OULUN AMMATTIKORKEAKOULU

Aleksi Sorri

ODOON KUSTOMOINTI JA KÄYTTÖÖNOTTO MYYNNIN TOIMINNANOHJAUSJÄRJESTELMÄKSI

ODOON KUSTOMOINTI JA KÄYTTÖÖNOTTO MYYNNIN TOIMINNANOHJAUSJÄRJESTELMÄKSI

Aleksi Sorri
Opinnäytetyö
Kevät 2024
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma, Ohjelmistokehityksen suuntautumisvaihtoehto

---

Tekijä: Aleksi Sorri
Opinnäytetyön nimi: Odoon kustomointi ja käyttöönotto myynnin toiminnanohjausjärjestelmäksi
Työn ohjaaja: Jouni Juntunen
Työn valmistumislukukausi ja -vuosi: Kevät 2024 Sivumäärä: 41

Opinnäytetyö on toteutettu toimeksiantona Anicare Oy:lle. Toimeksiantaja halusi ottaa yrityksessään käyttöön Odoo-toiminnanohjausjärjestelmän, jolla yritykselle saataisiin tehokkaampi toiminnanohjausjärjestelmä etenkin myynnin osa-alueelle. Opinnäytetyön tavoitteena oli käyttöönottaa järjestelmä yrityksen Ubuntu-palvelimelle ja räätälöidä toiminnanohjausjärjestelmää yrityksen tarpeisiin sopiviksi. Räätälöintiin kuului esimerkiksi asiakastietojen vienti järjestelmään, asiakastietokorttien muuttaminen yritykselle sopivaksi sekä ohjelmointirajapinnan luonti Odoo-toiminnanohjausjärjestelmän ja Procountor-taloushallinnon ohjelmiston välille.

Odoon räätälöinti tapahtui pääasiassa ohjelmoimalla järjestelmään uusia moduuleita. Muutokset asiakastietokorttiin on toteutettu luomalla moduuli, joka mahdollistaa näkymän muokkaamisen attribuuteilla. Tämän avulla voitiin poistaa tarpeettomia kenttiä ja luoda tilalle uusia tarpeiden mukaisesti. Ohjelmointirajapinnan luonti Odoon ja Procountorin välille tapahtui myös luomalla järjestelmään moduuli. Tämän moduulin tarkoituksena oli, että Odoossa luodut laskut ja laskun tiedot siirtyvät Procountor-ohjelmistoon. Moduuli myös varoitti laskun tekijää toiminnanohjausjärjestelmässä, jos laskusta puuttui pakollisia tietoja, kuten asiakkaan postiosoite.

Toiminnanohjausjärjestelmä luotiin Odoon versiolle 16.0. Moduulit olivat yhteensopivia järjestelmän kanssa ja näillä muutoksilla saatiin yritykselle tehokkaampi toiminnanohjausjärjestelmä ja parempi asiakkuudenhallinta myynnille. Opinnäytetyössä, kuitenkin jäi useampi toteutus tekemättä ajanpuutteen takia. Nämä kehitysideat jäävät järjestelmään jatkokehitysideoiksi. Näitä toteutuksia olisi ollut esimerkiksi se, että Odoo suorittaa kutsun taloushallinnon ohjelmistolle, jolloin saadaan laskun maksutila Odooseen. Toiminnanohjausjärjestelmän kustomointi onnistui kaiken kaikkiaan hyvin, vaikkakin ongelmia esiintyi moduulien ja Odoon yhteensopivuuden takia.

Asiasanat: Odoo, Toiminnanohjausjärjestelmä, Moduuli, Ohjelmointi, Ohjelmointirajapinta

ABSTRACT

Oulu University of Applied Sciences
Bachelor's Degree in Information Technology, Option of Software Development

Author: Aleksi Sorri
Title of thesis: Odoo customization and implementation as a sales ERP system
Supervisor: Jouni Juntunen
Term and year when the thesis was submitted: Spring 2024
Number of pages: 41

This thesis was implemented as an assignment for Anicare Oy. The company desired to implement the Odoo enterprise resource planning system within their company, which would further enhance the company's CRM especially for the sales area. The goal of the thesis was to implement the system on the company's Ubuntu server and customize the enterprise resource planning system to suit the company's needs. The customization included, for example, exporting customer data to the system, changing customer data cards to suit the company's needs, and creating an application programming interface between the Odoo ERP system and the Procountor financial management software.

Odoo was customized mainly by programming modules into the system. The changes to the customer information were implemented by creating a module that edits the view with attributes. With this, it was possible to remove unnecessary fields and create new ones according to needs. The creation of an application programming interface between Odoo and Procountor also took place by creating a module in the system. The purpose of this module was to transfer the invoices created in Odoo to Procountor software. The module also warned the creator of the invoice in Odoo if the invoice was missing mandatory information, such as the customer's address.

The enterprise resource planning system was created for Odoo version 16.0. The modules were compatible with the system and these changes gave the company a more efficient enterprise resource planning system and better customer relationship management for sales. In the thesis, however, several implementations were left undone due to lack of time and remain as further development ideas. One of the proposed development ideas involved Odoo making an API call to retrieve and update the payment status of invoices within the system. The customization of the enterprise resource planning system was overall successful, although there were few problems due to the compatibility of the modules on Odoo.

Keywords: Odoo, Enterprise Resource Planning, Module, Software Development, Application Programming Interface

SISÄLLYS

1 JOHDANTO

Toiminnanohjausjärjestelmät ovat nykyisin olennainen osa yritysten liiketoimintaa. Näiden rooli liiketoiminnassa on keskeinen, kun yritykset pyrkivät esimerkiksi parantamaan tehokkuutta, vähentämään kustannuksia tai hallinnoimaan yrityksen eri osa-alueita paremmin. Toiminnanohjausjärjestelmät tarjoavat käyttäjille tehokkaita työkaluja, kuten asiakkuuden-, tuotannon- tai myynninhallintaa.

Tässä opinnäytetyössä perehdytään siihen, kuinka avoimeen lähdekoodiin perustuvaa Odoo-toiminnanohjausjärjestelmää voidaan muokata sopivaksi myynnin työkaluksi. Odoossa on tarkoitus muokata järjestelmää luomalla moduuleita, joiden on tarkoitus tuoda uusia toiminnallisuuksia järjestelmään tai muokata esimerkiksi näkymiä sopivammaksi myynnille. Odoossa valmiina olevia moduuleja voidaan muokata esimerkiksi vastaamaan paremmin yrityksen myyntiprosessia muokkaamalla valmiina olevia näkymiä poistamalla myynnille tarpeettomia kenttiä. Lisäksi Odooseen tuodaan muun muassa asiakastiedot ja konfiguroidaan asetuksia valmiiksi käyttöönottoa varten. Opinnäytetyön loppuvaiheessa on tarkoituksena käyttöönottaa viimeistelty versio Odoosta.

Opinnäytetyö tehdään toimeksiantona Anicare Oy:lle. Anicare Oy on syksyllä 2016 perustettu ohjelmistoalan yritys, jonka pääkohteena on porotaloudessa myydyt porojen seurantalaitteet, joilta saadaan tarvittaessa myös tietoa eläimen elintilan seurantaan. Työn tavoitteena on luoda yrityksen myynnille uusi toiminnanohjausjärjestelmä, joka syrjäyttäisi yrityksen nykyisen ERP-järjestelmän ja helpottaisi muun muassa asiakastietojen hallintaa ja laskujen luomista tai seurantaa. Yrityksen alkuperäisessä ERP-järjestelmässä on myös useita puutteita. Näistä puutteista mainittakoon esimerkiksi, että ERP-järjestelmässä yhden asiakkaan tiedot ovat usealla eri välilehdellä tehden tiedon hakemisen hankalaksi. Lisäksi edellisessä järjestelmässä asiakaspalvelussa esille tulleet asiat ovat sekaisin tehden samaan asiakaspalveluun liittyvät puhelut haastaviksi. (Anicare 2016.)

Odoo on avoimeen lähdekoodiin perustuva toiminnanohjausjärjestelmä, jonka ensimmäistä versiota TinyERP alettiin luomaan vuonna 2005, mutta jonka nimi Odoo vakiintui vasta vuonna 2014. Odoo on kilpailukykyinen vaihtoehto muille toiminnanohjausjärjestelmille. Odoon myyntivaltti onkin sen edullisuus ja helppo muokattavuus, sillä se on avoimeen lähdekoodiin perustuva järjestelmä. Odoo tarjoaa täysin maksuttoman Community-version, jota tässä työssä käytetään ja

jossa muokkaaminen jää järjestelmän käyttäjän vastuulle. Odoolla on myös maksullinen Enterprise-versio, jossa tulee useita maksullisia moduuleja ja esimerkiksi rajoittamaton toiminnallinen tuki sovelluskehitykselle (Pinckaers 2023.). Odoon kustomointi tapahtuu tässä opinnäytetyössä pääasiassa erilaisten moodulien kautta, joiden toiminnat vaihtelevat yksinkertaisesta kenttien piilottamisesta laskujen ja Procountor taloushallinnon sovelluksen rajapinnan luomiseen. Toinen tapa, jolla Odoota tullaan muokkaamaan, on sen omien asetusten kautta. Asetuksia sovelluksessa on todella paljon, minkä vuoksi asia vaatii perehtymistä.

Raportissa kuvataan ensin Odoon toimintaperiaatteet yleisesti. Sen jälkeen työssä perehdytään siihen, miten sovellusta aletaan muokkaamaan ja mitä tarpeita on huomattu myynnin tarvitsevan sovellukseen. Tämän jälkeen kuvataan yhtä suurimmista itse tehdyistä moduuleista, joka luo rajapinnan Odoo-toiminnanohjausjärjestelmän ja Procountor-taloushallinnon ohjelmiston välille. Lopussa käydään läpi tehdyt työt ja arvioidaan toimeksiannon tavoitteiden saavuttamista ja mahdollisia puutteita.

2 TOIMINNANOHJAUSJÄRJESTELMÄT

Toiminnanohjausjärjestelmiä on useita ja niistä merkittävimmät ovat muun muassa SAP, Oracle NetSuite ja Odoo. Toiminnanohjausjärjestelmien yleinen periaate on auttaa yrityksiä hallinnoimaan useita liiketoiminnan osa-alueita yhdellä sovelluksella. Näitä osa-alueita on esimerkiksi taloushallinnointi, henkilöstö- ja asiakashallinnointi eli CRM (Customer relationship management), varastonhallinta ja tuotannonohjaus. (McCue 2022.)

Toiminnanohjausjärjestelmän valinta on hankala päätös, koska tarjolla on useita erilaisia järjestelmiä, joista jokainen tarjoaa hieman erilaisia ominaisuuksia. Esimerkiksi SAP-järjestelmä antaa käyttäjälle laajan vaihtoehdon erilaisia moduuleja, joissa on lähes kaikki toiminnanohjaukseen ja liiketoimintaan vaadittavat vaihtoehdot. Tässä kuitenkin miinuksena on se, että SAP-järjestelmä on kallis vaihtoehto etenkin pienelle yritykselle ja järjestelmän käyttöönotto on hidasta ja monimutkaista, jolloin vaihtoehtona onkin Odoo. Odoon ilmaisversio on avoimeen lähdekoodiin perustuva ja täten täysin muokattavissa ja omiin tarpeisiin räätälöitävissä oleva järjestelmä. (Khalid 2024.)

2.1 Toiminnanohjausjärjestelmien kehitys

Sanaa ERP (enterprise resource planning) käytettiin ensimmäisen kerran 1990-luvulla, mutta ERP-järjestelmiä on ollut jo 1960-luvulla. Tällöin yrityksen alkoivat käyttää niin sanottuja MRP (material requirement planning) -järjestelmiä, jotta voitiin paremmin pitää kirjaa tuotteista, myynnistä ja toimituksesta. Vasta 1990-luvulla yritykset alkoivat ottaa käyttöön ERP-järjestelmiä, kuin millaisina me tiedämme ne nykyään. Yrityksillä oli tuolloin samanlaisia tarpeita pitää kirjaa useista prosesseista, johon tarvittiin yhtä sovellusta, jolla tämä voitaisiin suorittaa. (Genius ERP 2024.)

Nykypäivänä ERP-järjestelmät ovat suositumpia kuin koskaan ennen. Tähän suurena syynä on yritysten suuret tietokannat, johon on välttämätöntä olla oma ERP-järjestelmä ja tarve yhdistää yrityksen useat toimialat yhteen sovellukseen. Lisäksi nykyteknologia sallii esimerkiksi tekoälyn hyödyntämisen yrityksen toiminnanohjauksessa. Tekoälyllä pystyy muun muassa tekemään ennustuksia datamalleista ja suorittamaan automaattisesti toistettavia tehtäviä. Nykyään toiminnanohjausjärjestelmiä otetaan käyttöön myös järjestelmän laajan integrointimahdollisuuden

takia. Yhteen ERP-järjestelemään on nykyään mahdollista yhdistää useita eri sovelluksia kuten taloushallinnon tai tuotannonhallinnan sovelluksia. Useat yritykset ovatkin siirtyneet nykyään pilvipohjaisiin järjestelmiin, jotka tarjoavat sovellukselle joustavuutta. (sama)

2.2 Käyttäjät, toiminnot ja kustannukset

Toiminnanohjausjärjestelmien yleinen periaate on tehostaa yritysten liiketoimintaa. Tämä yksinään on jo suuri syy yrityksille ottaa toiminnanohjausjärjestelmä käyttöön. Toiminnanohjausjärjestelmät yhdistävät yrityksen useat toimialat yhteen sovellukseen, mikä auttaa hallitsemaan eri osa-alueita. Useat toiminnanohjausjärjestelmät ovat tämän lisäksi räätälöitävissä olevia, mikä tarkoittaa, että yritys voi mukauttaa sovellusta omiin tarpeisiin sopivaksi. Käyttötarkoituksia yrityksille on useita ja ne voivat vaihdella työntekijöiden tuntien seurannasta jopa laajaksi kaikki yrityksen toimialueet yhdistäväksi järjestelmäksi. Lisäksi varsinkin isommilla yrityksillä eri sovelluksia voi olla käytössä jo useita. Tämä ei kuitenkaan ole ongelma toiminnanohjausjärjestelmää hankittaessa, koska mahdollisuuksien mukaan jo aikaisemmin käytössä olleet sovellukset tai ohjelmointirajapinnat ovat yhdistettävissä ERP-järjestelmään. (QAD 2024.)

Järjestelmää pääsevät organisaatiosta riippuen käyttämään kaikki sovellusta tarvitsevat. Toiminnanohjausjärjestelmissä on yleistä, että käyttäjäoikeuksia on mahdollista muuttaa tarpeiden mukaan ja esimerkiksi myynnin tehtävistä vastaavilla on käytössään tähän tarvittavat moduulit, mutta ei muuta. Tämä on myös hyvä asia myyjän näkökulmasta. On tärkeää, että sovellus ei mene sekavaksi sen takia, että on paljon erilaisia toimintoja, joita ei työssään tarvitse. Käyttäjäoikeuksien muuttaminen jää usein ohjelmistokehittäjälle tai yrityksestä vastaavalle henkilölle, jolla säilyy oikeus kaikkiin järjestelmän toimintoihin. (sama)

Toiminnanohjausjärjestelmät jaetaan usein toimintojen mukaan. Toimintoja on paljon ja vaihtelee järjestelmästä toiseen paljon. Yleisimmät toiminnanohjausjärjestelmien osat ja toiminnot ovat asiakkuudenhallinta (CRM), varastonhallinta, tuotannonohjaus, taloushallinto ja henkilöstöhallinto (HRM). Asiakkuudenhallinta toiminnanohjausjärjestelmissä kattaa yrityksen asiakastietojärjestelmän. Esimerkiksi Odoo-järjestelmässä CRM on oma moduulinsa ja auttaa yritystä pitämään asiakastiedot yhdellä välilehdellä, josta niitä on helppo jaotella esimerkiksi asiakkuuden mukaan. Asiakkuudenhallinnalla saadaan kaikki asiakkaan tarpeelliset tiedot helposti asiakastietoihin. Näistä tiedoista yleisimmät ovat perinteisesti nimi, puhelinnumero, sähköposti ja

postiosoite, mutta asiakastietoihin on mahdollista liittää niin asiakkaan omistamat tuotteet, laskut kuin asiakaspalvelussa ilmi tulleet tiedot. Asiakkuudenhallintaa on myös yrityksen tarpeiden mukaan mahdollista räätälöidä. Ohjelmointirajapintoja voidaan käyttää tässäkin hyödyksi esimerkiksi lähettämällä toiminnanohjausjärjestelmälle dataa asiakkaan omistamista tuotteista kuten paikkatiedot, akun tila tai laitteen toimintatila. (Tipalti 2024.)

Varastonhallinta ja tuotannonohjaus ovat kaksi samantapaista osa-aluetta, joita käytetään usein yrityksissä, joissa tuotetaan tavaroita tai palveluita. Varastonhallinta on tärkeää ja mahdollistaa helpon inventaarion ylläpitämisen. Tuotteille voidaan järjestelmässä esimerkiksi jakaa tuotenumerot ja viivakoodit. Näiden avulla tuotteita on helppo seurata niin valmistuksesta aina asiakkaalle asti. Tässä myös hyötynä mahdolliset asiakaspalvelutapaukset, jossa kyseessä on tietty tuotenumerolla haettava tuote, jota asiakaspalvelu koskee. Tuotannonohjaus auttaa yritystä pitämään kirjaa tuotteiden valmistuksen alkuvaiheista aina valmiin tuotteen lähetykseen asti. Tuotannonohjaus on monivaiheinen prosessi, jossa yhdistyvät esimerkiksi teknologian yrityksissä laitteiden komponenttien hankkiminen välittäjiltä, tuotteiden valmistaminen ja lähetykset. Tuotannonohjauksessa on myös tärkeää, että voidaan ylläpitää tietoa varastotasoista. Tuotannonohjaukseen on muun muassa mahdollista asettaa rajat tuotteille, jolloin tuotteen varastotilan alittuessa tuotetta saadaan automaattisesti tilattua lisää välittäjiltä. (Odoo 2024b.)

Taloushallinto nimensä mukaisesti auttaa yritystä hallinnoimaan yrityksen taloutta. Toiminnanohjausjärjestelmistä on usein mahdollista suoraan luoda lähetteitä, tarjouksia ja laskuja. Kun taloushallintoon otetaan mukaan asiakkuudenhallinta ja varastonhallinta, on laskujen tekeminen ja lähettäminen erittäin vaivatonta. Taloushallintoa voidaan kuitenkin taas räätälöidä yritykselle sopivaksi ja esimerkiksi liittää jo olemassa oleva taloushallinnon sovellus toimimaan yhdessä toiminnanohjausjärjestelmän kanssa ohjelmointirajapinnan avulla. Suurena hyötynä tässä on esimerkiksi rahan kulun seuranta. Erillisellä taloushallinnon sovelluksella voidaan seurata esimerkiksi laskujen tilaa ja tieto laskusta voidaan lähettää ohjelmointirajapinnalla toiminnanohjausjärjestelmässä nähtäväksi. (QAD 2024.)

Viimeinen mainitsemisen arvioinen ja yritysten suosima toiminto toiminnanohjausjärjestelmissä on henkilöstöhallinto (HRM eli human resources management). Koska toiminnanohjausjärjestelmä on yritykselle keskeinen sovellus, josta löytyvät miltei kaikki osa-alueet, on henkilöstöhallinto myös hyvä liittää järjestelmään. Henkilöstöhallinnossa voidaan pitää kirjaa työntekijöiden tehdyistä

työtunneista ja käytetyistä lomapäivistä ja hallita työntekijöiden työvuoroja. Henkilöstöhallintoon voidaan liittää myös yrityksen sisäiseen viestintään suunnattu ohjelma. (Jackley 2023.)

Näitä kaikkia toimintoja tarvitaan useimmissa yrityksissä. Yrityksen ei kuitenkaan tarvitse nykyään olla suuri, että voi ottaa käyttöön toiminnanohjausjärjestelmän. Toiminnanohjausjärjestelmiä ottavat käyttöön niin pk-yritykset, suuret yritykset kuin muutkin toimialat, kuten vähittäiskauppa tai terveydenhuolto. Järjestelmiä on nykyään sekä ilmaisia, että lisenssimaksulla ostettavia. Ilmaiset ovat usein avoimen lähdekoodin sovelluksia, mutta ne eivät välttämättä tarvitse ollenkaan ohjelmistokehittäjätietämystä. Kuitenkin, jos järjestelmän haluaa valmiina, on toiminnanohjausjärjestelmissä versioita, joissa hinta koostuu esimerkiksi valittujen moduulien ja sovelluksen käyttäjämäärän perusteella. Usein lisenssimaksullisissa versoissa hyötynä on myös asiakastuki ja ohjelmistokehittäjä, joka auttaa yritystä räätälöimään järjestelmän toiminnalleen sopivaksi. Muita kustannuksia, joita yrityksille voi syntyä toiminnanohjausjärjestelmästä, ovat esimerkiksi järjestelmän ylläpitokustannukset. Riippumatta siitä onko sovellus lisenssimaksullinen vai ei, on silti mahdollista, että yrityksellä on vähintään yksi henkilö palkattu pitämään järjestelmää yllä. Myös esimerkiksi laitteet voivat aiheuttaa ylimääräisiä kustannuksia yritykselle. Tämä on kuitenkin harvinaisempaa ja usein ongelmana vain suuremmilla yrityksillä, joissa dataa lähettäviä laitteita on useita ja yhteenlaskettu kulutus on jo merkittävä. (Tipalti 2024.)

2.3 Integraatiot

Toiminnanohjausjärjestelmissä on tärkeää sovelluksen joustavuus ja integroitavuus. Tästä syystä toiminnanohjausjärjestelmiin usein liitetäänkin ohjelmointirajapintoja eli application programming interface (API). Nämä rajapinnat mahdollistavat tiedonsiirron esimerkiksi kahden eri sovelluksen välillä. Onkin välttämätöntä, että ohjelmointirajapintoja käytetään toiminnanohjausjärjestelmissä, joihin yhdistetään useita eri toimialan sovelluksia ja toiminnallisuuksia. Näitä toiminnanohjausjärjestelmissä käytettyjä sovelluksia ovat esimerkiksi jo aiemmin mainitut talous- ja tuotannonhallinon sovellukset. Näissä sovelluksissa onkin tärkeää, että informaatio päivittyy reaaliajassa. Usein ERP-järjestelmiin tehdyt ohjelmistorajapinnat päivittyvät reaaliajassa ja täten mahdollistavat useiden eri toiminnallisuuksien hyödyntämisen järjestelmässä. Ohjelmointirajapintaa hyödyntävästä sovelluksesta mainittakoon esimerkkinä sovellus, joka lähettää taloushallinnon sovellukseen ERP-järjestelmässä tehdyt laskut ja päivittää ERP-järjestelmää maksetuista laskuista, jotka ovat taloushallinon sovelluksessa. (Gitlin 2024.)

Toiminnanohjausjärjestelmien integraatioista puhuttaessa esiin tulee usein modulaarisuus. Modulaariset ERP-järjestelmät mahdollistavat helpon tavan mukauttaa järjestelmää yrityksen omien tarpeiden mukaiseksi. Modulaarisuus tarkoittaa sitä, että ERP-järjestelmän käyttäjä voi itse valita toiminnallisuudet, joita haluaa sovellukseen. Tämä mahdollistaa myös tulevaisuudessa ERP-järjestelmän helpon mukauttamisen ja uusien moduulien asentamisen. Modulaarinen rakenne myös helpottaa ohjelmointirajapintojen kanssa muiden sovellusten yhdistämistä, koska moduulit toimivat itsenäisesti, mutta voivat vaihtaa tietoa muiden moduulien kanssa tarpeen mukaan. Modulaarisissa toiminnanohjausjärjestelmissä hyötynä on myös niiden nopea käyttöönotto. Näissä usein voidaan helposti asentaa tarvittavat moduulit ilman aiempaa kokemusta ERP-järjestelmistä tai ohjelmistokehityksestä. Modulaarisuus parantaa myös niin sovelluksen käyttötehokkuutta kuin kustannustehokkuutta. Käyttäjän ei tarvitse ottaa tarpeettomiksi kokemiaan moduuleja käyttöön, jolloin säästetään mahdollisissa kustannuksissa. (Vault-ERP 2024.)

2.4 Odoo toiminnanohjausjärjestelmänä

Odoo on erittäin tehokas ja käyttäjäystävällinen toiminnanohjausjärjestelmä. Modulaarisuutensa ja mukautettavuutensa ansiosta Odoo on vakiinnuttanut paikkansa muiden parhaiden toiminnanohjausjärjestelmien joukossa. Odoon käyttöönotto on tehty helpoksi niin pienelle organisaatiolle kuin myös isommalle yritykselle. Sovelluksen modulaarisuuden ansiosta toiminnanohjausjärjestelmä voidaan skaalata toiminnalle sopivaksi ja organisaation kasvaessa sovellusta voidaan entisestään jatkaa uusilla toiminnallisuuksilla. Toiminnanohjausjärjestelmää on myös mahdollista muokata ilman erillisiä lisenssikustannuksia. Tähän vaaditaan hieman sovelluskehittämistä, mutta tällä voidaan mukauttaa toiminnanohjausjärjestelmää entisestään yritykselle sopivaksi. Jos esimerkiksi Odoon valmiista moduuleista tai Odoon Apps Store -moduuleista ei löydy sovellukselle sopivaa, on järjestelmää mahdollista itse mukauttaa toiminnalle sopivaksi. Tähän lukeutuu uusien moduulien tekeminen itse, kenttien muuttaminen tai esimerkiksi valmiina olevien ohjelmointirajapintojen käyttö Odoo-järjestelmässä.

Odoo perustuu avoimeen lähdekoodiin ja sen lähdekoodi on vapaasti saatavilla ja muokattavissa. Kaikkea ei kuitenkaan tarvitse itse muokata ja Odoo tarjoaa tähän paljon valmiita moduuleita, jotka ovat helposti asennettavissa omaan sovellukseen samasta järjestelmästä. Tämän lisäksi Odoolla on oma Odoo Apps Store. Täällä muut käyttäjät voivat jakaa tai myydä omaa ohjelmistoaan muille

käyttäjille. Odoo Apps Storessa on vapaasti asennettavia moduuleita tarjolla jopa yli 45 000 kappaletta, joista Odoo 16 -käyttöjärjestelmälle sopivia on yli 19 000 kappaletta. Odoon kustomoinnissa on myös vahvasti läsnä erilaiset keskustelufoorumit kuten Odoon oma foorumi, Odoo Help – Forum, missä muut sovelluskehittäjät voivat auttaa toisia, kysyä kysymyksiä tai jakaa ideoitaan. (Odoo 2024c.)

Odooseen tulee joka vuosi uusi järjestelmäpäivitys ja tällä hetkellä Odoosta on julkaistuna versio 17.0. Opinnäytetyössä käytetty versio on lokakuussa 2022 julkaistu versio 16.0. Version lisäksi Odoosta on valittavissa myös Community- tai Enterprise-versio. Suurimmat erot näiden kahden version välillä on se, että Enterprise on kaupallinen versio Odoosta, kun taas Community-versio on ilmainen. Enterprise-versio kattaa hieman enemmän valmiita moduuleita ja takaa laajemman toiminnallisuuden verrattuna Community-versioon. Lisäksi Enterprise-versiolla on yritystuki, joka tarjoaa käyttäjille päivityksiä, korjauksia ja muita teknisiä apuja. Community-versio sen sijaan turvautuu yhteisötukeen, eli yhteisön jäsenten jakamaan tietoon ja avunantoon. (sama)

2.5 Odoon teknologiat

Odoo käyttää todella useita erilaisia teknologioita toimiakseen. Näitä teknologioita ovat esimerkiksi erilaiset ohjelmointikielet, kirjastot, verkkopalvelin, sivuston elementit ja tietokannat. Odoo käyttää sovelluksessaan pääasiassa kolmea eri ohjelmointikieltä. Nämä ovat Python, XML ja JavaScript. Python-ohjelmointikieltä käytetään esimerkiksi moduulien toimintojen ohjelmoimiseen. XML-kieltä käytetään sen sijaan Odoon näkymien ohjelmoimiseen. Sovelluskehittäjät, jotka tekevät Odooseen moduuleja joutuu harvemmin kirjoittamaan itse JavaScript-kieltä, koska sitä käytetään pääasiassa vain Odoon dynaamisiin toimintoihin. Tyylisivukielenä Odoossa käytetään CSS:ää ja erityisemmin SCSS (Sassy CSS) -komentosarjakieltä. Odoo pääasiassa käyttää SCSS-komentosarjakieltä web-näkymien tyylin muuttamiseen ja CSS-tyylisivukieltä käytetään vain lisäominaisuuksiin kuten muuttujiin. (W3Techs 2024.)

Odoo käyttää tietokantanaan avoimen lähdekoodiin perustuvaa PostgreSQL-relaatiotietokantaa. PostgreSQL toimii Odoon taustalla olevana tietokantana, joka tallentaa kaikki Odoossa käytetyt tiedot. Näitä tietoja ovat esimerkiksi asiakastiedot, tuotetiedot, tilaukset tai laskut. Odoon kannalta onkin tärkeää, että PostgreSQL on relaatiotietokanta. PostgreSQL tallentaa tiedot taulukkona ja voi näin ollen helposti käyttää samoja tietoja muissa tarpeissa. Tämä on Odoon tilanteessa hyödyllistä,

kun tietokanta mahdollistaa eri tietojen välisen suhteen. Esimerkkinä tästä on laskut ja niihin liittyvien tuotteiden välinen suhde. (PostgreSQL 2024.)

Odoo on myös modulaarinen toiminnanohjausjärjestelmä. Kuten jo aiemmin on mainittu, modulaariset toiminnanohjausjärjestelmät ovat erittäin helposti kustomoitavissa ja nopeita ottaa käyttöön. Odoo hyödyntää tämän lisäksi omaa object relational mapping (ORM) -rakennetta. Tämän ansiosta modulaarisuus on Odoossa entistä hyödyllisempää. ORM toimii Odoon moduulien ja tietokannan välissä niin sanotusti siltana ja mahdollistaa tietokannassa olevien tietojen hyödyntämisen moduuleissa. (Cybrosys Technologies 2019.)

2.6 Avoin lähdekoodi ja sen merkitys

Avoimessa lähdekoodissa on käyttäjällä vapaus tarkastella, muuttaa ja jakaa sovelluksen alkuperäistä ohjelmistoa. Tästä on hyötyä yrityksille esimerkiksi siten, että sovelluskehittäjät eivät rajoitu ainoastaan yrityksen omiin työntekijöihin vaan jokainen voi muokata sovellustaan haluamallaan tavalla. Tästä syystä avoin lähdekoodi on osalle yrityksiä suotuisaa mahdollistaen suuremman yhteisön osallistumisen. Eri osallistujat voivat tehdä sovellukseen esimerkiksi muutoksia, parannuksia tai lisätoimintoja, jotka nopeuttavat sovelluksen laajentumista. Suuren osallistumisen myötä sovellukselle saadaan myös uusia näkökulmia ja erilaisia lopputuloksia. Lisäksi ongelmatilanteissa voidaan turvautua yhteisön tukeen vähentäen riippuvuutta yksittäisestä ohjelmistokehittäjästä. Yritykset hyötyvät avoimesta lähdekoodista myös vähenevinä kustannuksina. Useimmiten avoimen lähdekoodin sovellukset ovat ilmaisia ja näin säästetään mahdollisissa lisenssikustannuksissa. Usein nämä sovellukset käyttävät myös vähemmän laitteistotehoa, mikä on taas yksi aspekti säästämiselle. (Cemazar 2022.)

Avoimen lähdekoodin käyttö myös parantaa sovellusten turvallisuutta. Yhteisön voimin sovelluksen koodia on tarkastelemassa useampi ihminen, joten sovelluksen haavoittuvuudet havaitaan helpommin. Haavoittuvuudet havaitaan myös nopeammin ja tästä syystä ne saadaan korjattua nopeammalla aikataululla, mikä jättää vähemmän aikaa hyödyntää koodin haavoittuvuutta. (sama)

Yrityksille hyötynä avoimesta lähdekoodista on myös sen testattavuus ja mukauttaminen. Avoimen lähdekoodin sovellusta voidaan räätälöidä sopivaksi ja testata, onko ratkaisu yritykselle sopiva. Vasta yrityksen laajentuessa se voi siirtyä kaupallisiin versioihin, jotta uudet tarpeet voitaisiin

kattaa. Näin ollen avoimen lähdekoodin sovellukset eivät ole vain kustannustehokkaita, vaan myös auttavat yrityksiä valitsemaan tarpeiden mukaisen ratkaisun ja antavat mahdollisuuden laajentaa myöhemmin tarpeiden mukaan. (sama)

3 ODOON TOIMINNALLISUUS JA KÄYTTÖÖNOTTO

Odoon käyttöönotto vaatii aluksi perehtymistä ohjelmistoympäristöön. Asetuksia on todella paljon ja ilman aiempaa kokemusta Odoon käyttöönotto voi olla aluksi haastavaa. Haasteita voi tulla aluksi eritoten sovelluskehittäjille, joille avautuu myös ohjelmistokehittäjän asetukset. Asetukset ei ole ainoa asia, joita Odoossa täytyy alussa muokata. Riippuen yrityksen tarpeista voi olla tarpeen lisätä esimerkiksi muita käyttäjiä, lisätä asiakastietoja tai asentaa uusia moduuleita.

Odoon Community-version ensiasennuksessa valmiiksi asennettuja moduuleita ovat esimerkiksi CRM, Invoicing ja Sales. Näiden moduulien avulla Odoossa saadaan jo yrityksen asiakastiedot ja laskutus hoidettua, mutta useissa tapauksissa liiketoiminnan vuoksi tarvitaan lisää moduuleita.

3.1 Odoon käyttöönotto

Odoon käyttöönotto alkaa sovelluksen asentamisella. Odoon lataussivulla (https://www.odoo.com/page/download) on vaihtoehdot aina uusimmasta versiosta 17.0 hieman vanhempaan 15.0-versioon. Lisäksi kaikista versiovaihtoehdoista on olemassa latausvaihtoehdot niin Community- kuin Enterprise-versioille. Odoosta on myös olemassa Online-versio, johon saa ilmaiseksi valita kaksi moduulivaihtoehtoa tai maksullisen vaihtoehdon, jossa hinta määräytyy valittujen moduulien ja sovelluksen käyttäjien määrän mukaan. Näistä versioista on myös hyötyä siten, ettei sovellusta tarvitse itse käyttää omalla palvelimella vaan Odoo tarjoaa tähän verkkopalvelun, jossa Odoo itse vastaa sovelluksen toimivuudesta online-ympäristössä. (Odoo 2024a.)

Latauksen jälkeen itse asennusvaiheet vaihtelevat käytetyn käyttöjärjestelmän mukaan, mutta yleisesti ottaen se sisältää ohjelmiston lataamisen, purkamisen ja tarvittavien riippuvuuksien asentamisen. Näitä riippuvuuksia sovelluksella on esimerkiksi PostgreSQL, joka voidaan valita asennettavaksi Odoon asennuksen yhteydessä. Windows-asennuksessa Odoo toimii verkko-osoitteessa https://localhost:8069 ja se voidaan käynnistää ja pysäyttää Windowsin palvelut-asetussivulla. Asennuksen jälkeen Odooseen usein tarvitsee tehdä välttämättömiä peruskonfigurointeja.

3.2 Käyttäjähallinta ja käyttäjäoikeudet

Kuten kaikissa toiminnanohjausjärjestelmissä myös Odoossa käyttäjähallinta ja käyttöoikeudet ovat tärkeä osa järjestelmää. Odoossa pystyy asetuksissa niin lisäämään uusia käyttäjiä, poistamaan olemassa olevia kuin vaihtamaan käyttäjäoikeuksia. Käyttäjäoikeuksien vaihtaminen yleensä tapahtuu organisaation osa-alueiden tasolla. Tämä tarkoittaa sitä, että käyttöoikeudet määräytyvät sen mukaan, millä organisaation osa-alueella käyttäjä on. Myynti ei esimerkiksi tarvitse kaikkia käyttöoikeuksia vaan tarpeiden mukaan myynnille voi riittää vain CRM- ja Invoicing -moduulit. Niin ikään ohjelmistokehittäjillä säilyy täydet oikeudet järjestelmään, jotta tarvittavat muutokset voidaan silti tehdä. Tällainen lähestymistapa on myös hyödyllistä myynnin käyttäjien näkökulmasta, koska Odoo voi olla sekava järjestelmä, jos myynnin näkymässä olisi myös tarpeettomia moduuleja kuten Manufacturing, joka vastaa tuotteiden valmistuksesta. Käyttäjäoikeuksia on mahdollista vaihtaa Odoossa myös erittäin spesifisti. Järjestelmässä on käyttäjäkohtaisesti mahdollista valita käyttöoikeudet niin moduuleille kuin yksittäisille kentille. Nämä käyttäjäoikeudet vaihtelevat kaikilla read-, write-, create- ja delete-vaihtoehdoittain. Tämä on harvinaisempi tapa muuttaa käyttäjäoikeuksia, mutta se voi tarpeen tullen osoittautua hyödylliseksi. (Odoo docs 2024a.)

Järjestelmän ylläpitäjät myös vastaavan käyttäjähallinnasta. Odoossa käyttäjät usein joko lisätään toiminnanohjausjärjestelmän ylläpitäjän toimesta tai kutsutaan sähköpostiviestiin liitetyllä linkillä. Tässä on hyötynä se, että järjestelmän ylläpito pysyy kontrollissa ylläpitäjillä ja heidän on mahdollista pitää käyttäjäoikeudet ajan tasalla. Lisäksi etuna on turvallisuus. On turvallisempaa, että käyttäjien lisääminen ja niihin kuuluvat käyttäjäoikeudet pidetään järjestelmän ylläpitäjillä, ja että ulkopuolisilla henkilöillä ei ole mahdollisuutta luoda käyttäjää järjestelmään itse. Odoo useimmiten kuitenkin toimii web-selaimella ja järjestelmää pääsee käyttämään yksinkertaisesti sähköpostilla ja salasanalla. (sama)

3.3 Asiakas- ja tuotetiedot

Asiakas- ja tuotetiedot ovat yleisimmät tiedot, joita yritykset, joilla on myytäviä tuotteita, lisäävät ja joita toiminnanohjausjärjestelmistä löytyy. Nämä tiedot luovat hyvän perustan myynnin toiminnanohjausjärjestelmälle antaen mahdollisuuden käyttää samoja kenttiä useammassa eri tarkoituksessa. Tässä on kyseessä relaatiotietokanta, jossa asiakkaan tai tuotteiden tietoja voidaan

käyttää eri moduuleissa ja eri käyttötarkoituksissa. Myös Odoo-toiminnanohjausjärjestelmään nämä tiedot on mahdollista lisätä. Asiakastiedot voidaan luoda Contacts-moduulissa ja tuotetiedot vastaavasti Inventory-moduulissa. On todennäköistä, että yrityksellä, joka harkitsee Odoo-ohjelmiston käyttöönottoa, on jo olemassa olevia asiakastietoja. Näitä ei kuitenkaan tarvitse yksi kerrallaan lisätä järjestelmään vaan Odoosta löytyy Contacts-moduulista mahdollisuus tuoda kaikki asiakastiedot kerralla järjestelmään esimerkiksi Excel-tiedostomuodossa. Excel-tiedostoon on myös kätevää lisätä asiakastiedoille jo Odoosta löytyvät kentät, kuten puhelinnumerot tai sähköpostit ja näin ollen tuotua kaikki tarvittavat tiedot kerralla. (Cybrosys Technologies 2024a.)

Kuten jo aiemmin on mainittu, asiakas- ja tuotetietojen lisääminen Odoo-järjestelmään on tärkeää myös järjestelmän relaatiotietokannan takia. Yleisesti ottaen monet myynnin- ja kaupanalan yritykset haluavat ottaa käyttöön toiminnanohjausjärjestelmässä myös muita moduuleja kuten Invoicing, joka vastaa laskutuksesta ja Manufacturing, joka vastaa tuotteiden valmistusprosesseista. On tärkeää, että järjestelmästä löytyy silloin relaatiotietokanta, joka pystyy käsittelemään tietoja eri moduuleissa ja eri käyttötarkoituksissa. Asiakas- ja tuotetietoja on myös mahdollista hyödyntää itse kehitetyissä moduuleissa. Odoossa näiden tietojen käyttäminen onnistuu, kun moduulia ohjelmoidessa moduuliin lisätään niiden tietojen omaavat modelit eli mallit. Asiakas- ja tuotetietojen lisääminen myös mahdollistaa niiden hyödyntämisen eri ohjelmointirajapintojen kanssa. (sama)

3.4 Moduulien lisääminen ja poistaminen

Odoon modulaarisuuden vuoksi moduulien lisääminen ja poistaminen on ohjelmistokehittäjälle miltei päivittäinen toimenpide. Odoon moduulien lisäämiselle on useita erilaisia tapoja ja Odoo tarjoaa moduulin asentamiselle myös helpon tavan itse järjestelmässä. Järjestelmässä olevassa Apps-moduulissa on laaja valikoima erilaisia moduuleja, jossa käyttäjä voi asentaa moduulin hyvin yksinkertaisesti. Kuitenkin, kun halutaan asentaa moduuleja, joita on itse ohjelmoitu tai jotka on ladattu kolmannen osapuolen kehittäjiltä Odoo Apps Storesta, ei asentaminen enää onnistu järjestelmän sisällä. Tästä poikkeuksena on moduulit, jotka sisältävät vain XML-tiedostomuotoja ja välttämättömän manifest.py-tiedoston, jolloin moduuli voidaan asentaa Odoon Apps-moduulissa import module -toiminnolla. Tämä toiminto kuitenkin hyväksyy vain datamoduuleja, jotka sisältävät XML-tiedostomuotoja tai staattisia resursseja. (Cybrosys Technologies 2023.)

Moduulit, jotka sisältävät enemmän tiedostomuotoja, täytyy tuoda järjestelmään hieman eri tavoin. Paikallisesti tietokoneella toimivissa järjestelmissä moduuleja voidaan asentaa esimerkiksi suoraan sovelluksen hakemistoon. Tässä ei tarvitse tietää muuta kuin hakemistopolku, jonne moduuli täytyy viedä, ja käynnistää järjestelmä uudestaan. Järjestelmän uudelleenkäynnistyksen jälkeen asennettu moduuli löytyy tutusta sovelluksen Apps-osiosta, josta moduuli voidaan ottaa järjestelmään käyttöön. Tämän jälkeen Odoo pystyy automaattisesti kokoamaan ja suorittamaan moduulit järjestelmässä, edellyttäen että ne on ohjelmoitu asianmukaisesti ja järjestelmässä ei ole muita haitallisia tekijöitä. Kolmansien osapuolien moduulien asentaminen paikallisesti tietokoneella toimivaan järjestelmään toimii samoin. Tässä erona on vain, että moduuli tulee itse ladata Odoon Apps Store -sivulta. Linux-pohjaisissa ympäristöissä moduulien asentamiselle on useita vaihtoehtoja. Toimiva vaihtoehto Linux-pohjaisen ympäristön moduulien asentamiselle on komentorivin kautta asentaminen, kuten nähdään kuvassa 1. Tällöin tietokoneen julkinen SSH-avain lisätään Odoon valtuutettuihin avaimiin, jolloin käyttäjällä on oikeus päästä Ubuntu-ympäristössä Odoon hakemistoon. Moduulit voidaan lisätä palvelimelle, kun tiedostot viedään oikeaan kohtaan hakemistossa. Tämän jälkeen moduulit taas näkyvät Odoo järjestelmän Apps-osiossa, josta ne voidaan ottaa järjestelmään käyttöön. (sama)

KUVA 1. Moduulin asentaminen komentorivin kautta.

Moduulien poistaminen tapahtuu käytännössä käänteisessä järjestyksessä verrattuna asentamiseen. Odoon Apps-moduulissa halutun moduulin kohdalla on uninstall-nappi, josta moduuli voidaan ottaa pois käytöstä. Tämä ei kuitenkaan vielä poista moduulia lopullisesti, jos se on viety järjestelmän hakemistoon. Jos moduuli halutaan poistaa kokonaan järjestelmästä, täytyy se poistaa kokonaan järjestelmän hakemistosta.

3.5 Hyödylliset konfiguroinnit

Suurten toiminnanohjausjärjestelmien mukana tulee myös suuri määrä konfiguroitavia asetuksia, raportteja tai Odoon tapauksessa moduuleja. Usein näiden konfigurointi ei ole välttämätöntä, mutta

se vähintäänkin helpottaa tai nopeuttaa järjestelmän käyttöä. Odoon tapauksessa jopa yksittäisten asetusten konfiguroinnilla voi olla merkittäviä hyötyjä. Tästä on esimerkkinä asiakkaan kielen muuttaminen. Asetuksen muuttaminen asiakaskohtaisesti kuulostaa pieneltä muutokselta. Odoo kuitenkin pystyy tämän asetuksen avulla muuttamaan suomeksi tehdyt laskut suoraan asiakkaan kieleksi asetetulle kielelle edellyttäen, että järjestelmään on asennettu kyseinen kieli asetuksista. Odoo pystyy muuttamaan laskuun myös valuutan suoraan asiakkaalle asetetun asuinmaan perusteella.

Muita konfiguroitavia asioita, joita on mahdollista tehdä Odoossa ja joita useat eivät edes tule ajatelleeksi, on yrityksen keskitetty sähköposti ja yrityksen sisäinen viestintä. Nämä molemmat on mahdollista suorittaa Odoo-järjestelmässä suoraan. Odoo antaa mahdollisuuden lähettää ja vastaanottaa sähköpostiviestejä suoraan sovellukseen keskitetystä sähköpostista. Tämä onnistuu, kun IMAP-protokolla otetaan sähköpostiviesteissä käyttöön, jolloin käyttäjän asettama sähköposti toimii järjestelmässä keskitettynä sähköpostina. Odoo tarjoaa myös mahdollisuuden yrityksen sisäiseen viestintään. Odoon Discuss-moduuli toimii samalla tavalla kuin useat kilpailevat organisaation sisäiset viestintäjärjestelmät. Moduulissa voidaan luoda keskusteluja, joihin voidaan liittää halutut käyttäjät. Myös yksityisviestien lähettäminen on sovelluksessa mahdollista.

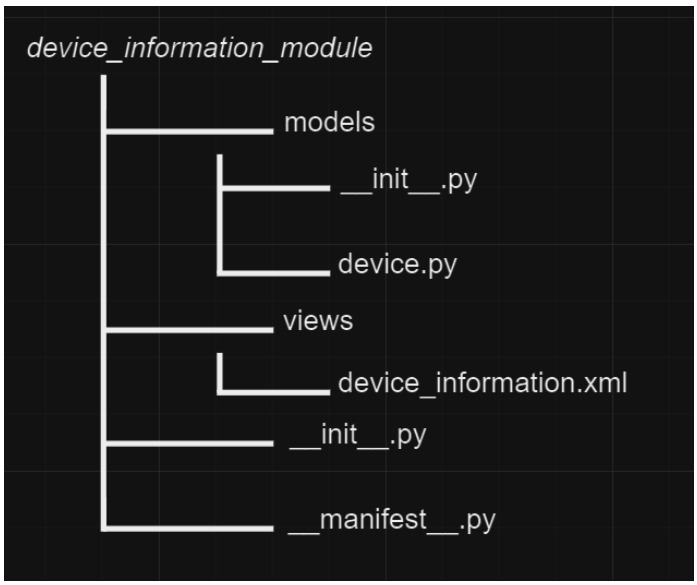
Odoossa raporttien, lomakkeiden ja käyttöliittymän konfigurointi on tehty mahdolliseksi ja auttamaan järjestelmän räätälöinnissä liiketoiminnan tarpeiden mukaiseksi. Odoon modulaarisuuden ansiosta näitä asetuksia on mahdollista muokata suoraan käyttöliittymästä. Esimerkiksi laskupohjaa muokkaamalla voidaan siirtää laskussa esitettäviä kenttiä haluttuihin kohtiin. Laskupohjan muutoksiin kuuluu myös viitenumeroiden muokkaaminen. Usein yrityksillä on laskuja jo ennen toiminnanohjausjärjestelmän käyttöönottoa. Näissä viitenumerot suurenevat yrityksen haluamalla tavalla, mutta viitenumerot on myös mahdollista muuttaa laskupohjaan sopivaksi. Muita laskupohjaan tulevia muutoksia ovat esimerkiksi yrityksen logo, yrityksen yhteystiedot tai laskun viivakoodi.

3.6 Moduulin rakenne

Toiminnanohjausjärjestelmiä ohjelmoidessa on tärkeää, että moduulin rakenne pysyy oikeana. Moduulissa ei tarvita suuria määriä tiedostoja, jotta moduuli toimisi, mutta moduulissa tulee kuitenkin olla tarvittavat tiedostot. Näihin tarvittaviin tiedostoihin käytännössä kuuluu vain tiedostot

__manifest__.py ja __init__.py. Manifest-tiedosto on Python-tiedosto ja pitää sisällään moduulin metadatan. Tässä tiedostossa esimerkiksi luokitellaan moduuli, annetaan moduulin nimi ja moduulin tekijä. Manifest-tiedostoon kirjoitetut tiedot ovat ne, jotka näkyvät Odoon käyttöliittymässä, kun moduulia aletaan asentamaan. Init-tiedosto täytyy olla myös samassa hakemistossa, kuin moduulin muut Python-tiedostot. Init-tiedosto pitää sisällään moduulin import-tiedot eli tiedot, jossa kerrotaan moduulille mitä tiedostoja sen pitää ladata, jotta moduulissa oleva koodi voi toimia.

Pelkillä manifest- ja init -tiedostoilla ei kuitenkaan paljoa saada aikaan. Muita tiedostoja, joita moduulit yleensä pitää sisällään ovat models, views, ja static. Models-kansio pitää sisällään moduulin toiminnalliset tiedostot. Nämä tiedostot ovat Python-tiedostoja ja yleensä moduulin suurimpia tiedostoja. Python on Odoon yleisin ohjelmointikieli, joten näissä tiedostoissa määritetään moduulin kaikki toiminnallisuus. Views-tiedosto on toinen erittäin yleinen tiedosto moduuleissa. Views-kansio pitää sisällään vähintäänkin yhden XML-tiedoston, joka Odoossa toimii käyttöliittymän merkintäkielenä. XML-tiedostoissa määritetään kaikki käyttöliittymässä näkyvä tieto, kuten missä yksittäinen tieto näkyy. Static-tiedosto pitää sisällään moduulin staattiset tiedostot. Näitä voi olla esimerkiksi moduulissa käytettävät kuvat ja JavaScript-tiedostot. Tällä moduulin rakenteella saadaan jo paljon aikaseksi ja usein ohjelmoijalla ei edes tarvitse muita tiedostoja moduuleissa. Kuvassa 2 nähdään tyypillinen moduulin rakenne. (Cybrosys Technologies 2024b.)



KUVA 2. Moduulin rakenne.

3.7 Odoon elementit

Odoossa on erittäin paljon erilaisia elementtejä. Elementit yhdessä luovat Odoon järjestelmään toiminnallisuuden ja käyttöliittymän. Odoossa elementtejä voidaan tarkastella ja muuttaa suoraan käyttöliittymässä. Tämä ei kuitenkaan ole suotavaa, koska Odoon järjestelmäpäivityksissä suoraan järjestelmässä tehdyt muutokset elementteihin voivat aiheuttaa ongelmia yhteensopivuuden kanssa. Sen sijaan elementtejä voidaan muokata ja luoda uusien moduulien avulla. Moduuleilla voidaan myös luoda uusia yhteyksiä muihin elementteihin. Näitä elementtejä on esimerkiksi fields, models ja views.

Odoossa jokainen kenttä kuuluu fields-elementtiin. Näitä kenttiä ovat esimerkiksi asiakastiedoissa asiakkaan nimi, puhelinnumero ja osoitetiedot. Fields-kenttiä on mahdollista muokata luomalla uusia moduuleja. Moduuleissa tietyssä näkymässä olevia kenttiä voidaan poistaa ja luoda lisää. Olemassa olevien kenttien muokkaaminen ei ole suotavaa, koska tämäkin voi aiheuttaa tulevien päivitysten kanssa yhteensopivuusongelmia. On siis suotavaa, että kenttien kanssa ainoastaan luodaan uusia tai poistetaan kenttä näkyvistä käyttöliittymästä. Tämä ei kuitenkaan tarkoita, että kenttä poistetaan kokonaan tietokannasta vaan ainoastaan, että sitä ei enää ole käyttöliittymässä. Ohjelmoinnilla on kuitenkin mahdollista muuttaa olemassa olevien kenttien tietoja siten, että niiden tiedot näkyvät uudessa kentässä. Tästä on hyötyä siinä, että voidaan tehdä uusi kenttä eri kenttätyypillä ilman, että alkuperäistä joudutaan poistamaan. Odoo tukee useita erilaisia kenttätyyppejä, kuten char, integer, float, date, many2one, one2many ja many2many. Char-kentässä tieto tallennetaan aakkosnumeerisena string-datatyyppinä. Integer-kentässä tieto tallennetaan kokonaislukuna. Float-kentässä tieto tallennetaan desimaalilukuna. Date-kentässä tieto tallennetaan päivämääränä. Many2one-, one2many- ja many2many -kenttiä käytetään eri mallien (models) välisten suhteiden määrittämiseen. (Odoo docs 2024b.)

Jokainen kenttä kuuluu suurempaan kategoriaan eli malleihin (models). Mallit pitävät sisällään paljon eri tietoja kuten XML-näkymät, kentät ja kenttien käyttöoikeudet. Tästä esimerkkinä on malli res.partner, joka pitää sisällään Contacts-moduulin asiakastietonäkymät. Tätä mallia muokkaamalla voidaan aikaansaada halutut muutokset asiakastietokorttiin. Mallit ovat Odoossa Python-luokkia, jossa määritetään moduulin toiminta. Mallit voidaan käyttää ja periä uusissa moduuleissa uusien mallien luomiseksi.

Kolmantena mainitsemisen arvoisena elementtinä on Odoossa olevat näkymät (views). Näkymät pitävät sisällään kaiken mitä käyttöliittymässä nähdään Odoossa. Näkymiä on mahdollista myös muuttaa moduulien avulla. Kuten muissakin elementeissä, myös näkymissä on mahdollista, mutta ei suositeltavaa, muokata niitä muokataan suoraan Odoo-järjestelmässä. Tässä ongelmana voi olla sama kuin edellisissäkin elementeissä, että näkymät aiheuttavat yhteensopivuusongelmia tulevien päivitysten kanssa. Odoo tukee kymmentä erilaista näkymätyyppiä. Yleisimmät näkymät ovat kuitenkin form, tree ja kanban. Form-näkymää käytetään yksittäisten tietueiden näyttämiseen ja muokkaamiseen. Tree-näkymää käytetään tietueiden näyttämiseen taulukkomuodossa. Kanban-näkymässä tietueet näytetään kortteina, joita voidaan siirrellä käyttöliittymässä taulukoiden sisällä. Uutta moduulia tehdessä vanha näkymä on mahdollista periä suoraan koodissa. Tästä on hyötyä etenkin tehtäessä moduulia, jossa käytetään samoja kenttiä kuin edellisessä näkymässä. Tällöin uuden näkymän tekemistä ei tarvitse aloittaa täysin alusta vaan uudessa moduulissa näkymänä on perityn mallin näkymä. Perittyä näkymää voidaan muokata esimerkiksi attribuuttien avulla. Attribuuteilla voidaan piilottaa uudessa moduulissa tarpeettomia kenttiä, joita perityssä näkymässä on saattanut olla. Lisäksi attribuuteilla voidaan muokata jo olemassa olevan kentän nimeä, joka ei muuta nimeä tietokannassa vaan ainoastaan näkymässä. (Odoo docs 2024c.)

4 TOIMEKSIANTAJAN LIIKETOIMINTATARPEET JA ODOON KUSTOMOINTI

Työ toimeksiantajalle alkoi katsauksella, mitä yritys tarvitsee. Tästä käytiin useita palavereita ja tultiin lopputulokseen, että yritykselle tarvittaisiin toiminnanohjausjärjestelmään ensisijaisesti myynninhallintaan. Tähän kuuluisi esimerkiksi uuden CRM-järjestelmän luonti, räätälöity asiakasvaiheiden hallinta ja paranneltu näkymä tuleville tapahtumille. Näillä muutoksilla toiminnanohjausjärjestelmästä saataisiin tehokkaampi sovellus myynnin jokapäiväisessä käytössä. Muita tarpeita, joita tuli esille, olivat esimerkiksi laskupohjamuutokset, Procountor APIn ja Odoo-järjestelmän keskeinen kommunikaatio sekä moduuli, jolla voidaan tarkastella asiakkaiden omistamia laitteita ja liittymiä. Etenkin Procountor APIn ja Odoo-järjestelmän välisellä ohjelmointirajapinnalla saataisiin tehostettua yrityksen laskutusprosessia. Huomattiin myös, että yritykselle olisi hyvä luoda toiminnanohjausjärjestelmä, koska yrityksellä oli useita osa-alueita toiminnassaan. Toiminnanohjausjärjestelmässä voitaisiin yhdistää yrityksen useat osa-alueet yhteen sovellukseen, jossa olisi myös yrityksen asiakastiedot, laskutus ja inventaario. Yrityksellä on myös useita työntekijöitä, joten keskitetty toiminnanohjausjärjestelmä olisi erittäin sopiva vaihtoehto toimeksiantajalle.

Toimeksiantajan liiketoimintatarpeiden tunnistamisen jälkeen oli välttämätöntä määrittää tarpeiden priorisointi. Koska myynninedistämisen katsottiin olevan pääasiallinen tavoite, alkoi työ keskittymällä aluksi CRM-järjestelmään ja sen konfigurointiin. Työ eteni vaiheittain ja toimeksiantajan kanssa käytiin tehtävien jälkeen uudelleenkatsausta seuraavista toimenpiteistä. Vasta työn myöhemmässä vaiheessa aloitettiin muiden moduulien tekeminen kuten Procountor API -moduuli. Procountor API -moduuli sekä asiakastietokorttiin tehdyt moduulit ovat kaksi suurinta moduulia, joihin työ on pääasiassa kiteytynyt ohjelmoinnin osalta. Muitakin moduuleja on tehty, mutta nämä kaksi moduulia liittyvät vahvasti myös muihin tehtyihin moduuleihin.

4.1 Odoon merkitys työssä

Odoo valikoitu tähän työhön sopivaksi toiminnanohjausjärjestelmäksi useista syistä. Keskeisimmät syyt olivat kustannustehokkuus ja räätälöintimahdollisuudet. Toimeksiantajalle oli tärkeää toiminnanohjausjärjestelmän hinta ja se, että järjestelmä saataisiin mukautettua mahdollisimman hyvin yritykselle sopivaksi. Molemmat näistä vaatimuksista Odoo kattaa hyvin sen Community-versiolla. Lisäksi Odoo pystyy vastaamaan erittäin hyvin toimeksiantajan tarpeisiin. Odoo voidaan

keskittää yrityksen usealle osa-alalle ja järjestelmää voivat käyttää kaikki yrityksen työntekijät tarpeen mukaan.

Odoon muokattavuus osoittautui todella hyväksi työtä varten. Alkuun Odoota saatiin kustomoitua toimeksiantajalle sopivaksi Odoon valmiilla moduuleilla ja ladattavissa olevilla kolmannen osapuolen moduuleilla. Myöhemmin kuitenkin järjestelmän tullessa tutuksi alkoi Odoon kustomointi ohjelmoimalla ja tähän kuului muun muassa omien moduulien tekeminen ja esimerkiksi Odoon oma asiakastietonäkymä oli pienellä perehtymisellä mukautettavissa oleva näkymä. Tästä saatiin työn edetessä myynnille parempi näkymä, jossa ei ole esimerkiksi turhia kenttiä asiakastietonäkymässä. Lisäksi kustomoinnilla saatiin tehtyä uusia kenttiä asiakastietonäkymään, jotka taas huomattiin tarpeellisiksi, ja näille saatiin tehtyä myös erilaisia toiminnallisuuksia, kuten asiakkaiden jaottelu niiden asiakasvaiheiden perusteella.

Sovelluskehittäjän näkökulmasta kaikki Odooseen tehdyt muutokset toteutettiin kahdella helposti lähestyttävällä ohjelmointikielellä: Pythonilla ja XML:llä. Python-kieltä käytettiin pääasiassa sovelluksen toiminnallisuuksien luomiseen, kun taas XML:llä saatiin aikaan uudet näkymät sovellukseen. Odoon avoimen lähdekoodin luonne edisti merkittävästi sovelluskehitystä ja tarjosi mahdollisuuden hyödyntää useita foorumeita, joilla keskusteltiin eri aiheista. (Odoo forum 2022.)

4.2 Moduulien suunnittelu ja kehitys toimintatarpeisiin sopiviksi

Uusien moduulien tekeminen alkoi usein tarpeesta saada toiminnanohjausjärjestelmään uusi toiminto tai muuttaa olemassa olevaa moduulia. Muiden työntekijöiden ja toimeksiantajan kanssa käytiin usein dialogia siitä, mitä toimintoja moduulin tulisi pitää sisällään. Kun moduulin toimintatarpeista päästiin yhteisymmärrykseen, alkoi moduulin suunnittelu. Moduulin suunnitteluun kuuluu muun muassa tietokannan, käyttöliittymän ja toteutuksen suunnittelu. Tietokannan suunnittelussa pääasiallisena osana oli eri kenttien välisten yhteyksien suunnittelu. Usein moduuleissa oli uusia kenttiä, joiden oli tarkoitus toimia yhteen jo Odoossa valmiina olevien moduulien ja mallien, kuten laskutuksen tai asiakastietonäkymän kanssa. Tämä onnistuu esimerkiksi ohjelmoimalla moduuli siten, että se perii valmiina olevan mallin (model), kuten res.partner, jolloin kyseinen moduuli voi käyttää kaikkia res.partner-mallin kenttiä ja yhteyksiä. Toisena vaihtoehtona on, että moduulille luodaan kokonaan uusi malli kuten new.model, johon ohjelmoidaan uudet kentät ja näkymät tilanteen mukaan.

Moduulissa toiminnallisuus ohjelmoidaan Python-tiedostoihin ja näkymät sen sijaan XML-tiedostoihin. Riippuen toimintatarpeista voi olla, että moduulissa ei ole joko toiminnallisuutta tai vastaavasti näkymiä ollenkaan. Moduulin on mahdollista olla puhtaasti toiminnallinen moduuli, jolloin näkymien ohjelmointi ei ole tarpeellista. Myös ohjelmointirajapintojen hyödyntäminen tapahtuu Python-tiedostojen sisällä, jolloin ei ole tarpeellista luoda XML-näkymiä Odooseen. Vastaavasti moduuli voi olla täysin uusi tai olemassa olevaa näkymää muokkaava. Tällöin toiminnallisten Python-tiedostojen käyttö ei ole tarpeellista. Tällaisia ovat esimerkiksi moduulit, joiden ainoa tarkoitus on muokata olemassa olevia näkymiä esimerkiksi lisäämällä tai poistamalla kenttiä näkymässä.

Usein uutta moduulia tehdessä ongelmaksi syntyy moduulin yhteensopivuus Odoon tietokannan kanssa. Tällöin paras tapa seurata moduulin ja Odoon vuorovaikutusta on Odoon oman lokitiedoston kautta. Lokitiedosto kerää Odoosta jatkuvasti paljon tietoa ja samaan tiedostoon voidaan kirjata omia lokitietoja moduulista. Tämä helpottaa ongelmien kartoittamista moduulin ja Odoon tietokannan välillä. Esimerkiksi yhtenä ongelmana ilmeni usein, että peritty malli res.partner ei tunnista käytettyä kenttää, jolloin lokitiedostoon tulee ilmoitus, mistä kentästä on kyse. Ilmoitus ongelmasta lokitiedoissa helpottaa Odoon ohjelmointia ja moduulin viimeistelyä.

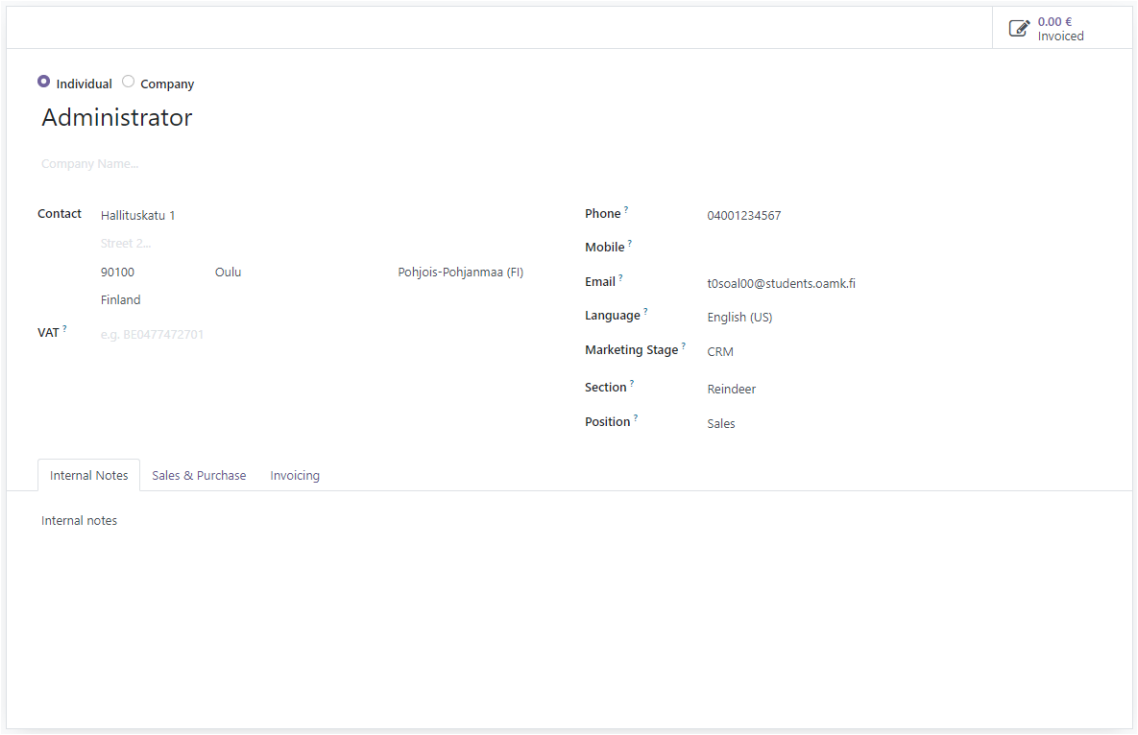
4.3 Asiakastietokortti muutokset

Asiakastietokortteihin tehtävät muutokset olivat ensimmäisiä ohjelmointia tarvittavia muutoksia, joita Odooseen tehtiin. Moduulin tekeminen lähti tarpeesta saada myynnille parempi asiakastietonäkymä Odooseen. Moduulin tarkoituksena oli visuaaliset päivitykset asiakastietonäkymään kuten kenttien peittäminen ja uusien kenttien luominen. Asiakastietonäkymiin tehtäviin muutoksiin ei moduulissa tarvinnut muuta kuin XML-tiedoston, jossa näkymän muutokset sai käyttämällä attribuutteja, kuten nähdään kuvassa 3. Elementtejä, jotka todettiin tarpeettomiksi ja päädyttiin piilottamaan moduulilla, olivat esimerkiksi asiakkaan profiilikuva, nettisivu ja asiakastietonäkymän alalaidassa oleva yhteystiedot ja osoitteet sivu. Näiden elementtien poistaminen edisti asiakastietonäkymän selkeyttä ja käyttökokemusta.

```
<xpath expr="//page[@name='sales_purchases']" position="before">
    <xpath expr="//page[@name='internal_notes']" position="move"/>
</xpath>
<xpath expr="//field[@name='website']" position="attributes">
    <attribute name="invisible">True</attribute>
</xpath>
```

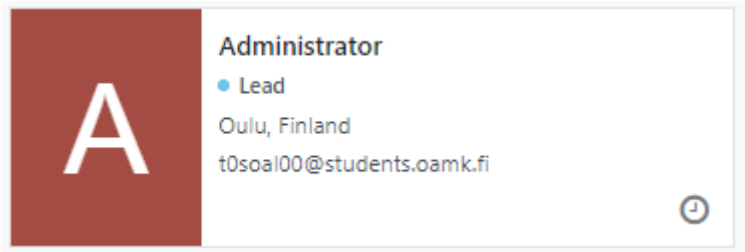
KUVA 3. XML-attribuutit.

Elementtien piilottamisen lisäksi moduulilla tehtiin olemassa oleviin kenttiin nimi muutoksia ja vaihdettiin kenttien ja sivujen järjestystä näkymässä. Nämäkin muutoksen olivat mahdollisia tehdä attribuuteilla suoraan XML-tiedostossa. Moduulilla kuitenkin luotiin myös uusia kenttiä asiakastietonäkymään, mitkä todettiin tarpeellisiksi. Näitä uusia kenttiä olivat markkinointivaihe, nimike ja asema. Asiakastietonäkymään tehdyt muutokset voidaan havaita kuvassa 4. Kaikki uudet kentät liittyivät asiakkaiden jaotteluun. Kentillä saatiin jaoteltua asiakkaita esimerkiksi karjatalouden, porotalouden ja wildlife-puolen välillä. Asiakastietonäkymässä puolen valitseminen on tehty helpoksi many2one-kentän avulla. Many2one-kenttä avaa näkymässä vetovalikon, josta puoli voidaan asiakkaalle määrittää. Uusien kenttien tekeminen tehdään moduulissa Python-tiedostoissa. Uudet kentät tuo toiminnallisuutta moduulille, joten ohjelmointia ei voi tehdä pelkästään XML-tiedostoissa.



KUVA 4. Muutettu asiakastietonäkymä.

Moduulissa tehtiin myös yhteen kenttään computed field -funktio. Alkuperäisesti asiakastietonäkymässä oleva tags-kenttä näytti saman tiedon myös asiakastietoluettelossa pienenä tekstinä, kuten nähdään kuvassa 5. Tämä helpotti asiakastietoluettelon lukemista, kun luettelossa nähtiin suoraan tämän tags-kentän valinta. Kuitenkin tämä kenttä oli many2many-kenttä, jolloin asiakkaalla pystyi olemaan monta eri asiakasvaihetta, mikä ei todellisuudessa ole mahdollista. Computed field -funktion avulla tämä toiminto kuitenkin saatiin toteutettua myös uudessa markkinointivaihe many2one-kentässä. Funktio katsoi asiakkaan markkinointivaihe kenttään asetetun tiedon ja samalla kopioi sen piilotettuun tags-kenttään. Vaikka tags-kenttä oli piilotettu ei sitä ole tietokannasta poistettu, joten tällä saatiin haluttu toiminto saavutettua myös tässä uudessa kentässä.

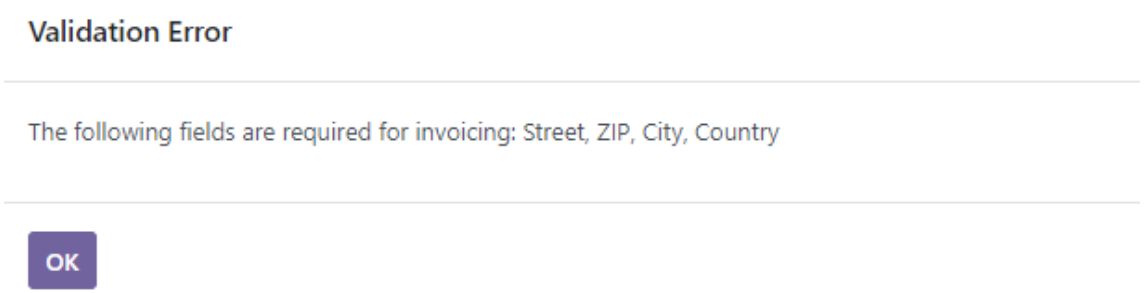


KUVA 5. Asiakastietoluettelon asiakas valinta.

4.4 Procountor API -moduuli

Odooseen merkittävin päivitys toteutettiin, kun syntyi tarve luoda rajapinta toiminnanohjausjärjestelmän ja Procountor-taloushallinnon ohjelmiston välille. Toimeksiantajan kanssa tuli selväksi, että yrityksen laskutus halutaan tapahtuvan Odoon järjestelmän kautta. Yritys kuitenkin käyttää myös Procountor-taloushallinnon sovellusta, joten näiden kahden sovelluksen välille oli välttämätöntä saada rajapinta. Procountor on suomalainen taloushallinnon ohjelmisto, joka tarjoaa monen kokoisille yrityksille erilaisia taloushallinnon ominaisuuksia. Ominaisuuksista mainittakoon myynti- ja ostolaskutus, kirjanpito ja raportointi sekä palkanlaskenta (Procountor 2024b). Näistä ominaisuuksista toiminnanohjausjärjestelmään haluttiin laskutusrivien siirtäminen Procountor-ohjelmistoon. Procountor tarjoaa ohjelmistokehittäjille ilmaisen testiympäristön heidän API-testauspalvelimellensa. Tällä ympäristöllä voidaan ohjelmoida toimiva rajapinta toiminnanohjausjärjestelmän ja Procountorin välille ilman erillisiä kustannuksia. Valmiin rajapinnan valmistuessa voidaan testiympäristö päivittää yhteensopivaksi Procountor-taloushallinnon version kanssa. (Procountor 2024a.)

Procountor API -moduulin tehtävä on lähettää tehdyn laskun laskutusrivit Procountor-taloushallinnon ohjelmistoon. Ennen kuin Procountorin ohjelmointirajapintaa voidaan käyttää, täytyy moduulilla ensin saada pääsytunnus (access_token). Moduuli hakee pääsytunnuksen aina laskua luodessa, jonka jälkeen tunnus on voimassa yhden tunnin. Pääsytunnuksen onnistuessa moduuli suorittaa funktion, joka hakee viimeisimmän luodun laskun ja kopioi laskusta asiakastiedot ja laskutustiedot ja siirtää ne POST-pyynnöllä Procountoriin. Moduulille on tehty myös paljon varoituksia, jotka ilmenevät suoraan toiminnanohjausjärjestelmässä siltä varalta, että laskua ei saatu siirrettyä Procountoriin oikein. Moduulissa on laitettu pakollisiksi riveiksi esimerkiksi asiakastiedot kuten nimi ja postiosoite. Ilman näitä tietoja moduuli varoittaa laskun luojaa puuttuvista tiedoista, kuvan 6 osoittamalla tavalla.



KUVA 6. Varoitus asiakastietojen puuttuvista kentistä.

Moduulissa etsitään tietoa kuvan 7 tavoin Odoon tietokannan kolmesta mallista account.move, account.move.line ja res.partner. Account.move-malli pitää sisällään laskun asiakastiedot ja Account.move.line sen sijaan pitää sisällään laskutusrivit. Res.partner-mallia tarvitaan, koska moduuli tarkastaa lähetettävän laskun ja lisää laskulle vastaavan asiakkaan tiedot. Moduulin funktio hakee mallien tiedot, jonka ansiosta lasku saadaan siirrettyä Procountoriin oikein. Ohjelmoidessa täytyi kuitenkin tehdä kolmelle kentälle uudelleenkartoitus. Nämä kentät olivat asiakkaan kieli ja asuinmaa sekä arvonlisävero. Uudelleenkartoitus täytyi tehdä, koska Odoo tallentaa esimerkiksi asiakkaan kielen amerikanenglanti muodossa en_US tai suomen kielen muodossa fi_FI, jolloin tieto ei ole yhteensopiva lähetettävän datan kanssa ja ei näy Procountor-ohjelmistossa oikealla tavalla. Kielen uudelleenkartoittamisella oli myös hyötynä se, että asiakkaan kielen mukaan saadaan vaihdettua laskun kieli automaattisesti oikeaksi. Sen sijaan asuinmaan ja arvonlisäveron uudelleenkartoittamisessa oli syynä ainoastaan tiedon lähettämiseen liittyvät syyt. Odoossa asuinmaa on kirjoitettu suomelle muodossa Finland, kun taas ohjelmointirajapinta vaatii, että tieto lähetään muodossa FINLAND. Moduulissa myös hyödynnettiin datetime ja odoo.exceptions -kirjastoja. Datetime-kirjastosta moduulissa käytettiin strftime-metodia, jonka avulla saatiin laskun päivämäärä ja laskun eräpäivä oikeaan muotoon laskun lähetystä varten. Päivämäärien tulee olla lähetettävässä datassa esimerkiksi muodossa ”2024-04-16”. Odoo.exceptions-kirjastoa tarvittiin sen sijaan, että järjestelmässä saatiin suoritettua aiemmin mainittu Validation Error -ilmoitus, jos laskua luotaessa asiakkaalta puuttuu pakollisia tietoja. Muita kirjastoja, joita moduulissa käytettiin, oli esimerkiksi models ja logging. Logging-kirjastolla voidaan tallentaa haluttuja tietoja Odoon lokitiedostoon, mikä helpottaa virheiden jäljittämistä ja järjestelmän toiminnan seuraamista.

```python
class SendInvoiceData(models.Model):
    _inherit = 'account.move'

    def action_post(self):
        last_invoice = self.env['account.move'].search([], order='id desc', limit=1)
        last_invoice_lines = self.env['account.move.line'].search([
            ('move_id', '=', last_invoice.id),
            ('product_id', '!=', False)
        ])
        partner_id = self.partner_id.id if self.partner_id else None
        partner = self.env['res.partner'].browse(partner_id)
```

KUVA 7. Laskun tietojen haku.

Procountor API -moduuli on ohjelmoitu Python-ohjelmointikielellä ja sille ei erikseen tehty omia näkymiä toiminnanohjausjärjestelmään. Moduuli suorittaa Python-ohjelman, kun Odoossa luodaan ja hyväksytään lasku. Moduuli lähettää laskun Procountor-ohjelmistolle vasta, kun moduuli on varmistanut laskun asiakkaan oikeat tiedot ja järjestelmän käyttäjä hyväksyy laskun ja painaa confirm-nappia. Tällä ehkäistään myös sitä, että moduuli ei esimerkiksi lähetä laskua, kun laskusta on tehty vasta luonnos. Moduuli käytti myös paljon muita Odoon alkuperäisiä moduuleja hyödyksi kuten Invoicing, Inventory ja Contacts. Invoicing-moduulissa lähetetään laskuja, joka suoraan hyödyntää Inventory ja Contacts -moduuleja. Inventory-moduuli pitää sisällään esimerkiksi laskutusriveissä olevat tuotteet, jotka kerätään lähetettävään dataan kuvan 8 tavoin. Contacts-moduulissa on kaikki toiminnanohjausjärjestelmään lisätyt asiakastiedot. Procountor API -moduuli toimii niin Invoicing-moduulissa laskua tehdessä kuin myös suoraan asiakastietonäkymästä tehtävällä laskulla.

```python
invoice_lines = []

for line in last_invoice_lines:
    product_name = line.name
    quantity = line.quantity
    price = line.price_unit
    discount = line.discount
    vat = line.tax_ids and line.tax_ids[0]
    vat_id = vat.id if vat else None
    remapped_vat_value = vat_mapping.get(vat_id, 0)

    invoice_lines.append({
        "product": product_name,
        "quantity": quantity,
        "unit": "PIECE",
        "unitPrice": price,
        "discountPercent": discount,
        "vatPercent": remapped_vat_value
        })
```

KUVA 8. Lähetettävät laskutusrivit.

Laskutusrivien hakemisen jälkeen moduuli yhdistää laskutusrivit muiden tietojen kanssa, joita taloushallinnon ohjelmistoon tarvitaan. Yhdistämisen jälkeen moduuli lähettää POST-pyynnön Procountor-ohjelmistolle, jossa lähetetään laskun tiedot, pääsytunnus, ohjelmointirajapinnan verkkotunnus sekä timeout-parametri. Tämän jälkeen Moduuli tarkastaa ohjelmointirajapinnan vastauksen ja antaa sille vastaavan lokimerkinnän. Kun moduulilta ei tule virhemerkintöjä ja laskun tiedot on saatu lähetettyä Procountor-ohjelmistoon, näkyy laskujen tiedot ohjelmistossa listana, joista voidaan valita haluttu lasku. Procountor-ohjelmistossa valitulle laskulle on paljon toimintoja, jonka takia Odoo on hyvä yhdistää ohjelmiston kanssa. Procountorissa liitetty lasku voidaan esimerkiksi merkata maksetuksi tai allekirjoittaa sähköisesti. Lisäksi Procountor-ohjelmistoa voidaan käyttää yrityksen laskujen arkistoimiseen. Ohjelmointirajapinnan ansiosta Odoota voidaan kuitenkin käyttää pääasiallisena sovelluksena laskutuksessa, vaikka yrityksellä olisikin käytössä Procountor-ohjelmisto. Odoo on myös hyvä yrityksen laskutussovellus relaatiotietokannan ansiosta. Odoon käyttöliittymässä laskun tekeminen on käyttäjäystävällistä ja laskua luodessa voidaan valikosta valita asiakas ja vetovalikoista laskutettavat tuotteet, määrät ja mahdolliset alennukset. Tuotteille voidaan myös määrittää hinta ja arvonlisäverokanta toiminnanohjausjärjestelmässä ja niitä ei tarvitse erikseen muuttaa, jos tilanne ei toisin vaadi.

5 TESTAUS JA VIIMEISTELY

Testaus ja viimeistely on olennainen osa toiminnanohjausjärjestelmän luomista. Ennen toiminnanohjausjärjestelmän käyttöönottoa on toimeksiantajan ja työntekijöiden kanssa käyty useita dialogeja toiminnanohjausjärjestelmän käytettävyydestä, uusista moduuleista ja konfiguroinneista. Keskustelujen päätteeksi on tultu haluttuun lopputulokseen ja suoritettu testauksia järjestelmän toimivuudesta. Moduulit ovat olleet ensisijaisesti testauksen kohteena. Järjestelmästä on haluttu luoda mahdollisimman yksinkertainen myynnin avuksi. Muita testauksen kohteita on ollut esimerkiksi Odoon sisäiset konfiguraatiot kuten keskitetty sähköposti. Lisäksi Odoosta on suoritettu testejä, jossa järjestelmää käytetään päivittäisessä käytössä ja katsotaan vastaako järjestelmä haluttuja tarpeita.

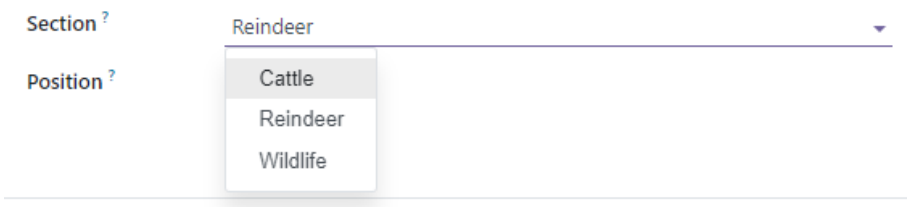
Testauksien jälkeen alkoi Odoon viimeistely. Yhtenä tärkeimpänä asiana pidettäköön toiminnanohjausjärjestelmän kustomoinnin jatkuvuutta. Tähän kuuluu se, että itse tehdyt kustomoinnit ovat myös muiden ymmärrettävissä. Viimeistelyyn kuuluikin paljon tehtyjen moduulien kommentointia. Kommentoinnin lisäksi yrityksen työntekijöiden kanssa käytiin keskustelua, jossa puheenaiheena olivat tehdyt moduulit ja miten niitä voisi jatkokehittää. Lisäksi tehdyistä töistä oli kattavaa dokumentaatiota, jossa kerrottiin jo tehdyistä toiminnoista ja tietoa, miten niihin on päädytty. Lisäksi dokumentaatioihin kuului myös listaa tehtävistä, joita voidaan jatkokehittää tulevaisuudessa. Toiminnanohjausjärjestelmälle luotiin myös arkkitehtuuri, jossa on kuvattu järjestelmää yleisellä tasolla ja yksittäisiä moduuleja paljon tarkemmin. Arkkitehtuuriin on myös lisätty sivu, jossa on taulukko toiminnanohjausjärjestelmässä käytettävistä käyttäjäoikeuksista.

5.1 Odoon viimeistely

Viimeisenä alkoi Odoo-toiminnanohjausjärjestelmän viimeistely. Ohjelmointi ja konfigurointi tapahtui toiminnanohjausjärjestelmässä suurelta osin paikallisesti tietokoneella, jonka jälkeen ajankohtaiseksi tuli järjestelmän siirtäminen yrityksen palvelimelle. Anicarella oli jo valmiiksi oma palvelin, johon luotiin uusi Odoo-tietokanta. Yrityksen Odoo on asennettu Ubuntu-palvelimelle ja toimii web-selaimella HTTPS-protokollalla. Web-selaimessa järjestelmää pääsee käyttämään kirjautumissivun jälkeen ja järjestelmässä voidaan luoda uusia käyttäjiä, jolloin työntekijät pääsevät kirjautumissivusta eteenpäin. Halutessaan järjestelmälle voidaan luoda keskitetty sähköposti,

jolloin järjestelmään voidaan liittää käyttäjiä sähköpostikutsulla. Ubuntu-palvelimella toimivan Odoon tiedostohakemistoon pääsee tarvittaessa komentorivillä muodostetulla SSH-yhteydellä.

Järjestelmään täytyi ensin tuoda tehdyt moduulit. Moduuleja tuotiin Odooseen kuusi kappaletta, ja moduulit saatiin tuotua järjestelmään komentorivin scp -r -komennolla. Moduulien tuonnissa järjestelmään ei tullut yhteensopivuusongelmia, koska uusi Ubuntu-palvelimella toimiva järjestelmä oli sama paikallisen järjestelmän kanssa eli versio 16.0. Järjestelmään lisättiin myös tarvittavia moduuleja, jotka löytyivät Apps-moduulista. Näitä moduuleja olivat muun muassa Contacts, Inventory ja Invoicing. Moduulien viennin jälkeen järjestelmään tehtiin useita konfigurointeja. Konfigurointeihin kuului esimerkiksi asiakaslistan tuonti järjestelmään. Asiakaslistan sai tuotua import records -toiminnolla, jolloin Odoo kopioi tiedot Excel-tiedostosta Contacts-moduuliin. Asiakastiedoissa oli kuitenkin useita kenttiä, jotka vaativat muokkaamista. Järjestelmään tuotiin moduuleja, jotka muokkasivat asiakastietonäkymää ja loi uusia kenttiä. Näihin kenttiin täytyi luoda tiedot järjestelmässä, jolloin kentät vastasivat niiden toimivuutta, kuten nähdään kuvassa 9. Tämän jälkeen asiakaslistalle luotiin filttereitä, jotta asiakkaat olisi helpompi jaotella luotujen kenttien mukaan. Muita konfigurointeja oli esimerkiksi uusien kielien asentaminen järjestelmään, käyttäjien lisääminen ja tuotteiden ja tuotetietojen luominen Inventory-moduuliin.



KUVA 9. Asiakastietokortin toimialuevetovalikko.

5.2 Moduulien toimivuus

Kaikki kuusi tuotua moduulia toimivat myös Ubuntu-palvelimelle asennetussa Odoossa. Kuten jo aiemmin mainittu usean moduulin tarkoitus oli muokata esimerkiksi asiakaskortin näkymää soveltumaan paremmin myynnin osa-alueelle. Lisäksi Procountor-taloushallinnon ja Odoon välille luotu ohjelmointirajapinta toimii nykyisessä versiossa. Muilla moduuleilla toiminnanohjausjärjestelmään luotiin uudet mallit, joihin voitiin luoda lisää kenttiä. Kentät olivat pääasiassa muotoa many2one eli järjestelmässä vetovalikon muodossa tai computed field -kenttiä, joilla saatiin järjestelmän kentille erilaisia toimintoja.

Moduulien ylläpito ei myöskään ole tarpeellista. Moduuleissa on tehty tarvittavat tiedot kentille, joten niitä ei tarvitse enää muokata Odoossa, ellei kentille halua lisää tietoja. Ainoastaan Procountor-ohjelmointirajapinta voi vaatia ylläpitoa tulevaisuudessa. Tämä riippuu pitkälti siitä miten paljon Procountor päivittää ohjelmointirajapintojaan. Procountorin sivuilla voi kuitenkin liittyä postituslistalle, jolloin ilmoitukset uusista rajapintapäivityksistä tulee sähköpostiin. Moduulien yhteensopivuutta ei ole kuitenkaan testattu tämänhetkisen uusimman Odoo-version kanssa eli versio 17.0. Toimeksiantajan kanssa on päädytty tulokseen, että säilytetään versio 16.0. ja halutessaan siirrytään uudempaan versioon tulevaisuudessa.

5.3 Toiminnanohjausjärjestelmän jatkokehittäminen

Toiminnanohjausjärjestelmälle jäi paljon jatkokehitettävää. Osa jatkokehitettävistä ideoista oli keskeneräisiä moduuleja, joihin ei saatu aikamääreessä haluttuja toimintoja. Osa jatkokehitysideoista oli sen sijaan listauksia, joita on tullut esille dialogeissa ja joita voi jatkokehittää opinnäytetyön jälkeen. Vaikka työni on pääasiassa keskittynyt myynnin edistämiseen ja CRM-järjestelmän paranteluun on Odoossa vielä paljon tekemistä ja mahdollisuuksia käyttöönottaa tulevaisuudessa. Esimerkiksi yrityksen laajentuessa voidaan nähdä tarpeelliseksi ottaa käyttöön tuotannonhallintaa, johon Odoolla on jo valmiiksi olemassa olevia moduuleja. Nämäkin moduulit ovat kustomoitavissa yrityksen tarpeisiin sopiviksi.

Moduuleissa jatkokehitettävää oli eritoten Procountor-ohjelmointirajapinnan kanssa. Moduulin tekeminen oli pitkä prosessi ja aikamääreet tulivat nopeasti vastaan. Toimintoja, jotka jäävät jatkokehitettäviksi on esimerkiksi se, että Odoon moduuli hakee vähintään kerran päivässä maksetut laskut ja merkkaa ne toiminnanohjausjärjestelmässä maksetuksi. Procountor API -moduulissa ei myöskään suoriteta kurssimuunnoksia. Anicarella on useita asiakkaita, jotka eivät ole Suomen sisällä ja jotka käyttävät eri valuuttaa kuin euroa. Tähän oli suunniteltu, että Odoo hakee rajapinnan kautta taloushallinnon sovelluksesta sen hetkiset valuuttakurssit ja muuttaa sen avulla laskun suoraan kyseiselle valuutalle oikealla summalla. Procountor API -moduulissa jäi myös hieman keskeneräiseksi laskupohja. Moduulissa olisi ollut mahdollista hyödyntää Procountorin laskupohjaa, mutta silloin lasku ei lähtisi Odoon keskitetystä sähköpostista.
Muita moduuleja, joiden ohjelmointi on ehditty aloittaa, mutta jää jatkokehitettäviksi on purchased products, device information ja license information -moduulit. Kaikkien näiden moduulien oli

tarkoitus tuoda lisätoimintoja asiakastietokortteihin. Asiakastietonäkymässä oli tarkoitus, että kaikista moduuleista olisi luotu nappi, jota klikkaamalla pääsisi näkemään moduulien toiminnot. Purchased products -moduulin tarkoitus oli näyttää asiakkaiden omistamat tuotteet ja liittymät. Tämä toiminto on olemassa jo Odoossa, mutta tarkoituksena olisikin, että saataisiin toiminnanohjausjärjestelmään edellisten asiakkaiden ostamat tuotteet näkyviin helposti. Device information ja license information -moduulien toiminnot olisi olleet lähestulkoon samanlaiset. Device information -moduulin oli tarkoitus näyttää asiakkaan laitteiden tiedot sivulla. Sivulla näkyisi esimerkiksi kaikkien laitteiden viimeisimmät paikkatiedot ja pariston tila. License information -sivulla sen sijaan oli tarkoituksena näkyä esimerkiksi laitteiden liittymien aktivointipäivät ja liittymien eräpäivät. Kaikki moduulit hakivat demo dataa ohjelmointirajapintasivuilta ja toimivat käytännössä. Ohjelmointirajapintoja ei kuitenkaan keretty luomaan yrityksen omien laite- ja liittymätietojen kanssa, jonka takia moduulit jäivät keskeneräisiksi.

Muita jatkokehitys ideoita, joita ei keretty ottaa edes tehtäväksi opinnäytetyön aikana on esimerkiksi Odoon Inventory-moduulissa tehtävät viivakodit. Odoolla on maksullisessa Enterprise-versiossa moduuli, jolla voidaan luoda tuotteille ja komponenteille viivakoodit. Näillä saataisiin helposti seurattua laitteen tuotantoa. Tuotantoa voitaisiin seurata aina laitteen komponenttien hankkimisesta aina asiakkaalle valmiin tuotteen lähettämiseen asti. Näin pysyy hyvä kirja tuotteista ja voidaan myös suorittaa hälytyksiä järjestelmässä, kun esimerkiksi tietyn komponentin tai tuotteen varastotilanne alitetaan. Yhtenä jatkokehitys ideana voidaan myös pitää mahdollisuutta siirtyä seuraavalle versiolle Odoosta eli versioon 17.0 tai ensi lokakuussa julkaistavaan versioon 18.0.

6 POHDINTA

Työn tavoitteena oli saada luotua toimiva toiminnanohjausjärjestelmä Anicare Oy:n myynnin avuksi muokkaamalla järjestelmää ja luomalla moduuleja auttamaan myyntiä työssään. Näihin tavoitteisiin päästiin melko hyvin ja ainoastaan jatkokehitettävää jäi vielä moduuleihin, sillä aikamääreet tulivat nopeasti vastaan. Työssä kuitenkin oppi todella paljon Odoo-toiminnanohjausjärjestelmästä, sillä kyseinen järjestelmä oli minulle entuudestaan tuntematon. Lisäksi työn aikana tuli lisää kokemusta ohjelmistokehittämisestä, sillä työssä tehdyt moduulit oli pääasiassa ohjelmoitu Python-ohjelmointikielellä. Työssä luotiin useita moduuleja, joista vain muutamaa ei otettu käyttöön viimeistellyssä versiossa, joka tuotiin yrityksen Ubuntu-palvelimelle. Näiden moduulien lisäksi viimeisteltyyn versioon kuului myös paljon järjestelmän konfigurointia. Konfigurointiin kuului esimerkiksi yrityksen edellisen CRM-järjestelmän asiakaslistan kopiointi uuteen toiminnanohjausjärjestelmään. Työssä erityisesti huomasi miten tärkeää ohjelmoinnissa on yhteensopivuus, koska moduulit, joita luotiin täytyi olla myös yhteensopivia Odoo-järjestelmän kanssa. Tämä usein oli yleisin ongelmien aiheuttaja, mutta huolellisuudella ja lokitiedostojen tarkastamisella Odoon ja moduulien yhteensopivuusongelmat sai selvitettyä. Lisäksi ohjelmointirajapinnoista tuli paljon lisää kokemusta, sillä ennen opinnäytetyötä minulla oli kokemusta ohjelmointirajapinnoista vain lyhyistä projekteista. Tämän työn aikana kuitenkin miltei puolissa moduuleista, joita ohjelmoin, käytettiin ohjelmointirajapintoja.

Jatkokehitettävää toiminnanohjausjärjestelmään jäi melko paljon. Tämä kuitenkaan ei ole suuri menetys sillä järjestelmä on todella laaja, joten on hankalaa edes saada kaikkia jatkokehitysideoita toteutettua ilman edeltävää kokemusta Odoo-järjestelmästä. Kuitenkin näitäkin jatkokehitysideoita olen kerennyt, jonkin verran aloittamaan ja vaikka ne ei kerennyt valmistua viimeiseen versioon, saatiin järjestelmään tuotua tärkeimmät moduulit. Näistä jatkokehitysideoista kuitenkin tehtiin kattavaa listausta ja pidettiin palavereja, jossa tiedotettiin muita järjestelmän parissa työskenteleviä, jotta heillä olisi mahdollisimman helppoa jatkaa jo aloitetuista jatkokehitysideoista.

Kaiken kaikkiaan opinnäytetyö sujui omasta mielestä hyvin. Päätavoitteena kuitenkin oli, että yritykselle saadaan parempi toiminnanohjausjärjestelmä ja nimenomaan CRM-järjestelmä, jossa asiakkaiden tiedot olisivat helpommin nähtävissä. Toiminnanohjausjärjestelmässä näkyvä asiakaslista suoriutuu hyvin yrityksen tarpeista. Asiakastietokortit sisältävät nykyään vain tietoja, joita myynnin tehtävissä tarvitaan ja asiakastiedoissa voidaan merkata esimerkiksi asiakaspalvelu

tilanteissa esille nousseet aiheet tai varata aikoja neuvotteluihin tai tapaamisiin. Procountor API -moduulilla luotu rajapinta taloushallinnon ohjelmistolle antaa myös paljon joustavuutta yritykselle, koska yritys voi pitää Odoota heidän pääasiallisena laskutusjärjestelmänänsä. Näiden uudistusten avulla yrityksen myynnillä on vähemmän sovelluksia, joita täytyy avata ja uudella toiminnanohjausjärjestelmällä saadaan suorittaa samat tehtävät yhdellä sovelluksella.

LÄHTEET

Anicare 2016. Tahdosta auttaa porotaloutta. Hakupäivä 11.1.2024. https://anicare.fi.

Cemazar, Sara Ana 2022. 10 biggest advantages of open-source software. Rocket.chat. Hakupäivä 3.2.2024. https://www.rocket.chat/blog/open-source-software-advantages.

Cybrosys Technologies 2019. ORM (Object Relational Mapping) in Odoo. Hakupäivä 18.3.2024. https://www.cybrosys.com/blog/orm-in-odoo.

Cybrosys Technologies 2023. How to install a custom module in Odoo 16. Hakupäivä 9.4.2024. https://www.cybrosys.com/blog/how-to-install-a-custom-module-in-odoo-16.

Cybrosys Technologies 2024a. Contacts- Odoo 16 Enterprise Book. Hakupäivä 8.4.2024. https://www.cybrosys.com/odoo/odoo-books/odoo-book-v16/contacts/.

Cybrosys Technologies 2024b. Odoo 16 Development book. Hakupäivä 21.3.2024. https://www.cybrosys.com/odoo/odoo-books/odoo-16-development/creating-odoo-modules/.

Genius ERP 2024. A Brief History of ERP. Hakupäivä 2.3.2024. https://www.geniuserp.com/resources/blog/a-brief-history-of-erps.

Gitlin, Jon 2024. ERP API integration: definition, example, tools, and more. Merge. Hakupäivä 5.3.2024. https://www.merge.dev/blog/erp-api-integration.

Jackley, Mark 2023. 10 Top ERP Modules and Their Features. Oracle. Hakupäivä 10.3.2024. https://www.oracle.com/erp/erp-modules/.

Khalid, Faiza 2024. Top 10 ERP Systems in 2024. DevTeam.Space. Hakupäivä 16.1.2024. https://www.devteam.space/blog/top-erp-systems/.

McCue, Ian 2022. ERP Modules: Types, Features & Functions. Oracle NetSuite. Hakupäivä 3.2.2024. https://www.netsuite.com/portal/resource/articles/erp/erp-modules.shtml.

Odoo 2024a. Compare Editions. Hakupäivä 16.1.2024. https://www.odoo.com/page/editions.

Odoo 2024b. Finally, a modern inventory system. Hakupäivä 7.4.2024. https://www.odoo.com/app/inventory.

Odoo 2024c. Trial. Hakupäivä 13.2.2024. https://www.odoo.com/trial.

Odoo docs 2024a. Access rights. Hakupäivä 8.4.2024. https://www.odoo.com/documentation/17.0/applications/general/users/access_rights.html.

Odoo docs 2024b. Fields and widgets. Hakupäivä 21.3.2024. https://www.odoo.com/documentation/16.0/applications/studio/fields.html.

Odoo docs 2024c. Views. Hakupäivä 21.3.2024. https://www.odoo.com/documentation/16.0/developer/reference/backend/views.html.

Odoo forum 2022. What is the Odoo programming language? How to learn Odoo Development? Hakupäivä 3.2.2024. https://www.odoo.com/forum/help-1/what-is-the-odoo-programming-language-how-to-learn-odoo-development-197369.

Pinckaers, Fabien 2023. Making companies a better place, one app at a time. Odoo. Hakupäivä 10.1.2024. https://www.odoo.com/blog/odoo-news-5/the-odoo-story-56.

PostgreSQL 2024. About. Hakupäivä 18.3.2024. https://www.postgresql.org/about/.

Procountor 2024a. Connect with Procountor. Hakupäivä 16.4.2024. https://dev.procountor.com/.

Procountor 2024b. Kaikki mitä tarvitset yrityksesi taloushallintoon. Hakupäivä 16.4.2024. https://procountor.fi/procountor/hinnasto/.

QAD 2024. A guide to enterprise resource planning (ERP). Hakupäivä 7.4.2024. https://www.qad.com/what-is-erp.

Tipalti 2024. The total guide to Odoo ERP. Hakupäivä 7.4.2024. https://tipalti.com/erp-integrations/odoo-erp/.

Vault-ERP 2024. Modular ERP: advantages and disadvantages. Hakupäivä 7.3.2024. https://blog.vault-erp.com/post/2021/10/21/modular-erp-advantages-and-disadvantages.

W3Techs 2024. Overview of web technologies used by Odoo.com. Hakupäivä 18.3.2024. https://w3techs.com/sites/info/odoo.com.