**PEMBUATAN APLIKASI IMAGE PROCESSING WATERMARK MENGGUNAKAN NETBEANS**

****

**OLEH :**

|  |  |  |
| --- | --- | --- |
| **Nama** | **:** | **Melyaty Rante Kala (202278)**  **Raga Adi Kuncoro (202283)**  **Welem Iswan Fransiscus (202308)**  **Ahmad Abroar (202309)**  **Nades Kola Samperuru (202307)** |
| **Kelas** | **:** | **5TKKO-H** |
| **Dosen** | **:** | **Nurdin S.Kom., MT.** |

PROGRAM STUDI

TEKNIK INFORMATIKA

UNIVERSITAS DIPA MAKASSAR

**2022/2023**

1. **Pendahuluan**

Watermarking adalah salah satu bentuk atau aplikasi dari ilmu steganografi, yaitu teknik menyembunyikan suatu data pada data yang lain. Watermarking merupakan proses penambahan kode secara permanen ke dalam citra digital yang ingin dilindungi hak ciptanya dengan tidak merusak citra aslinya dan tahan terhadap serangan. Perbedaan antara steganografi dan watermarking adalah jika pada steganografi media penampung yang disisipkan kode rahasia tidak berarti apa-apa, maka pada watermarking justru media penampung yang dilindungi kepemilikannya.

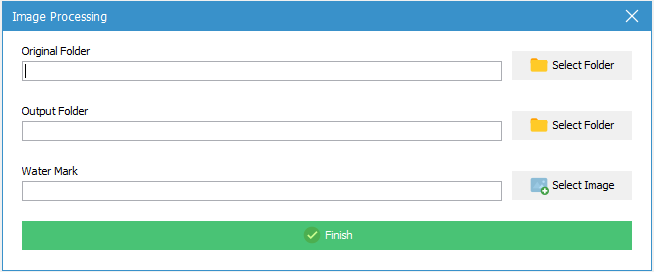
Watermarking merupakan suatu teknik penyembunyian data atau informasi rahasia ke dalam suatu data lainnya untuk ditumpangi, tetapi tidak disadari kehadirannya oleh indra manusia, yaitu indra penglihatan dan indra pendengaran. Selain itu, data yang sudah terwatermark harus tahan (robust) terhadap serangan-serangan, baik secara sengaja maupun tidak sengaja untuk menghilangkan data watermark yang terdapat di dalamnya.

Definisi lain dari watermarking adalah data tersembunyi yang ditambahkan pada sinyal pelindung (cover signal), sedemikian rupa sehingga penambahan tersebut tidak terlihat. Watermark bisa berupa kode yang membawa informasi mengenai pemilik hak cipta, pencipta, pembeli yang sah dan segala sesuatu yang diperlukan untuk menangani hak kepemilikan media digital. Watermark haruslah tidak mengubah isi media kecuali sedikit atau perubahan tersebut tidaklah tampak atau kurang begitu tampak bagi indera manusia.

1. Source Code

|  |
| --- |
| package com.abner.main;  import java.awt.AlphaComposite;  import java.awt.Desktop;  import java.awt.Graphics2D;  import java.awt.Image;  import java.awt.image.BufferedImage;  import java.io.File;  import java.io.IOException;  import java.util.concurrent.ThreadLocalRandom;  import javax.imageio.ImageIO;  import javax.swing.JOptionPane;  public class WaterMark {  public void showFiles(File file, String watermark, String output\_image) {  File aFile[] = file.listFiles();  int i = 0;  for (int j = aFile.length; i < j; i++) {  File files = aFile[i];  File originalFile = new File(file + "\\" + files.getName());  System.out.println(files.getName());  File mark = new File(watermark);  //Output file  int min = 1000, max = 9999;  int randomNum = ThreadLocalRandom.current().nextInt(min, max + 1);  String filename = "IMG-" + randomNum;  File outputFile = new File(output\_image + "\\" + filename + ".jpg");  addMark(mark, "png", originalFile, outputFile);  }  }  public void addMark(File mark, String type, File file, File dest) {  try {  BufferedImage image = ImageIO.read(file);  BufferedImage overlay = resize(ImageIO.read(mark), 150, 150);  int imageType = "png".equalsIgnoreCase(type) ? BufferedImage.TYPE\_INT\_ARGB : BufferedImage.TYPE\_INT\_RGB;  BufferedImage watermarked = new BufferedImage(image.getWidth(), image.getHeight(), imageType);  Graphics2D w = (Graphics2D) watermarked.getGraphics();  w.drawImage(image, 0, 0, null);  AlphaComposite alphaChannel = AlphaComposite.getInstance(AlphaComposite.SRC\_OVER, 1.0f);  w.setComposite(alphaChannel);  //position watermark  int centerX = image.getWidth() / 2;  int centerY = image.getHeight() / 2;  w.drawImage(overlay, centerX, centerY, null);  ImageIO.write(watermarked, type, dest);  w.dispose();  } catch (IOException e) {  e.printStackTrace();  }  }  private BufferedImage resize(BufferedImage img, int height, int width) {  Image tmp = img.getScaledInstance(width, height, Image.SCALE\_SMOOTH);  BufferedImage resized = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_ARGB);  Graphics2D g2d = resized.createGraphics();  g2d.drawImage(tmp, 0, 0, null);  g2d.dispose();  return resized;  }  } |

1. Screenshot Aplikasi



1. Hasil

* Watermark



* Original image



* Output

