

**A
Project
On
“Learn with Fun”**

Submitted to

Shiksha Mandal's

**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)**

In the Partial Fulfillment Of

B.Com. (Computer Application) Final Year

Submitted by

Shreya S. Joshi

Joice D. Awsare

Under the Guidance of

Pravin J. Yadao



Shiksha Mandal's

**G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)2021-202**

Shiksha Mandal's
G. S. COLLEGE OF COMMERCE & ECONOMICS, NAGPUR
(AUTONOMOUS)

CERTIFICATE

(2021 - 2022)

This is to certify that Mr. /Miss Shreya S. Joshi and Joice D. Awsare has completed their project on the topic of Learn with Fun prescribed by G. S. College of Commerce & Economics, Nagpur (Autonomous) for B.Com. (Computer Application) – Semester-VI.

Date:

Place: Nagpur

Pravin J. Yadao

Project Guide

External Examiner

Internal Examiner

ACKNOWLEDGEMENT

We take this opportunity to express our deep gratitude and whole hearted thanks to project guide Prof. Pravin Yadao, Coordinator for his guidance throughout this work. We are very much thankful to him for his constant encouragement, support and kindness.

We are also grateful to our teachers Prof. Rahul Tiwari, Prof. Sushma Gawande, Prof. Preeti Rangari, Prof. Prajkta Deshpande and Prof. Haresh Naringe for their encouragement, help and support from time to time.

We also wish to express our sincere thanks to Principal Dr. N. Y. Khandait for providing us wide range of opportunities, facilities and inspiration to gather professional knowledge and material without which this project could not have been completed.

Date:

Place: Nagpur

Shreya S. Joshi

Joice D. Awsare

Declaration

We **Miss. Shreya S. Joshi & Joice D. Awsare** here by honestly declare that the work entitled “**Learn with Fun**” submitted by us at G. S. College of Commerce & Economics, Nagpur (Autonomous) in partial fulfillment of requirement for the award of B.Com. (Computer Application) degree by Rashtrasant Tukadoji Maharaj, Nagpur University, Nagpur has not been submitted elsewhere for the award of any degree, during the academic session 2021-2022.

The project has been developed and completed by us independently under the supervision of the subject teacher and project guide.

Shreya S. Joshi

Joice D. Awsare

Date:

Place: Nagpur

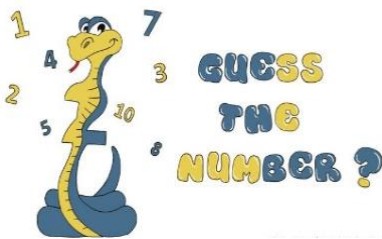
INDEX

Sr. No	Particulars	Page No	Sign	Remark
1.	INTRODUCTION	6-8		
2.	OBJECTIVES	9-11		
3.	PRELIMINARY SYSTEM ANALYSIS 3.1 Preliminary Investigation 3.2 Present System In Use 3.3 Flaws In Present System 3.4 Need For New System 3.5 Feasibility Study 3.6 Project Category	12-26		
4.	SOFTWARE AND HARDWARE REQUIREMENT SPECIFICATION	27-28		
5.	DETAILED SYSTEM ANALYSIS 5.1 Data Flow Diagram 5.3 Data System Analysis 5.4 Entity Relationship Diagram	29-31		
6.	SYSTEM DESIGN 6.1 Source Code 6.2 Input Screen and Output Screen	32-90		
7.	TESTING AND VALIDATION CHECKS	91-95		
8.	SYSTEM SECURITY MEASURES 8.1 Implementation 8.2 Evaluation 8.3 Maintenance	96-104		
9.	FUTURE SCOPE OF THE PROJECT	105-106		
10.	CONCLUSION	107-108		
11.	BIBLIOGRAPHY & REFERENCES	109-110		

INTRODUCTION

Now a days gaming has become a very effective way to make people learn something new and improve their capabilities. Gaming has created its own huge world today. People play it as fun, entertainment also to learn.

In our application we have introduced different types of games which are very helpful in many ways . They are competitive and helps to build our confidence. Games are becoming more user friendly as the time goes on. Smartphone users account for about 40% of the mobile phone market, considering the awareness among youngsters and the current trend as well as the cheaper rates. Many of these users enjoy playing games on their mobile phones. Having your favourite game in your mobile devices is in itself a mark of remarkable gesture.



Number guessing game is a game where you have to guess any number between 1 to 100, it will give you hints by telling, and your guessed number is greater or less than the correct number. You will get 6 chances to guess the right one.

Guessing games help you to think about numbers, their approximations and various combinations. A proper workout to the brain can be ensured through guessing games. These fun-filled, educative games come with plenty of interactive activities which makes math learning exciting.



Kid’s learning section is very interesting section for kids. It has colour guessing game, rhyming words, maths game and memory game. Kids will explore their knowledge through this application.

Memory power, this game shows different pictures and we need to find out the correct from the given options which will help to make kids memory stronger. Maths game will take out the fear of maths and kids will learn to solve mathematics like addition, subtraction, multiplication, division in this game. Colour guessing game will help kids to memorize colour’s name. Rhyming word, in this game different rhythms for particular word will be given.

They have to just choose whichever section they wanted to chose and that will open and they may just seat and watch and learn through it.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Sudoku is the Japanese word for "placement puzzle". The Sudoku puzzle consists of a series of grids.

The grids include one large 9 x 9 grid that houses, nine 3 x 3 smaller grids. The purpose of the game is to place a number from 1-9 in each of the grid cells.

You don't have to worry about finding the sum of the numbers of the rows, columns, like in Magic Squares

OBJECTIVES

- **Improve Cognitive Skills**

This application is mainly to boost cognitive skills which will help to think logically and take your every decision smartly. One has to constantly find ways and solutions to win the game; its the best way to exercise your brain.

- **Boost Concentration**

It is very important to concentrate on particular things. Whether you are studying or doing anything, having good concentration power is so important. This applications will help you to improve your concentration which will definitely help you to stay focused on a particular work.

- **Behavioural Improvement**

This application will mainly builds a behaviour of confidence and finding ways in difficulties. It will help to make a solution in different aspects of daily life. It will help to improve your behaviour.

- **Enhance Creativity**

This application focuses on enhancing your creativity . It will help you to think in innovative ways. And it aims to help you to challenge yourself to make creative opportunities to find your ways.

- **Improve thinking skills**

Learning to think critically may be one of the most important skills that today's children will need for the future. This application aims to improve thinking skills and to think critically.

- **Improves memory**

The more challenges you face, the quicker you will drop these notes as your brain will retain the information naturally. Through this application you will boost your memory power.

- **Improves motor skills**

This application will help your kid to be more active in their life and in play grounds. They will actively participate in other outdoor games too.

- **Improves logical thinking**

This application will help you to improve your logical thinking as the games will level up and go on harder and harder.

- **Gives a sense of accomplishment**

When a player successfully finishes the level infuses a player with the sense of accomplishment and satisfaction and confidence. Harder the game goes more of the joy players feels after completing it successfully.

- **Helps kids develop their problem-solving skills**

This application will definitely help your kid to develop a behaviour of problem solving. This application will help you kid inculcate this attitude which will help your kid in their life too.

- **Vocabulary Boosters**

This application will help kids for fun and to learn simultaneously. Poems and alphabets will help your kid to learn new vocabularies.

PRELIMINARY SYSTEM
ANALYSIS

Preliminary Investigation

Preliminary investigation is the first step. In this step, the system is investigated. The objective of this step is to conduct an initial analysis and findings of the system.

While searching for an idea we had only one think in our mind that it must be useful for the user. So we come up with an idea to build an application with different types of games so it may explore user's mind. It may help user not only in their present but also for future.

Present System In Use

Present system is all about enhancing the knowledge and skills of a child. Also helps your kid to learn new vocabularies and also to boost their confidence. It contains number guessing game, kids section and solving Sudoku.

Kids section have more categories like alphabets, poems and rhythms. With alphabet pictures will be shown and with poems sound effective will go on. This app will be easy to use and understand.

Flaws In Present System

In present system it might be difficult for younger kids to play number guessing game and Sudoku. They may find it difficult to play and understand so might be they will not be interested in playing that games.

Same like that elder children will be not interested in kids section because it has alphabets, poems and rhythms so they might be not interested in that section. Present system does not provide videos for anything.

Need For New System

Current application is somewhere has its limitations. With a new system we may fill all the required requirements which current application doesn't have.

It has limited games and limited things for kids but with new system we may provide more. We may be giving videos more frequency and also fast service.

With new system we can also improve security and store large amount of data.

Feasibility Study

As name suggests feasibility study is the feasibility analysis or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyse whether software product will be right in terms of development, implantation, contribution of project to the organization etc.

I. Technical Feasibility

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves the evaluation of the hardware, software, and other technical requirements of the proposed system. As an exaggerated example, an organization wouldn't want to try to put Star Trek's transporters in their building—currently, this project is not technically feasible.

II. Economic Feasibility

This assessment typically involves a cost/ benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility—helping decision-makers determine the positive economic benefits to the organization that the proposed project will provide.

III. Social Feasibility:

Social Feasibility is a determination of whether this proposed system will be acceptable to the people or not. This determination typically examines the probability of the project accepted by the group directly affected by the proposed system change.

- It describes the effect on users from the introduction of the new system considering whether there will be need for retraining the workforce.
- It describes how you propose to ensure user co-operation before changes are introduced.

Project Category

The Project Category is an application called “Learn with Fun”. This application is designed using python language and CSS. From starting page to ending page we have many pages linked together containing different information. Our application is an gaming and kids application which helps them for fun and learn simultaneously.

➤ **Programming Language Used In Project**

Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include –

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

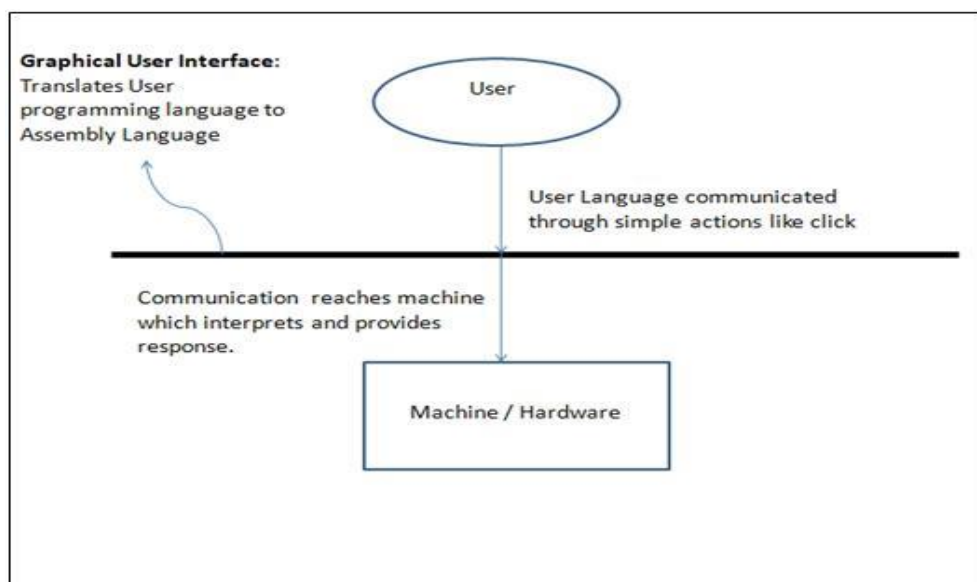
List of why Python is popular:

- The Python framework also has modules and packages, which facilitates code reusability.
- Python is open source. You can download it for free and use it in your application. You can also read and modify the source code.
- No Compilation of the code – the cycle of Edit-test-debug is fast
- Supports exception handling. Any code is prone to errors. Python generates exceptions that can be handled hence avoids crashing of programs.
- Automatic memory management. Memory management in Python involves a private heap(a data structure that represents a queue) containing all Python objects and data structures.

GUI

The graphical user interface is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation, instead of text-based user interfaces, typed command labels or text navigation. GUIs were introduced in reaction to the perceived steep learning curve of command-line interfaces (CLIs), which require commands to be typed on a computer keyboard.

The actions in a GUI are usually performed through direct manipulation of the graphical elements. Beyond computers, GUIs are used in many handheld mobile devices such as MP3 players, portable media players, gaming devices, smartphones and smaller household, office and industrial controls. The term GUI tends not to be applied to other lower-display resolution types of interfaces, such as video games (where head-up display (HUD) is preferred), or not including flat screens, like volumetric displays because the term is restricted to the scope of two-dimensional display screens able to describe generic information, in



the tradition of the computer science research at the Xerox Palo Alto Research Centre.

A GUI uses a combination of technologies and devices to provide a platform that users can interact with, for the tasks of gathering and producing information.

A series of elements conforming a visual language have evolved to represent information stored in computers. This makes it easier for people with few computer skills to work with and use computer software. The most common combination of such elements in GUIs is the windows, icons, text fields, canvases, menus, pointer (WIMP) paradigm, especially in personal computers.

The WIMP style of interaction uses a virtual input device to represent the position of a pointing device's interface, most often a mouse, and presents information organized in windows and represented with icons. Available commands are compiled together in menus, and actions are performed making gestures with the pointing device. A window manager facilitates the interactions between windows, applications, and the windowing system. The windowing system handles hardware devices such as pointing devices, graphics hardware, and positioning of the pointer.

In personal computers, all these elements are modelled through a desktop metaphor to produce a simulation called a desktop environment in which the display represents a desktop, on which documents and folders of documents can be placed. Window managers and other software combine to simulate the desktop environment with varying degrees of realism.

Entries may appear in a list to make space for text and details, or in a grid for compactness and larger icons with little space underneath for text. Variations in between exist, such as a list with multiple columns of items and a grid of items with rows of text extending sideways from the top.

Multi-row and multi-column layouts commonly found on the web are "shelf" and "waterfall". The former is found on image search engines, where images appear with a fixed height but variable length, and is typically implemented with the CSS property and parameter `display: inline-block;`. A waterfall layout found on Imgur and Tweetdeck with fixed width but variable height per item is usually implemented by specifying column-width.

MySQL

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr,^[12] MediaWiki Twitter and YouTube.

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL 5.6:

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM

- Triggers
- Cursors
- Updatable views
- Online Data Definition Language (DDL) when using the InnoDB Storage Engine.
- Information schema
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.[80]
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine
- Transactions with save points when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines
- SSL support
- Query caching
- Sub-SELECTs (i.e. nested SELECTs)
- Built-in replication support
 - Asynchronous replication: master-slave from one master to many slaves or many masters to one slave
 - Semi synchronous replication: Master to slave replication where the master waits on replication
 - Synchronous replication: Multi-master replication is provided in MySQL Cluster

- Virtual Synchronous: Self managed groups of MySQL servers with multi master support can be done using: Galera Cluster or the built in Group Replication plugin[89]
- Full-text indexing and searching[b]
- Embedded database library
- Unicode support[a]
- Partitioned tables with pruning of partitions in optimizer
- Shared-nothing clustering through MySQL Cluster
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.[c]
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

The developers release minor updates of the MySQL Server approximately every two months. The sources can be obtained from MySQL's website or from MySQL's GitHub repository, both under the GPL license.

Software And Hardware **Requirement Specification**

Hardware

Hardware, which is abbreviated as HW, refers to all physical components of a computer system, including the devices connected to it. You cannot create a computer or use software without using hardware. The screen on which you are reading this information is also a hardware.

- RAM :4GB and Above
- Hard disk : 320 GB and Above
- Keyboard
- Mouse

Software

Software is a set of instructions, data, or programs used to operate a computer and execute specific tasks. In simpler terms, software tells a computer how to function. It's a generic term used to refer to applications, scripts, and programs that run on devices such as PCs, mobile phones, tablets, and other smart devices. Software contrasts with hardware, which is the physical aspects of a computer that perform the work.

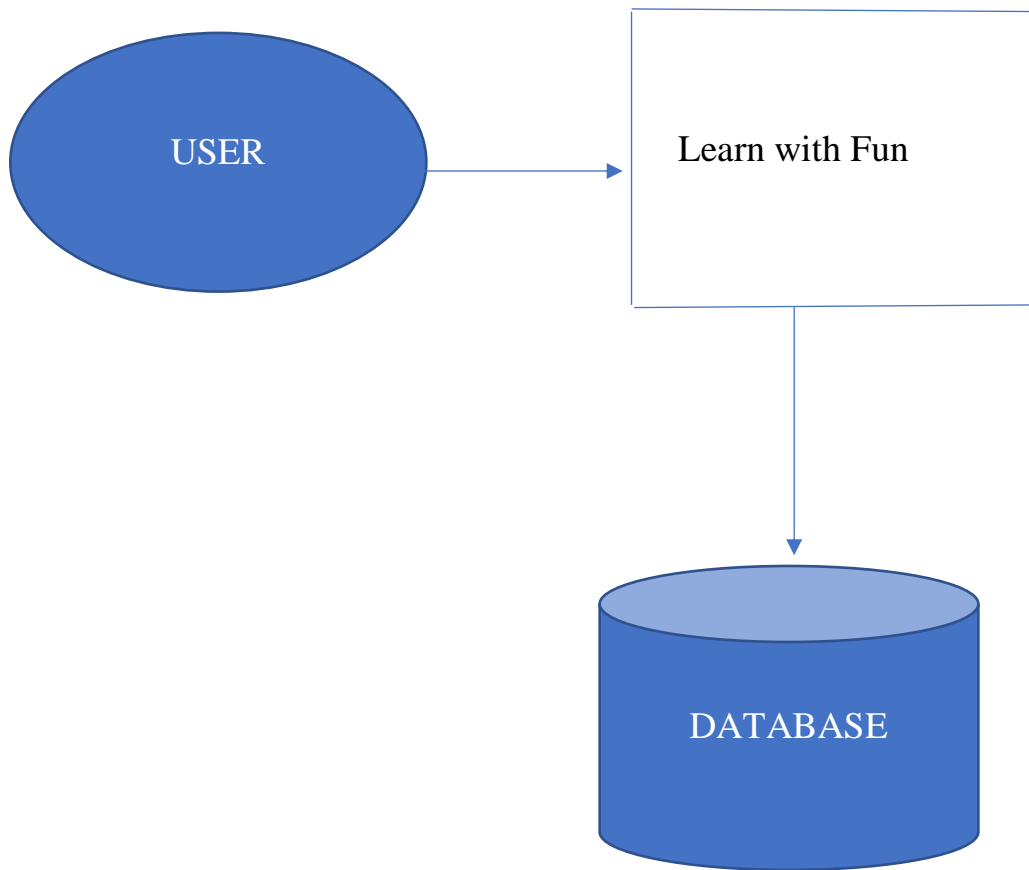
- Microsoft Window 10

Operating System

- Windows 10

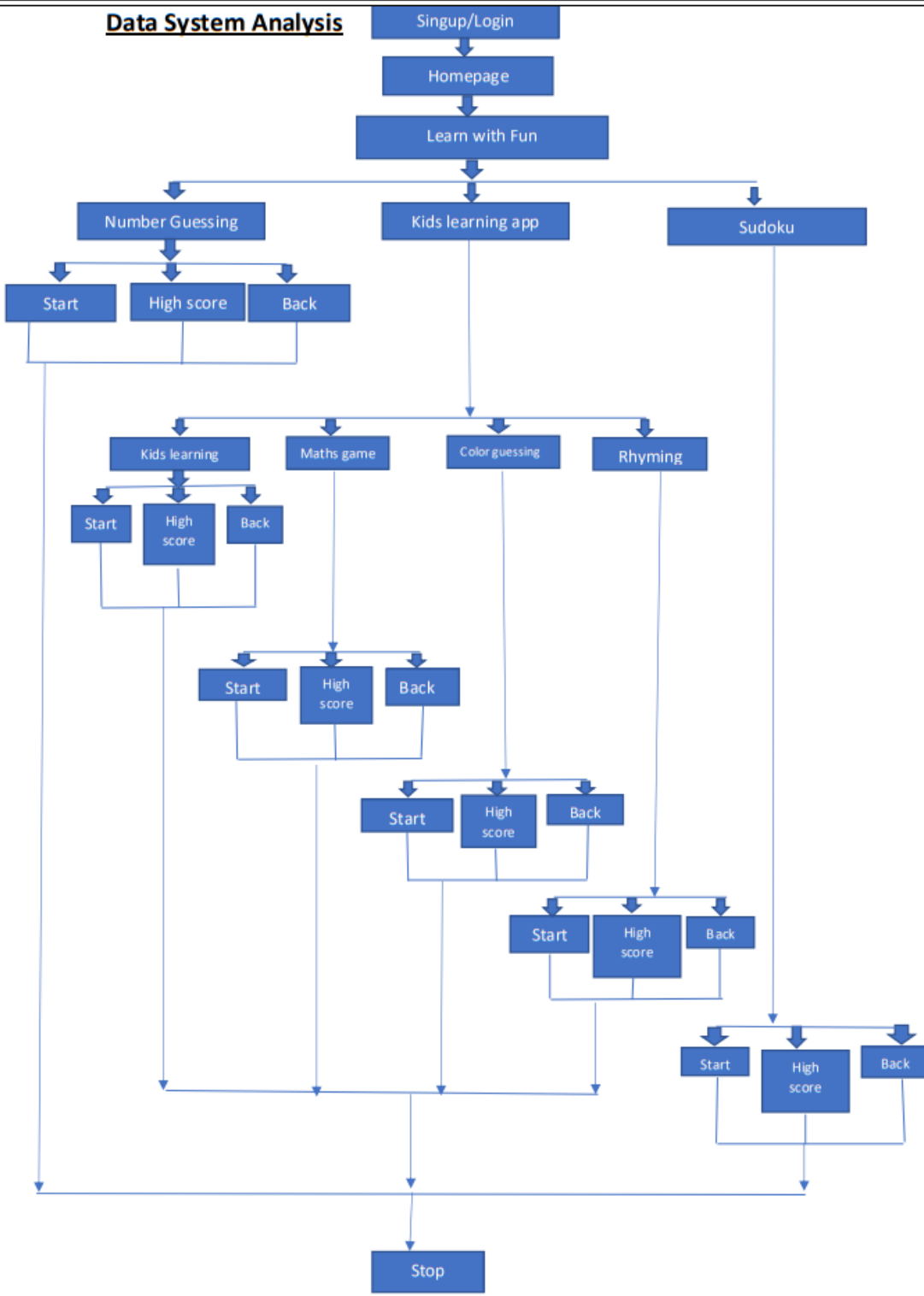
Detailed System Analysis

- **Data flow diagram:-**



- **Data System Analysis:**

Data System Analysis



31

SYSTEM DESIGN

- **Source code**

Sign up:-

```
#---Import Librarires
from tkinter import *
from tkinter import ttk
from tkinter import messagebox

#----Functions/Actions

def validation():
    if name.get()!="" and email.get()!="" and password.get()!="" and
gender.get()!="" and age.get()!="":
        email_check=email_validation()
        password_check=password_validataion()
        duplicate_check=duplicates()
        if email_check:
            if duplicate_check:
                messagebox.showinfo("information","User Email Already Exist")
            elif password_check:
                insertion_data()
                messagebox.showinfo("information","Data Inserted Sucessfully")
                shifting_form()
            else:
                messagebox.showerror("Error","Password should be (max 7
characters ,upper and lower case letter , number and symbol)")
            else:
                messagebox.showerror("Error","Email Fromat Is Not invalid")
        else:
            messagebox.showerror("Error","Fill All The Required Feilds")

def shifting_form():
    screen.destroy()
    import login

def insertion_data():
```

```
with open("insertion.txt",'+a') as wr:
```

```
wr.write(f"{name.get()},{email.get()},{password.get()},{gender.get()},{age.get()}\n")
```

```
def email_validation():
```

```
    temp = email.get()
```

```
    count=-1
```

```
    for i in temp:
```

```
        count+=1
```

```
        if i=="@":
```

```
            if temp[count:len(temp)]=="@gmail.com":
```

```
                return True
```

```
            else:
```

```
                return False
```

```
    else:
```

```
        return False
```

```
def password_validataion():
```

```
    temp = password.get()
```

```
    if len(temp)>=8:
```

```
        a,b,c,d=False,False,False,False
```

```
        #use multi variable for checking pass_validation
```

```
        for i in temp:
```

```
            x = ord(i)
```

```
            if x>=65 and x<=90:
```

```
                a=True
```

```
            elif x>=97 and x<=122:
```

```
                b=True
```

```
            elif x>=48 and x<=57:
```

```
                c=True
```

```
            else:
```

```
                d=True
```

```
        if a and b and c and d:
```

```
            return True
```

```
        else:
```

```
            return False
```

```
    else:
```

```

        return False
def duplicates():
    with open("insertion.txt") as rd:
        data = rd.readline()
        while data!="":
            collection = data.split(",")
            if collection[1]==email.get():
                return True
            data = rd.readline()
        else:
            return False

#---Graphical Interface
screen = Tk()
#---variables
name = StringVar()
email = StringVar()
password = StringVar()
gender = StringVar()
age = StringVar()
    #screen-ui
screen.geometry("700x600")
screen.maxsize(width="700",height="600")
screen.minsize(width="700",height="600")
screen.config(bg="deep sky blue")
screen.title("Registration")
    #Signuop Title
title = Label(text="SIGN
UP",font=("Arial",70,"bold"),padx="5",pady="5",bg="deep sky
blue",fg="yellow2").pack(pady="50")
    #signup frame
signup_frame = Frame(screen,width="500",height="400",bg="yellow2")
signup_frame.place(x="140",y="150")
    # signup frame widgets

#widget 1

```

```

fullname_label =
Label(signup_frame,width="15",padx="5",pady="5",text="User
Name",bg="yellow2",fg="deep sky
blue",font=("Calibri",15,'bold')).grid(row=0,column=0,padx="5",pady="5")
fullname_entry =
Entry(signup_frame,textvariable=name,selectbackground="deep sky
blue",selectforeground="black",font=("Calibri",15,"italic")).grid(row=0,colu
mn=1,padx="15",pady="5")
#widget 2
email_label =
Label(signup_frame,width="15",padx="5",pady="5",text="User
Email",bg="yellow2",fg="deep sky
blue",font=("Calibri",15,'bold')).grid(row=1,column=0,padx="5",pady="5")
email_entry =
Entry(signup_frame,textvariable=email,selectbackground="deep sky
blue",selectforeground="black",font=("Calibri",15,"italic")).grid(row=1,colu
mn=1,padx="15",pady="5")
#widget 3
password_label =
Label(signup_frame,width="15",padx="5",pady="5",text="User
Password",bg="yellow2",fg="deep sky
blue",font=("Calibri",15,'bold')).grid(row=2,column=0,padx="5",pady="5")
password_entry =
Entry(signup_frame,show="*",textvariable=password,selectbackground="de
ep sky
blue",selectforeground="black",font=("Calibri",15,"italic")).grid(row=2,colu
mn=1,padx="15",pady="5")
#widget 4
gender.set("Radio")
gender_label =
Label(signup_frame,width="15",padx="5",pady="5",text="User
Gender",bg="yellow2",fg="deep sky
blue",font=("Calibri",15,'bold')).grid(row=3,column=0,padx="5",pady="5")
gander_male =
Radiobutton(signup_frame,text="Male",bg="yellow2",value="Male",width=
"5",variable=gender).place(x="190",y="155")

```

```

gander_female =
Radiobutton(signup_frame,text="Female",bg="yellow2",width="5",value="
Female",variable=gender).place(x="260",y="155")
gander_others =
Radiobutton(signup_frame,text="Others",bg="yellow2",width="5",value="
Others",variable=gender).place(x="328",y="155")
#widget 5
age_label = Label(signup_frame,width="15",padx="5",pady="5",text="User
Age",bg="yellow2",fg="deep sky
blue",font=("Calibri",15,'bold')).grid(row=4,column=0,padx="5",pady="5")
values = [int(i) for i in range(5,111)]
age.set("")
age_combo =
ttk.Combobox(signup_frame,value=values,width="25",state="readonly",text
variable=age,font=("Arial",10,"italic")).grid(row=4,column=1)

#widget 6

signup_btn = Button(signup_frame,text="Sign Up",width="30",bg="deep
sky
blue",fg="yellow2",font=("Arial",15,"italic"),command=validation).grid(ro
w=5,columnspan=2,pady="15",padx="10")

screen.mainloop()

```

Registration form:-

```
from tkinter import *
```

```
from tkinter import messagebox
```

```
#-----Functions/Actions
```

```
def validation():
```

```
    if email.get()!="" and password.get()!="":
```

```
        u_checker = userchecker()
```

```
        email_checker = email_existance_checker()
```

```
        email_format = emailformat_checker()
```

```
    if email_format:
```

```
        if u_checker:
```

```
            messagebox.showinfo("Information","Sucessfully Login")
```

```
        else:
```

```
            if email_checker:
```

```
                messagebox.showerror("Error","Ur Password Is incorrect")
```

```
            else:
```

```
                messagebox.showerror("Error",'Ur Email Doesnot Exist , Please Move  
To SignUp')
```

```
        else:
```

```
            messagebox.showerror("Error","Email Format Is Invalid")
```

```

else:
    messagebox.showerror("Error", "Fill All The Required Feilds")

def emailformat_checker():
    temp = email.get()

    count=-1

    for i in temp:
        count+=1

        if i=="@":
            if temp[count:len(temp)]=="@gmail.com":
                return True

            else:
                return False

    else:
        return False

def userchecker():
    with open("insertion.txt") as rd:
        data = rd.readline()

        while data!="":
            temp = data.split(",")

```



```
        if temp[1]==email.get() and temp[2]==password.get():
            return True
        data = rd.readline()
    else:
        return False

def email_existance_checker():
    with open("insertion.txt") as rd:
        data = rd.readline()
        while data!="":
            temp = data.split(',')
            if temp[1]==email.get():
                return True
            data = rd.readline()
    else:
        return False

def shifting_form():
    screen.destroy()
```

```
import signup

#-----GUI

screen = Tk()

#-----Variables

email=StringVar()

password = StringVar()

#-----Screen GUI

screen.geometry("700x600")

screen.maxsize(width="700",height="600")

screen.minsize(width="700",height="600")

screen.config(bg="yellow2")

screen.title("Registration")

#Login Title

title = Label(text="LOG
IN",font=("Arial",72,"bold"),pady="5",bg="yellow2",fg="deep sky
blue").pack(pady="50")
```

```
#login frame

login_frame = Frame(screen,width="380",height="300",bg="deep sky blue")

login_frame.place(x="162",y="150")

# login frame widgets

#1

email_label = Label(width="15",padx="5",pady="5",text="User Email",bg="deep
sky blue",fg="yellow2",font=("Calibri",20,'bold'))

email_label.place(x="165",y="170")

#2

email_entry = Entry(textvariable=email,width="26",selectbackground="deep sky
blue",selectforeground="black",font=("Calibri",15,"italic"))

email_entry.place(x="215",y="210")

#3

password_label = Label(width="15",padx="5",pady="5",text="User
Password",bg="deep sky blue",fg="yellow2",font=("Calibri",20,'bold'))

password_label.place(x="188",y="240")

#4

password_entry =
Entry(textvariable=password,show="*",width="26",selectbackground="deep sky
blue",selectforeground="black",font=("Calibri",15,"italic"))

password_entry.place(x="215",y="280")
```

#5

```
login_btn = Button(text="Log  
In",width="23",font=("Arial",14,"italic"),bg="yellow2",fg="deep sky  
blue",command=validation)  
  
login_btn.place(x="218",y="330")
```

#6

```
txt=Label(text="not a member?",font=("Clibri",12,"italic"),bg="deep sky  
blue",fg="white").place(x="230",y="390")
```

#7

```
sgnup_btn = Button(text="Sign up  
now",width="14",font=("Calibri",10,"bold"),bg="deep sky  
blue",fg="white",command=shifting_form).place(x="347",y="390")
```

```
screen.mainloop()
```

Number guessing code:-

```
# Guess a number game
```

```
#Add feature to check if the input number was between 1-100
```

```
#Add levels to the game to make it more exciting
```

```
from random import randint #To generate a random number
```

```
name = input("Please Enter your name: ")
```

```
print("Welcome to my Number game, " + name)
```

```
def game():
```

```

rand_number = randint(0,100) #Generates a random number
print("\nI have selected a number between 1 to 100...")
print("You have 6 chances to guess that number...")
i = 1
r = 1
while i<7: #6 Chances to the user
    user_number = int(input('Enter your number: '))
    if user_number < rand_number:
        print("\n" + name + ", My number is greater than your guessed number")
        print("you now have " + str(6-i)+ " chances left" )
        i = i+1
    elif user_number > rand_number:
        print("\n" + name + ", My number is less than your guessed number")
        print("you now have " + str(6-i)+ " chances left" )
        i = i+1
    elif user_number == rand_number:
        print("\nCongratulations "+name+"!! You have guessed the correct
number!")
        r = 0;
        break
    else:
        print("This is an invalid number. Please try again")
        print("you now have  chances left" )
        continue
if r==1:
    print("Sorry you lost the game!!")
    print("My number was = " + str(rand_number))

```

```
def main():
    game()
    while True:
        another_game = input("Do you wish to play again?(y/n): ")
        if another_game == "y":
            game()
        else:
            break
main()
print("\nEnd of the Game! Thank you for playing!")
```

kids learning :

- **Kids learning :-**

```
from tkinter import *
from tkinter import messagebox
import random
```

```
def main():
    root = Tk()

    n = random.randint(1, 4)

    if n == 1:
        frame1 = Frame(root)
```

```
img1 = PhotoImage(file="bear.png")
label1 = Label(frame1, image=img1)
label1.grid(row=0)
```

```
img2 = PhotoImage(file="book.png")
label2 = Label(frame1, image=img2)
label2.grid(row=0, column=1)
```

```
img3 = PhotoImage(file="car.png")
label3 = Label(frame1, image=img3)
label3.grid(row=1)
```

```
img4 = PhotoImage(file="chair.png")
label4 = Label(frame1, image=img4)
label4.grid(row=1, column=1)
frame1.pack(side=TOP)
```

```
def des1():
    root.destroy()
    main()
```

```
refresh_button = Button(frame1, text="REFRESH", command= des1)
refresh_button.grid(row=2)
```

```
def ok1():
    root.destroy()
    root1 = Tk()
```

```

root1.geometry("220x300")

var_a1 = IntVar()
var_a2 = IntVar()
var_a3 = IntVar()
var_a4 = IntVar()
var_a5 = IntVar()
var_a6 = IntVar()
var_a7 = IntVar()
var_a8 = IntVar()
var_a9 = IntVar()
var_a10 = IntVar()

var_a = [var_a1, var_a2, var_a3, var_a4, var_a5, var_a6, var_a7, var_a8,
var_a9, var_a10]
x_a = ["Seal", "Bear", "Google", "Book", "Kite", "Car", "Tree", "Chair",
"Lion", "Elephant"]
y_a = 0
buttons_a = [Checkbutton(root1, text=x_a[i], variable=var_a[i]) for i in
range(10)]
for b in buttons_a:
    b.grid(row=y_a, sticky=W)
    y_a += 1

def result_1():
    count=0
    if var_a[1].get() == 1:

```



```

        count+=1
    if var_a[3].get() == 1:
        count+=1
    if var_a[5].get() == 1:
        count+=1
    if var_a[7].get() == 1:
        count+=1

    messagebox.showinfo("RESULT", "You've answered %d correct
answers" %count)

submit_button_1 = Button(root1, text="SUBMIT", command=result_1)
submit_button_1.grid(row=10)

def back():
    root1.destroy()
    main()

back_button = Button(root1, text="BACK", command=back)
back_button.grid(row=10, column=1)

root1.title("OPTIONS")
root1.mainloop()

ok_button_1 = Button(frame1, text=" OK ", command=ok1)
ok_button_1.grid(row=2, column=1)

```

```
elif n == 2:
    frame2 = Frame(root)

    img1 = PhotoImage(file="cheetah.png")
    label1 = Label(frame2, image=img1)
    label1.grid(row=0)

    img2 = PhotoImage(file="earth.png")
    label2 = Label(frame2, image=img2)
    label2.grid(row=0, column=1)

    img3 = PhotoImage(file="einstein.png")
    label3 = Label(frame2, image=img3)
    label3.grid(row=1)

    img4 = PhotoImage(file="elephant.png")
    label4 = Label(frame2, image=img4)
    label4.grid(row=1, column=1)

    def des2():
        root.destroy()
        main()

    frame2.pack(side=BOTTOM)
```

```

refresh_button = Button(frame2, text="REFRESH", command=des2)
refresh_button.grid(row=2)

def ok2():
    root.destroy()
    root1 = Tk()
    root1.geometry("220x300")

    var_a1 = IntVar()
    var_a2 = IntVar()
    var_a3 = IntVar()
    var_a4 = IntVar()
    var_a5 = IntVar()
    var_a6 = IntVar()
    var_a7 = IntVar()
    var_a8 = IntVar()
    var_a9 = IntVar()
    var_a10 = IntVar()

    var_a = [var_a1, var_a2, var_a3, var_a4, var_a5, var_a6, var_a7, var_a8,
var_a9, var_a10]
    x_a = ["Einstein", "Bear", "Earth", "Book", "Kite", "Car", "Tree",
"Cheetah", "Lion", "Elephant"]
    y_a = 0
    buttons_a = [Checkbutton(root1, text=x_a[i], variable=var_a[i]) for i in
range(10)]
    for b in buttons_a:

```

```

b.grid(row=y_a, sticky=W)
y_a += 1

def result_2():
    count=0
    if var_a[0].get() == 1:
        count+=1
    if var_a[2].get() == 1:
        count+=1
    if var_a[7].get() == 1:
        count+=1
    if var_a[9].get() == 1:
        count+=1

    messagebox.showinfo("RESULT", "You've answered %d correct
answers" %count)

submit_button_2 = Button(root1, text="SUBMIT", command=result_2)
submit_button_2.grid(row=10)

def back():
    root1.destroy()
    main()

back_button = Button(root1, text="BACK", command=back)
back_button.grid(row=10, column=1)

```

```
    root1.title("OPTIONS")

    root1.mainloop()

    ok_button_2 = Button(frame2, text=" OK ", command=ok2)
    ok_button_2.grid(row=2, column=1)

elif n == 3:
    frame3 = Frame(root)

    img1 = PhotoImage(file="google.png")
    label1 = Label(frame3, image=img1)
    label1.grid(row=0)

    img2 = PhotoImage(file="guitar.png")
    label2 = Label(frame3, image=img2)
    label2.grid(row=0, column=1)

    img3 = PhotoImage(file="laptop.png")
    label3 = Label(frame3, image=img3)
    label3.grid(row=1)

    img4 = PhotoImage(file="lion.png")
    label4 = Label(frame3, image=img4)
    label4.grid(row=1, column=1)

    frame3.pack()
```

```
def des3():
```

```
    root.destroy()
```

```
    main()
```

```
refresh_button = Button(frame3, text="REFRESH", command=des3)
```

```
refresh_button.grid(row=2)
```

```
def ok3():
```

```
    root.destroy()
```

```
    root1 = Tk()
```

```
    root1.geometry("220x300")
```

```
    var_a1 = IntVar()
```

```
    var_a2 = IntVar()
```

```
    var_a3 = IntVar()
```

```
    var_a4 = IntVar()
```

```
    var_a5 = IntVar()
```

```
    var_a6 = IntVar()
```

```
    var_a7 = IntVar()
```

```
    var_a8 = IntVar()
```

```
    var_a9 = IntVar()
```

```
    var_a10 = IntVar()
```

```
    var_a = [var_a1, var_a2, var_a3, var_a4, var_a5, var_a6, var_a7, var_a8,  
var_a9, var_a10]
```

```

x_a = ["Seal", "Bear", "Laptop", "Book", "Guitar", "Car", "Tree", "Chair",
"Lion", "Google"]
y_a = 0
buttons_a = [Checkbutton(root1, text=x_a[i], variable=var_a[i]) for i in
range(10)]
for b in buttons_a:
    b.grid(row=y_a, sticky=W)
    y_a += 1

def result_3():
    count=0
    if var_a[2].get() == 1:
        count+=1
    if var_a[4].get() == 1:
        count+=1
    if var_a[8].get() == 1:
        count+=1
    if var_a[9].get() == 1:
        count+=1

    messagebox.showinfo("RESULT", "You've answered %d correct
answers" %count)

submit_button_3 = Button(root1, text="SUBMIT", command=result_3)
submit_button_3.grid(row=10)

def back():

```

```
    root1.destroy()
    main()

    back_button = Button(root1, text="BACK", command=back)
    back_button.grid(row=10, column=1)

    root1.title("OPTIONS")

    root1.mainloop()

    ok_button_3 = Button(frame3, text=" OK ", command=ok3)
    ok_button_3.grid(row=2, column=1)

elif n == 4:
    frame4 = Frame(root)

    img1 = PhotoImage(file="mobile.png")
    label1 = Label(frame4, image=img1)
    label1.grid(row=0)

    img2 = PhotoImage(file="plane.png")
    label2 = Label(frame4, image=img2)
    label2.grid(row=0, column=1)

    img3 = PhotoImage(file="table.png")
    label3 = Label(frame4, image=img3)
    label3.grid(row=1)
```



```
img4 = PhotoImage(file="tree.png")
label4 = Label(frame4, image=img4)
label4.grid(row=1, column=1)

frame4.pack()

def des4():
    root.destroy()
    main()

refresh_button = Button(frame4, text="REFRESH", command=des4)
refresh_button.grid(row=2)

def ok4():
    root.destroy()
    root1 = Tk()
    root1.geometry("220x300")

var_a1 = IntVar()
var_a2 = IntVar()
var_a3 = IntVar()
var_a4 = IntVar()
var_a5 = IntVar()
var_a6 = IntVar()
var_a7 = IntVar()
var_a8 = IntVar()
```

```

var_a9 = IntVar()
var_a10 = IntVar()

var_a = [var_a1, var_a2, var_a3, var_a4, var_a5, var_a6, var_a7, var_a8,
var_a9, var_a10]
x_a = ["Plane", "Bear", "Google", "Book", "Table", "Car", "Tree",
"Mobile", "Chair", "Elephant"]
y_a = 0
buttons_a = [Checkbutton(root1, text=x_a[i], variable=var_a[i]) for i in
range(10)]
for b in buttons_a:
    b.grid(row=y_a, sticky=W)
    y_a += 1

def result_4():
    count=0
    if var_a[0].get() == 1:
        count+=1
    if var_a[4].get() == 1:
        count+=1
    if var_a[6].get() == 1:
        count+=1
    if var_a[7].get() == 1:
        count+=1

    messagebox.showinfo("RESULT", "You've answered %d correct
answers" %count)

```

```
submit_button_4 = Button(root1, text="SUBMIT", command=result_4)
submit_button_4.grid(row=10)
```

```
def back():
    root1.destroy()
    main()
```

```
back_button = Button(root1, text="BACK", command=back)
back_button.grid(row=10, column=1)
```

```
root1.title("OPTIONS")
```

```
root1.mainloop()
```

```
ok_button_4 = Button(frame4, text=" OK ", command=ok4)
ok_button_4.grid(row=2, column=1)
```

```
root.title("Kids Learning Game")
root.mainloop()
```

```
main()
```

- **Maths game:-**

```
import sqlite3
```

```

import random
db = sqlite3.connect('database.db')
c=db.cursor()
c.execute("CREATE TABLE IF NOT EXISTS data(name TEXT , highscore
REAL)")
db.commit()

def welcome(name,highscore): # Display the welcoming message
    print("          Hello {} ! \n\n          Welcome to Maths game \n\n
Your high score :
{} \n\n_____ \n
\n".format(name,highscore))
    input("Press Enter to start")

def main():

print("_____ \n
\n          [MATH
GAME] \n\n_____
___ \n")
    data=c.execute("SELECT * FROM data")

    # Count the number of the table's rows
    x=0
    for i in data:
        x=x+1

```

```

if x==0: # If empty (first run)
    # Ask about the user name
    ask=True
    while ask:
        try:
            name = input("[+] Please Enter your name : ")
            if name==" " or len(name)==0:
                print("Please Enter a valid value !")
            else:
                ask=False
        except ValueError:
            print("Please Enter a valid value")
            askName()

print("\n_____ \
n")
    c.execute("INSERT INTO data VALUES(?,?)",(name,"0")) # Set the name in
database
    db.commit()
    welcome(name,"0")
    highscore=0

else: # User played before
    data=c.execute("SELECT * FROM data")
    for i in data: # Get the name and highscore
        name=i[0]

```

```

    highscore=i[1]
    welcome(name,highscore)

score=0
lose=False
while lose!=True:
    calc=random.randint(0,1) # Generate a random calculation type 0=[+] 1=[*]

    if calc==0: #(+)
        fnum=random.randint(2,50) # First number
        snum=random.randint(2,50) # Second number
        result=fnum+snum
        inp="[+] {} + {} = " # User input text

    else: #(*)
        fnum=random.randint(1,10)
        snum=random.randint(1,10)
        result=fnum * snum
        inp="[+] {} * {} ="

def askUser(): # Ask user what is the result
    userInput=input(inp.format(fnum,snum))
    try:
        userInput=int(userInput) # Try to make input as int
        return userInput
    except: # If user input a text or leaft it empty
        print("Enter a valid value")

```

```

        askUser() # Reask again
    userInput=askUser()

    if userInput==result: # Correct answer
        score=score+1
        if score>highscore: # If this score is the high score
            c.execute("UPDATE data SET highscore = ? WHERE name =
?",(score,name)) # Change highscore in database
            db.commit()
            print("Correct ! , High score ! your score : {}".format(score)) # Show
message
        else: # The score is not the highscore
            print("Correct ! your score : {}".format(score))

    else: # Wrong answer
        score=0
        print("Oops ! Wrong Answer!")
        userAgain=input("[+] Play again? (y,n) : ").lower()
        if userAgain=="n":
            print("Good bye !")
            db.close()
            lose=True

main()

```

- **Color game:-**

```
from tkinter import *
import tkinter.font as font
import random

colors=["Red", "Orange", "White", "Black", "Green", "Blue", "Brown", "Purple",
"Cyan", "Yellow", "Pink", "Magenta"]
timer=60
score=0
displayed_word_color=""

# This function will be called when start button is clicked
def startGame():
    global displayed_word_color

    if timer==60:
        startCountDown()
        displayed_word_color=random.choice(colors).lower()
        display_words.config(text=random.choice(colors), fg=displayed_word_color)
        color_entry.bind('<Return>', displayNextWord)

# This function is to reset the game
def resetGame():
    global timer, score, displayed_word_color
    timer=60
```



```
score=0
displayed_word_color=""
game_score.config(text="Your Score : "+str(score))
display_words.config(text="")
time_left.config(text="Game Ends in : -")
color_entry.delete(0, END)
```

This function will start count down

```
def startCountDown():
```

```
    global timer
```

```
    if timer>=0:
```

```
        time_left.config(text="Game Ends in : "+str(timer)+"s")
```

```
        timer-=1
```

```
        time_left.after(1000,startCountDown)
```

```
    if timer==-1:
```

```
        time_left.config(text="Game Over!!!")
```

This function to display random words

```
def displayNextWord(event):
```

```
    global displayed_word_color
```

```
    global score
```

```
    if timer>0:
```

```
        if displayed_word_color==color_entry.get().lower():
```

```
            score+=1
```

```
            game_score.config(text="Your Score : "+str(score))
```

```
            color_entry.delete(0, END)
```

```
displayed_word_color=random.choice(colors).lower()
display_words.config(text=random.choice(colors), fg=displayed_word_color)
```

```
root=Tk()
root.title("Color Game")
root.geometry("500x200")
root.iconbitmap("game-console.ico")
```

```
app_font=font.Font(family='Helvetica', size=12)
```

```
game_desp="Game Description: Enter the color of the words displayed below. \n
And Keep in mind not to enter the " \
"word text itself "
```

```
myFont=font.Font(family='Helvetica')
```

```
game_description=Label(root, text=game_desp,font=app_font, fg="grey")
game_description.pack()
```

```
game_score=Label(root, text="Your Score : "+str(score),
font=(font.Font(size=16)), fg="green")
game_score.pack()
```

```
display_words=Label(root,font=(font.Font(size=28)), pady=10)
display_words.pack()
```

```
time_left=Label(root,text="Game Ends in : -", font=(font.Font(size=14)),
fg="orange")
time_left.pack()

color_entry=Entry(root, width=30)
color_entry.pack(pady=10)

btn_frame=Frame(root, width=80, height=40, bg='red')
btn_frame.pack(side=BOTTOM)

start_button=Button(btn_frame, text="Start", width=20, fg="black", bg="pink",
bd=0, padx=20, pady=10,command=startGame)
start_button.grid(row=0, column=0)

reset_button=Button(btn_frame, text="Reset",width=20, fg="black",bg="light
blue", bd=0, padx=20, pady=10, command=resetGame)
reset_button.grid(row=0, column=1)

root.geometry('600x300')
root.mainloop()
```

- **Rhyming learning :-**

```
"""Make rhyming words"""
```

```
import argparse
```

```
import re
```

```
import string
```

```
# -----
```

```
def get_args():
```

```
    """get command-line arguments"""
```

```
    parser = argparse.ArgumentParser(
```

```
        description='Make rhyming "words"',
```

```
        formatter_class=argparse.ArgumentDefaultsHelpFormatter)
```

```
    parser.add_argument('word', metavar='word', help='A word to rhyme')
```

```
    return parser.parse_args()
```

```
# -----
```

```
def main():
```

```
    """Make a jazz noise here"""
```

```
    args = get_args()
```

```

prefixes = list('bcdfghjklmnpqrstvwxyz') + (
    'bl br ch cl cr dr fl fr gl gr pl pr sc '
    'sh sk sl sm sn sp st sw th tr tw thw wh wr '
    'sch scr shr sph spl spr squ str thr').split()

start, rest = stemmer(args.word)
if rest:

    print('\n'.join(sorted([p + rest for p in prefixes if p != start])))
else:
    print(f'Cannot rhyme "{args.word}")

```

```

def stemmer(word):
    """Return leading consonants (if any), and 'stem' of word"""

    word = word.lower()
    vowels = 'aeiou'
    consonants = ".join(
        [c for c in string.ascii_lowercase if c not in vowels])
    pattern = (
        '([ + consonants + ]+)?' # capture one or more, optional
        '([ + vowels + ])' # capture at least one vowel
        '(.*)' # capture zero or more of anything
    )

```

```
match = re.match(pattern, word)
```

```
if match:
```

```
    p1 = match.group(1) or "
```

```
    p2 = match.group(2) or "
```

```
    p3 = match.group(3) or "
```

```
    return (p1, p2 + p3)
```

```
else:
```

```
    return (word, "")
```

```
# -----
```

```
def test_stemmer():
```

```
    """test the stemmer"""
```

```
    assert stemmer("") == ("", "")
```

```
    assert stemmer('cake') == ('c', 'ake')
```

```
    assert stemmer('chair') == ('ch', 'air')
```

```
    assert stemmer('APPLE') == ("", 'apple')
```

```
    assert stemmer('RDNZL') == ('rdnzl', "")
```

```
    assert stemmer('123') == ("", "")
```

```
# -----
```

```
if __name__ == '__main__':
```

```
    main()
```

Sudoku :-

- **Turtle:**

```
import turtle
import random
myPen = turtle.Turtle()
myPen.shape("turtle")
myPen.speed(20)

window = turtle.Screen()
window.bgcolor("#00B2C0")

def corner(turtle):
    for x in range(0, 6):
        turtle.left(15)
        turtle.forward(2)

def cornered_box(turtle, x1, y1, x2, y2, letter):
    turtle.penup()
    turtle.goto(x1, y1)
    turtle.pendown()
    turtle.color("black")
    turtle.fillcolor("black")
    turtle.begin_fill()
    for x in range(0, 4):
        turtle.forward(40)
        corner(turtle)
    turtle.end_fill()
```

```
turtle.penup()
turtle.goto(x4, y24)
turtle.color("white")
turtle.write(letter, None, None, "28pt bold")
```

```
cornered_box(myPen, -40, 20, -35, 35, "C")
cornered_box(myPen, 25, 20, 30, 35, "O")
cornered_box(myPen, -40, -45, -35, -30, "D")
cornered_box(myPen, 25, -45, 30, -30, "E")
```

```
myPen.color("white")
myPen.goto(-115, -92)
myPen.write("Anyone Can Learn", None, None, "24pt bold")
myPen.goto(-115, -127)
myPen.write("Anyone Can Teach", None, None, "24pt bold")
myPen.left(90)
```

```
myPen.goto(20, 105)
```

```
for x in range(0, 100):
    myPen.color("#%06x" % random.randint(0, 2**24 - 1))
myPen.color("white")
```


- **Solver:-**

```
import numpy as np
board = [
    [5,3,0,0,7,0,0,0,0],
    [6,0,0,1,9,5,0,0,0],
    [0,9,8,0,0,0,0,6,0],
    [8,0,0,0,6,0,0,0,3],
    [4,0,0,8,0,3,0,0,1],
    [7,0,0,0,2,0,0,0,6],
    [0,6,0,0,0,0,2,8,0],
    [0,0,0,4,1,9,0,0,5],
    [0,0,0,0,8,0,0,7,9]
]

def solve(board):
    # Find next empty cell
    findEmpty = emptyCell(board)

    if not findEmpty:
        return True # Board is filled
    else:
        row, column = findEmpty

    for i in range(1,10):
        if isValid(board, i, (row, column)):
            board[row][column] = i
```

```
    if solve(board):
        return True

    board[row][column] = 0

return False
```

```
def isValid (board, num, pos):
    # Check Row
    for i in range(9):
        if num == board[pos[0]][i]:
            return False
    # Check Column
    for i in range(9):
        if num == board[i][pos[1]]:
            return False

    # Check Sub Grid
    row = pos[0] // 3
    column = pos[1] // 3

    for i in range(row * 3, (row * 3) + 3):
        for j in range(column * 3, (column * 3) + 3):
            if num == board[i][j]:
                return False
    return True
```

```
def emptyCell(board):
    for row in range(9):
        for column in range(9):
            if board[row][column] == 0:
                return (row,column)
    return None
```

```
def printBoard(board):
    print(np.matrix(board))
```

- **GUI:-**

'''

GUI which allows you to solve any Sudoku puzzle.

The GUI has been created using Tkinter and a backtracking algorithm has been used.

'''

```
from tkinter import *
```

```
root = Tk()
```

```
root.geometry('330x370')
```

```
# Sudoku solver class
```

```
class SudokuSolver():
```

```

def __init__(self):
    self.setZero()
    self.solve()

# Set the empty cells to 0 (i.e. the cells you have not filled in)
def setZero(self):
    for i in range(9):
        for j in range(9):
            if filledBoard[i][j].get() not in ['1','2','3','4','5','6','7','8','9']:
                filledBoard[i][j].set(0)

# Solve using backtracking
def solve(self):
    # Find next empty cell
    findEmpty = self.emptyCell()

    if not findEmpty:
        return True
    else:
        row, column = findEmpty

    for i in range(1,10):
        if self.isValid(i, (row, column)):
            filledBoard[row][column].set(i)

            if self.solve():
                return True

```

```

        filledBoard[row][column].set(0)

    return False

    # Check row, column and subgrid(3x3 square) to see if number can be placed in
    cell
    def isValid (self, num, pos):
        # Check Row
        for i in range(9):
            if filledBoard[pos[0]][i].get() == str(num):
                return False
        # Check Column
        for i in range(9):
            if filledBoard[i][pos[1]].get() == str(num):
                return False

        # Check Sub Grid
        row = pos[0] // 3
        column = pos[1] // 3

        for i in range(row * 3, (row * 3) + 3):
            for j in range(column * 3, (column * 3) + 3):
                if filledBoard[i][j].get() == str(num):
                    return False
    return True

```

```
# Find empty cells, defined as cells filled with a zero
```

```
def emptyCell(self):
```

```
    for row in range(9):
```

```
        for column in range(9):
```

```
            if filledBoard[row][column].get() == '0':
```

```
                return (row,column)
```

```
    return None
```

```
# GUI class
```

```
class Interface():
```

```
    def __init__(self, window):
```

```
        self.window = window
```

```
        window.title("Sudoku Solver")
```

```
        font = ('Arial', 20)
```

```
        color = 'white'
```

```
# Create solve and clear button and link them to Solve and Clear methods
```

```
solve = Button(window, text = 'Solve', command = self.Solve)
```

```
solve.grid(column=3,row=20)
```

```
clear = Button(window, text = 'Clear', command = self.Clear)
```

```
clear.grid(column = 5,row=20)
```

```
# Initialise empty 2D list
```

```
self.board = []
```

```
for row in range(9):
```

```
    self.board += [["", "", "", "", "", "", "", "", ""]]
```

```

for row in range(9):
    for col in range(9):
        # Change color of cells depending on position in grid
        if (row < 3 or row > 5) and (col < 3 or col > 5):
            color = 'white'
        elif (row >= 3 and row < 6) and (col >=3 and col < 6):
            color = 'white'
        else:
            color = 'grey'

        # Make each cell of grid a entry box and store each user entry into the
        filledBoard 2D list

        self.board[row][col] = Entry(window, width = 2, font = font, bg = color,
            cursor = 'arrow', borderwidth = 2,
                highlightcolor = 'yellow', highlightthickness = 0,
            highlightbackground = 'black',
                textvariable = filledBoard[row][col])

        self.board[row][col].bind('<FocusIn>', self.gridChecker) # Link to better
        understand binding statements: https://www.python-
        course.eu/tkinter\_events\_binds.php#:~:text=Tkinter%20provides%20a%20mechanism%20to,and%20methods%20to%20an%20event.&text=If%20the%20defined%20event%20occurs,describing%20the%20event.

        self.board[row][col].bind('<Motion>', self.gridChecker)
        self.board[row][col].grid(row = row, column = col)

```

```
# Function to check if user enters a value which is not an int between 1 and 9
(valid numbers in Sudoku game).
```

```
# If entry is not valid, clear value
```

```
def gridChecker(self, event):
```

```
    for row in range(9):
```

```
        for col in range(9):
```

```
            if filledBoard[row][col].get() not in ['1','2','3','4','5','6','7','8','9']:
```

```
                filledBoard[row][col].set("")
```

```
# Call Sudoku solver class (called by solve button)
```

```
def Solve(self):
```

```
    SudokuSolver()
```

```
# Function to clear board (called by clear button)
```

```
def Clear(self):
```

```
    for row in range(9):
```

```
        for col in range(9):
```

```
            filledBoard[row][col].set("")
```

```
# Global 2D list which saved in the values the user enters on the GUI
```

```
# Each value in the 2D list is set as a StringVar(), a class in Tkinter
```

```
# which allows you to save values users enter into the Entry widget
```

```
filledBoard = []
```

```
for row in range(9):
```

```
    filledBoard += [["", "", "", "", "", "", "", "", ""]]
```

```
for row in range(9):
```

```
    for col in range(9):
```



```
filledBoard[row][col] = StringVar(root)
```

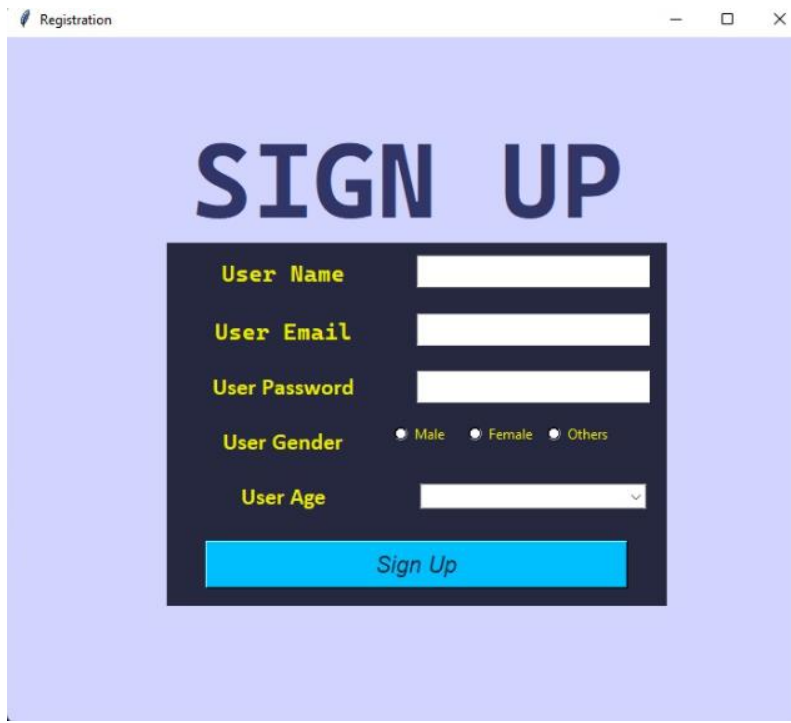
```
# Main Loop
```

```
Interface(root)
```

```
root.mainloop()
```

INPUT SCREEN:-

- Signup page:



The screenshot shows a web browser window titled "Registration". The main heading is "SIGN UP" in large, bold, dark blue letters. Below the heading is a dark blue form box with the following fields and options:

- User Name:** A white text input field.
- User Email:** A white text input field.
- User Password:** A white text input field.
- User Gender:** Three radio button options: "Male", "Female", and "Others".
- User Age:** A white dropdown menu.

At the bottom of the form box is a bright blue button labeled "Sign Up".

- Registration/login page:

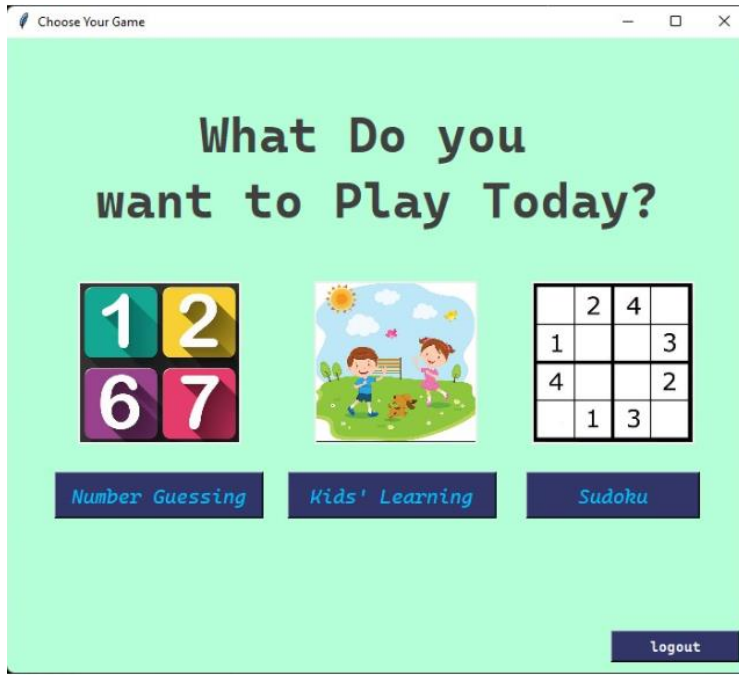


The screenshot shows a web browser window titled "Registration". The main heading is "LOG IN" in large, bold, dark blue letters. Below the heading is a blue form box with the following fields and options:

- User Email:** A white text input field.
- User Password:** A white text input field.

At the bottom of the form box is a yellow button labeled "Log In". Below the form box, there is a link "not a member?" followed by a dark blue button labeled "Sign up now".

- **Home page:**

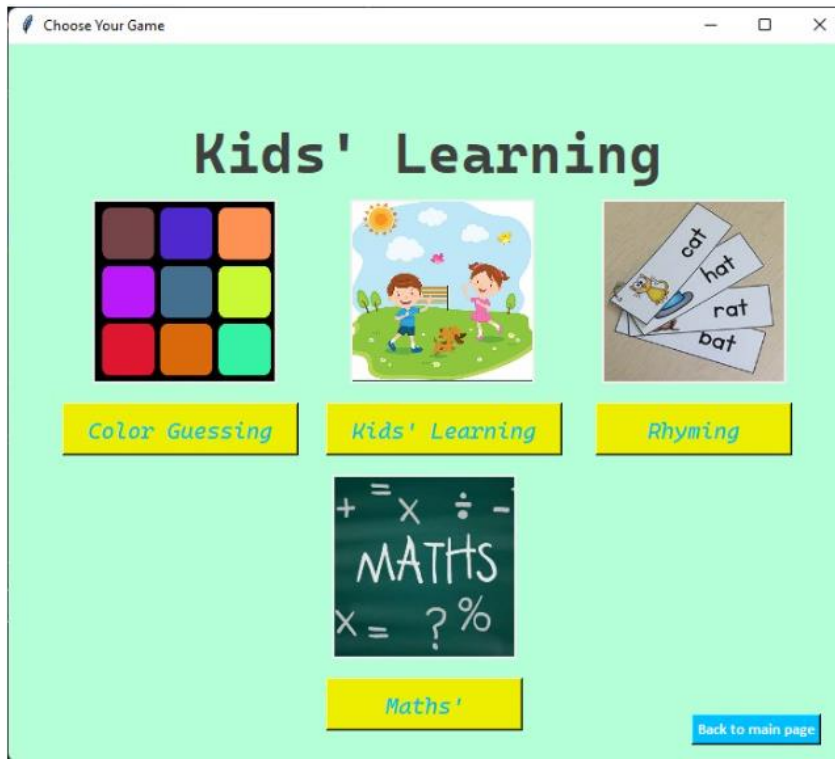


- **Number guessing:**

```
Please Enter your name: joice awsare
Welcome to my Number game, joice awsare

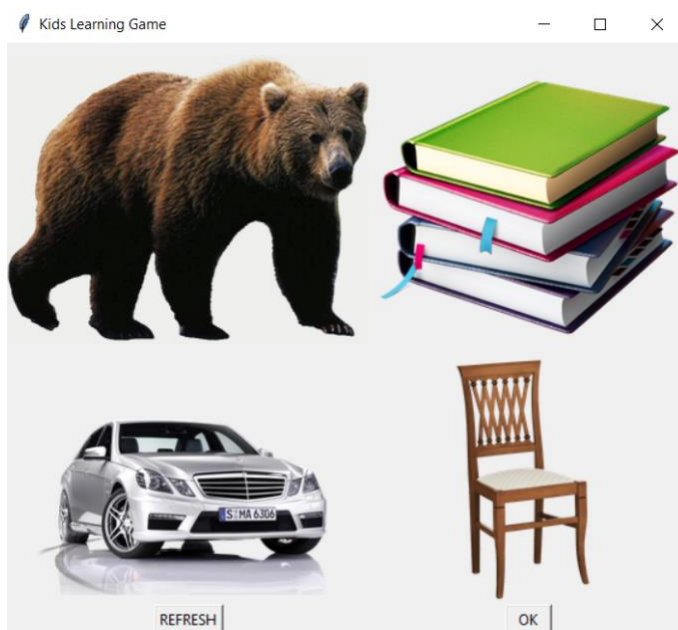
I have selected a number between 1 to 100...
You have 6 chances to guess that number...
Enter your number:
```

- **Kids learning home page:**



- **Kids learning app:**

1. **Kids learning:**



2. maths game:

```
[MATH GAME]

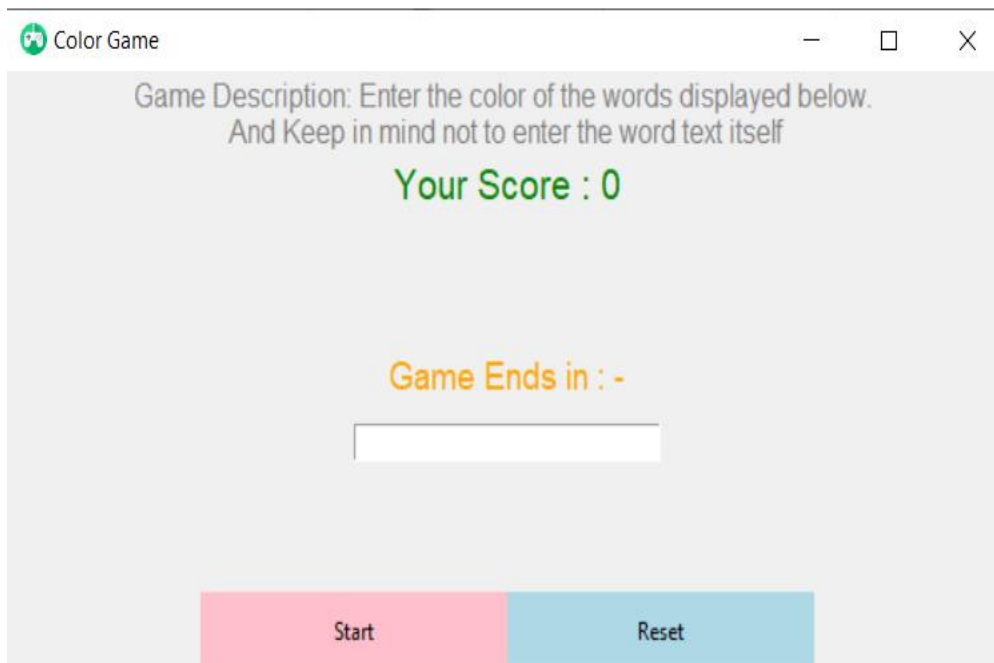
Hello amruta !

Welcome to Maths game

Your high score : 4.0

Press Enter to start
```

3. Colour guessing:



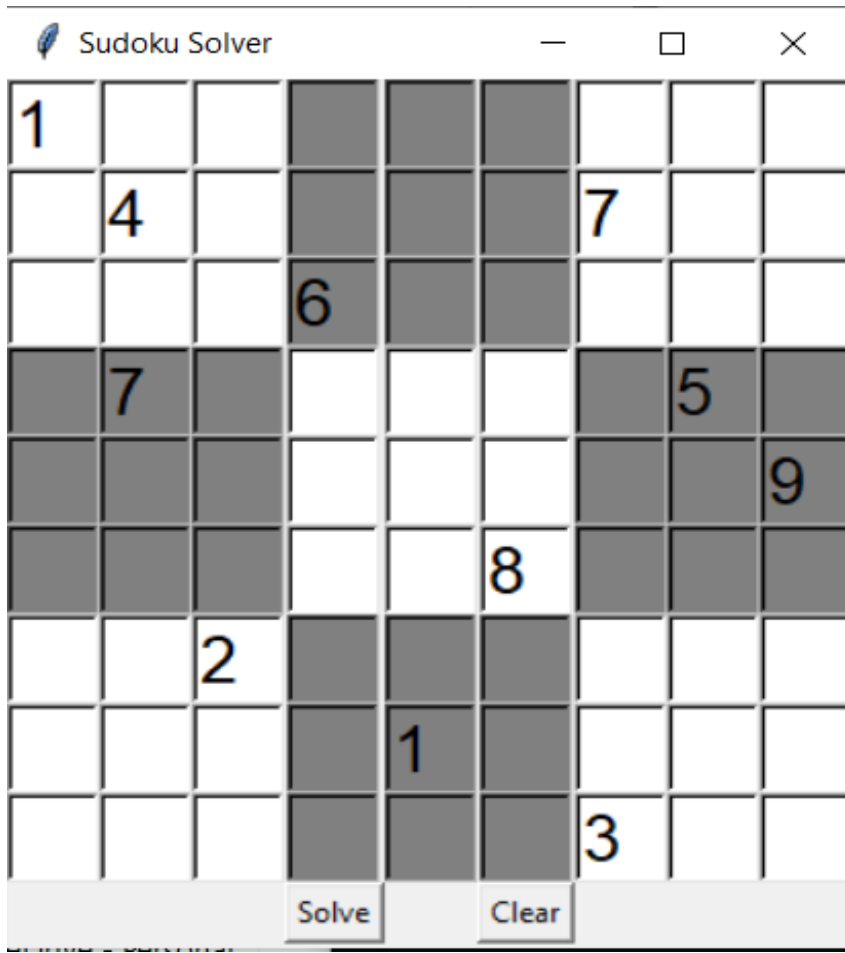
4. Rhyming game:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

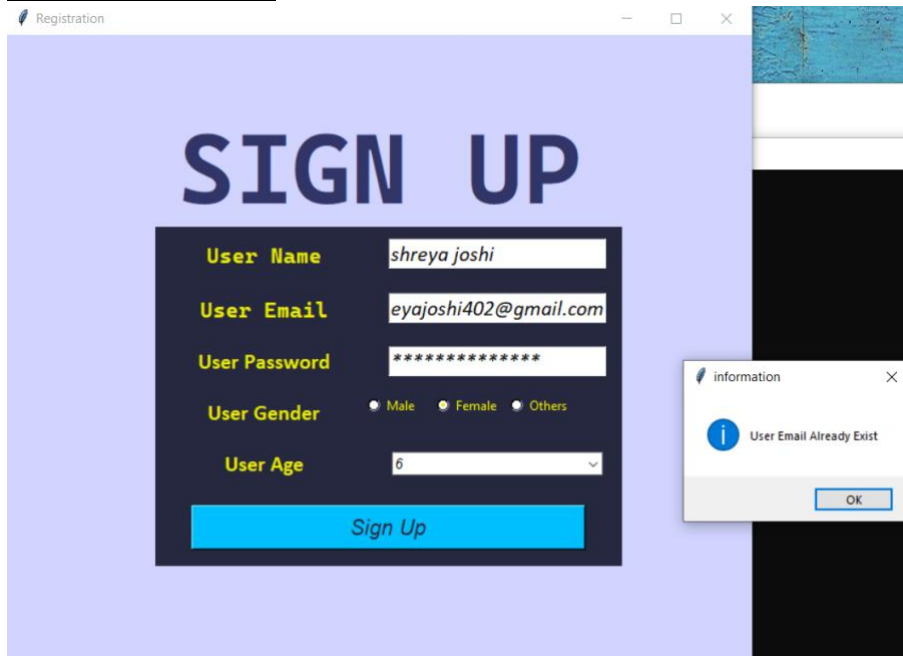
PS C:\Users\ravle\OneDrive\Desktop\shreya_joshi\project> & "C:/Program Files/Python310/python.exe" "c:/Users/ravle/OneDrive/Desktop/shreya_joshi/project/rhyming.py"
usage: rhyming.py [-h] word
rhyming.py: error: the following arguments are required: word
PS C:\Users\ravle\OneDrive\Desktop\shreya_joshi\project> █
```

- Sudoku:

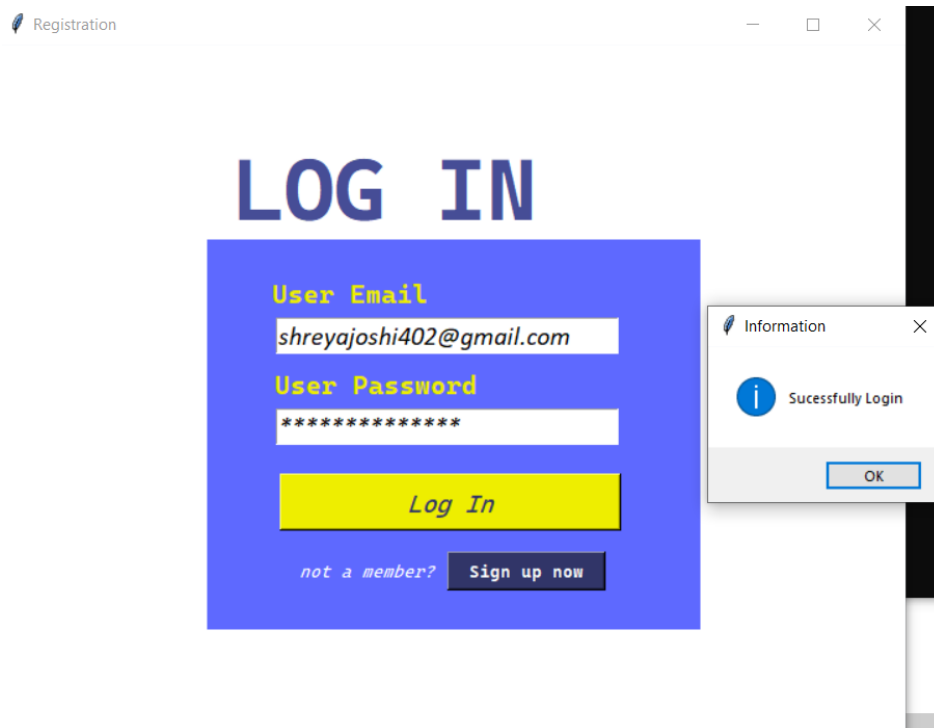


Output screen:-

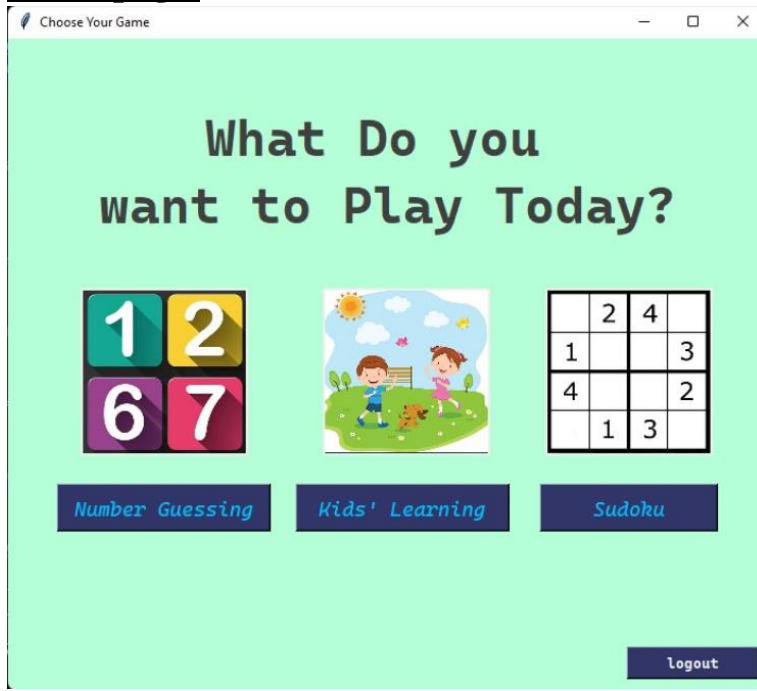
- Signup page:



- Registration/login page:



- **Home page:**



- **Number guessing:**

```
Please Enter your name: name_surname
Welcome to my Number game, name_surname

I have selected a number between 1 to 100...
You have 6 chances to guess that number...
Enter your number: 50

name_surname, My number is greater than your guessed number
you now have 5 chances left
Enter your number: 75

name_surname, My number is less than your guessed number
you now have 4 chances left
Enter your number: 63

name_surname, My number is less than your guessed number
you now have 3 chances left
Enter your number: 60

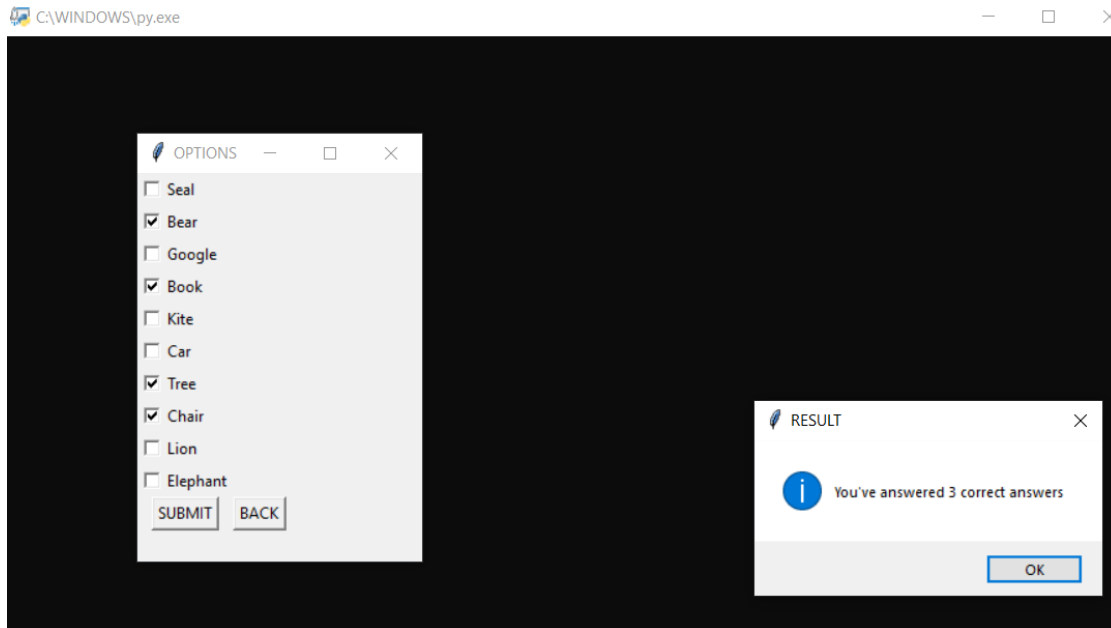
name_surname, My number is greater than your guessed number
you now have 2 chances left
Enter your number: 62

name_surname, My number is less than your guessed number
you now have 1 chances left
Enter your number: 61

Congratulations name_surname!! You have guessed the correct number!
Do you wish to play again?(y/n): |
```

- **Kids learning app:**

1.Memory power :



2.Maths game:

```
Hello amruta !

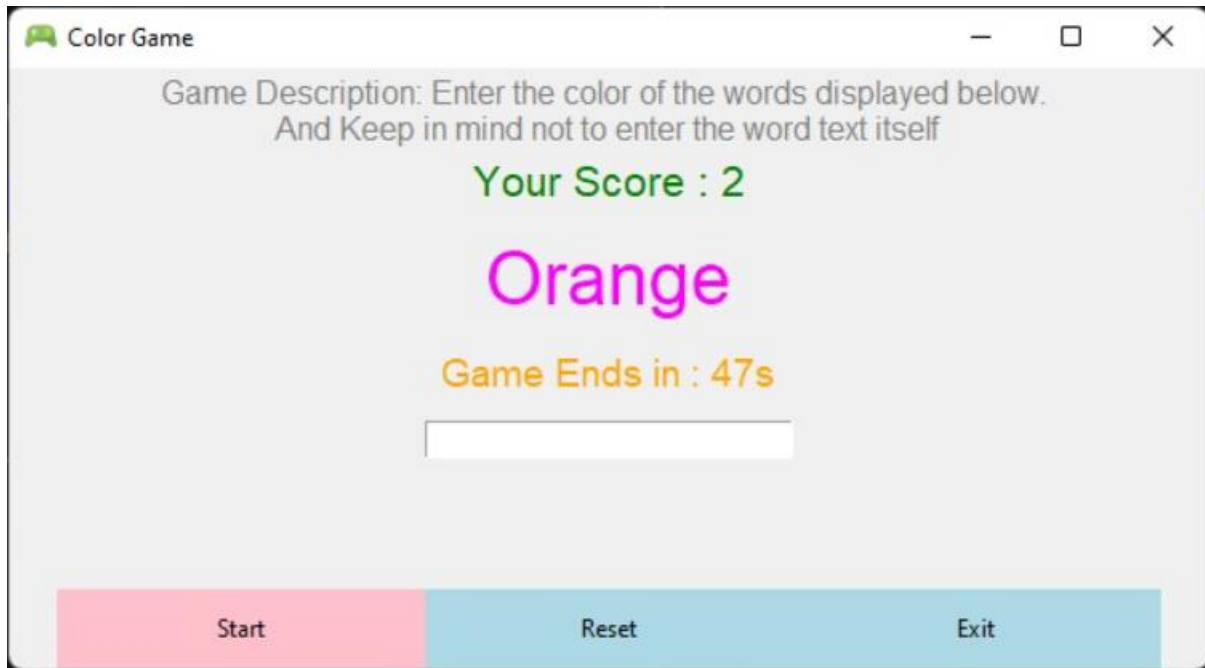
Welcome to Maths game

Your high score : 4.0

-----

Press Enter to start
[+] 2 + 17 = 19
Correct ! your score : 1
[+] 43 + 37 = 80
Correct ! your score : 2
[+] 43 + 30 = 73
Correct ! your score : 3
[+] 26 + 34 = 60
Correct ! your score : 4
[+] 25 + 40 = 65
Correct ! , High score ! your score : 5
[+] 46 + 36 = 82
Correct ! , High score ! your score : 6
[+] 8 * 8 =64
Correct ! , High score ! your score : 7
[+] 5 * 2 =10
Correct ! , High score ! your score : 8
[+] 8 + 47 = 56
Oops ! Wrong Answer!
[+] Play again? (y,n) :
```

3.Colour guessing :



4.Rhyming Word:

```
C:\Users\vaibh\PycharmProjects\lwf\venv\Scripts\python.exe C:/Users/vaibh/PycharmProjects/lwf/main.py
blord
bord
brord
chord
clord
cord
crod
dord
drord
flord
ford
frord
glord
gord
grord
hord
jord
kord
lord
mord
nord
plord
pord
```

- Sudoku:-

The image shows a window titled "Sudoku Solver" with a 9x9 grid. The grid contains the following numbers:

1	2	3	4	5	6	7	8	9
4	5	6A	7	8	9	1	2	3
7	8	9	1	2	3	4	5	6
2	1	4	3	6	5	8	9	7
3	6	5	8	9	7	2	1	4
8	9	7	2	1	4	3	6	5
5	3	1	6	4	2	9	7	8
6	4	2	9	7	8	5	3	1
9	7	8	5	3	1	6	4	2

At the bottom of the window, there are three buttons: "Solve", "Clear", and "quit".

Testing & Validation Checks

Testing

Testing plays important role to identify the quality of any software. Testing actually refers to detecting errors in the system. Before testing can begin, a test plan needs to be developed. Test plan actually includes the type of testing that has to be performed on the code, resources for testing, how the software will be tested. There are several types of testing during the test phase, that includes quality assurance testing (QAT), System Integration testing (SIT), and user acceptance testing (UAT).

- Quality Assurance (QA) Testing: In this the procedures and processes are checked. This means whether the instructions are executed as per the user requirements and commands.
- System Integration Testing (SIT): It verifies proper execution of software components and proper interfacing between components within the solution. the objective behind this testing is to validate that all software module dependencies are functionally correct and that data integrity is maintained between separate modules for the entire solution.
- User Acceptance Testing (UAT): This is the last phase of the software testing procedure. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. UAT is one of the final and critical software project procedures that must occur before newly developed software is rolled out to actual use.

Before testing the system, we need to consider following questions in our mind:

- What is the actual problem?
- How critical the problem is?

- Measures should be taken for the upcoming problems or errors?

Testing gives chance to upgrade or to improve if any drawbacks prevails in the application. Testing is generally done at two levels, testing of individual modules and testing entire system.

During system testing, the system is used experimentally to ensure that the software does not fall. that it will run according to its specification and in the way users expect. Testing is done throughout system development at various stages.

Following are the type of testing done in the project:

1. Program Testing: In this, we have to concentrate on the software part, system software should be free from errors. whether it is syntax error or logical error. In this system, we have done software testing and the output of this test is satisfactory. It fulfills all the conditions, which was required for the program testing.
2. Stress Testing: this test is conducted to check the performance of the system in main hours. It finds out how much workload the system can bear. In stress testing of this system, we come to know that this software can work easily and accurately at any condition. The concentration is made on the performance of the system by checking the giving input and their expected outputs.
3. Documentation Testing: this testing work to find out that whatever document supplied is satisfactory or there is a need to supply further document. In this system, all the documents which are supplied are satisfactory.

VALIDATION CHECKS

Data validation is the process of ensuring, at least as far as is possible, that the data given to a program by a user or from a file (essentially, the system's input) is of the correct type, and in the correct format.

There are however measures that can be taken to restrict the program's input to valid data. such measures involve the application of validation rules to any data being input to the program. In this system, Data validation rules can also make this system more user friendly, since they enable the program to warn the user immediately when there is a problem rather than simply allowing them to continue entering data until the program crashes or some other problem occurs.

In this proposed system, we have introduced the following data validation rules:

1. Value entered check: this is used for things like required fields in online forms where the user must enter some data (for example their username and password) and must not leave the field blank.
2. Permitted character check: it is useful for determining whether an input string contains valid characters. For example, a phone number may include the digits 09.
3. Limit check: It is used for numeric values that must either be greater than or equal to some lower limit, or less than or equal to some upper limit. For example, the limited number that a user can enter as a phone number is 10.

4. Confirmation check: At the time of creating an account in this system, it is used for determining whether the enter password and confirm password are same or not.
5. Email address check: At the time of creating an account, the system can only accept a valid email id. For example, “@gmail.com”.

System Security Measures

- **Physical:**

The sites containing computer systems must be physically secured against armed and malicious intruders. The workstations must be carefully protected.

- **Human:**

Only appropriate users must have the authorization to access the system. Phishing(collecting confidential information) and Dumpster Diving(collecting basic information so as to gain unauthorized access) must be avoided.

- **Operating system:**

The system must protect itself from accidental or purposeful security breaches.

- **Networking System:**

Almost all of the information is shared between different systems via a network. Intercepting these data could be just as harmful as breaking into a computer. Henceforth, Network should be properly secured against such attacks.

Security System Goals

1. Integrity:

The objects in the system mustn't be accessed by any unauthorized user & any user not having sufficient rights should not be allowed to modify the important system files and resources.

2. Secrecy:

The objects of the system must be accessible only to a limited number of authorized users. Not everyone should be able to view the system files.

3. Availability:

All the resources of the system must be accessible to all the authorized users i.e only one user/process should not have the right to hog all the system resources. If such kind of situation occurs, denial of service could happen. In this kind of situation, malware might hog the resources for itself & thus preventing the legitimate processes from accessing the system resources.

Implementation

Implementation refers to that stage of project during which the theory is turned into practice i.e. converting soft ideas into actual process. In this stage physical system specifications are converted into working and reliable solution. This is where the system is developed. It is followed by testing and then again implementation.

Implementation phases:

- Coding: this includes implementation of the design document into executable programming language code. The output of the coding phase is the source code for the software that acts as input to the testing and maintenance phase.
- Integration and Testing: It includes detection of errors in the software. The testing process starts with a test plan recognizes test-related activities , such as test case generation, testing criteria and resource allocation of testing. The code is tested and mapped against the design document created in the design phase.
- Installation: New system is installed and rolled out.

The steps involved in this phase are:

1. Acquisition and installation of hardware and software.
2. Conversion: It actually means to convert the old data to new format for proper functioning of the application in the new system.
3. User Training: User in this case has to be trained to use the system properly so that it is easy for them to grab control over the use of the application.
4. Documentation: This provides details of how to operate the given software, application and website.

The hardware and relevant software required for running the application must be installed and fully checked before implementation. In this phase conversion plays a crucial role. It actually means to convert the old data to a new format for proper functioning of the application in the new system. During the phase all the required programs are loaded onto user's computer. User must be trained.

The documentation is a complete description of the system from the users point of view as it provides details of how to operate the given software and application. It also includes certain error messages that a user is expected to encounter during its usage and solution to the expected problems. It involves detailed and step by step information of the project development so as to modify or update as per the new user requirements.

Evaluation

Evaluation is a strategy used to determine the success and impact of projects, programs, or policies. It requires the evaluator to gather important information to analyze the process and outcome of a certain project. Project evaluation prompts changes in internal workflow, detects patterns in the target audience of the project, plans for upcoming projects or reports the value of projects to external stakeholders.

Types of project evaluation

The following are common types of project evaluation to implement in your projects:

Pre-project evaluation

Before beginning a project, your team could evaluate whether it is feasible to complete successfully. This often takes place naturally in the developmental stage of projects and is crucial for the effective execution of the project. It is important that all involved are aware of the objectives and goals before work begins.

Ongoing evaluation

Throughout the life cycle of the project, you may use metrics to verify completed tasks. This includes budget, percentage of completed tasks and the overall quality of the work delivered so far. Try to remain focused on your original objectives and goals as the project is underway, so your team remains on track.

Post-project evaluation

After the project is complete, it is important to analyze the outcomes and impacts of the project. Outcomes help measure how effective the project was in meeting the objectives and goals set at the beginning. Impacts may determine how successful the project was in creating a tangible change for the target audience.

Self-evaluation

At any point in the life cycle of the project, an individual can conduct a self-evaluation. Self-evaluation analyzes if their work is contributing to greater objectives and goals. Recognizing strengths and weaknesses, measuring their successes, and determining the scope of their impact can increase their ability to work effectively as part of the team.

External evaluation

Another option is hiring external agencies to perform evaluations for your projects. These agencies typically have no prior connection or involvement in the project, leading to a high level of impartiality when conducting the evaluation and concluding. External evaluation is valuable for projects that include a large number of stakeholders or have several moving pieces.

Maintenance

All the activities that occur after the completion of the program come under the program maintenance. Moreover, it is not a part of the implementation process directly but, it plays an important role. Furthermore, the cost of maintenance is more than the cost of development. Functions of program maintenance include the following:

- Finding and correcting the errors.
- Modifying the program to enhance it. Besides, we modify it usually to adapt to some new laws or government policies. Moreover, we also have to modify it in case if we want to change the hardware or operating system.
- Update the documentation part.
- Add new features and functions.
- Remove useless parts of code.

Types of Program Maintenance

The types are as follows:

Corrective Maintenance

In this process, the developers fix the errors which come after the use of the software. Moreover, the users themselves point out these errors.

Preventive Maintenance

This acts as a precaution since we modify the code to avoid any errors in the future.

Adaptive Maintenance

The changes in our environment may sometimes require making changes in the software as well. Therefore, it is adaptive maintenance. For example, changing the software according to some new rules of the government.

Perfective Maintenance

This involves aiming at keeping the software up to date with the latest technology. Moreover, programmers include any recommendations from the users, in the software.

Future Scope Of The Project

- ❖ Our application “Learn with Fun” has limitations . Its having limited games and limited things for kids so in future we wanted to add more in our application which will help them to expand their knowledge abundantly.
- ❖ Our application is age limited as it is kids app but in future we wanted it make it for every age group so they may use it for their benefit.
- ❖ This application has its own limitations but further there should not limitations regarding to age or games. Every age group should love our application.
- ❖ We will be regularly updating our application will try to add and make it more informational and creative.
- ❖ We will try to make it more user friendly so it can ne easily accessible.
- ❖ We will be adding more sections in our application.
- ❖ We will increase compatibility of our application so it can be used both online or offline.

Conclusion

- We have created this application for kids so they might explore their knowledge and get more confidence and learn new vocabularies.
- We have 2 different sections one is for younger kids from age 5 to 10 and other is for elder kids of age 11 to 15.
- We have created this application using python language which is very popular. Our application to look more attractive and eyesothening which will attract kids more toward it.
- Our application also have many pros and cons which will be improved as per the user's preferences.
- This application's games are not only for the entertainment but its main focus is help your children to improve and gain confidence among themselves.
- New vocabularies, mathematics, ryhthmings, pictures will help them to build up their memory power.
- This application is all about improvement and 100% beneficial.

Bibliography & References

References:-

1. <http://.tutorialspoint.com>
2. <http:Google.co.in>
3. YouTube
4. <https://app.upgrad.com/4Xxq/qn7lj80w>
5. <http://www.mygreatlearning.com>