Testülesanne arendajatele, Hambaarsti juurde registreerimise rakendus

Tööd alustasin esimesest etapist ja töötasin järjest lõpuni. Pärast projekti käivitamist Intellij-s alustasin punktist 1.2. Selle raames muutsin kuupäeva lahtrit viisil, et sisend tuleks datetime-local tüüpi, sel viisil saab valida kasutaja kalendrist kellaaja ning kuupäeva millal ta soovib visiiti broneerida. Lisasin DTO-sse ja kontrollerisse validatsioonid, mis kontrollivad, et väljad poleks tühjad ning aeg oleks E-R, 10-16:00 vahemikus. Aja kontrolliks kirjutasin kontrollerisse meetodi „areWorkinghours()“. Samuti lisasin ka DTO ja Entity klassidele uued väljad mis salvestaks perearsti nime ning visiidi kuupäeva ja kellaaja.

Teisena lisasin rakendusele registreeringute vaatamise funktsionaalsuse. Eelnevalt olen tegelenud rohkem javax raamistikuga, kuid mitte nii palju Spring raamistiku ja thymeleaf-iga. Seetõttu läks antud etapiga ka rohkem aega kui eelmisega, kuid tutvusin thymeleaf-i võimalustega ja lõin uue html faili nimega findreg.html, kuhu kuvasin th:each abil registreeringud. Registreeringute vaate kasutajale kuvamiseks kasutatakse “ Vaata registreeringuid“ hyperlinki pealehel.

Seejärel lisasin samale vaatele juurde otsingu funktsionaalsuse. Otsida on võimalik hambaarsti nime järgi. Backend-is toimib otsing läbi for tsükli. Võetakse lehekülje parameetri dentistName väärtus, käiakse läbi kõik andmebaasis olevad broneeringud ja kuvatakse kasutajale kõik broneeringud, mis sisaldavad otsitud nime. Otsing ei erista suurt ja väikest tähte. Detailsele registreeringu vaatele viitamine käib läbi hyperlink-i. Detailse vaate kuvamiseks lõin regdetail.html dokumendi. Registreeringu sisu kuvamine ja tuvastamine käib läbi DentistVisitEntity ID parameetri. Kuna eelnevate etappide juures olin thymeleafiga juba jõudnud rohkem tutvuda oli ilmselt antud punkt minu jaoks kõige lihtsam ja võttis kõige vähem aega.

Neljanda punkti tegin samuti ülesandes olevas järjekorras. 4.1 etapis lisasin võimaluse registreeringuid kustutada. Registreeringud tuvastatakse ID järgi ja kustutakse. Kusutamiseks lõin DentistVisitService-i ja DentistVisitDao klassidesse uued meetodid, vastavalt removeVisit() ja delete(). Kasutaja saab visiiti kustutada „Kustuta visiit“ nupu abil.

Registreeringute uuendamiseks lõin samuti DentistVisitService-i ja DentistVisitDao klassidesse uued meetodid – updateVisit() ja update(). Registreering valitakse ID järgi, sarnaselt kustutamisel. Kasutaja saab visiiti kustutada „Uuenda visiiti“ nupu abil.

Broneeringu aegade kontrolliks lõin DentistVisitService klassi meetodi doesExist(). Meetod võtab sisendiks hambaarsti nime ja registeeringu aja ning otsib for tsükliga kõikide broneeringute seast kattuvusi. Juhul kui leitakse teine broneering sama hambaarsti nime ja ajaga, tagastab meetod väärtuseks „true“. Meetodit kasutatakse kontrolleris, nii broneeringu muutmisel kui ka esmasel registreerimisel ehk form.html ja regdetail.html dokumentides. Juhul kui broneering on juba olemas kuvatakse kasutajale vastav veateade.

Lõpuks lisasin ka kõigile html failidele hyperlinkidena viited, et kasutja saaks navigeerida rakenduse erinevate lehtede vahel.

Üldiselt oli ülesanne väga huvitav ja sain kasutada nii eelnevaid teadmisi kui ka tutvuda uute raamistike võimalustega. Kokku kulus ülesande lahendamiseks ligikaugu 17-18h. Minu jaoks olid keerulisemad osad seotud rakenduse front-end osaga ja kiiremini said lahendatud osad, kus tuli kirjutada back-end loogikat. Kohad kus jäin rohkem hätta lahendasin lugedes springi ning thymeleaf-i võimalustest nende kodulehtedel ja googledades erinevaid näiteid ning lahendusi.