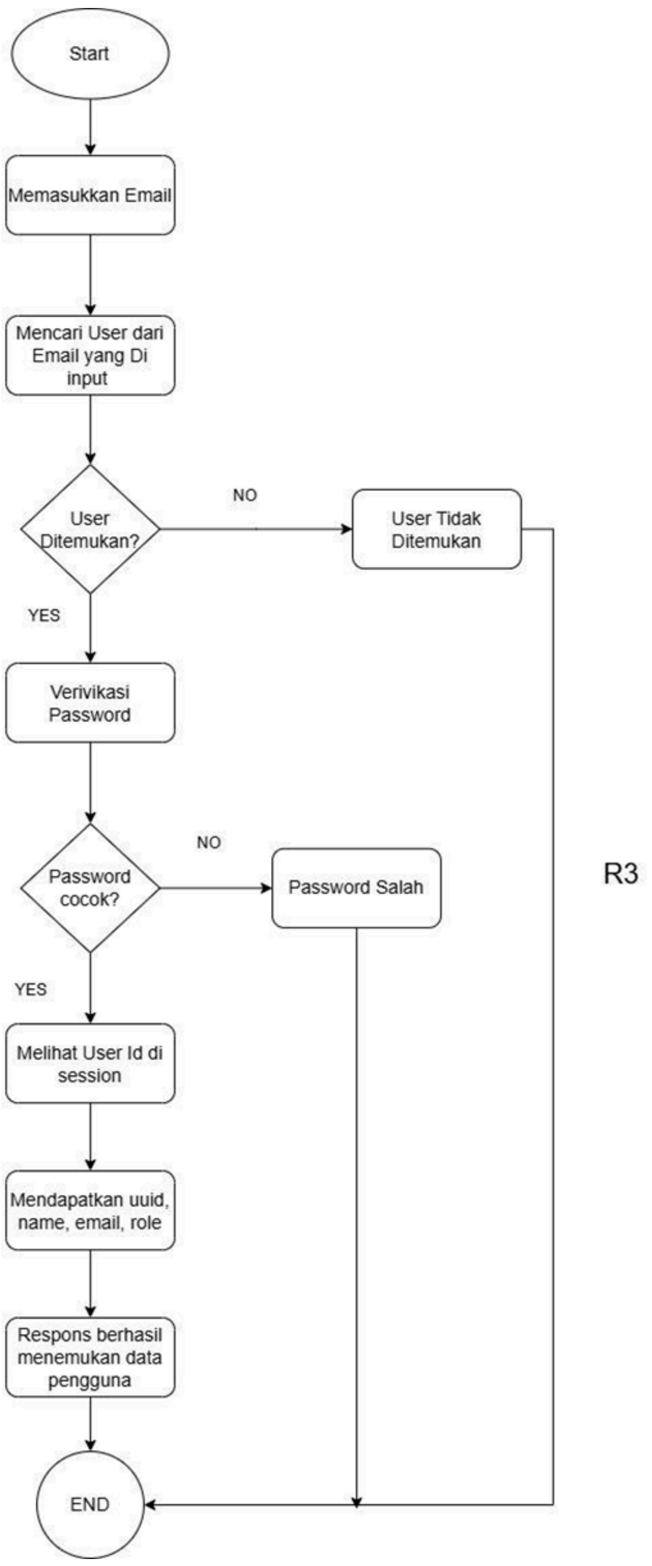
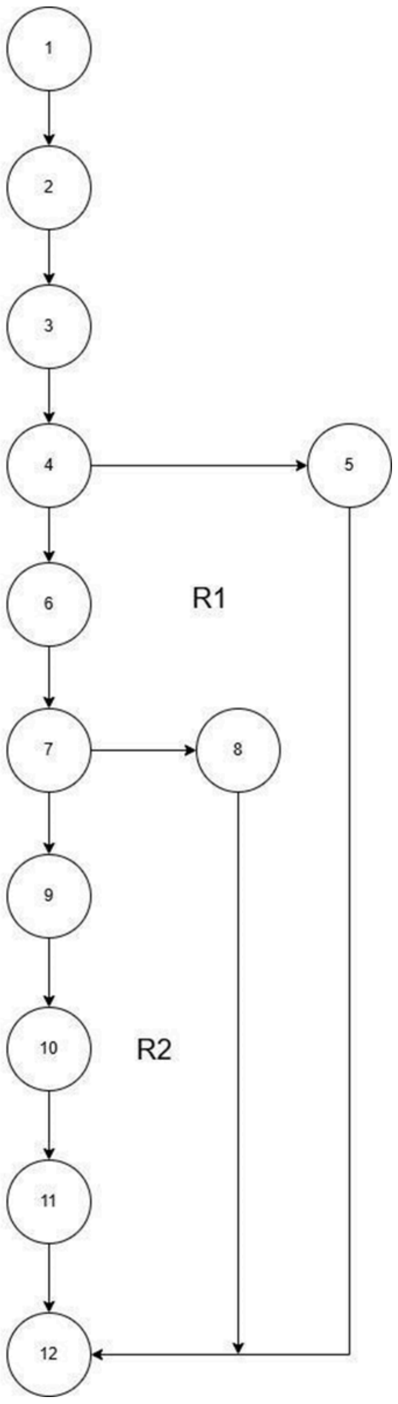
Pengujian White Box Project Posyandu

Login

```
export const Login = async (req, res) => {
  const users = await Users.findOne({
    where: {
      email: req.body.email
    }
  });
  if(!users) return res.status(404).json({msg: "User tidak ditemukan"});
  const match = await argon2.verify(users.password, req.body.password);
  if(!match) return res.status(400).json({msg: "Password salah"});
  req.session.userId = users.uuid;
  const uuid = users.uuid;
  const name = users.name;
  const email = users.email;
  const role = users.role;
  res.status(200).json({uuid, name, email, role});
}
```

Flowchart & FlowGraph Login


Start
Memasukkan Email
Mencari User dari Email yang Di input
User Ditemukan?
NO
User Tidak Ditemukan
YES
Verivikasi Password
Password cocok?
NO
Password Salah
YES
Melihat User Id di session
Mendapatkan uuid, name, email, role
Respons berhasil menemukan data pengguna
END
R3



Region	:	**3**
Edge	:	**13**
Node	:	**12**
Simpul Predikat	:	**2**
Independent path	:	**3**
		1-2-3-4-6-7-9-10-11-12
		1-2-3-4-6-7-8-12
		1-2-3-4-5-12

Kompleksitas Siklomatis : **V(G) = E - N + 2**

13 - 12 + 2 = 3

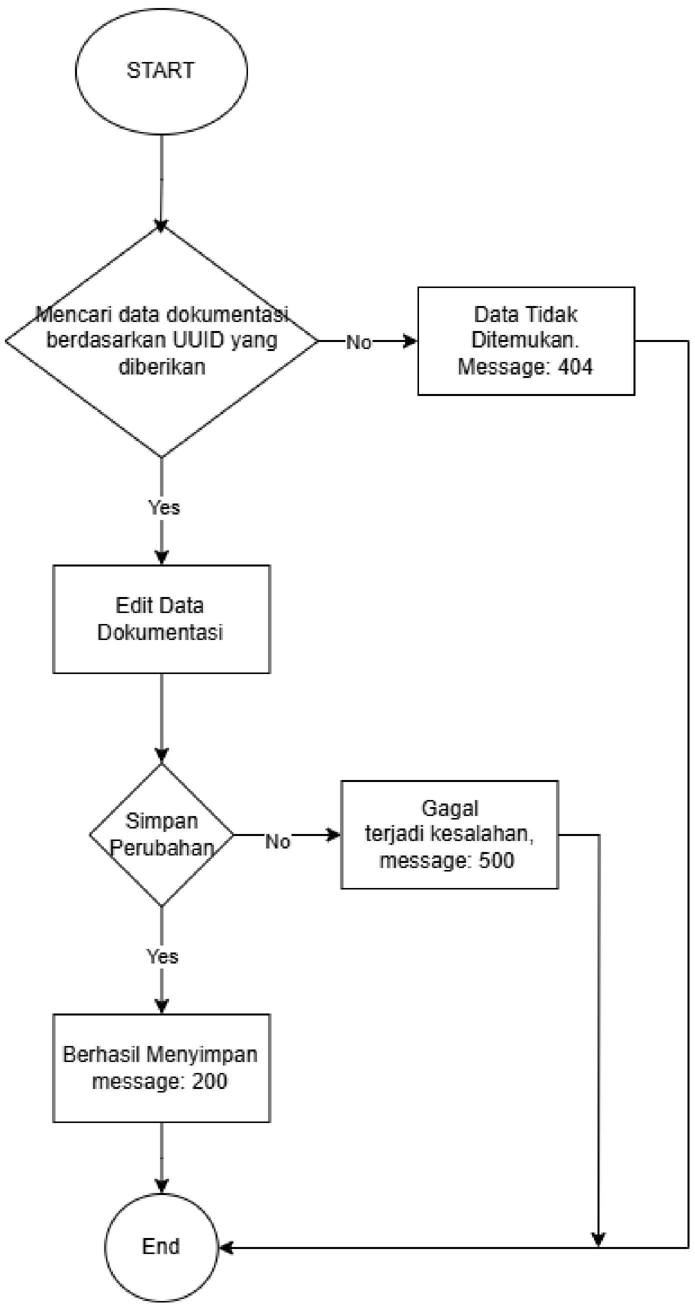
V(G) = P + 1

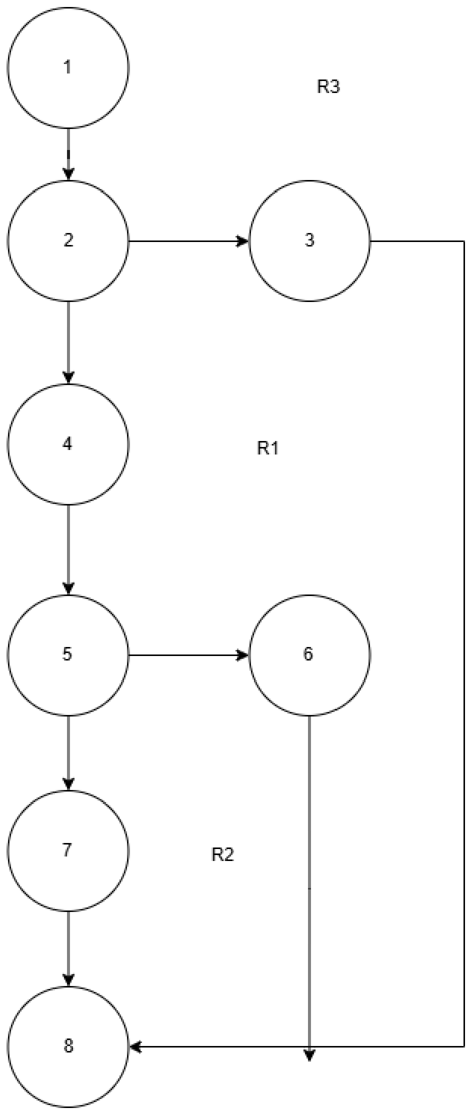
2 + 1 = 3

Update Dokumentasi

```
export const updateDokumentasi = async (req, res) =>{
  try {
    const dokumentasi = await Dokumentasi.findOne({
      where: {
        uuid: req.params.id
      }
    });
    if (!dokumentasi) return res.status(404).json({ msg: "Data tidak ditemukan" });

    const { judul, keterangan } = req.body;

    // Handle image update
    if (req.file) {
      const oldImagePath = path.join(process.cwd(), 'uploads', 'dokumentasi', dokumentasi.image);
      if (fs.existsSync(oldImagePath)) {
        fs.unlinkSync(oldImagePath); // Hapus gambar lama
      }
      dokumentasi.image = req.file.filename; // Simpan nama file baru
    }

    // Update data dokumentasi
    dokumentasi.judul = judul;
    dokumentasi.keterangan = keterangan;

    await dokumentasi.save(); // Simpan perubahan ke database

    res.status(200).json({ msg: "Data Dokumentasi berhasil diperbarui" });
  } catch (error) {
    console.error("Error in updateDokumentasi:", error);
    res.status(500).json({ msg: error.message });
  }
}
```

Flowchart & FlowGraph Update Dokumentasi Staff


START
Mencari data dokumentasi berdasarkan UUID yang diberikan
No
Data Tidak Ditemukan. Message: 404
Yes
Edit Data Dokumentasi
Simpan Perubahan
No
Gagal terjadi kesalahan, message: 500
Yes
Berhasil Menyimpan message: 200
End



N = 8 **Simpul Predikat = 2**

E = 9 **Independent Path = 3 (1- 2-3-8), (1-2-4-5-6-8), (1-2-4-5-7-8)**

R = 3

Kompleksitas Siklomatis = V(G) = E - N + 2

9 - 8 + 2 = 3

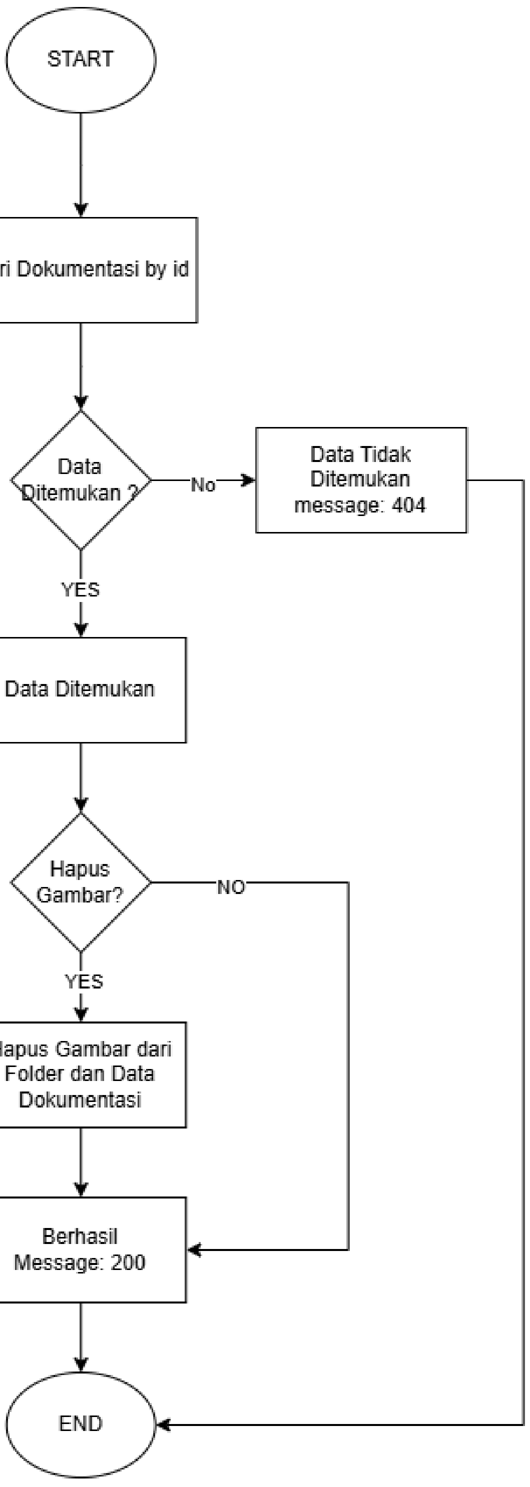
= V(G) = P + 1

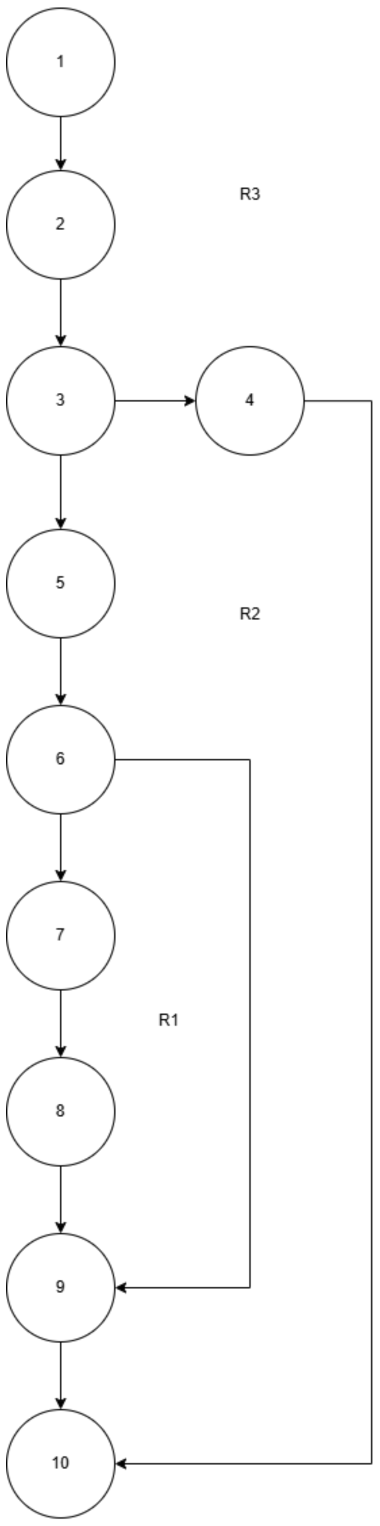
2 + 1

Delete Dokumentasi

```
export const deleteDokumentasi = async (req, res) =>{
  try {
    const dokumentasi = await Dokumentasi.findOne({
      where: {
        uuid: req.params.id
      }
    });

    if (!dokumentasi) {
      console.error("Data not found");
      return res.status(404).json({ msg: "Data tidak ditemukan" });
    }

    // Hapus file gambar dari folder uploads
    const imagePath = path.join(process.cwd(), 'dokumentasi', dokumentasi.image);
    if (fs.existsSync(imagePath)) {
      fs.unlink(imagePath, (err) => {
        if (err) {
          console.error(`Error deleting image ${dokumentasi.image}: ${err}`);
        } else {
          console.log(`Deleted image ${dokumentasi.image}`);
        }
      });
    }

    await Dokumentasi.destroy({
      where: {
        uuid: req.params.id,
      }
    });

    res.status(200).json({ msg: "Dokumentasi deleted successfully" });
  } catch (error) {
    console.error("Error in delete Dokumentasi:", error);
    res.status(500).json({ msg: error.message });
  }
}
```

Flowchart & Flowgraph Delete Dokumentasi Staff





N = 9 **Simpul Predikat = 2**

E = 10 **Independent Path = 3 (1- 2-3-4-10), (1-2-3-5-6-9-10), (1-2-3-5-6-7-8-9-10)**

R = 3

Kompleksitas Siklomatis = V(G) = E - N + 2

$$10 - 9 + 2 = 3$$

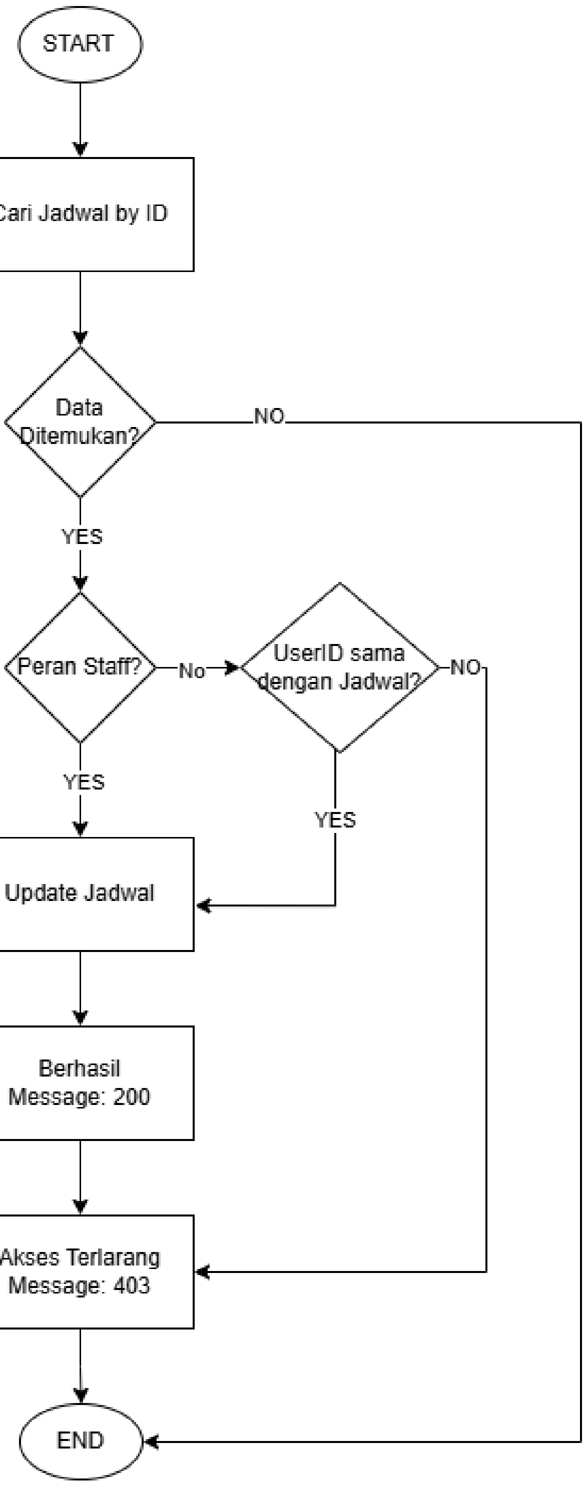
$$= V(G) = P + 1$$

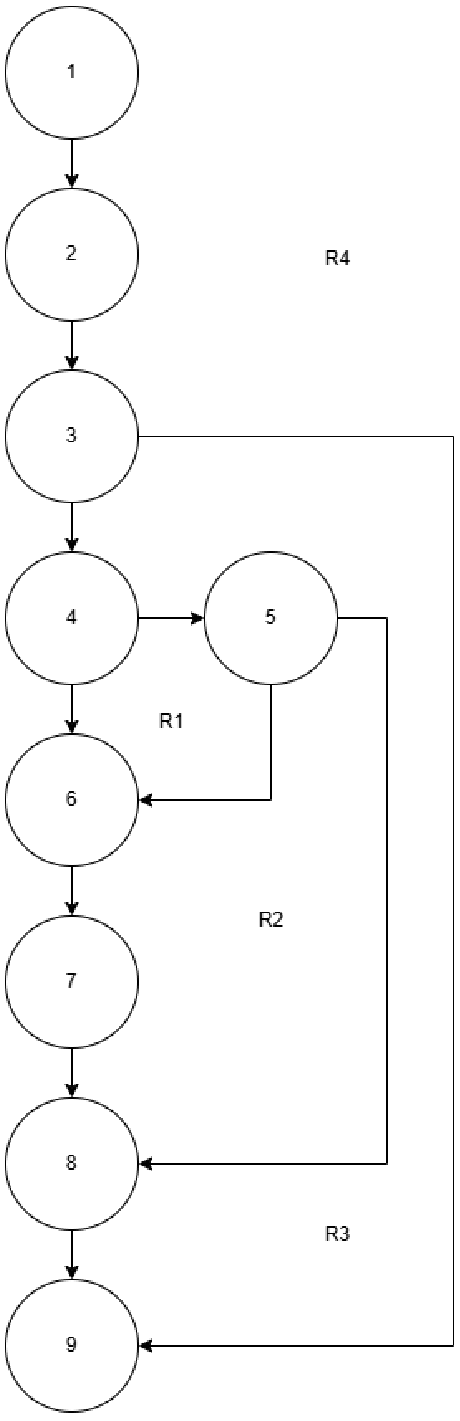
$$2 + 1 = 3$$

Update Jadwal

```
export const updateJadwal = async (req, res) =>{
  try {
    const Jadwal = await JadwalPelayanan.findOne({
      where:{
        uuid: req.params.id
      }
    });
    if(!Jadwal) return res.status(404).json({msg: "Data tidak ditemukan"});
    const {rw, kategori, jadwal} = req.body;
    if(req.role === "admin"){
      await JadwalPelayanan.update({rw, kategori, jadwal}, {
        where:{
          id: Jadwal.id
        }
      });
    }else{
      if(req.userId != Jadwal.userId) return res.status(403).json({msg: "Akses terlarang"});
      await JadwalPelayanan.update({rw, kategori, jadwal}, {
        where:{
          [Op.and]:[{id: Jadwal.id}, {userId: req.userId}]
        },
      });
    }
    res.status(200).json({msg: "Jadwal updated successfully"});
  } catch (error) {
    res.status(500).json({msg: error.message});
  }
}
```

Flowchart & FlowGraph Update Jadwal





N = 9 **Simpul Predikat = 3**

E = 11 **Independent Path = 4 (1- 2-3-9), (1-2-3-4-5-8-9), (1-2-3-4-5-6-7-8-9),(1,2,3,4,6,7,8,9)**

R = 4

Kompleksitas Siklomatis = V(G) = E - N + 2

11 - 9 + 2 = 4

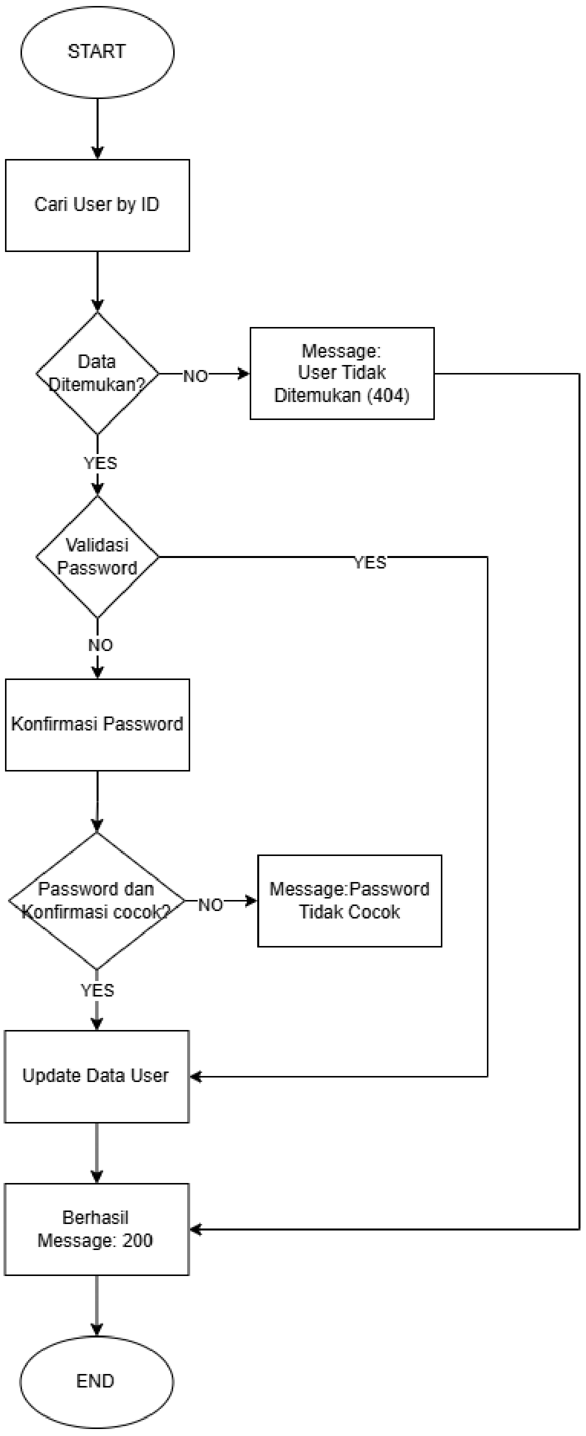
= V(G) = P + 1

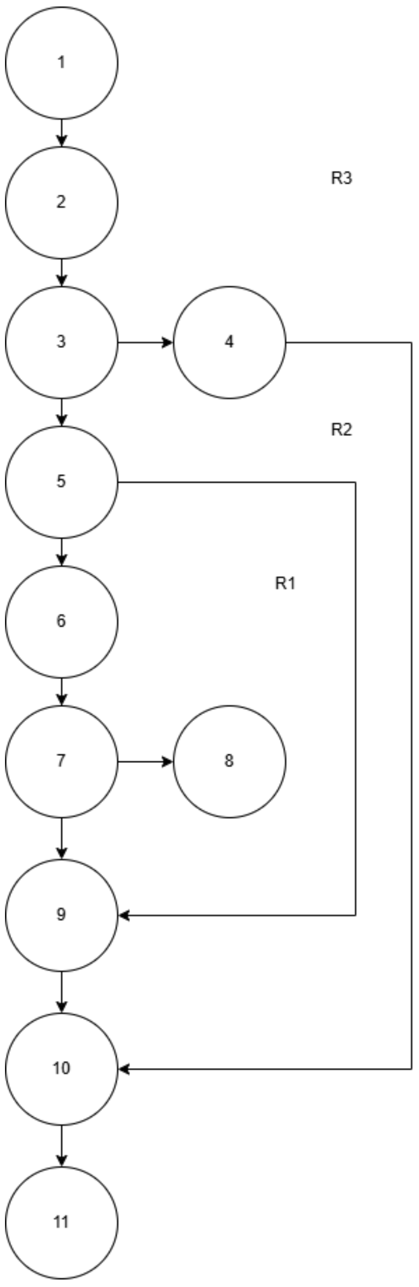
3 + 1 = 4

Update User

```
export const updateUsers = async (req, res) =>{
    const users = await Users.findOne({
        where: {
            uuid: req.params.id
        }
    });
    if(!users) return res.status(404).json({msg: "User tidak ditemukan"});
    const {name, email, password, confPassword, role} = req.body;
    let hashPassword;
    if(password === "" || password === null){
        hashPassword = users.password
    }else{
        hashPassword = await argon2.hash(password);
    }
    if(password !== confPassword) return res.status(400).json({msg: "Password dan Confirm Password tidak cocok"});
    try {
        await Users.update({
            name: name,
            email: email,
            password: hashPassword,
            role: role
        },{
            where:{
                id: users.id
            }
        });
        res.status(200).json({msg: "User Updated"});
    } catch (error) {
        res.status(400).json({msg: error.message});
    }
}
```

Flowchart & FlowGraph Update User





N = 11 **Simpul Predikat = 3**

E = 12 **Independent Path = 4 (1- 2-3-4-10-11), (1-2-3-5-9-10-11), (1-2-3-5-6-7-8),**

(1,2,3,5,6,7,9,10,11)

R = 3

Kompleksitas Siklomatis = V(G) = E - N + 2

12 - 11 + 2 = 3

= V(G) = P + 1

3 + 1 = 4