

# Sprawozdanie

Anna Hellmann Monika Krakowska Szymon Halski Michał Kowalik	Badania Operacyjne i Logistyka
Informatyka Techniczna III. rok, V. semestr, gr. 2 nr. zespołu	

Link do repozytorium:

<https://github.com/kl0cek/CPM-project->

## Cel projektu

Celem projektu CPM-project było przygotowanie szkieletu aplikacji webowej z wykorzystaniem frameworka Next.js oraz technologii wspierających, takich jak TypeScript, Tailwind CSS do obsługiwaną metody CPM.

## Metoda CPM (Critical Path Method)

Metoda ścieżki krytycznej to technika zarządzania projektami służąca do planowania i kontrolowania zadań składających się na cały projekt. Polega na identyfikacji najdłuższej sekwencji zależnych działań, które determinują minimalny czas realizacji projektu. W projekcie CPM-project metoda ta jest zaimplementowana w backendzie, gdzie na podstawie danych wejściowych (lista zadań z czasami i zależnościami) wyznaczany jest harmonogram oraz graficzna reprezentacja ścieżki krytycznej.

## Funkcja obliczająca ścieżkę krytyczną (CPM)

Ścieżka do pliku: /app/api/project/{projectId}/calculate/route.ts

W celu obliczenia harmonogramu projektu i wyznaczenia ścieżki krytycznej (CPM – *Critical Path Method*), zastosowano funkcję serwera `GET`, która pobiera zadania projektu z bazy danych, przetwarza ich zależności oraz oblicza wartości ES, EF, LS, LF, zapas czasu (slack) oraz oznacza zadania krytyczne. Poniżej znajduje się fragment kodu odpowiedzialny za te obliczenia:

```
tasks.forEach(task => {  
  tasksMap[task.id] = { ...task, ES: 0, EF: task.duration, LS: Infinity, LF: Infinity, slack: 0 };  
});
```

```

const inDegree: { [key: number]: number } = {};
const childrenMap: { [key: number]: number[] } = {};

tasks.forEach(task => {
  inDegree[task.id] = task.dependencies.length;
  childrenMap[task.id] = [];
});

tasks.forEach(task => {
  task.dependencies.forEach(depId => {
    if (childrenMap[depId]) {
      childrenMap[depId].push(task.id);
    }
  });
});

let queue: number[] = [];
tasks.forEach(task => {
  if (inDegree[task.id] === 0) queue.push(task.id);
});

let topoOrder: number[] = [];

while (queue.length > 0) {
  const tid = queue.shift();
  topoOrder.push(tid);

  const task = tasksMap[tid];
  childrenMap[tid].forEach(childId => {
    const child = tasksMap[childId];
    child.ES = Math.max(child.ES!, task.EF!);
    child.EF = child.ES! + child.duration;
    inDegree[childId]--;
    if (inDegree[childId] === 0) queue.push(childId);
  });
}

let projectDuration = Math.max(...tasks.map(task => task.EF!));

for (let i = topoOrder.length - 1; i >= 0; i--) {
  const tid = topoOrder[i];
  const task = tasksMap[tid];

  if (childrenMap[tid].length === 0) {
    task.LF = projectDuration;
  } else {

```

```

    task.LF = Math.min(...childrenMap[tid].map(childId => tasksMap[childId].LS!));
  }

  task.LS = task.LF! - task.duration;
  task.slack = task.LS! - task.ES!;
  task.isCritical = task.slack === 0;
});
return NextResponse.json({ projectDuration, tasks: Object.values(tasksMap) });}

```

Funkcja działa na zasadzie sortowania topologicznego z użyciem kolejki oraz późniejszej analizy wstecznej. Obliczone wartości pozwalają na identyfikację zadań krytycznych, które bezpośrednio wpływają na czas trwania całego projektu.

## Użyte technologie

**React + Next.js** – frontend aplikacji został zbudowany przy użyciu Reacta oraz frameworka Next.js (z wykorzystaniem App Routera), co umożliwia łatwe zarządzanie trasami i komponentami oraz wspiera SSR (server-side rendering).

**TypeScript** – statyczne typowanie zastosowane w całym projekcie frontendowym poprawia bezpieczeństwo kodu i ułatwia jego utrzymanie.

**Material UI (MUI)** – biblioteka komponentów UI w stylu Material Design, używana do budowy formularzy, tabel, list, kontenerów oraz nawigacji.

**React Flow Renderer** – biblioteka służąca do graficznej reprezentacji sieci zależności między zadaniami. Umożliwia wizualizację przepływu zadań i ścieżki krytycznej w formie grafu.

**Google Charts (react-google-charts)** – wykorzystana do budowy wykresu Gantta z oznaczeniem ścieżki krytycznej. Pozwala na interaktywne przedstawienie harmonogramu projektu.

**Next.js API Routes** – do obsługi backendowej logiki algorytmu CPM (pobieranie i przetwarzanie danych o zadaniach, obliczenia CPM, generowanie PDF). Komunikacja z backendem odbywa się poprzez `fetch()` w React.

**Tailwind CSS** - Narzędzie do stylowania interfejsu użytkownika w oparciu o klasy utility. Umożliwia szybkie i responsywne projektowanie UI.

**JSON** - Format wykorzystywany do przesyłania danych między frontendem a backendem (np. przesyłane są informacje o projektach, zadaniach i zależnościach).

## Funkcjonalności aplikacji

### CPM Project Management

CREATE PROJECT

Projects

asd

DELETE

monika

DELETE

przedstawienie

DELETE

*Widok listy projektów z możliwością tworzenia i usuwania projektów*

Add Task

ADD TASK

Tasks

[ID: 202] A (Duration: 5, Dependencies: )

EDIT

DELETE

[ID: 203] B (Duration: 7, Dependencies: )

EDIT

DELETE

[ID: 204] C (Duration: 6, Dependencies: 202)

EDIT

DELETE

[ID: 205] D (Duration: 8, Dependencies: 202)

EDIT

DELETE

[ID: 206] F (Duration: 4, Dependencies: 204)

EDIT

DELETE

[ID: 207] G (Duration: 2, Dependencies: 204)

EDIT

DELETE

[ID: 208] E (Duration: 3, Dependencies: 203)

EDIT

DELETE

[ID: 209] H (Duration: 5, Dependencies: 208, 205, 206)

EDIT

DELETE

CALCULATE CPM

EXPORT PDF

*Formularz do dodawania zadań z ich nazwami, id i trwaniem oraz lista zadań*

Network Diagram

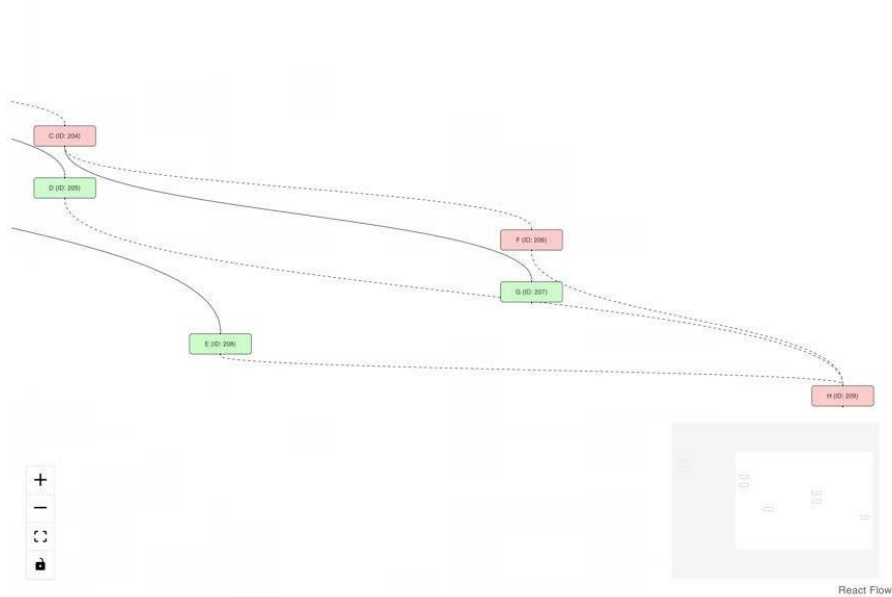
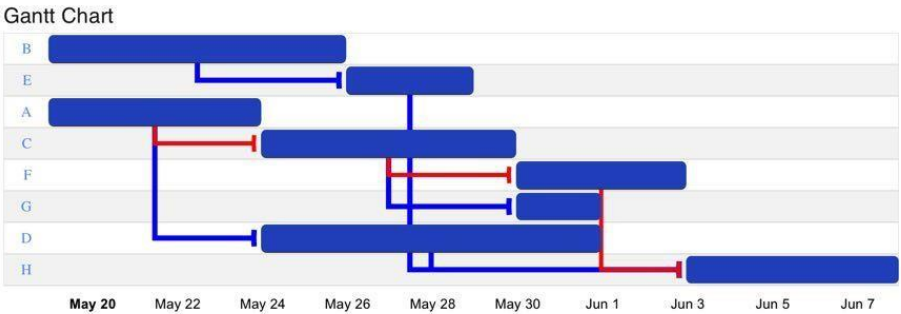


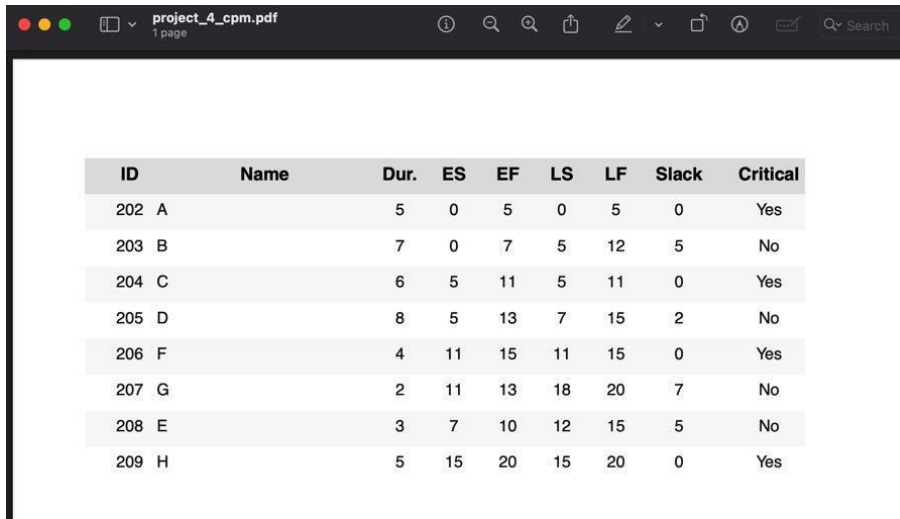
Diagram sieciowy z zależnościami między sieciami



Wykres Gantta wizualizujący harmonogram zadań

Tasks Details									
ID	Name	Duration	ES	EF	LS	LF	Slack	Critical	
202	A	5	0	5	0	5	0	Yes	
203	B	7	0	7	5	12	5	No	
204	C	6	5	11	5	11	0	Yes	
205	D	8	5	13	7	15	2	No	
206	F	4	11	15	11	15	0	Yes	
207	G	2	11	13	18	20	7	No	
208	E	3	7	10	12	15	5	No	
209	H	5	15	20	15	20	0	Yes	

Tabela z obliczonymi parametrami zadań

A screenshot of a PDF viewer window titled "project\_4\_cpm.pdf" with "1 page" indicated. The viewer's toolbar is visible at the top. The main content is a table with 9 columns: ID, Name, Dur., ES, EF, LS, LF, Slack, and Critical. The table contains 8 rows of data for activities A through H. The "Critical" column has values "Yes" or "No".

ID	Name	Dur.	ES	EF	LS	LF	Slack	Critical
202	A	5	0	5	0	5	0	Yes
203	B	7	0	7	5	12	5	No
204	C	6	5	11	5	11	0	Yes
205	D	8	5	13	7	15	2	No
206	F	4	11	15	11	15	0	Yes
207	G	2	11	13	18	20	7	No
208	E	3	7	10	12	15	5	No
209	H	5	15	20	15	20	0	Yes

*Eksport danych do pdf*

## Wnioski

Projekt pozwala na praktyczne zastosowanie metody ścieżki krytycznej (CPM) w zarządzaniu projektami. Dzięki implementacji algorytmu CPM możliwe jest automatyczne wyznaczanie zadań krytycznych oraz określanie najwcześniejszych i najpóźniejszych momentów rozpoczęcia i zakończenia poszczególnych czynności.

Rozwiązanie to ma duże znaczenie w kontekście planowania czasu trwania projektów. Aplikacja dostarcza nie tylko danych liczbowych, ale również graficzną wizualizację całego harmonogramu, co wspomaga analizę i podejmowanie decyzji.