

TALLINNA ÜLIKOOL

Haapsalu kolledž

Rakendusinformaatika

Jaanus Saago

HAAPSALU KOLLEDŽI TUNNIPLAANI NUTISEADME KUVA LOOMINE

Diplomitöö

Juhendaja: Priidu Paomets, MSc

Haapsalu 2023

SISUKORD

MÕISTED JA KASUTATUD LÜHENDID.....	4
SISSEJUHATUS.....	6
1. HAAPSALU KOLLEDŽI TUNNIPLAANI NUTISEADME KUVA RAKENDUSE PLANEERIMINE	8
1.1. Valikpraktika õppeaine raames läbiviidud uurimus ja selle tulemused	8
1.1.1. Uurimismetoodika ja andmete kogumine	8
1.1.2. Tulemused ja analüüs	9
1.2. Valikpraktika õppeaine raames loodud veebirakenduse prototüüp.....	9
2. MOBIILIRAKENDUSE ARENDAMISE VÕIMALUSED	11
2.1. Põlirakendus	11
2.2. Hübriidrakendus	14
2.3. Veebirakendus	16
2.4. Järeldus.....	18
3. ÜLEVAADE KASUTATAVATEST LÄHENEMISTEST JA TEHNOLOOGIAST....	19
3.1. Kasutusmugavus.....	19
3.1.1. Kasutajakogemus	19
3.1.2. Kasutatavus	20
3.1.3. Kasutajaliidese disain.....	21
3.2. Kasutatav tehnoloogia	22
4. HAAPSALU KOLLEDŽI TUNNIPLAANI NUTISEADME KUVA RAKENDUSE TEOSTUS	23
4.1. Nutiseadme kuva disain ja kasutusmugavus	23
4.2. Prototüüp	27
4.3. Arendus	28
4.3.1. Tailwind CSS-i lisamine	28
4.3.2. Progressiivse veebirakenduse loomine.....	30
4.4. Testimine	31
4.5. Tulemus ja võimalik edasiarendus	36
KOKKUVÕTE.....	37
SUMMARY	38

KASUTATUD ALLIKAD.....	39
LISA 1. VALIKPRAKTIKA ÕPPEAINE ANKEETKÜSIMUSTIK	44
LISA 2. RAKENDUSE KIRJELDUS PROGRESSIIVSELE VEEBIRAKENDUSELE	47
LISA 3. TESTIMISÜLESANDED	48

MÕISTED JA KASUTATUD LÜHENDID

CSS: (*Cascading Style Sheets*) laadistik, mille abil kirjeldatakse HTML dokumentide väljanägemist (Vallaste, s.a.).

Hübriidrakendus: (*Hybrid app*) veebirakendus, mis on paigaldatud kestrakenduse sisse ning mida on võimalik kasutada platvormideülelt ja seadmesse laadida (Singh & Shobha, 2021).

Kasutajakogemus: (*UX, User Experience*) kirjeldab seda mida kasutaja vaimselt ja füüsiliselt tunneb, tajub ning mõtleb toote kasutamise ajal ja vahetult enne või pärast toote kasutamist (ISO Standards, 2023a).

Kasutajaliides: (*UI, User interface*) vahend kasutajale arvutiprogrammiga suhtluseks (Vallaste, s.a.).

Kasutatavus: (*Usability*) mil määral saavad kasutajad läbi toote kasutamise jõuda soovitud eesmärkideni efektiivselt ja tõhusalt, tundes sealjuures rahulolu konkreetsetes kasutuskontekstis (ISO Standards, 2023b).

Pistikprogramm: (*Plugin*) moodul, millega on võimalik laiendada süsteemi omadusi või teenuseid (Vallaste, s.a.).

PWA: (*Progressive Web App*) veebirakenduse tüüp millel on rohkem lisafunktsionaalsusi võrreldes tavalise veebirakendusega (Majchrzak et al., 2018).

Põlirakendus: (*Native app*) traditsiooniline nutiseadme rakendus mis laetakse seadmes alla ning mis omab platvormipõhist arhitektuuri ja koodibaasi (Kuitonen, 2019).

Raamistik: (*Framework*) abistav tarkvara, riistvara või muu struktuur loodavale platvormile (Vallaste, s.a.).

Rakenduse kest: (*Application shell*) Progressiivse veebirakenduse komponent mis koosneb töötamiseks vajalikust minimaalsest HTML-i, CSS-i ja Javascripti koodist (Hume, 2017).

Rakendusliides: (*API, Application Programming Interface*) kirjeldatud reeglistik mis koosneb erinevatest vahenditest, näiteks kuidas rakendusprogramm suhtleb operatsioonisüsteemi või teise rakendusprogrammi teenustega (Vallaste, s.a.).

Taustteenus: (*Service Worker*) Javascripti fail, millel on erinevaid omadusi. Näiteks vastutab ressursside vahemällu salvestamise ja teenindamise eest, võimaldades PWA rakendust kasutada ilma võrguühenduseta (Malavolta et al., 2017).

Teek: (*library*) Moodul või alamprogramm, mis on valmiskompileeritud ning mida programm saab kasutada (Vallaste, s.a.).

Tõuketeavitus: (*Push Notification*) Rakenduse poolt saadetak teavitus, mis ilmub näiteks nutitelefonil lukustatud ekraanile (Adobe, 2023).

Utiliit: (*Utility*) programm, mis on loodud täitma spetsiifilist ülesannet (Vallaste, s.a.).

Veebirakendus: (*Web Application*) lai termin, mille all mõeldakse veebis olevat tarkvara (Vallaste, s.a.).

SISSEJUHATUS

Diplomitöö teemaks on Haapsalu kolledži tunniplaani rakenduse nutikuva loomine. Praegune Haapsalu kolledži tunniplaani on loodud aastal 2015 kolledži tudengi poolt diplomitööna. Valikpraktika raames läbiviidud uurimuse tulemusena andsid tudengid ja õppejõud tagasisidet, et tunniplaani ei ole mugav kasutada, ikoonid ei ole arusaadavad ning info saamiseks on vaja teha palju liigutusi. Lisaks puudub tänasel lahendusel skaleeruv nutikuva. Nutiseadmes on tunniplaani kasutamine väga ebamugav. Samas selgus uuringu tulemusel, et peaaegu pooled (46%) uuringus osalejatest kasutavad tunniplaani vaatamiseks ka nutiseadet. Inimesed tahavad nutiseadmes kasutusmugavust ja sellele pööratakse üha enam tähelepanu (Cajander et al., 2022). Järjest rohkem kasutatakse igapäevaselt nutitelefone (Bouchrika, 2023) ning seetõttu on õigustatud ka ootus teha igapäevased toimingud mugavalt läbi nutitelefoni. Eeltoodule tuginedes saab väita, et tänane Haapsalu kolledži tunniplaani ei ole ajakohane ning vajab uuendamist.

Valikpraktika õppeaine raames on käesoleval õppeaastal loodud uue tunniplaani veebiversioon. Veebiversiooni loomisel uuendati ja lisati uusi funktsionaalsusi ning muudeti tunniplaani disaini. Uue tunniplaani loomise juures ei arvestatud eraldi nutiseadme kasutusmugavusega. Seega ei ole ka valikpraktika õppeaine raames loodud lahendust mugav nutiseadmes kasutada. Kuna Haapsalu kolledž on avaldanud soovi uus tunniplaani rakendus kasutusele võtta, siis arendatakse valikpraktika õppeaines loodut edasi kahe diplomitöö raames. Ahti Irsi diplomitöö eesmärgiks on arendada uue tunniplaani veebiversiooni kood tasemeni mis võimaldaks rakenduse reaalselt kolledžis kasutusele võtta. Antud diplomitöö raames täiendatakse loodud rakendust, lisades sellele nutikuva koos lisafunktsionaalsustega.

Diplomitöö eesmärk on selgitada välja Haapsalu kolledži tunniplaani nutiseadme kuva sobivaim tehnoloogiline lahendus ning luua kasutajasõbralik valmislahendus.

Lähtuvalt uurimistöö eesmärgist on sõnastatud kolm uurimisküsimust:

1. Millised on praegu kasutuses oleva tunniplaani kuva kitsaskohad?
2. Milline on sobivaim tehnoloogiline lahendus Haapsalu kolledži tunniplaani nutiseadme kuva tegemiseks?
3. Milline Haapsalu kolledži tunniplaani nutiseadme kuva on kasutaja jaoks mugav?

Uurimisküsimuste põhjal on sõnastatud uurimisülesanded:

1. Selgitada välja erinevad nutiseadme kuva tegemise tehnoloogilised lahendused.
2. Tuua välja praegu kasutuses oleva tunniplaani nutikuva kitsaskohad.
3. Luua uus nutiseadme rakenduse kuva.
4. Testida loodud rakenduse kuva kasutusmugavust ja funktsionaalsust kolledži tudengite, õppejõudude ja haldustöötajatega.

Töö koosneb neljast peatükist, sissejuhatusest ja kokkuvõttest. Esimeses peatükis antakse ülevaade valikpraktika õppeaines läbiviidud uuringust, tuuakse välja tunniplaani olulisemad nõuded ning kirjeldatakse valikpraktikas valminud lahendust. Teises peatükis tuuakse välja mobiilirakenduse arendamise võimalused, võrreldakse neid omavahel ning valitakse tunniplaani nutikuva loomiseks sobivaim. Kolmandas peatükis antakse ülevaade kasutatavatest lahendustest ja tehnoloogiast. Kirjeldatakse kasutusmugavuse erinevaid aspekte ning kasutatud CSS-raamistikku. Neljandas peatükis kirjeldatakse nutiseadme kuva rakenduse teostust: disaini ja kasutusmugavust, prototüübi loomist, arendamist, testimist ning tulemust ja edasiarenduse võimalusi. Kõik töös kasutatavad joonised ja koodnäited on autori enda loodud, kui ei ole märgitud teisiti.

1. HAAPSALU KOLLEDŽI TUNNIPLAANI NUTISEADME KUVA RAKENDUSE PLANEERIMINE

Peatükis antakse ülevaade valikpraktika õppeaine raames läbiviidud uuringust ning selle tulemustest. Lisaks tuuakse välja kolledži poolt esitatud nõuded, mille täitmine peab tunniplaani rakenduses olema tagatud.

1.1. Valikpraktika õppeaine raames läbiviidud uurimus ja selle tulemused

Valikpraktika õppeaine raames on läbi viidud uurimus Haapsalu kolledži tunniplaani kasutajate vajaduste paremaks mõistmiseks. Uurimus korraldati kahes osas. Kõigepealt viidi läbi laiem kvantitatiivne uuring, kasutades ankeetküsimustikke. Teiseks viidi läbi intervjuu kolledži administratiivtöötajaga.

1.1.1. Uurimismetoodika ja andmete kogumine

Valikpraktika õppeaine käigus viidi läbi uurimus, mille eesmärk oli saada ülevaade, kuidas ja millistes seadmetes täna tunniplaani kasutatakse, mis on peamised vajadused ning olulised funktsionaalsused. Uuringu läbiviimiseks kasutati elektrooniliselt levitatavat ankeetküsimustikku (Lisa 1). Tegemist on kvantitatiivse uurimismeetodiga. Samas olid ankeetküsimustikus ka mõned avatud vastustega küsimused, mis võimaldasid saada laialdasemat ja mitmekülgsemat infot. Küsimustikus oli kokku 20 küsimust, millest enamik olid valikvastustega. Küsimustikule vastas 30 Haapsalu kolledžis õppivat tudengit ning viis õppejõudu. Ankeetküsimustik oli Google Forms platvormil ning seda levitati läbi meililisti kõigile kolledži tudengitele ja õppejõududele. Küsimustik oli aktiivne perioodil 14.-21. september 2022.

Lisaks viidi läbi ka intervjuu Haapsalu kolledži haldustöötajaga, kelle vastutada on tunniplaani administreerimine. Uurimisviisiks valiti intervjuu, kuna tunniplaani administreerimine on kolledžis ühe inimese vastutada ning see sisaldab kõige keerulisemat osa rakenduse loogikast. Intervjuu eesmärk oli mõista haldustöötaja vajadusi ning tajutud kasutusmugavust. Tegemist oli poolstruktureeritud intervjuuga, mis kestis veidi üle tunni aja. Intervjuu viidi läbi kolledžis.

1.1.2. Tulemused ja analüüs

Uuringu tulemusel selgus, et pooled (46%) ankeetküsimustikule vastanutest kasutavad tunniplaani vaatamiseks nutiseadet, sealjuures 37% vaatab tunniplaani peamiselt nutitelefonist. Need tudengid, kes vaatasid tunniplaani peamiselt telefonist, olid oluliselt vähem rahul tunniplaani üldise kasutusmugavusega kui need, kes seda tegid peamiselt arvutist (kolme palli skaalal 1,6 vs 2,3). Lisaks oli erinevus ka tunniplaani arusaadavuses selle esimesel avamisel. Nende jaoks, kes vaatasid tunniplaani peamiselt arvutist, oli arusaadavus esmasel avamisel kolme palli skaalal 2,5 ning telefoni kasutavatel tudengitel oli arusaadavus oluliselt väiksem ehk 1,7. Kasutusmugavuse kitsaskohtadest mainiti kõige enam, et ikoonid ei ole üheselt arusaadavad ning kirja suurus on liiga väike.

Administratiivtöötaja intervjuerimisel selgus, et on esinenud olukordi kus nutiseadmest on olnud vaja teha tunniplaanis muudatust, kuid tegevus on intervjueeritava hinnangul olnud väga ebamugav. Samuti tõi administratiivtöötaja välja, et halduse vaates on väga palju visuaalset müra, mis teeb vajamineva informatsiooni leidmise keeruliseks.

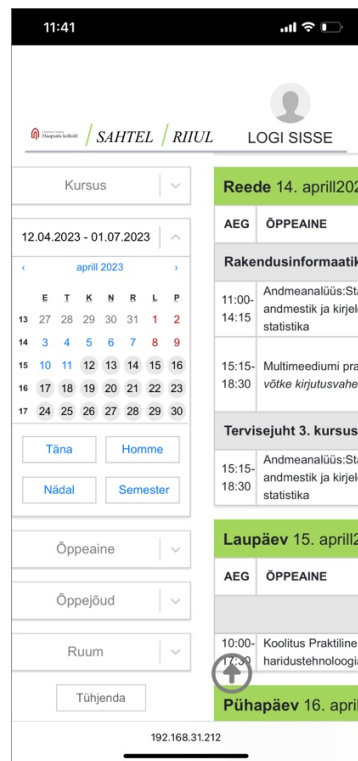
Uuringu tulemustele tuginedes saab öelda, et praegu kasutusel olev tunniplaan ei vasta nutiseadmes kasutajate vajadustele ja ootustele. Tunniplaani kasutamine nutiseadmes on ebamugav ning pole piisavalt arusaadav. Kuna suur hulk tudengeid kasutab tunniplaani vaatamiseks peamiselt nutiseadet, siis on vaja luua selleks kasutajale mugav võimalus.

1.2. Valikpraktika õppeaine raames loodud veebirakenduse prototüüp

Valikpraktika õppeaine raames selgitati koostöös Haapsalu kolledži töötajatega välja tunniplaani veebiversioonile olulisemad vajalikud nõuded, milleks olid:

- värvitoonide ja kirjastiilide vastavus Tallinna Ülikooli stiiliraamatuga;
- iseseisvate tööde jälgimise lisamise võimalus;
- filtrite säilitamine ka ilma sisse logimata;
- õppeinfo lahter peab toetama videoloengu lisamist ja vaatamist;
- õppeinfos peab nägema ka ainekaarti;
- tunniplaanis peab kasutajale kuvama lisatud iseseisvad tööd või videoloengu;

Tuginedes uuringu tulemustele ning kolledži nõuetele, loodi valikpraktika kursuse raames Haapsalu kolledži tunniplaani veebiversioon, milles ei ole arvesse võetud nutiseadmes mugavaks kasutamiseks vajalikke eripärasid (joonis 1).



Joonis 1. Valikpraktikas valminud tunniplaani kuva nutiseadmes

Diplomitöö tulemusena on kavas luua valikpraktika aine käigus tehtud veebirakendusele nutiseadme kuva lahendus. Eesmärk on säilitada kõik funktsionaalsused, kuid nende kuvamine ja lahendus võib kohati erineda, võttes arvesse nutiseadmes kasutamise eripärasid. Lisaks on planeeritud juurde luua ka selliseid funktsionaalsusi, mida veebiversioonis ei ole, kuid mis nutiseadmes rakendust kasutades annaks juurde kasutusmugavust.

2. MOBIILIRAKENDUSE ARENDAMISE VÕIMALUSED

Alates esimese iPhone'i turule jõudmist 2007. aasta alguses on mobiilirakendused muutunud järjest enam inimeste igapäevaelu osaks (Rakestraw et al., 2013). Esialgu loodi mobiilirakendused informatsiooni saamise ja produktiivsuse tõstmise eesmärgil, näiteks e-kirjad, ilmateade ja kalender, kuid viimase rohkema kui kümne aastaga on rakenduste eesmärgid ja kasutusviisid muutunud oluliselt mitmekesisemaks (Inukollu et al., 2014). Täna saab välja tuua kolm mobiilirakenduste kategooriat:

- põlirakendus;
- hübriidrakendus;
- veebirakendus.

Sobiva rakenduse kategooria valik sõltub rakenduse eesmärgist ning tingimustest millele see peab vastama (Litayem et al., 2015). Peatükis uuritakse erinevaid nutiseadme kuva tegemise tehnoloogilisi lahendusi ning leitakse nende seast sobivaim, mis rahuldab Haapsalu kolledži tunniplaani kasutajate vajadusi parimal viisil.

2.1. Põlirakendus

Põlismobiilirakenduse arendus viitab traditsioonile rakenduste arendamise viisile, kasutades platvormile omaseid programmeerimise keeli ja tarkvara arhitektuuri, et kirjutada koodibaasid, mis on omased vaid sellele platvormile. Erinevatel koodibaasidel on üksteisega väga vähe ühist ning ühe tundmine ei tee oluliselt kergemaks teisest aru saamist. (Kuitonen, 2019) Põlirakendused laaditakse telefoni alla vastavast rakenduste poest (Play Store/App Store) (Vilcek & Jakopc, 2017) ning seetõttu saavad nad ka nutiseadme võimalusi maksimaalselt ära kasutada (Litayem et al., 2015). Viimase kümne aasta jooksul on olnud erinevaid operatsioonisüsteeme, näiteks Android, Blackberry, iOS, Symbian ja Windows. Täna domineerivad turgu vaid kaks platvormi – iOS ja Android, millest populaarseim on Android (Statcounter, 2022). Isegi kui viimastel aastatel on kaks peamist platvormi olnud stabiilselt iOS ja Android, ei tähenda see, et nii ka jääb. Android on tõusnud kõige levinumaks operatsioonisüsteemiks viimase kümne aasta jooksul (Statcounter, 2022). Seega ei saa täna

kindlalt ennustada, mis toimub turul viie või kümne aasta pärast. Luues põlirakendust ainult ühes operatsioonisüsteemis, peab paraku arvestama mõningase ebakindlusega.

Põlirakenduse loomise viis on olnud kasutusel alates nutitelefonide loomise algusest (Rakestraw et al., 2013), kuid erinevad tööriistad ja tehnoloogiad on aastate jooksul arenenud. Androidile omane programmeerimise keel on Java või C++ (Google developers, s.a) ning alates 2017 on Androidi poolt toetatud ka Kotlin (Titus, 2018). Objective-C oli ainuke keel, mida kasutati iOS-i arenduses (Developer, 2023), kuni aastani 2014, mil Apple tutvustas enda programmeerimise keelt Swift (Developer, 2014). Põlirakenduse platvormipõhine lähenemine ei sunni arendajaid mitte ainult kasutama platvormile omaseid keeli, vaid ka soovitatud tarkvara arhitektuuri (Google developers, 2023). Androidi rakendused kasutavad tavaliselt Model-View-ViewModel (MVVM) arhitektuuri (Google developers, 2023) ning iOS rakendustes on soovituslik kasutada Model-View-Controller (MVC) arhitektuuri (Developer, s.a).

Erinevatel platvormidel on oma disaini juhised ning arenduse ja disaini juures tuleks jälgida rakenduse kasutajakogemust etteantud juhistele vastavalt. Kuna arendajad kasutavad platvormile omast keelt, saavad nad kasutada ka platvormi rakendusliidest ja platvormi poolt toetatud raamistikke. Lisaks on võimalik luua kasutajaliides, kasutades UI (user interface) komponente, mida pakub platvorm. (Developer, 2023) (Google developers, 2023) Operatsioonisüsteemi poolt toetatud ja pakutud UI, rakendusliidese ja raamistiku kasutamine on põlirakenduse suur eelis, sest see suurendab oluliselt kasutusmugavust. Rakenduse kasutajad ei pea õppima igat uut rakendust kasutama, vaid oskavad seda teha intuiitiivselt tänu varasematele kogemustele sarnaste rakendustega.

Potentsiaalsed rakenduse kasutajad jagunevad mitme platvormi vahel, mistõttu tuleb kõiki kaasata, kui soovitakse arendada erinevad rakendused eri platvormidele. Põlismobiilirakenduse samaaegne arendamine ja haldamine erinevatel platvormidel võib olla ajamahukas. Seejuures on Android platvormile rakenduse arendamine 30-40% aeganõudvam, kui iOS-i platvormile (Lamhaddab et al., 2019). Kuna rakendused peavad olema arendatud eri platvormide jaoks, kasutades erinevaid arendustehnoloogiaid, peavad arendajad arendama ja haldama ühe rakenduse jaoks mitmeid eraldiseisvaid koode (Kuitonen, 2019). Seega peavad arendajad haldama lisaks erinevatele keeltele ka erinevaid disainijuhiseid, raamistikke ja arhitektuuri. Lisaks ajakulule nõuab see ka oluliselt rohkem rahalisi vahendeid ning oskuseid.

Üks põlisrakenduse suurimaid eeliseid on see, et alla laetud rakendus võimaldab kasutada nutitelefonis riistvara, näiteks kaamerat (Litayem, 2015). Samuti saab alla laetud rakendust kasutada ilma internetiühenduseta, kui andmeid hoiustatakse otse seadmes. Veebirakendustel ei ole see võimekus veel piisavalt hea (Kuitonen, 2019). Lisaks töötleb allalaetav rakendus kiiremini infot ning on kasutaja jaoks ka kiirem kasutada (Litayem, 2015). See on oluline argument rakenduste puhul, mis töötlevad suures koguses keerulisi andmeid. Lihtsamate rakenduste puhul aga jääb eelis ebaselgeks. Üks puudus on ka rakenduste kasutamise järjepidevuse vähesus. 25% inimestest ei ava alla laetud rakendust enam kordagi peale esmast kasutamist ning ainult 32% inimestest kasutab rakendust rohkem kui 10 korda (Upland, 2023).

Kokkuvõtvalt saab öelda, et põlisrakenduse kasutamisel on erinevad positiivsed ja negatiivsed küljed, mis mõjutavad nii kasutusmugavust, koodi taaskasutust kui ka riist- ja tarkvarale ligipääsu. Ülevaade olulisematest tugevustest, puudustest ja võimalikest kasutusolukordadest on väljatoodud tabelis 1.

Tabel 1. Põlisrakenduse tugevused, puudused ja kasutus

Tugevused	Puudused	Millal kasutada
<ul style="list-style-type: none"> • suur kasutusmugavus; • maksimaalne ligipääs riist- ja tarkvarale; • saab kasutada ilma internetita; • kiire kasutada; • suur jõudlus. 	<ul style="list-style-type: none"> • kulukas ja ajamahukas; arendada ja hallata • peab alla laadima; • iga platvormi jaoks peab kasutama erinevaid programmeerimiskeeli ja disaini juhiseid; • kirjutatud koodi ei saa taaskasutada teisel platvormil. 	<ul style="list-style-type: none"> • on soov luua keeruline ja mahukas rakendus, mis kasutab seadme riist- ja tarkvara ning on olemas suurem ressurss rakenduse arendamiseks ja haldamiseks või soovitakse luua rakendus ainult ühele kindlale platvormile.

2.2. Hübriidrakendus

Hübriidrakenduse arendamine tagab platvormideülese lähenemise. Hübriidrakendused lubavad üht koodibaasi kasutada erinevatel platvormidel (Singh & Shobha, 2021). Platvormideülene rakenduste arendamine muutus levinumaks alles pärast seda kui arenduses hakati kasutama HTML5 märgendikeelt (Enihe & Joshua, 2020). Suurem osa hübriidrakenduste loomise raamistikest kasutavad veebipõhiseid HTML5, CSS-i ja JavaScripti keeli, mis pärast tõlgitakse selle operatsioonisüsteemi keelde, kus seda rakendust kasutatakse. Koodibaasis ei ole vaja pärast tõlkimist olulisi muudatusi teha. (Singh & Shobha, 2021) Hübriidraamistik tekitab veebirakenduse ümber konteinerkesta, mida on võimalik paigaldada alla laadimiseks erinevatele mobiiliplatvormidele (Kuitunen, 2019). Hübriidrakenduste kasutajaliidesed koosnevad enamasti veebikomponentidest, mis kasutavad pistikprogramme või teeke, et suhelda seadme süsteemispetsiifiliste funktsioonide, näiteks kaamera, geolokatsiooni ja muude riistvara sensoritega. Need teegid suhtlevad operatsioonisüsteemi rakendusliidestega, mis võivad olla platvormispetsiifilises keeles, nagu näiteks Java Androidi puhul. Hübriidrakendused näevad välja nagu tavalised rakendused ja neid laetakse alla platvormi rakendusepoodidest (App Store/ Play Store). (Singh & Shobha, 2021)

Hübriidrakenduste arendamiseks on erinevaid raamistikke, millest mõned on kogunud viimaste aastatega palju populaarsust ja kasutajaskonda. Peamised raamistikud Ionic, Flutter, .NET MAUI ja React Native erinevad üksteisest eri tegurite, näiteks programmeerimiskeele ja toetatud operatsioonisüsteemide poolest. (Singh & Shobha, 2021) React native on laialdaselt kasutatud raamistik mida toetavad veeb, Androidi ja iOS-i operatsioonisüsteemid. React Native on arendatud Facebooki poolt ning see kasutab rakenduse sees Javascripti peamise keelena ning veebipõhiseid keeli HTML-i ja CSS-i, et rakendust kirjeldada ja küljendada. (Kuitunen, 2019) React Native'i populaarsuse taga on võimekus matkida põlisrakenduste tunnetust läbi selle, et rakenduse siseselt kasutatakse veebikomponentide asemel platvormi põhiseid komponente. Ionic on loodud Drifty poolt ning kasutab veebipõhiseid keeli HTML ja CSS, et luua UI elemente. Ionic raamistik toetab veebi, Androidi ja iOS-i. Flutter on raamistik, mille on loonud Google ning see kasutab uut avatud lähetekoodiga programmeerimiskeelt Dart. Flutter toetab Androidi ja iOS-i operatsioonisüsteeme. .NET MAUI on Microsofti loodud platvormiülene raamistik, mis on avatud lähetekoodiga. .NET MAUI kasutab C#- ja XAML keel ning toetab

Androidi, iOS-i, macOS-i ja Windowsi platvorme. (Microsoft, 2023) React Native'i koodi taaskasutus on kuni 90%, Ionicul ja .NET MAUI-l 98% ning Flutter-i raamistikul 50-90% (Majchrzak et al., 2017).

Hübriidrakenduste kasutamise peamised eelised on aja ja raha kokkuhoid, kuna hübriidraamistikud on veebipõhised ja nende arendamisele kulub oluliselt vähem ressursi kui põlisrakenduste arendamiseks (Singh & Shobha, 2021). Hübriidrakendused pakuvad laial valikul UI- komponente ning kasutavad ära nii põlisrakenduse kui ka veebirakenduse tugevusi (Enihe & Joshua, 2020). Hübriidrakendusi toetavad eri platvormid ning seetõttu on nad kättesaadavamad, olenemata operatsioonisüsteemist (Singh & Shobha, 2021). Ei ole ka vaja hallata mitut erinevat rakendust. Samuti on koodi haldamine võrreldes põlisrakendustega lihtsam. Üks hübriidrakenduste populaarsuse põhjuseid on võimekus töötada ka ilma internetita. (Enihe & Joshua, 2020).

Hübriidrakendusel on samas ka olulisi puudusi. Rakendus on seotud ühe koodibaasiga, mistõttu kui leitakse viga või lisatakse uus võimekus, siis tuleb terve rakendus kõigil platvormidel uuesti läbi testida. Seda ka siis, kui uuendus puudutab ainult ühte platvormi. (Enihe & Joshua, 2020) Samuti võib kasutajaliides olla võrreldes põlisrakendusega piiratud funktsionaalsusega – hübriidrakenduse disaini puhul peab arvesse võtma kõigi platvormide eripärasid, sest muidu võib väheneda kasutusmugavus. Oleks küll võimalik luua eraldi kasutajaliidesed erinevatele platvormidele, kuid see vähendaks tugevalt hübriidrakenduse eelist põlisrakenduse ees (Kuitunen, 2019). Erinevad võimekused, näiteks 3D, võivad olla piiratud, kuna kasutatakse veebivaadet mis ei luba kasutada seadme täielikku võimekust. Puudusi on ka jõudlusega. Kogenud programmeerija on võimeline saavutama parema lõpptulemuse konkreetsel platvormil siis, kui ta arendab põlisrakendust, sest ristkompilaatorid ei suuda hübriidrakenduse puhul lihtsalt sama tulemust saavutada. (Enihe & Joshua, 2020) Ristkompilaator võimaldab ühel platvormil ehitatud rakendust tõlkida teise platvormi jaoks sobilikku formaati (Krishnan, 2021).

Hübriidrakenduse kasutamise positiivsed ja negatiivsed küljed on seotud näiteks koodi taaskasutusega, jõudluse ja kiirusega. Kokkuvõtte olulisematest tugevustest, puudustest ja võimalikest kasutusolukordadest on väljatoodud tabelis 2.

Tabel 2. Hübridrakenduse tugevused, puudused ja kasutus

Tugevused	Puudused	Millal kasutada
<ul style="list-style-type: none"> • keskmised arenduskulud; • kirjutatud koodi on võimalik taaskasutada teistel platvormidel; • töötab igal platvormil; • töötab ka ilma internetita; • võimalus kasutada ka ilma alla laadimata. 	<ul style="list-style-type: none"> • pigem keeruline hallata; • keskmine kasutusmugavus; • jõudlus ja kiirus on pigem väike; • ei ole saadaval kõik tarkvara ja riistvara võimekused. 	<ul style="list-style-type: none"> • on soov luua vähem keerukas rakendus mis töötab kõikidel platvormidel.

2.3. Veebirakendus

Veebirakendustele on hakatud rohkem tähelepanu pöörama peale esimese iPhone'i turule jõudmist, kuid varasemalt on veebipõhised rakendused põlisrakendustele oma funktsioonide ja kasutusmugavuse poolest alla jäänud (Steiner, 2018). Veebirakendused on ajas jõudsalt arenenud ning pärast seda, kui hakati kasutama progressiivset veebirakendust, sarnanevad veebirakendused oma funktsioonide ja kasutusmugavuse poolest üha rohkem põlisrakendustele (Majchrzak et al., 2018).

Veebirakendusi on eritüüpe, kuid selles peatükis on kirjeldatud lähemalt progressiivset veebirakendust (PWA). PWA on populaarsust kogunud alates 2015. aastast just tänu oma lisafunktsioonidele ja suurele kasutusmugavusele. Kuna sellel on skaleeruva veebirakenduse ees palju eeliseid, siis keskendutaksegi selles peatükis ainult progressiivsele veebirakendusele. Veebirakenduste arendamiseks kasutatakse keeli HTML, CSS ja JavaScript. (Majchrzak et al., 2018) PWA tugi on olemas näiteks raamistikel React, Angular, Vue, Preact ja PWABuilder (Borisa, 2022).

Frances Berriman ja Google Chrome'i arendaja Alex Russell löid termini progressiivne veebirakendus 2015. aastal. Nende sõnul kasutab PWA modernseid veebivõimekusi, et

pakkuda põlisrakendusele sarnast kasutajakogemust. (Khan et al., 2019) PWA on sisuliselt veebileht, millel on erinevad lisafunktsioonid. Progressiivne veebirakendus töötab eri platvormidel (iOS, Google Android jne) ja kõikidel veebilehitsejatel (Huber et al., 2021), seega on ta ligipääsetav erinevate seadmete kasutajatele. Lisaks tähendab see, et rakendust saab kirjutada ühe koodibaasiga. Seetõttu on PWA arendamise aeg ja kulud oluliselt väiksemad, kui näiteks põlisrakenduse puhul. PWA saab läbi veebilehitseja lisada töölauale, peale mida on rakendus kasutatav kui põlisrakendus. Rakenduse allalaadimine pole kohustuslik ning seda saab edukalt kasutada ka veebis. (Huber et al., 2021) PWA saab lisada nii App Store'i kui ka Google Play Store'i (Medium, 2019), tänu millele on kasutajatel laiem võimalus rakenduseni jõuda. PWA maht on oluliselt väiksem kui hübriid- ja põlisrakendusel ning ekraani kuva on skaleeruv (Huber et al., 2021). Kui on järgitud olulisi disaini- ja kasutusmugavuse printsiipe, on PWA kasutusmugavus ja -kogemus sarnane põlisrakendusele (Khan et al., 2019). Juhul kui rakendus on alla laetud, saab seda kasutada ka ilma internetiühendusega peamiselt tänu taustteenuse komponendile (Rego et al., 2019).

Taustteenus (*service worker*) on üks olulisemaid PWA komponente. See on JavaScripti fail, mis vastutab ressursside vahemällu salvestamise ja vahemällu salvestatud ressursside teenindamise eest, vähendades laadimisaega ja võimaldades rakendusel töötada ilma võrguühendusega. Samuti võimaldab taustteenus kasutada teavituste ja tõuketeavituste saatmise funktsioone. Lisaks vastutab taustteenus rakenduse uuenduste eest. (Malavolta et al., 2017) Veel üks oluline PWA komponent on *manifest*. See on JavaScripti põhine fail, mis võimaldab hoida rakenduse informatsiooni (nimi, ikoonid, kirjeldus) ja lubab laadida rakendust alla läbi veebilehitseja. Kolmas oluline PWA komponent on rakenduse kest, mis koosneb minimaalsest kasutajaliidese laadimiseks vajalikust HTML-i, CSS-i ja JavaScripti koodi hulgast. (Hume, 2017) Taustteenus saab rakenduse failid salvestada rakenduse kesta kui kasutaja rakendust esmakordselt kasutab. Edaspidistel kasutuskordadel laeb kasutajaliides koheselt vahemälust ja dünaamiline sisu laaditakse hiljem. (Rego et al., 2019) Tänu rakenduse kestile on PWA laadimiskiirus sama kui põlisrakenduse puhul.

Sõltuvalt operatsioonisüsteemist ja veebilehitsejast ei ole kõik tarkvara ja riistvara võimekused PWA-de puhul saadaval. Samuti ei tööta PWA-d hetkel veel vanades seadmetes, mis kasutavad vananenud veebilehitsejaid. (Dafda, 2022)

Sarnaselt põlis- ja hübriidrakendusele on ka PWA-l tugevused ja puudused, mis on seotud näiteks riist- ja tarkvaraga, kasutusmugavusega ning arenduskuludega. Ülevaade olulisematest tugevustest, puudustest ja võimalikest kasutusolukordadest on väljatoodud tabelis 3.

Tabel 3. Veebirakenduse tugevused, puudused ja kasutus

Tugevused	Puudused	Millal kasutada
<ul style="list-style-type: none"> • madalad arenduskulud; • rakendust on lihtsam luua ning uuendada; • töötab igal platvormil; • võimalus kasutada ka ilma alla laadimata; • võimalus saata teavitusi ja tõuketeavitusi; • suur kasutusmugavus; • töötab ilma internetita; • töötab kiiresti; • kasutab vähe mahtu. 	<ul style="list-style-type: none"> • ei ole saadaval kõik tarkvara ja riistvara võimekused; • ei tööta hästi vananenud veebilehitsejates. 	<ul style="list-style-type: none"> • on soov luua väheste ressurssidega kiire lahendus, millele pääsevad ligi erinevate platvormide kasutajad;

2.4. Järeldus

Tuginedes eelpool välja toodud lähenemiste plussidele ja miinustele valisin Haapsalu kolledži tunniplaani nutiseadme kuva lahenduse tarbeks progressiivse veebirakenduse. PWA kasuks räägivad kõige enam madalad haldus- ja arenduskulud. PWA arendamiseks piisab esirakenduse baaskeelte ja raamistike tundmisest ning pole vaja õppida lisaks põlisrakenduse või hübriidrakenduse loomiseks vajalikke raamistikke ja keeli. Seetõttu on PWA puhul Haapsalu kolledžil tunniplaani uuendamine ja haldamine edaspidi oluliselt lihtsam. Lisaks annab PWA kasutajale võimaluse otsustada, kas ta soovib rakenduse alla laadida või kasutada seda läbi veebilehitseja.

3. ÜLEVAADE KASUTATAVATEST LÄHENEMISTEST JA TEHNOLOOGIAST

Peatükis antakse ülevaade lähenemistest ja tehnoloogiast, mida diplomitöö teostamisel kasutatakse. Lähenemistest saab tuua välja kasutusmugavuse, kasutajakogemuse ja kasutatavuse. Tehnoloogiast kirjeldatakse lähemalt CSS-raamistikke ja põhjendatakse Tailwindi kasutamist.

3.1. Kasutusmugavus

Tänapäeva tehnoloogiakasutajad eeldavad seadmete kasutamisel kõrget rahulolu. Nad soovivad kasutada toodet ilma suurema pingutuseta ning täita oma eesmärgid kiiresti ja efektiivselt (Hinderks et al., 2022). Eduka digitaalse toote loomiseks ei piisa täna ainult küllaldaste funktsioonide pakkumisest, vaid arvestada tuleb ka erinevaid kvaliteediaspekte (Kashfi et al., 2019). Neist üks olulisemaid on üldine kasutusmugavus ja -kogemus, mille kasutaja toote kasutamisel saab (Buis et al., 2023). UX-il, UI-il ja kasutatavusel on ühine eesmärk: luua positiivne, selge ja tõhus interaktsioon toote või rakendusega.

3.1.1. Kasutajakogemus

Kasutusmugavus on muutunud järjest aktuaalsemaks tänu erinevate tehnoloogiate laialdasele kasutusevõtule ning sellele pööratakse täna rohkem tähelepanu kui 20 aastat tagasi (Silva et al., 2018). Kasutajakogemust (UX) on defineeritud järgnevalt: „kombineeritud kogemus sellest, mida kasutaja tunneb, tajub, mõtleb ning millele füüsiliselt ja vaimselt reageerib enne ja pärast toote kasutamist või vahetult selle ajal“ (ISO Standards, 2023a). Kasutusmugavuse komponendid on kasutatavus, kasulik ja soovitatav sisu, ligipääsetavus, usaldusväärsus, visuaalne meeldivus ja nauding. Kasutajakogemuse oluline kontseptsioon on protsess, mille käigus loovad kasutajad kogemusi alates toote esmakordsest kasutamisest ja hiljem terve kasutamise perioodi vältel. (Punchoojit & Hongwarittorn, 2017) Kasutajakogemust saab kirjeldada läbi kolme omaduse (Kim, 2015):

- holistiline lähenemine;
- kasutajakesksus;
- strateegiline väärtus.

Holistilise lähenemise all mõeldakse seda, et UX hõlmab erinevaid omadusi, mis ei hõlma ainult toote visuaalseid, kombatavaid ja auditoorseid aspekte, vaid ka seda, kuidas toode sobivas kasutuskeskonnas või -kontekstis toimib. Kasutajakesksuse all peetakse silmas seda, et kasutusmugavust hinnatakse läbi selle, kuidas kasutaja mõtleb, tunneb ja käitub. UX-i strateegiline väärtus on oluline ettevõtte toote või teenuse arendamisel. (Kim, 2015) Kasutajakogemusest lähtumine aitab disainida tooted ja teenused mis vastavad sihtgrupi vajadustele ja soovidele. Kasutajakogemuse mõõtmisel kasutatakse pigem kvalitatiivseid mõõdikuid ning paljud UX-i komponendid on mõõdetavad vaid subjektiivselt (Kim, 2015). Näiteks saab uurida kasutaja emotsioone toote kasutamisel. Kasutajakogemuse disainimise eesmärk ongi tekitada toote suhtes positiivsed tunded (rahuldust pakkuv, põnev, motiveeriv) ning minimeerida negatiivseid tundeid (igavus, tüdimus) (Punchoojit & Hongwarittorn, 2017).

Suure kasutusmugavuse positiivne tulemus on see, et toode sobib rohkem konkreetsele sihtrühmale, kellele see on mõeldud (Krueger et al., 2020). Samuti on kasutajad rohkem motiveeritud toodet tulevikus kasutama (Hassenzahl, 2010). Eelpool väljatoodu aitab kaasa kasutajate suuremale rahulolule (Alves et al., 2014).

3.1.2. Kasutatavus

Kasutatavus (*usability*) on defineeritud järgnevalt: „mil määral saavad kindlaksmääratud kasutajad toodet kasutada, et saavutada soovitud eesmärgid efektiivselt ja tõhusalt, luues rahulolu konkreetsetes kasutuskontekstis“ (ISO Standards, 2023b). ISO standardite (2023b) järgi on mõeldud efektiivsuse all saavutatud eesmärkide ootustele vastavust ning tõhususe all erinevate ressursside (aeg, energia) optimaalset kasutust.

Mobiiliseadmete ja arvutite kasutamise ning seadmetelt info omastamise vahel on suured erinevused. Mõned märkimisväärsed erinevused on näiteks erinev ekraani suurus, erinevus interaktsioonis ning visuaalse tähelepanu kasutamises. Kui arvutit kasutatakse pigem statsionaarselt, siis nutitelefoni ka kõndimise, jooksmise ning autoga sõitmise ajal. (Punchoojit & Hongwarittorn, 2017) Nendest erinevustest peab lähtuma ka disaini funktsionaalsus ning kasutatavus peab arvestama seadme eripäraga.

Kasutatavus on oluline iga toote või teenuse puhul, sest kui kasutajad ei saavuta oma eesmärke tõhusalt, kiiresti ja rahuldust pakuval viisil, siis nad hakkavad otsima alternatiivseid võimalusi enda eesmärkide täitmiseks (Punchoojit & Hongwarittorn, 2017). Tootel, mis on hästi kasutatav, peab olema täidetud kolm tingimust (Punchoojit & Hongwarittorn, 2017): (1) toode on kasutajale lihtsasti arusaadav ka esimesel kasutuskorral; (2) kasutajal on võimalik täita enda eesmärgid lihtsalt ning (3) toote kasutajaliidest ja selle kasutamist on hilisematel kasutustel lihtne meelde tuletada. Vastupidiselt UX-i eesmärkidele on need tulemused enamasti objektiivselt ja kvantitatiivselt mõõdetavad.

Kuigi kasutatavus ja UX on erinevad, pole nad üksteisest eraldatud. Kasutatavus on osa kasutajakogemusest. Näiteks kui toode on visuaalselt rahuldust pakkuv, võib see luua positiivse kasutajakogemuse. Või kui kasutatavus on madalal tasemel, siis võib see mõjutada üldist kasutajakogemust.

3.1.3. Kasutajaliidese disain

Kasutajaliides on üks olulisematest mobiilirakenduse komponentidest ning mõjutab otseselt seda, kuidas kasutajad rakendust tajuvad (Nguyen et al., 2018). Kasutajaliides on otsene suhtluspunkt kasutaja ja rakenduse vahel (Vogt & Meier, 2010). Seetõttu on vaja kujundada selline kasutajaliides, mis on kohandatud soovitud sihtrühma nõudmistele ja vajadustele (Boll & Brune, 2015). Kasutajaliidese aluseks on ekraani kujundus, kuid kasutajaliidese disain hõlmab ka seda, mida kasutaja tajub ning millega toodet kasutades füüsiliselt ja kontseptuaalselt kokku puutub (Benyon, 2019). Ehk disaini osa on kõik see, millele saab ekraanil vajutada, tajutud värvid ja helid ning kontseptuaalne loogika rakenduse või toote kasutamise taga. Tõhusa kasutajaliidese disain peab lähtuma erinevates distsipliinidest ja valdkondadest. Näiteks kasutaja füüsilisest ja kognitiivsest võimekusest, sotsioloogilisest kontekstist, arvutiteadusest ja informaatikast ning graafilisest disainist (Bennett et al. 2012).

UI oluline vahe UX-ist on see, et kui UX keskendub rohkem kasutaja üldisele kogemusele, siis UI keskendub rohkem toote väljanägemisele. UI hindamiseks kasutatakse kvantitatiivseid mõõtmisviise või kasutatavuse testimist. UI keskendub ka oluliselt vähem kasutaja tunnetele ja üldisele subjektiivsele kogemusele (Kim, 2015). Võrreldes kasutajakogemuse ja kasutatavusega, on UI disain kõige kitsam vaade kasutusmugavusele.

3.2. Kasutatav tehnoloogia

CSS-raamistikest on kõige kasutatavam Bootstrap, kuid Tailwind-i kasutajate arv on alates loomise ajast hüppeliselt tõusnud ning tänaseks on see populaarsuselt teine CSS-raamistik (Silkals, 2023). Tailwindi lõi 2017. aastal Adam Wathan ja Steve Schoger (Enes, 2022). See on algajasõbralik esirakenduse raamistik, mis kasutab CSS-keelt ja utiliitklasse, mis võimaldavad luua disaine väga paindlikult (Gerchev, 2022).

Utiliitklasside kasutamine on Tailwindi kõige suurem eelis Bootstrapi raamistiku ees, mis kasutab komponente. Kui komponentklassis on juba komplekteeritud erinev hulk väärtustatud CSS-parameetreid, siis utiliitklass sisaldab ühte väärtustatud CSS-parameetrit, mida vastavalt vajadusele kasutada. Komponendid võimaldavad teha veebiarendust kiiresti ja lihtsalt, kuid disaini loomine ei ole nii paindlik ning võimalused on rohkem piiratud. Utiliitklassid töötavad madalamal tasemel, mis annab nende kasutamisel suurema kontrolli ja paindlikkuse. Utiliitklasse on kerge kohandada, seega saab neid kasutades luua piiramatul võimalusel stiiliklasse. (Gerchev, 2022)

Erinevalt Bootstrapist ei paku Tailwind eelnevalt valmis ehitatud komponente, vaid võimaldab kasutada tööriistu ja CSS-klasse, et ehitada disainelemendid kiiresti ja paindlikult. Samuti jälgib Tailwind, mida kodeerimisel kirjutati, ning kustutab kõik selle, mida ei kasutatud. Seega erinevalt Bootstrapist ei ole projektis kasutamata CSS-klasse, mis vähendavad jõudlust. (Enes, 2022)

Tunniplaani arenduses on kasutatud Tailwindi raamistikku peamiselt seetõttu, et soovitakse suuremat vabadust disaini loomisel. Lisaks sellepärast, et arendatakse edasi valikpraktika raames tehtud tööd, mistõttu on oluline jälgida sama disaini struktuuri ka mobiili vaadetes. Seda võimaldab Tailwind teha hõlpsamini. Peale selle on Tailwindi suur tugevus ka ülearuse koodi kustutamise tõttu vabanev jõudlus.

4. HAAPSALU KOLLEDŽI TUNNIPLAANI NUTISEADME KUVA RAKENDUSE TEOSTUS

Peatükis antakse ülevaade Haapsalu kolledži tunniplaani nutiseadme kuva rakenduse teostusest. Kirjeldatakse disaini ja kasutusmugavust, prototüübi loomist, arendust ja testimist. Viimasena kirjeldatakse valminud rakendust ning selle potentsiaalseid edasiarenduse võimalusi.

4.1. Nutiseadme kuva disain ja kasutusmugavus

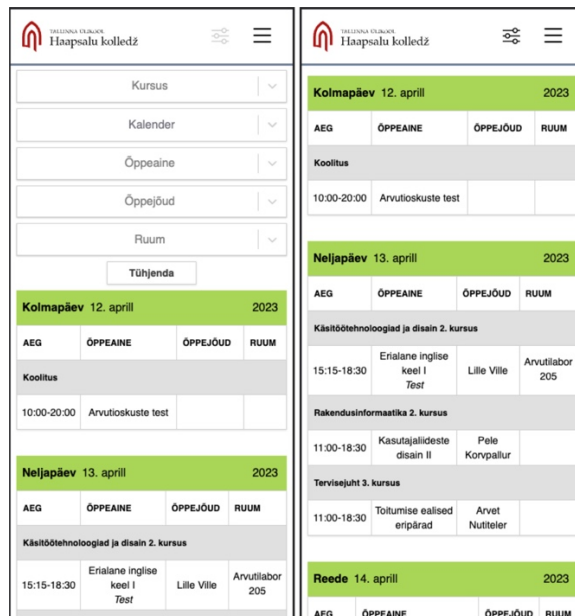
Nutiseadme kuva disaini osas lähtuti valikpraktika käigus valminud veebiversiooni disainist, mis on kooskõlas Tallinna Ülikooli stiiliraamatuga (Velvet, 2017). Stiiliraamatust lähtuvalt on valitud valikpraktikas loodud veebiversiooni värvid ja font. Samuti on jälgitud üldiseid veebidisaini parimaid praktikaid ning kasutusmugavust. Näiteks on jäetud elementide vahele piisavalt vaba ruumi, et ei tekiks visuaalset müra. Järgnevalt on kirjeldatud erinevaid disaini ja kasutusmugavus aspekte, mis on loodud nutiseadme kuva jaoks.

Kasutaja täpsuse parandamiseks on ikoonid suuremad kui veebiversioonis ja täidetavad sisestusvormid asuvad üksteise all (joonis 2).

The screenshot shows a mobile application interface for adding a lesson to the timetable. At the top, there is a header with the Haapsalu College logo and name. Below the header, there is a title bar for the 'LOENGU LISAMINE TUNNIPLAANI' (Adding Lesson to Timetable) screen. The main content area contains several dropdown menus for selecting 'Õppeaine' (Subject), 'Õppejõud' (Instructor), 'Kursus' (Course), and 'Ruum' (Room). Below these are input fields for 'Kuupäev' (Date), 'Maht' (Volume), 'Algus' (Start time, currently set to 10:00), 'Lõpp' (End time), and 'Lõuna' (Lunch, currently set to 'Ei' - No). A plus sign icon is visible at the bottom of the form.

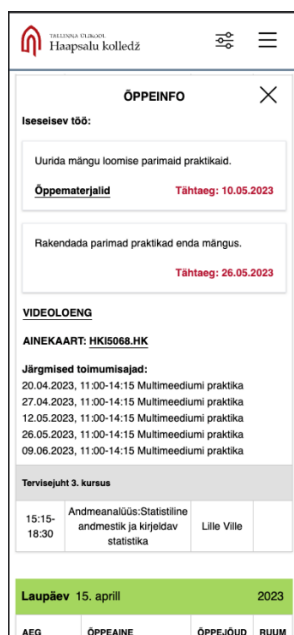
Joonis 2. Haldustöötaja vaade loengu lisamisel

Visuaalse müra vähendamiseks on tunniplaani filtrid vaikeväärtusena suletud ja avatavad päises olevast ikoonist (joonis 3).



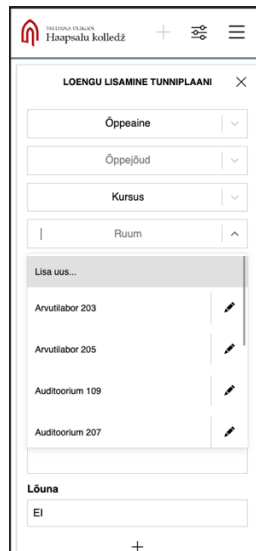
Joonis 3. Tunniplaani filtrid nutiseadmes avatuna ja suletuna

Kuna õppeinfos võib olla mitu sisestatavat tööd, siis on õppeinfo paremaks eristamiseks kasutatud kastide abi liigendamist (joonis 4).



Joonis 4. Tunniplaani õppeinfo vaade

Halduse vaates on rippmenüü valikute muutmine ja kustutamine ühe liigutuse võrra kiirem. Rippmenüü iga valiku juures on pliiatsi ikoon, mille kaudu saab kirjet muuta. Lisaks on nutiseadmes valikutevaheline reavahe suurem, et suurendada kasutaja täpsust (joonis 5).



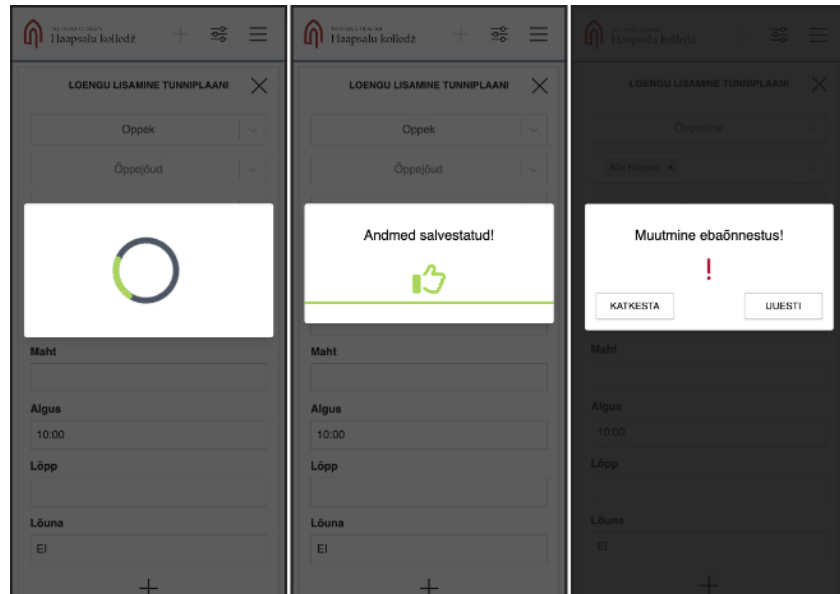
Joonis 5. Rippmenüü tunniplaani kirje lisamisel

Päises on kasutatud nuppude asemel universaalseid kasutajale kergesti äratuntavaid ikoone. Navigeerimine on lahendatud hamburgerimenüüga, millele vajutades avaneb külje pealt sisenev menüü (joonis 6).



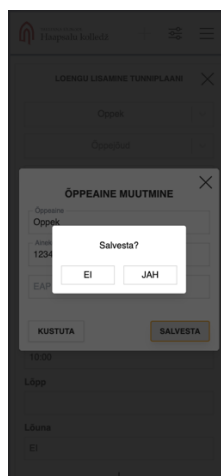
Joonis 6. Külje pealt avanev hamburgermenüü

Kasutusmugavuse seisukohast on üks oluline aspekt kasutaja tagasiside lehel tehtavate toimingute kohta. Juhul kui salvestamisel on kohustuslikud andmed puudulikud, antakse sellest kasutajale teada läbi kohustusliku välja punaselt kuvamise. Kasutajat teavitatakse rakendusliidese toimingu laadimisest, õnnestumisest ja ebaõnnestumisest (joonis 7).



Joonis 7. Kasutaja teavitused rakendusliidese suhtluses

Võimalike kasutajapoolsete vigade vältimiseks kuvatakse hüpikaken, kus küsitakse kriitilisele toimingule kinnitust. Näiteks kas kasutaja soovib tunniplaani kirje lisamist, muutmist ja kustutamist (joonis 8).



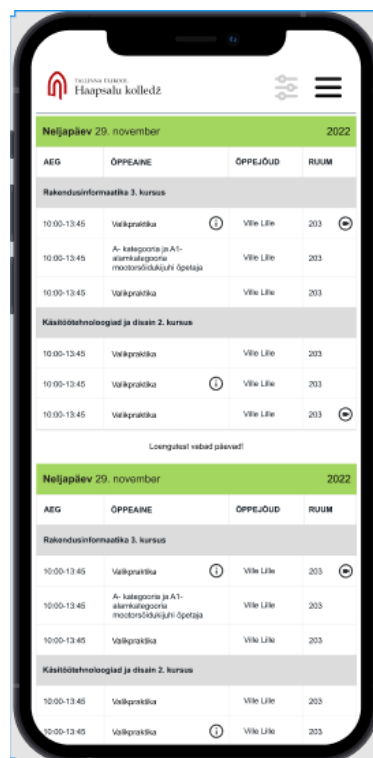
Joonis 8. Kinnituse modaalaken õppeaine muutmise vaates

4.2. Prototüüp

Prototüübi eesmärk on anda kasutajale võimalus lihtsustatud mudelit interaktiivselt kasutada, et saada tagasisidet lahenduse sobivuse kohta. Prototüüp võib sisaldada osalist või täielikku funktsionaalsust, eksisteerida paberil või digitaalselt. (Tammets, s.a.)

Prototüüp valmis Figma veebikeskkonnas. Figma on veebipõhine disaini tarkvara, mis sisaldab mitmed erinevaid funktsionaalsusi. Näiteks on võimalik luua veebirakenduse disain, muuta see interaktiivseks ja jagada seda kliendile või testida kasutajate peal.

Figma pakub ka erinevaid laiendusi, mida on võimalik prototüübi loomisel kasutada. Näiteks kasutati nutitelefoni imiteerivat kesta, et tajuda võimalikult hästi nutitelefoni ekraani (joonis 9). Lisaks kasutati tarkvara omadust muuta disaini vaated interaktiivseks. Näiteks kui kasutaja vajutab menüü ikoonile, kuvatakse talle vastav vaade. See omadus võimaldab luua kasutajale võimalikult reaalse prototüübi testkeskkonna. Prototüübi valmides testiti tulemust kasutajatega ja viidi lähtuvalt tagasisidest sisse muudatused.



Joonis 9. Figma laienduse poolt pakutav nutitelefoni kesta

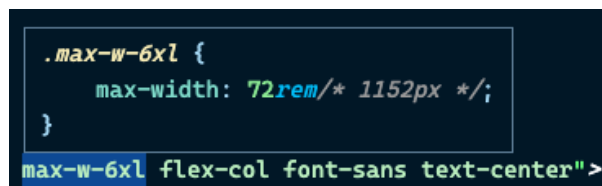
4.3. Arendus

Arendusega alustati pärast seda, kui prototüübi testimisesl vajalikuks osutunud parandused olid sisse viidud. Arendatud koodi hoiustamiseks kasutati Haapsalu kolledži *GitHub*-i koodihalduskeskkonda, tehes seal olevast tunniplaani koodist eraldi „diplomit66“ nimelise haru. Sellest omakorda tekitati vastavalt lahendatavale probleemile eraldi harud, näiteks Tailwindi implementeerimiseks.

4.3.1. Tailwind CSS-i lisamine

Esimene etapp hõlmas Tailwindi raamistiku kasutuselevõttu. Raamistiku kasutamine kulges ilma probleemideta, sest raamistiku ametliku dokumentatsiooni ülesehitus ja info otsimine oli arusaadav ja loogiline. Oluline eeldus raamistiku mugavaks kasutuseks oli hea CSSi tundmine.

Alguses prooviti valikpraktika käigus kirjutatud CSS-i asendada järk-järgult Tailwindi klassidega, kuid see põhjustas erinevaid probleeme. Näiteks kirjutas alamelemendis olev CSS raamistiku klassi väärtuse üle. Seetõttu eemaldati varem lisatud CSS ja alustati algusest. Raamistiku kasutusele lisasid mugavust koodiredaktorile lisatavad Tailwindi laiendused (joonis 10).



```
.max-w-6xl {  
  max-width: 72rem/* 1152px */;  
}  
  
max-w-6xl flex-col font-sans text-center">
```

Joonis 10. Koodiredaktori laienduse, mis selgitab klassi sisu

Tailwind pakub küll vaikimisi väga laia valikut erinevaid ära kirjeldatud stiile, kuid siiski võib tulla ette olukordi, kus on vajalik seda nimistut täiendada või muuta. Selleks on raamistikul konfiguratsioonifail, kus on võimalik muuta olemasolevaid või lisada endale vajalikke uusi stiile (koodinäide 1). Konfiguratsioonifaili omaduste kirjeldamiseks saab tuua järgnevad näited:

- Üheks väljakutseks osutus tunniplaani kirjade kuvamine väiksematel ekraanidel. Selleks tuli lisada raamistiku vaikeväärtusele ekraani suuruseid, mille rakendumisel kohandub paigutus vastavalt.

- Võimalus lisada rakenduses kasutavad värvid millele saab anda nimetuse ning läbi selle need projektis rakendada.
- Animatsioonide lisamine käib samuti konfiguratsiooni faili kaudu.

```
module.exports = {  
  theme: {  
    extend: {  
      colors: {  
        collegeGreen: "#a4d65e",  
      }  
    }  
  }  
};
```

Koodinäide 1. Rakenduses kasutatava värvi lisamine Tailwindi konfiguratsiooni faili

Rakendus sisaldas ka palju disainiomaduste poolest identseid elemente. Selleks, et mitte dubleerida stiiliomadusi mitme elemendi juures, pakub Tailwind võimalust kirjeldada stiile ühes globaalses klassis ning seda teiste samasuguste elementide küljes rakendada. Selleks loodi rakenduseülene stiilifail ning imporditi Tailwindi raamistik. Seejärel loodi stiilikogumid ja lisati need elemendi identsete elementide külge (koodinäide 2).

```
@tailwind base;  
@tailwind components;  
@tailwind utilities;  
  
.subject-info-tr {  
  @apply border-x border-borderGray text-left text-sm font-bold;  
}
```

Koodinäide 2. Uue stiilikomponendi lisamine Tailwindis

4.3.2. Progressiivse veebirakenduse loomine

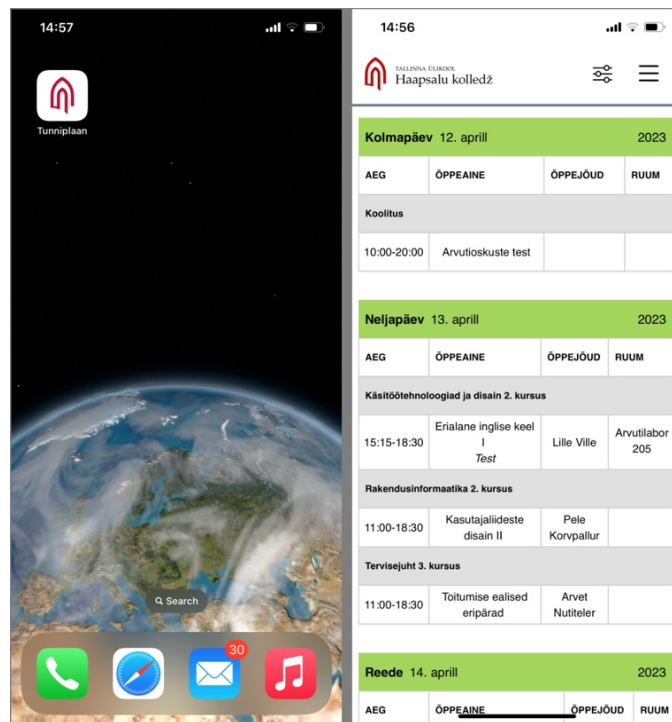
Viimane etapp koodis sisaldas progressiivse veebirakenduse (PWA) loomist. Eesrakenduses on kasutatud Reacti raamistikku, mis pakub ise juba võimekust luua suur osa PWA jaoks vajaminevatest konfiguratsioonifailidest.

Reacti raamistiku aluspõhja luues tuleb otsustada, kas loodav rakendus on PWA võimekusega või mitte. Kuna valikpraktikas loodud aluspõhi ei sisaldanud endas PWA võimekust, oli vajalik genereerida PWA toega Reacti projekt. Uuest projektist tuli olemasolevasse projekti kopeerida vajaminevad failid:

- `service-worker.js` – sisaldab näiteks veebibrauseri vahemällu salvestamise loogikat. Samuti on võimalik seal kirjeldada ka muid funktsionaalsusi, näiteks tõuketeavitusi.
- `serviceWorkerRegistration.js` – sisaldab loogikat, milles antakse veebibrauserile teada taustteenuse olemasolust. Oluline on registreerimis meetod ka `index.js` failis välja kutsuda.
- `manifest.json` – sisaldab rakenduse informatsiooni: nimi, kirjeldus ja ikoonid. Fail tuleb siduda ka `index.html` failiga.

Peale kopeerimist võis uue genereeritud Reacti projekti ära kustutada. Kui taustteenuse genereeritav loogika on juba vaikimisi paigas, siis rakenduse kirjeldust sisaldav `manifest.json` tuli eraldi rakenduse informatsiooniga ära kirjeldada (Lisa 2).

Ühe olulise uue teadmisenäna saab välja tuua, et React-raamistikku kasutades ei tööta PWA funktsionaalsus arendusfaasis, vaid rakendus tuleb kompileerida ja serveris tööle panna. Alles seejärel on võimalik kasutada PWA funktsionaalsust (joonis 11).



Joonis 11. Tunniplaani rakendus nutiseadme ekraanil ja avatuna

4.4. Testimine

Testimine oli arendusprotsessi oluline ja korduv osa. Testimist alustati esmasest prototüübist ja lõpetati valminud rakendusega. Kasutusmugavuse hindamise seisukohalt on testimine oluline tööriist, mille abil on võimalik tuvastada, kas kasutaja tunneb ennast rakendust kasutades hästi ning kas kasutus on lihtne ja loogiline (Tammets, s.a.).

Testijate leidmiseks pöörduti otse sihtgruppi kuuluvate inimeste poole. Peamiselt toimus testimine videosilla vahendusel. Ainult haldustöötajaga viidi testimine läbi kolledžis kohapeal. Prototüübi ja rakenduse testimisel tuli osalejatel jagada oma seadmest ekraani kuva, mida osaleja nõusolekul salvestati hilisema analüüsi jaoks. Veebis toimunud prototüübi testimisel osalesid kasutajad läbi nutiseadme. Rakendust testiti erisuguste nutiseadmetega. Kasutati nii erinevaid nutitelefonide mudelid kui ka kahe erineva tootja tahvelarvutid. Testimisel osalenute arv sõltus kasutajate tagasisidest. Kui uut infot enam ei lisandunud ja tagasiside hakkas korduma, otsustati testimisega lõpetada.

Testimise protsess koosnes erinevatest etappidest:

- Testimise protsessi tutvustus. Selgitati mida testitakse ning kuidas protsessis osaleda (näiteks mõelda valjult). Salvestamise jaoks küsiti luba.
- Esimene osa, milleks oli viie sekundi test. Kasutajal lasti vaadata viis sekundit rakenduse avalehte ja seejärel küsiti arusaadavuse seisukohalt erinevaid küsimusi.
- Teine osa testist, kus kasutaja võis vabalt kahe-kolme minuti jooksul rakenduses ringi vaadata.
- Kolmandas osas mängiti vastavalt kasutaja rollile läbi talle suunatud ülesandeid (Lisa 3). Ülesanded on tuletatud valikpraktikas valminud kasutajalugudest. Näiteks tuli kasutajal filtreid kasutades leida kõik tema kursuse ained.
- Viimane osa ehk kokkuvõte. Kasutaja sai küsida küsimusi või jagada mõtteid mis rakenduse juures veel meeldis või mis segadust tekitas.

Figmas loodud klikitava nutikuva prototüüp oli esimene, mida kasutajatega testiti. Testimises osales neli inimest, kellest üks oli kolledži tudeng. Testimise jooksul tuli esile mitmeid puuduseid. Üks korduvalt väljatoodud probleem puudutas rippmenüüde ja tekstisisestus kastide horisontaalset paiknemist. Enamus kasutajatest oli harjunud, et nutiseadmes paiknevad need vertikaalselt üksteise all. Samuti ei mahtunud kahe kõrvuti asetseva mitme valikuga rippmenüü valitud väärtused ekraanile ära. Tekkis olukord, kus üks kast hakkas allapoole suuremaks kasvama, mida üks osaleja hindas teda ärritavaks (joonis 12).

Joonis 12. Prototüübi vaade filtritest

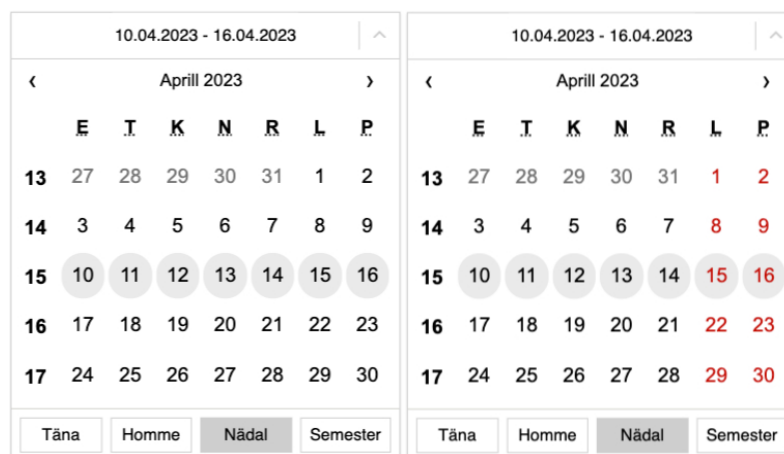
Teine testimisel osalenud inimeste poolt välja toodud puudus oli seotud õppeinfo ploki ja iseseisvate tööde vaatamisega. Osalejate jaoks polnud iseseisvad tööd piisavalt hästi üksteisest ja muust infost eristatavad (joonis 13).

Joonis 13. Prototüübi vaade õppeinfost

Testimisel toodi välja teisigi puudusi. Samas juhtisid testimisel osalenud inimesed tähelepanu ka tugevustele, nagu näiteks minimalistlik disain. Prototüübi testimisel väljatoodud puudused parandati enne disaini lisamist rakendusse.


Pärast prototüübi implementeerimist rakendusse oli võimalik alustada kasutusmugavuse testimist juba täisfunktsionaalses rakenduses. Rakenduse testimisel osales kaheksa kolledžiga seotud inimest: viis tudengit, kaks õppejõudu ja haldustöötaja. Samuti osales kaks kolledžiga mitteseotud inimest. Tänu täisfunktsionaalsusele saadi ka põhjalikumalt tagasisidet, mida kasutusmugavuse seisukohast parandada. Peamised probleemid puudutasid tunniplaani filtreid ning õppeinfo kuvamist ja lisamist.

Näiteks tõi tunniplaani filtrite puhul enamik testimisel osalenud inimesi välja, et nutikuval võiks kalender vaikimis kinni olla ning valitud vahemiku puhul nädalavahetuse päevad punased (joonis 14).




Joonis 14. Kalender enne testimist ja pärast muudatust.


Õppeinfo ploki kohta märkisid osalised näiteks, et iseseisvate tööde tähtaeg võiks rohkem välja paista. Mitu kasutaja soovis, et see oleks märgitud ära punasega. Õppeinfo lisamise vaates tõi õppejõud välja, et järgmise töö lisamise ikoon võiks olla kastist väljaspool. Lisaks tekitas segadust salvestamise ikoon. Üleval paremas servas asetseva salvestamise ikooni asemel sooviti pigem näha nuppu, mis võiks asetseda vaate allosas (joonis 15).




TALLINNA ÜLKOOL
Haapsalu kolledž

+





ÕPPEINFO MUUTMINE



Kommentaar:

Test

Iseseisev töö:


Iseseisva töö kirjeldus


Kirjuta suurtest silmadest esse

Tähtaeg:

15.04.2023

Õppematerjalide link





Videolengu link:

<https://Neti.ee>

SALVESTA

Rakendusinformaatika 2. kursus

Joonis 15. Õppeinfo muutmine ja lisamine pärast testimisest tulnud muudatusi

Halduse vaates toodi suurima puudusena välja lisamisvormi mitteautomaatselt sulgemine peale salvestamist. Samas meeldisid väga kasutajateavitused ja rikke või häire korral uuesti salvestamise võimalus.

Testimise tulemustele vastavalt said sisse viidud muudatused, mis leidsid mainimist mitme osaleja poolt. Ülejäänud tagasiside seisnes pigem kasutajate individuaalsetes eelistuses, näiteks värvitooniga seotud ettepanekud. Palju anti ka positiivset tagasisidet. Näiteks meeldisid tunniplaani minimalistlikus, kasutajateavitused ja värvide valik. Osalised pakkusid välja ka erinevaid ideid, mida rakenduse funktsionaalsuses võiks muuta või lisada. Näiteks õppeinfosüsteem ÕIS võiks olla seotud kolledži tunniplaaniga. Kuna paljud ettepanekud ei mahtunud diplomitöö fookusesse, siis lisati need kolledži GitHubi keskkonda loendina, et kolledžile laekunud tagasisidest ülevaadet luua.

4.5. Tulemus ja võimalik edasiarendus

Diplomitöö aluseks oleva arendustöö tulemusena kasutab kolledži uus tunniplaanimrakendus Tailwind CSS-i ja PWA omadusi, olles seejuures skaleeruv kõikides seadmetes ning pakkudes võimalust lisada rakendus nutiseadme ekraanile ja kasutada seda sarnaselt põlisrakendusele.

Tunniplaani rakendusele loodud nutikuva võimaldab pärast kolledžis kasutuselevõttu vaadata kasutajatel tunniplaani mugavalt lisaks arvutile ka erinevates nutiseadmetes. Tunniplaani haldust on nüüd samuti võimalik teha mugavalt otse nutiseadmest. Pärast Tailwindi kasutuselevõttu paranesid ka laua- ja sülearvuti vaates mitmed omadused. Näiteks kasutajat abistavad teavitused ning ekraani kuva skaleerumine.

Haapsalu kolledž on avaldanud soovi tunniplaani kasutusele võtta. PWA lahendus võimaldab tulevikus kolledži tunniplaanile lisada veel erinevaid funktsionaalsusi. Näiteks on võimalik saata kasutajale tõuketeavitusi, kui tunniplaani lisandub uus kirje või on lisatud iseseisvaid töid. Tulevikuarenduste jaoks saab taaskasutada diplomitöö käigus loodud erinevaid komponente. Edasiarendusena on võimalik muuta rakendus nutiseadme kirjasuurusele kohanduvaks. Näiteks et inimene, kes kasutab nutiseadmest suuremat kirjasuurust, saaks ka rakendust automaatselt suurema kirjaga kasutada.

Diplomitöö käigus loodud prototüüp, rakendus ja rakenduse kood on saadaval järgmistel aadressidel:

- Prototüüp: <http://shorturl.at/dloRV>
- Rakendus: <http://tunniplaani.hk.tlu.ee/>
- Rakenduse kood: <https://github.com/tluhk/HK-Tunniplaani/tree/diplomit66>

KOKKUVÕTE

Haapsalu kolledžil puudus seni kasutajasõbralik nutiseadme kuva. Praegu on tunniplaani kasutamine nutiseadmest ebamugav ning aeganõudev. Valikpraktika õppeaines läbiviidud uuringu tulemuste põhjal on tunniplaani kasutusmugavus madal ning tunniplaani kasutamine tekitab segadust. Inimestel on ootus kasutada erinevaid rakendusi mugavalt ka nutiseadmest. Nutikuva loomise vajadus tuli välja ka valikpraktika õppeaine käigus läbiviidud uuringu tulemustest.

Mobiilirakendust on võimalik luua põlisrakendusena, veebirakendusena ja hübriidina. Võrreldes erinevate lahenduste võimalusi ja puudusi otsustas autor luua nutiseadme kuva progressiivse veebirakendusena. PWA suurimaks plussiks on madalad haldus- ja arenduskulud. PWA arendamiseks piisab esirakenduse baaskeelte ja raamistike tundmisest ning pole vaja õppida lisaks põlisrakenduse või hübriidrakenduse loomiseks vajalikke raamistikke ja keeli. Seetõttu on PWA puhul Haapsalu kolledži jaoks edaspidi tunniplaani uuendamine ja haldamine oluliselt lihtsam. Lisaks annab PWA kasutajale võimaluse otsustada, kas ta soovib rakenduse alla laadida või kasutada seda läbi veebilehitseja.

Praktilises osas arendati edasi valikpraktika õppeaine raames loodud tunniplaani rakendust, arendades välja nutiseadme kuva ning lisades juurde funktsionaalsusi. Veendumaks selle kasutusmugavuses viis autor läbi testimise.

Diplomitöö eesmärk täideti – loodi uus kasutajasõbralik tunniplaanirakendus mis kasutab Tailwind CSS-i ja PWA omadusi.

SUMMARY

The name of this thesis is „Creating a Smart Device Display for Haapsalu College’s Timetable“. Three categories of mobile applications can be identified- native application, hybrid application and web application. The choice of the appropriate application category depends on the purpose of the application and the conditions it must meet (Litayem et al., 2015). Today's technology users expect high levels of satisfaction when using their devices. They expect to use the product without much effort and to fulfill their goals quickly and efficiently (Hinderks et al., 2022). The current College timetable was created in 2015 and it lacks a scalable smart display. During Elective Practice subject a new timetable was made, but only desktop version lacking comfort in smart devices. Based on the research carried out in Elective Practice subject, almost half of the students (46%) use also a smart device for viewing the timetable. Their feedback was that the timetable is confusing and not convenient to use.

Aim of this diploma thesis is to find out the most suitable technological solution for displaying the Haapsalu College timetable on a smart device and to create a user-friendly ready to use solution. Based on the purpose of the research, three research questions have been formulated:

- What are the disadvantages of the currently used timetable smart device display?
- What is the most suitable technological solution for displaying the timetable of Haapsalu College on a smart device?
- Which Haapsalu College timetable smart device display is the most user-friendly?

The method of this thesis research was to first compare mobile application solutions and find out which fulfills the needs of Haapsalu College the most and at the same time offers high level of user satisfaction. Secondly, the thesis is based on the research conducted during the Elective Practice subject. The method of research was an online questionnaire filled out by students and lecturers. Also an in-depth interview was carried out with the staff member who manages the timetable. Thirdly, prototype and ready to use solution were tested by the users.

Timetable application created as part of the elective practice subject was further developed, developing the display of the smart device and adding additional functionalities. As the result of the thesis, ready to use smart device friendly timetable was created which offers high level of user satisfaction.

KASUTATUD ALLIKAD

- Alves, R., Valente, P., & Nunes, N.J. (2014). The state of user experience evaluation practice. *NordiCHI '14: Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*. [2023, veebruar 6]. <https://doi.org/10.1145/2639189.2641208>
- Adobe. (2023). *Push notification guide for marketers – what they are, design tips, best practices, and more*. [2023, aprill 30]. <https://business.adobe.com/blog/basics/push-notification-guide>
- Bennett, K.P., Nagy, A.L. & Flach, J.M. (2012). *Handbook of Human Factors and Ergonomics*. Wiley & Sons.
- Benyon, D. (2019). *Designing User Experience: Fourth Edition*. Pearson Educacion.
- Boll, F., & Brune, P. (2015). User Interfaces with a Touch of Grey? – Towards a Specific UI Design for People in the Transition Age. *Procedia Computer Science*, 63, 511-516. [2023, veebruar 6]. <https://doi.org/10.1016/j.procs.2015.08.377>
- Borisa, K. (2022). *Progressive Web Apps (PWAs): A growing trend of 2022?* LinkedIn. [2023, märts11]. https://www.linkedin.com/pulse/progressive-web-apps-pwas-growing-tech-trend-2022-kesha-borisa/?trk=public_profile_article_view
- Bouchrika, I. (2023). *Mobile vs Desktop Usage Statistics for 2023*. Research. [2023, jaanuar 19]. <https://research.com/software/mobile-vs-desktop-usage>
- Buis, E.E.G., Ashby, S.S.R., & Kouwenberg, K.P.A. (2023). Increasing the UX maturity level of clients: A study of best practices in an agile environment. *Information and Software Technology*, 154. [2023, veebruar 6]. <https://doi.org/10.1016/j.infsof.2022.107086>
- Cajander, A., Larusdottir, M. & Geiser, J. L. (2022). UX professionals' learning and usage of UX methods in agile. *Information and Software Technology*, 151. [2023, jaanuar 19]. <https://doi.org/10.1016/j.infsof.2022.107005>
- Dafda, C. (2022). *Progressive Web Apps: Advantages and Disadvantages For Your Business*. GeekyAnts. [2023, märts11]. <https://geekyants.com/blog/progressive-web-apps-advantages-and-disadvantages-for-your-business/>
- Developer. (2014). *Swift Has Reached 1.0*. [2023, jaanuar 15]. <https://developer.apple.com/swift/blog/?id=14>

- Developer. (2023). *Documentation*. [2023, jaanuar 15].
<https://developer.apple.com/documentation/technologies>
- Enes, S. (2022). *Bootstrap vs Tailwind CSS. Which One Do You Need?* Medium. [2023 märts 12].
<https://medium.com/codex/bootstrap-vs-tailwind-css-which-one-do-you-need-9dfb3ef26ac0>
- Enihe, R. & Joshua, J., (2020). *Hybrid Mobile Application Development: a Better Alternative to Native*. Global Scientific Journals, 8. [2023, veebruar 6].
<https://doi.org/10.11216/gsj.2020.05.39825>
- Gerchev, I. (2022). *Tailwind CSS: Craft Beautiful, Flexible, and Responsive Designs*. Sitepoint. [2023, veebruar 6]. <https://www.sitepoint.com/premium/books/tailwind-css-craft-beautiful-flexible-and-responsive-designs/>
- Google Developers. (2023). *Developer Guides*. [2023, jaanuar 15].
<https://developer.android.com/guide>
- Hassenzahl, M. (2010). Experience design: technology for all the right reasons. *Synthesis Lectures on Human-centered Informatics*, 3, 1–95.
<https://doi.org/10.2200/s00261ed1v01y201003hci008>
- Hinderks, A., Mayo, F.J.D., Thomaschewski, J., & Escalona, M.J. (2022). Approaches to manage the user experience process in Agile software development: A systematic literature review. *Information and Software Technology*, 150. [2023, veebruar 6].
<https://doi.org/10.1016/j.infsof.2022.106957>
- Huber, S., Demetz, L., & Felderer, M. (2021). PWA vs the Others: A Comparative Study on the UI Energy-Efficiency of Progressive Web Apps. *Web Engineering*, 127, 464-474. [2023, märts11]. https://doi.org/10.1007/978-3-030-74296-6_35
- Hume, D.A. (2017). *Progressive Web Apps*. Manning.
- Inukollu, V.N., Keshamoni, D.D., Kang, T., & Inukollu, M. (2014). Factors influencing quality of mobile apps: role of mobile app development life cycle. *International Journal of Software*, 52, 15-34. [2023, jaanuar 15]. <https://doi.org/10.48550/arXiv.1410.4537>
- ISO Standards. (2023a). *ISO 9241-210:2019: Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems*. [2023, veebruar 6].
<https://www.iso.org/standard/77520.html>

- ISO Standards. (2023b). *ISO 9241-11:1998: Ergonomic requirements for office work with visual display terminals (VDTs)*. [2023, veebruar 6]. <https://www.iso.org/standard/16883.html>
- Kashfi, P., Feldt, R., & Nilsson, A. (2019). Integrating UX Principles and Practices into Software Development Organizations: a Case Study of Influencing Ivents. *Journal of Systems and Software*, 154, 37-58. [2023, veebruar 6]. <https://doi.org/10.1016/j.jss.2019.03.066>
- Khan, A.I., Al-Badi, A., & Al-Kindi, M. (2019). Progressive Web Application Assessment Using AHP. *Procedia Computer Science*, 155, 289-294. [2023, märts11]. <https://doi.org/10.1016/j.procs.2019.08.041>
- Kim, J. (2015). *Design for Experience: Where Technology Meets Design and Strategy*. Springer.
- Krishnan, A. (2021). *Cross Compilation with example*. Medium. [2023, aprill 11]. <https://arunk2.medium.com/cross-compilation-with-example-39ff29889afd>
- Krueger, A.E., Pollmann, K., Fronemann, N., & Foucault, B. (2020). Guided User Research Methods for Experience Design—a New Approach to Focus Groups and Cultural Probes. *Multimodal Technologies and Interaction*, 43. [2023, veebruar 6]. <https://doi.org/10.3390/mti4030043>
- Kuitunen, M. (2019). *Cross-platform mobiile application development with react native (bakalaureusetöö)*. Tampere University. [2023, jaanuar 19]. <https://trepo.tuni.fi/handle/123456789/27139>
- Lamhaddab, K., Lachgar, M. & Elbaamrani, K. (2019). Porting Mobile Applications from iOS to Android: A Practical Experience. *Mobile Information Systems*, 2019, 29. [2023, jaanuar 15]. <https://doi.org/10.1155/2019/4324871>
- Litayem, N., Dhupia, B., & Rubab, S. (2015). Review of Cross-Platforms for Mobile Learning Application Development. *International Journal of Advanced Computer Science and Applications*, 6(1). [2023, jaanuar 15]. <https://doi.org/10.14569/IJACSA.2015.060105>
- Majchrzak, T.A., Biorm-Hansen, A., & Gronli, T-M. (2018). Progressive Web Apps: the Definite Approach to Cross-Platform Development? *Hawaii International Conference on System Sciences*. [2023, märts11]. <https://doi.org/10.24251/HICSS.2018.718>

- Malavolta, I., Procaccianti, G., Noorland, P., & Vukmirovic, P. (2017). Assessing the Impact of Service Workers on the Energy Efficiency of Progressive Web Apps. *VU Research Portal (Elsevier)* . [2023, märts 11]. <https://doi.org/10.1109/MOBILESoft.2017.7>
- Medium. (2019). *PWA can be added to the App Store and Google Play Store*. [2023, märts 11]. <https://medium.com/progressivewebapps/pwa-can-be-added-to-the-app-store-and-google-play-store-33522751d37d>
- Microsoft. (2023). *What is .NET MAUI?*. [2023, april 11]. <https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui?view=net-maui-7.0>
- Nguyen, T. T., Minh Vu, P., Pham, H. V., & Nguyen, T.T. (2018). Deep learning UI design patterns of mobile apps. *ICSE-NIER '18: Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging*, 65-68. [2023, veebruar 6]. <https://doi.org/10.1145/3183399.3183422>
- Punchoojit, L., & Hongwarittorn, N. (2017). Usability Studies on Mobile User Interface Design Patterns: A Systematic Literature Review. *Advances in Human-Computer Interaction*, 16, 1-22. [2023, veebruar 6]. <https://doi.org/10.1155/2017/6787504>
- Rakestraw, T.L., Rangamohan V.E., & Kasuganti, R.R. (2013). *The mobile apps Industry: A case study*. Economics. [2023, jaanuar 19]. <https://www.aabri.com/manuscripts/131583.pdf>
- Rego, F., Portela, F., & Santos, M.F. (2019). Towards PWA in Healthcare. *Procedia Computer Science*, 160, 678-683. [2023, märts11]. <https://doi.org/10.1016/j.procs.2019.11.028>
- Silkalns, A. (2023). 12 Best Open-Source CSS Frameworks 2023. *Colorlib*. [2023 märts 12]. <https://colorlib.com/wp/free-css3-frameworks/>
- Silva, D.T.S., Silveira, M.S., Maurer, F., & Silveira, F.F. (2018). The Evolution of Agile UXD. *Information and Software Technology*, 102, 1-5. [2023, veebruar 6]. <https://doi.org/10.1016/j.infsof.2018.04.008>
- Singh, M., & Shobha, G. (2021). Comparative Analysis of Hybrid Mobile App Development Frameworks. *International Journal of Soft Computing and Engineering*, 10, 21-26. [2023, veebruar 6]. <https://doi.org/10.35940/ijscce.F3518.0710621>
- Statcounter. (2022). *Mobile Operating System Market Share Worldwide*. [2023, jaanuar 15]. <https://gs.statcounter.com/os-market-share/mobile/worldwide>

- Steiner, T. (2018). What is in a Web View: An Analysis of Progressive Web App Features When the Means of Web Access is not a Web Browser. *Companion of the Web Conference*. [2023, märts11]. <https://doi.org/10.1145/3184558.3188742>
- Tammets, P. (s.a.) *Kasutajakeskne disain ja prototüüpimine*. Tallinna Ülikool. <https://web.htk.tlu.ee/digitalu/disain/>
- Titus, J. (2017). *Empowering developers to build the best experiences across platforms*. Google Blog. [2023, jaanuar 15]. <https://blog.google/technology/developers/empowering-developers-build-best-experiences-across-platforms/>
- Upland. (2023). *25% of Users Abandon Apps After One Use*. [2023, jaanuar 15]. <https://uplandsoftware.com/localytics/resources/blog/25-of-users-abandon-apps-after-one-use/>
- Vallaste, H. (s.a.). E-teatmik. <http://www.vallaste.ee>
- Velvet. (2017). *Tallinna Ülikooli Stiiliraamat*. [2023, aprill 26]. <https://www.tlu.ee/public/CVI-2017/mobile/index.html>
- Vilcek, T., & Jakopc, T. (2017). Comparative analysis of tools for development of native and hybrid mobile applications. *40th International Convention on Information and Communication Technology, Electronics and Microelectronics*. [2023, jaanuar 19]. <https://doi.org/10.23919/mipro.2017.7973662>
- Vogt, J., & Meier, A. (2010). *An Adaptive User Interface Framework for eHealth Services based on UIML*. *Bleed 2010 Preceedings*. [2023, veebruar 6]. <https://aisel.aisnet.org/bleed2010/13/>

LISA 1. VALIKPRAKTIKA ÕPPEAINE ANKEETKÜSIMUSTIK

1. Millises seadmes tunniplaani peamiselt kasutad?

- Telefon
- Arvuti
- Muu

2. Kui arusaadav oli sulle Haapsalu Kolledži tunniplaan selle esmasel avamisel?

- Segane
- 1
- 2
- 3
- Arusaadav

3. Kas oled tutvunud tunniplaani juhendiga?

- Jah
- Ei

4. Mis teeb praeguse tunniplaani kasutamise lihtsaks?

5. Mis teeb praeguse tunniplaani kasutamise keeruliseks?

6. Milliseid järgnevatest funktsioonidest oled kasutanud?

- Tunniplaani printvaade
- Lae tunniplaani alla CSV formaadis
- Lae alla iCal fail
- Lisa tunniplaani Google kalendrisse
- Rss voog
- Vaata tunniplaani mobiilis
- Õpetus
- Tume stiil

- Mitte ühtegi eelnimetatutest

7. Kui arusaadavaks pead tunniplaanis kasutatavaid ikoone?

- Teaksin ilma kirjelduseta, milliseid funktsioone ikoonid tähistavad.
- Võin aimata, millised funktsioonid ikoonide taga on.
- Ilma kirjelduseta ei saaks ma aru, mida üks või teine ikoon tähistab.

8. Kui tihti kasutad tunniplaani kuvamisel koondtabeli võimalust?

- Tihti
- Vahel
- Harva
- Mitte kunagi

9. Kui tihti kasutad tunniplaani kuvamisel kuupäevade vahemiku määramist?

- Tihti
- Vahel
- Harva
- Mitte kunagi

10. Milliseid järgevatest kitsendustest kasutad tunniplaani otsingumootoris?

- Valin kursuse
- Valin õppeaine
- Valin õppejõu
- Valin ruumi
- Mitte ühtegi eelnimetatutest

12. Mida praeguse tunniplaani visuaali juures muudaksid?

13. Kas oled kokku puutunud parema tunniplaani süsteemiga kui Haapsalu Kolledži oma?

- Jah
- Ei

14. Millise kooli tunniplaaniga?

15. Mis sulle selle tunniplaani juures meeldis?

16. Mis on sinu roll Haapsalu Kolledžis?

- Tudeng
- Õppejõud
- Muu

17. Mitmendal kursusel sa õpid?

- Esimesel kursusel
- Teisel kursusel
- Kolmandal kursusel

18. Mis erialal sa õpid?

- Rakendusinformaatika
- Käsitöö tehnoloogiad ja disain
- Tervisejuht
- Liiklusohutus

19. Milliseid lisavõimalusi võiks tunniplaan sisaldada?

20. Kas soovid midagi lisada?

LISA 2. RAKENDUSE KIRJELDUS PROGRESSIIVSELE VEEBIRAKENDUSELE

```
{  
  
  "short_name": "Tunniplaan",  
  
  "name": "Haapsalu kolledži tunniplaan",  
  
  "icons": [  
  
    {  
  
      "src": "/assets/192x192.png",  
  
      "sizes": "192x192",  
  
      "type": "image/png"  
  
    },  
  
    {  
  
      "src": "/assets/512x512.png",  
  
      "sizes": "512x512",  
  
      "type": "image/png"  
  
    }  
  
  ],  
  
  "display": "standalone",  
  
  "background_color": "#fff",  
  
  "theme_color": "#fff"  
  
}
```

LISA 3. TESTIMISÜLESANDED

Roll	Testlood
Tudeng	<ul style="list-style-type: none"> • Proovige vaadata ainult enda kursuse toimumiskordi • Proovige vaadata ainult enda kursuse toimumiskordi • Proovige vaadata järgmise kuu toimumiskordi • Proovige sisse logida • Proovige vaadata millises aines on jäetud iseseisvad tööd • Proovige õppeinfost vaadata ainekaarti • Proovige leida tunniplaanist toimumiskorrad, mis on planeeritud videoloenguna
Õppejõud	<ul style="list-style-type: none"> • Proovige vaadata ainult teie poolt antavate õppeainete toimumiskordi • Proovige vaadata käesoleva nädala toimumiskordi • Proovige vaadata järgmise kuu toimumiskordi • Proovige sisse logida • Proovige lisada mõnele õppeainele 2 iseseisvate tööd • Prooviga lisada õppeainele videoloeng • Proovige õppeinfost vaadata ainekaarti
Haldus	<ul style="list-style-type: none"> • Proovige lisada kaks õppeaine toimumiskorda • Proovige lisada õppeaine toimumiskord lisades täiesti uue õppeaine ja õppejõu • Proovige muuta mõne õppejõu emailiaadress • Proovige muuta õppeaine toimumiskorda lisades veel üks õppejõud • Proovige kustutada mõni õppeaine • Proovige kustutada õppeaine toimumiskord • Proovige lisada toimumiskorrale videoloengu link ja kommentaar • Proovige lisada tunniplaani kirjetele ruume