

SmartNotes – Chytré poznámky pro iOS

SmartNotes – Intelligent Notes for iOS

Bc. Martin Šmíd

Diplomová práce

Vedoucí práce: Ing. Jan Plucar, Ph.D.

Ostrava, 2025



Zadání diplomové práce

Student:

Bc. Martin Šmíd

Studijní program:

N0613A140034 Informatika

Téma:

SmartNotes – Chytré poznámky pro iOS
SmartNotes – Intelligent Notes for iOS

Jazyk vypracování:

čeština

Zásady pro vypracování:

V digitální éře je efektivní správa informací zásadní nejen pro profesionály, ale i pro běžné uživatele, kteří pracují s velkým množstvím poznámek v různých formátech (text, fotografie, hlas). Současné nástroje pro zaznamenávání poznámek často postrádají dostatečnou sofistikovanost, což lze vyřešit chytrým systémem využívajícím umělou inteligenci.

Cílem diplomové práce je navrhnout a implementovat aplikaci SmartNotes pro iOS, která umožní spravovat poznámky obsahující text, fotografie a hlasové záznamy. Aplikace nabídne chytré vyhledávání a sumarizaci za pomoci AI, zefektivní správu a analýzu poznámek.

Body zadání:

1. Navrhnut a implementovat systém chytrých poznámek pro iOS platformu, který podporuje multimediální formáty.
2. Implementovat funkci Rich Text, která umožní formátování textu a integraci multimediálních příloh.
3. Integrovat funkci hlasového záznamu a umožnit jeho přepis a analýzu obsahu.
4. Vytvořit modul chytrého vyhledávání, který bude schopen analyzovat obsah poznámek v různých formátech (text, fotografie, hlasové záznamy).
5. Implementovat systém sumarizace poznámek, který na základě klíčových bodů poskytne uživatelům rychlý přehled.
6. Návrh využití umělé inteligence pro implementaci a rozšíření výše zmíněných služeb.
7. Srovnat aplikaci s existujícími aplikacemi.

Seznam doporučené odborné literatury:

- [1] Ravi Sethi, Software Engineering: Basic Principles and Best Practices, Cambridge University Press, 2022
- [2] ML Bear, Build and Learn: AI App Development for Beginners: Unleashing ChatGPT API with LangChain & Streamlit, ISBN-13 : 979-8856256528, 2023
- [3] Ben AuffarthBen Auffarth, Generative AI with LangChain: Build large language model (LLM) apps with Python, ChatGPT and other models, ISBN-13: 978-1835083468, 2023
- [4] Alahari, Jaswanth, et al. Enhancing iOS Application Performance through Swift UI: Transitioning from Objective-C to Swift. In: International Journal for Research Publication and Seminar. 2022. p. 391-401.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Jan Plucar, Ph.D.**

Datum zadání: 01.09.2024

Datum odevzdání: 30.04.2025

Garant studijního programu: prof. RNDr. Václav Snášel, CSc.

V IS EDISON zadáno: 04.11.2024 09:02:40

Abstrakt

Tato práce se zabývá vývojem mobilní aplikace SmartNotes pro iOS, která umožňuje správu textových, hlasových a PDF poznámek. Aplikace využívá nástroje umělé inteligence pro převod řeči na text, sémantické vyhledávání a automatickou summarizaci obsahu. Řešení je postaveno na frameworku React Native s využitím knihovny LangChain v jazyce Python a cloudových AI služeb. Výsledkem je funkční prototyp optimalizovaný pro každodenní použití.

Klíčová slova

chytré poznámky, iOS, mobilní aplikace, React Native, LangChain, umělá inteligence, převod řeči na text, summarizace, sémantické vyhledávání, PDF

Abstract

This thesis focuses on the development of the SmartNotes mobile application for iOS, designed to manage text, voice, and PDF notes. The application leverages artificial intelligence tools for speech-to-text conversion, semantic search, and automatic content summarization. The solution is built using the React Native framework, the LangChain library in Python, and cloud-based AI services. The result is a functional prototype optimized for everyday use.

Keywords

smart notes, iOS, mobile application, React Native, LangChain, artificial intelligence, speech-to-text, summarization, semantic search, PDF

Poděkování

Rád bych na tomto místě poděkoval panu Ing. Janu Plucarovi, Ph.D., za ochotné konzultace a výborné vedení během tvorby této práce.

Obsah

Seznam použitých symbolů a zkratek	8
Seznam obrázků	9
Seznam tabulek	10
1 Úvod	11
2 State of the Art	12
2.1 Mobilní aplikace a jejich vývoj	12
2.2 Poznámkové aplikace	15
2.3 Velké jazykové modely (LLMs)	17
2.4 Převod řeči na text (Speech to Text - STT)	17
2.5 Sumarizace textu	20
2.6 Shrnutí a motivace	21
3 Teoretická část	22
3.1 React Native	22
3.2 Expo	24
3.3 Ukládání a správa dat	25
3.4 Integrace velkých jazykových modelů	26
3.5 Monitoring mobilních aplikací	30
4 Praktická část	32
4.1 Analýza uživatelských požadavků	32
4.2 Bezpečnost a ochrana dat	37
4.3 Architektura	38
4.4 Nastavení vývojového procesu	42
4.5 Příprava vývojových nástrojů	42
4.6 Implementace	43

4.7	Příprava na testování aplikace	48
4.8	Testování aplikace	50
5	Závěr	56
	Literatura	57

Seznam použitých zkratok a symbolů

AI	– Artificial Intelligence
LLM	– Large Language Model
STT	– Speech-to-Text
NLP	– Natural Language Processing
UI	– User Interface
UX	– User Experience
SDK	– Software Development Kit
API	– Application Programming Interface
CI/CD	– Continuous Integration / Continuous Delivery
OTA	– Over-The-Air
JS	– JavaScript
EAS	– Expo Application Services
TLS	– Transport Layer Security

Seznam obrázků

2.1	Přehled a predikce počtu stažení aplikací dle kategorie [3]	13
2.2	Porovnání trendů nativních a multiplatformních technologií [6]	14
2.3	Chronologický přehled vydání Velkých jazykových modelů [11]	18
3.1	Komunikace pomocí JS bridge	23
3.2	Ukázka komunikace v JSI architektuře	23
3.3	Schéma přístupu ke správě a ukládání dat	25
4.1	Vysokoúrovňové schéma architektury aplikace	39
4.2	Aktivita repozitáře od začátku vývoje	42
4.3	Hlavní obrazovka aplikace	44
4.4	Diagram architektury chytrého vyhledávání	45
4.5	Chytré vyhledávání a detail hlasového záznamu	46
4.6	Poznámkový editor	47
4.7	Obrazovky summarizace a hlasového nahrávače	48
4.8	Teplotní mapy obrazovek	51
4.9	Obrazovka relace uživatele	52
4.10	Sentry - Sítový monitoring	53
4.11	Sentry - Mobilní Přehled	55

Seznam tabulek

3.1	Lokální (on-device) LLM modely vydané výrobci mobilních telefonů [28]	28
3.2	Srovnání frameworků pro práci s LLM: LangChain, LlamaIndex a Haystack [31] . . .	29

Kapitola 1

Úvod

V současné digitální éře se množství informací, se kterými jednotlivci i organizace každodenně pracují, neustále zvyšuje. Správa poznámek, ať už ve formě textu, obrázků nebo hlasových záznamů, se stala nedílnou součástí života mnoha uživatelů – od studentů a výzkumníků až po pracovníky ve firmách napříč obory. Přestože na trhu existuje řada nástrojů, které tuto potřebu řeší, často narazíme na jejich omezení z hlediska funkcionality, intuitivnosti nebo možnosti integrace umělé inteligence (AI).

Diplomová práce se zaměřuje na návrh a implementaci chytré poznámkové aplikace SmartNotes pro platformu iOS, která využívá moderní technologie včetně velkých jazykových modelů (LLMs) a převodu řeči na text. Cílem je vytvořit nástroj, který umožní uživatelům efektivně organizovat své poznámky v různých formátech, rychle v nich vyhledávat a automaticky z nich generovat přehledná shrnutí.

Aplikace bude navržena s důrazem na jednoduchost ovládání, ale zároveň na pokročilé AI funkcionality, jako je sémantické vyhledávání, automatická summarizace obsahu a analýza hlasových záznamů. Pro implementaci bude využit multiplatformní framework React Native spolu s moderním vývojovým prostředím Expo, což umožní rychlejší vývoj a snadnou údržbu aplikace.

V úvodních kapitolách práce je provedena analýza současného stavu poznámkových aplikací, vývojových přístupů a relevantních AI technologií. Následně je popsán návrh architektury aplikace a její konkrétní implementace včetně testování a porovnání s existujícími řešeními. Výsledkem této práce je funkční prototyp, který může sloužit jako základ pro další vývoj i případné nasazení do praxe.

Kapitola 2

State of the Art

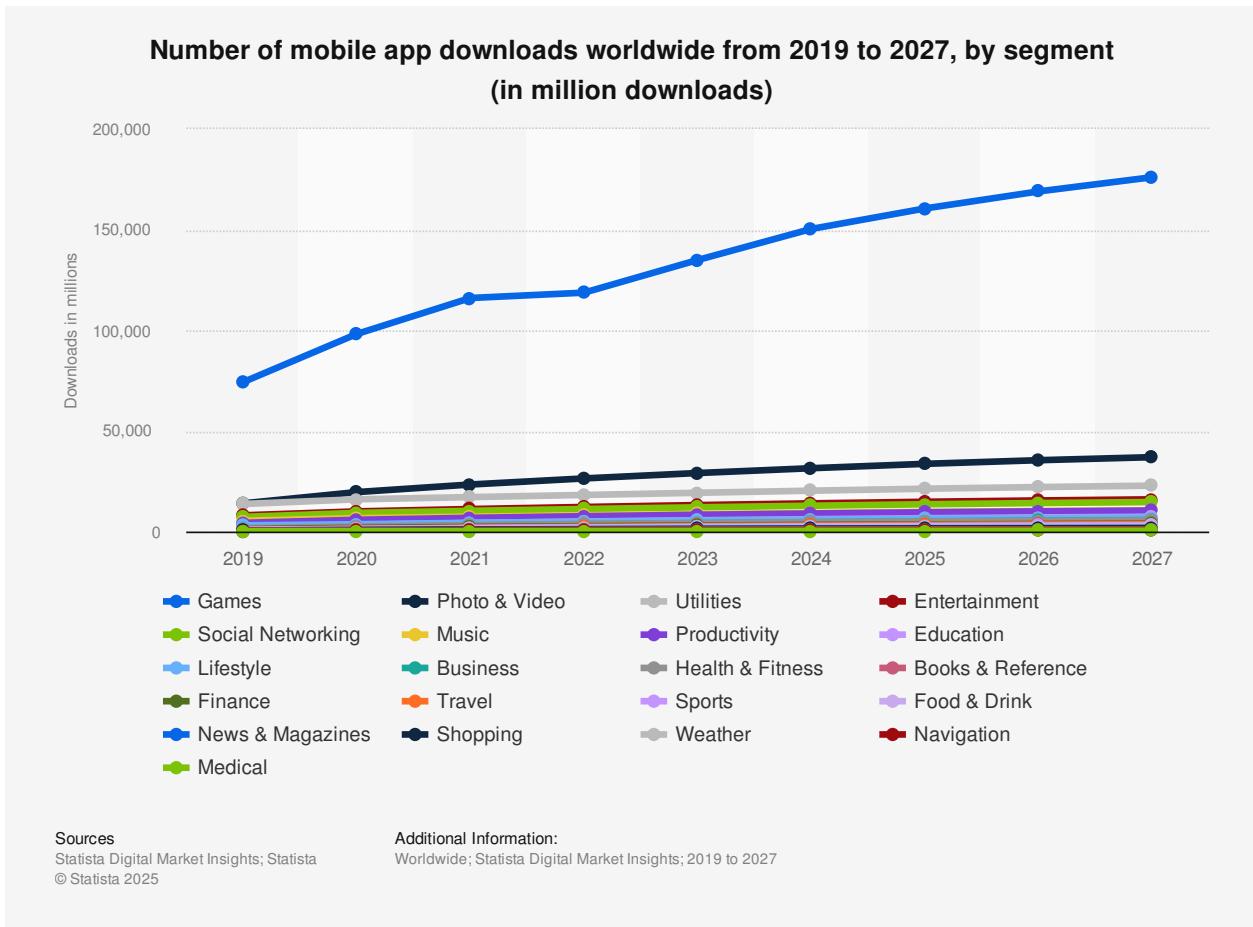
Kapitola State of the Art poskytuje přehled aktuálního stavu výzkumu a technologického pokroku v oblastech souvisejících s mobilními aplikacemi, umělou inteligencí (AI), zpracováním přirozeného jazyka (NLP) a převodem řeči na text (STT). Cílem je vymezit klíčové koncepty, na kterých je práce postavena, a objasnit, z jakých zdrojů a studií jsem čerpal. Kapitola slouží jako základ pro pochopení problematiky a východisko pro další analýzu a návrh řešení.

2.1 Mobilní aplikace a jejich vývoj

Mobilní aplikace představují dynamicky se vyvíjející oblast, kde vzorce užívání a spotřebitelské výdaje ovlivňují různé faktory, včetně technologického pokroku, globálních událostí a měnících se preferencí uživatelů. V posledních pěti letech prošel trh značnou volatilitou, která byla důsledkem zotavení po pandemii COVID-19 a v nedávné době také rostoucí popularity generativních AI aplikací, jako je ChatGPT. [1]

Pokud se podíváme na nejstahovanější aplikace podle kategorií, dlouhodobě dominují hry, následované kategorií obrázky a videa a na třetím místě jsou pomocné aplikace. Celkový přehled počtu stažení v milionech je uveden na obrázku 2.1.

Studie [2] zkoumá, jak zákaznická angažovanost (CE) a zákaznická zkušenosť (CX) ovlivňují kvalitu vztahů a lojalitu zákazníků při interakcích prostřednictvím mobilních aplikací oproti desktopovým prohlížečům. S rostoucím významem mobilních technologií podniky stále více využívají mobilní aplikace k budování zákaznických vztahů, avšak dosud existovalo jen málo poznatků o rozdílech v chování spotřebitelů mezi témito platformami. Pomocí strukturálního modelování rovnic (SEM) a analýzy dat od 420 zákazníků studie ukazuje, že CE a CX mají silnější pozitivní vliv na kvalitu vztahů a lojalitu, pokud interakce probíhají prostřednictvím mobilních aplikací, než když probíhají přes desktopové prohlížeče. Výsledky zdůrazňují strategický význam mobilních aplikací pro posílení zákaznických vztahů a lojality. Studie končí doporučeními pro podniky, které chtějí optimalizovat své digitální strategie zapojením zákazníků.



Obrázek 2.1: Přehled a predikce počtu stažení aplikací dle kategorie [3]

2.1.1 AI v mobilních aplikacích

Umělá inteligence se stala široce dostupnou na počítačích i mobilních zařízeních s uvedením prvního generativního AI chatbota od OpenAI v listopadu 2022. ChatGPT během prvních deseti dnů dosáhl 7,4 milionu stažení, dále Microsoft Copilot a nová čínská aplikace DeepSeek zaznamenaly v roce 2024 také značný počet stažení. Po pandemii se růst mobilních aplikací zpomalil, avšak zavádění generativní AI do editačních a chatovacích aplikací nadále podněcuje zájem uživatelů a zvyšuje počet stažení. [1]

Studie [4] zkoumá transformační dopad generativní umělé inteligence (AI) na digitální platformy, přičemž zdůrazňuje její potenciál v automatizaci tvorby obsahu, personalizaci uživatelských zkušeností a přetváření správy platforem. Ačkoliv generativní AI snižuje bariéry pro tvůrce obsahu, zároveň zintenzivňuje konkurenci a vyvolává etické otázky týkající se autentičnosti, dezinformací a zaujatosti AI. Digitální platformy musí řešit právní a regulační výzvy, včetně problémů s autorskými právy a rizik deepfake technologií, přičemž AI může zároveň sloužit jako nástroj pro detekci

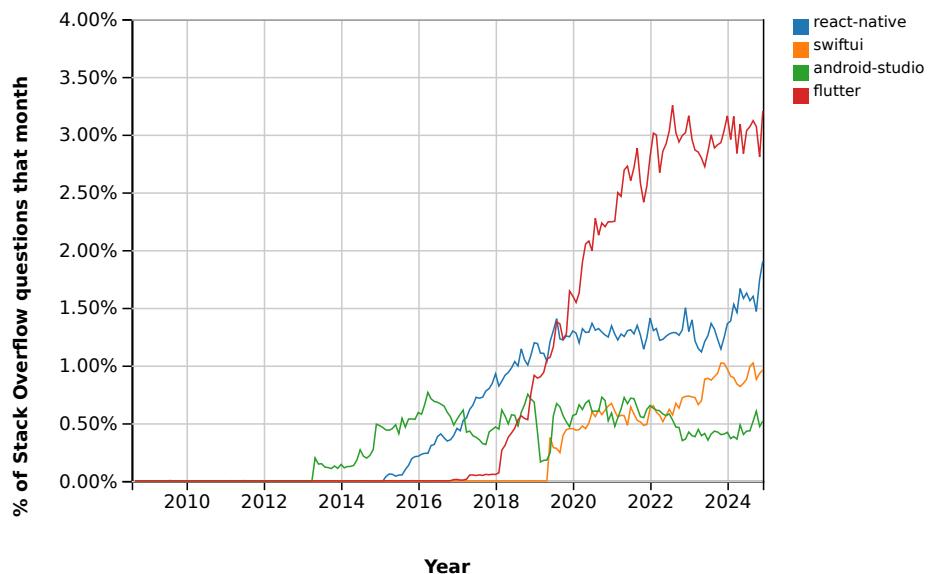
podvodů a řízení obsahu. Rychlý vývoj AI rovněž ohrožuje pracovní místa a prohlubuje digitální propast, což zdůrazňuje potřebu strategického řízení a odpovědných politik pro využití AI.

2.1.2 Nativní vs Multiplatformní vývoj

V současnosti je na App Store a Google Play dostupných více než 4,5 milionu aplikací, které cílí na více než 14 miliard mobilních telefonů po celém světě. Při rozhodování o vývoji mobilní aplikace je klíčové zvolit mezi nativním a multiplatformním přístupem, což ovlivňuje náklady, čas a funkčnost výsledné aplikace. [5]

Každý přístup má své výhody a nevýhody, těm se práce bude věnovat v další kapitole, nicméně z dostupných trendů portálu Stack Overflow můžeme porovnat nejvíce diskutované technologie na grafu 2.2.

Graf ukazuje výrazný nárůst zájmu o multiplatformní frameworky, zejména Flutter, který dominoje současným trendům. React Native také jako multiplatformní framework vykazuje silné postavení. Nativní technologie (Android Studio a SwiftUI) vykazují relativně stabilní úroveň s mírnými poklesy. Graf naznačuje rostoucí zájem vývojářů o multiplatformní vývoj, pravděpodobně kvůli opětovnému použití kódu a rychlejšímu vývoji, ale je důležité si uvědomit, že otázky na Stack Overflow jsou pouze jedním ukazatelem adopce technologií a mohou odrážet výzvy ve vývoji stejně jako popularitu.



Obrázek 2.2: Porovnání trendů nativních a multiplatformních technologií [6]

2.2 Poznámkové aplikace

V současné době hrají mobilní poznámkové aplikace zásadní roli v produktivitě uživatelů, přičemž integrace umělé inteligence (AI) výrazně zvyšuje jejich funkčnost. Mainstreamové aplikace začínají integrovat nové a rozšiřovat stávající funkcionality s podporou AI. Ať už se jedná o summarizaci, ukládání a transkript hlasových záznamů, chytré vyhledávání nebo analýzu dokumentů a textu poznámek.

2.2.1 Apple Notes

Apple Notes je bezplatná, předinstalovaná aplikace pro psaní poznámek na zařízeních Apple. Díky hluboké integraci s ekosystémem iOS a macOS je oblíbená pro svou jednoduchost, spolehlivost a postupné rozšiřování funkcí. Nabízí podporu pro text, obrázky, zvuk, skenování dokumentů, tabulky a další formáty, což z ní dělá všeestranný nástroj pro běžné i pokročilé uživatele.

Výhody

- **Zdarma a plně integrovaná do Apple ekosystému** – Poznámky se snadno synchronizují mezi iPhone, iPadem a Macem přes iCloud.
- **Postupné vylepšování** – Apple Notes byla původně velmi základní aplikací, ale dnes obsahuje funkce jako štítky, sdílení poznámek, pokročilé vyhledávání a podporu pro ručně psané poznámky.
- **Bezpečnost a soukromí** – Poznámky lze uzamknout heslem nebo biometrickým ověřením (Face ID/Touch ID).
- **Vylepšená práce s médií** – Možnost importovat obrázky, videa, skenovat dokumenty a přepisovat hlas na text.

Nevýhody

- **Omezená kompatibilita** – Aplikace je exkluzivně dostupná pouze pro Apple zařízení. Použití na Windows nebo Androidu je možné pouze přes iCloud.com, což není vždy ideální.
- **Absence pokročilé AI (do verze iOS 18)** – Apple Notes až donedávna postrádala jakékoli AI funkce. Až v iOS 18 přibyla možnost nahrávání a přepisu hlasových poznámek, podporovaný jazyk je pouze Angličtina. Další pokročilé AI funkce, jako generování textu nebo summarizace, zatím chybí. Navíc „Apple Intelligence“ zatím není dostupná v EU.

Apple Notes je ideální pro uživatele, kteří hledají jednoduchou, bezplatnou a spolehlivou aplikaci na poznámky s hlubokou integrací do Apple ekosystému. [3, 7, 8]

2.2.2 Microsoft OneNote

Microsoft OneNote je výkonná a všestranná aplikace na psaní poznámek, která se vyznačuje širokými možnostmi organizace a spolupráce. Umožňuje kombinovat text, rukopis, kresby, tabulky a multimedialní obsah, což z něj dělá ideální nástroj pro studenty, profesionály i týmy.

Výhody

- **Flexibilní organizace poznámek** – Digitální notebook s oddíly, pododdíly a stránkami umožňuje hlubokou hierarchii uspořádání.
- **Podpora rukopisu a převod na text** – Uživatelé mohou psát perem a OneNote dokáže rukopis automaticky převést na strojový text.
- **Integrace s Microsoft 365** – propojení s Wordem, Excellem a Outlookem usnadňuje práci a sdílení informací.
- **Spolupráce v reálném čase** – Ideální pro týmovou práci, umožňuje společné úpravy a sdílení anotací.

Nevýhody

- **Závislost na Microsoft ekosystému** – Integrace je optimalizovaná hlavně pro Microsoft produkty, což může být nevýhoda pro uživatele jiných platform.
- **Občasné zpomalení a zamrzání** – Při práci s velkým objemem dat může aplikace lagovat nebo se na chvíli zaseknout.
- **Nutnost Microsoft 365 pro pokročilé funkce** – Bez předplatného jsou dostupné pouze základní funkce a omezené úložiště.

Microsoft OneNote je skvělá volba pro uživatele, kteří potřebují strukturovaný a výkonný nástroj na poznámky s hlubokou integrací do Microsoft Office. Aplikace je dostupná zdarma s omezeným úložištěm, přičemž plná verze je součástí předplatného Microsoft 365, které začíná na 69,99 USD ročně. OneNote je k dispozici na platformách Windows, macOS, Android, iOS a také ve webové verzi. I přes některé nedostatky, jako je občasné zpomalení nebo závislost na Microsoft ekosystému, zůstává jedním z nejlepších řešení pro organizaci poznámek a týmovou spolupráci. [9, 10]

2.2.3 Obsidian

Obsidian představuje moderní přístup k organizaci poznámek, který umožňuje vytvářet komplexní a vzájemně propojenou digitální databázi. Oproti tradičním poznámkovým aplikacím nabízí pokročilé možnosti propojení mezi jednotlivými poznámkami, čímž se stává ideálním nástrojem pro lidi, kteří chtějí budovat vlastní wiki.

Hlavním benefitem Obsidianu je jeho rozšiřitelnost a vysoká míra přizpůsobení. Uživatelé mohou organizovat poznámky do složek, využívat interní odkazy pomocí dvojitých hranatých závorek [] a vizualizovat propojení mezi jednotlivými záznamy prostřednictvím grafického zobrazení.

Výhody

- **Interaktivní propojení poznámek** – Umožňuje vytvářet propojenou síť informací a budovat vlastní wiki.
- **Rozšiřitelný** – Možnosti úpravy uživatelského rozhraní a funkcionalit díky komunitním pluginům.

Nevýhody

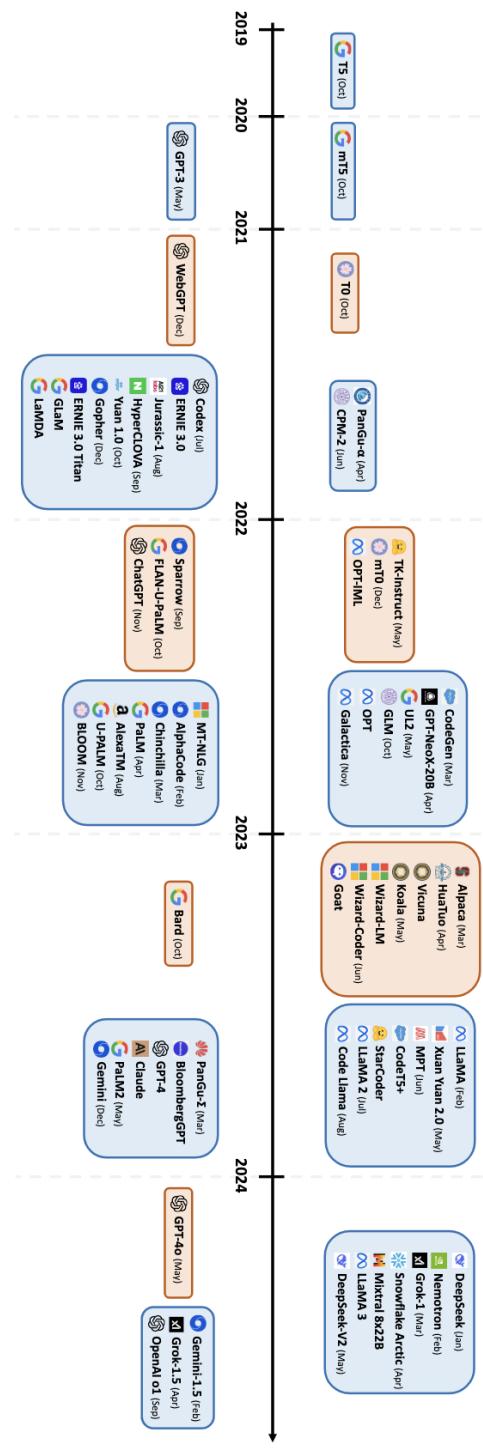
- **Strmá učební křivka** – Vyžaduje čas na pochopení konceptu propojených poznámek a efektivní využití všech funkcí.

2.3 Velké jazykové modely (LLMs)

Studie [11] poskytuje komplexní přehled rychle se vyvíjející oblasti velkých jazykových modelů (LLM) a zdůrazňuje jejich revoluční dopad na zpracování přirozeného jazyka (NLP). S rostoucí poptávkou po strojích schopných zvládat úlohy jako překlad, sumarizace či konverzace se LLM stávají klíčovou technologií, přičemž jejich vývoj je poháněn pokroky v transformerových architekturách, zvýšenými výpočetními kapacitami a rozsáhlými trénovacími daty. Studie se zabývá historickým vývojem NLP, přechodem od statistických modelů k neuronovým sítím a předtrénovaným jazykovým modelům (PLM), které umožnily efektivnější doladění a rozšíření na LLM. Autoři zdůrazňují rostoucí trend směrem k modelům laděným na instrukce a open-source variantám, a nabízejí systematické shrnutí architektury, trénování, multimodálních přístupů, datasetů, metod hodnocení a výzev v oblasti LLM. Studie tak poskytuje ucelený pohled na současný stav výzkumu a vývoje LLM a slouží jako cenný referenční zdroj pro odbornou komunitu.

2.4 Převod řeči na text (Speech to Text - STT)

Studie [12] poskytuje komplexní přehled o přímém převodu řeči na text, přičemž se zaměřuje na tři klíčové výzvy: výpočetní náročnost modelů, nedostatek dat a problémy s aplikací v praxi. Diskutovány jsou pokroky v řešení složitosti přímého překladu řeči, včetně modelů založených na Transformer architektuře, učení se na více úlohách a neautoregresivních přístupech. K překonání nedostatku dat se využívají metody jako augmentace dat, před trénováním modelů, destilace znalostí a vícejazyčné učení. Studie se dále věnuje praktickým problémům, jako je překlad v reálném čase, segmentace textu, rozpoznávání aktérů, rozpoznání pohlaví a jazyková variace. Do budoucna se



Obrázek 2.3: Chronologický přehled vydání Velkých jazykových modelů [11]

jako perspektivní směry výzkumu jeví integrace velkých jazykových modelů (LLMs) a zkoumání multimodálního překladu, například převodu řeči na řeč nebo překladu v rámci video obsahu.

Studie [13] se zaměřuje na WhisperX, systém pro přesnou transkripci dlouhých audiozáznamů s časovými značkami na úrovni jednotlivých slov. WhisperX vychází z modelu Whisper, což je rozsáhlý slabě supervizovaný model pro rozpoznávání řeči, avšak překonává jeho omezení v oblasti práce s dlouhými nahrávkami a nepřesnými časovými značkami. Mezi hlavní přínosy systému patří zlepšená časová přesnost díky využití Voice Activity Detection (VAD) pro efektivní segmentaci audia a vynucenému zarovnání fonémů, které zajišťuje přesné časové značky na úrovni slov. Dále umožňuje lepší transkripci dlouhých nahrávek díky metodě VAD Cut & Merge, která inteligentně segmentuje zvuk a zachovává přesnost a kontext, a dávkovému zpracování, jež zrychluje transkripcí až $12 \times$ oproti původnímu modelu Whisper. Kvalita transkripce a segmentace je vyšší, protože systém minimalizuje chyby, jako jsou halucinace (vkládání neexistujících slov) a opakování, a dosahuje větší přesnosti než původní Whisper a jiné ASR modely. Navíc podporuje vícejazyčnou transkripci a překlad, přepisuje více jazyků a umožňuje jejich překlad do angličtiny pomocí fonetického rozpoznávání pro lepší zarovnání. WhisperX tak nachází uplatnění v přepisu podcastů, schůzek, titulkování videí a diarizaci mluvčích, přičemž výrazně zlepšuje rychlosť a přesnost automatického rozpoznávání řeči (ASR).

Studie [14] představuje Speech-LLaMA, inovativní dekodérový přístup, který umožňuje přímou integraci řečových signálů do velkých jazykových modelů (LLMs) bez nutnosti mezikroku převodu řeči na text. Využitím CTC (Konekcionistická časová klasifikace) komprese a lehkého zvukového enkodéru model efektivně mapuje akustické vlastnosti do sémantického prostoru LLM, čímž významně zlepšuje multilingvální převod řeči na text. Testování na 13 jazyčích ukazuje, že Speech-LLaMA překonává tradiční enkodér-dekodér architektury a zároveň je o 40 % úspornější z hlediska parameterů. Studie dále zkoumá optimalizační techniky, jako je LoRA fine-tuning a ne-kauzální maskování pozornosti, které zvyšují výkon s minimálnimi výpočetními nároky. Tyto výsledky naznačují, že dekodérové modely představují efektivní a perspektivní alternativu pro zpracování řeči a umělou inteligenci v interakci člověka s počítačem.

Studie [15] ukazuje, jak modely pro převod řeči na text, konkrétně Whisper, mohou s vysokou přesností predikovat neurální aktivitu během zpracování řeči v reálném čase. Využitím elektrokortikografických (ECOG) záznamů ze 100 hodin spontánních konverzací autoři mapují akustické a jazykové prvky na mozkovou aktivitu, čímž odhalují hierarchickou organizaci zpracování řeči v mozku. Výzkum ukazuje, že senzorické a motorické oblasti kódují zvuky řeči, zatímco vyšší kortikální oblasti se podílejí na zpracování významu a syntaxe. Klíčovým objevem je časový průběh jazykového zpracování, kde mozek předvírá jazyk před samotným mluvením a extrahuje význam po vyslechnutí slov, což naznačuje dynamický prediktivní mechanismus. Tyto poznatky propojují neuronného a umělou inteligenci, přičemž ukazují potenciál hlubokého učení pro dekódování mozkových funkcí, vylepšení rozhraní mozek-počítač a rozvoj technologií rozpoznávání řeči pro neurologické poruchy.

Studie [16] zkoumá využití technologie převodu řeči na text (STT) středoškolskými studenty a ukazuje, že nejčastějšími uživateli byli studenti s poruchami učení (LDs), zatímco vícejazyční studenti a žáci běžného vzdělávání nevykazovali výraznou preferenci. STT pomohlo studentům psát delší eseje, snížit kognitivní zátěž a urychlit proces tvorby textu, avšak téměř polovina studentů se této technologie vyhýbala kvůli pocitu trapnosti, technickým problémům nebo obtížím při revizi mluveného textu. Učitelé vnímali STT pozitivně, zejména v kontextu vzájemného hodnocení textů a propojení běžné řeči s akademickým psaním, přičemž klíčem k úspěchu bylo vytvoření podpůrného prostředí. Výsledky naznačují, že STT představuje slibný, ale nedostatečně využívaný nástroj, který by mohl zlepšit plynulost psaní, pokud by se podařilo překonat sociální a technické bariéry.

2.5 Sumarizace textu

Studie [17] se zaměřuje na techniky summarizace textu založené na zpracování přirozeného jazyka (NLP), přičemž rozlišuje extraktivní a abstraktivní metody, které umožňují převod rozsáhlých dat do stručných a smysluplných shrnutí. Studie analyzuje nesupervizované, supervizované, metody posilovaného učení (reinforcement learning) a hybridní přístupy, hodnotí jejich silné a slabé stránky a popisuje praktické aplikace těchto metod. Mezi klíčové výzvy patří zachování sémantické soudržnosti, koreference a potřeba kvalitních trénovacích dat. Výzkum zdůrazňuje transformerové architektury (např. BERT, GPT) a posilované učení jako slibné směry vedoucí k přesnější a přirozenější summarizaci textu. Budoucí výzkum se zaměří na multilingvální summarizaci, specializované domény a zvýšení faktické přesnosti generovaných shrnutí, čímž přispěje k dalšímu rozvoji automatizovaného zpracování textu.

Studie [18] představuje T2SAM, model pro abstraktivní summarizaci textu založený na Transformeru, který využívá mechanismus self-attention k řešení problémů s koreferencí, čímž zlepšuje soudržnost a přesnost generovaných shrnutí. Model byl trénován na Inshorts News a DUC-2004 a dosáhl 48,50 % přesnosti v metrice ROUGE-1, čímž překonal stávající modely jako PEGASUS a BART, zároveň vyzkoušel výrazně nižší tréninkovou ztrátu. T2SAM efektivně generuje stručná a smysluplná shrnutí, řeší klíčové problémy související s odkazováním na zájmena a zároveň zachovává sémantickou integritu textu. Díky své schopnosti přizpůsobit se různým jazykům a specializovaným doménám tato studie zdůrazňuje význam self-attention mechanismů pro automatizovanou sumarizaci textů a otevírá nové možnosti pro jejich využití v širším spektru aplikací.

Studie [19] zkoumá, jak ChatGPT a Gemini zvládají summarizaci literárních textů, a odhaluje, že ChatGPT často generuje tzv. halucinace, což vede k nesprávným interpretacím postav, chybným přiřazením událostí a nadměrným zjednodušením. Analýza úryvků z románu The Solid Mandala od Patricka Whitea identifikovala 12 typů halucinací, přičemž ChatGPT měl problémy s faktickou přesností a soudržností vyprávění, často kvůli omezené velikosti slovní zásoby a heuristickému řešení koreferencí. Gemini se ukázal jako výkonnější, nebot lépe minimalizoval chyby v koreferenci a dokázal efektivněji pracovat s komplexními narativními strukturami. Studie zdůrazňuje potřebu

vylepšených AI modelů, které budou schopny věrné a kontextově přesné summarizace v literárním prostředí, aniž by docházelo ke zkreslování původního obsahu.

2.6 Shrnutí a motivace

Studie ukazují obrovský pokrok v oblastech umělé inteligence a možnostech využití. Jak bylo zmíněno ve studii [16], pro ulehčení a zrychlení některých aktivit se integrace AI do moderních nástrojů velice vyplatí, a do budoucna můžeme očekávat růst tímto směrem. Přestože existující poznámkové aplikace, jako Apple Notes, Microsoft OneNote či Obsidian, které již nabízejí určitou úroveň AI funkcionalit, často je jejich využití nepřehledné, moc komplikované, nebo naopak některé klíčové funkce zcela chybí. Zároveň se dosavadní výzkum nezabývá možnostmi integrací AI do mobilních aplikací, zejména v kontextu technologie React Native, což představuje zajímavou výzvu. Tato práce se proto zaměřuje na návrh a vývoj SmartNotes pro iOS, kde bude kladen důraz na jednoduchost ovládání, ale zároveň na bohatou integraci AI funkcionalit, včetně automatického přepisu řeči na text, inteligentního vyhledávání a analýzy obsahu. Cílem je vytvořit intuitivní poznámkovou aplikaci, která přehledně a efektivně propojí dostupné AI technologie a nabídne uživatelům intuitivní a výkonný nástroj pro správu poznámek optimalizovaný pro mobilní prostředí.

Kapitola 3

Teoretická část

Tato kapitola poskytuje teoretický rámec pro návrh a implementaci aplikace SmartNotes. Zaměřuje se na klíčové technologie, jako je React Native a framework Expo, jejichž kombinace umožňuje efektivní vývoj multiplatformních mobilních aplikací. Dále popisuje strategie správy dat, přístupy k integraci velkých jazykových modelů a možnosti jejich lokálního i cloudového nasazení.

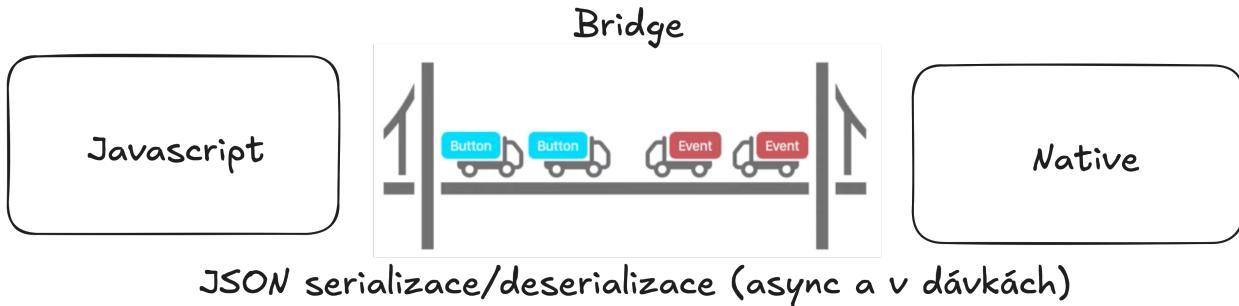
Součástí kapitoly je také přehled nástrojů pro práci s LLM, jako je LangChain a jeho alternativy, a přístup k monitorování výkonu a chování mobilních aplikací. Teoretické poznatky zde uvedené tvoří základ pro praktické rozhodování při návrhu a vývoji aplikace.

3.1 React Native

React Native je multiplatformní framework pro vývoj mobilních a webových aplikací, který umožňuje využívat JavaScript a React k vytváření nativních aplikací pro iOS a Android. Na rozdíl od přístupu založeného na WebViews poskytuje React Native přístup k nativním komponentám operačního systému, což zajišťuje vyšší výkon a lepší uživatelský zážitek.

3.1.1 Runtime a bridge

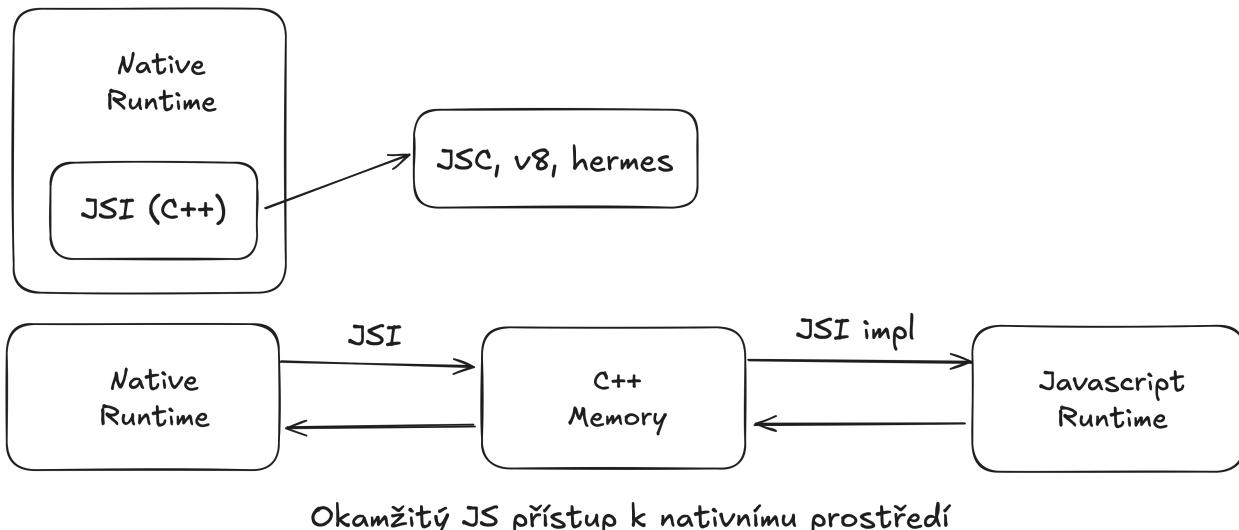
V tradiční architektuře React Native funguje JavaScriptový kód v samostatném vlákně, přičemž interaguje s nativními moduly prostřednictvím tzv. "Bridge"- mechanismu, který zprostředkovává komunikaci mezi JavaScriptovým a nativním prostředím/vláknem (např. Java/Kotlin pro Android a Objective-C/Swift pro iOS). Tento přístup však přináší určitou latenci, protože data mezi oběma prostředími jsou serializovaná a deserializovaná při každém volání a zároveň se jedná o asynchronní komunikaci.



Obrázek 3.1: Komunikace pomocí JS bridge

3.1.2 Nová bridgless architektura: JSI (JavaScript Interface)

Nová architektura React Native postupně přechází na JSI (JavaScript Interface), který odstraňuje původní Bridge a umožňuje přímé volání nativních funkcí z JavaScriptu s minimální režií. Místo serializace dat mezi prostředími nyní JavaScript může volat nativní C++ metody přímo, což výrazně zrychluje komunikaci a zlepšuje výkon aplikací. Nové změny umožňují synchronní a asynchronní komunikaci mezi JavaScript vláknem a Native vláknem, v minulosti byla komunikace kvůli architektuře Bridge pouze asynchronní a při zahlcení nemuselo nativní vlákno vědět, co v daný moment renderovat, a docházelo k nepříjemnému UX pro uživatele.



Obrázek 3.2: Ukázka komunikace v JSI architektuře

3.2 Expo

Expo je open-source framework a platforma pro vývoj React Native aplikací, která je v současné době standardně doporučována oficiální React Native dokumentací pro vývoj nových aplikací, usnadňuje vývoj mobilních aplikací pomocí předkonfigurovaných nástrojů, knihoven a služeb. Díky abstrakci složitostí nativního kódu umožňuje Expo vývojářům rychleji vytvářet, testovat a nasazovat aplikace bez nutnosti rozsáhlých úprav pro jednotlivé platformy. [20, 21]

Umožňuje okamžitý náhled aplikace na fyzických zařízeních bez nutnosti rekomplikace díky sandboxové aplikaci Expo Go. Expo také poskytuje sadu předpřipravených knihoven (SDKs) pro běžné nativní funkce, jako je přístup ke kameře, push notifikace, geolokace a další desítky nativních modulů, čímž snižuje závislost na nativních modulech třetích stran. [20]

Expo poskytuje EAS (Expo Application Services), což je sada cloudových služeb navržených týmem Expo za účelem zjednodušení a zefektivnění vývoje, sestavování a distribuce aplikací vytvorených pomocí Expo a React Native. EAS umožňuje vývojářům snadno sestavovat aplikace, publikovat aktualizace a odesílat hotové binární soubory do obchodů s aplikacemi, a to bez nutnosti složité konfigurace. [22]

Mezi klíčové komponenty této služby patří:

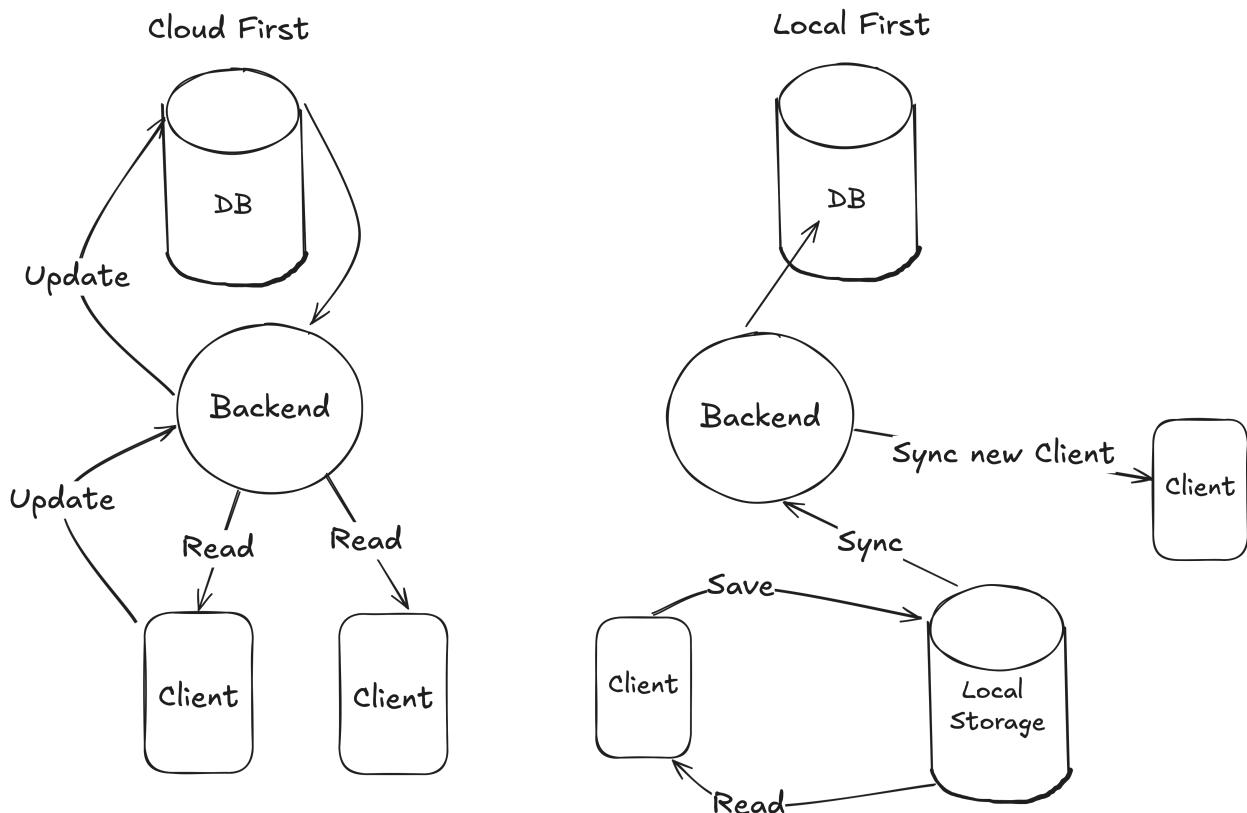
- **EAS Build** - EAS Build je cloudová služba poskytovaná společností Expo, která umožňuje vývojářům Expo a React Native projektů snadno vytvářet binární soubory aplikací pro distribuci. Tato služba zjednoduší proces sestavení aplikací pro distribuci tím, že poskytuje výchozí nastavení vhodná pro Expo a React Native projekty a umožňuje automatizaci procesu sestavení. [23]
- **EAS Update** - EAS Update je cloudová služba poskytovaná společností Expo, která umožňuje vývojářům aplikací rychle a efektivně doručovat aktualizace uživatelům bez nutnosti procházet schvalovacím procesem v obchodech s aplikacemi. Tato služba se integruje s knihovnou expo-updates a umožňuje aplikacím aktualizovat své nenativní části, jako je JavaScriptový kód, styly a obrázky, přímo "over-the-air"(OTA). [24]
- **EAS Workflows** - EAS Workflows zjednoduší a automatizují celý proces doručování aplikací v rámci React Native projektů. Umožňují vývojářům definovat sekvence úloh pro sestavení, testování, odeslání do obchodů s aplikacemi a spouštění vlastních skriptů. Díky podpoře konfigurací a závislostí mezi úlohami lze vytvořit efektivní a flexibilní CI/CD pipeline přizpůsobenou potřebám týmu. [25]

Na rozdíl od univerzálních nástrojů pro kontinuální integraci, jako jsou CircleCI nebo GitHub Actions, jsou EAS Workflows navrženy speciálně pro projekty postavené na frameworku Expo. Nabízejí předpřipravené typy úloh a optimalizované prostředí běžící v cloudu. Konfigurace se

provádí pomocí YAML souborů, přičemž všechny výstupy sestavení a aktualizací jsou dostupné přímo ve webové aplikaci Expo. [25]

3.3 Ukládání a správa dat

Při ukládání a správě uživatelských dat se v současné době využívá několik strategií, které se liší podle potřeb aplikace, požadavků na dostupnost, bezpečnost a rychlosť přístupu k datům. Mezi nejčastěji používané přístupy patří strategie Cloud-first a Local-first, které představují dva odlišné modely správy dat s vlastními výhodami a omezeními.



Obrázek 3.3: Schéma přístupu ke správě a ukládání dat

3.3.1 Cloud-first

Strategie Cloud-first se zaměřuje na primární ukládání dat v cloudových úložištích, což umožňuje snadnou synchronizaci mezi zařízeními, vzdálený přístup a škálovatelnost. Tento model využívají například Google Drive, iCloud nebo Firebase, které poskytují automatickou zálohu a ochranu proti ztrátě dat. Výhodou je snadná dostupnost dat odkudkoli, nevýhodou pak závislost na internetovém připojení a potenciální bezpečnostní rizika.

3.3.2 Local-first

Strategie Local-first klade důraz na lokální ukládání dat na zařízení uživatele, což zajišťuje rychlou odezvu aplikace, offline přístup a větší kontrolu nad daty. Tento přístup využívají aplikace jako Obsidian nebo Apple Notes, které umožňují offline práci a minimalizují riziko úniku citlivých informací do cloudu. Hlavním omezením je potřeba synchronizačního mechanismu pro více zařízení a riziko ztráty dat při poškození nebo ztrátě zařízení, pokud není implementována vhodná zálohovací strategie.

3.4 Integrace velkých jazykových modelů

Velké jazykové modely se staly klíčovým prvkem moderních aplikací, které pracují s přirozeným jazykem. Jejich schopnost analyzovat, sumarizovat a vyhledávat informace v rozsáhlých datech zásadně rozšiřuje možnosti uživatelských rozhraní i automatizace. Tato sekce se věnuje technickým přístupům k integraci LLM do mobilních aplikací, porovnává cloudové a lokální varianty nasazení a představuje nástroje jako LangChain, které zjednodušují práci s těmito modely v praxi.

3.4.1 Velké jazykové modely a jejich přínos

Velké jazykové modely (Large Language Models) představují pokročilé hluboké neuronové sítě, které byly natrénovány na rozsáhlých textových datech. Jejich architektura je založena na principu transformeru, který využívá mechanismus sebe-pozornosti (self-attention) k analýze a porozumění jazykovým strukturám. Tato architektura typicky zahrnuje enkodér a dekodér, které umožňují modelu chápout význam jednotlivých slov a jejich vzájemné vztahy v rámci vět i rozsáhlejších textových celků. [26]

LLM jsou schopny s vysokou přesností vykonávat většinu úloh z oblasti zpracování přirozeného jazyka (NLP). Tyto úlohy mohou být přizpůsobeny specifickým doménám, jako je zdravotnictví, marketing, správa znalostních databází, tvorba interní dokumentace či generování různých typů obsahu. [27]

Nasazení LLM do moderních aplikací významně přispívá k jejich chytrosti a užitečnosti. Díky schopnosti sledovat uživatelské chování a adaptovat se na individuální potřeby uživatele mohou tyto modely poskytovat personalizovaný obsah a zlepšovat celkový uživatelský zážitek. Například společnosti jako Amazon využívají velké jazykové modely k doporučování produktů, a to na základě analýzy historie nákupů, vyhledávání a dalších relevantních parametrů.

Podobně i fitness aplikace staví na těchto technologiích při tvorbě personalizovaných tréninkových plánů. Na základě uživatelských dat, stanovených cílů a výkonnostních ukazatelů dokáží generovat vhodné výzvy, které udržují uživatele motivované a napomáhají k dosažení konkrétních výsledků.

3.4.2 Cloudové vs. lokální modely

Nasazení velkých jazykových modelů (LLM) do reálných aplikací probíhá nejčastěji formou cloudové integrace, kdy se vstupní data odesílají na vzdálený server, kde probíhá zpracování modelem, a následně se výsledek vrací zpět do aplikace. Tento přístup je v současnosti dominantní, a to především z důvodu vysokých nároků LLM na výpočetní výkon, paměť i energetickou spotřebu, které běžná uživatelská zařízení zatím nejsou schopna dlouhodobě zvládnout. [28]

Cloudová architektura přináší několik nevýhod. Patří mezi ně vyšší latence, potřeba neustálého připojení k internetu a také otázky soukromí. Tyto faktory motivují současný posun směrem k lokálním (on-device) modelům, tedy nasazení modelu přímo na zařízení uživatele. [28]

Lokální modely nabízejí výrazně nižší latenci, bezkonkurenční ochranu soukromí, možnost využití modelu offline a výpočetní povinnost přenesenou na klienta, což snižuje náklady na infrastrukturu aplikace. [28]

Současně však lokální modely naráží na technická omezení. Běžně využívané cloudové LLM využívají obrovské množství parametrů, to vyžaduje vysoké množství operační paměti i výpočetního výkonu. Například model se 7B parametry spotřebuje přibližně 0,7 joulu na jeden vygenerovaný token, což znamená, že běžné zařízení jako iPhone s kapacitou baterie kolem 50 kJ by zvládlo interakci s takovým modelem pouze po dobu dvou hodin. K tomu je třeba přičíst i zvýšené zahřívání zařízení a možné problémy s výdrží a stabilitou systému. [28]

Příklady některých současných lokálních modelů:

- **Gemini Nano** - Mobilní operační systém bude v budoucnu poskytovat jazykový model a jeho inferenční infrastrukturu jako systémovou službu, podobně jako např. lokalizační nebo notifikační rozhraní. Uživatelé a vývojáři tak budou moci přistupovat k tzv. AI jádru (AI Core) prostřednictvím Google AI Edge SDK. Uvnitř této infrastruktury je nasazen model Gemini Nano, což je zmenšená verze větších cloudových modelů Gemini. Oproti nim nabízí nižší latenci a rychlejší odezvu díky běhu přímo na zařízení. AI Core zároveň zajišťuje distribuci modelu a správu paměti, což umožňuje efektivní využití zdrojů mobilního zařízení. Model Gemini Nano je navržen tak, aby co nejlépe využíval hardwarovou akceleraci. Trénován byl pomocí techniky destilace z větších verzí modelu Gemini a pro účely nasazení byl kvantizován na 4 bity, což umožňuje výrazné zmenšení velikosti bez výrazné ztráty přesnosti. I přes svou kompaktnost tak model nabízí špičkový výkon na mobilních zařízeních. [28]
- **Apple OpenELM a Ferret-v2** - Společnost Apple vyvinula vlastní velký jazykový model s názvem OpenELM, který je integrován přímo do systému iOS jako součást základních systémových služeb – podobně jako například sledování polohy. Model využívá tzv. vrstveně škálovatelnou architekturu, která umožňuje efektivní nasazení s celkovým počtem 1,1 miliardy parametrů. I přes relativně malou velikost dosahuje model o 2,36 % vyšší přesnosti než jeho

předchůdci, a to při využití pouze poloviny trénovacích dat. OpenELM je navržen tak, aby byl kompatibilní s frameworkm MLX, což umožňuje jeho přímé ladění a trénování přímo na zařízeních Apple. Paralelně s tím Apple představil i multimodální model Ferret-v2, který oproti předchozí verzi přináší řadu vylepšení, včetně podpory uzemnění napříč rozlišenimi, víceúrovňového vizuálního kódování (díky integraci encoderu DINOv2) a třífázového trénovacího režimu. Tato vylepšení výrazně zvyšují kvalitu zpracování obrazových dat a posilují schopnosti modelu v oblasti vizuálního porozumění. Oba modely jsou navrženy s důrazem na lokální nasazení na zařízení (on-device) a ukazují směr, kterým se ubírá vývoj umělé inteligence v rámci mobilních ekosystémů Apple. Díky tomu je možné poskytovat pokročilé AI funkce přímo v systému bez nutnosti připojení k internetu. [28]

Rok	Název modelu	Velikost modelu	Lokální	Cloud
2023	Google Gemini Nano	7B	✓	
2023	OPPO AndesGPT	7B	✓	✓
2024	Honor MagicLM	7B	✓	
2024	VIVO BlueLM	7B	✓	✓
2024	XiaoMi MiLM	6B	✓	
2024	Apple OpenELM	1.1B	✓	✓

Tabulka 3.1: Lokální (on-device) LLM modely vydané výrobci mobilních telefonů [28]

3.4.3 LangChain

V rámci návrhu a implementace aplikace SmartNotes byla jako klíčová komponenta pro práci s velkými jazykovými modely zvolena knihovna LangChain. Tento open-source framework poskytuje nástroje pro efektivní integraci LLM do aplikací, které využívají přirozený jazyk, a zároveň výrazně zjednodušuje jejich orchestraci a správu. LangChain je navržen modulárně a umožňuje snadné skládání komplexních řetězců, což z něj činí vhodné řešení pro moderní intelligentní aplikace, jako je SmartNotes. [29]

Základní koncepty LangChain:

- **Prompt** - Představuje vstup, který se předává velkému jazykovému modelu. V aplikacích s LLM bývá prompt zpravidla generován dynamicky a obsahuje uživatelský vstup, například otázku nebo ukázky předchozích příkladů, které mají modelu pomoci lépe pochopit požadovaný výstup, a také instrukce, jak má model daný vstup zpracovat. [30]
- **Řetězce (Chains)** - Umožňují definovat sekvenci kroků, které zpracovávají vstupní dotaz nebo data. Například nejprve vyhledání v databázi, poté sumarizace výsledků pomocí LLM. Řetězce mohou být jednoduché (např. vstup → LLM → výstup), ale i komplexní, obsahující více kroků včetně podmínek, smyček nebo paralelního zpracování. [30]

- **Agenti** - Představují autonomní komponenty, které umí na základě zadání rozhodnout, jaký nástroj nebo akci použijí. Agent může například rozhodnout, zda odpověď vytvoří přímo LLM, nebo zda je potřeba nejdříve získat doplňující informace z externího zdroje. To umožňuje větší flexibilitu a „inteligenci“ v chování systému. [30]
- **Paměť** - LangChain umožňuje uchovávat konverzační kontext mezi jednotlivými dotazy, čímž podporuje plynulou interakci s uživatelem. Paměť může být krátkodobá, nebo dlouhodobá, vázaná na konkrétní uživatele nebo sezení. [30]

3.4.4 Alternativy: LlamaIndex, Haystack

Kategorie	LangChain	LlamaIndex	Haystack
Flexibilita a přizpůsobení	Nabízí nejvyšší míru flexibility a přizpůsobení, ideální pro specifická NLP řešení	Specializovaný na indexování a dotazování dat, s menším důrazem na přizpůsobení	Poskytuje modulární a komplexní framework vhodný pro škálovatelná NLP řešení
Snadnost použití	Uživatelsky přívětivý pro tvorbu vlastních LLM aplikací, ale může vyžadovat více nastavení	Snadné použití pro indexování a dotazování textových dat	Komplexní, ale zároveň nejsložitější na začátek
Výkon a škálovatelnost	Výkon závisí na konkrétním LLM a návrhu aplikace	Optimalizováno pro rychlé dotazování na velkých textových datasetech	Navrženo pro škálovatelnost a vysoký výkon v produkčních prostředích

Tabulka 3.2: Srovnání frameworků pro práci s LLM: LangChain, LlamaIndex a Haystack [31]

Při zvažování alternativ k frameworku LangChain pro vývoj aplikací využívajících velké jazykové modely (LLMs) je vhodné zmínit dvě významné platformy: LlamaIndex a Haystack. [32, 33]

LlamaIndex (dříve známý jako GPT Index) je nástroj specializovaný na efektivní správu, indexaci a dotazování nad rozsáhlými textovými datovými sadami. Poskytuje robustní možnosti datové indexace a nabízí snadnou integraci s různými textovými zdroji a LLM modely. Díky tomu je vhodný zejména pro projekty zaměřené na správu znalostí a systémy pro vyhledávání informací, kde je klíčová flexibilita při práci s daty. [31, 34]

Naopak Haystack představuje komplexní E2E framework navržený pro tvorbu škálovatelných NLP systémů připravených pro nasazení do produkčního prostředí. Díky modulární architektuře umožňuje přizpůsobit jednotlivé komponenty, jako jsou retrievery a readery, podporuje různá dokumentová úložiště pro správu velkých datových sad a nabízí nástroje pro sestavování komplexních NLP pipeline. Haystack je vhodnou volbou pro aplikace typu sémantického vyhledávání nebo data-

zovacích systémů, zejména v podnikových prostředích, kde jsou vysoké nároky na škálovatelnost a výkon. [31, 34]

3.5 Monitoring mobilních aplikací

Monitorování aplikace je klíčovou zodpovědností vývojového týmu aplikace. Slouží k porozumění tomu, jak uživatelé aplikaci používají, a jakým způsobem je aplikace výkonná a náchylná na chyby. Díky tomu dokážeme odhalit technické problémy, problémy s interakcí, a následně optimalizovat a vylepšovat UX nebo výkonnost. Je třeba sbírat data a nastavit metriky, které nám jasně odhalí aktuální stav aplikace v produkčním prostředí nebo při otevřeném testování uživateli.

Kvůli komplexitě mobilních aplikací a množství zařízení, na které lze aplikace distribuovat, může velice snadno vzniknout problém, který může vzbudit frustraci uživatelů. Aby aplikace zůstala konkurenceschopná, organizace musí investovat do řešení, které jim pomůže s ochranou výkonnosti aplikace a rychlou detekcí problémů, aby bylo možné je co nejrychleji opravit a neohrozilo to tak konečné UX uživatelů. [35]

3.5.1 Výkonnost mobilní aplikace a metriky

Výkonnost aplikace definují měření, jako je doba načítání aplikace, jak náročná na hardware aplikace je, kolik úložného prostoru využívá a jak často aplikace končí fatální chybou. Tyto měření můžeme zadefinovat pomocí metrik: [36]

- **Akvizice uživatelů** - Počet nových uživatelů, což může pomoci se správným promováním aplikace.
- **Interakce uživatelů** - Metriky jako čas strávený v aplikaci, počet relací, retence uživatelů.
- **Kritické chyby aplikace** - Identifikace chyb, které mohou mít dopad na uživatele a jejich rychlá náprava.
- **Demografie uživatelů** - Nainstalovaná verze aplikace, operační systém telefonu, lokace, typ zařízení.

3.5.2 Doporučené praktiky

Následující seznam popíše doporučené praktiky při monitorování a optimalizaci výkonu aplikace. [36]

- **Prioritizace UX** - Úspěch aplikace záleží na uživatelském zážitku. Prioritizování UX optimalizováním výkonnosti aplikace a redukcí latencí načítání obrazovek povede k nárůstu, větší spokojenosti a k většímu zapojení ze strany uživatelů.

- **Minimalizace síťových požadavků** - Povede k redukci latence obrazovek a ke zlepšení načítání dat. Toho lze dosáhnout cachováním požadavků, kompresí dat a optimalizací multi-mediálních formátů.
- **Optimalizace výkonu baterie** - Je důležité myslet na spotřebu baterie. Aplikace, která bude extrémně výkonná, ale povede k rychlému vybití baterie, bude negativně hodnocena uživateli.
- **Neustálý monitoring** - Pomůže lépe identifikovat problémy a včas je řešit.

Kapitola 4

Praktická část

Praktická část práce se zaměřuje na návrh, vývoj a ověření funkčnosti mobilní aplikace SmartNotes, která je určena pro efektivní správu poznámek, dokumentů a hlasových záznamů. Klíčovým aspektem návrhu je využití technologií umělé inteligence, především pro automatický přepis mluveného slova, generování summarizací a sémantické vyhledávání obsahu. Důraz je kladen na vysokou použitelnost aplikace v reálném pracovním prostředí, ale i její adaptabilitu pro individuální potřeby uživatelů.

Úvodní část této kapitoly je věnována analýze uživatelských požadavků, která vychází z praktických scénářů použití a slouží jako východisko pro návrh klíčových funkcionalit. Následuje popis architektury systému, rozdělené do jednotlivých vrstev, a detailní přehled implementace hlavních modulů aplikace. Součástí je také část věnovaná bezpečnosti a ochraně dat, která reflektuje specifika práce s potenciálně citlivým obsahem a integraci služeb třetích stran poskytujících AI funkcionality.

Závěr kapitoly je věnován metodice testování a zhodnocení aplikace v praxi. Cílem je nejen ověřit technickou správnost řešení, ale také posoudit jeho přínos z pohledu koncového uživatele, a to s ohledem na výkon, uživatelskou přívětivost a celkovou spolehlivost systému.

4.1 Analýza uživatelských požadavků

Efektivní správa informací je v dnešní digitální době klíčová pro širokou škálu uživatelů, od profesionálů až po studenty. Cílem této analýzy je porozumět potřebám uživatelů poznámkových aplikací, definovat klíčové požadavky a určit, jak může umělá inteligence (AI) přispět ke zvýšení efektivity těchto nástrojů. Požadavky následně definujeme pomocí User Stories.

4.1.1 Průzkum potřeb uživatelů poznámkových aplikací

Na základě průzkumu a reálných zkušeností lze identifikovat několik klíčových oblastí, ve kterých uživatelé pocitují nedostatky u současných poznámkových aplikací:

- **Pokročilé vyhledávání** – Standardní vyhledávací nástroje jsou omezené na klíčová slova nebo nedokáží vyhledávat v hlasových záznamech, což není efektivní. Inteligentní sémantické vyhledávání může výrazně zlepšit UX.
- **Integrace multimediálního obsahu** – Možnost kombinovat text, dokumenty a hlasové záznamy do jednoho prostředí je klíčová pro lepší správu informací.
- **Sumarizace obsahu** – AI může pomoci s automatickým shrnutím dlouhých textů, zápisů ze schůzek, dokumentů nebo hlasových záznamů.
- **Rychlá dostupnost informací** – Uživatelé potřebují snadno a rychle přistupovat k relevantním informacím bez nutnosti procházet velké množství poznámek ručně.

Jedním z konkrétních příkladů této potřeby je situace v právnické společnosti, kde pracovníci často čelí nutnosti zaznamenávat schůzky s klienty, přidávat k nim poznámky a ukládat související dokumenty. Vzhledem k velkému množství klientů a případů je klíčové mít nástroj umožňující efektivní správu těchto informací, včetně možnosti automatické summarizace obsahu a pokročilého vyhledávání. Nemusí se jednat jenom o právnický sektor, jakákoli nutnost uchovávat a rychle summarizovat větší množství textu nebo hlasových nahrávek je skvělým příkladem použití pro takovou aplikaci.

4.1.2 User Story 1: Záznam a správa poznámek

Popis: Uživatelé potřebují efektivní způsob, jak vytvářet, organizovat a spravovat své poznámky v různých formátech, včetně textu, hlasových záznamů a PDF příloh. Aplikace umožní automatické propojení souvisejících poznámek, což usnadní přehlednost a navigaci mezi informacemi.

Cíle: Jako uživatel SmartNotes:

chci mít možnost vytvářet a spravovat své poznámky v různých formátech, abych si mohl zaznamenat důležité informace a snadno mezi nimi přepínat.

Funkční požadavky:

- Možnost vytvářet poznámky ve formátu text, hlas a PDF přílohy.
- Jedna poznámka propojuje všechny druhy poznámek.
- Editace poznámek a jejich přehledná organizace podle data poslední úpravy.
- Export poznámek do PDF formátu pro snadné sdílení.

Nefunkční požadavky:

- Okamžitá odezva - Systém musí neustále s uživatelem interagovat.
- Uživatelská přívětivost – Rozhraní musí být intuitivní a umožnit jednoduchou navigaci mezi poznámkami.

Akceptační kritéria:

- Pokud uživatel vytvoří novou poznámku, pak se tato poznámka uloží do systému a zobrazí v seznamu.
- Pokud uživatel edituje poznámku, pak se poznámka přesune do sekce dle data poslední editace.
- Pokud uživatel sdílí poznámku, pak bude vygenerován PDF soubor a otevře se nativní dialog pro sdílení.

4.1.3 User Story 2: Hlasové poznámky a AI summarizace

Popis: Uživatelé mohou zaznamenávat hlasové poznámky, které budou automaticky přepisovány do textu. Aplikace nabídne možnost sledování přepisu při přehrávání hlasového záznamu. Kromě toho bude k dispozici automatická summarizace klíčových bodů poznámky, aby uživatelé nemuseli číst dlouhé přepisy.

Cíle: Jako uživatel SmartNotes:

chci mít možnost zaznamenat hlasovou poznámku a nechat ji automaticky přepsat a summarizovat, abych nemusel ručně procházet celý záznam a rychle získal hlavní myšlenky.

Funkční požadavky:

- Možnost nahrát hlasovou poznámku přímo v aplikaci.
- Automatický přepis hlasového záznamu pomocí AI.
- Možnost sledování přepisu během přehrávání záznamu.
- Automatická summarizace obsahu poznámky s možností aktualizace.

Akceptační kritéria:

- Pokud uživatel nahraje hlasovou poznámku, pak systém automaticky převede její obsah na text.
- Pokud uživatel zobrazí přepis poznámky, pak bude mít možnost jej sledovat při přehrávání záznamu.
- Pokud uživatel vyžádá summarizaci poznámky, pak se zobrazí její souhrn včetně data poslední aktualizace.

4.1.4 User Story 3: Pokročilé vyhledávání a rychlý přehled

Popis: Uživatelé potřebují možnost rychlého vyhledávání poznámek, včetně hlasových a textových záznamů, pomocí sémantického vyhledávání. Dále aplikace umožní generování rychlého přehledu (Quick Recap), který poskytne souhrn klíčových bodů za zvolené časové období.

Cíle: Jako uživatel SmartNotes:

chci mít možnost rychle vyhledávat v poznámkách podle kontextu a generovat rychlý přehled, abych snadno našel důležité informace bez nutnosti ručního procházení obsahu.

Funkční požadavky:

- Sémantické vyhledávání podle klíčových slov a významu textu.
- Možnost vyhledávání v poznámkách, dokumentech a hlasových záznamech.
- V hlasových poznámkách bude zvýrazněn časový úsek, kde se hledaný výraz nachází.
- Quick Recap umožní generovat souhrnný přehled pro sekci poznámek dle poslední editace (např. Dnes, 7 dní zpět).

Nefunkční požadavky:

- Zvýraznění výsledků – U hlasových poznámek musí být možné přehrát danou část, kde se hledaný výraz vyskytuje.
- Sémantické vyhledávání – AI musí chápout kontext dotazu a nabídnout relevantní výsledky i při použití synonym nebo jiného jazyku.

Akceptační kritéria:

- Pokud uživatel zadá hledaný výraz, pak systém zobrazí poznámky, hlasové zprávy nebo dokumenty odpovídající nejen přesně, ale i významově.
- Pokud uživatel vyhledává v hlasových záznamech, pak se mu zobrazí časová informace, kde se hledaný výraz nachází.
- Pokud uživatel vyžádá Quick Recap pro danou sekci poznámek, pak se mu zobrazí souhrn klíčových bodů a akčních položek.

4.2 Bezpečnost a ochrana dat

Aplikace SmartNotes může pracovat s citlivými daty uživatelů, a proto je nezbytné zajistit adekvátní úroveň zabezpečení a ochrany soukromí. Jelikož aplikace funguje jako cloud-first služba, veškerá uživatelská data jsou uložena na centrálním serveru. To vyžaduje implementaci bezpečnostních mechanismů, které zajišťují šifrovanou komunikaci mezi klientskou aplikací a serverem a ochranu před neoprávněným přístupem.

iOS platforma navíc klade přísné požadavky na bezpečnostní standardy. Pokud server nesplňuje požadavky TLS (Transport Layer Security) a neprokáže platný certifikát důvěryhodné certifikační autority, systém iOS automaticky blokuje jakoukoli komunikaci se serverem. To chrání uživatele před potenciálními útoky typu man-in-the-middle a zajišťuje, že data nejsou přenášena nešifrovanou formou.

Vzhledem k integraci umělé inteligence (AI) a služeb třetích stran, které zpracovávají uživatelská data, je nutné analyzovat bezpečnostní aspekty spojené s těmito službami a zajistit transparentnost v nakládání s citlivými informacemi.

Na příkladu společnosti OpenAI lze najít několik incidentů, při kterých došlo k ohrožení či úniku uživatelských dat. Některé z těchto bezpečnostních problémů nesouvisely přímo s využíváním samotného modelu ChatGPT, ale se zranitelnostmi v rámci celé platformy. Jak uvádí článek [37], v jednom z případů došlo ke zneužití přihlašovacích údajů a API klíčů prostřednictvím malwaru, který infikoval zařízení uživatelů.

Kromě toho byly zaznamenány i konkrétní incidenty související s prompt inženýrstvím, tedy technikou vytváření a optimalizace vstupních dotazů pro model. V některých případech došlo k neúmyslnému vyzrazení testovacích dat, která chatbot vygeneroval v důsledku špatně konstruovaného nebo zneužitého promptu. To poukazuje na fakt, že i vstupní instrukce mohou být potenciálním bezpečnostním rizikem, pokud nejsou správně ošetřeny. [38]

V současné době rovněž existuje reálné riziko prompt injection útoků, kdy útočník navrhne prompt, jenž obsahuje skryté instrukce vedoucí například k uložení osobních údajů přihlášeného uživatele a jejich následnému odeslání na externí servery. Takový útok může být obzvláště nebezpečný ve chvíli, kdy uživatel nevědomky zkopíruje a použije prompt bez kontroly jeho obsahu. [39]

4.2.1 API volání třetích stran

Aplikace integruje LLM modely prostřednictvím API služeb třetích stran, jako jsou OpenAI, Google AI Studio nebo jiní poskytovatelé. Vzhledem k tomu, že tyto modely mohou zpracovávat citlivý obsah, je nezbytné rozumět tomu, jakým způsobem je s uživatelskými daty nakládáno.

Podle oficiální dokumentace [40, 41] k ochraně soukromí těchto služeb platí následující pravidla:

- **OpenAI API (placené tarify)** – Uživatelská data nejsou ukládána ani používána k trénování modelů. Za účelem detekce zneužití však mohou být požadavky a odpovědi uchovávány

po dobu až 30 dní, po které jsou automaticky odstraněny. Pro důvěryhodné zákazníky je navíc dostupný režim zero data retention, ve kterém nejsou data logována, to znamená, že těla požadavků a odpovědí existují pouze po dobu zpracování v paměti a poté jsou trvale smazána.

- **Google AI Studio API (tarif zdarma)** – Google uvádí, že využívá uživatelská data k trénování modelů, ale provádí filtrace citlivého obsahu.
- **Google AI Studio API (placená verze)** – V placených plánech Google AI API data neukládá ani nevyužívá k dalšímu zpracování.
- **Obecně** – Někteří poskytovatelé uchovávají data po dobu 30 dnů pro detekci zneužití služby, přičemž není vždy transparentní, zda mohou být tato data přístupná zaměstnancům.

4.2.2 Shrnutí k ochraně dat v aplikaci

Bezpečnost a ochrana dat jsou klíčovými aspekty návrhu aplikace SmartNotes. Šifrovaná komunikace, bezpečná autentizace uživatelů a transparentní nakládání s daty při integraci AI služeb třetích stran zajišťují vysokou úroveň ochrany soukromí. Přestože aplikace funguje jako cloud-first řešení, je důležité minimalizovat uchovávání citlivých informací a umožnit uživatelům kontrolu nad jejich daty.

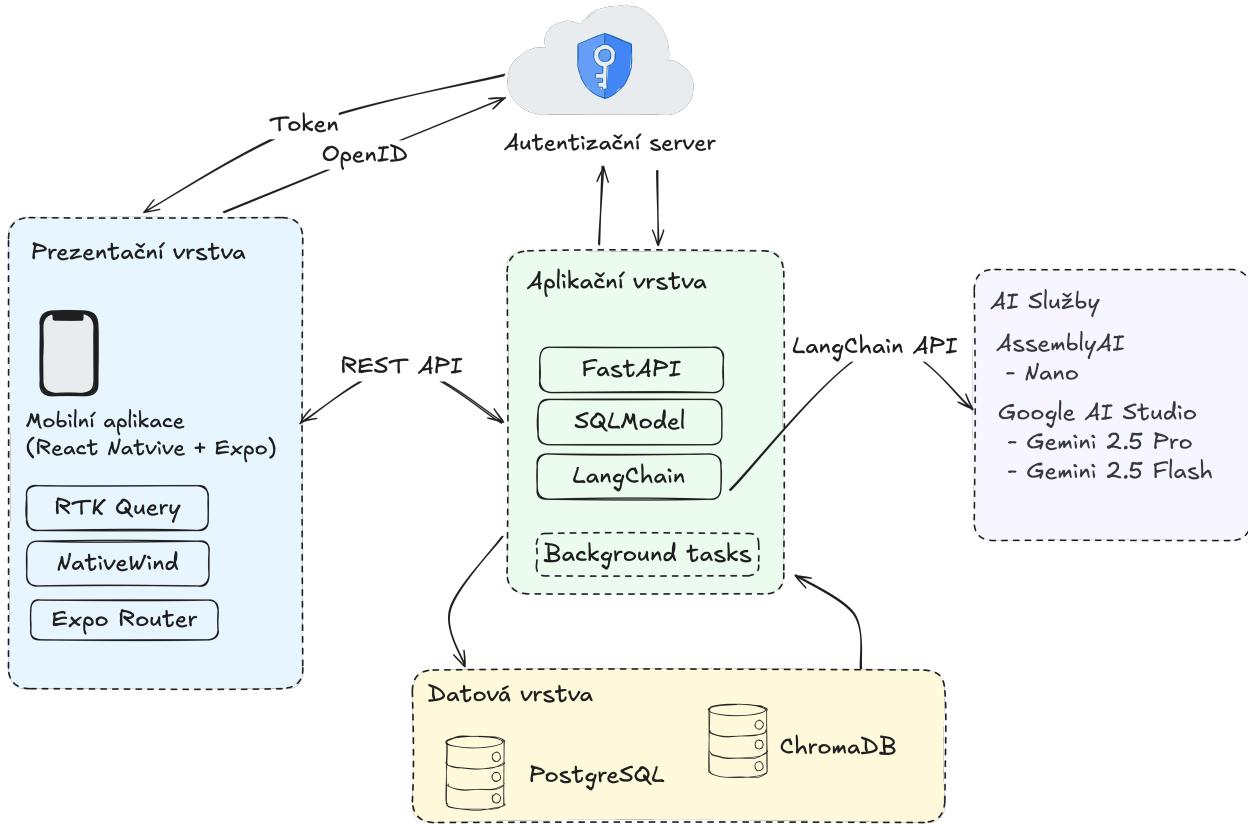
Jedním z možných řešení pro zvýšení bezpečnosti a nezávislosti na externích poskytovatelích by bylo hostování vlastního modelu pro zpracování AI funkcí. Existují open-source alternativy, jako například DeepSeek pro zpracování textových dotazů nebo OpenAI Whisper pro přepis hlasových poznámek, které by umožnily provoz AI modelů na vlastním serveru. Tento přístup by eliminoval závislost na třetích stranách a zajistil plnou kontrolu nad zpracovávanými daty.

Nicméně, vzhledem k tomu, že se v současné fázi jedná o prototyp aplikace, budou prozatím využívána API řešení třetích stran. Tento přístup umožní rychlejší implementaci a snadnější škálovatelnost, přičemž do budoucna zůstává možnost přechodu na vlastní hostované AI modely otevřená.

4.3 Architektura

Architektura aplikace se skládá ze dvou hlavních komponent, a to prezentační vrstvy, kterou tvoří nativní mobilní aplikace, a aplikační vrstvy, která zajišťuje správu dat, obsluhu požadavků a poskytování pokročilých funkcí, jež není možné efektivně realizovat přímo na mobilním zařízení.

V současné době se na mobilní zařízení postupně dostávají velké jazykové modely (LLMs), avšak pouze omezené procento zařízení disponuje dostatečným výpočetním výkonem pro jejich provoz. Z tohoto důvodu tato práce využívá klasický klient-server přístup, kde hlavní zpracování probíhá na straně serveru.



Obrázek 4.1: Vysokoúrovňové schéma architektury aplikace

4.3.1 Prezentační vrstva

Prezentační vrstva je realizována jako mobilní aplikace postavená na technologii **React Native** s využitím **Expo Managed Workflow**, což je doporučovaný standard přímo od vývojářů React Native podle oficiální dokumentace. Veškerá implementace je psaná v jazyce **TypeScript**.

Aplikace je strukturována podle metodologie **expo-router**, která využívá **file-based routing**. Tento přístup umožňuje organizaci jednotlivých obrazovek aplikace do logických a ucelených celků, čímž se zjednoduší navigace a správa kódu. Dále aplikace využívá **component-based architekturu**, což znamená, že jednotlivé části uživatelského rozhraní jsou implementovány jako znovuupoužitelné komponenty. Tímto způsobem lze efektivně spravovat kód a zajistit jeho snadnou rozšířitelnost.

Pro tvorbu komponent byla zvolena knihovna **React Native Reusables (RNR)**, která poskytuje CLI nástroje umožňující automatizované stahování předem definovaných komponent postavených na principech **shadcn**. Tento přístup nevyužívá klasické komponentní knihovny, ale pracuje s tzv. atomy, tedy základními prvky uživatelského rozhraní, ze kterých lze sestavovat složitější komponenty. Výhodou tohoto řešení je, že kód jednotlivých komponent je přímo součástí projektu ve formátu TSX souborů, což usnadňuje jejich přizpůsobení specifickým požadavkům aplikace.

Pro stylování aplikace je využita knihovna **NativeWind**, což je adaptace **Tailwind CSS** pro React Native. Tento přístup umožňuje efektivní správu stylů pomocí **utility-class** stylování, což zjednodušuje tvorbu designu komponent.

Komunikaci mezi klientem a serverem zajišťuje **RTK Query**, které nejen efektivně spravuje cache API volání, ale také umožňuje generování klientského kódu na základě **OpenAPI** specifikace. Balíček **@rtk-query/codegen-openapi** umožňuje vygenerovat kompletní sadu typovaných API funkcí (hooks) přímo z OpenAPI definice, což výrazně zjednoduší implementaci datových operací a snižuje riziko chyb při ručním programování API volání.

4.3.2 Aplikační vrstva

Jako aplikační server byl zvolen framework **FastAPI**, který představuje moderní a vysoce výkonné řešení pro tvorbu REST API. FastAPI je populární zejména díky své asynchronní architektuře, podpoře background tasků a nativní integraci s OpenAPI specifikací, což zjednoduší generování dokumentace a následné napojení klientských aplikací.

Pro práci s databází je využito rozšíření **SQLModel**, které kombinuje výhody **Pydantic modelu** (pro validaci dat) a **SQLAlchemy** (pro správu databázových operací). Díky tomu lze efektivně pracovat s ORM modelem, který podporuje různé datové zdroje a umožňuje snadnou migraci nebo škálování databázové vrstvy.

Další klíčovou integrací je knihovna **LangChain**, která umožňuje komunikaci s různými velkými jazykovými modely prostřednictvím jednotného API. Tento přístup zajišťuje flexibilitu při výběru a změně modelů podle aktuálních potřeb aplikace, aniž by bylo nutné měnit aplikační logiku.

FastAPI zároveň poskytuje automatické generování OpenAPI dokumentace, kterou lze vizualizovat pomocí **Swagger UI**. Tato dokumentace je dále využita pro generování API funkcí v prezentační vrstvě, což usnadňuje propojení mezi klientskou aplikací a serverem a zároveň minimalizuje riziko nesouladu mezi implementací API a jeho specifikací.

4.3.3 Datová vrstva

Realizace datové vrstvy závisí na dvou hlavních prvcích, relační databázi **PostgreSQL** a vektorové databázi **ChromaDB**.

PostgreSQL slouží jako primární úložiště pro persistenci a zpracování dat. Toto řešení bylo zvoleno zejména kvůli své široké popularitě, robustnosti a skutečnosti, že se jedná o open-source projekt s rozsáhlou komunitní podporou. PostgreSQL umožňuje efektivní správu relačních dat a poskytuje rozšířené možnosti optimalizace dotazů, což je klíčové pro výkonné a škálovatelné aplikace.

ChromaDB je rovněž open-source řešení, zaměřené na správu vektorových reprezentací dat. Hlavní výhodou je jeho nativní integrace s knihovnou LangChain, což umožňuje efektivní práci s embeddingy a vyhledáváním na základě podobnosti vektorů. Na rozdíl od cloudových služeb

pro správu vektorových databází je ChromaDB provozována lokálně na serveru, což přináší větší kontrolu nad daty a eliminuje závislost na externích poskytovatelích.

Datová vrstva je navržena tak, aby umožňovala transformaci perzistentních dat uložených v PostgreSQL do embeddingových reprezentací, které jsou následně ukládány ve vektorové databázi. Samotná ChromaDB však nemusí být perzistentní, což znamená, že embeddingy lze v případě potřeby generovat dynamicky ze základních datových struktur. Tento přístup umožňuje flexibilitu a možnost změnit LLM model pro vytváření embeddingových reprezentací.

4.3.4 Autentizace

Moderní autentizační mechanismy se stále více zaměřují na passwordless přístup, který nabízí řadu výhod oproti tradiční autentizaci založené na heslech. Hlavním přínosem tohoto přístupu je přenesení odpovědnosti za ověření identity na externí autorizační server, čímž se minimalizuje riziko úniku přihlašovacích údajů a zvyšuje se uživatelský komfort.

V této práci je implementována autentizace pomocí **OpenID Connect (OIDC)**, což je standardní autentizační vrstva postavená nad **OAuth 2.0**. Pro zajištění ověření uživatele byl zvolen poskytovatel Google, který umožňuje bezpečné přihlašování pomocí uživatelského účtu Google. Tento přístup zjednodušuje správu identity a zároveň poskytuje vysokou úroveň zabezpečení díky mechanismům, jako je dvoufaktorová autentizace (2FA) nebo detekce podezřelých přihlášení.

V budoucnu lze aplikaci snadno rozšířit o podporu dalších poskytovatelů identity, například **Apple ID** nebo dalších služeb podporujících standard OIDC. Díky tomu by bylo možné nabídnout uživatelům širší možnosti přihlášení a zároveň zachovat bezpečnostní standardy na vysoké úrovni.

4.3.5 Monitoring

Monitoring aplikace je klíčovou součástí testování a provozu v produkčním prostředí, jelikož umožňuje detekci chyb, sledování výkonnosti a analýzu uživatelského chování. V této práci jsou pro monitoring využity dvě služby - **Sentry** a **Vexo**.

Sentry je systém pro monitorování chyb v reálném čase, který poskytuje podrobné informace o výskytech chyb, což umožňuje jejich rychlou detekci a opravu. Kromě identifikace chyb dokáže Sentry sledovat výkonnost API volání i jednotlivých komponent aplikace, což pomáhá optimalizovat celkovou odezvu systému.

Vexo slouží jako nástroj pro analýzu uživatelského chování. Poskytuje přehled o interakcích uživatelů s aplikací prostřednictvím teplotních map obrazovek a záznamů uživatelských relací, což umožňuje detailní zpětnou vazbu týkající se používání aplikace. Tímto způsobem lze lépe pochopit uživatelskou cestu, identifikovat problematická místa v rozhraní a optimalizovat celkový uživatelský zážitek. Vexo rovněž umožňuje sledování typů zařízení, která uživatelé používají, což napomáhá při optimalizaci aplikace pro různé platformy.

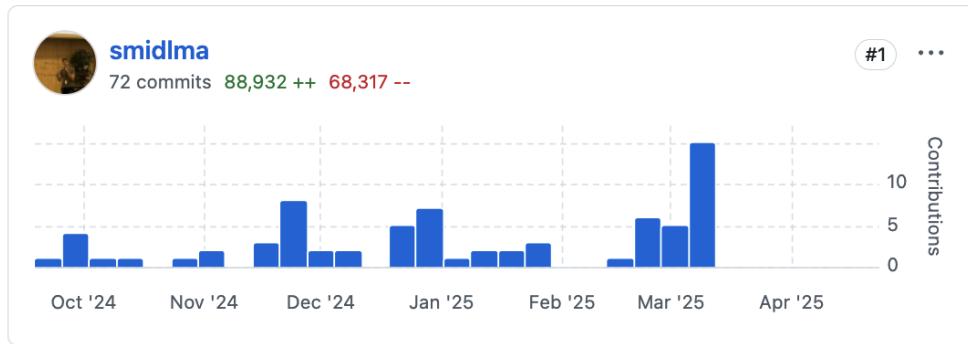
Kombinací těchto dvou nástrojů je zajištěn komplexní monitoring jak z hlediska technického výkonu aplikace, tak i z pohledu uživatelského chování, což umožňuje efektivní ladění a neustálé zlepšování aplikace.

4.4 Nastavení vývojového procesu

Vývoj aplikace probíhal agilním způsobem, přičemž hlavní metodologií byla inkrementální implementace, která zajišťovala postupné přidávání nových funkcionalit k již funkčnímu celku.

Na základě definovaných user stories byly jednotlivé úkoly organizovány ve formě ticketů v ticketovém systému Trello. Tento systém umožňoval efektivní správu vývoje, sledování průběhu úkolů a transparentní řízení priorit. Každá iterace se zaměřovala na postupné rozšiřování funkcionalit, přičemž důraz byl kladen na kontinuální integraci a testování.

Díky tomuto přístupu bylo možné pružně reagovat na nové požadavky, upravovat plán vývoje podle získané zpětné vazby a minimalizovat riziko vývoje nevyhovujícího řešení.



Obrázek 4.2: Aktivita repozitáře od začátku vývoje

4.5 Příprava vývojových nástrojů

Pro správu verzování kódu byl inicializován **Git** repozitář na mé GitHub profilu, který slouží jako centrální úložiště pro vývoj aplikace.

Jako hlavní vývojové prostředí bylo zvoleno **Visual Studio Code (VSCode)** s nainstalovanými rozšířeními pro zajištění kvality kódu. Pro statickou analýzu v prezentační vrstvě je využíván **ESLint** a **Prettier**, který zajišťuje konzistentní formátování kódu. Pro část aplikační vrstvy napsanou v jazyce Python je použit **Black formatter**, který slouží k automatickému formátování kódu dle stanovených standardů. Kromě těchto nástrojů je integrován také **Husky**, jenž umožňuje provádění kontroly kódu před jeho odesláním do vzdáleného GitHub repozitáře na commit a push.

Pro vývoj mobilní aplikace je nutné mít nainstalovanou aktuální verzi **Xcode**, která umožňuje sestavení aplikace pro cílové zařízení s operačním systémem iOS. Pro vývoj v development režimu

přímo na fyzickém zařízení bylo nutné vytvořit **Expo Development Build**. Tento build představuje nativní aplikaci, která je sestavena přímo pro konkrétní zařízení a obsahuje aktuální nativní knihovny. Každá změna v nativní části aplikace, tedy mimo JavaScript kód, vyžaduje aktualizaci tohoto buildu, aby reflektoval nejnovější úpravy v nativním prostředí.

4.6 Implementace

Tato sekce popisuje implementační detaily klíčových částí mobilní aplikace SmartNotes, jejichž cílem je usnadnit uživatelům správu poznámek včetně hlasových záznamů a připojených dokumentů. Aplikace byla navržena s důrazem na přehledné uživatelské rozhraní, intuitivní ovládání a integraci funkcí podporovaných umělou inteligencí, jako je chytré vyhledávání a summarizace obsahu.

Jednotlivé podsekce této kapitoly se věnují konkrétním modulům systému, od přehledu uložených poznámek přes poznámkový editor s podporou formátovaného textu a multimédií až po komponentu chytrého vyhledávání, která využívá vektorové databáze pro sémantické dotazování. Pozornost je věnována také technickým aspektům implementace, například práci s vektorovými embeddingy, strukturou metadat nebo integraci knihoven třetích stran.

Celý návrh byl ovlivněn praktickými požadavky na použitelnost v reálném prostředí, což se odráží například ve volbě komponent, způsobu interakce uživatele s obsahem nebo optimalizaci pro světlý a tmavý režim. Důraz byl kladen rovněž na modularitu jednotlivých částí, aby bylo možné systém v budoucnu dále rozšiřovat a přizpůsobovat specifickým potřebám cílových uživatelů.

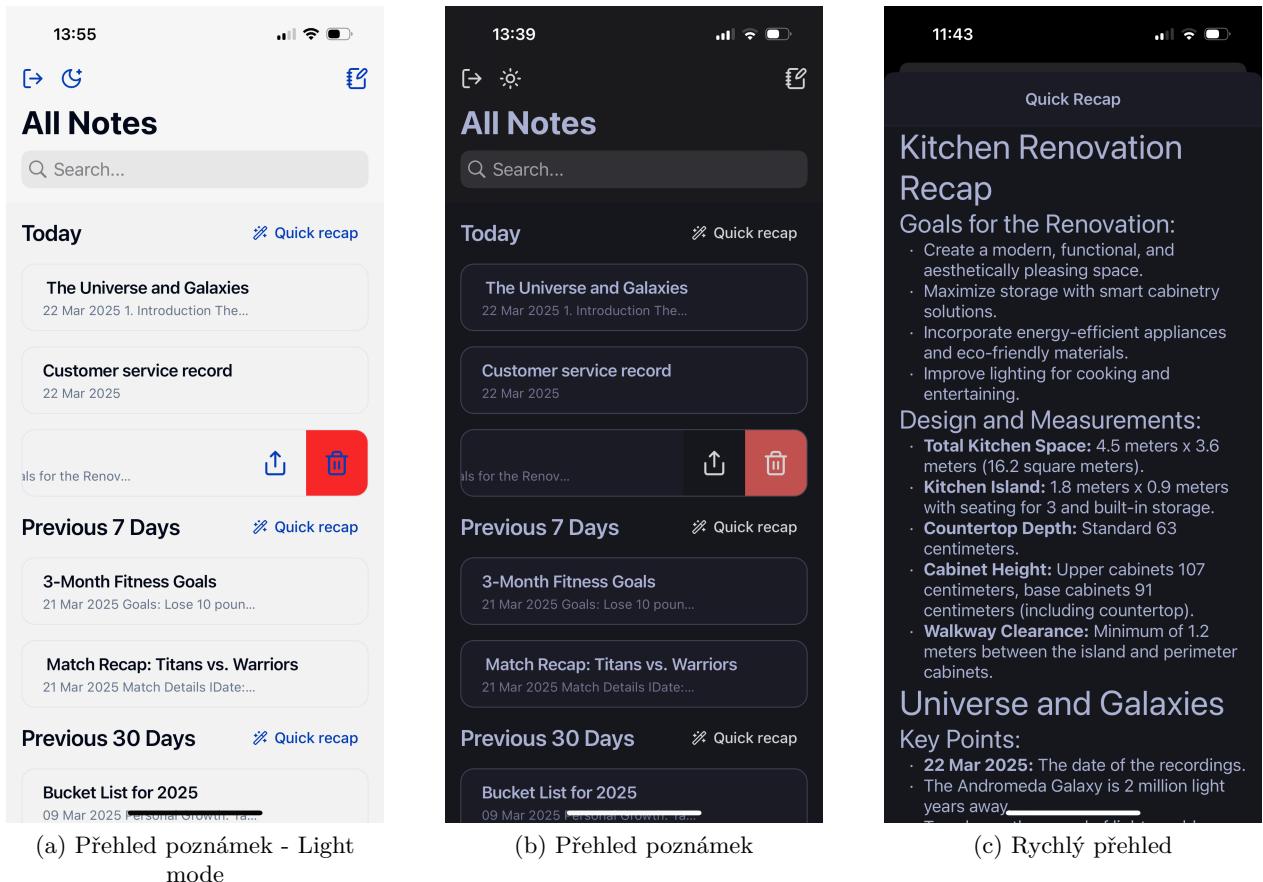
4.6.1 Přehled poznámek

Po úspěšném přihlášení je uživateli zobrazena hlavní obrazovka aplikace, která slouží jako výchozí bod pro správu poznámek. Rozhraní je navrženo s důrazem na jednoduchost, přehlednost a snadnou orientaci v uložených informacích.

Poznámky jsou na této obrazovce seskupeny dle data poslední úpravy, čímž je uživateli umožněn rychlý přístup k naposledy editovanému obsahu. Každá jednotlivá poznámka je zobrazena jako samostatný blok s možností provedení dvou základních akcí – sdílení poznámky a jejího odstranění.

V pravém horním rohu obrazovky je umístěno tlačítko pro vytvoření nové poznámky, které je trvale přístupné bez ohledu na aktuální pozici uživatele v seznamu. Bezprostředně dostupná je rovněž vyhledávací lišta, která slouží jako vstupní bod pro chytré vyhledávání napříč všemi poznámkami v aplikaci.

Každá skupina poznámek má dále k dispozici akci pro vygenerování rychlého přehledu (shrnutí). Tato funkce umožňuje souhrnné zobrazení obsahu všech poznámek v dané skupině, přičemž jsou summarizovány summarizace jednotlivých poznámek.



Obrázek 4.3: Hlavní obrazovka aplikace

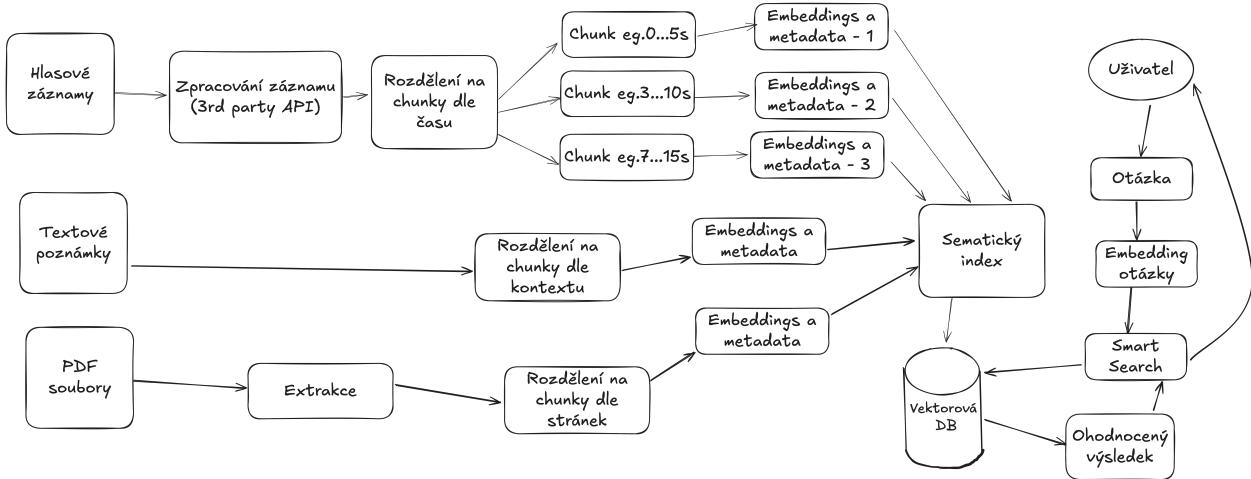
Uživatel má také možnost přepínat mezi světlým a tmavým režimem rozhraní, čímž si může vzhled aplikace přizpůsobit podle vlastních preferencí nebo aktuálních světelných podmínek. Design obrazovek je na obrázku 4.3.

4.6.2 Chytré vyhledávání

Chytré vyhledávání v aplikaci pracuje se všemi dostupnými uživatelskými daty a usiluje o nalezení co nejrelevantnějších odkazů na požadovaný obsah. Toho je dosaženo prostřednictvím předzpracování vstupních dat, jejichž převodu do vektorového prostoru a následného uložení do vektorové databáze.

Aby bylo možné uchovávat informace ve formátu vhodném pro vyhledávání, je nutné, aby vstupní data prošla předzpracováním. V rámci tohoto procesu jsou extrahována a strukturována metadata, která jsou následně připojena ke každému fragmentu dokumentu ve vektorové databázi.

Během předzpracování jsou využity specifické techniky, které se liší podle vstupního formátu zpracovávaných dat:



Obrázek 4.4: Diagram architektury chytrého vyhledávání

- **Textové poznámky** jsou zpracovány pomocí nástroje `HTMLHeaderTextSplitter`, který rozděluje obsah podle strukturálních HTML značek (`<h1>` až `<h3>`) s využitím překryvů (overlap) pro zachování kontextu mezi jednotlivými bloky.
- **PDF dokumenty** jsou rozdělovány pomocí `RecursiveCharacterTextSplitter`. Nejprve je dokument segmentován po jednotlivých stránkách a poté jsou tyto části dále děleny do logickejších celků (chunků), ke kterým jsou uložena metadata, například informace o čísle stránky, na které se daný fragment nachází.
- **Hlasové záznamy** představují nejjednodušší část systému. Díky využití přepisů s časovými razítkami (timestamps) pro každé slovo je možné přesně identifikovat, ve kterém čase se požadovaná informace nachází. Na základě těchto údajů jsou tvořeny segmenty (chunky) s časovými překryvy a do metadat jsou zaznamenány časy začátku a konce každého segmentu.

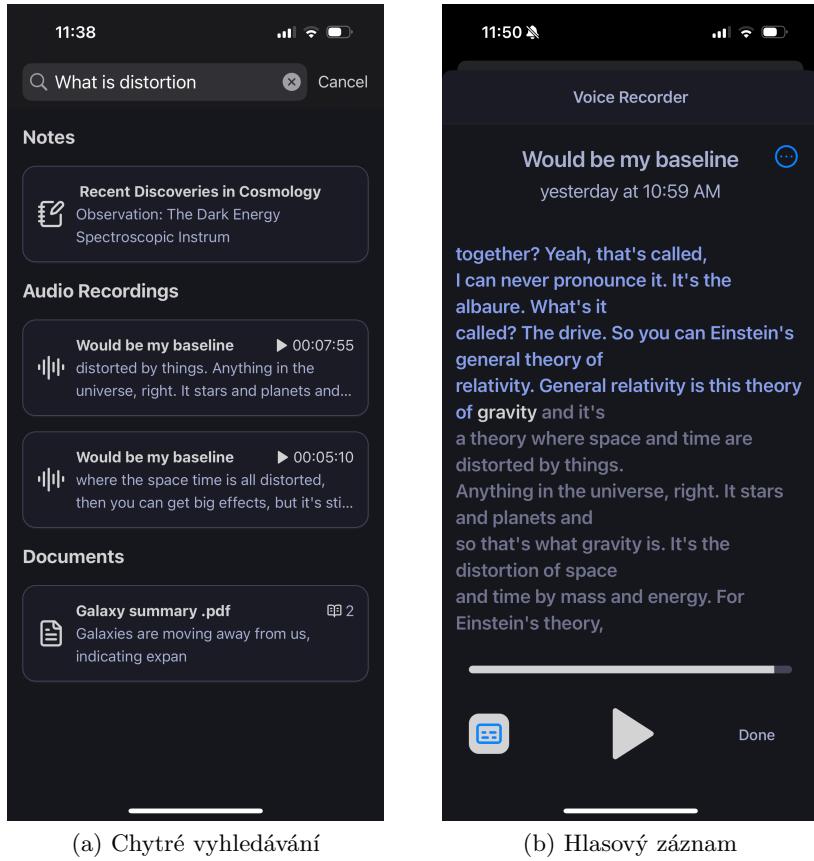
Díky tomuto přístupu může uživatel sémanticky přesněji vyhledávat, přičemž výsledky obsahují přímé odkazy na konkrétní místo v datech – například otevření PDF dokumentu na dané stránce nebo spuštění audiozáznamu přesně v čase, kdy byl relevantní obsah vysoven. Pokud je například vrácen výsledek s časem 07:55, uživateli se otevře přehrávač hlasového záznamu přesně v tomto čase a zároveň se zobrazí příslušný výsek přepisu.

Algoritmus sémantického vyhledávání paralelně vyhledává ve třech samostatných kolekcích vektorové databáze, konkrétně v kolekcích:

```

NOTES_COLLECTION_NAME = "note_embeddings"
VOICE_COLLECTION_NAME = "voice_embeddings"
DOCUMENT_COLLECTION_NAME = "document_embeddings"

```

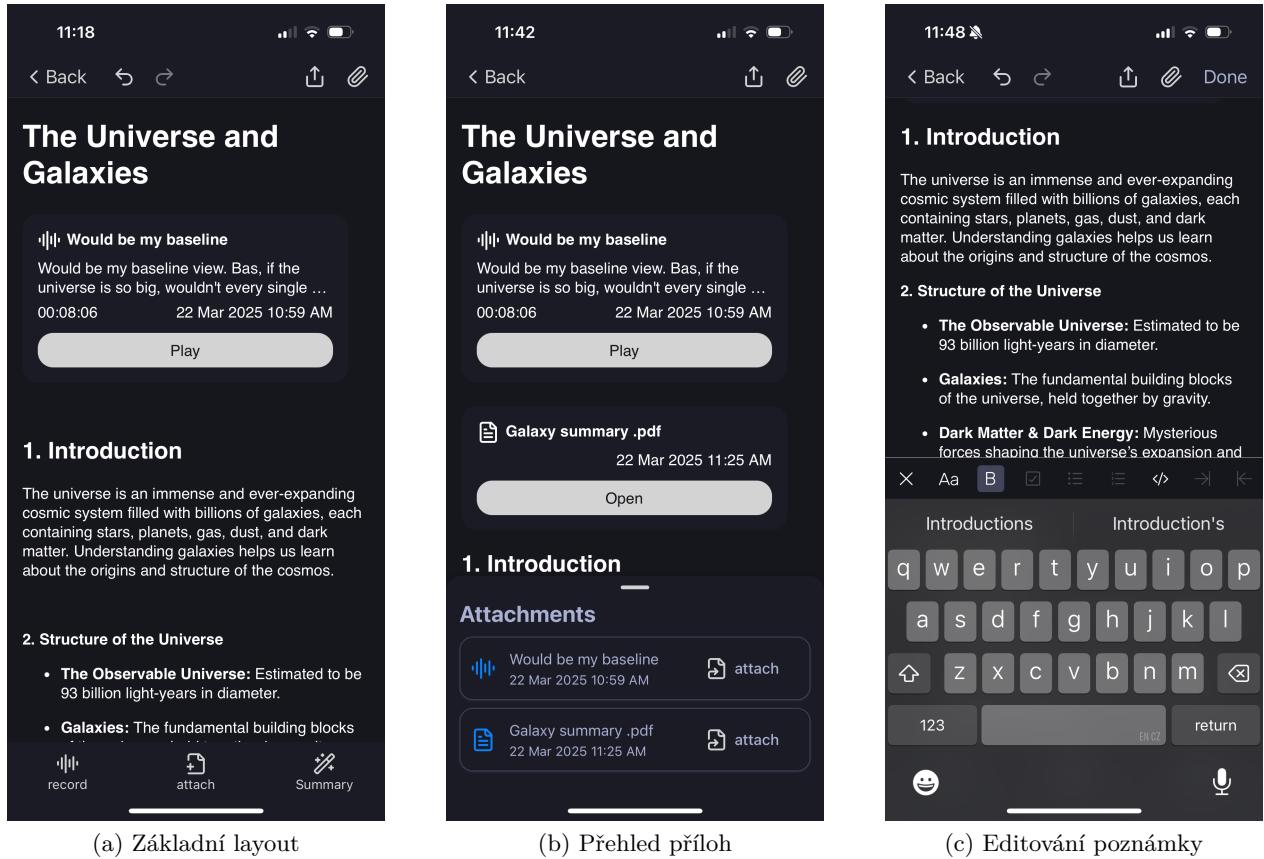


Obrázek 4.5: Chytré vyhledávání a detail hlasového záznamu

Po provedení vyhledávání nad těmito kolekcemi je dynamicky spočítán práh relevance (*threshold*) na základě skóre nalezených výsledků. Tento práh slouží k filtrování nerelevantních odpovědí a zajišťuje, že mezi výsledky budou pouze ty záznamy, které odpovídají dotazu s dostatečnou přesností. Finálním výstupem je seznam výsledků seřazený dle jejich relevance, který je následně zobrazen uživateli ve formě konkrétních odkazů na lokalizovaný obsah. Výsledek je uveden na obrázku aplikace 4.5.

4.6.3 Poznámkový editor

Komponenta pro práci s formátovaným textem (tzv. Rich Text editor) představuje v libovolné technologii složitou implementační výzvu, což platí i v prostředí React Native. V současné době neexistuje jednoznačné řešení tohoto problému, což je popsáno také v oficiální dokumentaci [42] Expa. Dostupných je několik alternativních přístupů, přičemž každý z nich má určitá omezení. V rámci této práce byla proto zvolena knihovna **10tap-editor**, což je wrapper založený na komponentě **WebView**, který integruje populární webový editor **tiptap**.



Obrázek 4.6: Poznámkový editor

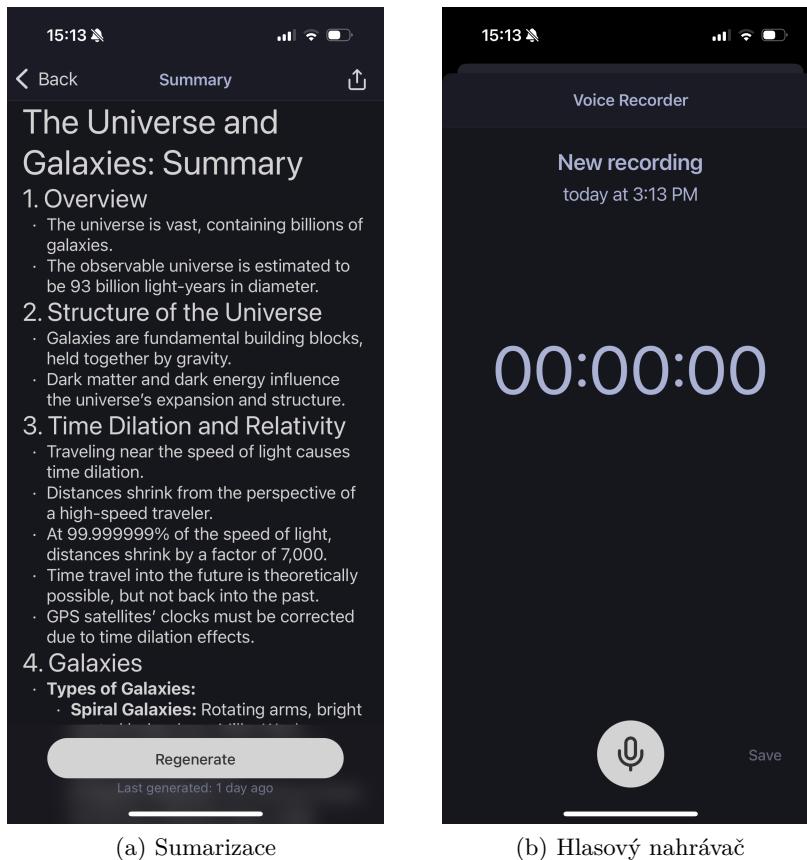
Zvolený editor 10tap-editor podporuje všechny základní operace a formátovací možnosti, které uživatel od Rich Text editoru očekává. Pro potřeby této práce však bylo nezbytné implementovat vlastní rozšíření, umožňující vkládání multimedialních prvků, konkrétně hlasových záznamů a dokumentů přímo do editoru. V průběhu implementace se objevilo několik technických problémů, které byly následně reportovány prostřednictvím platformy GitHub Issues a řešeny ve spolupráci se správci knihovny.

Pro vlastní implementaci potřebného rozšíření bylo nutné analyzovat stávající architekturu knihovny a její moduly. Implementace konkrétně spočívala v definování vlastního Node objektu reprezentujícího mediální komponentu v rámci WebView editoru. Dále bylo zapotřebí vytvořit rozšíření typu **BridgeExtension**, které zprostředkovává komunikaci mezi nativní částí aplikace a prostředím WebView. K přenosu zpráv a příkazů byly využity **obsluhy zpráv**, jejichž prostřednictvím byly integrovány specifické funkcionality nezbytné pro daný případ užití. Výsledný poznámkový editor je zobrazen na obrázcích 4.6.

Obrazovka poznámkového editoru disponuje rovněž dolním ovládacím panelem, který uživatelům umožňuje rychle a jednoduše nahrát nový hlasový záznam, vložit soubor nebo spustit summarizaci

aktuální poznámky. Při nahrávání hlasového záznamu se automaticky otevře systémové modální okno s jednoduchou nahrávací komponentou. Hlasový záznam je možné pořizovat také v režimu na pozadí, a to i v situacích, kdy je zařízení uzamčené.

Funkcionalita summarizace poznámky přesune uživatele do nového okna, ve kterém se zobrazí poslední provedená summarizace, pokud existuje. V opačném případě aplikace automaticky vygeneruje novou summarizaci. Uživateli je dále umožněno kdykoli ručně vygenerovat summarizaci novou a vidí i dobu vytvoření summarizace. Hotové poznámky a summarizace lze kdykoli jednoduše sdílet pomocí exportu do formátu PDF, a to přes standardní systémové dialogové okno pro sdílení souborů. Výsledek je na obrázku 4.7



Obrázek 4.7: Obrazovky summarizace a hlasového nahrávače

4.7 Příprava na testování aplikace

Testování aplikace je klíčovým krokem před jejím nasazením do produkčního prostředí. Proces zahrnuje jak distribuci mobilní aplikace pro interní testování, tak i nasazení serverové části aplikace.

Pro mobilní aplikaci je využita platforma **Apple Connect** a služba **TestFlight**, zatímco serverová část je hostována prostřednictvím služby **Railway**.

4.7.1 Distribuce a interní testování mobilní aplikace

Pro distribuci mobilní aplikace a její interní testování je nutné mít Apple Developer účet, který umožňuje generování potřebných certifikátů a správu buildů v platformě Apple Connect. Tato služba umožňuje nahrávání verzí aplikace, jejich správu a následnou distribuci testerům.

Ke zjednodušení celého procesu testování a distribuce je využito Expo a EAS. V rámci projektu mobilní aplikace je nutné přidat konfigurační soubor `eas.json`, který je nakonfigurován dle oficiální EAS dokumentace. Tento soubor definuje profily použité při sestavení aplikace na serverech EAS, dále environment proměnné, například URL k API, bezpečnostní klíče a další, které jsou dostupné na EAS serverech.

Po správném nastavení lze sestavit aplikaci pomocí příkazu:

```
eas build -platform ios -profile production
```

Tento příkaz provede sestavení aplikace na EAS serverech, přičemž výsledný build je uložen na Expo účet. Po dokončení buildu je možné aplikaci nahrát do Apple Connect příkazem:

```
eas submit -platform ios
```

Během tohoto procesu se systém dotáže na výběr konkrétního buildu, poté se propojí s Apple Connect serverem a zahájí jeho nahrávání.

Po úspěšném nahrání buildu je v Apple Connect nutné ručně vyplnit požadované informace a odeslat build ke schválení. Po schválení je možné aplikaci distribuovat testerům pomocí služby TestFlight. V rámci Apple Connect je vytvořena testovací skupina, do které jsou přidány e-mailové adresy testerů. Ti následně obdrží e-mail s pokyny pro stažení aplikace a její testování prostřednictvím aplikace TestFlight, která umožňuje testování aplikací před jejich oficiálním zveřejněním v App Store.

4.7.2 Nasazení serverové části aplikace

Serverová část aplikace je nasazena na platformu Railway, která umožňuje efektivní správu a škálování serverových služeb.

Proces nasazení zahrnuje několik kroků. Nejprve je nutné nastavit environment proměnné, které obsahují konfiguraci API a bezpečnostní klíče. Následně se propojí GitHub repozitář s Railway, což umožňuje automatizované nasazování při každé změně v repozitáři. Dále je nutné definovat root adresář a startovací příkaz, který se spouští při startu aplikace.

4.8 Testování aplikace

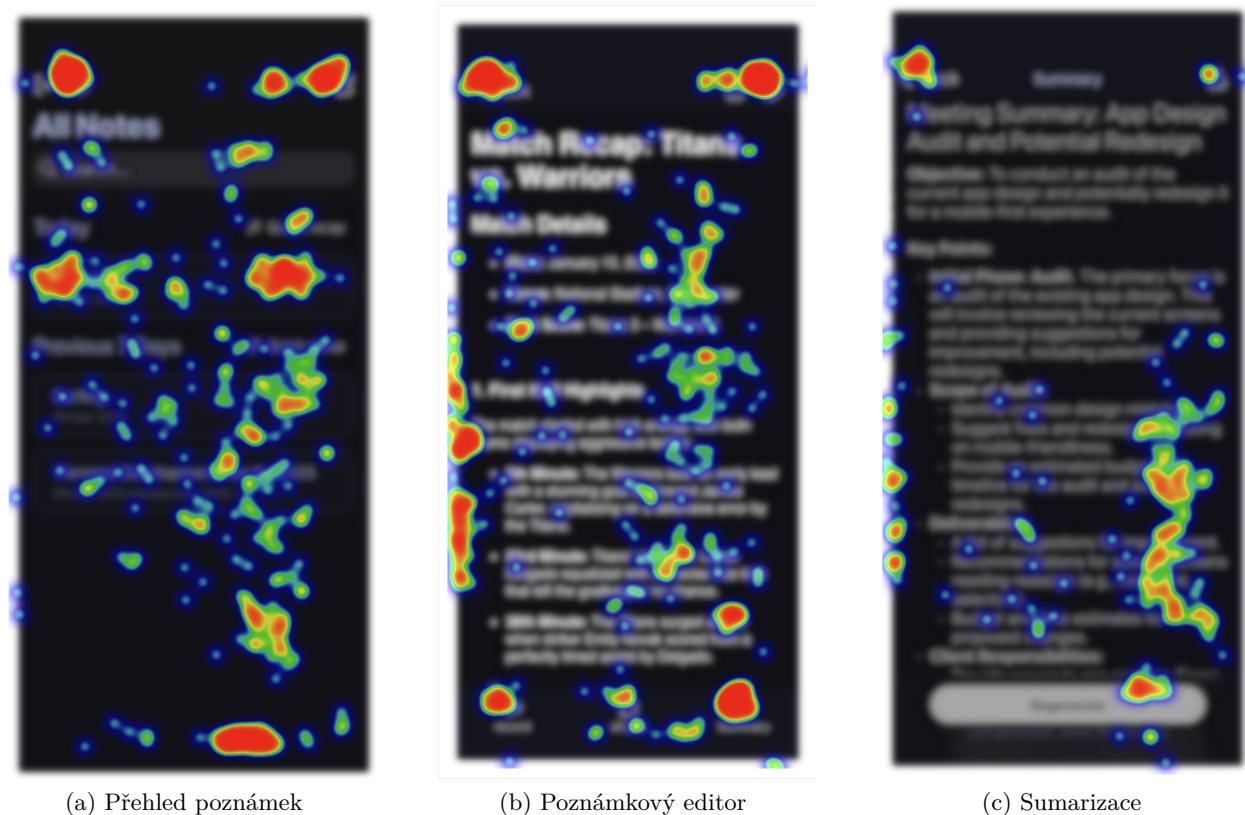
Do testování aplikace se zapojilo pět uživatelů. Testování probíhalo v prostředí softwarové firmy, kde primárním cílem aplikace bylo zachytit důležité momenty meetingů a podpořit produktivitu práce jednotlivých uživatelů. Nejčastěji se jednalo o meetingy týmu v souvislosti s agilními ceremoniemi, vedené v českém a slovenském jazyce. Vedoucí meetingu byl následně schopen sdílet sumarizaci meetingu prostřednictvím aplikace a také zpětně vyhledávat klíčové momenty díky chytrému vyhledávání, což bylo užitečné při hledání nesrovnaností v komunikaci mezi členy týmu. Dále byla aplikace využita pro zaznamenání komunikace s externími dodavateli, se kterými hovory probíhaly v anglickém jazyce. Uživatel zodpovědný za komunikaci byl schopen jednoduše doložit ostatním členům týmu výsledek jednání a měl možnost vytvářet poznámku týkající se tohoto konkrétního dodavatele. Také mohl zpětně vyhledávat a připomínat si fakta zmíněná během meetingu bez potřeby přehrávání celého hlasového záznamu. Uživatelé dále využívali aplikaci k osobním potřebám i mimo prostředí firmy.

4.8.1 Analýza chování uživatelů

Pro sledování a vyhodnocování interakce uživatelů s aplikací byly využity uživatelské relace a teplovní mapy jednotlivých obrazovek. Díky integraci nástroje Vexo bylo možné jednoduše sledovat chování uživatelů prostřednictvím přehledného dashboardu. Ten umožňuje přehrávání záznamů uživatelských relací, zobrazení četnosti interakcí s konkrétními prvky uživatelského rozhraní, přístup ke statistikám o celkovém počtu relací a základní monitorovací metriky.

Teplotní mapy slouží jako nástroj pro vizualizaci intenzity uživatelských interakcí na jednotlivých obrazovkách aplikace. V rámci této vizualizace označuje červená barva vysokou míru interakce, zelená střední a modrá nízkou. Na základě těchto barevných vzorců lze odvozovat, zda uživatelé správně chápou rozložení prvků v uživatelském rozhraní a zda intuitivně využívají nabízené funkcionality. V případech, kdy dochází k interakcím v oblastech, které interaktivní nejsou, je možné identifikovat tzv. nečekané interakce. Ty často signalizují problematické aspekty návrhu rozhraní, například nevhodné umístění ovládacích prvků, nedostatečnou vizuální zpětnou vazbu nebo chybějící indikaci interaktivních oblastí.

Na obrazovce s přehledem poznámek 4.8a byla zaznamenána nejvyšší míra interakce u tlačítka pro vytvoření nové poznámky, dále u ovládacího prvku dark/light mode aplikace a také v sekci „Dnes upravované“, kde uživatelé často vybírají poznámky k úpravě. Zvýšená míra interakce byla pozorována také v oblasti vyhledávací lišty, nicméně vzhledem k tomu, že se po kliknutí na tuto lištu na stejně obrazovce zobrazuje i komponenta chytrého vyhledávání, nelze jednoznačně určit, zda se interakce týká samotného vyhledávání, nebo předchozího stavu obrazovky. Tento nedostatek plyne z omezení nástroje Vexo, který v tomto případě nerozlišuje jednotlivé dynamické změny v rámci jedné obrazovky.

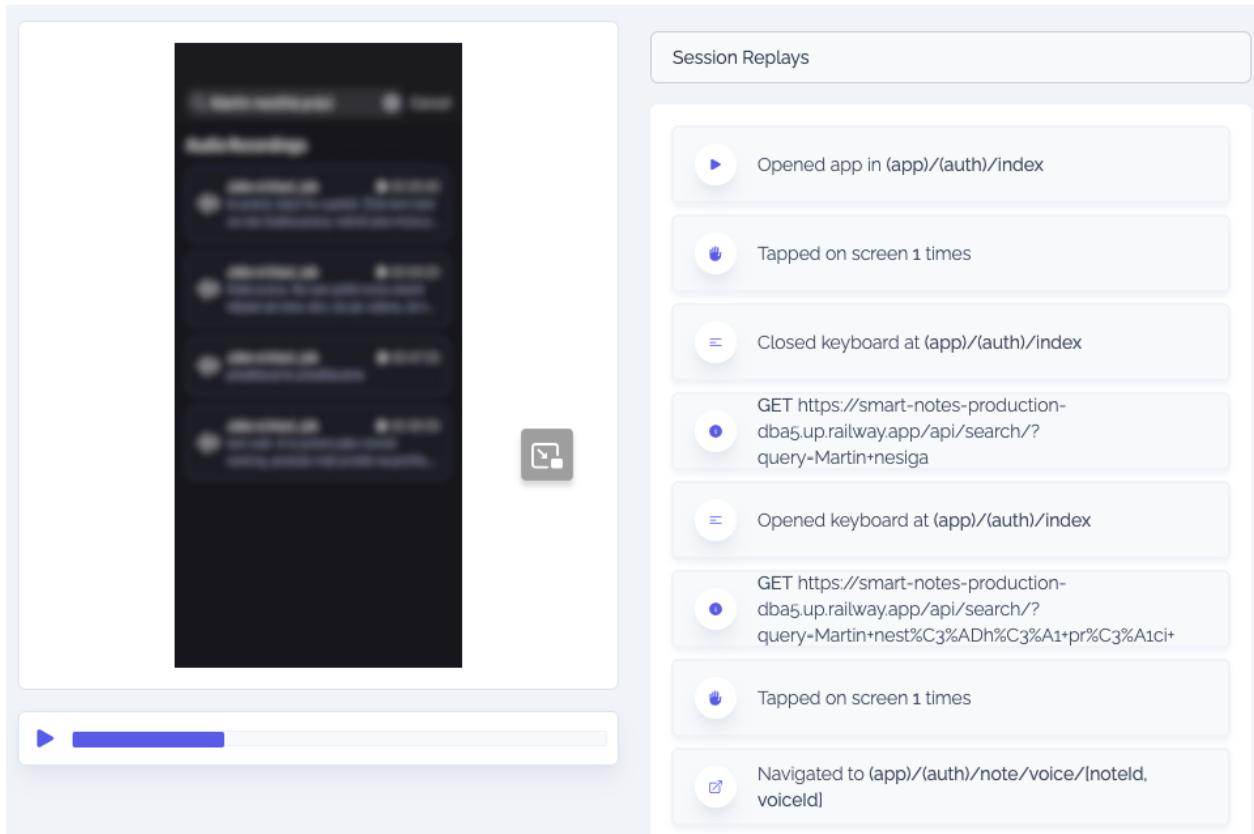


Obrázek 4.8: Teplotní mapy obrazovek

Na obrazovce editoru 4.8b bylo možné sledovat několik klíčových interakčních vzorců. Zaznamenaná byla například tendence uživatelů vracet se na předchozí obrazovku pomocí tlačítka „Zpět“ i pomocí gest typických pro iOS zařízení, především gesta přejetí prstem z levého okraje obrazovky. Výrazná míra interakce byla zaznamenána v dolní části obrazovky, kde se nachází ovládací panel editoru. Nejčastěji zde uživatelé využívali funkci sumarizace poznámky a nahrávání hlasového záznamu. Naproti tomu připínání dokumentů vykazovalo nižší četnost použití. Vysoká míra interakce byla rovněž zaznamenána u tlačítek pro sdílení a zobrazení příloh, což potvrzuje jejich vhodné umístění a dobrou dostupnost v rámci rozhraní. Malé interakce po celé obrazovce naznačují interakci s editorem poznámek.

Detailnější analýza chování uživatelů byla umožněna prostřednictvím přehrávání zaznamenaných uživatelských relací. Nástroj Vexo poskytuje videozáznamy reálných interakcí uživatelů s aplikací, včetně chronologicky zaznamenaných událostí, jako jsou kliknutí, přechody mezi obrazovkami nebo zadávání textu. Tyto záznamy umožňují přesně sledovat, jak uživatelé aplikaci používají, jaké akce vykonávají a zda při interakci dochází k neočekávanému chování.

Na obrázku 4.9 je zachycen příklad takové relace, kdy uživatel využívá chytré vyhledávání. Po zadání hledaného výrazu vybírá jeden ze zobrazených výsledků a následně otevírá hlasový záznam



Obrázek 4.9: Obrazovka relace uživatele

na konkrétní časové pozici, která odpovídá významovému kontextu hledaného výrazu. Tento typ vizuálního záznamu výrazně napomáhá při ladění uživatelského rozhraní i při ověřování správnosti implementace klíčových funkcionalit.

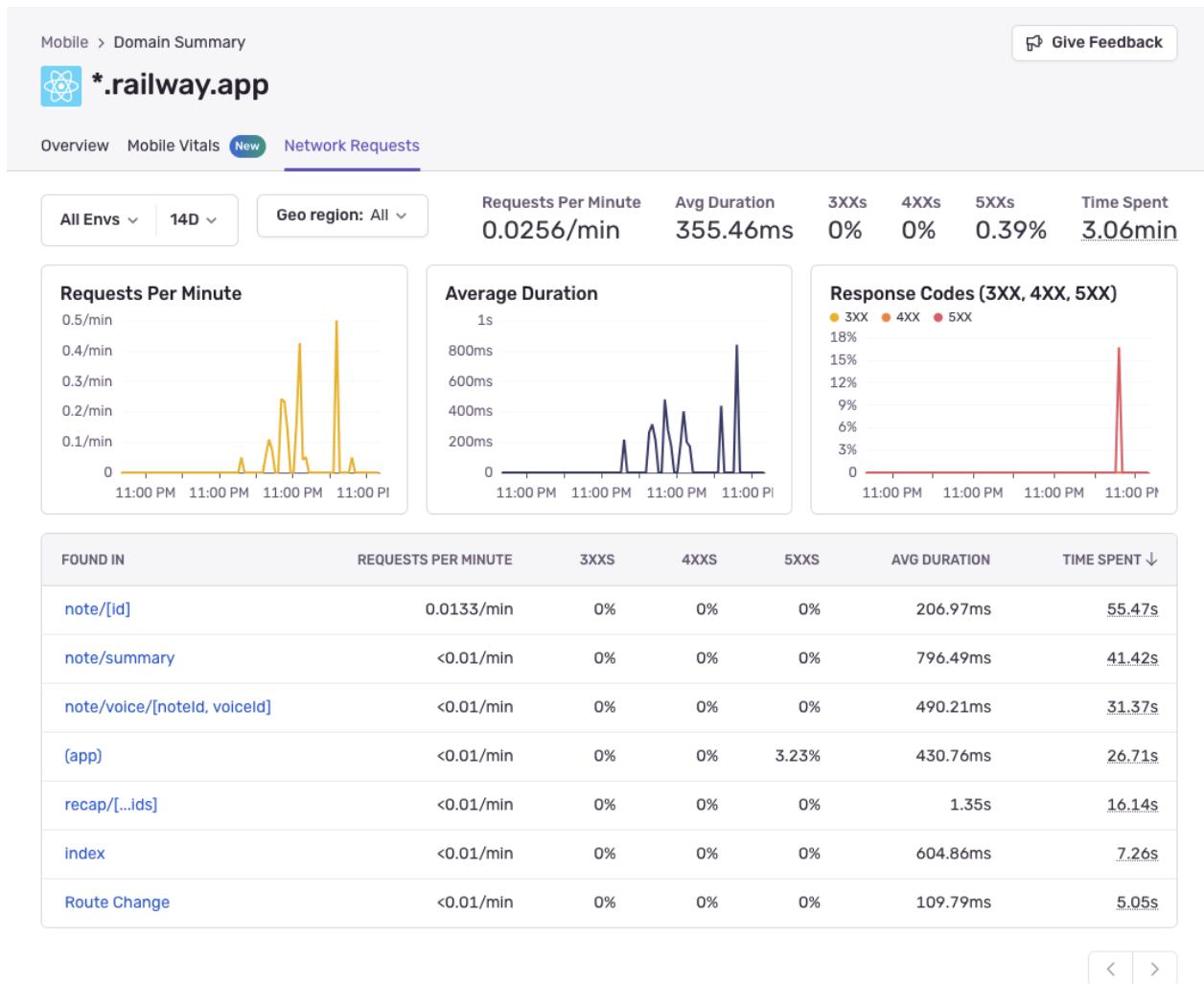
Přehrávání uživatelských relací rovněž pomohlo identifikovat konkrétní problémy v uživatelském rozhraní, které by jinak mohly zůstat nepovšimnutý. Například bylo zjištěno, že po uložení nově nahraného hlasového záznamu jsou někteří uživatelé zmateni a nenapadne je ihned kliknout na ikonu pro zobrazení příloh u dané poznámky. To vedlo k nejistotě ohledně toho, zda byla hlasová poznámka skutečně uložena a kde ji lze následně nalézt.

Tento problém byl dále konzultován s uživateli a na základě zpětné vazby bylo navrženo konkrétní řešení. Spočívalo ve změně samotné ikony, která působila nejednoznačně, a zároveň ve vizuálním zvýraznění právě uložené přílohy. Tímto způsobem bylo dosaženo lepší srozumitelnosti a intuitivního chování uživatelského rozhraní po provedení akce.

4.8.2 Monitorování výkonu

Pro sledování výkonnostních metrik v produkční verzi aplikace byl využit nástroj Sentry, který umožňuje nejen detekci chyb, ale také sledování chování aplikace v reálném čase prostřednictvím

síťových a aplikáčních metrik. V průběhu měsíčního testování se aplikace jevila jako stabilní, přičemž byl zaznamenán pouze jeden kritický incident při síťovém požadavku, který byl způsoben chybou na straně serveru. Na klientské straně aplikace nedošlo k žádným závažným výpadkům.



Obrázek 4.10: Sentry - Síťový monitoring

Průměrná doba trvání všech síťových požadavků byla přibližně **355,46ms**. Jednotlivé obrazovky jsou referovány jako navigační cesty v aplikaci. Nejčastěji využívaná obrazovka poznámkového editoru `/note/[id]` dosahovala průměrné doby odpovědi **206,97ms**, což je vyhovující i pro nasazení v produkčním prostředí. Vyšší doby odpovědi byly zaznamenány u požadavků, které na serverové straně využívaly API třetích stran, například u obrazovky pro summarizaci poznámky `/note/summary`, kde došlo v průměru k odezvě **796,49ms**. Výraznější latence byly pozorovány i u generování rychlých přehledů na obrazovce `/recap/[...ids]` s průměrnou hodnotou **1,35s**. Na základě těchto údajů je možné navrhnut optimalizované řešení, jako je například streamování od-

povědí ze serveru, přesunutí části síťových operací do pozadí aplikace a upravit vizuální zpětné vazby pro uživatele v průběhu dlouhotrvajících operací.

Je nutné zohlednit, že serverová část běžela na bezplatné infrastrukturu Railway, která je omezená výkonem i počtem současných požadavků. To se projevilo na výsledných časech odezvy, které jsou vyšší, než by byly v případě optimalizovaného produkčního prostředí.

Z pohledu výkonu aplikace při startu byly výsledky velmi uspokojivé. Studený start aplikace dosahoval v průměru **1,64s**, zatímco teplý start byl měřen s hodnotou **242,73ms**. Obě tyto hodnoty se nacházejí pod doporučenými limity (do 5 sekund pro studený start a do 1,5 sekundy pro teplý start) uváděnými v článku [43]. Tím je zajištěna pozitivní UX již při prvním spuštění aplikace. Je rovněž důležité zmínit, že pokud aplikace nenačte do 20 sekund, operační systém iOS ji automaticky ukončí.

Ve sledovaných transakcích 4.11 nebyly detekovány žádné zamrznuté snímky (frozen frames), což znamená, že nedocházelo k blokacím hlavního vlákna při vykreslování UI. Poměr pomalých snímků (slow frames) se pohyboval od **4,82%** u obrazovky summarizace až po **11,76%** u hlavní obrazovky aplikace. Nejvyšší zátěž v tomto směru vykazovala komponenta pro přehrávání hlasových záznamů (note/voice), kde bylo zaznamenáno **11,71%** pomalých snímků a současně až **21,04%** výskytu tzv. stall frames, tedy kdy JavaScript event loop trvá déle, než se očekává.

Dále byla zaznamenána metrika TTID (Time to Initial Display) s hodnotou **1,59s**, která reprezentuje čas do prvního zobrazení obsahu uživateli. Tato hodnota je opět v mezích, které lze očekávat.

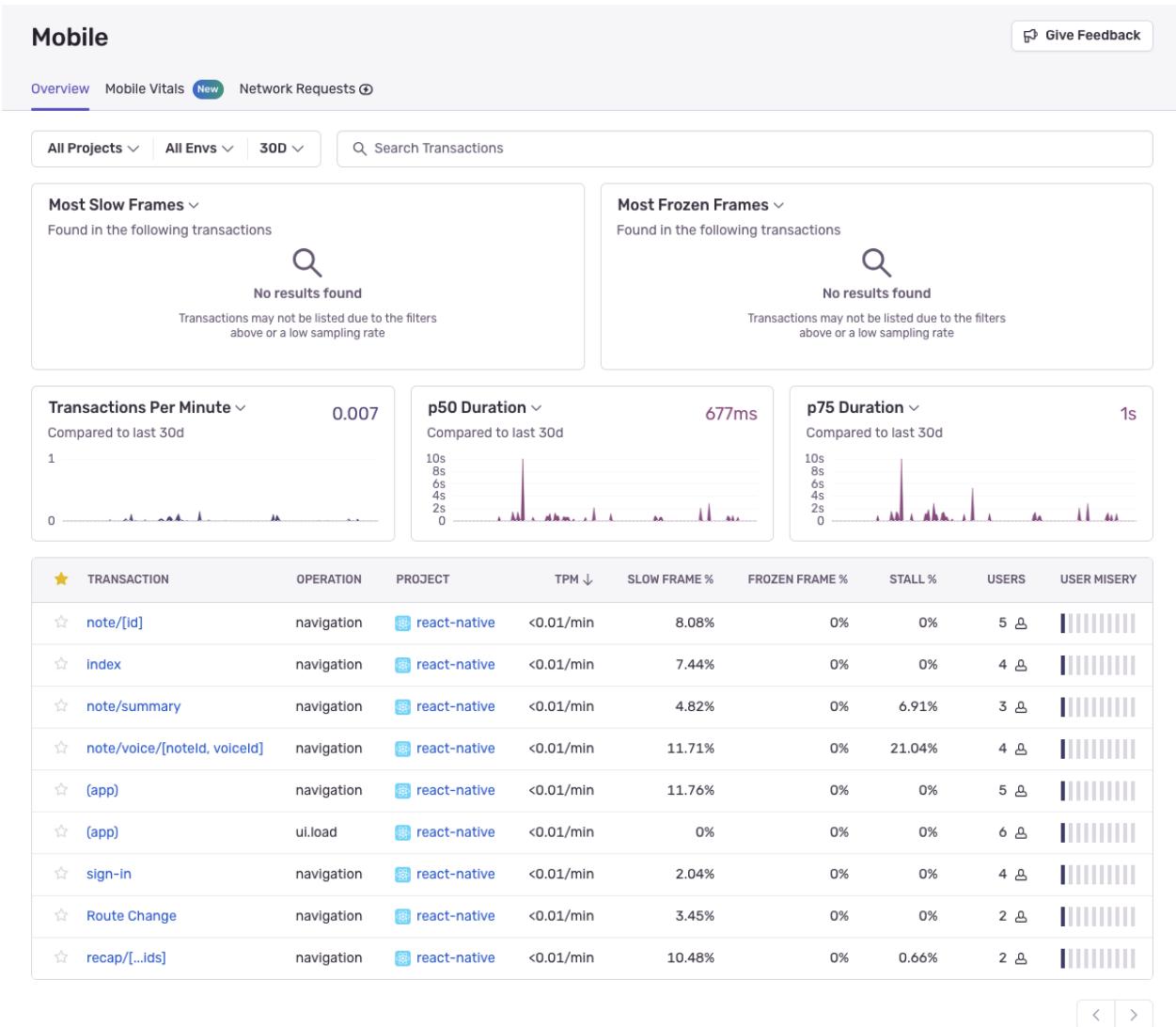
4.8.3 Závěr k testování aplikace

Testování aplikace SmartNotes probíhalo v reálném prostředí softwarové společnosti s cílem ověřit funkčnost, použitelnost a přínos hlavních funkcí aplikace při každodenním pracovním nasazení. Do testování se zapojilo celkem pět uživatelů, kteří aplikaci využívali při týmových schůzkách, jednáních s externími dodavateli i pro osobní účely. Aplikace byla nejčastěji používána v českém, slovenském a anglickém jazyce.

Během testování bylo vytvořeno 16 poznámek, jejichž celkový objem činil 132 773 znaků. Uživatelé nahráli přes 4 hodiny hlasových záznamů, které byly následně analyzovány a přepsány do textové podoby. Funkce summarizace byla využita 21krát. Poznámky byly rozšířeny o 12 pdf dokumentů. Uživatelé dále provedli 56 vyhledávacích dotazů prostřednictvím chytrého vyhledávání a ve výsledku bylo 18 poznámek nebo summarizací sdíleno v PDF formátu skrze chatovací aplikace.

Analýza teplotních map a uživatelských relací ukázala vysokou míru interakce v klíčových částech rozhraní, zejména v poznámkovém editoru, vyhledávání a při práci se summarizací. Zjištěné interakční vzorce potvrzily intuitivnost většiny funkcí, avšak odhalily také konkrétní nedostatky.

Z pohledu výkonu se aplikace ukázala jako stabilní. Výkonnostní monitoring přes nástroj Sentry odhalil pouze jeden závažný incident na straně serveru, jinak byla aplikace spolehlivá. Mírně vyšší



Obrázek 4.11: Sentry - Mobilní Přehled

odezvy byly zaznamenány u funkcí využívajících externí API (např. summarizace), což lze v budoucnu optimalizovat.

Ačkoli byla testovací skupina relativně malá, získané poznatky naznačují, že aplikace Smart-Notes má potenciál zefektivnit práci s informacemi. Testovaní uživatelé ocenili zejména funkce automatického přepisu hlasových záznamů, summarizace obsahu, chytré vyhledávání a jednoduchost aplikace. Funkce usnadnily zpětnou orientaci v obsahu schůzek a komunikaci. Zpětná vazba navíc ukázala, že členové testovací skupiny by aplikaci rádi využívali i pro osobní účely, což potvrzuje její univerzálnost a praktickou použitelnost.

Kapitola 5

Závěr

Cílem této diplomové práce bylo navrhnut a implementovat chytrou poznámkovou aplikaci Smart-Notes pro iOS, která využívá moderní technologie, včetně nástrojů umělé inteligence, ke správě textových, hlasových a PDF poznámek. Výsledná aplikace integruje funkce jako převod řeči na text, sémantické vyhledávání a automatickou summarizaci obsahu, čímž překračuje běžné schopnosti tradičních poznámkových nástrojů.

V teoretické části práce byly popsány klíčové technologie a principy, na nichž je aplikace postavena, nejnovější koncepty vývoje moderní aplikace ve frameworku React Native s frameworkem Expo, knihovnu LangChain. Dále byly představeny možnosti správy dat aplikace, koncepty integrace velkých jazykových modelů a úvod do monitoringu aplikací.

Praktická část práce se věnovala návrhu a vývoji samotné aplikace, od analýzy uživatelských požadavků, přes návrh architektury, implementaci jednotlivých modulů až po testování a vyhodnocení funkčnosti. Výsledný prototyp potvrdil, že začlenění prvků umělé inteligence do mobilní poznámkové aplikace výrazně přispívá k efektivitě práce s obsahem a pozitivně ovlivňuje UX.

Oproti existujícím řešením, jako jsou Apple Notes, Obsidian nebo Microsoft OneNote, přináší SmartNotes spojení jednoduchého uživatelského rozhraní se základními efektivními nástroji pro zpracování přirozeného jazyka. Aplikace je připravena pro další rozšiřování a případné nasazení do App Store.

Tato práce splnila stanovené cíle a zároveň otevřela prostor pro další rozvoj, například v oblasti personalizace poznámek, offline AI funkcionalit či změny přístupu k ukládání a správě dat. SmartNotes tak představuje příklad moderní poznámkové aplikace, která reaguje na aktuální technologické trendy a potřeby uživatelů v digitální éře.

Literatura

1. CECI, Laura. *Mobile app usage - statistics & facts* [online]. 2025. [cit. 2025-02-13]. Dostupné z: <https://www.statista.com/topics/1002/mobile-app-usage/%5C#topicOverview>.
2. KHAN, Imran; HOLLEBEEK, Linda D.; FATMA, Mobin; ISLAM, Jamid Ul; RATHER, Raouf Ahmad; SHAHID, Shadma; SIGURDSSON, Valdimar. Mobile app vs. desktop browser platforms: the relationships among customer engagement, experience, relationship quality and loyalty intention. *Journal of Marketing Management* [online]. 2023-02-12, roč. 39, č. 3-4, s. 275–297 [cit. 2025-02-22]. ISSN 0267-257X. Dostupné z DOI: 10.1080/0267257X.2022.2106290.
3. *Number of mobile app downloads worldwide from 2019 to 2027, by segment* [online]. 2025. [cit. 2025-02-15]. Dostupné z: <https://www.statista.com/forecasts/1262881/mobile-app-download-worldwide-by-segment>.
4. WESSEL, Michael; ADAM, Martin; MAJCHRZAK, Alexander Benlian Ann; THIES, Ferdinand. *Generative AI and its Transformative Value for Digital Platforms* [online]. [cit. 2025-02-22]. Dostupné z: https://www.jmis-web.org/cfps/JMIS_SI_Cfp_Generative_AI.pdf.
5. MOSCONI, Nicolás. *Native vs Cross-platform Development: Pros and Cons* [online]. 2025. [cit. 2025-02-13]. Dostupné z: <https://www.devlane.com/blog/cross-platform-apps-vs-native-apps-pros-cons>.
6. *Tag Trends* [online]. 2025. [cit. 2025-02-13]. Dostupné z: <https://trends.stackoverflow.co/?tags=react-native,swiftui,android-studio,flutter>.
7. *The 6 best note-taking apps in 2025* [online]. 2025. [cit. 2025-02-15]. Dostupné z: <https://zapier.com/blog/best-note-taking-apps/>.
8. *IOS 18: Recording Audio With Transcripts in Notes* [online]. 2024. [cit. 2025-02-15]. Dostupné z: <https://www.macrumors.com/how-to/ios-record-audio-transcribe-notes-app/>.
9. SOMANATHAN, Sudarshan. *10 Best Note-Taking Apps in 2025 (Free & Paid)* [online]. 2025. [cit. 2025-02-15]. Dostupné z: <https://clickup.com/blog/note-taking-apps/>.
10. *The 20 Best Note-taking Apps of 2025* [online]. 2025. [cit. 2025-02-15]. Dostupné z: <https://www.notta.ai/en/blog/best-note-taking-apps>.

11. *A Comprehensive Overview of Large Language Models* [online]. 2025. [cit. 2025-02-15]. Dostupné z: <https://arxiv.org/abs/2307.06435>.
12. XU, Chen; YE, Rong; DONG, Qianqian; ZHAO, Chengqi; KO, Tom; XIAO, Tong; ZHU, Jingbo. Recent Advances in Direct Speech-to-text Translation. *Arxiv* [online]. 2023 [cit. 2025-02-20]. Dostupné z DOI: [10.48550/arXiv.2303.00747](https://doi.org/10.48550/arXiv.2303.00747).
13. BAIN, Max; HUH, Jaesung; HAN, Tengda; ZISSERMAN, Andrew. *WhisperX: Time-Accurate Speech Transcription of Long-Form Audio*. 2023. Dostupné z arXiv: [2303.00747](https://arxiv.org/abs/2303.00747) [cs.SD].
14. WU, Jian; GAUR, Yashesh; CHEN, Zhuo; ZHOU, Long; ZHU, Yimeng; WANG, Tianrui; LI, Jinyu; LIU, Shujie; REN, Bo; LIU, Linquan; WU, Yu. On Decoder-Only Architecture For Speech-to-Text and Large Language Model Integration. *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* [online]. 2023-12-16, s. 1–8 [cit. 2025-02-22]. ISBN 979-8-3503-0689-7. Dostupné z DOI: [10.1109/ASRU57964.2023.10389705](https://doi.org/10.1109/ASRU57964.2023.10389705).
15. GOLDSTEIN, Ariel; WANG, Haocheng; NIEKERKEN, Leonard; ZADA, Zaid; AUBREY, Bobbi; SHEFFER, Tom; NASTASE, Samuel A.; GAZULA, Harshvardhan; SCHAIN, Mariano; SINGH, Aditi; RAO, Aditi; CHOE, Gina; KIM, Catherine; DOYLE, Werner; FRIEDMAN, Daniel; DEVORE, Sasha; DUGAN, Patricia; HASSIDIM, Avinatan; BRENNER, Michael; MATIAS, Yossi; DEVINSKY, Orrin; FLINKER, Adeen; HASSON, Uri. Deep speech-to-text models capture the neural basis of spontaneous speech in everyday conversations. *Biorxiv* [online]. 2023 [cit. 2025-02-22]. Dostupné z DOI: [10.1101/2023.06.26.546557](https://doi.org/10.1101/2023.06.26.546557).
16. LEVINE, Sarah; HSIEH, Hsiaolin; SOUTHERTON, Emily; SILVERMAN, Rebecca. How high school students used speech-to-text as a composition tool. *Computers and Composition* [online]. 2023, roč. 68 [cit. 2025-02-22]. ISSN 87554615. Dostupné z DOI: [10.1016/j.compcom.2023.102775](https://doi.org/10.1016/j.compcom.2023.102775).
17. AWASTHI, Ishitva; GUPTA, Kuntal; BHOGAL, Prabjot Singh; ANAND, Sahejpreet Singh; SONI, Piyush Kumar. Natural Language Processing (NLP) based Text Summarization - A Survey. *2021 6th International Conference on Inventive Computation Technologies (ICICT)* [online]. 2021-1-20, s. 1310–1317 [cit. 2025-02-23]. ISBN 978-1-7281-8501-9. Dostupné z DOI: [10.1109/ICICT50816.2021.9358703](https://doi.org/10.1109/ICICT50816.2021.9358703).
18. KUMAR, Sandeep; SOLANKI, Arun. An abstractive text summarization technique using transformer model with self-attention mechanism. *Neural Computing and Applications* [online]. 2023, roč. 35, č. 25, s. 18603–18622 [cit. 2025-02-23]. ISSN 0941-0643. Dostupné z DOI: [10.1007/s00521-023-08687-7](https://doi.org/10.1007/s00521-023-08687-7).
19. DELMONTE, Rodolfo; MARCHESEINI, Giulia; BUSETTO, Nicolò. How ChatGPT's Hallucinations (Compared to Gemini's) Impact Text Summarization with Literary Text. *Preprints* [online]. 2025 [cit. 2025-02-23]. Dostupné z DOI: [10.20944/preprints202501.2321.v1](https://doi.org/10.20944/preprints202501.2321.v1).
20. *Expo docs* [online]. 2025. [cit. 2025-02-26]. Dostupné z: <https://docs.expo.dev/>.

21. *React Native* [online]. 2025. [cit. 2025-02-26]. Dostupné z: <https://reactnative.dev/>.
22. *EAS* [online]. 2025. [cit. 2025-04-09]. Dostupné z: <https://expo.dev/eas>.
23. *EAS Build* [online]. 2024. [cit. 2025-04-09]. Dostupné z: <https://docs.expo.dev/build/introduction/>.
24. *EAS Update* [online]. 2025. [cit. 2025-02-26]. Dostupné z: <https://docs.expo.dev/eas-update/introduction/>.
25. *Get started with EAS Workflows* [online]. 2025. [cit. 2025-04-09]. Dostupné z: <https://docs.expo.dev/eas/workflows/get-started/>.
26. *What is LLM (Large Language Model)?* [online]. 2025. [cit. 2025-04-07]. Dostupné z: <https://aws.amazon.com/what-is/large-language-model/>.
27. RAIAAN, Mohaimenul Azam Khan; MUKTA, Md. Saddam Hossain; FATEMA, Kaniz; FA-HAD, Nur Mohammad; SAKIB, Sadman; MIM, Most Marufatul Jannat; AHMAD, Jubaer; ALI, Mohammed Eunus; AZAM, Sami. A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges. *IEEE Access* [online]. 2024, roč. 12, s. 26839–26874 [cit. 2025-04-07]. ISSN 2169-3536. Dostupné z DOI: 10.1109/ACCESS.2024.3365742.
28. XU, Jiajun; LI, Zhiyuan; CHEN, Wei; WANG, Qun; GAO, Xin; CAI, Qi; LING, Ziyuan. On-Device Language Models: A Comprehensive Review. *Arxiv* [online]. 2024 [cit. 2025-04-10]. Dostupné z DOI: 10.48550/arXiv.2409.00088.
29. *Introduction* [online]. 2025. [cit. 2025-04-10]. Dostupné z: <https://python.langchain.com/docs/introduction/>.
30. TOPSAKAL, Oguzhan; AKINCI, Tahir Cetin. Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast. *International Conference on Applied Engineering and Natural Sciences* [online]. 2023-07-26, roč. 1, č. 1, s. 1050–1056 [cit. 2025-04-10]. ISSN 2980-3209. Dostupné z DOI: 10.59287/icaens.1127.
31. *LangChain , LlamaIndex, or Haystack: Which Framework Suits Your LLM Needs?* [online]. 2024. [cit. 2025-04-10]. Dostupné z: <https://dkaarthick.medium.com/langchain-llamaindex-or-haystack-which-framework-suits-your-l1m-needs-7408fee8ab1e>.
32. *Haystack* [online]. 2025. [cit. 2025-04-10]. Dostupné z: <https://haystack.deepset.ai/>.
33. *Llamaindex* [online]. 2025. [cit. 2025-04-10]. Dostupné z: <https://www.llamaindex.ai/>.
34. *LlamaIndex vs LangChain vs Haystack: What Are the Differences?* [online]. 2023. [cit. 2025-04-10]. Dostupné z: <https://anakin.ai/blog/llamaindex-vs-langchain/>.
35. *Mobile App Monitoring: How To Do It Right* [online]. 2025. [cit. 2025-04-12]. Dostupné z: <https://middleware.io/blog/mobile-app-monitoring/>.

36. *What is mobile app monitoring?* [online]. 2025. [cit. 2025-04-12]. Dostupné z: <https://www.dynatrace.com/knowledge-base/mobile-app-monitoring/>.
37. *What happened in the ChatGPT data breach?* [online]. 2024. [cit. 2025-04-20]. Dostupné z: https://www.twingate.com/blog/tips/chatgpt-data-breach?utm_source=chatgpt.com.
38. *ChatGPT can leak training data, violate privacy, says Google's DeepMind* [online]. 2023. [cit. 2025-04-20]. Dostupné z: https://www.zdnet.com/article/chatgpt-can-leak-source-data-violate-privacy-says-googles-deepmind/?utm_source=chatgpt.com.
39. SCHWARTZMAN, Gregory. Exfiltration of personal information from ChatGPT via prompt injection [online]. 2024 [cit. 2025-04-20]. Dostupné z DOI: 10.48550/arXiv.2406.00199.
40. *Europe privacy policy* [online]. 2024. [cit. 2025-03-02]. Dostupné z: <https://openai.com/policies/privacy-policy/>.
41. *Dodatečné smluvní podmínky pro Gemini API* [online]. 2024. [cit. 2025-03-02]. Dostupné z: <https://ai.google.dev/gemini-api/terms?hl=cs%5C#data-use-paid>.
42. *Edit rich text* [online]. 2024. [cit. 2025-03-23]. Dostupné z: <https://docs.expo.dev/guides/editing-richtext/>.
43. *Overview of app launch times* [online]. 2025. [cit. 2025-04-05]. Dostupné z: <https://docs.newrelic.com/docs/mobile-monitoring/new-relic-mobile/get-started/introduction-app-launch-times/>.