

Kyykkäjengi: Eero Hakanen, Joona Nieminen & Tommi Poikolainen

# **DOKUMENTAATIO**

## COMP.CS.140 projektin dokumentaatio

# SISÄLLYSLUETTELO

1.	KÄYTTÖOHJE .....	1
2.	OHJELMAN RAKENNE .....	1
3.	OHJELMAN TOIMINTA JA TOTEUTETUT LISÄOMINAISUUDET .....	2
4.	SOVITTU JA TOTEUTUNUT TYÖNJAKO .....	4
5.	ONGELMAT JA PUUTTEET .....	4

# 1. KÄYTTÖOHJE

Ohjelman käynnistyessä ruudulle aukeaa kirjautumisikkuna. Mikäli ikkunassa ilmenevä kieli ei miellytä, saa sitä vaihdettua napista, jossa lukee ”fi”. Syötä tietosi asianmukaisesti kenttiin ja siirry ohjelmassa eteenpäin painamalla kielestä riippuen Save/Tallenna -nappia.

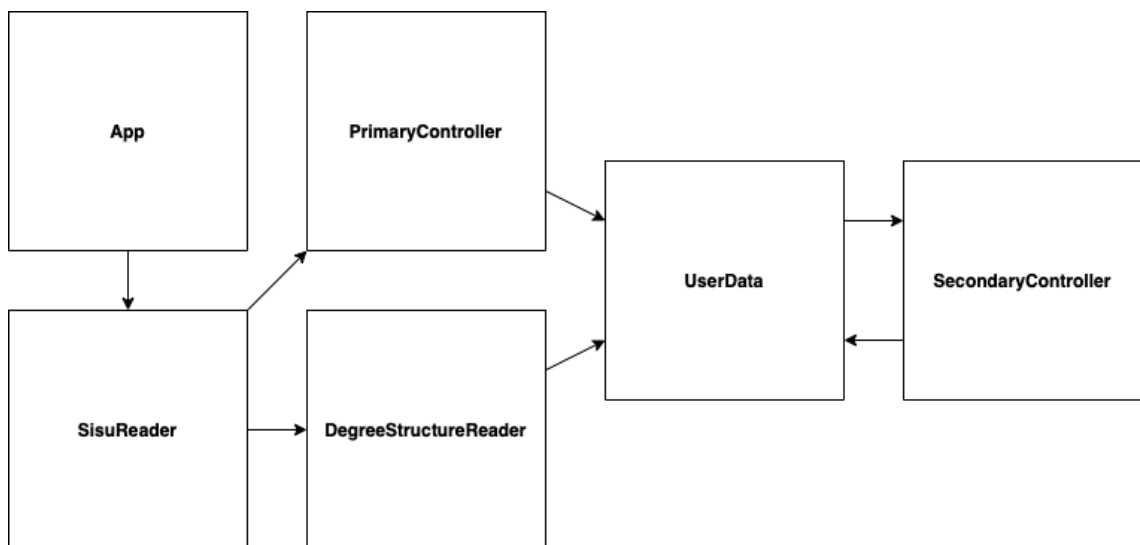
Hetken lataamisen jälkeen eteesi avautuu kurssivalintäkymä. Näkymä koostuu puusta, jossa pääsee etenemään eteenpäin aina tuplaklikkaamalla haluttua osaa. Halutun moduulin/kokonaisuuden avattuaan kursseja voi merkata suoritetuksi tuplaklikkaamalla sitä. Kurssi ilmestyy oikealla olevaan suoritettujen kurssien ikkunaan. Vastaavasti suoritettua kurssia vasemmassa ikkunassa tuplaklikkaamalla voi poistaa tämän kurssin suoritettujen listasta.

Kurssisuoritusten käsittelyn jälkeen voi palata kirjautumisikkunaan Back/Takaisin -napista. Takaisin kirjautumisikkunaan päästyä voit jälleen syöttää asianmukaisesti ikkunoihin jonkun muun opiskelijan tiedot tai kirjautua samoilla tunnuksilla sisään ja jatkaa omien tietojesi muokkaamista.

# 2. OHJELMAN RAKENNE

Ohjelman rakenne koostuu kuudesta luokasta. Alla oleva kuva havainnollistaa yksinkertaistetusti päällepäin luokkien suhteita. App-luokka käynnistää ohjelman ja hakee heti Sisureaderilla mahdolliset tutkinnot ensimmäisen ikkunan tutkintolaatikkoon. PrimaryController muodostaa ensimmäisen ikkunan ja suorittaa tietojen käsittelyn UserData-luokan kautta.

DegreeStructureReaderia käytetään hyväksi Sisusta vedetyn tutkintotiedoston avaamiseen ja käsittelyyn. Luokan avulla muodostetaan SecondaryControlleriin asianmukainen tutkintopuu. SecondaryControllerista palatessa tallennetaan vielä UserDatan tiedot ulkoiseen json-tiedostoon.



Rakenne haluttiin jakaa tällä tavoin juuri näihin luokkiin, sillä tällä tavoin ohjelmassa tapahtuvat asiat pysyvät selkeinä. On helppoa tietää, että missä luokassa tapahtuu ja mitä asioita.

### 3. OHJELMAN TOIMINTA JA TOTEUTETUT LISÄ-OMINAISUUDET

Toteutettuja lisäominaisuuksia on kaksi:

1. Kielen valinta: Ohjelman sekä kurssien ja tutkintojen kielet vaihtuvat toiseen tarvittaessa ensimmäisessä ikkunassa kielestä riippuen, joko "fi" tai "en" nappia painamalla.
2. Opiskelijadatan tallennus ja luku: Ohjelma tallentaa opiskelijan tiedot JSON-tiedostoon suljettaessa ja valikkoikkunasta kirjautumisikkunaan palatessa. Mikäli kirjautumiseen käytettyä opiskelijanumeroa vastaavan niminen tiedosto löytyy, tiedostosta luetaan suoritettut kurssit näkyviin ohjelmaan, jolloin niitä voidaan käsitellä lisää.

Ohjelman toiminta pohjautuu tietenkin emoluokkaan App. Luokan sisällä avataan/käynnistetään ikkuna ja tuodaan kaksi fxml-tiedostoa sekä yhden css-tiedoston mukaan koodin pariin. Ohjelma avaa ensin primary-ikkunan, eli kirjautumisikkunan. Tämän kautta ohjelman suoritus siirtyy luokkaan PrimaryController.

Luokan avulla käsitellään syötekenttiä, kielen vaihtamista sekä tutkintojen lisäämistä tutkintosityöteeseen. Tutkinnon syöttämistä helpottamaan on lisätty tekstin autocompleteminaisuus, joka myös pyörii tämän luokan sisällä. Syötteet annettuaan luokka muodostaa UserData-olion ja luokka tarkistaa, että onko tietokoneella jo olemassa vastaavalla opiskelijanumerolla varustettua json-tiedostoa, josta voisi suoritettuja kursseja lukea. Tämän tehtyään luokka antaa komennon ohjelmalle, joka siirtää ohjelman suorituksen toiseen ikkunaan. Tässä vaiheessa ohjelma lataa Sisureader-luokan avulla json-muotoisen tiedoston halutusta tutkinnosta. Tätä lähdetään sitten käsittelemään suoritusmerkintöjen avulla.

SecondaryController-luokka luo käynnistyessään kaksi ikkunaa. Vasemmalle tulee näkyviin Sisusta haetusta json-tiedostosta tehty kurssipuu. Oikealla taas on ikkuna, josta näkee suoritettut kurssit. Edellä mainitun kurssipuun muodostus tapahtuu varsinaisesti viimeisessä mainitsemattomassa luokassa DegreeStructureReaderissa. Luokkaan on asetettu tietyt säännöt, joilla komponentteja sijoitellaan rekursiivisten funktioiden avulla root-nimiseen Treeltemiin. SecondaryControllerissa annetaan luoduille kurssi-Treeltemille ominaisuus, jolla niitä voidaan merkata joko käydyksi tai poistaa käydyistä kursseista.

Opiskelijan tiedot ovat tallennettuja suorituksen ajaksi luokkaan UserData. Tämä luokka koostuu pääosin set- ja get-funktioista, eli toimii enimmäkseen tallennuskohteena ja tietojenkäsittelyn mahdollistajana. Ohjelman toisesta ikkunasta palatessa tai ohjelman vain suljettaessa kutsutaan App-luokassa olevaa saveUserData-funktiota. Tämän johdosta muodostuu json-muotoinen tiedosto, johon on tallennettuna opiskelijan tiedot ja tätä tiedostoa voidaan käyttää tietojen jatkokäsittelyyn.

## 4. SOVITTU JA TOTEUTUNUT TYÖNJAKO

Projektin alussa arvioidut tehtävät ja työt olivat jaettuna kolmeen osaan. Ensimmäisen osan eli Sisu-API:n toteutuksesta oli vastaamassa Joona ja suunnitelma pitikin loppupeleissä paikkansa, pois lukien Eeron pienen avun.

Seuraavaksi puhuttiin yleisesti kaiken taustalla tapahtuvan toteutuksesta (Back end) ja Eero tahtoi saada osakseen tämän työn. Todellisuudessa Eero sai hoidettua miltei kaiken tarvittavan, esim. kurssipuun luonnin kurssi-jsonista ja Tommi sekä Joona auttoivat tarvittaessa pienempien toimintojen luonnissa.

Viimeisenä Tommi halusi ottaa vastuulleen ohjelman grafiikan ja sen muodostamiseen vaadittavan koodin, mutta todellisuudessa Eero sai tehtyä vahvan, miltei valmiin pohjan mitä Tommi lähti sitten muokkaamaan ja kehittämään loppuun asti.

Testaamisesta sovittiin, että jokainen tekisi testin aina luomalleen ominaisuudelleen, mutta todellisuudessa testauksesta on vastannut vain Joona sekä Eero. Tommi on korvannut tätä panosta dokumentaation luonnilla ja kommentoinnilla.

## 5. ONGELMAT JA PUUTTEET

Projektin aikana on tullut törmättyä muutamaan kielelliseen/taidolliseen rajoitteeseen kurssipuuta vasempaan ikkunaan kirjoittaessa. Ikoneita ei kyetty mitenkään luomaan kurssien suorituksen merkkaukseksi. Sama asia kävi ilmi, kun ajatuksena oli luoda vain kurssista check boxeja ja muista tavallisia puun osia. Tästä syystä suoritusten merkkaukseksi siirryttiin siihen, että ohjelmassa käytetään toista ikkunaa näyttämään suorituksia.