

**Tugas Besar
Praktikum Kecerdasan Buatan**

**SISTEM PENYIRAMAN
TANAMAN OTOMATIS**

- Presented by: Kelompok 3

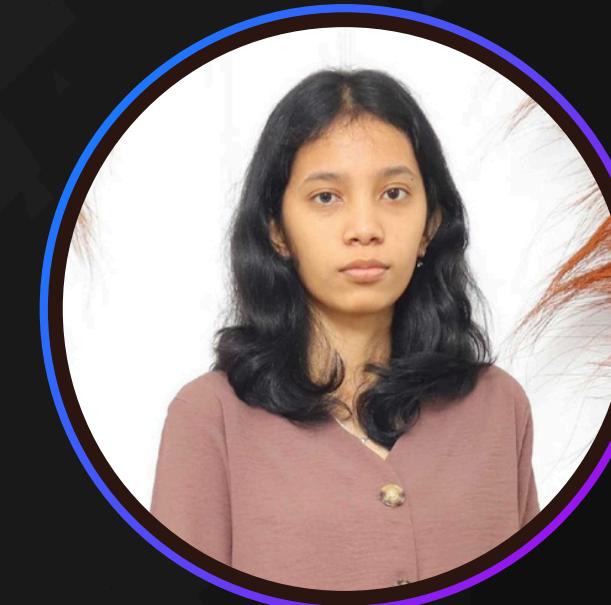
MEMBERS



Yohana Septamia
22-056



Keisyia
22-006



Melia Purnamasari
22-112



Yohana Marbun
22-065



Armila Sakinah
22-018

Overview

- Latar Belakang
- Tujuan
- Logic



Latar Belakang

Dalam menghadapi masalah perubahan iklim, efisiensi penggunaan air menjadi hal yang penting untuk diperhatikan, terutama selama masa musim kemarau. Sektor yang sangat bergantung pada ketersediaan air salah satunya adalah sektor pertanian. Metode penyiraman manual untuk mengairi lahan sering kali menyebabkan pemborosan air bagi para petani.

Oleh sebab itu, kami memiliki ide untuk mengembangkan sistem penyiraman tanaman otomatis ini. Sistem ini dapat mengetahui kapan tanaman harus disiram dan berapa banyak air yang diperlukan berdasarkan data yang diperoleh, sehingga diharapkan dapat membantu para petani dalam mengelola penggunaan air untuk perairan lahan mereka secara lebih efisien.

Proyek ini menunjukkan bahwa teknologi dapat menjadi solusi bagi para petani yang dapat membantu mereka dalam menghadapi masalah perairan lahan pada musim kemarau yang semakin tidak menentu.



Tujuan

Sistem ini dimaksudkan untuk membantu para petani dalam mengoptimalkan penggunaan air untuk pengairan lahan mereka dengan menghitung jumlah air yang diperlukan berdasarkan kondisi tanaman dan lingkungan pada lahan.

Harapannya, sistem penyiraman otomatis ini akan menjadi alat yang berguna bagi para petani sehingga pengelolaan sumber daya air mereka dapat dilakukan secara lebih efisien.

**Logic yang digunakan
pada sistem**

Apakah tanaman perlu disiram?

```
% apakah tanaman perlu disiram?  
:- dynamic tanah_kering/0.  
  
tanaman_perlu_disiram :- tanah_kering.  
tanaman_tidak_perlu_disiram :- not(tanah_kering).  
  
cek_kondisi_tanah(Kondisi) :- retract(tanah_kering), read(Kondisi), assert(Kondisi),  
|   |   |   |   |   (tanaman_perlu_disiram -> pump_on ; pump_off).  
  
pump_on :- (tanaman_tidak_perlu_disiram -> fail ; write(pump_on), nl, sleep(2), cek_kondisi_tanah(_)).  
pump_off :- tanaman_tidak_perlu_disiram.  
  
action :- (tanaman_perlu_disiram -> pump_on ; fail).
```

Rule untuk menentukan apakah tanaman perlu disiram berdasarkan kondisi tanah (kering/tidak)



Jenis penyiraman yang digunakan

```
% jenis irigasi apa yang cocok digunakan
:- op(200, xf, pilihanTumbuhan).
cabai pilihanTumbuhan.
kol pilihanTumbuhan.

:- op(800, xfx, jenis_penyiraman).
cabai jenis_penyiraman sprinkle.
kol jenis_penyiraman drip.

tentukan_jenis_penyiraman :- write("Tumbuhan: "), read(TumbuhanTerpilih),
| | | | | (pilihanTumbuhan(TumbuhanTerpilih) ->
| | | | | TumbuhanTerpilih jenis_penyiraman Apa, format('Jenis penyiraman: ~w.~n', [Apa]) ;
| | | | | fail).
```

Rule untuk menentukan jenis irigasi yang cocok digunakan berdasarkan jenis tanaman(cabai/kol) yang dipilih

Cek kecukupan air pada kontainer

```
% cek apakah air di container cukup untuk mengairi lahan
:- dynamic input_vol_container/1.
:- dynamic input_vol_1_irigasi/1.
:- dynamic input_jlh_irigasi/1.

check_if_exist(input_vol_container(VolumeContainer)) :- input_vol_container(VolumeContainer), VolumeContainer > 0.
check_if_exist(input_vol_1_irigasi(Volume1Irigasi)) :- input_vol_1_irigasi(Volume1Irigasi), Volume1Irigasi > 0.
check_if_exist(input_jlh_irigasi(JumlahIrigasi)) :- input_jlh_irigasi(JumlahIrigasi), JumlahIrigasi > 0.
```

Rule di atas akan memvalidasi inputan untuk digunakan pada rule selanjutnya.

Lanjutan rules

```
% base case
rule_cek_kecukupan_air(_, JumlahIrigasi, JumlahIrigasi, _, Hasil) :- !, Hasil='Cukup', write(Hasil).

% base case
rule_cek_kecukupan_air(Volume1Irigasi, JumlahIrigasi, IrigasiYangSudahDiAiri, VolumeContainer, Hasil) :-
    VolumeContainer < Volume1Irigasi, JumlahIrigasi =\= IrigasiYangSudahDiAiri, !,
    Hasil='Tidak Cukup', write(Hasil),
    KurangBrgpLiter is (Volume1Irigasi*(JumlahIrigasi-IrigasiYangSudahDiAiri)-VolumeContainer),
    nl, format('Kurang ~w liter', [KurangBrgpLiter]).

rule_cek_kecukupan_air(Volume1Irigasi, JumlahIrigasi, JumlahIrigasi1, VolumeContainer, Hasil) :-
    VolumeContainer >= Volume1Irigasi,
    SisaAir is VolumeContainer - Volume1Irigasi,
    IrigasiYangSudahDiAiri is JumlahIrigasi1 + 1,
    rule_cek_kecukupan_air(Volume1Irigasi, JumlahIrigasi, IrigasiYangSudahDiAiri, SisaAir, Hasil).

cek_kecukupan_air :- check_if_exist(input_vol_container(VolumeContainer)), check_if_exist(input_vol_1_irigasi(Volume1Irigasi)),
    check_if_exist(input_jlh_irigasi(JumlahIrigasi)),
    rule_cek_kecukupan_air(Volume1Irigasi, JumlahIrigasi, 0, VolumeContainer, _).
```

Rule untuk mengecek apakah volume air dalam kontainer cukup untuk melakukan irigasi berdasarkan volume air yang dibutuhkan untuk irigasi perbarisnya dan banyaknya total irigasi yang ada.

Banyak air yang dibutuhkan

```
cek_banyak_air_diperlukan :-  
    | | | | ((input_suhu(Suhu), input_kelembaban(Kelembaban), input_pH_tanah(Ph_tanah)) ->  
    | | | | (tingkat_kebutuhan_air(Suhu, Kelembaban, Ph_tanah, KebutuhanAir), jumlah_air_diperlukan(KebutuhanAir, JumlahAir),  
    | | | | format('Tingkat kebutuhan air: ~w.~nBanyak air diperlukan: ~w liter.~n', [KebutuhanAir, JumlahAir]),  
    | | | | tentukan_jenis_penyiraman) ; write("Butuh inputan!"), fail).
```

Rule untuk mengetahui jumlah air yang diperlukan untuk melakukan irigasi berdasarkan keadaan suhu, kelembaban, dan pH tanah pada lahan.



Lanjutan rules

```
% berapa banyak air yang dibutuhkan
:-dynamic input_suhu/1.
:-dynamic input_kelembaban/1.
:-dynamic input_pH_tanah/1.

tingkat_kebutuhan_air(Suhu, Kelembaban, Ph_tanah, KebutuhanAir) :-
    (Suhu > 35 -> SkorSuhu is 3;
     (Suhu >= 25, Suhu <= 35) -> SkorSuhu is 2;
     Suhu < 25 -> SkorSuhu is 1),
    (Kelembaban > 75 -> SkorKelembaban is 3;
     (Kelembaban >= 50, Kelembaban <= 75) -> SkorKelembaban is 2;
     Kelembaban < 50 -> SkorKelembaban is 1),
    (Ph_tanah > 7 -> Skor_pH_tanah is 3;
     (Ph_tanah >= 6, Ph_tanah <= 7) -> Skor_pH_tanah is 2;
     Ph_tanah < 6 -> Skor_pH_tanah is 1),
    TotalScore is SkorSuhu + SkorKelembaban + Skor_pH_tanah,
    (TotalScore >= 7 -> KebutuhanAir = 'high';
     (TotalScore >= 6, TotalScore < 7) -> KebutuhanAir = 'mid';
     TotalScore < 6 -> KebutuhanAir = 'low').

jumlah_air_diperlukan(KebutuhanAir, JumlahAir) :- (KebutuhanAir = 'low' -> Val is 1500 ;
    KebutuhanAir = 'mid' -> Val is 3000 ;
    KebutuhanAir = 'high' -> Val is 4000),
    (check_if_exist(input_vol_1_irigasi(_)) -> input_vol_1_irigasi(Volume1Irigasi) ; Volume1Irigasi is 1),
    (check_if_exist(input_jlh_irigasi(_)) -> input_jlh_irigasi(JumlahIrigasi) ; JumlahIrigasi is 1),
    JumlahAir is Val * Volume1Irigasi * JumlahIrigasi.
```



DEMO

Thank You