Winter 12-2020

NDËRTIMI I NJË "TO-DO" APLIKACIONI ME TEKNOLOGJINË FLUTTER

Emnolina Brahimi
University for Business and Technology - UBT

Programi për Shkenca Kompjuterike dhe Inxhinierise

NDËRTIMI I NJË "TO-DO" APLIKACIONI ME TEKNOLOGJINË FLUTTER

Shkalla Bachelor

Emnolina Brahimi

Dhjetor / 2020
Prishtinë

Programi për Shkenca Kompjuterike dhe Inxhinierise

Punim Diplome
Viti akademik 2015-2016

Emnolina Brahimi

Ndërtimi i një “To-Do” Aplikacioni me teknologjinë flutter

Mentori: MSc.Lavdim Menxhiqi

Dhjetor / 2020

Ky punim është përpiluar dhe dorëzuar në përmbushjen e kërkesave të pjesshme për Shkallën Bachelor

ABSTRAKT

Aplikacionet për telefon mobil viteve të fundit kanë filluar të zëvendësojnë shumë veprimtari që në të shkuarën i kemi bërë në mënyrë primitive te themi. Nevoja për të kursyer sa më shumë kohë, energji dhe para ka bërë që të krijohen aplikacione të ndryshme, inovative dhe mbi të gjitha aplikacione që ofrojnë zgjidhje të problemeve.
Smartphone-ët por edhe shumë pajisje të tjera elektronike si tabletet dhe smartwatch janë bërë aq të nevojshme sa që pothuajse secili person i rritur mund të posedoj një të tillë.

Duke parë kërkesën dhe nevojën në tregun tonë për zgjidhje të problemeve përmes aplikacioneve, si fokus kryesor kemi marrë zhvillimin e një aplikacioni ‘’To Do List’’.
Ky aplikacion do të zëvendësoj bllokun e vjetër të shënimeve, ku mbaheshin të dhëna të rëndësishme për kryerjen e obligimeve ditore, javore, mujore apo edhe vjetore. Aplikacioni jonë do të jetë i qasshëm nga të gjithë që posedojnë një smartphone me sistem operativ Android dhe do të mund të instalohet nga dyqanet online për aplikacione. Si tipare kryesore aplikacioni do ta ketë mundësinë që të shtohen detyra/obligime në formë të një liste, të ndara në kategori, mundësi të vendosjes së datës, përfundimin si dhe fshirjen e një detyre/obligimi.

Fjalët Kyçe: *Aplikacione për telefon mobil, Sistemi Operativ Android*

MIRËNJOHJE/FALENDERIME

Falenderimi më i madh i takon Zotit të plotëfuqishëm që deri në këto momente pata dhe vazhdoj të gëzoj shëndet mendor dhe fizik për të arritur t'i dal mbanë cdo sfide e ndër më të rëndësishmet përfundimi i studimeve bachelor në Universitetin për Biznes dhe Teknologjisë(UBT), fakulteti për Inxhinieri dhe Shkenca Kompjuterike.

Arritja deri në këtë pikë nuk ishte e lehtë, por me ndihmën e palodhshme emocionale e financiare të familjes sime cdo sfidë u bë vullneti im për të vazhduar përpara, andaj sot dhe gjithë jetën u jam mirënjohse pa mbarim nënës, babait, bashkëshortit e vëllezërve.

Gjithashtu falënderime të sinqerta shpreh për shoqerinë dhe kolegët e mi të cilët kanë qenë gjithmonë të gatshëm për bashkëpunim dhe për diskutime rreth çdo problemi që kam hasur gjatë studimeve të mia.

Edukimi është vlerë e madhe e jetës, andaj shpreh falenderime dhe mirënjohje për gjithë stafin e departamentit të shkencave kompjuterike.
Në mënyrë të vecantë falenderoj Prof.MSc Lavdim Menxhiqi për kohën, durimin dhe mbështetjen profesionale gjatë punimit të temës së diplomës.

Faleminderit të gjithë atyre që në një mënyrë apo një tjetër u bënë pjesë të rrugëtimit tim gjatë gjithë kësaj kohe.

PËRMBAJTJA

LISTA E FIGURAVE IV
LISTA E TABELAVE IV
1. HYRJE 1
2. DEKLARIMI I PROBLEMIT 3
3. SHQYRTIMI I LITERATURËS 4
3.1 Zhvillimi i mobile aplikacioneve 5
3.1.1 Sistemet Operative për zhvillimin e mobile aplikacioneve 7
3.1.2 Gjuhët programuese dhe framework-ët për zhvillimin e mobile aplikacioneve 9
3.2 Gjuhët programuese të përdorura në aplikacionin tonë 11
3.2.1 Dart 11
3.2.2 Flutter 12
3.2.3 SQLite 14
3.2 Krahasimi i aplikacioneve si To Do List 15
4. METODOLGJIA 16
5. REZULTATET 17
5.1. Përshkrimi inxhinierik 17
5.1.1 Kërkesat funksionale 18
5.1.2 Kërkesat jo-funksionale 18
5.2. Teknologjitë 19
5.2.1 Material Design 19
5.2.2 Widgets 20
5.2.3 Gestures 21
5.3. Përshkrimi i moduleve të aplikacionit 23
5.4.1 Ndarja e tasqeve sipas kategorise 23
5.4.2 Shtimi i një tasku të ri 27
5.4.3 Shfaqja e listës së tasqeve 31
5.4.4 Caktimi i datës së taskut 32
5.4.5 Përfundimi i një tasku 34
5.4.6 Fshirja e një tasku 35
6. DISKUTIME DHE PËRFUNDIME 37
7. REFERENCAT 38

LISTA E FIGURAVE

Figura 1. Dart Logo 11
Figura 2. Flutter Logo 12
Figura 3.SQLite Logo 14
Figura 4. Krahasimi i aplikacioneve 15
Figura 5. Database table 17
Figura 6. Database Diagram 17
Figura 7. Kodi i veglave 20
Figura 8. Pjesë e kodit 21
Figura 9. Ikona e aplikacionit në ekran 23
Figura 10. Pjesë e kodit 25
Figura 11. Pjesë e kodit 26
Figura 12. Faqja e parë e aplikacionit 27
Figura 13. Dritarja për shtimin e një tasku 29
Figura 14. Dritarja e tasqeve të shkruara 31
Figura 15. Vendosja e datës së një tasku 33
Figura 16. Përfundimi i një tasku 34
Figura 17. Fshirja e një tasku 35

LISTA E TABELAVE

Tabela 1. Krahasimi i aplikacioneve 15

1. HYRJE

Jemi në vitin 2020, koha kur gjithqka është teknologji, interneti ka përfshi tashmë cdo sferë të jetës dhe është bërë stil i jetesës në këto vitet e fundit. Megjithëse në Kosovë industria e teknologjisë informative ende shihet si industri e re disa kompani kanë arritur të sjellin inovacion, të zhvillojnë zgjidhje interesante si softuerike ashtu edhe harduerike, duke ju përgjigjur kështu kërkesave të klientëve.

Përballja e vazhdueshme gjatë gjithë kohës me obligime të shumëta ka qenë gjithmonë, edhe në atë kohë njerëzit kanë ndjer nevojën që këto obligime që i kanë ti shkruajnë diku në mënyrë që të mos i harrojnë. Këto obligime mund të kenë qenë në baza ditore, javore, mujore apo edhe vjetore. Më parë njerëzit janë detyruar që këto obligime ti shkruajnë në një bllok shënimesh, apo në ndonjë letër që e kanë pasur afër, por kjo letër e ky bllok shënimesh shpesh ka ndodhur të jetë harruar apo edhe humbur diku, kështu duke përfunduar pa kryer obligimet që kanë qenë të shënuara dhe të mbesin prapa në punë, shkollim apo cfardo qoftë ajo ngjarje e cila është munguar vetëm pse ka humbur blloku i shënimeve. Pastaj të mbaje një bllok vetem që të shkruaje ngjarjet dhe obligimet që ke për të kryer në mesin e shumë gjërave të tjera që njeriu duhet të mbaj me vete, ishte një tjetër obligim më vete. Humbja e kohës dhe energjisë duke u munduar të mbajnë në mend dhe të mbushin blloqe të shumëta me shënime ishte një anë negative shumë e madhe edhe për kompanitë e vendit. Tani në korniza të zgjeruara mund të themi se blloku i shënimeve ekziston edhe sot, madje sot secili ka një bllok të shënimeve që e posedon me vete pothuajse 24/7.

Në vitin 1994 u zbulua smartphone-i i parë i cili ishte zanafilla e shumë zgjidhjeve kreative e të thjeshta që pasuan në vitet në vazhdim. Më vonë në vitet e 2000-ta smartphone-ët evoluan shumë sa i përket ndërfaqes së përdoruesit. Këta smartphone filluan të zhvillonin aplikacione të ndryshme që ishin zgjidhje e shumë problemeve, prurje të reja në jetën e njerëzve, e në mesin e këtyre aplikacioneve zuri vend edhe blloku i shënimeve në një variant modern.

Në vitin 2020 sipas faqës bankmycell, 3.5 Billion ose 44.69% e popullsisë së botës posedon një smartphone.

Në përgjithësi smartphone-ët janë pajisje elektronike të vogla të cilat mund të mbahen me vete gjatë gjithë kohës, dhe kështu po ndodh, secili që ka një smartphone e ka me vete gjatë gjithë kohës dhe përmes aplikacioneve të ndryshme kryen edhe punë të ndryshme. Kështu një bllok të shënimeve të thjeshtë si aplikacion disa smartphone e kanë madje të para-instaluar.

Ky bllok i shënimeve i evoluar në një listë të obligimeve mund të instalohet nga dyqanet online të aplikacioneve.

Si punim të përfundimit të studimeve, së bashku me mentorin kemi zgjedhur të zhvillojmë një bllok shënimesh në formë të një liste të obligimeve apo detyrave ditore, javore, mujore dhe vjetore. Ky aplikacion do të përmbajë ndarje të detyrave sipas kategorisë dhe nevojës së përdoruesit, vendosjen e datës për një detyre, shtimin, përfundimin dhe fshijrjen e një detyre. Secila kategori për tu dalluar nga tjetra do të jetë me ngjyrë të vecantë dhe do ta ketë ikonën përkatëse. Secili task që përfundohet do të dekorohet me një vijë të drejtë mbi te pa u larguar nga lista. Tasqet të cilat përdoruesi dëshiron ti fshij do të pyetet me anë të një alert dialogu se a është i sigurt për fshirjen e taskut, por edhe pse një task fshihet ai përsëri mund të gjendet në dritaren “Recently Deleted” dhe mund të rikthet prapa në listën e tasqeve.

Aplikacioni do të jetë në dispozicion për secilin që ka një smartphone dhe që përdor Android si sistem operativ. Aplikacioni do të mund të instalohet nga dyqani Play Store online pa pagesë.

2. DEKLARIMI I PROBLEMIT

Në vitet e fundit zgjidhjet për pothuajse cdo problem asnjëhere nuk kanë qenë më të lehta në kuptimin figurativ. Këto zgjidhje të shumëta i ofron avancimi i lartë i teknologjisë. Në kohët e më hershme zakonisht njerëzit janë munduar që obligimet e tyre ditore apo javore ti mbajnë në mend, disa i kanë shkruar obligimet e tyre në letër të bardhë, apo edhe kanë bartur me vete blloqe shkrimi. Të gjithë e dimë se të mbajturit mend e një obligimi edhe mund të bëhet por të mbajturit mend e shumë obligimeve e sidomos atyre javore, mujore apo edhe vjetore është pothuajse e pa mundur. Zgjidhje nuk ishte as mbajtja e shënimeve në letër të bardhë apo blloqe shkrimi pasi që ashtu si mendimet edhe letra e blloqet mund te harroheshin diku, apo edhe të humbisnin fare. Një telefon mobil smart sot është pjesë e pandashme e jetës për mbi 3 bilion e popullësisë së botës [1], shkurt është dicka që e ke me vete në cdo moment. Avancimi i lartë i teknologjisë ka mundësuar që përmes aplikacioneve në telefon mobil të zëvendësohen shumë aktivitete që janë bërë më parë të themi në mënyrë primitive. Në këtë mënyrë është ndjerë nevoja që edhe blloku i shënimeve të zëvendësohet me një aplikacion në telefon mobil.

3. SHQYRTIMI I LITERATURËS

Në këtë kapitull do të analizojmë dhe shqyrtojmë se cka janë aplikacionet për telefon mobil, rëndësia dhe nevoja e përdorimit të tyre viteve të fundit. Në kuadër të këtijë kapitulli gjithashtu do të shpjegojmë se si zhvillohet një aplikacion mobil, cilat janë platformat për sistemet operative që përdoren nga smartphone-ët, cilat janë gjuhët programuese dhe framwork-ët që na duhen për të zhvilluar një aplikacion mobil, si dhe në fund do të krahasojmë dy aplikacione të llojit të njejtë me aplikacionin tonë, të kuptojmë më qartë përparësitë dhe mangësitë e secilit aplikacion. Si hapa të parë të procesit të zhvillimit të aplikacionit To Do List kemi ndërmarr:

1. Strategjinë si fazë e parë e zhvillimit të aplikacioneve mobile, duke evoluar idenë për një aplikacion të suksesshëm.
2. Analizimi dhe Plani janë fazë ku idea e aplikacionit fillon të marrë formë dhe të kthehet në një projekt aktual.
3. UI/ UX Dizajni ku perfshihet pothuajse i gjithë suksesi i aplikacionit.
4. Zhvilli i aplikacionit faza ku fillon puna për sjelljen në jetë të aplikacionit
5. Testimi i aplikacionit ku merren parasysh të gjitha lëshimet që kanë mundur të ndodhin gjatë zhvillimit të komplet aplikacionit.

3.1 Zhvillimi i mobile aplikacioneve

Mobile app development (shq: Zhvillimi i aplikacioneve në telefon mobil) është një proces i cili zhvillon një aplikacion për telefon mobil. Këto aplikacione mund të jenë të para-instaluara gjatë fabrikimit të telefonit mobil apo edhe pastaj nga vetë përdoruresi. [2] Aplikacionet në telefon mobil zhvillohen nga programer të gjuhëve programuese apo framework-ëve për mobile. Si pjesë e rëndësishme e procesit të zhvillimit të një aplikacioni për telefon mobil është edhe pjesa e dizejnimit të aplikacionit e njohur si User Interface(UI) që në shqip mund ta shpjegojmë si pamje, ndërfaqe të asaj qka përdoruesi sheh. Mobile UI konsiderohen dritaret, kontekstet, hyrjet etj. Përdoruesi është fokusi i dizajnit të një aplikacioni, interface-i i përfshinë së bashku komponentët e hardware-it dhe software-it. Në industrinë e zhvillimit të aplikacioneve i kemi dy platforma dominuese. Njëra është iOS platformë nga kompania Apple. Platforma iOS është sistem operativ nga Apple që përdoret nga smartphone-ët iPhone. Platforma tjetër është Android nga Google. Sistemi operativ Android është një sistem operativ i përdorur jo vetëm nga Google por edhe nga shumë kompani të tjera për të ndërtuar smartphone apo pajisje të tjera smart. Megjithëse ka disa ngjashmëri midis këtyre dy platformave kur ndërtohen aplikacionet, zhvillimi për iOS kundrejt zhvillimit për Android përfshin përdorimin e kompleteve të ndryshme të zhvillimit të softuerit (SDK) dhe mjeteve të ndryshme të zhvillimit. Ndërsa Apple përdor iOS ekskluzivisht për pajisjet e veta, Google e vë Android në dispozicion të kompanive të tjera me kusht që ato plotësojnë kërkesa specifike siç janë përfshirja e disa aplikacioneve të Google në pajisjet që ato dërgojnë.

Ekzistojnë katër qasje kryesore të zhvillimit kur ndërtohen aplikacione telefon mobil:

- Native Mobile Applications
- Cross- Platform Native Mobile Applications
- Hybrid Mobile Applications
- Progressive Web Applications

Secila prej këtyre qasjeve për zhvillimin e aplikacioneve mobile ka grupin e vet të përparësive dhe disavantazheve. Kur zgjedhin qasjen e duhur të zhvillimit për projektet e tyre, zhvilluesit marrin në konsideratë përvojën e dëshiruar të përdoruesit, burimet e kompjuterit dhe tiparet vendase të kërkuara nga aplikacioni, buxhetin e zhvillimit, synimet kohore dhe burimet në dispozicion për të mirëmbajtur aplikacionin.

Në aplikacionin tonë ne kemi përdorur qasjen Native Mobile Application dhe si vegël kemi përdorur Android Studio software i zhvilluar nga Google.

Native Mobile Application do të thotë që aplikacionet shkruhen në gjuhë programuese dhe framwork që ofrohen nga direkt nga platformat dhe funksionojnë drejtpërdrejt në sistemin operativ të pajisjes siç janë iOS dhe Android.

Zhvillimi i aplikacioneve mobile po bëhet më kritik për shumë biznese me më shumë se 3 miliardë njerëz në të gjithë botën që përdorin smartphone, më shumë se 1.5 miliardë përdorin tableta që nga viti 2019. Përdoruesit, mesatarisht, kalojnë 90 përqind të kohës së tyre të lirë në aplikacione dhe ka më shumë se 700 milion shkarkime aplikacionesh nga dyqane online të ndryshme aplikacionesh. [3]

Zhvilluesit mund të ndërtojnë aplikacione për qindra miliona pajisje duke synuar të dyja këto platforma.

3.1.1 Sistemet Operative për zhvillimin e mobile aplikacioneve

Më lartë folëm pak për dy sistemet operative më të mëdha për zhvillimin e aplikacioneve për telefon mobil iOS dhe Android, tani po hyjmë në detaje për shpjegimin e tyre.

iOS është një sistem operativ i krijuar dhe zhvilluar nga Apple Inc. ekskluzivisht për harduerin e tij. Është sistemi operativ që fuqizon shumë nga pajisjet mobile të kompanisë, përfshirë iPhone dhe iPod Touch; termi gjithashtu përfshinte versionet që ekzekutohen në iPad derisa emri iPadOS u prezantua me versionin 13 në 2019. Është sistemi operativ më i instaluari numër dy në botë pas Android. Është baza për tre sisteme të tjera operative të bëra nga Apple: iPadOS, tvOS dhe watchOS. [4]

I zbuluar në vitin 2007 për iPhone-nin e gjeneratës së parë, iOS është zgjeruar që prej asaj kohe për të mbështetur pajisje të tjera të Apple si iPod Touch (Shtator 2007) dhe iPad (Janar 2010).

Që nga marsi 2018, App Store i Apple përmban më shumë se 2.1 milion aplikacione iOS, 1 milion prej të cilave janë origjinale për iPad. [5]

Këto aplikacione mobile janë shkarkuar në mënyrë kolektive më shumë se 130 miliardë herë. Versione të këtijë sistemi operativ lëshohen cdo vit. Versioni aktual është iOS 14, i cili u lëshua për publikun në 16 Shtator, 2020. [6]

ky version solli shumë ndryshime në ndërfaqen e përdoruesit, duke përfshirë aftësinë për të vendosur widget në ekranin kryesor, një ndërfaqe kompakte për Siri dhe thirrjet telefonike dhe aftësinë për të ndryshuar si shfletuesin e parazgjedhur të internetit ashtu edhe aplikacionet e postës elektronike(e-mail). iOS ofron siguri të lartë në harduerin dhe softuerin e tij. Më shumë se 2 milion aplikacione iOS janë në dispozicion për të shkarkuar në App Store Apple. [7]

Sa i përket sistemit operativ Android për dallim nga iOS, ky është një sistem open source bazuar në një version të modifikuar të bërthamës Linux dhe softuerëve të tjetër me open source, i krijuar kryesisht për pajisje me në ekran si smartphone dhe tabletë. Android është zhvilluar nga një konsorcium zhvilluesish i njohur si Open Handset Alliance dhe i sponsorizuar komercialisht nga Google. U zbulua në nëntor 2007, me pajisjen e parë komerciale Android të lëshuar në shtator 2008. Është softuer falas dhe me burim të hapur; kodi i tij burimor është i njohur si Android Open Source Project (AOSP), i cili është licencuar kryesisht nën licencën Apache. Sidoqoftë, shumica e pajisjeve Android dërgohen me programe shtesë të para-instaluar [8], më e rëndësishmja Shërbimet Google Mobile (GMS) [9]që përfshin aplikacione thelbësore të tilla si Google Chrome, platformën e shpërndarjes dixhitale Google Play dhe platformën e zhvillimit të shoqëruar të Shërbimeve Google Play. Rreth 70 për qind e telefonave inteligjentë Android përdorin ekosistemin

e Google; [10] ekosistemet dhe pirunët konkurrues Android përfshijnë Fire OS (të zhvilluar nga Amazon) ose LineageOS. Sidoqoftë emri dhe logo "Android" janë marka tregtare të Google që imponojnë standarde për të kufizuar pajisjet "e paçertifikuar" jashtë ekosistemit të tyre për të përdorur markën Android. Kodi burimor është përdorur për të zhvilluar variante të Android në një varg të pajisjeve të tjera, të tilla si konzollat e lojërave, kamerat dixhitale, media player-et portativ, PC-të dhe të tjerët, secila me një ndërfaqe përdorimi të specializuar. Disa derivate të mirënjohura përfshijnë Android TV për televizione dhe Wear OS për veshje, të dyja të zhvilluara nga Google. Paketat e softuerëve në Android, të cilët përdorin formatin APK, shpërndahen zakonisht nëpër dyqane të aplikacioneve si Google Play Store, Samsung Galaxy Store dhe Huawei AppGallery, ose platforma me burim të hapur si Aptoide ose F-Droid. Android ka qenë sistemi operativ më i shitur në të gjithë botën në smartphone që nga viti 2011 dhe në tabletë që nga viti 2013. Që nga maji i vitit 2017, ai ka mbi dy miliardë përdorues aktivë mujorë, baza më e madhe e instaluar e çdo sistemi operativ dhe që nga gushti i vitit 2020, Google Play Store përmban mbi 3 milion aplikacione. [11] Versioni aktual i qëndrueshëm është Android 11, i lëshuar më 8 shtator 2020.

3.1.2 Gjuhët programuese dhe framework-ët për zhvillimin e mobile aplikacioneve

Një gjuhë programuese i tregon kompjuterit se çfarë duhet të bëjë. Çdo gjuhë programuese përmban një sintaksë dhe një grup të veçantë rregullash, të cilat duhet të ndiqen sa herë që shkruhet kodi. Një framework(kornizë) softuer është e ndërtuar mbi një gjuhë programimi. Si për shembull Rails është një framework i gjuhës programuese Ruby, Django është një framework i gjuhës programuese Python, në rastin tonë, ne kemi zhvilluar aplikacionin përmes framework-ut Flutter të gjuhës programuese Dart.

Përmes kësaj gjuhe dhe framework-ut të saj të zhvilluar nga Google ne kemi zhvilluar aplikacionin tonë për sistemin operativ Android dhe kemi përdorur disa librari për sistemin operativ iOS.

Për të ndërtuar aplikacione për sistemin operativ Android kemi gjuhë dhe framework të ndryshëm, e që disa prej tyre po i shpjegojmë më poshtë.

Java ishte gjuha e paracaktuar për të shkruar aplikacione Android që kur platforma Android u prezantua në vitin 2008. Java është një gjuhë programimi e orientuar në objekte që u zhvillua fillimisht nga Sun Microsystems në 1995 (tani është në pronësi të Oracle). Ishte një gjuhë shumë e popullarizuar si gjuhë e pastër e orientuar në objekte (krahasuar me C ++) dhe u adaptua shpejt nga platforma Android. Java përpilohet në "bytecode" që interpretohet gjatë kohës së ekzekutimit nga Makina Virtuale Java (JVM) që ekzekutohet në sistem operativ. Kritikët e Java thonë se Java ka nevojë për shumë kod "boilerplate" për të bërë një detyrë të thjeshtë, dhe konceptet si përjashtimet janë të vështira për t'u kuptuar. Deri më tani, kjo është gjuha më e përdorur gjerësisht për zhvillimin e aplikacioneve Android. [12]

Një tjetër gjuhë programuese për zhvillimin e aplikacioneve për telefon mobil është edhe gjuha Kotlin. Në vitin 2017 Google njoftoi se do të mbështesë Kotlin si një gjuhë alternative të klasit të parë për programimin Android. Kotlin është i ndërveprueshëm me Java, dhe të gjitha libraritë Java mund të thirren nga Kotlin. Me fjalë të tjera, Kotlin është një formë më e rregullt e Java-s.

Ndër gjuhët e shumëta programuese për zhvillimin e aplikacioneve mobile, mes tyre ne kemi zgjedhur gjuhën programuese Dart me framework-un Flutter.

Megjithëse Dart për herë të parë u bë publik nga Google në 2011, 2017 ishte publikimi i parë i qëndrueshëm.

Një nga aftësitë unike të Flutter është që Flutter vjen me libraritë e saj widget UI që bazohet në modelin e Google Material si dhe widgetet UI të ngjashme me iOS. [12]

Dart në sintaksë duket si gjuha programuese C. Skedarët Dart të përdorur në aplikacionet Flutter përpilohen dhe paketohen në një skedar binar (.apk ose .ipa) dhe ngarkohen në dyqanet e aplikacioneve online. [13]

Versioni i parë i Flutter ishte i njohur si emri i koduar "Sky" dhe ekzekutohej në sistemin operativ Android. Ajo u zbulua në samitin e zhvilluesve të Dartit 2015, [6] me qëllimin e deklaruar për të qenë në gjendje të jepte vazhdimisht me 120 korniza për sekondë. [14]

Më 6 maj, 2020, u lëshuan Dart SDK në versionin 2.8 dhe Flutter në versionin 1.17.0, ku iu shtua mbështetja për Metal API, duke përmirësuar performancën në pajisjet iOS (afërsisht 50%).

Për të ndërtuar aplikacione për sistemin operativ iOS mund të përdorim gjuhë të ndryshme programuese dhe framework të ndryshëm si dhe duhet të keni një llogari zhvilluesi Apple dhe Xcode IDE në një kompjuter Mac. Ju nuk mund të ndërtoni dhe të korrigjoni në mënyrë efektive vetëm në një makinë Windows. Xcode vjen me të gjitha paketat e kërkuara të zhvillimit të Apple: SDK-të, një redaktues kodi, mjetet e përpilimit / ndërtimit, simuluesit dhe një korrigjues. Në vazhdim po i marim disa shembuj të gjuhëve programuese dhe framework-ëve për zhvillimin e mobile aplikacioneve. [12]

Objektivi-C ishte gjuha e parë e mbështetur nga Apple për zhvillimin e aplikacioneve mobile iOS. Është një gjuhë e orientuar në objekte (OO) që e nxjerr sintaksën e gjuhës nga C dhe orientium në objekte nga SmallTalk. Një nga kritikat e zakonshme të gjuhës është se sintaksa e saj duket e ngathët si e folur dhe kllapat katrore janë të vështira për tu korrigjuar. Është një gjuhë e qëndrueshme dhe e pjekur, megjithëse ka pasur disa vite përdorim të gjerë.

Që kur Apple prezantoi Swift, popullariteti i Objektivit-C ka rënë ndjeshëm për zhvillimin e ri të celularëve iOS.

Një tjetër gjuhë programuese për zhvillimin e aplikacioneve për telefon mobil është edhe gjuha Swift. Apple prezantoi Swift në 2014 si një specifikim gjuhësor dhe e bëri të disponueshëm për zhvillimin e aplikacioneve mobile në Xcode në 2015. Pas një riparimi të madh të gjuhës Swift në 2016 (v 3.0), Swift ka tejkaluar Objective-C si gjuhë. Megjithëse Swift dhe Objective-C mund të bashkëjetojnë, domethënë libraritë e shkruara në Objective-C mund të përdoren në Swift, Apple po e bën shumë të qartë që Swift është zgjedhja e re e paracaktuar për zhvillimin e aplikacioneve iOS. Swift është një gjuhë më e lehtë, më e thjeshtë dhe më kompakte krahasuar me Objective-C. Zhvilluesit e Objektivit-C nuk kanë ndonjë problem për të kaluar në Swift.

Në rastin tonë për sistemin operativ iOS ne kemi përdorur vetëm disa librari të gjuhës programuese Dart me framework-un Flutter. [12]

3.2 Gjuhët programuese të përdorura në aplikacionin tonë

Në vijim do t'i përshkruajmë gjuhët programuese me të cilat kemi vendosur që ta zhvillojmë aplikacionin tonë për listën e shënimeve 'To-Do list'.

3.2.1 Dart

Figura 1. Dart Logo

Dart është një gjuhë programuese për aplikacione në shumë platforma. Është zhvilluar nga Google për herë të parë në vitin 2011 dhe përdoret për të ndërtuar mobile, desktop, server dhe web aplikacione. Dart është i bazuar në objekte, klasë, garbage-collected dhe ka sintaksë të stilit të gjuhës programuese C. Dart mund të kompajlohet me kodin e vet ose edhe të Javascript. Dart është i standartizuar me ECMA (European Computer Manufacturers Association). Dart është i ngjashëm me gjuhët Swift, C# dhe Java. Dart është open source dhe mund të përdoret nga të gjithë pa pagesë. Ajo që e bënë Dart të dalloj nga gjuhët tjera programuese është se Dart ka një sintaksë që është mjaftë e lehte për tu lexuar dhe mësuar dhe i përafërohet shumë gjuhës së njerëzve.
Dart u zbulua në konferencën GOTO në Aarhus, Danimarkë, 10-12 tetor 2011. [15] rojekti u themelua nga Lars Bak dhe Kasper Lund. [16] Dart 1.0 u lancua në 14 nëntor 2013.
Aplikacionet e Dart "transpilohen" në JavaScript dhe ato ekzekutohen natyrshëm në shfletuesit e aktivizuar me Dart. Me librari kualitative dhe veglat të ndryshme, Dart operon si në klient ashtu edhe në server për një proces zhvillimi të qëndrueshëm.
Ekzistojnë katër mënyra për të ekzekutuar kodin Dart: Një shembull i thjeshtë në Dart:

- Compiled as JavaScript- I përpiluar si JavaScript, kodi Dart është i njohur nga të gjithë shfletuesit kryesorë, pa pasur nevojë që shfletuesit të adoptojnë Dart.
- Stand-alone- Pakoja e zhvillimit të softuerit Dart (SDK) dërgohet me një Dart VM të pavarur, duke lejuar që kodi Dart të ekzekutohet në një mjedis të ndërfaqes së linjës komanduese.
- Ahead-of-time compiled- Kodi i Dart mund të përpilohet AOT në kodin e makinës (grupe udhëzimesh native).

- Native- Dart 2.6 me dart2native kompailerin për të përpiluar në kodin ekzekutiv të vetë-përmbajtur.

Në vazhdim një shembull i Hello World në gjuhën programuese Dart:

```
void main() {
  print('Hello, World!');
}
```

3.2.2 Flutter

Figura 2. Flutter Logo

Flutter është një mjet teknilogjik i ndërtuar nga Google i cili perdoret për të ndërtuar aplikacione për mobile(telefon), web(shfletues) dhe desktop(dritare) me një kod te vetëm. Flutter së pari u lansua në vitin 2017.

Flutter ka dy pjesë kryesore:

- SDK(Software Development Kit) një koleksion i veglave që na ndihmon të zhvillojmë aplikacione. Kjo përfshin veglat për të kompajluar kodin tuaj në kod te makinës (iOS dhe Android).
- Një framework(UI Library e bazuar në widgets) një koleksion i elementeve që mund të ripërdoren dhe të personalizohen sipas dëshirës si buttons, text inputs, sliders etj.

Disa aplikacione që janë zhvilluar me Flutter janë: Google Ads, AliBaba Group, BMW, Groupon, eBay, Philips etj.

Framework-u Flutter përmban dy grupe widget-ash që përputhen me gjuhët specifike të dizajnit: Miniaplikacionet Material Design implementojnë gjuhën e dizajnit të Google me të njëjtin emër, dhe miniaplikacionet Cupertino zbatojnë udhëzimet e ndërfaqes njerëzore iOS të Apple. Libraria e Fondacionit, e shkruar në Dart, ofron klasa dhe funksione themelore që përdoren për të ndërtuar aplikacione duke përdorur Flutter, të tilla si API për të komunikuar me motorin etj. Flutter përdor

një larmi widget-ash për të ofruar një aplikacion plotësisht funksional. Këto widget janë arkitektura framework-ut e Flutter.

Në vazhdim një shembull i Hello World programi me Flutter:

```
import 'package:flutter/material.dart';

void main() => runApp(HelloWorldApp());

class HelloWorldApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {

    //MaterialApp acts as a wrapper to the app and
    //provides many features like title, home, theme etc
    return MaterialApp(
      title: 'Hello World App',

      //Scaffold acts as a binder that binds the appBar,
      //bottom nav bar and other UI components at their places
      home: Scaffold(

        //AppBar() widget automatically creates a material app bar
        appBar: AppBar(
          title: Text('Hello World App'),
        ),

        //Center widget aligns the child in center
        body: Center(
          child: Text('Hello World'),
        ),
      ),
    );
  }
}
```

3.2.3 SQLite

Figura 3.SQLite Logo

SQLite është një librari e gjuhës C e cila implementon një bazë të të dhënave të vogël, të shpejtë, të pavarur, me besueshmëri të lartë me tipare të plota. SQLite është baza e të dhënave më e përdorura në botë dhe është e integruar në të gjithë telefonat celularë dhe shumicën e kompjuterëve dhe futet në pako brenda aplikacioneve të tjera të panumërta që njerëzit përdorin çdo ditë.

Kodi për SQLite është në domenin publik dhe kështu është falas për përdorim për çdo qëllim, komercial ose privat. Ndryshe nga shumica e bazave të të dhënave të tjera SQL, SQLite nuk ka një proces të veçantë të serverit. SQLite lexon dhe shkruan direkt në skedarët e zakonshëm të diskut. SQLite është testuar me shumë kujdes para çdo lëshimi dhe ka një reputacion për të qenë shumë i besueshëm. Shumica e kodit burimor SQLite i kushtohet thjesht testimit dhe verifikimit.

D. Richard Hipp projektoi SQLite në pranverën e vitit 2000 ndërsa punonte për General Dynamics në kontratë me Marinën e Shteteve të Bashkuara. Hipp po projektonte një softuer të përdorur për një sistem të kontrollit të dëmtimit brenda shkatërruesve të raketave të drejtuara, të cilat fillimisht përdorën HP-UX me një bazë të dhënash IBM Informix. SQLite filloi si një shtesë e Tcl. Qëllimet e dizajnit të SQLite ishin të lejonin që programi të operohej pa instaluar një sistem të menaxhimit të bazës së të dhënave ose duke kërkuar një administrator të bazës së të dhënave. Hipp e bazoi sintaksën dhe semantikën në ato të PostgreSQL 6.5. SQLite është një nga katër formatet e rekomanduara për ruajtjen afatgjatë të grupeve të të dhënave të miratuara për përdorim nga Libraria e Kongresit.

3.2 Krahasimi i aplikacioneve si To Do List

Pas analizimit të sistemeve operative, gjuhëve programuese dhe framework-ëve në tabelen me numër 1 do t'i krahasojmë dy aplikacionet më të përdorura me aplikacionin tonë.

Emërtimi	**Microsoft To-Do**	**Todoist**	**To Do List**
Gjuha programuese	HTML/JS/CSS	Python	Dart me Flutter
Platforma	Android iOS	Android iOS Windows Desktop	Android iOS
Cmimi	Falas	Me pagesë	Falas

Tabela 1. Krahasimi i aplikacioneve

Microsoft To-Do është një aplikacion i zhvilluar nga ekipi i Wunderlist, është cloud-based dhe integron Office 365. Për dallim nga aplikacioni jonë ky aplikacion lejon të krijoni taska apo ti importoni ato nga Wunderlist dhe Outlook.

Todoist u rrit në popullaritet me kalimin e viteve dhe ka evoluar në mënyrë të konsiderueshme duke përfshirë tipare më të fuqishme. Ky aplikacion është më i sofistikuar dhe nga i cili kemi mar ide të cilat i kemi trajtuar dhe kombinuar në aplikacionin tonë.

4. METODOLGJIA

Në mënyrë që të kuptojmë dhe të informohemi më shumë rreth kësaj teme dhe teknologjive të tyre kemi hulumtuar në materiale, artikuj shkencor dhe libra të autorëve të ndryshëm, ku në metodologjinë tonë kemi përfshirë disa metoda si:

Përmes shfletimit të literaturës kemi analizuar aplikacione të ngjajshme që përdoren në botë, teknologjinë që këto aplikacione përdorin dhe kemi kuptuar mendimin e zhvilluesve të këtyre aplikacioneve.

Analizimi i të dhënave – Kemi hulumtuar materiale të ndryshme rreth funksionimit të aplikacioneve tjera të ngjashme me aplikacionin që ne do e zhvillojmë, kemi analizuar gjithashtu se si këto aplikacione u sjellin benefite dhe ju ndihmojnë përdoruesve në cdo kohë dhe për cdo lloj obligimi që kanë.

Krahasimi i aplikacioneve – metodë kjo që është përdorur për të kuptuar përparsitë dhe mangësitë e aplikacioneve që kemi analizuar më lartë.

Zhvillimi i sistemit – Pasi kemi anlaizuar aplikacionet e ngjajshme dhe funksionet e tyre më të shpeshta që kanë, jemi munduar të zhvillojmë një sistem i cili i përshtatet kërkesave për tregun tonë.

5. REZULTATET

Në vazhdim do të shpejgojmë pikat se si kemi arritur deri tek rezultati i zhvillimit të aplikacionit “To-Do list”.

5.1. Përshkrimi inxhinierik

To-Do list është një aplikacion i cili i mundëson përdoruesve që të mbajnë një plan ditor, javor, mujor ose vjetor të obligimeve që kanë. Ky aplikacion është i bazuar në teknologjinë Flutter e cila është një vegël e krijuar nga Google që përdor gjuhën programuese Dart dhe përdor SQLite si bazë e të dhënave.

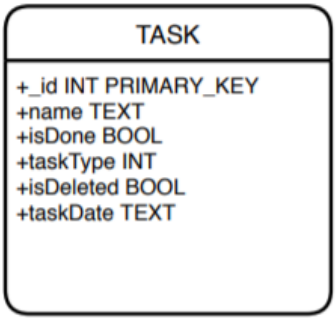

Figura 4. Database table

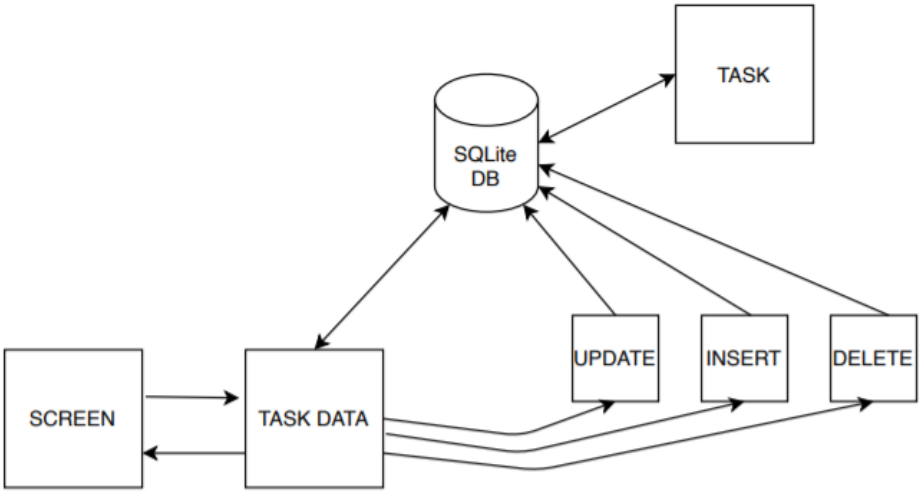

Figura 5. Database Diagram

5.1.1 Kërkesat funksionale

- Aplikacioni duhet të ndaj tasqet në kategori
- Aplikacioni duhet të ketë mundësinë ti caktoj datë taskut
- Aplikacioni duhet të lejoj mundësinë e vendosjes së datës edhe më larg se një vit
- Aplikacioni duhet të pyes para se të fshij një task
- Aplikacioni duhet të mundësoj shfaqjen e listës së tasqeve
- Aplikacioni duhet ti mundësoj përdoruesit që ti shoh tasqet e fshira
- Aplikacioni duhet ti numëroj tasqet e krijuara
- Aplikacioni duhet të mundësoj perfundimin e një tasku

5.1.2 Kërkesat jo-funksionale

- Aplikacioni duhet të jetë i bazuar në teknologjinë mobile
- Aplikacioni duhet të ofrojë përkrahje 24/7.
- Aplikacioni duhet të jetë i qasshëm nga çdo telefon mobil që ka sistemin operativ Android
- Aplikacioni duhet të ketë butona me ikonat përkatëse
- Aplikacioni duhet të ketë Checbox për përfundimin e një tasku

5.2. Teknologjitë

Në këtë kapitull do t'i përshkruajmë dhe analizojmë teknologjitë që kemi përdorur për zhvillim të aplikacionit. Poashtu do të përshkruajmë edhe gjuhët programuese që janë përdorur për zhvillimin e aplikacionit.

5.2.1 Material Design

Material Design është një librari në flutter e zhvilluar nga Google në vitin 2014. Importohet përmes linkut: import package:flutter/material.dart në krye të kodit. Material Design cilësohet si një arritje shumë e suksesshme në dizajn sa i përket eksperiencës së përdoruesit.

Disa nga widgets që përfshihen në librarinë Material janë AlertDialog, AppBar, Checkbox, Flatt Buttons, Text Fileds, DateTime Pickers etj..., të cilat i kemi përdorur në aplikacionin tonë.

Disa nga klasat dhe metodat më të rëndësishme në Flutter po i shpjegojmë më poshtë.

MaterialApp është një klasë e paradefinuar në Flutter. Është pothuajse komponenti kryesor në gjithe aplikacionin. Ne mund të ju qasemi të gjitha komponentave dhe veglave(widgets) të tjera të dhëna nga Flutter përmes klasës MaterialApp. Vegla(widget-i) MaterialApp krijon një numër veglash të dobishme në krye të aplikacionit, duke përfshirë këtu edhe një Navigator(drejtues) i cili menaxhon widgets e tjera të identifikuara si stringje, e që ndryshe quhen edhe "routes"(rrugë). Navigator përdoret për të shfletuar në ekrane të ndryshe ne aplikacionin tuaj.

MyAppBar është poashtu një widget i cili krijon një container(kuti) me një lartësi prej 56 pixel (device-independent) dhe një padding(mbushje) të mbrendshëm prej 8 pixel-eve në të dyja anët djathtas dhe majtas. Brenda në container widget-i MyAppBar përdor një rresht për të organizuar femijët e tijë(children).

MyScaffold është një widget që organizon fëmijet e saj në një kolonë vertikale. Në maje të kolonës paraqet një instancë të MyAppBar duke i dërguar appbar-it një text widget për ta perdorur si titull. Në figurën 2 shohim një pjesë të kodit që paraqet më së miri veglat që i spjeguam më lartë.

```dart
import 'package:flutter/material.dart';

main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(context) {
    return MaterialApp(
      home: Scaffold(
        body: Center(
          child: Text('Hello, Flutter'),
        ),
      ),
    );
  }
}
```

Figura 6. Kodi i veglave

5.2.2 Widgets

Në Flutter cdo gjë është një widget(vegël). Widgets janë blloqe ndërtuese të aplikacionit. Cdo element në ekranin e një flutter aplikacioni është një widget. Widgets përshkruajnë se si do të duket pamja e tyre duke pasur parasyshë konfigurimin dhe gjendjen e tyre aktuale. Kemi shumë lloje të widgets, ato më baziket janë: text, column, row, stack, container e shumë të tjera.

Disa lloje të widgets që ne i kemi përdorur në aplikacionin tonë po i shpjegojmë më poshtë.

StatefulWidget është një widget gjendja e të cilit mund të ndryshoj pasi që ndërtohet. Gjendjet e një statefulwidget mund të ndryshojne disa her gjatë jetëgjatesisë së tyre. Kjo thjesht do të thotë se gjendja e një aplikacioni mund të ndryshoj shumë-herë me përdorimin e ndryshëm të variablave, inputeve, te dhënave.

Ndërsa një satelesswidget është një widget gjendja e te cilit nuk mund te ndryshoj pasi të ndërtohet.

Në figurën 7 shohim se si kemi përdorur Card Widget-in në kod në aplikacionin tonë. Card është një panel me kënde pak të rrumbullakosura dhe një hijezim prapa si te lartësuar. Një card mund të përdoret për të shfaqur informacione si një lokacion gjeografik apo një fotoalbum apo detajet e një kontakti etj. Brenda në Card kemi përdorur një tjetër widget i cili është

GridView widget që paraqet elementet në formë të një tabele. Gridview paraqet elementet në dimensionin 2-D.

```dart
import 'package:todoey_flutter/screens/all.dart';

class Profile extends StatefulWidget {
  @override
  _ProfileState createState() => _ProfileState();
}

class _ProfileState extends State<Profile> {
  @override
  Widget build(BuildContext context) {
    myCard(iconcode, route, icontitle, colour) {
      return Card(...); // Card
    }

    return Scaffold(
        backgroundColor: Color(0xfff6f6f7),
      appBar: AppBar(...), // AppBar
      body: Column(
        children: <Widget>[
        Expanded(
          child: GridView.count(...), // GridView.count
        ), // Expanded
        SizedBox(),
        Container(
            padding: const EdgeInsets.all(10.0),
            width: 420.0,
          child: Column(...), // Column
        ) // Container
      ], // <Widget>[]
    )); // Column, Scaffold
  }
}
```

Figura 7. Pjesë e kodit

5.2.3 Gestures

Është një librari në flutter që detekton gjestet dhe importohet në krye të kodit me linkun: import package:flutter/gestures.dart. Ky widget përpiqet që ti njeh gjestet dhe tu përgjigjet thirjeve non-null. Në aplikacionin tonë kemi përdorur disa her widgetin gesture detector. Poashtu kemi përdorur edhe klasën InkWell që është një hapësirë drejtkëndore dhe që i përgjigjet shtypjes në ekran e cila implementon këtë efekt dhe na jep mundësinë që të dekorojmë butonin në prekje me metodot si splashcolor që kur shtypim ne atë pjesë shpërndan ngjyrë, të cilën metodë e kemi

përdorur në aplikacionin tonë. InkWell poashtu mund të përdoret në vend të gesturedetector. Më poshtë shohim pjesë të kodit ku kemi përdorur gesture detector dhe InkWell.

```
GestureDetector(
  onTap: (){
    Navigator.push(context, MaterialPageRoute(builder: (context) =>
Profile()),
    );
  },
```

dhe,

```
child: Ink(
  color: Colors.white,
  child: InkWell(
    splashColor: Colors.grey,
    onTap: () {
      Navigator.push(
        context,
        MaterialPageRoute(builder: (context) => route),
      );
    },
```

5.3. Përshkrimi i moduleve të aplikacionit

Më poshtë do të pershkruajmë hap pas hapi ndërfaqen e përdoruesit (user interface). Ndërfaqja e përdoruesit është bërë shumë e thjeshtë që i përshtatet secilës moshë të përdoruesve, duke filluar nga nxënësit e deri tek të moshuarit. Aplikacioni jonë është një bllok shënimesh modern ku secili do të ketë mundësinë të ruaj obligimet apo detyrat që kanë. Në figurën më poshtë kemi paraqitur një pamje se si duket aplikacioni jonë në mesin e aplikacioneve të tjera në një smart phone. Aplikacioni ynë ka të vendosur një ikonë që paraqet qartë një listë të shënimeve, kjo për ti'a bërë secilit përdorues më të thjesht, më të lehtë dhe më të shpejtë gjetjen e aplikacionit. Aplikacioni mund të instalohet nga dycanet online si Play Store dhe është pa pagesë.

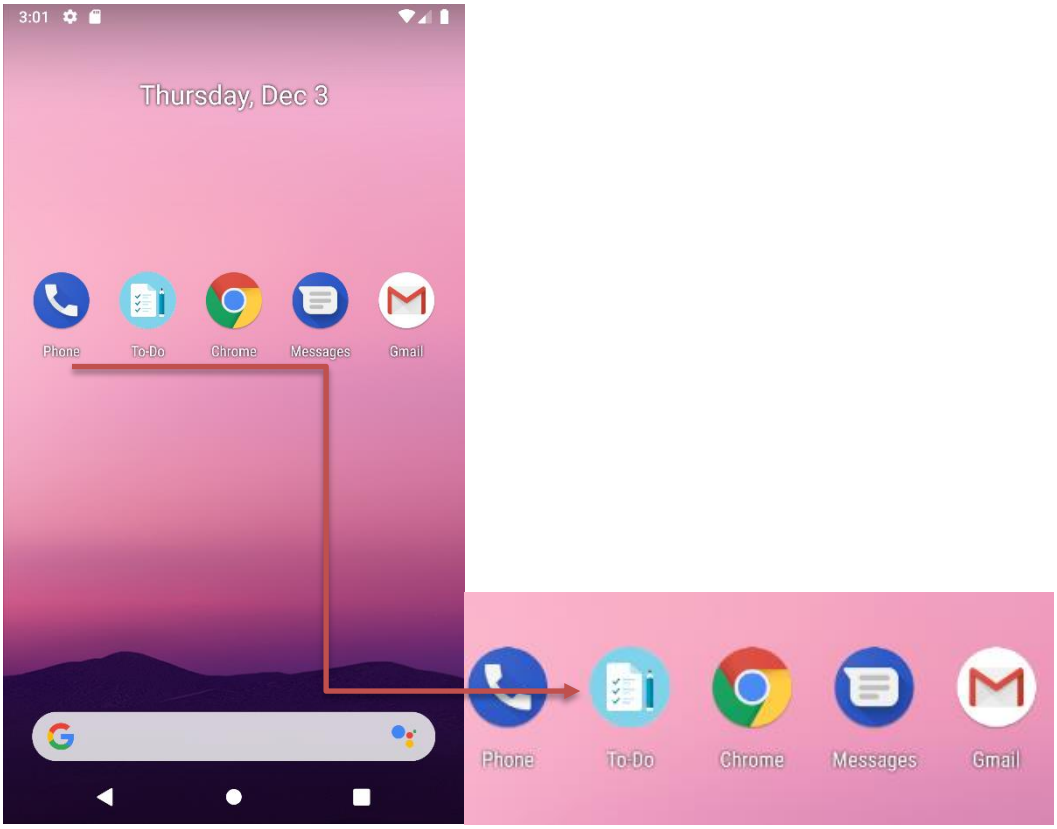

Figura 8. Ikona e aplikacionit në ekran

5.4.1 Ndarja e tasqeve sipas kategorise

Dritarja e parë qe i hapet përdoruesit kur klikon në ikonën e aplikacionit (Figura 10) To-Do paraqet një listë lëvizëse të ikonave të ndara në kategori sipas llojit të taskut(detyrës) brenda klasës 'Profile' e cila është e trashëguar nga klasa StatefulWidget. Kjo klasë na jep mundësinë që të bëjmë ndryshime pasi që të ndërtohet. Në listën lëvizëse secila kategori ka edhe ikonën përkatëse dhe

secila ikonë është ka nëntitullin dhe dekorohet kur të shtypet(klikohet) nga përdoruesit. Siq shihet edhe në kodin më poshtë lista e ikonave është krijuar me ndihmën e Card e cila është një flutter widget që si parametra pret iconcode-numrin e iconës te marr nga paketa e ikonave flutter, route-rrugën se ku do të na dergoj kjo ikonë, icontitle- titullin e ikonës dhe colour- ngjyrën e ikonës. Mbrenda në Card widget kemi përdorur Grid view widget e cila i paraqet ikonat në listë 2D lëvizëse. Më poshtë shihet vetem një pjesë e shkurt jo e plotë e kodit se si e kemi përdorur card widget mbrenda ne Grid view widget duke i thirur parametrat e krijuar në klasën 'Profile':

```
GridView.count(
  shrinkWrap: true,
  primary: true,
  padding: const EdgeInsets.all(20.0),
  crossAxisSpacing: 10.0,
  crossAxisCount: 2,
  children: <Widget>[
    myCard(
      58902,
      All(),
      'All',
      ALLColor,
    ),
```

Ndërsa në Figurën 8 shihet pjesë e madhe e kodit që kemi përdorur per ti bërë ikonat të klikueshme, dhe me metodën onTap:

```
onTap: () {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => route),);
},
```

tregojmë rrugën se ku duhet të na dërgoj ikona e caktuar.

```dart
import 'package:todoey_flutter/screens/all.dart';

class Profile extends StatefulWidget {
  @override
  _ProfileState createState() => _ProfileState();
}

class _ProfileState extends State<Profile> {
  @override
  Widget build(BuildContext context) {
    myCard(iconcode, route, icontitle, colour) {
      return Card(...); // Card
    }

    return Scaffold(
        backgroundColor: Color(0xfff6f6f7),
      appBar: AppBar(...), // AppBar
      body: Column(
          children: <Widget>[
        Expanded(
          child: GridView.count(...), // GridView.count
        ), // Expanded
        SizedBox(),
        Container(
            padding: const EdgeInsets.all(10.0),
            width: 420.0,
          child: Column(...), // Column
        ) // Container
          ], // <Widget>[]
        )); // Column, Scaffold
  }
}
```

Figura 9. Pjesë e kodit

```
class _ProfileState extends State<Profile> {
  @override
  Widget build(BuildContext context) {
    myCard(iconcode, route, icontitle, colour) {
      return Card(
        color: Colors.white,
        child: Ink(
          color: Colors.white,
          child: InkWell(
            splashColor: Colors.grey,
            onTap: () {
              Navigator.push(
                context,
                MaterialPageRoute(builder: (context) => route),
              );
            },
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              crossAxisAlignment: CrossAxisAlignment.center,
              children: <Widget>[
                //icon
                CircleAvatar(...), // CircleAvatar
                //title
                Text(...), // Text
              ], // <Widget>[]
            ), // Column
          ), // InkWell
        ), // Ink
      ); // Card
    }
```

Figura 10. Pjesë e kodit

Një pasqyrë e qartë se si duken këto pjesë të kodit në User interface shihet në figurën 10. Pra lista 2D lëvizëse e ndarë në kategori të detyrave (tasqeve) ku si lloje të kategorive kemi: All (Të gjitha) ku shfaqen të gjitha tasqet e krijuara pa marr parasysh se në cilën kategori dhe mund të krijohen tasqe të pa-kategorizuara, Work(Puna) ku shfaqen dhe krijohen tasqe në lidhje me punën, Home(Shtëpi) ku shfaqen dhe krijohen tasqe në lidhje me shtëpinë, Personal(Personale) ku shfaqen dhe krijohen tasqe personale, Freetime(Koha e lirë) ku shfaqen dhe krijohen tasqe në lidhje me kohën e lirë, Grocery(Blerjet) ku shfaqen dhe krijohen tasqet në lidhje me blerjet dhe në fund kemi

vetëm ikonën Recently Deleted(Së fundi të fshira) që paraqet një listë të të gjitha tasqeve të fshira nga cila do kategori tjetër.

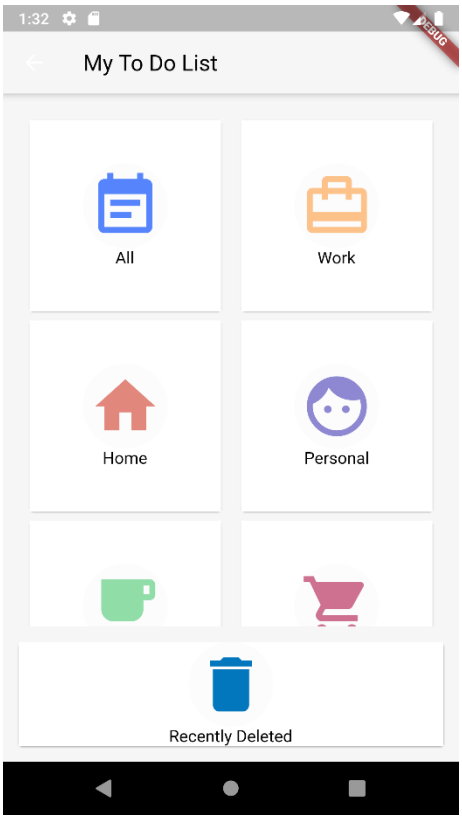

Figura 11. Faqja e parë e aplikacionit

5.4.2 Shtimi i një tasku të ri

Kur klikojmë në cilën do prej ikonave me kategorinë që dëshirojmë përveq ikonës 'Recently Deleted' ne mund të krijojmë një task të ri, varësishtë nga lloji i kategorizimit të taskut. Marrim shembull se po klikojmë në ikonën Work e që si file e gjejmë me emrin e klasës Work.dart, e cila është e trashëguar nga klasa StatefulWidget dhe duket si ne figurën 11. Në krye të kësaj dritare kemi një listë-ikonë e cila na kthen prap në dritaren e Profile, poshtë ikonës kemi titullin e që është një text widget ashtu sikur se edhe numërimi i tasqeve të krijuara që bëhet me anë të kësaj pjese të kodit:

```
Text(

'${Provider.of<TaskData>(context).taskCountByType(Contastants.WORK_TASKS)}
Tasks',
  style: TextStyle(
    color: Colors.white,
    fontSize: 18,
```

```
  ),
),
```

ku rreshti i kodit ‘Provider.of<TaskData> (context). taskCountByType (Constants. WORK_TASKS) } ‘ e përditëson parametrin taskCountByType sa herë që krijohet një task i ri në kategorinë e caktuar.

Në fund të dritares kemi një floatingbutton i cili është ai kryesori që hap dritaren për të shtuar një task të ri dhe që për përdoruesit duket si në figurën 11, ky buton që vjen nga Material Design dhe zakonishtë përdoret për veprime pozitive si “krijo”, “shperndaj”, “shto” e që në rastin tonë e kemi përdorur për të shtuar një task. Në projektin tonë kodi për këtë buton duket kështu:

```
floatingActionButton: FloatingActionButton(
    backgroundColor: Color(0xfffdc289),
    child: Icon(Icons.add),
    onPressed: () {
      showModalBottomSheet(
          context: context,
          isScrollControlled: true,
          builder: (context) => SingleChildScrollView(
              child:Container(
                padding: EdgeInsets.only(bottom:
MediaQuery.of(context).viewInsets.bottom),
                child: AddTaskScreen(Contastants.WORK_TASKS),
              )
          )
      );
    }
),
```

Ky buton na dërgon në një klasë tjetër e që është klasa AddTaskScreen ku sigurishtë se edhe kjo klasë vjen nga klasa StatefulWidget.

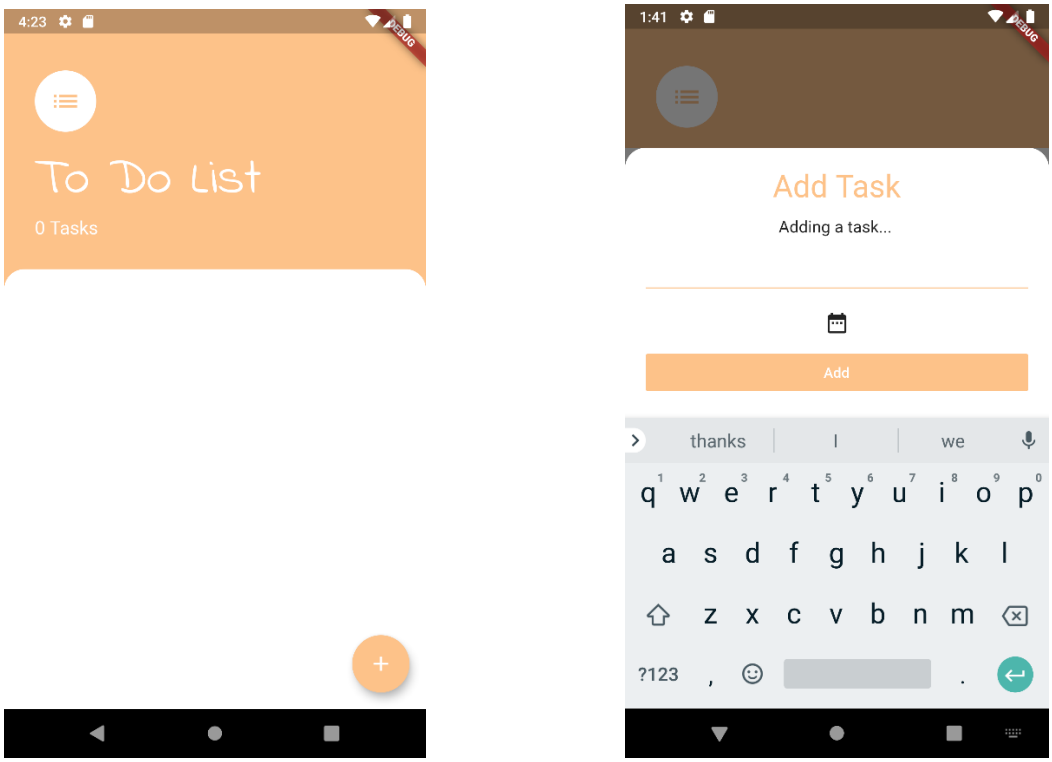

Figura 12. Dritarja për shtimin e një tasku

Siq shihet edhe nga figura 11 pas klikimit në Floating Button-in me ikonën ‘add’ hapet kjo dritare në formë të një modali që përmban një titull, hapësirën për të shkruajtur (text field), një kalendar ikonë që do ta shpjegojmë në pikën 5.4.4, dhe Flat Butoni me ikonën ‘add’ i cili pasi ta shkruajmë taskun e shton atë në listën e tasqeve në kategorinë në të cilën jemi e që në rastin tonë shtohet në kategorinë e klasës Work.

```
class AddTaskScreen extends StatefulWidget {

  int addingFrom;
  AddTaskScreen([int addingFrom = 1]) {
    this.addingFrom = addingFrom;

  }

  @override
  _AddTaskScreenState createState() => new _AddTaskScreenState();
}

class _AddTaskScreenState extends State<AddTaskScreen> {
  String _selectDate="";
  String newTaskTitle='';

  Future<Void> openDatePicker(BuildContext) async {...}

  @override

  Widget build(BuildContext context) {

    return Container(
      color: Color(0xff757575),
      child: Container(...), // Container
    ); // Container
  }
}
```

Figura 12. Pjesë e kodit

Nga kodi më lartë në figurën 12 shohim se në klasën AddTaskScreen kemi deklaruar nje variabël te tipit integer addingFrom që na ndihmon të dijmë se nga cila kategori jemi duke shtuar një task, gjithashtu ketë variabël e kemi përdorur edhe për të rregulluar ngjyrat e dritarës, secila dritare ka ngjyrat e veta përkatese për llojin e kategorisë, poashtu kemi deklaruar edhe nje variabël te tipit String newTaskTile që na vjen si instancë e klasës TaskTile e cila është një klasë e trashëguar nga klasa StatlessWidget. Pasi që përdoruesi përfundon së shkruari një task i ipet edhe mundësia ti vendos datë taskut modul të cilin do ta shqyrtojmë në pikën 5.4.4, dhe pastaj duke klikuar në Flattbutton-in me ikonën ‘add’ e shtojmë taskun e krijuar me këtë pjesë të kodit:

```
onPressed: () {
  Provider.of<TaskData>(context).addTask(this.newTaskTitle,
widget.addingFrom, this._selectDate);
  log('task ' + this.newTaskTitle + ' - _selectDate' + this._selectDate);
  Navigator.pop(context);
},
```

ku funksionin e provider e kemi spjeguar në pikën më lartë.

5.4.3 Shfaqja e listës së tasqeve

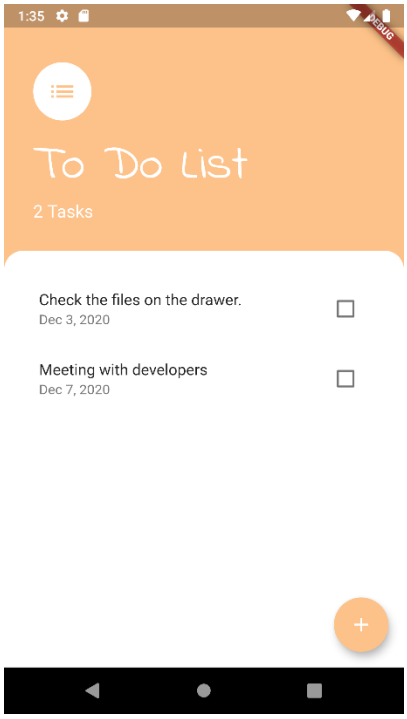

Figura 13. Dritarja e tasqeve të shkruara

Nga figura 14 shohim se si përdoruesit do ti shafqet lista e tasqeve që ka krijuar ne dritaren e kategorisë së caktuar. Kjo listë përmban tasqet e krijuara me datën e caktuar nga përdoruesi dhe në fund të rreshtit të secilit task shihet një kuti me anë të së cilës kur të klikohet vendoset shenja e standardizuar tick që tregon përfundimin e taskut, poashtu me klikimin në këtë kuti dekorohet tasku me anë të "line-through" – vizë nëpërmjet në shqip. Në këtë dritare poashtu shohim edhe numerimin e tasqeve si dhe listë-ikonën që na kthen prapa ne dritaren Profile.

Më poshtë shohim pjesë të rëndësishme të kodit i cili arrinë të shfaqë këtë listë të tasqeve:

```
class _TasksListState extends State<TasksList> {
  @override
  Widget build(BuildContext context) {
    return Consumer<TaskData>(
      builder: (context, taskData, child) {
        return ListView.builder(
          itemBuilder: (context, index) {
            final task = taskData.tasks[index];
            return task.taskType == widget.bucketType
                ? TaskTile(
                    taskTitle: task.name,
                    taskDate: task.taskDate,
                    isChecked: task.isDone,
                    checkboxCallback: (checkboxState) {
```

```
                        taskData.updateTask(task);
                      },
                      longPressCallback: () {
                        showAlertDialog(context, taskData, task);
//                      taskData.deleteTask(task);
                      },
                    )
                  : widget.bucketType == 0 && task.taskType != 6
                      ? TaskTile(
                          taskTitle: task.name,
                          taskDate: task.taskDate,
                          isChecked: task.isDone,
                          checkboxCallback: (checkboxState) {
                            taskData.updateTask(task);
                          },
                          longPressCallback: () {
                            showAlertDialog(context, taskData, task);
                          },
                        )
                      : new Container();
              },
              itemCount: taskData.taskCount,
            );
          },
        );
    }
}
```

5.4.4 Caktimi i datës së taskut

Caktimin e datës së një tasku e bëjmë duke klikuar në kalendar-ikonën e marrë nga paketa e ikonave në flutter, ku edhe do të hapet kalendari i datës së saktë ku përdoruesi do të ketë mundësi që të caktoj datën të cilën dëshiron. Në këtë kalendar përdoruesi do të ketë mundësi të caktoj vetëm një datë prej tre vite me herët se data aktuale dhe 3 vite më vonë se data aktuale. Kur përdoruesi të ketë përfunduar së vendosuri datën e dëshiruar duke shtypur butonin OK data do te shfaqet nën taskun e krijuar, ose të kundërtën duke shtypur butonin Cancel do të anuloj veprimin e vendosjes së datës. Kjo më së miri shihet në figurën më poshtë.

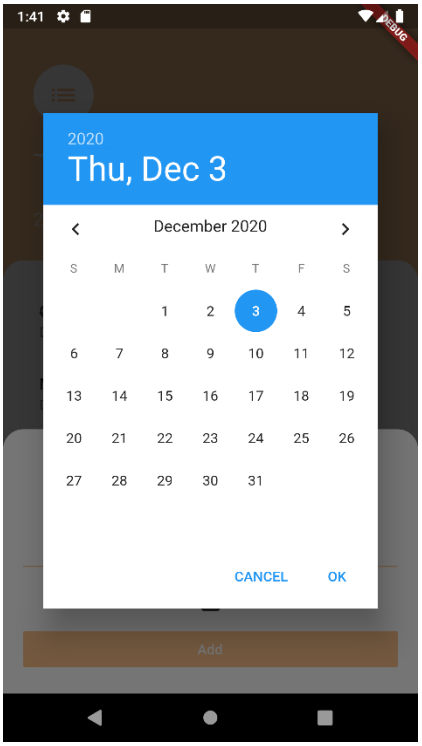

Figura 14. Vendosja e datës së një tasku

Të gjithë këtë aktivitet e mundësojnë rreshtat e kodit më poshtë, ku shihet vargu i caktimit të datës dhe formati i datës:

```
class _AddTaskScreenState extends State<AddTaskScreen> {
  String _selectDate="";
  String newTaskTitle='';

  Future<Void> openDatePicker(BuildContext) async {
    final DateTime d = await showDatePicker(
        context: context,
        initialDate: DateTime.now(),
        firstDate: new DateTime(DateTime.now().year - 3),
        lastDate: new DateTime(DateTime.now().year + 3));
    if(d !=null){
      setState(() {
        _selectDate = new DateFormat.yMMMd("en_US").format(d).toString();
      });
    }
  }
```

5.4.5 Përfundimi i një tasku

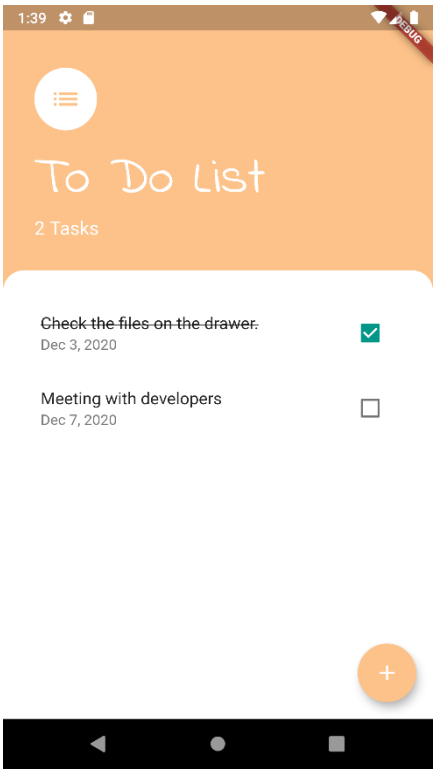

Figura 15. Përfundimi i një tasku

Pas analizimit të kërkesave të përdoruesve potencial të aplikacionit, përfundimin e një tasku e kemi bërë të duket si në figurën 15. Taskut të përfunduar i vendoset një vizë mbi dhe checkbox bëhet true duke u vendosur kështu shenja e standardiuzuar tick në kutinë e taskut. Më poshtë shohim rreshtat e kodit që mundësojnë paraqitjen e këtyre veprimeve në Userinterface:

```
@override
Widget build(BuildContext context) {
  return ListTile(
    onLongPress: longPressCallback,
    title: Text(
      taskTitle,
      style: TextStyle(
          decoration: isChecked ? TextDecoration.lineThrough : null),
    ),
    subtitle: Text(
      taskDate
    ),
    trailing: Checkbox(
      activeColor: Colors.teal,
      value: isChecked,
      onChanged: checkboxCallback,
    ),
  );
```

Sic e shpjeguam edhe më lartë në gjuhën programuese përmes kushtit ternar kontrollojmë se a është tasku i përfunduar apo jo, nëse po atëhere tekstin ose taskun e dekorojme me një vizë mbi, ndërsa në checkbox vendosim shenjën tick me ngjyrë.

5.4.6 Fshirja e një tasku

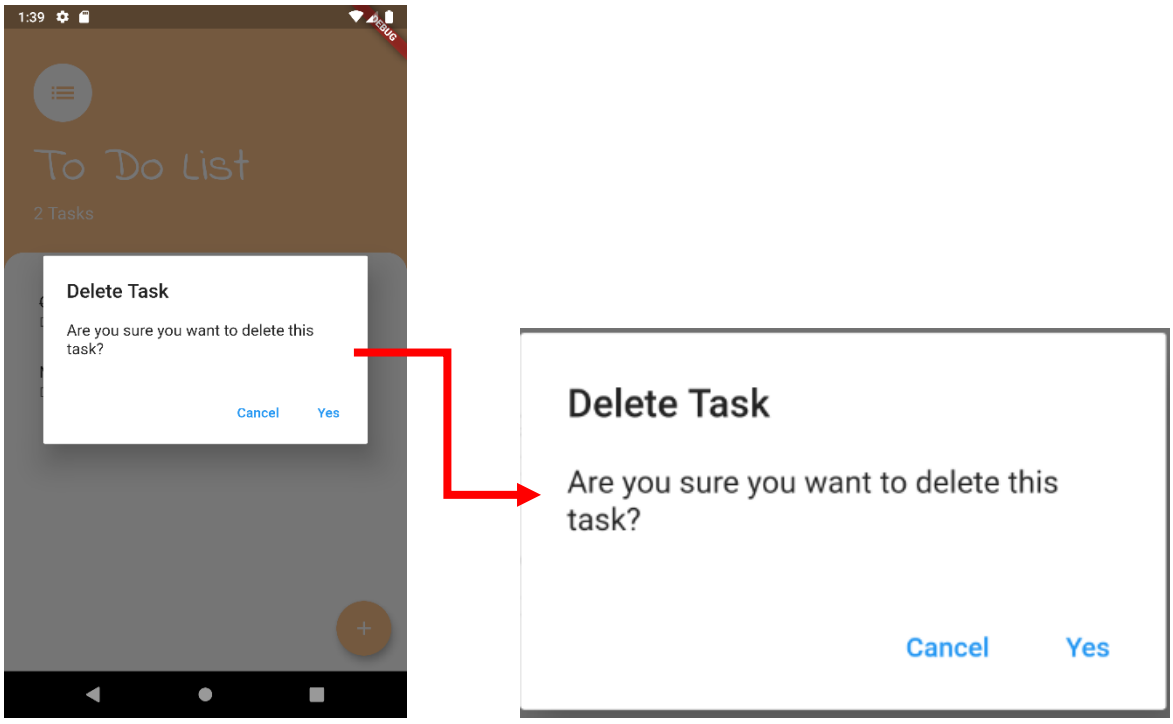

Figura 16. Fshirja e një tasku

Kur përdoruesi dëshiron të fshijë një task gjithqka që duhet të bëjë është të mbajë të shtypur taskun për 2 deri në 3 sekonda dhe do ti paraqitet një alert dialog që do ta pyes nëse është i sigurtë që dëshiron të fshijë taskun e caktuar. Butoni Yes e fshin taskun dhe e dërgon taskun ne dritaren e 'Recently Deleted', ndërsa butoni Cancel e anulon këtë veprim. Arsyeja pse kemi vendosur të krijojmë këtë alert dialog është se shpesh herë na ndodh të fshiejmë dicka gabimishtë dhe për të mos pasur pasoja të humbjes së ndonjë mbledhje, ngjarje apo obligimi të rëndësishëm përdoruesit i terhiqet vëmendja me anë të kërkesës sonë për fshierjen e taskut. E me gjithë atë prap tasku mbetet për një kohë në dritaren "Recently Deleted" ku përdoruesi mund ta rikthej taskun e fshirë. Kodi më posht tregon me mirë se si është arritu ky veprim:

```
showAlertDialog(BuildContext context, taskData, Task task) {
  // Create button
  Widget cancelButton = FlatButton(
    child: Text('Cancel'),
    onPressed: () {
      Navigator.of(context).pop();
    },
  );

  Widget okButton = FlatButton(
    child: Text("Yes"),
    onPressed: () {
      taskData.deleteTask(task);
      Navigator.of(context).pop();
    },
  );
```

Ndërsa kjo pjesë e kodit tregon se si shfaqet alert dialogu, i cili përmban një titull, një text field dhe dy butona.

```
AlertDialog alert = AlertDialog(
    title: Text("Delete Task"),
    content: Text("Are you sure you want to delete this task?"),
    actions: [
      cancelButton,
      okButton,
    ],
  );

  // show the dialog
  showDialog(
    context: context,
    builder: (BuildContext context) {
      return alert;
    },
  );
}
```

6. DISKUTIME DHE PËRFUNDIME

Në këtë punim kemi analizuar vështirësitë e të gjithë atyre që përballen me obligime të shumëta gjatë gjithë kohës, si takime e ngjarje të ndryshme. Jemi mundur dhe besoj shumë se ja kemi dalur që me aplikacionin që kemi zhvilluar ti japim një zgjidhje shumë të mirë.

Në këto vitet e fundit aplikacionet kanë filluar të marrin një vëmendje nga të gjithë, duke filluar nga fëmijët, nxënësit, mësimdhënësit, punëtor, menaxher e madje pothuajse cdo profesion tjetër që ekziston. Smartphone-ët e tani në përgjithësi aplikacionet janë bërë pjesë e pandashme e punës dhe jetës së cdo njerit.

Kemi krahasuar mirë dhe analizuar në detaje dy aplikacione të mëdha që janë zhvilluar nga kompani të mëdha sic është ajo e Microsoft dhe Doist.ltd. Microsoft To Do është një aplikacion i cili ka me miliona shkarikme dhe është ndër aplikacionet për mbajtjen e një liste për detyrat apo obligimet e gjithë secilit nxënës, puntor apo edhe njerëzve të thjesht të cilët pa marr parasysh profesionet apo gjendjes kanë obligimet të cilat ju duhet ti shkruajnë për të mos i harruar. Ky aplikacion ka disa vecori si integrimi i Microsoft Office 365 dhe email-it, ku me kohë shpresojmë që edhe në aplikacionin tonë të mund të integrojmë dhe të bëjmë lidhje me aplikacionet tjera për zgjidhje më të mira për të gjithë përdoruesit. Todois është një tjetër aplikacion me karakteristika të mira, i cili i shfaq tasqet(detyrat) në formë të një liste ku mund edhe të shtohen shokë, kolegë për të bashkëpunuar me tasqet(detyrave) e juaja, kjo është një alternativë shumë e mirë për ekipe që punojnë në projekte të përbashkëta dhe shkruarja e detyrave në një listë të përbashkët ju ndihmon në suksesin më të shpejtë të projektit. Këtë alternative për aplikacionin tonë nuk e kemi parë të arsyeshme në këtë fazë pasi që kemi synuar dicka më të thjeshtë që të mund të përdoret nga të gjithë, e edhe nga persona të moshuar, të pa punë por që kanë obligime të tjera të ndryshme.

Poashtu ne nga krahsimi i këtyre dy aplikacioneve me aplikacionin tonë kemi vërejtur se përfundimi i një detyre bëhet duke u larguar detyra nga lista, në aplikacionin tonë me sugjerimin e mbi 10 personave që kemi pyetur kemi vendosur që përfundimin e një detyre ta dekorojmë me një vijë të drejtë mbi të, kështu duke mos e larguar taskun nga lista, duke i dhënë mundësi përdoruesit që të mund të shoh taskun të cilin e ka përfunduar, karakteristikë kjo e aplikacionit tonë që bënë dallimin nga aplikacionet e tjera.

7. REFERENCAT

[1] S. O'Dea, «Statista,» 10 2020. [Në linjë]. Available:https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/#:~:text=The%20number%20of%20smartphone%20users,the%20100%20million%20user%20mark..

[2] P. C. ". A. E. S. 2. VisionMobile, «Wikipedia,» Plum Consulting, [Në linjë]. Available: https://en.wikipedia.org/wiki/Mobile_app_development#:~:text=Mobile%20app%20development%20is%20the,digital%20assistants%20or%20mobile%20phones.&text=Mobile%20UI%20considers%20constraints%2C%20contexts,mobility%20as%20outlines%20for%20design..

[3] Pixelfield, «📱The ultimate guide to mobile app development - tips, prices & more!". ,» 2 12 2019. [Në linjë]. Available: https://pixelfield.co.uk/app-development/.

[4] Apple Open Source,25 09 2020. [Në linjë]. Available: https://opensource.apple.com/.

[5] Charting The Explosive Growth of the App Store, 15 10 2018. [Në linjë]. Available: https://www.lifewire.com/how-many-apps-in-app-store-2000252.

[6] Apple, iOS 14 is available today, 16 09 2020. [Në linjë]. Available: https://www.apple.com/newsroom/2020/09/ios-14-is-available-today/.

[7] D. T. Akshay, What Is The iOS Operating System?, 31 12 2019. [Në linjë]. Available: https://digitaltechakshay.medium.com/.

[8] MakeUseOf, Is Android Really Open Source? And Does It Even Matter?, [Në linjë]. Available: https://www.makeuseof.com/tag/android-really-open-source-matter/.

[9] Android – Google Mobile Services, 21 10 2018. [Në linjë]. Available: https://www.android.com/gms/.

[10] A. D. Magazine, «Google and Android Are Not the Same... and That's a Good Thing,» 29 07 2020. [Në linjë]. Available: https://appdevelopermagazine.com/google-and-android-are-not-the-same...-and-that's-a-good-thing/.

[11] AppBrain, "Number of Android applications on the Google Play store",12 08 2020.

[Në linjë]. Available: https://www.appbrain.com/stats/number-of-android-apps.

[12] A. Chebbi, Choosing the best programming language for mobile app development, 02 07 2019. [Në linjë]. Available: https://developer.ibm.com/articles/choosing-the-best-programming-language-for-mobile-app-development/#:~:text=Java%20was%20the%20default%20language,it%20is%20owned%20by%20Oracle)..

[13] S. Rahman, A simplified introduction to Dart and Flutter,04 03 2019. [Në linjë]. Available: https://www.freecodecamp.org/news/https-medium-com-rahman-sameeha-whats-flutter-an-intro-to-dart-6fc42ba7c4a3/.

[14] A. Ron, Google's Dart language on Android aims for Java-free, 120 FPS apps, Ars Technica, 01 04 2015. [Në linjë]. Available: https://arstechnica.com/gadgets/2015/05/googles-dart-language-on-android-aims-for-java-free-120-fps-apps/.

[15] G. conference, "Dart, a new programming language for structured web programming",10 10 2011. [Në linjë]. Available:http://gotocon.com/aarhus2011/presentation/Opening%20Keynote:%20Dart,%20a%20new%20programming%20language%20for%20structured%20web%20programming.

[16] L. Seth,What is Dart, 16 08 2014. [Në linjë]. Available: http://radar.oreilly.com/2012/03/what-is-dart.html.

[17] Sky: An Experiment Writing Dart for Mobile (Dart Developer Summit 2015), 2015 [Në linjë] Available:https://www.youtube.com/watch?v=PnIWl33YMwA&ab_channel=GoogleDevelopers.