Kan du bygge en digital løsning med AI?

Introduktion

2-6 lektioner

> *I denne aktivitet skal I være digitale innovatører! I skal løse et ægte problem fra jeres egen hverdag ved at skabe en fungerende webapp. Men i stedet for at skrive tusindvis af linjer kode fra bunden, skal I bruge* ***AI til hjælp*** *(vibecoding) – altså at styre og instruere en kunstig intelligens til at skrive og rette koden for jer – og måske lærer om webudvikling undervejs.*

Sikkerhed

VIGTIGT – læs dette sikkerhedsafsnit, inden du går i gang!

A) Opmærksomhed

Når I skal kode apps, især med AI, er der et par forholdsregler I skal tage højde for, før app'en må udgives.

B) Forholdsregler

- **AI:** Brug aldrig personlige oplysninger (navne, adresser, adgangskoder) i jeres instrukser (prompts) til AI'en.
- **Data:** Vores apps skal gemme data i browseren via localStorage. Det betyder, at data bliver på den enkelte enhed og ikke sendes til en usikker server på nettet.
- **Ophavsret:** Tjek at I ikke bruger beskyttede billeder eller logoer uden

tilladelse.
- **Etik:** Tjek at appen ikke indeholder følsom information i koden.

C) Førstehjælp

Skulle I komme til at glemme en eller flere af forholdsreglerne, bedes I kontakte jeres lærer, så vedkommende kan hjælpe med det fixe det. I værste fald skal I starte forfra.

D) Oprydning

Når I er færdige med jeres projekt, logger I ud og sletter alt der ikke skal gemmes mere. Hvis I gerne vil gemme jeres kode lokalt, kan I zippe det og uploade til jeres drev eller andet sted.

□□ Materialer

I skal bruge følgende:

- Computere med internetadgang.
- Adgang til **SkoleGPT** (eller anden godkendt sprogmodel)
- Adgang til en online kodemiljø-browser (fx Edbit WebSkole).
- Papir, tuscher og post-its til wireframes og brugertest.
- Evt. kan eleverne udgive deres webapp gennem Edbit AppSkole

□ Vejledning

Følg vejledningen trin for trin:

1) Opstart og brainstorm

1. I arbejder sammen i makkerpar.
2. Dagens mission er at gå fra ét konkret problem i jeres hverdag til en færdig webapp, som kan installeres eller bruges direkte på mobil eller computer.
3. **Brainstorm:** Find et simpelt hverdagsmønster, I vil forbedre.
    - *Eksempler:* En “hvad skal vi spise”-generator, en lektie-tracker, et

lykkehjul til pligter eller en simpel pointtæller til spil.

4. Tegn en **wireframe** (en hurtig papirskitse af skærmbilledet) med knapper, felter og overskrifter.
5. Opret projekt eller lav en plan med Edbit Skoleprojekt.

2) Kodning med AI

Gode råd:

- Bed AI skrive koden på **engelsk**, så den ikke kommer til at oversætte syntaksen til danske ord, med mærkelige bogstaver.
- Brug "**KISS**"-princippet (Keep It Simple Stupid!). For at formindske fejl og ikke miste overblikket, så tag ét element ad gangen og få lavet koden til det. Sæt elementerne sammen til sidst.
- **Debug** ved at prompte koden tilbage og fortæl hvad du tror fejlen er.
- Prompting er en **proces**. Hvis det fejler, så prøv igen på en anden måde.

Opret kode

1. Åbn jeres valgte AI (fx SkoleGPT) og kodeditor (fx Edbit WebSkole).
2. **Første prompt:** Bed AI'en om at bygge grundstrukturen ud fra jeres papirskitse.
    - *"Skriv koden til en simpel webapp. Alt skal samles i én HTML-fil (inklusive CSS og JavaScript). Appen skal have følgende funktioner: [beskriv jeres idé]."*

3. Kopier koden fra AI'en over i kodeditoren og se jeres app vågne til live!

Forbedring af kode

1. Vis jeres første, rå udgave af appen til et andet par.
    - Tjek: Virker de grundlæggende knapper?

2. Nu skal koden finpudses, designet skal gøres lækkert, og funktionerne

skal udvides via **iterativ forbedring med AI**.

3. **Lokal lagring:** Hvis jeres app skal gemme data, så husk at instruere AI'en i at bruge localStorage, så appens data ikke forsvinder, når siden opdateres.
    - *"Opdater koden, så den gemmer [f.eks. min to-do liste] i browserens localStorage."*

4. Test appen grundigt og ret til via **prompts**:
    - *Design: "Gør baggrunden mørk, knapperne runde og teksten større."*
    - *Funktionalitet: "Tilføj en knap til at slette et punkt igen."*

5. **Brugertest:** Byt computer med et nabopar i 15 minutter.
6. Observer hvordan de bruger jeres app:
    - Hvad overser de?
    - Hvad driller?
    - Notér 2 konkrete ting, der skal forbedres.

7. Gå tilbage til AI'en og giv den instruktionerne til at rette de fejl, I fandt under brugertesten.

3) Gem og udgiv

1. Jeres app skal gøres klar til udgivelse, så rigtige brugere kan tilgå den via et link eller en QR-kode. I kan fx udgive gennem WebSkolen ved at klikke "Del" i højre hjørne, hvorefter webappen udgives på Edbit AppSkole. Når projektet er del får I et link ligesom "https://portal.edbit.dk/appskole/TKGYl8jX_7z".
2. Generer evt. en QR-kode, der peger direkte til jeres nye webapp.
    - **"App Store":** Alle lægger deres skærm frem eller hænger en QR-kode op på væggen.
    - Gå rundt, prøv hinandens apps, og giv digital feedback.
    - **Projekttanke:** Overvej hvordan denne app (eller en videreudvikling) vil kunne indgå som dit innovative produkt til fx 9. klasses projektopgave!

Opsamling

Prøv at svare på følgende spørgsmål:

- Hvordan forklarede I jeres idé til AI'en? Hvilke oplysninger var I nødt til at skære væk (*abstrahere*) for, at AI'en kunne forstå opgaven?
- Hvad er fordelene og ulemperne ved at brugen gemmes i localStorage i stedet for på en central server/database? (Tænk på datasikkerhed vs. funktioner).
- Hvad betyder det for fremtidens arbejdsmarked, at alle kan "kode" en app via AI uden at kende det reelle programmeringssprog?
- På en skala fra 1-10, hvor meget løser jeres app det oprindelige hverdagsproblem? Hvilke funktioner mangler der, for at den ville kunne sælges i en rigtig App Store?

Faglig forklaring+

Lærervejledning+

Aktiviteten er tilrettelagt ud fra teknologiforståelse i folkeskolen med særligt fokus på **Computationel tænkning**, **Digital design og designprocesser** samt **Teknologianalyse**.

Formålet er at afmystificere programmørrollen ved at lade AI håndtere syntaksen, mens eleverne styrer den overordnede logik, brugeroplevelsen (UI/UX) og den innovative problemløsning.

Lærerens rolle

1. **Lektion 1-2 (Scope-styring):** Elever vil ofte bygge et nyt *Fortnite* eller *TikTok*. Tving dem ned i målestok. Appen skal kunne løse ét helt simpelt problem.
2. **Lektion 3-4 (Prompting & Debugging):** Når koden ikke virker, skal du undgå selv at rette koden. Lær i stedet eleverne at kopiere fejlmeddelelsen fra browserens konsol eller beskrive fejlen direkte til AI'en: *"Når jeg trykker på knappen X, sker der ingenting. Hvad er der galt i koden?"*
3. **Lektion 5-6 (Udgivelse):** Hvis ønsket er at udgive elevernes produkter, skal læreren enten:

    - Lade eleverne bruge Edbit Kode-editor og gemme/udgive derigennem
    - På forhånd at have en mappe klar på webserveren/elev-hjemmesiden, hvor elevernes HTML-filer hurtigt kan placeres, så de oplever værdien af at udgive et funktionelt produkt.

Kildehenvisning+

- https://portal.xn--forst-qra.dk/course/asXQ-hverdagens-sm-a-og-store-problemer