Scoring System and Administration on Wushu Championship

Group : 5

Name : Asia Illumina Lessy

Nur Iqbal Maulana

Class : 3CS1

CEP CCIT

FAKULTAS TEKNIK INDONESIA

2024

PROJECT ON

Object Oriented Programming

Developed By

1. Asia Illumina Lessy
2. Nur Iqbal Maulana

Scoring System and Administration on Wushu Championship

Batch Code : 1CC6

Start Date : September 9, 2024

End Date : October 2, 2024

Name Of Faculty : Ivan Firdaus, S. T.

Name Of Developer :

1. Nur Iqbal Maulana
2. Asia Illumina Lessy

Date Of Submisson : September 30,2023

CERTIFICATE

This is to certify that the report titled " Scoring System and Administration on Wushu Championship", embodies the original work done by Asia Illumina Lessy and Nur Iqbal Maulana. Project in partial fulfillment of their course requirement at NIIT.

Coordinator :

Ivan Firdaus, S. T.

ACKNOWLEDGEMENT

The author expresses his gratitude to Allah SWT for all the abundance of grace and mercy. His mercy and grace, and do not forget the shalawat and greetings we send to the Prophet Muhammad SAW, so that we can complete this project with the title " Scoring System and Administration on Wushu Championship" and without him we would not be able to complete this project on time. Time that has been calculated, and the author also wants to thank Mr Ivan Firdaus, S. T., as the supervisor who has provided suggestions and advice that are very helpful to the author in writing this project. Although there are many challenges and obstacles that we face in making this project, we can finally complete it. Finally, we were able to complete this project. The author realizes that this assignment is still far from perfection, and if colleagues and lecturers are willing to provide suggestions and criticism, then this assignment is not perfect. Supervisors are pleased to provide suggestions and criticism for the sake of the perfection of this project, and we as writers will be greatly helped. We, as writers, will be greatly helped by these suggestions and criticisms.

BACKGROUND

Wushu is a traditional Chinese martial art that includes two main aspects: Taolu and Sanda. Taolu involves regular practice or tao with beautiful and intricate movements, judged on technique, strength, and artistic expression. Meanwhile, Sanda focuses on standing combat with athletes using martial techniques to attack and defend, judged on the effectiveness of attacks, and throwing opponents to the ground. Wushu also has roots in traditional Chinese martial arts and has developed into an international competitive sport, blending elements of performing arts with martial arts skills.

Nowadays, Wushu is a fairly popular sport. So many kinds of competitions are made. As technology advances, Wushu activists begin to deliver their innovations, to improve the effectiveness and efficiency of the course of the competition. Scoring is one of the technologies developed to realize the goal and increase the popularity of the Wushu sport itself.

The project this time will discuss a general overview of the use and implementation of OOP in the Wushu championship. The aim is to provide a further overview and understanding of how the OOP can be applied to an event, especially a championship event.

I hope this project will provide a useful insight and facilitate an understanding of how the OOP using Python Language is estimated to be used in a Wushu Championship. May this report be useful to all those involved in the use and the development of technology, to the sports activists, as well as provide a deep understanding of the application of Object Oriented Programming.

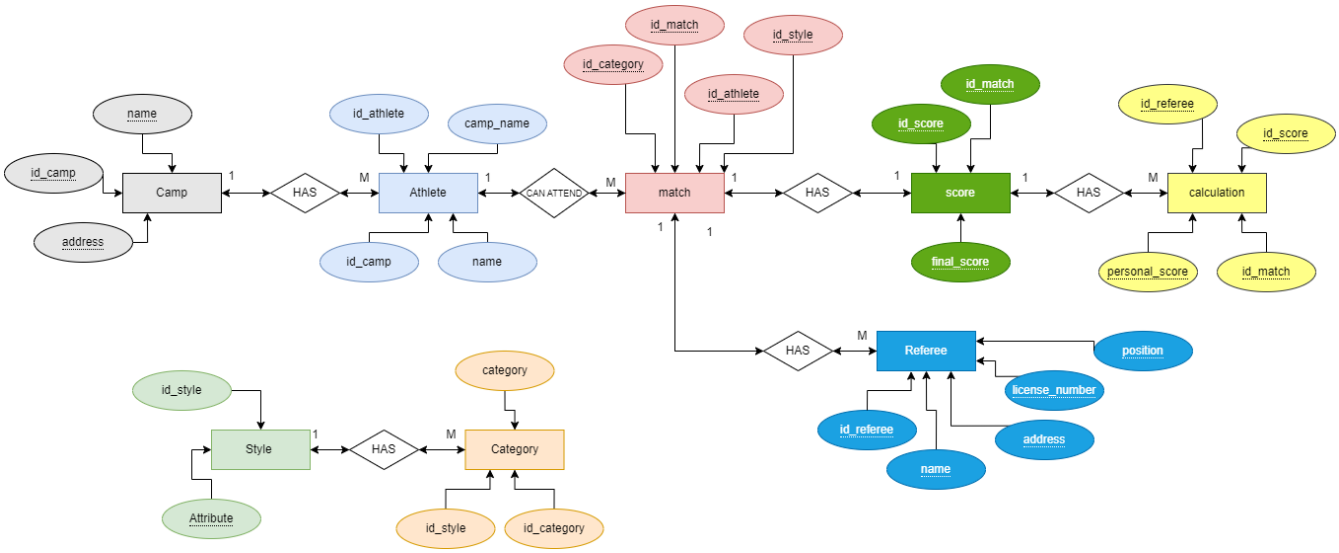
SYSTEM ANALYSIS

System Summary: Wushu championships are organised to facilitate disciplines to demonstrate their abilities and become new career paths, ranging from athletes, coaches, referees, and Professional IT teams related in championships to bureaucratic affairs. This project will briefly explain how OOP can be implemented into sport competition.

System Processes:

1. Creating the database on MySQL Server
2. Implementing OOP Principal using Python Language:
    a. Changing participant's name:
        i. Updating participat's name in database with the new (correct) name
    b. Scoring System
        i. Showing Match information based on database
        ii. Inserting Referee information, score by referee, and inserting final score to database(using procedure)
        iii. Deleting the previous information inserted to the database system such as referee information, score by referee, and final score

ENTITES RELATIONAL DIAGRAM

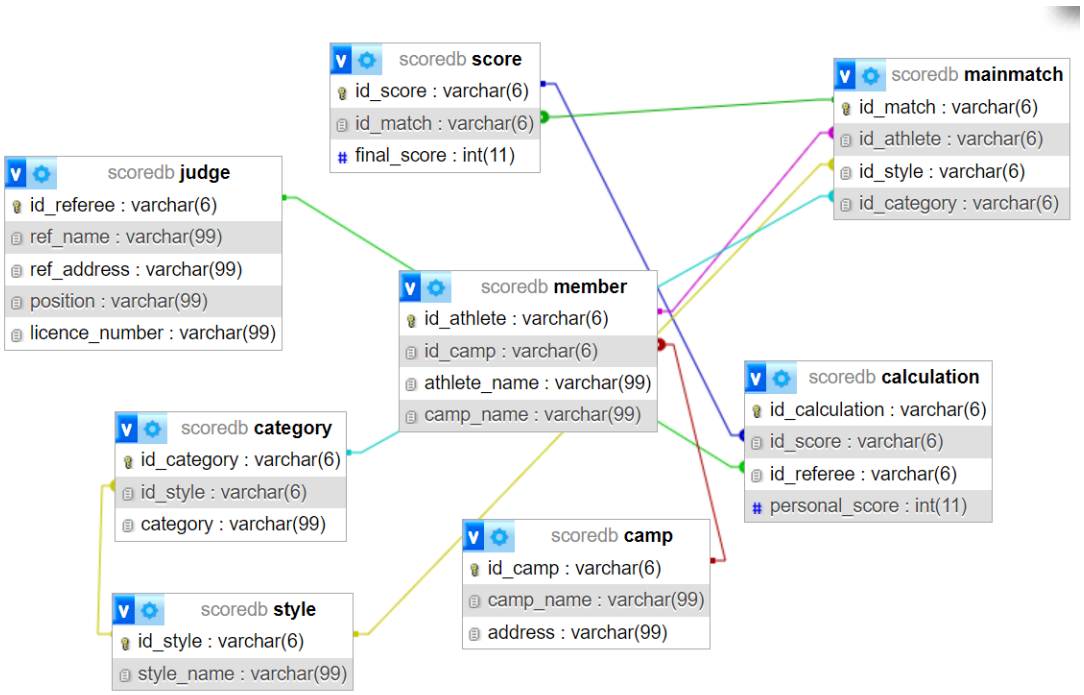


ENTITIES

Number of Entities : 9

1. Camp
2. Member
3. Style
4. Category
5. Match
6. Referee
7. Score
8. Calculation of Score

DATABASE DIAGRAM



DATABASE CONTENTS

id_camp	camp_name	address
S01	WUSHU UDAYANA	DENPASAR
S02	BLUE SPIDER	TANGERANG
S03	8 NAGA	BANDUNG
S04	SWI LABA-LABA SAKTI	JAKARTA
S05	ALMA WUSHU	SALATIGA
S06	YAYASAN INTI BAYANGAN	JAKARTA
S07	UNNES	JAKARTA
S08	TIAN EN	SURAKARTA
S09	HARMONY WUSHU INDONESIA	BOGOR
S10	ACMA	BOGOR
S11	WUXIA ACADEMY	SIDOARJO

Table of Camps

id_athlete	id_camp	athlete_name	camp_name
7000	S06	enrico	INTI BAYANGAN
7001	S06	NIMAS	INTI BAYANGAN
7002	S01	NI LUH RATNA	UDAYANA
7003	S01	KOMANG AGUS	UDAYANA
7004	S09	ZOURA	HARMONY WUSHU
7005	S09	GOVIN	HARMONY WUSHU
7006	S10	ALICIA	ACMA
7007	S10	JACKY	ACMA
7008	S05	CAROLINE	ALMA WUSHU
7009	S05	KEZIA	ALMA WUSHU
7010	S08	BRIGITA	TIAN EN
7011	S08	AXELIA	TIAN EN

Table of Member (Participants)

id_referee	ref_name	ref_address	position	licence_number
W101	Agus	Semarang	Side Judge	LIS-NAS2022
W102	Asep	Bandung	Side Judge	LIS-DJB2019
W103	Jeje	Jakarta	Juri Sisi	LIS-INT2022
W104	Darmaji	Yogyakarta	Chairman	LIS-NAS2020
W105	Edward	Surabaya	Side Judge	LIS-NAS2022
W106	Jo	Depok	Side Judge	LIS-DJB2020
W107	Johar	Jakarta	Vice Chairman	LIS-NAS2022
W108	Lena	Jakarta	Side Judge	LIS-NAS2020
W109	Nyoman	Bali	Chief of Field	LIS-NAS2021

Table of Judges

id_style	style_name
J01	DRUNKEN MASTER
J02	MONKEY KING
J03	CHANG QUAN
J04	JIAN SHU
J05	DAO SHU
J06	QIANG SHU
J07	GUN SHU
J08	CHUJI NAN QUAN
J09	MOK YAN JONG
J10	CHUJI NAN GUN
J11	NAN QUAN
J12	NAN DAO
J13	NAN GUN
J14	TAIJI QUAN
J15	TAIJI JIAN
J16	BAAT JARM DAO
J17	DUILIAN
J18	JITI EMPTY HANDS

Table of Styles

id_category	id_style	category
D010	J01	FREESTYLE
D011	J02	WEAPON
D012	J02	EMPTY HAND
D013	J03	SD
D014	J03	SMP
D015	J03	SMA
D016	J04	SD
D017	J04	SMP
D018	J04	SMA
D019	J04	MAHASISWA
D020	J06	SMA
D021	J06	FREESTYLE
D022	J07	SMA
D023	J07	MAHASISWA
D100	J10	SD
D111	J11	SMP
D112	J11	SMA
D113	J11	FREESTYLE

Table of Categories

id_match	id_athlete	id_style	id_category
A001	7005	J01	D010
A002	7003	J01	D010
A003	7000	J01	D011
A004	7007	J01	D012
A005	7005	J03	D014
A006	7002	J03	D022
A007	7004	J03	D022
A008	7000	J04	D014
A009	7004	J04	D111
A010	7008	J04	D017
A011	7001	J06	D117
A012	7005	J06	D012
A013	7001	J07	D023
A014	7001	J11	D115
A015	7002	J11	D113
A016	7005	J11	D113
A017	7003	J12	D114
A018	7003	J13	D119

Table of Main Match

id_calculation	id_score	id_referee	personal_score
A001_1	N001	W101	98
A001_2	N001	W102	98
A001_3	N001	W103	98
A001_4	N001	W104	98
A001_5	N001	W105	90
A002_1	N002	W101	97
A002_2	N002	W102	96
A002_3	N002	W103	95
A002_4	N002	W104	94
A002_5	N002	W105	93

Table of Calculation (Used)

id_score	id_match	final_score
N001	A001	96
N002	A002	95

Table of Score (Used)

SCRIPT

```
import tkinter as tk
from tkinter import ttk
import mysql.connector

# Connect to the database
db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="scoredb"
)

# Create a GUI window
window = tk.Tk()
window.title("Wushu Championship")

# Create a frame for the main menu
main_frame = ttk.Frame(window, padding="10")
main_frame.pack(fill="both", expand=True)

# Create a label and button for the main menu
label = ttk.Label(main_frame, text="Welcome To Wushu Championship!")
label.pack(pady=10)

button_frame = ttk.Frame(main_frame)
button_frame.pack(pady=10)

change_name_button = ttk.Button(button_frame, text="Change Participant's
Name", command=lambda: change_name())
change_name_button.pack(side=tk.LEFT, padx=10)

evaluate_button = ttk.Button(button_frame, text="Scoring System",
command=lambda: evaluate())
evaluate_button.pack(side=tk.LEFT, padx=10)

reset_button = ttk.Button(button_frame, text="Reset Data", command=lambda:
reset_data())
reset_button.pack(side=tk.LEFT, padx=10)

def reset_data():
    # Reset final score di tabel score
    cursor = db.cursor()
    cursor.execute("UPDATE score SET final_score = NULL")
    db.commit()

    # Hapus semua data di tabel calculation
    cursor.execute("TRUNCATE TABLE calculation")
    db.commit()

    # Tampilkan pesan konfirmasi
```

```
    confirm_window = tk.Toplevel(window)
    confirm_window.title("Data Reset")

    confirm_label = ttk.Label(confirm_window, text="Process Done!")
    confirm_label.pack(pady=10)

    confirm_button = ttk.Button(confirm_window, text="OK", command=lambda:
confirm_window.destroy())
    confirm_button.pack(pady=10)


def change_name():
    # Create a new window for changing athlete names
    change_name_window = tk.Toplevel(window)
    change_name_window.title("Change Participant's Name")

    # Create a frame for the athlete list
    athlete_frame = ttk.Frame(change_name_window, padding="10")
    athlete_frame.pack(fill="both", expand=True)

    # Create a label and listbox for the athlete list
    label = ttk.Label(athlete_frame, text="Participants List:")
    label.pack(pady=10)

    # Create a scrollbar for the listbox
    scrollbar = tk.Scrollbar(athlete_frame)
    scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

    # Create a listbox with scrollbar
    athlete_listbox = tk.Listbox(
        athlete_frame, width=30, yscrollcommand=scrollbar.set
    )
    athlete_listbox.pack(pady=10, fill="both", expand=True)

    # Configure the scrollbar
    scrollbar.config(command=athlete_listbox.yview)

    # Populate the athlete listbox
    cursor = db.cursor()
    cursor.execute("SELECT id_athlete, athlete_name FROM member")
    athletes = cursor.fetchall()
    for athlete in athletes:
        athlete_listbox.insert(tk.END, f"{athlete[0]} - {athlete[1]}")

    # Create a frame for the new name entry
    new_name_frame = ttk.Frame(change_name_window, padding="10")
    new_name_frame.pack(fill="both", expand=True)

    # Create a label and entry for the new name
    label = ttk.Label(new_name_frame, text="New Name:")
    label.pack(pady=10)

    new_name_entry = ttk.Entry(new_name_frame, width=30)
    new_name_entry.pack(pady=10)

    # Create a button to update the athlete name
    update_button = ttk.Button(
        new_name_frame,
        text="Update",
        command=lambda: update_athlete_name(athlete_listbox, new_name_entry),
    )
```

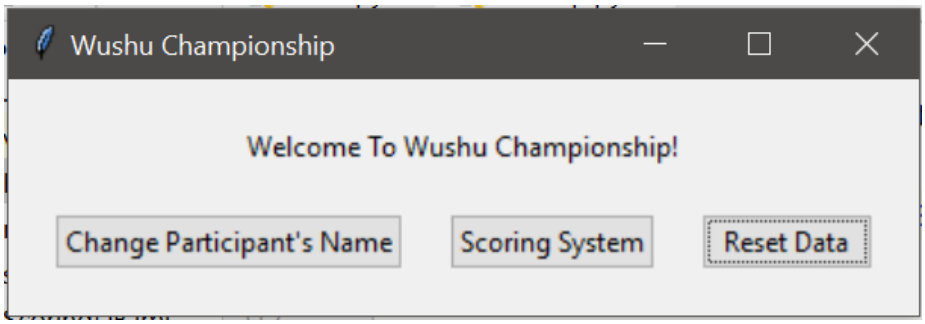
```
    update_button.pack(pady=10)


def update_athlete_name(athlete_listbox, new_name_entry):
    # Get the selected athlete and new name
    selected_athlete = athlete_listbox.get(athlete_listbox.curselection())
    new_name = new_name_entry.get()

    # Update the athlete name in the database
    cursor = db.cursor()
    cursor.execute(
        "UPDATE member SET athlete_name = %s WHERE id_athlete = %s",
        (new_name, selected_athlete.split(" - ")[0]),
    )
    db.commit()

    # Close the change name window
    change_name_window.destroy()


def evaluate():
    global evaluate_window, mainmatch_frame
    evaluate_window = tk.Toplevel(window)
    evaluate_window.title("Scoring System")

    # Create a frame for the mainmatch information
    mainmatch_frame = ttk.Frame(evaluate_window, padding="10")
    mainmatch_frame.pack(fill="both", expand=True)

    # Create a label for the mainmatch information
    label = ttk.Label(mainmatch_frame, text="Main Match Information:")
    label.pack(pady=10)

    # Get the first mainmatch information
    cursor = db.cursor()
    cursor.execute(
        """
        SELECT mainmatch.id_match, mainmatch.id_athlete, member.athlete_name,
        member.id_camp, camp.camp_name, mainmatch.id_style, style.style_name
        FROM mainmatch
        JOIN member ON mainmatch.id_athlete = member.id_athlete
        JOIN camp ON member.id_camp = camp.id_camp
        JOIN style ON mainmatch.id_style = style.id_style
        WHERE mainmatch.id_match = 'A001'
        """
    )
    mainmatch_info = cursor.fetchone()

    mainmatch_label = ttk.Label(
        mainmatch_frame,
        text=f"Match ID: {mainmatch_info[0]}, Athlete: {mainmatch_info[2]}, "
             f"Sasana: {mainmatch_info[4]}, Jurus: {mainmatch_info[6]}"
    )
    mainmatch_label.pack(pady=10)

    # Create a frame for the referee entries
    referee_frame = ttk.Frame(evaluate_window, padding="10")
    referee_frame.pack(fill="both", expand=True)

    # Create labels and entries for the referees
    referee_labels = []
```

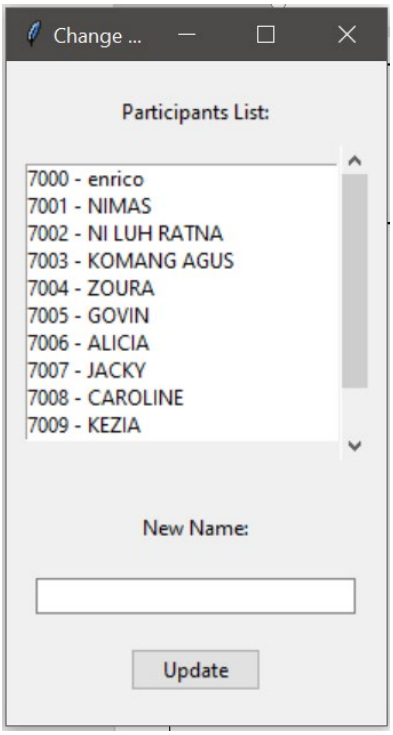
```
    referee_entries = []
    for i in range(5):
        label = ttk.Label(referee_frame, text=f"Wasit {i+1}:")
        label.pack(pady=5)
        referee_labels.append(label)

        entry = ttk.Entry(referee_frame, width=30)
        entry.pack(pady=5)
        referee_entries.append(entry)

    # Create a button to submit the referee scores
    submit_button = ttk.Button(
        referee_frame,
        text="Submit",
        command=lambda: submit_referee(mainmatch_info, referee_entries,
referee_frame),
    )
    submit_button.pack(pady=10)


def submit_referee(mainmatch_info, referee_entries, referee_frame):
    referee_ids = [entry.get() for entry in referee_entries]

    referee_frame.destroy()

    score_frame = ttk.Frame(evaluate_window, padding="10")
    score_frame.pack(fill="both", expand=True)

    score_labels = []
    score_entries = []
    for i in range(5):
        label = ttk.Label(score_frame, text=f"Score {i+1}:")
        label.pack(pady=5)
        score_labels.append(label)

        entry = ttk.Entry(score_frame, width=30)
        entry.pack(pady=5)
        score_entries.append(entry)

    submit_score_button = ttk.Button(
        score_frame,
        text="Submit",
        command=lambda: submit_scores(mainmatch_info, referee_ids,
score_entries, score_frame),
    )
    submit_score_button.pack(pady=10)


def submit_scores(mainmatch_info, referee_ids, score_entries, score_frame):
    scores = [entry.get() for entry in score_entries]

    cursor = db.cursor()
    cursor.execute("SELECT id_score FROM score WHERE id_match = %s",
(mainmatch_info[0],))
    id_score = cursor.fetchone()[0]

    for i, score in enumerate(scores):
        cursor.execute(
            "INSERT INTO calculation (id_calculation, id_score, id_referee,
personal_score) "
            "VALUES (%s, %s, %s, %s)",
```

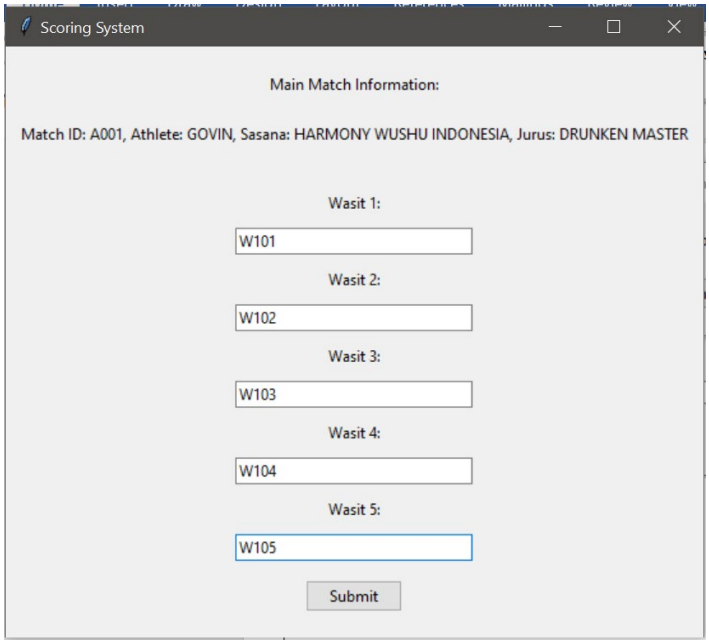
```
            (f"{mainmatch_info[0]}_{i+1}", id_score, referee_ids[i], score),
        )
    db.commit()

    average_score = sum(map(float, scores)) / len(scores)

    cursor.execute(
        "UPDATE score SET final_score = %s WHERE id_match = %s",
        (average_score, mainmatch_info[0]),
    )
    db.commit()

    score_frame.destroy()

    global result_frame
    result_frame = ttk.Frame(evaluate_window, padding="10")
    result_frame.pack(fill="both", expand=True)

    result_label = ttk.Label(result_frame, text=f"Final Score:
{average_score:.2f}")
    result_label.pack(pady=10)

    mainmatch_label = ttk.Label(
        result_frame,
        text=f"Match ID: {mainmatch_info[0]}, Athlete: {mainmatch_info[2]}, "
             f"Sasana: {mainmatch_info[4]}, Jurus: {mainmatch_info[6]}"
    )
    mainmatch_label.pack(pady=10)

    next_match_button = ttk.Button(
        result_frame,
        text="Next Match",
        command=lambda: next_match(mainmatch_info, result_frame),
    )
    next_match_button.pack(pady=10)


def next_match(mainmatch_info, result_frame):
    result_frame.destroy()

    global mainmatch_frame

    for widget in mainmatch_frame.winfo_children():
        widget.destroy()

    cursor = db.cursor()
    cursor.execute(
        """
        SELECT mainmatch.id_match, mainmatch.id_athlete, member.athlete_name,
        member.id_camp, camp.camp_name, mainmatch.id_style, style.style_name
        FROM mainmatch
        JOIN member ON mainmatch.id_athlete = member.id_athlete
        JOIN camp ON member.id_camp = camp.id_camp
        JOIN style ON mainmatch.id_style = style.id_style
        WHERE mainmatch.id_match > %s
        ORDER BY mainmatch.id_match LIMIT 1
        """,
        (mainmatch_info[0],),
    )
    next_mainmatch_info = cursor.fetchone()
```

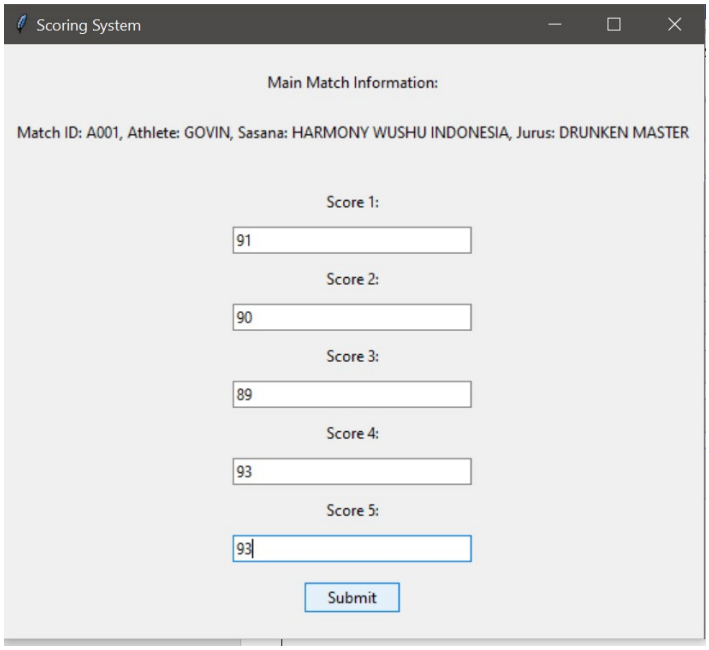
```
    next_mainmatch_label = ttk.Label(
        mainmatch_frame,
        text=f"Match ID: {next_mainmatch_info[0]}, Athlete:
{next_mainmatch_info[2]}, "
             f"Sasana: {next_mainmatch_info[4]}, Jurus:
{next_mainmatch_info[6]}"
    )
    next_mainmatch_label.pack(pady=10)

    # Create a frame for the referee entries
    referee_frame = ttk.Frame(mainmatch_frame, padding="10")
    referee_frame.pack(fill="both", expand=True)

    # Create labels and entries for the referees
    referee_labels = []
    referee_entries = []
    for i in range(5):
        label = ttk.Label(referee_frame, text=f"Wasit {i+1}:")
        label.pack(pady=5)
        referee_labels.append(label)

        entry = ttk.Entry(referee_frame, width=30)
        entry.pack(pady=5)
        referee_entries.append(entry)

    # Create a button to submit the referee scores
    submit_button = ttk.Button(
        referee_frame,
        text="Submit",
        command=lambda: submit_referee(next_mainmatch_info, referee_entries,
referee_frame),
    )
    submit_button.pack(pady=10)


# Start the GUI loop
window.mainloop()
```

DOCUMENTATION

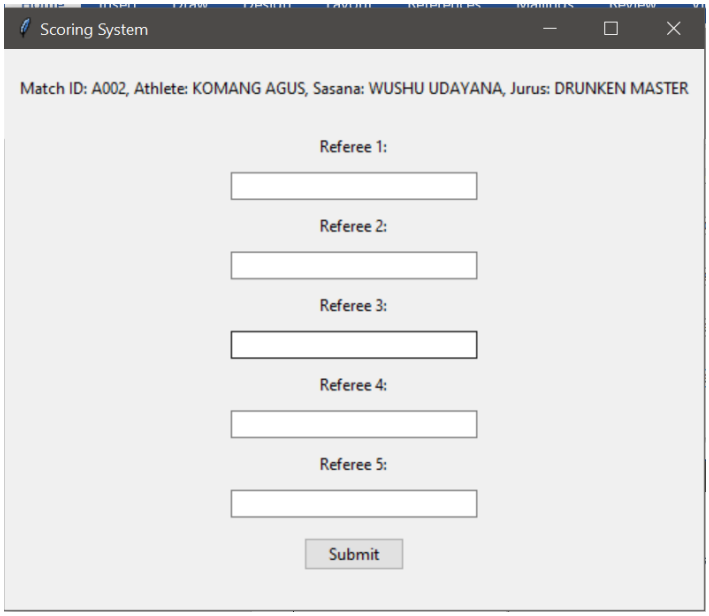


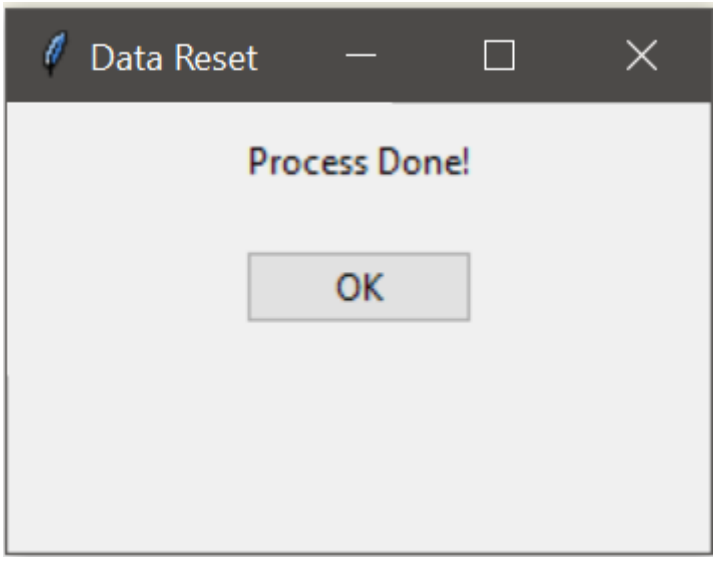



Scoring System
Main Match Information:
Match ID: A001, Athlete: GOVIN, Sasana: HARMONY WUSHU INDONESIA, Jurus: DRUNKEN MASTER
Wasit 1:
W101
Wasit 2:
W102
Wasit 3:
W103
Wasit 4:
W104
Wasit 5:
W105
Submit


Scoring System
Main Match Information:
Match ID: A001, Athlete: GOVIN, Sasana: HARMONY WUSHU INDONESIA, Jurus: DRUNKEN MASTER
Score 1:
91
Score 2:
90
Score 3:
89
Score 4:
93
Score 5:
93
Submit


Scoring System
Main Match Information:
Match ID: A001, Athlete: GOVIN, Sasana: HARMONY WUSHU INDONESIA, Jurus: DRUNKEN MASTER
Final Score: 91.20
Match ID: A001, Athlete: GOVIN, Sasana: HARMONY WUSHU INDONESIA, Jurus: DRUNKEN MASTER
Next Match


Scoring System
Match ID: A002, Athlete: KOMANG AGUS, Sasana: WUSHU UDAYANA, Jurus: DRUNKEN MASTER
Referee 1:
Referee 2:
Referee 3:
Referee 4:
Referee 5:
Submit


Data Reset
Process Done!
OK

SYSTEM REQUIREMENT

Hardware:

1. Lenovo Ideapad Slim 3

Software:

1. XAMPP
2. MySQL
3. Intellij IDEA
4. Visual Studio Code
5. Microsoft Word

FILE PROJECT DETAILS	
Group 5 Paper.PDF	Paper File
PPT Group 5.PDF	Presentation File