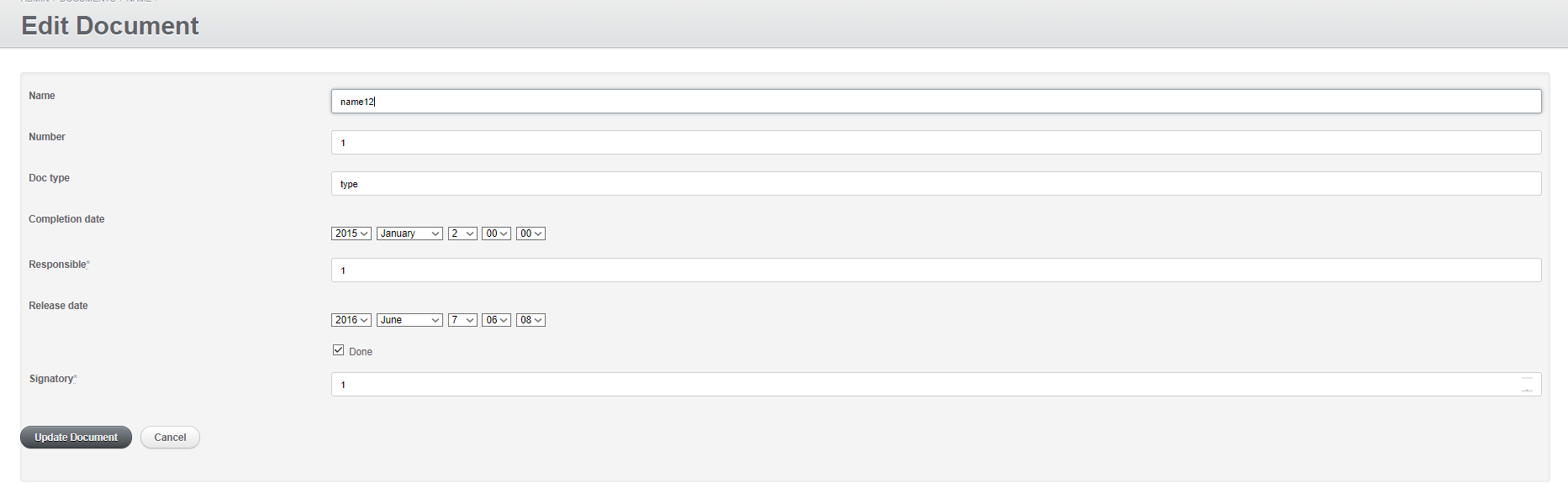


Рис.17 Графический интерфейс программы. Редактирование документа

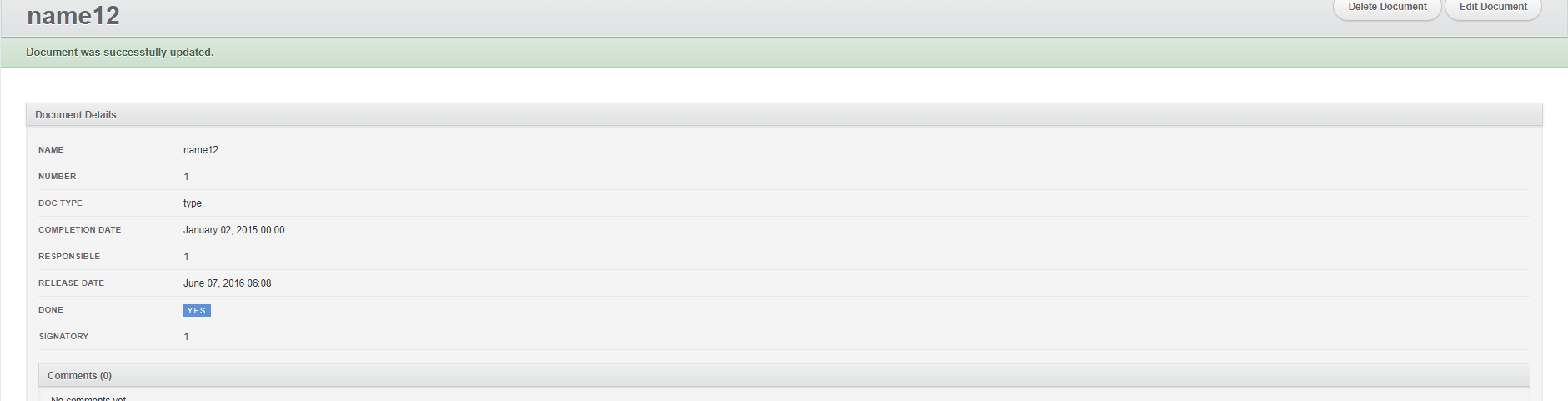


Рис.18 Графический интерфейс программы. Редактирование документа

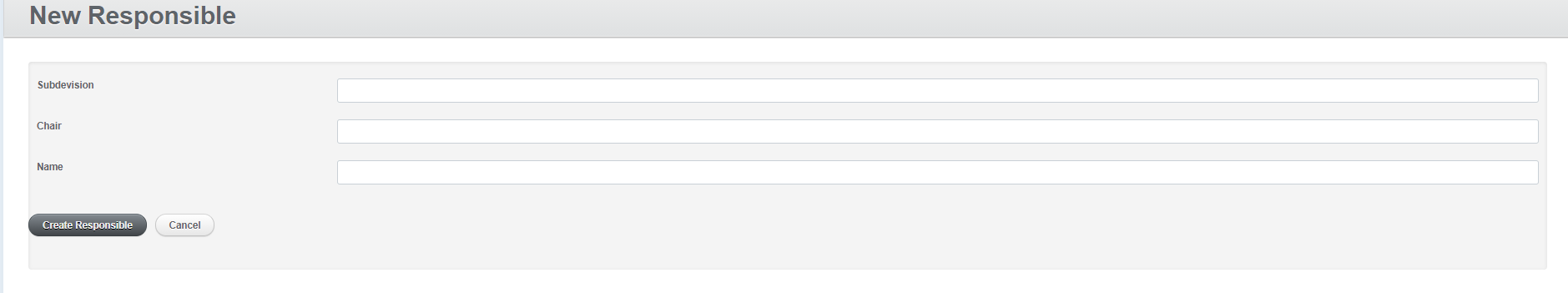


Рис.19 Графический интерфейс программы. Создание нового сотрудника

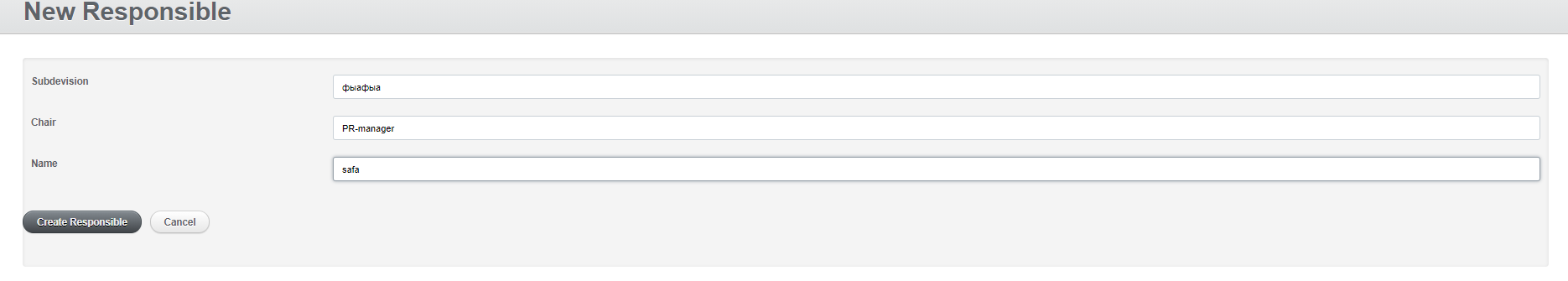


Рис.20 Графический интерфейс программы. Создание нового сотрудника

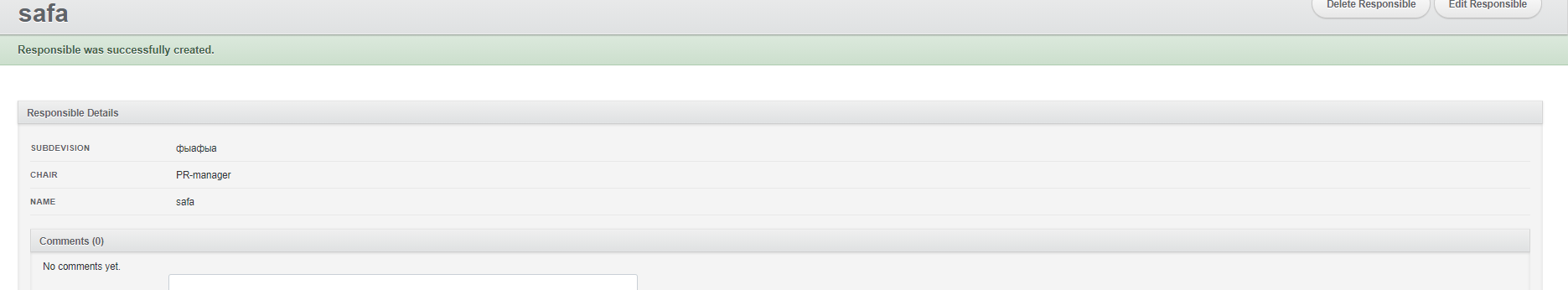


Рис.21 Графический интерфейс программы. Создание нового сотрудника

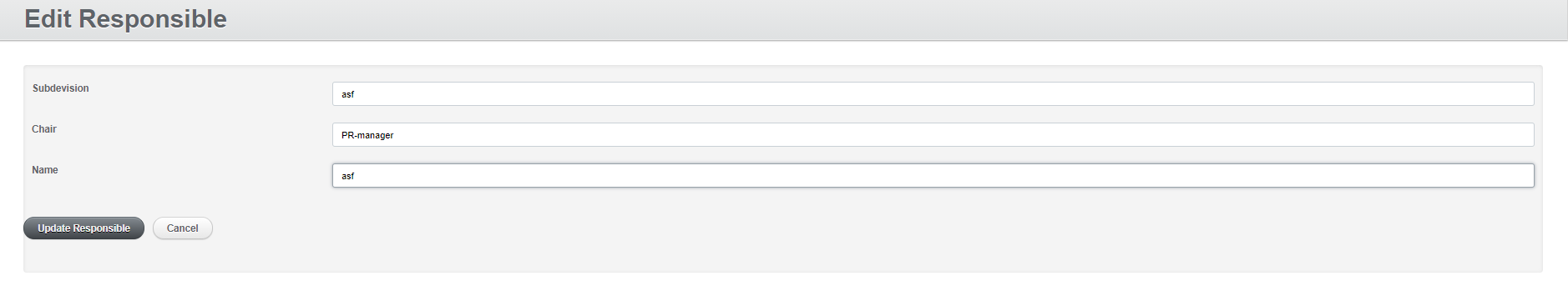


Рис.22 Графический интерфейс программы. Редактирование сотрудника

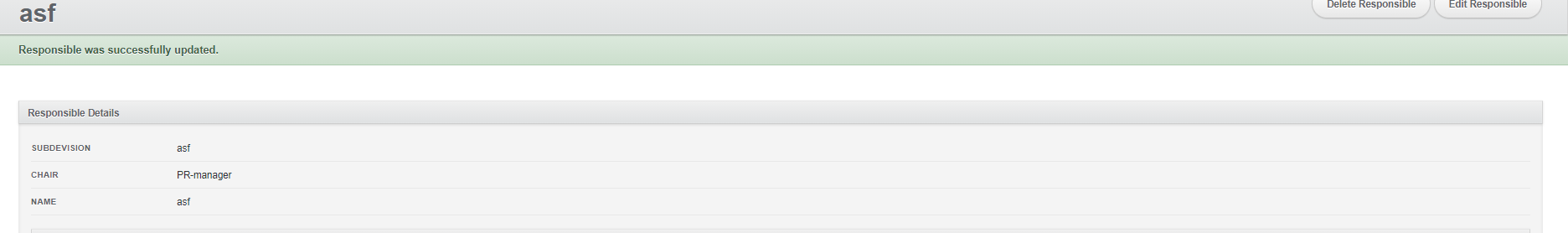


Рис.23 Графический интерфейс программы. Редактирование сотрудника

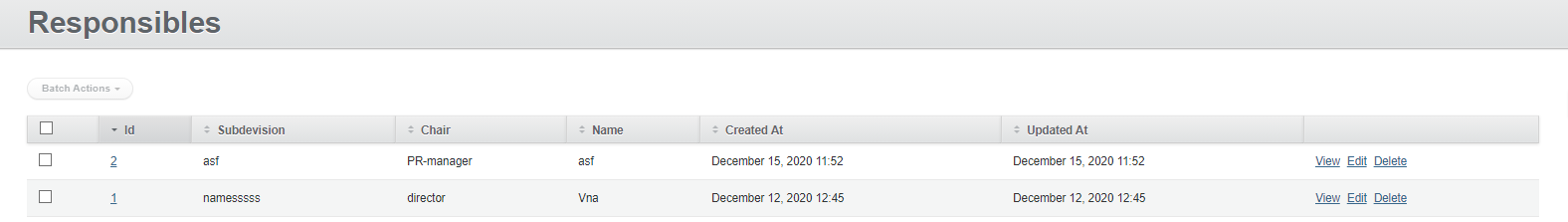


Рис.24 Графический интерфейс программы. Редактирование сотрудника

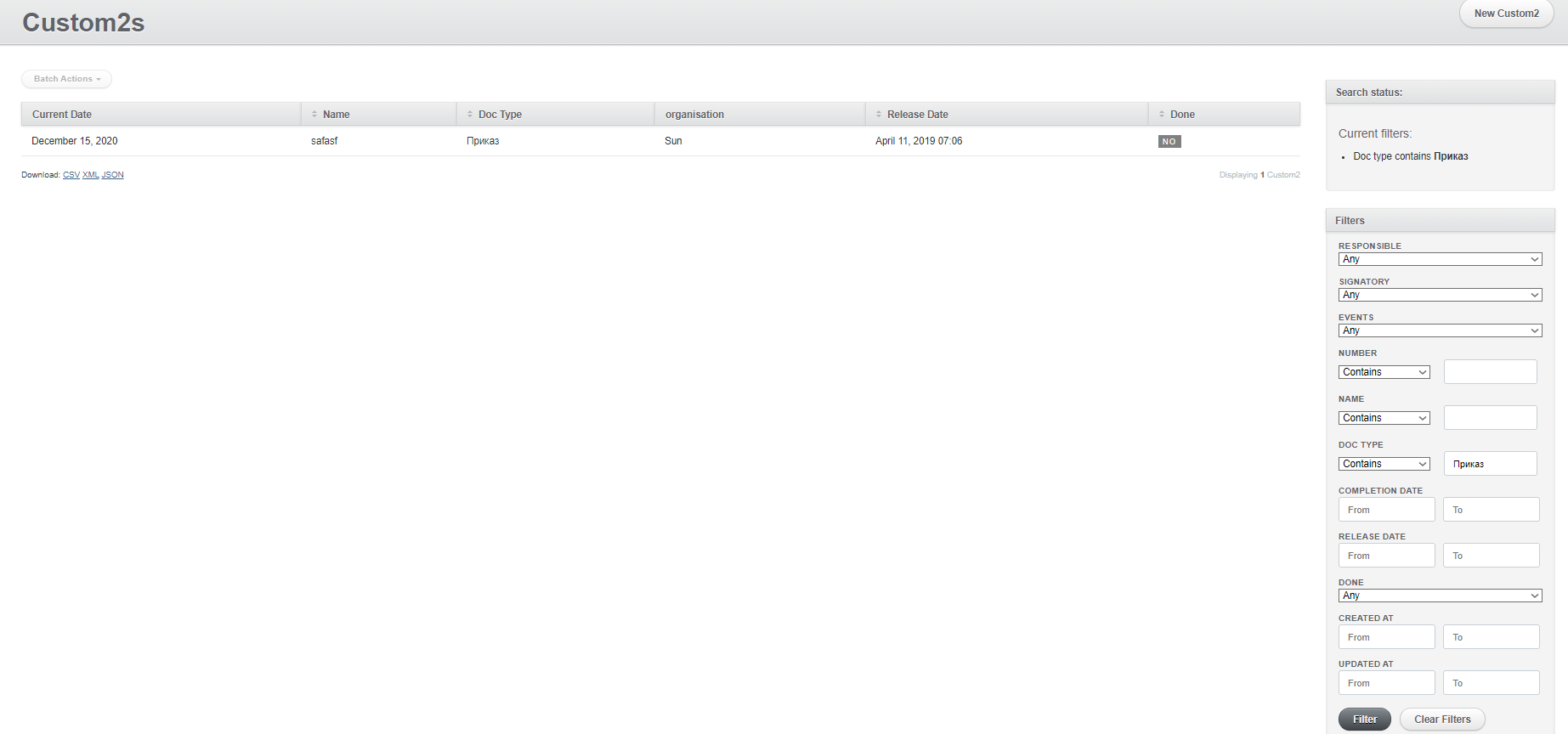


Рис.25 Графический интерфейс программы. Третье задание

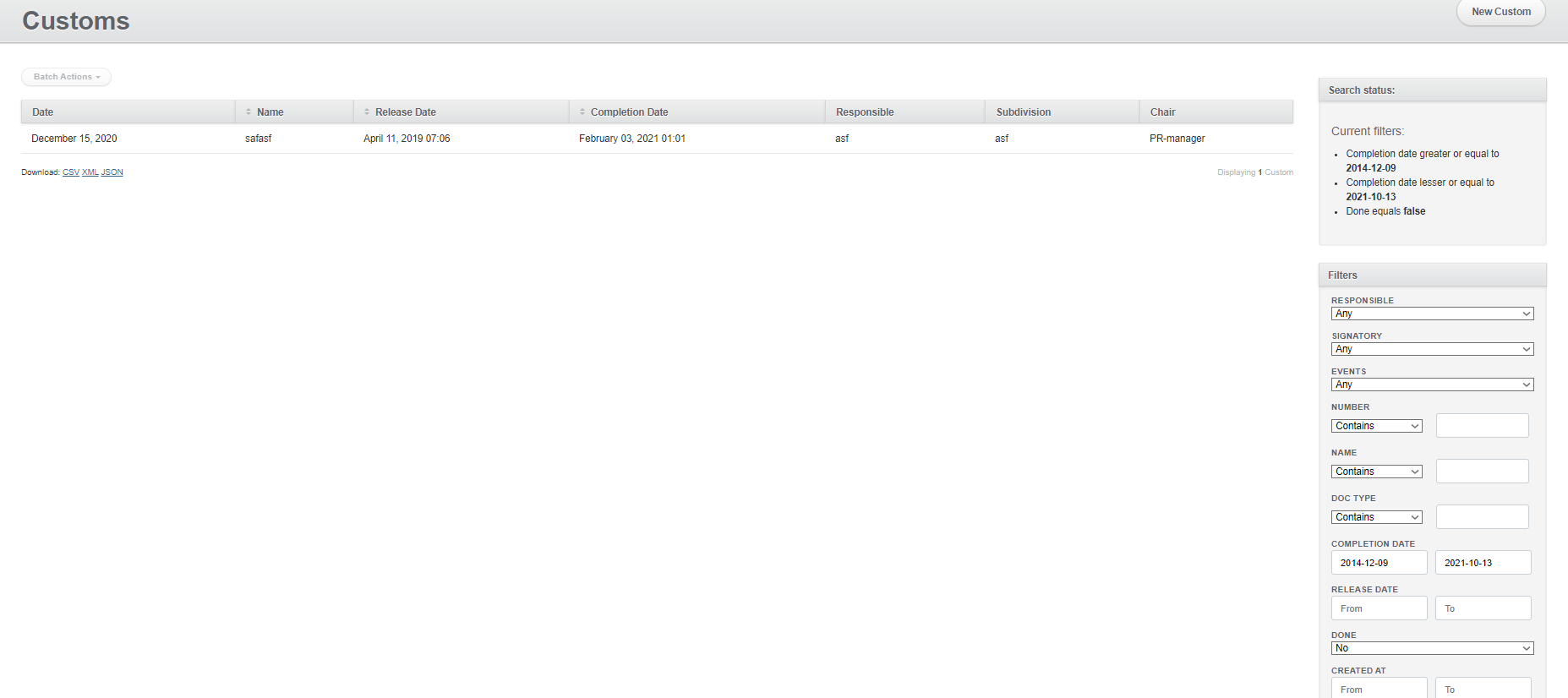


Рис.26 Графический интерфейс программы. Первое задание

За отображение самой БД на странице отвечают файлы, созданные (позже дополненные) при создании программы. Они имеют следующий код:

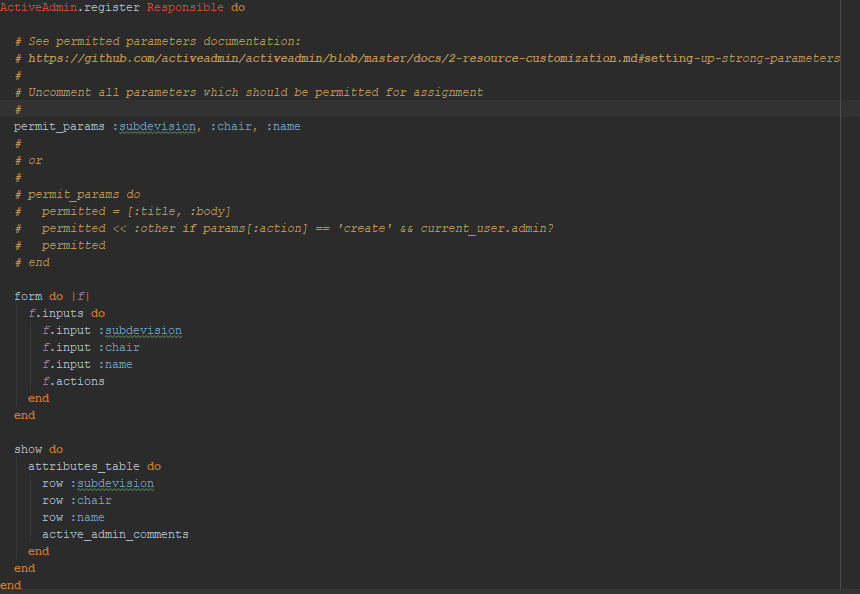


Рис.27 Код. Работники

Здесь permit\_params – это данные на странице, которые должны изменяться, колонка form отвечает за форму для редактирования и колонка show отвечает за отображение содержимого базы данных.

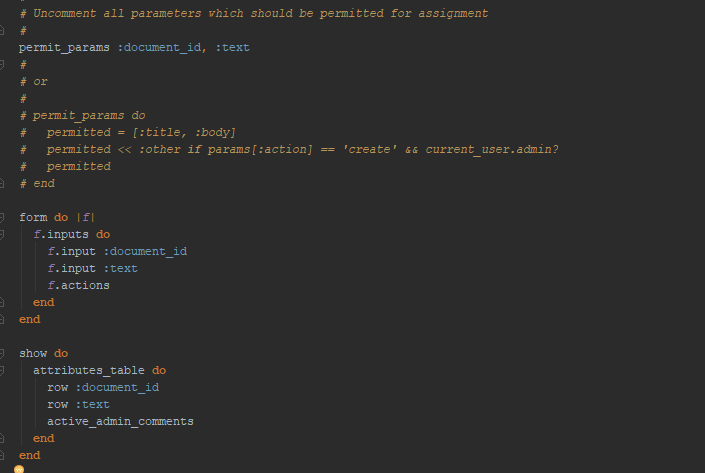


Рис.27 Код. События

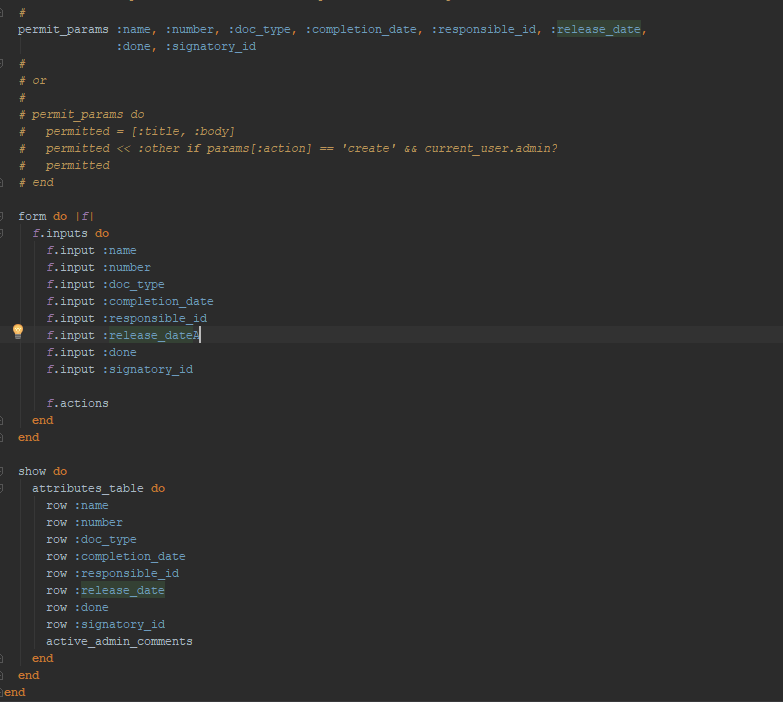


Рис.28 Код. Документы

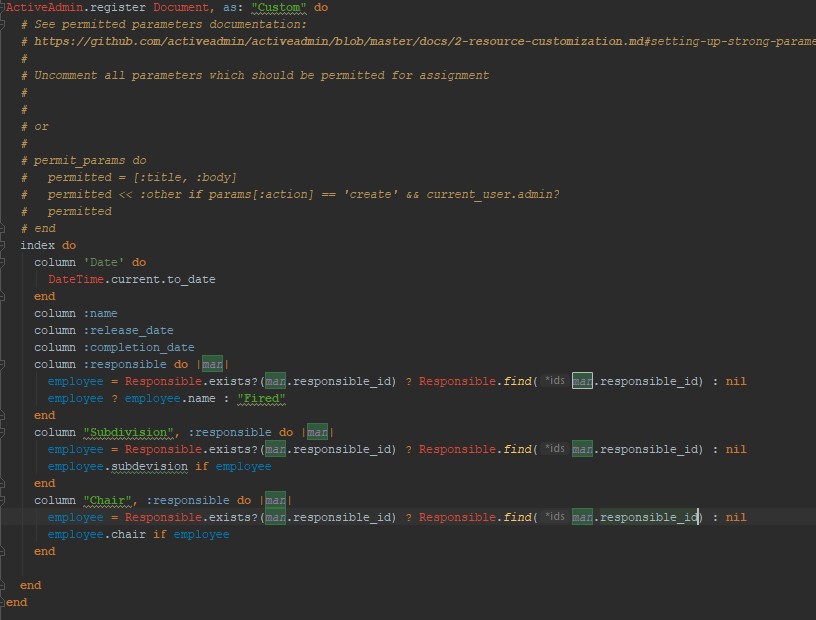


Рис.29 Код. Custom

Здесь происходит проверка на соответствие id в Responsibles и responsible\_id в Documents. Если он существует, то мы записываем его в таблицу, если нет, то мы записываем, что работник был уволен.

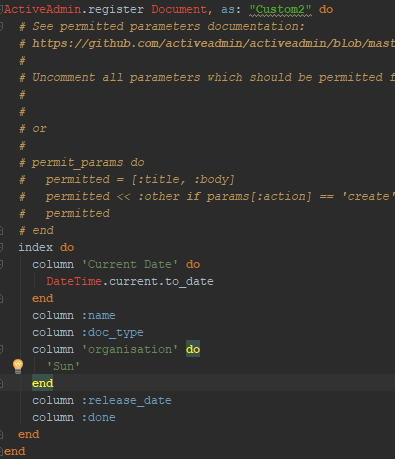


Рис.30 Код. Custom2s

Сами файлы, которые делают запрос к базе данных и передают их создаются автоматически, при создании проекта. Также сами создаются файлы, необходимые для отображения модели базы данных. Чтобы их сгенерировать в консоли приложения необходимо было прописать rails generate scaffold и название модели, для каждой из которой, в дальнейшем, сгенерируется 8 views файлов, 1 модель, 1 миграция и 1 контроллер. В файлах миграции, для соответствующих таблиц мы прописываем их поля.

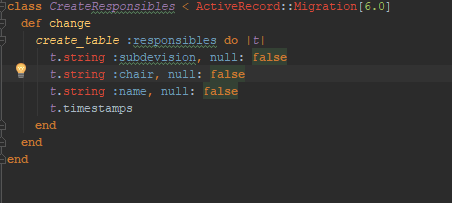


Рис.31 Код. Файл миграции для Responsibles

Для создания интерфейса был применен гем activeadmin.

После этого мы прописываем rake db:migrate, что применяет миграцию. Для того, чтобы сгенерировать файлы, в которых, в дальнейшем, мы будем прописывать столбцы для отображения их на форме, мы в консоли прописываем rails generate active\_admin:resource и потом название модели, после чего создаются файлы в которых остаётся прописать строки и столбцы для их отображения