
# Tekoälyn hyödyntäminen datankeruussa

5.12.2024

## Johdanto

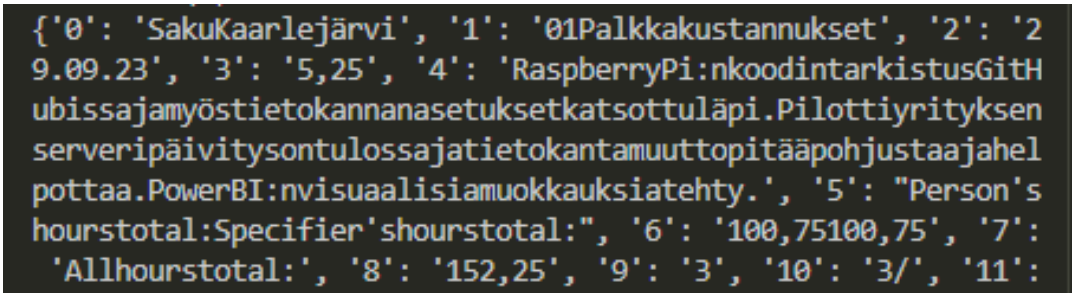
PDF-tiedostot ovat yksi yleisimmistä dokumenttitiedostoista, joita käytetään erilaisiin tarkoituksiin, kuten laskuihin, sopimuksiin, e-kirjoihin ja muihin dokumentteihin. PDF-tiedostoissa halutaan myös kerätä dataa, jota voitaisiin käsitellä ja hyödyntää. PDF-tiedostojen käsittelemistä vaikeuttaa niiden epälooginen järjestys. Pythonille on kehitetty  erilaisia PDF-tiedostojen käsittelykirjastoja, joista valitaan ominaisuuksiltaan hyödyllisin tähän VauhtiData-hankkeen pilottiin. Kolme esimerkkivaihtoehtoa ovat PyPDF2-, pdfminer.six- ja pdfplumber –kirjastot, jotka pystyvät tuomaan dataa ulos PDF-tiedostoista. Tässä hankkeen pilotissa on testattu pdfplumber, joka on kehitetty pdfminer.six -lähdekoodista.

## Tekoäly apuna ongelmanratkaisussa

Kun käytetään tekoälypalveluita, kuten OpenAI:n luomaa ChatGPT:ä, on hyvä ottaa huomioon tietoturvallisuus ja välttää lähettämästä sille mitään arkaluontoista, kuten henkilökohtaisia tietoja, salasanoja muita tärkeitä tietoja, jotka ei kuulu kenellekään ulkopuoliselle henkilölle tai edes robotille. ChatGPT kerää käyttäjiltään kaiken tiedon, mitä sille syötetään (Veale, 2023).

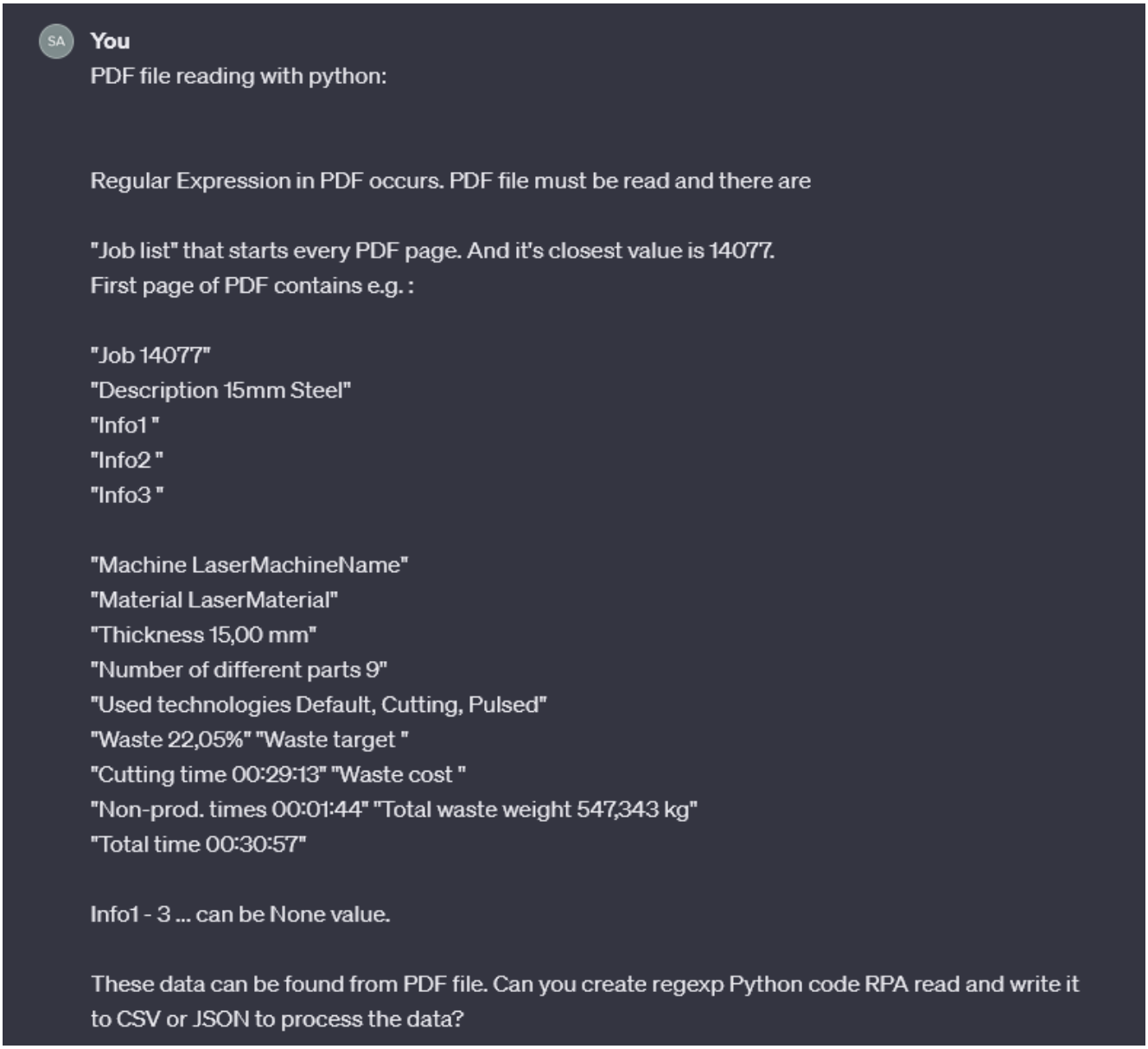
Aivan ensimmäiseksi tässä hankkeen pilotissa tekoälyä on pyydetty luomaan ohjelmakoodi, joka prosessoi PDF-tiedostoja tuoden niistä tekstiä komentoriville. Muokkaamalla koodia niin, että siihen lisätään PDF-

tiedostojen polun sijainti, jolla pystytään testaamaan tekoälyn antamaa koodin toimivuutta. Tiedoston lukeminen onnistuu ja käsittelemätön data on esitetty kuvassa 1., joka on Reportronic-ohjelmasta ladatun PDF-tiedoston ulostulo. Kuvasta 1. voidaan huomata, että teksti on epäselvää eikä sitä pystytä käsittelemään eikä siten hyödyntämäänkään.



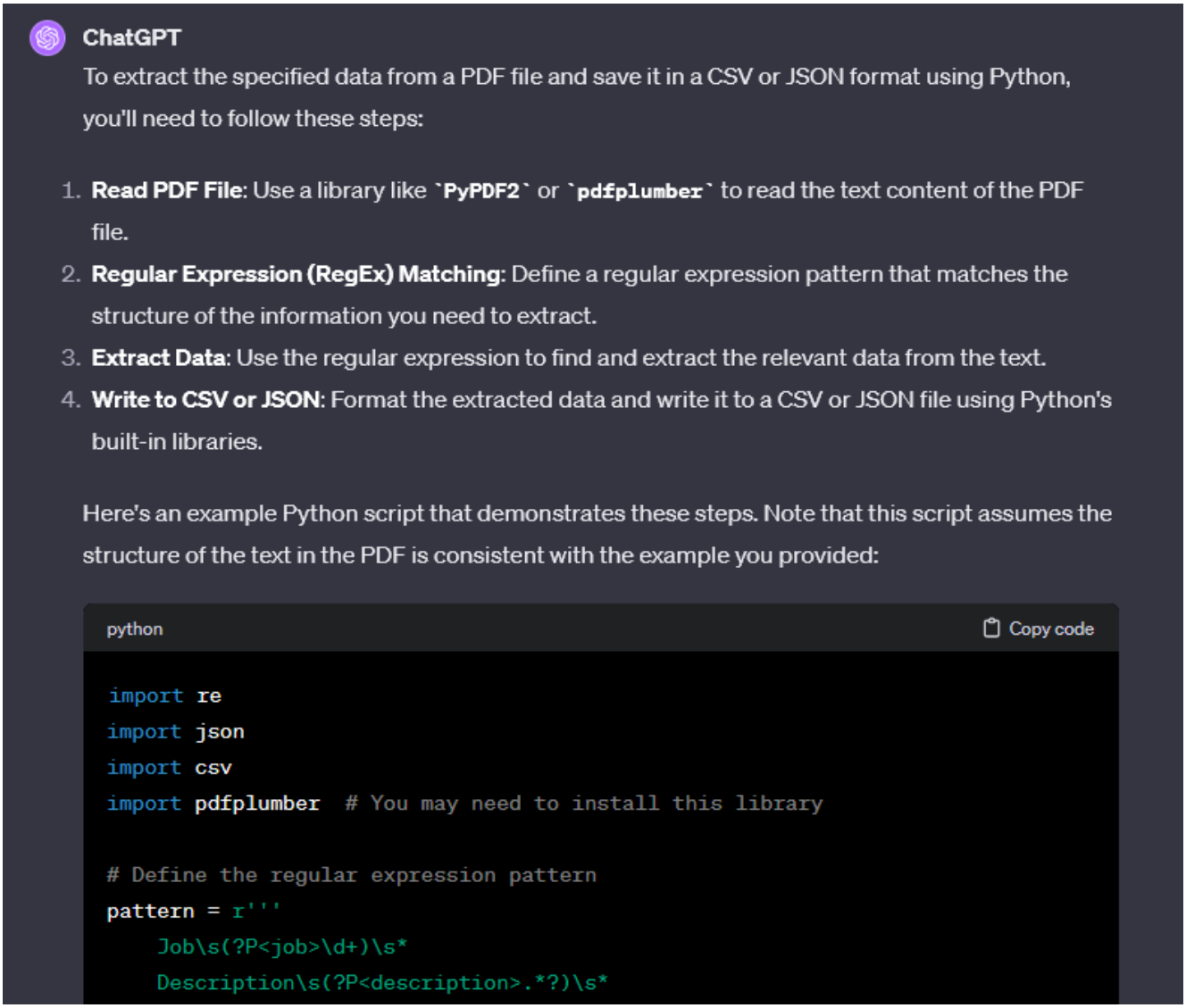
Kuva 1. Kuvankaappaus PDF-tiedoston muunnoksesta tekstiksi.

Johtopäätöksenä on se, että testattua koodia pitää muokata, jotta saadaan laadukkaampaa dataa PDF-tiedostoista. Jotta koodin toimivuutta saataisiin validoitua, niin seuraavaksi on luettava läpi pilotissa käsiteltävien PDF-tiedostojen tekstit ja manuaalisesti kirjoittaen ylös halutut etsittävät tiedot ja niiden toistuvat otsikot. Tekoälyä pyydetään nyt apua ongelmanratkaisussa näiden otsikoiden ja vaaditun datan suhteen.




Kuva 2. Kysymys ongelmaan ChatGPT:ltä.

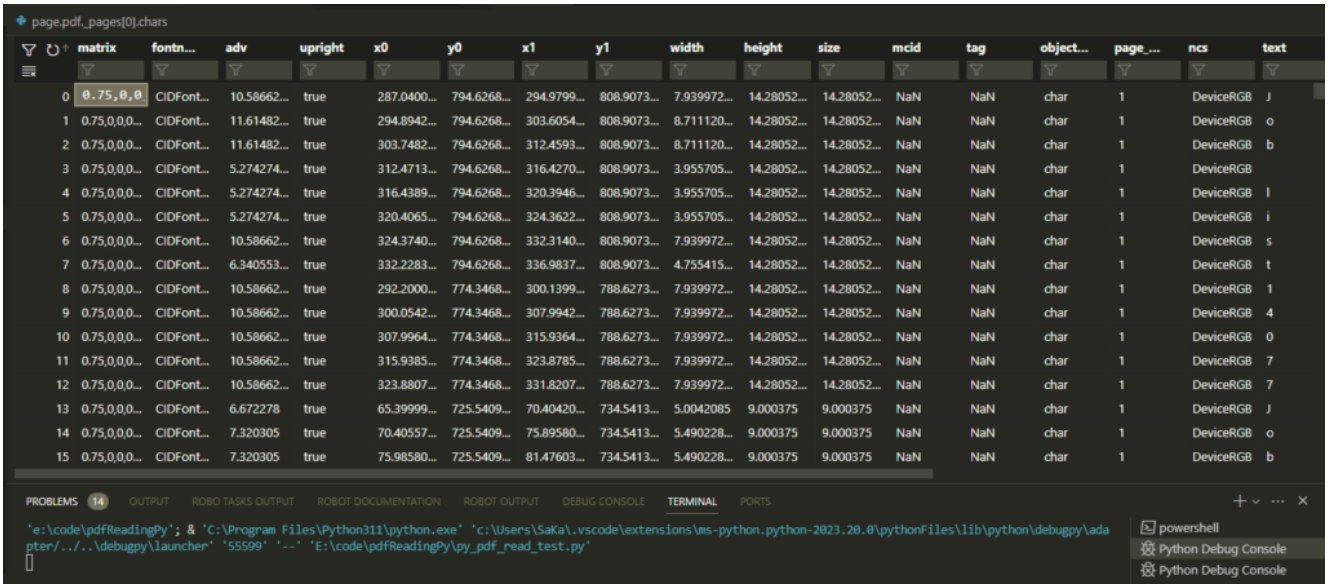
Kuvassa 2. tekoälylle on lähetetty viesti sisältäen etsittävät otsikot ja niiden mahdolliset tiedot, jotka halutaan kerätä ja mitä ongelmia halutaan ratkaista Python-ohjelmointikielellä. Tässä esimerkissä on käytetty englannin kieltä promptien kirjoittamiseen omasta tottumuksesta, mutta tällaisessa tilanteessa ChatGPT-työkalu todennäköisesti pystyy hyvin tulkkaamaan kysymykset ja vastaukset, jos käytetään suomen kieltä. Erityisesti on huomioitava, että tekoälyltä saadut viestit ja vastaukset eivät ole aina oikein sekä niiden toimivuutta ja luotettavuutta on suositeltava aina tarkistaa.




Kuva 3. Ongelman ratkaisua.

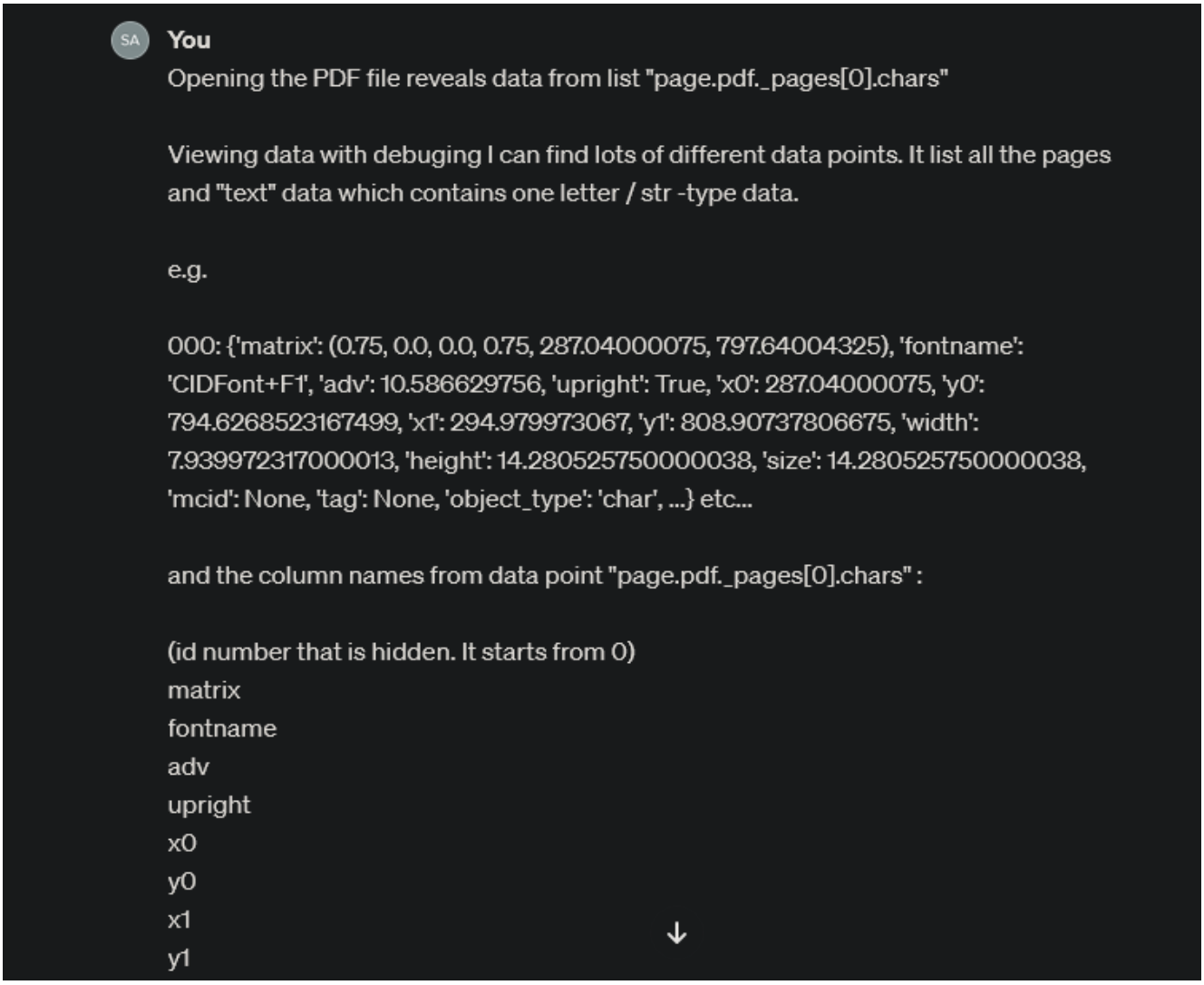
Kuvassa 3. esitetään ChatGPT:n kirjoittama vastaus kuvassa 2. esitettyyn kysymykseen, jossa aluksi kerrotaan käyttäjälle aluksi suunnitelma datankeruuseen. Siinä esitellään, kuinka luodaan aloituskoodi, jolla pystytään avaamaan PDF-tiedosto ja tuomaan pyydettyjä toistuvia kaavoja, kuten “Job” tai “Description”, komentoriville, sekä viemällä saadut tiedot CSV- ja JSON-tiedostoihin.

Tässä esimerkkikoodissa käytetään tekoälyn suosittelemaa pdfplumber-kirjastoa, joka perustuu pdfminer.six -lähdekoodiin (jsvine, i.a.). Edelleen testaamalla tekoälyltä saatua koodia käyttäen pilotin PDF-tiedostoja, koodi ei kuitenkaan löydä esitettyjä toistuvia kaavioita, joten koodia tulee välttämättä muokata ja testata. Debuggauksella pyritään selvittämään, että mitä pdfplumber-kirjaston avulla on kerätty PDF-tiedostoista ja mikä olisi mahdollinen ratkaisu ongelmaan. Koodissa on “with pdfplumber.open(pdffile) as pdf”, joka avaa PDF –tiedostot ja tämän avulla pystyttiin löytämään ratkaisu ongelmaan.



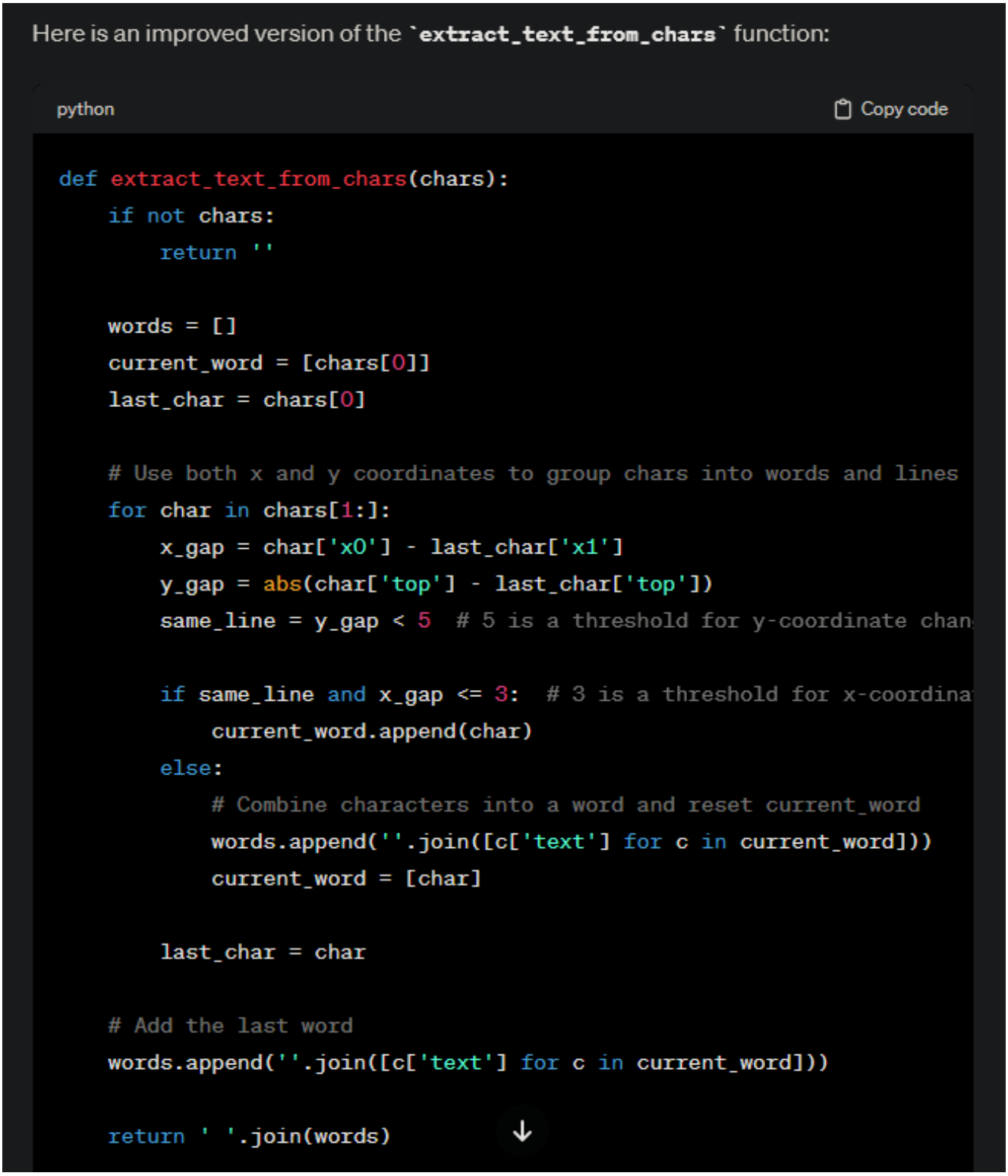

Kuva 4. Koodin debuggaamista.

Kuvassa 4. esitetyssä debuggauksessa “pdf” muuttujaa, löytyy kohta “chars”, joka sisältää tiedot “matrix” ja “fontname”, jotka aluksi eivät kerro mitään. Kuitenkin avaamalla Visual Studio Codessa tämän muuttujan “View Value in Data Viewer”, pystytään tarkemmin selaamaan koko datakokonaisuutta ja sen sisältöä. Kuvassa 4. nähdään, että jokainen sana on jaettu kirjaimiin “text” -sarakkeessa ja niille on määritetty x ja y –akselisijainnit. Tämän avulla pystyttäisiin paikantamaan jokaisen halutun tekstin sijainnit ja varmistamaan tarkasti, mistä dataa on kerätty. Antamalla nämä löydetyt tiedot ChatGPT:lle, niin pystyttiin jatkamaan ongelman ratkaisemista.



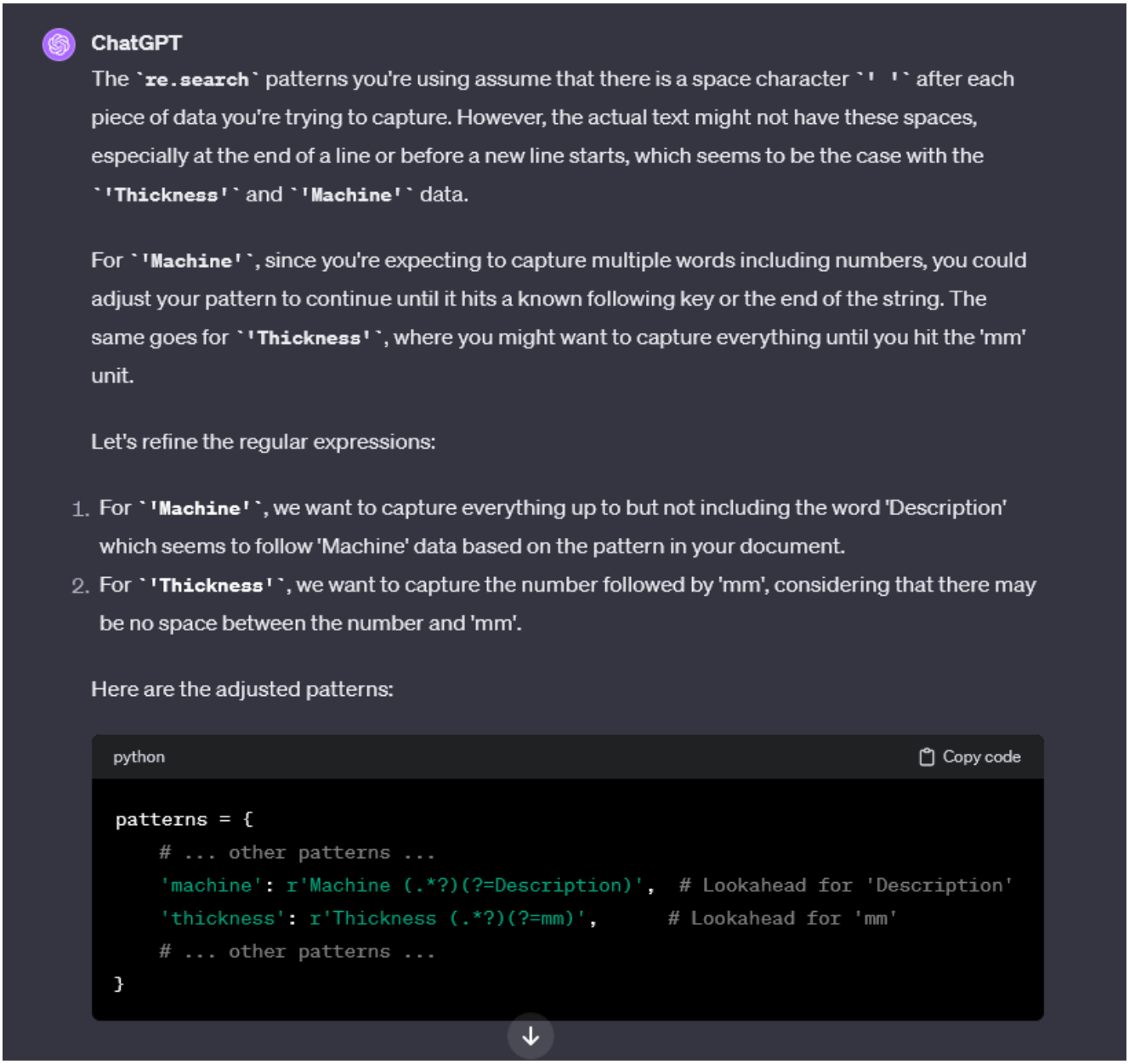

Kuva 5. Debuggaamista tekoälyn avulla.

Kuvassa 5. tekoälylle on annettu käsiteltäväksi kuvassa 4. esiintyneitä havaintoja koordinaateista. Tämän tiedon avulla tekoäly rakensi y-akseliin raja-arvoja, jotka määrittävät kerättävän datan sijainnin. Tekoäly on olettanut, että dataa löydetään ainoastaan yhdeltä sivulta kappale kerrallaan, ottamatta huomioon samalla rivillä olevia tekstejä tai dataa. Pelkästään tämän viestin avulla, tekoäly ei kuitenkaan pystynyt kirjoittamaan toimivaa koodia, jolloin taas tulee testata koodia ja kysyä tarkentavia kysymyksiä tekoälyltä.




Kuva 6. Raja-arvot X ja Y –akseleille.

Useiden komentojen ja korjausten jälkeen, tekoälyn tukena kirjoitettu koodi löytää kirjainten ja tekstien sijainnit loogisemmassa järjestyksessä, kuin lähtötilanteessa. Kuvassa 6. koodiin on määritetty raja-arvoja tekstien sijainnille. Ongelmaksi koitui PDF-tiedostojen sivut, jotka sisältävät useita sarakkeita. Ilman rajoja, koodi keräsi samalta riviltä kaikki tiedot, tietämättään sarakkeiden olemassaolosta ja näin luoden hämmentävää dataa. Lisäämällä raja-arvot, huomataan edelleen datan ulostulossa visuaalisia virheitä, jotka tuovat ongelmaa datan käsittelyyn.




Kuva 7. Toistuvien kaavojen raja-arvot.

Kuvassa 7. on ChatGPT:n antama vastaus virheellisen datan korjauksesta. Tekoälylle kerrottiin, että se leikkasi “Machine” ja “Thickness” datan useaan osaan. Esimerkiksi “Thickness” oli jaettu välilyönnein ja dataksi tuli “15,00” ja lisäsi ylimääräisen sarakkeen “mm”. Tekoälyn vastaus kertoi, kuinka rajoittaa löydettävä dataa. “Machine” kerää kaiken seuraavaan “Description” otsikkoon asti ja “Thickness” kerää vain numerot ilman “mm” merkintöjä.

# Johtopäätös

ChatGPT:n kaltaiselle tekoälylle voidaan suoraan viedä eri tiedostoja, kuten kuvia, videoita ja dokumentteja. Mutta ChatGPT:tä käytettäessä on otettava huomioon arkaluontoisten tietojen käsittely ja niistä syntyvät tietoturvariskit. Toki tänne olisi voinut viedä kaikki PDF-tiedostot ja kysyä, miten tästä kerätään nämä halutut tiedot ja samalla ruokkia mahdollisesti tekoälyn tietokantaa datalla, joka ei kuulu sille. Olisi myös mahdollista käyttää tekoäly palvelua käsittelemään PDF-tiedostoja API-kutsujen avulla, mutta se olisi tässä tapauksessa

resurssien ja myös energian tuhlausta. Kaikki ongelmat eivät vaadi tekoälyä ratkaisemaan ne puolestamme, mutta tekoäly voi auttaa ongelmanratkaisuissa.

Tekoälymallit ovat hyödyllisiä koodin kirjoittamisessa ja ongelmanratkomisen avustuksessa. ChatGPT pystyy luomaan alkeellista ohjelmakoodia, koska monet suuret ohjelmat ovat useiden rivien ja tekstien pituisia ja tokenien määrä ei vielä riitä tuottamaan yhdellä viestillä ohjelmakokonaisuutta. ChatGPT:n kaltaiset verkossa toimivat tekoälypalvelut kehittyvät jatkuvasti nopeudeltaan ja myös tuoden enemmän tokeneita sekä uudempaa kerättyä dataa käyttäjille.

Tässä pilotissa on havaittu, että monet koodinpätkät, joita on saatu ChatGPT:ltä, ovat olleet virheellisiä tai eivät toimi lainkaan, vaan vaativat välillä suuria tai pieniä korjauksia. Myös useasti ChatGPT:tä joutuu huomauttamaan, että unohdit koodissa antamasi muuttujat tai arvot. Joskus on hyvä ChatGPT:n kanssa luoda täysin uusi viestiketju ja sinne viedä tuolloin toimiva koodi edellisestä viestiketjusta, että se unohtaisi jotkin asiat, mitkä se välttämättä haluaa lisätä takaisin koodiin, joka ei edes toimi tai tuhoaa koodin toimivuuden.

Positiivisesti tekoäly antaa nopeita pieniä ohjelmakoodeja, joita soveltamalla voidaan päästä nopeasti perille siitä, kuinka esimerkiksi käytetään Pythonille luotua PDFPlumber-kirjastoa ja kuinka nopeasti päästään käsiksi PDF-tiedoston raakadataan. Tekoäly ei opeta koodaamaan, ellei siltä pyydetä erikseen selitystä, miten sen tekemät koodit tai muokkaukset toimivat. Monesti korjausten yhteydessä se vain lisää ilman selitystä joitain tekstiä rivien väliin, tuoden ongelmia, jos ei ole kokemusta koodaamisesta tai debuggaamista.

Artikkeli on valmisteltu osana Datasta vauhtia valmistavan teollisuuden (VauhtiData) -hanketta, ja haluamme kiittää hankkeen ja tämän artikkelin rahoittamisesta Etelä-Savon elinkeino-, liikenne- ja ympäristökeskusta. Kyseinen hanke on Euroopan unionin osarahoittamana.

**Saku Kaarlejärvi**
asiantuntija, TKI
SeAMK

Saku Kaarlejärvi on tietotekniikan insinööri, joka on erikoistunut ohjelmointiin, tekoälyyn ja sulautettuihin järjestelmiin. Hänellä on kokemusta tuotannonohjauksen mittauksessa käytetyistä ratkaisuista, Power BI -raporttien rakentamisesta sekä älyteknologioiden mahdollisuuksien tutkimisesta hyvinvointi- ja terveysaloilla.

**Juha-Matti Arola**
projektipäällikkö, TKI
SeAMK

Juha-Matti Arola on kokenut projektipäällikkö, joka on erikoistunut EU-rahoitteisiin ESR-, JTF- ja EAKR-hankkeisiin, virtuaalisten työ- ja oppimisympäristöjen kehittämiseen sekä tiedolla johtamisen hyödyntämiseen. Hän toimii oman toimensa ohella SeAMKin asiakastiedonhallinnan, CRM:n, pääkäyttäjänä. Hän yhdistää vahvan teknologisen ja pedagogisen osaamisen tukemaan pk-yritysten ja aluekehityksen innovaatioita.

# Lähteet

Jsvine. (i.a) *pdfplumber*. https://github.com/jsvine/pdfplumber

Veale, K. (2023) MakeUseOf. *Does ChatGPT Have Privacy Issues?* https://www.makeuseof.com/chatgpt-privacy-issues/