GRAFIKA KOMPUTER KELAS J

KELOMPOK 5

Nama Kelompok :

145150207111067 Robihamanto

145150201111066 Andre Rizal Sinaga

145150201111068 Bossarito Putro

145150201111110 Rendy Mardiansyah

Modelling Labirin Coban Rondo Batu, Jawa Timur

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS BRAWIJAYA**

**2016**

Proposal Grafika Komputer 2016

DAFTAR ISI

Contents

[LATAR BELAKANG 28](#_Toc452585837)

[Tujuan dan Manfaat 28](#_Toc452585838)

[Batasan Masalah 29](#_Toc452585839)

[Tinjauan pusataka 1 30](#_Toc452585840)

[Tinjauan pusataka 2 30](#_Toc452585841)

[Deskripsi Karya 31](#_Toc452585842)

[Objek - Objek Dalam Karya 31](#_Toc452585843)

[Penerapan Modul 35](#_Toc452585844)

[Interaksi User 35](#_Toc452585845)

[Biografi Kelompok 37](#_Toc452585846)

[Pembagian Kerja 41](#_Toc452585847)

DAFTAR GAMBAR

[Gambar 1. 1 Labirin Coban rondo (dari atas) 31](#_Toc452585742)

[Gambar1. 2 Labirin Coban rondo dari atas samping 32](#_Toc452585743)

[Gambar1. 3 Implementasi Labirin Coban Rondo (dari atas) 32](#_Toc452585744)

[Gambar1.4 Implementasi Labirin Coban Rondo (Dari depan) 33](#_Toc452585745)

[Gambar 1.5 Pohon 33](#_Toc452585746)

[Gambar 1.6 Implementasi Bagian-Bagian Labirin 34](#_Toc452585747)

[Gambar 1.7 Air Mancur Coban rondo 34](file:///D:\BRAWIJAYA%20UNIVERSITY\SEMESTER%204\Grafika%20Komputer\Project%20Akhir\Grafkom_J_5_ProposalFP.docx#_Toc452585748)

DAFTAR TABEL

[Table 1 Atribut 5](#_Toc454302974)

[Table 2 Inisialisasi 7](#_Toc454302975)

[Table 3 Process Mouse 7](#_Toc454302976)

[Table 4 Process Keyboard 8](#_Toc454302977)

[Table 5 Render 12](#_Toc454302978)

[Table 6 Create Wall 14](#_Toc454302979)

[Table 7 Draw Wall 17](#_Toc454302980)

[Table 8 Draw Skybox 19](#_Toc454302981)

[Table 9 Draw Tree 21](#_Toc454302982)

[Table 10 Enable Night 22](#_Toc454302983)

[Table 11 Disable Night 22](#_Toc454302984)

[Table 12 load Texture 23](#_Toc454302985)

[Table 13 Destroy 24](#_Toc454302986)

[Table 14 Update FPS 24](#_Toc454302987)

[Table 15 get Delta 24](#_Toc454302988)

[Table 16 get Time 25](#_Toc454302989)

[Table 17 Create 25](#_Toc454302990)

[Table 18 initGL 25](#_Toc454302991)

[Table 19 main 26](#_Toc454302992)

[Table 20 Run 27](#_Toc454302993)

[Table 21 Pembagian Kerja 41](#_Toc454302994)

DAFTAR SOURCE CODE

Table 1 Atribut

|  |
| --- |
| Atribute |
| public static final int DISPLAY\_HEIGHT = 480;  public static final int DISPLAY\_WIDTH = 640;  public static final Logger LOGGER = Logger.getLogger(Labyrinth.class.getName());  /\*\*  \* Defines if the application is resizable.  \*/  private static final boolean resizable = true;  /\*\*  \* The position of the player as a 3D vector (xyz).  \*/  private static Vector3f position = new Vector3f(0, 0, 0);  /\*\*  \* The rotation of the axis (where to the player looks). The X component  \* stands for the rotation along the x-axis, where 0 is dead ahead, 180 is  \* backwards, and 360 is automically set to 0 (dead ahead). The value must  \* be between (including) 0 and 360. The Y component stands for the rotation  \* along the y-axis, where 0 is looking straight ahead, -90 is straight up,  \* and 90 is straight down. The value must be between (including) -90 and  \* 90.  \*/  private static Vector3f rotation = new Vector3f(0, 0, 0);  /\*\*  \* The minimal distance from the camera where objects are rendered.  \*/  private static final float zNear = 0.3f;  /\*\*  \* The maximal distance from the camera where objects are rendered.  \*/  private static final float zFar = 100f;  /\*\*  \* The distance where fog starts appearing.  \*/  private static final float fogNear = 9;  /\*\*  \* The distance where the fog stops appearing (fully black here)  \*/  private static final float fogFar = 13f;  /\*\*  \* The color of the fog in rgba.  \*/  private static final Color fogColor = new Color(0f, 0f, 0f, 0.00001f);  /\*\*  \* Defines if the application utilizes full-screen.  \*/  private static final boolean fullscreen = false;  /\*\*  \* Defines the walking speed, where 10 is the standard.  \*/  private static int walkingSpeed = 10;  /\*\*  \* Defines the mouse speed.  \*/  private static int mouseSpeed = 2;  /\*\*  \* Defines if the applications prints its frames-per-second to the console.  \*/  private static final boolean printFPS = false;  /\*\*  \* Defines the maximum angle at which the player can look up.  \*/  private static final int maxLookUp = 85;  /\*\*  \* Defines the minimum angle at which the player can look down.  \*/  private static final int maxLookDown = -85;  /\*\*  \* Defines the field of view.  \*/  private static final int fov = 68;  private Texture m\_texture;  private int m\_texID[] = new int[10];  public final static int NO\_WRAP = -1;  private static int fps;  private static int delta;  private static long lastFPS;  private static long lastFrame; |

Table 2 Inisialisasi

|  |
| --- |
| public void resizeGL() |
| public void resizeGL() {  //2D Scene  glViewport(0, 0, DISPLAY\_WIDTH, DISPLAY\_HEIGHT);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glOrtho(-1.0f, 1.0f, -1.0f, 1.0f, -1.0f, 1.0f);  glPushMatrix();  glMatrixMode(GL\_MODELVIEW);  glLoadIdentity();  glPushMatrix();  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluPerspective(fov, (float) Display.getWidth() / (float) Display.getHeight(), zNear, zFar);  glMatrixMode(GL\_MODELVIEW);  glLoadIdentity();  glEnable(GL\_DEPTH\_TEST);  glEnable(GL\_TEXTURE\_2D);  glEnable(GL\_BLEND);  glEnable(GL\_ALPHA\_TEST);  glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA);  glHint(GL\_PERSPECTIVE\_CORRECTION\_HINT, GL\_NICEST);  } |

Table 3 Process Mouse

|  |
| --- |
| ProcessMouse() |
| public void processMouse() {  if (Mouse.isGrabbed()) {  float mouseDX = Mouse.getDX() \* mouseSpeed \* 0.16f;  float mouseDY = Mouse.getDY() \* mouseSpeed \* 0.16f;  if (rotation.y + mouseDX >= 360) {  rotation.y = rotation.y + mouseDX - 360;  } else if (rotation.y + mouseDX < 0) {  rotation.y = 360 - rotation.y + mouseDX;  } else {  rotation.y += mouseDX;  }  if (rotation.x - mouseDY >= maxLookDown && rotation.x - mouseDY <= maxLookUp) {  rotation.x += -mouseDY;  } else if (rotation.x - mouseDY < maxLookDown) {  rotation.x = maxLookDown;  } else if (rotation.x - mouseDY > maxLookUp) {  rotation.x = maxLookUp;  }  }  while (Mouse.next()) {  if (Mouse.isButtonDown(0)) {  Mouse.setGrabbed(true);  }  if (Mouse.isButtonDown(1)) {  Mouse.setGrabbed(false);  }  }  } |

Table 4 Process Keyboard

|  |
| --- |
| processKeyboard() |
| public void processKeyboard() {  boolean keyUp = Keyboard.isKeyDown(Keyboard.KEY\_UP) || Keyboard.isKeyDown(Keyboard.KEY\_W);  boolean keyDown = Keyboard.isKeyDown(Keyboard.KEY\_DOWN) || Keyboard.isKeyDown(Keyboard.KEY\_S);  boolean keyLeft = Keyboard.isKeyDown(Keyboard.KEY\_LEFT) || Keyboard.isKeyDown(Keyboard.KEY\_A);  boolean keyRight = Keyboard.isKeyDown(Keyboard.KEY\_RIGHT) || Keyboard.isKeyDown(Keyboard.KEY\_D);  boolean flyUp = Keyboard.isKeyDown(Keyboard.KEY\_SPACE);  boolean flyDown = Keyboard.isKeyDown(Keyboard.KEY\_LSHIFT);  boolean moveFaster = Keyboard.isKeyDown(Keyboard.KEY\_LCONTROL);  boolean moveSlower = Keyboard.isKeyDown(Keyboard.KEY\_TAB);  if (moveFaster && !moveSlower) {  walkingSpeed \*= 4f;  }  if (moveSlower && !moveFaster) {  walkingSpeed /= 10f;  }  if (keyUp && keyRight && !keyLeft && !keyDown) {  float angle = rotation.y + 45;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyUp && keyLeft && !keyRight && !keyDown) {  float angle = rotation.y - 45;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyUp && !keyLeft && !keyRight && !keyDown) {  float angle = rotation.y;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyDown && keyLeft && !keyRight && !keyUp) {  float angle = rotation.y - 135;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyDown && keyRight && !keyLeft && !keyUp) {  float angle = rotation.y + 135;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyDown && !keyUp && !keyLeft && !keyRight) {  float angle = rotation.y;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = -(walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyLeft && !keyRight && !keyUp && !keyDown) {  float angle = rotation.y - 90;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (keyRight && !keyLeft && !keyUp && !keyDown) {  float angle = rotation.y + 90;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  if (flyUp && !flyDown) {  double newPositionY = (walkingSpeed \* 0.0002) \* delta;  position.y -= newPositionY;  }  if (flyDown && !flyUp) {  double newPositionY = (walkingSpeed \* 0.0002) \* delta;  position.y += newPositionY;  }  if (moveFaster && !moveSlower) {  walkingSpeed /= 4f;  }  if (moveSlower && !moveFaster) {  walkingSpeed \*= 10f;  }  if (Keyboard.isKeyDown(Keyboard.KEY\_F)) {  System.out.println(position.y);  float angle = rotation.y;  Vector3f newPosition = new Vector3f(position);  float hypotenuse = (walkingSpeed \* 0.0002f) \* delta;  float adjacent = hypotenuse \* (float) Math.cos(Math.toRadians(angle));  float opposite = (float) (Math.sin(Math.toRadians(angle)) \* hypotenuse);  newPosition.z += adjacent;  newPosition.x -= opposite;  position.z = newPosition.z;  position.x = newPosition.x;  }  while (Keyboard.next()) {  if (Keyboard.isKeyDown(Keyboard.KEY\_C)) {  position = new Vector3f(0, 0, 0);  rotation = new Vector3f(0, 0, 0);  }  if (Keyboard.isKeyDown(Keyboard.KEY\_O)) {  mouseSpeed += 1;  System.out.println("Mouse speed changed to " + mouseSpeed + ".");  }  if (Keyboard.isKeyDown(Keyboard.KEY\_L)) {  if (mouseSpeed - 1 > 0) {  mouseSpeed -= 1;  System.out.println("Mouse speed changed to " + mouseSpeed + ".");  }  }  if (Keyboard.isKeyDown(Keyboard.KEY\_Q)) {  System.out.println("Walking speed changed to " + walkingSpeed + ".");  walkingSpeed += 1;  }  if (Keyboard.isKeyDown(Keyboard.KEY\_Z)) {  System.out.println("Walking speed changed to " + walkingSpeed + ".");  walkingSpeed -= 1;  }  if (Keyboard.isKeyDown(Keyboard.KEY\_N)) {  enableNight();  }  if (Keyboard.isKeyDown(Keyboard.KEY\_M)) {  disableNight();  }  }  } |

Table 5 Render

|  |
| --- |
| Render() |
| void render() {  glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);  delta = getDelta();  glLoadIdentity();  glRotatef(rotation.x, 1, 0, 0);  glRotatef(rotation.y, 0, 1, 0);  glRotatef(rotation.z, 0, 0, 1);  glTranslatef(position.x, position.y, position.z);  if (resizable) {  if (Display.wasResized()) {  glViewport(0, 0, Display.getWidth(), Display.getHeight());  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluPerspective(fov, (float) Display.getWidth() / (float) Display.getHeight(), zNear, zFar);  glMatrixMode(GL\_MODELVIEW);  glLoadIdentity();  }  }  //Draw SkyBox  drawSkyBox(80, 40);      //Draw Floor  glBindTexture(GL\_TEXTURE\_2D, m\_texID[6]);  glBegin(GL\_QUADS);  glTexCoord2f(0, 0);  glVertex3f(-50, -0.5f, 50f);  glTexCoord2f(0, 100);  glVertex3f(50f, -0.5f, 50f);  glTexCoord2f(100, 100);  glVertex3f(50f, -0.5f, -50f);  glTexCoord2f(100, 0);  glVertex3f(-50f, -0.5f, -50f);  glEnd();  //Draw wall  glBindTexture(GL\_TEXTURE\_2D, m\_texID[0]);  glBegin(GL\_QUADS);  createWall();  glEnd();    //Draw tree  for (int i = 1; i <= 10; i++) {  glTranslatef(1.5f, 0f, 0);  drawTree(0.5f,0.5f);  }  for (int i = 1; i <= 20; i++) {  glTranslatef(-1.5f, 0f, 0);  drawTree(0.5f,0.5f);  }  } |

Table 6 Create Wall

|  |
| --- |
| createWall() |
| void createWall(){  glColor3f(1.0f, 1.0f, 1.0f);  // Outer front left  drawWall(-1.0f, -12.0f, -1.5f, -2.5f);  // Outer front right  drawWall(0.0f, 13.0f, -1.5f, -2.5f);  // Outer Back  drawWall(12.5f, -11.0f, -24.5f, -25.5f);  // Outer Left  drawWall(-11.5f, -12.0f, -2.0f, -25.5f);  // Outer Right  drawWall(12.5f, 12.0f, -2.0f, -25.5f);  // Vertical left to right  //Line 1  drawWall(-9.5f, -10.0f, -3.5f, -5.5f);  drawWall(-9.5f, -10.0f, -6.5f, -8.5f);  drawWall(-9.5f, -11.5f, -10.5f, -11.5f);  drawWall(-9.5f, -10.0f, -12.5f, -23.5f);  drawWall(-6.5f, -10.0f, -22.5f, -23.5f);  //Line 2  drawWall(-7.5f, -8.0f, -3.5f, -7.5f);  drawWall(-7.5f, -8.0f, -8.5f, -12.5f);  drawWall(-7.5f, -8.0f, -12.5f, -13.5f);  drawWall(-7.5f, -8.0f, -17.5f, -21.5f);  drawWall(-4.5f, -8.0f, -20.5f, -21.5f);  //Line 3  drawWall(-5.5f, -6.0f, -5.5f, -9.5f);  drawWall(-3.5f, -6.0f, -10.5f, -11.5f);  drawWall(-5.5f, -8.0f, -14.0f, -15.0f);  drawWall(-5.5f, -6.0f, -12.5f, -19.5f);  // Line 4  drawWall(-3.5f, -4.0f, -3.5f, -6.5f);  drawWall(-2.5f, -3.0f, -7.5f, -8.5f);  drawWall(-3.5f, -4.0f, -7.5f, -13.5f);  drawWall(-3.5f, -4.0f, -14.5f, -23.5f);  drawWall(0.5f, -3.0f, -15.5f, -16.5f);  drawWall(-2.5f, -3.0f, -19.5f, -20.5f);  drawWall(-2.5f, -3.0f, -22.5f, -23.5f);  // Line 5  drawWall(-1.5f, -2.0f, -5.5f, -8.5f);  drawWall(-1.5f, -2.0f, -9.5f, -16.5f);  drawWall(5.5f, -2.0f, -11.5f, -12.5f);  drawWall(-1.5f, -2.0f, -17.5f, -21.5f);  drawWall(3.5f, -2.0f, -21.0f, -22.0f);  drawWall(-0.5f, -1.0f, -22.5f, -25.5f);  // Line 6  drawWall(0.5f, 0.0f, -3.5f, -10.5f);  drawWall(0.5f, 0.0f, -16.5f, -18f);  drawWall(2.5f, 0.0f, -19.0f, -20.0f);  drawWall(0.5f, 2.0f, -21.5f, -23.5f);  // Line 7  drawWall(2.5f, 2.0f, -5.5f, -11.5f);  drawWall(3.5f, 2.0f, -15.5f, -16.5f);  drawWall(1.5f, 3.0f, -15.5f, -19.0f);  drawWall(2.0f, 3.f, -15.5f, -19.0f);  // Line 8  drawWall(3.5f, 5.0f, -3.5f, -10.5f);  drawWall(3.5f, 5.0f, -11.5f, -16.5f);  drawWall(3.5f, 5.0f, -17.5f, -22.0f);  drawWall(3.0f, 6.0f, -22.5f, -23.5f);  // Line 9  drawWall(5.5f, 9.0f, -5.5f, -6.5f);  drawWall(5.5f, 7.0f, -5.5f, -12.5f);  drawWall(6.0f, 8.0f, -9.25f, -11.0f);  drawWall(5.0f, 7.25f, -13.25f, -14.5f);  drawWall(6.0f, 8.5f, -15.0f, -16.0f);  drawWall(4.0f, 7.5f, -17.5f, -18.5f);  drawWall(5.0f, 6.5f, -19.0f, -23.5f);  drawWall(6.0f, 7.5f, -15.5f, -20.0f);  drawWall(7.0f, 9.0f, -18.0f, -19.0f);  // Line 10  drawWall(7.0f, 9.5f, -7.25f, -8.5f);  drawWall(7.0f, 9.5f, -11.5f, -12.5f);  drawWall(8.0f, 9.5f, -5.5f, -12.5f);  drawWall(6.5f, 9.0f, -21.0f, -22.0f);  drawWall(7.5f, 9.0f, -18.0f, -21.0f);  drawWall(6.5f, 8.0f, -22.5f, -25.5f);  // Line 11  drawWall(9.75f, 11.25f, -4.5f, -14.5f);  drawWall(8.25f, 11.25f, -13.5f, -14.5f);  drawWall(8.5f, 10.0f, -13.5f, -17.5f);  drawWall(10.5f, 12.0f, -15.25f, -16.0f);  drawWall(8.0f, 11.0f, -16.5f, -17.5f);  drawWall(9.5f, 12.0f, -17.5f, -19.0f);  drawWall(9.5f, 11.0f, -17.5f, -19.5f);  drawWall(9.5f, 11.0f, -20.5f, -23.5f);  drawWall(8.0f, 11.0f, -22.5f, -23.5f);  // Horizontal from down to up  drawWall(10.75f, -9.5f, -3.5f, -4.5f);  drawWall(-5.5f, -7.5f, -6.5f, -7.5f);  drawWall(-7.5f, -10.0f, -8.5f, -9.5f);  drawWall(-7.5f, -10.0f, -12.5f, -13.5f);  drawWall(-7.5f, -10.0f, -15.5f, -16.5f);  } |

Table 7 Draw Wall

|  |
| --- |
| drawWall() |
| public void drawWall(float xKiri, float xKanan, float zDepan, float zBelakang) {  float x, z;  if (resizable) {  x = Math.abs(xKanan - xKiri);  z = Math.abs(zDepan - zBelakang);  }  //Kanan  glTexCoord2f(0, 0);  glVertex3f(xKanan, -0.5f, zDepan);  glTexCoord2f(0, 1);  glVertex3f(xKanan, 0.5f, zDepan);  glTexCoord2f(z, 1);  glVertex3f(xKanan, 0.5f, zBelakang);  glTexCoord2f(z, 0);  glVertex3f(xKanan, -0.5f, zBelakang);  // //Kiri  glTexCoord2f(0, 0);  glVertex3f(xKiri + 0.5f, -0.5f, zDepan);  glTexCoord2f(0, 1);  glVertex3f(xKiri + 0.5f, 0.5f, zDepan);  glTexCoord2f(z, 1);  glVertex3f(xKiri + 0.5f, 0.5f, zBelakang);  glTexCoord2f(z, 0);  glVertex3f(xKiri + 0.5f, -0.5f, zBelakang);  //Belakang  glTexCoord2f(0, 0);  glVertex3f(xKanan, -0.5f, zBelakang);  glTexCoord2f(0, 1);  glVertex3f(xKanan, 0.5f, zBelakang);  glTexCoord2f(x, 1);  glVertex3f(xKiri + 0.5f, 0.5f, zBelakang);  glTexCoord2f(x, 0);  glVertex3f(xKiri + 0.5f, -0.5f, zBelakang);  // Depan  glTexCoord2f(0, 0);  glVertex3f(xKanan, -0.5f, zDepan);  glTexCoord2f(0, 1);  glVertex3f(xKanan, 0.5f, zDepan);  glTexCoord2f(x, 1);  glVertex3f(xKiri + 0.5f, 0.5f, zDepan);  glTexCoord2f(x, 0);  glVertex3f(xKiri + 0.5f, -0.5f, zDepan);  //Atas  if (z > x) {  glTexCoord2f(0, 0);  glVertex3f(xKanan, 0.5f, zDepan);  glTexCoord2f(0, x);  glVertex3f(xKiri + 0.5f, 0.5f, zDepan);  glTexCoord2f(z, 1);  glVertex3f(xKiri + 0.5f, 0.5f, zBelakang);  glTexCoord2f(z, 0);  glVertex3f(xKanan, 0.5f, zBelakang);  } else {  glTexCoord2f(0, 0);  glVertex3f(xKanan, 0.5f, zDepan);  glTexCoord2f(0, 1);  glVertex3f(xKanan, 0.5f, zBelakang);  glTexCoord2f(x, 1);  glVertex3f(xKiri + 0.5f, 0.5f, zBelakang);  glTexCoord2f(x, 0);  glVertex3f(xKiri + 0.5f, 0.5f, zDepan);  }  glEndList();  } |

Table 8 Draw Skybox

|  |
| --- |
| drawSkyBox() |
| private void drawSkyBox(float width, float height) {  width = width / 2;  // front side  glBindTexture(GL\_TEXTURE\_2D, m\_texID[1]);  glBegin(GL\_QUADS);  glTexCoord2d(0, 0);  glVertex3f(-width, -0.5f, -width);  glTexCoord2d(1, 0);  glVertex3f(width, -0.5f, -width);  glTexCoord2d(1, 1);  glVertex3f(width, height, -width);  glTexCoord2d(0, 1);  glVertex3f(-width, height, -width);  glEnd();  // right side  glBindTexture(GL\_TEXTURE\_2D, m\_texID[2]);  glBegin(GL\_QUADS);  glTexCoord2d(0, 0);  glVertex3f(width, -0.5f, -width);  glTexCoord2d(1, 0);  glVertex3f(width, -0.5f, width);  glTexCoord2d(1, 1);  glVertex3f(width, height, width);  glTexCoord2d(0, 1);  glVertex3f(width, height, -width);  glEnd();  // back side  glBindTexture(GL\_TEXTURE\_2D, m\_texID[3]);  glBegin(GL\_QUADS);  glTexCoord2d(0, 0);  glVertex3f(width, -0.5f, width);  glTexCoord2d(1, 0);  glVertex3f(-width, -0.5f, width);  glTexCoord2d(1, 1);  glVertex3f(-width, height, width);  glTexCoord2d(0, 1);  glVertex3f(width, height, width);  glEnd();  // left side  glBindTexture(GL\_TEXTURE\_2D, m\_texID[4]);  glBegin(GL\_QUADS);  glTexCoord2d(0, 0);  glVertex3f(-width, -0.5f, width);  glTexCoord2d(1, 0);  glVertex3f(-width, -0.5f, -width);  glTexCoord2d(1, 1);  glVertex3f(-width, height, -width);  glTexCoord2d(0, 1);  glVertex3f(-width, height, width);  glEnd();  // top side  glBindTexture(GL\_TEXTURE\_2D, m\_texID[5]);  glBegin(GL\_QUADS);  glTexCoord2d(1, 1);  glVertex3f(width, height, -width);  glTexCoord2d(0, 1);  glVertex3f(width, height, width);  glTexCoord2d(0, 0);  glVertex3f(-width, height, width);  glTexCoord2d(1, 0);  glVertex3f(-width, height, -width);  glEnd();  } |

Table 9 Draw Tree

|  |
| --- |
| drawTree() |
| private void drawTree(float width, float height) {  width = width / 2;  glBindTexture(GL\_TEXTURE\_2D, m\_texID[7]);  glBegin(GL\_QUADS);  // front quads  glTexCoord2d(0, 0);  glVertex3f(-width, -0.5f, 0);  glTexCoord2d(1, 0);  glVertex3f(width, -0.5f, 0);  glTexCoord2d(1, 1);  glVertex3f(width, height, 0);  glTexCoord2d(0, 1);  glVertex3f(-width, height, 0);  // cross quads  glTexCoord2d(0, 0);  glVertex3f(0, -0.5f, width);  glTexCoord2d(1, 0);  glVertex3f(0, -0.5f, -width);  glTexCoord2d(1, 1);  glVertex3f(0, height, -width);  glTexCoord2d(0, 1);  glVertex3f(0, height, width);  glEnd();  } |

Table 10 Enable Night

|  |
| --- |
| enableNight() |
| public void enableNight(){  glEnable(GL\_FOG);  {  FloatBuffer fogColours = BufferUtils.createFloatBuffer(4);  fogColours.put(new float[]{fogColor.r, fogColor.g, fogColor.b, fogColor.a});  glClearColor(fogColor.r, fogColor.g, fogColor.b, fogColor.a);  fogColours.flip();  glFog(GL\_FOG\_COLOR, fogColours);  glFogi(GL\_FOG\_MODE, GL\_LINEAR);  glHint(GL\_FOG\_HINT, GL\_NICEST);  glFogf(GL\_FOG\_START, fogNear);  glFogf(GL\_FOG\_END, fogFar);  glFogf(GL\_FOG\_DENSITY, 0.005f);  }  } |

Table 11 Disable Night

|  |
| --- |
| disableNight() |
| public void disableNight(){  glDisable(GL\_FOG);  } |

Table 12 load Texture

|  |
| --- |
| loadTexture t() |
| private void loadTexture(int texID, String path, boolean transparent, int filter, int wrap) {  m\_texture = new Texture();  if (!m\_texture.load(path)) {  System.out.println("Failed to load texture\n");  return;  }  glBindTexture(GL\_TEXTURE\_2D, texID);  if (transparent) {  glTexImage2D(GL\_TEXTURE\_2D, 0, GL\_RGBA, m\_texture.getWidth(),  m\_texture.getHeight(), 0, GL\_RGBA, GL\_UNSIGNED\_BYTE,  m\_texture.getImageData());  } else {  glTexImage2D(GL\_TEXTURE\_2D, 0, GL\_RGB, m\_texture.getWidth(),  m\_texture.getHeight(), 0, GL\_RGB, GL\_UNSIGNED\_BYTE,  m\_texture.getImageData());  }  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MIN\_FILTER, filter);  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MAG\_FILTER, filter);  if (wrap == -1) {  return;  }  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_S, wrap);  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_T, wrap);  } |

Table 13 Destroy

|  |
| --- |
| destroy() |
| void destroy() {  Keyboard.destroy();  Mouse.destroy();  Display.destroy();  } |

Table 14 Update FPS

|  |
| --- |
| updateFPS() |
| private static void updateFPS() {  if (getTime() - lastFPS > 1000) {  if (printFPS) {  System.out.println("FPS: " + fps);  }  fps = 0;  lastFPS += 1000;  }  fps++;  } |

Table 15 get Delta

|  |
| --- |
| getDelta |
| private static int getDelta() {  long currentTime = getTime();  int delta = (int) (currentTime - lastFrame);  lastFrame = getTime();  return delta;  } |

Table 16 get Time

|  |
| --- |
| getTime() |
| private static long getTime() {  return (Sys.getTime() \* 1000) / Sys.getTimerResolution();  } |

Table 17 Create

|  |
| --- |
| create() |
| public void create() throws LWJGLException {  Display.setDisplayMode(new DisplayMode(DISPLAY\_WIDTH, DISPLAY\_HEIGHT));  Display.setResizable(resizable);  Display.setTitle("Labyrinth");  Display.create();  //Create Keyboard  Keyboard.create();  //Create Mouse  Mouse.setGrabbed(false);  Mouse.create();  initGL();  resizeGL();  } |

Table 18 initGL

|  |
| --- |
| initGL() |
| public void initGL() {  //2D Initialization  glColor3b((byte) 255, (byte) 0, (byte) 0);  glClearColor(1.0f, 1.0f, 1.0f, 0.0f);  IntBuffer textureIDBuffer = BufferUtils.createIntBuffer(10);  glGenTextures(textureIDBuffer);  m\_texID[0] = textureIDBuffer.get(0);//Wall  m\_texID[1] = textureIDBuffer.get(1);//Sky West  m\_texID[2] = textureIDBuffer.get(2);//Sky East  m\_texID[3] = textureIDBuffer.get(3);//Sky North  m\_texID[4] = textureIDBuffer.get(4);//Sky South  m\_texID[5] = textureIDBuffer.get(5);//Sky Up  m\_texID[6] = textureIDBuffer.get(6);//Grass  m\_texID[7] = textureIDBuffer.get(7);//Tree  // Load 1 texture  loadTexture(m\_texID[0], "C:/libs/LWJGL/res/images/wall.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[1], "C:/libs/LWJGL/res/images/0.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[2], "C:/libs/LWJGL/res/images/90.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[3], "C:/libs/LWJGL/res/images/180.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[4], "C:/libs/LWJGL/res/images/270.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[5], "C:/libs/LWJGL/res/images/up.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[6], "C:/libs/LWJGL/res/images/grass.png", false, GL\_LINEAR, GL\_REPEAT);  loadTexture(m\_texID[7], "C:/libs/LWJGL/res/images/tree.png", true, GL\_LINEAR, NO\_WRAP);  } |

Table 19 main

|  |
| --- |
| main() |
| public static void main(String[] args) {  Labyrinth main = null;  try {  main = new Labyrinth();  main.create();  main.run();  } catch (Exception ex) {  LOGGER.log(Level.SEVERE, ex.toString(), ex);  } finally {  if (main != null) {  main.destroy();  }  }  } |

Table 20 Run

|  |
| --- |
| main() |
| void run() {  while (!Display.isCloseRequested() && !Keyboard.isKeyDown(Keyboard.KEY\_ESCAPE)) {  if (Display.isVisible()) {  processKeyboard();  processMouse();  render();  }  Display.update();  Display.sync(100);  }  } |

PENDAHULUAN

# LATAR BELAKANG

Labirin Coban Rondo merupakan salah satu dari berbagai wahana yang ada di kawasan wisata alam air terjun Coban Rondo, Labirin atau taman sesat ini menjadi magnet baru untuk menarik wisatawan untuk datang. Letak yang tak jauh dari pintu masuk pertama kawasan hutandi coban rondo mempermudah bagi para pengunjung untuk menemukan tempat yang satu ini.

Tidak cuma labirin, sebenarnya wahana tersebut merupakan bagian dari area outbound. Taman sesat (labirin) tersebut disusun dengan memanfaatkan tanaman hijau yang menjulang tinggi sekitar 2 meter yang menyerupai dinding-dinding, meski untuk ukuran sebenarnya tak begitu besar dan luas namun cukup membuat para pengunjung kebingungan untuk sampai pada pusat labirin, suatu ruang di tengah labirin yang di tandai dengan kolam dan air mancur serta beberapa buah bangku. Tak sedikit yang mencoba memasuki labirin ini meminta petunjuk teman yang memantau dari menara yang terletak di pintu masuk labirin. Meskipun cara tersebut curang namun tak akan ada yang menegur Anda, jadi tenang saja. Jangan kaget juga soalnya saat disana nanti akan menemukan banyak orang yang bingung dan mendapatkan banyak panduan dari teman mereka yang berada di menara yang kadang sangat lucu.

# Tujuan dan Manfaat

* Tujuan

1. Merancang dan membuat sebuah desain dan model labirin Coban Rondo yang mampu berkomunikasi secara visual dengan user.
2. Merancang dan membuat program pada bahasa pemogram yang mengimplementasikan fungsi – fungsi pada OpenGL untuk membuat desain dan model labirin Coban Rondo
3. Merancang dan membuat ilustrasi labirin Coban Rondo yang bisa dinikmati experiencenya secara visual

* Manfaat

1. Di Bidang Pariwisata

Ilustrasi Labirin Coban Rondo dapat membantu memperkenalkan wisata Coban Rondo. Dengan penggunakan bentuk objek yang dibuat sedemikian rupa ini diharapkan mengetahui bentuk dan simulasinya.

1. Di Bidang Hiburan

Ilustrasi Coban Rondo dalam grafika komputer yang berupa efek animasi yang dapat membuat pengguna dapat menikmati sensasi berkeliling labirin sesuai dengan aslinya sehingga pengguna terhibur dan tertarik.

1. Di Bidang Perancangan

Pada bidang ini grafik komputer digunakan untuk membuat desain dan model jembatan Labirin Coban Rondo digunakan untuk mendesain suatu arsitektur Labirin Coban Rondo,dan simulasi tentang Labirin Coban Rondo. Menggunakan alat desain grafis seperti OpenGL. menhasilkan suatu model yang nyaris sama seperti aslinya.

# Batasan Masalah

1. Pembuatan project akhir ini berfokus pada metode yang telah dipelajari seperti Drawing 2D & 3D, transformasi, input user, texture mapping, viewing, blending, lighting, dan shading

.

1. Kerumitan dalam menyamakan model yang dibuat dengan mdoel aslinya sehingga dapat terlihat semirip mungkin, penerapan sintaks - sintaks yang digunakan dalam membuat objek tersebut, serta penerapan fungsi - fungsi untuk membuat objek lebih interaktif.

tinjauan pustaka

# Tinjauan pusataka 1

Aplikasi yang digunakan :

Netbeans.

Apliaksi ini kami gunakan sebagai compiler yang kami gunakan untuk mengcompile kode - kode dalam pembuatan project ini

# Tinjauan pusataka 2

Modul OpenGL yang digunakan :

Lighweight Java Game Library(LWJGL).

Lightweight Java Game Library (LWJGL) merupakan salah satu jenis game engine yang berjalan dengan dasar bahasa java dan dibawah lisesnsi BSD.

deskripsi karya

# Deskripsi Karya

Project ini selain dapat memvisualisasikan labirin Coban Rondo secara grafik juga ada game interaktifnya. Dalam pemviasualisasinya game ini dibuat semirip mungkin dengan labirin Coban Rondo yang asli dan dengan visualisasi yang dapat menampilkan view dalam berbagai sudut view. Dari sisi game project ini dapat membawa pengguna seakan-akan masuk dalam labirin yang asli dengan efek 3D.

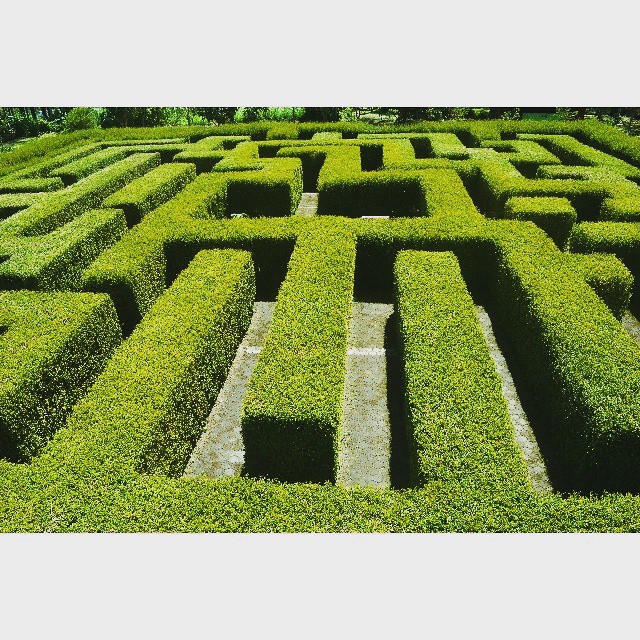
# Objek - Objek Dalam Karya

Objek yang ada dalam project ini

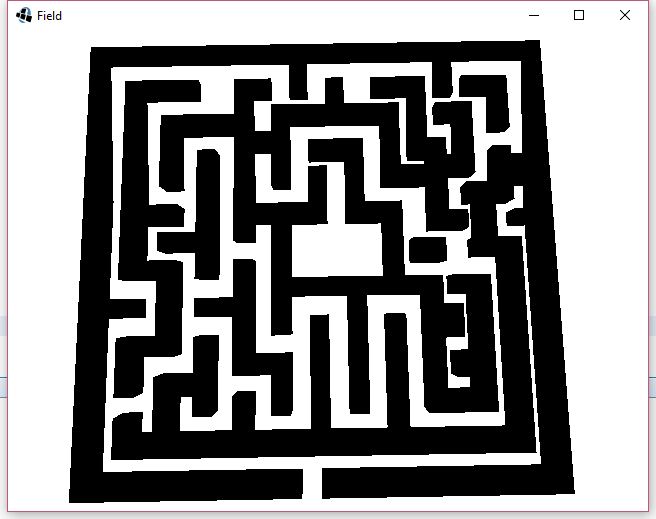
* Pembatas labirin



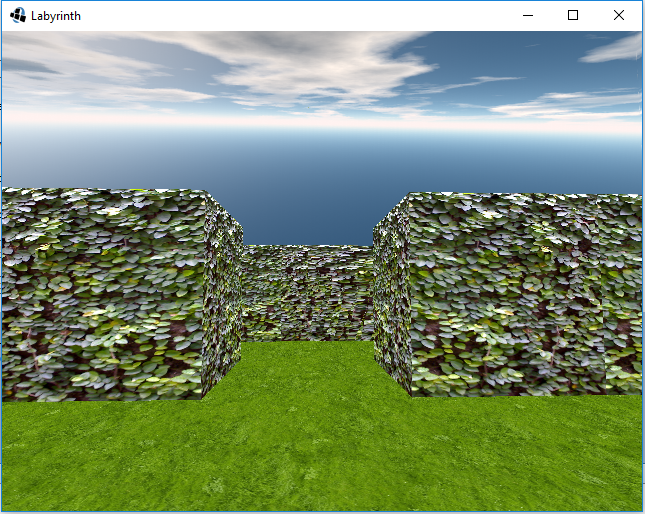
Gambar 1. 1 Labirin Coban rondo (dari atas)



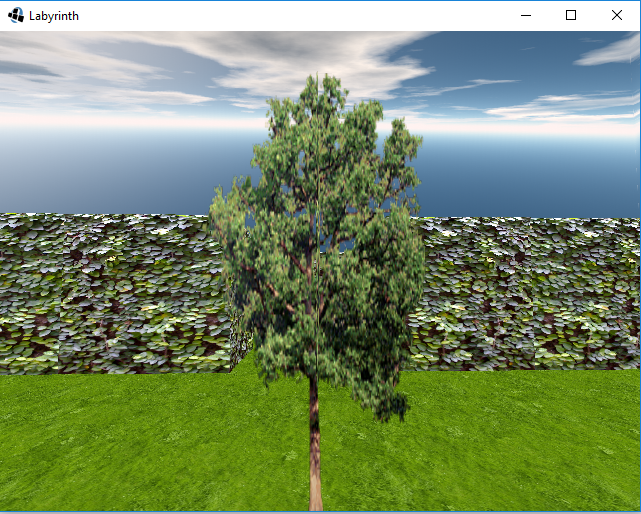
Gambar1. 2 Labirin Coban rondo dari atas samping



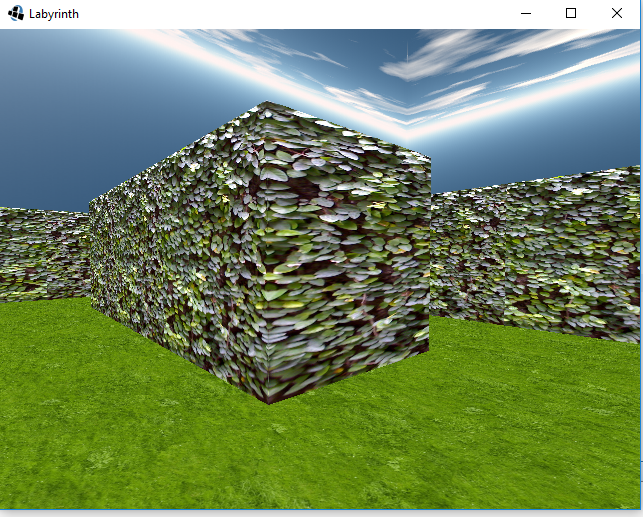
Gambar1. 3 Implementasi Labirin Coban Rondo (dari atas)



Gambar1.4Implementasi Labirin Coban Rondo (Dari depan)



Gambar 1.5 Pohon



Gambar 1.6 Implementasi Bagian-Bagian Labirin

* Orang
* Air mancur



Gambar 1.7 Air Mancur Coban rondo

* Bangku
* Menara

# Penerapan Modul

Sebutkan modul-modul yang dipakai pada Final Project ini, minimal meliputi: Drawing 2D & 3D, transformasi, input user, texture mapping, viewing, blending, lighting, dan shading.

Jelaskan dipakai untuk apa modul-modul tersebut.

# Interaksi User

Interaksi yang dapat dilakukan oleh user diantaranya adalah dapat menggerakkan viewing sesuai dengan keingginanya, dan user pula dapat memainkan seseorang untuk bergerak melewati labirin tersebut.

jadwal pelaksanaan

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Bulan(2016) | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Kegiatan |  | Bulan 2 | | |  | Bulan 3 | | |  | Bulan 4 | | |  | Bulan 5 | | |  | Bulan 6 | | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Penentuan Judul |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pencarian referensi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementasi ke dalam kode program |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Evalusasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Laporan akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

anggota kelompok

# Biografi Kelompok

Ketua :

**Biografi Ketua**

1. Nama : Robihamanto

2. NIM : 145150207111067

3. Tempat dan tanggal lahir : Serang, 4 Mei 1996

4. Jenis kelamin : Laki – Laki

5. Fakultas : Fakultas Ilmu Komputer

6. Jurusan : Teknik Informatika

7. Semester : 4 (Empat)

8. Jumlah SKS yang pernah ditempuh : 60

9. Alamat Rumah asal : Jl. Purna Bhakti 2 No 46 A Kp.Gurugui Kec. Taktakan Serang, Banten.

10. Alamat Rumah di Malang : Jalan Kalpataru Gang 1 No. 83C, Lowokwaru, Malang

11. Organisasi : DPM Filkom, BCC Filkom, K-Risma Filkom, Rkim Universitas Brawijaya.

Anggota :

**Biografi Anggota 1**

1. Nama : Bossarito Putro

2. NIM : 145150201111068

3. Tempat dan tanggal lahir : Karanganyar, 29 November 1995

4. Jenis kelamin : Laki – Laki

5. Fakultas : Fakultas Ilmu Komputer

6. Jurusan : Teknik Informatika

7. Semester : 4 (Empat)

8. Jumlah SKS yang pernah ditempuh : 60

9. Alamat Rumah asal : Kodokan RT 04/01, Papahan, Tasikmadu, Jawa Tengah

10. Alamat Rumah di Malang : Jalan Kalpataru Gang 1 No. 83C, Lowokwaru, Malang

11. Organisasi : Eksekutif Mahasiswa Informatika (EMIF) 2016

**Biografi Anggota 2**

1. Nama : Andre Rizal Sinaga

2. NIM : 145150201111066

3. Tempat dan tanggal lahir : Medan, 8 Oktober 1996

4. Jenis kelamin : Laki – Laki

5. Fakultas : Fakultas Ilmu Komputer

6. Jurusan : Teknik Informatika

7. Semester : 4 (Empat)

8. Jumlah SKS yang pernah ditempuh : 60

9. Alamat Rumah asal : Jalan Jati 3 gang Ampera 2 No.4A Medan

10. Alamat Rumah di Malang : Jalan Sumbersari gang IV No.261A Malang

11. Organisasi : Dewan Perwakilan Mahasiswa Filkom 2016

**Biografi Anggota 3**

1. Nama : Rendy Mardiansyah

2. NIM : 145150201111110

3. Tempat dan tanggal lahir : Sidoarjo, 9 Maret 1996

4. Jenis kelamin : Laki – Laki

5. Fakultas : Fakultas Ilmu Komputer

6. Jurusan : Teknik Informatika

7. Semester : 4 (Empat)

8. Jumlah SKS yang pernah ditempuh : 60

9. Alamat Rumah asal : Jalan Mayjen Panjaitan gang 19 no 47

10. Alamat Rumah di Malang : Jalan Raya lengudi RT05 RW2 Kriilan, Drinoyo Gresik

11. Organisasi : -

# Pembagian Kerja

Table 21 Pembagian Kerja

|  |  |  |  |
| --- | --- | --- | --- |
| Nama/Nim | Jurusan/Angkatan | Lama Kerja | Keterangan |
| Robihamanto/ 145150207111067 | Informatika/2014 | 2 Jam/Minggu | Implementasi model/objek kedalam kode program, Pembuatan Labirin dan textureing. |
| Andre Rizal Sinaga / 145150201111066 | Informatika/2014 | 2 Jam/Minggu | Pencarian Referensi, Pembuatan langit dan lantai |
| Bossarito Putro / 145150201111068 | Informatika/2014 | 2 Jam/Minggu | Pencarian Referensi, Pembuatan laporan, Pembuatan effect siang dan malam |
| Randy Mardiansyah / 145150201111110 | Informatika/2014 | 2 Jam/Minggu | Pencarian Referensi |

daftar pustaka

<http://3.bp.blogspot.com/-Pf_wkk40R5k/VqmkF4Lcq4I/AAAAAAAAANU/-lqappXqT-Y/s640/12568187_1128881723790344_2142983493_n.jpg>

<https://scontent.cdninstagram.com/hphotos-xfa1/t51.2885-15/e15/11241386_426416100869247_2000815524_n.jpg>

<http://www.mibah.com/labirin-coban-rondo-pujon-malang.html>