



# TDP003 Projekt: Egna datormiljön

## systemdokumentation

Författare

Jessk378, Nikni292



Höstterminen 2015  
Version 0.5

2015-10-06

## 1 Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
0.1	skapade dokument för skrivning	151006
0.2	La in sekvensdiagram och översikts bild	151008
1.0	Färdigställde information om presentationslagret	151008

# Innehåll

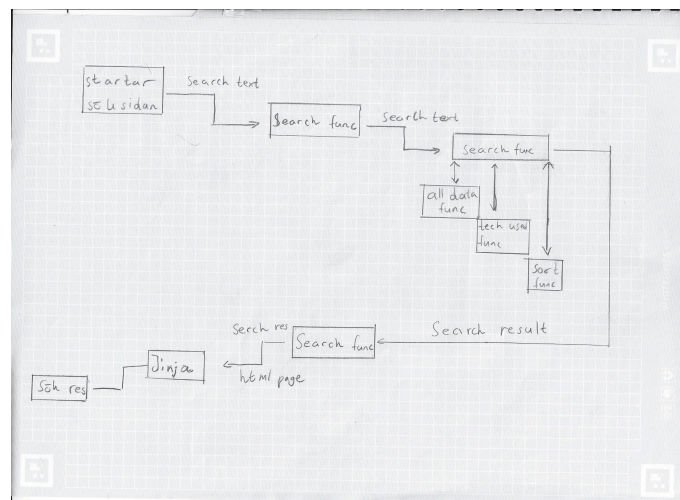
<b>1</b>	<b>Revisionshistorik</b>	<b>1</b>
<b>2</b>	<b>Introduktion</b>	<b>3</b>
<b>3</b>	<b>Sekvensdiagram</b>	<b>3</b>
<b>4</b>	<b>Övergripande bild</b>	<b>4</b>
<b>5</b>	<b>Datalagret</b>	<b>5</b>
5.1	Kursen datalager specifikationer . . . . .	5
5.1.1	Load-funktionen . . . . .	5
5.1.2	Get-project-count-funktionen . . . . .	5
5.1.3	Get-project-funktionen . . . . .	5
5.1.4	Get-techniques-funktionen . . . . .	5
5.1.5	Get-technique-stats-funktionen . . . . .	5
5.1.6	Search-funktionen . . . . .	6
5.2	Övergripande bild av Funktioner . . . . .	6
5.2.1	Load-funktionen . . . . .	6
5.2.2	Get-project-count-funktionen . . . . .	6
5.2.3	Get-project-funktionen . . . . .	6
5.2.4	Get-techniques-funktionen . . . . .	6
5.2.5	Get-technique-stats-funktionen . . . . .	6
5.2.6	Search-funktionen . . . . .	6
5.2.7	Get-all-data-funktionen . . . . .	7
5.2.8	Techs-used-funktionen . . . . .	7
5.2.9	Sort-data-funktionen . . . . .	7
5.2.10	Get-sort-order-funktionen . . . . .	7
5.3	Detaljerad beskrivning av funktionerna . . . . .	8
5.3.1	Load-funktionen . . . . .	8
5.3.2	Get-project-count-funktionen . . . . .	8
5.3.3	Get-project-funktionen . . . . .	8
5.3.4	Get-techniques-funktionen . . . . .	8
5.3.5	Get-techniques-stats-funktionen . . . . .	8
5.3.6	Search-funktionen . . . . .	8
5.3.7	Get-all-data-funktionen . . . . .	8
5.3.8	Techs-used-funktionen . . . . .	9
5.3.9	Sort-data-funktionen . . . . .	9
5.3.10	Sort-order-funktionen . . . . .	9
<b>6</b>	<b>Ramverk / Flask</b>	<b>10</b>
6.1	Funktioner . . . . .	10
6.1.1	Data och Config - load funktionerna . . . . .	10
6.1.2	Index-funktionen . . . . .	10
6.1.3	Projects-funktionen . . . . .	10
6.1.4	Project-funktionen . . . . .	10
6.1.5	Techniques-funktionen . . . . .	10
6.1.6	Search-funktionen . . . . .	10
<b>7</b>	<b>Felhantering och loggning</b>	<b>10</b>

## 2 Introduktion

I detta dokument kommer ni ta del av hur våran kod är tänkt att fungera. Ni kommer se sekvensdiagram över hur våra funktioner fungerar med varandra, samt bilder med förklaringar på specifika kodstycken.

## 3 Sekvensdiagram

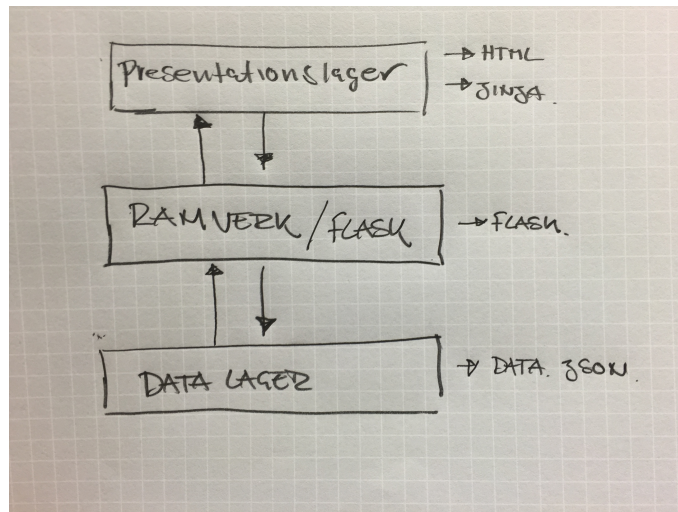
Användaren står på söksidan och gör en sökning. Sök strängen skickas till flask som då kallar på sök funktionen i datalagret. Den utförs i tre steg, först hittas alla projekt som innehåller strängen, den kollar sedan om det finns angivna tekniker om det finns filtrerar den ut alla projekt som inte har med teknikerna, det sista sök funktionen gör är att sortera datat utifrån vad som angetts. Den skickar sedan tillbaka den sökta datan till flask som kallar på jinja att ladda in resultat i sök hmtl sidan.



Figur 1: Sekvensdiagram över searchfunktionen tillsammans med presentationslager

## 4 Övergripande bild

Bilden nedan visar simpelt de tre lager vi arbetar med. Presentationslagret är delen som visar html sidor. De laddas med hjälp av jinja. Jinja blir tilldelad sidorna av Flask som ligger i ramverket. Innehållet i htmlsidorna läses från en .json fil. Den filen läses in i datalagret. Datalagret tar även hand om sökningar i data.json filen. Flask kommer kalla på specifika delar av .json filen beroende på vad användaren skickar för förfrågan. Flask returnerar sedan html sidan som ska laddas tillsammans med informationen som ska fylla sidan.



Figur 2: Övergripande bild på hur datalager, ramverk och presentationlager kommunicerar med varandra

## 5 Datalagret

Här nedan kommer det finnas förklaring på funktioner som vi implemterat i vårt datalager. Här finns även sekvensdiagrammet. Det kommer till en början vara översiktligt och sedan gå mer in på djupet av varje funktion och hur de fungerar tillsammans med andra funktioner.

### 5.1 Kursen datalager specifikationer

I denna sektionen finns kursens krav på vad de olika funktionerna i datalagret ska ha. För varje funktioner kommer det finnas en koppling mellan funktionen vi skrivit och kursenskrav.

#### 5.1.1 Load-funktionen

Kursenskrav: Ladda en .json fil och returnera en lista med alla projekt. Load ska ladda från en fil med UTF-8 teckenkodning. Vid error ska funktionen returnera "None".

Parameterar: filnamn, en sträng som innehåller namnet på json filen vi vill ladda.

Returnerar: en lista, listan ska inne hålle alla project från json filen eller "None"

Vår funktion: 5.2.1 Load-funktionen.

#### 5.1.2 Get-project-count-funktionen

Kursenskrav: Få antalet project i en lista av projekt.

Parametrar: En databaslista som man får ifrån Load-funktionen.

Returnerar: Antalet project i listan.

Vår funktion: 5.2.2 Get-project-count-funktionen.

#### 5.1.3 Get-project-funktionen

Kursenskrav: Hämtar ett speciferat project från databaslistan utifrån angivet ID. Om project inte finns returneras "None"

Parametrar: En databaslista som man får ifrån Load-funktionen.

Projekt id:et, ett nummber som anger vilket projekt som ska hämtas

Returnerar: En tabell med all data från ett projekt eller "None".

Vår funktion: 5.2.3 Get-project-funktionen.

#### 5.1.4 Get-techniques-funktionen

Kursenskrav: Hämtar en lista med alla tekniker från våra projekt, de ska vara i alfabetisk ordning.

Parametrar: En databaslista som man får ifrån Load-funktionen.

Returnerar: En alfabetiskt sorterad lista med namnen på alla tekniker i databaslistan.

Vår funktion: 5.2.4 Get-techniques-funktionen.

#### 5.1.5 Get-technique-stats-funktionen

Kursenskrav: hämta och returnerar statistik från alla tekniker speciferat i databaslistan. Nyckeln i tabellen är namnet på tekniken och nycklens värde är en lista med en tabell inuti. Den tabellen ska ha "id" som första nyckel med projektets "id" som värde. Den andra nyckeln är "name" den har projekets namn som värde. Id:ets och namnet värde ges av att tekniken ingår i det specifika projektet vi tittar på. I tabellen med id:et och namnen ska vara sorterade efter id:ets värde.

Parametrar: En databaslista som man får ifrån Load-funktionen.

Returnerar: En tabell med teknikernas statistik (se ovan).

Vår funktion: 5.2.5 Get-technique-stats-funktionen.

### 5.1.6 Search-funktionen

Kursenskrav: Hämtar och sorterar projekten som matchar sökningen som angetts.

Parametrar: En databaslista som man får ifrån Load-funktionen.

En variabel "sort\_by", namnet på fältet som funktionen ska sortera efter. En variabel

En variabel "sort\_order", anger om sorteringen ska ske fallande eller stigande.

En lista av tekniker som det returnerade projektet måste ha med. En tom lista betyder att fältet ska ignoreras.

En sträng variabel "search", en fri text sökning.

En lista, "search\_fields", fältet som vi vill att fri text sökning ska ske i. Om den är tom returneras inget. Om den är "None" ska alla fält sökas.

Returnerar: En lista med tabeller som innehåller alla projekt som matchade sökningen.

Vår funktion: ?? Search-funktionen.

## 5.2 Övergripande bild av Funktioner

Den här delen beskriver översiktligt hur funktionerna fungerar även hur de hänger ihop med andra funktioner.

### 5.2.1 Load-funktionen

Den här funktionen laddar in all information om projekten som vi kommer ha i vår portfolio. Den hämtar projekten ur en .json fil. Den returnerar sedan en lista med varje projekt som ett element i listan. I de listan elementen ligger projektet informationen i form av en tabell. In värdet i denna funktionen är .json filens namn.

### 5.2.2 Get-project-count-funktionen

Funktionen hämtar alla projekt i vår databas och räknar antalet projekt som vi har i databasen. Den returnerar sedan antalet projekt i databasen. In värdet i denna funktion är databas filen som load-funktionen skapade.

### 5.2.3 Get-project-funktionen

Denna funktion får ett id på projektet vi vill ha och hela databasen. Vi returnerar sedan det projektet som matchar emot id:et vi fick in. Om id:et inte finns i något projekt returneras "None".

### 5.2.4 Get-techniques-funktionen

Den här funktionen tar hela databasen som in värde. Den iterar sedan över hela databasen och hämtar alla tekniker som användts i alla projekt. Den returnerar sedan en sorterad lista över alla tekniker som varit med i projekten.

### 5.2.5 Get-technique-stats-funktionen

Funktionen har databasen som in värde. Den anropar sedan Get-techniques-funktionen och sparar teknikerna. Den går sedan igenom databasen för att se om specifika projekt innehåller tekniker vi fick av Get-techniques-funktionen. Om projektet innehåller en teknik lägger vi till projektet i en lista i form av en tabell med namn och id som nycklar och projektets namn och id som värde. Vi returnerar sedan listan med tabellerna i sig.

### 5.2.6 Search-funktionen

Search funktionen har sex stycken in värden, databasen, sort\_by (värdet vi väljer att sortera efter), sort\_order (descending eller ascending beroende på om vi vill ha störst först eller inte när vi sorterar), techniques (om vi vill sortera efter specifika tekniker), search (ordet eller orden vi vill söka efter), search\_fields (vilka fält i

databasen som vi vill söka i). Det först search funktionen gör är att anropa get-all-data-funktionen för att få vilka projekt som innehåller det vi söker efter. Vi kollar sedan om anroparen vill sortera projekten efter specificerade projekt detta gör den genom att anropa techs-used-funktionen. Sist sorterar Search-funktionen utifrån sort\_by och sort\_order detta genom att anropa sort-data-funktionen.

#### 5.2.7 Get-all-data-funktionen

Invärdet i den här funktionen är databasen, search och search\_fields. Hela den här funktionens syfte är att hämta projekten som passar sökningen som användare angett. Funktionen börjar med att kolla om search och search\_fields är "None", är dom "None" returneras hela databasen. Funktionen fortsätter om förra påståendet inte stämmer. Den kollar då om search\_fields är "None" är den "None" söker vi efter search i hela databasen, funktionen returnerar projekten vi hittade med search. Vi kollar sedan om Search\_fields är tom om den är tom returner funktionen "None". Sist går funktionen genom när användare angett både search\_fields och search. Då kollar vi i alla projekt som innehåller search\_fields och om projektet innehåller det fältet går vi in i det fältet och kollar om fältet innehåller search värdet. Stämmer det returnerar vi alla projekt som har det fältet och den sökningen.

#### 5.2.8 Techs-used-funktionen

Funktionen får invärdena search (de projekten som sökningen gav oss) och techs (vilka tekniker som ska ingå i projekten). Funktionen kollar om angivna tekniker finns med i projekten som search gav oss. Den returnerar sedan projekten som innehöll de projekten som stämde överens med angivna tekniker.

#### 5.2.9 Sort-data-funktionen

Den här funktionen tar in datat vi fick i search-funktionen, sort\_by och sort\_order. Funktionen sorterar sedan efter värdet angett i sort\_by. I funktionen anropas även get-sort-order, den bestämmer om det ska vara fallande eller stigande sortering.

#### 5.2.10 Get-sort-order-funktionen

Invärdet är sort\_order och funktionen bestämmer om sorteringen ska ske i fallande eller stigande ordning.



## 5.3 Detaljerad beskrivning av funktionerna

En djupare nivå av hur funktionerna fungerar och hur de fungerar med varandra.

### 5.3.1 Load-funktionen

Vi använder angivet namn som vi fick som invärde i ett “with” block för att öppna filen. Eftersom filen vi läser in ifrån är en .json fil kan vi lätt läsa in filens innehåll med den inbyggda funktionen “json.load”. Vi returnerar sedan listan med databasen. Om det av någon anledning inte går att läsa filen returnerar vi “None” som databas.

### 5.3.2 Get-project-count-funktionen

Funktionen kollar längden av listan, altså antalet element som finns i listan. Funktionen returnerar sedan det numret vi fick fram.

### 5.3.3 Get-project-funktionen

Vi itererar igenom databasen och kollar specifikt på varje projekt. Vi jämför sedan projektets id med id:et som använder angett. Om id:et finns i något projekt returnerar vi projektet som det stämde överens med. Om angivet id inte finns i något projekt returnerar funktionen “None”.

### 5.3.4 Get-techniques-funktionen

Funktionen iterar över hela inladdade databasen, den letar sedan upp fältet “techniques-used”. För varje teknik i “techniques-used” lägger vi till den teknik i en lista om listan inte redan innehåller den tekniken. Vi sorterar sedan teknikerna i listan i bokstavsordning. Funktionen returnerar sedan den sorterade listan.

### 5.3.5 Get-techniques-stats-funktionen

Funktionen börjar med att anropa get-techniques och sparar undan alla använda tekniker som använts i databasen. Vi iterar sedan igenom sparade tekniker för att kolla på varje teknik en i taget. Vi skapar en dict med tekniken som nyckel. Vi iterar nu över varje projekt. Om tekniken är med i projektet lägger vi till projekts namn och id som värde för nyckeln. Innan vi börjar med nästa teknik sorterar vi värdena som tillhör varje nyckel efter id:ets storlek. Vi börjar sedan om med nästa teknik och köra samma process. När vi gått igenom alla tekniker returnerar vi listan med tabellen.

### 5.3.6 Search-funktionen

Alla invärden i vår search funktion har fördefinierade värden ifall inget värde skulle skickas med. Deras start värde är: sort\_by=“start\_date”, sort\_order=“desc”, techniques=“None”, search=“None”, search\_fields=“None”. Search anropar sedan get-all-datafunktionen som ger tillbaka en lista. Den lista skickar search in techs-used-funktionen för att se om om sökresultaten stämmer överens med teknikerna som angetts vid anropet. Detta sker bara så länge någon teknik har angetts annars hoppas detta steg över. Sista funktionaliteten som search har är att den skickar in den sökta och filtrerade databasen i sort-data-funktionen för att sortera sökresultatet enligt användarens specifikationer. Den sökta, filtrerade och sorterade listan returneras sedan.

### 5.3.7 Get-all-data-funktionen

Vi börjar funktionen med att titta ifall search är “None” OCH om search\_fields är “None”. Om detta stämmer skickar vi tillbaka hela databasen som sökresultat. Nästa koll är om search\_fields är “None”, om detta stämmer söker vi i alla fält genom att iterera över varje projekt för sig och kolla om search får några träffar i projektet. Om search får en träff läggs det träffade projektet till i en lista. Detta sker tills det inte finns fler projekt. När alla är genomsökta skickar vi tillbaka alla projekt som fick en sökträff. Härnäst kollar

vi om `search_fields` är tom, om detta stämmer returnerar vi "None" som sökresultat. Om ingen tidigare "if" sats stämmer vet vi att `search` och `search_fields` har ett värde. Så då itererar vi över insickad databas. Vi kollar sedan om sökresultat är en lista, en int eller en sträng. Om det är en sträng gör vi sökning och sökresultat till gemener. Nästa steg kollar om fältet matchar `search_fields` och att `search` matchar något i databasen. Om allt detta stämmer lägger vi till projektet vi hittade alla saker i en lista. När vi gått igenom alla projekt och kollat alla fält och sökvärden returnerar vi en lista av alla projekt som matchade sökningen.

### 5.3.8 Techs-used-funktionen

Vi börjar genom att itererar över sökvärdet som `search`-funktionen skickade in. Vi iterar sedan över alla tekniker i fall att vi vill sortera efter flera tekniker. Vi kollar sedan om teknikerna finns med i projekten. Om teknikerna är med lägger vi till projektet i en lista. Vi returnerar sedan den listan med filtrerade projekt.

### 5.3.9 Sort-data-funktionen

sorterar den sökda data mängden som `search` skickade genom att köra "sort" funktionen. Med nycklen som vi sorterar efter som värdet `sort_by` hade. Vi sorterar i ordning efter `reverse` eller inte beroende värdet som `sort-order`-funktionen returnerar. Om den returnerar `True` sorterar vi i fallande ordning om den returnerar `False` sorterar vi i fallande ordning.

### 5.3.10 Sort-order-funktionen

kollar om `sort_order` är "desc"(descending) om `sort_order` är "desc" returnerar funktionen `True`. Annars returnerar funktionen `False`.

## 6 Ramverk / Flask

Nedan finns funktionerna som vi skrivit i flask för att kunna ladda html sidorna med inbäddade python funktioner.

### 6.1 Funktioner

Detta är funktionerna som kallar på jinja och laddar in html sidor så de visas i vår portfolio.

#### 6.1.1 Data och Config - load funktionerna

I dessa funktioner har vi hårdkodat namnet på .json filerna. Config-load: Laddar en config fil med information som är i till exempel header och footer.

Data-load: Laddar databasen från en fil med information som vi sedan använder på våra hemsidor.

#### 6.1.2 Index-funktionen

Denna funktion läser in senaste projektet samt hämtar information via config-load funktionen, för att generera statisk information som finns på alla sidor. Detta kan vara till exempel information av skapare, kontakt information etc.

#### 6.1.3 Projects-funktionen

Html sidan som den här funktionen skapar är till för att visa alla projekt i form av en lista. Denna funktion läser in alla projekt från databasen och skickar de vidare till presentationslagret där en html sida skapas. Mallen som används för att visa projekten är den samma som används i search-funktionen.

#### 6.1.4 Project-funktionen

Html sidan som den här funktionen skapar är till för att visa ett specifikt projekt beroende på vilket id. Funktionen kommer åt id:et via urlen. Denna funktion hämtar information om ett specifikt projekt ur databasen och skickar det till presentationslagret.

#### 6.1.5 Techniques-funktionen

Html sidan som den här funktionen skapar är till för att visa alla tekniker och projekt som använder den specifika tekniken. Denna funktion hämtar alla tekniker och de projekt som använder en viss teknik och skickar detta till presentationslagret. Användarna kan klicka på ett specifikt projekt och komma till project-funktionen.

#### 6.1.6 Search-funktionen

Html sidan som den här funktionen skapar är till för att visa en sökruta och klickbara check boxar. Man kan skriva en söksträng i rutan och filtrera projekt som har specifika tekniker. Sökresultat visas sedan under checkboxarna. Denna funktion skiljer sig från de andra då den hanterar både GET och POST, i det första skedet (GET) så visas ett formulär för användaren där man kan skriva in ett sökord och klicka i tekniker. När användaren söker på ett ord så anropas funktionen med POST och då hämtas träffande projekt ur databasen. Dessa presenteras sedan för användaren med samma mall som i projects-funktionen.

## 7 Felhantering och loggning

Fin felhantering och logging kommer finnas beskriven i den här sektionen.