# Список используемых библиотек для настольного приложения с графическим интерфейсом, установкой и запуска на компьютере конечного пользователя. (СУБД).

Разработка проектов осуществляется с использованием языка программирования Python.

Библиотеки:

1. **PySide6:**
   * Описание: PySide6 представляет собой оболочку/интерфейс для создания графического пользовательского интерфейса.
   * Лицензия: LGPL (Lesser General Public License) – свободная лицензия, обеспечивающая гибкость в использовании библиотеки в коммерческих проектах.
2. **Peewee:**
   * Описание: Peewee – это ORM (Object-Relational Mapping) для удобной работы с базами данных.
   * Лицензия: MIT – открытая лицензия, обеспечивающая свободу использования, модификации и распространения кода.
3. **sqlite3:**

* Описание: sqlite3 является встроенным модулем Python, предоставляющим легкую и быструю базу данных SQLite. Он идеально подходит для небольших и средних проектов, где требуется легкая и портативная база данных.
* Лицензия: LGPL – это та же лицензия, что и у psycopg2, что обеспечивает гибкость в использовании в различных проектах.

1. **pandas:**
   * Описание: pandas – библиотека для обработки и анализа данных.
   * Лицензия: BSD-3-Clause License – открытая лицензия с минимальными ограничениями.
2. **statistics:**
   * Описание: statistics – встроенная библиотека Python для выполнения статистических операций.
   * Лицензия: Python Software Foundation License – лицензия, обеспечивающая свободу использования в Python-проектах.
3. **openpyxl:**
   * Описание: openpyxl – библиотека для работы с файлами Excel (xlsx).
   * Лицензия: MIT License – открытая лицензия, позволяющая свободное использование и модификацию кода.
4. **numpy:**
   * Описание: numpy – библиотека для работы с массивами и матрицами чисел.
   * Лицензия: BSD License – открытая лицензия, предоставляющая свободу использования и распространения.

# Структура проекта

AddUserDialog.py: Файл, содержащий класс диалогового окна для добавления нового пользователя.

AllDbScroller.py:это файл, содержащий класс для прокрутки содержимого базы данных.

auth.py: Содержит логику аутентификации пользователей.

authWindow.spec: Файл, связанный с настройками аутентификации.

BD.spec: это файл с описанием базы данных и настройками связанными с ней.

ChangeLogWindow.py: Файл, содержащий класс окна для отображения изменений.

ContractWidget.py: Виджет для отображения контрактов.

CurrencyWindow.py: Файл, содержащий класс окна для работы с валютами.

database\_initialized.marker: Маркер, указывающий, что база данных была инициализирована.

DBtest.py: Файл с тестами базы данных.

debug.py: Файл с отладочным кодом.

debugWindow.py: Окно для отображения отладочной информации.

EditUserDialog.py: Файл с классом диалогового окна для редактирования пользователя.

initialize\_db.py: Скрипт для инициализации базы данных.

insertPanel.py: Виджет для вставки панели.

insertPanelContract.py: Виджет для вставки панели контракта.

InsertWidgetContract.py: Виджет контракта.

InsertWidgetCurrency.py: Виджет валюты.

InsertWidgetNMCK.py: Виджет НМЦК.

InsertWidgetNMCK\_2.py: Виджет НМЦК 2 метод.

InsertWidgetNMCK\_3.py: Виджет НМЦК 3 метод.

InsertWidgetNMCK\_4.py: Виджет НМЦК 4 метод.

LoadCsv.py: Скрипт для загрузки данных из CSV.

MainWindow.py: Файл с классом главного окна.

models.py: Файл, содержащий описание моделей базы данных.

parserV3.py: Скрипт для парсинга данных.

start.py: Ваш главный файл, который запускает приложение.

statisticWidget.py: Виджет для отображения статистики.

test.db: Файл базы данных для тестов.

# Исходный код

Импорт библиотек и методов.

from PySide6.QtWidgets import QWidget, QVBoxLayout, QHBoxLayout, QLabel, QLineEdit, QPushButton, QMessageBox

from PySide6.QtGui import QIcon, QPixmap

from PySide6.QtCore import Qt

from auth import AuthManager

from models import UserLog

from initialize\_db import initialize\_database

from MainWindow import Ui\_MainWindow

from datetime import datetime

class AuthWindow(QWidget):

"""Класс для создания окна авторизации.

Атрибуты:

auth: Экземпляр класса AuthManager для управления аутентификацией пользователей.

username\_edit: QLineEdit для ввода имени пользователя.

password\_edit: QLineEdit для ввода пароля пользователя.

login\_button: QPushButton для запуска процесса аутентификации.

"""

def \_\_init\_\_(self):

"""Инициализация окна авторизации и компонентов GUI."""

super(AuthWindow, self).\_\_init\_\_()

# Установка параметров окна

self.setWindowTitle("Окно авторизации")

self.setGeometry(100, 100, 1000, 600)

# Инициализация объекта для аутентификации

self.auth = AuthManager()

# Инициализация базы данных

initialize\_database()

# Создание компонентов GUI

main\_layout = QVBoxLayout()

pics\_layout = QHBoxLayout()

form\_layout = QVBoxLayout()

form\_logPass = QVBoxLayout()

form\_layoutForLogPAs = QFormLayout()

label\_layout = QVBoxLayout()

# Добавление изображений

image\_label\_top\_left = QLabel()

pixmap = QPixmap("Pics/4.png")

pixmap = pixmap.scaledToWidth(50)

image\_label\_top\_left.setPixmap(pixmap)

image\_label\_top\_left.setAlignment(Qt.AlignLeft | Qt.AlignTop)

pics\_layout.addWidget(image\_label\_top\_left)

image\_label\_top\_right = QLabel()

pixmap = QPixmap("Pics/4.png")

pixmap = pixmap.scaledToWidth(50)

image\_label\_top\_right.setPixmap(pixmap)

image\_label\_top\_right.setAlignment(Qt.AlignRight | Qt.AlignTop)

pics\_layout.addWidget(image\_label\_top\_right)

main\_layout.addLayout(pics\_layout)

# Надпись

self.label = QLabel()

self.label.setMaximumWidth(400)

self.label.setWordWrap(True)

self.label.setText("БАЗА ДАННЫХ ...")

self.label.setAlignment(Qt.AlignCenter)

font = QFont()

font.setPointSize(14)

self.label.setFont(font)

self.username\_label = QLabel("Имя пользователя:")

self.username\_edit = QLineEdit("")

self.username\_edit.setPlaceholderText("Введите ваше имя пользователя")

self.username\_edit.setMaximumWidth(200)

lock\_icon = QIcon("Pics/icons8-пользователь-30.png")

self.username\_edit.addAction(lock\_icon, QLineEdit.LeadingPosition)

self.password\_label = QLabel("Пароль:")

self.password\_edit = QLineEdit()

self.password\_edit.setEchoMode(QLineEdit.Password)

self.password\_edit.setPlaceholderText("Введите ваш пароль")

self.password\_edit.setMaximumWidth(200)

self.password\_edit.setAlignment(Qt.AlignLeft)

self.login\_button = QPushButton("Войти")

self.login\_button.clicked.connect(self.authenticate)

self.login\_button.setFixedWidth(320)

lock\_icon = QIcon("Pics/icons8-пароль-30.png")

self.password\_edit.addAction(lock\_icon, QLineEdit.LeadingPosition)

login\_icon = QIcon("Pics/icons8-вход-в-систему,-в-кружке,-стрелка-вправо-30.png")

self.login\_button.setIcon(login\_icon)

# Добавление компонентов на форму

label\_layout.addWidget(self.label)

label\_layout.setAlignment(Qt.AlignCenter)

self.label.move(0, -100)

form\_layoutForLogPAs.addRow(self.username\_label, self.username\_edit)

form\_layoutForLogPAs.addRow(self.password\_label, self.password\_edit)

form\_layoutForLogPAs.setFormAlignment(Qt.AlignCenter)

form\_layout.addWidget(self.login\_button)

form\_layout.setAlignment(Qt.AlignCenter)

form\_logPass.addLayout(form\_layoutForLogPAs)

form\_logPass.addLayout(form\_layout)

main\_layout.addLayout(label\_layout)

main\_layout.addLayout(form\_logPass)

self.setLayout(main\_layout)

def authenticate(self):

"""Метод для аутентификации пользователя при нажатии кнопки 'Войти'."""

username = self.username\_edit.text()

password = self.password\_edit.text()

user = self.auth.authenticate(username, password)

if user:

QMessageBox.information(self, "Успех", "Вы успешно авторизировались!")

UserLog.create(username=username, login\_time=datetime.now())

ui = Ui\_MainWindow(username)

ui.show()

self.close()

else:

QMessageBox.warning(self, "Ошибка", "Ошибка входа")

if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication([])

auth\_window = AuthWindow()

auth\_window.show()

app.exec()