Sistem Penjadwalan Kegiatan Harian yang Fleksibel Menggunakan Representasi Himpunan, Fungsi, dan Relasi Graf

Ishak Palentino Napitupulu - 13524022
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
ishaknapitupulucxr2@gmail.com , 13524022@std.stei.itb.ac.id

***Abstrak*—Makalah ini menyajikan rancangan sistem penjadwalan aktivitas harian yang didasarkan pada konsep himpunan, fungsi, dan relasi graf dari matematika diskrit. Setiap tugas direpresentasikan sebagai blok waktu dengan durasi yang telah ditentukan, dan secara otomatis didistribusikan ke dalam tampilan kalender tujuh hari berdasarkan parameter masukan seperti jenis tugas, total waktu yang dibutuhkan (bobot), dan tenggat waktu. Sistem menghasilkan beberapa opsi distribusi berbasis seed, memungkinkan pengguna untuk memilih alokasi waktu yang paling sesuai sebelum jadwal difinalisasi. Selain itu, sistem mendukung jadwal harian/mingguan rutin, pengecualian waktu bebas yang ditentukan pengguna, dan interaksi visual dengan kalender serta blok tugas. Pendekatan ini menunjukkan penerapan praktis dari logika matematika diskrit dalam mengelola waktu secara efektif.**

Kata Kunci*—Sistem penjadwalan; teori himpunan; relasi; fungsi; kombinatorika; graf; manajemen waktu; matematika diskrit

I. PENDAHULUAN

Manajemen waktu merupakan salah satu *soft skill* yang sangat penting dalam kehidupan ini, khususnya bagi mahasiswa yang memiliki kegiatan akademik dan non-akademik yang beragam dan cukup *demanding*. Untuk dapat menjalani perkuliahan dengan lancar, mahasiswa harus dapat menyeimbangkan tuntutan akademik maupun non-akademik dengan efektif, efisien, dan juga fleksibel. Namun, hal ini tentu saja menjadi tantangan yang tidak mudah bagi mahasiswa, khususnya mahasiswa tahun pertama. Tidak semua mahasiswa tahun pertama dapat mengatur waktu mereka dengan efektif karena banyak di antara mereka mengalami kesulitan dalam beradaptasi dengan lingkungan kampus mengingat bahwa kehidupan di kampus sangat berbeda ketika masih berada di pendidikan menengah atas.

Untuk mengatasi hal tersebut, dibutuhkan suatu sistem pendukung yang dapat membimbing mahasiswa untuk mengatur waktu mereka secara efisien dan sekonsisten mungkin. Sistem juga harus dapat memenuhi batasan atau preferensi dari pengguna. Hal ini bertujuan agar pengguna tidak merasa ‘terkejut’ akibat gaya kehidupan yang berubah drastis secara langsung.

Makalah ini memperkenalkan sistem penjadwalan (*scheduler*) harian yang didesain dengan pendekatan matematika diskrit, yaitu konsep himpunan, fungsi, relasi kombinatorika, dan graf. Pengguna dapat meng-*input* beberapa kegiatan – kegiatannya yang diasosiasikan dengan bobot (*weight*) berupa total waktu pengerjaan beserta waktu tenggat pekerjaannya. Kemudian program akan mendistribusikan kegiatan kegiatan ini berupa blok kegiatan ke dalam slot waktu pada kalender dinamis. Penjadwalan dipertimbangkan dengan berbagai kemungkinan distribusi waktu (*seed*) yang kemudian disarankan kepada pengguna untuk kemudian dipilih secara interaktif. Hal ini bertujuan untuk meningkatkan fleksibilitas pengguna dalam penyesuaian jadwal berdasarkan situasi dan kondisi pengguna.

Sistem juga menyediakan berbagai fitur – fitur pelengkap lainnya, seperti waktu bebas (*free time*) yang memungkinkan pengguna untuk menentukan interval-interval waktut tanpa penjadwalan kegiatan, serta jadwal kegiatan periodik yang merupakan kegiatan-kegiatan yang pasti dilakukan pengguna dalam periode waktu tertentu, seperti berkuliah, berolahraga, atau pun makan.

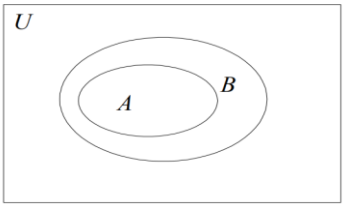
Oleh karena itu, dengan memanfaatkan prinsip-prinsip dasar matematika diskrit, sistem ini bukan hanya menjadi alat bantu semata saja, melainkan juga sebagai contoh nyata penerapan teori matematika dalam kehidupan nyata yang dapat meningkatkan kualitas pengguna dalam mengelola watku dengan baik.

II. LANDASAN TEORI

A. Himpunan

Himpunan merupakan salah satu konsep dasar matematika diskrit yang merujuk pada kumpulan objek ak terurut dan berbeda satu sama lain. Objek di dalam himpunan disebut elemen, unsur, atau anggota.

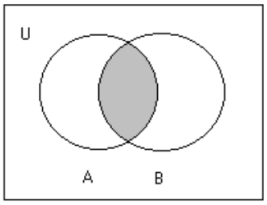
1. Himpunan Bagian (*Subset*)



Himpunan A dikatakan himpunan bagian dari himpunan B (ditulis A⊆B) jika dan hanya jika setiap elemen A merupakan elemen dari B.

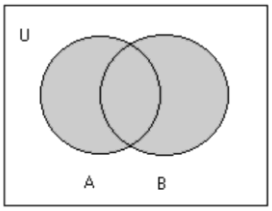
2. Operasi Dasar

- Irisan (*intersection*)



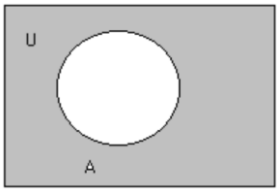
Notasi : A ∩ B = { x | x ∈ A dan x ∈ B }

- Gabungan (*Union*)



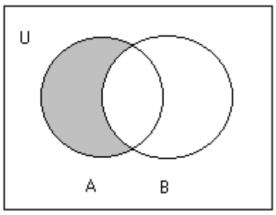
Notasi : A ∪ B = { x | x ∈ A atau x ∈ B }

- Komplemen (*complement*)



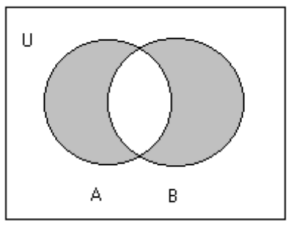
Notasi : A = { x | x ∈ U, x ∉ A }

- Selisih (*difference*)



Notasi : A – B = { x | x ∈ A dan x ∉ B } = A ∩ B

- Beda Setangkup (*Symmetric Difference*)



Notasi: A ⊕ B = (A ∪ B) – (A ∩ B) = (A – B) ∪ (B – A)

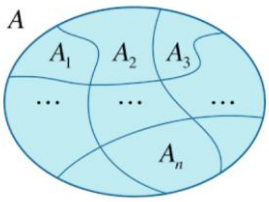
3. Himpunan Kuasa (*Power set*)

Himpunan kuasa (*Power set*) dari himpunan A adalah himpunan yang elemennya adalah semua himpunan bagian dari A

Notasi: P (A) atau 2^A

Jika |A| = m, maka $|P(A)| = 2^m$.

4. Partisi

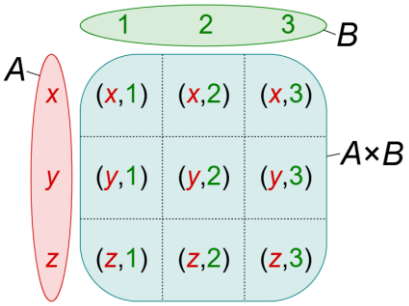


Partisi dari sebuah himpunan A adalah sekumpulan himpunan bagian tidak kosong A1 , A2 , … An dari A sedemikian sehingga: (i) A1 ∪ A2 ∪ … ∪ An = A, dan (ii) Ai ∩ Aj = ∅ untuk i ≠ j

5. Multiset

Multiset merupakan perluasan dari konsep himpunan biasa, di mana sebuah elemen dapat muncul lebih dari satu kali dalam satu kumpulan. Berbeda dengan himpunan biasa yang hanya mencatat keberadaan elemen, multiset juga memperhitungkan jumlah kemunculan (multiplisitas) dari setiap elemen.

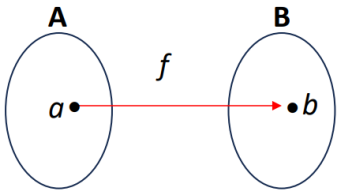
6. Perkalian Kartesian (*Cartesian Product*)



Perkalian kartesian adalah Hasil dari kombinasi semua pasangan terurut antara dua himpunan, ditulis A×B yang sangat berguna dalam pemodelan data dua dimensi atau lebih.

Notasi: $A \times B = \{(a, b) \mid a \in A \text{ dan } b \in B\}$

B. *Fungsi*



Fungsi adalah relasi khusus yang menghubungkan setiap elemen dari himpunan domain ke tepat satu elemen di kodomain.

Suatu fungsi $f : A \rightarrow B$ adalah aturan yang memetakan setiap $a \in A$ ke tepat satu $b \in B$. Himpunan semua input disebut domain, sedangkan codomain adalah himpunan tujuan, dan range adalah semua nilai output yang benar-benar tercapai.

Sifat Fungsi

- Injektif (satu-ke-satu)

  Fungsi yang memetakan setiap elemen berbeda di domain ke elemen yang berbeda di kodomain. Tidak ada dua elemen domain yang memiliki hasil yang sama di kodomain.

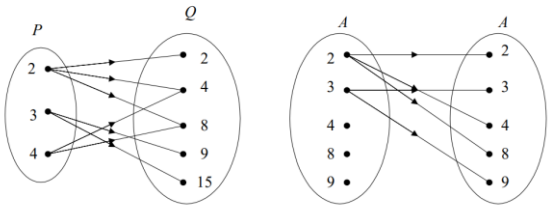
- Surjektif (Onto)

  Fungsi Surjektif (Onto)*: F*ungsi di mana setiap elemen dalam kodomain memiliki setidaknya satu elemen domain yang dipetakan kepadanya. Dengan kata lain, seluruh kodomain tercakup oleh range (hasil fungsi).

- Bijektif (berkoresponden satu-ke-satu)

  Fungsi bijektif adalah fungsi yang bersifat injektif dan surjektif secara sekaligus. Setiap elemen domain dipetakan ke elemen yang unik di kodomain, dan semua elemen kodomain tercakup. Jadi, terdapat korespondensi satu-ke-satu dan onto antara domain dan kodomain.

C. *Relasi*



Relasi adalah konsep yang lebih umum dari fungsi. Suatu relasi dari himpunan A ke himpunan B adalah subhimpunan dari hasil kali Cartesian $A \times B$, berisi pasangan terurut (a, b).

Relasi biner R antara himpunan A dan B adalah himpunan bagian dari $A \times B$.
Notasi: $R \subseteq (A \times B)$

Sifat Relasi

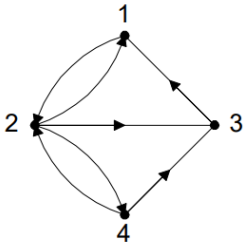
- Refleksif: (a, a) termasuk dalam relasi R untuk setiap a di A.
- Simetris: Jika (a, b) termasuk dalam R, maka (b, a) juga termasuk dalam R.
- Transitif: Jika (a, b) dan (b, c) termasuk dalam R, maka (a, c) juga harus termasuk.
- Antisimetri, irrefleksif, dan sifat lain tergantung kebutuhan sistem.

D. *Graf*

Graf adalah struktur matematika yang terdiri atas simpul (vertex) dan sisi (edge). Graf digunakan untuk merepresentasikan hubungan antar objek. Graf dinyatakan sebagai G = (V,E) di mana:

- V adalah himpunan simpul
- E adalah himpunan sisi

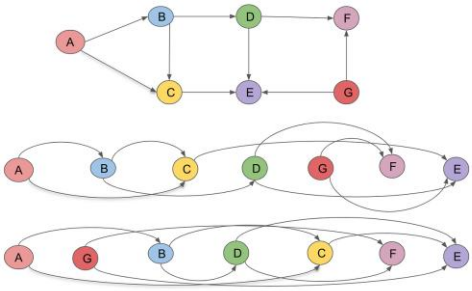
1. Graf Berarah (*Directed Graph*)



Graf berarah (directed graph) adalah graf di mana setiap sisi memiliki arah, dari satu simpul menuju simpul lain. Sisi berarah dilambangkan sebagai pasangan terurut (u,v) yang berarti arah dari simpul u ke simpul v.

Graf berarah digunakan untuk menyatakan urutan atau ketergantungan antara simpul-simpul.

2. Penyusunan Topologis (*Topological Sort*)



Topological sort adalah metode untuk mengurutkan simpul-simpul dalam sebuah graf berarah tanpa siklus (Directed Acyclic Graph/DAG) secara linear, sehingga untuk setiap sisi berarah (u→v), simpul u muncul sebelum simpul v dalam urutan tersebut.

Syarat agar suatu graf dapat diberlakukan *topological sort*, graf tersebut harus memenuhi:

- Graf berarah
- Bukan Graf siklus (acylic)

Urutan hasil topological sort tidak selalu unik. Jika terdapat beberapa simpul tanpa prasyarat, urutan simpul dapat bervariasi. Topological sort hanya berlaku untuk graf berarah tanpa siklus. Jika terdapat siklus, maka urutan yang valid tidak dapat ditentukan.

E. *Kombinatorika*

Kombinatorika mempelajari teknik-teknik penghitungan jumlah cara penyusunan, pemilihan, maupun pengelompokan elemen dari suatu himpunan terbatas. Fokus utamanya adalah bagaimana menyusun atau memilih objek berdasarkan aturan tertentu, baik dengan memperhatikan urutan maupun tidak.

1. Permutasi

Permutasi adalah susunan urutan dari objek-objek yang berbeda. Dalam permutasi, urutan elemen penting.

Jika terdapat n objek yang berbeda, maka jumlah permutasi semua objek tersebut adalah

$$P(n) = n!$$

Jika hanya menyusun r objek dari total n objek, maka jumlah permutasi adalah:

$$P(n,r) = \frac{n!}{(n-r)!}$$

2. Kombinasi

Kombinasi adalah cara memilih sejumlah objek dari sekumpulan objek tanpa memperhatikan urutan.

Jika memilih rrr objek dari total nnn objek berbeda, maka banyaknya kombinasi adalah:

$$C(n,r) = \frac{n!}{r!(n-r)!}$$

III. PEMBAHASAN DAN IMPLEMENTASI

A. *Tujuan Sistem*

Sistem penjadwalan kegiatan ini didesain dalam membantu pengguna dalam merencanakan kegiatan harian mereka dengan memperhitungkan durasi (*weight*), prioritas, dan tenggat waktu (*deadline*) untuk setiap kegiatan. Sistem ini juga dapat memperhitungkan interval waktu bebas (*free time*) dimana sistem tidak akan mengalokasi kegiatan apa pun di interval waktu tersebut.

Karena banyaknya kombinasi atau cara pendistribusian kegiatan , maka sistem mencari semua dan menyarankan beberapa cara pendistribusian dalam bentuk *seed* sehingga pada akhirnya pengguna dapat memilih *seed* mana yang paling memungkinkan untuk diterapkan berdasarkan preferensi mereka.

B. *Deskripsi Umum Sistem*

Sistem dikembangkan sebagai aplikasi berbasis web yang menampilkan kalender harian dengan slot waktu yang dapat diisi oleh kegiatan pengguna. Sistem menampilkan fitur - fitur:

- Tampilan kalender harian dinamis x dalam bentuk blok waktu
- Window *pop up* untuk input kegiatan yang mencakup nama, bobot (durasi total *schedule*) dan tenggat waktu (*deadline*).
- Pilihan cara pendistribusian jadwal (seed) yang dapat dipilih oleh pengguna sesuai dengan preferensi
- Penandaan *schedule* selesai dengan perubahan warna blok sebagai indikator visual
- Waktu nyata (real-time) yang ditampilkan pada layer
- Fitur untuk input interval waktu kosong (free time) dimana tidak boleh ada blok waktu yang di distribusikan di interval waktu kosong ini

C. *Implementeasi*

1. Arsitektur Sistem

Sistem penjadwalan kegiatan harian ini dibangun berdasarkan tiga lapisan utama yang terhubung satu sama lain, yaitu:

- Lapisan Antarmuka (Tampilan Depan / Frontend):

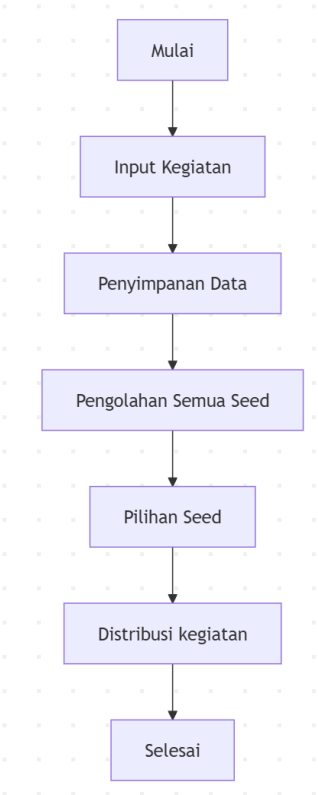
  Ini adalah bagian sistem yang langsung berinteraksi dengan pengguna. Di sinilah

pengguna melihat tampilan kalender, mengisi formulir kegiatan, memilih jadwal, serta menandai kegiatan yang sudah selesai. Semua elemen visual seperti warna blok, jam, dan tombol-tombol berada di lapisan ini.

- Lapisan Logika (Pengolah Utama / Logic Layer):
  Lapisan ini bekerja di belakang layar untuk menjalankan semua proses utama. Di sini sistem akan memproses input dari pengguna, menyusun jadwal berdasarkan algoritma matematika diskrit, mengatur urutan kegiatan, dan memastikan tidak ada jadwal yang bentrok atau melanggar aturan.

- Lapisan Penyimpanan sementara (Database):
  Bagian ini bertugas menyimpan data pengguna seperti daftar kegiatan, jadwal sementara, status selesai/tidaknya kegiatan, dan pilihan seed yang telah dicoba. Penyimpanan ini bersifat sementara dan biasanya dilakukan di dalam browser pengguna, sehingga tidak memerlukan server eksternal.

2. Alur Sistem Distribusi *Schedule*



Penjelasan alur:

1) Mulai: Pengguna membuka aplikasi.

2) Input kegiatan: Pengguna mengisi nama kegiatan, total waktu alokasi (*weight*), dan tenggat waktu (*deadline*).

3) Penyimpanan data: Data kegiatan disimpan di dalam *database.*

4) Pengolahan semua seed: Sistem akan Menyusun semua cara (*seed*) untuk distribusi jadwal.

5) Pilihan seed: Sistem menampilkan semua cara penyusunan jadwal kepada pengguna.

6) Distribusi kegiatan: Cara penyusunan jadwal yang dipilih pengguna kemudian di akan ditampilkan di kalender dinamis.

3. Struktur Data

- kegiatan[]
  Merupakan daftar semua kegiatan yang diinput pengguna.

  - name: nama atau jenis kegiatan (misalnya "Belajar", "Makan")

  - weight: durasi kegiatan dalam satuan jam atau slot waktu

  - deadline: batas waktu penyelesaian kegiatan

  - Contoh:

```
[
  { name: "Belajar", weight: 2, deadline: "2025-06-21-10:00"},
  { name: "Makan", weight: 1, deadline: "2025-06-21-08:00"}
]
```

- Slot[]
  Merupakan daftar semua kegiatan yang merepresentasi blok-blok kegiatan dalam satu hari.

  - waktu: penanda jam, misalnya "08:00"

  - kegiatan: nama kegiatan yang dijadwalkan pada slot itu (atau null jika kosong)

- Contoh:

```
[
  { waktu: "07:00", kegiatan: null },
  { waktu: "08:00", kegiatan: "Makan" },
  { waktu: "09:00", kegiatan: "Belajar" }
]
```

- relasiKegiatan[]
  Merupakan daftar yang berisi semua relasi antar kegiatan. Biasanya diwakili dalam bentuk graf berarah, menggunakan format *adjacency list*

  - kunci (key) yang mewakili kegiatan awal

  - nilai (value) yang merupakan kegiatan yang bergantung pada key

  - Contoh:

```
const relasiKegiatan = {
  "Makan": ["Belajar"],
  "Belajar": ["Olahraga"]
};
```

  - penjelasan:
    'Belajar' hanya boleh dilakukan setelah 'Makan'.
    'Olahraga hanya boleh dilakukan setelah 'Belajar'.

- seed[]
  Merupakan daftar yang berisi semua cara pendistribusian kegiatan

  - Contoh:

```
const seed = {
  seed1: [ "Makan", "Belajar", "Belajar", null, ... ],
  seed2: [ "Belajar", "Makan", "Belajar", null, ... ],
  seed3: [ "Belajar", null, "Makan", "Belajar", ... ]
};
```

4. Algoritma Pendistribusian Jadwal

Program ini dibangun untuk menghasilkan beberapa alternatif penjadwalan kegiatan harian secara otomatis berdasarkan urutan yang logis. Fungsi utama dari program ini adalah generateSeeds, yang akan menyusun tiga versi jadwal (disebut *seed*) dengan pendekatan distribusi yang berbeda-beda, namun tetap mengikuti urutan topologis kegiatan.

- generateSeeds

```
function generateSeeds(activities, relations = {},
  freeTime = [], fixedSchedule = []) {
  const totalSlots = 24;
  const seeds = {};
  const topoSorted =
  topologicalSort(activities.map(a => a.name), relations);
  const sortedActivities =
  topoSorted.map(name => activities.find(a => a.name === name));

  const strategies = {
    seed1: startFrom(0),
    seed2: startFrom(Math.floor(totalSlots / 2)),
    seed3: zigZagOrder(totalSlots)
  };

  for (let [key, slotOrder] of Object.entries(strategies)) {
    const schedule = Array(totalSlots).fill(null);

    for (let fixed of fixedSchedule) {
      for (let i = fixed.start; i <
        fixed.start + fixed.duration; i++) {
        if (i < totalSlots && !freeTime.includes(i)) {
          schedule[i] = fixed.name + " (rutin)";
        }
      }
    }

    for (let activity of sortedActivities) {
      let duration = activity.duration;
      for (let i = 0; i < slotOrder.length && duration > 0; i++) {
        const idx = slotOrder[i];
        if (idx < totalSlots && schedule[idx] ===
          null && !freeTime.includes(idx)) {
          schedule[idx] = activity.name;
          duration--;
        }
      }
    }

    seeds[key] = schedule;
  }

  return seeds;
}
```

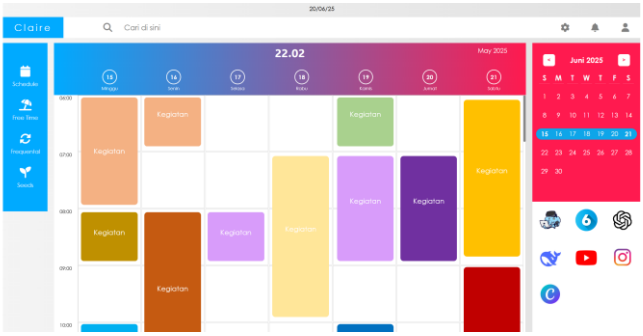
Di dalam fungsi generateSeeds, sistem pertama-tama menentukan jumlah total slot dalam satu hari, yaitu 24 slot (dengan asumsi satu slot mewakili satu jam). Kemudian, sistem melakukan topological sort terhadap daftar kegiatan untuk memastikan urutan penjadwalan tidak melanggar ketergantungan antar kegiatan. Setelah itu, tiga strategi penempatan slot diterapkan: strategi pertama (seed1) mengisi slot dari awal hari, strategi kedua (seed2) dimulai dari pertengahan hari, dan strategi ketiga (seed3) menggunakan pola zig-zag dari ujung ke tengah hari. Untuk setiap strategi, sistem membuat array jadwal kosong, lalu memasukkan jadwal tetap (fixed schedule), dan selanjutnya menempatkan kegiatan berdasarkan hasil urutan topologis ke slot-slot kosong yang tersedia, dengan tetap menghindari slot waktu bebas (freeTime).

- Fungsi pendukung lainnya

```
function startFrom(start) {
  return [...Array(24).keys()].slice(start).
  concat([...Array(start).keys()]);
}

function zigZagOrder(n) {
  const result = [];
  let left = 0, right = n - 1;
  while (left <= right) {
    if (left !== right) result.push(left, right);
    else result.push(left);
    left++;
    right--;
  }
  return result;
}

function topologicalSort(nodes, edges) {
  const visited = new Set();
  const temp = new Set();
  const result = [];

  function visit(n) {
    if (visited.has(n)) return;
    if (temp.has(n)) return;
    temp.add(n);
    (edges[n] || []).forEach(visit);
    temp.delete(n);
    visited.add(n);
    result.unshift(n);
  }

  nodes.forEach(visit);
  return result;
}
```

Beberapa fungsi pembantu digunakan untuk mendukung proses distribusi ini. Fungsi startFrom(start) menghasilkan urutan slot waktu yang melingkar dari titik tertentu, sementara zigZagOrder(n) menghasilkan urutan zig-zag dari awal dan akhir menuju tengah. Fungsi topologi Sort(nodes, edge) mengurutkan aktivitas berdasarkan hubungan ketergantungan menggunakan pendekatan Depth-First Search (DFS), yang memastikan bahwa kegiatan yang bergantung disusun dalam urutan yang logis.

5. Tampilan UI
   Berikut merupakan tampilan kasar UI website/aplikasi.



IV. KESIMPULAN

Sistem penjadwalan aktivitas harian yang dikembangkan dalam studi ini secara efektif menggabungkan berbagai konsep matematika diskrit ke dalam proses distribusi kegiatan. Dengan menggunakan himpunan, relasi, fungsi, graf, *topological sort*, beserta kombinatorika dapat membuat jadwal harian yang logis, efisien, dan responsif terhadap berbagai kondisi pengguna.

Sistem ini menyediakan fleksibilitas kepada pengguna dengan berbagai fitur pemilihan jadwal yang beragam yang memungkinkan pengguna memilih cara pengaturan jadwal alternatif yang sesuai dengan preferensi dan situasi maupun kondisi. Selain itu, fitur seperti jadwal rutin harian, *free time*, dan interaksi visual dengan blok aktivitas dapat meningkatkan efektifitass dan fleksibilitas pengguna dalam menggunakan aplikasi ini.

Dari implementasi yang dilakukan, dapat disimpulkan bahwa pendekatan berdasarkan matematika diskrit sangat relevan dan efektif untuk membangun sistem penjadwalan yang terstruktur dan fleksibel.

V. UCAPAN TERIMA KASIH

Pertama-tama penulis ingin mengucapkan rasa syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat yang Ia berikan sehingga penulis dapat menyelesaikan makalah ini dengan tepat waktu. Penulis juga menyampaikan terimakasih kepada para dosen pengampuh mata kuliah IF1220, khususnya Pak Arrival Dwi Sentosa, S.Kom., M.T, karena telah mengajar dan membiming penulis selama berkuliah matematika diskrit.

REFERENSI

[1] GeeksforGeeks, "Set in Discrete Mathematics," Available: https://www.geeksforgeeks.org/set-theory-in-mathematics/. Accessed 20 June 2025.

[2] GeeksforGeeks, "Functions in Discrete Mathematics," Available: https://www.geeksforgeeks.org/functions-in-mathematics/. Accessed 20 June 2025.

[3] Cuemath, "Relations in Mathematics," Available: https://www.cuemath.com/algebra/relations/. Accessed 20 June 2025.

[4] Tutorialspoint, "Discrete Mathematics – Relations," Available: https://www.tutorialspoint.com/discrete_mathematics/discrete_mathematics_relations.htm. Accessed 20 June 2025.

[5] Calcworkshop, "Relations in Math: Definition & Examples," Available: https://www.calcworkshop.com/relations-functions/relations/. Accessed 20 June 2025.

[6] GeeksforGeeks, "Graph Data Structure And Algorithms," Available: https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/. Accessed 20 June 2025.

[7] OpenDiscreteMath (openmathbooks.org), "Topological Sorting," Available: https://discrete.openmathbooks.org/dmoi3/sec_graph-applications.html. Accessed 20 June 2025.

[8] GeeksforGeeks, "Topological Sorting," Available: https://www.geeksforgeeks.org/topological-sorting/. Accessed 20 June 2025.

[9] Tutorialspoint, "Discrete Mathematics – Combinatorics," Available: https://www.tutorialspoint.com/discrete_mathematics/discrete_mathematics_combinatorics.htm. Accessed 20 June 2025.

[10] Cuemath, "Combinatorics in Math," Available: https://www.cuemath.com/combinatorics/. Accessed 20 June 2025.

[11] R. Munir, "https://informatika.stei.itb.ac.id/," [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/. Accessed 20 June 2024.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 20 Juni 2025

Ishak Palentino Napitupulu – 13524022