For at være en god elev skal man vide, hvordan man lærer. Dette kræver både koncentration og tid, men hvad gør man, når koncentrationen svigter, eller man har svært ved at holde fokus i længere perioder?

Pomodoro-teknikken er en studiemetode, der opdeler arbejdstiden i intervaller – typisk 25 minutters fokuseret arbejde efterfulgt af en 5-minutters pause, med en længere pause efter fire arbejdssessioner. Ikke alle har en Pomodoro-timer til rådighed, og mange ønsker ikke at bruge penge på et fysisk objekt. Derfor kan en gratis app være en enkel løsning på dette problem.

Dette projekt vil fokusere på følgende problemstilling:

Hvordan kan vi ved hjælp af de metoder og værktøj fra programmering lave et Pomodoro program og hvordan kan en Pomodoro-app forbedre produktiviteten og koncentrationen hos brugere, og hvilke faktorer spiller en rolle i dens effektivitet?

For at besvare denne problemstilling vil projektet undersøge:

* Hvad Pomodoro-teknikken er, og hvordan den fungerer i en digital kontekst.
* Hvordan en Pomodoro-app påvirker brugernes arbejdsvaner og produktivitet.
* Hvilke begrænsninger og potentielle forbedringer der kan implementeres for at optimere en Pomodoro-apps effektivitet.

Andre mulige projektemner kunne inkludere:

* Et tegneprogram.
* Et virtuelt akvarium.
* Et program til at lære tegnsprog (helst ikke)

Kravspecifikation:

Pomodoro app’en skal kunne udføre en række funktioner:

Hårde krav indebærer følgende:

* Sætte en timer på 25 minutter.
* Tracke hvor mange ”pomodoro” der er gået
* Inkorpere en tracker af hvor mange ”pomodoro” der er gået og indføre en større pause når der er gået 4 ”pomodoro”
* Tracke for hver dag hvor mange ”pomodoro” der bliver brugt pr. dag
* Eksporter den tracket data i form af en læsbar fil.

Bløde krav bliver hermed:

* At have en brugevenlig ui
* At have en ui som appelerer til målgruppen

**Teori og metode**

Pomodoro-teknikken er en tidsstyringsmetode udviklet af Francesco Cirillo i slutningen af 1980'erne. Teknikken går ud på at opdele arbejdstiden i fokuserede intervaller også kaldt Pomdoro’er, typisk på 25 minutter, efterfulgt af en kort pause på 5 minutter. Efter fire sådanne arbejdsintervaller tages en længere pause på 15-30 minutter. Metoden er opkaldt efter det italienske ord for "tomat", da Cirillo oprindeligt brugte et køkkenur formet som en tomat til at holde styr på tiden.

En af de største fordele ved Pomodoro-teknikken er dens evne til at øge produktiviteten og koncentrationen. De faste tidsintervaller gør det lettere at undgå overspringshandlinger, da man arbejder intenst i en kort periode og derefter belønnes med en pause. Derudover kan teknikken hjælpe med at reducere mental udmattelse, fordi regelmæssige pauser sikrer, at hjernen får tid til at restituere. Den strukturerede tilgang kan også gøre store opgaver mere overskuelige ved at bryde dem ned i mindre dele.

På trods af dens fordele har Pomodoro-teknikken også nogle ulemper. En af de mest nævneværdige er, at den ikke altid passer til arbejdsopgaver, der kræver længere perioder med fordybelse. For eksempel kan en kompleks opgave kræve mere end 25 minutters uafbrudt arbejde, og en pause kan i sådanne tilfælde virke forstyrrende. Desuden kan teknikken være svær at implementere i arbejdsmiljøer, hvor uforudsete afbrydelser er almindelige, da det kan være vanskeligt at holde fast i de stramme tidsintervaller.

Samlet set kan Pomodoro-teknikken være en effektiv metode til at forbedre fokus og arbejdsflow, men den er ikke nødvendigvis den bedste løsning for alle typer af opgaver og arbejdssituationer.

I dette projekt anvendes pseudokode, rutediagrammer og klassediagrammer for at gøre udviklingen mere struktureret og effektiv. Programmet er objektorienteret og udviklet i Processing’s Java for Android, hvilket sikrer en interaktiv og brugervenlig mobilapplikation.

Appen fungerer som en Pomodoro-timer, der automatisk skifter mellem 25 minutters arbejde og 5 minutters pause, med en længere 30-minutters pause efter fire cyklusser. En rød cirkel fungerer som visuel timer, og en start/stop-knap giver brugeren kontrol over nedtællingen. Der afspilles en lydnotifikation, når en session afsluttes, og appen registrerer den samlede studietid.

Projektet inkluderer en modulær kodeopbygning med veldefinerede klasser samt grafiske diagrammer for at sikre en overskuelig og veldokumenteret udviklingsproces.

**Hovedtekst/Analyseafsnit**

* **Dokumentation for overvejelser i planlægningsfasen.**

Inden projektet overhovedet begynde er det vigtigt at planlægge. Som det første blev der oprettet en GitHub organisation. I denne organisation kunne der laves en projektforløb

Efter dette blev der begyndt et rutediagram. Dette diagram er den første indblik i hvordan

* **Redegørelse for algoritmen der udgør programmets ’kerne’, brug pseudokode og rutediagrammer.**

START PROGRAM

INITIALISER:

- Opret en Pomodoro-instans

- Opret en Button-instans i midten af skærmen

- Indlæs lydfil ("alarm.mp3")

- Sæt isRunning = FALSE (timer er stoppet)

LOOP (DRAW-FUNKTION):

- Ryd skærmen (Baggrund)

- Opdater Pomodoro (hvis isRunning = TRUE)

- Tegn timerens cirkel og vis tid tilbage

- Tegn start/stop-knappen

HVIS BRUGER TRYKKER PÅ KNAPPEN:

- Skift isRunning (TRUE <-> FALSE)

- Opdater knappens tekst ("Start" eller "Stop")

- Start eller stop timeren i Pomodoro

KLASSE: Pomodoro

INITIALISER:

- Sæt arbejdstid til 25 min

- Sæt kort pause til 5 min

- Sæt lang pause til 30 min

- Sæt cycles = 0 (antallet af Pomodoro-sæt)

- Sæt totalStudyTime = 0 (tid brugt på arbejde)

- Sæt isBreak = FALSE (start i arbejdsmodus)

- Sæt timer til arbejdstid

- Sæt lastMillis til nuværende tid

FUNKTION: update()

HVIS isRunning:

- Beregn tid der er gået (elapsedTime)

- Træk elapsedTime fra timer

- Opdater lastMillis

HVIS timer <= 0:

- HVIS isBreak:

- Skift til arbejdstid

- Inkrementer cycles

- HVIS cycles % 4 == 0 → Sæt lang pause

- ELLERS → Sæt normal arbejdstid

- ELLERS (hvis arbejdsperiode er slut):

- Tilføj arbejdstid til totalStudyTime

- Skift til pause

- Sæt timer til kort pause

- Afspil alarm-lyd

FUNKTION: display()

- Tegn rød cirkel som visuel timer

- Vis tid tilbage som tekst over cirklen

FUNKTION: toggleTimer(state)

- Sæt isRunning til state

- Opdater lastMillis

FUNKTION: formatTime(seconds)

- Konverter sekunder til MM:SS-format

KLASSE: Button

INITIALISER:

- Gem position, størrelse og tekst

FUNKTION: display()

- Tegn en hvid knap med sort kant og tekst

FUNKTION: isClicked(x, y)

- RETURNER TRUE, hvis brugeren har trykket inden for knappen

FUNKTION: setText(newText)

- Opdater knaptekst

SLUT PROGRAM