FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Bakalářská práce

Implementace nových nástrojů pro Laird Digitalization Platform: Aplikace pro kontrolu balení výrobků

Studijní program:	B0613A140005 Informační technologie
Studijní obor:	Aplikovaná informatika
Autor práce:	**Lukáš Iker**
Vedoucí práce:	Ing. Vratislav Žabka, Ph.D. Ústav nových technologií a aplikované informatiky

Liberec 2026

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL



Zadání bakalářské práce

Implementace nových nástrojů pro Laird Digitalization Platform: Aplikace pro kontrolu balení výrobků

Jméno a příjmení:	**Lukáš Iker**
Osobní číslo:	M23000161
Studijní program:	B0613A140005 Informační technologie
Specializace:	Aplikovaná informatika
Zadávající katedra:	Ústav nových technologií a aplikované informatiky
Akademický rok:	2025/2026

Zásady pro vypracování:

1. Seznamte se s vývojovým prostředím a technologiemi používanými pro tvorbu webových aplikací v prostředí ASP.NET Web Forms s využitím jazyka Visual Basic, JavaScriptu a databázového systému provozovaného na serveru IIS.
2. Navrhněte a realizujte webovou aplikaci pro kontrolu balení výrobků prostřednictvím skenování 2D kódů na balicích štítcích. Aplikace bude sloužit k ověřování správnosti a úplnosti jednotlivých balení dle požadavků společnosti.
3. Navrhněte architekturu systému a implementujte aplikační logiku včetně komunikace mezi uživatelským rozhraním a databází. Dbejte na přehledné a uživatelsky přívětivé rozhraní.
4. Otestujte aplikaci v několika fázích – od testování jednotlivých modulů ve vývojovém prostředí, přes nasazení na testovací server, až po ověření funkčnosti v produkčním prostředí. Vyhodnoťte výsledky testování a navrhněte možná zlepšení či rozšíření aplikace.
5. Výsledkem práce bude plně funkční webová aplikace připravená k provozu a dokumentace popisující návrh, implementaci a testování systému.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	30 – 40 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	čeština

Seznam odborné literatury:

1. Zak, Diane. Programming with Microsoft Visual Basic 2019/2022 (9. vydání, 2022). ISBN-10: 0357674006; ISBN-13: 9780357674000.
2. Lock, Andrew. ASP.NET Core in Action (2nd ed., 2021). ISBN-10: 1617298301; ISBN-13: 9781617298301.
3. Liew, Voon Kiong. Visual Basic 2022 Made Easy (2023). ISBN-13: 9798378920327.

Vedoucí práce:	Ing. Vratislav Žabka, Ph.D. Ústav nových technologií a aplikované informatiky
Datum zadání práce:	17. října 2025
Předpokládaný termín odevzdání:	11. května 2026

L.S.

doc. Ing. Josef Černohorský, Ph.D.
děkan

doc. Ing. Josef Chaloupka, Ph.D.
garant studijního programu

V Liberci dne 17. října 2025

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

7. května 2026 Lukáš Iker

Implementace nových nástrojů pro Laird Digitalization Platform: Aplikace pro kontrolu balení výrobků

Abstrakt

Cílem této bakalářské práce je navrhnout, implementovat a nasadit rozšíření interní aplikační platformy Laird Digitalization Platform ve společnosti Laird Technologies. Výsledkem práce je nový modul BLSPackageBuilder, který slouží k elektronické kontrole balení výrobků prostřednictvím skenování 2D kódů na vnějších a vnitřních balicích štítcích.

Aplikace je postavena na technologii ASP.NET Web Forms s jazykem Visual Basic .NET a databází Microsoft SQL Server. Modul zajišťuje identifikaci operátora, načtení vnějšího obalu, opakované skenování vnitřních balení, validaci správnosti skenovaných dat, detekci duplicitních štítků a sledování průběžného naplnění krabice. Uživatelské rozhraní je přizpůsobeno provozu na průmyslových čtečkách Zebra a je navrženo tak, aby operátor mohl většinu práce provádět pouze pomocí skenování.

Součástí práce je také rozšíření existujícího modulu Dashboards o nový pohled Package Builder určený pro pracovníky oddělení kvality. Tento pohled umožňuje dohledávání historie balení, filtrování validních, duplicitních a chybných záznamů a export dat do souboru Microsoft Excel. Řešení bylo otestováno v lokálním prostředí, na testovacím serveru i v produkčním provozu. Součástí práce je rovněž základní vyhodnocení provozních dat, které ukazuje využitelnost uložených záznamů pro zpětnou analýzu chybovosti balení.

Klíčová slova: Laird Digitalization Platform, BLSPackageBuilder, ASP.NET Web Forms, Visual Basic .NET, Microsoft SQL Server, IIS, kontrola balení, 2D kód, průmyslová čtečka, Zebra, *dashboard* kvality

Implementation of New Tools for the Laird Digitalization Platform: Product Packaging Inspection Application

Abstract

The aim of this bachelor thesis is to design, implement and deploy an extension of the internal Laird Digitalization Platform used by Laird Technologies. The result of the thesis is a new module called BLSPackageBuilder, which provides electronic inspection of product packaging by scanning 2D codes on outer and inner packaging labels.

The application is built using ASP.NET Web Forms, Visual Basic .NET and Microsoft SQL Server. The module provides operator identification, loading of an outer package, repeated scanning of inner packages, validation of scanned data, detection of duplicate labels and monitoring of the current filling progress of the box. The user interface is optimized for industrial Zebra barcode scanners and is designed so that most operations can be performed only by scanning.

The thesis also includes an extension of the existing Dashboards module with a new Package Builder view for quality department employees. This view allows users to search packaging history, filter valid, duplicate and incorrect records, and export data to a Microsoft Excel file. The solution was tested in a local environment, on a test server and in production operation. The thesis also includes a basic evaluation of operational data, showing that the stored records can be used for retrospective analysis of packaging errors.

Keywords: Laird Digitalization Platform, BLSPackageBuilder, ASP.NET Web Forms, Visual Basic .NET, Microsoft SQL Server, IIS, packaging inspection, 2D code, industrial barcode scanner, Zebra, quality *dashboard*

PODĚKOVÁNÍ

Rád bych poděkoval panu Ing. Vratislavu Žabkovi, Ph.D., za odborné vedení, cenné rady a podporu během tvorby této bakalářské práce. Dále bych rád poděkoval panu Bc. Zdeňku Horčičkovi a panu Zdeňku Bínovi za ochotu, konzultace a podporu při realizaci práce, zejména za sdílení praktických znalostí týkajících se interních systémů, výrobních procesů a postupů využívaných ve společnosti. Poděkování patří také společnosti Laird Technologies za umožnění realizace této práce v reálném výrobním prostředí a za poskytnutí potřebných podkladů, technických prostředků a prostoru pro vývoj, testování a nasazení aplikace.

Lukáš Iker

OBSAH

Seznam tabulek **10**

Seznam obrázků **11**

Seznam zkratek **12**

1 Úvod **13**
1.1 Kontext a motivace projektu ... 13
1.2 Zadání projektu ... 14
1.3 Cíle projektu ... 14
1.4 Poznámka ke zdrojovým kódům ... 15

2 Teoretická část **16**
2.1 ASP.NET Web Forms ... 16
2.2 Visual Basic .NET ... 16
2.3 JavaScript ... 17
2.4 Bootstrap a CSS ... 17
2.5 Microsoft SQL Server ... 17
2.6 IIS ... 17
2.7 Průmyslové čtečky Zebra ... 18

3 Návrh řešení **19**
3.1 Přehled platformy ... 19
3.2 Návrh modulu BLSPackageBuilder ... 19
3.2.1 Diagram logiky aplikace ... 20
3.3 Návrh rozšíření *dashboardu* kvality ... 22
3.4 Návrh datové struktury ... 22

4 Implementace **24**
4.1 Struktura souborů modulu ... 24
4.2 Načtení stránky a výběr View ... 24
4.3 Identifikace zařízení pomocí *cookie* ... 25
4.4 Zpracování skenovaného kódu ... 25
4.5 Zpracování vnějšího obalu ... 26
4.6 Zpracování vnitřního balení ... 27
4.7 Asynchronní zpracování pomocí UpdatePanel ... 29
4.8 Zobrazení seznamu naskenovaných balení ... 29
4.9 Klientská logika — potlačení softwarové klávesnice ... 31
4.10 Modální dialogy ... 32

4.11 Implementace pohledu Package Builder 33
4.11.1 Filtrační lišta 34
4.11.2 SQL dotaz a propojení tabulek 35
4.11.3 Tabulkové zobrazení 36
4.11.4 Export do Microsoft Excel 36
4.12 Nasazení aplikace 37

5 Testování 39
5.1 Testování v lokálním prostředí 39
5.2 Testování na testovacím serveru 39
5.3 Testované scénáře balicí aplikace 40
5.4 Testování *dashboardu* Package Builder 41
5.5 Ověření v produkčním prostředí 41
5.6 Výsledky testování 42

6 Vyhodnocení provozních dat 43
6.1 Vývoj duplicitních skenů v čase 44
6.2 Vývoj skenů s nesprávným dílem v čase 44
6.3 Porovnání duplicit a nesprávných dílů 45

7 Závěr 46
7.1 Možnosti dalšího rozvoje 46

8 Použitá literatura 48

SEZNAM TABULEK

3.1 Struktura tabulky BPB_OuterPackages 23
3.2 Struktura tabulky BPB_InnerPackages 23
6.1 Souhrnné statistiky problematických skenů 43

SEZNAM OBRÁZKŮ

3.1 Diagram logiky aplikace BLSPackageBuilder 21

4.1 Diagram logiky zpracování vnitřního štítku 28

4.2 Spuštěná aplikace s rozkliknutým detailem naskenovaného zá-.....30
znamu

4.3 Ukázky chybových modálních dialogů v aplikaci 33

4.4 Pohled Package Builder v modulu Dashboards 34

4.5 Detail levé části filtrační lišty (WOID, Box No, Part Number, Inner 35
Pack No)

4.6 Detail pravé části filtrační lišty (From / To, Status, tlačítka Apply 35
a Clear)

6.1 Počet načtených duplicitních štítků v čase 44

6.2 Počet načtených štítků s nesprávným číslem dílu v čase 45

SEZNAM ZKRATEK

ADO.NET	ActiveX Data Objects .NET (knihovna pro přístup k datům)
ASP.NET	Active Server Pages .NET (webový framework společnosti Microsoft)
BID	Box Identifier (identifikátor vnějšího balení)
BLS	označení výrobní lokace Lisovna
CSS	Cascading Style Sheets (Kaskádové styly)
EID	Employee Identifier (Identifikátor zaměstnance)
ERP	Enterprise Resource Planning (Plánování podnikových zdrojů)
GUID	Globally Unique Identifier (Globálně jedinečný identifikátor)
IBID	Inner Box Identifier (identifikátor vnitřního balení)
IIS	Internet Information Services (webový server společnosti Microsoft)
JS	JavaScript
MES	Manufacturing Execution System (Výrobní informační systém)
QAD	název používaného ERP systému
SQL	Structured Query Language (Strukturovaný dotazovací jazyk)
SVG	Scalable Vector Graphics (škálovatelná vektorová grafika)
URL	Uniform Resource Locator (adresa webového zdroje)
USB	Universal Serial Bus (sériové rozhraní)
VB.NET	Visual Basic .NET
WOID	Work Order Identifier
XLSX	Office Open XML Spreadsheet (formát sešitu Microsoft Excel)

1 ÚVOD

Digitalizace výrobních procesů je v průmyslovém prostředí důležitým nástrojem pro zvýšení spolehlivosti, dohledatelnosti a efektivity práce. Ruční nebo papírové postupy jsou postupně nahrazovány elektronickými systémy, které umožňují kontrolovat jednotlivé kroky výroby v reálném čase a zároveň vytvářet elektronickou stopu pro pozdější vyhodnocení.

Tato bakalářská práce se zabývá návrhem, implementací a nasazením webové aplikace **BLSPackageBuilder**, která vznikla jako nový modul interní platformy **Laird Digitalization Platform** ve společnosti Laird Technologies. Úkolem aplikace je elektronická kontrola balení výrobků pomocí skenování 2D kódů na vnějších a vnitřních balicích štítcích.

Řešení je určeno pro použití přímo ve výrobním prostředí. Operátor pracuje s průmyslovou čtečkou od společnosti Zebra, pomocí které skenuje zaměstnanecký štítek, štítek vnějšího obalu a následně jednotlivá vnitřní balení. Aplikace ověřuje správnost dílu, detekuje duplicitní skeny a sleduje naplnění krabice podle plánovaného množství.

Součástí práce je také rozšíření existujícího modulu **Dashboards** o nový pohled **Package Builder** určený pro pracovníky oddělení kvality. Tento pohled umožňuje dohledávat historii balení, filtrovat záznamy podle vybraných parametrů a exportovat data do tabulkového souboru pro další zpracování.

Práce je rozdělena do několika částí. Nejprve je popsán kontext a motivace projektu, zadání a cíle práce. Následuje stručný přehled použitých technologií, návrh řešení, návrh datové struktury a implementace hlavních částí aplikace. Samostatná kapitola je věnována testování v lokálním, testovacím a produkčním prostředí. V další části jsou vyhodnocena provozní data získaná po nasazení aplikace. Závěr shrnuje dosažené výsledky a uvádí možné směry dalšího rozvoje.

1.1 Kontext a motivace projektu

Společnost Laird Technologies se zabývá výrobou součástek určených především pro elektrotechnický a automobilový průmysl. Vzhledem k rozsahu výroby je pro společnost důležité, aby jednotlivé výrobní a kontrolní operace byly přehledné, dohledatelné a co nejméně závislé na ručním ověřování.

Pro většinu hlavních podnikových procesů společnost využívá ERP systém QAD. Tento systém však nepokrývá všechny lokální požadavky jednotlivých výrobních oblastí. Z tohoto důvodu vznikla v roce 2018 v liberecké pobočce vlastní apli-

kační platforma s názvem **Laird Digitalization Platform**, která formou menších dedikovaných aplikací doplňuje ERP systém o chybějící funkcionalitu.

V souvislosti s požadavky na sledovatelnost výrobků a snižování chybovosti vznikla potřeba vyvinout nový modul, který by zajišťoval elektronickou kontrolu balení výrobků na výstupu z linky. Tato kontrola má eliminovat chybně sestavená balení, snížit počet reklamací ze strany zákazníků a zajistit elektronickou stopu o průběhu balení každé krabice.

1.2 Zadání projektu

Hlavním zadáním projektu je navrhnout a implementovat webovou aplikaci s pracovním názvem BLSPackageBuilder jako nový modul již existující platformy, která:

- Validuje přihlášeného uživatele.
- Umožňuje načtení či založení vnějšího obalu (krabice) skenováním 2D kódu na jeho štítku.
- Opakovaně umožňuje přijímat a validovat skeny vnitřních balení (štítky jednotlivých dílů).
- Ověřuje, že se jedná o správný díl, který je určen do daného balení, detekuje duplicitní skeny vnitřních štítků a sleduje postup plnění krabice.
- Po naplnění krabice signalizuje uživateli ukončení balení a informuje ho o postupu k přechodu na další balení.

Mezi vedlejší zadání projektu patří:

- Rozšíření existujícího modulu **Dashboards** o nový pohled **Package Builder** určený pro pracovníky oddělení kvality. Tento pohled umožní filtrovat a prohlížet historii balení podle zadaných parametrů a exportovat zobrazená data do souboru Microsoft Excel.

1.3 Cíle projektu

Hlavním cílem práce je úspěšně navrhnout, vytvořit a nasadit novou aplikaci do výrobního prostředí a zajistit její plynulé propojení se stávající platformou. Konkrétní cíle lze shrnout takto:

- **Vytvoření nové aplikace pro kontrolu balení.** Aplikace bude schopna zpracovávat 2D kódy ve formátu specifickém pro společnost a v reálném čase validovat skenovaná data oproti databázi.
- **Integrace s existující platformou.** Aplikace bude využívat sdílené vnitřní knihovny, například knihovnu `Commons` pro práci s databází. Zároveň bude používat jednotnou databázi platformy a stejný způsob nasazení jako ostatní moduly.
- **Optimalizace pro průmyslové čtečky Zebra.** Uživatelské rozhraní musí být přizpůsobeno omezenému displeji čteček, které se ve společnosti využívají, a musí spolehlivě fungovat bez nutnosti používat fyzickou klávesnici. Ta by v provozu zbytečně komplikovala práci s dotykovým rozhraním.

- **Rozšíření *dashboardu* kvality.** Pracovníci kvality musí mít k dispozici přehledný nástroj pro dohledávání záznamů o balení, identifikaci chybných skenů a export dat pro další analýzu.
- **Minimalizace provozních výpadků.** Aplikace bude nasazena tak, aby nebyl narušen běžný chod výroby a celé platformy.

1.4 Poznámka ke zdrojovým kódům

Vzhledem k tomu, že bakalářská práce byla zpracovávána v prostředí soukromé společnosti, byly při tvorbě této práce zohledněny požadavky společnosti na ochranu interního know-how. Kompletní zdrojové kódy a databázové schéma proto nejsou součástí příloh této práce. V textu jsou uvedeny pouze vybrané a předem schválené úseky kódu, které ilustrují navržená řešení, aniž by ohrozily citlivé části interních postupů.

2 TEORETICKÁ ČÁST

Teoretická část práce poskytuje stručný popis technologií, které byly v rámci projektu využity. Vzhledem k tomu, že nová aplikace navazuje na již existující platformu, byl výběr technologií dán především požadavkem na kompatibilitu se stávajícím prostředím.

2.1 ASP.NET Web Forms

ASP.NET Web Forms je *framework* od společnosti Microsoft, který umožňuje tvorbu webových aplikací založenou na principu oddělených stránek (`.aspx`) a souborů kódu na pozadí (`.aspx.vb`). Tento *framework* využívá serverové ovládací prvky, událostní model a mechanismy pro uchování stavu stránky mezi jednotlivými požadavky [1].

Díky této architektuře lze snadno propojit uživatelské rozhraní přímo s aplikační logikou, databází a dalšími zdroji, spravovat stav pomocí *session* a využívat vestavěné ovládací prvky, například `GridView`, `MultiView` nebo `ListView`.

I když je tato technologie v nových projektech využívána méně často, v podnikovém prostředí se stále objevuje díky své kompatibilitě se staršími systémy. V kontextu této práce je její volba dána tím, že celá platforma byla v minulosti postavena právě na ASP.NET Web Forms a všechny existující moduly tuto technologii využívají.

Klíčovým mechanismem *frameworku* jsou serverové ovládací prvky, které jsou definovány v souboru `.aspx` a zpracovávány na straně serveru. Stav těchto prvků mezi jednotlivými požadavky je uchováván prostřednictvím mechanismu ViewState. Pro asynchronní aktualizaci části stránky bez nutnosti jejího úplného znovunačtení nabízí *framework* komponentu `UpdatePanel` ve spolupráci s objektem `ScriptManager` [1]. Tento mechanismus byl v aplikaci využit pro plynulé zpracování skenovaných dat bez rušivého překreslení celé stránky.

2.2 Visual Basic .NET

Visual Basic .NET, zkráceně VB.NET, je objektově orientovaný programovací jazyk od společnosti Microsoft určený pro vývoj aplikací na platformě .NET [2]. V projektu a v celé platformě je VB.NET používán pro psaní serverové logiky aplikací.

Syntaxe jazyka je navržena tak, aby byla čitelná a přístupná i méně zkušeným programátorům, přičemž nabízí podporu objektově orientovaného programování včetně dědičnosti, rozhraní a generik [3], [4]. Stejně jako ASP.NET Web Forms je VB.NET na platformě využíván od jejího začátku, tudíž nebylo třeba jej měnit.

2.3 JavaScript

JavaScript je skriptovací jazyk používaný zejména na straně klienta, kde zajišťuje interaktivitu a dynamické chování webových aplikací [5]. V této práci je JavaScript využit ke třem hlavním účelům:

- Potlačení softwarové klávesnice na čtecích zařízeních Zebra pro udržení aktivního skenovacího pole.
- Zobrazování modálních dialogů jako zpětné vazby pro uživatele po skenech.
- Dynamické ovládání filtračních prvků v pohledu **Package Builder** modulu **Dashboards**.

Díky JavaScriptu lze výrazně zlepšit uživatelský dojem a celkovou responzivitu aplikace, protože část chování probíhá přímo v prohlížeči bez nutnosti úplného znovunačítání stránky.

2.4 Bootstrap a CSS

Bootstrap je open-source *framework* pro tvorbu responzivního uživatelského rozhraní. Obsahuje připravené styly a komponenty pro rozvržení stránky, formuláře, tlačítka nebo modální dialogy [6]. V tomto projektu byl Bootstrap využit ve verzi 5.3, a to zejména pro modální dialogy, potvrzení chybových hlášení a základní rozvržení stránky. Bootstrap je do projektu načítán z lokální sítě.

Vlastní vizuální styl modulu je definován v souboru `scan.css`, který obsahuje barvy, rámečky, stínování a typografii. Tento přístup centralizace stylů do jednoho souboru byl převzat z ostatních modulů platformy. Jakákoliv vizuální úprava se proto provádí změnou v tomto souboru.

2.5 Microsoft SQL Server

Microsoft SQL Server je relační databázový systém od společnosti Microsoft určený pro ukládání, správu a dotazování nad daty pomocí jazyka SQL [7]. V prostředí .NET je komunikace s databází často realizována pomocí knihoven ADO.NET.

ADO.NET je sada tříd pro přístup k datům v prostředí .NET Framework a umožňuje aplikacím pracovat s relačními databázemi a dalšími datovými zdroji [8]. V rámci platformy je využívána přímá komunikace přes ADO.NET prostřednictvím sdílené pomocné knihovny `Commons`, která zapouzdřuje metody pro práci s databází, například `GetDSFromSelect` nebo `ExecuteSQLnQ`, a poskytuje další pomocné funkce.

2.6 IIS

IIS (Internet Information Services) je webový server společnosti Microsoft určený pro hostování webových aplikací a služeb v prostředí Windows [9]. V kontextu této práce není konfigurace serveru měněna, protože platforma na něm již běží a nový modul se na ni pouze připojuje jako nová složka v rámci existujícího projektu.

2.7 Průmyslové čtečky Zebra

Průmyslové čtečky Zebra jsou specializovaná zařízení určená pro skenování čárových a 2D kódů ve výrobním a logistickém prostředí. V závodu jsou nasazeny modely s operačním systémem Android, konkrétně modely TC22, které disponují dotykovou obrazovkou, skenovacím tlačítkem a interním skenovacím modulem [10].

Dokumentace zařízení TC22/TC27 popisuje také technologii DataWedge, která zajišťuje zpracování načtených čárových kódů a jejich předání do cílové aplikace [10]. Pro účely této práce je zásadní, že čtečka odesílá naskenovaný 2D kód do aktivního textového pole jako sekvenci znaků zakončenou znakem konce řádku, podobně jako kdyby uživatel text napsal na klávesnici.

3 NÁVRH ŘEŠENÍ

V této kapitole je popsán návrh modulu BLSPackageBuilder a souvisejícího pohledu Package Builder v rozhraní pro pracovníky kvality.

3.1 Přehled platformy

Platforma představuje vlastní nadstavbu nad ERP systém QAD. Slouží k monitorování výrobních procesů, podpoře bezpapírové výroby a jako objednávkový systém materiálu. Skládá se z několika klíčových modulů, mezi které patří:

- Dashboards — přehledové rozhraní pro vedoucí pracovníky a kvalitu.
- TabletInterface — rozhraní určené pro tablety u jednotlivých strojů.
- PackLabeler — modul pro tisk štítků na vnitřní balení.
- MaterialOrders — interní objednávkový systém materiálů.
- Další interní moduly platformy — MES, vratné obaly a další pomocné nástroje.

Úkolem této práce bylo navrhnout a vytvořit nový modul **BLSPackageBuilder** a rozšířit existující modul **Dashboards** o nový pohled určený pracovníkům oddělení kvality.

3.2 Návrh modulu BLSPackageBuilder

Cílem modulu BLSPackageBuilder je provádět kontrolu při balení výrobků na výstupu z výrobní linky. Operátor pracuje výhradně s průmyslovou čtečkou Zebra a interakce s aplikací probíhá především prostřednictvím skenování 2D kódů na výrobních a zákaznických štítcích. Aplikace musí pracovat ve třech logických fázích:

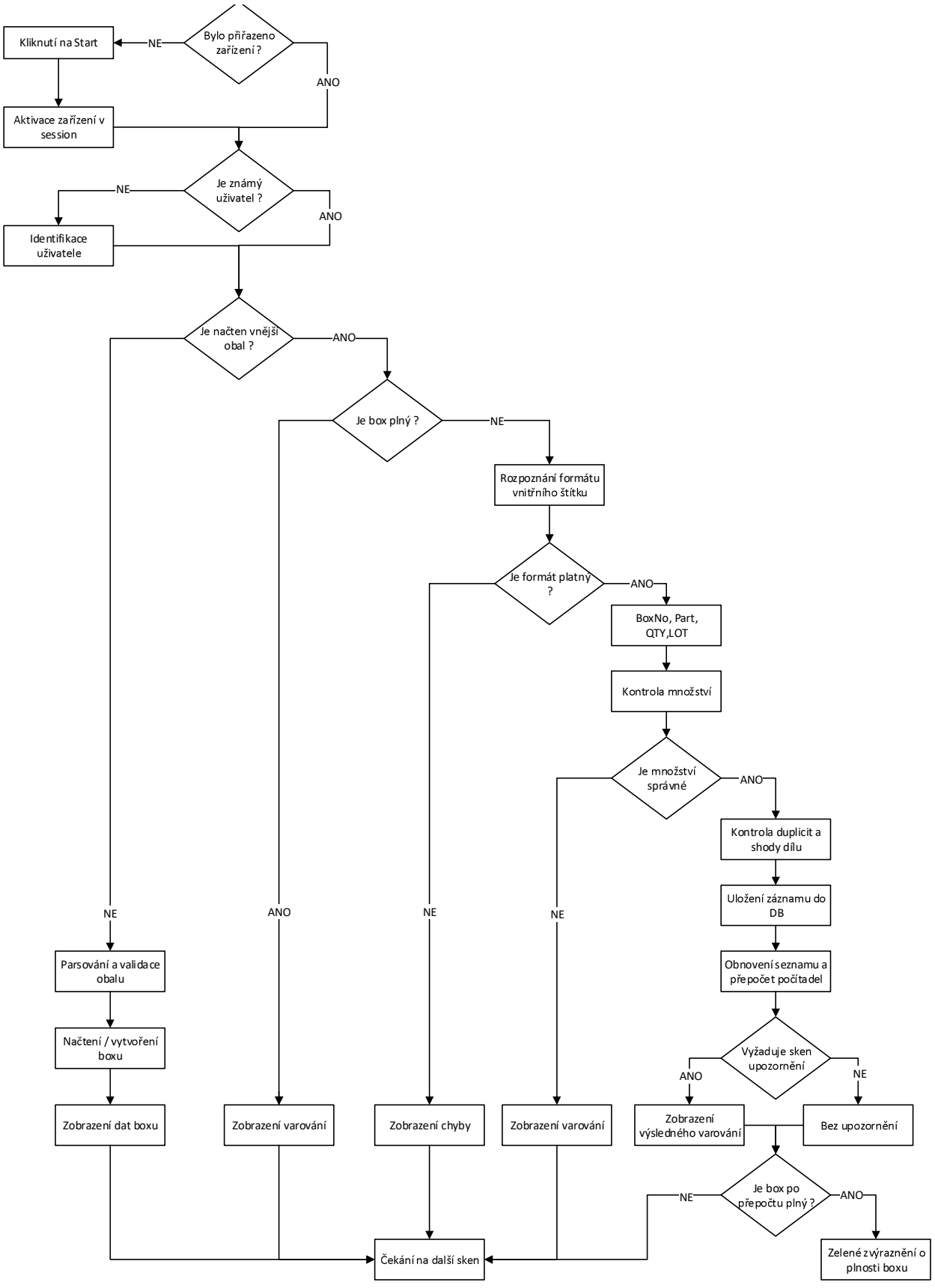
1. **Identifikace uživatele** - po identifikaci zařízení následuje přihlášení operátora skenováním zaměstnaneckého štítku. Toto rozdělení umožňuje, aby na jednom tabletu mohlo postupně pracovat více operátorů.
2. **Vlastní kontrola balení** - po identifikaci operátora se aktivuje hlavní pohled aplikace, ve kterém probíhá celý balicí proces. Operátor nejprve naskenuje vnější obal (krabici), což nastaví referenční číslo dílu a požadovanou kapacitu krabice.
3. **Kontrola vnitřních skenů** - následně uživatel postupně skenuje jednotlivá vnitřní balení, která se aplikace pokouší zařadit do aktuální krabice. Pokud v průběhu procesu vznikne chyba, aplikace o ní uživatele ihned informuje pomocí modálního dialogu.

Při skenování každého vnitřního balení musí aplikace provést tyto kroky validace:

- **Rozpoznání formátu kódu** - vnitřní štítek obsahuje data v interním formátu společnosti, který se odlišuje od formátu vnějšího obalu. Aplikace musí kód rozpoznat, rozparsovat a získat z něj číslo dílu, číslo balení, množství a číslo šarže.
- **Kontrola shody čísla dílu** - číslo dílu z vnitřního štítku se musí shodovat s číslem dílu uvedeným na vnějším obalu. Pokud se neshoduje, jedná se o pokus o vložení nesprávného dílu do krabice a operátor je upozorněn modálním dialogem.
- **Detekce duplicit** - pokud je stejný vnitřní štítek naskenován opakovaně, aplikace tento záznam označí příznakem duplicity a zabrání jeho započtení do plnění krabice. Uživatel je znovu upozorněn na chybu.
- **Kontrola plnění** - aplikace v reálném čase přepočítává součet úspěšně naskenovaných kusů a porovnává jej s plánovanou kapacitou krabice. Při dosažení cílového množství je krabice označena jako plná a operátor obdrží signalizaci o plnosti krabice.

3.2.1 Diagram logiky aplikace

Pro lepší přehlednost je celá logika aplikace shrnuta v diagramu uvedeném na Obrázek 3.1. Diagram zachycuje všechny stavy aplikace — od identifikace zařízení a uživatele přes načtení vnějšího obalu až po jednotlivé kroky validace skenu vnitřního balení a finální signalizaci plné krabice.



Obrázek 3.1: Diagram logiky aplikace BLSPackageBuilder

3.3 Návrh rozšíření *dashboardu* kvality

Pro pracovníky kvality bylo nutné navrhnout takový pohled, který by jim umožnil rychle prohledávat historii skenů, snadno identifikovat chybná balení a získávat tak data pro další analýzu. Pohled má být integrován do existujícího modulu **Dashboards** jako další položka v hlavním navigačním menu pod názvem **Package Builder**.

Klíčové funkce tohoto pohledu jsou:

- **Filtrování záznamů** - podle čísla pracovního příkazu (WOID), čísla boxu, čísla dílu, čísla vnitřního balení a časového rozmezí.
- **Filtrování podle stavu skenu** - možnost vyfiltrovat pouze záznamy s nesprávným číslem dílu, pouze duplicitní skeny, pouze validní záznamy nebo všechny současně.
- **Tabulkové zobrazení** - propojení dat z tabulek vnějších a vnitřních balení s možností řazení podle libovolného sloupce.
- **Stránkování** zobrazených dat pro lepší načítání a plynulou práci s velkým objemem záznamů.
- **Export do tabulkového souboru** - uložení dat do formátu Microsoft Excel pro další zpracování v reportech.

Pohled je navržen tak, aby používal stejné styly jako již existující *dashboardy* a tím zajišťoval plynulý přechod mezi různými typy informačních přehledů.

3.4 Návrh datové struktury

Aplikace pracuje se dvěma novými databázovými tabulkami v rámci databáze platformy. Tyto tabulky slouží k uložení informací o zahájené kontrole balení a o jednotlivých naskenovaných vnitřních štítcích.

První tabulkou je `BPB_OuterPackages`, která ukládá informace o vnějším obalu. Každý záznam představuje jedno zahájené balení. Její struktura je uvedena v Tabulka 3.1.

Tabulka 3.1: Struktura tabulky BPB_OuterPackages

Sloupec	Význam
BID	Jedinečný identifikátor vnějšího balení.
EID	Identifikátor zaměstnance, který kontrolu balení provádí.
Created	Datum a čas vytvoření záznamu.
BoxNo	Číslo vnějšího boxu, tedy krabice.
Part	Číslo baleného dílu.
Quantity	Plánované množství kusů v balení.
LOTNo	Číslo výrobní šarže.
PackDate	Datum balení.
WOID	Číslo pracovního příkazu.

Druhou tabulkou je BPB_InnerPackages, která ukládá jednotlivé naskenované vnitřní štítky vztahující se ke konkrétnímu vnějšímu obalu. Její struktura je uvedena v Tabulka 3.2.

Tabulka 3.2: Struktura tabulky BPB_InnerPackages

Sloupec	Význam
IBID	Jedinečný identifikátor záznamu vnitřního balení.
BID	Odkaz na nadřazené vnější balení v tabulce BPB_OuterPackages.
BoxNo	Číslo vnitřního boxu.
Part	Číslo naskenovaného dílu.
Quantity	Množství uvedené na naskenovaném štítku.
LOTNo	Číslo šarže uvedené na naskenovaném štítku.
DuplicateValidBID	Odkaz na balení, ve kterém byl nalezen stejný štítek již dříve naskenovaný jako validní; pokud se nejedná o duplicitu, hodnota je NULL.
ScannedOn	Datum a čas provedení skenu.

Vazba mezi tabulkami je realizována pomocí sloupce BID. Hodnota BID v tabulce BPB_InnerPackages odkazuje na konkrétní záznam v tabulce BPB_OuterPackages. Díky tomu je možné ke každému vnějšímu balení dohledat všechny vnitřní štítky, které byly během balení naskenovány.

Nesprávně naskenované díly jsou v tabulce BPB_InnerPackages označeny tak, že jejich číslo boxu je doplněno o příznak -X na konci. Tento přístup umožňuje jednoduché filtrování chybných záznamů pomocí SQL dotazu LIKE '%-X'.

4 IMPLEMENTACE

Implementace aplikace navazuje na již existující prostředí, proto byla při výběru technologií preferována kompatibilita a co nejmenší zásah do stávajícího řešení. Jádro aplikace je provozováno na ASP.NET Web Forms s VB.NET a jako databáze je používán Microsoft SQL Server.

Hlavní výzvou implementace byla adaptace na nenormalizovanou databázi platformy, která je postavena na tabulkách a pohledech bez pevného ER diagramu, a dále specifické požadavky na chování uživatelského rozhraní na čtečkách Zebra. Níže jsou popsány klíčové části implementace.

4.1 Struktura souborů modulu

Modul BLSPackageBuilder se skládá z následujících souborů:

- `Default.aspx` — prezentační vrstva, definuje uživatelské rozhraní pomocí ASP.NET serverových ovládacích prvků.
- `Default.aspx.vb` — soubor kódu na pozadí (Code-Behind), obsahuje veškerou serverovou logiku zpracování skenů, validaci a komunikaci s databází.
- `scan.css` — vlastní stylový předpis pro modul.
- `app.js` — klientský JavaScript zajišťující ovládání skenovacího pole a modálních dialogů.

Bootstrap 5.3 je načítán z lokální sítě, čímž je odstraněna nutnost načítat knihovnu z online prostředí.

4.2 Načtení stránky a výběr View

Stránka využívá komponentu `MultiView` se třemi hlavními pohledy:

- `VTabletIdentification` — samotné spuštění aplikace.
- `VUserIdentification` — identifikace uživatele dle jeho pracovního čísla.
- `VPackageBuilder` — hlavní pohled pro práci s aplikací.

Přepínání mezi pohledy je řízeno serverovou logikou v události `LoadComplete` na základě hodnot uložených v *session*. Při zapnutí aplikace se uživateli aktivuje pohled `VTabletIdentification`, kde uživatel stiskne tlačítko **Start**. Pokud není v *session* uloženo identifikační číslo uživatele (`Session("UserID")`), aktivuje se pohled `VUserIdentification`. Po dokončení startu a identifikaci uživatele se aktivuje hlavní pohled aplikace `VPackageBuilder`.

Skenovací textové pole je viditelné pouze v pohledech identifikace uživatele a hlavního pohledu — v pohledu `VTabletIdentification` je tlačítko **Start**, které slouží k inicializaci aplikace.

Ukázka kódu v jazyce VB.NET na Kód 1 zachycuje zjednodušenou logiku výběru aktivního pohledu.

```
Private Sub _Default_LoadComplete(sender As Object, e As EventArgs) _
        Handles Me.LoadComplete
    If String.IsNullOrEmpty(Session("BPBDeviceID")) Then
        MVPackageBuilder.SetActiveView(VTabletIdentification)
    Else
        If String.IsNullOrEmpty(Session("UserID")) Then
            MVPackageBuilder.SetActiveView(VUserIdentification)
        Else
            MVPackageBuilder.SetActiveView(VPackageBuilder)
        End If
    End If
    TextBox.Visible = (MVPackageBuilder.ActiveViewIndex <> 0)
End Sub
```

Kód 1: Výběr aktivního pohledu podle stavu zařízení a uživatele

4.3 Identifikace zařízení pomocí *cookie*

V platformě se běžně používá `tabletID` k identifikaci konkrétního zařízení a k určení, ke kterému stroji je tablet přiřazen. Čtečky používané touto aplikací, ale tímto identifikátorem nedisponují. Pro účely této aplikace není nutné rozlišovat jednotlivé čtečky samostatně; stačí ověřit, že záznam vznikl pomocí zařízení určeného pro balení. Z toho důvodu jsou čtečky identifikovány náhradním způsobem — přes *cookie*, která se vytvoří při prvotní inicializaci stisknutím tlačítka **Start**. Všechny čtečky tak sdílejí stejný identifikátor, což umožňuje potvrdit, že záznam opravdu pochází z některé z nich.

Samotná *cookie* `bpbdevid.lairdplatform` má platnost tři měsíce. V události `Page_Init` se před vlastním zpracováním stránky zkontroluje, zda je *cookie* přítomná, a pokud ano, načte se z ní hodnota a uloží do *session*. Zároveň je platnost *cookie* prodloužena o další tři měsíce, aby přihlášené zařízení zůstalo identifikováno i při delších provozních pauzách.

Tento celý mechanismus zajišťuje, že po prvotní inicializaci čtečky (stisknutí tlačítka Start) je zařízení automaticky rozpoznáno a při každém dalším načtení stránky se operátor může věnovat výhradně skenování bez nutnosti dalších akcí na operátorově straně.

4.4 Zpracování skenovaného kódu

Jádrem celé aplikace je procedura `ProcessScannedData`, která je volána při každé změně hodnoty skenovacího pole. Ke změně může dojít jak načtením 2D kódu pomocí čtečky, tak případně ručním vstupem od uživatele. Procedura podle obsahu načteného řetězce a podle aktuálního stavu aplikace rozhodne, o jaký typ kódu se jedná, a následně provede odpovídající akci.

Aplikace rozlišuje tři základní typy vstupů:

- Identifikátor zaměstnance.
- Štítek vnějšího obalu (krabice).
- Štítek vnitřního balení.

Každé volání procedury zároveň vytvoří unikátní identifikátor požadavku ve formátu GUID. Tento identifikátor je společně s podrobnými informacemi o průběhu zpracování uložen do logovacího souboru. Díky tomu je možné zpětně dohledat konkrétní zpracování skenu v produkčním prostředí. Pokud například operátor nahlásí problém, lze podle času a GUID identifikátoru dohledat přesný záznam v logu a zjistit, jaká data byla načtena, jak byla rozparsována a ve které části zpracování došlo k chybě.

Logy jsou ukládány do textových souborů pojmenovaných podle data ve formátu `blspackagebuilder_yyyy-MM-dd.txt`. Starší logovací soubory jsou po uplynutí 30 dnů automaticky mazány, aby nedocházelo k jejich neomezenému hromadění.

Logovací mechanismus je navržen tak, aby byl přehledný i při větším množství záznamů. Pro každý sken je v logu vytvořena samostatná sekce, která obsahuje začátek zpracování, rozparsované hodnoty z 2D kódu, výsledek validace a případně také informaci o zápisu do databáze.

4.5 Zpracování vnějšího obalu

Při naskenování vnějšího obalu aplikace nejprve rozparsuje obsah 2D kódu na jednotlivé hodnoty. Z kódu jsou získána především tato pole: číslo boxu, číslo dílu, množství, číslo šarže, datum balení a číslo pracovního příkazu (`WOID`).

Po rozparsování následuje validace načtených dat. Aplikace ověřuje, zda jsou přítomna všechna povinná pole a zda číslo pracovního příkazu existuje v databázi. Pokud některá z kontrol selže, uživatel je upozorněn chybovým dialogem a aplikace nepokračuje v založení balení.

Pokud je vnější obal validní, aplikace vytvoří nový záznam v tabulce `BPB_OuterPackages`, případně dohledá již existující záznam, pokud byl daný box načten dříve. Identifikátor tohoto záznamu (`BID`) je následně uložen do skrytého pole `HFBID` na stránce. Toto pole slouží jako vazba mezi aktuálně zobrazeným boxem a všemi následně skenovanými vnitřními baleními.

Identifikátor `BID` je dále použit jako parametr pro datový zdroj seznamu vnitřních balení a jako hlavní klíč pro všechny další operace prováděné nad aktuálním boxem.

Současně jsou základní informace o načteném boxu zobrazeny ve statistickém panelu uprostřed stránky. Operátor tak ihned vidí číslo dílu, plánované množství, aktuální počet zabalených kusů, číslo boxu a zbývající počet kusů do naplnění krabice.

Pokud se jedná o vnější obal, který byl v aplikaci již dříve zpracován, nejsou uživateli zobrazeny pouze základní informace ve statistickém panelu, ale také kontrolní tabulka s historií dosavadních skenů vztahujících se k danému boxu. Operátor tak může přímo na obrazovce vidět, jaké vnitřní štítky již byly k boxu načteny, kolik kusů bylo započteno a zda se v historii daného balení vyskytují chybné nebo duplicitní záznamy.

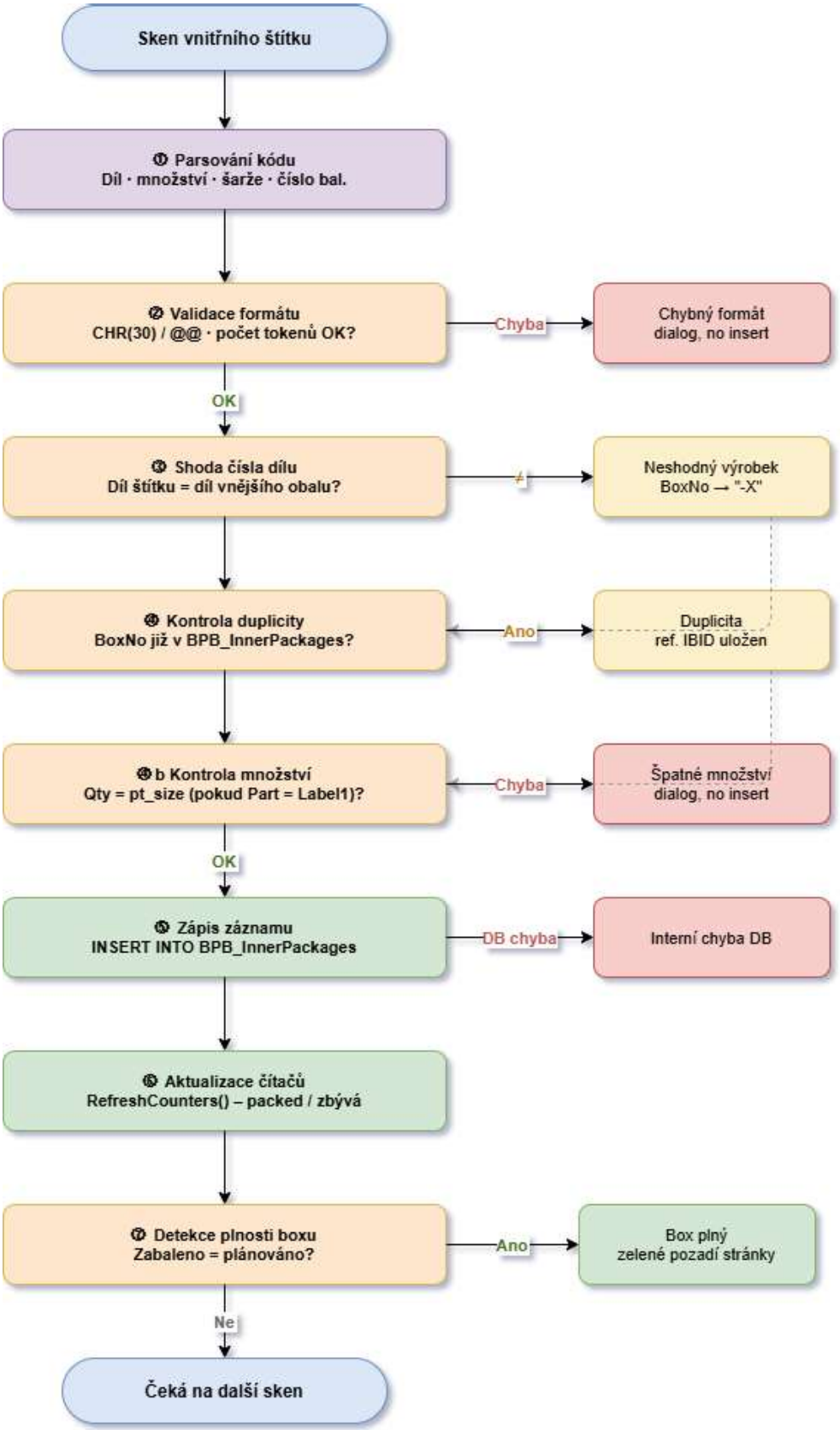
4.6 Zpracování vnitřního balení

Po úspěšném načtení vnějšího obalu aplikace přejde do režimu opakovaného skenování vnitřních balení. V tomto režimu uživatel postupně skenuje jednotlivé štítky dílů, které mají být vloženy do aktuální krabice. Každý sken je samostatně rozparsován, zvalidován a podle výsledku buď uložen jako validní záznam, nebo označen jako chybný.

Při každém skenu vnitřního balení probíhá tato posloupnost kroků:

1. **Parsování kódu** — z načteného řetězce jsou získány hodnoty čísla dílu, množství, čísla šarže a čísla vnitřního balení.
2. **Validace formátu** — aplikace ověřuje, zda se podařilo z kódu získat všechna povinná pole. Pokud formát kódu neodpovídá očekávanému formátu, je zobrazen chybový dialog a žádný záznam se do databáze neukládá.
3. **Kontrola shody čísla dílu** — číslo dílu z vnitřního štítku se porovná s číslem dílu aktuálně načteného vnějšího obalu. Pokud se čísla neshodují, jedná se o pokus vložit do krabice nesprávný díl. V takovém případě je záznam uložen do tabulky `BPB_InnerPackages` s číslem boxu doplněným o příznak `-X`, například `BOX12345-X`, a operátor je upozorněn varovným vyskakovacím dialogem.
4. **Kontrola duplicity** — aplikace ověřuje, zda stejný vnitřní štítek nebyl již dříve naskenován jako validní. Duplicita je vyhodnocována podle kombinace čísla vnitřního balení a čísla šarže. Pokud je duplicita nalezena, nový záznam je označen hodnotou ve sloupci `DuplicateValidBID`, která odkazuje na původní validní balení.
5. **Zápis záznamu** — pokud sken projde všemi kontrolami, je uložen do tabulky `BPB_InnerPackages` jako validní položka aktuálního boxu.
6. **Aktualizace čítačů** — po zápisu skenu do databáze se zavolá procedura `RefreshCounters`, která znovu vypočítá aktuální počet zabalených kusů a počet kusů zbývajících do naplnění boxu.
7. **Detekce plnosti boxu** — pokud počet zabalených kusů dosáhne plánovaného množství, aplikace vyhodnotí box jako plný. Uživateli je zobrazena informační hláška a pozadí stránky se změní na zelenou barvu, aby byl stav úspěšného balení boxu dobře viditelný i v provozním prostředí.

Průběh zpracování každého skenu vnitřního balení je schematicky zachycen na Obrázek 4.1. Diagram zobrazuje všechny rozhodovací kroky včetně chybových větví — při každé neúspěšné kontrole je operátor informován modálním dialogem a záznam není započten do plnění boxu.



Obrázek 4.1: Diagram logiky zpracování vnitřního štítku

Tento postup zajišťuje, že každý štítek projde stejnou sekvencí kontrol a že je v databázi uchována kompletní historie všech provedených skenů. Díky tomu lze zpětně dohledat nejen správně zabalené položky, ale také chybné nebo duplicitní pokusy o skenování.

4.7 Asynchronní zpracování pomocí UpdatePanel

Aby skenování probíhalo plynule a bez zbytečného překreslování celé stránky, je hlavní část aplikace obalena komponentou `UpdatePanel`. Tato komponenta umožňuje, aby se při každém skenu provedl pouze asynchronní *postback* na server. Odpovědí serveru je následně aktualizace pouze vybrané části stránky, nikoliv její úplné znovunačtení.

Tento přístup je v aplikaci důležitý hlavně kvůli provozu na čtečkách. Při úplném překreslení stránky by docházelo k problikávání rozhraní, ztrátě fokusu ze skenovacího pole a v některých případech také k automatickému zobrazení softwarové klávesnice. To by výrazně zpomalovalo celý proces balení, protože operátor by po každém skenu musel znovu řešit stav uživatelského rozhraní.

Triggery komponenty `UpdatePanel` jsou nastaveny především na změnu hodnoty skenovacího pole. Vedle toho jsou do asynchronního zpracování zahrnuta také tlačítka **Odhlásit** a **Změnit box**. Díky tomu lze všechny hlavní operace provádět bez úplného obnovení stránky a aplikace si udržuje plynulé chování i při rychlém opakovaném skenování.

4.8 Zobrazení seznamu naskenovaných balení

Tabulka se seznamem již naskenovaných vnitřních balení je v hlavním pohledu zobrazena pomocí komponenty `ListView`. Tato komponenta je napojena na datový zdroj `SqlDataSource`, který načítá pouze záznamy patřící k aktuálně zpracovávanému vnějšímu balení. Filtrování je provedeno podle hodnoty `BID`, která je uložena ve skrytém poli `HFBID`.

Každý záznam v tabulce je vykreslen jako samostatná karta. Karta obsahuje pořadové číslo skenu, číslo vnitřního balení, množství a stav daného záznamu. Stav je zobrazován zkráceně, například jako `OK`, `Dup`, `Part` nebo `Full`, aby byl přehledný i na menším displeji čtečky.

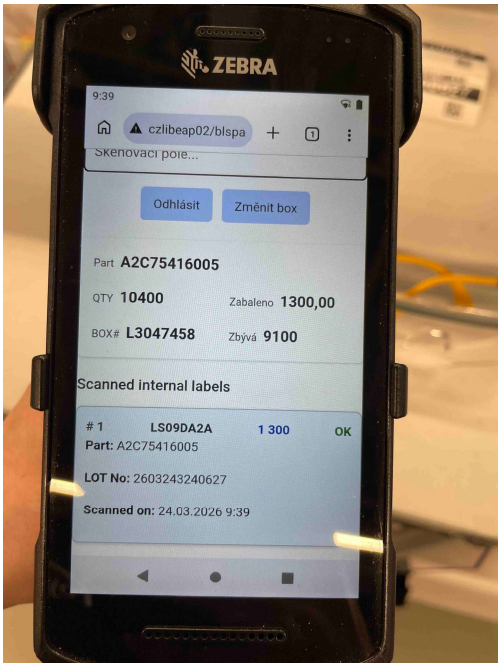
Pro rychlou orientaci operátora jsou karty také barevně odlišeny dle výsledků validace:

- **Žlutá** (`error-dup`) — duplicitní záznam.
- **Červená** (`error-part`) — nesprávné číslo dílu.
- **Bílá** (`basic`) — úspěšně naskenovaný záznam před naplněním krabice.
- **Zelená** (`error-full`) — validní záznam zobrazený ve stavu, kdy je aktuální krabice již naplněna.

Barevné třídy jsou jednotlivým kartám přiřazovány v události `lvInner_ItemDataBound`, která se vykonává při bindování dat pro každý zá-

znam v seznamu. Díky tomu lze u každé karty individuálně vyhodnotit její stav a nastavit odpovídající vzhled.

Karty jsou zároveň rozbalitelné. Po kliknutí na kartu se zobrazí detailnější informace o konkrétním skenu, například číslo dílu, číslo šarže a čas provedení skenu. Operátor tak má v základním pohledu pouze nejdůležitější informace, ale v případě potřeby může rychle zobrazit i detail daného záznamu. Ukázka rozkliknuté karty je zachycena na Obrázek 4.2.



Obrázek 4.2: Spuštěná aplikace s rozkliknutým detailem naskenovaného záznamu

Zjednodušená ukázka přiřazení vizuálních tříd a stavových popisků je uvedena v Kód 2.

```
    Select Case True
        Case isPartErr
            card.Attributes("class") += " error-part"
            status.InnerText = "Part"
        Case isDup
            card.Attributes("class") += " error-dup"
            status.InnerText = "Dup"
        Case isFull
            card.Attributes("class") += " error-full"
            status.InnerText = "Full"
        Case Else
            card.Attributes("class") += " ok"
            status.InnerText = "OK"
    End Select
```

Kód 2: Přiřazení vizuálního stavu kartě naskenovaného záznamu

4.9 Klientská logika — potlačení softwarové klávesnice

Jednou z hlavních výzev při implementaci uživatelského rozhraní bylo zabránit zobrazování softwarové klávesnice na čtečkách s operačním systémem Android. Skenovací textové pole musí být stále aktivní, aby do něj mohla čtečka zapisovat načtené 2D kódy. Zároveň ale nesmí při jeho aktivaci docházet k otevření softwarové klávesnice.

Pokud by se klávesnice po každém zaměření pole zobrazovala, zakryla by podstatnou část obrazovky a operátor by ji musel opakovaně ručně zavírat. To by celý proces skenování zpomalovalo a zhoršovalo použitelnost aplikace při provozu ve výrobě.

Řešení je implementováno v souboru `app.js` a je založeno na kombinaci dvou mechanismů:

- Nastavení atributu `inputmode="none"` na prvku `TextBox`. Tento atribut slouží jako informace pro mobilní prohlížeč, že dané vstupní pole nevyžaduje zobrazení softwarové klávesnice.
- Použití funkce `disableSoftKeyboard`, která před zavoláním metody `.focus()` dočasně nastaví vstupnímu poli atribut `readonly` a ihned po zaměření pole jej znovu odebere.

Druhý mechanismus funguje tak, že prohlížeč v okamžiku zaměření vyhodnotí pole jako pouze pro čtení, a proto softwarovou klávesnici nezobrazí. Atribut `readonly` je ale následně okamžitě odstraněn, takže při samotném skenu už je pole znovu zapisovatelné a hardwarová čtečka do něj může vložit načtená data.

Zjednodušená podoba funkce `disableSoftKeyboard` je uvedena v Kód 3.

```
function disableSoftKeyboard(el) {
    if (!el) return;
    const scrollPos = { x: window.scrollX, y: window.scrollY };
    el.setAttribute('readonly', 'readonly');
    try { el.focus({ preventScroll: true }); } catch { el.focus(); }
    el.removeAttribute('readonly');
    window.scrollTo(scrollPos.x, scrollPos.y);
}
```

Kód 3: Funkce `disableSoftKeyboard` pro zaměření skenovacího pole bez zobrazení softwarové klávesnice

Funkce si zároveň ukládá aktuální pozici stránky před zaměřením pole a po dokončení fokusu ji obnovuje. Tím se zabrání nechtěnému posunu stránky, ke kterému může na mobilních zařízeních při práci se vstupními poli docházet.

Funkce `disableSoftKeyboard` je volána ve všech situacích, kdy by mohlo dojít ke ztrátě fokusu skenovacího pole. Jedná se zejména o zavření modálního dialogu, kliknutí na kartu záznamu, dokončení asynchronního *postbacku* přes

`UpdatePanel` či návrat prohlížeče z neaktivního stavu po přepnutí okna nebo uspání a znovuprobuzení zařízení.

Tímto řešením je zajištěno, že skenovací pole zůstává pro operátora prakticky neustále připravené k dalšímu skenu, ale bez toho, aby se na obrazovce zobrazovala rušivá softwarová klávesnice.

4.10 Modální dialogy

Pro zpětnou vazbu operátorovi jsou v aplikaci použity dva typy modálních dialogů:

- **Chybový dialog** (`errorModal`) — červený dialog s ikonou kříže. Používá se při nesprávném dílu, duplicitním skenu, neplatném formátu kódu nebo při interní chybě aplikace.
- **Úspěšný dialog** (`successModal`) — zelený dialog s ikonou fajfky. Používá se hlavně pro signalizaci naplnění boxu a pro informační hlášky.

Dialogy jsou implementovány pomocí *frameworku* Bootstrap. Vizuální část dialogů je doplněna vlastními SVG animacemi, které vykreslují kruh, kříž nebo fajfku. Animace jsou řešeny pomocí CSS vlastnosti `stroke-dashoffset`, díky které se ikona při zobrazení dialogu postupně vykreslí.

Ze serverové části aplikace jsou dialogy vyvolávány z VB.NET kódu pomocí metody `ScriptManager.RegisterStartupScript`. Server po zpracování skenu vloží do odpovědi JavaScriptový příkaz, který na straně klienta zavolá odpovídající funkci. Pro chybové hlášky je používána funkce `showError(msg, title)` a pro úspěšné nebo informační hlášky funkce `showSuccess(msg, title)`.

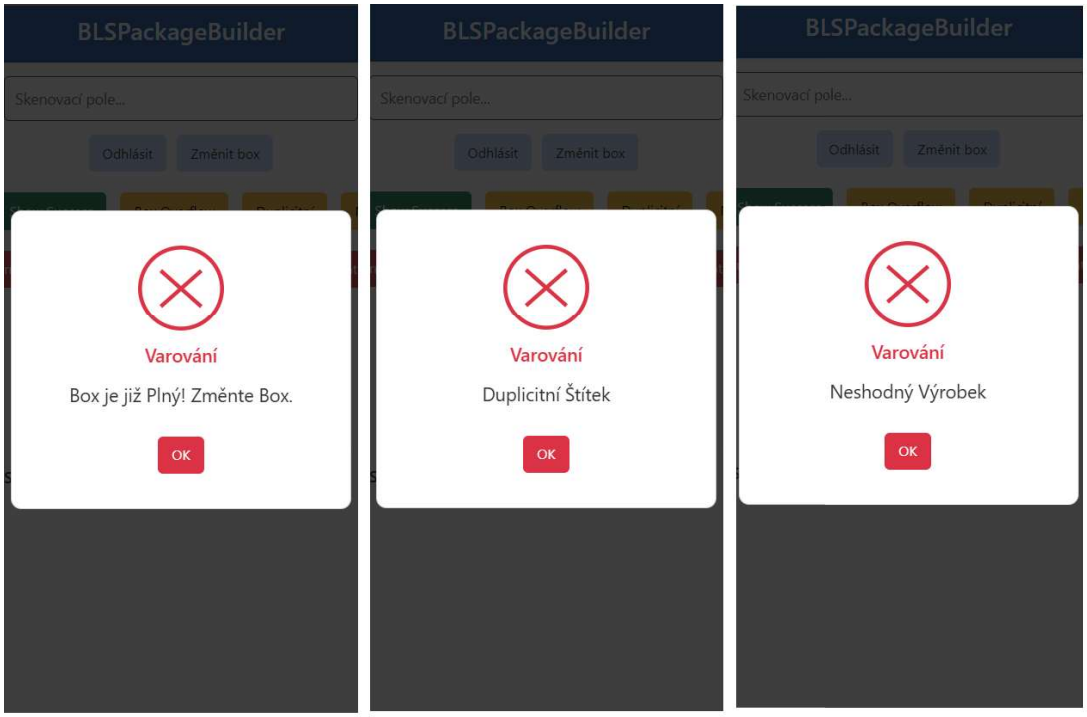
Tento způsob je vhodný hlavně proto, že aplikace běží uvnitř komponenty `UpdatePanel`. Po asynchronním *postbacku* je tak možné zobrazit výsledek zpracování skenu bez úplného znovunačtení celé stránky.

Aby modální dialog nezmizel při případném neočekávaném *postbacku* nebo překreslení části stránky, ukládá se informace o aktuálně zobrazeném dialogu do `localStorage` prohlížeče. Ukládá se typ dialogu, jeho titulek a zobrazená zpráva. Při dalším načtení nebo překreslení stránky JavaScript zkontroluje, zda je v `localStorage` uložený poslední dialog. Pokud ano, dialog znovu zobrazí.

Během zobrazení modálního dialogu je zároveň zablokována běžná práce s aplikací. Operátor nemůže pokračovat ve skenování ani používat ostatní prvky stránky. Aktivní zůstává pouze samotný dialog a jeho potvrzovací tlačítko. Toto omezení je záměrné, protože chybové a informační hlášky přímo souvisí s průběhem balení a jejich přehlédnutí by mohlo vést k pokračování v nesprávném procesu.

Operátor tak musí každou zobrazenou hlášku nejprve vědomě potvrdit. Až po potvrzení dialogu je aplikace znovu připravena ke skenování dalšího kódu. Tím je zajištěno, že uživatel byl s výsledkem poslední operace seznámen a nemůže

jej neúmyslně přeskočit rychlým dalším skenem. Ukázky chybových hlášení jsou uvedeny na Obrázek 4.3.

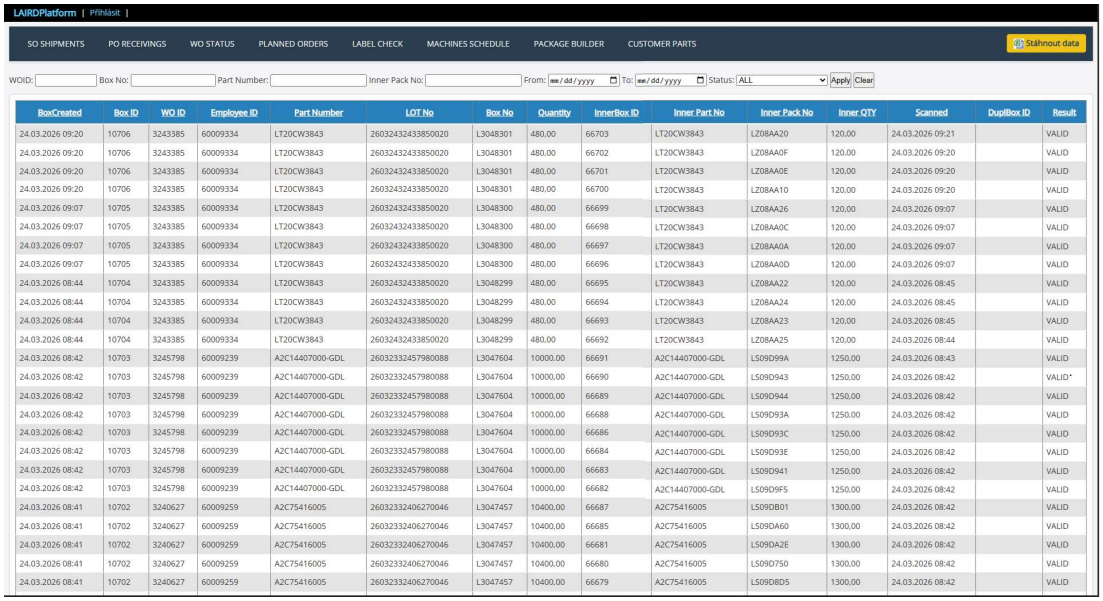


Obrázek 4.3: Ukázky chybových modálních dialogů v aplikaci

4.11 Implementace pohledu Package Builder

Pohled **Package Builder** je realizován jako další položka v komponentě `MultiView` modulu **Dashboards**. Je tedy součástí stejného rozhraní jako ostatní existující *dashboardy* využívané na platformě, například **SO Shipments**, **PO Receivings**, **WO Status**, **Planned Orders**, **Label Check**, **Machines Schedule** a **Customer Parts**.

Aktivace pohledu probíhá kliknutím na tlačítko **PACKAGE BUILDER** v hlavním menu *dashboardů*. Po jeho zvolení se zobrazí tabulkový přehled záznamů z aplikace BLSPackageBuilder společně s filtrační lištou a možností exportu dat do souboru `.xlsx` (Microsoft Excel).



Obrázek 4.4: Pohled Package Builder v modulu Dashboards

4.11.1 Filtrační lišta

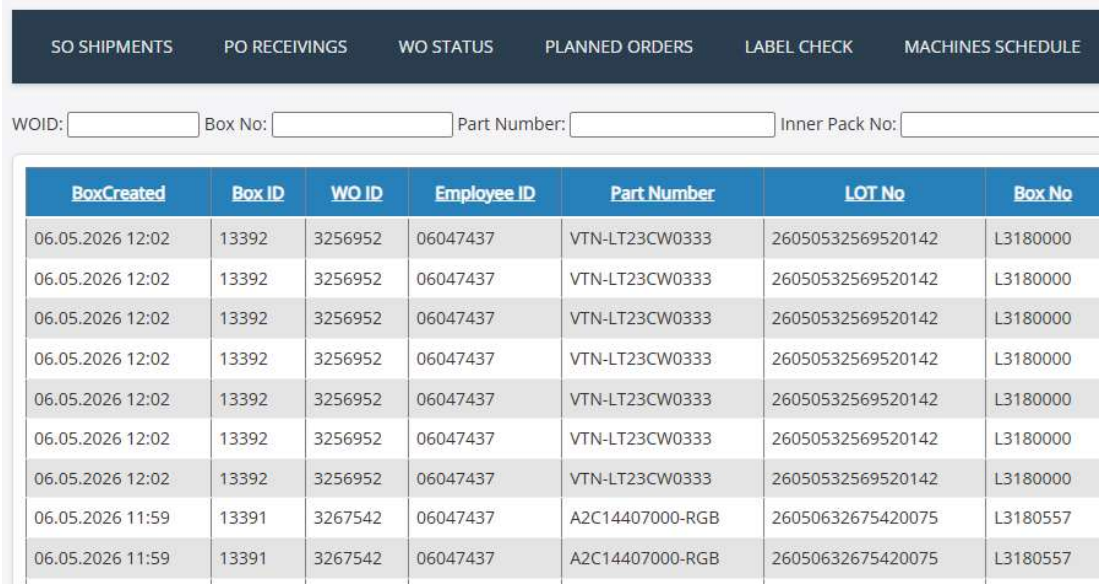
V horní části pohledu je umístěna filtrační lišta, která umožňuje pracovníkům kvality rychle dohledávat konkrétní balení podle pracovního příkazu nebo podle chybového stavu. Filtr je navržen tak, aby bylo možné vyhledávat jak podle údajů vnějšího obalu, tak podle údajů jednotlivých vnitřních balení.

Filtrační lišta obsahuje tato pole:

- **WOID** — číslo pracovního příkazu. Jedná se o numerický filtr s přesnou shodou.
- **Box No** — číslo boxu. Textový filtr používá vyhledávání pomocí `LIKE`.
- **Part Number** — číslo dílu.
- **Inner Pack No** — číslo vnitřního balení.
- **From / To** — časový rozsah, kdy mohl být sken proveden.
- **Status** — výběr výsledného stavu záznamu. `ALL`, `WRONG PART`, `DUPLICATE` a `VALID`.

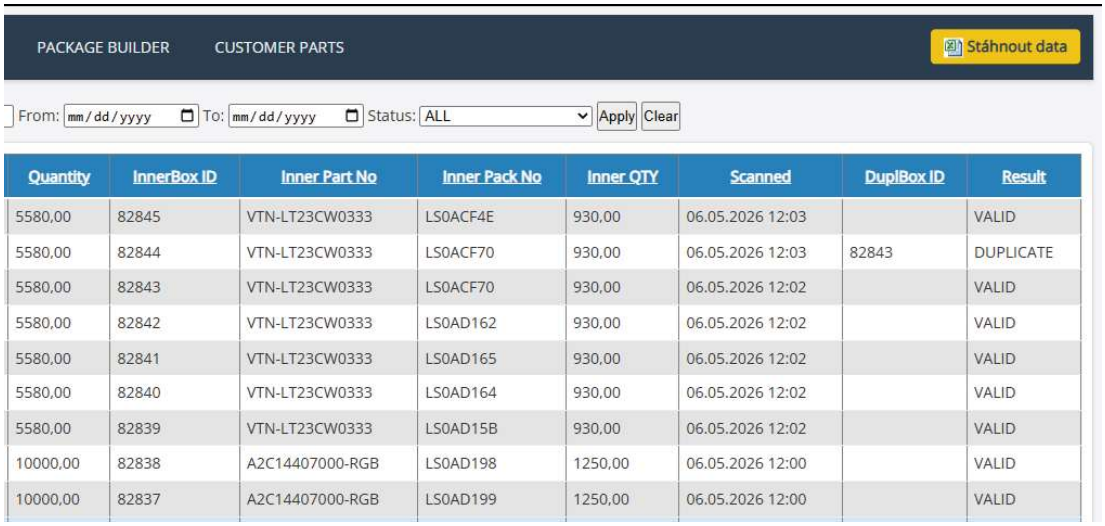
Vedle filtračních polí jsou umístěna tlačítka **Apply** a **Clear**. Tlačítko **Apply** aplikuje zadané filtry na tabulku a tlačítko **Clear** filtry vyčistí. Pro pohodlnější používání je pomocí JavaScriptu zachycena také klávesa Enter ve filtračních polích. Pokud ji uživatel stiskne, automaticky se vyvolá stejná akce jako při kliknutí na tlačítko **Apply**.

Na Obrázek 4.4 je vidět celkové rozložení pohledu včetně filtrační lišty, tabulky záznamů a tlačítka pro export dat. Detail levé části filtrační lišty je zobrazen na Obrázek 4.5 a pravá část s časovým rozsahem a tlačítky na Obrázek 4.6.



BoxCreated	Box ID	WO ID	Employee ID	Part Number	LOT No	Box No
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 12:02	13392	3256952	06047437	VTN-LT23CW0333	26050532569520142	L3180000
06.05.2026 11:59	13391	3267542	06047437	A2C14407000-RGB	26050632675420075	L3180557
06.05.2026 11:59	13391	3267542	06047437	A2C14407000-RGB	26050632675420075	L3180557

Obrázek 4.5: Detail levé části filtrační lišty (WOID, Box No, Part Number, Inner Pack No)



Quantity	InnerBox ID	Inner Part No	Inner Pack No	Inner QTY	Scanned	DuplBox ID	Result
5580,00	82845	VTN-LT23CW0333	LS0ACF4E	930,00	06.05.2026 12:03		VALID
5580,00	82844	VTN-LT23CW0333	LS0ACF70	930,00	06.05.2026 12:03	82843	DUPLICATE
5580,00	82843	VTN-LT23CW0333	LS0ACF70	930,00	06.05.2026 12:02		VALID
5580,00	82842	VTN-LT23CW0333	LS0AD162	930,00	06.05.2026 12:02		VALID
5580,00	82841	VTN-LT23CW0333	LS0AD165	930,00	06.05.2026 12:02		VALID
5580,00	82840	VTN-LT23CW0333	LS0AD164	930,00	06.05.2026 12:02		VALID
5580,00	82839	VTN-LT23CW0333	LS0AD15B	930,00	06.05.2026 12:02		VALID
10000,00	82838	A2C14407000-RGB	LS0AD198	1250,00	06.05.2026 12:00		VALID
10000,00	82837	A2C14407000-RGB	LS0AD199	1250,00	06.05.2026 12:00		VALID

Obrázek 4.6: Detail pravé části filtrační lišty (From / To, Status, tlačítka Apply a Clear)

4.11.2 SQL dotaz a propojení tabulek

Datový zdroj pro *dashboard* `DSPackageBuilder` načítá data z tabulek `BPB_OuterPackages` a `BPB_InnerPackages`. Tyto tabulky jsou propojeny pomocí `INNER JOIN`, a to přes klíč `BID`, který představuje vazbu mezi vnějším obalem a jeho naskenovanými vnitřními baleními.

Z výsledku dotazu jsou vybírány údaje potřebné pro kontrolu a zpětnou analýzu balení. Jedná se například o čas vytvoření boxu, identifikátory balení, číslo pracovního příkazu, čísla dílů, množství, čísla šarží a čas jednotlivých skenů.

Součástí dotazu je také dopočítaný virtuální sloupec `Result`, který určuje stav každého záznamu. Tento sloupec je vytvořen pomocí příkazu `CASE`, jehož zjednodušená podoba je uvedena v Kód 4.

```
CASE
    WHEN RTRIM(i.BoxNo) LIKE '%-X' THEN 'WRONG PART'
    WHEN i.DuplicateValidBID IS NOT NULL THEN 'DUPLICATE'
    ELSE 'VALID'
END AS Result
```

Kód 4: Vyhodnocení výsledného stavu záznamu v SQL dotazu

Výsledná hodnota sloupce Result rozděluje záznamy do tří hlavních skupin. Záznam s příznakem `-X` na konci čísla boxu je vyhodnocen jako nesprávný díl. Záznam s vyplněnou hodnotou `DuplicateValidBID` je vyhodnocen jako duplicita. Všechny ostatní záznamy jsou označeny jako validní.

Filtr stavu ve `WHERE` toto rozdělení respektuje. Pracovník kvality tak může zobrazit pouze konkrétní typ záznamů, který požaduje. Například pouze duplicitní skeny, jen záznamy s nesprávným dílem nebo pouze validní záznamy.

4.11.3 Tabulkové zobrazení

Filtrovaná data jsou zobrazena pomocí komponenty `GridView`. Tabulka má zapnuté řazení pomocí `AllowSorting=true` a stránkování pomocí `AllowPaging=true`. Velikost jedné stránky je nastavena na 100 záznamů, aby bylo možné pohodlně pracovat i s větším množstvím dat bez zbytečně dlouhého načítání stránky.

Tabulka obsahuje sloupce odpovídající klíčovým údajům potřebným pro analýzu kvality. Zobrazuje například čas vytvoření boxu, identifikátor BID, číslo pracovního příkazu WOID, identifikátor zaměstnance, číslo dílu, číslo šarže, číslo boxu, množství, čas skenu a sloupec Result, který zobrazuje výsledek validace.

Díky možnosti řazení může pracovník kvality rychle měnit pohled na data podle aktuální potřeby. Může například seřadit záznamy podle času skenu, čísla pracovního příkazu nebo výsledku validace.

4.11.4 Export do Microsoft Excel

V pravém horním rohu pohledu je umístěno tlačítko **Stáhnout data**, které slouží k exportu aktuálně zobrazených dat do souboru Microsoft Excel. Po kliknutí na toto tlačítko je na serverové straně spuštěna metoda `lbExportPB_Click`.

Tato metoda nejprve načte aktuálně filtrovaná data z datového zdroje `DSPackageBuilder` do objektu `DataView`. Následně z nich vytvoří objekt `DataTable`, který představuje tabulkovou podobu exportovaných dat. Pokud je v komponentě `GridView` aktivní řazení, metoda je také aplikuje na exportovaná data, aby stažený soubor odpovídal pořadí zobrazenému uživateli v *dashboardu*. Zjednodušená ukázka exportu je uvedena na Kód 5.

```
Dim ex As New ExcelDocumentCompiler(dt)
ex.WriteDocument()
Session("XlsObject") = ex
Session("XlsObjectName") = "PackageBuilder_" _
    & Now.ToString("yyyyMMddHHmmss")
ScriptManager.RegisterStartupScript(Me, Me.GetType(), "OpenWindow",
    "window.location.href='dochelper/xlshandler.ashx';", True)
```

Kód 5: Příprava souboru XLSX a předání exportu prohlížeči

Po přípravě dat je vytvořena instance pomocné třídy `ExcelDocumentCompiler`, která z předané tabulky vygeneruje sešit ve formátu XLSX. Vygenerovaný sešit je uložen do `XlsObject`.

Nakonec je pomocí `ScriptManager.RegisterStartupScript` vyvolán přechod na obsluhovač `xlshandler.ashx`, který připravený soubor odešle prohlížeči ke stažení.

Název vygenerovaného souboru obsahuje časové razítko ve formátu `PackageBuilder_yyyyMMddHHmmss.xlsx`. Díky tomu lze jednotlivé exporty snadno odlišit i v případě, že pracovník kvality provede více exportů za sebou.

4.12 Nasazení aplikace

Vývoj aplikace probíhal v prostředí Visual Studio 2022. Modul BLSPackageBuilder nebyl vytvářen jako samostatná webová aplikace, ale jako nová část již existující platformy. Celá platforma je řešena jako jeden větší ASP.NET Web Forms projekt, ve kterém jsou jednotlivé funkční moduly umístěny v samostatných složkách. Nový modul byl proto přidán do složky `BLSPackageBuilder`, podobně jako ostatní existující části platformy.

Po dokončení implementace byl celý projekt zkompilován a nasazen na firemní server IIS, na kterém již běží zbytek platformy. Nasazení tedy nevyžadovalo vytvoření nového serverového prostředí, nového webu ani samostatného aplikačního poolu.

Postup nasazení lze shrnout do následujících kroků:

- **Kompilace projektu** — protože je platforma řešena jako jeden společný projekt, je při změně modulu nutné znovu sestavit celý projekt. Výsledkem sestavení jsou zkompilované knihovny v adresáři `bin`, které obsahují serverovou logiku celé platformy včetně nového modulu. Soubory kódu na pozadí (`.aspx.vb`) a návrhové soubory (`.designer.vb`) se na server samostatně nekopírují, protože jejich obsah je zahrnut ve zkompilované podobě v adresáři `bin`.

- **Příprava souborů —** před nasazením byly připraveny soubory potřebné pro běh nové aplikace. Přenášen byl celý adresář `bin` platformy a dále soubory nového modulu ve složce `BLSPackageBuilder`. Jednalo se zejména o stránky `.aspx`, klientské skripty `JS`, styly `CSS`, obrázky a další statické soubory potřebné pro běh uživatelského rozhraní.
- **Kontrola verze *frameworku* —** cílový server používá odpovídající verzi .NET Frameworku. Aplikace byla kompilována pro .NET Framework 4.8, proto muselo být stejné prostředí dostupné i na serveru.
- **Konfigurace IIS —** v případě tohoto modulu nebyla nutná žádná dodatečná konfigurace serveru. Modul byl nasazen jako nová složka v rámci existujícího webu, takže nebylo potřeba vytvářet nový web, nový aplikační pool ani samostatnou doménu.
- **Vlastní nasazení —** připravené soubory byly přeneseny do příslušného adresáře existující platformy na serveru. Po přístupu na URL adresu nové složky IIS modul inicializoval jako součást běžící platformy a stránka byla dostupná pro použití.

Tento způsob nasazení odpovídá běžnému provozu celé platformy. BLSPackageBuilder tak využívá stejné serverové prostředí, stejný adresářový základ, stejnou konfiguraci a stejný způsob distribuce jako ostatní již existující moduly.

5 TESTOVÁNÍ

Testování aplikace probíhalo ve třech hlavních fázích. Nejprve byla aplikace ověřována v lokálním vývojovém prostředí, následně byla nasazena na samostatný testovací server a poslední fáze probíhala přímo v produkčním prostředí výroby závodu. Tento postup umožnil postupně ověřit funkčnost aplikační logiky, komunikaci s databází, chování aplikace při práci s fyzickými čtečkami a připravenost aplikace pro reálný provoz.

5.1 Testování v lokálním prostředí

První fáze testování probíhala v lokálním prostředí. Aplikace byla spuštěna na `localhostu` a sloužila především k průběžnému ověřování implementované funkcionality během prvotního vývoje a následného rozvoje.

V této fázi bylo ověřováno hlavně správné přepínání jednotlivých pohledů aplikace, převzetí správné *cookie* pro identifikaci čteček, identifikace uživatele na základě jeho osobního čísla, načítání vnějšího obalu, skenování vnitřních balení, zápis dat do databáze a aktualizace různých polí na obrazovce.

Pro ověření chování s reálnou čtečkou byla k počítači připojena fyzická čtečka Zebra TC22 přes USB. Čtečka předávala naskenovaný 2D kód jako vstup z klávesnice do aktivního textového pole. Tím bylo možné testovat aplikaci stejným způsobem, jakým by ji později používali operátoři ve výrobě.

Během lokálního testování bylo ověřeno, že aplikace správně zpracovává 2D kódy vnějších i vnitřních balení, rozpoznává typ skenu a podle aktuálního stavu aplikace provádí odpovídající operace.

5.2 Testování na testovacím serveru

Po dokončení základního ověření v lokálním prostředí byla aplikace nasazena na samostatný testovací server, který byl pro účely této práce nastaven. Toto prostředí sloužilo k ověření chování aplikace mimo vývojářský počítač a k otestování provozu s více čtecími zařízeními najednou.

Na testovacím serveru bylo ověřováno především:

- správné načtení aplikace,
- funkčnost modulu BLSPackageBuilder jako součásti existující platformy,
- komunikace aplikace s databází,
- současná práce více čteček,
- správné zpracování skenů z různých zařízení,
- stabilita aplikace při opakovaném skenování,

- správné zobrazování modálních dialogů a stavových informací.

V rámci testování bylo k aplikaci připojeno několik čteček současně. Cílem bylo ověřit, zda aplikace zvládne běžný provozní scénář, kdy několik operátorů pracuje s balením zároveň. Každá čtečka pracovala ve vlastní relaci prohlížeče a skeny byly zapisovány do databáze jako samostatné záznamy.

Testování potvrdilo, že aplikace dokáže zpracovávat skeny z více zařízení najednou bez vzájemného ovlivňování jednotlivých pracovišť. Kódy byly rozpoznávány správně, validace probíhala podle očekávání a záznamy byly ukládány do odpovídajících tabulek.

5.3 Testované scénáře balicí aplikace

V průběhu lokálního testování i testování na testovacím serveru byly ověřeny hlavní provozní scénáře aplikace:

- **Identifikace zařízení** — po stisknutí tlačítka **Start** došlo k vytvoření *cookie* `bpbdevid.lairdplatform` a aplikace přešla do pohledu identifikace uživatele. Po obnovení stránky bylo zařízení automaticky rozpoznáno.
- **Identifikace uživatele** — při zadání platného identifikátoru zaměstnance aplikace přešla do hlavního pohledu balení. Při neplatném identifikátoru zobrazila chybový dialog a zůstala v pohledu identifikace uživatele.
- **Načtení vnějšího obalu** — po naskenování štítku vnějšího obalu aplikace rozparsovala číslo boxu, číslo dílu atd. Validní záznam byl uložen do tabulky `BPB_OuterPackages`. Při skenu neplatného záznamu aplikace správně vyhodnotila chybu formátu.
- **Opakované načtení stejného boxu** — pokud byl naskenován box, který již v databázi existoval, aplikace dohledala existující záznam a zobrazila jeho historii místo vytvoření duplicitního vnějšího balení.
- **Skenování správných vnitřních balení** — aplikace ověřila shodu čísla dílu s aktuálním vnějším obalem, zapsala validní záznam do tabulky `BPB_InnerPackages`, do kontrolní tabulky v samotné aplikaci a aktualizovala počet zabalených a zbývajících kusů.
- **Skenování nesprávného dílu** — při naskenování vnitřního balení s jiným číslem dílu aplikace zobrazila chybový dialog a záznam označila příznakem `-X`.
- **Detekce duplicity** — při opakovaném skenu stejného vnitřního štítku aplikace rozpoznala duplicitu, označila nový záznam hodnotou `DuplicateValidBID` a nezapočítala jej do plnění boxu.
- **Naplnění boxu** — po dosažení plánovaného množství aplikace označila box jako plný, změnila barvu pozadí tabulky a zobrazila informační dialog.
- **Změna boxu a odhlášení** — bylo ověřeno, že tlačítko **Změnit box** vrací aplikaci do stavu čekání na nový vnější obal a tlačítko **Odhlásit** ukončuje relaci uživatele.

5.4 Testování *dashboardu* Package Builder

Součástí testování bylo také ověření pohledu **Package Builder** v modulu **Dashboards**. Tento pohled slouží pracovníkům kvality k dohledávání historie balení při reklamaci a ke kontrole chybných nebo duplicitních skenů.

Testování probíhalo nad daty vytvořenými během testování balicí aplikace. Po provedení skenů byly záznamy dohledávány v *dashboardu* a byla ověřována správnost jejich zobrazení.

Otestovány byly hlavně tyto funkce:

- **Filtrování bez parametrů** — zobrazení všech dostupných záznamů.
- **Filtrování podle WOID** — ověření, že se zobrazí pouze záznamy příslušného pracovního příkazu.
- **Filtrování podle čísla boxu** — ověření vyhledávání podle čísla vnějšího i vnitřního boxu.
- **Filtrování podle čísla dílu** — dohledání záznamů podle konkrétního čísla dílu.
- **Filtrování podle časového rozsahu** — ověření, že *dashboard* zobrazí pouze záznamy z vybraného období.
- **Filtrování podle stavu** — ověření zobrazení stavů `ALL`, `WRONG PART`, `DUPLICATE` a `VALID`.
- **Řazení sloupců** — kontrola vzestupného i sestupného řazení podle vybraných sloupců.
- **Stránkování** — ověření správného přechodu mezi stránkami při větším množství záznamů.
- **Export do Excelu** — ověření, že exportovaný soubor obsahuje odpovídající filtrovaná a seřazená data a že se jedná o platný formát.

5.5 Ověření v produkčním prostředí

Po úspěšném otestování na testovacím serveru byla aplikace nasazena do produkčního prostředí. Produkční ověření probíhalo nejprve omezeně, aby bylo možné sledovat chování aplikace v reálném provozu bez okamžitého rozšíření na všechna pracoviště. V první fázi byla aplikace nasazena na dvě vybraná pracoviště v lokaci BLS (Lisovna), kde ji používali proškolení operátoři.

V produkčním prostředí bylo ověřováno především:

- správné chování aplikace na firemním IIS serveru,
- správné fungování čteček ve výrobním prostředí,
- zpracování reálných 2D kódů ze štítků,
- zápis dat do databáze,
- správné vyhodnocení nesprávných dílů a duplicit,
- použitelnost rozhraní pro operátory,
- dostupnost záznamů v *dashboardu* pro pracovníky kvality.

Během omezeného provozu na těchto pracovištích byly průběžně sledovány databázové záznamy i logovací soubory. Zároveň byla sbírána zpětná vazba od

operátorů a pracovníků kvality, aby bylo možné ověřit nejen technickou funkčnost aplikace, ale také její použitelnost v běžném výrobním procesu.

Na základě prvního provozu byly provedeny drobné úpravy textů chybových hlášení a zpřesnění některých informačních zpráv. Po těchto úpravách byla aplikace postupně zpřístupněna i na dalších pracovištích.

5.6 Výsledky testování

Testování potvrdilo, že aplikace splňuje požadavky zadání. Aplikace správně zpracovává 2D kódy vnějších i vnitřních balení, ověřuje shodu čísla dílu, detekuje duplicitní skeny a sleduje postup plnění boxu.

Na testovacím serveru bylo ověřeno, že aplikace zvládá současnou práci více čteček. Jednotlivé relace se navzájem neovlivňovaly a skeny byly ukládány správně podle aktuálního balení a uživatele.

V produkčním prostředí bylo potvrzeno, že aplikace je použitelná pro běžný provoz na balicích linkách. Čtečky Zebra předávaly kódy správně, modální dialogy poskytovaly operátorům srozumitelnou zpětnou vazbu a mechanismus potlačení softwarové klávesnice nebránil skenování.

Pohled **Package Builder** v *dashboardu* poskytl pracovníkům kvality potřebný přehled nad historií balení. Filtrování, řazení, stránkování a export do Excelu fungovaly podle očekávání a umožnily dohledávat validní, duplicitní i chybné záznamy.

Výsledkem testování bylo ověření, že aplikace je připravena pro provoz ve výrobním prostředí. Zároveň byly identifikovány možné oblasti pro budoucí rozšíření, například doplnění detailnějších statistik chybovosti, zvýraznění nejčastějších typů chyb nebo automatické reporty pro pracovníky kvality.

6 VYHODNOCENÍ PROVOZNÍCH DAT

Po nasazení aplikace do produkčního prostředí byla ukládaná data využita také pro základní vyhodnocení provozu aplikace. Cílem této části bylo ověřit, zda aplikace kromě okamžité kontroly balení poskytuje také užitečná data pro zpětnou analýzu chybovosti a pro práci oddělení kvality.

Vyhodnocení bylo provedeno nad záznamy vytvořenými během běžného provozu aplikace.

Pozornost byla zaměřena na hlavní typy problematických skenů:

- **Duplicitní skeny** — záznamy, u kterých je vyplněna hodnota `DuplicateValidBID`.
- **Skeny s nesprávným dílem** — záznamy, u kterých je číslo boxu s příznakem `-X`.

Tyto dva stavy jsou z hlediska kvality nejdůležitější. Duplicitní sken znamená, že se uživatel pokusil znovu načíst štítek, který již byl v systému evidován jako validní. Sken s příznakem `-X` naopak znamená, že číslo dílu na vnitřním štítku neodpovídalo číslu dílu aktuálně načteného vnějšího obalu.

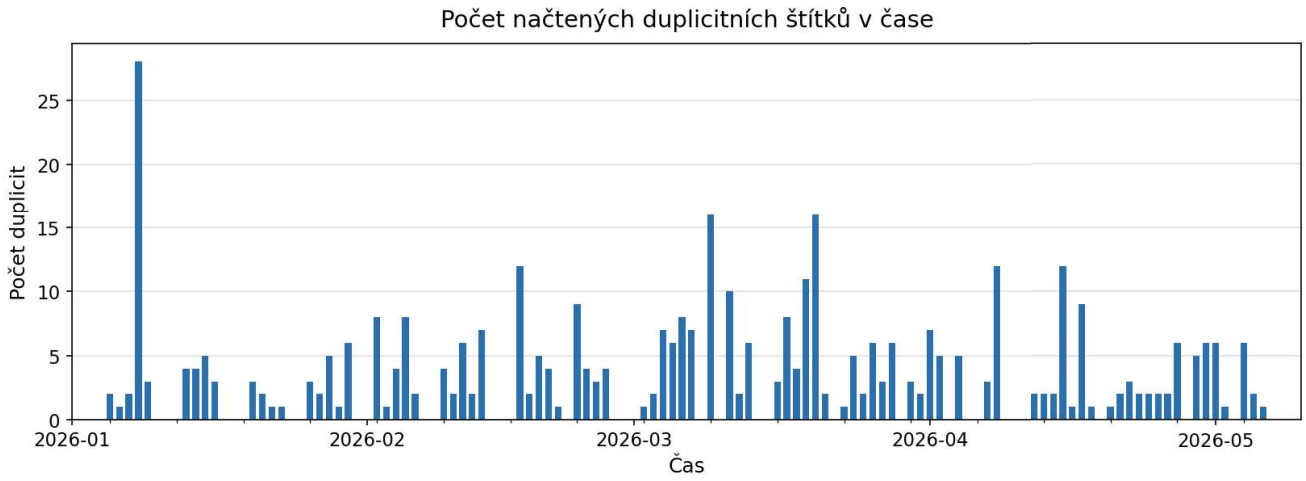
Z grafů byly dopočítány základní souhrnné hodnoty uvedené v Tabulka 6.1. Tabulka kromě podílu mezi problematickými skeny uvádí také podíl vůči všem uloženým skenům ve sledovaném exportu. Celkový počet všech skenů v tomto exportu bylo zhruba 83 tisíc záznamů.

Tabulka 6.1: Souhrnné statistiky problematických skenů

Typ záznamu	Součet	Průměr na aktivní den	Medián na aktivní den	Podíl z problematických skenů	Podíl ze všech skenů
Duplicitní skeny	409	4,6	3	94,9 %	0,49 %
Skeny s nesprávným dílem	22	2,0	2	5,1 %	0,03 %
Celkem problematické skeny	431	–	–	100 %	0,52 %

6.1 Vývoj duplicitních skenů v čase

První graf zobrazuje počet načtených duplicitních štítků v čase. Každý sloupec v Obrázek 6.1 představuje počet duplicitních skenů zaznamenaných v daném dni.



Obrázek 6.1: Počet načtených duplicitních štítků v čase

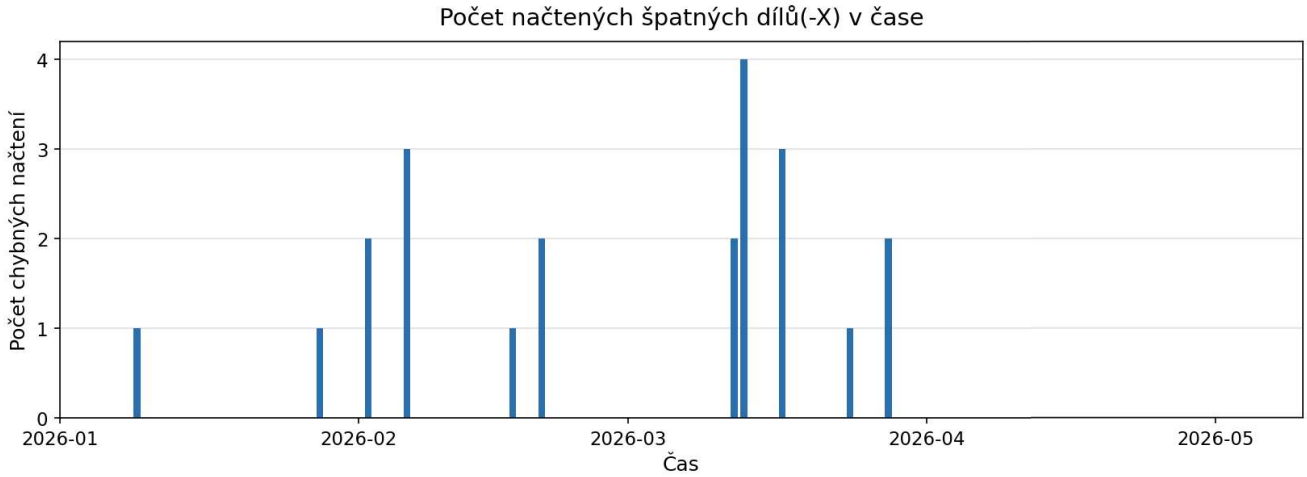
Z grafu je patrné, že duplicitní skeny se v provozu vyskytují pravidelně, ale jejich počet není rozložen rovnoměrně. Ve většině aktivních dnů se jedná o jednotky případů, zatímco v některých dnech dochází k výrazným špičkám. Souhrnně bylo zachyceno 409 duplicitních skenů, průměr na den s alespoň jednou duplicitou byl 4,6 záznamu a medián činil 3 záznamy.

Průběh může naznačovat, že duplicity nevznikají pouze náhodně, ale často souvisí s konkrétními provozními situacemi. Může se jednat například o opakované načtení stejného štítku při nejistotě operátora, nebo o chybovost při práci na více strojích.

Důležité je, že aplikace tyto skeny správně zachytila a označila jako duplicitní. Bez elektronické evidence by podobné případy nebylo možné jednoduše zpětně dohledat a část z nich by mohla v krajním případě vyústit v reklamace.

6.2 Vývoj skenů s nesprávným dílem v čase

Druhý graf zobrazuje počet skenů, při kterých bylo detekováno nesprávné číslo dílu. Tyto záznamy jsou v databázi označeny příznakem `-X` a jejich vývoj je zobrazen v Obrázek 6.2.



Obrázek 6.2: Počet načtených štítků s nesprávným číslem dílu v čase

Ve srovnání s duplicitními skeny je počet nesprávných dílů výrazně nižší. Většina sledovaného období neobsahuje žádný takový záznam nebo pouze jednotlivé případy. Celkem bylo zachyceno 22 skenů s nesprávným dílem. Průměr na den, ve kterém se tento problém objevil, byl 2,0 záznamu a medián činil 2 záznamy.

Tento výsledek je z hlediska provozu pozitivní, protože sken nesprávného dílu představuje závažnější chybu než duplicita. Zatímco duplicita často znamená pouze opakované načtení stejného štítku, nesprávný díl může znamenat reálný pokus vložit do krabice jiné zboží, než které odpovídá vnějšímu obalu.

Aplikace v těchto případech funguje jako okamžitá kontrola. Operátor je upozorněn chybovým dialogem a záznam je uložen do databáze jako chybný. Díky tomu lze později dohledat, kdy k pokusu došlo, u jakého boxu a o jaký díl se jednalo.

6.3 Porovnání duplicit a nesprávných dílů

Z porovnání obou grafů vyplývá, že nejčastějším problematickým stavem jsou duplicitní skeny. Ty tvoří 94,9 % všech problematických záznamů, zatímco skeny s nesprávným číslem dílu tvoří 5,1 %. Vůči všem uloženým skenům se však jedná pouze o 0,52 % záznamů, takže problematické skeny představují malou část celkového provozu aplikace. To odpovídá očekávanému provoznímu chování, protože opakované načtení stejného štítku může vzniknout poměrně snadno například nepozorností nebo opakováním skenu.

Naopak výskyt nesprávných dílů je výrazně nižší. To naznačuje, že samotný proces balení do krabic je z hlediska výběru správného dílu relativně stabilní, ale zároveň potvrzuje smysl zavedené kontroly. I jednotlivé případy nesprávného dílu mohou být z pohledu kvality důležité, protože bez kontroly by mohly vést k vytvoření špatného balení a následné reklamace.

7 ZÁVĚR

V této bakalářské práci byl navržen a implementován nový modul **BLSPackageBuilder** pro platformu Laird Digitalization Platform, který provádí elektronickou kontrolu balení výrobků prostřednictvím skenování 2D kódů na štítcích. Modul byl úspěšně nasazen do produkčního prostředí výrobního závodu a uveden do běžného provozu.

Aplikace splňuje všechny požadavky zadání. Provádí identifikaci uživatele a vnějšího obalu, důsledně validuje vnitřní balení z hlediska shody čísla dílu, detekce duplicit a sledování plnění boxu. Operátorovi poskytuje okamžitou vizuální i textovou zpětnou vazbu. Logování každé operace formou unikátně identifikovaných záznamů zároveň umožňuje zpětnou analýzu problémů v produkčním prostředí.

Současně byl rozšířen modul **Dashboards** o nový pohled **Package Builder**, který poskytuje pracovníkům oddělení kvality nástroj pro filtrování historie skenů, identifikaci chybných balení a export dat do tabulek aplikace Microsoft Excel. Pohled navazuje na vzhled a ovládání ostatních *dashboardů*, takže jej lze používat bez odlišného způsobu práce.

Součástí práce bylo také základní vyhodnocení provozních dat získaných po nasazení aplikace. Analýza ukázala, že nejčastějším problematickým stavem jsou duplicitní skeny, zatímco skeny s nesprávným číslem dílu se vyskytují výrazně méně často. Ukládaná data se tak ukázala jako použitelná nejen pro okamžitou kontrolu balení, ale také pro zpětné sledování chybovosti a další práci v oddělení kvality.

V průběhu vývoje byla jako největší technická výzva identifikována otázka potlačení softwarové klávesnice na zařízeních Zebra při zachování plné funkčnosti hardwarového skenu. Zvolené řešení se ukázalo jako spolehlivé a bylo prověřeno ve vývojovém, testovacím i produkčním prostředí.

7.1 Možnosti dalšího rozvoje

V rámci dalšího rozvoje aplikace lze do budoucna zvážit následující rozšíření:

- **Notifikační mechanismus pro kvalitu** — aplikace by mohla sledovat počet chybových skenů v krátkém časovém úseku a při překročení nastavené hranice automaticky upozornit pracovníky kvality. Upozornění by mohlo obsahovat čas, pracoviště, typ chyby a odkaz na filtrovaný přehled záznamů v *dashboardu*.

- **Generování protokolu o kontrole balení —** po naplnění každé krabice by aplikace mohla automaticky vygenerovat tiskový protokol shrnující průběh balení, který by se mohl přiložit fyzicky k zásilce nebo elektronicky archivovat.
- **Statistický přehled přímo v *dashboardu* —** provozní data ukázala, že informace o duplicitách a nesprávných dílech mají využití i pro dlouhodobé sledování chybovosti. Do budoucna by proto bylo vhodné doplnit do pohledu Package Builder automatické grafy a statistiky podle období, pracoviště, operátora nebo čísla dílu.
- **Migrace platformy na novější .NET —** dlouhodobě je vhodné zvážit přechod celé platformy z ASP.NET Web Forms na moderní ASP.NET Core, což by přineslo vyšší výkon a snazší údržbu kódu. Tento krok by však vyžadoval přepracování všech modulů platformy, nikoliv pouze tohoto.

8 POUŽITÁ LITERATURA

[1] MACDONALD, Matthew. *Pro ASP.NET 4.5 in VB*. 5. Apress, 2012. ISBN 978-1-4302-4329-8.

[2] MICROSOFT. *Visual Basic Documentation*. Online. 2026. Dostupné z: https://learn.microsoft.com/en-us/dotnet/visual-basic/. [citováno 2026-05-04].

[3] ZAK, Diane. *Programming with Microsoft Visual Basic 2019/2022*. 9. Cengage Learning, 2022. ISBN 978-0-357-67400-0.

[4] LIEW, Voon Kiong. *Visual Basic 2022 Made Easy: A Comprehensive Guide to Visual Basic 2022 Programming*. Independently published, 2023. ISBN 979-8-3789-2032-7.

[5] DUCKETT, Jon. *JavaScript and JQuery: Interactive Front-End Web Development*. Wiley, 2014. ISBN 978-1-118-53164-8.

[6] SPURLOCK, Jake. *Bootstrap*. O'Reilly Media, 2013. ISBN 978-1-4493-4391-0.

[7] PETKOVIC, Dusan. *Microsoft SQL Server 2019: A Beginner's Guide*. 7. McGraw-Hill Education, 2020. ISBN 978-1-260-45787-9.

[8] MICROSOFT. *ADO.NET*. Online. 2025. Dostupné z: https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/. [citováno 2026-05-04].

[9] STANEK, William R. *Windows Server 2012 R2 Inside Out: Services, Security, & Infrastructure*. Microsoft Press, 2014. ISBN 978-0-7356-8561-1.

[10] ZEBRA TECHNOLOGIES. *TC22/TC27 Touch Computer Product Reference Guide*. 2024. [citováno 2026-05-04]. Dostupné z: https://www.zebra.com/content/dam/zebra_new_ia/en-us/manuals/mobile-computers/tc22--27/tc22-tc27-prg-en.pdf