TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Melvin Kriisa 155252IAPB

EESTIKEELNE MOBIILIRAKENDUS IOS PLATVORMIL EESTIS KASVAVATE SEENTE TUVASTAMISEKS

Bakalaureusetöö

Juhendaja: Viljam Puusep
MSc

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Melvin Kriisa

04.01.2021

Annotatsioon

Käesoleva lõputöö eesmärgiks on luua hõlpsasti kasutatav Eestis kasvavate seente tuvastamise mobiilirakendus *iOS* platvormile. Loodav rakendus on eestikeelne ning võimaldab tuvastada seent või seeni reaalajas kaameraga skaneerides. Lisaks on kasutajal võimalus positsioneerida ning oma asukoha ümbrust vaadelda kasutades kaarti. Rakenduses saab söögikõlblikkuse kategooriate vahel valides näha erinevate seente nimekirja. Samuti on võimalus tutvuda konkreetse seene põhjaliku kirjelduse ning pildiga. Söögikõlblikkuse kategooriaid on rakenduses läbivalt kuvatud erinevate värvidega: roheline, oranž, punane.

Töös kirjeldab autor olemasolevaid lahendusi *iOS* operatsioonisüsteemil, seehulgas tuues välja positiivseid ja negatiivseid omadusi. Samuti kirjeldatakse töö ettevalmistamisprotsessi, valminud rakenduse vaateid, funktsionaalsust ning analüüsitakse seeneliikide tuvastamistäpsust. Antakse ülevaade võimalikest rakenduse edasiarendamistest ja viisidest tuvastamistäpsuse tõstmiseks.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 23 leheküljel, 5 peatükki, 20 joonist, 1 tabelit.

Abstract

Estonian iOS Mobile Application for the Identification of Mushrooms Growing in Estonia

The aim of this thesis is to create a user-friendly mobile application for *iOS* platform, which detects mushrooms growing in Estonia. Application is in Estonian and allows to identify mushrooms by scanning with the camera in real time. As an additional feature user has the opportunity to position and to observe surroundings of his location using a map. User can also see a list of different mushrooms, which are divided by edibility: edible, edible after cooking, poisonous. Different colours, as green, orange and red, are used to display categories of edibility throughout the application.

The thesis provides an overview of the existing mushroom detecting applications on *iOS* platform, including the positive and negative features. The mobile views and functionality of the completed application are also described and the detection accuracy of mushrooms is analysed. An overview of possible further developments of the application and ways to increase detection accuracy are given.

The implementation of user-friendly mobile application was done by using programming language *Swift*. *Python* was used to generate image links to a text file based on information in *CSV* file format documents. Documents were obtained from the *Mushroom Observer* website. Images were downloaded from that text file using the *cURL* tool. Images were used to train the *Core ML* model in *Create ML*. The basic functionality was tested on the basis of pocketbook pictures.

The thesis is in Estonian and contains 23 pages of text, 5 chapters, 20 figures, 1 table.

Lühendite ja mõistete sõnastik

Adobe XD	Vahend disaini prototüüpide valmistamiseks
API	Application Program Interface ehk tarkvaraliides
Apple	Riist- ja tarkvara tootev ning arendav rahvusvaheline ettevõte
App Store	Apple poolt loodud platvorm rakenduste jagamiseks
Core ML	Raamistik, mille abil saab integreerida masinõppe mudeleid iOS rakendustesse
CocoaPods	Tarkvarakomplektide ja pakkide haldamissüsteem
Create ML	Raamistik, mille abil saab luua Core ML mudeleid
CSV failiformaat	Tekstifail, milles andmed eraldatud komadega
cURL	Vahend andmete ülekandmiseks serverist ja/või serverisse
DB Browser for SQLite	Vahend andmebaasi haldamiseks
GitLab	Veebipõhine tarkvaraarendusplatvorm
IA	Arvutisüsteemide instituut
iOS	Apple poolt loodud nutiseadmete operatsioonisüsteem
iPhone	Apple poolt toodetud nutitelefon
Mudel	Kogumik andmetest, mis on masinõppe meetodiga treenitud, et tuvastada sarnaste tunnustega väärtusi
Mushroom Observer	Veebileht, kust hangiti seene pildid mudeli treenimiseks
Python	Programmeerimiskeel
SQLite	Andmebaasi raamistik
Swift	Programmeerimiskeel, mis on välja arendatud Apple poolt
Xcode	Apple poolt loodud programmeerimiskeskkond

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

1.1 Taust

Esimesed seened tekkisid Maale ligikaudu pool miljardit aastat tagasi. Juba sellest ajast on seened olnud hinnatud osa nii loomade kui ka inimeste toidulaual. Samuti kasutatakse seeni meditsiinis ja lõngatööstuses. Vaatamata sellele, et seente korjamine on tänapäeval üheks vaba aja veetmise viisiks, on inimeste teadmised seente söödavuse ja mürgisuse kohta kohati puudulikud. Seened kasvavad aedades, parkides ja metsades, kuid tihtipeale on nende välimus petlik või eemaletõukav ning seetõttu on levinud arusaam, et konkreetne seen pole söödav ja tuleks hoiduda selle korjamisest [1].

Tänapäeva moodsas ühiskonnas räägitakse palju tervislikust toitumisest ja jätkusuutliku toidusüsteemi loomisest. Üheks oluliseks väljakutseks on liha tarbimise vähendamine ning uudsete lihatoodete välja töötlemine. Sellega seoses on hakatud seeni lisama seguna lihatoodetesse ning samuti on püütud neid ka asendada. Seened sisaldavad antioksüdante ning omavad antimikroobseid omadusi. Seened on kõrge toiteväärtusega. Nad sisaldavad vähe kaloreid ja on rikkad kiudainete ning valkude poolest. Tuginedes eelnevale on võimalik lihatoodete toiteväärtust tõsta ning seeläbi parandada toidu säilimist [2], [3].

Masinõpe on andmete analüüsimise metoodika, mille tulemusena suudab süsteem tuvastada sarnasusi ja mustreid ning teha eelneva põhjal otsustavaid valikuid ilma, et inimene peaks ise otseselt sekkuma. Tänapäeval kasutatakse masinõpet näiteks meditsiini-, äri- ning tööstusvaldkonnas. Samuti võimaldavad kaasaegsed nutiseadmed kasutada olemasolevaid masinõppe algoritme erinevate probleemide lahendamiseks. Näiteks üheks nutiseadmete turvaelemendiks olev näotuvastus kasutab masinõppe algoritme [4].

1.2 Probleem

Eestis on teadaolevalt umbes 5500 erinevat seeneliiki. Söödavaid seeni on sealhulgas aga ligikaudu 400. Nendest umbes 60 on hästi ning lihtsalt äratuntavad ning leidnud

heakskiidu korjamiseks ning tarbimiseks. See-eest mürgiseid seeneliike on Eestis umbes 200 piires. Looduses võib kohata mitmeid seeneliike, mis on välimuselt sarnased, kuid söögikõlbulike omaduste poolest võib tegemist olla nii mürgiste kui ka söödavate seentega. Mitmed seeneliigid võivad põhjustada inimestel ohtlikke mürgistusi, mitmesuguseid tervisekahjustusi või koguni letaalset lõppu [5].

Eesti etnobioloogias leidub teavet seente kohta üpriski vähe. Tänapäeval korjatakse seeni rohkem kui vanasti [5]. Seni on seente tundmaõppimiseks ning identifitseerimiseks kasutatud teavikuid, mis sisaldavad põhjalikke kirjeldusi ning pilte seente tuvastamiseks, kuid mille kasutamine on tihti aeganõudev ning seeneliigi lõplik kindlaks määramine on keeruline. Objektide tuvastus piltidelt masinõppe kaudu on aastatega kasvanud, kuid just seente tuvastus fotode põhjal pole veel nii laialdaselt levinud. Peamiselt on olemas ingliskeelsed mobiilirakendused, kuid eestikeelsed *iOS* versioonid lõputöö kirjutamise ajal puudusid. Lisaks leiab autor, et praeguste versioonide kasutajamugavus pole kõige parem.

1.3 Eesmärk

Käesoleva lõputöö eesmärk on luua hõlpsasti kasutatav eestikeelne mobiilirakendus, mis töötab *iOS* operatsioonisüsteemil ning suudab tuvastada Eesti metsades kasvavaid seeni kaamerapildi põhjal. Kuna autoril puudus varasem kogemus *iOS* arendusega ning sooviti selles valdkonnas areneda, otsustati rakendus teha just sellele platvormile. Rakenduse abil on lihtne, mugav ning kerge leida teavet ning detailseid kirjeldusi erinevate seente kohta. Seene tuvastamisel pakutakse välja kuni viis sarnaste väliste tunnustega seent. Pakkumine hõlmab endas pilte, seente nimetusi ning söögikõlblikkust.

1.4 Ülevaade tööst

Töö alguses antakse ülevaade ettevalmistusest, tööprotsessist ja kasutatud tehnoloogiatest. Kirjeldatakse ja analüüsitakse olemasolevate rakenduste funktsionaalsust ning kasutajamugavust. Kolmandas peatükis näidatakse, milline rakendus valmis koos funktsionaalsusega. Neljandas peatükis tuuakse välja tulemused ning analüüsitakse rakenduse põhifunktsionaalsust. Samuti kirjeldatakse võimalusi rakenduse edasiarendamiseks.

2 Metoodika

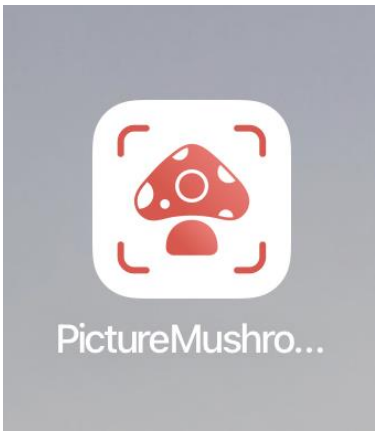
Käesolevas peatükis analüüsitakse olemasolevate rakenduste (*iOS* platvormil) kasutajamugavust ning funktsionaalsust. Tuuakse välja nende positiivsed ja negatiivsed omadused. Lisaks kirjeldatakse ettevalmistusprotsessi ning arendust. Kirjeldatavad rakendused olid lõputöö kirjutamise ajal vabalt saadaval *App Store*'s ning rakendusi analüüsiti *iPhone X* vaate põhjal.

2.1 Olemasolevate lahenduste kirjeldus ja disaini analüüs

Lahenduste analüüsil võeti aluseks kasutajaliideste ja kasutajamugavuse disainimise printsiibid ning põhimõtted [6]. Rakendustel on seente tuvastamise funktsionaalsus ning võimalik näha seene põhjalikku kirjeldust.

2.1.1 Picture Mushroom – Mushroom ID

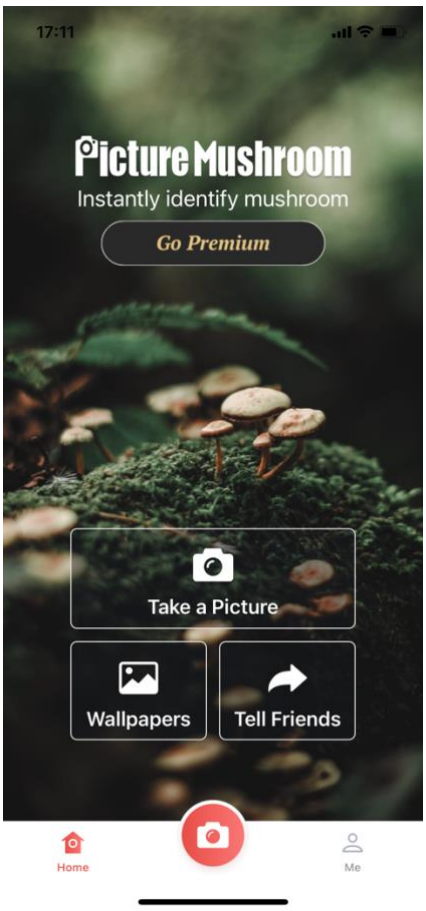
Rakendus on mitmekeelne, kuid eesti keel siiski puudub. Põhiikoonil (Joonis 1) paikneb keskel üksik seen ning seda ümbritsevad nurkades raamid. Ikoon annab selgelt märku, et tegu on seene skaneerimisega.



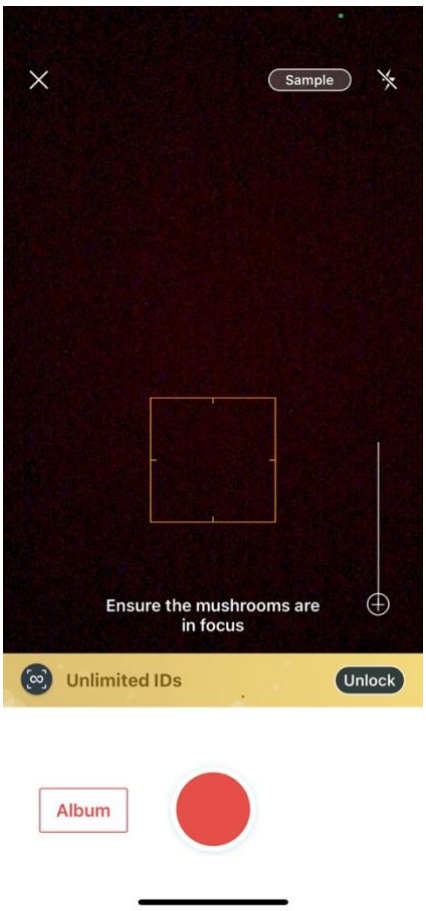
Joonis 1. *Picture Mushroom - Mushroom ID* põhiikoon

Rakenduse avanedes hakkab silma navigatsiooniriba, millel paiknevad 3 nuppu. Nupud on selgelt eristatavad ja etteaimatav talitus on ootustele vastav. *Home* vaates (Joonis 2) leiab kasutaja kokku 4 nuppu, mis on samuti selgekujulised ning lisatud on selgitavad

fraasid. Võimalik on liituda *Premium* paketiga, mille tasu on 21.49 eurot aasta kohta. *Premium* pakett sisaldab piiramatul hulgal seente tuvastamisi, rohkem informatsiooni seente kohta, unikaalseid kõrgkvaliteedis taustapilte ning puuduvad rakendusesisesed reklaamid. Lisaks saab jagada või soovitada rakendust teistele inimestele, mille tulemusena genereeritakse viide rakendusele *App Store*'st. Taustapiltide vaates näeb ainult nelja erinevat pilti. Vaate kerimine pole võimalik ja kerimise viipe peale näidatakse kasutajale *Premium* paketiga liitumise vaadet. Samuti saab pilte märkida meeldivaks või nutitelefoni galeriisse laadida. Seente tuvastamisvaates (Joonis 3) puudub reaalajas kaameraga kindlaks tegemise võimalus. See-eest võib koheselt pildistada või hoopiski üles laadida juba olemasolev telefonist. Kaamerat saab juhtida järgmiste funktsioonidega: pildi suurendamine, tagavalgustus. Kasutajal on võimalus ka selles vaates liituda *Premium* paketiga. Vaates leiab mitmeid nuppe, kuid stiililt ühtseid tunnuseid on vähe.



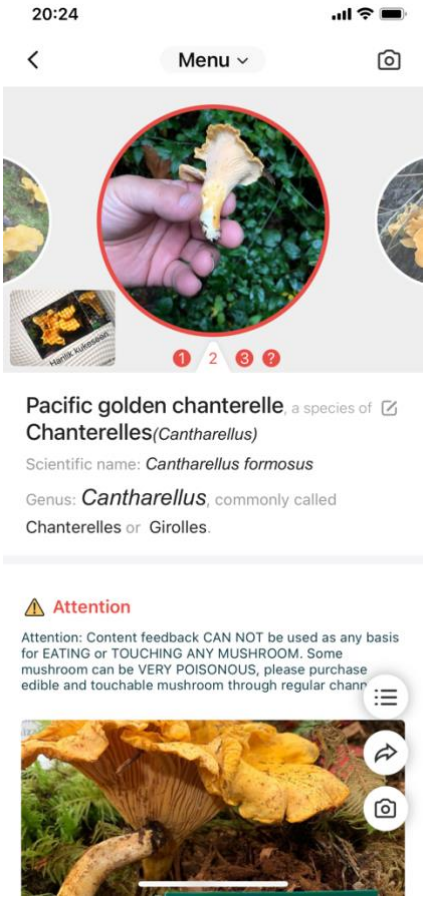
Joonis 2. *Home* vaade



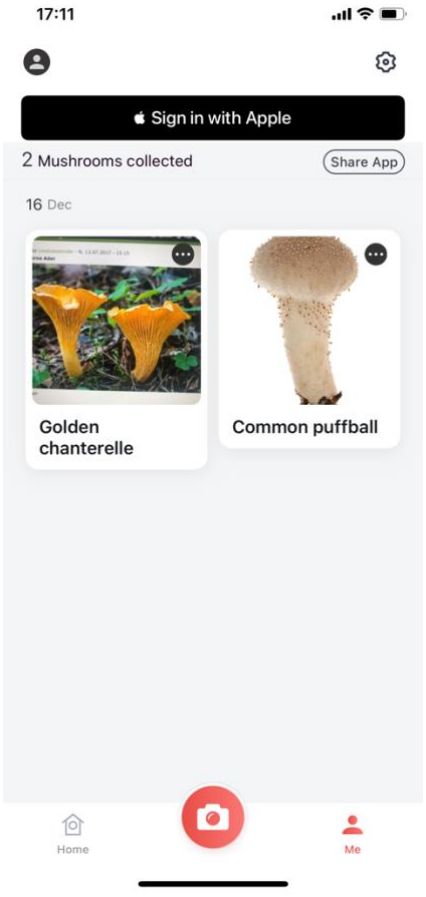
Joonis 3. Tuvastamisvaade

Tuvastatud seene vaates (Joonis 4) pakutakse ringikujulistes elementides ja erinevaid sarnaste tunnustega seeni. Ringi ümbritsev äär on punast värvi, mis aga ei ole seotud

antud seene söögikõlblikkusega ehk ei edastata visuaalselt infot. Samuti leiab konkreetse seene kohta allapoole kerides tarbimise hoiatuse, põhjaliku kirjelduse, korduma kippuvate küsimuste sektsiooni ning galerii samasse seeneliiki kuuluvate seentega. Galerii paikneb nii vaate alguses kui ka lõpus, mis pole kõige mugavam lahendus. Kirjelduses esitatav üldine tekst on ühetaoline, alapealkirju pole välja toodud ning selle tulemusena võib väga lihtsalt järje kaotada kirjeldust lugedes. Ülemisel navigatsiooniribal paiknevad nupud, dubleerivad all paremal nurgas olevaid nuppe. Tuvastatava seene vaade on kokkuvõtlikult kirju ja visuaalselt keerukas on eristada erinevaid elemente ning ühtne stiil pole ka kõige paremini realiseeritud. Rakendusse on võimalik sisse logida *Apple*'i kontot kasutades, lisada profiilipilt ning vahetada hüüdnime. Seente ajaloo vaates (Joonis 5) leiab seni tuvastatud seened ning neid on võimalik ajaloost eemaldada ning samas ka tulemusi muuta.



Joonis 4. Tuvastatud seene vaade



Joonis 5. Ajaloo vaade

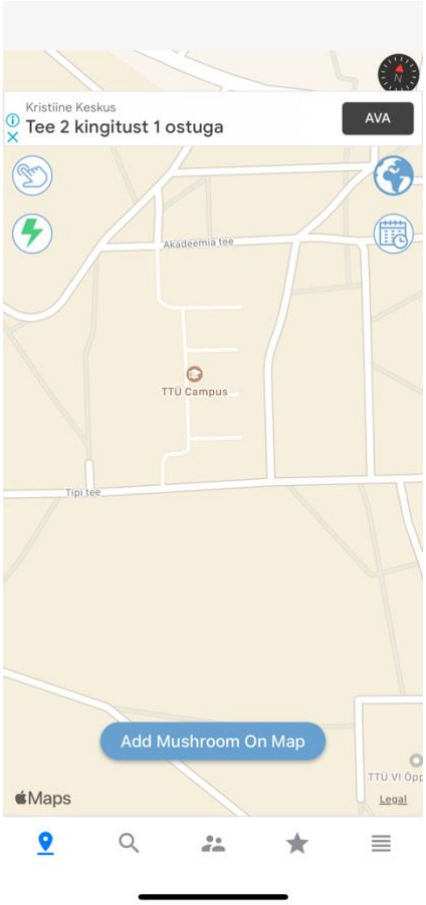
2.1.2 Mushroom Identificator

Rakendus on ainult inglise keeles, kuid see-eest on võimalik tuvastada seeni nii pilte üles laadides kui ka kaameraga reaalajas skaneerides. Põhiikoonil (Joonis 6) on olemas luup, mis osutab seenele. Selline kooslus edastab rakenduse kaudu visuaalselt vajalikku infot kui ka seene tuvastamist ja uurimist. Rakenduse kasutusmugavust tõstavad erinevad paketid. *Premium* pakett, mille aastatasu on 4.99. Paketis sisaldub reklaamivaba rakenduse kasutus, andmete sünkroniseerimine pilveserverisse, seente kaardistamise võimalus ja viktoriini viimaste tasemete avanemist. Täiendavalt on võimalik soetada ühekordne laiendatud versioon, mille tasu on 8.99 eurot. Laiendatud versioon võimaldab reklaamivabadust, seente käsitsi kaardistamist ning uusimate tasemete avanemist viktoriinis. Tavaversiooni täidavad aegajalt reklaamid kogu ekraani suuruses ning püsivalt asetsevad reklaamiribad nii tuvastamise, kaardi, kui ka seene kirjelduse vaates.

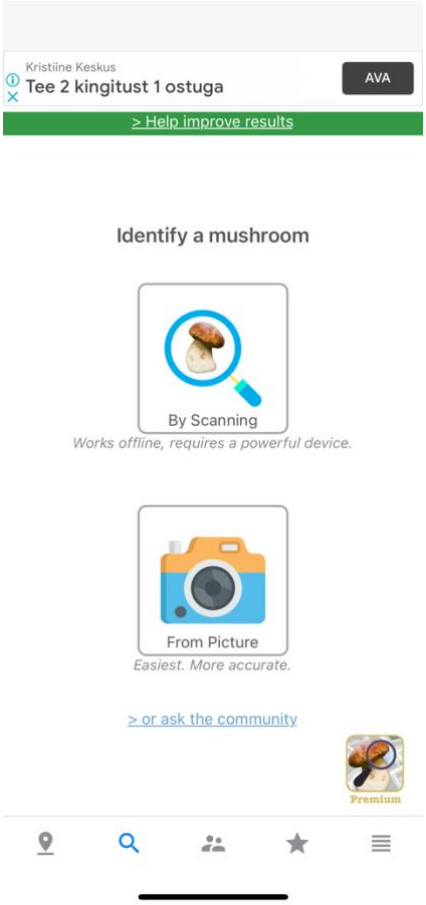
Joonis 6. *Mushroom Identificator* põhiikoon

Rakenduse navigatsiooniribal on mitu nuppu, mis suunavad erinevatele vaadetele: kaardivaade, tuvastamisvaade, kogukonnavaade, viktoriinivaade, seente nimekirjavaade. Kaardivaates (Joonis 7) on võimalik valida nii satelliidi kui ka tavakaardi disaini vahel. Võimalus ka vaadata mõnede seente kasvuaega kuude lõikes. Küll aga puudub positsioneerimisnupp, mis aitaks kasutajal hõlpsasti tuvastada enda asukoht kaarti sirvides. Tuvastamisvaates (Joonis 8) saab pilti nii üles laadida kui ka reaalajas seent skaneerida. Mõlema valiku puhul küsitakse kasutaja käest, kas soovitakse lisada informatsiooni seene kohta, et suurendada tuvastamise täpsust. Kaameraga tuvastades antakse koheselt ette kolm võimalikku vastet. Enne kaameraga tuvastamise alustamist teavitatakse kasutajat, kuidas võiks järgnev protsess kulgeda. Täienduseks on ka indikaator, mis edastab tuvastamise kestvust. Indikaatori protsess lõpupoole aeglustub ning jääb seisma ja ei lõppegi automaatselt. Pea igale elemendile on lisatud juurde

selgitav tekst. See kõik muudab kogu rakenduse kirjuks ehk kokkuvõtlikult on vaadetel liigseid elemente. Viktoriini vaates on kasutajal võimalus panna proovile oma teadmised seente tundmise alal, mis toimub punktide peale. Ette on antud neli erinevat pilti ja keskel ülaosas esitatud seeneliik. Automaatsed piirangud ekraani sättimiseks on halvasti realiseeritud, sest autori seadmes on osa elemendist ekraanivaatest väljas.



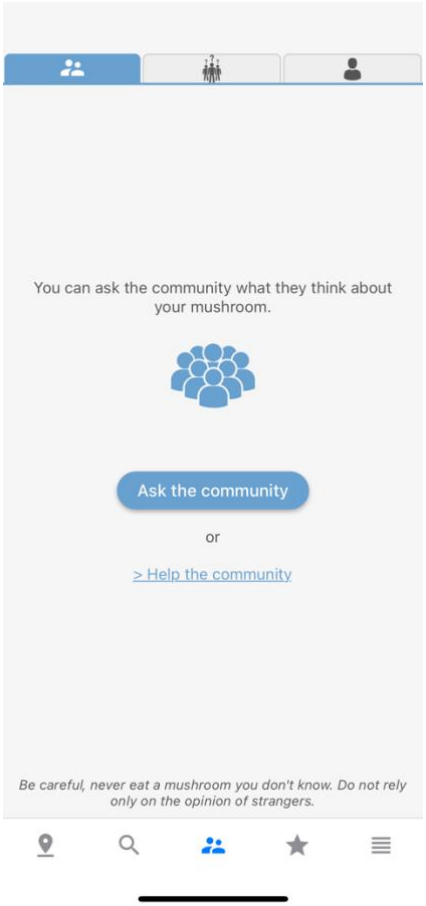
Joonis 7. Kaardivaade



Joonis 8. Tuvastamisvaade

Kogukonnavaates (Joonis 9) saab taotleda informatsiooni seene kohta, kus vastajateks on tavaliselt sama rakenduse kasutajad. Ülemisel ribal navigeerides, võimaldatakse kasutajal näha oma viktoriini tulemusi ning vastata teiste kasutajate seente tuvastamise taotlustele. Seene tuvastustaotluse esitamiseks peab ennast registreerima kasutajaks. Ka selles vaates on automaatsed ekraani piirangud nõrgalt realiseeritud. Nimekirjavaates on võimalus valida seente söögikõlblikkuse kategooriate vahel. Dubleeritav on jällegi seente kasvuaja näitamise võimalus kuude lõikes ning kasutada saab ka otsingu funktsionaalsust. Seene detailvaates (Joonis 10) näeb pilte, kirjeldust ning söögikõlblikkust. Kirjeldus on osaliselt

jaotatud alateemadeks, kuid alapealkirju pole kasutajale erimärgistatuna välja toodud. Piltide kuvamist on kasutatud selles vaates topelt: nii all kui ka üleval. Topelt elementide kuvamine muudab vaate pikemaks. Seda rakendust kasutades tuleb süveneda, sest informatsiooni ja elemente on vaadetel liiga palju.



Joonis 9. Kogukonnavaade

Joonis 10. Detailvaade

2.2 Ülevaade tööriistadest

Rakendust arendati *Xcode* keskkonnas ning peamiseks arenduskeeleks oli *Swift*. *Xcode* sisaldab simulaatorit, millega saab testida erinevaid *iOS* mudeleid virtuaalselt [7]. Kaamera testimiseks pidi kasutama reaalset seadet. Peamiselt testigi rakendust reaalsel seadmel, milleks oli *iPhone X*.

Swift on kaasaegne programmeerimiskeel, mis on loodud arendajatele mugavamaks kasutamiseks. Keele eesmärgiks on asendada *C*-põhiseid keeli, muuta süntaks lihtsamini loetavamaks ning määratlemata käitumiste puhul olla ohutum [8].

Lokaalse andmebaasina kasutati *SQLite* raamistikku, mille lähtekood on avalik ning võimalik kasutada erinevatel eesmärkidel [9]. Loodi *CSV* failiformaadiga dokument, kuhu lisati seeneliikide kohta informatsiooni, hiljem imporditi andmebaasi. Andmebaasi haldati *DB Browser for SQLite* programmi kasutades. Seente andmete hankimiseks kasutati seenestaja taskuraamatut [10].

Create ML on masinõppe vahend *macOS*'il (*Apple* arvuti operatsioonisüsteem), millega saab luua võimsaid mudeleid. Erinevateks mudeli tüüpideks on: objekti- ja pildituvastus, helituvastus, tekstituvastus, tabelite ja liigutuste klassifitseerimine [11]. Pilte, mudeli treenimise jaoks, koguti veebilehelt nimega *Mushroom Observer*, kus kasutajad postitavad pilte seentest ning otsivad infot seente tuvastamiseks. Piltide kasutamine veebilehekülgelt on lubatud [12].

Python'it kasutati *CSV* failiformaadi dokumentidest (sisaldus info *Mushroom Observer* veebilehe piltidest) linkide genereerimiseks. Linkide kaudu toimus piltide allalaadimine *cURL* vahendiga.

Adobe XD programmi kasutati navigatsiooniriba ikoonide disainimiseks ning valmistamiseks. Versioonihalduseks kasutati *GitLab*'i, mis tagas kindla salvestuse repositooriumisse lõputööga seotud dokumentide ja lähtekoodidega.

2.3 Ülevaade protsessist

Esmase sammuna võeti ühendust *Mushroom Observer* veebilehe administraatoriga, kes saatis lahkelt *CSV* failiformaadis dokumendid, mille abil oli võimalik seeneliikide pildid alla laadida. *Python*'is loodi koodijupp (Lisa 2), mis kasutas *Mushroom Observer* dokumente ning, mis tekitas iga seeneliigi kohta tekstifaili, kuhu olid lisatud lingid igale pildile. Käsurealt, *cURL* vahendiga, sai pildid alla laetud kiiremini kui seda võinuks teha *Python*'is. Alla laetud pilte oli ligikaudu 30000, kuid nende hulgas polnud kõik õiged seeneliigid ja pildid, seega esines palju müra, mida pidi sorteerima.

Algselt valiti 90 erinevat seeneliiki, kuid sorteerimise käigus jäi alles vaid 79. Valiku määramisel arvestati iga seenerühma kuuluva seeneliigiga. Seente andmed koguti seenestaja taskuraamatu [10] abil *CSV* formaadis faili ning hiljem imporditi andmebaasi.

Create ML'i abil katsetati erinevate parameetritega treenimisprotseduure ja valmis mudeleid. Rakenduse funktsionaalsust arendati *Xcode*'s ja testiti simulaatoril ning reaalsel seadmel *iPhone X*. *GitLab*'is hoiti kõike faile, mis seotud lõputööga.

Pärast rakenduse valmimist katsetati seente tuvastamise täpsust seenestaja taskuraamatu [10] piltide põhjal, mida ei kasutatud mudeli treenimisel.

3 Tulemused

Käesolevas peatükis tuuakse välja ning kirjeldatakse valminud rakendust. Rakendusele arendati 5 vaadet: kaardivaade ehk avavaade, tuvastamisvaade, tuvastatud seene vaade, nimekirjavaade, detailvaade.

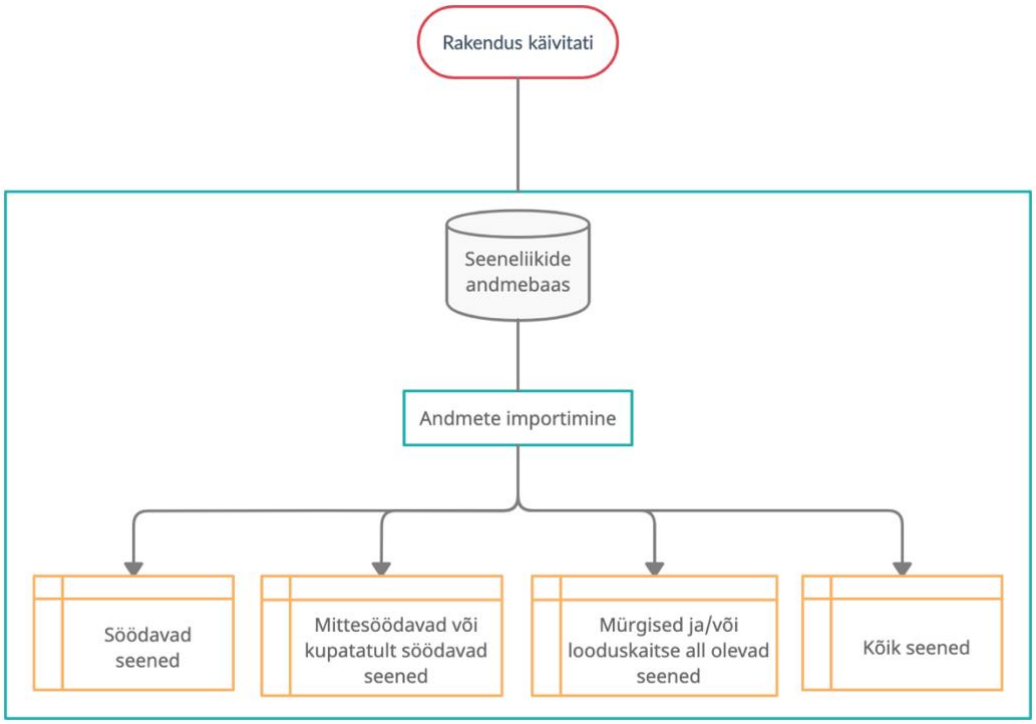
3.1 Üldine funktsionaalsus

Automaatsed ekraani piirangud on seadistatud igal vaatel. See kindlustab elementide korrektset paiknemist nii suurematel kui ka väiksematel ekraanidel. Rakendusele on pandud rõhku visuaalse ning intuitiivse informatsiooni jagamisele. Navigatsiooniriba on all püsiv kogu rakenduse kasutamise vältel. Sel asetsevad kolm nuppu. Esimene nupp viitab kaardivaatele, teine ehk keskmine nupp viitab tuvastamisvaatele, kolmas nimekirjavaatele, kust saab edasi liikuda detailvaatesse. Kõikidel nuppudel on ühine animatsioon. Animatsiooni lisamiseks kasutati välist koodipakki, nimega *Ramotion animated-tab-bar*, mis oli vabalt kasutatav [13]. Vaadete vahetamisel navigatsiooniriba kaudu toimub vastava nupu animeerimine ning musta värvi ikoon värvub roheliseks. Sellega teavitatakse visuaalselt, millise vaate kasutaja valis. Rakenduses on läbivalt kasutatud kolme erinevat värvi: roheline, oranž ja punane. Need viitavad seene söögikõlblikkusele: roheline – söödav, oranž – sööda kupatatult või mittesöödav, punane – mürgine, looduskaitse all.

Rakenduse esmakordsel käivitamisel küsitakse luba kasutada asukohapõhiseid andmeid kaardivaate toimimiseks ning kaameravaate avanemisel küsitakse luba kaamera kasutamiseks seente tuvastamiseks.

Diagrammil (Joonis 11) on kujutatud andmebaasi protsess pärast rakenduse käivitamist. Klass nimega *AppDelegate* hoolitseb rakenduse käitumise eest käivitamisel ning samuti on mõeldud rakenduse kesksete andmestruktuuride deklareerimiseks [14]. Lokaalsest andmebaasist toimub seeneliikide andmete importimine ja objektidena lisamine massiividesse. Kuna seente andmed rakenduses ei muutu, on lihtsam kasutada korra deklareeritud massiive. Kokku on neli massiivi, mis on jaotatud söögikõlblikkuse

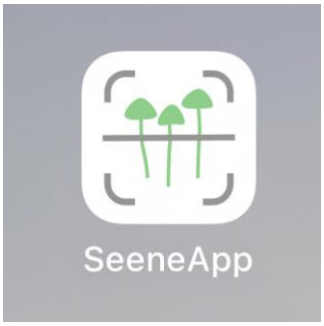
identifikaatori abil gruppidesse: 1 – söödavad, 2 – mittesöödavad või kupatatult söödavad, 3 – mürgised ja/või looduskaitse all olevad seened. Neljas massiiv sisaldab kõiki seeni.



Joonis 11. Andmebaasi importimisprotsess

3.2 Põhiikoon

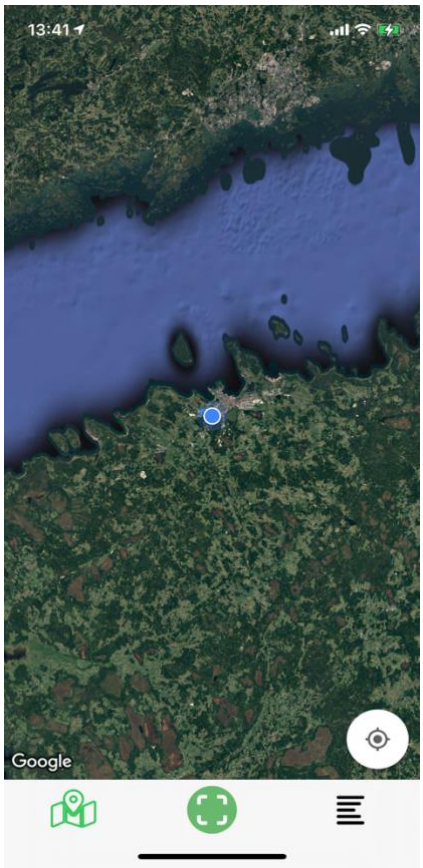
Valminud rakenduse põhiikoon (Joonis 12) sisaldab kolme rohelist seent, mida ümbritsevad nurkades raamid, mis on üheks objektituvastuse sümboliks. Keskel paikneb seeni läbiv joon. Ikooni elementide kooslus on põhimõttelt analoogne skanneri liikuva skaneerimisanduriga.



Joonis 12. Valminud rakenduse põhiikoon

3.3 Kaardivaade

Kaardivaade (Joonis 13) on üksiti ka esimene vaade, mis avaneb rakenduse käivitamisel. Seda saab kasutada ümbruskonna uurimiseks näiteks metsas. Kaart paikneb ekraanil maksimaalselt, mis annab kasutajale võimaluse suuremat ala korraga vaadelda. Üksiti saab ka positsioneerida paremal all nurgas oleva nupuga. Satelliidi kaardi versioon võimaldab paremini jälgida metsatukkasid ja lagendikke. Kaart baseerub *Google Maps'*i tarkvaraarenduskomplektil. Kasutajale tähendab seda, et kaarti uuendatakse automaatselt. *Google Maps*'i tarkvaraarenduskomplekti kasutamiseks peab registreerima rakenduse *Google Cloud Platform*'is ning vastu saab *API* võtme, mis võimaldab autentida vastavat rakendust [15]. Tarkvaraarenduskomplekt on integreeritud rakendusse *CocoaPods*'i abil, mis on tarkvarakomplektide ja pakkide haldamissüsteem [16].

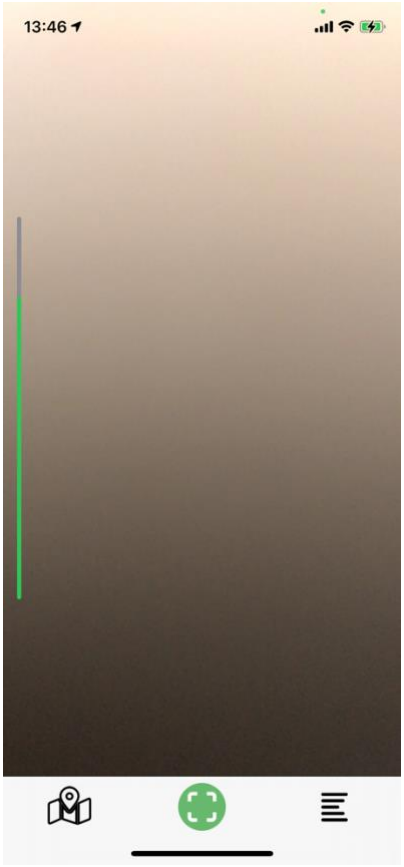


Joonis 13. Kaardivaade

3.4 Tuvastamisvaade

Tuvastamisvaatel (Joonis 14) paikneb vasakul servas indikaator, mis näitab tuvastamisprotsessi kulgu. Kaamerapilt on paigutatud ekraanile maksimaalselt. Sellega

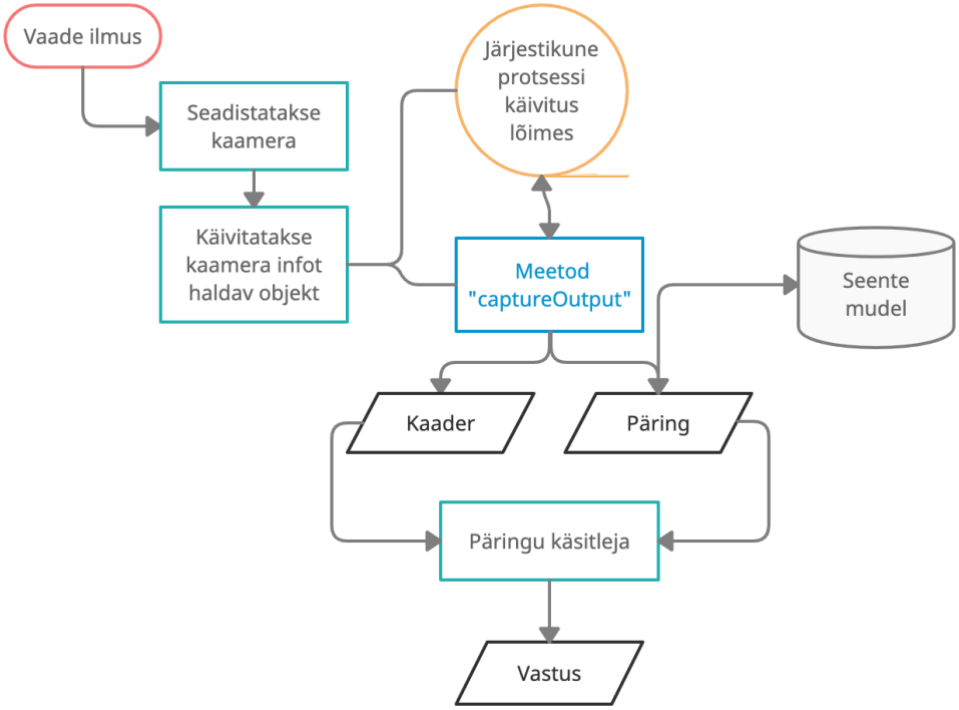
antakse kasutajale võimalus seent paremini jälgida telefoni ekraanilt. Vaate avanemisel hakkab pea koheselt tööle tuvastamisprotsess, mis on üksiti ka antud rakenduse põhifunktsionaalsus.



Joonis 14. Tuvastamisvaade

Diagrammil (Joonis 15) on kujutatud mudeli integreerimist ning tuvastamisprotsessi loogikat. Meetod nimega *captureOutput* (Lisa 3) kutsutakse välja peale iga uue kaadri saabumist, kasutab ühe parameetrina puhvrit, mis omakorda sisaldab kaadrit ning infot puhvri kohta. Puhvrit kasutatakse päringu käsitleja objektis (*VNImageRequestHandler*) parameetrina ning sellest objektist kutsutakse omakorda välja meetod *perform*, mis täidab etteantud päringu. Etteantud päring koosneb *VNCoreMLRequest* objektist, mille parameetriks on viide seeneliikide mudelile. Peale igat päringu vastust võetakse vastuses olevast massiivist esimesel kohal paiknev klassifikaatori nimetus, mis ühtib andmebaasis asuva seeneliigi eestikeelse nimetusega, ning lisatakse tuvastatud seente sõnastikku, mille võtmeks on seeneliigi nimetus ja väärtuseks tuvastamiste arv. Esimese kümne vastuse puhul ei toimu lisamist sõnastikku selleks, et kasutaja jõuaks kaamera seenele osutada. Sõnastikus olemasolevale seenele lisatakse igakordsel tuvastamisel üks ilmnemine juurde. Tuvastamine lõpetatakse kui kaadreid on kokku püütud 120. Tuvastamisprotsessi

indikaator on ühendatud kaadrite püügi arvuga. *iPhone X* puhul oli 120 kaadri püüdmise ajaks umbes 12 sekundit. Sõnastik sorteeritakse enim tuvastamisi kogunud seente järgi. Pärast seda võetakse kuni viis esimest seeneliigi nimetust sõnastikust, leitakse kõikide seente massiivist vastava klassifikaatori nimetusega seeneliik ning lisatakse objektidena massiivi, mida hiljem kasutatakse tuvastatud seene vaates.



Joonis 15. Mudeli integreerimine

3.5 Tuvastatud seene vaade

Tuvastatud seene vaates (Joonis 16) näidatakse tõenäosuse poolest kuni viis kõige sarnasemat seent otsitavale. Esimesel kohal on kõige tõenäolisem vaste. Seeneliigid paiknevad eelnevas vaates valminud massiivis. Kasutajale meenutatakse, et käesolev rakendus ei pruugi õigeid vasteid anda. Juhul kui elemente on rohkem kui üks, saab nende vahel kerida vasakule, paremale. Kerides paigutub element automaatselt ekraani keskele Elementides on välja toodud seene pilt, eesti- ja ladinakeelne nimetus, söögikõlblikkus ning tagasi tuvastamisvaatesse suunav nupp. Vaatele ilmuvate elementide taustavärvid viitavad visuaalselt seene söögikõlblikkusele ning vaate üldine taust on viimase kaamerapildi hägustatud jäädvustus.

Joonis 16. Tuvastatud seene vaade

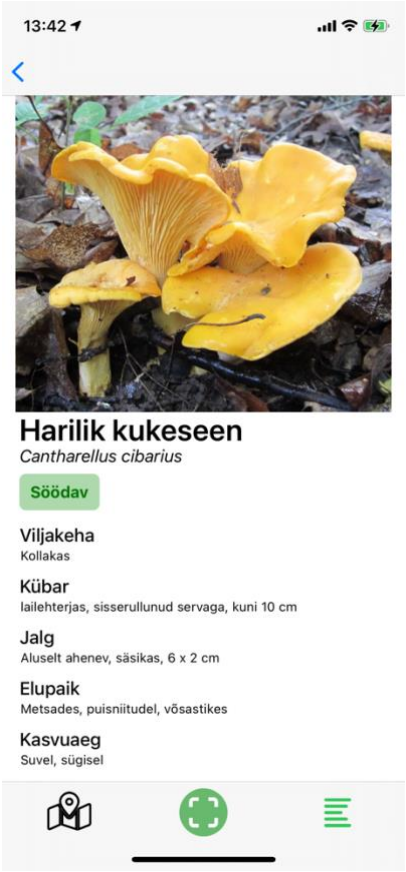
3.6 Nimekirjavaade

Nimekirjavaates (Joonis 17) asetseb üleval navigatsiooniriba, millega saab valida erinevate seente nimekirjade vahel, mis on jaotatud kolme kategooriasse: söödavad, kupatatult söödavad ja mittesöödavad ning mürgised ja/või looduskaitse all olevad seened. Iga, navigatsiooniribal paikneva, kategooria tekstivärv muutub vastavalt nimekirjas asuvate seente söögikõlblikkusele. Nimekirjad on keritavad. Igal real asuva seene kohta on välja toodud pilt, eesti- ja ladinakeelne nimetus. Klõpsates real asuvale seenele, suunatakse edasi klõpsatud seene detailvaatesse. Kategooriatele omaselt kuvatakse neile vastavate massiivide sisu.

Joonis 17. Nimekirjavaade

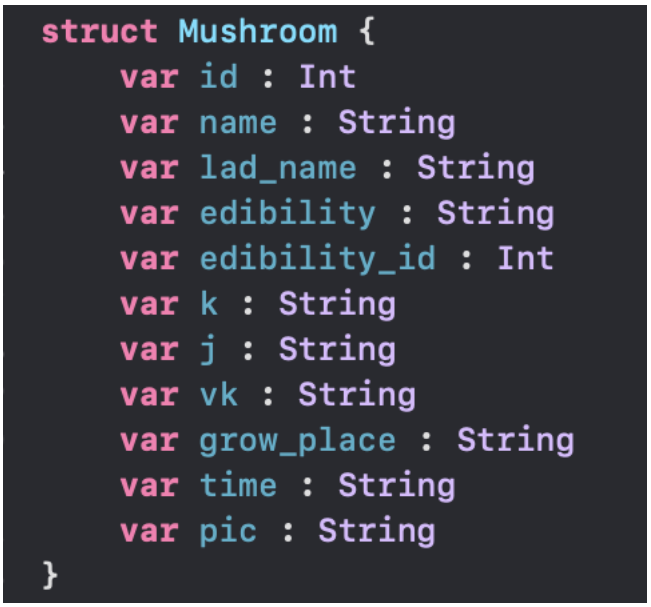
3.7 Seene detailvaade

Seene detailvaates (Joonis 18) saab kasutaja näha seente nimekirjas valitud vastava seene põhjalikku kirjeldus. Vaate üleval vasakus nurgas paikneb nupp, mis suunab tagasi nimekirjavaatesse. Informatsioon paikneb vaatel ülevalt alla ning selgelt. Esimesel kohal on seeneliigi pilt. Seejärel näeb seene eesti- ja ladinakeelset nimetust. Söögikõlblikkuse elemendi värvid on vastavalt söögikõlblikkuse kategooriatele. Seene kirjelduse osas näeb viite erinevat pealkirja: viljakeha, kübar, jalg, elupaik, kasvuaeg. Kõik pealkirjad on kasutajale märgatavamaks tehtud. Viljakeha pealkiri hõlmab endas seene üldist välimust, lõhna, maitset. Järgmised kaks pealkirja kirjeldavad seene kübara ja jala suurust, kuju ning muid iseärasusi. Elupaiga pealkirja all peetakse silmas seente kasvukohta looduses ning kasvuaja all seente valdavat kasvuperioodi.



Joonis 18. Seene detailvaade

Rakenduse taustal hoitakse seeni objektidena (Joonis 19) massiivides. Objektis sisalduvad seeneliigi andmed: unikaalne identifikaator, eesti- ja ladinakeelne nimetus, söögikõlblikkuse identifikaator ja söögikõlblikkuse informatsioon, lisaks informatsioon seeneliigi viljakeha, kübara, jala, kasvukoha ning kasvuaja kohta ja viide pildile. Pilte hoitakse kausta *Assets* alamkaustas *mush_pics*. *Assets* kaust genereeritakse, kui luuakse uus projekt *Xcode*'s. Tänu sellele kaustale on piltide haldamine lihtsamaks tehtud, see tähendab, et ei ole vaja lisa koodi kirjutada [17].



Joonis 19. Seene objekti struktuur

4 Rakenduse põhifunktsionaalsuse analüüs ja võimalikud täiendused

Käesolevas peatükis antakse ülevaade põhifunktsionaalsuse ehk seente tuvastamistäpsuse tulemustest. Arutletakse rakenduse võimalike edasiarenduste üle.

4.1 Tuvastamistulemused

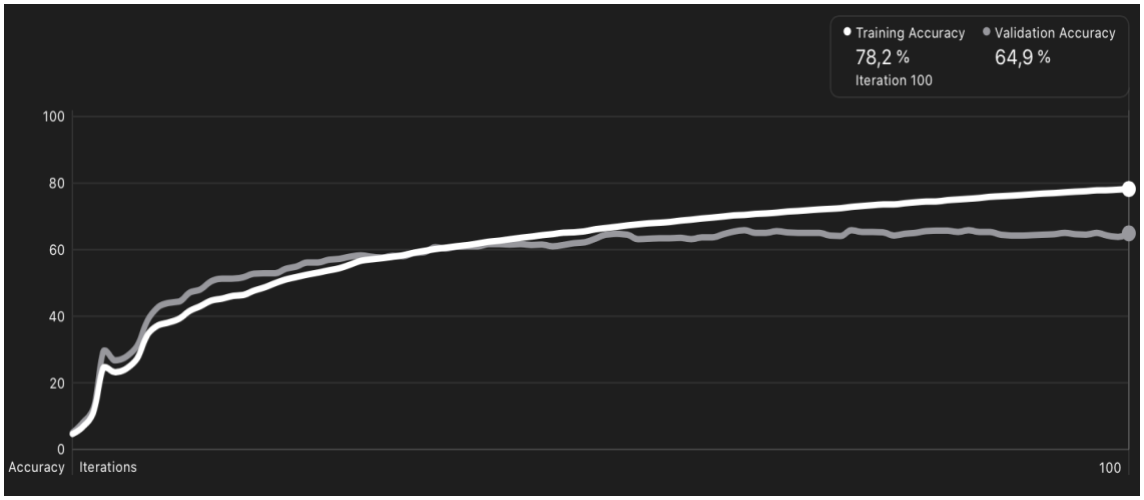
Mudeli treenimisel oli kokku 79 seeneliiki ning ligi 15000 pilti. Osadel seeneliikidel oli rohkem pilte, teistel vähem. Rakenduse tuvastamistäpsuse kontrolliks valiti juhuslikult 21 erinevat seent ning sooritati 5 mõõtmist seenestaja taskuraamatu piltide põhjal [10]. Mõõtmised sooritati valgusküllases keskkonnas ning pandi kirja tabelisse (Tabel 1). Tulemusi vaadeldi kahes kategoorias: kui tihti esineb esimese vastena ning kui tihti esineb esimese kolme vaste seas.

Tabel 1. Seeneliikide esinemiste arv tuvastamisel

Seeneliigi nimetus	**Esinemiste arv esimese vastena**	**Esinemiste arv esimese kolme vaste seas**
Harilik kukeseen	4	5
Harilik karikseen	3	5
Limatünnik	0	1
Panter-kärbseseen	5	5
Liudkogrits	0	2
Suur sirmik	5	5
Austerservik	0	2
Kuuseriisikas	0	0
Harilik kivipuravik	2	5
Murumuna	5	5
Kollane pilvik	5	5
Limavöödik	0	5
Metsšampinjon	1	4
Kollariisikas	0	2

Näsariisikas	0	1
Lilla ebaheinik	5	5
Tavariisikas	2	4
Kuusepilvik	4	5
Soomusheinik	5	5
Hiid-punalehik	0	1
Seepheinik	5	5
Kokku:	**51**	**77**

Treenitud mudeli valideerimisandmete graafikul (Joonis 20) näeb treenimise ja valideerimise täpsust. Seadistusteks olid määratud 100 iteratsiooni ja andmete suurendamise meetoditeks olid piltide peegeldumine ning pöörlemine.



Joonis 20. Treenitud mudeli tuvastamise graafik

4.2 Tuvastamistulemuste analüüs ja järeldused

Kokku esinesid valitud seeneliigid esimese vastena 51 korda maksimaalse 105 korra seast ning esimese kolme vaste seas 77 korda. Mõõtmiste ja treenitud mudeli graafiku põhjal võib järeldada, et tuvastamistõenäosus on vähemalt 60%. Oli ka olukordi, kus seeneliik paiknes esimese viie vaste seas, kuid neid tabelis ei kajastatud. Tuvastamistäpsuse tõstmiseks tuleks lisada rohkem pilte mudeli treenimiseks. Samuti võiksid pildid olla korrektsed, see tähendab, et pildil asetseb ainult üks või mitu seent, mitte muud segavad faktorid. Siinkohal tasuks mainida, et käsitsi tuvastamist kontrolliti raamatu piltide põhjal ning mudeli treenimisel eraldas programm automaatselt valideerimiseks pilte. Seetõttu

võib tegelik tuvastusprotsent hoopis kõrgem olla. Reaalse seene puhul on võimalik kaameraga erinevate nurkade alt skaneerida. Põhjalikemate järelduste tegemiseks peaks rakendust testima reaalsetel seentel.

4.3 Võimalikud edasiarendused

Rakenduse kasutamiseks kommertseesmärkidel peaks eelnevalt ühendust võtma *Mushroom Observer* veebilehega seotud isikutega ning seenestaja taskuraamatu [10] autoritega tingimuste kooskõlastamiseks.

Igal kasutajal võiks olla võimalus lisada rakendusse seeneliikide kasvukohad, ilma, et teised kasutajad seda näeksid. Seente kasvukohtade jagamise funktsioon teistega võiks olla valikuline. Lisaks satelliidikaardile võiks olla linnade, tänavate, maanteede nimede nähtavusfunktsioon valikuline.

Tuvastamisvaates on hetkel automaatne tuvastamisprotsessi algatus. Selle asendus võiks olla järgmine: kasutaja käivitab tuvastamisprotsessi nupuvajutusega, mis oleks mugavam. Peale seene tuvastust kuvatakse kasutajale võimalikud vasted. Vastetele klõpsates võiks avaneda võimalus näha seene detailset infot ja/või lisada oma lemmikute kogusse. Rakendusse võiks lisada rohkem seeneliike ning veelgi detailsemat infot. Ka nimekirjavaates võiks esineda otsing seeneliikide hõlpsamaks leidmiseks. Võhikule võiks olla võimalus tutvuda seente viljakehade ehitusega. Samuti viljakehade ehitusega kaasnevate mõistete selgitused. Mürgistuste kohta võiks olla info saadaval ning esmaabi nõuetele vastavalt käitumise korral õpetus.

Probleemide ja ettepanekute korral peaks olema võimalus ja/või informatsioon ühenduse võtmiseks rakenduse valmistajatega.

5 Kokkuvõte

Käesoleva lõputöö eesmärgiks oli luua mugav, eestikeelne ja Eestis kasvavate seente tuvastamise mobiilirakendus. Kuna autor polnud varasemalt kokku puutunud *iOS* arendusega ning sooviti selles valdkonnas end arendada, loodi rakendus just sellele operatsioonisüsteemile. Samuti polnud lõputöö kirjutamise ajal teist eestikeelset seente tuvastamise rakendust *iOS* platvormil saadaval.

Rakenduse põhifunktsionaalsuseks oli seente tuvastamise võimalus kaameraga reaalajas seent skaneerides. Lisaks on kasutajal võimalus positsioneerida ning jälgida oma asukohta kaardil. Samuti saab valida seeneliikide söögikõlblikkuse kategooriate vahel ning neile vastavalt seeneliikide põhjalikke kirjeldusi uurida. Rakenduses on läbivalt kasutatud informatsiooni edastamist värvidega, mis tähistatud järgmiselt: roheline – söödavad, oranž – mittesöödavad või kupatatult söödavad, punane – mürgised ja/või looduskaitse all olevad seened.

Rakenduse põhifunktsionaalsuse täpsuse selgitamiseks kasutati taskuraamatus olevate pilte ning treenitud mudeli enda andmeid. Selgus, et tuvastamise täpsus on vähemalt 60%. Põhjalikemate järelduste tegemiseks peaks kasutama rakendust reaalsel seenel. Tuvastamistäpsuse tõstmiseks peaks pilte rohkem koguma mudeli treenimiseks ning jälgima, et piltidel asuks ainult seen või seened.

Rakenduse täiendamisel võib arvestada järgmiste aspektidega: lisada rohkem seeneliike ja nende põhjalikemaid kirjeldusi, seente kasvukohtade määramine kaardil, probleemide ja/või ettepanekute korral ühenduse võtmise võimalus.

Kasutatud kirjandus

[1] C. M. Christensen, “Edible Mushrooms”, Minneapolis: University of Minnesota Press, 1977.

[2] “Jätkusuutliku toidusüsteemi loomine: Eli strateegia”, Euroopa Parlament. [Võrgumaterjal]. Saadaval: https://www.europarl.europa.eu/news/et/headlines/society/20200519STO79425/jatkusuutliku-toidussusteemi-loomine-eli-strateegia. [Kasutatud 20 Detsember 2020].

[3] A. P. Montes, E. Rangel-Vargas, J. M. Lorenzo, L. Romero, and E. M. Santos, “Edible mushrooms as a novel trend in the development of healthier meat products,” *Current Opinion in Food Science*, vol. 37, pp. 118–124, Feb. 2021.

[4] E. Alpaydin, “Machine learning: the new AI”, Cambridge, Massachusetts: MIT Press, 2016.

[5] “Eesti seenestik”, Eesti Entsüklopeedia. [Võrgumaterjal]. Saadaval: http://entsyklopeedia.ee/artikkel/eesti_seenestik. [Kasutatud 22 Detsember 2020].

[6] “The 4 Golden Rules of UI Design”, Adobe. [Võrgumaterjal]. Saadaval: https://xd.adobe.com/ideas/process/ui-design/4-golden-rules-ui-design/. [Kasutatud 24 Detsember 2020].

[7] “Introducing Xcode 12”, Apple. [Võrgumaterjal]. Saadaval: https://developer.apple.com/xcode/. [Kasutatud 24 Detsember 2020].

[8] “About Swift”, Swift. [Võrgumaterjal]. Saadaval: https://swift.org/about/. [Kasutatud 24 Detsember 2020].

[9] “What is SQLite?”, SQLite. [Võrgumaterjal]. Saadaval: https://www.sqlite.org/index.html. [Kasutatud 25 Detsember 2020].

[10] K. Kalamees, V. Liiv, “400 Eesti seent”, Tartu: Eesti Loodusfoto, 2005.

[11] “Create ML”, Apple. [Võrgumaterjal]. Saadaval: https://developer.apple.com/machine-learning/create-ml/. [Kasutatud 25 Detsember 2020].

[12] “Introduction”, Mushroom Observer. [Võrgumaterjal]. Saadaval: https://mushroomobserver.org/observer/intro. [Kasutatud 25 Detsember 2020].

[13] “Ramotion animated-tab-bar”, GitHub. [Võrgumaterjal]. Saadaval: https://github.com/Ramotion/animated-tab-bar. [Kasutatud 25 Detsember 2020].

[14] “UIApplicationDelegate”, Apple. [Võrgumaterjal]. Saadaval: https://developer.apple.com/documentation/uikit/uiapplicationdelegate. [Kasutatud 25 Detsember 2020].

[15] “Maps SDK for iOS”, Google. [Võrgumaterjal]. Saadaval: https://developers.google.com/maps/documentation/ios-sdk/overview. [Kasutatud 25 Detsember 2020].

[16] “What is CocoaPods?”, CocoaPods. [Võrgumaterjal]. Saadaval: https://cocoapods.org/. [Kasutatud 26 Detsember 2020].

[17] “Adding Images”, Apple. [Võrgumaterjal]. Saadaval: https://developer.apple.com/library/archive/documentation/ToolsLanguages/Conceptual/Xcode_Overview/AddingImages.html. [Kasutatud 26 Detsember 2020].

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Melvin Kriisa

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Eestikeelne mobiilirakendus iOS platvormil Eestis kasvavate seente tuvastamiseks“, mille juhendaja on Viljam Puusep
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

04.01.2021

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 – Python'is valminud kood seeneliikide linkide genereerimiseks tekstifaili

```python
import csv

#example url = "https://images.mushroomobserver.org/1280/2.jpg"
nameToFind = "Sarcosphaera coronaria"
species_folder = "Kroonliudik/"
mushTablesLocation = "/Users/mk/Desktop/mushTables/"
root_folder = "/Users/mk/Desktop/trainset/"
url = "https://images.mushroomobserver.org/"
pic_format = ".jpg"
txt_file_to_write = "seened"
mushroomFile = root_folder + species_folder + txt_file_to_write + ".txt"
picSize = "1280"
names_csv_file = "names.csv"
obs_csv_file = "observations.csv"
img_csv_file = "images_observations.csv"
obs_id = name_id = image_id = 0

def doCsv(fileLocation):
    fileNameCSV = fileLocation + names_csv_file
    with open(fileNameCSV) as namesCSV:
        readerNames = csv.reader(namesCSV, delimiter='\t')
        for row in readerNames:
            text_name = row[1]
            if nameToFind in text_name:
                name_id = row[0]
                obsDo(fileLocation, name_id)

def obsDo(fileLocation, name_id):
    observationsCSV = fileLocation + obs_csv_file
    with open(observationsCSV) as observationsCSVFile:
        readerObs = csv.reader(observationsCSVFile, delimiter='\t')
        for obsRow in readerObs:
            if name_id == obsRow[1]:
                obs_id = obsRow[0]
                mapDo(fileLocation, obs_id)
                # possibility = obsRow[7]

def mapDo(fileLocation, obs_id):
    txtFile = open(mushroomFile, "a")
    maptomapCSV = fileLocation + img_csv_file
    with open(maptomapCSV) as maptomapCSVFile:
        readerMapToMap = csv.reader(maptomapCSVFile, delimiter='\t')
        for mapRow in readerMapToMap:
            if obs_id == mapRow[1]:
                image_id = mapRow[0]
                result = url + picSize + "/" + image_id + pic_format + "\n"
                txtFile.write(result)

# doCsv(mushTablesLocation)
```

Lisa 3 – Mudeli integreerimise ja tuvastamise koodinäidis

```
func captureOutput(_ output: AVCaptureOutput, didOutput sampleBuffer:
    CMSampleBuffer, from connection: AVCaptureConnection) {
    count = count + 1

    guard let pixelBuffer: CVPixelBuffer =
        CMSampleBufferGetImageBuffer(sampleBuffer) else { return }

    guard let model = try? VNCoreMLModel(for: Mushrooms1(configuration:
        MLModelConfiguration()).model) else { return }

    let request = VNCoreMLRequest(model: model) { (finishedReq, err) in
        guard let results = finishedReq.results as? [VNClassificationObservation]
            else { return }
        guard let firstObserv = results.first else { return }

        if !self.skipCount.contains(self.count) {
            if let val = self.detectedMushrooms[firstObserv.identifier] {
                self.detectedMushrooms[firstObserv.identifier] = val + 1
            } else {
                self.detectedMushrooms[firstObserv.identifier] = 1
            }
        }
        self.updateProgressBar()
        if self.count == 120 {
            self.handleShowingResults()
        }
    }
    try? VNImageRequestHandler(cvPixelBuffer: pixelBuffer, options:
        [:]).perform([request])
}
```