Harjoitustyö: Ruokapäiväkirja

Mikko Pakkanen / H8699

1. **Asennus**

   git clone https://github.com/MikPak/IIO13200-ASPNET-OHJELMOINTI.git

   Avaa Visual Studiolla tiedosto IIO13200.sln kloonatun repositorion juuresta, jolloin aukeaa kurssilla suoritetut tehtävät, mukaanlukien harjoitustyö. Aseta *"Harjoitustyo – Ruokapaivakirja"* startup-projektiksi klikkaamalla projektin nimen päältä ja valitsemalla *"Set as startup project"*

   Harjoitustyössä on käytetty *ASP.NET MVC 5 Boilerplatea* (http://rehansaeed.com/asp-net-mvc-boilerplate), joka on tekijänsä sanojen mukaisesti ammattimainen ASP.NET MVC-pohja turvallisten, nopeiden, luotettavien ja mukautuvien web-applikaatioiden luomiseen ja joka tarjoaa minimaalisen määrän koodia Microsoftin MVC-mallin päälle.

   Lisäksi harjoitustyössä on käytetty seuraavia teknologioita: jQuery-javascript kirjasto, Bootstrap-framework, Modernizr-javascript kirjasto, LESS-stylesheet kieli.

   Asennuksessa ei pitäisi tarvita konfiguroida mitään, vaan repositorion kloonaaminen ja .sln-tiedoston avaaminen Visual Studiolla pitäisi riittää. *Entity Framework 6* käyttää paikallista tietokantaa ja tietokannan luonti pitäisi tapahtua ensimmäisen ajon (debug) aikana harjoitustyökansion *App_Data* alle. Visual Studion pitäisi myös asentaa projektin käyttämät NuGet-paketit automaattisesti.

   Projektia ensimmäistä kertaa käynnistäessä (debug) tulee ilmoitus itse allekirjoitettavasta *SSL-sertifikaatista*, johon voi vastata kyllä ja hyväksyä itse allekirjoitettu varmenne.

2. **Mitä tekee ja miksi**

Ruokapäiväkirja on nimensä mukaisesti simppeli sivusto, johon sivuston käyttäjät voivat rekisteröidä tunnuksen, kirjautua sisään ja tallentaa kalenteriin tietoa. Periaatteessa kalenteriin voi laittaa mitä tahansa tietoa, mutta esitysmielessä projekti on nimetty ruokapäiväkirjaksi. Kalenteriin lisätyt tiedot tallennetaan paikalliseen tietokantaan. Käyttäjän kirjautuessa sisään aukeaa kalenterinäkymä ja käyttäjä voi lisätä klikkaamalla lisätä uuden tapahtuman kalenteriin. Jos tietokantaan on tallennettu tietoa, ne myös haetaan kun käyttäjä on kirjautuneena sisään ja lataa sivun, jolla kalenteri on.

Toteutetut toiminnallisuudet eroavat huomattavasti vaatimusmäärittelyssä määritellyistä toiminnallisuuksista, koska tein vaatimusmäärittelyn hyvin nopeasti ja aikaisessa vaiheessa. Harjoitustyön päämäärä myös muuttui hieman sitä tehdessä. Minulla ei myöskään ollut vaatimusmäärittelyä tehdessä vielä selkeätä kuvaa siitä millaisen palvelusta haluan, vaan sekin alkoi hahmottumaan yhä enemmän ja enemmän harjoitustyötä tehdessä.

Vaatimusmäärittelyssä listatuista toiminnoista toteutui kuitenkin *käyttäjien rekisteröinti, käyttäjien autentikointi* sekä *”reseptien” (tässä tapauksessa tiedon) lisääminen kalenteriin ja näyttäminen kalenterissa*.

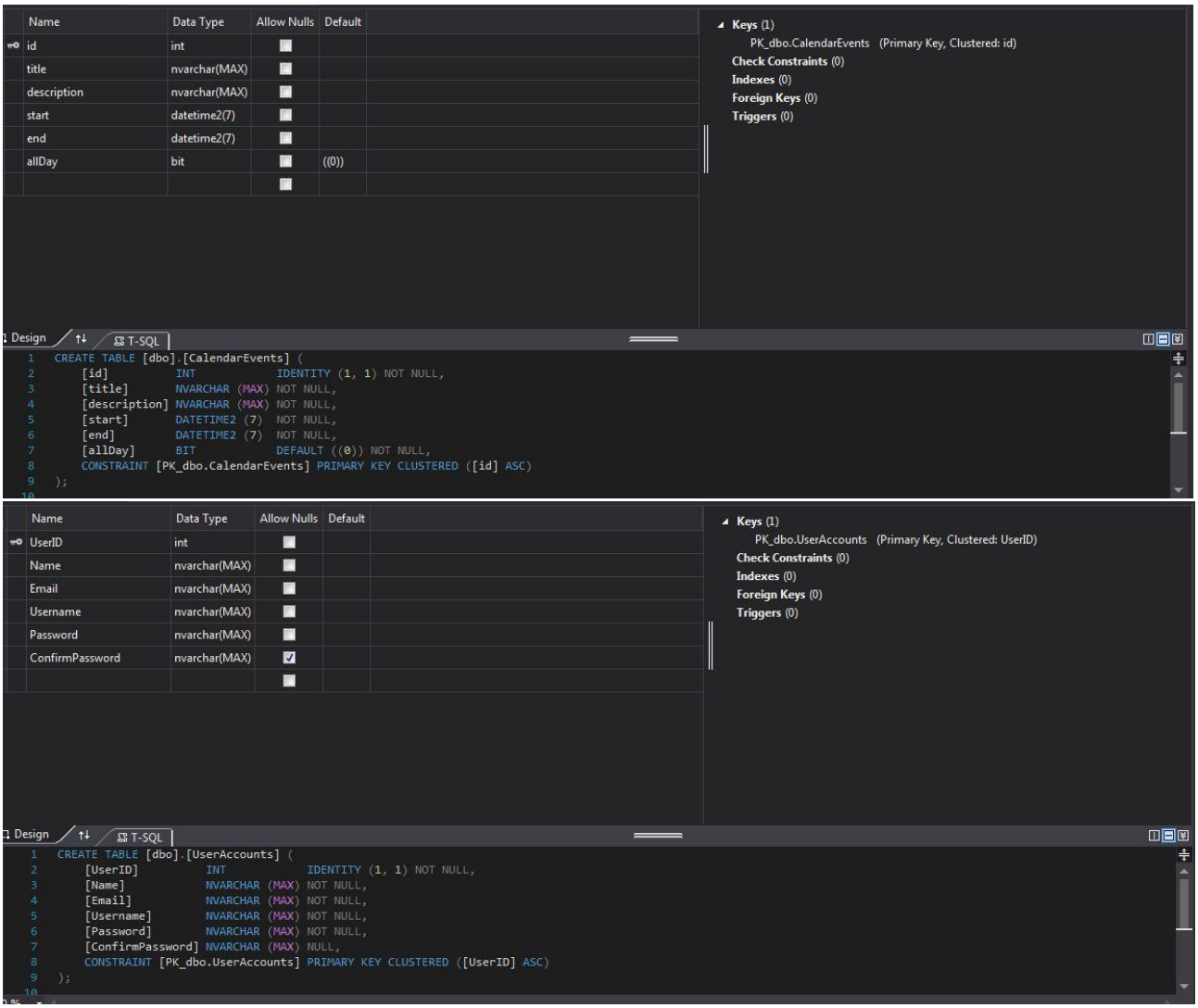
3. **Kuvaruutukaappaukset**
   Lienee ilmiselvää, joten en listaa niitä tässä, sivustolla on käytännössä kaksi näkymää; ns. *laskeutumissivu* / *etusivu* sekä *kalenterinäkymä* kun käyttäjä on kirjautunut sisään.

4. **Ohjelman tarvitsemat /mukana tulevat tiedostot/tietokannat**
   Sivusto käyttää Entity Framework 6:n luomaa paikallista tietokantaa, joka luodaan ensimmäisen ajon yhteydessä harjoitustyökansion *App_Data* kansion sisälle, *.mdf*-formaattiseen tiedostoon.

   Tietokanta sisältää kaksi taulua; *UserAccounts* sekä *CalendarEvents*.

   Tietokantaan pääsee kätevästi käsiksi vaikkapa Visual Studion omalla Server Explorerilla, kun luo sillä uuden yhteyden ja valitsee harjoitustyökansion sisällä olevan *App_Data*-kansion ja sieltä *.mdf*-formaatissa olevan tiedoston.

   Ohessa vielä kuvakaappaukset näistä kahdesta taulusta, joista ilmenee rakenne sekä luontilauseet:



5. Tiedossa olevat ongelmat ja bugit sekä jatkokehitysideat

Bugeja:
Kalenteriin päivän kohdalle tietoa lisätessä tieto menee valitun päivän kohdalle, mutta kun tietokannasta haetaan tietoa, ilmestyy tieto näkyviin kalenteriin yhden päivän aiemmin kuin pitäisi.

Jatkokehitysideat:
Näitä on paljon. Kalenterin yleisen käytettävyyden parantaminen, kalenterin tietojen muokkaaminen, kalenterin tietojen siirtäminen päivältä toiselle, kalenterin tietojen poistaminen, käyttäjäkohtaiset kalenteritapahtumat, ruokareseptien luonti omana luokkanaan ja näiden tallentaminen tietokantaan sekä mahdollisuus valita näitä reseptejä kun lisätään tietoa kalenteriin, ravintoarvojen laskeminen per 100g käyttäen apuna erilaisia

internetistä löytyviä palveluita, jotka listaavat elintarvikkeiden ravintoarvoja, esim. *Fineli.* Mahdollisuus näyttää ja halutessaan käyttää muiden käyttäjien lisäämiä ruokareseptejä. Ja niin edelleen..

6. **Mitä opittu, mitkä olivat suurimmat haasteet, mitä kannattaisi tutkia/opiskella lisää jne**

   Olen oppinut harjoitustyötä tehdessä jonkin verran lisää asioita MVC-mallista sekä paljon asioita ASP.NET-frameworkista, myös jonkin verran tuli opittua Javascriptiä, kun kalenteri on *jQuery:llä* kirjoitettu.

   Mikäli jatkaisin projektia niin luultavasti opiskelisin lisää MVC-mallista sekä ASP.NET-frameworkista sekä jQuerystä.

7. **Tekijät, vastuiden ja työmäärän jakautuminen sekä tekijöiden perusteltu ehdotus arvosanaksi**

   Tekijät: Mikko Pakkanen, ASIO-tunnus: H8699
   Työmäärä: 30-40 tuntia
   Ehdotus arvosanaksi: 5, koska mielestäni käytin harjoitustyössä sellaisia teknologioita, joita ei juurikaan käyty kurssilla läpi; MVC-malli, jQuery sekä ominpäin perehdytty Muhammed Rehan Saeedin luomaan ASP.NET MVC Boilerplateen sekä käytetty sitä projektin pohjana. Ottaen myös huomioon sen, että tein harjoitustyön yksin, olen mielestäni arvosanan 5 ansainnut.