Wojciech Matraś WCY21KA1S1 80742

Sprawozdanie PJF nr 2.

Aplikacja do notatek dla osób z ograniczonym widzeniem

Wymagania:

Aplikacja powinna:

-umożliwiać budowę notatki

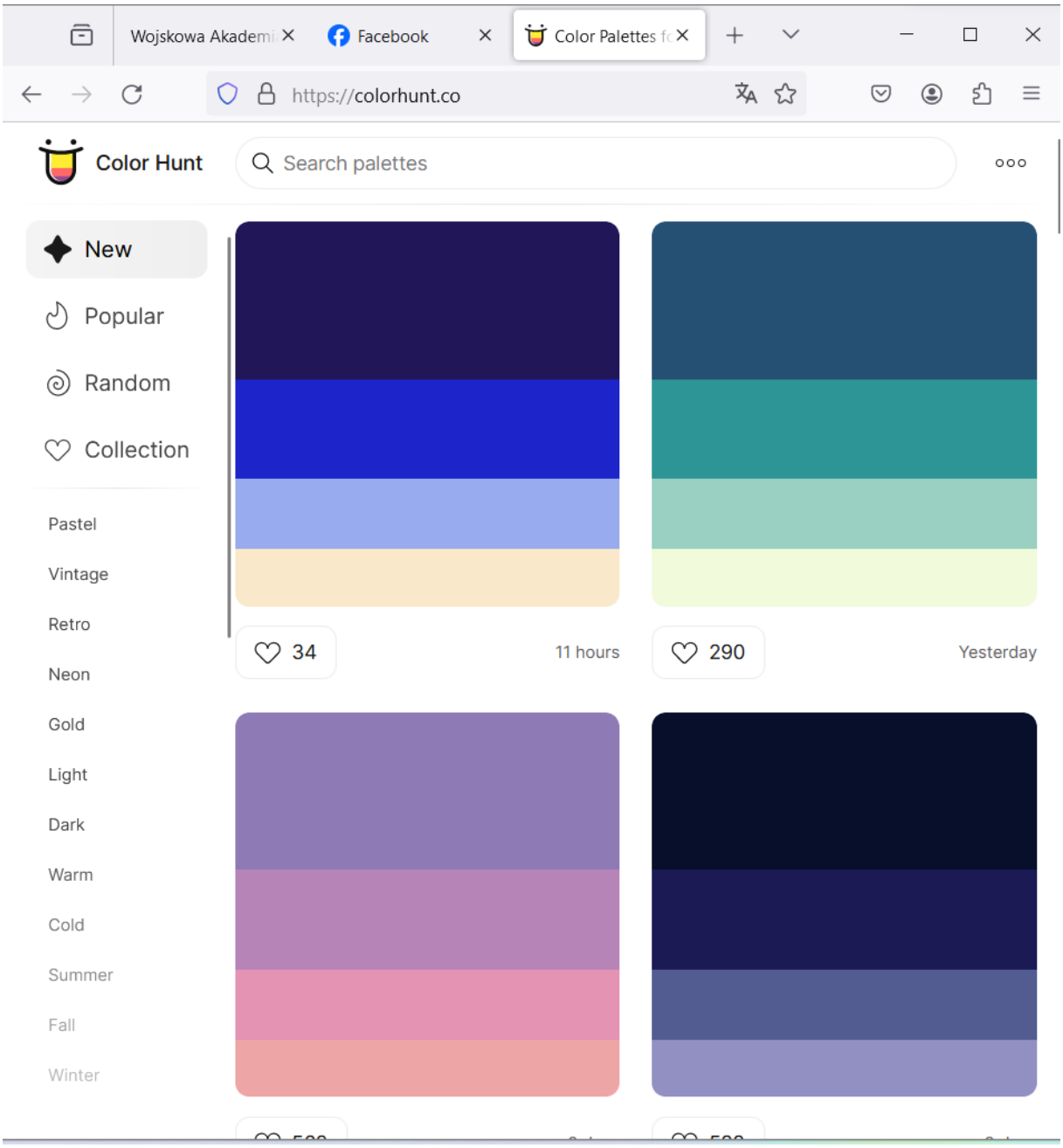
-segregowanie notatek

-usuwanie notatek

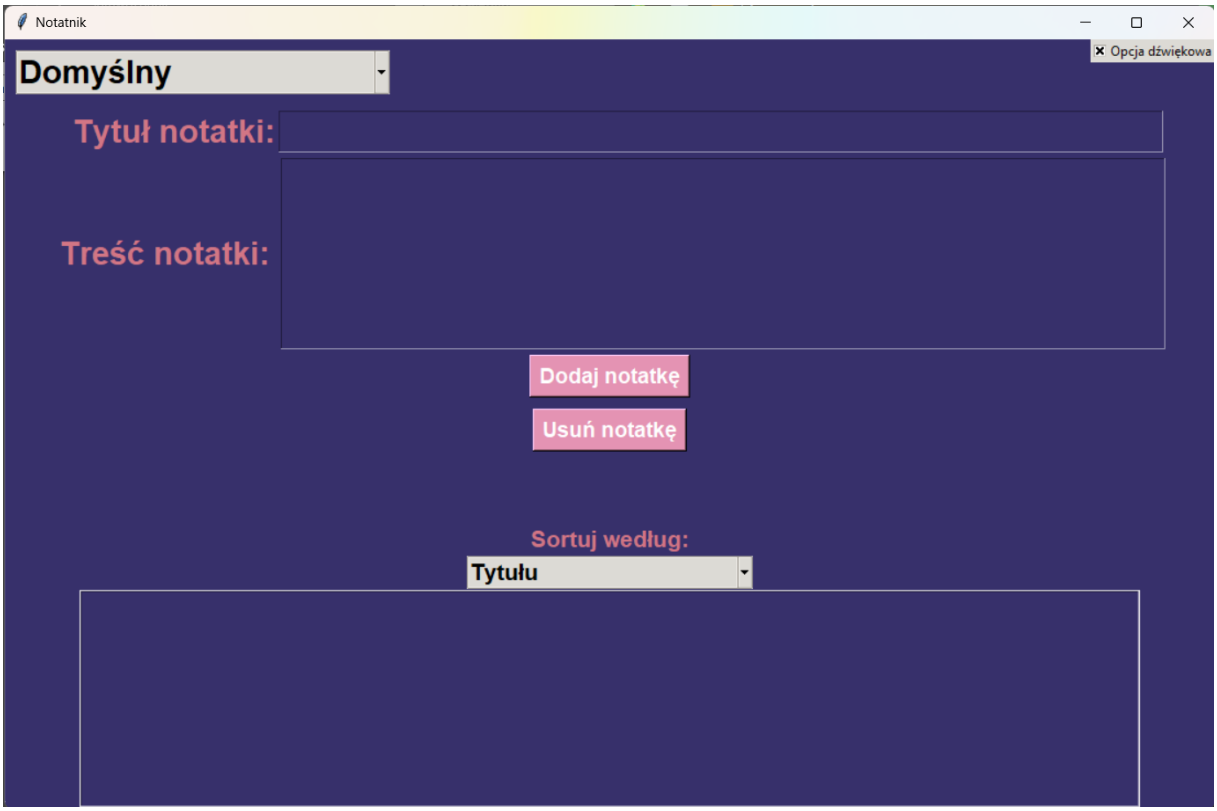
Stworzona aplikacja:

Założenia:

Program będzie miał dużą czcionkę, by osobom niedowidzącym łatwiej było czytać. Interface musi mieć duże kontrasty. By ułatwić używanie programów stworzyłem 4 różne interface'y, wykorzystałem palety barw ze strony: https://colorhunt.co/

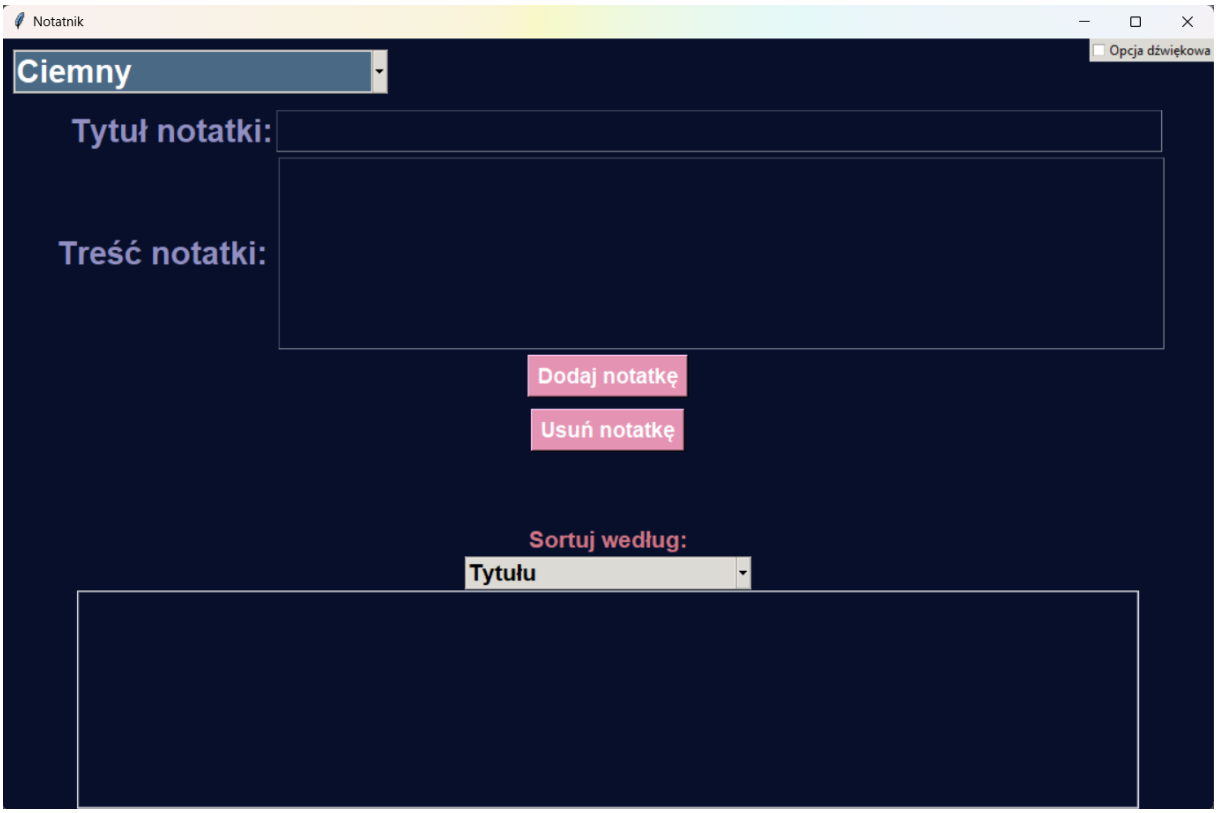


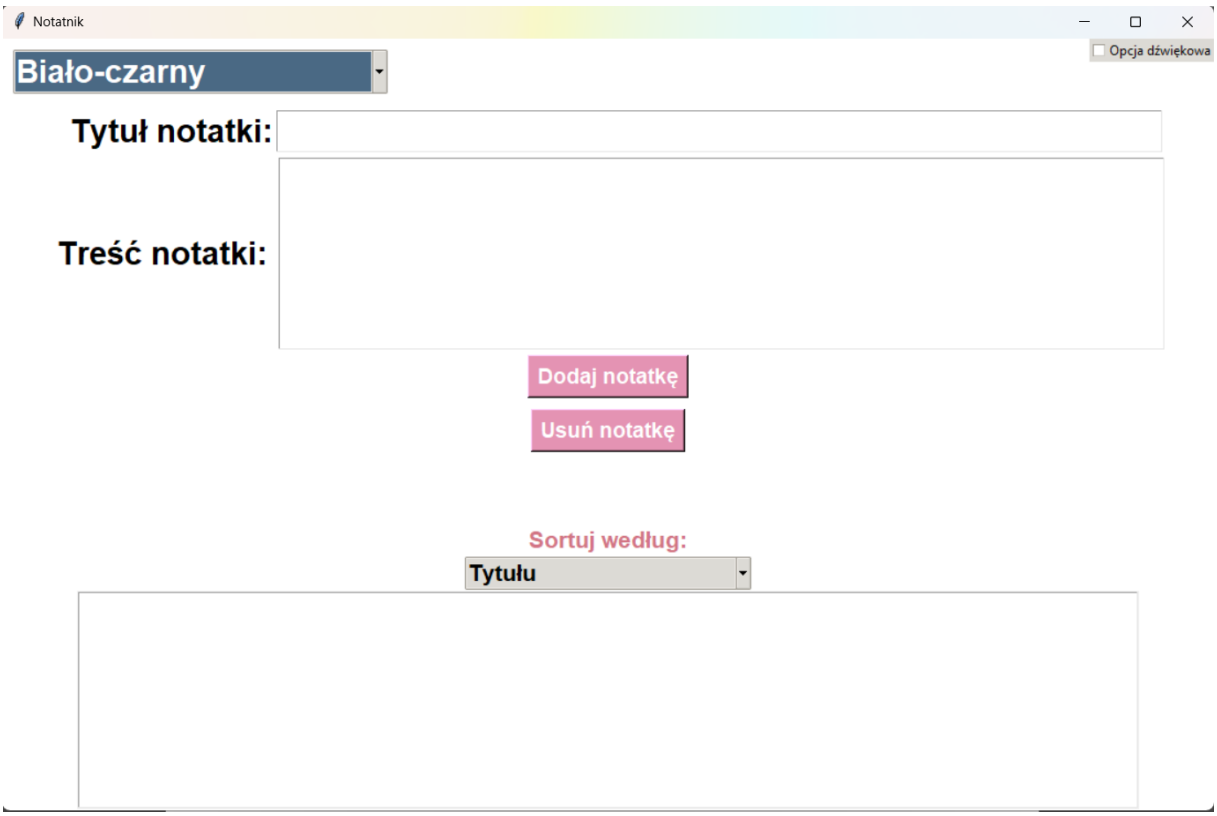
Ekran startowy:

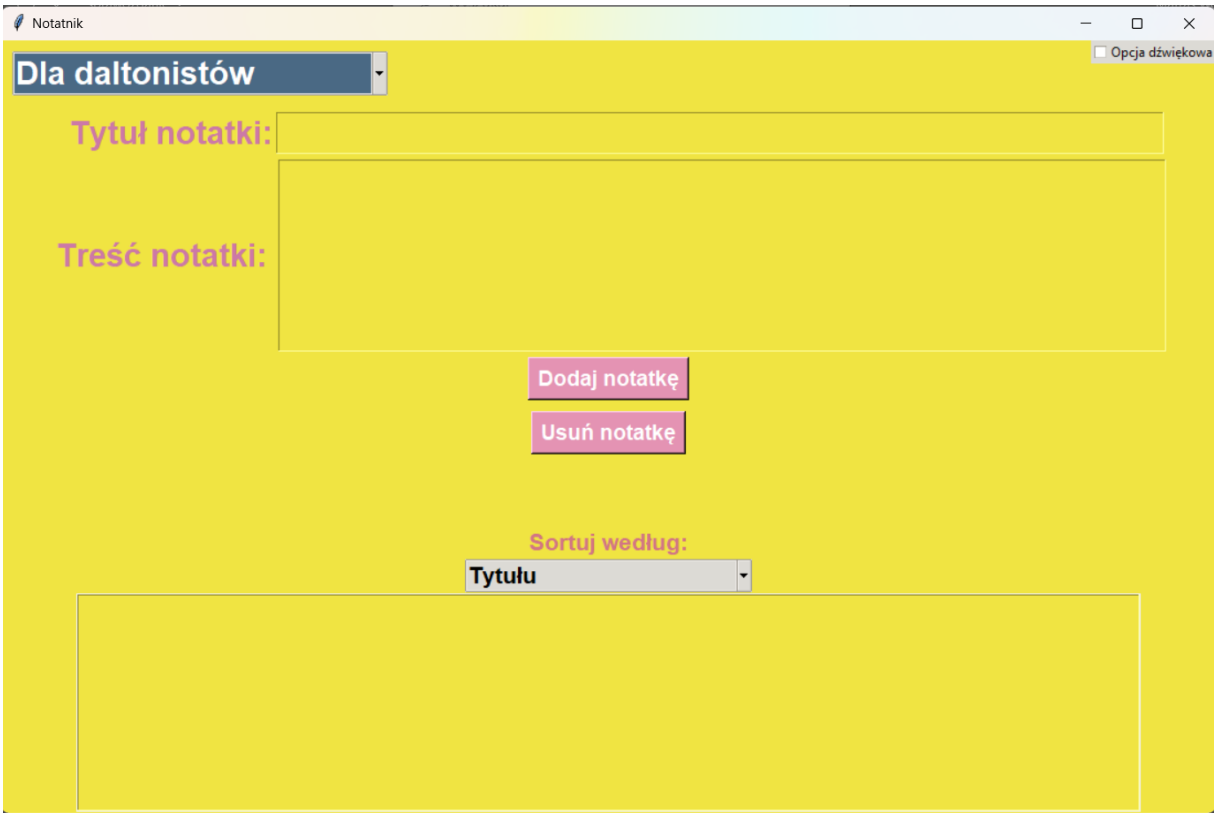


3 inne kolorystyki:

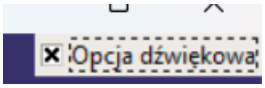
(wybierane poprzez Listę w lewym górym rogu)


Notatnik
Opcja dźwiękowa
Ciemny
Tytuł notatki:
Treść notatki:
Dodaj notatkę
Usuń notatkę
Sortuj według:
Tytułu


Notatnik
Opcja dźwiękowa
Biało-czarny
Tytuł notatki:
Treść notatki:
Dodaj notatkę
Usuń notatkę
Sortuj według:
Tytułu



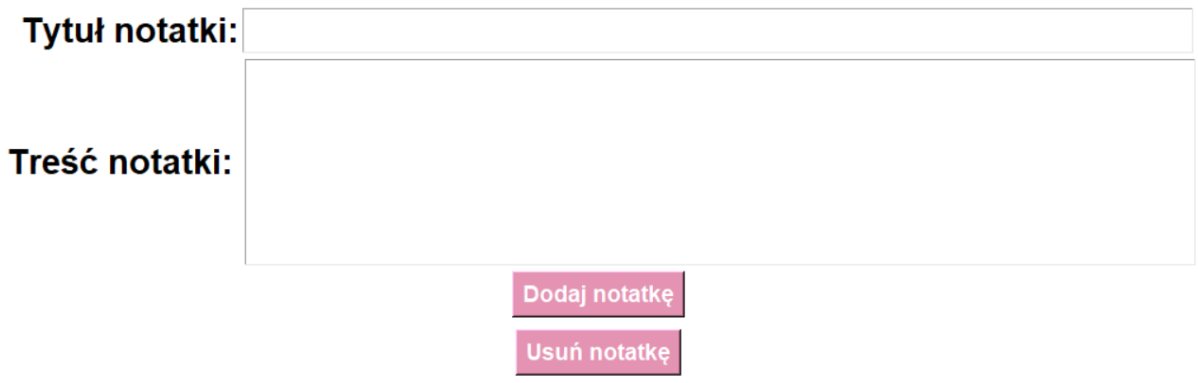
Na starcie zaznacza nam się opcja czytania interfejsu (program jest stworzony z myślą o niedowidzących)



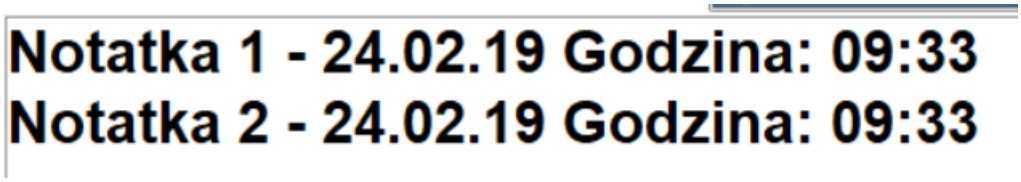
Jednak w każdej chwili jest możliwość wyłączenia tej opcji poprzez odhaczenie.

Opcje notatnika:

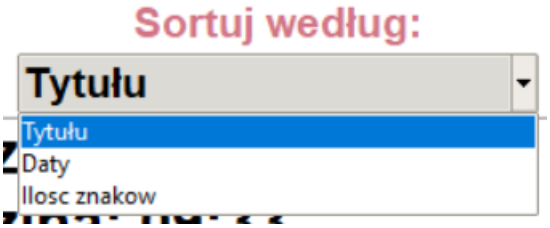
Notatnik jest bardzo prosty, ponieważ target jakim są osoby niedowidzące w większości są w wieku starszym dlatego obsługa skomplikowanych aplikacji może sprawiać im problemy.



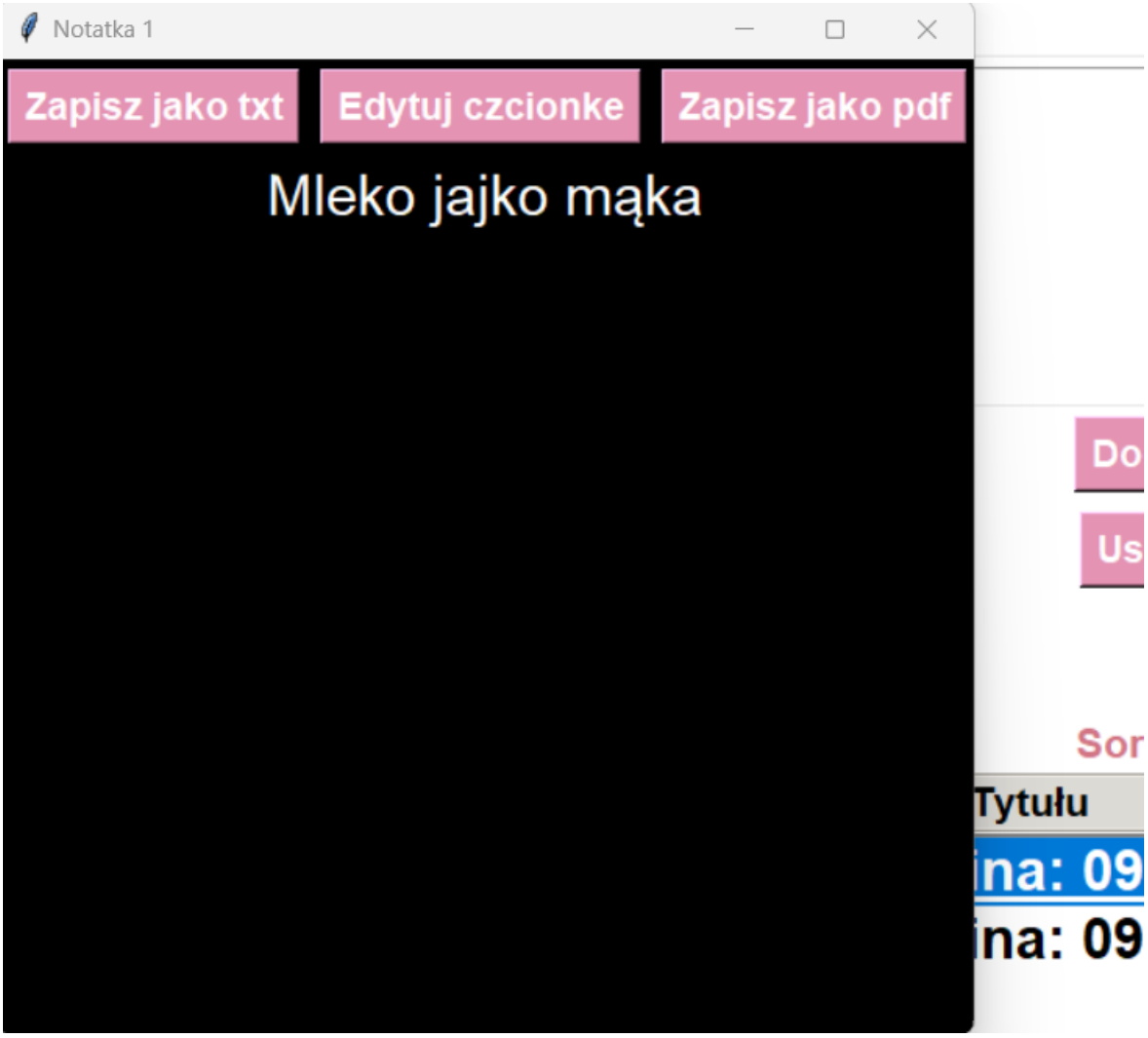
Każda notatka dodana, ma tytuł datę dodania, oraz oczywiście treść.



Jest możliwość sortowania notatek na 3 sposoby:

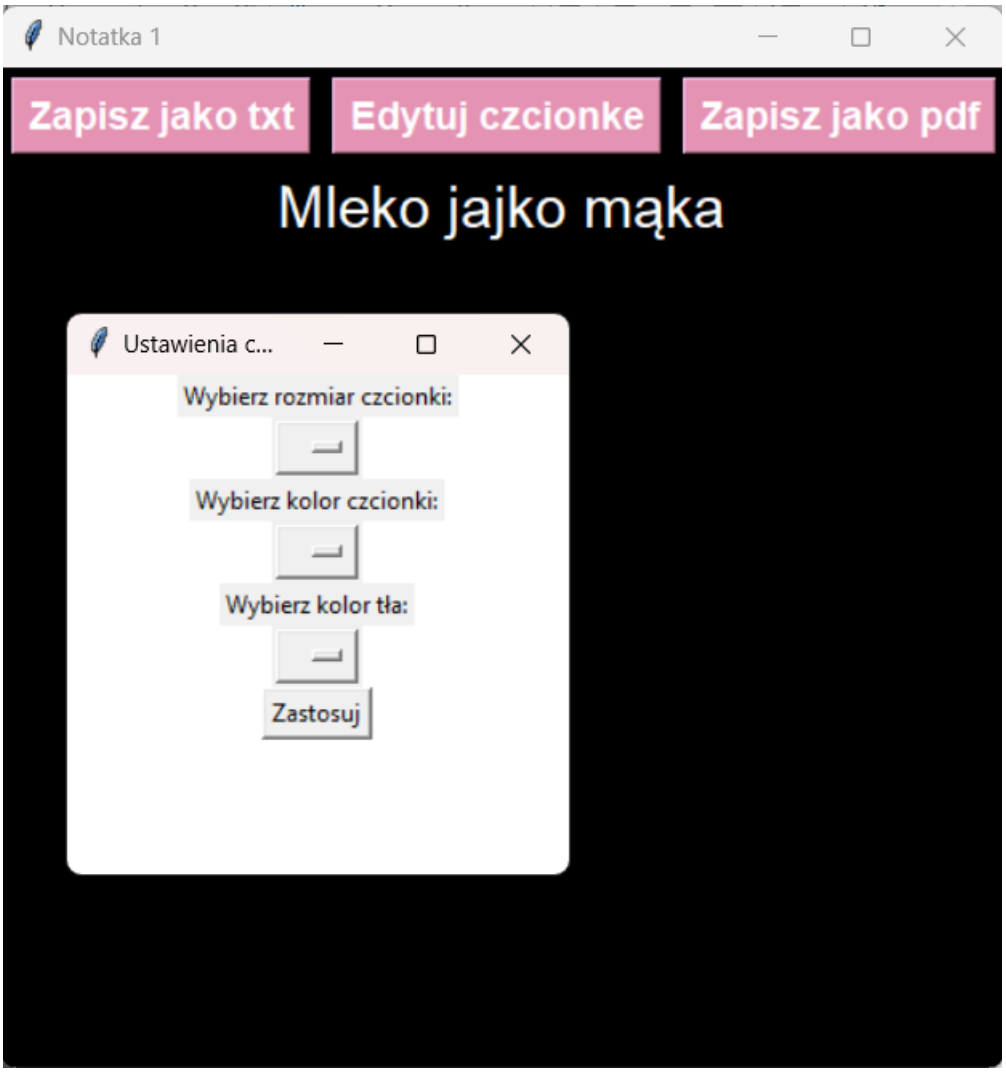


Ostatecznie odczytywanie notatek wygląda w ten sposób:

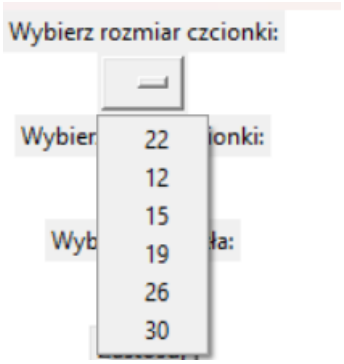


Ekran odczytu notatki ma kontrastować z ekranym głównym notatnika dlatego użyłem reversed kolorów, by zwiększyć ten kontrast. 3 przyciski mają pomóc w przejrzystości, kolor ich i wielkość są taki by były widoczne.

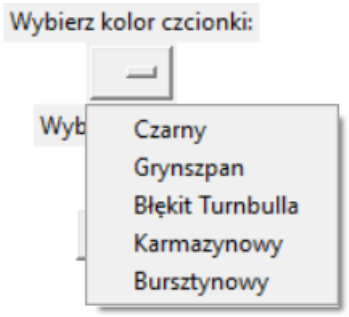
Przycisk Edytuj Czcionke przenosi nas do jeszcze 1 okienka:



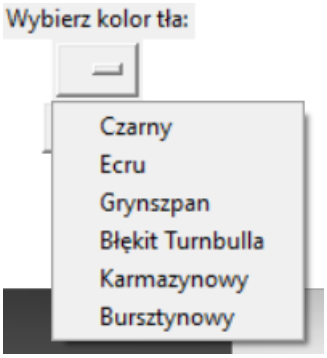
Mamy do wyboru, rozmiar czcionki pliku pdf



, kolor tej czcionki



Oraz kolor tła pdf’a (by w łatwy sposób edytować tło użyłem gotowych plików jpg z szukanym kolorem)



Bez zmiany ustawień plik pdf będzie w wersji: 22, czarna czcionka, białe tło.

Przykładowy plik pdf

Notatka
maka, mleko, jajka

Program w pythonie:

```
import tkinter as tk
from tkinter import messagebox
from tkinter import ttk
from datetime import datetime
import pyttsx3
from fpdf import FPDF

class NoteApp:
    def __init__(self, gui):
        self.gui = gui
        self.gui.title("Notatnik")
        self.gui.geometry("1100x700")
        self.gui.configure(bg="#37306B")
        self.engine = pyttsx3.init()
        self.sound_enabled = tk.BooleanVar()
        self.sound_enabled.set(True)
        self.selected_size = 22
        self.selected_font_color = "black"
        self.backgroung_color = "white"
        self.colorSet = {
            "Domyślny": {"bg": "#37306B", "fg": "#D27685"},
            "Ciemny": {"bg": "#070F2B", "fg": "#9290C3"},
            "Biało-czarny": {"bg": "white", "fg": "black"},
            "Dla daltonistów": {"bg": "#f0e442", "fg": "#cc79a7"}
        }

        self.colorWart = tk.StringVar()
        self.colorWart.set("Domyślny")
        colorZmienna = list(self.colorSet.keys())
        self.colorChanger = ttk.Combobox(self.gui,
textvariable=self.colorWart, values=colorZmienna, state="readonly",
font=('Helvetica', 22, 'bold'))
        self.colorChanger.place(x=10, y=10)
        self.colorChanger.bind("<<ComboboxSelected>>", self.colorChange)

        style = ttk.Style()
        style.theme_use('classic')

        self.Frame = tk.Frame(self.gui, bg=self.colorSet["Domyślny"]["bg"])
        self.Frame.pack(pady=60, padx=10)
        self.sound_toggle = ttk.Checkbutton(self.gui, text="Opcja
```

```
dźwiękowa", variable=self.sound_enabled, onvalue=True, offvalue=False,
command=self.toggle_sound)
        self.sound_toggle.place(relx=1.0, rely=0, anchor="ne")
        self.tytul = tk.Label(self.Frame, text="Tytuł notatki:",
font=('Helvetica', 22, 'bold'), bg=self.colorSet["Domyślny"]["bg"],
fg=self.colorSet["Domyślny"]["fg"])
        self.tytul.grid(row=0, column=0, padx=0, pady=0, sticky="e")
        self.SortujWedlug = tk.Label(self.gui, text="Sortuj według:",
font=('Helvetica', 16, 'bold'),
                                     bg=self.colorSet["Domyślny"]["bg"],
fg=self.colorSet["Domyślny"]["fg"])
        self.SortujWedlug.pack(side=tk.TOP, padx=0, pady=0)
        self.SortujZmienna = tk.StringVar()
        self.SortujZmienna.set("Tytułu")
        sort_by_options = ["Tytułu", "Daty", "Ilosc znakow"]

        style = ttk.Style()
        style.theme_use('clam')

        self.SortujWybrana = ttk.Combobox(self.gui,
textvariable=self.SortujZmienna, values=sort_by_options, state="readonly",
font=('Helvetica', 16, 'bold'))
        self.SortujWybrana.pack(side=tk.TOP, padx=0, pady=0)
        self.SortujWybrana.bind("<<ComboboxSelected>>", self.sort)
        self.TytulBox = tk.Entry(self.Frame, width=50, font=('Helvetica',
22, 'bold'), bg=self.colorSet["Domyślny"]["bg"],
fg=self.colorSet["Domyślny"]["fg"])
        self.TytulBox.grid(row=0, column=1, padx=(0, 5), pady=5)

        self.Label1 = tk.Label(self.Frame, text="Treść notatki:",
font=('Helvetica', 22, 'bold'), bg=self.colorSet["Domyślny"]["bg"],
fg=self.colorSet["Domyślny"]["fg"])
        self.Label1.grid(row=1, column=0, padx=5, pady=5, sticky="e")

        self.BoxNotatki = tk.Text(self.Frame, width=50, height=5,
font=('Helvetica', 22, 'bold'), bg=self.colorSet["Domyślny"]["bg"],
fg=self.colorSet["Domyślny"]["fg"], wrap=tk.WORD)
        self.BoxNotatki.grid(row=1, column=1, padx=0, pady=0)
        self.BoxNotatki.bind("<Return>", lambda event: "break")

        self.AddButton1 = tk.Button(self.Frame, text="Dodaj notatkę",
command=self.dodajNotatke, bg="#E493B3", fg="white",font=('Helvetica', 14,
'bold'))
        self.AddButton1.grid(row=2, columnspan=2, padx=5, pady=5)


        self.Button2 = tk.Button(self.Frame, text="Usuń notatkę",
command=self.delete, bg="#E493B3", fg="white",font=('Helvetica', 14,
'bold'))
        self.Button2.grid(row=3, columnspan=2, padx=5, pady=5)

        self.NotatkiWszyskie = tk.Listbox(self.gui, width=60, height=15,
font=('Helvetica', 22, 'bold'), bg=self.colorSet["Domyślny"]["bg"],
fg=self.colorSet["Domyślny"]["fg"])
        self.NotatkiWszyskie.pack(padx=1, pady=1)

        self.NotatkiWszyskie.bind("<ButtonRelease-1>", self.display_note)
        self.TytulBox.bind("<Return>", lambda event: self.dodajNotatke())
        print(self.sound_enabled.get())

        self.TytulBox.bind("<Enter>", lambda event: self.speak("Pole tytułu
```

```
notatki"))
        self.SortujWybrana.bind("<Enter>", lambda event: self.speak("Sortuj
według:"))
        self.colorChanger.bind("<Enter>", lambda event: self.speak("Pole
zmiany kolorystyki"))
        self.NotatkiWszyskie.bind("<Enter>", lambda event:
self.speak("Lista wszystkich notatek"))
        self.BoxNotatki.bind("<Enter>", lambda event: self.speak("Pole do
wpisywania notatki"))
        self.AddButton1.bind("<Enter>", lambda event: self.speak("Dodaj
notatkę"))
        self.Button2.bind("<Enter>", lambda event: self.speak("Usuń
notatkę"))

    def toggle_sound(self):
        if not self.sound_enabled.get():
            messagebox.showinfo("Dźwięk włączony", "Opcja dźwiękowa jest
teraz wyłączona.")


    def speak(self, text):
        if (self.sound_enabled.get() == True):
            self.engine.say(text)
            self.engine.runAndWait()
    def colorChange(self, event=None):
        colorSet = self.colorWart.get()
        bgColor = self.colorSet[colorSet]["bg"]
        fgColor = self.colorSet[colorSet]["fg"]
        self.gui.configure(bg=bgColor)
        self.SortujWedlug.configure(bg=bgColor)
        self.Frame.configure(bg=bgColor)
        self.tytul.configure(bg=bgColor, fg=fgColor)
        self.TytulBox.configure(bg=bgColor, fg=fgColor)
        self.Label1.configure(bg=bgColor, fg=fgColor)
        self.BoxNotatki.configure(bg=bgColor, fg=fgColor)
        self.AddButton1.configure(bg="#E493B3", fg="white")
        self.NotatkiWszyskie.configure(bg=bgColor, fg=fgColor)
        self.Button2.configure(bg="#E493B3", fg="white")



    def dodajNotatke(self, event=None):
        title = self.TytulBox.get().strip()
        BoxNotatki = self.BoxNotatki.get("1.0", "end-1c").strip()
        if title:
            note_title = title
        elif len(BoxNotatki) > 20:
            note_title = BoxNotatki[:20]
        else:
            note_title = BoxNotatki

        date_str = datetime.now().strftime('%y.%m.%d Godzina: %H:%M')
        self.NotatkiWszyskie.insert(tk.END, f"{note_title} - {date_str}")
        with open(f"{note_title}.txt", "w") as f:
            f.write(BoxNotatki)
        self.TytulBox.delete(0, tk.END)
        self.BoxNotatki.delete("1.0", tk.END)


    def delete(self):
        zaznaczony = self.NotatkiWszyskie.curselection()
```

```
        if zaznaczony:
            note_title = self.NotatkiWszyskie.get(zaznaczony).split(" -
")[0]
            self.NotatkiWszyskie.delete(zaznaczony)

            with open(f"{note_title}.txt", "r") as f:
                BoxNotatki = f.read()
            messagebox.showinfo(note_title, BoxNotatki)

        else:
            messagebox.showwarning("Uwaga", "Wybierz notatkę do
usunięcia.")


    def display_note(self, event=None):
        zaznaczony = self.NotatkiWszyskie.curselection()
        if zaznaczony:
            note_title = self.NotatkiWszyskie.get(zaznaczony).split(" -
")[0]

            with open(f"{note_title}.txt", "r") as f:
                BoxNotatki = f.read()
            note_window = tk.Toplevel(self.gui)
            note_window.title(note_title)

note_window.configure(bg=self.colorSet[self.colorWart.get()]["fg"])
            note_window.geometry("500x500")

            self.AddButton3 = tk.Button(note_window, text="Zapisz jako
txt",
                                        command=lambda:
self.zapiszTXT(note_title, BoxNotatki), bg="#E493B3",
                                        fg="white", font=('Helvetica', 14,
'bold'))
            self.AddButton3.place(x=5, y=5)
            self.AddButton4 = tk.Button(note_window, text="Zapisz jako
pdf",
                                        command=lambda:
self.zapiszPDF(note_title, BoxNotatki), bg="#E493B3",
                                        fg="white", font=('Helvetica', 14,
'bold'))
            self.AddButton4.place(x=340, y=5)
            self.AddButton5 = tk.Button(note_window, text="Edytuj
czcionke",
                                        command=self.options, bg="#E493B3",
                                        fg="white", font=('Helvetica', 14,
'bold'))
            self.AddButton5.place(x=165, y=5)
            Label1 = tk.Label(note_window, text=BoxNotatki,
font=("Helvetica", 22),
                              bg=self.colorSet[self.colorWart.get()]["fg"],
                              fg=self.colorSet[self.colorWart.get()]["bg"],
wraplength=400)
            #self.AddButton3.bind("<Enter>", lambda event:
self.speak("Zapisz jako txt"))
            #self.AddButton4.bind("<Enter>", lambda event:
self.speak("Zapisz jako pdf"))
            #self.AddButton5.bind("<Enter>", lambda event:
self.speak("Edytuj czcionke i tło"))
            Label1.pack(padx=20, pady=50)
```

```
    def options(self):
        optionsGui = tk.Toplevel(self.gui)
        optionsGui.configure(bg=self.colorSet[self.colorWart.get()]["bg"])
        optionsGui.title("Ustawienia czcionki i tła")
        optionsGui.geometry("250x250")

        def apply_settings():
            print(VarFontSize)
            print(varFontColor.get())
            print(int(VarFontSize.get()))
            self.selected_size = str(VarFontSize.get())
            print(self.selected_size)
            self.selected_font_color = str(varFontColor.get())
            self.backgroung_color = str(varBgColor.get())
            optionsGui.destroy()

        VarFontSize = tk.StringVar()
        varFontColor = tk.StringVar()
        varBgColor = tk.StringVar()

        tk.Label(optionsGui, text="Wybierz rozmiar czcionki:").pack()
        tk.OptionMenu(optionsGui, VarFontSize, "22", "12", "15", "19",
"26", "30").pack()

        tk.Label(optionsGui, text="Wybierz kolor czcionki:").pack()
        tk.OptionMenu(optionsGui, varFontColor, "Czarny", "Grynszpan",
"Błękit Turnbulla", "Karmazynowy", "Bursztynowy").pack()

        tk.Label(optionsGui, text="Wybierz kolor tła:").pack()
        tk.OptionMenu(optionsGui, varBgColor, "Czarny", "Ecru",
"Grynszpan", "Błękit Turnbulla", "Karmazynowy",
                      "Bursztynowy").pack()

        tk.Button(optionsGui, text="Zastosuj",
command=apply_settings).pack()

        optionsGui.mainloop()

    def zapiszPDF(self, title, content):
        pdf = FPDF()
        pdf.add_page()
        size=int(self.selected_size)
        print(self.selected_font_color)

        pdf.set_font("Arial", size=size)
        print(self.backgroung_color)
        if(self.selected_font_color == "Grynszpan"):
            pdf.set_text_color(0, 166, 147)
        if(self.selected_font_color == "Błękit Turnbulla"):
            pdf.set_text_color(8,46,121)
        if(self.selected_font_color == "Karmazynowy"):
            pdf.set_text_color(220,20,60)
        if(self.selected_font_color == "Bursztynowy"):
            pdf.set_text_color(255, 191, 0)
        if (self.backgroung_color == "Czarny"):
            pdf.image("czarny.jpg", x=0, y=0, w=pdf.w, h=pdf.h)
        if (self.backgroung_color == "Ecru"):
            pdf.image("ecru.jpg", x=0, y=0, w=pdf.w, h=pdf.h)
        if (self.backgroung_color == "Grynszpan"):
            pdf.image("grynszpan.jpg", x=0, y=0, w=pdf.w, h=pdf.h)
        if (self.backgroung_color == "Błękit Turnbulla"):
```

```
            pdf.image("blekit.jpg", x=0, y=0, w=pdf.w, h=pdf.h)
        if (self.backgroung_color == "Karmazynowy"):
            pdf.image("karmazynowy.jpg", x=0, y=0, w=pdf.w, h=pdf.h)
        if (self.backgroung_color == "Bursztynowy"):
            pdf.image("bursztyn.jpg", x=0, y=0, w=pdf.w, h=pdf.h)
        pdf.cell(200, 10, txt=title, ln=True)
        pdf.cell(200, 10, txt=content, ln=True)
        pdf.output(f"{title}.pdf")
        messagebox.showinfo("Zapisano", "Notatka została zapisana jako
PDF.")

    def zapiszTXT(self, title, content):
        with open(f"{title}.txt", "w") as f:
            f.write(content)
        messagebox.showinfo("Zapisano", "Notatka została zapisana jako
TXT.")
    def sort(self, event=None):
        sort_by = self.SortujZmienna.get()
        notes = list(self.NotatkiWszyskie.get(0, tk.END))  # Pobranie
notatek z listy
        if sort_by == "Tytułu":
            self.NotatkiWszyskie.delete(0, tk.END)
            sorted_notes = sorted(notes)
            for note in sorted_notes:
                self.NotatkiWszyskie.insert(tk.END, note)
        elif sort_by == "Daty":
            self.NotatkiWszyskie.delete(0, tk.END)
            notatkiDAty = [(note.split(" - ")[1], note) for note in notes]
            sorted_notes = sorted(notatkiDAty)
            for date, note in sorted_notes:
                self.NotatkiWszyskie.insert(tk.END, note)
def main():
    root = tk.Tk()
    root.title("Notatnik")
    app = NoteApp(root)
    root.bind("<Return>", app.dodajNotatke)
    root.mainloop()

if __name__ == "__main__":
    main()
```