VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Valtteri Kerttula

AI JA LUONNOLLISEN KIELEN KÄSITTELY

Luonnollisen kielen käsittely ja sen hyödyntäminen osana energiaseurannan sovellusta

Tekniikka
2023

VAASAN AMMATTIKORKEAKOULU
Tietotekniikka

TIIVISTELMÄ

Tekijä	Valtteri Kerttula
Opinnäytetyön nimi	AI ja luonnollisen kielen käsittely
Vuosi	2023
Kieli	suomi
Sivumäärä	49 + 25 liitettä
Ohjaaja	Anna-Kaisa Saari

Luonnollisen kielen käsittely (engl. Natural Language Processing, NLP) tarjoaa tehokkaita toimintoja ja työkaluja sovelluksille. Opinnäytetyön tavoitteena oli tutkia NLP-teknologian tuomia mahdollisuuksia, sekä suunnitella ja toteuttaa NLP-teknologiaa hyödyntävä mikropalvelu Suomen kehittyneempään energiaseurannan sovellukseen, EcoReactioniin. Opinnäytetyössä tutkittiin myös yleisesti NLP-teknologian tuomia haasteita ja mahdollisuuksia.

Luonnollisen kielen käsittely on tutkimusalueena laaja ja se perustuu teoreettisen kielitieteen ja tietojenkäsittelytieteen tekniikoihin ja menetelmiin. Opinnäytetyössä tutkittiin luonnollisen kielen käsittelyn teoriapohjaa kielitieteen teoreettisesta näkökulmasta, sekä tietojenkäsittelytieteen teknisestä näkökulmasta. Keskeisiä käsitteitä opinnäytetyössä ovat muun muassa NLP, NLU, NLG ja NLI. Tutkimusaineisto koostuu ajankohtaisista tutkimusaihetta koskevista kirjoista, artikkeleista sekä verkkolähteistä.

NLP-teknologiaa hyödyntävä mikropalvelu toteutettiin onnistuneesti. Samalla tehtiin havaintoja NLP-teknologioita tarjoavien palveluiden haasteista ja mahdollisuuksista. Opinnäytetyön aikana opittiin NLP-teknologian tehokkaan hyödyntämisen käytännöt asiakaskohtaisen datan tarjoamiseen. Lopuksi käytiin läpi toteutettu mikropalvelu, suunnitteluun vaikuttavat tekijät sekä opinnäytetyötä tehdessä havaitut jatkokysymykset ja oivallukset.

Avainsanat	NLP, NLU, NLG, NLI, luonnollisen kielen prosessointi, luonnollisen kielen ymmärtäminen, luonnollisen kielen tuottaminen, luonnollisen kielen käyttöliittymä, tekoäly, Wit.ai

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikka

ABSTRACT

Author	Valtteri Kerttula
Title	AI and Natural Language Processing
Year	2023
Language	Finnish
Pages	49 + 25 Appendices
Name of Supervisor	Anna-Kaisa Saari

Natural Language Processing (NLP) offers effective functions and tools for applications. The aim of the thesis is to explore the possibilities brought by NLP technology and to design and implement a microservice that utilizes NLP technology for a more advanced energy monitoring application in Finland, EcoReaction. The challenges and opportunities of NLP technology were also examined in the thesis.

The field of Natural Language Processing is wide-ranging and is based on techniques and methods from theoretical linguistics and computer science. In this thesis, the theoretical foundation of natural language processing is examined from both a linguistic and technical perspective. Key concepts in this thesis include NLP, NLU, NLG, and NLI. The research material consists of current literature, articles, and online sources related to the research topic.

A microservice utilizing NLP technology was successfully implemented. During the project, observations were made on the challenges and opportunities of NLP technologies. Effective practices for utilizing NLP technology to provide customer-specific data were learned. Finally, the implemented microservice, factors affecting the design, and further questions and insights discovered during the project are discussed.

Keywords	NLP, NLU, NLG, NLI, natural language processing, natural language understanding, natural language generation, natural language interface, artificial intelligence, Wit.ai

SISÄLLYS

KUVIO- JA TAULUKKOLUETTELO

LIITELUETTELO

LIITE 1. Salassa pidettävä aineisto. Toiminnallisuuksien havainnollistaminen syötteen tulkitsemisessa.

LIITE 2. Salassa pidettävä aineisto. Esimerkki tavoitteista, joita pyritään toteuttamaan energiaseurannan sovellukseen.

LIITE 3. Salassa pidettävä aineisto. Projektin arkkitehtuuria havainnollistettu sekvenssikaaviolla.

LIITE 4. Salassa pidettävä aineisto. Projektin tiedostorakennetta kuvattuna.

LIITE 5. Salassa pidettävä aineisto. Esimerkki mikropalvelun tarjoamaan rajapintaan tehtävästä pyynnöstä.

LIITE 6. Salassa pidettävä aineisto. Esimerkki mikropalvelun vastauksesta pyynnön syötteen perusteella.

LIITE 7. Salassa pidettävä aineisto. Esimerkki NLP-teknologian tuottamasta vastauksesta syötteen perusteella.

LIITE 8. Salassa pidettävä aineisto. Kuvakaappaus Swagger-työkalun tuottamasta dokumentaatiosta koskien mikropalvelun rajapintaa.

LIITE 9. Salassa pidettävä aineisto. Tietokannan luontiskripti tietokannan rakenteen luomiseksi.

LIITE 10. Salassa pidettävä aineisto. Tietokantakuvaus.

LIITE 11. Salassa pidettävä aineisto. Ohjain (engl. controller) -luokan kuvaus.

LIITE 12. Salassa pidettävä aineisto. Palvelu (engl. service) -luokka käsittelee sille saapuneen pyynnön.

LIITE 13. Salassa pidettävä aineisto. Wit.ai:n vastauksen käsittely.

LIITE 14. Salassa pidettävä aineisto. WitAiResponse -luokka.

LIITE 15. Salassa pidettävä aineisto. Käyttäjän aiempien viestien käsittely.

LIITE 16. Salassa pidettävä aineisto. Palvelu -tason luokka käsittelee ymmärretyn syötteen, ja tekee toiminnallisuuksia ”trait” -tyypin perusteella.

LIITE 17. Salassa pidettävä aineisto. Aikaisempien viestien käsittelyyn tarkoitetun palvelun esittelyluokka.

LIITE 18. Salassa pidettävä aineisto. Aikaisempien viestien käsittelyyn tarkoitetun palvelun toteutusluokka.

LIITE 19. Salassa pidettävä aineisto. Metodi, joka tarkistaa onko viestin piirre määritelty vastaamaan toimenpideluokkaa.

LIITE 20. Salassa pidettävä aineisto. Aikaisempien viestien käsittelyyn tarkoitetun palvelun toteutusluokka.

LIITE 21. Salassa pidettävä aineisto. Toimenpideluokkien määrittely (liitteessä ”actionHandler”).

LIITE 22. Salassa pidettävä aineisto. Abstrakti luokka, josta toimenpideluokat periytyvät.

LIITE 23. Salassa pidettävä aineisto. Esimerkki toimenpideluokasta, joka hyödyntää energiaseurannan sovelluksen käyttäjäkohtaista dataa vastauksen tuottamiseen.

LIITE 24. Salassa pidettävä aineisto. Esimerkki luokasta, jossa määritellyt piirre tyypit esitellään.

LIITE 25. Salassa pidettävä aineisto. Testiluokka, jossa on toteutettu yksikkötestejä JUnit- ja Mockito-kirjastoja hyödyntäen.

TERMIT JA LYHENTEET

API	Ohjelmointirajapinta (engl. *Application Programming Interface*).
BETA	Ohjelmiston julkaisuelinkaareen kuuluva julkaisuvaihe, jossa ohjelmisto alkaa lähellä varsinaisen vakaan version julkaisua, mutta sisältää vielä puutteita.
DESERIALISOINTI	Menetelmä, jossa serialisoiti data muutetaan ohjelmallisesti ymmerretyksi dataobjektiksi.
ENTITEETTI	Yleistermi sille, jonka kielellinen ilmaus voi havaita tai tulkita.
HTTP	Tiedonsiirtoon käytettävä protokolla (engl. *Hypertext Transfer Protocol*).
INTENTIO	Aikomus tai pyrkimys.
JSON	Standardin mukainen tietoformaatti (engl. *JavaScript Object Notation*).
KONTEKSTUAALISUUS	Tekstin yhteys muihin teksteihin ja kielenulkoiseen maailmaan.
MIGRAATIO	Siirtymistä yhdestä teknologisesta ratkaisusta toiseen.
MIKROPALVELU	Pieni itsenäinen prosessi, joka kommunikoi muiden prosessien kanssa.

NLG	Luonnollisen kielen tuottaminen (engl. *Natural Language Generation*).
NLI	Luonnollisen kielen käyttöliittymä (engl. *Natural Language Interface*).
NLP	Luonnollisen kielen käsittely (engl. *Natural Language Processing*).
NLU	Luonnollisen kielen ymmärtäminen (engl. *Natural Language Understanding*).
REST	Arkkitehtuurityyli ohjelmointirajapintojen toteuttamiseen (engl. *Representational State Transfer*).
SERIALISOINTI	Menetelmä, jossa dataobjekti muutetaan helposti siirrettävään muotoon, kuten JSON-tietoformaattiin.
TOIMENPIDELUOKKA	Määritelmä, jolla tarkoitetaan liitteissä esiintyviä ”actionHandler” -luokkia.

1 JOHDANTO

Energiankulutuksen seuranta ja hallinta ovat tärkeitä tekijöitä kestävän kehityksen ja ympäristöystävällisen toiminnan kannalta. Luonnollisen kielen käsittely (engl. Natural Language Processing, NLP) ja sen hyödyntäminen voi tarjota tehokkaita työkaluja ja toiminnallisuuksia sovelluksille, jotka pyrkivät helpottamaan ja parantamaan energian seurantaa.

Opinnäytetyön toimeksiantaja on Wapice Oy, joka tarjoaa Suomen suosituimman energian seuranta- ja raportointityökalun, EcoReactionin. Opinnäytetyön tavoitteena on tutkia NLP-teknologian tarjoamia mahdollisuuksia ja toteuttaa EcoReactioniin mikropalvelu, joka tarjoaa luonnollisen kielen käsittelyyn ohjelmointirajapinnan (engl. Application Programming Interface, API). Tämän mikropalvelun avulla voidaan tulkita ja analysoida käyttäjän syöte, sekä muodostaa tarkoituksenmukainen vastaus NLP-teknologian tuottaman analyysin perusteella. Tämä mikropalvelu mahdollistaa kokonaan uuden luonnollisen kielen käyttöliittymän (engl. Natural Language Interface, NLI) EcoReactionin tarjoaman datan tarkasteluun.

Työssä käydään lyhyesti läpi luonnollisen kielen prosessointiin liittyvää teoriaa, perusteita ja menetelmiä sekä erilaisia teknologiavaihtoehtoja. Työn painopisteenä on tarkastella NLP-teknologioiden tuomia mahdollisuuksia ja haasteita, sekä kykyä integroida teknologia osaksi olemassa olevaa sovellusta. Työssä vertaillaan myös erilaisia NLP-teknologioita tarjoavia palveluita, kuten Dialogflow, Amazon Lex, LUIS ja Azure Cognitive Service for Language, jotka tarjoavat opinnäytetyön tavoitteen vaatimia toiminnallisuuksia ja käyttötapauksia. Vertailun avulla pyritään saamaan syvempi ymmärrys eri palveluiden tarjoamista ominaisuuksista.

Opinnäytetyön tavoitteena on löytää parhaiten soveltuva NLP-teknologiaa tarjoava palvelu projektin tarkoitukseen ja käyttötapaukseen. Lisäksi myös tutkia, miten NLP:tä on mahdollista hyödyntää olemassa olevassa palvelussa. NLP-

teknologioita voidaan käyttää sekä kirjoitetun, että puhutun kielen analysointiin, mutta tämä työ käyttötapauksineen keskittyy kirjoitetun kielen prosessointiin ja ymmärtämiseen.

Luonnollisen kielen ymmärtäminen (engl. Natural Language Understanding, NLU) on osa NLP-teknologiaa. Tutkielmassa esitellään yksi mahdollinen toteutustapa siitä, miten NLU-teknologioilla voidaan sujuvoittaa sovelluksen tarjoaman palvelun saatavuutta. Opinnäytetyössä tarkastellaan EcoReactioniin toteutetun mikropalvelun arkkitehtuuria, testausta sekä käyttäjän syötteen pohjalta tehtävien valintojen logiikkaa. Lisäksi tarkastelussa on myös tämän käyttötapauksen teknologiavalintojen tuomat hyödyt ja haasteet. Tutkielman lopussa perehdytään yleisesti NLP-teknologioiden tuomiin haasteisiin ja mahdollisuuksiin.

2 YHTEISTYÖKUMPPANI

Luvussa käsitellään toimeksiantajaa, Wapice Oy:tä, joka on suomalainen ohjelmistotalo. Lisäksi luvussa käsitellään EcoReactionia, johon opinnäytetyössä käsitelty NLP-teknologiaa tarjoava mikropalvelu on integroitu.

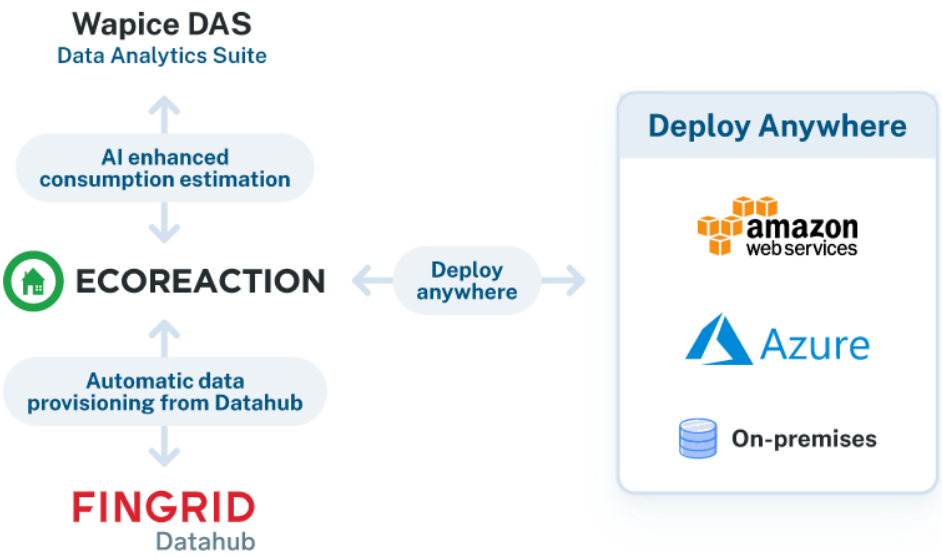
2.1 Wapice Oy

Wapice on täyden palvelun ohjelmistoyritys, jonka toteuttamia ratkaisuja on käytössä alan johtavilla teollisuusyrityksillä ympäri maailmaa. Wapice tarjoaa palveluja muun muassa analytiikan, tekoälyn, big datan, tietoturvan, tietoteknisen konsultoinnin, sulautettujen järjestelmien ja älykkäiden energiapalveluiden parissa. Palveluiden lisäksi Wapicen tunnetuimpia tuotteita ovat muun muassa IoT-TICKET, Summium ja EcoReaction.[1]

2.2 EcoReaction

EcoReaction on Suomen johtava ja edistynein energiaraportointijärjestelmä. Se on suunnattu muun muassa isoille energiayhtiöille, mutta asiakkaisiin lukeutuu monilla eri markkinoilla toimivia yhtiöitä. EcoReaction antaa energiayhtiölle mahdollisuuden tarjota asiakkailleen tehokkaat työkalut energiakäyttäytymisen seurantaan ja tarkkailuun. Se toimii joko asiakkaan valitsemalla paikallisella palvelimella tai pilvipohjaisena palveluna (**Kuvio 1.**), jolloin se tarjoaa nopean skaalautuvuuden käyttäjäkuorman muutoksiin sekä saumattoman päivitysprosessin ilman katkoksia.[1]

[1] Wapice Oy. 2023. EcoReaction.



Kuvio 1. EcoReaction esittely.[1]

3 LUONNOLLISEN KIELEN KÄSITTELY

Luonnollisella kielellä, eli ihmisryhmän käyttämällä kielellä, tarkoitetaan kieltä, joka on kehittynyt luonnollisin tavoin.[2] Keinotekoisina kielinä voidaan pitää esimerkiksi ohjelmointikieliä. Luonnollisen kielen käsittelyn tutkimusalue on laaja ja se perustuu erilaisten tieteenalojen, kuten teoreettisen kielitieteen ja tietojenkäsittelytieteen tekniikoihin ja oivalluksiin, mutta siihen vaikuttaa myös matemaattinen logiikka, sekä psykologia.[3] Voidaan siis todeta, että luonnollisen kielen käsittely, eli NLP, viittaa tietokoneen kykyyn käsitellä ihmisten käyttämää luonnollista kieltä.[4]

Luvussa tarkastellaan lyhyesti luonnollisen kielen käsittelyn teoriaa, sekä tarkastellaan muutamia NLP-teknologian osa-alueita, jotka ovat opinnäytetyön tavoitteeseen liittyen keskiössä.

3.1 Teoria

NLP-teknologiat perustuvat useisiin periaatteisiin ja tekniikoihin, joita voidaan käyttää luonnollisen kielen käsittelyn sovellusten kehittämisessä. Luonnollisen kielen prosessoinnissa ja ymmärtämisessä on useita keskeisiä osa-alueita, kuten morfologia, kielioppi, semantiikka, pragmatiikka, syntaksi ja kontekstuaalisuus sekä koneoppiminen.

Morfologia käsittelee sanojen muotoja ja niiden taivutusta, kuten juurimuodon, suffiksien ja prefiksien analysointia.[5]

---

[2] Tieteen termipankki 2023. Kielitiede: Luonnollinen kieli.
[3] University of London 2013. Introduction to natural language processing.
[4] Jyväskylän yliopisto. 2019. Graph-based exploration and clustering analysis of semantic spaces.
[5] Karlsson, F. Yleinen kielitiede. 2012. Kustannusosakeyhtiö Gaudeamus. s. 83.

Kielioppi käsittelee sanojen rakennetta ja niiden järjestäytymistä lauseiksi, kuten sanaluokkien määrittämistä ja lauserakenteiden analysointia.[6]

Semantiikka keskittyy kielen merkitykseen, kuten sanojen, lauseiden ja tekstien merkityksen ymmärtämiseen, synonyymien ja antonyymien, konnotaatioiden ja denotaatioiden sekä sanastollisen semantiikan analysointiin.[7]

Pragmatiikka käsittelee kielen käytön kontekstuaalisia ja sosiaalisia näkökohtia, kuten viestintätilanteiden analysointia.[8]

Syntaksi keskittyy lauseiden rakenteisiin ja niiden sääntöihin, kuten lauseenjäsenten, lausekkeiden ja lauseenjäsennysten analysointiin.[9]

Kontekstuaalisuus, on tärkeä tekijä luonnollisen kielen ymmärtämisessä. Se viittaa siihen, että kielen merkitys ja tulkinta riippuvat laajemmasta yhteydestä, missä kieltä käytetään. Kontekstuaalisuus vaikuttaa luonnollisen kielen ymmärtämiseen useilla tavoilla. [10]

Koneoppiminen on myös olennainen osa NLP:tä. Siihen sisältyy erilaisia menetelmiä ja algoritmeja, kuten esimerkiksi sentimenttianalyysi, tekstin luokittelu, entiteettien tunnistus ja kielen tuottaminen. Koneoppimisen avulla tietokoneet voivat oppia ja soveltaa sääntöjä kielen käsittelyssä suuresta datamäärästä.[11]

---

[6] Roinila, M. G. W. Leibnizin rationaalinen kielioppi. 2023.
[7] Tampereen yliopisto. Kielen osajärjestelmät. 2009.
[8] Tieteen termipankki. 2023. Kielitiede: Pramagtiikka.
[9] Tieteen termipankki. 2023. Kielitiede: Syntaksi.
[10] Krupsky, S. 2021. Natural Language Processing Introduction: what is Natural Language Processing (NLP)? NLP Cloud. Blogi.
[11] Hamborg, F & Karsten, D. 2021. NewsMTSC: A Dataset for (Multi-)Target-dependent Sen-timent Classification in Political News Articles. *In Proceedings of the 16th Conference of the Euro-pean Chapter of the Association for Computational Linguistics: Main Volume*.

Voidaan todeta, että luonnollisen kielen käsittelyssä kielitieteet ja tietojenkäsittelytieteet ovat keskeisessä roolissa. Kielitieteet tarjoavat ymmärrystä kielen rakenteista, kuten morfologiasta, kieliopista, syntaksista, pragmatiikasta ja kontekstuaalisuudesta, mikä auttaa analysoimaan kieltä ja sen ominaisuuksia. Tietojenkäsittelytieteet puolestaan tarjoavat tekniikoita ja menetelmiä kielen automaattiseen prosessointiin, kuten koneoppimiseen, luokitteluun, tiedonhaun ja käsityksen toteuttamiseen.

3.2 Tekniikat ja menetelmät

Luonnollisen kielen käsittely jakautuu useampaan eri osa-alueeseen ja menetelmään, joita voidaan hyödyntää erilaisissa käyttötapauksissa. Seuraavaksi käydään läpi keskeisiä menetelmiä ja tekniikoita, joita hyödynnetään toteutuneessa sovelluksessa.

3.2.1 NLU - Luonnollisen kielen ymmärtäminen

Luonnollisen kielen ymmärtäminen on keskeinen tekniikka luonnollisen kielen käsittelyssä, ja se mahdollistaa tekstin ymmärtämisen ja analysoinnin syvällisellä tasolla. NLU-menetelmät keskittyvät kielen ymmärtämiseen ja sen syvälliseen analysointiin. NLU-menetelmät voivat sisältää tekstin segmentoinnin, sanaluokkien tunnistamisen, nimien tunnistamisen, syntaktisen parsinnan ja semanttisen roolin tunnistamisen, mitkä auttavat ymmärtämään tekstin merkityksen ja rakenteen[12].

[12] Navigli, R. 2015. Department of Computer Science. Natural Language Undestanding: Instructions for (Present and Future).

Luonnollisen kielen ymmärtäminen on opinnäytetyön käyttötapauksessa keskeisessä roolissa. Käyttäjän syötteen ymmärtämisellä ja sen pohjalta tehdyillä toimilla, voidaan tarjota käyttäjälle tämän haluama data nopeasti ja tehokkaasti.

3.2.2 NLG - Luonnollisen kielen tuottaminen

Luonnollisen kielen tuottaminen (engl. Natural Language Generation, NLG), eli NLG-menetelmät, keskittyvät kielen tuottamiseen koneellisesti. NLG-menetelmät voivat sisältää tekstin luonteen, rakenteen ja tyylin tuottamisen eri tarkoituksiin, kuten raporttien, uutisten tai vastausten tuottamiseen.[13]

Yksi yleinen sovellusalue NLG-menetelmille on vastausten tuottaminen ennaltamäärättyihin vastauspohjiin perustuen. Tässä lähestymistavassa vastauspohjat luodaan etukäteen, jolloin sovellus käyttää vastauspohjia tuottaessaan oikeat vastaukset eri kysymyksiin tai tilanteisiin. Tämä voi olla hyödyllistä esimerkiksi chatboteissa, asiakaspalvelu- tai kysymys-vastaus-sovelluksissa, joissa tiettyihin kysymyksiin on valmiiksi määritellyt vastaukset.[14] Opinnäytetyön käyttötapauksessa on hyödynnetty ennaltamääriteltyjä vastauspohjia vastauksen tuottamiseen, tämän menetelmän tehokkuuden ja yksinkertaisuuden takia.

Vaativissa käyttökohteissa hyödynnetään usein koneoppimista, erityisesti syväoppimista. Nämä menetelmät mahdollistavat käyttämään suuria määriä tekstidataa mallin kouluttamiseen. Tämän ansiosta NLG-menetelmä voi tuottaa

[13] Dale, R., Di Eugenio, B. & Scott, D. 1998. Introduction to Special Issue on Natural Language Generation.

[14] Jones, K., Altancu, E., Virginia, N. Franquiera, L., Wang, Y., & Shujun, L. 2022. University of Cyber Security for Society. A Comprehensive Survey of Natural Language Generation Advances from the Perspective of Digital Deception.

monipuolisempaa tekstiä, joka perustuu laajaan opetettuun dataan, mukaan lukien käyttäjän antamaan syötteeseen.[14]

3.2.3 NLI – Luonnollisen kielen käyttöliittymä

Luonnollisen kielen käyttöliittymä mahdollistaa käyttäjän ja sovelluksen vuorovaikutuksen luonnollisen kielen avulla, ilman erillistä graafista käyttöliittymää. NLI siis tarjoaa käyttäjille helpon tavan kommunikoida sovelluksen kanssa. NLI:n avulla käyttäjä voi esittää kysymyksiä, antaa komentoja tai ilmaista tarpeitaan luonnollisella kielellään käyttöliittymälle.[15]

3.3 Käyttökohteet

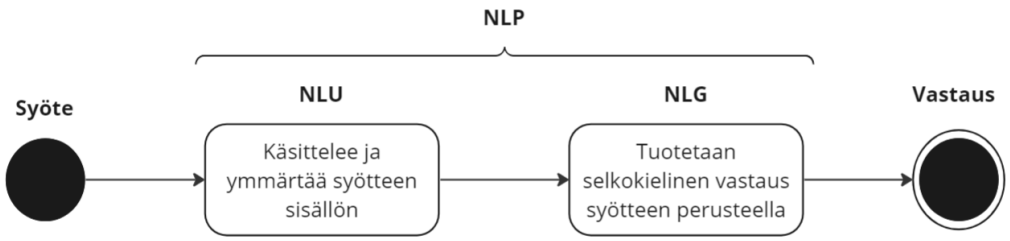
NLP-teknologioille on useita käyttökohteita ja niitä voidaan soveltaa monissa eri kokonaisuuksissa. Yksi merkittävä käyttökohde on tiedonhaun parantaminen. NLP-menetelmiä voidaan käyttää tiedonhaussa, tietokantojen kyselyissä, dokumenttien analysoinnissa ja tiedonhallintajärjestelmissä. NLP voi auttaa organisaatioita löytämään ja järjestämään suuria määriä tekstidataa nopeasti ja tehokkaasti, mikä voi parantaa päätöksentekoa ja tietojen hyödyntämistä.[16]

NLP-teknologioita voidaan hyödyntää myös terveydenhuollosta. Esimerkiksi potilaiden kirjalliset sairauskertomukset ja hoitoraportit sisältävät valtavia määriä tekstidataa, jota voidaan analysoida NLP-menetelmien avulla. Tätä menetelmää voidaan käyttää tukena potilastietojen analysointiin, diagnoosien tekemiseen, hoitosuunnitelmien kehittämiseen ja hoitotulosten seurantaan.[17]

---

[15] Halfsa, S. D., Ikramullah M. L., Khalid M. M. & Bukhari S. A. C. 2023. Frameworks for Querying Databases Using Natural Language: A Literature Review.
[16] Carlstedt, M. 2017. Mälarden University. Using NLP and context for improved search result in specialized search engines.
[17] Jain, K. 2021. NLP/Deep Leep Techniques in Healthcare for Decision Making.

Yksi merkittävä, ja tämän opinnäytetyön kannalta tärkeä, käyttöalue NLP-teknologioille, on asiakaspalvelu ja viestintä. NLP-menetelmiä voidaan hyödyntää chatbotien, virtuaalisten avustajien ja muiden automaattisten asiakaspalveluratkaisujen kehittämisessä. NLP:n avulla voidaan ymmärtää asiakkaiden kysymyksiä, kommentteja ja palautetta, sekä vastata niihin nopeasti ja tarkasti (**Kuvio 2.**). Lisäksi NLP-menetelmät voivat auttaa tunnistamaan asiakkaiden tunteita ja mielipiteitä sosiaalisesta mediasta ja muista avoimista lähteistä, mikä voi auttaa yrityksiä saamaan arvokasta tietoa asiakaskokemuksen parantamiseksi.[18]



Kuvio 2. Käyttäjän syötteen ymmärtäminen (NLU) ja vastauksen tuottaminen (NLG), ovat osa NLP-teknologiaa.

NLP-teknologian avulla voidaan siis tarjota kokonaan uusi käyttöliittymä, NLI, käyttäjän ja sovelluksen välille. NLI mahdollistaa sen, ettei esimerkiksi energia-käyttäytymisen tarkasteluun vaadita erillistä sovellusta tai perinteistä graafista käyttöliittymää. On kuitenkin hyvä muistaa, että graafinen käyttöliittymä tarjoaa omat vahvuutensa, varsinkin kun tarkastellaan monimuotoisia ja monikerroksisia sovelluksia. Voidaan todeta, että NLP-teknologioiden käyttökohteet ovat laajat ja monipuoliset, ja ne jatkavat kehittymistään uusien innovaatioiden ja sovellusten myötä.

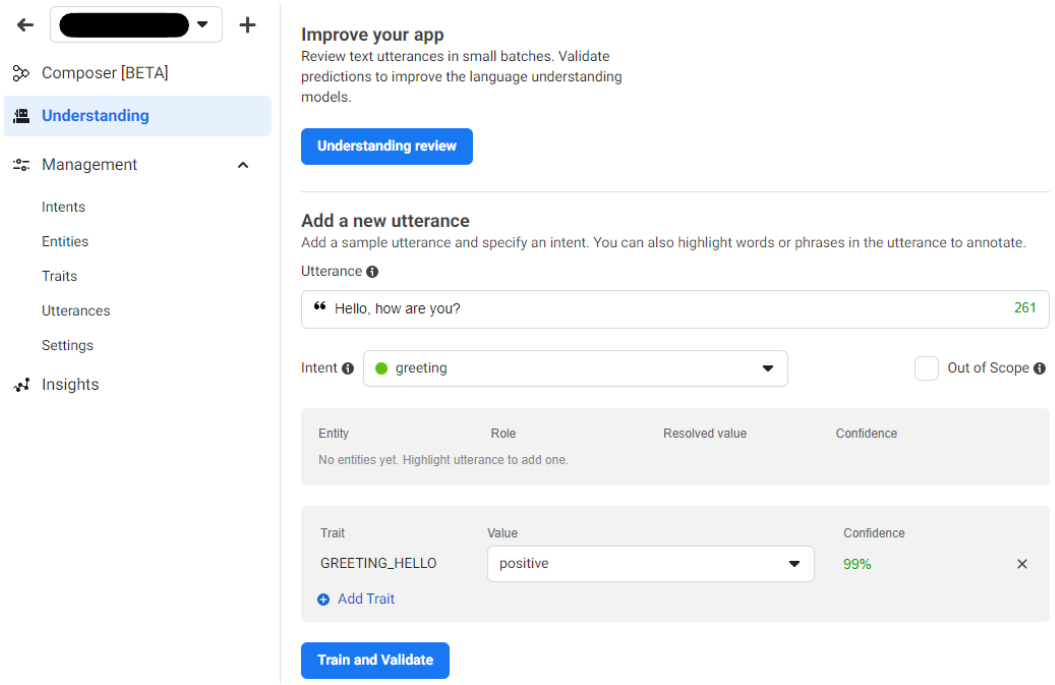
[18] Mashaabi, M., Alotaibi, A., Qudaih, H., Alnashwan, R. & Al-Khalifa, H. 2021.

4 WIT.AI

Wit.ai on yksi ensimmäisistä NLP-teknologiaa hyödyntävistä alustoista, joka on tarjonnut avoimen lähdekoodin ratkaisuja kehittäjille. Wit.ai valikoitui NLP-teknologian tarjoajista opinnäytetyön käyttötapaukseen. Wit.ai on erikoistunut ennen kaikkea luonnollisen kielen ymmärtämiseen. Wit.ai tarjoaa kehittäjille rajapinnan, jonka avulla voidaan toteuttaa sovelluksia, jotka ymmärtävät ja analysoivat ihmisen kirjoittamaa tai puhuttua kieltä. Wit.ai perustettiin vuonna 2013 ranskalaisen teknologiayrityksen Wit.ai Inc:n toimesta ja myöhemmin se myytiin teknologiajätti Metalle. Omistajanvaihdos on tuonut Wit.ai:lle lisää resursseja ja osaamista, jotka mahdollistavat entistä vahvemman ja paremman NLP-teknologian kehittämisen.[19] [20]

Wit.ai on tullut tunnetuksi innovatiivisena ja kehittyvänä NLP-teknologian tarjoajana, ja se on kasvattanut kehittäjäyhteisöään suureksi, yli 300 000 kehittäjän joukoksi.[19] Suuri kehittäjäyhteisö tuo mahdollisuuden hyödyntää yhteisön kokemusta ja ratkaisua osana sovelluskehitystä. Wit.ai tarjoaa kokonaisuudessaan paljon hyviä ominaisuuksia, kuten selkeän käyttöliittymän (**Kuvio 3.**), sekä kattavan dokumentaation, jotka mahdollistavat teknologian tehokkaan kehityksen ja käytön.

[19] Wit.ai. Getting Started With Wit.ai. 2023.
[20] Wit.ai is joining Facebook. 2015.



Kuvio 3. Wit.ai:n käyttöliittymä kuvattuna 04/2023.

4.1 Teknologian valinta

Monien vaihtoehtojen vertailun jälkeen Wit.ai tuli valituksi tähän työhön useista syistä. Wit.ai:n integroiminen moniin erilaisiin sovelluksiin ja palveluihin on tehty todella joustavaksi sen tarjoaman RESTful-rajapinnan ansiosta. Myös avoimen lähdekoodin luonne sekä dokumentaation selkeys auttavat kehittäjää pääsemään nopeasti alkuun. Wit.ai tarjoaa opinnäytetyön tekohetkellä kirjoitetun kielen analysointiin tuen 132 kielelle, sekä tuen puhutun kielen ymmärtämiseen yhteensä 32 kielelle. Laaja tuki eri kielille on erittäin hyödyllistä sovelluksissa, joissa käsitellään monikielisiä syötteitä ja käyttäjävuorovaikutuksia. Kaikkien eri kielien toimivuutta ei tarkastella tässä työssä, mutta dokumentaatio kertoo monen kielen olevan BETA-vaiheessa. Opinnäytetyössä käsitellään suomen, ruotsin sekä englannin kieltä. Niiden kanssa työskennellessä ei havaittu ongelmia.

Avoin lähdekoodi tarkoittaa, että Wit.ai:n lähdekoodiin on vapaa pääsy. Avoin lähdekoodi johtaa siihen, että suurella kehittäjäyhteisöllä on mahdollisuus tutkia, kehittää ja antaa palautetta, joka takaa jatkuvan kehityksen. Avoin lähdekoodi

myös mahdollistaa kustomoinnin ja laajentamisen tarpeiden mukaan, mikä on erityisen houkuttelevaa monimutkaisten projektien vaatimusten kannalta.

Metan tuki ja sitoutuminen Wit.ai:hin on myös tärkeä tekijä sen pitkän aikavälin kehityksen ja tuen kannalta. Meta on suuri teknologiayritys, jolla on resursseja ja asiantuntemusta NLP-teknologiassa. Meta on sitoutunut jatkamaan Wit.ai-alustan kehitystä ja ylläpitoa pitkällä aikavälillä, mikä tarjoaa luottamusta sen kestävään tukeen ja päivityksiin tulevaisuudessa. Näiden ansiosta Wit.ai tarjoaa mahdollisuuden pitkäaikaiseen tukeen ja kehitykseen.[18]

Wit.ai on hyvä valinta yksityishenkilöille sekä yrityksille, sillä teknologian käyttö on ilmaista niin ei-kaupallisissa kuin kaupallisissakin käyttökohteissa. Wit.ai on toteuttanut viralliset rajapinnat Node.js-, Python-, Ruby-, Go- sekä Unity-ohjelmointikielille.[18] Avoimen lähdekoodin ansiosta myös muille ohjelmointikielille on rakennettu epävirallisia rajapintoja. Seuraavaksi tarkastellaan NLP-teknologiaa tarjoavia vertailukohteita, joiden joukosta Wit.ai on valikoitunut.

4.1.1 Vertailukohteet

Dialogflow on Google Cloudin tarjoama NLP-teknologia. Palvelun etuja ovat sen helppokäyttöisyys, dokumentaatio ja laaja kielituki 122 eri kielelle. Dialogflow mahdollistaa puhutun sekä kirjoitetun kielen analysoinnin. Se ei tarjoa avoimen lähdekoodin ratkaisua, eikä se ole ilmainen kaupalliseen käyttöön. Suuria määriä syötteitä käsiteltäessä, kannattaa huomioida, että palvelun kustannukset koostuvat esimerkiksi rajapintaan tehtävien pyyntöjen määrästä.[21]

[21] Natural Language AI. Google Cloud. 2023.

Amazon Lex on Amazon Web Servicen (AWS) tarjoama NLP-teknologia. Palvelun yksi merkittävimmistä eduista on sen integraatiomahdollisuudet AWS-ekosysteemiin, skaalautuvuus sekä valmiit mallit yleisiin keskustelusovelluksiin. Amazon Lex tarjoaa laajan tuen eri ohjelmointikielille, mutta sen luonnollisen kielen tuki oli vertailukohteiden suppein 13:sta kielellään. Palvelu ei ole avointa lähdekoodia eikä se tue suomen kieltä. Palvelun käytöstä koituu myös kustannuksia rajapinnan käsittelemiin pyyntöihin pohjautuen.[22]

LUIS, eli Language Understanding Intelligent Service on Microsoftin kehittämä NLP-teknologia. Palvelu ei tarjoa tukea suomen kieleen ja sen käytöstä koituu kustannuksia. Palvelun etuja ovat sen helppokäyttöisyys sekä integraatiomahdollisuudet Microsoftin Azure-ekosysteemiin. Microsoft on kuitenkin ilmoittanut 30.9.2022, että LUIS palvelun tuki tullaan lopettamaan 2025 vuoteen mennessä. Microsoft suosittelee, että nykyisille LUIS palvelua hyödyntäville kokonaisuuksille tehdään migraatio niin ikään Microsoftin tarjoamaan Azure Cognitive Service for Language -palveluun.[23] [24]

Azure Cognitive Service for Language kokoaa yhteen aiemmin erillisinä tarjotut Cognitive Services -palvelut, kuten Text Analytics, QnA Maker ja LUIS. Palvelu sisältää useita valmiiksi määritettyjä ominaisuuksia, joita voi käyttää suoraan sovelluksissa ilman tarvetta kouluttaa omia tekoälymalleja. Tämä mahdollistaa nopean ja helpon integroinnin sovelluksiin. Toisaalta, palvelu tarjoaa myös mahdollisuuden mukauttaa tekoälymalleja oman datan perusteella, mikä antaa lisää joustavuutta ja tarkkuutta sovellusten vaatimuksiin. Palvelun etuja LUIS:n tavoin ovat sen helppokäyttöisyys sekä integraatiomahdollisuudet Azure-

[22] Amazon Web Services. Amazon Comprehend. 2023.
[23] Microsoft. Migrate to conversational language understanding before Language Understanding (LUIS) is retired on 1 October 2025. 2022.
[24] Microsoft Azure. Language Understanding (LUIS). 2023.

ekosysteemiin. Palvelu tarjoaa laajan kielituen, josta löytyy tuki myös suomen kielelle. Negatiivisena puolena voidaan mainita, että palvelu on maksullinen. Tämä voi olla rajoite joillekin kehittäjille, etenkin, jos kyse on pienestä tai budjettirajoitteisesta projektista.[25]

4.1.2 Johtopäätökset

Markkinoilla on paljon erilaisia NLP-teknologioita tarjoavia palveluita, mutta opinnäytetyön käyttötapauksessa vertailukohteet piti rajata muutaman tunnetuimman palveluntarjoajan joukkoon (**Kuvio 4.**). Wit.ai valittiin muiden vaihtoehtojen joukosta useiden etujen perusteella. Näitä etuja olivat helppokäyttöisyys, avoin lähdekoodi, jatkuva kehittyminen, kattava dokumentaatio sekä kattavin kielituki. Kattava kielituki ja suomen kielen ymmärtäminen olivat opinnäytetyön käyttötapauksen vaatimuksissa tärkeässä asemassa. Vaatimukset johtuivat siitä, että käyttötapauksen palvelu, EcoReaction, sijoittuu erityisesti Suomen markkinoille.

[25] Microsoft Build. What is Azure Cognitive Service for Language? 2023. Artikkeli.

Alusta	Tukee suomen kieltä	Ilmainen kaupalliseen käyttöön	Avoin lähdekoodi
Wit.ai	KYLLÄ	KYLLÄ	KYLLÄ
Amazon Lex	EI	EI	EI
LUIS	EI	EI	EI
Dialogflow	KYLLÄ	EI	EI
Azure Cognitive Service for Language	KYLLÄ	EI	EI

Kuvio 4. NLU-teknologioiden vertailu.

Wit.ai:n käyttö on ilmaista, mutta sen dokumentaatiossa ei ole yksityiskohtaisia ohjeita tai rajoituksia koskien suurta määrää prosessipyyntöjä. Tämä aiheuttaa kysymyksen siitä, kuinka hyvin palvelu suoriutuu suuressa kuormituksessa. Tämä voi tuoda epävarmuutta, sillä jotkin kilpailijat mainitsevat selvästi dokumentaatiossaan prosessipyyntöjen määrän rajoitukset tai parhaat käytännöt. Prosessipyyntöjen määrän rajoitukset on suositeltavaa ottaa huomioon, kun arvioidaan palvelun skaalautuvuutta integroitaessa sitä omaan käyttökohteeseen, erityisesti projekteissa, joissa on korkea liikennemäärä tai suuri käyttäjäkunta.

On tärkeää huomata, että skaalautuvuus ei välttämättä ole ongelma kaikille Wit.ai:ta käyttäville palveluille. Palvelu voi olla toimiva monissa sovelluksissa, erityisesti niissä, jotka käsittelevät suuria määriä dataa, mutta myös sovelluksissa,

joissa datan prosessointitarve ei ole suuri. Tällainen käyttökohde on esimerkiksi opinnäytetyössä käytetty käyttökohde.

NLP-teknologiaa valittaessa huomioidaan integroitavan kohteen palvelinympäristö. Opinnäytetyön käyttötapauksessa palvelinympäristö voi vaihdella. Azure- tai AWS-ekosysteemissä toimivan kohteen integraatiossa voidaan kuitenkin huomioida Cognitive Service for Language- tai Amazon Lex -teknologioiden hyödyt. Näiden teknologioiden valmis tuki Azure- ja AWS-ekosysteemeille parantaa integraation sujuvuutta ja tehokkuutta.

4.2 Toiminnallisuus

Wit.ai tarjoaa kattavan dokumentaation, jonka avulla kehittäjä pääsee tutustumaan Wit.ai:n tarjoamiin toiminnallisuuksiin. Tässä kappaleessa käydään läpi kolme keskeisessä roolissa olevaa käsitettä, joiden avulla käyttäjä pääsee kehittämään sovelluksen toiminnallisuutta. Nämä käsitteet ovat ”intent”, ”entity” sekä ”trait”. Näiden käsitteiden hyödyntämistä syötteen tulkitsemisessa opinnäytetyön käyttötapauksessa havainnollisestaan liitteessä 1.

Intentio (engl. Intent) on Wit.ai:n yksi tärkeimmistä toiminnallisuuksista tai käsitteistä. Sitä käytetään ilmaisemaan käyttäjän aikomusta tai tarkoitusta syötteestä. Dokumentaatiossa mainitaan esimerkki, jossa syöte on ”Paljonko on toimiston lämpötila?”, jonka kohdalla tunnistetaan, että aikomus on saada toimiston lämpötila, eikä esimerkiksi olla tilaamassa pizzaa. Tälle syötteelle luodaan oma aikomus nimellä ”temperature_get”.[19]

Entiteetti (engl. Entity) on Wit.ai:n tarjoama toiminnallisuus tai käsite, mitä käytetään merkityksellisen tiedon eristämiseen syötteestä. Dokumentaatiossa mainitaan esimerkki, jossa entiteettiä käytetään eristämään lämpötilaan liittyvä tieto syötteestä. Esimerkiksi syötteessä ”Nosta lämpötila 30 asteeseen” voisi entiteetti olla ”30 asteeseen”.[19]

Piirre tai ominaisuus (engl. Trait) on toiminnallisuus tai käsite, mikä antaa lisätietoja entiteetille tai luokittelee tiettyjä intentioita. Ominaisuus mahdollistaa entiteettien tarkemman määrittelyn ja antaa lisää tietoa syötteen ymmärtämiseen. Täten ne auttavat erottamaan tiettyjä intentioita yksityiskohtaisimmilla piirteillä syötteestä. Dokumentaatiossa on mainittu esimerkki, jossa ominaisuutta ”wit/greetings” käytetään tunnistamaan syötteen tervehdys sen eri muodoissa ilman tarkempaa määrittelyä, kuten ”hello” tai ”hi”.[19]

4.3 Teknologian kouluttaminen

Wit.ai:n kouluttaminen on mahdollista suorittaa joko web-käyttöliittymän tai API:n kautta. API mahdollistaa datan syöttämisen suoraan Wit.ai:lle ilman manuaalista syöttöä käyttöliittymän kautta, mikä voi olla hyödyllistä tietyissä tilanteissa, erityisesti, kun koulutus halutaan suorittaa suuren datamäärän kanssa. Esimerkiksi, jos dataa on saatavilla jo valmiiksi lokitiedostoista tai taulukoista (**Taulukko 1.**), voidaan API:a hyödyntäen automatisoida koulutus erillisen ohjelman tai skriptin avulla. Tämä säästää aikaa ja vaivaa suurien datamäärien kanssa.

API myös mahdollistaa Wit.ai:n integroimisen osaksi laajempaa julkaisuputkea (engl. pipeline) tai koulutusprosessin automatisoimisen osaksi jatkuvaa integrointia ja julkaisua (engl. Continuous Integration and Continuous Delivery, CI/CD). Näin API tarjoaa tehokkaan ja skaalautuvan tavan kouluttaa Wit.ai:ta erilaisissa käyttötapauksissa.[26] Tässä opinnäytetyössä keskitytään kouluttamiseen web-käyttöliittymän kautta, sillä se mahdollistaa kouluttamisen nopeasti ja interaktiivisesti. Käyttöliittymän käyttö koulutuksessa on hyödyllistä esimerkiksi opinnäytetyön käyttötapauksessa, kun halutaan nopeasti iteroida ja kokeilla erilaisia koulutusdatan ja parametrien kombinaatioita.

[26] GitHub. wit-ai/wit-api-only-tutorial. 2023.

Taulukko 1. Esimerkki Wit.ai:n koulutukseen soveltuvasta datasta.[26]

I want to make a new appointment	appt_make
Show my appointments	appt_show

Teknologian koulutus käyttöliittymän kautta tapahtuu antamalla käyttöliittymälle syötteitä, joista se pyrkii automaattisesti analysoimaan ja havaitsemaan eri intentioita, entiteettejä sekä piirteitä. Wit.ai tarjoaa muutaman valmiin intention, entiteetin sekä piirteen, mutta omien vaihtoehtojen luominen on helppoa. Kun syöte on annettu, tulee syötteestä valita haluamansa sana tai kokonaisuus, jonka jälkeen tästä voi luoda oman intention, entiteetin tai piirteen. Näiden luomisen jälkeen, on mahdollista kokeilla toista vastaavanlaista syötettä, josta Wit.ai:n tulisi automaattisesti tunnistaa määritellyt intentiot, entiteetit sekä piirteet. Mikäli näin ei tapahdu, voi syötteestä valita tarkoituksen mukaiset sanat tai kokonaisuudet, ja liittää ne haluamaansa käsitteeseen. Toistojen jälkeen Wit.ai:n kyky tunnistaa parametrejä syötteestä paranee eksponentiaalisesti, mikä tekee koulutuksesta erittäin tehokasta.

Wit.ai:n laaja kielituki tuo koulutukseen paljon etuja, ja laaja kielituki tekee koulutuksesta varsin tehokasta. Koulutus voidaan toteuttaa vain yhdellä kielellä, mutta analysoidessa syötteitä, ne voidaan ymmärtää monella kielellä. Tämä tuo valtavan edun integroitaessa NLP-teknologiaa osaksi monikielistä sovellusta. Opinnäytetyön käyttötapauksessa tätä testattiin niin, että koulutus tapahtui aina suomen kielellä, mutta Wit.ai ymmärsi syötettä sekä ruotsin että englannin

kielellä. Wit.ai hyödyntää myös yhteisön opetusdataa parantaakseen tunnistukseen vaikuttavia algoritmejään. [27]

Kaiken kaikkiaan voidaan todeta, että Wit.ai:n koulutus on suoraviivaista ja helppoa, ja se pystyy tunnistamaan paljon erilaisia variaatioita jo muutamilla kymmenillä opetetuilla syötteillä. Tämä tekee Wit.ai:sta tehokkaan ja skaalautuvan ratkaisun monenlaisiin NLP-pohjaisiin sovelluksiin ja palveluihin.

[27] Qaffas, A., A. Improvement of Chatbots Semantics Using Wit.ai and Word Sequence Kernel: Education Chatbot as a Case Study. 2019.

5 LUONNOLLISEN KIELEN YMMÄRTÄMINEN OSANA SOVELLUSTA

Luvussa käydään läpi opinnäytetyön yhteydessä toteutetun projektin tavoitteita, arkkitehtuuria, testausta sekä muita toteutukseen liittyviä oivalluksia, huomioita ja johtopäätöksiä. Toteutetun mikropalvelun lähdekoodi on salassa pidettävää aineistoa, joten niihin viitataan tulevissa kappaleissa pääosin liitteinä.

5.1 Tavoite

Opinnäytetyön yhteydessä toteutetussa projektissa oli tavoitteena suunnitella ja toteuttaa mikropalvelu, joka mahdollistaa käyttäjän syötteen ymmärtämisen sekä selkokielisen vastauksen tuottamisen syötteen perusteella. Mikropalvelu toteutettiin osaksi Wapicen EcoReaction -palvelua, jossa se tarjoaa käyttäjille kokonaan uuden luonnollisen kielen rajapinnan, palvelun tarjoaman datan tarkasteluun.

Mikropalveluarkkitehtuuri on suosittu lähestymistapa sovellusten kehittämisessä osana tehokasta ja ketterää kehitystä, koska se tarjoaa joustavuutta ja skaalautuvuutta. Mikropalveluarkkitehtuurin avulla sovellus voidaan hajauttaa pienempiin, itsenäisiin mikropalveluihin, jotka voivat olla eri tiimien vastuulla. Hajauttaminen mahdollistaa mikropalveluiden itsenäisen kehityksen ja julkaisun, parantaa skaalautuvuutta sekä antaa mahdollisuuden valita parhaiten sopiva teknologia ja työkalut kuhunkin mikropalveluun.[28]

Projektin tavoitteena oli tutkia NLP-teknologian tarjoamaa potentiaalia sekä muodostaa parempi käsitys siitä, miten NLP-teknologiaa voidaan hyödyntää osana isompaa sovelluskokonaisuutta. Samalla pyrittiin kehittämään EcoReactionia eteenpäin sen liiketoimintatavoitteissa (Liite 2).

[28] Posta, C. 2016. Microservices for Java Developers.

5.2 Ohjelmointikäytännöt

Projektissa hyödynnetyt ohjelmointikäytännöt juontavat juurensa Robert C. Martinin kirjaan Clean Code, jossa muun muassa koodin selkeys ja ylläpidettävyys ovat tärkeässä roolissa. Clean Code -kirja on toiminut vahvana inspiraationa palvelun arkkitehtuurin ja nimeämiskäytäntöjen kehittämisessä. Projektin aikana on pyritty vakiintuneiden ja hyväksi havaittujen käytäntöjen noudattamiseen, joista yhtenä esimerkkinä Guard Clause -tekniikka (**Kuvio 5. ja 6.**), joka parantaa koodin luettavuutta, ylläpidettävyyttä ja voi myös vaikuttaa positiivisesti suorituskykyyn, poistamalla tarpeettomat sisäkkäiset ehdot ja haarat.[29]

Kuvio 5. Esimerkki, jossa hyödynnetään Guard Clause -käytäntöä.

[29] Gélinas, M. 2021. Guard Clauses Explained: *What is guard clause?*

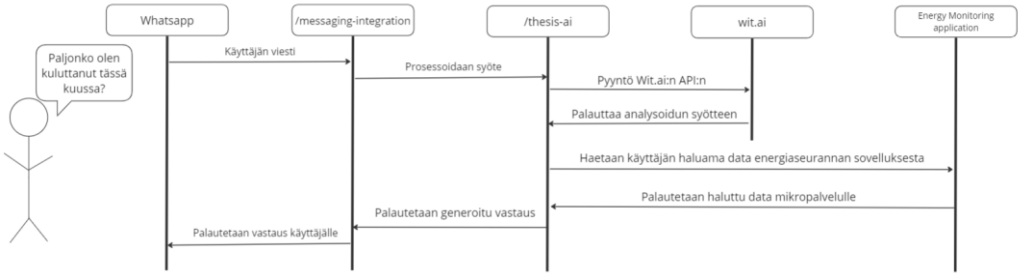
Kuvio 6. Esimerkki, jossa ei hyödynnetä Guard Clause -käytäntöä.

5.3 Arkkitehtuuri ja tietoturva

Mikropalvelun ohjelmointikieleksi valittiin Java, jonka kanssa hyödynnetään Spring-kehystä. Java valittiin ohjelmointikieleksi, koska muut sovelluskokonaisuuden mikropalvelut on myös toteutettu samalla ohjelmointikielellä. Tavoitteena oli toteuttaa kehitettävä mikropalvelu samalla arkkitehtuurilla ja ohjelmointikäytännöillä, joita on hyödynnetty myös EcoReactionissa. Näin kokonaisuus on yhtenäinen ja yhtenäisesti ylläpidettävä. Kommunikaatio sovelluksen muiden mikropalveluiden (**Kuvio 7.**) kanssa tapahtuu RESTful-arkkitehtuurityylin tuomien rajapintojen avulla, kuten liitteissä 3 ja 4 käy ilmi. Toteutetut rajapinnat perustuvat HTTP-protokollaan, jonka vuoksi toteutettu mikropalvelu on helppo yhdistää toimimaan muiden mikropalveluiden kanssa.

Rajapinta, jonka kautta mikropalvelu kommunikoi muiden mikropalveluiden kanssa, ei ole julkinen. Tämän vuoksi syötteen tarkistukset, validoinnit ja autentikointi tapahtuvat ennen rajapintaan saapumista. Näitä varten sovelluskokonaisuudessa on olemassa oma mikropalvelu, joka hoitaa ja vastaa syötteen tarkistuksista, autentikoinnista sekä validoinnista.

Kuviossa 7 esitellään, että mikropalvelu ei ole sidoksissa integroitavaan kohteeseen, vaan kuviossa 7 oleva "Energy Monitoring application" voidaan helposti vaihtaa haluttuun palveluun. Tämä toteutustapa tekee mikropalveluarkkitehtuurista hyvin monipuolisen, koska samaa mikropalvelua voidaan hyödyntää muissakin käyttökohteissa.

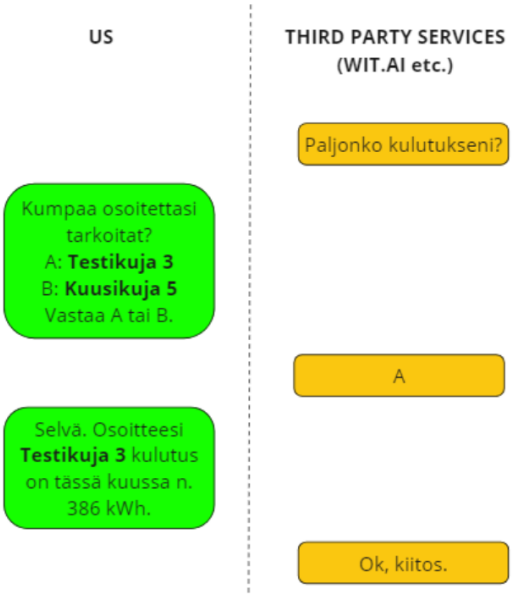


Kuvio 7. Pelkistetty kuva arkkitehtuurista, josta ilmenee sovelluskokonaisuuden eri mikropalveluita.

Arkkitehtuuria suunniteltaessa on tärkeää huomata, että palvelussa käsitellään myös yksityisyyden suojaan kuuluvaa materiaalia. Esimerkiksi sen sijaan, että käyttäjä syöttäisi oman osoitteensa suoraan viestikenttään, palvelun tulee pyytää käyttäjää valitsemaan oikean osoitteensa valmiista vastausvaihtoehdoista. Tämä toteutustapa antaa mahdollisuuden käsitellä yksityisyyden suojaan kuuluvaa materiaalia siten, ettei se kulje NLP-teknologiaa tarjoavien palveluiden kautta, ja täten henkilökohtaisia tietoja ei päädy kolmansien osapuolien käytettäväksi.

Kuviossa 8 nähdään, että keltaisella pohjalla olevat tekstit kulkevat NLP-teknologian tarjoavan palvelun kautta, kun taas vihreällä pohjalla olevat tekstit tulevat suoraan omasta palvelusta. Tämä on yksinkertainen, mutta helppo tapa minimoida mahdollista yksityisyyden suojaan kuuluvan tiedon luovuttamista kolmansille osapuolille.

Mikropalvelu on toteutettu niin, ettei käyttäjän yksilöivää tunnistetietoa siirry NLP-teknologiaa tarjoavalle palvelulle, vaan pyynnön yksilöintiin käytetään jokaisen pyynnön kohdalla erikseen generoitua tunnistetta. Tällöin NLP-teknologia käsittelee kaikki sille tulleet pyynnöt anonymisoituna datana, eikä ole kykenevä rakentamaan yksilöllisiä käyttäjäprofiileja.



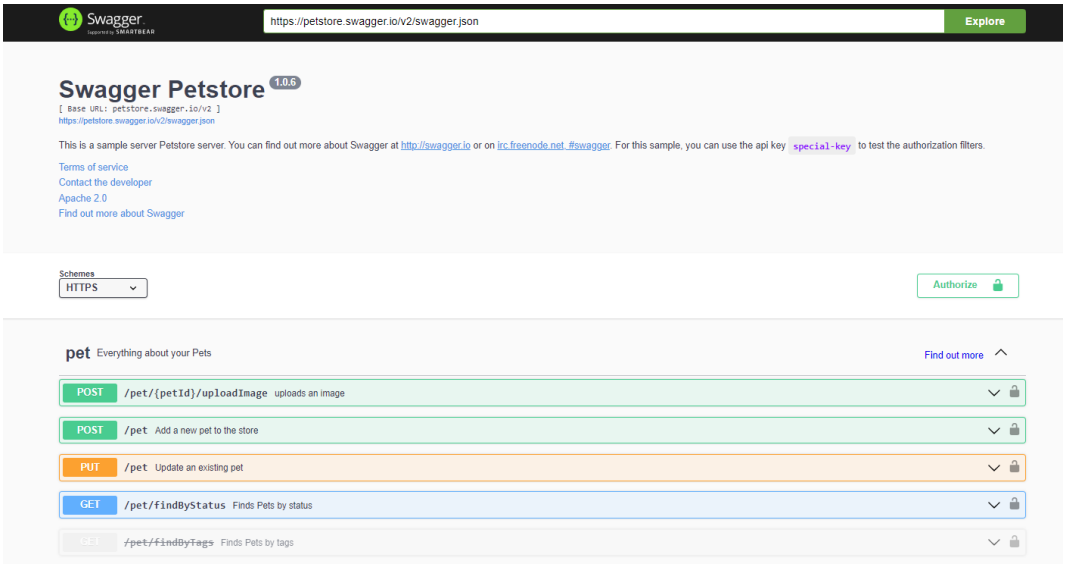
Kuvio 8. Palvelu ohjaa käyttäjää valitsemaan halutun osoitteen vastausvaihtoehdoista.

5.4 Rajapintakuvaus

Mikropalvelun tuottama rajapinta on yksinkertainen käyttää, ja se tuottaa annetun syötteen ja käyttäjän yksilöllisen tunnuksen perusteella vastauksen syötteen perusteella haluttuun dataan (Liitteet 5 ja 6). Toteutetun mikropalvelun rajapinnasta on tarjolla dokumentaatio, joka on luotu käyttäen avoimeen lähdekoodiin perustuvaa Swagger-työkalua (Liite 8).

Swagger on työkalu, joka antaa mahdollisuuden luoda visuaalista dokumentaatiota rajapinnoista. Rajapintojen ymmärtämistä ja testausta helpottaakseen, Swagger tarjoaa helppokäyttöisen ja interaktiivisen käyttöliittymän, jonka avulla kehittäjien on helppo kokeilla ja testata rajapintoja suoraan selaimessa (**Kuvio 9.**).[30]

[30] Ponelat, J., S. & Rosenstock L., L. 2022. Manning. Designing APIs with Swagger and OpenAPI.



Kuvio 9. Esimerkki Swagger-työkalun tarjoamasta rajapintadokumentaatiosta.

5.5 Tietokantakuvaus

Mikropalvelun tietokantana on MariaDB 10.10.2. MariaDB on relaatiotietokanta. Relaatiomalli organisoi ja hallitsee tietoja taulukkoina, joissa on rivejä ja sarakkeita. MariaDB on avoimen lähdekoodin tietokantajärjestelmä, joka on haarautunut MySQL:stä vuonna 2009. MariaDB:n kehitys on jatkunut itsenäisesti siitä lähtien.[31]

Mikropalvelun tarvitsemat taulut ja niiden luomiseen tarkoitetut SQL-komennot on kuvattu liitteessä 9. Mikropalvelun käyttämistä tauluista on saatavilla tietokantakuvaus (Liite 10), josta ilmenee taulujen sisältämät sarakkeet. Taulut sisältävät muun muassa käyttäjän aiemmin palveluun syöttämän viestin, sekä NLP-teknologian viestistä tuottaman analyysin. Käyttäjän aiempia viestejä voidaan hyödyntää esimerkiksi NLP-teknologian kouluttamiseen ja virhetilanteiden

[31] MariaDB Foundation. History of MariaDB in brief. 2023.

analysointiin. Taulujen tarjoama data voi myös auttaa yritystä kehittämään palvelujaan vastaamaan asiakkaiden odotuksia ja tarpeita paremmin.

5.6 Palvelun ja käyttäjän välinen vuorovaikutus

Tietokanta mahdollistaa käyttäjän aiempien viestien käsittelyn ohjelmallisesti, mikä parantaa vuorovaikutusta käyttäjän ja palvelun välillä. Aiempien viestien käsittely näkyy erityisesti tilanteissa, joissa käyttäjälle on tuotettu eri vastausvaihtoehtoja. Näissä tilanteissa käyttäjä voi valita yhden vaihtoehdon, esimerkiksi vastaamalla syötteellä "a" (**Kuvio 8.**). Tällöin mikropalvelu käsittelee käyttäjän syötettä ”a”, hyödyntämällä käyttäjän aiempia viestejä syötteen käsittelyyn. Tämä auttaa luomaan sujuvamman ja jatkuvamman vuorovaikutuksen.

Aiempien viestien käsittely mahdollistaa myös sen, että palvelu pystyy seuraamaan käyttäjän viestien kontekstia ja ymmärtämään uudet viestit oikeassa kontekstissa. Oikean kontekstin ymmärtäminen mahdollistaa sen, että palvelu tarjoaa relevantteja vastauksia tai toiminnallisuuksia käyttäjän tarpeiden mukaisesti, esimerkiksi aiemmin mainittuja vastausvaihtoehtoja valittaessa (**Kuvio 8.**).

5.7 Sovellustason toiminnallisuus ymmärretyn syötteen pohjalta

Sovellustason toiminnallisuus perustuu ymmärrettyyn syötteeseen ja sen pohjalta tehtyihin valintoihin, jotka mahdollistavat oikean vastauksen tuottamisen käyttäjälle. Kun mikropalvelu vastaanottaa syötteen (Liite 11), se tarkistaa pyynnön lähettäjän tunnistetiedot ja syötteen sisällön. Mikäli syöte vastaa määritettyjä rajaehtoja, välittää palvelu syötteen Wit.ai:lle analysoitavaksi (Liite 12).

Wit.ai palauttaa analysoidun syötteen mikropalvelulle JSON-muodossa (Liite 7). Yksinkertaistettu esimerkki Wit.ai:n tuottamasta vastauksesta on esitetty kuviossa 10. JSON-muotoinen vastaus muutetaan WitAiResponse-luokan olioksi

(engl. object) (Liite 14) käyttäen ObjectMapper-kirjastoa. Mikäli JSON-muotoista vastausta ei voida muuntaa WitAiResponse-olioksi, generoidaan ja tulostetaan virheilmoitus (JsonProcessingException).

JSON-muotoisen vastauksen muuttaminen olioksi tapahtuu prosessin avulla, jossa JSON-muotoinen data puretaan ohjelmalliseksi ymmärretyksi olioksi. Tätä kutsutaan deserialisoinniksi. Deserialisoinnin jälkeen oliota voidaan käsitellä ja hyödyntää ohjelmassa tarvittavalla tavalla, kuten datan tallentamiseen, käsittelyyn tai näyttämiseen käyttäjälle.

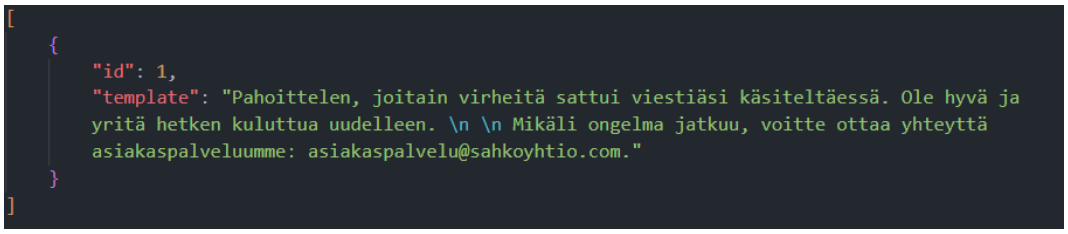
```
{
  "entities": {
    "fuel:fuel": [{
      "body": "bensan",
      "confidence": 0.9995,
      "end": 19,
      "entities": {},
      "id": "3441495742780019",
      "name": "fuel",
      "role": "fuel",
      "start": 13,
      "type": "value",
      "value": "bensan"
    }],
    "price:price": [{
      "body": "hinta",
      "confidence": 0.9995,
      "end": 40,
      "entities": {},
      "id": "905756017452139",
      "name": "price",
      "role": "price",
      "start": 35,
      "type": "value",
      "value": "hinta"
    }]
  },
  "intents": [{
    "confidence": 0.9964791957731618,
    "id": "929433634952910",
    "name": "price_question"
  }],
  "text": "Moi, mitä on bensan keskimääräinen hinta Suomessa?",
  "traits": {
    "GET_FUEL_PRICE": [{
      "confidence": 0.9964447508713363,
      "id": "535219105476270",
      "value": "PETROL"
    }]
  }
}
```


Kuvio 10. Esimerkki Wit.ai:n tuottamasta vastauksesta.

Deserialisoinnin jälkeen palvelu käsittelee käyttäjän aiempia viestejä (Liitteet 17 ja 18). Palvelu tarkistaa, onko käyttäjälle vastattu viimeisen tunnin aikana sellaisella

viestillä, johon palvelu odottaa vastausta. Jos tällainen viesti löytyy, palvelu osaa ottaa sen huomioon käyttäjän uutta syötettä käsiteltäessä (Liite 15). Kun tiedetään, odotetaanko aikaisempiin viesteihin vastausta, siirrytään seuraavaan vaiheeseen, joka on toteutettu omassa palveluluokassa. Tässä palveluluokassa on kaksi tärkeää metodia:

Ensimmäinen metodi (Liite 19) on julkinen ja sen tehtävänä on valita viestin ja sen sisältämän piirteen perusteella, mitä toimenpiteitä tulee tehdä. Metodi tarkistaa, vastaako viestin piirre sellaista toimenpideluokkaa (liitteissä nimellä ”actionHandler” esiintyvät luokat), joka on määritelty konfiguraatiossa. Jos näin ei ole, käyttäjän syöte on ymmärretty, mutta piirrettä vastaavaa toimenpideluokkaa ei ole määritelty (Liite 21). Tämän vuoksi metodi vaihtaa käsiteltävän viestin piirteeksi jatkokäsittelyä varten ”ERROR” -tyypin, jolloin käyttäjälle osataan palauttaa vastaus, jossa ilmenee, että viestiä käsiteltäessä sattui virhe (**Kuvio 11.**).



```
[
    {
        "id": 1,
        "template": "Pahoittelen, joitain virheitä sattui viestiäsi käsiteltäessä. Ole hyvä ja
        yritä hetken kuluttua uudelleen. \n \n Mikäli ongelma jatkuu, voitte ottaa yhteyttä
        asiakaspalveluumme: asiakaspalvelu@sahkoyhtio.com."
    }
]
```


Kuvio 11. Käyttäjälle palautuva virheviesti, kun virhe tapahtuu viestin käsittelyn aikana.

Toinen metodi (Liite 20) on yksityinen metodi, joka toimii ensimmäisen metodin tukena. Mikäli nykyinen viesti sisältää vain lisätietoja ja käyttäjän edelliseen vastaukseen odotetaan lisätietoja, palauttaa metodi edellisen viestin piirteen. Jos ehdot eivät täyty, metodi palauttaa nykyisen viestin piirteen.

Toimenpideluokat vastaavat selkokielisen vastauksen tuottamisesta käyttäjälle, eli NLG:stä. Toimenpideluokat periytyvät abstraktista luokasta, jossa toimenpideluokkia yhdistävät metodit ovat (Liite 22). Liitteessä 23 esitellään yksi toimenpideluokka, joka toteuttaa vastauksen tuottamisen energiaseurannan sovelluksesta saatavaa käyttäjäkohtaista dataa hyödyntäen. Toimenpideluokka

siis tuottaa käyttäjälle vastauksen, ja palauttaa sen viestinnästä vastaavalle mikropalvelulle, joka taas välittää viestin käyttäjälle. Lopuksi mikropalvelulla saapunut syöte, käyttäjätiedot, Wit.ai:n toteuttama analysointi sekä tuotettu vastaus tallennetaan tietokantaan.

Uusia piirteitä määritellessä, piirteen tyyppi pitää esitellä niille tarkoitetussa enum-luokassa (Liite 24). Enum-luokka on erityinen luokkatyyppi, joka sisältää ennalta määritellyt vakioarvot. [32] Tyypin määrittelyn jälkeen oikea tyyppi pitää määritellä vastaamaan haluttua toimenpideluokkaa. Määrittely tapahtuu mikropalvelun konfiguraatiotiedostossa (Liite 21). Toteutustapa mahdollistaa, ettei varsinaiseen koodipohjaan tarvitse tehdä muutoksia uusia toimenpideluokkia lisätessä.

5.8 Testaus

Testaus on tärkeä osa ohjelmistokehitysprosessia, erityisesti kun tuotetaan sovelluksia oikeaan tuotantoympäristöön. Testaaminen auttaa varmistamaan, että sovellus toimii odotetulla tavalla ja että se täyttää sille asetetut vaatimukset. Toteutetun mikropalvelun kohdalla testaus toteutettiin yksikkötestauksena. Yksikkötestit auttavat varmistamaan, että sovelluksen eri osat toimivat oikein eristettynä muista osista.

Yksikkötestaus on yksi tärkeä testausmuoto, joka keskittyy yksittäisten komponenttien, kuten metodien ja luokkien, testaamiseen erillisinä yksikköinä.

[32] Java.Lang.Enum Class. Tutorials Point. 2023.

Yksikkötestauksen avulla voidaan tarkastella sovelluksen toiminnallisuutta ja havaita mahdollisia virheitä tai puutteita jo varhaisessa vaiheessa.[33]

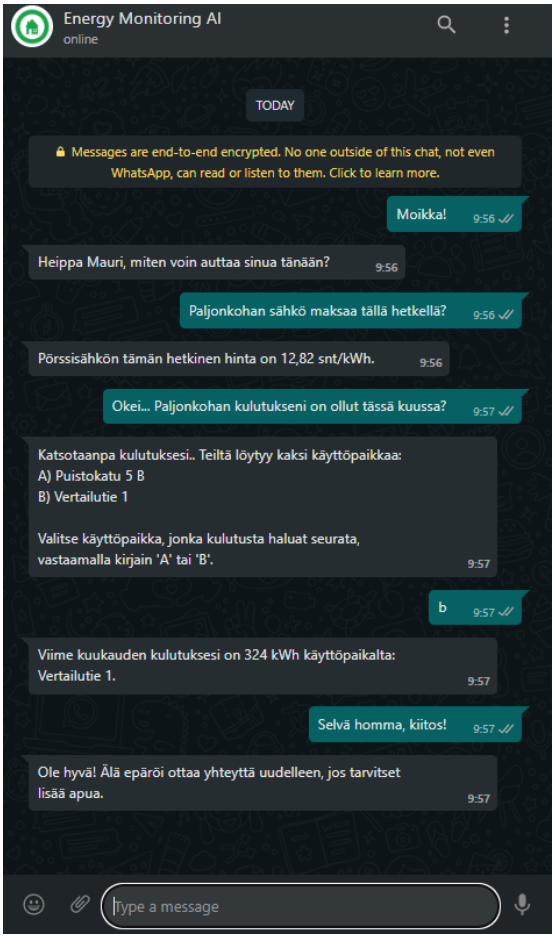
Mikropalvelun yksikkötestit on toteutettu käyttäen Junit- ja Mockito-kirjastoja. Näiden kirjastojen avulla on toteutettu kattava yksikkötestausstrategia. Strategia auttaa varmistamaan palvelun laadun ja vähentämään virheiden riskiä, varsinkin uusia ominaisuuksia lisätessä. Junit mahdollistaa testitapauksien luonnin ja odotetun tuloksen määrittelyn, kun taas Mockito mahdollistaa muiden komponenttien simuloinnin ja testien eristämisen.[34]

5.9 Johtopäätökset

Toteutetun mikropalvelun tavoitteena oli tuottaa EcoReactioniin luonnollisen kielen rajapinta, joka mahdollistaa käyttäjän kommunikoinnin palvelun kanssa käyttäen luonnollista kieltä. Tässä tavoitteessa onnistuttiin ja samalla saatiin paljon hyödyllistä osaamista ja kokemusta NLP-teknologian integroimiseksi osaksi olemassa olevaa sovellusta.

Kuvioissa 12 ja 13 nähdään, kuinka käyttäjä kommunikoi sovelluksen kanssa käyttäen luonnollista kirjoitettua kieltä. Vaikka opetusdata oli rajallista, oppi NLP-teknologia nopeasti ymmärtämään erilaisia variaatioita, ja mikropalvelun kanssa kommunikointi tuntui sujuvalta. Kuviossa 13 nähdään, kuinka tilanteessa, jossa NLP-teknologia ei ymmärrä käyttäjän syötettä, tuotetaan käyttäjälle silti tarkoituksenmukainen vastaus.

[33] Khorikov, V. 2020. Unit testing. *Principles, Practises and Patterns.* Manning.
[34] Grzejszczak, M. 2014. Mockito Programming Cookbook. *Hot Recipes for Mockito Development*.

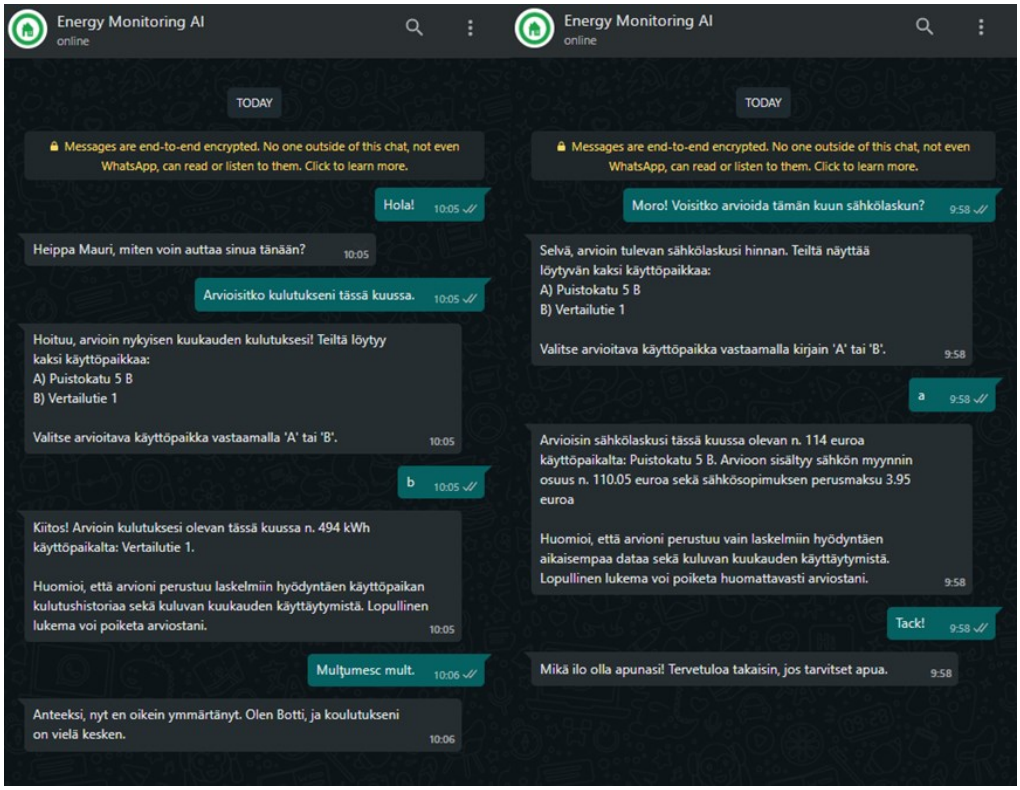


Kuvio 12. NLP osana energiaseurannan sovellusta, esimerkki 1.

Mikropalvelu käyttää tällä hetkellä vain osaa Wit.ai:n tarjoamasta analyysistä ja vastauksesta käyttäjän aikeen ymmärtämiseen ja vastauksen tuottamiseen. Tämä nousee esiin testitapauksissa, joissa käyttäjä ilmaisee yksityiskohtaisemman kysymyksen, kuten tarkemman aikaikkunan, jolta ajalta haluaisi kulutustaan seurata. Kuitenkin, mikropalvelu saa jo nyt tarvittavan datan syötteen analysoinnista yksityiskohtaisemman vastauksen generoimiseen, mutta tätä ei vielä opinnäytetyön käyttötapauksessa hyödynnetä, koska se ei vastannut opinnäytetyön keskeisimpiä tavoitteita.

Toteutettu mikropalvelu käyttää vastauksen tuottamiseen syötteen analysoinnista muodostuvaa piirre -ominaisuutta (engl. trait). Wit.ai:n tarjoama piirre -ominaisuus on yksityiskohtaisempi, kuin esimerkiksi intentio -ominaisuus (engl. intent). NLP-teknologia vaatii piirre -ominaisuuden analysoinnissa

enemmän koulutusdataa, kuin intentio -ominaisuuden analysoinnissa. Piirre -ominaisuuden käyttö merkitsee kuitenkin syötteen yksityiskohtaisempaa ymmärrystä. Esimerkkinä tästä voidaan mainita kuviossa 13 nähtävä syöte ”Arvioisitko kulutukseni tässä kuussa”. Analysoinnin tuottama intentio tästä syötteestä olisi ”arvioi kulutus”, kun taas analysoinnin tuottama piirre tästä syötteestä olisi ”arvioi kulutus *tässä kuussa*”. Tällä esimerkillä havainnollistetaan, että piirre -ominaisuus tarjoaa yksityiskohtaisemman analyysin, kun taas intentio -ominaisuus on kokonaisvaltaisempi.



Kuvio 13. NLP osana energiaseurannan sovellusta, esimerkki 2.

Kehitetyssä mikropalvelussa käytetty toteutustapa mahdollistaa kaikkien Wit.ai:n tarjoamien ominaisuuksien, kuten intention, entiteetin ja piirteen käytön. Toteutustapa mahdollistaa helpon integroimisen myös muihin sovelluskokonaisuuksiin, joissa voi olla erilaisia vaatimuksia NLP-teknologialle. Voidaan siis todeta, että Wit.ai:n tarjoamat ominaisuudet mahdollistavat kehittäjille monipuoliset mahdollisuudet räätälöidä NLP-teknologian tuottama

analyysi vastaamaan optimaalisesti omaa käyttötarkoitusta. Esimerkiksi käyttötapauksessa, jossa halutaan esittää käyttäjälle mahdollisesti hyvin yksityiskohtaista dataa, kuten tarkastella omaa kulutustaan vain tietyltä ajanjaksolta, on tärkeää hyödyntää myös syötteestä ymmärrettyjä entiteettejä, joista voidaan poimia juuri käyttäjän haluama ajanjakso yksityiskohtaisen vastauksen tuottamiseen. Toisaalta käyttötapauksessa, jossa näin yksityiskohtaiseen vastaukseen ei ole tarvetta, voidaan vastaus tuottaa pelkästään esimerkiksi intention tai piirteen perusteella.

6 YHTEENVETO JA POHDINTA

Opinnäytetyön tavoitteena oli tutkia NLP-teknologian tarjoamia mahdollisuuksia ja haasteita, sekä toteuttaa NLP-teknologiaa hyödyntävä mikropalvelu osaksi EcoReactionia. Vaikka NLP-teknologia tarjoaa kiinnostavia mahdollisuuksia esimerkiksi asiakaspalvelun automatisointiin, on tärkeää tunnistaa myös sen rajoitukset. Käytettävän NLP-teknologian luotettavuus ja tarkkuus voivat vaihdella suuresti, mikä voi johtaa virheellisiin tai epätarkkoihin vastauksiin asiakkaille. Lisäksi NLP-järjestelmät ovat herkkiä kielen monimutkaisuuksille, kuten kieliopillisille virheille, slangille tai kulttuurisille viitteille, mikä voi aiheuttaa haasteita varsinkin monikielisissä asiakaspalveluympäristöissä.

Tuloksista kuitenkin huomataan, että nykyiset NLP-teknologiaa tarjoavat alustat ovat hyvin kehittyneitä. Ne tarjoavat kattavan kielituen ja älykkäät kielimallit syötteiden analysointiin. NLP-teknologian hyödyntäminen avaa myös uudenlaisia mahdollisuuksia sovellusten ja käyttäjien väliseen vuorovaikutukseen. Siinä missä perinteiset graafiset käyttöliittymät ovat olleet vallitseva tapa käyttää sovelluksia, NLP-teknologia voi tarjota kokonaan uuden rajapinnan, NLI:n, joka mahdollistaa käyttäjän ja sovelluksen kommunikaation luonnollisemmalla ja intuitiivisemmalla tavalla.

Graafisella käyttöliittymällä, sekä NLI:llä, on omat vahvuutensa ja heikkoutensa, ja valinta niiden välillä riippuu sovelluksen tarpeista ja käyttötapauksesta. Graafisen käyttöliittymän etuja ovat sen visuaalinen selkeys ja nopeus sekä kyky tarjota visuaalista kontrollia käyttäjän syöttämiin tietoihin. Graafinen käyttöliittymä on myös hyvä valinta monimutkaisempien toimintojen hallintaan ja kompleksisen datan visualisointiin.

NLP-teknologia puolestaan tarjoaa käyttäjille mahdollisuuden käyttää sovelluksia puheen ja äänen kautta, mikä voi olla luonteva ja tehokas tapa kommunikoida sovellusten kanssa. NLP-teknologian tarjoamat hyödyt voivat olla erityisen tärkeitä käyttäjille, joilla on esimerkiksi motorisia rajoitteita. NLP-teknologia mahdollistaa

myös paremman henkilökohtaisen käyttökokemuksen ja sitoutumisen sovellukseen. NLP-teknologian tarjoamiin mahdollisuuksiin voi kuitenkin liittyä tietoturvaan ja yksityisyyteen liittyviä riskejä, jotka tulee ottaa huomioon sovelluskehityksessä.

NLP-teknologioita hyödyntäessä tulee miettiä, mitä siltä halutaan. NLP-teknologia tarjoaa kokonaan uuden rajapinnan käyttäjän ja sovelluksen välille, joka voi joissain tapauksissa jopa korvata perinteisen käyttöliittymän kokonaan, mutta sillä on silti haasteensa. Toisaalta, graafisen käyttöliittymän sekä NLI:n yhdistäminen voi tarjota entistä paremman käyttökokemuksen käyttäjille.

Yhtenä lisätutkimusten kohteena voisi olla NLP-teknologian vaikutus käyttäjäkokemukseen ja sen vaikutus käyttäjän sitouttamiseen. Käyttäjien mielipiteitä ja kokemuksia NLP-pohjaisista käyttöliittymistä voidaan tutkia, jotta pystytään selvittämään NLP:n hyötyjä ja haasteita verrattuna perinteisiin graafisiin käyttöliittymiin. Tulosten perusteella voitaisiin saada arvokasta tietoa käyttäjäkokemuksen parantamisesta ja käyttäjien sitoutumisen edistämisestä NLP-teknologiaa hyödyntävissä sovelluksissa.

LÄHTEET

Amazon Web Services. Amazon Comprehend. 2023. Viitattu 20.3.2023. https://aws.amazon.com/comprehend/.

Carlstedt, M. 2017. Mälarden University. Using NLP and context for improved search result in specialized search engines. Viitattu 13.3.2023. https://mdh.diva-portal.org/smash/get/diva2:1088795/FULLTEXT01.pdf.

Dale, R., Di Eugenio, B. & Scott, D. 1998. Introduction to Special Issue on Natural Language Generation. Viitattu 2.3.2023. https://aclanthology.org/J98-3001.pdf.

Gélinas, M. 2021. Guard Clauses Explained: *What is guard clause?* Viitattu 20.3.2023. https://maximegel.medium.com/what-are-guard-clauses-and-how-to-use-them-350c8f1b6fd2.

Grzejszczak, M. 2014. Mockito Programming Cookbook. *Hot Recipes for Mockito Development*. Viitattu 20.4.2023. https://www.javacodegeeks.com/wp-content/uploads/2016/09/Mockito-Programming-Cookbook.pdf.

GitHub. wit-ai/wit-api-only-tutorial. 2023. Viitattu 20.2.2023. https://github.com/wit-ai/wit-api-only-tutorial.

Halfsa, S. D., Ikramullah M. L., Khalid M. M. & Bukhari S. A. C. 2023. Frameworks for Querying Databases Using Natural Language: A Literature Review. Viitattu 20.4.2023. https://arxiv.org/ftp/arxiv/papers/1909/1909.01822.pdf.

Hamborg, F & Karsten, D. 2021. NewsMTSC: A Dataset for (Multi-)Target-de-pendent Sen-timent Classification in Political News Articles. *In Proceedings of the 16th Conference of the Euro-pean Chapter of the Association for Computational Linguistics: Main Volume*. Viitattu 22.2.2023. https://aclanthology.org/2021.eacl-main.142/.

Jain, K. 2021. NLP/Deep Leep Techniques in Healthcare for Decision Making. Viitattu 11.4.2023. https://www.iomcworld.org/open-access/nlpdeep-learning-techniques-in-healthcare-for-decision-making.pdf.

Java.Lang.Enum Class. 2023. Viitattu 22.4.2023. https://www.tutorialspoint.com/java/lang/pdf/java_lang_enum.pdf.

Jones, K., Altancu, E., Virginia, N. Franquiera, L., Wang, Y., & Shujun, L. 2022. University of Cyber Security for Society. A Comprehensive Survey of Natural Language Generation Advances from the Perspective of Digital Deception. Viitattu 10.4.2023. https://arxiv.org/pdf/2208.05757.pdf.

Jyväskylän yliopisto. 2019. Graph-based exploration and clustering analysis of semantic spaces. Viitattu 18.3.2023. https://jyx.jyu.fi/bitstream/handle/123456789/67708/1/veremyevym.pdf.

Karlsson, F. Yleinen kielitiede. 2012. Kustannusosakeyhtiö Gaudeamus. Viitattu 11.3.2023.

Khorikov, V. 2020. Unit testing. *Principles, Practises and Patterns.* Manning. Viitattu 20.2.2023. https://sd.blackball.lv/library/unit_testing_(2020).pdf.

Krupsky, S. 2021. Natural Language Processing Introduction: what is Natural Language Processing (NLP)? NLP Cloud. Blogi. Viitattu 20.1.2023. https://nlpcloud.io/introduction-what-is-nlp-natural-language-processing.html.

Mashaabi, M., Alotaibi, A., Qudaih, H., Alnashwan, R. & Al-Khalifa, H. 2021. Viitattu 21.3.2023. https://arxiv.org/ftp/arxiv/papers/2212/2212.09523.pdf.

MariaDB Foundation. History of MariaDB in brief. 2023. Viitattu 20.4.2023. https://mariadb.org/en/#history.

Microsoft. Migrate to conversational language understanding before Language Understanding (LUIS) is retired on 1 October 2025. 2022. Viitattu 18.2.2023. https://azure.microsoft.com/en-us/updates/language-understanding-retirement/.

Microsoft Azure. Language Understanding (LUIS). 2023. Viitattu 25.3.2023. https://www.luis.ai/.

Microsoft Build. What is Azure Cognitive Service for Language? 2023. Article. Viitattu 15.4.2023. https://learn.microsoft.com/en-us/azure/cognitive-services/language-service/overview.

Natural Language AI. Google Cloud. 2023. Viitattu 20.4.2023. https://cloud.google.com/natural-language.

Navigli, R. 2015. Department of Computer Science. Natural Language Undestanding: Instructions for (Present and Future). Viitattu 7.3.2023. https://www.ijcai.org/proceedings/2018/0812.pdf.

Ponelat, J., S. & Rosenstock L., L. 2022. Manning. Designing APIs with Swagger and OpenAPI. Viitattu 18.3.2023. https://dl.ebooksworld.ir/books/Designing.APIs.with.Swagger.and.OpenAPI.Ponelat.Rosenstock.Manning.9781617296284.EBooksWorld.ir.pdf.

Posta, C. 2016. Microservices for Java Developers. Viitattu 18.4.2023. https://pepa.holla.cz/wp-content/uploads/2016/10/microservices-for-java-developers.pdf.

Roinila, M. G. W. Leibnizin rationaalinen kielioppi. 2023. Viitattu 13.3.2023. https://www.mv.helsinki.fi/home/mroinila/Leibnizin%20rationaalinen%20kielioppi.pdf.

Qaffas, A., A. Improvement of Chatbots Semantics Using Wit.ai and Word Sequence Kernel: Education Chatbot as a Case Study. 2019. Viitattu 25.3.2023. https://www.mecs-press.org/ijmecs/ijmecs-v11-n3/IJMECS-V11-N3-3.pdf.

Tampereen yliopisto. Kielen osajärjestelmät. 2009. Viitattu 15.3.2023. https://web.archive.org/web/20141109090242/http://www.sis.uta.fi/infim/infim_2011/informaatiotutkimus/kurssisivut/a34/osajarjestelmat.html.

Tieteen termipankki. 2023. Kielitiede: Luonnollinen kieli. Viitattu 10.4.2023. https://tieteentermipankki.fi/wiki/Kielitiede:luonnollinen kieli.

Tieteen termipankki. 2023. Kielitiede: Pramagtiikka. Viitattu 10.4.2023. https://tieteentermipankki.fi/wiki/Kielitiede:pragmatiikka.

Tieteen termipankki. 2023. Kielitiede: Syntaksi. Viitattu 10.4.2023. https://tieteentermipankki.fi/wiki/Kielitiede:syntaksi.

University of London. 2013. Introduction to natural language processing. Viitattu 9.3.2023. https://www.london.ac.uk/sites/default/files/study-guides/introduction-to-natural-language-processing.pdf.

Wapice Oy. 2023. EcoReaction. Viitattu 25.4.2023. https://www.wapice.com/fi/tuotteet/ecoreaction.

Wit.ai. Getting Started With Wit.ai. 2023. Viitattu 21.3.2023. https://wit.ai/docs.

Wit.ai is joining Facebook. 2015. Viitattu 21.3.2023. https://medium.com/wit-ai/wit-ai-is-joining-facebook-deff3745fcf5.

LIITTEET

LIITE 1. Salassa pidettävä aineisto. Toiminnallisuuksien havainnollistaminen syötteen tulkitsemisessa.

LIITE 2. Salassa pidettävä aineisto. Esimerkki tavoitteista, joita pyritään toteuttamaan energiaseurannan sovellukseen.

LIITE 3. Salassa pidettävä aineisto. Projektin arkkitehtuuria havainnollistettu sekvenssikaaviolla.

LIITE 4. Salassa pidettävä aineisto. Projektin tiedostorakennetta kuvattuna.

LIITE 5. Salassa pidettävä aineisto. Esimerkki mikropalvelun tarjoamaan rajapintaan tehtävästä pyynnöstä.

LIITE 6. Salassa pidettävä aineisto. Esimerkki mikropalvelun vastauksesta pyynnön syötteen perusteella.

LIITE 7. Salassa pidettävä aineisto. Esimerkki NLP-teknologian tuottamasta vastauksesta syötteen perusteella.

LIITE 8. Salassa pidettävä aineisto. Kuvakaappaus Swagger-työkalun tuottamasta dokumentaatiosta koskien mikropalvelun rajapintaa.

LIITE 9. Salassa pidettävä aineisto. Tietokannan luontiskripti tietokannan rakenteen luomiseksi.

LIITE 10. Salassa pidettävä aineisto. Tietokantakuvaus.

LIITE 11. Salassa pidettävä aineisto. Ohjain (engl. controller) -luokan kuvaus.

LIITE 12. Salassa pidettävä aineisto. Palvelu (engl. service) -luokka käsittelee sille saapuneen pyynnön.

LIITE 13. Salassa pidettävä aineisto. Wit.ai:n vastauksen käsittely.

LIITE 14. Salassa pidettävä aineisto. WitAiResponse -luokka.

LIITE 15. Salassa pidettävä aineisto. Käyttäjän aiempien viestien käsittely.

LIITE 16. Salassa pidettävä aineisto. Palvelu -tason luokka käsittelee ymmärretyn syötteen, ja tekee toiminnallisuuksia ”trait” -tyypin perusteella.

LIITE 17. Salassa pidettävä aineisto. Aikaisempien viestien käsittelyyn tarkoitetun palvelun esittelyluokka.

LIITE 18. Salassa pidettävä aineisto. Aikaisempien viestien käsittelyyn tarkoitetun palvelun toteutusluokka.

LIITE 19. Salassa pidettävä aineisto. Metodi, joka tarkistaa onko viestin piirre määritelty vastaamaan toimenpideluokkaa.

LIITE 20. Salassa pidettävä aineisto. Aikaisempien viestien käsittelyyn tarkoitetun palvelun toteutusluokka.

LIITE 21. Salassa pidettävä aineisto. Toimenpideluokkien määrittely (liitteessä ”actionHandler”).

LIITE 22. Salassa pidettävä aineisto. Abstrakti luokka, josta toimenpideluokat periytyvät.

LIITE 23. Salassa pidettävä aineisto. Esimerkki toimenpideluokasta, joka hyödyntää energiaseurannan sovelluksen käyttäjäkohtaista dataa vastauksen tuottamiseen.

LIITE 24. Salassa pidettävä aineisto. Esimerkki luokasta, jossa määritellyt piirre tyypit esitellään.

LIITE 25. Salassa pidettävä aineisto. Testiluokka, jossa on toteutettu yksikkötestejä JUnit- ja Mockito-kirjastoja hyödyntäen.