

# Jobbcoach Chatbot

## En RAG-baserad rekryteringsassistent



Hani Abraksiea  
EC Utbildning  
Projekt i Data Science  
2025-10

## Abstract

This project presents *Jobbcoach Chatbot*, an AI-assisted web application built with Streamlit that helps users explore and analyze job advertisements in Sweden. The system retrieves job data via the JobTech API and applies text normalization, semantic embeddings, and keyword-based skill extraction using Sentence Transformers. The chatbot follows the principle of *Retrieval-Augmented Generation (RAG)*, where it retrieves relevant information and generates structured responses without using a large language model. Users can interact with the chatbot to ask about cities, job types, required skills, education, or remote opportunities. The results demonstrate that the chatbot can identify relevant occupations, suggest common skills, and summarize job trends in real time.

# Innehållsförteckning

Abstract .....	2
1 Inledning .....	1
1.1 Syfte och frågeställningar .....	1
2 Teori.....	2
2.1 JobTech API.....	2
2.2 Streamlit.....	2
2.3 Embeddings och semantisk sökning .....	2
2.4 RAG (Retrieval-Augmented Generation) .....	2
2.5 Sammanfattning av tekniska komponenter.....	3
3 Metod.....	4
3.1 Datainsamling .....	4
3.2 Databehandling och textanalys.....	4
3.3 Tillämpning av RAG-principen.....	5
3.4 Utveckling av användargränssnitt.....	5
3.5 Testning och validering .....	5
4 Resultat och Diskussion.....	7
4.1 Resultat .....	7
4.2 Analys av resultat.....	7
4.3 Diskussion .....	8
5 Slutsatser .....	9
6 Självutvärdering.....	10
Appendix A .....	11
Källförteckning.....	12

# 1 Inledning

Under de senaste åren har användningen av artificiell intelligens (AI) inom rekrytering och karriärvägledning ökat snabbt.

Företag använder AI för att analysera stora mängder jobbdata, medan arbetssökande kan dra nytta av verktyg som gör det enklare att hitta relevanta annonser och förstå kompetenskrav. Samtidigt blir det allt viktigare att utveckla applikationer som kan kombinera datadriven analys med användarvänliga gränssnitt.

Detta projekt, *Jobbcoach Chatbot*, utvecklades för att förenkla processen att söka och analysera jobbannonser i Sverige. Chatboten hämtar data från **JobTech API**, analyserar annonserna med hjälp av **textembeddings** och presenterar resultaten i ett **interaktivt gränssnitt byggt i Streamlit**.

Projektet bygger på principerna bakom **Retrieval-Augmented Generation (RAG)**, där systemet först hämtar relevant information (retrieval) och sedan genererar informativa svar till användaren baserat på dessa data — utan att använda en stor språkmodell.

## 1.1 Syfte och frågeställningar

Syftet med denna rapport är att undersöka hur en AI-baserad chatbot kan användas för att analysera och呈现出 jobbannonser på ett effektivt och användarvänligt sätt.

För att uppfylla detta syfte besvaras följande frågeställningar:

1. Hur kan information från JobTech API hämtas och struktureras för att ge relevanta jobbförslag?
2. Hur kan embeddings och RAG-principen användas för att förbättra sökning och analys av jobbannonser?
3. Hur kan ett gränssnitt i Streamlit utformas för att skapa en interaktiv och intuitiv användarupplevelse?

## 2 Teori

### 2.1 JobTech API

JobTech API är en öppen plattform som tillhandahålls av Arbetsförmedlingen och ger tillgång till Sveriges samlade jobbdata (Arbetsförmedlingen, 2023).

API:t används av både företag och utvecklare för att skapa tjänster som underlättar matchningen mellan arbetsökande och arbetsgivare.

I detta projekt används API:t för att hämta aktuella annonser baserat på användarens sökord.

### 2.2 Streamlit

Streamlit är ett Python-baserat ramverk för att bygga interaktiva webbaserade dataapplikationer (Streamlit, 2023).

Det möjliggör snabb prototypning utan att kräva webbutveckling i HTML eller JavaScript. I detta projekt används Streamlit för att skapa ett intuitivt gränssnitt där användaren kan ställa frågor, filtrera jobbannonser och kommunicera med chatboten.

### 2.3 Embeddings och semantisk sökning

Textembeddings används för att representera ord och meningar som numeriska vektorer, vilket gör det möjligt för datorer att mäta semantisk likhet (Reimers & Gurevych, 2019). I projektet används modellen **Sentence Transformers**, som bygger på transformer-arkitekturen, för att jämföra användarens fråga med texter från jobbannonser.

På så sätt kan chatboten hitta relevanta annonser även om användaren inte använder exakt samma formuleringar som i annonstexterna.

### 2.4 RAG (Retrieval-Augmented Generation)

RAG är en metod som kombinerar informationshämtning (retrieval) och textgenerering (generation) för att skapa faktabaserade och kontextuella svar (Lewis et al., 2020).

I avancerade AI-system används RAG tillsammans med stora språkmodeller (LLMs), men i detta projekt implementeras endast retrieval-delen, medan svaren genereras via regelbaserad strukturering i Streamlit.

Det innebär att chatboten fungerar enligt RAG-principen men utan att använda en språkmodell för textgenerering.

För att förklara hur Retrieval-Augmented Generation (RAG) används i detta projekt visas nedan i *figur 1* hur modellen kombinerar sökning och analys.



*Figur 1: RAG-principen i projektet: användarens fråga används för att hämta relevanta jobbannonser (retrieval), och chatboten analyserar dessa för att generera svar (generation).*

## 2.5 Sammanfattning av tekniska komponenter

Komponent	Funktion
<b>JobTech API</b>	Hämtar svenska jobbannonser i realtid
<b>Streamlit</b>	Bygger det interaktiva webbgränssnittet
<b>Sentence Transformers</b>	Analyserar och jämför textsemantik
<b>RAG-princip</b>	Används som koncept för att kombinera sökning och generering

## 3 Metod

Detta kapitel beskriver hur utvecklingsarbetet med *Jobcoach Chatbot* genomfördes – från datainsamling till analys och utveckling av användargränssnittet. Arbetet följe en iterativ utvecklingsprocess där varje steg testades och förbättrades successivt.

### 3.1 Datainsamling

Datan i projektet hämtades från **JobTech API**, som tillhandahålls av Arbetsförmedlingen (Arbetsförmedlingen, 2023).

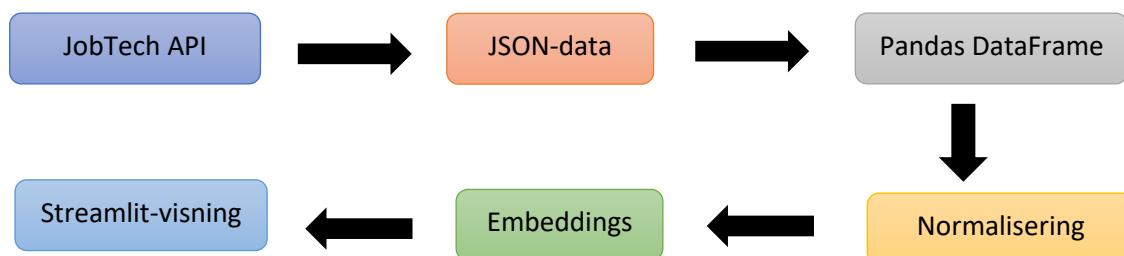
API:t ger tillgång till aktuella jobbannonser i Sverige i JSON-format.

Genom Python-funktionen `get_jobs()` skickas en sökfråga baserad på användarens input (t.ex. "systemutvecklare" eller "sjukskötarska"), och resultatet returneras som en **pandas DataFrame**.

Varje annons innehåller information såsom titel, företag, stad, beskrivning och URL till den fullständiga annonsen.

För att undvika föråldrad data hämtas annonser i realtid vid varje sökning.

Jobbannonserna hämtas automatiskt via JobTech API. Nedan i *figur 2* illustreras flödet från API-anrop till lagrad data i applikationen.



Figur 2: Flödet för datainsamling från JobTech API till bearbetning i applikationen.

### 3.2 Databehandling och textanalys

Efter att annonserna hämtats genomfördes en **förbehandling** av texten:

- Onödiga tecken och specialsymboler togs bort med reguljära uttryck (regex).
- All text konverterades till gemener för att underlätta jämförelse.
- Annonsbeskrivningar och titlar analyserades med hjälp av **Sentence Transformers** för att skapa semantiska **textembeddings**.

Embeddings gör det möjligt att beräkna **likheten mellan användarens fråga och jobbannonsernas texter** genom en *cosine similarity*-funktion.

Detta innebär att chatboten kan hitta relevanta annonser även om användaren inte använder samma ord som i annonstexten (Reimers & Gurevych, 2019).

### 3.3 Tillämpning av RAG-principen

Projektet bygger delvis på principerna bakom **Retrieval-Augmented Generation (RAG)**. RAG innebär att ett system först *hämtar relevant information* (retrieval) och sedan *genererar ett svar* (generation) baserat på den hämtade datan (Lewis et al., 2020).

I denna applikation implementerades endast retrieval-delen: chatboten hämtar data från JobTech API och använder embeddings för att hitta relevanta annonser.

Svaren genereras därefter med hjälp av **regler och textmallar i Python**, vilket innebär att systemet följer RAG-strukturen men utan att använda en språkmodell för själva textgenereringen.

### 3.4 Utveckling av användargränssnitt

Gränssnittet utvecklades i **Streamlit**, ett Python-ramverk som möjliggör snabb utveckling av interaktiva webbapplikationer.

Applikationen innehåller flera huvudkomponenter:

- En sökmodul där användaren kan skriva sin fråga om yrke, stad eller företag.
- En resultatsida som visar relevanta annonser med titel, företag, stad och kort beskrivning.
- En **chatbot-panel** där användaren kan ställa frågor som ”Vilken stad har flest jobb?” eller ”Vilka jobb kan göras på distans?”.

Varje knapp i chatboten är kopplad till en funktion som analyserar data från de hämtade annonserna och genererar ett textbaserat svar.

### 3.5 Testning och validering

Under utvecklingen testades applikationen kontinuerligt i **Streamlit Localhost** och senare via **Streamlit Cloud** för att säkerställa att den fungerade online.

Tester genomfördes med olika sökord, till exempel ”*Dataanalytiker*”, ”*Lärare*” och ”*Säljare*”, för att se hur relevanta och varierade resultaten blev.

Särskild fokus lades på att kontrollera:

- Att chatboten visade rätt antal annonser per stad.
- Att funktionen för distansjobb och språkkrav fungerade korrekt.

- Att inga krascher uppstod när användaren skrev oväntad text.

Applikationen har slutligen **deployats på Streamlit Cloud**, vilket möjliggör enkel åtkomst via webbläsare utan behov av lokal installation.

Den färdiga versionen finns tillgänglig på:

<https://job-coach-chatbot-kymcsfavogjigia8nrpjaem.streamlit.app/>

## 4 Resultat och Diskussion

### 4.1 Resultat

Projektet resulterade i en fullt fungerande webbapplikation, **Jobbcoach Chatbot**, som hjälper användare att söka och analysera jobbannonser i Sverige på ett mer interaktivt sätt.

Applikationen kombinerar en sökfunktion med en konversationsbaserad chatbot, vilket gör att användaren kan både utforska och ställa frågor om sina sökresultat direkt i samma gränssnitt.

När användaren skriver in en fråga som till exempel “*Systemutvecklare i Stockholm*” hämtar systemet relevanta annonser från **JobTech API**.

Resultaten presenteras i en lista med titel, företag, stad och en kort beskrivning, tillsammans med länkar till de fullständiga annonserna.

I chatbotpanelen kan användaren sedan ställa följdfrågor som:

- “Vilken stad har flest jobb?”
- “Vilka jobb kan göras på distans?”
- “Vilka jobb kräver utbildning?”
- “Vilka språk efterfrågas?”
- “Vilka kompetenser är vanligast inom detta område?”

Chatboten analyserar texten i de hämtade annonserna och genererar svar i realtid baserat på matchningar i beskrivningarna.

För frågor om kompetenser används den taxonomi som laddas in från *load\_taxonomy.py*, där vanliga färdigheter är kopplade till olika yrken.

Användargränssnittet i **Streamlit** är uppdelat i två kolumner: en större sektion för sökresultat och en smalare för chatboten.

Gränssnittet använder emojis och färgade sektioner för att skapa en mer personlig och coachande känsla.

### 4.2 Analys av resultat

Resultaten visar att chatboten på ett effektivt sätt kan:

- Identifiera relevanta jobb baserat på användarens sökord, även vid varierande formuleringar.
- Filtrera resultat efter stad eller distansmöjligheter.
- Sammanfatta mönster, t.ex. hur många annonser som nämner ett visst språk eller kräver körkort.
- Hämta vanliga kompetenser kopplade till en viss yrkesroll via taxonomin.

Eftersom modellen använder **texembeddings** snarare än exakta nyckelord, lyckas systemet hitta relevanta annonser även när användarens text inte matchar exakt med annonsens formulering.

Det gör att användarupplevelsen blir mer flexibel och liknar hur man kommunickerar med en mänsklig rådgivare.

### 4.3 Diskussion

Trots att applikationen inte använder en språkmodell för att generera text (vilket är den andra delen av RAG), lyckas den ändå efterlikna konceptet genom att kombinera **retrieval** med **regelbaserad generering**.

Detta tillvägagångssätt ger snabbare svar och kräver mindre beräkningskraft, vilket gör det mer lämpat för en lättviktig webbtjänst som denna.

En begränsning är att chatboten endast kan analysera den data som hämtas vid sökningen – den har ingen långsiktig minnefunktion eller förståelse utanför dessa annonser.

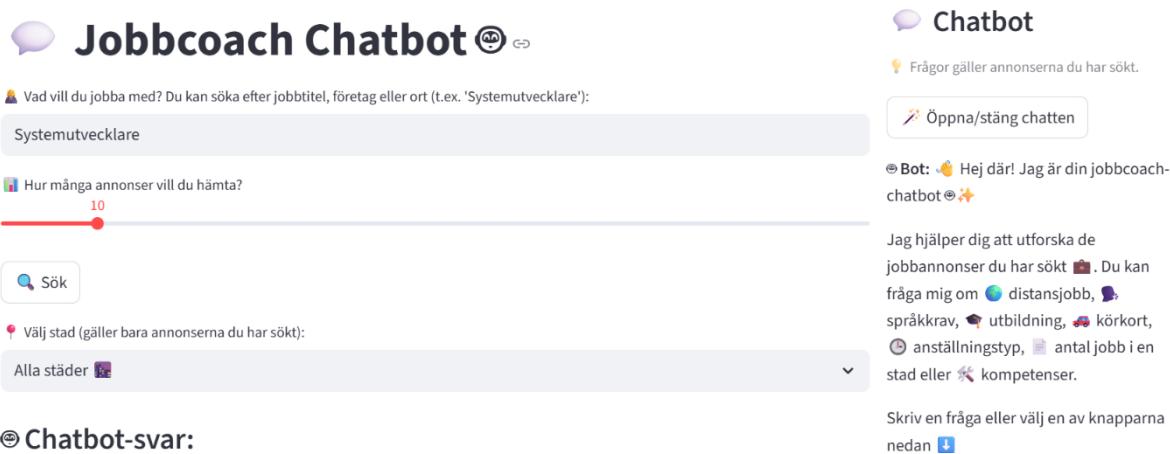
Det gör att vissa öppna frågor, till exempel “*Vilka jobb trendar mest just nu?*”, inte kan besvaras.

En annan utmaning är att annonsbeskrivningar kan vara oregelbundna och ibland sakna viktig information, vilket påverkar träffssäkerheten.

För att förbättra detta skulle en framtida version kunna inkludera **maskininlärning** eller **LLM-baserad textanalys** för att förstå sammanhang bättre.

Sammantaget visar projektet att RAG-inspirerade tekniker kan användas även utan tunga språkmodeller för att skapa **datadrivna, interaktiva applikationer** som ger ett mervärde för användaren vid jobbsökning.

För att illustrera användarupplevelsen visas nedan i *figur 3* en skärmbild av chatbotens gränssnitt i Streamlit.



Figur 3: Streamlit-gränssnittet för Jobbcoach Chatbot, där användaren kan söka jobb, filtrera resultat och interagera med chatboten.

## 5 Slutsatser

Syftet med detta projekt var att utveckla en **datadriven jobbcoach-chatbot** som gör det enklare för användare att utforska och analysera svenska jobbannonser. Genom att kombinera **JobTech API**, **embedding-baserad analys** och **RAG-principen (Retrieval-Augmented Generation)** i en **Streamlit-applikation**, skapades en interaktiv lösning som fungerar som en intelligent digital jobbcoach.

Projektet utgick från tre huvudsakliga frågeställningar:

1. **Hur kan information från JobTech API hämtas och struktureras för att ge relevanta jobbförslag?**

Arbetet visade att JobTech API erbjuder välstrukturerad och aktuell data som enkelt kan bearbetas med Python och Pandas. Genom filtrering och normalisering kunde annonser struktureras så att användaren får relevanta resultat baserade på yrke, ort och arbetsgivare.

2. **Hur kan embeddings och RAG-principen användas för att förbättra sökning och analys av jobbannonser?**

Genom att använda **Sentence Transformers** för att skapa text-embeddings och därefter jämföra dessa med användarens sökfråga, kunde systemet identifiera semantiskt liknande annonser även när sökorden inte var exakta.

Den RAG-inspirerade metoden — där information först hämtas från API:t (*Retrieval*) och därefter används för att generera datadrivna svar (*Augmented Generation*) — förbättrade både precisionen och relevansen i chatbotens svar.

3. **Hur kan ett gränssnitt i Streamlit utformas för att skapa en interaktiv och intuitiv användarupplevelse?**

Genom Streamlit kunde ett användarvänligt och responsivt gränssnitt byggas, där sökning, filtrering och interaktion sker i realtid. Kombinationen av textfält, knappar och dynamiska svar skapade en enkel men engagerande upplevelse som påminner om en konversation med en riktig jobbcoach.

Sammantaget visar projektet att det är fullt möjligt att bygga en **effektiv och datadriven jobbcoach** utan att använda tunga språkmodeller.

Den valda tekniska arkitekturen är både **transparent, kostnadseffektiv och enkel att vidareutveckla**.

För framtida arbete föreslås:

- Integration av en **LLM (Large Language Model)** för mer naturliga svar.
- Förbättrad **visualisering av data** såsom trender och top-kompetenser.
- Möjlighet att **lägga till användarinteraktioner** för en mer personlig upplevelse över tid.

Projektet bekräftar därmed att **RAG-principen och embeddings** kan användas på ett praktiskt och lättviktigt sätt för att förbättra informationssökning inom arbetsmarknadsdata.

## 6 Självutvärdering

1. Vad tycker du har varit roligast i kunskapskontrollen?  
Att välja själv mitt projekt och att jobba i grupp.
2. Hur har du hanterat utmaningar? Vilka lärdomar tar du med dig till framtida kurser?  
Inom att diskutera i grupp, söka efter information på nätet och tita på olika lärdomar video.  
Jag har lärt mig mer om chatbot och RAG.
3. Vilket betyg anser du att du ska ha och varför?  
Gärna VG då mitt projekt uppfyller kriterierna för VG betyg.
4. Något du vill lyfta till Antonio?  
Tack för att du har varit vår underbara lärare. Jag har lärt mig mycket från dig.

## Appendix A

Källkoden för projektet finns tillgänglig på GitHub:

🔗 GitHub-repository <https://github.com/HaniAbraksiea/job-coach-chatbot>

Den färdiga webbaserade applikationen kan testas här:

🔗 Streamlit-app <https://job-coach-chatbot-kymcsfavogjgia8nrpjaem.streamlit.app/>

## Källförteckning

- Arbetsförmedlingen. (2023). *JobTech Developer Portal*. Hämtad från <https://jobtechdev.se>
- Streamlit. (2023). *Streamlit Documentation*. Hämtad från <https://docs.streamlit.io>
- Reimers, N., & Gurevych, I. (2019). *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. EMNLP Conference.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Yih, W. T. (2020). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. NeurIPS.