Projektrapport

IBA

Dkpbi23a2a

Eksamensprojekt – ”personregistrering”

Gruppe: Mohammad Fahad, Nickolaj, Allan

Forside:

Login: JECO og SKQU

Indholdsfortegnelse

[Problemanalysen 1](#_Toc165097224)

[ Research 1](#_Toc165097225)

[ Problemformulering 1](#_Toc165097226)

[ Problemstilling 1](#_Toc165097227)

[Målgruppeanalyse 2](#_Toc165097228)

[ User Story 2](#_Toc165097229)

[ Kravanalyse 2](#_Toc165097230)

[Projektforløb 2](#_Toc165097231)

[ Afgrænsning 2](#_Toc165097232)

[ Succeskriterier 2](#_Toc165097233)

[ Kravspecifikation 2](#_Toc165097234)

[Design 2](#_Toc165097235)

[ Design værktøjer 2](#_Toc165097236)

[ Adfærds analyse 2](#_Toc165097237)

[ Arkitektur design 2](#_Toc165097238)

[ Koncept evaluering 2](#_Toc165097239)

[ Teknologi beskrivelse 2](#_Toc165097240)

[ Mock-up 2](#_Toc165097241)

[Prototype 2](#_Toc165097242)

[ Værktøjer 3](#_Toc165097243)

[ Brugergrænseflade 3](#_Toc165097244)

[ Systemudvikling 3](#_Toc165097245)

[ Validering og verificering 3](#_Toc165097246)

[Diskussion 3](#_Toc165097247)

[ Problemformulering 3](#_Toc165097248)

[ Kvalitet 3](#_Toc165097249)

[ Metoder 3](#_Toc165097250)

[ Perspektivering 3](#_Toc165097251)

[Konklusion 3](#_Toc165097252)

[Litteraturliste 3](#_Toc165097253)

[Bilag: 3](#_Toc165097254)

# Resumé:

# Problemanalysen

Research:

1. Mindre research efter eksisterende personregistrerings systemer til uddannelsesinstitut og vi skulede til IBAs eget system og har taget det som udgangspunkt. Der valgt at have typer ansatte Studerende, Pedel/kantine og Underviser, de skal have adgang til lidt forskellige informationer.
   1. Studerende: Egen information (Fornavn, Efternavn, Mail), hold/kursus, lokale.
   2. Underviser: Egen information, som studerende, hold/kursus, lokale og tilmeldte studerende til kurser.
   3. Pedel/Kantine: Egen information (fornavn, Efternavn, Mail.
2. ”En simpel login”: Dette refererer typisk til design og funktionalitet af GUI. Og indebærer ikke nødvendigvis fraværet af sikkerhedsforanstaltninger som kryptering. Vi fortolker det, at selve brugergrænsefladen er ukompliceret og let at bruge. Der udføres dybere research senere i analysen vedrørende sikkerhed og kryptering.
3. Gemme/Læse IP-adresser: Ligger op til at kunne justere konfigurationsindstillinger uden at skulle re-deploy eller direkte ændre i selve applikationen. Et nyttigt område, hvor der er brug for fleksibilitet angående brug af forskellige databaser. Altså en dynamisk ændring uden dybere indgreb i funktionaliteten. Her vil der også fortages dybere reseach senere i analysen.
4. Diverse udleverede teknologier, har vi kørt research på for at styrke disse. Og for at kunne implementere sikkerhed og OOP ind.

Problemformulering:

Hvordan kan vi designe og udvikle et brugervenligt, sikret og skalerbart personregistreringssystem baseret på MS SQL Server og C# til en uddannelsesinstitution i Danmark, som opfylder følgende krav:

1. Muliggør adgang for studerende til deres personlige oplysninger, tilmeldte kurser og oplysninger om lokaler.
2. Giver undervisere adgang til personlige oplysninger, kurser, lokaledetaljer og lister over tilmeldte studerende.
3. Giver øvrige personale adgang til personlige oplysninger.
4. Tillader administration af applikationen gennem en admin adgang, hvor brugere (ansatte og studerende), kurser, lokaler og kursustilmeldinger kan håndteres.
5. Inkluderer at logge alle aktiviteter for at lette fejlsøgning og opklaring af misforståelser, som kun adm har adgang til

Udviklingenen skal omfatte:

* En C#-baseret Enterprise-applikation til databaseadministration med en simpel UI, som er kompatibel på tværs af Mac og Windows.
* En C# Web Server-applikation til håndtering af read-only dataanmodninger fra en klient browser-applikation via et REST API, med logning af forespørgsler i XML/JSON-format.
* Dynamisk konfigurerbar databaseforbindelsesindstillinger gemt i JSON-format for nem vedligeholdelse og skalerbarhed.

Problemstilling:

Udviklingen af et personregistreringssystem for en dansk uddannelsesinstitution kræver en omhyggelig tilgang til design og funktionalitet for at sikre sikker og effektiv håndtering af ansatte og studerendes data. Overvejelser skal omfatte:

1. **Database Design**: Skabe en databasestruktur i 3NF, der understøtter nøjagtig og effektiv håndtering af datamængden.
2. **Enterprise Applikation**:
3. Udvikling af en brugervenlig brugergrænseflade.
4. Implementering af en enkel login-dialog til autentificering.
5. Evnen til at håndtere og udnytte Stored procedures, Funktionel procedures og Views i databasen.
6. Fleksibilitet i konfiguration af databaseforbindelser, med mulighed for at ændre forbindelsesdetaljer som IP og login oplysninger.
7. Funktionalitet til dynamisk at læse og gemme konfigurationsdata, såsom IP-adresser, fra JSON filer.
8. **C# Web Server Applikation**:
9. Konstruktion af en C# webserver, der sikrer pålidelig kommunikation med en klient-baseret browser-applikation.
10. Brug af HTML, CSS og JavaScript til at udforme en interaktiv og responsiv brugergrænseflade.
11. Implementering af et Rest API for datatransmissioner, som både modtager og sender data i JSON format.
12. Garanti for read-only adgang til databaseinformation gennem brug af SQL-views, fra Bruger-GUI.
13. Detaljeret logning af alle henvendelser til webserveren i JSON format til understøttelse af sporbarhed og fejlfinding. XX
14. **Sikkerhed**:
15. Anvendelse af sikkerhedstiltag for databeskyttelse inklusive datakryptering, brugeradgangsstyring og overholdelse af databeskyttelsesregler.
16. **Test og Dokumentation**:
17. Gennemførelse af omfattende systemtests for at bekræfte dataflows og komponentintegreringens integritet.
18. Grundig dokumentation af alle systemaspekter, som tjener som referencemateriale for nuværende og fremtidige brugere og udviklere.
19. Grundig dokumentation af alle systemaspekter, som tjener som referencemateriale for nuværende og fremtidige brugere og udviklere.

# Målgruppeanalyse

User Story:

**Bruger**: Administrator

**Handling**: Tilføj ny medarbejder eller studerende til systemet

**Mål**: Registrere nye ansatte eller studerende i systemet for at sikre korrekt håndtering og opbevaring af deres data.

**User Story:**

Som administrator ønsker jeg at kunne tilføje nye medarbejdere eller studerende til systemet for at sikre, at alle relevante oplysninger(lokaler og kurser) er korrekt registreret og opbevaret i databasen.

**Bruger**: Studerende

**Handling**: At studerende kan se deres egne data

**Mål**: At give den studerende adgang til at se deres egne data for at kunne holde styr på deres tilmeldinger og kurser.

**User Story:**

Som studerende ønsker jeg at kunne se mine tilmeldte lokaler og kurser for at have en oversigt over mine aktuelle studieplan og aktiviteter.

Kravanalyse:

**Funktionelle krav (Administrator):**

1. Brugeren skal kunne logge ind på systemet ved hjælp af et simpelt login-dialogvindue.
2. Brugeren skal have adgang til en formular til at indtaste oplysninger om den nye medarbejder eller studerende.
3. Systemet skal validere de indtastede oplysninger for at sikre korrekthed.
4. Efter godkendelse skal de indtastede oplysninger gemmes i databasen.
5. Systemet skal kunne behandle og lagre ikke-fast information som f.eks. IP-adresse i XML- eller JSON-filer.
6. Brugeren skal have mulighed for at ændre IP-adresse, loginoplysninger osv. Til MS SQL Serveren.
7. **Ikke-funktionelle krav:**
8. Systemet skal være baseret på en MS SQL Server opbygget med database tabeller i 3NF normalisering.
9. Enterprise-applikationen skal udvikles i C#.
10. UI-teknologien kan være valgt efter udviklerens præference, og det skal fungere på både Mac og Windows.
11. Systemet skal kunne håndtere stored procedures, funktioner, views osv.
12. Kommunikationen mellem Web Server og klienten skal være baseret på Rest API og XML/JSON-beskeder.
13. Web serveren skal logge alle request i XML/JSON-baseret tekstfil.
14. C# Web Server-applikationen skal udvikles og implementeres til at håndtere kommunikationen mellem en readonly klient browser applikation og MS SQL Serveren ved hjælp af HTML, CSS og JavaScript.

**Funktionelle krav(burger):**

1. Brugeren skal kunne logge ind på systemet med deres personlige loginoplysninger.
2. Efter vellykket login skal brugeren have adgang til en oversigtsside, der viser deres tilmeldte lokaler og kurser.
3. Brugeren skal kunne filtrere eller søge efter specifikke oplysninger, f.eks. kurser eller datoer.
4. Systemet skal vise oplysningerne på en brugervenlig måde, f.eks. i form af en liste eller et kalenderformat.
5. **Ikke-funktionelle krav:**
6. Brugergrensesnittet skal være let at navigere og intuitivt at bruge for studerende med forskellige tekniske færdigheder.
7. Systemet skal have hurtig respons og være tilgængeligt med minimal ventetid for at give en god brugeroplevelse.
8. Brugerdata skal være sikret og beskyttet mod uautoriseret adgang eller misbrug, f.eks. gennem kryptering af data under transmission og opbevaring.
9. Systemet skal være kompatibelt med forskellige webbrowsere og enheder, så studerende kan få adgang til deres data fra forskellige platforme, herunder computere, tablets og smartphones.

# Projektforløb

Afgrænsning:

* Det dynamiske i en kalender funktion vedrørende studerende og ansatte.
* ADM login kan kun arbejde på PC Microsoft.
* Login sikkerhed -

Succeskriterier:

* Opgave:
  + Gruppen godkender opgave løsningen, når opgavekrav er fuldført efter pensum og aftale.
* Milepæl:
  + Når alle opgaver i milepælen er godkendt af gruppen.
* Rapport:
  + Når der er en fyldestgørende rapport jf til kravene i projektbeskrivelsen.
  + En klar sammenhæng mellem problem, løsning og applikation.
  + Rapporten opfylder de beskrevne krav.
  + alle i gruppen er tilfredse, så kan den godkendes.
* Applikationen:
  + Kravspecifikationen er opfyldt og godkendt af gruppen.
  + Direkte sammenhæng mellem Problemstilling og resultat.
* Projekt:
  + Når præsentationen er vel forsvaret og overstået.

Kravspecifikation: Detaljerede tekniske og brugerorienterede krav.

# Design

Planlægning og skitsering af systemets arkitektur og brugerinteraktion:

* Design værktøjer: Anvendelse af software som UML-diagrammer til strukturering.
* Adfærds analyse: Studie af systemets reaktioner under forskellige betingelser.
* Arkitektur design: Opbygning af databasen og applikationerne.
* Koncept evaluering: Vurdering af det initielle designforslag - Det involverer at sammenholde konceptet med projektets mål og kravspecifikationer for at identificere styrker, svagheder, og potentielle forbedringer. Dette trin inkluderer:
  + Gennemgang af systemets layout og navigationsflow.
  + Evaluering af brugergrænsefladens intuitivitet og tilgængelighed.
  + Vurdering af datamodellens skalérbarhed og fremtidssikring.
  + Sikkerhedsevaluering, herunder beskyttelse af følsomme data.
  + Teknisk gennemgang af systemarkitekturen og dens evne til at understøtte vedligeholdelse og udvidelse.
  + Feedback indsamling fra potentielle brugere og tekniske eksperter.
* Teknologi beskrivelse: Beskrivelse af anvendte teknologier og frameworks.
* Mock-up: Visuel prototype af brugergrænsefladen.

# Prototype

Evaluering og justering af design baseret på mock-up erfaringer:

* Værktøjer: Revision af anvendte design- og udviklingsværktøjer.
* Brugergrænseflade: Optimering af interaktionsdesign.
* Systemudvikling: Implementering af ændringer i systemet.
* Validering og verificering: Sikring af at systemet møder alle kravspecifikationer.

# Diskussion

Refleksion over projektets forløb og resultater:

* Problemformulering: Evaluering af Problemformuleringen.
* Kvalitet: Evaluering af systemets funktionalitet og brugervenlighed.
* Metoder: Vurdering af anvendte forsknings- og udviklingsmetoder.
* Perspektivering: Fremtidige forbedringer og potentielle nye features.

# Konklusion

(Opsummering af projektets resultater og læring.)

# Litteraturliste

# Bilag:

Arbejdsfordeling