INTELLIGENT CHAT-BASED SYSTEM FOR DIGITAL DOCUMENT PROCESSING WITH LARGE LANGUAGE MODEL INTEGRATION

STANIMIRA R. MLADENOVA, HRISTO PARASKEVOV

ABSTRACT: *This paper presents the design and implementation of an intelligent chat-based system for automated processing of digital documents. The system combines Discord as a conversational interface, an asynchronous Python bot, format-specific extraction modules for PDF, DOCX and PPTX, and an external large language model accessed through GroqCloud. The processing pipeline preserves useful structural markers such as Word table rows and PowerPoint slide boundaries before forming document-grounded prompts. Functional tests confirm correct extraction and interactive processing of heterogeneous files, controlled handling of unsupported or damaged inputs, and response times suitable for conversational use in the tested environment. Current limitations include lack of OCR, partial treatment of visual content and bounded document context supplied to the language model.*

KEYWORDS: *document processing; chatbot; large language models; Discord; Python; digital documents.*

DOI: https://doi.org/10.46687/YOWB9149

ИНТЕЛИГЕНТНА ЧАТ-БАЗИРАНА СИСТЕМА ЗА ОБРАБОТКА НА ЦИФРОВИ ДОКУМЕНТИ С ИНТЕГРАЦИЯ НА ГОЛЯМ ЕЗИКОВ МОДЕЛ*

СТАНИМИРА Р. МЛАДЕНОВА, ХРИСТО ПАРАСКЕВОВ

АБСТРАКТ: *В статията е представено проектирането и реализирането на интелигентна чат-базирана система за автоматизирана обработка на цифрови документи. Системата обединява Discord като разговорен интерфейс, асинхронен бот на Python, специализирани модули за извличане на съдържание от PDF, DOCX и PPTX и външен голям езиков модел чрез GroqCloud. При обработката се съхраняват полезни структурни маркери, включително редовете на таблици в Word и границите между слайдовете в PowerPoint. Проведените функционални тестове потвърждават коректното извличане и интерактивната обработка на разнородни файлове, както и контролираното поведение при проблемни входни данни. Ограниченията на текущата версия са свързани с липсата на OCR, непълната обработка на визуално съдържание и ограничения контекст, подаван към езиковия модел.*

1 Въведение

Бързото развитие на генеративния изкуствен интелект променя начина, по който потребителите търсят и интерпретират информация в цифрови документи. Големите езикови модели, изградени върху трансформаторната архитектура [1], позволяват задаване на въпроси на естествен език, резюмиране и извличане на факти от предоставен контекст. При практическото им приложение обаче остава необходимостта от надежден механизъм за приемане на документ, извличане на съдържанието му, запазване на полезната структура и ограничаване на отговора до наличната информация.

Проблемът е особено видим при хетерогенни файлови формати. PDF документите могат да съдържат сложен ред на четене, DOCX файловете комбинират параграфи и таблици, а презентациите разпределят информацията между отделни слайдове, текстови

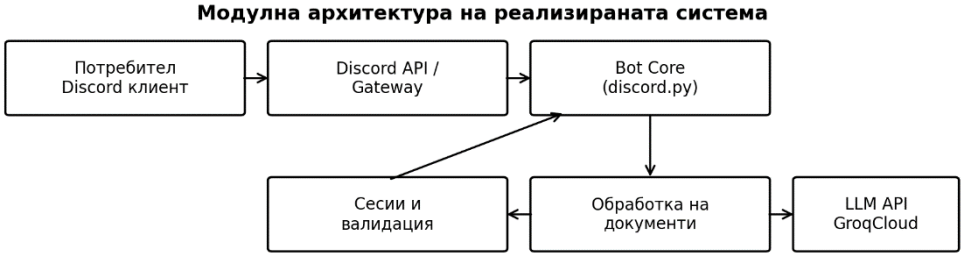
* Статията е частично финансирана по проект № РД-08-88/02.02.2026

полета и таблични обекти. Изследванията върху отговаряне на въпроси по документи показват, че структурата на документа е важна за коректното намиране и използване на контекста [2]. От друга страна, подаването на много голям текстов контекст към езиков модел не гарантира еднакво надеждно използване на информацията във всички негови части [3]. Поради това в по-мащабни решения се използват подходи за извличане на релевантни фрагменти, включително Retrieval-Augmented Generation (RAG) [4].

Целта на настоящата разработка е да се реализира модулна чат-базирана система, която приема PDF, DOCX и PPTX файлове, извлича текстовото им съдържание, формира контекст към голям езиков модел и връща резултата в разговорна среда. За интерфейс е избрана платформата Discord, чиито Gateway механизми поддържат двупосочна комуникация в реално време чрез WebSocket, а Application Commands предоставят нативни команди за взаимодействие с приложения [5]. Основният акцент е върху единния работен процес за различни формати, асинхронната обработка и контролираното поведение при грешки, без да се претендира за нов метод за обучение на езиков модел.

2 Архитектура на системата

Реализираната система е организирана модулно, така че комуникацията с платформата, обработката на файловете и взаимодействието с езиковия модел да бъдат логически отделени. Това намалява зависимостта между компонентите и позволява при необходимост да бъде заменен конкретен доставчик на AI услуга или потребителски интерфейс, без да се променя цялото приложение. Основните компоненти са показани на Фиг. 1.



Фиг. 1: Архитектура на реализираната чат-базирана система

Потребителят подава файл и команда през Discord. Събитието се приема от Discord Gateway и се обработва от бот, реализиран на Python с библиотеката discord.py. За операции, които могат да надхвърлят времето за непосредствен отговор на интеракцията, се използва отложен асинхронен отговор, така че клиентът да получи потвърждение, а обработката да продължи без блокиране. Входните данни преминават през валидиране на файловия формат и обработка на изключения. След извличане на текста се формира заявка към външния езиков модел и полученият резултат се адаптира към ограниченията на Discord.

В тестваната версия за AI обработка е използван моделът llama-3.3-70b-versatile чрез API на GroqCloud [6]. Моделът е част от семейството Llama 3, предназначено за многоезични и други задачи с естествен език [7]. Връзката с конкретния модел е изолирана в отделен модул, което позволява последваща подмяна без промяна на модулите за работа с документи.

3 Обработка на цифрови документи

Съществена част от системата е преобразуването на различни файлови формати в унифициран текстов контекст. При това не се цели премахване на цялата структура на

документа. Напротив, когато форматът го позволява, се запазват маркери, които подпомагат проследяването на произхода и логическите връзки между елементите.

Таблица 1

Модули за извличане и съхранявана структура

Формат	Библиотека	Основна обработка	Запазвана структура	Основно ограничение
PDF	PyMuPDF	Извличане на текст по страници; визуализация на страница	Граници на страници	Липса на OCR; зависимост от текстовия слой
DOCX	python-docx	Последователно обхождане на параграфи и таблици	Ред на параграфи; редове на таблици	Сложни вградени обекти не се интерпретират
PPTX	python-pptx	Обхождане на слайдове, текстови полета и таблици	Маркер за номер на слайд; таблични редове	Графики и изображения не се анализират

При PDF файловете се използва PyMuPDF, който позволява извличане на текста от всяка страница и при необходимост нейното графично рендиране. Документацията на библиотеката също отбелязва, че при PDF редът на извлечения текст може да се различава от визуалния ред, което е присъщо ограничение при някои документи [8]. В текущата реализация PDF без четим текстов слой се разпознава като неподходящ за директен анализ и потребителят получава контролирано съобщение за необходимост от OCR.

При DOCX съдържанието се обхожда на ниво XML елементи, като параграфите и таблиците се добавят към общия текст в реда на появата им. Клетките от един табличен ред се свързват със символен разделител, така че връзката между стойностите да не се губи при подаване към езиковия модел. При PPTX се обхождат последователно слайдовете и техните shapes. Преди съдържанието на всеки слайд се добавя маркер с неговия номер, а таблиците се извличат ред по ред. Така резултатът остава текстов, но запазва част от логическата организация на оригиналния документ.

4 Интеграция с езиков модел и организация на взаимодействието

След извличането на съдържанието системата съставя заявка, която съдържа системна инструкция, документен контекст и потребителска задача. Системната инструкция задава на модела да се придържа към предоставения документ и при липса на необходимата информация да не допълва отговора с непотвърдени твърдения. Този подход представлява контекстно ограничаване на генерацията, но в текущата версия не е реализирана пълна RAG архитектура с векторен индекс. Вместо това към модела се подава ограничен фрагмент от извлечения текст, което е важно ограничение при големи документи и е една от основните насоки за бъдещо развитие.



Фиг. 2: Обобщен поток за обработка на документ и генериране на отговор

Потребителският интерфейс предоставя операции за задаване на въпрос, резюмиране, превод, извличане на текст и генериране на аудио. Полученият отговор се връща чрез Discord като ефемерно съобщение, когато е необходимо резултатът да остане видим само

за инициатора. Тъй като платформата ограничава дължината на едно съобщение, дългите резултати се разделят на части. Сесийният механизъм позволява последващи действия върху вече качения документ, без повторно изпращане на файла при всяка команда.

От гледна точка на надеждността се проверяват разширението и типът на файла, а изключенията при четене или мрежова комуникация се прихващат и преобразуват в разбираемо съобщение към потребителя. API ключовете се зареждат от променливи на средата, а не се съхраняват директно в програмния код. Разполагането на приложението в облачна среда позволява ботът да работи като постоянен фонов процес и да поддържа WebSocket връзката с Discord.

5 Тестване и анализ на резултатите

Функционалното тестване е проведено с контролирани документи в трите поддържани формата и с негативни сценарии. Целта е да се провери целият работен поток - от приемането на файла до връщането на отговор - а не да се изгражда самостоятелен benchmark за качеството на езиковия модел. Поради това резултатите следва да се интерпретират като инженерна верификация на системата, а не като статистическо сравнение на различни LLM модели.

Таблица 2
Обобщение на проведените функционални тестове

Сценарий	Тестов вход	Наблюдаван резултат	Оценка
PDF	7-страничен текстов документ	Извлечен е текстът от всички страници; генерирано е резюме за около 1,8 s	Успешен
DOCX	10 параграфа и таблица	Извлечени са параграфите и табличните редове; отговорено е на въпрос по таблицата	Успешен
PPTX	8 слайда	Текстът е извлечен по слайдове; маркерите за номерата се запазват при обработката	Успешен
Проблемни файлове	JPG, празен DOCX, прекъсната AI връзка	Файловете/грешките се обработват контролирано без срив на бота	Успешен
Време за реакция	Стандартни файлове до 5 MB	Извличане под 1 s; AI генерация в наблюдавания тестов диапазон 1,2-2,5 s	Приемливо за чат

Тестовете показват, че единният интерфейс може да обслужва различни документни формати, без потребителят да използва отделни инструменти за тяхното предварително преобразуване. Особено полезно е запазването на табличните редове при DOCX и маркерите за слайдове при PPTX, защото така генерираният отговор може по-лесно да бъде съпоставен с оригиналния документ. Негативните тестове потвърждават, че неподдържан формат, празен документ или прекъсване на външната услуга не водят до неконтролирано прекратяване на бота.

Резултатите имат и ясни ограничения. В текущата версия липсва OCR, поради което сканирани PDF файлове и изображения без текстов слой не могат да бъдат анализирани. Вградените графики и изображения в DOCX и PPTX не се интерпретират семантично. Освен това AI функциите работят с ограничена част от извлечения контекст, което може да доведе до пропускане на информация при много големи документи. Изследванията за работа с дълъг контекст показват, че простото увеличаване на подадения текст не решава напълно този проблем [3]. Подходящо продължение е въвеждането на retrieval модул, който избира релевантни фрагменти [4], като при това запазва структурните маркери на документа, в съответствие с изводите за structure-aware document QA [2].

6 Заключение

Представена е работеща чат-базирана система за обработка на PDF, DOCX и PPTX документи с интеграция на голям езиков модел. Основният практически резултат е единен

асинхронен работен процес, който съчетава приемане и валидиране на файлове, форматно-специфично извличане на съдържание, частично запазване на документната структура, формиране на контекст към LLM и връщане на резултата в разговорен интерфейс. Проведените тестове потвърждават функционалната приложимост на архитектурата и устойчивото поведение при типични грешки.

Разработката не решава напълно задачата за анализ на големи и мултимодални документи. Следващите стъпки са интегриране на OCR, семантично търсене и RAG за големи колекции, количествена оценка на точността върху набор от контролни въпроси и сравнение на различни стратегии за представяне на структурирано документно съдържание. Това би позволило инженерната платформа да се използва и като експериментална среда за по-задълбочени изследвания върху надеждното отговаряне на въпроси по цифрови документи.

ЛИТЕРАТУРА:

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I. Attention Is All You Need. Advances in Neural Information Processing Systems 30 (2017), 5998-6008.

[2] Saad-Falcon, J., Barrow, J., Siu, A., Nenkova, A., Yoon, S., Rossi, R.A., Dernoncourt, F. PDFTriage: Question Answering over Long, Structured Documents. Proceedings of EMNLP 2024: Industry Track (2024), 153-169.

[3] Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P. Lost in the Middle: How Language Models Use Long Contexts. Transactions of the Association for Computational Linguistics 12 (2024), 157-173.

[4] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., Kiela, D. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems 33 (2020).

[5] Discord. Gateway; Interactions & Application Commands. Discord Developer Documentation. Available: https://docs.discord.com/developers/ (accessed 10.09.2026).

[6] Groq, Inc. Groq API Documentation. Available: https://console.groq.com/docs/ (accessed 10.09.2026).

[7] Grattafiori, A. et al. The Llama 3 Herd of Models. arXiv:2407.21783 (2024).

[8] Artifex Software, Inc. PyMuPDF Documentation: Text Extraction. Available: https://pymupdf.readthedocs.io/ (accessed 10.09.2026).

Станимира Р. Младенова
студентка IV курс, специалност „Софтуерно инженерство“
Шуменски университет „Епископ Константин Преславски“, Факултет по математика и информатика
E-mail: 2230150002@shu.bg

Христо Параскевов
ORCID ID: 0000-0001-8097-6131
ШУ „Епископ Константин Преславски“
e-mail: h.paraskevov@shu.bg