SAVONIA
ammattikorkeakoulu

OPINNÄYTETYÖ - AMMATTIKORKEAKOULUTUTKINTO
TEKNIIKAN JA LIIKENTEEN ALA

ANDROID-SOVELLUS ADHD-DIAGNOSOIDUILLE KÄYTTÄJILLE

TEKIJÄ Tommi Riihiluoma

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Tutkinto-ohjelma Tietotekniikan tutkinto-ohjelma	
Työn tekijä Tommi Riihiluoma	
Työn nimi Android-sovellus ADHD-diagnosoiduille käyttäjille	
Päiväys 5.5.2022	Sivumäärä 33
Toimeksiantaja Juha Ruuskanen	

Tiivistelmä

Opinnäytetyön aiheena oli Android-sovellus, jolla käyttäjät, joilla on todettu keskittymishäiriö, voisivat helpommin järjestää päivittäistä elämäänsä sovelluksen tarjoamien ajanhallinta- ja harjoiteaktiviteettien avulla. Projektin keksijä ja tilaaja oli Juha Ruuskanen.

Sovellus toteutettiin Javalla ja XML:llä Android Studio -ohjelmistoympäristöä käyttämällä, ja se suunniteltiin käytettäväksi mobiililaitteilla. Muita projektissa hyödynnettyjä teknologioita olivat HTML, CSS ja JavaScript.

Tämä raportti käsittelee sovelluksen suunnittelu ja kehitysvaihetta. Projektin aikana kokeiltiin erilaisia ominaisuuksia ja tekniikoita, mutta osa niistä jätettiin pois viimeisistä versioista. Kehitysprosessi osoittautui kuitenkin kattavaksi opiskelukokemukseksi, jossa ohjelmistoa toimitettiin asiakkaalle tilaustyönä.

Avainsanat

Android, Java, ADHD, mobiililaite

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author Tommi Riihiluoma			
Title of Thesis Android Application for Users with Diagnosis of ADHD			
Date	5 May 2022	Pages	33
Client Mr Juha Ruuskanen			

Abstract

The aim of this thesis was to develop an Android application with which the people diagnosed with attention deficit hyperactivity disorder (ADHD) could more easily organize their daily lives through time management and exercise activities provided in the application. The project was commissioned by Juha Ruuskanen.

The application was implemented with Java and XML using Android Studio IDE, and it was designed to be used on mobile devices. Other technologies utilized in the project were HTML, CSS and JavaScript.

This report covers the planning and the development process of the application. A set of techniques and features were implemented and experimented with along the project, but some of them were ultimately not included in the final versions. However, the development process proved to be a comprehesive learning experience from a standpoint of delivering software to a client.

Keywords

Android, Java, ADHD, mobile device

SISÄLTÖ

KÄSITTEET

ADHD	Keskittymishäiriö (englanniksi Attention Deficit Hyperactivity Disorder) on toimintakykyä heikentävä kehityksellinen häiriö, jonka keskeisiin oireisiin kuuluu tarkkaamattomuus (Käypä hoito 2019).
Rajapinta	Rajapinta tai ohjelmistorajapinta on ohjelmisto- tai laitteistokokonaisuuden osa, jonka kautta tietoa viestitetään eri komponenttien välillä.
Ohjelmointiympäristö	Kehitys- tai ohjelmointiympäristö (englanniksi Integrated Development Environment) on tietokoneohjelma, jota käytetään ohjelmistojen luomiseen.
Kirjasto	Kokoelma ohjelmakoodia, jota voidaan hyödyntää sovelluskehityksessä liittämällä se osaksi kehitettävää ohjelmistoa.
Olio-ohjelmointi	Ohjelmointitekniikka, jossa luotava ohjelma jäsennellään eri osiin luokkarakenteita käyttämällä (Mooc julkaisuaika tuntematon).
Muuttuja	Ohjelmoinnissa muuttujat esittävät tietokoneen muistiin tallennettavaa arvoa, jota käytetään toimintojen suorittamiseen.
Luokka	Olio-ohjelmoinnissa luokkia käytetään ohjelmiston käsittelemän tiedon kuvaamiseen ja jakamiseen eri kokonaisuuksiksi, ja niihin liittyvän toiminnallisuuden määrittelemiseen (Mooc julkaisuaika tuntematon).
Ilmentymä	Ilmentymä tai olio on muuttuja, joka on määritelty luokan mukaan.
Metodi	Metodi tai funktio on nimetty osa ohjelman koodia, jota voidaan käyttää kutsumalla sitä ohjelman koodissa (Mooc julkaisuaika tuntematon).
Säie	Ohjelmoinnissa käytettävä tekniikka, joka toimii eräänlaisena jonona, jossa toimintoja suoritetaan. Säikeitä voi käyttää ohjelmissa yhdenaikaisesti, jolloin tehtävien suoritusnopeus tehostuu. Joissakin ohjelmointitekniikoissa säikeiden käyttäminen on välttämätöntä.
Näkymä	Käyttöliittymän elementti Android-sovelluksissa, jonka toiminta on määritetty View-luokassa, ja siitä johdetuissa aliluokissa. Näkymä voi olla esimerkiksi tekstikenttä, kuva tai painike (Android for Developers 2022).
Kuuntelija	Kuuntelija (englanniksi listener) on ohjelmoinnissa käytettävä tekniikka, jossa vahditaan johonkin asiaan kohdistuvaa tiettyä tapahtumaa, esimerkiksi käyttöliittymän painikkeeseen kohdistuvaa painallusta.

1 JOHDANTO

Älypuhelinten ja Internet-yhteyden saatavuuden yleistymisen myötä on tultu vaiheeseen, jossa niiden käyttöä pyritään lisäämään jokaisella alalla. Lähes kaikki omistavat ainakin yhden älylaitteen, mikä mahdollistaa nopean pääsyn tietolähteisiin sekä erilaisiin toimintoihin, joita laitteilla on mahdollista suorittaa. Tämä tuottaa kysymyksen: Miten ohjelmistotekniikan tarjoamia työkaluja on mahdollista hyödyntää tarjoamaan kuluttajille helposti käytettäviä ja saatavilla olevia palveluita?

Tämän opinnäytetyön tavoitteena on luoda Android-mobiililaitteilla käytettävä sovellus, jota ADHD-oireiset voivat käyttää päivittäisessä ajanhallinnassaan. Sovellukseen toivotut toiminnot ja teemat, ja mahdollisuudet niiden toteuttamiseen käytännön tasolla, käydään läpi työn toimeksiantajan kanssa projektin aloitusvaiheessa. Lisäksi niihin liittyvistä muutoksista keskustellaan myös työn edetessä. Tarkoituksena on suunnitella käytettävyydeltään ja saatavuudeltaan mahdollisimman helppokäyttöinen sovellus mobiililaitteille suurikokoisen kohderyhmän käyttöön.

Projektin aikana sovelluksen arkkitehtuuriin, kuten tekniikoihin ja ulkoasuun, liittyvää viestintää käydään läpi suullisesti tapaamisten yhteydessä, sekä sähköisen viestinnän kautta. Itse ohjelmiston luomiseen tarvitaan Internet-yhteys viestintää ja erilaisten resurssien hankintaa varten, ja myös tietokone, jolla käytetään Android Studio -ohjelmistoympäristöä sovelluksen kehittämistä varten.

2 ANDROID

Android on ilmainen, avoimen lähdekoodin käyttöjärjestelmä, joka on suunniteltu pääasiassa kosketusnäytöllisille mobiililaitteille. Sen lähdekoodi on julkaistu Android Open Source Project (AOSP) -nimellä, ja Apache-lisenssin alaisena, mikä mahdollistaa ohjelmiston käytön, muokkaamisen ja jakamisen luvan ehtojen mukaan ilman korvausvelvollisuutta. (Elprocus 2021.) Käyttöjärjestelmän kehitti ohjelmistoyritys Android, Inc., jonka Google osti vuonna 2005 (Investopedia 2021). Nykyään Androidia kehittää Open Handset Alliance -yhtymä, jota Google rahoittaa, ja se on ollut myydyin käyttöjärjestelmä älypuhelimilla ja taulutietokoneilla vuodesta 2017 lähtien (Tynkkynen 2018). Puhelimien ja taulutietokoneiden lisäksi Android-järjestelmiä käytetään myös älykelloissa, televisioissa ja jopa autoissa.

KUVA 1. Googlen vuodesta 2019 lähtien käyttämä kaupallinen Android-logo (Googlen blogi 2019)

Android muodostuu pinoista ohjelmistokomponentteja, jotka voidaan jakaa viiteen osaan ja neljään päätasoon. Ohjelmistopinon alimmalla tasolla on Linux kernel, joka on käyttöjärjestelmän ydin. Se prosessoi ohjelmistolta saapuvia pyyntöjä ja ohjaa järjestelmän olennaisia toimintoja, kuten muistia, laitteistoa ja yhteyksiä. (Elprocus 2020.) Ytimestä seuraavalla tasolla on Hardware Abstraction Layer (HAL), joka toimii rajapintana laitteiston ja ohjelmistokehyksen välillä. HAL-abstraktiotaso mahdollistaa laitteiston hyödyntämisen järjestelmässä niiden vaihtelevista ominaisuuksista riippumatta. (Android for Developers 2021; Wikiopisto 2019.)

Seuraavalla tasolla sijaitsevat niin kutsutut natiivikirjastot, joita voidaan käyttää Androidin natiivikehityspaketin (NDK) avulla sovelluskehityksessä. Nämä kirjastot mahdollistavat C- ja C++-ohjelmointikielten käytön sovelluksissa, joita voidaan hyödyntää ohjelmoitaessa laskennallisesti vaativia sovelluksia tai ohjatessa laitteiden fyysisiä komponentteja. (Android for Developers 2020.) Samalla tasolla natiivikirjastojen kanssa ovat myös Android Runtime –ohjelmistokerros, joka sisältää sovellusten ajoympäristönä käytettävän virtuaalikoneen, sekä kirjaston, jossa on suurin osa Java-ohjelmointikielen toiminnallisuudesta (Wikiopisto 2019).

Tätä ylemmällä tasolla on sovelluskehysrajapinta, joka on kirjoitettu myös Javalla. Sovelluskehys mahdollistaa Android-käyttöjärjestelmän ominaisuuksien hyödyntämisen sovelluksia kehitettäessä. Kehys sisältää erilaisia Java-luokkia, joita käytetään esimerkiksi käyttöliittymän suunnittelussa ja resurssien ja ilmoitusten hallinnassa. Ylin käyttöjärjestelmän taso muodostuu Android-laitteelle asennetuista sovelluksista, joita käyttämällä käyttäjä kommunikoi järjestelmän kanssa. Laitteille on aina asennettuna tyypillisiä perussovelluksia, kuten puhelin-, tekstiviesti-, selain- ja kalenterisovellukset, ja niitä voi hankkia lisää sovelluskaupoista tai muista lähteistä. (Android for Developers 2020; Wikiopisto 2019.)

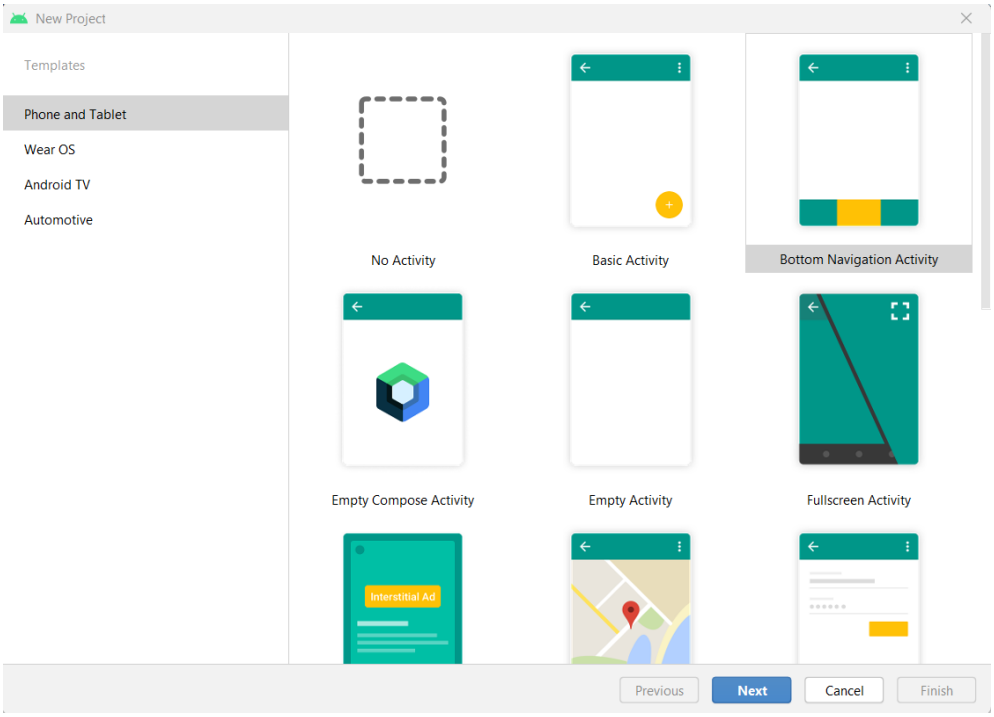
Androidista on tehty sen ensijulkaisun jälkeen aina uudempia versioita, joissa uusia ominaisuuksia ja parannuksia on lisätty. Ensimmäinen kaupallinen versio julkaistiin vuonna 2008 (Javatpoint 2021). Android-versioiden tarjoamaa toiminnallisuutta kuvataan API-tasolla, kokonaislukunumerolla, joka kasvaa uudempien versioiden mukana. Android 1.0 -käyttöjärjestelmäversion API-taso on 1, kun taas Android 11 -versio on tasoa 30, ja näin ollen Androidiin on tullut vuosien aikana valtavasti muutoksia. Androidin sovelluskehys on suunniteltu niin, että sen uudemmat versiot ovat pääasiallisesti yhteensopivia aikaisempien versioiden kanssa, mutta sen API-tasot on otettava huomioon sovelluskehityksessä, kun päätetään, mitkä käyttöjärjestelmän versiot tulevat tukemaan luotavaa ohjelmistoa. (Android for Developers 2022.)

3 KEHITYSTYÖKALUT

Projektin luontiin käytettiin paperisten luonnosten ja muistiinpanojen lisäksi tietokonetta sekä sovelluskehityksessä tarvittavia tekniikoita ja ohjelmistoja. Ohjelmointikielten, verkkopalveluiden ja ohjelmointiympäristön käyttöä varten haettiin ohjeita Internetistä.

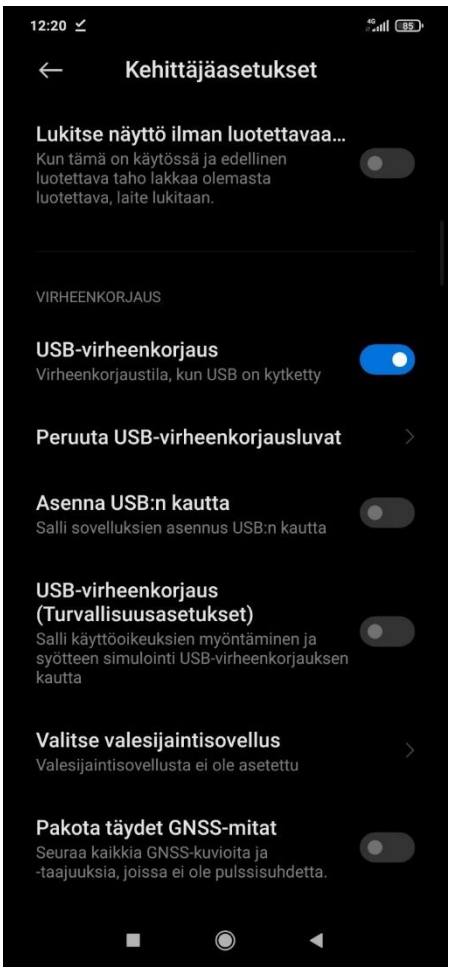
3.1 Android Studio

Android Studio on virallinen ohjelmointiympäristö Android-ohjelmistokehitykseen, ja se perustuu JetBrainsin IntelliJ-kehitysympäristöön. (WhatIs.com 2022; Mobiili.fi 2014.) Ympäristö tukee ohjelmistojen luomista mm. Java-, Kotlin-, C-, C++, ja C#-ohjelmointikielillä, joista Kotlin on ollut Googlen suosittelema kieli vuodesta 2019 lähtien (Android Authority 2019). Studio tarjoaa erilaisia projektipohjia sovelluksien tekoon puhelimille, taulutietokoneille, televisioille sekä älykelloille ja ajoneuvoille, ja lisäksi luotaessa projekteihin aktiviteetteja on mahdollista käyttää valmiiksi täydennettyjä malleja.

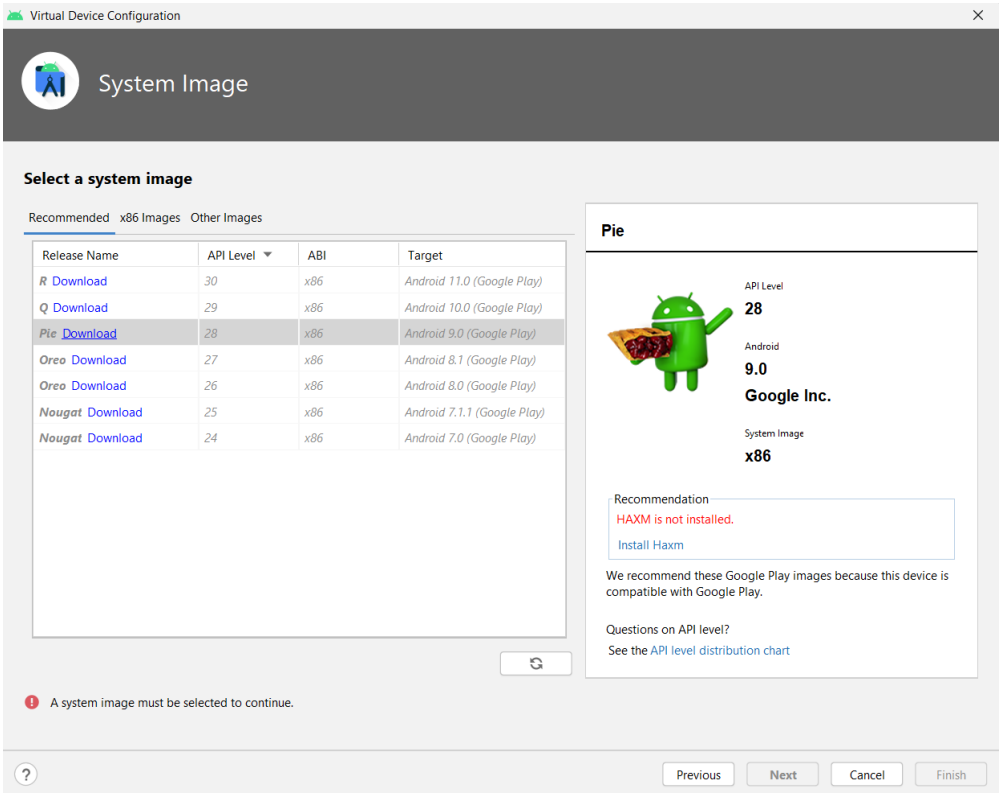


KUVA 2. Android Studion tarjoamia projekti- ja käyttöliittymäpohjia (Riihiluoma, 1.3.2022)

Android Studiolla kehitettäviä sovelluksia voidaan oikeiden Android-laitteiden lisäksi suorittaa virtuaalisesti ympäristön mukana asennettavalla Android Emulatorilla. Emulaattorilla on mahdollista simuloida useita erilaisia laitteita vaihtelevilla käyttöjärjestelmäversioilla, mikä helpottaa huomattavasti sovellusten koeajoa, kun laitteita ei tarvitse hankkia erikseen. Ohjelman ajaminen fyysisellä laitteella onnistuu liittämällä se tietokoneeseen USB-liitännällä, ja valitsemalla se käytettäväksi laitteeksi Android Studiossa. Laitteen asetuksista täytyy kuitenkin ensin ottaa kehittäjätila käyttöön, ja laitettava kehittäjäasetusvalikosta USB-virheenkorjausasetus päälle.



KUVA 3. Android-laitetta on mahdollista käyttää sovellusten koeajoon ottamalla kehittäjätila käyttöön. (Riihiluoma, 2.3.2022)



KUVA 4. Virtuaalista laitetta luotaessa voidaan laitemallin lisäksi valita haluttu käyttöjärjestelmän versio (Riihiluoma, 2.3.2022)

Android-sovellukset on allekirjoitettava digitaalisesti jaettaessa niitä muille laitteille. Koeajettaessa sovelluksia Android Studiossa ne allekirjoitetaan automaattisesti, mutta sovelluskauppoihin julkaisua varten on tehtävä erillinen konfiguraatio. (Android for Developers 2022.) Jaettavien ja asennettavien Android-sovellusten tiedostoformaatti on APK (Application Package Kit) (NextPit 2022).

3.2 Java

Java on Sun Microsystemsin vuonna 1995 julkaisema olio-ohjelmointikieli ja ohjelmistoalusta, jota käytetään useilla eri laitteistoilla. Java-ohjelmien suorittamiseen tarvitaan Java Runtime Environment -ohjelmisto (JRE), joka pitää sisällään Java Virtual Machine (JVM) -virtuaalikoneen sekä Java-kirjastoja. Java-ohjelmistojen kehittämistä varten on saatavilla Java Development Kit (JDK). (Java 2022.)

3.3 XML ja JSON

XML (Extensible Markup Language) on tekstiformaatti, jolla kuvataan järjestettyä tietoa. Toinen laajalti käytössä oleva formaatti on JSON (JavaScript Object Notation), jota käytetään tietojenkäsittelyssä XML:n tavoin datan lähettämisessä ja tallentamisessa. Tavanomaisia käyttökohteita kyseisille formaateille ovat mm. sovellusten väliset rajapinnat ja käyttöliittymät. (W3C 2015.)

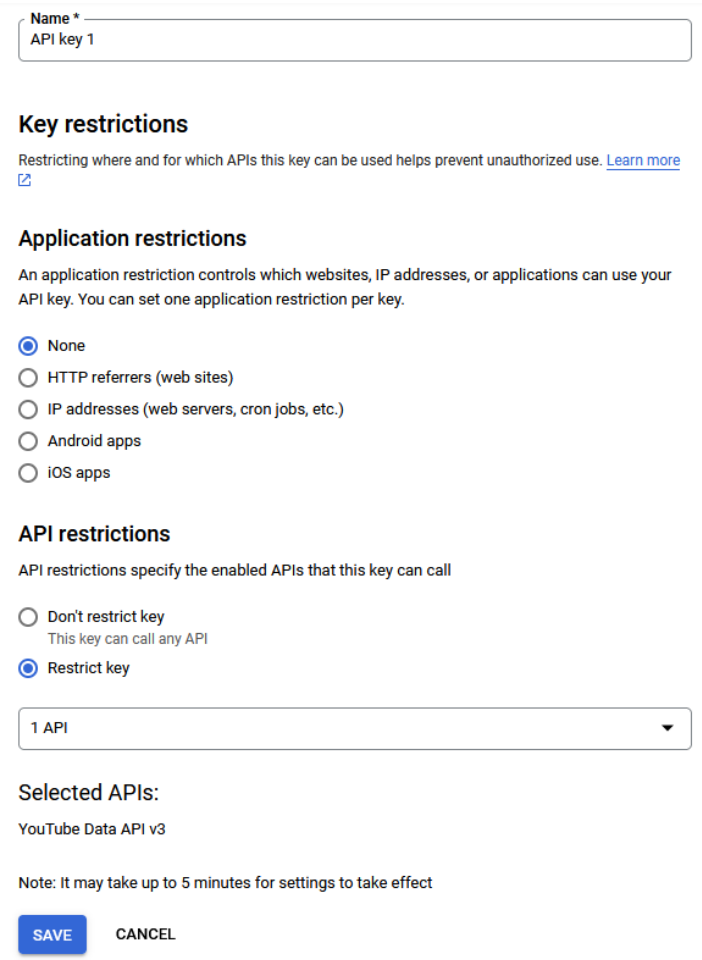
3.4 HTML, CSS ja JavaScript

HTML (Hypertext Markup Language) on verkkosivujen rakenteen ja sisällön luomiseen käytetty merkintäkieli, jota voidaan tulkita verkkoselainohjelmilla (Lellunpaja 2020). HTML-dokumenttien ulkoasu määräytyy Cascading Style Sheets (CSS) -tyyliohjekielen mukaan, jota käytetään mm. elementtien värin, koon, fontin ja asettelun määrittämiseen (MDN 2022).

JavaScript on ohjelmointikieli, jota käyttämällä verkkosivuista voidaan tehdä dynaamisia, mikä tarkoittaa niiden sisällön muuttumista ajankohdan tai käyttäjän toiminnan mukaan. JavaScript-koodilla sivustoille voidaan luoda toiminnallisuutta (Linkki 2019), johon voi kuulua esimerkiksi tiedon vastaanotto ja prosessointi, animaation suorittaminen tai HTML-elementtien ja niiden sisällön tai CSS-koodin manipulointi. Verkkoselainten lisäksi JavaScriptiä voidaan käyttää myös Node.js-ajoympäristössä palvelinten toiminnallisuuden ohjelmoimiseen (MDN 2022). HTML-, CSS- ja JavaScript-koodia käytetään yleensä yhdessä verkkosivujen luomiseen, ja niitä on mahdollista käyttää keskenään samoissa tiedostoissa.

3.5 YouTube Data API

YouTube Data API on Googlen tarjoama rajapinta, jonka avulla YouTube-videopalvelun toimintoja voidaan käyttää ohjelmistoissa. Rajapinnan käyttöön tarvitaan Google-tili, jolla kirjaudutaan Google Cloud Platform –verkkosovellukseen. (Google Developers 2020). Sovelluksessa on valittavissa useita eri rajapintoja eri käyttötarkoituksiin, joita voidaan käyttää luomalla projekti ja konfiguroimalla sille API-avain, jota käytetään tunnisteena tehtäessä kutsuja rajapintoihin.



KUVA 5. API-avaimille voidaan asettaa rajoituksia, jotka määrittävät, mistä ja mihin rajapintoihin niillä voidaan lähettää kutsuja (Riihiluoma, 25.3.2022)

3.6 Microsoft Azure

Microsoft Azure on Microsoftin omistama pilvipalvelu, joka tarjoaa laskentaan, analytiikkaan, tietovarastoihin ja verkon infrastruktuuriin liittyviä palveluita. Azuren järjestelmässä hyödynnetään virtualisointia, joka mahdollistaa usean tietokoneyksikön emuloimisen yksittäisellä palvelimella. (YouTube 2018; TechTarget 2020.) Palveluiden käytöstä veloitetaan kausittain sopimuksen mukaan, mutta Azure tarjoaa myös ilmaisia palveluita sekä kokeilujaksoja.

4 SUUNNITTELU

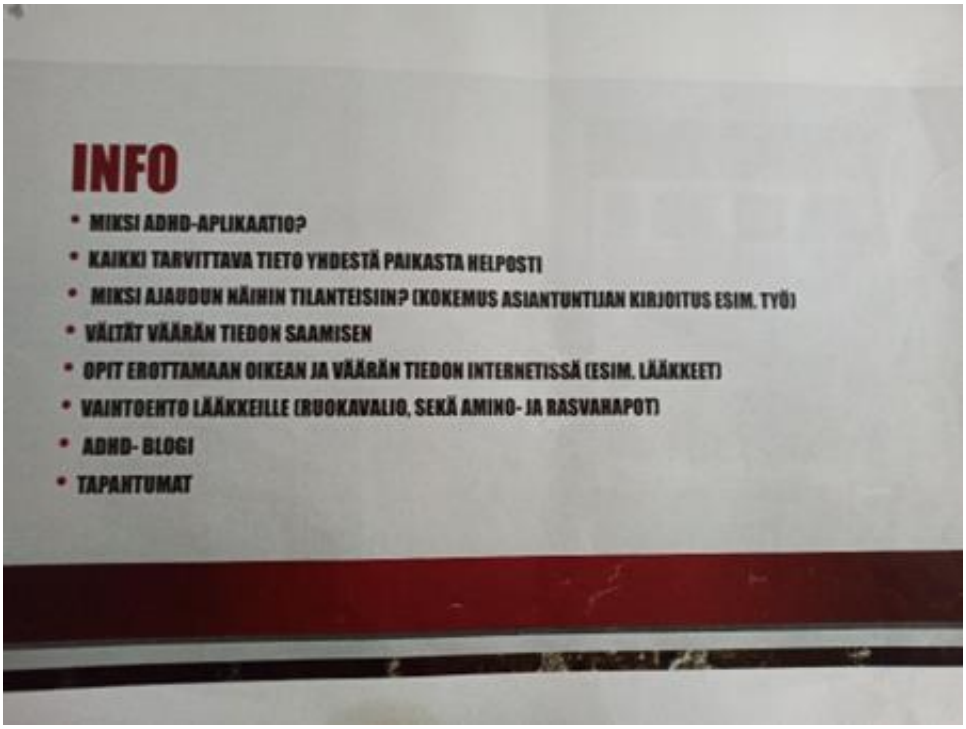
Projektin sisältöä ja työn kulkua suunniteltiin yhdessä tilaajan kanssa alusta alkaen. Suunnitelmissa huomioitiin työn valmisteluun liittyvät osa-alueet, joita pyrittiin noudattamaan sen edetessä.

4.1 Tavoitteet

Projektin suunnitteluvaiheessa sen tavoitteista keskusteltiin asiakkaan ja suunnittelijan välisissä tapaamisissa. Asiakas esitti sovelluksen ominaisuuksiin liittyvät toiveensa ja aikataulun tuotteen valmistumiseen. Sovelluksen kehittäjä esitti erilaisia tapoja toteuttaa ne, ja kuinka ne käytännössä voitaisiin sisällyttää ohjelmistoon. Todettiin, että projektin lopullisena tuotoksena tulisi olla asiakkaan tarpeisiin riittävä sovelluksen prototyyppi, joka toiminnoiltaan ja tekniikoiltaan samalla täyttäisi opintoihin kuuluvan lopputyön laajuuden vaatimukset. Sovelluksen valmistumisella ei olisi täsmällistä aikarajaa, vaan se valmistuisi suunnittelijan oman aikataulun mukaan.

4.2 Sovelluksen vaatimukset

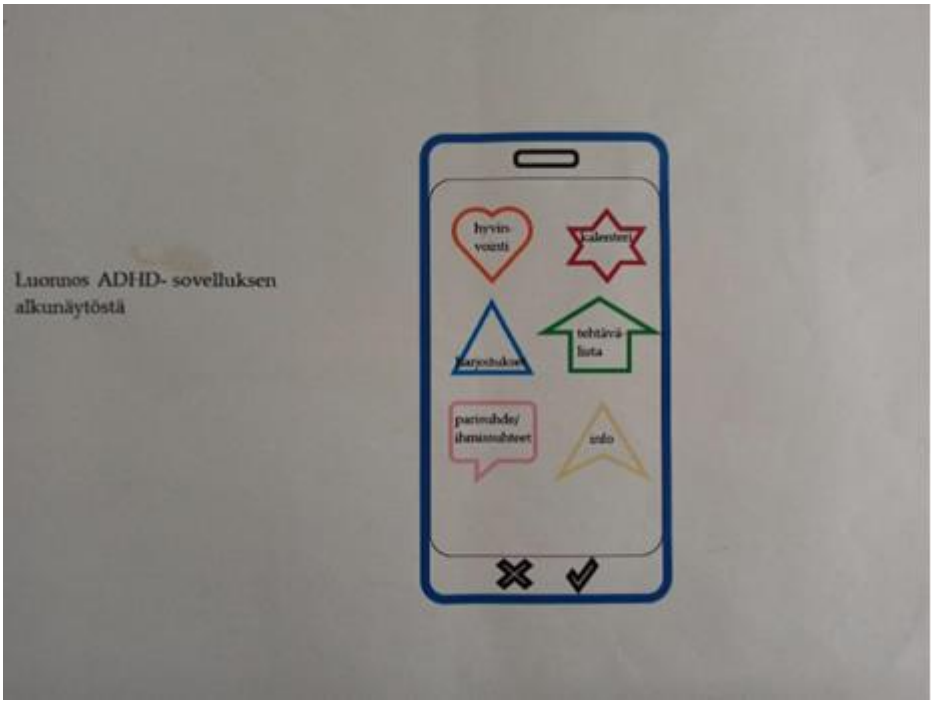
Asiakas luovutti sovelluksen luontia varten dokumentteja, joissa sen keskeisiä toimintoja ja rakennetta oli kuvattu muistiinpanoilla ja kuvallisilla esimerkeillä. Lisäksi asiakirjoissa oli esitetty syitä sovelluksen tarpeellisuudelle, ja kuinka siihen kuuluvat aihealueet voitaisiin ryhmitellä eri kokonaisuuksiksi. Luonnosten tarkoituksena oli auttaa saamaan visuaalinen näkemys myös sovelluksessa käytettävästä teemasta, jota muokattaisiin sopivammaksi sekä kehittäjän että asiakkaan yhteisten näkemyksien mukaan projektin etenemisen myötä.



KUVA 6. Muistiinpanoja liittyen sovellukseen mahdollisesti sisällytettävään tietoon (Riihiluoma, 7.9.2021)

Projektin suunnitteluvaiheessa todettiin, että sovelluksen ulkoasu ja sen sisältämät toiminnot tulisi esittää käyttäjälle mahdollisimman siistissä ja yksinkertaisessa muodossa. Pelkistetympi käyttöliit-

tymä ja sen grafiikka tulisivat mukailemaan sovelluksen käyttötarkoitusta, eli suuremmat koristeelliset elementit jätettäisiin pois. Otettaessa huomioon sovelluksen kohteena oleva käyttäjäryhmä, voidaan tämän todeta tarjoavan heille helpon käyttökokemuksen, ja samalla korostavan projektin tavoitetta: tavoitteena on tarjota helppokäyttöinen työkalu arkielämän hallintaan.

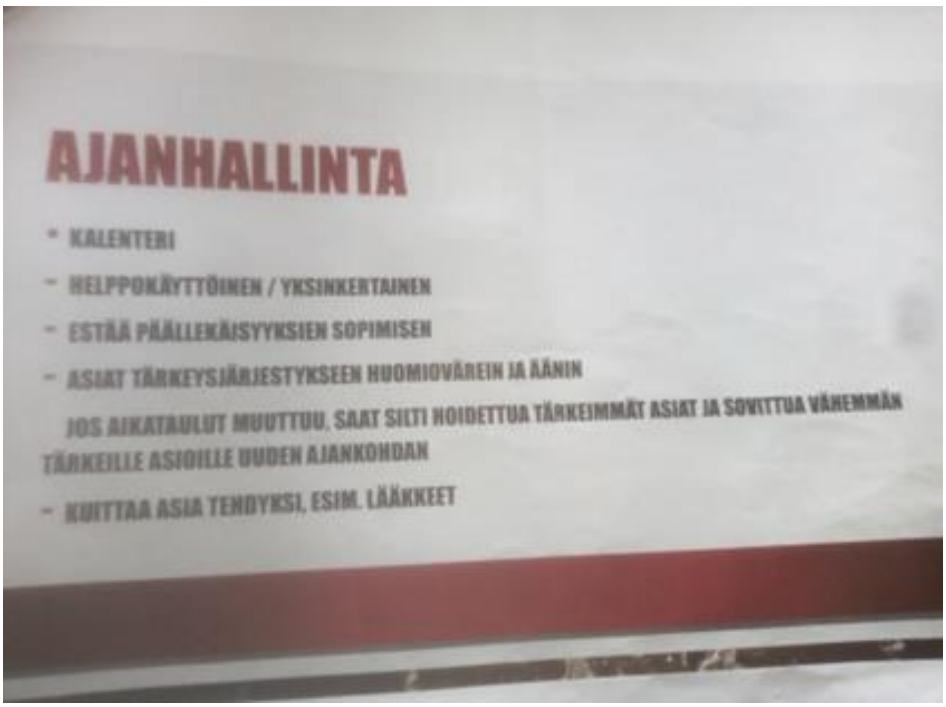


KUVA 7. Luonnos sovelluksen aloitusnäytöstä ja aihealueista (Riihiluoma, 7.9.2021)

4.3 Toiminnot ja käyttöliittymä

Sovellusta suunniteltaessa asiakas oli laatinut ehdotuksia siihen laitettavista toiminnoista, jotka sisältäisivät eri aihealueisiin liittyviä tietoja ja toimintoja. Nämä toiminnot tulisivat auttamaan käyttäjää suunnittelemaan päivittäisiä toimia tarjoamalla keinoja tehtävienhallintaa ja ajankäyttöä varten, sekä näyttämällä tietoa yleisistä keskittymishäiriöön liittyvistä asioista ja erilaisista keskittymistä helpottavista toimista.

Yksi tärkeimmistä sovellukseen kuuluvista työkaluista oli niin kutsuttu kalenteritoiminto, jolla käyttäjä voisi merkitä itselleen muistiin tulevia tapahtumia, ja joista sovellus antaisi hänelle myöhemmin muistutuksen. Tärkeinä yksityiskohtina kyseisessä toiminnossa olivat myös esto päällekkäisten tapahtumien luomiselle, sekä muistutusta vaativien asioiden luominen ja esittäminen käyttäjälle yksinkertaisesti ja tärkeysjärjestyksessä, esimerkiksi erilaisia huomiovärejä ja -ääniä hyödyntämällä. Lisäksi käyttäjän tulisi aikataulujen muuttuessa pystyttävä muuttamaan myös sovellukseen laittamiaan tietoja, ja kuittaamaan tekemänsä asiat tehdyiksi. Toinen ajankäyttöön liittyvä toiminto olisi tehtävälista, johon käyttäjä pystyisi kirjaamaan itselleen muistiin asioita, joita hän voisi kyetessään tehdä, ja näin estää tekemättömien töiden kerääntymisen.



KUVA 8. Sovelluksen ajanhallintatoimintoihin liittyviä muistiinpanoja (Riihiluoma, 18.9.2021)

Muina sovelluksen olennaisista toiminnoista tulisivat olemaan sivut, joista käyttäjä voisi oppia katsomalla ja kuuntelemalla keskittymistä helpottavia harjoitteita, ja lukemaan ADHD:hen liittyvää tietoa. Videon ja äänen muodossa esitetyt harjoitteet ja kirjallinen tieto tultaisiin jakamaan erillisille sivuille, joista käyttäjä voisi tarkastella haluamaansa aihetta. Infosivun - eli valikon, jossa kirjallinen tieto olisi esitettynä - pääasiallinen tarkoitus olisi tarjota kaikki keskittymishäiriöön liittyvä tieto käyttäjälle helposti yhdestä paikasta, ja samalla estää käyttäjää saamasta väärää tietoa aiheesta. Aloitusvaiheessa sovellukseen oli myös suunnitelmassa sisällyttää hyvinvointiin ja ihmissuhteisiin liittyvää materiaalia, mutta kyseiset aiheet jätettiin myöhemmin pois toteutuksesta.

4.4 Kommunikaatio

Projektin tilaajan ja kehittäjän väliseen kommunikointiin kuului tapaamisia, joissa suunniteltiin sovellukseen tarvittavia ominaisuuksia, tarkasteltiin toteutettuja asioita ja pohdittiin muutosten tarpeellisuutta työn etenemisen aikana. Edistymistä käytiin läpi myös sähköisen viestinnän kautta, joka mahdollisti nopean työhön kuuluvien tulosten esittämisen ja palautteen vastaanottamisen osapuolten välillä.

5 TOTEUTUS

Projektin valmistumisella ei ollut tarkkaa aikamäärettä, vaan sen kehitystyöstä vastaava pystyi tekemään suunnittelu- ja ohjelmointityötä oman aikataulunsa mukaan. Jo ennen työn aloittamista itse sovelluksen valmistuminen priorisoitiin, jotta asiakas saisi sen omaan käyttöönsä mahdollisimman pian.

5.1 Aloitusnäyttö

Käyttäjää varten oli tärkeää toteuttaa selkeä päävalikko, josta käsin sovelluksen eri toimintoja voisi hallinnoida. Aikaisessa kehitysvaiheessa toimintojen osa-alueet oli jaettu kuuteen ryhmään, minkä johdosta aloitusnäyttöön päätettiin sijoittaa kuusi eri painiketta, ja luonnostella niitä jokaista varten erilaiset kuvakkeet. Kehitystyön aikana käytettiin ajoittain tilapäistä grafiikkaa sovelluksen kuvittamiseen ennen lopullisen ulkoasun viimeistelyä. Tähän käyttötarkoitukseen ladattiin erilaisia kuvatiedostoja Internetistä, joita pystyttiin tarvittaessa muokkaamaan GIMP-kuvankäsittelyohjelman avulla halutun mallisiksi. Uusien kuvatiedostojen luontiin käytettiin myös Microsoft Paint -ohjelmaa.

KUVA 9. Microsoft Paint -ohjelmalla luonnosteltu logo sovellukselle (Riihiluoma, 23.11.2021)

Käyttöliittymän tekniikassa päätettiin käyttää Androidin sovelluskehityspakettiin kuuluvaa ImageButton-luokkaa, jota voidaan käyttää kuvallisten painikkeiden luomiseen. Kuvakkeen painaminen vei käyttäjän päävalikosta sen kuvaamaa toimintoa vastaavalle sivulle, josta pystyi palaamaan takaisin painamalla laitteen ´takaisin´-nappia. Sivujen välinen navigointi toteutettiin tekemällä niitä jokaista varten oma aktiviteetti. Aktiviteetti koostuu sen ohjelmakoodin sisältävästä Java-luokasta ja käyttöliittymän määrittävästä XML-tiedostosta. Toimiakseen aktiviteettia vastaava XML-elementti täytyy olla merkittynä sovelluksen olennaisia tietoja sisältävään XML-tyyppiseen AndroidManifest-tiedostoon, minkä Android Studio tekee automaattisesti uuden aktiviteetin luonnin yhteydessä (Android for Developers 2021). Aktiviteetteihin siirtyminen tapahtuu luomalla Intent-luokan ilmentymä ja antamalla sille kohteena olevaan aktiviteettiin liittyvät tiedot parametreinä. Niistä takaisinpäin navigoitaessa luodaan uusi ilmentymä, jota hyödyntäen aktiviteetin tila asetetaan lopetetuksi.

KUVA 10. Tilaajan antama kuvatiedosto aloitusvalikon käyttöliittymää varten (Riihiluoma, 23.11.2021).

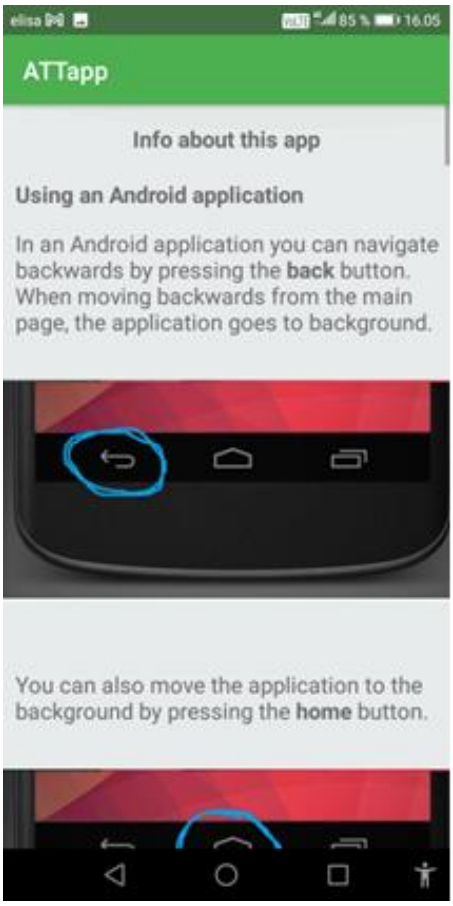
Päävalikkosivulle sijoitettiin myös asetusvalikko, josta käsin käyttäjällä oli mahdollisuus tarkastella sovellusta koskevia tietoja sekä muuttaa sovelluksen kieltä. Valikko tehtiin käyttämällä Androidin sovelluskehityksessä yleisesti käytettävää Android Option Menu -valikkoa, jota kuvaa näytön yläreunassa oleva kolmen pisteen kuvake. Sitä painamalla voi avata valikon, johon on mahdollista määrittää haluttu määrä valintoja ja valintaryhmiä.

```
@TargetApi(Build.VERSION_CODES.N)
private static Context updateResources(Context context, String lang) {

    Locale locale = new Locale(lang);
    Locale.setDefault(locale);

    Configuration config = context.getResources().getConfiguration();
    config.setLocale(locale);
    config.setLayoutDirection(locale);

    return context.createConfigurationContext(config);
}
```

KUVA 11. Java-metodi, jota käytetään sovelluksen lokalisaation vaihdossa (Riihiluoma, 2.2.2022

Sovellukseen päätettiin toteuttaa vaihtoehto muuttaa käyttöliittymän kieli suomeksi, englanniksi tai ruotsiksi, jonka käyttäjä voisi vaihtaa valikon kautta. Kielen vaihtaminen onnistui luomalla sitä varten luokan, joka sisälsi kieliasetuksen vaihtoon tarvittavat metodit, ja tekemällä jokaisen kielen käännöksiä varten omat XML-formaattiset tekstitiedostot. Androidissa kielten määrittämiseen käytetään kielikoodeja yhdessä Locale-luokan kanssa. Lisäksi valikkoon lisättiin valinta, joka avaa sovelluksen käyttöä kuvaavan tietosivun, jolla ohjeistetaan sen toimintojen käyttöä, sekä kaikkien Android-sovellusten hallinnointia Android-käyttöjärjestelmässä. Valikkoon lisättiin myös maininta sovelluksessa käytettyjen kuvakkeiden tekijästä.

KUVA 12. Lopullinen versio sovelluksen päävalikosta (Riihiluoma, 23.9.2021)



KUVA 13. Sovelluksen käyttöohjesivu (Riihiluoma, 23.9.2021)

5.2 Muistiinpanot

Sovellukseen oli määrä tehdä sivu, jolla käyttäjä voisi kirjoittaa itselleen muistiinpanoja, ja näin estää tekemättömien tehtävien kasautumista. Tehtävät asiat voisi halutessaan merkitä tärkeiksi, jolloin ne erottuisivat muista. Kuten kaikissa sovelluksen osa-alueissa, oli tärkeää myös suunnitella käyttöliittymä mahdollisimman selkeäksi käyttäjää varten.

Ensimmäisenä oli päätettävä tekniikka muistiinpanojen esittämiseen, ja mietittävä, miten se ja muut graafiset elementit esitettäisiin käyttäjälle. Toteutuksessa päätettiin esittää käyttäjän tallentamat tehtävät allekkain selattavan listan muodossa. Jos käyttäjä ei olisi luonut itselleen tehtäviä muistiin, hänelle näytettäisiin listan sijaan viesti, joka kertoisi, ettei tehtäviä ole kyseisellä hetkellä. Uuden tehtävän voisi luoda painamalla ponnahdusikkunan avaavaa nappia, jossa käyttäjä pystyisi syöttämään tehtävälleen nimen ja asettamaan sen joko tärkeäksi tai ei-tärkeäksi. Tärkeät tehtävät lajiteltaisiin listan ensimmäisiksi, ja ne kirjattaisiin punaisella tekstillä korostamaan niiden ensisijaisuutta muihin tehtäviin nähden. Listan ja muiden aktiviteetissa olevien graafisten elementtien toteuttaminen onnistui Androidin sovelluskehityspaketin tarjoamia Java-luokkia hyödyntämällä. Listassa olevien tehtävien esittämiseen käytettiin ListView-näkymää, johon käyttäjän luomat tapahtumiin liittyvät tiedot tallennettiin käyttämällä ArrayAdapter-, ArrayList-, Gson- ja SharedPreferences-luokkia.

KUVA 14. Muistiinpanosivu, jolle on laitettu tehtäviä esimerkiksi (Riihiluoma, 18.9.2021)

ArrayList-luokkaa hyödyntäen voidaan tallentaa useita muuttujia dynaamiseen taulukkoon, josta niitä pystytään hakemaan, muokkaamaan tai poistamaan. ArrayAdapter-luokan avulla ArrayList-luokan ilmentymään tallennettuja muuttujia voidaan muuntaa ListView-näkymässä esitettäviksi teksteiksi. Käyttäjän luomien tehtävien esittämistä varten tehtiin UserTask-niminen luokka, mikä mah-

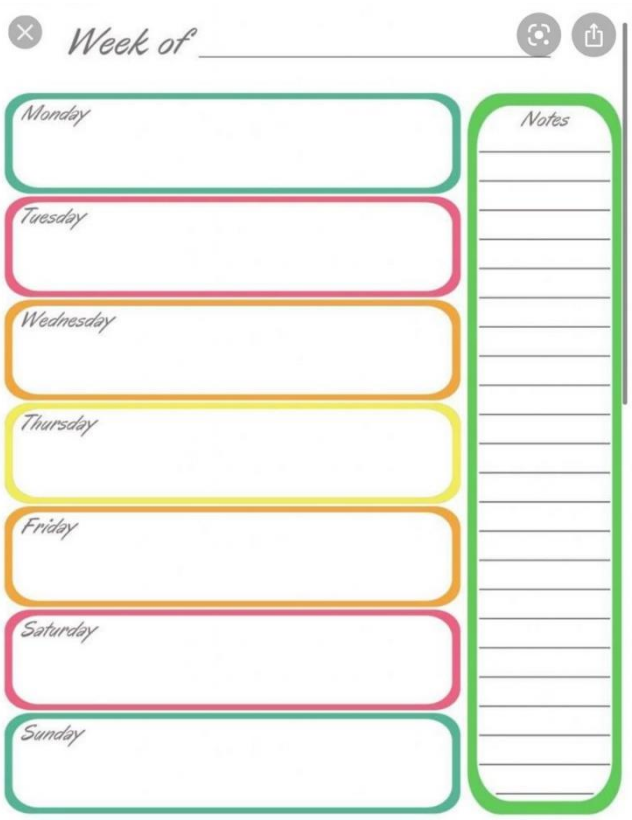
dollisti nimen ja tärkeyden määrittämisen tehtäville. Usean ArrayList-muuttujan tekeminen mahdollisti tehtävien lajittelemisen niiden tärkeyden perusteella yhteen taulukkoon, jossa ne olivat järjestetty luontiajankohdasta riippumatta niin, että tärkeiksi määritetyt tehtävät olivat ensimmäisenä. Antamalla järjestetty taulukko parametrina ArrayAdapter-muuttujalle, se saatiin esitettyä käyttäjälle ListView-näkymässä.

Toiminto uuden tehtävän luontiin toteutettiin lisäämällä käyttöliittymään nappi käyttämällä Button-luokkaa, sekä toteuttamalla erillinen DialogFragment-luokan perivä luokka, jota käytettiin tehtävän tekoon tarvittavan lomakkeen esittämiseen. DialogFragment-luokkaa käyttämällä käyttäjälle voidaan esittää ponnahdusikkuna, johon voidaan tavanomaisen aktiviteetin käyttöliittymän tavoin liittää erilaisia näkymiä ja viestejä. Lomakkeeseen laitettiin EditText-näkymä käyttäjän tehtävälle antaman nimen lukemista varten, sekä ToggleButton-näkymä tehtävän tärkeyden määrittämistä varten. Molemmille kentille annettiin myös nimiöt TextView-näkymillä niiden merkitysten selkeyttämiseksi käyttäjälle.

```
private void saveListData(ArrayList<UserTask> list) {

    SharedPreferences prefs = PreferenceManager.getDefaultSharedPreferences( context: this);
    SharedPreferences.Editor editor = prefs.edit();
    Gson gson = new Gson();
    String json = gson.toJson(list);
    System.out.println("\n" + json + "\n");
    editor.putString("taskList", json);
    editor.apply();
}
```

KUVA 15. Muistiinpanosivun luokkatiedostossa oleva luokan jäsenmetodi, joka tallentaa käyttäjän muistiinpanot sovelluksen muistiin (Riihiluoma, 16.11.2021)

Jotta käyttäjän luomat tehtävät eivät olisi kadonneet pois aktiviteetista navigoitaessa, ne tallennettiin sovelluksen muistiin merkkijonomuuttujana SharedPreferences-luokan avulla. Tehtävät saatiin tallennettua tekstimuotoon käyttämällä Gson-luokkaa, joka mahdollistaa ArrayList-muuttujaan tallennettujen arvojen muuntamisen JSON-tekstiformaattiin. Käyttäjän palatessa muistiinpanosivulle, tallennettu tekstimuuttuja haettiin sovelluksen muistista SharedPreferences-muuttujan avulla, ja se muunnettiin takaisin käyttäjälle esitettävään graafiseen muotoon. Sama tallennusprosessi suoritettiin myös aina käyttäjän tehdessä muutoksia olemassa olevaan listaan.



KUVA 16. Muistiinpanosivun graafisen ulkoasun päivittävä kuvake (Riihiluoma, 13.2.2022)

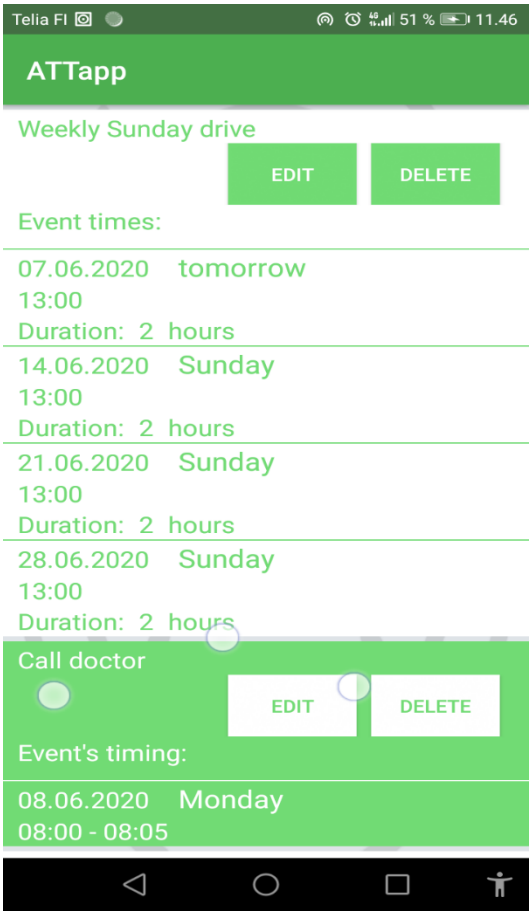
Sivun grafiikkaa haluttiin lopulta päivittää erilaiseksi, ja sen visuaalista ilmettä enemmän huolitelluksi. Aktiviteetin käyttöliittymä korvattiin muistiota esittävällä kuvakkeella, jonka tarkoitus oli toimia esimerkkinä näyttävämmästä toteutustavasta sovelluksen mahdollista jatkokehitystä ajatellen. Käyttöliittymän elementtejä varten on mahdollista tehdä erilaisia graafisia ulkoasuja ja animaatioita, mutta ajan säästämiseksi niitä ei käytetty projektissa. Sovelluksen oli tarkoitus toimia mallina sen konseptin esittelyssä, joten pelkän kuvan katsottiin olevan riittävä.

5.3 Kalenteri ja muistutukset

Työmäärältään laajin osa projektissa oli toiminnon teko sovellukseen, jonka avulla käyttäjä voisi luoda kalenteriin tapahtumia, joista annettaisiin myös muistutus ennen niiden alkamisajankohtaa. Toiminnon käyttötarkoituksena oli pohjimmiltaan suunnitteluvaiheessa ideoitu konsepti, jossa käyttäjä voisi muistuttaa itseään muun muassa lääkkeiden säännöllisestä ottamisesta. Kehitystyön aikana toiminnon käyttöliittymän ulkoasuun tehtiin muutoksia useaan kertaan sen etenemisen aikana, minkä seurauksena myös toiminnallisuus muuttui viimeisessä sovelluksen kehitysvaiheessa. Erilaisia ulkonäkövaihtoehtoja kokeiltiin sekä asiakkaan että kehittäjän aloitteesta molempien ehdotusten mukaisesti.

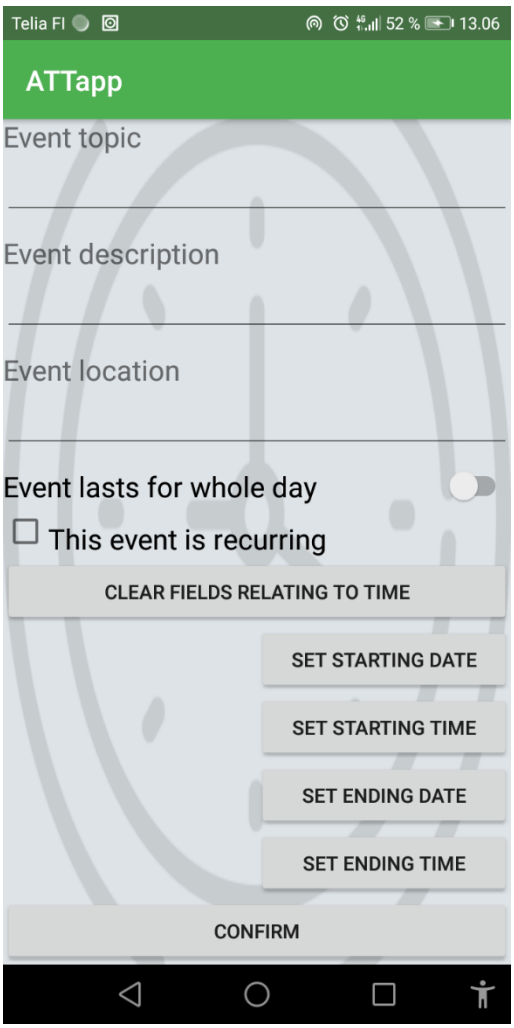
Kalenterisivun käyttöliittymän asettelu toteutettiin aluksi kehittäjän näkemyksen mukaan. Kalenterista haetut tapahtumat esitettiin käyttäjälle allekkain aikajärjestyksen mukaan listassa, jossa jokaisesta tapahtumasta näytettiin otsikko ja sijainti sekä sen ilmentymien ajankohdat ja kesto riippuen siitä, oliko tapahtuma asetettu toistuvaksi. Tapahtumat erotettiin toisistaan väriteeman avulla laittamalla joka toinen tapahtumien tiedot sisältävä näkymä listassa eri väriseksi lukemisen helpottamiseksi. Kaikille tapahtumille asetettiin listaan myös omat painikkeet, joista tapahtuman voisi poistaa

laitteen kalenterista tai sitä voisi muokata. Muokkaa-painikkeen painaminen avasi sivun alareunassa erilaisista näkymistä laaditun lomakkeen, jonka kentät olivat täytetty kyseisen tapahtuman tiedoilla. Samaa lomaketta käytettiin myös täysin uuden tapahtuman luomiseen, minkä käyttäjä pystyi tekemään painamalla Luo tapahtuma –painiketta.



KUVA 17. Lista puhelimen kalenterista haetuista tapahtumista (Riihiluoma, 9.2.2022)

Kalenterin hallinnoimiseen käytettiin Android-järjestelmään kuuluvaa Calendar Provider –ohjelmointirajapintaa, joka mahdollistaa laitteen sekä kolmansien osapuolten palveluihin liittyvien käyttäjätilien kalenterien tarkastelun ja muokkaamisen. Tapahtumien käsittelemistä varten päätettiin kuitenkin käyttää ainoastaan laitteen oman kalenterin tallennustilaa, eli käyttäjän tekemät ja muokkaamat tapahtumat tulisivat näkymään vain yksittäisellä laitteella, jolla sovellusta käytetään.



KUVA 18. Tapahtumien luomiseen ja muokkaamiseen käytettävä lomake (Riihiluoma, 9.2.2022)

Android-sovelluksessa käyttöliittymää hallitsevat toiminnot suoritetaan niin kutsutulla pääsäikeellä, joka tukkeutuessaan voi hidastaa sovelluksen toimintaa, tai jopa kaataa sen kokonaan (Android Authority 2018). Tästä syystä aikaa vievät prosessit, kuten Internettiin vietävät kutsut ja tietokanta-haut, täytyy suorittaa taustalla ajettavalla, erillisellä säikeellä. Tätä varten kalenteriaktiviteetissa käytettiin Androidin sovelluskehityspakettiin kuuluvaa AsyncTask-luokkaa, jota hyödyntäen kalenteria hallinoivaan rajapintaan kohdistuvat pyynnöt pystytään suorittamaan taustaprosessina. Haku-, luonti-, muokkaus- ja poistotoiminnoille toteutettiin omat AsyncTask-luokasta periytyvät sisäkkäiset luokat, joiden jäsenmetodeihin määritettiin niihin tarvittavat parametrit.

```
private class EventEdit extends AsyncTask<String, Void, int[]> {

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        System.out.println("Attempting to edit an event.");
    }
    @Override
    protected int[] doInBackground(String... params) {

        int eventId;
        long startTime, endTime, recStartTime;
        String title, location, recur, duration, timezone;
        int allDay;

        eventId = Integer.parseInt(params[0]);
        title = params[1];
        location = params[2];
        startTime = getTimestampFromFormattedDateString(params[3]);
        endTime = getTimestampFromFormattedDateString(params[4]);
        recStartTime = getTimestampFromFormattedDateString(params[5]);
        duration = params[6];
        recur = params[7];
        allDay = Integer.parseInt(params[8]);
```

KUVA 19. Sovelluksen kalenterisivun luokkatiedostossa oleva sisäkkäinen luokka, jonka ilmentymän execute-metodia kutsumalla voidaan muokata kalenterissa olevaa tapahtumaa (Riihiluoma, 10.11.2021)

Kalenterissa olevien tapahtumien hakemiseen liittyviä toimintoja varten tehtiin kolme AsyncTask-luokan perivää aliluokkaa. InstanceSearch-luokkaa käytettiin kalenterista löytyvien tapahtumien ilmentymien hakemiseen 28 päivän ajalta. Tapahtuman ilmentymällä tarkoitetaan sen ajankohtaa, joita yhdellä tapahtumalla voi olla yksi tai useampia sen toistuvuuden mukaan. Ilmentymien tietojen käsittelyä varten luotiin FetchedInstance-niminen luokka, joka piti sisällään niihin liittyvien rajapinnan kautta saatujen relaatiotietokannan taulujen kenttiä vastaavat muuttujat, joihin tallennettiin niiden kokonaislukutunnisteet sekä aloitus- ja lopetusajankohdat.

Tapahtumien muiden tietojen hakemista varten toteutettiin EventSearch-luokka. Käyttämällä löydettyjen ilmentymien kokonaislukutunnisteita voitiin kalenterista hakea myös kaikki niitä vastaavien tapahtumien tiedot, joiden käsittelyä varten tehtiin FetchedEvent-niminen luokka. Tämän luokan muuttujiin tallennettiin tapahtuman tunnisteet, otsikko, sijainti ja sen keston pituus. Muodostamalla löydetyistä tapahtumista ja niiden ilmentymistä ArrayList-tyyppisen taulukon, ne voitiin lajitella käyttöliittymässä listan muotoon ListView-näkymään käyttämällä RecyclerView-luokkaa.

Tapahtumien luomista, muokkaamista ja poistamista varten toteutettiin EventCreation-, EventEdit- ja EventDelete-luokat. Käyttäjä pystyi tekemään uuden tapahtuman kalenteriin avaamalla sivun alalaitaan lomakkeen, ja täyttämällä sen haluamillaan tiedoilla. Tapahtumalle oli annettava luotaessa ainakin otsikko ja aloitusaika. Toistuvaa tapahtumaa luotaessa oli määritettävä myös sen kesto ja päivämäärä, johon asti se tulisi toistumaan, tai tapahtuman toistumien lukumäärä. Jotta käyttäjän

luoman tapahtuman ajankohta ei olisi mennyt päällekkäin toisten tapahtumien kanssa, oli suoritettava tarkistusprosessi, jota varten tehtiin EventCheck-niminen aliluokka. Luokkaa käytettiin aina käyttäjän tehdessä tai muokattaessa tapahtumaa. Tarkistuksessa kalenterista haettiin kaikki tapahtumien ilmentymät pitkältä aikaväliltä ja verrattiin niitä vasta luotujen ilmentymien aloitus- ja lopetusajankohtiin. Jos luonnissa tai muokkauksessa kalenteriin syötetyn ilmentymän aloitus- tai lopetusaika sijoittui jonkin kalenterissa jo olevan ilmentymän aloitus- ja lopetusajan väliin, katsottiin kyseisen tapahtuman menevän päällekkäin toisen tapahtuman kanssa, ja se poistettiin kalenterista EventDeleteAfterCheck-luokkaa käyttämällä tapahtuman kokonaislukutunnisteen perusteella. Tieto tapahtumien onnistuneesta luonnista, muokkauksesta, poistamisesta tai niiden päällekkäisyyksistä esitettiin käyttäjälle ponnahdusikkunassa DialogFragment-luokan perivien aliluokkien avulla.

Samoin kuin muistiinpanojen tekoa varten olevalla sivulla, asettelua päätettiin myöhemmin muuttaa yksinkertaisempaan, graafisesti miellyttävämpään versioon, jossa käyttöliittymän tekstipainotteisia elementtejä oli vähennetty. Kalenterin tapahtumista koostuva lista korvattiin seinäkalenteria esittävällä kuvalla. Hälytystoiminto ja tapahtumien luontiin käytettävä lomake jätettiin entiselleen, koska sille ei aloitettu enää suunnittelemaan korvaavaa esitystapaa.

KUVA 20. Kalenterisivulle sijoitettu kuvake (Riihiluoma, 9.2.2022)

5.4 Harjoitusvideot

Sovellukseen tehtiin myös sivu, jolla voisi katsella erilaisia keskittymishäiriön hallintaan auttavia harjoitusvideoita. Android-alustalla on mahdollisuus käyttää lukuisia eri tekniikoita videoiden näyttämiseen, joten kehitystyön aikana jouduttiin pohtimaan, mitkä niistä soveltuisivat parhaiten tähän projektiin.

```
function getVideoData(id) {

	if (id != "") {

		var basePlaylistUrl = "https://www.googleapis.com/youtube/v3/playlistItems";
		var part = "?part=snippet%2CcontentDetails%2Cstatus";
		var playlistId = "&playlistId=".concat(id);
		var key = "&key=AIzaSyBICrgVnaEyNpDfHkOB7jt55djtX3FleNE";
		var url = basePlaylistUrl.concat(part);
		url = url.concat(playlistId);
		url = url.concat(key);

		performQuery(url, valuateResponseState, parseVideoData, drawVideoLinks);
	}
}
function performQuery(u, callback, callbackTwo, callbackThree) {

	var http = new XMLHttpRequest();
	http.open("GET", u);
	http.send();

	http.onreadystatechange = function() {

		if (callback(http.readyState)) {

			// passing response and callback function to parse function
			callbackTwo(http.responseText, callbackThree);
		}
	}
}
```

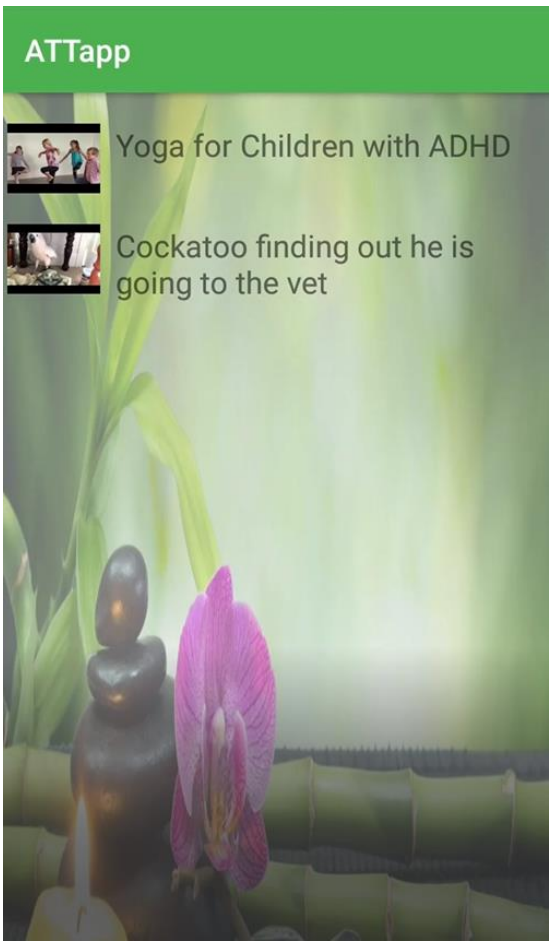
KUVA 21. Videoiden tietojen haun suorittavia JavaScript-funktiota (Riihiluoma, 19.3.2022)

Sivulle haluttiin lista tarjolla olevista harjoitusvideoista, joista käyttäjä voisi valita katsottavan videon. Asettelua päädyttiin kokeilemaan ensin toteuttamalla videolista HTML-sivuna, joka esitettiin käyttäjälle hänen siirtyessään harjoitusvideoaktiviteettiin. HTML-sivulle siirtyminen tapahtui avaamalla se Google Chrome -verkkoselainsovelluksessa antamalla sivun verkko-osoite URI-tyyppisenä parametrinä Intent-luokan ilmentymälle. Videot ja niihin liittyvät tiedot haettiin YouTube-videopalvelun rajapinnasta JSON-tekstiformaattina, josta videoiden otsikot ja kuvakkeet parsittiin allekkain listan muotoon HTML-elementteihin. Videon kuvaketta painamalla käyttäjä siirtyi toiselle sivulle, johon YouTube-video oli upotettuna. HTML-sivuille siirtyminen sovelluksesta tehtiin mahdolliseksi isännöimällä ne julkisella palvelimella Microsoft Azure –palvelussa. Edellä kuvattu toteutus päädyttiin kuitenkin hylkäämään palvelimen käytöstä aiheutuvien kulujen vuoksi, ja koska kyseinen toteutustapa pystyttiin korvaamaan yksinkertaisemmalla menetelmällä.

```
	<script type="text/javascript">

		document.addEventListener("DOMContentLoaded", function(event) {

			if (typeof(Storage) !== "undefined") {

				console.log("id: " + sessionStorage.getItem("watchVideoId"));
				var id = sessionStorage.getItem("watchVideoId");
				document.getElementById("videoFrame").src = "https://www.youtube.com/embed/".concat(id + "?rel=0");
			} else {
				document.getElementById("frame").removeChild(document.getElementById("videoFrame"));
				var pError = document.createElement("p");
				pError.innerHTML = "This browser does not support web storage.";
				document.getElementById("frame").appendChild(pError);
			}
		});
	</script>
	<style>
		html, body {
			height:100%;
		}
		.container {
			height:100%;
			width:100%;
		}
		iframe {
			height:100%;
			width:100%;
		}
	</style>
</head>
<body>
	<div class="container" id="frame">
		<iframe id="videoFrame" allow="autoplay" allowFullScreen ></iframe>
	</div>
</body>
```

KUVA 22. Videosivun HTML-, CSS- ja JavaScript-koodia (Riihiluoma, 19.3.2022)

Sivun toteutusta päädyttiin muuttamaan niin, että harjoitusvideot näytettiin sovelluksen aktiviteetin käyttöliittymässä verkkosivun sijaan. Videoiden tietojen hakemiseen rajapinnasta käytettiin HTTP-kutsuihin käytettävää Volley-kirjastoa, jonka JsonObjectRequest-luokan avulla voidaan vastaanottaa JSON-formaattinen tekstiparametri. Videoiden tietojen käsittelyä varten luotiin VideoData- ja Thumbnail-luokat, joiden jäsenmuuttujiin tallennettiin videoiden tunnisteita, otsikoita ja kuvakkeita koskevat tiedot. Kuvakkeiden lataamiseen tehtiin AsyncTask-luokasta johdettu DownloadImageTask-luokka, joka haki kuvatiedostot niiden URL-osoitteistaan. Videot esitettiin allekkain lisäämällä niiden kuvakkeille ja otsikoille käyttöliittymään näkymät dynaamisesti videoiden määrän mukaan. Jokainen rivi generoitiin ImageView- ja TextView-näkymästä, jotka lisättiin niitä ympäröivään LinearLayout-näkymään. Rivielementtiin kohdistuvalle painamiselle asetettiin kuuntelija, jota hyödyntäen videoiden tunnisteet saatiin haettua taulukon indeksistä rivinumeron perusteella. Valitun videon katselu tapahtui erillisessä aktiviteetissa hakemalla käyttäjän valitsema video YouTube-palvelusta Intent-luokan avulla vastaanotetun tunnisteen mukaan.



KUVA 23. Harjoitusvideosivulla videot ovat listattuna allekkain, joista käyttäjä voi valita haluamansa videon (Riihiluoma, 9.2.2022)

Videolistaan haettiin myös ADHD:hen liittymättömiä satunnaisia videoita, joiden tarkoitus oli toimia esimerkkeinä käyttöliittymää ja ohjelmakoodia luotaessa. Käytössä olevassa sovelluksessa sen ylläpitäjät voisivat helposti esittää haluamiaan videoita sovelluksessa luomalla sitä varten YouTube-kanavan, jolloin videot voitaisiin hakea kyseisen kanavan tunnistetta käyttäen.

5.5 Tietomateriaali

Sovellukseen luotiin aktiviteetti, jossa esitettiin ADHD:tä käsittelevää tietoa. Tietomateriaalina päädyttiin käyttämään Käypä hoito –verkkosivun ADHD:hen liittyvää tietoa, joka on kansallisia hoitosuosituksia terveydenhuollon tueksi laativa verkkosivusto (Käypä hoito 2020). Käyttäjän siirtyessä aktiviteettiin sivusto avattiin antamalla sen URL-osoite Intent-luokan ilmentymälle, ja avaamalla se verkkoselainsovelluksessa.

6 YHTEENVETO

Sovelluksen kehittämisvaihe oli pitkä ja hitaasti etenevä. Syinä tähän olivat puutteellisuus projektin kehittäjän tietotaidossa sekä motivaatiossa opiskella työn toteuttamiseksi vaadittua materiaalia ja tehdä sovelluskehitystyötä. Tästä syystä myös tilaajan ja kehittäjän välisessä kommunikaatiossa kehittäjän tilaajalle antama informaatio projektin etenemisestä jäi ajoittain vähäiseksi johtuen jaksoista, joiden aikana projektin edistämiseksi tehtyä työtä ei tapahtunut projektin kehittäjän osalta lainkaan. Työn etenemisen hitaus johti myös tilanteisiin, joissa kehittäjä jatkoi sovelluksen ominaisuuksien tekemistä liian pitkään omien näkemystensä mukaisiksi. Näin ollen voidaan todeta, ettei projektin etenemisen aikatauluttamisessa onnistuttu alkuperäisen suunnitelman mukaan.

Lopulta tilaajan käyttöön saatiin kuitenkin prototyyppiversiot sovelluksesta, joita pystyttiin käyttämään konseptin esittelyyn. Asiakkaalle luovutettiin kahden hieman toiminnallisuuksiltaan erilaisten sovelluksen versioiden projektitiedostot, ja lisäksi ohjelmistosta kuvattiin esittelyvideo. Asiakas ilmoitti osallistuneensa sovelluksen kanssa kilpailuun ja saaneen siitä palkinnon, joten projektin tulosten voidaan katsoa olevan varsin riittäviä. Sovellusta oli mahdollista jatkokehittää eteenpäin tekemällä siihen enemmän toimintoja, ja etsimällä tai luomalla uusia graafisia elementtejä ja animaatioita sovellukseen sen käyttöliittymästä voisi tehdä myös näyttävämmän.

Projekti tarjosi hyvän mahdollisuuden harjoitella ohjelmiston kehitystä asiakkaan toiveiden mukaisesti. Kehittäjän näkökulmasta katsottuna opittiin, että muita osapuolia tulee tiedottaa usein työn edistymisestä, jotta muutoksia voidaan tarvittaessa tehdä ajoissa ajan säästämiseksi. Projektissa saatiin opeteltua myös Javan ohjelmointitekniikoiden käyttöä Android-ympäristössä, ja harjoitettua tiedon etsintää ja sen soveltamista. Erityisesti sovelluksen kalenteritoimintojen ohjelmointi harjoitti ohjelmiston suunnittelussa tarvittavaa ongelmanratkaisukykyä.

LÄHTEET

Googlen blogi 2019. Kuva Androidin logosta. Verkkojulkaisu. Blog.google – Blogi Android-brändistä. Julkaistu 22.8.2019. https://blog.google/products/android/evolving-android-brand. Viitattu 27.5.2021.

Elprocus 2021. What is an Android Operating System & Its Features. Verkkojulkaisu. Elprocus.com - Tietotekniikkaa käsittelevä verkkosivusto. Päivitetty 2021. https://www.elprocus.com/what-is-android-introduction-features-applications. Viitattu 22.5.2021

Investopedia 2021. Android Operating System. Verkkojulkaisu. Investopedia.com - Markkinoita käsittelevä verkkosivusto. Päivitetty 3.2.2021. https://www.investopedia.com/terms/a/android-operating-system.asp. Viitattu 22.5.2021.

Teemu Tynkkysen kandidaatintyö 2018. Android-Ekosysteeemin Historia, Nykyisyys ja Tulevaisuudenskenaariot. Verkkojulkaisu. Lutpub.lut.fi - Lappeenrannan yliopiston kandidaatin tutkintojen opinnäytetyöt. https://lutpub.lut.fi/bitstream/handle/10024/158386/Android-ekosystee-min%20historia%20nykyisyys%20ja%20tulevaisuudenskenaariot.pdf?sequence=1&isAllowed=y. Viitattu 25.5.2021.

Android for Developers 2021. Platform Architecture. Verkkojulkaisu. Developer.android.com – Kehittäjille suunnattu verkkosivusto. Päivitetty 11.3.2021. https://developer.android.com/guide/platform. Viitattu 25.5.2021.

Wikiopisto 2019. Android arkkitehtuuri. Verkkojulkaisu. Fi.wikiversity.org – Avoin ja vapaa oppimisen verkkoyhteisö. Päivitetty 19.7.2019. https://fi.wikiversity.org/wiki/Android_arkkitehtuuri. Viitattu 25.5.2021.

Android Authority 2018. Verkkojulkaisu. Androidauthority.com - Android-käyttöjärjestelmää käsittelevä verkkosivusto. Päivitetty 4.5.2018. https://www.androidauthority.com/kotlin-coroutines-asynchronous-programming-858566. Viitattu 6.1.2022.

Käypä hoito 2020. Verkkojulkaisu. Kaypahoito.fi - Hoitosuosituksia laativia verkkosivusto. Päivitetty 26.6.2020. https://www.kaypahoito.fi/kaypa-hoito. Viitattu 1.2.2022.

Android for Developers 2020. Get started with the NDK. Verkkojulkaisu. Developer.android.com - Kehittäjille suunnattu verkkosivusto. Päivitetty 30.9.2020. https://developer.android.com/ndk/guides. Viitattu 23.2.2022.

JavatPoint 2021. Android Versions. Verkkojulkaisu. Javatpoint.com - Kursseja tarjoava verkkosivusto. Päivitetty 2021. https://www.javatpoint.com/android-versions. Viitattu 1.3.2022.

Android for Developers 2022. What is API level? Verkkojulkaisu. Developer.android.com - Kehittäjille suunnattu verkkosivusto. Päivitetty 9.2.2022. https://developer.android.com/guide/topics/manifest/uses-sdk-element#ApiLevels. Viitattu 1.3.2022.

NextPit 2022. What is an APK file and how to install AKPs on Android? Verkkojulkaisu. Nextpit.com - Teknologiaa ja elektroniikkaa käsittelevä sivusto. Päivitetty helmikuussa 2022. https://www.nextpit.com/android-for-beginners-what-is-an-apk-file. Viitattu 1.3.2022.

WhatIs.com 2022. APK file (Application Package Kit file format). Verkkojulkaisu. Whatis.techtarget.com - Tietotekniikkaan liittyvää opetusmateriaalia tarjoava sivusto. Päivitetty 2022. https://whatis.techtarget.com/definition/APK-file-Android-Package-Kit-file-format. Viitattu 1.3.2022.

Mobiili.fi 2014. Google julkaisi Android Studio 1.0:n. Verkkojulkaisu. Mobiili.fi - Mobiilimaailman uutisia ja tapahtumia käsittelevä sivusto. Päivitetty 9.12.2014. https://mobiili.fi/2014/12/09/google-julkaisi-android-studio-1-0n/. Viitattu 1.3.2022.

Android Authority 2019. I want to develop Android apps — What languages should I learn? Verkkojulkaisu. Androidauthority.com - Android-käyttöjärjestelmää käsittelevä verkkosivusto. Päivitetty 10.8.2019. https://www.androidauthority.com/develop-android-apps-languages-learn-391008/. Viitattu 2.3.2022.

Android for Developers 2022. Sign your app. Verkkojulkaisu. Developer.android.com - Kehittäjille suunnattu verkkosivusto. Päivitetty 11.2.2022. https://developer.android.com/studio/publish/app-signing. Viitattu 9.3.2022.

YouTube 2018. How does Microsoft Azure work? Verkkojulkaisu. Youtube.com - Googlen omistama videopalvelu. Päivitetty 18.6.2018. https://www.youtube.com/watch?v=KXkBZCe699A. Viitattu 13.3.2022.

TechTarget 2020. Microsoft Azure. Verkkojulkaisu. Techtarget.com - IT-alan markkinoista tietoa tarjoava sivusto. Päivitetty huhtikuussa 2020. https://www.techtarget.com/searchcloudcomputing/definition/Windows-Azure. Viitattu 13.3.2022.

Java 2022. What is Java technology and why do I need it? Verkkojulkaisu. Java.com - Java-ohjelmointikielen kotisivu. Päivitetty 2022. https://www.java.com/en/download/help/whatis_java.html. Viitattu 13.3.2022.

Google Developers 2020. YouTube Data API Overview. Verkkojulkaisu. Developers.google.com - Googlen tuotteiden dokumentaatioita tarjoava sivusto. Päivitetty 29.7.2020. https://developers.google.com/youtube/v3/getting-started. Viitattu 13.3.2020

W3C 2015. XML Essentials. Verkkojulkaisu. W3.org - Kansainvälinen Internet-standardeja kehittävä yhteisö. Päivitetty 2015. https://www.w3.org/standards/xml/core. Viitattu 14.3.2022.

Lellunpaja 2020. HTML - Mikä se on. Verkkojulkaisu. Lellunpaja.fi - Digitaalisuudesta, nettisivuista ja sosiaalisesta mediasta tietoa tarjoava yritys. Päivitetty 25.2.2020. https://www.lellunpaja.fi/html-alkeet.html. Viitattu 15.3.2022.

MDN 2022. CSS: Cascading Style Sheets. Verkkojulkaisu. Developer.mozilla.org - Verkkosovellusten teknologioita dokumentoiva sivusto. Päivitetty 21.1.2022. https://developer.mozilla.org/en-US/docs/Web/CSS. Viitattu 15.3.2022.

MDN 2022. Node.js. Verkkojulkaisu. Developer.mozilla.org - Verkkosovellusten teknologioita dokumentoiva sivusto. Päivitetty 18.2.2022. https://developer.mozilla.org/en-US/docs/Glossary/Node.js. Viitattu 16.3.2022.

Linkki 2019. Verkkosivut ja ohjelmointi. Verkkojulkaisu. Kirjat.it.jyu.fi/linkki - Digitaalinen oppimateriaali tieto- ja viestintätekniikkaan. Päivitetty 2019. https://kirjat.it.jyu.fi/linkki/nettisivut_ohjelmointi/javascript.html. Viitattu 16.3.2022.

MOOC julkaisuaika tuntematon. Luokka ja olio. Verkkojulkaisu. Mooc.fi - Helsingin yliopiston tietojenkäsittelytieteen osaston verkkokursseja tarjoava sivusto. https://ohjelmointi-19.mooc.fi/osa-4/3-luokka-ja-olio. Viitattu 18.3.2022.

MOOC julkaisuaika tuntematon. Johdatus olio-ohjelmointiin. Verkkojulkaisu. Mooc.fi - Helsingin yliopiston tietojenkäsittelytieteen osaston verkkokursseja tarjoava sivusto. https://ohjelmointi-20.mooc.fi/osa-4/1-johdatus-olio-ohjelmointiin. Viitattu 18.3.2022.

Käypä hoito 2019. ADHD (aktiivisuuden ja tarkkaavuuden häiriö). Verkkojulkaisu. Kaypahoito.fi - Hoitosuosituksia laativia verkkosivusto. Päivitetty 4.4.2019. https://www.kaypahoito.fi/hoi50061. Viitattu 18.3.2022.

MOOC julkaisuaika tuntematon. Metodit ja ohjelman jakaminen pienempiin osiin. Verkkojulkaisu. Mooc.fi - Helsingin yliopiston tietojenkäsittelytieteen osaston verkkokursseja tarjoava sivusto. https://ohjelmointi-20.mooc.fi/osa-2/4-metodit. Viitattu 18.3.2022.

Android for Developers 2022. View. Verkkojulkaisu. Developer.android.com - Kehittäjille suunnattu verkkosivusto. Päivitetty 17.3.2022. https://developer.android.com/reference/android/view/View. Viitattu 18.3.2022.