|  |
| --- |
|  |

|  |
| --- |
| **DSK2/DVK2: PROJ Underhåll** |
| Grupp 5 – KidControl |
| VT2012 |
| |  |  | | --- | --- | | Andreas Spagnoli | 850124-0611 | | Pritesh Patel | 881230-2951 | | Robin Gredin | 840512-0232 | | Simon Krait  David Rogers  Marko Vuolevi  Nicole Danielsson  Stefan Berglin  Alexander Paratzaklidis | 891207-1951  831112-0338  810505-0259  870728-0304  870727-0271  910602-5779 | |
|  |
|  |
|  |
|  |

Innehållsförteckning

[Vad är KidControl? 1](#_Toc325723320)

[Underhållsinformation 2](#_Toc325723321)

[Mjukvaruuppdateringar 3](#_Toc325723322)

[Hårdvaruunderhåll 4](#_Toc325723323)

[JavaScript 4](#_Toc325723324)

[Utbildning 5](#_Toc325723325)

[Administrativa roller 5](#_Toc325723326)

[Genomgång av källkod för felsökning och underhåll av systemet 7](#_Toc325723327)

[Struktur 7](#_Toc325723328)

[Administration 7](#_Toc325723329)

[Aktuell information 7](#_Toc325723330)

[Blogg 8](#_Toc325723331)

[Forum 8](#_Toc325723332)

[Matsedel 8](#_Toc325723333)

[Närvaro 9](#_Toc325723334)

[Statistik 9](#_Toc325723335)

[Login 9](#_Toc325723336)

[Databas 9](#_Toc325723337)

# Vad är KidControl?

KidControl är ett socialt medium där lärare och föräldrar kan kommunicera direkt eller indirekt med varandra genom ett interaktivt forum och bloggar, där det mesta av förskolans information kan nås.   
  
I dagens samhälle dyker det upp allt fler smarta telefoner, surfplattor och datorer. Detta innebär att även infrastrukturen i samhället ständigt kommer att behöva växa, vilket innefattar även landets förskolor varav många har förlegade tankesätt eller system. Här ser vi en enorm potential att slå samman några problem till lösningar som både föräldrar och personal skulle behöva idag.    
  
Tanken är att KidControl som innehåller en hel del andra funktioner än dessa förlegade system ska överta mycket av det manuella pappersarbetet eller liknande som idag skapar en del problem och missförstånd på många förskolor. Med KidControl har föräldrar tillgång till sina barns vardag genom bloggar och forum. Med dessa kan de ta del av de aktiviteter deras barn varit med om under dagen och även får information som kan tänkas påverka deras barns vardag.   
  
I dagsläget finns det en hel del föräldrar som önskar sig en större förståelse och inblick i vad som försiggår på förskolan under dagarna. KidControl bidrar med en lösning till detta, genom att tillhandahålla en gemensam plats för kommunikation och information.

# Underhållsinformation

Efter KidControl har implementeras i en förskolas system måste det även underhållas för att kunna hålla det uppdaterat och redo att stå emot de hot som förekommer på Internet.

Hur underhållet går till är beror helt på hur det är bestämt att lägga upp tjänsten. Systemet kan användas både som ett centraliserat system och som ett decentraliserat system där varje förskola har sin egen server.  En centraliserad server för flera förskolor blir enklare att underhålla eftersom det innebär att en eller flera tekniker kan jobba med servern. Här slipper kunden det arbete som rör det tekniska i systemet. Dock så kan en del kostnader bli relativt höga samt att en del administrativa uppgifter kan ta något längre tid att genomföra. En decentraliserad server innebär att systemet skulle behöva installeras och administreras lokalt vilket har sina fördelar då kunden får större kontroll av systemet och att en del administrativa uppgifter kan genomföras direkt på plats. Den decentraliserade varianten kan även innebära minskade kostnader för kunden då det är upp till varje förskola att anordna sina resurser. Nackdelen är att kunden blir ansvarig för alla säkerhetsåtgärder såsom backup, lagar och restriktioner.  
  
En användare är kopplad till en specifik förskola och har därmed användaruppgifter direkt relaterade med den aktuella förskolans system. Detta upplägg innebär att en användare bara har tillgång till information om sina egna barn om de går på en förskola som använder sig av KidControl. Ur säkerhetssynpunkt är detta viktigt men underlättar också underhållet av systemet eftersom det finns naturliga avgränsningar för användare. Det medför också att en användare som har barn vid mer än en förskola kommer ha mer än ett konto, en för varje förskola. Detta blir en nackdel för användaren i och med att det blir mer information att hålla reda på. Det kan också ses som en fördel i och med att användaren får en väldigt distinkt separering mellan de olika förskolorna.  
  
Systemet använder sig av en vältestad och välanvänd mjukvara såsom MySQL för databashantering och Apache som webbserver. Denna blandning av olika servertekniker gör att det inte är helt enkelt att få igång det faktiska systemet. Det krävs även en del tekniskt kunnande för att installera alla dessa på en maskin och få dem att fungera tillsammans. Förutom MySQL och Apache måste även PHP installeras, samt någon typ av DNS-hanterare.  
  
Det kommer inte krävas särskilt mycket underhåll av databasen då den är uppbyggd för att automatiskt rensa ut gammal information vid fastställda tidpunkter. Apacheservern kommer heller inte kräva mycket underhåll. Det enda riktiga hotet skulle vara om det skedde en hårdvarukrasch eller någon sorts riktad attack mot systemet. Tanken är att systemet skall komma förinställt med en grundläggande design och GUI som sedan skall gå att ändra om enkelt om det skulle behövs eller önskas. Förinställt i detta fall innebär att det skall finnas färdiga inställningar för webbservern och PHP som kan användas samt också att en katalogstruktur för systemets webbsida skall stå klar. För att vidare förenkla användandet av systemet är det tänkbart med en automatiserad installationsprocess. Med denna lösning skulle exempelvis ett paket med MySQL, Apache, PHP och så vidare installeras automatiskt med fördefinierade inställningar, detta för att förenkla så mycket som möjligt för produktägare och kunder.  
  
System är i sig väldigt enkelt att underhålla och kräver inte några speciella avancerade kunskaper. Det kräver dock lite eftertanke eftersom de funktioner som finns för underhåll har direkt påverkan på databasen. Med detta i åtanke användas separata funktioner för administratörskonton. Tanken är att varje förskola skall minst ha en och maximalt ett litet fåtal konton för administratörsfunktionerna. Det är tänkt att det skall finnas utförliga och enkla manualer för underhållet av systemet. Dessa bör innehålla ordentliga beskrivningar av samtliga funktioner och tydligt beskriva vad konsekvenserna av varje funktion är.  
  
Teknisk kunnig personal behövs främst vid installationen av system och i de fall någonting skulle gå väldigt fel. Dessa fel är bland annat om någon hårdvara skulle gå sönder eller något extremt med mjukvaran skulle gå fel som skulle kräva ominstallation av hela eller delar av systemet. Man kan också tänka sig att riktade attacker mot systemet skulle förstöra det så pass att en tekniker skulle behövas för att återställa både säkerheten och funktionen i systemet.  
  
Databasunderhållet ligger då direkt på den eller de personerna med tillgång till administratörskontot. Det är tänkt att använda flera triggers i databasen för att rensa ut gammal data men detta har dock inte kunnat implementeras och testas än då fullständiga rättigheter på den server som använts saknats på den använda databasen. Det är fullt möjligt att rensa ut gammal data manuellt men med tanke på hur mycket data det faktiskt kan bli är det inte speciellt effektivt. Det går även att enkelt göra så att de triggers som rensar databasen på gammal data även skapar backup på dessa. Om och var detta skall lagras bestämmas av de som styr förskolans verksamhet. Det kan också finnas lagar som styr lagring av denna typ av data, speciellt då barn är involverade.

## Mjukvaruuppdateringar

Systemet bygger på en vältestad mjukvara för servern. Mjukvaran är väldigt robust och konflikter uppstår sällan. De fel som uppstår är ofta inte så pass allvarliga att standardfunktioner slutar fungera. Systemet är till större del uppbyggd på grundläggande funktioner. Om fel uppstår i den nämnda mjukvaran skulle felen med största sannolikhet visa sig som felsorterad data eller andra mindre allvarliga fel.  
  
När man dock måste uppdatera mjukvaran behövs en del tekniskt kunnande. I vanliga fall uppdateras mjukvara inom närliggande versioner för att konfirmera att alla funktioner fortfarande fungerar. Även om det ofta är relativt ofarligt att uppdatera bör man inte göra det om det inte finns någon anledning till det. Säkerhetsuppdatering är oftast specifika och det är väldigt osannolikt att just de funktioner som finns i systemet störs. Vid alla uppdatering bör dock systemet testas noggrant innan man genomför den faktiska uppdateringen. Detta är speciellt viktigt när man byter mellan olika versioner. Då finns risken att vissa funktioner faktiskt försvinner helt och man får därför testa grundligt innan man kan tänka sig att genomföra dessa uppdateringar. Det kan också finnas starka skäl att ändra koden till att använda nyare funktioner för att optimera systemet.  
  
Uppdateringarna av servermjukvaran går i allmänhet enkelt att utföra då all mjukvara som används är testad. Detta betyder att sättet att uppdatera också har blivit testat vilket gör det enkelt att byta mellan olika versioner. Man kan i allmänhet fortsätta att använda samma filer för inställningar och använda samma katalogstruktur. I exempelvis fallet med PHP sparas .ini-filen för inställningar och sedan installeras nya versioner som ersätter den gamla .ini filen.

## Hårdvaruunderhåll

Olika hårdvaror används beroende på om systemet är centraliserat eller decentraliserat. Om en decentraliserad lösning av systemet körs räcker det i princip med en vanlig hemmadator, det enda som krävs är en uppkoppling, tillförlitlig strömförsörjning och att servern får stå orörd dygnet runt, året runt. Sedan bör man även ha en backuplösning som skyddar all information i databasen. Då det bara är en server eller databas som skall skyddas behöver inte backuplösningen vara särskilt avancerad, det räcker i princip med någon sorts hårdvarubackup såsom en RAID-lösning eller molnlösning. Även en kombination av dessa två skulle vara en effektiv lösning. En annan lösning skulle vara att samtliga backuper från samtliga förskolor sparas på en central server som administreras av en ytterligare instans det vill säga en tredje part, även med denna lösning rekommenderas såklart att en lokal backup utförs.  
  
Däremot med en centraliserad lösning krävs det ordentlig tilltagen hårdvara, uppkoppling med hög bandbredd som kan hantera ett flertal användare uppkopplade samtidigt. Detta betyder att servern antingen kommer behöva stå på ett professionellt server-hotell eller att kunden själv har ordentliga serverhallar. Denna lösning innebär en mycket mer strikt och säker backup. Dock behöver lagringen av backupen vara någorlunda säker på grund av att det kommer finnas information om väldigt många olika förskolor på den. Ett annat tankesätt är att sprida ut backups och spara dem på olika geografiska platser.

## JavaScript

Då systemet kommer att användas av både föräldrar och personal är det väldigt viktigt att det fungerar med så många webbläsare som möjligt ute på marknaden. Det är något som har genomsyrat tankesättet under hela utvecklingsprocessen. De flesta webbläsare har JavaScript aktiverat när de används hemifrån, men då föräldrarna lämnat sina barn på förskolan och troligtvis befinner sig på ett arbete under denna tid är det troligt att de använder sig av någon annat medium än de vanligtvis använder hemma. Dessa andra medium kan vara en dator på en arbetsplats eller en surfplatta av något slag. Det är inte alltid dessa medium har stöd för JavaScript eller har det aktiverat. Risken finns därför att det uppstår en situation där en förälder inte kan använda systemet och potentiellt inte kan leverera viktig information till förskolan eller få reda på viktig information om ens eget barn.  
  
KidControl är därför utvecklat för att kunna fungera med minsta möjliga förutsättningar vilket bidrar till att det totala underhållet kommer minska i längden. Genom att produkten skall gå att nås via samtliga medier med internetåtkomst och alla möjliga webbläsare så slipper man en hel del underhåll. Om en användare brukar en webbläsare som inte har JavaScript funktionalitet aktiverat kommer denne inte mötas av några felmeddelanden, detta kommer att bidra till mindre supportärenden från användarna och istället kan tiden läggas på att förbättra det befintliga systemet.

## Utbildning

Då systemet kommer att förbättras med tiden och fler funktioner är tänkta att tillkomma i framtiden så behövs ständig användaråterkoppling och utbildning. Detta skulle kunna ses som ett slags underhåll då det hjälper produktägarna att skapa en mer slagkraftig produkt som även förnöjer användarnas upplevelser. Med jämna mellanrum, förslagsvis årligen eller varje halvår, så bör det skickas ut frågor till användaren angående systemet. Detta för att underlätta utbildningsprocessen av systemet.   
  
Då administratörerna har ett väldigt stort ansvar så krävs det att dessa vet vad de gör. Därför kommer dessa att få träffa representanter från KidControl som lär dem att använda systemet på ett korrekt och effektivt sätt. I och med att de klarar av att använda systemet korrekt kommer de bidra med att underhålla systemet på en lokal nivå. Det medför att enklare problem kommer kunna lösas på plats av administratörerna istället för att ta hjälp utifrån.

## Administrativa roller

Då det finns en hel del uppgifter som kommer behövas administreras så är det rekommenderat att en del personer med specifika roller tillsätts.   
  
Administratörsrollen är en väsentlig del för att produkten skall fungera. Denna har en del av de mest grundläggande och viktiga uppgifterna. Den skall se till att man kan lägga till, redigera samt ta bort information hos den specifika förskolan vilket gör rollen till väldigt central. Utan en aktiv och ansvarig administratör så faller stora delar av systemets värde då användare inte kommer kunna ha tillgång till systemet och att informationen inte är passande.   
  
För att underlätta för förskolorna så rekommenderas det att vidare dela in administratörsrollen i mindre delar eller att ha mer än en aktiv administratör. En moderator skulle kunna ha ansvaret för att se över informationen som läggs till i systemets olika delar. Detta för att exempelvis se till så att inget olämpligt skrivs, och vidare ifall det har förekommit olämpligt innehåll kunna ta bort detta.  
  
En annan roll skulle vara den rent administrativa rollen för att kunna lägga till, ta bort och redigera användare i systemet. Genom att dela upp arbetet så blir det mindre och lättare att utföra. Det skall inte vara nödvändigt att behöva anställa ytterligare en person för att underhålla systemet, utan det skall vara möjligt att vara administratör vid sidan av sitt läraryrke.  
  
Administratören har en egen vy för att lättare kunna underhålla vissa delar i systemet. Dessa ligger under ett menyalternativ vid namn Administation. Där kan man enkelt lägga till, ta bort och redigera användare av olika slag.   
  
De personer som innehar dessa roller bör ha lite mer ingående insikt om vad de faktiska funktionerna gör. Dessa funktioner har direkt påverkan på databasen och om det blir helt fel kan det vara svårt att ändra det. Det är rekommenderat att personen som fyller dessa roller skall veta precis vad det är de vill göra och inte gör något impulsivt utan att ha en aning om den gjort.

# Genomgång av källkod för felsökning och underhåll av systemet

## Struktur

Systemet är uppdelat i flera olika funktioner vilka också ligger i sin egen katalogstruktur på webbservern. Varje funktion har en egen huvudfil som ligger direkt i katalogen Pages och en katalog som också ligger under katalogen Pages. I allmänhet följs samma uppbyggnad över alla delar i systemet. Denna uppbyggnad är gjord på så sätt att det finns en central klass för funktioner som oftast döpt *funktion*Model.php samt flera filer med formulär, som är döpta *funktion*Form.php, som länkar till en fil för databaskoppling för just det formuläret som är döpt *funktion*Method.php.

## Administration

Administrationsvyn är olika beroende på vilken rättighetsnivå användaren har. Filen heter *person.php*. Är användaren inloggad som administratör ser denne knappar som leder till funktioner för att lägga till, editera och ta bort barn, målsmän eller anställda från systemet. Dessa funktioner går igenom ett antal formulär som underlättar valet av rätt person att ändra för att sedan leda till en metod som i sin tur gör ändringarna som krävs i databasen. Alla knappar går via en fil som heter *personFunction.php* som sedan dirigerar vidare till rätt formulär eller funktion. Det finns en fil som heter *personModel.php* som har funktioner för att hämta data från databasen. Det finns också två olika objekt till stöd för denna by. Dessa är *childObject.php* för att få all data kopplad till ett barn samt *relativeObject.php* för all data som är kopplad till målsmän.

## Aktuell information

Aktuella informationens centrala vy ligger i *currentInfo.php* under pages och är den vyn där en hel månads informationsinlägg syns. Denna vy verifierar rättighetsnivån och visar rätt knappar beroende på användaren rättigheter. Det som syns på framsidan ligger i *start.php* och genereras i den vyn. Där visas också länkar utefter rättighetsnivå. Bara de tre senaste inläggen visas där varje inlägg maximalt får sträcka sig 30 dagar bakåt i tiden.

Funktionerna för Aktuell Information ligger i *currentInfoModel.php* och sköter all hämtning av data från databasen. Alla formulär för att fylla i och redigera Aktuell Information finns i filer med *Form* i slutet av filnamnet. Dessa är länkade till filer med *Method* i slutet av sitt filnamn som i sin tur ändrar eller lägger till data i databasen. Det finns också en fil som heter *removeInfo.php* som är till för att helt ta bort ett inlägg från databasen.

## Blogg

Bloggen är uppbyggd i en central vy. Denna vy ligger i filen *Blogg.php* och är den som visar all data som skall visas. Det är i denna fil som det också bestämmer vad som skall visas för vem. De flesta funktioner för att hämta data från databasen ligger *bloggModel.php* och har funktioner som alla hämtar eller ändrar i databasen. Alla knappar som syns i bloggen går via en mellanklass som heter *bloggFunction.php* som då dirigerar till rätt formulär eller funktioner. De formulär som finns har olika namn och innehåller olika saker men filnamnet slutar alltid på Form som exempelvis addBloggForm.php. Dessa formulär länkar alla vidare till en fil som har filnamn som slutar med Method istället för Form som till exempel addBloggMethod.php. De sistnämnda filerna är de som gör ändringar i databasen. Det finns också en objektklass vid namn *bloggPostObject.php* som innehåller den data som varje bloggpost behöver ha tillgång till. Dessa data är; titel, text, författare, timestamp, en array med kommentarer samt ett antal Id-värden för att underlätta senare hantering mot databasen.

## Forum

Forum är uppbyggt i en central vy. Den vyn ligger i filen *Forum.php* och är den som faktiskt visar all data som skall visas. Till skillnad från bloggen kan samtliga se allt innehåll och är inte bundna till sina respektive avdelningar. Dock så finns det olika rättighetsnivåer som bestämmer vad respektive användare kan göra i forumet. De flesta funktioner för att hämta data från databasen ligger *forumModel.php* och har funktioner som alla hämtar eller ändrar i databasen. Alla knappar som syns i forumet går via en mellanklass som heter *forumFunction.php* som dirigerar användaren till rätt formulär eller funktion. De formulär som finns har olika namn men filnamnet slutar alltid på Form som till exempel newThreadForm.php. Dessa formulär länkar alla vidare till en fil som har filnamn som slutar med Method istället för Form så som newThreadMethod.php. De sistnämnda filerna är de som gör ändringar i databasen. Det finns också två stycken objektklasser som heter *categoryObject.php* som innehåller den data som varje forumkategori behöver ha tillgång till och *forumPostObject.php* som innehåller den data som varje forumpost behöver ha tillgång till. För *categoryObject.php* är dessa data titel, editerbar, beskrivning och eventuell koppling till föräldrarkategori. För f*orumPostObject.php* är dessa data titel, text, författare, timestamp och en eventuell koppling till ett föräldrainlägg, samt ett antal Id-värden för att underlätta senare hantering mot databasen.

## Matsedel

Huvudvyn i matsedeln är som i Aktuell Information den uttökade vyn och ligger i *menu.php*. Denna vy visar som vanligt bara de knappar för de som har rätt rättigheter. Vyn på huvudsidan ligger precis som i Aktuell Information i *start.php* och visar bara dagens meny. Det finns ett objekt för menyn som heter *menuObject.php* som innehåller den data som behövs. Detta objekt är avgränsat till att innehålla data för bara en typ av rätt och bara för en dag. Det finns också formulär och metodfiler för att lägga till och ändra i menyn.

## Närvaro

Närvaron är uppbyggd i en central vy. Denna vy ligger i filen *attendance.php* och är den som faktiskt visar all data som skall visas. Dock så är det två helt skilda vyer beroende på rättighetsnivå, ena vyn där en förälder kan ändra närvaro för sina barn och en vy där en anställd kan överblicka samtliga barna närvaro på respektive avdelning. De flesta funktioner för att hämta data från databasen ligger *attendanceModel.php* och har funktioner som alla hämtar eller ändrar i databasen. Nästan alla knappar som syns i forumet går via samma fil som heter *attendance.php* som då utför avsedd aktivitet. Under bägge vyerna finns möjlighet att bläddra mellan veckodagar och veckor.

## Statistik

Denna vy kan bara ses av användare inloggade på administratörskonton. I själva vyn, som heter *checkIP.php*, finns det två drop-down-menyer som är till för att bestämma ett tidsintervall för statistiken. Användarens IP adress loggas när man loggar in i systemet via *ClientIPLogger.php* som ligger i rotkatalogen för webbservern istället för under pages.

## Login

Loginvyn syns först när man kommer till hemsidan. Har användaren inte en aktiv session och försöker ta sig direkt till en specifik del av sidan blir denna direkt länkad till inloggningssidan som heter *login.php*. Logiken för att se om lösenordet stämmer ligger i *loginLogic.php* och själva lösenordet är sparat hashat i databasen.

## Databas

Det finns en wrapper klass för att hantera PHP-kopplingar mot databasen. Denna heter *database.php* och kan ligga utanför webbserverns publika katalogstruktur för extra säkerhet. Det är på den här som information om databasen ligger, så som adress och inloggningsuppgifter. Den har ett antal funktioner för att underlätta kopplingen mot databasen. Skillnaden mellan *query* och *megaQuery* funktionerna är att query öppnar och stänger kopplingen mot databasen om så behövs medan megaQuery inte gör detta. Det är enbart till för att kunna optimera kod.