Lauri Pihlajamäki

**Active Directoryn käyttäjien hallinta ASP.NET-sovelluksella**

Opinnäytetyö
Syksy 2016
SeAMK Tekniikka
Tietotekniikan tutkinto-ohjelma


ASP.NET on kehittyvä ja lisättävien kirjastojen avulla muokattavissa oleva alusta verkkosivujen tekoa varten. Sovelluskehyksen avulla voidaan luoda monimutkaisia web-sovelluksia yksinkertaisilla käyttöliittymillä.

Avainsanat: Active Directory, ADAM, versionhallinta, autentikointi, autorisointi, ASP.NET.
The aim of this thesis was to design and implement an ASP.NET web site that could retrieve a list of users from Active Directory and ADAM. The web site would also enable the resetting of a selected user’s password. Version control was used to store the process of development.

The first half of the thesis introduces the history of directory services and focuses on Microsoft’s Active Directory and ADAM. The thesis also studies the history of ASP.NET-development. Next the term and some examples of version control systems are explained. The thesis also describes the authentication and authorization within web sites. The final part of the thesis focuses on the development of the ASP.NET web application.

The ASP.NET web framework is still in development and continues to evolve with community made libraries. The ASP.NET framework makes it possible to produce complex programs with a simplistic user interface.

Keywords: Active Directory, ADAM, version control, authentication, authorization, ASP.NET.
SISÄLTÖ

Opinnäytetyön tiivistelmä........................................................................................................2
Thesis abstract ........................................................................................................................3
SISÄLTÖ......................................................................................................................................4
Kuvioluettelo.............................................................................................................................6
1 JOHDANTO ..........................................................................................................................8
   1.1 Työn tausta ....................................................................................................................8
   1.2 Työn tavoite ..................................................................................................................8
   1.3 Työn rakenne ...............................................................................................................8
   1.4 Tietohallinnon Helpdesk-palvelu Jelppari .................................................................9
2 HAKEMISTOPALVELUT ....................................................................................................10
   2.1 Active Directory ..........................................................................................................11
   2.2 AD-palvelun rakenne .................................................................................................11
   2.3 Nimeämiskäytännöt ....................................................................................................12
   2.4 ADAM ...........................................................................................................................14
3 ASP.NET ............................................................................................................................16
   3.1 Yleistä ..........................................................................................................................17
   3.2 ASP.NET-tuotantomallit .............................................................................................17
   3.3 ASP.NET-kontrollerit ....................................................................................................18
4 VERSIONHALLINTA ..........................................................................................................20
   4.1 Versionhallinnan vaihtoehdot ....................................................................................20
   4.2 Git .................................................................................................................................23
   4.3 TortoiseSVN................................................................................................................25
       4.3.1 TortoiseSVN-ohjelman ominaisuudet .................................................................27
5 AUTENTIKOINTI JA AUTORISOINTI .............................................................................29
   5.1 Autentikointitavat .......................................................................................................29
       5.1.1 Windows ...............................................................................................................29
       5.1.2 Forms ....................................................................................................................30
6 ASP.NET-SOVELLUKSEN TOTEUTUS ..........................................................................32
   6.1 Versionhallinta .............................................................................................................32
   6.2 ASP.NET-sovelluksen kuvaus ....................................................................................34
6.3 Active Directory -haun toteutus ................................................................. 37
6.4 Salasanan uusinta ....................................................................................... 41
6.5 Ennakoiva haku ......................................................................................... 42
6.6 Salasanan lähetys tekstiviestinä ................................................................. 43
7 YHTEENVETO JA POHDINTA ................................................................. 45
LÄHTEET ........................................................................................................ 46
LIITTEET ........................................................................................................ 49
Kuvioluettelo

Kuvio 1. Active Directoryn hierarkinen järjestelmä .................................................. 12
Kuvio 2. Yleisimmät objektien tyypit nimipalveluiden toteutuksessa .......................... 14
Kuvio 3. Esimerkki verkosta, jossa on käytössä ADAM ........................................... 15
Kuvio 4. MVC-malli .................................................................................................. 18
Kuvio 5. Paikallinen versionhallinta -diagrammi ..................................................... 21
Kuvio 6. Keskitetty versionhallinta -diagrammi ......................................................... 22
Kuvio 7. Hajautettu versionhallinta -diagrammi ....................................................... 23
Kuvio 8. Esimerkki kuinka muut versionhallintajärjestelmät näkevät tiedostot .. 24
Kuvio 9. Gitin tilannekuvien varastointi ajan kuluessa ......................................... 25
Kuvio 10. TortoiseSVN Windowsin resurssienhallinnassa ....................................... 26
Kuvio 11. GNU-lisenssi .......................................................................................... 27
Kuvio 12. TortoiseSVN-kuvakepäällykset ................................................................. 27
Kuvio 13. Forms-autentikoinnin toimintaperiaate ................................................... 31
Kuvio 14. TortoiseSVN-ohjelman asennus ja valinnaiset lisäosat ............................. 33
Kuvio 15. Sovelluksen versiohistoria ......................................................................... 34
Kuvio 16. Sovelluksen käyttöliittymä ....................................................................... 35
Kuvio 17. Testikäyttäjän tietojen tarkastelu ............................................................ 36
Kuvio 18. ASP.NET-mallipohjan valitseminen .......................................................... 37
Kuvio 19. ASP.NET Designer -sivu salasanan vaihdon tuloksesta ........................ 42
Kuvio 20. AutoCompleteExtender-kontrollerit ......................................................... 43
Käytetyt termit ja lyhenteet

**AD**  Active Directory. Hakemistopalvelu, joka varastoi tietoa verkon tietokoneista ja käyttäjistä.


**ADsPath**  AD-polut. Tapa viitata AD-palvelussa oleviin objekteihin LDAP-standardin mukaisesti.

**AJAX**  Asynchronous JavaScript And XML. Joukko web-sovelluskehityksen tekniikoita, joiden avulla sovelluksesta saadaan vuorovaikutteisempi.

**ASP.NET**  Active Server Pages.NET. Microsoftin Web-sovellusten luomiseen tarkoitettu ohjelmointimalli.

**DN**  Distinguished Name. AD:n nimeämiskäytäntö, jolla viitataan tiettyyn objektiin AD-palvelussa.

**HTML**  Hyper Text Markup Language. Internetsivujen luomisessa käytettävä koodikieli.


**Schema**  Objektia kuvaava kaava. Listaa objektin attribuutit ja tiedot.

**Versionhallinta**  Järjestelmä, jolla tallennetaan alkuperäinen ohjelma ja siihen tehdyt muutokset.
1 JOHDANTO

1.1 Työn tausta

Seinäjoen Ammattikorkeakoulun opiskelijat ja opettajat kirjautuvat koulun tietokoneisiin omilla tunnuksillaan. Tunnukksille on asetettu omat salasanat, jotka vanhenevat tietyn ajan kuluessa. Jos opiskelija ei vaihda salasanaansa sitä ennen, tunnuksset menevät lukkoon ja salasana on uusittava ATK-teen kautta.

Helpdesk-palvelu Jelppari on osa SeAMKin tietohallintoa. Jelppari hoitaa tietoteknisen avun lisäksi opiskelijoiden ja opettajien salasanojen uusimisen. Tähän prosessiin Jelppari kuluttaa huomattavan määrän aikaa, sillä opiskelijan ja tunnusten tiedot on todennettava kolmen erillisen ohjelman kautta ennen salasanan uusimista. Voisiko tästä prosessia millään nopeuttaa?

Tietohallinto on kehitänyt web-sovelluksen, jolla voitiin vaihtaa opiskelijoiden salasanoja, mutta siitä puuttui ominaisuuksia, joten Jelppari ei voi käyttää sitä.

1.2 Työn tavoite


1.3 Työn rakenne

Luvussa 2 kerrotaan nimipalveluista ja hakemistopalveluista, sekä Active Directorysta ja sen rakenteesta. Luvussa kerrotaan myös ADAM-palvelusta. Luvussa 3 kerrotaan ASP.NET-ohjelmistokehyksestä ja sen toimintamalleista. Neljännessä luvussa käydään läpi versionhallinnasta yleistä tietoa, kerrotaan sekä
Git-versionhallinnasta että tarkemmin työssä käytetystä TortoiseSVN-versionhallintaohjelmasta.


1.4 Tietohallinnon Helpdesk-palvelu Jelppari

2 HAKEMISTOPALVELUT


Nimipalveluiista puhuttaessa vastaan tulee käsitye "hakemistopalvelut". Useissa nimipalveluissa on ominaisuuksia, jotka mahdollistavat hakemistopalveluiden (directory services) käytön. Hakemistopalvelu käsittelee objekteja (object) niiden nimien perusteella. Hakemistopalvelu mahdollistaa myös tavan muokata objekteja niiden ominaisuuksien (attributes) avulla. Käytettäessä hakemistopalveluita verkon resurssseista puhutaan objekteina, ja palvelun avulla voidaan hakemistojen objekteja muokata. (Talvivaara [Viitattu 14.5.2016].)

**Objekti** tarkoittaa yleistä käsitettä niistä asioista, mistä hakemistopalveluun talletetaan tietoa. Objektilla voidaan tarkoittaa esimerkiksi tietokonetta, tulostinta, käyttäjää, jne. Objektin tiedot tallennetaan objektin ominaisuuksiin eli attribuutteihin. **Attribuutit** voivat tarkoittaa esim. käyttäjätunnusta, nimeä, puhelinnumeroa, käyttöoikeuksia tai tulostimesta puhuttaessa sen valmistajan tunnusta, tulostusnopeutta, käytettävän paperin kokoa jne. Attribuutit riippuvat käytössä olevasta hakemistopalvelusta, ja mitä tietoa objekteihin voidaan tallentaa. (Talvivaara [Viitattu 14.5.2016].)
Esimerkkejä hakemistopalveluista ovat esimerkiksi Unix-ympäristöissä käytetty **Network Information System (NIS)** ja Novell-ympäristöjen **Novell Directory Service (NDS)**. **NIS** on tietokoneverkkoja varten suunniteltu hakemistopalvelu, joka antaa verkkoon kytkeytyneille laitteille tarvittavat tiedot kyseisestä verkosta ja sen palveluista, kuten käyttäjistä, tulostimista ja muista verkoista. **NDS** on samankaltainen hakemistopalvelu, joka tarjoaa tietoa verkon tiedosto- ja tulostuspalveluista. (Talvivaara [Viitattu 14.5.2016].)

Muita hakemistopalveluita ovat Microsoftin **Active Directory** ja SUN Microsystemsin **SunONE Directory Server**. Kehittyvien hakemistopalveluiden avulla isonkin organisaation tiedot voidaan hallita sujuvasti ja tehokkaasti. (Talvivaara [Viitattu 14.5.2016].)

### 2.1 Active Directory


### 2.2 AD-palvelun rakenne


2.3 Nimeämiskäytännöt


ADsPaths. AD-polun kuvataan LDAP-standardien mukaisen kieliputun ja sääntöjen avulla. Esimerkiksi kuvion 1 microsoft.com-juuren AD-polku kuvataan seuraavasti:
LDAP://dc=microsoft,dc=com

Esimerkissä DC tarkoittaa verkoalueen nimen osaa (Domain Name Component). (Richards, Allen & Lowe-Norris 2006, 17.)

**DN.** DN-nimellä (Distinguished Name) viitataan tiettyyn objektiin hakemistopuussa (Directory Information Tree). Esimerkiksi *microsoft.com*-verkoalueen Users-säiliössä sijaitsevan järjestelmävalvojan tilin AD-polku kuvataan seuraavasti:

LDAP://cn=Administrator,cn=Users,dc=microsoft,dc=com

DN-nimessä LDAP–tunniste on poistettu, joten se kuvataan näin:

<cn=Administrator,cn=Users,dc=microsoft,dc=com>

**RDN.** Suhteellista DN-nimeä (RDN) käytetään viittamaan tiettyyn objektiin, joka sijaitsee kyseessä olevan isä-säiliön sisällä. Esimerkiksi edellisen DN-nimen RDN kuvataan näin:

<cn=Administrator>

**OU.** Nämä polut ovat yhdistelmiä etuliitteistä ja nimistä, jotka on erotettu yhtä-suuruus-merkeillä (=). Active Directoryssä yleinen Organizational Unit (OU) ilmaistaan AD-polussa seuraavalla tavalla:

<cn=Matti Virtanen,ou=Revontuli IT Oy,dc=microsoft,dc=com>

Kuvio 2. Yleisimmät objektien tyypit nimipalveluiden toteutuksessa (Wahl, Kille & Howes 1997.)

2.4 ADAM

**ADAM** (Active Directory Application Mode) on kevennetty versio Active Directorysta. ADAM-ohjelmalla voidaan tehdä samoja toimintoja kuin Active Directoryllä, mutta ilman tarvetta asentaa tiettyä lisäosia, joita vaaditaan Active Directoryyn käyttöä varten. ADAM voi toimia palvelimella tai suoraan Windowsin työpöydällä omana ohjelmana. ADAM tukee myös autentikointi- ja autorisointipalveluja. Se voi tukea jo olemassa olevaa Windowsin tai paikallisen tietokoneen autentikointia. ADAM-ohjelmalla on myös oma autentikointi-palvelunsa, jonka voi kohdentaa turvaamaan tiettyjä ohjelmia. (Richards, Allen & Lowe-Norris 2006, 408-409.)

Kuvio 3. Esimerkki verkosta, jossa on käytössä ADAM (Technet Microsoft 2014.)
3 ASP.NET


3.1 Yleistä


3.2 ASP.NET-tuotantomallit


Web Pages on yksinkertainen malli dynaamisten web-sovellusten luomiseen. Malli perustuu yhteen staatiseen HTML-sivuun, jonka lähdekoodiin kirjoitetaan tarvittavat toiminnot. Lähdekoodissa käytetään Razor-syntaksia dynaamisten toimintojen luomiseen. (FitzMacken 2014.)

Web Forms. Web Forms on kehittyneempä versio aiemmasta ASP-sivujen ohjelmoinnista. Sivujen teossa käytetään edelleen kontrollereita, mutta Web Forms tukee .NET-viitekehyksen koodikieliä, kuten C# ja VB.NET. (Microsoft [Viitattu 22.11.2016].)

MVC. MVC (Model View Controller) tarkoittaa arkkitehtonista mallia, jossa sovellus eritellään kolmeksni eri osaksi. Nämä osat ovat malli (Model), näkymä (View) ja kontrolleri (Controller), jotka on kuvattu kuviossa 4. (Microsoft ASP.NET Team 2009.)

3.3 ASP.NET-kontrollerit

ASP.NET web -sovellusten luonnissa voidaan käyttää useita eri tyypisiä kontrolloja. Nämä tyypit ovat HTML server-, validation-, user- ja web server–kontrollerit. (Microsoft [Viitattu 7.11.2016].)

HTML server -kontrollerit ovat tyyppillisä HTML-elementtejä, jotka on muokattu suorittamaan palvelimen kautta kirjoitettua koodia. Tämä mahdollistetaan lisäämällä HTML-elementtiin 'runat="server"'-attribuutti. (Microsoft [Viitattu 7.11.2016].)
Validation-kontrollerit tarkistavat käyttäjän antamat tiedot virheiden varalta, esimerkiksi tekstikentän sisällön tarkistamisen. Jos tekstikenttä on tyhjä tai sisältää tiettyjä erikoismerkkejä, kuten kaksoispisteitä, suoritusta ei toteuteta ennen kuin tekstikentän sisältö on määritysten mukaista. (Microsoft [Viitattu 8.11.2016].)


Web server -kontrollerit ovat palvelinkontrolleja, jotka muodostavat pääosan ASP.NET-sovelluksen sisällöstä. Palvelinkontrollerit sisältävät laajan kirjaston yleisiä standardi-kontrollereita, kuten myös monimutkaisimpia kontrolloita, kuten kalentereita tai taulukoita. Kontrolloja luodessa kerrotaan kontrollin tyyppi, sen ID ja runat="server"-attribuutti. (Microsoft [Viitattu 7.11.2016].) Esimerkiksi painike-kontrolleri kuvataan sovelluksen tyylitiedostossa seuraavalla tavalla:

```html
<asp:button attributes runat="server" id="Button1" />
```
4 VERSIONHALLINTA


4.1 Versionhallinnan vaihtoehdot

Ohjelman varmuuskopioinnin voi tehdä yksinkertaisesti kopioimalla tiedostot toiseen kansioon, mutta tämä altistaa helposti virheille. Kansiot saattavat mennä sekaisin, jolloin muokataan väärä tiedostoja. Tätä ongelmaa varten kehitettiin paikallinen versionhallintajärjestelmä. (Chacon & Straub 2009.)

Kuvio 5. Paikallinen versionhallinta -diagrammi
(Chacon & Straub 2009.)

Ongelman ratkaisuksi kehitettiin keskitetyt versionhallintajärjestelmät. **Keskityssä versionhallinnassa** jokaisella projektilla on yksi ainoa tietokanta, joka sijaitsee omalla palvelimellaan. Palvelimen tietokanta sisältää tiedostojen eri versiot, ja ohjelmoijat voivat muokata niitä ja luoda niistä omat versionsa. Palvelimen versionhallinnan valvoja voi seurata mitä tiedostoja kukin ohjelmoija muokkaa. (Chacon & Straub 2009.)
Vaikka keskitetystä versionhallintajärjestelmästä on lukuisia myönteisiä puolia verrattuna paikalliseen versioon, se sisältää riskinsä. Kaiken tiedon sijoittaminen yhdelle palvelimelle voi aiheuttaa tuhoisat seuraukset, jos varmuuskopioita ei ole tallella ja tietokanta tuhoutuu. Palvelimen huollon aikana ohjelmoijat eivät voi tehdä yhteistyötä eivätkä lisätä uusimpia tiedostoja järjestelmään. Sama ongelma ilmenee paikallisessa versionhallintajärjestelmässä. (Chacon & Straub 2009.)

Hajautettu versionhallintajärjestelmä ei kärsi paikallisen ja keskitetyn järjestelmän ongelmista. Järjestelmän käyttäjät hakevat viimeisimmän version lisäksi koko tietolähteestä. Tämän ansiosta palvelimen tietokannan tuhoutuminen ei ole enää ongelma, sillä tiedot voidaan kopioida takaisin palvelimelle kenen tahansa käyttäjän tietokoneelta. Monet hajautetut järjestelmät on kehitetty toimimaan useiden etätielolähteiden kanssa, joten niiden avulla saman projektin työstäminen useiden eri kehittäjien kanssa samanaikaisesti on mahdollista. (Chacon & Straub 2009.)
4.2 Git


Jakamattomuudella tarkoitetaan sitä, että järjestelmä keskittyy täysin senhetkiseen tehtävään. Järjestelmän tulisi suorittaa toiminto täysin tai antaa
hylkäävä vastaus. Tarkoituksena on välttää osittaisia operaation suorituksia, jottei osa tiedoista katoaisi suorittamisen yhteydessä. (Ravishankar 2013, 16.)

Useimmat muut versionhallintajärjestelmät (CVS, Perforce, Bazaar ym.) näkevät käyttäjän datan kokonaisena tiedostoja ja niihin tehtyynä muutoksina. Git on tehokas suorittamaan miljoonien tiedostojen käsittelyn sekunneissä. Tämä perustuu Gitin kykyyn käsittää tiedostot. (Ravishankar 2013, 16.)

Kuvio 8. Esimerkki kuinka muut versionhallintajärjestelmät näkevät tiedostot (Ravishankar 2013, 17.)

Git ottaa tilannevedoksen (snapshot) kaikista tiedostoista, jolloin se näkee tiedostojen väliset yhteydet ja suunnittelee toimintansa niiden perusteella. Joka kerta kun ohjelmaa muokataan, Git tekee uuden tilannevedoksen. Git ei siis tallenna uusia versioita muuttumattomista tiedostoista, vaan säilyttää aiemmat tilannevedokset ja jättää viittauksen niihin uudessa versiossa. Tämän ansiosta Git pystyy suoriutumaan tehokkaasti eri toiminoissa. (Ravishankar 2013, 17.)
Kuvio 9. Gitin tilannekuvien varastointi ajan kuluessa (Ravishankar 2013, 17.)

Git pitää huolen tiedostojen **turvallisuudesta** salaamalle ne tarkistussumman avulla SHA-1-tarkistukseksi. Salaamisen jälkeen niihin viitataan tarkistussumman avulla. Tarkisteen pituus on 40 heksadesimaalista (0–9 ja a–f) merkkiä. Tarkistussumma luodaan tiedoston rakenteen perusteella. Tämän ansiosta on mahdotonta muuttaa salattuja tiedostoja ilman että Git huomaisi ne. (Ravishankar 2013, 18.) Esimerkki tarkistussummasta voi olla seuraavanlainen:

24b9da6552252987aa493b52f8696cd6d3b00373

4.3 TortoiseSVN

TortoiseSVN on versionhallintaohjelma Windowsille ja perustuu Apache Subversionin (SVN) avoimeen lähdekoodiin. TortoiseSVN pitää kirjaa tiedostojen muutoksista. Se eroaa Gitistä, koska sen graafinen käyttöliittymä integroituu Windowsin Resurssienhallintaan. (Küng, Onken & Large [Viitattu 19.3.2016].) Tätä
oman ohjelman versionhallinnan voi suorittaa suoraan Windowsin työpöydällä (kuvio 10).

Kuvio 10. TortoiseSVN Windowsin resurssienhallinnassa (The TortoiseSVN Team 2016.)

Ohjelmaa kehitetään GNU General Public License (GPL) -lisenssillä eli ohjelmassa käytetty koodi on vapaasti kaikille käytössä. Myös lähdekoodi on avointa, joten oman kaupallisen version teko on mahdollista. (GNU 1991.) Lisenssi on kuvattu kuviossa 11.
Kuvio 11. GNU-lisenssi
(GNU 1991)

4.3.1 TortoiseSVN-ohjelman ominaisuudet

TortoiseSVN-ohjelman graafisen käyttölőyynnän ansiosta ohjelma tarjoaa tiettyjä ominaisuuksia, jotka eroavat Gitin-versionhallintaohjelmasta. (Küng, Onken & Large [Viitattu 21.3.2016].) Esimerkiksi kuviossa 12 näkyvät ohjelman kuvakepäälykset.

Kuvio 12. TortoiseSVN-kuvakepäälykset
(The TortoiseSVN Team 2016.)

Kuvakepäälykset kuvaavat tiedostojen tilan versionhallinnassa. Vihreät kuvakkeet ilmaisevat tiedostojen olevan ajan tasalla. Punaiset ristit taas tarkoittavat, että kyseiset tiedostot on poistettu uusimmasta versiosta.
TortoiseSVN voi hyödyntää myös Subversion-kirjaston ominaisuuksia, kuten versioituvia hakemistoja. Subversion pystyy tallentamaan versionhallintaan sekä tiedostot että kansiot tallentamalla koko kansioon muutokset. (Küng, Onken & Large [Viitattu 21.3.2016].)
5 AUTENTIKOINTI JA AUTORISOINTI


**Autentikoinnilla** tarkoitetaan käyttäjän tai palvelun tunnistamista. Autentikoinnissa ohjelma lukee käyttäjän antaman tunnuksen ja salasanan ja vertaa niitä tietokannassa olevaan tietoon. Jos tiedot varmistuvat oikeiksi, ohjelma todentaa käyttäjän. (Apple 2012.)

**Autorisoinnilla** tarkoitetaan käyttäjän oikeutta suorittaa tietty toiminto. Ero autentikointiin tulee ilmi monimutkaisimmissa järjestelmissä. Esimerkiksi jos yhdellä tietokoneella on kaksi eri käyttäjää, jotka kirjautuvat sisään omilla tunnuksillaan. Kumpikin voi luoda omia tiedostojaan, mutta he eivät voi muokata toistensa tiedostoja, sillä heiltä puuttuu oikeutet saa varten. (Apple 2012.)

5.1 Autentikointitavat

Seuraavaksi käsitellään muutamia esimerkkejä autentikointi- ja autorisointi-palveluista.

5.1.1 Windows


5.1.2 Forms


Kun tunnistautuminen on vahvistettu, luodaan käyttäjää varten tiketti (Forms Authentication Ticket), joka sisältää tiedoja käyttäjästä ja kertoo ohjelmalle, että käyttäjällä on tarvittavat oikeudet. Tiketti säilytetään tyypillisesti selaimen evästetiedoissa, jolloin käyttäjän ei tarvitse joka kerta kirjautua sisään saman istunnon aikana. (Mitchell 2008.)

Kuviossa 13 havainnollistetaan kuinka Forms-autentikoinnilla toimiva ASP.NET-sivu käyttäytyy, kun yritetään suorittaa tunnistamatonta toimintoa.
Kuvio 13. Forms-autentikoinnin toimintaperiaate (Mitchell 2008.)

6 ASP.NET-SOVELLUKSEN TOTEUTUS


6.1 Versionhallinta

Sovelluksen työstämisen aikana joka muutoksen jälkeen lisättiin uusi versio sovelluksesta versionhallinnan palvelimelle. Kuviossa 15 on esitettynä osa sovelluksen versiohistoriasta.
34

Kuvio 15. Sovelluksen versiohistoria

6.2 ASP.NET-sovelluksen kuvaus


Salasanan nollaus

Kirjoita nimi muodossa "Sukunimi, Etunimi", Kirjoita puhelinnumero yhteen. Textboxissä Enter-painikkeella oikeaa haun suorittamanen toimii vain Internet Explorer ja Firefox -selaimissa. Hae käyttäjää seuraavilla arvoilla:

<table>
<thead>
<tr>
<th>Nimi</th>
<th>Etsi nimen perusteella</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-Tunnus</td>
<td>Hae käyttäjätunnuksen perusteella</td>
</tr>
<tr>
<td>Puh.nro</td>
<td>Hae puhelinnumeron perusteella</td>
</tr>
</tbody>
</table>

Tyhjää

<table>
<thead>
<tr>
<th>Nimi</th>
<th>Tunnus</th>
<th>Kuvaus</th>
<th>Kännykkä</th>
<th>Synt. aika</th>
<th>Tunnukseen tila</th>
<th>BadPwdCount</th>
<th>PwdLastSet</th>
<th>Last Logon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opiskelija, Eli</td>
<td>k8765432</td>
<td>LITA20</td>
<td>Opiskelija</td>
<td>Toiminnassa</td>
<td>badPwdCount</td>
<td>ci löytynyt</td>
<td>20.5.2016 14:39:28</td>
<td>lastLogon ci löytynyt</td>
</tr>
</tbody>
</table>

Kuvio 16. Sovelluksen käyttöliittymä

Kuviossa 16 havainnollistetaan myös testipiskelijan haun tulos. Listassa näkyy opiskelijan nimi, käyttäjätunnus, lyhyt kuvaus opiskelijasta, puhelinnumero, syntymäaika, tunnuksen tila (lukittu tai vanhentunut), viimeisin salasanan vaihto ja viimeisin kirjautuminen. Kun valitaan tietty opiskelija, avautuu uuteen ikkunaan opiskelijan tiedot. Ikkunassa on vaihtoehdot salasanan uusimista varten (kuvio 17).
Kuvio 17. Testikäyttäjän tietojen tarkastelu

Kuvio 18. ASP.NET-mallipohjan valitseminen

6.3 Active Directory -haun toteutus


Ohjelmakoodissa tutkitaan ensin, mikä hakukenttä on valittu. Hakukenttien painikekontrollerit lähetttävät kyseisen hakukentän attribuutin ja sisällön hakukooodiin. Hakukentän attribuutin avulla alustetaan hakukooodi käytämään oikeaa hakukriteeriä. Seuraavassa koodiesimerkissä (esimerkki 1) opiskelijan nimeä varten tarkoitettu hakukenttä lähetttää attribuutin "nimi" hakukooodiin, jolloin hakukooodi käyttää attribuuttia "displayName" hakukriteerinä. "displayName"-attribuutti on Active Directoryn resurssien attribuutti, jolla tarkoitetaan AD-palvelussa sijaitsevien käyttäjien nimiä.
if (attribute.Equals("nimi"))
{
    attributeForQuery = "displayName";
    string[] querystringToSplit = receivedQueryString.Split(' ');
    queryString = querystringToSplit[0] + " " + querystringToSplit[1];
}
if (attribute.Equals("opiskelijaTunnus"))
{
    attributeForQuery = "sAMAccountName";
    queryString = receivedQueryString;
}
if (attribute.Equals("numero"))
{
    attributeForQuery = "mobile";
    queryString = receivedQueryString;
    queryString.Trim();
}

Esimerkki 1.

Ensiksi alustetaan tarvittava DataTable-taulukko. Tähän tauluksoon syötetään hakutuloksen tiedot. Taulukkoon lisätään hakua varten tarpeelliset sarakkeet, jotka nimetään Active Directoryn attribuuttien mukaan.

    DataTable table = new DataTable();
    DataRow oneRow;

    table.Columns.Add("displayName");
    table.Columns.Add("sAMAccountName");
    table.Columns.Add("Description");
    table.Columns.Add("mobile");
    table.Columns.Add("schacDateOfBirth");
    table.Columns.Add("userAccountControl");
    table.Columns.Add("badPwdCount");
    table.Columns.Add("pwdLastSet");
    table.Columns.Add("lastLogon");

Esimerkki 2.


    DirectoryEntry AD;
    DirectorySearcher getAllUsers;
    DirectoryEntry ADAM;
    DirectorySearcher getAllUsersADAM;
XmlDocument readXML = new XmlDocument();
XmlNode DC = readXML.SelectSingleNode("Configuration/DC");
XmlNode allOUs = readXML.SelectSingleNode("Configuration/searchOUs");
XmlNode ADRoot = readXML.SelectSingleNode("Configuration/ADRoot");
XmlNode ADAM = readXML.SelectSingleNode("Configuration/ADAM");
XmlNode ADAMRoot = readXML.SelectSingleNode("Configuration/ADAMRoot");
string empID = null;

AD = new DirectoryEntry("LDAP://" + DC.InnerText.ToString() + "/" + ADRoot.InnerText.ToString());
getAllUsers = new DirectorySearcher(AD);

Esimerkki 3.


getAllUsers.Filter = "(& (objectCategory=user)(" + attributeForQuery + "=*" + queryString + ")(!pwdLastSet=0))";

Esimerkki 4.

Seuraavaksi määritellään, mitkä attribuutit haetaan hakukriteerin suodattamista käyttäjistä.

getAllUsers.PropertiesToLoad.Add("displayName");
ggetAllUsers.PropertiesToLoad.Add("sAMAccountName");
ggetAllUsers.PropertiesToLoad.Add("Description");
ggetAllUsers.PropertiesToLoad.Add("badPwdCount");
ggetAllUsers.PropertiesToLoad.Add("pwdLastSet");
ggetAllUsers.PropertiesToLoad.Add("lastLogon");
ggetAllUsers.PropertiesToLoad.Add("employeeID");
ggetAllUsers.PropertiesToLoad.Add("userAccountControl");
ggetAllUsers.PropertiesToLoad.Add("mobile");
ggetAllUsers.PropertiesToLoad.Add("distinguishedName");

Esimerkki 5.


foreach (SearchResult oneResult in getAllUsers.FindAll())
oneRow = table.NewRow();

    if (oneResult.Properties.Contains("displayName"))
    {
        oneRow["displayName"] = oneResult.Properties["displayName"][0].ToString();
    }
    else oneRow["displayName"] = "Displayname ei löytynyt";
    if (oneResult.Properties.Contains("employeeID"))
    {
        empID = oneResult.Properties["employeeID"][0].ToString();
    }

Esimerkki 6.


getAllUsers.Dispose();
    AD.Close();

    ADAM = new DirectoryEntry("LDAP://" + ADAMIP.InnerText.ToString() + "/" + ADAMRoot.InnerText.ToString());
    getAllUsersADAM = new DirectorySearcher(ADAM);
    getAllUsersADAM.Filter = "(uid=" + empID + ")";
    getAllUsersADAM.SearchScope = SearchScope.Subtree;
    getAllUsersADAM.PropertiesToLoad.Add("schacDateOfBirth");
    getAllUsersADAM.PropertiesToLoad.Add("mobile");

    try
    {
        foreach (SearchResult ADAMResult in getAllUsersADAM.FindAll())
        {

            Esimerkki 7.

Lopuksi katkaistaan ADAM-yhteys ja lisätään tiedot DataTablen riviin. DataTable siirretään GridView-kontrolleriin.

    }
    getAllUsersADAM.Dispose();
    ADAM.Close();

    table.Rows.Add(oneRow);
}
gridViewADAM.DataSource = table;
gridViewADAM.DataBind();
Esimerkki 8.

GridView-kontrolleri esittää haetut tiedot taulukkomuodossa (kuvio 16). Jos tietyn opiskelijan tunnuksset ovat vanhentuneet, esitetään kyseisen opiskelijan rivi punaisena.

6.4 Salasanan uusinta


Kuvio 19. ASP.NET Designer -sivu salasanan vaihdon tuloksesta

6.5 Ennakoiva haku

Kun kirjoitetaan opiskelijan nimeä hakukenttään, ennakoiva haku suorittaa hakutoimintoa taustasuorituksena ja tuo listan opiskelijoista hakukenttään. Tämän haun avulla voidaan täsmentää haluttu opiskelija hakutuloksista.


### 6.6 Salasanan lähetys tekstiviestinä

Salasanan vaihdon yhteydessä voi valita vaihtoehton, jolla uusi salasana lähetetään opiskelijalle tekstiviestinä. Tekstiviestin lähetettä Jelpparin oma palvelin, johon on ohjelmoinut Jelpparin toimesta tekstiviestejä lähetettävä sovellus. Tähän palvelimeen otetaan SQL-yhteys, jonka jälkeen luodaan SQL-komentona
tekstiviestinä lähetettävä viesti. Koodi SQL-komennon luomiseen kuvataan liitteessä 3.
7 YHTEENVETO JA POHDINTA

Opinnäytetyön tavoitteena oli suunnitella ja toteuttaa ASP.NET-sovellus, joka pystyy hakemaan ja uusimaan SeAMKin opiskelijoiden salasanoja. Työssä tarkasteltiin versionhallintaa ja sen käyttöä osana sovelluksen kehitystä. Osana web-sovellusten kehitystä kerrottiin myös tietoturvasta.


Tämän työn tekijällä oli vain vähän kokemusta ASP.NET-sovellusten kehityksestä, joten haasteita oli monia. Testaamisessa huomattiin jälkikäteen, että tietyt toiminnot eivät onnistuneet paikallisessa ympäristössä, vaan sovellus piti siirtää SeAMKin tietohallinnon palvelimelle. Näiden toimintojen testaus jäi vähälle, sillä testaus piti ulkoistaa tietohallinnon työntekijöille, joilla oli kiireitä muiden projektien kanssa. Sovelluksesta saatii kuitenkin hyvää mallipohja jatkokehitystä varten.

Loppujen lopuksi työstä sai todella paljon kokemusta hakemistopalveluista, versionhallinnasta ja ASP.NET-sovellusten kehittämisestä. Työssä saatii toteutettua web-sovellus, joka hakee SeAMKin opiskelijoiden tiedot hakusanan perusteella Active Directorystä.

ASP.NET-sovelluksilla voidaan luoda ohjelma jotka suorittavat monimutkaisia toimintoja helpolla käyttöliittymällä. Tämän ansiosta sovelluksen käyttäjältä ei vaadita erityistä tietoteknistä osaamista. ASP.NET-sovellukset ovat edelleen kehityksessä, joten näiden sovellusten suosio varmasti kasvaa.
LÄHTEET


Microsoft Corporation. [Viitattu 29.10.2016]. Saatavissa:

Microsoft ASP.NET Team. 2009. ASP.NET MVC Overview. [www-dokumentti].
Microsoft Corporation. [Viitattu 25.10.2016]. Saatavissa:
https://www.asp.net/mvc/overview/older-versions-1/overview/asp-net-mvc-overview

[Viitattu 12.10.2016]. Saatavissa: https://www.asp.net/aspnet

Microsoft. Ei päiväystä. ASP.NET Web Server Controls Overview. [www-
dokumentti]. Microsoft Corporation. [Viitattu 7.11.2016]. Saatavissa:

Microsoft. Ei päiväystä. ASP.NET Validation Controls. [www-dokumentti].
Microsoft Corporation. [Viitattu 8.11.2016]. Saatavissa:
https://msdn.microsoft.com/en-us/library/debza5t0.aspx

Microsoft. Ei päiväystä. ASP.NET User Controls. [www-dokumentti]. Microsoft

Microsoft. Ei päiväystä. Gridview control. [www-dokumentti]. Microsoft

Microsoft. Ei päiväystä. Introduction to ASP.NET Web Forms. [www-dokumentti].
Microsoft Corporation. [Viitattu 22.11.2016]. Saatavissa:
https://www.asp.net/web-forms/what-is-web-forms

Microsoft. Ei päiväystä. Windows Authentication <windowsAuthentication>. [www-
dokumentti]. Microsoft Corporation. [Viitattu: 28.9.2016]. Saatavissa:
http://www.iis.net/configreference/system.webserver/security/authentication/window
dowsauthentication

Microsoft Corporation. [Viitattu: 29.9.2016] Saatavissa: http://www.asp.net/web-
forms/overview/older-versions-security/introduction/security-basics-and-asp-net-support-cs

Saatavissa: https://www.nuget.org/packages/AjaxControlToolkit/


**LIITTEET**

Liite 1. Koodi opiskelijan DN-nimen hausta ja salasanan vaihdosta

Liite 2. Ennakoivan haun metodi

Liite 3. Koodi tekstiviestin lähetyksestä SQL-komentona
LIITE 1. Koodi opiskelijan DN-nimen hausta ja salasanan vaihdosta

```csharp
public enum objectClass
{
    user, group, computer
}
public enum returnType
{
    distinguishedName, ObjectGUID
}

public string GetObjectDistinguishedName(string usersAMAccountName)
{
    XmlDocument readXML = new XmlDocument();
    XmlNode DC = readXML.SelectSingleNode("Configuration/DC");
    XmlNode ADRoot = readXML.SelectSingleNode("Configuration/ADRoot");

    string distinguishedName = string.Empty;
    string connectionPrefix = "LDAP://" + DC.InnerText.ToString() + "/" + ADRoot.InnerText.ToString();
    DirectoryEntry entry = new DirectoryEntry(connectionPrefix);
    DirectorySearcher mySearcher = new DirectorySearcher(entry);

    mySearcher.Filter = "(&(objectClass=user)(| (cn = "+ usersAMAccountName + ")(sAMAccountName = " + usersAMAccountName + "))");
    SearchResult result = mySearcher.FindOne();
    if (result == null)
    {
        throw new NullReferenceException
        ("Unable to locate the distinguishedName for the user "+ usersAMAccountName + " in the " + DC.InnerText.ToString() + "/" + ADRoot.InnerText.ToString() + " domain");
    }
    DirectoryEntry directoryObject = result.GetDirectoryEntry();

    distinguishedName = "LDAP://" + directoryObject.Properties["distinguishedName"].Value;
    entry.Close();
    entry.Dispose();
    mySearcher.Dispose();
    return distinguishedName;
}

public void ResetPassword(string userDistinguishedName, string password)
{
    DirectoryEntry uEntry = new DirectoryEntry(userDistinguishedName);
    uEntry.Invoke("SetPassword", new object[] { password });
    uEntry.Properties["LockOutTime"].Value = 0;
    uEntry.Close();
}
```
LIITE 2. Ennakoivan haun metodi

```csharp
[WebMethod]
public string[] GetUserListFromAD(string prefixText)
{
    ArrayList allUsers = new ArrayList();
    DirectoryEntry AD;
    DirectorySearcher getAllUsers;

    XmlDocument readXML = new XmlDocument();
    XmlNode DC = readXML.SelectSingleNode("Configuration/DC");
    XmlNode allOUs = readXML.SelectSingleNode("Configuration/searchOUs");

    bool amountOfWords = prefixText.ToLowerInvariant().Contains(',', ' ');
    string searchFilter = "displayName";

    foreach (XmlNode oneOU in allOUs)
    {
        AD = new DirectoryEntry(@"LDAP://" + DC.InnerText.ToString() + "/" + oneOU.InnerText.ToString());
        getAllUsers = new DirectorySearcher(AD);

        getAllUsers.Filter = "(&(objectCategory=user)(" + searchFilter + "=*" + prefixText + ")userAccountControl=512)(pwdLastSet=0)"
        getAllUsers.SearchScope = SearchScope.Subtree;
        getAllUsers.PropertiesToLoad.Add("displayName");
        getAllUsers.PropertiesToLoad.Add("Description");

        try
        {
            foreach (SearchResult oneResult in getAllUsers.FindAll())
            {
                string userToAdd;

                if (!allUsers.Contains(oneResult.Properties[searchFilter][0].ToString()))
                {
                    userToAdd = oneResult.Properties[searchFilter][0].ToString();

                    if (oneResult.Properties.Contains("Description"))
                    {
                        userToAdd += " " + oneResult.Properties["Description"].ToString();
                    }

                    allUsers.Add(userToAdd);
                }
            }
        }
        catch (DirectoryServicesCOMException e)
        {
        }
        getAllUsers.Dispose();
        AD.Close();
    }
```
getAllUsers.Dispose();
AD.Close();
}
}
allUsers.Sort();

string[] array = new string[allUsers.Count];
allUsers.CopyTo(array);

return array;
public bool SendSMSwithPwd(string passwordReceived, bool isCheckBoxChecked, bool isCheckBoxChangePwdChecked)
{
    // Viesti tekstiviestiin
    string messageToEncode = "Hei! Salasanasi on vaihdettu! Uusi salasana on: " + passwordReceived
    + " Terveisin Jelppari!";
    string s_url_encoded = HttpUtility.UrlEncode(messageToEncode);

    XmlDocument readXMLSMS = new XmlDocument();
    readXMLSMS.Load(HttpContext.Current.Server.MapPath(@"SMSconfig.xml");

    XmlNode _sqlsrv = readXMLSMS.SelectSingleNode("SendSMS/SMSGatewaySrv");
    XmlNode _sqldb = readXMLSMS.SelectSingleNode("SendSMS/SMSGatewayDB");
    XmlNode _email = readXMLSMS.SelectSingleNode("SendSMS/emailSenderAddress");
    XmlNode _smtpsrv = readXMLSMS.SelectSingleNode("SendSMS/smtpSrv");
    XmlNode _logfile = readXMLSMS.SelectSingleNode("SendSMS/logFile");
    XmlNode _max = readXMLSMS.SelectSingleNode("SendSMS/maxMessages");

    string userRealName = null;
    string lastSqlIdentity = null;

    conn.Open();

    SqlCommand command;
    SqlDataReader ident;

    userRealName = getAttributeTest(User.Identity.Name.ToString().Substring(User.Identity.Name.ToString().LastIndexOf(@"\") + 1), "cn");

    string numberMobile = Session["mobile"].ToString();

    if (IsValidPhone(numberMobile))
    {
        // First, insert new outgoing entry
        command = new SqlCommand("INSERT INTO Outgoing (Number, Message) VALUES ('" + numberMobile + ", ' + s_url_encoded + ")", conn);
        command.ExecuteNonQuery();
        command.Dispose();

        // Second get inserted identity (SELECT SCOPE_IDENTITY())
        command = new SqlCommand("SELECT SCOPE_IDENTITY() FROM Outgoing", conn);
        ident = command.ExecuteReader();
        while (ident.Read())
        {
            lastSqlIdentity = ident[0].ToString();
        }
        ident.Dispose();
    }
command.Dispose();

// Third, make log about send
command = new SqlCommand("INSERT INTO OutgoingLogs (senderID, senderName, DateTime, Number, Message, OutgoingId, Successful) VALUES ('" + User.Identity.Name.ToString() + ',', "' + userRealName + ", " + numberMobile + ", " + s_url_encoded + ", " + lastSqlIdentity + ", 0)", conn);
command.ExecuteNonQuery();
command.Dispose();

// Lokiviesti omaan tekstitiedostoon:
createLogMessage(Session["userSAMAccountName"].ToString(), isCheckBoxChecked, isCheckBoxChangePwdChecked);

searchByName = false;
searchByID = false;
searchByPhone = false;

conn.Close();

Session.Add("SMSstatus", "Kyllä");
return true;