OTP2 Raportti

Vladimir Gavrilov, Heikki Ketoharju, Kari Lampi, Suvi Rannisto

Sisällys

1 Johdanto

1.1 Projektin tavoite

Tässä dokumentissa esitellään Ohjelmointiprojekti 2 -kurssin projektityö, sen tavoite, ominaisuudet ja toiminnallisuudet sekä arkkitehtuuri. Projektityön tavoitteena oli tutustua scrum-projektinhallintaan ja ketterään ohjelmistokehitykseen. Projektihallintajärjestelmänä käytettiin selainkäyttöistä Agilefant-pilviversiota. Lisäksi tuli toteuttaa monikielisyys, jotta projektia voi kansainvälistää ja lokalisoida.

1.2 Projektityön aihe

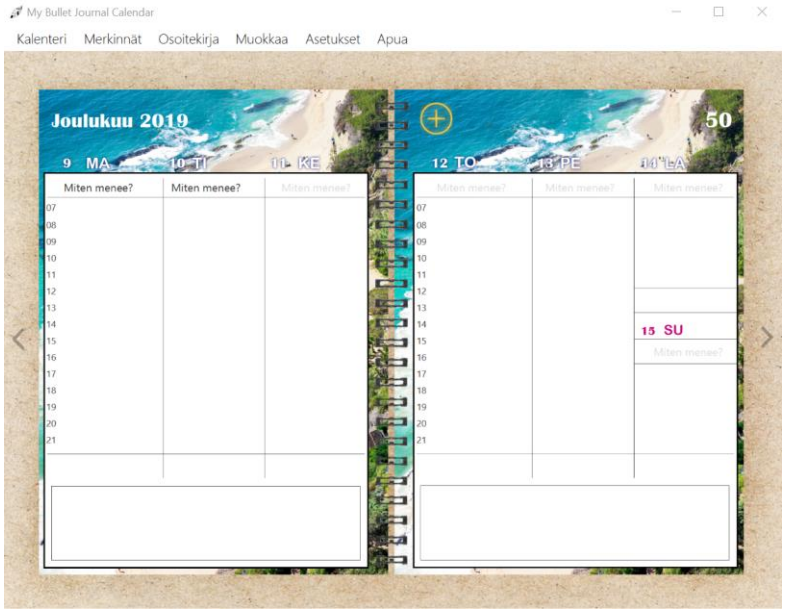
Projektityön aiheena on Bullet Journal -tyylinen kalenteri. Bullet Journal on henkilökohtaiseen organisointiin kehitetty metodi, jonka kehitti suunnittelija Ryder Carroll. Metodi organisoi aikataulutuksen, muistutukset, to-do -listat, mietteet sekä muut organisoitavat tehtävät yhteen muistivihkoon. Metodin oleellisena osana on myös luovuus ja erilaiset kynät, tarrat ja washi-teipit, joilla merkintöjen muistiinpaneminen on mielekästä ja inspiroivaa. Visuaalisuudella pyritään myös kasvattamaan motivaatiota arjen tehtävien loppuunsaattamiselle.

Kuva 1: Bullet Journal muistivihko, jossa erilaisia metodille tyypillisiä merkintätapoja.

2 Projektityön visio

2.1 Kalenterinäkymä

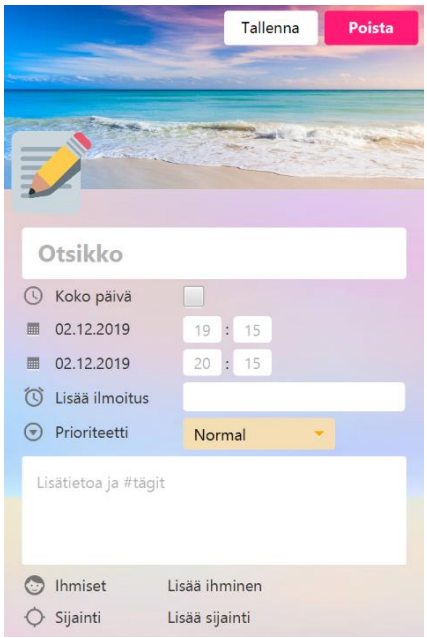
Omassa kalenterissamme on viikkonäkymä, johon voi lisätä kahdentyyppisiä merkintöjä: tavallisia merkintöjä sekä to-do-listoja. Merkinnät ja to-do -merkinnät voivat sisältää ihmisiä, hashtageja ja paikkoja. Kalenterin viikkonäkymää pystyy muokkaamaan itselleen mieluisaksi fontein, värein ja kuvin. Päiviä voi selailla joko vaihtamalla viikkonäkymää aina yhden viikon eteenpäin tai taaksepäin, tai valitsemalla kalenteri-popupista päivän, jolloin aukeaa päivän sisältävän viikon viikkonäkymä.



Kuva 2: Ohjelmointiprojektin kalenterinäkymä.

2.2 TAVALLINEN MERKINTÄ

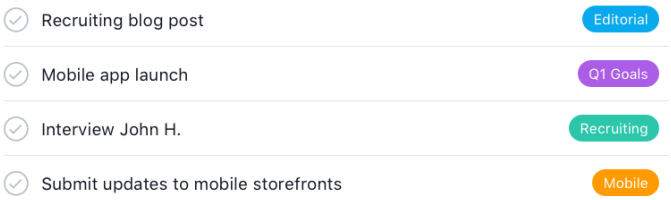
Tavallinen merkintä sisältää otsikon, kellonajan, päivämäärän ja muistiinpanon sekä mahdolliset prioriteetin, hälytyksen, ihmiset, hashtagit ja paikannimet. Merkintöjä voi selailla erikseen myös merkinnät-listasta, josta voi helpommin selailla haluamiaan merkintöjä ilman kalenterinäkymää.



Kuva 3: Hahmotelma merkintöjen, to-do -tehtävien, ihmisten ja paikkojen lisäämis- ja muokkausikkunoista.

2.3 TO-DO -MERKINTÄ

To-do -merkintä sisältää kaikki samat ominaisuudet kuin tavallinenkin merkintä, mutta kellonaika puuttuu. Tämän lisäksi to-do -merkinnässä on tieto, onko merkintä tehty vai tekemättä. To-do -listat sijoitetaan aina päiväsarakkeen alaosaan, ei mihinkään tietylle kellonajalle. To-do -merkintöjä voi lisäksi tarkastella erillisen listanäkymän kautta.



Kuva 4: Mahdollinen esillepano käyttöliittymän päivän kohdalla. Värikkäät boksit sisältäisivät prioriteetit.

2.4 Kellonaika ja päivämäärä

Merkintää lisättäessä kellonaika ja päivämäärä määräytyvät sen mukaan, missä kohtaa hiiren kursori on klikattaessa viikkonäkymää. Kellonaikaa voi myös muokata merkinnän luonti- tai muokkausvaiheessa joko kirjoittamalla tekstikenttään kellonajan, jonka tekstintunnistus tunnistaa, tai kellonajalle tarkoitettuun tekstikenttään. Päivämäärän asettaminen toimii samoin kuin kellonajankin asettaminen. Jotta käyttäjä tietää, että kellonaika tai päivämäärä on kirjoitettu oikein ja tunnistettu merkintä-tekstikentästä, ne värjätään. Jos värjäystä ei tapahdu, tekstin muoto on väärä ja näitä tietoja ei aseteta.

2.5 Hälytykset

Merkinnöille voi halutessaan asettaa hälytyksen. Hälytyksen ajankohdan voi asettaa mieleisekseen suhteessa merkintään: 5, 10, 20, 30, 45 tai 60 minuuttia ennen merkintää. Myös vapaavalintaisen hälytysajan asettaminen on mahdollista.

2.6 Prioriteetit

Merkinnöille voi asettaa prioriteetteja, ja näin lajitella merkinnät tärkeyden mukaan listanäkymässä. Prioriteetti näytetään käyttöliittymässä värikoodin avulla.

2.7 Hashtagit

Tavallisiin merkintöihin ja to-do -merkintöihin voi lisätä hashtageja, joita voi luoda lennosta merkintöjä luodessa tai muokatessa. Hashtagien tarkoituksena on lajitella merkintöjä erilaisiin kategorioihin, jolloin tietynlaisten merkintöjen etsiminen on helpompaa. Hashtagin voi lisätä merkintään kirjoittamalla suoraan merkinnän tekstikenttään #-merkin ja halutun merkkijonon.

2.8 Ihmiset ja paikat

Merkintöihin voi lisätä ihmisiä ja paikkoja, jos esimerkiksi on tapaaminen tietyn henkilön kanssa jossain tietyssä paikassa, ja sen haluaa merkitä muistiin. Ihmisten ja paikkojen luonti voi tapahtua merkinnän luonnin yhteydessä tai erikseen omassa muokkaus välilehdessä. Ihmisen luontiin tarvitaan etu- ja sukunimi, jotka ovat pakollisia kenttiä. Siihen voi lisäksi lisätä nimimerkin, puhelinnumeron, osoitteen, sähköpostin ja työpaikan. Jos nimimerkki on tiedossa, merkintää luotaessa tai muokattaessa ihmisen lisääminen @-merkin jälkeen ehdotetaan nimimerkki-nimeä eikä etu- ja sukunimeä. @-merkin jälkeen kirjoitettaessa käytetään syötön täydennystä, jotta oikea ihminen löytyy, jos se on tietoihin tallennettu.

Samat ominaisuudet pätevät paikan lisäämisessä ja luomisessa. Paikan ominaisuuksia ovat nimi ja osoite, joista nimi on pakollinen. Paikan lisääminen tapahtuu kirjoittamalla merkkijono 'at' ja sen jälkeen osoite. Siinäkin pätee syötteen automaattinen täydennys.

3 Toiminnallisuudet

3.1 TÄHÄN ASTI TOTEUTETUT TOIMINNALLISUUDET

3.1.1 Päivämäärien määrittely ja DatePicker

Päivämäärien määrittely on toteutettu Javan Calendar-luokalla. Kun kalenterisovellus käynnistetään, se automaattisesti hakee tämän hetkisen päivämäärän ja avaa kalenterinäkymän siltä viikolta. Päivät haetaan aina Calendar-luokan avulla niin, että etsitään oikea viikkonumero, josta Calendar-luokka poimii sen viikon päivämäärät. Kuukausien, vuosien ja viikkojen määrittely tulee Calendar-luokasta ja ne poimitaan omissa metodeissaan.

Näkymässä pystyy selaamaan viikon kerrallaan eteen- ja taaksepäin kahdesta nuolikuvakkeesta. Tämä toiminnallisuus on toteutettu kahdella metodilla, jotka laskevat kahden eri kuvakkeen painalluksia. Nämä painallukset lasketaan yhteen, jolloin saadaan tämänhetkisestä viikosta katsottuna aina oikea seuraava tai edellinen viikko.

Päivien selaaminen tapahtuu DatePickerin avulla. Kun DatePickeristä on valittu päivämäärä, otetaan sen value eli päivämäärä ja haetaan jälleen viikkonumero, jonka kautta oikeat päivämäärät kalenterin viikkonäkymään saadaan. DatePicker ja viikko eteen- ja viikko taaksepäin-napit eivät vielä ole kytketty toisiinsa, joten jos DatePickerin jälkeen painaa viikko eteen- tai viikko taaksepäin-nappia, päivämäärät määräytyvät jälleen tämänhetkisen päivämäärän mukaan eikä DatePickerissä valitun päivän mukaan.

3.1.2 Fonttien hakeminen ja muokkaaminen

Päivien nimien ja päivämäärien, kuukauden, vuoden ja viikon fonttien vaihtaminen onnistuvat kaikki erikseen, paitsi kuukauden, vuoden ja viikon, jotka ovat tarkoituksella aina samaa fonttia keskenään. Fonttien vaihto tapahtuu niin, että käytettävät fontit haetaan käyttäjän tietokoneelta, josta ne napataan menu-itemeina menulistaan. Tästä listasta käyttäjä valitsee mieluisensa fontin ja painaa ok, jolloin fonttien vaihto tapahtuu. Cancel-nappia ei ole vielä toteutettu.

3.1.3 Tavallisten merkintöjen luonti, muokkaaminen ja poistaminen, merkintöjen ominaisuudet

Tavallisten merkintöjen luonti, muokkaaminen ja poistaminen toimivat niin frontista tietokantaan asti. Merkintää luotaessa merkintään lisätään pakollisena otsikko sekä aloitus- ja lopetuspäivä. Muita tietoja ovat aloitus- ja lopetuskellonaika, prioriteetti merkintöjen listausta varten, ilmoitusten lisääminen ja lisätietoa kentän täyttäminen.

Merkinnät näkyvät frontissa, mutta ne eivät ihan osu oikeille kellonajoille. Paddingeissä tai fontin koossa voi olla korjauksen aihetta, jotta ne osuisivat oikeille paikoilleen. Myöskään päällekkäisiä merkintöjä emme ole ehtineet huomioimaan, vaan merkinnät näkyvät nyt päällekkäin eikä esimerkiksi vierekkäin.

Merkintöjen ominaisuuksista ihmisten ja sijainnin lisääminen ei toteutettu.

3.1.4 Monikielisyys käännökset

Lokalisointiin tarvittiin kielikäännökset, jotka tehtiin neljälle eri kielelle suomeksi, ruotsiksi, englanniksi ja tanskaksi. Kielikäännökset laitettiin resurssitiedostoihin, joista settings-luokka etsii oikean kielikäännöksen. Kaikki muut toiminnallisuudet saatiin tehtyä, mutta piirtoa varsinaiseen näkymään emme ehtineet tehdä.

3.2 TOTEUTTAMATTOMAT TOIMINNALLISUUDET

3.2.1 To-do -merkintöjen luonti

Päätimme, ettemme ala erittelemään to-do -merkintöjä, sillä tavallisissa merkinnöissä oli jo haastetta kerrakseen.

3.2.2 Tekstintunnistus

Merkintöihin on tarkoitus toteuttaa tekstintunnistus. Tämän avulla merkinnän voi tehdä yhdellä tekstirivillä, ja kalenteri tunnistaa automaattisesti tekstistä kellonajat, henkilöt, paikat, hashtagit, todo-merkinnät ja muun tarpeellisen.

Tätä aloitettiin tekemään, mutta ei saatu täysin toimintavarmaksi, joten sitä ei projektiin tullut.

Muutamia esimerkkejä:

- Lounas @Iiris kanssa klo 12:30-14:00 at Lasipalatsin kahvio
- @Matti syntymäpäivät la klo 14:00 at Hakaniemi
- Tentti #Metropolia #opiskelu at Myllypuron kampus ke klo 13:00 /muistuta 1h ennen

Tavoitteena siis on, että yksinkertaisia, suhteellisen yleisesti käytettyjä tai muista sovelluksista tuttuja merkintätapoja (mentionit, hashtagit, “klo”) käyttämällä voi luoda merkintöjä kalenteriin halutuille kellonajoille ja päiville.

Tätä varten toteutetaan tekstintunnistussysteemi, joka käy merkinnän läpi. Suunnitelmana on yhdistää yksinkertaisia säännönmukaisia lausekkeita, sekä joitain recursive descend parserin ominaisuuksia, kuten varattujen sanojen tunnistaminen, ja niiden perässä olevan merkkijonon käsitteleminen rekursiivisella algoritmilla.

Tavoitteena on, että merkinnän tekstiä analysoimalla muodostetaan rakenteellinen tulkinta, joka sisältää tietoa merkinnän ajasta, paikasta ynnä muusta. Tämä rakenteellinen tulkinta tallennetaan merkintäolioon alkuperäisen merkintätekstin rinnalle, ja tulkinnasta johdetaan merkinnän attribuutit, kuten kellonaika, paikka, tai merkintään liittyvät henkilöt.

Tämän tulkinnan pohjalta merkinnän tekstiin myös lisätään tunnisteita, joiden avulla käyttäjä voi havaita tekstintunnistuksen tulokset. Esimerkiksi värjätään kellonaika tai muutetaan osoite klikattavaksi linkiksi.

3.2.3 Ihmisen ja paikan luonti

Uuden ihmisen tai paikan lisääminen ei vielä ole mahdollista frontista käsin. Ihmisten ja paikkojen lisäämisominaisuudella on tarkoitus yhdistää niin sanottu osoitekirja, joka kalentereissa yleensä löytyy. Näillä ominaisuuksilla voi myös hakea tiettyjä merkintöjä samalla tyylillä kuin hashtageillä.

3.2.4 Merkintöjen, hashtagien ja paikkojen listaus

Listauksella pyritään tekemään kalenterin merkintöjen selaamisesta ja etsimisestä helpompaa. Merkinnät mahtuvat pienempään tilaan ja ne ovat selkeämmin esillä, kun ylimääräinen grafiikka on taustalta pois. Tähän on myös pyrkimys yhdistää Hae-ominaisuus.

3.2.5 Moduulit

Moduuleilla tarkoitamme erilaisia ylimääräisiä ominaisuuksia kalenterissa, kuten mielialatracker tai päivän sääennuste. Tällaiset trackerit kuuluvat yhtenä osana Bullet Journal -metodiin.

Moduuleita on tarkoitus voida poistaa ja lisätä joustavasti. Lisäksi niiden paikkaa, kokoa ja ulkonäköä voi muuttaa.

Tämän päätimme jo alussa, että emme toteuta näitä.

3.2.6 Tooltipit

Tooltipeillä on tarkoitus helpottaa aloittelevan käyttäjän tutustumista sovellukseen. Koska kyseessä on Bullet Journal -tyylinen kalenteri, sovelluksesta on pyritty tekemään mahdollisimman paperisen kalenterin tai muistivihon näköinen. Tämä tarkoittaa, että painikkeiden ulkonäkö ei muistuta esimerkiksi buttoneita. Täten on muilla keinoilla tarkoitus ohjeistaa aloittelevaa käyttäjää, mitä kaikkea kalenterissa voi muokata ja missä mikäkin toiminnallisuus on.

3.2.7 Värien vaihtaminen ja koon muuttaminen fonteissa

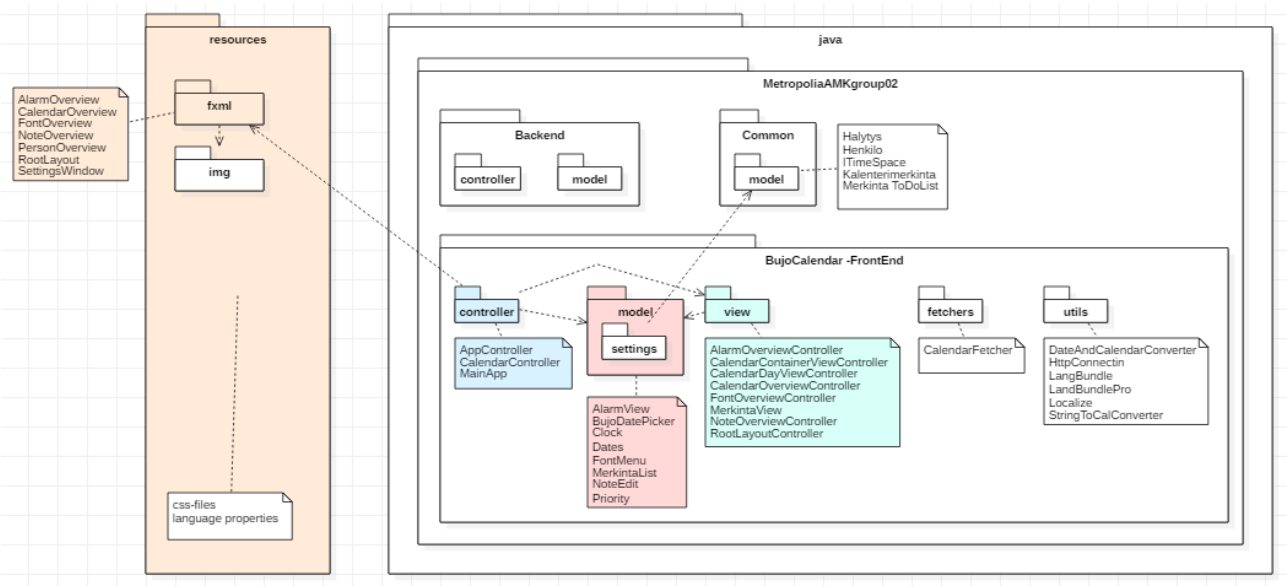
Värien vaihtaminen on tarpeellista ainakin headereissa, koska headerin kuvan vaihtaminen voi vaikeuttaa kuukauden, vuoden ja viikkonumeron näkemistä. Myös koon muuttaminen on tärkeää, sillä fontit ovat erikokoisia ja uudella fontilla tekstit eivät välttämättä mahdu kokonaan näkyviin labeleissaan.

3.2.8 Kuvien lisääminen ja vaihtaminen

Kuvien lisääminen ja vaihtaminen kuuluu oleellisena osana Bullet Journaliin. Haluamme, että käyttäjä voi lisätä kalenteriin tarroja, ja muokata myös kalenterinäkymän yläreunassa näkyviä otsakekuvia sekä kalenterin taustakuvaa.

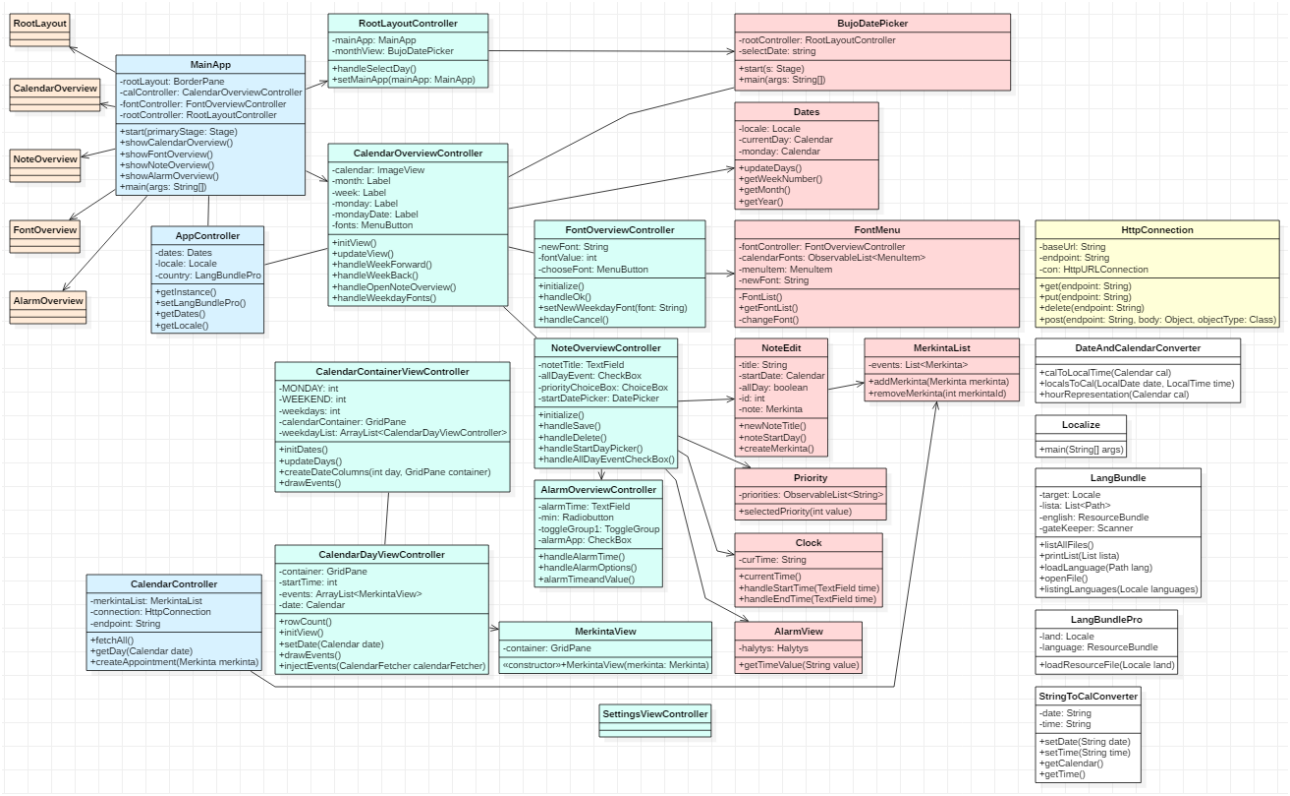
4 Sovelluksen arkkitehtuuri

4.1 PAKETTIKAAVIO KOKO SOVELLUKSESTA



Sovelluksen sisäinen tiedonsiirto perustuu asiakassovelluksen ja palvelimen väliseen yhteistyöhön rajapinnan välityksellä. Yhteydenpidossa asiakas ja palvelin hyödyntävät HTTP-protokollaa, jonka kautta tiedonsiirto suoritetaan. Oliot käännetään Jackson-kirjastolla Java-olioista JSON-merkkijonoiksi, jotka lähetetään HTTP-protokollan välityksellä vastaanottavalle sovellukselle. Vastaanottavassa päässä JSON-merkkijonot käännetään Jackson-kirjastolla takaisin Java-olioiksi. Tämä prosessi on mahdollista tehdä sekä palvelimesta asiakassovellukseen että asiakassovelluksesta takaisin palvelimelle. Palvelimella olioiden Hibernate-ominaisuuksia käytetään olioiden tallentamiseksi suoraan tietokantaan.

4.2 LUOKKAKAAVIO KÄYTTÖLIITTYMÄSTÄ



Kuva 5: Luokkakaavio frontendistä.

Käyttöliittymä käynnistyy MainApp-luokasta. MainApp käynnistää RootLayout-näkymän, jonka sisään tulee CalendarOverview-näkymä. Tämä näyttää kalenterin yleisnäkymän. Kalenterinäkymän sisällä toteutetaan viikkonäkymän piirto CalendarContainerView-näkymäluokan kautta. Jokaista päivää käsittelee oma päivänäkymä-luokkansa, joka sisältää listan tapahtumanäkymäolioita. Näin yksittäinen tapahtumaolio voidaan käskeä piirtämään itsensä.

Dates-malli huolehtii päivämäärien määrittämisestä ja päivittämisestä CalendarOverview-näkymään. Fonttien muokkauksesta huolehtii FontOverview-luokka, joka kutsuu Menu-luokkaa hakemaan käyttäjän koneelta kaikki käytettävät fontit menulistaan. Kun fontti on valittu, fontin nimi kulkee FontOverview-luokasta MainAppin kautta CalendarOverview-luokkaan, josta fonttien vaihto tapahtuu.

RootLayout-luokassa olevasta menubarista voi valita Selaa päiviä -toiminnallisuuden, joka avaa BujoDatePicker-luokan. BujoDatePicker-luokka sisältää DatePickerin, joka sisältää kuukausinäkymän ja siitä voi valita tietyn päivän nopeasti ja sujuvasti. Kun päivä on valittu, kalenteri avaa viikkonäkymän sen päivän kohdalta. Tieto valitusta päivästä kulkee RootLayout-luokasta MainApp-luokan kautta CalendarOverview-luokkaan.

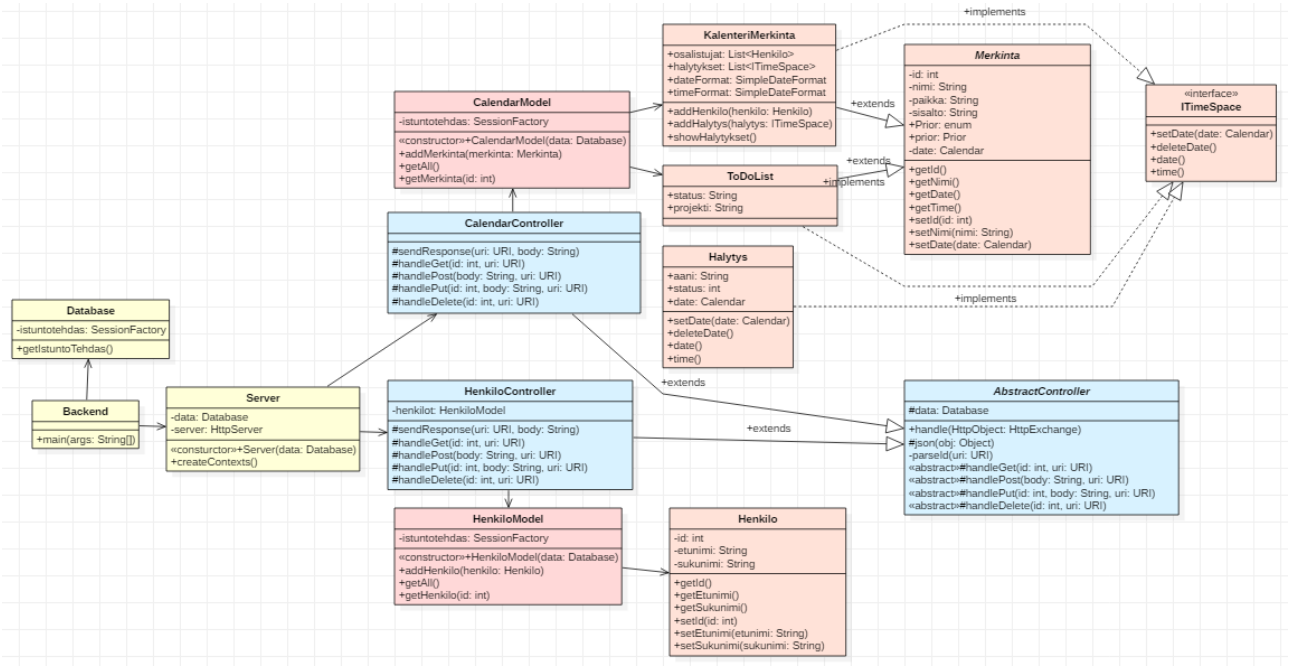
4.2.1 HTTP-clientin toteutus

Frontend-sovellukseen on rakennettu minimaalinen, yksinkertainen HTTP-client, joka osaa tehdä kutsuja Backendille, ja muuntaa vastaukset JSON-merkkijonoista Java-olioksi. Client on käytännössä yksittäinen, HTTPConnection-niminen olio, joka luodaan aina tarpeen vaatiessa. Jokainen olion ilmentymä edustaa yksittäistä HTTP-kutsua tiettyyn endpointtiin.

HTTPConnection osaa tehdä automaattisesti edestakaisen muunnoksen JSON-merkkijonosta Java-olioksi ja takaisin. Tähän käytetään Jackson-muunnoskirjastoa. JSON-muunnos on frontendille ja backendille yhteinen toiminnallisuus. Siksi se on sijoitettu common-pakkaukseen, josta sen perivät sekä backendin AbstractController että frontendin HTTPConnection-luokka.

Näin samaa koodia ei tarvitse kirjoittaa uudelleen, ja mahdolliset JSON-muunnokseen liittyvät bugit tulee korjattua läpi koko sovelluksen.

4.3 LUOKKAKAAVIO BACKEND-PUOLESTA



Kuva 6: Luokkakaavio backendistä.

4.3.1 Sovelluksen rakenne

Sovellus käynnistyy Backend-luokkaa kutsumalla. Backend alustaa tietokantaa säilyttävän olion, sekä käynnistää HTTP-serverin kusumalla Server-oliota. Server-oliossa konfiguroidaan endpointtien kytkennät niitä vastaaviin kontrollereihin.

4.3.2 Serverin toteutus

Backendiin on toteutettu yksinkertainen HTTP-server, joka osaa reagoida tyypillisimpiin verbeihin: GET, PUT, DELETE ja POST, ja tunnistaa erilaisia endpointteja, kuten /calendar tai /henkilo. Jokaiselle endpointille voi määrittää oman kontrollerin, joka käsittelee kyseisen endpointin. Kontrollerissa eri HTTP-verbit käsitellään omissa metodeissaan.

Serverin yleinen toiminnallisuus on sijoitettu abstraktiin kontrolleriin, joka hoitaa HTTP-yhteyden yksityiskohdat. Abstrakti kontrolleri vaatii, että sen perivät kontrolleriluokat toteuttavat määrätyt metodit handleGet, handlePut, handlePost ja handleDelete.

4.3.3 Tietokantaan tallennus ja tietojen hakeminen sieltä

Backendissä voi tallentaa ja hakea tietoa tietokannasta Hibernatella. Tietokantaan tallennetaan esimerkiksi kaikki merkinnät, merkintöjen ominaisuudet ja muut kalenteriin tehtävät muutokset.

Tietokantaan tietoa tallentavat Hibernate-oliot ovat samalla myös Jackson-olioita. Niihin sisältyy siis sekä tieto tietokantaan tallennettavasta datasta, että JSON-muunnokselle lähetettävistä kentistä.

Kontrollerit luovat ja hakevat Hibernate-oliot, ja siirtävät ne eteenpäin JSON-muunnokselle. Näin palvelimen endpointtia kutsumalla saa vastaukseksi JSON-muotoisen esityksen Java-oliosta.

Lähteet

Kuva 1: https://media.newyorker.com/photos/5d701d7c9e4fc500085d1f29/master/w_727,c_limit/Russell-BulletJournal.jpg

Kuva 4: https://luna1.co/2a0e29.png