Ohjelmistokehittäjän päiväkirja

Teemu Laine

Haaga-Helia ammattikorkeakoulu
Tietojenkäsittelyn tradenomin tutkinto
Amk-opinnäytetyö
2024

Tiivistelmä

Tekijä(t)
Teemu Laine

Tutkinto
Tradenomi

Raportin/Opinnäytetyön nimi
Ohjelmistokehittäjän päiväkirja

Sivu- ja liitesivumäärä

Työhön sisältyy ohjelmistokehittäjän päiväkirja, jonka aikana työn tekijä on kirjannut ja analysoinut työtehtäviään ja reflektoinut niitä viikoittain. Tavoitteeksi päiväkirjan pidon ajalle asetettiin tekninen kehittyminen ohjelmistokehityksen saralla sekä projektiosaamisen ja vuorovaikutustaitojen kehittäminen.

Päiväkirjan pidon aikana tekijä oppi paljon generatiivisen tekoälyn teknologioista ja mahdollisuuksista sekä ohjelmistotestaamisesta. Generatiivinen tekoäly asettaa ohjelmistotestaamiselle uudenlaisia haasteita, mutta perinteisemmät testausmenetelmät pysyvät myös arvossaan.

Projektiosaamisen osalta kehittymistä tapahtui erityisesti aika-arvioiden tekemisessä sekä etätyöskentelyn etujen sekä haasteiden tunnistamisessa. Etätyön edut tuo työhyvinvointi- ja tehoetuja monissa tehtävissä, mutta tietyissä tuotekehitysprojektien vaiheissa myös kasvotusten suoritettavat palaverit ja aivoriihet ovat perusteltuja.

Vuorovaikutustaidoista tekijä oppi erityisesti selkeän kommunikoinnin tärkeydestä sekä tekniikoista ohjelmistokehitystöiden tekemiseen yhteisvoimin. Esimerkiksi pariohjelmointi on tekniikka, joka voi edistää muun muassa työn tehostamista ja tiedon siirtämistä henkilöltä toiselle.

Päiväkirjan pitäminen edisti edellä mainittujen asioiden, eli teknisen osaamisen sekä projekti- ja vuorovaikutustaitojen kehittymisen lisäksi, tekijän kykyä jäsennellä työskentelyään sekä sisäistää oppimiaan asioita nopeammin.

Asiasanat
Tekoäly, chattibotti, digitaalinen avustaja, ohjelmistokehittäjä, ohjelmistotestaaminen

Sisällys

1 Johdanto 1
2 Nykytilanteen kuvaus 2
2.1 Työtehtävät 2
2.2 Työskentelymenetelmät 2
2.2.1 Scrum 2
2.2.2 Kanban 3
2.3 Teknologiat 3
2.3.1 Docker 3
2.3.2 Temporal 3
2.3.3 FastAPI 3
2.3.4 Vue.js 4
2.3.5 ElasticSearch 4
2.3.6 Suuret kielimallit 4
2.4 Sidosryhmät työpaikalla 4
2.5 Vuorovaikutustaidot työpaikalla 5
3 Päiväkirja 6
3.1 Viikko 1 6
3.1.1 Viikkoreflektointi 23.-27.10.2023: 8
3.2 Viikko 2 9
3.2.1 Viikkoreflektointi 30.10.2023 – 3.11.2023 10
3.3 Viikko 3 11
3.3.1 Viikkoreflektointi 6.-10.11.2023 12
3.4 Viikko 4 12
3.4.1 Viikkoreflektointi 13.11.2023–17.11.2023 14
3.5 Viikko 5 14
3.5.1 Viikkoreflektointi 20.11-24.11.2023 16
3.6 Viikko 6 17
3.6.1 Viikkoreflektointi 27.11.-1.12.2023 18
3.7 Viikko 7 19
3.7.1 Viikkoreflektointi 4.12.2023-8.12.2023 21
3.8 Viikko 8 21
3.8.1 Viikkoreflektointi 11.12.2023 – 15.12.2023 23
3.9 Viikko 9 23
3.9.1 Viikkoreflektointi 18.12.-22.12.2023 24
3.10 Viikko 10 25

3.10.1 Viikkoreflektointi 8.1.–12.8.2024 26
3.11 Viikko 11 27
3.11.1 Viikkoreflektointi 15.1.-17.1.2024 28
3.12 Viikko 12 28
3.12.1 Viikkoreflektointi 22.1.-26.1.2024 29
3.13 Viikko 13 29
3.13.1 Viikkoreflektointi 29.1. – 2.2.2024 31
4 Pohdinta 32
5 Viitteet 34

1 Johdanto

Tämä opinnäytetyö on päiväkirjamuotoinen opinnäytetyö, jonka merkinnöissä kuvaan päivittäisiä työtehtäviäni ohjelmistokehittäjänä. Kirjoitan jokaisena työpäivänä kuvauksen päivän aikana työstämistäni tehtävistä ja viikon lopuksi analyysin työviikosta. Päiväkirjaa pidän aikavälillä 23.10.2023 – 2.2.2024.

Työnantajani on yritys, joka tarjoaa erilaisille organisaatioille laajaa oppimisalustaa sekä tuottaa itse sisällön alustalle. Oppimissisältöä tarjotaan pääosin talon sisällä kehitetyn web-sovelluksen kautta. Alustalla on tarjolla laaja kurssivalikoima liittyen muun muassa aihealueisiin kuten digiosaaminen, ohjelmistokehitys, veroasiat, kirjanpito ja johtaminen. Työnantaja toimii neljässä eri maassa; Suomessa, Ruotsissa, Norjassa sekä Tanskassa. Yritys työllistää kokonaisuudessaan noin 150 ihmistä.

Ohjelmistokehitystiimi koostuu noin 15 ohjelmistoalan ammattilaisesta. Tiimiin kuuluu kehittäjiä, testaajia, tuoteomistajia sekä käyttöliittymäsuunnittelija.

Tavoitteena opinnäytetyölle on kehittää teknistä osaamistani liittyen webteknologioihin ja tekoälykielimalleihin sekä kehittää projektiosaamistani ja vuorovaikutustaitojani tiimityöskentelyssä.

2 Nykytilanteen kuvaus

2.1 Työtehtävät

Olen työskennellyt työnantajani palveluksessa hieman reilun vuoden ajan ja olen päässyt hyvin vauhtiin ohjelmistoalan ammattilaisena. Työnkuvaani on kuulunut muun muassa käyttöliittymäkehitystä, bugien korjaamista, taustajärjestelmien kehitystä sekä uusien palveluiden suunnittelua ja toteutusta. Päivittäiset työtehtäväni koostuvat tyypillisesti ohjelmointitehtävistä sekä kokouksista.

Työtehtäväni koostuvat sekä web-sovelluksen käyttöliittymän että alustan taustajärjestelmien kehittämisestä. Pääasiallisina ohjelmointikielinä toimivat Python, JavaScript/TypeScript ja PHP. Omat vahvuuteni ovat JavaScriptissä sekä Pythonissa.

Teknisen osaamisen lisäksi työssä on tärkeää hahmottaa isoja kokonaisuuksia sekä tunnistaa mahdollisia teknisiä pullonkauloja. Lisäksi omien ajatusten ja teknisten konseptien kommunikointi muille sidosryhmille on keskiössä.

Kaiken kaikkiaan olen tähän mennessä suoriutunut hyvin työtehtävistäni ja noussut nopeasti aloittelijatasolta taitavaksi suoriutujaksi. Minulle pystyy antamaan haastavampiakin teknisiä tehtäviä yksin työstettäväksi ilman kokeneemman kehittäjän valvontaa. Lisäksi minulle on annettu myös ohjelmistoarkkitehtuuriin liittyviä tehtäviä. Pystyn ottamaan haltuun uusia teknologioita tarvittaessa nopeasti ja nautin paljon uusien asioiden opettelusta.

2.2 Työskentelymenetelmät

Pääasiallinen käyttämämme kehitysmenetelmä on Scrum, mutta myös Kanban-menetelmä ja perinteisempiä projektimenetelmiä on käytössä.

2.2.1 Scrum

Scrum on tuotteiden ja palveluiden kehittämiseen tarkoitettu ketterä menetelmä. Scrumin ytimessä on mm. tehtävälistat, määrätyn mittaiset työskentelyiteraatiot, kokousrutiinit sekä tavoite julkaisukelpoiseen tuotteeseen iteraatioiden päättyessä (Rubin, K. 2014, alaluku “What is Scrum?”). Kirjassaan ”Essential Scrum: A Practical Guide to the Most Popular Agile Process” alaluvussa ”Why Scrum?” Kenneth S. Rubin kuvailee työskentely-ympäristöä, johon liittyy paljon epävarmuuksia ja tuntemattomia tekijöitä, soveltuvaksi Scrum-menetelmälle.

2.2.2 Kanban

Kanban on kehitysmenetelmä, jonka metodologiaa pidän yksinkertaisempana kuin Scrumin. Myös Kanbanissa käytetään tehtävälistaa. Tiimi määrittelee tehtävien suorittamiseen vaiheet, joiden läpi kaikki tehtävät käyvät. Ajatuksena on priorisoida tehtävälistan tehtäviä ja niitä poimitaan käymään läpi tämä ennalta määrätty prosessi. Menetelmän ytimessä on Kanban-taulu, jolla tehtävien edistymistä visualisoidaan. Eri vaiheet prosessissa voivat olla esimerkiksi ”Tehtävän analysointi”, ”Tehtävän toteutus”, ”Toteutuksen validointi/testaaminen” ja ”Toimitus loppukäyttäjälle”. (Brechner, E. 2015, luku 2)

2.3 Teknologiat

Päiväkirjan kirjoittamisen ajan työtehtäväni keskittynevät uuden tekoälypohjaisen palvelun kehittämiseen alustalle. Palvelu olisi tarkoitus julkaista Beta-versiona lähiaikoina käyttäjille. Työtehtävissäni tulen tarvitsemaan osaamista monista eri teknologioista, joita esittelen seuraavaksi.

2.3.1 Docker

Docker mahdollistaa sovellusten paketoimisen niin kutsuttuihin kontteihin. Kontit ovat kevyitä, helposti erilaisiin ympäristöihin siirreltäviä paketteja, jotka ovat suunniteltu helpottamaan sovellusten ajamista erilaisissa ympäristöissä. Kontit yrittävät taata sen, että riippumatta ajoympäristöstä, sovellus toimii samalla tavoin. (Docker Inc.)

2.3.2 Temporal

Temporal on avoimen lähdekoodin ohjelmisto, jonka avulla voi rakentaa virheenkestäviä sovelluksia. Sen ydinkäsitteisiin kuuluu ”workflowt”, jotka automaattisesti käsittelevät sovelluksen virhetiloja ja uudelleenyrityslogiikkaa. Temporalia hyödyntäviä sovelluksia voi kirjoittaa Python-, Go-, Java-, PHP, TypeScript- tai C#-ohjelmointikielillä. (Temporal Technologies Inc.)

2.3.3 FastAPI

FastAPI on avoimen lähdekoodin webkehityskirjasto, joka on suunniteltu helppokäyttöiseksi ja nopeaksi tavaksi kehittää suorituskykyisiä rajapintoja Python-kielellä. Se on kirjoitettu toimimaan saumattomasti Pydantic-datavalidaatiokirjaston kanssa. FastAPI:lla kirjoitetut sovellukset tuottavat myös automaattisesti OpenAPI-spesifikaation mukaista dokumentaatiota. (Lubanovic, B, 2023, alaluku ”What is FastAPI?”)

2.3.4 Vue.js

Vue on avoimen lähdekoodin JavaScript-kirjasto, jolla voi kirjoittaa reaktiivisia webkäyttöliittymiä. Vuella rakennetaan sovelluksia kirjoittamalla Vue-komponentteja, jotka eristävät komponentin logiikan, pohjan ja tyylit yhteen tiedostoon. (Vue.js -dokumentaatio)

2.3.5 ElasticSearch

ElasticSearch on hakukonemoottori, joka on rakennettu Apache Lucene -kirjaston päälle. ElasticSearch on suunniteltu suurten datamäärien nopeaan käsittelyyn. Sen avulla voi rakentaa tehokkaita reaaliaikaisia hakukoneita, jonka lisäksi sitä hyödynnetään muun muassa myös erilaisissa data-analytiikan tehtävissä. (Konda, M. 2023, luku 1)

2.3.6 Suuret kielimallit

Jay Alammarin ja Marten Grootendorstin ”Hands-On Large Language Models” -kirjan alaluvussa ”A Recent History of Language AI” kuvataan kieltä käsittelevien tekoälymallien jakautuvan kolmeen kategoriaan: generatiiviset mallit, tekstivektorointimallit sekä luokittelumallit.

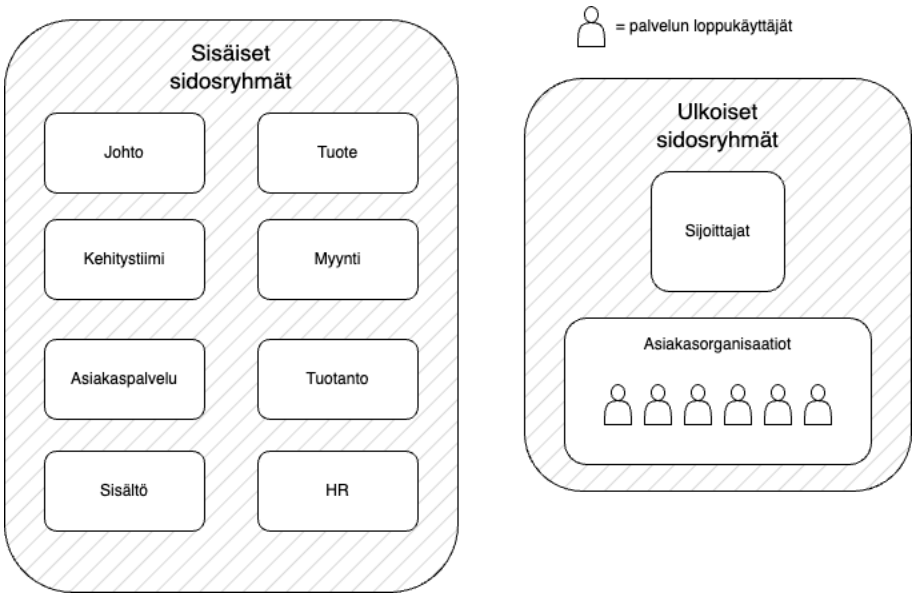
Tekstivektorointimalleilla pyritään tallentamaan tekstimuotoisen datan semantiikkaa vektorimuotoon. Ensimmäisiä tämänkaltaisia vektorointimalleja oli word2vec -malli, joka julkaistiin vuonna 2013. (Alammar, J. 2024, alaluku ”Better Represantions with Dense Vector Embeddings”)

Suurten kielimallien kehittämiselle keskeisiä käänteitä oli ”Attention Is All You Need” -tutkimuksen julkaiseminen, joka esitteli uudenlaisen muuntajaksi (käännös sanasta ”transformer”) kutsutun tekoälymalliarkkitehtuurin. Tämän tutkimuksen esittelemään arkkitehtuuriin nojaten alettiin kouluttamaan uudenlaisia generatiivisia tekoälymalleja, joita kutsutaan GPT-malleiksi (Generative Pre-trained Transformer). GPT-malleille annetaan syötteenä tekstiä ja malli vastaa todennäköisimmällä jatkolla syötteeseen. Muuntaja-arkkitehtuuri poiki myös uudenlaisia vektorirepresentaatiomalleja, kuten BERT-mallit (Bidirectional Encoder Representations from Transformers). Näitä uudelle muuntaja-arkkitehtuurille rakennettuja tekoälymalleja kutsutaan tyypillisesti suuriksi kielimalleiksi. (Alammar, J. 2024, alaluvut “Attention Is All You Need”, ”Representation Models: Encoder-Only Models” ja “Generative Models: Decoder-Only Models”)

2.4 Sidosryhmät työpaikalla

Sisäisiä sidosryhmiä on monia. Uusien ominaisuuksien kehittämiseen vaadittavat päätökset kulkevat tuotetiimin kautta. Tuotetiimi ideoi, validoi ja määrittelee uusia ominaisuuksia, jonka jälkeen ne siirtyvät kehitystiimin työlistalle. Bugeja voi ilmoittaa kuka tahansa organisaatiossa, mutta suurin

osa bugi-ilmoituksista tulee asiakaspalvelun kautta loppukäyttäjiltä. Myös muut sisäiset sidosryhmät ilmoittavat bugeista liittyen erityisesti sisäisiin työkaluihin, joita on käytössä monia.



Kuva 1 Sidosryhmät

2.5 Vuorovaikutustaidot työpaikalla

Kehitystiimi tekee töitään pääsääntöisesti etätyönä. Kehitystiimillä on päivittäinen lyhyt kokous, jossa kerrotaan päivän tehtävistä ja mahdollisista esteistä työntekoon liittyen. Viikon aikana on useita muita kokouksia liittyen muun muassa tehtävänkuvausten tarkentamiseen ja työn aikatauluttamiseen. Työyhteisö on kansainvälinen, jonka vuoksi työkielenä on englanti.

Etäkokoukset kaikkea ne vaativat osallistujilta aktiivista otetta. Omasta mielestäni esimerkiksi kameran päällä pitäminen on pieni teko, jolla voi viestiä olevansa läsnä kokouksessa. Sen lisäksi aktiivisuus kokouksissa vaatii hieman ylimääräistä ponnistelua, koska vuorovaikuttamiseen liittyvä sanaton viestintä puuttuu etäkokouksissa.

Työssä on myös tärkeää osata kommunikoida teknisistä asioista kehitystiimin ulkopuolisille henkilöille ymmärrettävällä tavalla. Joskus teknologia mahdollistaa asioita, joita on vaikea tietää, jos ei ole siihen perehtynyt ja toisaalta välillä teknologia asettaa esteitä tai hankaluuksia kehitysideoille.

3 Päiväkirja

3.1 Viikko 1

23.10.2023

Työskentelin aamun chattibotin kehityksen parissa. Projektissa hyödynnetään ElasticSearchia ja koodi, joka indeksoi datan hakukoneeseen oli hieman rikki. Ohjelmointivirheen syy selvisi suhteellisen nopeasti.

Iltapäivällä aloitin kirjoittamaan uutta ominaisuutta chattibotin taustajärjestelmiin. Chattibotin hyödyntämään ElasticSearchiin pitäisi indeksoida vielä uusia dokumentteja, joita chattibotti voi hyödyntää tietolähteenään. Ominaisuus on hyvin vastaava jo aiemmin kirjoittamani kanssa, joten sen toteuttaminen käy oletettavasti nopeasti.

24.10.2023:

Jatkoin uuden ominaisuuden kirjoittamista. Refaktoroin koodia hieman sillä huomasin, että koodissa alkoi olemaan toistoa. Olen joskus kuullut sanottavan, että kun koodatessa toistaa itseänsä kolme kertaa, on aika kirjoittaa jotain uusiksi.

Työstin uutta ominaisuutta, joka liittyi datan keruuseen ja aggregointiin alustaltamme. Sain ominaisuuden valmiiksi. Sen jälkeen parantelin aiemmin kirjoittamaani rajapintakoodia, jotta se hyödyntää paremmin Pythonin asyncio -moduulin ominaisuuksia sekä DTO-malleja.

DTO-mallien hyödyntämiseen jäi vielä töitä tulevaisuutta ajatellen. Tällä hetkellä mallit ovat hieman ”tynkiä” eli ne toimivat vain toiseen suuntaan. Tarkoittaen, että pystyn muuntamaan tietokantaentiteettejä Datoiksi mutta en tietokantaentiteettejä DTO -objekteiksi.

25.10.2023:

Jatkoin DTO-luokkien viilaamista. Nyt luokilla voi hoitaa myös tietokantaoperaatiot.

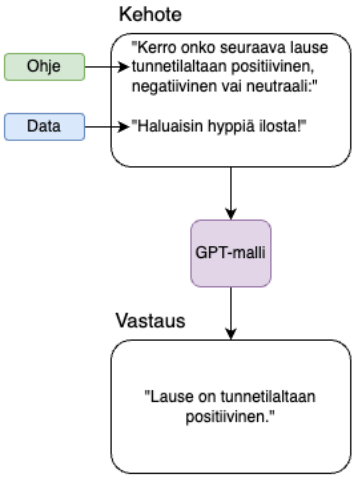
Näiden työstämisen jälkeen oli muutamia palavereja ja kirjoitin joitain SQL-kyselyitä liittyen datamigraatioihin. Chattibotti hyödyntää tekstitysdokumentteja, joiden tunnistenumerot ovat muuttuneet. Tunnistenumerot tulisi päivittää dokumentteihin. Keräsin tarvittavan datan ja muokkasin sitä hieman Excelissä. Nyt data on valmisteltu, jotta voin kirjoittaa skriptin, joka päivittää vanhat dokumentit.

26.10.2023

Tämän päivän aikana kirjoitin sovellukseen lisää päätepisteitä rajapintaan. Lisäsin mahdollisuuden käyttäjälle hakea kaikki käymänsä chatit sekä hakea viimeisimmän keskustelun.

Tämän lisäksi käytin melko paljon aikaa GPT-kielimallille syötetyn kehotteen viilaamiseen. Olemme käyttäneet projektissa tähän mennessä OpenAI:n gpt-3.5-turbo-1106-mallia, joka on keskustelumuotoisiin kehotteisiin optimoitu GPT-malli (Azure -dokumentaatio 2024). Minun on ollut mallin kanssa haasteena saada se seuraamaan kehotteissa annettuja ohjeita.

Kehote (engl. ”prompt”) on teksimuotoinen syöte, joka generatiiviselle tekoälymallille annetaan. Kehotteella voidaan ohjata mallin ulosantia haluttuun suuntaan. Jay Alammarin ja Maarten Grootendorstin kirjan ”Hands-On Large Language Models” alaluvussa ”The Basic Ingredients of a Prompt” kuvaillaan yksinkertaisen kehotteen rakenne kaksiosaiseksi. Se sisältää osiot ”ohje” ja ”data”. Ohjeella kuvataan tehtävää, jonka kielimallin halutaan suorittavan ja dataan sisällytetään annetun ohjeen kohde (kuva 2).



Kuva 2 Yksinkertainen kehoterakenne

GPT-3.5-mallista on tarjolla useita versioita rajapintojen kautta. Vaikuttaa siltä, että OpenAI Playgroundissa käytettävä versio mallista seuraa paremmin vastausten muotoiluun liittyviä kehotteita. Huomenna aion tutkia voiko käyttämämme Azure OpenAI -rajapinnan kautta vaihtaa uudempaan versioon kielimallista.

27.10.2023

Tänään kirjoitin testejä, joilla voi ajaa kyselyitä chattibottia vasten. Kyselyt muotoillaan seuraavanlaisesti JSON-muodossa:

```
[
    {
        "question": "Do you have content about Excel?",
        "language": "en"
    },
    {
        "question": "Onko teillä sisältöä Excelistä?",
        "language": "fi"
    },
    {
        "question": "Har ni innehåll om Excel?",
        "language": "se"
    }
]
```

Testiskripti ajaa jokaisen kyselyn chattibottia vasten ja tuottaa tiedoston, johon tallentuu lähetetty kysely, generoitu hakusana, GPT-mallin vastaus sekä hakusanalla löydetyt sisällöt alustalta.

Eilen tutkimani uudempi GPT-malli ei ollut vielä saatavana Azure-ympäristössä.

3.1.1 Viikkoreflektointi 23.-27.10.2023:

Viikko kului pienien juttujen viilaamisessa. Huomasin omassa koodissani paljon puutteita, joita korjailin samalla kun lisäsin ominaisuuksia kirjoittamaani sovellukseen. Olen kuullut sanottavan, että koodia tulisikin kirjoittaa uudelleen jatkuvasti, kun siihen lisää uusia ominaisuuksia tai korjaa bugeja. Viikko myös vahvisti osaamistani liittyen ElasticSearchiin. ElasticSearch on erinomainen työkalu, kun sen perusteet on saanut sisäistettyä. Ohjelmiston uusimmissa versioissa on tuotu mukaan vektorihakualgoritmeja, joita voi halutessaan yhdistää perinteisiin avainsanapohjaisiin hakuihin (ElasticSearch B.V.). Vaikka ElasticSearchin vektorihaku ei nopeudessa pärjää, joidenkin testien mukaan, muille vektoritietokannoille, sen kehittyneet ominaisuudet perinteisimmissä hakumuodoissa tekevät siitä oivan kandidaatin myös vektorihauille, sillä harvoin hakuja tehdään pelkästään vektorien perusteella.

Lisäksi työskentelin sovelluksemme hyödyntämän GPT-mallille syötettävien kehotteiden parissa. Kehotteiden viilaaminen on melko haastavaa, koska mahdollisia syötteitä, joita käyttäjä voi antaa chattibotille on käytännössä rajaton määrä. Kaikkien käyttötapausten testaaminen on mahdotonta.

Paras tapa testata lienee luoda joukko testikysymyksiä, jotka ajetaan aina kun sovellukseen tehdään muutoksia. Vastaukset tulisi validoida testaajien toimesta ja hyväksytylle testille tulisi antaa jokin prosenttimuotoinen raja-arvo, joka testipatterin kysymyksistä täytyy olla ”hyväksyttäviä”, jotta testi läpäistään.

Opinnäytetyön tavoitteista tämä viikko vahvisti erityisesti teknistä osaamista.

3.2 Viikko 2

30.10.2023

Chattibotin Pythonilla kirjoittamani sovellus alkaa olla perustoiminnallisuuksiltaan kunnossa.

Tänään dokumentoin sovellustason arkkitehtuuria. Lisäksi tutustuin hieman PHP-sovelluksen koodiin, sillä saan varmastikin itse kirjoittaa uutta koodia tähänkin sovellukseen. Olen aiemmin työskennellyt kyseisen sovelluksen parissa vain hieman.

Lisäksi kokoustimme yhden frontend-kehittäjämme kanssa liittyen chattibotin käyttöliittymään. Näytin hieman kehittämäni backend-sovelluksen rajapintaa ja sen tuotttamia vastauksia.

31.10.2023

Tänään kirjoitin alustavaa koodia PHP-sovellukseen, mutta en vielä edennyt kovin pitkälle sillä suunnitelmat sovelluksen osalta täytyy validoida muiden kehittäjien ja tuoteomistajien kanssa. Lisäksi viilasin Python-sovellusta ja GPT-mallille syötettäviä sisäisiä prompteja. Promptien saaminen kohdalleen on haastavaa. Chat-botille voi laittaa syötteeksi käytännössä mitä vaan, joka tekee kaikkien käyttötapausten testaamisen todella hankalaksi. Tällä hetkellä koostamme ’testipatteria’ kysymyksiä, joita voimme syöttää botille ja validoida näitä vastauksia.

1.11.2023

Tänään huomasin Python FastAPI-sovelluksessa pieniä ongelmia liittyen asynkroniseen koodiin. Käytin hieman aikaa optimoidakseni koodia, jotta se toimisi hyvin FastAPI-kirjaston kanssa. Ongelma oli siinä, että osa ohjelmointikirjastoista, joita käytän projektissa eivät sovellu suoraan Pythonin asynkronisiin ominaisuuksiin. Kirjastot ovat siis suunniteltu synkronisiksi. API-rajapinnan päätepisteet, jotka sisältävät tällaista synkronista koodia eivät saisi olla asynkronisiksi merkittyjä tai muuten synkroninen koodi saattaa hidastaa koodin suorittamista, kun palvelulle tehdään useita kyselyjä samanaikaisesti.
Tällä hetkellä FastAPI-sovellus pyörii kehitysympäristössä vain yhdellä Uvicorn-prosessilla, joka ei

myöskään ole kovin hyvä ratkaisu suorituskyvyn kannalta, jos sovellusta ajetaan tuotantoympäristössä. Tähän ongelmaan Uvicorn-dokumentaatio, tarjoaa helpoimmaksi ratkaisuksi käyttää Gunicornia, jota voi käyttää prosessinhallintatyökaluna (Uvicorn-dokumentaatio, osio ”Running with Gunicorn”). Tämä täytyy vielä laittaa kohdalleen ennen kuin sovellus siirtyy testiympäristöihin.

2.11.2023

Tänään meillä oli toimistolla ’työpaja’ liittyen tekoälyprojektiimme. Keskustelimme päivän mittaan projektin eri osa-alueista: teknisestä toteutuksesta, brändäyksestä ja käyttäjälähtöisestä kehityksestä.

Päivän aikana opin paljon tuotekehityksen eri osa-alueista ja erityisesti miten monialaista osaamista toimivan tuotteen kehittäminen vaatii. Esimerkiksi käyttäjien tarpeiden kartoittaminen ei ole helppo tehtävä. Analyysiin voi käyttää esimerkiksi dataa palvelun käyttämisestä tai käyttäjäkyselyitä/-haastatteluita. Nämäkään tekniikat eivät kuitenkaan anna valmiita vastauksia, vaan niistä on sovellettava ratkaisu, jonka käyttökokemus on sujuva.

3.11.2023

Pidin lyhyemmän päivän tänään. Iltapäivällä oli koolle kutsumani palaveri, jossa keskustelimme muiden backend-kehittäjiemme kanssa Symfony-sovellukseen tehtävistä muutoksista liittyen tekoälyprojektimme beta-julkaisuun. Symfony on avoimen lähdekoodin PHP-ohjelmointikehys, joka tarjoaa ohjelmistoarkkitehtuurin, -komponentteja ja -kirjastoja, joiden tulisi nopeuttaa websovellusten kehittämistä (Porebski, B., Przystalski, K., Nowak L. 2011, luku 1.2.2.).

Palaverissa tuli hyvää palautetta liittyen suunnitelmiini ja sain suunnitelmia hieman järkeistettyä. Ensi viikolla pystyn aloittamaan PHP-koodin kirjoittamisen kunnolla.

3.2.1 Viikkoreflektointi 30.10.2023 – 3.11.2023

Tämä viikko oli hieman sekalainen, koska työskentelin sekä FastAPI-sovelluksen että Symfony-sovelluksen parissa, jonka lisäksi meillä oli ”toimistopäivä”, jotka ovat melko harvinaisia. Toimistolla käyminen teki ehdottomasti hyvää ja mietinkin että voisin ottaa tavoitteeksi käydä toimistolla työskentelemässä esimerkiksi joka toinen viikko ainakin yhden päivän ajan. Monia asioita on helpompi kommunikoida kollegoiden kanssa kasvotusten ja muutenkin on virkistävää nähdä työkavereita kasvotusten. Tämä viikko edisti opinnäytetyöni tavoitteita liittyen vuorovaikutustaitoihin ja projektiosaamiseen.

Viikon tehtävät olivat melko sekalaisia, koska vaihdoin liukuen kehitystyötä Symfonyn puolelle. Symfonyssa on opeteltavaa ja odotan, että ensi viikon alku voi olla hieman takkuista tämän takia.

3.3 Viikko 3

6.11.2023

Tänään tein hieman lyhyemmän päivän, koska minulla on kertynyt jonkin verran tunteja tuntipankkiin. Aloin kirjoittamaan PHP-sovellukseen koodia. Ensiksi generoin OpenAPI -generaattorilla rajapintakirjaston kehittämääni Python-sovellusta vasten. OpenAPI on standardi, jolla voi kuvailla HTTP-pohjaisten rajapintojen rakennetta käyttäen JSON- tai YAML-tiedostoa (Rosenstock, L., Ponelat, J. 2022, luku 1.3.). OpenAPI -generaattorilla voi luoda OpenAPI -kuvaukseen perustuen rajapintakirjastoja eri kielille. Suurin osa ajasta kului generoidun kirjaston käyttöönotossa. Aivan päivän päätteeksi aloin kirjoittamaan logiikkaa rajapintakirjaston ympärille.

7.11.2023

Jatkoin PHP Symfony -sovelluksen työstämistä. Minulla on ollut hieman vaikeuksia päästä vauhtiin kehityksen kanssa, koska Symfony ei ole ennestään kovin tuttu minulle. Lisäksi taistelin ongelman kanssa, joka minulla on ollut aiemminkin eli PHP Xdebug -työkalun konfiguroinnin kanssa. Xdebug on työkalu, joka mahdollistaa pysäytyspisteiden asettamisen koodiin, jotka pysäyttävät koodin suorittamisen ja mahdollistavat sovelluksen tilan tarkkailun tuossa pisteessä (Mann, E. 2023. luku 13). Tällä kertaa kirjoitin projektin README-tiedostoon ylös ohjeet debuggaajan asetusten viilaamiseen, jotta jatkossa saan sen toimimaan nopeammin.

8.11.2023

Jatkoin Symfony-sovelluksen API-päätepisteiden työstämistä ja sain päivän päätteeksi ensimmäisen pull requestin laitettua katselmoitavaksi. Olen kirjoittanut sen verran vähän PHP:tä, että odotan muilta kehittäjiltä tulevan jotain kommenttia koodiini. Symfonyn perusominaisuudet alkavat pikkuhiljaa hahmottumaan, mikä helpottaa tekemistä. Lisäksi meillä oli kokous liittyen tekoälyprojektiin. Kokouksessa keskityttiin enempi käyttöliittymään kuin teknisiin asioihin. Huomenna jatkan uusien juttujen parissa Symfonyn kanssa.

9.11.2023

Jatkoin Symfony-sovelluksen parissa. Kirjoitin logiikkaa käyttäjän pääsyn auktorisoimiseksi uusiin ominaisuuksiin. Lisäksi taistelin hieman OpenAPI PHP-annotaatioiden kanssa, joita en ole käyttänyt ennen.

Iltapäivällä pidin projektista pienen esittelyn uudelle puolalaiselle työkaverilleni. Hän liittyy projektiin ja onkin hyvä saada projektiin mukaan uusi kehittäjä.

10.11.2023

Kirjoitin Symfony-sovellukseen uusia päätepisteitä rajapintaan. Kysyin hieman apua työkaveriltani, joka antoi hieman vinkkejä jo aiemmin kirjoittamaani koodiin liittyen käyttäjien auktorisointiin. Samalla, kun kirjoitin uutta koodia refaktoroin hieman aiemmin kirjoittamaani työkaverini neuvojen perusteella. Muutosten jälkeen uusia päätepisteitä määritellessä voi päätepisteen merkitä syntaksilla, joka takaa, että käyttäjällä on tarvittavat käyttöoikeudet kutsujen tekemiseen kyseiseen päätepisteeseen.

3.3.1 Viikkoreflektointi 6.-10.11.2023

Työskentelin viikon aikana paljon PHP:n parissa. Viikon alussa eteneminen oli hieman hidasta, koska en ole hetkeen kirjoittanut PHP ja Symfony-juttuja. Muutenkin kontekstin vaihtaminen ohjelmointityössä vie aina aikaa. Jo kehitysympäristön pystyttäminen ja työkalujen konfiguroiminen on usein yllättävän työlästä. Juttelin jonkin verran alkuviikosta kokeneemman PHP-kehittäjän kanssa ja häneltä tuli paljon hyviä vinkkejä ja yleiskuvaa Symfony-kirjastosta. Huomaan, että minulla on vielä paljon oppimista olio-ohjelmointiin liittyen. Joidenkin konseptien, kuten riippuvuuksien injektoimisen (engl. ”dependency injection”), soveltaminen käytännössä vaatii vielä harjoittelua. Riippuvuuksien injektoiminen on tekniikka, jolla pyritään mahdollistamaan ohjelmakoodi, joka on vain löyhästi kytketty ympäröivään koodiin (Seemann, M. ja Deursen, S. 2019, luku 1.1.). Riippuvuuksien injektoiminen mahdollistaa mm. koodin helpomman testaamisen.
Yleisesti viikko sujui ihan kohtuullisesti, vaikka en ehkä edennyt ihan niin nopeasti kuin toivoin. Etukäteen arvioidessa kuinka kauan jokin ohjelmointitehtävä vie aikaa, ei käytännössä koskaan osaa ottaa huomioon kaikkia tekijöitä. Joskus olen kuullut sanottavan, että arvioidessa työtehtävien pituutta kannattaa noudattaa seuraavanlaista kaavaa: tee ensimmäinen aika-arvio, tuplaa ensimmäinen arvio ja lisää vielä siihen kolmekymmentä prosenttia. Viikko edisti teknistä osaamistani liittyen PHP-ohjelmointikieleen ja Symfony-kehykseen.

Työstämääni projektiin saadaan nyt lisää käsiä tekemään, mikä varmasti edistää projektia nopeammin ja myös auttaa pitämään laadun korkealla. Kun enemmän silmäpareja on tarkistamassa ohjelmiston toteutusta, on lopputulema parempi.

3.4 Viikko 4

13.11.2023

Jatkoin vielä Symfony-sovelluksen parissa. Viimeistelin vielä perjantaina kirjoittamaani koodia ja laitoin koodin eteenpäin katselmoitavaksi. Tämän jälkeen aloin jälleen työstämään Python-sovellusta. Rajapinnasta puuttuu vielä joitain päätepisteitä.

14.11.2023

Toteutin Python-sovellukseen rajapintaan puuttuvia päätepisteitä. FastAPI-sovelluksessa on vielä paljon työstettävää. Olemme alustavasti puhuneet projektiin liittyneen kehittäjän kanssa millaisia muutoksia tulemme tekemään sovellukseen, jotta sen ylläpitäminen ja automaattinen testaaminen on helpompaa tulevaisuudessa. Hän on puhunut jonkin verran niin sanotusta ”Domain Driven Development” -mallista, joka tähtää siihen että koodipohja ja yrityskieli käyttävät samaa terminologiaa. Minun tulee vielä opiskella aihepiiriä ymmärtääkseni sitä paremmin.

15.11.2023

Työskentelin aamulla ElasticSearch-ongelman parissa. Olemme nostamassa ElasticSearchin (ES) versionumeroa v7:stä v8:aan järjestelmässämme. PHP-sovelluksemme ei aivan suorilta ollut valmis vaihtoon. Ongelma kulminoitui lopulta hyviin pieneen asiaan, joka liittyi ElasticSearchin käyttämiin aikaformaatteihin. Vaihdoimme erään kentän aikaformaatin Unix-muotoon, jonka jälkeen sovellus vaikuttaa toimivan niin kuin ennenkin. Toinen ongelma liittyi ElasticSearchin Docker-kuvaan asetettuun asetukseen, joka esti uusien indeksien luomisen automaattisesti. Ongelmat eivät vaatineet suuria koodimuutoksia, mutta niiden selvittämiseen kului paljon aikaa.

Työkaverini katselmoi eilen PHP-koodiani ja teki muutosehdotuksia, jotka toteutin ja sen jälkeen työstin samojen neuvojen avulla toisen päätepisteen koodin vastaamaan ehdotuksia.

16.11.2023

Aamulla katselmoin työkaverini frontend-koodia.

Olemme keskittyneet nyt Python-sovelluksen viemiseen testiservereille. Työkaverini on kysellyt jonkin verran liittyen sovellukseen, sillä hän työstää testiympäristön pystyttämistä ja tuohon ympäristöön sovellusten julkaisun putkittamista. Lisäksi toisella työkaverillani on ollut pienen tauon jälkeen ongelmia saada hänen kehitysympäristönsä pystyyn, jossa olen yrittänyt avustaa.

Hieman koodiakin sain kirjoitettua. Python-sovelluksessa on tullut ilmi pari bugia, joita ehdin hieman tutkia ja myös aloittamaan korjausta.

17.11.2023

Työskentelin tänään vain muutaman tunnin. Meillä oli tiimimme kokous iltapäivällä, jossa keskusteltiin tavoitteista saada sovellus testiympäristöihin ensi viikon alkupuolella. Meillä on ensi viikolla tapaaminen toimistolla, jossa haluamme päästä koekäyttämään sovellusta.

Kokouksen jälkeen keskustelimme työkaverini kanssa hieman vielä Python-sovelluksen arkkitehtuurista, ja hän näytti minulle hieman käytännössä minkälaista rakennetta hakee ajatuksillaan koodille.

3.4.1 Viikkoreflektointi 13.11.2023–17.11.2023

Viikon teemana on ollut monipuolisuus ja jatkuva oppiminen ohjelmistokehityksen eri osa-alueilla. Viikon aikana työskentelin useiden teknologioiden parissa: viimeistelin Symfony-sovellusta, kehitin Python-sovellusta FastAPIn kanssa, ja ratkoin ElasticSearchin päivityksestä johtuvia ongelmia PHP-sovelluksessamme. Ongelmien ratkaiseminen ja uusien konseptien, kuten Domain Driven Developmenttiin tutustuminen, korostaa uusien asioiden oppimisen tärkeyttä alalla.

Lisäksi tiimityöskentely ja yhteistyö nousivat viikon aikana keskeisiksi. Koodin katselmointi, työkavereiden auttaminen teknisissä ongelmissa, ja keskustelut Python-sovelluksen arkkitehtuurista toivat esille tiimin merkityksen ja yhteistyön tärkeyden kehitysprojekteissa. Yhteenvetona, viikko on tarjonnut laajan kirjon teknisiä haasteita ja mahdollisuuksia oppia tiimityöskentelyn arvosta ohjelmistokehityksessä.

3.5 Viikko 5

20.11.2023

Jatkoin viime viikolla ilmenneiden bugien työstämistä. Olen tehnyt kehitystyötä toistaiseksi pitkälti lokaalissa ympäristössä, enkä osannut/muistanut ottaa huomioon tiettyjä muuttujia, jotka tulevat kuvioon kun sovellusta viedään lähemmäs tuotantoa vastaavaa ympäristöä. Tästä johtuen, on ilmennyt muutama ongelma, jotka ovat kuitenkin (onneksi) olleet suhteellisen helppoja ratkaistavia.

Keskustelin myös jonkin verran projektin käyttöliittymäkehittäjän kanssa liittyen rajapinnan vastausformaatteihin.

21.11.2023

Aloitin päivän hieman tavallista myöhemmin. Keskustelimme pitkän tovin projektiin jokin aika sitten liittyneen itseäni kokeneemman kehittäjän kanssa. Häneltä on tullut kehittävää palautetta projektista ja otamme lähiaikoina tavoitteeksi uudelleen järjestellä koodipohjaa (ns. refaktorointi), jotta sen ylläpidettävyys ja koodista keskustelu olisi hieman helpompaa. Olen pitkälti koodannut uuden taustasovelluksen itsekseni ja kokemattomuuttani koodista on puuttunut rakennetta. On erittäin hyvä, että projektiin on liittynyt uusi kehittäjä, jolla on tällaista näkemystä.

22.11.2023

Tänään olemme keskittyneet siihen, että saamme sovelluksen toimimaan testiympäristössämme. Sovellus hyödyntää ElasticSearchiin rakennettua hakuindeksiä, jonka rakentaminen on melko pitkä prosessi. Koko indeksin rakentaminen alusta loppuun kestää noin 2 tuntia. Suurin syy pitkälle ajalle on se, että vektoroimme suuren määrän tekstityksiä text-embedding-ada-002-tekstivektorointimallin läpi. Malli tuottaa 1536 ulottuuvuksisen vektorirepresentaation sille annetusta tekstisyötteestä ja se on suoriutunut paremmin suurimmassa osassa tehtyjä testejä, kuin aiemmat OpenAI:n julkaisemat vektorirepresantaatiomallit (OpenAI, Inc. 2022). Rajapinta, jonka kautta vektorointia tehdään, sallii vain tietyn määrän liikennettä minuutissa, joka on suurin rajoittava tekijä ajan puolesta indeksiä rakennettaessa. Laskin arviolta, että ajoimme mallin läpi noin 20 miljoonaa sanaa. Indeksi rakentui alusta loppuun onnistuneesti. Tämän jälkeen manuaalisesti testailin sovellusta ja varmistin että se toimii, sillä huomenna vuorossa on koko tiimin kesken työpaja toimistolla liittyen ominaisuuteen.

23.11.2023

Tänään suuntasin toimistolle pitkästä aikaa. Tiimillämme oli työpaja liittyen projektiin, jonka kanssa olen työskennellyt viime aikoina. Saapuessani toimistolle huomasin heti, että sovellus ei toiminut, joka aiheutti hieman hikoilua. Jostain syystä taustasovelluksen tietokantayhteys oli mennyt poikki yön aikana, mutta yhteys korjaantui itsekseen melko pian sen jälkeen, kun sitä vasten oli lähettänyt muutaman kutsun. Tein ongelmasta tiketin, mutta en päässyt perehtymään sen juurisyyhyn vielä.

Työpajassa testailimme sovelluksen nykyistä versiota, ideoimme uusia ominaisuuksia, tunnistimme heikkouksia sekä mahdollisia ratkaisuja niihin ja kirjasimme projektille seuraavia askeleita. Ilmi tuli myös muutamia bugeja käyttöliittymä- että taustasovellustasolla. Teki hyvää nähdä tiimiläisiä kasvotusten ja vaihtaa ajatuksia projektiin liittyen. Kasvotusten monista asioista on helpompi keskustella ja saada viestinsä perille. Kaiken kaikkiaan päivä oli onnistunut.

24.11.2023

Tänään yritin hieman järjestellä tikettejä Kanban-taulullamme, jotta saisin hieman selkeämmän kuvan siitä mihin keskittyä seuraavaksi. Sen jälkeen yritin tutkia eilen ilmennyttä ongelmaa tietokantayhteyden kanssa, mutta en oikein päässyt ongelman juurisyyn ääreen, koska ongelman toistaminen ei onnistunut. Keskustelin aiheesta myös työkaverini kanssa, mutta asiaan ei tullut lisää selkoa. Päädyimme siihen että kommentoimme havaintomme asiasta tehtyyn tikettiin ja jätämme sen vielä seurantatilaan, jos ongelma toistuu.

Toinen ongelma liittyi mahdolliseen muistivuotoon sovelluksessa. ElasticSearch koodista puuttui koodinpätkä, joka sulkee yhteyden ElasticSearchiin, joka aiheutti RAM-muistin kanssa ongelmia, kun monta käyttäjää käytti sovellusta samanaikaisesti. Sain ongelman korjattua suht nopeasti.

3.5.1 Viikkoreflektointi 20.11-24.11.2023

Työpaja toimistolla oli tuottoisa päivä ja se kirkasti hyvin projektin tavoitteita ja seuraavia askeleita. Lisäksi teki hyvää käydä toimistolla näkemässä kasvotusten työkavereita.

Projekti etenee ja olemme saaneet sovelluksen käyttöliittymän ja taustasovelluksen julkaistuiksi testiservereillämme. Täytyy muistaa iloita pienistä saavutuksista matkan varrella. Uusien kehittäjien liittyminen projektiin on ollut erittäin positiivinen asia, vaikka alkuun se onkin vaatinut itseltä paljon kommunikointia joka suuntaan. Koodipohjan uudelleen muotoilu on hyvä ottaa työn alle jo tässä vaiheessa projektia.

Olen varma, että tulen oppimaan uudelta kokeneemmalta työkaveriltani paljon. Pystyn jo tuottamaan toimivaa koodia, mutta erityisesti automaattisesti testattavan koodin osalta minulla on vielä paljon opittavaa.

Khorikov ilmaisee kirjassaan ”Unit Testing Principles, Practices and Patterns”, että ohjelmakoodi, jota on hankala yksikkötestata, on yleensä myös heikkolaatuista koodia. Hänen mielestään yksikkötestien päämäärä on mahdollistaa ohjelmistoprojektin kestävä kasvu. Ilman testejä uusien ominaisuuksien kehittäminen sekä ohjelmointivirheiden korjaaminen hidastuu mitä suuremmaksi ohjelmistoprojekti kasvaa. (Khorikov, V. 2020, luku 1.2.)

Yksikkötesti määritellään kirjassa seuraavasti; yksikkötesti testaa pienen osan ohjelmakoodia (eli yksikön), testi suoritetaan nopeasti ja testi tapahtuu eristettynä muista ohjelmakoodin osista. Kirjassa pureudutaan vielä syvemmälle muun muassa siihen miten eristetyn koodiyksikön voi määritellä. (Khorikov, V. 2020, luku 2.1.)

Viikko kehitti sekä projektiosaamiseen liittyviä taitojani sekä teknistä osaamista liittyen automaattiseen ohjelmistotestaamiseen.

27.11.2023

Sovellukseen on vielä toteuttamatta putki, joka prosessoi ja indeksoi opintopolkudokumentteja. Aloin suunnittelemaan koodia tätä ominaisuutta varten. Ominaisuus vastaa paljolti jo aiemmin kirjoittamiani putkia. Iltapäivällä aloitin ominaisuuden kirjoittamisen ja etenenkin hyvin nopeasti siinä.

Kun oli tarkoitus alkaa testailemaan ominaisuutta, ilmeni joitain ongelmia kehitysympäristössä, jotka hidastivat työtä. Jouduin taistelemaan noin tunnin ongelmien kanssa ennen kuin pääsin kokeilemaan uutta koodia.

Keskustelimme myös toisen kehittäjän kanssa sovelluksen arkkitehtuurista. Nykymuodossaan sovellus jakaa tietokannan toisen sovelluksen kanssa, joka ei ole kovin hyvä ratkaisu pitkässä juoksussa, sillä tämän kaltaisista jaetuista riippuvuuksista ilmenee usein yllättäviä ongelmia. Jos joku kehittää toista sovellusta joka jakaa tietokantaa kehittämämme sovelluksen kanssa, hän saattaa tehdä huomaamattaan/tietämättään muutoksia jotka rikkovat toisen sovelluksen. Puhuimmekin alustavasti, että yritämme purkaa tämän riippuvuuden mahdollisimman pitkälle lähiaikoina.

3.6 Viikko 6

28.11.2023

Tänään työstin bugia / uutta toivetta, jonka vaatimuksena oli, että chatbotti palauttaa maksimissaan kolme kurssia suositellessaan sisältöä. Ongelman ratkaiseminen ohjelmallisesti on hieman hankalaa. Kielimallille tarjotaan kontekstina hakutuloksista sisältöä, josta se voi poimia käyttäjän kysymykseen/viestiin sopivia ehdotuksia. Voisimme rajata hakutuloksia siten, että näytämme mallille vain 3 hakutulosta, mutta tämä taas saattaisi vaikuttaa vastausten laatuun negatiivisesti.

Paras ratkaisu tässä kohtaa lienee pyytää kielimallia kauniisti palauttamaan maksimissaan kolme suositusta vastauksessaan. Tässä ongelmana on se, että käyttämämme GPT-malli on hieman oikukas seuraamaan annettuja pyyntöjä. OpenAIn uusin GPT-malli vaikuttaa kuitenkin pärjäävän ohjeiden seuraamisessa paremmin. Me emme vielä pysty hyödyntämään mallia ikävä kyllä sillä Azuren pilvipalvelut eivät vielä tarjoa mallia vasta kuin esikatselussa. Olemme jo tehneet pieniä testejä sillä ja malli vaikuttaakin huomattavasti paremmalta ohjeiden seuraamisessa.

Kielimallien (ja jossain määrin vektorihaun) kanssa painiminen on omanlainen haasteensa. Olen hieman miettinyt jo miten näitä tulisi testata ja juttelin hieman työkaverini kanssa tästä aiheesta. Ehdotammekin muille projektissa työskenteleville, että alamme kehittämään pientä työkalua, jolla pystymme automaattisesti arvioimaan kielimalleihin nojaavia osia sovelluksestamme eli hakua ja vastausten generointia.

29.11.2023

Keskustelimme lisää työkalusta jota aiomme kehittää lähiaikoina. Haluamme aloittaa sovelluksen hakuominaisuuksien testaamisella systemaattisesti. Työkaverillani oli hyviä ideoita ominaisuuden

testaamisesta ja hän esitteli myös testaamiseen liittyviä periaatteita ja kuinka olio-ohjelmointi voi auttaa niiden toteuttamisessa.

Työstimme yhdessä vielä sovelluksen koodipohjaa, jotta sen testaaminen olisi helpompaa jatkossa.

30.11.2023

Kokeilimme testiympäristössä uutta sisäistä promptia käyttämällemme kielimallille, jotta se tuottaisi vastauksia useammin haluamassamme muodossa. Teimme melko suuria muutoksia promptiin ja tulokset eivät olleet kovin hyviä. Uudella promptilla bottimme tuotti jatkuvasti ”hallusinoituja” vastauksia, joten päätimme palata aiempaan promptiin.
Tämä kokemus oli omiaan osoittamaan, että tarvitsemme systemaattisen tavan testata kielimallin tuottamia vastauksia. Kun teemme muutoksia, jotka vaikuttavat kielimallin ulosantiin, emme voi joka kerta olettaa testaajan käyvän läpi kymmeniä eri skenaarioita ja arvioivan vastauksia manuaalisesti. Esittelimmekin idean testaustyökalusta tuoteomistajalle ja testaajille, jotka ymmärsivät tarpeen. Tuon työkalun toteuttaminen siirtyykin prioriteeteissamme melko korkealle.

1.12.2023

Tänään aloitin kirjoittamaan koodia palauteominaisuutta varten chattibotille. Tuotteelle haluttiin helppo tapa kerätä käyttäjiltä palautetta. Käyttöliittymätasolla tämä tarkoittaa mahdollisuutta antaa botin luomalle viestille palautetta vapaamuotoisella tekstikentällä sekä yläpeukku/alapeukku -napilla. Taustajärjestelmissä tämä tarkoittaa uutta endpointia, joka tallentaa tietokantaan palautteen sekä viittauksen viestiin johon palaute liittyy. Pääsin melko pitkälle tuon kirjoittamisessa ja työkaverini jatkoi siitä mihin jäin lopetellessani päivän.

3.6.1 Viikkoreflektointi 27.11.-1.12.2023

Viikko sisälsi ohjelmakoodin kirjoittamista, uusien työkalujen suunnittelua ja nykyisen koodipohjan parantamista.

Keskustelumme sovelluksen arkkitehtuurista paljasti joitain riippuvuuksia sovelluksien välillä, jotka olisi hyvä purkaa pitkässä juoksussa. Sovimme pyrkivämme purkamaan nämä riippuvuudet pian.

Lisäksi kohtasimme ongelmia kielimallin ulosannin kanssa, joka osoitti myös muille tiimiläisillemme, että tarvitsemme jonkinlaisia työkaluja kielimallien testaamiseen. Sovimmekin, että työkalun kehittäminen tulee olla korkealla prioriteettilistallamme. Uskon, että tämä työkalu tulee olemaan arvokas lisä laadunvarmistuksessamme ja kyvyssämme kehittää kielimallipohjaisiaratkaisuja nopeasti lähitulevaisuudessa.

Viikko oli täynnä teknisiä haasteita ja uusia oivalluksia. Henkilökohtaisesti sain arvokasta oppia joustavuudesta, ennakoivasta ongelmanratkaisusta ja tiimityöskentelystä.

3.7 Viikko 7

4.12.2023

Katselmoin aamusta työkaverini viimeistelemää koodia, joka näytti oikein hyvältä. Palautteen lisäksi lisäsimme metadata -taulun jokaisen botin tuottaman viestin yhteyteen. Tuossa taulussa pidetään muun muassa kielimallin generoima hakusana keskustelulle. Tuolla hakusanalla haetaan vektorihaulla sisältöä alustaltamme, jota kielimalli voi hyödyntää vastauksissaan.

Jatkoimme vielä koodipohjan refaktorointia. Lisäsimme koodipohjaan riippuvuuksien ”injektoimista” (dependency injection). Käytännössä tämä tarkoittaa, että jokaiselle loogiselle kokonaisuudelle määritellään käyttöliittymä/rajapinta, jota vasten uusia komponentteja toteutetaan. Ohjelmaan määritellään riippuvuuksia komponenteista ja nuo riippuvuudet populoidaan/täytetään ohjelmallisesti. Tällä saavutettava hyöty on se, että yksittäisten komponenttien testaaminen on tällöin paljon helpompaa. Voimme esimerkiksi vaihtaa komponentteja helposti ’jäljitelmäkomponentteihin’ (mocked component) käyttämällä samaa käyttöliittymää, kun haluamme testata vain jotain tiettyä osaa ohjelmastamme.

5.12.2023

Huomasimme aamulla ongelman tietokantatasolla liittyen kehittämäämme palauteominaisuuteen. Koska noita palautteita kirjoitettiin samaan tietokantaan, jota toinen sovellus käyttää, kohtasimme ongelman, jossa toisen sovelluksen käyttämä ORM-kirjasto ei tunnistanut yhden sarakkeen formaattia, joka aiheutti sovelluksen kaatumisen.
Ongelma osoitti toteen minkälaisia ongelmia tietokantojen (ja muiden riippuvuuksien) jakaminen sovellusten kesken voi aiheuttaa. Keskustelimme aiheesta työkaverini kanssa jo pari viikkoa sitten ja päätimme että nyt on aika siirtää kehittämämme sovellus omalle tietokannalleen. Uuden tietokannan ja tietokantakäyttäjien konfigurointi oli tehty sisäisten työkalujemme puolesta helpoksi. Ongelmaksi tuli, että osa tauluihin tallennettavista tiedoista oli relaatiossa vanhassa tietokannassa oleviin tauluihin. Teimme toistaiseksi päätöksen purkaa nuo relaatiot ja kirjoittaa puuttuvat tiedot uusiin sarakkeisiin tauluissa. Käytännössä tämä tarkoittaa, että toistaiseksi toisinnamme jonkin verran dataa, kun tallennamme uusia viestejä ja niihin liittyviä sisältöjä alustallamme. Aiomme korjata tämän ongelman kuitenkin lähiaikoina.

6.12.2023

Itsenäisyyspäivä, vapaa.

7.12.2023

Keskustelimme työkaverini kanssa hieman siitä miten FastAPI -sovellustamme pyöritetään jatkossa. Olemme käyttäneet kehityksessä Uvicorn ASGI -palvelinta, joka on suunniteltu pyörittämään asynkronisia Python-sovelluksia. Haluamme kuitenkin pyörittää useampia Uvicorn-prosesseja samanaikaisesti ja pystyä jakamaan kuormaa niiden välillä. Tämän vuoksi päätimme, että tuomme Uvicorn-instansseja käynnistämään ja niiden välillä kuormaa jakamaan (hämmentävän samantyyppisellä nimellä varustetun) Gunicornin. Gunicorn on varsinaisesti WSGI-palvelinsovellus, mutta siinä on ominaisuuksia, jotka ovat hyödyllisiä myös Uvicornin kanssa. Tärkeimpänä se, että se pystyy käynnistämään Uvicorn-sovelluksia ja tasaamaan kuormaa/kyselyjä näiden sovellusinstanssien välille.

Tämän asian tutkimiseen ja selvittelyyn meni osa päivästäni. Varsinaiset koodimuutokset olivat suhteellisen yksinkertaisia, sitten kun kokonaiskuva oli selvä. Huomasin päivän lopulla myös, että FastAPI-sovelluksen käynnistämisestä täytyy irrottaa erilliseen käynnistysskriptiin Temporal Worker -prosessien käynnistäminen. Temporal Worker -prosessi poimii Temporal serverin suoritusjonosta tehtäviä suoritettavaksi ja palauttaa niiden lopputulokset takaisin serverille (Temporal Technologies, Inc.). Mietin ongelmaa, mutta en päässyt siinä mihinkään lopputulemaan. Kerroin havainnostani työkaverilleni, jolla oli ajatus siitä, miten asia tulisi hoitaa.

8.12.2023

Tänään ”julkaisimme” sovelluksen nykyisen version sisäiseen käyttöön yrityksessämme. Tarkoituksena on kerätä palautetta ja ideoita sovelluksen jatkokehittämisestä lähitulevaisuudessa. Osa päivästä kului seuratessa, että sovellus toimii testiympäristössä ja tuotin myös muutamia CSV-raportteja liittyen sovelluksen kautta annettuihin palautteisiin. Paljon aikaa meni sisäiseen kommunikointiin tänään.

Huomasin myös bugin siinä, kuinka generoitu hakusana tallennetaan tietokantaan ja kirjoitin siihen korjauksen.

3.7.1 Viikkoreflektointi 4.12.2023-8.12.2023

Viikon aikana ratkaistiin muutamia ongelmia liittyen sovellukseen, jotka olivat jääneet muiden tehtävien taakse prioriteettilistalla. Ongelmat liittyivät jaettujen riippuvuuksien purkamiseen ja sovelluksen pyörittämiseen palvelimella. Viikko sisälsi paljon uutta ja melko spesifiä teknistä opeteltavaa liittyen aiheisiin kuten FastAPI, Uvicorn, Gunicorn ja Temporal.

Julkaisimme sovelluksen sisäiseen käyttöön, jotta saisimme kerättyä ideoita ja palautetta sovelluksen jatkokehitykseen. Tiedämme jo, että botin tarjoamien vastausten laadussa on parannettavaa vielä. Ensi viikolla pääsemme toivottavasti aloittelemaan automaattisen testityökalun kirjoittamista sovellukselle, joka tulee valmistuttuaan nopeuttamaan huomattavasti sovelluksen generatiivisten osien kehittämistä haluttuun suuntaan.

Kaiken kaikkiaan viikko oli antoisa, keskittyen tekniseen nippelitietoon ja sen tiedon tuomiseen käytäntöön.

3.8 Viikko 8

11.12.2023

Katselmoin työkaverini toteuttaman erillisen käynnistyksen Temporal Worker -prosesseille. Hän hyödynsi prosessien käynnistämiseen Supervisor -nimistä prosessinhallintatyökalua. Supervisor on UNIX-pohjaisille käyttöliittymille suunniteltu Python-ohjelma, jolla voi hallinnoida ja monitoroida prosesseja (Agendaless Consulting and Contributors, 2024). Nyt Supervisor pitää huolen, että Temporal Worker -prosessit käynnistyvät, kun sovelluksen Docker -kontti käynnistyy sekä, jos prosessit kaatuvat syystä tai toisesta, ne käynnistetään uudelleen.

Lisäksi suunnittelimme tarkemmin testityökalua, jonka kirjoittamista aiomme aloittaa pian.

12.12.2023

Aamupäivästä huomasimme testiympäristössä ongelmia sen jälkeen, kun otimme uuden version käyttöön sovelluksesta. Saimme painia ongelman kanssa jonkin aikaa. Yritin auttaa ongelman selvittämisessä parhaani mukaan, mutta lopulta kollegani löysi ongelman, joka johtui ilmeisesti vahingosta, jossa ympäristön pystyttämisessä käytetty skripti oli manuaalisesti keskeytetty kesken suorituksen. Tämä sekoitti testiympäristön ympäristömuuttujat, joka rikkoi sovelluksen.

Tämän jälkeen aloin orientoitumaan suunnittelemamme testityökalun toteuttamiseen. Ensimmäisen testipatterin tarkoituksena on testata sovelluksemme hakuominaisuuksia. Haku on kaksivaiheinen

– ensimmäisessä vaiheessa generoidaan ja vektoroidaan kahden eri tekoälymallin avulla hakusana chatin keskusteluhistoriasta, jonka jälkeen suoritetaan itse haku ElasticSearchia vasten. Tätä varten tarvitsemme testidataa eli testitapauksia sekä sisältöä alustaltamme, jota vasten hakuja tehdään. Aloimme työkaverini kanssa hahmottelemaan testauskoodin rakennetta ja kirjoittamaan testejä.

13.12.2023

Luin ja katselmoin koodia, jota työkaverini oli tuottanut eilen illalla. Hän oli päässyt jo pitkälle testien perusrakenteessa.

Jatkoin koodin kirjoittamista. Kirjoitin koodia, joka luo uuden ElasticSearch-indeksin ja indeksoi sinne testisisältöä. Indeksin rakenne vastaa varsinaisen sovelluksen indeksin rakennetta.

14.12.2023

Kirjoitin logiikka, jolla testitapaukset ladataan JSON-tiedostoista testeihin. Muutin hieman testidatan rakennetta, jotta sen hallinnointi on helpompaa jatkossa.

Keskustelimme myös toisesta testipatterista, jolla testattaisiin varsinaista kielimallin tuottamaa ulosantia. Ensimmäinen testi tuota varten olisi rakenteeltaan hyvin samanlainen kuin hakutesti, mutta testin tarkoitus olisi selvittää vastaako kielimalli olettamallamme sisällöllä.

Jatkossa haluaisimme myös lisätä testejä, joilla pystyisimme testaamaan kielimallin tuottamien ”hallusinaatioiden” määrää. Benji Edwardsin Ars Technica -verkkojulkaisussa julkaistussa artikkelissa ”Why ChatGPT and Bing Chat are so good at making things up” kerrotaan, että kielimallien yhteydessä puhuttaessa ”hallusinaatioista” tarkoitettavan tilanteita, joissa kielimallin tuottama teksti sisältää paikkansapitämättömiä väitteitä. Artikkelin mukaan kielimallien on havaittu keksivän muun muassa olemattomia kirjojen nimiä, tieteellisiä tutkimuksia sekä lakitekstiä.

15.12.2023

Työkaverini viimeisteli ensimmäisen testipatterin. Hän kirjoitti myös koodia, joka tuottaa testeistä helposti luettavan HTML-muotoisen raportin. Nämä kirjoittamamme testit ovat siinä mielessä erityislaatuisia, että koska testaamme kielimalleja, joiden ulosanti on jossain määrin arvaamatonta, emme voi olettaa jokaisen testitapauksen suoriutuvan täydellisesti joka toistolla. Tämän vuoksi joudumme määrittelemään jonkin raja-arvon, jonka testien pisteytyksen täytyy ylittää, jotta testi katsotaan läpäistyksi. Testitapaukset siis toistetaan aina useaan kertaan ja näistä toistoista lasketaan keskiarvo. Siinä mielessä nämä testit ovat hyvin erilaisia verrattuna esimerkiksi perinteisiin yksikkötesteihin, joilla todennetaan binäärisesti, onko testi läpäisty vai ei.

Keskustelimme myös testitapausten määrittelystä, joka on näissä testeissä kenties se vaikein osuus. Puhuimme alustavasti, että hyödyntäisimme testitapausten keräämiseen sovelluksemme käyttöliittymää ja sen palautteenanto ominaisuutta. Testitapausten kerääminen vaatii omaa logiikkaansa, jota työstämme ensi viikolla. Sovimme myös tapaamisen ohjelmistotestaajiemme kanssa, jotka suurilta osin tulevat määrittelemään testitapaukset näille testeille.

3.8.1 Viikkoreflektointi 11.12.2023 – 15.12.2023

Aloimme työstämään testityökalun koodia. Tässä kohtaa huomaa hyödyt, siitä että olemme purkaneet riippuvuuksia koodista aiempina viikkoina. Koodista on nyt mahdollista eristää osia ja testata vain yksittäisiä loogisia kokonaisuuksia kerrallaan.

Itse työkalun kirjoittaminen on ollut hauskaa. Uskon, että tämä lähestymistapa tulee poikimaan hyviä tuloksia, kun jatkamme varsinaisten sovellusominaisuuksien kehittämistä joululomien jälkeen. Kielimallit voivat olla arvaamattomia ulosannissaan ja ennen tätä en oikein hahmottanut miten niitä voisi systemaattisesti testata. Nämä suhteellisen yksinkertaiset testit tulevat auttamaan meitä myös, kun sovellukselle tulee uusia vaatimuksia. Voimme rakentaa uusia testipattereita tämän pohjan päälle. Mielenkiintoinen lähestymiskulma kielimallin tuottamien vastausten laadulliseen arviointiin olisi vastausten arvioiminen käyttämällä kielimallia itseään taikka toista kielimallia. Tästä aihepiiristä täytyy vielä lukea / opiskella lisää.

Ensi viikolla pitäisi kirjoittaa vielä toinen testipatteri, joka testaa botin lopullista vastausta samaan tyyliin kuin tällä viikolla kirjoitetuissa testeissä testataan hakua.

Viikko vahvisti teknistä osaamistani erityisesti kielimallipohjaisten sovellusten testaamiseen ja yleisesti ohjelmistotestaamiseen liittyen.

3.9 Viikko 9

18.12.2023

Aloitin kirjoittamaan testiä, joka testaa botin tuottamia vastauksia. Käytin pohjana viime viikolla kirjoittamaani testiä, mutta jouduin tekemään kuitenkin merkittäviä muutoksia koodiin. Minulta kului aika lailla koko päivä tämän testin kirjoittamiseen.

19.12.2023

Työkaverini jatkoi vielä eilen siitä mihin jäin ja kirjoitti osuuden, jossa testistä tuotetaan raportti HTML-muodossa. Testilogiikka itsessään on käytännössä valmista.

Jatkoin kirjoittamalla koodia, joka automaattisesti luo testitapauksia testidatasta. Käytännössä tätä testipatteria varten määrittelemme sisällöt, jotka oletamme hausta jo löytyneen ja syötämme määrittelemämme sisällöt kielimallille kontekstiksi, josta malli poimii suosituksia käyttäjälle.

20.12.2023

Päivä oli täynnä erilaisia kokouksia. Muun muassa esittelimme testityökaluamme projektiryhmälle ja keskustelimme mahdollisista uusista käyttökohteista/vaatimuksista sovelluksellemme.

21.12.2023

Pidimme ohjelmistotestaajiemme kanssa työpajan, jossa esittelimme heille millä tavoin olemme toteuttaneet testit ja sovimme miten keräämme käyttöliittymän kautta testitapauksia. Sovimme, että ohjelmistokehitystiimin jäsenet voivat luoda näitä testitapauksia ja niitä luodaan antamalla palautetta erityisellä syntaksilla palautteen tekstikentässä.

Kirjoitin koodia, joka luo testitapauksia automaattisesti perustuen käyttöliittymän kautta kerättyihin palautteisiin. Käytännössä testitapaukset parsitaan palautteista, joissa on käytetty erityisiä avainsanoja, jonka jälkeen on syötetty sisältöjä, joita testitapauksen luoja olettaa saavansa vastauksena testitapaukseen.

22.12.2023

Siistin ja korjailin aiemmin kirjoittamaani koodia ja lisäsin testitapausten parsimiseen logiikkaa, joka ottaa huomioon vielä yhden erityistapauksen, josta sovimme ohjelmistotestaajien kanssa.

3.9.1 Viikkoreflektointi 18.12.-22.12.2023

Itse testityökalu on pitkälti valmis tämän viikon puurtamisen jälkeen. Testitapausten keräämisessä päästään alkuun joululomien jälkeen. Siihen keskittyy pääosin ohjelmistotestaajamme, mutta myös minä ja työkaverini osallistumme tähän.

Kaiken kaikkiaan viime viikkojen työskentely on ollut avartavaa ohjelmistotestaamisen ja erityisesti generatiivisen tekoälyn testaamisen osalta. Työmme mahdollistaa sovelluksen kehittämisen ketterämmin jatkossa.

Viime ajat töissä ovat olleet melko intensiivisiä uuden oppimisen suhteen. Se on ehdottomasti tämän työn suola, mutta samaan aikaan huomaan, että loma tulee varmasti tarpeeseen. Alkuvuodesta jatketaan taas samaan tahtiin ja toivottavasti pääsemme pian hyödyntämään kehittämäämme testityökalua myös sovelluksen ydinominaisuuksien kehittämisessä.

Generatiivinen tekoäly on ottanut viimeisen vuoden aikana hirmuisia loikkia eteenpäin ja vaikka osa siihen liittyvästä hehkutuksesta on varmastikin liiallista, uskon sen muuttamaan teknologia-maailmaa merkittävästi.

McKinseyn raportissa Chui ja hänen kollegansa arvioivat, että suurin osa generatiivisen tekoälyn vaikutuksista talouteen tulisi asiakaspalvelu-, myynti-, markkinointi- ja ohjelmointityön muutosten kautta. Lisäksi arvioidaan, että generatiivinen tekoäly voisi parantaa merkittävästi tiedon kulkemista organisaatiossa. Vuonna 2012 McKinsey arvioi tietotyöläisten käyttävän jopa 20 prosenttia työajastaan tiedon etsimiseen ja keräämiseen, jossa generatiivisen tekoälyn nähdään mahdollistavan huomattavia tehostuksia. Raportissa esitellään erilaisia potentiaalisia käyttötapauksia generatiiviselle tekoälylle; älykkäämpiä asiakaspalvelubotteja, tehokkaampaa data-analytiikkaa strukturoimattomasta datasta, personoituja palveluita, nopeampaa ja tehokkaampaa sisällönluontia, tekoälyohjelmointiavustajia ja niin edelleen. (Chui ja kollegat 2023, McKinsey & Company, s. 12–21)

Näillä työkaluilla vaikuttaakin siis olevan potentiaalia alentaa ihmisten ja tietokoneiden välistä kitkaa huomattavasti. On erittäin mielenkiintoista työskennellä näiden teknologioiden parissa myös tulevaisuudessa! Tämä viikko kehitti erityisesti teknistä osaamistani liittyen kielimalleihin.

3.10 Viikko 10

8.1.2024

Palasin joululomilta töihin. Työpäivä sujui pääosin orientoituessa uudestaan ohjelmoinnin ja tämän kyseisen projektin syövereihin. Keskustelimme työkaverin kanssa, siitä mitä hän oli tehnyt viime viikolla. Hän oli jatkanut vielä viimeistelyjä testityökalun suhteen. Puhuimme myös, että minä alan tekemään alustavia testitapauksia, jotta pääsemme eteenpäin työkalun kanssa.

9.1.2024

Tämän päivän käytin pääosin testitapausten kasaamiseen. Käytännössä tämä tapahtui käymällä keskustelua chattibotin kanssa ja merkkaamalla haluttuja keskusteluja testitapauksiksi antamalla palautetta ja lisäämällä palautteeseen erikoissana merkiksi, että palautteen on tarkoitus olla testitapaus. Loin näitä palautteita testiympäristössämme. Lisäksi päivään sisältyi kaksi palaveria.

10.1.2024

Keskustelimme alustavasti generatiivisen tekoälyn hyödyntämisestä myös muissa konteksteissa, kuten asiakaspalvelussa. Periaatteessa tämä teknologia mahdollistaa huomattavasti älykkäämpiä asiakaspalvelubotteja, kuin tähän mennessä on ollut mahdollista toteuttaa.

Lisäksi päivitimme odotuksia tekoälyavustajaprojektimme julkaisuaikatauluista. Tällä hetkellä tähtäämme julkaisuun helmikuun lopulla tai maaliskuun alkupuolella.

Tämän jälkeen käytin jonkin verran aikaa testitapausten luomiseen, näitä alkaa olla kasassa toista kymmentä.

11.1.2024

Tänään päivitin Python-sovelluksemme käyttämään OpenAI:n uusinta Python-kirjastoa rajapinnoilleen, joka on ensimmäinen major-versio kirjastosta eli v1.0. Tein hieman refaktorointia ja koodin siistimistä samalla.

Tämän jälkeen ajoin skriptin, jolla keräsin tuottamani testitapaukset JSON-muotoon. Skriptissä oli vielä joitain bugeja, jotka olivat kuitenkin nopea korjata. Ensimmäisen testisetin on tarkoitus toimia jonkinlaisena ”regressiotestinä”. Sen tapaukset ovat lähtökohtaisesti ns. ”happy path” tapauksia ja niitä on tarkoitus suorittaa, kun teemme muutoksia esimerkiksi hakusanojen generointiin, jotta saamme jonkinlaista varmuutta, että sovelluksemme laatu ei ainakaan heikkene. Testitapaukset versioidaan koodipohjan mukana.

12.1.2024

Työkaverini näytti miten suunnittelemamme testin saa konfiguroitua siten, että sen saa ajettua Bitbucket Pipelines-ympäristössä. Tämän jälkeen pääsimme ajamaan testejä ensimmäisen kerran. Testin ajaminen on melko hidasta, sillä OpenAI:n rajapinnoilla on käyttörajoituksia, jotka täytyy ottaa testiä ajaessa huomioon. Testi valmistui noin 5 minuutissa. Sen tulokset olivat erittäin hyvät, sillä, kuten aiemmin mainitsin, kyseessä oli ns. ”happy path” testitapauksia. Ajamme samat testit uudelleen jatkossa tehtyämme muutoksia koodipohjaan tai päivitettyämme kielimallia.

Tämän jälkeen aloimme suunnittelemaan työkaverini kanssa siirtymistä käyttämään OpenAI:n ”Function calling” rajapintaa.

3.10.1 Viikkoreflektointi 8.1.–12.8.2024

Paljon aikaa tästä viikosta kului testitapausten kasaamiseen. Rehellisesti sanottuna tuo tehtävä oli melko tylsää ja hidasta. Oli kuitenkin hyvä nähdä testisetti toiminnassa ja testityökalumme pyörivän viikon lopulla.

On mukava huomata, että pitkän projektin jonkinlainen virstanpylväs, tuotantoon julkaiseminen, häämöttää jo horisontissa. Projektissa on vielä tekemistä, mutta aikataulu tuntuu kuitenkin kohtuulliselta.

3.11 Viikko 11

15.1.2024

Tänään aloimme jo työstämään ”Function calling” rajapinnan käyttöönottoa. Ideana on antaa kielimallille mahdollisuus kutsua funktioita. GPT mallille annetaan syötteessä tiettyyn JSON-datamalliin muotoiltuja funktioita, joista kielimalli voi valita mitä funktiota kutsuu. Oma sovelluksemme suorittaa varsinaisen funktion suorittamisen ja palauttaa seuraavaan syötteeseen GPT-mallille suoritetun funktion tulokset. (OpenAI Inc.)

Meidän tapauksessamme vaihtoehdot kielimallille ovat ”vastaa suoraan” tai ”suorita haku ja vastaa”. Jos käyttäjä esimerkiksi aloittaa keskustelun lähettämällä viestin ”Hei!”, silloin todennäköisesti on turha suorittaa sisältöhakua, vaan kielimalli voi päättää vastaavansa suoraan käyttäjälle ilman hakukontekstia.

16.1.2024

Jatkoin vielä eilisen muutoksen parissa. Työstimme ominaisuutta yhdessä työkaverini kanssa ja saimmekin sen melko hyvälle mallille. Saimme koodin valmiiksi ja siirsimme sen testiympäristöön, jossa ajoimme myös kasaamamme regressiotestit.

Tämän jälkeen aloin työstämään ”rate limiting” ominaisuutta. Koska kielimallin käyttämiseen liittyy juoksevia kuluja, haluamme estää väärinkäytön mahdollisuudet, joten päätimme toteuttaa viestien lähettämiseen rajan siihen, kuinka monta viestiä käyttäjä saa lähettää minuutissa.

17.1.2024

Jatkoin vielä viestirajoitusominaisuuden parissa. Sain sen valmiiksi alkuiltapäivästä ja lähetin sen työkaverilleni arvioitavaksi.

18.1.-19.1.2024

Olin sairaana.

3.11.1 Viikkoreflektointi 15.1.-17.1.2024

Kielimallien tarjoamat mahdollisuudet ovat varsin mielenkiintoisia. Malleille voi antaa mahdollisuuden kutsua funktioita eli ne voivat tavallaan tehdä päätöksiä. Korkean riskin päätösten antaminen kielimallien käsiin vaikuttaa kuitenkin rehellisesti sanottuna vielä liian riskialttiilta. Kun seuraa näihin teknologioihin liittyvää uutisointia, vaikuttaa siltä, että monet ovat antamassa näiden mallien käsiteltäväksi hyvinkin hankalia päätöksiä. Itse en rehellisesti sanottuna usko, että tämä teknologia on vielä siinä pisteessä, että sillä pystyisi automatisoimaan esimerkiksi monimutkaista asiantuntijatyötä.

Siitä huolimatta kielimallit ovat monessa paikassa hyödyllistä teknologiaa, ja niillä on paikka monessakin sovellutuksessa. On hienoa olla rakentamassa käytännön sovellusta näiden teknologioiden ympärille. Tämä viikko kehitti osaamistani kielimalliteknologioissa ja niiden soveltamisessa käytännön ongelmiin.

3.12 Viikko 12

22.1.2024

Työstin pientä bugia yksikseni ja myöhemmin työkaverini keskustelimme, miten käytämme OpenAI:n Function calling -rajapintaa. Totesimme, että voisimme pudottaa yhden kutsun OpenAI:ta päin pois, muotoilemalla Function call -kutsu toisella tavalla. Aikaa kului OpenAI -dokumentaation tutkimisessa.

23.1.2024

Aamupäivästä työstin Function call -kutsun uudelleenmuotoilua.

Työkaverini kertoi minulle ajatuksistaan liittyen Temporal Workfloweihin, joita käytämme rakentamaan ElasticSearch-indeksin. Hän haluaisi uudelleen järjestellä ja kirjoittaa koodia, jotta se olisi suorituskykyisempää, jonka lisäksi hän haluaisi lisätä koodiin riippuvuuksien injektoimista (engl. ”dependency injection”), jotta kyseiset koodin osat olisivat testattavissa.

24.1.2024

Työkaverini katselmoi koodiani, johon hän jätti pari pientä kommenttia. Korjasin nuo hänen pyytämät asiat. Tämän lisäksi päivän aikana oli palavereita, ja iltapäivällä aloimme pariohjelmoimalla työstämään eilen keskustelemiamme muutoksia.

Pariohjelmointi on tekniikka, jossa kaksi henkilöä työstää koodia samanaikaisesti. Toinen henkilöistä on ”kuski” ja toinen on ”suunnistaja”. ”Kuski” kirjoittaa koodia ja ”suunnistaja” katsoo koodin

yleiskuvaa ja antaa ohjeita kuskille. Pariohjelmoinnilla voidaan saavuttaa tavoitteita, kuten tehokkaampi työskentely tai teknisten taitojen kehittäminen. (Bolboaca, A. 2021 luku 2, alaluvut" Defining pair programming" ja "Situations when pair programming can help")

Saimme edistettyä ominaisuutta melko pitkälle tätä tekniikkaa hyödyntäen.

25.1.2024

Työkaverini oli vielä lopetettuani eilen, työstänyt muutoksia liittyen koodin optimoimiseen ja riippuvuuksien injektointiin. Katselmoin tuota koodia. Jouduin kyselemään työkaveriltani muutamasta kohdasta hieman tarkennuksia, koska en aivan ymmärtänyt joitain asioita liittyen riippuvuuksien hallintaan.

Työn alle tuon osalta jäi varsinaisten funktionaalisten testien laatiminen noille ohjelmakoodin osille.

26.1.2024

Työpaikallani oli virkistyspäivä.

3.12.1 Viikkoreflektointi 22.1.-26.1.2024

Yhteistyö kokeneemman työkaverini kanssa on sujunut hyvin ja hän vaikuttaa mielellään auttavan minua, vaikka saatan välillä kysellä "tyhmiä". Pariohjelmointi on mielenkiintoisen oloinen tekniikka, jolla on varmasti paikkansa hyödyllisenä työkaluna tietyissä tilanteissa.

Bolboacan kirjassa "Practical Remote Pair Programming" avataan, että pariohjelmoinnilla voidaan saavuttaa tehokkaampaa työskentelyä työn limittämisen kautta. Esimerkiksi testaaja ja ohjelmistokehittäjä voivat pariohjelmoimalla saavuttaa päämääränsä nopeammin kuin eristämällä koodaamisen ja testaamisen eri työvaiheisiin. Esimerkkinä kirjassa annetaan myös tilanne, jossa uusi ohjelmistokehittäjä liittyy projektiin, jolloin pariohjelmoinnista voi olla paljon apua tiedon siirtämisessä henkilöltä toiselle.

Käytännössä pariohjelmointia täytyy kuitenkin harjoitella ja se ei varmasti sovi kaikille luonnetyypeille. Itse huomasin alkavani hermoilemaan ja että minulla oli vaikeuksia keskittyä työkaverini puheeseen, kun olin "kuskin" paikalla.

Viikko kehitti teknisiä taitojani sekä vuorovaikutustaitojani etätyössä.

3.13 Viikko 13

29.1.2024

Työstin funktionaalisia testejä Temporal Workfloweille. Tarvitsin tähän työkaverini apua, koska en ihan hahmottanut, mitä osia funktionaalisessa testissä voi jättää testien ulkopuolelle. Funktionaalisissa testeissä testataan, jotain yksittäistä toiminnallisuutta kokonaisuudessaan, mutta se mikä rajaa toiminnallisuuden on joskus hanka määritellä.

30.1.2024

Tänään korjasin ongelman aiemmin kirjoittamamme testityökalun kanssa. Käytännössä testityökalu epäonnistui välillä rakentamaan tarvitsemansa ElasticSearch indeksin, koska ElasticSearch Docker-kontti ei ollut vielä valmis vastaanottamaan kutsuja.

Sen lisäksi työkaverini ehdotti muutoksia siihen, miten mallipohjat kielimallille annettaville syötteille tuotaisiin koodiin. Jälleen homman ydin oli siinä, kuinka mallipohjien käyttöä olisi helpompi testata muutosten jälkeen.

31.1.2024

Työstin tuota muutosta, josta keskustelimme työkaverini kanssa. Tähän sain kulumaan merkittävän osan päivästäni. Loppupäivästä työkaverini katselmoi koodia ja pyysi vielä joitain muutoksia, esimerkiksi lisäämään, joitain testejä uudelle luokalle ja sen metodeille.

1.2.2024

Viimeistelin eilen aloittamani muutokset ja kirjoitin työkaverini pyytämät testit. Tämän jälkeen huomasin, että muutokset rikkoivat kielimallin testaamiseen tarkoitetun työkalun, joten jouduin korjaamaan joitain osia siitä koodista. Kun projektit kasvavat, alkaa huomaamaan miten pienetkin muutokset voivat vaativa muutoksia hyvin monessa paikassa koodia.

2.2.2024

Keskustelimme työkaverini kanssa rakentamastamme työkalusta kielimallin testaamiseen. Jo aiemmin olimme miettineet tarvitsevamme työkalua, jolla voisi havaita kielimallien tuottamia ”hallusinaatioita”. Työkaverini oli törmännyt aiemmin Ragas-nimiseen työkaluun, joka voisi olla hyötyä kyseisen ongelman kanssa.

Ragas on avoimen lähdekoodin projekti, joka on keskittynyt luomaan työkaluja, joilla voi arvioida automaattisesti RAG-tyyppisten (Retrieval Augmented Generation) kielimalleihin perustuvien sovellusten tuottamia vastauksia. Ragas tarjoaa erilaisia metriikoita, joilla arvioida kielimallin suoriutumista, kuten ”Faithfulness” -niminen metriikka arvioi kielimallin vastauksen todenperäisyyttä suhteessa kielimallille annettuun kontekstiin. (Exploding Gradients)

Lisäksi työkaverini halusi lisätä syötteitä kasaavaan koodiin enemmän testejä, jotka vahvistavat, että syötteet rakentuvat oletetulla tavalla.

3.13.1 Viikkoreflektointi 29.1. – 2.2.2024

Viikon aikana opin jälleen lisää testaamisesta ja muista teknisistä teemoista. Funktionaalisten testien kirjoittaminen oli hieman haastavaa, mutta olen jo nähnyt niiden hyödyn, koska työkaverini on kirjoittanut niitä aiempiin osiin koodipohjaa. Käytännössä niiden avulla voi välttää manuaalista testaamistyötä muutosten jälkeen.

Oli mielenkiintoista myös lukea Ragas-projektista, joka yrittää ratkaista juuri niitä ongelmia, joiden kanssa olemme painineet. Tuon projektin tavoitteena on tarjota työkaluja testidatan generointiin, metriikoita kielimallien vastausten laadun objektiiviseen arvioimiseen sekä työkaluja kielimallipohjaisten sovellusten monitorointiin (Exploding Gradients). Tulemme varmasti lähitulevaisuudessa tutustumaan näihin Ragas-projektin käyttämiin tekniikoihin tarkemmin.

Viikon aikana opin paljon teknisistä teemoista kuten kielimallien testaamisesta ja perinteisestä ohjelmistotestaamisesta.

4 Pohdinta

Päiväkirjan pitämisen aikana työt ovat keskittyneet erityisesti tekoälykielimalliin pohjautuvan sovelluksen kirjoittamiseen. Työtehtävät ovat keskittyneet sovelluksen arkkitehtuurin suunnitteluun ja ohjelmointitöihin. Erityistä huomiota viikkojen aikana sai perinteiset automaattiset ohjelmistotestaamisen teemat, kuten yksikkötestit ja funktionaaliset testit sekä kielimallien testaamiseen liittyvät haasteet ja tekniikat, joita haasteiden ratkaisemiseen voi soveltaa. Lisäksi opin paljon tuotekehittämisen eri vaiheista ja monialaisen osaamisen tärkeydestä tuotekehitystiimissä. Tiivis yhteistyö kokeneemman ohjelmistokehittäjän sekä muiden tiimimme jäsenten kanssa opetti paljon myös vuorovaikuttamisesta työympäristössä, koodikatselmoinneista ja kritiikin vastaanottamisesta sekä sosiaalisten taitojen keskeisyydestä ohjelmistokehitystyössä.

4.1 Teknologiaopit

Perinteiset automaattiset testit, kuten yksikkötestit pyrkivät vähentämään regressioiden määrää alati muuttuvissa koodipohjissa. Regressioksi kutsutaan tilannetta, jossa jokin toiminnallisuus lakkaa toimimasta tarkoitetulla tavalla, jonkin tapahtuman, yleensä koodimuutoksen, jälkeen. Hyvien testien laatiminen vaatii alkuun vaivannäköä, mutta mahdollistavat pitkässä juoksussa ohjelmistoprojektin kestävän kehittymisen. (Khorikov, V. 2020, luku 1.2.)

Opin päiväkirjan pitämisen aikana paljon perinteisestä ohjelmistotestaamisesta työkaveriltani, joka oli erityisen perehtynyt automaattisen testaamisen tekniikoihin. Testattavan koodin arvo ei ole pelkästään sen testattavuudessa vaan tämänkaltainen koodi on usein myös luettavampaa ja helpommin lähestyttävämpää myös muille ohjelmistokehittäjille. Näistä opeista pystyn varmasti ammentamaan myös tulevaisuudessa ohjelmistokehittäjänä.

Tekoälykielimalleihin nojaavat sovellukset ovat tuoneet testaamiseen uudenlaisia haasteita. Kielimallit voivat olla ongelmallisia tuottaen esimerkiksi paikkaansa pitämättömiä tai haitallisia väitteitä (Guo, Z. ja kollegat 2023).

Tämänkaltaisten ongelmien ratkaisemiseen ei ole vielä kovin paljon tietoa vielä saatavilla, sillä tekoälymallien ympärille ei ole vielä rakennettu sovelluksia kovin pitkään. Pitäessäni päiväkirjaa, kehitimme testipatteria kielimallin ulosannin testaamiseen. Jo tuo testipatteri oli hyödyllinen tarkoituksiimme, mutta identifioimme vielä tiettyjä käyttötapauksia, joissa toisenlaiset testit voisivat hyödyttää meitä. Tulevina kuukausina tulemme vielä perehtymään enemmän kielimallien testaamiseen.

Testaamisen lisäksi opin päiväkirjan pidon aikana paljon generatiivisesta tekoälystä, joka on tullut ryminällä it-maailmaan viimeisten vuosien aikana. Tämä teknologia mahdollistaa uudenlaisia sovel-

luksia ja tulee varmasti tehostamaan ja jopa automatisoimaan monia töitä. OpenAI ja muut yritykset, jotka kehittävät näitä teknologioita, tarjoavat yhä kehittäjäystävällisempiä rajapintoja uusimpien ja hienoimpien kielimallien käyttämiseen.

Lisäksi kerrytin osaamista moniin webkehityksen teknologioihin, kuten ElasticSearchiin, Dockeriin ja FastAPIin liittyen. Kokonaiskuvani tuotantokelpoisen sovelluksen vaatimuksista parani huomattavasti.

4.2 Vuorovaikuttamisen ja projektiosaamisen opit

Päiväkirjan pitämisen aikana opin hahmottamaan paremmin tuotekehitysprojektien elinkaarta ja keskeisiä yhteistyön mekanismeja. Kasvotusten tapahtuneet työpajat eri toimintojen välillä olivat erittäin hyödyllisiä, sillä niiden jälkeen kaikilla projektiin osallistuvilla oli kokonaisvaltaisempi kuva siitä mitä tavoittelemme. Tämä on omiaan tehostamaan työtä sekä motivoimaan työntekijöitä. Ohjelmistokehittäjän roolissa keskeistä on osata kommunikoida tekniseen toteutukseen liittyviä tekijöitä ymmärrettävällä tavalla muille sidosryhmille. Esimerkiksi joskus teknologiavalinnat voivat rajoittaa käyttötapauksia tai jonkin ominaisuuden kehittäminen on todella työlästä. Joskus taas teknologia voi mahdollistaa täysin uudenlaisia ratkaisuja, joista on vaikea tietää ilman syvää teknologiaosaamista.

Etänä tehtävässä työssä keskeisiä vuorovaikuttamisen kanavia ovat esimerkiksi Teams-puhelut ja viestiminen. Selkeä ja hyvin muotoiltu kirjoitusasu on viesteissä mielestäni tärkeää, erityisesti jos viesti on tarkoitettu suuremmalle yleisölle. Etätyöskentelyssä ei tule myöskään vältellä ihmisten lähestymistä suorilla puheluilla tai viesteillä tarpeen vaatiessa. Ohjelmistokehittäjän työssä on tiettyjä työskentelytekniikoita, joita voi hyödyntää myös etänä. Esimerkiksi pariohjelmoiminen on tekniikka, joka voi auttaa uusia tiimin jäseniä pääsemään sisään koodipohjiin nopeammin. Se voi myös yleisesti tehostaa työskentelyä, koska usein parina työskennellessä voidaan välttää virheitä, joita syntyisi, jos vain yksi henkilö työskentelisi koodin parissa. Tätä tekniikka hyödyntääkseen etänä tulee kuitenkin löytää sopivat työkalut, jotta teknologia tulisi mahdollisimman vähän esteeksi. Kaikille suurille koodieditoreille löytyy kuitenkin pariohjelmointiin soveltuvat työkalut.

5 Viitteet

Adrian, B. 2021. Practical Remote Pair Programming. Packt Publishing. E-kirja. Luettu: 26.9.2024.

Alammar, J. & Grootendorst, M. 2024. Hands-On Large Language Models. O'Reilly Media. E-kirja. Luettu: 19.9.2024.

Brechner, E. 2015. Agile Project Management with Kanban. Microsoft Press. E-kirja. Luettu: 1.10.2024.

Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., Zemmel, R. 2023. The economic potential of generative AI. McKinsey & Company. Luettavissa: https://www.mckinsey.com/~/media/mckinsey/business%20functions/mckinsey%20digital/our%20insights/the%20economic%20potential%20of%20generative%20ai%20the%20next%20productivity%20frontier/the-economic-potential-of-generative-ai-the-next-productivity-frontier.pdf. Luettu: 1.10.2024.

Docker -dokumentaatio. What is Docker?. Docker Inc. Luettavissa: https://docs.docker.com/get-started/docker-overview/. Luettu 12.9.2024.

Edwards, B. 2023. Why ChatGPT and Bing Chat are so good at making things up. Ars Technica. Luettavissa: https://arstechnica.com/information-technology/2023/04/why-ai-chatbots-are-the-ultimate-bs-machines-and-how-people-hope-to-fix-them/. Luettu: 13.9.2024.

ElasticSearch -dokumentaatio. K-nearest neigbor search. ElasticSearch B.V. Luettavissa: https://www.elastic.co/guide/en/elasticsearch/reference/8.8/knn-search.html. Luettu: 12.9.2024.

Guo, Z., Jin, R., Liu, C., Huang, Y., Shi, D., Yu, S., Liu, Y., Li, J., Xiong, B., Xiong, D. 2023. Evaluating Large Language Models: A Comprehensive Survey. Tianjin University. Luettavissa: https://arxiv.org/pdf/2310.19736. Luettu: 4.9.2024.

Kenneth, R. 2012. Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison-Wesley Professional. E-kirja. Luettu: 25.9.2024.

Khorikov, V. 2020. Unit Testing Principles, Practices, and Patterns. Manning Publications. E-kirja. Luettu: 11.9.2024.

Konda, M. 2023. ElasticSearch in Action, Second Edition. Manning Publications. E-kirja. Luettu: 30.9.2024.

Lubanovic, B. 2023. FastAPI. O'Reilly Media, Inc. E-kirja. Luettu: 14.9.2024.

Mann, E. 2023. PHP Cookbook. O'Reilly Media, Inc. E-kirja. Luettu: 21.9.2024.

OpenAI -rajapintadokumentaatio. Function calling. OpenAI. Luettavissa: https://platform.openai.com/docs/guides/function-calling. Luettu: 1.10.2024.

OpenAI, Inc 2022. New and improved embedding model. Luettavissa: https://openai.com/index/new-and-improved-embedding-model/. Luettu: 9.10.2024.

Porebski, B., Przystalski, K., Nowak, L. 2011. Building PHP Applications with Symfony™, CakePHP, and Zend® Framework. Wrox. E-kirja. Luettu: 2.10.2024.

Ragas -dokumentaatio. Core Concepts. Exploding Gradients. Luettavissa: https://docs.ragas.io/en/stable/concepts/index.html#. Luettu: 13.9.2024.

Seemann, M., van Deursen, S. 2019. Dependency Injection Principles, Practices, and Patterns. Manning Publications. E-kirja. Luettu: 2.10.2024.

Supervisor -dokumentaatio, 2024. Introduction. Agendaless Consulting and Contributors. Luettavissa: http://supervisord.org/introduction.html. Luettu: 3.10.2024.

Temporal -dokumentaatio. Why Temporal? Temporal Technologies Inc. Luettavissa: https://docs.temporal.io/evaluate/why-temporal. Luettu 20.9.2024.

Temporal -dokumentaatio. What Is a Temporal Worker? Temporal Technologies Inc. Luettavissa: https://docs.temporal.io/workers. Luettu: 1.10.2024.

Uvicorn -dokumentaatio. Introduction. Luettavissa: https://www.uvicorn.org/. Luettu: 2.10.2024.

Vue.js -dokumentaatio. Introduction. Luettavissa: https://vuejs.org/guide/introduction.html. Luettu 30.9.2024.