**“DOCUMENTATION OF DICE ROLLING SIMULATOR PROJECT”**

**Submitted To:** Mam Imrana Rafiq **Submitted By:**

Shees Ul Haq 21011598-191

Malik Shahnawaz 21011598-190

Hadabia Amjad 21011598-182

Noor Fatima 21011598-163

**Course Title:**

Software Construction and Development **Course Code:**

SE-327

**Semester :**

5th-C

**Department:**

Software Engineering



Table of content

[1. Dice Rolling Simulator: 3](#_Toc156565609)

[2. Abstract 3](#_Toc156565610)

[3. Target Audience: 4](#_Toc156565611)

[4. Project Goals and Objectives: 4](#_Toc156565612)

[i) Educational Exploration: 4](#_Toc156565613)

[ii) User Engagement and Interactivity: 4](#_Toc156565614)

[iii) Versatility and Customization: 4](#_Toc156565615)

[iv) Accessibility and Inclusivity: 4](#_Toc156565616)

[v) Integration of Technology: 5](#_Toc156565617)

[5. Tools and Technology: 5](#_Toc156565618)

[a. Visual Studio Code 5](#_Toc156565619)

[b. Python Language 5](#_Toc156565620)

[c. Tkinter Library 5](#_Toc156565621)

[d. Random Module 5](#_Toc156565622)

[6. Flow Chart: 6](#_Toc156565623)

[7. Model Architecture: 7](#_Toc156565624)

[7.1. Dice Roll Window 8](#_Toc156565625)

[Purpose: 8](#_Toc156565626)

[Components: 8](#_Toc156565627)

[Working: 8](#_Toc156565628)

[7.2 Dice Window 9](#_Toc156565629)

[Purpose: 9](#_Toc156565630)

[Components: 9](#_Toc156565631)

[Working: 9](#_Toc156565632)

[8. Source Code: 9](#_Toc156565633)

[9. Validation and Verification: 15](#_Toc156565634)

[10. Result 16](#_Toc156565635)

[11. Conclusion 16](#_Toc156565636)

[12. References: 17](#_Toc156565637)

# 1. Dice Rolling Simulator:

In a world increasingly dominated by technology, the allure of traditional games remains timeless. Among these, the simple act of rolling dice has been a source of entertainment for centuries. However, with the advent of digital tools, the tangible experience of rolling physical dice is being replaced by virtual simulations. The "Dice Rolling Simulator" aims to bridge the gap between the traditional and the modern, providing an engaging platform to explore the principles of randomness and probability.

At its core, the dice rolling simulator is more than just a game; it's an exploration of randomness. Randomness, a concept deeply embedded in the fabric of our universe, is often challenging to comprehend. Through this project, we delve into the fundamental nature of chance, discovering how a simple roll of dice can emulate the unpredictability that governs various aspects of life.

# 2. Abstract

The project “Dice Roll Simulator” is designed to provide an entertaining small software for the user. It is mainly built to let people spend their leisure time by rolling dice. In this project we have given up to 6 dice, which a user has the full authority to choose the number of dice he wants. He can select more than one dice at a time. In this way he can play with the system to entertain himself.

Beyond its entertainment value, the dice rolling simulator serves as an educational tool. By engaging with the simulator, users can develop a practical understanding of probability theory. The project encourages participants to analyze outcomes, understand distribution patterns, and make informed predictions – skills applicable across various disciplines, from mathematics and statistics to decision-making in everyday life.

The Dice Rolling Simulator is designed to be user-friendly yet comprehensive. Users can customize the number and types of dice, explore different probability distributions, and even simulate complex scenarios. The interactive nature of the simulator ensures an immersive experience, allowing participants to experiment with variables and observe the resulting outcomes in real-time.

# 3. Target Audience:

This project caters to a diverse audience. Whether you're a student seeking a hands-on approach to probability theory, an educator looking for a dynamic teaching tool, or a casual gamer interested in the art of chance, the Dice Rolling Simulator offers a unique and enjoyable experience for all.

# 4. Project Goals and Objectives:

## i) Educational Exploration:

Foster a deeper understanding of randomness and probability among users. Develop interactive features that allow users to experiment with different dice combinations. Provide educational content explaining the principles of probability theory and its real-world applications. Create scenarios that encourage users to apply probability concepts in a practical context.

## ii) User Engagement and Interactivity:

Create an engaging and user-friendly platform that encourages active participation. Design an intuitive and visually appealing interface for users of all ages. Implement customizable options for users to choose the number and types of dice, as well as other relevant parameters. Incorporate real-time feedback mechanisms to enhance user interaction and learning.

## iii) Versatility and Customization:

Provide a versatile platform that caters to a diverse audience with varying levels of interest and expertise. Allow users to simulate different dice combinations, from standard six-sided dice to More complex variations. Enable customization of scenarios to explore specific probability distributions or gaming scenarios. Accommodate both casual users and those seeking in-depth exploration of probability concepts.

## iv) Accessibility and Inclusivity:

Ensure that the simulator is accessible to a broad audience, including students, educators, and casual gamers. Optimize the platform for usability across different devices and screen sizes.

Provide educational resources in multiple formats, accommodating various learning styles. Consider inclusivity by designing features that cater to users with diverse abilities and preferences.

## v) Integration of Technology:

Leverage technology to enhance the user experience and expand the project's capabilities.

Explore possibilities for incorporating augmented reality (AR) or virtual reality (VR) elements in future iterations.

Consider machine learning algorithms to enhance the adaptability and intelligence of the simulator. Integrate collaborative features, allowing users to engage with the simulator together for shared learning experiences.

# 5. Tools and Technology:

The following tools and technology have been used in the project:

## a. Visual Studio Code

## b. Python Language

## c. Tkinter Library

## d. Random Module

1. **Visual Studio Code**

Visual Studio Code, also commonly referred to as **VS Code**, is a [source-code editor](https://en.wikipedia.org/wiki/Source-code_editor) developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) for [Windows,](https://en.wikipedia.org/wiki/Windows) [Linux](https://en.wikipedia.org/wiki/Linux) and [macOS](https://en.wikipedia.org/wiki/MacOS)[.[13]](https://en.wikipedia.org/wiki/Visual_Studio_Code#cite_note-TechCrunch-17) Features include support for [debugging,](https://en.wikipedia.org/wiki/Debugging) [syntax highlighting,](https://en.wikipedia.org/wiki/Syntax_highlighting) [intelligent code completion,](https://en.wikipedia.org/wiki/Intelligent_code_completion) [snippets,](https://en.wikipedia.org/wiki/Snippet_(programming)) [code refactoring,](https://en.wikipedia.org/wiki/Code_refactoring) and embedded [Git.](https://en.wikipedia.org/wiki/Git) Users can change the [theme,](https://en.wikipedia.org/wiki/Theme_(computing)) [keyboard shortcuts,](https://en.wikipedia.org/wiki/Keyboard_shortcut) preferences, and install [extensions](https://en.wikipedia.org/wiki/Plug-in_(computing)) that add functionality.

In the [Stack Overflow](https://en.wikipedia.org/wiki/Stack_Overflow) 2023 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool among 86,544 respondents, with 73.71% reporting that they use it. The survey also found Visual Studio Code to be used more by those learning to code than by professional developers (78% vs. 74%).

1. **Python Language**

Python is a [high-level,](https://en.wikipedia.org/wiki/High-level_programming_language) [general-purpose programming language.](https://en.wikipedia.org/wiki/General-purpose_programming_language) Its design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with the use of [significant indentation.](https://en.wikipedia.org/wiki/Off-side_rule)

Python is [dynamically typed](https://en.wikipedia.org/wiki/Type_system#DYNAMIC) and [garbage-collected.](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) It supports multiple [programming paradigms,](https://en.wikipedia.org/wiki/Programming_paradigm) including [structured](https://en.wikipedia.org/wiki/Structured_programming) (particularly [procedural)](https://en.wikipedia.org/wiki/Procedural_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) and [functional programming.](https://en.wikipedia.org/wiki/Functional_programming) It is often described as a "batteries included" language due to its comprehensive [standard library.](https://en.wikipedia.org/wiki/Standard_library)

[Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) began working on Python in the late 1980s as a successor to the [ABC programming language](https://en.wikipedia.org/wiki/ABC_(programming_language)) and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000. Python 3.0, released in 2008, was a major revision not completely [backwardcompatible](https://en.wikipedia.org/wiki/Backward_compatibility) with earlier versions. Python 2.7.18, released in 2020, was the last release of Python 2.

Python consistently ranks as one of the most popular programming languages, and has gained widespread use in the [machine learning](https://en.wikipedia.org/wiki/Machine_learning) community.

**iii) Tkinter Library**

**Tkinter** is a [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) [binding](https://en.wikipedia.org/wiki/Language_binding) to the [Tk](https://en.wikipedia.org/wiki/Tk_(software)) [GUI](https://en.wikipedia.org/wiki/Graphical_user_interface) toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's [*de facto* standard](https://en.wikipedia.org/wiki/De_facto_standard) GUI. Tkinter is included with standard [Linux,](https://en.wikipedia.org/wiki/Linux) [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) and [macOS](https://en.wikipedia.org/wiki/MacOS) installs of Python.

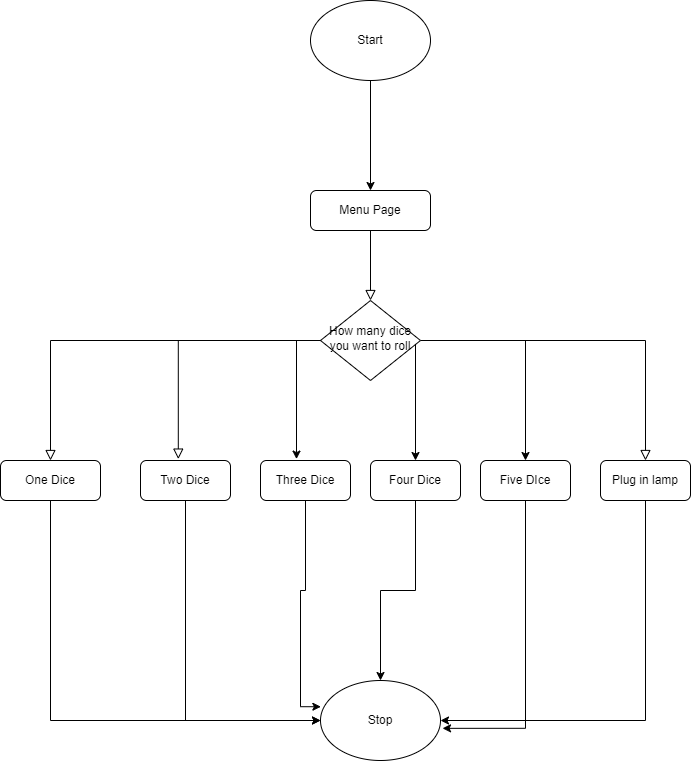
The name *Tkinter* comes from *Tk interface*. Tkinter was written by [Steen Lumholt](https://en.wikipedia.org/w/index.php?title=Steen_Lumholt&action=edit&redlink=1) and [Guido van Rossum,](https://en.wikipedia.org/wiki/Guido_van_Rossum) then later revised by Fredrik Lundh. Tkinter is [free software](https://en.wikipedia.org/wiki/Free_software) released under a [Python license.](https://en.wikipedia.org/wiki/Python_license)

**iv) Random Module:**

To achieve the randomization of numbers we have also used the random module in this project.

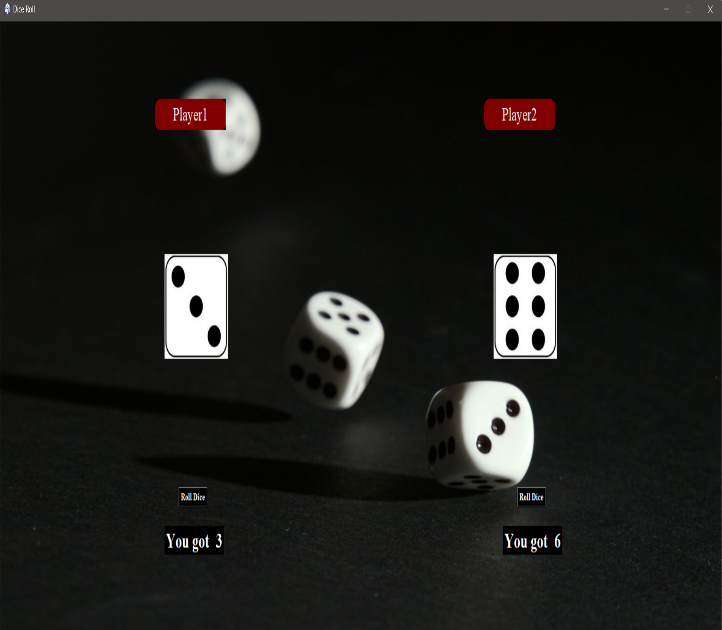
This module helps the programmers to generate random numbers both real and integer in python.

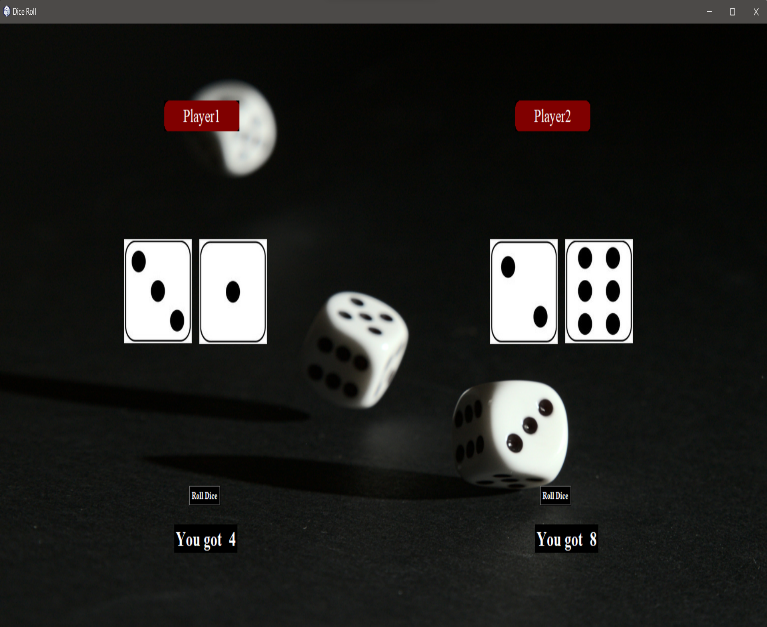
## 6. Flow Chart:

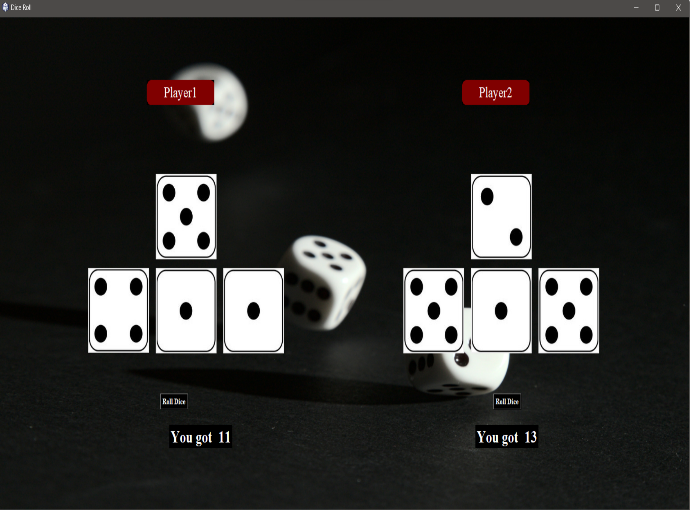
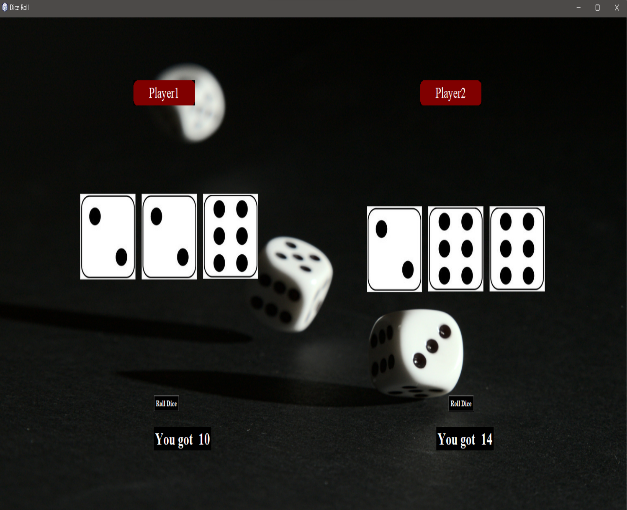


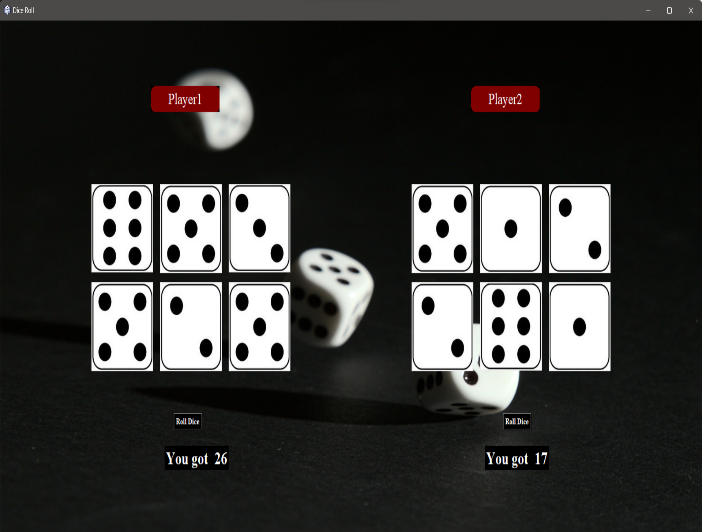
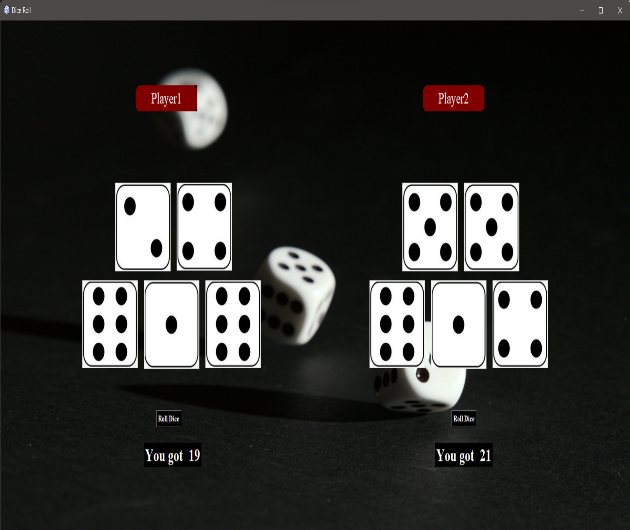
# 7. Model Architecture:









****

### 

### 7.1. Dice Roll Window

### Purpose:

This window is designed to provide the various buttons to the user. He can select one according to his taste. He has the option of selecting various dice.

### Components:

This window consists of six Buttons Widgets labelled with ‘One Dice’, ‘Two Dice’, ‘Three Dice’, ‘Four Dice’, ‘Five Dice’ and ‘Six Dice’. Apart from the buttons, there is a text Widget which has used to give a message to the user that how many dice he wants to roll?

### Working:

When a user selects a button, there opens another window, which consists of a Button Widget to let the user roll the dice.

### 7.2 Dice Window

### Purpose:

This Window has been designed to provide the user the experience of rolling dice. The User can roll the dice in this window.

### Components:

This window consists of one of more dice images and a Button Widget. The button widget allows the user to roll the dice.

### Working:

When a user clicks on the Roll Dice Button, it rolls the dice by generating random numbers on the dice. The user can roll the dice any number of times.

# 8. Source Code:

import tkinter as tkD  
from PIL import Image, ImageTk  
import random  
# from diceProject.dice import Dice  
#---------------------------------------------------------------------------------------------------------------------  
#that class is for one (1) Dice  
  
class one\_dice:  
 def one(self):  
 windows = tk.Toplevel()  
 windows.geometry("500x360")  
 windows.title("Dice Roll")  
  
   
 dice = ['dice1.jpg','dice2.jpg','dice3.jpg','dice4.jpg','dice5.jpg','dice6.jpg',]  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
  
 label1 = tk.Label(windows, image= image1)  
  
 label1.image = image1  
  
 label1.place(x=30, y=50)  
  
   
 def roll\_dice():  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label1.configure(image=image1)  
 label1.image = image1  
  
 button = tk.Button(windows, bg='black', fg= 'white', font='Times 10 bold', text='Roll Dice', command= roll\_dice)  
 button.place(x=250, y=10)  
 windows.mainloop()  
   
  
#----------------------------------------------------------------------------------------------------------------------  
#that class is for two dice  
  
class twoDice:  
 def two(self):  
 windows = tk.Toplevel()  
 windows.geometry("500x360")  
 windows.title("Dice Roll")  
  
 dice = ['dice1.jpg','dice2.jpg','dice3.jpg','dice4.jpg','dice5.jpg','dice6.jpg',]  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
  
 label1 = tk.Label(windows, image= image1)  
 label2 = tk.Label(windows, image=image2)  
  
 label1.image = image1  
 label2.image = image2  
  
 label1.place(x=60, y=100)  
 label2.place(x=300,y=100)  
  
 def roll\_dice():  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label1.configure(image=image1)  
 label1.image = image1  
   
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label2.configure(image=image2)  
 label2.image = image2  
   
 button = tk.Button(windows, bg='black', fg= 'white', font='Times 10 bold', text='Roll Dice', command= roll\_dice)  
 button.place(x=250, y=10)  
 windows.mainloop()  
  
#-----------------------------------------------------------------------------------------------------------------------  
#that class is for three dice  
  
class ThreeDice:  
 def three(self):  
 windows = tk.Toplevel()  
 windows.geometry("500x360")  
 windows.title("Dice Roll")  
  
 dice = ['dice1.jpg','dice2.jpg','dice3.jpg','dice4.jpg','dice5.jpg','dice6.jpg',]  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
  
 label1 = tk.Label(windows, image= image1)  
 label2 = tk.Label(windows, image=image2)  
 label3 = tk.Label(windows, image=image3)  
  
 label1.image = image1  
 label2.image = image2  
 label3.image = image3  
  
 label1.place(x=30, y=50)  
 label2.place(x=180,y=50)  
 label3.place(x = 330, y = 50)  
  
 def roll\_dice():  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label1.configure(image=image1)  
 label1.image = image1  
   
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label2.configure(image=image2)  
 label2.image = image2  
   
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label3.configure(image=image3)  
 label3.image = image3  
   
 button = tk.Button(windows, bg='black', fg= 'white', font='Times 10 bold', text='Roll Dice', command= roll\_dice)  
 button.place(x=250, y=10)  
 windows.mainloop()  
  
#-----------------------------------------------------------------------------------------------------------------------  
#This class is for four dice  
  
class fourDice:  
 def four(self):  
 windows = tk.Toplevel()  
 windows.geometry("500x360")  
 windows.title("Dice Roll")  
  
 dice = ['dice1.jpg','dice2.jpg','dice3.jpg','dice4.jpg','dice5.jpg','dice6.jpg',]  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image4 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
  
 label1 = tk.Label(windows, image= image1)  
 label2 = tk.Label(windows, image=image2)  
 label3 = tk.Label(windows, image=image3)  
 label4 = tk.Label(windows, image=image4)  
  
 label1.image = image1  
 label2.image = image2  
 label3.image = image3  
 label4.image = image4  
  
 label1.place(x=30, y=50)  
 label2.place(x=180,y=50)  
 label3.place(x = 330, y = 50)  
 label4.place(x =30, y = 200)  
  
 def roll\_dice():  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label1.configure(image=image1)  
 label1.image = image1  
   
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label2.configure(image=image2)  
 label2.image = image2  
   
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label3.configure(image=image3)  
 label3.image = image3  
   
 image4 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label4.configure(image=image4)  
 label4.image = image4  
   
 button = tk.Button(windows, bg='black', fg= 'white', font='Times 10 bold', text='Roll Dice', command= roll\_dice)  
 button.place(x=250, y=10)  
 windows.mainloop()  
  
#-----------------------------------------------------------------------------------------------------------------------  
#This class is for five dice  
  
class fiveDice:  
 def five(self):  
 windows = tk.Toplevel()  
 windows.geometry("500x360")  
 windows.title("Dice Roll")  
  
 dice = ['dice1.jpg','dice2.jpg','dice3.jpg','dice4.jpg','dice5.jpg','dice6.jpg',]  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image4 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image5 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
  
 label1 = tk.Label(windows, image= image1)  
 label2 = tk.Label(windows, image=image2)  
 label3 = tk.Label(windows, image=image3)  
 label4 = tk.Label(windows, image=image4)  
 label5 = tk.Label(windows, image=image5)  
  
 label1.image = image1  
 label2.image = image2  
 label3.image = image3  
 label4.image = image4  
 label5.image = image5  
  
 label1.place(x=30, y=50)  
 label2.place(x=180,y=50)  
 label3.place(x = 330, y = 50)  
 label4.place(x =30, y = 200)  
 label5.place(x = 180, y = 200)  
  
 def roll\_dice():  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label1.configure(image=image1)  
 label1.image = image1  
   
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label2.configure(image=image2)  
 label2.image = image2  
   
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label3.configure(image=image3)  
 label3.image = image3  
   
 image4 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label4.configure(image=image4)  
 label4.image = image4  
   
 image5 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label5.configure(image = image5)  
 label5.image = image5  
   
 button = tk.Button(windows, bg='black', fg= 'white', font='Times 10 bold', text='Roll Dice', command= roll\_dice)  
 button.place(x=250, y=10)  
 windows.mainloop()  
  
#-------------------------------------------------------------------------------------------------------------------------  
#This class is for six dice  
  
class sixDice:  
 def six(self):  
 windows = tk.Toplevel()  
 windows.geometry("500x360")  
 windows.title("Dice Roll")  
  
 dice = ['dice1.jpg','dice2.jpg','dice3.jpg','dice4.jpg','dice5.jpg','dice6.jpg']  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image4 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image5 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 image6 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
  
 label1 = tk.Label(windows, image= image1)  
 label2 = tk.Label(windows, image=image2)  
 label3 = tk.Label(windows, image=image3)  
 label4 = tk.Label(windows, image=image4)  
 label5 = tk.Label(windows, image=image5)  
 label6 = tk.Label(windows, image=image6)  
  
 label1.image = image1  
 label2.image = image2  
 label3.image = image3  
 label4.image = image4  
 label5.image = image5  
 label6.image = image6  
  
 label1.place(x=30, y=50)  
 label2.place(x=180,y=50)  
 label3.place(x = 330, y = 50)  
 label4.place(x =30, y = 200)  
 label5.place(x = 180, y = 200)  
 label6.place(x = 330, y=200)  
  
 def roll\_dice():  
 image1 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label1.configure(image=image1)  
 label1.image = image1  
   
 image2 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label2.configure(image=image2)  
 label2.image = image2  
   
 image3 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label3.configure(image=image3)  
 label3.image = image3  
   
 image4 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label4.configure(image=image4)  
 label4.image = image4  
   
 image5 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label5.configure(image = image5)  
 label5.image = image5  
   
 image6 = ImageTk.PhotoImage(Image.open(random.choice(dice)))  
 label6.configure(image=image6)  
 label6.image = image6  
   
 button = tk.Button(windows, bg='black', fg= 'white', font='Times 10 bold', text='Roll Dice', command= roll\_dice)  
 button.place(x=250, y=10)  
 windows.mainloop()  
  
#\_------------------------------------------------------------------------------------------------------------------------  
  
  
one = one\_dice()  
two = twoDice()  
three = ThreeDice()  
four = fourDice()  
five = fiveDice()  
six = sixDice()  
  
  
windows = tk.Tk()  
windows.geometry("500x360")  
windows.title("Dice Roll")  
  
label1 = tk.Label(windows, text="How many dice you want to Roll")  
label1.place(x = 180, y = 10)  
  
# button = tk.Button(windows, text="One Dice", command=)  
button1 = tk.Button(windows, text='One Dice',command=one.one)  
button2 = tk.Button(windows, text='Two Dice', command= two.two )  
button3 = tk.Button(windows, text='Three Dice', command=three.three)  
button4 = tk.Button(windows, text='Four Dice', command=four.four)  
button5 = tk.Button(windows, text='Five Dice', command=five.five)  
button6 = tk.Button(windows, text='Six Dice', command=six.six)  
  
button1.place(x = 230, y = 50)  
button2.place(x = 230, y = 100)  
button3.place(x = 230, y = 150)  
button4.place(x = 230, y = 200)  
button5.place(x = 230, y = 250)  
button6.place(x = 230, y = 300)  
windows.mainloop()

# 9. Validation and Verification:

The verification and validation of any system is as necessary as food for human living things. Validation process includes that it meets as the

Validation and verification are two critical processes in ensuring the reliability, functionality, and effectiveness of a system. These processes are often used interchangeably, but they refer to different aspects of system evaluation.

Verification is the process of evaluating a system or component during or after the development phase to determine whether it meets the specified requirements. We have verified the software by testing its different aspects. Each time it is rolled it generates new numbers within the specified range.

Whereas, validation is the process of evaluating a system or component during or after the development phase to determine whether it satisfies the intended use and meets the user requirements. This system does as we intended. It provides an entertaining game with error free.

# 10. Result

The Dice Rolling game makes a good impression on our test users. Where they loved to play with that game. Its user friendly interface makes them to attract on the software. It’s different choice of dice makes them to use the game more efficiently and it also makes a good impact on the user to make them stress free.

# 11. Conclusion

In conclusion, we have designed and implemented Dice Rolling Simulator with state to provide a verifiable, decentralized, game plateform. Now anyone can simply click on a button and get his next number on dice. It is our hope that this game will help people to get entertained. The culmination of this project brings forth a platform that not only entertains but also educates, providing users with a unique and immersive experience in the realm of probability and randomness.

## 12. References:

* <https://docs.python.org/3/library/tkinter.html>
* <https://www.youtube.com/watch?v=CIm5vfsfgO0&ab_channel=WsCubeTech>• https://www.wikipedia.org

Page