МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Донецкий национальный технический университет»

Факультет ИСП

Кафедра ПИ им Л.П.Фельдмана

Лабораторная работа № 4

на тему: «Создание самодокументирующегося кода»

по курсу: «Профессиональная практика программной инженерии»

Проверил:

асс. каф. ПИ им. Л.П.Фельдмана Филипишин Д.А.

Выполнил:

ст. гр. ПИ-21В

Павловский В.В.

Донецк-2024

Цель работы – научиться добавлять в программный код специальным образом оформление докблок-комментарии, для последующей автоматической генерации API reference, а также познакомиться с форматом оформления документации DocBook.

**Вариант 29**. Эмулятор консольных игр ((одной из dendy, sega megadrive, ps1/2/3/4, dreamcast), например, ePSXe, Gens, Snes);

1. Регистрация на GitHub (рис. 1). Аккаунт: [WunderWaffelSS](https://github.com/WunderWaffelSS). Репозиторий: [RetroEmuX](https://github.com/WunderWaffelSS/RetroEmuX).

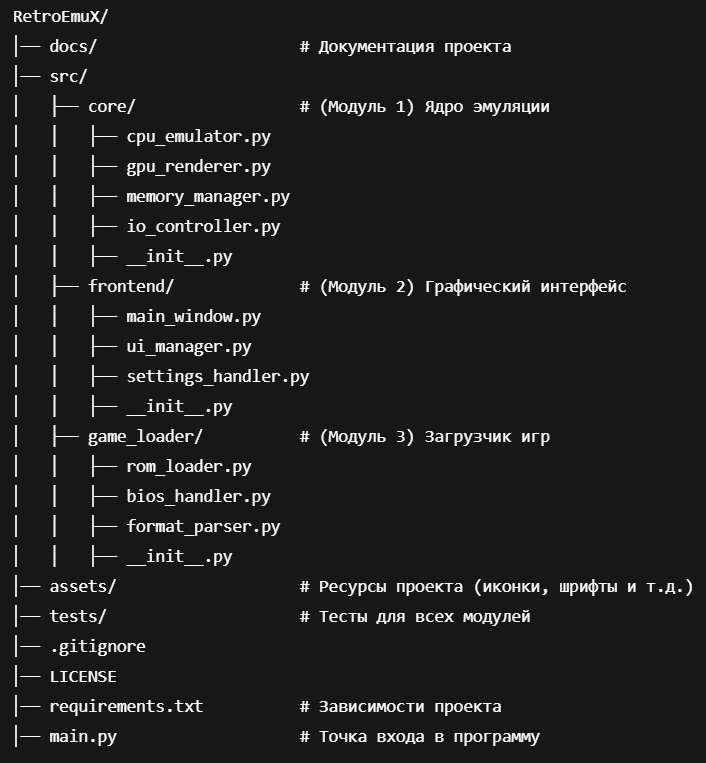


Рисунок 1 – Структура проекта

import tkinter as tk  
from tkinter import filedialog, messagebox  
from frontend.ui\_manager import UIManager  
from game\_loader.rom\_loader import ROMLoader  
  
  
class MainWindow:  
 def \_\_init\_\_(self):  
 self.root = tk.Tk()  
 self.root.title("RetroEmuX")  
 self.root.geometry("800x600")  
  
 self.ui\_manager = UIManager(self.root, self)  
 self.rom\_loader = ROMLoader()  
 self.loaded\_rom = None  
  
 self.ui\_manager.setup\_ui()  
  
 def load\_game(self):  
 *"""Открывает диалог выбора файла и загружает игру."""* file\_path = filedialog.askopenfilename(  
 title="Выберите ROM-файл",  
 filetypes=[("ROM Files", "\*.bin;\*.iso;\*.rom;\*.cue")]  
 )  
  
 if file\_path and self.rom\_loader.load\_rom(file\_path):  
 self.loaded\_rom = file\_path  
 messagebox.showinfo("Игра загружена", f"Файл {file\_path} успешно загружен.")  
 self.ui\_manager.enable\_start\_button()  
 else:  
 messagebox.showerror("Ошибка", "Не удалось загрузить игру.")  
  
 def start\_emulation(self):  
 *"""Имитация запуска эмуляции."""* if self.loaded\_rom:  
 messagebox.showinfo("Эмуляция", f"Запуск игры: {self.loaded\_rom}")  
 self.ui\_manager.enable\_stop\_button()  
 else:  
 messagebox.showerror("Ошибка", "Сначала загрузите игру.")  
  
 def stop\_emulation(self):  
 *"""Имитация остановки эмуляции."""* messagebox.showinfo("Эмуляция", "Эмуляция остановлена.")  
 self.ui\_manager.enable\_start\_button()  
  
 def run(self):  
 *"""Запускает главное окно."""* self.root.mainloop()  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = MainWindow()  
 app.run()

Рисунок 2 – Пример файл с ДокБлоками main\_window.py

Sphinx — это популярный инструмент для генерации документации, который изначально был разработан для Python, но теперь используется для создания документации на различных языках программирования, включая C++.

Sphinx работает с текстовыми файлами в формате reStructuredText (.rst) и генерирует документацию в различных форматах, таких как HTML, LaTeX, PDF и других. Для работы с кодом на C++ часто используется расширение Breathe, которое позволяет интегрировать Sphinx с Doxygen, обеспечивая более удобное извлечение комментариев из исходных файлов C++.

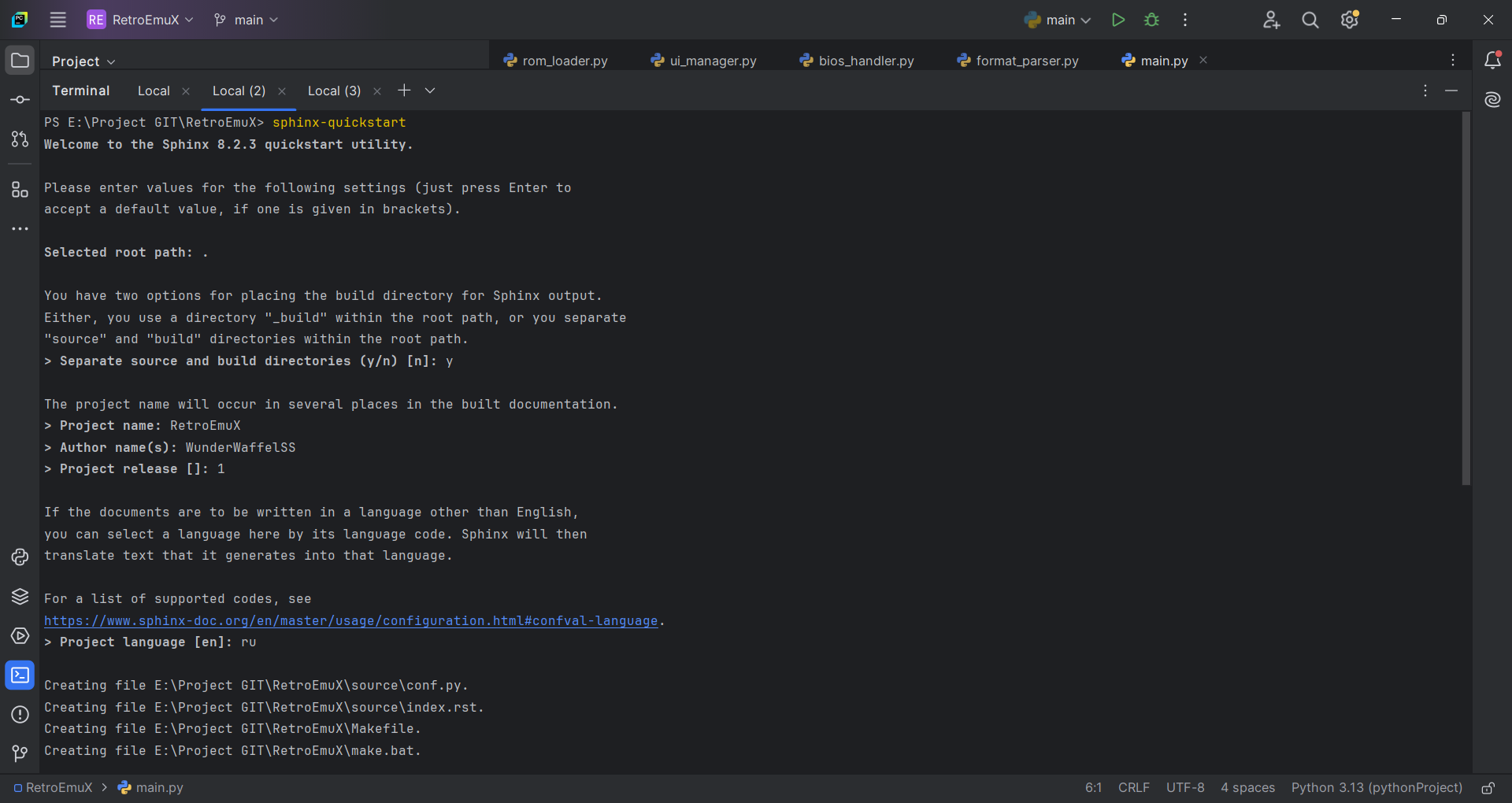


Рисунок 3 – Пример начала работы с Sphinx

Структура полученных файлов представлена на рисунке 3.

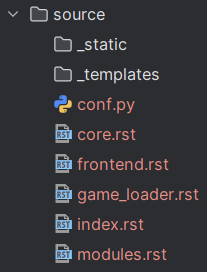


Рисунок 4 – Структура папки source

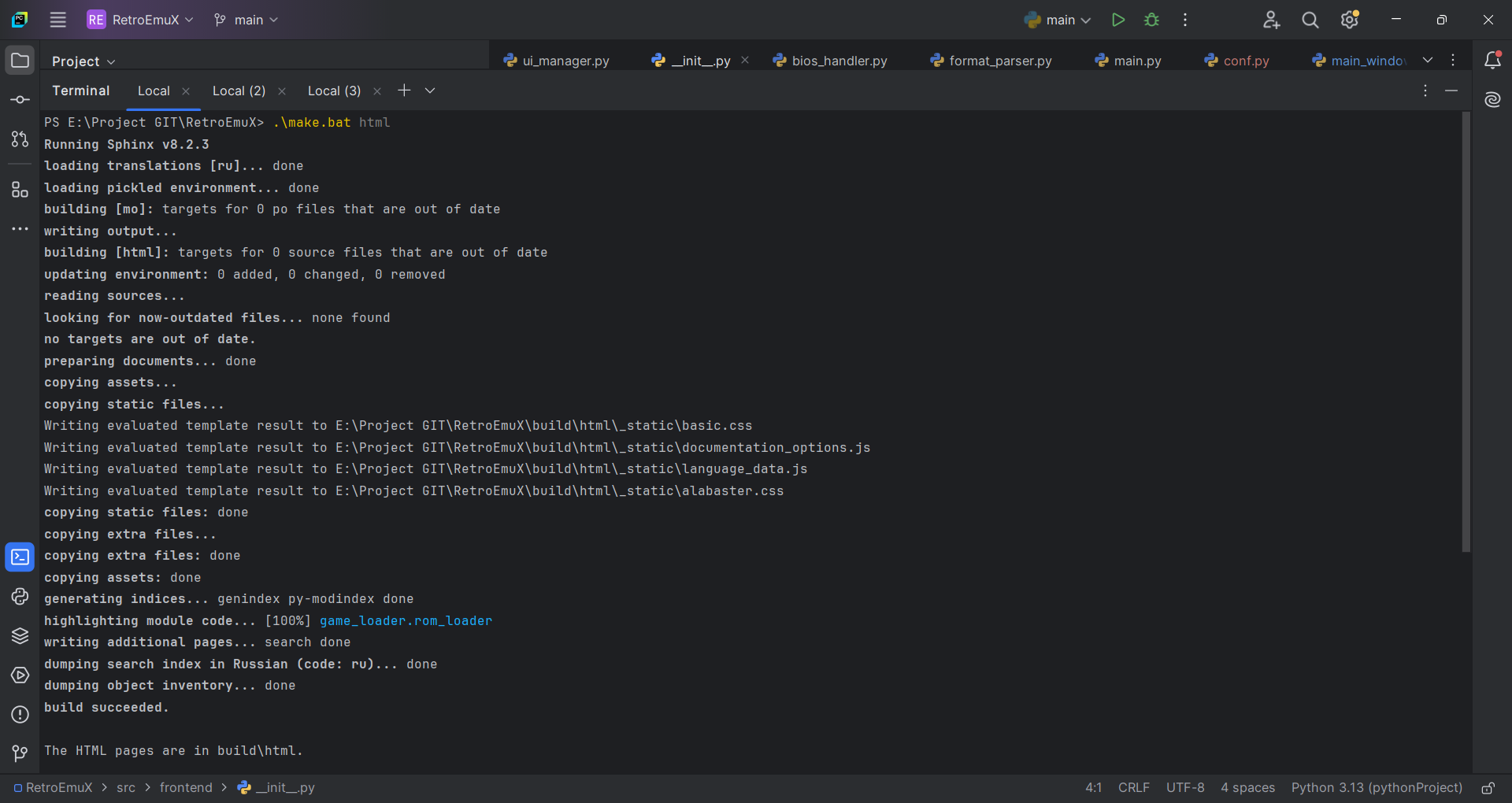


Рисунок 5- Пример создания автоматической документации через make.bat

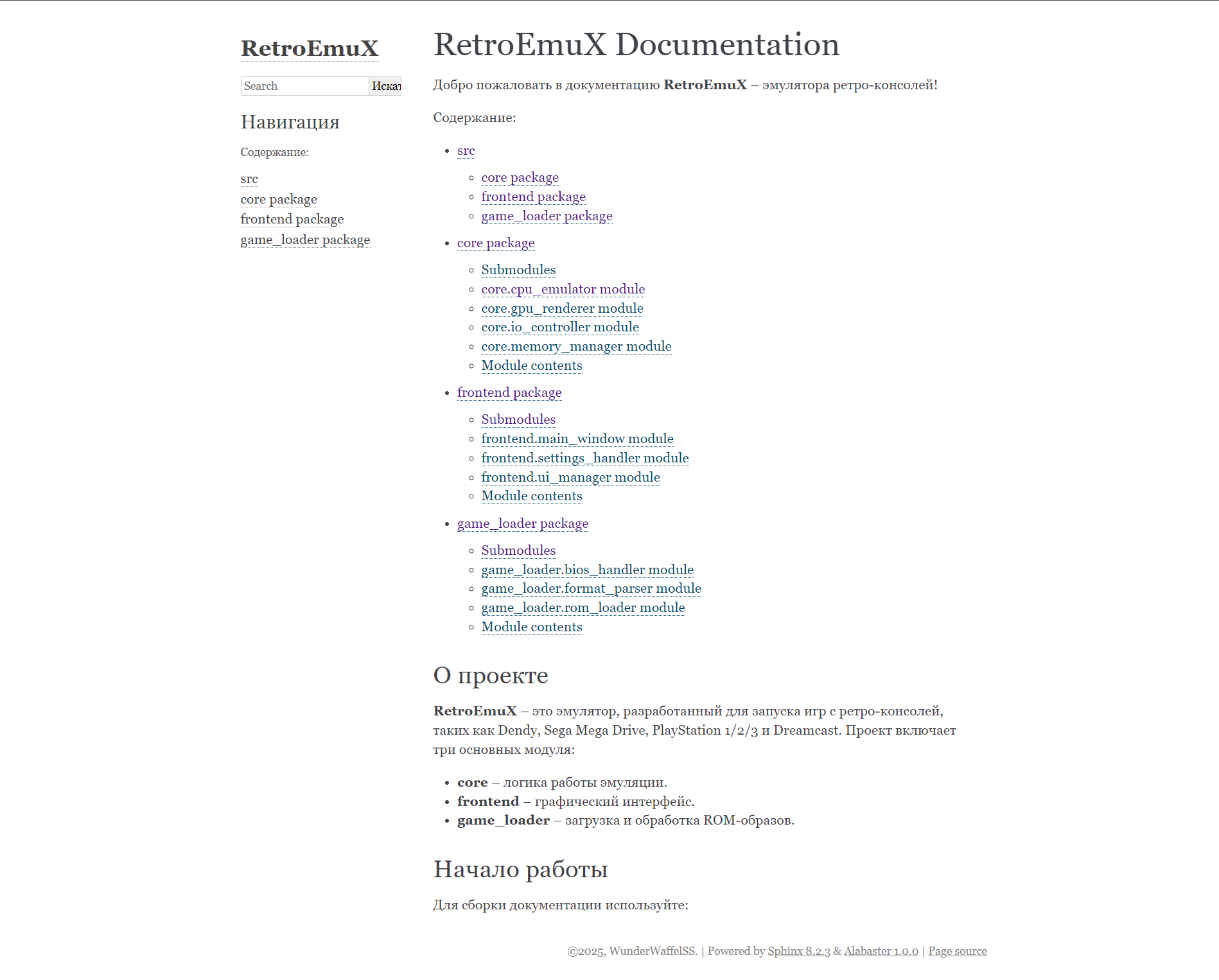


Рисунок 6 – Главная страница получившейся документации

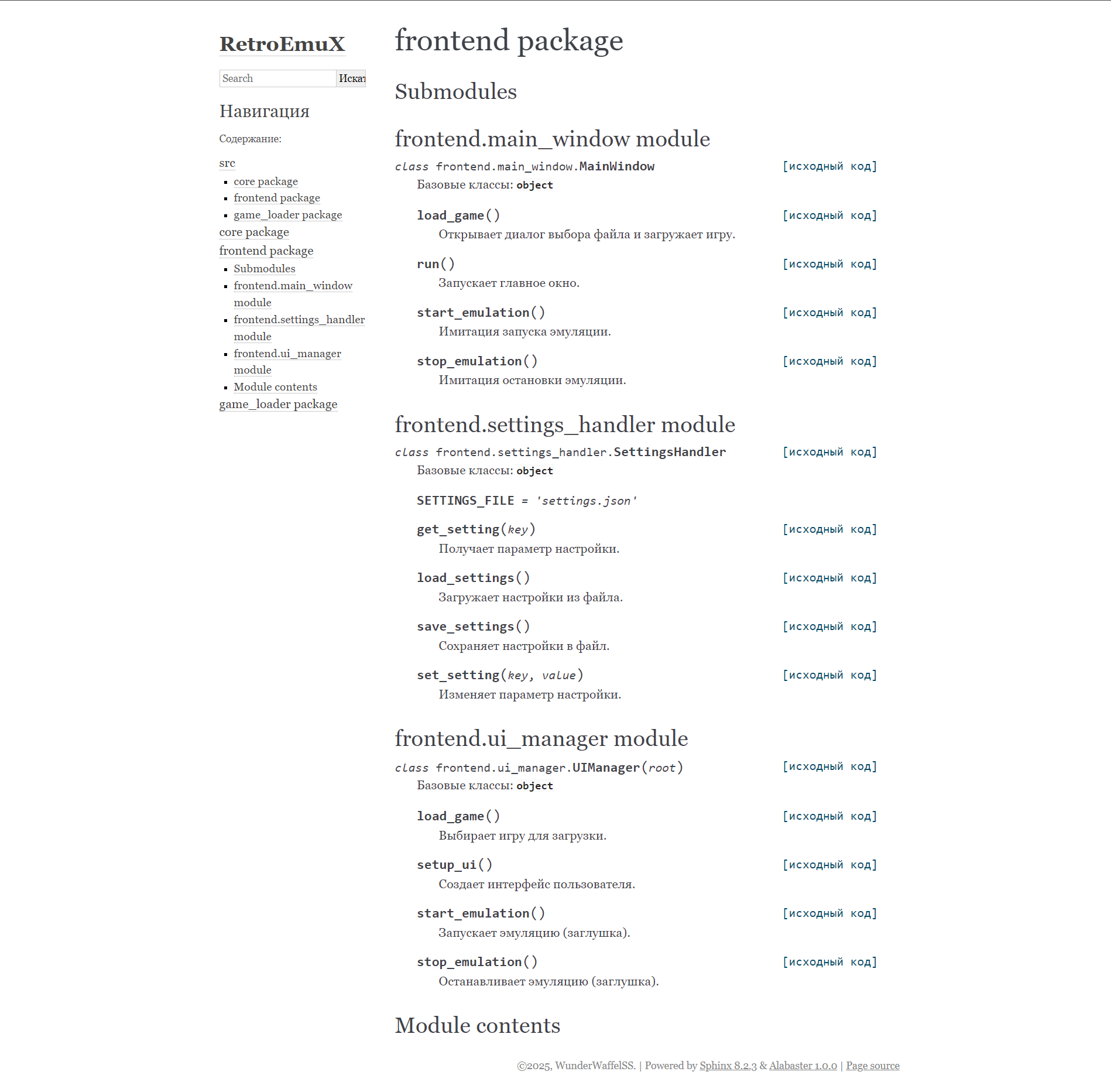


Рисунок 7 – Пример просмотра модуля frontend



Рисунок 8 - Пример просмотра кода файла main\_window.py

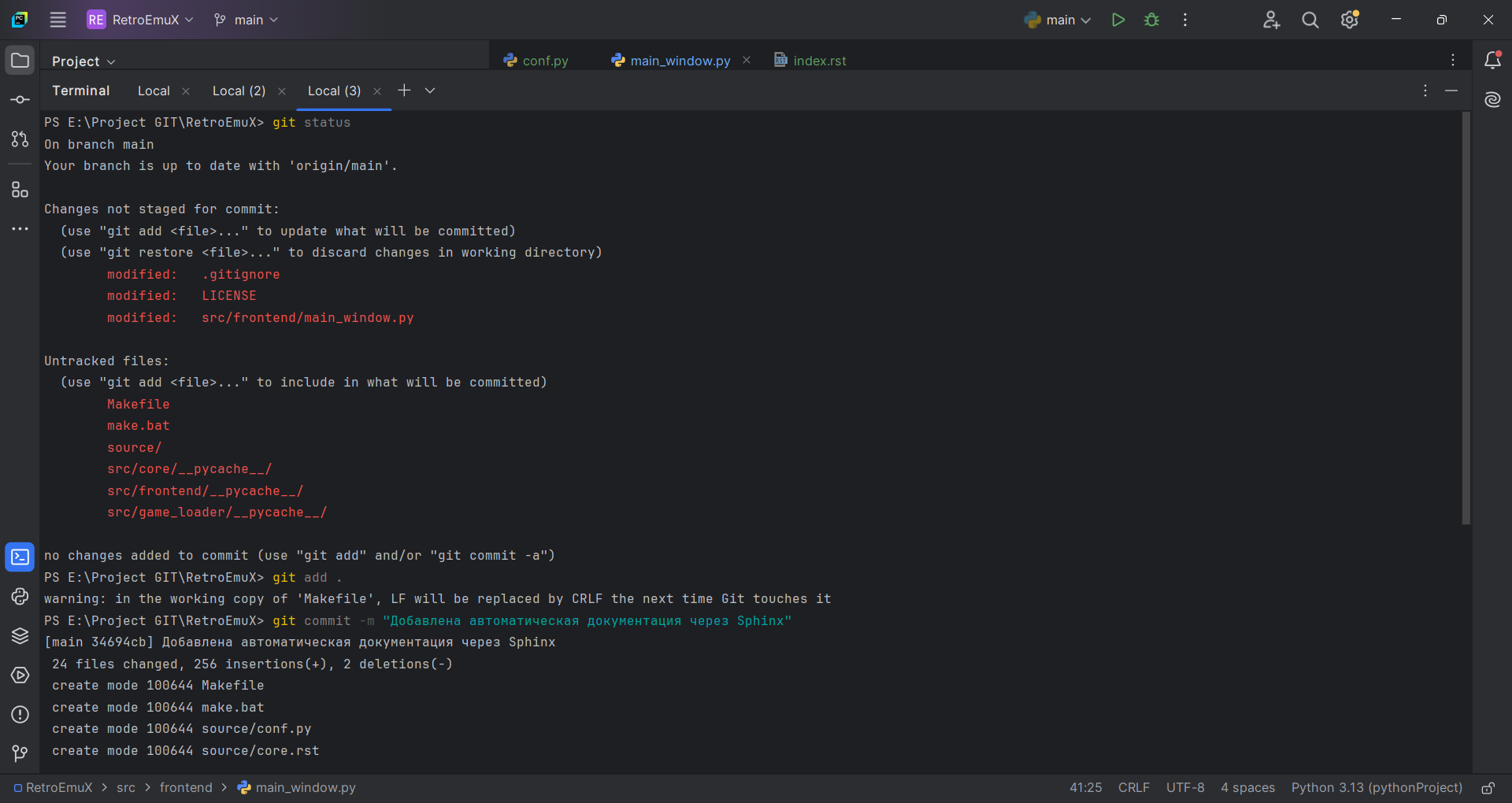


Рисунок 9 – Коммит изменений