

Tekoäly koodarikaverina

9.6.2025

Tekoäly on läsnä ja integroitunut tavalla tai toisella meidän jokapäiväiseen arkeemme. Tekoälytyökaluja on kehitetty jo vuosia, ja nyt varsinkin suurten kielimallien nopean kehityksen kanssa tekoälyn läsnäoloa on lisätty kaikkiin mahdollisiin sovelluksiin tai työkaluihin. Joistakin tekoälytyökaluista ei tunnu olevan hyötyä lainkaan, jotkin tekoälytyökalut ovat vain tunkeutuvia ja lisäävät potentiaalisia tietoturvariskejä.

Ennakoivaa tekstiä

Vibe Coding, eli “fiiliskoodailu” on nykyhetken trendi tekoälymaailmassa. Yksinkertaisesti tekoälylle lähetetään viestiä ja pyydetään koodia. Tekoälylle voidaan myös lähettää esimerkiksi kopioitua tai omaa kirjoitettua koodia. Tämän jälkeen voidaan kysyä tekoälyä tekemään haluttuja muutoksia koodiin ja hyvällä tuurilla tai onnistuneella promptauksella saadaan mahdollisesti toimivaa koodia.

Entä jos ei haluta käyttää tekoälypalveluita ja lähettää monien satojen tai tuhansien rivien lähdekoodia tietoturvasyistä? On olemassa ohjelmointiympäristösovelluksia tai niille kehitettyjä lisäosia, jotka pystyvät lukemaan ja auttamaan koodin kirjoittamisessa aktiivisesti ja paikallisesti. Continue.dev on lisäosa, joka on kehitetty Visual Studio Code:lle ja JetBrains ohjelmistoympäristöön ja jonka avulla pystytään käyttämään koodari apuna tekoälypalveluita ja myös paikallisia tekoälymalleja. (Continue.dev, i.e.)

Itse hyödynnän Continue.dev lisäosaa Visual Studio Codessa ja laitan omasta tottumuksesta sekä kokemuksesta LM Studion suorittamaan tekoälymalleja taustalle kattavan tekoälymallivalikoiman sekä niiden konfigurointien vuoksi. Ollama -palvelua suositellaan ja se on ensimmäisenä vaihtoehtona, kun lähdetään konfiguroimaan paikallista tekoälymallia Continue.dev lisäosan kanssa.

Tekstillä täytetään tyhjät rivit

Moni tekoälymalli osaa luoda tekstiä, mutta niitä ei ole tarkoitettu koodin luomiseen kesken kirjoittamisen. Fill-in-Middle tekoälymallit ovat parhaita tähän käyttötarkoitukseen. Fill-in-Middle on tarkoitettu täyttämään lauseita ennakoimalla tekstin “keskikohdan”. Yleensä tekoäly lähtee tuottamaan tekstiä sen lopusta, mutta FIM –mallit ovat koulutettu tiedoilla, jotka ovat pilkottu kolmeen eri osaan luoden tokenit “prefix”, “middle” ja “suffix” (Bavarian, Jun, Tezak, Schulman, McLeavey, Tworek & Chen, 2022, osio 1).

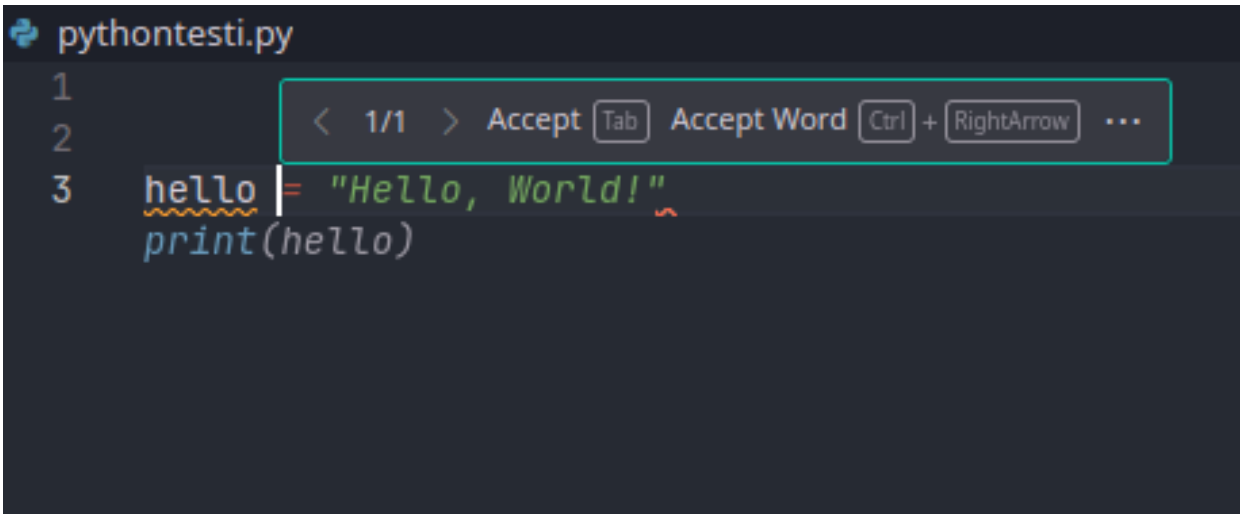
Pienikokoisia koodaritekoälymalleja ei ole suuresti saatavilla. Qwen2.5 Coder malli on ainoita uusimpia malleja, jotka ovat koulutettu ohjelmointia varten (Hugging Face, i.e.-a). Qwen3 on julkaistu huhtikuun loppupuolella ja se on tällä hetkellä uusin versio päättelykyvyllä koulutettu tekoälymalli (Huggingface.co, i.e.-b).

FIM ominaisuuden esittämisessä on käytetty tekoälymallia WhiteRabbitNeo 2.5 Qwen-2.5-Coder-7B, joka on epäsensuroitu Qwen2.5 Coder 7B –malli (Hugging Face. i.e.-c). Vähennettyyn sensuroinnin lisäksi tekoälymalliin on lisätty koulutusmateriaalia, mikä voi olla uudempaa dataa kuin alkuperäisessä mallissa.

```
4   models:
21    - name: AutoCompleteModel
22      provider: lmstudio
23      model: bartowski/whiterabbitneo-2.5-qwen-2.5-coder-7b
24      apiBase: http://localhost:1234/v1/
25      roles:
26      - chat
27      - edit
28      - autocomplete
29      - apply
30      defaultCompletionOptions:
31        maxTokens : 2048
32
33    - name: ChatToolUseModel
34      provider: lmstudio
35      model: bartowski/whiterabbitneo-2.5-qwen-2.5-coder-7b
36      apiBase: http://localhost:1234/v1/
37      roles:
38      - chat
39      - edit
40      - apply
41      capabilities:
42      - tool_use
43
44    - name: Nomic Embed
45      provider: lmstudio
46      model: nomic-ai/nomic-embed-text-v2-moe.Q8_0.gguf
47      roles:
48        - embed
```

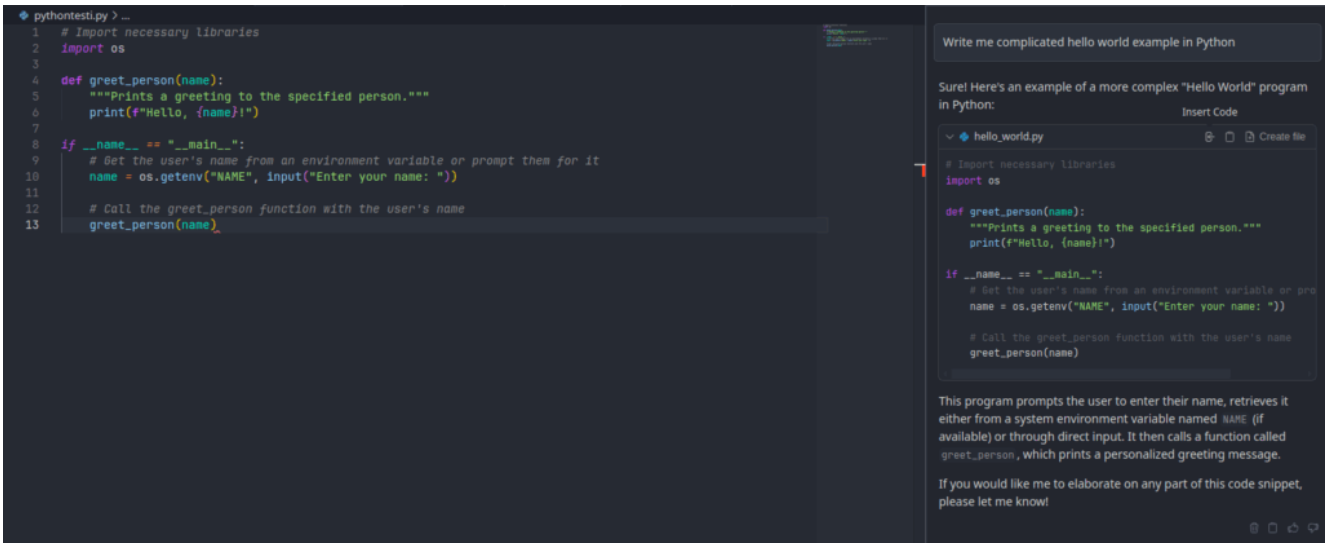
Kuva 1. Paikallisten tekoälymallien asetukset Continue.dev lisäosassa.

Kuvassa 1 on konfiguraatiotiedosto asetuksien vaihtamista varten. Pystyn tämän avulla määrittämään ennakoivan tekstinsyötön WhiteRabbitNeo-2.5 mallille, sekä erikseen voin myös valita toisen tekoälymallin Chat –ominaisuudelle. Toinen syy, miksi esimerkissä käytetään WhiteRabbitNeo 2.5:sta on sen ominaisuus tuottaa tekstiä ja viestitellä samaan aikaan. Ei tarvitse käyttää kahta eri mallia, jotka käyttävät enemmän näytönohjaimen tai keskusmuistin kapasiteettiä.



Kuva 2. Ennakoivaa tekstiä.

Kuvassa 2 Tekoälymalli ennakoivasti esitti, mitä mahdollisesti tullaan kirjoittamaan. Kuvasta 2. nähdään, että muutokset voidaan hyväksyä Tab -näppäimellä tai yksi sana kerrallaan käyttämällä Ctrl + oikeanuoli -näppäintä.



Kuva 3. Koodin generointi chatissä.

Continue.dev lisäosassa on mahdollista viestitellä tekoälymallin kanssa ja siirtämään sen tuottamaa koodia suoraan tekstieditoriin. Kuvassa 3 on kysytty tekoälymallilta monimutkaista “Hello world” esimerkki koodia. On myös mahdollista viedä tekstieditorista tekstiä tai koodia chattiin ja kysyä korjauksia tai selvittää virheitä. Maalamalla osoittimen avulla haluamansa alueen ja painamalla korostettua “Chat Ctrl + L” näppäinyhdistelmää kuvan 4. mukaisesti, pystytään siirtämään valittu teksti chattiin ja kysyä siltä tekemään

muutoksia tälle koodille.



Kuva 4. Tekstin vienti chattiin.

Tekoäly selittää tehdyt muutokset vaatimusten perusteella ja tämän jälkeen, pystytään siirtämää muokkaukset takaisin tekstieditoriin.

Tekoäly apuna vai ilman?

Tekoäly koodariapurina jäänyt tänne eikä lähde millään pois. Monet palvelut, kuten CoPilot, Mistral, Claude ja ChatGPT ovat kehittäneet tai tuomassa koodariapureita enemmän saataville lisäosina tai palveluina koodareille käyttäen API palveluita. Tekoälyavusteinen koodaus auttaa taklaamaan toistuvat koodausrutiinit esimerkiksi “loop” tai “if-else” –lausekkeiden logiikan kirjoittamisessa. Tekoäly myös nopeuttaa koodin kirjoitusta, sillä se lukee tekstiä aktiivisesti ja esimerkkinä pystyy ehdottamaan muutoksia, jotka sopivat oman koodin muuttujien kanssa. Tekoälymallista voi olla avuksi aloittelevalle koodarille tai koodaukselle kiinnostuneelle henkilölle. On myös tärkeää, että tietää itse koodaamisesta ja sen perusteista, eikä vain anna tekoälylle kaikkea vastuuta koodaamisesta. Koodin lukemisen ymmärtäminen on tärkeä taito ja on myös tärkeää kysyä koodari kaverilta, mitä muutoksia se on tehnyt.

vAI:lla tuottavuutta? -hanke on Euroopan unionin osarahoittama. Lisää tietoa hankkeesta löydät hankkeen verkkosivuilta https://projektit.app-staging.seamk.hion.dev/alykkaat-teknologiat/vailla-tuottavuutta/**Saku Kaarlejärvi**
asiantuntija, TKI
SEAMK

Saku Kaarlejärvi toimii asiantuntijana vAI:lla tuottavuutta? -hankkeessa ja on tietotekniikan insinööri, joka on erikoistunut ohjelmointiin, tekoälyyn ja sulautettuihin järjestelmiin.

Lähteet

Continue.dev. (i.e.) https://www.continue.dev/

Bavarian, M., Jun, H., Tezak, N., Schulman, J., McLeavey, C., Tworek, J., & Chen, M. (2022). Efficient

training of language models to fill in the middle. arXiv. https://arxiv.org/abs/2207.14255

Hugging Face. (i.e.-a). Qwen2.5-Coder-3B-Instruct-GGUF. https://huggingface.co/Qwen/Qwen2.5-Coder-3B-Instruct-GGUF

Hugging Face. (i.e.-b). Qwen3-8B. https://huggingface.co/Qwen/Qwen3-8B

Hugging Face. (i.e.-c). WhiteRabbitNeo-2.5-Qwen-2.5-Coder-7B. https://huggingface.co/bartowski/WhiteRabbitNeo-2.5-Qwen-2.5-Coder-7B-GGUF