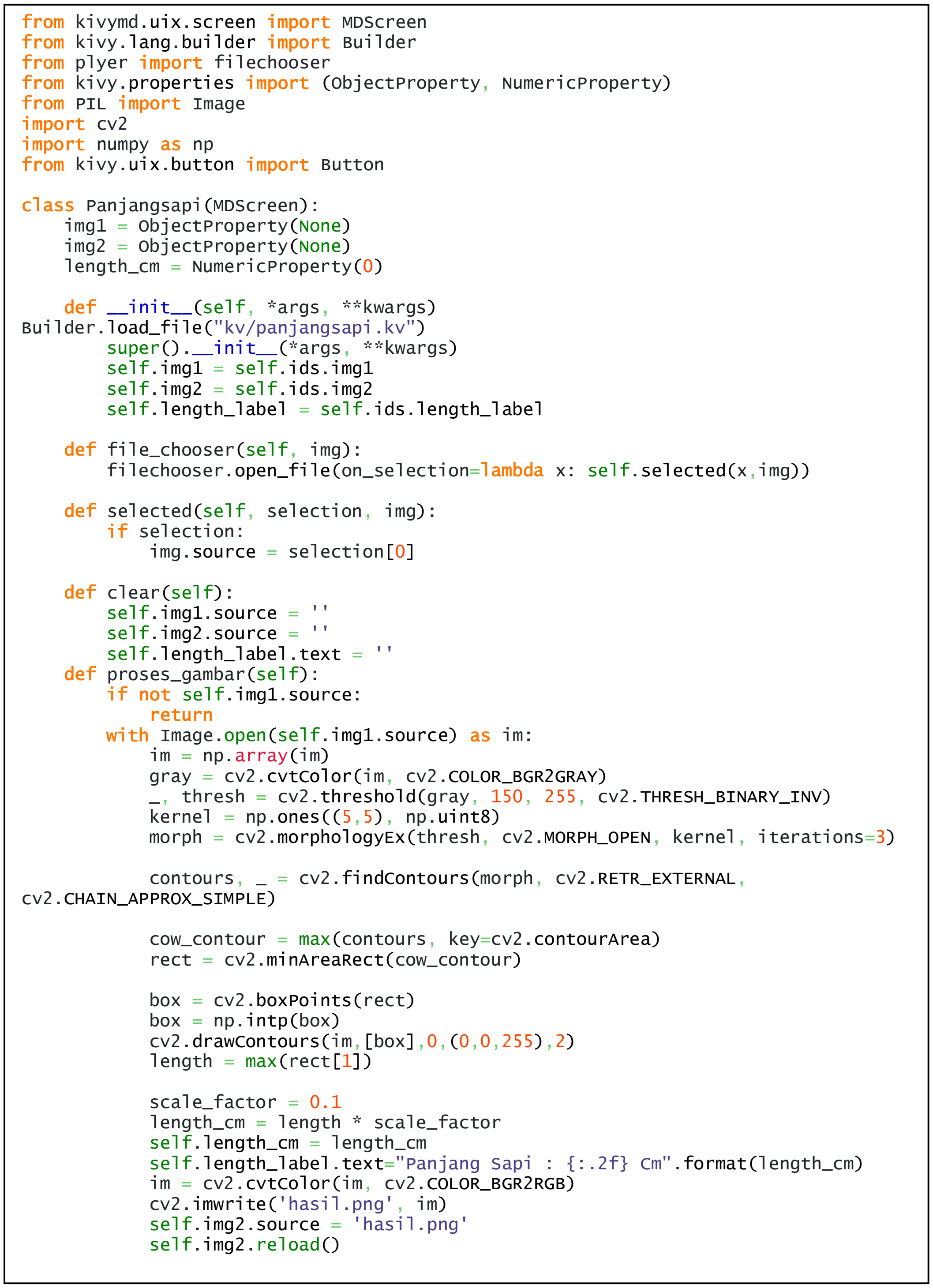
LAMPIRAN – LAMPIRAN

DOKUMENTASI CODE

```python
from kivy.app import App
from kivymd.app import MDApp
from kivy.uix.screenmanager import Screen,ScreenManager
from kivy.core.text import LabelBase
from layar.screens import*
from kivymd.app import MDApp


class WindoManager(ScreenManager):
    pass

class Test(MDApp):
    DEBUG = 1
    #untuk hotreload
    CLASSES = {
        'Welcome':'layar.welcome',
        'Masuk':'layar.masuk',
        'Signup':'layar.signup',
        'Panjangsapi':'layar.panjangsapi',
        'Lingkardada':'layar.lingkardada',
        'Rumus' : 'layar.rumus'
    }
    AUTORELOADER_PATHS = [
        ('.', {'recursive': True})
    ]
    KV_FILES = [
        'kv/welcome.kv',
        'kv/masuk.kv',
        'kv/signup.kv',
        'kv/panjangsapi.kv',
        'kv/lingkardada.kv',
        'kv/rumus.kv'
    ]
    def build(self):
        self.wm = WindoManager()
        screens = [
            Welcome(name ="welcome"),
            Masuk(name= "masuk"),
            Signup(name="signup"),
            Panjangsapi(name="panjangsapi"),
            Lingkardada(name="lingkardada"),
            Rumus(name="rumus")
        ]
        for screen in screens:
            self.wm.add_widget(screen)

        return self.wm

if __name__ == '__main__':
    LabelBase.register(name="MPoppins", fn_regular="aset\\font\\Poppins-Medium.ttf")
    LabelBase.register(name="BPoppins", fn_regular="aset\\font\\Poppins-SemiBold.ttf")
    Test().run()
```

Gambar kode program main.py



```
from kivymd.uix.screen import MDScreen
from kivy.lang.builder import Builder
from plyer import filechooser
from kivy.properties import (ObjectProperty, NumericProperty)
from PIL import Image
import cv2
import numpy as np
from kivy.uix.button import Button

class Panjangsapi(MDScreen):
    img1 = ObjectProperty(None)
    img2 = ObjectProperty(None)
    length_cm = NumericProperty(0)

    def __init__(self, *args, **kwargs)
Builder.load_file("kv/panjangsapi.kv")
        super().__init__(*args, **kwargs)
        self.img1 = self.ids.img1
        self.img2 = self.ids.img2
        self.length_label = self.ids.length_label

    def file_chooser(self, img):
        filechooser.open_file(on_selection=lambda x: self.selected(x,img))

    def selected(self, selection, img):
        if selection:
            img.source = selection[0]

    def clear(self):
        self.img1.source = ''
        self.img2.source = ''
        self.length_label.text = ''
    def proses_gambar(self):
        if not self.img1.source:
            return
        with Image.open(self.img1.source) as im:
            im = np.array(im)
            gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
            _, thresh = cv2.threshold(gray, 150, 255, cv2.THRESH_BINARY_INV)
            kernel = np.ones((5,5), np.uint8)
            morph = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=3)

            contours, _ = cv2.findContours(morph, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

            cow_contour = max(contours, key=cv2.contourArea)
            rect = cv2.minAreaRect(cow_contour)

            box = cv2.boxPoints(rect)
            box = np.intp(box)
            cv2.drawContours(im,[box],0,(0,0,255),2)
            length = max(rect[1])

            scale_factor = 0.1
            length_cm = length * scale_factor
            self.length_cm = length_cm
            self.length_label.text="Panjang Sapi : {:.2f} Cm".format(length_cm)
            im = cv2.cvtColor(im, cv2.COLOR_BGR2RGB)
            cv2.imwrite('hasil.png', im)
            self.img2.source = 'hasil.png'
            self.img2.reload()
```

Gambar kode program mencari panjang sapi

```python
from kivymd.app import MDApp
from kivymd.uix.screen import MDScreen
from kivy.lang.builder import Builder
from plyer import filechooser
from kivy.properties import ObjectProperty, ListProperty,NumericProperty
from PIL import Image
from kivy.graphics import Line, Color

class Lingkardada(MDScreen):
    img1 = ObjectProperty(None)
    points = ListProperty([])
    lines = ListProperty([])
    circumference_cm = NumericProperty(0)

    def __init__(self, *args, **kwargs):
        Builder.load_file("kv/lingkardada.kv")
        super().__init__(*args, **kwargs)
        self.img1 = self.ids.img1

    def file_chooser(self, img):
        filechooser.open_file(on_selection=lambda x: self.selected(x,img))

    def selected(self, selection, img):
        if selection:
            img.source = selection[0]

    def clear(self):
        self.img1.source = ''
        self.points = []
        self.circumference_cm = 0
        self.ids.circumference_label.text = "Lingkar dada: 0.0 Cm"
        for line in self.lines:
            self.img1.canvas.remove(line)
        self.lines = []

    def on_touch_down(self, touch):
        if self.img1.collide_point(*touch.pos):
            self.points.append(touch.pos)
            with self.img1.canvas:
                Color(1, 0, 0)
                line = Line(points=(touch.pos[0]-5, touch.pos[1]-5), width=3)
                self.lines.append(line)
        return super().on_touch_down(touch)

    def on_touch_up(self, touch):
        if self.img1.collide_point(*touch.pos):
            self.points.append(touch.pos)
            with self.img1.canvas:
                Color(1, 0, 0)
                line = Line(points=self.points, width=3)
                self.lines.append(line)
        return super().on_touch_up(touch)

    def calculate_circumference(self):
        if len(self.points) > 2:
            circumference = 0
            for i in range(len(self.points)-1):
                x1, y1 = self.points[i]
                x2, y2 = self.points[i+1]
                diameter = ((x2 - x1) ** 2 + (y2 - y1) ** 2) ** 0.5
                circumference += diameter

            x1, y1 = self.points[-1]
            x2, y2 = self.points[0]
            diameter = ((x2 - x1) ** 2 + (y2 - y1) ** 2) ** 0.5
            circumference += diameter

            cm_per_pixel = 0.45
            self.circumference_cm = circumference * cm_per_pixel
            self.ids.circumference_label.text = f"Lingkar dada:
{self.circumference_cm:.2f} Cm"
```

Gambar menghitung lingkar dada

```
from kivymd.app import MDApp
from kivymd.uix.screen import MDScreen
from kivy.lang.builder import Builder
from kivy.uix.label import Label
from kivy.properties import NumericProperty

class Rumus(MDScreen):
    lingkar_dada = NumericProperty()
    panjang_badan = NumericProperty()

    def __init__(self, **kwargs):
        Builder.load_file("kv/rumus.kv")
        super().__init__(**kwargs)

    def on_pre_enter(self, *args):
        lingkardada =
self.manager.get_screen("lingkardada")
        circumference_cm = lingkardada.circumference_cm
        self.ids.circumference_label.text = "Lingkar dada :
{:.2f} Cm".format(circumference_cm)

        panjangsapi =
self.manager.get_screen("panjangsapi")
        length_cm = panjangsapi.length_cm
        self.ids. length_label.text = "Panjang sapi :
{:.2f} Cm".format(length_cm)

    def hitung_bobot_sapi(self):
        bb = round(((self.lingkar_dada ** 2) +
(self.panjang_badan ** 2)) / 10815.15, 2)
        self.ids.bobot_sapi.text = "Berat Badan Sapi : {}
kg".format(str(bb).replace('.', ''))



    def set_lingkar_dada(self, value):
        if value:
            self.lingkar_dada = float(value)
        else:
            self.lingkar_dada = 0.0

    def set_panjang_badan(self, value):
        if value:
            self.panjang_badan = float(value)
        else:
            self.panjang_badan = 0.0
```

Gambar kode program perhitungan penaksiran

DOKUMENTASI PENGAMBILAN SEMPEL DATA SAPI

Gambar pengukuran panjang badan

Gambar pengukuran panjang badan

QUEEN FA
Breeding &
Jl. Sawo RT 1/ RW 4 Sum
HP. 081