Gra Mistrz Klawiatury

(SimplyType)

1. Uczestnicy projektu:

- Mateusz Basiura
- Maksym Boiar
- Filip Chałupka

2. Linki

Repozytorium: https://github.com/mboiar/SimplyType

Projekt: https://github.com/users/mboiar/projects/1/views/1

Dokumentacja: https://github.com/mboiar/SimplyType/blob/main/docs/documentation.txt

3. Cel projektu

Opis:

Gra Mistrz Klawiatury to jedna z popularnych gier edukacyjnych, której celem jest poprawa szybkości i dokładności pisania na klawiaturze. Gracze muszą wprowadzać tekst z ekranu do swojej klawiatury, unikając popełnienia błędów i starając się robić to jak najszybciej.

Gra składa się z różnych poziomów trudności, które obejmują różne tematy, takie jak nauka języków obcych, pisanie kodu lub wprowadzanie danych. Każdy poziom składa się z zestawu zdań lub słów, które gracz musi wpisać jak najszybciej i najdokładniej.

Głównym celem gry jest doskonalenie umiejętności pisania na klawiaturze, co może przyspieszyć pracę na komputerze i zwiększyć efektywność w pracy. Osiągnięcie wysokiej dokładności i szybkości pisania na klawiaturze może być bardzo przydatne w wielu zawodach, takich jak dziennikarstwo, marketing internetowy, programowanie, czy nawet sprzedaż.

Gra Mistrz Klawiatury może być również przydatna w codziennym życiu, na przykład podczas pisania e-maili, notatek lub dokumentów. Poprawa umiejętności pisania na klawiaturze może przyspieszyć pisanie i zmniejszyć ilość popełnianych błędów, co z kolei zwiększa ogólną produktywność.

Podsumowując, celem gry Mistrz Klawiatury jest poprawa umiejętności pisania na klawiaturze, co może przyspieszyć pracę na komputerze, zwiększyć efektywność w pracy oraz być przydatne w codziennym życiu. Poprzez systematyczne ćwiczenia gracz może poprawić swoje umiejętności i osiągnąć wyższy poziom szybkości i dokładności pisania.

Funkcjonalność:

- Baza danych złożona z 225 fraz
- Do wyboru słowa jak i zdania
- Wybór wersji językowej
- Wybór motywu interfejsu

- Statystyki gry
- Różne tryby gry (Nauka, Wyzwanie, Zen)
- Możliwość powiększania bazy danych przez użytkowników
- Możliwość zapisu gry

Przedstawienie kodu:

Przedstawiono tylko najważniejsze fragmenty ze względu na kolosalną ilość linijek kodu (ok. 5000 linijek).

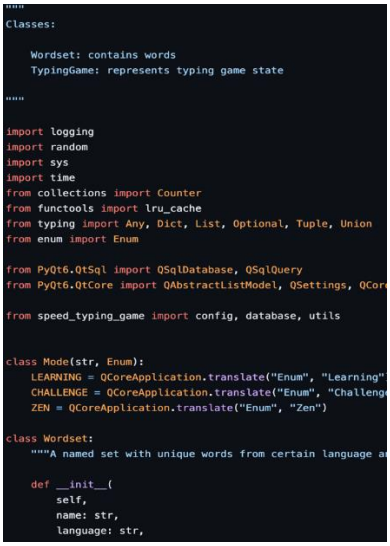
- database.py - komunikacja z bazą danych

```python
import logging
from functools import lru_cache
from typing import Iterable, List, Tuple, Optional
from collections import Counter
import time
import datetime

from PyQt6.QtSql import QSqlDatabase, QSqlQuery, QSqlQueryModel

from speed_typing_game import config, models, utils

logger = logging.getLogger(__name__)
createWordsetTableQueryString = f"""
            CREATE TABLE IF NOT EXISTS {config.WORDSET_TABLE} (
                id INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE NOT NULL,
                name VARCHAR(40) NOT NULL,
                language_code VARCHAR(6) NOT NULL,
                difficulty INTEGER
            )
            """

createWordTableQueryString = f"""
            CREATE TABLE IF NOT EXISTS {config.WORD_TABLE} (
                id INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE NOT NULL,
                content VARCHAR(50) NOT NULL,
                wordset_id INTEGER NOT NULL,
                FOREIGN KEY (wordset_id)
                    REFERENCES {config.WORDSET_TABLE}(id)
                    ON DELETE CASCADE,
                UNIQUE(content, wordset_id)
            )
            """

createGameTableQueryString = f"""
```

- main.py – program główny

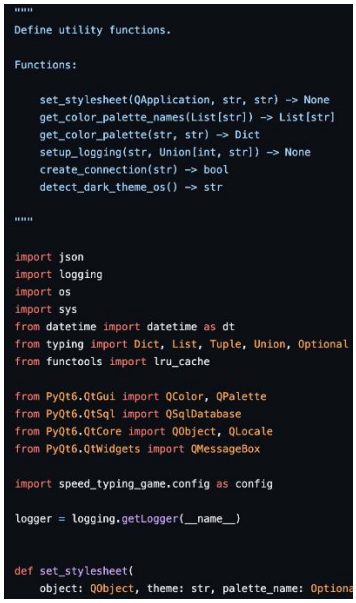
```python
"""
Setup application and run it's main loop.

Functions:

    main() -> None

"""
import logging
import os
import sys

import PyQt6.QtCore as QtCore
from PyQt6.QtGui import QIcon, QGuiApplication
from PyQt6.QtWidgets import QApplication

from speed_typing_game import config
from speed_typing_game.utils import (create_connection, get_color_palette,
                                     get_color_palette_names,
                                     get_supported_locale, set_stylesheet,
                                     setup_logging)
from speed_typing_game.views import MainWindow


def configure_app(app: QApplication) -> None:
    setup_logging("console", config.LOGGING_LEVEL)
    logger = logging.getLogger(__name__)
    if not create_connection(config.DB, config.CON_NAME):
        sys.exit(1)

    translator = QtCore.QTranslator()
    # system_locale = QtCore.QLocale.system().name()
    # locale = get_supported_locale()[0]
    QtCore.QCoreApplication.setApplicationName(config.PROJECT_NAME)
    QtCore.QCoreApplication.setOrganizationName("AGHTech")
    settings = QtCore.QSettings()
    if settings.contains("styles/theme"):
```

- models.py – klasy przechowujące stan gry i zbiory słów



```
"""
Classes:

    Wordset: contains words
    TypingGame: represents typing game state

"""

import logging
import random
import sys
import time
from collections import Counter
from functools import lru_cache
from typing import Any, Dict, List, Optional, Tuple, Union
from enum import Enum

from PyQt6.QtSql import QSqlDatabase, QSqlQuery
from PyQt6.QtCore import QAbstractListModel, QSettings, QCor

from speed_typing_game import config, database, utils


class Mode(str, Enum):
    LEARNING = QCoreApplication.translate("Enum", "Learning"
    CHALLENGE = QCoreApplication.translate("Enum", "Challeng
    ZEN = QCoreApplication.translate("Enum", "Zen")

class Wordset:
    """A named set with unique words from certain language a

    def __init__(
        self,
        name: str,
        language: str,
```

- utils.py – funkcje pomocnicze



```
"""
Define utility functions.

Functions:

    set_stylesheet(QApplication, str, str) -> None
    get_color_palette_names(List[str]) -> List[str]
    get_color_palette(str, str) -> Dict
    setup_logging(str, Union[int, str]) -> None
    create_connection(str) -> bool
    detect_dark_theme_os() -> str

"""

import json
import logging
import os
import sys
from datetime import datetime as dt
from typing import Dict, List, Tuple, Union, Optional
from functools import lru_cache

from PyQt6.QtGui import QColor, QPalette
from PyQt6.QtSql import QSqlDatabase
from PyQt6.QtCore import QObject, QLocale
from PyQt6.QtWidgets import QMessageBox

import speed_typing_game.config as config

logger = logging.getLogger(__name__)


def set_stylesheet(
    object: QObject, theme: str, palette_name: Optiona
```

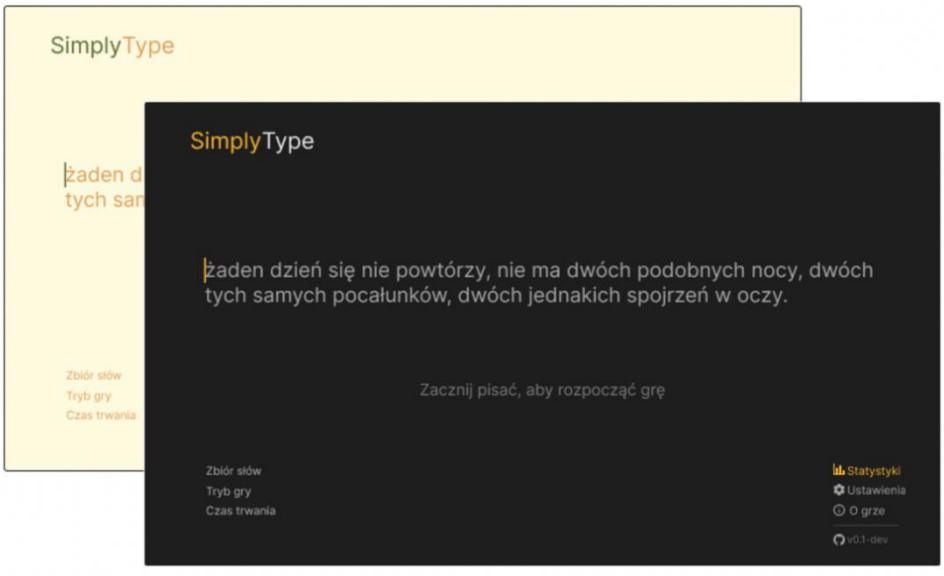
- views.py – interfejs GUI i jądro projektu

```
"""
Classes:

    MainTypingArea(QLineEdit)
    TranslucentWidget(QWidget)
    TypingHintLabel(QLabel)
    MainWindow(QWidget)

"""
import logging
import sys
from typing import Dict, List, Tuple, Optional
from collections import Counter
import time

import PyQt6.QtCore as QtCore
import PyQt6.QtGui as QtGui
from PyQt6.QtCore import QCoreApplication, QSe
from PyQt6.QtGui import (QCursor, QDesktopServi
                         QKeyEvent, QMouseEvent
from PyQt6.QtWidgets import (QButtonGroup, QFi
                             QHBoxLayout, QLabe
                             QPushButton, QVBox
from PyQt6 import QtWidgets
from PyQt6 import QtSql

from speed_typing_game import config, models, c
from speed_typing_game.models import TypingGame
from speed_typing_game.utils import set_stylesh
from speed_typing_game import main


class MainTypingArea(QLineEdit):
    """A class representing a typing area in th

    def __init__(
```

- save_wordset_to_database.py - skrypt pomocniczy

```
import sys

from speed_typing_game import config, database, models, utils

if __name__ == "__main__":
    if not utils.create_connection(config.DB, config.CON_NAME):
        sys.exit(1)
    database.delete_wordset_table()
    wordsets = [models.Wordset.from_file(i) for i in sys.argv[1:]]
    database.add_wordsets_to_database(wordsets)
```

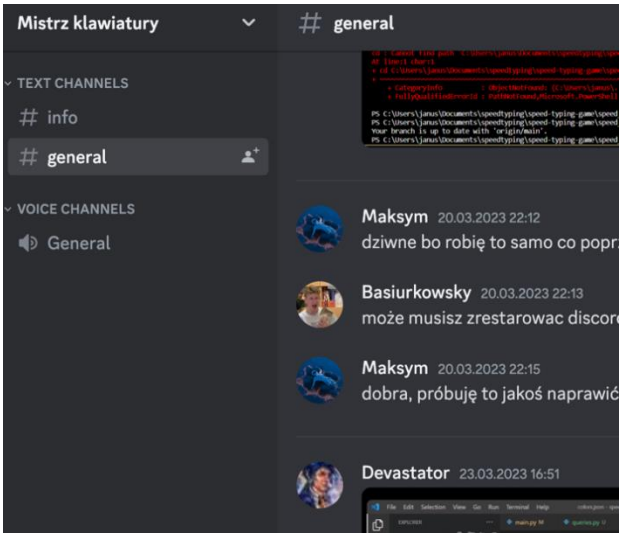
Wygląd interfejsu:



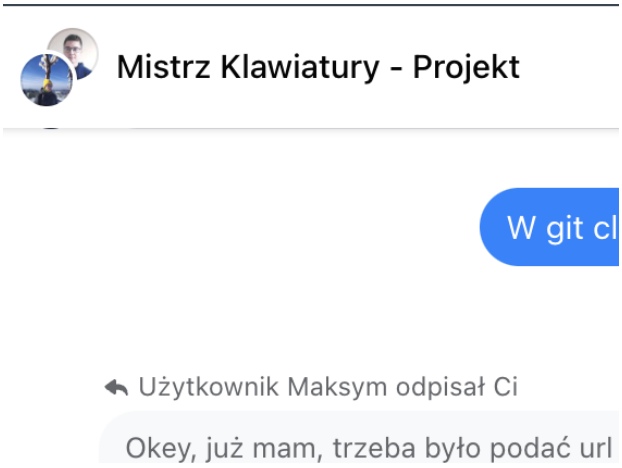
Dzięki opcji zmienianie motywu interfejsu, można balansować między jasnym, a ciemnym trybem.

4. Organizacja Pracy

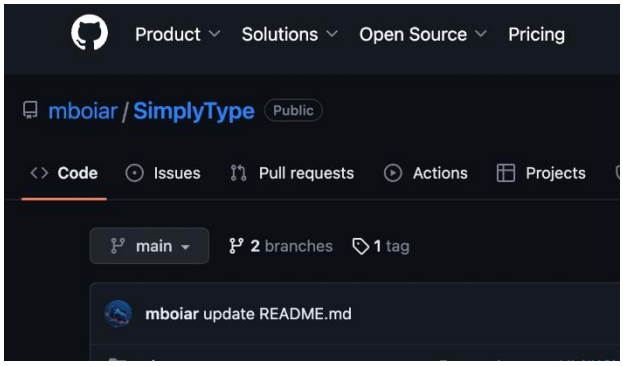
Repozytorium GitHub (publiczne) było głównym miejscem pracy nad projektem, gdzie umieszczano wszystkie pliki związane z kodem, prowadzono dokumentację oraz umieszczono bazę danych. Projekt napisano w języku Python, korzystając z środowiska PyCharm/VS Code do pisania i synchronizacji kodu. Ponadto, zespół w celu komunikowania się między sobą używał komunikatorów Discord i Messenger.



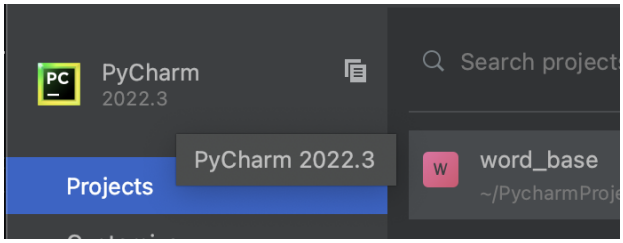
Discord



Messenger



GitHub



PyCharm

Wprowadzono podział na role i sekcje, aby usprawnić pracę w zespole:
- Maksym Boiar – senior deweloper, CEO projektu, sekcja ogólna gry
- Mateusz Basiura – sekcja bazy danych gry, dokumentacja i formalności projektu
- Filip Chałupka – sekcja statystyk i UI design

Dziennik zdarzeń:

⇒ 07.03 – Założenie projektu w GitHub Projects, utworzenie wstępnej wersji kodu i założenie serwera Discord przez Maksyma, Mateusz i Filip w tym czasie założyli grupę na Messengerze i zrobił research dotyczący wyglądu kodu projektu.
⇒ 15.03 – Maksym spersonalizował interfejs gry, dodał opcję wyboru języka i motywu interfejsu
⇒ 17.03 – Maksym utworzył interfejs GUI i walidację wpisywanego teksu na bieżąco. Stworzono interfejs GUI dla okna głównego aplikacji oraz zaimplementowano walidację tekstu wpisywanego przez użytkownika
⇒ 18.03 – Filip zaimplementował podliczanie statystyk gry
⇒ 19.03 – Mateusz utworzył wstępną bazę danych słów, wgrał ją w projekt, Maksym w tym czasie dodał rejestrację zdarzeń, przebiegu i działania programu (log) i zaimplementował rozpoczęcie/zakończenie gry.
⇒ 23.03 – Filip dodał koncepcję ustawień ekranu gry i przykładowymi przejściami pomiędzy interfejsami
⇒ 24.03 – Maksym sfaktoryzował kod programu, Mateusz wgrał ostateczną wersję bazy danych, przetłumaczoną na inne języki.
⇒ 28.03 – Mateusz dodał zbiór słów i wyrazów, Maksym dodał interakcję z bazą danych i utworzył okna wyskakujące
⇒ 10.04 – Maksym zbudował aplikację na Windows (.exe), oraz usprawnił ustawienia gry, tryby i opcję zapisu stanu gry

5. Jak to ma wyglądać?

Gra Mistrz Klawiatury jest grą z wbudowanym interfejsem graficznym GUI (graphical user interface) oraz jest aplikacją desktopową, czyli przeznaczoną tylko i wyłącznie na komputery. Gra polega na wpisywaniu tekstu z ekranu do swojej klawiatury, starając się zrobić to jak najszybciej i jak najdokładniej. Gracz wybiera poziom trudności i rodzaj tekstu, który chce wpisać, a następnie rozpoczyna grę. W trakcie gry na ekranie pojawiają się słowa lub zdania, które gracz musi przepisać na klawiaturze.

6. Projekt został utworzony przy pomocy narzędzi:

- GitHub

- PyCharm

- VS Code

- System kontroli wersji Git

- Komunikator Discord i Messenger

7. Project BackLog:

Ewolucja projektu:

- Zaprojektowanie szkieletu programu

- Utworzenie bazy zdań i fraz

- Opracowanie funkcji wypisywania tekstu na ekran, oraz opcji korygowania błędów

- Dodanie statystyk do gry

- Opracowanie graficznego interfejsu użytkownika

- Dodanie ustawień gry

Priorytety:

- doprowadzenie projektu do stanu surowego, posiadającego najważniejsze cechy, aby można było łatwo implementować i testować nowe ustawienia, opcje i możliwości gry

- utworzenie opcji poglądu statystyk gry oraz opcji zapisu stanu gry

8. Wkład wykonawców projektów:

Podczas projektu, każdy członek zespołu został przypisany do odpowiedniego zakresu prac, aby odpowiadał jego umiejętnościom.

Mateusz Basiura

Mail: matbas@student.agh.edu.pl

Zrealizowane zadania:

- utworzenie bazy danych składającą się z pojedynczych wyrazów jak i całych zdań i fraz w języku polskim i angielskim

- utworzenie sprawozdań z projektu oraz dokumentacji

- zorganizowanie grupy na Messengerze i research projektu

Aspekt	Parametry	Wkład
Role	Wymienić	Sekretarz, Researcher, Junior developer
Kodowanie	Liczba linii kodu	229 linijek kodu – słowa, frazy w bazie danych
	Funkcje (wymienić)	Baza danych
Repozytorium	Liczba commit-ów	1
	Liczba utworzonych gałęzi	2
	Gałąź (używana – nazwa)	feature
	Liczba połączonych gałęzi	1
	Liczba dni aktywności GIT	
Dokumentowanie	Liczba standup-ów	2
	Opisy na Wiki	Tak
Aktywność	Liczba zrealizowanych zadań	1
	Szacowana liczba godzin	10
	Ocena procentowego wkładu	25%

Maksym Boiar

Mail: boiar.max@gmail.com

Zrealizowane zadania:

- interfejs GUI
- komunikacja z bazą danych
- utworzenie strony na GitHubie i Discorda
- funkcjonalność gry

Aspekt	Parametry	Wkład

Role	Wymienić	CEO, Senior developer
Kodowanie	Liczba linii kodu	4840 linijki kodu
	Funkcje (wymienić)	Tryby gry, implementacja zasad, zapis/odczyt stanu, odczyt zbioru słów z pliku
Repozytorium	Liczba commit-ów	15
	Liczba utworzonych gałęzi	7
	Gałąź (używana – nazwa)	Wiele gałęzi
	Liczba połączonych gałęzi	7
	Liczba dni aktywności GIT	
Dokumentowanie	Liczba standup-ów	5
	Opisy na Wiki	Tak
Aktywność	Liczba zrealizowanych zadań	4
	Szacowana liczba godzin	80
	Ocena procentowego wkładu	60%

Filip Chałupka

Mail: chalupka@student.agh.edu.pl

Zrealizowane zadania:

- implementacja podliczania statystyk gry
- research projektu
- UI/UX Design

Aspekt	**Parametry**	**Wkład**
Role	Wymienić	Junior developer, Researcher

Kodowanie	Liczba linii kodu	4
	Funkcje (wymienić)	Statystyki gry, UI/UX Design
Repozytorium	Liczba commit-ów	1
	Liczba utworzonych gałęzi	2
	Gałąź (używana – nazwa)	statystyki
	Liczba połączonych gałęzi	1
	Liczba dni aktywności GIT	
Dokumentowanie	Liczba standup-ów	2
	Opisy na Wiki	Tak
Aktywność	Liczba zrealizowanych zadań	1
	Szacowana liczba godzin	6
	Ocena procentowego wkładu	15%

9. Podsumowanie

W trakcie projektu starano się rozdzielić zadania w taki sposób, aby każdy członek zespołu miał swoją rolę w realizacji projektu. W trakcie tworzenia programu napotkano trudności, ale udało się osiągnąć wszystkie cele i funkcjonalności. Głównym problemem była implementacja interfejsu GUI i komunikacji z bazą danych, co wymusiło obszerne wykorzystywanie poleceń logujących zmiany lokalnych zmiennych i testowanie najmniejszych obszarów kodu osobno w celu wykrycia obficie występujących błędów. Problemem było też między innym organiczna możliwość dodawania zmian przy pomocy systemu kontroli wersji Git na serwer GitHub przez Mateusza. Problem został rozwiązany poprzez wysłanie przez Mateusza, plików do zamieszczenia, do Maksyma który bez problemu umieścił owe pliki na GitHub. Kolejnym problemem była ilość kodu, która przytłaczała developerów i opóźniała ich pracę. Mimo pewnych przeszkód, prace zostały zakończone powodzeniem.