**A Mini Project Report on**

**BLACKJACK WITH PYTHON**

**Submitted to partial fulfillment of the requirements for the award of the degree**

**of**

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE & ENGINEERING**

**By**

**A. ADITYA---------------------- (17911A05J4)**

**Under the Esteemed Guidance of**

**M. Venkateswarlu**

**Asst. Professor**



Department of Computer Science & Engineering

**VIDYA JYOTHI INSTITUTE OF TECHNOLOGY**

**(An Autonomous Institution)**

**(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)**

**Aziz Nagar Gate, C.B. Post, Hyderabad-500075**

**2020-2021**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**CERTIFICATE**

This is to certify that the project report titled “**Blackjack with Python**” is being submitted by **A. ADITYA (17911A05J4)** in partial fulfilment for the award of the Degree of Bachelor of Technology in **Computer Science & Engineering,** is a record of bonafide work carried out by him under my guidance and supervision. These results embodied in this project report have not been submitted to any other University or Institute for the award of any degree of diploma.

**Internal Guide Head of Department**

M. Venkateswarlu Dr. B. Vijaya Kumar

Asst. Professor Professor

**External Examiner**

**DECLARATION**

I, **A. ADITYA** hereby declare that the project entitled, **“BLACKJACK WITH PYTHON”** submitted for the degree of Bachelor of Technology in Computer Science and Engineering is original and has been done by me and this work is not copied and submitted anywhere for the award of any degree.

**Date: 17 February 2021 A. ADITYA (17911A05J4)**

**Place: Hyderabad**

**ACKNOWLEDGEMENT**

I wish to express sincere gratitude to the project guide, **M. Venkateswarlu**, Asst. Professor, Vidya Jyothi Institute of Technology, Hyderabad for his timely cooperation and valuable suggestions while carrying out this work. It is his kindness that made me learn more.

I am grateful to **Dr. B. Vijayakumar**, Professor and HOD, department of CSE, for his help and support during our academic year.

I whole-heartedly convey my gratitude to Principal **Dr. A. Padmaja** for her constructive encouragement.

I would like to take this opportunity to express my gratitude to our Director **Dr. E. SAI BABA REDDY** for providing necessary infrastructure to complete this project.

I would thank my parents and all the faculty members who have contributed to our progress through the course to come to this stage.

**A. ADITYA (17911A05J4)**

**ABSTRACT**

**BLACKJACK WITH PYTHON**

In this project, a card game known as Blackjack is implemented by using the Python programming language. This game is played in such a way that both the client and user are given two cards each and out of all these cards only a single card of the dealer is visible to everyone. Based on this, the goal of the player is to determine whether to take another card so that he could he could get a total score that would be greater than that of the dealer but less than 21 and ultimately win the game.

Blackjack is a very famous casino game worldwide. Here, it will be played between the user (client) and the dealer (computer). The deck is unlimited in size and therefore the game cards remain perpetually. The goal of Blackjack is to get 21 (the best score) or less by adding up the value of the cards. Numbered cards have the value of the number on the card. Face cards (Jack, Queen, and King) have a value of 10. The Ace card can have a value of one or 11.

The player with the highest score but less than 21 compared to the dealer wins the game. Anyone with a score greater than 21 loses the game. This is called a "bust" also known as a "boot". Every card has an equal probability of being drawn and all the face cards count as 10 except Ace card which can either count as 11 or 1. The person (dealer / player) with a score as closest to 21 but not greater than that is proclaimed the winner.

**INDEX**

**S.NO TITLE PAGE NO**

**Abstract**

1. **Introduction 1**
2. **Literature Survey 3**
3. **Feasibility Study 4**
   1. Economic Feasibility 4
   2. Technical Feasibility 4
   3. Social Feasibility 5
   4. Schedule Feasibility 5
4. **System Requirement Specifications 6**
   1. Existing System 6
   2. Proposed System 6
   3. System Requirements 7

4.3.1 Software Requirements 7

4.3.2 Hardware Requirements 7

* 1. Requirements Definition 7

4.4.1 Functional Requirements 7

4.4.2 Non-Functional Requirements 8

1. **System Design 9**
   1. Data Flow Diagram 9
   2. UML diagrams 11
      1. Use Case Diagram 11
      2. Collaboration Diagram 12
      3. Activity Diagram 12
      4. Class Diagram 14
      5. Sequence Diagram 15
      6. State Chart Diagram 16
2. **Implementation 17**

6.1 Proposed methodology 17

6.2 Module Description 18

6.2.1 Card Dealing Module 18

6.2.2 Player Module 18

6.2.3 Dealer Module 18

6.2.4 Game Module 18

6.2.5 User Interaction Module 19

6.2.6 Main Display Module 19

**7. Software Environment 20**

7.1 Python 20

7.2 PyCharm 21

7.3 Repl.it 22

1. **Sample code 23**
2. **System Testing 26**
   1. Types of Tests 26

9.1.1 Unit testing 26

* + 1. Integration testing 27
    2. Performance testing 28
    3. Verification and validation 28

**10. Result and Output Screens 29**

**11. Conclusion 35**

**References 36**

**LIST OF FIGURES**

**S.NO TITLE PAGE NO**

* 1. Data Flow Diagram 9
  2. Use Case Diagram 11
  3. Collaboration Diagram 12
  4. Activity Diagram 12
  5. Class Diagram 14
  6. Sequence Diagram 15
  7. State Chart Diagram 16

**CHAPTER-1**

**INTRODUCTION**

Blackjack is a member of the global family of banking games known as Twenty-One, whose relatives include the British game of Pontoon and the European game, Vingt-et-Un. It is a comparing card game between one player and a dealer, where the player in turn competes against the dealer. It is played with one or more decks of 52 cards, and is the most widely played casino banking game in the world. This will be one of interesting application that one can work on and implement in real time world. The user interface must be simple and easy to understand. This will be one of the applications that will maintain the details of the user and the cards that are currently being utilized in the game.

The basic aim of the project is to provide a less addictive and text-based version of the famous casino game called Blackjack. Blackjack with Python is a Web-based application that is designed to store, process, retrieve and analyse information concerned with the administrative, sequential working and management within the blackjack system. This project aims at maintaining all the information related to clients or players, their choices and ultimately the outcome of the game at the end.

Through this application any person who is interested in playing the less addictive but more constructive and better version of the game can directly and immediately play it. The project is to provide transparency in this field, make the process of playing these kinds of games more intuitive rather than making compulsive towards gambling and betting in general. Blackjack with Python can be used by the users to play the blackjack game where count of the cards drawn is stored, manipulated and finally used to determine the winner.

The main objective of this application is to automate the complete operations of the blackjack game. The system needs to maintain hundreds of records. To develop a web-based portal to facilitate the co-ordination between user and computer’s cards. This system makes playing the game in a convenient text-based format, which can be provided in a sound, ethical and acceptable manner, consistent with the long-term well-being of the community. It is noted by many well-established research institutions that these kind of watered-down casino games which do not include any betting or gambling features are somewhat better at curbing chronic addiction to these games.

The system will provide the user the option to look at the details of the cards he is holding and the face up card of the computer. It allows the user to choose between the hit and stand options to play the game. The system provides a simple and quick interaction between the user and the system. A sub objective of this project is also for me learn the workings of the Python programming language, it’s packages, collections, functions and syntax.

This system is user friendly even for first timers. Anyone who is capable of reading and understanding can play the game. As soon as any update pertaining to the cards with which the game is being played occurs, it is immediately and swiftly reflected on the system and the user can see the updates in real time.

**CHAPTER-2**

**LITERATURE SURVEY**

Blackjack with Python is the text-based simulation of the famous casino game known as Blackjack. This section discusses findings and observations done by some research works on web-based and text-based games available online. The gathered information on these related papers strengthens and supports the research study. Gambling and gaming activities have become increasingly recognised as sharing many common features at a structural and aesthetic level. Both have also been implicated as contributing to harm through excessive involvement. Despite this, relatively little attention has been given to the fundamental characteristics that differentiate these two classes of activity, especially in situations where the boundaries between them may be particularly hard to distinguish. As the Internet offers a new venue for gambling, the risks for engaging in pathological behaviours are potentially increased. In light of this, a systematic literature review was conducted in order to shed further light on the relationship between gambling on the Internet and possible addiction by assessing Internet gambling in general and addictive gambling on the Internet specifically. It was clearly noted that these kinds of activities available to everyone on their fingertips results in the detriment of people as a whole. Also, it observed that the existing system has disadvantages of utilization of eye-catching graphics and pre-fixed sum of money to gamble with. Thus, this project developed has text-based system to play the game in a more constructive way.

The purpose of these literature reviews was to collect information on how simple text-based versions of these games are a bit better at curbing and rebounding from those activities. Based on the reviews, it was found out that text-based provide convenience and efficiency to the system users. These users can either be people with chronic addiction who want to get better or generally people and first timers to play the game.

**CHAPTER-3**

**FEASIBILITY STUDY**

The feasibility of the project is studied during this phase and some cost estimates are analysed with a normal plan for the project is taken forward. During the system analysis the feasibility study is carried to make sure that the project is done smoothly and not going to be a load on any system.

Following are the four considerations involved in the feasibility study:

1. ECONOMICAL FEASIBILITY

2. TECHNICAL FEASIBILITY

3. SOCIAL FEASIBILITY

4. SCHEDULE FEASIBILITY

**3.1 ECONOMICAL FEASIBILITY:**

This study is mainly stressed to check the monetary impact of the system on the organization. It deals with the sum of money that the company can put forth on the research and also for the enhancement is checked. Here we should make sure that the ideal framework should be developed within the spending limit, most of this can be obtained by using many of the advancements which are freely accessible and only purchasing the products that are unreservedly accessible.

**3.2 TECHNICAL FEASIBILITY:**

This study is mainly related to find the technical feasibility, which means it entirely focus on the technical requirements of the framework. There should not be extreme interest on the accessible technical resources for any of the system.

If the interest is high on the accessible technical assets it will prompt to high demands of the client. The system which was ought to should have modest necessity, with only few or no changes are required to actualize the proposed system.

**3.3 SOCIAL FEASIBILITY:**

This study handles with the level of acknowledgement of the system by the user. This study also includes the way of training the user to utilize the system efficiently. The user ought to use the system productively and should not compromise with the system. The measure the user accepts the system relies on the way we educate the user by legitimate procedures and should make him acquainted with the system. So that the user feels that he knows the system and his certainty levels will raise to work with the system which is an invited accomplishment for the project.

## 3.4 SCHEDULE FEASIBILITY:

In this study our main focus is centred on the time to complete the project. A project is said to be fizzled if it takes more computational time to complete before it is being useful. In accordingly, this means it is an estimate to find out how much time the system will take to fully develop, and to find out whether it can be finished in the specified time by using few techniques like payback period. Schedule feasibility is a measure to find how meaningful the project schedule is designed. Whether project is started in time? Is deadline reasonable? Will it be finished in time? And whether deadline is necessary or not. A small deviation can be implied in the original schedule which was opted at the beginning of the project. The project development is doable as far as schedule is considered.

**CHAPTER-4**

**SYSTEM REQUIREMENTS**

**4.1 EXISTING SYSTEM:**

There are many Blackjack simulation games online, of which, the most popular is the one by Washington Post. **Disadvantages:** But all these simulations are animation based and do not help to curb addiction but instead increase it towards these kind of activities by promoting things like betting by giving fake money from the start and using psychological addiction triggers like extremely attractive graphics. These systems are mainly used for entertainment but may also result in chronic addiction to casino games.

**4.2 PROPOSED SYSTEM:**

The proposed system is very much attainable and usable. This online system is completely text-based and as a result, it is not too appealing but is also interesting enough to start playing the game. This system does not promote addictive actions but instead encourages logical thinking by prompting the users to make precise calculations and make their own decisions to win the game ultimately. **Advantages:** It is proven by many research establishments that these kinds of watered-down casino game experiences are effective in treating addiction towards casino games like Blackjack to some extent. The users can run the program and start playing the game immediately where the information is updated and uploaded automatically and does not need anyone to do so. In this system, the communication is solely established between the user or client and the system or computer. The whole process is friendly, efficient and handy.

**4.3 SYSTEM REQUIREMENTS:**

**4.3.1 SOFTWARE REQUIREMENTS:**

* OS : Windows 7 and above (with any web browser)
* Front end : Python
* Back end : PyCharm or Repl.it

**4.3.2 HARDWARE REQUIREMENTS:**

* RAM : 1GB and Higher
* Processor : Intel i3 or above
* Hard Disk : 2GB or above

## 4.4 REQUIREMENTS DEFINITION:

After the severe continuous analysis of the problems that raised in the existing system, we are now familiar with the requirement that are required by the current system. The requirements that the system needs are categorized into the functional and non-functional requirements. These requirements are listed below:

### FUNCTIONAL REQUIRMENTS:

Functional requirement defines which functions or features that are to be incorporated in any system to fulfil the business requirements and to be acknowledged by the clients. On the premise, the functional requirements specify relationship between the inputs and outputs. All the operations to be performed on the input data to obtain output are to be specified. This includes specifying the validity checks on the input and output data, parameters affected by the operations and the other operations, which must be used to transform the inputs into outputs. Functional requirements specify the behaviour of the system for valid input and outputs.

### 

### 4.4.2 NON-FUNCTIONAL REQUIREMENTS:

Non-functional requirements provide a description of features, characteristics and capacity of the system and furthermore it may constraints the boundaries of the proposed system.

The following are the non-functional requirements that are essential depending on the performance, cost, control and gives efficiency and ease of use.

Based on the above explained non-functional pre-requisites are as follows:

* User friendly
* System should provide better accuracy
* To perform efficiently with better throughput and response time

**CHAPTER-5**

**SYSTEM DESIGN**

**5.1 DATA FLOW DIAGRAM:**

After starting the game, two cards for the player and a card for the computer are generated. The values for all these cards are interpreted and stored appropriately and game continues in the same way as long as the user rejects another card or gets cards whose value is more than 21. After this, the counts are compared and the winner is declared.

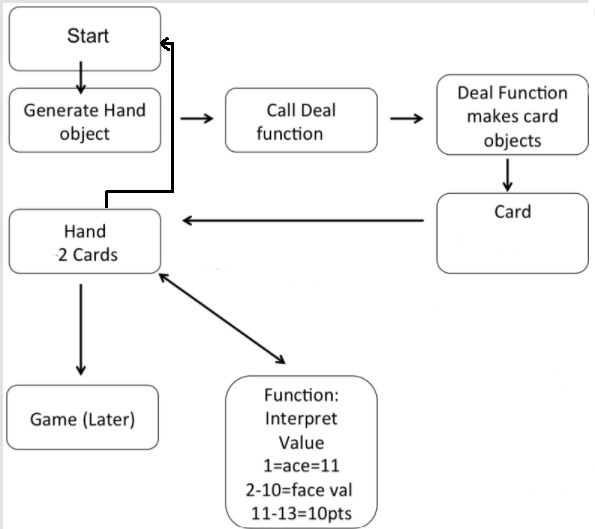
****

Fig 5.1.1.1: Data Flow Diagram

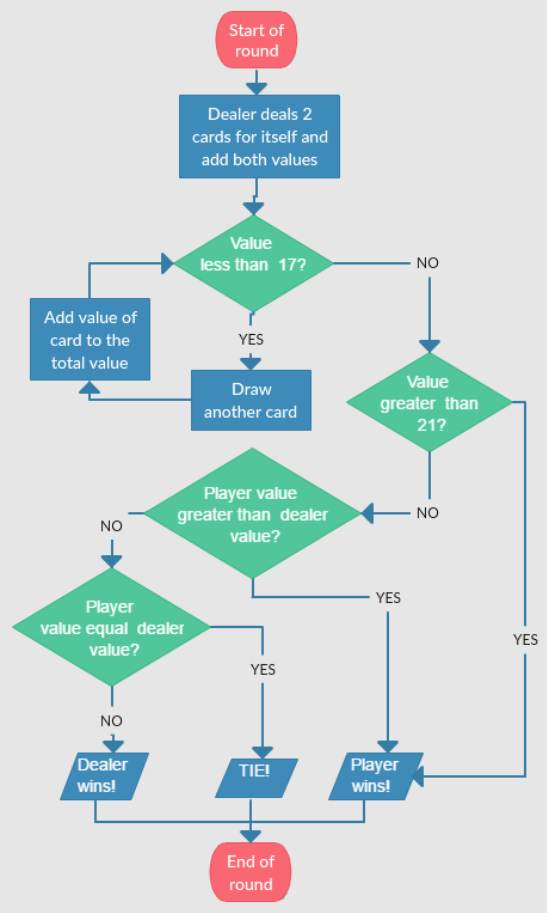


Fig 5.1.1.2: Flowchart

**5.2 UML DIAGRAMS:**

UML diagram is a diagram that is designed based on Unified Modelling Language with the aim to visually represent the system with roles, actors, anchors etc to understand and maintain the system easily. By using this we can better understand flaws or errors in the system so that we can maintain or alter the system properly.

Different types of UML diagrams include:

* + Class diagram
  + Activity diagram
  + Use case diagram
  + Sequence diagram
  + Component diagram and many more.

**5.2.1 USECASE DIAGRAM:**

Use case diagram also represented as behaviour diagrams. These diagrams are used to explain the set of actions that system need to be performed in accordance with the external user.

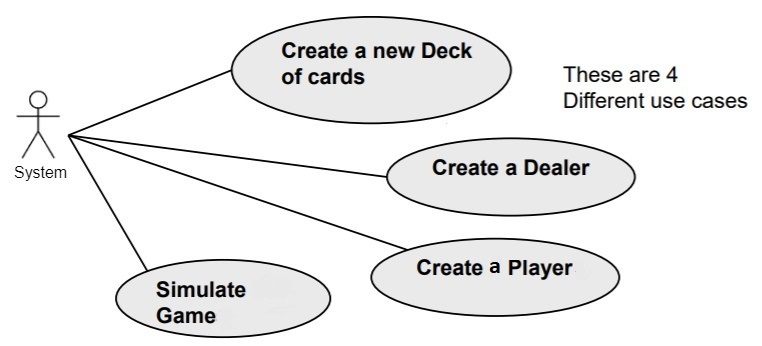


Fig 5.2.1: Use case Diagram

**5.2.2 COLLABORATION DIAGRAM:**

Collaboration diagram is a type of UML diagram. It is also known as communication diagram. Collaboration diagram is similar to flow chart. It provides us the visual representation of how different software objects interact within system architecture.

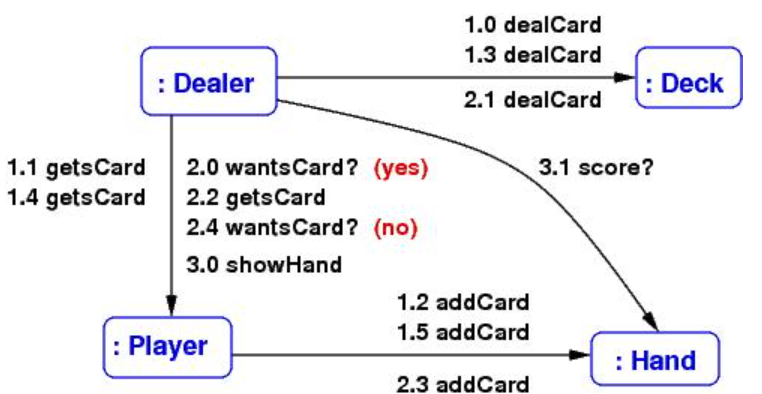
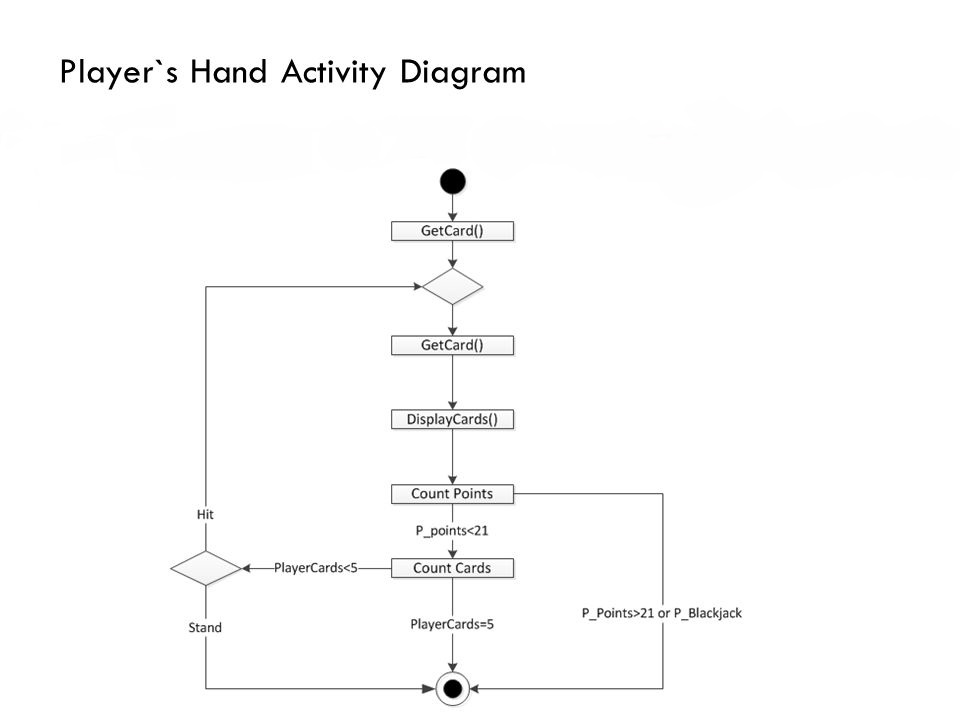
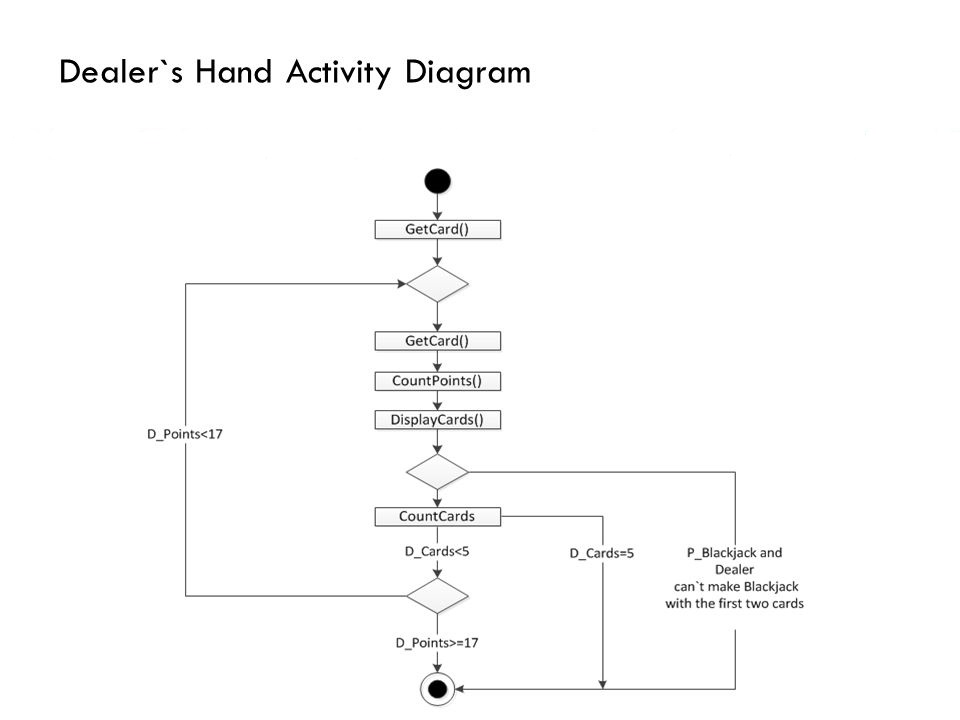


Fig 5.2.2: Collaboration Diagram

**5.2.3 ACTIVITY DIAGRAM:**

Activity diagram is one of the important UML diagrams, which is used to represent the dynamic aspects of the system. It is basically flow chart to represent the flow of activity from one activity the other.

****

****

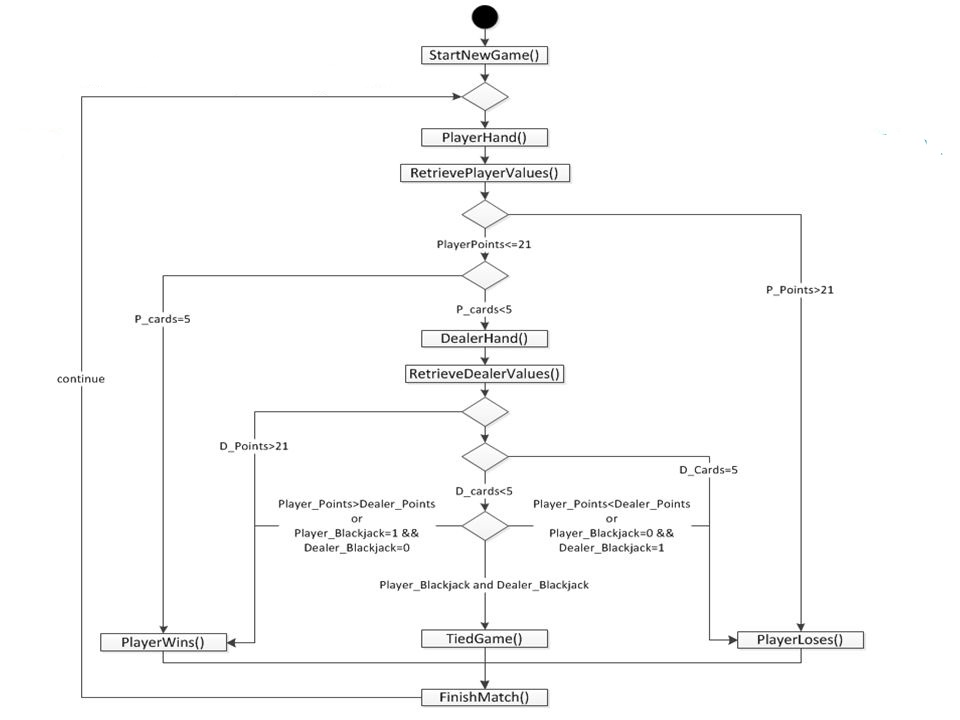
****

Fig 5.2.3: Activity Diagram

**5.2.4 CLASS DIAGRAM:**

Class diagram is a type of UML diagram that provides visually the overview of the architecture in the terms of classes and methods. It shows the relationship with various classes and their dependencies. It is similar to a flow chart in which classes are represented by using rectangular boxes. These boxes contain three fields namely name of the class, attributes of the class, methods related to that class.

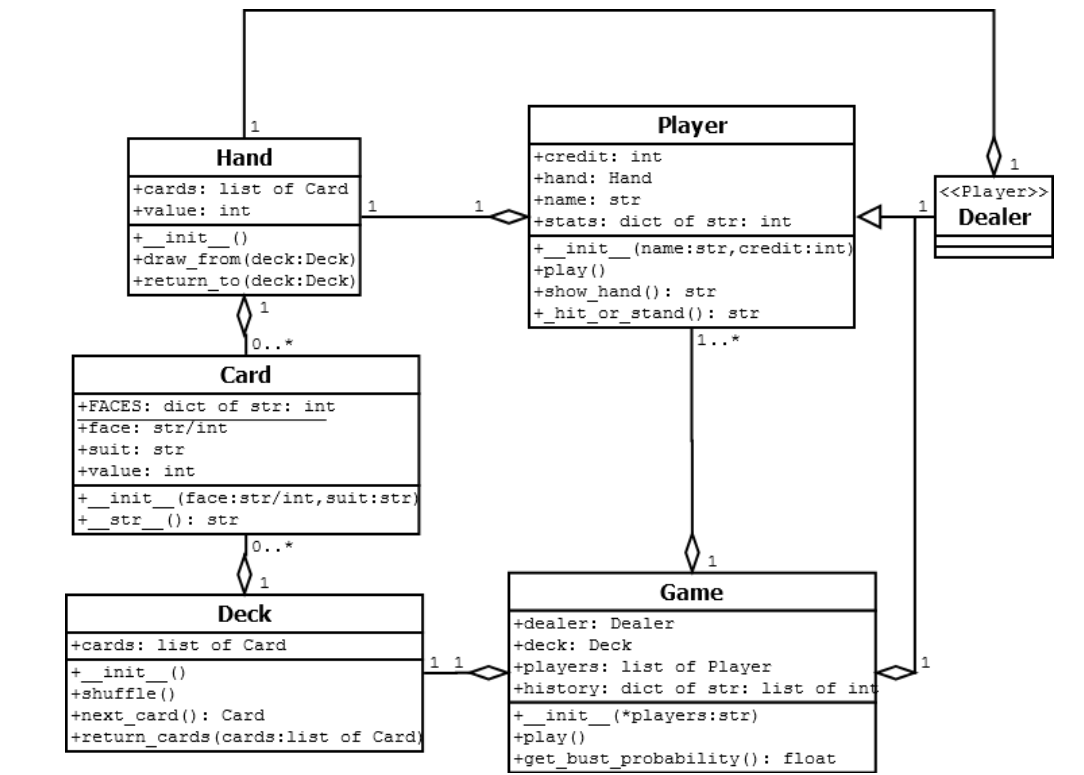
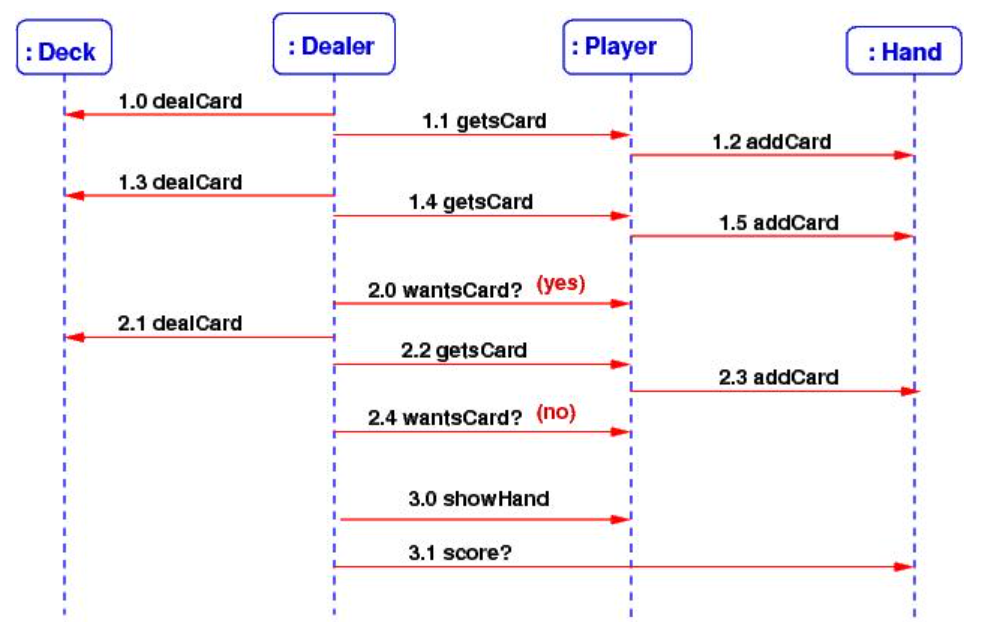


Fig 5.2.4: Class Diagram

**5.2.5 SEQUENCE DIAGRAM:**

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.



A more detailed version with an example testcase and it’s sequence diagram can be given as follows,

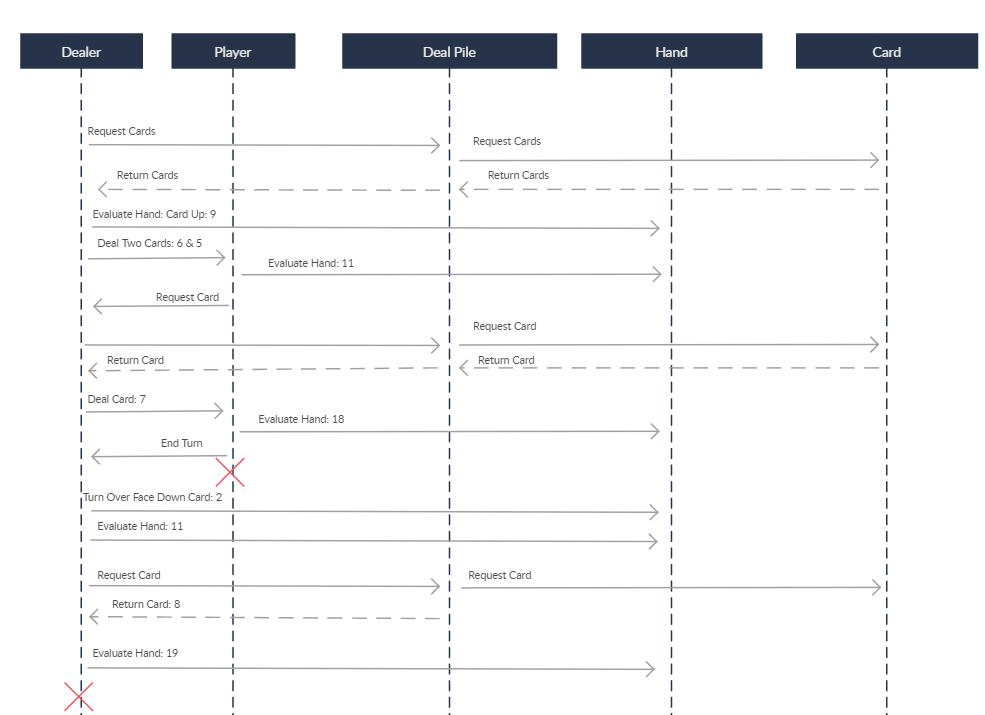


Fig 5.2.5: Sequence Diagram

**5.2.6 STATECHART DIAGRAM:**

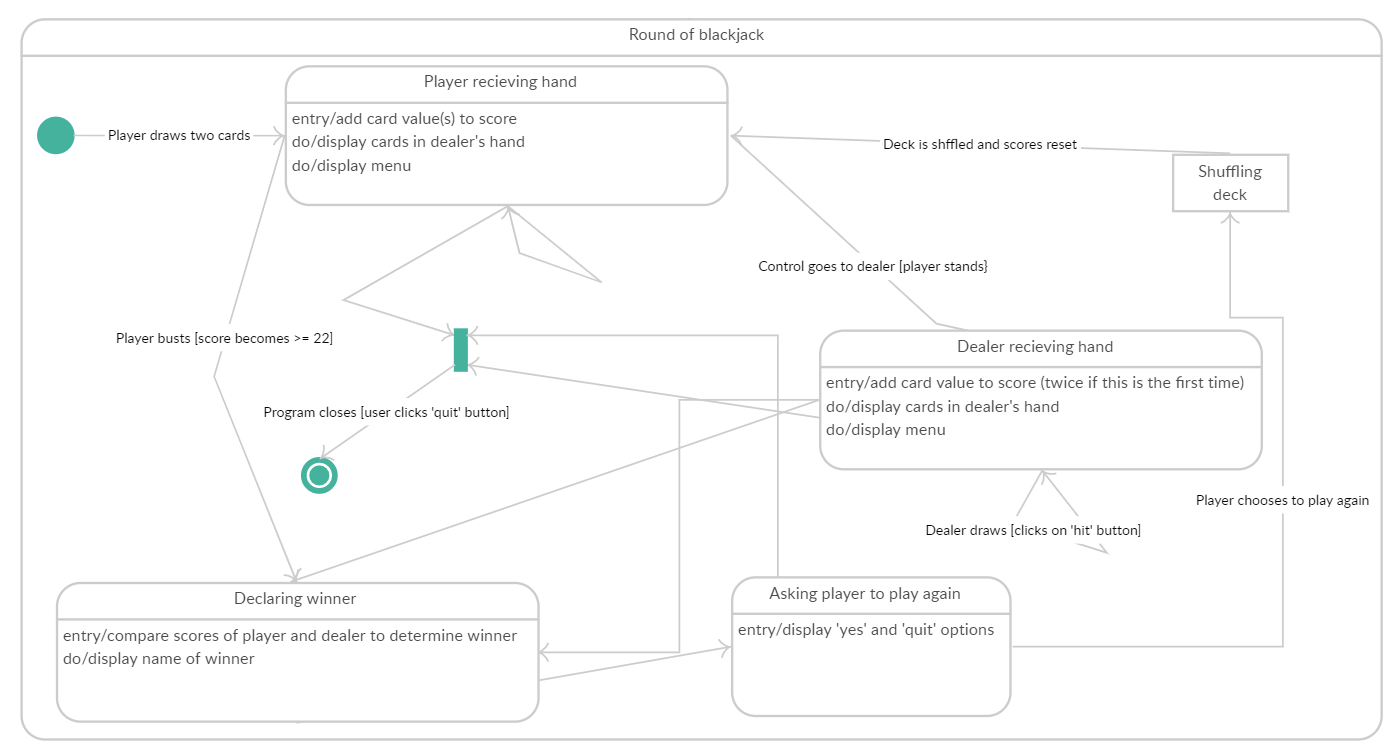
State chart diagram is used to describe the behaviour of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction.**

Fig 5.2.6: State Chart Diagram

**CHAPTER-6**

**IMPLEMENTATION**

The game starts off with a prompt asking the user to start the game. Once the game begins, an ASCII art of two playing cards and the word blackjack appears. Along with this, the user is shown his two cards and the face up card of the computer, all of which are randomly picked from an unlimited deck of cards containing cards from numbers 2 through 9 along with Ace, Jack, Queen and King. These are called face cards. The ace can either be counted as 1 or 11 based on the count. This feature is automated so, the user does not have to worry about it. Once the user is having the two cards, he is given a choice to either take another card, also called as “hit” or to stay with his current cards and wait for computer’s turn, also called as “stand”. At some point, either of them stops taking more cards. Then, their counts are compared. The one whose count exceeds 21, is immediately declared loser. If both are under 21, the one with highest value cards is declared the winner. If both have the same count, the game is tied and this called as “push”. There might also be a scenario where either the user or the computer has a card count of exactly 21. This is called “blackjack” and the one who possess these cards is immediately declared the winner irrespective of their opponent’s cards. When the user does not want to play the game anymore, he can express it in the prompt. Until then, he can play the game as many rounds as he wants to with renewed decks and cards each and every time picked randomly.

**6.1 PROPOSED METHODOLOGY:**

This application provides necessary options to serve users the utmost ease of use to play the game. The users will have complete control and each and every step of the game is highly transparent and interesting. The options that are provided by this application are:

* + Extra card retrieval or rejection
  + User access control
  + Cards and their count for user and computer
  + Maintain and update unique card counts for each game and many more

This phase is initiated after the system has been tested and accepted by the user. In this phase, the system is installed to support the intended functions. System performance is compared to performance objectives established during the planning phase. Implementation includes user notification, user training, installation of hardware, installation of software onto production computers, and integration of the system into daily work processes.

This phase continues until the system is operating in production in accordance with the defined user requirements.

**6.2 MODULE DESCRIPTION:**

Blackjack with Python is an online game based predominantly on Python. The purpose of this project was to develop a blackjack game in text-based format to assist in the betterment of my Python skills along with providing ability for users to play one of the better versions of the game with ease of use. This project includes mainly six modules as follows,

**6.2.1** **CARD DEALING MODULE:**

This module is responsible for the distribution of cards to either the player or the computer at any point of the game. Any random card can be drawn from unlimited and unbiased decks of 52 cards each and be given to whoever requests it.

**6.2.2 PLAYER/USER MODULE:**

This module is responsible for efficient management and storage of the values of all the cards currently being held by the user. It includes processes like keeping the total count of cards and displaying it to the user.

**6.2.3 DEALER/COMPUTER MODULE:**

This module is similar to the previous module except, this is used for the efficient management and storage of the cards held by the computer, keeping the total and displaying the face card to the user.

**6.2.4 GAME MODULE:**

The basic building block of the entire project is this module. In this module, the entire logic for playing the game exists. It will take care of all the possible testcases, their respective outputs. The logic under this module strictly follows each and every rule of the original blackjack game which results in an authentic experience.

**6.2.5 USER INTERACTION MODULE:**

This module plays a vital role. The function of the module can be understood from the name itself. The sole purpose of this module is to take prompts from the user and invoke the functions corresponding to those prompts precisely and efficiently.

**6.2.6 MAIN DISPLAY MODULE:**

From the name itself, we can understand that the task of this module is to manage what kind of display is shown to the user under what circumstances. Each scenario has a custom display which are invoked by this module precisely when those scenarios take place.

**CHAPTER-7**

**SOFTWARE ENVIRONMENT**

**7.1 PYTHON:**

Python is a high level, interpreted and general-purpose dynamic programming language that focuses on code readability. It has fewer steps when compared to Java and C. It was founded in 1991 by developer Guido Van Rossum. It is used in many organizations as it supports multiple programming paradigms. It also performs automatic memory management.

It is used for:

* Web development (server-side)
* Software development
* Mathematics
* System scripting

**7.1.1 What can Python do?**

* Python can be used on a server to create web applications.
* Python can be used alongside software to create workflows.
* Python can connect to database systems. It can also read and modify files.
* Python can be used to handle big data and perform complex mathematics.
* Python can be used for rapid prototyping, or for production-ready software development.

**7.1.2 Why Python?**

* Python has a simple syntax similar to the English language.
* Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
* Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
* Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
* Python can be treated in a procedural way, an object-oriented way or a functional way.

### 7.1.3 Python Syntax compared to other programming languages:

* Python was designed for readability, and has some similarities to the English language with influence from mathematics.
* Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
* Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

**7.1.4 It’s Open Source:**

Technically the point is that it is an open source project and they release patches often. **Support available**

Online Support is available for using Python. Along with this, Python has very large number of packages, libraries and pre-defined functions among others which makes life easier.

**7.2 PyCharm:**

**PyCharm** is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda. PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license.

**7.2.1 Features:**

* Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes.
* Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages.
* Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others.
* Support for web frameworks: Django, web2py and Flask.
* Integrated Python debugger.
* Integrated unit testing, with line-by-line code coverage.
* Google App Engine Python development.
* Version control integration. Support for scientific tools like Matplotlib, NumPy etc.

**7.3 Repl.it:**

Repl.it is an online IDE (integrated development environment). Its name comes from the acronym REPL, which stands for "read–evaluate–print loop". The service was created by Jordanian programmers Amjad Masad, Faris Masad, and Jordanian designer Haya Odeh, in 2016. Repl.it allows users to write code and build apps and websites using a browser. Additionally, Repl.it allows users to share projects. Interactive programming environments are referred to as "Repls" by the service. Users can create workspaces for projects in popular programming languages, granting them access to containers, in which the code is executed. Over 50 programming languages are supported, including popular languages such as Python, Ruby and Java.

**CHAPTER-8**

**SAMPLE CODE**

|  |
| --- |
|  |
|  |  |
|  | import random |
|  |  |
|  | cards = [11, 2, 3, 4, 5, 6, 7, 8, 9, 10, A, K, J] |
|  | choice = input("Do you want to play a game of Blackjack? Type 'y' or 'n': ").lower() |
|  |  |
|  | **# Card Dealing module** |
|  | def deal\_card(): |
|  | return random.choice(cards) |
|  |  |
|  | while choice == "y": |
|  | clear() |
|  | print(logo) |
|  |  |
|  | **# Player Module** |
|  | player\_cards = [] |
|  | player\_card1 = deal\_card() |
|  | player\_card2 = deal\_card() |
|  | if player\_card1 == 11 and player\_card2 == 11: |
|  | player\_card2 = 1 |
|  | current\_score = player\_card1 + player\_card2 |
|  | player\_cards.append(player\_card1) |
|  | player\_cards.append(player\_card2) |
|  | print(f"Your cards: {player\_cards}, current score: {current\_score}") |
|  |  |
|  | **# Dealer Module** |
|  | computer\_cards = [] |
|  | computer\_card1 = deal\_card() |
|  | computer\_score = computer\_card1 |
|  | computer\_cards.append(computer\_card1) |
|  | print(f"Computer's first card: {computer\_card1}") |
|  |  |
|  | computer\_next\_card = deal\_card() |
|  | if computer\_score > 10 and computer\_next\_card == 11: |
|  | computer\_cards.append(1) |
|  | computer\_next\_card = 1 |
|  | else: |
|  | computer\_cards.append(computer\_next\_card) |
|  | computer\_score += computer\_next\_card |
|  |  |
|  | **# Game Module** |
|  | computer\_blackjack = False |
|  | if computer\_score == 21: |
|  | computer\_blackjack = True |
|  | if current\_score == 21 and computer\_blackjack == True: |
|  | print("Computer and player have a blackjack!\n") |
|  | print("Push(draw game)!🤯") |
|  | break |
|  |  |
|  | stop\_game = False |
|  | if current\_score == 21: |
|  | print("You have a blackjack! You win!🌟") |
|  | choice = "n" |
|  | else: |
|  | **# User Interaction Module** |
|  | draw\_card = input("Type 'y' to get another card, type 'n' to pass: ") |
|  | while draw\_card == "y": |
|  | next\_card = deal\_card() |
|  |  |
|  | if current\_score > 10 and next\_card == 11: |
|  | player\_cards.append(1) |
|  | next\_card = 1 |
|  | else: |
|  | player\_cards.append(next\_card) |
|  | current\_score += next\_card |
|  |  |
|  | print(f"Your cards: {player\_cards}, current score: {current\_score}") |
|  | print(f"Computer's first card: {computer\_card1}") |
|  |  |
|  | if current\_score > 21: |
|  | print("You went over! Computer wins.😤") |
|  | stop\_game = True |
|  | draw\_card = "n" |
|  | elif current\_score == 21: |
|  | print("You have a blackjack! You win!🔥") |
|  | stop\_game = True |
|  | draw\_card = "n" |
|  | else: |
|  | draw\_card = input("Type 'y' to get another card, type 'n' to pass: ") |
|  |  |
|  | **# Main Display Module** |
|  | if stop\_game == False: |
|  | final\_score = 0 |
|  | for card in player\_cards: |
|  | final\_score += card |
|  | print(f"\nYour final hand: {player\_cards}, final score: {final\_score}") |
|  |  |
|  | while computer\_score < 17: |
|  | if computer\_score < final\_score: |
|  | computer\_next\_card = deal\_card() |
|  | if computer\_score > 10 and computer\_next\_card == 11: |
|  | computer\_cards.append(1) |
|  | computer\_next\_card = 1 |
|  | else: |
|  | computer\_cards.append(computer\_next\_card) |
|  | computer\_score += computer\_next\_card |
|  | else: |
|  | break |
|  | print(f"Computer's final hand: {computer\_cards}, final score: {computer\_score}\n") |
|  |  |
|  | if computer\_score > 21: |
|  | print("Computer went over! You win.😁") |
|  | choice = "n" |
|  | else: |
|  | if final\_score == computer\_score: |
|  | print("Push(draw game)!😎") |
|  | elif final\_score > computer\_score: |
|  | print("You win!😃") |
|  | else: |
|  | print("Computer wins!😭") |
|  |  |
|  | choice = input("Do you want to play a game of Blackjack? Type 'y' or 'n': ") |
|  | if choice == "y": |
|  | clear() |
|  | else: |
|  | print("Goodbye.") |

**CHAPTER-9**

**SYSTEM TESTING**

The main use of testing is to find out errors. Testing is the way toward attempting to find each possible flaw or shortcoming in a work item. It gives a way to deal with check the helpfulness of parts, sub-assemblies, social occasions just as a finished thing It is the path toward working on programming with the point of ensuring that the Software system satisfies its necessities and customer wants and does not bomb in an unacceptable manner. There are various sorts of test. Each test type keeps an eye on a specific testing need. Testing permits to expel the mistakes and improve the framework execution. There are numerous kinds of tests which enables us to improve our venture execution and to make it mistake free. What's more we likewise have tests which encourage us to check singular modules autonomously and furthermore to check complete framework together according to our convenience.

**9.1 TYPES OF TESTS:**

**9.1.1 UNIT TESETING:**

Unit testing incorporates the arrangement of analyses that favour that within program basis is working properly, and that program information sources produce significant yields. It checks whether little segments are working appropriately or not. Every single decision branch and inside code stream should be endorsed. It is the attempting of individual programming units of the application .it is done after the completion of an individual unit before fuse. This is an auxiliary attempting, that relies upon learning of its improvement and is prominent. Unit tests perform fundamental tests at section level and test a specific business system, application, or possibly structure plan. Unit tests ensure that all of a thoughtful method for a business technique performs unequivocally to the recorded points of interest and contains obviously portrayed data sources and foreseen results.

A unit test encourages you to discover which part is broken in your application and fixes it quicker.

**9.1.2 INTEGRATION TESTING:**

Integration tests are expected to test joined programming modules to choose whether they everything considered continue running as one program. Testing is an event driven and is dynamically stressed over the crucial after effect of screens or fields. Combination tests show that in spite of the way that the sections were autonomously satisfied, as showed up by successfully unit testing, the gathering of portions are correct and unsurprising. Combination testing is expressly away for revealing the issues that rise up out of the gathering of these portions.

Integration testing permits to discover blunders because of unexpected communication between the framework and the sub-framework segments. We test the product in order to test and to identify all the potential mistakes in our undertaking once we complete the source code and before conveying it to the clients.

The techniques for performing tests. These techniques provide guidance for testing:

* To test the internal logic of the software components.
* To test the input and output domains of a programs and to uncover the errors in program function, behaviour and performance.

We can test the software by using two methods:

* White Box testing: In this the internal logic program is being checked by using different test case design techniques.
* Black Box testing: In this the software requirements are tested by using different test case design techniques.

Both the above-mentioned techniques help in finding out the maximum number of errors with minimal time and effort.

## 9.1.3 PERFORMANCE TESTING:

This test is done to find the run-time performance of the software with the context of the integrated system. These tests can be carried out throughout the testing process. For example, the performance of individual modules are accessed during white box testing under unit testing.

## 

## 9.1.4 VERIFICATION AND VALIDATION:

Testing procedure is a piece of subject alluding to checking and approval of our task. We have to find the framework determinations and we should attempt to meet the details of the client and to fulfil the client, for this reason, we need to check and approve the item and we have to ensure that everything is working appropriately. Check and approval are the two unique things. One is performed to guarantee that the product is working accurately and to implement a particular usefulness and the other is done to guarantee if the client prerequisites are appropriately met or not by the finished result.

Check is progressively similar to 'would we say we are building the item, right?' and approval is increasingly similar to 'would we say we are building the correct item?'.

**CHAPTER-10**

**RESULTS AND OUTPUT SCREENS**

**10.1 PROMPT:**

This is the beginning of the project. At this stage, the user is asked if they would like to play the game. On clicking “y”, they are redirected to the actual starting point of the game.

****

Figure 10.1: Prompt

**10.2 GAME START:**

Here, an ASCII art of blackjack is displayed along with the first two cards of the user and the single card of the computer. Based on this, the user should type his decision to proceed or stay with their cards.

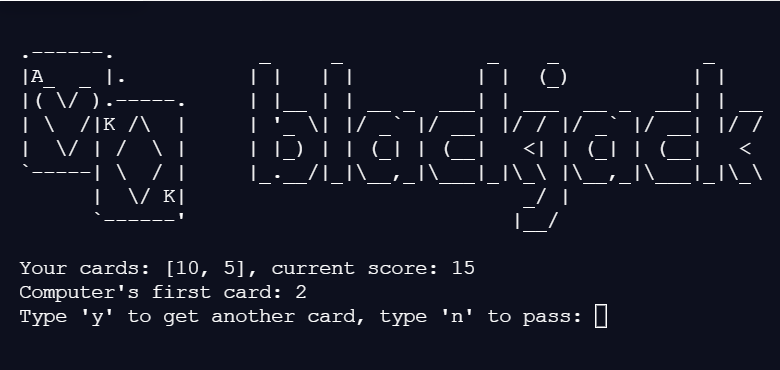
****

Figure 10.2: Game Start

**10.3 USER WINS:**

The details of the user and computer are updated at every stage of the game. When the user wins, either directly or if the computer crosses 21, all the cards of both the user and computer are shown and the display indicates that the user has won.



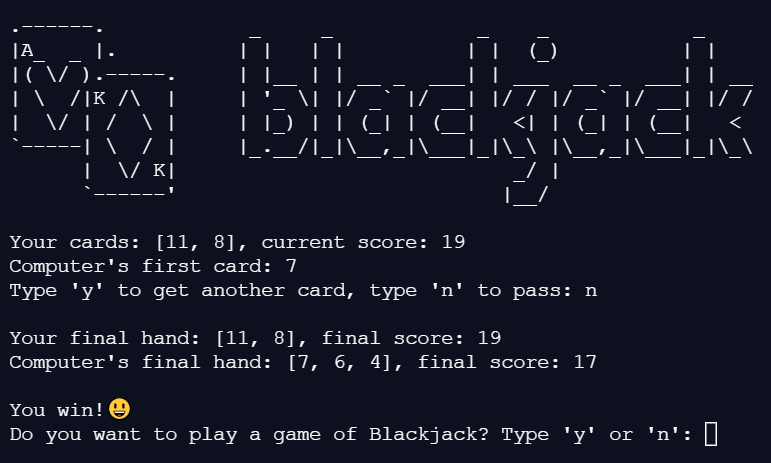
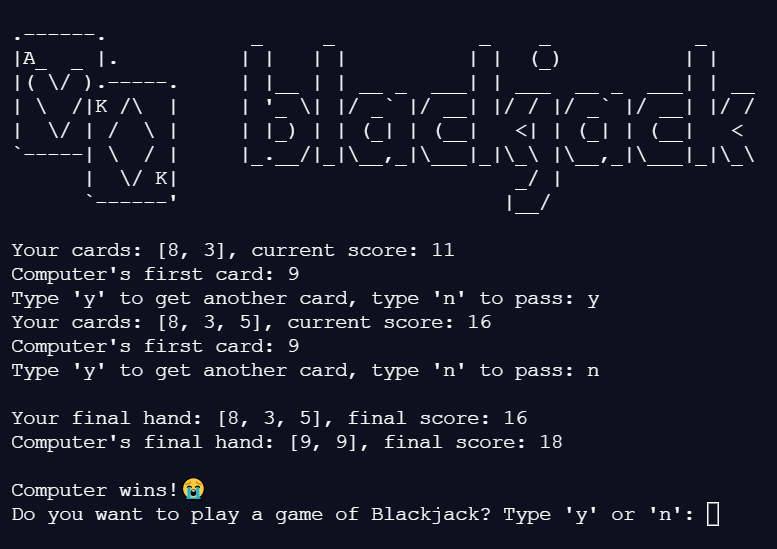


Figure 10.3: User Wins

**10.4 COMPUTER WINS:**

The details of the user and computer are updated at every stage of the game. When the computer wins, either directly or if the user crosses 21, all the cards of both the user and computer are shown and the display indicates that the computer has won.



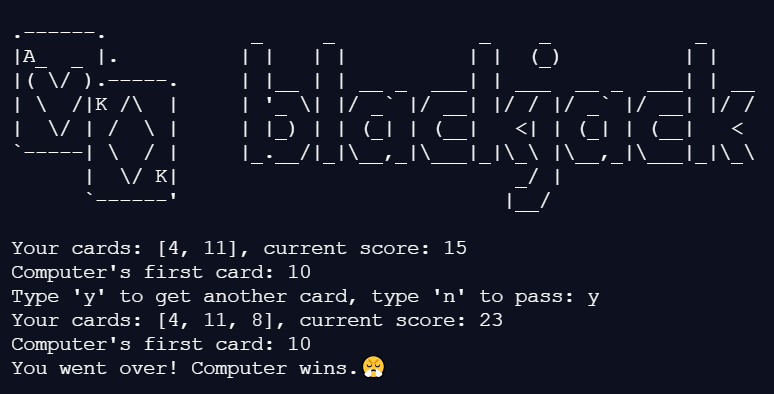


Figure 10.4: Computer Wins

**10.5 DRAW GAME:**

When the user decides to stay with his cards and computer too ends up with the cards having the same count as the user, a draw game occurs. In Blackjack terms, this scenario is also known as a push. The probability of this kind of scenario arising is very low.

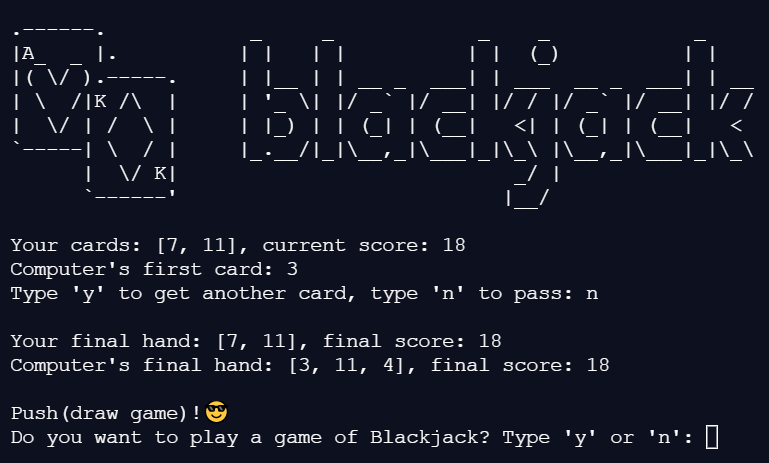


Figure 10.5: Draw Game

**10.6 BLACKJACK:**

At any stage of the game, if either the user or the computer holds cards resulting in a value count of the exact number 21, it is said that they have a blackjack. According to the rules, whenever this happens, the holder of these cards is immediately declared as the winner irrespective of their opponent’s cards.

****

****

Figure 10.6: Blackjack

**10.7 GAME END:**

When the user does not want to play the game anymore, they can simply type “n” into the prompt and game stops immediately with a goodbye.

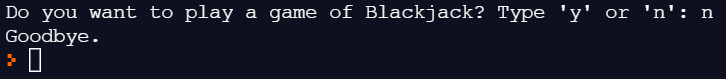
****

Figure 10.7: Game End

**CHAPTER-11**

**CONCLUSION**

Technology is introducing new innovations day by day, thus reducing the time required to do things. The proposed system can be used to reduce the time taken to play the game as it is only text-based and does not include any heavy loads like animations. This application can be used by the people interested in generally playing the game or people with compulsive behaviours towards these games. The application provides a way of communication and synchronization between the user and the computer. Along with this, the application can also simulate the events of the real game like storing values of all cards in play and determining the winner based on that among many others.

With the theoretical inclination of our syllabus it becomes very essential to take the at most advantage of any opportunity of gaining practical experience that comes along. The building blocks of this Mini Project “BLACKJACK WITH PYTHON” was one of these opportunities. It gave me the requisite practical knowledge to supplement the already taught theoretical concepts thus making me more competent as a computer science engineer. The project from a personal point of view also helped me in understanding the following aspects of project development:

* The planning that goes into implementing a project.
* The importance of proper planning and an organized methodology.
* The learning of Python programming language and co-ordination of different facets within a successful project.

The project also provided the opportunity of interacting with our professors and to gain from their best experience.

**REFERENCES**

[1] [www.wikipedia.com](http://www.wikipedia.com)

[2] [www.python.org](http://www.python.org)

[3] [www.w3schools.com](http://www.w3schools.com)

[4] [www.washingtonpost.com](http://www.washingtonpost.com)

[5] [www.blackjackapprenticeship.com](http://www.blackjackapprenticeship.com)