ВАРНЕНСКИ СВОБОДЕН УНИВЕРСИТЕТ „ЧЕРНОРИЗЕЦ ХРАБЪР”
ФАКУЛТЕТ „МЕЖДУНАРОДНА ИКОНОМИКА И АДМИНИСТРАЦИЯ”
КАТЕДРА „КОМПЮТЪРНИ НАУКИ”

ДИПЛОМНА РАБОТА

"Разработване на виртуален асистент с изкуствен интелект з а подпомагане на психичното здраве "

"Development of an AI-powered Virtual Assistant for Mental Health Support"

Дипломант: Мирела Свиленова Йосифова
Специалност: Информатика и компютърни науки
Фак.№: 193010038

Ръководител: Доц. д-р Веселина Спасова

Варна, април 2024

Анотация

Проектът "Разработване на виртуален асистент с изкуствен интелект за подпомагане на психичното здраве" представлява иновативно начинание, целящо да отговори на нарастващата нужда от достъпни и ефективни услуги за психично здраве. В съвременния свят, където психичните разстройства продължават да засягат милиони хора глобално, този проект се стреми към създаването на технологично решение, което да улесни достъпа до психологическа помощ и подкрепа чрез използването на напредъка в областите на изкуствения интелект (ИИ) и обработката на естествен език (NLP).

Основавайки се на микросервисна архитектура и включващ използването на съвременни технологии като React.js за фронтенд разработка, Node.js за бекенд решения, MongoDB за управление на бази данни и модели на OpenAI GPT за генериране на отговори от изкуствения интелект, проектът предлага виртуален асистент, способен на емпатично и персонализирано взаимодействие с потребителите. Тази платформа е насочена към предоставянето на незабавна подкрепа за лица, изпитващи стрес, тревожност, депресия и други психични трудности, като по този начин допринася за намаляването на чувството за изолация и подобряване на общото психично благосъстояние.

Един от ключовите аспекти на дипломната работа е акцентът върху конфиденциалността и безопасността на потребителските данни, което е от съществено значение при работата с чувствителна информация, свързана с психичното здраве. Платформата е проектирана така, че да гарантира анонимност и

сигурност на всяко взаимодействие, като по този начин се стреми към изграждането на доверие сред потребителите.

В заключение, проектът за разработване на виртуален асистент с ИИ за подпомагане на психичното здраве не само демонстрира иновативно използване на съвременни технологии за адресиране на критични обществени предизвикателства, но също така поставя основите за бъдещи изследвания и разработки в областта. Призивът за продължаващи иновации и изследвания подчертава важността на технологичния прогрес в стремежа към подобряване на психичното благосъстояние и достъпа до психологически услуги на глобално ниво.

Увод

В днешната динамична и все по-напрегната среда, проблемите свързани с психичното здраве се превръщат във все по-значимо предизвикателство за обществото. Според Световната здравна организация (СЗО), проблемите с психичното здраве са сред водещите причини за общото заболяваемост и увреждане по света. Въпреки това, много хора все още намират трудности в търсенето на подкрепа поради стигма, разходи и логистични предизвикателства. В този контекст, разработването на иновативни и достъпни методи за предоставяне на психична подкрепа е от критично значение за подобряването на общественото здраве и благополучие.

Виртуалните асистенти с изкуствен интелект (ИИ) предлагат вълнуваща възможност за трансформиране на начина, по който хората получават подкрепа за психичното здраве. Тези системи могат да предложат незабавна, анонимна и персонализирана помощ, достъпна 24 часа в денонощието, което е особено важно за лица, изпитващи стрес, тревожност, депресия и други психични проблеми. Възможността за текстова комуникация с ИИ, който използва напреднали техники за обработка на естествен език (NLP) и машинно обучение (ML), предоставя на потребителите възможност за взаимодействие в удобна и непринудена обстановка. Това не само допринася за намаляването на стигмата, свързана с търсенето на помощ, но и улеснява достъпа до подкрепа за тези, които могат да се колебаят да потърсят лице в лице консултации.

Целта е разработването на платформа за психично здраве, базирана на изкуствен интелект, която предлага персонализирана подкрепа за лица с психични заболявания или изпитващи временни психологически трудности. Основната мисия на

платформата е да демократизира достъпа до психологическа подкрепа, като същевременно гарантира поверителността и анонимността на потребителите. Тя има потенциала да служи като допълнителен ресурс към традиционните методи за лечение на психични заболявания, като предоставя на потребителите възможност да изследват и управляват своето психично състояние в безопасна и подкрепяща среда.

Дипломната работа демонстрира, че "AI Mental Assistant App" успешно интегрира различни технологични решения за създаване на ефективна платформа за психологическа подкрепа. Откроени са следните основни резултати:

- Приложението предлага високо ниво на персонализация и точност при отговорите благодарение на интегрираните AI и NLP технологии.

- Системата поддържа висока степен на сигурност и поверителност на потребителските данни, съответствайки на съвременните изисквания за защита на личната информация.

- Потребителският интерфейс е оценен високо от крайните потребители за удобство и лесна навигация.

Проектът ще се фокусира върху разработването на интуитивен потребителски интерфейс, който да е лесен за използване от лица от всички възрастови групи. Също така ще се акцентира върху сигурността и защитата на личните данни, като се прилагат най-съвременни технологии и протоколи за криптиране. Включването на модули за обратна връзка и персонализация е от съществено

значение, тъй като те ще позволят на платформата постоянно да се адаптира и усъвършенства в отговор на нуждите на своите потребители.

Този проект представлява важна стъпка към преодоляването на бариерите за достъп до качествена психологическа подкрепа. Чрез интеграцията на напреднали технологии и изкуствен интелект, платформата има потенциала не само да подобри живота на индивидите, изпитващи психични трудности, но и да допринесе за обществото като цяло, като насърчава по-отворен и подкрепящ диалог за психичното здраве.

Глава 1: Общ преглед

1.1 Анализ на Съществуващи Решения

С развитието на технологиите, особено в сферата на информационните технологии и изкуствения интелект (ИИ), нараства интересът към разработването на иновативни решения за подкрепа на психичното здраве. Този раздел ще предостави анализ на съществуващите технологични решения, като се фокусира върху приложения, платформи и системи, които използват ИИ за диагностика, терапия и подкрепа в областта на психичното здраве. Ще бъдат разгледани техните основни характеристики, предимства, ограничения и въздействие върху потребителите.

1.1.1 Обзор на Съществуващи Технологични Решения

Съществуващите решения могат да бъдат категоризирани в няколко основни групи:

- **Мобилни Приложения за самопомощ и управление на стреса:** Тези приложения предлагат различни методи за релаксация, медитация, дневни упражнения за управление на стреса и анксиозността. Примери включват "Headspace", "Calm" и "Moodfit", които предоставят персонализирани съвети и упражнения, базирани на потребителските предпочитания и поведение.

- **Платформи за онлайн терапия:** Тези платформи свързват потребителите с лицензирани терапевти чрез видео, гласови обаждания или текстови съобщения. "Talkspace" и "BetterHelp" са сред водещите представители в тази категория, предлагайки достъп до професионална психологическа помощ от всяко място и по всяко време.

- **Системи за ранно откриване и мониторинг:** Разработени са интелигентни системи, които използват машинно обучение и анализ на данни за ранно откриване на признаци на психични заболявания. Те анализират поведенчески модели, език на тялото и глас, за да идентифицират потенциални рискове за психично здраве.

- **Интерактивни асистенти и чатботове:** Използването на ИИ за създаване на виртуални асистенти, които предоставят емпатични и персонализирани отговори на потребителите. Примери като "Woebot" и "Wysa" използват NLP за разбиране и реагиране на емоционалните състояния на потребителите, предлагайки подкрепа и упражнения за справяне с анксиозност и депресия.

1.1.2 Анализ на предимства и ограничения

Предимствата на използването на технологии за подкрепа на психичното здраве включват:

- Достъпност и удобство: Технологичните решения предлагат лесен и бърз достъп до подкрепа, което е особено полезно за хора в отдалечени или недостатъчно обслужвани райони.
- Анонимност и ненатрапчивост: Много потребители предпочитат анонимността, която предлагат онлайн платформите, тъй като това намалява стигмата свързана с търсенето на психологическа помощ.
- Персонализация: ИИ и анализът на данни позволяват на решенията да бъдат персонализирани според нуждите и предпочитанията на отделния потребител.

Ограниченята включват:

- Проблеми със сигурността на данните при ъхранението и обработката на лични и чувствителни данни изисква строги мерки за сигурност, за да се предотврати злоупотреба или изтичане на информация.
- Автоматизираните системи все още се сблъскват с предизвикателства при правилната интерпретация на сложните човешки емоции и състояния.
- Не може напълно да замени човешката емпатия и разбиране, които идват от личния контакт с терапевт.

Технологичните решения в областта на психичното здраве представляват значителен напредък към предоставянето на достъпна, ефективна и иновативна подкрепа. Въпреки наличните предизвикателства и ограничения, потенциалът за подобряване на живота на милиони хора по света, страдащи от психични заболявания, е огромен. Бъдещите усилия трябва да се съсредоточат върху подобряването на точността, надеждността и сигурността на тези решения, както и на разширяването на тяхната достъпност и приемане от широката общност.

1.2 Ролята на ИИ и NLP в психологията и психиатрията

Изкуственият интелект (ИИ) и обработката на естествен език (NLP) играят все по-значима роля в областта на психологията и психиатрията, предлагайки нови възможности за диагностика, мониторинг и терапия на психични заболявания.

ИИ системите могат да анализират големи обеми от данни – като текст, реч и физиологични показатели – за да идентифицират модели, които могат да бъдат свързани с определени психични състояния. Например, алгоритмите на ИИ могат да анализират речта и писането на индивида за признаци на депресия или тревожност, позволявайки ранно откриване и интервенция.

Чатботовете и виртуалните асистенти, задвижвани от ИИ и NLP, предлагат непрекъснато достъпна подкрепа и могат да провеждат базирани на доказателства терапевтични сесии, като когнитивно-поведенческа терапия (КПТ). Тези технологии позволяват на

потребителите да разработват и практикуват стратегии за справяне с психологически трудности в комфорта на собствения им дом.

ИИ може да анализира отговорите на лечение на отделни пациенти и да предложи персонализирани препоръки за подобрение на терапевтичния процес. Това включва адаптиране на лечебни планове според специфичните нужди и предпочитания на пациента, което може значително да повиши ефикасността на лечението.

В заключение, технологиите в областта на ИИ и NLP предлагат значителни възможности за подобряване на психологическата подкрепа и терапия. Те не само осигуряват по-широк достъп до помощ и ресурси за хората с психични проблеми, но също така предлагат нови методи за ранно откриване, мониторинг и персонализирано лечение на психични заболявания. Въпреки това, е важно да се отбележат и предизвикателствата, свързани със защитата на личните данни, етиката на използването на ИИ в психологията и необходимостта от по-нататъшни изследвания за оценка на дългосрочната ефикасност и безопасност на тези подходи.

Глава 2: Проектиране

2.1 Основни функции

В основата на разработването на платформата за подкрепа на психичното здраве стоят ключови функции, които определят ефективността и полезността ѝ за крайните потребители. Тези функции са разработени с цел да адресират специфичните нужди на потребителите и да осигурят безопасна, интуитивна и подкрепяща среда за тях. В следващите подраздели ще разгледаме детайлно всяка от тези функции.

2.1.1 Взаимодействие с Изкуствен Интелект

Основната цел на взаимодействието с ИИ е да се създаде интуитивен, емпатичен и персонализиран диалог между виртуалния асистент и потребителя. ИИ използва обработка на естествен език (NLP) и машинно обучение (ML), за да анализира въпросите и отговорите на потребителите, да идентифицира емоционални тенденции и да предоставя подходящи отговори.

Използването на NLP за анализ и разбиране на естествения език позволява на системата да обработва човешката реч или текст. ML алгоритмите се обучават върху големи обеми от данни, за да подобряват своята способност за отговор в реално време.

- **Персонализация**: Системата адаптира своите отговори според предпочитанията и историята на поведението на потребителя, осигурявайки персонализиран подход към всеки индивид.

- **Обратна Връзка и Обучение:** Включването на механизъм за обратна връзка позволява на системата да се самообучава и оптимизира въз основа на интеракциите с потребителите.

2.1.2 Управление на потребителите

Управлението на потребителите е критична функция, която позволява на системата да идентифицира, регистрира и следи активността на всеки потребител. То включва автентикация, съхранение на профили, следене на предпочитания и история на диалози. За управление на потребителите могат да бъдат използвани следните технологии и стратегии:

- Автентикация и регистрация: Използването на сигурни методи за регистрация и вход, като например OAuth или двуфакторна автентикация, осигурява защита на потребителските акаунти.

- Управление на профили: Системата позволява на потребителите да персонализират своите профили, включително настройки за поверителност и предпочитания за взаимодействие.

- Анализ на потребителското поведение: Събирането и анализирането на данни относно поведението и взаимодействията на потребителите допринася за подобряването на услугата.

2.1.3 Сигурност и Поверителност

Сигурността и поверителността са от особена важност при съхранението и обработката на лични и чувствителни данни. Разработените мерки за сигурност включват използването на

криптография за шифроване на данни, защита на акаунтите чрез силни аутентификационни механизми и внедряването на протоколи за сигурна комуникация като HTTPS. Тези мерки са насочени към предотвратяване на неоторизиран достъп и изтичане на информация.

2.1.4 Обратна Връзка

Функцията за обратна връзка позволява на потребителите да докладват своите впечатления, проблеми или предложения за подобрение на платформата. Тази информация е критична за непрекъснатото подобрение на системата и адаптирането ѝ към нуждите на потребителите.

2.1.5 Спешна Подкрепа

Платформата предоставя механизъм за спешна подкрепа, който се активира при идентифициране на потребител в кризисно състояние. Този механизъм може да включва автоматично предоставяне на информация за спешни контакти или пренасочване към професионални служби за психично здраве.

2.1.6 Персонализирана Подкрепа

Изкуственият интелект и анализът на данни се използват за осигуряване на персонализирана подкрепа за потребителите. Това включва адаптиране на съдържанието, съветите и предложенията за упражнения в зависимост от индивидуалните нужди, поведение и предпочитания на потребителя.

2.2 База Данни

2.2.1 MongoDB

Дизайнът на базата данни за "AI Mental Assistant App" е съсредоточен върху поддържането на потребителски профили, чат сесии, съобщения и обратна връзка. Използвайки MongoDB, една документно-ориентирана NoSQL база данни, можем да създадем гъвкава схема, която да отговаря на динамичната природа на данните в приложение за чат. Ето предложение за схеми на колекциите:

Колекция: Users

Записва информация за потребителите на приложението.

- **id**: Уникален идентификатор за всеки потребител.
- **username**: Потребителско име.
- **email**: Имейл адрес на потребителя.
- **hashedPassword**: Хеширана парола за безопасно съхранение.
- **createdAt**: Дата на създаване на акаунта.
- **updatedAt**: Дата на последно обновление на акаунта.

Колекция: ChatSessions

Управлява чат сесиите между потребителите и AI.

- **id**: Уникален идентификатор за всяка чат сесия.
- **userId**: Идентификатор на потребителя, който започва сесията.

- **status**: Статус на сесията (например, активна, завършена).
- **createdAt**: Дата на създаване на сесията.
- **endedAt**: Дата на завършване на сесията.

Колекция: Messages

Съхранява съобщенията, изпратени по време на чат сесиите.

- **id**: Уникален идентификатор за всяко съобщение.
- **chatSessionId**: Идентификатор на чат сесията, към която принадлежи съобщението.
- **userId**: Идентификатор на потребителя, изпратил съобщението.
- **messageText**: Текст на съобщението.
- **sentAt**: Дата и час на изпращане на съобщението.
- **direction**: Посока на съобщението (от потребител към AI или от AI към потребител).

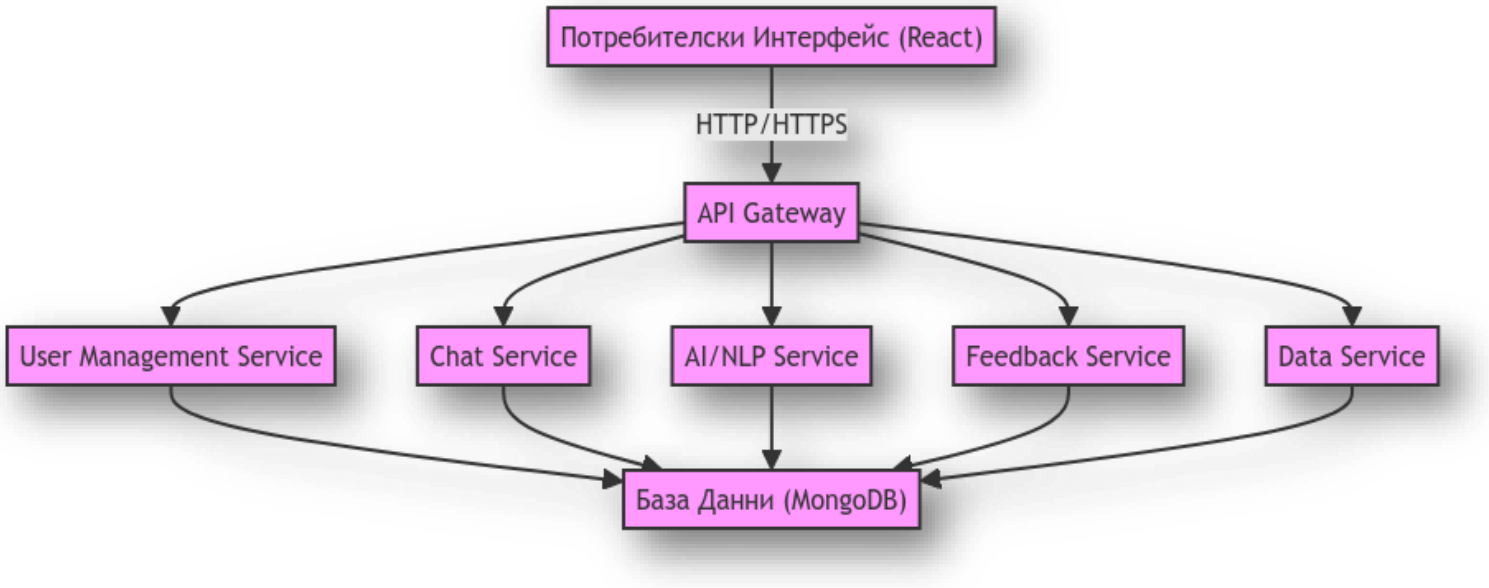
Колекция: Feedback

Позволява на потребителите да дават обратна връзка за приложението.

- **id**: Уникален идентификатор за всяка обратна връзка.
- **userId**: Идентификатор на потребителя, дал обратната връзка.
- **feedbackText**: Текст на обратната връзка.

- **rating**: Оценка, дадена от потребителя.
- **createdAt**: Дата на създаване на обратната връзка.

Този дизайн на базата данни осигурява необходимата гъвкавост и ефективност за управление на данните в "AI Mental Assistant App". Използването на MongoDB позволява лесно скалиране и адаптация на схемата с течение на времето, което е идеално за динамични приложения като това.



Фиг. 1 Архитектура на приложението

1. **Users и ChatSessions: Един-към-Много (1:N)**

Всяка инстанция от **Users** може да започне множество **ChatSessions**. Тази взаимовръзка е типичен пример за модела

"един към много", където един потребител инициира различни сесии на чат през времето.

Във всеки запис в **ChatSessions** се съхранява **userID** като външен ключ, който се отнася до уникалния идентификатор на потребителя в **Users**.

2. Users и Messages чрез ChatSessions

Въпреки че **Messages** са свързани директно с **ChatSessions**, те също имат непряка връзка с **Users** чрез **ChatSessions**. Това означава, че всеки **Message** може да бъде свързан с конкретен **User**, което позволява проследяване на изпращача на всяко съобщение.

Messages съдържат **userID**, което е външен ключ, свързващ съобщението с конкретен потребител. Това позволява лесно извличане на всички съобщения, изпратени от даден потребител, независимо от сесията на чата.

3. ChatSessions и Messages: Един-към-Много (1:N)

Всяка **ChatSession** може да включва множество **Messages**. Тази взаимовръзка е основна за структурата на чата, където сесиите обикновено се със тоят от поредица от съобщения.

В **Messages** се съхранява **sessionID** като външен ключ, който идентифицира чат сесията, към която принадлежи всяко съобщение.

4. Users и Feedback: Един-към-Много (1:N)

Един потребител може да предостави множество обратни връзки за приложението. Това позволява на потребителите да изразяват своите мнения и предложения по отношение на различни аспекти на системата.

Във всеки запис в Feedback се съхранява userID като външен ключ, което позволява всяка обратна връзка да бъде свързана с конкретен потребител.

2.3 CI/CD и Управление на версиите

В съвременната разработка на софтуер, практиките на непрекъсната интеграция (Continuous Integration, CI) и непрекъсната доставка (Continuous Delivery, CD) са критични за ускоряване на процесите на тестване и доставка на код. В допълнение, управление на версиите играе ключова роля за осигуряване на проследимост, възстановимост и колаборация по време на разработка. Инструменти като GitHub Actions и Git са от централно значение за имплементацията на CI/CD и управлението на версии на проекта "AI Mental Assistant App".

2.3.1 GitHub Actions

GitHub Actions е автоматизирана система за CI/CD, която позволява да се създават, тестват и разгръщат приложения директно от GitHub. Системата предлага гъвкавост чрез дефиниране на процеси (workflows) в YAML конфигурационни файлове, улеснявайки автоматизацията на тестове, сглобяване, и разгръщане на софтуерни проекти.

GitHub Actions е инструмент за автоматизация, който позволява на "AI Mental Assistant App" непрекъсната интеграция (CI) и непрекъсната доставка (CD). Този процес включва следните ключови дейности:

1. **Автоматично Тестване**

   GitHub Actions улеснява конфигурирането на работни потоци (workflows), които автоматично стартират тестове всеки път, когато се направи push или pull request в репозитория. Това гарантира, че всяко ново изменение е преминало необходимите тестове за стабилност и съвместимост преди интегрирането му в основния клон на кода.

2. **Автоматично Сглобяване и Разгръщане:**

   GitHub Actions може да се използва за автоматизиране на процеса на сглобяване и разгръщане на приложението в различни среди (тестова, предпродукционна, продукционна). Това осигурява бърз и последователен достъп до последните версии на софтуера и улеснява управлението на версиите и разгръщането в множество среди.

3. **Управление на Зависимости:**
   Автоматизирането на обновленията на зависимостите чрез GitHub Actions помага в поддържането и актуализирането на библиотеки и други зависимости, като същевременно се улеснява идентифицирането и коригирането на потенциални сигурностни проблеми.

2.3.2 Git

Git е децентрализирана система за управление на версии, поддържаща колаборативната разработка на софтуер. Системата предоставя инструменти за проследяване на промени в кода, възстановяване на версии и разрешаване на конфликти.

Интеграция с "AI Mental Assistant App" позволява:

- **Управление на версиите и възстановяване**: Съхраняване на история на промените, улеснявайки възстановяване на предишни версии при необходимост.

- **Разклонения и сливания:** При разработването могат да се създават отделни клонове за различни функционалности, които след тестове могат да бъдат сливани в основния код.

- **Решаване на Конфликти:** Git осигурява ефективни механизми за идентифициране и разрешаване на кодови конфликти, което е важно при колаборативна работа.

Интеграцията на GitHub Actions и Git осигуряват солидна основа за непрекъсната интеграция, доставка и управление на версии. Тези инструменти улесняват автоматизацията на различни процеси и гарантират високо качество и стабилност на кода. Така проектът може ефективно да се адаптира към промени, да управлява зависимости и да поддържа високо ниво на сигурност и надеждност на софтуера.

2.4 Сигурност

Сигурността е критичен аспект на всеки софтуерен продукт, особено когато става въпрос за приложения, които обработват лични данни и комуникации. В контекста на "AI Mental Assistant App", се използват няколко ключови технологии за гарантиране на сигурността на данните и взаимодействията на потребителите: JWT (JSON Web Tokens), HTTPS и bcrypt.

2.4.1 JWT (JSON Web Tokens)

JWT е компактен, URL-безопасен метод за представяне на твърдения между две страни. В контекста на уеб приложения,

токените JWT се използват за аутентикация на потребители и сесии и за пренос на информация между клиента и сървъра.

При успешно влизане в системата, сървърът генерира токен JWT, който се изпраща обратно към клиента. Токенът съдържа идентификатор на потребителя и/или друга информация, която потребителят може да използва за последващи заявки.

Клиентът изпраща този токен като част от HTTP header за всяка заявка, която изисква аутентикация. Сървърът проверява валидността на токена при всяка заявка.

2.4.2 HTTPS и bcrypt

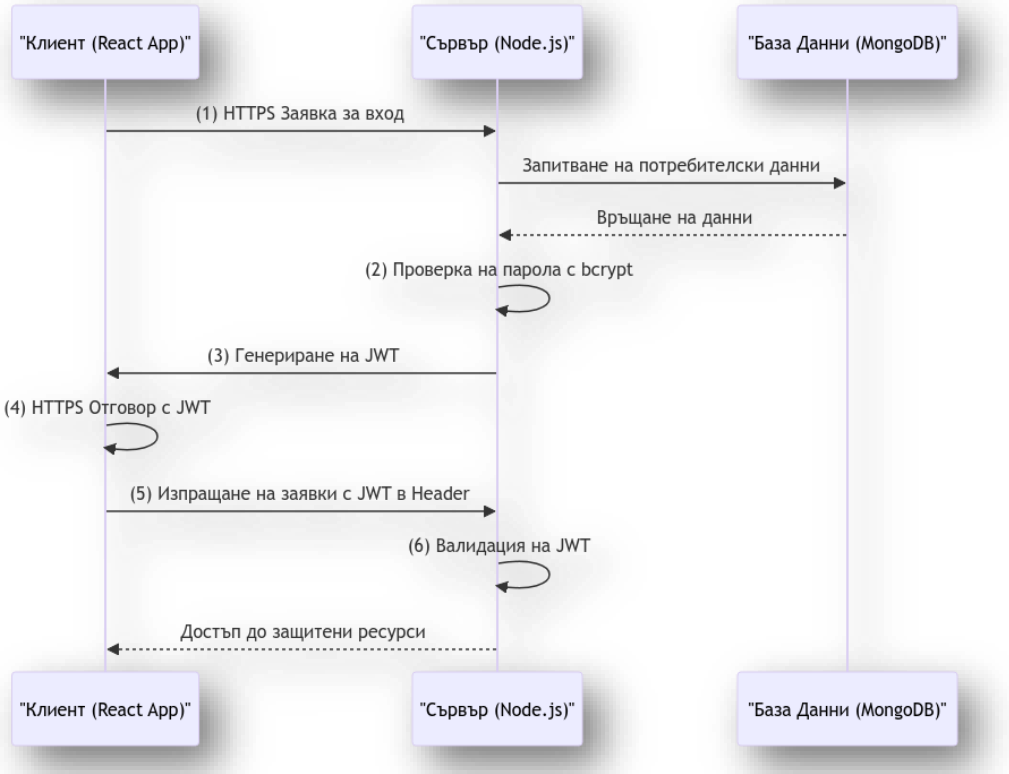
HTTPS (Hypertext Transfer Protocol Secure) е разширение на HTTP, което добавя слой на сигурност, като използва протоколи като SSL/TLS за криптиране на комуникациите между клиента и сървъра. Този протокол гарантира, че всички данни, изпратени по интернет, са криптирани, което предотвратява възможността за прехващане, прочитане или модифициране от трети лица.

Всички комуникации между потребителския интерфейс (React) и бекенд сървъра (Node.js) на "AI Mental Assistant App" се извършват посредством HTTPS, за да се гарантира защитата на потребителските данни, включително лична информация и данни за вход.

bcrypt е метод за криптографично хеширане на пароли, разработен да използва посоляване и да изпълнява многократно хеширане. Тази техника е проектирана да е устойчива на атаки чрез

брутфорс и използване на rainbow tables, като същевременно предлага механизъм за лесно настройване на трудността на хеширането в зависимост от производителността на хардуера.

При създаването на потребителски акаунт или промяна на парола в "AI Mental Assistant App", въведената от потребителя парола се хешира използвайки bcrypt, преди да бъде съхранена в базата данни. Това осигурява, че дори ако данните от базата бъдат компрометирани, паролите на потребителите остават защитени.



Фиг.2 Процес на управление на потребителската идентификация и сесии

Описание на всяка стъпка от фиг.2 :

1. HTTPS Заявка за Вход

Когато потребителят въведе своето потребителско име и парола в уеб формуляра на приложението, тези данни се изпращат към сървъра чрез защитена HTTPS връзка. Използването на HTTPS е критично за гарантиране на сигурността на данните в транзит, тъй като тя криптира информацията, предотвратявайки възможността

за прехващане и четене на чувствителните данни от неоторизирани лица.

2. Проверка на Парола с bcrypt

След получаване на данните за вход, сървърът извлича хешираната версия на паролата на потребителя от базата данни. За хеширане на пароли често се използва библиотеката bcrypt поради нейните силни криптографски качества, които включват добавяне на "сол" и възможността за настройка на трудността на хеширането.

Сървърът използва bcrypt за да сравни предоставената парола с хешираната версия в базата данни. Ако двата хеша съвпаднат, проверката е успешна.

3. Генериране на JWT

При успешна проверка на паролата, сървърът генерира JSON Web Token (JWT). Този токен се създава с използването на секретен ключ, наличен само на сървъра, и може да съдържа различни данни като идентификатор на потребителя, ролите на потребителя и валидността на токена.

4. HTTPS Отговор с JWT

Генерираният JWT се изпраща обратно на клиента чрез защитен HTTPS отговор. Този токен служи като удостоверение за

идентификация за бъдещи заявки от клиента към сървъра, позволявайки му да достъпва защитени ресурси без необходимостта от повторно предоставяне на потребителско име и парола.

5. Изпращане на Заявки с JWT в Header

За следващите заявки към защитени ресурси на сървъра, клиентът прикача JWT в HTTP header (обикновено като Authorization header). Това позволява на сървъра лесно да идентифицира и удостовери клиента посредством токена.

6. Валидация на JWT

При получаване на заявка, сървърът извлича и валидира JWT от HTTP header. Тази валидация включва проверка за целост и изтичане на срока на валидност на токена, както и потвърждение, че токенът е подписан със същия секретен ключ, използван при създаването му. Ако токенът е валиден, сървърът предоставя достъп до запитания ресурс; в противен случай заявката се отхвърля.

Този процес е от съществено значение за поддържане на сигурността и целостта на потребителските данни в среда на взаимодействие и е ключов за предотвратяване на неоторизиран достъп до чувствителни ресурси.

2.5 Мониторинг и Логване

Мониторингът и логването са критични за поддържането на здравето и производителността на всяка система, особено за сложни приложения като "AI Mental Assistant App". Те осигуряват

ценни данни за поведението на системата, помагат за идентифициране на потенциални проблеми и спомагат за оптимизацията на производителността. В тази връзка, инструменти като Grafana и Kibana играят ключова роля в процесите на мониторинг и логване, като предлагат мощни възможности за визуализация и анализ.

Започвайки със събирането на метрики, е необходимо да конфигурираме компонентите на "AI Mental Assistant App" да генерират и изпращат данни към сървър за мониторинг. Използването на клиентски библиотеки като prom-client за Node.js позволява лесното интегриране с Prometheus, което е популярен инструмент за събиране на метрики.

2.5.1 Настройка на Prometheus

След като метриките се генерират, Prometheus се конфигурира да ги събира чрез своите job-ове за изстъргване (scraping). Това включва добавяне на конфигурационни правила в prometheus.yml, които определят къде и как често Prometheus трябва да събира данни.

2.5.2 Конфигуриране на Grafana

С Prometheus като източник на данни, Grafana се конфигурира да извлича тези метрики за визуализация. Това включва създаването на дашборди с графики, които показват ключови показатели за производителността (KPIs), като забавяне, брой заявки, грешки и др.

Логването на "AI Mental Assistant App" изисква конфигуриране на приложението да изпраща логове до Elasticsearch. Използването на библиотеки като winston или bunyan с поддръжка на Elasticsearch позволява логовете да бъдат автоматично изпращани и индексирани.

2.5.3 Настройка на Logstash

Ако е необходима допълнителна обработка на логовете преди те да бъдат съхранени в Elasticsearch, Logstash може да се използва като посредник. Той може да филтрира, преобразува и обогатява логовете преди тяхното съхранение, улеснявайки последващия анализ.

2.5.4 Използване на Kibana

С логовете вече съхранени в Elasticsearch, Kibana се използва за създаване на визуализации и дашборди. Това включва конфигуриране на Kibana да извлича и визуализира данни от Elasticsearch, позволявайки анализ на логове чрез търсене, филтриране и агрегация.

Интеграцията на Grafana и Kibana предлага комплексно решение за мониторинг и логване на "AI Mental Assistant App". Използването на тези инструменти не само подобрява способността за наблюдение и анализ на системата в реално време, но също така осигурява ценни инсайти за оптимизация на производителността и подобряване на потребителското изживяване. Тяхната интеграция улеснява бързото идентифициране и разрешаване на проблеми, допринася за по-висока стабилност на системата и подпомага непрекъснатото усъвършенстване на приложението.

2.6 Разгръщане (Deployment)

Deployment (разгръщането) на приложения е фундаментален етап в жизнения цикъл на разработка на софтуер, който прави приложението достъпно за крайните потребители. В съвременната разработка на софтуер, платформите като услуга (PaaS) като Heroku предлагат значителни улеснения за разгръщане, управление и мащабиране на приложения. Тази секция разглежда как Heroku може да бъде използван за ефективно разгръщане на "AI Mental Assistant App".

2.6.1 Heroku

Heroku е облачна платформа, която позволява лесно да се публикуват, управляват и мащабират приложения в облака. Тя поддържа множество програмни езици, включително Node.js, Ruby, Python, Java, PHP, и Go, което я прави атрактивен избор за разработване с различен технологичен фокус.

Основните характеристики на Heroku са:

- Heroku предлага прост и интуитивен процес за пускане на приложения чрез Git. С помощта на Git push Heroku автоматично сглобява и пуска приложението
- Позволява лесно мащабиране на приложения с помощта на "dynos", които са леки контейнери, в които се изпълняват приложения.

- Предлага богат набор от добавки (add-ons) за бази данни, мониторинг, анализ на логове и други, които могат лесно да бъдат интегрирани в приложенията.

Процес на разгръщане в Heroku включва:

- **Подготовка на Приложението**

Първата стъпка за deployment на "AI Mental Assistant App" на Heroku включва подготовката на кода на приложението. Това включва конфигуриране на зависимостите и дефиниране на процесни типове в Procfile, който указва как Heroku трябва да стартира различни процеси на приложението.

- **Deployment Чрез Git**

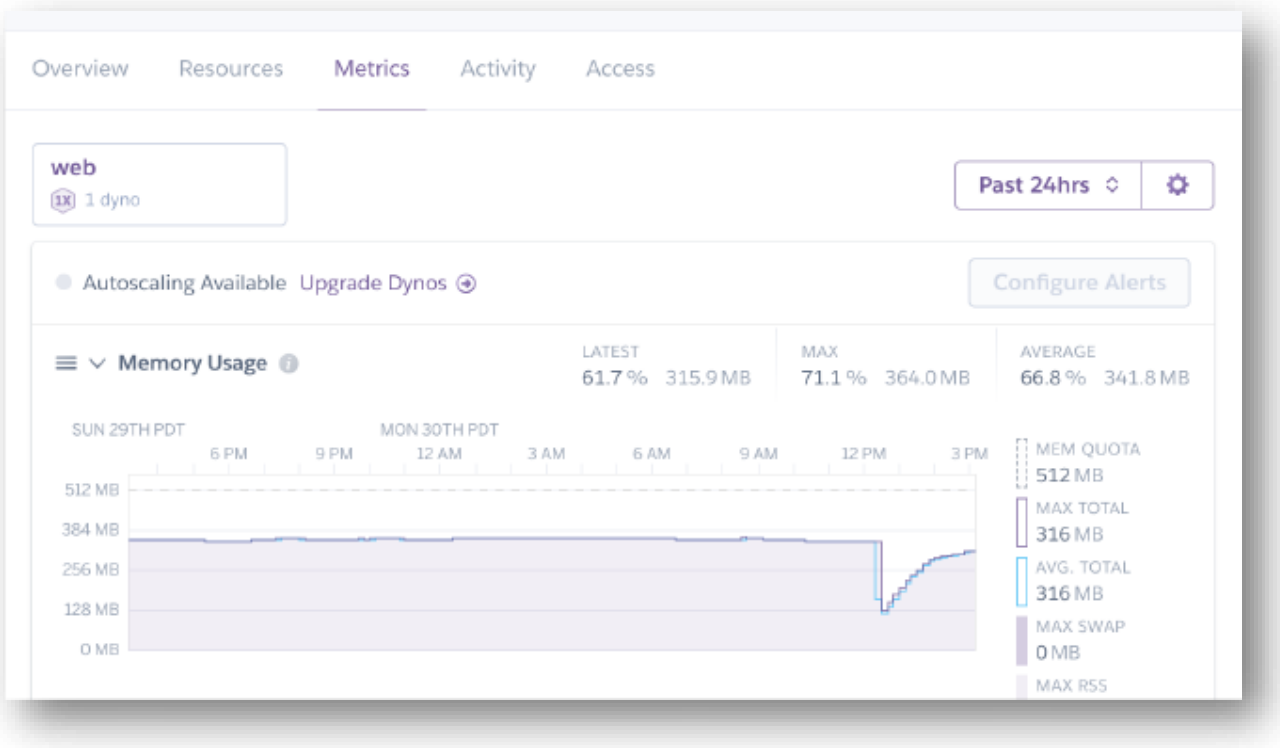
След като приложението е подготвено, кодът се изпраща в Heroku чрез Git. Heroku автоматично разпознава използваните технологии и сглобява приложението чрез съответните buildpacks. По време на този процес, Heroku също така инсталира всички необходими зависимости, както е указано във файловете за конфигурация като package.json за Node.js приложения.

- **Мониторинг и Управление**

След успешно разгръщане, Heroku предоставя и инструменти като Heroku Dashboard (фиг.3) или Heroku CLI (Command Line Interface) за мониторинг на състоянието на приложението, преглед на логовете, мащабиране на ресурсите и извършване на други управленски задачи.

- **Използване на Add-ons**

За да подобри функционалността и производителността на "AI Mental Assistant App", могат да бъдат интегрирани различни Heroku add-ons. Например, add-ons за бази данни като Heroku Postgres или Redis могат да бъдат използвани за управление на данни, докато add-ons за мониторинг като New Relic осигуряват детайлен анализ на производителността.

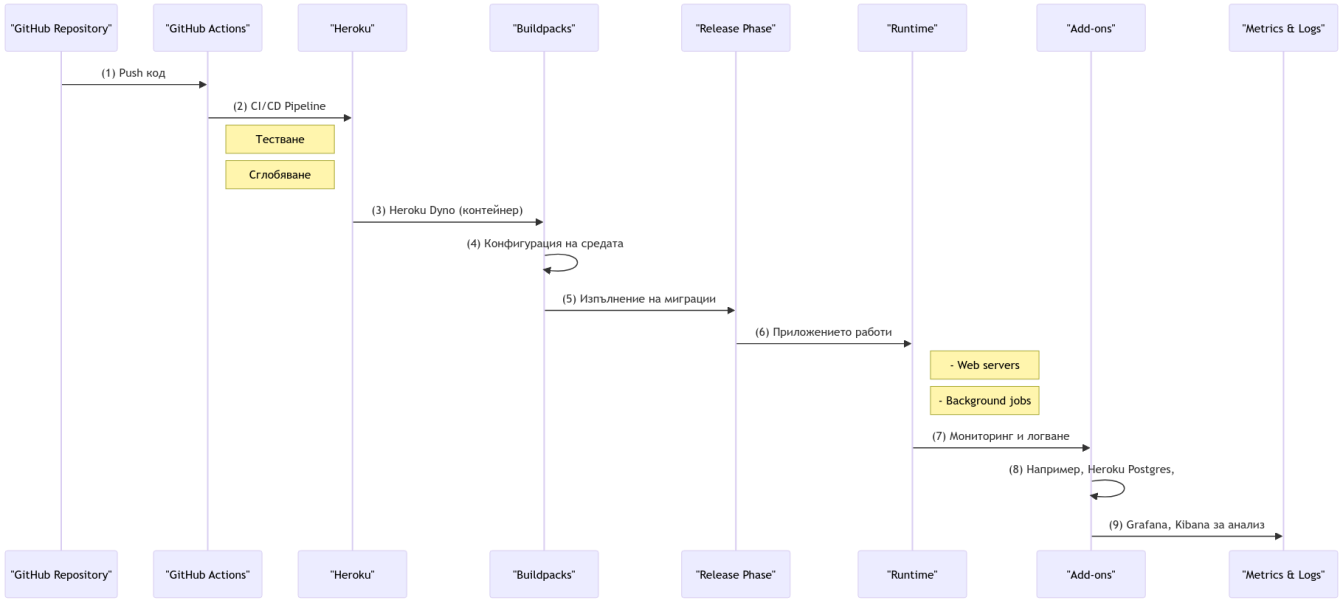


Фиг.3 Heroku Dashboard

Интеграцията на Heroku в процеса на разработка и разгръщане на "AI Mental Assistant App" предлага удобна, мащабируема и ефективна платформа за разгръщане на приложения. Това не само улеснява бързото пускане на нови версии, но също така подобрява

управлението, мониторинга и мащабирането на приложението, което е критично за поддържането на висока производителност и отлично потребителско изживяване. Heroku предоставя мощни инструменти, които могат значително да подобрят цикъла на разработка и да улеснят непрекъснатото подобрение на приложението.

Тази схема на инфраструктурата и процесите, включени в разгръщането и управлението на приложението в Heroku, подкрепяйки високата наличност и улеснявайки мониторинга и трабълшутинга:



Фиг. 4 Инфраструктурата и процесите на разгръщане и управление на приложението в Heroku

Push Код

Публикуване на направените промени в кода в репозитория на GitHub. Това може да включва нови функции, подобрения или корекции на бъгове. Това е началната точка за всеки цикъл на разработка, където кодът се актуализира и подготвя за следващите етапи на тестване и разгръщане.

CI/CD Pipeline

- **Инструмент:** GitHub Actions
- Автоматично извършва тестване и сглобяване на кода при всяко push събитие.

Това включва стартиране на автоматизирани тестове за да се провери дали новият код отговаря на очакваните стандарти и не нарушава съществуващата функционалност. Автоматизацията на тези процеси увеличава ефективността и намалява възможността за човешка грешка при тестване и разгръщане.

Heroku Dyno

След като кодът е успешно преминал през CI/CD pipeline, той се разгръща в Dyno в Heroku, където приложението ще бъде хоствано. Dyno предоставя изолирана среда, където приложението може да работи, гарантирайки ресурси и сигурност.

Buildpacks автоматично конфигурират средата на приложението и инсталират нужните зависимости, което е ключово за стартирането на приложението.Те позволяват инфраструктурата и зависимостите се управляват автоматично.

Release Phase е Стъпка в Heroku, където се изпълняват скриптове за миграции и други задачи преди приложението да стане достъпно онлайн. Тя гарантира, че всички необходими предварителни настройки са изпълнени и приложението е напълно готово за стартиране.

Runtime е фазата, в която приложението работи активно, обслужвайки заявки от потребители. Нейната ефективност е критична за представянето и отзивчивостта на приложението.

За Мониторинг и Логване са използвани инструментите **Grafana** и **Kibana.** Тяхната функционалност е непрекъснато наблюдение и записване на дейността на приложението, включително събиране на метрики и логове. Така тези данни помагат за идентифициране на проблеми в работата на приложението, анализ на производителността и оптимизация на ресурсите.

Add-ons са допълнителни услуги, предлагани от Heroku, като бази данни и инструменти за анализ. Те предлагат интегрирани решения за увеличаване на функционалността и производителността на приложението.

Схемата на фиг.4 представя цялостния жизнен цикъл на приложението от разработка до разгръщане и поддръжка, гарантирайки, че всички аспекти на процеса са автоматизирани, мониторирани и оптимизирани за най-добро представяне.

Структурата на директорията на приложението

Структурата на директорията на прилоприложението "AI Mental Assistant App" е организирана по начин, който разделя логиката на приложението, ресурсите и зависимостите му по чист и интуитивен начин. Това ще улесни разработката и поддръжката на кода.

AI_Mental_Assistant_App/

```
│
├── client/                # Директория за фронтенд частта (React)
│   ├── public/            # Статични файлове, като index.html
│   ├── src/               # Изходен код на React приложението
│   │   ├── components/    # React компоненти
│   │   ├── services/      # Сервиси за връзка с API
│   │   ├── App.js         # Основният React компонент
│   │   └── index.js       # Входна точка на React приложението
│   ├── package.json       # NPM зависимости и скриптове за фронтенда
│
├── server/                # Директория за бекенд частта (Node.js + Express)
│   ├── config/            # Конфигурационни файлове
│   ├── models/            # Mongoose модели за MongoDB
│   ├── routes/            # Express маршрути
│   ├── services/          # Бизнес логика и интеграции с външни API-та
│   ├── app.js             # Конфигурация на Express приложението
│   └── server.js          # Стартиране на Express сървъра
│   ├── package.json       # NPM зависимости и скриптове за бекенда
│
├── .env                   # Конфигурационни променливи за околната среда
├── .gitignore             # Файлове и директории игнорирани от git
├── README.md              # Документация на проекта
└── package.json           # Общи NPM зависимости и скриптове
```

Проектът е разделен на две основни директории – client за фронтенда и server за бекенда. Това разделение улеснява управлението на зависимостите и разработката на отделните части на приложението.

- **Компонентен подход за React:** Всички React компоненти са организирани в директорията components, което улеснява намирането и поддръжката им.

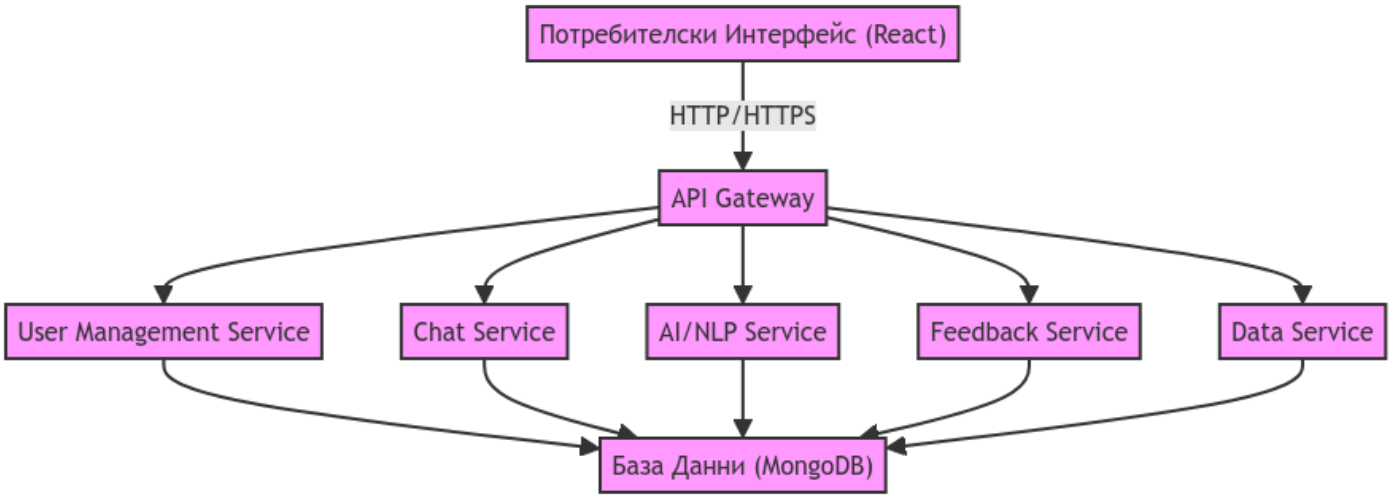
- **Модели и маршрути за Express:** Директориите models и routes в сървърната част съдържат съответно дефинициите на данните (чрез Mongoose модели) и маршрутите (чрез Express), които определят API ендпойнтите.

- **Services:** И бекендът, и фронтендът имат директории services, които обикновено съдържат логиката за обработка на бизнес процесите и връзката с външни API-та.

- **Конфигурация и секрети:** Използването на .env файлове помага за съхранението на конфигурационни променливи и секрети извън контрола на версиите за повишаване на сигурността.

ГЛАВА 3: Разработка на платформата

3.1 Основни Архитектурни Компоненти



Фиг.5 Микросървисна Архитектура

- User Management Service управлява регистрацията, аутентикацията и профилите на потребителите.
- Chat Service обработва комуникацията в реално време между потребителите и AI моделите.
- AI/NLP Service използва NLP (естествен езиков процесинг) за анализ и генериране на отговори, базирани на машинно обучение.
- Feedback Service събира и обработва обратна връзка от потребителите за подобряване на услугите.

- Data Service управлява взаимодействието с базата данни и обработката на данни.

Тази архитектура и технологичен стек са избрани с цел да предоставят гъвкава, скалируема и сигурна платформа, която да отговори на нуждите на потребителите за психическа подкрепа и да осигури лесна поддръжка и разширение в бъдеще.

Основните технологии, които са използвани за Backend разработка са: Node.js - за създаване на мощен и гъвкав сървърен слой поради неговата богата екосистема от библиотеки. За Frontend разработка: React за уеб - за разработката на интерактивен и модерен потребителски интерфейс.

3.1.1 Компоненти на платформата

Frontend (React)

- socket.io-client: Използва се за създаване на двупосочна комуникация в реално време между клиентите (потребителските устройства) и сървъра. Това е ключово за функционалността на чата на живо.

- HTTP/HTTPS Заявки: Изпращат се към RESTful API, разработен на Node.js, за операции, които не изискват реално времево обновление, като регистрация на потребител, влизане и извличане на история на съобщенията.

Backend **(Node.js + socket.io)**

- socket.io: Слуша за входящи връзки от клиенти и обработва събития като изпращане на съобщения, присъединяване към чат стаи и т.н.
- REST API: Обработва HTTP/HTTPS заявките от frontend за управление на потребителите, сесиите и други не-реалновремеви операции.

Mongoose (за Node.js)

ORM (Object-Relational Mapping) библиотека, която улеснява взаимодействието между Node.js приложенията и MongoDB. Backend използва Mongoose за да създаде, чете, обновява и изтрива данни от базата данни, свързани с потребителски акаунти, чат сесии и съобщения.

Интеграция с AI (OpenAI GPT)

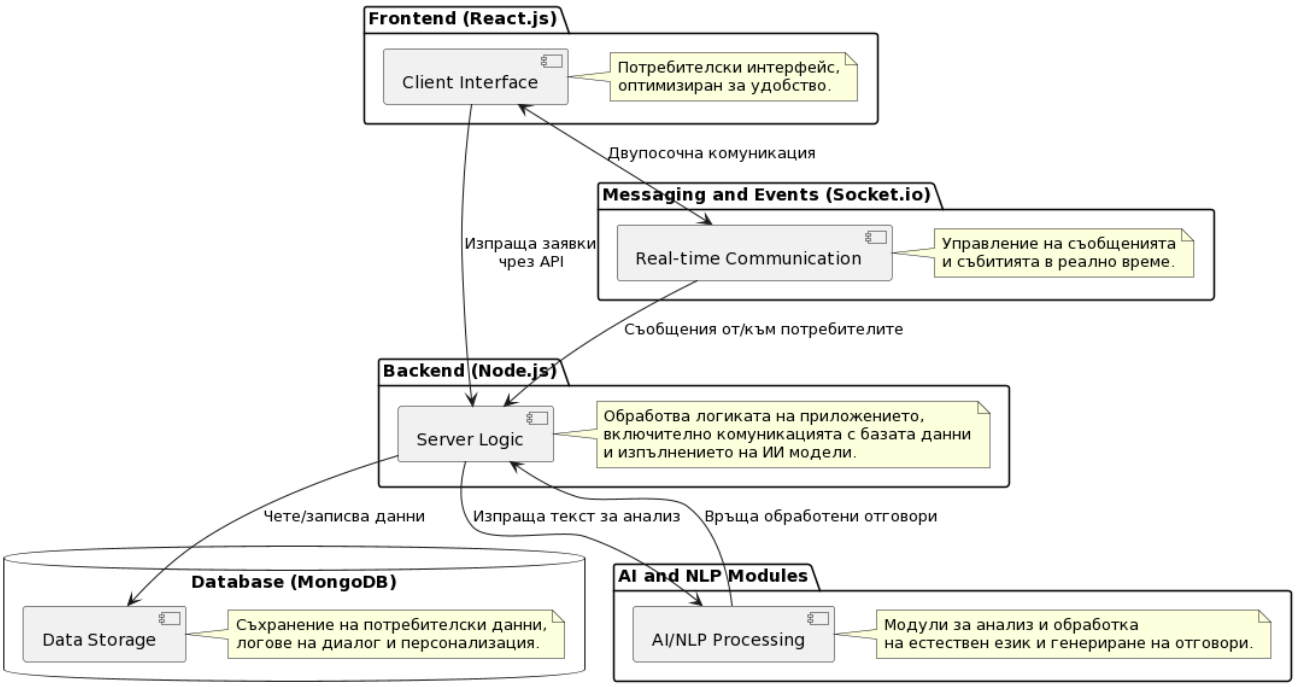
Backend компонентът прави HTTP заявки към OpenAI API, изпращайки текст от потребителските съобщения и получавайки персонализирани отговори от AI модела.

- Обработка и отговори: Получените отговори от AI се препращат обратно към съответния клиент чрез socket.io в реално време, позволявайки на потребителя да вижда отговора в чата на живо.

Deployment

Heroku използва контейнери, наречени dynos, за хостване на приложения. Приложението, включително всички негови компоненти, се разпространява в Heroku dyno, което улеснява мащабирането и управлението.

За интеграция с външни услуги, включително бази данни като MongoDB, Heroku предлага различни добавки, които могат да бъдат лесно интегрирани и управлявани чрез Heroku платформата.



- *Фиг.6* Компоненти на платформата

С тази архитектура, "AI Mental Assistant App" осигурява ефективно взаимодействие между всички компоненти, като поддържа скалируемост, гъвкавост и висока производителност, което е важно за осигуряването на качествена услуга за потребителите.

3.1.2 Чат на живо и Real-time комуникация

Чатът на живо и реално временната комуникация са от съществено значение за "AI Mental Assistant App", тъй като те предоставят директен, интерактивен канал за потребителите да получават подкрепа и насоки. Тази система използва съвременни технологии като React за фронтенда, Node.js и socket.io за бекенда, MongoDB за база данни и AI/NLP услуги за обработка на език и генериране на отговори. Разглеждането на всеки компонент и техния взаимен допир демонстрира как съвместната им работа осигурява ефективно и навременно обслужване на потребителите.

Потребителски интерфейс (React)

Фронтендът на приложението е разработен използвайки React, популярна JavaScript библиотека за създаване на интерактивни потребителски интерфейси. Той използва socket.io-client за създаване на двупосочна комуникационна връзка със сървъра, позволявайки на потребителите да изпращат и получават съобщения в реално време без необходимостта от презареждане на страницата.

Сървър (Node.js + socket.io)

Бекендът управлява всички аспекти на комуникацията в реално време. Използвайки Node.js за сървърната логика и **socket.io** за

управление на WebSockets, сървърът обработва входящите и изходящите съобщения, поддържа чат сесии и координира взаимодействията между различните услуги и базата данни.

API Gateway

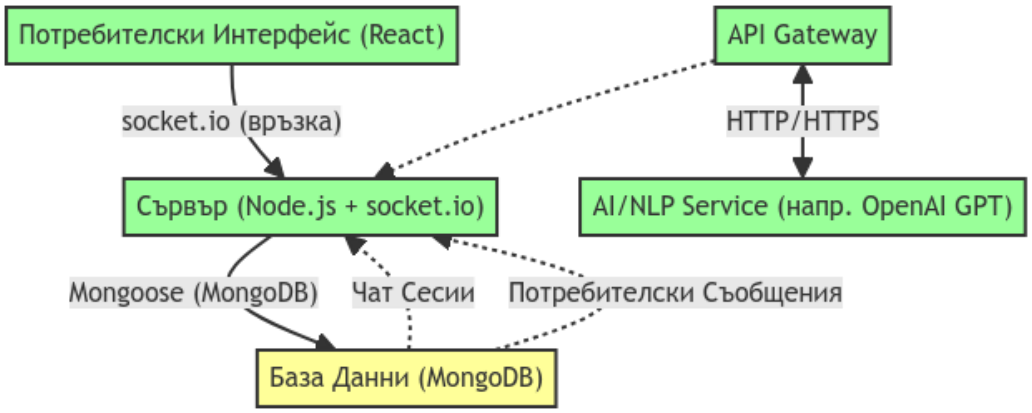
API Gateway служи като централна точка за управление на всички HTTP/HTTPS заявки, пренасочвайки ги към съответните микросервиси. Това включва управление на заявки към AI/NLP услугата, която обработва и генерира отговори на база текста, получен от чат сесиите.

AI/NLP Service

Използва се външна услуга като OpenAI GPT за анализ и обработка на потребителските запитвания, което позволява на системата да предложи интелигентни, контекстуални и персонализирани отговори. Взаимодействието между сървъра и AI услугата се извършва чрез HTTP/HTTPS заявки, което гарантира бърза и сигурна комуникация.

База Данни (MongoDB)

Всички данни, свързани с потребителските профили, чат сесии, съобщения и друга информация, се съхраняват в MongoDB. Използването на Mongoose улеснява взаимодействието между Node.js приложението и MongoDB, осигурявайки лесен за използване интерфейс за моделиране и управление на данни.



Фиг.7 Схема на Чат на живо

– Фронтендът, където потребителите взаимодействат с чат интерфейса. Използва **socket.io-client** за създаване на двупосочна връзка в реално време със сървъра.

– Бекендът управлява връзките на чата, сесиите и съобщенията в реално време чрез **socket.io**. Той също така служи като мост между потребителския интерфейс и базата данни, както и AI/NLP услугата.

– API Gateway- управлява всички входящи HTTP/HTTPS заявки от фронтенда и ги пренасочва към съответния микросервис, включително AI/NLP услугата за обработка на текст и генериране на отговори.

– AI/NLP Service (напр. OpenAI GPT), която еъншна услуга, която обработва запитванията, изпратени от чат сесиите, и генерира

персонализирани отговори. Комуникацията между сървъра и AI услугата се извършва чрез HTTP/HTTPS заявки.

– База Данни (MongoDB): Съхранява всички данни, свързани с потребителите, чат сесиите и съобщенията.

Процес на Комуникация:

– Потребителският интерфейс (React) инициира **socket.io-client** връзка със сървъра.

– Когато потребител изпрати съобщение, то се изпраща през сокет връзката към сървъра.

– Сървърът получава съобщението и може да го изпрати до AI/NLP услугата за анализ и генериране на отговор, ако съобщението изисква такъв.

– Всяко съобщение и отговор се записват в базата данни за съхранение и бъдеща анализа. AI генерираният отговор се изпраща обратно към потребителя през същата сокет връзка.

3.1.3 AI и Машинно обучение

Изкуственият интелект (AI) и машинното обучение (ML) са ключови компоненти в развитието на технологии за подкрепа на

психичното здраве. В контекста на "AI Mental Assistant App", тези технологии позволяват разработката на интелигентни системи, способни да разбират и обработват човешки емоции, да провеждат ефективни диалози и да предоставят персонализирани съвети и подкрепа. Тази секция разглежда различните аспекти на интеграцията на AI и ML в приложението, включително разработката на модели, обучение, интеграция и етични съображения.

Разработка на AI и ML Модели

AI и ML модели в "AI Mental Assistant App" се фокусират върху следните основни задачи:

- Естественообработка на език (NLP): Разработка на модели за разбиране и генериране на естествен език, което включва задачи като анализ на сентименти, класификация на намерения и извличане на информация.
- **Анализ на емоции:** Използване на техники за машинно обучение за анализ на текстови и гласови данни, за да се идентифицират човешките емоции и настроения.
- **Персонализация:** Разработка на препоръчителни системи, които персонализират взаимодействието и отговорите на асистента въз основа на потребителското поведение и предишни взаимодействия.

- **NLP Библиотеки:** Използване на библиотеки като NLTK, SpaCy, и TensorFlow или PyTorch за построяване на модели за обработка на език.
- **Обучение на данни:** Събиране и предварителна обработка на обучаващи данни, включително текстови корпуси от публични източници и специализирани набори от данни, свързани с психичното здраве.
- **Събиране на Данни:** Идентифициране и събиране на адекватни и разнообразни данни, които отразяват различни аспекти на човешките емоции и диалоги.
- **Анотация:** Ръчно анотиране на данни за обучение с помощта на експерти по психология и езикознание за усъвършенстване на точността и надеждността.
- **Моделиране:** Избор и конфигуриране на подходящи алгоритми и архитектури на модели, включително невронни мрежи и ансамблови методи.
- **Оценка:** Провеждане на оценка на моделите с помощта на метрики като точност, прецизност и отзивчивост.
  Използване на крос-валидация и тестови набори от данни за проверка на обобщаващата способност на моделите.

Интеграция на AI и ML в Приложението

1. **Внедряване на Модели**

Разработка на RESTful APIs за интеграция на ML модели с други компоненти на приложението, като чат и управление на потребителски данни. Използване на микросервисна архитектура за разгръщане и мащабиране на AI функционалности.

2. **Етични Съображения и Защита на Данните**

Потребителите трябва да разбират как техните данни се използват и имат контрол върху тяхното използване необходимо е провеждане на анализи за идентифициране и коригиране на всякакви видове предубеждения в обучаващите данни и моделите.

Интеграцията на AI и машинното обучение в "AI Mental Assistant App" е фундаментална за разработката на ефективни и интелигентни функционалности за подкрепа на психичното здраве. Тези технологии позволяват на приложението да предложи персонализирана помощ, разбиране на естествения език и адаптивно взаимодействие с потребителя.

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

# Примерен датасет
sentences = [
    "I feel great today!",
    "I am so sad, it's a bad day.",
    "I am excited about the concert.",
    "I am afraid I won't make it on time."
]

# Емоциите са кодирани като: радост = 0, тъга = 1, вълнение = 2, страх = 3
labels = [0, 1, 2, 3]

# Токенизация и подготовка на текстови данни
tokenizer = Tokenizer(num_words=1000, oov_token="<OOV>")
tokenizer.fit_on_texts(sentences)
sequences = tokenizer.texts_to_sequences(sentences)
padded_sequences = pad_sequences(sequences, padding='post')

# Създаване на модела
model = Sequential([
    Embedding(1000, 16, input_length=padded_sequences.shape[1]),
    LSTM(64, return_sequences=True),
    Dropout(0.2),
    LSTM(32),
    Dense(16, activation='relu'),
    Dense(4, activation='softmax')
])

# Компилиране на модела
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Обучение на модела
model.fit(padded_sequences, labels, epochs=10)

# Предсказване на емоции
test_sentence = ["I am so happy to see you!"]
test_seq = tokenizer.texts_to_sequences(test_sentence)
test_padded = pad_sequences(test_seq, padding='post', maxlen=padded_sequences.shape[1])
prediction = model.predict(test_padded)
emotion = ['Joy', 'Sadness', 'Excitement', 'Fear'][tf.argmax(prediction[0]).numpy()]
print("Predicted Emotion:", emotion)
```

Фиг.8 Код за NLP модел

За да се интегрира AI и машинно обучение в "AI Mental Assistant App", особено за естественообработка на език (NLP), анализ на емоции и персонализирани препоръки, можем да използваме Python заради богатата екосистема от библиотеки в тази област. Ето примерен код, който демонстрира как може да се създаде основен NLP модел, използвайки библиотеката TensorFlow и Keras за обработка на език и класификация на емоции:

- **Подготовка на данните**

Tokenizer:

Този клас от библиотеката Keras се използва за конвертиране на текстовите данни в числови последователности. Всеки уникален дума в датасета се присвоява на уникален индекс. Това е необходимо, защото моделите за машинно обучение работят с числа, а не със суров текст.

pad_sequences:

След като всички текстове са преобразувани в списъци от индекси, pad_sequences се използва за уеднаквяване на дължината на всички входни последователности. Това става чрез добавяне на нули в края на по-кратките последователности, ако те са по-малки от максималната допустима дължина. Това е важно, защото невронните мрежи изискват входни данни с еднакъв размер.

- **Създаване на модела**

Embedding слой:

Този слой преобразува числовите последователности от токени в гъсти вектори на по-високо ниво. Всеки индекс се картографира към вектор с фиксиран размер, което позволява модела да научи по-добро представяне на думите в контекста на задачата.

LSTM слоеве:

Два слоя от LSTM (Long Short-Term Memory) са използвани за анализ на времевите зависимости в текста. LSTM е вид рекурентна невронна мрежа, която е способна да запомня информация за дълги периоди от време, което е идеално за обработка на естествен език.

Dropout:

Слой за предотвратяване на преобучване чрез случайно "изключване" на неврони по време на обучението, което помага да се предотврати зависимостта на модела от конкретни аспекти на тренировъчните данни.

Dense слоеве:

Два плътни (Dense) слоя се използват за класификация на емоциите. Последният слой обикновено има брой изходи, равен на броя на класовете емоции, и използва софтмакс функция, която преобразува логитите (резултати от предишния слой) в вероятности.

- **Компилиране и обучение на модела**

Моделът се компилира с оптимизатор 'adam', който е ефективен и широко използван метод за оптимизация, и функция на загубата

'sparse_categorical_crossentropy', подходяща за многокласова класификация където класовете са представени като цели числа.

Моделът се обучава върху подготвените тренировъчни данни, като по време на процеса се използват валидационни данни за оценка на производителността и избягване на преобучване.

Предсказване на емоции

След като моделът е обучен, той може да се използва за предсказване на най-вероятната емоция на нови данни. Това става чрез подаване на нов текст, който е подготвен по същия начин като тренировъчния набор, и моделът връща вероятностите за всяка класифицирана емоция.

Този процес обхваща целия път от подготовката на данните до предсказванията и позволява на модела да анализира и класифицира емоциите в текста, което е централно за функционалността на "AI Mental Assistant App".

3.2 Микросървисна Архитектура

Микросървисната архитектура е модел за разработка на софтуер, който структурира приложенията като съвкупност от слабо свързани услуги. В контекста на "AI Mental Assistant App", микросервисната архитектура позволява модуларност, гъвкавост и мащабируемост, което е идеално за управление на различните функционалности на приложението. Тази секция разглежда пет ключови сервиса: User Management Service, Chat Service, AI/NLP Service, Feedback Service и Data Service.

User Management Service

User Management Service управлява всички аспекти на потребителските данни и сесии, включително регистрация, вход, профилна информация и сесионни данни. Основната му задача е да осигури сигурен и ефективен начин за управление на потребителски идентификатори и удостоверяване.

- Автентикация: Използването на JWT (JSON Web Tokens) за сигурна автентикация и OAuth за интеграция със социални медии.
- База данни: Интеграция с база данни като MongoDB или PostgreSQL за съхранение на потребителска информация.
- APIs: RESTful API интерфейс за интеграция с други микросервиси.

Chat Service

Chat Service обработва всички аспекти на комуникацията в реално време между потребителите и виртуалния асистент. Този сервис управлява сесиите на чат, съхранява историята на съобщенията и улеснява динамичното взаимодействие.

- Използване на WebSocket за реално време комуникация.
- Съхранение на съобщения: Внедряване на системи за съхранение като Redis или Cassandra за бърз достъп и ретенция на съобщения.
- Микросервисна интеграция: Взаимодействие с AI/NLP Service за обработка на входящи съобщения.

AI/NLP Service

AI/NLP Service е сърцето на "AI Mental Assistant App", осигуряващ анализ и обработка на естествен език, разбиране на потребителските запитвания и генериране на адекватни отговори.

- Използване на Google Cloud Natural Language
- Разработка на модели за машинно обучение, които могат да се обучават и оптимизират въз основа на обратна връзка.
- Интеграция: Тясно сътрудничество с Chat Service за осигуряване на своевременни и точни отговори.

Feedback Service

Feedback Service управлява събирането и анализа на потребителска обратна връзка, което е ключово за непрекъснатото подобрение на приложението.

- Data Analytics: Използване на аналитични инструменти за обработка и визуализация на обратна връзка.
- Интеграция с Data Service за съхранение и анализ на събраните данни.

Data Service

Data Service управлява централизирано съхранение и обработка на данни за всички микросервиси, осигурявайки консистентност и надеждност на данните в приложението.

- База Данни: Използване на скалируеми и гъвкави бази данни като PostgreSQL или MongoDB.
- Data Lakes: Разработка на Data Lakes за анализ и хранене на големи обеми неструктурирани данни.
- APIs: Създаване на API за безпроблемна интеграция и достъп до данни от други сервиси.

Микросервисната архитектура на "AI Mental Assistant App" предлага модуларност, улеснява мащабирането и подобрява устойчивостта на системата. Всяка от гореспоменатите услуги играе важна роля в общата функционалност на приложението и неговата способност да предоставя ефективна подкрепа за психичното здраве на своите потребители. Тази структура позволява ефективно разпределение на ресурси, бърза итерация и подобрения, както и по-добра устойчивост на отказ.

3.3 Интерфейс на потребителя и взаимодействие с виртуалния асистент

Интерфейсът на потребителя е разработен с фокус върху удобството и интуитивността. При влизане в платформата, потребителите се посрещат от чат прозорец, където могат директно да започнат разговор с виртуалния асистент. Възможно е да се задават въпроси или да се изразяват чувства в свободна форма, което позволява на асистента да предоставя персонализирани съвети, упражнения за справяне или просто утешителни думи.

Интерфейсът също така предлага опции за персонализация, като например настройки за предпочитания в комуникацията или теми, които потребителите искат да избегнат.

Взаимодействие с виртуалния асистент

Виртуалният асистент е проектиран да имитира емпатично и разбиращо човешко взаимодействие чрез използването на напреднали ИИ и NLP технологии. Основни характеристики на взаимодействието включват:

- Възпроизвеждане на естествен език: Потребителите могат да формулират своите заявки в свободна форма, както биха говорили с човек, което позволява на асистента да анализира и да отговори адекватно.

- Емпатични отговори: Асистентът използва отговори, които отразяват разбиране и съчувствие към състоянието и нуждите на потребителя, като по този начин се стреми да изгради доверие и да намали чувството на изолация.

- Предоставяне на персонализирани ресурси: Въз основа на разговора, асистентът може да предложи персонализирани съвети, упражнения за самопомощ или информация за професионални услуги за психично здраве.

- Обучение и адаптация: Изкуственият интелект се обучава от всеки разговор, което позволява непрекъснато подобрение на качеството на взаимодействието и персонализацията на отговорите.

Проектиране на Интерфейса

Използван е минималистичен дизайн, който улеснява навигацията и фокусира вниманието върху чат прозореца. Чиста и ненатрапчива оформление, което не отвлича вниманието от основната задача комуникацията с виртуалния асистент.

- **Чат Прозорец:**

Разполага с ясно видимо място за въвеждане на текст и история на разговора, която потребителите могат лесно да прелистват.

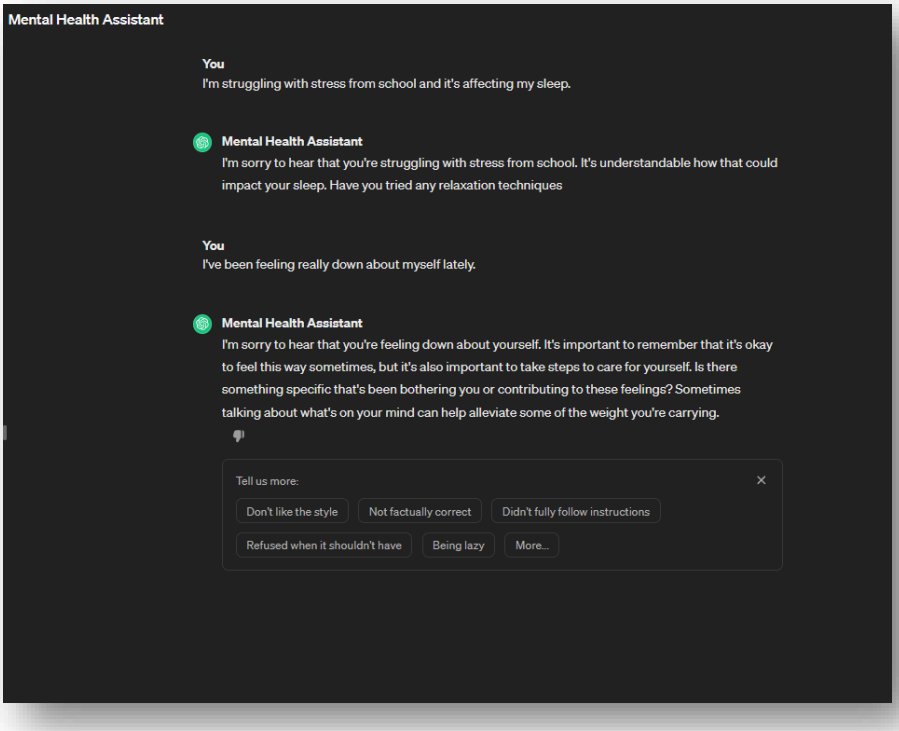
Позволява на потребителите да въвеждат своите заявки в свободна форма, като по този начин имитирате разговор с човек.

Кодът на фиг.9 демонстрира как да се създаде основен чат прозорец, който включва поле за въвеждане на съобщения и област за показване на историята на чат:

В този пример, socket.io-client се използва за свързване със сървъра и за изпращане/получаване на съобщения. useState хукът управлява състоянието на текущото съобщение и историята на чата, докато useEffect хукът слуша за нови съобщения от сървър

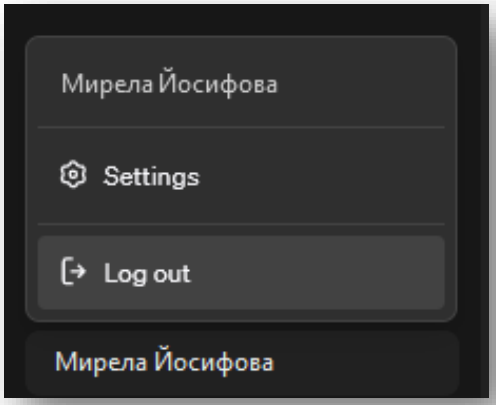
```jsx
import React, { useState, useEffect } from 'react';
import io from 'socket.io-client';

const socket = io.connect('http://localhost:3000');

function Chat() {
  const [message, setMessage] = useState('');
  const [chatHistory, setChatHistory] = useState([]);

  useEffect(() => {
    socket.on('chat message', (msg) => {
      setChatHistory((chatHistory) => [...chatHistory, msg]);
    });
  }, []);

  const sendMessage = (e) => {
    e.preventDefault();
    if (message !== '') {
      socket.emit('chat message', message);
      setMessage('');
    }
  };

  return (
    <div className="chat-container">
      <ul className="chat-history">
        {chatHistory.map((msg, index) => (
          <li key={index}>{msg}</li>
        ))}
      </ul>
      <form className="chat-form" onSubmit={sendMessage}>
        <input
          type="text"
          value={message}
          onChange={(e) => setMessage(e.target.value)}
          placeholder="Type a message..."
          className="chat-input"
        />
        <button type="submit" className="send-button">Send</button>
      </form>
    </div>
  );
}

export default Chat;
```

Фиг.9 Чат прозоц

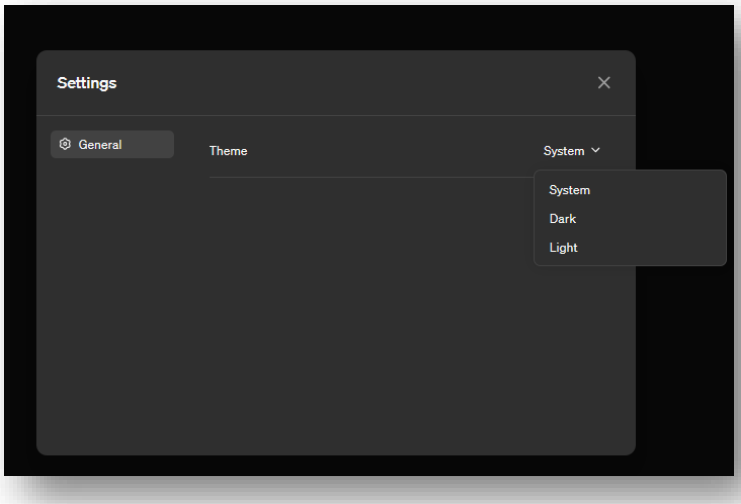


Фиг.10 Чат Интерфейс

Настройки за персонализация, които позволяват на потребителите да адаптират интерфейса и взаимодействието според своите предпочитания. Включени са опции за избор на теми, които потребителите предпочитат или искат да избегнат



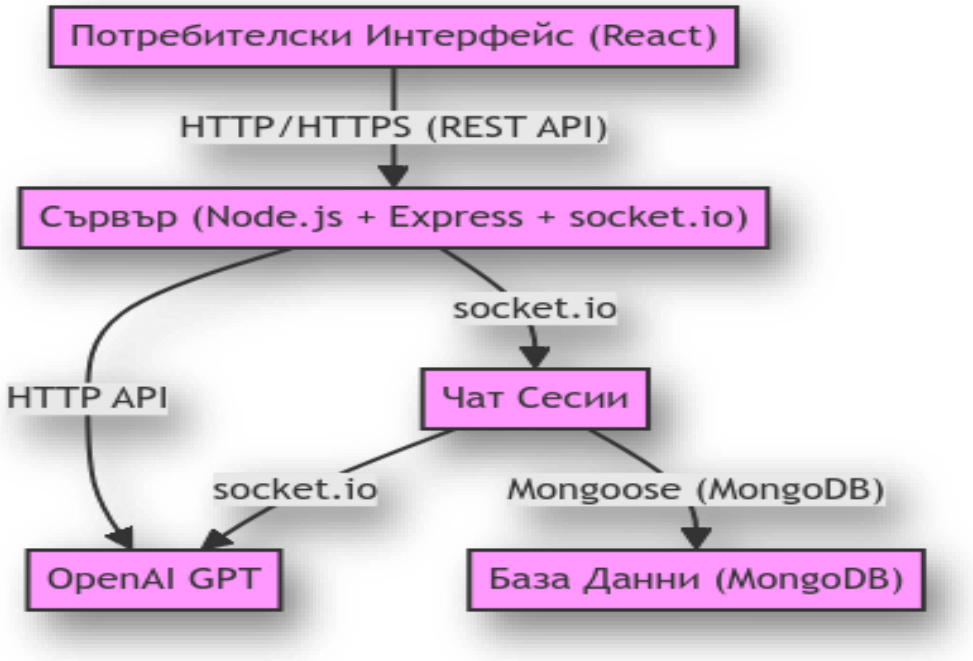
.



Фиг. 11 Персонализирани опции

Ефективното взаимодействие между компонентите на "AI Mental Assistant App" е критично за предоставянето на надеждни и своевременни услуги на потребителите. Тази секция детайлно разглежда как различните компоненти на архитектурата

комуникират помежду си, подчертавайки механизмите на взаимодействие и техните взаимоотношения.



Фиг. 12 Компоненти на архитектурата

Потребителите взаимодействат с приложението чрез React базирания интерфейс. Те изпращат съобщения и получават отговори чрез интерфейса.

Backend сървърът обработва както HTTP/HTTPS заявки чрез REST API (за аутентикация, управление на потребителски данни и

т.н.), така и реално времева комуникация чрез socket.io за функционалността на чата на живо.

– Използва HTTP API за комуникация с OpenAI GPT, изпращайки потребителски въпроси и получавайки интелигентни, персонализирани отговори от AI модела.

– Всички потребителски данни, история на чат сесиите и съобщения се съхраняват в MongoDB. Mongoose се използва за улеснение на взаимодействието между Node.js приложението и MongoDB.

– Приложението се разпространява и управлява чрез Heroku, където различни компоненти като базата данни могат да бъдат добавени като услуги чрез Heroku Add-ons.

3.4 Предизвикателства при разработката и прилагането на платформата

Разработката и прилагането на сложни платформи за подкрепа на психичното здраве като "AI Mental Assistant App" са свързани с множество технически, оперативни и етични предизвикателства. Тези предизвикателства изискват внимателно планиране, стратегическо решаване на проблеми и иновативен подход. Тази секция разглежда ключовите предизвикателства, с които се сблъсква проектът, и предлага възможни решения за тяхното преодоляване.

Интеграция на други технологии

Различните компоненти на платформата изискват интеграция на множество технологии, включително React, Node.js, MongoDB, socket.io, и AI/NLP услуги, което може да доведе до сложности в съвместимостта и комуникацията между тях.

- Решение: Използването на контейнеризация с Docker и оркестрация с Kubernetes може да помогне за стандартизиране на развойната среда и улесняване на интеграцията и разгръщането на компонентите.

Поддържането на висока производителност и надеждност при мащабиране на приложението за голям брой потребители е сложно и ресурсоемко

- Решение: Имплементирането на микросервисна архитектура позволява еластично мащабиране на отделни компоненти на приложението. Използването на

автоматични мащабиращи групи и натоварване балансери също може да подобри управлението на натоварванията.

Управление на данните

Съхранението, обработката и анализът на големи обеми данни от чат сесии и потребителски взаимодействия изискват комплексни решения за управление на данни и защита на личната информация. Необходимо е да се използва на технологии за big data анализ като Apache Spark и осигуряване на съответствие с GDPR и други законодателни изисквания за защита на данните.

Обновяване и поддръжка

Регулярното обновяване и поддръжка на приложението за поддържане на сигурността и актуализиране на функционалността могат да бъдат ресурсоемки.

- Решение: Автоматизация на процесите за CI/CD с инструменти като Jenkins или GitHub Actions за улеснение на непрекъснатите доставки и разгръщане. Създаване на автоматизирани тестови сценарии за увереност в качеството на софтуера.

Защитата на личната информация

Защитата на личната информация и осигуряването на анонимността на потребителите е от съществено значение, особено при приложения за психично здраве.

Важна е разработка на строги политики за поверителност и сигурност, включително използване на силно шифроване за

съхранение и трансфер на данни и осигуряване на прозрачност относно употребата на данни.

Безпристрастност на AI

Гарантирането, че AI моделите не проявяват предвзетост и дискриминация към определени групи потребители, е ключово за етичното им използване.

- **Решение:** Разработване на механизми за мониторинг и ревизия на AI модели за идентифициране и коригиране на всяка потенциална предвзетост. Инкорпориране на разнообразие в обучаващите данни и използване на техники за машинно обучение, които промотират справедливост.

3. 5 Възможности за бъдещо развитие и подобрение

"AI Mental Assistant App" е платформа, която е разработена с цел да подкрепя психичното здраве на потребителите чрез интелигентни чат услуги. Въпреки постигнатия напредък, има множество възможности за бъдещо развитие и подобрение на платформата. Тази секция изследва различни аспекти, които могат да бъдат усъвършенствани, като се фокусира върху иновации в технологията, разширяване на функционалността и усъвършенстване на потребителския опит.

Усъвършенстване на AI и машинното Обучение

Въпреки напредъка в NLP, разбирането на сложни човешки емоции и намерения остава предизвикателство.

- **Решение:** Интеграция на по-нови и по-мощни модели за естествена обработка на езика, като GPT-4 или по-нови версии, които предлагат подобрена способност за разбиране и генериране на текст. Изследване на техники за трансферно обучение, за да се адаптират общите модели специфично към контекста на психичното здраве.

Текущите системи за анализ на емоции могат да пропускат нюансите в потребителското изразяване, особено в многоезични настройки.

- **Решение:** Разработка на модели, които използват мултимодален подход, комбиниращ текстови данни с гласови и визуални данни, за по-добро разбиране на емоциите. Интегриране на адаптивни алгоритми, които се учат от интеракции в реално време и се адаптират към културните особености на потребителите.

Разширение на функционалността

Потребителите често търсят психологическа помощ, когато вече са изправени пред сериозни затруднения.

- **Решение:** Създаване на връзки със здравни системи и професионалисти, за да се осигури бърза професионална интервенция, когато системата идентифицира потребител в криза. Разработване на протоколи за автоматично уведомление на специалисти при определени тригери.

Задържането на потребителите и поддържането на тяхното ангажиране може да бъде предизвикателство.

- **Решение:** Разработка на интерактивни елементи, като интегриране на виртуална реалност и геймификационни техники, за да се увеличи ангажираността и ефективността на терапевтичните сесии. Предоставяне на виртуални сценарии, които имитират социални ситуации или предоставят релаксиращи среди.

Подобряване на интерфейса и достъпността

Сложните интерфейси могат да отблъскват потребители, особено тези с ограничени технологични умения или специални нужди, за това той трябва да е интуитивен, лесен за използване . Като се използват гласови команди и адаптивни текстови формати.

Ограничената езикова поддръжка може да препятства достъпа до услуги за немалка част от потребителите.

- Решение: Разширяване на езиковата поддръжка в приложението, включително добавяне на допълнителни езици и диалекти. Работа с местни експерти и лингвисти за осигуряване на точни и културно адекватни преводи.

3.6 Предизвикателства при разработката и прилагането

Разработката и прилагането на комплексни софтуерни решения като "AI Mental Assistant App" съпътства редица предизвикателства, които могат да възпрепятстват ефективността и успеха на проекта. Тези предизвикателства варират от технически и операционни до етични и регулаторни въпроси. Тази глава изследва ключовите предизвикателства, свързани с разработката и прилагането на платформата, и предлага стратегии за тяхното преодоляване.

Технически Предизвикателства

Интеграцията на различни технологични стекове и платформи, като React, Node.js, MongoDB, и AI/NLP услуги представлява сложност поради потенциални несъвместимости и конфликти. Решението е използването на контейнеризация (напр. Docker) и микросервисна архитектура за осигуряване на съвместимост и изолация на компонентите. Стандартизиране на API спецификациите и протоколите за комуникация.

Справянето с високото натоварване и осигуряването на непрекъсваема услуга изисква ефективно скалиране и управление на ресурсите. Решението е прилагане на автоматизирани мащабируеми решения и натоварване балансиране чрез облачни

услуги като AWS или Azure. Използване на автоматични скалиращи политики и ресурсен мониторинг.

Оперативни предизвикателства

Безопасното управление и защита на чувствителните потребителски данни, особено в контекста на психичното здраве, изисква строги мерки за сигурност и поверителност.

- Имплементация на надеждни криптографски методи за шифроване на данни и строго съответствие с регулации като GDPR. Въвеждане на роли и политики за достъп, базирани на най-малкия необходим привилегии.

- Осигуряването на висока надеждност и непрекъснатост на услугите изисква сложни стратегии за възстановяване при срив. За разработка на робустни стратегии за бекъп и възстановяване. Използване на аварийни възстановителни планове и разпределени данни центрове за минимизиране на прекъсванията

3.7 Етични и регулаторни предизвикателства

Разработката на софтуерни приложения, които събират и обработват лична информация, изисква строго спазване на етични норми и законодателни изисквания. Решението е внедряване на политики за поверителност, които са ясни, прозрачни и съобразени с международни регулации. Провеждане на редовни юридически и етични прегледи на приложението.

Предубежденията в обучаващите данни могат да доведат до нежелани и несправедливи изкривявания в поведението на AI системите. Трябва да се използват методи за обективизация и дебиазиране в процеса на обучение на модели. Внедряване на процедури за регулярен аудит и преглед на моделите за избягване на предубеждения.

Предизвикателствата при разработката и прилагането на "AI Mental Assistant App" изискват целенасочени усилия за преодоляване. Техническите и оперативни стратегии за решаване на проблемите трябва да бъдат съчетани със стриктно спазване на етични и регулаторни стандарти. С такъв интегриран подход, "AI Mental Assistant App" може да продължи да разширява своята функционалност и да подобрява своите услуги, като същевременно остава надежден и доверен ресурс за подкрепа на психичното здраве.

Заключение

"AI Mental Assistant App" представлява иновативно приложение, разработено с цел подкрепа на психичното здраве чрез използване на изкуствен интелект (AI) и обработка на естествен език (NLP). Това приложение съчетава съвременни технологии с практически психологически подходи, за да предостави на потребителите анонимна, достъпна и персонализирана подкрепа. В рамките на различните глави на дипломната работа бяха разгледани различни аспекти на проекта, като фокусът беше насочен към технологичната архитектура, процесите на разработка, предизвикателствата и потенциалните подобрения на системата.

За разработката на "AI Mental Assistant App" са интегрирани различни технологии, включително React.js за потребителския интерфейс, Node.js и Express за бекенд логиката, MongoDB за управление на данни, и socket.io за реално временна комуникация.

Използването на модерни AI и NLP технологии позволява анализ и обработка на потребителските запитвания, предоставяйки уместни и персонализирани отговори, които подпомагат психологическото благополучие на потребителите. Прииложени са строги мерки за сигурност за защита на потребителската информация, като се гарантира съответствие с международни регулации за защита на данните.

През процеса на разработка и прилагане на "AI Mental Assistant App" има редица технически, операционни и етични предизвикателства:

- За преодоляване на техническите предизвикателства се използваха контейнеризация и микросервисна архитектура.

- Автоматизирани стратегии за мащабируемост и ресурсен мониторинг помагат за поддържане на производителността.
- Разработени бяха политики и процедури за защита на личната информация и обективност на AI моделите.

Дипломната работа предоставя основа за непрекъснато усъвършенстване и разширение. Бъдещите направления включват:

- Интегриране на по-сложни AI модели за подобряване на разбирането и взаимодействието.
- Разработка на функционалности за обработка на мултимедийни данни като глас и изображения.
- Усъвършенстване на персонализационните алгоритми за осигуряване на още по-индивидуализирано изживяване.
- Адаптиране на платформата за различни културни и езикови групи.

"AI Mental Assistant App" демонстрира значителен потенциал за трансформиране на начина, по който хората получават подкрепа за психичното здраве. Въпреки предизвикателствата, систематичните усилия в технологичното развитие, управлението на данни, сигурността, и етичните практики са ключови за успеха на приложението. С продължаващите иновации и ангажираност към подобрения, "AI Mental Assistant App" може да продължи да играе критична роля в подобряването на психичното благосъстояние на своите потребители.

ЛИТЕРАТУРА

- Miner, A. S., Milstein, A., & Hancock, J. T. (2019). Talking to machines about personal mental health problems. Journal of the American Medical Association, 322(13), 1266-1268.
- Vaidyam, A. N., Wisniewski, H., Halamka, J. D., Kashavan, M. S., & Torous, J. B. (2019). Chatbots and conversational agents in mental health: A review of the psychiatric landscape. Canadian Journal of Psychiatry, 64(7), 456-464.
- Luxton, D. D. (2020). Artificial intelligence in psychological practice: Current and future applications and implications. Professional Psychology: Research and Practice, 51(5), 470-478.
- Bendig, E., Erb, B., Schulze-Thuesing, L., & Baumeister, H. (2021). The next generation: Chatbots in clinical psychology and psychotherapy to foster mental health – A scoping review. Verhaltenstherapie, 31(2), 140-150.
- Shum, H.-Y., He, X., & Li, D. (2018). From Eliza to XiaoIce: Challenges and opportunities with social chatbots. Frontiers of Information Technology & Electronic Engineering, 19(1), 10-26.
- Abd-Alrazaq, A., Alajlani, M., Alalwan, A. A., Bewick, B. M., Gardner, P., & Househ, M. (2019). An overview of the features of chatbots in mental health: A scoping review. International Journal of Medical Informatics, 132, 103978.

Резюме на дипломната работа

Темата на дипломната работа е разработването на "AI Mental Assistant App" - приложение за подкрепа на психичното здраве, използващо технологии за изкуствен интелект и обработка на естествен език. Актуалността на темата произтича от нарастващата нужда от достъпни и ефективни решения за психологическа подкрепа в контекста на глобалния ръст на психични заболявания и стрес в съвременното общество. Приложението цели да предостави анонимен, леснодостъпен и персонализиран начин за потребителите да получават незабавна подкрепа.

Обект на изследването

Предметът на изследването е разработването и прилагането на технологии за изкуствен интелект и машинно обучение в контекста на психичното здраве. Обектът включва анализ на потребителското поведение, обработка на естествен език и реализация на реалновремева комуникация чрез вградения чатбот. Изследването се фокусира върху разработването на модулни и микросервисни архитектури, които поддържат мащабируемост и ефективност на приложението.

Задачи на Изследването

Основните задачи, разгледани в работата, включват:

- Разработка на потребителски интерфейс с React, който да осигури интуитивно взаимодействие за потребителите.
- Интеграция на backend системи с Node.js и Express за управление на чат сесии и потребителски данни.
- Имплементация на микросервисна архитектура за гъвкаво мащабиране и поддръжка на приложението.
- Използване на AI и NLP за анализ на потребителски запитвания и генериране на адаптивни отговори.

Обобщения, Изводи и Резултати

Дипломната работа демонстрира, че "AI Mental Assistant App" успешно интегрира различни технологични решения за създаване на ефективна платформа за психологическа подкрепа. Откроени са следните основни резултати:

- Приложението предлага високо ниво на персонализация и точност при отговорите благодарение на интегрираните AI и NLP технологии.

- Системата поддържа висока степен на сигурност и поверителност на потребителските данни, съответствайки на съвременните изисквания за защита на личната информация.

- Потребителският интерфейс е оценен високо от крайните потребители за удобство и лесна навигация.

Развитието на "AI Mental Assistant App" също идентифицира потенциални възможности за бъдещо усъвършенстване, като разширяване на функционалността за поддръжка на повече езици и интеграция с други медицински и здравни платформи. В заключение, проектът представя значим принос в областта на дигиталните технологии за подкрепа на психичното здраве и поставя основите за бъдещи иновации в сектора.

Заключение

"AI Mental Assistant App" представлява значителен напредък в дигитализацията на психичното здраве, като предоставя иновативни решения за поддръжка и лечение на лица с психически заболявания. Продължаващото развитие и усъвършенстване на платформата ще допринесат за нейната ефективност и ще увеличат нейната способност да помага на все повече потребители по света.

Декларация за оригиналност

Долуподписаният / Долуподписаната Мирела Свиленова Йосифова

декларирам, че настоящата работа е мое лично дело.

Декларирам също така, че съм спазил(а) изискванията за авторско право по отношение на използваните източници и не съм използвал(а) неправомерно чужди текстове, без да посоча техния автор и източник.

Заглавие на дипломната работа:

"Разработване на виртуален асистент с изкуствен интелект за подпомагане на психичното здраве "

Дата: 18.04.2024 Подпис:.....