VYSOKÁ ŠKOLA POLYTECHNICKÁ JIHLAVA

Aplikovaná informatika

Systém pro hlídání dostupnosti elektronických součástek

Bakalářská práce

Autor práce: Michal Zoubek

Vedoucí práce: doc. Ing. Zbyněk Bureš, Ph.D.

Jihlava 2024

Vysoká škola polytechnická Jihlava

Tolstého 16, 586 01 Jihlava

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Autor práce:	**Michal Zoubek**
Studijní program:	Aplikovaná informatika
Obor:	Aplikovaná informatika
Garant studijního programu:	Ing. Lenka Kuklišová Pavelková, Ph.D.
Název práce:	**Systém pro hlídání dostupnosti elektronických součástek**
Vedoucí práce: doc.	Ing. Zbyněk Bureš, Ph.D.
Cíl práce:	Vytvořte systém pro hlídání dostupnosti a nejlepší ceny elektronických součástek na e-shopech. Systém bude periodicky zjišťovat dostupnost a cenu zadaných součástek na vybraných e-shopech. V případě zjištění nové dostupnosti či nalezení výhodnější ceny zobrazí oznámení uživateli. Systém bude rovněž uchovávat databázi vývoje cen a množství jednotlivých součástek na e-shopech a bude možné graficky znázornit historii změn. Systém bude implementován jako lokální desktopová aplikace pro OS Windows.

Abstrakt

Bakalářská práce se zaměřuje na vývoj a implementaci programu pro monitorování dostupnosti a cen elektronických součástek z vybraných e-shopů. Cílem práce je hotový program, který dle zadaných produktů od uživatele sleduje požadované informace o produktech a upozorňuje na případně změny. V teoretické části je detailně popsána problematika, zdroje vstupních dat pro program a následný návrh řešení za použití vybraných technologií. Praktická část je pak věnována implementaci těchto řešení do hotového programu. Jeho vytvoření, instalace a testování funkčnosti.

Klíčová slova

C#, .NET, API, e-shop, reporting

Abstract

This bachelor's thesis focuses on the development and implementation of a program for monitoring the availability and prices of electronic components from selected e-shops. The aim of the thesis is to create a functional program that, based on user-specified products, monitors relevant product information and alerts the user to any changes. The theoretical part provides a detailed description of the issues, sources of input data for the program, and the subsequent solution design using selected technologies.

The practical part is devoted to implementing these solutions into the finished program, including its creation, installation, and functionality testing. The program is designed to track desired information about products and notify users of any changes, contributing to improved user awareness and decision-making when it comes to purchasing electronic components online.

Keywords

C#, .NET, API, e-shop, reporting

Prohlašuji, že předložená bakalářská práce je původní a zpracoval/a jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil/a autorská práva (ve smyslu zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, v platném znění, dále též „**AZ**“).

Byl/a jsem seznámen/a s tím, že na mou bakalářskou práci se plně vztahuje **AZ**, zejména § 60 (školní dílo).

Podle § 47b zákona o vysokých školách souhlasím se zveřejněním své práce podle Směrnice pro vedení, vypracování a zveřejňování závěrečných prací na VŠPJ, a to bez ohledu na výsledek obhajoby.

Beru na vědomí, že VŠPJ má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom/a toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem VŠPJ, která má právo ode mě požadovat přiměřený příspěvek na úhradu nákladů, vynaložených vysokou školou na vytvoření díla (až do jejich skutečné výše), z výdělku dosaženého v souvislosti s užitím díla či poskytnutím licence.

V Jihlavě dne 1. ledna 2024

…….

Podpis studenta/ky

Poděkování

Rád bych poděkoval svému vedoucímu bakalářské práce doc. Ing. Zbyňku Burešovi, Ph.D., za cenné rady, spolupráci a možnost pracovat na této práci pod jeho vedením. Manželce, která mi byla trpělivou oporou a synovi za to, že moc neplakal.

Obsah

Seznam obrázků

Seznam zkratek

- API - Application Programming Interface
- URL - Uniform Resource Locator
- TME - Transfer Multisort Elektronik
- IDE - Integrated Development Environment

Úvod

Bakalářská práce se zaměřuje na vývoj a implementaci systému pro sledování dostupnosti elektronických součástek a monitorování jejich cenového vývoje na vybraných elektronických obchodech. Hlavním cílem je splnění požadavků společnosti RWetc s.r.o., která si klade za úkol zajistit dostatečné množství součástek pro svůj výrobní proces za optimální cenu. Zároveň je třeba sledovat dostupnost nedostatkových součástek.

V rámci práce bude vytvořen program, který poběží na pozadí operačního systému Windows. Na základě uživatelských vstupů bude program automaticky a pravidelně shromažďovat informace, sledovat ceny a jejich historii, a zaznamenávat dostupné skladové zásoby. Dále bude generovat reporty obsahující požadované informace o elektronických součástkách dostupných v elektronických obchodech.

Vytvořený program umožní uživateli přijímat upozornění o změnách ve dostupnosti nebo cenách. Upozornění budou automaticky generována při výrazných změnách, anebo budou poskytována na vyžádání uživatele při jeho vstupu do programu.

1 Teoretická část

V následující části práce bude popsáno zadání problému k řešení a zdroje dat ze kterých se budou čerpat informace pro běh programu.

1.1 Popis problematiky

Moderní výrobní procesy jsou neoddělitelně spojeny s používáním elektronických součástek, které hrají klíčovou roli v konstrukci a fungování elektronických zařízení. Důležitým aspektem pro efektivní a bezproblémový průběh výroby je správné a včasné zajištění potřebných součástek. Pro firmu RWetc s.r.o. se stala strategickou prioritou optimalizace zásobování elektronickými součástkami pro svůj výrobní proces, s důrazem na dosažení nejlepší možné ceny.

Cílem práce je vytvořit a implementovat program, který umožní automatické sledování dostupnosti, cen a vývoje cen elektronických součástek na vybraných e-shopech:

- tme.eu
- mouser.com
- soselectronics.com
- farnell.com

Program bude pracovat na pozadí operačního systému Windows a periodicky sbírat a analyzovat relevantní informace, aby bylo možné efektivně reagovat na změny v cenách a dostupnosti součástek.

1.2 Zdroje vstupních dat

Pro dosažení úspěšného plnění stanovených cílů je nezbytné podrobněji zkoumat zdroje dat, ze kterých bude program čerpat informace. V nadcházející části budou podrobně rozebrány a popsány klíčové e-shopy, které budou monitorovány: tme.eu, mouser.com, soselectronics.com a farnell.com. Pro každý z těchto e-shopů bude analyzováno a zdokumentováno dostupné rozhraní pro získávání dat, jeho funkcionality a možnosti získávání relevantních dat týkajících se dostupnosti a cen elektronických součástek. Tato informace bude sloužit jako základní pilíř pro následný návrh a implementaci programu.

1.2.1 TME.eu (Transfer Multisort Elektronik)

Zde jsou uvedeny informace o e-shopu tme.eu *(Tme, 2023)* a způsobu získávání potřebných dat pro program.

- **URL:** https://www.tme.eu/
- **Způsob získávání informací:** TME.eu poskytuje rozhraní API, které umožňuje externím aplikacím přistupovat k jejich katalogu produktů, cenám a dostupnosti.
- **Funkcionality API** *(Tme API User Manual, 2023)***:**
    - /Product/Search: Vrátí o produktu dle hledaného vzorce.
    - /Product/SearchParameters: Vrátí informace o možném filtru, který lze použít ve funkctionalitě Product/Search.

- /Products/Autocomplete: Vrátí návrh produktu založený na vstupu.
- /Products/GetCategories: Vrátí seznam všech produktových kategorií.
- /Products/GetDeliveryTime: Vrátí předpokládanou dobu doručení produktu.
- /Products/GetPrices: Vrátí výpis cen pro poskytnutý soupis produktů.
- /Products/GetParameters: Vrátí soupis produktových parametrů
- /Products/GetPricesAndStocks: Vrátí soupis cen a dostupného množství dle poskytnutého výpisu produktů.
- /Products/GetProducts: Vrátí základní informace o produktech aktuálně dostupných v nabídce.
- /Products/GetProductsFiles: Vrátí seznam dostupných fotografií a dokumentů souvisejících s produktem.
- /Products/GetStocks: Vrátí soupis dostupného množství produktů dle poskytnutého výpisu produktů.
- /Products/GetSymbols: Vrátí seznam symbolů produktů, které jsou momentálně dostupné.
- /Products/GetSimilarProducts: Vrátí podobné produkty, dle produktu na vstupu.
- /Products/GetRelatedProducts: Vrátí příbuzné produkty, dle produktu na vstupu.
- /Utils/GetCountries: Vrátí seznam zemí, kde je systém podporován.
- /Utils/GetLanguages: Vrátí seznam jazyků, které systém podporuje.
- /Utils/Ping: Vrátí status API rozhraní.

1.2.2 Mouser.com

Zde jsou uvedeny informace o e-shopu mouser.com *(Mouser electronics, 2023)* a způsobu získávání potřebných dat pro program.

- **URL:** https://www.mouser.com/
- **Způsob získávání informací:** Mouser.com má své vlastní API, které umožňuje získávat informace o produktech, cenách a dostupnosti.
- **Funkcionality API** *(Mouser APIs, 2023)***:**
  - GET: /api/v{version}/cart: Vrátí informace o košíku.
  - POST: /api/v{version}/cart: Aktualizuje nákupní košík na e-shopu.
  - POST: /api/v{version}/cart/items/insert: Vloží zboží do košíku.
  - POST: /api/v{version}/cart/items/update: Aktualizuje zboží v košíku.
  - POST: /api/v{version}/cart/remove: Odstraní zboží z košíku.
  - POST: /api/v{version}/cart/insert/schedule: Umožňuje vložit a nastavit dobu doručení položek v košíku.
  - POST: /api/v{version}/cart/update/schedule – Umožní upravit dobu doručení položek v košíku.
  - POST: /api/v{version}/cart/deleteall/schedule – Umožní odstranit všechny nastavené doručovací doby z košíku.
  - POST: /api/v{version}/order/item/CreateCartFromOrder – Vytvoří nový košík z objednávky

- GET: /api/v{version}/orderhistory/ByDateFilter – Vrátí historii objednávek dle poskytnutého filtru.
- GET: /api/v{version}/orderhistory/ByDateFilter – Vrátí historii objednávek dle poskytnutého rozpětí datumů.
- POST: /api/v{version}/order/options/query: Vrátí dostupné způsoby doručení, platební metody, daňové certifikáty, účty a uložené informace o platebních kartách.
- GET: /api/v{version}/order/currencies: Vrátí soupis dostupných měn, dle fakturační adresy
- GET: /api/v{version}/order/countries: Vrátí soupis zemí.
- POST: /api/v{version}/order: Umožňuje vložit novou objednávku, nebo vrací souhrn aktuální objednávky.
- POST: /api/v{version}/order/CreateFromOrder: Vytvoří novou objednávku z historické objednávky, nebo vrátí informace o historické objednávce.
- GET: /api/v{version}/order/{orderNumber}: Vrátí detaily objednávky dle zadaného čísla objednávky.
- POST: /api/v{version}/search/keyword: Umožnuje najít informace o součástkách dle klíčového slova.
- POST: /api/v{version}/search/partnumber: Umožňuje najít informace o produktech dle čísla produktu.

1.2.3 SOS Electronics:

Zde jsou uvedeny informace o e-shopu SOS electronic *(SOS electronic, 2023)* a způsobu získávání potřebných dat pro program.

- **URL:** https://www.soselectronic.com/
- **Způsob získávání informací:** SOS Electronic rovněž nabízí API, které umožňuje automatizovaný přístup k jejich katalogu a informacím o produktech.
- **Funkcionality API** *(SOS version 1.0, 2023)***:**
  - POST: /auth/token/password: Slouží k ověření autenticity klienta.
  - POST: /auth/token/refresh: Slouží k obnovení a aktualizaci přístupového tokenu.
  - POST: /auth/token/tokeninfo: Slouží k získání informací o přístupovém tokenu.
  - POST: /auth/token/revoke: Slouží k odhlášení a odebrání přístupu tokenu.
  - GET: /docs: Slouží k získání dokumentace API.
  - GET: /invoices: Slouží k získání informací o fakturách.
  - GET: /invoices/{invoice_id}: Slouží k získání informací o konkrétní faktuře dle zadaného identifikačního čísla faktury.
  - GET: /orders/open: Vrátí seznam otevřených objednávek.
  - GET: /orders/open/products: Vrátí seznam dosud nedodaných produktů.
  - GET: /orders/{order_id}: Vrátí informace o konkrétní objednávce dle zadaného identifikačního čísla objednávky.
  - GET: /products: Vrátí výpis produktů a informace o nich dle zadaných produktů na vstupu.

- GET: /products/{ordernr}: Vrátí výpis produktů a informace o nich z konkrétní objednávky dle zadaného čísla objednávky.

1.2.4 Farnell.com:

Zde jsou uvedeny informace o e-shopu Farnell *(Farnell, 2023)* a způsobu získávání potřebných dat pro program.

- **URL:** https://farnell.com/
- **Způsob získávání informací:** Také Farnell nabízí možnost získávat informace o jejich produktech pomocí API rozhraní.
- **Funkcionality API** *(Product Search API (REST), 2023)***:**
  - callInfo.responseDataFormat: Určuje výstupní formát.
  - callInfo.omitXmlSchema: Umožňuje odstranit Namespace z XML výstupu.
  - callInfo.callback: Umožňuje použít obalovací funkci na výstupu při použití formát JSON.
  - term: Vstupem je vybraný identifikátor požadovaného produktu. Vrací informace o produktu.
  - storeInfo.id: Určuje region, pro který se mají vybrat lokální data o produktu.
  - callInfo.apiKey: Klíč pro přístup k aplikaci.
  - userInfo.signature: Potvrzuje zákazníka, kvůli smluvním cenám produktů.
  - userInfo.timestamp: Určuje čas, při získávání smluvních cen.
  - userInfo.customerId: Identifikuje zákazníka, kvůli smluvním cenám produktů.
  - resultsSettings.offset: Povinná položka při hledání dle klíčového slova.
  - resultsSettings.numberOfResults: Povinná položka při hledání dle klíčového slova. Upřesňuje počet výsledků
  - resultsSettings.refinements.filters: Určuje doplňující filtry při vstupu.
  - resultsSettings.responseGroup: Upresňuje data která se vrátí na výstupu.

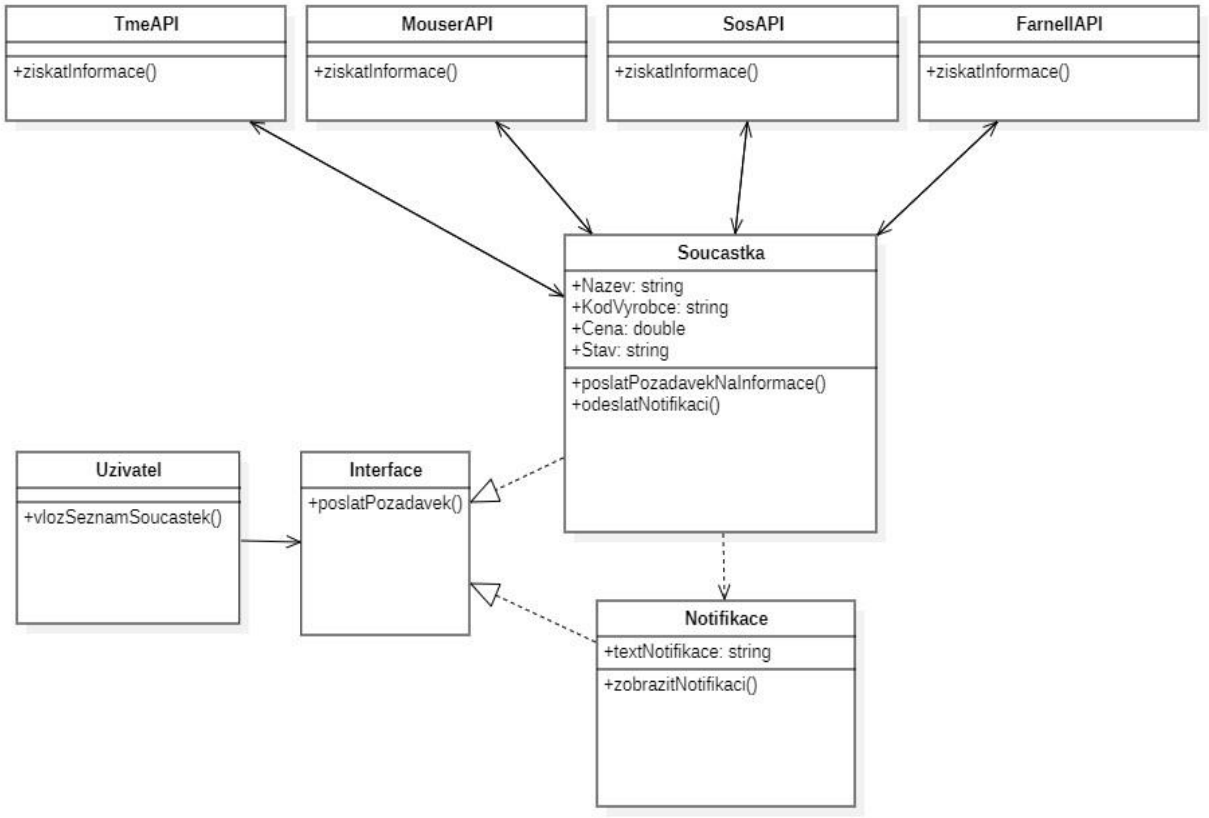
Pro každý z těchto e-shopů je klíčové získat a porozumět informacím uvedeným v jejich dokumentaci API. To zahrnuje autentizační postupy, povolené typy dotazů, dostupné filtry a formáty odpovědí. Pouze pečlivou analýzou těchto API lze následně navrhnout a implementovat efektivní program, který bude schopen získávat potřebné informace pro sledování dostupnosti a cen elektronických součástek.

2 Návrh řešení

V další části bude podrobně popsáno navrhované řešení pro vytvoření programu. Zahrnuje objektový návrh programu, diagramy běhu programu a chování uživatelů a následné informace a vybraných technologiích pro realizaci programu samotného.

2.1 Objektový návrh

Následná část obsahuje objektový návrh systému. Diagram identifikuje klíčové objekty v systému a znázorňuje jejich instance a vzájemné interakce prostřednictvím a relací. V objektovém návrhu jsou zahrnuty informace o objektech, jejich atributech a vlastnostech. Objektový diagram slouží k lepšímu pochopení programu a umožňuje vizuálně zobrazit jednotlivé objekty a jejich propojení.



Obrázek 1: Objektový diagram systému

Zdroj: vlastní práce

Z objektového návrhu je patrné, že uživatel, který s programem pracuje vloží přes Interface (uživatelské rozhraní) seznam součástek o kterých chce získat informace. Systém následně každou z těchto součástek zpracuje a pošle požadavek na API vybraných e-shopů, aby zjistil všechny potřebné informace o komponentě. API informace zjistí u prodejců a pošle zpět do systému. Informace jsou následně zpracovány a je odeslána notifikace pro uživatele do uživatelského rozhraní, ve kterém jsou pak aktuální informace zobrazeny.

2.2 Databázový model

Tato sekce popisuje databázový model. Databáze programu je jednoduchou strukturou o dvou tabulkách:

- **Components**
- **ComponentPrices**

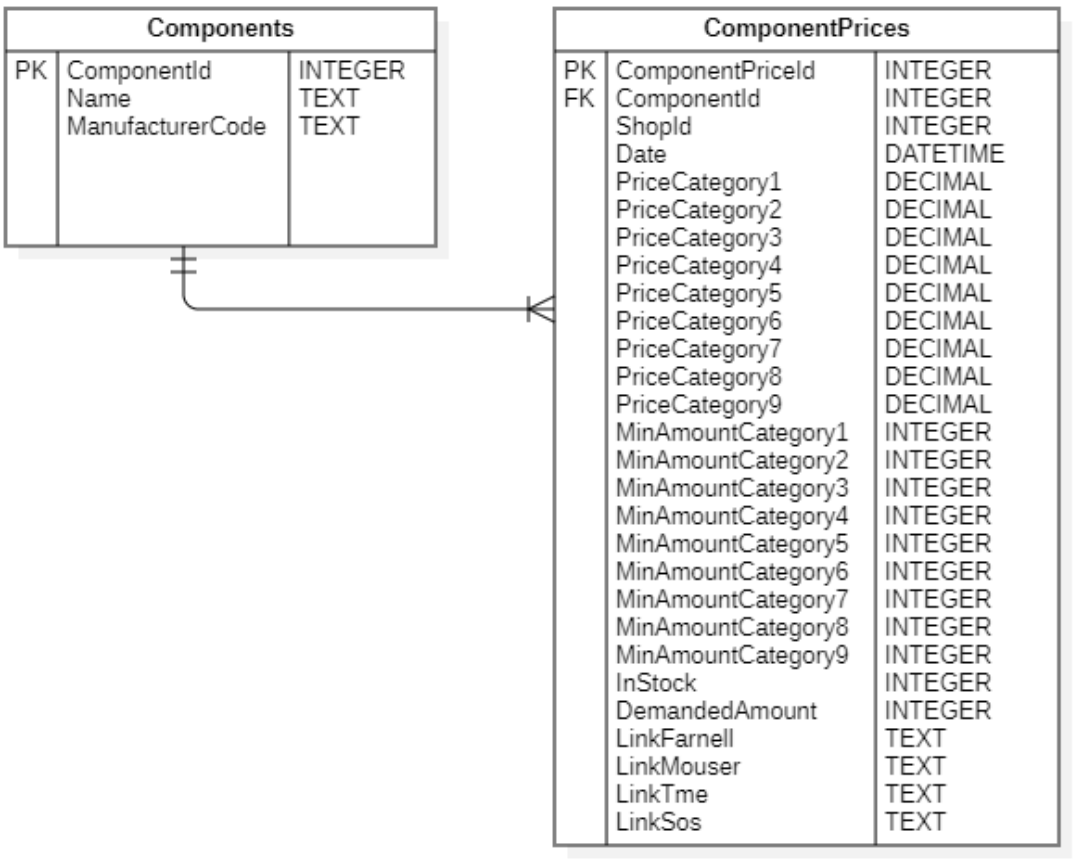
V tabulce Components jsou uloženy základní informace o zjištěných součástkách, uloženy v následujících sloupečcích.

- **ComponentId:** Primární klíč, interní identifikační číslo záznamu součástky.
- **Name:** Název součástky zjištěný při prvním zjišťování informací o součástce.
- **ManufacturerCode:** Kód výrobce součástky. Načítá se ze vstupního souboru uživatele.

V tabulce ComponentPrices jsou následně ukládány veškeré informace zjištěné při jednotlivých instancích dotazování z e-shopů.

- **ComponentPriceId:** Primární klíč, identifikační číslo záznamu.
- **ComponentId:** Cizí klíč, odkazuje na součástku: záznam z tabulky Components.
- **ShopId**: Identifikační číslo e-shopu. Jedná se o interní očíslování e-shopů pro následnou identifikaci. Očíslování je následné.
    - **1:** Farnell
    - **2**: Mouser
    - **3**: Tme
    - **4**: Sos
- **Date:** Datum a čas zjištění informací z e-shopů a vložení záznamu.
- **PriceCategory1:** Cena součástky, dle zjištěného nejnižšího možného koupitelného množství.
- **PriceCategory2**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **PriceCategory3**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **PriceCategory4**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **PriceCategory5**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **PriceCategory6**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **PriceCategory7**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.

- **PriceCategory8**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **PriceCategory9**: Cena součástky, dle další možné zjištěné množstevní a cenové kategorie. Hodnota může zůstat prázdná z důvodu nastavení prodejních kategorií v dotazovaném e-shopu.
- **MinAmountCategory1:** Minimální koupitelné množství součástek v konkrétní e-shopu, zjištěné při dotazování. Koresponduje s PriceCategory1.
- **MinAmountCategory2:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory2.
- **MinAmountCategory3:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory3.
- **MinAmountCategory4:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory4.
- **MinAmountCategory5:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory5.
- **MinAmountCategory6:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory6.
- **MinAmountCategory7:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory7.
- **MinAmountCategory8:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory8.
- **MinAmountCategory9:** Množství součástek dle další možné zjištěné množstevní a cenové kategorie v konkrétním e-shopu, zjištěné při dotazování. Hodnota může zůstat prázdná. Koresponduje s PriceCategory9.
- **InStock:** Dostupné množství součástek v e-shopu, v momentě dotazování.
- **DemandedAmount:** Požadované množství součástek k dotázání. Je zadáno vstupem od uživatele.
- **LinkFarnell:** Aktuální odkaz na součástku v dobře dotazování v e-shopu Farnell.
- **LinkMouser:** Aktuální odkaz na součástku v dobře dotazování v e-shopu Mouser.
- **LinkTme:** Aktuální odkaz na součástku v dobře dotazování v e-shopu Tme.
- **LinkSos:** Aktuální odkaz na součástku v dobře dotazování v e-shopu Sos.



Obrázek 2: Databázový ER diagram

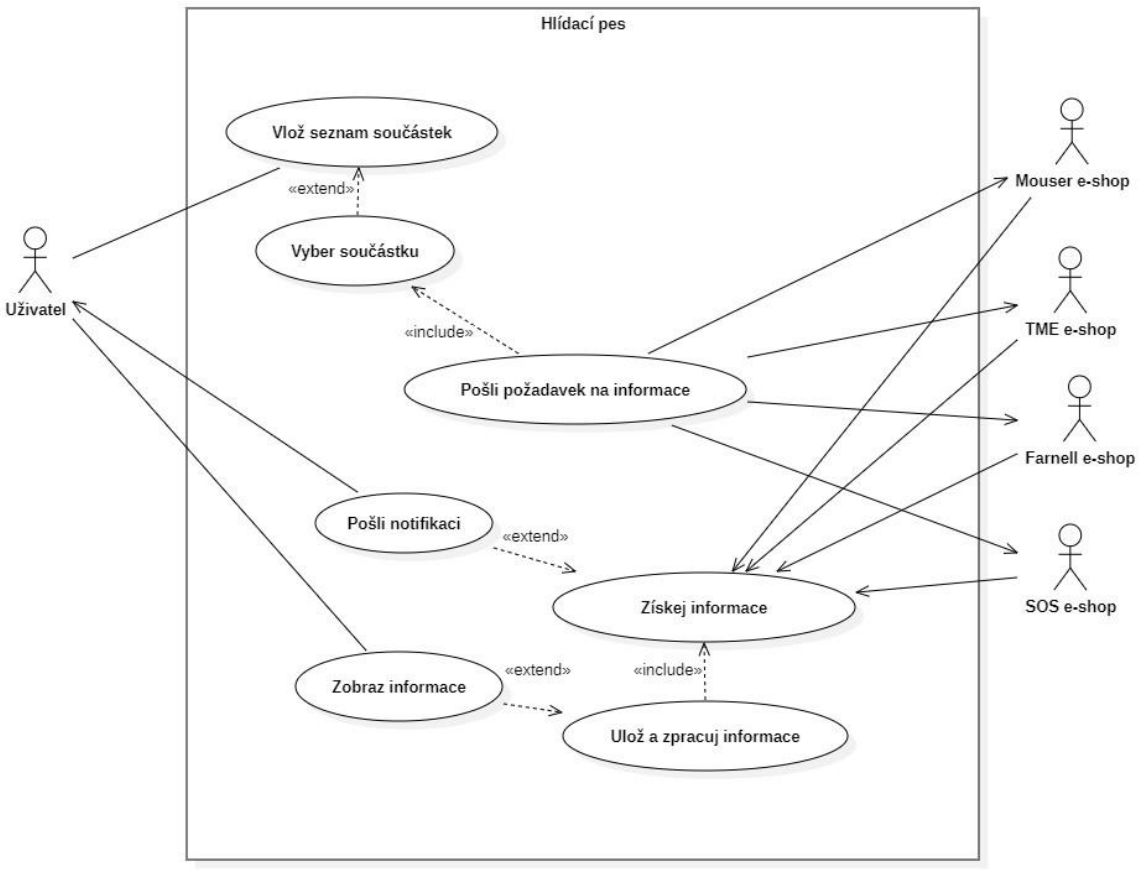
Zdroj: vlastní práce

2.3 Diagramy

V následující části jsou zobrazeny další diagramy popisující funkčnost systému z různých hledisek fungování.

2.3.1 Use case diagram

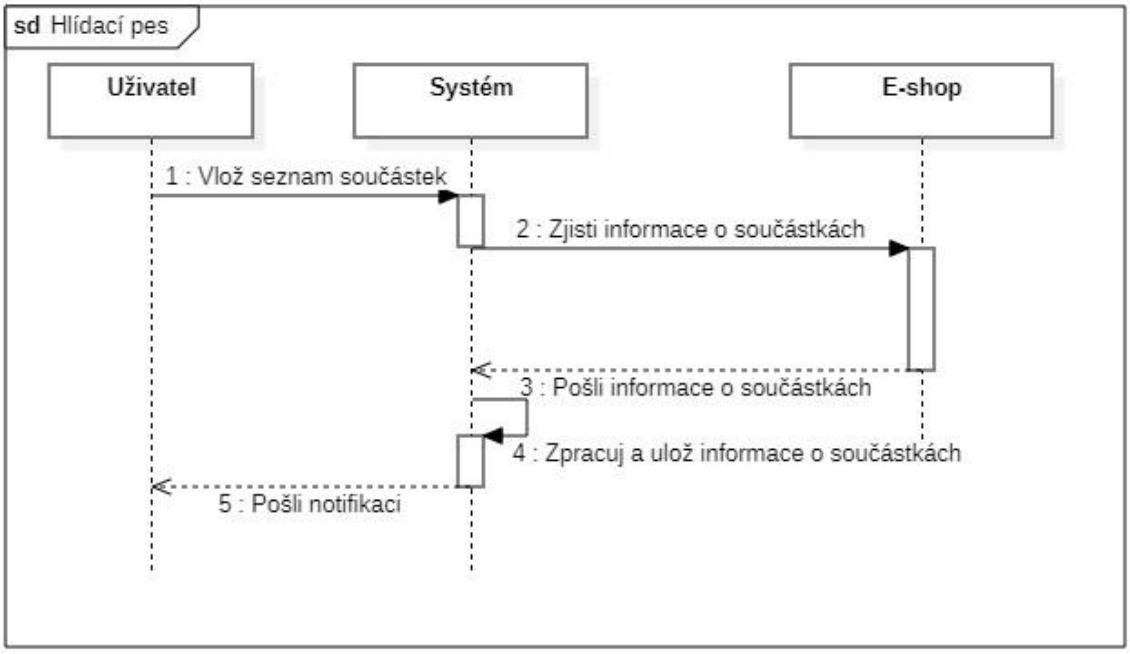
Use Case Diagram je grafický model v oblasti softwarového inženýrství, který ilustruje interakce mezi různými aktéry (uživateli nebo systémy) a systémem. Use case je scénář, který popisuje, jak aktér interaguje se systémem za určitých podmínek a zobrazuje vztahy mezi nimi *(IBM, 2021)*.



Obrázek 3: Use case diagram
Zdroj: vlastní práce

2.3.2 Sekvenční diagramy

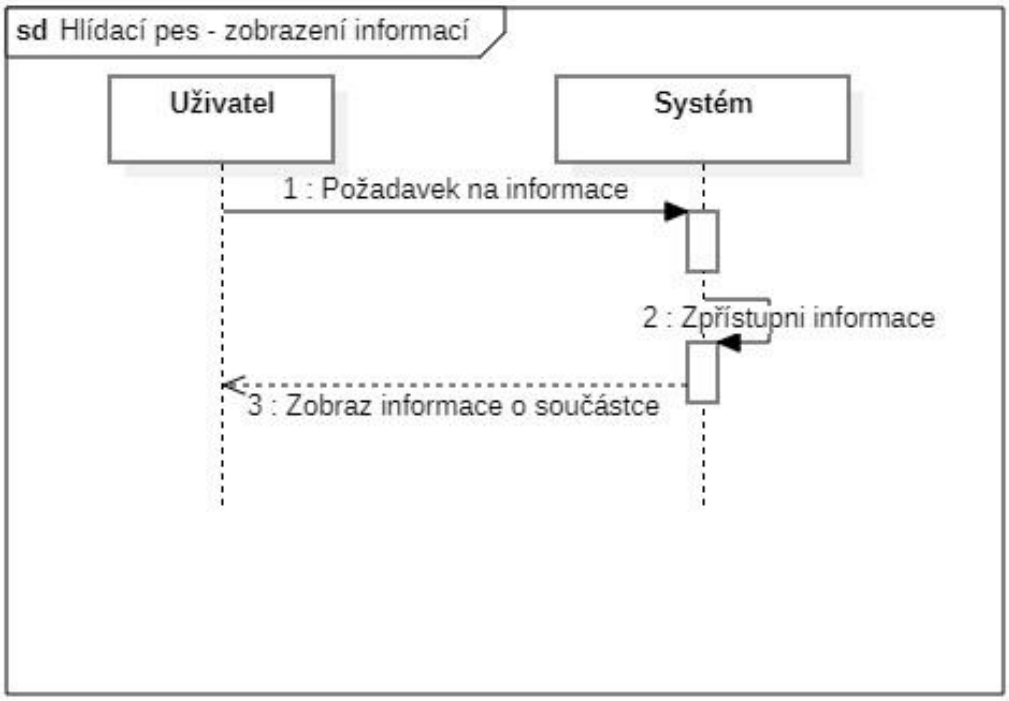
Sekvenční diagram zobrazující zjištění informací ilustruje interakce mezi různými objekty nebo entitami v systému v průběhu času *(IBM, 2023)*. V tomto scénáři je možné vidět chování systému při vložení seznamu součástek, kdy je po zadání vstupu seznam zpracován a následně poslán požadavek na informace o obsažených součástkách požadovanému e-shopu, který požadavek zpracuje a vrátí odpověď s informacemi. Po získání těchto informací je následně zobrazena notifikace uživateli s výstupem.



Obrázek 4: Sekvenční diagram - zjištění informací o součástkách

Zdroj: vlastní práce

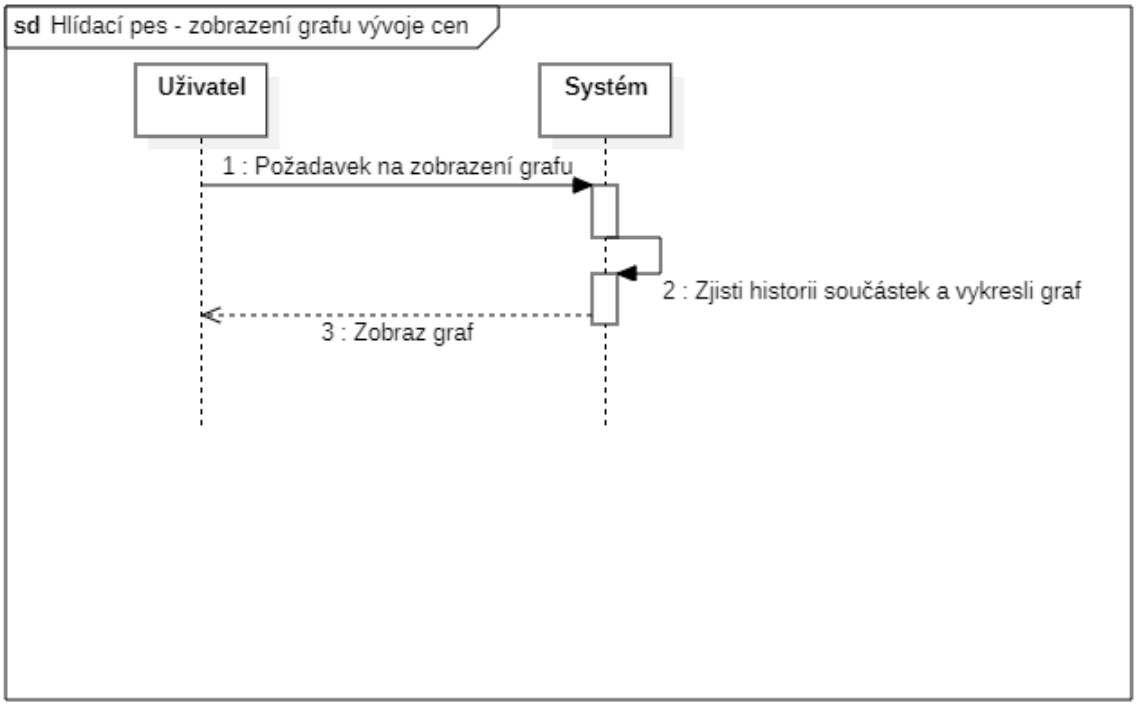
V následujícím diagramu je znázorněno zobrazení již zjištěných informací o součástkách, uživateli. Uživatel pošle požadavek systému, který zjistí a zpřístupní uložené informace o požadovaných součástkách a následně uživateli pošle zprávu s těmito daty.



Obrázek 5: Sekvenční diagram - zobrazení informací na vyžádání

Zdroj: vlastní práce

Další diagram ukazuje postup vykreslení grafu s informacemi o historii cen u požadované součástky.



Obrázek 6: Sekvenční diagram – zobrazení grafu vývoje cen

Zdroj: vlastní práce

2.4 Použité technologie

Po zhodnocení zadané problematiky a požadavků na běh programu, kdy je primární funkcionalita zjišťování aktuálních informací o součástkách u dodavatelů, dle vstupů uživatele a následné zobrazení zjištěných informací tomuto uživateli, bylo patrné, že program poběží na lokálním počítači operačním systémem Windows s připojením do internetu, a bude obsluhován právě jednou osobou, bez nutnosti sdílení zjištěných informací.

Pro implementaci programu tak byl vybrán programovací jazyk C# s využitím Frameworku .NET. Kdy program a hlavní uživatelské rozhraní bude vytvořeno pomocí formulářové aplikaci Windows Forms v programu Visual Studio 2022.

2.4.1 C#

C# je moderní, objektově orientovaný programovací jazyk, který je vhodný pro vývoj desktopových programů, obzvláště programů cílených na operační systémy Windows, jelikož byl vyvinut společností Microsoft. C# využívá pro tvorbu programů komponenty a poskytuje prostředí ve kterém je možná snadná organizace a údržbě kódu. Má čistou syntaxi a dobrou podporu pro objektově orientované programování. Dále klade velký důraz na bezpečnost a výkon, takže lze díky různým integrovaným funkčnostem vytvářet robustní a spolehlivé programy. *(Microsoft, 2023a)*.

2.4.2 .NET

.NET je vývojový framework postavený na programovacím jazyce C#. Poskytuje širokou škálu knihoven a nástrojů, což usnadňuje vývoj. Aplikace vyvinuté v .NET mohou dosahovat vysokého

výkonu. Také má integrované bezpečnostní funkce, které pomáhají chránit aplikace před různými hrozbami, a lze využít například identitu a přístupová práva. Just-in-time kompilace a optimalizace na úrovni platformy přispívají k efektivitě programu. .NET má velkou komunitu vývojářů a bohatou dokumentaci, což může být užitečné při řešení problémů a hledání optimálních postupů. *(Microsoft, 2023a)*.

2.4.3 Windows Forms

Windows Forms (WinForms) představuje technologii v rámci .NET Framework, zaměřenou na tvorbu uživatelsky přívětivých aplikací pro operační systém Windows. Pro program hlídacího psa na sledování cen a dostupnosti elektronických součástek je Windows Forms dobrou volbou díky svému přístupu orientovanému na design. Umožňuje rychlý vývoj uživatelského rozhraní pomocí rychlého přetažení ovládacích prvků, což je klíčové pro snadnou implementaci přehledného a intuitivního rozhraní pro uživatele.

Pomocí Windows Forms lze snadno propojit uživatelské rozhraní s back-endem aplikace, což umožňuje efektivní zpracování dat a rychlou odezvu na uživatelské interakce. Dále poskytuje strukturovaný přístup k vývoji, což usnadňuje udržovatelnost a škálovatelnost aplikace. To je důležité v případě, že projekt bude v budoucnu rozšiřován o další funkcionality. *(Microsoft, 2023b)*.

2.4.4 Visual Studio 2022

Visual Studio 2022 je integrované vývojové prostředí (IDE) od společnosti Microsoft, které nabízí vývojářům kompletní sadu nástrojů pro návrh, vývoj, ladění a nasazení programů. Visual Studio je optimalizováno pro vývoj v .NET Frameworku a nabízí vynikající podporu pro C#, což z něj dělá ideální nástroj pro tvorbu programů na platformě Windows.

Poskytuje hlubokou integraci s .NET technologiemi a knihovnami. Nabízí širokou škálu nástrojů pro ladění kódu, od sledování hodnot proměnných po analýzu výkonu a odhalování chyb. Což je výhodné pro využití různých funkcí .NET pro komunikaci s internetem, práci s daty apod.

Obsahuje nástroje pro správu kódu, včetně refaktorizace a automatické dokončování kódu, což zvyšuje produktivitu a efektivitu programování a přispívá k čistému a dobře udržovatelnému kódu. *(Microsoft, 2023c)*.

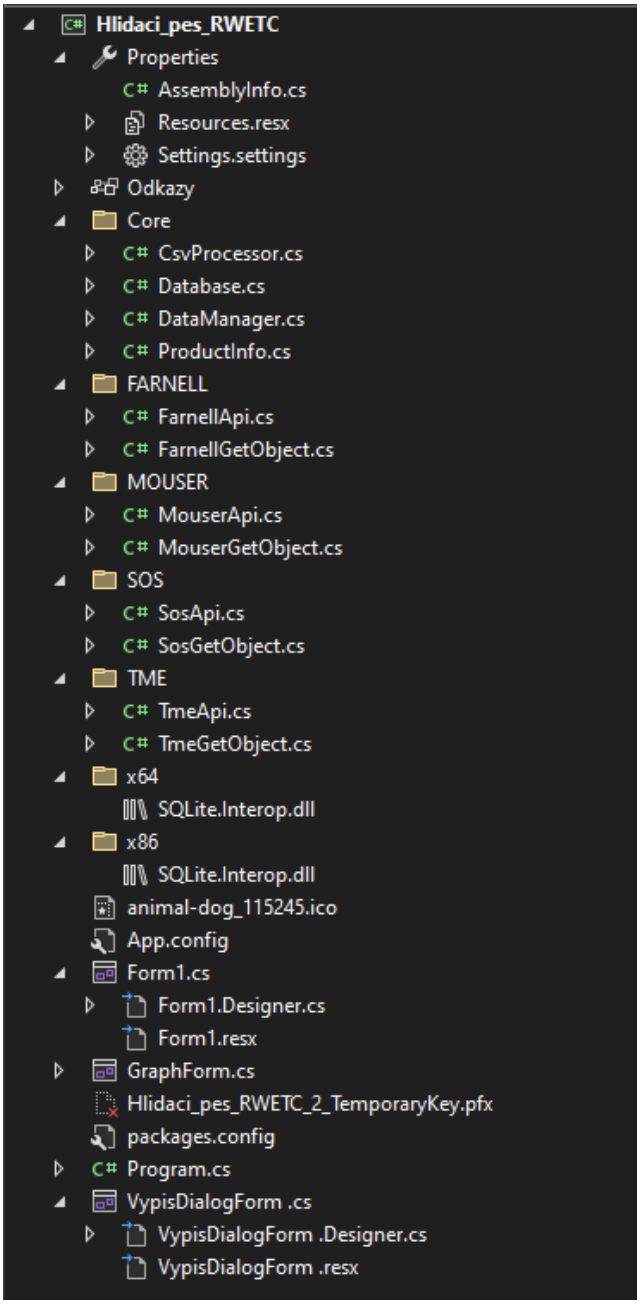
3 Realizace řešení

V této kapitole bude popsán již realizovaný program, jeho struktura, veškeré jeho části a funkce.

Program běží z větší části na hlavním formulářovém okně programu.

3.1 Struktura programu

Program je strukturován do několika základních složek, které obsahují specifické podřazené třídy a soubory.



Obrázek 7: Struktura programu

Zdroj: vlastní práce, Visual Studio 2022

Následuje popis složek a souborů.

3.1.1 Projektový název a základní složky

Projekt: Hlídací pes RWETC: Hlavní projekt obsahující všechny potřebné soubory a složky.

3.1.2 Složka Properties

AssemblyInfo.cs: Tento soubor obsahuje metadata o sestavení, jako je název, popis, verze atd.

Resources.resx: Zde jsou uložené zdroje, které mohou být používány napříč celým projektem, jako jsou například řetězce nebo obrázky.

Settings.settings: Tento soubor obsahuje uživatelská nastavení a konfigurace.

3.1.3 Složka Odkazy (References)

Obsahuje odkazy na externí knihovny a závislosti, které projekt potřebuje.

3.1.4 Složka Core

CsvProcessor.cs: Třída zodpovědná za zpracování CSV souborů, extrakci dat a jejich předání dalším částem aplikace.

Database.cs: Třída pro práci s SQLite databází, včetně připojení, dotazů a tříd podřízených objektů.

DataManager.cs: Třída správce dat, která koordinuje zpracování informací z hlavního programu a ukládáním dat do databáze.

ProductInfo.cs: Třída obsahující jednotnou strukturu pro práci s daty o zjištěných produktech ze všech e-shopů.

3.1.5 Složka FARNELL

FarnellApi.cs: Implementace třídy pro připojení a pro komunikaci s API společnosti Farnell.

FarnellGetObject.cs: Třída pro zpracování získaných dat z API zjištěná pomocí třídy FarnellApi.

3.1.6 Složka MOUSER

MouserApi.cs: Implementace třídy pro připojení a pro komunikaci s API společnosti Mouser.

MouserGetObject.cs: Třída pro zpracování získaných dat z API zjištěná pomocí třídy MouserApi.

3.1.7 Složka SOS

SosApi.cs: Implementace třídy pro připojení a pro komunikaci s API společnosti SOS.

SosGetObject.cs: Třída pro zpracování získaných dat z API zjištěná pomocí třídy SosApi.

3.1.8 Složka TME

TmeApi.cs: Implementace třídy pro připojení a pro komunikaci s API společnosti TME.

TmeGetObject.cs: Třída pro zpracování získaných dat z API zjištěná pomocí třídy MouserApi.

3.1.9 Složky x64 a x86

SQLite.Interop.dll: Knihovna pro komunikaci s SQLite databází pro 64bitové a 32bitové systémy.

3.1.10 Ostatní soubory

animal-dog_115245.ico: Ikona používaná v aplikaci.

App.config: Konfigurační soubor aplikace obsahující nastavení jako jsou řetězce připojení a další konfigurační parametry.

3.1.11 Formuláře

Form1.cs: Hlavní formulář aplikace.

Form1.Designer.cs: Automaticky generovaný kód pro návrh formuláře.

Form1.resx: Zdroje použité ve formuláři, jako jsou například lokalizované řetězce.

GraphForm.cs: Formulář pro zobrazení grafů.

VypisDialogForm.cs: Formulář pro výpis upozornění uživateli.

VypisDialogForm.Designer.cs: Automaticky generovaný kód pro návrh formuláře.

VypisDialogForm.resx: Zdroje použité ve formuláři.

3.1.12 Certifikát

Hlídací_pes_RWETC_2_TemporaryKey.pfx: Klíč použitý pro podepisování aplikace.

3.1.13 Konfigurační soubor

packages.config: Seznam závislostí a knihoven, které projekt používá.

3.1.14 Hlavní vstupní bod

Program.cs: Hlavní vstupní bod aplikace, obsahující metodu Main, která spouští aplikaci.

3.2 Třídy programu

V této kapitole se zaměříme na klíčové třídy, které tvoří základ aplikace pro sledování a porovnávání cen elektronických součástek. Pro lepší porozumění funkcionality aplikace bude tato kapitola detailně popisovat jednotlivé třídy, jejich konstrukci, metody a interakce. Analyzujeme, jak tyto třídy spolupracují na dosažení celkového cíle aplikace, a jak jejich implementace přispívá k uživatelsky přívětivému a efektivnímu nástroji pro sledování cen.

3.2.1 Popis třídy CsvManager

Třída CsvManager je navržena pro správu souborů CSV v rámci projektu Hlídací pes RWETC. Tato třída poskytuje metody pro operace jako načítání, přidávání, čtení obsahu, přejmenovávání a mazání CSV souborů. Níže je detailní popis jednotlivých metod a funkcí této třídy.

Konstruktor

- **CsvManager(string directoryPath)**: Konstruktor třídy, který přijímá cestu k adresáři, kde budou CSV soubory uloženy. Pokud adresář neexistuje, vytvoří ho.

Metody

- **GetCsvFiles():** Tato metoda načítá všechny CSV soubory z určeného adresáře a vrací seznam jejich názvů bez přípony .csv.
- **AddCsvFile(string fileName, string content):** Metoda přidává nový CSV soubor do adresáře. Nejprve kontroluje, zda soubor s daným názvem již neexistuje. Pokud existuje, vyvolá výjimku. Pokud ne, zapíše obsah do nového souboru.
- **GetCsvContent(string fileName):** Tato metoda čte a vrací obsah vybraného CSV souboru. Pokud soubor neexistuje, vrací null. Při čtení používá FileStream a StreamReader pro zajištění bezpečného přístupu k souboru.
- **RenameCsvFile(string oldFileName, string newFileName):** Metoda přejmenovává existující CSV soubor. Nejprve kontroluje, zda soubor s novým názvem již neexistuje. Pokud existuje, vyvolá výjimku. Pokud ne, přejmenuje soubor.
- **DeleteCsvFile(string fileName):** Tato metoda smaže specifikovaný CSV soubor. Pokud soubor existuje, pokusí se ho smazat. Při selhání vyvolá výjimku.

Shrnutí

Třída CsvManager poskytuje kompletní sadu funkcí pro správu CSV souborů v rámci zadaného adresáře. Díky dobře strukturovaným metodám umožňuje bezpečné a efektivní přidávání, čtení, přejmenovávání a mazání CSV souborů, což zajišťuje flexibilní správu dat v aplikaci Hlídací pes RWETC. Tato třída je klíčová pro manipulaci se soubory CSV a integraci dat z těchto souborů do dalších částí aplikace.

3.2.2 Popis třídy Database

Třída Database je navržena na vytvoření a správu SQLite databáze pro systém hlídacího psa.

Konstruktor

- **ComponentDatabase(string databaseName):** Konstruktor třídy, který přijímá název databáze. Nastaví název a cestu k databázi.

Metody

- **ComponentDatabase GetInstance():** Vrací instanci třídy ComponentDatabase. Pokud instance ještě neexistuje, vytvoří ji a inicializuje databázi.
- **InitializeDatabase():** Vytváří tabulky Components a ComponentPrices v databázi, pokud ještě neexistují.
- **AddComponent(string componentName, string manufacturerCode):** Přidává nový záznam do tabulky Components, pokud komponenta se stejným ManufacturerCode ještě neexistuje. Vrací id nově přidané komponenty.
- **AddComponentPrice(int componentId, string store, decimal price1000Qty, decimal price400Qty, decimal price100Qty, decimal price10Qty, string storeLink1000Qty, string storeLink400Qty, string storeLink100Qty, string storeLink10Qty, string stockStatus, int minimumOrderQuantity, int requiredQuantity):** Přidává nový záznam do tabulky ComponentPrices pro danou komponentu a obchod.
- **ComponentExists(string manufacturerCode):** Kontroluje, zda v databázi existuje komponenta se stejným ManufacturerCode. Vrací true, pokud existuje, jinak vrací false.
- **GetLatestComponentPrices():** Získává nejnovější záznamy o cenách komponent z databáze.
- **GetSecondLatestComponentPriceByComponentId(int componentId, string store):** Získává druhý nejnovější záznam o ceně komponenty pro daný obchod.
- **GetComponents():** Získává všechny komponenty z databáze.
- **GetComponentsSearch(string searchString):** Vyhledává komponenty podle zadaného řetězce.
- **GetPriceData(int componentId, string store, string quantityType):** Získává data o cenách komponent pro konkrétní obchod a komponentu pro vykreslení grafů.

Pomocné metody

- **ExecuteNonQuery(string query, Dictionary<string, object> parameters):** Vykonává SQL dotazy, které nevracejí data (např. INSERT, UPDATE, DELETE).
- **TryParseDecimal(SQLiteDataReader reader, string columnName):** Bezpečně získává decimal hodnotu z SQLite data readeru.
- **TryParseInt(SQLiteDataReader reader, string columnName):** Bezpečně získává integerovou hodnotu z SQLite data readeru.
- **TryParseString(SQLiteDataReader reader, string columnName):** Bezpečně získává string hodnotu z SQLite data readeru.

Shrnutí

Třída ComponentDatabase je navržena pro správu databáze komponent a jejich cen pomocí SQLite databázového systému. Používá pattern Singleton, aby zajistila, že v aplikaci existuje pouze jedna instance této třídy. Metody této třídy umožňují inicializaci databáze, přidávání nových komponent a jejich cen, kontrolu existence komponent, a získávání dat pro analytické účely. Pomocné metody zajišťují bezpečnou manipulaci s daty a vykonávání SQL dotazů. Třída je klíčová pro efektivní správu dat o komponentech a jejich cenách v aplikaci.

3.2.3 Popis třídy DataManager

Třída DataManager slouží jako centrální bod pro zpracování a ukládání dat o komponentách získaných z různých API.

Konstruktor

- **DataManager():** Inicializuje novou instanci třídy DataManager. Vytváří instance API klientů pro různé obchody (Farnell, Mouser, TME, SOS) a připojuje se k databázi komponent.

Metody

- **ProcessFarnellApiResponse(string term, string formattedDateTime, int demandedAmount):** Získává a zpracovává odpověď z API Farnell pro zadaný termín. Přidává data o produktu do databáze.
- **ProcessMouserApiResponse(string term, string formattedDateTime, int demandedAmount):** Získává a zpracovává odpověď z API Mouser pro zadaný termín. Přidává data o produktu do databáze.
- **ProcessTmeApiResponse(string[] terms, string formattedDateTime, List<int> demandedAmounts):** Získává a zpracovává odpověď z API TME pro zadané termíny. Přidává data o produktech do databáze.
- **ProcessSOSApiResponse(string term, string formattedDateTime, int demandedAmount):** Získává a zpracovává odpověď z API SOS pro zadaný termín. Přidává data o produktu do databáze.
- **AddDataToDatabase(ProductInfo productInfo, string date, int demandedAmount, int shopId):** Přidává data o komponentě a její ceně do databáze. Pokud komponenta neexistuje, přidá ji nejdříve do tabulky Components a poté přidá její cenu do tabulky ComponentPrices. Metoda se používá pro všechny e-shopy kromě Tme.
- **AddDataToDatabaseTme(List<ProductInfo> productInfos, string date, List<int> demandedAmounts, string url):** Přidává data o více komponentech a jejich cenách do databáze. Používá se specificky pro API TME. Kontroluje, zda počet požadovaných množství odpovídá počtu produktů. Metoda je vytvořena speciálně pro e-shop Tme, z důvodu rozdílného typu odpovědi z API než u ostatních e-shopů.

Shrnutí

Třída obsahuje metody pro zpracování odpovědí z API obchodů Farnell, Mouser, TME a SOS, a pro přidávání získaných dat do databáze. Konstruktor třídy inicializuje API klienty a připojuje se k databázi. Třída poskytuje robustní mechanizmus pro správu a ukládání informací o komponentách a jejich cenách, což usnadňuje sledování a analýzu cenových změn v různých obchodech.

3.2.4 Popis třídy ProductInfo

ProductInfo je pomocná třída se strukturou objektu produktu, do kterého se ukládají data při zjištění z e-shopů.

3.2.5 Popis třídy FarnellApi

Jedná se o hlavní třídu pro komunikaci s API e-shopu Farnell.

Metody

- **GetFarnellApiResponse(string itemCode):** Tato metoda posílá asynchronní HTTP GET požadavek na API Farnell, aby získala informace o produktu na základě zadaného kódu produktu (itemCode).

Shrnutí

Třída FarnellApi slouží k získání dat o produktech z API e-shopu Farnell. Obsahuje metodu GetFarnellApiResponse, která konstruuje URL požadavek na základě vstupních parametrů a posílá ho pomocí HttpClient. Odpověď z API je zpracována a vrácena jako JSON řetězec. Pokud dojde k chybě během požadavku, metoda vrací zprávu s popisem chyby. Třída je navržena tak, aby byla robustní a zvládla různé typy chyb, které mohou nastat při komunikaci se serverem.

3.2.6 Popis třídy FarnellGetObject

Třída pro zpracování získaných informací z e-shopu Farnell.

Metody

- **GetProductInfo(string jsonInput, string itemCode):** Tato metoda zpracovává JSON vstup z API odpovědi a extrahuje informace o produktu na základě zadaného kódu produktu (itemCode). Pokud produkt není nalezen nebo je formát dat nesprávný, vyvolá výjimku.
- **GetPriceGroup(Price[] prices, int group):** Tato metoda získává cenu z určité cenové skupiny (group) z pole cen (prices). Jedná se o pomocnou metodu z důvodu dynamických změn v množství a cenách u produktů.
- **GetAmountGroup(Price[] prices, int group):** Tato metoda získává minimální množství z určité cenové skupiny (group) z pole cen (prices). Jedná se o pomocnou metodu z důvodu dynamických změn v množství a cenách u produktů.
- Dále třída obsahuje pomocné třídy se strukturou objektu produktu, získaného jako odpoveď z API e-shopu.

Shrnutí

Třída FarnellGetObject slouží k extrahování informací o produktu z JSON odpovědi získané z API Farnell. Hlavní metodou je GetProductInfo, která deserializuje JSON řetězec do objektu a extrahuje potřebné informace o produktu. Třída obsahuje pomocné metody GetPriceGroup a GetAmountGroup pro získání specifických cenových a množstevních informací z pole cen. V případě chyb při parsování nebo neplatných dat vyvolá výjimky, které pomáhají identifikovat a řešit problémy.

3.2.7 Popis třídy MouserApi

Jedná se o hlavní třídu pro komunikaci s API e-shopu Mouser.

Metody

- **GetMouserApiResponse(string itemCode):** Tato metoda asynchronně odesílá HTTP POST požadavek na API Mouser se zadaným kódem položky (itemCode). Po úspěšném volání API vrací JSON odpověď jako řetězec. Pokud dojde k chybě, vrací zprávu o chybě.
- Dále obsahuje pomocné třídy reprezentující strukturu vráceného JSON objektu z API e-shopu Mouser.

Shrnutí

Třída MouserApi je navržena k interakci s API Mouser a získávání informací o produktech na základě zadaného kódu položky. Klíčovou metodou této třídy je GetMouserApiResponse, která odesílá HTTP POST požadavek obsahující kód položky ve formátu JSON a vrací JSON odpověď od API.

3.2.8 Popis třídy MouserGetObject

Třída pro zpracování získaných informací z e-shopu Mouser.

Metody

- **GetProductInfo(string jsonInput, string itemCode):** Tato metoda deserializuje JSON vstup do objektu MouserRootObject, ověří, zda byly nalezeny výsledky, a pokud ano, vrátí objekt ProductInfo obsahující informace o produktu. V případě chyby nebo nenalezení produktu vyhodí výjimku.
- **GetPriceGroup(PriceBreak[] priceBreaks, int group):** Tato metoda vrací cenu pro specifikovanou cenovou skupinu z pole PriceBreak[]. Pokud daná skupina existuje, vrátí cenu, jinak vrátí nulu.
- **GetAmountGroup(PriceBreak[] priceBreaks, int group):** Tato metoda vrací minimální množství pro specifikovanou množstevní skupinu z pole PriceBreak[]. Pokud daná skupina existuje, vrátí množství, jinak vrátí nulu.
- **ParseStockLevel(string availability):** Tato metoda analyzuje řetězec dostupnosti a vrátí úroveň zásob jako celé číslo. Pokud se nepodaří převést na celé číslo, vrátí nulu.

Shrnutí

Třída MouserGetObject je navržena k získávání a parsování informací o produktech z JSON odpovědi API Mouser. Klíčovou metodou je GetProductInfo, která zpracovává JSON vstup a vrací objekt ProductInfo obsahující detailní informace o produktu.

3.2.9 Popis třídy SosApi

Jedná se o hlavní třídu pro komunikaci s API e-shopu Sos.

Konstruktor

- Konstruktor SosApiClient inicializuje novou instanci třídy HttpClient a nastavuje základní URL pro API a přihlašovací údaje pro autentizaci.

Metody

- **GetAccessTokenAsync():** Metoda GetAccessTokenAsync je asynchronní a jejím účelem je získání přístupového tokenu. Provádí to pomocí základní autentizace, kdy je hlavička Authorization nastavena na hodnotu Basic a zašifrovaný řetězec přihlašovacích údajů. Tato metoda odesílá POST požadavek na URL https://api-customer.sos.sk/auth/token/password s tělem požadavku obsahujícím uživatelské jméno a heslo. Pokud je odpověď úspěšná, metoda uloží získaný token do proměnné _accessToken a vrací jej. V případě neúspěšné odpovědi metoda vyhazuje výjimku s chybovým kódem.
- **GetProductsAsync(string accessToken, string query = null, string brand = null, string sorter = null, int limit = 10, int offset = 0):** Metoda GetProductsAsync je také asynchronní a její úkolem je získání seznamu produktů z API. Nejprve deserializuje předaný přístupový token a získá skutečný token, který pak použije k nastavení hlavičky Authorization s hodnotou Bearer. Metoda připraví query string s parametry pro dotaz, značku, seřazení, limit a offset a odesílá GET požadavek na URL https://api-customer.sos.sk/products s tímto query stringem. Pokud je odpověď úspěšná, metoda vrací obsah odpovědi jako řetězec. V případě neúspěšné odpovědi metoda vyhazuje výjimku s chybovým kódem.
- **GetDocumentationAsync():** Metoda GetDocumentationAsync je určena pro získání dokumentace API. Nastaví hlavičku a odesílá GET požadavek na URL https://api-customer.sos.sk/docs. Pokud je odpověď úspěšná, metoda vrací obsah odpovědi jako řetězec. V případě neúspěšné odpovědi metoda vyhazuje výjimku s chybovým kódem. Metoda byla vytvořena jako prvotní testovací metoda připojení k API z důvodů jednoduchosti zaslaného požadavku. V samotném běhu programu se nepoužívá.
- Dále obsahuje pomocné třídy reprezentující strukturu vráceného JSON objektu z API e-shopu Sos.

Shrnutí

Třída SosApiClient je navržena pro komunikaci s API SOS pomocí základní autentizace a přístupových tokenů. Obsahuje dvě hlavní metody: GetAccessTokenAsync, která získává přístupový token pro autentizaci; GetProductsAsync, která získává seznam produktů podle různých parametry. Všechny metody používají instanci HttpClient pro odesílání HTTP požadavků a vrací obsah odpovědi jako řetězec v případě úspěchu, nebo vyhazují výjimku v případě chyby.

3.2.10 Popis třídy SosGetObject

Třída pro zpracování získaných informací z e-shopu Sos.

Metody

- **GetProductInfo(string jsonInput):** Tato metoda je zodpovědná za extrakci informací o produktu z JSON vstupu předaného z třídy SosApi. Nejprve deserializuje JSON řetězec do instance třídy RootObject. Poté zkontroluje, zda deserializace byla úspěšná a zda objekt obsahuje produkty. Pokud ne, vyhodí výjimku. Pokud je první produkt nalezen, metoda vrací novou instanci třídy ProductInfo naplněnou informacemi o ceně, minimálním množství, skladové dostupnosti, názvu, kódu výrobce a odkazu na produkt.

V případě chyby při parsování JSON vstupu metoda vyhazuje výjimku s chybovou zprávou.

- **GetPriceGroup(Pricelist[] prices, int group):** Tato pomocná metoda získává cenu pro danou skupinu množství. Prochází pole prices a podle zadané skupiny vrací odpovídající cenu jako datový typ decimal. Pokud pole obsahuje cenu pro zadanou skupinu, metoda ji vrátí, jinak vrátí nulu.
- **GetAmountGroup(Pricelist[] prices, int group):** Tato pomocná metoda získává minimální množství pro danou skupinu. Prochází pole prices a podle zadané skupiny vrací odpovídající množství jako int. Pokud pole obsahuje množství pro zadanou skupinu, metoda jej vrátí, jinak vrátí nulu.
- Třída SosGetObject používá několik dalších tříd k reprezentaci dat, která jsou deserializována z JSON vstupu. Tyto třídy jsou RootObject, Product, Pricelist, UnitPrice a Availability.

Shrnutí

Třída SosGetObject je navržena k extrakci a zpracování informací o produktech z JSON dat. Hlavní metodou je GetProductInfo, která deserializuje JSON vstup, extrahuje informace o produktu a vrací je ve formě instance třídy ProductInfo. Pomocné metody GetPriceGroup a GetAmountGroup jsou používány k získání ceny a minimálního množství pro různé skupiny množství. Třída využívá několik dalších tříd k reprezentaci struktury JSON dat, což usnadňuje deserializaci a práci s daty.

3.2.11 Popis třídy TmeApi

Jedná se o hlavní třídu pro komunikaci s API e-shopu Tme.

Metody

- **GetTmeApiResponseURL(string[] symbols):** Tato metoda asynchronně získává URL součástek z TME API. Připraví potřebné parametry, včetně země, jazyka a seznamu symbolů, které jsou následně zakódovány a podepsány. Poté provede POST požadavek na API a vrací odpověď jako řetězec.
- **EncodeParams(Dictionary<string, string> apiParams):** Tato metoda zakóduje parametry API do URL kódovaného formátu. Přijímá slovník parametrů a vrací zakódovaný řetězec.
- **GenerateApiSignature(string input, string key):** Tato metoda generuje podpis API pomocí algoritmu HMAC-SHA1. Přijímá vstupní řetězec a klíč a vrací Base64 zakódovaný podpis.
- **GenerateApiSignature2(string method, string uri, Dictionary<string, string> parameters, string key):** Tato metoda generuje API podpis pomocí metody HMAC-SHA1, podobně jako GenerateApiSignature, ale s dalším krokem řazení parametrů. Přijímá HTTP metodu, URI, slovník parametrů a klíč a vrací Base64 zakódovaný podpis. Metoda byla vytvořena z úprav API na straně e-shopu.
- **UrlEncode(string s):** Tato metoda kóduje zadaný řetězec do URL kódovaného formátu. Vrací zakódovaný řetězec s velkými písmeny v hexadecimálním vyjádření.

- **PingTmeApi():** Tato metoda asynchronně pinguje TME API za účelem ověření dostupnosti. Připraví parametry a provede POST požadavek na API, vrací odpověď jako řetězec. Metoda byla vytvořena jako způsob testování připojení a získávání odpovědi z API. V programu se reálně nepoužívá.
- **GetTmeApiResponse2(string[] symbols):** Tato metoda je novější verzí GetTmeApiResponseURL, která byla v programu zakomentována. Nová verze byla vytvořena z důvodů změn v API ze strany e-shopu. Metoda asynchronně získává informace o produktech z TME API. Připraví potřebné parametry, zakóduje je a podepíše. Poté provede POST požadavek na API a vrací odpověď jako řetězec.
- Dále obsahuje pomocné třídy reprezentující strukturu vráceného JSON objektu z API e-shopu Tme.

Shrnutí

Třída TmeApi je navržena pro interakci s TME API za účelem získání informací o produktech, jejich cenách a dostupnosti. Obsahuje metody pro vytváření a odesílání požadavků na API, kódování parametrů a generování podpisů. Metody GetTmeApiResponseURL a GetTmeApiResponse2 jsou klíčové pro získání dat z API, zatímco pomocné metody jako EncodeParams a GenerateApiSignature2 jsou používány pro přípravu požadavků.

3.2.12 Popis třídy TmeGetObject

Třída pro zpracování získaných informací z e-shopu Tme.

Metody

- **GetProductInfo(string jsonInput):** Tato metoda přijímá JSON řetězec vrácnený z třídy TmeApi jako vstup a vrací seznam objektů ProductInfo, které obsahují informace o cenách, dostupnosti a kódu produktu. Nejprve se JSON řetězec deserializuje do objektu typu TmeRootObject, kontroluje se, zda jsou data platná, a následně se iteruje přes každý produkt v seznamu a získávají se informace o cenách a dostupnosti pomocí pomocných metod GetPriceGroup a GetAmountGroup. Vytvořené ProductInfo objekty se přidávají do seznamu, který je nakonec vrácen.
- **GetPriceGroup(Price[] prices, int group):** Tato pomocná metoda přijímá pole cen a číslo skupiny ceny a vrátí odpovídající cenu pro danou skupinu.
- **GetAmountGroup(Price[] prices, int group):** Tato pomocná metoda přijímá pole cen a číslo skupiny a vrátí odpovídající množství produktu pro danou skupinu.
- **GetProductUrl(string json):** Tato metoda přijímá JSON řetězec jako vstup a vrací URL adresu produktu. Nejprve se řetězec deserializuje do objektu JObject, pak se získává hodnota "ProductInformationPage" a doplňuje se protokol.

Shrnutí

Třída TmeGetObject slouží k získávání informací o produktech ze vstupního JSON formátu získaného z TME API. Metoda GetProductInfo deserializuje vstupní JSON do objektu a extrahuje informace o cenách, dostupnosti a kódech produktů. Pomocné metody GetPriceGroup a GetAmountGroup slouží k získání konkrétních cenových a množstevních údajů. Metoda

GetProductUrl získává URL adresu produktu z JSON. Třída poskytuje přehledný způsob manipulace s daty produktů z TME.

3.2.13 Popis třídy Form1

Třída Form1 je hlavní třídou běhu programu, ve které se nachází formulář, kde probíhá většina práce a ovládání programu.

Konstruktor

Konstruktor třídy Form1 inicializuje instanci třídy. Při vytváření instance třídy Form1 jsou prováděny následující kroky:

- **Inicializace komponent**: Volání metody InitializeComponent(), která inicializuje komponenty formuláře, jako je například nastavení vlastností prvků formuláře a jejich umístění.
- **Nastavení cest**: Inicializuje proměnnou tableFolderPath, což je cesta k adresáři pro tabulky, které jsou umístěné ve složce s programem.
- **Inicializace objektů**: Inicializace objektů csvManager a componentDb. Objekt csvManager je inicializován s cestou k adresáři pro tabulky vstupů. Objekt componentDb je inicializován voláním metody GetInstance třídy ComponentDatabase s názvem souboru databáze.
- **Nastavení událostí**: Nastavení událostí pro prvky formuláře, jako je například stisknutí tlačítka nebo dvoj-klik na buňku tabulky.
- **Nastavení výchozích hodnot**: Nastavení výchozích hodnot pro některé prvky formuláře, jako je například výběr výchozí položky v comboBoxTime nebo zjištění zaškrtnutí notifikací v checkBoxNotifikace.

Metody

- **buttonUploadTable_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonUploadTable". Zobrazuje dialogové okno pro výběr CSV souboru. Po výběru souboru se jeho obsah načte a přidá do instance csvManager, což je objekt třídy CsvManager. Poté se aktualizuje obsah listBoxTables, který zobrazuje seznam načtených souborů.
- **RefreshListBox():** Tato metoda aktualizuje obsah listBoxTables. Nejprve vyčistí jeho obsah a poté načte seznam souborů pomocí metody GetCsvFiles() z instance csvManager a přidá je do listBoxTables.
- **buttonRename_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonRename". Získává vybraný soubor v listBoxTables, zobrazuje dialogové okno pro zadání nového názvu a pokud je nový název zadaný a liší se od původního, provádí přejmenování souboru tabulky v csvManager a aktualizuje obsah listBoxTables.
- **ShowRenameDialog(string prompt, string title, string defaultText):** Tato metoda zobrazuje dialogové okno pro zadání nového názvu souboru. Uživatel může zadat nový název a potvrdit jej. Metoda vrátí nový název nebo null, pokud uživatel zruší dialogové okno.

- **buttonDelete_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonDelete". Získává vybraný soubor v listBoxTables a po potvrzení odstraní soubor z csvManager a aktualizuje obsah listBoxTables.
- **buttonOverit_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonOverit". Spouští asynchronní metodu Zjistit_async(), která provádí zjištění informací o součástkách. Po dokončení nebo přerušení operace obnoví stav formuláře.
- **Timer_Tick(object sender, EventArgs e):** Tato metoda je volána při vypršení intervalu časovače. Vyčistí tabulku dataGridViewResultApi a znovu provede zjištění informací o součástkách pomocí metody Zjistit_async().
- **Progress_Tick(object sender, EventArgs e):** Tato metoda je volána při vypršení intervalu časovače pro zobrazení průběhu. Aktualizuje hodnotu průběžného pokroku v progressBarTimer.
- **reset_form():** Tato metoda slouží k resetování formuláře na výchozí stav. Zastavuje běh časovače a progress baru, nastavuje hodnotu a viditelnost progress baru, a vypíná tlačítko pro zastavení periodické aktualizace. Také nastavuje přístupnost různých ovládacích prvků na formuláři podle potřeby.
- **buttonExport_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonExport". Zobrazuje dialogové okno pro výběr cesty a názvu souboru. Pokud uživatel vybere soubor, metoda exportuje data z tabulky dataGridViewResultApi do CSV souboru na zvolené cestě.
- **Zjistit_async():** Tato asynchronní metoda provádí zjištění informací o součástkách z různých zdrojů. Získává obsah vybraného CSV souboru, zpracovává data a aktualizuje tabulku dataGridViewResultApi s informacemi o cenách a dostupnosti součástek. Také obsahuje logiku pro zobrazení notifikací o levnějších alternativách součástek a vyplňování tabulky dataGridViewComponents po úspěšném zjištění informací.
- **Zjistit_async(CancellationToken token):** Jedná se o přetíženou metodu Zjistit_async(). Tato asynchronní metoda slouží k získání informací o součástkách a jejich vypsání do tabulky. Přijímá parametr token typu CancellationToken, který umožňuje zrušení operace. Metoda získává obsah vybrané tabulky, zpracovává data a aktualizuje tabulku dataGridViewResultApi. Obsahuje také logiku pro zobrazení notifikací o levnějších alternativách součástek a vyplnění tabulky dataGridViewComponents po úspěšném zjištění informací. Metoda se používá při jednorázovém zjišťování informací o součástkách, přijímá informace o stisknutí tlačítka buttonCancel pomocí tokenu CancellationToken a v případě jeho stisknutí zastavuje svůj běh.
- **buttonCancel_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonCancel". Zruší běžící operaci jednorázového zjišťování informací, která je prováděna v metodě Zjistit_async(CancellationToken token) s pomocí CancellationTokenSource _cancellationTokenSource.
- **setComponentsAll():** Tato metoda vyplňuje tabulku dataGridViewComponents všemi součástkami získanými z databáze. Nejprve vyčistí obsah tabulky a pak naplní řádky daty o všech součástkách.
- **setComponentsSearch(string search):** Tato metoda vyplňuje tabulku dataGridViewComponents součástkami odpovídajícími zadanému hledání. Nejprve

vyčistí obsah tabulky a pak naplní řádky daty o součástkách získaných z databáze na základě zadaného hledání.

- **buttonSearch_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonSearch". Zavolá setComponentsSearch() nebo setComponentsAll() na základě obsahu textového pole pro vyhledávání textBoxSearch.
- **textBoxSearch_KeyDown(object sender, KeyEventArgs e):** Tato metoda je volána při stisknutí klávesy v textovém poli pro vyhledávání textBoxSearch. Pokud je stisknutá klávesa Enter, zavolá setComponentsSearch() nebo setComponentsAll() na základě obsahu textového pole.
- **buttonVyvojCen_Click(object sender, EventArgs e):** Tato metoda je volána po stisknutí tlačítka "buttonVyvojCen". Zobrazuje graf vývoje cen vybrané součástky v novém formuláři GraphForm na základě vybraného řádku v tabulce dataGridViewComponents.

Shrnutí

Třída Form1.cs obsahuje logiku pro ovládání uživatelského rozhraní aplikace. Zahrnuje různé události tlačítek a funkcí pro manipulaci s daty zobrazenými ve formuláři. Základními částmi této třídy jsou:

- **Metody pro obsluhu událostí tlačítek:** Obsahují kód pro reakci na stisknutí tlačítek ve formuláři, jako je např. tlačítko pro export dat, vyhledávání součástek, zastavení operací apod.
- **Metody pro zpracování dat:** Obsahují kód pro získání a zpracování dat ze zdrojů, jako jsou CSV soubory nebo API poskytovatelé součástek. Zpracovaná data jsou následně zobrazena ve formuláři.
- **Metody pro aktualizaci uživatelského rozhraní:** Obsahují kód pro aktualizaci zobrazených dat ve formuláři, jako je například aktualizace tabulek po získání nových informací.
- **Ostatní pomocné metody:** Obsahují pomocné metody pro manipulaci s daty nebo pro ovládání průběhu operací ve formuláři.

Celkově tato třída slouží jako řadič pro ovládání a správu aplikační logiky uživatelského rozhraní aplikace. Zahrnuje metody pro interakci s uživatelem, zpracování dat a aktualizaci zobrazení.

3.2.14 Popis třídy GraphForm

Třída GraphForm poskytuje funkcionalitu pro zobrazení a manipulaci s grafem vývoje cen součástek. Níže je detailní popis konstruktoru, metod a celkového účelu třídy.

Konstruktor

- **public GraphForm()**: Konstruktor inicializuje instanci třídy GraphForm. Při inicializaci načte instanci třídy ComponentDatabase pro manipulaci s databází součástek. Dále nastavuje některé vlastnosti grafu, jako je text a okraj, a konfiguruje vzhled mřížek na ose X a Y.

Metody

- **public GraphForm(int componentId, string manucode, string name)**: Tento konstruktor slouží k vytvoření instance třídy GraphForm s konkrétními parametry součástky.

Nastavuje popisek grafu na základě zadaného výrobce a názvu součástky. Načítá data součástky pro zobrazení v grafu pomocí metody LoadChartData. Aktualizuje text titulu okna.

- **private void LoadChartData(int componentId)**: Metoda načítá data součástky z databáze pro zobrazení v grafu. Pro každý obchod (Farnell, Mouser, TME, SOS) načítá data o cenách součástky a vytváří příslušné série pro graf. Upravuje rozsah os grafu tak, aby obsáhl všechna načtená data ze všech sérií. Přidává všechny série do grafu.

Shrnutí

Třída GraphForm poskytuje uživatelské rozhraní pro zobrazení grafu vývoje cen součástek. Umí zobrazit data o cenách součástek z různých obchodů v čase a poskytuje možnost vyhledávat a porovnávat ceny součástek. Je navržena tak, aby byla uživatelsky přívětivá a intuitivní pro uživatele.

3.2.15 Popis třídy VypisDialogForm

Třída VypisDialogForm obsahuje konstruktor a několik metod, které umožňují inicializaci, manipulaci a obsluhu událostí dialogového okna s textovým výstupem informací o zjištěných změnách cen součástek. Zde je popis jednotlivých částí:

Konstruktor

- **public VypisDialogForm(DateTime date, string[] data)**: Konstruktor inicializuje instanci třídy VypisDialogForm. Přijímá datum a pole řetězců data, které jsou zobrazeny v textovém poli dialogového okna. Během inicializace se nastavuje obsah textového pole na základě poskytnutých dat a umisťuje se tlačítko pro zavření dialogového okna.

Metody

- **public void PridejText(DateTime date, string text)**: Metoda umožňuje přidávat další text do textového pole dialogového okna po jeho vytvoření. Přijímá datum a textový řetězec text, který je přidán na konec textového pole. Pokud již v textovém poli existuje obsah, nový text je přidán na nový řádek za poslední záznam.
- **private void buttonVypisDismiss_Click(object sender, EventArgs e)**: Obslužná metoda pro událost kliknutí na tlačítko pro zavření dialogového okna. Po kliknutí na tlačítko je dialogové okno zavřeno.
- **private void VypisDialogForm_Resize(object sender, EventArgs e)**: Obslužná metoda pro událost změny velikosti dialogového okna. Zajišťuje, že tlačítko pro zavření dialogového okna zůstává vycentrováno v rámci okna.
- **private void CenterButton()**: Metoda sloužící k vycentrování tlačítka pro zavření dialogového okna v rámci okna. Aktualizuje pozici tlačítka vzhledem k velikosti okna.

Shrnutí

Třída VypisDialogForm poskytuje prostředky pro zobrazení dialogového okna s textovým výstupem a umožňuje manipulaci s jeho obsahem pomocí metod pro přidávání textu a obsluhu událostí. Dialogové okno obsahuje textové pole pro zobrazení textu a tlačítko pro jeho zavření.

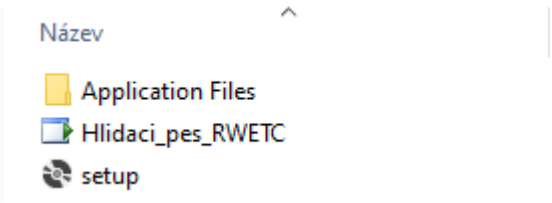
4 Běh programu

Tato kapitola se zaměřuje na detailní popis práce s programem a běhu programu. Cílem je poskytnout jasný přehled o tom, jak aplikace funguje od spuštění až po vykonání hlavních funkcí. Kapitola také představí interakci uživatele s programem, včetně načítání dat, jejich zpracování a prezentace výsledků. Detailní analýza jednotlivých kroků běhu programu přispěje k lepšímu porozumění jeho vnitřní logiky a architektury.

4.1 Instalace programu

Tato část se věnuje procesu instalace programu. Instalace je navržena tak, aby byla co nejjednodušší a nejintuitivnější pro uživatele. Cílem je zajistit, aby uživatelé mohli rychle a bez problémů nainstalovat a začít používat aplikaci, aniž by museli čelit technickým komplikacím.

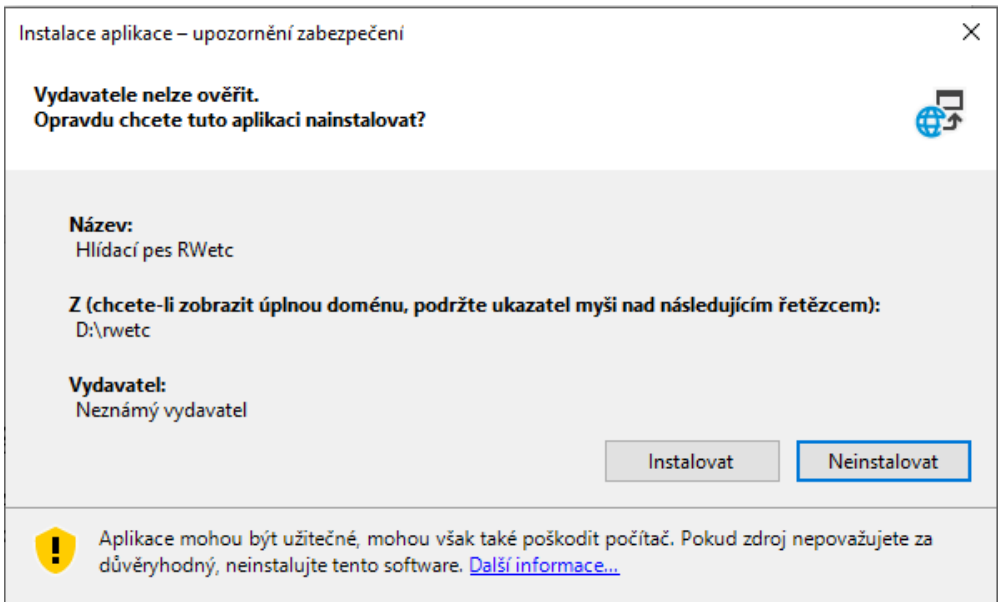
Hotový program byl publikován přes integrovanou funkci „Publikace“ aplikace Visual Studio 2022. Tato funkce vytvoří instalační balíček souborů se zdrojovými soubory.



Obrázek 8: Instalační balíček souborů

Zdroj: vlastní práce

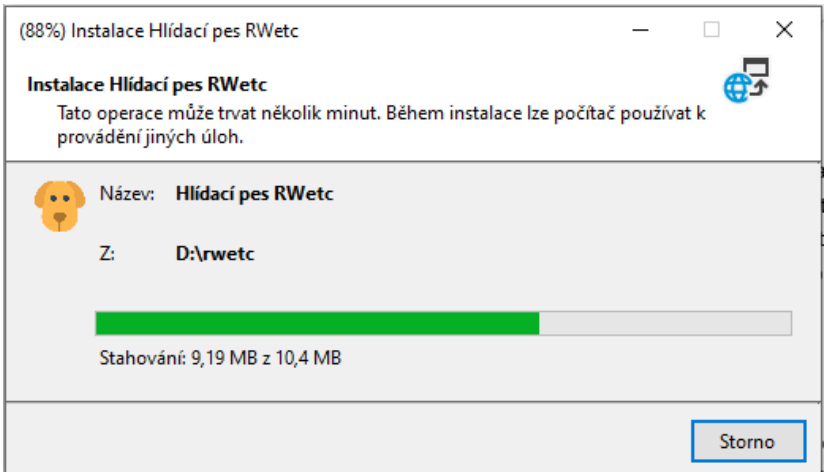
Spuštění instalace proběhne otevřením souboru Hlidaci_pes_RWETC, který zobrazí dialogové okno s potvrzením instalace.



Obrázek 9: Potvrzení instalace

Zdroj: vlastní práce

Po potvrzení proběhne instalace programu.



Obrázek 10: Průběh instalace

Zdroj: vlastní práce

Po dokončení instalace se vytvoří ikona na ploše uživatelské počítače a v nabídce „Start“, kterými je možné program spustit. Tím je instalace dokončena.

4.2 Práce s programem

Tato kapitola se zaměřuje na podrobný popis práce s programem. Cílem této části je poskytnout uživatelům jasné a srozumitelné instrukce k efektivnímu používání programu. Kapitola se bude věnovat jednotlivým funkcionalitám, které program nabízí, a krok za krokem provede procesem práce pro zjištění informací o požadovaných elektronických součástkách.

4.2.1 Vstupní soubor

Vstupem pro zjišťování dat je uživatelem vložený CSV soubor, ve kterém je na každém řádku, (kromě prvního hlavičkového) uvedená jedna hledaná součástka, její kódy a požadované množství.

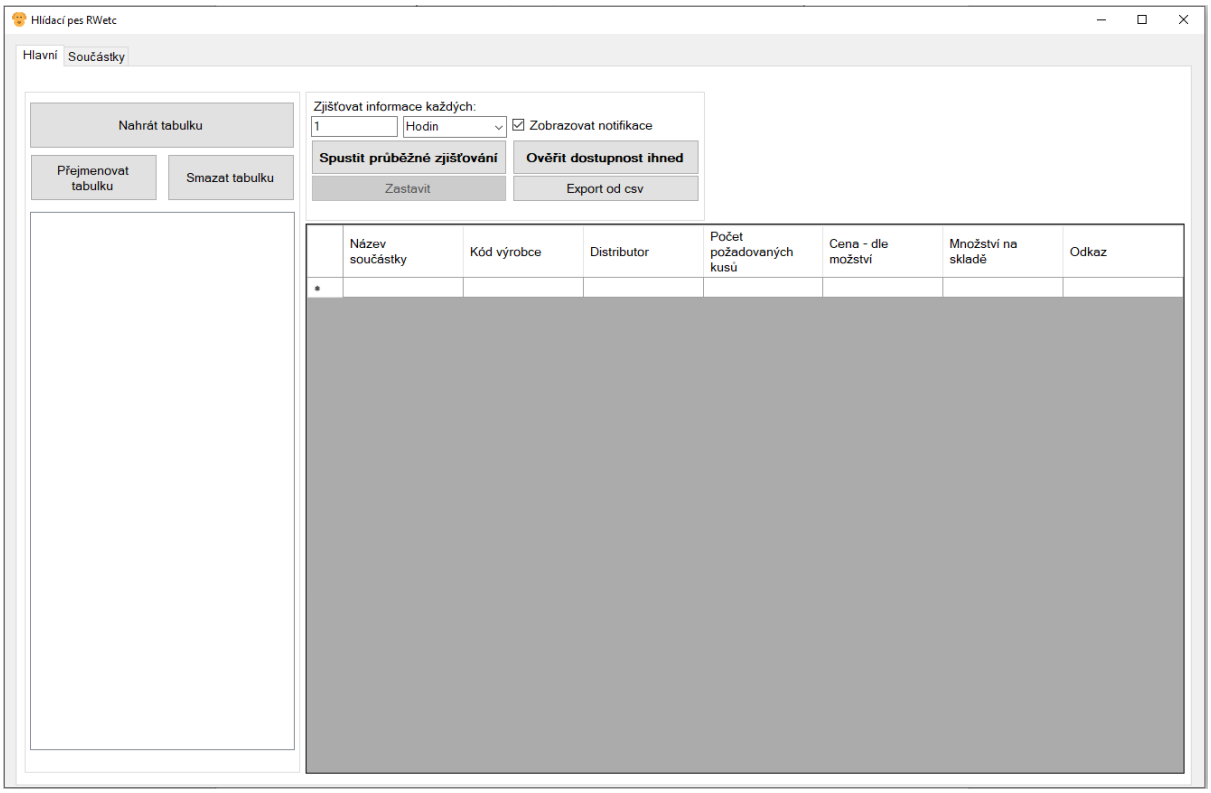
```
KOD VYROBCE;KOD TME ESHOPU;KOD SOS ESHOPU;POZADOVANE MNOZSTVI
2211-12-301;2211-12-301;9398;23
10-2J09.1069;10-2J09.1069;352357;22
```

Obrázek 11: Ukázka obsahu vstupního souboru

Zdroj: vlastní práce

4.2.2 Hlavní část programu

Po spuštění se uživateli otevře okno hlavního programu, které obsahuje dvě záložky. Záložka, na které probíhá hlavní chod programu se nazývá „Hlavní“. Při prvním spuštění neobsahuje program žádné informace o součástkách ani vstupní tabulky pro dotazování.



Obrázek 12: Hlavní část programu po prvním spuštění

Zdroj: vlastní práce

Hlavní část programu je rozdělená na dvě záložky: „Hlavní“ a „Součástky“.

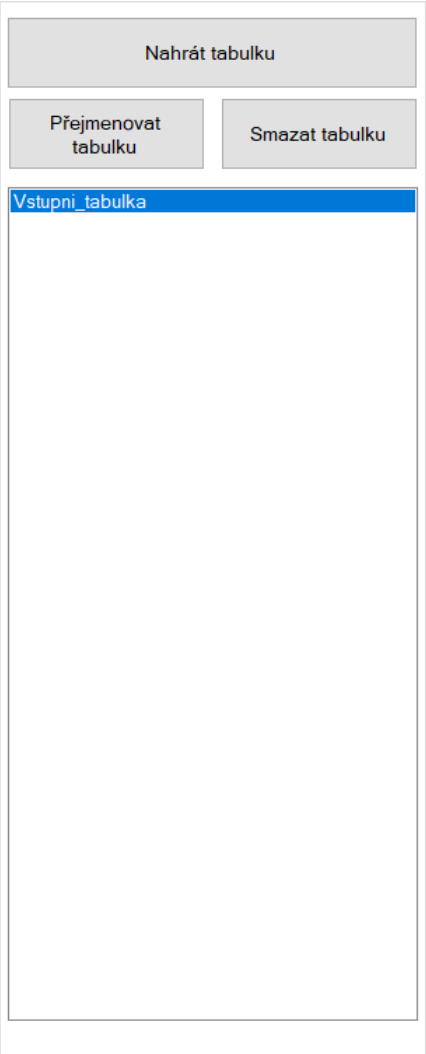
Záložka „Hlavní“ je dále rozdělená na tři sekce:

- sekce vstupů
- sekce ovládání
- sekce pro zobrazení

Sekce vstupů

Sekce vstupů obsahuje tři ovládací tlačítka a výpis vstupních tabulek:

- **Nahrát tabulku:** Tlačítko otevře dialogové okno pro výběr vstupního CSV souboru, po výběru soubor uloží a zobrazí ve výpisu vstupů.
- **Přejmenovat tabulku:** Tlačítko otevře dialogové okno pro zadání nového názvu pro vybranou vstupní tabulku, po vložení nového názvu se zkontroluje, zda již název neexistuje u jiného vstupního souboru, pokud je unikátní v tabulku přejmenuje.
- **Smazat tabulku**: Tlačítko, které vymaže vybranou vstupní tabulku.
- **Výpis tabulek:** Zobrazuje všechny nahrané vstupní tabulky. Aktualizuje se po každém vložení nové tabulky, přejmenování nebo smazání tabulky.



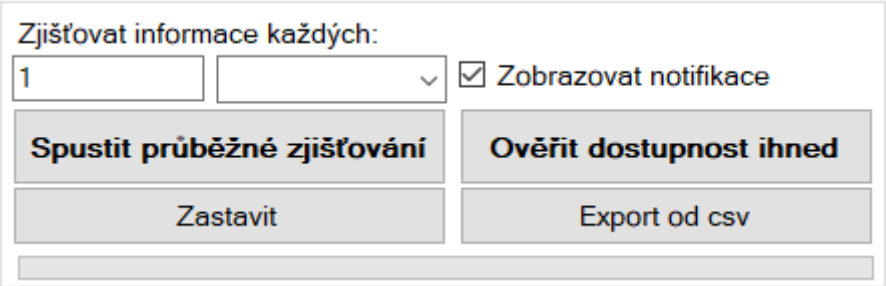
Obrázek 13: Sekce vstupů
Zdroj: vlastní práce

Sekce ovládání

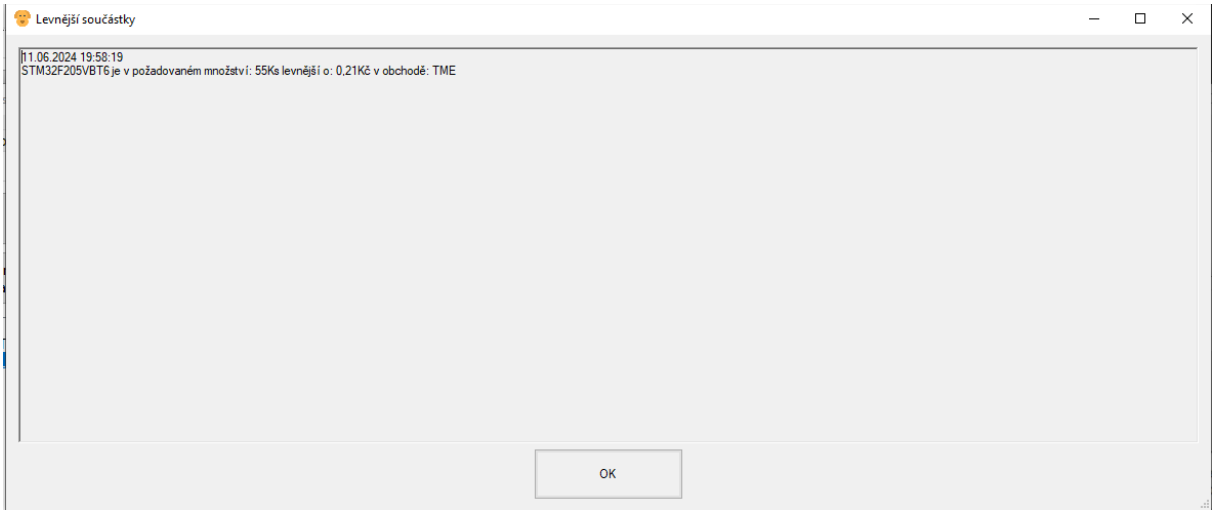
Sekce ovládání obsahuje čtyři tlačítka, textové pole, combobox, checkbox a progressbar:

- **Textové pole:** Vstup slouží k zadání časového hodnoty v číselném formátu. Pro nastavení intervalu při průběžném dotazování na e-shopy. Přijímá pouze číselné hodnoty.
- **Combobox:** Slouží k výběru časové jednotky, pro nastavení intervalu při průběžném dotazování na e-shopy.
- **Checkbox „Zobrazovat notifikace“:** Nastavuje možnost zapnout zobrazování vyskakovacího okna s upozorněním na levnější ceny součástek dle zjištěných informací. Výchozí nastavení této možnosti je ve stavu zapnuta.
- **Spustit průběžné zjišťování:** Tlačítko po stisknutí ihned zjistí informace o součástkách, a následně pravidelně dotazuje tyto informace dle kombinace zadaných hodnot z textového vstupu a vstupu comboboxu.
- **Zastavit:** Tlačítko po stisknutí zastaví proces průběžného zjišťování. Tlačítko je výchozím stavu nepřístupné, zpřístupní se pouze po zapnutí průběžného zjišťování a po jeho zastavení se opět znepřístupní.

- **Progressbar:** Zobrazuje průběh intervalu průběžného zjišťování dle kombinace zadaných hodnot z textového vstupu a vstupu comboboxu. Defaultně je skrytý, zobrazuje se pouze pokud je spuštěn proces průběžného zjišťování.
- **Ověřit dostupnost ihned:** Tlačítko, které jednorázově zjistí informace o součástkách z vybraného vstupního souboru.
- **Export do csv:** Tlačítko po stisknutí vybere informace z tabulky v sekci zobrazení, otevře dialogové okno pro uložení souboru a po potvrzení cesty a názvu vytvoří CSV soubor s obsahem tabulky se zjištěnými informacemi o součástkách.



Obrázek 14: Sekce ovládání
Zdroj: vlastní práce



Obrázek 15: Vyskakovací okno s informacemi o levnějších součástkách
Zdroj: vlastní práce

Sekce pro zobrazení

Sekce pro zobrazení obsahuje tabulku, tlačítko a textové pole.

- **Tabulka:** Zobrazuje zjištěné informace o dotazovaných součástkách. Pokud je zjištěná cena u součástky v konkrétním e-shopu nižší než v posledním dostupném záznamu, podbarví se řádek v tabulce zeleně. Data se vypisují do sloupečků:
  - Název součástky
  - Kód výrobce
  - Distributor
  - Počet požadovaných kusů

- Cena – dle množství
- Množství na skladě
- Odkaz

- **Textové pole:** Zobrazuje informaci o tom, že právě probíhá dotazovaní na e-shopy. Ve výchozím stavu je skryté. Je zobrazeno pouze v průběhu jednorázového ověřování dostupnosti, spuštěného tlačítkem „Ověřit dostupnost ihned“. Slouží jako indikátor, že program zjišťuje požadované informace, protože může trvat několik vteřin, než se vrátí odpovědi z API jednotlivých e-shopů.
- **Zrušit:** Tlačítko, které v případě stisknutí zruší proces jednorázového ověřování. Ve výchozím stavu je skryté. Je zobrazeno pouze v průběhu jednorázového ověřování dostupnosti, spuštěného tlačítkem „Ověřit dostupnost ihned“. Existuje proto, aby uživatel v případě dlouhé prodlevy odpovědi z API jednotlivých e-shopů mohl přerušit tento úkon.

Obrázek 16: Sekce pro zobrazení v průběhu jednorázového dotazování

Zdroj: vlastní práce

Název součástky	Kód výrobce	Distributor	Počet požadovaných kusů	Cena - dle možství	Množství na skladě	Odkaz
STMICROELECT...	STM32F205VBT6	Farnell	55	Požadované mno...	0	https://cz.farnell.c...
STMICROELECT...	STM32F205VBT6	Mouser	55	184,7	624	https://cz.mouser....
STMICROELECT...	STM32F205VBT6	TME	55	173,5	0	https://www.tme.e...
STM32F205VBT6	T1193161	SOS	55	Požadované mno...	0	https://www.sosel...

Obrázek 17: Tabulka výpisu informací o zjištěných součástkách

Zdroj: vlastní práce

4.2.3 Součástky

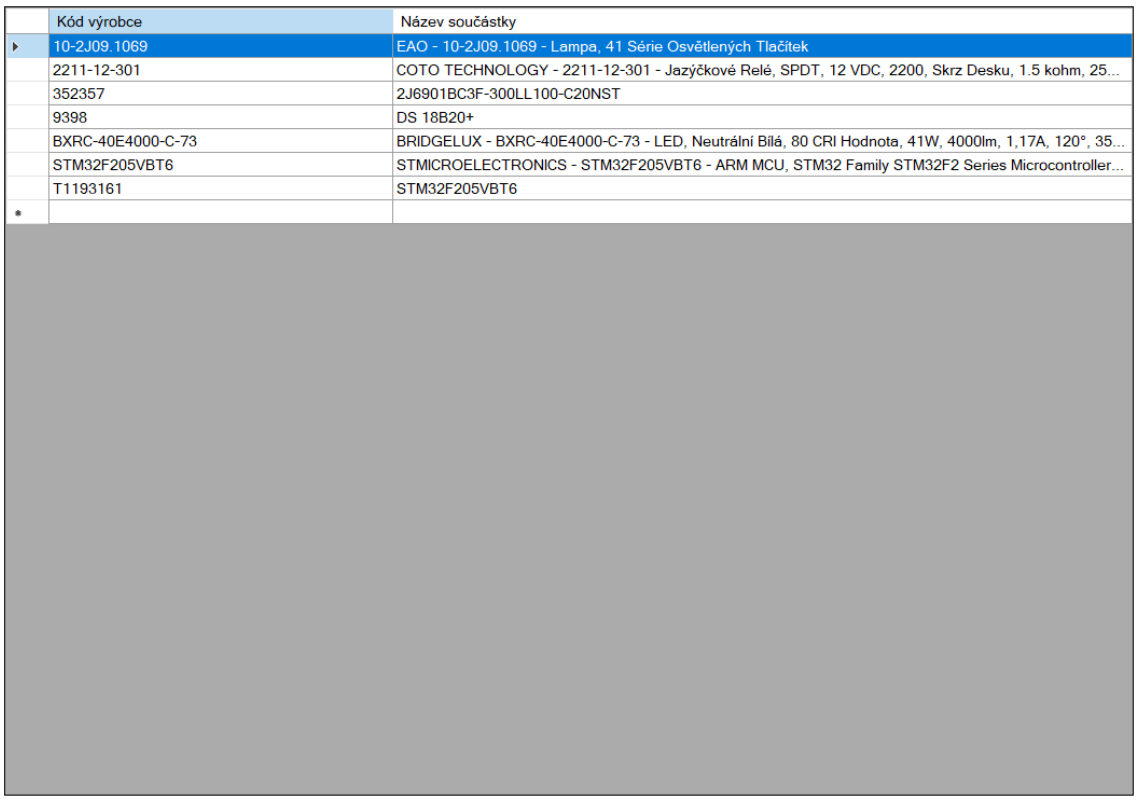
Druhou záložkou ve spuštěném programu je záložka „Součástky“. Tato záložka obsahuje výpis všech součástek, které byly historicky nalezeny a umožňuje zobrazovat údaje o vývoji jejich cen. Záložka je rozdělena na dvě sekce:

- Sekce pro zobrazení
- Sekce ovládání

Sekce pro zobrazení

Sekce pro zobrazení obsahuje tabulku.

- **Tabulka:** Ve výchozí pozici zobrazuje seznam všech součástek již zjištěným předchozím dotazováním. Případně výpis součástek, které obsahují vyhledávanou frázi. Data se vypisují do sloupečků:
    - Kód výrobce
    - Název součástky



Kód výrobce	Název součástky
10-2J09.1069	EAO - 10-2J09.1069 - Lampa, 41 Série Osvětlených Tlačítek
2211-12-301	COTO TECHNOLOGY - 2211-12-301 - Jazýčkové Relé, SPDT, 12 VDC, 2200, Skrz Desku, 1.5 kohm, 25...
352357	2J6901BC3F-300LL100-C20NST
9398	DS 18B20+
BXRC-40E4000-C-73	BRIDGELUX - BXRC-40E4000-C-73 - LED, Neutrální Bílá, 80 CRI Hodnota, 41W, 4000lm, 1,17A, 120°, 35...
STM32F205VBT6	STMICROELECTRONICS - STM32F205VBT6 - ARM MCU, STM32 Family STM32F2 Series Microcontroller...
T1193161	STM32F205VBT6

Obrázek 18: Tabulka součástek

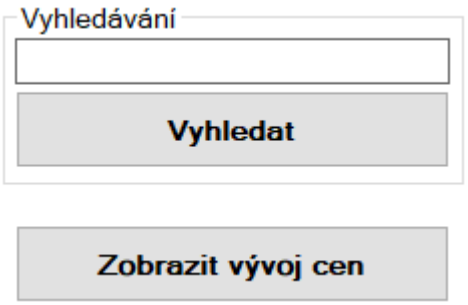
Zdroj: vlastní práce

Sekce ovládání

Sekce ovládání obsahuje vstupní textové pole a dvě tlačítka.

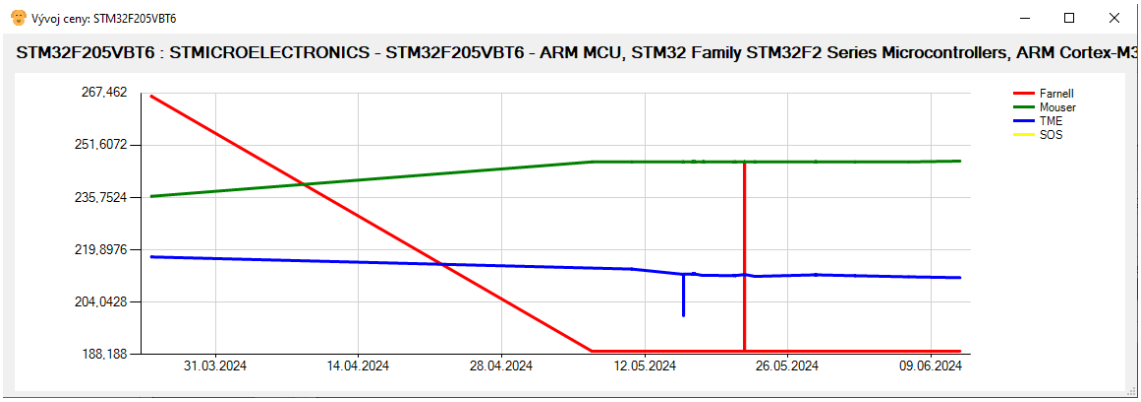
- **Textové pole:** Slouží jako vstup pro vyhledávání pouze určitých součástek, závislých na zadané frázi. Pokud je pole prázdné, zobrazí všechny existující součástky.
- **Vyhledat:** Tlačítko, které po stisknutí příjme vstup z textového pole a spustí proces vyhledávání součástek. Které obsahují buď v kódu nebo v názvu požadovanou frázi. Po dokončení procesu se nalezené součástky zobrazí v tabulce.

- **Zobrazit vývoj cen:** Tlačítko, které otevře nové dialogové okno, ve kterém je zobrazen graf vývoje cen, dle vybrané součástky z tabulky. Graf zobrazuje vývoj cen dle nalezených hodnot z dostupné historie vyhledávání součástky.



Obrázek 19: Sekce ovládání v části součástek

Zdroj: vlastní práce



Obrázek 20: Dialogové okno se zobrazeným grafem vývoje cen součástky

Zdroj: vlastní práce

Závěr

4.3 Možnosti rozšíření

Program Hlídací pes RWEtc poskytuje užitečné nástroje pro monitorování a porovnávání cen a dostupnosti elektronických součástek. Nicméně, existuje několik možností, jak by mohl být program dále rozšířen a vylepšen, aby lépe sloužil potřebám uživatelů. Níže jsou popsány některé klíčové možnosti rozšíření tohoto programu.

1. Integrace s dalšími distributory

V současné verzi program podporuje omezený počet distributorů. Rozšíření na další významné distributory by mohlo zvýšit užitečnost programu. To zahrnuje nejen přidání dalších distributorů, ale také zajištění automatické aktualizace seznamu distributorů a jejich cenových a skladových informací.

2. Pokročilá analýza dat

Aktuální program nabízí základní zobrazení dat v tabulkové a grafické formě. Přidání pokročilých analytických funkcí, jako jsou trendy cen, analýza dostupnosti, by mohlo uživatelům poskytnout cenné informace pro jejich rozhodování.

3. Upozornění a notifikace

Rozšíření systému notifikací by umožnilo uživatelům přijímat upozornění na základě vlastních preferencí. Uživatelé by mohli nastavit specifické podmínky, za kterých chtějí být upozorněni, například když cena určité součástky klesne pod určitou hodnotu, nebo když se skladová dostupnost změní.

5. Rozšířené možnosti exportu dat

Kromě současného exportu do CSV by mohly být přidány další formáty exportu, jako je XML, JSON nebo přímý export do formátů podporovaných databázovými systémy. To by uživatelům poskytlo větší flexibilitu při práci s daty mimo program.

6. Vylepšené grafické zobrazení

Zlepšení grafických funkcí programu by zahrnovalo možnost vytváření různých typů grafů a vizualizací, jako jsou sloupcové grafy, koláčové grafy, nebo interaktivní grafy s možností přiblížení a filtrování dat. Takové vizualizace by uživatelům poskytly lepší náhled na data a usnadnily jejich analýzu.

Rozšíření a vylepšení těchto funkcí by nejen zvýšilo efektivitu programu, ale také by mohlo přilákat širší uživatelskou základnu a zajistit lepší adaptaci v různých pracovních prostředích. Vývoj těchto funkcí by měl být prováděn s ohledem na zpětnou vazbu uživatelů, aby co nejlépe odpovídaly jejich potřebám a očekáváním.

4.4 Závěr

Tato bakalářská práce se zaměřila na vývoj a implementaci programu Hlídací pes RWETC, jehož hlavním účelem je monitorování cen a dostupnosti elektronických součástek u různých distributorů. Program byl navržen tak, aby uživatelům poskytoval snadný přístup k aktuálním informacím, čímž jim umožňuje efektivnější rozhodování při nákupu a správě zásob.

V průběhu práce byly popsány klíčové komponenty programu, jeho architektura, způsob implementace a hlavní funkce. Program také nabízí uživatelsky přívětivé rozhraní, které umožňuje jednoduché zobrazení a export dat, což usnadňuje práci uživatelům.

Dále byly navrženy možnosti rozšíření programu, které zahrnují integraci s dalšími distributory, pokročilou analýzu dat, vylepšené notifikace, uživatelská rozhraní, rozšířené možnosti exportu dat a vylepšené grafické zobrazení. Tyto návrhy představují potenciální směry budoucího vývoje programu, které by mohly dále zlepšit jeho funkčnost a uživatelskou zkušenost.

Realizace programu Hlídací pes RWEtc přinesla řadu technických a praktických výzev, které byly úspěšně překonány díky průběžnému zkoumání a hledání řešení. Výsledný program poskytuje spolehlivý nástroj pro monitorování cen a dostupnosti součástek, což přispívá k optimalizaci nákupních procesů a efektivnějšímu řízení zásob ve firmě zadavatele.

Seznam použité literatury

FARNELL. Farnell. *Farnell* [online]. c2023 [cit. 2023-11-24]. Dostupné z: https://farnell.com/

IBM. Explore the UML sequence diagram. *IBM* [online]. 2023 [cit. 2023-12-21]. Dostupné z: https://developer.ibm.com/articles/the-sequence-diagram/

IBM. Use-case diagrams. *IBM* [online]. 2021 [cit. 2023-12-21]. Dostupné z: https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case

MICROSOFT. A tour of the C# language. *Microsoft* [online]. c2023 [cit. 2023-12-19]. Dostupné z: https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/

MICROSOFT. Desktop Guide (Windows Forms .NET). *Microsoft* [online]. c2023 [cit. 2023-12-20]. Dostupné z: https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-8.0

MICROSOFT. What is Visual Studio? *Microsoft* [online]. c2023 [cit. 2023-12-20]. Dostupné z: https://learn.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022

Mouser APIs. *Mouser APIs* [online]. [cit. 2023-11-23]. Dostupné z: https://api.mouser.com/api/docs/ui/index?_gl=1*ekvs34*_ga*Nzc2MTQ1MTM0LjE2OTg3NjcyNzM.*_ga_15W4STQT4T*MTY5ODc2NzI3Mi4xLjEuMTY5ODc2NzMyOS4zLjAuMA..

MOUSER ELECTRONICS, INC. Mouser electronics. *Mouser electronics* [online]. c2023 [cit. 2023-11-24]. Dostupné z: https://mouser.com/

Product Search API (REST) - Description. *Partner.element14.com* [online]. c2011-2023 [cit. 2023-12-18]. Dostupné z: https://partner.element14.com/docs/read/Product_Search_API_REST__Description

SOS ELECTRONIC. SOS electronic. *SOS electronic* [online]. c1991-2023 [cit. 2023-11-24]. Dostupné z: https://www.soselectronic.com/

SOS version 1.0. *SOS version 1.0* [online]. [cit. 2023-11-23]. Dostupné z: https://api-customer.sos.sk/docs

TME. TME electronic components. *TME electronic components* [online]. c2023 [cit. 2023-11-24]. Dostupné z: https://www.tme.eu/

TME API User Manual. In: *TME API User Manual* [online]. s. 1-60 [cit. 2023-11-23]. Dostupné z: https://developers.tme.eu/documentation

Icon-icons.com. Online. Https://icon-icons.com/icon/Animal-Dog/115245. C2021-2024. Dostupné z: https://icon-icons.com/icon/Animal-Dog/115245. [cit. 2024-06-17].

No Package. Online. 2016. Dostupné z: https://docs.google.com/document/d/1Yyq_HN6uw2Q5EqqAvhNbgOragIxqOZ7wC3PsaBWb6Vg/edit#heading=h.xavbkju1i5zb. [cit. 2024-06-17].

Přílohy

Příloha 1: Flash disk se zdrojovými kódy programu, instalačními soubory a testovacím vstupním souborem.