

21, BAGRAM GAME

INTERDISCIPLINARY PROJECT

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

NAIDU SAI SRINIVAS
(Reg. No – 41110845)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

CATEGORY - 1 UNIVERSITY BY UGC

Accredited "A++" by NAAC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119

APRIL - 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **NAIDU SAI SRINIVAS (41110845)** who carried out the Project entitled “**21, BAGRAM GAME**” under my supervision from January 2024 to April 2024.

Internal Guide

DR. A. DEEPA, B.E., Ph.D.

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

Submitted for Interdisciplinary Viva Voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **NAIDU SAI SRINIVAS (Reg. No- 41110845)**, hereby declare that the Project Report entitled “**21, BAGRAM GAME**” done by me under the guidance of **DR. A. DEEPA, B.E., Ph.D.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE:

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph. D., Dean**, School of Computing, and **Dr. L. Lakshmanan, M.E., Ph.D. Head of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **DR. A. DEEPA, B.E., Ph.D.** for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

"21, Bagram" is a classic game where players take turns counting up from 1 to 21, with the player who calls "21" being eliminated. This simple yet engaging game can be implemented using the Python programming language, with various versions possible. This project focuses on a specific implementation of "21, Bagram" between a player and the computer. The player can choose to start first or second, and a list of numbers is provided before the player's turn for convenience. If consecutive numbers are not given as input, the player is disqualified. The game continues until a player is forced to call "21," resulting in their elimination. To enhance the implementation, SQL is utilized to manage and store game data. This integration allows for tracking player statistics, game outcomes, and other relevant information, enhancing the overall gaming experience. Through this combination of Python and SQL, "21, Bagram" becomes not just a fun game but also an opportunity for players to engage with data management concepts in a practical context. Twenty plus one is a game which progresses by counting up 1 to 21, with the player who calls "21" is eliminated. It can be played between any number of players. Implementation This is a simple 21 number game using Python programming language. The game illustrated here is between the player and the computer. There can be many variations in the game. The player can choose to start first or second. The list of numbers is shown before the Player takes his turn so that it becomes convenient. If consecutive numbers are not given in input then the player is automatically disqualified. The player loses if he gets the chance to call 21 and wins otherwise.

TRAINING CERTIFICATE



Certificate of Completion

Is Awarded to

Naidu Sai Srinivas

Upon successfully completed the Bootcamp Training on SQL and Python for 40
hrs with a Mini Project in 21 Bagram Game
from 05-Feb -2024 to 26-Mar -2024



Transforming the skill landscape

A handwritten signature in black ink.

Mr. Nikhil Barshikar

Managing Director
IMARTICUS LEARNING

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	ix
1	INTRODUCTION	1
2	LITERATURE SURVEY	
	2.1 Review on existing system	2
	2.2 Inferences and challenges in existing system	2
3	ANALYSIS AND DESIGN OF PROPOSED SYSTEM	
	3.1 Necessity for proposed system	4
	3.2 Hardware and software requirements	4
	3.2.1 Aim	5
	3.2.2 Scope	6
	3.3 Architecture diagram	7
	IMPLEMENTATION OF PROPOSED SYSTEM	
4	4.1 Setting up the database	8
	4.2 Python implementation	9
	4.3 Advantages and disadvantages	10
	4.4 Coding	11
	RESULTS AND DISCUSSIONS	
	5.1 Screenshots	15
5	CONCLUSION AND FUTURE ENHANCEMENT	19
6	REFERENCES	20

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
3.3	Architecture	7
4.2.1	Creating database	8
4.2.2	Creating game table	9
4.3.1	Package Installer for Python	9
4.3.2	Python-SQL connection	10
5.1.1	Sample Output 1	15
5.1.2	Sample Output 2	15
5.1.3	Sample Output 3	16
5.1.4	Sample Output 4	16
5.1.5	Sample Output 5	17

CHAPTER 1

INTRODUCTION

Welcome to the world of "21, Bagram," a captivating game of strategy, cunning, and quick thinking. In this introduction, we'll delve into the essence of the game, its rules, and the excitement it brings to players of all ages.

"21, Bagram" is a game deeply rooted in the art of counting and strategy. The premise is simple: players take turns counting aloud from 1 to 21, with each player allowed to say up to three consecutive numbers. However, there's a catch - the player who calls out "21" is eliminated from the game. Sounds easy, right? Think again. The simplicity of the rules belies the complexity of the strategies involved. Every move must be carefully calculated to avoid being the one to utter the dreaded "21." Do you play it safe and stick to lower numbers, or do you risk it all and aim for a higher count, hoping your opponents will falter before you do? "21, Bagram" is a game of psychological warfare as much as it is about numerical prowess. You must bluff, strategize, and anticipate your opponents' moves to emerge victorious. It's a test of wit and nerve, where every decision matters and every number counts.

Whether you're playing against friends, family, or formidable AI opponents, "21, Bagram" promises endless hours of entertainment and intellectual stimulation. So gather your allies, sharpen your mind, and prepare for the ultimate counting showdown. Are you ready to take on the challenge of "21, Bagram"? Twenty plus one is a game which progresses by counting up 1 to 21, with the player who calls "21" is eliminated. It can be played between any number of players. Implementation This is a simple 21 number game using Python programming language. The game illustrated here is between the player and the computer. There can be many variations in the game. The player can choose to start first or second. The list of numbers is shown before the Player takes his turn so that it becomes convenient. If consecutive numbers are not given in input then the player is automatically disqualified. The player loses if he gets the chance to call 21 and wins otherwise.

CHAPTER 2

LITERATURE SURVEY

A literature survey for the "21 Number Game" (or "Bagram Game") could involve researching various aspects of the game, including its history, variations, implementations, and potential applications.

2.1 Review on Existing System:

The existing system for the "21, Bagram" game presents a straightforward implementation of the classic counting game. Here are some key aspects of the current system:

- Simple Implementation: The game is implemented using the Python programming language, making it accessible to a wide range of players. The simplicity of the code allows for easy understanding and quick deployment of the game.
- Player vs. Computer: The current implementation features a game between a human player and the computer. This provides a single-player experience where the player can challenge their counting skills against an AI opponent.
- Basic Rules: The rules of the game are clearly defined, with players taking turns counting from 1 to 21. The player who says "21" is eliminated from the game. This straightforward rule set ensures that gameplay is intuitive and easy to grasp.
- Limited Variation: While the basic rules are implemented effectively, the current system lacks significant variation or complexity. There is potential to introduce additional features, such as different game modes, difficulty levels, or multiplayer options, to enhance replayability and engagement.

2.2 Inferences and Challenges in Existing System:

Inferences:

Simplicity: The existing system prioritizes simplicity, making it accessible to a wide audience. The straightforward implementation allows players to quickly understand the rules and mechanics of the game, facilitating easy engagement.

Basic Gameplay Mechanics: The core gameplay mechanics of counting from 1 to 21 and avoiding saying "21" are effectively implemented. This ensures that the game retains its classic charm and provides a nostalgic experience for players familiar with the traditional version.

Player vs. Computer Interaction: The inclusion of a player versus computer mode enables solo play, allowing individuals to challenge themselves against an AI opponent. This adds replay value to the game and provides entertainment for those without other players available.

Challenges:

Limited Variation: One of the primary challenges in the existing system is the lack of variation or depth in gameplay. With only a player versus computer mode and no additional features or game modes, the replayability of the game may be limited over time.

Absence of Data Persistence: The absence of data persistence, such as saving game states or player statistics, poses a challenge in terms of tracking progress and providing a more personalized experience for players. Integrating a database system to address this challenge could enhance the overall gaming experience.

User Interface Improvements: Another challenge lies in enhancing the user interface (UI) to make the game more visually appealing and user-friendly. Improving UI elements.

CHAPTER 3

ANALYSIS AND DESIGN OF PROPOSED SYSTEM

3.1 Necessity for Proposed System:

- **Enhanced Gameplay Features:** Introducing additional gameplay features such as different game modes, difficulty levels, and multiplayer options can significantly enhance the depth and replayability of the game. These features cater to a broader range of players and provide more variety in gameplay experiences.
- **Improved User Experience:** A proposed system can focus on improving the user interface (UI) and overall user experience (UX) of the game. Clearer prompts, interactive elements, visual enhancements, and intuitive controls can make the game more engaging and enjoyable for players, leading to increased player satisfaction and retention.
- **Data Persistence and Player Progression:** Implementing data persistence through a database system allows for features such as saving game states, tracking player statistics, and recording high scores. This enhances player progression, provides a sense of achievement, and encourages continued engagement with the game over time.

3.2 Hardware and Software Requirements:

Hardware Requirements:

Computer: A desktop or laptop computer capable of running the chosen programming language (e.g., Python) and any necessary development environments.

Processor: A modern processor with sufficient processing power to execute the game logic and AI algorithms efficiently.

Memory (RAM): Adequate RAM to support the game's runtime requirements, including storing game states and managing AI decision-making processes.

Display: A monitor or screen to display the game graphics and user interface.

Input Devices: Keyboard and mouse for player interaction with the game.

Software Requirements:

Operating System: The game should be compatible with popular operating systems such as **Windows, macOS, or Linux.**

Programming Language: The game can be developed using a programming language such as Python, which offers simplicity and flexibility for game development.

Development Environment: Software development environments such as PyCharm, Visual Studio Code, or IDLE can be used for coding and debugging the game..

Database System (Optional): If implementing features such as data persistence or player statistics, a database system like SQLite or MySQL may be required.

Version Control System: Using a version control system like Git can help manage code changes, collaborate with other developers, and track project history.

Text Editor: A text editor for writing and editing code files, such as Sublime Text, Atom, or Notepad++, can be used alongside the development environment.

3.2.1 Aim:

The "21, Bagram" game is designed with the primary objective of providing players with an entertaining and engaging experience. The game aims to captivate players by offering fun gameplay mechanics, appealing visuals, and interactive elements that keep them engaged for extended periods. By immersing players in a strategic environment, "21, Bagram" offers a unique and enjoyable gaming experience that challenges their minds and reflexes.

Another critical objective of the "21, Bagram" game is to provide strategic depth. The game offers meaningful decisions and tactics that players can employ during gameplay, requiring them to carefully plan their moves, anticipate their opponents' actions, and adapt their strategies to outsmart their rivals. By offering a range of possible actions and outcomes, "21, Bagram" encourages players to think critically and strategically, providing a rich and rewarding gaming experience.

In addition to entertainment and strategic depth, "21, Bagram" aims to help players develop and hone various skills. The game requires players to use their numerical reasoning, critical thinking, and decision-making abilities under pressure, providing a challenging yet rewarding environment for skill development. By requiring players to count tactically, bluff convincingly, and manage risk effectively, "21, Bagram" fosters the growth of these essential

skills in a fun and engaging way.

Finally, "21, Bagram" strives to be accessible to players of all ages and skill levels. The game features clear rules, intuitive controls, and adjustable difficulty settings, ensuring that both casual and experienced gamers can enjoy the game and find it approachable. By offering a range of difficulty levels, "21, Bagram" ensures that players can find a level of challenge that suits their abilities and preferences, making the game enjoyable for everyone.

3.2.2 Scope:

The "21, Bagram" game is a strategic counting game that challenges players to think ahead and make quick decisions. The game is played in turns, with each player allowed to say up to three consecutive numbers, starting from 1 and going up to 21. The player who calls out "21" is eliminated, and the last player remaining wins the round. The scope of the game encompasses various aspects that define its features, functionalities, and potential for expansion.

The core gameplay mechanics involve counting from 1 to 21 in turns, with each player allowed to say up to three consecutive numbers. The game can feature a single-player mode where players compete against computer-controlled opponents. The scope includes developing an AI system capable of providing challenging and varied gameplay experiences, with adjustable difficulty levels to cater to different skill levels.

Additional game modes and variations can expand the scope and increase replay value. This could include timed modes, multiplayer tournaments, special challenges, or themed variations of the core gameplay mechanics.

The game's visual design is an essential aspect of its scope. This includes creating appealing and intuitive user interfaces, visually engaging backgrounds and environments, and animations that enhance the overall gaming experience.

The game's sound and music design are also within the scope of the game. This includes creating sound effects that enhance the gameplay experience, as well as background music that sets the tone and atmosphere of the game.

The scope also includes implementing a progression system that rewards players for their achievements and encourages them to continue playing. This could include unlocking new game modes, customization options, or other rewards that provide a sense of accomplishment and motivation.

3.3 Architecture Diagram:

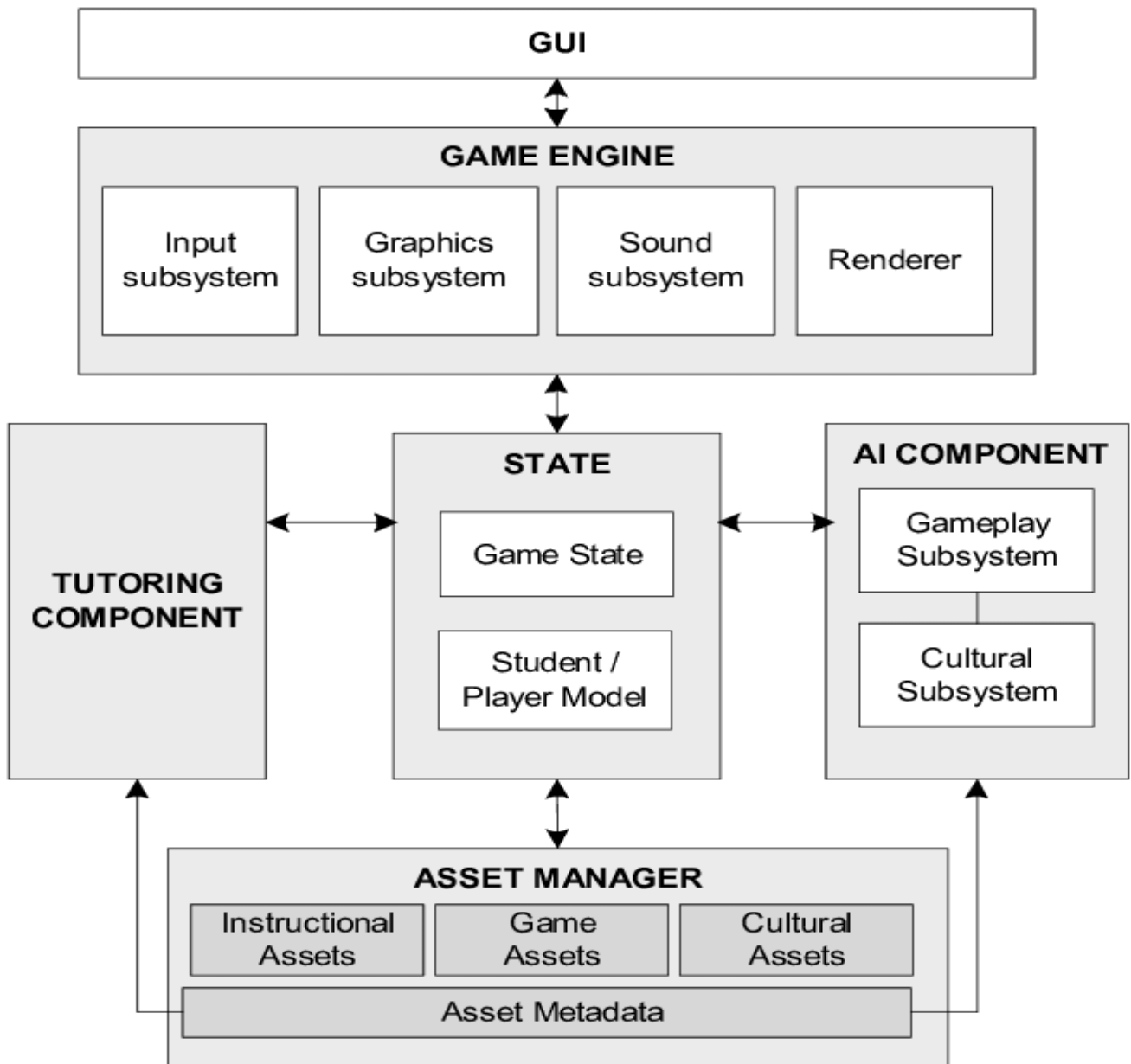


Fig.3.3: Architecture Diagram

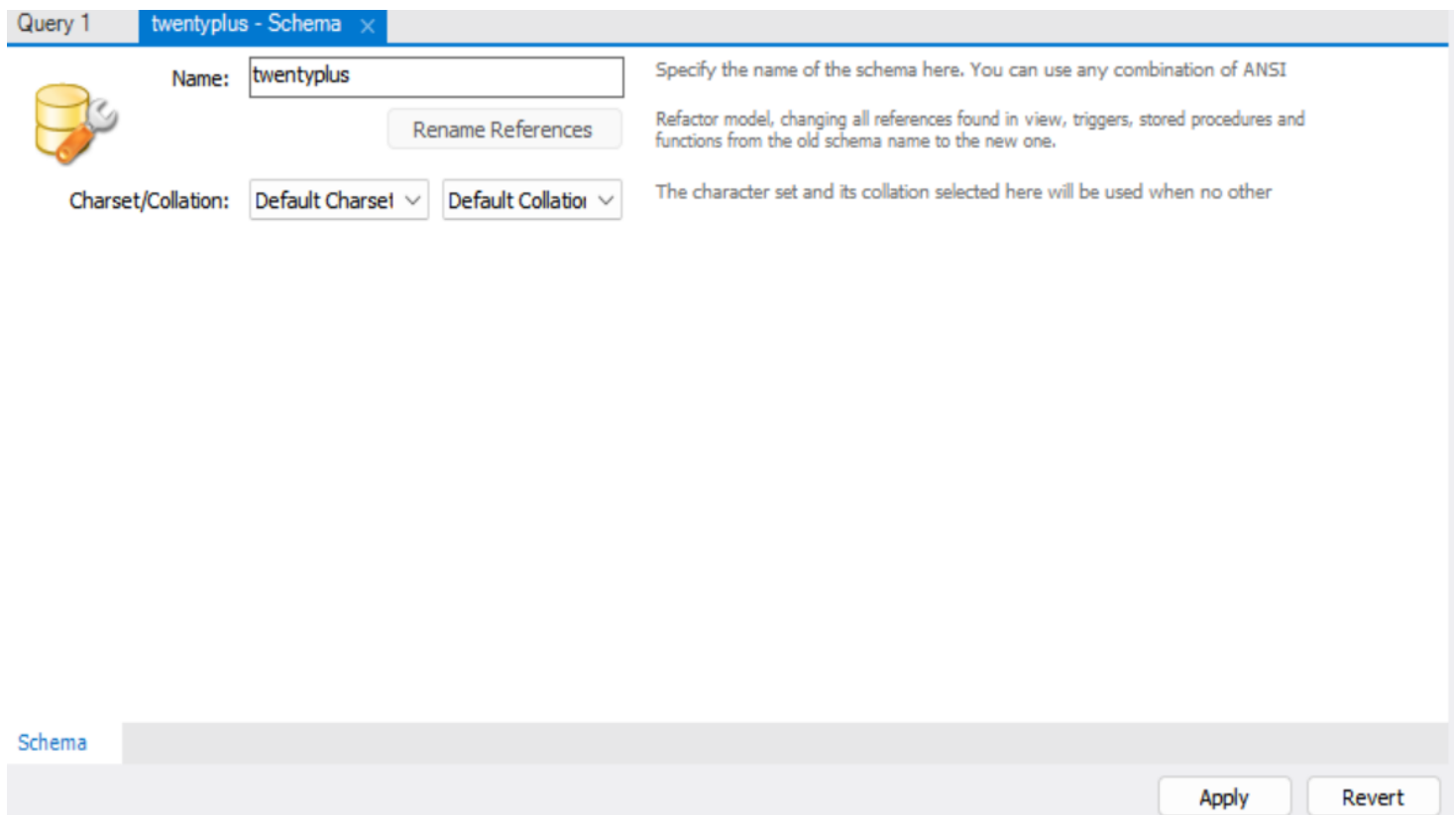
The architecture for "21 Bagram" game involves client-side interface for user interaction, server-side logic for game management, deck and card management for dealing and shuffling, and security measures along with thorough testing for reliability.

CHAPTER 4

IMPLEMENTATION OF PROPOSED SYSTEM


4.1 Setting up the database:

First, ensure you have MySQL installed and create a database named hangman.



The screenshot shows the 'Create Schema' dialog box in MySQL. The 'Name' field is set to 'twentyplus'. The 'Charset/Collation' section shows 'Default Charset' and 'Default Collation' dropdowns. The 'Apply' button is highlighted.

Query 1 | twentyplus - Schema ×

 Name: Specify the name of the schema here. You can use any combination of ANSI

Refactor model, changing all references found in view, triggers, stored procedures and functions from the old schema name to the new one.

Charset/Collation: The character set and its collation selected here will be used when no other

Schema

Fig.4.2.1: Creating Database

The above figure gives details about how to create a database in MySQL which will be used to store results of the game.

Then, create a table named login table to store player data:

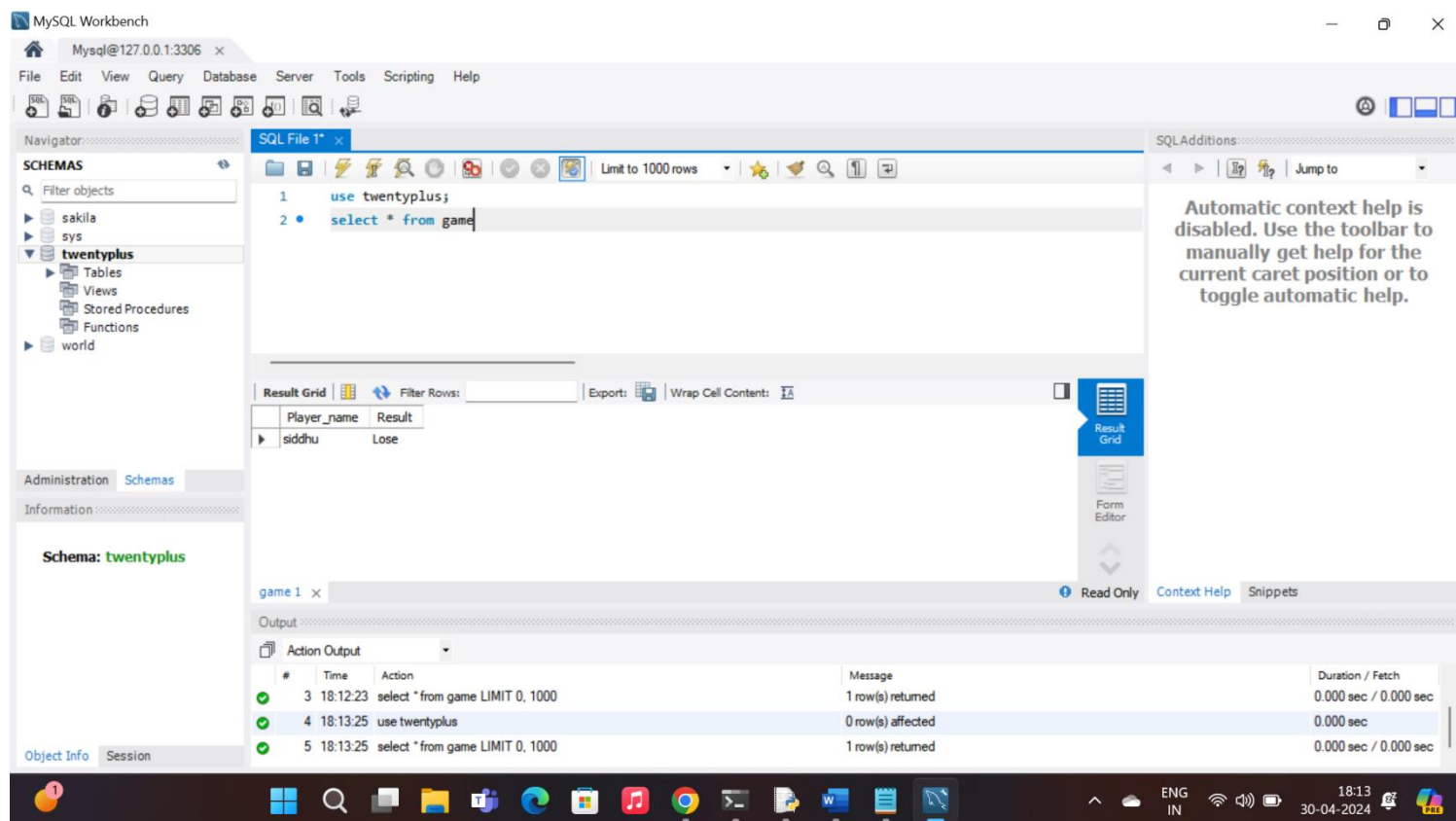


Fig. 4.2. 2: Creating game table.

The above figure shows the commands which uses twentyplus database to create a table called game and displays the table.

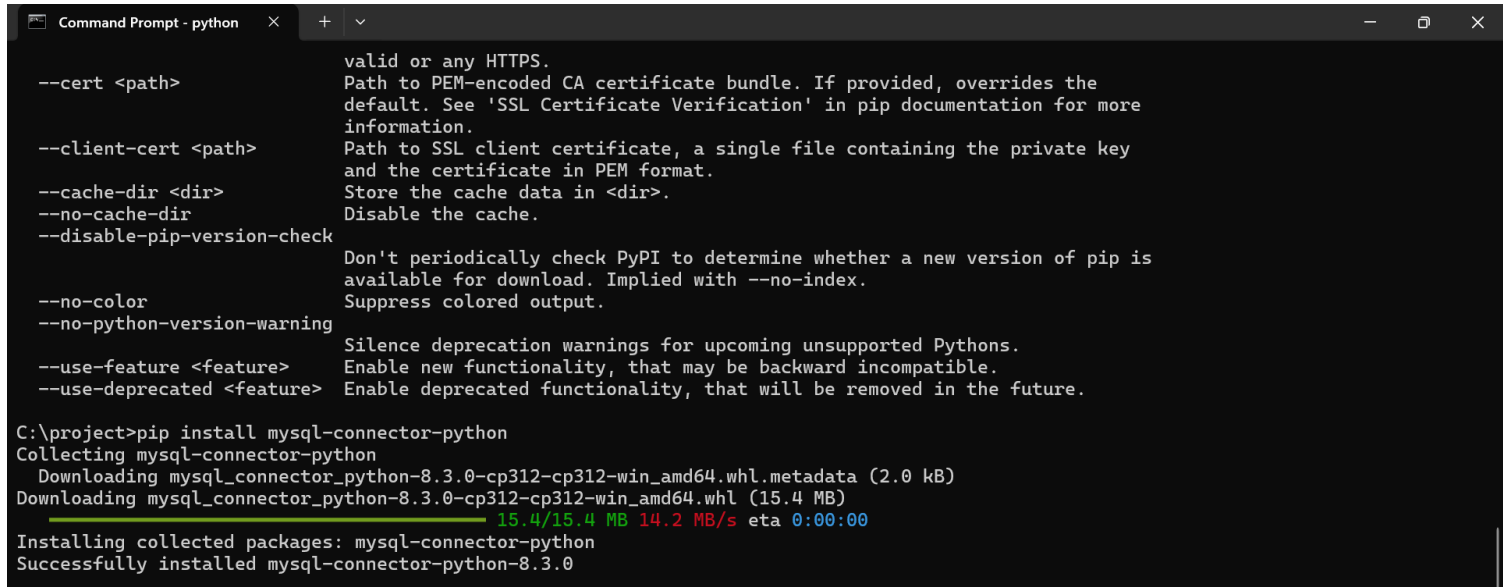
4.2. PYTHON IMPLEMENTATION:

Now, let's implement the 21, bagram game in Python, integrating with MySQL.



Fig. 4.3.1: Python package install

This command is executed in command prompt to install the package mysql.connector which is used to connect the python code with the MySQL database.



```
valid or any HTTPS.
Path to PEM-encoded CA certificate bundle. If provided, overrides the
default. See 'SSL Certificate Verification' in pip documentation for more
information.
--client-cert <path> Path to SSL client certificate, a single file containing the private key
and the certificate in PEM format.
--cache-dir <dir> Store the cache data in <dir>.
--no-cache-dir Disable the cache.
--disable-pip-version-check Don't periodically check PyPI to determine whether a new version of pip is
available for download. Implied with --no-index.
--no-color Suppress colored output.
--no-python-version-warning Silence deprecation warnings for upcoming unsupported Pythons.
--use-feature <feature> Enable new functionality, that may be backward incompatible.
--use-deprecated <feature> Enable deprecated functionality, that will be removed in the future.

C:\project>pip install mysql-connector-python
Collecting mysql-connector-python
  Downloading mysql_connector_python-8.3.0-cp312-cp312-win_amd64.whl.metadata (2.0 kB)
  Downloading mysql_connector_python-8.3.0-cp312-cp312-win_amd64.whl (15.4 MB)
    15.4/15.4 MB 14.2 MB/s eta 0:00:00
Installing collected packages: mysql-connector-python
Successfully installed mysql-connector-python-8.3.0
```

Fig. 4.3.2: Python-SQL Connection

This command prompt window shows the process of installing the package mysql.connector using pip and the prompt message of successful installation

4.3 Advantages and Disadvantages:

Advantages of the "21, Bagram" Game:

- **Easy to Learn:** The game has simple rules that are easy to understand, making it accessible to players of all ages and skill levels.
- **Quick Gameplay:** Rounds of the game are typically short, allowing for quick and casual gameplay sessions that fit well into busy schedules.
- **Improves Numerical Skills:** Playing the game involves counting and strategic thinking, which can help improve numerical skills and mental arithmetic abilities.
- **Social Interaction:** "21, Bagram" is often played in social settings, providing opportunities for social interaction, friendly competition, and bonding among players.
- **No Special Equipment Required:** The game can be played with just a group of people and doesn't require any special equipment or materials, making it convenient and accessible.
- **Adaptable Rules:** The game's rules can be easily modified or adapted to create variations or add complexity, allowing for customized gameplay experiences.

Disadvantages of the "21, Bagram" Game:

- **Limited Depth:** The game's simplicity may lead to limited depth or strategic complexity compared to more intricate games, potentially resulting in shorter engagement periods.
- **Repetitive Gameplay:** The core mechanics of counting and avoiding saying "21" may become repetitive over time, leading to decreased interest or boredom for some players.
- **Luck Factor:** While strategy plays a role in the game, luck also plays a significant part, as players' success can depend on the numbers they are dealt and the actions of other players.
- **Lack of Solo Play:** "21, Bagram" is primarily designed for multiplayer interaction, so solo players may not find it as engaging or enjoyable without other participants.
- **Limited Variation:** Without additional features or variations, the game may lack long-term replay value for players seeking diverse or evolving gameplay experiences.
- **Potential for Misinterpretation:** In some variations, the rules of the game may be subject to interpretation or misunderstanding, leading to confusion or disputes among players.

4.4 Coding:

Python code to play 21 Number game

```
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1",user="root",password='siddhu@3248',database='twentyplus')
if mydb.is_connected():
    print("Connected to database successfully")
mycursor=mydb.cursor()
# returns the nearest multiple to 4
def nearestMultiple(num):
    if num >= 4:
        near = num + (4 - (num % 4))
    else:
        near = 4
    return near

def lose1():
    print("\n\nYOU LOSE !")
    print("Better luck next time !")
    insert()
    exit(0)
```

```

# checks whether the numbers are consecutive
def check(xyz):
    i = 1
    while i < len(xyz):
        if (xyz[i] - xyz[i - 1]) != 1:
            return False
        i = i + 1
    return True

# starts the game
def start1():
    xyz = []
    last = 0
    while True:
        print("Enter 'F' to take the first chance.")
        print("Enter 'S' to take the second chance.")
        chance = input('> ')

        # player takes the first chance
        if chance == "F":
            while True:
                if last == 20:
                    lose1()
                else:
                    print("\nYour Turn.")
                    print("\nHow many numbers do you wish to enter?")
                    inp = int(input('> '))

                    if inp > 0 and inp <= 4:
                        comp = 4 - inp
                    else:
                        print("Wrong input. You are disqualified from the game.")
                        lose1()

                    i, j = 1, 1

                    print("Now enter the values")
                    while i <= inp:
                        a = input('> ')
                        a = int(a)
                        xyz.append(a)
                        i = i + 1

                    # store the last element of xyz.

```

```

last = xyz[-1]

# checks whether the input
# numbers are consecutive
if check(xyz):
    if last == 21:
        lose1()

    else:
        # "Computer's turn."
        while j <= comp:
            xyz.append(last + j)
            j = j + 1
        print("Order of inputs after computer's turn is: ")
        print(xyz)
        last = xyz[-1]
else:
    print("\nYou did not input consecutive integers.")
    lose1()

# player takes the second chance
elif chance == "S":
    comp = 1
    last = 0
    while last < 20:
        # "Computer's turn"
        j = 1
        while j <= comp:
            xyz.append(last + j)
            j = j + 1
        print("Order of inputs after computer's turn is:")
        print(xyz)
        if xyz[-1] == 20:
            lose1()
        else:
            print("\nYour turn.")
            print("\nHow many numbers do you wish to enter?")
            inp = int(input('> '))
            inp = int(inp)
            if inp > 0 and inp <= 4:
                i = 1
                print("Enter your values")
                while i <= inp:
                    xyz.append(int(input('> ')))
                    i = i + 1
                last = xyz[-1]

```

```

        if check(xyz):
            near = nearestMultiple(last)
            comp = near - last if near-last<4 else 3
        else:
            print("\nYou did not input consecutive integers.")
            lose1()
    else:
        print("Wrong input.You are disqualified from the game.")
        lose1()
    print("\n\nCONGRATULATIONS !!!")
    global result
    result="Win"
    insert()
    print("YOU WON !")
    exit(0)
else:
    print("wrong choice")
query="insert into game values(%s,%s)"
val=(name,result)
mycursor.execute(query,val)
mydb.commit()

```

```

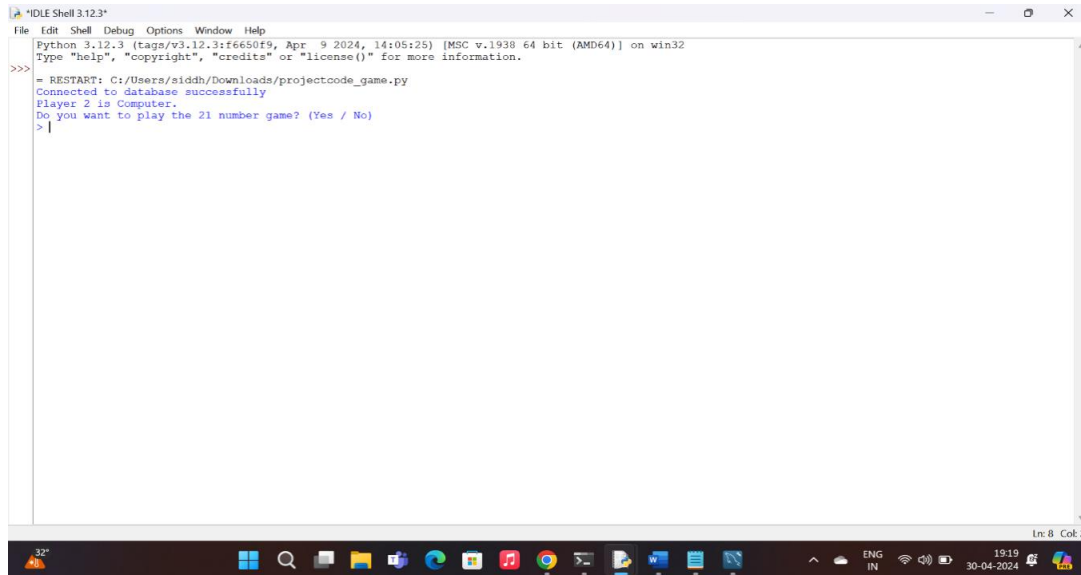
game = True
result="Lose"
while game:
    print("Player 2 is Computer.")
    print("Do you want to play the 21 number game? (Yes / No)")
    ans = input('> ')
    if ans == 'Yes' or ans=='yes':
        name=input("Enter your name:")
        start1()
    else:
        print("Do you want quit the game?(yes / no)")
        nex = input('> ')
        if nex == "yes" or 'YES':
            print("You are quitting the game...")
            exit(0)
        elif nex == "no" or 'NO':
            print("Continuing...")
        else:
            print("Wrong choice")

```

CHAPTER 5

RESULTS AND DISCUSSION

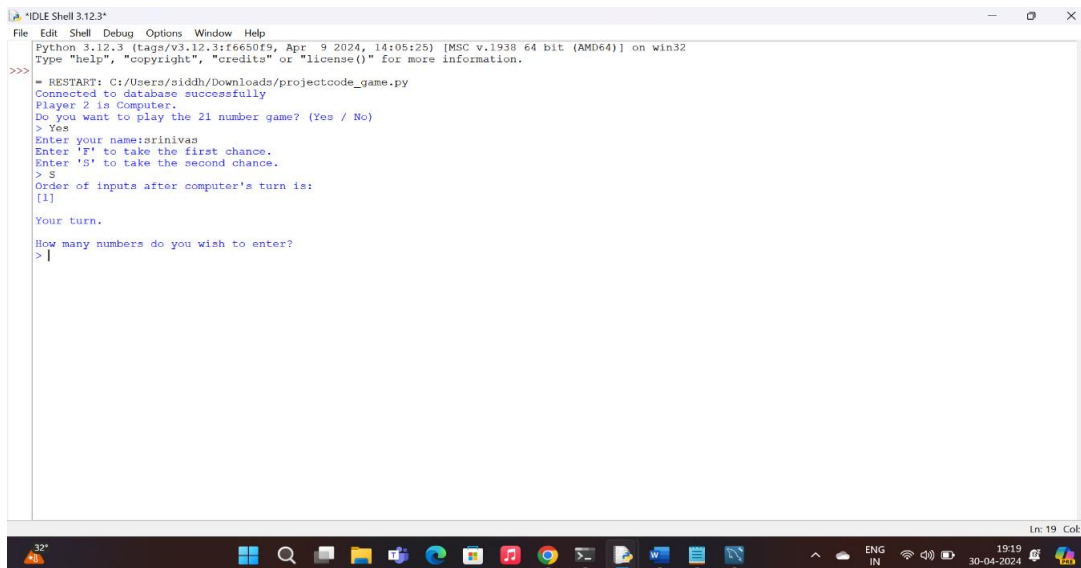
5.1 Screenshots:



```
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/siddh/Downloads/projectcode_game.py
Connected to database successfully
Player 2 is Computer.
Do you want to play the 21 number game? (Yes / No)
> |
```

Fig.5.1.1: Sample output 1

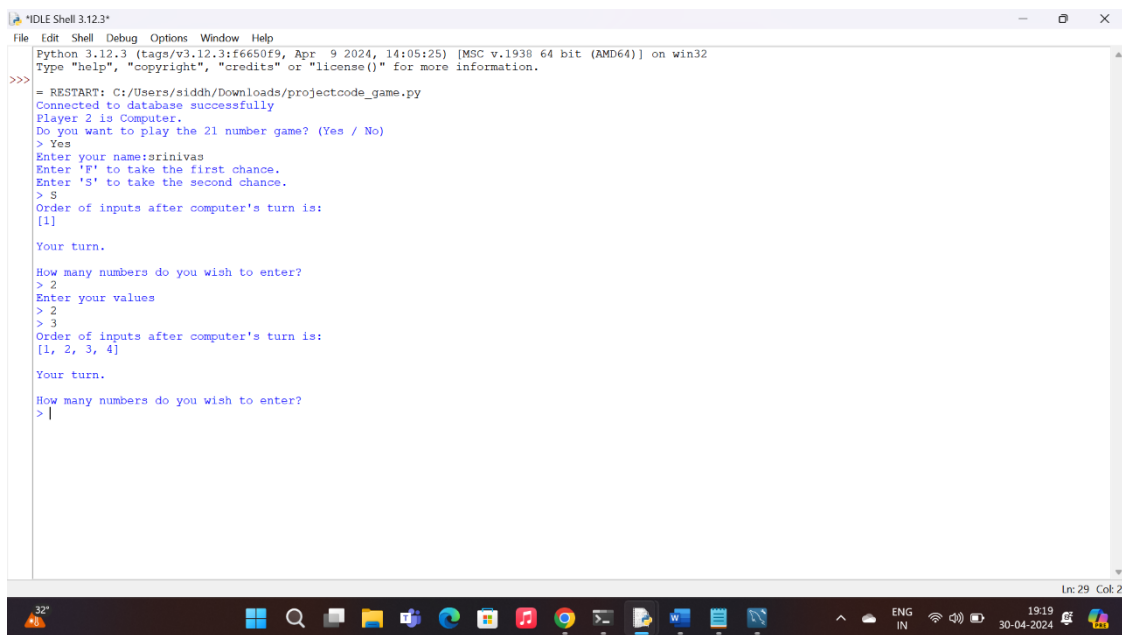
The above figure shows the prompt messages given to the user like about database connectivity, computer is player 2 and whether user wants to play the game or not



```
= RESTART: C:/Users/siddh/Downloads/projectcode_game.py
Connected to database successfully
Player 2 is Computer.
Do you want to play the 21 number game? (Yes / No)
> Yes
Enter your name: srinivas
Enter 'F' to take the first chance.
Enter 'S' to take the second chance.
> S
Order of inputs after computer's turn is:
[1]
Your turn.
How many numbers do you wish to enter?
> |
```

Fig.5.1.2: Sample output 2

The above figure shows about the user entering his detail like name and whether he wants to take first chance or second chance to enter the values.



```
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/siddh/Downloads/projectcode_game.py
Connected to database successfully
Player 2 is Computer.
Do you want to play the 21 number game? (Yes / No)
> Yes
Enter your name:srinivas
Enter 'F' to take the first chance.
Enter 'S' to take the second chance.
> S
Order of inputs after computer's turn is:
[1]

Your turn.

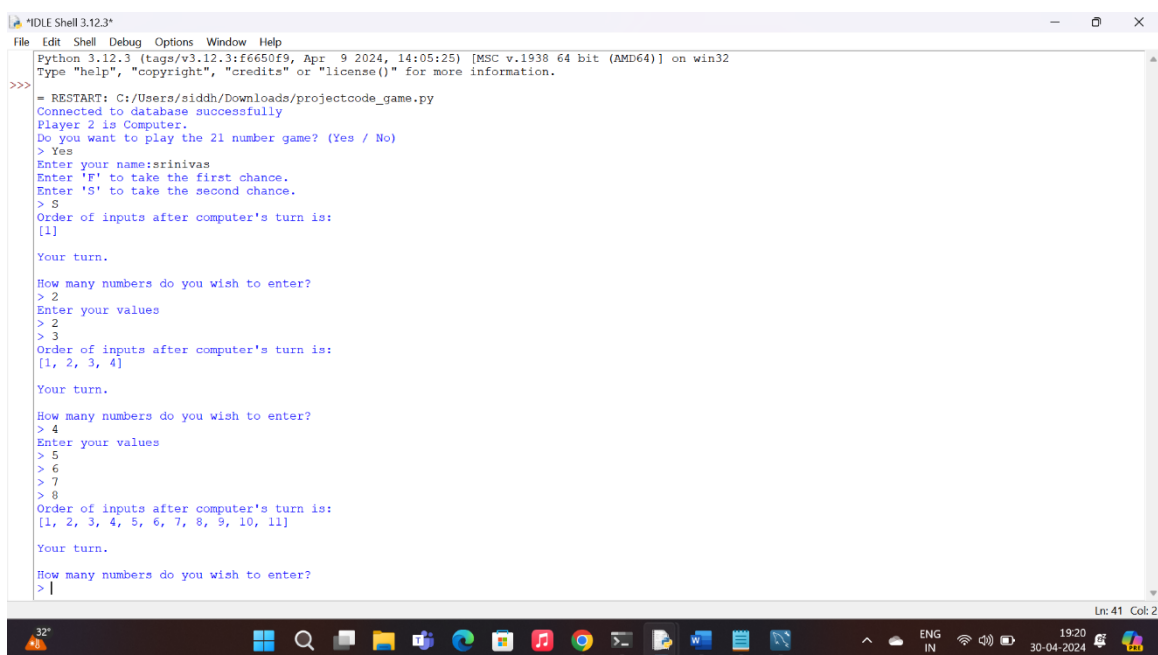
How many numbers do you wish to enter?
> 2
Enter your values
> 2
> 3
Order of inputs after computer's turn is:
[1, 2, 3, 4]

Your turn.

How many numbers do you wish to enter?
> 1
```

Fig.5.1.3: Sample output 3

The above figure shows that user takes the second chance to enter values and wished to enter 2 values after computer has entered 1 ,so the list becomes [1,2,3].



```
Python 3.12.3 (tags/v3.12.3:f6650f9, Apr 9 2024, 14:05:25) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/siddh/Downloads/projectcode_game.py
Connected to database successfully
Player 2 is Computer.
Do you want to play the 21 number game? (Yes / No)
> Yes
Enter your name:srinivas
Enter 'F' to take the first chance.
Enter 'S' to take the second chance.
> S
Order of inputs after computer's turn is:
[1]

Your turn.

How many numbers do you wish to enter?
> 2
Enter your values
> 2
> 3
Order of inputs after computer's turn is:
[1, 2, 3, 4]

Your turn.

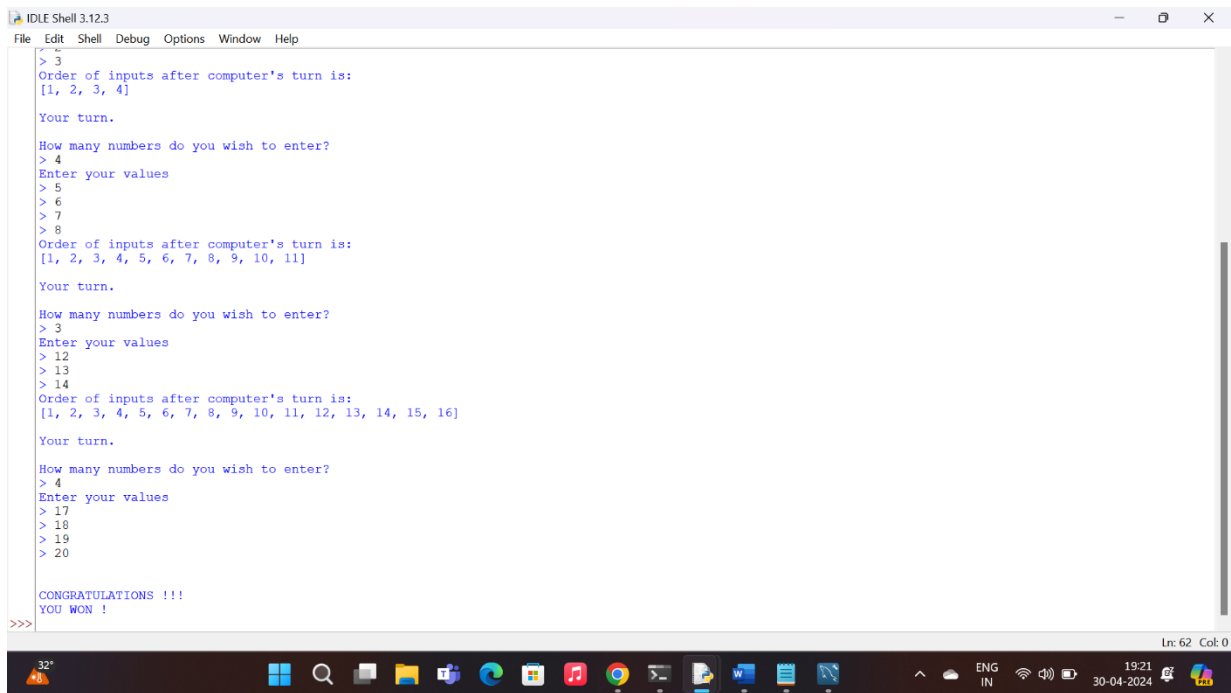
How many numbers do you wish to enter?
> 4
Enter your values
> 5
> 6
> 7
> 8
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]

Your turn.

How many numbers do you wish to enter?
> 1
```

Fig.5.1.4: Sample output 4

The above figure shows the user has wished to enter 4 values when the list was [1,2,3,4] so the new list will be [1,2,3,4,5,6,7,8,9,10,11] where the numbers from 9 to 11 are entered by computer.



```

IDLE Shell 3.12.3
File Edit Shell Debug Options Window Help
> 3
Order of inputs after computer's turn is:
[1, 2, 3, 4]
Your turn.
How many numbers do you wish to enter?
> 4
Enter your values
> 5
> 6
> 7
> 8
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]
Your turn.
How many numbers do you wish to enter?
> 3
Enter your values
> 12
> 13
> 14
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]
Your turn.
How many numbers do you wish to enter?
> 4
Enter your values
> 17
> 18
> 19
> 20
CONGRATULATIONS !!!
YOU WON !
>>>

```

Fig.5.1.5: Sample output 5

The above figure shows the final result of the game where the user wins the game by entering 4 values after 16 which means that 21 value should be entered by computer.

Architecture

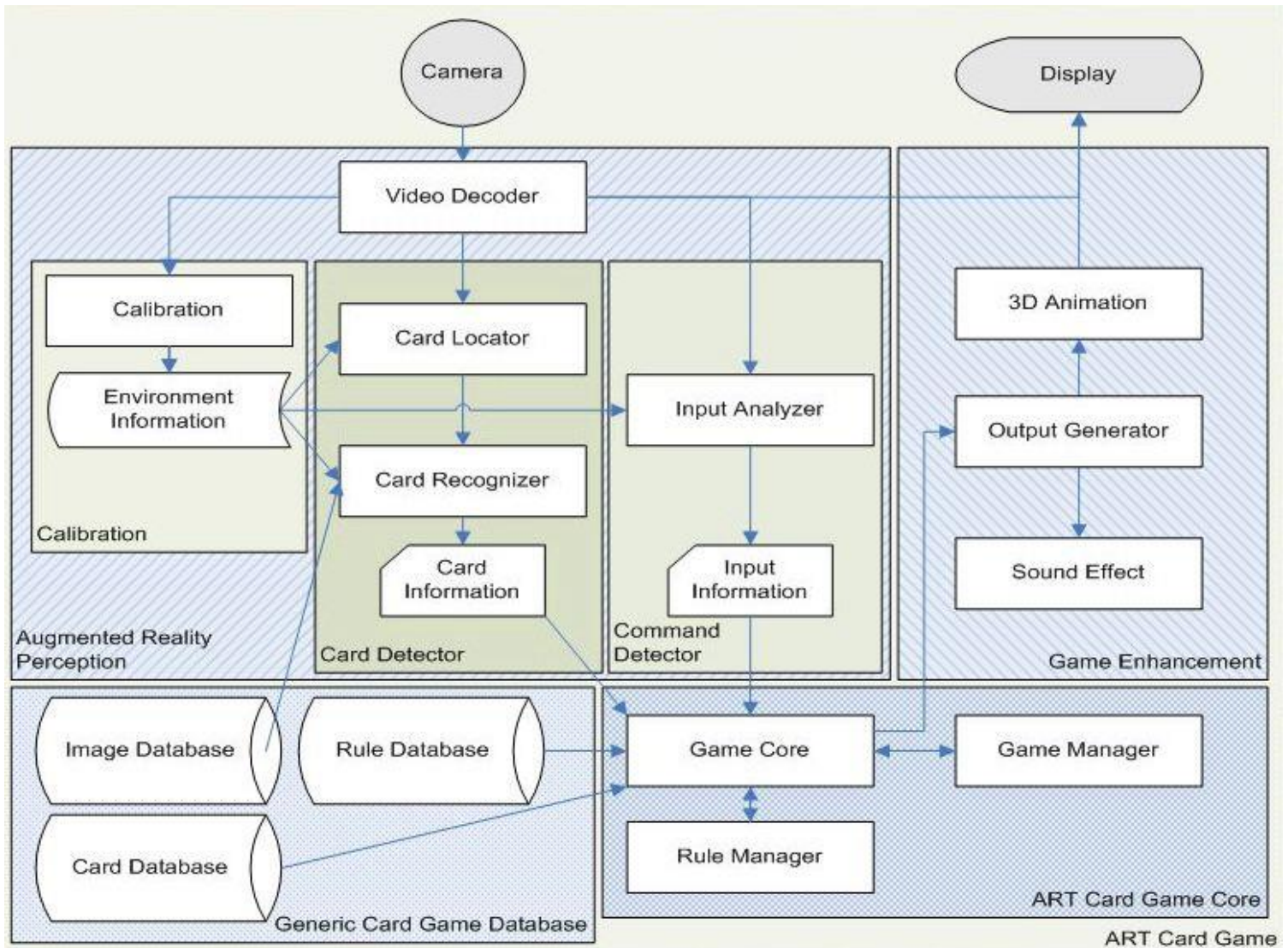


FIG: ARCHITECTURE DIAGRAM

The architecture for "21 Bagram" game involves client-side interface for user interaction, server-side logic for game management, deck and card management for dealing and shuffling, and security measures along with thorough testing for reliability.

CHAPTER 6

CONCLUSION

In conclusion, the implementation of the 21, Bagram game using Python programming language showcases a simple yet engaging game between a player and the computer. By leveraging SQL to connect to Python, the game progresses smoothly, allowing for seamless interaction between the player and the computer.

Key features of the game include the ability for the player to choose their starting position, ensuring flexibility and engagement. Additionally, the display of the list of numbers before the player's turn enhances convenience and clarity during gameplay.

The implementation enforces rules such that disqualification occurs if consecutive numbers are not provided as input, maintaining the integrity of the game. Ultimately, the player's objective is to strategically avoid calling "21," leading to victory, while the computer presents a challenging opponent.

Overall, the 21, Bagram game implementation provides an enjoyable and interactive experience, demonstrating the versatility and functionality of Python programming in gaming applications. With further refinements and variations, this game has the potential for widespread appeal and continued enjoyment among players of all skill levels.

REFERENCES

- [1] Flask Documentation: <https://flask.palletsprojects.com/en/2.0.x/> -Official documentation for Flask web framework (if applicable).
- [2] Game Development Books: "Invent Your Own Computer Games with Python" by Al Sweigart "Python Game Programming by Example" by Alejandro Rodas de Paz
- [3] Online Python Courses: Platforms like Coursera, Udemy, and Codecademy offer courses on Python programming and game development.
- [4] PyInstaller Documentation: <https://www.pyinstaller.org/documentation.html> - Documentation for PyInstaller tool for packaging Python applications into standalone executables.
- [5] Python Documentation: <https://docs.python.org/> - Official documentation for Python programming language.
- [6] Python Programming Tutorials on YouTube: Various YouTube channels offer tutorials on Python programming, game development, and database integration.
- [7] Stack Overflow: <https://stackoverflow.com/> - Community-driven question and answer website for programming-related queries.
- [8] SQL Books "Learning SQL" by Alan Beaulieu.
- [9] "SQL Cookbook" by Anthony Molinaro.