Projektidokumentti

Nimi	Paavo Jännes
Opiskelijanumero	728861
Opinto-ohjelma	Tietotekniikka
Opiskeluvuosi	2

Sovelluksen hakee neljän eri lounasravintoloiden ruokalistat kyseisten ravintoloiden omilta verkkosivuilta ja kokoaa ne yhteen käyttäjän näkyville. Käyttäjä voi ilmoittaa sovellukselle suosikki ruoka-aineita, erityisruokavalioita sekä suosikki ravintolansa ja sovellus suosittelee näiden mukaan lounaita eri ravintoloista. Ohjelma toimii tekstipohjaisella käyttöliittymällä tekstieditorissa, joten projektin vaikeusaste on helppo.

Ohjelman saa käyntiin ajamalla UI -tiedoston. Tämän jälkeen ohjelma lataa käyttäjän näkyville neljän ravintolan senhetkisen lounaslistan käyttäjän antamien tietojen mukaan. Komentoja on yhteensä viisi kappaletta ja ne syötetään kirjoittamalla komento tekstieditorin terminaaliin.

help	kertoo eri komennot ja niiden toiminnan
close	sulkee ohjelman
edit	komennolla päivitetään käyttäjän tietoja
update	päivittää ravintoloiden ruokalistat UI:ssa
info	antaa tietoja ravintoloista

Ohjelma voidaan jakaa kolmeen osaan: ruokalistojen hakeminen netistä, käyttäjätietojen hankkiminen ja tallennus sekä ruokalistojen järjestyksen määrittäminen. Varsinaisia luokkia ohjelmassa on tasan yksi, ja luokan ainoalla metodilla p() tulostetaan ravintoloiden ruokalistoja käyttäjän näkyville. Suurin ongelma ohjelman tekemisessä oli käsitellä kaikkien ravintoloiden ruokalistoja samalla tavalla ja Menu -luokka tarjoaa tähän ratkaisun generalisoimalla kaikki ruokalistat samanlaisiksi. Näin esimerkiksi ruokalistojen tulostus helpottuu huomattavasti. Käyttöliittymälle ei ole erillistä luokkaa, vaan käyttöliittymä on itsessään pelkkä funktio. Toinen vaihtoehto luokkajaolle olisi ollut tehdä käyttäjälle oma luokka ja täten mahdollistaa esimerkiksi useamman käyttäjän luomisen sovellukselle. Toisaalta sovellusta käytetään konsolin avulla ja näin ollen jokainen eri laite, jolla sovellusta käytetään, on oma yksittäinen käyttäjä. Realistisessa käyttötilanteessa käyttäjä katsoo ruokalistat omalta laitteeltaan, eikä näin ollen useampaa käyttäjää tarvita. Sovelluksen

käyttöliittymä toteutetaan konsolissa ja komennot käsitellään match - case rakenteella, ja näin ollen käyttöliittymä ei tarvitse esimerkiksi luokkia graafisille elementeille tai komennoille. (UML kaavio dokumentin lopussa)

Ohjelmassa käytetään algoritmia ruokalistojen järjestyksen määrittämiseen. Ruokalistojen järjestykseen vaikuttavat käyttäjän antamat tiedot ja näitä verrataan annosten sisältämiin sanoihin ja merkkeihin. Jokainen ohjelman ravintola ilmoittaa annoksen diettikoodit (G, L, V…) ja algoritmi käy nämä läpi vertaamalla niitä käyttäjän ilmoittamiin ruokavalioihin. Kun annos sisältää käyttäjän ilmoittaman merkinnän, algoritmi antaa annokselle yhden -1 pisteen. Sama toistetaan käyttäjän ilmoittamien suosikkiruoka-aineiden kohdalla, ja niitä verrataan annoksen sisältämiin sanoihin. Lopulta algoritmi palauttaa (-) -merkkisen luvun, jonka avulla ohjelma järjestää ruokalistat käyttäjän näkyville, mitä pienempi lukema sitä paremmin ravintolan ruokalista sopii käyttäjälle. Algoritmin ongelma on, että se ei valitettavasti aina tunnista kaikkia merkkejä tai sanoja ja näin ollen algoritmin tulos ei ole täysin tarkka. Algoritmi kaikesta huolimatta kuitenkin toimii riittävällä tasolla ohjelman funktionaalisuuden kannalta.

Ohjelma käyttää pääasiassa scala.mutablen Buffereita väliaikaisen tiedon tallentamiseen. Lisäksi käytössä on funktioiden yhteisiä muuttujia, joiden avulla helpotetaan funktioiden välistä toimintaa. Bufferi valikoitui listojen muodoksi sen joustavuuden sekä monikäyttöisyyden johdosta. Joustavat listatyypit helpottivat ohjelman tekemistä sekä jälkikäteen muokkaamista. Toinen vaihtoehto olisi ollut selvittää tehokkuuden kannalta optimaalisin listatyyppi jokaiseen eri käyttötarkoitukseen, mutta koska ohjelma itsessään on melko simppeli ja kuormittaa konetta suhteellisen vähän, yksinkertainen listatyyppiratkaisu ajaa asiansa loistavasti.

Ohjelma käsittelee sekä HTML -tiedostoja verkossa että userData.txt -tekstitiedostoa lokaalisti. Tekstitiedostossa sijaitsee käyttäjän antamat tiedot yhdellä rivillä muodossa lempiravintola;lempi raaka-aineet;allergiat/ruokavaliot. Eri raaka-aineet ja ruokavaliomerkinnät erotetaan toisistaan pilkuilla. Käyttäjän ei tarvitse luoda erikseen tekstitiedostoja, vaan tiedosto on projektissa valmiiksi ja samaa tiedostoa käytetään aina uudelleen. Verkossa olevat HTML -tiedostot käsitellään scala-scraper -kirjaston avulla ja kirjaston funktioilla kerätään haluttu data verkosta ohjelman käyttöön.

Ohjelman testaus toteutettiin empiirisesti ajamalla ohjelmaa sekä yksittäisiä funktioita, eikä erillistä yksikkötestausta tehty. Sovellusta tehdessä ei tullut kertaakaan eteen sellaista tilannetta, jossa olisin kokenut tarvitsevani erillisiä yksikkötestejä jonkin bugin tai ongelman ratkaisemiseksi. Ohjelman testaus toteutettiin siis täysin kuten olin suunnitellut: testejä kirjoitetaan mikäli erillistä tarvetta niille ilmenee ja muutoin testaus tapahtuu ajamalla ohjelmaa.

Ohjelmassa on tiedossa tällä hetkellä kaksi bugia. Ruokalistojen järjestyksen määrittävä algoritmi ei tunnista kaikkia merkkejä tai sanoja sekä se myös huomioi isot kirjaimet esimerkiksi ruoka-annoksen nimessä. Esimerkiksi Lohikeiton iso L kirjain tulkitaan laktoosittoman ruokavalion merkiksi ja näin ollen se saattaa vaikuttaa ruokalistojen järjestykseen. Toinen bugi liittyy Sodexon Valimon ravintolaan. Ohjelmaa tehdessä kyseinen ravintola ilmoitti sivuillaan joka päivälle kaksi ateriaa, mutta nyttemmin annosten määrä vaihtelee satunnaisesti. Ohjelma on ravintolan kohdalla koodattu siten että annoksia olisi 2kpl/päivä, joten kyseisen ravintolan kohdalla päiväkohtaiset annokset eivät välttämättä tulostu oikein.

Ohjelman kolme parasta puolta ovat mielestäni käyttäjätietojen tallennus tekstitiedostoon, ruokalistojen tulostus Menu -luokan avulla sekä hyvin yksinkertainen, mutta toimiva käyttöliittymä. Käyttäjätietojen tallennus tekstitiedostoon tekee ohjelmasta miellyttävämmän käyttää ja on muutenkin mielestäni vaativampi ja parempi ratkaisu kuin käsitellä käyttäjätietoja pelkästään listoilla. Menu -luokan avulla tulostaminen yksinkertaistaa ohjelmaa merkittävästi ja käyttöliittymän yksinkertaisuus on vain yleisesti miellyttävää. Ohjelman huonot puolet ovat aiemmin mainittu epätarkasti toimiva algoritmi sekä gatherer -tiedostossa tapa, jolla ohjelma lisää listaan tämänpäiväisen ruokalistan. Kummassakin on paljon toisteisuutta ja jonkinasteista epävarmuutta toimivuudessa. Lisäksi ohjelma hakee joka kerta uudelleen ruokalistat verkosta, mikä hidastaa ohjelman toimintaa. Tämän olisi voinut ratkaista esimerkiksi tallentamalla verkkosivujen HTML -tiedoston lokaalisti ja käyttää sitä ruokalistojen hakemiseen.

Ohjelma tehtiin suunnilleen suunnitellussa järjestyksessä, eli ensimmäisenä tehtiin käyttöjärjestelmä sekä funktiot ruokalistojen hakemiseen verkosta. Tämän jälkeen tehtiin järjestelmät käyttäjätietojen tallentamiseen ja lopuksi tehtiin nämä kaksi osa-aluetta yhdistävät tekijät, kuten Menu -luokka ja editor -tiedoston funktiot. Projekti eteni erittäin jouhevasti ja työtunneissa mitattuna sain kaiken tehtyä suunniteltua nopeammin. Työläin vaihe oli mainituista viimeisin, johon kului yhteensä noin 20 tuntia. Yhteensä projektiin meni noin 40 tuntia, kun suunnitteluvaiheessa arvioin työmäärän olevan noin 73 tuntia.

Kokonaisuudessaan ohjelma on mielestäni hyvä ja toimii kuten on tarkoituskin. Ohjelmassa on käytetty useita mielestäni fiksuja ratkaisuja, kuten käyttäjätietojen tallennus tekstitiedostoon sekä ruokalistojen tulostus Menu -luokan avulla. Toisaalta järjestysalgoritmia voisi parantaa toimivammaksi ja gatherer -tiedoston funktioita muokata hieman vähemmän toisteisiksi. Mielestäni sovellus täyttää kaikki ennalta määrätyt kriteerit, eikä sovellusta tehdessä olla menty sieltä mistä aita on matalin. Sovellukseen voisi suhteellisen helposti lisätä ominaisuuden yksittäisen ravintolan ruokalistojen katsomiselle viikkotasolla tai ylipäätään lisätä lisää ravintoloita sovellukseen. Sovellusta tehdessä pyrin tekemään funktioista mahdollisimman abstrakteja ja uskoisin että ohjelmaan olisi helppo lisätä uusia ominaisuuksia pieniä muokkauksia tekemällä.

Jos pitäisi tehdä jotain toisin, lähtisin projektiin kovemalla itseluottamuksella ja tekisin sovellukselle myös graafisen käyttöliittymän. Mielestäni sovellus on kokonaisuudessaan onnistunut ja täyttää annetut kriteerit.

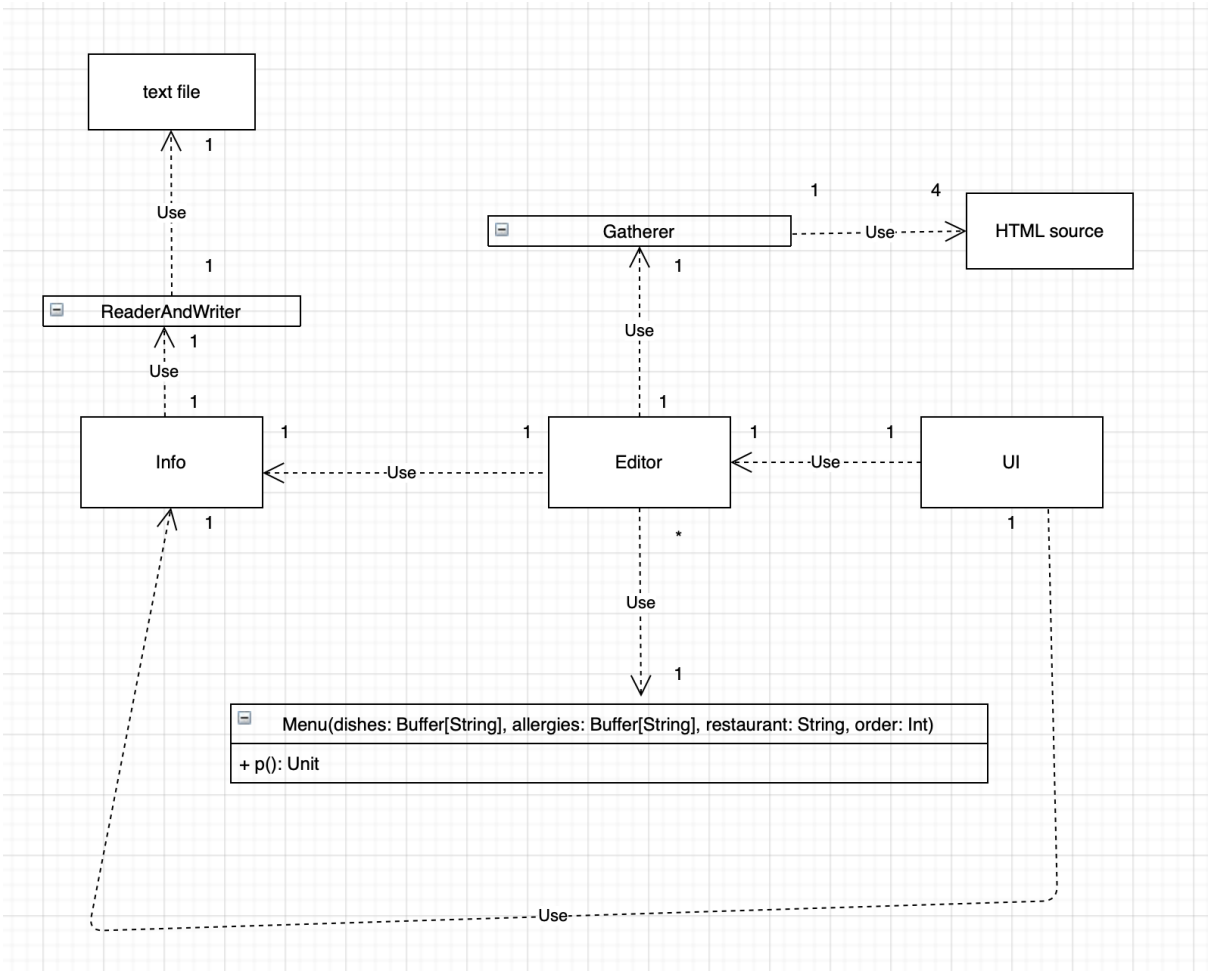
Lähteet

Scala-scraper https://github.com/ruippeixotog/scala-scraper

Päivämäärän saaminen ohjelman käyttöön
https://stackoverflow.com/questions/44089758/in-scala-how-to-get-the-day-of-the-week

Jsoup -selain https://jsoup.org

UML




```
Command: info
Sodexo Alppikortteli, Alppikatu 2 00530 Helsinki, Lunch 10:15-13:30 Mon-Fri
Sodexo Valimo, Metallimiehenkuja 2 02150 Espoo, Lunch 10:30-13:30 Mon-Fri
Mau-Kas, Vuorimiehentie 5 02150 Espoo, Lunch 10:30-14:00 Mon-Fri
Palmia Kaupungintalo, Sofiankatu 1-3 00170 Helsinki, Lunch 10:30-13:30 Mon-Fri

Command:
```

```
Type 'help' for list of commands
Command: help
Command 'edit' -> edit preferences, diets and the favourite restaurant
Command 'update' -> refreshes the view
Command 'close' -> closes the app
Command 'info' -> information about the restaurants

Command: edit
Tell your preferences by answering the following questions.
The app will save the given information so you don't need to do this every time you use the app :)
Information is used for ordering the menus so the app will show you first the menus which fits your preferences the most.
Favourite restaurant will always be the first restaurant in the listing
Information has to be given in a correct form.

What's your favourite restaurant?: (Sodexo Alppikortteli, Sodexo Valimo, Palmia Kaupungintalo, Mau-Kas) Mau-Kas
Which food items do you prefer?: (separarate the items with a comma, for example kana, kanaa, riisiä) kana
Do you have allergies or some special diet?: (separarate the items with a comma and use diet codes 'G', 'V', 'L' etc., for example G, L, V) G, L

Saving...

Saved successfully!

Loading menus...
```

```
Mau-Kas FAVOURITE
Cornflakeskanaa ja currymajoneesia / Cornflakes chicken with curry mayo M, KA, VS, *
Ruusukaali-aurajuustovuokaa saksanpähkinöillä / Brussels sprout & blue cheese stew with walnuts G, L, *
Bataattisosekeittoa / Sweet potato pure soup G, L
Teryakitofusalaattia / Teryaki tofu salad G, M, SO, VS
Raparperipaistosta / Rhubarb bake L, *


Palmia Kaupungintalo
Varhaiskaali-jauhelihalaatikkoa M G L, puolukkaa
Yrttivoikastikkeessa haudutettua kampelaa L G
Kaupungintalon pinaattiohukaisia L, pyydä VEG
Pähkinäistä valkosipulikeittoa L G
Fetajuustosalaattia L G
Päivän jälkiruokaherkku kahvin ja teen kera


Sodexo Alppikortteli
Jauheliha Bolognese kastiketta ja penne pastaa L
Punajuuri-soijagratiinia G


Sodexo Valimo
Nakkistroganoff ja höyrytettyä perunaa G L
Vegaani Soija - kasviswokkia ja höyrytettyä riisiä M


Command:
```

```
Hello, 28-4-2021 menus are:

Loading menus...

Sodexo Alppikortteli
Jauheliha Bolognese kastiketta ja penne pastaa L
Punajuuri-soijagratiinia G


Palmia Kaupungintalo
Varhaiskaali-jauhelihalaatikkoa M G L, puolukkaa
Yrttivoikastikkeessa haudutettua kampelaa L G
Kaupungintalon pinaattiohukaisia L, pyydä VEG
Pähkinäistä valkosipulikeittoa L G
Fetajuustosalaattia L G
Päivän jälkiruokaherkku kahvin ja teen kera


Mau-Kas
Cornflakeskanaa ja currymajoneesia / Cornflakes chicken with curry mayo M, KA, VS, *
Ruusukaali-aurajuustovuokaa saksanpähkinöillä / Brussels sprout & blue cheese stew with walnuts G, L, *
Bataattisosekeittoa / Sweet potato pure soup G, L
Teryakitofusalaattia / Teryaki tofu salad G, M, SO, VS
Raparperipaistosta / Rhubarb bake L, *


Sodexo Valimo
Nakkistroganoff ja höyrytettyä perunaa G L
Vegaani Soija - kasviswokkia ja höyrytettyä riisiä M
```