LAPORAN PROJECT
PEMROGRAMAN VISUAL
SADAR KULIT

Disusun Oleh :

MUHAMMAD RIZKI ASSAMSULI

F1D022146

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MATARAM

2025

A. Deskripsi Aplikasi

SadarKulit aplikasi desktop yang dirancang untuk meningkatkan kesadaran masyarakat terhadap kesehatan kulit. Aplikasi ini memungkinkan pengguna untuk melakukan deteksi dini terhadap beberapa jenis penyakit kulit hanya dengan mengunggah foto. Tujuannya adalah untuk memberikan informasi awal dan edukasi, bukan untuk menggantikan diagnosis medis profesional.

Proyek ini dilatarbelakangi oleh minimnya pengetahuan masyarakat dalam mengenali gejala awal penyakit kulit, yang seringkali menghambat penanganan yang tepat. Dengan memanfaatkan teknologi Machine Learning, SadarKulit menyediakan alat bantu yang mudah diakses untuk identifikasi awal, lengkap dengan informasi mengenai penyebab dan saran penanganan dasar.

B. Stack List

Front-End:

- Python (PyQt5)

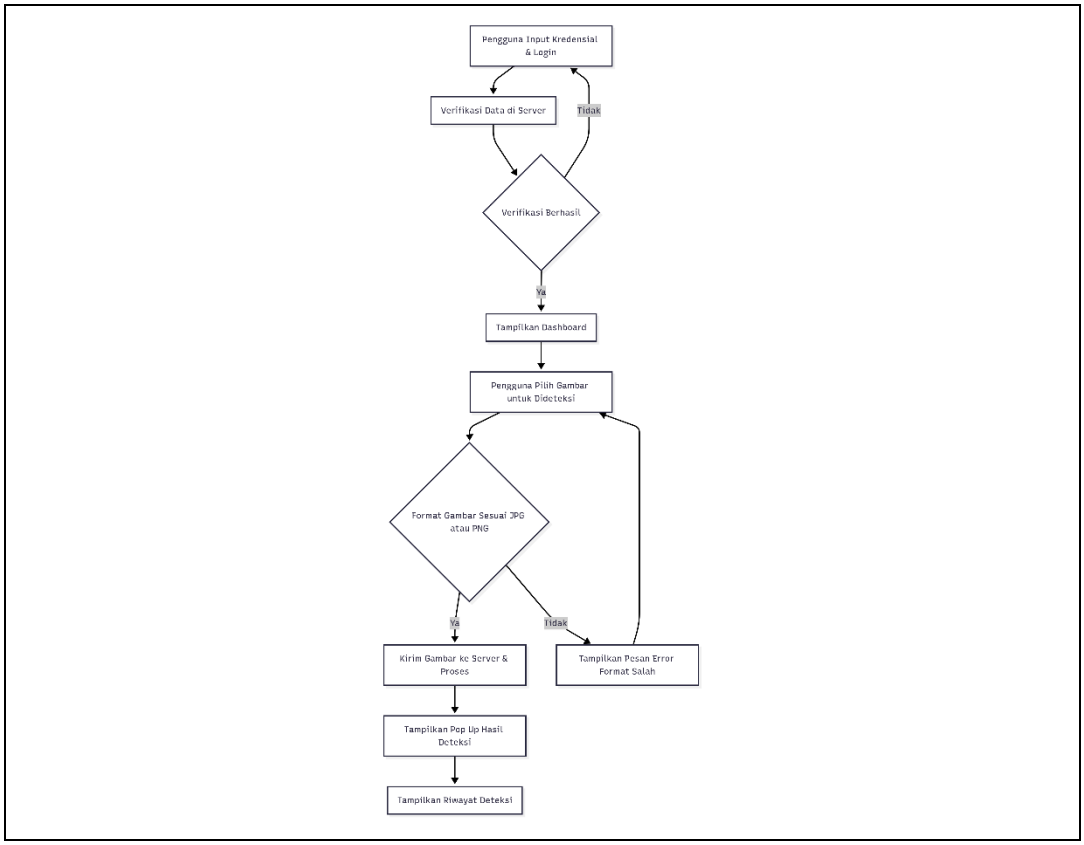
Back-End:

- Node.js
- Express.js

Database:

- MongoDB

C. Flow Aplikasi



D. Code

dashboard_widget.py

```
# Load History
    def load_history(self):
        try:
            response = requests.get(f"{API_URL}/history",
headers=self.api_headers)
            response.raise_for_status()
            self.history_data = response.json()

            print("=== DATA YANG DITERIMA DARI BACKEND ===")
            for item in self.history_data:
                print(item)
            print("=======================================")

            self.populate_table()
        except requests.exceptions.RequestException as e:
            QMessageBox.critical(self, 'Error', f"Gagal memuat
riwayat:\n{e}")
. . . . .
```

```
def perform_prediction(self, file_path):
        loading_msg = QMessageBox()
        loading_msg.setWindowTitle("Info")
        loading_msg.setText("Sedang mengirim gambar dan
melakukan prediksi...")
        loading_msg.setStandardButtons(QMessageBox.NoButton
)
        loading_msg.show()
        QApplication.processEvents()

        try:
            mime_type, _ = mimetypes.guess_type(file_path)
            if not mime_type:
                mime_type = 'application/octet-stream'

            with open(file_path, 'rb') as image_file:
                files = {
                    'image': (os.path.basename(file_path),
image_file, mime_type)
                }
                response = requests.post(
```

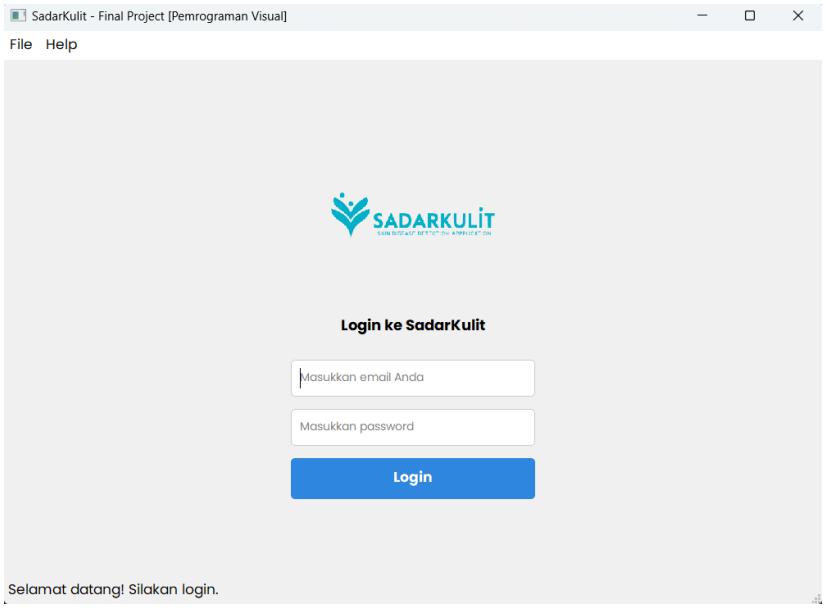
```
                        f"{API_URL}/predict",
                        headers=self.api_headers,
                        files=files
                    )
                    response.raise_for_status()

                    result = response.json()
                    loading_msg.close()

                    disease_name = (
                        result.get('detectedDisease') or
                        result.get('predicted_disease') or
                        'Tidak diketahui'
                    )
                    disease_name =
bersihkan_nama_penyakit(disease_name)
                    confidence = result.get('confidence',
'N/A')

                    QMessageBox.information(
                        self, "Hasil Prediksi",
                        f"Penyakit yang
terdeteksi:\n\n{disease_name}\n\nTingkat Keyakinan:
{confidence}"
                    )
                    self.load_history()
. . . . .
```

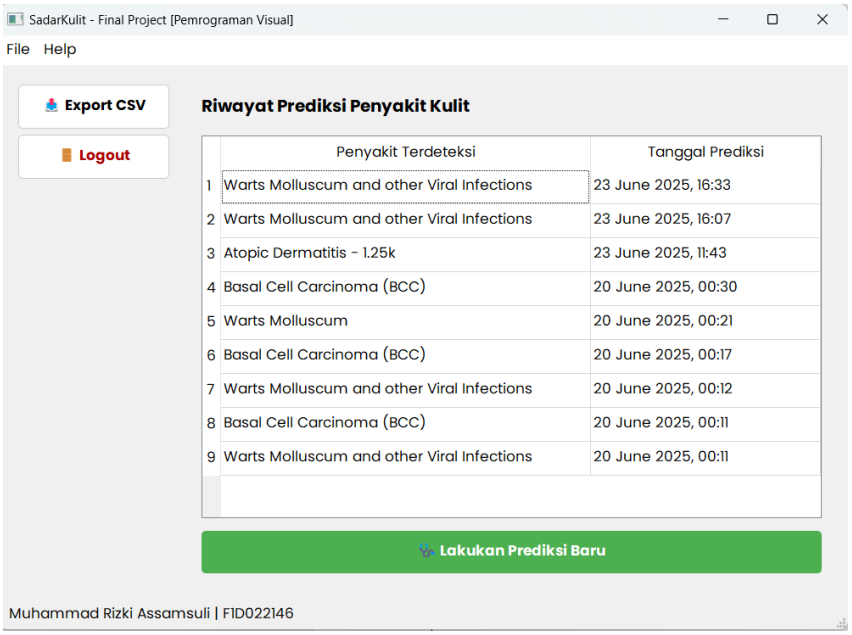
Login.py

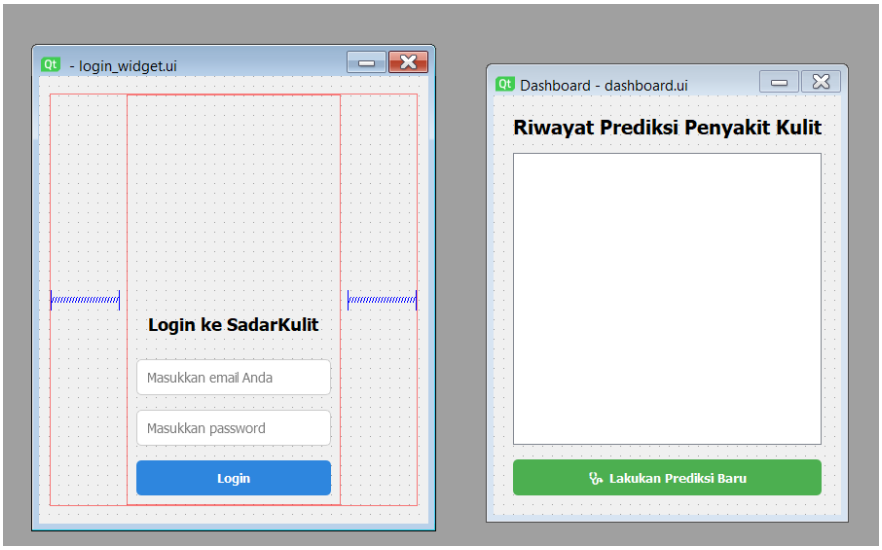
```
class LoginWidget(QWidget):
    login_successful = pyqtSignal(str, dict)

    def __init__(self, parent=None):
        super().__init__(parent)
        loadUi("ui/login_widget.ui", self)  # Pastikan path
benar

         # Tampilkan logo dari file
        pixmap = QPixmap("assets/images/logo.png")
        self.logoLabel.setPixmap(pixmap.scaled(200, 200,
aspectRatioMode=1))  # 100x100 dan preserve rasio
        self.logoLabel.setStyleSheet("margin-bottom:
10px;")

        # Event binding
        self.loginButton.clicked.connect(self.attempt_login
)
```

```
            self.passwordInput.returnPressed.connect(self.attem
    pt_login)

        def attempt_login(self):
            email = self.emailInput.text().strip()
            password = self.passwordInput.text().strip()

            if not email or not password:
                QMessageBox.warning(self, 'Input Kosong',
    'Email dan password tidak boleh kosong.')
                return

            try:
                response = requests.post(
                    f"{API_URL}/auth/login",
                    json={'email': email, 'password': password}
                )
                response.raise_for_status()
                data = response.json()

                token = data.get('token')
                user_data = data.get('user')
                if token and user_data:
                    self.login_successful.emit(token,
    user_data)
                else:
                    QMessageBox.warning(self, 'Login Gagal',
    'Respons dari server tidak valid.')

            except requests.exceptions.RequestException as e:
                error_message = "Email atau password salah."
                if e.response:
                    try:
                        backend_error =
    e.response.json().get('error')
                        if backend_error:
                            error_message = backend_error
                    except Exception:
                        pass
                else:
                    error_message = f"Gagal terhubung ke
    server.\n\n{e}"

                QMessageBox.critical(self, 'Login Error',
    error_message)
```

E. Hasil Aplikasi






- login_widget.ui
Login ke SadarKulit
Masukkan email Anda
Masukkan password
Login
Dashboard - dashboard.ui
Riwayat Prediksi Penyakit Kulit
Lakukan Prediksi Baru