A Project Report on

"Puzzle Game"

Submitted in partial fulfillment of the requirement for
Degree in Bachelor of Engineering (Information Technology)

By
Krishna Saroj (5020155)
Apurv Mule (5020131)
Vivek R Dhanade(5020114)
Anthony Sayapogu(5020156)

Guided by:
Prof. Dhanashree Hadsul

Department of Information Technology
Fr. Conceicao Rodrigues Institute of Technology
Sector 9A, Vashi, Navi Mumbai – 400703

University of Mumbai
2021-2022

CERTIFICATE

This is to certify that the project
entitled

“Puzzle Game”

In partial fulfillment of degree of **B.E**. in **Information Technology** for term work of the project is approved.

Submitted By

Krishna Saroj
Apurv Mule
Vivek R Dhanade
Anthony Sayapogu

External Examiner **Internal Examiner**

Head of Department **Guide**

Principal

Date: **College Seal**

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Krishna Saroj (5020155)

Apurv Mule(5020131)

Vivek R Dhanade(5020114)

Anthony Sayapogu(5020156)

Date:

Abstract

A sliding tile puzzle is a combination puzzle that will challenge a player to slide pieces along certain routes to establish a certain end-configuration. It is also known as the 8 puzzle as it generally consists of 3 X 3 greed and one of the parts among the 9 is kept blank and is slided throughout the board to get the desired image.

The piece to be moved may consist of simple shapes, or they may be imprinted with colors, patterns, sections of a larger picture , numbers, or letters. And in our project we will be going with pictures and we will allow users to choose any image from their machine and start playing with it.

This puzzle will prohibit lifting any piece off the board. This property separates sliding puzzles from rearrangement puzzles. Hence, finding moves and the paths opened up by each move within the two-dimensional confines of the board are important parts of solving sliding tile puzzles.

The sliding puzzle is a classical puzzle game for players of all ages, enjoyed by children and adults. It is an easy and brain teasing game consisting in moving the pieces to the correct square in order to assemble the picture. The pieces can be moved to the square next to them, if it is not occupied by another piece.
Players will have to use their logic skills to complete this puzzling and addictive game. At the end, the empty square must be in bottom right corner.

Introduction

Sliding puzzle Game In Java :

A **sliding puzzle** is a combination puzzle that challenges a player to slide (frequently flat) pieces along certain routes (usually on a board) to establish a certain end-configuration. The pieces to be moved may consist of simple shapes, or they may be imprinted with colors, patterns, sections of a larger picture (like a jigsaw puzzle), numbers, or letters.

Sliding puzzles are essentially two-dimensional in nature, even if the sliding is facilitated by mechanically interlinked pieces (like partially encaged marbles) or three-dimensional tokens. As this example shows, some sliding puzzles are mechanical puzzles. However, the mechanical fixtures are usually not essential to these puzzles; the parts could as well be tokens on a flat board that are moved according to certain rules.

Unlike other tour puzzles, a sliding block puzzle prohibits lifting any piece off the board. This property separates sliding puzzles from rearrangement puzzles. Hence, finding moves and the paths opened up by each move within the two-dimensional confines of the board are important parts of solving sliding block puzzles.

The oldest type of sliding puzzle is the fifteen puzzle, invented by Noyes Chapman in 1880. Chapman's invention initiated a puzzle craze in the early 1880s. From the 1950s through the 1980s sliding puzzles employing letters to form words were very popular. These sorts of puzzles have several possible solutions, as may be seen from examples such as Ro-let(a letter-based fifteen puzzle)Scribe-io (4x8), and.Lingo

The best of them are deceptively simple in appearance. A child can understand the idea, and it might look like child's play to solve the puzzle. You don't have to learn any complicated rules. Yet the solution can be so complex that it seems impossible. Some solutions involve 100 moves or more. The puzzles demand logic, problem-solving and sequential thinking skills, combined with a dash of intuition and a healthy amount of patience.

Sliding puzzles started as actual mechanical devices, blocks of wood or plastic in a frame. But they were easy to translate into computer programs and to offer over the Internet. Hundreds of sliding puzzles are available online, and now you can play sliding puzzles on your smartphones and mobile devices. We have just created a small part of it as our project.

P.T.O

Literature Survey

The purpose of a literature review is to gain an understanding of the existing research and debates relevant to a particular topic or area of study, and to present that knowledge in the form of a written report.

Children at the Preschool age i.e. at 3-5 years old shows progressively motoric, cognitive, social and social skill development. In daily life, children must be given a stimulus to help the improvement and development of cognitive function at the early age. Playing at the early-age for children is very necessary. By playing, the learning process will be effective and promptly perceived when they play. One of the media of playing for children that function to stimulate cognitive development is puzzle game. Puzzle is a piece of thin paper comprising 2-3 pieces even 4-6 pieces that are made of wood or cardboard. One of the benefits of playing is good for development of children's cognitive side.

In this digital era and with the pandemic all around the world, we need a digital form of puzzle which is designed for children. We have puzzles for various age groups but not many for preschool children . Here we wanted to work out and brought up this project to build a puzzle game to learn the alphabets.

Related Work:

Java:

Java is an object-oriented, class-based, concurrent, secured and general-purpose computer-programming language. It is a widely used robust technology. It was developed by James Gosling and team of Sun Microsystems. It's a cross-platform language and programs are easily portable from one platform to another.

Package java.awt.image:

Provides classes for creating and modifying images. Images are processed using a streaming framework that involves an image producer, optional image filters, and an image consumer. This framework makes it possible to progressively render an image while it is being fetched and generated.

Moreover, the framework allows an application to discard the storage used by an image and to regenerate it at any time. This package provides a number of image producers, consumers, and filters that you can configure for your image processing needs.

Package java.awt.image.BufferedImage :

Java BufferedImage class is a subclass of Image class. It is used to handle and manipulate the image data. A BufferedImage is made of ColorModel of image data. All BufferedImage objects have an upper left corner coordinate of (0, 0).

Package javax.swing.ImageIcon :

Many Swing components, such as labels, buttons, and tabbed panes, can be decorated with an icon — a fixed-sized picture. An icon is an object that adheres to the **Icon** interface. Swing provides a particularly useful implementation of the Icon interface:**ImageIcon**, which paints an icon from a GIF, JPEG, or PNG image.

Here Image class is used to hold image resource while BufferedImage and ImageIcon are used to split image to multiple images.

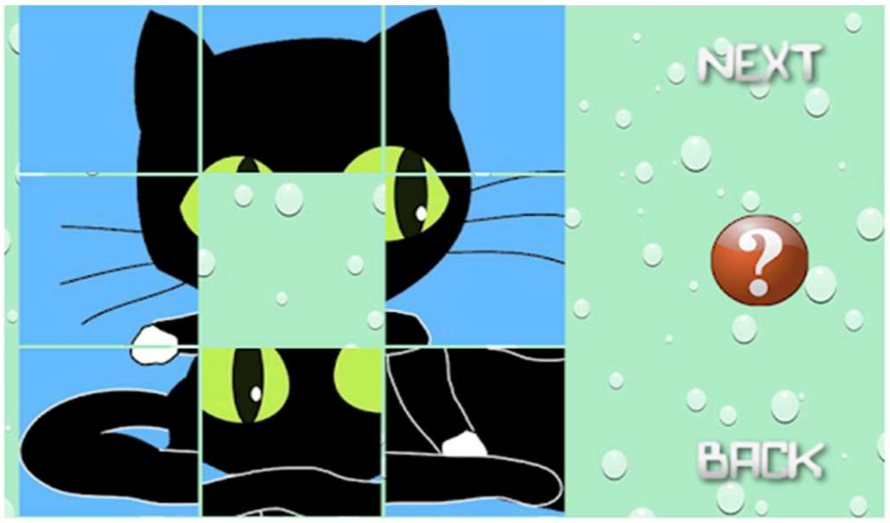
Existing Systems

- **Kids Slide Puzzle**

This is a great educational puzzle game for kids. Slide the puzzles to create the correct image. It contains a collection of beautiful colorful pictures suitable for toddlers and kids. Designed for ages between 2-9 years.

Children will see a set of tiles on a grid and will have to slide the puzzle pieces into the correct place to create the original image. Kids enjoy solving the puzzle and are delighted by the cute images. At the same time they learn and improve their observation and problem solving skills

$\longrightarrow$ GAME ICON



- **<u>Sliding Puzzle Cartoon & Animals</u>**

Use high resolution images of cartoons and animals.

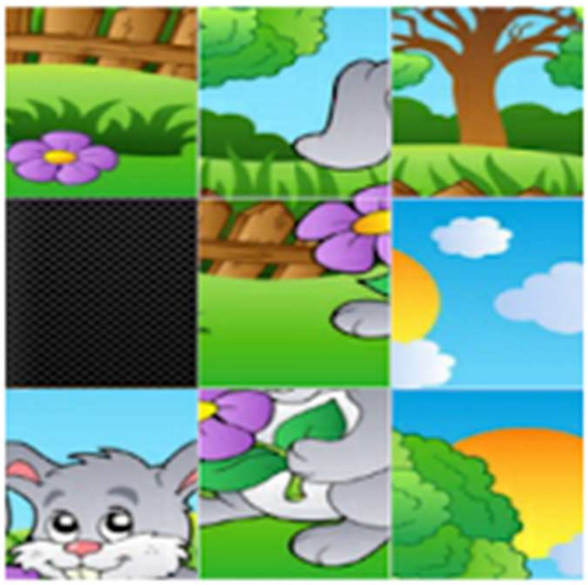
➢ How to play

> To solve the puzzle, you must move the blocks and place them in the correct order. To do this, you can move the photos one by one to an empty block until all the blocks are sorted in the correct order.

You can move one or more pieces at a time to fill an empty block. You may need to move already aligned pieces to be able to "jump" one of the following blocks. If you do, remember the route and reorder it back to its original location.

Using hints:

With the hint option, all blocks are displayed in order. This allows you to plan your move better.



Scope

As the execution of the game panel starts a window is generated which has four buttons at the bottom namely START, PAUSE, SHUFFLE and LOAD.The game starts when the user clicks the first start button .After clicking a default image that is already set in the program appears in a sorted way then the user can shuffle it by clicking the shuffle button at this point the timer in the game starts the user can click and slide through the pieces if its a kid and is unable to sort the image he can click a button provided at the top left corner which shows how the actual image looks so he will get an idea of how the arrangement needs to be proceeded if he or she is stuck .if the user has to wait for a while before his next move he can simply pause the game with the Pause button provided .and can again click on it to resume from where he left off .As the game starts with the default image a user can get bored with playing the same image so he can click the LOAD button which will direct the user towards the files section of his computer he can choose from there any suitable image he likes to create a puzzle .The extension of the image chosen should only be jpg or a png image .Once the user selects the desired image he can simply click ok and selected image forms on the game window then the user can shuffle it and start the game with the image he has chosen For closing the application the user needs to click on the close button at the top right corner .all the buttons are shown forward in output which will give a clear idea on the format of the game.

Problem Statement

To develop an educational puzzle game primarily focused on younger children which will help them develop and improve their physical, emotional and cognitive skills from an young age.

Proposed System

Slider Puzzle Game is a simple and entertaining puzzle game. In Slider Puzzle Game, a picture is torn down into smaller pieces of pictures and is scrambled from their correct position. One has to move the pieces to their correct position to complete the puzzle. A piece can only be moved to an empty position.

Slider Puzzle Game also tracks the time taken to complete the puzzle. The timer option helps creating a sense of competitiveness amongst the users which adds on to the beauty of the game. It helps user to rate and anlyse his/her performance taking into consideration how quickly or slowly the puzzle is getting solved.

Slider Puzzle Game also provides user the option to select any image from his/her device which can be used to play the puzzle game. Solving the same puzzles time and again could in a way, make it little boring and tedious for the user after some point & time. Slider Puzzle Game thus provides user this flexibility which helps maintaining users enthusiasm in the game.

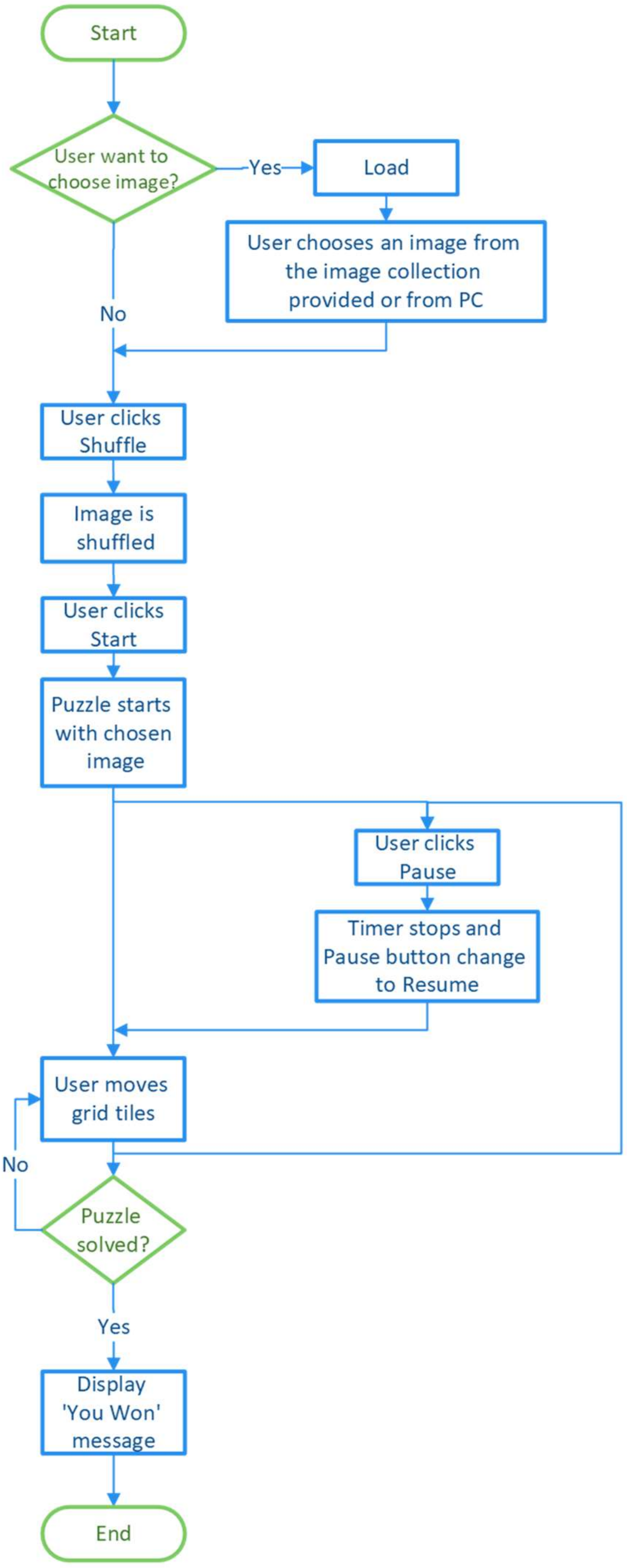
Proposed System(continued)

The game does not have any complicated rules which makes it even more user friendly and easy to understand. The puzzle solving basically demands logic, problem-solving and sequential thinking skills and last but not the least, a healthy amount of patience.
As the younger children will try solving these puzzles, automatically they will start developing these skills with time which will be immensely helpful for them in their near future. Thus anyone playing the game will be able to enjoy the game to the fullest.

Hardware and Software Requirements

- Memory: 4GB is preferred or else 2GB is
  also capable to run it.
- Graphics Card: Not required or normal cards like NVIDIA GeForce 7100 / nForce 630i.
- CPU: Intel Pentium 2.00GHz.or any dual core processor
  System .
- Size on disk: 100-400 MB.
- OS: Windows 7 SP1,Windows 10 x64 bit
- Software: Java 17 or higher

Flow Diagram


Start
User want to choose image?
Yes
Load
User chooses an image from the image collection provided or from PC
No
User clicks Shuffle
Image is shuffled
User clicks Start
Puzzle starts with chosen image
User clicks Pause
Timer stops and Pause button change to Resume
User moves grid tiles
No
Puzzle solved?
Yes
Display 'You Won' message
End

Program Code

1]Game.java

```
package zatackcoder;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.MouseEvent;
import java.awt.image.BufferedImage;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
public class Game extends javax.swing.JFrame implements ActionListener{
   public Game() {
      initComponents();
      gamePanelJP.setVisible(false);
   }
   @SuppressWarnings("unchecked")
   private void initComponents() {
      controlPanel = new javax.swing.JPanel();
      startStopJB = new javax.swing.JButton();
      playPauseJB = new javax.swing.JButton();
      shuffleJB = new javax.swing.JButton();
      loadJB = new javax.swing.JButton();
      gamePanelJP = new javax.swing.JPanel();
      timerPanel = new javax.swing.JPanel();
      closeJL = new javax.swing.JLabel();
      timeJL = new javax.swing.JTextField();
      helpJL = new javax.swing.JLabel();
      setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
      setBackground(new java.awt.Color(150, 50, 255));
      setIconImage(new ImageIcon(getClass().getResource("/img/logo.png")).getImage());
      setLocationByPlatform(true);
      setUndecorated(true);
      getContentPane().setLayout(new java.awt.BorderLayout());
      controlPanel.setBackground(new java.awt.Color(204, 204, 255));
      controlPanel.setFocusable(false);
      controlPanel.setOpaque(false);
      controlPanel.setPreferredSize(new java.awt.Dimension(300, 40));
      controlPanel.setLayout(new java.awt.GridLayout(1, 3));
```

```
startStopJB.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
        startStopJB.setForeground(new java.awt.Color(0, 153, 153));
        startStopJB.setText("START");
        startStopJB.setToolTipText("Start or Stop");
        startStopJB.setBorder(null);
        startStopJB.setBorderPainted(false);
        startStopJB.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        startStopJB.setFocusable(false);
        startStopJB.setMaximumSize(new java.awt.Dimension(85, 31));
        startStopJB.setMinimumSize(new java.awt.Dimension(85, 31));
        startStopJB.setPreferredSize(new java.awt.Dimension(85, 31));
        startStopJB.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                startStopJBActionPerformed(evt);
            }
        });
        controlPanel.add(startStopJB);
        playPauseJB.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
        playPauseJB.setForeground(new java.awt.Color(0, 153, 153));
        playPauseJB.setText("PLAY");
        playPauseJB.setToolTipText("Play or Pause");
        playPauseJB.setBorder(null);
        playPauseJB.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        playPauseJB.setFocusable(false);
        playPauseJB.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                playPauseJBActionPerformed(evt);
            }
        });
        controlPanel.add(playPauseJB);
        shuffleJB.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
        shuffleJB.setForeground(new java.awt.Color(0, 153, 153));
        shuffleJB.setText("SHUFFLE");
        shuffleJB.setToolTipText("Shuffle Tiles");
        shuffleJB.setBorder(null);
        shuffleJB.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        shuffleJB.setFocusable(false);
        shuffleJB.setOpaque(false);
        shuffleJB.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                shuffleJBActionPerformed(evt);
```

```
 }
});
controlPanel.add(shuffleJB);
loadJB.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
loadJB.setForeground(new java.awt.Color(0, 153, 153));
loadJB.setText("SELECT");
loadJB.setToolTipText("Load Image");
loadJB.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
loadJB.setFocusable(false);
loadJB.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      loadJBActionPerformed(evt);
   }
});
controlPanel.add(loadJB);
getContentPane().add(controlPanel, java.awt.BorderLayout.SOUTH);
gamePanelJP.setBackground(new java.awt.Color(204, 204, 255));
gamePanelJP.setFocusable(false);
gamePanelJP.setMaximumSize(new java.awt.Dimension(240, 200));
gamePanelJP.setMinimumSize(new java.awt.Dimension(240, 200));
gamePanelJP.setPreferredSize(new java.awt.Dimension(240, 200));
gamePanelJP.setRequestFocusEnabled(false);
gamePanelJP.setLayout(new java.awt.GridLayout(4, 4));
getContentPane().add(gamePanelJP, java.awt.BorderLayout.CENTER);
timerPanel.setBackground(new java.awt.Color(204, 204, 255));
timerPanel.setFocusCycleRoot(true);
timerPanel.setPreferredSize(new java.awt.Dimension(300, 25));
timerPanel.setLayout(new java.awt.BorderLayout());
closeJL.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
closeJL.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
closeJL.setText("X");
closeJL.setToolTipText("Close");
closeJL.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
closeJL.setFocusable(false);
closeJL.setMaximumSize(new java.awt.Dimension(10, 10));
closeJL.setMinimumSize(new java.awt.Dimension(20, 20));
closeJL.setPreferredSize(new java.awt.Dimension(25, 25));

closeJL.addMouseListener(new java.awt.event.MouseAdapter() {
   public void mouseClicked(java.awt.event.MouseEvent evt) {
      closeJLMouseClicked(evt);
   }
});
timerPanel.add(closeJL, java.awt.BorderLayout.EAST);
```

```
    timeJL.setEditable(false);
    timeJL.setBackground(new java.awt.Color(0, 102, 102));
    timeJL.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N
    timeJL.setForeground(new java.awt.Color(255, 255, 255));
    timeJL.setHorizontalAlignment(javax.swing.JTextField.CENTER);
    timeJL.setText("0:0");
    timeJL.setToolTipText("");
    timeJL.setBorder(null);
    timeJL.setCursor(new java.awt.Cursor(java.awt.Cursor.MOVE_CURSOR));
    timeJL.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
      public void mouseDragged(java.awt.event.MouseEvent evt) {
        timeJLMouseDragged(evt);
      }
    });
    timeJL.addKeyListener(new java.awt.event.KeyAdapter() {
      public void keyPressed(java.awt.event.KeyEvent evt) {
        timeJLKeyPressed(evt);
      }
    });
    timerPanel.add(timeJL, java.awt.BorderLayout.CENTER);
    helpJL.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
    helpJL.setForeground(new java.awt.Color(0, 153, 153));
    helpJL.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
    helpJL.setText("*");
    helpJL.setToolTipText("Left Click to see Image and Right Click to Hide Controls");
    helpJL.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
    helpJL.setFocusable(false);
    helpJL.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
    helpJL.setPreferredSize(new java.awt.Dimension(25, 25));
    helpJL.addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseClicked(java.awt.event.MouseEvent evt) {
        helpJLMouseClicked(evt);
      }
    });
    timerPanel.add(helpJL, java.awt.BorderLayout.LINE_START);
    getContentPane().add(timerPanel, java.awt.BorderLayout.PAGE_START);
    pack();
  }// </editor-fold>//GEN-END:initComponents
    private void initGame(){
    buttons=new JButton[15];
    img=new ImageIcon(getClass().getResource("/img/appl2.jpg")).getImage();
    icon=splitImage(img);
    for(int i=0;i<buttons.length;i++){
        buttons[i]=new JButton();
        buttons[i].setFocusable(false);
```

```
buttons[i].setSize(60, 50);
            buttons[i].addActionListener(this);
            buttons[i].setIcon(icon[i]);
            gamePanelJP.add(buttons[i]);
      }
      swapB=new JButton();
      swapB.setIcon(new ImageIcon(getClass().getResource("/img/blank.png")));
      gamePanelJP.add(swapB);
      gamePanelJP.setVisible(true);
      gamePanelJP.updateUI();
      }
      private void changeImage() {
         gamePanelJP.removeAll();
         for (int i = 0; i < buttons.length; i++) {
            buttons[i].setIcon(icon[i]);
            gamePanelJP.add(buttons[i]);
         }
         swapB = new JButton();
         swapB.setIcon(new ImageIcon(getClass().getResource("/img/blank.png")));
         gamePanelJP.add(swapB);
         gamePanelJP.updateUI();
      }
      private void shuffleJBActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
   FIRST:event_shuffleJBActionPerformed
         shuffle();
      }//GEN-LAST:event_shuffleJBActionPerformed
      private void closeJLMouseClicked(java.awt.event.MouseEvent evt) {//GEN-
   FIRST:event_closeJLMouseClicked
       if(evt.getButton()==MouseEvent.BUTTON1){
         this.dispose();
         System.exit(1);
       }
      }//GEN-LAST:event_closeJLMouseClicked
      private void timeJLKeyPressed(java.awt.event.KeyEvent evt) {//GEN-
   FIRST:event_timeJLKeyPressed
         if(evt.getKeyCode()==KeyEvent.VK_LEFT){
            this.setLocation(this.getLocation().x-10, this.getLocation().y);
         }
         if(evt.getKeyCode()==KeyEvent.VK_RIGHT){
            this.setLocation(this.getLocation().x+10, this.getLocation().y);
         }
         if(evt.getKeyCode()==KeyEvent.VK_UP){
            this.setLocation(this.getLocation().x, this.getLocation().y-10);
         }
         if(evt.getKeyCode()==KeyEvent.VK_DOWN){
            this.setLocation(this.getLocation().x, this.getLocation().y+10);
         }
      }//GEN-LAST:event_timeJLKeyPressed
```

```
    private void loadJBActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_loadJBActionPerformed
        JFileChooser filechooser=new JFileChooser();
        int response=filechooser.showOpenDialog(startStopJB);
        if(response==JFileChooser.APPROVE_OPTION){
            img=new ImageIcon(filechooser.getSelectedFile().toString()).getImage();
            icon=splitImage(img);
            changeImage();
        }
    }//GEN-LAST:event_loadJBActionPerformed
    private void startStopJBActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_startStopJBActionPerformed
      if(startStopJB.getText().equals("START")){
       timer=new Timer(timeJL);
       timer.start();
       playPauseJB.setText("PAUSE");
       startStopJB.setText("STOP");
       gamePanelJP.removeAll();
       initGame();
      }else{
          timer.stopT();
          startStopJB.setText("START");
          playPauseJB.setText("PLAY");
          timeJL.setText("0:0");
          gamePanelJP.setVisible(false);
      }
       //shuffle();
    }//GEN-LAST:event_startStopJBActionPerformed
    private void playPauseJBActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_playPauseJBActionPerformed
        if (playPauseJB.getText().equals("PAUSE")) {
            playPauseJB.setText("PLAY");
            timer.pauseT();
        } else {
            playPauseJB.setText("PAUSE");
            timer.resumeT();
        }
    }//GEN-LAST:event_playPauseJBActionPerformed
    private void helpJLMouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_helpJLMouseClicked
if(evt.getButton()==MouseEvent.BUTTON1){
           new Help(img,this.getX()+this.getWidth(),this.getY()).setVisible(true);
       }
       if(evt.getButton()==MouseEvent.BUTTON3){
```

```
 if(controlPanel.isVisible()){
          this.setSize(this.getWidth(), this.getHeight()-40);
          controlPanel.setVisible(false);
          }else{
            this.setSize(this.getWidth(), this.getHeight()+40);
            controlPanel.setVisible(true);
 }
      }
    }//GEN-LAST:event_helpJLMouseClicked

    private void timeJLMouseDragged(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_timeJLMouseDragged
       setLocation(evt.getXOnScreen()-175,evt.getYOnScreen());
    }//GEN-LAST:event_timeJLMouseDragged

    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Game.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Game.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (IllegalAccessException ex) {
```

```
java.util.logging.Logger.getLogger(Game.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Game.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        }
        //</editor-fold>
        /* Create and display the form */

        java.awt.EventQueue.invokeLater(new Runnable() {
            @Override
            public void run() {
                new Game().setVisible(true);
            }
        });
    }

    public void shuffle(){
        gamePanelJP.removeAll();
         for(int i=0;i<100;i++){
            int x=(int)(Math.random()*15);
            gamePanelJP.add(buttons[x]);
        }
         gamePanelJP.add(swapB);
         gamePanelJP.updateUI();
    }

    public void check() {
        boolean won = false;
        for (int i = 0; i < 15; i++) {
            if (buttons[i] == gamePanelJP.getComponent(i)) {
                won = true;
            } else {
                won = false;
                break;
            }
        }
        if (won) {
            timer.stopT();
            swapB.setIcon(icon[15]);
            JOptionPane.showMessageDialog(gamePanelJP,"You Won! "+timeJL.getText());
            startStopJB.setText("START");
            playPauseJB.setText("PLAY");
        }
    }
```

```
 public void move(JButton btn){

if(swapB==gamePanelJP.getComponentAt(btn.getX()+btn.getWidth(),btn.getY())){
          int swapIndex=gamePanelJP.getComponentZOrder(swapB);
          int btnIndex=gamePanelJP.getComponentZOrder(btn);
          gamePanelJP.add(btn,swapIndex);
          gamePanelJP.add(swapB,btnIndex);
 gamePanelJP.updateUI();
       }
       if(swapB==gamePanelJP.getComponentAt(btn.getX()-
btn.getWidth(),btn.getY())){
          int swapIndex=gamePanelJP.getComponentZOrder(swapB);
          int btnIndex=gamePanelJP.getComponentZOrder(btn);
          gamePanelJP.add(btn,swapIndex);
          gamePanelJP.add(swapB,btnIndex);
          gamePanelJP.updateUI();
       }

if(swapB==gamePanelJP.getComponentAt(btn.getX(),btn.getY()+btn.getHeight())){
         int swapIndex=gamePanelJP.getComponentZOrder(swapB);
          int btnIndex=gamePanelJP.getComponentZOrder(btn);
          gamePanelJP.add(btn,swapIndex);
          gamePanelJP.add(swapB,btnIndex);
          gamePanelJP.updateUI();
       }
       if(swapB==gamePanelJP.getComponentAt(btn.getX(),btn.getY()-
btn.getHeight())){
          int swapIndex=gamePanelJP.getComponentZOrder(swapB);
          int btnIndex=gamePanelJP.getComponentZOrder(btn);
          gamePanelJP.add(btn,swapIndex);
          gamePanelJP.add(swapB,btnIndex);
          gamePanelJP.updateUI();

       }
   }

   public ImageIcon[] splitImage(Image img){
       ImageIcon[] splittedIcons = new ImageIcon[16];
          BufferedImage bi=new BufferedImage(img.getWidth(null),
img.getHeight(null), BufferedImage.TYPE_INT_RGB);
          Graphics g=bi.createGraphics();
          g.drawImage(img, 0, 0, null);
          int width = bi.getWidth();
          int height = bi.getHeight();
          int pos = 0;
for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
               img = bi.getSubimage(j * (width / 4), i * (height / 4), width / 4, height /
4).getScaledInstance(85, 55, 2);
               splittedIcons[pos] = new ImageIcon();
               splittedIcons[pos].setImage(img);
               pos++;
            }
         }
 return splittedIcons;
   }
   private javax.swing.JButton swapB;
   private javax.swing.JButton[] buttons;
   private javax.swing.ImageIcon[] icon;
   private java.awt.Image img;
   private Timer timer;
   // Variables declaration - do not modify//GEN-BEGIN:variables
   private javax.swing.JLabel closeJL;
   private javax.swing.JPanel controlPanel;
   private javax.swing.JPanel gamePanelJP;
   private javax.swing.JLabel helpJL;
   private javax.swing.JButton loadJB;
   private javax.swing.JButton playPauseJB;
   private javax.swing.JButton shuffleJB;
   private javax.swing.JButton startStopJB;
   private javax.swing.JTextField timeJL;
   private javax.swing.JPanel timerPanel;
   // End of variables declaration//GEN-END:variables
   @Override
   public void actionPerformed(ActionEvent e) {
      move((JButton)e.getSource());
      check();
   }
}
```

P.T.O

2]Help.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package zatackcoder;
import java.awt.Image;
import java.awt.image.BufferedImage;
import javax.swing.ImageIcon;
public class Help extends javax.swing.JFrame {
    public Help(Image img,intx,int y) {
initComponents();
setLabelIcon(img);
this.setLocation(x, y);
    }
    private void setLabelIcon(Image img){
BufferedImage bi=new BufferedImage(img.getWidth(null), img.getHeight(null),
BufferedImage.TYPE_INT_RGB);
bi.createGraphics().drawImage(img, 0, 0, null);
img=bi.getScaledInstance(jLabel1.getWidth(),jLabel1.getHeight(), 1);
        jLabel1.setIcon(new ImageIcon(img));
    }
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {
        jLabel1 = new javax.swing.JLabel();
        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
setAlwaysOnTop(true);
setIconImage(new ImageIcon(getClass().getResource("/img/logo.png")).getImage());
setLocationByPlatform(true);
setUndecorated(true);
        jLabel1.setPreferredSize(new java.awt.Dimension(240, 200));
        jLabel1.addMouseListener(new java.awt.event.MouseAdapter() {
 public void mouseClicked(java.awt.event.MouseEventevt) {
```

```
 jLabel1MouseClicked(evt);
            }
        });
getContentPane().add(jLabel1, java.awt.BorderLayout.CENTER);
 pack();
    }// </editor-fold>//GEN-END:initComponents

    private void jLabel1MouseClicked(java.awt.event.MouseEventevt)
{//GEN-FIRST:event_jLabel1MouseClicked
this.dispose();
    }//GEN-LAST:event_jLabel1MouseClicked

    /**
     * @param args the command line arguments
     */
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JLabel jLabel1;
    // End of variables declaration//GEN-END:variables
}
```

P.T.O.

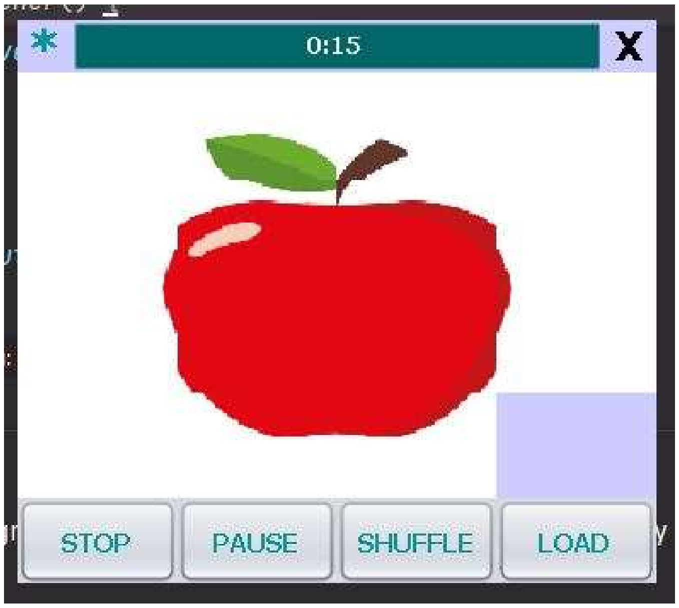
3]Timer.java

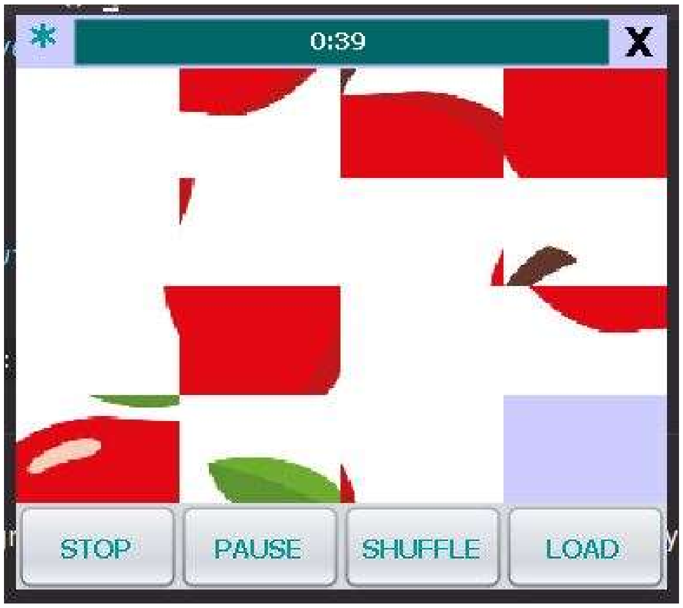
```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package zatackcoder;
import javax.swing.JTextField;
/**
 *
 * @author Rajesh Kumar Sahanee
 */
public class Timer extends Thread{
boolean  pause;
boolean  exit;
JTextFieldtf;
int min=0,sec=0;
   public Timer(JTextFieldtf){
this.tf=tf;
   }
   @Override
   public void run(){
      min=0;sec=0;
      while (!exit) {
        if(!pause){
timeLoop();
        }else{

        }
      }
   }

   public void timeLoop(){
            try {
tf.setText("" + min + ":" + sec);
         if (sec == 59) {
            min += 1;
            sec = 0;
         }
Thread.sleep(1000);
         sec++;
 } catch (InterruptedException ex) {
System.out.println(ex);
           }
    }
   public synchronized void resumeT(){
       pause=false;
   }
   public synchronized void pauseT(){
       pause=true;
   }
public synchronized void stopT(){
       exit=true;
   }
}
```

P.T.O.

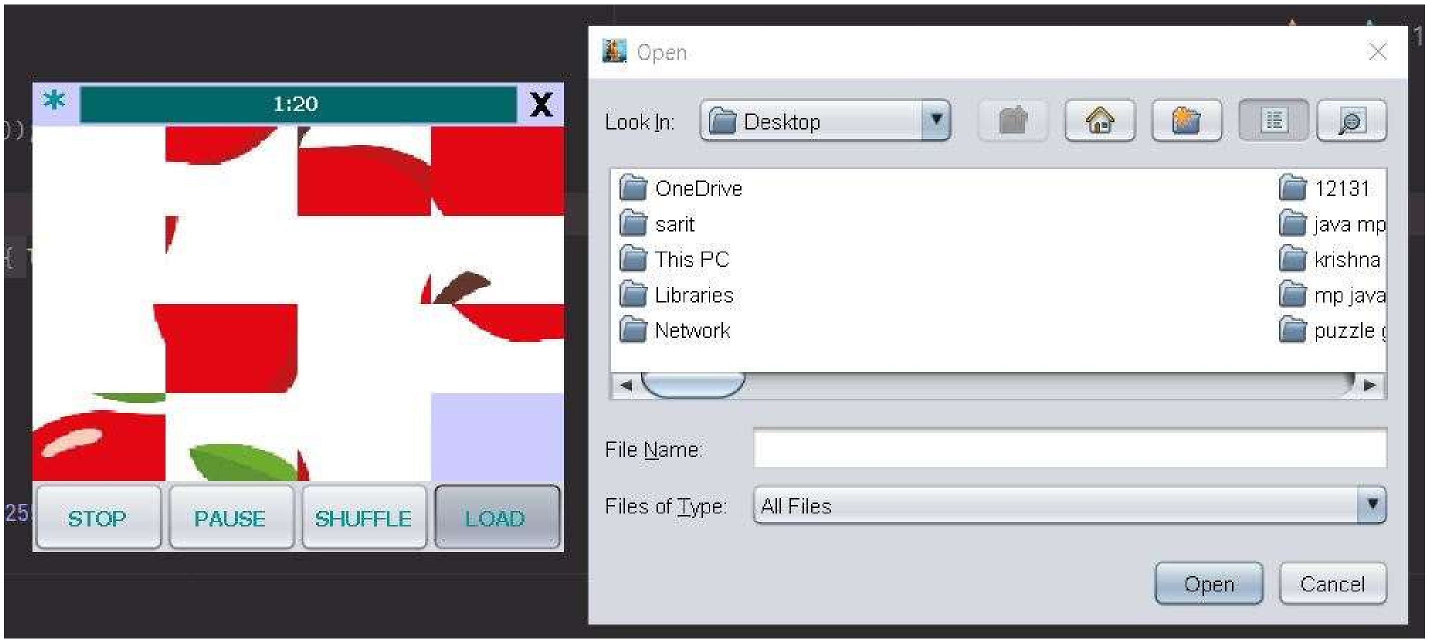
Output

This is the default image that will be shown to the user and they just need to click the Shuffle button to rearrang e the slides and start playing with the image.

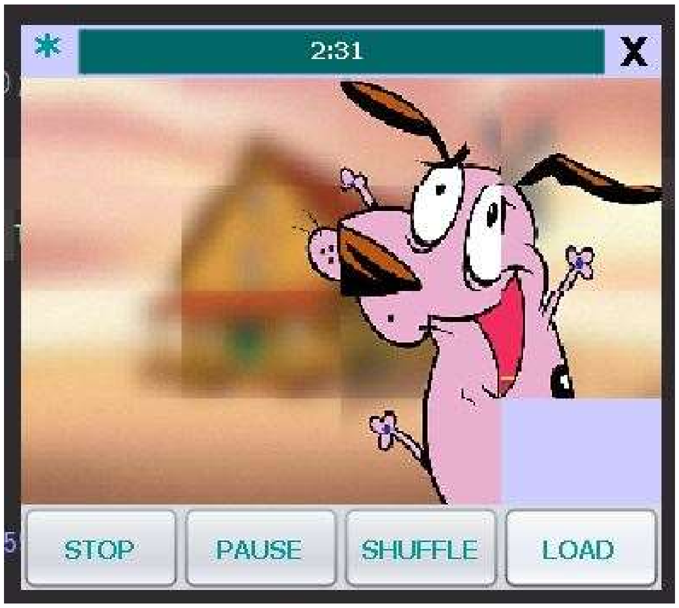


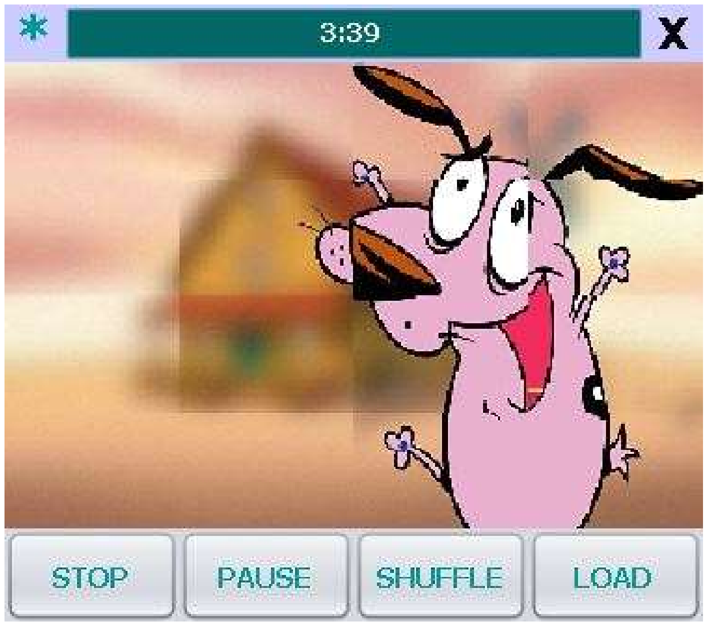


We even provide the option for choosing any image from the users device and for that they just have to click the load button and choose the desired image.

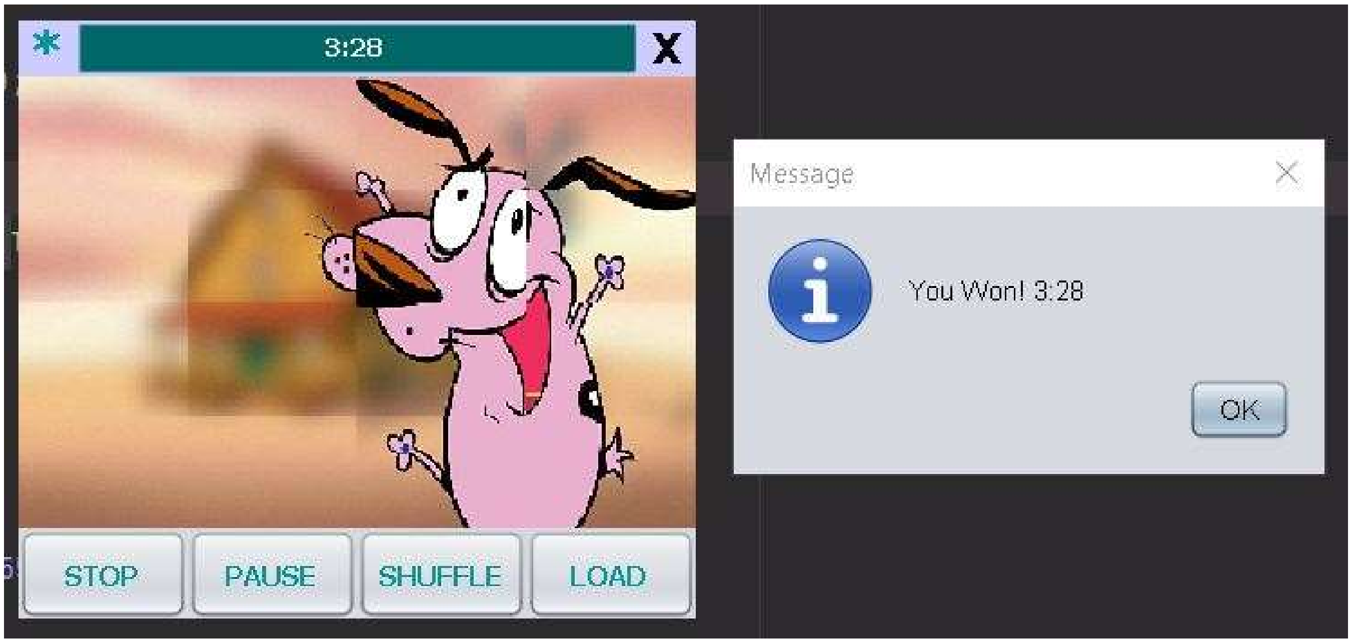


After selecting the desired image, user starts solving the puzzle.





Once you have completed the puzzle, you will be shown the final message and will also show your time taken to solve the puzzle.



You Won! 3:28

Conclusion

We are proposing a digital education based game which will be helpful in developing the logical thinking at younger age and will focus on problem solving ability of the user. Puzzles are great **for helping young brain develop and grow**. That's because the brain looks for patterns in our world and puzzles are a true patterning activity. Patterning is also the foundation of reading, math and logic skills. This game doesn’t demand for higher specifications so it’s easier to recommend it.

Thank You