COMP.CS.140, Group2788, Sisu-projektin dokumentaatio

- Antti Nylund, 150487311
- Joonas Kärnä, 50218075
- Kalle Pakarinen, 050036943

1. Ohjelman toiminta ja ominaisuudet

1.1 Toteutettu ohjelma

Toteutettu ohjelma on yksinkertaistettu versio usean yliopiston käyttämästä Sisu-järjestelmästä. Ohjelmalla käyttäjä voi tarkastella saatavilla olevia tutkinto-ohjelmia sisältöineen, yliopiston tarjoamia kursseja, sekä seurata omien opintojensa etenemistä.

1.2 Ohjelman kulku

Ohjelman käynnistyttyä ensimmäinen näkymä käyttäjälle on kirjautumisikkuna. Käyttäjä pääsee kirjautumaan opiskelijanumerollaan sisään. Onnistuneen kirjautumisen jälkeen käyttäjälle avautuu pääikkunan Student info välilehti. Välilehdessä käyttäjä pääsee selaamaan vaihtoehtoisia tutkinto-ohjelmia ja valitsemaan tutkinto-ohjelman. Valittua tutkinto-ohjelmaa voi vaihtaa milloin tahansa pääikkunassa. Edistymisen tarkastelu onnistuu myös tässä välilehdessä. Välilehteä vaihtamalla päästään Courses and structure-näkymään, jossa tukinnon rakenteen (puunäkymän) voi tulostaa ja sitä voi tarkastella. Kursseja voi lisätä omaan tutkintoonsa ja viemällä kursorin kurssinimen päälle, tulee näkyviin lisätietoja kurssista. Käyttäjä voi kirjautua ulos Logout-napista, jolloin tehdyt muutokset tallentuvat seuraavaa käyttökertaa varten, ja näkymä siirtyy takaisin kirjautumisikkunaan. “Complete course” nappia painamalla käyttäjä voi valitsemansa kurssinsa muuttaa suoritetuksi. Ohjelman suoritus päättyy Exit-nappia painamalla.

1.3 Ohjelman ominaisuudet

Perusominaisuudet

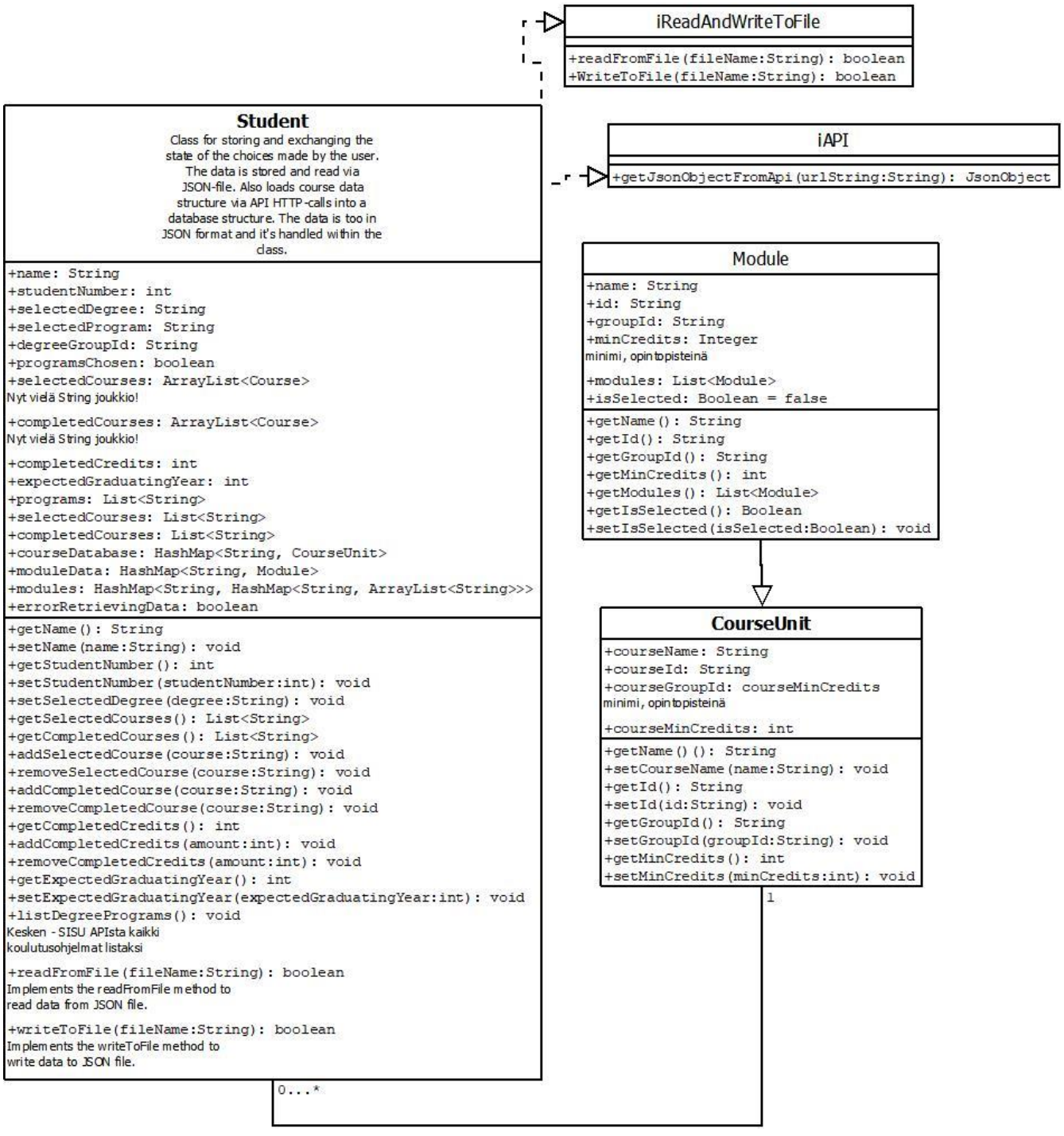
- Ohjelmaan on toteutettu omatoimisesti graafinen käyttöliittymä.
- Ohjelma hakee rakennetiedot Sisun API:sta.
- Näytettävää tutkintoa voi vaihtaa ja opiskelijan tutkinnon tilanne esitetään ohjelmassa.
- Ohjelma selviää hallitusti asetustiedostojen käsittelyssä tapahtuvista virheistä.
- Ohjelmalle on toteutettu yksikkötestejä.

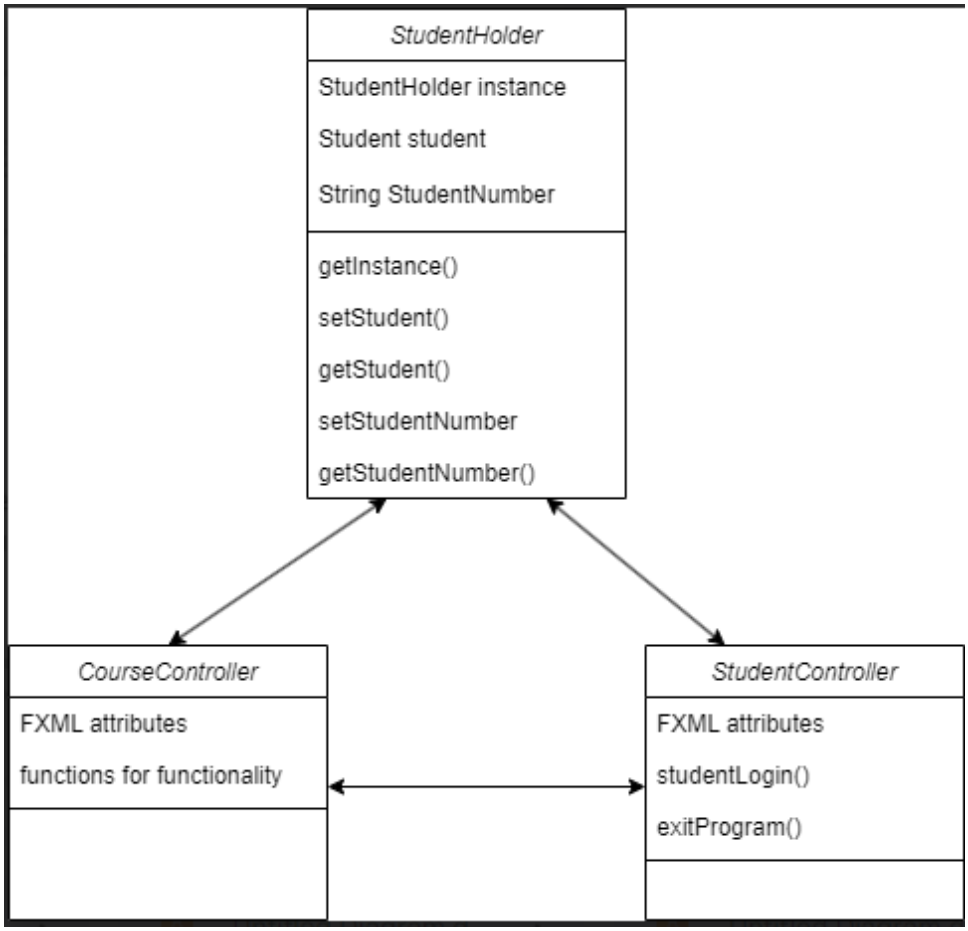
Toteutetut lisäominaisuudet

- Ohjelmassa on kaksi ikkunaa: kirjautumis- sekä pääikkuna.
- Kurssista näytetään lisätietoja näytöllä, kun hiiren kursori viedään kyseisen kurssin yläpuolelle.
- Opiskelijan tietoja voidaan muokata asennus välilehdeltä kirjautumisen jälkeen.
- Css tiedostosta on haettu grafiikka asetukset ohjelmaan.

2. UML-luokkakaavio

Kaavio on kaksiosaisena, koska luokkien toteuttajat tekivät omat kaavionsa omista luokistansa.





3. Luokkien vastuujako

Käyttöliittymään liittyvä toiminallisuus koostuu kolmesta luokasta. CourseController-luokka vastaa käyttöliittymässä pääikkunan toteutuksesta, sekä ajon aikaisesta hallinnasta, StudentController-luokka vastaa kirjautumisikkunasta. StudentHolder luokka siirtää olion tietoja instanssina toiselta controllerilta toiselle.

Tietojen tallentamista varten tehtiin Module-luokasta periytyvät luokat CourseUnit, DegreeModule ja DegreeProgramme, joihin oli tarkoitus talletaa kursseihin ja moduuleihin liittyvät tiedot (esim. Opintopisteet, kuvaus, yms.), joskin CourseUnit oli ainoa luokka joka ehdittiin toteuttamaan.

Suurin toiminnallisuus on totetutettu Student-luokkaan, se vastaa JSON datan hakemisesta SISU:n API:sta, käyttäjän tilan hallinnoimisesta suorituksen aikana sekä tilan tallentamisesta ulos kirjauduttaessa.

4. Työnjako

Työtä on jaettu käyttäen Trello-taulua ja pitämällä viikottaisia Teams palavereja. Trellossa ollaan kommentoitu kortteja ja lisäksi ollaan pidetty yhteyttä Telegramilla matalalla kynnyksellä.

Välillä on ollut parikin palaveria viikossa. Linkki Trello-tauluun: https://trello.com/invite/b/XQUYY7zT/ATTIc7e06ab44a87ed561218300c395030616EFD0C73/sisu-projekti

Antti on työstänyt Student luokkaa ja vastannut siis Json datan luvusta SISU:n API:sta ja luokan toteutuksesta. Myös Student olion talletuksesta JSON muodossa .json tiedostoon ja sen avaamisesta takaisin luokan olioksi.

Kalle on toteuttanut luokkarakennetta, sairastuminen viimeisellä viikolla vähensi työmäärää.

Joonas on työskennellyt GUI:n parissa ja ohjelmien toiminnaliisuksien toteuttamisessa sinne.

5. Käyttöohje

Tiedostosta löytyy vähintään käyttäjän opiskelijanumero jolla käyttäjä kirjautuu sisään. Valmiina tiedostossa käyttäjä numerolla 4. Kirjauduttuaan hän täydentää tietojaan ja valitsee tutkinnon valittavien olevien listasta. Sitten hän voi edelleen valita opintosuunnan.

Toisella sivulla listataan kurssit ja niiden moduulit puumaisessa hierarkisessa tasossa. Sieltä käyttäjä voi merkitä kursseja jo suoritetuiksi. Tai lisätä kurssejaan valittuihin kursseihinsa.

Kirjauduttaessa ulos tilanne tallentuu JSON tiedostoon ohjelman juureen, ja on luettavissa sieltä uudelleenkirjauduttaessa samalla tunnuksella.

6. Tiedossa olevat ongelmat tai puutteet

JSON dataa ei haeta suoraan luokkaan, vaan parsitaan raakadataa: Module luokasta periytyvä CourseUnit ei hae halutusti JSON dataa luokkaan menetelmällä gson.fromJson(jsonObject, Module.class); (aika loppui yrittäessä).

Lisäksi kurssipuun hakeminen on hidas prosessi, koska kurssidataa haetaan kurssiolioihin erikseen API:sta. API luku kannattaisi toteuttaa ehkä kerralla nopeuden puolesta, nyt ei ehditty optimoida.

Kurssitiedoista luodaan oliot ja niihin tietoja, mutta moduuleista ei, ne kulkevat ohjelmassa Student luokan säiliöissä tekstimuotoisina niminä.