# AI Chatbot

## Powered by LlaMa 3


Magesh Babu Madhana Gopal

EC Utbildning

ProjectKursen

2024-12

# Abstract

In this project, the aim is to develop a Large Language Model (LLM)-powered chatbot application designed to provide accurate and context-aware responses by leveraging external knowledge. To achieve this, we selected Llama 3 8B Instruct, an open-source LLM developed by Meta, known for its advanced instruction-following capabilities and scalability.

To enhance the chatbot's performance, we incorporated a Retrieval-Augmented Generation (RAG) approach, utilizing LlamaIndex for data ingestion and vector store indexing. We employed Hugging Face's embedding model to convert textual data into vector representations, allowing efficient retrieval of relevant information. A streamlit-based interface was developed to provide a seamless, interactive user experience, enabling users to interact with the chatbot in real-time.

The Llama 3 8B Instruct model, deployed on Azure, ensures high scalability, reliability, and performance. By combining these components, we have built a robust chatbot capable of generating contextually accurate and informative responses, effectively integrating external knowledge to enhance the user experience.

# Innehållsförteckning

# 1 Inledning

I dagens snabba affärslandskap håller artificiell intelligens (AI) på att bli en integrerad del av nästan alla branscher. Företag anammar alltmer AI-teknik för att förbättra sina produkter och effektivisera verksamheten, i syfte att utnyttja dess potential för att driva effektivitet och innovation. Ett område där AI har en betydande inverkan är bearbetningen och förståelsen av naturligt språk. Företag, särskilt de som hanterar stora volymer dokument, investerar ofta avsevärda resurser i att hantera och utvinna värdefulla insikter från deras innehåll.

Large Language Models (LLMs), en kärnaspekt av AI, har revolutionerat hur företag interagerar med textdata. Dessa modeller möjliggör avancerade textbehandlingsfunktioner, automatiserar uppgifter som innehållsextraktion, frågesvar och dokumentanalys. Med tanke på det växande beroendet av dokument och databaser för affärskritisk information, antas LLM alltmer för att förbättra produktiviteten och kundservicen.

Detta projekt fokuserar på att ta itu med den viktiga affärsutmaningen är att optimera hanteringen av textdata genom AI-drivna lösningar. Specifikt strävar vi efter att bygga en chatbot som utnyttjar externa dokument eller uppladdat innehåll för att ge korrekta, sammanhangsmedvetna svar. Genom att integrera dokumentbaserade AI-svar kommer denna chatbot att göra det möjligt för företag att effektivt hantera kundförfrågningar, få tillgång till relevant information från stora dokument och minska den manuella ansträngning som krävs för att bearbeta och extrahera användbar data. Genom detta tillvägagångssätt försöker vårt projekt demonstrera den transformativa potentialen hos AI inom dokumenthantering och kundinteraktion, vilket erbjuder betydande fördelar för företag som vill förbättra operativ effektivitet och kundupplevelse. Figur 1 beskriver var LLM placeras i hela konceptet för AI.

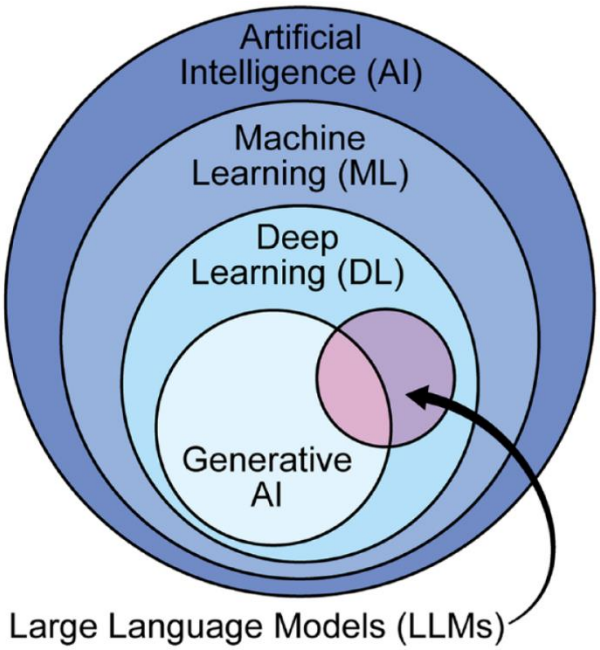


*Figure 1 - AI Division*

# 2 Teori

## 2.1 Large Language Model (LLM)

Large Language Models (LLM) är avancerade system för artificiell intelligens designade för att utföra ett brett spektrum av naturliga språkbehandlingsuppgifter (NLP). Dessa uppgifter inkluderar textgenerering, översättning, sammanfattning, sentimentanalys, kodgenerering och mer. LLM:er byggs med hjälp av djupinlärningstekniker och utnyttjar vanligtvis en transformatorarkitektur, som gör att de kan bearbeta och förstå komplexa mönster i stora datamängder.

LLM:er utbildas på stora mängder textdata, hämtade från olika domäner som böcker, artiklar, kodarkiv och webbinnehåll. Detta gör det möjligt för dem att fånga semantiska, syntaktiska och kontextuella relationer mellan ord och fraser. Till exempel kan en LLM urskilja om ordet "rätt" hänvisar till en riktning, korrekthet eller juridiska rättigheter baserat på dess sammanhang. (cloud flare, u.d.)

**Fördelar:** Large language models (LLM) erbjuder exceptionella problemlösningsmöjligheter över ett brett spektrum av applikationer genom att presentera information på ett tydligt och konversationssätt. Deras mångsidighet gör att de kan utföra uppgifter som språköversättning, sentimentanalys, svara på frågor och lösa matematiska problem. När LLM:er bearbetar mer data och införlivar ytterligare parametrar, förbättras deras prestanda kontinuerligt, och drar nytta av "in-context learning", där modeller lär sig nya uppgifter från minimala exempel utan ytterligare utbildning. Detta gör det möjligt för dem att anpassa och förbättra snabbt med färre resurser, vilket visar deras effektivitet och förmåga att snabbt lära sig av kontextdrivna uppmaningar.

**Begränsning:** Large language models (LLM) är kraftfulla tekniska verktyg, men de möter flera begränsningar och utmaningar som påverkar deras noggrannhet, tillförlitlighet och etiska användning. En viktig fråga är hallucinationer, där en LLM genererar utdata som är falska eller inte relaterade till användarens avsikt. Till exempel kan en LLM felaktigt hävda att han besitter känslor eller fabricerar information, eftersom den förutsäger nästa sannolika ord utan att helt förstå mänsklig mening. Dessa hallucinationer uppstår eftersom LLM:er förlitar sig på syntaktiska mönster snarare än sann förståelse.

Ett annat problem är säkerheten, eftersom felaktig hantering av LLM kan leda till dataläckor, nätfiske och spridning av felaktig information. Skadliga användare kan utnyttja dessa modeller för att sprida partiska ideologier eller skapa spam, vilket kan få långtgående konsekvenser. På samma sätt är partiskhet en vanlig fråga. Uppgifterna som används för att utbilda LLM återspeglar samhälleliga och demografiska skillnader; sålunda kan utdata sakna mångfald eller vidmakthålla stereotyper, vilket äventyrar deras rättvisa och inkluderande.

Samtycke och etisk användning av data innebär också utmaningar. LLM:er utbildas på omfattande datauppsättningar, ofta skrapade från internet utan lämpliga behörigheter. Detta väcker farhågor om kränkningar av immateriella rättigheter, vilket kan ses i stämningar som Getty Images fall mot AI-missbruk.

Slutligen kräver skalningen och distributionen av LLM enorma resurser och teknisk expertis. Att träna dessa modeller är tidskrävande och kräver avancerad hårdvara, distribuerade system och specialiserad kunskap. Trots deras anmärkningsvärda kapacitet framhäver dessa utmaningar vikten av noggrann hantering, etisk tillsyn och ständiga förbättringar vid utveckling och implementering av LLM. (Elastic, u.d.)

### 2.1.1 Transformer Architecture

Transformatorarkitektur är en hörnsten i modern naturlig språkbehandling (NLP) och djupinlärning, vilket revolutionerar hur modeller behandlar sekventiell data som text. Introducerats i den framstående artikeln "Attention Is All You Need" av Vaswani et al. 2017 introducerade den konceptet självuppmärksamhet, vilket gör att modeller kan fokusera på relevanta delar av inmatningssekvenser, vilket möjliggör parallell bearbetning och förbättrad kontextförståelse. Figur 2 illustrerar transformatornätverkets arkitektur.

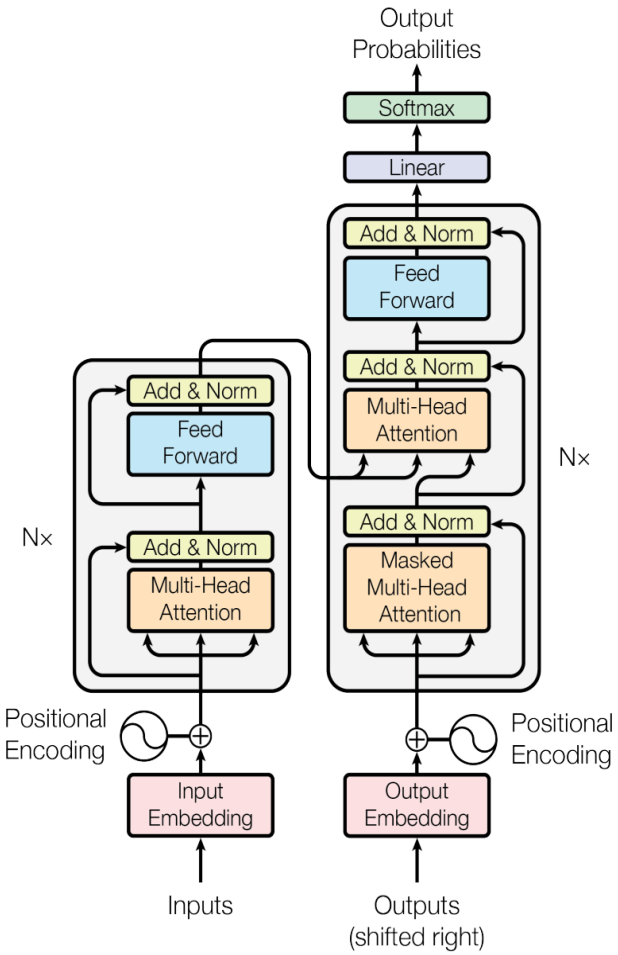


*Figure 2 - Transformer Architecture*

Transformatorer utgör grunden för stora språkmodeller (LLM) som GPT, BERT och Metas LLaMA, vilket driver framsteg inom översättning, sammanfattning och generativa uppgifter. De exemplifierar övergången till arkitekturer som prioriterar sammanhang och beräkningseffektivitet, vilket sätter scenen för moderna AI-genombrott.

Nyckelkomponenterna i transformatormodellen är ingångsrepresentation, självuppmärksamhetsmekanism, multi-head uppmärksamhet, feedforward neurala nätverk, normalisering och kvarvarande anslutning, och kodare och avkodare struktur.

**Input Representation**: Indata, som text, tokeniseras till mindre enheter (ord eller underord) och representeras som inbäddningar i ett högdimensionellt utrymme. Positionell kodning läggs till inbäddningar för att behålla sekvensinformationen eftersom transformatorer saknar inneboende ordningsmedvetenhet.

**Self-Attention Mechanism**: Självuppmärksamhet beräknar hur mycket fokus varje token i sekvensen ska ge varannan token. Detta görs med hjälp av tre matriser härledda från indata:

- **Query (Q)**: Representerar den aktuella token.
- **Nyckel (K)**: Kodar relevansen för alla tokens.
- **Värde (V):** Kodar den kontextuella betydelsen av alla tokens.

Uppmärksamhetspoängen beräknas genom att ta en punktprodukt av frågan och nyckelmatriserna, normalisera med softmax och multiplicera med värdematrisen. Denna mekanism gör det möjligt för modellen att överväga relationer över hela sekvensen samtidigt.

**Multi-Head Attention**: Istället för att utföra en enda uppmärksamhetsberäkning använder transformatorn flera uppmärksamhets-"huvuden" som var och en lär sig olika relationer och funktioner. Dessa huvuden fungerar parallellt, vilket ger en rikare förståelse för input.

**Feedforward Neural Networks**: Efter uppmärksamhetslager bearbetas tokens genom helt anslutna feedforward-lager för att lära sig komplexa mönster. Dessa appliceras oberoende på varje token och är avgörande för att omvandla kontextuell information till användbara funktioner.

**Normalization and Residual Connections**: Varje lager använder lagernormalisering för att stabilisera inlärning och kvarvarande anslutningar för att bekämpa försvinnande gradienter och förbättra gradientflödet under träning.

**Encoder-Decoder Structure**: Transformatorn är uppdelad i Encoder och Decoder. Encoder bearbetar inmatningssekvenser för att generera kontextmedvetna representationer. Avkodaren använder dessa representationer för att generera utdatasekvenser (t.ex. översättningar, textkompletteringar). Kodaren och avkodaren består båda av staplade lager av uppmärksamhets- och feedforward-komponenter. *(Ashish Vaswani, 2017)*

## 2.2 Retreival Augumented Generation (RAG)

RAG (Retrieval-Augmented Generation) är ett banbrytande ramverk inom naturlig språkbehandling som kombinerar två kritiska AI-funktioner: hämtning och generering. Den utnyttjar styrkorna hos stora språkmodeller (LLM) samtidigt som den övervinner deras begränsningar genom att införliva externa kunskapskällor.

RAG förbättrar prestandan hos generativa modeller genom att utöka deras utdata med faktisk noggrannhet och domänspecifik relevans. Till skillnad från fristående språkmodeller, som helt och hållet förlitar sig på förutbildade parametrar och kan hallucinera felaktig information, integrerar RAG extern data dynamiskt under svarsgenereringsprocessen. Figur 3 förklarar hur extern data matas in i LLM. (AWS Amazon, u.d.)

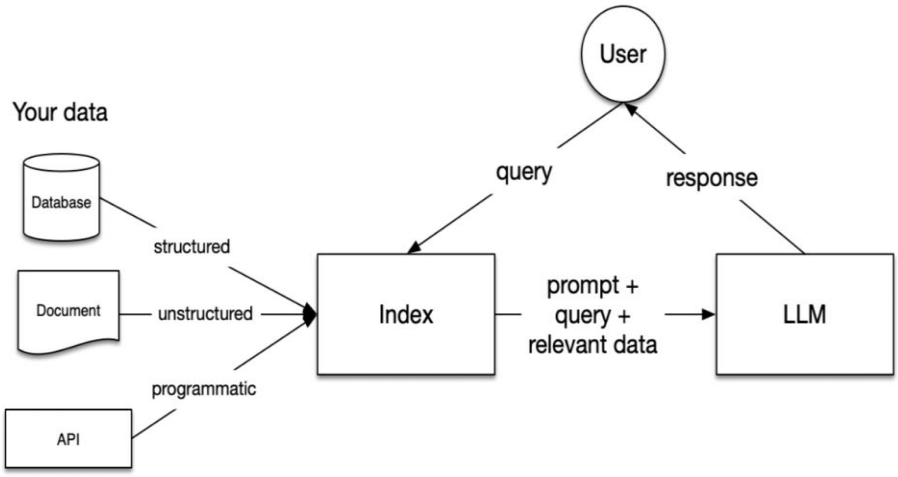


*Figure 3 - RAG Concept*

Processen börjar med skapandet av extern data, ett arkiv med information utanför LLM:s utbildningsdata. Dessa data kan komma från API:er, databaser, dokumentarkiv eller andra källor och kan finnas i olika format som textfiler, databasposter eller detaljerade dokument. Data konverteras sedan till numeriska representationer (vektorer) med hjälp av inbäddningstekniker och lagras i en vektordatabas, vilket bildar ett kunskapsbibliotek tillgängligt för den generativa AI-modellen.

När en användare skickar en fråga är nästa steg att hämta relevant information. Inmatningsfrågan konverteras till en vektor och matchas mot de lagrade vektorerna i databasen för att identifiera den mest relevanta informationen. Till exempel, i ett chatbot-scenario på arbetsplatsen, om en anställd frågar om årlig semester, kan systemet hämta policydokument och den anställdes ledighetsuppgifter baserat på de matematiska beräkningarna av relevans. Detta säkerställer att chatboten ger ett skräddarsytt svar snarare än ett generiskt.

Den hämtade informationen används sedan för att utöka LLM:s uppmaning. Genom snabb konstruktion förbättras användarens input genom att lägga till den hämtade kontextuella informationen, vilket gör att modellen kan generera ett svar som är både korrekt och kontextuellt informerat. Denna process överbryggar gapet mellan statisk förtränad kunskap och dynamisk realtidsinformation.

För att hålla den externa informationen korrekt och relevant är det viktigt att uppdatera kunskapsbasen med jämna mellanrum. Detta kan uppnås genom realtidsuppdateringar eller batchbearbetning, vilket säkerställer att de inbäddade representationerna av dokument förblir aktuella. Det här steget är avgörande i scenarier där informationen ändras ofta, till exempel regulatoriska dokument, produktmanualer eller finansiella register.

**Fördelar**: Retrieval-Augmented Generation (RAG) erbjuder betydande affärsfördelar, särskilt genom att öka produktiviteten och operativ effektivitet. Genom att möjliggöra snabb åtkomst till relevant information i realtid från externa källor, ger RAG användare möjlighet att hämta data direkt utan att vada genom massiva datamängder. Detta påskyndar beslutsfattandet, minskar arbetsbelastningen för anställda och gör att de kan fokusera på strategiska prioriteringar. Dessutom, genom att jorda utdata i faktiska, väl anskaffade data, minimerar RAG fel och fördomar, vilket förbättrar noggrannheten och tillförlitligheten hos informationen, vilket skapar större förtroende bland användarna.

Dessutom leder RAG till kostnadsbesparingar och förbättrat resursutnyttjande. Med sin förmåga att dynamiskt komma åt relevant data kan organisationer minska behovet av omfattande finjusteringar av stora språkmodeller (LLM). Detta sänker kostnaderna för utbildning och modelloptimering. Genom att automatisera kunskapsintensiva uppgifter gör RAG dessutom att IT-resurser kan omdirigeras till andra kritiska områden. I slutändan resulterar detta i bättre kundupplevelser eftersom RAG-baserade applikationer ger snabba, exakta svar, förbättrar kundnöjdheten och främjar positiva interaktioner. (Cole, u.d.)

# 3 Metod

I det här projektet använder vi konceptet Retrieval-Augmented Generation (RAG) för att förbättra prestandan hos en chatbot genom att göra det möjligt för den att generera svar baserade på externa datakällor. Integrationen av RAG med stora språkmodeller (LLM) ger chatboten möjlighet att ge kontextuellt relevanta och korrekta svar genom att införliva extern kunskap i sina svar. För att implementera vårt projekt med detta tillvägagångssätt på ett effektivt sätt krävs en kombination av verktyg och ramverk, vi använder en uppsättning specialiserade verktyg och ramverk för att ta itu med följande aspekter:

- Datavektorisering
- Effektiv förfrågning av externa kunskapsbaser
- Utveckling av användargränssnitt
- Modellutbyggnad

## 3.1 Llama 3 8B Instruct

För att skapa en Large language models (LLM)-driven chatbotapplikation är det avgörande att välja rätt LLM. Bland de många tillgängliga alternativen valde vi Llama 3 8B Instruct, en öppen källkod LLM utvecklad av Meta.

Llama 3 8B Instruct är en banbrytande stor språkmodell tränad på 8 miljarder parametrar, vilket markerar ett betydande steg i språkmodellernas kapacitet. Det ökade antalet parametrar förbättrar dess förmåga att fånga komplexa språkmönster och kontextuella nyanser, vilket möjliggör mer exakta och sammanhangsmedvetna svar.

Llama 3 8B Instruct har finjusterats på storskaliga instruktionsbaserade datauppsättningar, som fokuserar på uppgifter som att svara på frågor, tillhandahålla förklaringar, sammanfattningar och utföra språkdrivna uppgifter med hög effektivitet. Beteckningen "Instruera" belyser dess fokus på att noggrant följa detaljerade instruktioner, vilket säkerställer att genererade svar är i linje med användarnas förväntningar. Detta gör den särskilt väl lämpad för applikationer som chatbots, Q&A-system och konversations-AI.

Llama 3 8B Instruct bygger på de grundläggande styrkorna hos tidigare modeller som Llama 2 men innehåller specifika förbättringar som är skräddarsydda för instruktioner som följer uppgifter, vilket förbättrar dess användbarhet i chatbot-applikationer. Dess förmåga att bibehålla sammanhang över flera interaktioner, kombinerat med dess instruktionsjustering, säkerställer sammanhängande och korrekta svar, vilket gör det till ett kraftfullt val för applikationer som kräver komplex, sammanhangsmedveten kommunikation.

## 3.2 Verktyg och ramar

### 3.2.1 HuggingFace Inbäddningsmodell

Hugging Face är en AI-plattform och community med öppen källkod som specialiserar sig på Natural Language Processing (NLP) och maskininlärning. Det tillhandahåller ett bibliotek med förtränade modeller, verktyg och datauppsättningar för uppgifter som textklassificering, översättning, sammanfattning och inbäddningar.

För att möjliggöra effektiv hämtning av information för ramverket Retrieval-Augmented Generation (RAG) är det viktigt att konvertera textdata till ett högdimensionellt vektorformat som fångar dess semantiska betydelse. För detta ändamål använder vi den förutbildade inbäddningsmodellen "BAAI/bge-small-en-v1.5" från Hugging Face. Denna inbäddningsmodell är en lätt, förutbildad inbäddningsmodell utformad för att generera täta vektorrepresentationer av engelsk text. Denna modell är en del av Beijing Academy of Artificial Intelligence (BAAI) svit med språkmodeller, optimerade för semantisk sökning, textlikhet och hämtningsuppgifter. Den är baserad på transformatorarkitektur, som utmärker sig på att fånga kontextuella relationer mellan ord och fraser. Denna inbäddningsmodell spelar en avgörande roll för att möjliggöra effektiv hämtning för chatboten.

### 3.2.2 LlamaIndex

LlamaIndex är ett robust ramverk designat för att överbrygga gapet mellan externa datakällor och stora språkmodeller (LLM) som OpenAI:s GPT, Meta's Llama eller andra liknande modeller. Det ger applikationer som chatbots och Q&A-system möjlighet att förbättra LLM:er med extern kunskap i realtid. Det underlättar skapandet av intelligenta system som integrerar stora mängder data med de generativa kapaciteterna hos LLM, vilket möjliggör hämtningsförstärkta svar.

LlamaIndex-biblioteken adresserar begränsningen av LLM genom att aktivera:

- Intag av extern data: Strukturerad eller ostrukturerad data från databaser, dokument, API:er m.m.
- Indexering av data: Effektiv lagring och sökning av data i ett format som är tillgängligt för LLM.
- Sömlös integration: Koppla samman LLM med indexerad data för att ge sammanhangsmedvetna svar.

I vårt projekt förlitar vi oss på Simple Directory Reader för dataintag och Vector Store Index för effektiv indexering och hämtning av textdata.

**Simple Directory Reader:** Det är ett verktyg som tillhandahålls av LlamaIndex-biblioteket, som används för att mata in data från lokala filkataloger. Det här verktyget förenklar processen att ladda och förbereda textdata för indexering.

**Vector Store Index:** Det är indexeringsmekanismen vi använder för att lagra och hämta data baserat på semantisk likhet.

Genom att kombinera dessa två bearbetar, lagrar och hämtar vi data effektivt för att stödja vår chatbots RAG-funktioner. Denna pipeline säkerställer att extern kunskap är lättillgänglig och används effektivt för att generera kontextuellt korrekta svar.

### 3.2.3 Streamlit

Streamlit är ett Python-baserat bibliotek med öppen källkod designat för att skapa interaktiva webbapplikationer. Streamlit förenklar processen att distribuera maskininlärningsmodeller och applikationer med minimal kodningsansträngning, vilket gör det till ett idealiskt val för vårt projekt.

Vi använder Streamlit för att vara värd för chatboten som ger en tillgänglig plattform där användare kan interagera med chatboten. Den ansluter sömlöst med back-end RAG-systemet, inklusive LlamaIndex, Hugging Face-inbäddningar och den distribuerade Llama 3-modellen på Azure. En användare anger sina frågor via en textinmatningswidget och chatboten bearbetar inmatningen och visar svaren i ett konversationsformat.

### 3.2.4 Microsoft Azure

Azure är en molnplattform som erbjuds av Microsoft och tillhandahåller ett brett utbud av tjänster, inklusive virtuella maskiner, databaser, maskininlärning, AI-verktyg och mer. Det gör det möjligt för utvecklare att bygga, distribuera och hantera applikationer genom ett globalt nätverk av datacenter.

För att säkerställa en tillförlitlig och skalbar distribution av vår chatbots Large Language Model (LLM) använder vi Microsoft Azure som tillhandahåller robust infrastruktur och verktyg för värd för AI-modeller och applikationer. Specifikt har vi valt Llama 3 8B Instruct, en toppmodern språkmodell, för att driva chatbotens svar.

Azure säkerställer att modellen interagerar smidigt med Streamlit-gränssnittet och LlamaIndex-hämtningssystemet, vilket skapar en sömlös end-to-end pipeline för chatboten.

# 4 Slutsatser

Projektet har framgångsrikt utvecklat en AI-driven chatbot-applikation som använder Llama 3 8B Instruct, en banbrytande storspråksmodell av Meta. Genom att implementera ett ramverk för Retrieval-Augmented Generation (RAG) förbättrade vi chatbotens förmåga att generera kontextuellt korrekta svar genom att integrera extern kunskap i konversationsflödet.

Även om den här applikationen fungerar som en grundläggande exempelimplementering, har den betydande potential för skalning och förbättring. För närvarande lagras vektoriserad data i temporärt minne, som raderas vid omstart av kärnan. Att införliva en vektordatabas skulle möjliggöra beständig lagring och effektiv hämtning av vektoriserade data, vilket förbättrar applikationens skalbarhet och tillförlitlighet. Dessutom skulle en mer kraftfull LLM-modell kunna förbättra noggrannheten och robustheten hos chatboten ytterligare.

Projektet understryker den växande rollen för AI-tekniker som LLM:er för att automatisera dokumentanalys, förbättra produktiviteten och leverera överlägsna kundupplevelser. Det ligger i linje med den bredare trenden av företag som använder AI för att effektivisera verksamheten och extrahera handlingskraftiga insikter från stora volymer innehåll, vilket i slutändan visar den enorma potentialen hos LLM:er för att driva innovation och effektivitet i dagens datadrivna värld.

# Källförteckning

Ashish Vaswani, N. S. (2017). Attention Is All You Need.

*AWS Amazon*. (n.d.). Retrieved from https://aws.amazon.com/what-is/retrieval-augmented-generation/

*cloud flare*. (n.d.). Retrieved from https://www.cloudflare.com/en-gb/learning/ai/what-is-large-language-model/

Cole. (n.d.). *Codingscape*. Retrieved from https://codingscape.com/blog/rag-101-what-is-rag-and-why-does-it-matter

*Elastic*. (n.d.). Retrieved from https://www.elastic.co/what-is/large-language-models