A PROJECT REPORT ON

# RUBIK CUBE SOLVER AND ENCRYPTION

**SUBMITTED IN PARTIAL FULFILLMENT OF**

THE DEGREE M. Sc.IT (PART-II)


SUBMITTED BY

**SHANTANU JADHAV**


2022-2023

UNDER THE GUIDANCE OF

**DR. RAKHI GUPTA**

AND CO-GUIDANCE OF

**Prof. NASHRA GOWALKAR**


**UNIVERSITY OF MUMBAI**
**MASTER OF SCIENCE IN**
**INFORMATION TECHNOLOGY**
**KISHINCHAND CHELLARAM COLLEGE**
**D.W. ROAD, CHURCHGATE, MUMBAI: 400033**

**KISHINCHAND CHELLARAM COLLEGE**
CHURCHGATE, MUMBAI – 400 020.

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**M.SC. PART- II**

# CERTIFICATE

This is to certify that the practical done at **K.C. College** by

**MR/MS.** _____

**(**Seat No:_____) in partial fulfillment for M.SC. (I.T.) Degree Examination has been found satisfactory. This Practical journal had not been submitted for any other examination and does not form part of any other course undergone by the candidate.

| | | |
|---|---|---|
| **Signature** | **Signature** | **Signature** |
| **Lecturer-In-Charge** | **External Examiner** | **Course Coordination** |
| **Guided By** | **Examined By** | **Certified By** |

**College Stamp**

# ACKNOWLEDGEMENT

The success and outcome of this project were possible by the guidance and support from many people. I am incredibly privileged to have got this all along with the achievement of my project. It required a lot of effort from each individual involved in this project with me and I would like to thank them.

First of all, thanks to my parents for giving encouragement, enthusiasm and invaluable assistance to me. Without all this, I might not be able to complete my final year project and their cooperation at every stage.

Second, I would like to thank our honorable Principal,Dr. Tejashree V.Shanbhag and our Head of Department (B.Sc. IT and M.Sc. IT), Dr. Rakhi Gupta, for providing me the opportunity to do the project work at Kishinchand Chellaram College and also providing necessary facilities required for completion of this project.

Third, I would also like to express my sincere gratitude towards my project guide Prof. Nashra Gowalkar whose guidance and care made the project successful. She helped me every time when needed and she gave right direction towards completion of project.

I would also like to thank all my professors and all the other non-teaching staff of the M.Sc. IT Department of the college for all the help, support and co-operation that they offered during the course of this project.

Lastly, I would like to thank each and every person who directly or indirectly helped me in the completion of the project especially my peers who supported me throughout my project.

# INTRODUCTION

# ABSTRACT

A Rubik Cube is an interesting puzzle invented by 'Erno Rubik'. With the use of certain algorithms, it can be solved easily. There are many variations of the Rubik cube nowadays but the most basic one is the 3x3x3 Rubik's cube. We created a system that helps a user solve a Rubik's cube from any starting condition. The starting configuration of the cube is entered and displayed on the computer monitor to select the appropriate color for each cube face. Once a valid starting configuration has been provided, a visualization of the Rubik's cube helps guide the user towards solving the cube. Each time a designated button is pressed, one section of the displayed Rubik's cube rotates and the colors of the cubelet faces are updated to reflect the current state of the cube The algorithm used to determine the appropriate sequence of moves is a well known seven step method for solving the cube. If users accurately perform each of the rotations they see in the visualization on their physical cube, they should have a solved Rubik's cube in fewer than 200 rotations.

If we scramble a rubik cube many patterns can be formed that we can be used to encrypt a file. This project does the same, First we have to choose a file of our while ('.txt', '.png,' , '. Pdf' ) and use use the rubik cube to encrypt the file. Once the file is encrypted it cant be opened or read by anyone  to decrypt the file the same pattern of the rubik should be shown to the camera.

# PROBLEM DEFINITION

It was very difficult for normal people to solve rubik cube, because it has lots of patterns and it is very difficult to solve for normal person who have not done it in the past or the one who don't have knowledge about it. It is very difficult to solve without knowing patterns so for the person not knowing how to solve it will be impossible to solve rubik cube. Sometimes one wrong move would lead to more trouble and it will become more worse to solve the cube.

Encrypting files is a common way to protect sensitive information from unauthorized access.If the file is not encrypted it can be read by anyone thus giving up important and secret information to an unwanted person causing many problems.

# PROBLEM METHODOLOGY

Solving Rubik Cube can start out as curiosity for some and even end up as a hobby. It is quite interesting activity. However, learning to solve a Rubik's cube takes a lot of patience, it requires a lot of time but inevitably, it is truly rewarding. With help of our application, it will be  easy for one person to solve Rubik Cube. One can Learn how to solve and also be creative and productive.

It will help us to solve a Rubik cube with minimal number of steps. Person not knowing how to solve still can easily solve it with help of this application. It will show the shortest path with minimum number of steps to solve, so that it will be fast and efficient to solve. This will help us to solve the cube in less time.

Encrypting your files can give you an additional layer of security A  simple 3x3 rubik cube can be scrambled into 43 quintillion combination this amount of password creation generation makes it very hard to crack the password

With this application you can encrypt your file and use a the patterns created on the one of face of the rubik cube has a password  and the file woudnt be decrypted till the same pattern in used. This use of rubik cube has a password creates a physical key which can only be created by the owner of the files which will be a more secured that our normal password.

# REVIEW OF LITERATURE

The following papers were studied in order to get an overview of the techniques use to solve the rubik cube

**Rubik Cube Introduction:**
In the mid-1970s, Ernő Rubik worked at the Department of Interior Design at the Academy of Applied Arts and Crafts in Budapest. Although it is widely reported that the Cube was built as a teaching tool to help his students understand 3D objects, his actual purpose was solving the structural problem of moving the parts independently without the entire mechanism falling apart. He did not realise that he had created a puzzle until the first time he scrambled his new Cube and then tried to restore it Rubik applied for a patent in Hungary for his "Magic Cube" (Hungarian: Bűvös kocka) on 30 January 1975, and HU170062 was granted later that year.

**Thistlethwaite's algorithm**
The breakthrough, known as "descent through nested sub-groups" was found by Morwen Thistlethwaite; details of Thistlethwaite's algorithm were published in Scientific American in 1981 by Douglas Hofstadter. The approaches to the cube that led to algorithms with very few moves are based on group theory and on extensive computer searches. Thistlethwaite's idea was to divide the problem into subproblems. Where algorithms up to that point divided the problem by looking at the parts of the cube that should remain fixed, he divided it by restricting the type of moves that could be executed. In particular he divided the cube group into the following chain of subgroups:

- $G_0 = \langle L, R, F, B, U, D \rangle$
- $G_1 = \langle L, R, F, B, U^2, D^2 \rangle$
- $G_2 = \langle L, R, F^2, B^2, U^2, D^2 \rangle$
- $G_3 = \langle L^2, R^2, F^2, B^2, U^2, D^2 \rangle$
- $G_4 = \{1\}$

Next he prepared tables for each of the right coset spaces . For each element he found a sequence of moves that took it to the next smaller group. After these preparations he worked as follows. A random cube is in the general cube group . Next he found this element in the right coset space . He applied the corresponding process to the cube. This took it to a cube in . Next he looked up a process that takes the cube to , next to  and finally to .

Although the whole cube group  is very large (~4.3×1019), the right coset spaces  and  are much smaller. The coset space  is the largest and contains only 1082565 elements. The number of moves required by this algorithm is the sum of the largest process in each step.

**Kociemba’s Algorithm**

Kociemba's algorithm is a method for solving the Rubik's Cube puzzle, developed by Herbert Kociemba in 1992 . It is considered to be one of the fastest and most efficient algorithms for solving the cube. Kociemba's algorithm uses group theory and mathematical algorithms to search for the optimal solution. It improves upon Thistlethwaite's algorithm by reducing the number of intermediate groups from four to two . This reduction in the number of intermediate groups makes Kociemba's algorithm faster and more efficient than Thistlethwaite's algorithm.

Optimal solutions for Rubik's Cube refer to the shortest ways to solve the cube. The maximal number of face turns needed to solve any Rubik's Cube is 20, and the maximal number of quarter turns is 26 . Kociemba's algorithm aims to find the shortest possible solution to the Rubik's Cube puzzle by using mathematical algorithms and group theory . The algorithm searches through different combinations of moves to find the optimal solution.

Kociemba's algorithm is widely used in Rubik's Cube solving programs today. It has been shown to be highly effective and efficient in solving the Rubik's Cube puzzle . The algorithm is also used in conjunction with other algorithms to solve more complex Rubik's Cube configurations.

# SURVEY OF TECHNOLOGY

# DOMAIN: ARTIFICIAL INTELLIGENCE

Artificial intelligence (AI) makes it possible for machines to learn from experience, adjust to new inputs and perform human-like tasks. You can think of AI as being a form of intelligence that is used to solve problems, come up with solutions, answer questions, make predictions, or offer strategic suggestions. Because AI can do all these things, it's become incredibly important to modern businesses and other types of organizations.

AI systems work by combining large sets of data with intelligent, iterative processing algorithms to learn from patterns and features in the data that they analyze. Each time an AI system runs a round of data processing, it tests and measures its own performance and develops additional expertise. Because AI never needs a break, it can run through hundreds, thousands, or even millions of tasks extremely quickly, learning a great deal in very little time, and becoming extremely capable at whatever it's being trained to accomplish. The goal of AI science is to build a computer system that is capable of modeling human behavior so that it can use human-like thinking processes to solve complex problems.

AI automates repetitive learning and discovery through data. Instead of automating manual tasks, AI performs frequent, high-volume, computerized tasks. AI adds intelligence to existing products. Many products you already use will be improved with AI capabilities, much like Siri was added as a feature to a new generation of Apple products. Automation, conversational platforms, bots and smart machines can be combined with large amounts of data to improve many technologies. Upgrades at home and in the workplace, range from security intelligence and smart cams to investment analysis.

AI adapts through progressive learning algorithms to let the data do the programming. AI finds structure and regularities in data so that algorithms can acquire skills. Just as an algorithm can teach itself to play chess, it can teach itself what product to recommend next online. And the models adapt when given new data.

AI analyzes more and deeper data using neural networks that have many hidden layers. AI achieves incredible accuracy through deep neural networks. For example, your interactions with Alexa and Google are all based on deep learning. And these products keep getting more accurate the more you use them. In the medical field, AI techniques from deep learning and object recognition can now be used to pinpoint cancer on medical images with improved accuracy.

AI gets the most out of data. When algorithms are self-learning, the data itself is an asset. The answers are in the data. You just have to apply AI to find them. Since the role of the data is now more important than ever, it can create a competitive advantage. If you have the best data in a competitive industry, even if everyone is applying similar techniques, the best data will win.

# Sub domain: Computer Vision

Computer vision is an area of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos, and other visual input—and take action or make recommendations based on that information. If AI enables computers to think, computer vision enables them to see, observe and understand.

Computer vision works essentially the same as human vision, except humans have a head start. Human vision has the advantage of a lifetime of context to train how to distinguish objects from each other, how far they are, whether they are moving, and whether there is something wrong with an image.

Computer vision trains machines to perform these functions, but must do so in much less time using cameras, data and algorithms, rather than retinas, optic nerves and visual cortex. Because a system trained to inspect products or track manufacturing assets can analyze thousands of products or processes per minute and spot imperceptible defects or problems, it can quickly surpass human capabilities.

Computer vision needs a lot of data. It analyzes the data over and over again until it can spot the differences and eventually recognize the images. For example, to train a computer to recognize car tires, it needs to be fed a huge number of images of tires and tire-related items so that it can detect the differences and recognize a tire, especially a tire without punctures.

Machine learning uses algorithmic models that allow a computer to learn about the context of visual data. If enough data is passed through the model, the computer will "look" at the data and learn to distinguish one image from another. Algorithms allow the machine to learn on its own rather than being programmed by someone to recognize an image.

A CNN helps the "look" of a machine learning or deep learning model by dividing images into pixels that are given labels or labels. It uses the labels to perform convolutions (a mathematical operation on two functions to create a third function) and predict what it "sees". The neural network runs convolutions and checks the accuracy of its predictions in a series of iterations until the predictions begin to come true. This is then recognizing or seeing images in a human-like manner.

# Hardware Requirement:

- **Camera**
    - 720p Webcam (External / Inbuilt)
- **Laptop/PC:**
- **Recommended Specification :**
    - Intel (R) Core(TM) i5-6200U CPU @ 2 GHz or faster.
    - RAM: 4 GB
    - Storage: 500 GB
    - Architecture: 32-bit or 64-Bit Operating System

# SOFTWARE REQUIREMENTS

- Python 3.8 is used for this project
- Operating System: Windows 10 and above
  Mac OS Big Sur

## Visual Studio Code :

VS Code has been tested on the following platforms:

- OS X Yosemite
- Windows 7 (with .NET Framework 4.5), 8.0, 8.1 and 10 (32-bit and 64-bit)
- Linux (Debian): Ubuntu Desktop 14.04, Debian 7
- Linux (Red Hat): Red Hat Enterprise Linux 7, CentOS 7, Fedora 23

Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference

Visual Studio Code is a source-code editor made by Microsoft with the Electron Framework,
for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

Programming language:Python :

Python is the programming dialect utilized all through the extend. Python was the dialect of
choice for this venture. This was a straightforward choice for numerous reasons.Python as a dialect has an gigantic community behind it. Any issues that may
be experienced can be effectively illuminated with a trip to Stack Flood.
Python is among the foremost well known dialects on the location which makes it exceptionally likely there will be a coordinate reply to any inquiry.
Python has
an plenitude of effective apparatuses prepared for logical computing. Bundles such as Numpy, Pandas, and SciPy are openly accessible and well recorded. Bundles such as these can significantly decrease, and rearrange the code required to compose a given program. This makes cycle speedier.

Package Used:

- OpenCv
  - Numpy
  - Kociemba
  - Sql lite

**OpenCv:**

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was design for real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

• Image/video I/O, processing, display
• Object/feature detection
• Geometry-based monocular or stereo computer vision • Computational photography
• Machine learning & clustering

**Numpy:**

NumPy is a Python library used for working with arrays.
It also has functions for working in the domain of linear algebra, fourier transform, and matrices. NumPy stands for Numerical Python.

At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance.

**Installation:**
Mac and Linux users can install NumPy via pip command:

*pip install numpy*

Windows does not have any package manager analogous to that in linux or mac. We have to download the pre-built windows installer for NumPy. And then install the packages manually.

**Kociemba:**

This Python bundle contains two comparable executions (in C and Python) of Herbert Kociemba's two-phase calculation for tackling Rubik's 3d shape.
The two-phase calculation does not ensure that the delivered arrangement is the most brief conceivable. Instep, it gives you a "great sufficient" arrangement in a really brief time. You'll be able actualize extra checks on beat of this library, for illustration, to not deliver any moves on the off chance that the 3d shape is as of now illuminated.

**SQL lite:**

SQLite may be a C library that gives a lightweight disk-based database that doesn’t require a partitioned server process and permits getting to the database employing a nonstandard variation of the SQL inquiry dialect. A few applications can utilize SQLite for inner data storage. It’s too conceivable to model an application utilizing SQLite and after that harbour the code to a bigger database such as PostgreSQL or Prophet.

# REQUIREMENTS AND SYSTEM ANALYSIS

- The exceptionally to begin with stage in any framework creating life cycle is preparatory examination. The achievability think about may be a major portion of this stage. A degree of how useful or commonsense the advancement of any data framework would be to the organization is the possibility ponder.
- The achievability of the advancement computer program can be examined in terms of the taking after viewpoints:

**1) Operation Feasibility :**

Business Performance is a measure of how well the demand solves the problem using the time specified during the run and meets the requirements determined during the development needs assessment. Effectiveness examines the organization's willingness to sup port the planning process. This is probably the hardest part of the process. To determine this value, it is important to understand management's commitment to the offer. If the request is initiated by management, there is likely to be management support and acceptance and implementation of the system.However, it is important that the working group accept the change. Study is the possibility of effective use after development. If users have problems with the new system, it will not produce the desired results. Evaluates the feasibility of the system based on the PIECES framework.

**2) Technical Feasibility:**

A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements.

- Intel (R) Core(TM) i5-6200U CPU @ 2 GHz or faster.
- RAM: 4 GB
- Storage: 500 GB
- Architecture: 32-bit or 64-Bit Operating System

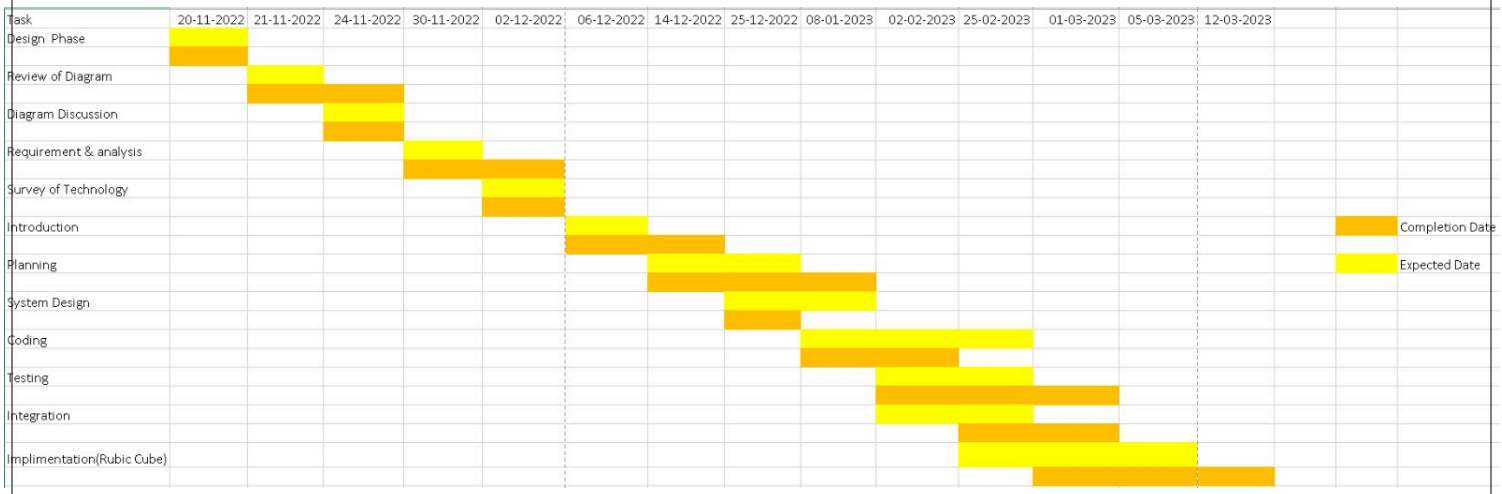
1. **Economic Feasibility:**
   The software will be available as a free version with basic features and a premiumversion with advanced features.

2. **Legal Feasibility:**
   All the software's requirements for this project are free and open source. There are no special hardware requirements other than the basics.

| Sr.No. | Contents | Proposed Date | Submission Dates | Teacher's Signature | Remark |
|---|---|---|---|---|---|
| 1 | **Investigation** | | | | |
| | Project Fixing | 14/10/2022 | | | |
| | Synopsis | | | | |
| 2 | **Analysis** | | | | |
| | Project History | 19/10/2022 | | | |
| | Requirement Gathering | | | | |
| | Objective & scope of the project | | | | |
| | Problems with existing system | | | | |
| | Advantages of proposed system | | | | |
| | Feasibility study | 24/10/2022 | | | |
| | Cost benefits analysis | | | | |
| | Requirement Specification | | | | |
| | Tools and Technology | | | | |
| 3 | **Design Phase** | | | | |
| | Detailed Lifecycle of the project (Logical design) | 03/11/2022 | | | |
| | ER Diagram | | | | |
| | Use Case Diagram | | | | |
| | Activity Diagram | | | | |
| | System Flowchart | | | | |
| | Sequence Diagram | | | | |
| | Data Flow Diagram | | | | |
| 4 | **Coding Phase** | | | | |
| | Forms | | | | |
| | Modules Design | | | | |
| | Validating Forms/ Application | | | | |
| 5 | **Testing Phase** | | | | |
| | Module Testing/Unit Testing | | | | |
| | Integration Testing | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | System Testing | | | | |
| | Acceptance Testing | | | | |
| **6** | **Evaluation and Enhancement** | | | | |
| | System maintenance and future | | | | |
| | Enhancement | | | | |
| | User Manual | | | | |
| **7** | **Review** | | | | |
| **8** | **Project/Black-Book and Back-up Softcopy Submission** | 27/04/2023 | | | |

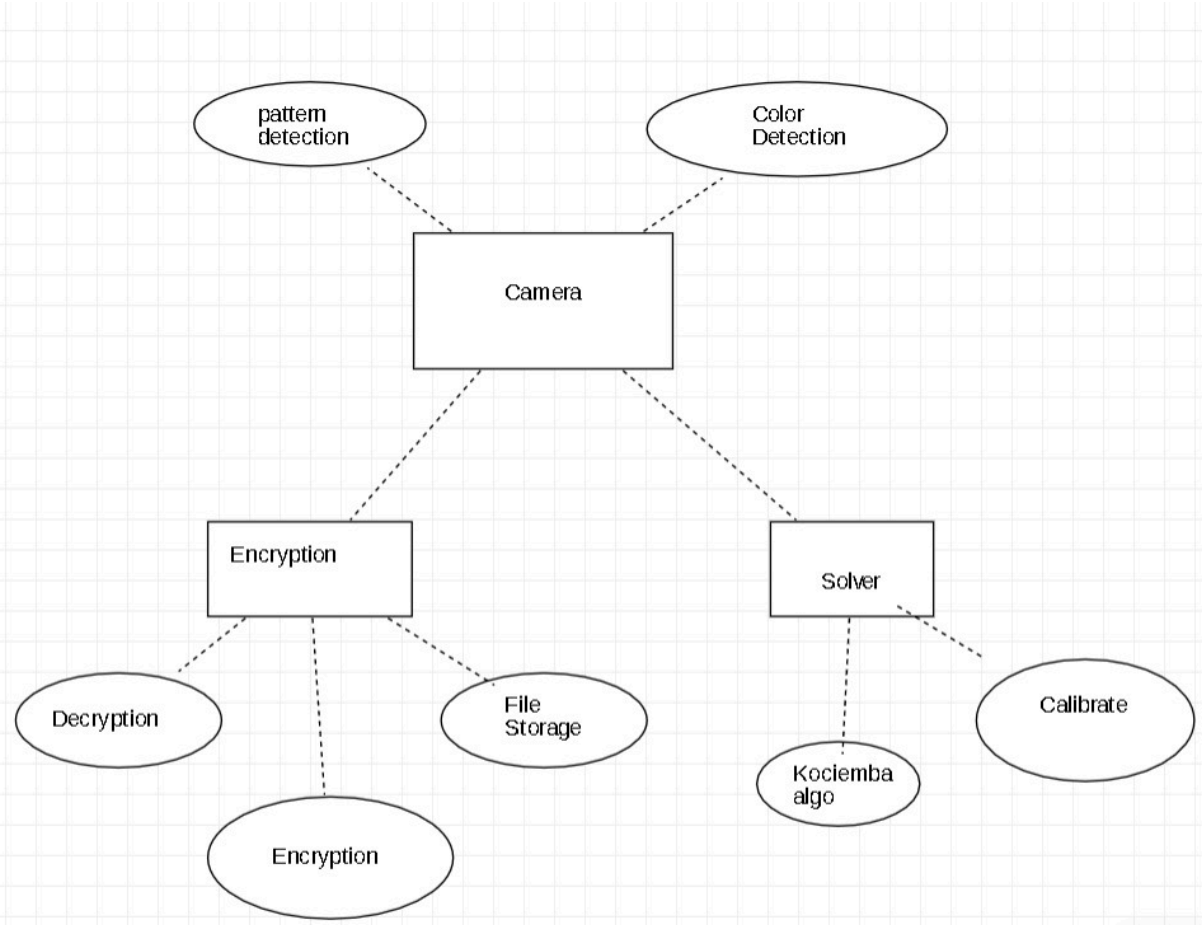
# GANTT CHART


Task
20-11-2022
21-11-2022
24-11-2022
30-11-2022
02-12-2022
06-12-2022
14-12-2022
25-12-2022
08-01-2023
02-02-2023
25-02-2023
01-03-2023
05-03-2023
12-03-2023
Design Phase
Review of Diagram
Diagram Discussion
Requirement & analysis
Survey of Technology
Introduction
Planning
System Design
Coding
Testing
Integration
Implimentation(Rubic Cube)
Completion Date
Expected Date

# SYSTEM DESIGN

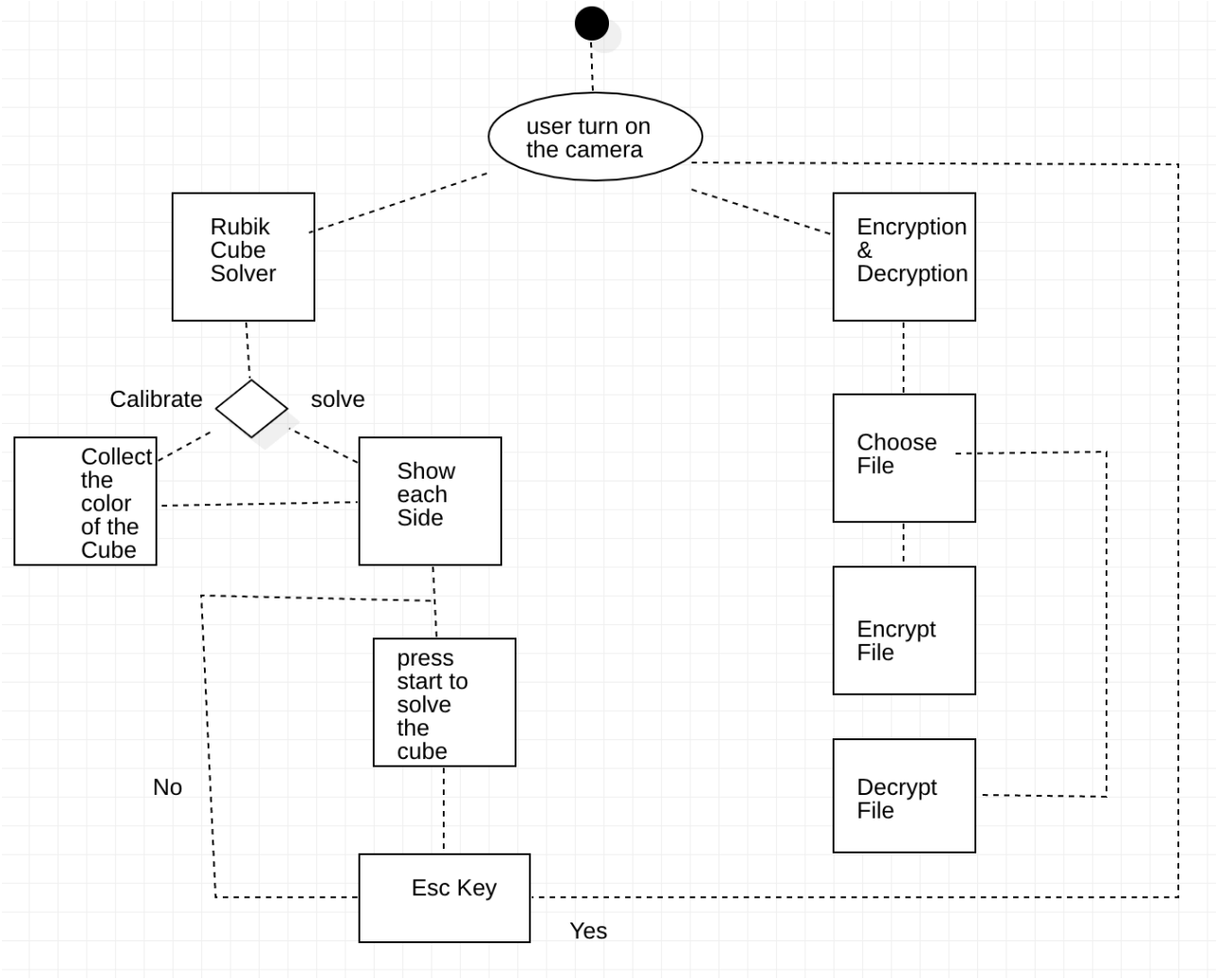
# ER DIAGRAM

Entity-Relationship Model (ER Model) is a data modeling technique that graphically illustrates an information system's entities and the relationships between those entities. An ER Diagram is a conceptual and representational model of data used to represent the entity framework infrastructure.



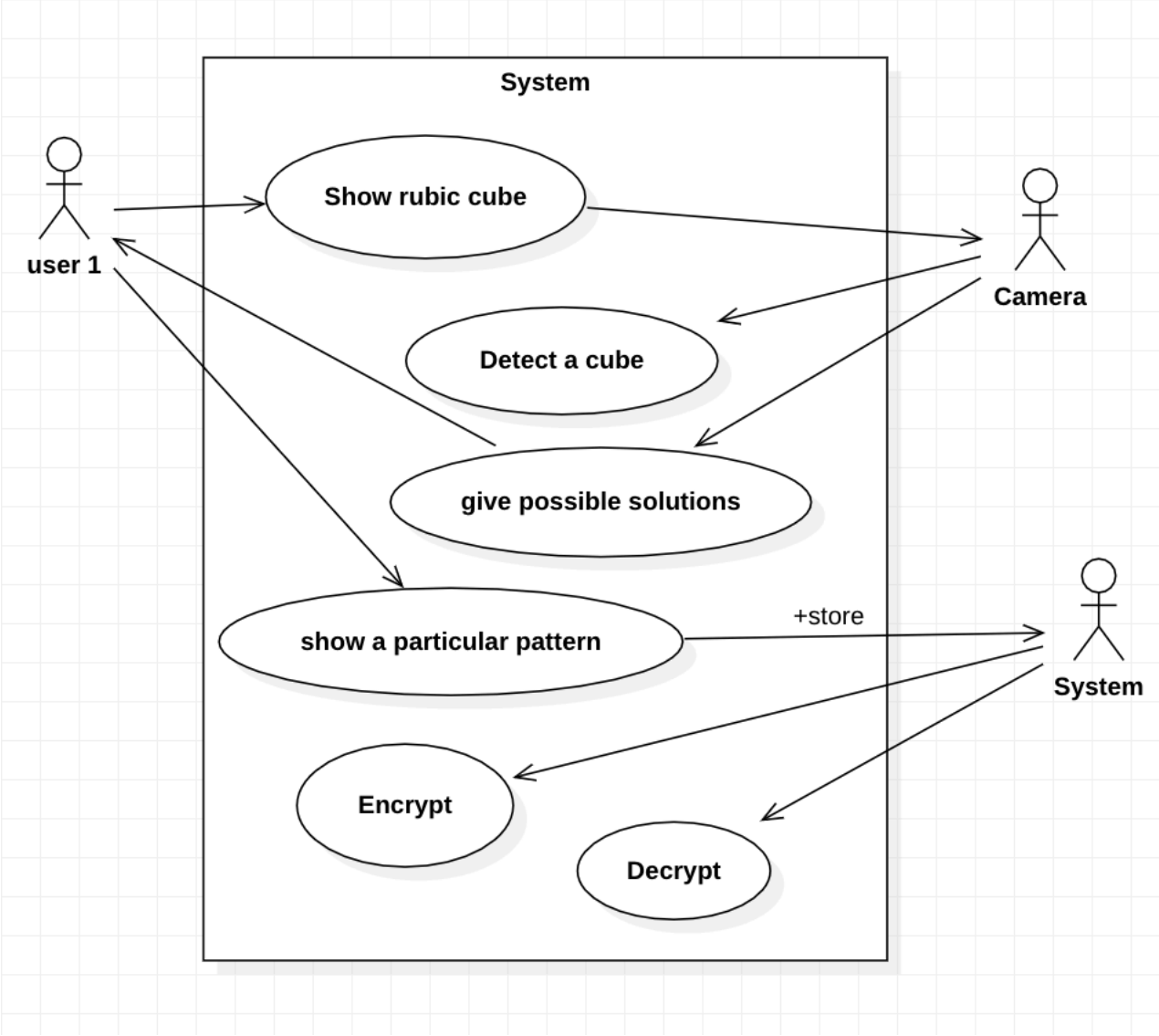
# ACTIVITY DIAGRAM

An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system.
An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.



# USE CASE DIAGRAM

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

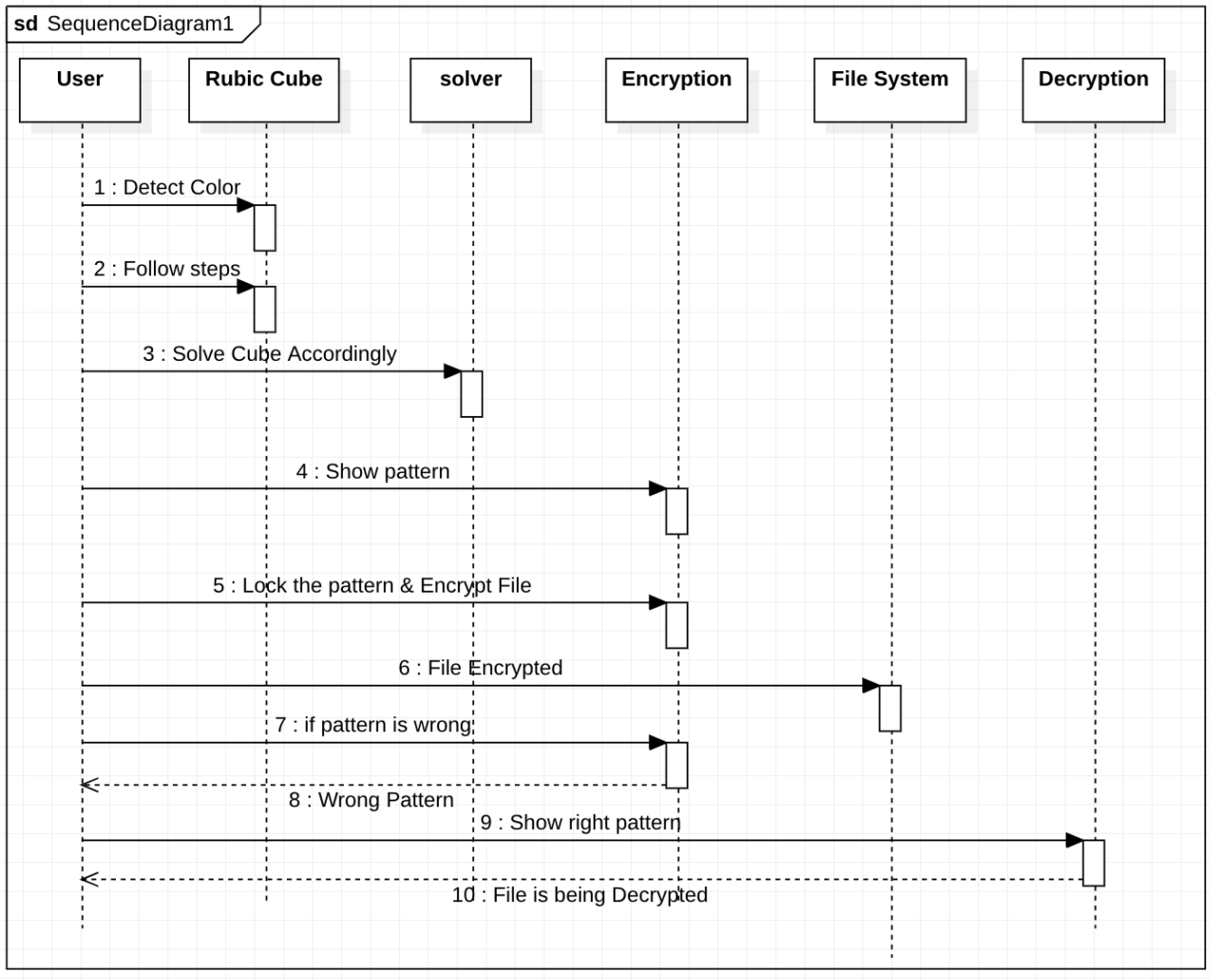


# SEQUENCE DIAGRAM

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. Sequence diagrams are sometimes known as event diagrams or event scenarios.

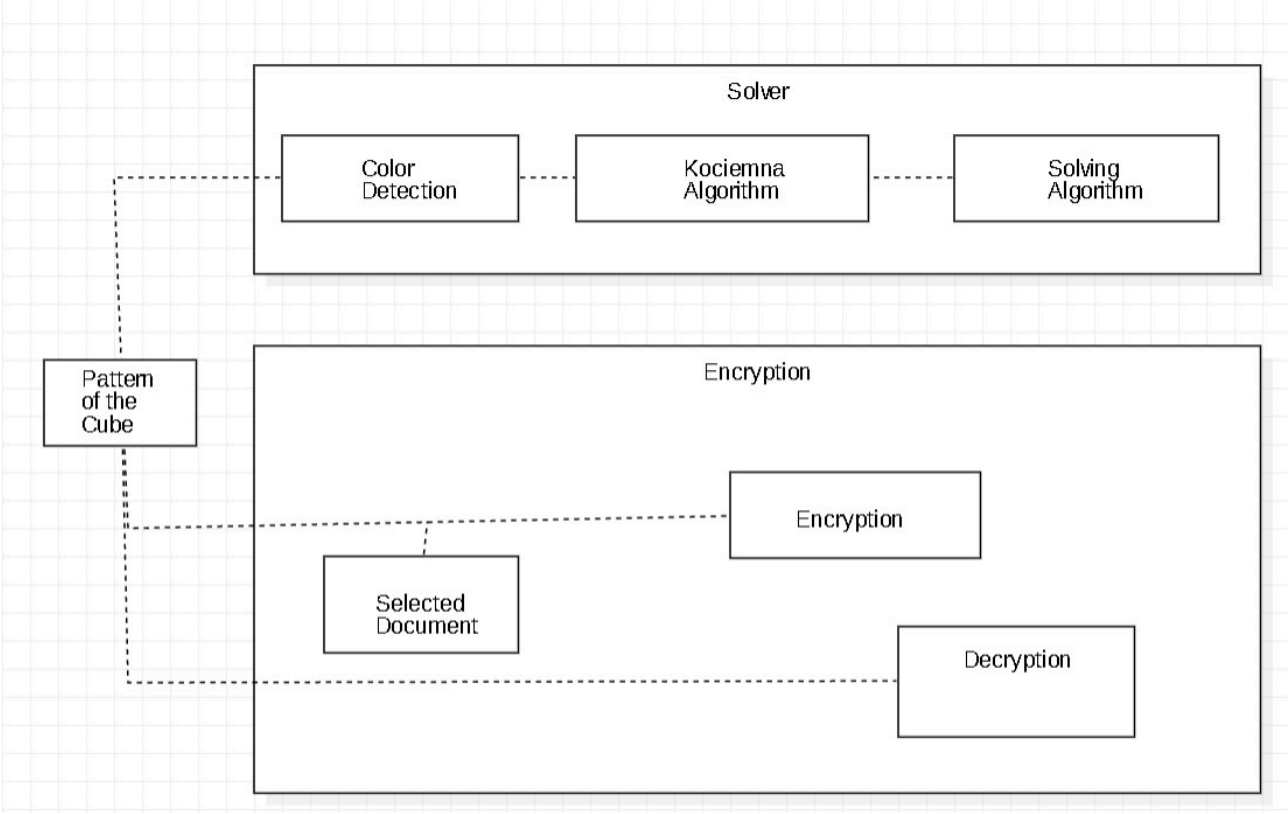
Sequence diagrams can be useful references for businesses and other organizations. Try drawing a sequence diagram to:

- Represent the details of a UML use case.
- Model the logic of a sophisticated procedure, function, or operation.
- See how objects and components interact with each other to complete a process.
- Plan and understand the detailed functionality of an existing or future scenario.



# DATAFLOW DIAGRAM

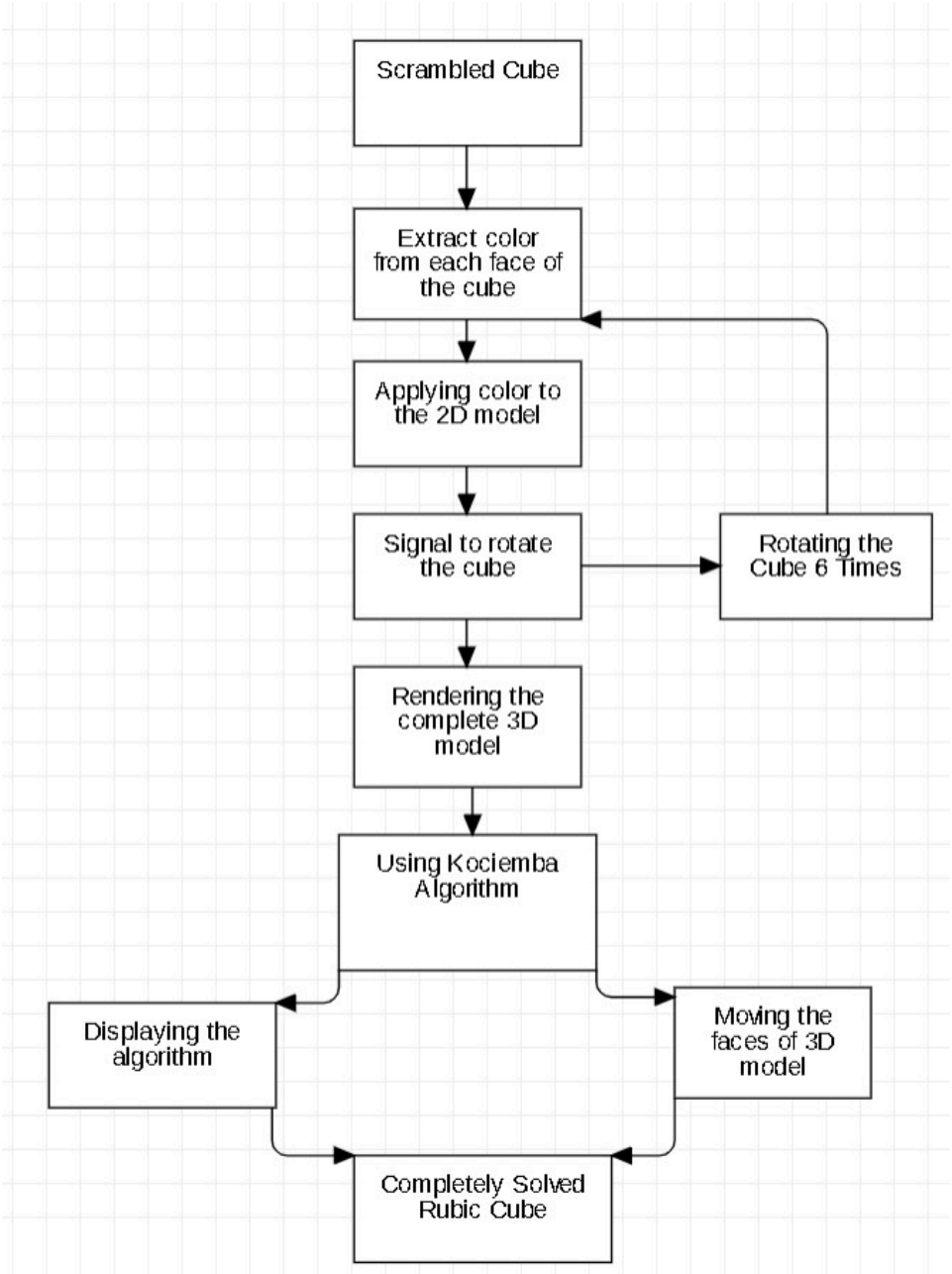
A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.



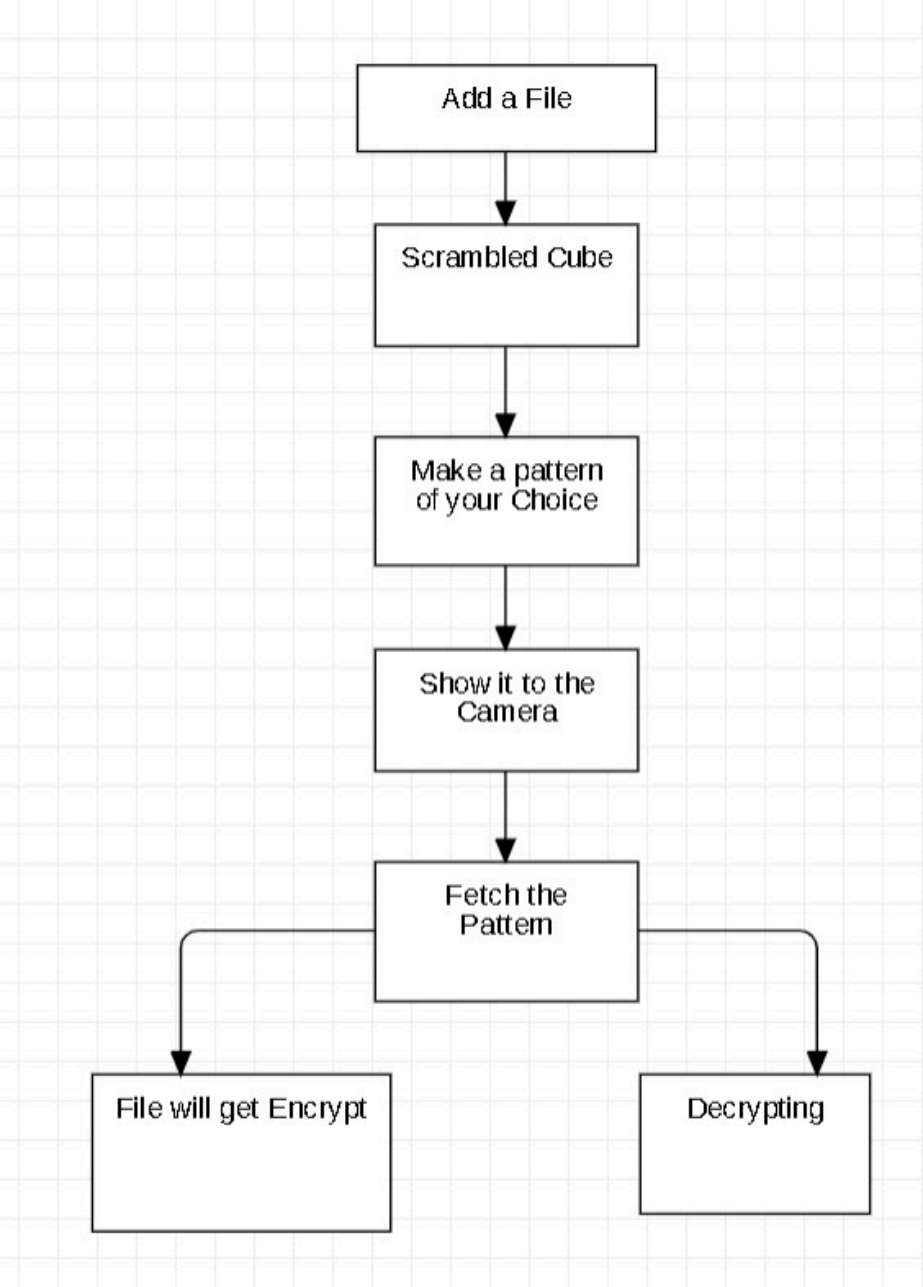
# SYSTEM FLOWCHART

A flowchart is a graphical representation of steps. It was originated from computer science as a tool for representing algorithms and programming logic but had extended to use in all other kinds of processes. Nowadays, flowcharts play an extremely important role in displaying information and assisting reasoning. They help us visualize complex processes, or make explicit the structure of problems and tasks. A flowchart can also be used to define a process or project to be implemented.

**Rubik cube Solver**



# Encryption using cubic cube



# SYSTEM IMPLEMENTATION

# Code listing

## main.py

```
import cv2
from PIL import Image
import time
import tkinter as tk
from PIL import Image, ImageTk
from tkinter.ttk import *
import pandas as pd
import numpy as np
import recorder as rc
from tkinter import ttk
import tkinter.messagebox as mb
import traceback
import kociema_module as kc
import colorExtractor as ce
from colorlabeler import cubestr
import kociemba
from tkinter import filedialog
import os
import shutil
from cryptography.fernet import Fernet
import base64
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.kdf.hkdf import HKDF
import sqlite3 as sl


con = sl.connect('rubiks.db')


with con:
    con.execute("""
        CREATE TABLE IF NOT EXISTS COLORS (
            id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
            color_name TEXT NOT NULL,
            color_lower TEXT NOT NULL,
            color_upper TEXT NOT NULL
        );
    """)


faces=""
result={}
converted=""
final_string={}
data=""
```

```
def convert(string):
    li = list(string.split(" "))
    return li


def toggleSolveText():
    if Solve_button['text'] == 'Solve':
        Solve_button['text'] = 'Solving'
        Record_button['state'] = 'disabled'
    elif Solve_button['text'] == 'Solving':
        Solve_button['text'] = 'Solve'

def toggleEncryptText():
    if Encrypt_button['text'] == 'Encrypt':
        Record_button['state'] = 'disabled'
        Decrypt_button['state'] = 'disabled'
        Encrypt_button['state'] = 'normal'


def toggleDecryptText():
    if Decrypt_button['text'] == 'Decrypt':
        Record_button['state'] = 'disabled'
        Decrypt_button['state'] = 'normal'
        Encrypt_button['state'] = 'disabled'


def toggleRecordText():
    global converted
    global result
    global data
    #print(result)
    try:

        for key in result:
            converted = converted+ result[key]
        if(Record_button['text'][0]!="S"):
            final_string[Record_button['text'][0]]=list(converted)
        converted=""
        #print(final_string)
        if len(final_string)==6:
            data=cubestr(final_string)
            try:
                Solve_label['text'] = str(kociemba.solve(data))
            except Exception as e:
                print(e)
                Solve_label['text'] = str(e)+": Try Again"
            #Record_button['text'] = 'Start Camera'
            #print("here",kociemba.solve(data))
    except Exception as e:
        print(e)

    if Record_button['text'] == 'Start Camera':
        Record_button['text'] = 'Front'
    elif Record_button['text'] == 'Front':
        Record_button['text'] = 'Upper'
    elif Record_button['text'] == 'Upper':
```

```
        Record_button['text'] = 'Down'
    elif Record_button['text'] == 'Down':
        Record_button['text'] = 'Left'
    elif Record_button['text'] == 'Left':
        Record_button['text'] = 'Right'
    elif Record_button['text'] == 'Right':
        Record_button['text'] = 'Back'
    elif Record_button['text'] == 'Back':
        Record_button['text'] = 'Recorded'


def toggleCalibrateText():
    print("result",Calibrate_button['text'], result)
    if Calibrate_button['text']!="Calibrate":
        sql = 'INSERT INTO COLORS (color_name, color_lower,color_upper) values(?, ?, ?)'
        data = [str(Calibrate_button['text']),str(result.split('*')[0]),str(result.split('*')[1])]
        with con:
            con.execute(sql, data)
    if Calibrate_button['text'] == 'Calibrate':
        Calibrate_button['text'] = 'Red'
    elif Calibrate_button['text'] == 'Red':
        Calibrate_button['text'] = 'Blue'
    elif Calibrate_button['text'] == 'Blue':
        Calibrate_button['text'] = 'Green'
    elif Calibrate_button['text'] == 'Green':
        Calibrate_button['text'] = 'Yellow'
    elif Calibrate_button['text'] == 'Yellow':
        Calibrate_button['text'] = 'White'
    elif Calibrate_button['text'] == 'White':
        Calibrate_button['text'] = 'Orange'
    elif Calibrate_button['text'] == 'Orange':
        Calibrate_button['text'] = 'Calibrate'


class FileSaver():
    def __init__(self):
        file_path=None
        self.file_path=file_path
        self.get_file_path()
        self.save_file()

    def get_file_path(self):
        self.file_path= filedialog.askopenfilename(title = "Select A File")

    def save_file(self):
        global filename
        directory = os.getcwd()
        #print(directory)
        origin=self.file_path
        filename=origin.split('/')[-1]
        target=directory.replace("\\","/")+'/'+filename
        print(origin,target)
        try:
            shutil.copy2(origin,target)
            browse_button['text']=filename
            print(filename)
```

# Methodology explained

## Two-Phase

The core of the 2-phase approach is this:

- Solve the cube into a state with a certain property.
- Solve the rest of the cube.

If we select F2L as our property, we get:

- Solve the first two layers.
- Solve the last layer.
    - If you solve F2L with the fewest moves possible, then solve the resulting LL case optimally, you'll end up with a decently short solution. However, there might be a shorter solution: in that case, the first phase might take more moves, but get us an easy LL case (or an LL skip!).

Iterative Deepening (for Phase 1)

- Let's say our first solution took 15 moves for F2L and 11 moves for LL. That means that our cube takes at most 15 + 11 = 26 moves to solve.
- So let's try looking at all the solutions that take 15 moves for F2L. If any of them give us an LL that takes fewer than 11 moves, we can find a shorter solution. In fact, we can extend this and keep looking. Here's our plan:
    - Try all the solutions that take 15 moves for F2L and fewer than 26 - 15 = 11 moves for LL.
    - Try all the solutions that take 16 moves for F2L and fewer than 26 - 16 = 10 moves for LL.
    - Try all the solutions that take 17 moves for F2L and fewer than 26 - 17 = 9 moves for LL.
    - ...
    - Try all the solutions that take 25 moves for F2L and fewer than 26 - 25 = 1 moves for LL.
    - Try all the solutions that take 26 moves for F2L and fewer than 26 - 26 = 0 moves for LL.

If there is a solution that takes fewer than 26 moves for the entire cube, F2L clearly has to be finished in at most 26 moves. Therefore, our optimal overall solution has to be in one of these cases.

Now let's say that along the way we find a solution that takes 17 moves for F2L and 8 moves for LL. Now we know that the entire cube can be solved in 17 + 8 = 25 moves. We abort the original 26-plan and continue our search wtih a new upper bound:

- Try all the solutions that take 17 moves for F2L and fewer than 25 - 17 = 8 moves for LL.
- Try all the solutions that take 18 moves for F2L and fewer than 25 - 18 = 7 moves for LL.
- ...
- Try all the solutions that take 25 moves for F2L and fewer than 25 - 25 = 0 moves for LL.

```
        except FileNotFoundError:
            print("File not found")


def Encrypter():
    hkdf = HKDF(algorithm=hashes.SHA256(),length=32,salt=None, info=None,
backend=default_backend())
    global keye
    print("result_en",result)
    l=""
    for key in result:
        l = l+ result[key]
    print(l)
    keye = base64.urlsafe_b64encode(hkdf.derive(bytes(l,'utf-8')))
    print("enkeye",keye)
    fernet = Fernet(keye)
    with open(filename, 'rb') as file:
        original = file.read()
    encrypted = fernet.encrypt(original)
    with open(filename, 'wb') as encrypted_file:
        encrypted_file.write(encrypted)
    file.close()
    Solve_label['text'] = "Encrypted"
    Encrypt_button['state'] = 'disabled'
    Decrypt_button['state'] = 'normal'



def Decrypter():
    print("result_dec",result)
    l=""
    for key in result:
        l = l+ result[key]
    print(l)
    hkdf = HKDF(algorithm=hashes.SHA256(),length=32,salt=None, info=None,
backend=default_backend())
    new_keye = base64.urlsafe_b64encode(hkdf.derive(bytes(l,'utf-8')))
    print("keye",keye)
    print("new_keye",new_keye)
    if not new_keye==keye:
        Solve_label['text'] = "Wrong password"
    else:
        fernet = Fernet(new_keye)
        with open(filename, 'rb') as enc_file:
            encrypted = enc_file.read()
        decrypted = fernet.decrypt(encrypted)
        with open(filename, 'wb') as file:
            file.write(decrypted)
        file.close()
        Solve_label['text'] = "Decrypted"
        Encrypt_button['state'] = 'normal'
        Decrypt_button['state'] = 'disabled'


def draw_label(face, frame):
        #text = "Class: {}".format(lbl)
```

```
        text=str(face)
        pos = (10,30)
        scale = 1
        thickness = 2
        lineType = 2
        fontColor = (0, 0, 255)
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(frame,
                    text,
                    pos,
                    font,
                    scale,
                    fontColor,
                    thickness,
                    lineType)
        return frame

class MainWindow:
    global faces
    def __init__(self,window,cap,action,face,dt):
        self.window = window
        self.cap = cap
        self.action=action
        self.dt=dt
        self.interval = 'idle'
        if result!="Error" and cap.isOpened()==False:
            self.cap=cv2.VideoCapture(0)
        if Record_button['text']!="Start Camera" or  Calibrate_button['text']!="Calibrate":
            print("2",Calibrate_button['text'],Record_button['text'])
            self.update_image()

    def update_image(self):
        global result
        global data
        global cap
        try:
            if Record_button['text']!="Start Camera" or Solve_button['text']=="Solving" or
Calibrate_button['text']!="Calibrate":
                faces=Record_button['text']
                #print("here")
                self.robj = self.action(self.cap.read()[1],faces[0],self.dt)
                if Solve_button['text']=="Solving" or Calibrate_button['text']!="Calibrate":
                    draw_label("",self.robj.frame)
                else:
                    draw_label("Face: "+faces[0],self.robj.frame)
                result=self.robj.result
                if result=="Completed":
                    Solve_button['text'] = 'Solve'
                    Record_button['state'] = 'normal'
                    Record_button['text'] ='Start Camera'
                elif result=="Error":
                    Solve_button['text'] = 'Solve'
                    Record_button['state'] = 'normal'
                    Record_button['text'] ='Start Camera'
                    self.cap.release()
                    cv2.destroyAllWindows()
```

```
                placeholderImage(self.window)
            elif self.robj != []:
                try:
                    if self.robj.frame!=[]:
                        #print("jerr")
                        #cv2.imshow("frame",self.robj.frame)
                        #self.frame=cv2.flip(self.frame,1)
                        self.frame=cv2.resize(self.robj.frame,(int(ui_w/1.5),int(ui_h/1.5)))
                        image = cv2.cvtColor(self.frame, cv2.COLOR_BGR2RGB)
                        image = Image.fromarray(image)
                        image = ImageTk.PhotoImage(image)
                        self.window.configure(image=image)
                        self.window.image = image
                        self.window.after(self.interval, self.update_image)
                    except Exception as e:
                        print(e)
                        self.cap.release()
                        cv2.destroyAllWindows()
                        placeholderImage(self.window)

            else:
                self.cap.release()
                cv2.destroyAllWindows()
                placeholderImage(self.window)
        except Exception as e:
            print(e)
            traceback.print_exc()
            #self.cap.release()
            print("Error")

    def __del__(self):
        print ('stopped')


def placeholderImage(panel):
    image = cv2.imread('background.png')
    image = cv2.resize(image, (600, 480))
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = Image.fromarray(image)
    image = ImageTk.PhotoImage(image)
    panel.configure(image=image)
    panel.image = image


if __name__ == '__main__':
    root = tk.Tk()
    root.title('Rubik Cube Solver')
    root.geometry('800x600')
    root.resizable(0, 0)
    ui_w=800
    ui_h=600
    photo = tk.PhotoImage(file = "logo.png")
    root.iconphoto(False, photo)

    style = ttk.Style()
    style.theme_use("xpnative")
```

```
    style.map("C.TButton",
          foreground=[('pressed', '#34B7F1'), ('active', 'red')],
          background=[('pressed', 'black'), ('active', '#25D366')]
          )
    style.configure('C.TButton', font=('Helvetica', 15),background="#4876d0")
    style.configure('C.TLabel', font=('Helvetica', 12 ),background="#4876d0" ,foreground="white")
    style.configure('C.TEntry', font=('Helvetica', 12, 'bold'),foreground="#4876d0")

    video_window = ttk.Label(root,padding=-2,compound="image")
    video_window.place(x=ui_w/20,y=ui_h/40, width=ui_w/1.5,height=ui_h/1.5)

    cap = cv2.VideoCapture(0)

    Calibrate_button = ttk.Button(root, text="Calibrate", style="C.TButton", command=lambda :
[toggleCalibrateText(),MainWindow(video_window,cap,ce.colorExtractor,"","")])
    Calibrate_button.place(x=ui_w/1.3,y=ui_h/35, width=150,height=50)

    Record_button = ttk.Button(root, text="Start Camera", style="C.TButton", command=lambda :
[toggleRecordText(),MainWindow(video_window,cap,rc.recorder,"","")])
    Record_button.place(x=ui_w/1.3,y=ui_h/8, width=150,height=50)


    Solve_button = ttk.Button(root, text="Solve", style="C.TButton", command=lambda :
[toggleSolveText(),MainWindow(video_window,cap,kc.kociemba_module,"",data)])
    Solve_button.place(x=ui_w/1.3,y=ui_h/4.5, width=150,height=50)

    browse_button = ttk.Button(root, text = "Open File",style="C.TButton",  command=FileSaver)
    browse_button.place(x=ui_w/1.3,y=ui_h/2.9, width=150,height=50)

    Encrypt_button = ttk.Button(root, text="Encrypt", style="C.TButton", command=lambda :
[toggleEncryptText(), Encrypter()])
    Encrypt_button.place(x=ui_w/1.3,y=ui_h/2.3, width=150,height=50)

    Decrypt_button = ttk.Button(root, text="Decrypt", style="C.TButton", command=lambda :
[toggleDecryptText(), Decrypter()])
    Decrypt_button.place(x=ui_w/1.3,y=ui_h/1.9, width=150,height=50)
    Decrypt_button['state'] = 'disabled'

    Solve_label = ttk.Label(root, text="Result Moves will be shown here", style="C.TLabel",
anchor=tk.CENTER)
    Solve_label.place(x=ui_w/20,y=ui_h/1.4, width=ui_w/1.1,height=ui_h/5)


    root.mainloop()
```

## Code used to extract and calibrate the color :

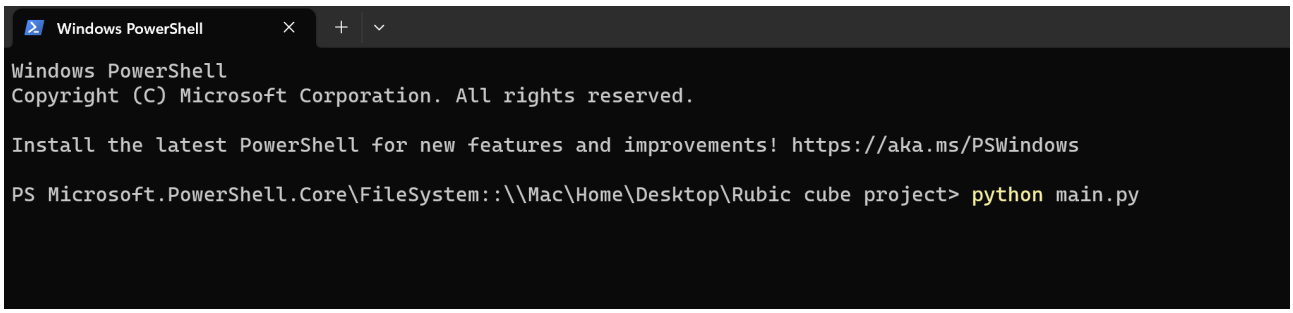
## colorExtractor.py

```
import cv2
import numpy as np
from PIL import Image

class colorExtractor():
    x1=100
    x2=150
    y1=150
    y2=200
    x=int((x1+x2)/2)
    y=int((y1+y2)/2)
    result=[]
    def __init__(self,frame,face,dt):
        image_hsv = None
        self.frame=frame
        self.extract()

    def pick_color(self):
            pixel = image_hsv[self.y,self.x]
            upper =  np.array([pixel[0] + 10, pixel[1] + 10, pixel[2] + 40])
            lower =  np.array([pixel[0] - 10, pixel[1] - 10, pixel[2] - 40])
            #print(lower, upper)
            data = np.zeros((50, 50, 3), dtype=np.uint8)
            for i in range(0,50):
                for j in range(0,50):
                    data[i,j] = (upper+lower)/2
            img = Image.fromarray(data, 'HSV')
            self.result=str(pixel[0] - 10)+","+str(pixel[1] - 20)+","+str(pixel[2] - 40)+"*"+str(pixel[0] +
10)+","+str(pixel[1] + 20)+","+str(pixel[2] + 40)
            new=cv2.cvtColor(np.array(img), cv2.COLOR_HSV2BGR)
            cv2.imshow('captured color',new)



    def extract(self):
        global image_hsv
        cv2.rectangle(self.frame, (self.x1,self.y1), (self.x2,self.y2), (0, 0, 0), 3)
        image_hsv = cv2.cvtColor(self.frame,cv2.COLOR_BGR2HSV)
        self.pick_color()
        return self.frame
```

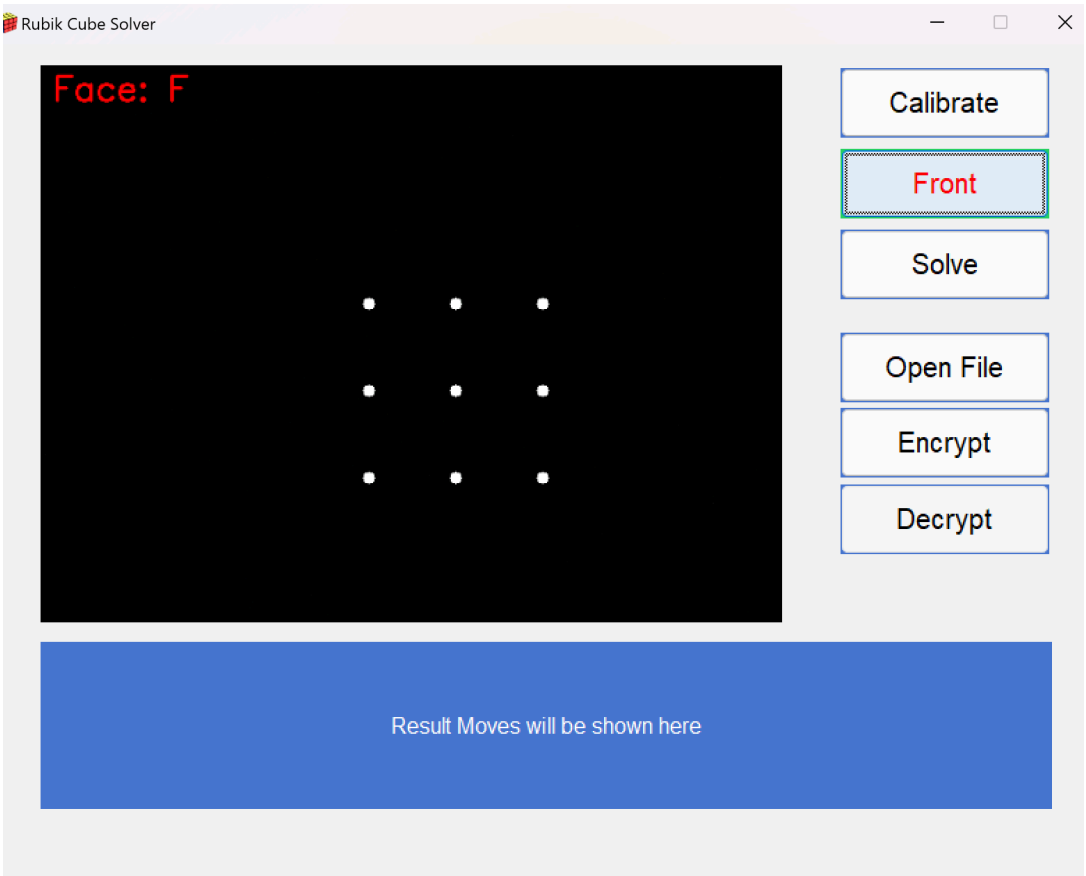
If we keep going, we reach a point where any solution we haven't considered must be longer than the one we have -- so our solution is optimal. For 3x3x3, we happen to know that the optimal solution must be at most 20 moves... but that's only because we know God's number. The approach works even if we don't know God's number (so can be used for any twisty puzzle).

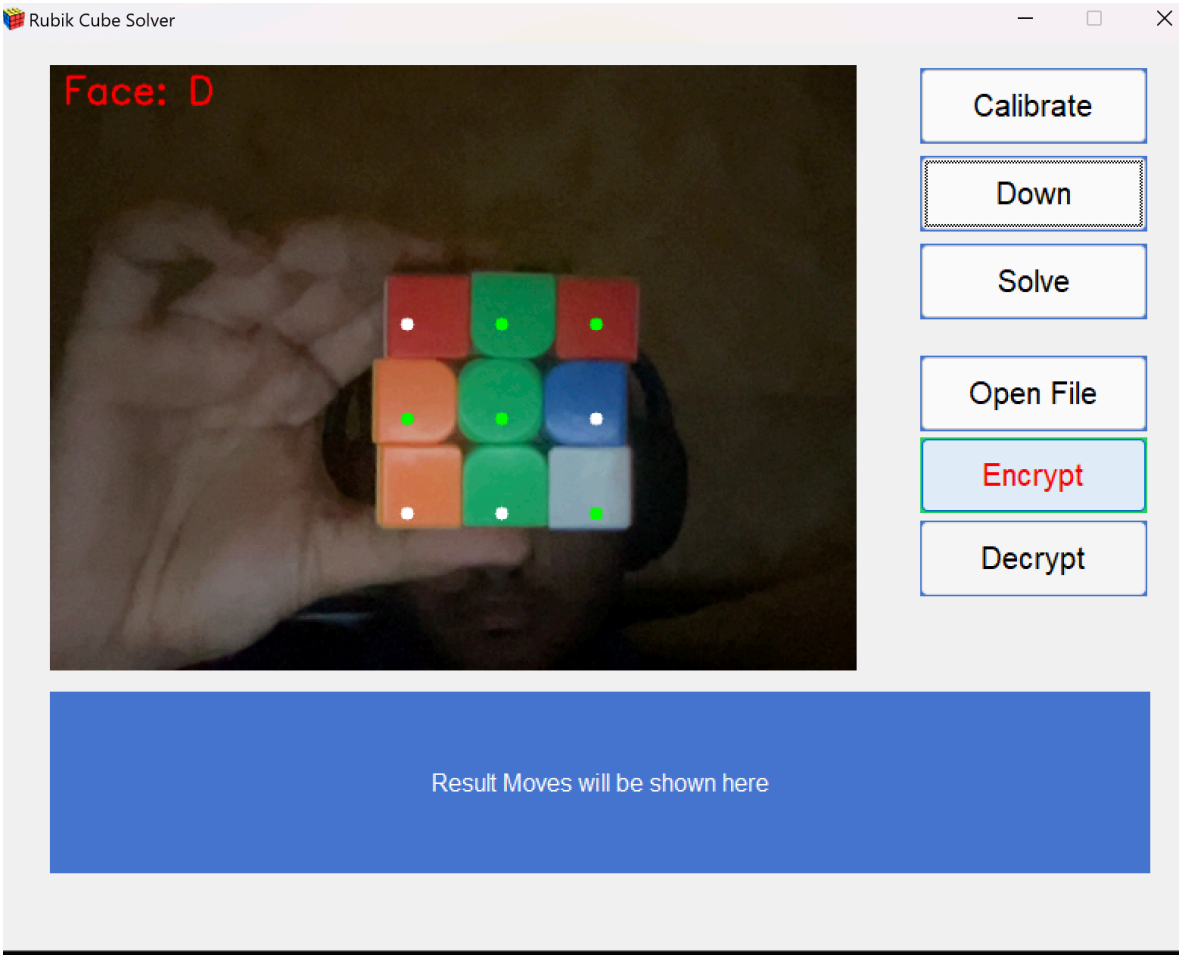
# USER MANUAL

# SCREENSHOT



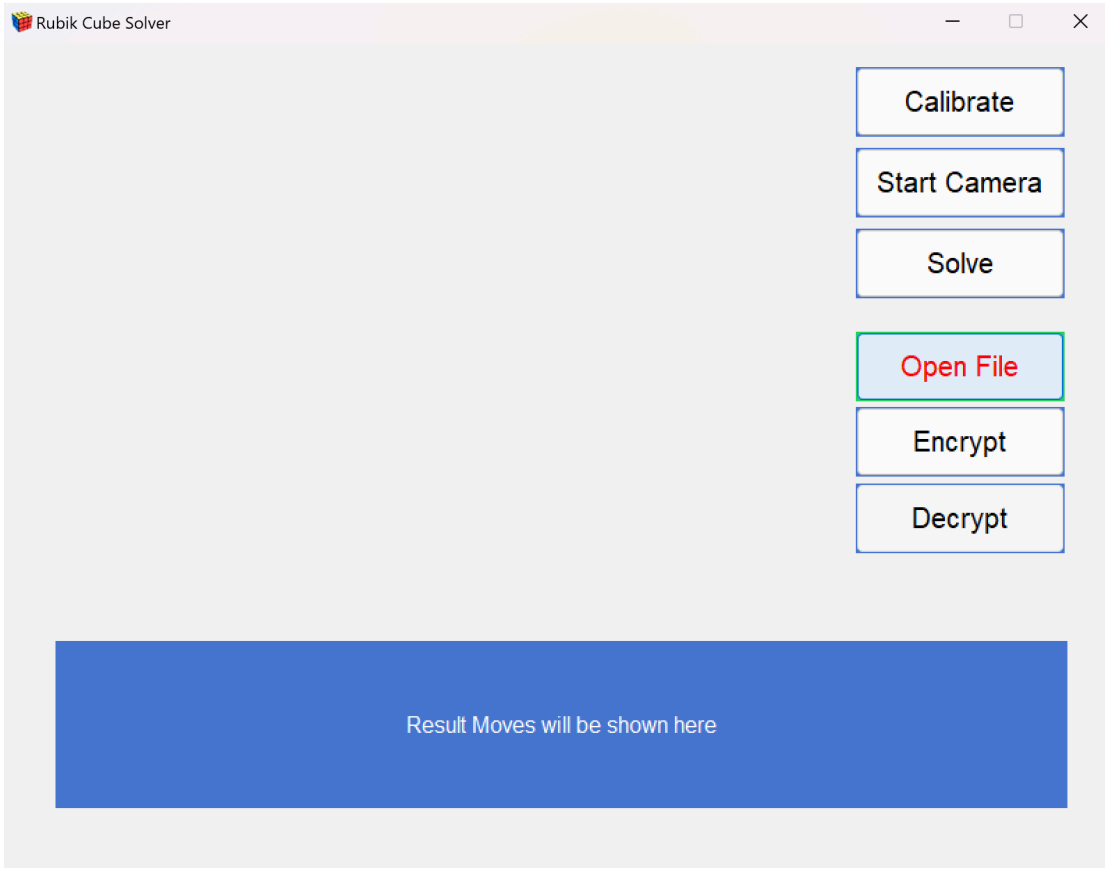

# SOLVER:

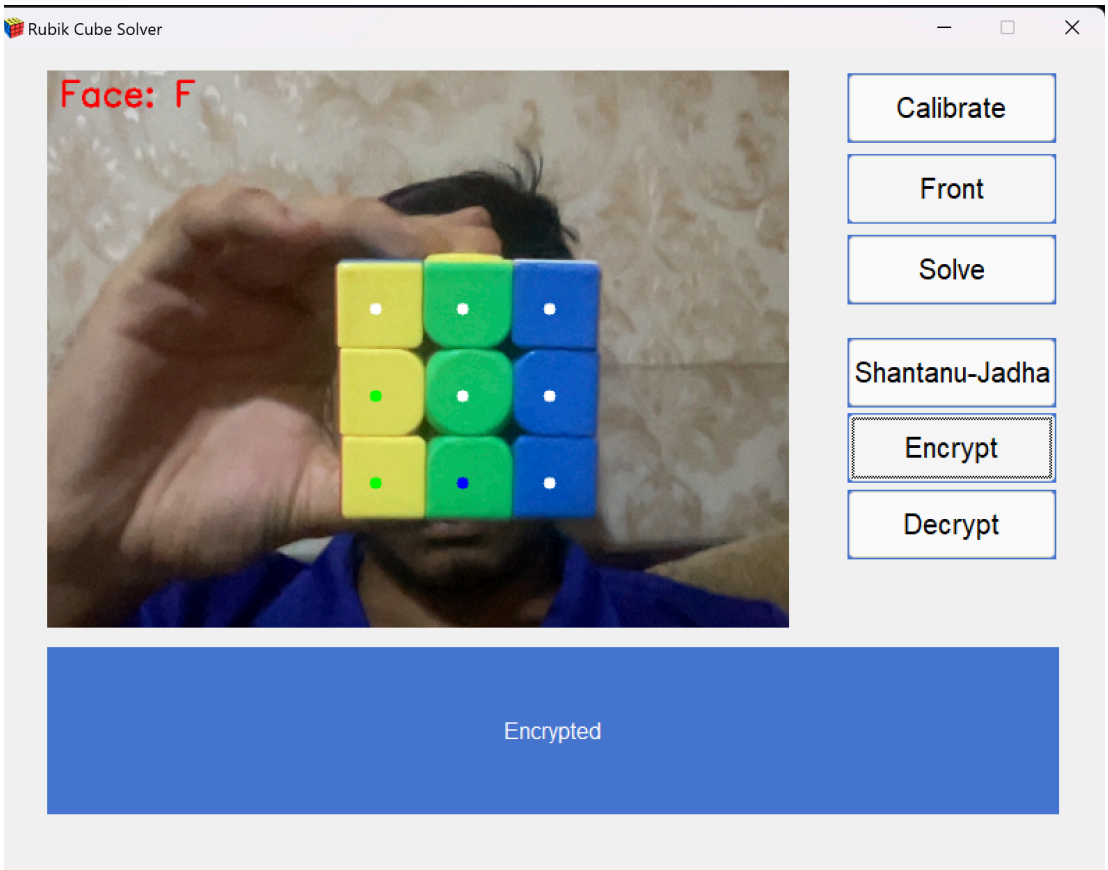


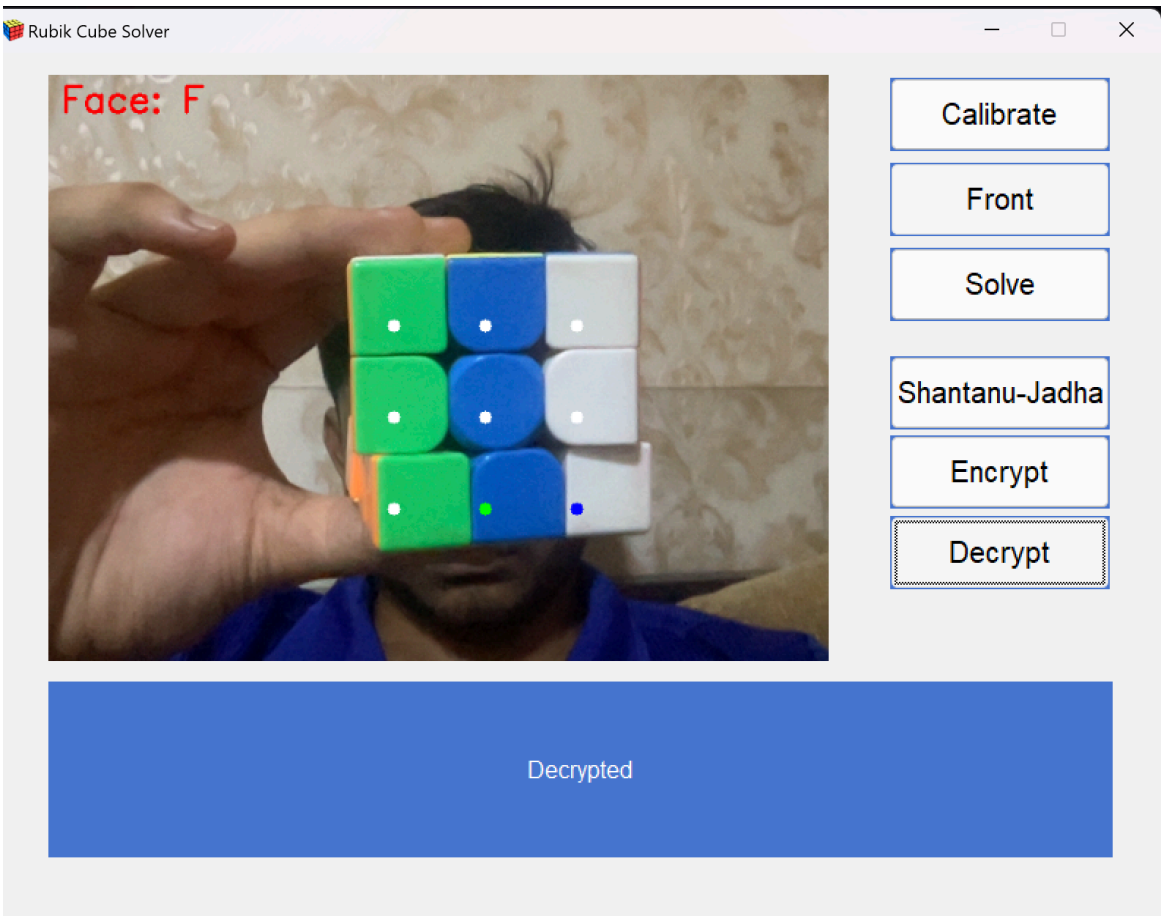

Rubik Cube Solver
Face: D
Calibrate
Down
Solve
Open File
Encrypt
Decrypt
Result Moves will be shown here


Recorded
Solving
Open File
Encrypt
Decrypt

# ENCRYPTION


Rubik Cube Solver
Calibrate
Start Camera
Solve
Open File
Encrypt
Decrypt
Result Moves will be shown here


Rubik Cube Solver
Face: F
Calibrate
Front
Solve
Shantanu-Jadha
Encrypt
Decrypt
Encrypted


Rubik Cube Solver
Face: F
Calibrate
Front
Solve
Shantanu-Jadha
Encrypt
Decrypt
Decrypted

# SYSTEM TESTING

# TESTING METHODOLOGY

Testing is a process of executing a program with the intent of finding errors. A good test case is one that has a high probability of finding an as yet undiscovered error.

A successful test is one that uncovers an as yet undiscovered error. If testing is conducted successfully, it will uncover error in the software and testing demonstrates that software functions appear to be working according to specification, that behavior and performance requirements appear to have been met. In addition, data collected as testing provide a good indication of software reliability and some indication of software quality as a whole. However, testing cannot show the absence of errors and defects, it can only show that errors and defects are present.

**Testing Methodology to be adopted:**

All testing should be traceable to customer requirements

- Test should be planned long before testing begins.
- Testing should begin in small scale and progress towards large scale
- Exhaustive testing is not possible.
- To be most effective testing should be conducted by independent third party.

**Testing Methods:**
Test must be designed with the highest likelihood to find possible errors in the system to avoid major problems before the system goes live. There are two methods to design the test cases.

**Verification and Validation:**
Verification refers to the set of activities that ensure, software correctly implements a specific function. Validation refers to different set of activities that ensures the software that has been built is traceable to customer requirements.

| Types of Test | Will Test be Preformed | Comments/Explanation | Software Component |
|---|---|---|---|
| Requirements Testing | YES | Needs to be done to cope up with changing environment | • Fluctuation in lighting condition<br>• Different shades of the color on thee Rubik cube |
| Unit | YES | Maximum number of defects is found. Each component of code was tested or analyzed accordingly not only to ensure the best quality of the developed software but also to make sure that code behaves in the same way as it was intended to. Unit testing was performed as and when the component was developed | • The color detection<br>• Calibration of colors<br>• Rubik cube solver giving proper algorithm<br>• Uploading file is smooth<br>• Encryption works<br>• Decryption works |
| Integration | YES | All the well-developed sub- system are integrated together and tested called as integration testing. | • Camera works for both the solver a encryption module<br>• Calibrated color works for both module |
| Performance | YES | Performance is the major criteria for evaluating any type of the system. It holds importance and is tested likewise. | • The color detection from the rubik cube went smoothly<br>• The speed of alogrithm thinking about the next move<br>• Decryption part for the correct password authentication |

# Test Environment:

**Software Items:**

- Windows 11
- Internet connection
- Python 3.8

**Hardware Items:**

- Personal Computer/Laptop
- Wireless connection or connecting cable
- Webcam

# Test Case:

| | |
|---|---|
| Test Id | T1 |
| Input | Calibrating Color |
| Output | Colors from the rubik cube got stored in BGR format in colorlabeler.py |
| Status | Pass |

| | |
|---|---|
| Test Id | T2 |
| Input | Rubik cube color capturing |
| Output | Colors from the rubik cube got correctly identified |
| Status | Pass |

| | |
|---|---|
| Test Id | T3 |
| Input | Solver working |
| Output | The solver giving the correct algorithm and working smoothly |
| Status | Pass |

| | |
|---|---|
| Test Id | T4 |
| Input | Encryption |
| Output | Encrypting the selected file |
| Status | Pass |

| | |
|---|---|
| Test Id | T5 |
| Input | Choosing the file |
| Output | File explorer got opened a the selected file got load into the application |
| Status | Pass |

| | |
|---|---|
| Test Id | T6 |
| Input | Decryption |
| Output | Decrypted the file when the correct password was given |
| Status | Pass |

## Limitation of the Solution:

- Sometimes the correct red and orange is indentified as the same.
- The Calibration module is not smooth.
- Sometimes the solver doesn't  catch the correct color due to the lighting condition

# COST AND BENEFIT ANALYSIS

# DEVELOPING A COST BENEFIT ANALYSIS

- Investigations of software development practices, processes & techniques frequently report separately on the costs & benefits of a phenomenon under study, but rarely adequately address the combined bottom-line implications.
- In particular, tensions between the quality & productivity effects are hard to reconcile, making objective, high-level insights elusive.
- The organization will obviously be able to gain benefits such a saving in operating cost, reduction in paperwork, better utilization of human resources & more image increasing goodwill.

**Initial Cost:**

- The initial cost of setting up the system will include the cost of hardware software (OS, add-on software, utilities) & labor (setup & maintenance).
- The same has to bear by the user.

**Running Cost:**

- Besides, the initial cost the long term cost will include the running cost for the system including the cost for human resources and other resources, cost of update/renewal of various related software.

# COST EVALUATION

The primary reason of cost & schedule estimation is to enable the client or developer to perform a cost benefit analysis & for project monitoring & control. The cost of a project is a function of many parameters. Foremost among them is the size of the project. Other factors that affect the cost are programmer ability, experience of the developers in the area, complexity of the project, & reliability requirements. It is also due to the requirements of software, hardware & human resources. Cost required for the project is to install the software & hardware requirements.

**Total Implementation Cost:**

-

Total implementation costs are the present value costs calculated over the length of the project. Because there might be overlapping in costs of implementing architectural & policy recommendation when some recommendations may be necessary to mitigate multiple categories of threats, total implementation cost are the sum of all present value costs of implementation minus any overlapping costs.

**Net Project Value:**

Net project value is the present value of saving form the total benefits of implementation recommendations minus total cost of implementing recommendations minus total cost of implementing recommendations. The higher the net, project value is the better.

**Formula:**

**Net Project Value (NV) = Total Benefits - Total Implementations Cost**

## Total System Value:

Net worth is the present value less the expected loss from the threat of no deduction from the fair va lue of the project. This entails that companies that don't add services still have to pay a dangerous a mount. If the class of the threat is reduced, its additional cost is applied, otherwise the initial cost is applied. The total cost of the system considers the situation where the cost of assets is high and the t otal cost of the system is low because the solution does not resolve the expensive threat.

## Costing:

The Total Costing for this project has been:

**TOTAL COST = TOTAL MANHOURS * COST PER HOUR**

= 380 * 120

**= 45,600**

# CONCLUSION AND FUTURE ENHANCEMENT

# CONCLUSION

Solving Rubik Cube can start out as curiosity for some and even end up as a hobby. It is quite interesting activity. However, learning to solve a Rubik's cube takes a lot of patience, it requires a lot of time but inevitably, it is truly rewarding. With help of our application, it will be  easy for one person to solve Rubik Cube. One can Learn how to solve and also be creative and productive.

It will help us to solve a Rubik cube with minimal number of steps. Person not knowing how to solve still can easily solve it with help of this application. It will show the shortest path with minimum number of steps to solve, so that it will be fast and efficient to solve. This will help us to solve the cube in less time.

Encrypting your files can give you an additional layer of security A  simple 3x3 rubik cube can be scrambled into 43 quintillion combination this amount of password creation generation makes it very hard to crack the password

With this application you can encrypt your file and use a the patterns created on the one of face of the rubik cube has a password  and the file woudnt be decrypted till the same pattern in used. This use of rubik cube has a password creates a physical key which can only be created by the owner of the files which will be a more secured that our normal password.

# LIMITATIONS

- The Rubik cube solver is only works for the 3x3 rubik cube an not for any other type of rubik cube
- The encryption password can only be set for one and only one face
- The stored file is cannot be decrypted after the applications is reopened because there is no database in the project where the password can be stored
- The color detection is a hit and miss due to the specific need of lighting condition and it need to be calibrated for every new cube.

# FUTURE ENHANCEMENTS

- To add solvers for different types of rubik cubes
- To make the encryption part for storing password even after the application is closed
- Allowed 6 faces us be the password
- To enhance the color capturing capability of the application so it can be used for any type of cube without calibrating.

# REFERENCES

## BIBLIOGRAPHY:

I have gathered information required to understand the project concept and the ideas used from various online resources:

Herbert Kociemba's : Model Used in the Project
Kkoomen Github : Understanding of computer vision used in the project

## REFERENCE:

Algorithm : http://kociemba.org/

Flowchart: https://app.zenflowchart.com/flowchart/wlhMe61vWIJxHcl3pmgU

Github: https://github.com/kkoomen