ДОДАТОК А

Публікація

Міністерство освіти і науки України

NURE

Харківський національний університет радіоелектроніки

ЗБІРНИК

студентських наукових статей

«Автоматизація та приладобудування»

«Automation and Development of Electronic Devices»

ADED-2024

(Випуск 2)

[електронне видання]

KITAP

http://nure.ua/department/kafedra-komp-yuterno-integrovanih-tehnologiy-avtomatizatsiyi-ta-mehatroniki-kitam

iTeD

http://itez.zntu.edu.ua/

http://kafea.kdu.edu.ua

Харків 2024

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки
кафедра комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(КІТАР)

ЗБІРНИК

студентських наукових статей

«Автоматизація та приладобудування»

«Automation and Development of Electronic Devices»

ADED-2024

(Випуск 2)

[електронне видання]

Харків 2024

УДК 658.8.339

АВТОМАТИЗАЦІЯ ІНДЕТИФІКАЦІЇ ВАНТАЖІВ НА БОНДОВИХ СКЛАДАХ

Д.Кривенко
Харківський національний університет радіоелектроніки
Україна, 61166, Харків, пр. Науки, 14
Email: denys.kryvenko@nure.ua

Анотація: Робота розглядає впровадження автоматизації на бондових складах для покращення обліку та зберігання товарів, зокрема через використання RFID, QR-кодів, роботів, конвеєрних і автономних систем. Це дозволяє знизити помилки, скоротити час обробки та витрати. Як напрямок для подальших досліджень запропоновано розробку програмного забезпечення для автоматизації митних процедур і логістики.

Ключові слова: автоматизація, бондові склади, RFID, QR-коди, роботизація, конвеєрні системи, автономні транспортні засоби, сортувальні системи, митні процедури, складська логістика, програмне забезпечення.

AUTOMATION OF CARGO IDENTIFICATION AT BONDED WAREHOUSES

D.Kryvenko
Kharkiv National University of Radio Electronics
Ukraine, 61166, Kharkiv, pr. Nauky, 14
Email: denys.kryvenko@nure.ua

Abstract: The paper considers the introduction of automation in bonded warehouses to improve the accounting and storage of goods, in particular through the use of RFID, QR codes, robots, conveyor and autonomous systems. This helps to reduce errors, reduce processing time and costs. As an area for further research, the author suggests developing software for automating customs procedures and logistics.

Keywords: automation, bonded warehouses, RFID, QR codes, robotics, conveyor systems, autonomous vehicles, sorting systems, customs procedures, warehouse logistics, software.

ВСТУП.У сучасному світі обсяг міжнародної торгівлі невпинно зростає, що, у свою чергу, призводить до збільшення кількості вантажів, які потребують проходження митних процедур і зберігання в спеціалізованих складах, зокрема митних та бондових складах.

Бондові склади (від англ. *bonded warehouse*) — це спеціалізовані склади, які служать для тимчасового зберігання товарів під митним контролем до завершення митного оформлення або до сплати необхідних митних зборів та податків. Вони виконують важливу роль у міжнародній торгівлі, оскільки дають можливість компаніям зберігати товари без необхідності негайно оплачувати митні збори, що є важливим для оптимізації їх грошових потоків і бізнес-процесів.

У той же час, традиційні методи ідентифікації та обліку вантажів, що базуються на ручній праці, все більше не відповідають вимогам сучасності. Такий підхід часто призводить до помилок, затримок у процесах і значних витрат на операційну діяльність. Система обліку, яка покладається на людську увагу, не завжди може забезпечити потрібну точність і швидкість, особливо при збільшенні обсягів торгівлі та складності митних процедур [1-5].

Автоматизація процесів ідентифікації вантажів стає необхідним кроком для оптимізації роботи бондових складів і забезпечення ефективного виконання митних процедур.

Автоматизація полягає у впровадженні технологій, систем або програмного забезпечення, які дозволяють виконувати різноманітні завдання та процеси без необхідності постійного втручання людини. Основною метою автоматизації є підвищення продуктивності, точності та швидкості виконання робіт, а також зниження витрат та мінімізація людського фактору, що здатне значно зменшити ймовірність помилок [2, 6-9].

Сьогодні для автоматизації процесів використовуються сучасні технології, такі як **RFID-мітки** (радіочастотна ідентифікація), **QR-коди** та **алгоритми комп'ютерного зору**. Завдяки

їм значно зростає швидкість обробки інформації, знижується ймовірність помилок, а також підвищується прозорість операцій. Це дозволяє митним і бондовим складам працювати швидше, точніше і зменшити витрати на операційну діяльність, що особливо важливо для підтримки високої конкуренції на міжнародному ринку.

Типи автоматизації складських приміщень. Роботи для збору замовлень (також відомі як pick-and-place роботи) - це пристрої, які виконують завдання збору товарів з полиць складу для комплектування замовлень. Ці роботи можуть бути як мобільними, так і стаціонарними, і вони можуть працювати в різних типах складів. Їхнє основне завдання - переміщення товарів з однієї точки в іншу, що значно прискорює процес обробки замовлень.

Використання таких роботів дає змогу прискорити процес пошуку і транспортування товарів складом, що значно скорочує час, який витрачається на виконання кожної операції. На відміну від людей, роботи виключають людські помилки, що підвищує точність складання і зменшує кількість неправильно комплектованих замовлень. Це особливо важливо в середовищах з високим обсягом замовлень, де навіть невеликі помилки можуть вплинути на ефективність роботи складу. Крім того, роботи для збирання замовлень можуть бути гнучкими та адаптуватися до різних типів товарів, що робить їх універсальними та ефективними для різних складських операцій.

Наприклад, роботи з маніпуляторами здатні витягувати товари з полиць і доставляти їх у потрібні місця для пакування або сортування, а мобільні роботи переміщуються складом, транспортуючи стелажі з товаром до робочих місць співробітників, які потім беруть потрібні товари для комплектації замовлень. Це скорочує час, який працівники витрачають на пошук і транспортування товарів.

Конвеєрні системи – це механізми, які автоматизують процес транспортування товарів по складу. Конвеєри можуть бути використані для переміщення товарів між різними зонами складу, такими як зони зберігання, сортування, пакування або відвантаження. Залежно від потреб складу, конвеєрні системи можуть бути як лінійними, так і з поворотними або з можливістю вертикального переміщення. [9-16].

Це скорочує час, необхідний для перенесення товарів вручну, і знижує ймовірність помилок, пов'язаних з людським фактором. Конвеєри дають змогу організувати складську логістику таким чином, щоб товари переміщалися в потрібні місця без зайвих затримок, що прискорює обробку замовлень і покращує загальну продуктивність складу. Вони також можуть бути адаптовані до різних умов і вимог, включно з роботою з великогабаритними або дрібними товарами.

Наприклад, лінійні конвеєри переміщують товари за прямими маршрутами від однієї точки складу до іншої. Вони часто використовуються для переміщення коробок або палетів із товаром, тоді як інтелектуальні конвеєри можуть змінювати напрямок руху товару залежно від різних чинників, наприклад, завантаженості певних зон складу або поточного обсягу товарів.

Автоматичні транспортні засоби (AGV) - це автономні роботи, які використовуються для транспортування товарів по складу. AGV дають змогу автоматизувати транспортування товарів, значно знижуючи залежність від людської праці. Це не тільки скорочує витрати на персонал, а й мінімізує ризики, пов'язані з помилками під час транспортування.

Ці транспортні засоби можуть працювати автономно, слідуючи заздалегідь заданими маршрутами або використовуючи сучасні сенсори для навігації в реальному часі. Це дає змогу підвищити безпеку на складі, виключаючи необхідність втручання людини в процес перевезення товарів, що зі свого боку знижує ймовірність травм і пошкоджень. AGV також мають високу гнучкість, оскільки можуть адаптуватися до змін у конфігурації складу і мінливих обсягів роботи.

Наприклад, сучасні AGV можуть використовувати лазерні або візуальні сенсори для орієнтації в просторі та навігації без необхідності у фіксованих шляхах. Це дає більшу гнучкість і дає змогу AGV адаптуватися до змін у структурі складе [5].

Також AGV можуть працювати у складі флотилії, координуючи свої дії з іншими транспортними засобами, що дає змогу ефективно розподіляти навантаження та маршрути.

Автоматичні сортувальні системи призначені для швидкого і точного розподілу товарів за різними категоріями або напрямками на складі. Ці системи можуть працювати з різними типами товарів, використовуючи сенсори, камери, RFID-мітки та інші технології для ідентифікації та сортування, що дає змогу швидко та точно розподілити товари за різними категоріями або зонами для подальшого опрацювання, що особливо важливо на великих складах із високою товарообігом.

Використання автоматичних сортувальників [9] дає змогу знизити кількість помилок, оскільки всі операції виконуються за допомогою сенсорів і програмного забезпечення, що виключає людський фактор. Ці системи можуть бути налаштовані для сортування товарів за безліччю різних критеріїв, як-от розмір, вага або призначення, що робить процес гнучкішим і адаптованим до специфіки складу.

У результаті прискорюється обробка замовлень, підвищується точність і зменшуються витрати на трудові ресурси, що призводить до поліпшення загальної продуктивності складу.

Прикладом можуть слугувати конвеєрні, ротаційні сортувальники та інтелектуальні системи сортування.

Конвеєрні сортувальники - це системи, які використовують конвеєри для переміщення товарів різними зонами складу, що дає змогу ефективно організувати потік матеріалів і прискорити процеси сортування та розподілу.

Ці системи оснащені спеціальними механізмами, які можуть перенаправляти товар у потрібні напрямки залежно від заздалегідь визначених критеріїв. Сортування товарів може відбуватися за різними параметрами, такими як розмір, вага, тип продукції або місце призначення.

Наприклад, товари можуть бути спрямовані в зону пакування або відправлення залежно від їхніх характеристик або місця доставки. Такий підхід значно знижує час, необхідний для переміщення товарів між різними ділянками складу і покращує точність розподілу продукції. Конвеєрні сортувальники особливо корисні для складів з великим обсягом однотипної продукції, де важливо забезпечити високу швидкість і точність сортування.

Ротаційні сортувальники використовують поворотні механізми для перенаправлення товарів у потрібні зони складу. Це дає змогу ефективно розділити товари за різними категоріями або маршрутами, що прискорює процес обробки замовлень.

На відміну від лінійних конвеєрів, які можуть пересувати товар по прямій лінії, ротаційні системи здатні керувати товаром на поворотах і в різних напрямках, забезпечуючи більшу кількість можливих шляхів для його переміщення. Такі системи часто використовуються у великих розподільчих центрах, де товари можуть бути спрямовані в різні зони для подальшого оброблення або пакування, залежно від їхніх характеристик або типу замовлення.

Ротаційні сортувальники підвищують гнучкість складської логістики та дають змогу швидко адаптуватися до змін у потоці товарів, покращуючи швидкість комплектації замовлень і зменшивши час на їх обробку.

Інтелектуальні системи сортування застосовують передові технології, як-от датчики, камери та алгоритми штучного інтелекту, для автоматичного розпізнавання та сортування товарів. Ці системи можуть точно ідентифікувати товари за допомогою різних сенсорів, враховуючи параметри, як-от розмір упаковки, штрих-коди, RFID-мітки або навіть особливості форми товару.

Сортування може відбуватися за безліччю критеріїв, що дає змогу системі працювати з різноманітними типами продукції та адаптуватися до змін в асортименті складу. Камери і датчики можуть аналізувати товар у реальному часі, ухвалюючи рішення про його сортування з високою точністю, що мінімізує помилки, пов'язані з людським фактором.

Інтелектуальні системи сортування особливо ефективні в середовищах із великими об'ємами різноманітної продукції, де потрібна швидка й точна обробка різних товарів, включно з дрібними деталями, коробками, або нестандартними упаковками. Такі системи значно прискорюють процеси комплектації замовлень, покращуючи загальну ефективність і знижуючи витрати на трудові ресурси.

ВИСНОВКИ. Отже, зі зростанням міжнародної торгівлі збільшується потреба в ефективному зберіганні товарів, що потребують митних процедур, зокрема на бондових

складах. Традиційні методи обліку товарів, що базуються на ручній праці, вже не відповідають вимогам сучасності, призводячи до помилок і затримок. Тому автоматизація процесів, таких як використання RFID, QR-кодів, роботів для збору замовлень, конвеєрних і автономних транспортних систем, а також автоматичних сортувальних систем, стає необхідною для підвищення ефективності, точності і швидкості роботи складів. Ці технології значно знижують витрати на операційну діяльність і помилки, підвищують продуктивність та гнучкість складських процесів.

Напрямком для подальших досліджень може бути розробка програмного забезпечення для автоматизації митних процедур і складської логістики, що забезпечить інтеграцію сучасних технологій для оптимізації обробки вантажів на різних етапах міжнародної торгівлі.

ЛІТЕРАТУРА

1. Основи наукових досліджень : підручник / І. Ш. Невлюдов, Ю. М. Олександров, А. О. Андрусевич, О. О. Чала ; М-во освіти і науки України, Харків. нац. ун-т радіоелектроніки. – Prague : OKTAN PRINT, 2024. – 468 с. DOI https://doi.org/10.46489/ONDNP-24-12
2. Vzhesnievskyi M. Автоматизація внутрішньо-складських виробничих логістичних процесів для впровадження концепції industry 4.0: енергоощадливість, продуктивність, мобільність, модульність, автономність / M. Vzhesnievskyi, O. Chala // Системи управління, навігації та зв'язку. Збірник наукових праць. – Полтава: ПНТУ, 2024. – Т. 2 (76). – С. 34-38. – doi:https://doi.org/10.26906/SUNZ.2024.2.034.
3. Шостенко С. С. Архітектура програмного забезпечення для супроводження автоматизованих систем оповіщення на виробництві / С. С. Шостенко, О. О. Чала // Виробництво & Мехатронні Системи 2022 : зб. тез. доп. VI-ої Міжнар. конф., 21-22 жовтня 2022 р. – Харків, 2022. – С. 115-117.
4. Чала, О., Сливка, А. (2023) Рівні засобів IIoT в інформаційних технологіях. Виробництво & Мехатронні Системи: матеріали VII Міжнародної конференції , Харків, С. 51-60.
5. Вжеснєвський М. О. Розробка кінематичної схеми транспортувального шатлу для внутрішньоскладської виробничої логістики / М. О. Вжеснєвський, О. О. Чала, Ю. В. Ромашов // Комп'ютерно-інтегрованих технології, автоматизація та робототехніка : матеріали І-ої Всеукр. конф., Харків, 16-17 травня : тези доповідей. – Харків, 2024. – С. 6-10. URI https://openarchive.nure.ua/handle/document/26473
6. Lighting Control Module Software Development / Y. Vizir, O. Chala, S. Maksymova, Ahmad Alkhalaileh // Journal of Universal Science Research, 2024. – Vol. 2(2). – P. 29–42. URI https://openarchive.nure.ua/handle/document/27513
7. Nevliudov, I., Vzhesnievskyi, M., Romashov, Y., & Chala, O. (2023). Mathematical modeling of mechatronic shuttles as automation objects for multilevel systems of intra-warehouse logistics. INNOVATIVE TECHNOLOGIES AND SCIENTIFIC SOLUTIONS FOR INDUSTRIES, (4(26), 135–144. https://doi.org/10.30837/ITSSI.2023.26.135
8. Igor, N., Svitlana, M., Olena, C., Artem, B., & Maksym, V. (2023). Automated Logistics Processes Improvement in Logistics Facilities. Multidisciplinary Journal of Science and Technology, 3(3), 157-170.
9. Zharikova, I., Nevliudov, I., Maksymova, S., & Chala, O. (2023). Automatic Machine of Plastic Bottles and Aluminum Cans Collection for Recycling. Journal of Universal Science Research. – 2023. – ₮ 1(11). – P. 169–178
10. Гіль, А., Чала, О., Филипенко, О. (2021). Промислові інтерфейси та протоколи передачі даних інтегрованих систем для автоматизованого управління в умовах Industry 4.0. Виробництво & Мехатронні Системи 2021: матеріали V-ої Міжнародної конференції, Харків, 127-30.
11. I. Nevliudov, A. Bronnikov, O. Chala "Improvement and Optimization of Automated Logistics Processes in Logistics Premises," 2023 IEEE 5th International Conference on Modern Electrical and Energy System (MEES), Kremenchuk, Ukraine, 2023, pp. 1-6, doi: 10.1109/MEES61502.2023.10402386.

12. Gurin, Dmytro, et al. "Using Convolutional Neural Networks to Analyze and Detect Key Points of Objects in Image." Multidisciplinary Journal of Science and Technology 4.9 (2024): 5-15

13. A Small-Sized Robot Prototype Development Using 3D Printing / I. Nevliudov, V. Yevsieiev, S. Maksymova, O. Chala // CAD In Machinery Design Implementation and Educational Issues (CADMD'2023) : proceedings of the XXXI International Conference. (Conference in memory of Professor Jerry Wrobel), Suprasl, 26-28 October, 2023. – Suprasl, 2023. – P. 12.

14. O. Chala, A. Bronnikov, N. Igor and D. Mospan, "The Use of Neural Networks for the Technological Objects Recognition Tasks in Computer-Integrated Manufacturing," 2022 IEEE 4th International Conference on Modern Electrical and Energy System (MEES), Kremenchuk, Ukraine, 2022, pp. 1-5, doi: 10.1109/MEES58014.2022.10005750.

15. Buts D. Signals Collisions Detection In Wireless Networks / D. Buts, O. Chala, S. Maksymova // Journal of Universal Science Research. – 2023. – № 1(11). – P. 156–168.

16. Basiuk, V., Maksymova, S., Chala, O., & Abu-Jassar, A. (2024). COMMAND SYSTEM FOR MOVEMENT CONTROL DEVELOPMENT. Multidisciplinary Journal of Science and Technology, 4(6), 248-255 ▪

ДОДАТОК Б

Код вікна авторизації користувачів

```
public partial class LoginWindow : Window
{
    public LoginWindow()
    {
        InitializeComponent();
    }
    private void Login_Click(object sender, RoutedEventArgs e)
    {
        PerformLogin();
    }
    private void Input_KeyUp(object sender, KeyEventArgs e)
    {
        if (e.Key == Key.Enter)
        {
            PerformLogin();
        }
    }
    private void PerformLogin()
    {
        string login = LoginBox.Text ?? "";
        string password = PasswordBox.Text ?? "";
        if (string.IsNullOrWhiteSpace(login) || string.IsNullOrWhiteSpace(password))
        {
            ShowError("⚠ Введіть логін та пароль");
            return;
        }
        try
        {
            using (var db = new DatabaseContext())
            {
                var user = db.Users.FirstOrDefault(u => u.Username == login && u.Password == password);

                if (user != null)
                {
                    try
                    {
                        var mainWindow = new MainWindow();
                        mainWindow.Show();
                        Close();
                    }
                    catch (Exception ex)
                    {
                        string desktopPath = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);
                        string errorFile = Path.Combine(desktopPath, "CRASH_REPORT.txt");
                        string errorMessage = $"ЧАС: {DateTime.Now}\n" +
                                              $"ПОМИЛКА: {ex.Message}\n" +
                                              $"ДЕТАЛІ (StackTrace):\n{ex.StackTrace}";
                        File.WriteAllText(errorFile, errorMessage);
                        ShowError("CRASH! Див. файл на роб. столі");
                    }
                }
                else
                {
                    ShowError("❌ Невірний логін або пароль");
                }
            }
        }
        catch (Exception exDb)
        {
            ShowError($"Помилка БД: {exDb.Message}");
        }
    }
    private void ShowError(string message)
    {
        ErrorText.Text = message;
        ErrorText.IsVisible = true;
    }
    private void Exit_Click(object sender, RoutedEventArgs e)
    {
        Close();
    }
    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК В

Код головного вікна

```
public partial class MainWindow : Window
{
    private List<StorageUnit> _allStorageUnits = new();
    private List<Product> _allProducts = new();
    private List<CustomsDeclaration> _allDeclarations = new();

    public MainWindow()
    {
        InitializeComponent();

        this.AddHandler(PointerPressedEvent, (sender, e) =>
        {
            var source = e.Source as Visual;
            if (source == null) return;

            if (source.FindAncestorOfType<DataGridRow>() != null ||
                source.FindAncestorOfType<Button>() != null ||
                source.FindAncestorOfType<ScrollBar>() != null ||
                source.FindAncestorOfType<DataGridColumnHeader>() != null)
            {
                return;
            }

            if (GoodsGrid != null) GoodsGrid.SelectedItem = null;
            if (ProductsGrid != null) ProductsGrid.SelectedItem = null;
            if (DeclarationsGrid != null) DeclarationsGrid.SelectedItem = null;

        }, RoutingStrategies.Tunnel);

        QuestPDF.Settings.License = LicenseType.Community;
        LoadDataFromDatabase();
    }

    public void DataGrid_PointerPressed(object? sender, PointerPressedEventArgs e)
    {
    }

    private void LoadDataFromDatabase()
    {
        using var db = new DatabaseContext();

        _allStorageUnits = db.StorageUnits
            .Include(u => u.Product)
            .Include(u => u.Declaration)
            .Include(u => u.Location)
            .ToList();
        UpdateGoodsGrid(_allStorageUnits);

        _allProducts = db.Products.ToList();
        UpdateProductsGrid(_allProducts);

        _allDeclarations = db.CustomsDeclarations.ToList();
        UpdateDeclarationsGrid(_allDeclarations);
    }

    private void UpdateGoodsGrid(IEnumerable<StorageUnit> items)
    {
        var list = items.ToList();
        int c = 1;
        foreach (var item in list) { item.DisplayNum = c++; }

        var grid = this.FindControl<DataGrid>("GoodsGrid");
        if (grid != null) grid.ItemsSource = list;
    }

    private void UpdateProductsGrid(IEnumerable<Product> items)
    {
        var list = items.ToList();
        int c = 1;
        foreach (var item in list) { item.DisplayNum = c++; }

        var grid = this.FindControl<DataGrid>("ProductsGrid");
        if (grid != null) grid.ItemsSource = list;
    }

    private void UpdateDeclarationsGrid(IEnumerable<CustomsDeclaration> items)
    {
        var list = items.ToList();
        int c = 1;
        foreach (var item in list)
        {
```

```
            item.DisplayNum = c++;
            var relatedUnit = _allStorageUnits.FirstOrDefault(u => u.DeclarationId == item.Id);
            item.FirstQr = relatedUnit?.QrCodeHash ?? "-";
        }

        var grid = this.FindControl<DataGrid>("DeclarationsGrid");
        if (grid != null) grid.ItemsSource = list;
    }
    private void SearchBox_KeyUp(object sender, KeyEventArgs e)
    {
        var searchBox = sender as TextBox;
        string text = searchBox?.Text?.ToLower() ?? "";

        var tabControl = this.FindControl<TabControl>("MainTabControl");
        int tabIndex = tabControl.SelectedIndex;

        switch (tabIndex)
        {
            case 0:
                var filteredGoods = string.IsNullOrWhiteSpace(text)
                    ? _allStorageUnits
                    : _allStorageUnits.Where(x =>
                        (x.Product.Name != null && x.Product.Name.ToLower().Contains(text)) ||
                        (x.Product.Sku != null && x.Product.Sku.ToLower().Contains(text)) ||
                        (x.Declaration.DeclarationNumber != null &&
claration.DeclarationNumber.ToLower().Contains(text)) ||
                        (x.BatchNumber != null && x.BatchNumber.ToLower().Contains(text))
                      ).ToList();
                UpdateGoodsGrid(filteredGoods);
                break;

            case 1:
                var filteredProducts = string.IsNullOrWhiteSpace(text)
                    ? _allProducts
                    : _allProducts.Where(x =>
                        (x.Name != null && x.Name.ToLower().Contains(text)) ||
                        (x.Sku != null && x.Sku.ToLower().Contains(text))
                      ).ToList();
                UpdateProductsGrid(filteredProducts);
                break;

            case 2:
                var filteredDecl = string.IsNullOrWhiteSpace(text)
                    ? _allDeclarations
                    : _allDeclarations.Where(x =>
                        (x.DeclarationNumber != null && x.DeclarationNumber.ToLower().Contains(text))
                      ).ToList();
                UpdateDeclarationsGrid(filteredDecl);
                break;
        }
    }
    private void TogglePane_Click(object sender, RoutedEventArgs e)
    {
        var splitView = this.FindControl<SplitView>("MainSplitView");
        splitView.IsPaneOpen = !splitView.IsPaneOpen;
    }
    private async void AddUser_Click(object sender, RoutedEventArgs e)
    {
        var addUserWin = new AddUserWindow();

        await addUserWin.ShowDialog(this);
    }
    private async void OpenUsersManager_Click(object sender, RoutedEventArgs e)
    {
        var usersWin = new UsersWindow();
        await usersWin.ShowDialog(this);
    }
    private async void AddGoods_Click(object sender, RoutedEventArgs e)
    {
        var addWindow = new AddGoodsWindow();
        await addWindow.ShowDialog(this);
        LoadDataFromDatabase();
    }
    private async void EditGoods_Click(object sender, RoutedEventArgs e)
    {
        var grid = this.FindControl<DataGrid>("GoodsGrid");
        var selectedItem = grid.SelectedItem as StorageUnit;

        if (selectedItem == null)
        {
            await new MessageWindow("⚠ Оберіть запис для редагування!").ShowDialog(this);
            return;
        }

        var editWin = new EditGoodsWindow(selectedItem);
        await editWin.ShowDialog(this);
        LoadDataFromDatabase();
    }
    private async void DeleteGoods_Click(object sender, RoutedEventArgs e)
```

```
    {
        var grid = this.FindControl<DataGrid>("GoodsGrid");
        var selectedItem = grid.SelectedItem as StorageUnit;

        if (selectedItem == null)
        {
            await new MessageWindow("❌ Оберіть товар у таблиці!").ShowDialog(this);
            return;
        }

        var confirmWindow = new ConfirmWindow();
        await confirmWindow.ShowDialog(this);
        if (!confirmWindow.IsConfirmed) return;

        using (var db = new DatabaseContext())
        {
            var itemToDelete = db.StorageUnits
                .Include(u => u.Product)
                .Include(u => u.Declaration)
                .Include(u => u.Location)
                .FirstOrDefault(u => u.Id == selectedItem.Id);

            if (itemToDelete != null)
            {
                int prodId = itemToDelete.ProductId;
                int declId = itemToDelete.DeclarationId;
                int locId = itemToDelete.LocationId ?? 0;
                string qrCodeName = itemToDelete.QrCodeHash;

                db.StorageUnits.Remove(itemToDelete);

                bool isProductUsed = db.StorageUnits.Any(u => u.ProductId == prodId && u.Id != ToDelete.Id);
                if (!isProductUsed && prodId != 0)
                {
                    var p = db.Products.Find(prodId);
                    if (p != null) db.Products.Remove(p);
                }

                bool isDeclUsed = db.StorageUnits.Any(u => u.DeclarationId == declId && u.Id != ToDelete.Id);
                if (!isDeclUsed && declId != 0)
                {
                    var d = db.CustomsDeclarations.Find(declId);
                    if (d != null) db.CustomsDeclarations.Remove(d);
                }

                bool isLocUsed = db.StorageUnits.Any(u => u.LocationId == locId && u.Id != itemToDelete.Id);
                if (!isLocUsed && locId != 0)
                {
                    var l = db.StorageLocations.Find(locId);
                    if (l != null) db.StorageLocations.Remove(l);
                }

                if (!string.IsNullOrEmpty(qrCodeName))
                {
                    try
                    {
                        string folderPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "QR_Codes");
                        string filePath = Path.Combine(folderPath, $"{qrCodeName}.png");
                        if (File.Exists(filePath)) File.Delete(filePath);
                    }
                    catch (Exception ex) { Debug.WriteLine(ex.Message); }
                }

                db.SaveChanges();
            }
        }
        LoadDataFromDatabase();
    }
    public async void DeleteProductRef_Click(object sender, RoutedEventArgs e)
    {
        var grid = this.FindControl<DataGrid>("ProductsGrid");
        var selectedProduct = grid.SelectedItem as Product;
        if (selectedProduct == null) return;

        using (var db = new DatabaseContext())
        {
            bool isUsed = db.StorageUnits.Any(u => u.ProductId == selectedProduct.Id);
            if (isUsed)
            {
                await new MessageWindow($"⛔ Товар '{selectedProduct.Name}' використовується на
ді!").ShowDialog(this);
                return;
            }

            var confirm = new ConfirmWindow();
            await confirm.ShowDialog(this);
            if (confirm.IsConfirmed)
            {
```

```
                var p = db.Products.Find(selectedProduct.Id);
                if (p != null) { db.Products.Remove(p); db.SaveChanges(); }
            }
        }
        LoadDataFromDatabase();
    }
    public async void DeleteDeclaration_Click(object sender, RoutedEventArgs e)
    {
        var grid = this.FindControl<DataGrid>("DeclarationsGrid");
        var selectedDecl = grid.SelectedItem as CustomsDeclaration;
        if (selectedDecl == null) return;

        using (var db = new DatabaseContext())
        {
            bool isUsed = db.StorageUnits.Any(u => u.DeclarationId == selectedDecl.Id);
            if (isUsed)
            {
                await new MessageWindow("⛔ По цій декларації ще є товари на складі!").ShowDialog(this);
                return;
            }

            var confirm = new ConfirmWindow();
            await confirm.ShowDialog(this);
            if (confirm.IsConfirmed)
            {
                var d = db.CustomsDeclarations.Find(selectedDecl.Id);
                if (d != null) { db.CustomsDeclarations.Remove(d); db.SaveChanges(); }
            }
        }
        LoadDataFromDatabase();
    }
    private async void ShowReport_Click(object sender, RoutedEventArgs e)
    {
        var tabControl = this.FindControl<TabControl>("MainTabControl");
        int currentTab = tabControl.SelectedIndex;

        switch (currentTab)
        {
            case 0:
                var grid0 = this.FindControl<DataGrid>("GoodsGrid");
                var unit = grid0.SelectedItem as StorageUnit;
                if (unit == null) { await new MessageWindow("⚠ Оберіть запис у таблиці!").ShowDialog(this); rn;
}
                new ReportWindow(unit).Show();
                break;

            case 1:
                var grid1 = this.FindControl<DataGrid>("ProductsGrid");
                var product = grid1.SelectedItem as Product;
                if (product == null) { await new MessageWindow("⚠ Оберіть товар у днику!").ShowDialog(this);
return; }
                new ReportWindow(product, _allStorageUnits).Show();
                break;

            case 2:
                var grid2 = this.FindControl<DataGrid>("DeclarationsGrid");
                var decl = grid2.SelectedItem as CustomsDeclaration;
                if (decl == null) { await new MessageWindow("⚠ Оберіть декларацію!").ShowDialog(this); rn; }
                new ReportWindow(decl, _allStorageUnits).Show();
                break;
        }
    }
    public async void ExportProductReport_Click(object sender, RoutedEventArgs e)
    {
        var grid = this.FindControl<DataGrid>("ProductsGrid");
        var selectedProd = grid.SelectedItem as Product;
        if (selectedProd == null) return;

        var win = new ReportWindow(selectedProd, _allStorageUnits);
        win.Show();
    }
    public async void ExportDeclarationReport_Click(object sender, RoutedEventArgs e)
    {
        var grid = this.FindControl<DataGrid>("DeclarationsGrid");
        var selectedDecl = grid.SelectedItem as CustomsDeclaration;
        if (selectedDecl == null) return;

        var win = new ReportWindow(selectedDecl, _allStorageUnits);
        win.Show();
    }
    private async void ExportFullReport_Click(object sender, RoutedEventArgs e)
    {
        List<StorageUnit> dbData;
        using (var db = new DatabaseContext())
        {
            dbData = db.StorageUnits
                .Include(u => u.Product)
                .Include(u => u.Declaration)
```

```
                .Include(u => u.Location)
                .ToList();
        }

        if (dbData.Count == 0)
        {
            await new MessageWindow("❌ Склад порожній! Немає даних для звіту.").ShowDialog(this);
            return;
        }
        try
        {
            var topLevel = TopLevel.GetTopLevel(this);
            var file = await topLevel.StorageProvider.SaveFilePickerAsync(new FilePickerSaveOptions
            {
                Title = "Зберегти повну відомість",
                DefaultExtension = "pdf",
                SuggestedFileName = $"Відомість_Повна_{DateTime.Now:dd-MM-yyyy}",
                FileTypeChoices = new[] { new FilePickerFileType("PDF Document") { Patterns = new[] { "*.pdf" }
            });

            if (file == null) return;

            var document = Document.Create(container =>
            {
                container.Page(page =>
                {
                    page.Size(PageSizes.A4.Landscape());
                    page.Margin(1, Unit.Centimetre);
                    page.PageColor(Colors.White);
                    page.DefaultTextStyle(x => x.FontSize(10));

                    page.Header().Height(3, Unit.Centimetre).Row(row =>
                    {
                        row.RelativeItem().Column(col =>
                        {
                            col.Item().Text("OLE-PRO WAREHOUSE").SemiBold().FontSize(14);
                            col.Item().Text("ІНВЕНТАРИЗАЦІЙНИЙ ОПИС").FontSize(18).Bold().Underline();
                            col.Item().Text("Повний список залишків з
.FontSize(10).Italic().FontColor(Colors.Grey.Medium);
                        });

                        row.ConstantItem(200).Column(col =>
                        {
                            col.Item().AlignRight().Text($"Дата: {DateTime.Now:dd.MM.yyyy}");
                            col.Item().AlignRight().Text($"Час: {DateTime.Now:HH:mm}");
                            col.Item().AlignRight().Text($"Всього позицій: {dbData.Count}");
                        });
                    });

                    page.Content().PaddingVertical(10).Table(table =>
                    {
                        table.ColumnsDefinition(columns =>
                        {
                            columns.ConstantColumn(30);
                            columns.RelativeColumn(3);
                            columns.RelativeColumn(2);
                            columns.RelativeColumn(1);
                            columns.RelativeColumn(1);
                            columns.RelativeColumn(2);
                            columns.RelativeColumn(2);
                        });

                        table.Header(header =>
                        {
                            IContainer H(IContainer c) =>
ckground(Colors.Grey.Lighten3).Border(1).BorderColor(Colors.Grey.Lighten1).Padding(5);
                            header.Cell().Element(H).Text("#");
                            header.Cell().Element(H).Text("Найменування");
                            header.Cell().Element(H).Text("Артикул");
                            header.Cell().Element(H).Text("Кіл-ть");
                            header.Cell().Element(H).Text("Од.");
                            header.Cell().Element(H).Text("Декларація");
                            header.Cell().Element(H).Text("Місце");
                        });

                        int i = 1;
                        foreach (var item in dbData)
                        {
                            var bg = i % 2 == 0 ? Colors.White : Colors.Grey.Lighten4;
                            IContainer C(IContainer c) =>
ckground(bg).Border(1).BorderColor(Colors.Grey.Lighten2).Padding(5);

                            table.Cell().Element(C).Text(i.ToString());
                            table.Cell().Element(C).Text(item.Product?.Name ?? "-");
                            table.Cell().Element(C).Text(item.Product?.Sku ?? "-");
                            table.Cell().Element(C).Text(item.Quantity.ToString());
                            table.Cell().Element(C).Text(item.Product?.Unit ?? "-");
                            table.Cell().Element(C).Text(item.Declaration?.DeclarationNumber ?? "-");
                            table.Cell().Element(C).Text(item.Location?.FullAddress ?? "-");
```

```
                        i++;
                    }
                });

                page.Footer().AlignRight().Text(x => { x.Span("Сторінка "); x.CurrentPageNumber(); an(" з ");
x.TotalPages(); });
                });
            });

            using var stream = await file.OpenWriteAsync();
            document.GeneratePdf(stream);
            await new MessageWindow("✅ Повна відомість успішно збережена!").ShowDialog(this);
        }
        catch (Exception ex) { await new MessageWindow($"❌ Помилка: {ex.Message}").ShowDialog(this); }
    }
    private void OpenQrFolder_Click(object sender, RoutedEventArgs e)
    {
        string folderPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "QR_Codes");
        if (!Directory.Exists(folderPath)) Directory.CreateDirectory(folderPath);

        try
        {
            Process.Start(new ProcessStartInfo
            {
                FileName = folderPath,
                UseShellExecute = true,
                Verb = "open"
            });
        }
        catch (Exception ex) { new MessageWindow($"Не вдалося відкрити папку: {ex.Message}").Show(); }
    }
    private void OpenQrInTable_Click(object sender, RoutedEventArgs e)
    {
        var button = sender as Button;
        var unit = button.DataContext as StorageUnit;
        string qrCode = unit?.QrCodeHash;

        if (unit == null)
        {
            var decl = button.DataContext as CustomsDeclaration;
            if (decl != null) qrCode = decl.FirstQr;
        }

        if (string.IsNullOrEmpty(qrCode) || qrCode == "-") return;

        var qrWindow = new QrDisplayWindow(qrCode);
        qrWindow.ShowDialog(this);
    }
    private async void ScanQr_Click(object sender, RoutedEventArgs e)
    {
        var topLevel = TopLevel.GetTopLevel(this);
        string qrFolderPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "QR_Codes");
        if (!Directory.Exists(qrFolderPath)) Directory.CreateDirectory(qrFolderPath);

        var startLocation = await topLevel.StorageProvider.TryGetFolderFromPathAsync(qrFolderPath);
        var files = await topLevel.StorageProvider.OpenFilePickerAsync(new FilePickerOpenOptions
        {
            Title = "Оберіть зображення QR-коду",
            AllowMultiple = false,
            SuggestedStartLocation = startLocation,
            FileTypeFilter = new[] { FilePickerFileTypes.ImagePng, FilePickerFileTypes.ImageAll }
        });

        if (files.Count == 0) return;

        string imagePath = files[0].Path.LocalPath;
        string decodedText = null;

        try
        {
            var barcodeReader = new ZXing.Windows.Compatibility.BarcodeReader();
            barcodeReader.Options.PossibleFormats = new List<BarcodeFormat> { BarcodeFormat.QR_CODE };

            using (var bitmap = new System.Drawing.Bitmap(imagePath))
            {
                var result = barcodeReader.Decode(bitmap);
                if (result != null) decodedText = result.Text;
            }
        }
        catch (Exception ex)
        {
            await new MessageWindow($"Помилка читання: {ex.Message}").ShowDialog(this);
            return;
        }

        if (string.IsNullOrEmpty(decodedText))
        {
            await new MessageWindow("❌ QR-код не знайдено.").ShowDialog(this);
```

```
            return;
        }

        using (var db = new DatabaseContext())
        {
            var foundUnit = db.StorageUnits
                .Include(u => u.Product)
                .Include(u => u.Declaration)
                .Include(u => u.Location)
                .FirstOrDefault(u => u.QrCodeHash == decodedText);

            if (foundUnit != null)
            {
                new ReportWindow(foundUnit).Show();
            }
            else
            {
                await new MessageWindow($"⚠ Код зчитано: {decodedText}\nАле в базі такого товару
є.").ShowDialog(this);
            }
        }
    }
    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed) BeginMoveDrag(e);
    }
    private void CloseApp_Click(object sender, RoutedEventArgs e) => Close();
    private void MaximizeApp_Click(object sender, RoutedEventArgs e)
    {
        WindowState = WindowState == WindowState.Maximized ? WindowState.Normal : WindowState.Maximized;
    }
    private void MinimizeApp_Click(object sender, RoutedEventArgs e) => WindowState = WindowState.Minimized;
}
```

ДОДАТОК Г

Код вікна внесення нового товару

```
public partial class AddGoodsWindow : Window
{
    public AddGoodsWindow()
    {
        InitializeComponent();
    }

    private void Save_Click(object sender, RoutedEventArgs e)
    {
        string pName = ProductName.Text ?? "Без назви";
        string pSku = ProductSku.Text ?? "";
        string pUnit = ProductUnit.Text ?? "шт";
        string dNum = DeclNumber.Text ?? "Без номера";
        string lZone = LocZone.Text ?? "A";
        string lShelf = LocShelf.Text ?? "1";
        string lCell = LocCell.Text ?? "1";
        double quantity = (double)(Qty.Value ?? 0);
        string batch = BatchNum.Text ?? "";

        using var db = new DatabaseContext();

        var product = new Product { Name = pName, Sku = pSku, Unit = pUnit };
        var declaration = new CustomsDeclaration { DeclarationNumber = dNum, EntryDate = DateTime.Now,
ExpirationDate = DateTime.Now.AddDays(1095), };
        var location = new StorageLocation { ZoneCode = lZone, ShelfNumber = lShelf, CellNumber = lCell };
        string uniqueId = Guid.NewGuid().ToString().Substring(0, 6).ToUpper();
        string qrCodeValue = $"WMS-{DateTime.Now:yyMM}-{uniqueId}";
        try
        {
            string folderPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "QR_Codes");

            if (!Directory.Exists(folderPath))
            {
                Directory.CreateDirectory(folderPath);
            }
            using var qrGenerator = new QRCodeGenerator();
            var qrCodeData = qrGenerator.CreateQrCode(qrCodeValue, QRCodeGenerator.ECCLevel.Q);
            var qrCode = new PngByteQRCode(qrCodeData);
            byte[] qrCodeImage = qrCode.GetGraphic(20);
            string filePath = Path.Combine(folderPath, $"{qrCodeValue}.png");
            File.WriteAllBytes(filePath, qrCodeImage);
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"Помилка збереження QR: {ex.Message}");
        }
        var storageUnit = new StorageUnit
        {
            Product = product,
            Declaration = declaration,
            Location = location,
            Quantity = (float)quantity,
            BatchNumber = batch,
            ExpDateProduct = DateTime.Now.AddYears(2),
            QrCodeHash = qrCodeValue
        };

        db.StorageUnits.Add(storageUnit);
        db.SaveChanges();

        Close();
    }
    private void Cancel_Click(object sender, RoutedEventArgs e)
    {
        Close();
    }
    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК Д

Код вікна редагування товару

```
public partial class EditGoodsWindow : Window
{
    private readonly int _unitId;

    public EditGoodsWindow()
    {
        InitializeComponent();
    }
    public EditGoodsWindow(StorageUnit unitToEdit)
    {
        InitializeComponent();
        _unitId = unitToEdit.Id;
        ProductName.Text = unitToEdit.Product.Name;
        ProductSku.Text = unitToEdit.Product.Sku;
        ProductUnit.Text = unitToEdit.Product.Unit;

        DeclNumber.Text = unitToEdit.Declaration.DeclarationNumber;

        LocZone.Text = unitToEdit.Location.ZoneCode;
        LocShelf.Text = unitToEdit.Location.ShelfNumber;
        LocCell.Text = unitToEdit.Location.CellNumber;

        Qty.Value = (decimal)unitToEdit.Quantity;
        BatchNum.Text = unitToEdit.BatchNumber;
    }

    private void Save_Click(object sender, RoutedEventArgs e)
    {
        using var db = new DatabaseContext();

        var unit = db.StorageUnits
            .Include(u => u.Product)
            .Include(u => u.Declaration)
            .Include(u => u.Location)
            .FirstOrDefault(u => u.Id == _unitId);

        if (unit != null)
        {
            unit.Product.Name = ProductName.Text ?? "";
            unit.Product.Sku = ProductSku.Text ?? "";
            unit.Product.Unit = ProductUnit.Text ?? "";

            unit.Declaration.DeclarationNumber = DeclNumber.Text ?? "";

            unit.Location.ZoneCode = LocZone.Text ?? "";
            unit.Location.ShelfNumber = LocShelf.Text ?? "";
            unit.Location.CellNumber = LocCell.Text ?? "";

            unit.Quantity = (float)(Qty.Value ?? 0);
            unit.BatchNumber = BatchNum.Text ?? "";

            db.SaveChanges();
        }

        Close();
    }
    private void Cancel_Click(object sender, RoutedEventArgs e)
    {
        Close();
    }
    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК Е

Код вікна підтвердження

```
public partial class ConfirmWindow : Window
{
    public bool IsConfirmed { get; private set; } = false;
    public ConfirmWindow()
    {
        InitializeComponent();
    }
    private void Yes_Click(object sender, RoutedEventArgs e)
    {
        IsConfirmed = true;
        Close();
    }

    private void No_Click(object sender, RoutedEventArgs e)
    {
        IsConfirmed = false;
        Close();
    }
    private void TitleBar_PointerPressed(object sender, Avalonia.Input.PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК Ж

Код вікна повідомлення

```
public partial class MessageWindow : Window
{
    public MessageWindow(string message)
    {
        InitializeComponent();
        MessageText.Text = message;
    }
    private void Ok_Click(object sender, RoutedEventArgs e)
    {
        Close();
    }
    private void TitleBar_PointerPressed(object sender, Avalonia.Input.PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК И

Код вікна відображення qr-коду

```
public partial class QrDisplayWindow : Window
{
    public QrDisplayWindow() { InitializeComponent(); }

    public QrDisplayWindow(string qrCodeHash)
    {
        InitializeComponent();

        QrText.Text = qrCodeHash;

        if (!string.IsNullOrEmpty(qrCodeHash))
        {
            using (QRCodeGenerator qrGenerator = new QRCodeGenerator())
            {
                QRCodeData qrCodeData = qrGenerator.CreateQrCode(qrCodeHash, QRCodeGenerator.ECCLevel.Q);
                PngByteQRCode qrCode = new PngByteQRCode(qrCodeData);
                byte[] qrBytes = qrCode.GetGraphic(20);

                using (var stream = new MemoryStream(qrBytes))
                {
                    QrImage.Source = new Bitmap(stream);
                }
            }
        }
    }
    private void Close_Click(object sender, RoutedEventArgs e)
    {
        Close();
    }
    private void TitleBar_PointerPressed(object sender, Avalonia.Input.PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК К

Код вікна генерації звітності

```
public partial class ReportWindow : Window
{
    private readonly StorageUnit? _unit;
    private readonly Product? _product;
    private readonly CustomsDeclaration? _declaration;
    private readonly List<StorageUnit> _dataList = new();
    private readonly string _reportType;

    public string CurrentDate => DateTime.Now.ToString("dd.MM.yyyy HH:mm");

    public ReportWindow()
    {
        InitializeComponent();
    }
    public ReportWindow(StorageUnit unit)
    {
        InitializeComponent();
        DataContext = this;
        _reportType = "UNIT";
        _unit = unit;

        UnitPanel.IsVisible = true;
        ListPanel.IsVisible = false;

        ReportTitle.Text = "КАРТКА СКЛАДСЬКОГО ОБЛІКУ";

        U_Name.Text = unit.Product.Name;
        U_Sku.Text = unit.Product.Sku;
        U_Qty.Text = $"{unit.Quantity} {unit.Product.Unit}";
        U_Batch.Text = unit.BatchNumber;
        U_Decl.Text = unit.Declaration.DeclarationNumber;
        U_Exp.Text = unit.Declaration.ExpirationDate.ToString("dd.MM.yyyy");
        U_Loc.Text = unit.Location.FullAddress;
        U_QrText.Text = unit.QrCodeHash;

        if (!string.IsNullOrEmpty(unit.QrCodeHash))
        {
            var qrBytes = GenerateQrCodeBits(unit.QrCodeHash);
            using var stream = new MemoryStream(qrBytes);
            U_QrImage.Source = new Avalonia.Media.Imaging.Bitmap(stream);
        }
    }
    public ReportWindow(Product product, List<StorageUnit> allStock)
    {
        InitializeComponent();
        DataContext = this;
        _reportType = "PROD";
        _product = product;
        _dataList = allStock.Where(u => u.ProductId == product.Id).ToList();

        UnitPanel.IsVisible = false;
        ListPanel.IsVisible = true;

        ReportTitle.Text = $"ЗВІТ ПО ТОВАРУ: {product.Sku}";
        InfoText.Text = $"Найменування: {product.Name}\nАртикул: {product.Sku}\nОд. вим: {product.Unit}";
        TotalText.Text = $"Всього на складі: {_dataList.Sum(x => x.Quantity)} {product.Unit}";

        ProductTable.IsVisible = true;
        ProductTable.ItemsSource = _dataList;
    }
    public ReportWindow(CustomsDeclaration decl, List<StorageUnit> allStock)
    {
        InitializeComponent();
        DataContext = this;
        _reportType = "DECL";
        _declaration = decl;
        _dataList = allStock.Where(u => u.DeclarationId == decl.Id).ToList();

        UnitPanel.IsVisible = false;
        ListPanel.IsVisible = true;

        ReportTitle.Text = "МИТНИЙ ЗВІТ (МД)";
        InfoText.Text = $"МД: {decl.DeclarationNumber}\nДата: {decl.EntryDate:d}\nТермін до:
{decl.ExpirationDate:d}";
        TotalText.Text = $"Всього позицій: {_dataList.Count}";

        DeclTable.IsVisible = true;
        DeclTable.ItemsSource = _dataList;
    }

    private byte[] GenerateQrCodeBits(string text)
```

```
    {
        using var qrGenerator = new QRCoder.QRCodeGenerator();
        var qrCodeData = qrGenerator.CreateQrCode(text, QRCoder.QRCodeGenerator.ECCLevel.Q);
        var qrCode = new QRCoder.PngByteQRCode(qrCodeData);
        return qrCode.GetGraphic(20);
    }

    private async void ExportPdf_Click(object sender, RoutedEventArgs e)
    {
        string fileName = _reportType switch
        {
            "UNIT" => $"Картка_{_unit?.Product.Sku}",
            "PROD" => $"Товар_{_product?.Sku}",
            "DECL" => $"МД_{_declaration?.DeclarationNumber.Replace("/", "-")}",
            _ => "Report"
        };

        var topLevel = TopLevel.GetTopLevel(this);
        var file = await topLevel.StorageProvider.SaveFilePickerAsync(new FilePickerSaveOptions
        {
            Title = "Зберегти PDF",
            DefaultExtension = "pdf",
            SuggestedFileName = fileName,
            FileTypeChoices = new[] { new FilePickerFileType("PDF Document") { Patterns = new[] { "*.pdf" } } }
        });

        if (file == null) return;

        try
        {
            var document = Document.Create(container =>
            {
                container.Page(page =>
                {
                    page.Size(PageSizes.A4);
                    page.Margin(1, Unit.Centimetre);
                    page.PageColor(Colors.White);
                    page.DefaultTextStyle(x => x.FontSize(11));

                    page.Header().Row(r =>
                    {
                        r.RelativeItem().Text("OLE-PRO WAREHOUSE").Bold();
                        r.RelativeItem().AlignRight().Text(DateTime.Now.ToString("g"));
                    });

                    page.Content().PaddingVertical(10).Column(col =>
                    {
                        col.Item().AlignCenter().Text(ReportTitle.Text).FontSize(14).Bold().Underline();
                        col.Item().Height(10);

                        if (_reportType == "UNIT" && _unit != null)
                        {
                            col.Item().Table(table =>
                            {
                                table.ColumnsDefinition(cd => { cd.ConstantColumn(150); cd.RelativeColumn(); });
                                void Row(string l, string v)
                                {
                                    table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(l).SemiBold();
                                    table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(v);
                                }

                                Row("Найменування", _unit.Product.Name ?? "-");
                                Row("Артикул", _unit.Product.Sku ?? "-");
                                Row("Кількість", $"{_unit.Quantity} {_unit.Product.Unit}");
                                Row("Партія", _unit.BatchNumber ?? "-");
                                Row("Декларація", _unit.Declaration.DeclarationNumber ?? "-");
                                Row("Місце зберігання", _unit.Location.FullAddress ?? "-");
                            });
                            col.Item().Height(20);
                            col.Item().Border(1).Padding(5).AlignCenter().Text($"QR: {_unit.QrCodeHash}");
                        }
                        else
                        {
                            col.Item().Border(1).Padding(5).Text(InfoText.Text);
                            col.Item().Height(20);
                            col.Item().Table(table =>
                            {
                                if (_reportType == "PROD")
                                {
                                    table.ColumnsDefinition(cd => { cd.RelativeColumn(2); cd.RelativeColumn(1);
cd.RelativeColumn(2); cd.RelativeColumn(1); });
                                    table.Header(h => { h.Cell().Text("МД").Bold(); h.Cell().Text("Кіл-
ть").Bold(); h.Cell().Text("Місце").Bold(); h.Cell().Text("Партія").Bold(); });
                                    foreach (var item in _dataList)
                                    {

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.Declaration.DeclarationNumber);

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.Quantity.ToString());
```

```
table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.Location.FullAddress);

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.BatchNumber);
                                    }
                                }
                                else if (_reportType == "DECL")
                                {
                                    table.ColumnsDefinition(cd => { cd.RelativeColumn(3); cd.RelativeColumn(2);
cd.RelativeColumn(1); cd.RelativeColumn(2); });
                                    table.Header(h => { h.Cell().Text("Товар").Bold();
h.Cell().Text("Артикул").Bold(); h.Cell().Text("Кіл-ть").Bold(); h.Cell().Text("Місце").Bold(); });
                                    foreach (var item in _dataList)
                                    {

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.Product.Name);

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.Product.Sku);

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text($"{item.Quantity} {item.Product.Unit}");

table.Cell().BorderBottom(1).BorderColor("CCC").Padding(5).Text(item.Location.FullAddress);
                                    }
                                }
                            });
                        }
                    });

                    page.Footer().Row(r =>
                    {
                        r.RelativeItem().Text("Комірник: _________________");
                        r.RelativeItem().AlignRight().Text("М.П.");
                    });
                });
            });

            using var stream = await file.OpenWriteAsync();
            var pdfData = document.GeneratePdf();
            await stream.WriteAsync(pdfData, 0, pdfData.Length);
            await new MessageWindow("✅ Файл збережено").ShowDialog(this);
        }
        catch (Exception ex)
        {
            await new MessageWindow(ex.Message).ShowDialog(this);
        }
    }

    private void Close_Click(object sender, RoutedEventArgs e) => Close();

    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК Л

Код вікна відображення користувачів

```
public partial class UsersWindow : Window
{
    public UsersWindow()
    {
        InitializeComponent();
        LoadUsers();

        this.AddHandler(PointerPressedEvent, (sender, e) =>
        {
            var source = e.Source as Visual;
            if (source == null) return;

            if (source.FindAncestorOfType<DataGridRow>() != null ||
                source.FindAncestorOfType<Button>() != null ||
                source.FindAncestorOfType<ScrollBar>() != null ||
                source.FindAncestorOfType<DataGridColumnHeader>() != null)
            {
                return;
            }

            var grid = this.FindControl<DataGrid>("UsersGrid");
            if (grid != null)
            {
                grid.SelectedItem = null;
            }

        }, RoutingStrategies.Tunnel);
    }

    private void LoadUsers()
    {
        using (var db = new DatabaseContext())
        {
            var users = db.Users.ToList();

            int counter = 1;
            foreach (var user in users)
            {
                user.DisplayNum = counter++;
            }

            var grid = this.FindControl<DataGrid>("UsersGrid");
            grid.ItemsSource = users;
        }
    }

    private async void AddUser_Click(object sender, RoutedEventArgs e)
    {
        var addWin = new AddUserWindow();
        await addWin.ShowDialog(this);

        LoadUsers();
    }

    private async void DeleteUser_Click(object sender, RoutedEventArgs e)
    {
        var grid = this.FindControl<DataGrid>("UsersGrid");
        var selectedUser = grid.SelectedItem as User;

        if (selectedUser == null)
        {
            await new MessageWindow("⚠ Оберіть користувача зі списку!").ShowDialog(this);
            return;
        }

        if (selectedUser.Username.ToLower() == "admin")
        {
            await new MessageWindow("⛔ Видаляти головного адміністратора заборонено!").ShowDialog(this);
            return;
        }

        var confirmWindow = new ConfirmWindow();
        await confirmWindow.ShowDialog(this);

        if (!confirmWindow.IsConfirmed) return;

        try
        {
            using (var db = new DatabaseContext())
            {
```

```
                var userToDelete = db.Users.Find(selectedUser.Id);

                if (userToDelete != null)
                {
                    db.Users.Remove(userToDelete);
                    db.SaveChanges();
                }
            }

            LoadUsers();
            await new MessageWindow("✅ Користувача видалено.").ShowDialog(this);
        }
        catch (Exception ex)
        {
            await new MessageWindow($"❌ Помилка при видаленні: {ex.Message}").ShowDialog(this);
        }
    }
    private void Close_Click(object sender, RoutedEventArgs e) => Close();
    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }
}
```

ДОДАТОК М

Код вікна додавання нового користувача

```
public partial class AddUserWindow : Window
{
    public AddUserWindow()
    {
        InitializeComponent();
    }

    private void TitleBar_PointerPressed(object sender, PointerPressedEventArgs e)
    {
        if (e.GetCurrentPoint(this).Properties.IsLeftButtonPressed)
        {
            BeginMoveDrag(e);
        }
    }

    private void OnSaveClick(object sender, RoutedEventArgs e)
    {
        string username = UsernameBox.Text;
        string password = PasswordBox.Text;
        string role = RoleBox.Text;

        if (string.IsNullOrWhiteSpace(username) || string.IsNullOrWhiteSpace(password))
        {
            return;
        }

        try
        {
            using (var db = new DatabaseContext())
            {
                var newUser = new User
                {
                    Username = username,
                    Password = password,
                    Role = string.IsNullOrWhiteSpace(role) ? "User" : role
                };

                db.Users.Add(newUser);
                db.SaveChanges();
            }
            Close();
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine("Помилка збереження: " + ex.Message);
        }
    }

    private void Cancel_Click(object sender, RoutedEventArgs e)
    {
        Close();
    }
}
```

ДОДАТОК Н

Демонстраційний матеріал

№ документа	Позначення	Найменування	Додаткові відомості
		Текстові документи	
1	ГЮИК. 401456.003 ПЗ	Пояснювальна записка	А4, 107 с.
		Додаткові матеріали	
2		Додаток А	А4, 8 с.
3		Додаток Б	А4, 1 с.
4		Додаток В	А4, 7 с.
5		Додаток Г	А4, 1 с.
6		Додаток Д	А4, 1 с.
7		Додаток Е	А4, 1 с.
8		Додаток Ж	А4, 1 с.
9		Додаток И	А4, 1 с.
10		Додаток К	А4, 3 с.
11		Додаток Л	А4, 2 с.
12		Додаток М	А4, 1 с.
13		Додаток Н	А4, 13 с.

					ГЮИК. 401456.003 ВД			
Змін.	Арк.	Номер докум.	Підпис	Дата				
Розроб.		Кривенко			Розроблення програмного модуля комп'ютеризованої системи для автоматизованого визначення вантажів на бондових складах	Літера	Аркуш	Аркушів
Перевір.		Чала				Н	1	1
						Кафедра КІТАР ХНУРЕ		
Н.контр.		Демська						
Затв.		Невлюдов						