|  |  |
| --- | --- |
|  | |
| **Právní skener - multiplatformní aplikace k ověřování právních informací** | |
|  | |
| Jiří Einšpigl | |
|  | |
|  |  |
| Bakalářská práce  2025 |  |
|  |  |

Obsah obrázku text, snímek obrazovky, Písmo, dokument

Obsah vygenerovaný umělou inteligencí může být nesprávný.

Obsah obrázku text, dopis, snímek obrazovky, dokument

Obsah vygenerovaný umělou inteligencí může být nesprávný.

**PROHLÁŠENÍ AUTORA BAKALÁŘSKÉ PRÁCE**

**Beru na vědomí, že**

* odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
* bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
* jedno vyhotovení bakalářské práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
* na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
* podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
* podle § 60 odst. 2 a 3 mohu užít své dílo – bakalářskou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
* pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
* pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

**Prohlašuji, že**

* jsem na bakalářské práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jako spoluautor;
* při tvorbě této práce jsem použil nástroje generativních modelů AI ChatGPT, dostupné z https://chatgpt.com/ a Gemini, dostupné z <https://gemini.google.com> za účelem kontroly pravopisu, opravy gramatiky a vhodnějšího formulování textů. Po použití tohoto nástroje jsem provedl kontrolu obsahu a přebírám za něj plnou zodpovědnost.
* odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.

Ve Zlíně, dne .............................. ...............................................................

podpis autora

ABSTRAKT

Cílem bakalářské práce je návrh a vývoj multiplatformní aplikace, která uživatelům umožní snadno ověřovat platnost právních předpisů uvedených v dokumentech. Aplikace využívá technologii optického rozpoznávání znaků (OCR) k převodu textu z fotografií nebo naskenovaných dokumentů do digitální podoby. Následně je analyzovaný text automaticky zpracován a prostřednictvím API databáze zakonyprolidi.cz ověřována platnost identifikovaných právních norem. Výsledná aplikace je implementována pomocí frameworků Ionic a Angular a je dostupná jako progresivní webová aplikace i mobilní aplikace pro platformy Android a iOS. Práce popisuje návrh, implementaci a testování aplikace včetně použitých technologií a architektury. Výsledkem je prototyp, který zjednodušuje a zrychluje práci s právními dokumenty v praxi.

Klíčová slova: multiplatformní aplikace, optické rozpoznávání znaků (OCR), právní předpisy, Ionic, Angular, Tesseract.js, zakonyprolidi.cz, API, Android, iOS, PWA

ABSTRACT

The aim of this bachelor’s thesis is to design and develop a cross-platform application that enables users to easily verify the validity of legal regulations mentioned in documents. The application uses Optical Character Recognition (OCR) technology to convert text from photographs or scanned documents into a digital form. The extracted text is then automatically analyzed, and the validity of identified legal references is verified through the public API of the zakonyprolidi.cz database. The final application is implemented using the Ionic and Angular frameworks and is available as both a Progressive Web Application (PWA) and a mobile application for Android and iOS platforms. The thesis describes the design, implementation, and testing of the application, including the technologies and architecture used. The result is a prototype that simplifies and accelerates work with legal documents in practice.

Keywords: cross-platform application, Optical Character Recognition (OCR), legal regulations, Ionic, Angular, Tesseract.js, zakonyprolidi.cz, API, Android, iOS, PWA

Rád bych poděkoval paní prof. Ing. Zuzaně Komínkové Oplatkové, Ph.D., za pohodu při spolupráci, rady a čas, který mi během vedení této bakalářské práce věnovala.

Kapitola o bezpečnosti (není uživatelský účet, tak se nemusí řešit šifrování atd.)

řešit bezpečnost stahování z API

jsem už nedodělal… ☹

Přidat proč jsem použil tesseract a ne jiné OCR (jen zmínka… ne pořádně)

Jsou stránky na správných místech? (Odsazení na vazbu) - v tom online wordu to asi není vidět...

Zdroje?

Příloha?

Jediná příloha bude aplikace…

OBSAH

[Úvod 11](#_Toc199588380)

1. [teoRetická část 12](#_Toc199588381)

[1 Nástroje a vývojová prostředí 13](#_Toc199588382)

[1.1 Ionic 13](#_Toc199588383)

[1.2 Capacitor 13](#_Toc199588384)

[1.3 Cordova 14](#_Toc199588385)

[1.4 Node.js 14](#_Toc199588386)

[1.5 Visual Studio Code 14](#_Toc199588387)

[1.6 Android Studio 15](#_Toc199588388)

[1.7 Xcode 15](#_Toc199588389)

[2 optické rozpoznávání znaků 16](#_Toc199588390)

[2.1 Amazon Textract 16](#_Toc199588391)

[2.2 Apple Vision Framework 17](#_Toc199588392)

[2.3 EasyOCR 17](#_Toc199588393)

[2.4 Google ML Kit 17](#_Toc199588394)

[2.5 Microsoft Azure Computer Vision 18](#_Toc199588395)

[2.6 Tesseract 18](#_Toc199588396)

[2.7 Scanbot SDK 19](#_Toc199588397)

[3 Programovací jazyky a Framework 20](#_Toc199588398)

[3.1 Angular 20](#_Toc199588399)

[3.2 Javascript 20](#_Toc199588400)

[3.3 Typescript 21](#_Toc199588401)

[3.4 HTML 21](#_Toc199588402)

[3.5 CSS 22](#_Toc199588403)

[3.6 Java 22](#_Toc199588404)

[3.7 Swift 22](#_Toc199588405)

[4 Mobilní aplikace 23](#_Toc199588406)

[4.1 Webová aplikace 23](#_Toc199588407)

[4.2 Nativní Aplikace 23](#_Toc199588408)

[4.3 Android 23](#_Toc199588409)

[4.4 IOS 24](#_Toc199588410)

[4.5 Hybridní aplikace 24](#_Toc199588411)

[5 Právní databáze a api 25](#_Toc199588412)

1. [praktická část 26](#_Toc199588413)

[6 Návrh aplikace 27](#_Toc199588414)

[6.1 Analýza požadavků 27](#_Toc199588415)

[6.1.1 Funkční požadavky 27](#_Toc199588416)

[6.1.2 Nefunkční požadavky 28](#_Toc199588417)

[6.1.3 Uživatelské scénáře 29](#_Toc199588418)

[7 návrh řešení 30](#_Toc199588419)

[7.1 Architektura aplikace 30](#_Toc199588420)

[7.2 Zvolený technologický stack 31](#_Toc199588421)

[8 návrh uživatelského rozhraní 32](#_Toc199588422)

[8.1 Skenovat 32](#_Toc199588423)

[8.2 Hledat 33](#_Toc199588424)

[8.3 Detail 33](#_Toc199588425)

[8.4 Nastavení 34](#_Toc199588426)

[8.5 Oblíbené 34](#_Toc199588427)

[8.6 Uživatel 34](#_Toc199588428)

[9 Implementace 35](#_Toc199588429)

[9.1 OCR a extrakce textu 35](#_Toc199588430)

[9.2 Vyhledávání právních norem 35](#_Toc199588431)

[9.3 Volání API 36](#_Toc199588432)

[9.4 Zobrazení textu s označením předpisů 37](#_Toc199588433)

[10 Testování 38](#_Toc199588434)

[10.1 Testované scénáře 38](#_Toc199588435)

[10.1.1 Ukázka testovacích případů 39](#_Toc199588436)

[10.2 Zjištěné problémy 40](#_Toc199588437)

[10.3 Možnosti zlepšení 40](#_Toc199588438)

[Závěr 41](#_Toc199588439)

[Seznam použité literatury 43](#_Toc199588440)

[Seznam obrázků 49](#_Toc199588441)

[Seznam tabulek 50](#_Toc199588442)

[Seznam použitých symbolů a zkratek 51](#_Toc199588443)

[Seznam příloh 52](#_Toc199588444)

Úvod

Impulsem pro vznik této aplikace byla zkušenost z praxe mého otce, který se profesně zabývá oblastí bezpečnosti práce. Při své činnosti často pracuje s dokumenty obsahujícími odkazy na právní normy, jejichž platnost musí ručně ověřovat. Vzhledem k tomu, že mnoho dokumentů je dostupných pouze v tištěné podobě, je nejprve nezbytné je přepsat do digitální podoby, aby s nimi bylo možné dále pracovat a aktualizovat je. Tento časově náročný a opakující se proces lze výrazně zjednodušit využitím moderních technologií.

Cílem této bakalářské práce je navrhnout a implementovat multiplatformní aplikaci, která umožní uživatelům naskenovat dokument, převést jej do digitální podoby a následně automaticky ověřit platnost zmíněných právních norem. Výsledná aplikace poskytuje uživatelsky přívětivé rozhraní jak ve formě progresivní webové aplikace (PWA), tak i jako nativní aplikace pro Android a iOS.

Přínos aplikace spočívá především v úspoře času a snížení administrativní zátěže při práci s právními dokumenty. Je určena pro odborníky i běžné uživatele, kteří potřebují rychle a spolehlivě zjistit aktuálnost právních norem, zejména pokud mají k dispozici pouze fyzické dokumenty.

Pro vývoj aplikace byl zvolen multiplatformní framework Ionic v kombinaci s Angular, který umožňuje efektivní vývoj pro více platforem současně. Pro ověřování platnosti právních předpisů je využito API stránky zakonyprolidi.cz a optického rozpoznávání znaků (OCR) využitím knihovny Tesseract.js, která nabízí dostatečnou přesnost rozpoznání a snadnou integraci do multiplatformních projektů.

Tato bakalářská práce je rozdělena do několika kapitol. V teoretické části jsou nejprve představeny klíčové technologie a vývojová prostředí, používané OCR nástroje a jejich porovnání, stejně jako možnosti vývoje multiplatformních aplikací. Praktická část se zaměřuje na analýzu požadavků, návrh architektury a výběr konkrétních technologií, dále detailně popisuje samotnou implementaci, testování a vyhodnocení výsledků. V závěru práce jsou shrnuty dosažené poznatky, zhodnocen přínos aplikace v praxi a navrženy možnosti dalšího rozvoje.

|  |  |
| --- | --- |
|  | teoRetická část |

Nástroje a vývojová prostředí

Pro vznik multiplatformní mobilní a webové aplikace je nezbytné zvolit vhodné vývojové nástroje a prostředí, které umožní efektivní tvorbu, testování a nasazení aplikace na různé platformy. V této kapitole jsou popsány hlavní technologie, které byly při vývoji aplikace použity. Patří mezi ně nástroje pro vývoj uživatelského rozhraní, běhová prostředí, integrovaná vývojová prostředí (IDE) i platformy umožňující přístup k nativním funkcím zařízení. Výběr jednotlivých nástrojů byl ovlivněn požadavky na multiplatformnost, snadnou integraci s webovými technologiemi a podporu moderního vývoje mobilních aplikací.

Ionic

Ionic je open-source sada nástrojů pro uživatelské rozhraní (UI), která je speciálně navržena pro vytváření mobilních aplikací s využitím webových technologií, jako jsou HTML, CSS a JavaScript nebo TypeScript. Poskytuje sadu předpřipravených komponent uživatelského rozhraní a nástrojů, které napodobují nativní vzhled a chování platforem iOS a Android. [28][31]

Ionic se integruje s populárními front-end frameworky, jako jsou Angular, React a Vue.js, přičemž jej lze využít i samostatně bez přímé závislosti na konkrétním frameworku. Framework se zaměřuje na interakci uživatelského rozhraní (UI) a uživatelské zkušenosti (UX) aplikace, včetně ovládacích prvků UI, interakcí, gest a animací. [28]

Ionic umožňuje vývojářům psát kód pouze jednou a nasadit jej na více platformách, včetně nativních aplikací pro iOS a Android, desktopových aplikací (pomocí Electronu) a webu jako progresivní webové aplikace (PWA). K nativnímu nasazení do obchodů s aplikacemi využívá Capacitor (viz, podkapitola 1.2) nebo Cordova (viz, podkapitola 1.3), nebo běží v prohlížeči jako PWA. Adaptivní stylování zajišťuje, že aplikace vypadají a chovají se nativně na každé platformě. Ionic abstrahuje specifické nuance jednotlivých platforem, což vývojářům umožňuje soustředit se na základní logiku a uživatelské rozhraní aplikace. [17][28][60]

Capacitor

Capacitor je open-source nativní běhové prostředí, které vývojářům usnadňuje vytváření multiplatformních aplikací pro iOS, Android a PWA pomocí webových technologií (JavaScript, HTML, CSS). Byl navržen týmem Ionic jako moderní nástupce frameworku Cordova, který staví na jejich zkušenostech s prostředím Cordova a nabízí vylepšenou alternativu s plným přístupem k nativním systémovým vývojovým nástrojům (SDK). Capacitor přebírá existující webové aplikace a spouští je jako nativní aplikace na různých platformách, přičemž poskytuje rozhraní, které umožňuje vývojářům využívat funkce nativního systému pomocí JavaScriptu. S nativními projekty pro iOS a Android zachází jako se zdrojovými artefakty, což umožňuje konfiguraci a sestavení/testování specifické pro platformu pomocí nativních IDE (Xcode a Android Studio).[15][21][26][28]

Cordova

Apache Cordova je open-source framework pro vývoj mobilních aplikací, který umožňuje vytvářet aplikace pomocí standardních webových technologií, jako jsou HTML5, CSS3 a JavaScript. Umožňuje tvorbu hybridních aplikací tím, že zabalí webovou aplikaci do nativního kontejneru (WebView), který zajišťuje zobrazení uživatelského rozhraní a komunikaci s nativními funkcemi zařízení prostřednictvím pluginů. Cordova tak vytváří most mezi webovými technologiemi a nativními funkcemi zařízení pomocí sady JavaScriptových API. [27][30]

Node.js

Node.js je open-source, multiplatformní běhové prostředí pro JavaScript, které umožňuje spouštění JavaScriptového kódu mimo webový prohlížeč. Je navrženo primárně pro vývoj serverových aplikací a podporuje tvorbu škálovatelných a výkonných síťových služeb. Node.js je postaven na JavaScriptovém enginu V8, který vyvíjí společnost Google a který je používán také v prohlížeči Chrome. Díky tomu dosahuje vysokého výkonu při zpracování vstupně-výstupních (I/O) operací. Velkou výhodou Node.js je možnost využívat JavaScript jednotně na straně klienta i serveru, což přispívá k vyšší konzistenci kódu, sdílení logiky a efektivnější práci vývojářských týmů. Ekosystém Node.js zahrnuje rozsáhlou knihovnu modulů dostupných prostřednictvím správce balíčků npm (Node Package Manager), což výrazně urychluje vývoj. [42][65]

Visual Studio Code

Visual Studio Code je bezplatný, lehký, ale výkonný editor zdrojového kódu vyvinutý společností Microsoft, dostupný pro Windows, macOS, Linux a webové prohlížeče. Jeho cílem je poskytnout vývojářům základní nástroje pro rychlý cyklus od psaní kódu přes sestavení až po ladění, čímž nabízí zjednodušené prostředí ve srovnání s plnohodnotnými integrovanými vývojovými prostředími.[16][55]

VS Code přichází s integrovanou podporou JavaScriptu, TypeScriptu a node.js, ale díky rozsáhlému ekosystému rozšíření lze doinstalovat podporu pro většinu běžně používaných programovacích jazyků a nástrojů. VS Code si mezi vývojáři získal výjimečnou popularitu díky své rychlosti, všestrannosti, přizpůsobitelnosti a velkému množství dostupných rozšíření. Je široce využíván jak začátečníky, tak zkušenými profesionály, zejména při vývoji webových a multiplatformních aplikací. [23][56]

Android Studio

Android Studio je oficiální integrované vývojové prostředí speciálně pro vývoj aplikací pro operační systém Android, postavené na platformě IntelliJ IDEA společnosti JetBrains. Poskytuje komplexní sadu nástrojů a funkcí přizpůsobených každé fázi vývoje aplikací pro Android, od návrhu uživatelského rozhraní po psaní kódu, ladění, testování a nasazení aplikací. Android Studio podporuje mnoho programovacích jazyků pro nativní vývoj pro Android, včetně jazyků Java, Kotlin a C++. [6][53]

Hlavním účelem prostředí Android Studio je zvýšit produktivitu vývojářů a zefektivnit proces vytváření vysoce kvalitních nativních aplikací pro Android, které lze publikovat v obchodě Google Play. Nabízí optimalizované prostředí se specifickými nástroji pro Android a vylepšenou integraci s platformou Android a sadou SDK. Android Studio usnadňuje vytváření aplikací, které mohou plně využívat možnosti a funkce operačního systému a zařízení Android. [6][37][48]

Xcode

Xcode je bezplatné integrované vývojové prostředí od společnosti Apple, které slouží jako základní nástroj pro vývoj nativních aplikací napříč všemi platformami Apple, včetně iOS, macOS, watchOS, tvOS a visionOS.Poskytuje komplexní sadu nástrojů, které vývojářům umožňují navrhovat, kódovat, testovat a ladit své aplikace v jednotném prostředí. Mezi klíčové funkce patří prediktivní doplňování kódu, interaktivní náhledy uživatelského rozhraní, výkonný debugger a integrace se správou verzí pomocí Git. Xcode také zahrnuje nástroje jako Interface Builder pro vizuální návrh UI a podporu SwiftUI pro moderní vývoj aplikací. Pro testování jsou k dispozici simulátory různých zařízení a Instruments pro analýzu výkonu. Xcode Cloud usnadňuje kontinuální integraci a distribuci aplikací. [9][41]

optické rozpoznávání znaků

Optické rozpoznávání znaků (OCR) představuje klíčovou technologii, která umožňuje převod textu z obrázků, naskenovaných dokumentů nebo fotografií do strojově čitelné podoby. Tato transformace otevírá široké možnosti pro další zpracování, vyhledávání, editaci a analýzu informací, čímž se OCR stává stále důležitější v mnoha oblastech. Od správy dokumentů a automatizace zadávání dat až po zlepšení přístupnosti a podporu pokročilých aplikací. V současné době existuje rozsáhlá škála OCR řešení, od volně dostupných knihoven až po komplexní cloudové služby a specializované vývojové sady. [51]

Existují různé typy OCR, včetně inteligentního rozpoznávání znaků (ICR) pro ručně psaný text a optického rozpoznávání slov (OWR) zaměřeného na celá slova. Technologie OCR prochází neustálým vývojem. Původní metody, založené na porovnávání tvarů s předdefinovanými šablonami znaků, byly omezené v přesnosti a schopnosti zpracovávat různé fonty a kvalitu obrazu. S nástupem umělé inteligence a strojového učení, zejména technik hlubokého učení​, ​jako jsou konvoluční neuronové sítě (CNN) a rekurentní neuronové sítě (RNN), dosáhla přesnost OCR výrazného zlepšení. Tyto pokročilé modely se učí komplexní vzory přímo z rozsáhlých souborů dat textu a obrázků, což jim umožňuje lépe rozpoznávat variace ve fontech, stylech i ručně psaném textu. [1][4][51]

Níže jsou popsány nejběžnější a nejznámější technologie OCR, které byly v rámci této práce zvažovány. Kromě těchto řešení však existuje i celá řada dalších nástrojů a platforem zaměřených na optické rozpoznávání znaků, například ABBYY FineReader, Engine Readiris, OCRopus či aiScript a mnoho dalších, které se liší svými funkcemi, mírou přesnosti i možnostmi integrace. [2]

Amazon Textract

Amazon Textract je plně spravovaná služba od Amazon Web Services. Hluboká integrace se strojovým učením mu umožňuje překonat jiné OCR tím, že rozumí rozvržení dokumentu, identifikuje vztahy mezi textovými prvky a extrahuje strukturované informace, jako jsou páry klíč-hodnota a tabulková data, z komplexních dokumentů, jako jsou faktury, účtenky a průkazy totožnosti. [4]

Dostupnost synchronních i asynchronních možností zpracování v Textractu ukazuje jeho přizpůsobivost široké škále aplikačních potřeb a požadavků na výkon. Synchronní zpracování je navrženo pro jednostránkové dokumenty, kde je kritická nízká latence, a poskytuje okamžité výsledky. Asynchronní zpracování rozšiřuje podporu na vícestránkové dokumenty a umožňuje analýzu velkých dokumentů na pozadí bez blokování toku aplikace. [4]

Apple Vision Framework

Apple Vision Framework je výkonný framework dostupný na platformách iOS, macOS a tvOS, který poskytuje vývojářům širokou škálu nástrojů a API pro provádění různých úloh počítačového vidění, s důrazem na analýzu a zpracování obrazu, včetně robustních funkcí OCR. Kromě OCR nabízí framework rozmanité možnosti, jako je detekce a rozpoznávání obličejů, detekce a sledování objektů, odhadování polohy lidského a zvířecího těla, registrace obrazu a výpočet estetického hodnocení obrazu. [11]

Vision Framework je navržen tak, aby prováděl úlohy počítačového vidění přímo na zařízeních Apple, přičemž využívá hardware a software zařízení pro zvýšení výkonu a zajištění soukromí uživatelů tím, že citlivá data zůstávají na zařízení. Framework podporuje rozpoznávání textu ve značném počtu jazyků, což jej činí vhodným pro internacionalizované aplikace. [10][11]

EasyOCR

EasyOCR je uživatelsky přívětivá open-source Python knihovna speciálně navržená pro OCR, která se může pochlubit podporou více než 80 jazyků, včetně češtiny, což z ní činí všestranný nástroj pro extrakci textu z obrázků v mnoha jazycích. Mezi klíčové vlastnosti EasyOCR patří jednoduchost použití, vyžadující minimální kód pro implementaci funkcí OCR, a relativně vysoká přesnost při rozpoznávání textu z různých obrazových zdrojů. Open-source povaha EasyOCR spolu s jeho implementací v Pythonu významně snižuje vstupní bariéru pro vývojáře a výzkumníky, kteří chtějí začlenit funkce OCR do svých projektů, zejména v komunitách datové vědy a strojového učení, kde je Python široce používaným jazykem. [29][47]

Google ML Kit

Google ML Kit je komplexní sada snadno použitelných nástrojů a API pro strojové učení, speciálně navržená pro vývojáře mobilních aplikací (cílí na platformy Android i iOS), která jim umožňuje snadno integrovat výkonné funkce ML, včetně vysoce efektivních funkcí OCR, do svých mobilních aplikací. ML Kit nabízí vývojářům flexibilitu volby mezi cloudovým zpracováním OCR, které využívá výkonnou cloudovou infrastrukturu Google pro potenciálně vyšší přesnost a podporu širší škály jazyků, a zpracováním OCR na zařízení, které nabízí výhody rychlosti, funkčnosti offline a zvýšeného soukromí dat zpracováním obrázků přímo na zařízení uživatele. [22]

Microsoft Azure Computer Vision

Microsoft Azure Computer Vision je komplexní sada cloudových služeb umělé inteligence, která nabízí širokou škálu funkcí pro analýzu vizuálního obsahu, se silným důrazem na OCR. Umožňuje přesnou extrakci tištěného i ručně psaného textu z různých formátů obrázků a dokumentů, přičemž podporuje mnoho jazyků, což z ní činí všestranné řešení pro globální aplikace. [40]

Azure rozlišuje mezi dvěma OCR rozhraními: standardním OCR API, které je vhodné pro rychlé synchronní zpracování jednodušších scénářů (např. text z dopravních značek nebo etiket), a Read API, které je speciálně navrženo pro extrakci textu z fotografií a souborů dokumentů, včetně běžně používaných vícestránkových formátů, s využitím modelů hlubokého učení, které zajišťují vysokou přesnost bez ohledu na typ textového povrchu či pozadí.[38][39]

Tesseract

Tesseract je vysoce ceněný open-source engine pro OCR, který je široce uznáván pro svou přesnost a rozsáhlou jazykovou podporu, což z něj činí oblíbenou volbu pro různé OCR aplikace a výzkumné projekty. V současné době je vyvíjen a udržován společností Google, což zajišťuje jeho další vývoj a vylepšování. [54]

Jeho všestrannost je patrná z toho, že se často používá nejen jako samostatný OCR engine, ale také jako základní komponenta při vývoji dalších OCR aplikací, nástrojů a knihoven. K dispozici je široká škála trénovaných datových souborů pro mnoho jazyků, které podporují jak starší OCR engine, tak novější a přesnější engine založený na LSTM (Long Short-Term Memory). Jeho neustálý vývoj a aktivní podpora komunity přispívají k jeho průběžnému vylepšování a zajišťují, že zůstává cenným zdrojem pro akademické i komerční aplikace vyžadující vysoce kvalitní extrakci textu. [54]

Scanbot SDK

Scanbot SDK je specializovaná softwarová vývojová sada (SDK) pečlivě navržená pro bezproblémovou integraci funkcí skenování dokumentů a OCR do mobilních aplikací běžících na platformách iOS i Android. Nabízí sadu pokročilých funkcí speciálně přizpůsobených pro práci s dokumenty na mobilních zařízeních, včetně přesné detekce okrajů dokumentů, automatické korekce perspektivy pro narovnání zkosených skenů a vysoce kvalitních funkcí OCR optimalizovaných pro obrázky zachycené mobilními zařízeními. Scanbot SDK podporuje rozsáhlou škálu jazyků, a klíčovou vlastností je jeho schopnost fungovat zcela offline na zařízení koncového uživatele, což zajišťuje přísné soukromí a bezpečnost dat. SDK je kompatibilní s nativním vývojem pro iOS a Android, stejně jako s běžnými hybridními vývojovými platformami, což vývojářům poskytuje flexibilitu při vývoji. [49][50]

Programovací jazyky a Framework

Angular

Angular je webový framework, který umožňuje vývojářům vytvářet rychlé a spolehlivé aplikace. Je postaven na jazyce TypeScript a je aktivně vyvíjen a udržován společností Google. Angular poskytuje rozsáhlou sadu nástrojů, aplikačních programových rozhraní a knihoven, které zjednodušují a zefektivňují proces vývoje webových aplikací. Tento framework je navržen pro budování takzvaných single-page aplikací, kde se veškerý kód potřebný pro běh aplikace načte do prohlížeče při prvním přístupu uživatele a následné interakce probíhají dynamicky bez nutnosti opětovného načítání celé stránky. [7][33]

Komponentově orientovaný model Angularu představuje základní stavební kámen jeho architektury. Tato koncepce umožňuje rozdělit složité uživatelské rozhraní na menší, nezávislé a opakovaně použitelné části, známé jako komponenty. Každá komponenta zapouzdřuje svou vlastní logiku a definuje, jak bude daná část uživatelského rozhraní vypadat a fungovat. Tento přístup významně usnadňuje vývoj, testování a údržbu aplikací a podporuje lepší organizaci kódu a celkovou škálovatelnost projektu. [7][57]

Angular je primárně určen pro vývoj dynamických a interaktivních uživatelských rozhraní webových aplikací, také se často využívá pro vývoj progresivních webových aplikací. [18]

Javascript

JavaScript je dynamický, interpretovaný programovací jazyk s podporou objektově orientovaného, imperativního a funkcionálního programování. Původně byl navržen pro přidání interaktivity na webové stránky, ale díky běhovému prostředí Node.js se rozšířil i na serverovou stranu vývoje. JavaScript je jedním ze tří základních pilířů World Wide Webu, vedle HTML pro strukturu a CSS pro stylizaci. [34][35][62]

Na straně klienta je JavaScript nezastupitelný pro vytváření interaktivních uživatelských rozhraní. Umožňuje manipulovat s Document Object Model pro dynamickou změnu obsahu a vzhledu webových stránek, reagovat na uživatelské události (např. kliknutí myši, stisknutí klávesy), provádět asynchronní požadavky na server a vytvářet vizuální efekty a animace. [5]

S příchodem běhového prostředí Node.js získal JavaScript významné postavení i na straně serveru. Node.js umožňuje vývojářům používat JavaScript k vytváření webových serverů, API, serverové logiky a real-time aplikací. [5]

Typescript

TypeScript je open-source programovací jazyk vyvinutý společností Microsoft. Je to silně typovaný jazyk, který staví na JavaScriptu a rozšiřuje jeho možnosti přidáním statického typování. TypeScript je nadmnožinou JavaScriptu, což znamená, že jakýkoli platný JavaScript kód je zároveň platným TypeScript kódem. TypeScript přidává syntaxi pro explicitní definování typů proměnných, parametrů funkcí a návratových hodnot. [64]

Před spuštěním se kód v TypeScriptu překládá do čistého JavaScriptu pomocí kompilátoru TypeScript, aby mohl být spuštěn ve webových prohlížečích nebo v běhových prostředích jako Node.js. Hlavním cílem TypeScriptu je zlepšit proces vývoje velkých a komplexních JavaScriptových aplikací tím, že poskytuje lepší nástroje pro odhalování chyb a usnadňuje údržbu kódu díky statickému typování. [64][24]

HTML

HyperText Markup Language je standardní značkovací jazyk, který se používá k vytváření a strukturování obsahu webových stránek. HTML dokument se skládá z řady elementů, které jsou definovány pomocí značek a atributů. Tyto značky říkají webovým prohlížečům, jak mají zobrazit text, obrázky, odkazy a další multimediální prvky na webové stránce. [25][59]

HTML není programovací jazyk, ale spíše jazyk pro anotaci textu, který prohlížeči sděluje, jak má daný obsah interpretovat a zobrazit, poskytuje sémantický význam obsahu, což pomáhá prohlížečům, vyhledávačům a asistenčním technologiím lépe porozumět struktuře a důležitosti různých částí stránky. [20]

CSS

Cascading Style Sheets je jazyk pro stylování, který se používá k popisu prezentace dokumentu napsaného v HTML nebo XML. CSS umožňuje oddělit obsah webové stránky od jejího vizuálního vzhledu, včetně rozvržení, zarovnání, barev a písem. CSS se skládá ze souboru pravidel, kde každé pravidlo definuje, jak by měl být určitý HTML element zobrazen v prohlížeči, což umožňuje vývojářům přesně kontrolovat vzhled a rozvržení webových stránek a zajišťovat tak konzistentní uživatelský zážitek. [19][32][58]

Java

Java je víceplatformový, objektově orientovaný programovací jazyk. Je navržen tak, aby umožnil vývojářům napsat kód jednou a spouštět ho kdekoli, kde je podporována Java Virtual Machine (JVM). Java se kompiluje do bytecode, který je interpretován JVM, což zajišťuje nezávislost na hardwaru. Java je široce používána pro vývoj rozsáhlých webových aplikací a webových služeb a byla dlouhou dobu primárním jazykem pro vývoj nativních aplikací pro platformu Android. [36][61]

Swift

Swift je moderní, výkonný a intuitivní programovací jazyk vyvinutý společností Apple pro vývoj aplikací na všech jejích platformách. Je navržen tak, aby byl bezpečný, rychlý a snadno se učil, s expresivní a stručnou syntaxí. Swift je open-source a podporuje více paradigmat programování, včetně protokolově orientovaného, objektově orientovaného a funkcionálního programování. Byl představen v roce 2014 jako nástupce jazyka Objective-C pro vývoj v ekosystému Apple. [8][52]

Mobilní aplikace

Mobilní aplikace jsou softwarové programy navržené pro použití na mobilních zařízeních, jako jsou chytré telefony a tablety. Jsou optimalizovány pro dotykové ovládání a menší obrazovky mobilních zařízení. [44][63]

Webová aplikace

Webová aplikace je softwarový program, ke kterému uživatelé přistupují prostřednictvím webového prohlížeče. Na rozdíl od mobilních aplikací, webové aplikace nevyžadují instalaci na zařízení uživatele, místo toho běží na webových serverech. Jednou z klíčových vlastností webových aplikací je nezávislost na konkrétním operačním systému. K webové aplikaci lze obvykle přistupovat z jakéhokoli zařízení s webovým prohlížečem a připojením k internetu. Nevýhodou webových aplikací je nemožnost využití přístupu ke všem hardwarovým a softwarovým funkcím zařízení, jako fotoaparátu, GPS modulu, a dalších. [3][63]

Nativní Aplikace

Nativní aplikace je typ softwaru, který je vyvinut speciálně pro konkrétní operační systém mobilního zařízení, jako je Android nebo iOS. Tyto aplikace jsou navrženy tak, aby plně využívaly specifické funkce a hardware daného zařízení. Díky této optimalizaci obvykle nabízejí vysoký výkon, rychlou odezvu a plynulý uživatelský zážitek. Nicméně, klíčovou charakteristikou nativních aplikací je, že je nutné vyvíjet samostatné verze pro každý operační systém, což znamená, že aplikace napsaná pro Android nebude fungovat na iOS a naopak. [14][63]

Android

Nativní aplikace pro operační systém Android jsou vyvíjeny s využitím programovacích jazyků, jako je Java nebo stále populárnější Kotlin. Tyto aplikace jsou uživatelům dostupné ke stažení a instalaci z oficiálního obchodu Google Play. Společnost Google poskytuje vývojářům vývojové prostředí Android Studio, které obsahuje veškeré potřebné nástroje pro návrh, vývoj, testování a ladění nativních aplikací pro Android. [43][46]

IOS

Nativní aplikace pro operační systém iOS, který pohání zařízení společnosti Apple jako iPhone a iPad, jsou primárně vyvíjeny v programovacích jazycích Swift a Objective-C. Tyto aplikace jsou dostupné ke stažení prostřednictvím obchodu Apple App Store. Společnost Apple poskytuje vývojářům integrované vývojové prostředí Xcode, které obsahuje veškeré potřebné nástroje pro návrh, vývoj, testování a ladění nativních aplikací pro iOS. Apple klade velký důraz na kvalitu a uživatelský zážitek svých produktů, což se odráží i v přísných pravidlech a požadavcích na vývoj aplikací pro iOS. [12][43][46][66]

Hybridní aplikace

Hybridní aplikace představuje zajímavou kombinaci prvků nativních a webových aplikací. Tyto aplikace jsou vyvíjeny s využitím standardních webových technologií, jako jsou HTML pro strukturu, CSS pro vizuální styl a JavaScript pro interaktivitu. Následně je tento webový kód „zabalen“ do nativního kontejneru, což umožňuje aplikaci běžet na různých mobilních platformách, jako je Android a iOS, z jediné kódové báze. Tento přístup nabízí vývojářům možnost vytvořit jednu aplikaci, která může oslovit širší publikum bez nutnosti psát samostatný kód pro každou platformu zvlášť. Nicméně, je důležité poznamenat, že výkon hybridních aplikací může být ve srovnání s čistě nativními aplikacemi mírně nižší. [13][63]

Právní databáze a api

Pro ověřování platnosti právních předpisů byly zvažovány dvě veřejně dostupné právní databáze s možností programového přístupu prostřednictvím API e-Sbírka a Zákony pro lidi.

Pro potřeby této práce byla zvolena databáze Zákony pro lidi, a to především z důvodu jednodušších požadavků na přístup k API a přehlednější struktury vracených dat ve formátu JSON. Tato služba nabízí snadno využitelné rozhraní pro vyhledávání a ověřování právních norem, což výrazně usnadňuje integraci do vyvíjené mobilní aplikace. [67]

|  |  |
| --- | --- |
|  | praktická část |

Návrh aplikace

Před samotnou samotným vývojem je nezbytné provést návrh aplikace, který vychází z analýzy uživatelských potřeb a technických možností dostupných nástrojů.

V rámci návrhu aplikace je třeba specifikovat funkční a nefunkční požadavky, navrhnout architekturu systému, zvolit vhodný technologický stack a definovat jednotlivé komponenty, které budou zajišťovat klíčové funkcionality, jako jsou rozpoznávání textu, detekce zákonů, práce s API a vizualizace výsledků. Součástí návrhu je rovněž struktura uživatelského rozhraní a popis typických uživatelských scénářů, které demonstrují, jak bude aplikace v praxi používána.

Cílem této bakalářské práce je vyvinout multiplatformní mobilní a webovou aplikaci, která uživatelům umožní snadno ověřovat platnost právních předpisů uvedených v dokumentech. Aplikace využívá technologii optického rozpoznávání znaků (OCR) k převodu tištěného nebo naskenovaného textu do digitální podoby a následně prostřednictvím API databáze zakonyprolidi.cz ověřuje, zda jsou zmíněné právní normy platné.

Analýza požadavků

Cílem této kapitoly je identifikovat a specifikovat klíčové požadavky na vyvíjenou aplikaci. Tyto požadavky jsou rozděleny na funkční, nefunkční a doplněny o ukázkové uživatelské scénáře, které ilustrují typické případy použití aplikace.

Funkční požadavky

Funkční požadavky definují konkrétní chování a schopnosti systému, které mají být implementovány.

* **Načtení vstupního souboru** – aplikace umožní uživateli nahrát dokument ve formátu .txt, .docx nebo fotografii ve formátu .jpg, .png, případně naskenovat pomocí fotoaparátu zařízení.
* **Extrahování textu pomocí OCR** – z nahraného obrázku je pomocí OCR extrahován text.
* **Vyhledání právních předpisů v textu** – aplikace automaticky identifikuje číselná označení právních předpisů.
* **Načtení informací z API** – aplikace odešle dotaz na API databáze zakonyprolidi.cz a získá informace o konkrétním předpise.
* **Zobrazení výsledků** – aplikace zobrazí platnost nalezených předpisů a umožní uživateli zobrazit jejich detailní znění.
* **Označení platnosti normy** – jednotlivé předpisy jsou vizuálně odlišeny podle aktuální platnosti (např. zelená – platný, červená – neplatný).

Nefunkční požadavky

Nefunkční požadavky specifikují obecné charakteristiky systému, které se netýkají přímo funkcionality.

* **Výkon** – rozpoznání textu a načtení dat z API by mělo probíhat v přijatelném čase, ideálně do několika sekund.
* **Kompatibilita** – aplikace má být použitelná na platformách Android, iOS a webu, v různých rozlišeních a na zařízeních.
* **Uživatelská přívětivost** – rozhraní aplikace musí být intuitivní, přehledné a přizpůsobené dotykovému ovládání.
* **Spolehlivost** – aplikace musí správně reagovat i na neúplné nebo chybné vstupy a poskytovat zpětnou vazbu uživateli.
* **Bezpečnost** – přenos dat z/do API musí probíhat přes zabezpečené spojení (HTTPS).

Uživatelské scénáře

Následující scénáře popisují typické způsoby interakce uživatele s aplikací.

* **Scénář 1: Rychlá kontrola právního předpisu z tištěného dokumentu**

Uživatel spustí aplikaci, vyfotí část textu obsahující zmínku o zákonu. Aplikace rozpozná text pomocí OCR, najde označení zákona a ověří jeho platnost. Výsledek se zobrazí s barevným označením a odkazem na detailní znění.

* **Scénář 2: Nahrání dokumentu a hromadné ověření předpisů**

Uživatel nahraje celý text dokumentu (např. interní směrnice ve formátu .docx). Aplikace identifikuje všechny zmíněné zákony, provede jejich ověření a uživateli zobrazí přehled platnosti všech nalezených předpisů.

* **Scénář 3: Vyhledání konkrétního zákona**

Uživatel přejde na záložku „Vyhledávání“, zadá číselné označení zákona (např. „262/2006 Sb.“) a aplikace mu vrátí informaci o platnosti a výpis textu.

návrh řešení

Tato kapitola se věnuje návrhu architektury systému a volbě konkrétních technologií použitých při vývoji aplikace. Hlavním cílem bylo navrhnout takové multiplatformní řešení, které zajistí uživatelům pohodlný a konzistentní přístup jak nativně v mobilních zařízeních, tak prostřednictvím webového prohlížeče.

Architektura aplikace

Aplikace je navržena jako klientská, s architekturou založenou na vzdáleném zpracování dat. Funkce optického rozpoznávání znaků (OCR) jsou realizovány prostřednictvím serverové služby, která běží mimo samotné uživatelské zařízení a zajišťuje potřebný výpočetní výkon. Pro ověřování platnosti právních předpisů aplikace využívá veřejně dostupné REST API databáze zakonyprolidi.cz, které poskytuje aktuální informace o legislativních normách.

**Klíčové komponenty:**

* **Uživatelské rozhraní (UI)** – vytvořené pomocí frameworku Ionic a Angular, obsahuje navigaci v záložkách, formuláře a výpisy.
* **OCR vrstva** – využívá knihovnu Tesseract.js pro extrakci textu z nahraných obrázků.
* **Zpracování textu** – pomocí regulárních výrazů se z textu extrahují číselná označení zákonů.
* **Ověření platnosti** – komunikace s API serveru zakonyprolidi.cz vrací informace o aktuálním stavu předpisů.
* **Vizualizace výsledků** – výsledky jsou zobrazeny s barevným označením.

Zvolený technologický stack

Zvolený technologický stack popisuje tabulka níže.

Tabulka Technologický stack

|  |  |
| --- | --- |
| **Technologie** | **Účel** |
| Ionic + Angular | Vývoj multiplatformní aplikace (Android, iOS, PWA) |
| TypeScript | Hlavní programovací jazyk aplikace (Angular, business logika) |
| JavaScript | Podpora některých knihoven a běhového prostředí |
| HTML/CSS | Tvorba uživatelského rozhraní (struktura, stylování) |
| Tesseract.js | OCR engine pro převod textu z obrázků |
| Regex (JavaScript) | Vyhledávání čísel zákonů v textu |
| API zakonyprolidi.cz | Získávání informací o právních předpisech |
| Capacitor | Přístup k nativním funkcím zařízení (kamera apod.) |

(zdroj: vlastní)

návrh uživatelského rozhraní

Aplikace je rozdělena do několika hlavních záložek.

Skenovat

Stránka pro nahrávání obrázků nebo přímé skenování.

|  |  |
| --- | --- |
| Obrázek 1, Obrázek  Obrázek Stránka sken (web)  (zdroj: vlastní) | Obrázek 1, Obrázek  Obrázek Stránka sken (native)  (zdroj: vlastní) |

Hledat

Stránka s ručním vyhledáváním konkrétních právních norem.

Detail

Stránka s podrobnostmi a celým zněním předpisu.

|  |  |
| --- | --- |
| Obsah obrázku text, elektronika, snímek obrazovky, software  Obsah vygenerovaný umělou inteligencí může být nesprávný., Obrázek  Obrázek Stránka hledat  (zdroj: vlastní) | Obsah obrázku text, snímek obrazovky, software  Obsah vygenerovaný umělou inteligencí může být nesprávný., Obrázek  Obrázek Stránka detail  (zdroj: vlastní) |

Nastavení

Stránka s nastavením aplikace.

Oblíbené

Stránka pro zobrazení uživatelem uložených právních norem, s možností offline přístupu (Funkcionalita není v aktuální verzi implementována, plánováno jako budoucí rozšíření).

Uživatel

Stránka pro přihlášení a správu uživatelského účtu (Funkcionalita není v aktuální verzi implementována, plánováno jako budoucí rozšíření).

|  |  |
| --- | --- |
| Obrázek 1, Obrázek  Obrázek Stránka nastavení  (zdroj: vlastní) | Obsah obrázku text, snímek obrazovky, software, Operační systém  Obsah vygenerovaný umělou inteligencí může být nesprávný., Obrázek  Obrázek Stránka uživatel  (zdroj: vlastní) |

Implementace

Tato část se věnuje konkrétním aspektům vývoje aplikace. Soustředí se na řešení jednotlivých problémových oblastí a popis použitých technik.

OCR a extrakce textu

Pro extrakci textu z obrázků je využívána serverová instance Tesseract OCR.

Obrazová data jsou z aplikace odeslána na vzdálený server, kde probíhá samotné rozpoznání textu. Tento přístup přináší výhodu vyššího výkonu a přesnosti díky větší výpočetní kapacitě.

Odesílání obrázku na server probíhá pomocí HTTP požadavku, který následně vrací výsledek ve formátu prostého textu.

Vyhledávání právních norem

Pro hledání čísel norem se používají regulární výrazy.

Ukázka kódu proměnné s regulárním výrazem:

// Regex pro nalezení čísla zákonu ("123/2025")

private readonly numberRegex = /\d{1,4}\/\d{2,4}/g;

Volání API

Pro ověření platnosti normy aplikace komunikuje se serverem zakonyprolidi.cz. API vrací informace o dané normě ve formátu JSON, včetně data účinnosti a platnosti.

Ukázka API responce ve formátu JSON:

{

  "Version": "2.x",

  "Base": "https://www.zakonyprolidi.cz",

  "Result": {

    "Batch": "84",

    "Code": "2006-262",

    "Collection": "cs",

    "DeclareDate": "/Date(1145570400000+0200)/",

    "DocId": 3053218,

    "DocType": 4,

    "EffectFrom": "/Date(1167606000000+0100)/",

    "EffectTill": null,

    "File": "2006c084z0262",

    "Href": "/cs/2006-262",

    "LastUpdate": "/Date(1747349184383+0200)/",

    "Number": "262",

    "PublishDate": "/Date(1149631200000+0200)/",

    "Quote": "262/2006 Sb.",

    "Short": "Zákoník práce",

    "Title": "Zákon zákoník práce",

    "Year": 2006

  }

}

Zobrazení textu s označením předpisů

Uživatel vidí text s předpisy, kde jsou záznamy vizuálně označeny.

* **Zelená** – norma je platná a účinná.
* **Červená** – norma byla zrušena, není účinná.
* **Šedá** – norma byla rozpoznána, ale nebylo možné ověřit platnost (norma neexistuje, překlep, chyba API response, atd.).

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Obsah vygenerovaný umělou inteligencí může být nesprávný., Obrázek

Obrázek Označení předpisů

(zdroj: vlastní)

Testování

Tato kapitola se věnuje ověření správnosti a funkčnosti vyvinuté aplikace. Testování probíhalo manuálně na různých typech zařízení a pro různé vstupní scénáře. Cílem testování bylo zjistit, zda aplikace správně detekuje text, rozpoznává odkazy na právní předpisy, komunikuje s API a zobrazuje očekávané výsledky.

Testované scénáře

Pro testování byly použity:

* 4 dokumenty z reálné praxe o délce 68000 až 85000 znaků a kolem 80 právních norem, kde aplikace nerozpozná 1 až 2 normy s překlepy
* 7 skenů reálných dokumentů, kde aplikace správně rozpozná text i normy
* 30+ fotografií kde aplikace rozpozná text podle kvality fotografie (rozmazání, kontrast, ostrost).

Ukázka testovacích případů

Ukázka testovacích případů pokrývající klíčové funkcionality aplikace.

Tabulka Ukázka testovacích případů

|  |  |  |  |
| --- | --- | --- | --- |
| **Scénář** | **Vstup** | **Očekávaný výstup** | **Výsledek** |
| Nahrání obrázku s právním předpisem | Fotografie | Aplikace text rozpozná, normy rozpozná a správně označí | OK |
| Pořízení fotografie s nízkým kontrastem | Fotografie | Text je částečně rozpoznán, obsahuje znaky navíc, některé znaky chybí | Problém |
| Ruční vyhledávání | Zadání „262/2006“ do vyhledávače | Aplikace zobrazí správný předpis a data platnosti | OK |
| Nahrání dokumentu obsahující více zákonů | Dokument obsahující 83 norem | Většina norem rozpoznána, většina správně barevně označena | S nedostatky |
| Nahrání dokumentu obsahující více zákonů | Dokument obsahující 76 norem | Všechny normy rozpoznány a správně barevně označeny | OK |

(zdroj: vlastní)

Zjištěné problémy

**OCR selhává při horší kvalitě obrazu** – rozmazané fotografie, nízká světelnost, barevný text nebo ručně psaný text vedou k nepřesnostem v rozpoznávání. Tento problém byl řešen ruční úpravou v nativní aplikaci fotoaparátu (úprava jasu, kontrastu, zaostření) nebo pořízením nové fotografie.

**Regulární výraz je náchylný na netypické formáty a překlepy** – některé neobvyklé zápisy zákonů nejsou správně detekovány.

ukázka překlepů z testovaných dat: „NV č. 291/2015 5Sb“, „Vyhl,.č. 219/2004Sb“

**API vrací chybu v datech** – v některých případech API vrací chybný formát času.

ukázka chybného formátu: „/Date(-139456800000+0200)/“

Možnosti zlepšení

Na základě testování byly identifikovány oblasti pro budoucí zlepšení aplikace.

* Implementace odkazů pro snadné přesměrování v textech na stránce předpisu.
* Implementace předzpracování obrázků pro zlepšení přesnosti OCR.
* Implementace OCR s vyšší přesností správného rozpoznání znaků.
* Optimalizace regulárních výrazů pro detekci i s překlepy.
* Implementace našeptávače do vyhledávače.

Závěr

Tato bakalářská práce se zabývá návrhem a vývojem multiplatformní aplikace pro ověřování platnosti právních předpisů s využitím technologie optického rozpoznávání znaků (OCR) a veřejně dostupného API portálu Zákony pro lidi. V rámci práce jsou popsány klíčové nástroje a technologie (včetně frameworků pro vývoj multiplatformních aplikací, různých OCR řešení a právních databází), následně je detailní návrh uživatelského rozhraní, architektura systému, implementace klíčových funkcí a testování vyvinuté aplikace.

Výsledná aplikace umožňuje uživatelům nahrávat právní dokumenty, extrahovat z nich text a získat přehled o platnosti jednotlivých právních předpisů. Přestože se během vývoje a testování objevily některé problémy – zejména v oblasti přesnosti OCR a formátování dat – podařilo se vytvořit funkční prototyp, který potvrzuje potenciál automatizace tohoto typu činností.

Hlavním přínosem aplikace je zjednodušení a urychlení ověřování právních norem, možnost práce s papírovými i digitálními dokumenty a univerzální dostupnost napříč různými platformami, což v praxi přináší úsporu času i minimalizaci chyb způsobených ručním přepisem či vyhledáváním.

Do budoucna by bylo vhodné aplikaci dále rozvíjet, například vylepšením přesnosti OCR (nasazením pokročilejších modelů a předzpracováním obrazových dat před samotným rozpoznáváním), zavedením uživatelských účtů, možností offline režimu integrací offline varianty OCR (například rozšíření o offline Tesseract.js nebo nasazení nativních knihoven Apple Vision Framework či Google ML Kit pro nativní aplikace) a přístupem k uloženým normám či rozšířením o pokročilé nástroje pro analýzu právního textu. Dále se nabízí zlepšení procházení právních norem a vylepšení uživatelského prostředí aplikace.

Vyvinutý prototyp demonstruje potenciál moderních technologií pro digitalizaci a automatizaci právních procesů. Další vývoj a rozšíření aplikace může přinést užitek nejen odborníkům, ale i běžným uživatelům, kteří potřebují rychle a spolehlivě ověřit platnost právních předpisů uvedených v jakémkoli dokumentu.

Seznam použité literatury

1. ADAPTECH s.r.o. *Co je to OCR?* Online. Dostupné z: <https://www.adaptech.cz/news/co-je-to-ocr>. [cit. 2025-04-22].
2. AI GEN. *7 Most Effective OCR Software in 2024 (Free & Paid)* Online. Dostupné z: <https://aigencorp.com/en/most-effective-ocr-software/>. [cit. 2025-04-22].
3. AMAZON WEB SERVICES. *What is a Web App?* Online. Dostupné z: <https://aws.amazon.com/what-is/web-application/>. [cit. 2025-04-22].
4. AMAZON WEB SERVICES. *What is Amazon Textract? - Amazon Textract - AWS Documentation* Online. Dostupné z: <https://docs.aws.amazon.com/textract/latest/dg/what-is.html>. [cit. 2025-04-22].
5. AMAZON WEB SERVICES. *What is Javascript (JS)?* Online. Dostupné z: <https://aws.amazon.com/what-is/javascript/>. [cit. 2025-04-22].
6. ANDROID DEVELOPERS. *Meet Android Studio* Online. Dostupné z: <https://developer.android.com/studio/intro>. [cit. 2025-04-22].
7. ANGULAR. *What is Angular?* Online. Dostupné z: <https://angular.dev/overview>. [cit. 2025-04-22].
8. APPLE DEVELOPER. *Swift* Online. Dostupné z: <https://developer.apple.com/swift/>. [cit. 2025-04-22].
9. APPLE DEVELOPER. *Xcode* Online. Dostupné z: <https://developer.apple.com/xcode/>. [cit. 2025-04-22].
10. APPLE. *Recognizing Text in Images | Apple Developer Documentation* Online. Dostupné z: <https://developer.apple.com/documentation/vision/recognizing-text-in-images>. [cit. 2025-04-22].
11. APPLE. *Vision | Apple Developer Documentation* Online. Dostupné z: <https://developer.apple.com/documentation/vision/>. [cit. 2025-04-22].
12. APPMYSITE. *Native mobile app development: Here's all you need to know* Online. Dostupné z: <https://www.appmysite.com/blog/complete-guide-to-native-mobile-app-development/>. [cit. 2025-04-22].
13. APPSFLYER. *Hybrid app | AppsFlyer Mobile Marketing Glossary* Online. Dostupné z: <https://www.appsflyer.com/glossary/hybrid-app/>. [cit. 2025-04-22].
14. BUILDER.AI. *Native app definition: Examples, hybrid app comparison & more* Online. Dostupné z: <https://www.builder.ai/glossary/native-app>. [cit. 2025-04-22].
15. CAPACITORJS.COM. *Capacitor – Documentation* Online. Dostupné z: <https://capacitorjs.com/docs>. [cit. 2025-04-22].
16. CODE.VISUALSTUDIO.COM. *Visual Studio Code FAQ* Online. Dostupné z: <https://code.visualstudio.com/docs/supporting/FAQ>. [cit. 2025-04-22].
17. DAILY.DEV. *Ionic Framework Basics for Beginners* Online. Dostupné z: <https://daily.dev/blog/ionic-framework-basics-for-beginners>. [cit. 2025-04-22].
18. FUSIONCHARTS. *Beginners Guide On Angular Frameworks* Online. Dostupné z: <https://www.fusioncharts.com/blog/beginners-guide-on-angular-frameworks/>. [cit. 2025-04-22].
19. GEEKSFORGEEKS. *CSS Introduction* Online. Dostupné z: <https://www.geeksforgeeks.org/css-introduction/>. [cit. 2025-04-22].
20. GEEKSFORGEEKS. *HTML Introduction* Online. Dostupné z: <https://www.geeksforgeeks.org/html-introduction/>. [cit. 2025-04-22].
21. GITHUB. *ionic-team/capacitor: Build cross-platform Native Progressive Web Apps for iOS, Android, and the Web ⚡️* Online. Dostupné z: <https://github.com/ionic-team/capacitor>. [cit. 2025-04-22].
22. GOOGLE. *Recognize text in images with ML Kit on Android - Google for Developers* Online. Dostupné z: <https://developers.google.com/ml-kit/vision/text-recognition/v2/android>. [cit. 2025-04-22].
23. HOSTINGER. *What is VS Code: an overview of the popular code editor and its features* Online. Dostupné z: <https://www.hostinger.com/tutorials/what-is-vs-code>. [cit. 2025-04-22].
24. INVEDUS OUTSOURCING. *What is TypeScript? Definition, History, Features and Uses* Online. Dostupné z: <https://invedus.com/blog/what-is-typescript-definition-history-features-and-uses-of-typescript/>. [cit. 2025-04-22].
25. INVESTOPEDIA. *HyperText Markup Language (HTML): What It Is and How It Works* Online. Dostupné z: <https://www.investopedia.com/terms/h/html.asp#:~:text=HyperText%20Markup%20Language%20(HTML)%20is,a%20webpage's%20words%20and%20images>. [cit. 2025-04-22].
26. IONIC BLOG. *Capacitor: Everything You've Ever Wanted to Know* Online. Dostupné z: <https://ionic.io/blog/capacitor-everything-youve-ever-wanted-to-know>
27. IONIC.IO. *What is Apache Cordova Framework* and What is the Difference from PhoneGap? Online. Dostupné z: <https://ionic.io/resources/articles/what-is-apache-cordova>. [cit. 2025-04-22].
28. IONICFRAMEWORK.COM. *Introduction to Ionic - Documentation* Online. Dostupné z: <https://ionicframework.com/docs>. [cit. 2025-04-22].
29. JAIDEDAI. *JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc. - GitHub* Online. Dostupné z: <https://github.com/JaidedAI/EasyOCR>. [cit. 2025-04-22].
30. JSCRAMBLER. *Top 12 Frameworks for Hybrid Mobile Apps* Online. Dostupné z: <https://jscrambler.com/blog/12-frameworks-for-mobile-hybrid-apps>. [cit. 2025-04-22].
31. KELLTON. *The Ionic Edge: Exploring the top Ionic features and its benefits* Online. Dostupné z: <https://www.kellton.com/kellton-tech-blog/what-is-ionic-framework-and-its-top-features>. [cit. 2025-04-22].
32. MDN Web Docs. *CSS: Cascading Style Sheets* Online. Dostupné z: <https://developer.mozilla.org/en-US/docs/Web/CSS#:~:text=Cascading%20Style%20Sheets%20(CSS)%20is,speech%2C%20or%20on%20other%20media>.  [cit. 2025-04-22].
33. MDN Web Docs. *Getting started with Angular* Online. Dostupné z: [*https://developer.mozilla.org/en-US/docs/Learn\_web\_development/Core/Frameworks\_libraries/Angular\_getting\_started*](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Frameworks_libraries/Angular_getting_started). [cit. 2025-04-22].
34. *MDN Web Docs. JavaScript* Online. Dostupné z:[*https://developer.mozilla.org/en-US/docs/Web/JavaScript*](https://developer.mozilla.org/en-US/docs/Web/JavaScript). [cit. 2025-04-22].
35. *MDN Web Docs. What is JavaScript?* Online. Dostupné z: [*https://developer.mozilla.org/en-US/docs/Learn\_web\_development/Core/Scripting/What\_is\_JavaScript#:~:text=JavaScript%20is%20a%20scripting%20or,3D%20graphics%2C%20scrolling%20video%20jukeboxes%2C*](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript#:~:text=JavaScript%20is%20a%20scripting%20or,3D%20graphics%2C%20scrolling%20video%20jukeboxes%2C). [cit. 2025-04-22].
36. MICROSOFT AZURE. *What is Java?* Online. Dostupné z: <https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-java-programming-language>. [cit. 2025-04-22].
37. MICROSOFT LEARN. *Get started with native Android development on Windows* Online. Dostupné z: <https://learn.microsoft.com/en-us/windows/android/native-android>. [cit. 2025-04-22].
38. MICROSOFT. *Azure AI Vision with OCR and AI | Microsoft Azure* Online. Dostupné z: <https://azure.microsoft.com/en-us/products/ai-services/ai-vision>. [cit. 2025-04-22].
39. MICROSOFT. *OCR for images - Azure AI Vision* Online. Dostupné z: <https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/concept-ocr>. [cit. 2025-04-22].
40. MICROSOFT. *What is Azure AI Vision? - Learn Microsoft* Online. Dostupné z: <https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview>. [cit. 2025-04-22].
41. NETGURU. *What Is Xcode and How to Use It?* Online. Dostupné z: <https://www.netguru.com/blog/what-is-xcode-and-how-to-use-it>. [cit. 2025-04-22]

Seznam obrázků

[Obrázek 1 Stránka sken (web) 32](#_Toc199588292)

[Obrázek 2 Stránka sken (native) 32](#_Toc199588293)

[Obrázek 3 Stránka hledat 33](#_Toc199588294)

[Obrázek 4 Stránka detail 33](#_Toc199588295)

[Obrázek 5 Stránka nastavení 34](#_Toc199588296)

[Obrázek 6 Stránka uživatel 34](#_Toc199588297)

[Obrázek 7 Označení předpisů 37](#_Toc199588298)

Seznam tabulek

[Tabulka 1 Technologický stack 31](#_Toc199588304)

[Tabulka 2 Ukázka testovacích případů 39](#_Toc199588305)

Seznam použitých symbolů a zkratek

API Application Programming Interface – aplikační programové rozhraní

OCR Optical Character Recognition – optické rozpoznávání znaků

PWA Progressive Web App – progresivní webová aplikace

UI User Interface – uživatelské rozhraní

UX User Experience – uživatelská zkušenost

HTML HyperText Markup Language – značkovací jazyk pro webové stránky

CSS Cascading Style Sheets – kaskádové styly pro webové stránky

JS JavaScript – Programovací jazyk používaný pro tvorbu webových aplikací

TS TypeScript – Nadmnožina jazyka JavaScript s podporou statického typování

VS Code Visual Studio Code – editor zdrojového kódu od Microsoftu

IDE Integrated Development Environment – integrované vývojové prostředí

REST Representational State Transfer – architektura webových API

JSON JavaScript Object Notation – datový formát pro přenos strukturovaných dat

V8 JavaScript engine vyvíjený společností Google

LSTM Long Short-Term Memory – typ neuronové sítě využívaný v OCR

ML Machine Learning – strojové učení

SDK Software Development Kit – sada nástrojů pro vývoj aplikací

JVM Java Virtual Machine – virtuální stroj pro spouštění aplikací v jazyce Java

npm Node Package Manager – správce balíčků pro JavaScript a Node.js

PDF Portable Document Format – formát pro elektronické dokumenty

regex Regular Expression – regulární výraz pro hledání vzorů v textu

iOS Mobilní operační systém společnosti Apple

Android Mobilní operační systém vyvíjený společností Google

Git Distribuovaný systém pro správu verzí

HTTP/HTTPS HyperText Transfer Protocol / Secure – protokol pro přenos dat (šifrovaný)

Seznam příloh

Příloha P I: Zdrojový kód aplikace

Příloha P I: Název první přílohy

Obsah první přílohy: Zdrojový kód aplikace