Haaga-Helia
ammattikorkeakoulu Oy

Extranet-sivun toteutus koodigeneraattorilla

Emilia Salo

Opinnäytetyö
Tietojenkäsittelyn
koulutusohjelma
2018

Haaga-Helia
ammattikorkeakoulu Oy

Tiivistelmä

Tekijä(t)
Emilia Salo

Koulutusohjelma
Tietojenkäsittelyn koulutusohjelma

Raportin/Opinnäytetyön nimi
Extranet-sivun toteutus koodigeneraattorilla

Sivu- ja liitesivumäärä
21 + 2

Opinnäytetyön nimi englanniksi
Implementation of the extranet-page with code generator

Opinnäytetyön tavoitteena oli selvittää koodigeneraattoreiden käytön hyötyjä ja mahdollisia rajoituksia M-Files -pohjaisen extranet-sivun rakentamiseen. Extranet-sivulla pitäisi pystyä selaamaan MFSQL-tietokannasta haettua tilausdataa omalla käyttäjätunnuksellaan.

Teoriaosuudessa käydään läpi taustalla olevat teknologiat, M-Files ja MFSQL. Tämän jälkeen perehdytään valittuun ohjelmistokehykseen, ASP.NET MVC teknologiaan ja perehdytään tarkemmin kahteen koodigeneraattoriin. Tutkittavat koodigeneraattorit, Code On Time ja ASP.NET Maker, ovat valikoituneet sen mukaan, että tuottavat ASP.NET MVC -pohjaista koodia.

Tutkimusosassa vertaillaan kolmea menetelmää luoda määritelty extranet-sivu. Näihin kolmeen menetelmään kuuluu Code On Time -generaattori, ASP.NET Maker -generaattori, sekä ASP.NET -sivun toteuttaminen täysin alusta asti itse ohjelmoimalla. Vertailussa keskitytään erityisesti siihen, kuinka nopeasti sivun saa luotua, kuinka kustomoitavaa tuotettu koodi on, millaiset autentikointi komponentit ovat mahdollisia, sekä kuinka testattavaa koodi on.

Opinnäytetyön lopussa pohditaan eri menetelmien hyviä ja huonoja puolia, sekä millaiset käyttäjät eri menetelmistä hyötyvät. Lisäksi pohditaan omia opinnäytetyön tavoitteiden toteutumista. Loppuun on liitetty Code On Time -generaattorilla luodun demosivun kuvankaappauksia.

Asiasanat
ASP.NET, Extranet, Koodigeneraattori, M-Files

Sisällys

1 Johdanto

Ohjelmistokehittäjän työssä, erityisesti uutta ohjelmistokieltä käytettäessä, voi yksinkertaisenkin web-sovelluksen luomiseen mennä turhan paljon aikaa. Kiireisessä projektiarjessa helpottavien työkalujen löytäminen on tarpeen. Tähän yhtenä vaihtoehtona ovat koodigeneraattoreiden käyttö ohjelmoinnin tukena.

Koodigeneraattorilla tässä yhteydessä tarkoitetaan työkalua, joka tuottaa (generaattorin mahdollisten asetusten mukaan) valmista ohjelmistokoodia (Techopedia). Esimerkiksi generaattoriin voidaan yhdistää tietokanta, jonka pohjalta generaattori luo valmiin koodipohjan, sekä käyttöliittymän annetun datan pohjalta.

Tämän kyseisen opinnäytetyön idea lähti kirjoittajan omasta kokemuksesta M-Files projekteissa. Joissakin tapauksissa projektissa on tarpeen rakentaa M-Files-dataan pohjautuva web-sovellus, eräänlainen extranet-sivu. Tästä inspiroituneena lähdettiin selvittämään mahdollisuuksia tällaisen sivun rakentamiseen ja päädyttiin vertailemaan ASP.NET sivun ohjelmointia alusta asti itse, sekä vaihtoehtoisesti sivun rakentamista koodigeneraattoreita hyödyntäen. Kiinnostavaa on, kuinka pitkälle koodigeneraattorilla tehdyn sivun saa, kuinka helposti ja nopeasti. Onko koodigeneraattoreiden käyttö kannattavaa vai asettaako niiden käyttö joitakin rajoitteita?

Tätä lähdetään opinnäytetyössä selvittämään. Tutkimuksen alussa käydään läpi sivun pohjalla olevat teknologiat, sekä vaihtoehtoiset menetelmät extranet-sivun rakentamiseen. Empiirisessä osassa vertaillaan näitä vaihtoehtoisia menetelmiä muutamasta eri suunnasta. Lopputulemana olisi tarkoitus saada käsitys koodigeneraattoreiden hyödyistä, sekä rajoitteista, sekä siitä, onko generaattoreiden käyttö kannattavampaa kuin ASP.NET -sivun rakentaminen alusta asti itse. Loppuun on liitetty kuvankaappauksia yhdellä generaattoreista luodusta demosivusta.

2 M-Files ja MFSQL sovelluksen pohjana

Tutkimuksessani keskityn tietyillä vaatimuksilla kehitettyihin sivuihin. Valinta on tehty vastaamaan omissa työtehtävissäni M-Files projekteissa vastaavia vaatimuksia. Kehitettävä extranet-sivu tulee rakentumaan M-Files:in ja MFSQL:n päälle.

2.1 M-Files

M-Files on suomalaisen M-Files Oy:n tuottama tiedonhallintajärjestelmä. M-Files on käytössä tuhannella yrityksellä Suomessa ja sen lisäksi 6000 yrityksellä Suomen ulkopuolella. M-Files voidaan räätälöidä monenlaiseen tarpeeseen eri yritysten vaatimusten ja tarpeiden mukaan. Yrityksen omien sivujen mukaan käyttötarpeita ovat muun muassa tiedonhallinta, dokumentinhallinta, sopimustenhallinta ja laadunhallinta. Lisäksi M-Filesin viimeisimpään versioon kuuluu myös älykerros (IML), joka mahdollistaa tiedon tallennuspaikasta riippumatta pääsyn kaikkeen liiketoimintatietoon. Sen lisäksi, että M-Files voidaan ottaa yrityksissä käyttöön sellaisenaan, se on myös mahdollista integroida yrityksellä jo käytössä oleviin tietokantoihin, sekä ERP- ja CRM -järjestelmiin. (M-Files)

M-Filesin ominaispiirteisiin kuuluu, että se perustuu perinteisen kansiorakenteen sijaan metatietorakenteeseen. Metatietorakenteella tarkoitetaan, että tiedoilla ja tiedostoilla ei ole fyysistä sijaintia, vaan niitä haetaan ja organisoidaan metatietojen perusteella. Metatiedot ovat dokumenttiin tai tietoon liittyvää ominaisuustietoa, kuten tarjousdokumenttiin liittyen tarjouksen osapuolet tai tarjouksen hyväksymispäivä. Metatietorakenteen hierarkia rakentuu dokumenttivarastosta, kohdetyypeistä, luokkaryhmistä, luokista, ominaisuusmäärittelyistä ja arvolistoista. (M-Files koulutusmateriaali 2017)

M-Filesin avulla on myös mahdollista yksinkertaisesti hallita yksittäisten dokumenttien, metatietojen tai näkymien luku- ja muokkausoikeuksia. M-Filesissa voidaan luoda käyttäjiä ja käyttäjäryhmiä. Näille voidaan näkymiä, tietoja, tai dokumentteja luodessa ja muokatessa antaa erilaisia oikeuksia. (M-Files)

M-Files-ohjelmisto koostuu useammasta eri osasta. M-Files Desktop on se osa järjestelmää, joka näyttää sisällön eri järjestelmissä, ja sisältää ne toiminnallisuudet, joita peruskäyttäjä voi tehdä. M-Files Admin on järjestelmänvalvojille tarkoitettu työkalu, M-Files Server hallitsee sisällön keskitettyä tallentamista ja jakamista ja M-Files Web mahdollistaa

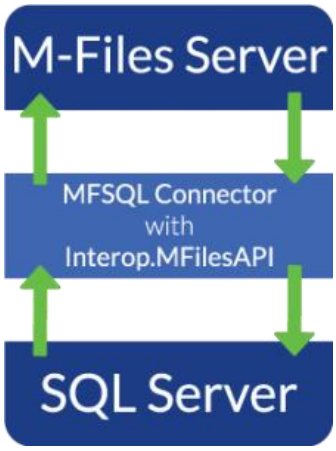
ohjelmiston käytön www-selaimen avulla. Muita osia ovat M-Files asennusohjelma, M-Files Desktop -asetukset, Näytä tilatietoja, sekä Automaattiset päivitykset -osat. (M-Files 2015.3 Käyttöopas)

M-Files tarjoaa kehittäjille käyttöön kaksi ohjelmointirajapintaa: COM/.NET API, sekä M-Files Web Service (MFWS). M-Files COM/.NET on kaikkein kattavin sovellusliittymä ja se tarjoaa rajapinnat sekä käyttäjä, sekä hallinnollisille toiminnoille. Rajapintaa voidaan käyttää mistä tahansa ohjelmasta, joka voi olla vuorovaikutuksessa COM:n kanssa. API on tarjolla sekä 32-, että 64-bittisinä versioina ja siihen tavallisimmin viitataan lisäämällä viittaus Visual Studioon. API:a voidaan käyttää lähes minkä tahansa toiminnon löytämiseen tai käsittelyyn M-Files varaston sisällä. API:n tuettuja kieliä ovat mm. VB.NET, C#, Visual Basic, VBScript ja C++. M-Files Web Service taas on REST-rajapintaa vastaava verkkopalvelu, joka on vuorovaikutuksessa M-Files Web Access -palvelun kanssa. Erona COM API:in ovat muun muassa se, että Web Service:n voidaan ottaa yhteyttä mistä tahansa ympäristöstä, joka pystyy tekemään HTTP-pyyntöjä. COM API on käytettävissä vain Windows ympäristössä. MFWS myös tukee kaikkia käyttäjätason toimintoja, mutta ei tue (ainakaan ilman lisäkonfigurointeja) hallinnollisia toimintoja. (M-Files Developer portal)

2.2 MFSQL

MFSQL Connector on Laminin Solutions: in tuottama viitekehys, joka mahdollistaa M-Files:n ja MS SQL:n, eli Microsoft SQL serverin, välisen vuorovaikutuksen. Vuorovaikutus mahdollistaa muun muassa kaksisuuntaisen integraation M-Files:n ja muiden sovellusten välille. MFSQL Connector:in avulla saadaan käyttöön kehittyneitä metatiedon hallinta- ja siivoustyökaluja. Lisäksi sovelluksen avulla voidaan luoda M-Files Add-On sovelluksia käyttämällä SQL:ää laajennettuna tietokantana näille sovelluksille. (Laminin Solutions 2018)

Connector on suunnattu lähinnä viitekehykseksi sovelluskehittäjille tai konsulteille, eikä niinkään loppukäyttäjille. Todennäköisesti tavallinen M-Files käyttäjä ei tule koskaan olemaan tekemisissä suoraan Connectorin kanssa, siitä huolimatta, että loppukäyttäjän käyttämät erikoissovellukset saattavat käyttää sitä toimiakseen. (Laminin Solutions 2018)



Kuva 1. MFSQL toimintaperiaate (Laminin Solutions 2018)

Mahdollistaakseen metatiedon käytön Connector tuo metatiedon M-Filesista LSConnect Wrapper asemblyllä joka pakkaa interop.M-Files API:n ja päivittää SQL serverin kaksisuuntaisesti käyttämällä erilaisia T-SQL menetelmiä, jotka ovat vuorovaikutuksessa assemblyn kanssa. Connector toimii yhden nimetyn M-Files varaston kanssa. Jokaiselle M-Files varastolle, jotka vaativat vuorovaikutusta palvelimen kanssa, täytyy asentaa omansa. Connector pitää sisällään suurimman osan nimetyn M-Files varaston metatiedosta ja luokkien yksityiskohdista. Fyysisiä tiedostoja Connector ei pura, mutta mahdollistaa vuorovaikutuksen tiedostojen metatietoihin. Huomionarvoista on myös, että Connector ei vaadi, eikä käytä tavallista M-Files SQL varaston tietokantaa, vaan toimii täysin erillisessä ja itsenäisessä SQL tietokannassa. (Laminin Solutions 2018)

3 ASP.NET ja koodigeneraattorit

Web-ohjelmistokehykseksi on valikoitunut ASP.NET. Tähän tulokseen on päädytty, koska M-Files API tukee C# ohjelmistokieltä ja MFSQL käyttää Microsoft SQL serveriä, joten on helpointa käyttää Microsoftin tuottamaa ohjelmistokehystä. Tästä syystä myös vertailtavat koodigeneraattorit on valikoituneet sellaisiksi, että tuottavat ASP.NET koodia, eikä muita tässä tutkimuksessa huomioida. Tässä luvussa käydään läpi mitä ASP.NET ja koodigeneraattorit ovat, sekä perusominaisuudet kahdesta vertailtavasta koodigeneraattorista; Code On Time ja ASP.NET Maker. Koodigeneraattoreiden ominaisuuksia käydään perusteellisemmin läpi luvussa 4.

3.1 .NET Framework

.NET Framework on Microsoftin tuottama ohjelmistokehys, joka toimii Windowsilla kehitysalustana ja tarjoaa erilaisia palveluita .NET Frameworkia käyttäville sovelluksille. .NET Framework rakentuu kahdesta olennaisesta komponentista: ajoympäristöstä eli Common Language Runtimesta (CLR) ja luokkakirjastosta (.NET Framework Class Library). CLR käsittelee siis käynnissä olevia sovelluksia, ja luokkakirjasto tarjoaa testattua, uudelleenkäytettävää koodia, jota kehittäjät voivat kutsua omista sovelluksistaan.

.NET Framework -ohjelmistokehys tarjoaa erilaisia palveluita, muun muassa muistin hallintaan, versionhallintaan, sekä erilaisia kehitysympäristöjä ja teknologioita. .NET Framework pitää myös sisällään yleisen tyyppijärjestelmän, joka mahdollistaa useamman ohjelmointikielen yhteensopivuuden. (Microsoft 2018)

3.2 ASP.NET

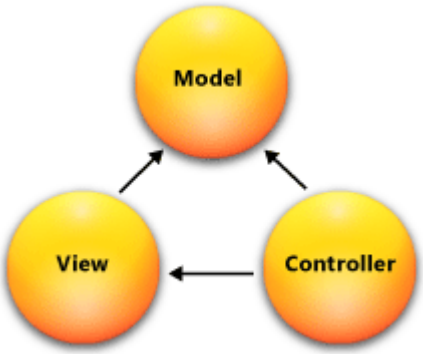
ASP.NET on Microsoftin kehittämän .NET Frameworkin sisältämä ilmainen avoimen lähdekoodin web-ohjelmistokehys, jolla voidaan rakentaa erilaisia palvelimella suoritettavia dynaamisia web-sovelluksia. ASP.NET sisältää kattavan ohjelmistoinfrastruktuurin, ohjelmointimallin ja web-sovellusten kehitykseen tarvittavat palvelut. ASP.NET toimii HTTP-protokollan päällä ja käyttää standardimetodeja ja käytäntöjä selaimen ja palvelimen väliseen kommunikointiin. ASP.NET koodia voidaan kirjoittaa kaikilla .NET Frameworkin tukemilla kielillä, joita ovat mm. C#, Visual Basic, JavaScript, HTML ja CSS. (Microsoft 2010)

ASP.NET tarjoaa kolme erilaista viitekehystä, joilla voidaan luoda verkkosovelluksia. Nämä kolme ovat Web Forms, Asp.NET MVC ja ASP.NET Web Pages. Nämä kolme teknologiaa alkavat kaikki olla jo suhteellisen vanhaa teknologiaa, mutta tästä syystä niitä

voidaan pitää melko vakaina. Vaihtoehtona palvelinpohjaiselle ASP.NET:lle on uudempi ASP.NET Core, joka on pilvipohjainen. (Microsoft 2010)

3.3 ASP.NET MVC

ASP.NET MVC ohjelmistokehys luotiin vaihtoehdoksi Web Forms teknologialle MVC-pohjaisten web-sovellusten luomiseen. MVC on kevyt ja helposti testattavissa oleva ohjelmistokehys, joka on integroitu olemassa olevien ASP.NET ominaisuuksien kanssa yhteensopivaksi (esimerkiksi ASP.NET Membershipiin tai ASP.NET Identityyn perustuva autentikointi).



Kuva 2. MVC arkkitehtuuri (Microsoft)

Lyhenne MVC tulee Model-View-Controller arkkitehtuurista. MVC on standardi suunnittelumalli, joka muodostuu kolmesta komponentista: mallista (model), näkymästä (view) ja käsittelijästä (controller). Mallilla tarkoitetaan sovelluksen osia, jotka toteuttavat datalähteiden, esimerkiksi tietokantataulujen käsittelyn. Nämä malliosat yleensä hoitavat tiedon haun ja tallentamisen tietokantaan. Näkymillä taas tarkoitetaan osia, joilla muokataan, miten tietoa näytetään, eli tuottavat sovelluksen käyttöliittymän. Näkymillä voidaan näyttää esimerkiksi tietokantataulusta haettua tietoa myyntitilauksista tekstinä, alasvetovalikkona tai valintaruutuina. Käsittelijät ovat niitä osia, jotka käsittelevät käyttäjän vuorovaikutuksen, käsittelevät mallia ja valitsevat näkymän, joka näkyy käyttöliittymässä. MVC-sovelluksissa näkymä ainoastaan näyttää tietoa ja käsittelijä käsittelee käyttäjän syöttämää tietoa ja vuorovaikutusta sovelluksen ja käyttäjän toimintojen välillä.

MVC-arkkitehtuuri helpottaa luomaan sovelluksia, joissa sovelluksen eri tasot ovat eroteltuina. Tämä mahdollistaa selkeän kehityksen myös monimutkaisemmille sovelluksille, kun sovelluksen koodi on selkeästi jäsennelty. Kehittäjä tietää jo valmiiksi mihin osiin mikäkin koodi kuuluu kirjoittaa (Tietokantayhteyteen liittyvät malliin, käyttöliittymän ulkoasuun liittyvät näkymiin ja niin edelleen). Lisäksi ASP.NET MVC helpottaa automaattisten testien kirjoittamista ja mahdollistaa myös yksittäisten osien testaamisen muusta sovelluksesta erillään. (Microsoft 2009)

3.4 ASP.NET Core

ASP.NET Core on avoimen lähdekoodin ohjelmistokehys modernien pilvipohjaisten web-sovellusten luomiseen. ASP.NET Core ohjelmistokehys on perinteisestä ASP.NET:stä siinä mielessä poikkeava, että se on palvelimen sijaan pilvipohjainen.

ASP.NET Core -pohjaisia sovelluksia voidaan kehittää ja ajaa Windows ympäristön lisäksi myös macOS tai Linux -ympäristöissä. ASP.NET Core tukee samoja ohjelmointikieliä, kuin ASP.NET yleisestikin. ASP.NET Corea voi käyttää joko paikallisesti tai pilvessä, ja samoin sitä voidaan suorittaa .NET Frameworkilla tai .NET Corella. .NET Frameworkia käytettäessä on tietenkin otettava huomioon, että se toimii vain Windowsilla, eikä tue pilvipalvelua. (Microsoft 2018)

Myös ASP.NET Corea voidaan rakentaa hyödyntäen MVC arkkitehtuuria. Uusimpana lisäyksenä MVC arkkitehtuurissa ASP.NET Coressa voidaan hyödyntää Razor Pages -ominaisuutta. Razor Pages on sivupohjainen malli web-käyttöliittymän rakentamiseen. (Microsoft 2018)

3.5 Koodigeneraattorit

Koodigeneraattoreilla tässä tapauksessa tarkoitetaan sovelluksia, joilla pystytään tuottamaan pohja web-sovellukselle ottamalla yhteys tietokantaan. Tutkimukseen valitut koodigeneraattorit luovat koodipohjan kehittäjän tietokannasta ja valituista asetuksista. Se, miten pitkälle pääsee kirjoittamatta yhtään koodia, riippuu generaattorista. Koodigeneraattorilla voidaan tuottaa yksinkertainen sovellus, jopa täysin ilman koodin kirjoittamista, tai sillä saatetaan vain tuottaa helposti muokattava koodipohja. Koodigeneraattoreiden tarkoitus on nopeuttaa ja helpottaa sovelluksen pohjan tuottamisessa, sekä myös tarjota asetuksia ja ominaisuuksia, joilla sovelluksen jatkokehittäminen on helpompaa.

3.5.1 Code On Time

Code On Time on amerikkalainen yritys ja sovelluskehitysympäristö, jolla luodaan tietokantapohjaisia web-sovelluksia. Kehittäjä luo projektin, yhdistää sen generaattorissa haluamaansa tietokantaan, tuo haluamansa tietokantataulut ja rakentaa taulujen väliset suhteet. Code On Time generaattori sitten generoi tämän tietokantarakenteen pohjalta verk-

kosivun. Verkkosivua voidaan tämän jälkeen muokata mieleisekseen generaattorissa asetuksia muuttamalla, tai Visual Studiossa tekemällä muutoksia luotuun koodiin. (Code On Time)

Code On Time luo tietokannan pohjalta MVC, eli Model-View-Controller (malli-näkymä-käsittelijä) -tyyppisen rakenteen. Generaattori luo mallit, kun valitaan tietokantayhteydestä halutut taulut ja sen jälkeen tauluista käsittelijät, sivut ja näkymät. Jos tietokantataulujen välille on rakennettu relaatioita, nämä suhteet luodaan valmiiksi. Tällöin master-detail suhteella olevista tauluista on helppo rakentaa sivuja pelkästään komponentteja paikalleen vetämällä. Suhde voidaan myös rakentaa generaattorissa jälkikäteen, jos suhteita ei haluta tietokantaan asti rakentaa. Samoin käsittelijä voidaan rakentaa muustakin kuin tietokantataulusta, esimerkiksi toisen sovelluksen web service-kutsulla hakemalla tietoa. Yksinkertaisuudessaan sivun voi siis rakentaa suoraan koodaamatta yhdistämällä tietokanta, ja mukauttamalla sivut haluamaansa järjestykseen. (Code On Time)

Code On Time generaattorin luomaa ASP.NET MVC -tyyppistä pohjaa voi muokata joko C# tai Visual Basic ohjelmointikielillä. Lisäksi erilaisilla business rule:illa voi luoda monimutkaisiakin liiketoimintarakenteita. Business ruleja voidaan kirjoittaa niin SQL:llä, JavaScriptillä tai sovelluksen ohjelmointikielellä eli C#:lla tai Visual Basicilla. Koodin muokkaus tapahtuu Visual Studio kehitysympäristössä. Code On Time:n luoma koodi mukailee ASP.NET MVC rakennetta, mutta sisältää paljon Code On Timen omaa kustomointia. Code On Time luo pohjan ja koodin ja asetusmahdollisuuksien lisäksi suoraan sovellukselle myös käyttöliittymän. Käyttöliittymälle voi valita teeman valmiista mahdollisuuksista tai halutessaan muokata css-tiedostoja omanlaisekseen. Käyttöliittymä on valmiiksi responsiivinen ja toimii siis myös mobiililaitteilla. Käyttöliittymässä on myös paljon ominaisuuksia, joilla sovelluksen loppukäyttäjäkin voi itse muokata ulkoasua. (Code On Time)

Code On Timea on mahdollista käyttää ilmaiseksi, mutta kaikki ominaisuudet ja päivitykset saa käyttöön vain ostamalla käyttöoikeuksia. Ilmaisella versiolla on mahdollista tuottaa yksinkertainen, rajoitetusta määrästä tietokantatauluja koostuva sovellus. Ilmaisessa versiossa voidaan tehdä yksinkertaisia business rule -rakenteita ja siinä on mahdollisuus käyttää käyttäjien autentikointiin ainoastaan Code On Time:n suoraan tarjoamaa ASP.NET Membership:ä. Riippuen sitten siitä, onko valmis maksamaan Standard, Premium tai Unlimited -versiosta, generaattoriin ja koodin kustomointiin saadaan uusia lisämahdollisuuksia. Unlimited versiolla voidaan muun muassa luoda kustomoitu käyttäjien autentikointi käyttäen mitä tahansa tarjoajaa, esimerkiksi Googlea tai Facebookia. Muita ominaisuuksia joita maksulliset versiot tuovat ovat muun muassa mahdollisuus rajata nä-

kymien ja tietojen oikeuksia erilaisilla monimutkaisemmillakin säännöillä, mahdollisuus ladata tiedostoja sovelluksesta, muokata sovelluksen automaattisesti luomia kaavioita, sekä tuottaa sovellukselle REST-rajapinta. Standard versio maksaa 349 dollaria, Premium 899 dollaria ja Unlimited 1699 dollaria vuodessa. (Code On Time)

Code On Time generaattoria kehitetään koko ajan ja erityisesti mobiili- ja pilviratkaisuihin on tulossa paljon kehitystä lähitulevaisuudessa. (Code On Time)

3.5.2 ASP.NET Maker

ASP.NET Maker on koodigeneraattori, joka luo tietokannan pohjalta ASP.NET Core 2.0 MVC web-sovelluksen. Tietokantana voi käyttää SQL Serveriä, MySQL:ää, PostgreSQL:ää, Oraclea tai Microsoft Access tietokantaa. Kehityskielenä toimii C#.

ASP.NET Maker yhdistetään tietokantaan, ja tämän jälkeen generaattori luo pohjan web-sovellukselle. ASP.NET Maker luo sovelluksen, joka mahdollistaa käyttäjille haluttujen tietojen katsomisen, muokkaamisen, hakemisen ja poistamisen. ASP.NET luo yksinkertaisen bootstrapilla toteutetun käyttöliittymän, sekä html ja css -tiedostot muokkaamista varten. (ASP.NET Maker 2018)

3.6 Visual Studio -kehitysympäristö

Visual Studio on Microsoftin kehittämä ohjelmistokehitysympäristö, jossa on paljon hyödyllisiä lisäosia tukemaan ohjelmistokehitystä. Visual Studiolla voidaan luoda erilaisia sovelluksia, kuten web-, pilvi-, mobiili- tai Windows-sovelluksia. Visual Studiossa voidaan käyttää yleisimpiä ohjelmointikieliä kuten C# ja Visual Basicia, sekä hyödyntää ASP.NET viitekehyksiä. Visual Studio sisältää lisäkomponentit sovelluksen kehittämiseen ja testaamiseen. Visual Studiosta löytyy myös valmiita pohjia, joilla on helppo luoda viitekehyksiin sopivia ohjelmia, kuten ASP.NET MVC pohjaisia ratkaisuja. (Microsoft)

4 Tutkimuksen tavoite ja suunnitelmakuvaus

Tutkimuksen tavoitteena on vertailla kolmea menetelmää toteuttaa Extranet-sivu. Extranet sivun pohjalla on M-Files ja M-Files:sta MFSQL-tietokantaan tuotu data, jota voidaan käsitellä Microsoft SQL Serverissä. Kyseiset pohjavaatimukset ovat valikoituneet kirjoittajan omasta työelämäkokemuksesta ja tarpeesta. Tavoitellulla Extranet-sivulla on mahdollista selata, muokata ja poistaa tietoja myyntitilauksista. Myyntitilauksella näkyy myyntitilausrivit. Sivulle on mahdollista kirjautua omalla käyttäjätunnuksella ja eri käyttäjätunnuksille on mahdollista rajoittaa sivun tietojen näkymistä. Lisäksi sivulla olisi hyvä pystyä lataamaan tilauksiin liittyviä dokumentteja. Dokumentit eivät ole tallennettuna tietokantaan, vaan ne haetaan käyttäen M-Filesin tarjoamaa REST-rajapintaa.

Tutkimusmenetelmänä on tutkimusaiheen kartoitus ja testaus. Tutkimuksessa tutkitaan Extranet sivun toteutusta kolmella eri menetelmällä. Nämä kolme menetelmää ovat:

- Sivun toteutus alusta asti itse ohjelmoiden ASP.NET-ohjelmointikielellä Visual Studiossa
- Sivun toteutus Code On Time -koodigeneraattorilla
- Sivun toteutus ASP.NET Maker -koodigeneraattorilla

Kirjoittaja kerää ja tutkii jokaiseen menetelmään liittyvää aineistoa, sekä testaa pienimuotoisesti jokaista menetelmää käytännössä. Menetelmiä vertaillaan erityisesti seuraavista näkökulmista:

- Aika: kuinka kauan perustoiminnallisuuksien toteuttamiseen menee aikaa?
- Sovelluksen kustomointi: kuinka paljon valmista koodia on mahdollista kustomoida? Millaisia rajoitteita?
- Autentikointi: miten käyttäjien autentikointi on toteutettu tai on mahdollista toteuttaa?
- Testaus: kuinka testattavaa koodi on tällä menetelmällä?

Jokainen menetelmä käydään läpi yksitellen ja vertaillaan menetelmien välillä. Lopuksi käydään vielä läpi mahdollisia muita selvitystyön ja testauksen ohella esiin tulleita asioita. Tutkimuksen päätelmät ja kirjoittajan omat pohdinnat käydään läpi luvussa 6.

4.1 Extranet sivu (käyttäjätarinat)

Tavallisena käyttäjänä pystyn kirjautumaan sivulle, josta näen minulle oikeutettuja tilaustietoja käyttäjätunnukseni roolin mukaan luokiteltuna. Lisäksi minun on mahdollista ladata sivulta dokumentteja.

Admin-tason käyttäjänä pystyn tavallisen käyttäjän oikeuksien lisäksi luomaan ja poistamaan käyttäjätunnuksia.

4.2 Lähtötaso

Kirjoittajalla itsellään on kokemusta puolen vuoden ajalta Code On Time:sta, sekä sitä kautta ASP.NET ohjelmoinnista. Kirjoittaja on myös käynyt muutaman verkkokurssin aiheesta. Tämä siis tulee ottaa huomioon vertailun edetessä. Code On Time on entuudestaan tutuin, ASP.NET ohjelmointi siitä seuraavana. ASP.NET Maker on tutkimusta aloitettaessa kirjoittajalle vielä täysin vieras.

5 Menetelmien vertailu

5.1 Aika

Tässä osiossa käydään läpi, kuinka kauan yksinkertaisen sovelluspohjan rakentamiseen menee aikaa kussakin menetelmässä.

ASP.NET

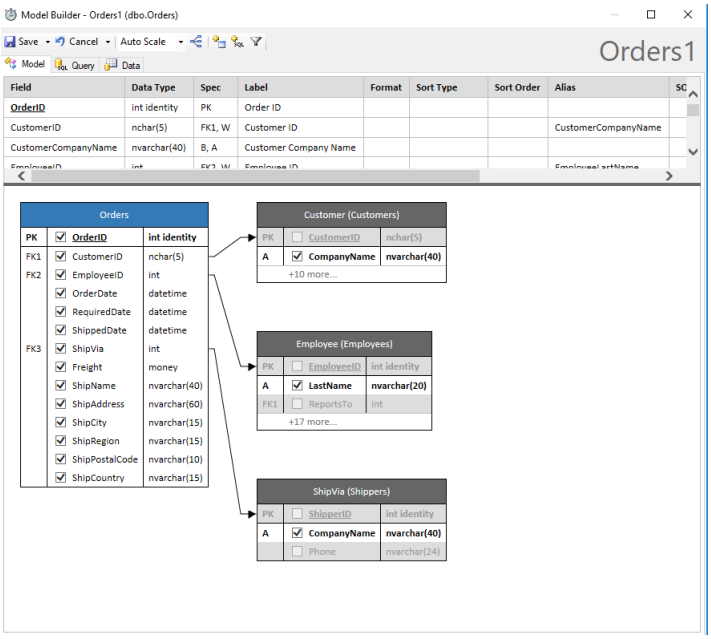
ASP.NET MVC-sovelluksen teko alusta asti itse aloitetaan Visual Studio-kehitysympäristössä. Visual Studiossa osaa luoda rakenteen sovellukselle, kun uutta projektia aloitettaessa valitaan ”ASP.NET MVC Application”. Visual Studio luo kansiorakenteen, sekä websovellukselle tarpeelliset pohjat ja tiedostot, esimerkiksi konfiguraatiotiedostot. Itse käsittelijät ja näkymät, toiminnallisuudet, sekä käyttöliittymän ulkoasu, täytyy kuitenkin luoda itse. (BestDotNetTraining 2015. Youtube)

Yhteys MFSQL tietokantaan täytyy myös tehdä Visual Studiossa. Projektiin on lisättävä Microsoft SQL Serveristä saatava tietokantatiedosto. Tämän jälkeen projektissa on luotava tarvittavat mallit, eli osat, jotka käsittelevät tietokantatauluista haun ja tietokantatauluihin tallentamisen. Visual Studiossa on tätä helpottavia työkaluja. (Microsoft 2013)

ASP.NET sovelluksen tekeminen alusta asti itse vaatii paljon koodaamista, sekä tietämystä siitä mitä tekee. Kyse on päivien työstä, jopa yksinkertaisen tietokantapohjaisen sovelluksen kohdalla. Perusrakenteen, tietokantayhteyden ja toiminnallisuuksien lisäksi aikaa menee käyttöliittymän ulkoasuun ja tietojen näyttämisen asetteluun.

Code On Time

Code On Time -generaattorissa sovellus rakennetaan käyttämällä Code On Time-sovellusta, jonka voi asentaa Code on Time:n omilta sivuilta. Avattaessa ohjelma voidaan luoda uusi projekti. Uutta projektia luodessa yhdistetään aluksi haluttuun tietokantaan (tässä tapauksessa MFSQL tietokantaan Micrososoft SQL serverillä) ja tämän jälkeen ”Model Builder”-osiossa valitaan tietokantataulut, jotka sovellukseen halutaan ottaa mukaan. Mikäli tietokantatauluun ei ole taulujen välisiä suhteita luotu, on se mahdollista Model Builderissä. (Code On Time)

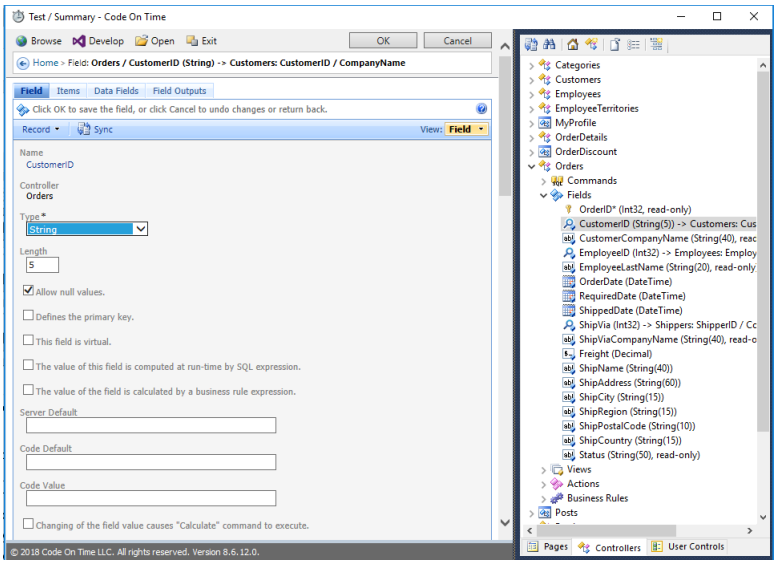


Kuva 3. Esimerkki Model Builder:stä

Generaattori ohjaa läpi koko sovelluksen luontiprosessin (tähän sisältyy teeman valinta, autentikoinnin valinta ja muita ominaisuuksia) ja tämän jälkeen generoi sovelluksen. Tämän jälkeen päästään muokkaamaan käsittelijöitä (controllers) ja näkymiä (views). Generaattori on luonut automaattisesti jokaisesta taulusta käsittelijän ja sivun, jossa on näkymä taulun sisällöstä. Näkymään kuuluu perus listanäkymä. Klikattaessa listan objektia, avautuu lomakenäkymä. Lomakkeelta löytyy muokkaus- ja poistotoiminnallisuudet.

Käsittelijää, sivua ja näkymää on helppo muokata generaattorisovelluksessa. Sovelluksessa voi päättää mitkä taulun tiedot halutaan näyttää, muotoilla taulun tietoja, sekä asettaa mitkä tiedot ovat loppukäyttäjän muokattavissa. Näkymissä voidaan muokata lisäksi, miten master-detail relaatiolla olevia tietoja halutaan näyttää. Code On Time luo automaattisesti myös valitun teeman mukaisen ulkoasun. (Code On Time)

Generaattorilla työskentely ja perustoiminnallisuuksien teko Code On Timella on yksinkertaista ja pitkälle automatisoitua. Sovelluspohjan saa helposti rakennettua pelkästään ymmärtämällä taustalla olevaa tietokantaa, sekä seuraamalla Code On Timen sivuilta löytyviä ohjeita. Tässä kohtaa ei ole tarpeellista katsoakaan taustalla pyörivää koodia tai ymmärtää sitä juurikaan (vaikka Code On Time sellaisen luokin). Yksinkertaisen sovelluksen luomiseen menee siis arviolta puolisen tuntia aikaa.

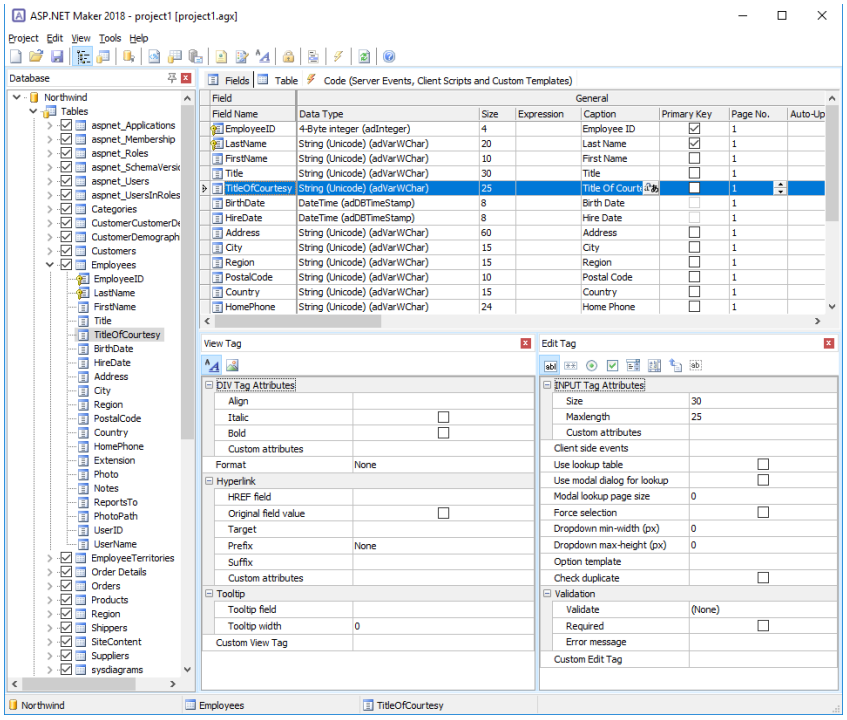


Kuva 4. Code On Time generaattori

ASP.NET Maker

Myös ASP.NET Makerissa sovelluksen luonti aloitetaan sovelluksessa yhdistämällä ensin haluttuun tietokantaan. Tämän jälkeen generaattorissa on mahdollista valita web-sovellukseen näkyviin haluttavat taulut, näkymät ja tiedot. Näiden pohjalta generaattori luo yksinkertaiset taulukkonäkymät, jotka näkyvät luodulla verkkosivulla. ASP.NET Maker myös generoi koodipohjan, joka vastaa hyvin perinteistä ASP.NET MVC rakennetta.

Yksinkertaisen, ulkoasultaan minimalistisen sovelluksen saa Code On Time:n tavoin rakennettua todella nopeassa ajassa. (amr elgarhy 2009, Vimeo)



Kuva 5. ASP.NET Maker generaattori

5.2 Sovelluksen kustomointi

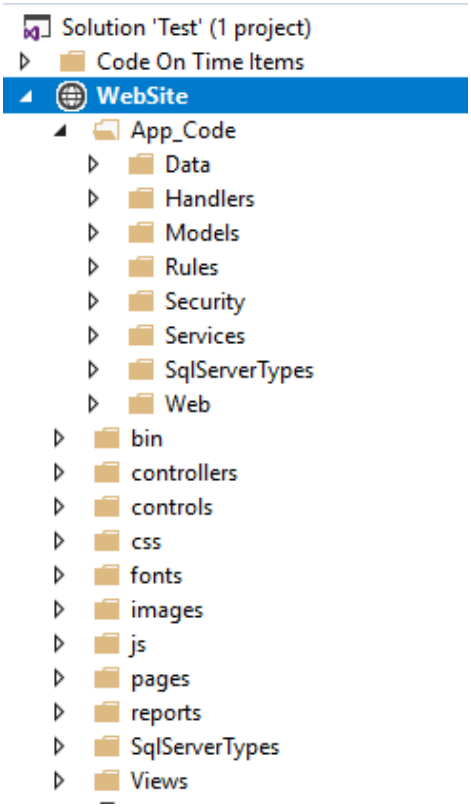
Tässä osiossa käydään läpi jokaisen menetelmän kohdalla, kuinka helposti kustomoitavissa luotu koodi on, sekä miten helppoa kerran luotua on jälkikäteen muuntaa. Osiossa selvitetään myös erityisesti koodigeneraattoreiden kohdalla millaisia rajoitteita sovellus saattaa aiheuttaa.

ASP.NET

ASP.NET MVC koodia alusta asti itse luodessa sovellus on juuri niin kustomoitavissa, kuin kehittäjän omat taidot riittävät. Sovelluksen muutokset ovat mahdollisia missä tahansa osaa koodia, jos vain tietää mitä tekee. Jo valmiiksi tehty koodi on muutettavissa uuteen muotoon missä vaiheessa tahansa. Kustomointiin ei ole valmiita pohjia, mutta ohjeita jonkin verran verkossa. Rajoitteita ei käytännössä ole.

Code On Time

Code On Time generoi koodia, joka on kehittäjän valinnoista riippuen C# tai Visual Basic:lla tehtyä. Koodi on luotu suoraan Visual Studio projektiksi ja sen voi avata Visual Studiossa suoraan generaattorista.



Kuva 6. Code On Time projektin rakenne

Yleisimpiä ja helpoimpia tapoja kustomoida Code On Time-koodia on kirjoittaa palveluntarjoajan omia ohjeita mukaillen business rule:ja, joilla voidaan esimerkiksi laskea kenttien arvoja, asettaa tietojen näkymisille käyttäjärajoitteita, sekä tehdä muita monimutkaisia ehtolausekkeita.

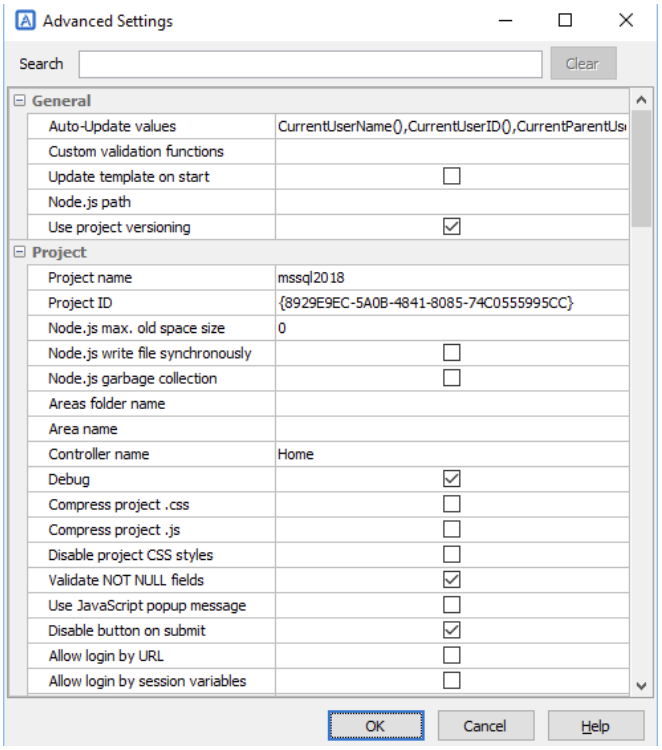
Code On Time on kuitenkin luonut paljon tiedostoja, joita on lähes mahdotonta, tai ainakin monimutkaista muuttaa. Code On Time:n perustoiminnallisuuksien ainaisen toimimisen varmistamiseksi sovellus kirjoittaa yli monia olennaisia tiedostoja jokaisella generointikerralla. Lisäksi on joitakin tiedostoja, joihin Code On Time on lisännyt mahdollisuuden syrjäyttää joitakin toiminnallisuuksia omilla toiminnallisuuksia, mutta tämä vaatii usein uuden yli kirjoittavan tiedoston luontia.

Code On Time:n koodi on siis jonkin verran kustomoitavissa, mutta tietyin sovelluksen määrittelemien rajoitusten mukaan. Code On Time on tehnyt monimutkaisia toiminnallisuuksia, joita ei voi muokata. Tämä käy järkeen, mutta joissakin tapauksissa, joissa sovellusta haluttaisiin kustomoida paljonkin, tämä voi tuottaa ongelmia ja jopa pidentää kehittämiseen menevää aikaa. Code On Time:lla on jonkin verran ohjeita kustomointiin, mutta monimutkaisemmissa tilanteissa vaaditaan kehittäjältä jo paljonkin taitoa, sekä yhteydenpitoa palveluntarjoajan kanssa.

ASP.NET Maker

ASP.NET Maker luo automaattisesti koodit käsittelijöille ja näkymille ja tallentaa ne sovelluksessa ilmoitettuun kansioon. Kaikki tiedostot ovat avattavissa Visual Studiossa, mutta varsinaista Visual Studio-projektia se ei luo. Joitakin osia, kuten html ja css-tiedostoja, voidaan muokata suoraan generaattorissa.

ASP.NET Maker:ssa lisäasetuksissa on paljon koodiin liittyviä asetuksia, jossa voidaan lisätä erilaisia tiedostoja sovellukseen, kuten JavaScript ja Node.js -tiedostoja. (ASPMaker, Tools)



Kuva 7. ASP.NET Maker Advanced Settings

5.3 Autentikointi

Tässä osiossa käydään läpi, millaisia autentikointikeinoja sovellukseen voidaan eri menetelmillä tuottaa.

ASP.NET

Luodessa sovelluksen täysin itse, luonnollisesti on kehittäjän itsensä päätettävissä, minkälaisen autentikoinnin sivullensa haluaa. Kirjautumiskomponenttien teko saattaa ainakin vasta-alkajalle olla monimutkaista ja aikaa vievää. Vaihtoehtoja on kuitenkin monia ja verkosta etsimällä löytää selkeitä ohjeita yleisimpiin toteutuksiin. Esimerkiksi Microsoftin

omista ASP.NET ohjeistuksista löytyy ohjeita ASP.NET Identity:n käyttöön ottamiseen, sekä ohjeita kuinka sovelluksissa voidaan käyttää OAuth tarjoajia, kuten Facebookia tai Googlea. (Microsoft)

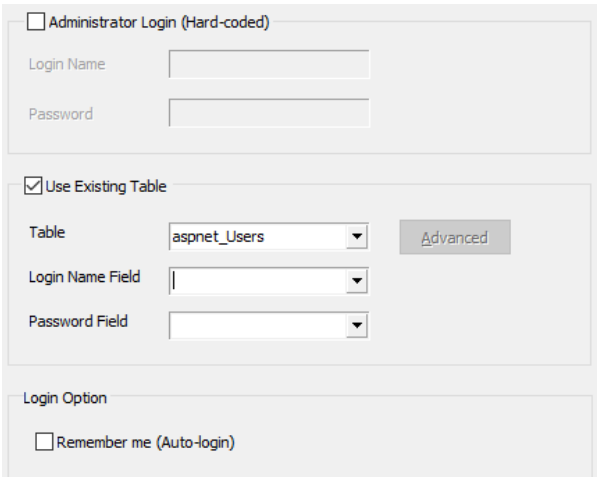
Code On Time

Code On Time:n ilmaisen version mukana tulee mahdollisuus antaa sovelluksen luoda ASP.NET Membership:iin perustuvan autentikoinnin. Tällöin koodigeneraattori luo automaattisesti ASP.NET Membership-taulut tietokantaan, sekä luo järjestelmänvalvoja-roolilla kirjautuville käyttäjille mahdollisuuden luoda, muokata ja poistaa käyttäjätunnuksia käyttöliittymässä. (Code On Time)

Mikäli on valmis generaattorin käytöstä maksamaan, mukana tulee myös mahdollisuus integroida Active Directory, Windows, Google, tai Facebook-autentikointi. Näihin vaihtoehtoihin löytyy myös opastusta Code On Time:n omilta sivuilta. Myös täysin oman kustomoidun autentikoinnin ja kirjautumisen luominen on mahdollista maksullisessa versiossa, jolloin toki ollaan taas omien taitojen varassa. Code On Time itse suosittelee kustomoiduissa autentikointimalleissa käyttämään OAuth-tarjoajien palveluita, mutta täysin omanlaisensa, esimerkiksi REST:iin perustuvan autentikoinnin rakentaminen on mahdollista. (Code On Time)

ASP.NET Maker

ASP.NET Maker:in yksinkertaisin autentikointi perustuu myös tietokantatauluihin. ASP.NET Maker:in Security -asetuksista pääsee valitsemaan haluamansa taulun. Lisäasetuksista pystyy valitsemaan vaihtoehdoiksi myös Windows, tai LDAP autentikoinnin.



Kuva 8. ASP.NET Maker Security -asetukset

Generaattorissa pystyy myös luomaan useampia eri tasoisia käyttäjärooleja. ASP.NET Maker generaattorin valmiit vaihtoehdot ovat kuitenkin melko vähäiset autentikoinnin suhteen. Täysin oman autentikoinnin voi tässäkin tapauksessa itse luoda, mutta ASP.NET Maker:in kanssa ei löydy valmiita ohjeistuksia ainakaan kovin helposti. (ASP.NET Maker)

5.4 Testaus

Parhaiden käytäntöjen mukaan koodin olisi hyvä olla testattavissa yksinkertaisin keinoin. Visual Studiosta löytyy ”Unit Test” -projektilisäosa, joka on mahdollista lisätä mihin tahansa Visual Studio projektiin. Tämä on hyödyllinen tapa lisätä erilaisia testejä ASP.NET projektiin.

ASP.NET MVC projektia alusta asti luodessa tämä on kaikista yksinkertaisinta, sillä kun verkon ohjeita web-sovelluksen luomiseen seuraa, löytyy myös vinkkejä kuinka aina ominaisuuksia projektiin lisätessä lisätään myös testejä näille ominaisuuksille.

Myös Code On Time:n Visual Studio projektiin on mahdollista lisätä testiprojekti, mutta se vaatii hieman lisätyötä, eikä kaikki välttämättä toimi ihan samalla tavalla kuin olettaisi. Sama tilanne on myös ASP.NET Makerin kanssa, joka ei Visual Studio -projektia luo, mutta sellaiseksi se on manuaalisesti mahdollista luoda. Testejä on mahdollista tehdä ja kannattaakin tehdä, mutta ei välttämättä voi luottaa kaiken toimivan täysin samalla tavalla kuin tavallisessa Visual Studio projektissa-

5.5 Muut esille tulleet asiat

ASP.NET-ohjelmointikieli on sen verran iäkäs ja paljon käytetty, että verkosta on hyvin helppoa löytää ohjeistusta ja verkkokursseja, joilla käydään läpi, kuinka ASP.NET MVC sovellus tehdään alusta asti itse. Koodigeneraattoreiden kohdalla toisaalta kokemusta ja artikkeleita, sekä sovellusten käyttäjiä, on yleisesti ottaen vähemmän, joten ohjeiden osalta ollaan pitkälti palveluntarjoajan omien sivujen, ohjeiden ja foorumeiden varassa, jotka joiltain osin eivät ole ajan tasalla. Code On Timella vaikuttaisi olevan kattavammat ohjeet omilla sivuillaan, sekä YouTubessa, ASP.NET Makerilla tarkentavia ohjeita haastavampi löytää. Molemmilla, sekä Code On Timella, että ASP.NET Makerilla on kuitenkin omat asiakastukensa (sähköpostilla), sekä kehittäjille foorumit, joilla voi kysyä apua. Avun saaminen voi kuitenkin olla hidasta kielimuurien ja aikaerojen takia, vaikka käyttäisikin maksullista versiota sovelluksesta.

Toinen huomionarvoinen asia on, että generaattoreita kehitetään koko ajan ja bugeja saattaa ilmetä palveluntarjoajan koodissa, jota ei itse ole mahdollista muokata. Tällöin joudutaan odottamaan uutta julkaisua, joka saattaa hidastaa toimintaa. Päästään taas siihen, että palveluntarjoajan kanssa kirjeenvaihtoon voi mennä paljon aikaa.

6 Pohdinta

Tämän pienimuotoisen tutkimuksen lopputuloksena ainakin kirjoittajan omasta näkökulmasta koodigeneraattoreiden käyttö nopeuttaa ASP.NET projekteissa alkuun pääsemistä. Koodigeneraattoreita käytettäessä säästetään aikaa muun muassa tietokantayhteyden ja käyttöliittymän rakentamisessa. Erityisesti koodigeneraattoreista hyötyvät sellaiset käyttäjät, joilla on vähemmän kokemusta ohjelmoinnista.

Koodigeneraattoreita kannattaa ensimmäisenä käyttää pienimuotoisissa projekteissa, sekä sellaisissa projekteissa, joissa web-sovelluksen ominaisuudet eivät tarvitse olla kovin monimutkaisia. Tämä siksi, että monimutkaisemmat kustomoinnit vaativat monesti pitkääkin tutkimista. Mutta jos aikaa on perehtyä koodigeneraattoreiden, sekä niiden tuottaman koodin kustomointiin tarkemmin, voi koodigeneraattoreista olla hyötyä monimutkaisemmissakin projekteissa. Kun ei tarvitse käyttää alun toiminnallisuuksiin paljoa aikaa, mahdollistetaan, että monimutkaisemmille toiminnallisuuksille jää enemmän aikaa.
Myös näiden monimutkaisempien toiminnallisuuksien rakentaminen voi helpottua koodigeneraattoreiden myötä.

Kahdesta vertaillusta koodigeneraattorista Code On Time jäi mieleen parempana vaihtoehtona jo selkeämpien verkkosivujen avulla, josta ohjeita ja tutorialeja oli helppo löytää. Code On Time:n kohdalla täytyy kuitenkin ottaa huomioon, että kirjoittajalla on ollut kauemmin aikaa perehtyä kyseiseen generaattoriin. Lisäksi kirjoittajalla on henkilökohtaisia kokemuksia Code On Time:n tuen kanssa yhteistyöstä, joka on toiminut hyvin. Valmistumiskiireiden takia ASP.NET Maker:in tutkiminen jäi paljon pintapuoleisemmaksi verrattuna Code On Time:en. ASP.NET Maker:ia olisi voinut tässä työssä tutkia enemmänkin.

Mitä tulee sovelluksen luomiseen alusta asti ASP.NET MVC -ohjelmistokehystä hyödyntäen, on se varmasti suosituin keino vanhempien ohjelmistokehittäjien keskuudesta. Niille, keillä on jo valmiiksi kokemusta ASP.NET -ohjelmoinnista, on helpompi aloittaa alusta asti itse, eikä opetella uuden generaattorin käyttöä. Tällaisillekin käyttäjille generaattorien käyttö voisi kuitenkin olla hyödyllistä aiemmin mainittujen yksinkertaisten sovellusten luomisessa.

Aihetta olisi voinut varmasti tutkia enemmänkin, tutkia muitakin generaattorivaihtoehtoja, sekä testata tarkemmin eri menetelmillä kehittämistä. Aihe on kiinnostava, mutta aikatau-

lun puolesta liitteenä on vain Code On Time:lla luodun demon kuvankaappauksia. Pääpiirteittäin päästiin kuitenkin tavoitteeseen erottaa koodigeneraattorin hyödyt ja rajoitukset Extranet-sivun luomisessa.

Lähteet

Amr elgarhy 2009. ASP.Net Maker intro tutorial. Katsottavissa: https://vimeo.com/2943734. Katsottu: 11.5.2018.

ASP.NET Maker 2018. Luettavissa: http://www.hkvstore.com/aspnetmaker/. Luettu: 8.5.2018.

ASP.NET Maker 2018. Tools. Luettavissa: http://www.hkvstore.com/aspnetmaker/doc/tools.htm. Luettu: 11.5.2018.

BestDotNetTraining 2015. Create your first MVC 5.2 Web Application – Video 1 – MVC Tutorial. Katsottavissa: https://www.youtube.com/watch?v=2C2xJRHy8cM. Katsottu: 10.5.2018.

Code On Time. Getting Started. Luettavissa: https://codeontime.com/learn/getting-started/creating-app. Luettu: 10.5.2018.

Code On Time. Security / Membership & Role Providers. Luettavissa: https://codeontime.com/learn/security/membership-role-providers/overview. Luettu: 20.5.2018.

Code On Time 2018. Luettavissa: https://codeontime.com/. Luettu: 22.4.2018.

Code On Time 2018. Buy. Luettavissa: https://codeontime.com/buy. Luettu: 22.4.2018.

Code On Time 2018. Roadmap. Luettavissa: https://codeontime.com/roadmap. Luettu: 22.4.2018.

Laminin Solutions 2018. MFSQL. Luettavissa: https://lamininsolutions.atlassian.net/wiki/spaces/MFSQL/pages/21200910/Introduction. Luettu: 20.4.2018.

M-Files Developer portal. Luettavissa: http://developer.m-files.com/. Luettu: 15.4.2018.

M-Files M-Files ja tietoturva. Luettavissa: https://www.m-files.com/fi/m-files-security. Luettu: 15.4.2018.

M-Files 2016. M-Files 2015.3 Käyttöopas. Luettavissa: https://www.m-files.com/user-guide/2015.3/fin/. Luettu: 15.4.2018.

M-Files 2017. Koulutusmateriaali: Sessio 1 – metatietorakenteen hierarkia ja käsitteet.

M-Files 2018. Luettavissa: www.m-files.com. Luettu: 15.4.2018.

Microsoft. Visual Studio IDE. Luettavissa: https://www.visualstudio.com/vs/. Luettu: 20.4.2018.

Microsoft 2009. ASP.NET MVC Overview. Luettavissa: https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions-1/overview/asp-net-mvc-overview. Luettu: 20.4.2018.

Microsoft 2010. ASP.NET overview. Luettavissa: https://docs.microsoft.com/en-us/aspnet/overview. Luettu: 20.4.2018.

Microsoft 2013. ASP.NET MVC 4 Models and Data Access. Luettavissa: https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions/hands-on-labs/aspnet-mvc-4-models-and-data-access. Luettu: 10.5.2018.

Microsoft 2018. Get started with the .NET Framework. Luettavissa: https://docs.microsoft.com/en-us/dotnet/framework/get-started/index. Luettu: 20.4.2018.

Microsoft 2018 Introduction to ASP.NET Core. Luettavissa: https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-2.1. Luettu: 21.4.2018.

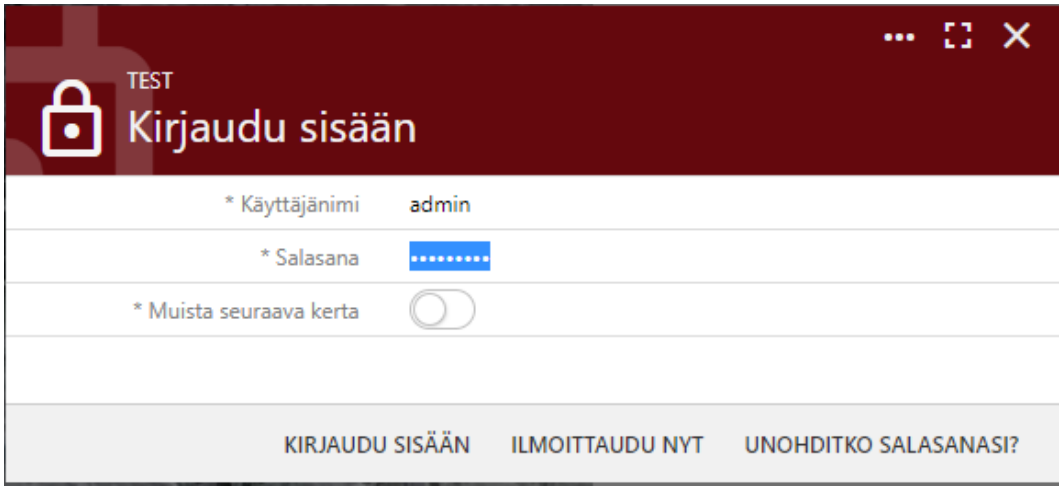
Microsoft 2018. Introduction to Razor Pages in ASP.NET Core. Luettavissa: https://docs.microsoft.com/en-us/aspnet/core/mvc/razor-pages/index?view=aspnetcore-2.1&tabs=visual-studio. Luettu: 21.4.2018.

Techopedia. Code Generator. Luettavissa: https://www.techopedia.com/definition/17062/code-generator. Luettu: 20.5.2018

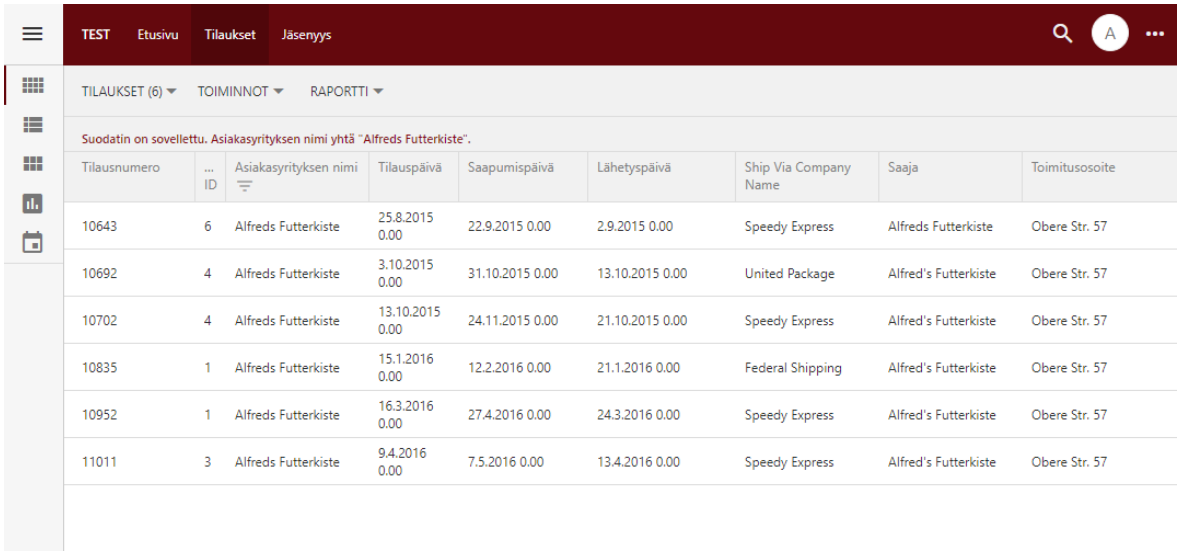
Liitteet

Liite 1. Kuvankaappaukset Code On Time:lla toteutetusta Extranet-sivusta.

Demon data on haettu Northwind -testitietokannasta (Microsoft)

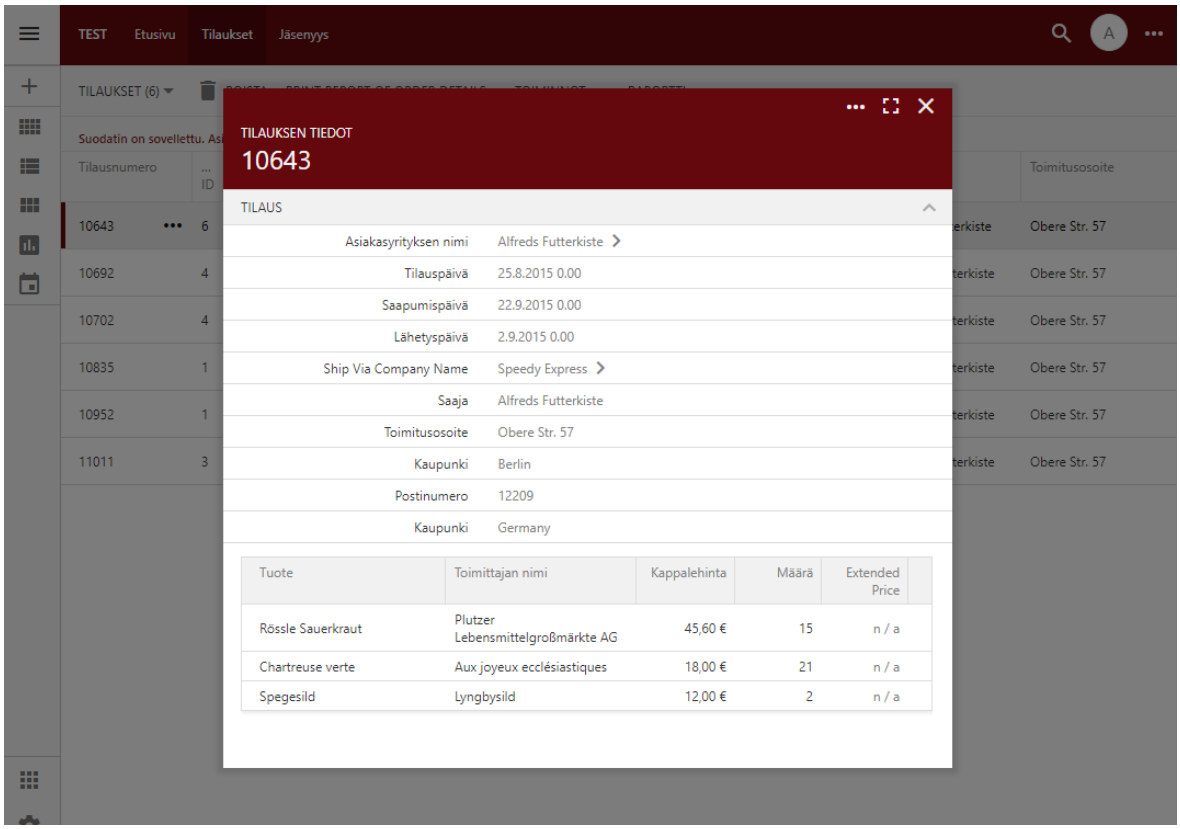


Kuva 9. Kirjautumisikkuna

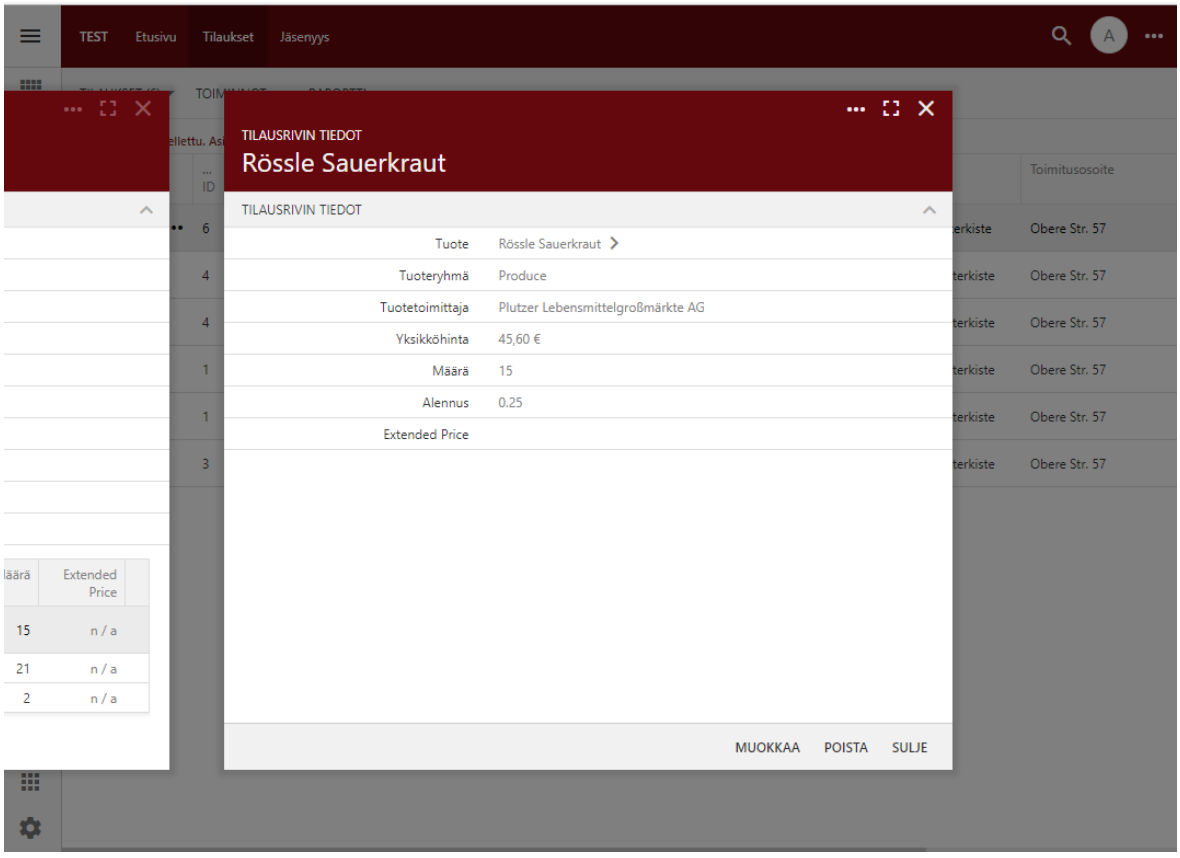


Tilausnumero	... ID	Asiakasyrityksen nimi	Tilauspäivä	Saapumispäivä	Lähetyspäivä	Ship Via Company Name	Saaja	Toimitusosoite
10643	6	Alfreds Futterkiste	25.8.2015 0.00	22.9.2015 0.00	2.9.2015 0.00	Speedy Express	Alfreds Futterkiste	Obere Str. 57
10692	4	Alfreds Futterkiste	3.10.2015 0.00	31.10.2015 0.00	13.10.2015 0.00	United Package	Alfred's Futterkiste	Obere Str. 57
10702	4	Alfreds Futterkiste	13.10.2015 0.00	24.11.2015 0.00	21.10.2015 0.00	Speedy Express	Alfred's Futterkiste	Obere Str. 57
10835	1	Alfreds Futterkiste	15.1.2016 0.00	12.2.2016 0.00	21.1.2016 0.00	Federal Shipping	Alfred's Futterkiste	Obere Str. 57
10952	1	Alfreds Futterkiste	16.3.2016 0.00	27.4.2016 0.00	24.3.2016 0.00	Speedy Express	Alfred's Futterkiste	Obere Str. 57
11011	3	Alfreds Futterkiste	9.4.2016 0.00	7.5.2016 0.00	13.4.2016 0.00	Speedy Express	Alfred's Futterkiste	Obere Str. 57

Kuva 10. Lista tilauksista



Kuva 11. Tilauksen lisätiedot



Kuva 12. Tilaukseen liittyvän tuotteen lisätiedot