**Projektuppgift**

*DT162G*

**Javascriptbaserad webbutveckling: Projekt**

AI Datorer AB:s MERN-stacksbaserade Intranät

**Max Karlstedt**

**MITTUNIVERSITETET  
Avdelningen för informationssystem och -teknologi**

**Författare:** Max Karlstedt, [maka2207@student.miun.se](mailto:xxxxx0000@student.miun.se)  
**Utbildningsprogram:** Webbutveckling, 120 hp  
**Huvudområde:** Datateknik  
**Termin, år:** HT, 2023

# Sammanfattning

**Syfte:** Webbprojektet avser AI Datorer AB:s MERN-stacksbaserade Intranät i syfte att kunna inventera sitt lager av datorkomponenter såväl som systemadministratörens förmåga att administrera behöriga användare med i sin tur olika behörigheter till det MERN-stacksbaserade Intranätet. **Teori:** a. **Metod:** a. **Konstruktion:** a. **Resultat:** a. **Slutsatser:** a. Källförteckning och eventuella bilagor tillkommer.

# Innehållsförteckning

[1 Introduktion 1](#_Toc154900035)

[1.1 Bakgrund och kunden 1](#_Toc154900036)

[1.2 Kundens målgrupp 1](#_Toc154900037)

[1.3 Kundens budget 1](#_Toc154900038)

[1.4 Juridiska överväganden 1](#_Toc154900039)

[1.5 Överenskommen kravspecifikation 2](#_Toc154900040)

[1.6 Kapitelöversikt 3](#_Toc154900041)

[2 Teori 4](#_Toc154900042)

[2.1 MVC-arkitekturen 4](#_Toc154900043)

[2.2 MERN-stacken 4](#_Toc154900044)

[2.2.1 MongoDB 4](#_Toc154900045)

[2.2.2 ExpressJS 4](#_Toc154900046)

[2.2.3 ReactJS 4](#_Toc154900047)

[2.2.4 NodeJS 4](#_Toc154900048)

[2.3 TailwindCSS 4](#_Toc154900049)

[2.4 REST API 5](#_Toc154900050)

[3 Metod 6](#_Toc154900051)

[3.1 Integrerad utvecklingsmiljö 6](#_Toc154900052)

[3.2 Publicering 6](#_Toc154900053)

[4 Konstruktion 7](#_Toc154900054)

[4.1 MongoDB+REST API 7](#_Toc154900055)

[4.2 NodeJS+ExpressJS 7](#_Toc154900056)

[4.3 ReactJS+TailwindCSS 7](#_Toc154900057)

[5 Resultat 8](#_Toc154900058)

[5.1 Registrering, inloggning och utloggning 8](#_Toc154900059)

[5.2 Hantera datorkomponenter 8](#_Toc154900060)

[5.3 Hantera användare 8](#_Toc154900061)

[6 Slutsatser 9](#_Toc154900062)

[Källförteckning 10](#_Toc154900063)

# Introduktion

Projektrapporten inleder med att berätta om kunden, deras bakgrund, målgrupp, budget (det vill säga, avgränsningar i vad som ska levereras), juridiska överväganden, samt den slutgiltiga överenskomna kravspecifikationen mellan parterna. Avslutningsvis under introduktionskapitlet presenteras de resterande kapitlen i form av en kapitelöversikt.

## Bakgrund och kunden

AI Datorer AB bedriver försäljning av datorkomponenter online sedan ett par år tillbaka. Företagets hantering av sina datorkomponenter sker dock mycket ineffektivt då data om datorkomponenterna är lagrade på ostrukturerade vis. Således är bolaget i behov av ett underhållbart intranät vilket i sin tur är lätt att förvalta för framtida ansvariga inom bolaget.

Bland annat önskar kunden att via en ny slags databas kunna lägga till, ändra, radera såväl som utläsa diverse verksamhetsnödvändig data om datorkomponenterna (namn, beskrivning, pris, antal, status huruvida de är begagnade eller nya, samt förknippade bilder till komponenten ifråga) de har till salu.

Även möjligheter att - via en så kallad systemadministratör - hantera användare (skapa nya, samt utläsa/radera och/eller ändra befintliga) och deras behörigheter att kunna sköta arbetet (inventariehantering av datorkomponenterna) via intranätet önskas av kunden. Med "användare" menas anställda vid bolaget AI Datorer AB.

## Kundens målgrupp

I och med att det rör sig om ett intranät åt bolaget så är målgruppen deras anställda såväl som den ansvariga systemadministratören vilket kommer att ha den högsta nivån av behörigheter i det färdiga MERN-stacksbaserade Intranätet.

## Kundens budget

Kunden är redo att lägga en del på denna beställning då det handlar om att kunna sköta inventeringen av datorkomponenter såväl som administrera ansvariga anställda att kunna göra det via det MERN-stacksbaserade intranätet.

## Juridiska överväganden

Kunden har gett samtycke till uppdragstagaren att ta fram en MongoDB där alla deras nuvarande data om datorkomponenter såväl som dess anställda kommer att överföras till.

## Överenskommen kravspecifikation

Följande har kunden kommit överens om att det ska implementeras i detta MERN-baserade webbprojekt:

* En backend med hjälp av NodeJS, ExpressJS och MongoDB där databas i MongoDB kommer att innehålla information om datorkomponenter och bolagets anställda. Här ska det gå att "CRUD:a" (skapa, utläsa, uppdatera, och radera) all data.
* En frontend med hjälp av ReactJS, TailwindCSS och ett JavaScript-baserat REST API som konsumerar denna backend så att bolagets anställda samt dess utnämnda Systemadministratör kan logga in i intranätet för att "CRUD:a" nödvändiga databasdata.
* Besökare vid intranätet ska kunna registrera sig vars konton sedan måste aktiveras av Systemadministratören innan de registrerade användarna kan logga in för att se något från intranätets databasdata.
* Endast en Systemadministratör med användarnamnet sysadmin kan registrera sig och får då den dedikerade rollen som Systemadministratör för intranätet.
* Registrerade användare kan först endast utläsa datorkomponenter medan övriga behörigheter måste läggas till manuellt av sysadmin. Behörigheter som går att få för en given användare är: utläsa, skapa nya, ändra och/eller radera befintliga datorkomponenter och/eller dess bilder. Med andra ord kan en användare - om så önskas - få behörigheterna att endast ändra bilder i datorkomponenter men inte något annat i datorkomponenterna eller ändra dem på något annat vis (radera, ändra, skapa nya).
* Systemadministratören ska ha en särskild undersida i frontend som endast syns för användarnamnet sysadmin där användare går att hantera: utläsa användare utifrån om de är (av)blockerade, (in)aktiverade, skapa nya användare med valfria behörigheter, ändra befintliga användare, radera befintliga användare.
* sysadmin har alla behörigheter som standard när kontot har skapats.
* En datorkomponent i MongoDB-databasen består av: id, namn, beskrivning, pris, antal, status (ny eller begagnad), kategorier och tillhörande bilder för datorkomponenten ifråga.
* En användare i MongoDB-databasen består av: registrerat användarnamn, samma användarnamn i endast små tecken, fullständigt namn, krypterat lösenord, en stränglista över roller (behörigheter) för användaren, en åtkomstnyckel, en uppdateringsnyckel, status om kontot är (av)blockerat, status om kontot är (in)aktiverat, och datum för senast inloggning för användaren ifråga.
* Utlästa datorkomponenter ska även inkludera särskilda knappar för att kunna utläsa, ändra eller radera en vald datorkomponent (beroende på behörighetsnivå; saknas behörighet för ändring så ska den knappen ej finnas och den undersidan ska ej gå att kunna navigera till).
* När en datorkomponent skapas i intranätet ska det gå att skapa det med eller utan bilder. Bilder som väljs ska kunna förhandsvisas och enskilda bilder ska kunna ändras/raderas innan själva datorkomponenten sedan skapas/läggs upp i intranätet.

## Kapitelöversikt

I kommande kapitel kommer följande att avhandlas:

[Kapitel 2](#_Teori) går igenom grundläggande teori för MVC-arkitekturen, MERN-stacken (MongoDB, ExpressJS, ReactJS, NodeJS), CSS-ramverket TailwindCSS, samt REST API.

[Kapitel 3](#_Metod) lägger fram metoden för hur det hela kommer att genomföras under de kommande kapitlen därpå genom att berätta om (utvecklings)verktyg som används samt hur det hela har publicerats.

[Kapitel 4](#_Konstruktion) berör konstruktionen av hela MERN-stacken indelat i de olika delarna: MongoDB+REST API, ExpressJS+NodeJS, ReactJS+TailwindCSS.

[Kapitel 5](#_Resultat) visar upp bilder och beskrivningar på de färdiga implementeringarna och hur de förhåller sig till kundens kravspecifikationer, det vill säga, i vilken utsträckning de har uppfyllt enligt kundens kravspecifikationer.

[Kapitel 6](#_Slutsatser) är slutsatser med reflektioner om (projekt)arbetet samt knyter ihop säcken för hela projektrapporten. Sedan följer vedertagen källförteckning och eventuella bilagor.

# Teori

Detta kapitel redogör grundläggande teori som behövs för att förstå de valda implementerade lösningarna åt AI Datorer AB där MERN-stacken inom webbutveckling kommer att användas.

## MVC-arkitekturen

MVC-arkitekturen eller *Model-view-controller*-arkitekturen är ett mjukvarudesignmönster för att ta fram användargränssnitt där logiken är fördelade i olika delar: en modell (eng. *model*) för vad för data som är tillgänglig, kontrollers (eng. *controllers*) som bestämmer vilka data som ska visas för vilka användare och vyer (eng. *views*) vilket är där dessa modellvalda data sedan visas(2).

Ursprungligen användes MVC-arkitekturen först i skrivbordsapplikationer men efter att internet slog igenom runtom i världen så blev det även vanligt där för webbaserade applikationer(2, *Use in web applications*). Exempel på webbaserade applikationer som använder sig av MVC-arkitekturen är bland annat AngularJS, VueJS, och Laravel(3).

Modellen i (2, *Model*).

(2, *View*).

(2, *Controller*).

## MERN-stacken

a.

### MongoDB

a

### ExpressJS

a

### ReactJS

a

### NodeJS

a

## TailwindCSS

a

## REST API

a

# Metod

Följande kapitel redogör för vilka verktyg som har använts i webbprojektet.

## Integrerad utvecklingsmiljö

I webbprojektet har *Microsoft Visual Studio Code* (VSCode) i *Windows 10 Pro Eng* använts som integrerad utvecklingsmiljö för att skriva all HTML-, CSS- & JS-kod. Hämtat här: <https://code.visualstudio.com/>.

I det inlämnade GitHub-repot för hela webbprojektet (både klient & server i ett och samma GitHub-repo) - tillgängligt här <https://github.com/WebbkodsLarlingen/dt162g-projekt-maka2207> - så har följande npm-paket och deras rekommenderade versioner använts:

* "bcrypt": "^5.1.1"
* "cookie-parser": "^1.4.6"
* "cors": "^2.8.5"
* "dotenv": "^16.3.1"
* "express": "^4.18.2"
* "jsonwebtoken": "^9.0.2"
* "mongodb": "^6.3.0"
* "nodemon": "^3.0.1"
* "react": "^18.2.0"
* "react-dom": "^18.2.0"
* "react-scripts": "5.0.1"
* "tailwindcss": "^3.3.6"

De första åtta paketen gäller för serverdelen (NodeJS, ExpressJS, MongoDB) medan de fyra resterande gäller för klientdelen (ReactJS, TailwindCSS).

Beträffande databasen MongoDB så har användargränssnittet [MongoDB Compass Version 1.41.0](https://www.mongodb.com/try/download/compass) använts för att kunna lättare felsöka databasens användning, medan [MongoDB Community Server version 7.0](https://www.mongodb.com/try/download/community) har använts för själva lokala MongoDB-databasservern - med anslutningsadress: mongodb://localhost:27017.

## Publicering

Publicering förekommer endast i den utsträckningen att [GitHub-repot](https://github.com/WebbkodsLarlingen/dt162g-projekt-maka2207) laddas ned och körs lokalt förutsatt att rätt MongoDB Community Server version 7.0 finns installerat och att det då kan anslutas till anslutningsadressen: mongodb://localhost:27017.

# Konstruktion

Detta kapitel avhandlar det faktiska praktiska arbetet i hur saker och ting har lösts samt varför de har lösts på de sätt de har lösts på.

## MongoDB+REST API

a

## NodeJS+ExpressJS

a

## ReactJS+TailwindCSS

a

# Resultat

I detta näst sista kapitel innan slutsatser redogörs AI Datorer AB:s färdiga webbaserade intranät samt dess funktionaliteter. Först redogörs hur det ser ut att registrera sig, logga in och även logga ut. Sedan redovisas hur det går till att hantera datorkomponenter i form av att utläsa, skapa nya, ändra och/eller radera befintliga datorkomponenter. Till sist visas det upp hur det går till för den utnämnda Systemadministratören att hantera användarna genom utläsning, skapande, ändrande och radering av användare.

## Registrering, inloggning och utloggning

a.

## Hantera datorkomponenter

a.

## Hantera användare

a.

# Slutsatser

a.

Källförteckning

1. B. Dayley. *Node.js, MongoDB and Angular Web Development (Second Edition)*. USA: Pearson Education Inc. 2018
2. Wikipedia, "Model-view-controller", <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller> Ändrad senast: 2023-12-22 Hämtad: 2023-12-31
3. Wikipedia, "Comparison of server-side web frameworks", [https://en.wikipedia.org/wiki/Comparison\_of\_server-side\_web\_frameworks#Comparison\_of\_features](https://en.wikipedia.org/wiki/Comparison_of_server-side_web_frameworks%23Comparison_of_features) Ändrad senast: 2023-12-29 Hämtad: 2023-12-31
4. Wikipedia, "Node.js", <https://en.wikipedia.org/wiki/Node.js> Ändrad senast: 2023-12-30 Hämtad: 2023-12-31
5. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
6. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
7. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
8. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
9. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
10. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
11. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
12. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
13. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
14. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
15. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
16. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
17. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
18. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
19. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
20. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
21. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
22. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
23. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
24. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
25. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
26. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
27. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
28. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
29. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
30. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
31. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
32. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
33. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31
34. Wikipedia, "", Ändrad senast: Hämtad: 2023-12-31