




Vibe Coding



Budowanie aplikacji przy z Claude i porównanie z
innymi modelami
Raport badawczy



Czym jest vibe coding?

Vibe coding to swobodny styl programowania, w którym głównym celem jest płynność, kreatywność i przyjemność z pisania kodu, często bez sztywnego planu.

Programista działa intuicyjnie, eksplorując pomysły w czasie rzeczywistym, często przy muzyce i w luźnej atmosferze. To podejście sprzyja szybkiemu prototypowaniu i rozwijaniu pomysłów „na flow”.



Przebieg badania

1. Wybór modeli jezykowych na podstawie ich dostepnosci, ceny, opinii uzytkownikow
2. Kazdy model bedzie mial na celu utworzenie tej samej aplikacji
3. Modele dostana te same prompty
4. Code review oraz Manual Testy przeprowadzone przeze mnie
5. Porownanie i podsumowanie

Wybór modeli do wykonania zadania

CLAUDE

- + darmowy (z ograniczeniami)
- + stworzony do programowania

OPENAI O4.1

- + prekursor
- + uniwersalny
- lepsze wersje są płatne

DEEPSEEK

- + w pełni darmowy,
- + open-source,
- + wbudowane css preview,
- oceniany

Co będą tworzyć?

Docelową aplikacją ma być frontendowy web app - to-do app, napisany w TS, przy użyciu framework'a Angular 19.

Czemu Angular?



Ponieważ jest to szybko zmieniający się framework (2 duże update na rok).

Będziemy oczekiwać najlepszych praktyk od modeli, przy użyciu najbardziej aktualnych metod, praktyk.

Wymagania funkcjonalne

- dodawanie nowych taskow
- wyswietlanie tablic to - do, done
- zaznaczanie na done
- edycja taska
- usuwanie taska

Zasady

1. Jeden projekt - maks. 20 promptów
2. Każdy projekt dostanie te same prompty, w przypadku błędów - maks. 2 prompty na unikalne poprawienie (jedyny moment kiedy prompty beda sie różnic).
3. Projekty mają korzystać z najlepszych praktyk i najbardziej aktualnych rozwiązań.
4. Nie ingeruję w kod.

Zasady oceniania

1. Jakość kodu
2. Wydajność aplikacji
3. Dokładność generowanego kodu
4. Styl wizualny
5. Występowanie problemów technicznych
6. Koszta

CLAUDE

model: claude 3.5 sonnet

1. Tell me, what is the latest official version of Angular as of June 10, 2025?

Based on the search results, I can tell you that Angular 20.0 is the latest official version as of June 10, 2025. It was released on May 28, 2025 and brings several major new features and improvements:

Claude 3.5 sonnet odpowiedział poprawnie

2. Tworzenie projektu

I'll help you create a modern To-Do application using Angular 20's latest features and best practices. Let's start by creating a new project.

First, I'll run the Angular CLI command to create a new project with the latest features enabled:

```
ng new todo-app --standalone --ssr  
--style=css --routing=true  
--skip-tests=false --strict=true  
--zoneless
```

Ask every time ▾

Skip Shift+⌘

Run ⌘↵

2.1 Pierwsze błędy

```
Error: Unknown argument: zoneless
PS C:\Users\pkosm\PorownanieModeliJezykowych>
```

aczkolwiek...

```
I apologize for the error. It seems the --zoneless flag is not yet available in the CLI. Let's create the project without it and we'll add zoneless mode manually after:
```

Cursor (edytor) sam, bez zadnego innego prompta znalazł ten błąd i go naprawił.

2.2 Pierwsze kroki

Utworzenie projektu, dodanie stylów,
komponentów, cleanup oraz testowanie

Let's also add some global styles to make our app look better:

```
# styles.css +62 •  
  
html,  
body {  
  margin: 0;  
  padding: 0;  
}  
  
body {  
  font: 14px 'Helvetica Neue', Helvetica, Arial, sans  
  line-height: 1.4em;  
  background: #f5f5f5;  
  color: #4d4d4d;  
  min-width: 230px;  
  max-width: 100%;  
}
```

Now let's clean up the unnecessary files that were generated by the Angular CLI:

✕ Delete file # app.component.css

✕ Delete file 📄 app.component.html

✕ Delete file 📄 app.component.spec.ts

Let's start the development server to see our application in action:

```
cd todo-app && ng serve
```

Ask every time ▾

Skip Shift+⌘

Open terminal ↗

2.3 Testowe “odpalenie” projektu

X [ERROR] NG1: Property 'todoService' is private and only accessible within class 'TodoList'. [plugin angular-compiler]

```
src/app/todo-list.ts:24:24:
 24 |         @for (todo of todoService.allTodos(); track todo.id) {
    |                        ~~~~~
```

X [ERROR] NG1: Property 'todoService' is private and only accessible within class 'TodoList'. [plugin angular-compiler]

```
src/app/todo-list.ts:31:28:
 31 |         (change)="todoService.toggleTodo(todo.id)">
    |                        ~~~~~
```

X [ERROR] NG1: Property 'todoService' is private and only accessible within class 'TodoList'. [plugin angular-compiler]

```
src/app/todo-list.ts:35:27:
 35 |         (click)="todoService.removeTodo(todo.id)">
    |                        ~~~~~
```

X [ERROR] NG1: Property 'todoService' is private and only accessible within class 'TodoList'. [plugin angular-compiler]

```
src/app/todo-list.ts:54:13:
 54 |         {{ todoService.activeTodos().length }} items left
    |             ~~~~~
```

X [ERROR] NG1: Property 'todoService' is private and only accessible within class 'TodoList'. [plugin angular-compiler]

```
src/app/todo-list.ts:58:19:
 58 |         (click)="todoService.clearCompleted()">
    |                        ~~~~~
```

Watch mode enabled. Watching for file changes.

3. Naprawianie błędów private

Proste błędy szybko zostały naprawione, kosztowało nas to jednak dodatkowy prompt. Aplikacja prezentuje się następująco:

todos

What needs to be done?

0 items left

Clear completed

todos

What needs to be done?

do the dishes

halo

2 items left

Clear completed

PRZYCISK
NIE DZIAŁA

4. Przerabianie aplikacji pod nas

Przycisk

ADD TASK

nie działa...

Task List

Add Task

No tasks yet. Add your first task above!

5. Dalsze próby naprawienia add tasks

```
<form [formGroup]="taskForm" (ngSubmit)="addTask()" class="add-task-form">
  <div class="form-group">
    <textarea
      formControlName="description"
      placeholder="Enter task description..."
      required
      rows="3"
      class="task-input">
    </textarea>
  </div>
```

Znalezione przeze mnie błędy sugerują, że validacja nie odbywa się poprawnie, natomiast funkcjonalność, którą wybrał Claude -

```
#newTodo
(keyup.enter)="addTodo(newTodo)"
autofocus>
```

nie ma żadnego sensu - "enter" dodaje nową linijkę, a nie task...

6-8. Dodanie mockowych zadań, usuwanie

W tym momencie aplikacja zaczynała wyglądać obiecująco.

Niestety, żadna logika nie działała, po paru promptach na temat naprawy, przystąpiłem do dalszej pracy.

Task List

Add Task

☒ Complete Angular tutorial
Created: Jun 14, 2025, 6:28:56 PM

☐ Review pull requests
Created: Jun 15, 2025, 4:28:56 PM

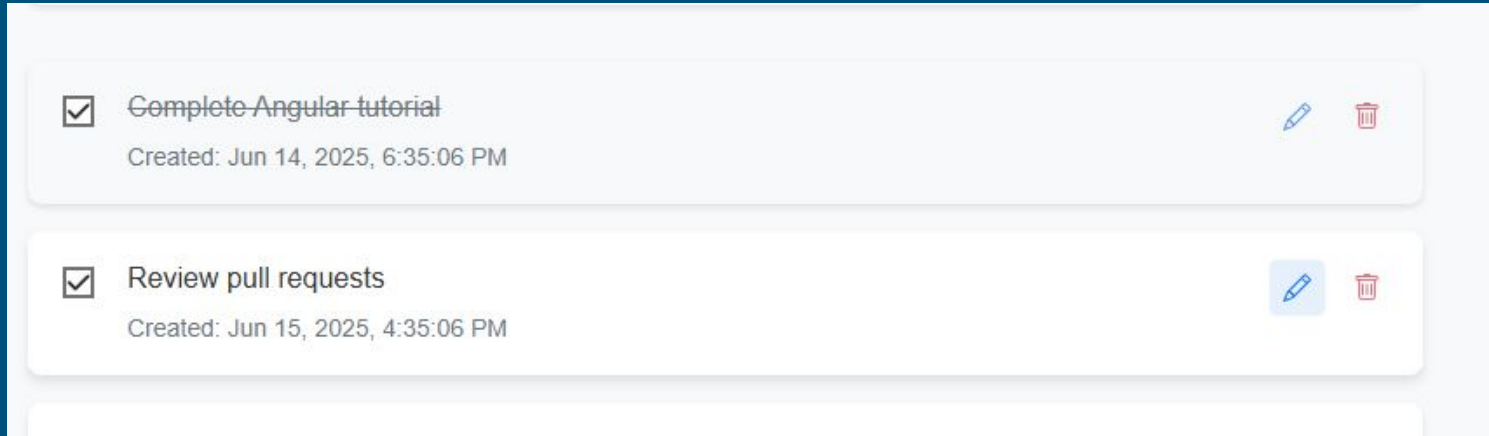
☐ Plan team meeting agenda
Created: Jun 15, 2025, 5:58:56 PM

2 pending

1 completed

Clear completed

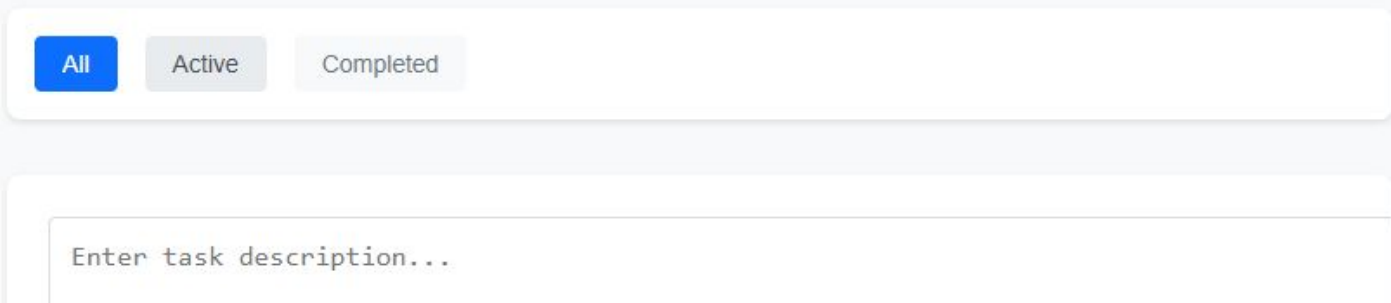
9-12. Add task editing, when clicking on a icon next to a task is should be editable. After saving the changes should be visible immiediately.



Przyciski Edit zostały dodane - ponownie nie były one funkcjonalne.

13. Dodawanie filtrowania

Task List



A UI mockup for a 'Task List' application. It features a header with the title 'Task List'. Below the header is a horizontal bar containing three filter buttons: 'All' (highlighted in blue), 'Active' (light gray), and 'Completed' (light gray). Below this bar is a large, empty rectangular area intended for displaying task items. At the bottom of the mockup is a text input field with the placeholder text 'Enter task description...'.

Ponownie - bez funkcjonalności, dlaczego?

Poruszę ten temat w podsumowaniu, pomimo, że próbowałem przekonać Claude do odejścia od tego tematu.

14-15. Dodanie localStorage

Error loading tasks from localStorage: ReferenceError: localStorage is not defined

Claude na początku nie poradził sobie z wdrożeniem localStorage do aplikacji. Po jednym prompcie naprawił to i localStorage zaczął działać

taskFilter	all
tasks	[[{"id":"t1","userId":"u1","title":"Master Angular","summary":"Learn all th...

16-20. Pozostałe funkcjonalności + poprawki

Pozostałe funkcjonalności - dodanie Angular Animations, użycie cdk material angulara, drag and dropy zostały dodane przez Claude, aczkolwiek ich przetestowanie jest niemożliwe, ze względu na brakującą logikę w komponentach.

PODSUMOWANIE

Aplikacja finalnie prezentuje się następująco:

Task List

All

Active

Completed

Enter task description...

mm/dd/yyyy



Add Task



Complete Angular tutorial

Created: Jun 14, 2025, 7:05:57 PM Due: Jun 14, 2025



Review pull requests

Created: Jun 15, 2025, 5:05:57 PM Due: Jun 16, 2025



Plan team meeting agenda

Created: Jun 15, 2025, 6:35:57 PM Due: Jun 15, 2025



2 pending

1 completed

Clear completed

Problemy

Współpraca niestety nie przebiegła pomyślnie, finalnie otrzymaliśmy schludną i elegancką, aczkolwiek нефunkcjonalną aplikację.

Niedziałające przyciski, nielogiczne interfejsy wskazują na brak “człowieczego” nadzoru.

Czemu funkcjonalność nie została zachowana? Claude od początku usilnie starał się używać nie logicznego rozwiązania - dodawania zadań poprzez metody keyup/keydown - które nie były używane poprawnie.

Problemy c.d.

Pomimo nakierowywania modelu, przyciski nadal nie działały, co ostatecznie spowodowało negatywny rezultat testu.

Wystarczyły małe zmiany, które niestety Claude pomijał za każdym razem.

Co poszło dobrze?

Już na pierwsze pytanie - o najbardziej aktualną wersję Angulara - Claude odpowiedział poprawnie (koniec maja - Angular 20).

Ponadto, Claude poprawnie używał najlepszych i najbardziej aktualnych technik

```
private isBrowser = isPlatformBrowser(inject(PLATFORM_ID));  
private tasks = signal<Task[]>(this.loadTasks());  
private currentFilter = signal<TaskFilter>(this.loadFilter());  
  
constructor() {
```

tzn. private zmiennych, signalsów.

Zalety c.d.

Kod jest wyposażony w wiele przydatnych komentarzy:

```
const tasks = JSON.parse(savedTasks);

// Convert string dates back to Date objects and ensure order property exists
return tasks.map((task: any, index: number) => ({
  ...task,
  createdAt: new Date(task.createdAt),
  dueDate: task.dueDate ? new Date(task.dueDate) : null,
```

```
description: 'Plan team meeting agenda',
createdAt: new Date(Date.now() - 30 * 60 * 1000), // 30 minutes ago
dueDate: new Date() // Today
```

Ocena

1. Jakość kodu

Niska – użyte zostały złe funkcje, co skutkowało brakiem działania przycisków.

2. Wydajność aplikacji

Nie można ocenić, aplikacja nie działała prawidłowo

3. Dokładność wygenerowanego kodu

Niska - aplikacja nie działała, co wskazuje na błędny kod.

Ocena c.d

4. Styl wizualny

- Aplikacja była bardzo ładna i estetyczna

5. Występowanie problemów technicznych

Wysokie - niedziałające przyciski to bardzo duży problem funkcjonalny.

6. Koszta

Istnieje wersja darmowa - wykorzystana w badaniu.

OPEN AI

model: ChatGPT 4o

1. Wersja Angulara

As of **June 10, 2025**, the **latest official stable version of Angular** is **v20.0.3**, released on **June 11, 2025**

`endoflife.date` +4

To recap the recent major releases:

- **Angular 20** launched **May 28, 2025**, with the latest patch (v20.0.3) arriving on **June 11, 2025** `en.wikipedia.org`
- Prior to that, **Angular 19** was released on **November 19, 2024**, followed by ongoing updates

`en.wikipedia.org` +8

In summary:

- ✓ **Angular 20.0.3** is the current stable release as of mid-June 2025.

Ponownie - poprawna dane.

2. Inicjalizacja aplikacji

Tak wygląda pierwszy prompt przy budowie aplikacji poprzez chat GPT.

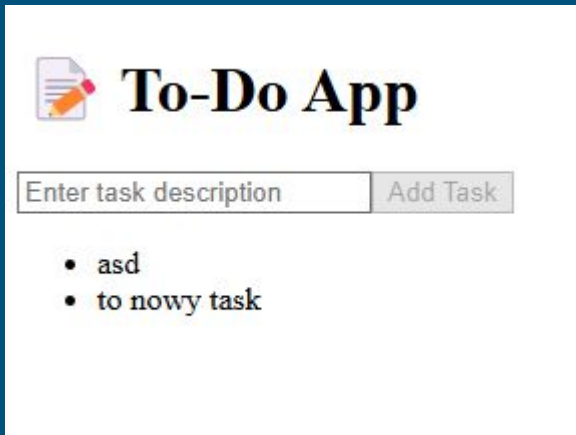
Na razie wszystko przebiegło pomyślnie, można zwrócić uwagę na emotikony.



To-Do App

todo works!

3. Dodanie listy zadań



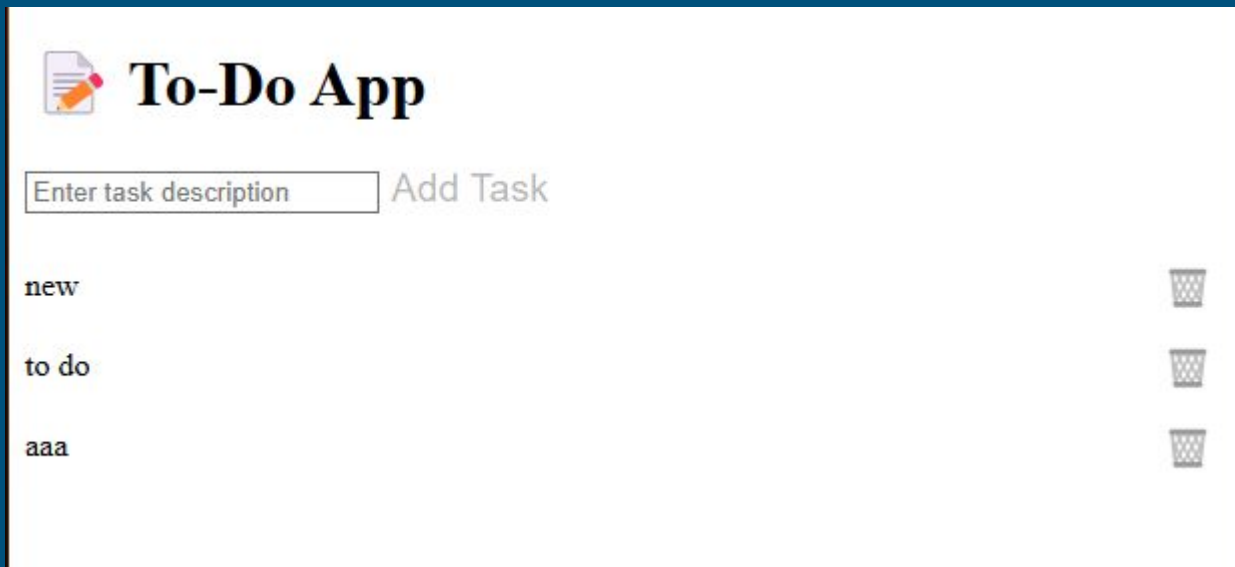
Chat GPT dodał bez problemu pole z działającym przyciskiem na nowe zadania.

Co ciekawe, nie wykorzystał wcześniej utworzonego komponentu "todo", zamiast tego zignorował go i nie zutilizował, stworzył nowy, pełniący tą samą funkcję.

4. Usuwanie zadań

Następnym zadaniem było dodanie nowej funkcjonalności - usuwanie dodanych przez użytkownika zadań. Logika działa bez problemu. Zwrócić uwagę można

Natomiast na brak dobrych technik UX, przyciski znajdują się po drugiej stronie ekranu, a przycisk Add Task jest słabo widoczny.



5. Naprawienie stylu



To-Do App

Add Task

new



to do



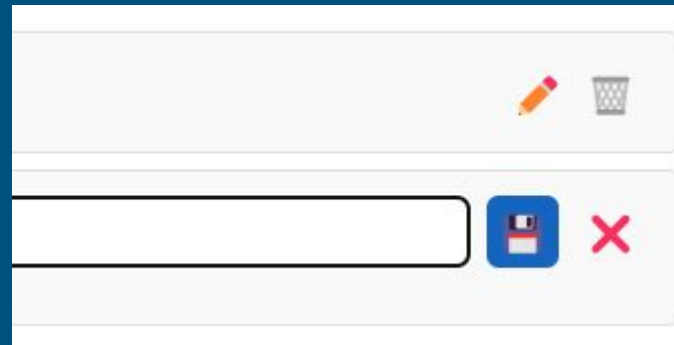
aaa



Postanowiłem poprosić ChataGPT o naprawienie UI/UX poprzez dodanie lepszych stylów. Funkcjonalność została zachowana, a styl poprawiony.

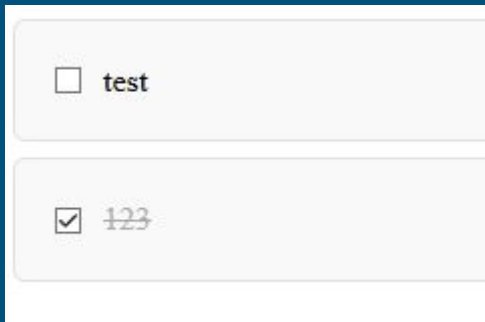
7. Edycja zadań

Dodanie funkcjonalności edytowania zadań przebiegło poprawnie. Można zwrócić uwagę, na dobrze dobrane kolory, style oraz ikony.



Kod ze strony technicznej zostanie podsumowany po finalnej wersji aplikacji.

8. Oznaczanie zadań jako wykonanych



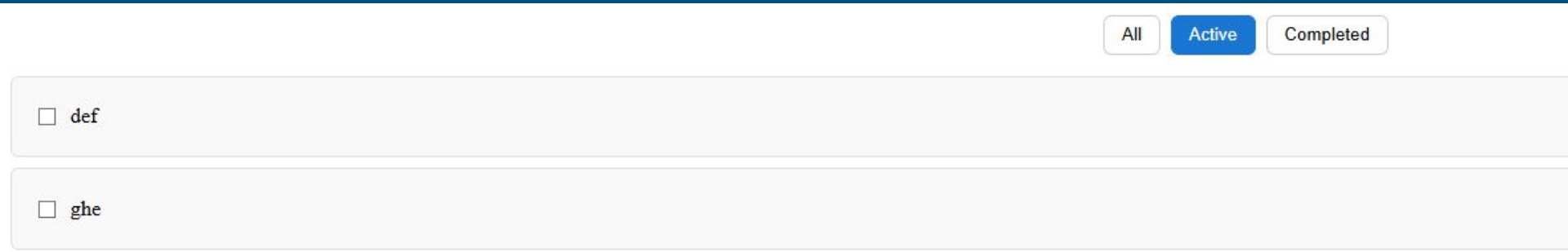
A screenshot of a task list interface. It contains two items, each with a checkbox and a text label. The first item has an unchecked checkbox and the label 'test'. The second item has a checked checkbox and the label '123'.

<input type="checkbox"/>	test
<input checked="" type="checkbox"/>	123

Dodanie następnej funkcjonalności nie sprawiło żadnych problemów.

Zadania możemy teraz “odhaczać”.

9. Filtrowanie zadań



ChatGPT dodał filtrowanie zadań na podstawie ich statusu.

Przyciski są wystylowane, animowane. Aplikacja na ten moment prezentuje się obiecująco i działa płynnie.



10-12. Integracja listy zadań z localStorage

To zadanie spowodowało chwilową stagmację. ChatGPT miał problem z wzięciem pod uwagę wszystkich dotychczasowych kontekstów komponentu.


Po dwóch promptach zużytych na poprawki, localStorage został dodany.

```
▼ [{id: 1, description: "abc", done: false}, {id: 2, description: "def", done: false},...]  
  ▶ 0: {id: 1, description: "abc", done: false}  
  ▶ 1: {id: 2, description: "def", done: false}  
  ▶ 2: {id: 3, description: "ghj", done: false}
```


13. Datowanie zadań + ostrzeżenie

<input type="checkbox"/> 123123	(Due: 2025-05-05) Overdue! 
<input type="checkbox"/> abc abc	(Due: 2030-02-02) 

Rezultat zadania prezentuje się następująco. Dodany został date-picker.

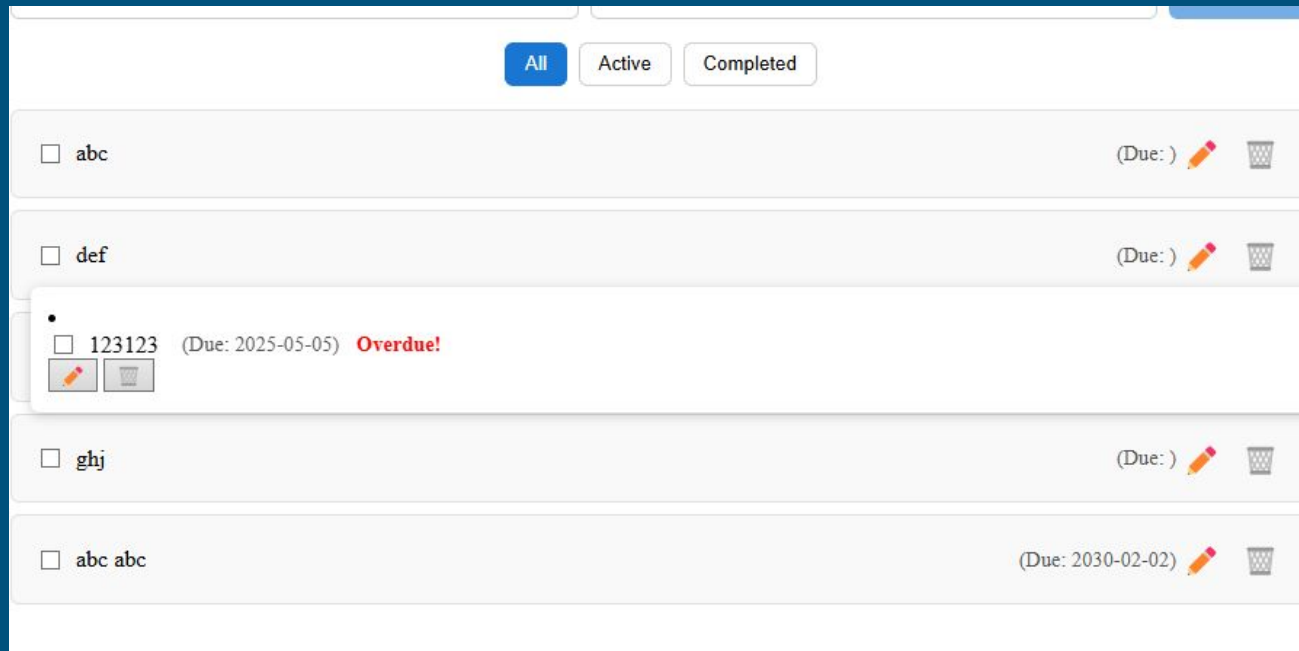


A

14-16. Drag and Drop

Jedyny problem
był związany z
integracją
biblioteki z
odpowiednią
wersją Angulara.

Funkcjonalność
zaczęła działać
po rozwiązaniu
problemu.



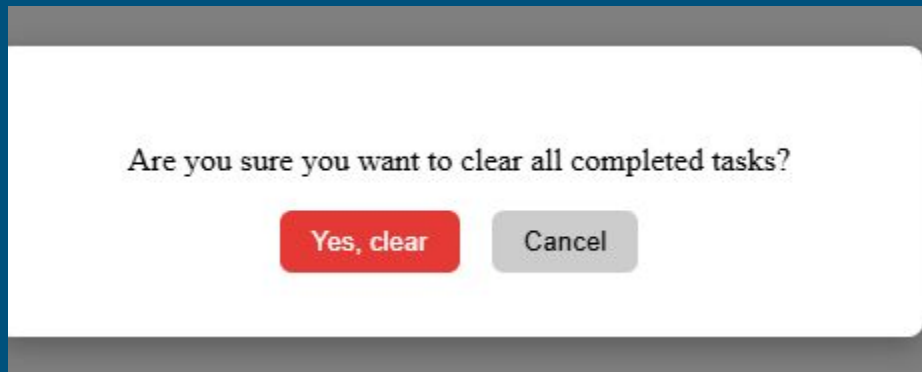
17-19. Filtrowanie wykonanych zadań i usuwanie ich za pomocą jednego przycisku



Przycisk został dodany, ale po dwóch dodatkowych promptach. Problemem było nie użycie angularowych signalsów, co dwukrotnie powodowało error.

Gdy error został przedstawiony, ChatGPT wyprodukował nowy, poprawiony kod.

20. Dodanie modal dla usuwania zadań



Podsumowanie

To-Do App





Add Task


All

Active

Completed

☒ ghj

(Due:)  

 Clear Completed

Zalety

Wszystkie problemy natury technicznej były proste do rozwiązania. Głównie opierały się na odniesieniu do złej wersji Angular lub niepoprawnym użyciu Angularowych signals.

Aplikacja działa stabilnie i płynnie, użyte zostały nowoczesne techniki programistyczne - signals, computed.

Nowo dodane funkcje nie powodowały regresji, a sama aplikacja jest intuicyjna i schludna.

Wady

```
*ngIf="showConfirmModal()">
```

ChatGPT w template HTML używał przestarzałych od najnowszej wersji metod `ngIf`, `ngFor` co jest istotnym błędem technicznym, ponieważ powoduje to komplikacje przy przechodzeniu na kolejne wersje. Ponadto cała logika aplikacji nie jest rozdzielona na mniejsze części - wszystko jest w jednym komponencie.

Nie występują `services`, osobne pliki na interfejsy.

Ocena

1. Jakość kodu

Wysoka - kod oparty na nowoczesnych technikach, z wyjątkiem ngIf, ngFor

2. Wydajność aplikacji

Bardzo dobra, aplikacja działa płynnie

3. Dokładność generowanego kodu

Wysoka - wszystkie funkcjonalności działały poprawnie

Ocena c.d.

4. Styl wizualny

Średni - aplikacja prezentuje się dobrze, jest czytelna i użyteczna, ale nie wyróżnia się wyjątkową estetyką.

5. Występowanie problemów technicznych

Brak większych problemów - aplikacja działa stabilnie

6. Koszta

20\$ miesięcznie (lub pare darmowych promptów dziennie)

DEEPSEEK

model: r1

1. Wersja Angulara

As of June 10, 2025, the latest official stable version of Angular is Angular 19.

Angular follows a regular release schedule, with major updates every 6 months. Here's a quick breakdown of recent versions leading up to June 2025:

- Angular 16 – Released in May 2023
- Angular 17 – Released in November 2023
- Angular 18 – Released in May 2024
- Angular 19 – Released in November 2024 (and remains the latest stable version as of June 2025)

Tutaj po raz pierwszy otrzymaliśmy błędną wersję Angulara. W maju 2025 została wydana wersja Angular 20.

2. Inicjacja projektu

Aplikacja po pierwszym zapytaniu prezentuje się następująco:

Moją uwagę przykuło brak rozbicia elementów na różne pliki. Cała logika była trzymana w jednym pliku

Angular 19 To-Do App

2.1

Deep seek nie rozdzielił zawartości komponentu na różne pliki - .ts, .html, .scss, wszystko jest zbite w jednym pliku.

Jest to bardzo duży błąd, tworzy to nieczytelne komponenty, w których ciężko odnaleźć szukane zmienne, funkcje itp.

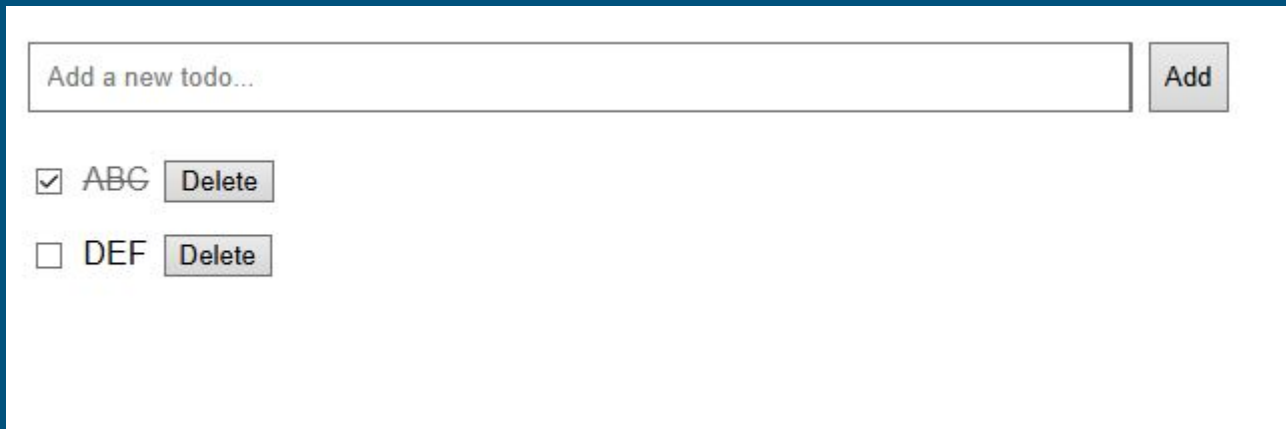
```
@Component({
  selector: 'app-root',
  standalone: true,
  imports: [TodoFormComponent, TodoListComponent],
  template: `
    <div class="container">
      <h1>Angular 19 To-Do App</h1>
      <app-todo-form (addTodo)="todoService.addTodo($event)" />
      <app-todo-list
        [todos]="todoService.todos()"
        (toggleTodo)="todoService.toggleTodo($event)"
        (deleteTodo)="todoService.deleteTodo($event)"
      />
    </div>
  `,
  styles: [
    `
    .container {
      max-width: 600px;
      margin: 0 auto;
      padding: 20px;
      font-family: sans-serif;
    }
    h1 {
      text-align: center;
    }
  `
  ],
})
export class AppComponent {
  constructor(public todoService: TodoService) {}
}
```

2.2

Aczkolwiek, Deep seek
dodał wiele funkcjonalności
“od siebie”.

Od razu mamy możliwość usuwania zadań oraz oznaczania ich jako wykonane.

Jako jedyny model nie posiada walidacji danych.



The screenshot shows a web application interface for a todo list. At the top, there is a text input field with the placeholder text "Add a new todo..." and an "Add" button to its right. Below the input field, there are two list items. The first item is "ABC" with a checked checkbox to its left and a "Delete" button to its right. The second item is "DEF" with an unchecked checkbox to its left and a "Delete" button to its right.

3. Edycja zadań



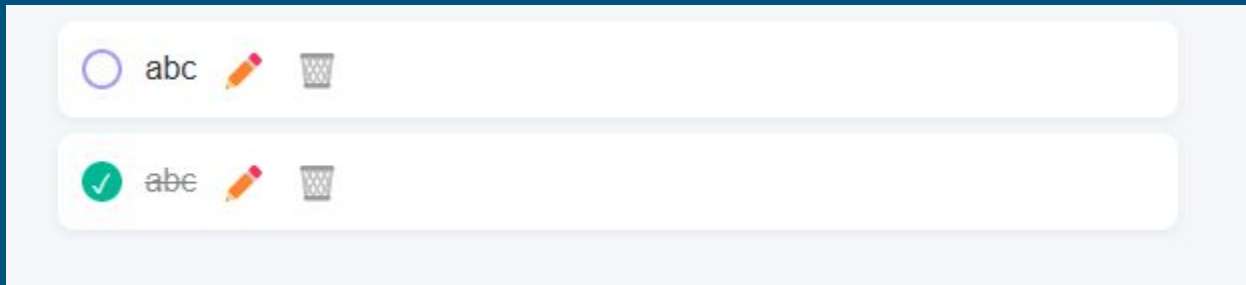
Dodanie funkcjonalnego przycisku edycji zadań nie było problemem.

Uwagę można zwrócić na brak stylu, aplikacja nie wygląda estetycznie.

4-6. Stylizowanie aplikacji

Przy próbie dodania stylu w aplikacji, deepseek dodał pliki scss, jednocześnie nie importując ich do plików .ts (poprzez odniesienie się do nich w pliku .ts).

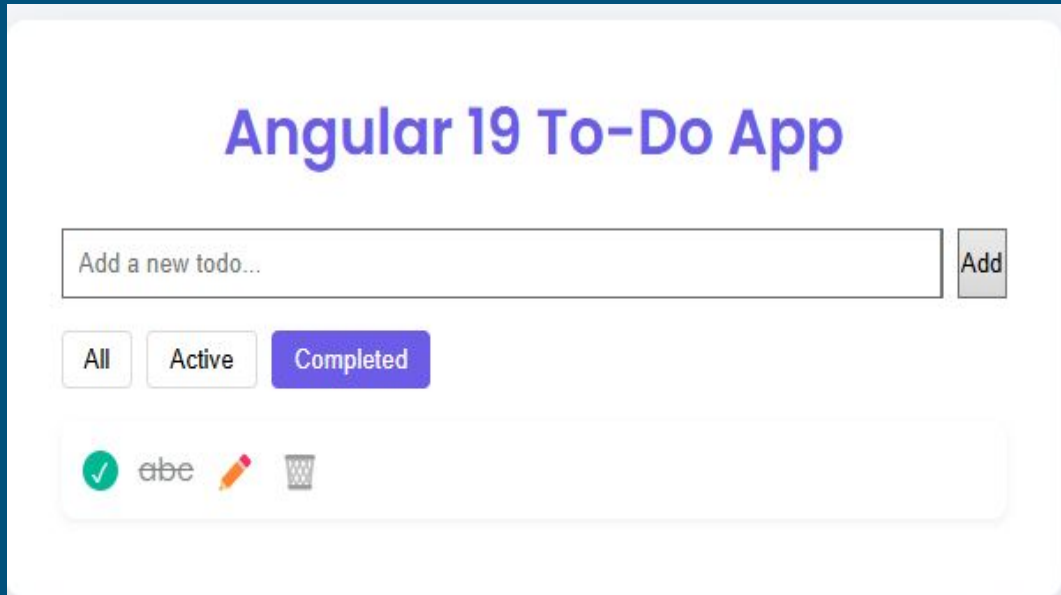
Po promptach naprawiających ten problem, zadania prezentują się następująco:



7. Dodanie filtrowania

Dodanie funkcjonalności filtrowania zadań odbyło się bez problemów.

Deepseek stworzył nowy komponent - filter, co jest bardzo dobrą praktyką. Poprzednie modele tego nie uwzględniały.



8-10. Implementacja localStorage

Zaimplementowanie localStorage na początku sprawiło problem deepseekowi, natomiast po przesłaniu błędów w aplikacji, mogliśmy przejść do dalszego rozwijania programu.

LocalStorage zaczął działać poprawnie.



```
▼ [{id: 1750097666451, text: "bc", completed: false},...]  
  ▶ 0: {id: 1750097666451, text: "bc", completed: false}  
  ▶ 1: {id: 1750097667267, text: "ab c", completed: false}  
  ▶ 2: {id: 1750097667974, text: "ede", completed: false}
```

11-13. Dodanie dat

Ponownie występowały problemy, Deepseek nie radzi sobie z tak dużą liczbą zależności na tym etapie projektu.

Każdy nowy prompt powodował np. nierenderowanie się poprzednich funkcjonalności itd.

Ostatecznie, funkcjonalność została wdrożona.

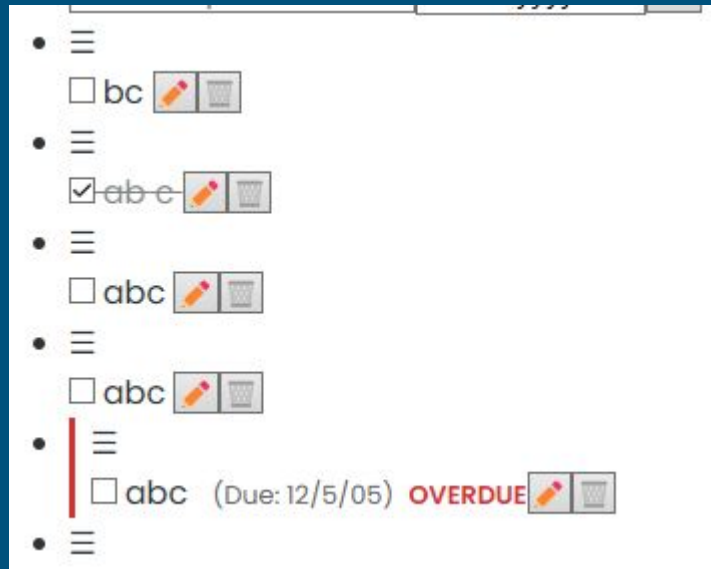
- 
- ☐ abc (Due: 12/5/05) **OVERDUE**  
 - ☐ test (Due: 10/10/30)  

14-15. Drag and drop

Implementacja Drag and Drop nie była większym problemem dla Deep seek'a.

Co ciekawe, Deep seek nie skorzystał z wbudowanego w Angulara modułu drag and drop.

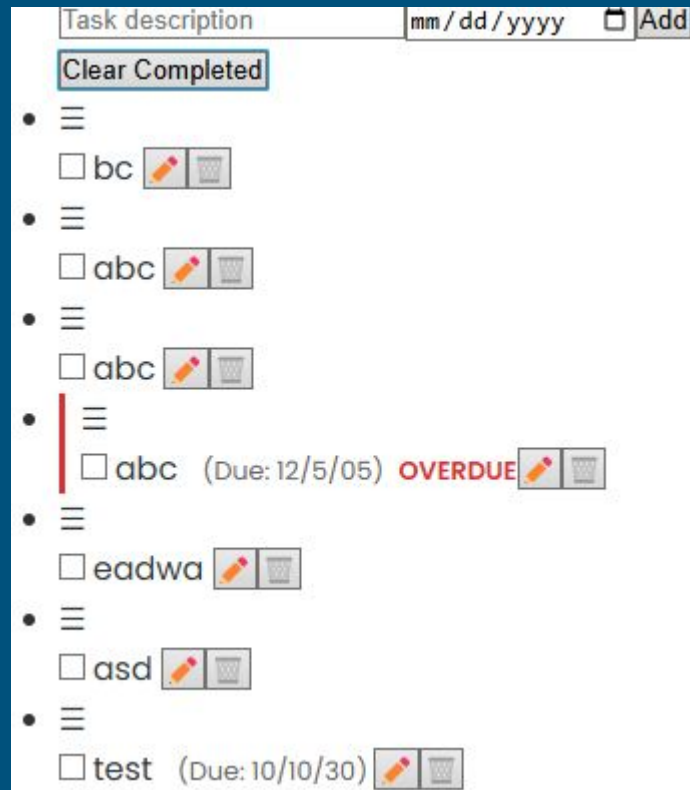
Logika tego została napisana w 100% sama.



16-17. Usunięcie wszystkich wykonanych zadań

Nowa funkcjonalność - usuwanie wszystkich wykonanych zadań.

Ponownie, największym problemem był problem w poruszaniu się po strukturze całego projektu. Niektóre elementy nie były poprawnie filtrowane, co wymagało poprawek.

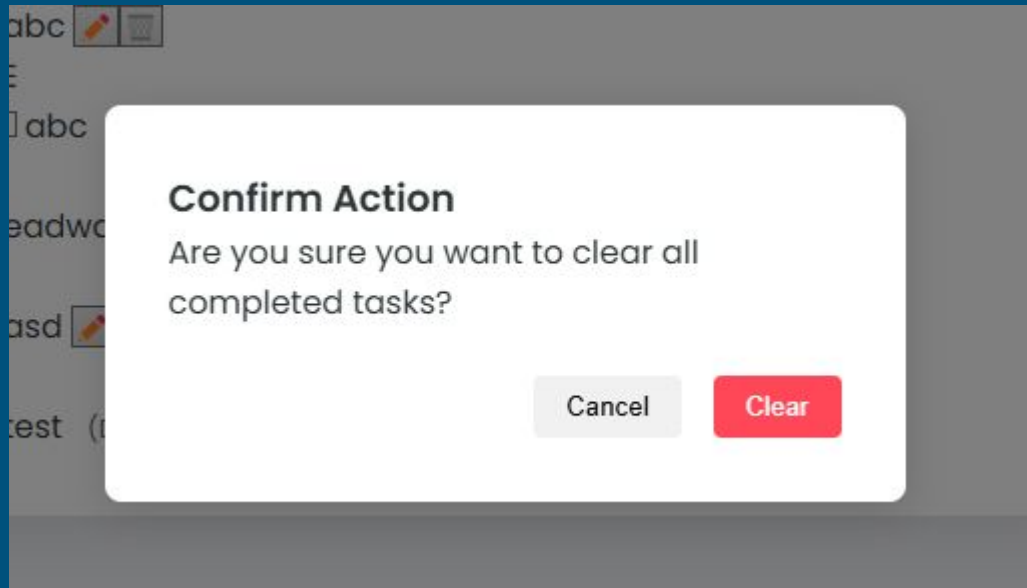


18-19. Modal z potwierdzeniem usunięcia

Deepseek dodał nową funkcjonalność, tworząc nowy komponent - bardzo poprawna technika.

Co ciekawe, po raz pierwszy w projekcie użył poprawnej instrukcji warunkowej w template:

```
@if (showModal) {  
  <app-confirmation-modal  
    (confirmed)="clearCompleted()"  
    (canceled)="showModal = false"  
  ></app-confirmation-modal>  
}
```



Zalety

Deepseek jest jedynym w pełni darmowym modelem, stosował on, jako jedyny model, dobry podział aplikacji na różne komponenty (modal, service, todolist, todoform etc.).

Nie było również problemów z funkcjonalnością przycisków, logiki.

W niektórych miejscach, wykorzystał nowoczesne podejścia (@if zamiast ngIf).

Wady

Deepseek pozostawiał po sobie najmniej komentarzy. Jednak największą wadą tego modelu było umieszczanie całej logiki komponentu w jednym pliku - powoduje to niską czytelność i trudności w skalowaniu projektu.

Dominowały przestarzałe techniki, jednocześnie jako jedyny model, nie potrafił wskazać najbardziej aktualnej wersji.

Styl wizualny był najmniej rozbudowany, a sam Deepseek miał problem z łączeniem styli (.scss) z komponentem (styleUrls).

Ocena

1. Jakość kodu

Średnia - zastosowano nowoczesne rozwiązania, ale cała logika komponentów była upchana w jednym pliku, co mocno utrudnia czytelność i przyszłe utrzymanie. Brakowało również komentarzy w kodzie.

2. Wydajność aplikacji

Poprawna - aplikacja działała sprawnie, bez problemów

3. Dokładność generowanego kodu

Dobra – funkcjonalności były zaimplementowane prawidłowo, ale struktura zaczynała się gubić w miarę rozrastania aplikacji.

Ocena c.d.

4. Styl wizualny

Aplikacja była najbrzydsza spośród wszystkich, brak spójności wizualnej, problemy ze stylami między komponentami (styleUrls były źle używane lub pomijane).

5. Występowanie problemów technicznych

Umiarkowane – użyto głównie przestarzałych dyrektyw (*ngIf, *ngFor), choć w jednym miejscu pojawiło się nowoczesne @if. Problemy ze strukturą komponentów zaczynały wpływać na skalowalność.

6. Koszta

DeepSeek jest w pełni darmowy, co jest dużym atutem.

PODSUMOWANIE

PODSUMOWANIE

Kryterium	chatGTP	Claude	DeepSeek
1. Jakość kodu	Wysoka (<i>nowoczesny, poprawny</i>)	Niska (<i>złe funkcje</i>)	Średnia
2. Wydajność aplikacji	Bardzo dobra	Brak działania	Dobra
3. Dokładność funkcjonalna	W pełni działająca	Nie działała	Poprawna
4. Styl wizualny	Średni	Bardzo ładna	Najbrzydsza
5. Problemy techniczne	Przestarzałe dyrektywy	Krytyczne (funkcje nie działały)	Przestarzałe dyrektywy
6. Koszta	\$ 20 USD / msc	\$ 20 USD/msc	0 USD

Powyżej przedstawiona tabela przedstawia krótkie podsumowanie przetestowanych modeli.

Który model okazał się najlepszy?

Claude, pomimo najwyższych oczekiwań, okazał się najgorszy - celem projektu, było ukazanie, który model dla osoby z brakiem doświadczenia programistycznego okaże się najlepszym wyborem. Claude pomimo wielu prób napraw, nie podołał wyznaczonemu zadaniu.

Deepseek jako darmowy, szybki prototyp okazał się konkurencyjnym zamiennikiem, aczkolwiek przy szybko rozrastających się aplikacjach, prawdopodobnie nie byłby najlepszym rozwiązaniem.

ChatGPT utworzył stabilny, skalowalny i działający projekt, który nadaje się do realnego użycia. Gdyby nie przestarzałe dyrektywy i zbyt małe rozbieżności projektu na mniejsze komponenty, byłby bardzo dobrym modelem dla Vibe Coderów.