

**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.

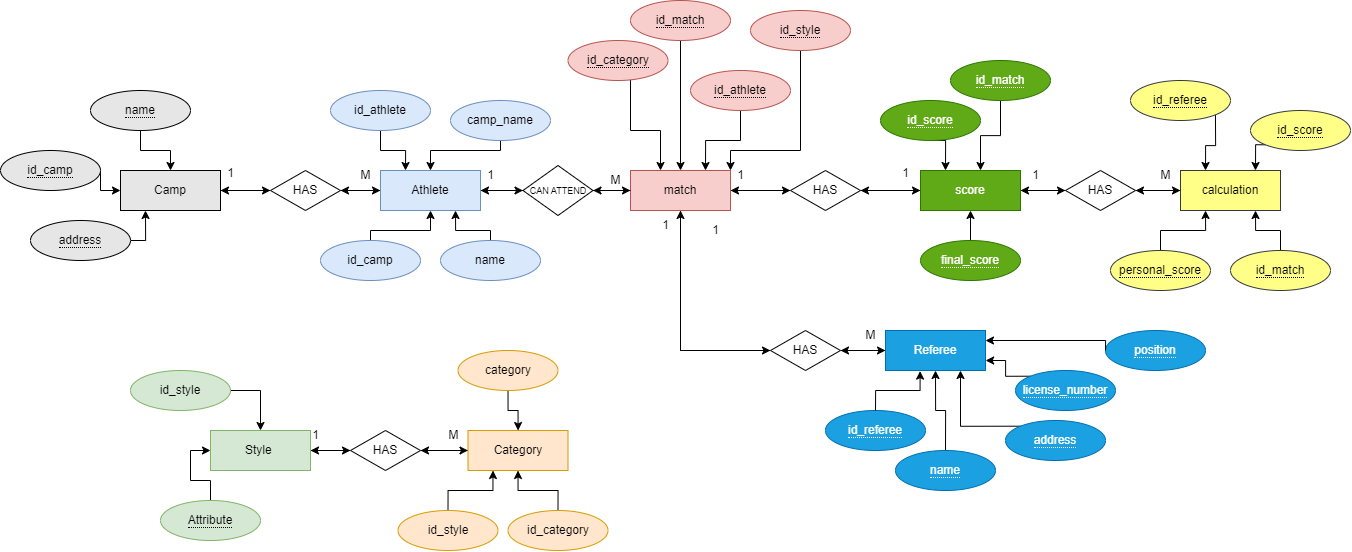
**SYSTEM ANALYSIS**

**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:
   1. Changing participant’s name:
      1. Updating participat’s name in database with the new (correct) name
   2. Scoring System
      1. Showing Match information based on database
      2. Inserting Referee information, score by referee, and inserting final score to database(using procedure)
      3. Deleting the previous information inserted to the database system such as referee information, score by referee, and final score

**ENTITES RELATIONAL DIAGRAM**

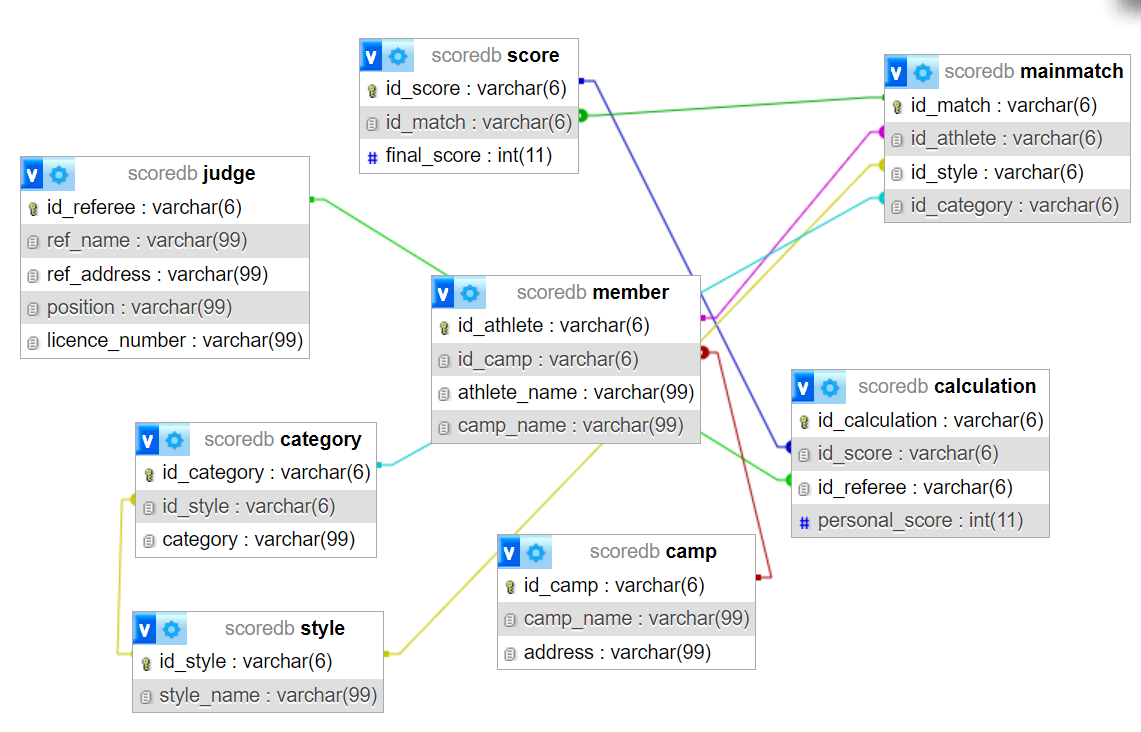
****

**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**



**TABLE OF ATTRIBUTES**

**DATABASE CONTENTS**

****

Table of Camps



Table of Member (Participants)



Table of Judges



Table of Styles

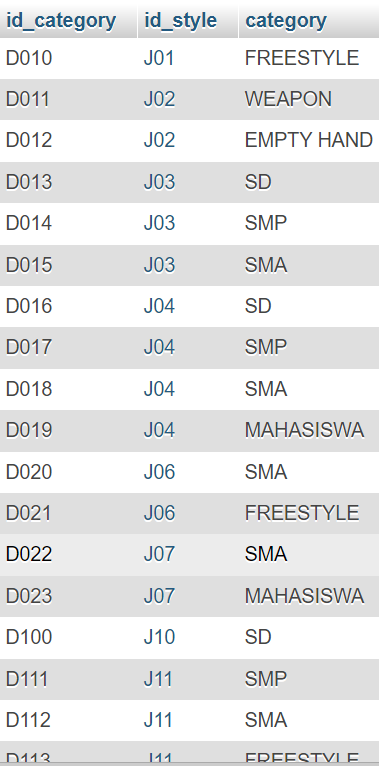


Table of Categories

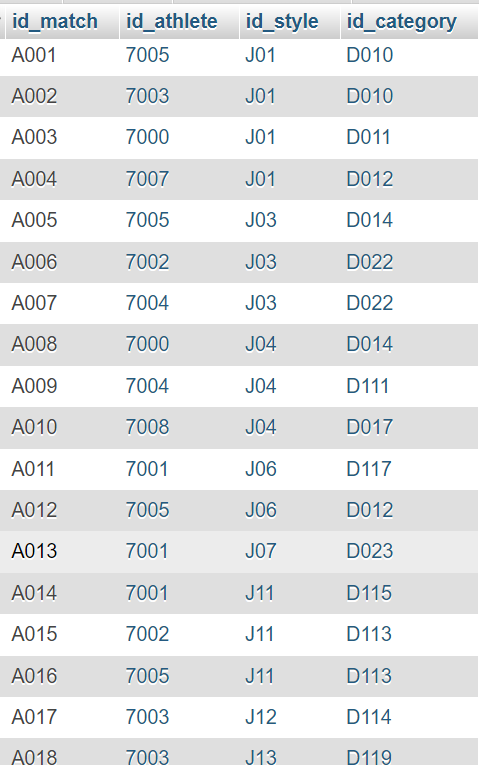


Table of Main Match

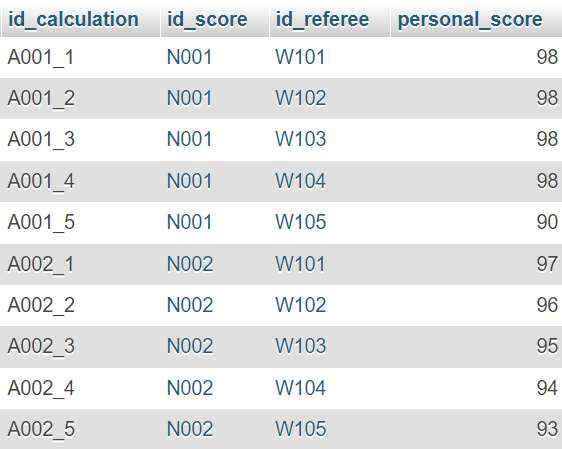


Table of Calculation (Used)

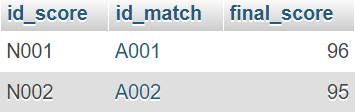
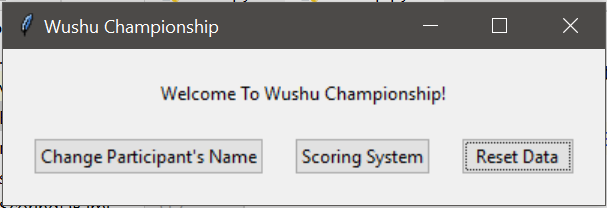


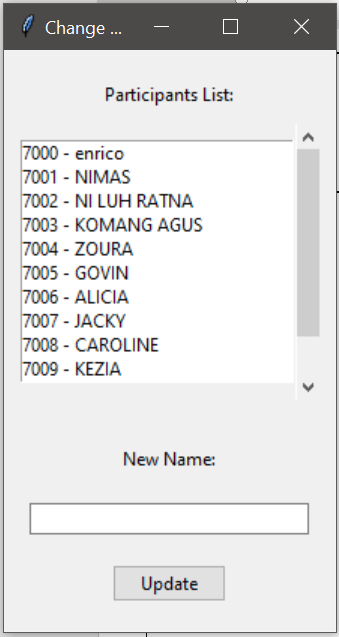
Table of Score (Used)

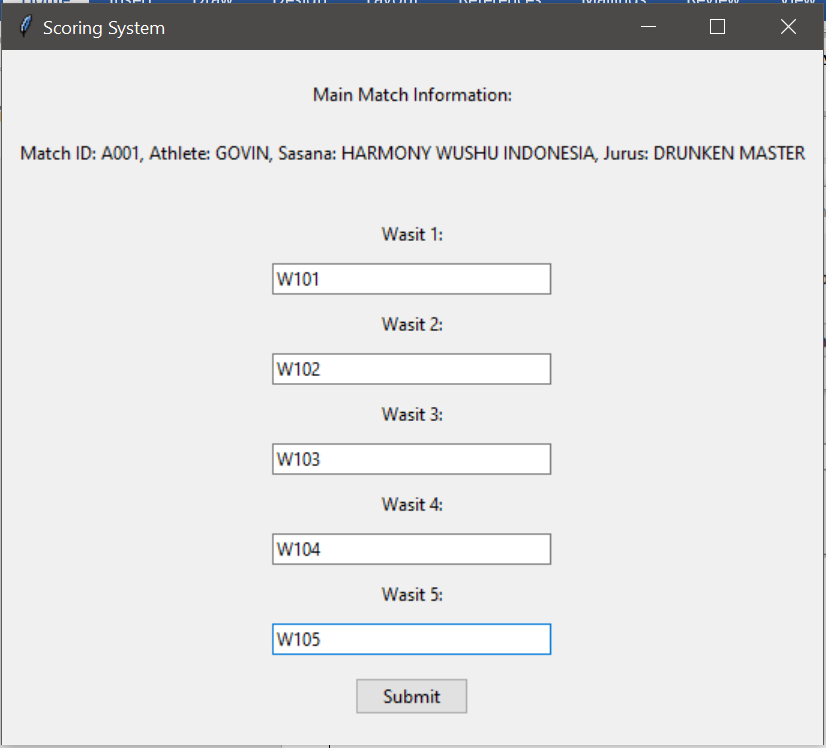
**SCRIPT**

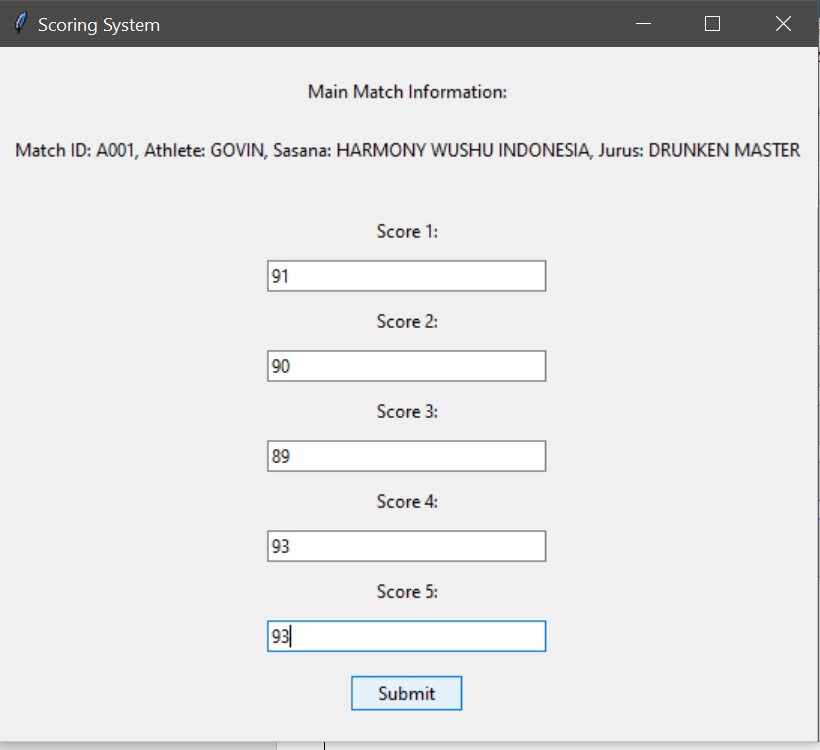
|  |
| --- |
| import tkinter as tk from tkinter import ttk import mysql.connector  *# Connect to the database* db = mysql.connector.connect(  host="localhost",  user="root",  password="",  database="scoredb" )  *# Create a GUI window* window = tk.Tk() window.title("Wushu Championship")  *# Create a 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**

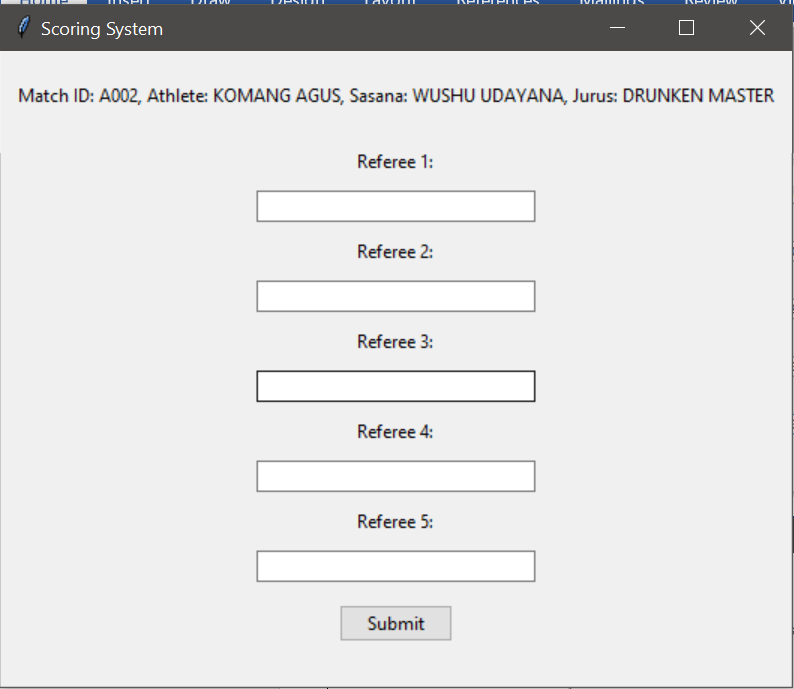


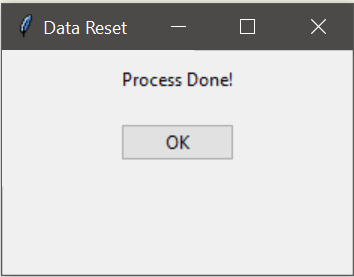












**SYSTEM REQUIREMENT**

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 |