

Fraværssystem
Af Jonas Larsen, Benjamin Thomsen og Kim Olesen

UCN, Sofiendalsvej 60

Vejleder: Steffen Rahbek Vutborg

Anslag: 61675



Figur 1 - Database forsidebillede [1]

Indholdsfortegnelse

Indledning.....	4
Problemformulering	4
Kravspecifikation	5
Introduktion til projektet.....	5
Funktionelle krav.	5
Ikke funktionelle krav.	5
Projektet målsætninger	5
Afgrænsning.	6
Planlægning/arbejdsmetoder.....	6
Kanban.....	6
Gannt diagram.....	7
Virksomhedsbeskrivelse af UCN	7
Organisationsdiagram UCN	7
PEST-analyse.....	8
Politiske faktorer	8
Økonomiske faktorer.....	8
Sociokulturelle faktorer	9
Teknologiske faktorer	9
SWOT/TOWS-analyse	9
SMART – Strategiske mål.....	9
Boston analyse.....	10
Adizes lederroller.....	11
Kulturanalyse	12
Mintzbergs organisationsformer.....	12
Trompenaars og Hampden-Turners kulturtyper	13
Payback projection	14
Delkonklusion	15
Teknisk analyse	15
Udviklingsplatform	15
Opsætning af pi	15
Boot	16
RFID.....	17
RFID i hverdagen.....	18
Hardware	18

I2C protokollen – Inter Integrated Circuit	18
SPI-protokollen – Serial Peripheral Interface	19
El diagram – MFRC 522 - PI.....	20
El diagram – LCD – PI	20
Software	21
MariaDB.....	21
Opsætning af MariaDB	21
Bruger i MariaDB	22
Backup	24
HeidiSQL	25
Opsætning af databasen i HeidiSQL	25
Opsætning af tabeller	25
Studerende	25
Skemaitt.....	26
Itt og dat	27
Sikkerhed	27
Generel sikkerhed.....	28
SQL-Injections.....	28
Parametriserede forespørgsler	28
Sikkerhed i Administrator Interface.	28
Forbindelse til MariaDB fra Raspberry Pi.	28
Fysisk sikkerhed	30
Versioner af interface	30
Version 1 - CLI Interface	30
Version 2 – GUI Interface	30
Flow fra RFID-tag til DB.....	31
DB-diagram.....	32
Gennemgang af koden	33
Tjekin.py	33
Test	37
Test af lcd display.	43
Konklusion	44
Perspektivering.....	45
Bibliografi.....	46

Indledning

Uddannelsesinstitutioner i Danmark har et stort problem. Frafald på uddannelserne giver panderynker på selv de mest garvede undervisere, samt ledelsen på institutionerne. Det er en generel tendens på uddannelsesinstitutioner over hele landet, at frafaldsprocenten er meget høj. Værst står det til på erhvervsuddannelserne. I perioden fra 2015-2020 har 47% af dem der startede på en erhvervsuddannelse, ikke gennemført den. [2]. Frafaldsprocent har været nogenlunde stabil siden 2011 på 50%.

Dette gør sig også gældende på UCN. Generelt er der en frafaldsprocent på omkring 30%, hvilket er lavere end på den generelle erhvervsuddannelse. Dette kan forklares ved at UCN tilbyder en blanding af forskellige erhvervs-, og professionsbachelor uddannelser. Hvor professionsbachelor uddannelsen har et lavere frafald. Derudover oplever UCN rekrutteringsproblemer. UCN har oplevet et fald på 9% i elev optaget, ift. til 2021 niveauet. UCN vurderer i deres 2022 årsrapport, at dette fald vil forsætte i 2023, og i de kommende år. [3] Det giver anledning til bekymring. Da UCN mister økonomisk støtte fra staten/kommunen. Dette kan medføre at UCN kan ende med at lukke ned for uddannelsesretninger. Det formodes, at der er en forbindelse mellem fravær og frafald.

På UCN Sofiendsdalsvej, bliver elevernes fravær registreret manuelt, via et Excel ark. Dette system skal automatiseres, så eleverne selv skal registrere fremmøde via RFID med nøglekort eller polet. Registreringen skal dernæst sendes til en database, hvor underviserne kan hente fraværs data på den enkelte elev for at hjælpe udsatte elever før, de falder fra uddannelsen. Fraværssystemet skal bruges som et samtaleværktøj for medarbejderne på UCN.

Problemformulering

På baggrund af overstående indledning, har vi formuleret en problemformulering:

- Hvordan kan vi lave et RFID baseret it-system, der kan behandle studerendes fravær?

Grundet ressourcerne der bliver brugt i det danske skolesystem, er det særdeles vigtigt at underviserne har et værktøj de kan bruge i dagligdagen til at indhente data på elever. For på den måde, at kunne "gribe" elever, før fraværet udvikler sig til et frafald. Projektet er opbygget af de 4 fag, som vi har modtaget undervisning i. Fagene er som følgende:

- Netværksteknologi
- Programmering
- Indlejrede systemer
- Projektstyring og forretningsforståelse

Kravspecifikation

Introduktion til projektet.

UCN ønsker sig et system til fremmødere registrering. I den nuværende løsning registrerer underviserne de studerendes fremmøde i Excel. UCN vil gerne frigive nogle ressourcer fra underviserne, så de kan fokusere på deres kerneopgaver, og i stedet ligge ansvaret for fremmødere registreringen over på de studerende. Med den nuværende løsning, er det svært at have et overblik over den studerendes fravær, da underviserne dokumenterer i hver deres Excel ark. I dette projekt skal fremmødere registreringen automatiseres og bruges til at forebygge fravær, før det bliver til frafald.

Funktionelle krav.

- Udtrække fravær på den enkelte studerende.
- Udtrække fravær klassevis.
- Oprette ny studerende.
- Oprette ny klasse.
- Slette klasse.
- Slette studerende.
- Redigere studerendes fremmøde.
- Bruger interface, som kan bruge ovenstående funktioner.

Ikke funktionelle krav.

Krav til løsningen:

- Systemet skal selv boote, ved strømsvigt.
- Systemet skal være hurtigt, eks. hente fremmøde for en klasse på under 2 sekunder.
- Systemet skal programmeres i Python/SQL.
- Systemet skal kommunikere med database.
- Systemet skal bruge en sikkerforbindelse til databasen

Krav til projektet:

- Tid – projektet skal afleveres d.12. juni.
- Kvalitet – Prototypen skal være funktionel ved projekt afslutning.

Projektet målsætninger

Formål:

- Systemet er et værktøj til underviserne.
- Flytte ressourcer fra fraværsregistrering til kerneopgaver.

Mål:

- Mindske frafald på uddannelser.
- Vækste UCN som organisation.

Afgrænsning.

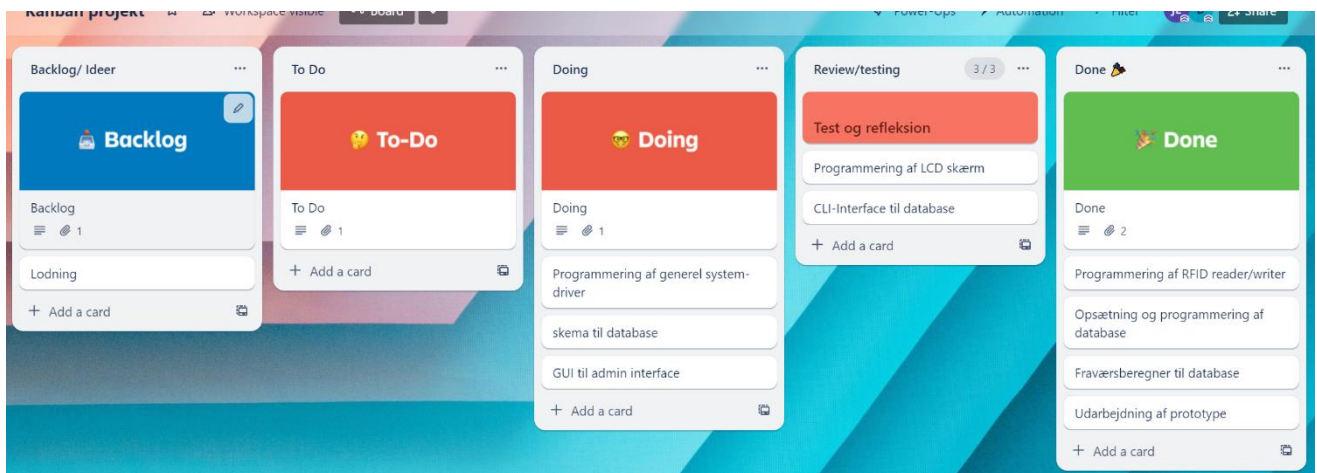
Projektet er ikke blevet implementeret med redundans for at undgå øget kompleksitet. I erhvervsmæssig sammenhæng ville redundans være nødvendig for at opretholde en konstant opetid af serveren og udstyret til at generere korrekte statistikker. Adgangsstyring er ikke blevet implementeret på grund tidsbegrænsninger og manglende erfaring hos udviklerne. Myndighederne ville kræve adgangs styring, hvis systemet skulle implementeres på bredere skala. Rapporten tager ikke højde for GDPR-regler eller lignende, da dette ville kræve yderligere ressourcer og omfattende undersøgelser.

Valget af Raspberry Pi som udviklingsplatform er baseret på erfaringer fra to semestre på it-teknolog uddannelsen. Andre alternativer er ikke blevet undersøgt, men det antages, at en mikrocontroller som Raspberry Pico/Zero ville være en billigere løsning både komponent, samt strømmæssigt.

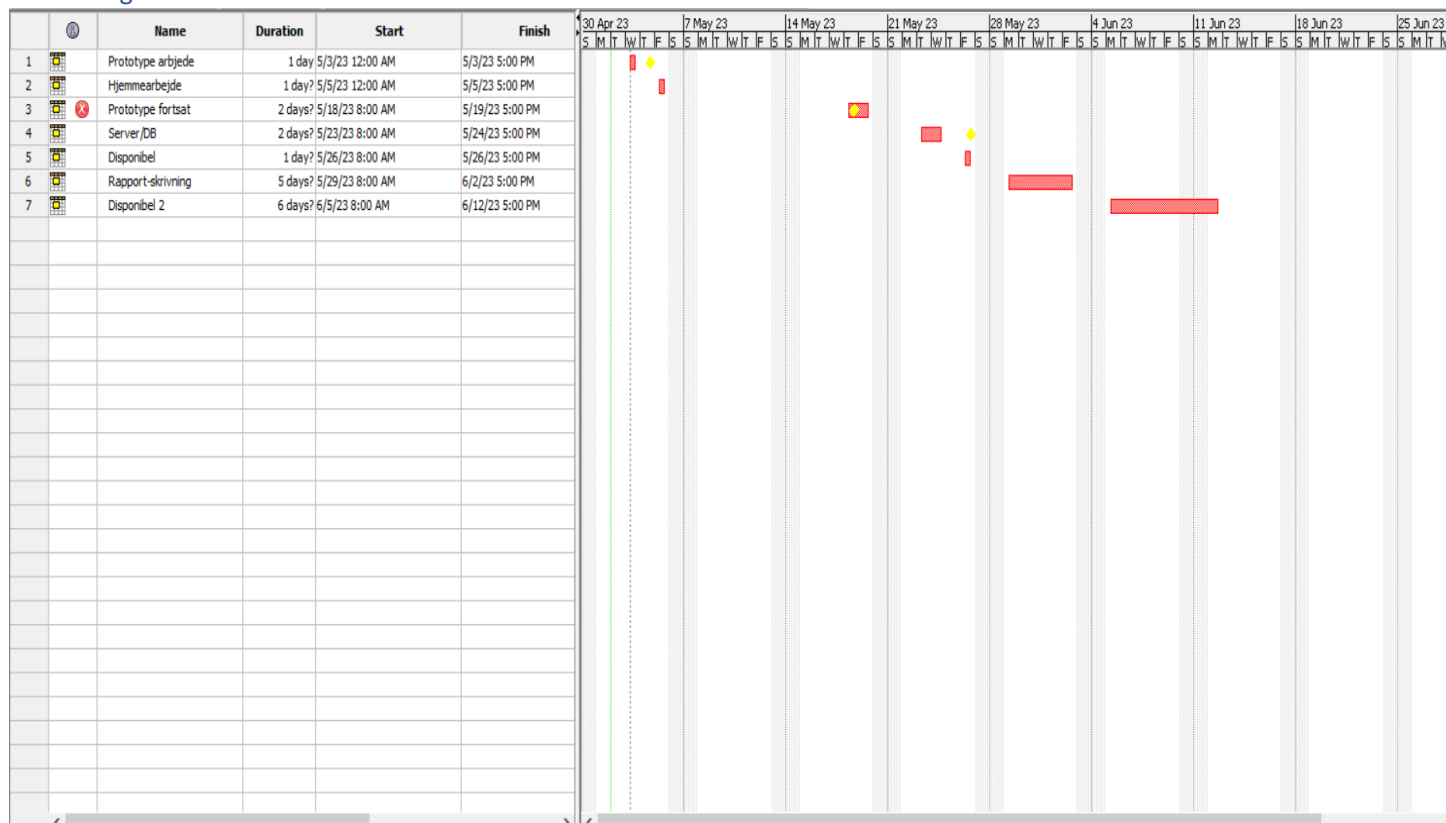
Planlægning/arbejdsmetoder.

Kanban

Kanban board hjælper med at planlægge et projekt. I dette tilfælde er projektet et RFID-system til UCN. Et Kanban board er et værktøj, hvor man deler arbejdsopgaverne op. Boardet er delt op i 5 kategorier: Backlog, to-do, doing, test/review og done. Backlog er til idéer, der muligvis vil blive brugt til at løse projektet. To-do er de godkendte idéer fra backloggen. Doing er de idéer, der er i gang med at blive lavet færdige. Test/review er der, hvor de tester om idéerne virker, så ryger de i kategorien done. Det kan også være de skal en tur tilbage i doing eller idéen skal slettes helt, hvis det ikke virker ordentligt.



Gantt diagram



Her et udsnit af det gannt diagram, som projektgruppen lavede i starten af perioden for at give et overblik over tidshorizonten. Der er ikke taget højde for aflyste dage og fritid. Dette er udelukkende selvstudie-dage og helligdage. De disponible dage, er dage hvor projektgruppen har mødtes og drøftet det generelle workflow og udført de opgaver der har været tilgængelige og passende

Virksomhedsbeskrivelse af UCN

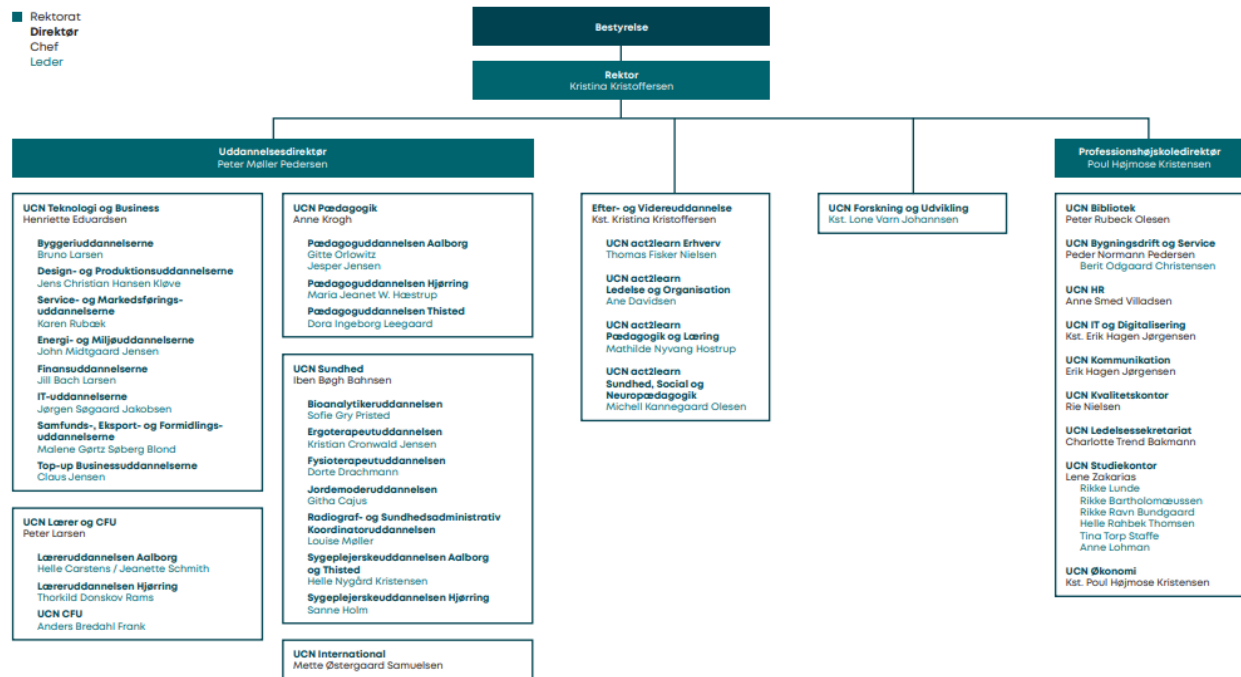
UCN udbyder 40 forskellige uddannelser, hvoraf 15 er erhvervsuddannelser, 24 professionsbachelors og 1 erhvervsuddannelse. UCN har skoler i Thisted, Hjørring og Aalborg. UCN har omkring 9000 studerende, og ca. tilsvarende på efter- og videreuddannelse. UCN har omkring 900 medarbejdere, fordelt på 3 skoler. [4]. UCN har i 2022 haft en omsætning på 738.6 millioner, hvilket gør UCN til en stor virksomhed. UCN har en egenkapital på 225 millioner, hvilket viser at økonomien er stabil. [3]

Organisationsdiagram UCN

Formålet med at opdele en organisation efter de forskellige principper er en klar ansvarsfordeling. Dette medfører, at medarbejdere ikke får modsatrettede opgaver. Samtidig med at der ikke er konkurrence fra lederne om den enkelte medarbejder. Den mest brugte er linjestabs princippet.

UCN's organisation er opdelt efter linjestabs princippet. Fordelen ved at UCN bruger linjestabs princippet er, at eksperternes viden bliver udnyttet til fulde. Og ansvarsfordelingen er klart defineret. Ulempen er mulige konflikter mellem linje og stab. UCN er en dyb organisation, dette ses ved mange forgreninger og mange mellemledere. Risikoen ved en dyb organisation er lange kommunikations veje, hvilket bidrager til langsommelige beslutningsprocesser.

Organisationsdiagram



Figur 22 - Organisationsdiagram, [5]

PEST-analyse

Politiske faktorer

UCN er en selvejende organisation. Flertallet af organisationer i det offentlige er politisk styrede og finansierede. Dette gør sig også gældende for UCN. Hvert 4. år forhandles der en strategisk rammeaftale mellem UCN's bestyrelse og uddannelses- og forskningsministeren. Målene i den strategiske rammeaftale tager udgangspunkt i UCN's kerneopgaver. Som er at uddanne kompetent arbejdskraft til det private og offentlige arbejdsmarked. Politiske faktorer er derfor afgørende i en organisation som UCN. Politisk styrede organisationer skal være omstillingsparate, da verdensbilledet hurtigt kan forandre sig. Dette kunne f.eks. være udflytning af uddannelsesretninger til yderområder, eller en politisk beslutning om at nedlægge uddannelser på fremmedsprog.

Økonomiske faktorer

UCN er afhængig af de offentlige finanser. Det er politisk bestemt, hvor mange penge UCN modtager af staten, for at uddanne de studerende. I 2019 stod staten/kommunerne for 92,3% af den samlede finansiering til det danske uddannelsessystem. [6]. UCN har en frafaldsprocent på 30%, hvilket betyder, at UCN mister tilskud for millioner, når studerende forlader studiet grundet fravær. Grundet den høje frafaldsprocent, kan UCN være nødsaget til at reducere omkostninger i forbindelse med personale, videreuddannelse/efteruddannelse af undervisere og i værste fald lukke specifikke uddannelsesretninger ned. Hvilket kan medføre, at kvaliteten på uddannelserne falder.

Fra 2023 bliver reglerne omkring tilskud ændret. Uddannelsesinstitutionerne modtager ikke taxametertilskud for elever der gennemfører, men i stedet taxametertilskud for aktive og indskrevne elever. [2]

Sociokulturelle faktorer

Ifølge underviserne på UCN er der et godt arbejdsmiljø. Underviserne er glade for at møde på arbejde, og de har et godt sammenhold. Det formodes, at uddannelsesniveaue generelt er højt blandt underviserne, og at de har mulighed for at videre, -efteruddanne sig, hvis de ønsker det. Flere fra staben har arbejdet på UCN i mere end 30år. Det sociale sammenhold er godt, og flere af medarbejderne ses også privat. Derudover bliver der afholdt mange sociale arrangementer, hvad enten det er ølsmagning eller andre aktiviteter. Tidligere og pensionerede medarbejdere er også deltagende i dette. [7]

Teknologiske faktorer

UCN uddanner mennesker til virkeligheden. Derfor er det essentielt, at der undervises i nuværende og kommende teknologier, som de studerende kommer til at møde, når de er færdig uddannede. Dette medfører, at UCN er nødsaget til at være opdaterede på udstyr samt, at undervisernes viden holder trit med tiden. UCN bruger forholdsvis mange ressourcer på forskningsprojekter samt undervisernes muligheder for at lave en ph.d.-afhandling.

SWOT/TOWS-analyse

STYRKER	SVAGHEDER
<ul style="list-style-type: none"> – Kompetent og erfaren stab – Godt miljø og sammenhold – Gode faglige faciliteter til både lærere og elever 	<ul style="list-style-type: none"> - Frafald -
MULIGHEDER	TRUSLER
<ul style="list-style-type: none"> - Mulighed for at formindske fravær ved tidlig indgriben - Mindske økonomisk tab - Mindre manuelt arbejde - Lavere risiko for, elever bliver "glemt" under registrering af fravær (elevens eget ansvar) 	<ul style="list-style-type: none"> - Ny teknologi - Andre relevante uddannelser - Høj frafaldsprocent

SMART – Strategiske mål

Problemstillingen hos UCN er fravær. Og det formodes at der er en sammenhæng mellem fravær og frafald. Overordnet betyder dette, at UCN modtager færre penge fra staten/kommunen grundet færre studerende. Når UCN modtager færre penge, kan det påvirke undervisningskvaliteten og resultere i afskedigelse af medarbejdere. Dette kan forgrene sig til top-up uddannelserne, hvis der ikke er medarbejdere til at varetage undervisningen, kan studieretningen risikere at blive lukket ned. Dette vil medføre endnu færre studerende. Derfor er det essentielt for UCN at bryde denne negative vækst.

Bogstaverne i SMART står for:

- Specifikt mål
- Målbart
- Accepteret
- Realistisk
- Tidsafgrænset

Det specifikke mål for UCN er at nedbringe de studerendes fravær. For dermed at blive ved med at vækste, bibeholde medarbejdere, og deres respektive kompetencer. Og samtidig være en økonomisk sund forretning.

Det specifikke mål er i denne sammenhæng målbart. UCN laver hvert år en årsrapport, hvori det fremgår, hvor stort optaget af studerende har været, og hvor mange studerende, der går på UCN.

Er målet accepteret. Det må antages, at målet er accepteret af alle involverede parter. Da målet er ensbetydende med jobsikkerhed for alle medarbejderne.

Er målet realistisk? Det er en generel tendens i undervisningsmiljøet, at mange studerende har meget fravær, og der er et stort frafald på mange uddannelser. Med en tidlig indsats målrettet studerende med et unormalt fravær, er det muligt gennem samtale at rette op på fraværet, og dermed undgå et frafald.

Målet er tidsafgrænset. UCN forventer, at den negative tendens forsætter de kommende år, før de forventer en forbedring. [3] Det er dermed et langsigtet mål at rette op på frafaldet på UCN.

Boston analyse

Fraværssystemet som er under udvikling, ligger umiddelbart i spørgsmålstegnet. Virksomheden bag fravær systemet er nye på markedet, så der er ikke mange, der kender til deres system og/eller dem og deres virksomhed, men dog kan de være heldige med et par gode investeringer i især marketing, så har de potentiale til at få produktet rykket hen i stjerne kategorien. Grunden til de har en chance for at komme i stjerne kategorien er, at deres system ikke udelukkende kan bruges til fravær på skole. Systemet kan forholdsvist nemt laves om, så det kan bruges til andre virksomheder, hvor elevers fravær måske ikke er så relevant. Det kunne f.eks. være et system, der logger medarbejders arbejdstid i en virksomhed. Desuden, så skal det være et produkt, der er nogen, der går og mangler. Det er ikke noget ved at prøve at sælge noget, der ikke er interesse for. Desuden, så er der andre virksomheder, der laver lignende løsninger, så produktet skal kunne skille sig ud på den ene eller anden måde fra det de laver. Virksomheden kan også risikere, at deres produkt kommer i hundekategorien. Som sagt før, så er der andre virksomheder, der laver lignende løsninger, så der skal være en vis konkurrenceevne om så det er pris, funktioner osv. Derudover, så har langt de fleste virksomheder/personer også en vis grad for loyalitet hos andre, de har handlet med før, så der kan være en generel tendens til, at andre der leder efter et lignende produkt, vælger en udbyder, de måske allerede har handlet med før eller har hørt godt om fra andre. Det kan jo gå begge veje, men eftersom virksomheden ny på arbejdsmarkedet, så er der langt større sandsynlighed for, at de bliver valgt fra i modsætning til større og anerkendte virksomheder. Derudover, så kan

marketing også gå galt, hvilket også ville medføre, at virksomheden højst sandsynligt bliver valgt fra. Umiddelbart vil det aldrig blive en malkeko, da det er et system, der skal kunne holde længe uden, der eventuelt skal udskiftes dele eller hele systemer. Systemet skal sælges som engangsbetaling, så der kommer ikke til at være et konstant penge flow fra "gamle" kunder, f.eks. i form af et abonnement. Derfor menes der, at malkeko er et uopnåeligt mål for dette produkt.

Adizes lederroller

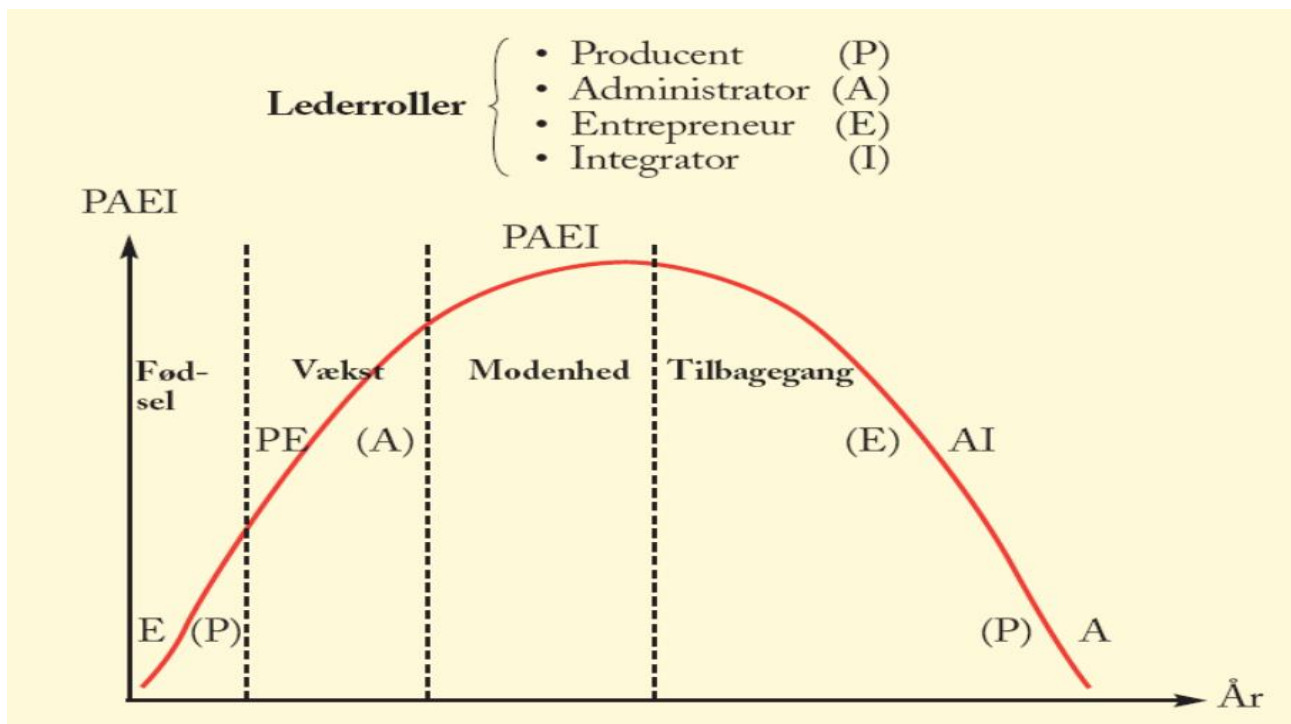
Adizes har beskrevet 4 lederroller, som hver især har nogle kompetencer, der er nødvendige for at løse daglige ledelsesopgaver i en virksomhed. Den perfekte leder besidder alle egenskaberne/kompetencerne – men nogle af dem er modstridende, og det er derfor ikke en mulighed, at en leder besidder dem alle. Det er derfor essentielt, at der er en ledelsesstab som komplimenterer hinanden, hvor de som ledelsesgruppe, dækker alle rollerne. For at en leder skal kunne samarbejde med sit ledelsesteam, er det nødvendigt at lederen besidder mindst et småt bogstav i alle rollerne, da det ellers ikke er muligt at få et ordentligt samarbejde med ledelse staben.

Figur 8.6 Adizes' fire lederroller		
	Fokuserer på	
	Produkt	Proces
Tidsmæssigt fokus Langt sigt	Entrepreneurrollen <ul style="list-style-type: none"> – Kreativ og innovativ – Finder nye produkter og nye metoder – Tænker strategisk – Stiller spørgsmål til det bestående – Risikovillig – Udvikling 	Integratorrollen <ul style="list-style-type: none"> – Integrerer i et fællesskab – Indgår kompromiser – Skaber motivation og korpsånd – Leder gennem teamwork – Skaber udvikling hos medarbejderne – Skaber sammenhold
Tidsmæssigt fokus Kort sigt	Producentrollen <ul style="list-style-type: none"> – Resultat- og handlingsorienteret – Stort præstationsbehov – Tager beslutninger – Flittig og travl – Medarbejderne bliver hjælpere – Faglig viden 	Administratorrollen <ul style="list-style-type: none"> – Opstiller mål og regler – Kontrollerer og evaluerer – Skaber systematik – Analyserer sig frem til den rigtige løsning – Bureaukrati – Ordenssans

Kilde: Adapteret fra Adizes, Ichak: How to Solve the Mismanagement Crisis, Dow Jones/Irwin, 1979.

Figur 3 - Adizes 4 lederroller [8]

Jørgen Sjøgaard Jakobsen er uddannelsesleder på It-uddannelserne. Jørgen beskrives som værende resultat orienteret, iderig og giver medarbejderne meget frihed under ansvar. Ud fra denne korte beskrivelse af Jørgen, kan det argumenteres for at Jørgen får Adizes kode PaEi, da egenskaberne der beskrives findes under entreprenørrollen og ligeledes producentrollen. Det må antages, at Jørgen har et lille bogstav i de andre kategorier, da der muligvis havde været en mere kompetent kandidat til stillingen.



Figur 3.3 - PAEI model [9]

Hvis det antages at analysen på Jørgen Søgaard Jakobsen er korrekt, så er det muligt, at Jørgen er blevet ansat i UCN til at vækste. Elev optaget er de seneste år faldet på UCN. UCN forventer faldende elev optag de kommende år.

Kulturanalyse

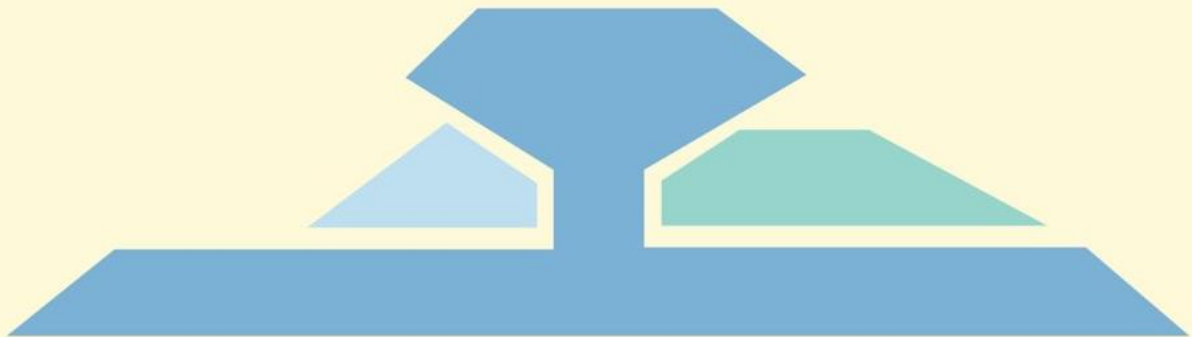
Mintzbergs organisationsformer

Ifølge Mintzberg findes der 5 forskellige organisationsformer.

- Den simple struktur
- Fagbureaukratiet
- Adhocratiet
- Maskinbureaukratiet
- Den divisionaliserede form

Oftest er det Fagbureaukratiet, som bliver benyttet af uddannelsessteder, hospitaler etc. Karakteristika for Fagbureaukratiet er, at det er produktionskernen, der er dominerende. Som det ses i bunden af figuren nedenfor.

Fagbureaukratiet



Figur 4 4 - Fagbureaukratiet, [10]

Arbejdsopgaverne er komplekse, og det forudsætter at medarbejderne er højtuddannede/specialiserede. [11]Graden af autonomi er høj. Medarbejderne har mulighed for selv at strukturere og tilrettelægge undervisningen, når de opfylder de centrale kundskabsfærdigheder, som det forventes at eleverne modtager.

Trompenaars og Hampden-Turners kulturtyper

Som tidligere beskrevet, er Jørgen Søgaard Jakobsen målrettet, og resultat orienteret. Dette passer ind i nedenstående figur. Der er lighed mellem ledelse og medarbejdere. Fokus er rettet på organisationens mål, og alle arbejder hen imod dette. Ifølge Trompenaars foretages arbejde i disse organisationer i teams eller projektgrupper. Dette formodes også at være tilfældet på UCN. Medarbejderne betragter sig selv som eksperter og professionelle indenfor deres eget felt. Og for at løse helhedsorienterede projekter, er det nødvendigt med involvering af yderligere eksperter for at løse opgaven/projektet.

Medarbejderne fokuserer på deres egne respektive karrierer, og er loyale overfor det faglige og ikke organisationen. Motivationen kommer indefra, og bunder i faglig og personlig stolthed. [12]

Organisationsdesign og kulturtyper	
Organisationsform	Kulturtyper
Adhockrati	Kuvøse
Fagbureaukrati	Det styrede missil
Maskinbureaukrati	Eiffeltårnet
Basale form	Familien

Fig. 9.8 Organisationsdesign og kulturtyper.

Figur 5.5 - Organisations/kultur typer, Organisationen, 2022, kap 9

Payback projection

Item	Year 1	Year 2	Year 3
Hardware	10.000	7000	7000
Hardware vedligehold	Klares internt.		
Arbejds løn 64 timer	153.600 ¹	0	0
Fastholdelse af studerende ²	2.200.000	2.200.000	2.200.000
Besparelse i alt	~2.000.000	~2.000.000	~2.000.000

Hvis der tages udgangspunkt i 18 klasser med 30 elever i hver. Så er der $18 \times 30 = 540$ elever på UCN IT. Med et frafald på 30%, er der et frafald på 162 elever. Ifølge UCN-medarbejder, er forhåbningerne at det nye fraværs system, kan redde 4-5 elever pr klasse. Med et udgangspunkt på 18 klassen, er det $4,5 \times 18 = 81$ elever der gribes, før der sker frafald. $81 \times 40.000 = 3.240.000$ kr i alt. Tallet er ikke fyldestgørende. Da studerende, som dropper ud på første semester, også påvirker kommende indtjening på følgende semestre.

Da UCN har en frafaldsrate på 30%, må dette være spredt ud over alle semestre. Vi antager at det største frafald på uddannelserne, sker på første semester.

For ikke at overkomplicere udregningen, antages det at 2 ud af 3 af dem, der dropper ud, dropper ud på første semester. Udregningen bliver derfor $3.240.000 \times 0,66 = 2.200.000$ pr semester. Dertil kommer det, at den sidste

¹ Tallet er et skøn. 3 personer til 800/timen. Sammenlagt 64 brugt i alt.

² Ved fastholdelse af 4,5 studerende i gennemsnit.

3. del dropper ud efterfølgende. Og nye studerende starter det kommende semester. Dette medfører at tallet bliver større.

- Lcd skærm – ca 65 kroner
- PCF8574 – ca 16 kroner
- MFRC522 – ca 30 kroner
- Ca 5-8 kroner pr tag/card
- Mikrokontroller ca 50-75 kroner

	Pris i kr	Pris pr. klasse (30 elever)
LCD MODULE 162B SERIES	65	1
PCF8574 - portexpander	9	1
MFRC522 - RFID reader/writer	30	1
RFID Kort	8	30
ESP8266, IOT MAINBOARD	60	1
Totalpris pr. klasse (30 elever)	404	

Her er der lavet et regneark over, hvad det kommer til at koste pr. enhed samt kort til hver elev i klassen. Prisen kan selvfølgelig variere lidt, hvis der er færre/flere elever i klassen, der er bare blevet taget udgangspunkt i en klasse på 30 elever.

Delkonklusion

Da frafald på uddannelser er en generel tendens i samfundet, så kan det det blive svært at komme problemet til livs. Og det er usikkert, om et nyt fraværs system kan ændre dette. Problemet skal muligvis tages op fra politisk side. Og en mulighed kunne være at indføre mødepligt, som det netop er på gymnasier. Derudover kunne højt fravær medføre en økonomisk konsekvens for studerende, hvor størstedelen af deres indtægt er SU. Dette kan dog også være en motivationsfaktor for elever, at de bliver ved med at møde op. Der er heller ikke garanti for, at fraværssystemet kommer til at påvirke frafald meget, da skoleelever ofte fortryder valg af uddannelse, og disse elever vil med høj sandsynlighed stadig vælge at droppe ud.

Teknisk analyse

Udviklingsplatform

Der gøres brug af en Raspberry pi 4B som udviklingsplatform

Opsætning af pi

Først og fremmest, så skal SPI og I2C slås til. Dette gøres ved at skrive:

```
sudo raspi-config
```

Tryk derefter på "Interfacing Options". Derinde vil der være en SPI og I2C option, begge af disse skal være slået til. Herefter man genstarter PI'en.

For at få adressen til LCD skærmen skal man skrive:

```
sudo apt-get install i2c-tools
```

Derefter skriver man:

```
i2cdetect -y 1
```

Det vil give et output, der fortæller, hvilke I2C adresser, der er i brug.

Boot

I tilfælde af strømsvigt og evt. andre situationer, hvor udviklingsplatformen skal starte op igen. Er det nødvendigt at pythonfilen starter automatisk, i stedet for at medarbejderne på skolen skal starte filen op manuelt. Da dette er ressourcekrævende. Dette opnås igennem systemctl på Raspberry Pi.

For at lave en service på Raspberry Pi, benyttes følgende fremgang:

sudo nano /lib/systemd/system/tid.service, hvilket åbner en blank fil. Dernæst indsættes følgende.

```
[Unit]
Description=tjek
After=network.target

[Service]
ExecStart=/usr/bin/python3 /home/kim29/code/sp/tjekin.py
User=kim29
Group=kim29
Restart=always
StandardOutput=append:/home/kim29/code/sp/fejl.log
[Install]
WantedBy=multi-user.target
```

Ved genstart af systemet, venter tid.service på at netværket er startet op. Dette gøres for at sikre, at der er etableret internetforbindelse på Raspberry pi, da filen kræver internetadgang. Hvis der ikke er internetforbindelse tilgængelig, vil filen ikke kunne køre korrekt og resultere i en fejl. Herefter specificeres kommandoen til at bruge Python3 til at køre filen og filens sti indsættes som argument. Servicen er sat til at genstarte tjekin.py, hvis der opstår en fejl eller lign. For at kunne fejlfinde, hvis servicen ikke fungerer korrekt, kan en logfil give information om eventuelle fejl eller problemer, der opstår under kørslen. Således, at disse problemer kan identificeres og rettes.

sudo systemctl daemon-reload – Alle systemctl services reloades, så den nye service er inkluderet.

sudo systemctl enable tid.service – Aktiverer den nye service.

Sudo systemctl start tid.service – Den nye service startes op.

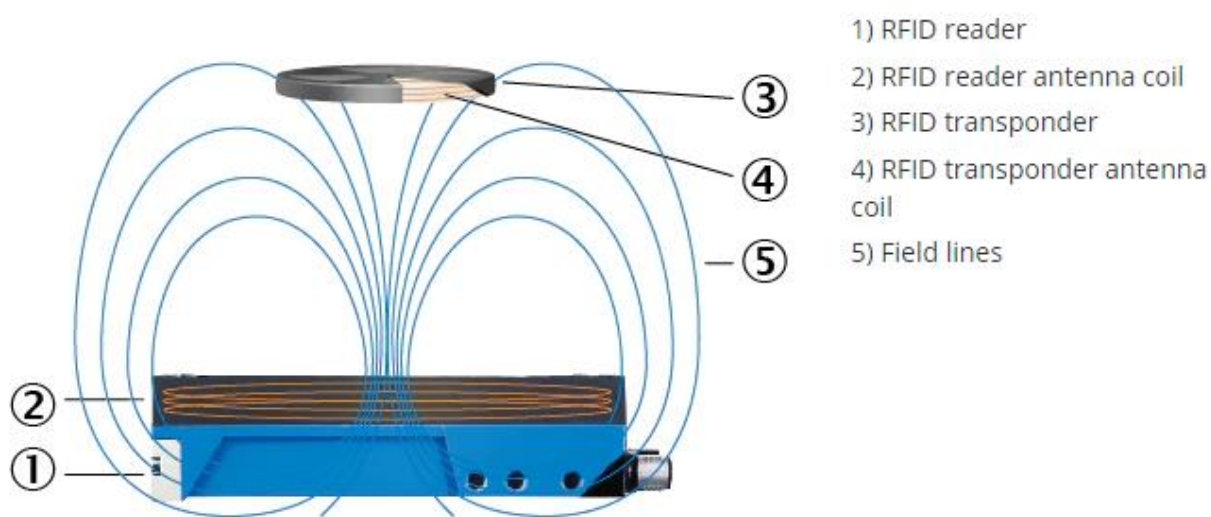
RFID

Radiofrekvens identifikation, også kaldet RFID er en form for trådløs kommunikation, der forekommer over radiobølger. Et RFID-system, består af 3 dele, en antenne, transceiver og en transponder. Ift. dette projekt, er antennen og transceiveren sat sammen for at forme en RFID læser. Transponderen i denne forstand er derved, det pågældende RFID tag. Denne læser sender radiobølger ud for at aktivere tagget, tagget sender så radiobølger tilbage hvor det bliver lavet om til data. [13]. RFID har en bred funktionalitet indenfor flere områder såsom automatisering, i form af access control, identifikation og sporing, både af mennesker, dyr og varer. Her kan man spare meget manuel dataindtastning og arbejdskraft. RFID bliver også brugt indenfor automobilbranchen til identifikation af dele. Dog selvom RFID er yderst smart og tilgængelig teknologi, bringer det også mange sikkerhedsrisici med sig. Alle mennesker kan lytte med på disse radiobølger, så længe det ikke er tilstrækkeligt krypteret, ydermere kan man nemt kloner et RFID-tag ved brug af enten en reader eller andet teknologi, koblet sammen med det rigtige software. [14] Der er også knyttet en del privatlivs risici ift. Hvilket data der ligger på kortet.

Der er passive og aktive RFID-tags. Aktive har deres egen strømforsyning, og kan derfor sende på længere afstand, hvorimod det passive tager strøm via elektromagnetiske bølger fra antennen. Ydermere er der semi-passive tags, de har en strømforsyning der kører kredsløbet, mens kommunikation køres af læser. [13]

Der er 4 typer af frekvens-intervaller for RFID

1. Lavfrekvens. Dette interval ligger fra 30KHz til og med 500KHz
2. Højfrekvens. Dette interval ligger fra 3MHz til og med 30MHz. Det er dette interval der bruges ift projektet
3. Ultra-høj-frekvens. Dette interval ligger fra 300MHz til og med 960Mhz
4. Mikrobølge RFID-systemer. Dette ligger på 2.45 GHz og kan læses næsten op til 10 meter væk



Figur 6 6 - RFID Reader [15]

RFID i hverdagen

Langt de fleste mennesker har stiftet bekendtskab med RFID. Teknologien bliver i vid udstrækning brugt på hoteller hele verdenen over. Det er nemt at lave adgangskort til hvem end, der skal bruge et, så derfor er det heller ikke galt, hvis medarbejdere eller kunder mister deres, og skal bruge et nyt kort.

RFID bliver i højere grad brugt i forskellige virksomheder verden over. Det kunne f.eks. være i Michelin dæk. Virksomheden anslår at i 2024 er teknologien at finde i alle dæk de producerer. Teknologien bliver allerede brugt i lastbils dæk, og motorsport. [16]

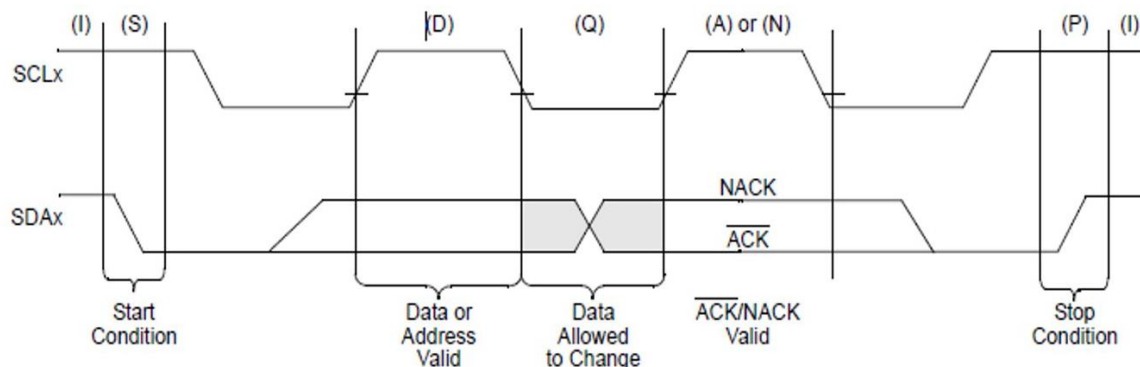
Derudover er teknologien ved at vinde indpas på lagre. RFID gør det nemt at få et overblik over lagerstyring. Samtidig er det muligt at spare mandetimer på at kunne scanne varer via RFID, i stedet for en standard strejkode, hvis varerne er placeret i kasser, da dette kan være besværligt, da kasserne skal placeres korrekt, for at få en aflæsning.

Hardware

Systemet består af en Raspberry pi 4B med et 16x02 Alpha numerisk display modul med en indbygget PCF8574 I/O port expander, hvor der bruges I2C. Denne port expander, omdanner den serielle I2C data om til parallelle data, der bruges til display. [17] Det betyder også at der kun er 4 forbindelser i stedet for 16. Disse forbindelser er således GND, VCC(5v), SDA (Serial Data) og SCL (Serial Clock). Til læse/skrive af RFID, bruges et MIFARE RC522 RFID-modul til kontaktløs kommunikation ved 13.56 MHz. Til denne bruges SPI-protokollen.

I2C protokollen – Inter Integrated Circuit

I2C er en synkron seriel multi-master/slave kommunikations protokol, der gør brug af disse 2 SDA og SCL-forbindelser. I2C sender data bit efter bit (serielt), og bliver synkroniseret efter det master styrede SCL signal. Når en master vil kommunikere med en slave, sender den adressen af den slave, der skal kommunikeres med ud til alle slaver på bussen, disse slaver sammenligner så adressen med deres egen for at finde ud af, hvilken der skal oprettes forbindelse til. Bit 0 specificerer om master vil Read/write. [18]

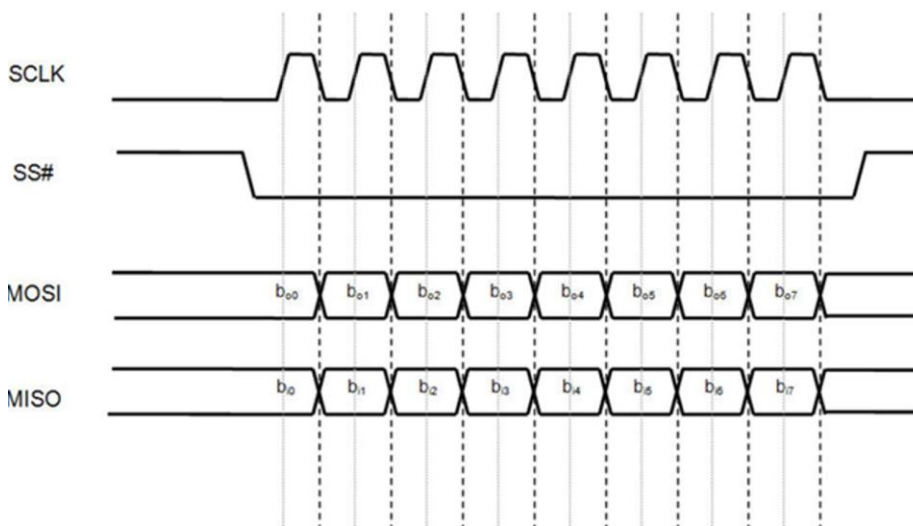


Figur 77 - i2c Bus protocol states, Steffen rahbek vutborg, 2022, UCN [18]

- Bus-idle(I) både data og clock er HIGH efter stop condition og før start
- Start data transfer(S) Når bussen er ledig, vil SDA linjen gå LOW mens SCL er HIGH. Dette er start condition for data transfer.
- Data-valid(D) Data er valid når SDA signalet har været i stabil tilstand i clock periodens HIGH tilstand
- Wait/Invalid(Q) Når clock signalet rammer lav, skal data skiftes.
- ACK/NACK(N) Modtageren af data vil trække SDA linjen lav eller høj tilstand for at give ACK/NACK
- Stop-data(P) vil gøre SDA linjen HØJ mens SCL er HØJ.

SPI-protokollen – Serial Peripheral Interface

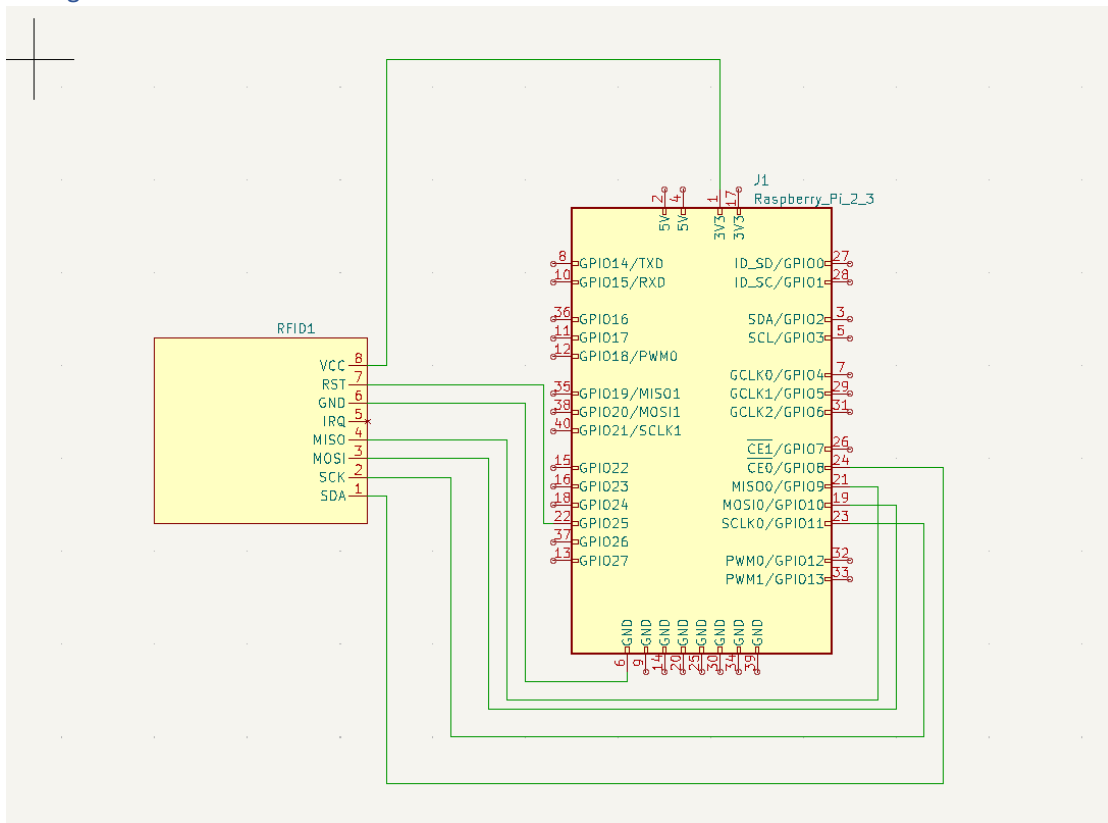
I modsætning til I2C er SPI en synkron single-masterkommunikations protokol. Den gør brug af 4 forbindelser som er således, SCLK (Signal-clock) som laves af master enheden og sendes ud til alle slaver, SS/CS(Slave-select) styres af master til at vælge hvilken slave enhed der skal kommunikeres med, MOSI (Master out-Slave in) sender data ud fra master enheden til slave enheden og MISO (Master in-Slave Out) anvendes af master enheden til at modtage data fra slave enheden, dette signal styres af slave enheden [18]



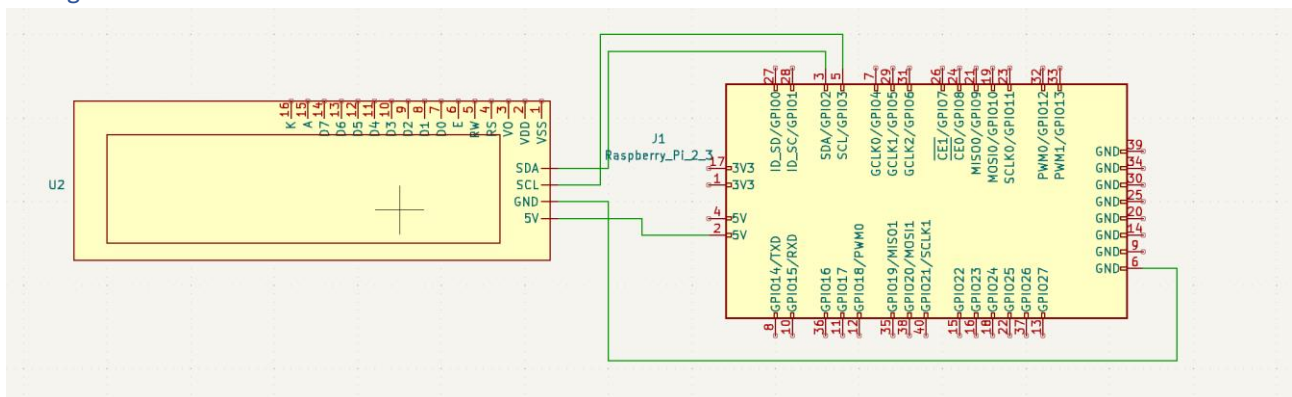
Figur 8 – eksempel på SPI kommunikation, Steffen rahbek vutborg, UCN 2022 [19]

Når en master enhed vil sende/modtage data, starter den med at trække SS linjen lav for enheden, den vil kommunikere med. Den aktiverer også clock signalet her for både master og slave. Efter dette, generere master enheden data på MOSI mens den læser data fra MISO [18]

El diagram – MFRC 522 - PI



El diagram – LCD – PI



Her er en Raspberry Pi tilsluttet til LCD skærmen. Man kan se der kun gøres brug af 4 forbindelser grundet PCF8574 port expander

Software

MariaDB

MariaDB er en relationel database, der organiserer data i tabeller og etablerer relationer mellem dem. Dette gør det muligt at gemme og forbinde data på tværs af forskellige tabeller. For eksempel kan studerendes oplysninger være gemt i en tabel, mens deres fravær registreres i en anden tabel. Disse tabeller kan forbindes via SQL kommandoer.

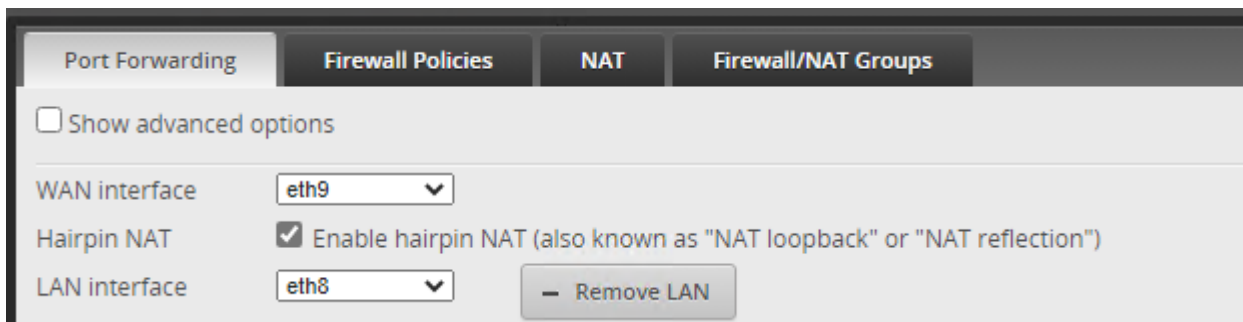
MariaDB tilbyder forskellige versioner, herunder en Community Edition, der er gratis og open source. Denne udgave kan bruges frit, og giver brugerne mulighed for at tilpasse og modificere kildekoden. Der er også en kommerciel version, MariaDB Enterprise, som tilbyder ekstra funktioner og support til virksomheder. Priserne for MariaDB Enterprise varierer afhængigt af virksomhedens behov og krav.

Opsætning af MariaDB

Denne opsætning er blevet brugt under hele projektforsløbet, for at kunne rette og ændre i tabeller, kolonner osv. via internettet. Omkring projektets afslutning, blev dette ændret, da der var et behov for mere sikkerhed. Forbindelsen blev i stedet oprettet gennem SSH. Gennem projektforsløbet har der været brugt brugere til MariaDB, som et oprettet specifikt til at kunne forbinde over internettet.

For at etablere en forbindelse til MariaDB databaseserveren er det nødvendigt at åbne en specifik netværksport. MariaDB anvender som standard port 3306 til kommunikation. Da MariaDB serveren er bag en EdgeRouter 12P, skal port 3306 konfigureres til at være åben på routeren. Dette kan gøres ved at oprette en portforwarding-regel, der tillader trafik gennem port 3306 til serveren.

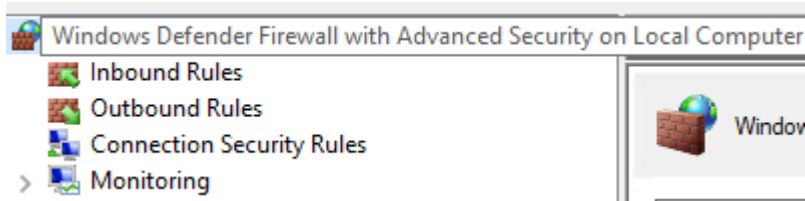
Ved at tilføje en portforwarding-regel på routeren kan ekstern trafik, som forbindelser fra internettet, nå frem til MariaDB databaseserveren. Dette gør det muligt for eksterne enheder at oprette forbindelse til og interagere med databasen via port 3306.



Porten bliver tilføjet, og porten er nu åben.

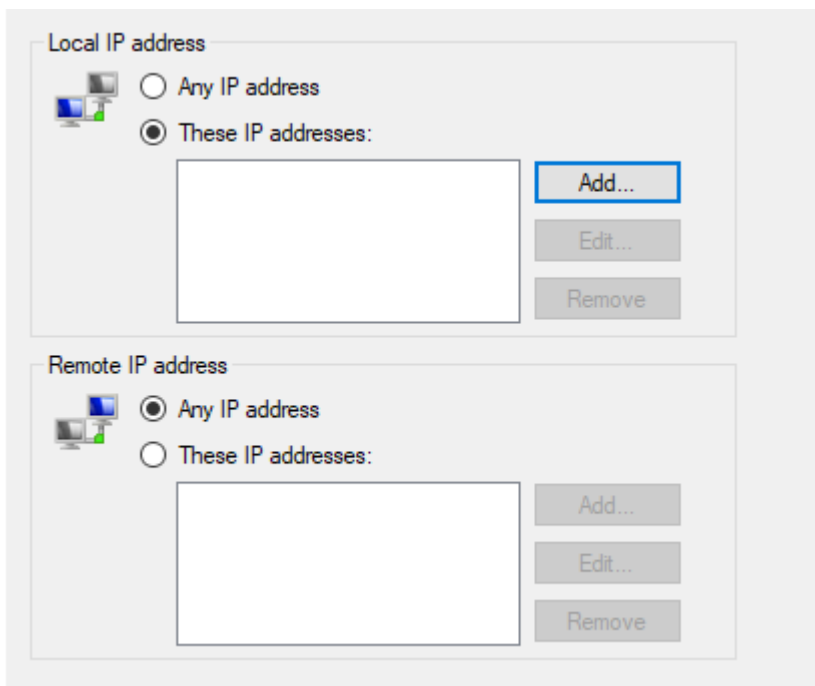


Derefter skal porten tilføjes i Windows server firewall, da det ikke er muligt at lave en forbindelse. Dette gøres ved at lave en indgående regel under Windows firewall indstillinger. Hvor port 3306 får lov til at blive brugt til at danne en forbindelse mellem server og Raspberry Pi.



I Windows firewall er det muligt at konfigurere hvilke ip-adresser, der skal have adgang til databasen. Udenfor billedet, er det muligt at lave et scope, så ip-adresser i en given range har adgang til serveren.

Dette er der ikke gjort brug af under projektet.



Bruger i MariaDB

For at lave en bruger i MariaDB, indtastes følgende i MySQL Client til MariaDB. Dette er et terminal vindue.

```
MariaDB [(none)]> CREATE USER 'benjamin'@'localhost' IDENTIFIED BY '@gruppe3';  
Query OK, 0 rows affected (0.038 sec)
```

Forespørgslen fortæller, at brugeren er oprettet.

Ved at bruge kommandoen: `SELECT user FROM mysql.user;` kommer en tabel frem med brugere i MariaDB

```
MariaDB [(none)]> SELECT User FROM mysql.user;
```

```
MariaDB [test2]> select user from mysql.user;
```

User
root
testperson
root
root
root
benjamin
kim
mariadb.sys
root

```
9 rows in set (0.046 sec)
```

Derefter får Benjamin alle privilegier til databasen test2.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON test2.* TO 'benjamin'@'localhost';
Query OK, 0 rows affected (0.054 sec)
```

Benjamin skal have mulighed for at tilføje og slette i de forskellige tabeller. Derfor er det nødvendigt, at testpersonen har rettigheder til dette. Benjamin er begrænset til databasen test2, og ikke eventuelle andre databaser, som ligger på samme server. For at opretholde sikkerheden er MariaDB som standard konfigureret til kun at tillade lokale forbindelser. Dette betyder, at adgang til databaser, tabeller osv. Kun er tilladt fra samme lokale netværk, hvor MariaDB-serveren kører. Det betyder, at eksterne klienter eller fjernforbindelser ikke har adgang til databasen, hvilket minimerer risikoen for uautoriseret adgang eller angreb udefra. Denne konfiguration sikrer, at kun lokale applikationer eller processer, der kører lokalt, kan oprette forbindelse til og ændre i databasen. [20] Benjamin bruges i det administrative program, som kan hente fravær, rette i tabeller osv.

I filen tjekin.py anvendes en anden bruger, da denne bruger, ikke har behov for de samme rettigheder. Brugeren skal have læsetilgang til tabellerne, men kun have mulighed for at indsætte data i fremmødetabellerne. Dette opnås på følgende måde:


- GRANT SELECT, INSERT, UPDATE ON test2.itt TO 'kim'@'localhost';
- GRANT SELECT, INSERT, UPDATE ON test2.dat TO 'kim'@'localhost';
- GRANT SELECT ON test2.studerende TO 'kim'@'localhost';
- GRANT SELECT ON test2.skemaitt TO 'kim'@'localhost';
- GRANT SELECT ON test2.skemadat TO 'kim'@'localhost';
- FLUSH PRIVILEGES;

Brugeren "kim" bliver tildelt læse-skriverettigheder til tabellen itt og dat i databasen "test2" i de to første linjer. Dette giver brugeren mulighed for at redigere fremmødetabellerne itt og dat, hvilket er nødvendigt for opdatering, når den studerende tjekker ind med RFID-kort/tags. Programmet kontrollerer klassen i studerendetabellen, så det er vigtigt, at programmet har læserettigheder til denne tabel. Da programmet kun skal læse og ikke skrive, får brugeren kun læserettigheder. På samme måde gælder det for de følgende to linjer, hvor brugeren kim får læserettigheder til tabellerne skemaitt og skemadat. For at opdatere brugerrettighederne anvendes "Flush PRIVILEGES" kommandoen.

Backup

Af sikkerhedsmæssige hensyn, bliver der lavet backup af databasen ugentligt. Måden dette er opnået på er følgende:

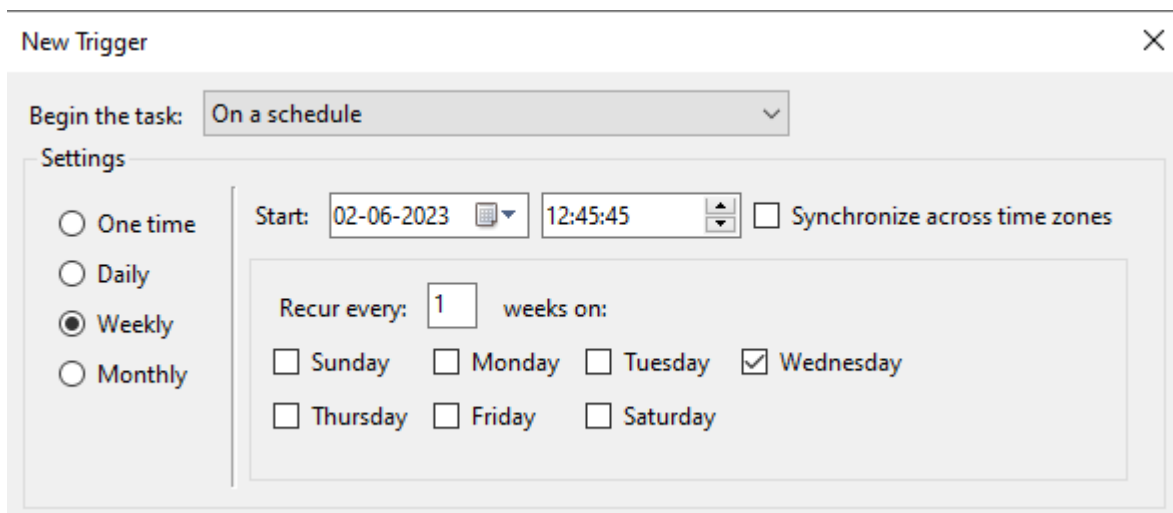
På serveren er der lavet en batchfil. Mysqldump er en hjælpe funktion i MariaDB. Dette bliver brugt til at oprette sikkerhedskopier af databaser. Derefter bliver bruger og password defineret. Endelig bliver filen gemt som backup.sql på c drevet.

 *test2backup.bat - Notepad

File Edit Format View Help

```
mysqldump -u root -pgruppe3 test2 > C:\backup\backup.sql
```

Batchfilen oprettes i task Schedule. Og derefter sættes den til at køre onsdag ugentligt.



New Trigger

Begin the task: On a schedule

Settings

☐ One time

☐ Daily

☒ Weekly

☐ Monthly

Start: 02-06-2023 12:45:45 ☐ Synchronize across time zones

Recur every: 1 weeks on:

☐ Sunday ☐ Monday ☐ Tuesday ☒ Wednesday

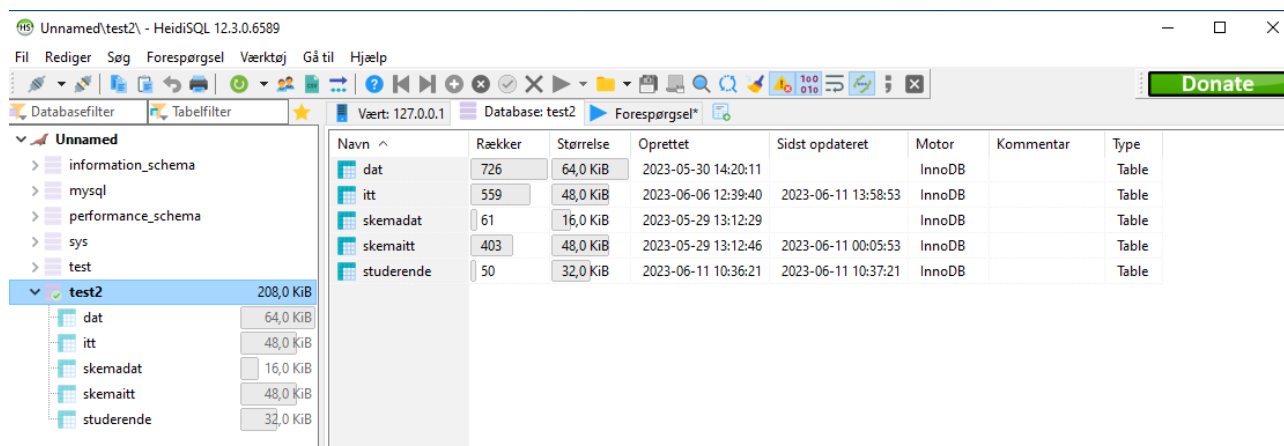
☐ Thursday ☐ Friday ☐ Saturday

HeidiSQL

HeidiSQL er en grafisk brugergrænseflade (GUI) til administration af MySQL-databaser. Hvilket gør det simpelt at oprette, ændre og administrere databaser, tabeller osv. HeidiSQL er open source, hvilket betyder kilden er frit tilgængelig og gratis at bruge.

Opsætning af databasen i HeidiSQL

Til at opsætte databasen er HeidiSQL blevet brugt. Nedenstående screenshot viser opsætningen af databasen. På billedet vises test2, som er selve databasen.



Opsætning af tabeller

Nedenunder er der listet 5 tabeller.

- studerende
- itt
- dat
- skemaitt
- skemadat

Studerende

#	Navn	Datatype	Length/Set	Usigne...	Tillad N...	Zerofill	Standard
1	start	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	curdate()
2	sid	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
3	navn	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
4	klasse	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL

Start er dato, hvor den studerende er startet på skolen. Sid er forkortelsen af studie id. Datatypen er INT, hvilket er en heltalstype. Og det er en unik key. Derfor kan den ikke forekomme i mere end en række. Dette er nødvendigt, da studerende kun kan forekomme en gang i denne tabel, og ydermere skal der heller ikke ske fejl med, at to studerende har det samme studie ID.

Navn indgår i en kolonne, da det er besværligt at huske studie id på alle de studerende. Navn gør det nemmere at skelne studerende fra hinanden. Når der søges efter tilstedeværelse/fravær.

Klasse beskriver hvilken klasse den studerende går i. Denne kolonne bruges RFID.py til at se, hvilken fraværs tabel, skal bruges når der registreres fremmøde.

Dette er et udsnit af studerende tabellen. Der er indsat 2 klasser. ITT og DAT. Med 25 studerende i hver klasse.

2023-02-02	21	Ella Nielsen	itt
2023-02-02	22	Felix Hansen	itt
2023-02-02	23	Luna Pedersen	itt
2023-02-02	24	Liam Christensen	itt
2023-02-02	25	Maja Møller	itt
2023-02-02	26	Lærke Olsen	DAT
2023-02-02	27	Malthe Jensen	DAT
2023-02-02	28	Nora Nielsen	DAT
2023-02-02	29	Magnus Pedersen	DAT
2023-02-02	30	Emilie Christensen	DAT
2023-02-02	31	Oscar Møller	DAT

Skemaïtt

Kolonner: + Tilføj × Fjern ▲ Op ▼ Ned							
#	Navn	Datatype	Length/Set	Usigne...	Tillad NULL?	Zerofill	Standard
1	dato	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
2	fag	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL

Skema er simpelt opbygget. Det er dato, hvilket er angivet til date i datatype, så den automatisk sætter dato på ved tilføjelse af række. Fag er bare text/string. Skemaïtt bruges til klassen ITT, og skemadat bruges til klassen DAT. Skemaet er hardcodet og indsat for at tjekke, om den studerende har undervisning den pågældende dag. Da der ikke skal registreres tilstedeværelse ved dage, hvor der ikke er planlagt undervisning.

Dette er et udsnit af skemaïtt:

test2.skema: 61 antal rækker i alt (tilnærmelsesvis)

dato	fag
2023-02-02	Programmering
2023-02-03	Netværksteknologi
2023-02-06	Netværksteknologi
2023-02-07	Indlejrede systemer
2023-02-08	Programmering
2023-02-13	Indlejrede systemer
2023-02-14	Netværksteknologi
2023-02-15	Indlejrede systemer
2023-02-16	Programmering
2023-02-20	Netværksteknologi

Itt og dat

Kolonner: + Tilføj × Fjern ▲ Op ▼ Ned							
#	Navn	Datatype	Length/Set	Usigne...	Tillad NULL?	Zerofill	Standard
1	dato	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
2	sid	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL
3	fm	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL

Itt og dat består af dato, sid og fm (fremmøde).

Dato er med i denne tabel, da det er fraværstabellen. Når studerende bliver registreret i systemet, kommer dato med i registreringen. For at det er muligt at udregne tilstedeværelse/fravær er det nødvendigt at se hvilke datoer, den studerende har været til stede.

SID viser hvilken studerende, som registrerer sig tilstede for undervisningen.

Fm står for fremmøde og er en heltalstype, der angiver om den studerende er til stede eller ej. Når den studerende tjekker ind i systemet, sættes værdien af fm til 1. Fm-kolonnen bruges til at registrere den studerendes fravær. Derfor vil der kun være værdien 1 i denne kolonne, og ingen andre værdier. Ved fravær, så bliver intet indsat i tabellen, dette er for at spare på størrelsen af databasen. Både itt og dat (Fremmøde tabeller) er identiske og bruges til samme formål. De bruges til at adskille fravær fra de to forskellige klasser, så databasen bliver delt op i lidt flere dele, hvilket ultimativt sørger for, at det bliver hurtigere at hente data, da den ikke skal kigge gennem tabeller med flere millioner entries.

Dette er et udsnit af itt:

test2.f1: 1.398 antal rækker i alt (tilnæ

dato	sid	fm
2023-02-02	3	1
2023-02-02	6	1
2023-02-02	9	1
2023-02-02	13	1
2023-02-02	14	1
2023-02-02	15	1
2023-02-02	18	1
2023-02-02	19	1
2023-02-02	21	1
2023-02-02	23	1

Ud fra dette udsnit, er det muligt at se studerende 3,6,9,13 osv., har været til stede d.2/2 til undervisningen.

Sikkerhed

Det anbefales at give hver aktør specifikke rettigheder til kun at kunne læse og skrive på de relevante databaser og tabeller i MariaDB. Dette øger sikkerheden ved at begrænse adgangen til kun det nødvendige.

Ved at tildele individuelle brugere kun det nødvendige i stedet for at bruge root-brugeren kan det forebygges mod utilsigtede ændringer eller uautoriseret adgang til andre tables eller følsomme data. Det er vigtigt at være opmærksom på, at root-brugeren har omfattende rettigheder, herunder evnen til at slette en database uden at få en advarsel. Og derfor bør brugen af root-brugeren begrænses til nødvendige administrative opgaver.

Generel sikkerhed

SQL-Injections

Da størstedelen af systemet er baseret på SQL, giver det mening at sikre sig mod SQL-injections. Derfor gøres der hyppigt brug af parametriserede-forespørgsler, med “%s” eller “?” placeholders. Dette betyder at der ikke indgår dynamisk data direkte i forespørgslen og derved er det sværere at manipulere, eftersom det er adskilt.

Parametriserede forespørgsler

Handler om at adskille user input fra forespørgslerne ved at lave dem om til parametre, da disse parametre ikke kan indeholde eksekverbar kode. [21]

```
def klasseFravær():  
    hvilkentabel = input10.get()  
    # Hent studerende fra den valgte klasse.  
    klasse = "SELECT sid, navn FROM studerende WHERE klasse = %s"  
    skemaNavn = "skema" + hvilkentabel  
    cur.execute(klasse,(hvilketabel,))
```

Her er et eksempel på parametriserede forespørgsler. Input bliver pakket ind i “hvilketabel” og herefter brugt som parameter i execute statement på klasse query

Sikkerhed i Administrator Interface.

I administrator interfacet, er det vigtigt, at en uønsket person ikke får adgang til programmet, da man kan slette tabeller derindefra. Det er ikke muligt at slette tabellen med alle de studerende, da denne tabel er det vigtigste for hele fraværssystemet, og det skal som udgangspunkt altid være til stede. Alle andre tabeller kan dog slettes gennem interfacet, så hvis der er en elev eller en anden uvedkommende, der får adgang til interfacet, så kan det være katastrofalt for hele fraværssystemet. Dog skal de have lidt viden om, hvad de forskellige tabeller hedder, hvis de skal kunne slette dem.

Forbindelse til MariaDB fra Raspberry Pi.

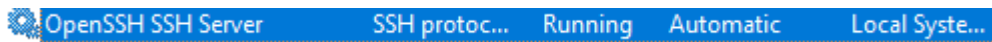
For at sende data sikkert over netværket er forskellige muligheder blevet undersøgt. Det første valg var OpenSSL, men dette kunne ikke fungere med opsætningen af databasen og Raspberry Pi. I stedet blev det via en SSH. For at lave en SSH-forbindelse til MariaDB, skal OpenSSH server installeres på Microsoft 2022 serveren.

```
# Install the OpenSSH Server  
Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
```

Kilde: [22]

Denne kan installeres i “add roles and features” i server management, eller via powershell, på Windows server 2022, som vist i ovenstående screenshot. Derefter er det muligt at etablere forbindelse til Windows server 2022. Nedenstående kode viser implementeringen af SSH med Python biblioteket SSHTunnel.

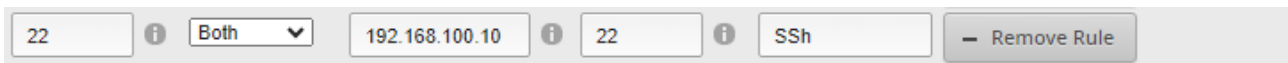
For at servicen starter ved boot, er det nødvendigt at sætte den til automatic inde i services.msc.



SSH-delen i tjekin.py

```
29  ∨ with SSHTunnelForwarder(  
30      (ssh_host, ssh_port),  
31      ssh_username=ssh_user,  
32      ssh_password=ssh_password,  
33      remote_bind_address=('127.0.0.1', db_port)  
34  
35  ∨ ) as tunnel:  
36  ∨     conn = mysql.connector.connect(  
37      host=db_host,  
38      port=tunnel.local_bind_port,  
39      user=db_user,  
40      password=db_password,  
41      database=db_database  
42  )
```

Porten åbnes på routeren, så det er muligt at forbinde via SSH til Windows serveren.



SSH står for secure shell, og det er en kryptografisk netværksprotokol. Som bruges til at sende data over usikre netværk, eller forbinde til fjern computere.

Validerings processen foregår på forskellige måder. Symmetrisk, også kaldet shared key kryptering, hvor klienten og serveren deler samme nøgle. Asymmetrisk, hvor der bruges en public og en private key. Og endelig hashing. Hashing laver en tilfældig string, baseret på en række elementer, pakkenummer, den symmetriske nøgle, og data der sendes. Derefter sendes signaturen til hosten. Hosten laver den samme signatur, for at kontrollere om de stemmer overens, for at validere om pakken er blevet ændret. [23]

Sikkerhedsmæssigt er det bedre at bruge keys, i stedet for at logge ind direkte gennem SSH, med brugernavn og adgangskode.

Fysisk sikkerhed

Versioner af interface

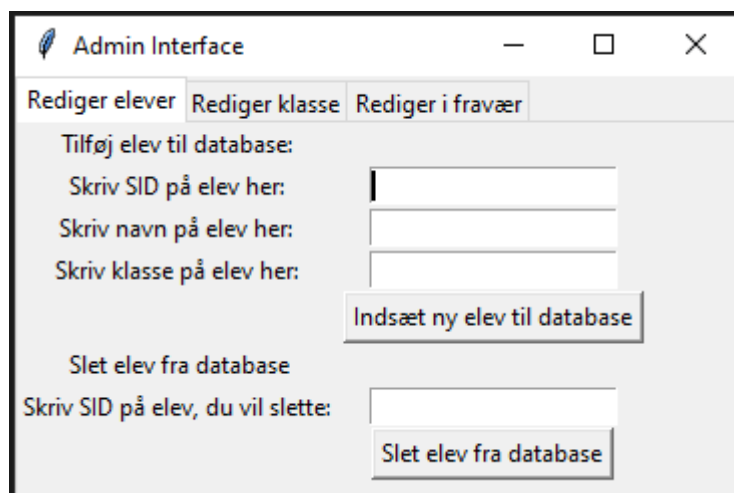
Version 1 - CLI Interface

Selve administrator interface begyndte med at være baseret i terminalen. Dog var alle enige om, at det ikke var helt lige så brugervenligt, da det kan virke ret kompliceret for folk, der ikke er vant til at bruge CLI. CLI-interfacet var bygget op, så man kunne skrive a, b eller c for at komme ind i en specifik menu, og derefter var det tal for at komme længere ind i menuen (kig på diagram i starten af Gennemgang af koden afsnittet længere nede). CLI-versionen er som sagt lidt sværere at bruge, hvis man ikke er vant til tekstbaseret computerbrug, men det blev forsøgt at gøre så brugervenligt som muligt, dog syntes alle det kunne gøres bedre i en ny version, der er grafisk i stedet for.

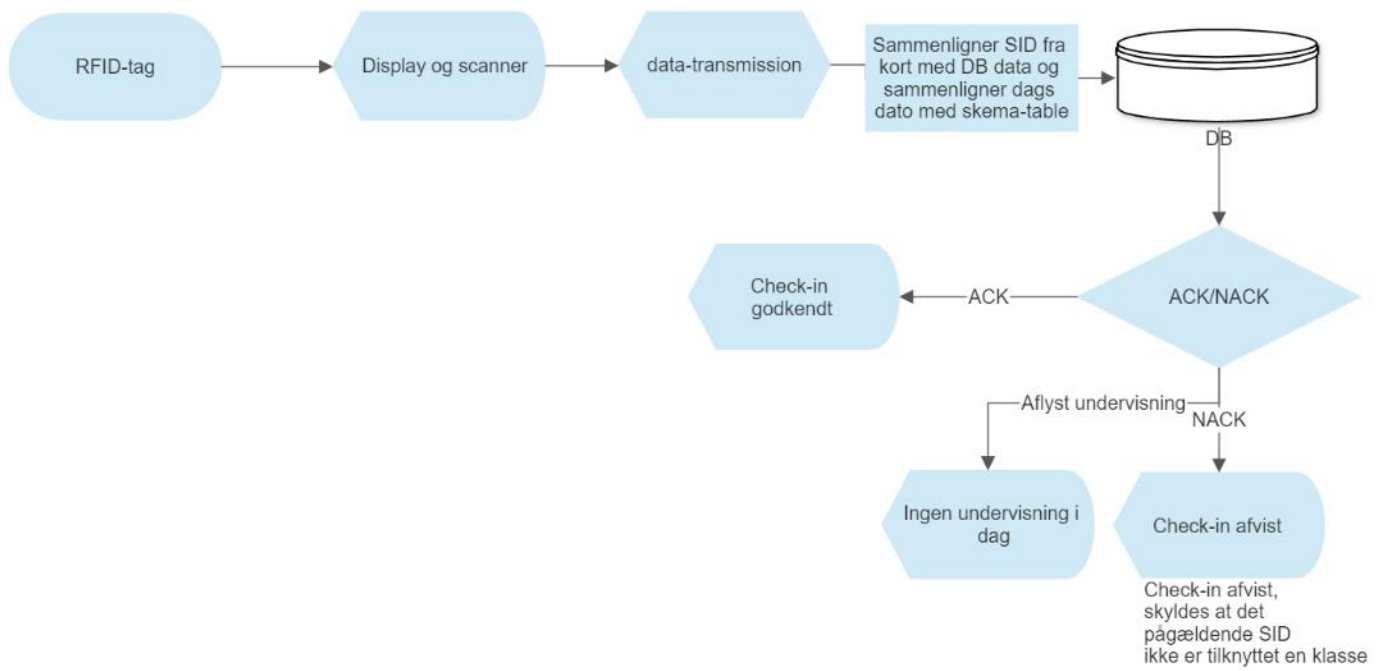
```
C:\Users\benja\Desktop>python3 CLIMENU.  
Tryk a for at redigere elever.  
Tryk b for at redigere i klasser.  
Tryk c for at redigere i fravær.
```

Version 2 – GUI Interface

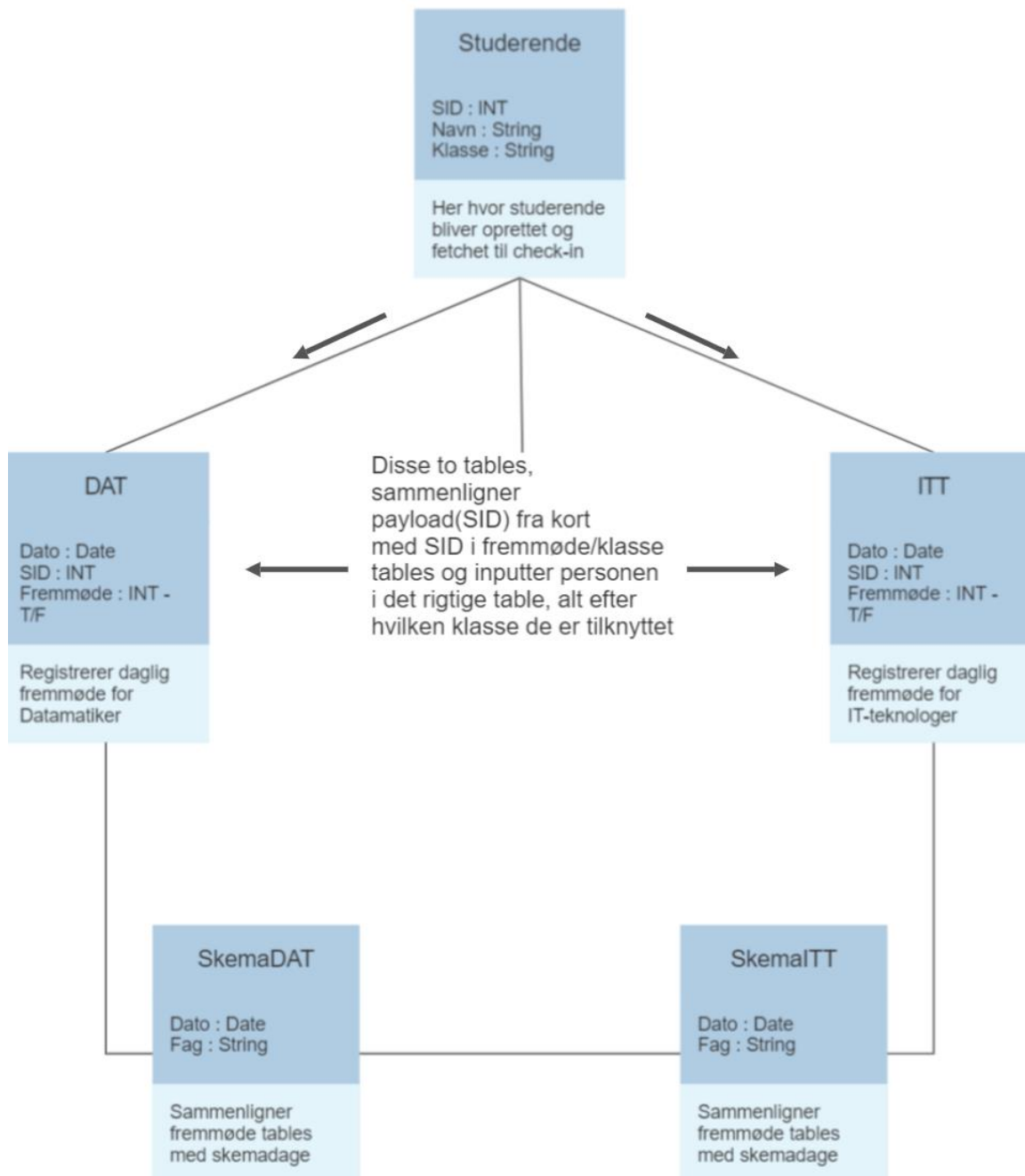
Som nævnt før, så var alle enige i, at CLI baseret interface ikke er det mest hensigtsmæssige. Derfor blev interface ændret. Det grafiske blev lavet ved hjælp af Tkinter biblioteket i Python. Det er et forholdsvis let bibliotek at lære, dog kan det hurtigt blive svært, da det desværre er ret nemt at miste overblikket over de næsten helt ens linjer af kode. Alligevel blev det lavet i Tkinter, da det er en simpel metode at lave grafisk design med Python. Derudover, kan det give et nogenlunde billede af, hvordan det færdige GUI eventuelt kunne se ud. Designet er ikke æstetisk, men funktionelt. En eventuel senere version, ville kunne gøres mere grafisk. Men dette skulle laves med et andet værktøj end Tkinter, da det ville nok være essentielt for at gøre det bedre og pænere. Nedenfor er et billede af startsiden, når man åbner programmet.



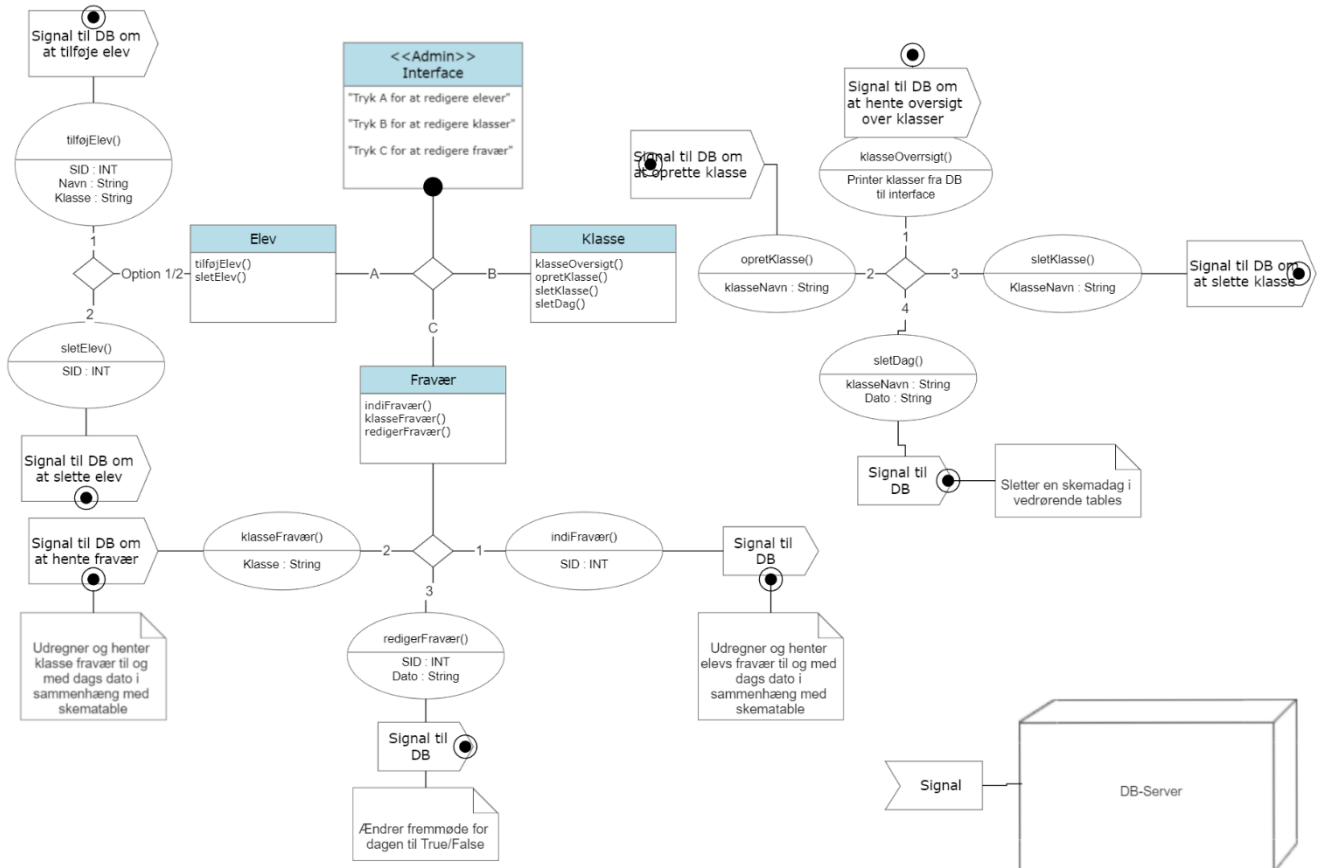
Flow fra RFID-tag til DB



DB-diagram



Gennemgang af koden



Tjekin.py

```
from time import sleep
from mfrc522 import SimpleMFRC522
from datetime import date
import mysql.connector
from sshtunnel import SSHTunnelForwarder
reader = SimpleMFRC522()
```

```
conn = mysql.connector.connect(  
    host="79.171.148.163",  
    user="root",  
    passwd="gruppe3",  
    database="test2"  
)
```

De forskellige biblioteker, som bliver brugt i koden, bliver importeret. Sleep bliver brugt i starten af koden for at undgå at Python itererer for hurtigt ned over koden. Dernæst importeres mfr522 læseren, der muliggør at der kan læses og skrives til et RFID kort/tag. Dato bliver brugt til at indsætte dato i tabeller længere nede i koden.

SSH og MariaDB opsætning.

```
# SSH tunnel konfiguration
ssh_host = '79.171.148.163'
ssh_port = 22
ssh_user = 'kim'
ssh_password = '@gruppe3'

# MariaDB konfiguration
db_host = '127.0.0.1'
db_port = 3306
db_user = 'kim'
db_password = 'gruppe3'
db_database = 'test2'
```

Ovenstående variabler indsættes, for at lave en SSH forbindelse til Microsoft server 2022.

```
with SSHTunnelForwarder(
    (ssh_host, ssh_port),
    ssh_username=ssh_user,
    ssh_password=ssh_password,
    remote_bind_address=('127.0.0.1', db_port)
) as tunnel:

    conn = mysql.connector.connect(
        host=db_host,
        port=tunnel.local_bind_port,
        user=db_user,
        password=db_password,
        database=db_database
```

Efterfølgende bliver der dannet en forbindelse fra Raspberry Pi til databaseserveren. Host er databaseserverens ip, dernæst bruger og kodeord for at logge ind. Nederst angives hvilken database, som skal forbindes til.

```
while True:
    try:
        print("Hold en tag tæt på læseren")

        _, sid = reader.read() # Ignorerer uid på RFID-tag ved at bruge underscore, da dette vil være pladsspild i
databasen.
        print("sid:", sid)
        sleep(2)

        cur = conn.cursor()
```

Dernæst startes en uendeligt while True løkke. På RFID kortet/tag, er der hardcoded et UID, som består af en talrække. Dette er det unikke id, som er tilknyttet det specifikke kort/tag. Grundet at der vil blive mange registreringer i databasen, på det enkelte uid er dette udeladt. I stedet bliver kort/tag skrevet med et kortere studie id, hvilket gør uid overflødigt. Derfor er det udeladt. Underscore udelader uid, og tager i stedet kun sid. Som i dette system er fra 1-50. Hvilket sparer plads i databasen.

Cur = conn.cursor() , gør at det er muligt at håndtere resultaterne af en databaseforespørgsel. Samt at eksekvere sql kommandoer, og få adgang til resultaterne heraf.

```
# Henter klasse for den studerende baseret på sid
hentklasse = "SELECT klasse FROM studerende WHERE sid = %s"
cur.execute(hentklasse, (sid,))
klasse_result = cur.fetchone()
```

hentklasse er en variabel, som indeholder en sql kommando. Denne kommando kigger på sid, og henter klasse fra studerende tabellen. Dette bruges længere nede i koden. Dernæst bliver kommandoen eksekveret. Cur.fetchone() med resultatet af kommandoen bliver lagt ind i en variabel, som bliver brugt i nedenstående kode.

```
if klasse_result:
    # Hvis der er en klasse tilknyttet til den pågældende studerende
    klasse = klasse_result[0]
    skemaNavn = "skema" + klasse
    dato = date.today()
    cur.execute("SELECT dato FROM {} WHERE dato = %s".format(skemaNavn),(dato,))
    result = cur.fetchone()
    if result:
        # Hvis der er et resultat, betyder det, at der er undervisning i dag
        fm = 1 # Sætter fm til 1, hvilket betyder fremmøde, og 1 indsættes i fm i

        # Tjekker om rækken allerede eksisterer i tabellen
        cur.execute("SELECT sid FROM {} WHERE dato = %s AND sid = %s".format(klasse) ,(dato,sid))
        existing_row = cur.fetchone()
```

Hvis der er indhold i klassevariablen, bliver denne lavet om til en ny variabel klasse. Indholdet bliver dernæst lavet om til endnu en variabel, som sætter skema foran. Så i tilfælde at at klassen er it, kommer den nye variabel til at hedde skemaitt. Dernæst tjekker koden, om der er undervisning. Hvis der er et resultat, tjekker koden, om rækken allerede eksisterer.

```

if existing_row:
    # Hvis rækken allerede eksisterer, ignorer indsættelsen
    lcd.clear()

    lcd.write_string("OK tjek.")
    sleep(3)
    lcd.clear()

    else:
        # Hvis rækken ikke eksisterer, indsæt data i tabellen
        cur.execute("INSERT INTO {} (dato, sid, fm) VALUES (%s, %s, %s)".format(klasse),(dato, sid, fm))
        conn.commit()
        lcd.clear()
        lcd.write_string("Indtjekning ok.")
        sleep(3)
        lcd.clear(

```

Hvis der er et resultat i existing_row, bliver der skrevet ok tjek. Hvis der ikke er et resultat, bliver data indsat i en ny række, og eksekveret. Ændringerne gemmes derefter til tabellen.

```

else:
    # Hvis den studerende forsøger at tjekke ind, når der ikke er undervisning ifølge skemaet.
    lcd.clear()

    lcd.write_string("Du har ikke undervisning i dag.")

    sleep(3)

    lcd.clear()

else:
    # Hvis der ikke er nogen klasse tilknyttet til den pågældende studerende
    lcd.clear()

    lcd.write_string("Ugyldig studerende.")
    sleep(3)
    lcd.clear()

```

De to sidste else sætninger fører tilbage til tidligere kode. Den første fører tilbage til skema tjekket. Om den studerende har undervisning. Den sidste fører tilbage til klasse. Hvis den studerende, ikke er registreret i en af klasserne.

Test

De funktionelle krav, som beskrevet i kravspecifikationen:

- Udtrække fravær på den enkelte studerende.
- Udtrække fravær klassevis.
- Oprette ny studerende.
- Oprette ny klasse.
- Oprette nyt skema
- Slette klasse.
- Slette studerende.
- Redigere studerendes fremmøde.
- Bruger interface, som kan bruge ovenstående funktioner.

Oprette ny studerende.

Ved oprette en ny elev, tilføjes denne til studerende tabellen.

Tilføj elev til database

Skriv SID på elev her:

Skriv navn på elev her:

Skriv klasse på elev her:

Slet elev fra database

2023-02-02	49	Valdemar Nielsen	DAT
2023-02-02	50	Mille Møller	DAT
2023-06-12	55	Jens Hansen	itt

Slette studerende.

Jens Hansen som blev tilføjet til studerende tabellen, bliver slettet.

Slet elev fra database

Skriv SID på elev, du vil slette:

2023-02-02	49	Valdemar Nielsen	DAT
2023-02-02	50	Mille Møller	DAT

Oprette ny klasse.

Tilføje en tabel, til en ny klasse.

Tilføj ny klasse

Skriv navn på ny klasse på 3 bogstaver, f.eks. dat eller itt:

Tilføj ny klasse

test2

208,0 Ki

byg

32,0 Ki

dat

64,0 Ki

itt

48,0 Ki

test2.byg: 0 antal rækker i alt (tilnærmelsesvis)

SID	Fravær	Dato
-----	--------	------

Oprette nyt skema.

Skemaet bliver oprettet i administrator interface.

Opret nyt skema

Skriv skema + navn på klasse, f.eks. skemaitt eller skemadat:

Opret skema

test2

224,0 Ki

byg

32,0 Ki

dat

64,0 Ki

itt

48,0 Ki

skemabyg

16,0 Ki

skemadat

16,0 Ki

test2.skemabyg: 0 antal rækker i alt (tilnærmelsesvis)

Dato	Fag
------	-----

Slette tabel skema/klasse/studerende

Slet table fra database

Skriv navn på table, der skal slettes:

Slet table

▼ test2	192,0 KiB
dat	64,0 KiB
itt	48,0 KiB
skemabyg	16,0 KiB
skemadat	16,0 KiB

Redigere en skoledag

Her slettes d 2/2 fra tabel skemaidd.

test2.skemaidd: 69 antal rækker i alt (tilnærmelsesvis)

dato ▲1	fag
2023-02-02	Programmering
2023-02-03	Netværksteknologi
2023-02-06	Netværksteknologi

Slet skemadag fra skema table

Skriv navn på klasse, der skal have slettet dag:

Skriv dato på skemadag, der skal slettes (YYYY-MM-DD):

Slet skemadag

test2.skemaidd: 68 antal rækker i alt (tilnærmelsesvis)

dato ▲1	fag
2023-02-03	Netværksteknologi
2023-02-06	Netværksteknologi

Oversigt over klasser

De to forskellige klasser vises.

Tryk her for at se klasse oversigt

Tryk her for at se klasse oversigt

{'DAT', 'itt'}

Se fravær på en enkelt elev.

Malthe Jensen har 54% fravær.

Se fravær på én elev

Skriv UID på elev, du vil have fravær for:

Malthe Jensen har 54.24% fravær

[Se fravær](#)

Fravær for en hel klasse.

Nedenunder ses fravær for hele itt klassen.

Her kan du se en hel classes fravær

Skriv navn på klasse, du vil have fravær for:

{Mathilde Jensen har 75.76% fravær
 }{Emma Nielsen har 63.64% fravær
 }{Noah Hansen har 57.58% fravær
 }{Olivia Pedersen har 57.58% fravær
 }{William Christensen har 54.55% fravær
 }{Ida Møller har 57.58% fravær
 }{Lucas Olsen har 60.61% fravær
 }{Clara Jensen har 60.61% fravær
 }{Emil Petersen har 63.64% fravær
 }{Freja Hansen har 57.58% fravær
 }{Magnus Nielsen har 58.82% fravær
 }{Sofia Pedersen har 60.29% fravær
 }{Victor Christensen har 54.41% fravær
 }{Isabella Møller har 55.88% fravær
 }{Mikkel Olsen har 51.47% fravær
 }{Emilie Jensen har 58.82% fravær
 }{Noah Christensen har 66.18% fravær
 }{Laura Larsen har 57.35% fravær
 }{Mads Rasmussen har 57.35% fravær
 }{Albert Sørensen har 60.29% fravær
 }{Ella Nielsen har 47.06% fravær
 }{Felix Hansen har 63.24% fravær
 }{Luna Pedersen har 54.41% fravær
 }{Liam Christensen har 60.29% fravær
 }{Maja Møller har 60.29% fravær
 }

[Se fravær for klasse](#)

Redigere fravær for en studerende.

test2.itt: 560 antal rækker i alt (tilnærmelses)

dato	▲1	sid	fm
2023-02-02		13	1
2023-02-02		23	1

Rediger i fravær

Skriv UID på elev, der skal have ændret fravær:

Skriv dato på dag, der skal ændres (YYYY-MM-DD):

[Rediger fravær](#)

test2.itt: 559 antal rækker i alt (tilnærmelsesvis)

dato	sid	fm
2023-02-02	14	1
2023-02-02	18	1

Krav til løsningen:

- Systemet skal selv boote, ved strømsvigt.
- Systemet skal være hurtigt, eks. hente fremmøde for en klasse på under 2 sekunder.
- Systemet skal programmeres i Python/SQL.
- Systemet skal kommunikere med database.
- Systemet skal bruge en sikkerforbindelse til databasen

Ved boot/strømsvigt skal systemet boote.

Dette er blevet tjekket på følgende måde:

```
kim29@kimmo:~$ sudo systemctl status tid.service
● tid.service - tjek
   Loaded: loaded (/lib/systemd/system/tid.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2023-06-12 10:25:07 CEST; 29s ago
     Process: 504 ExecStartPre=/bin/sleep 60 (code=exited, status=0/SUCCESS)
    Main PID: 1258 (python3)
       Tasks: 4 (limit: 1599)
          CPU: 1.538s
      CGroup: /system.slice/tid.service
              └─1258 /usr/bin/python3 /home/kim29/code/sp/tjekin.py

Jun 12 10:23:55 kimmo systemd[1]: Starting tjek...
Jun 12 10:25:07 kimmo systemd[1]: Started tjek.
kim29@kimmo:~$
```

Yderligere starter systemet selv op, hvis der skulle opstå en fejl.

Systemet skal være hurtigt, eks. hente fremmøde for en klasse på under 2 sekunder.

For at lette arbejdsbyrden på systemet. Er fraværs tabellen delt op i klassefravær, så der er en fraværstabel på hver klasse. Dette er blevet gjort på baggrund af at hvis alt fraværs data blev sat ind i samme tabel, ville denne tabel indeholde 220*9000 rækker på et år. Omkring 1.900.000. Dette ville resultere i en database, som ville reagere langsomt, når der skal hentes data ud. Når data skal hentes på en studerede, vil dette være hurtigt. Men hvis der skal hentes fravær på en hel klasse, vil dette tage længere tid, som det kan ses nedenunder.

```
Maja Møller har -69.23% fravær
Query tid: 10.653244018554688 seconds
```

Denne forespørgsel bruger 10 sekunder på at hente hele itt klassen, i en database, hvor der er omkring 1.000.000. rækker, hvilket svarer til ca. et halvt års fravær.

Nedenunder vises den tid, det tager når fraværet er klasse inddelt, i stedet for at alt fraværet er samlet i en tabel.

```
Mille Møller har 15.25% fravær
Query tid: 1.1952717304229736 seconds
```

Begge forespørgsler er blevet brugt lokalt. Det må antages at tage længere tid, når det kører over internettet. Når det kører over internettet, kan det være svært at finde en fastlagt tid på søgning i databasen, da internettets hastighed varier meget i løbet af en dag.

Systemet skal bruge en sikkerforbindelse til databasen.

Forbindelsen til databasen, sker via SSH. Ved at bruge Wireshark, kan det verificeres, at data sendes via SSH.

997	51.018013	fe80::3894:bb5d:ba8...	fe80::c95:ab27:597f:bd1d	SSH	134	Client: Encrypted packet (len=60)
998	51.022821	fe80::c95:ab27:597f...	fe80::3894:bb5d:ba8e:afa1	SSH	134	Server: Encrypted packet (len=60)
1005	51.813732	fe80::3894:bb5d:ba8...	fe80::c95:ab27:597f:bd1d	SSH	126	Client: Encrypted packet (len=52)
1007	51.859526	fe80::3894:bb5d:ba8...	fe80::c95:ab27:597f:bd1d	SSH	126	Client: Encrypted packet (len=52)
1011	52.020257	fe80::c95:ab27:597f...	fe80::3894:bb5d:ba8e:afa1	SSH	118	Server: Encrypted packet (len=44)

Test af lcd display.

Når den studerende tjekker ind, og denne har undervisning den pågældende dag, vises dette på lcd displayet.



Efterfølgende bliver rækken indsat på i tabellen i databasen, som vises her under.

dato	sid	fm
2023-06-05	21	1

Hvis den studerende er i studerende tabellen, men ikke har undervisning den pågældende dag, så får den studerende følgende besked i lcd displayet.



Hvis den studerende allerede er tjekket ind, og denne tjekker ind igen, vises følgende på lcd displayet. Forskellen på denne besked og indtjekning ok, er at rækken ikke indsættes i tabellen igen.



Hvis den studerende, ikke har en klasse tilknyttet, regnes sid som ugyldigt, og følgende besked printes på lcd displayet.



Resten af funktionerne er gennemtestede, og virker efter hensigten.

Konklusion

I projektperioden har vi været vidt omkring diverse løsninger af systemet. Der har været op og nedture med de mange arbejdsopgaver. Det har været svært at få mange af løsningerne til at fungere, og hver gang vi fik noget til at fungere, så var der noget andet, der var stoppet med at fungere. Med database opsætning og forskellige funktioner, bliver det hurtigt komplekst især, hvis skal indsættes nye kolonner i tabellerne for at opnå den ønskede virkning af funktionerne. Der er mange opmærksomhedspunkter for hver funktion. Et eksempel kunne være tilføjes af en ny studerende. Hvis en elev starter i en klasse midt inde i et skoleår, kan den fremgangsmåde, som projektet startede med at bruge til at udregne fravær ikke bruges, da den nye elev ville få fravær fra semesterstart.

I starten af opgaven opstillede vi spørgsmålet: Hvordan kan vi lave et RFID baseret it-system, der kan behandle studerendes fravær.

Vi valgte at løse problemformuleringen ved at lave et system, der automatisk giver fravær til personer, der ikke stempler sig selv ind via deres RFID kort, og gemmer det hele i en database, der ligger på serveren på skolen. Databasen er opsat med 5 tabeller, som har en relation til hver især. Fra et sikkerhedsperspektiv har det været nødvendigt, at enten kryptere data, eller sende det via en sikker forbindelse. Da krypteringen fejlede, valgte vi i stedet at sende data via en SSH-forbindelse.

Vi har forsøgt at optimere databasen ved at gemme så få datarækker i databasen som muligt, så det ikke kommer til at fylde for meget på serveren. Derfor valgte vi, at databasen IKKE skal logge folk, der er fraværende, systemet logger kun, hvis man er til stede. I stedet tæller systemet antal dage, de har skulle møde sammen, samt de dage de har været til stede, og udregner en fraværprocent ud fra disse data.

Jævnfør de krav, som vi selv havde til projektet, mener vi at kravene er blevet opfyldt. Systemet er hurtigt til at hente data, har de påkrævede funktioner, og sikkerheden er forsøgt opfyldt. Som udgangspunkt har vi valgt kun at bruge Open Source-software, til udførsel af RFID-systemet, dette for at holde omkostningerne nede.

Hjælper et fraværssystem reelt på problemet med frafald på uddannelser? Umiddelbart mener vi, at det har en begrænset effekt. Det mener vi, da der kan være mange andre faktorer end fravær, der spiller ind, når folk vælger at droppe ud af deres uddannelse. Der er mange, der måske finder ud af, at de hellere vil have en anden uddannelse, der er nogen folk, der starter bare for at få SU (de vil selvfølgelig blive opdaget hurtigere). Desuden, så er der en rimelig klar forbindelse mellem fravær og frafald, men om eleven selv får lov til at holde styr på sit eget fravær eller læreren gør det, så kommer det højst sandsynligt ikke til at ændre på om, eleven vælger at droppe ud eller ej.

Perspektivering

Sideløbende med projektet har vi prøvet at erfare ny viden i form af forebyggelse mod SQL-injections og har efter bedste evne, forsøgt at implementere parametriserede forespørgsler. Det er noget der ville være fuldt implementeret ved en eventuel fuldført produktion af systemet. Som systemet er bygget nu, gøres der brug af parametriserede forespørgsler i alle funktioner i administrator interface. Dog ikke input validation eller sanitization. Da vores erfaring var utilstrækkelig. Havde erfaringen været der, kunne man med fordel, lave nogle funktioner til at køre strings igennem og tjekke efter længden, og give den forskellige karakter og længde constraints for at opretholde en god input-validation, ydermere kunne man udvikle en funktion til at escape strings for at systemet havde en vis grad sanitization.

Vi valgte at afgrænse redundancy i denne projektperiode, da det hurtigt kan blive meget omfattende. Dog ville det være et emne der skulle tages op og implementeres ved produktion. Backup pt, består af ugentlige lokale eksporteringer af diverse tabeller, men i og med at det er en enkelt central database, ville det være at foretrække at alt data blev eksporteret til cloud og ikke lokalt. Derudover ville det være bedst organiseret med en funktion til automatisk at slette forældet data og dermed holde en ren og responsiv database.

For at øge sikkerheden, udtænkte vi nogle mulige udvidelser der kunne øge sikkerheden i systemet. Der kunne med fordel anvendes en tastatur til at indtaste en unik kode ved check-in for at forebygge mod RFID kloning. Ydermere kunne der knyttes en key til hvert studienummer på database niveau for at undgå at folk kan gætte sig til studienummeret og manipulere systemet. Udover disse sikkerhedsmæssige ekstra-funktioner, ville der med fordel kunne implementeres funktioner rettet mod brugervenlighed for administratorer og lærere. En funktion til at notificere når en elev når en vis fraværsprocent, enten i form af mail eller sms og en funktion til at indsætte data i DB via import af en CSV-fil, dette kan udføres på mange måder og ville hjælpe administratorer med at oprette hele klasser og skemaer. Samtidig kunne det give god mening at bruge et større display ved tjek ind, og gøre indtjekningen mere grafisk. Og evt. give de studerende mulighed, for at bruge deres telefon til indtjekning.

Bibliografi

- [1] M. R. Shanika Wickramasinghe, »BMC,« 09 12 2021. [Online]. Available: <https://www.bmc.com/blogs/dbms-database-management-systems/>. [Senest hentet eller vist den 11 06 2023].
- [2] »FT,« 19 09 2022. [Online]. Available: https://www.ft.dk/da/statsrevisorerne/nyheder/2022/09/beretning_21_2021. [Senest hentet eller vist den 17 05 2023].
- [3] UCN, »Ucn årsrapport 2022,« UCN, Aalborg, 2022.
- [4] UCN, »UCN,« [Online]. Available: <https://ucn.dk/om-ucn/tal-og-fakta>. [Senest hentet eller vist den 17 05 2023].
- [5] UCN, »Organisationsdiagram,« UCN.
- [6] Danmarks Statistik, »Danmarks Statistik,« [Online]. Available: <https://www.dst.dk/da/Statistik/nyheder-analyser-publ/nyt/NytHtml?cid=44274>. [Senest hentet eller vist den 18 05 2023].
- [7] H. M. Hvarregaard, Interviewee, *Interview med Henrik*. [Interview]. 26 03 2023.
- [8] E. Staunstrup, Organisationen, Trojka, 2022.
- [9] E. Staunstrup, Organisationen, Trojka, 2017.
- [10] C. a. Line bruun, »organisationskultur,« [Online]. Available: <https://sites.google.com/site/organisationskultur3hhb/home>. [Senest hentet eller vist den 20 05 2023].
- [11] forlaget94, »forlaget94,« [Online]. Available: <http://forlaget94.dk/cms/wp-content/uploads/F94-TM-2012-B1-K1-Prove.pdf>. [Senest hentet eller vist den 27 05 2023].
- [12] E. staunstrup, Organisationen, 2021.
- [13] S. amsler, »techtarger.com,« marts 2021. [Online]. Available: <https://www.techtarger.com/iotagenda/definition/RFID-radio-frequency-identification>. [Senest hentet eller vist den 22 maj 2023].
- [14] B. Mehl, »getkisi,« 1 dec 2022. [Online]. Available: <https://www.getkisi.com/blog/how-to-copy-access-cards-and-keyfobs>. [Senest hentet eller vist den 7 juni 2023].
- [15] Sick, »Sick,« [Online]. Available: <https://www.sick.com/dk/da/topics-and-knowledge/rfid-know-how-identification-using-rfid-technology/w/rfid/>. [Senest hentet eller vist den 17 05 2023].
- [16] »Michelin,« [Online]. Available: <https://www.michelin.dk/rfid>. [Senest hentet eller vist den 17 05 2023].

- [17] S. campbell, »Circuitbasics.com,« 06 07 2017. [Online]. Available: <https://www.circuitbasics.com/raspberry-pi-i2c-lcd-set-up-and-programming/>. [Senest hentet eller vist den 05 2023].
- [18] S. >. Vutborg, »Professionshøjskolen UCN,« 2022. [Online]. Available: https://teams.microsoft.com/_?culture=da-dk&country=DK&lm=deeplink&lmsrc=homePageWeb&cmpid=WebSignIn#/apps/3e0a4fec-499b-4138-8e7c-71a9d88a62ed/sections/MyNotebook. [Senest hentet eller vist den 23 05 2023].
- [19] S. R. Vutborg, »sharepoint.ucn,« 2022. [Online]. Available: https://teams.microsoft.com/_?culture=da-dk&country=DK&lm=deeplink&lmsrc=homePageWeb&cmpid=WebSignIn#/apps/3e0a4fec-499b-4138-8e7c-71a9d88a62ed/sections/MyNotebook. [Senest hentet eller vist den 2 06 2023].
- [20] MariaDB.com, »MariaDB.com,« [Online]. Available: <https://mariadb.com/kb/en/connecting-to-mariadb/>. [Senest hentet eller vist den 31 05 2023].
- [21] W3Schools, »SQL-injection,« 2019. [Online]. Available: https://www.w3schools.com/sql/sql_injection.asp. [Senest hentet eller vist den 19 maj 2023].
- [22] R. Harwood, »microsoft.com,« 1 11 2023. [Online]. Available: <https://www.spiceworks.com/it-security/network-security/articles/what-is-ssh/>. [Senest hentet eller vist den 3 juni 2023].
- [23] V. kanade, »spiceworks.com,« 25 april 2023. [Online]. Available: <https://www.spiceworks.com/it-security/network-security/articles/what-is-ssh/>. [Senest hentet eller vist den 1 juni 2023].

Figur 1 - Database forsidebillede [1].....	1
Figur 2 - Organisationsdiagram, [5]	8
Figur 3 - PAEI model [9]	12
Figur 4 - Fagbureaukratiet, [10]	13
Figur 5 - Organisations/kultur typer, Organisationen, 2022, kap 9	14
Figur 6 - RFID Reader [15]	17
Figur 7 - i2c Bus protocol states, Steffen rahbek vutborg, 2022, [18]	18