Министерство образования московской области

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

московской области

«ВОСКРЕСЕНСКИЙ КОЛЛЕДЖ»

ОТЧЕТ

ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

ПМ.02 «ОСУЩЕСТВЛЕНИЕ ИНТЕГРАЦИИ ПРОГРАММНЫХ МОДУЛЕЙ»

09.02.07 «Информационные системы и программирование»

студент\_\_ДП\_\_ группы\_\_4\_\_

Шкуренков Дмитрий Сергеевич

Ф.И.О.

Руководитель практики

от предприятия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_/

Руководитель практики

от ГБПОУ МО ВК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_/

Воскресенск

2021

ОГЛАВЛЕНИЕ

[Введение 3](#_Toc101114241)

[1. Моделирование предметной области 4](#_Toc101114242)

[1.1 Построение UML-диаграмм классов и прецедентов 4](#_Toc101114243)

[1.2 Построение диаграмм IDEF0 5](#_Toc101114244)

[1.3 Построение диаграмм IDEF1X 6](#_Toc101114245)

[1.4 Построение диаграмм IDEF3 7](#_Toc101114246)

[1.5 Создание схемы данных 8](#_Toc101114247)

[2. Основная часть 10](#_Toc101114248)

[2.1. Тестирование и отладка 10](#_Toc101114249)

[2.1.1. Составление плана тестирования 10](#_Toc101114250)

[2.1.2. Составление тестовых сценариев 11](#_Toc101114251)

[2.1.3 Осуществление тестирования программного продукта и оформление тестовой документации 15](#_Toc101114252)

[2.2. Построение схемы итоговых приемо-сдаточных испытаний 19](#_Toc101114253)

[2.3 Работа с системой контроля версий 21](#_Toc101114254)

[2.4 Создание установочных файлов программного продукта 25](#_Toc101114255)

[2.5 Проведение испытаний и мероприятий по интеграции программного продукта в информационную систему организации 28](#_Toc101114256)

[Заключение 30](#_Toc101114257)

[Список литературы 31](#_Toc101114258)

[Приложение А. 32](#_Toc101114259)

[Приложение Б. 37](#_Toc101114260)

[Приложение В 40](#_Toc101114261)

# Введение

Цель производственной практики:

приобретение первоначального практического опыта для последующего освоения общих и профессиональных компетенций по виду профессиональной деятельности ПМ.02. Осуществление интеграции программных модулей.

Задачи производственной практики:

повышение качества профессиональной подготовки; закрепление приобретённых теоретических знаний; сбор, анализ и обобщение материалов для подготовки материалов отчета по практике.

Производственная практика организуется и проводится в ОУ. Оценка по производственной практике выставляется по факту выполнения заданий под руководством преподавателя. Отчет по учебной практике оформляется в соответствии с требованиями, указанными в методических рекомендациях.

Прохождение практики повышает качество профессиональной подготовки, позволяет закрепить приобретаемые теоретические знания, способствует социально-психологической адаптации на местах будущей работы. Методические рекомендации предназначены для того, чтобы помочь студентам подготовиться к эффективной деятельности в качестве техника-программиста. Выполнение заданий практики поможет студентам быстрее адаптироваться к условиям работы в сфере проектирования и разработки программного обеспечения.

Объект изучения в ходе практики:

Предметная область данной практики является программа по переобучению безработных граждан.

Предмет исследования:

Требования к программному обеспечению, моделирование предметной области, составление технического задания, тестирование программных модулей, составление тест-кейсов.

Практическая значимость:

Приложение разрабатывается и тестируется для Воскресенского управления социальной защитой населения.

# Моделирование предметной области

## 1.1 Построение UML-диаграмм классов и прецедентов

UML прецедентов

Прецедент, также: вариант использования, сценарий использования — спецификация последовательностей действий (варианты последовательностей и ошибочные последовательности) в Унифицированном языке моделирования (UML), которые может осуществлять система, подсистема или класс, взаимодействуя с внешними действующими лицами (англ. Actors).

Данная диаграмма показывает связи между гражданами и сотрудниками. Гражданин оставляет заявку на переобучение. А сотрудник обрабатывает заявление и предоставляет подходящие курсы для желаемой работы гражданина.

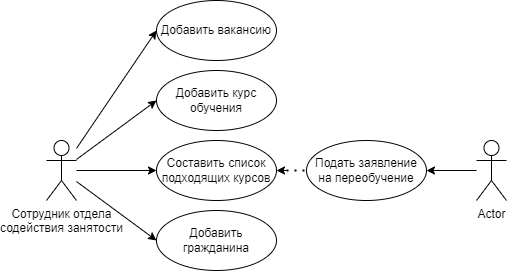


Рисунок 1 – Диаграмма UML прецедентов

UML классов

Диаграмма классов — структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

В данной диаграмме показана структура иерархии системы. Классы Tag vacancies, Tag Course, Desired job унаследованы от общего класса Tag Catalog. А классы Course of Study и Vacancies содержат в себе классы Tag Course, Tag vacancies соответственно

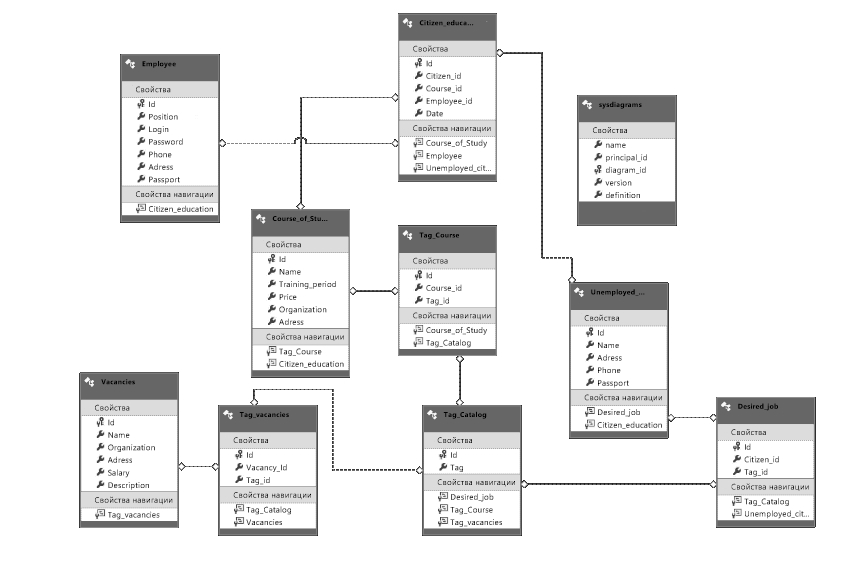


Рисунок 2 – Диаграмма UML классов

## 1.2 Построение диаграмм IDEF0

IDEF0 - нотация графического моделирования, используемая для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающих эти функции.

Диаграмма состоит из трех типов документов: графических диаграмм, текста, глоссария. Эти документы имеют перекрестные ссылки друг на друга.

Графическая диаграмма - главный компонент модели, содержащий блоки, стрелки, соединения блоков и стрелок. Диаграмма является единицей описания системы и располагается на отдельном листе.

Блоки представляют основные функции моделируемого объекта. Эти функции могут быть декомпозированы на составные части и представлены в виде более подробных диаграмм.

В данной предметной области диаграмма описывает функцию обработки заявления граждан.

Функцию обработки заявлений выполняет сотрудник, в ней есть входные данные такие как фамилия имя отчество гражданина его паспорт и список интересующих его вакансий так же список всех доступных курсов обучения, программа сопоставляет интересующие вакансии и курсы обучения с помощью каталога тегов, при этом сотрудник руководствуется инструкцией по общению с клиентом, на выходе функции выводится список подходящих курсов.

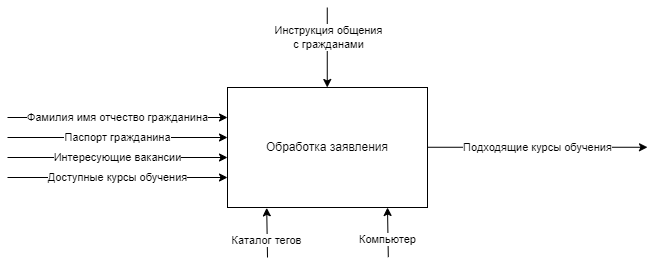


Рисунок 3 – диаграмма IDEF0

## 1.3 Построение диаграмм IDEF1X

Integration DEFinition for information modeling (IDEF1X) язык моделирования данных для разработки семантики моделей данных. IDEF1X используется для формирования графических представлений информационных моделей, которые отражают структуру и семантику информации внутри среды или системы.

IDEF1X позволяет строить семантические модели данных, которые могут служить для поддержки управления данными как ресурсом, интеграции информационных систем и построения компьютерных баз данных. Этот стандарт является частью семейства языков моделирования IDEF в области программной инженерии.

В данной диаграмме описываются связи между таблицами, Сотрудники, Курсы обучения, Теги курсов обучения, Безработные граждане, Желаемая работа, Вакансии, Теги вакансий, Каталог тегов, Переобучение граждан.



Рисунок 4 – диаграмма IDEF1Х

## 1.4 Построение диаграмм IDEF3

IDEF3 является стандартом документирования технологических процессов, происходящих на предприятии, и предоставляет инструментарий для наглядного исследования и моделирования их сценариев. Сценарием называется описание последовательности изменений свойств объекта, в рамках конкретного процесса.

В данной предметной области с помощью IDEF3 рассматривается процесс обработки заявления. Изначально сотрудник заносит в базу гражданина, после чего составляет интересующие его вакансии и начинается поиск подходящих курсов обучения, из этого списка выбирается один курс и на него направляется гражданин.

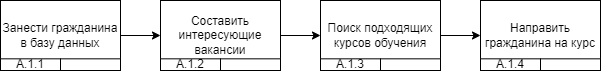


Рисунок 5 – диаграмм IDEF3

## 1.5 Создание схемы данных

Структура схемы базы данных для программного обеспечения Воскресенского отделения социальной защиты населения (рисунок 5). Схема базы данных представляет собой логическую конфигурацию либо целой реляционной базы данных, либо ее части. Схема может существовать как в виде наглядного представления базы данных, так и в виде набора формул (также именуемых «условиями целостности»), которые регулируют ее устройство. Эти формулы выражаются с помощью языка описания данных, например, SQL. Будучи частью словаря данных, схема показывает, как связаны между собой сущности, из которых состоит база данных (таблицы, представления, хранимые процедуры и так далее).

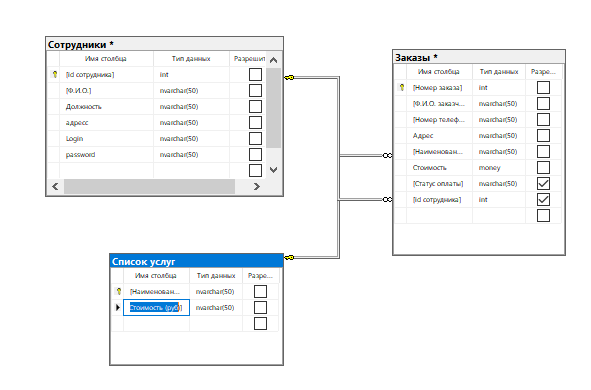


Рисунок 5 –Схема базы данных

Таким образом строится схема базы данных для программного обеспечения Воскресенского отделения социальной защиты населения

# 2. Основная часть

## 2.1. Тестирование и отладка

## 2.1.1. Составление плана тестирования

В план отладки входят следующие этапы:

* Сравнение программы со схемой алгоритма;
* Визуальный контроль программы на экране дисплея или визуальное изучение распечатки программы и сравнение ее с оригиналом на программном бланке;

Для документирования ошибок, происходящих во время тестирования, будут использоваться тест-кейсы. Шаблон для них следующий.

|  |  |
| --- | --- |
| Шапка | |
| Короткое описание |  |
| проект |  |
| компонент приложения |  |
| номер версии |  |
| Серьезность |  |
| Приоритет |  |
| Статус |  |
| Автор |  |
| Назначен на |  |
| окружение | |
| ОС |  |
| Описание | |
| Шаги воспроизведения |  |
| фактический результат |  |
| ожидаемый результат |  |
| Описание ошибки |  |
| Дополнение | |
| Ход решения |  |

* Трансляция программы на машинных язык. На этом этапе выполнятся синтаксические ошибки;
* Редактирование внешних связей и компоновка программы. На этапе редактирования внешних связей программных модулей программа-редактор внешних связей, или компоновщик задач, обнаруживает такие синтаксические ошибки, как несоответствие числа параметров в описании подпрограммы и обращении к ней, вызов несуществующей стандартной программы;
* Выполнение программы. После устранения обнаруженных транслятором и редактором внешних связей (компоновщиком задач) синтаксических ошибок переходят к следующему этапу – выполнению программы на ЭВМ на машинном языке: программа загружается в оперативную память, в соответствие с программой вводятся исходные данные и начинается счет. Проявление ошибки в процессе вода исходных данных или в процессе счета приводит к прерыванию счета и выдаче диагностического сообщения рабочей программы. Проявление ошибки дает повод для выполнения отладочных действий.

Будут проводиться следующие тесты:

* Нагрузочное-тестирование;
* Стресс тестирование;
* Функциональное тестирование;
* Кроссплатформенность;
* Юзабилити тест.

Контрольные примеры (тесты) – это специально подобранные задачи, результаты которых заранее известны или могут быть определены без существенных затрат.

Контрольным примером для данного приложения является добавление курса, подходящей для него вакансии и безработного гражданина, после чего обработать заявление гражданина присвоив интересующую его вакансию, которая была добавлена ранее.

В результате должен выводиться список подходящих курсов и в котором будет ранее созданный курс (Что означает успешную обработку заявления).

## 2.1.2. Составление тестовых сценариев

**Тест-кейс — это чёткое описание действий, которые необходимо выполнить, для того чтобы проверить работу программы (поля для ввода, кнопки и т.д.). Данное описание содержит: действия, которые надо выполнить до начала проверки — предусловия; действия, которые надо выполнить для проверки — шаги описываются в следующих пунктах:**

* **производится авторизация в личный кабинет;**
* **развертывание окна со всеми элементами, которые изменяют свою форму под полноэкранный режим;**
* **производится поиск гражданина по паспорту, таким способом, что ели ввести одно значение, оно будет подбирать все цифры и выявлять нужного пользователя из БД.**

**Целью документации тестового примера является определение и передача конкретных условий, которые необходимо проверить, чтобы дать возможность оценить систему.**

**Тестовые сценарии мотивируются многими вещами, но обычно включают в себя подмножество вариантов использования, характеристик производительности и рисков в проекте.**

В данном тест кейсе будет производится нагрузочный тест и стресс тест.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № п/п | Выполняемое действие | Тип используемого объекта и его название | Количество одновременно обрабатываемых объектов | Входные условия | Процент загрузки процессора | Процент загрузки оперативной памяти | Процент занятости жёсткого диска |
| 1. |  |  |  |  |  |  |  |
| 2. |  |  |  |  |  |  |  |
| 3. |  |  |  |  |  |  |  |
| 4. |  |  |  |  |  |  |  |

В данном тест кейсе будет производится кроссплатформенный тест.

|  |  |  |
| --- | --- | --- |
| Название версии | Статус | Скриншот |
| Windows 10 |  |  |
| Windows 8.1 |  |  |
| Windows 7 |  |  |

|  |  |  |
| --- | --- | --- |
| Название: | Создание заказа | |
| Функция: | Функция обработки заявления | |
| Аппаратная среда: | 1. Процессор: Intel(R) Core(TM) i5-7600 CPU @ 3.50GHz (4 CPUs), ~3.5GHz 2. Установленная оперативная память (RAM): 8192MB RAM 3. Видеоадаптер: NVIDIA GeForce GTX 1060 6GB 4. Дисковые устройства: HDD – 1 ТБ | |
| Действие | Ожидаемый результат | Фактический результат |
| Предусловие: |  | |
| Открыть в приложении функцию обработки заявления | Окно открыто |  |
| Шаги теста: |  | |
| Выбрать гражданина и присвоить вакансии | Данные введены |  |
| Нажать кнопку «Обработать заявку» | Появилось окно с подходящими курсами |  |
| Постусловие: |  |  |
| Выбрать интересующий курс | Гражданин назначен на курс обучения |  |
| ИТОГ: |  |  |

|  |  |  |
| --- | --- | --- |
| Название: | Обновление таблиц | |
| Функция: | Функция обновления таблиц по истечению таймера | |
| Аппаратная среда: | 1. Процессор: Intel(R) Core(TM) i5-7600 CPU @ 3.50GHz (4 CPUs), ~3.5GHz 2. Установленная оперативная память (RAM): 8192MB RAM 3. Видеоадаптер: NVIDIA GeForce GTX 1060 6GB   Дисковые устройства: HDD – 1 ТБ | |
| Действие | Ожидаемый результат | Фактический результат |
| Предусловие: |  | |
| Заполнить все поля для добавления гражданина | Поля заполнены |  |
| Шаги теста: |  | |
| Нажать на кнопку добавить гражданина | Гражданин добавлен |  |
| Постусловие: |  |  |
| Список обновился | В список добавился новый гражданин |  |
| ИТОГ: |  |  |

В данных тест кейсах в дальнейшем будут проводиться тестирование программных модулей. Проверяется (Фактический результат, Статус (проход/ неудача)).

Usability-тест

Продолжение таблицы 14

|  |  |  |
| --- | --- | --- |
| Характеристика | Соответствие | Описание |
| Информативность системы |  |  |
| Понятность |  |  |
| Запасные выходы |  |  |
| Однозначность |  |  |
| Предусмотрительность |  |  |
| Наглядность |  |  |
| Гибкость и эффективность |  |  |
| Лаконичность и точность |  |  |
| Лояльность к ошибкам |  |  |
| Постоянная справка |  |  |
| Конфиденциальность ввода |  |  |
| Субъективное удобство использования |  |  |

## 2.1.3 Осуществление тестирования программного продукта и оформление тестовой документации

Тестирование программного обеспечения — процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

Нагрузочное-тестирование

Тестовое окружение:

1. Процессор: Intel(R) Core(TM) i5-7600 CPU @ 3.50GHz (4 CPUs), ~3.5GHz
2. Установленная оперативная память (RAM): 8192MB RAM
3. Видеоадаптер: NVIDIA GeForce GTX 1060 6GB
4. Дисковые устройства: HDD – 1 ТБ

|  |  |  |
| --- | --- | --- |
| Процент загрузки процессора | Процент загрузки оперативной памяти | Процент занятости жёсткого диска |
| 4 | 37 | 2 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № п/п | Выполняемое действие | Тип используемого объекта и его название | Количество одновременно обрабатываемых объектов | Входные условия | Процент загрузки процессора | Процент загрузки оперативной памяти | Процент занятости жёсткого диска |
| 1. | Обновление таблиц | Окно «Просмотр вакансий»  БД «Education» | 1 окно  1 таблица  Все записи | Все записи из таблиц безработные граждане | 25 | 44 | 4 |
| 2. | Добавить вакансию | Окно «Добавление вакансии»  БД «Education» | 1 окно  1 таблицы  1 запись | Добавление записи в таблицу вакансий | 22 | 38 | 3 |
| 3. | Поиск гражданина по паспорту | Окно «Обработка заявлений»  БД «Education» | 1 окно  1 таблица  Все записи | нет | 23 | 40 | 4 |
| 4. | Обработка и вывод подходящих курсов | Окно «обработка заявлений»  БД «Education» | 1 окно  2 таблица  28 записей | Выбран гражданин  Выбраны интересующие вакансии | 34 | 58 | 30 |

Результаты функциональных тестов.

|  |  |  |
| --- | --- | --- |
| Название: | Создание заказа | |
| Функция: | Функция обработки заявления | |
| Аппаратная среда: | 1. Процессор: Intel(R) Core(TM) i5-7600 CPU @ 3.50GHz (4 CPUs), ~3.5GHz 2. Установленная оперативная память (RAM): 8192MB RAM 3. Видеоадаптер: NVIDIA GeForce GTX 1060 6GB 4. Дисковые устройства: HDD – 1 ТБ | |
| Действие | Ожидаемый результат | Фактический результат |
| Предусловие: |  | |
| Открыть в приложении функцию обработки заявления | Окно открыто |  |
| Шаги теста: |  | |
| Выбрать гражданина и присвоить вакансии | Данные введены |  |
| Нажать кнопку «Обработать заявку» | Появилось окно с подходящими курсами |  |
| Постусловие: |  |  |
| Выбрать интересующий курс | Гражданин назначен на курс обучения |  |
| ИТОГ: |  |  |

|  |  |  |
| --- | --- | --- |
| Название: | Обновление таблиц | |
| Функция: | Функция обновления таблиц по истечению таймера | |
| Аппаратная среда: | 1. Процессор: Intel(R) Core(TM) i5-7600 CPU @ 3.50GHz (4 CPUs), ~3.5GHz 2. Установленная оперативная память (RAM): 8192MB RAM 3. Видеоадаптер: NVIDIA GeForce GTX 1060 6GB   Дисковые устройства: HDD – 1 ТБ | |
| Действие | Ожидаемый результат | Фактический результат |
| Предусловие: |  | |
| Заполнить все поля для добавления гражданина | Поля заполнены |  |
| Шаги теста: |  | |
| Нажать на кнопку добавить гражданина | Гражданин добавлен |  |
| Постусловие: |  |  |
| Список обновился | В список добавился новый гражданин |  |
| ИТОГ: |  |  |

Программное обеспечение было предложено целевой аудитории. Предлагалось также заполнить таблицу юзабилити теста. Объединив все предложения и замечания целевой аудитории, была сформирована следующая таблица:

|  |  |  |
| --- | --- | --- |
| Характеристика | Соответствие | Описание |
| Информативность системы | 7/10 | В приложении нет обозначений авторизованного пользователя;  Не поясняется, назначился ли курс гражданину;  Во время регистрации при вводе неправильных данных не указывается, где неправильно; |
| Понятность | 10/10 | Приложение разделено на интуитивно понятные вкладки |
| Запасные выходы | 10/10 | На каждом окне есть кнопка отмены и выхода |
| Однозначность | 10/10 |  |
| Предусмотрительность | 4/10 | Можно вводить цифры в ФИО;  Не масштабируемый интерфейс;  Можно ввести слишком большие номера телефонов и номера паспорта что вызывает ошибку;  После создания вакансии поля о вакансии не обнуляются; |
| Наглядность | 6/10 | Кнопка регистрации никак не обозначена; |
| Гибкость и эффективность | 5/10 | Присутствует только базовый функционал. Никаких других функций нет; |
| Лаконичность и точность | 7/10 | Программа лаконична с незначительными недочётами. |
| Лояльность к ошибкам | 10/10 | Нельзя открыть несколько окон регистрации;  Можно совершать ошибки и не будет останавливаться работа программы; |
| Постоянная справка | 10/10 | Справочная информация присутствует на всех формах и она понятна; |
| Конфиденциальность ввода | 7/10 | Показывается пароль при входе в приложение; |
| Субъективное удобство использования | 10/10 |  |

|  |  |  |
| --- | --- | --- |
| Название версии | Статус | Скриншот |
| Windows 10 | Запуск успешный |  |
| Windows 8.1 | Запуск успешный |  |
| Windows 7 | Запуск успешный |  |

## 2.2. Построение схемы итоговых приемо-сдаточных испытаний

Приёмо-сдаточные испытания – испытания, которым подвергают программное обеспечение с целью установления соответствия его определённым требованиям.

Основная цель проведения приёмо-сдаточных испытаний – контроль соответствия продукции требованиям, которые установлены для данной категории испытаний, а также контрольному образцу для определения возможности приёмки продукции.

Приёмо-сдаточные испытания необходимо проводить на функционирующем объекте, а для их проведения должно быть предъявлено техническое задание и методика испытаний.

В ходе приёмо-сдаточных испытаний должны проверятся следующие аспекты:

1) полнота и качество реализации функций программного обеспечения;

2) выполнение каждого требования, который относится к интерфейсу системы;

3) работа персонала в диалоговом режиме;

4) средства и методы восстановления работоспособности программного обеспечения после отказов;

5) комплектность и качество эксплуатационной документации.

В схеме итоговых приёмо-сдаточных испытаний указывается следующая информация:

1) перечень объектов, выделенных в системе для испытаний, а также перечень требований, которым должны соответствовать объекты;

2) критерии приёмки системы и её частей;

3) условия и сроки проведения испытаний;

4) инициалы лиц, ответственных за проведение испытаний;

5) методику испытаний и обработки их результатов;

6) перечень оформляемой документации.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Номер услуги | Наименование | Описание | Период оказания | | Заказчик | Статус | Подпись заказчика |
| с | по |
|  |  |  |  |  |  |  |  |

В соответствии с параметрами, указанными в данном пункте производственной практики, была составлена схема итоговых приёмочно-сдаточных испытаний, которая представлена в таблице.

## 2.3 Работа с системой контроля версий

Система контроля версий – программное обеспечение для облегчения работы с изменяющейся информацией. Она позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, а также определять лицо, сделавшего то или иное изменение.

Основная цель системы контроля версий – возврат к исходной точке при большом количестве недочётов, ошибок в проекте.

Git – это гибкаясистема контроля версий, дающая массу возможностей не только разработчикам программных продуктов, но и писателям для изменения, дополнения и отслеживания изменения «рукописей» и сюжетных линий, и учителям для корректировки и развития курса лекций, и администраторам для ведения документации, и для многих других направлений, требующих управления историей изменений. Поэтому данный проект разрабатывается с использованием системы Git.

1. Сперва нужно создать репозиторий

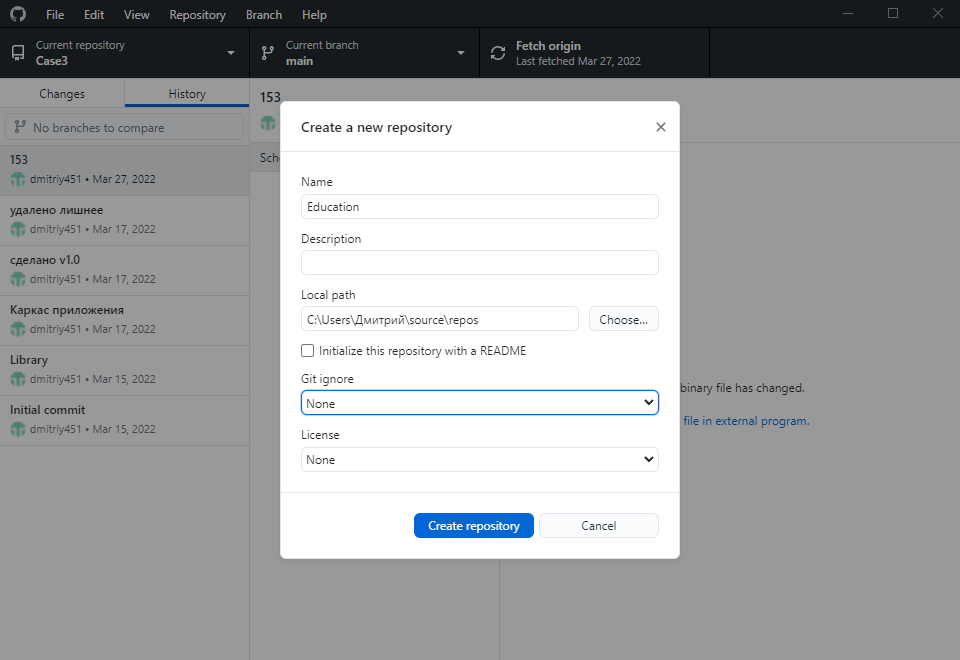


Рисунок 6 – Созадние репозитория

1. После чего нужно добавить проект в репозиторий

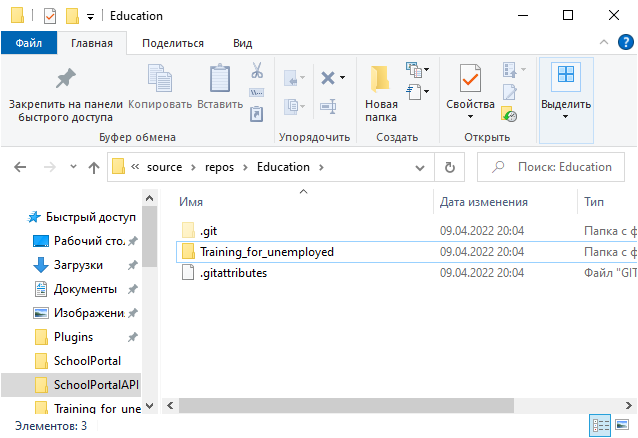


Рисунок 7 – Добавление проекта в репозиторий

1. После чего нужно сохранить изменения и опубликовать на сервере

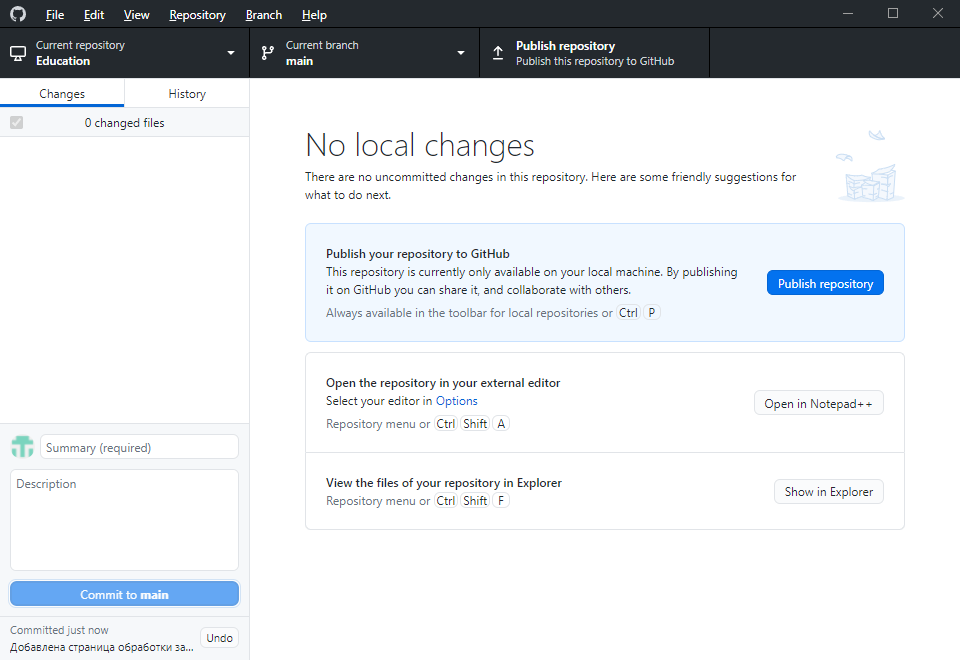


Рисунок 8 – Сохранение изменений

1. В окне GitHub появились новые файлы их изменения нужно сохранить в репозиторий добавив название и описание изменения

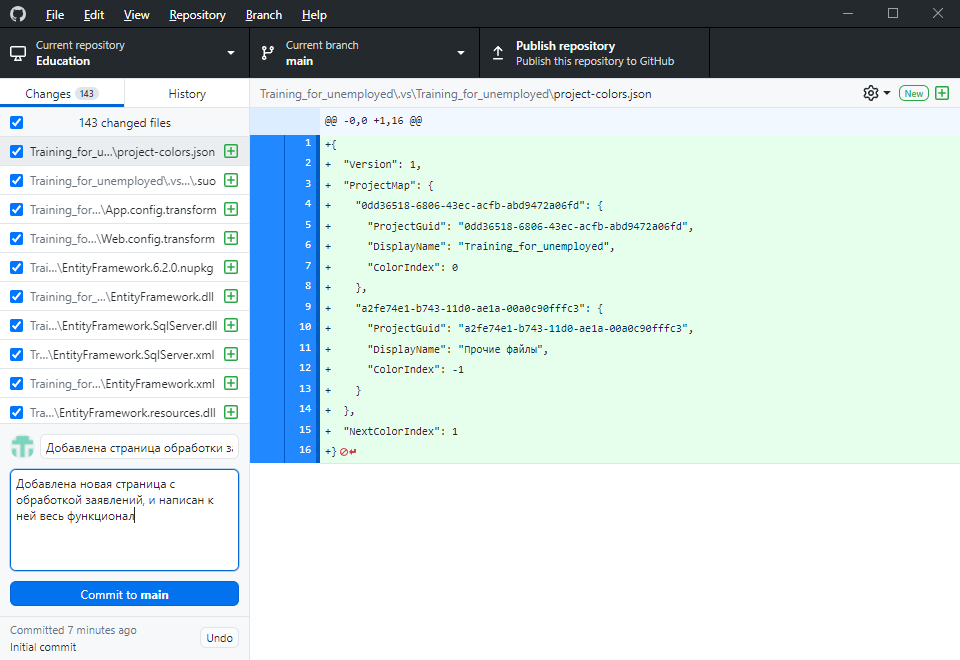


Рисунок 9 – Добавление файлов

1. Затем нужно изменения нужно запушить на сервер Git

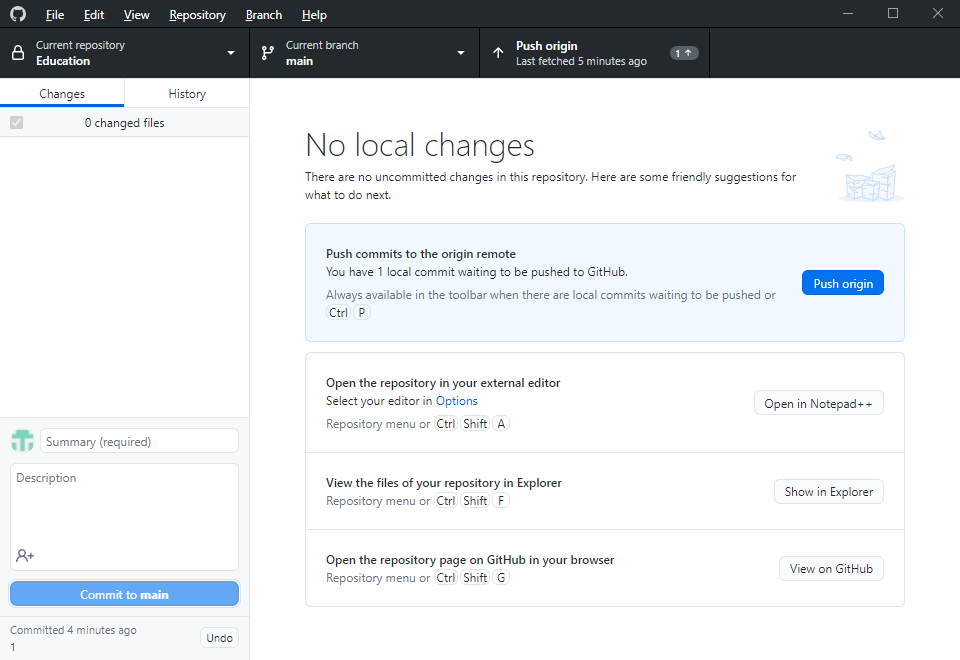


Рисунок 10 – Push изменений на сервер

Эти действия нужно проделывать каждый раз, когда выполнен какой-либо пункт из большого проекта

## 2.4 Создание установочных файлов программного продукта

Для корректного распространения программного обеспечения, его стоит упаковать в установочный файл, создаваться он будет с помощью программы Smart Install Maker. Сперва нужно заполнить информацию о программе (см рисунок 21).

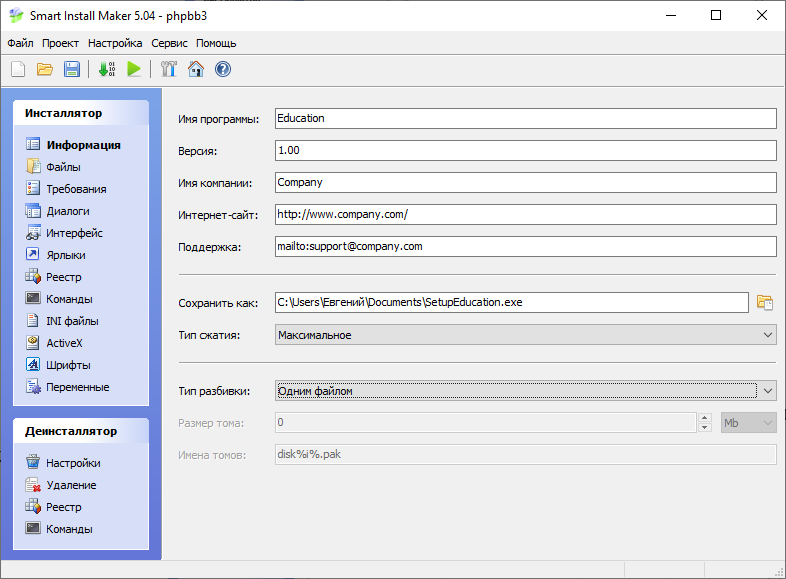


Рисунок 11 – Информация о продукте

После чего нужно выбрать файлы, которые должны устанавливаться (см рисунок 22).

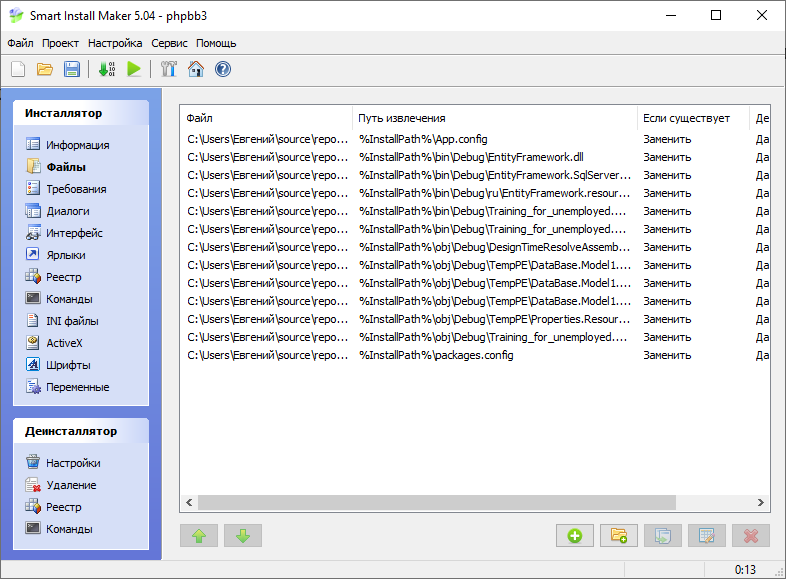


Рисунок 12 – Устанавливаемые файлы

После чего нужно создать установочный файл и проверить его:

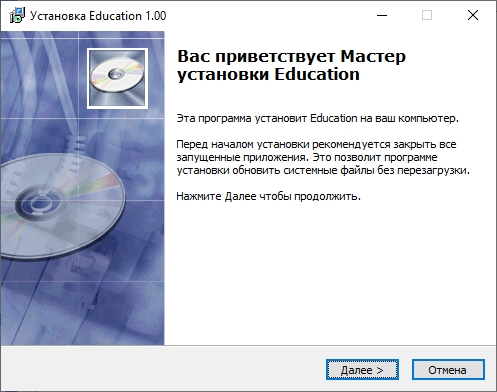


Рисунок 13 – Главное окно

На данном окне нужно ознакомиться с основной информацией об установке. После ознакомления нажать кнопку «Далее». Будет совершен переход на следующее окно с выбором пути загрузки

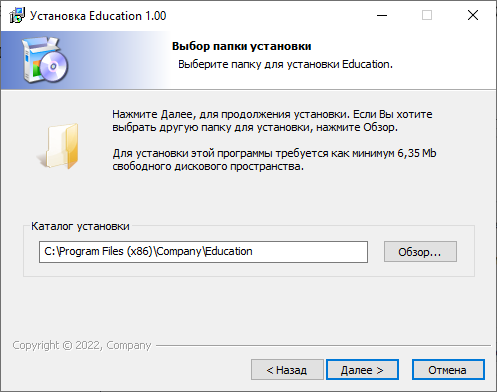


Рисунок 14 – Выбор пути установки

На данном окне нужно ввести предпочитаемый каталог установки. Нужно обратить внимание на требуемое количество памяти и свободное, для комфортной работы свободной должно быть в разы больше требуемой. Так же нужно запомнить указанный путь так как в будущем нужно из него скопировать базу данных. После чего нужно нажать кнопку «Далее».

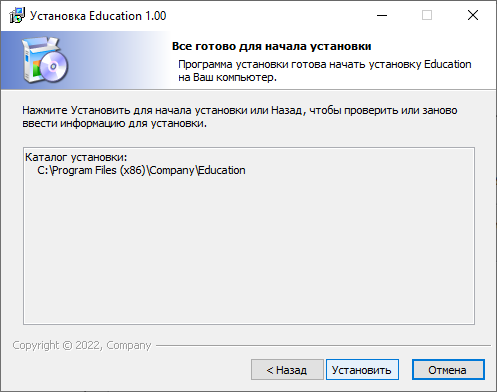


Рисунок 15 – Проверка введённых данных

На данном окне можно ещё раз проверить введённый путь и после проверки пути нужно нажать на кнопку «Установить» откроется окно установки.

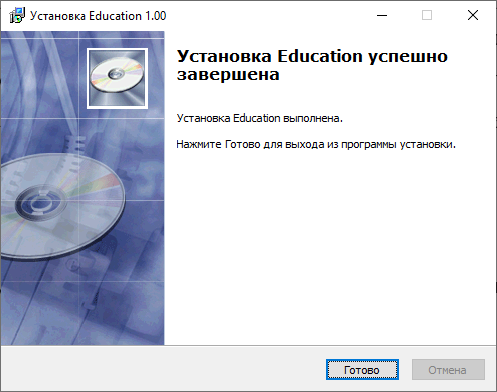


Рисунок 16 – Конечное Окно

Благодаря установщику приложение можно разворачивать на любом компьютере.

## 2.5 Проведение испытаний и мероприятий по интеграции программного продукта в информационную систему организации

Интеграция программного продукта – элемент процесса разработки программного обеспечения, в ходе которого его отдельные компоненты объединяются в единое целое.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Номер услуги | Наименование | Описание | Период оказания | | Заказчик | Статус | Подпись заказчика |
| с | по |
| 1 | Проверка работоспособности ИС | Проверка работоспособности ПК и установленных на них ПП | 18.02.2022 | 12.04.2022 | Ермачков С.В. | Пройдено |  |
| 2 | Установка лицензированного программного обеспечения, указанного в техническом задании | Установка всех необходимых программ для работы программного обеспечения, прописанные в техническом задании | 18.02.2022 | 12.04.2022 | Ермачков С.В. | Пройдено |  |
| 3 | Инсталляция разработанного приложения | Инсталляция разработанного приложения на компьютер пользователя | 18.02.2022 | 12.04.2022 | Ермачков С.В. | Пройдено |  |
| 4 | Настройка приложения | Настройка приложения до его полной работоспособности | 18.02.2022 | 12.04.2022 | Ермачков С.В. | Пройдено |  |
| 5 | Тестирование работоспособности разработанного приложения | Тестирование работоспособности разработанного приложения на работоспособность | 18.02.2022 | 12.04.2022 | Ермачков С.В. | Пройдено |  |

# Заключение

В ходе проделанной работы были решены следующие задачи:

* Изучена предметная область, итогом которой является словесное описание и формализованное описание предметной области в виде диаграмм;
* Выполнено моделирование предметной области;
* Выполнено определение требований к программному продукту;
* Выполнено тестирование программных модулей;
* Выполнена отладка приложения.

Приложение имеет удобный интерфейс, интуитивно-понятные связи между диалогами ввода информации, а также быстрый подбор курсов.

Тестирования, проведённые во время разработки, исключают возможность ошибок, а приложение просто в установке и эксплуатации, что позволяет очень быстро внедрить его и начать работу.

Был получен дополнительный опыт разработки проектной документации. Данный полученный опыт помогает в проектировании и разработки документации к различным проектам. Так, например, опыт разработки технического задания помогает проще ориентироваться в самой структуре документа. Также данный опыт помогает легче разрабатывать документацию к проектам.

В ходе выполнения практики были приобретены умения анализа предметной области, написания отчета.

В ходе выполнения практики был получен опыт работы в Воскресенсков управлении социальной защиты населения.

В ходе выполнения практики была выполнена документация, связанная с приложением.

В итоге проделанной работы было создана документация, связанная с приложением.

# Список литературы

1. ГОСТ 51188 - 98 «Защита информации» - Методы проведения испытаний программных средств на наличие компьютерных вирусов (Пункт 5.1);
2. Информационные технологии. Анализ и проектирование информационных систем / Рочев Константин Васильевич. -М: Лань, 2019. – 128 с.;
3. Осуществление интеграции программных модулей: учебник для студ. учреждений сред. проф. образования/ В.П. Зверева, А.В. Назаров. – М.: Издательский центр «Академия», 2018. – 256 с.;
4. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Эрих Гамма, Джон Влисидис, Ричард Хелм, Ральф Джонсон. - СПб: Питер,2016. – 366 с.;
5. Путь программиста / Джон Сонмез. - СПБ: Питер, 2016. – 448 с.;
6. Совершенный код. Мастер-класс / Стив Макконнелл. – СПб: БХВ, 2017. – 896 с.;
7. Чистый код: создание, анализ и рефакторинг / Роберт Мартин. – СПб: Питер, 2021. – 464 с.;
8. Как тестируют в Google / James A, Whittaker Jason, Arbon Jeff Carollo. – СПб: Питер 2019. – 272 с.;
9. Perfect Software and other illusions about testing / Gerald M. Weinberg, – New York: Dorset House 2017. – 128 с.;
10. Тестирование программного обеспечения. Базовый курс / Святослав Куликов. – Минск: Четыре четверти, 2020. – 362 с.;

Интернет-источники:

1. https://www.matburo.ru/Examples/Files/Games5.pdf (8 декабря 2021 года / 17:25);
2. https://habr.com/ru/post/163681/ (11 декабря 2021 года / 10:38);
3. https://intuit.ru/studies/courses/66/66/lecture/27219 (11 декабря 2021 года / 12:24);
4. https://mydocx.ru/6-141987.html (11 декабря 2021 года / 14:07);
5. https://matworld.ru/linear-programming/matrichnye-igry.php (12 декабря 2021 года / 13:38);

# Приложение А.

Руководство программиста

Структура и оформление документа устанавливаются в соответствии с ГОСТ 19.105-78 - Единая система программной документации. Общие требования к программным документам. Настоящий стандарт устанавливает общие требования к оформлению программных документов для вычислительных машин, комплексов и систем независимо от их назначения и области применения и предусмотренных стандартами Единой системы программной документации (ЕСПД) для любого способа выполнения документов на различных носителях данных.

Назначение и условия применения программы и характеристики программы.

Данное программное обеспечение может предоставлять гражданам интересующие их курсы основываясь на предпочитаемых вакансиях. Для начала сотрудник должен авторизоваться в системе используя свои логин и пароль. (Рисунок 16).

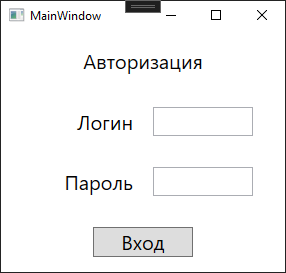


Рисунок 17 - Авторизация

После чего сотрудник переходит на окно навигации где он может работать с гражданами, вакансиями и курсами обучения.

* + 1. Работа с вакансиями

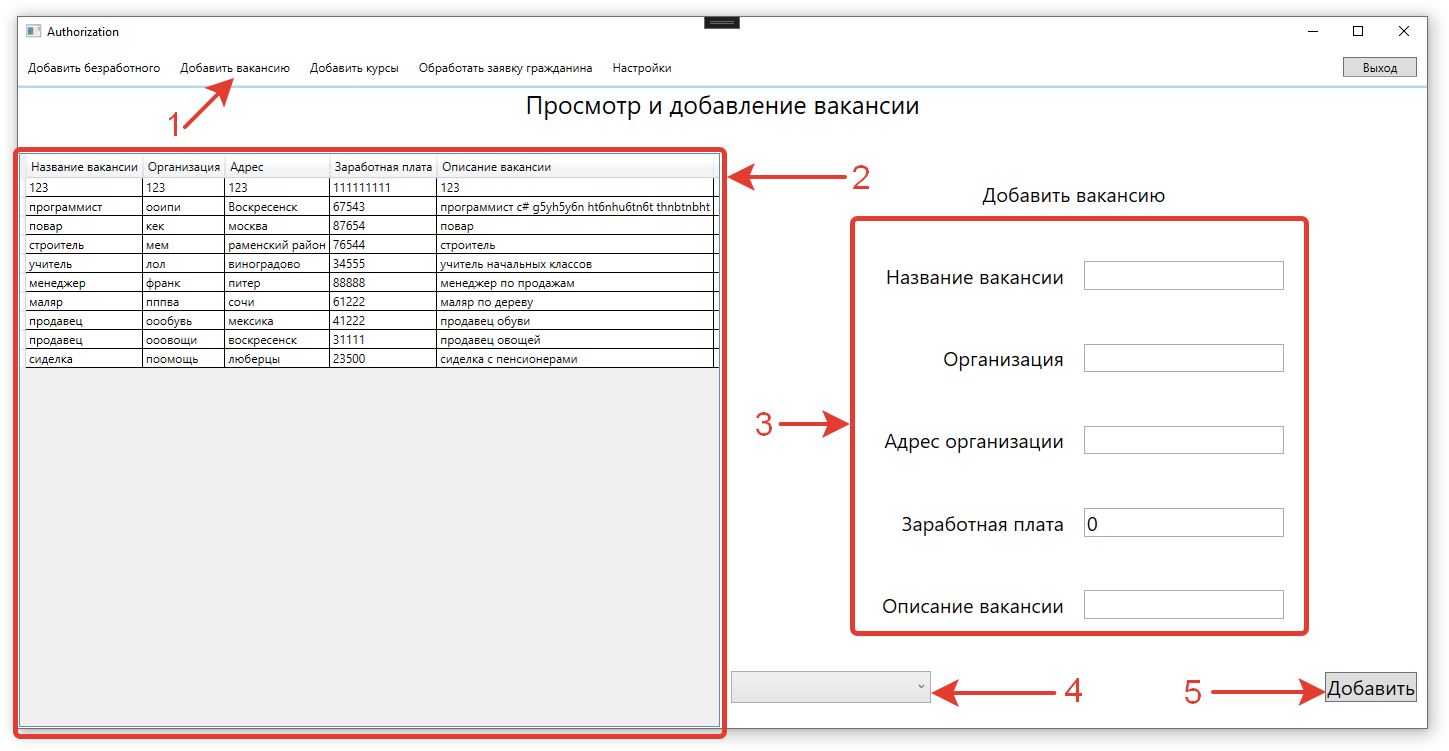


Рисунок 18 – Форма просмотра и добавления вакансий

Для работы с вакансиями нужно перейти во вкладку «Добавить вакансию». В левой части окна можно посмотреть список уже созданных вакансий (на скриншоте помечено цифрой 2), в правой части реализована функция добавления новой вакансии. Чтобы добавить вакансию нужно заполнить все предложенные поля (на скриншоте помечено цифрой 3). После заполнения всех полей нужно присвоить подходящие для данной вакансии теги (на рисунке помечено цифрой 4). После того как все поля заполнены и теги добавлены нужно нажать на кнопку «Добавить» (на рисунке помечено цифрой 5).

* + 1. Работа с курсами

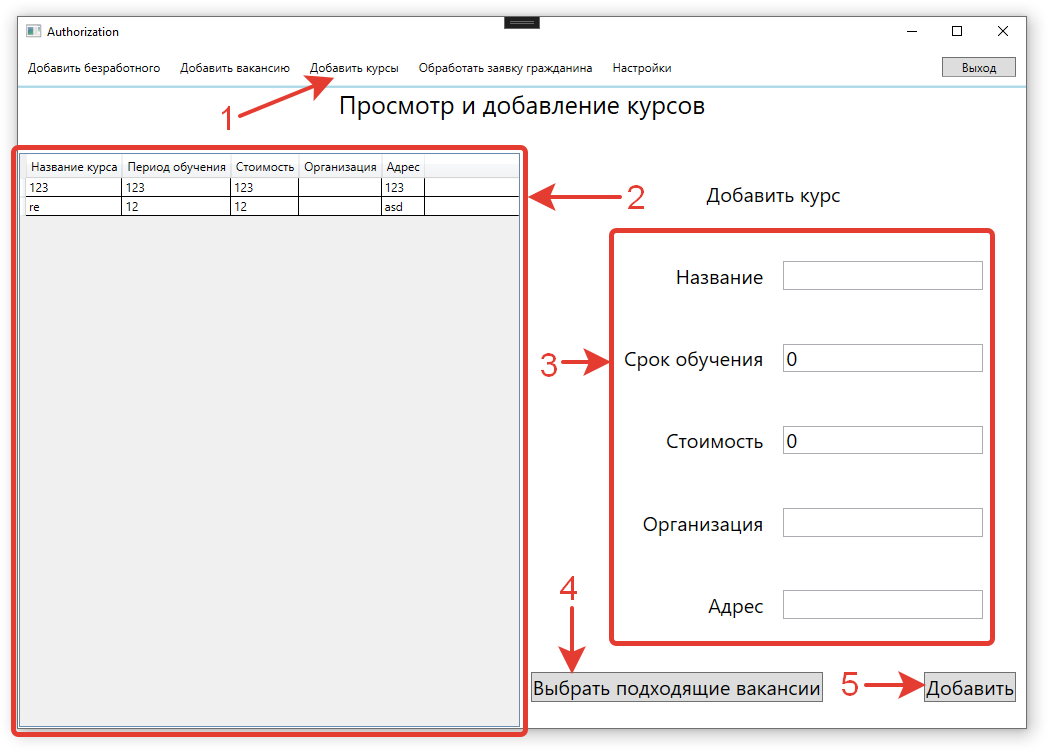


Рисунок 19 – Форма просмотра и добавления курсов

Для работы с курсами обучения нужно перейти во вкладку «Добавить курсы». В левой части окна можно посмотреть список уже созданных курсов (на скриншоте помечено цифрой 2), в правой части реализована функция добавления нового курса. Чтобы добавить курс нужно заполнить все предложенные поля (на скриншоте помечено цифрой 3). После заполнения всех полей нужно присвоить подходящие для данного курса вакансии (на рисунке помечено цифрой 4). После того как все поля заполнены и подходящие вакансии добавлены нужно нажать на кнопку «Добавить» (на рисунке помечено цифрой 5).

* + 1. Работа с безработными гражданами

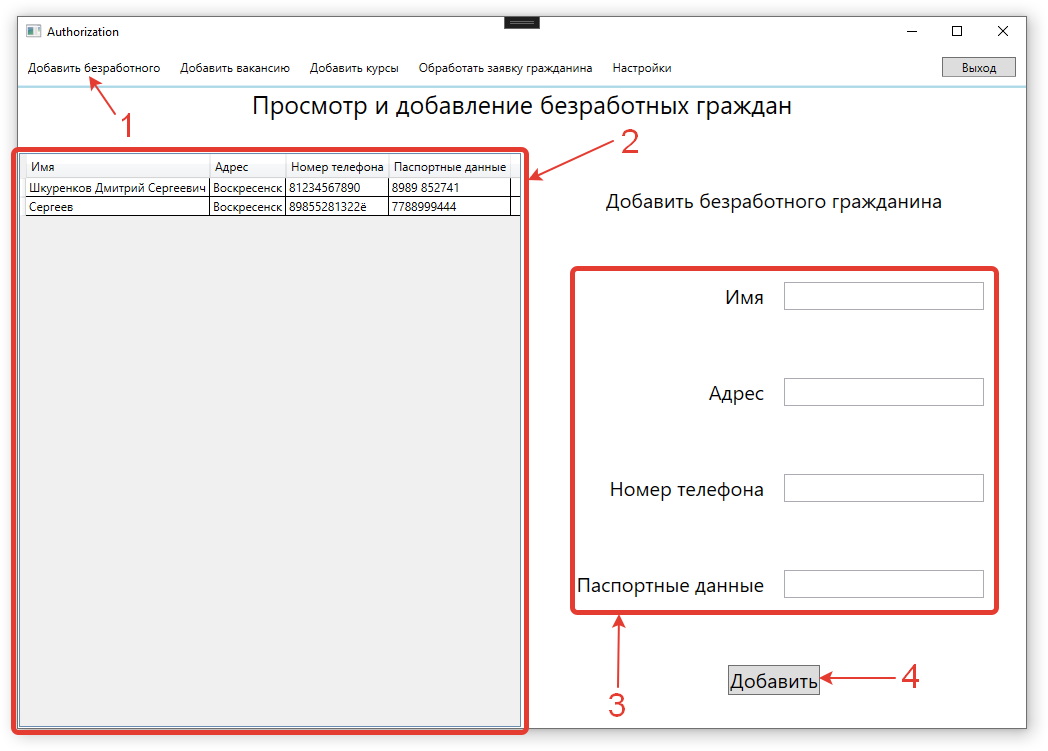


Рисунок 20 – Форма просмотра и добавления безработных граждан

Для работы с безработными гражданами сперва нужно перейти во вкладку «Добавить безработного», проверить есть ли заявитель в текущей базе, если нет, то нужно добавить гражданина. В левой части окна можно посмотреть список уже добавленных граждан (на скриншоте помечено цифрой 2), в правой части реализована функция добавления нового гражданина. Чтобы добавить гражданина нужно заполнить все предложенные поля (на скриншоте помечено цифрой 3) После того как все поля нужно нажать на кнопку «Добавить» (на рисунке помечено цифрой 4).

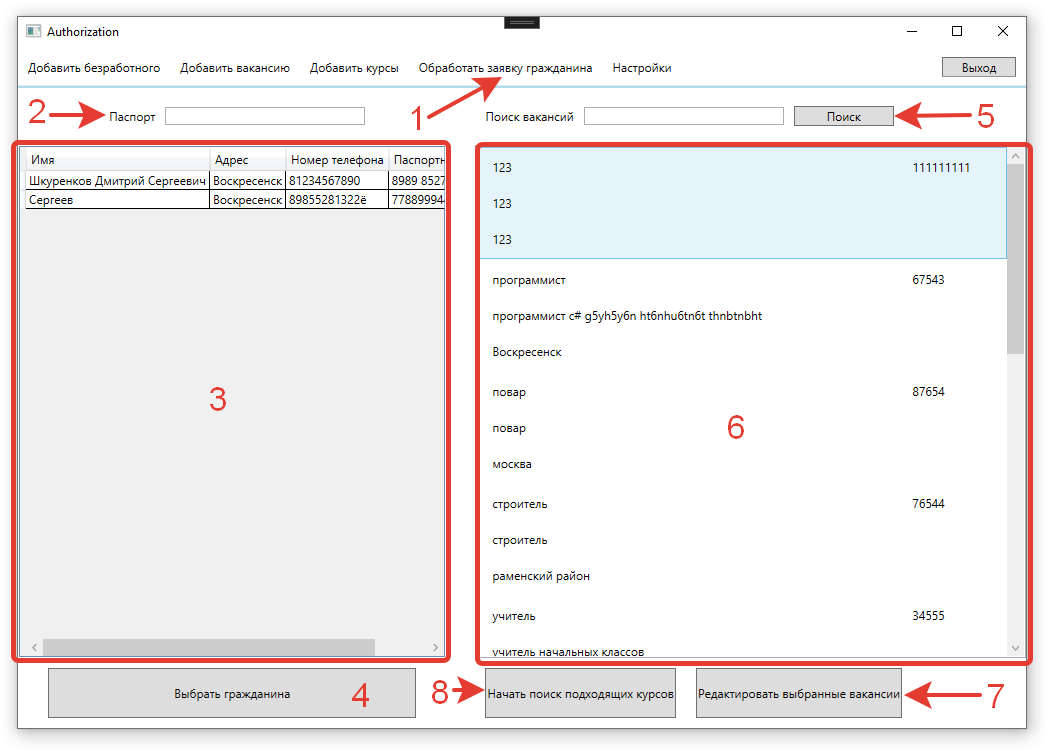


Рисунок 21 – Форма обработки заявок граждан

Теперь нужно перейти во вкладку «Обработать заявку гражданина» (на рисунке отмечено цифрой 1). В поле Паспорт (на рисунке отмечено цифрой 2) ввести паспорт заявителя и из списка выбрать соответствующего гражданина (на рисунке отмечено цифрой 3) и нажать кнопку «Выбрать гражданина» (на рисунке отмечено цифрой 4).

После выбора гражданина нужно ввести в поисковую строку интересующие его вакансии и нажать «Поиск» (на рисунке отмечено цифрой 5).

# Приложение Б.

Контрольный пример

Контрольный пример – документ, который оформляет программное обеспечение и позволяет осуществить полную проверку функционирования разработанного программного модуля. Для проведения контрольного примера выступает функции добавления курса, подходящей для него вакансии и безработного гражданина, после чего обработать заявление гражданина присвоив интересующую его вакансию, которая была добавлена ранее.

В результате должен выводиться список подходящих курсов и в котором будет ранее созданный курс (Что означает успешную обработку заявления).

Для добавления вакансии нужно перейти на вкладку «Добавить вакансию» и заполнить все необходимые поля, добавить тег С#, после чего нажать на кнопку «добавить».

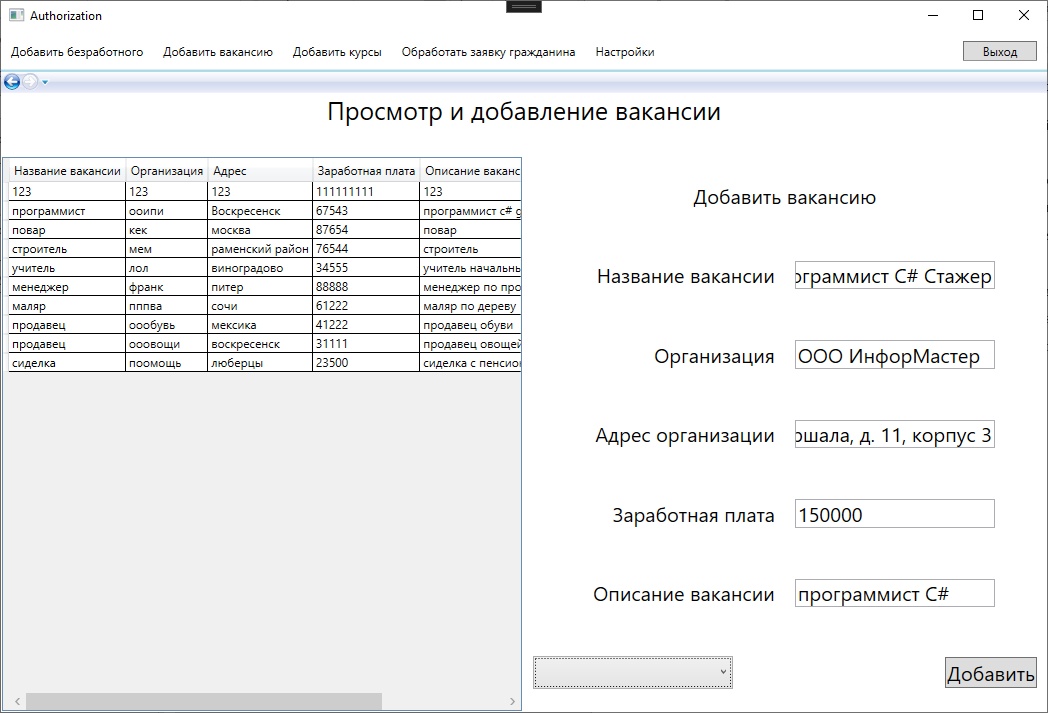


Рисунок 22 – Функция добавления вакансии

После добавления вакансии нужно добавить подходящий курс обучения. Необходимо перейти на вкладку «Добавить курсы» и заполнить все необходимые поля, присвоить подходящую ранее созданную вакансию, после чего нажать на кнопку «добавить».

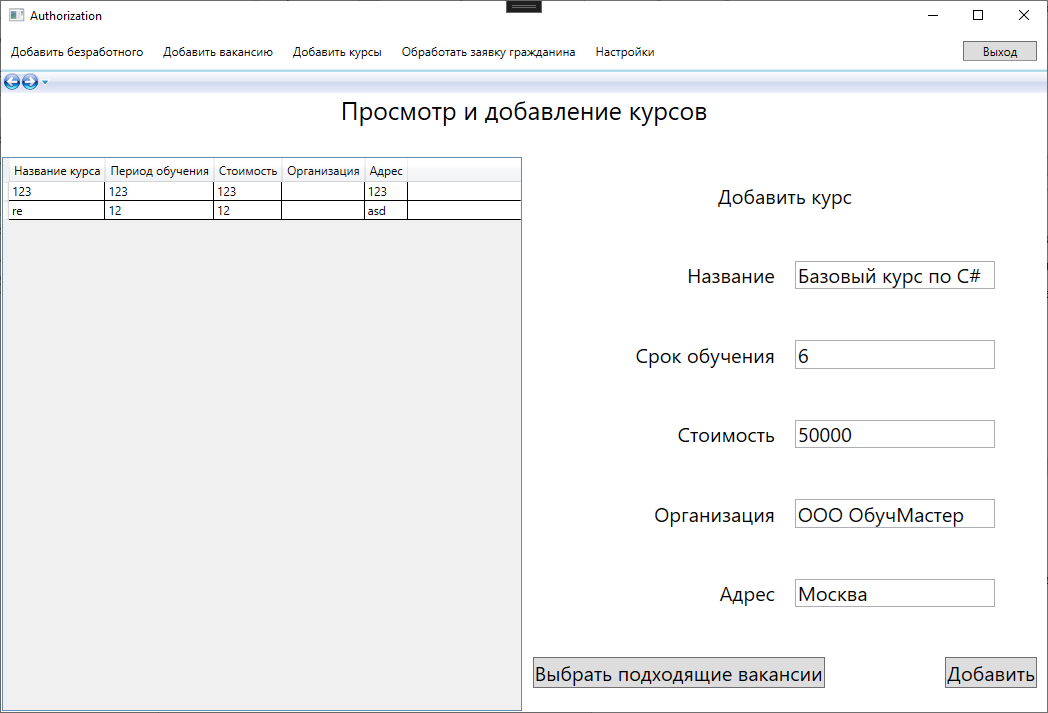


Рисунок 23 – Функция добавления курса

Теперь необходимо перейти на вкладку «Добавить безработного» и заполнить все необходимые поля, присвоить подходящую ранее созданную вакансию, после чего нажать на кнопку «добавить».



Рисунок 24 – Функция добавления гражданина

Теперь необходимо обработать заявку нового гражданина присвоив ранее созданную интересующую вакансию и начать поиск подходящих курсов.

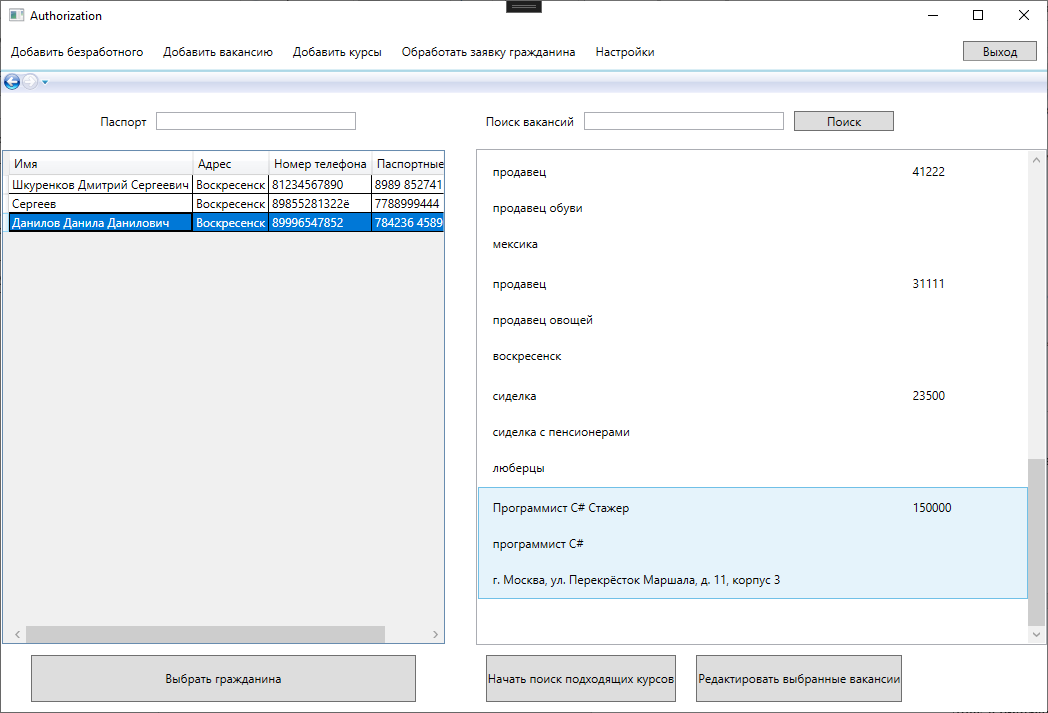


Рисунок 25 – Функция обработки заявлений граждан



Рисунок 26 – Страница подходящих курсов

Как итог выводится ранее добавленный курс, это значит, что контрольный пример был пройден.

# Приложение В

Диск с программой