Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

PLZEŇ, 2022 **JAN TUPÝ**

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2021/2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jan TUPÝ**
Osobní číslo:	**A19B0391P**
Studijní program:	**B0714A150005 Kybernetika a řídicí technika**
Specializace:	**Umělá inteligence a automatizace**
Téma práce:	**Hlasová asistentka při vaření**
Zadávající katedra:	**Katedra kybernetiky**

Zásady pro vypracování

1. Prostudujte problematiku hlasových dialogových systémů a seznamte se s hlasovou platformou SpeechCloud.
2. Navrhněte hlasový dialog asistující uživateli při vaření, od seznamu surovin až po průvodce během samotného vaření. Analyzujte možnosti využití veřejné databáze receptů či recepty z internetu.
3. Dialog realizujte jako webovou aplikaci (na PC či Raspberry Pi) a otestujte.

Rozsah bakalářské práce: **30 – 40 stránek A4**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná**

Seznam doporučené literatury:

Dodá vedoucí práce.

Vedoucí bakalářské práce: **Ing. Luboš Šmídl, Ph.D.**
Katedra kybernetiky

Datum zadání bakalářské práce: **15. října 2021**
Termín odevzdání bakalářské práce: **23. května 2022**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Prof. Ing. Josef Psutka, CSc.
vedoucí katedry

V Plzni dne 15. října 2021

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne 23. 05. 2022 .....

podpis

Poděkování

Chtěl bych tímto rád poděkovat panu Ing. Luboši Šmídlovi, Ph.D. za odborné vedení, trpělivost a věcné připomínky k této bakalářské práci.

Abstrakt

Tato práce se zabývá vývojem webové aplikace, která uživateli asistuje při vaření. Nejprve je zde shrnuta problematika hlasových dialogových systémů a jsou zde představy technologie, které jsou potřebné pro tvorbu webové aplikace. Dále se práce věnuje návrhu struktury, uživatelskému rozhraní, ale také návrhem samotného dialogu, který uživatele provází během vaření. Následuje analýza možností získávání receptů z internetu. Výsledkem této práce je webová aplikace provázející uživatele při vaření, a to od samotného výběru receptu přes seznam surovin až po jednotlivé instrukce z postupu.

Klíčová slova

hlasová asistentka při vaření, hlasový dialogový systém, SpeechCloud, extrakce dat z webu, grafické uživatelské rozhraní

Abstract

The topic of this thesis is development of a web application that assists the user with cooking. First, voice dialog systems are summarized and the technologies needed to create a web application are introduced. Furthermore, the thesis describes structural design, user interface, but also the design of the dialogue itself, which helps the user during cooking. After that follows an analysis of the possibilities of obtaining recipes from the internet. The result of this thesis is a web application that helps the user with cooking, includnig the selection of the recipe, going over the list of ingredients as well as individual recipe steps.

Key words

voice assistant for cooking, voice dialogue system, SpeechCloud, web scraping, graphical user interface

Obsah

1 Úvod

Komunikace pomocí řeči je jeden z nejstarších a nejpřirozenějších způsobů, jak si mezi sebou lidé předávají informace. Zavedení tohoto způsobu interakce mezi člověkem a strojem je ovšem poměrně nová záležitost, která začala být zkoumána ve druhé polovině minulého století [1]. Systémy schopné komunikovat s člověkem pomocí řeči jsou označované jako hlasové dialogové systémy (HDS) a jejich hlavním rysem je právě schopnost přijmout a zpracovat lidskou řeč.

Aplikace HDS umožňuje lidem automatizovat a zjednodušovat úlohy, které by jinak museli obstarávat sami, což je často nepraktické a ekonomicky nevýhodné. HDS jsou typicky nasazovány v situacích, kde je jiná forma interakce s uživatelem nemožná nebo nepraktická. Příkladem může být řízení dopravního prostředku, kdy má řidič zaměstnané oči a ruce, takže nemůže se systémem interagovat pomocí textu. Další významnou oblastí, kde jsou HDS využívané, je pomoc lidem s omezenými zrakovými schopnostmi, pro které by předávání a získávání informací pomocí vizuálních rozhraní bylo značně nepraktické nebo zcela nemožné.

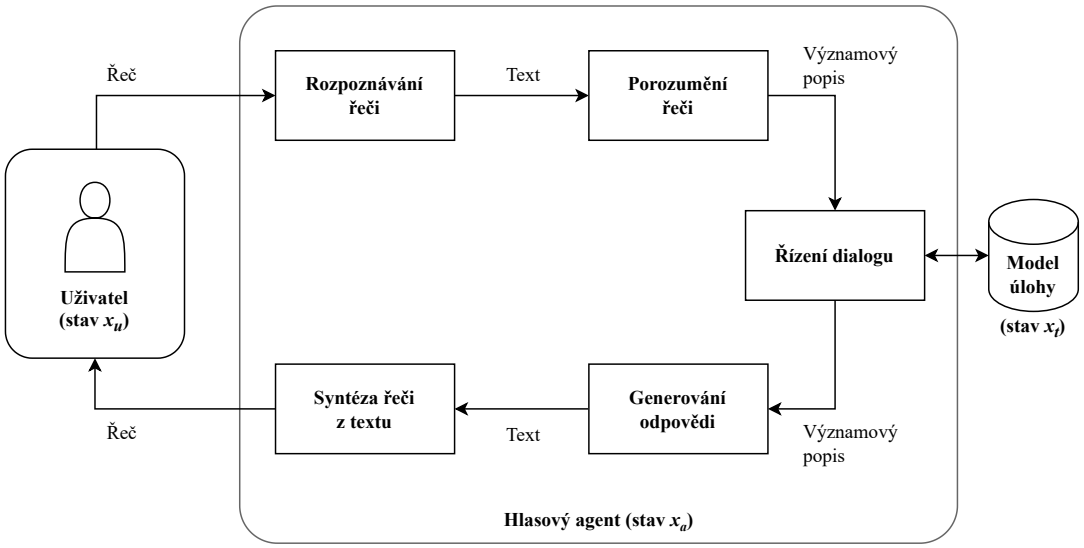
Tématem této práce je využití HDS jako asistentky při vaření. Hlavní motivací pro tuto práci je situace, kdy má uživatel při vaření často zaměstnané nebo jen špinavé ruce a listování v kuchařce nebo klikání na displej či klávesnici je tak problematické. Hlasová asistentka by mohla nabídnout možnost automatického hledání receptů z internetu, předčítání jednotlivých potřebných surovin a následně i samotného postupu vaření, což by spolu se základními hlasovými příkazy řešilo výše zmíněné problémy. Uživatel by tak mohl snadno postupovat podle receptu, aniž by se musel stále vracet ke kuchařce či telefonu a před každým dotykem si musel umýt ruce.

Cílem této práce je navržení HDS, který bude uživateli asistovat během vaření. Systém uživateli umožní vyhledat recept, poskytnout seznam potřebných surovin, informace o výsledném množství jídla a době přípravy a samozřejmě diktovat uživateli samotný postup. Systém bude realizovaný formou webové aplikace, což nabízí uživateli možnost výběru, zda chce systém ovládat pomocí klávesnice a myši, dotykové obrazovky a nebo hlasových příkazů. Možnosti ovládání pomocí klávesnice, myši nebo dotykové obrazovky jsou vhodné například pro výběr receptu, kdy uživatel ještě nezačal vařit a tak by použití těchto zařízení nemělo představovat problém. Grafický výstup je pak vhodný pro zobrazení fotografie výsledného pokrmu, který bývá u téměř každého receptu.

2 Hlasové dialogové systémy

Hlasové dialogové systémy (HDS) přinášejí nové rozhraní, které lidem umožňuje přirozenou a efektivní výměnu informací mezi uživatelem a strojem pomocí řeči [2]. V hlasových dialogových systémech probíhá hlasový dialog, který má zpravidla nějaký cíl. V HDS je několik modulů, které jsou potřeba k realizaci hlasového dialogu.

Automatické rozpoznávání řeči (ASR) slouží k převedení mluvené řeči do slovní podoby. Tato podoba je s využitím **porozumění řeči** (SLU) převedena do strojové reprezentace textu. Pomocí této reprezentace řídí **dialogový manažer** (DM) samotný dialog a popřípadě i řešenou úlohu a generuje významový popis (akci), který je **generátorem odpovědi** (NLG) převeden na textovou podobu, která je připravena k **syntéze řeči** (TTS). Schéma HDS je zobrazeno na Obrázku 1. [3]



Obrázek 1: Schéma hlasového dialogového systému (převzato a upraveno) [3]

Stavy hlasového dialogového systému

Celkový stav hlasového dialogového systému (x_{hds}) vznikne složením vnitřních stavů uživatele, hlasového agenta a řešené úlohy. Lze vyjádřit pomocí zápisu:

$$\text{stav HDS} = [\text{stav uživatele, stav agenta, stav úlohy}], \tag{1}$$

nebo symbolicky:

$$x_{hds} = [x_u,\ x_a,\ x_t]. \tag{2}$$

Stav HDS reprezentuje všechnu informaci potřebnou k úspěšnému pokračování hlasového dialogu.

Stav uživatele je založen především na cíli uživatele a na jeho samotném účelu zahájení interakce. Dalšími důležitými faktory je mentální rozložení uživatele a jeho znalosti a zkušenosti, díky kterým mu může být dialog přizpůsobován na míru.

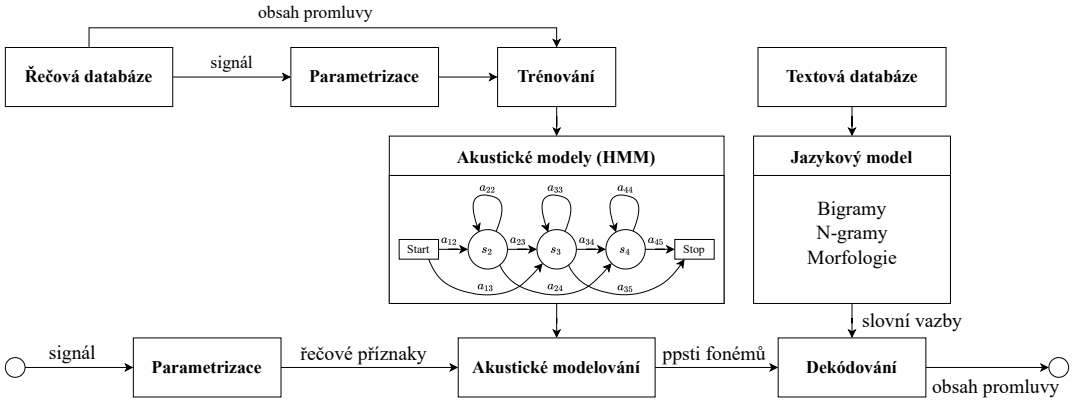
Stav agenta zahrnuje veškerou komunikaci od začátku hlasového dialogu. Aktualizace stavu se provádí v modulu řízení dialogu vždy po nově rozpoznané promluvě. Podle historie rozpoznaných promluv lze u některých HDS parametrizovat jednotlivé moduly (ASR, SLU, NLG, TTS).

Stav úlohy reprezentuje řešenou úlohu. Kvůli své složitosti není úloha modelována přímo v řízení dialogu, ale jako samostatný model, a tudíž lze pozorovat pouze nepřímo pomocí řízení a výstupu úlohy.

2.1 Automatické rozpoznání řeči

Automatické rozpoznávání řeči je převod řečového signálu na text [4]. Rozpoznávání řeči je velmi obtížné kvůli velké variabilitě řečového signálu. Stejnou promluvu vysloví každý řečník jinak, dokonce i stejný řečník vysloví tutéž promluvu pokaždé odlišně. Navíc v řečovém signálu se projeví jakákoliv změna prostředí (rušivé zvuky, akustika místnosti) nebo přenosového kanálu (řeč přes telefon, změna mikrofonu). [5]

Pro jednoduché **rozpoznávání izolovaných slov** se používají metody souhrnně nazývané *srovnání se vzorem*. Pro každé slovo musí být v databázi uložený referenční vzor. Při porovnávání se vstupní signál porovnává postupně se všemi referenčními vzory a výsledkem rozpoznávání je slovo, které je vstupní promluvě nejvíce podobné. Pro rozpoznávání **souvislé řeči** je zapotřebí použít **statistický přístup**, který je založen na statistických modelech. Grafické znázornění schéma na Obrázku 2.



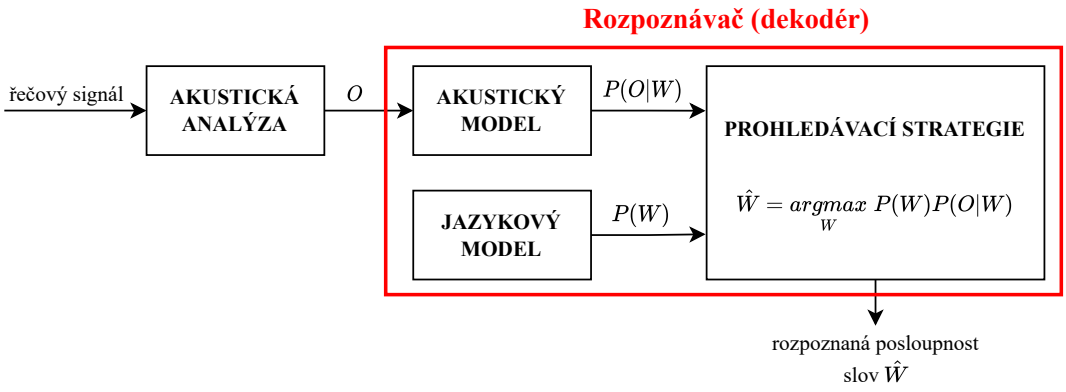
Obrázek 2: Blokové schéma rozpoznávání řeči (převzato a upraveno) [6]

2.1.1 Parametrizace

Cílem parametrizace řeči je extrakce zásadních informací z řečového signálu pro účely rozpoznávání řeči, neboli snaha odstranit co možná nejvíce nepodstatných informací z řečového signálu. Využitím principu stacionarity, lze signál zpracovávat po kouskách (≈10-30 ms). Stejným hláskám (fonémům) budou odpovídat stejné **parametry**. Podle Shannonova vzorkovacího teorému musí být vzorkování 2x rychlejší, než je nejrychleji se měnící frekvence v signálu. Výsledkem parametrizace je posloupnost příznakových vektorů O. Mezi nejčastější metody parametrizace patří parametrizace typu LPC, PLP a MFCC. [7]

2.1.2 Statistický přístup

Řečníkovo promluvu lze zapsat pomocí vektoru $W = [w_1, w_2, ..., w_N]$, který reprezentuje původní posloupnost slov. Pomocí akustické analýzy s použitím metody pro parametrizaci řeči vznikne posloupnost příznakových vektorů $O = [o_1, o_2, ..., o_M]$. Hlavním úkolem rozpoznávače (dekodéru) je zjistit nejlepší odhad $\hat{W}$ původní posloupnosti slov W. Jelikož dekodér pracuje na statistickém principu, určí jako výsledek takovou posloupnost slov $\hat{W}$, která je pro daný signál nejpravděpodobnější. [5]



Obrázek 3: Rozpoznávání řeči - dekodér (převzato a upraveno) [5]

Matematicky to lze vyjádřit ve tvaru:

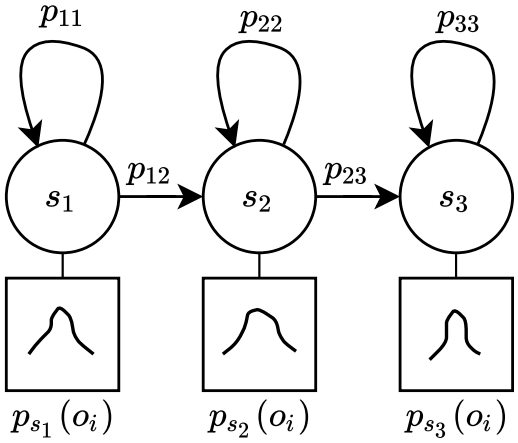
$$\hat{W} = \arg\max_{W} P(W|O) = \arg\max_{W} \frac{P(O|W)\,P(W)}{P(O)} = \arg\max_{W} P(O|W)\,P(W) \quad (3)$$

Pro zjednodušení lze vynechat pravděpodobnost $P(O)$, která sice ovlivňuje celkovou hodnotu pravděpodobnosti, ale nikoliv výběr maxima. Pro zjištění zbylých pravděpodobností je zapotřebí akustický model - P(O|W) a jazykový model - P(W).

(1) Akustický model

Akustický model zahrnuje nejen hlasový trakt, tj. proces přeměny myšlenek na akustické vlny, ale také technické parametry mikrofonu a akustické vlastnosti prostředí, ve kterém se pronesená řeč šíří. Z tohoto důvodu se proto často stává, že i při malé změně jednoho z uvedených parametrů akustického kanálu je nutné vyměnit celý akustický model. [5]

Od počátku 80. let se pro modelování začaly používat *skryté Markovovy modely (HMM)* [8], které využívají model hlásky (Obrázek 4).



Obrázek 4: Model hlásky (převzato a upraveno) [5]

Za předpokladu, že hlasové ústrojí vyslovuje nějakou hlásku, nachází se přitom v jednom ze tří stavů (s_1, s_2, s_3) a produkuje přitom postupně zvuky, které se po daném zpracování transformují do posloupnosti příznaků $O = [o_1, o_2, ..., o_i, ..., o_M]$. Jednotlivé zvuky jsou v každém stavu $p_{sj}(o_i)$ produkovány s jinou pravděpodobností, protože nejsou jednoznačně určeny. To samé platí i pro hlasové ústrojí, které může zůstat ve stejném stavu p_{jj} nebo přejít do stavu následujícího p_{jk}. Modely slov se vytvářejí řetězením modelů hlásek. Lze vytvořit všechna slova, která se nachází v předem definovaném slovníku, který obsahuje i fonetickou transkripci, tj. informace o tom, z jakých hlásek se dané slovo skládá. Řetězením modelů slov vzniknou věty. [5]

Parametry modelu se nastavují automaticky na základě trénovacích dat, které obsahují nahrané promluvy a jejich co nejpřesnější popis. Grafická znázornění na Obrázku 2. V současnosti se místo HMM používají spíše *hluboké neuronové sítě (DNN)* [9, 10] nebo kombinace obou přístupů HMM-DNN.

(2) Jazykový model

Jazykový model určuje pravděpodobnost $P(W)$, neboli takovou pravděpodobnost, že řečník vysloví danou posloupnost slov W. Parametry modelu se nastavují automaticky pomocí textových databází (korpusů), ve kterých se zjišťují počty jednotlivých slovních n-tic.

Nejpřesnější jazykový model by měl být založený na **všech předchozích slovech**. To znamená, že pravděpodobnost každého slova v jakkoliv dlouhé promluvě by měla být vyjádřena v závislosti na všech předchozích slovech. V případě, že posloupnost slov W obsahuje K slov, lze její pravděpodobnost vyjádřit podle tzv. *řetězového pravidla*. [5] Matematický zápis je ve tvaru:

$$\begin{aligned} P(W) &= P(w_1^K) = P(w_1, w_2, \ldots, w_K) \\ &= P(w_1)P(w_2|w_1^1)P(w_3|w_1^2) \ \ldots \ P(w_K|w_1^{K-1}) \\ &= \prod_{i=1}^{K} P(w_i|w_1^{i-1}) \end{aligned} \tag{4}$$

Nicméně takovýto model obsahuje extrémně mnoho parametrů, které je velmi složité automaticky určit z trénovacích dat. Proto se v praxi používá aproximace modelu, která předpokládá, že pravděpodobnost slova je závislá pouze na $n-1$ předchozích slovech $\rightarrow$ tzv. **N-gramový model**. Z předchozí rovnice (4) lze aproximovat podmíněnou pravděpodobnost slova w_i, která závisí na všech předchozích slovech, na závislost pouze na $n-1$ předchozích slovech. Matematický zápis je ve tvaru:

$$P(w_i|w_1^{i-1}) \approx P(w_i|w_{i-n+1}^{i-1}) \tag{5}$$

A pravděpodobnost celé posloupnosti slov W lze aproximovat do tvaru:

$$P(W) = P(w_1^K) \approx \prod_{i=1}^{K} P(w_i|w_{i-n+1}^{i-1}) \tag{6}$$

Pro N-gramový jazykový model se nejčastěji používá:

- **unigram** $n = 1$ $\rightarrow$ nezávisí na předchozích slovech $P(w_i)$
- **bigram** $n = 2$ $\rightarrow$ závisí na jednom předchozím slově $P(w_i|w_{i-1})$
- **trigram** $n = 3$ $\rightarrow$ závisí na dvou předchozích slovech $P(w_i|w_{i-2}, w_{i-1})$

Kromě N-gramového modelu existují i jiné způsoby trénování. V případě, že nejsou k dispozici žádná trénovací data nebo k řešení úlohy stačí využít malý slovník, lze využít přístup založený na tzv. *formálních gramatikách*. V opačném případě, kdy je k dispozice velká množina trénovacích dat, se využívají *neuronové sítě*. [5]

2.2 Porozumění řeči

Modul porozumění řeči poskytuje rozhraní mezi přirozeným jazykem a strojovým algoritmem řízení, jehož hlavním úkolem je převod řeči na významový popis, který je potřeba pro následné provedení požadované akce. Pro zvýšení přesnosti porozumění lze využít více alternativních hypotéz rozpoznávání řeči z tzv. n-best hypotéz, které umožní generovat více alternativních významových hypotéz. Pro převod lexikální podoby do strojové reprezentace se využívá několik přístupů. [3]

2.2.1 Znalostní přístup

Znalostní přístup je založený na bezkontextové gramatice [11] a na syntaktické analýze (*parsing*), což je proces, který rozhoduje, zda dané slovo patří do jazyka generované gramatikou. Gramatika obvykle pracuje se syntaktickými kategoriemi (jako jsou například podstatná jména, přídavná jména, slovesa, apod.) a stanovuje, jak mohou být tyto kategorie vzájemně propojeny. Získání gramatických pravidel pro extrakci sémantické znalosti je poměrně nákladný proces, který vyžaduje účast lidského experta v daném oboru.

2.2.2 Statistický přístup

Vhodná forma gramatiky může být generovaná automaticky pomocí strojového učení a uchována ve formě parametrů modelu. Trénování probíhá na anotovaném korpusu tak, že pro velké množství přepsaných řečových promluv se odhadnou parametry modelu. Hlavní výhoda oproti znalostnímu přístupu je, že metody odhadování parametrů jsou nezávislé na úloze a jazyce. Díky tomu je tato technika mnohem pružnější a lépe přenositelná na jiné tématické úlohy. [11]

2.2.3 Kombinace obou přístupů

Z důvodu účinnosti jsou předchozí přístupy (2.2.1, 2.2.2) kombinovány.

Lokální význam mají sémantické entity (čas, datum, jména, položka z databáze, pozdrav). Využívá se zde algoritmus založený na celočíselném programování.

Globální význam mají sémantické koncepty (odjezdy, schůzky, souhlas, požadavek, odpověď). Využívají se zde metody strojového učení a je to možnost, jak podpořit příznaky z lokálního významu. [12]

2.3 Řízení dialogu

Za řízení hlasového dialogu je odpovědný **dialogový manažer**, který na základě stavu dialogu (popisovaný v kapitole 2) vybírá vhodnou strategii a odpovídající akce, které zajišťují cílové chování systému. Úlohou dialogového manažera je tedy organizace vazby mezi jednotlivými moduly a zajišťování komunikace HDS s uživatelem a řešenou úlohou/externí aplikací (řízení robota, databáze, apod.). Řízení dialogu má různé způsobu vedení, strategie a také různou strukturu dialogu. [3, 11]

2.3.1 Vedení dialogu

Nejpřirozenější způsob vedení dialogu se zaměřuje na řešení případných nejasností v promluvě, které můžou být způsobené uživatelem, systémem nebo komunikačním kanálem a prostředím. Dialogový manažer může tyto nejasnosti odstranit prostřednictvím správně zformulovaného dotazu uživateli, kde může žádat doplňující informace nebo potvrzení k již proběhlé promluvě. Existují dva hlavní přístupy pro potvrzování informací [11]:

- **Explicitní ověřování** - potvrzení obsahu promluvy formou odpovědi *ano/ne*
- **Implicitní ověřování** - potvrzení obsahu je včleněno vždy do následující otázky systému, uživatel tak může stále opravovat opakovanou hodnotu

2.3.2 Strategie řízení dialogu

Strategie specifikuje akci pro další stav. V každém stavu úlohy je podstatné vyřešit dílčí cíle dialogové úlohy. Dílčí cíle se většinou zaměřují na některou z následujících úloh [11]:

- **potvrzení** → zjištění správnosti rozpoznané informace
- **zotavení z chyby** → náprava chyby po špatném porozumění systému
- **opětovná pobídka** → postup po neobdržené očekávané informaci
- **dokončení** → zjištění chybějící informace od uživatele
- **omezení** → redukce rozsahu požadavku
- **uvolnění** → zvětšení rozsahu požadavku
- **zjednoznačnění** → řešení nekonzistentního vstupu od uživatele
- **pozdrav/zakončení** → zjištění začátku a konce interakce

Strategie řízení se může dělit podle iniciativy. Existují 3 základní typy - *iniciativa systému*, *iniciativa uživatele* a *smíšená iniciativa*.

V dialogu s **iniciativou systému** zajišťuje systém řízení celého dialogu, pokládá otázky a předkládá způsob řešení úlohy. Při **iniciativě uživatele** je hlasový dialog založený na příkazech. Uživatel řídí přímo hlasového agenta popřípadě i řešenou úlohu. **Smíšená iniciativa** je kombinací předchozích možností, kde v libovolné fázi dialogu může uživatel nebo systém převzít iniciativu. [3]

Další dělení řízení může být podle struktury dialogu. První možností je tzv. **turn based dialog**, kde se uživatel a hlasový agent střídají až po dokončení promluvy, tj. nemají možnost vzájemného přerušení. Hlasový agent čeká na úplný vstup uživatele a následně vygeneruje odpověď a předloží ji uživateli. Tento postup (dvojice dotaz a odpověď) je označován jako *dialogová obrátka (turn)*. Stav hlasového agenta se mění v diskrétních okamžicích. Druhou možností je tzv. **inkrementální dialog**, kde na rozdíl od první možnosti hlasový agent zpracovává vstup uživatele průběžně a v případě jakékoliv nejednoznačnosti může uživatele okamžitě přerušit neboli skočit mu do řeči (tzv. *barge-in*). Stav hlasového agenta se zde mění průběžně. [3]

2.4 Generování odpovědi

Generování odpovědi je převod významové reprezentace z řízení dialogu na textovou podobu, které se využívá k syntéze řeči. Jednodušší přístup využívá pro generování řeči vyplňování šablon. Tento přístup je obtížnější pro modifikaci a lokalizaci do jiného jazyka. Pro složitější generování odpovědi se využívá statistický model, který se trénuje pomocí strojového učení na velkém korpusu ručně psaných textů. [3]

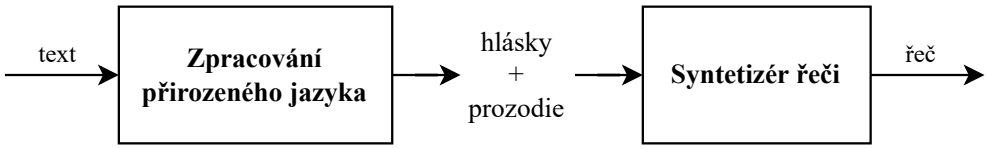
Generovaný text by měl působit přirozeně a vytvářené texty by měly být správně podle pravidel, morfologie (skloňování, časovaní) a pravopisu. Pro přiblížení k reálným odpovědím lze aplikovat možnost alternativních reprezentací odpovědí. To znamená, že generované odpovědi mohou mít ve stejných situacích stejný význam, ale jinou lexikální podobu.

Alternativním přístupem generování řeči je použití tzv. *end-to-end* systémů, které využívají strojové učení (často LSTM [1]) pro trénování algoritmu na velkém množství trénovacích dat. Tento přístup je více popsán v článcích [13, 14]. V některých případech může být generování odpovědi prováděno přímo v řízení dialogu, ale naopak i modul NLG může provádět akce spojené spíše s modulem TTS (např. prozódie).

[1] LSTM *(Long short-term memory)* je umělá neuronová síť používaná v oblastech umělé inteligence a hlubokého učení.

2.5 Syntéza řeči

Syntéza řeči je proces umělého vytváření řeči. Systém syntézy řeči (TTS) umožňuje převod libovolné textové realizace odpovědi hlasového agenta na řeč. Systém je rozdělen na dvě části, kde první část je zaměřena na **zpracování přirozeného jazyka** (NLP) a druhou část tvoří **syntetizér řeči**. [15] Grafické znázornění na Obrázku 5.

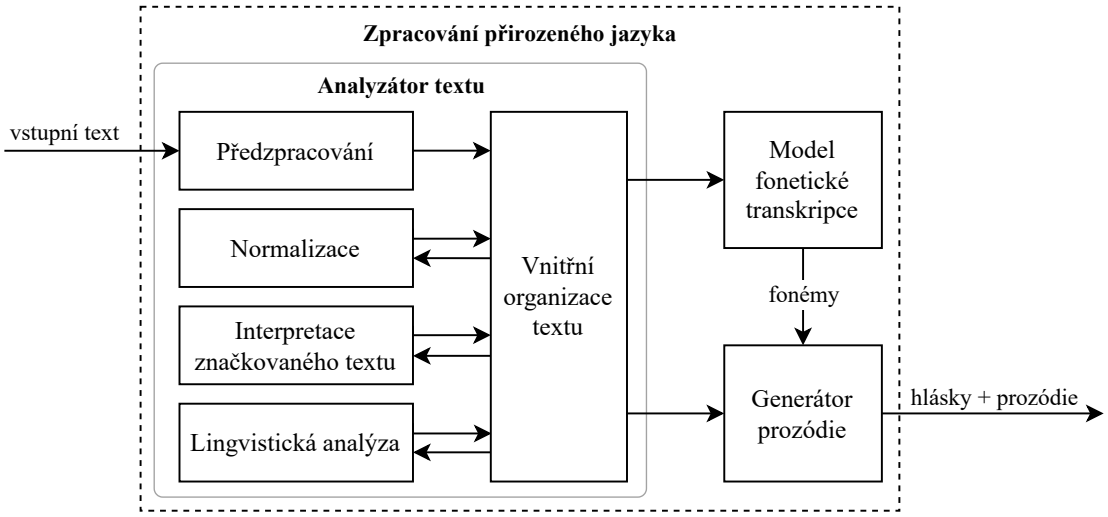


Obrázek 5: Schéma TTS systému (převzato a upraveno) [15]

U syntézy řeči se hodnotí hlavně její přirozenost a srozumitelnost, v ideálním případě by měla být syntetická řeč nerozlišitelná od řeči člověka, což ovšem nemusí být vždy výhodné.

2.5.1 Zpracování přirozeného jazyka

Zpracování přirozeného jazyka je proces převodu textu na výslovnostní podobu. Průběh samotného zpracování je znázorněno na následujícím Obrázku 6.



Obrázek 6: Schéma zpracování přirozeného jazyka (převzato a upraveno) [15]

Vstupní text nejprve musí projít analýzou, která má za úkol odstranit nejednoznačnosti a přepsat text do plné slovní formy. Text nejprve musí projí předzpracováním, kde se detekuje typ textu, filtrují se přebytečné znaky (*formátovací, bílé*) a probíhá

detekce struktury textu (tzv. *tokenizace*). Pomocí normalizace lze přepsat např. číslovky, letopočty, zkratky, symboly, atd. do plné slovní formy (tzv. *verbalizace*). Při verbalizaci je potřeba dbát i na správné skloňování, které je pro morfologicky bohaté jazyky výrazně obtížnější. Interpretace značkovaného textu má za úkol zvýraznění vybraných vlastností jako je nastavení emotivních stylů (smutek, radost, zloba) nebo vkládání expresivního prvku (nádech, pauza, zakašlání). Lingvistická analýza se skládá z více částí, kde se zkoumá např. skladbu slova, kontext okolních slov a dělení věty na slovní úseky (*fráze*). [15, 16]

Fonetická transkripce převede předzpracovaný text do fonetické formy použitím fonetických pravidel a slovníků. K nesprávnému převodu může dojít ve chvíli, kdy pro dané slovo neexistuje žádné pravidlo a navíc se nenachází ani ve slovníku (cizí slova, názvy měst, jména osob). Posledním krokem zpracování je generování prozódie, kam patří popis intonace, rychlosti, hlasitosti, přízvuku, rytmu a členění řeči. Výstupem je tedy přepsaný text do výslovnostní podoby (fonémy + prozódie). [15, 16]

2.5.2 Syntetizér řeči

Syntetizér je zařízení (program, SW) pro umělé vytváření řeči. Vstupem do syntetizéru je posloupnost hlásek, jinými slovy fonetická informace o tom, jaký význam bude mít daná řeč, a prozodická informace o tom, jak se má dané řeč vytvořit. Syntetizér řeči je jádro každého TTS systému.

2.5.3 Základní přístupy k syntéze řeči

Základní přístup k syntéze řeči je signálový přístup (*konkatenační syntéza*) a modelový přístup (*statistická parametrická syntéza*).

Konkatenační syntéza používá přímo části přirozeného jazyka tzv. řečové jednotky, které se ukládají do inventáře. Řetězením řečových jednotek pak vzniká řeč, která napodobuje řečníka z inventáře. Nejpoužívanější technika konkatenační syntézy je *syntéza výběrem jednotek (unit selection)*. U této metody výběru jednotek je důležité množství a kvalita zdrojových nahrávek a jejich pečlivá anotace. Dopad chybné anotace je rozebrán v článku [17]. Pořizováním nahrávek při perfektních akustických podmínkách (akustické studium) lze dosáhnout velmi přirozené syntetické řeči pro daný hlas a styl mluvy. Je kladen důraz na vhodný výběr každé jednotky v závislosti na kontextu. Nevýhodou této metody je problém se změnou stylu řeči nebo hlasu. [15, 18]

Statistická parametrická syntéza využívá řečové signály k natrénování statistických modelů, ze kterých se generují řečové parametry. Řeč se následně generuje z řečových parametrů pomocí *vokodéru*. Tento přístup dosahuje rozumné kvality syntetické řeči i z menšího počtu nahrávek. K natrénování modelů se dříve používaly skryté Markovské modely (HMM), ale v současné době se dává přednost spíše hlubokým neuronovým sítím (DNN). Jelikož řeč je generována ze statistických modelů, dosahuje tento přístup akusticky horší kvality zvuku (bzučení, přehlazení), ale naopak umožňuje větší flexibilitu (změna hlasu, styl řeči) změnou parametrů modelu. [15]

V posledních letech je výzkumně velmi žhavé téma hluboká neuronová síť **WaveNet**, která je sice výpočetně extrémně náročná, ale dosahuje nejlepší kvality syntézy řeči. Metoda je více rozebírána v článcích [19, 20, 21].

2.6 Vyhodnocení

Vyhodnocení celého HDS probíhá pomocí jednoznačně definovaného experimentu, u kterého se stanový cíl dialogu. Za objektivní metriky lze považovat dosažení cíle a počet obrátek (*turn*), kterých bylo potřeba k jeho dosažení. Za subjektivní hodnocení lze považovat přirozenost a plynulost dialogu. Kromě vyhodnocení celého dialogu se používá i **vyhodnocení dílčích modulů** [3]:

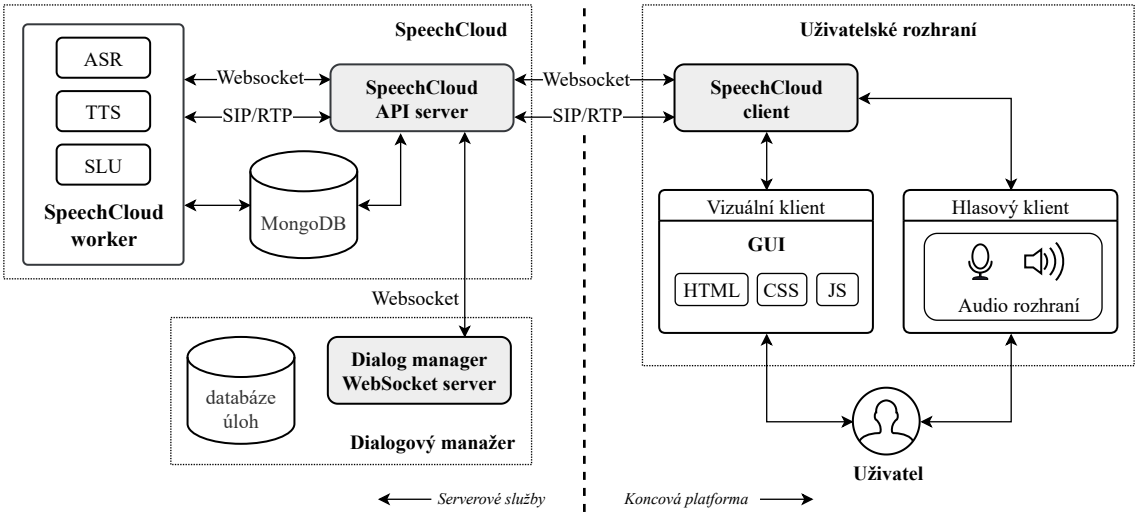
- **ASR** → přesnost rozpoznávání
- **SLU** → přesnost porozumění
- **NLG** → ověření správnosti generovaných promluv
- **TTS** → srozumitelnost a přirozenost generování řeči

3 Platforma SpeechCloud

SpeechCloud je platforma, která poskytuje jednoduché řešení pro přístup k řečovým technologiím z široké škály koncových zařízení. SpeechCloud je navržen výhradně jako interní řešení pro poskytování flexibilního rozhraní pro různé moduly související s řečí, jako je automatické rozpoznávání řeči (ASR), porozumění mluvené řeči (SLU) a syntéza řeči (TTS). [22]

3.1 Architektura

Architektura SpeechCloudu je navržena takovým způsobem, který je vhodný pro realizaci multimodálního dialogu. Pro každého klienta je vytvořena pomocí adresy URL nová relace (session). Architekturu lze rozdělit do tří částí (dialogový manažer, SpeechCloud a uživatelské rozhraní), které jsou graficky znázorněné na Obrázku 7.



Obrázek 7: Architektura SpeechCloudu (převzato a upraveno) [22]

3.1.1 Komunikace

Jednotlivé části architektury SpeechCloud jsou propojeny pomocí standardizovaných protokolů, včetně protokolu Websocket, který pro výměnu řídících zpráv používá formát JSON, pro výměnu audio signálu jsou použity protokoly SIP/RTP (Session Description Protocol/Real-time Transport Protocol) pro implementaci VoIP (Voice over Internet Protocol). [22]

3.2 SpeechCloud API server

Pro spojení webového prohlížeče s řečovými moduly jsou zde použity protokoly SIP a RTP, které přenáší zvukové pakety. [22]

Platforma SpeechCloud, pro řešení vztahů mezi klienty, jednotlivými moduly řeči a relacemi, používá databázi MongoDB. Databáze ukládá všechny události a metody volané během relace, ale také obsahuje všechny zpracované zvukové záznamy generované během relace. [22]

3.2.1 Worker

SpeechCloud API server na začátku každé relace přiřadí konkrétního SpeechCloud workera k dané relaci, který inicializuje hlasové moduly (ASR, TTS, SLU).

Výsledek rozpoznávání (ASR) je generován jako událost, kterou lze zachytit (DM, JS) pomocí posluchače událostí (tzv. *lisener*). Při použití modulu TTS je výsledný zvuk směřován přímo ke klientovi. Modul SLU využívá algoritmu SED (Semantic Entity Detection), který používá předdefinované gramatiky ve formátu SRGS (Speech Recognition Specification)[2] k popisu entit. [22]

3.3 Dialogový manažer

Dialogový manažer není přímo součástí platformy SpeechCloud, ale je implementován samostatně jako Websocket server. Dialogový manažer má na starost řízení dialogu, které lze libovolně měnit v závislosti na koncové platformě. [23]

Dialogový manažer používá volání asynchronních metod poskytovaných platformou SpeechCloud, ale také příkazy, které čekají na události SpeechCloudu. [23]

3.4 Uživatelské rozhraní

Uživatelské rozhraní je realizováno formou webové aplikace, která má vizuální výstup (vizuální klient) a lze ji ovládat pomocí klávesnice a myši, ale také za pomocí hlasu (hlasový klient). Vizuální a hlasový klient je propojen se SpeechCloud client, který funguje jako most mezi koncovou platformou (webovou aplikací) a zbytkem architektury. [22]

[2] `https://www.w3.org/TR/speech-grammar/`

4 Použité technologie

V následující části jsou popsány vybrané technologie, které jsou potřeba k vytvoření uživatelského rozhraní. Bylo vybráno pět technologií, a to HTML, CSS, SVG, DOM a WebSocket.

4.1 HTML

HTML *(Hypertext Markup Language)* je **hypertextový značkovací jazyk** určený k vytváření jednoduchých webových stránek. Původním smyslem HTML bylo vytváření, formátování a stylování zdrojových dokumentů internetových stránek. Postupem času se ovšem zmíněné stylování přesunulo pod křídla takzvaných kaskádových stylů CSS, které nabízejí mnohem více možností nejen z pohledu designu, ale i animací.

HTML používá definované značky, nazývané **tagy**. Tagům se přiřazují atributy a hodnoty, jež jednotlivým prvkům stránky přikládají určitou roli. Prostřednictvím HTML tagů se tak například určuje, kde budou odkazy a kam budou odkazovat nebo kde bude obrázek a odkud je bude prohlížeč čerpat.

Struktura HTML dokumentu:

Každý ze zmíněných HTML dokumentů by měl zachovávat alespoň základní strukturu:

- **!DOCTYPE html** – deklarace typu dokumentu
- **html** – kořenový element, který zastřešuje celý dokument
- **head** – hlavička určující metadata pro dokument, typicky obsahuje: scripty, kusy JavaScript kódu, odkaz na CSS či titulek
- **meta** – určující metadata
- **title** – titulek stránky, který se následně zobrazuje v záložce prohlížeče
- **body** – obsah dokumentu

4.2 CSS

Kaskádové styly, známé také pod zkratkou CSS *(Cascading Style Sheets)* jsou moderním jazykem umožňujícím účinné formátování stránek psaných ve značkovacích

jazycích HTML, XHTML či XML. Umožňují naprosté oddělení vzhledu dokumentu od jeho obsahu. Kaskádování názvů pochází ze zadaného schématu priority, aby se určilo, které pravidlo stylu se použije, pokud určitému prvku odpovídá více než jedno pravidlo.

4.3 SVG

SVG *(Scalable Vector Graphics)* neboli škálovatelná **vektorová grafika**, je značkovací jazyk a formát souboru, který popisuje dvojrozměrnou vektorovou grafiku pomocí XML. Formát SVG je základním otevřeným formátem pro vektorovou grafiku na webových stránkách. HTML5 umožňuje vložit kód SVG obrázku přímo do kódu HTML webové stránky. S jednotlivými prvky se dají provádět různé transformace (translace, rotace, ...) a snadno se propojí s JavaScriptem.

4.4 DOM

DOM *(Document Object Model)* je platformě i jazykově nezávislé aplikační rozhraní (API) určené pro přístup k objektům HTML, XHTML či XML dokumentům. DOM má stromovou strukturu a je vystavěn v paměti prohlížeče po načtení webové stránky. Skládá se z elementů, atributů, textů, komentářů, atd. Prvky DOMu a jejich vlastnosti, mohou být modifikovány například JavaScriptem.

4.5 WebSocket

WebSocket je počítačový komunikační protokol, který poskytuje plně duplexní komunikační kanály přes jediné TCP spojení. Protokol WebSocket umožňuje interakci mezi webovým prohlížečem (nebo jinou klientskou aplikací) a serverem, což usnadňuje přenos dat ze serveru a na server v reálném čase. Po navázání spojení může server poskytovat standardizovaný způsob odesílání obsahu klientovi, aniž by jej klient nejprve požadoval. Protokol také umožňuje předávání zpráv tam a zpět při zachování otevřeného připojení. Tímto způsobem může probíhat obousměrná konverzace mezi klientem a serverem.

5 Hlasová asistentka při vaření

Hlasová asistentka při vaření je realizována formou webové aplikace, která asistuje uživateli při vaření. Aplikace umožňuje prohlížet jednotlivé recepty hledaného jídla, nabízí seznam surovin a také uživatele provází během samotného vaření.

5.1 Návrh struktury

Navrhovaná struktura dialogu odpovídá architektuře popisované v předchozí kapitole 3 a je znázorněna na Obrázku 7. Aplikace je tedy rozdělená na tři hlavní části, a to na dialogového manažera, SpeechCloud a na uživatelské rozhraní.

Dialogový manažer je psán v programovacím jazyce Python a pro správné řízení dialogu je zde potřeba nadefinovat jeho chování. Dialogový manažer společně se SpeechCloudem je detailněji popsán v předchozí kapitole 3.

5.2 Návrh uživatelského rozhraní

Uživatelské rozhraní je realizované formou webové aplikace a je rozděleno na dvě části, na hlasové rozhraní a na grafické rozhraní. Lze tedy aplikaci ovládat pomocí dotykových displejů, myši, klávesnice, ale také pomocí hlasu.

5.2.1 Hlasové rozhraní

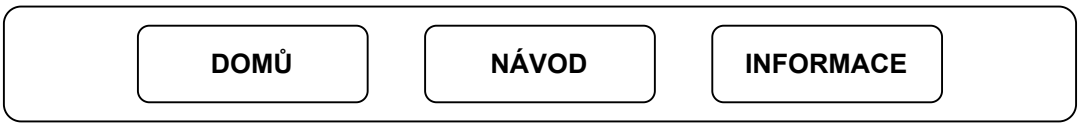
Hlasové rozhraní umožňuje ovládání hlavní části aplikace pomocí hlasu, díky kterému lze například vyhledávat libovolný recept. Po načtení receptu lze ovládat i zbytek aplikace prostřednictvím *hlasových příkazů*, které jsou popsány v následující kapitole 5.4.1.

5.2.2 Grafické rozhraní

Grafické rozhraní slouží jako vizuální výstup pro uživatele. Webová aplikace je vytvářena prostřednictvím technologií HTML, CSS (popsané v předchozí kapitole 4), které jsou doprovázeny programovacím jazykem JavaScript, který zajišťuje dynamický obsah webu. Pro zjednodušenou manipulaci s obsahem stránky a pro snazší reakce na události je zde použita i JavaScriptová knihovna jQuery[3].

[3] `https://jquery.com/`

Webová aplikace se skládá ze tří webových stránek (sekcí), které jsou vzájemně propojené odkazy. Jednotlivé webové stránky pojí jednotný vzhled a navigace (menu), která je znázorněna na následujícím Obrázku 8.

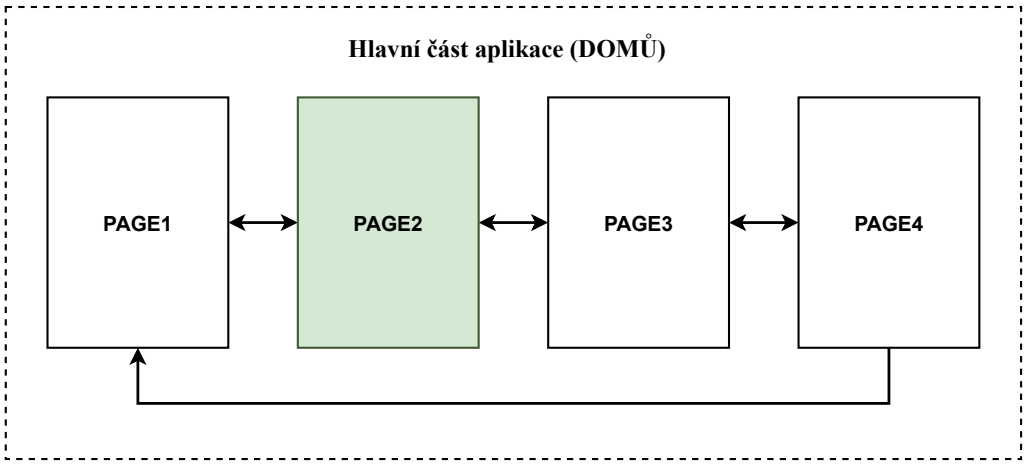


Obrázek 8: Návrh navigační lišty (menu)

Webová aplikace je tedy rozdělena na 3 sekce: DOMŮ, NÁVOD, INFORMACE. V následující části budou jednotlivé sekce blíže popsány.

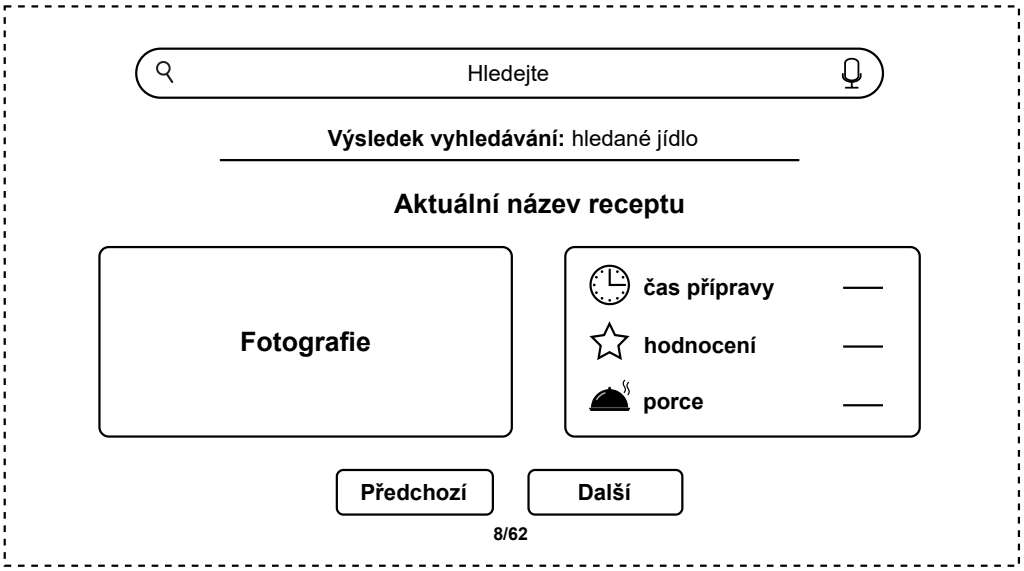
(1) DOMŮ

Tato sekce je považována za domovskou stránku a nachází se zde hlavní část aplikace, tj. samotná hlasová asistentka. Tato část je pouze na jedné webové stránce (HTML), ale podle interakce s uživatelem přepisuje dynamicky svůj obsah novými daty, místo načítání celých nových stránek (tzv. *single-page application*). Tento způsob umožňuje udržování stavu dialogu v rámci jedné relace (session), ale také zajišťuje rychlejší přechod mezi jednotlivým obsahem. Domovskou stránku lze tedy rozdělit podle obsahu na jednotlivé části tzv. *PAGES*. Hlavní část aplikace obsahuje celkem 4 části (PAGE1, PAGE2, PAGE3, PAGE4), přitom v jednu chvíli může být zobrazená (aktivní) pouze jedna z nich. Jednotlivé části (PAGE) lze postupně procházet, ale je umožněno se ve stejném pořadí i vracet. Navíc z poslední části (PAGE4) se lze vrátit zpět na začátek, tím dojde k restartování stránky a zahájí se nová relace (session). Grafické znázornění na Obrázku 9.



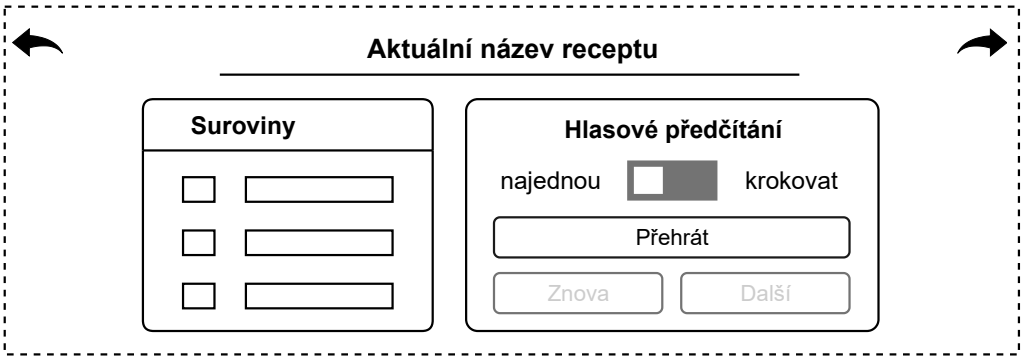
Obrázek 9: Rozdělení hlavní části aplikace. Aktivní PAGE2.

PAGE1 (viz Obrázek 10) umožňuje vyhledávat libovolný recept pomocí vyhledávací lišty. Při úspěšném vyhledávání se vždy zobrazí první nalezený recept. Na stránce se konkrétně zobrazí název hledaného jídla, aktuální název receptu, fotografie receptu a tabulka se základními informacemi o receptu (čas přípravy, hodnocení, počet porcí). V dolní části stránky se nachází tlačítka pro přepínaní jednotlivých receptů. Pod tlačítky se nachází ukazatel aktuálního receptu z celkových nalezených receptů.



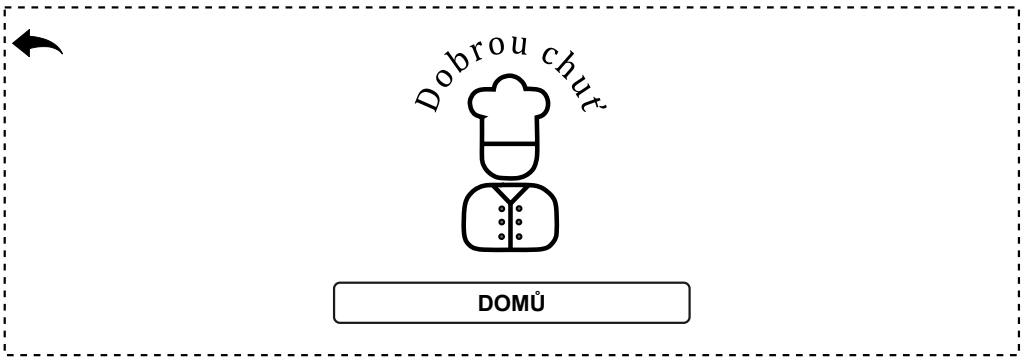
Obrázek 10: Návrh PAGE1

PAGE2 a **PAGE3** jsou si svým návrhem velmi podobné. PAGE2 zobrazuje seznam surovin a PAGE3 zobrazuje postup receptu. V obou částech se nachází panel pro hlasové předčítání položek (surovin nebo jednotlivých instrukcí z postupu). U hlasového předčítání lze přepínat pomocí switche dva režimy (najednou a krokovat). Buď lze položky přehrát všechny *najednou* a nebo je lze možno *krokovat* postupně po jedné. V horní části stránky jsou umístěné šipky pro přechody mezi PAGE. Grafické znázornění na Obrázku 11.



Obrázek 11: Návrh PAGE2 (pro PAGE3 obdobně)

PAGE4 je pouze závěrečná stránka, ze které se lze vrátit zpět na PAGE3 pomocí šipky v levém horním rohu a nebo lze ukončit aktuální relaci a vrátit se zpět na začátek sekce DOMŮ (tedy PAGE1). Grafické znázornění na Obrázku 12.



Obrázek 12: Návrh PAGE4

V sekci DOMŮ se nachází také **ovládací panel**, který je umístěný na stránce v dolní části a je společný pro všechny PAGES. V panelu se nachází ikony, které slouží pro informativní účely nebo pro zobrazení/skrytí daného okna pod panelem. Význam jednotlivých ikon je blíže popsán v kapitole 5.7.1.

(2) NÁVOD

Tato sekce slouží uživateli jako přehled všech možných hlasových příkazů, které může během dialogu použít. Hlasové příkazy nemají vždy pro všechny části (PAGE) stejný význam, proto jsou hlasové příkazy rozepsány pro každou část dialogu (PAGE) zvlášť a k nim je vždy vysvětlena odpovídající akce, kterou lze tímto příkazem docílit.

(3) INFORMACE

Tato sekce slouží uživateli k získání základních informacích o webové aplikaci. Nejprve uživatele seznámí s obecným popisem hlavní části aplikace a její strukturou. Dále nabídne přehled jednotlivých částí s krátkým popisem. Vysvětluje význam jednotlivých ikon v ovládacím panelu, které slouží pro nastavení dialogu a pro informativní účely. Seznamuje uživatele s podmínkami pro používání hlasového ovládání. Dále je zde uvedeno poděkování, kontakt na autora a formulář pro odeslání libovolného dotazu.

5.3 Návrh dialogu

Aplikace je navrhována jako multimodální dialog, takže si uživatel může vybrat způsob ovládání (viz kapitola 5.2). Dialog byl navrhován tak, aby byl pro uživatele dostatečně intuitivní a přirozený. Při návrhu dialogu byl kladen důraz na samotný cíl dialogu, tedy poskytnout asistenci uživateli při vaření a vycházelo se také z předpokladů, které jsou popsány v následujícím odstavci.

Po načtení hlavní části webové aplikace se zobrazí stránka s vyhledávací lištou, která umožňuje vyhledávat libovolný recept. Předpokládá se, že v tento moment je uživatel přítomný u zařízení, na kterém je zobrazená právě tato webová aplikace. Z tohoto důvodu není rovnou aktivní rozpoznávání řeči. V případě, že by chtěl uživatel daný recept zadat pomocí hlasu, musí kliknout na ikonu mikrofonu ve vyhledávací liště, která rozpoznávání řeči aktivuje. Toto řešení zajišťuje, že v případě že uživatel načte hlavní část webové aplikace, ale jídlo se rozhodne nevyhledat hned, nemusí běžet hlasové rozpoznávání řeči zbytečně.

Po úspěšném vyhledávání daného receptu se automaticky aktivuje hlasové rozpoznávání řeči. Od tohoto momentu je možné dojít do cíle i výhradně pomocí hlasu, nevýhodou je, že v případě probíhající syntézy, nelze použít hlasové ovládání. Při hlasové syntéze je hlasové rozpoznávání řeči vypnuté, aby nedocházelo k rozpoznávání syntetizované řeči. Dále dialog umožňuje vybrat konkrétní recept, zobrazit a přehrát seznam surovin a seznam jednotlivých instrukcí v postupu.

5.3.1 Řízení

Pro správné fungování dialogu je nutné vhodným způsobem navrhnou jeho řízení. Celkové řízení dialogu zajišťuje dialogový manažer, který je na straně serveru, společně s JavaScriptem, který je na straně klienta (uživatelské rozhraní). Dialogový manažer má na starost především vyhledávání receptů na internetu (popisováno v kapitole 5.5). Ve stažených receptech zpracovává informace (hodnocení, čas přípravy, počet porcí, suroviny, postup) tak, aby byly připravené na zobrazení v grafickém rozhraní a text byl připraven pro syntézu řeči (popisováno v kapitole 5.6). Upravená data receptu následně postupně podle potřeby přeposílá do JavaScriptu. Dialogový manažer také zpracovává výsledky hlasového rozpoznávání řeči (hledané jídlo) resp. porozumění řeči (hlasové příkazy - kapitola 5.4.1). JavaScript má na starost především zobrazovaný obsah webu, který se dynamicky mění podle interakce s uživatelem, ale také zpracovává hlasové příkazy a jednotlivé recepty, které dostává

od dialogového manažera. JavaScript také zajišťuje posílání textu na syntézu řeči do SpeechCloudu.

5.3.2 Komunikace

Podle navrhované struktury (viz kapitola 5.1) lze aplikaci rozdělit na tři části, konkrétně na dialogového manažera, SpeechCloud a na uživatelské rozhraní. Jednotlivé části mezi sebou musí umět komunikovat. Komunikaci na straně dialogového manažera zajišťuje Websocket server a v uživatelském rozhraní je komunikace realizována pomocí SpeechCloud clienta, který je v tomto případě implementován v JavaScriptu. Tyto dvě části komunikují přes SpeechCloud, kde komunikaci zajištuje SpeechCloud API server. Ten je navíc propojen s moduly, které souvisejí s řečí (ASR, SLU, TTS). Pro výměnu řídících zpráv se používá formát JSON, který využívá protokol Websocket (kapitola 4.5) a pro výměnu řečového signálu (řeč uživatele nebo syntetizované řeč) se používají protokoly SIP/RTP.

U multimodálního dialogu se může lišit komunikace mezi jednotlivými částmi struktury dialogu, podle způsobu použití (hlasová a vizuální interakce), i přes to, že uživatel má stejný požadavek. Při využití hlasové interakce je komunikace mezi jednotlivými částmi vždy složitější.

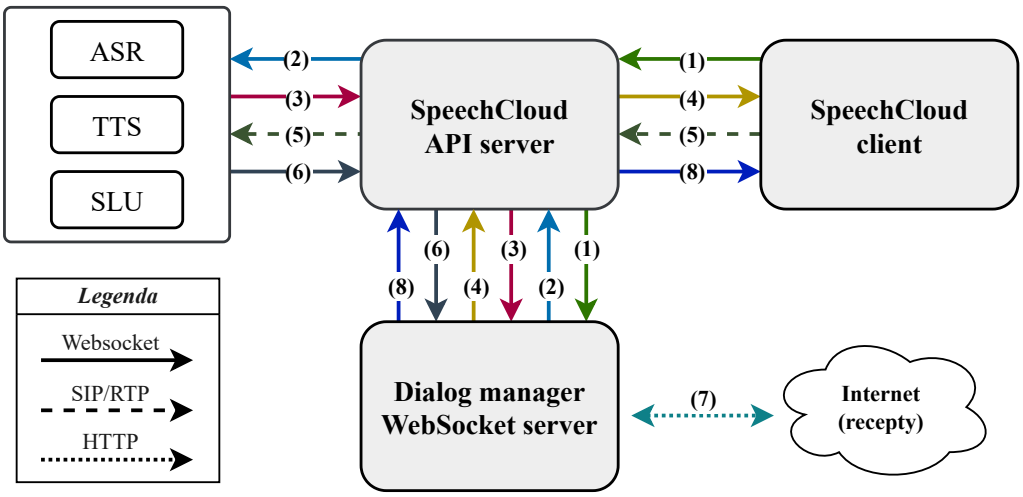
Například pokud uživatel chce vyhledat dané jídlo, komunikace mezi jednotlivými částmi bude vypadat následujícím způsobem:

1. **Vizuální interakce**
   Uživatel zadá jídlo pomocí vyhledávací lišty. SpeechCloud (SC) client přes SC API Server pošle název jídla do dialogovýho manažera (DM), ten vyhledá dané jídlo na internetu a nalazené recepty pošle zpět přes SC API server k SC clientovi, který výsledné recepty může prezentovat uživateli.

2. **Hlasová interakce**
   u hlasové interakce je komunikace mezi jednotlivými modely složitější, proto je pro lepší představu znázorněná na Obrázku 13. Komunikace je rozdělena do následujících bodů:

   (1) uživatel klikne na ikonu mikrofonu ve vyhledávací lište, která pošle řídící zprávu (požadavek o aktivaci ASR) přes SC API server do DM
   (2) DM pošle řídící zprávu do SC API serveru, který aktivuje modul ASR
   (3) po aktivaci modul ASR vygeneruje událost o tom, že je aktivní a pošle zpět přes SC API server do DM

(4) DM odešle informaci o aktivaci ASR přes SC API server k SC clientovi
(5) uživatel pomocí hlasu zadá požadované jídlo a zvuková stopa se odešle přes SC API server do modulu ASR
(6) modul ASR rozpozná příchozí řeč a textovou podobu pošle přes SC API server do DM
(7) DM vyhledá recepty požadovaného jídla na internetu
(8) DM odešle nalezené recepty přes SC API server k SC clientovi, který je prezentuje uživateli



Obrázek 13: Ukázka způsobu komunikace (hlasová interakce)

5.3.3 Reprezentace stavu

Dialog pro správné fungování využívá především stav řešené úlohy. Stav řešené úlohy je ovlivněn hlavně tím, jaká PAGE je právě aktivní. Na PAGE1 určuje stav především to, zda byl už vyhledán nějaký recept, popřípadě na pořadí zobrazovaném receptu. Na PAGE2 a PAGE3 závisí stav hlavně na tom, zda je spuštěné hlasové předčítání. V případě, že je spuštěné, záleží na nastavení switche (najednou/krokovat) a na aktuální položce, která je právě přehrávaná. Stav ovlivňuje také to, zda je spuštěné hlasové rozpoznávání (ASR/SLU) nebo právě probíhá syntéza řeči (TTS).

DM využívá stav především pro získávání receptů z internetu a pro správné vykonání akcí po hlasovém rozpoznávání/porozumění. JavaScript využívá stav hlavně pro změnu obsahu na stránce a pro vykonávání dalších akcí související s interakcí.

5.4 Porozumění řeči

Pro porozumění řeči je zde aplikován znalostní přístup (popisován v kapitole 2.2.1), který využívá bezkontextovou gramatiku k popisu určitých slov/entit a jejich význam převádí na významovou reprezentaci. Porozumění řeči je realizováno pomocí platformy SpeechCloud (popisována v kapitole 3), která podle definovaných gramatik ve formátu SRGS resp. ABNF dokáže určit význam jednotlivých slov/entit. Modul SLU rozhoduje pomocí algoritmu SED, jestli dané slovo patří do jazyka generované gramatikou.

Gramatika byla navrhována takovým způsobem, aby se s její pomocí mohla webová aplikace přirozeně ovládat, ale zároveň, aby gramatika nebyla příliš složitá a byla pro všechny uživatele dostatečně intuitivní. Příklad jednoduché definice ABNF gramatiky:

$$\texttt{\$command} = (\ (\underbrace{\texttt{další | následující | dále}}_{=\text{ alternativní slova}})\ \underbrace{\{\texttt{next}\}}_{=\text{ tag}}\ );$$

V další části bude *tag* označován jako *hlasový příkaz*. Pokud tedy uživatel v tomto příkladě vysloví jedno z alternativních slov (další, následující, dále), gramatika vrátí daný hlasový příkaz (`next`). Každý hlasový příkaz má daný význam, který je navíc ovlivněn stavem - PAGES (popisováno v kapitole 5.3.3). Gramatika je definovaná pomocí jednoho slova nebo maximálně dvou slov. Je velmi jednoduchá z důvodu, že většina hlasových příkazů může být aktivována pomocí jednoslovného názvu na tlačítkách ve webové aplikaci.

Například pokud je zobrazený recept na PAGE1 a uživatel by chtěl pomocí hlasového příkazu zobrazit další recept může vyslovit buď jednoslovný příkaz „***další***“ a nebo použít delší formulaci např. „*Prosím načíst* ***další*** *recept.*“, v obou případech se aktivuje hlasový příkaz `next`, který načte další recept v případě, že je k dispozici.

Pro zadávání jídla pomocí hlasu není dané jídlo vybíráno podle gramatiky, ale opačným způsobem. Tedy z rozpoznané promluvy jsou odstraněna tzv. *balastní slova*. Hlavní důvod tohoto přístupu je ten, že všechny názvy jednotlivých receptů je téměř nemožné pokrýt v rámci definované gramatiky a zároveň umožňuje přirozený způsob zadávání. Například pokud uživatel bude chtít vařit *guláš*, nemusí vyslovit pouze konkrétní slovo *guláš*, ale může jídlo vyhledat přirozeně pomocí věty, kde budou odstraněna balastní slova. Například u promluvy *„~~Dnes~~ ~~bych~~ ~~chtěl~~ ~~vařit~~ guláš.“* jsou odstraněná předem definovaná balastní slova (přeškrtnutá) a zbude konkrétní jídlo.

5.4.1 Hlasové příkazy

Webovou aplikaci lze ovládat pomocí hlasových příkazů. Pro každou PAGE jsou povolené jen přesně dané příkazy, navíc stejný příkaz může mít na jednotlivých PAGE jiný význam, z tohoto důvodu bude v následující části poskytnut přehled jednotlivých povolených příkazů a k nim odpovídající akce pro jednotlivé PAGE.

Přehled povolených příkazu na jednotlivých PAGE:

PAGE1 (Vyhledávání a procházení jídel)

- `next` → zobrazí další recept
- `previous` → zobrazí předchozí recept
- `selection` → vybere aktuálně zobrazený recept a zobrazí PAGE2

PAGE2 a PAGE3 (Zobrazení a přehrávání surovin nebo postupu)

- `change_switch` → změní nastavení switche (najednou/krokovat)
- `previous_page` → zobrazí předchozí stranu (PAGE1/PAGE2)
- `next_page` → zobrazí následující stranu (PAGE3/PAGE4)
- `play` → spustí hlasové předčítání
- `stop` → ukončí hlasové předčítání
- `continue` → pokračuje v hlasovém předčítání (pokud je pozastavené)
- `again`* → přehraje znovu poslední surovinu/instrukci z postupu
- `next`* → přehraje další poslední surovinu/instrukci z postupu

  (lze použít při spuštěném hlasovém přehrávání s nastavením switche na krokovat)*

PAGE4 (Závěr)

- `home` → ukončení aktuální relace a zobrazení PAGE1
- `previous_page` → zobrazí předchozí stranu PAGE3
- `test_voice`* → přehrání ukázky hlasu
- `end_dialog`* → ukončí dialog (relaci) a zobrazí PAGE1

  (lze použít pro všechny PAGES)*

Seznam jednotlivých příkazů a k nim odpovídající akce jsou k dispozici uživateli přímo ve webové aplikaci v sekci NÁVOD nebo také v hlavní části aplikace, kde lze v ovládacím panelu zapnout NÁPOVĚDU pro hlasové příkazy.

5.5 Extrakce dat z internetu

Důležitou součástí webové aplikace jsou recepty. Jelikož existuje spousta internetových stránek s recepty, které jsou volně přístupné a obsahují velké množství dat, nabízí se tedy možnost, získávat recepty přímo z webových stránek. Ruční kopírování

receptů a ukládání na lokální server je sice jedna z možností, jak recepty z webových stránek získat, nicméně tato metoda je velmi pomalá a neefektivní. Pro automatické a rychlé stahování dat lze použít tzv. *web scraping*.

Web scraping je metoda, které umožňuje extrakci dat z internetu. Používá se požadavek protokolu HTTP, konkrétně metoda GET, k získání informací z konkrétní webové stránky pomocí daného URI, vrátí se odpověď v podobě HTML stromu. K získání konkrétní informace z HTML je důležitá analýza struktury webových stránek, ze které se web scraping provádí.

Pro webovou aplikaci jsou získávány recepty z jedné webové stránky[4], pro kterou byla provedena důkladná analýza její HTML struktury (pro lepší představu, znázorněná na Obrázku 14). Po vyhledání receptu se zobrazí údaj o celkovém počtu receptů a přehled jednotlivých receptů, které jsou zobrazovány podle počtu na několika stránkách, mezi kterými lze přepínat. Nicméně tento přehled nabízí pouze název receptu, jeho krátký popis a popř. i jeho fotografii, nicméně všechna data o receptu nelze na této stránce najít. Pro konkrétní data o receptu je zapotřebí daný recept rozkliknout, tím se načte nové HTML, které obsahuje všechny data o receptu ve formátu JSON, který je navíc dodržován podle daného schéma[5] pro recepty. Tyto informace lze využít pro extrakci dat z této stránky.

Pro získávání dat budou důležité tyto informace:

1. celkový počet nalezených receptů
2. odkazy na jednotlivé recepty z přehledu
3. data jednotlivých receptů (formát JSON)

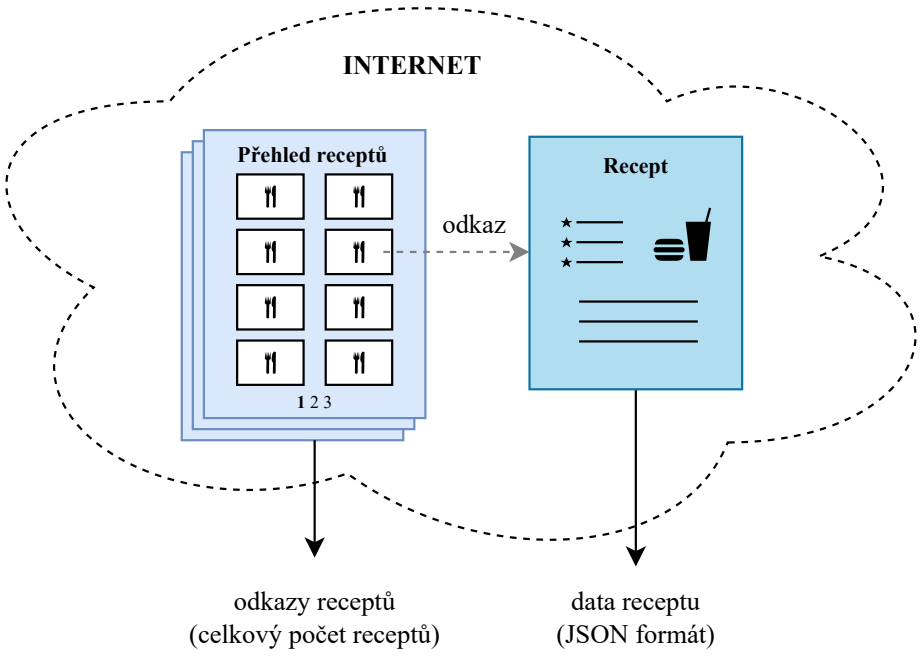
Extrakce dat z webové stránky probíhá v dialogovém manažeru následujícím způsobem. Uživatel zadá ve webové aplikaci dané jídlo, následně DM vyhledá jídlo na webové stránce, zjistí informaci o celkovém počtu receptů a stáhne všechny odkazy na recepty z přehledu na první stránce jedním požadavkem. Podle stažených odkazů lze stáhnout data jednotlivých receptů, ovšem v tomto případě lze získat jedním požadavkem pouze data o jednom receptu. Z úspory přenosu dat se nestahují všechny recepty najednou, ale pouze určité stanovené množství a při každém načtení dalšího receptu ve webové aplikaci se stáhne v DM další recept z webové stránky podle následujícího staženého odkazu.

[4] `https://www.recepty.cz/`

[5] `https://schema.org/Recipe`

Pokud je zásoba stažených odkazů na recepty, které ještě nebyly staženy, menší než určité množství, zkontroluje se, jestli je k dipozici na webových stránkách další stránka s přehledem receptů (tj. pokud celkové množství stažených odkazů na recepty je menší než celkový počet nalezených receptů) a pokud ano, stáhnou se všechny odkazy z následující stránky.

Tímto způsobem lze automaticky získávat recepty v reálném čase. Stažené recepty před odesláním z DM do webové aplikace (JS) se musí ještě zpracovat (viz následující kapitola 5.6).



Obrázek 14: Struktura webová stránky s recepty

5.6 Zpracování informací

Získané data z webových stránek jsou přijímané ve formátu JSON, který je dodržován podle standardizovaného schéma pro recepty. Schéma se skládá z jednotlivých položek, ke kterým je přiřazena hodnota. V daném schématu se nachází mnoho položek, nicméně webová aplikace využívá jen část z nich, které jsou uvedeny v následujícím přehledu.

Přehled potřebných položek (název - popis):

- `name` - název receptu
- `image` - fotografie receptu (URL adresa)
- `aggregateRating` - hodnocení receptu, složené z dalších položek
  - `ratingValue` - číselné hodnocení receptu (0-5)
  - `reviewCount` - celkový počet hodnocení
- `totalTime` - celkový čas vaření
- `recipeYield` - počet porcí
- `recipeIngredient` - seznam jednotlivých ingrediencí (surovin)
- `recipeInstructions` - postup receptu

Některé z vyjmenovaných položek je nutné před odesláním do webové aplikace zpracovat takovým způsobem, aby jednotlivé informace byly ve vhodné formě pro prezentování uživateli a potřebný text byl připraven na syntézu řeči (přepis do plné slovní formy, skloňování). V následující části budou jednotlivé úpravy blíže popsány.

Základní informace o receptu

(1) čas přípravy (`totalTime`)

Hodnota `totalTime` je ve formátu *"PT* + `numberMin` + *M"*, např. *"PT15M"*. V tomto případě je cílem zobrazit celkový čas jako *15 minut*. Nicméně správné skloňování slova *„minuta“* je ovlivněno počtem minut (`numberMin`). Proto je nutno ošetřit správné skloňování pro všechny případy následujícím způsobem:

- `numberMin` $= 0 \rightarrow$ neuvedeno
- `numberMin` $= 1 \rightarrow$ minuta (př. 1 minuta)
- `numberMin` $\in < 2; 4 > \rightarrow$ minuty (př. 3 minuty)
- `numberMin` $\geq 5 \rightarrow$ minut (př. 13 minut)

(2) hodnocení (`aggregateRating`)

Hodnota `aggregateRating` se skládá ze dvou položek, z `ratingValue` jejíž

hodnota je desetinné číslo v intervalu $< 0; 5 >$ a z `reviewCount` jejíž hodnota udává počet recenzí. Uživateli je prezentován upravený výsledná formát ve tvaru "`ratingValue`/5" (př. hodnocení 2.74/5), pokud je počet recenzí `reviewCount` > 0. V opačném případě je výsledné hodnocení prezentováno jako *"neznáme"*.

(3) počet porcí (`recipeYield`)
Hodnota `recipeYield` je ve formátu "`numberYield` + `stringYield`", kde `numberYield` je číslo udávající počet porcí a `stringYield` je textový řetězec, který nabývá hodnot *"porce"* nebo *"porcí"* podle počtu porcí `numberYield`. Skloňování je v tomto případě vyřešené. Zbývá vyřešit pouze poslední případ, kdy položka `recipeYield` nabývá hodnoty *"0 porcí"*. Potom výsledná hodnota bude prezentována uživateli jako *"neuvedeno"*.

Ingredience receptu

Položka `recipeIngredient` obsahuje seznam ingrediencí. U jednotlivých surovin je zapotřebí upravit jejich množství a jednotky (resp. zkratky), aby byly připravené na syntézu řeči. Pro správnou výslovnost jednotlivých ingrediencí je zapotřebí upravit:

1. **přepis zkratky na plnou slovní formu**
   (např. 1 ks vejce, úprava: ks → kus)
2. **správná koncovka jednotky**
   (např. 1 kus, 2 kusy, 5 kusů)
3. **správný tvar číslovky** (výjimky pouze pro množství 1 a 2)[6]
   (např. jedno balení, jeden kus, jedna lžíce, dva kusy, dvě lžíce)

Bod číslo 1. a 2. platí pro všechny možné jednotky (zkratky), které se používají při zadávání jednotlivých ingrediencí. Dále je potřeba upravit výslovnost i pro desetinná čísla. Nejčastější desetinné části se přepíší do tvaru:

- 0.25 → čtvrt
- 0.50 → půl
- 0.75 → tři čtvrtě

Posledním krokem úpravy pro desetinná čísla je nutnost rozlišení, zda desetinné číslo začíná *nulou* nebo *nenulovým kladným číslem*. Znázorněno na následujících příkladech (příklad: výchozí výslovnost TTS → výslovnost po úpravě):

1. **číslo začínající nulou**

[6] Pro ostatní množství (čísla) lze využít výchozí výslovnost modulu TTS.

- *např. 0.5 hrnek:* nula tečka pět hrnek → půl hrnku
- *např. 0.2 hrnek:* nula tečka dva hrnek → žádná celá dva hrnku

2. **číslo začínající nenulovým kladným číslem**
   - *např. 2.5 lžička:* dva tečka pět lžička → dva a půl lžičky
   - *např. 3.2 lžička:* tři tečka dva hrnek → žádná celá dva lžičky

Postup receptu

Položka `recipeInstructions` obsahuje postup receptu, který je reprezentován jedním textovým řetězcem (blokem). Pro účely hlasové asistentky by bylo vhodné tento postup rozdělit na dílčí části (jednotlivé instrukce postupu). Hlasová asistentka pak bude moct přehrávat daný postup receptu postupně neboli krokovat postup po jedné instrukci.

Postup lze rozdělit na jednotlivé instrukce tím, že pomocí interpunkčního znaménka (tečky) se rozdělí blok postupu na jednotlivé věty. Problém je ovšem v tom, že tečka není jenom na konci věty, ale je také umísťována například za zkratkami (např., tzv., min., atd.) nebo je součástí desetinného čísla. Zachování desetinného čísla, lze docílit tím, že věty budou rozdělovány pomocí tečky a mezery (`".␣"`). Nicméně někdy se může stát, že mezera za větou bude chybět. Je zapotřebí tedy přidat mezeru pouze za tečkou na konci věty, kde daná mezera chybí, ale zároveň nepřidat mezeru k tečce u desetinného čísla. To lze docílit pomocí správného regulárního výrazu. Jednotlivé kroky dělení postupu jsou tedy následující:

1. přidat chybějící mezeru za tečku pomocí regulárního výrazu
2. odstranit tečku u jednotlivých zkratek
3. rozdělit postup na jednotlivé instrukce podle tečky a mezery (`".␣"`)
4. přidat tečku zpět k jednotlivým zkratkám
5. krátké instrukce než je určená min. délka, spojit s předchozí instrukcí

5.7 Realizace a testování

Dialogový manažer využívá asynchronní programování, které umožňuje obsluhovat multimodální dialog s uživatelem a zároveň vyhledávat recepty na internetu. Webová aplikace je realizována formou responzivního web designu, který zajišťuje, že jsou stránky optimalizovány pro různé druhy zařízení. To znamená, že se webová aplikace dobře zobrazí nejen na počítači, ale i na mobilním telefonu nebo tabletu.

Pro responzivitu webu nebyl použit žádný framework, ale pouze vlastní CSS styly. Všechny ikony (mimo ikon ve vyhledávací lište), závěrečný obrázek i pozadí jsou ve vektorovém formátu SVG a byly vytvářeny ve vektorovém editoru Inscape. Při realizaci webové stránky probíhalo průběžné testovaní (komunikace, vzhled a responzibilita na jednotlivých zařízeních, syntéza řeči, hlasové rozpoznávání, atd.).

5.7.1 Ovládací panel

Na Obrázku 15 je zobrazen ovládací panel, který se nachází ve webové aplikaci v dolní části stránky. V panelu se nacházejí ikony, které slouží k nastavení dialogu nebo pro informativní účely.

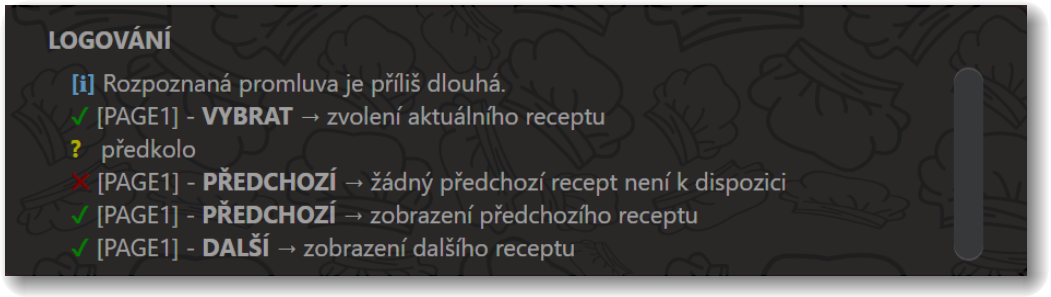
Obrázek 15: Ovládací panel

Význam jednotlivých ikon je následující:

1. **IKONY** *(levá strana)*
   - **Logování** - historie rozpoznaných příkazů/promluv
   - **Nápověda** - zobrazení hlasových příkazů podle PAGE
   - **Nastavení hlasu** - možnost výběru hlasu a přehrání ukázky hlasu
   - **Komentování dialogu**

2. **IKONY** *(pravá strana)*
   - **Syntéza řeči** - zelená → probíhá, červená → neprobíhá
   - **Hlasové rozpoznávání** - zelená → zapnuté, červená → vypnuté
   - **Indikátor řeči** - barva ikony podle intenzity signálu (tzv. *semafor*)

Kliknutím na ikonu **logování** se zobrazí okno, ve kterém jsou zaznamenány všechny rozpoznané hlasové příkazy nebo promluvy (pokud nejsou příliš dlouhé). V logovacím okně jsou zobrazeny pouze rozpoznané příkazy, které jsou povolené pro danou PAGE, ale může nastat případ, kdy daný příkaz bude pro danou PAGE povolený, nicméně kvůli aktuálnímu stavu dialogu nebude možné hlasový příkaz vykonat. Například pokud je uživatel na PAGE1 a je zobrazen první recept, ale uživatel požaduje hlasovým příkazem načíst předchozí recept, potom hlasový příkaz bude zaznamenán v logovacím okně, ale nebude vykonán. Může nastat i situace, kdy uživatel vysloví povolený příkaz pro danou PAGE a stav dialogu, ale příkaz přesto vykonán nebude, pro takové účely slouží právě logovací okno, kde si může ověřit, zda byl hlasový příkaz správně rozpoznán. Grafické znázornění logovacího okna na Obrázku 16.



Obrázek 16: Logovací okno

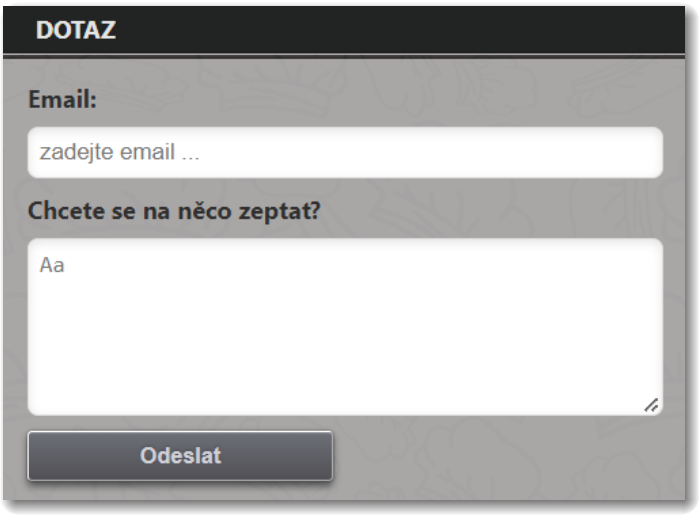
Pokud je vypnuté **hlasové komentování**, hlasová asistentka slouží pouze pro hlasové předčítání ingrediencí nebo jednotlivých instrukcí z postupy. V opačném případě, kdy je hlasové komentování zapnuté, hlasová asistentka navíc komentuje i průběh dialogu. V následující části budou popsány jednotlivé akce, které hlasová asistentka komentuje v případě, že je zapnuté hlasové komentování:

1. **zapnutí/vypnutí hlasového komentování**
   „Zapnuli jste hlasové komentování.“
   „Hlasové komentování bylo ukončeno.“
2. **výsledek vyhledávání**
   „Hledaný recept:“ název *„Celkový počet nalezených receptů:“* číslo
   „Omlouváme se, ale nenalezli jsme žádný recept.“
3. **spuštění/ukončení hlasového předčítání**
   „Následuje hlasové předčítání ingrediencí/postupu.“
   „Hlasové předčítání skončilo.“
4. **změna způsobu předčítání (switche)**
   „Změna způsobu předčítání. Hlasové předčítání se bude krokovat po jednom.“
   „Změna způsobu předčítání. Hlasové předčítání se bude předčítat najednou.“
5. **hlasový příkaz nelze použít**
   „Jste na začátku. Žádný předchozí recept není k dispozici.“
   „Hlasový příkaz nelze použít. Hlasové předčítání není spuštěné.“
   + mnoho dalších.
6. **přechod mezi PAGE**
   - PAGE1 → PAGE2
     „Stránka zobrazující ingredience receptu.“
   - PAGE2 → PAGE3
     „Stránka zobrazující postup receptu.“
   - PAGE3 → PAGE4
     „Gratulujeme, máte vařeno! Přejeme dobrou chuť.“

5.7.2 Zpětná vazba uživatele

Webová aplikace je ve vývojové fázi, proto je umístěn v sekci INFORMACE formulář (znázorněný na Obrázku 17), který slouží pro poskytnutí zpětné vazby. Pokud uživateli nebude webová aplikace během používání správně fungovat nebo bude mít nějakou připomínku/nápad na zlepšení aplikace, může poskytnou zpětnou vazbu vývojáři pomocí formuláře. Ve formuláři je zapotřebí vyplnit emailovou adresu, která slouží pro případnou odpověď uživateli a konkrétní obsah dané zprávy/dotazu.

Email je odeslán do dialogového manažera, který následně odešle email vývojáři aplikace pomocí protokolů SMTP a SSL. Tento způsob umožňuje vývojáři rychlou reakci na příchozí zprávu/dotaz a může tak uživateli poskytnou rychlou odpověď.



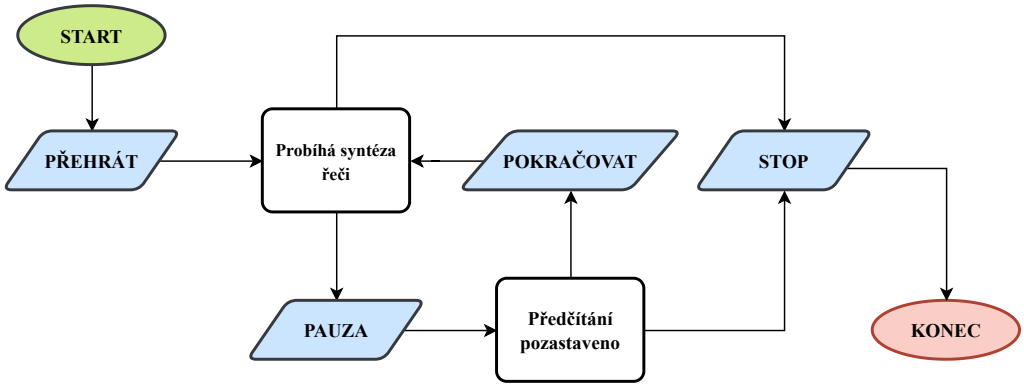
Obrázek 17: Formulář (zpětná vazba uživatele)

5.7.3 Hlasové předčítání

Webová aplikace poskytuje hlasové předčítání ingrediencí nebo jednotlivých instrukcí z postupu. Hlasové předčítání se ovládá pomocí panelu (znázorněn na Obrázku 11), kde lze nastavit způsob předčítání (najednou/krokovat). V panelu se nacházejí tlačítka, která slouží k řízení hlasového předčítání. Tlačítka se ale dynamicky mění podle způsobu přehrávání a stavu dialogu. Ovládací panel lze ovládat i pomocí hlasových příkazů, ale pouze pokud neprobíhá syntéza řeči. V následující části bude blíže vysvětleno, v jakých stavech lze jednotlivá tlačítka použít v závislosti na způsobu předčítání.

1. **Způsob předčítání - najednou**
   Pomocí tlačítka *Přehrát* spustí uživatel hlasové přehrávání, které začne v tomto

režimu přehrávat postupně všechny položky. V ten samý moment se tlačítko *Přehrát* změní na dvě tlačítka, a to na tlačítko *Stop*, kterým lze ukončit hlasové předčítání, a tlačítko *Pauza*, které hlasové předčítání pozastaví a změní se na tlačítko *Pokračovat*. Kliknutím na tlačítko *Pokračovat* se opět spustí hlasové předčítání od položky, u které bylo hlasové předčítání pozastaveno. Po přehrání poslední položky se hlasové předčítání ukončí a zobrazí se opět pouze tlačítko *Přehrát*. Pro lepší představu je logika tlačítek znázorněná na následujícím Obrázku 18 formou vývojového diagramu.



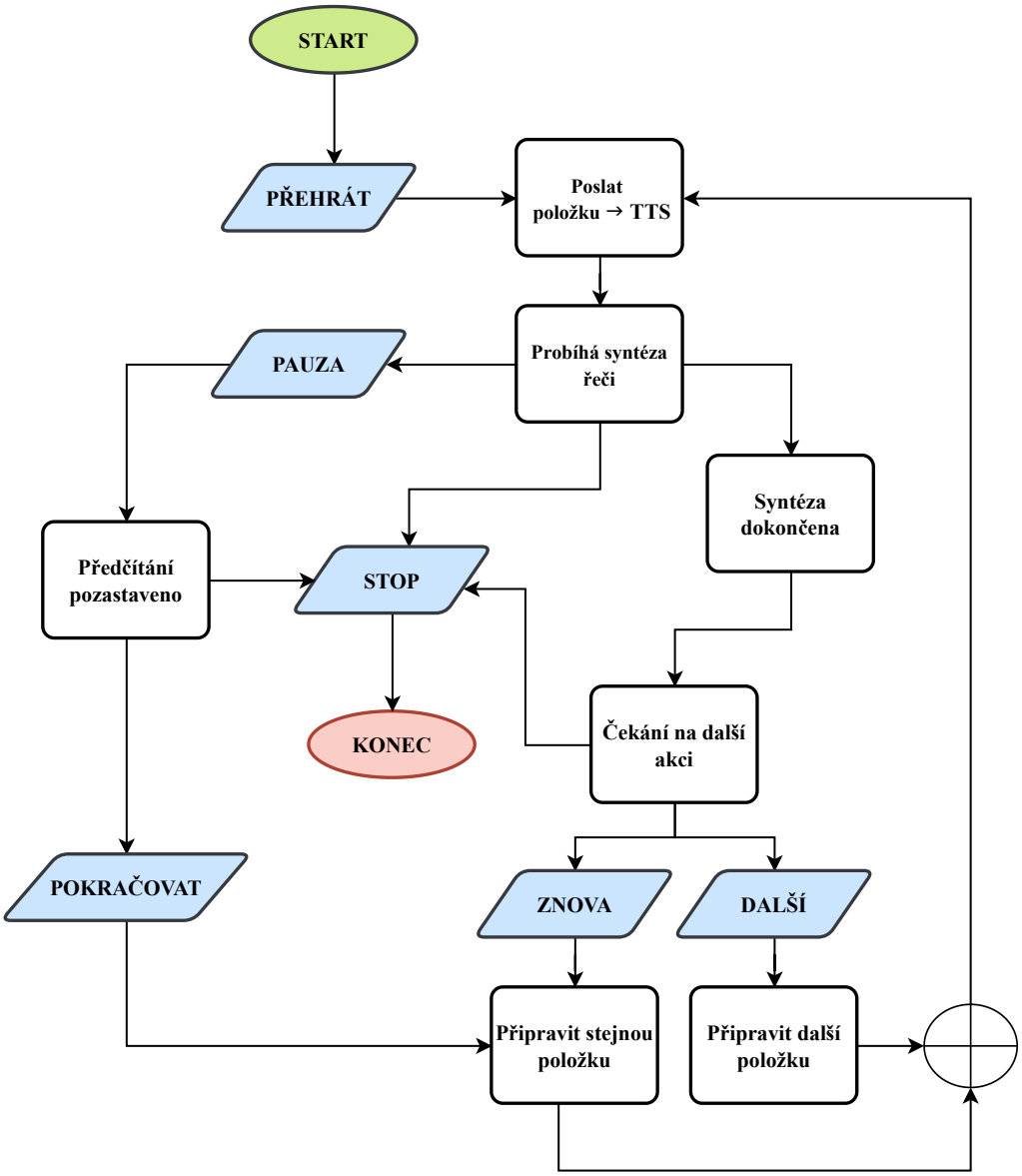
Obrázek 18: Vývojový diagram hlasového předčítání (najednou)

2. **Způsob předčítání - krokovat**

   Pomocí tlačítka *Přehrát* se spustí hlasové předčítání, které začne v tomto režimu předčítat položky vždy pouze po jedné. Pokud probíhá syntéza řeči, logika používání tlačítek je stejné jako při 1. způsobu předčítání. Po přehrání jedné položky se syntéza řeči ukončí a hlasová asistentka čeká na další pokyn. Uživatel může hlasové předčítání ukončit pomocí tlačítka *Stop*, ale také přehrát další položku pomocí tlačítka *Další* nebo přehrát znovu poslední položku pomocí tlačítka *Znova*. Pokud se přehraje poslední položka, hlasové předčítání se neukončí automaticky, ale musí ho uživatel ukončit sám. Je to z důvodu, že po přehrání poslední položky má uživatel pořád možnost poslední položku přehrát ještě jednou pomocí tlačítka *Znova*. Pro lepší představu je logika tlačítek znázorněná na následujícím Obrázku 19 formou vývojového diagramu.

5.7.4 Průběh dialogu

Dialog začíná na PAGE1, kde se po načtení webové stránky zobrazí vyhledávací lišta, přes kterou lze vyhledat libovolný recept s využitím klávesnice nebo pomocí



Obrázek 19: Vývojový diagram hlasového předčítání (krokovat)

hlasu. Při hlasovém zadávání je potřeba kliknout na ikonu mikrofonu ve vyhledávací liště a řídit se pokyny, které se v liště zobrazí. Mohou nastat 3 následující případy:

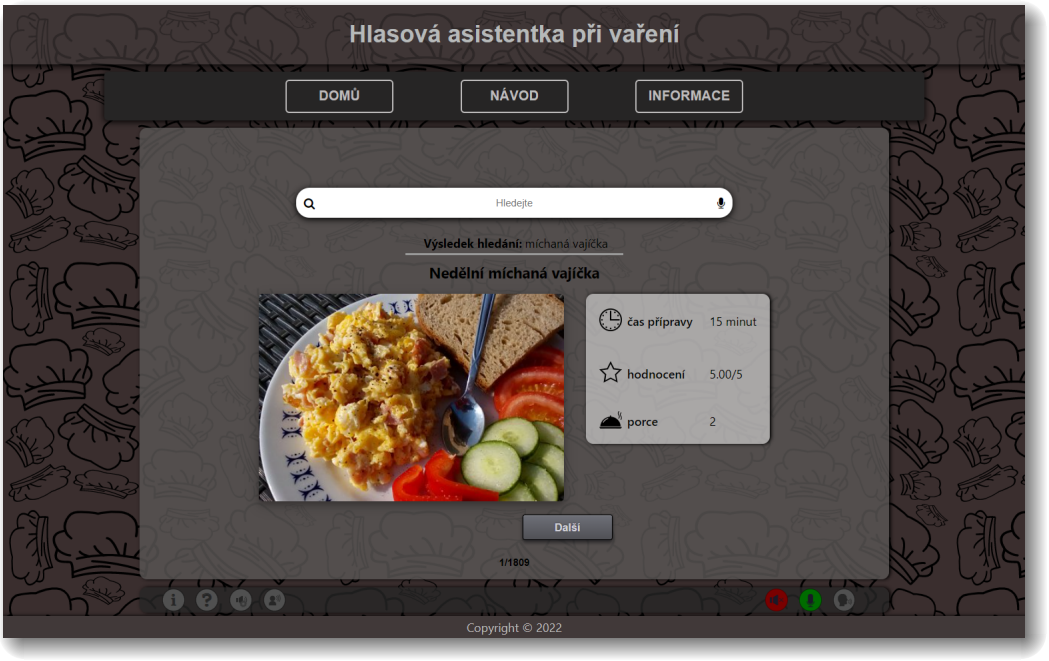
1. **Probíhá inicializace rozpoznávání řeči.** *(pouze na začátku relace)*
2. **Prosím čekejte.**
3. **Poslouchám.**

Uživatel může začít zadávat jídlo pomocí hlasu v případě, že se ve vyhledávací liště zobrazí 3. možnost. Uživatel kromě informace z vyhledávací lišty může využít i ikonu hlasového rozpoznávání v ovládacím panelu. Když ikona změní svojí barvu na zelenou, tak to značí, že je rozpoznávání aktivní.

Následně zadané jídlo vyhledá dialogový manažer na internetu (kapitola 5.5) a poté data o receptu zpracuje (kapitola 5.6). Zpracované recepty posílá postupně do JS, který jednotlivé informace zobrazuje ve webové aplikaci. Uživatel pak může jednotlivé recepty libovolně procházet. Pro rychlejší zobrazení následujícího receptu je jeho fotografie dopředu stažená ze severu webových stránek, odkud se získávají recepty a načtená do paměti prohlížeče.

Uživatel daný recept může vybrat kliknutím na fotografii nebo na název receptu, ale také pro výběr může využít hlasový příkaz. Tím se dostane na PAGE2, kde se zobrazí ingredience receptu, uživatel zkontroluje, že má všechny potřebné suroviny pro daný recept, buď podle zobrazeného seznamu na webové stránce nebo může využít jednu z možností hlasového předčítání (kapitola 5.7.3). Pokud uživatel zjistí, že některá z ingrediencí mu chybí, může se vrátit zpět na PAGE1 a pokračovat ve výběru receptu. V opačném případě, pokud má všechny potřebné suroviny, může pokračovat na PAGE3, kde se nachází postup receptu. Jednotlivé instrukce receptu jsou zobrazeny v tabulce. Opět je zde možnost využít hlasové předčítání. PAGE4 slouží jako závěrečné strana, ze které se může vrátit zpět na PAGE1 a pokračovat ve výběr jiného receptu.

Vzhledem k tomu, že je aplikace responzivní, přizpůsobuje svůj vzhled rozměrům zařízení. Na Obrázku 21 je zobrazena ukázka aplikace na mobilním zařízení a na Obrázku 20 je zobrazena ukázka aplikace na počítači (aktivní PAGE1).



Obrázek 20: Ukázka aplikace na počítači

Obrázek 21: Ukázka aplikace na mobilním zařízení (iOS)

6 Závěr

Cílem této práce bylo navržení HDS, který bude uživateli asistovat při vaření a následná realizace dialogu ve formě webové aplikace.

První část práce je věnována teoretickému popisu problematiky HDS, kde byly postupně představeny jednotlivé části HDS. Byl popsán účel, funkce a principy modulů ASR, SLU, NLG, DM, TTS. Po nich bylo rozebráno samotné vyhodnocení dialogu v HDS.

Druhá část práce blíže popisuje hlasovou platformu SpeechCloud, která byla využita pro realizaci systému v této práci. Byla popsána její architektura a funkčnost. Následně byly představené některé klíčové technologie nezbytné pro realizaci webové aplikace i zbytku systému. Jsou jimi především HTML, CSS, DOM, WebSocket.

Hlavní část práce představoval návrh HDS a s ním spojené webové aplikace. Byla popsána struktura vycházející právě z platformy SpeechCloud, následně byl rozebrán návrh uživatelského rozhraní, který kombinuje možnost hlasové i grafické interakce s uživatelem. Po dokončení návrhu byla pozornost věnována hlasovým příkazům, pomocí kterých lze systém ovládat. Při výběru těchto příkazů, ale také i celého grafického rozhraní a struktury celého uživatelského rozhraní, byl kladen důraz na to, aby bylo ovládání snadno zapamatovatelné a jednoduché. Důvodem je snaha udržet aplikaci a systém uživatelsky přívětivou a pohodlnou na používání.

Jednou z nezbytných funkcí je schopnost systému vyhledat recepty podle požadavků uživatele. Této části byla věnována další část práce, kde je popsán princip web scrapingu receptů z vybraných internetových stránek a jejich následné zpracování.

Výsledkem je funkční webová aplikace, která asistuje uživateli při vaření, jak bylo avizováno v zadání. Aplikace umožňuje uživateli vyhledat recepty, které následně předloží s fotografií a nechá uživatele, aby si vybral konkrétní provedení (recept) hledaného pokrmu. Po výběru aplikace zobrazí a případně přečte uživateli seznam potřebných surovin. Když se uživatel ujistí, že má vše připraveno, tak jej systém doprovází v průběhu vaření tak, že diktuje jednotlivé kroky receptu (nebo jej může přečíst celý najednou, dle volby uživatele). Uživatel může změnit hlas, který systém používá k syntéze, dále může sledovat výpis rozpoznaných příkazů (vhodné pro testování), nebo využít nápovědy dostupné z ovládacího panelu.

Aplikace udržuje v současné implementaci stav dialogu pouze v rámci jedné relace. Pokud tedy uživatel aplikaci ukončí v průběhu vaření, není možné se vrátit zpět do

původního stavu. Jedním z možných rozšíření je právě řešení tohoto problému, které by mohlo spočívat v tom, že by byl udržován stav mimo aplikaci (například někde v databázi) a uživatelé by se přihlašovali svým účtem nebo pomocí prohlížečových cookies. Toto rozšíření by umožňovalo také integraci nějakého nákupního seznamu pro chybějící suroviny, možnost uložit si recepty jako oblíbené a podobně. Jako další možné rozšíření se naskýtá schopnost systému získávat recepty z více různých webů a možnost výsledky nějak inteligentně filtrovat a řadit.

Současná verze běžící webové aplikace je na adrese:
`https://cak.zcu.cz:9444/edu_dialog_start/jtupy//index.html`

Literatura

1. ROE, David B.; WILPON, Jay G. (ed.). *Voice Communication Between Humans and Machines*. Washington, DC: The National Academies Press, 1994. ISBN 978-0-309-04988-7. Dostupné z DOI: 10.17226/2308.
2. ZUE, Victor; SENEFF, Stephanie. Spoken Dialogue Systems. In: *Springer Handbook of Speech Processing*. Ed. BENESTY, Jacob; SONDHI, M. Mohan; HUANG, Yiteng Arden. Berlin: Springer Berlin Heidelberg, 2008, s. 705–722. ISBN 978-3-540-49127-9. Dostupné z DOI: 10.1007/978-3-540-49127-9_35.
3. ŠVEC, Jan. *Hlasové dialogové systémy* [učební text]. 2021. Západočeská univerzita v Plzni.
4. YU, Dong; DENG, Li. Automatic Speech Recognition. In: *Automatic Speech Recognition*. Springer London, 2015, s. 705–722. ISBN 978-1-4471-5779-3. Dostupné z DOI: 10.1007/978-1-4471-5779-3.
5. IRCING, Pavel. *Rozpoznávání řeči , akustické a jazykové modelování* [učební text]. 2021. Západočeská univerzita v Plzni.
6. MOTTL, Patrik. *Laboratoř zpracování řeči* [online] [cit. 2022-05-05]. Dostupné z: https://fel.cvut.cz/cz/vv/tymy/speechlab.
7. PSUTKA, Josef. *Analýza a zpracování řečového signálu, parametrizace řeči* [učební text]. 2021. Západočeská univerzita v Plzni.
8. IBE, Oliver C. 14 - Hidden Markov Models. In: IBE, Oliver C. (ed.). *Markov Processes for Stochastic Modeling (Second Edition)*. Second Edition. Oxford: Elsevier, 2013, s. 417–451. ISBN 978-0-12-407795-9. Dostupné z DOI: 10.1016/B978-0-12-407795-9.00014-1.
9. MONTAVON, Grégoire; SAMEK, Wojciech; MÜLLER, Klaus-Robert. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*. 2018, roč. 73, s. 1–15. ISSN 1051-2004. Dostupné z DOI: 10.1016/j.dsp.2017.10.011.
10. LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. *Nature*. 2015, s. 436–444. ISSN 1476-4687. Dostupné z DOI: 10.1038/nature14539.
11. PSUTKA, J.; MÜLLER, L.; MATOUŠEK, J.; RADOVÁ, V. *Mluvíme s počítačem česky*. Prague: Academia, 2006. ISBN 80-200-1309-1.
12. MATOUŠEK, Jindřich; ŠVEC, Jan. *Řečové techlologie: Od výzkumu k praxi* [učební text]. 2015. Západočeská univerzita v Plzni.

13. CARBONE, Ginevra; SARTI, Gabriele. ETC-NLG: End-to-end Topic Conditioned Natural Language Generation. 2020. Dostupné z DOI: 10.48550/ARXIV.2008.10875.
14. WEN, Tsung-Hsien; GASIC, Milica; MRKSIC, Nikola; SU, Pei-Hao; VANDYKE, David; YOUNG, Steve. *Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems*. arXiv, 2015. Dostupné z DOI: 10.48550/ARXIV.1508.01745.
15. MATOUŠEK, Jindřich. *Syntéza řeči: Zpracování textu a syntéza řeči z textu* [učební text]. 2021. Západočeská univerzita v Plzni.
16. SAGISAKA, Y. Speech synthesis from text. *IEEE Communications Magazine*. 1990, roč. 28, č. 1, s. 35–41. Dostupné z DOI: 10.1109/35.46669.
17. MATOUŠEK, Jindřich; TIHELKA, Daniel; ŠMÍDL, Luboš. On the Impact of Annotation Errors on Unit-Selection Speech Synthesis. In: SOJKA, Petr; HORÁK, Aleš; KOPEČEK, Ivan; PALA, Karel (ed.). *Text, Speech and Dialogue*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, s. 456–463. ISBN 978-3-642-32790-2. Dostupné z DOI: 10.1007/978-3-642-32790-2_55.
18. TAYLOR, P. Unit-selection synthesis. In: *Text-to-Speech Synthesis*. Cambridge Univer. Press, 2009, s. 474–516. Dostupné z DOI: 10.1017/CBO9780511816338.018.
19. OORD, Aaron van den; DIELEMAN, Sander; ZEN, Heiga; SIMONYAN, Karen; VINYALS, Oriol; GRAVES, Alex; KALCHBRENNER, Nal; SENIOR, Andrew; KAVUKCUOGLU, Koray. *WaveNet: A Generative Model for Raw Audio*. arXiv, 2016. Dostupné z DOI: 10.48550/ARXIV.1609.03499.
20. WU, Yi-Chiao; KOBAYASHI, Kazuhiro; HAYASHI, Tomoki; TOBING, Patrick Lumban; TODA, Tomoki. Collapsed Speech Segment Detection and Suppression for WaveNet Vocoder. In: *Proc. Interspeech 2018*. 2018, s. 1988–1992. Dostupné z DOI: 10.21437/Interspeech.2018-1210.
21. RETHAGE, Dario; PONS, Jordi; SERRA, Xavier. A Wavenet for Speech Denoising. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, s. 5069–5073. Dostupné z DOI: 10.1109/ICASSP.2018.8462417.
22. ŠVEC, Jan; NEDUCHAL, Petr; HRÚZ, Marek. Multi-modal communication system for mobile robot. 2022.
23. ŠVEC, Jan. *Knihovna SpeechCloud.dialog* [interní dokument]. 2020. Západočeská univerzita v Plzni.

Seznam obrázků

Seznam použitých zkratek

Zkratka	Popis (Poznámka)
ABNF	Augmented Backus–Naur Form, rozšířená Backusova–Naurova forma
API	Application Programming Interface, aplikační programovací rozhraní
ASR	Automatic Speech Recognition, automatické rozpoznávání řeči
CSS	Cascading Style Sheets, kaskádové styly
DM	Dialog Manager, dialogový manažer (správce)
DNN	Deep Neural Network, hluboké neuronové sítě
DOM	Document Object Model, objektový model dokumentu
GUI	Graphical User Interface, grafické uživatelské rozhraní
HDS	Hlasový Dialogový Systém
HMM	Hidden Markov Model, skryté Markovské modely
HTML	HyperText Markup Language , hypertextový značkovací jazyk
HTTP	HyperText Transfer Protocol (protokol)
JS	JavaScript (programovací jazyk)
JSON	JavaScript Object Notation (způsob zápisu dat)
LPC	Linear Predictive Coding, lineární prediktivní kódování
LSTM	Long Short-Term Memory, dlouhá krátkodobá paměť (neuronová síť)
MFCC	Mel-Frequency Cepstral Coefficient, mezifrekvenční kepstrální koef.
NLG	Natural Language Generation, generování přirozeného jazyka
NLP	Natural Language Processing, zpracování přirozeného jazyka
PLP	Perceptual Linear Prediction, percepční lineární predikce
RTP	Real-Time Transport Protocol (protokol)
SC	SpeechCloud (platforma)
SED	Semantic Entity Detection, detekce sémantické entity
SIP	Session Initiation Protocol (protokol)
SLU	Spoken Language Understanding, porozumění mluvené řeči
SMTP	Simple Mail Transfer Protocol (protokol)
SSL	Secure Sockets Layer (protokol)
SVG	Scalable Vector Graphics, škálovatelná vektorová grafika
TCP	Transmission Control Protocol (protokol)
TTS	Text to Speech, syntéza řeči z textu
URI	Uniform Resource Identifier, jednotný identifikátor zdroje
URL	Uniform Resource Locator, jednotný lokátor zdroje