Workshop II Copilot: Skriv Effektive Prompter til VSCode GitHub Copilot

## Mål

Ved afslutningen af denne workshop vil du:

- Forstå opbygningen af en effektiv Copilot-prompt

- Øve dig i at skrive prompts, der genererer præcis kode

- Lære at forfine prompts for bedre resultater

## Forudsætninger

- GitHub Copilot installeret og aktiveret i VS Code

- Grundlæggende programmeringsforståelse i JavaScript

## Øvelser

Øvelse 1: Prompt-Engineering (25 minutter)

Lad os se, hvordan forskellige detaljeringsniveauer i prompts påvirker Copilots output.

1. Opret en ny fil med html login form og ny fil kaldet `user\_validation.js`

2. Prøv disse prompts i rækkefølge og observer, hvordan Copilots forslag forbedres:

\*\*Basal Prompt:\*\*

```javascript

// validere email

```

\*\*Bedre Prompt:\*\*

```javascript

// funktion til at validere email-format

```

\*\*Bedste Prompt:\*\*

```javascript

// Funktion validateEmail der tjekker:

// - indeholder @ og mindst ét punktum

// - ingen mellemrum tilladt

// - minimumslængde på 15 tegn

// returnerer true hvis valid, ellers false

```

💡 \*\*Diskussion\*\*: Bemærk hvordan mere kontekst og krav giver mere præcise forslag.

Øvelse 2: Prompt-Mønstre (15 minutter)

Prøv disse effektive prompt-mønstre:

1. Opret en ny fil kaldet `data\_transformer.html`

2. Skriv en prompt ved hjælp af hvert mønster:

\*\*Input/Output-mønster:\*\*

```javascript

// input: array af tal [1,2,3,4,5]

// output: summen af kvadrater [1,4,9,16,25]

// funktionens navn: squareNumbers

```

3 Skriv en prompt at vise array frem i html

Øvelse 3: Forfining af Prompter (15 minutter)

Nogle gange giver din første prompt ikke det ønskede resultat. Lad os øve os i at forfine:

1. Start med denne prompt:

```javascript

// generer adgangskode

```

2. Forfin den gradvist:

```javascript

// generer sikker adgangskode

```

```javascript

// generer sikker adgangskode med:

// - minimum 12 tegn

// - mindst ét stort og ét lille bogstav

// - mindst ét tal og ét specialtegn

```

Du har gennemført denne lab, når du:

- Har genereret kode ved hjælp af alle tre promptniveauer i Øvelse 1

- Har oprettet funktioner ved hjælp af begge prompt-mønstre i Øvelse 2

- Har øvet dig i prompt-forfining i Øvelse 3

## Tips

- Vær specifik om funktionsnavne, parametre og returværdier

- Inkluder eksempler, hvis adfærden ikke er indlysende

- Brug klar formatering i dine prompts (punktopstillinger, linjeskift)

- Hvis du ikke kan lide Copilots første forslag, prøv at forfine din prompt

## Refleksionsspørgsmål

1. Hvilket prompt-mønster fungerede bedst for dig?

2. Hvordan påvirkede detaljegraden i dine prompts Copilots output?

3. Hvilke forskelle bemærkede du mellem din oprindelige og din forfinede prompt?

---

Kilder:

Børne- og Undervisningsministeriet. (2022). *Didaktiske principper i erhvervsinformatik*. EMU. Retrieved January 29, 2025, from <https://emu.dk/eud/erhvervsinformatik/didaktiske-principper>