TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Saara Denisov 205848IADB

Finantskande periodiseerija arendus Directo äritarkvarale

Bakalaureusetöö

Juhendaja: Meelis Antoi
Magistrikraad

Tallinn 2025

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: Saara Denisov

02.01.2025

Annotatsioon

Käesoleva bakalaureusetöö eesmärgiks on luua raamatupidamistarkvara finantskandele periodiseerimise funktsionaalsus.

Töö katab olemasolevate lahenduste ja sobilike tehnoloogiate analüüsi, kirjeldab arenduse käiku ning testimist. Töös kirjeldatakse ka edasiarenduse võimalusi.

Arenduse tulemusena valmib periodiseerimise funktsionaalsus, mis katab Directo äritarkvara klientide vajadusi ning seeläbi võimaldab firmal pärandsüsteemi finantskande dokumendi ning selle periodiseerija kasutusest loobuda.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 30 leheküljel, 5 peatükki, 5 joonist, 1 tabelit.

Abstract

Development of Periodization Modal for Transaction Document for Directo Business Software

The aim of current thesis is to create a periodization modal for transactions in a business software using modern technologies and solutions.

The thesis contains the analysis of existing solutions and suitable technologies for solving this problem, describes the flow of the development and the testing process. Possibilities for further development are also noted.

As a result, a periodization functionality will be created that meets the needs of Directo business software's clients, enabling the company to phase out the use of the legacy system's transaction document and its periodization tool.

The thesis is in Estonian and contains 30 pages of text, 5 chapters, 5 figures, 1 table.

Lühendite ja mõistete sõnastik

API	Ingl k *Application Programming Interface*. Rakendusliides
Automaattestid	Programmiliselt kirjutatud testid
Eesrakendus	Ingl k *front-end application*. Tarkvara osa, millega kasutaja otseselt suhtleb
Finantskanne	Raamatupidamises kajastatav tehing kontode, objektide ja projektide tasandil
Hüpikaken	Väike aken või dialoogikast, mis ilmub kasutaja arvutiekraanile automaatselt või veebilehe kasutamise käigus
HTTP	Ingl k *Hypertext Transfer Protocol*. Protokoll, mis määrab kindlaks veebisüsteemide vahel andmete edastamise
Modaalaken	Algses aknas avanev sekundaarne aken kasutaja tähelepanu suunamiseks
Monoliitne arhitektuur	Ühe tervikuna arendatud ja juurutatud süsteem
Raamistik	Tööriistade ja reeglite kogum
REST	Ingl k R*epresentational State Transfer*. Tarkvaraarhitektuuri laad, mis määrab veebirakenduste loomiseks kindlad piirid
Tagarakendus	Ingl k *back-end application*. Tarkvara osa, mis haldab äriloogikat, andmete töötlemist ja salvestamist
Teek	Ingl k *library*. Korduvkasutatavate koodikomponentide kogum
Viisard	Interaktiivne abivahend, mis aitab kasutajal teha keerukamaid operatsioone

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Töö valmimise ajal töötas autor äritarkvara firmas Directo, kus viidi läbi tarkvarasüsteemide uuendamist. Ettevõttes on alustatud ligikaudu 20 aastat tagasi loodud ning järk-järgult kasvanud pärandsüsteemi koodi tänapäevaste ja tõhusamate tehnoloogiate peale üle viimisega, et tagada paremini skaleeruvad ja tulevikukindlamad lahendused. Antud ülemineku protsessi käigus viidi uutele lahendustele üle dokumendivaateid, sealhulgas ka finantskande dokumenti, millel oli puudu periodiseerimise funktsionaalsus. Sellest tulenevalt katab käesolev bakalaureusetöö raamatupidamistarkvarades vajalikule periodiseerimise protsessile analüüsi käigus sobivaima lahenduse leidmist, selle arendamist ning testimist.

Lõputöö autor soovib töö tulemusena luua lahenduse, mis kataks võimalikult paljudes levinud olukordades kasutajate vajadusi ning vähendaks sundolukordi andmete käsitsi sisestamiseks. Seeläbi vähendaks raamatupidamistarkvara kasutaja töömahtu ja automatiseerimine aitaks vältida inimfaktorist tulenevaid vigu. See on finantsandmete ning rutiinsete tegevuste puhul tähtis.

Töö koosneb viiest peatükist. Sissejuhatus annab ülevaate töö taustast. Teises peatükis kirjeldatakse probleemi ja olemasolevaid lahendusi. Analüüsi peatükis tutvustatakse analüüsi tulemusi. Neljandas peatükis kirjeldatakse teostatud lahendust ja selle käiku. Viimases peatükis kirjeldatakse tehtud töö tulemusi ning arutletakse võimalike edasiste arenduste teemal.

1.1 Metoodika

Kuna varasemalt on periodiseerimise probleemile lahendusi loodud Directo pärandsüsteemis ning konkureerivates tarkvarades, on probleemi edukaks lahendamiseks tarvis esmalt neid väljatöötatud lahendusi uurida. Analüüsitakse nende vastavust kasutajate vajadustele ja kasutusmugavust, aga ka võimalikke puudusi. Selgitatakse välja sobivaimad tehnoloogiad, struktuur ja lahendusviisid arenduse teostamiseks.

Tuginedes analüüsi tulemustele luuakse uus periodiseerija lahendus, mis võimaldab kasutajatel jagada summasid vastavalt nende valitud perioodidele ja mida saab kasutusele võtta Directo raamatupidamistarkvara osana. Seetõttu peab arvestama, et loodav lahendus oleks ühilduv olemasoleva Directo finantskande dokumendivaatega ning selle lisadega.

Praktilist osa testitakse nii käsitsi kui ka automaattestidega, et tagada lahenduse töökindlus. Arendusele järgneval ajaperioodil kogutakse tarkvara klientide tagasisidet ning analüüsitakse seda. Analüüsi tulemusena ja turumuudatuste järgi selgub vajadus edasisteks arendusteks ning lahenduse äriline kasu.

2 Ülevaade probleemist

Periodiseerimine võimaldab ettevõtetel raamatupidamises kulud ja tulud õigetele perioodidele jaotada, kui arve esitatakse pikema perioodi eest. Eriti tähtis on see siis, kui periood katab mitut majandusaastat või raamatupidamisperioodi, nagu näiteks igakuine teenuslepingu tasu või pikaajalised projektid. Sellisel juhul tagab periodiseerimine, et aruannetes ettevõtte majandustegevus õigesti kajastuks. Finantsandmete ebatäpsuste korral ei saa ettevõtte juhtkond teha teadlikke otsuseid või ei õnnestu ettevõtte maksukohustusi õigesti kajastada.

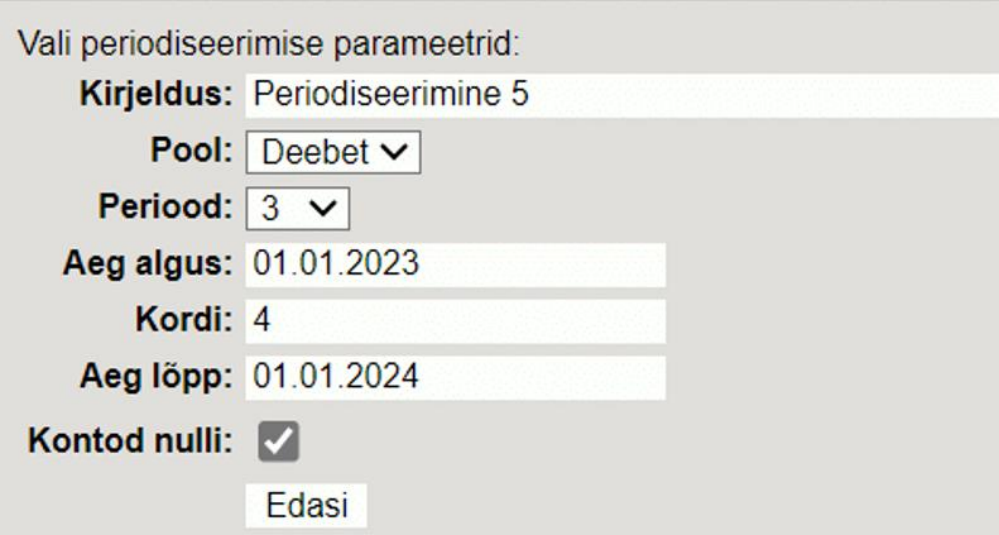
Manuaalne periodiseerimine on ajakulukas ja veaohtlik. Eriti keerukaks võib see muutuda suurte andmemahtude või tihti muutuvate lepingute puhul, kus on vajadus tulemust korduvalt muuta. Seetõttu on oluline arendada töökindel IT-lahendus, mis oleks lihtsasti kasutatav ning vastaks erinevate kasutajate vajadustele. Efektiivne periodiseerimise süsteem vähendaks töökoormust, kiirendaks protsesse ja minimeeriks inimliku eksimuse võimaluse, andes ettevõttele täpsed ja usaldusväärsed andmed, millele toetudes saaks teha informeeritud äriotsuseid.

2.1 Eksisteerivad lahendused

Käesoleva töö raames uuriti lähemalt Directo tarkvara pärandsüsteemi periodiseerijat ning konkureerivate ettevõtete periodiseerimise lahendusi. Peamiselt analüüsiti olemasolevate lahenduste funktsionaalsust ja kasutajasõbralikkust, et mõista nende eeliseid ja puudusi. Analüüsi käigus kujundati arusaam, mis funktsioonid on neis lahendustes kriitilise tähtsusega ja kuidas saaks neid täiustada või automatiseerida.

2.1.1 Directo pärandsüsteemi periodiseerija

Ettevõtte pärandsüsteemil on periodiseerimise funktsionaalsus, millel on mitmeid puudujääke. Pärandsüsteemis on olukord lahendatud hüpikaknaga, mille tõttu jääb periodiseeritud finantskande dokument eraldi aknas aktiivseks. Hüpikakna esimene samm võimaldab valida poolt, kuupäevi, periodiseerimiste arvu, kordi ja intervalli.



Joonis 1. Directo pärandsüsteemi periodiseerimise esimene samm

Teisel sammul saab valida kontosid ja käibemaksukoodi, samuti välja arvutatud summasid. Teise sammu sooritamisel luuakse kanded ning näidatakse õnnestumist kinnitavat sõnumit. Seejärel saab hüpikakna sulgeda ning lahti jäänud aknas finantskannet värskendada.

Joonis 2. Directo pärandsüsteemi periodiseerimise teine samm

Selle lahenduse korral ilmnevad sisestamisel tehtud vead alles periodiseerimise sooritamisel. Ühtlasi näeb periodiseeritava summa deebet või kreedit summa puudumisel sellekohast teavet alles teisel sammul, kui kasutaja on kulutanud aega kuupäevade valimisele. Autori hinnangul eeldab selle periodiseerija kasutamine eelnevalt juhendi lugemist. Lahendus pole eriti intuitiivne. Tarkvara kliendid on aja jooksul tõstatanud erinevaid probleeme, mis kasutamisel on tekkinud, osa neist pole lahendatud.

2.1.2 Konkureerivate ettevõtete periodiseerijad

Konkureerivate äritarkvarade periodiseerimise lahenduste uurimist raskendab olukord, et tihti õnnestub neid lähemalt uurida vaid tarkvara ärikliendina. Lõputöö raames uuriti lähemalt Directoga konkureerivate HansaWorldi ja Briox äritarkvarade periodiseerimise lahendusi.

HansaWorld äritarkvara finantsmoodulis on võimalus periodiseerida kulusid ja tulusid. Sealne lahendus eeldab periodiseerimise mudeli täitmist ning mudeli kasutamisele järgnev käitumine erineb sõltuvalt müügiarve või ostuarve periodiseerimisest. Kasutaja saab valida kande kuupäevasid, sisestades väärtusi „kuude“ ja „päevade“ tulpadesse. Periodiseeritud summade suhteid saab muuta, sisestades protsente. Autori hinnangul jätab HansaWorldi lahendus kasutajale aga palju eksimisruumi, näiteks tuleks periodiseerimise mudelit koostades tabeli viimasele reale eraldi võrdusmärk (=) sisestada, et midagi arvesummast kaduma ei läheks [1]. Sellised kasutaja lisaliigutused nõuavad eraldi kasutaja aega ja tähelepanu.

Probleemi lahendab natuke teisiti Briox raamatupidamistarkvara, kus koostatud müügiarve periodiseerimiseks avaneb aken, kus saab sisestada andmeid sama arve periodiseerimiseks. Kogu informatsioon on näha ühekorraga ning käib hetkel käsitletava arve kohta. Funktsionaalsus võimaldab näiteks valida perioodi või intervalli kuudes, muuta kirjeldust, sisestada kontosid ja muuta käibemaksuga periodiseerimise sätet. Sealhulgas kuvatakse kasutajale periodiseerimise summat ja tehtavate periodiseerimiste arvu. Periodiseerimise tulemusi saab nii salvestada kui esitada. See on autori arvates kasulik, sest kasutaja ei pea kohe tehtud tööd kinnitama ning vajadusel on kerge muudatusi teha. Samuti saab hiljem kõiki periodiseerimisi otsida, vaadata ning neid sooritamiseks märkida.

Briox tarkvara lahendus sarnaneb teatud aspektides Directo pärandsüsteemi periodiseerimise kasutusvooga, kuid valikute hulk ja tulemuse kohandamise võimalused on siiski oluliselt piiratud. Näiteks saab tekkivate summade suurusi muuta alles pärast periodiseerimise teostamist arve ridadelt. Kasutajamugavuse ja tööks kuluva aja mõistes pole selline lahendus autori hinnangul optimaalne.

2.2 Uue lahenduse skoop

Käesoleva bakalaureusetöö raames käsitletakse periodiseerimise funktsionaalsuse loomist. Selle hulka kuulub kasutajate vajaduste ja probleemi lahendamiseks sobivate tehnoloogiate analüüs, optimaalse disaini väljatöötamine, planeeritud arenduse teostamine ja selle testimine.

Antud funktsionaalsus loodi juba eksisteerivale äritarkvarale, mis tähendab, et arendustöös pidi arvestama mitte ainult uue lahenduse kvaliteediga, vaid ka olemasoleva süsteemi arhitektuuri ja tehnoloogiatega ühildumisega. Samuti järgiti ettevõtte poolt kehtestatud standardeid ja juhiseid, mis tagasid arenduse sujuva integreerimise tarkvarakeskkonda. Sihiks võeti hästi hallatav ja teiste süsteemi osadega integreeritav lahendus.

Töö ei hõlma andmebaasi valikut ja loomist, kuna need on juba olemas ja peavad selle arenduse raames tarkvara pärandsüsteemiga ühtima.

3 Analüüs

Käesolev peatükk annab ülevaate arenduse jaoks koostatud analüüsist. Selgitatakse välja nii arenduse funktsionaalsed kui ka mittefunktsionaalsed nõuded, analüüsitakse parimaid võimalikke lahendusviise, põhjendatakse probleemi lahendamiseks tehtud tehnoloogiate valikut ja otsuseid arhitektuuri ning disaini osas.

3.1 Nõuete määramine

Periodiseerija funktsionaalsust kaardistades võeti aluseks tarkvara pärandsüsteemi lahendus. Töö raames loodav uus periodiseerija peab ühilduma olemasoleva finantskande dokumendivaatega ning selle lisadega. Soovitava tulemuse nõuded kaardistati koostöös tootejuhi ja kasutajakogemuse disaineriga, lähtudes tarkvara kasutajate vajadustest.

Üks peamisi nõudeid uuele lahendusele on kuupõhise ja päevapõhise periodiseerimise võimaldamine. Periodiseerimise viis tuleneb kasutaja periodiseerimise perioodi algus- ja lõppkuupäevade valikust. Periodiseerimine on kuupõhine, kui algus- ja lõppkuupäeva vahele jäävad täiskuud, ehk alguskuupäeva ja lõppkuupäeva vahe on võrdne ühega või periodiseerimise algus- ja lõppkuupäevadeks on vastavalt kuu esimene ja viimane päev ning perioodi kuuluvate täiskuude arv jagub periodiseerimise intervalliga. Periodiseerimiste arv on võrdne täiskuude arvuga juhul, kui periodiseerimise intervall on võrdne ühega. Seega saab rakendada kuupõhist periodiseerimist ka siis, kui kannete kuupäevadeks pole kuu esimesed päevad, aga valitud perioodi kuuluvate täiskuude arvu saab jagada periodiseerimise intervalliks valitud väärtusega. Kuupõhise periodiseerimise korral jagatakse periodiseeritavat summat periodiseerimiste arvuga. Vastav tulemus on iga perioodi periodiseeritud summa, välja arvatud juhul, kui summasid on vaja ümardada. Vastavalt ärireeglitele lisatakse summa jagamisel tekkinud jääk alati viimase periodiseeritud kande summale. Võimalusel jaotatakse summad nii, et negatiivsete kannete tekkimist vältida.

Päevapõhise periodiseerimise korral jagatakse periodiseeritav kogusumma ajaperioodi kuuluvate päevade arvuga, arvutades nii välja iga päeva keskmise summa. Seejärel

korrutatakse tulemus iga perioodi jaoks sellesse kuuluvate päevade arvuga. Summade jaotamisel ja ümardamisel järgitakse kuupõhise periodiseerimisega sarnaselt samu reegleid.

Lahendus peab kuvama välja arvutatud perioodide summasid, võimaldama kasutajal neid muuta ning arvutama ümber viimast või esimest summat, et periodiseerimise kogusumma muutumatuna püsiks. Negatiivse summa tekkimisel tuleb kasutajat hoiatada. Vigaste sisendite puhul on vaja kasutajale viga kuvada.

Loodav periodiseerija peab kasutama tarkvara dokumendivaadete mustandisüsteemi, et pärast periodiseeritud kannete koostamist oleks võimalik kasutajal mugavalt tehtud muudatused tagasi võtta. Finantskande mustandi ridadele periodiseeritud ridu lisades peab säilima ridade ajaline järjekord ning see peab jääma korrektseks ka hiljem ridade juurde lisamisel.

Lisaks peab arvestama asjaoluga, et lahendus oleks arvutiekraanilt mugavalt kasutatav kuni kolme eri konto üheaegse periodiseerimisega. See tähendab, et disainist lähtuvalt mahuks horisontaalselt ekraanile korraga kõigi periodiseeritavate kontode info.

Tabel 1. Periodiseerija nõuded

Nõue	Nõude tüüp
Kasutaja saab sisestada perioodide arvu, intervalli ja kuupäevi	Funktsionaalne
Periodiseerija arvutab summasid vastavalt kuu- või päevapõhisele periodiseerimisele	Funktsionaalne
Jaotatud summadest kuvatakse muudetavat ülevaadet	Funktsionaalne
Sisendeid peab valideerima ja kasutajat vajadusel teavitama	Funktsionaalne
Periodiseeritud summad salvestatakse finantskande ridade mustandi tabelisse õige ajalise järjenumbriga	Funktsionaalne
Periodiseerija peab integreeruma olemasoleva süsteemiga	Funktsionaalne
Kuni kolme konto info mahub korraga periodiseerijas arvutiekraanile	Mittefunktsionaalne

3.2 Tehnoloogiate valik

Kuna käsitletav lahendus kavandatakse juba olemasolevale süsteemile, mõjutas see tugevalt töö teostuseks kasutatavate tehnoloogiate valikut.

Andmebaasi struktuur ning tabelid jäid olemasoleva süsteemiga võrreldes töö raames samaks ning midagi juurde ei lisatud, sest eksisteerivad tabelid katsid arendatava lahenduse nõuded. Kui antud lahendus oleks valminud olemasolevast süsteemist iseseisvana ning ühilduvust poleks arvesse võetud, siis oleks autor analüüsinud võimalusi struktuuri optimeerimiseks ja ümberkujundamiseks, et võimalusel parandada päringute täitmise kiirust.

Eesrakenduse jaoks oli senises koodibaasis kasutusel TypeScript programmeerimiskeel, osaliselt ka JavaScript. TypeScript on täiendus JavaScriptile, mis võimaldab lisada tüübikirjeldusi, see muudab koodi arusaadavamaks ja aitab tüübivigu enne käitusaega avastada [2]. Kasutajaliideste loomiseks kasutati React raamistikku. React võimaldab andmete muutumisel uuendada kasutajale kuvatavaid komponente ilma kogu lehte uuesti laadimata [3]. See parandab rakenduse jõudlust, ühtlasi lihtsustavad üksteisest eraldatud komponendid rakenduse loomist ja haldamist. React koos tugevalt tüübitud TypeScripti keelega katavad autori hinnangul hästi kavandatava lahenduse eesrakenduse tehnoloogia vajadusi, seetõttu muudatusi sisse ei viidud. Eesrakenduse olekuhalduseks kasutati lisaks Reduxi teeki, sest see võimaldab hallata kogu rakenduse olekut ühes kohas, muutes andmete jagamise komponentide vahel lihtsamaks [4].

Tagarakenduse jaoks oli ettevõttes kasutusel C# programmeerimiskeel ja .NET raamistik, mis on laialdaselt tuntud oma jõudluse, turvalisuse ja integreerituse poolest [5]. Probleemi oleks saanud lahendada ka näiteks Java programmeerimiskeelt kasutades. Java on hea jõudlusega, paljude teekidega ja hea ühilduvusega [6]. Selle kasutamiseks on autoril tekkinud vilumus. Autori hinnangul sobis siiski C# ja .NET periodiseerimise funktsionaalsuse tagarakenduse vajadustega hästi, sest võimaldab sujuvat integratsiooni olemasoleva infrastruktuuriga.

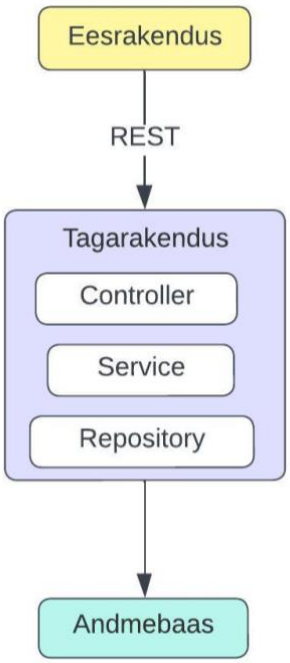
Automaattestide kirjutamiseks otsustati kasutada Jest testimisraamistikku. Antud raamistik valiti seetõttu, et see töötab TypeScripti ja Reacti projektidel, vajab vähest konfigureerimist, on laialt levinud, pakub lihtsaid meetodeid testide kirjutamiseks ja käivitamiseks ning autor on sellega varasemalt tuttav [7].

3.3 Arhitektuur

Olemasolev süsteem koosnes ühest terviklikust rakendusest, kus kõik funktsionaalsused on omavahel tihedalt seotud, seega tegu on arhitektuuriliselt monoliitse süsteemiga [8].

Suhtlus käis REST arhitektuuri järgi, mis võimaldab andmete vahetust süsteemi osade vahel lihtsal ja standardsel viisil. Rakenduses kasutatakse standardiseeritud HTTP meetodeid. REST API vastab vigade korral selgete ja informatiivsete HTTP staatuse koodidega, mis võimaldab tekkivaid probleeme hõlpsasti tuvastada.

Süsteem on kihiline ja suurem osa äriloogikast kuulub tagarakenduse osasse. See tagab hea hallatavuse. Eesrakenduse kiht kuvab kasutajale andmeid ja haldab kasutaja interaktsioone. Tagarakenduse arhitektuuris on eraldi kihid kasutajaliidese ja äriloogika vahelise suhtluse jaoks. Äriloogika kihis rakendatakse ärireegleid ja käideldakse äriprotsesse. Suhtlus andmebaasiga kuulub omaette kihti. See võimaldab selget eraldatust eri tasandite vahel ja tagab, et ühe kihi muudatused ei mõjuta otseselt teisi kihte. Sellise arhitektuuri puhul on andmete liikumine selgem ja hästi määratletud. Samuti saab tulevikus juurde lisada uusi teenuseid või funktsioone ilma kogu süsteemi struktuuri muutmata.



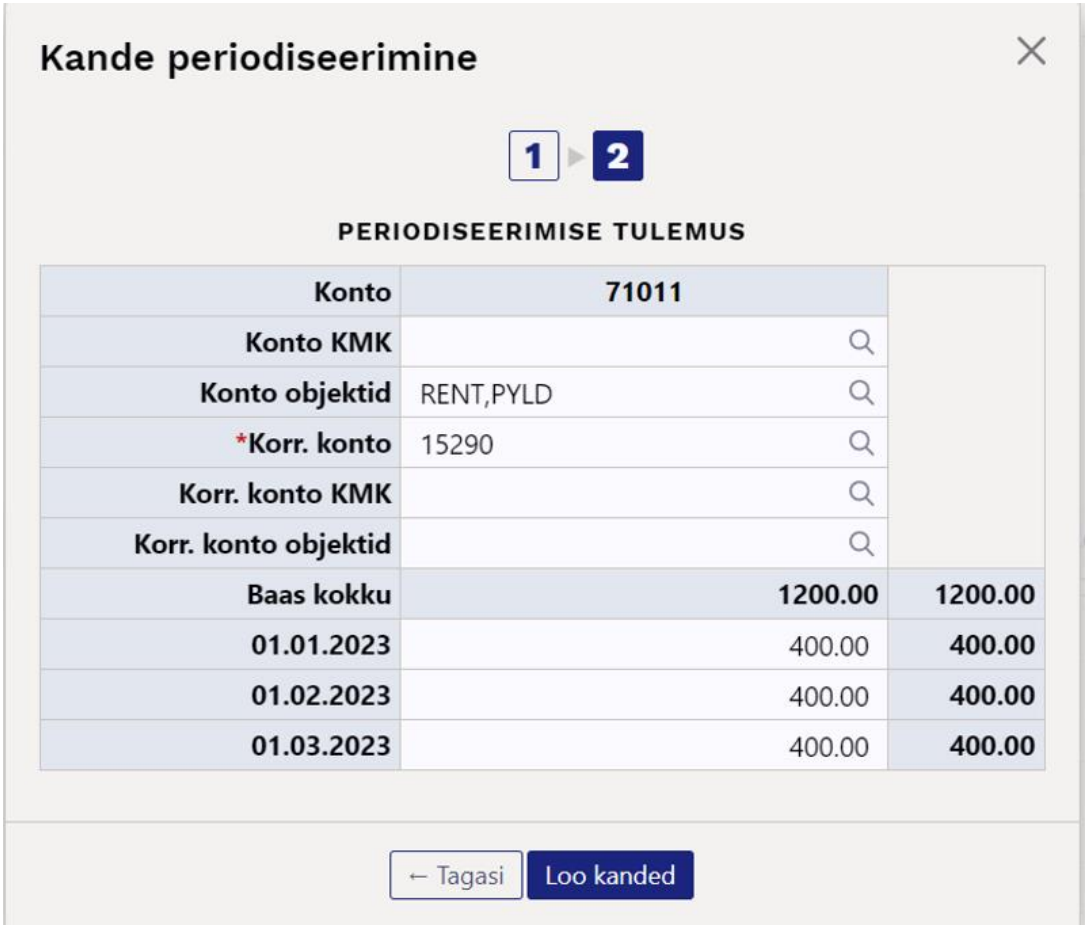
Joonis 3. Planeeritava lahenduse arhitektuur

3.4 Veebirakenduse disain

Kande periodiseerimise protsessi visuaalse poole teostamiseks otsutati kasutada modaalakent, sest sekundaaraken aitab juhtida kasutaja tähelepanu vaid neile andmetele, mis on sel hetkel ülesande täitmiseks kõige olulisemad [9]. Samuti pole modaalakent kasutades võimalik kasutajal samas aknas finantskande dokumendi andmeid muuta.

Modaalaknasse loodi mitmeetapiline viisard, mis annab kasutajale selged juhised ja sammud, et vajalikud andmed sisestataks õiges järjekorras. Viisard jagab andmete valimise, sisestamise ja muutmise protsessi kaheks. Esmalt saab viisardis valida ajaperioodi, kuupäevi ja periodiseerimise sagedust kuudes, järgmise sammuna saab kasutaja näha, muuta ja sisestada kulukontosid, objekte ja summasid. Protsessi on kasulik nii liigitada, kuna tegevuste järjekord püsib igal korral sama ja kuvatav info sõltub kasutaja sisestatud andmetest [10].

Joonis 4. Modaalakna esimene vaheaken



Joonis 5. Modaalakna teine vaheaken

Viisardi etappide vahel saab liikuda modaalakna jaluse nuppude ja päises olevate etapi numbrite kaudu eeldusel, et järgmise etapi nägemiseks vajalikud väljad on korrektselt täidetud. Kasutaja sisestatud andmete valideerimiseks kuvatakse kohtspikreid ning teavitusi akna paremal küljel. Vigadest märku andmine ja viisardis edasi liikumise takistamine aitab vältida vigaseid tulemusi ning teavitab kasutajat vajalikest muudatustest. Lahendus kasutab ka juba tarkvaras olemasolevat asetaja komponenti, miniotsingut ja mustandisüsteemi, et järgida süsteemis ühist joont. Sellega tagatakse tarkvara kasutajale sujuv kogemus.

3.5 Valideerimine

Nõuete kohaselt oli tarvis lisada ka kasutaja sisendite valideerimise funktsionaalsus. Kuupäevade ja arvude sisestamiseks on periodiseerimisel kindlad ärireeglid, mille järgimist saab kontrollida. Teises vaheaknas saab kasutaja lisada aga juba tarkvarasse lisatud objekte, kontosid ja käibemaksukoode. Nende valideerimiseks süsteemis ühtne lahendus puudus. Ettevõtte arendusmeeskond on otsusel valideerimised tulevikus kõigi sarnaste olukordade jaoks ühtselt lahendada, seega anti juhised hetkel olukorra jaoks eri sisendite jaoks eraldi API otspunktid koostada.

Tagarakenduse poolel koostatakse sisendi kontrollimiseks andmebaasi päringud. Eesrakenduse pool märgitakse sisendikaste vigaseks, takistatakse kasutajal viisardis edasi liikumist ning kasutatakse tekkinud veast kasutaja teavitamiseks tarkvaras juba olemasolevat teavituste kuvamise süsteemi. Samuti arvestatakse olukorraga, kus mõnel juhul on kasutajal lubatud sisendiks kirjutada komaga eraldatult mitu objekti.

Eelkirjeldatud valideerimise funktsionaalsus aitab vältida kasutajal tekkivat segadust ning valede andmetega kande ridu.

4 Veebirakenduse arendus

Veebirakenduse arendusel pöörati tähelepanu koodi taaskasutusele. See kiirendab arendusprotsessi ja -kulu, vähendab vigade tekkimise tõenäosust ja tõstab skaleeritavust [11].

Lahenduse arendamisel järgiti agiilset arendusmeetodit, mis võimaldas paindlikku töökorraldust. Arendustsüklite lõpus hinnati töö tulemust ja täpsustati vajadusel täiendavaid nõudeid. See aitas tagada, et lõplik lahendus vastaks kõige paremini töö eesmärkidele ja klientide vajadustele.

Käsiteldavat lahendust arendas peamiselt autor üksi. Paralleelselt lahenduse arendamisega seda ka testiti. Testimisse panustasid lisaks töö autorile ka ettevõtte arendusmeeskonna tootespetsialist-testija ning finantsvaldkonna tootejuht.

4.1 Olemasolevate eesrakenduse komponentide kasutamine

Eesrakenduse jaoks kasutati mitmeid tarkvaras olemasolevaid komponente, nagu näiteks registri asetajat ja miniotsingut. Need lihtsustavad kasutajal oma äritarkvara süsteemi lisatud kontode, objektide ja käibemaksukoodide valimist. Lisaks on nende kasutamine tarkvara kliendile juba tuttav ja muudab seega finantskande periodiseerija kasutamise tarkvara kliendile intuitiivsemaks. Olemasolevate komponentide kasutamine kiirendab arendusprotsessi, kuna komponentide disain on juba kasutajamugavuse seisukohalt optimeeritud ja ei nõua täiendavat analüüsi, samuti ei kulu komponentide kirjutamisele ja testimisele täiendavat arendusressurssi.

Analüüsist tulenevalt pidi periodiseerija ka kasutaja sisendeid kontrollima. Pärast analüüsi peatükis kirjeldatud tagarakenduse päringute loomist rakendati kasutaja vigadest teavitamiseks eesrakenduse olemasolevat märguannete näitamise süsteemi ning sisendiväljade vigase oleku kuvamist. Teavitussüsteemi standardiseeritud veateated hõlbustavad vigade mõistmist. Loodud lahendusel eesrakenduse komponentide kasutusele võtmisel probleeme ei tekkinud.

4.2 Mustandisüsteem

Üks loodava lahenduse nõuetest oli tarkvara mustandisüsteemi kasutamine. Selleks kasutati andmete salvestamiseks andmebaasi kannete mustandi tabelit. Tulenevalt tabeli ülesehitusest ja finantskande dokumendivaate loogikast oli tarvis lisada igale uuele periodiseeritud reale järjekorranumber. Sealjuures pidi säilitama dokumendi ridade ajalise järjestuse. Selgust ja head tava arvestades oli vajalik, et mitme sama kuupäevaga kande rea puhul kuvataks esimesena kõik deebet read ning nende järel samas järjekorras kreedit read. Rea järjekorranumbri arvutuseks koostati valem, mida iga uue periodiseeritud rea lisamisel kasutatakse (vt lisa 2). Õige järjekorranumbri loogika kohene lisamine väldib ka edasiste vigade tekkimist hilisemate finantskande dokumendil tehtavate muudatustega. Kirjeldatud loogika lisamisega tagati soovitud käitumine tarkvara finantskande dokumendi mustandite kasutamiseks.

4.3 Testimine

Projekti peamine testimine viidi läbi manuaalselt, sest arenduse funktsionaalsus on piisavalt väikse mahuga. Manuaalset testimist viidi läbi paralleelselt ees- ja tagarakenduse arendamisega. Arendust testis nii autor ise kui ka arendusmeeskonna testija ja tootejuht.

Lisaks kirjutas autor ka automaatteste arenduse kvaliteedi tagamiseks ja arendusprotsessi aja vähendamiseks. Automaattestimine on tarkvara testimise meetod, kus automatiseeritud vahendid kontrollivad tarkvara toimivust ja vastavust nõuetele [12]. Testid tõstsid arenduse tõhusust ja võimaldasid hiljem teha koodis muudatusi kindlustundega, et põhifunktsionaalsus jääb toimima.

5 Hinnang loodud lahendusele

Valminud arendus oli ettevõttele ja klientidele kasulik, kuna lahendus täitis ootused ja lahendas senised probleemid. Protsessi automatiseerimine vähendas inimlike vigade esinemise riski ja kiirendas tarkvaras tehtava töö kulgu.

Ettevõte võttis uue finantskande periodiseerija oma tarkvaras kasutusele ning seda saavad kasutada ka teiste riikide kliendid. Pärandsüsteemi periodiseerijale pole vajadust teha enam parandusi ning arendusmeeskond saab selle koos pärandsüsteemi finantskande dokumendiga kasutuselt kõrvaldada. See aitab ettevõtet pärandsüsteemilt uutele lahendustele ülemineku eesmärgi suunas.

Loodud funktsionaalsus oli piisavalt kohandatav, et ettevõtte sai selle kasutusele võtta sama tarkvara simulatsiooni dokumendivaatel. Sellega hoiti ettevõttes kokku arendusressurssi. Võrreldes tarkvara pärandsüsteemi lahendusega saab käesoleva töö käigus valminud periodiseerijale tulevikus hõlpsasti vastavalt vajadustele lisada uusi või kohandada olemasolevaid funktsioone. Arhitektuuri selgus lihtsustab ka koodi haldust ja vajadusel teostatavaid veaotsinguid. Olukord tõestab, et uus lahendus on üles ehitatud paindlikult.

5.1 Võimalused edasi arenduseks

Arenduse jätkuks oleks võimalus luua teistsuguse küljendusega vaade väga suure mahuga kannetele, kus periodiseeritavaid summasid ja kontosid on palju. Käesoleva lahendusega ei ole sellisel juhul andmete vaatamine kasutaja jaoks mugav. Kuna väga mahukad kanded ning nende periodiseerimine on erand ja arendusressurss on piiratud, siis antud arenduse käigus sellele ei keskendutud.

Praegune periodiseerimise modaalaken on küll kasutatav väikestel ekraanidel nagu mobiiltelefonid, aga autori hinnangul saaks kitsaste ekraanide jaoks vaadet selgemaks muuta. Käesoleva arenduse käigus mobiilvaate disainile rõhku ei pandud, kuna Directo tarkvara finantskande dokumendile tehakse muudatusi valdavalt arvutis.

Tulenevalt olukorrast, et ühe kande periodiseerimise periood võib ulatuda mitmete aastate peale, võiks arendatud funktsionaalsusel olla täiendavaid maksude arvestamise seadistusi lisaks käibemaksukoodile. Autori arvates võib ette tulla olukordi, kus perioodi kestel jõustuvad maksumuudatused, aga valminud periodiseerijal puudub võimalus selliste reeglite lisamiseks ning sel juhul tuleb periodiseerimine teostada mitmes eri osas või käsitsi. Arenduse nõuetesse see funktsionaalsus ei kuulunud.

Kasutajakogemusele aitaks veel kaasa see, kui süsteem õpiks kasutaja korduvaid mustreid ja õpiks selle põhjal kasutaja tööprotsessi optimeerima. Süsteem võiks näiteks eelnevatele valikutele toetudes automaatselt sooritada perioodide jaotust. Käesoleva lahenduse skoopi sellised funktsionaalsused ei mahtunud ressursi- ja arenduskulukuse tõttu.

6 Kokkuvõte

Käesoleva töö eesmärgiks oli luua periodiseerimise funktsionaalsus, mis oleks paindlik, aitaks vähendada vigade esinemist andmesisestuses ja tõhustaks töövoogu. Töö käigus analüüsiti olemasolevaid lahendusi, tuvastati nende kitsaskohti ning töötati koos ettevõtte meeskonnaga välja lahendus, mis võimaldab efektiivsemalt kandeid periodiseerida. Analüüsi käigus selgitati välja sobivaimad tehnoloogiad ja lahendusviisid arendatava funktsionaalsuse teostamiseks ning neid rakendati väljatöötatud lahenduse arendamisel. Arendusressursi efektiivseks kasutamiseks ja tarkvara ühtse stiili järgimiseks pöörati tähelepanu koodi taaskasutusele ning kasutati tarkvara olemasolevaid komponente. Lahenduse testimisprotsess kinnitas lahenduse nõuetele vastamist.

Uus modaalaken võimaldab sisendite valideerimist, perioodide kohandamist ja andmete automaatset jaotamist kasutaja sisendite alusel. Lahendus aitab ennetada võimalikke vigu ja tagab sisestatud andmete kvaliteedi.

Töö tulemusena valmis tööriist, mis tõstab kasutaja töö efektiivsust. Antud lahendus võeti Directo tarkvaras kasutusele.

7 Kasutatud kirjandus

[1] Excellent, „Kuidas ostuarvega saab jagada kulud tuleviku perioodidele?,“ 22 oktoober 2024. [Võrgumaterjal]. Loetav aadressil: https://www.excellent.ee/kasutajatugi/kuidas-ostuarvega-saab-jagada-kulud-tuleviku-perioodidele/. [Kasutatud 19 november 2024].

[2] Microsoft, TypeScript ametlik dokumentatsioon, [Võrgumaterjal]. Loetav aadressil: https://www.typescriptlang.org/. [Kasutatud 15 november 2024].

[3] Meta Platforms Inc, React ametlik dokumentatsioon, [Võrgumaterjal]. Loetav aadressil: https://react.dev/. [Kasutatud 15 november 2024].

[4] Geeks for geeks, „Why we need Redux in React?,“ 14 märts 2024. [Võrgumaterjal]. Loetav aadressil: https://www.geeksforgeeks.org/why-we-need-redux-in-react/. [Kasutatud 25 november 2024].

[5] Fahim ul Haq, „Why C# and .NET are still relevant in 2024,“ 01 märts 2023. [Võrgumaterjal]. Loetav aadressil: https://www.educative.io/blog/c-sharp-dot-net-relevance. [Kasutatud 15 november 2024].

[6] Medium, „Is Java an Excellent Choice for Backend Development? | Advantages of Using Java in Backend Development,“ 4 oktoober 2023. [Võrgumaterjal]. Loetav aadressil: https://medium.com/@shivakumar.h2kinfosys/is-java-an-excellent-choice-for-backend-development-d175494e27ab. [Kasutatud 16 november 2024].

[7] OpenJS Foundation, JestJs ametlik dokumentatsioon, [Võrgumaterjal]. Loetav aadressil: https://jestjs.io/. [Kasutatud 14 november 2024].

[8] R. Awati, „What is monolithic architecture in software?,“ [Võrgumaterjal]. Loetav aadressil: https://www.techtarget.com/whatis/definition/monolithic-architecture. [Kasutatud 23 november 2024].

[9] J. Juviler, „What Is a Modal and When Should I Use One?,“ 17 September 2022. [Võrgumaterjal]. Loetav aadressil: https://blog.hubspot.com/website/modal-web-design. [Kasutatud 22 november 2024].

[10] A. Coyle, „How to Design a Form Wizard,“ 5 September 2017. [Võrgumaterjal]. Loetav aadressil: https://coyleandrew.medium.com/how-to-design-a-form-wizard-b85fe1cc665a . [Kasutatud 23 november 2024].

[11] OpsLevel, „What is code reuse and why is it important?,“ 11 juuli 2024. [Võrgumaterjal]. Loetav aadressil: https://www.opslevel.com/resources/what-is-code-reuse-and-why-is-it-important. [Kasutatud 20 november 2024].

[12] SmartBear, „Automated Testing,“ [Võrgumaterjal]. Loetav aadressil: https://smartbear.com/learn/automated-testing/. [Kasutatud 25 november 2024].

[13] Directo OÜ, „Finantskanne,“ [Võrgumaterjal]. Loetav aadressil: https://wiki.directo.ee/et/fin_kanne. [Kasutatud 20 november 2024].

[14] Riigi Teataja, 20 november 2002. [Võrgumaterjal]. Loetav aadressil: https://www.riigiteataja.ee/akt/127122016003. [Kasutatud 15 november 2024].

[15] Hansa Financials, „Hansa Financials Finantsmoodul,“ [Võrgumaterjal]. Loetav aadressil: https://www.excellent.ee/pdf/Finants.pdf. [Kasutatud 20 november 2024].

[16] W. Kenton, „Accrue: Definition, How It Works, and 2 Main Types of Accruals,“ [Võrgumaterjal]. Loetav aadressil: https://www.investopedia.com/terms/a/accrue.asp. [Kasutatud 15 november 2024].

[17] A. Tuovila, „What Are Accruals? How Accrual Accounting Works, With Examples,“ 25 juuni 2024. [Võrgumaterjal]. Loetav aadressil: https://www.investopedia.com/terms/a/accruals.asp. [Kasutatud 20 november 2024].

[18] Briox, „Periodiseeri müügiarveid,“ [Võrgumaterjal]. Loetav aadressil: https://briox-ee.zendesk.com/hc/et-ee/articles/360009750300-Periodiseeri-m%C3%BC%C3%BCgiarveid. [Kasutatud 16 november 2024].

[19] Briox, „Accrue a Customer Unvoice,“ [Võrgumaterjal]. Loetav aadressil: https://help.briox.fi/hc/en-us/articles/115002504672-Accrue-a-Customer-Invoice. [Kasutatud 16 november 2024].

[20] Microsoft, .NET ametlik dokumentatsioon, [Võrgumaterjal]. Loetav aadressil: https://learn.microsoft.com/en-us/dotnet/. [Kasutatud 15 november 2024].

[21] A. Kolawa ja D. Huizinga, Automated Defect Prevention: Best Practices in Software Management, 2007.

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

Mina, Saara Denisov

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teos „Finantskande periodiseerija arendus Directo äritarkvarale, mille juhendaja on Meelis Antoi
   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.

02.01.2025

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 – Valem periodiseeritud rea lisamise järjenumbri arvutuseks

```
deebetrea number = deebetrea number andmebaasi tabelis
kord = periodiseerimise kord (praeguse rea kuupäeva järjenumber)
ridade koguarv = lisatavate periodiseeritud ridade kogunumber
rea järjenumber = praegu lisatava rea järjenumber ridade koguarvust
ridade arv = kande dokumendil olemasolevate ridade arv

deebetrea number = kord * ridade koguarv + rea järjenumber + ridade arv
```