МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**Факультет информационных технологий и робототехники**

Кафедра программного обеспечения информационных систем и технологий

**Отчёт по лабораторной работе № 3**

по дисциплине: “Разработка требований”

на тему: “ **Картотека агенства недвижимости**”

Вариант 10

|  |  |
| --- | --- |
| Выполнил:  Приняла: | Студент группы 10701323 Шаплавский Н.С.  Ст. преподаватель Барышев А.А. |

Минск – 2024

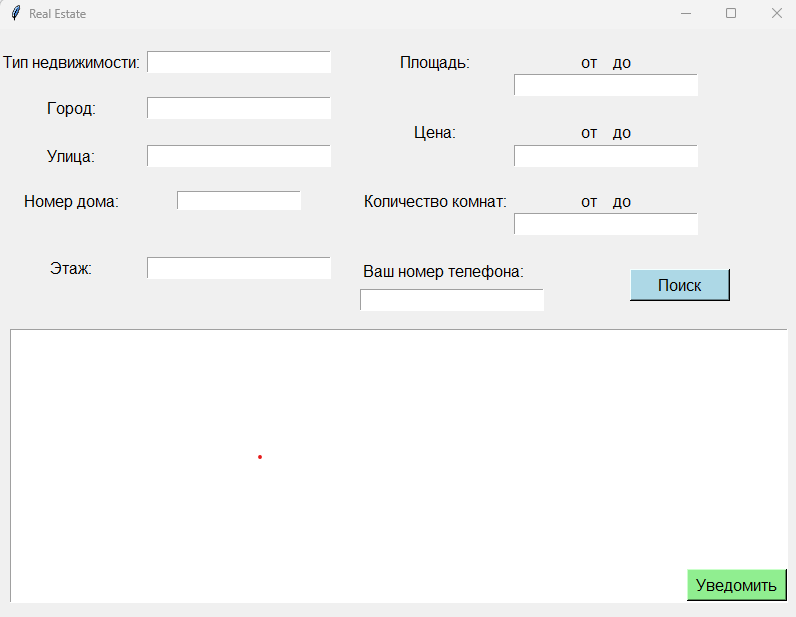
**Цель работы**

Разработать программный продукт в соответствии с заданным вариантом.

**Код программы:**

from tkinter import \*  
from modules.db\_manager import DBManager  
  
  
class App:  
 def \_\_init\_\_(self):  
 self.manager = DBManager()  
 self.manager.create\_table()  
 self.manager.create\_call\_table()  
 self.root = Tk()  
 self.root.title("Real Estate")  
 self.root.geometry("800x600")  
 self.font = ("Arial", 12)  
  
  
  
 self.create\_widgets()  
 self.root.mainloop()  
  
 def create\_widgets(self):  
  
 Label().grid(row=0, column=0)  
 Label().grid(row=2, column=0)  
 Label().grid(row=4, column=0)  
 Label().grid(row=6, column=0)  
 Label().grid(row=8, column=0)  
 Label(text=" ").grid(row=0, column=2)  
 Label().grid(row=9, column=0)  
 Label().grid(row=12, column=0)  
  
 Label(self.root, text="Тип недвижимости: ", font=self.font).grid(row=1, column=0)  
 self.type\_entry = Entry(self.root, font=self.font)  
 self.type\_entry.grid(row=1, column=1)  
  
 Label(self.root, text="Город: ", font=self.font).grid(row=3, column=0)  
 self.city\_entry = Entry(self.root, font=self.font)  
 self.city\_entry.grid(row=3, column=1)  
  
 Label(self.root, text="Улица: ", font=self.font).grid(row=5, column=0)  
 self.street\_entry = Entry(self.root, font=self.font)  
 self.street\_entry.grid(row=5, column=1)  
  
 Label(self.root, text="Номер дома: ", font=self.font).grid(row=7, column=0)  
 self.number\_entry = Entry(self.root)  
 self.number\_entry.grid(row=7, column=1)  
  
  
  
 Label(self.root, text="Площадь: ", font=self.font).grid(row=1, column=3)  
 Label(self.root, text= "от до", font=self.font).grid(row=1, column=4)  
 self.square\_entry = Entry(self.root, font=self.font)  
 self.square\_entry.grid(row=2, column=4)  
  
 Label(self.root, text="Цена: ", font=self.font).grid(row=4, column=3)  
 Label(self.root, text= "от до", font=self.font).grid(row=4, column=4)  
 self.price\_entry = Entry(self.root, font=self.font)  
 self.price\_entry.grid(row=5, column=4)  
  
 Label(self.root, text="Количество комнат: ", font=self.font).grid(row=7, column=3)  
 Label(self.root, text= "от до", font=self.font).grid(row=7, column=4)  
 self.rooms\_entry = Entry(self.root, font=self.font)  
 self.rooms\_entry.grid(row=8, column=4)  
  
 Label(self.root, text="Этаж: ", font=self.font).grid(row=10, column=0)  
 self.floor\_entry = Entry(self.root, font=self.font)  
 self.floor\_entry.grid(row=10, column=1)  
  
 Button(  
 self.root,  
 text="Поиск",  
 font=self.font,  
 command=self.search,  
 width=10,  
 height=1,  
 bg="lightblue",  
 ).place(x=630, y=240)  
  
 Label(text="Ваш номер телефона: ", font=self.font).place(x=360, y=230)  
 self.phone\_entry = Entry(self.root, font=self.font)  
 self.phone\_entry.place(x=360, y=260)  
  
 self.result\_text = Text(self.root, height=15, width=86, font=self.font)  
 self.result\_text.place(x=10, y=300)  
  
 Button(  
 self.root,  
 text="Уведомить",  
 font=self.font,  
 command= self.compl\_call,  
 width=10,  
 height=1,  
 bg="lightgreen",  
 ).place(x=687, y=540)  
  
 def compl\_call(self):  
 phone = self.phone\_entry.get()  
 estate\_type = self.type\_entry.get()  
 city = self.city\_entry.get()  
 street = self.street\_entry.get()  
 number = self.number\_entry.get()  
 square = self.square\_entry.get()  
 price = self.price\_entry.get()  
 rooms = self.rooms\_entry.get()  
 floor = self.floor\_entry.get()  
 self.manager.add\_call(estate\_type, city, street, number, square, price, rooms, floor, phone)  
 def search(self):  
 estate\_type = self.type\_entry.get()  
 city = self.city\_entry.get()  
 street = self.street\_entry.get()  
 number = self.number\_entry.get()  
 square = self.square\_entry.get()  
 price = self.price\_entry.get()  
 rooms = self.rooms\_entry.get()  
 floor = self.floor\_entry.get()  
  
 result = self.manager.find\_estate(estate\_type, city, street, number, square, price, rooms, floor)  
  
 self.result\_text.delete(1.0, END)  
  
 if len(result) == 0:  
 self.result\_text.insert(1.0, "Ничего не найдено \nНажмите на кнопку 'Уведомить' чтобы уведомить вас о появлении подходящего варианта")  
 else:  
 print(result)  
 self.result\_text.insert(END, f"Найдено {len(result)} вариантов\n\n\n")  
  
 for i, row in enumerate(result):  
  
 text = "-" \* 50 + "\n"  
 text += f"Недвижимость №{i+1}"  
 text += f"\n Тип недвижимости: {row[1]} \n Город: {row[2]} \n Улица: {row[3]} \n Номер дома: {row[4]} \n Площадь: {row[5]} \n Цена: {row[6]} \n Количество комнат: {row[7]} \n Этаж: {row[8]} \n Комментарий: {row[9]} \n"  
 text += "\n\n\n"  
  
  
 self.result\_text.insert(END, text)  
App()

**Интерфейс пользоателя:**

****

**Руководство пользователя**

**Программный модуль «Картотека абонентов недвижимости»**

**1. Назначение программы**

Программный модуль «Картотека абонентов недвижимости» предназначен для сотрудников службы поддержки и бухгалтерии. Программа позволяет искать информацию об абонентах по их фамилии и добавлять новые данные. Информация об абонентах хранится в базе данных, которую можно редактировать через интерфейс программы.

**2. Описание интерфейса**

Главное окно (поиск данных):

- Позволяет искать информацию об абонентах по фамилии. Пользователь может выбрать категорию данных (данные владельца, номер телефона, задолженности) и найти соответствующую информацию.

**3. Использование программы**

- В главном окне программы введите фамилию или номер телефона.

- Нажмите кнопку «Поиск».

- Программа выполнит поиск информации об абоненте и отобразит её.

- Для добавления абонента заполните необходимые поля и нажмите кнопку «Сохранить».

**4. Особенности программы**

- Программа хранит данные в базе данных.

- Поиск выполняется на основе фамилии абонента. Если абонент найден, его данные загружаются для отображения в программе.

**Руководство системного программиста**

**Проверка функционирования программы**

**1. Проверка исходных данных:**

- Убедитесь, что файлы данных, используемые программой, присутствуют в указанной директории и имеют корректную структуру. Программа использует файл:

- `data.db` — для хранения данных об абонентах (ФИО, номер телефона, задолженности).

**2. Проверка работоспособности интерфейса**:

- Для проверки работы пользовательского интерфейса запустите программу и убедитесь, что основные функции доступны:

- Ввод данных для поиска информации.

- Добавление новых абонентов.

- Получение информации из базы данных в зависимости от введённых данных.

**3. Проверка обработки ошибок:**

- Программа должна корректно обрабатывать следующие ситуации:

- Некорректный ввод или отсутствие данных.

- Отсутствие записей в базе данных.

- Попытка поиска несуществующего абонента. В этих случаях программа должна выдавать соответствующие сообщения об ошибках.

**Руководство программиста**

**1. Основная информация**

- Данное руководство содержит информацию для эксплуатации программного обеспечения «Картотека абонентов недвижимости». Модуль предназначен для хранения и управления данными об абонентах, а также для предоставления сведений службе поддержки и бухгалтерии.

**2. Основные компоненты программного модуля**

1. Файлы данных:

- Программа использует базу данных для хранения и управления данными:

- `data.db` — хранит информацию об абонентах.

2. Основные формы программы:

- Программа содержит одну форму:

- `main.py` — главная форма для поиска и отображения информации.

**Руководство оператора**

1. Описание интерфейса программы

1. Главное окно (main.py):

- Поля для ввода ФИО и номера телефона: сюда оператор вводит фамилию или номер абонента для поиска данных.

- Кнопка «Поиск»: инициирует процесс поиска данных по введённой информации.

- Поле результата: отображает найденную информацию или сообщение о том, что данные не найдены.

**2. Процедуры взаимодействия с программой**

1. Поиск данных об абоненте:

- Запустите программу.

- В главном окне введите фамилию или номер телефона абонента.

- Нажмите кнопку «Поиск».

- Программа отобразит результат в поле «Результат».

**3. Обработка ошибок и рекомендации**

- Отсутствие данных: если программа не может найти данные по введённому абоненту, убедитесь, что информация введена корректно. Если ошибка повторяется, обратитесь к администратору для проверки целостности файлов данных.

- Добавление абонента: если кнопка «Сохранить» неактивна или данные не добавляются, проверьте, что все обязательные поля заполнены.

**Результат работы программы:**

**Поиск недвижимости**