Opinnäytetyö (AMK)

Tieto- ja viestintätekniikka

2024

Heikki Järvinen

Tekoälyn hyödyntäminen elokuvasuositusjärjestelmässä

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

Opinnäytetyö (AMK) | Tiivistelmä

Turun ammattikorkeakoulu

Tieto- ja viestintätekniikka

2024 | 41 sivua

Heikki Järvinen

Tekoälyn hyödyntäminen elokuvasuositusjärjestelmässä

Suositusjärjestelmät ovat keskeinen osa nykypäivän verkkopalveluita, kuten elokuvien suoratoistopalveluita ja kaupallisia verkkokauppoja. Viime vuosina tekoäly on noussut merkittäväksi tekijäksi teknologiateollisuudessa. Tekoälyä voidaan hyödyntää suositusjärjestelmissä tarjoamaan parempia suosituksia ja uusia ominaisuuksia.

Opinnäytetyön tavoitteena oli suunnitella ja kehittää tekoälyä hyödyntävä elokuvasuositusjärjestelmä. Aihe valittiin aikaisemman mielenkiinnon suositusjärjestelmiin ja tekoälyn nopean kehityksen vuoksi. Suositusjärjestelmässä käytettiin OpenAI:n ChatGPT 4o API:a tuottamaan suosituksia käyttäjän tietojen perusteella. Järjestelmää käytetään verkkosivulla, joka rakennettiin Next.js -ohjelmistokehyksellä. Verkkosivun rakentamisessa käytettiin myös muita moderneja teknologioita ja kahta tietokantaa.

Opinnäytetyön tulos oli toimiva suositusjärjestelmä sovellus. Käyttäjät saavat hyödyllisiä ja relevantteja elokuvasuosituksia sovellukselta. Jatkokehitysideana on parantaa nykyistä sivua ja lisätä ominaisuuksia. Opinnäytetyön tuloksia voi hyödyntää tekoälypohjaisten verkkosovellusten ja suoratoistopalveluiden kehityksessä.

Asiasanat:

tekoäly, suositusjärjestelmä, verkkosovellus

Bachelor’s / Master’s Thesis | Abstract

Turku University of Applied Sciences

Degree programme

2024 | 41 pages

Heikki Järvinen

Artificial intelligence in a movie recommendation system

Recommendation systems are a central part of today's online services, such as movie streaming platforms and commercial online stores. In recent years, artificial intelligence has emerged as a significant factor in the technology industry. AI can be utilized in recommendation systems to provide better suggestions and new features.

The aim of the thesis was to examine how AI can be utilized in a recommendation system and to design and develop an AI-powered movie recommendation system. The subject was chosen based on an interest in recommender systems and artificial intelligence. The recommendation system used OpenAI's ChatGPT API to generate recommendations based on user data. The system is used on a website built with the Next.js software framework. Other modern technologies and two databases were also used in building the website.

The result of the thesis was a functional recommendation system application. Users receive useful and relevant movie recommendations from the application. As a future development idea, improving the current website and adding new features is suggested. The results of the thesis can be utilized in the development of AI-based web applications and streaming services.

Keywords:

artificial intelligence, recommendation system, web application

Sisältö

Kuvat

Käytetyt lyhenteet

API	Application Programming Interface, ohjelmointirajapinta (AWS 2024)
CDN	Content Delivery Network, sisällönjakeluverkko (Next.js 2024)
CSS	Cascading Style Sheets, verkkosivujen tyylien määrittely (Tailwind CSS 2024)
LLM	Large Language Model, suuri kielimalli (OpenAI 2023)
GPT	Generative Pre-trained Transformer, yleinen suuri kielimalli. (OpenAI 2023)
SEO	Search Engine Optimization, hakukoneoptimointi (Next.js 2024)
SSG	Static site generation, staattisten sivujen luominen (Next.js 2024)

1 Johdanto

Tämä opinnäytetyö tutkii tekoälyn hyödyntämistä elokuvasuositusjärjestelmässä. Suositusjärjestelmät ovat keskeinen osa nykypäivän verkkopalveluita, kuten elokuvien suoratoistopalveluita ja kaupallisia verkkokauppoja. Näiden palveluiden käyttäjät odottavat personoituja ja relevantteja suosituksia, jotka vastaavat heidän mieltymyksiään ja kiinnostuksen kohteita.

Viime vuosina tekoälykehitys on kiihtynyt huomattavasti ja monet yritykset ovat julkaisseet tekoälysovelluksia ja chattibotteja. Monia tekoälytekniikoita on lisätty suositusjärjestelmiin parantamaan käyttäjäkokemusta ja käyttäjätyytyväisyyttä. Tekoäly mahdollistaa tarkempien ja laadukkaampien suositusten luomisen verrattuna perinteisiin järjestelmiin. (Zhang ym. 2020.)

Opinnäytetyön tavoitteena oli luoda tekoälyä hyödyntävä elokuvasuositusjärjestelmä. Suositusjärjestelmä toimii verkkosovelluksena, jossa käyttäjät voivat keskustella elokuvasuosituksia antavan tekoälyn kanssa. Käyttäjä voi lisätä itsestään tietoja, joiden avulla tekoäly voi antaa tarkempia ja parempia suosituksia.

Aihe valittiin aikaisemmasta kiinnostuksesta suositusjärjestelmiin ja tekoälyyn. Aikaisemmalla kurssilla rakennettu elokuvasuositusjärjestelmä herätti mielenkiinnon lähestyä suositusjärjestelmän rakentamista erilaisilla työkaluilla. Tekoäly sovellusten nopea viimeaikainen kehitys (Zhang ym. 2020) herätti myös kiinnostusta suurien kielimallien hyödyntämiseen sovelluskehityksessä. Työn oppimistavoite oli tutkia ja oppia tekoälyn hyödyntämistä modernissa verkkosovelluskehityksessä.

Opinnäytetyössä tutkitaan yleisiä suositusjärjestelmiä ja arvioidaan niiden sopivuutta elokuvasuositusjärjestelmään, sekä selvitetään erilaisia tekoälyjä, kuten ChatGPT:tä, ja arvioidaan ChatGPT:n soveltuvuutta suositusjärjestelmään.

2 Suositusjärjestelmät

Melville ja Sindhwanin (2010) mukaan suositusjärjestelmien tarkoitus on luoda käyttäjille merkityksellisiä ja hyödyllisiä suosituksia. Monet yritykset ja palvelut käyttävät suositusjärjestelmiä tarjotakseen asiakkaille miellyttäviä tuotteita. Esimerkiksi Youtube suosittelee videoita käyttäjilleen, perustuen käyttäjän aiemmin katsomiin videoihin tai videoita, joita samankaltaiset käyttäjät ovat katsoneet (Goodrow 2021). Suositusjärjestelmät voidaan yleisellä tasolla jakaa yhteistoiminnallisiin ja sisältöpohjaisiin suositusjärjestelmiin. Jotkin järjestelmät ovat myös niin sanottuja hybridijärjestelmiä, jotka yhdistävät yhteistoiminnallisen ja sisältöpohjaisen järjestelmän ominaisuuksia.

2.1 Yhteistoiminnallinen suositusjärjestelmä

Yhteistoiminnallinen suositusjärjestelmä (engl. collaborative filtering) hyödyntää käyttäjän antamia arvosteluja tuotteista tai asioista ja vertaa sitä muiden käyttäjien antamiin arvosteluihin. Yhteistoiminnalliset järjestelmät voidaan jakaa naapuri -pohjaisiin ja malli -pohjaisiin järjestelmiin. (Melville & Sindhwan 2010.)

Naapuri -pohjaisessa järjestelmässä valitaan pieni osa käyttäjistä perustuen niiden samanlaisuuteen aktiivisen käyttäjän kanssa. Valittujen käyttäjien painotettuja arvosteluja käytetään muodostamaan ennustuksia aktiivisen käyttäjän toiveista ja preferensseistä. Naapuri-pohjainen järjestelmä tunnetaan myös nimellä muisti -pohjainen järjestelmä. (Zhang ym. 2020.)

Malli-pohjaisessa järjestelmässä suosituksia luodaan matemaattisten ja tilastotieteellisten mallien perusteella. Mallit rakennetaan käyttäjien antamien arviointien pohjalta. (Melville & Sindhwan 2010.)

Yhteistoiminnallisen suositusjärjestelmän hyviä puolia ovat sen yksinkertaisuus. Se on helppo toteuttaa ja se tuottaa tarkkoja tuloksia. Järjestelmä ei vaadi tarkkaa tietoa tuotteista tai asioista, joita se suosittelee, eikä myöskään

käyttäjistä. Järjestelmä tarvitsee tiedon ainoastaan käyttäjien arvosteluista luodakseen suosituksia.

Järjestelmän huono puoli on ”cold-start”-ongelma. Ongelma voi syntyä uusissa järjestelmissä tai silloin, kun järjestelmään lisätään uusi kohde tai käyttäjä. Uudella kohteella ei ole dataa, jonka perusteella järjestelmä voisi antaa tarkkoja suosituksia. Tämä ongelma korjaantuu, kun järjestelmä saa lisää dataa, jonka perusteella antaa suosituksia. (Lika ym. 2014.)

2.2 Sisältöpohjainen suositusjärjestelmä

Sisältöpohjaisissa suositusjärjestelmissä käytetään kohteiden ominaisuuksia ja käyttäjän mieltymyksiä suosituksien luomiseen. Esimerkiksi elokuvia voidaan suositella tällä tavalla. Jos käyttäjä pitää Taru Sormusten Herrasta elokuvista, kyseinen käyttäjä pitää myös muista fantasiaelokuvista. Järjestelmä siis suosittelee kohteita, jotka jakavat ominaisuuksia keskenään. Kohteelle yleensä annetaan ominaisuuksia, kuten kuvaus tai metadataa, joiden perustella se voidaan yhdistää muihin kohteisiin. (Melville & Sindhwan 2010.)

Järjestelmä luo käyttäjistä profiilin, johon mallinnetaan käyttäjän mieltymykset. Profiiliin kerätään myös tietoa käyttäjän toiminnasta järjestelmässä. (Zhang ym. 2020.)

2.3 Hybridisuositusjärjestelmä

Nimensä mukaisesti hybridisuositusjärjestelmä on yhteistoiminnallisen ja sisältöpohjaisen suositusjärjestelmän yhdistelmä. Hybridisuositusjärjestelmä tarjoaa käyttäjille suosituksia tuotteista tai sisällöstä yhdistämällä useita erilaisia suositusmenetelmiä. (Burke 2014.)

Hybridisuositusjärjestelmät tarjoavat tarkempia suosituksia, koska ne yhdistävät useita erilaisia suositustekniikoita ja enemmän dataa, jonka perusteella tuottaa

suosituksia. Hybridijärjestelmät välttävät yleisiä suositusjärjestelmien ongelmia, esimerkiksi ”cold-start”, niukkuus ja skaalausongelmat, jotka ovat yleisiä yhden metodin suositusjärjestelmissä. (Burke 2014.)

2.4 Elokuvasuositusjärjestelmä

Hyvä esimerkki elokuvasuositusjärjestelmästä löytyy Netflixiltä. Netflixin suositusjärjestelmä käyttää algoritmeja ja monipuolisia datalähteitä tuottamaan käyttäjille suosituksia. Netflixillä on miljoonia käyttäjiä, jotka käyttävät päivittäin palvelua. Käyttäjät ovat yllättävän huonoja valitsemaan monien vaihtoehtojen välillä, ja he voivat joutua ylianalysoinnin loukkuun, jolloin he valitsevat "ei mikään edellä mainituista" tai tekevät huonoja valintoja. On tärkeää tarjota käyttäjille hyviä suosituksia, jotta he pysyvät asiakkaina. Tyypillinen käyttäjä menettää kiinnostuksensa 90 sekuntia kestäneen selailun jälkeen. On siis tärkeää tarjota käyttäjille kiinnostavaa sisältöä mahdollisimman nopeasti. (Gomez-Uribe & Hunt 2015.)

Netflix käyttää useita erilaisia algoritmeja parantaakseen käyttäjäkokemusta. Yhteistoiminnallinen suositusjärjestelmä luo pohjan Netflixin suositusjärjestelmälle. Malli käyttää käyttäjä-käyttäjä ja kohde-kohde yhteistoiminnallista suodatusta. Käyttäjiä yhdistetään samankaltaisten katsomistapojen mukaan. Kohteita arvioidaan niiden arvostelujen ja vuorovaikutusten perusteella. (Krysik 2024.)

Muita järjestelmiä, joita Netflix käyttää, ovat muun muassa henkilökohtainen videorankkaaja (Personalized Video Ranker: PVR), Top-N videorankkaaja, Trendaavat nyt ja Jatka katsomista. Lisäksi heillä on myös video-video-samankaltaisuus ja näyttöalgoritmi, joka valitsee ja järjestää rivit jokaiselle sivulle ottaen huomioon käyttäjän kannalta relevanssin ja sivun monimuotoisuuden. (Gomez-Uribe & Hunt 2015.)

PVR-järjestelmä järjestää koko videokatalogin tai valitut osat henkilökohtaisesti kullekin jäsenelle. Top-N videorankkaaja tuottaa parhaat muutamat henkilökohtaiset suositukset koko katalogista kullekin jäsenelle. Trendaavat nyt

hyödyntää lyhytaikaisia ajallisia trendejä, ja Jatka katsomista järjestää äskettäin katsotut videot arvioiden, haluaako käyttäjä jatkaa katsomista vai ei. Video-videosamankaltaisuus perustuu käyttäjän aiempiin katselutottumuksiin. (Gomez-Uribe & Hunt 2015.)

Netflixin hakutoiminto perustuu myös useisiin algoritmeihin, jotka yhdistävät toistotiedot, hakutiedot ja metatiedot löytääkseen relevantit videot ja näyttämään ne käyttäjille. (Gomez-Uribe & Hunt 2015.)

Yli 80 % Netflixin katselutunneista tulee suositusjärjestelmän suosittelemista elokuvista ja tv-sarjoista (Krysik 2024). Tämä osoittaa suositusjärjestelmien tärkeyden suoratoistopalveluissa ja muissa verkkopalveluissa.

2.5 Suositusjärjestelmien riskit ja haitat

Suositusjärjestelmät ja niiden käyttö tarjoavat suuria hyötyjä, mutta myös mahdollisia haittoja. Suositusjärjestelmät tekevät suosituksia datan perusteella, joka voi olla puolueellista. Järjestelmiä voidaan myös säätää suosimaan tietynlaista sisältöä. Jos järjestelmä säädetään suosimaan klikkejä tai vuorovaikutuksia, se saattaa suositella enemmän tunnettua sisältöä, kuin vähemmän tunnettua. Data voi myös olla bottien tai väärennettyjen arvostelujen manipuloimaa. Esimerkiksi videon katselumäärää voidaan korottaa boteilla, joka johtaa videon suositteluun oikeille käyttäjille. (Chatterjee 2023.)

Suositusjärjestelmät tarvitsevat toimiakseen dataa. Datan keräämiseen ja säilömiseen liittyy yksityisyysriskejä ja ongelmia. Yleistä suositusjärjestelmille hyödyllistä dataa ovat esimerkiksi selaushistoria, ostoshistoria ja henkilökohtaiset tiedot. Datan säilöminen tuo mukanaan myös riskejä. Suuret tietokannat ovat tietomurtoriskejä. Yritykset saattavat myös käyttää käyttäjien dataa tarkoituksiin, joita ei ole selvitetty käyttäjille. (Chatterjee 2023.)

Merkittävä suositusjärjestelmien aiheuttama ongelma on kuplautuminen (engl. filter bubble). Kuplautuminen syntyy, kun algoritmit priorisoivat sisältöä, jotka sopivat käyttäjän aiempaan toimintaan. Tämä voi johtaa yksipuoliseen tiedon

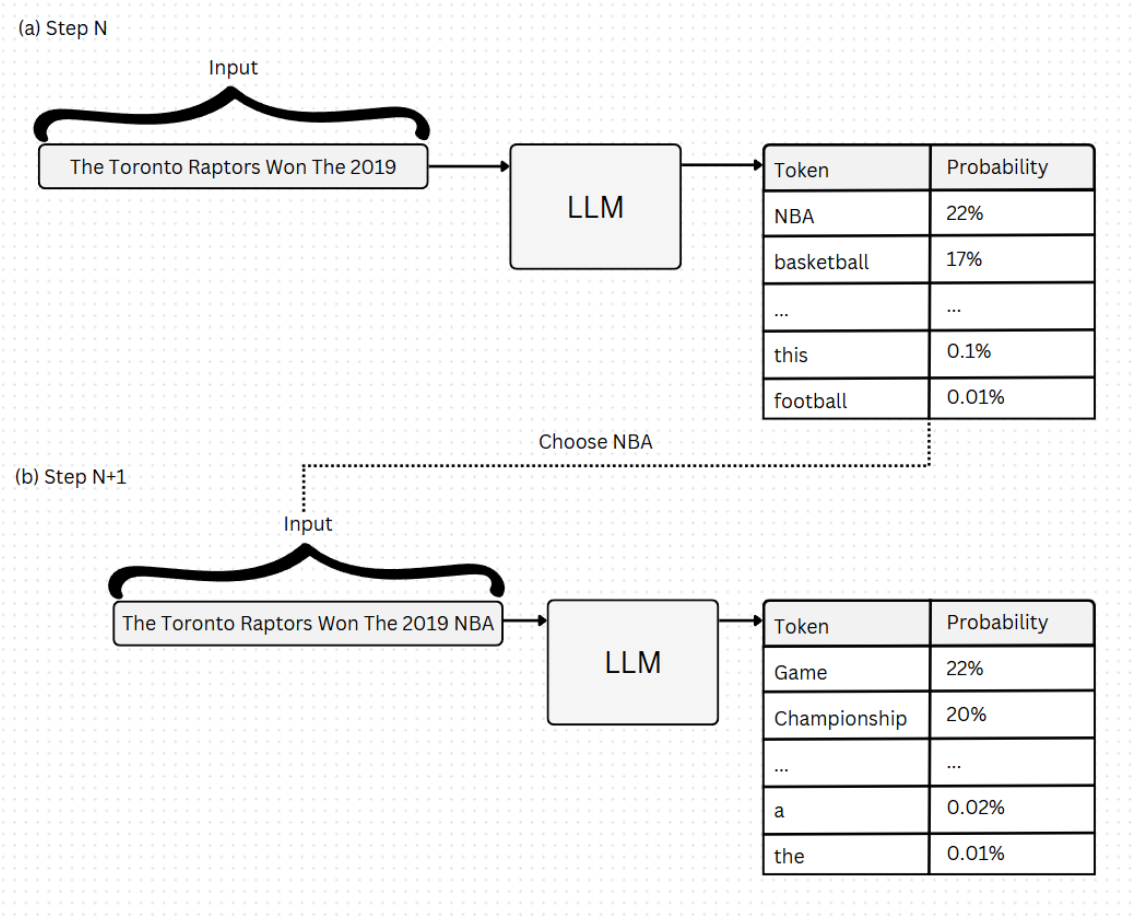
saamiseen ja monipuolisen tiedon piilottamiseen. Sosiaalisessa mediassa ja suositusjärjestelmien kautta levitetty sisältö voi näkyä vain samanlaista sisältöä katsoville käyttäjille. Käyttäjät saattavat kokea sisällön toistuvaksi, jos järjestelmä suosittelee vain samankaltaisia kohteita. (Anand ym. 2021.)

Kuplautumisen välttämiseen ja vähentämiseen on kehitetty tekniikoita. Algoritmeja säätämällä voidaan lisätä sisällön monimuotoisuutta ja tarjota vaihtoehtoisia näkökulmia. Käyttäjille voidaan antaa enemmän mahdollisuuksia vaikuttaa nähtyyn sisältöön. Käyttäjiä voidaan myös kannustaa itse tekemään hakuja ja etsiä erilaisia tietolähteitä. (Anand ym. 2021.)

3 Suuret kielimallit

GPT-mallit ovat OpenAI:n kehittämiä suuri kielimalleja (engl. Large language model (LLM)). Mallien toiminta perustuu sanojen ja lauseiden riippuvuuteen toisistaan. Kielimalli rakennetaan prosessoimaan ja ymmärtämään tekstisyötteitä ja luomaan niihin vastauksia. Näitä syötteitä kutsutaan prompteiksi (engl. prompt). Mallit koulutetaan suurella määrällä tekstiä, jota ei ole merkitty. Mallit oppivat yleisiä kielirakenteita ja yleistä tietoa. (Zi ym. 2023.)

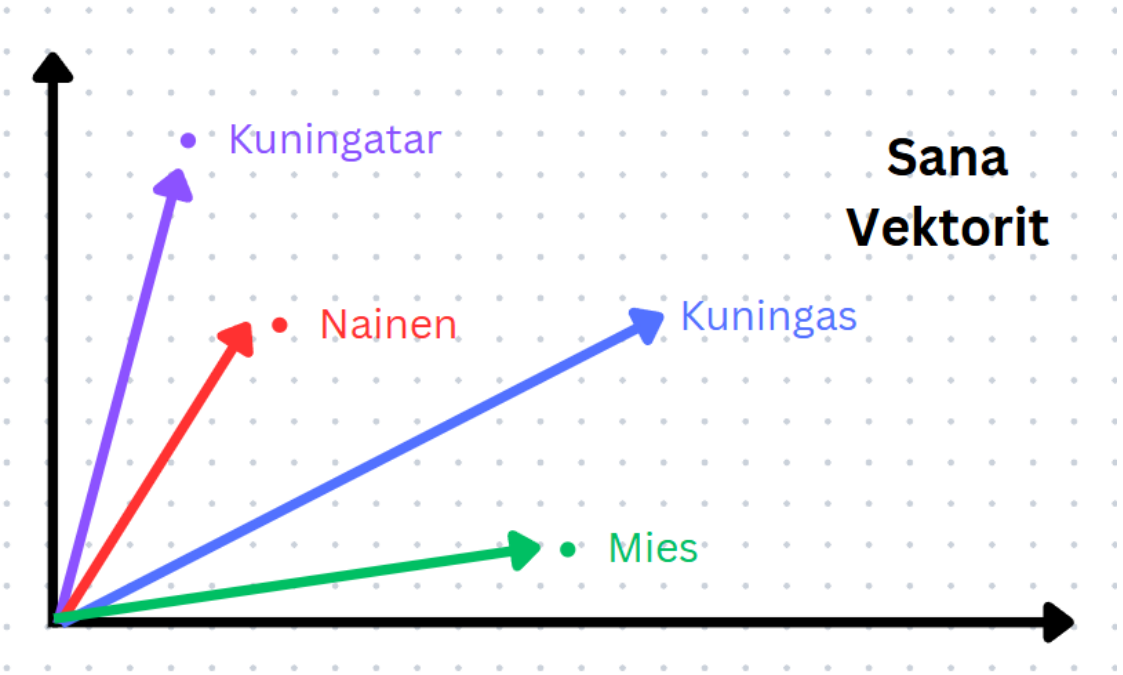
LLM:t toimivat ennustamalla seuraavaa todennäköisintä sanaa aikaisempien sanojen perusteella. Kuvassa 1 nähdään, miten kielimallit ennustavat seuraavaa sanaa. LLM saa syötteeksi lauseen ”The Toronto Raptors won the 2019” ja antaa sanoille todennäköisyyksiä. Todennäköisin sana ”NBA” valitaan ja lisätään syötteeseen. Prosessi toistetaan niin monta kertaa, että lause tulee valmiiksi. Teknisellä tasolla prosessissa tapahtuu kolme vaihetta. Ensiksi syöttö muutetaan merkkiupotuksiksi (engl. token embedding). Toisessa vaiheessa seuraava sana ennustetaan decoder-only transformerilla. Lopuksi seuraava sana valitaan todennäköisyyksien perusteella. (Zi ym. 2023.)



Kuva 1. Esimerkki suurien kielimallin tekstin luomisprosessista. (Zi ym. 2023)

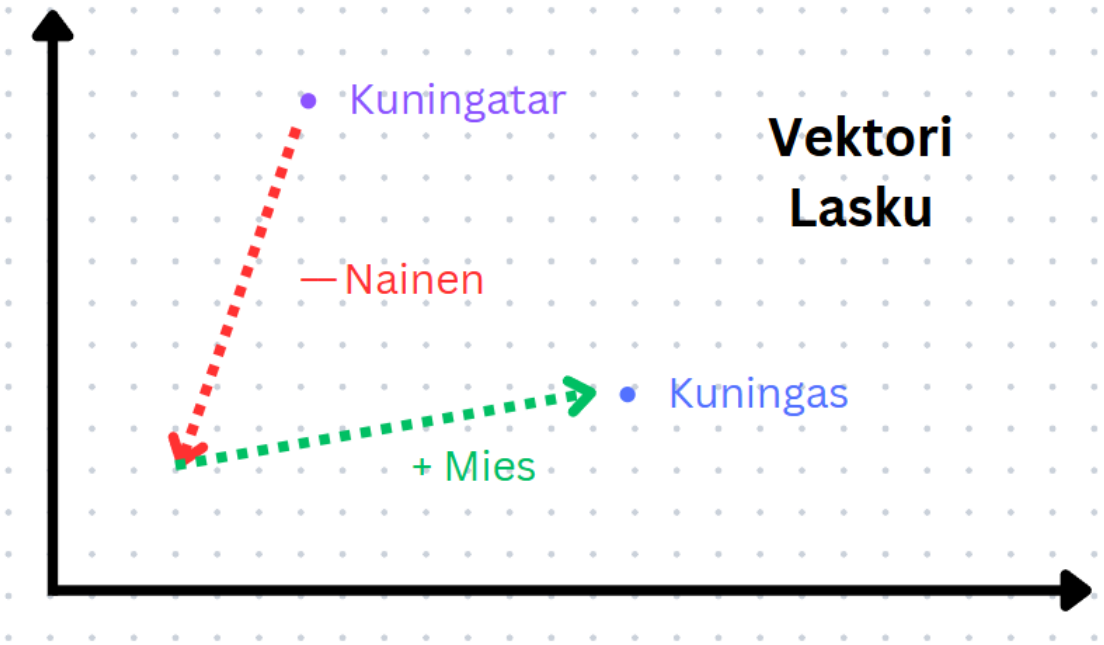
Kun LLM vastaanottaa teksti syötteen, teksti muutetaan merkeiksi (engl. token) ja syötetään mallille merkkijonona. Merkki voi sisältää sanan tai osan sanaa riippuen sanan pituudesta. ChatGPT hinnoittelu perustuu käytettyjen merkkien määrään. Malleilla on enimmäismäärä merkkejä, joita se voi käyttää. GPT-3.5-turbo mallilla tämä raja on 4097 (OpenAI 2024a). Sekä syöte että ulostulo lasketaan tähän rajaan.

Merkeille määritetään vektoreita, joiden avulla kielimallit ymmärtävät sanojen yhteyksiä toisiinsa. Otetaan tarkasteluun sanat mies, nainen, kuningas ja kuningatar. Kuvassa 2 on annettu sanoille vektorit. Kuninkaan ja miehen vektorit ovat lähellä toisiaan, kuten myös kuningattaren ja naisen.



Kuva 2. Esimerkki sanojen Kuningatar, Naine, Kuningas ja Mies vektoreista. (Colyer 2016)

Vektorien avulla voimme löytää sanoille yhteyksiä. Esimerkiksi kuvassa 3 jos vähennämme kuningattaren vektoreista naisen vektorit ja lisäämme siihen miehen vektorit, päädymme kuninkaan vektoreihin.



Kuva 3. Esimerkki vektori laskusta Kuningatar - Nainen + Mies = Kuningas. (Colyer 2016)

Sanoilla voi olla satoja tai tuhansia vektoreita kuvailemassa sitä. Vektoreista muodostuvien verkostojen avulla kielimallit muodostavat yhteyksiä ja ymmärrystä sanoista. (Metzger 2022.)

Vektoreita eli merkkiupotuksia voidaan luoda joko laskemalla ne sääntöpohjaisilla metodeilla (esimerkiksi tf-idf) tai oppimalla mallintamalla (esimerkiksi Word2vec). Merkkijonoihin lisätään vielä sanan sijainnin kertova vektori ja väliaikaiset merkit, jotka kertovat syötteen alun ja lopun. (Zi ym. 2023.)

Malli käsittelee merkkijonon usean eri transformer kerroksen läpi. Jokainen kerros keskittyy tiettyyn osaan syötettä ja yrittää löytää toisiinsa liittyviä asiayhteyksiä. ChatGPT ymmärtää keskustelun kontekstin. Se käyttää ”self-attention” mekanismia ymmärtämään koko keskustelu historian, ja osaa ottaa sen huomioon generoidessaan vastauksia. Tähän kuitenkin pätee aiemmin mainittu merkki raja, joka määrittää kuinka pitkän keskustelun malli voi muistaa. (ChatGPT 2024.)

Suurien kielimallien koulutus aloitetaan esikoulutuksella (engl. pre-training). Esikoulutuksessa mallille syötetään suuri määrä dataa ilman ohjeita. Datasetit koostuvat verkkoindeksoinnista, kirjoista ja Wikipedia sivuista. Uusimpien mallien koulutuksessa käytetyt datasetit ovat kooltaan satoja miljardeja merkkejä. (Zi, ym. 2023.)

Esikoulutuksen jälkeen GPT malli voidaan hienosäätää (fine-tune) tiettyä tarkoitusta varten. Hienosäätäminen tarkoittaa mallin kouluttamista pienemmällä koulutusmateriaalilla tiettyä käyttötarkoitusta varten. Esimerkiksi tekstigenerointi, tekstimäärittely tai kysymyksiin vastaaminen. (Zi ym. 2023.)

3.1 Suurien kielimallien rajoitukset ja riskit

Suurien kielimallien suurin rajoitus on niiden luotettavuus. Mallit eivät ole täysin luotettavia ja ne ”hallusinoivat” välillä. Hallusinaatiot tarkoittavat väärää tai keksittyä tietoa, jonka malli väittää totuudeksi. Hallusinoinnilla on monia syitä. Mallit koulutetaan datalla, joka voi sisältää väärää tietoa. Jos koulutusdatassa on väärää tai ristiriitaista tietoa, malli saattaa tuottaa sekavia viestejä. Mallien ulostulot tulee aina tarkistaa, varsinkin tärkeissä käyttötarkoituksissa. (OpenAI 2023.)

Suurien kielimallien tuottama ulostulo sisältää riskejä vaarallisista ohjeista, bugisesta koodista tai epätarkasta informaatiosta. Mallien koulutuksessa tulee ottaa huomioon riskien vähentäminen. Mallien pitää analysoida syötteitä ja arvioida voiko niihin vastata turvallisesti. (OpenAI 2023.)

Suurien kielimallien kouluttaminen on kallista. Mallien kouluttaminen vaatii satoja näytönohjaimia ja kuluttaa runsaasti sähköä. Korkeat infrastruktuuri ja rahalliset kulut tekevät kielimalleista saavuttamattomia monille yrityksille. Tämä voi luoda monopoliaseman yrityksille, joilla on resursseja kielimallien koulutukselle. (Zi ym. 2023.)

Suurien kielimalleihin liittyy yksityisyys- ja turvallisuusongelmia. Carlin ym. (2021) esitti tutkimuksen, jonka mukaan kielimalli voidaan saada paljastamaan yksityisiä tietoja koulutusdatasta. Esimerkiksi nimiä, puhelinnumeroja ja sähköposteja. Kielimallien käyttö vaatii turvallisuustoimia, jotka estävät yksityisten tietojen leviämisen. (Zi ym. 2023.)

Tekijänoikeudet ovat nousseet tärkeäksi keskustelunaiheeksi suurien kielimallien koulutuksessa. Rikkooko mallien koulutukseen käytetty data tekijänoikeuksia? Asiasta on käynnissä useita oikeudenkäyntejä, joten laillisesti tällä hetkellä asiaan ei ole selvyyttä. Toinen kysymys on, että kuka omistaa suurien kielimallin ulostulon. Suojaako tekijänoikeus kielimallien tuotoksia? Asiaan ei ole vielä saatu laillisesti selvitystä. (Quintais 2023.)

3.2 Miksi tekoäly suositusjärjestelmässä

Suositusjärjestelmät, jotka käyttävät perinteisiä numero tai tähtiarvosteluja ovat jo hyvin kehittyneitä. Ne eivät kuitenkaan pysty hyödyntämään suurta määrää teksti-informaatiota kuten metadataa, käyttäjien luomia tunnisteita tai kirjoitettuja arvosteluja. Suuret kielimallit mahdollistavat teksti-informaation hyödyntämisen suositusjärjestelmissä. Tilanteissa, joissa arvosteluja on vähän ja perinteiset mallit kohtaisivat cold-start-ongelman, suuret kielimallit voivat luoda yhteyksiä kohteiden välillä esimerkiksi kohteiden kuvausten perusteella. (Zhang ym. 2020)

Vapaasta teksti-informaatiosta on hyötyä myös, kun dataa on paljon. Suuret kielimallit voivat löytää uusia yhteyksiä datasta, joita perinteiset suositusjärjestelmät eivät löydä. Yhteydet parantavat suositusten tarkkuutta ja läpinäkyvyyttä. (Zhang ym. 2020.)

Liu, ym. (2023) tutkimus arvioi ChatGPT:n suosituskykyä. Tutkimuksen mukaan ChatGPT pärjää hyvin arvostelujen ennustamisessa, mutta huonosti kohteiden suorassa suosittelussa verrattuna muihin kehittyneisiin suositusjärjestelmiin. ChatGPT pärjäsi muita järjestelmiä paremmin arvostelujen tiivistämisessä ja luomaan selityksiä käyttäjän mieltymyksistä. On myös otettava huomioon, että tutkimuksen ChatGPT mallia ei ollut koulutettu suositusjärjestelmä käyttöä

varten, joten tulokset saattavat parantua, jos malli koulutetaan suosittelua varten. (Liu ym. 2023.)

4 Sovelluksessa käytetyt teknologiat

4.1 Next.js -ohjelmistokehys

Next.js on suosittu avoimen lähdekoodin full-stack ohjelmistokehys. Se on suunniteltu helpottamaan React-pohjaisten verkkosivujen kehittämistä. React on Metan kehittämä ohjelmistokehys, jossa käyttöliittymät koostuvat yksittäisistä komponenteista. (React 2024.)

Next.js:n tärkeimpiin ominaisuuksiin kuuluu palvelimella renderöinti. Se tarkoittaa HTML koodin luontia palvelimella ennen kuin se lähetetään käyttäjän verkkoselaimelle. Tämä mahdollistaa sivujen nopeamman lataamisen ja se parantaa hakukoneoptimointia (SEO). Next.js mahdollistaa myös staattisten sivujen luomisen (SSG). SSG:n avulla HTML-tiedostot luodaan sivun rakentamisen aikana, jonka jälkeen ne voidaan tarjota suoraan sisällönjakeluverkolta (CDN). Tämä toteutus on äärimmäisen nopea ja erityisen hyödyllinen sivuilla, joiden sisältö ei vaihdu usein. (Next.js 2024.)

Next.js käyttää tiedostoihin pohjautuvaa reititysjärjestelmää. Tämä helpottaa sivujen järjestämistä projektissa. Sivuja on helppo lisätä ja järjestää ilman monimutkaista konfiguraatiota. (Next.js 2024.)

4.2 Shadcn/UI komponentti kirjasto

Shadcn/UI on kokoelma erilaisia React komponentteja, joita voi vapaasti käyttää React ohjelmissa. Komponentit ovat saavutettavia ja valmiiksi tyyliteltyjä, mutta niitä voi myös muokata itse. Shadcn/UI komponentit tukevat myös tummaa tilaa, joka mahdollistaa monipuolisempien verkkosivujen tekemisen. Tumma tila tarkoitta kirkkaiden värien muuttamista tummempiin, joka saattaa vähentää silmien rasitusta (Cummins 2018). Shadcn/UI tukee yleisiä ohjelmistokehyksiä, kuten Tailwind CSS, jota käytetään myös tässä projektissa. (Shadcn/UI 2024.)

4.3 Tailwind CSS -ohjelmistokehys

Tailwind CSS on Cascading Style Sheets (CSS) ohjelmistokehys, joka mahdollistaa erilaisen lähestymistavan käyttöliittymän rakentamiselle. Tailwind CSS käyttää matalan tason käyttöluokkia, joilla määritellään tyylit suoraan HTML elementteihin. Käyttöluokilla voidaan määrittää yleiset muotoilut, kuten marginaalit, värit ja fontit. Esimerkiksi Tailwind CSS luokka ”m-4” lisää elementtiin ”1rem” marginaalin. (Tailwind CSS 2024.)

Tailwind CSS mahdollistaa responsiivisen muotoilun eri kokoisille näytöille. Pienille näytöille ei mahdu sama määrä informaatiota kuin suurille näytöille. Tailwind CSS -luokkien avulla voidaan määrittää, miten sisältö näytetään eri kokoisilla näytöillä. (Tailwind CSS 2024.)

Tailwind CSS:n avulla voi välttää kokonaan CSS koodin kirjoittamisen, koska lähes kaikki tarvittava muotoilu löytyy käyttöluokista. Esimerkiksi tässä projektissa en kirjoittanut ollenkaan CSS koodia. (Tailwind CSS 2024.)

4.4 MongoDB tietokanta

MongoDB on NoSQL-tietokanta, joka hyödyntää dokumenttipohjaista datamallia. Tässä mallissa data tallennetaan JSON-dokumentteihin, ja jokaisella dokumentilla voi olla oma struktuurinsa, mikä mahdollistaa erilaisten tietorakenteiden tallentamisen samaan tietokantaan. Tämä poikkeaa perinteisistä relationaalisista tietokannoista, joissa tietomalli on ennalta määritelty. MongoDB:tä käytetään usein sovelluksissa, joissa tietomallin joustavuus ja dynaamisuus ovat tärkeitä. (MongoDB 2024.)

4.5 Pinecone DB tietokanta

Pinecone ei ole perinteinen tietokantapalvelu, kuten MongoDB tai MySQL. Pinecone tarjoaa vektoritietokantoja moniulotteisen vektoridatan tallennusta varten. Moniulotteista vektoridataa käytetään monissa suositusjärjestelmissä,

hakukoneissa ja tekoälysovelluksissa. Pinecone hoitaa moniulotteisen datan indeksoimisen ja hakujen tekemisen, jolloin kehittäjille jää enemmän aikaa sovelluksen kehittämiselle. Pinecone tarjoaa reaaliaikaista indeksointia, lähimmän naapurin hakua ja tukee suuria tietokantoja. (Pinecone 2024.)

4.6 OpenAI API

OpenAI tarjoaa ohjelmointirajapinnan (Application Programming Interface API) kautta monenlaisia tekoälypalveluita erilaisiin käyttötarkoituksiin. Palveluihin kuuluu Large Languae Models (LLM) GPT-3,5 ja GPT-4, Tekstigenerointi, Kuva prosessointi ja paljon muuta. (OpenAI 2024a.)

API on suunniteltu skaalaavaksi, ja se pystyy vastaamaan suureen määrään pyyntöjä pienellä viiveellä. OpenAI API on helppo integroida omiin sovelluksiin ja verkkosivuihin. Se tukee suosittuja ohjelmointikieliä ja ohjelmointikehyksiä. (OpenAI 2024a.)

4.7 Clerk käyttäjähalllinta-alusta

Clerk on todennus- ja käyttäjähallinta-alusta, jota voidaan käyttää verkkosivujen ja mobiilisovellusten todennuksen ja käyttäjähallinnan toteuttamiseen. Alusta sisältää ominaisuuksia, kuten sisäänkirjautuminen, rekisteröityminen, salasanojen hallinta ja käyttäjäprofiilien hallinta. Clerk integroituu erilaisiin sovellusympäristöihin ja tarjoaa ratkaisuja käyttäjien hallinnan toteuttamiseen. (Clerk 2024.)

4.8 TMDb API

The Movie Database API rajapinta tarjoaa tietoa elokuvista, televisio-ohjelmista ja näyttelijöistä. TMDb on jatkuvasti päivitetty ja tarjoaa pääsyn laajaan elokuva ja tv-ohjelma tietokantaan. API sisältää yksityiskohtaista dataa elokuvista, kuten

nimen, genren, arvostelut ja paljon muuta. Se tukee hakuominaisuuksia, jolla voi hakea tietoa tietystä elokuvasta. API:n käyttö vaatii avaimen, jonka saa kirjautumalla sivulle. (The Movie Database 2024.)

5 Sovelluksen toteutus

5.1 Idea

Idea tähän opinnäytetyöhön alkoi aikaisemman kurssin työstä. Object Oriented Programming -kurssilla teimme ryhmätyönä elokuvasuositusjärjestelmän. Järjestelmä käytti yhteistoiminnallista suositusjärjestelmää ja netistä ladattua elokuvatietokantaa suositusten luomiseen. Ohjelma oli kirjoitettu Python-ohjelmointikielellä ja sillä oli graafinen käyttöliittymä, vaikkakin melko yksinkertainen ja vanhanaikainen. Käyttäjä antoi järjestelmälle elokuvan nimen ja järjestelmä antoi listan samankaltaisista hyvin arvostelluista elokuvista. Ohjelman antamat suositukset olivat yleisesti ottaen hyviä.

Ohjelma oli kuitenkin hyvin yksinkertainen. Se antoi suosituksia pelkästään elokuvan perusteella. Käyttäjän omilla mieltymyksillä ei ollut minkäänlaista vaikutusta suosituksiin. Tästä ohjelmasta syntyi idea tähän työhön.

5.2 Suunnittelu

Ohjelman suunnittelu alkoi ohjelmistokehyksen valitsemisella. Ohjelman vaatimuksena oli olla selkeä, helppokäyttöinen ja moderni. Aluksi tarkoitus oli tehdä ohjelma Reactilla, mutta huomasin, että monissa työpaikkahakemuksissa oli toivottu Next.js osaamista. Tästä syystä valitsin Next.js-ohjelmistokehyksen tätä työtä varten. Next.js sisältää Reactin ja monia muita hyödyllisiä sovelluskehitystyökaluja, joita hyödynnetään tässä työssä.

Seuraavaksi oli päätettävä, millainen suositusjärjestelmä ohjelmaan tarvitaan. Heräsi idea tekoälyn käytöstä järjestelmässä. Aluksi työhön valittiin tekoälyksi OpenAI:n ChatGPT 3,5 Turbo. Se valittiin tarpeeksi hyvän keskustelukyvyn ja hinnan perusteella. 3,5 Turbo maksaa syöttönä $0,50 per 1 miljoonaa merkkiä ja $1,50 per 1 miljoonaa merkkiä ulostulona (OpenAI 2024a).

Sovelluksen kehityksen aikana kuitenkin myös OpenAI:n API:t kehittyivät. 2024 heinäkuussa OpenAI julkaisi GPT-4o-mini mallin. Kyseinen malli sopii sovellukseen täydellisesti. Se on älykkäämpi kuin 3,5 Turbo ja samaan aikaan halvempi. GPT-4o-mini maksaa vain $0,150 per 1 miljoonaa merkkiä syöttönä ja $0,600 per 1 miljoonaa ulostulona. (OpenAI 2024a.)

Ohjelman käytön kannalta keskeinen ominaisuus on chatti, jossa voi keskustella tekoälyn kanssa. ChatGPT luo käyttäjän tietojen ja chat keskustelun perusteella suosituksia käyttäjälle. Chatin käyttö suositusjärjestelmässä tarjoaa käyttäjille miellyttävän ja yksilöllisen käyttäjäkokemuksen. Käyttäjät saavat suosituksia chatin kautta.

5.3 Ideoita tekoälyn hyödyntämiselle

Seuraavat ideat tekoälyn hyödyntämisestä suositusjärjestelmässä ideoitiin ChatGPT:n avulla työn suunnitteluvaiheessa. Tässä työssä ei ole tarkoitus rakentaa kaikkia ideoita. Ideoita voi hyödyntää mahdollisesti jatkokehityksessä tulevaisuudessa.

Tekoälyn käyttö avaa täysin uusia mahdollisuuksia suositusjärjestelmille. Tekoälyä voidaan kouluttaa elokuvien ja tv-ohjelmien sisällöllä, eli tekoäly laitetaan katsomaan sisältö. Tällä tavalla tekoäly pystyy käsittelemään valtavia määriä tietoa tehokkaasti. Se voi analysoida tuhansia elokuvia, sarjoja ja käyttäjäprofiileja nopeasti ja luotettavasti, mikä olisi mahdotonta tai erittäin hidasta ilman tekoälyä. Koulutuksen jälkeen tekoäly pystyy vastaamaan tarkemmin käyttäjän kysymyksiin sisällöstä. (Law 2022.)

Kuten aikaisemmin mainittiin, suuri ongelma käyttäjille on sisällön löytämiseen kuluva aika. Tekoälyn avulla voidaan vähentää sopivan sisällön löytämiseen kuluvaa aikaa. Kun käyttäjille tarjotaan parempia suosituksia, sisällön etsimiseen käytetty aika vähenee ja käyttäjätyytyväisyys paranee. Hyvin toimiva suositusjärjestelmä pitää käyttäjät sitoutuneina palveluun, koska se tarjoaa heille jatkuvasti kiinnostavaa katsottavaa. Tämä voi lisätä palvelun käyttöä ja

vähentää käyttäjien siirtymistä kilpaileviin palveluihin. (Gomez-Uribe & Hunt 2015.)

Tekoälyä voidaan myös hyödyntää tiettyjen vuorosanojen löytämiseen. Tekoäly voi luonnollisen kielen käsittelyä hyödyntäen prosessoida puhetta tai tekstiä. Sen jälkeen tekoäly voi etsiä yhteyksiä elokuvien käsikirjoituksiin. Tekoälyn haku ei tee hakuja sanasta sanaan vaan ymmärtää myös kontekstin. Tämä mahdollistaa parafraasien ja muuten epätarkkojen hakujen käytön, jos käyttäjä ei muista sanoja tarkkaan. (Law 2022.)

5.4 Käyttäjä data

Yleensä suoratoistopalveluilla on dataa käyttäjistään. Tämän datan avulla palvelut voivat antaa suosituksia käyttäjilleen. Mitä enemmän ja hyvä laatuista dataa on, sitä parempia suosituksia voidaan antaa. Tämä työ tehtiin ilman valmista dataa käyttäjistä, joten piti löytää tapa saada se käyttäjiltä.

Käyttäjistä luodaan profiili, jonka tiedot syötetään tekoälylle. Käyttäjä voi luoda profiilin kirjautumalla suositusjärjestelmä sovellukseen. Kirjautumisen sovellukseen hoitaa Clerk, joka tarjoaa laajat tunnistautumisominaisuudet ja vahvan turvallisuuden. Palveluun voi luoda uuden käyttäjän tai kirjautua olemassa olevilla Google tunnuksilla.

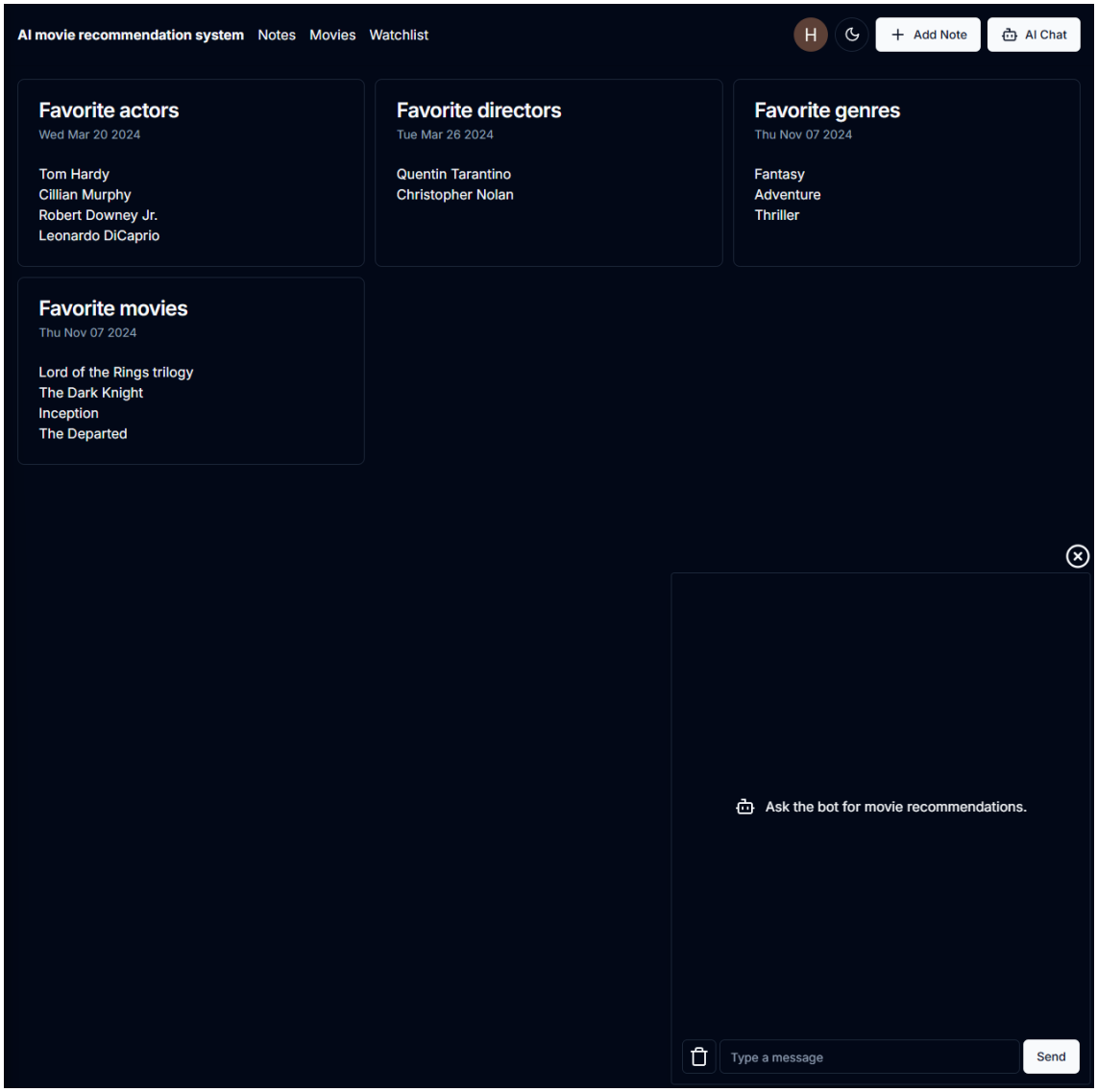
Kun käyttäjä kirjautuu ensimmäistä kertaa sovellukseen, käyttäjältä kysytään kysymyksiä elokuvamieltymyksiin liittyen. Käyttäjälle annetaan lista elokuvagenreistä (esim. toiminta, komedia, kauhu), joista voi valita omat suosikkinsa. Käyttäjät voivat valita mahdollisia suosikkinäyttelijöitä tai -ohjaajia. Käyttäjät voivat myös valita mahdolliset suoratoistopalvelut, joiden elokuva tarjonta voidaan ottaa huomioon suosituksissa.

Käyttäjä data tallennetaan muistilapuille, joiden otsikon ja sisällön käyttäjä saa päättää itse. Muistilappujen sisältö on melko vapaamuotoinen, joten käyttäjät voivat vapaasti lisätä haluamaansa tietoa. Kuvassa 4 nähdään käyttöliittymä

käyttäjätietosivulta, sekä chatti-ikkuna alaoikealla. Kuvassa nähdään myös esimerkki, miten käyttäjä voi käyttää muistilappuja tietojen antamiseen.

Käyttäjät voivat kerätä tekoälyn antamia elokuvasuosituksia katsomislistalle. Tämän jälkeen käyttäjät voivat merkitä elokuvat katsotuiksi, jolloin järjestelmä ei suosittele niitä enää, ellei käyttäjä erityisesti pyydä jo katsottuja elokuvia. Katsotut elokuvat antavat järjestelmälle lisää dataa, jonka perusteella suositella elokuvia.

Käyttäjien profiili ja katselulista tallennetaan MongoDB tietokantaan. Data on tallennettu turvallisesti ja käyttäjillä on mahdollisuus poistaa kerätty data ja profiili tietokannasta.



Kuva 4. Sovelluksen käyttöliittymä käyttäjätiedot sivulla

5.5 Suositusjärjestelmän korvaaminen tekoälyllä

Tutkimuksen perusteella paras käyttökohde ChatGPT:lle suosittelijana on toimia keskustelu kontekstissa. ChatGPT toimii kuin elokuvaekspertti, joka tietää kaikki elokuvat, jotka kuuluvat sen koulutusdataan. ChatGPT:ltä voi vapaasti kysyä elokuvasuosituksia tai muuta tietoa elokuviin liittyen. Tässä työssä mallia ei ole erikseen koulutettu elokuvadatalla, vaan käytetään valmiiksi olevaa koulutusdataan kuuluvaa elokuvatietämystä. Tämä ratkaisu saattaa aiheuttaa

ongelmia kaikkein uusimpien elokuvien kohdalla, jotka eivät sisälly mallin koulutusdataan.

Sovellus kerää käyttäjistä tietoja, jotka syötetään ChatGPT:lle. Mitä enemmän käyttäjätietoja on, sitä parempia ja tarkempia suosituksia malli pystyy antamaan. Käyttäjä tietoja tulee myös päivittää ja hienosäätää, jotta ChatGPT pystyy tuottamaan relevantteja suosituksia.

Koska työssä ei käytetä omaa elokuvakoulutusdataa, ChatGPT:n antamat suositukset tulee tarkistaa. Ilman tarkistusta ChatGPT saattaa hallusinoida elokuvia, joita ei oikeasti ole olemassa. The Movie Database API tarjoaa helpon tavan saada elokuva dataa, jolla tarkistetaan elokuvien olemassaolo.

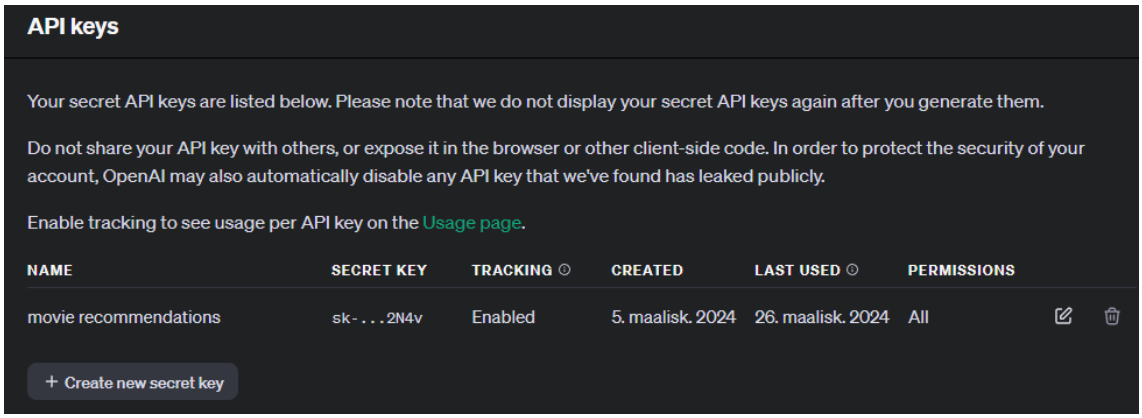
Prompteihin tulee myös kiinnittää huomiota. Promptit eli viestit ChatGPT:lle tulee muodostaa tavalla, joka kertoo selkeästi, mitä käyttäjä haluaa tapahtuvan. Prompteihin pitää siis sisällyttää esimerkiksi suositusten pyytäminen. Esimerkiksi:” Based on my favorite genres and actors, suggest new movies,” tai “Suggest movies similar to those on my watchlist”.

5.6 ChatGPT API:n implementointi

ChatGPT API:n käyttö vaatii OpenAI käyttäjätilin (OpenAI. 2024a). Open AI:n API referenssi tarjoaa laajan dokumentaation API:n ominaisuuksista. Sivulta löytyy ohjeita erilaisten ominaisuuksien käytöstä, kuten teksti-, audio- tai kuvagenerointi. Tässä työssä käytetään chat ja embedding API:a. On myös otettava huomioon, että API:n käyttö on maksullista. Tilille on lisättävä vähintään 5 $, jotta API:a pääsee käyttämään. Mallien käyttö on ainakin kehitysvaiheessa halpaa. Tämän työn kehityksen ja testauksen aikana on kulutettu vasta 0,03 $. Kulutusta voi seurata OpenAI kehittäjäsivulta (OpenAI 2024b.)

OpenAI kehittäjäsivulle kirjautumisen jälkeen API avain sivulla (Kuva 5.) voi luoda uuden salaisen avaimen (OpenAI 2024b). Avain pitää tallentaa

turvalliseen paikkaan, ettei sitä päästä käyttämään väärin. Avainta ei myöskään kannata hukata, koska sitä ei voi saada takaisin. Jos avain katoaa, pitää luoda uusi avain.



Kuva 5. OpenAI API avain sivu. Sivulla voi luoda uusia avaimia, hallita luotuja avaimia ja poistaa niitä. (OpenAI 2024b.)

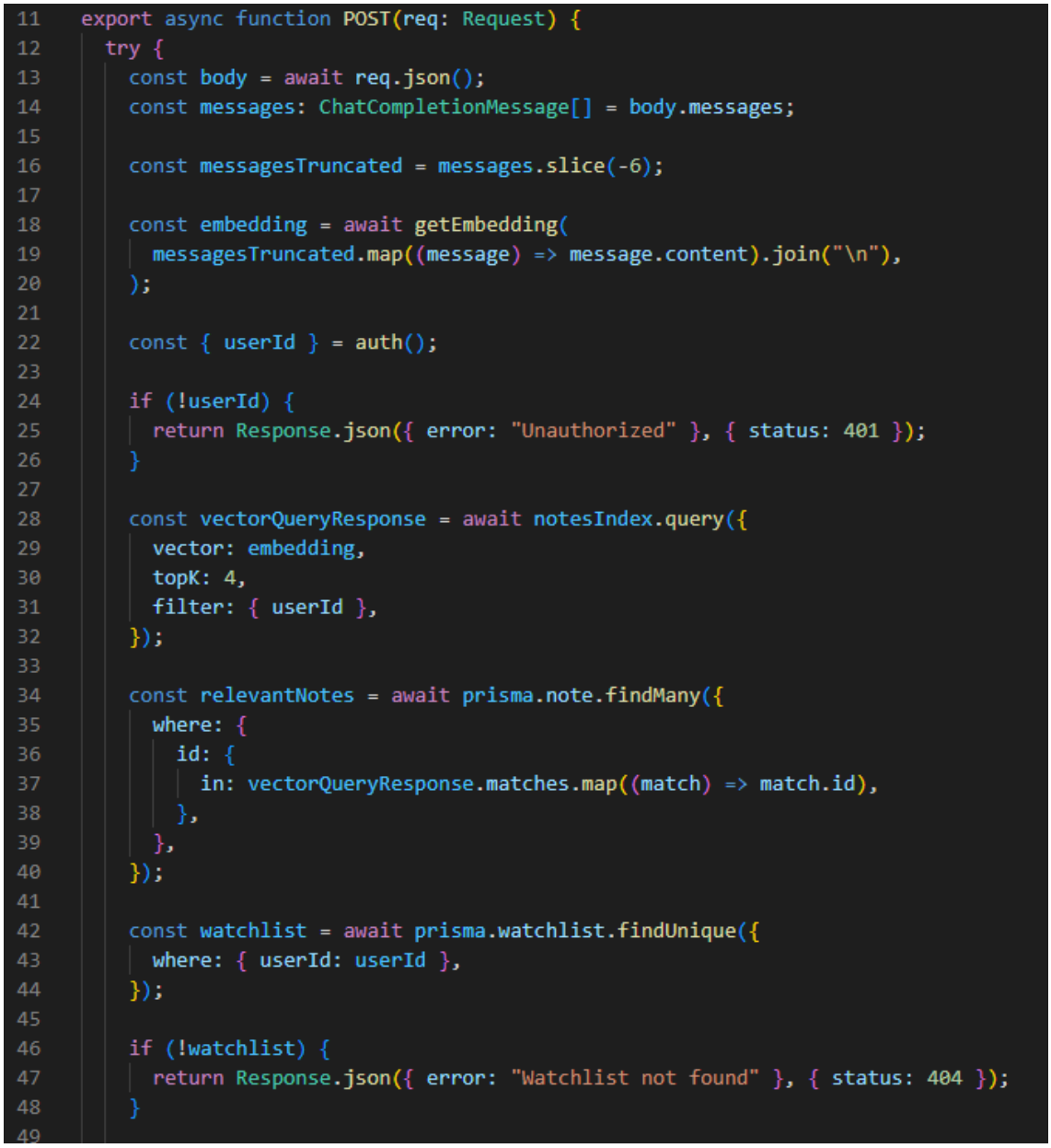
Avaimen luomisen jälkeen asennetaan OpenAI API:n kirjastot. OpenAI tarjoaa viralliset API kirjastot Pythonille, Node.js:lle ja .NET:lle. Muille kielille on olemassa kehittäjäyhteisön tekemiä kirjastoja (OpenAI. 2024d). Tässä projektissa käytetään Next.js ohjelmistokehystä, joka sisältää Node.js ohjelmistokehyksen, joten valitaan Node.js kirjasto. Kirjasto asennetaan komentorivikomennolla `npm install openai`.

Seuraavaksi lisätään API avain paikalliseen environment-muuttujaan. Tämän jälkeen APIn käyttö voidaan aloittaa.

ChatGPT:n kanssa keskustelua varten pitää luoda chatti-ikkuna ja syöttää API:lle tarvittavat tiedot. Luodaan Next.js dokumentaation mukainen api route tekemällä seuraavanlainen kansiorakenne: src/app/api/chat/route.ts. Route.ts tiedostoon voidaan nyt määrittää millaisia pyyntöjä osoitteeseen /api/chat lähetetään. Määritetään POST-pyyntö, jolla saadaan lähetettyä chat-viestejä.

Kuvassa 6 nähdään, miten pyyntö on rakennettu. Osoitteeseen lähetetty pyyntö vastaanotetaan JSON muodossa ja talletetaan vakioon `body`. Viesteistä

valitaan 6 viimeisintä syötettäväksi tuleviin viesteihin ja talletetaan vakioon `messagesTruncated`. Viesteistä muodostetaan `embedding`, kutsumalla `getEmbedding` funktiota. Varmistetaan, että käyttäjä on kirjautuneena ja hoidetaan siihen liittyvä virhetarkistus. Seuraavaksi haetaan vektoritietokannasta tärkeimmät tiedot viesteihin liittyen. Muuttujalla `topK` määritetään, kuinka paljon tietoa käyttäjästä haetaan. Nykyisellä `topK:4` asetuksella haetaan relevanteimmat 4 muistilappua tietokannasta, jotka syötetään myöhemmin ChatGPT:lle. Lopuksi tarkistetaan, että käyttäjällä on olemassa oleva katselulista.



```
export async function POST(req: Request) {
  try {
    const body = await req.json();
    const messages: ChatCompletionMessage[] = body.messages;

    const messagesTruncated = messages.slice(-6);

    const embedding = await getEmbedding(
      messagesTruncated.map((message) => message.content).join("\n"),
    );

    const { userId } = auth();

    if (!userId) {
      return Response.json({ error: "Unauthorized" }, { status: 401 });
    }

    const vectorQueryResponse = await notesIndex.query({
      vector: embedding,
      topK: 4,
      filter: { userId },
    });

    const relevantNotes = await prisma.note.findMany({
      where: {
        id: {
          in: vectorQueryResponse.matches.map((match) => match.id),
        },
      },
    });

    const watchlist = await prisma.watchlist.findUnique({
      where: { userId: userId },
    });

    if (!watchlist) {
      return Response.json({ error: "Watchlist not found" }, { status: 404 });
    }

```

Kuva 6. Kuvassa haetaan tarvittavat tiedot ChatGPT API:lle.

Kuvassa 7 määritellään `systemMessage`, jossa annetaan ohjeet, miten ChatGPT:n tulisi käyttäytyä tässä keskustelussa. Chatille annetaan ohjeeksi käyttäytyä elokuva eksperttinä. Seuraavaksi chatille annetaan kirjautuneen käyttäjän katselulistan elokuvien otsikot luettavaksi, jonka jälkeen annetaan käyttäjän tiedot ja mieltymykset. Lopuksi annetaan ohjeeksi kutsua `addMovieToWatchlist` funktiota, jos käyttäjä pyytää elokuvan lisäämistä katselulistaan. Funktioiden käytöstä myöhemmin lisää.

```
const systemMessage: ChatCompletionSystemMessageParam = {
  role: "system",
  content:
    "You are a movie expert. You can answer user's questions about movies and give recommendations. " +
    "Here is the users watchlist: " +
    watchlist.movieTitles.join(", ") +
    ". " +
    "Here are some notes that might help you answer the user's question: " +
    relevantNotes
      .map((note) => `Title: ${note.title}\n\nContent:\n${note.content}`)
      .join("\n\n") +
    ". You can ask the user if they want to add a movie to their watchlist using the `addMovieToWatchlist` function with the movie title.",
};

const response = await openai.chat.completions.create({
  model: "gpt-4o-mini",
  stream: true,
  messages: [systemMessage, ...messagesTruncated],

  functions: [
    {
      name: "addMovieToWatchlist",
      description: "Add a movie to the user's watchlist",
      parameters: {
        type: "object",
        properties: {
          movieTitle: {
            type: "string",
            description: "The title of the movie to add to the watchlist",
          },
        },
        required: ["movieTitle"],
      },
    },
  ],
  function_call: "auto",
});
```

Kuva 7. Kuvassa näkyy systemMessage määritys ja response objektin määritys

Seuraavaksi on määritelty `response`, jonka ChatGPT API tuottaa. Määritetään `model`, jota keskustelu käyttää. Tässä tapauksessa valitaan `gpt-4o-mini`. `Stream` määritetään olemaan `true`, jolloin teksti suoratoistetaan chattiin. Tämä luo efektin, jossa teksti ilmestyy kirjain kerrallaan chattiin. Käyttäjäkokemus paranee, kun käyttäjä ei joudu odottamaan koko viestin generointia ennen kuin sitä voi alkaa lukemaan. `Messages` kohtaan määritetään `systemMessage` ja käyttäjän lähettämät viestit. Functions kohtaan määritetään funktiot, joita ChatGPT voi käyttää. Funktioille määritellään nimi, kuvaus ja parametrit, jotta ChatGPT ymmärtää, miten funktiota tulisi käyttää.

Funktioiden käyttö ChatGPT:n avulla avaa monenlaisia mahdollisuuksia sovellukselle. Funktioiden avulla ChatGPT voi vaikuttaa sovelluksen muihin toimintoihin. Prosessi toimii aluksi määrittämällä ChatGPT:lle tietyt lauserakenteet, jotka viittaavat haluun lisätä elokuva katselulistalle. Esimerkiksi ”Lisää suositukset katselulistalle”. Ymmärrettyään käyttäjän tarkoituksen ChatGPT API lähettää pyynnön elokuvien lisäämisestä käyttäjän katselulistalle.

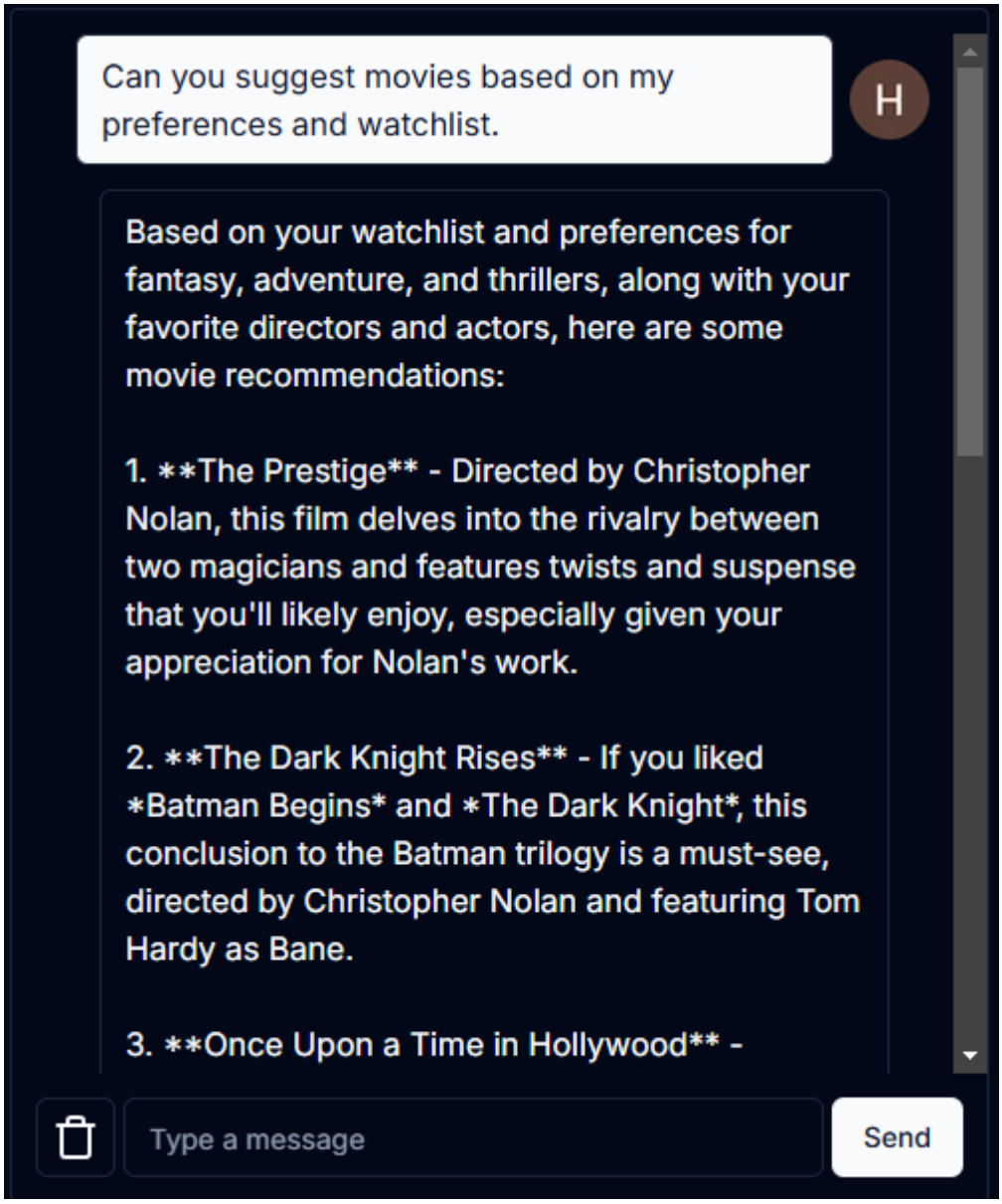
Tämän jälkeen sovelluksen backend hoitaa elokuvien lisäämisen ja tallentaa tiedot tietokantaan.

Esimerkki tapahtumasarja:

1. Käyttäjä: ”Lisää Inception katselulistalleni.”
2. ChatGPT tunnistaa komennon ja antaa käyttäjälle vahvistuksen: ”Selvä, lisätään Inception katselulistalle”.
3. Next.js sovellus tekee tarvittavat toimenpiteet ja lisää pyydetyn elokuvan käyttäjän katselulistalle ja tallentaa tiedon tietokantaan.
4. Onnistuneen lisäyksen jälkeen ChatGPT voi vielä vahvistaa käyttäjälle, että lisäys onnistui tai ilmoittaa epäonnistumisesta, jos lisäys epäonnistui.

5.7 Sovelluksen käyttö

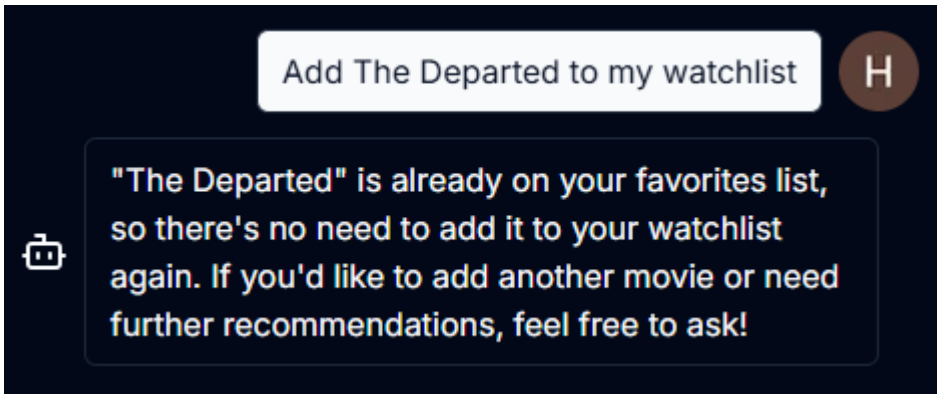
Käyttäjät saavat elokuvasuosituksia sovelluksen chatin kautta. Kuvassa 8 nähdään chatin käyttöliittymä. Kuvassa käyttäjä pyytää tekoälyltä suosituksia käyttäjän mieltymysten ja katselulistan perusteella. ChatGPT vastaa suosittelemalla erilaisia elokuvia ja perustelemalla miksi elokuvaa suositellaan. Keskustelupohjaisen käyttökokemuksen ansiosta käyttäjä voi pyytää lisää informaatiota elokuvista tai muita suosituksia.



Kuva 8. Sovelluksen chatin käyttöliittymä.

Käyttäjä voi myös pyytää ChatGPT:tä lisäämään elokuvan katselulistalle. Kuvassa 9 näytetään esimerkki tapahtumasta. Käyttäjä pyytää elokuvan lisäyksen katselulistalle ja ChatGPT kutsuu funktiota, joka suorittaa operaation. Ohjelma lisää elokuvan käyttäjän katselulistaan ja ChatGPT luo vastauksen käyttäjälle. Vastausta luodessa luetaan käyttäjän tiedot ja katselulista, jossa on lisätty elokuva. Tämä johtaa kuvassa nähtyyn tapahtumaan, jossa ChatGPT väittää, että elokuva on jo katselulistassa, vaikka se lisättiin sinne samassa

vaiheessa keskustelua. Ongelma on todennäköisesti korjattavissa antamalla paremmat ohjeet ChatGPT:lle tai muokkaamalla sovelluksen logiikkaa.



Kuva 9. Elokuvan lisääminen katselulistaan.

6 Johtopäätökset ja kehitysideat

Opinnäytetyön tutkimuksen ja käytännön toteutuksen perusteella voidaan sanoa ChatGPT:n soveltuvan suositusjärjestelmä käyttötarkoitukseen. ChatGPT tarjoaa järkeviä ja hyödyllisiä suosituksia. Järjestelmä ottaa käyttäjästä tarjotut tiedot huomioon suosituksia luodessaan ja tarjoaa käyttäjäystävällisen käyttökokemuksen. Käyttäjät pystyvät myös hyödyntämään chatin tarjoamia ominaisuuksia, kuten elokuvien lisäämisen katselulistaan.

Työn käytännön osuudessa esiteltiin, miten ChatGPT API implementoidaan verkkosovellukseen. ChatGPT API:lla on monenlaisia käyttötarkoituksia verkkosovelluksissa ja palveluissa. ChatGPT sopii parhaiten chattipohjaisiin käyttötarkoituksiin, joissa se voi toimia ihmisen kaltaisena keskusteluagenttina.

Sovelluksen kehityksen aikana opin paljon modernista verkkosovelluskehityksestä ja tekoälyn hyödyntämisestä erilaisiin tarkoituksiin. Varsinkin Next.js ohjelmistokehyksen ja ChatGPT API:n käytöstä opin paljon.

Työn aikana heräsi monenlaisia jatkokehitysideoita. Ohjelman kehittämistä voidaan jatkaa ja lisätä uusia ominaisuuksia. Yksi idea olisi laajentaa sovellusta lisäämällä siihen yhteisöominaisuuksia, kuten elokuva-arvosteluja tai katselulistojen jakamista. Chatin API:lle voisi määritellä lisää käytettäviä funktioita, jotka parantavat käyttäjäkokemusta.

Chattiin voisi lisätä äänikomennot. Suurin osa suoratoistopalveluiden käyttäjistä katsoo ohjelmia televisiosta. Chatin käyttö kaukosäätimellä ei ole kovinkaan luontevaa, joten äänikomentojen käyttö parantaisi käyttäjäkokemusta huomattavasti. Käyttäjät voisivat keskustella elokuvista tv:n kanssa ja selata valikoimaa äänikomennoilla.

Kokonaisuudessaan työn tulokset korostavat tekoälyn merkitystä moderneissa suositusjärjestelmissä ja verkkosovelluksissa. ChatGPT API:n hyödyntäminen elokuvasuositusjärjestelmässä osoitti, kuinka tekoäly voi tarjota käyttäjille entistä henkilökohtaisempia ja käyttäjäystävällisempiä ratkaisuja. Työn tuloksia voidaan hyödyntää suositusjärjestelmien ja verkkosovellusten jatkokehityksessä

esimerkiksi dynaamisten suositusten, paremman käyttäjäkokemuksen tai skaalautuvampien järjestelmien kehityksessä. Näiden tulosten pohjalta on mahdollista ideoida tulevaisuutta, jossa tekoäly toimii keskeisenä osana käyttäjien arkea, tarjoten räätälöityjä ja intuitiivisia ratkaisuja eri tarpeisiin. Työ luo pohjaa uusille innovaatioille ja tutkimuksille, jotka voivat edelleen syventää tekoälyn roolia suositusjärjestelmissä ja sovelluskehityksessä.

Lähteet

Anand, V. Yang, M. Zhao, Z. Mitigating Filter Bubbles within Deep Recommender Systems. 2021 Viitattu 9.12.2024 https://ar5iv.org/html/2209.08180

AWS 2024. What is an API. Viitattu 17.11.2024 https://aws.amazon.com/what-is/api/

Burke, R. 2014. Hybrid Web Recommender Systems. Viitattu 10.11.2024. https://web.archive.org/web/20140912085014/http://www.dcs.warwick.ac.uk/~acristea/courses/CS411/2010/Book%20-%20The%20Adaptive%20Web/HybridWebRecommenderSystems.pdf

Carlini, N. Tramer, F. Wallace, E. Jagielski, Herbert-Voss, A. Lee, K. Roberts, A. Brown, T. Song, D. Erlingsson, U. Oprea, A. Raffel, C. 2021. Extracting Training Data from Large Language Models. Viitattu 10.11.2024 https://arxiv.org/abs/2012.07805

ChatGPT 2024. (22. Marraskuu 2024 versio). [Suurikielimalli]. Viitattu 4.12.2024 https://chatgpt.com/

Chatterjee, S. 2023. Recommendation Systems: Ethical Challenges and the Regulatory Landscape. Viitattu 9.12.2024 https://www.holisticai.com/blog/recommendation-systems

Clerk 2024. The most comprehensive User Management Platform. Viitattu 22.4.2024 https://clerk.com/

Colyer, A. 2016 The amazing power of word vectors. Viitattu 16.10.2024 https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/

Cummins, E. 2018. Dark mode is easier on your eyes-and battery. Viitattu 11.11.2024 https://www.popsci.com/night-dark-mode-design/

Gomez-Uribe, C. Hunt, N. 2015 The Netflix Recommender System: Algorithms, Business Value, and Innovation. ACM Trans. Manage. Inf. Syst. 6, 4, Article 13 Viitattu 7.4.2024 https://dl.acm.org/doi/10.1145/2843948

Goodrow, C. 2021 On YouTube’s recommendation system. Viitattu 20.4.2024 https://blog.youtube/inside-youtube/on-youtubes-recommendation-system/

Krysik, A. 2024 Netflix Algorithm: How Netflix Uses AI to Improve Personalization. Viitattu 3.12.2024 https://stratoflow.com/how-netflix-recommendation-algorithm-work/

Law, M. Visual search engines: the role of AI and machine vision. 2022. Viitattu 9.12.2024 https://aimagazine.com/articles/visual-search-engines-the-role-of-ai-and-machine-vision

Lika, B; Kolomvatsos, K. & Hadjiefthymiades, S. 2014. Facing the cold start problem in recommender systems. Expert Systems with Applications. Volume 41, Issue 4, Part 2. Viitattu 22.4.2024 https://doi.org/10.1016/j.eswa.2013.09.005

Liu, J. Liu, C. Zhou, P. Lv, R, Zhou. Zhang, Yan. 2023. Is ChatGPT a Good Recommender? A Preliminary Study. Viitattu 11.11.2024 https://arxiv.org/pdf/2304.10149

Melville, P. Sindhwani, V. 2010. Recommender Systems. Encyclopedia of Machine Learning. Chapter No. 338. Viitattu 27.3.2024 https://www.prem-melville.com/publications/recommender-systems-eml2010.pdf

Metzger, S. 2022 A Beginner's Guide to Tokens, Vectors, and Embeddings in NLP. Viitattu 16.10.2024 https://medium.com/@saschametzger/what-are-tokens-vectors-and-embeddings-how-do-you-create-them-e2a3e698e037

MongoDB 2024. Database. Deploy a multi-cloud database. Viitattu 22.4.2024 https://www.mongodb.com/products/platform/atlas-database

Next.js 2024. The React Framework fo the Web. Viitattu 22.4.2024 https://nextjs.org/

OpenAI 2023. GPT-4 Technical Report. Viitattu 22.4.2024 https://arxiv.org/pdf/2303.08774.pdf

OpenAI 2024a. Introduction. Viitattu 11.11.2024 https://platform.openai.com/docs/api-reference/introduction

OpenAI 2024b. OpenAI developer platform Viitattu 22.4.2024 https://platform.openai.com/docs/overview

OpenAI 2024c. Pricing. Viitattu 10.11.2024 https://openai.com/api/pricing/

OpenAI 2024d. Libraries. Viitattu 3.12.2024 https://platform.openai.com/docs/libraries#community-libraries

Pinecone 2024. Build knowledgeable AI. Viitattu 22.4.2024 https://www.pinecone.io/

Quintais, J. P. 2023. Generative AI, Copyright and the AI Act. Viitattu 10.11.2024 https://copyrightblog.kluweriplaw.com/2023/05/09/generative-ai-copyright-and-the-ai-act/

React 2024. React The library for web and native user interfaces. Viitattu 22.4.2024 https://react.dev/

Rockwell, A. 2017. The History of Artificial Intelligence. Viitattu 22.4.2024 https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/

Shadcn/UI 2024. Introduction. Viitattu 22.4.2024 https://ui.shadcn.com/docs

Tailwind CSS 2024. Rapidly build modern websites without ever leaving your HTML. 22.4.2024 https://tailwindcss.com/

The Movie Database. 2024. Getting Started. Viitattu 11.11.2024 https://developer.themoviedb.org/docs/getting-started

Zhang, Q; Lu, J. & Jin, Y. 2020. Artificial intelligence in recommender systems. *Complex Intell. Syst.* **7**, 439–457. Viitattu 19.4.2024 https://doi.org/10.1007/s40747-020-00212-w

Zi, W; El Asri, L; Prince, S. 2023 A High-level Overview of Large Language Models. Viitattu 16.10.2024 https://www.borealisai.com/research-blogs/a-high-level-overview-of-large-language-models/