НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ
І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

УДК 004.9:631.559:633

«ПОГОДЖЕНО»	**«ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ»**
Декан факультету інформаційних технологій	Завідувач кафедри комп'ютерних наук
Глазунова О.Г., д.п.н., професор	Голуб Б.Л., к.т.н., доцент
_____	_____
____ _____ 202_р.	**____ _____ 202_ р.**

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему _____ Система обліку та аналізу використання запасів сільськогосподарського підприємства_____

Спеціальність ___121_____Інженерія програмного забезпечення_____

_____
(код і назва)

Освітня програма __Програмне забезпечення інформаційних систем___

_____
(назва)

Орієнтація освітньої програми _освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Гарант освітньої програми

_____ _____ _____
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Керівник магістерської кваліфікаційної роботи

_____ _____ _____
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Виконав _____ **_____Липинець Я.В._____**
(підпис) (ПІБ студента)

КИЇВ-2023

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет (ННІ)
_____

ЗАТВЕРДЖУЮ
Завідувач кафедри_____
_____
_____ _____ _____
(науковий ступінь, вчене звання) (підпис) (ПІБ)
“____” _____20_____року

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ СТУДЕНТУ

_____
(прізвище, ім’я, по батькові)

Спеціальність
_____
(код і назва)

Освітня програма
_____
(назва)

Орієнтація освітньої програми_____
(освітньо-професійна або освітньо-наукова)

Тема магістерської кваліфікаційної роботи_____
_____
_____

затверджена наказом ректора НУБіП України від “____”_____20___р. №_____

Термін подання завершеної роботи на кафедру _____
(рік, місяць, число)

Вихідні дані до магістерської кваліфікаційної роботи _____
_____
_____

Перелік питань, що підлягають дослідженню:
1. _____
2. _____
3. _____

Перелік графічного матеріалу (за потреби) _____
_____

Дата видачі завдання “_____”_____20___ р.

Керівник магістерської кваліфікаційної роботи _____ _____
(підпис) (прізвище та ініціали)

Завдання прийняв до виконання _____ _____
(підпис) (прізвище та ініціали студента)

ЗМІСТ

ВСТУП

В сучасних умовах аграрна галузь є однією з ключових та забезпечує як продовольчу безпеку для населення так і можливість розвитку на зовнішньому ринку економіки України. Ускладненість логістичних процесів та інколи неможливість використання традиційних шляхів перевезення продукції вимагає підвищення ефективності використання наявних можливостей зберігання та пошуку нових методів управління складськими запасами. Це в першу чергу знижує потенційні витрати через неефективність зберігання запасів, а в другу чергу дозволяє підвищити прибутковість та життєздатність аграрного підприємства загалом.

Точність та достовірність результатів обліку запасів підприємства дуже часто залежить від людського фактору – а саме наскільки точно оператор складу або комірник вносить дані про поповнення та використання запасів. При цьому, навіть ненавмисне спотворення даних може спричиняти помилки та складність точного оцінювання поточного стану складу. Так, наприклад, при поставці готової продукції або сировини на склад єдиним контролюючим органом є комірник, який вносить інформацію в базу даних чи таблицю на власний розсуд, що може призводити до помилок чи ненавмисного внесення недостовірних даних.

Використання програмних засобів, що дозволяє б в оперативному режимі здійснювати облік та визначати ефективність використання складських запасів підприємства на сьогоднішній день використовується не достатньо широко або лише для великих підприємств з корпоративними інформаційними системами. В той же час згідно статистичних даних 90% відсотків аграрних підприємств України складають саме малі підприємства, для яких більш ефективним та економічно доцільним є використання простих програмних рішень для автоматизації обліку та аналізу використання запасів.

Таким чином, актуальність розробки системи обліку та аналізу використання запасів сільськогосподарського підприємства є актуальною та затребуваною для як для малих так і для великих підприємств.

У даній роботі здійснюється постановка задачі та розробка системи обліку та аналізу використання запасів сільськогосподарського підприємства на основі телеграм-боту з підключенням бази даних. Фактично телеграм бот буде використовуватись як інтерфейс користувача який дозволяє отримувати, зберігати в базу даних, та при необхідності надавати інформацію про запаси на складі.

Вибір засобу отримання даних від операторів або працівників обумовлюється тим, що телеграм є розповсюдженим менеджером який сьогодні встановлено майже на кожному смартфоні. При цьому, використання коду користувача телеграму може бути використано для автентифікації користувачів та їх розподілення на різні ролі. Таким чином, впровадження та експлуатація такої системи супроводжується лише витратами на розробку та навчання персоналу, без використання особливих додаткових програмно-апаратних засобів.

У першому розділі роботи розглядається специфіка процесу обліку та аналізу використання запасів сільськогосподарського підприємства та існуючі системи, які наразі використовуються на вітчизняному та на світовому ринках для невеликих аграрних підприємств.

У другому розділі розглядаються технічні аспекти реалізації системи обліку та аналізу використання запасів сільськогосподарського підприємства та обґрунтування використання інструментарію.

Третій розділ роботи присвячено процесу розробки системи обліку та аналізу використання запасів сільськогосподарського підприємства.

У четвертому розділі здійснено тестування розробленої системи та проведено аналіз потенційного використання системи і майбутнього розвитку з урахуванням підвищення складності системи та врахування додаткових параметрів обліку та аналізу.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ СИСТЕМИ ОБЛІКУ ТА АНАЛІЗУ ВИКОРИСТАННЯ ЗАПАСІВ СІЛЬСЬКОГОСПОДАРСЬКОГО ПІДПРИЄМСТВА

1.1. Методичні основи обліку та аналізу використання запасів сільськогосподарського підприємства

Оцінка запасів на підприємствах здійснюється згідно з витратами на їх придбання або отримання а також при їх використанні. При цьому основним регламентуючим ці процеси документом є Закон України «Про бухгалтерський облік та фінансову звітність» від 16.07.1999 № XIV (зі змінами і доповненнями) [1]. Згідно з цим законом здійснюється ведення та управління процесом обліку підприємств в Україні, в тому числі облік запасів сільськогосподарських підприємств.

Оцінка запасів підприємства є важливою частиною бухгалтерського та фінансового обліку оскільки запаси напряму впливають на кінцеві економічні показники ефективності діяльності організації. Великі коливання ціни на запаси готової продукції та сировини в аграрній сфері призводять до того, що іноді за рік ціна може змінюватись більше ніж на 100% за календарний рік на окремі види продукції. Це пов’язано з коливаннями цін на світовому ринку, складною політичною ситуацією та специфікою ринку сільськогосподарської продукції. Тому своєчасна та точна оцінка запасів необхідна для проведення аналізу ефективності діяльності організації. Як правило, вартість запасів оцінюється за собівартістю при їх внутрішньому використанні, тобто у якості сировини для виготовлення готової продукції. Якщо ж запаси будуть використовуватись для реалізації без переробки, то оцінку здійснюють за ринковою ціною. В цьому випадку дата постачання запасу дозволяє оцінити точну вартість ресурсу з урахуванням поточної ціни на ринку на певний момент часу [2].

Точна оцінка вартості запасів дозволяє відповідно точно оцінити собівартість кінцевої продукції та визначити рентабельність і прибутковість основної діяльності сільськогосподарського підприємства.

Окрім часу здійснення складських операцій та моменту фактичну внесення запасів на склад важливу роль в управлінні підприємством та оцінці ефективності основної діяльності відіграє оцінка стану запасів – тобто фактична кількість різних видів продукції та сировини на складі [3].

Процес обліку підприємства регламентується П(С)БО 9 «Запаси». Окрім здійснення бухгалтерського обліку, який є стандартизованим та чітко формалізованим процесом і регламентується законодавством, підприємство в тому чи іншому вигляді здійснює облік запасів на оперативному рівні. При цьому, процес здійснення обліку можна представити у вигляді наступної схеми (Рис. 1.1.)

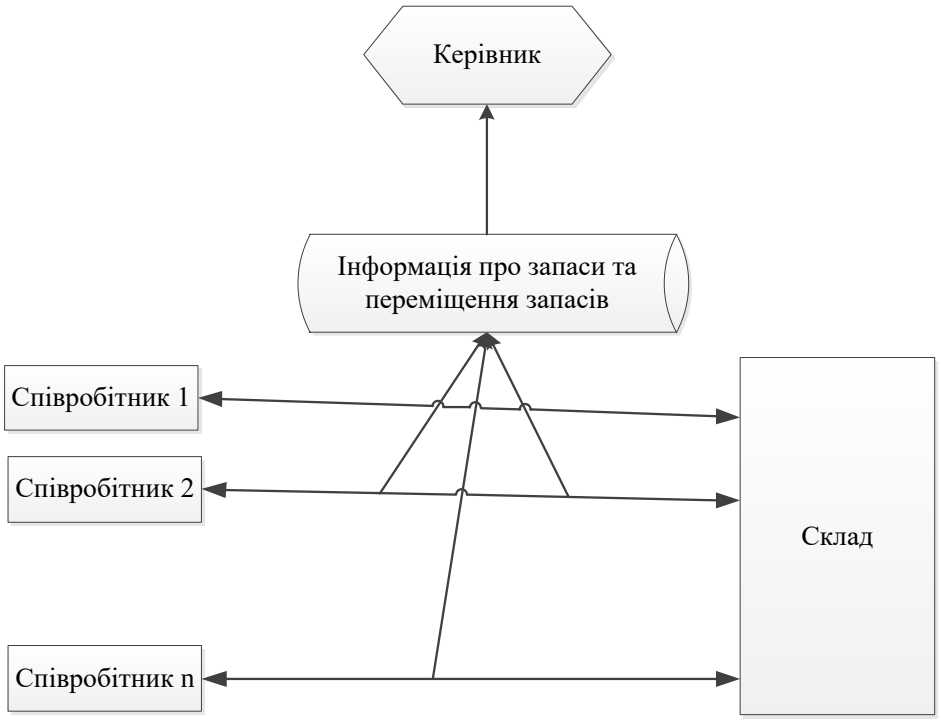


Рис. 1.1. Процес здійснення оперативного обліку запасів

Співробітники підприємства можуть здійснювати як постачання на склад так і отримання запасів зі складу. Сторонні контрагенти також можуть

взаємодіяти зі складом, але лише через співробітників підприємства – наприклад через комірника. Тобто, стороння персона без дозволу не має права взаємодіяти безпосередньо зі складом, тому розглядати взаємодію з іншими організаціями та контрагентами в рамках задачі оперативного обліку запасів в даній роботі не будемо, звузивши коло користувачів лише до внутрішніх співробітників [4].

Складський облік представляє собою кількісний облік оперативного типу, за який відповідає комірник, тобто матеріально відповідальна особа підприємства. В деяких випадках роль комірника може виконувати власник підприємства – наприклад така ситуація є поширеною для малих підприємств. Дані такого складського обліку вносяться у книгу складського обліку, яка представляє собою відомість або двовимірну таблицю з наступними полями:

- дата складської операції,
- тип складської операції: внесення на склад або отримання зі складу,
- особа, яка здійснює складську операцію,
- особа, яка перевіряє та контролює складську операцію,
- вид (назва, артикул) продукції, запасів, товару, який вноситься на склад або отримується зі складу,
- кількість продукції, запасів або товару.

Комірник або інша матеріально відповідальна особа перевіряє кількість та якість запасів. Так, наприклад, на одному складі можуть зберігатись запаси різних категорій або сортів. Кожен окремий вид продукції або запасів як правило має шифр або номер згідно внутрішньої документації підприємства, що забезпечує єдині позначення у всій інформаційній системі підприємства.

Кожна операція з переміщення запасів або товарів на склад та зі складу супроводжується визначенням залишку відповідного запасу/товару на складі, або залишок підраховується в кінці дня. Використання автоматизованих систем обліку замість паперових відомостей дозволяє отримувати інформацію про залишок на складі одразу ж після внесення відповідних даних про операцію, а не в кінці звітного дня або тижня. Це дозволяє здійснювати без додаткових

затримок отримувати оперативну інформацію про стан складу та ефективність використання існуючих складських приміщень. Так, наприклад, деякі складські приміщення можуть бути пустими та не використовуватись, а деякі навпаки будуть переповненні що не дозволить здійснювати внесення запасів на склад.

Окрім здійснення поточного контролю та внесення даних у відомості про кожну складську операцію, комірник або матеріально відповідальна особа складає підсумкові звіти за певні періоди – тиждень, місяць, квартал та рік. Відповідні звіти використовуються при здійсненні бухгалтерського та фінансового обліку у якості первинних документів для аналізу ефективності діяльності підприємства та визначення основних показників фінансово-економічної діяльності організації [5].

Комірник вносить записи про складські операції та здійснює нумерацію починаючи з певного звітного періоду – місяця, кварталу, року. Кожен запис містить окрім операції та кількості переміщених запасів дату, номер складу якщо їх декілька. Також може фіксуватись залишок продукції чи запасів на складі після завершення операції.

З певною періодичність, наприклад раз на місяць чи в кінці кварталу матеріально відповідальні особи віддають журнали про переміщення складських запасів бухгалтеру для внесення цих відомостей у систему бухгалтерського обліку.

Бухгалтерія підприємства як правило розподіляє різні ресурси за групами та позначає номенклатурними номерами, які також можуть міститись і в відомостях про рух складських запасів.

Бухгалтер може виконувати функції контролера або аудитора та здійснювати перевірку складу на точність ведення обліку у певні дня вибірково окрім перевірки поданих журналів.

1.2. Автоматизація процесів обліку та аналізу використання запасів сільськогосподарського підприємства

Використання паперового документообігу або використання звичайних електронних таблиць для здійснення оперативного обліку та зберігання інформації про складські операції на сьогоднішній день зустрічається у невеликих підприємствах. Це пов’язано з високою вартістю впровадження та використання корпоративних інформаційних систем в межах невеликого підприємства, яке не потребує великої кількості різних модулів чи додатків, а система обліку не вимагає складних рішень. При цьому, відсутність єдиної інформаційної системи та ведення обліку у вигляді розрізнених таблиць чи документів має наступні недоліки:

- висока ймовірність здійснення помилки через людський фактор,
- можливість внесення недостовірних даних або коригування існуючих даних,
- відсутність інформації про стан складських запасів для оперативного аналізу та моніторингу керівництвом,
- високі витрати часу на оформлення підсумкових документів за звітні періоди для бухгалтерського обліку.

Ручне внесення інформації про складську операцію комірником супроводжується надалі також внесенням інформації вручну у звітні документи та у звіти для керівництва. Як результат, дублювання операцій та дублювання інформації може призводити до помилок, як навмисних так і ненавмисних. Так, наприклад, комірник може пропустити один із записів які знаходяться у загальній відомості при підсумуванні і формуванні звіту. Знайти таку помилку майже неможливо до моменту ревізії та перевірки всього складу. При цьому, специфіка складських запасі сільськогосподарського підприємства не дозволяє здійснити повну ревізію та отримання абсолютно точних результатів. Наприклад, для того щоб визначити залишки зерна у складі необхідно його повністю з цього складу перемістити та зважити. Інші методи можуть давати

лише приблизну оцінку з високою похибкою, через те що зерно різних сортів та при різній вологості може відповідно мати різну щільність та різну вагу при тому самому об’ємі. Таким чином, отримання точної та достовірної інформації про складські операції дозволяє підвищити точність обліку запасів та уникнути необхідності повної ревізії складу [6].

Використання інформаційної системи з захистом від несанкціонованого доступу та з реєстрацією всіх дій користувачів дозволяє також зменшити кількість потенційних втручань сторонніх осіб або навмисного змінення інформації що зберігається у складських відомостях. Так, наприклад, використання розрахункових полів у базі даних дозволяє завжди отримувати точну інформацію на відміну від введених вручну підсумкових значень, які були розраховані у сторонньому програмного забезпеченні. Запис кожної операції, яка була здійснена на складі, не дозволяє додавати операції зі зміненою датою заднім числом, або коригувати вже внесені показники.

Для керівника організації чи особи що приймає рішення в першу чергу важливо вчасно отримувати точну та достовірну інформацію про стан складських запасів. Це необхідно для своєчасного здійснення торгівельних операцій та ефективного планування збутової діяльності підприємства вцілому. Так, наприклад, якщо не відома точна кількість запасів на даний момент на складі, а лише приблизна кількість – неможливо точно визначити чи достатньо поточної кількості для здійснення основної діяльності підприємства. Як результат, виникає необхідність створення резервних запасів, які в свою чергу призводять до виведення з обороту та задіяння на складі додаткових ресурсів. Отримання звітів один раз на місяць або навіть один раз на тиждень інколи не є достатньо ефективним. Під час збирання врожаю спостерігається висока інтенсивність поповнення складу і як результат лише за один день може суттєво змінитись показник заповнення складу - наприклад до 10% від загальної місткості складу.

В кінці звітного періоду комірник як правило формує підсумкову документацію про стан складських запасів та підсумовує всі операції які

відбувались за цей період. Так, наприклад, порівнюється кількість запасів що було поставлено на склад та які було отримано зі складу. В результаті формується залишок, який додається до минулого показника. Таким чином формується загальна кількість складських запасів. При цьому використання електронних таблиць або інших засобів реєстрації потребує витрат часу та не завжди дозволяє отримати всі необхідні дані досить швидко та у потрібній для бухгалтерського обліку формі.

Здійснення аналізу руху складських запасів та здійснення складських операцій у розрізі різних співробітників, видів продукції, часу або інших параметрів може проводитись лише додатковими засобами, наприклад з використанням вбудованих інструментів групування та підведення проміжних підсумків електронних таблиць.

Таким чином, використання автоматизованої системи обліку та аналізу складських запасів дозволяє вирішувати наступні задачі:

- зниження імовірності виникнення помилок у відомостях та звітній діяльності через людський фактор,

- уникнення можливості внесення недостовірних даних або коригування існуючих даних,

- своєчасне надання інформації про стан складських запасів для оперативного аналізу та моніторингу керівництвом,

- зниження витрат часу на оформлення підсумкових документів за звітні періоди для бухгалтерського обліку,

- підвищення ефективності використання існуючих складських запасів підприємства,

- зменшення резервів складських запасів.

1.3. Огляд існуючих програмних рішень для обліку та аналізу використання запасів сільськогосподарського підприємства

На сьогоднішній день на вітчизняному ринку спостерігається певний дефіцит спеціалізованого програмного забезпечення для обліку роботи складу через те що користувачі довго орієнтувались на використання російських продуктів, що займали суттєву частину WMS (систем управління складом). Це призвело до того, що на поточний момент деякі підприємства або використовують старі версії програмного забезпечення без підтримки розробника, або переходять на інші платформи [7].

При цьому системи управління складом вирішують ряд задач окрім виключно обліку запасів:

- розміщення запасів,
- моніторинг переміщення запасів,
- розподіл працівників складу,
- управління перевезеннями та логістикою,
- розробка планів закупівлі запасів,
- підтримка сканерів штрихкодів,
- забезпечення інформаційної бази для списання основних засобів виробництва,
- можливість формування та друку звітних документів за заданими шаблонами,
- синхронізація з торгівельною вітриною інтернет-магазину або іншими ресурсами,
- експортування звітів.

Використання комерційних систем обліку роботу складу є поширеною та позитивною практикою для середніх та великих підприємств та, безумовно, підвищує ефективність роботи організації. Але, при цьому, не завжди можуть враховуватись специфічні вимоги предметної області. Так, наприклад, при

зберіганні деякі ресурси, такі як зерно, можуть змінювати свою вагу при зміні вологості повітря, та при цьому можуть вимагати внесення певних виправлень в систему обліку.

Крім того, системи складського обліку як правило реалізовані у вигляді десктопних або веб-додатків, з підтримкою мобільних версій. Для вирішення простих задач обліку внесення на склад використання таких додатків може бути занадто складним.

Архітектура системи а також зберіганні інформації у внутрішній базі даних як правило не є відкритою для кінцевого користувача інформацією, що також іноді може створювати труднощі в реалізації. Тобто, будь-яке внесення змін в систему може здійснюватися лише розробником системи.

Для сучасного бізнесу особливо важливим параметром функціонування системи обліку є точність – тобто фактична кількість запасів на складі обов'язково має відповідати кількості вказаній в базі даних. Інакше можуть виникати ситуації коли потрібно використати ресурс, якого фактично немає на складі, і від цього залежить діяльність виробництва чи постачання товару контрагентам. Крім того, товари та ресурсі на складі мають бути враховані таким чином, щоб в разі потреби можна було легко знаходити потрібні позиції та легко формувати звітну документацію. Тобто, навіть людина яка не розбирається в технічних нюансах роботи програмного забезпечення має розібратись в інтерфейсі додатку та виконати необхідні задачі. Тому при виборі системи обліку в першу чергу потрібно враховувати зручність інтерфейсу а не наявність великої кількості функцій та сучасних елементів дизайну в оформленні.

Окрім обліку наявності поточної кількості ресурсів на складі існуючі системи також часто дозволяють враховувати потреби в ресурсах та формувати план закупівель.

Серед найбільш поширених вітчизняних систем обліку складу можна виділити Dilovod, що забезпечує наступний функціонал:

- ведення обліку за різними категоріями та видами товарів,

- автоматизація інвентаризації та списання основних засобів,
- підтримка сканерів штрихкодів,
- контроль роботи з контрагентами,
- наявність великої кількості шаблонів документів,
- формування звітної документації по аналізу складських запасів,
- ведення кількох складів,
- інтеграція із службами доставки (Нова пошта, Делівері і т.д.),
- інтеграція з інтренет банкінгом (Монобанк, Приват24 і т.д.),
- інтеграція з маркетплейсами (Розетка, Пром.юа і т.д.).

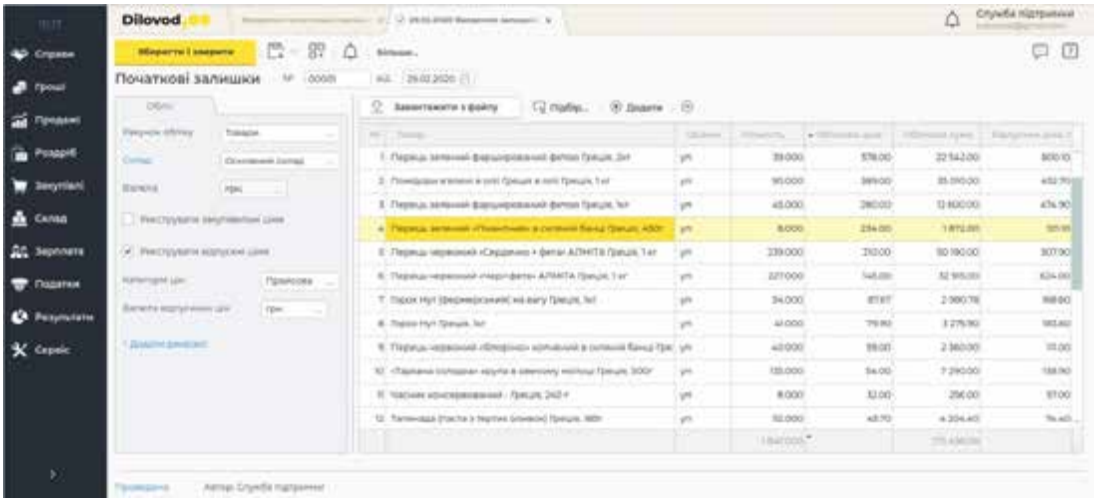


Рис. 1.2. Інтерфейс системи обліку Dilovod

Також досить популярною системою є УкрСклад, що надає користувачам наступні можливості:

- управління рузом товарів,
- ведення документообігу,
- контроль залишків на складі,
- контроль розрахунків з контрагентами,
- підтримка сканерів штрих-кодів,
- фінансовий аналіз,
- експорт звітності,

- інтеграція з сервісами розсилок.

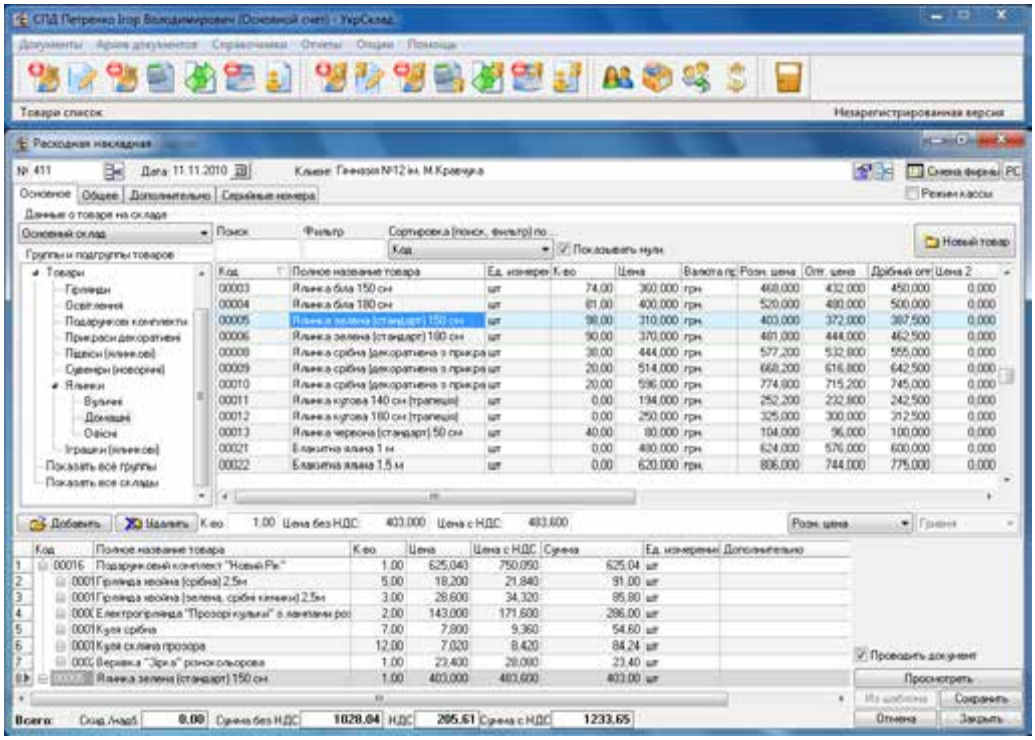


Рис. 1.2. Інтерфейс системи обліку УкрСклад

Також слід відмітити універсальну crm систему KeyCRM, що включає окремий модуль для роботи зі складами:

- формування бази товарів,
- можливість групування інформації про залишки різних складів чи магазинів,
- переміщення товарів між складами,
- можливість резервування товару,
- можливість списанні матеріальних цінностей,
- імпорт та експорт документації,
- проведення інвентаризації,
- підтримка сканерів штрих-кодів,

- управління виробничими процесами.

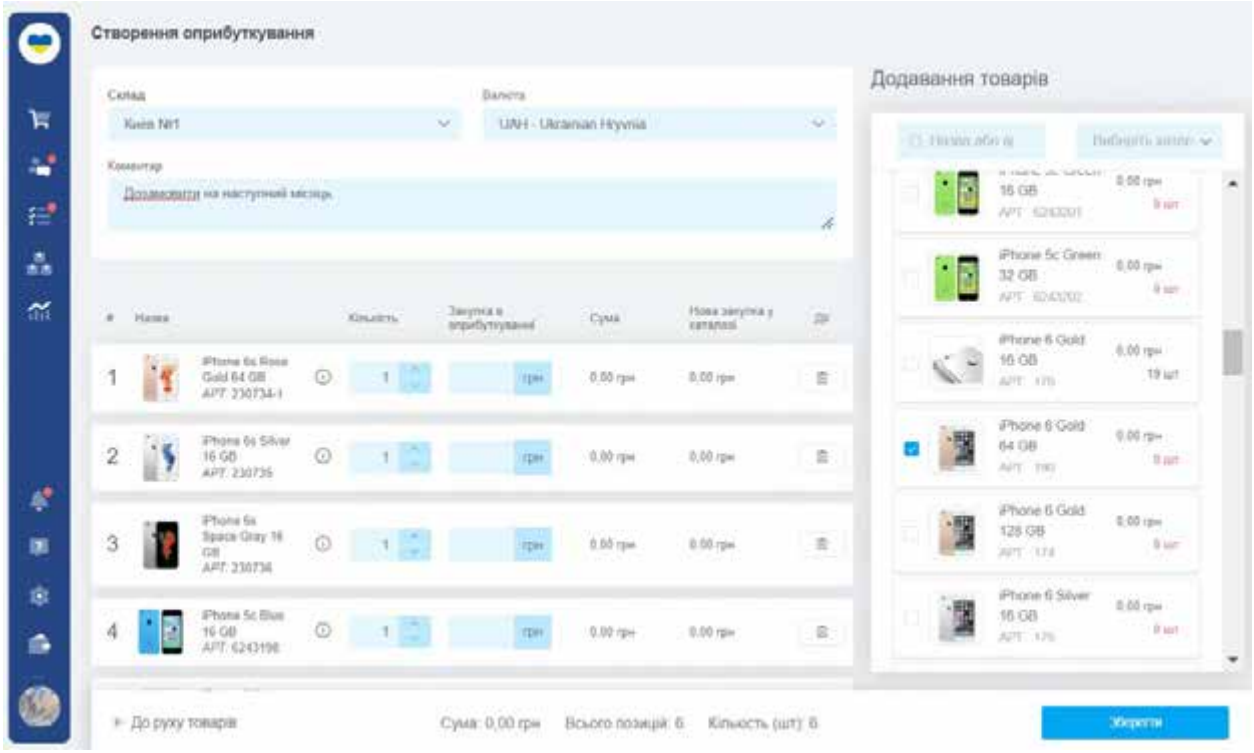


Рис. 1.2. Інтерфейс системи обліку KeyCRM

Автоматизація складського обліку з використанням відповідних систем дозволяє знизити витрати часу на проведення рутинних операцій з обліку, інвентаризації, контролю взаємодії з контрагентами. Використання таких система є вимушеною необхідністю в сучасному висококонкурентному середовищі.

Основним недоліком перелічених систем є їх великий розмір та орієнтація на великі підприємства, що ускладнює можливості їх використання на невеликих агрофірмах з одним складом та невеликим асортиментом готової продукції що потребує обліку. Крім того, перелічені системи є платними та працюють за підпискою.

РОЗДІЛ 2. ВИБІР ІНСТРУМЕНТАРІЮ ДЛЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ ТА АНАЛІЗУ ВИКОРИСТАННЯ ЗАПАСІВ СІЛЬСЬКОГОСПОДАРСЬКОГО ПІДПРИЄМСТВА

2.1. Вибір інструменту для отримання інформації від користувачів

При виборі інструменту для отримання інформації від користувачів в першу чергу потрібно з’ясувати вимоги, що висуваються до системи обліку та аналізу використання запасів сільськогосподарського підприємства, які запропоновані одним з працівників підприємства:

- можливість внесення даних у систему в будь-який момент з будь-якого пристрою, без використання додаткового програмного та апаратного забезпечення,

- можливість використання мобільного пристрою або персонального комп’ютера для користування системою,

- відсутність доступу до системи сторонніх користувачів,

- неможливість коригування записів про складську операцію оператором чи комірником, якщо ця операція вже записана в систему,

- відображення в системі дати здійснення складської операції,

- окреме зберігання інформації про внесення запасу на склад та отримання запасу зі складу.

Згідно з вимогами від одного з потенційних користувачів систему пропонується використовувати для її реалізації телеграм-бот.

Телеграм вже давно перестав бути звичайним месенджером, його функціонал постійно розширюється. На сьогоднішній день існує велика кількість телеграм ботів, які використовуються як для розваг – прослуховування музики та перегляд відео, публікацій новин, так і для виконання професійних задач – консультування клієнтів та користувачів банків чи комунальних підприємств, отримання інформації від користувачів інформаційної системи [8].

Телеграм бот або чат-бот представляє обліковий запис особливого типу, який може самостійно відповідати на повідомлення користувачів та отримувати від користувачів повідомлення. Телеграм бот дозволяє автоматизувати рутинні операції комунікації з користувачами, які не вимагають участі людини та можуть бути автоматизовані з використання мов програмування. Для використання телеграм бота створюється обліковий запис типу bot та відповідний API до нього, який дозволить керувати телеграм ботом через програмне середовище.

Телеграм боти можуть автоматично приймати та відправляти повідомлення, а також здійснювати обробку прийнятих повідомлень – наприклад зберігати в базу даних для подальшої обробки, передавати іншим користувачам, надавати інформацію згідно запиту. Телеграм боти розробляються програмістами та запускаються на сервері.

Сьогодні більшість компаній використовують телеграм боти для різних задач – для консультування клієнтів або для розширення функціоналу власних інформаційних систем. Так, наприклад такий бот як @privatbankbot – розроблений для того щоб клієнти «Приватбанку» могли переказувати кошти, перевіряти рахунок, здійснювати конвертацію валют. Тобто, фактично, виконує деякий функціонал поширеного та відомого мобільного додатку Приват24.

Використання телеграм-ботів для сучасного бізнесу відкриває нові можливості охоплення більш широкої аудиторії, так як використання телеграму для деяких користувачів зручніше і вони не хочуть використовувати мобільні або веб застосунки. Розгортання та впровадження телеграм-боту не вимагає великих витрат, так як для роботи з телеграм ботом потрібно лише зайти в чат та почати його використовувати [9].

Основні переваги використання телеграм-бота у якості системи обліку та аналізу використання запасів сільськогосподарського підприємства:

1. Можливість використання телеграм-бота на будь-якому сучасному смартфоні чи комп’ютері підключеному до мережі інтернет. На сьогоднішній день використання телеграм-ботів є розповсюдженою практикою серед малих

та великих підприємств та широко використовується для підтримки та доповнення існуючих інформаційній систем а також у якості систем для консультаційних послуг.

2. Цілодобовий доступ до телеграм боту. При розміщенні на сервері телеграм бот дозволяє обробляти повідомлення від користувачів та надавати їм відповіді абсолютно в автономному режимі без участі адміністратора чи розробника системи.

3. Кожен користувач месенджеру телеграм має унікальний логін та код користувача, які при необхідності можуть використовуватись для фільтрації та автентифікації користувачів. Тобто, навіть знаючи ім’я телеграм боту отримати доступ до нього та працювати з ним зможуть лише ті користувачі, які мають право його використовувати, наприклад комірник, працівники складу чи керівник підприємства.

4. Мінімальні витрати для розгортання та використання системи. Для розгортання серверу з телеграм-ботом при невеликій кількості користувачів можливе використання звичайного персонального комп’ютера підключеного до мережі інтернет.

5. Можливість використання різних середовищ розробки. На сьогоднішній день існує велика кількість бібліотек для різних мов програмування, що дозволяє розробляти систему на базі телеграм-бота з використанням великої кількості різноманітних інструментів.

6. Можливість підключення бази даних. Телеграм бот може використовувати СУБД для зберігання інформації яка поступає від користувачів а також надавати кінцеву інформацію керівнику або відповідальній особі.

7. Користувач телеграм-боту використовує лише доступний для нього інтерфейс та фізично не зможе видалити вже внесений запис, якщо це не передбачено функціоналом бота. Тобто, обробляються лише ті запити по роботі з базою даних, які були запрограмовані в цьому боті. Ніяким іншим чином отримати доступ до файлів бази даних отримати неможливо, так як він може

зберігатись на сервері а не на локальному комп’ютері користувача. Крім того використання бази даних дозволяє здійснювати експорт даних до інших інформаційних систем – наприклад існуючої системи бухгалтерського обліку яка вже використовується на підприємстві.

8. Швидкість взаємодії. Більшість користувачів телеграм-ботів позитивно відгукуються про їх використання насамперед через те, що можна миттєво отримати потрібну відповідь та не чекати поки звільниться оператор [10].

Незважаючи на те, що телеграм-боти все ширше використовуються в різних галузях та сферах бізнесу, вони мають і деякі недоліки або слабкі сторони у порівнянні і іншими системами:

- необхідність використання додатку телеграм та залежність від роботи серверу телеграм,

- функціональність телеграм боту залежить від інтерфейсу та може суттєво обмежуватись через примітивні інструменти керування,

- обмеженість користувацького інтерфейсу що обумовлена достатньо простими елементами дизайну,

- рівень безпеки передачі повідомлень частково залежить від серверів телеграм.

Таким чином, телеграм-бот представляє собою достатньо сучасне та поширене рішення, яке може бути використане для оперативного обліку та для підтримки існуючої інформаційної системи бухгалтерського обліку, але не може повноцінно замінити корпоративну інформаційну систему через обмеженість інтерфейсу та елементів керування. Тим не менш, використання такого рішення дозволить спростити роботу з отримання інформації від користувачів та зберігати отримані від користувачів відомості в базу даних, а також при необхідності представляти керівнику чи особі що приймає рішення відповідні оброблені дані про використання чи поповнення складських запасів у розрізі певних періодів або залежно від користувача що здійснював складську операцію.

2.2. Вибір інструменту для зберігання та обробки інформації

Телеграм бот може зберігатись та бути запущеним на будь-якому сервері підключеному до мережі інтернет, для невеликих проектів на етапі розробки для цього може використовуватись звичайний комп'ютер. При бот може відповідати на певні запитання із власної бази відповідей або за певним алгоритмом, а може також зберігати отриману від користувачів інформацію. При розробці телеграм-бота який буде використовуватись у якості системи обліку та аналізу складських запасів обов'язковою задачею є зберігання інформації про складські операції з метою використання цих даних для подальшого аналізу а також забезпечення обліку [11].

Для зберігання отриманих від користувачів даних доцільно використовувати окрему базу даних або зберігання даних в окремі файли. Використання баз даних надає переваги у порівнянні з окремими файлами так як у майбутньому простіше організувати процес аналізу внесених даних. Крім того, СКБД (системи керування базами даних) мають вбудовані інструменти що спрощують роботу з даними та мінімізують кількість потенційних помилок. Так, наприклад, можна встановити поле як унікальне, і в результаті неможливо буде створити ще один запис із повторюваним значенням – користувач може отримати повідомлення про помилку або такий запис буде проігноровано залежно від налаштувань програми [12].

Крім того, СКБД можуть використовуватись надалі окремо від основного додатку, телеграм боту – база даних зберігається в окремому файлі або може фізично розташовуватись на окремому сервері. Телеграм бот може підключатись до віддаленого серверу для отримання або запису інформації в таку базу даних. Надалі до цієї бази даних може отримати доступ керівник або аналітик та за допомогою СКБД здійснити необхідні запити.

У якості мови програмування для на якій буде написано додаток телеграм бот обрано мову Python через її поширеність, легкість використання та велику кількість бібліотек що забезпечують широкий функціонал та можливості

виконання різних задач. Мова програмування Python має вбудовану бібліотеку для роботи с базою даних SQLite, що спрощує процес розробки та забезпечує стабільну роботу додатку без необхідності використання бібліотек від сторонніх організацій.

SQLite представляє собою полегшену систему керування базами даних реляційного типу. Вона розроблена у вигляді бібліотека в якій реалізовано стандарт SQL. SQLite може використовуватись без обмежень так як початковий код поширюється безкоштовно. Фінансова підтримка розробників здійснюється групою компаній до яких входять такі відомі корпорації як Adobe, Oracle.

Від аналогічних продуктів SQLite відрізняється тим, що тут не використовується клієнт-серверний підхід. Рушій не є окремим процесом, натомість надається бібліотека яка компілюється разом з програмою і таким чином рушій включається в програму та стає її складовою частиною. У якості протоколу обміну інформації використовуються API. Цей підхід дозволяє знизити витрати, спрощує програмний код та зменшує час відгуку. Вся база даних зберігається в одному файлі на комп'ютері де використовується застосунок. Так як перед початком виконання транзакції весь файл в якому зберігається база даних блокується – забезпечується простота реалізації [13].

Одночасно декілька потоків або процесів можуть здійснювати читання з бази даних. Але здійснювати запис можна буде лише в тому випадку, коли у даний момент часу не здійснюється якийсь інший запис – тобто не обслуговується якийсь запит на запис. Для того щоб забезпечити одночасне прийняття запитів на запис та уникнення помилок можливе використання автоматичного повторення запитів. Архітектура рушія дозволяє використовувати базу даних як на вбудованих так і на виділених системах з достатньо великими масивами даних.

Слід виділити наступні особливості SQLite:

- транзакції атомарні, міцні, ізольовані та послідовні,
- реалізація значної частини стандарту SQL92,
- зберігання бази даних в одному файлі крос-платформенного типу,

- можливість встановлення без конфігурації – не потрібна інсталяція та адміністрування,
- мінімальний розмір коду,
- підтримка гігабайтного розміру рядка та терабайтних баз даних,
- висока швидкість для найпоширеніших операцій,
- простий у використанні,
- початковий код добре прокоментований та гілки на 100% покриті тестами,
- можливість використання у десятках мов програмування,
- відсутність зовнішніх залежностей,
- підтримка різних операційних систем,
- можливість використання автономного клієнта інтерфейсу командного рядка у якості СКБД.

Ефективність використання SQLite підтверджується тим, що дану базу даних використовують як формат зберігання даних у таких популярних та сучасних застосунках:

- Google Chrome
- Google Gears
- Zotero
- XUL
- Mendeley
- Amarok
- Songbird Banshee
- Gajim
- F-Spot[en]

Окрім перерахованих переваг SQLite також має певні слабкі місця, які обмежують сферу застосування та накладають певні обмеження на розроблювану систему:

- відсутність підтримки великої кількості одночасних запитів. Для уникнення конфліктів та помилок необхідно заздалегідь опрацьовувати можливі конфліктні моменти,
- відсутність підтримки клієнт-серверної архітектури та віддаленого доступу через мережу інтернет,
- через обмежену підтримку складно використовувати систему для високонавантажених систем,
- некомерційність проекту обумовлює відсутність технічної підтримки,
- обмежена функціональність так як основне призначення – операції CRUD (create, read, update, delete).

Легкість та простота SQlite обумовлює можливість використання в різних галузях, наприклад:

- застосування при розробці мобільних застосунків на різних платформах для зберігання налаштуваннь, журналів подій, контактів,
- застосування при розробці невеликих десктопних додатків у яких немає потреби використання складних баз даних,
- застосування при розробці додатків до вбудованих систем та портативних пристроїв – мультимедійного обладнання, маршрутизаторів, трекерів, розумних пристроїв та пристроїв інтернету речей,
- зберігання даних у веб-браузерах,
- навчальні проекти для роботи з мовою SQL,

Таким чином, не зважаючи на ряд слабких місць, SQlite має значні переваги для розроблюваного проекту системи обліку та аналізу складських запасів у порівнянні з аналогами – перш за все це легкість та простота використання що дозволить застосувати її в телеграм-боті без ускладнень та можливість написання додатку на мові Python без використання додаткових бібліотек. Існуючі недоліки SQlite на даний момент і в поточних масштабах системи не накладає обмежень на використання розроблюваної системи.

2.3. Вибір інструменту для аналізу використання запасів сільськогосподарського підприємства

Для керівника або особи яка використовує результати внесених у систему даних про рух складських запасів першочергове значення мають вимоги до інформації що отримана від системи – а саме точність та зручність використання отриманих даних.

У розроблюваній системі обліку та аналізу складських запасів кінцевими користувачами інформації виступають два актори – бухгалтер та керівник. Бухгалтер може використовувати інформацію в табличному вигляді для формування бухгалтерської звітності. Для керівника або аналітика доцільно представляти звітну інформацію не у табличному вигляді а у вигляді підсумків за певний часовий період по певному складу, або ж у графічному вигляді [14].

При наявності доступу до каталогу де знаходиться файл бази даних керівник або бухгалтер може самостійно відкрити базу даних та експортувати потрібну інформацію, але це не дуже зручно, крім того не завжди є можливість фізичного доступу до файлу бази даних. Таким чином для керівника більш зручним способом отримання даних є отримання звіту безпосередньо у додатку. Так, наприклад, керівних може отримувати звітну інформацію по стану складських запасів по певному складу, кількість переміщених запасів на склад або зі складу, кількість отриманих або взятих ресурсів певним працівником.

Так як кожен користувач телеграм має унікальний ідентифікатор, можна налаштувати телеграм-бот таким чином щоб комірник або інші користувачі окрім керівника не мали доступу до звітної підсумкової інформації.

Однією з найбільш поширених бібліотек для візуалізації даних є бібліотека matplotlib. Дана бібліотека легко підключається до мови програмування Python та надає розширені можливості створення графічних об’єктів на основі рядів даних [15].

Бібліотека matplotlib дозволяє будувати двовимірні графіки, а також підтримує трьох вимірну графіку. Отримані зображенні можуть генеруватись в різних форматах та можуть бути використані у графічному інтерфейсі користувача, вебдодатках. Бібліотека була написана Джоном Хантером, який також здійснює підтримку.

Остання стабільна версія бібліотеки matplotlib 1.2.0 яка потребує використання мінімальної версії Python 2.6. Незважаючи на те, що бібліотека побудована на принципах об’єктно-орієнтованої парадигми, вона використовує процедурний інтерфейс.

Бібліотека matplotlib дозволяє будувати велику кількість графіків та діаграм:

- диграми розсіювання,
- лінійні графіки,
- стовпчасті діаграми та гістограми,
- секторні діаграми,
- контурні графіки,
- спектрограми,
- поля градієнтів.

Окрім побудування діаграм користувач може вказувати підписи осей координат, налаштовувати сітку, вставляти підписи, використовувати нестандартну шкалу або координати [16].

Зберігати побудовані графіки можна в наступних поширених форматах:

- GPEG,
- PDF,
- PNG,
- SVG,
- TIFF,
- EPS,
- EMF,
- SVGZ,

- RGBA.

Так, наприклад, виконання наступного коду:

```
from pylab import *
plot(range(1, 20),
     [i * i for i in range(1, 20)], 'ro')
savefig('example.png')
show()
```

Дозволяє побудувати наступний графік:

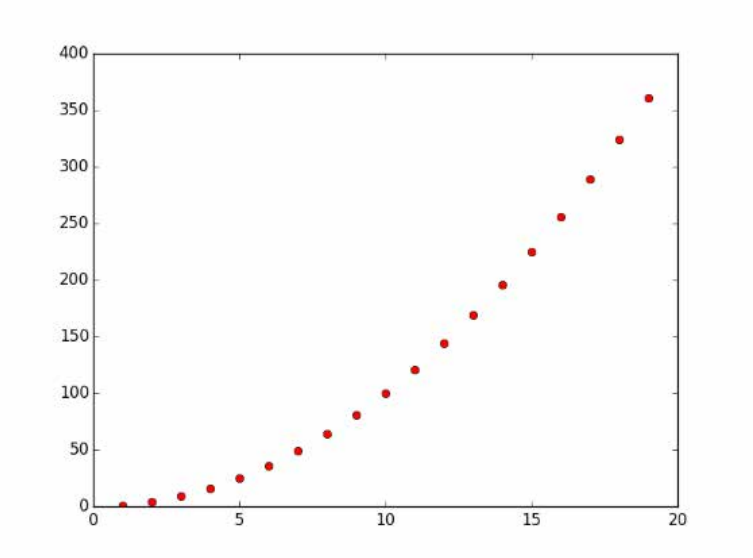


Рис. 2.1. Приклад побудування графіку у matplotlib

Таким чином, використання бібліотеки matplotlib дозволяє легко та швидко будувати діаграми для візуалізації інформації з системи для демонстрації керівнику або іншим користувачам системи.

РОЗДІЛ 3. РОЗРОБКА ДОДАТКА СИСТЕМИ ОБЛІКУ ТА АНАЛІЗУ ВИКОРИСТАННЯ ЗАПАСІВ СІЛЬСЬКОГОСПОДАРСЬКОГО ПІДПРИЄМСТВА

3.1. Розробка додатку для отримання інформації від користувачів

Перед тим як приступити до процесу розробки додатку виділимо основних учасників (акторів) та побудуємо діаграму прецедентів, яка визначатиме основні функції що буде визначати функціональність побудованої системи.

Першим актором виступає комірник або працівник складу, який безпосередньо здійснює складські операції – вносить ресурс на склад або отримує ресурс зі складу. На діаграмі прецедентів позначимо комірника як співробітника. Фактично в деяких малих підприємствах керівник може одночасно виконувати функції комірника.

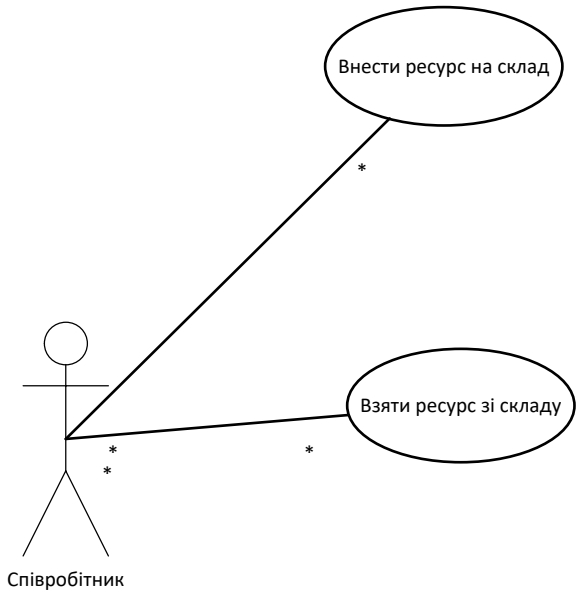


Рис. 3.1. Діяльність співробітника на діаграмі прецедентів

Окрім комірника обов’язковим учасником системи є керівник, який у майбутньому буде використовувати звітну інформацію для подальшого аналізу або прийняття рішень. Керівник може отримувати інформацію про поточний

стан складу або складів – визначати кількість залишків, отримувати інформацію про те скільки операцій та якого типу здійснив певний працівник, кількість операцій на певну дату.

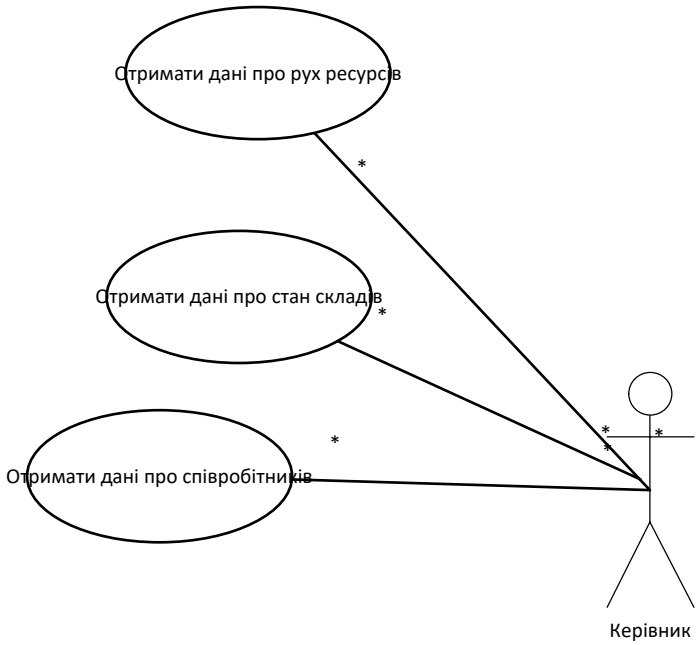


Рис. 3.2. Діяльність керівника на діаграмі прецедентів

Бухгалтер отримує інформацію про поточний стан складу та кількість операцій з переміщення ресурсів. Далі ці дані можуть співставлятись з даними про сплату ресурсів чи розрахунки з контрагентами. Бухгалтер як правило не аналізує роботи певних працівників або розподіл ресурсів між різними складами. Основна задача бухгалтера – оцінка точності обліку складу, а також використання цих даних у якості первинної документації для ведення бухгалтерського обліку.

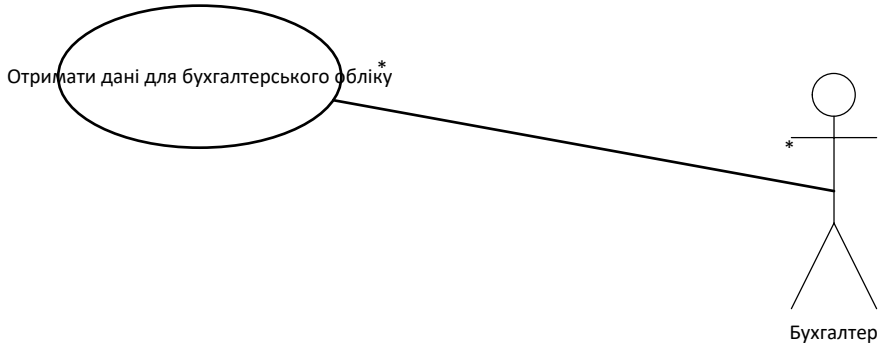


Рис. 3.3. Діяльність бухгалтера на діаграмі прецедентів

У майбутньому до системи можуть додаватись інші актори. Наприклад, контрагенти також можуть вносити інформацію в систему про постачання запасів на склад або отримання зі складу. Таким чином, може здійснюватися подвійний облік одної і тої самої операції з одного боку комірником і з іншого боку контрагентом.

Також у системі у майбутньому може приймати участь у якості актора виробництво, яке буде використовувати відповідні ресурси для здійснення основної діяльності підприємства. Так, наприклад, один із співробітників може вносити план виробництва або інші планові показники для того щоб враховувати потреби підприємства у ресурсах.

У якості середовища розробки обрана IDE PyCharm та мова програмування Python.

Першим кроком є підключення відповідних бібліотек.

Бібліотека sqlite3 є вбудованою та використовується для підключення та роботи з базою даних.

Бібліотека telebot встановлюється додатково та використовується для роботи з месенджером телеграм.

Бібліотека matplotlib встановлюється додатково та використовується для візуалізації інформації – побудування графіків.

Інформація про токен телеграм-боту має бути прихована від сторонніх користувачів і тому зберігається в окремому файлі конфігурації, де також за

необхідності може бути шлях до бази даних або інші параметри конфігурації програми.

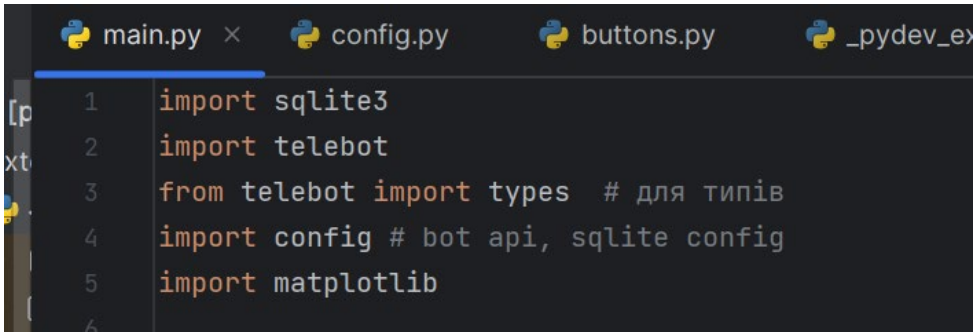


Рис. 3.4. Підключення бібліотек

Для того щоб створити телеграм-бота, використовується відповідний токен, який був створений при звертанні до головного бота на ім’я BotFather. База даних на даний момент зберігається у тому ж каталозі що і основна програма у каталозі db.

```
7
8   bot = telebot.TeleBot(config.token)
9
10  conn = sqlite3.connect('db/database.db', check_same_thread=False)
11  cursor = conn.cursor()
```

Рис. 3.5. Отримання токену телеграм-бота

Для того щоб здійснювати запис у базу даних буде використано 2 методи - def db_table_val та def db_table_val2. Перший метод буде використовуватись для того щоб обробляти операції внесення на склад, тобто додавати ресурси на склад. Другий метод буде використовуватись що б обробляти операції отримання ресурсу зі складу.

Обидва методи будуть приймати однакові параметри – код користувача, кількість ресурсу, код ресурсу, код складу, і здійснюють записи у різні таблиці – дебет та кредит.

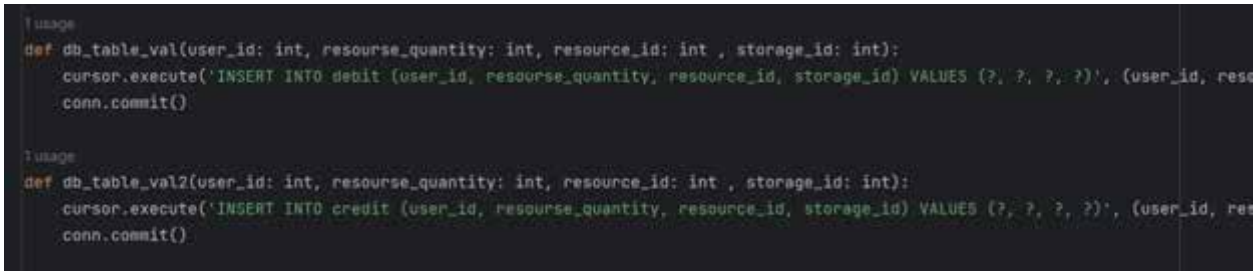


Рис. 3.6. Підключення бази даних

Єдина команда яку приймає бот у якості команди є команда \start, в результаті виконання будуть викликані дві кнопки – отримати звіт та складські операції. Дані кнопки по суті містять рядок символів, який буде передаватись при натисканні як текстове повідомлення для бота. Саме обробка такого текстового рядка та його інтерпретація буде сприйматись як певна дія користувача.

Також окрім кнопок з можливими варіантами дій користувач побачить повідомлення з власним ім’ям. Для отримання імені користувача використовується ім’я користувача телеграм, який надіслав повідомлення.

Рис. 3.7. Обробка команди старт

Для обробки команд, які поступають від користувача у вигляді натискань на кнопки використовується метод @bot.message_handler який приймає текст.

Далі у програмі додається декоратор даного методу, який буде аналізувати введений текстовий рядок (натискання кнопки).

Так, при натисканні кнопки отримати звіт буде виведено повідомлення в телеграм користувачу про те що звіт виведено в консоль, та безпосередньо у консоль буде виводитись звіт через метод fetchall.

```
@bot.message_handler(content_types=['text'])
def func(message):

    if (message.text == "👋 Отримати звіт"):
        cursor.execute("SELECT * FROM debit")
        print(cursor.fetchall())
        bot.send_message(message.chat.id, text="Звіт виведено в консоль)")
```

Рис. 3.8. Обробка кнопки отримати звіт

Якщо користувач телеграм-бота натискає на кнопку складські операції то йому пропонується меню що складається з наступних кнопок:

- внести на склад,
- взяти зі складу,
- повернутись в головне меню.

```
    elif (message.text == "❓ Складські операції"):
        markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn1 = types.KeyboardButton("Внести на склад")
        btn2 = types.KeyboardButton("Взяти зі складу")
        back = types.KeyboardButton("Повернутись в головне меню")
        markup.add(btn1, btn2, back)
        bot.send_message(message.chat.id, text="Оберіть операцію", reply_markup=markup)
```

Рис. 3.9. Обробка кнопки складські операції

Якщо користувач телеграм-бота натискає на кнопку внести на склад то йому виводиться наступна інструкція: Введіть через пробіл кількість(кг) номер продукції номер складу. Телеграм бот може обробити одне повідомлення за

один раз, тому доцільно приймати одне повідомлення у вигляді текстового рядка, яке далі буде розділено на складові. Це може бути не зовсім зручно для користувача, але з іншого боку даний механізм простіше в реалізації та кожна операція може бути відслідкована, тобто кожен запис у таблицю вноситься за один раз, а не поступово заповнюються поля.

```
elif (message.text == "Внести на склад"):
    bot.send_message(message.chat.id, "Введіть через пробіл кількість(кг)  номер продукції номер складу")
```

Рис. 3.10. Обробка кнопки внести на склад

Якщо користувач телеграм-бота натискає на кнопку взяти зі складу то йому виводиться наступна інструкція: Введіть через пробіл кількість(кг) з мінусом номер продукції номер складу.

```
elif message.text == "Взяти зі складу":
    bot.send_message(message.chat.id, text="Введіть через пробіл кількість з мінусом(кг)  номер продукції номер складу")
```

Рис. 3.11. Обробка кнопки взяти зі складу

Якщо користувач телеграм-бота натискає на повернутись в головне меню то відповідно знову виводяться дві доступні кнопки – отримати звіт та складські операції.

```
elif (message.text == "Повернутись в головне меню"):
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    button1 = types.KeyboardButton("👋 Отримати звіт")
    button2 = types.KeyboardButton("❓ Складські операції")
    markup.add(button1, button2)
    bot.send_message(message.chat.id, text="Ви повернулись в головне меню", reply_markup=markup)
```

Рис. 3.12. Обробка кнопки повернутись в головне меню

Всі інші повідомлення, не описані у перелічених кнопках будуть сприйматись телеграм ботом як текстовий рядок, що містить у собі інформацію про складську операцію – внесення на склад або отримання зі складу.

Тобто кожне таке повідомлення програма буде намагатись перетворити у числові дані та записати у базу даних. На даний момент виключення не використовуються для аналізу некоректних даних.

```
else:

    b = message.text.split()
    a=int(b[0])
    k=int(b[1])
    m = int(b[2])
    if (a>0):
        us_id = message.from_user.id
        res_quantity = a
        res_id=k
        stor_id=m
        db_table_val(user_id=us_id, resourse_quantity=res_quantity, resource_id=res_id, storage_id=stor_id)
    elif (a<0):
        us_id = message.from_user.id
        res_quantity = -a
        res_id = k
        stor_id=m
        db_table_val2(user_id=us_id, resourse_quantity=res_quantity, resource_id=res_id, storage_id=stor_id)
```

Рис. 3.13. Обробка повідомлень для запису в базу даних

Таким чином, представлений код телеграм боту дозволяє обробляти команди та натискання кнопок зі сторони користувача та відповідно виводити меню, повертатись в головне меню або здійснювати запис у базу даних.

3.2. Розробка бази даних додатку

У якості мови програмування для на якій буде написано додаток телеграм бот обрано мову Python, що має вбудовану бібліотеку для роботи с базою даних SQLite – це спрощує процес розробки та забезпечує стабільну роботу додатку без необхідності використання бібліотек від сторонніх організацій.

SQLite представляє собою полегшену систему керування базами даних реляційного типу. Вона розроблена у вигляді бібліотека в якій реалізовано стандарт SQL. SQLite може використовуватись без обмежень так як початковий код поширюється безкоштовно. Для налаштування бази даних використаємо SQLiteStudio. У вкладці Database обираємо Add a database.

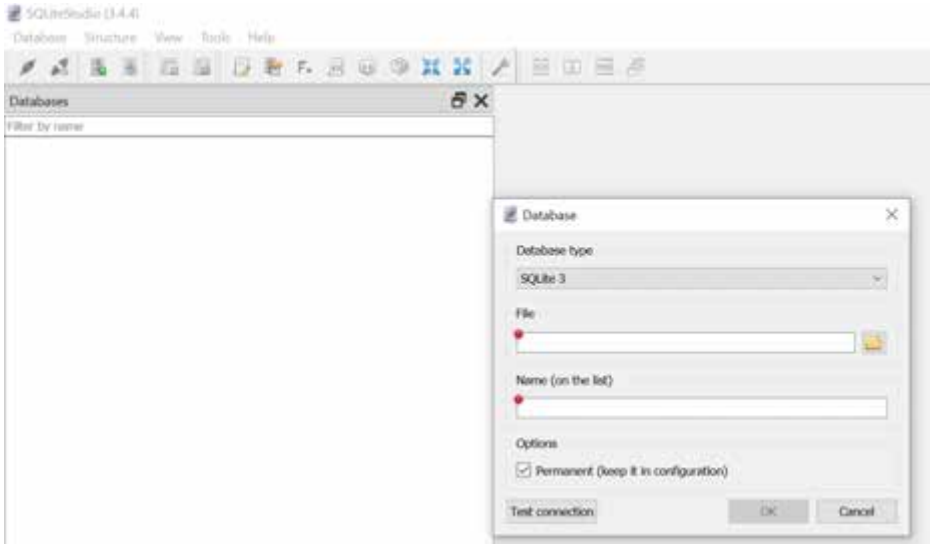


Рис. 3.14. Додавання бази даних у SQLiteStudio

Далі створюємо новий файл бази даних та обираємо каталог проекту де буде зберігатись файл бази даних. Для цього оберемо окрему папку db, де і буде знаходитись файл з базою даних. Так як в попередньому розділі вже було вказано шлях та назву бази даних, то використаємо цю назву для іменування файлу бази даних – database. Розширення присвоюється файлу автоматично. Тепер у списку доступних файлів з’являється файл бази даних, далі через контекстне меню обираємо пункт Connect to the database, в результаті чого

встановлюється з’єднання з базою даних. Після цього доступно два підпункти – Таблиці та Представлення. Для додавання таблиці потрібно натиснути кнопку Create a table.

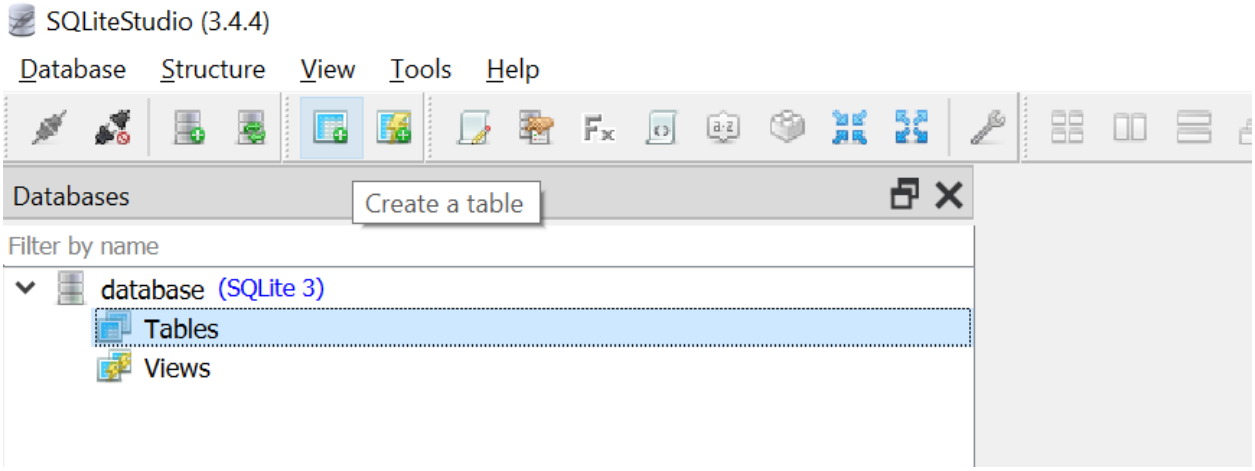


Рис. 3.15. Додавання таблиці у SQLiteStudio

У полі для введення введемо ім’я таблиці – debit. Дана таблиця буде зберігати дані про прихідні операції – тобто операції внесення ресурсів на склад.

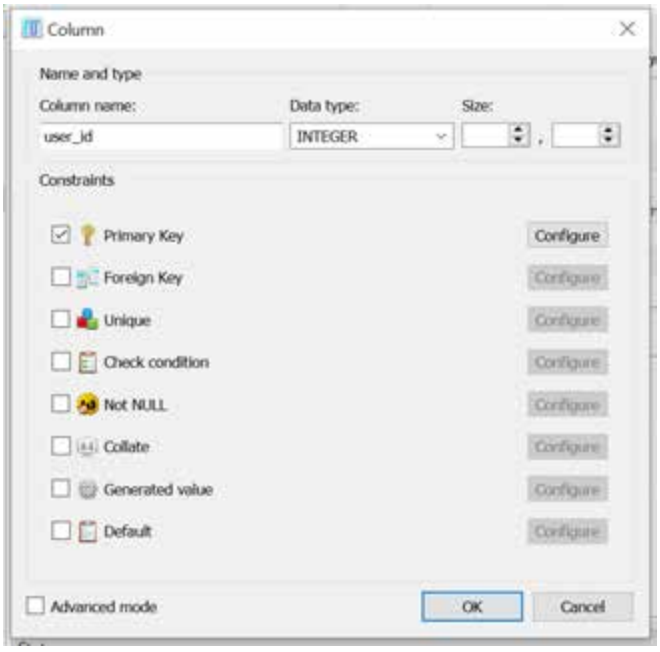
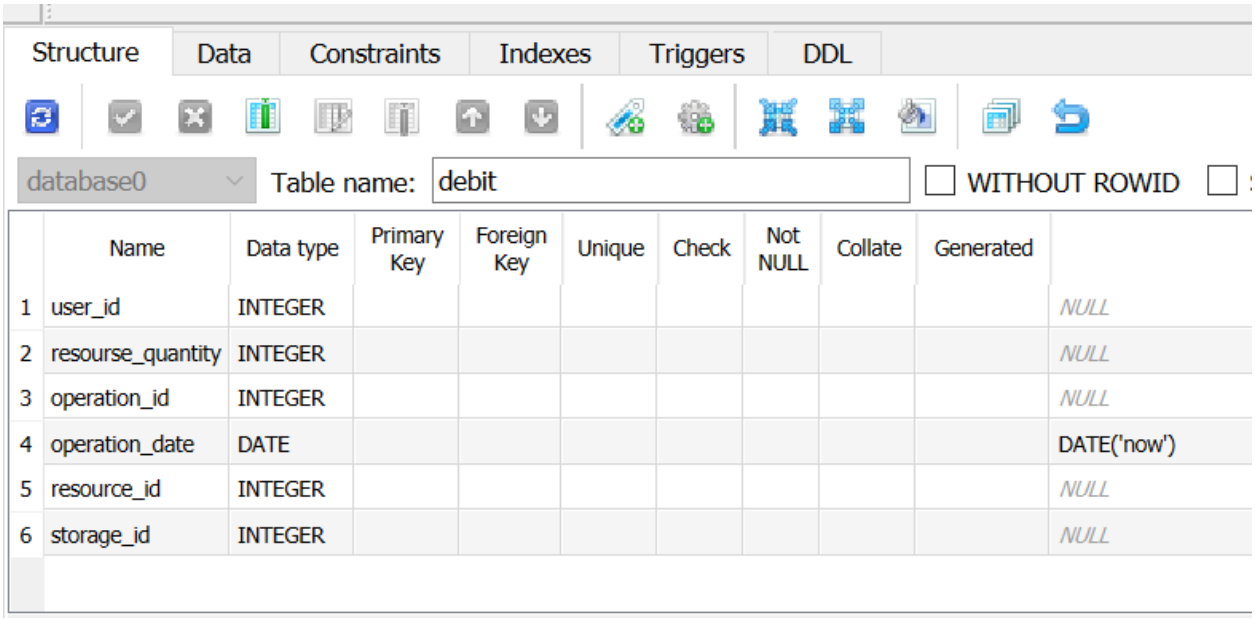


Рис. 3.16. Додавання поля код користувача у таблицю debit

Унікальним ключем яким може однозначно ідентифікувати будь-якого користувача телеграм є код користувача. Код користувача можна отримати з повідомлення та зберегти у таблицю. Дане поле буде мати цілочислений тип, та буде використовуватись як зовнішній ключ.



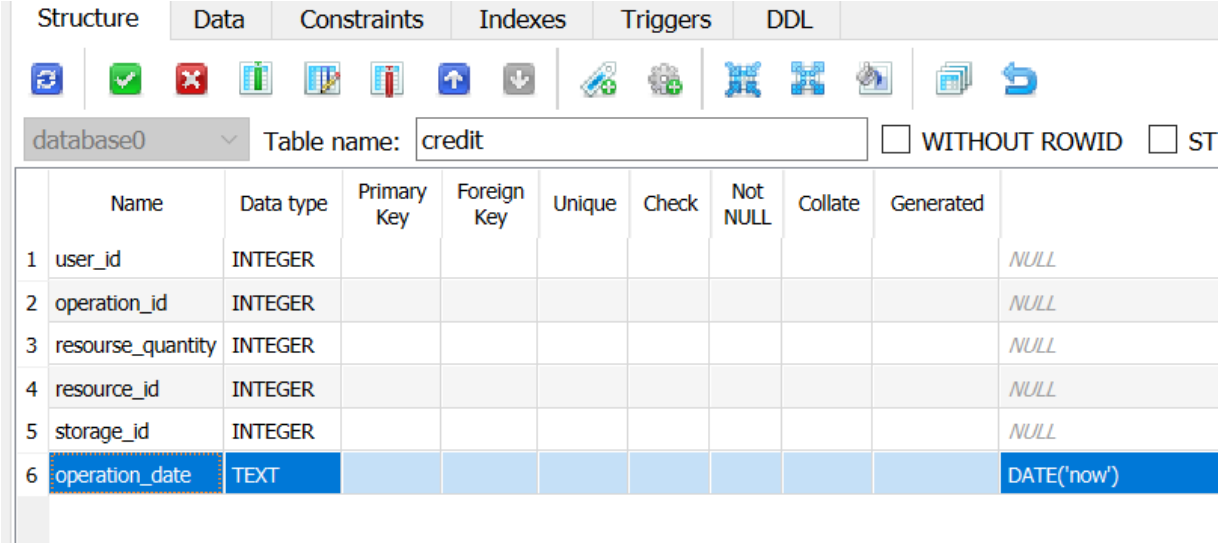
	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	user_id	INTEGER								NULL
2	resourse_quantity	INTEGER								NULL
3	operation_id	INTEGER								NULL
4	operation_date	DATE								DATE('now')
5	resource_id	INTEGER								NULL
6	storage_id	INTEGER								NULL

Рис. 3.17. Структура таблиці debit

Таблиця debit має наступні поля:

- user_id – код користувача, що отримується з повідомлення та використовується для ідентифікації користувача,
- resurse_quantity – кількість ресурсу, що вноситься на склад працівником, тип цілочисельний, за умовчанням на даний момент використовується міра у кілограмах (зернові, кукурудза і т.д.),
- operation_id – код операції, що визначає порядковий номер внесення ресурсу на склад. Код операції може використовуватись для того що б уникнути втручання в базу даних, так як зміна одного запису призведе до зміни нумерації у всій таблиці,
- resource_id – код ресурсу, що визначає тип ресурсу що зберігається на складі, тип поля цілочисельний,

- storage_id – код складу, що визначає номер складу на який вноситься ресурс, тип поля цілочисельний.



	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	user_id	INTEGER								NULL
2	operation_id	INTEGER								NULL
3	resourse_quantity	INTEGER								NULL
4	resource_id	INTEGER								NULL
5	storage_id	INTEGER								NULL
6	operation_date	TEXT								DATE('now')

Рис. 3.18. Структура таблиці credit

Таблиця credit має наступні поля:

- user_id – код користувача, що отримується з повідомлення та використовується для ідентифікації користувача,
- resurse_quantity – кількість ресурсу, отримується на склад працівником, тип цілочисельний, за умовчанням на даний момент використовується міра у кілограмах (зернові, кукурудза і т.д.),
- operation_id – код операції, що визначає порядковий номер отримання ресурсу зі складу. Код операції може використовуватись для того що б уникнути втручання в базу даних, так як зміна одного запису призведе до зміни нумерації у всій таблиці,
- resource_id – код ресурсу, що визначає тип ресурсу що зберігається на складі, тип поля цілочисельний,
- storage_id – код складу, що визначає номер складу з якого отримується ресурс, тип поля цілочисельний.

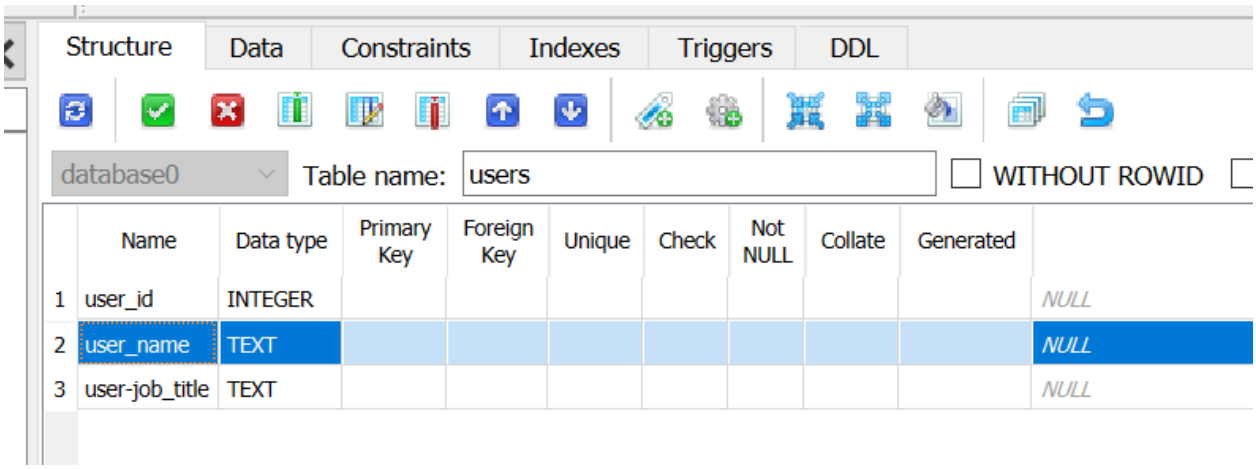


Рис. 3.19. Структура таблиці users

Таблиця users має наступні поля:

- user_id – код користувача, що отримується з повідомлення та використовується для ідентифікації користувача, дане поле може використовуватись як ключ та буде унікальним для кожного запису – кожен користувач має власний унікальний код, крім того дане поле буду слугувати зв'язком з таблицями у які вносяться дані про складські операції,
- user_name – ім'я користувача, поле має текстовий формат,
- user_job_title – назва посади користувача, поле має текстовий формат.

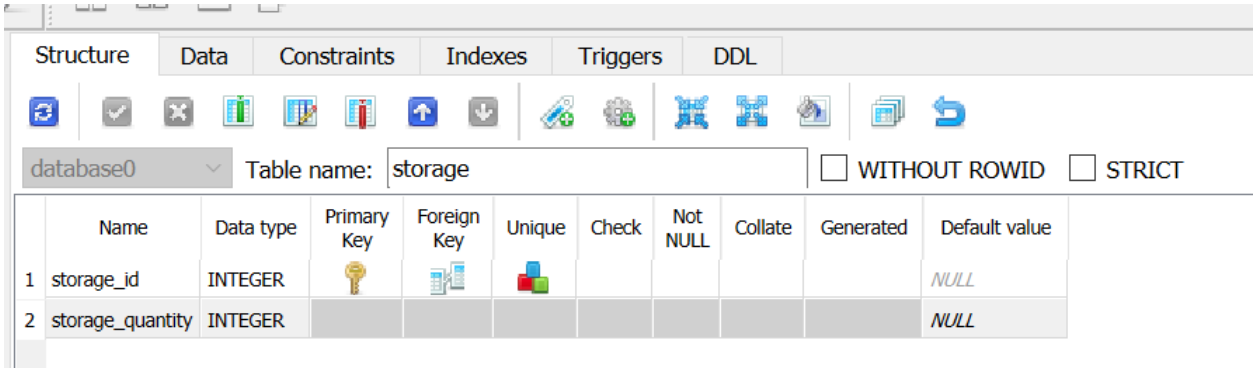


Рис. 3.20. Структура таблиці storage

Таблиця storage має наступні поля:

- storage_id – код складу, що визначає номер складу, тип поля цілочисельний, дане поле використовується як ключове тому що однозначно ідентифікує певний склад а також використовується як зовнішній зв’язок для того щоб отримувати дані з таблиць debit та credit про відповідні операції та визначати кількість ресурсу на складі,
- resurse_quantity – кількість ресурсу, що зберігається на складі – розрахункове поле що визначається за допомогою отриманих із таблиць debit та credit даних про складські операції.

Зв’язок між таблицями можна представити у вигляді схеми рис. 3.21.

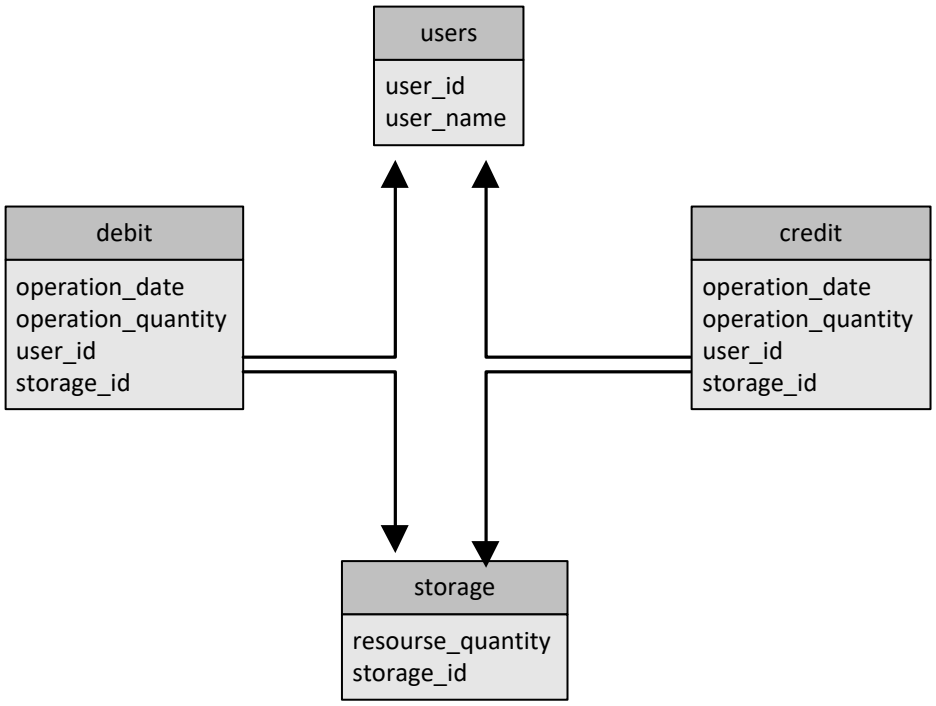


Рис. 3.21. Зв’язок між таблицями

Таким чином, структура сховища даних представлена чотирма таблицями:

- users – зберігається інформація про користувачів системи

- storage – зберігається інформація про склад або склади та поточний стан запасів
- debit – інформація про операції внесення на склад
- credit – інформація про операції отримання зі складу

Зв'язок між таблицями дозволяє отримувати інформацію про складські операції та групувати їх по певному складу або по певному працівнику.

3.3. Тестування додатку

З технічної точки зору тестування це запуск та робота програмної системи на деякій множині початкових даних і звірці одержуваних результатів із заздалегідь відомими (еталонними) з метою встановлення відповідності різних властивосте

Сьогодні до програмних продуктів висуваються досить високі вимоги щодо якості через високу конкуренцію та широку пропозицію в багатьох сферах ринку програмного забезпечення. Якість визначається двома факторами: специфікаціями виробника та специфікаціями споживача. Специфікації виробника формують об'єктивні вимоги до програмного забезпечення, що виробляються в умовах вільної конкуренції.

Іншими словами, встановлюючи вимоги до кінцевого продукту, виробники припускають, що програмне забезпечення буде конкурентоспроможним і матиме найнижчу можливу вартість розробки. Специфікації замовника представляють загальні очікування кінцевих користувачів щодо продукту, який розробляється. Задоволення обох цих специфікацій розуміється як якість програмного забезпечення.

Звичайно, не всі очікування можуть бути виражені формально, і не завжди можливо виміряти, чи відповідає продукт очікуванням. Тому прийнято говорити про вимірювані очікування. Ці очікування виражаються в

специфікаціях, які є заздалегідь визначеними, формальними вимогами до кінцевого продукту.

Тестування є частиною забезпечення якості кінцевого продукту, оскільки дозволяє контролювати відповідність специфікації на всіх етапах розробки.

Тому тестування слід вважати необхідним і важливим етапом у розробці програмного забезпечення.

Загальною метою тестування є виявлення дефектів програмного забезпечення, що є однією з цілей забезпечення якості єдиного процесу розробки програмного забезпечення. Однак тестування не обмежується лише виявленням дефектів, а й включає в себе управління виправленням виявлених дефектів. Таким чином, цілі тестування програмного забезпечення можна визначити як:

- виявлення дефектів на різних етапах життєвого циклу програми (під час розробки, під час супроводу тощо);

- перевірка того, чи виявлені дефекти були успішно усунені;

- забезпечення того, що зміни, пов'язані з усуненням виявлених дефектів, були внесені перевірка того, що зміни, пов'язані з усуненням виявлених дефектів, не призвели до появи нових дефектів у системі.

Виходячи з цілей, перед тестуванням ставляться наступні завдання:

- виявлення помилок в моделі системи,

- виявлення помилок в кодуванні,

- виявлення помилок і несправностей у взаємодії системи з навколишнім середовищем і з зовнішніми компонентами і системами,

- виявлення помилок в інтеграції програмного забезпечення - виявлення помилок в продуктивності системи,

- виявлення помилок в перевантаженні системи,

- виявлення помилок в роботі системи виявлення нестабільності програмного забезпечення під навантаженням,

- виявлення нестабільності програмного забезпечення через некоректне введення даних або збої при збільшенні обсягів оброблюваних даних,

- виявлення вразливостей у системі безпеки програми та можливого несанкціонованого доступу.

Вважається, що програма працює коректно, якщо виконуються такі критерії:

- Якщо отримано коректні дані, програма видає коректні результати.

- Якщо отримано некоректні дані, програма їх відкидає.

- Прийняття як правильних, так і неправильних даних не призводить до "зависання" або "аварійного завершення роботи" програми.

Тестування програмного забезпечення (англ. Software Testing) – техніка контролю якості, що перевіряє відповідність між реальною і очікуваною поведінкою програми завдяки кінцевому набору тестів, які обираються певним чином. Техніка тестування також включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою оцінки.

Тестування пронизує весь життєвий цикл програмного забезпечення, від розробки до безперервної фази експлуатації, і безпосередньо пов'язане з вимогами та управлінням змінами. Це пов'язано з тим, що метою тестування є забезпечення відповідності програмного забезпечення заявленим вимогам.

Таким чином, при тестуванні розробленої системи будемо оцінювати наступні параметри:

- відповідність вимогам, які висувались до програного додатку перед початком розробки та враховували потреби кінцевого користувача системи, відповідність задачам замовника;
- коректна обробка всіх можливих варіантів коректних дій від всіх користувачів системи;
- коректна обробка всіх можливих варіантів некоректних дій та введення некоректних даних від всіх користувачів системи;
- виконання всі потрібних задач за прийнятний час;
- зручність використання;
- сумісність з різними платформами та операційними системами;
- зручний користувацький інтерфейс.

Перед початком розробки до програмного додатку висувались наступні вимоги:

- запис операцій отримання зі складу та внесення на склад у базу даних,
- неможливість втручання в записи в базі даних, неможливість видалення даних про вже записані операції без адміністратора системи,
- підтримка системи максимально можливою кількістю пристроїв та операційних систем,
- можливість перегляду звітної інформації бухгалтером та керівником.

РОЗДІЛ 4. ПРАКТИЧНЕ ВИКОРИСТАННЯ ДОДАТКУ ТА АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

4.1. Приклад реального використання додатку

Система обліку та аналізу використання запасів сільськогосподарського підприємства працює наступним чином:

1. Внесяться дані про складські операції:
   - вид операції
   - кількість ресурсу
   - час здійснення операції
   - прізвище співробітника
2. Дані зберігаються на сервері
3. Керівник отримує дані у вигляді таблиць через СКБД або у вигляді графіків, діаграм
4. Бухгалтер отримує дані у вигляді таблиць які використовуються як первинні дані для бухгалтерського обліку

Рис. 4.1. Принцип роботи системи обліку та аналізу складських запасів підприємства

Першим кроком роботи з додатком є команда старт для телеграм-бота, в результатові якої телеграм бот отримує від користувача наступні дані – ім’я користувача, яке зареєстроване в телеграмі а також код користувача. На основі кода користувача при необхідності можливе відфільтрування незареєстрованих користувачів. Код користувача є унікальним та саме на основі цього коду можливий доступ до телеграм бота (рис. 4.2.).

При необхідності можливе використання різної кількості команд, але на даний момент телеграм бот обробляє лише основну команду старт. Так, наприклад, можливе використання інших команд для реалізації іншого функціоналу телеграм бота – для звітності та внесення можуть використовуватись різні команди.

Рис. 4.2. Обробка команди старт

Після отримання команди старт телеграм бот виводить привітання із вказанням імені користувача, що було отримано з повідомлення. Крім

привітання телеграм бот виводить дві кнопки для того щоб користувач обрав наступну дію – отримати звіт або здійснити складську операцію (рис. 4.3.).

Рис. 4.3. Обробка кнопки складські операції

Після натискання користувачем кнопки складські операції користувач може далі натиснути три можливі кнопки:

- Внести на склад
- Взяти зі складу
- Повернутись в головне меню

Запропоновані кнопки обробляються телеграм ботом як звичайні текстові повідомлення, але наявність кнопок у порівнянні з текстом набагато спрощує роботу користувача.

Якщо користувач обирає внесення на склад то телеграм бот підказує користувачу що далі потрібно ввести через пробіл кількість, номер продукції та номер складу(рис. 4.4.).

Рис. 4.4. Обробка кнопки Внести на склад

На даний момент робиться припущення про те, що користувачу заздалегідь відомі номер доступних складів, а також номери продукції. За потреби можливе виведення інформації для користувача наприклад у наступному вигляді:

1 – пшениця,

2 – ячмінь,

3 – кукурудза,

4 – рапс,

5 – насіння соняшника.

Після того як користувач ввів дані про внесення ресурсу на склад, якщо все було коректно то телеграм бот пропонує йому обрати ще одну операцію.

Або користувач може вийти в головне мені, або закрити телеграм бот якщо робота з системою закінчена і більше не буде здійснено ніяких операцій на даний момент (рис. 4.5.).

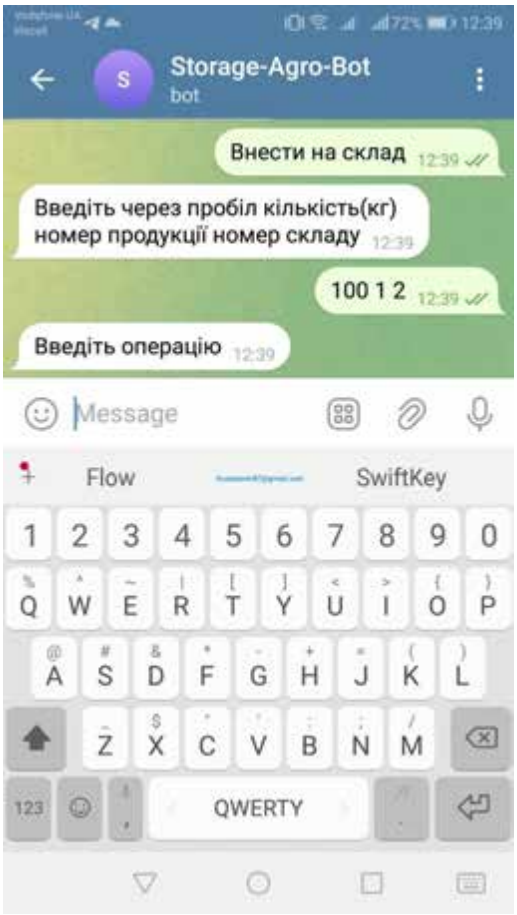


Рис. 4.5. Обробка внесеної інформації про внесення на склад

При обробці кнопки отримання зі складу так само поки що робиться припущення про те, що користувачу заздалегідь відомі номер доступних складів, а також номери продукції. За потреби можливе виведення інформації для користувача наприклад у наступному вигляді:

1 – пшениця,

2 – ячмінь,

3 – кукурудза,

4 – рапс,

5 – насіння соняшника.

Рис. 4.6. Обробка кнопки взяти зі складу

Після того як користувач ввів дані про отримання ресурсу зі складу, якщо все було коректно то телеграм бот пропонує йому обрати ще одну операцію. Або користувач може вийти в головне мені, або закрити телеграм бот якщо робота з системою закінчена і більше не буде здійснено ніяких операцій на даний момент (рис. 4.6.).

Обробка будь-яких інших повідомлень окрім запропонованих кнопок та внесення інформації про здійснення складської операції на даний момент не обробляється та ігнорується телеграм-ботом. За необхідності такий функціонал легко може бути доданий, наприклад користувач у разі некоректно обраної команди може отримати наступні повідомлення:

- оберіть іншу команду, дана команда недоступна,
- оберіть інший склад,
- оберіть інший код ресурсу,

- введіть кількість у вигляді числа.

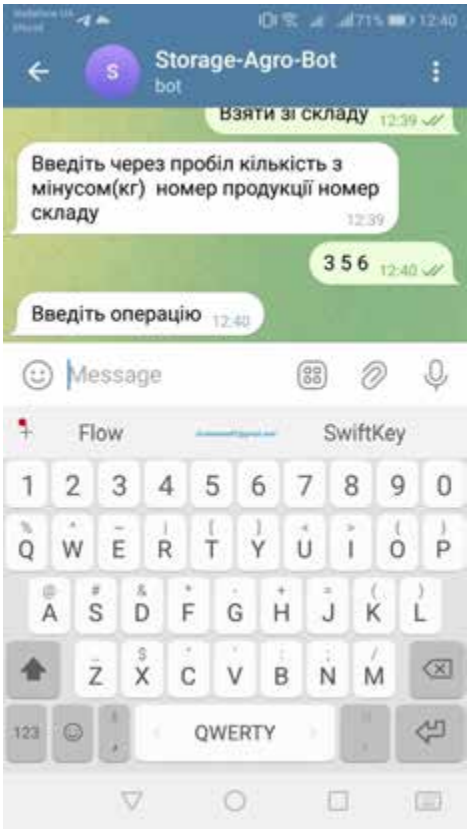


Рис. 4.8. Обробка внесеної інформації про отримання зі складу

При натисканні кнопки повернутись в головне меню користувач відповідно отримує повідомлення про те що він повернувся в головне меню а також отримує можливість обрати одну із доступних кнопок головного меню – отримати звіт чи обрати складські операції (рис. 4.9).

Також при необхідності можливе додавання інших підрозділів або розділів у меню – внесення інформації про користувачів у базу даних, які надалі зможуть вносити інформацію про складські операції. Відповідно, додавати інших користувачів може тільки керівник, який має відповідні права що перевіряються за його унікальним кодом користувача.

Телеграм бот може приймати не лише текстові повідомлення, а також і зображення. Для розширення функціоналу за вимогою замовника у майбутньому телеграм бот може приймати фотографії які будуть

підтверджувати проведення складської операції та зберігатись у базі даних в окремому полі.

Рис. 4.9. Повернення в головне меню

Повідомлення про складські операції отримані та оброблені телеграм ботом зберігаються у формі записів у таблицях бази даних – у таблиці дебіт та таблиці кредит залежно від операції.

4.2. Подальший розвиток та розширення функціональних можливостей додатку

Результатом виконання дипломної роботи є телеграм бот що є виступає як система обліку та аналізу використання запасів сільськогосподарського підприємства. На поточний момент система дозволяє вводити інформацію про кількість внесеного на склад ресурсу (наприклад, пшениця, ячмінь і т.д.), номер складу на який вноситься даний ресурс, а також вид операції – тобто видача зі

складу чи внесення на склад. Дата здійснення операції автоматично вставляється у базі даних та не може бути скоригована користувачем. Крім того, кінцевий користувач не має права змінювати чи видаляти вже зафіксовану інформацію, так як користується системою через інтерфейс телеграм-бота і не має доступу до файлів бази даних. Використання унікального коду користувача телеграм дозволяє фільтрувати сторонніх користувачів та розділяти рівні доступу для різних користувачів – тобто звичайний користувач не може бачити підсумкові дані або звіти.

В даний момент основною реалізованою функціональною задачею системи є не заміна, а лише доповнення існуючої системи бухгалтерської звітності та заміни частини операцій складського обліку що раніше виконувались у вигляді ведення паперових журналів обліку або заповнення електронних таблиць. Перевага використання телеграм-бота у якості основного інтерфейсу користувача полягає у поширеності даного месенджеру та легкості розгортання системи – для початку використання достатньо лише мати пристрій з встановленим месенджером. Програма може запускатись з будь-якого сервера підключеного до мережі інтернет. На відміну від мобільних додатків, які на сьогоднішній день також є досить популярними рішеннями для оптимізації та автоматизації бізнес-процесів невеликих підприємств, телеграм-бот має менший функціонал але при цьому простіше у розгортанні та використанні.

Побудована у SQLite база даних може використовуватись як сховище даних для розробленого телеграм-бота а також до неї може отримати прямий доступ керівник або бухгалтер, якщо необхідно експортувати таблиці з даними про складські операції та використати їх для заповнення бухгалтерської звітності.

Також було проведено тестування телеграм боту. В результаті проведення позитивного та негативного тестування було використано відповідно коректні та некоректні дані які було успішно та адекватно оброблено системою.

Майбутній розвиток побудованої системи може полягати у розширенні функціоналу та можливості використання додатковими категоріями користувачів – наприклад, контрагентами чи іншими категоріями користувачів окрім комірника, керівника та бухгалтера.

Так, наприклад, важливою задачею в управління аграрним підприємством окрім управління запасами є облік інших товарно-матеріальних цінностей, таких як паливо.

Облік використання та залишків палива є актуальною задачею в управлінні аграрним підприємством через високу вартість даного ресурсу та неможливість його заміни при виконанні більшості основних робіт підприємства.

На відміну від інших ресурсів, контроль використання палива може також включати моніторинг ефективності його використання безпосередньо під час виконання основних робіт. Для цього використовуються датчики контролю рівня палива на транспортні засоби, які дозволяють отримувати інформацію про використання палива на кожному окремому автомобілі чи тракторі. Тобто, здійснюється порівняння фактичних та нормативних показників витрат як в ручному так і в автоматичному режимі. При суттєвому перевищенні нормативних показників а також їх порівнянні з показниками що вказані при отриманні ресурсу зі складу можливий аналіз ефективності використання палива. Також виявлення фактів нераціонального використання палива може бути повідомлено керівнику чи бригадиру за допомогою системи оповіщення яка встановлюється безпосередньо на транспортному засобі. Повноцінний та всебічний аналіз ефективності використання палива за допомогою пристроїв вимірювання рівня палива, датчиків GPS що визначають переміщення техніки та системи управління складськими запасами дозволяє отримати об’єктивну оцінку та визначити можливі шляхи підвищення ефективності використання палива на підприємстві.

При цьому, у якості окремої категорії користувачів телеграм-боту як системи обліку може виступати категорія водіїв які здійснюють експлуатацію

транспортних засобів та відповідно отримують на складі та використовують паливо. Кожна операція з отримання палива на складі (заправки на станції) може фіксуватись в загальній базі даних за допомогою телеграм-бота та надалі використовуватись при формуванні загальної відомості використання палива. Керівник або аналітик може отримувати дані з цієї бази даних та оцінювати використання палива певним працівником за звітний період, порівнювати цей показник з нормативними витратами при виконанні робіт які були закріплені за цим працівником, і на основі цієї інформації робити висновки про ефективність використання палива конкретним працівником.

Обмеженість графічного інтерфейсу телеграм-боту не дозволяє його використовувати для всіх задач обліку на підприємстві, а лише як додатку що спрощує процес отримання інформації про складські операції. Тому подальший розвиток розробленої системи може полягати у зміні платформи та розробці веб-додатку який буде мати більш функціональний інтерфейс та окрім складського обліку дозволятиме виконувати задачі бухгалтерського обліку.

ВИСНОВКИ

Основним результатом роботи є розроблена система обліку та аналізу використання запасів сільськогосподарського підприємства що реалізована як телеграм бот написаний на мові програмування Python з підключеною базою даних SQLite та можливістю візуалізації інформації з використанням бібліотеки matplotlib.

На поточний момент система дозволяє вводити інформацію про кількість внесеного на склад ресурсу (наприклад, пшениця, ячмінь і т.д.), номер складу на який вноситься даний ресурс, а також вид операції – тобто видача зі складу чи внесення на склад. Дата здійснення операції автоматично вставляється у базі даних та не може бути скоригована користувачем. Крім того, кінцевий користувач не має права змінювати чи видаляти вже зафіксовану інформацію, так як користується системою через інтерфейс телеграм-бота і не має доступу до файлів бази даних. Використання унікального коду користувача телеграм дозволяє фільтрувати сторонніх користувачів та розділяти рівні доступу для різних користувачів – тобто звичайний користувач не може бачити підсумкові дані або звіти.

Основною задачею системи є не заміна, а лише доповнення існуючої системи бухгалтерської звітності та заміни частини операцій складського обліку що раніше виконувались у вигляді ведення паперових журналів обліку або заповнення електронних таблиць. Перевага використання телеграм-бота у якості основного інтерфейсу користувача полягає у поширеності даного месенджеру та легкості розгортання системи – для початку використання достатньо лише мати пристрій з встановленим месенджером. Програма може запускатись з будь-якого сервера підключеного до мережі інтернет. На відміну від мобільних додатків, які на сьогоднішній день також є досить популярними рішеннями для оптимізації та автоматизації бізнес-процесів невеликих підприємств, телеграм-бот має менший функціонал але при цьому простіше у розгортанні та використанні.

Побудована у SQLite база даних може використовуватись як сховище даних для розробленого телеграм-бота а також до неї може отримати прямий доступ керівник або бухгалтер, якщо необхідно експортувати таблиці з даними про складські операції та використати їх для заповнення бухгалтерської звітності.

Також було проведено тестування телеграм боту. В результаті проведення позитивного та негативного тестування було використано відповідно коректні та некоректні дані які було успішно та адекватно оброблено системою.

Майбутній розвиток побудованої системи може полягати у розширенні функціоналу та можливості використання додатковими категоріями користувачів – наприклад, контрагентами чи іншими категоріями користувачів окрім комірника, керівника та бухгалтера.

Таким чином, поставлене у роботі завдання розробки системи обліку та аналізу використання запасів сільськогосподарського підприємства можна вважати повністю виконаним. Розроблена система протестована та може використовуватись для виконання реальних задач ведення складського обліку та дозволяє підвищити ефективність роботи підприємства. Основною цільовою аудиторією системи є невеликі аграрні підприємства, які розміщують продукцію на власних складах та на даний момент не використовують засоби автоматизації складського обліку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Національне положення (стандарт) бухгалтерського обліку 9 "Запаси" 2. Аналіз господарської діяльності / За ред. Барабаш Н.. – Київ: КНЕУ, 2002. – 552 с.

3. Голов С.Ф. Управлінський облік. Підручник. – Київ : Лібра, 2003. – 704 с

4. Грабова Н. М. Облік основних господарських операцій в бухгалтерських проводках : навчальний посібник [для студ. вищ. Навчальних закладів] / Н. М. Грабова, Ю. Г. Кривоносов. – К. : АСК, 2006. – 416 с.

5. Івахненко С. В. Інформаційні технології в організації бухгалтерського обліку та аудиту : [навч. посіб.] / С. В. Івахненко. – К. : Знання-Прес, 2003. – 349 с.

6. Рибалко О. М. Вдосконалення обліку виробничих запасів / О. М. Рибалко, О. В. Болдуєва // Держава та регіони. Серія : Економіка та підприємництво. – 2008. – № 3. – С. 210 – 215.

7. Цвентарна В. Питання ведення обліку виробничих запасів на підприємстві / В. Цвентарна. [Електронний ресурс]. – Режим доступу : http://sophus.at.ua

8. Hunter J. D. Matplotlib: A 2D Graphics Environment // Computing in Science and Engineering — AIP Publishing, 2007. — Vol. 3, Iss. 1. — P. 766. — ISSN 1521-9615; 1558-366X — doi:10.1109/MCSE.2007.55

9. Robert H. Is Python pass-by-reference or pass-by-value? [Електронний ресурс] / Heaton Robert. – 2014. – Режим доступу до ресурсу: https://robertheaton.com/2014/02/09/pythons-pass-by-object-reference-asexplained-by-philip-k-dick/.

10. Frederik Lundh. Python Standard Library. O'Reilly & Associates, 2001

11. Ivan Van Laningham. Teach Yourself Python in 24 Hours. Sams, 2000

12. Rashi Gupta. Making use of Python. Wiley, 2002

13. Alan Gauld. Learn to Program Using Python: A Tutorial for Hobbyists, Self-Starters, and Those Who Want to Learn the Art of Programming. Addison-Wesley Professional, 2001

14. M. Lutz. Learning Python, Fifth Edition. – O'Reilly Media. – Sebastopol, 2013. URL: https://bit.ly/2Qno5z9

15. Lisa Tagliaferri. How To Code in Python 3. – DigitalOcean, New York City, New York, USA. URL: https://do.co/2XR7hc8

16. Креневич А.П. Python у прикладах і задачах. Частина 2. Об'єктно-орієнтоване програмування. Навчальний посібник – К.: ВПЦ "Київський Університет", 2020. – 152 с

17. Мнушка О.В. Об'єктно-орієнтоване програмування мовою Python: навчальний посібник для студентів напрямів підготовки 122 Комп'ютерні науки та 121 Інженерія програмного забезпечення / О.В. Мнушка, В.М. Савченко, О.Б. Маций – Х.: ХНАДУ, 2021. – 228 с. ISBN 978-617-7912-88-9

18. Data model. – URL: https://docs.python.org/3.8/reference/datamodel.html

19. PEP 0 – Index of Python Enhancement Proposals (PEPs).– URL: https://www.python.org/dev/peps/

20. Links to Python related information in Ukranian. – URL: https://wiki.python.org/moin/UkranianLanguage

21. Stackoverflow. – URL:https://stackoverflow.com/

22. Костюченко А.О. Основи програмування мовою Python: навчальний посібник. Ч.: ФОП Баликіна С.М., 2020. 180 с.

23. Python 3.8.0 documentation. URL: https://docs.python.org/3/ (дата звернення: 12.10.2023).

24. Матвійчук С.В., Жуковський С.С. Практикум програмування Python / C++ на e-olymp.com (збірник задач з рекомендаціями до їх розв'язання). Житомир: Вид-во ЖДУ ім. І. Франка, 2019. 232 с.

25. Програмування на мові Python (3.x). Початковий курс. URL: https://sites.google.com/site/pythonukr/ (дата звернення: 12.10.2023).

26. Програмування числових методів мовою Python / А. В.Анісімов, А.Ю. Дорошенко, С. Д. Погорілий, Я. Ю. Дорогий. – Київ: Видавничо-поліграфічний центр "Київський університет", 2014. – 640 с. – (Підручник).

27. Downey A., Elkner J., Meyers Ch. How to Think Like a Computer

Scientist: Learning with Python. - Wellesley, Massachusetts: Green Tea Press, 2002. - 290 pp.

28. Tutorialspoint / Python – [Електронний ресурс]. – Режим доступу: http://www.tutorialspoint.com/python/

29. Python's documentation, tutorials, and guides are constantly evolving. – [Електронний ресурс]. – Режим доступу: https://www.python.org/doc/

30. Основи програмування. Python. Частина 1 [Електронний ресурс]: підручник для студ. спеціальності 122 "Комп'ютерні науки", спеціалізації "Інформаційні технології в біології та медицині" / А. В. Яковенко ; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2018. – 195 с.

Додаток А.

```
import sqlite3
import telebot
from telebot import types  # для типів
import config # bot api, sqlite config



bot = telebot.TeleBot(config.token)

conn = sqlite3.connect('db/database.db', check_same_thread=False)
cursor = conn.cursor()

def db_table_val(user_id: int, resourse_quantity: int, resource_id: int , storage_id: int):
        cursor.execute('INSERT INTO debit (user_id, resourse_quantity, resource_id, storage_id) VALUES (?, ?, ?, ?)', (user_id, resourse_quantity, resource_id, storage_id))
        conn.commit()

def db_table_val2(user_id: int, resourse_quantity: int, resource_id: int , storage_id: int):
        cursor.execute('INSERT INTO credit (user_id, resourse_quantity, resource_id, storage_id) VALUES (?, ?, ?, ?)', (user_id, resourse_quantity, resource_id, storage_id))
        conn.commit()

@bot.message_handler(commands=['start'])
def start(message):
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    btn1 = types.KeyboardButton("👋 Отримати звіт")
    btn2 = types.KeyboardButton("❓ Складські операції")
    markup.add(btn1, btn2)
    bot.send_message(message.chat.id,
                     text="Вітаю, {0.first_name}! Це бот для обліку складських запасів".format(
```

```
message.from_user), reply_markup=markup)


@bot.message_handler(content_types=['text'])
def func(message):
    a=0
    k=0
    m=0
    if (message.text == "👋 Отримати звіт"):
        cursor.execute("SELECT * FROM debit")
        print(cursor.fetchall())
        bot.send_message(message.chat.id, text="Звіт виведено в консоль)")

    elif (message.text == "❓ Складські операції"):
        markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
        btn1 = types.KeyboardButton("Внести на склад")
        btn2 = types.KeyboardButton("Взяти зі складу")
        back = types.KeyboardButton("Повернутись в головне меню")
        markup.add(btn1, btn2, back)
        bot.send_message(message.chat.id, text="Оберіть операцію",
reply_markup=markup)

    elif (message.text == "Внести на склад"):
        bot.send_message(message.chat.id, "Введіть через пробіл кількість(кг)
номер продукції номер складу")


    elif message.text == "Взяти зі складу":
        bot.send_message(message.chat.id, text="Введіть через пробіл кількість з
мінусом(кг)  номер продукції номер складу")


    elif (message.text == "Повернутись в головне меню"):
        markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
        button1 = types.KeyboardButton("👋 Отримати звіт")
```

```
        button2 = types.KeyboardButton(" ? Складські операції")
        markup.add(button1, button2)
        bot.send_message(message.chat.id, text="Ви повернулись в головне меню",
reply_markup=markup)
    else:

        b = message.text.split()
        a=int(b[0])
        k=int(b[1])
        m = int(b[2])
        if (a>0):
            us_id = message.from_user.id
            res_quantity = a
            res_id=k
            stor_id=m
            db_table_val(user_id=us_id,                resourse_quantity=res_quantity,
resource_id=res_id, storage_id=stor_id)
        elif (a<0):
            us_id = message.from_user.id
            res_quantity = -a
            res_id = k
            stor_id=m
            db_table_val2(user_id=us_id,               resourse_quantity=res_quantity,
resource_id=res_id, storage_id=stor_id)
        print(b[0])
        print(b[1])
        print(b[2])
        bot.send_message(message.chat.id, text="Введіть операцію")



bot.polling(none_stop=True)
```