**МИНОБРНАУКИ РОССИИ**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«Чувашский государственный университет имени И.Н. Ульянова»**

**(ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»)**

**Факультет информатики и вычислительной техники**

**Кафедра вычислительной техники**

**ЗАДАНИЕ**

**студенту-практиканту**

***Захарову Владимиру Владимировичу, ИВТ-42-23***

для прохождения учебной практики (ознакомительной практики) на

***кафедре вычислительной техники факультета информатики и вычислительной техники ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»***

1. [Ведение](http://www.xn-----8kcodrdcygecwgg0byh.xn--p1ai/vvedenie-kursovoi-raboty) и [оформление дневника практики](http://www.xn-----8kcodrdcygecwgg0byh.xn--p1ai/dnevnik-po-praktike).
2. Прохождение инструктажа по охране труда, технике безопасности, пожарной безопасности, а также ознакомление с правилами внутреннего трудового распорядка организации, предоставляющей место для прохождения практики.
3. Выполнение индивидуального задания:

* ознакомление с базой практики (профильной организацией), выпускаемой продукцией, структурой исследовательских, проектно-конструкторских и проектно-технологических подразделений, их ролью, задачами и взаимосвязями с другими подразделениями;
* изучение технологии создания программных средств вычислительной техники и автоматизированных систем: tkinter - набор инструментов разработки программного обеспечения для создания кроссплатформенных приложений с использованием языка программирования Python
* приобретение и закрепление навыков разработки алгоритмов и программ: верстка экранов десктопного приложения, построение и управление свойствами виджетов, работа со стилями, навигацией и архитектурой, разработка стейт-менеджмента и управление жизненным циклом приложения, установка и применение библиотек для реализации основного функционала, а также управления базой данных и обработки ошибок;
* приобретение навыков разработки и оформления отчетности и (или) программной документации/ документации пользователя;
* оформление отчета по практике в соответствии с рекомендациями п.п. 6,7 программы практики.

1. Планируемый результат: приложение для администрирования таблиц, с возможностью добавления, удаления и фильтрации данных, изменения существующих записей.

Руководитель практики от кафедры \_\_\_\_\_\_\_\_\_\_\_/ Юрьева Е.В.

Дата выдачи задания 26 июня 2024 года

Согласовано:

Руководитель практики от профильной организации \_\_\_\_\_\_\_\_\_\_/ Юрьева Е.В.

Дата согласования 26 июня 2024 года

**МИНОБРНАУКИ РОССИИ**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«Чувашский государственный университет имени И.Н. Ульянова»**

**(ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»)**

**Факультет информатики и вычислительной техники**

**Кафедра вычислительной техники**

ОТЧЕТ

О УЧЕБНОЙ ПРАКТИКЕ

(ОЗНАКОМИТЕЛЬНОЙ ПРАКТИКЕ)

на базе ***кафедры вычислительной техники факультета информатики и вычислительной техники ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»***

|  |  |  |
| --- | --- | --- |
| Обучающийся 1 курса, направление подготовки  «Информатика и вычислительная техника», группа ИВТ-42-23 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Захаров В. В. |
|  | подпись, дата |  |
|  |  |  |
|  |  |  |
| Руководитель,  ст.преподаватель кафедры вычислительной техники | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ подпись, дата | Юрьева Е.В. |
|  |  |
|  |  |  |
| Руководитель от профильной организации, ассистент кафедры вычислительной техники | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Васильев Н.С. |
|  | подпись, дата |  |
|  |  |  |
| Заведующий кафедрой  вычислительной техники, доцент | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ подпись, дата | Щипцова А.В. |
|  |  |
|  |  |  |
|  |  |  |

Чебоксары 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 4

[1 Теоретическая часть 5](#_Toc171367494)

[1.1 Язык программирования Python 5](#_Toc171367496)

[1.2 Библиотека tkinter 7](#_Toc171367497)

[1.3 Библиотека SQLAlchemy 9](#_Toc171367498)

[1.4 Среда разработки PyCharm 11](#_Toc171367499)

[2 Практическая часть](#_Toc171367502) 10

[2.1 Техническое задание 12](#_Toc171367503)

[2.2 Анализ поставленной задачи и выбор пути ее решения](#_Toc171367504) 13

[2.3 Создание моделей таблиц](#_Toc171367504) 13

[2.4 Добавление записи](#_Toc171367505) 14

[2.5 Отображение данных](#_Toc171367506) 15

[2.6 Поиск записей](#_Toc171367507) 15

[2.7 Обновление данных](#_Toc171367508) 17

[2.8 Удаление данных 18](#_Toc171367509)

[2.9 Экспорт в XML 18](#_Toc171367509)

[2.10 Чтение XML файла 20](#_Toc171367509)

[ЗАКЛЮЧЕНИЕ 21](#_Toc171367511)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 22](#_Toc171367512)

ВВЕДЕНИЕ

Цели практики:

– получение профессиональных умений и опыта технологической (проектно-технологической) деятельности;

– закрепление, расширение и углубление теоретических и практических знаний умений и навыков, полученных ранее при изучении дисциплин учебного плана.

Задачи практики задачи:

* изучение технологии создания программных средств вычислительной техники и автоматизированных систем: tkinter - набор инструментов разработки программного обеспечения для создания кроссплатформенных приложений с использованием языка программирования Python
* приобретение и закрепление навыков разработки алгоритмов и программ: верстка экранов десктопного приложения, построение и управление свойствами виджетов, разработка стейт-менеджмента и управление жизненным циклом приложения, установка и применение библиотек для реализации основного функционала, а также управления базой данных и обработки ошибок;
* приобретение навыков разработки и оформления отчетности и (или) программной документации/ документации пользователя;
* оформление отчета по практике в соответствии с рекомендациями п.п. 6,7 программы практики.
* Планируемый результат: приложение для администрирования таблиц, с возможностью добавления, удаления и фильтрации данных, изменения существующих записей.

**1 Теоретическая часть**

**1.1 Язык программирования Python**

Python — это высокоуровневый язык программирования, созданный Гвидо ван Россумом и выпущенный в 1991 году. Он разработан с акцентом на простоту, удобочитаемость и продуктивность, что делает его одним из самых популярных языков программирования в мире. Python широко используется в веб-разработке, анализе данных, искусственном интеллекте, автоматизации и многих других областях.

Преимущества Python:

* **Простота и читаемость:** Python известен своим лаконичным и понятным синтаксисом, что позволяет писать программы быстрее и с меньшими усилиями. Его код легко читать, что упрощает поддержку и расширение приложений.
* **Динамическая типизация:** В Python тип переменных определяется автоматически во время выполнения, что упрощает написание кода, однако требует больше внимания к потенциальным ошибкам в рантайме.
* **Большая стандартная библиотека:** Python предоставляет обширную стандартную библиотеку, включающую инструменты для работы с текстом, файлами, сетями, веб-сервисами и многими другими задачами.
* **Множество фреймворков и библиотек:** Python имеет тысячи сторонних библиотек для самых разных задач, от веб-разработки (Django, Flask) до машинного обучения (TensorFlow, PyTorch) и анализа данных (Pandas, NumPy).
* **Кроссплатформенность:** Программы, написанные на Python, могут запускаться на различных операционных системах без необходимости внесения изменений в код.

Основные принципы работы с Python:

* **Динамическая типизация:** Python использует динамическую типизацию, что позволяет переменным менять типы в процессе выполнения программы. Это упрощает разработку, но увеличивает вероятность ошибок, которые обнаруживаются только во время выполнения.
* **Поддержка парадигм программирования:** Python поддерживает как объектно-ориентированное, так и процедурное программирование, что позволяет выбрать подходящий стиль для каждой задачи. Также в Python есть поддержка функционального программирования с использованием лямбда-функций и итераторов.
* **Конкурентность и параллелизм:** Python предлагает несколько способов работы с многопоточностью и параллелизмом, включая модули threading, multiprocessing и asyncio, что делает его подходящим для создания приложений с высокой степенью конкурентности.
* **Интерпретируемый язык:** Python является интерпретируемым языком, что означает, что программы на Python выполняются строчка за строчкой без необходимости предварительной компиляции. Это ускоряет процесс разработки и тестирования.
* **Универсальность:** Python может использоваться для разработки самых разных типов приложений: от простых скриптов автоматизации до сложных веб-приложений и систем искусственного интеллекта.

Python — это мощный и универсальный язык программирования, который сочетает в себе простоту разработки с возможностью создавать высокофункциональные приложения для различных сфер, будь то наука о данных, разработка веб-приложений, автоматизация задач или искусственный интеллект.

**1.2 Библиотека tkinter**

tkinter — это стандартная библиотека Python для создания графических пользовательских интерфейсов (GUI). Она позволяет разрабатывать кроссплатформенные настольные приложения, которые могут работать на Windows, macOS и Linux, используя один и тот же код. tkinter интегрирован в стандартную библиотеку Python, что делает его доступным для всех пользователей Python без необходимости установки дополнительных зависимостей.

**Преимущества tkinter:**

– **Кроссплатформенность**: tkinter позволяет создавать приложения, которые работают на различных операционных системах, включая Windows, macOS и Linux, используя общий код. Это значительно упрощает процесс разработки и тестирования.

– **Интеграция с Python**: поскольку tkinter является частью стандартной библиотеки Python, его использование не требует установки сторонних пакетов. Это делает его идеальным для быстрого прототипирования и разработки простых приложений.

– **Легкость использования**: tkinter предоставляет разработчикам набор готовых к использованию компонентов, таких как кнопки, метки, текстовые поля и выпадающие меню. Это упрощает процесс создания графического интерфейса для приложений.

– **Гибкость в настройке интерфейса**: компоненты tkinter могут быть легко настроены в зависимости от потребностей приложения. Можно создавать как простые, так и сложные интерфейсы с помощью комбинаций стандартных виджетов.

– **Сообщество и ресурсы**: tkinter имеет большое сообщество разработчиков, а также множество учебных ресурсов, примеров и документации, что делает его легким для изучения и применения.

**Основные принципы работы с tkinter:**

– **Компоненты**: в tkinter приложения строятся с помощью виджетов (компонентов). Виджеты включают кнопки, текстовые поля, метки, фреймы и другие элементы интерфейса. Виджеты могут быть объединены в более сложные структуры для создания полноценных приложений.

– **Событийно-ориентированная модель**: tkinter использует событийную модель для управления взаимодействиями с пользователем. Каждое действие, такое как нажатие кнопки или ввод текста, генерирует событие, которое можно обработать с помощью соответствующих обработчиков.

– **Менеджеры компоновки**: tkinter предлагает три различных менеджера компоновки (pack, grid и place), которые позволяют гибко управлять расположением виджетов в окне приложения.

– **Адаптивный интерфейс**: несмотря на свою простоту, tkinter поддерживает создание адаптивных интерфейсов, которые корректно отображаются на различных размерах экрана и устройствах.

– **Быстрое прототипирование**: благодаря своей простоте и интеграции с Python, tkinter отлично подходит для быстрого создания прототипов, позволяя разработчикам сосредоточиться на логике приложения, не тратя много времени на настройку графического интерфейса.

**1.3 Библиотека SQLAlchemy**

SQLAlchemy — это популярная библиотека для работы с базами данных в Python, которая предоставляет гибкий и мощный инструмент для взаимодействия с различными системами управления базами данных (СУБД). SQLAlchemy объединяет два основных подхода к работе с базами данных: работу на уровне SQL-запросов (Core) и объектно-реляционное отображение (ORM), что делает ее универсальным решением для разных типов приложений. SQLAlchemy поддерживает множество СУБД, включая SQLite, PostgreSQL, MySQL, Oracle и Microsoft SQL Server, и позволяет разработчикам абстрагироваться от конкретного SQL-синтаксиса, обеспечивая кросс-СУБД совместимость.

Преимущества SQLAlchemy:

* **Объектно-реляционное отображение (ORM):** SQLAlchemy ORM позволяет работать с базами данных через Python-объекты, что упрощает взаимодействие с данными и делает код более читаемым и удобным для поддержки. Это снижает потребность в написании сложных SQL-запросов.
* **Гибкость и мощь SQLAlchemy Core:** для тех, кто предпочитает работу с «чистым» SQL, SQLAlchemy предоставляет Core API, который позволяет создавать и выполнять SQL-запросы вручную, сохраняя контроль над каждым аспектом работы с базой данных.
* **Кросс-СУБД совместимость:** SQLAlchemy абстрагирует работу с разными СУБД, позволяя разработчикам менять базы данных без необходимости переписывать запросы и миграции.
* **Миграции и управление схемами:** с помощью расширения Alembic, SQLAlchemy поддерживает управление схемой базы данных и миграции, что упрощает работу с изменениями в структуре данных.
* **Отложенная загрузка и связи:** SQLAlchemy поддерживает сложные связи между таблицами (один к одному, один ко многим, многие ко многим) и предлагает возможность отложенной загрузки данных, что позволяет оптимизировать работу с большими наборами данных.
* **Расширяемость и интеграция:** SQLAlchemy легко интегрируется с другими библиотеками и фреймворками, такими как Flask и Django, и поддерживает настраиваемые плагины и расширения для более сложных решений.

Основные принципы работы с SQLAlchemy:

* **Модели и таблицы:** при использовании ORM разработчики создают Python-классы (модели), которые отображаются на таблицы в базе данных. Каждый экземпляр класса представляет строку в соответствующей таблице, а атрибуты класса — это столбцы таблицы.
* **Запросы:** SQLAlchemy ORM позволяет писать запросы через Python-код, используя объектные методы для фильтрации, сортировки и объединения данных. Запросы создаются через сессии, которые представляют транзакции.
* **Сессии и транзакции:** SQLAlchemy использует сессии для управления транзакциями с базой данных. Все запросы и изменения в базе данных выполняются через сессии, что обеспечивает контроль за выполнением операций.
* **Миграции с Alembic:** для управления изменениями в базе данных, SQLAlchemy поддерживает Alembic, который позволяет создавать и применять миграции — изменения в структуре базы данных, такие как добавление новых таблиц или изменение столбцов.
* **Связи и каскады:** SQLAlchemy поддерживает установление связей между таблицами через внешние ключи и предоставляет механизмы управления каскадными операциями (например, каскадное удаление связанных записей).

SQLAlchemy — это мощный и гибкий инструмент для работы с базами данных в Python, который подходит как для простых приложений, так и для крупных проектов с большими объемами данных и сложной логикой.

**1.4** **Среда разработки Pycharm**

PyCharm — это интегрированная среда разработки (IDE) для Python, созданная компанией JetBrains. PyCharm является одним из самых мощных и популярных инструментов для разработки приложений на Python, включая использование библиотеки tkinter для создания графических интерфейсов.

PyCharm доступен в двух версиях: бесплатной (Community Edition) и коммерческой (Professional Edition). Обе версии предоставляют обширный набор инструментов для разработки Python-приложений, а также возможности для работы с другими языками и технологиями.

Преимущества Pycharm при работе с Python:

* **Поддержка графических интерфейсов:** PyCharm предоставляет интеграцию с библиотекой tkinter, что позволяет удобно разрабатывать графические интерфейсы (GUI). IDE поддерживает автодополнение, подсветку синтаксиса и другие функции для облегчения работы с кодом, связанным с интерфейсом.
* **Расширенные инструменты отладки:** PyCharm предлагает мощные инструменты отладки, которые упрощают процесс поиска и исправления ошибок в Python-приложениях с графическим интерфейсом. С помощью встроенного отладчика можно легко отслеживать выполнение программы, анализировать переменные и проверять выполнение кода.
* **Автодополнение и подсказки:** PyCharm предоставляет интеллектуальные подсказки и автодополнение для Python и библиотек, включая tkinter. Это помогает ускорить процесс написания кода и предотвратить ошибки на ранних стадиях.
* **Интеграция с Git:** PyCharm имеет встроенную поддержку системы контроля версий Git, что позволяет легко управлять версиями проекта, отслеживать изменения в коде и эффективно работать в команде.
* **Множество расширений и плагинов:** PyCharm поддерживает большое количество плагинов, которые можно использовать для интеграции с различными технологиями и инструментами, включая библиотеки для работы с графическими интерфейсами, тестирование и другие важные аспекты разработки.

**2 Практическая часть**

**2.1 Технические задание**

Разработать приложение, содержащее в себе следующие функции:

* Добавление записей
* Отображение данных
* Поиск записей
* Обновление данных в таблице
* Удаление записей
* Экспорт одной из таблиц на выбор в .xml файл
* Чтение .xml файла и представление его содержания на экране в виде таблицы

Требования:

– Приложение должно поддерживать функциональность для редактирования всех таблиц: добавления, изменения и удаления данных;

– При изменении данных должна выполняться проверка на правильность вводимых значений;

– Все экраны, включая меню и таблицы, должны иметь кнопки для навигации, позволяющие возвращаться к предыдущему экрану;

– Обработка исключений должна быть реализована для всех операций, связанных с редактированием данных;

– Типы полей в базе данных должны соответствовать назначению поля, например, тип поля «Дата договора» — структура дата/время, тип поля «Серия паспорта» — число и т.д.;

– Дизайн может быть выполнен в свободной форме в соответствии с предпочтениями.

**2.2 Анализ поставленной задачи и выбор пути её решения**

На данном этапе был проведен анализ поставленного технического задания, результатом которого стал следующий план работ:

1. Построить модели таблиц, используя SQLAlchemy, согласно заданию.

2. Подключить базу данных и реализовать все функции, связанные с ней с помощью ORM.

3. Реализовать графический интерфейс для каждой из вкладок соответствующих функций, используя библиотеку tkinter

**2.3 Создание моделей таблиц**

Для каждой из таблиц был создан отдельный класс, который содержит в себе все названия столбцов и их параметры. В зависимости от столбца использовались разные типы данных. Так, например, для имени, фамилии и отчества используется тип данных string, а для кода преподавателя используется int. Так же реализована валидация данных: в случае если пользователь вводит некорректные данные программа выдаёт ошибку и просит пользователи ввести данные нужного типа.

**2.4 Добавление записи**

Вкладка добавления записи в базу данных содержит в себе выбор таблиц, при совершении которого обновляется количество полей, в зависимости от количества столбцов в выбранной таблице. Функция для обновления полей:

def update\_add\_fields(self, event=None):

for widget in self.add\_fields\_frame.winfo\_children():

widget.destroy()

table\_display = self.add\_table\_var.get()

table\_name = next(key for key, value in table\_display\_names.items() if value == table\_display)

self.add\_entries = {}

mapping = column\_mappings[table\_name]

for i, (\_, column) in enumerate(mapping.items()):

ttk.Label(self.add\_fields\_frame, text=f"{column}:").grid(row=i, column=0, padx=5, pady=5)

entry = ttk.Entry(self.add\_fields\_frame, width=50)

entry.grid(row=i, column=1, padx=5, pady=5)

self.add\_entries[column] = entry

Вид вкладки представлен на рисунке 1

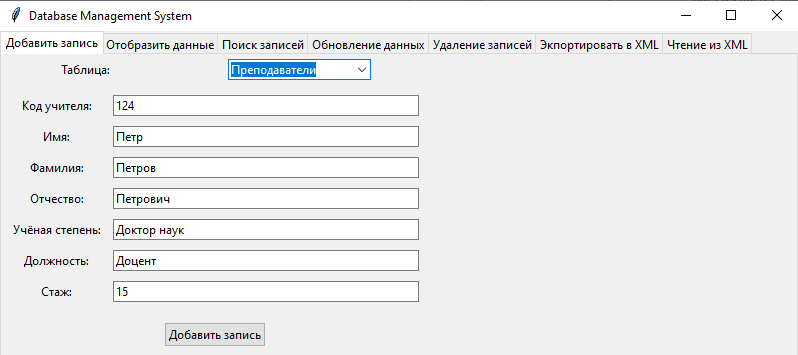


Рисунок 1 – Добавление записи

**2.5 Отображение данных**

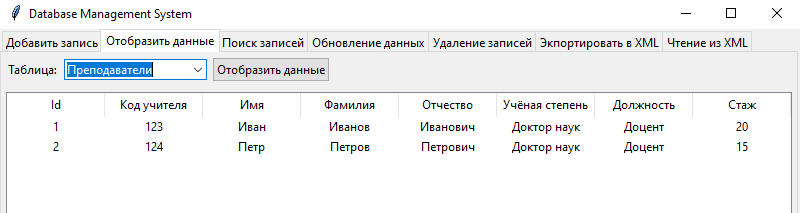
Отображение данных реализовано с помощью виджета TreeView для отображения данных в виде таблицы. Вид вкладки представлен на рисунке 2 

Рисунок 2 – Отображение данных

**2.6 Поиск записей**

Поиск произоводится по всей таблице, при совпадении выводится строка/строки из таблицы. Отображение данных в виде таблицы реализовано через виджет TreeView. Исходный код функции поиска:

def search\_database(table\_name, search\_term):

if table\_name not in table\_map:

raise ValueError(f"Неверное имя таблицы")

model = table\_map[table\_name]

try:

query = session.query(model)

filters = []

for column in model.\_\_table\_\_.columns:

if isinstance(column.type, String):

filters.append(column.ilike(f'%{search\_term}%'))

else:

filters.append(column == search\_term)

query = query.filter(or\_(\*filters))

results = query.all()

readable\_results = []

for result in results:

result\_dict = {column.name: getattr(result, column.name) for column in model.\_\_table\_\_.columns}

readable\_results.append(result\_dict)

return readable\_results

except Exception as e:

raise ValueError(f"Произошла ошибка во время поиска: {str(e)}")

finally:

session.close()

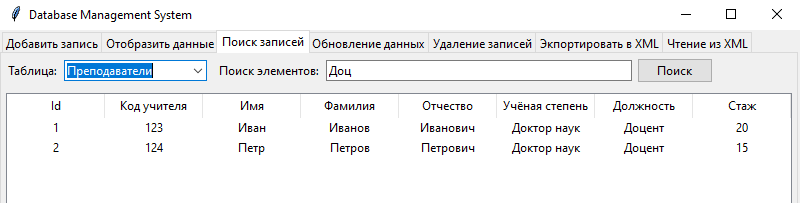
Вид представлен на рисунке 3.

Рисунок 3 – Поиск записей

**2.7 Обновление данных**

При обновлении данных, так же как и при добавлении данных реализована функция обновления полей, в зависимости от выбора таблицы. Вид представлен на рисунке 4.

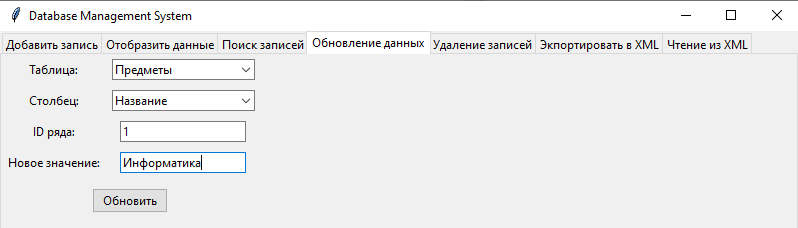


Рисунок 4 – Обновление данных

**2.8 Удаление записей**

Интерфейс удаления записей реализован так же как и интерфейс обновления данных с отличием в количетсве комбобоксов. Вид представлен на рисунке 5.

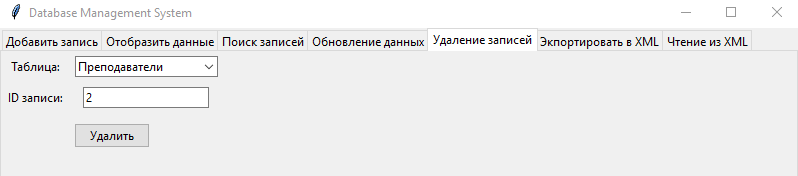


Рисунок 5 – Удаление записей

**2.9 Экспорт в XML**

Функция экспорта таблицы в xml файл реализована с помощью встроенных в Python библиотек xml и os. Исходный код функции экспорта:

def export\_database\_to\_xml(output\_directory, selected\_table): # экспорт данных в XML

inspector = inspect(engine)

try:

os.makedirs(output\_directory, exist\_ok=True)

all\_tables = inspector.get\_table\_names()

if selected\_table not in all\_tables:

return f"Выбранная таблица {selected\_table} не существует в базе данных."

file\_path = os.path.join(output\_directory, f"{selected\_table}.xml")

root = ET.Element(selected\_table)

columns = [col['name'] for col in inspector.get\_columns(selected\_table)]

result = session.execute(text(f"SELECT \* FROM {selected\_table}"))

for row in result:

record = ET.SubElement(root, 'record')

for col, value in zip(columns, row):

field = ET.SubElement(record, col)

field.text = str(value) if value is not None else ''

xml\_str = minidom.parseString(ET.tostring(root)).toprettyxml(indent=" ")

with open(file\_path, 'w', encoding='utf-8') as f:

f.write(xml\_str)

return f"Данные из таблицы {selected\_table} записаны в файл {file\_path}"

except Exception as e:

return f"Произошла ошибка при экспорте данных: {str(e)}"

finally:

session.close()

Вид представлен на рисунке 6.

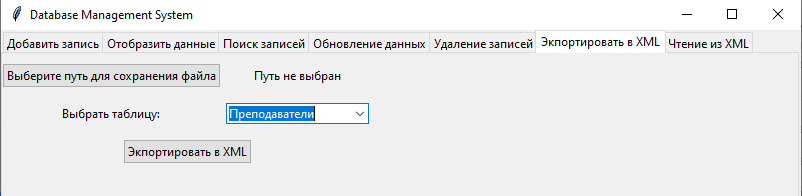


Рисунок 6 – Экспорт в XML

**2.10 Чтение XML файла**

Обработка xml файла производится через библиотеку xml, отображение данных реализовано с помощью виджета TreeView. Вид представлен на рисунке 7.

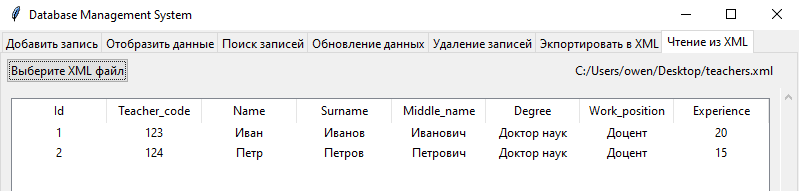


Рисунок 7 – Чтение из XML

**ЗАКЛЮЧЕНИЕ**

В ходе прохождения учебной практики был приобретен опыт разработки кроссплатформенных приложений с использованием библиотеки tkinter языка программирования Python.

В ходе разработки приложения был проведен анализ, активно использовались средства тестирования и отладки.

Работа над проектом поспособствовала улучшению навыков планирования рабочего дня, а также эффективного распределения задач и ресурсов.

Прохождение практики позволило увидеть, какие реальные задачи стоят перед разработчиками приложений в настоящее время.

Техническое задание было реализовано в полном объеме. Требования к заданию были учтены.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Документация Python [Электронный ресурс]. URL: <https://docs.python.org/3/> (дата обращения 26.06.2024).

2. Документация tkinter [Электронный ресурс]. URL: https://docs.python.org/3/library/tkinter.html (дата обращения 26.06.2024).

3. Документация SQLAlchemy [Электронный ресурс]. URL: https://docs.sqlalchemy.org/en/20/ (дата обращения 26.06.2024).

4. Документация PyCharm [Электронный ресурс]. URL: https://www.jetbrains.com/help/pycharm/getting-started.html (дата обращения 26.06.2024).

5. Гост 7.32-2017 [Электронный ресурс]. URL: https://cs.msu.ru/sites/cmc/files/docs/2021-11gost\_7.32-2017.pdf 　(дата обращения 26.06.2024).

**МИНОБРНАУКИ РОССИИ**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«Чувашский государственный университет имени И.Н. Ульянова»**

**(ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»)**

**Факультет информатики и вычислительной техники**

**Кафедра вычислительной техники**

РАБОЧИЙ ГРАФИК (ПЛАН)

ПРОВЕДЕНИЯ УЧЕБНОЙ ПРАКТИКИ

(ОЗНАКОМИТЕЛЬНОЙ ПРАКТИКИ)

на базе ***кафедры вычислительной техники факультета информатики и вычислительной техники ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»***

**Захарова Владимира Владимировича, ИВТ-42-23**

*09.03.01 Информатика и вычислительная техника, направленность (профиль) Программное обеспечение средств вычислительной техники и автоматизированных систем*

| №  п/п | Разделы (этапы)  практики | Виды работ на практике, включая самостоятельную работу студентов | Трудоемкость,  час | Дата |
| --- | --- | --- | --- | --- |
| 1. | Организация практики, подготовительный этап | Оформление на практику, прохождение инструктажа по охране труда, технике безопасности, пожарной безопасности, а также ознакомление с правилами внутреннего трудового распорядка организации, предоставляющей место для прохождения практики | 9 | 26.06.2024 |
| 2. | Производственный этап | Обучение и работа на рабочем месте в качестве стажера-практиканта в соответствии с индивидуальным заданием | 72 | 26.06.2024-  05.07.2024 |
| 3. | Подготовка отчета | Сбор, обработка и систематизация фактического и литературного материала | 24 | 06.07.2024-  09.07.2024 |
| 4. | Защита отчета | Получение отзыва на рабочем месте  Публичная защита отчета | 3 | 09.07.2024 |
|  | ИТОГО |  | 108 |  |

Руководитель практики от кафедры \_\_\_\_\_\_\_\_\_\_\_/ Юрьева Е.В.

Дата выдачи графика 26 июня 2024 года

Согласовано:

Руководитель практики от профильной организации \_\_\_\_\_\_\_\_\_/ Юрьева Е.В.

Дата согласования 26 июня 2024 года

ДНЕВНИК

ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ (ОЗНАКОМИТНЛЬНОЙ ПРАКТИКИ)

на базе ***кафедры вычислительной техники факультета информатики и вычислительной техники ФГБОУ ВО «ЧГУ им. И.Н. Ульянова»***

**Захарова Владимира Владимировича, ИВТ-42-23**

*09.03.01 Информатика и вычислительная техника, направленность (профиль) Программное обеспечение средств вычислительной техники и автоматизированных систем*

| №  п/п | Разделы (этапы)  практики | Виды работ на практике, включая самостоятельную работу студентов | Трудоемкость,  час | Дата |
| --- | --- | --- | --- | --- |
| 1. | Организация  практики, подготовительный  этап | Оформление на практику, прохождение инструктажа по охране труда, технике безопасности, пожарной безопасности, а также ознакомление с правилами внутреннего трудового распорядка организации, предоставляющей место для прохождения практики | 9 | 26.06.2024 |
| 2. | Производственный  этап | Обучение и работа на рабочем месте в качестве стажера-практиканта в соответствии с индивидуальным заданием: | 72 |  |
| Анализ задания, изучение предметной области, просмотр учебных материалов | 9 | 26.06.2024 |
| Установка необходимого ПО и дополнительных библиотек, создание проекта | 9 | 27.06.2024 |
| Организация архитектуры, навигации между вкладками | 9 | 28.06.2024 |
| Создание моделей для базы данных, подключение к ней | 9 | 29.06.2024 |
| Реализация функций для работы с базой данных | 9 | 01.07.2024 |
| Внедрение механизмов обработки ошибок | 9 | 02.07.2024 |
| Разработка интерфейса приложения | 9 | 03.07.2024 |
| Тест приложения с различными данными и всеми возможными ошибками | 9 | 04.07.2024 |
| Доработка дизайна приложения | 9 | 05.07.2024 |
| 3. | Подготовка отчета | Сбор, обработка и систематизация фактического и литературного материала | 24 | 06.07.2024-  09.07.2023 |
| 4. | Защита отчета | Получение отзыва на рабочем месте  Публичная защита отчета | 3 | 09.07.2024 |
|  | ИТОГО |  | 108 |  |

Студент практикант \_\_\_\_\_\_\_\_\_\_\_\_/Захаров В.В.

Руководитель практики от профильной организации \_\_\_\_\_\_\_\_\_/ Юрьева Е.В.

Дата составления 09 июля 2024 года

ОТЗЫВ РУКОВОДИТЕЛЯ

УЧЕБНОЙ ПРАКТИКИ

(ПРАКТИКИ ПО ПОЛУЧЕНИЮ ПЕРВИЧНЫХ ПРОФЕССИОНАЛЬНЫХ УМЕНИЙ И НАВЫКОВ, В ТОМ ЧИСЛЕ ПЕРВИЧНЫХ УМЕНИЙ И НАВЫКОВ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ ДЕЯТЕЛЬНОСТИ)

Захаров Владимир Владимирович, ИВТ-42-23

Студент проходил практику кафедра вычислительной техники с «26» июня 2024 по 09» июля 2024.

За время прохождения практики студент Захаров Владимир Владимирович:

− ознакомился с рабочим местом студента-практиканта, правилами внутреннего распорядка организации,

− изучил и подтвердил знание требований техники безопасности, охраны труда, противопожарных мероприятий при работе с электронно-вычислительной техникой;

− ознакомился с составом стандартных библиотек системы программирования;

− изучил организацию библиотеки tkinter;

− получил навыки создания десктопного приложения с использованием tkinter;

– изучил возможности библиотеки tkinter;

− выполнил поставленные задачи.

– составил отчет по практике.

По результатам практики Захаров Владимир Владимирович,

продемонстрировал \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ уровень\* теоретических знаний по информатике и программированию, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ уровень самостоятельности и исполнительности, получил первичные профессиональные умения и навыки в области разработки программных проектов, в том числе навыки научно-исследовательской деятельности и заслуживает оценки \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики от кафедры \_\_\_\_\_\_\_\_\_\_\_/ Юрьева Е.В.

Дата «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_ г.

\* высокий/невысокий/низкий