ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка програмного забезпечення оптимізації агрономічних процесів»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Арсеній НИКИТЕНКО
(підпис)

Виконав: здобувач вищої освіти групи ППЗ-51
Арсеній НИКИТЕНКО

Керівник: Володимир САДОВЕНКО
к.ф-м.н., доцент

Рецензент: _____

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____Ірина ЗАМРІЙ
« ____» _____2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Никитенку Арсенію Олексійовичу

1. Тема кваліфікаційної роботи: «Розробка програмного забезпчення оптимізації агрономічних процесів»
Керівник кваліфікаційної роботи професор кафедри ІПЗ, к.ф.-м.н., доцент Володимир САДОВЕНКО
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025 р. № 56.

2. Строк подання кваліфікаційної роботи «02 червня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: у процесі виконання кваліфікаційної роботи було використано наступні офіційні джерела, що слугували базою для реалізації програмного продукту: офіційна документація C#, документація з використання Visual Studio , документація по .NET, офіційна документація SQLite

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз існуючих підходів до автоматизації агрономічних процесів.

2. Визначення функціональних та нефункціональних вимог до системи.
3. Проєктування архітектури програмного забезпечення.
4. Реалізування системи нагадувань у середовищі C# (Windows Forms).
5. Проведення тестування програмного продукту.
6. Оцінення результатів та формулювання перспективи розвитку програми.

5. Перелік графічного матеріалу: *презентація*

1. Технічні вимоги до системи нагадувань
2. Структурна схема архітектури програмного забезпечення
3. Інтерфейс користувача системи нагадувань про агрономічні роботи
4. ER-діаграма структури даних
5. Схема трирівневої архітектури програмного забезпечення
6. Результати тестування у вигляді графіка
7. Набір скріншотів взаємодії з користувачем

6. Дата видачі завдання «25» лютого 2025 р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	25.02-06.03.2025	
2	Аналіз та дослідження існуючих аналогів	07.03-13.03.2025	
3	Дослідження матеріалів для аналізу існуючих рішень самостійної роботи здобувача	14.03-20.03.2025	
4	Аналіз переваг та недоліків існуючих програмних засобів	21.03-25.03.2025	
5	Дослідження функціональних та нефункціональних вимог до застосунку	26.03-03.04.2025	
6	Проєктування інтерфейсу користувача	04.04-18.04.2025	
7	Тестування	19.04-28.04.2025	
8	Оформлення роботи: вступ, висновки, реферат	29.04-05.05.2025	
9	Розробка демонстраційних матеріалів	06.05-10.05.2025	
10	Попередній захист роботи	12.05-16.05.2025	

Здобувач вищої освіти _____ Арсеній НИКИТЕНКО

(підпис)

Керівник
кваліфікаційної роботи _____ Володимир САДОВЕНКО

(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 56 стор., 9 рис., 12 табл., 19 джерел.

Мета роботи – розробка програмного забезпечення для нагадування про агрономічні роботи, яке дозволяє автоматизувати планування сезонних завдань, зменшити ризик пропуску важливих агрономічних подій та підвищити ефективність управління агровиробничими процесами.

Об'єкт дослідження – процеси організації та автоматизації виконання сезонних агрономічних робіт.

Предмет дослідження – програмне забезпечення системи нагадувань про агрономічні роботи, зокрема його архітектура, функціональні можливості, інтерфейс та ефективність використання.

Короткий зміст роботи: У роботі проаналізовано сучасні програмні рішення у сфері аграрної автоматизації та обґрунтовано необхідність створення простої автономної системи нагадувань для агрономічних задач. Сформульовано функціональні та нефункціональні вимоги до системи. Побудовано архітектуру програмного продукту з використанням C# і SQLite. Реалізовано інтерфейс користувача, базу даних, функції створення, редагування та нагадування про аграрні завдання. Проведено тестування, експериментальну перевірку та порівняльний аналіз з аналогами.

Ключові слова: агрономія, автоматизація, нагадування, C#, Windows Forms, аграрні процеси, база даних, планування, інтерфейс, тестування.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ПЗ – програмне забезпечення

БД – база даних

UI – інтерфейс користувача (User Interface)

C# – мова програмування C-Sharp

SQLite – вбудована реляційна система керування базами даних

.NET – програмна платформа від Microsoft для розробки програм

ER-діаграма – діаграма «сутність–зв’язок» (Entity-Relationship)

CRUD – базові операції з даними: створення (Create), читання (Read), оновлення (Update), видалення (Delete)

Windows Forms – бібліотека для створення графічного інтерфейсу в середовищі .NET

UX – досвід користувача (User Experience)

ВСТУП

У нинішньому сільському господарстві ефективність агрономічних процесів відіграє ключову роль у досягненні стабільних врожаїв і зменшенні витрат. Планування та своєчасне виконання агрономічних робіт — запорука продуктивного землеробства. Проте в багатьох малих та середніх господарствах управління цими процесами досі ведеться вручну, що призводить до втрати часу, порушення технологічних строків і зниження врожайності.

У зв’язку з цим автоматизація планування та нагадування про важливі етапи обробки ґрунту, посіву, внесення добрив та збору урожаю набуває особливої актуальності. Створення програмного забезпечення, яке дозволяє агроному не пропустити важливу роботу, значно покращує організацію виробничого процесу.

Метою дипломної роботи є розробка програмного забезпечення у вигляді системи нагадувань по агрономічним роботам, за допомогою C# та Windows Forms.

В процесі дослідження вирішувалися наступні завдання:

- Проаналізувати існуючі підходи до автоматизації агрономічних процесів.
- Визначити функціональні та нефункціональні вимоги до системи.
- Спроєктувати архітектуру програмного забезпечення.
- Реалізувати систему нагадувань у середовищі C# (Windows Forms).
- Провести тестування програмного продукту.
- Оцінити результати та сформулювати перспективи розвитку програми.

Об’єктом дослідження є процес організації та планування агрономічних робіт у фермерському господарстві.

Предметом дослідження є методи та засоби розробки програмного забезпечення для автоматизації планування агрономічних дій і формування нагадувань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Особливості організації агрономічних робіт

Організація агрономічних робіт у сільському господарстві є складним багатоступеневим процесом, що охоплює планування, виконання та контроль за агротехнічними заходами протягом усього вегетаційного періоду культур. Основними типами таких робіт є:

• Обробка ґрунту (оранка, культивація);

• Посівні роботи;

• Внесення мінеральних та органічних добрив;

• Обприскування засобами захисту рослин;

• Зрошення (при потребі);

• Збирання врожаю.

Кожен із цих етапів має суворі терміни виконання, залежні від кліматичних умов, типу ґрунтів, обраної культури та регіону. Невчасне виконання хоча б однієї з операцій може негативно вплинути на врожайність, якість продукції та загальну рентабельність господарства.

У великих агрохолдингах для організації робіт часто використовуються складні автоматизовані системи управління господарством (AgriTech, ERP-системи, GPS-моніторинг техніки тощо). Натомість у малих і середніх фермерських господарствах найчастіше застосовуються простіші інструменти — блокноти, Excel-таблиці або навіть усна фіксація. Це значно ускладнює планування та контроль, особливо в умовах сезонного навантаження.

Особливістю агрономічної діяльності є також потреба враховувати:

• біологічні особливості культур (період проростання, стійкість до захворювань);

• погодні умови (опади, температура, вологість ґрунту);

• ресурсні можливості господарства (техніка, працівники, засоби захисту).

Усе це створює високу потребу у простому, доступному програмному рішенні, яке дозволить:

• фіксувати поля, культури та їхню історію;

• планувати строки проведення робіт;

• отримувати нагадування про наближення важливих подій;

• забезпечувати простий інтерфейс для агронома без потреби в глибоких IT-знаннях.

Таким чином, використання спеціалізованого програмного забезпечення дозволяє зменшити людський фактор у плануванні, уникнути пропуску ключових робіт та оптимізувати агропроцеси навіть у малих господарствах.

1.2 Проблеми та недоліки існуючих рішень

Незважаючи на розвиток цифрових технологій у сільському господарстві, значна частина малих та середніх агропідприємств продовжує користуватися застарілими або неефективними підходами до планування агрономічних робіт. Більшість сучасних рішень мають низку недоліків, що ускладнюють їх впровадження та використання в реальних умовах.

1. Відсутність спеціалізованих інструментів для малих господарств

Більшість доступних рішень — це складні ERP-системи для агрохолдингів (наприклад, Cropio, Agroholding ERP, 1С-Агро), які потребують значних ресурсів для впровадження: кваліфікованого персоналу, технічної підтримки, постійного оновлення програмного забезпечення. Малим фермерським господарствам такі системи часто не по кишені як фінансово, так і організаційно. [8]

2. Низький рівень автоматизації у малих фермерів

У багатьох випадках агрономи або керівники фермерських господарств ведуть записи у блокнотах або Excel-файлах. Це призводить до частих помилок, втрати інформації та неможливості централізовано відстежувати статус виконання робіт.

За оцінками, лише близько 15% площ українських с/г угідь використовують технології точного землеробства (precision agriculture), а ринок цифрових агротехнологій становить приблизно 60–70 млн [1]. При цьому 30% фермерів в Україні використовують навіть базові елементи цифровізації (наприклад, GPS-моніторинг, точне внесення ЗЗР) .

3. Відсутність функцій нагадування

Навіть якщо господарство використовує базове ПЗ (наприклад, таблиці або просту CRM-систему), у ньому зазвичай відсутні функції автоматичного нагадування про строки виконання польових робіт. Це створює ризик втрати врожайності через пропущені або затримані операції.

4. Складність інтерфейсу

Існуючі програми, навіть ті, що доступні онлайн, часто перевантажені функціями та складні в освоєнні для людей без технічної освіти. Аграрію важливо отримати простий та інтуїтивно зрозумілий інструмент, який не потребує тривалого навчання.

5. Недостатня адаптація до конкретних культур

Часто в ПЗ немає гнучкого механізму формування індивідуального плану робіт залежно від типу культури, регіону та кліматичних умов. Типові шаблони не враховують реальні строки і особливості вирощування конкретної культури.

Незважаючи на швидкі темпи розвитку цифрової агросфери, багато малих та середніх фермерських господарств не мають доступу до ефективних цифрових інструментів. Нижче наведено дані, які це підтверджують:

В Європі менш ніж 10% дрібних фермерських господарств застосовують технології точного землеробства, навіть для базових операцій (обприскування — ~4,6 %, точне удобрення — ~17,7 %) . Основні причини: висока вартість ADS-платформ, нестача цифрового досвіду та складність пристосування до малих полів[2].

6. Велика частка фермерів користуються Інтернетом — але не ПЗ

В Україні приблизно 92 % аграріїв, тобто близько 2,8 млн осіб, використовують Інтернет для пошуку інформації — однак рецептори цифрових рішень на цьому обмежуються, а спеціальних інструментів використовують значно менше [5].

Отже, технічні рішення недостатньо адаптовані під малі господарства — відсутня гнучкість, доступність та ефективність саме для невеликих земельних ділянок. Висока вартість і складність — потрібні цифрові навички, які часто відсутні у аграріїв дрібного бізнесу, а інформаційний потенціал не використовуються повною мірою — багато хто залишається на рівні пошуку інформації, а не безпосереднього використання ПЗ.

Ці фактори підтверджують необхідність створення доступного, простого та адаптованого рішення, яке дозволить аграріям отримати інструмент для планування, обліку та нагадувань про агрономічні роботи без великих витрат і складної установки.

1.3 Огляд програмних засобів для автоматизації агросектору

Наразі у аграрному секторі існує велика кількість програмних рішень, спрямованих на автоматизацію та оптимізацію аграрних процесів. Однак більшість із них орієнтовані на середні та великі агропідприємства. У цьому підрозділі розглянемо основні види і приклади таких програм, а також проаналізуємо їхні переваги та обмеження для малих господарств.

1. АгроERP-системи

Це повнофункціональні системи управління агробізнесом, які охоплюють бухгалтерію, контроль техніки, планування сівозмін, облік ЗЗР (засобів захисту рослин), логістику тощо.

Приклади:

Cropio — це інноваційна система дистанційного моніторингу та управління аграрними земельними ресурсами, яка активно використовується в понад 50 країнах світу, зокрема в Україні. Вона дозволяє агрономам, менеджерам та фермерам контролювати стан полів у режимі реального часу завдяки супутниковим знімкам і вбудованій аналітиці. Програма складається із трьох базових модулів: стан посівів, агрооперації (планування робіт), телематика (GPS моніторинг)[6].

Основні функції:

• Моніторинг вегетації на полях через супутникові NDVI-індекси;

• Контроль техніки на полі (GPS-трекінг);

• Формування карт полів, аналіз зон урожайності;

• Планування польових робіт та контроль їх виконання.

Переваги:

• Працює в режимі онлайн;

• Підтримує інтеграцію з технікою;

• Детальна аналітика по кожному полю.

Недоліки:

• Висока вартість ліцензії;

• Надлишкова складність для малих фермерів;

• Потрібна стабільна якісна інтернет-мережа на місцях.

Agro Office, Soft.Farm — українські системи з можливістю GPS-моніторингу техніки, обліку робіт, інтеграції з 1С.

Agro Office — програмний комплекс української розробки, що надає інструменти для ведення агропідприємства: від створення сівозміни до обліку витрат на гектар. Він дозволяє вести земельний облік, контролювати витрати ресурсів і планувати технологічні операції.

Основні функції:

• Складання технологічних карт;

• Геоінформаційна система для моніторингу полів;

• Облік пального, витрат ЗЗР і добрив;

• Інтеграція з 1С та GPS-обладнанням.

Soft.Farm — ще одна українська система для комплексного управління агробізнесом. Система поєднує супутниковий моніторинг, контроль техніки,

картографію, облік сільгоспробіт[10].

Основні функції:

• Візуалізація меж полів, карт урожайності;

• Ведення журналу агронома;

• Відстеження шляхів руху техніки;

• Облік робочого часу механізаторів.

Перевагами обох систем є адаптивність до українських реалій, підтримання української мови та інтеграція з GPS та обліковими системами.

Недоліками є складний інтерфейс для не IT-користувача, потрібне навчання персоналу та не завжди є мобільна версія з простим нагадуванням про роботи.

OneSoil, EOS Crop Monitoring — платформи з інтеграцією супутникових даних, прогнозів урожайності.

OneSoil — безкоштовна платформа для аграріїв, яка пропонує супутниковий моніторинг та аналітику стану полів [12].

Основні функції:

• Автоматичне виявлення меж полів;

• NDVI-аналіз для виявлення проблемних ділянок;

• Рекомендації щодо диференційованого внесення добрив;

• Порівняння продуктивності полів між роками.

Розробка української компанії EOS Data Analytics. Це веб-платформа з високою точністю аналітики супутникових знімків [9].

Основні функції::

• Оцінка стану посівів у реальному часі;

• Погодні дані, карта вологості ґрунту;

• Прогнозування ризиків (посуха, заморозки, бур’яни);

• Система оповіщень для агронома.

Перевагами обох платформ є безкоштовний базовий доступ, зручний інтерфейс, мобільна версія та вони не потребують установки складного обладнання.

Недоліками є орієнтованість більше на аналітику, ніж на нагадування чи облік робіт, не реалізована функція детального планування календаря польових заходів та немає гнучкої кастомізації під конкретне господарство.

2. Мобільні агрододатки

Мобільні аграрні додатки стають дедалі популярнішими серед фермерів завдяки простоті використання, доступності та зручності. Вони дозволяють отримувати оперативну інформацію, вести польовий журнал, формувати календар робіт, а також отримувати сповіщення щодо погодних умов чи оптимального часу обробки культур. Найбільш популярними серед таких додатків є AgriApp, Agroptima, Сільський експерт.

AgriApp — мобільний застосунок, популярний в Індії, орієнтований на малих фермерів. Він надає доступ до агрономічних порад, прогнозів погоди, рекомендацій щодо ЗЗР (засобів захисту рослин), а також дозволяє комунікувати з агроконсультантами.

Основні функції:

• Каталог культур з рекомендаціями щодо обробітку;

• Оцінка стану рослин на основі фото;

• Поради щодо вибору добрив, пестицидів;

• Прогноз погоди, календар заходів.

Переваги:

• Простий інтерфейс;

• Високий рівень локалізації для регіонів Індії;

• Часто оновлюється з новими порадами.

Недоліки:

• Відсутня українська або навіть англійська локалізація в повному обсязі;

• Зосереджений на специфічних культурах, характерних для Азії;

• Немає функції персональних нагадувань.

Agroptima — іспанський агрододаток, розроблений спеціально для обліку польових робіт. Дає змогу фермерам вести історію робіт, витрат, культур, створювати інтерактивні карти полів.

Основні функції:

• Журнал робіт (час, площа, вид обробки);

• Облік ресурсів: добрива, паливо, праця;

• Експорт звітів у PDF або Excel;

• Відстеження витрат за ділянками.

Переваги:

• Інтуїтивно зрозумілий інтерфейс;

• Доступний мобільний додаток для Android та iOS;

• Можна використовувати офлайн із подальшою синхронізацією.

Недоліки:

• Платна підписка на повний функціонал (від ~100 €/рік);

• Немає україномовної підтримки;

• Немає нагадувань по роботах — лише облік.

Сільський експерт — український мобільний додаток, орієнтований на фермерів, агрономів і консультантів. Він містить довідкову інформацію про культури, хвороби, добрива, пестициди та техніку.

Основні функції::

• Каталог сільгоспкультур із порадами з вирощування;

• Інформація про шкідників, хвороби, препарати;

• Рекомендації щодо сівозміни, захисту рослин;

• Погода, новини аграрного сектору.

Переваги:

• Повністю україномовний інтерфейс;

• Безкоштовний доступ;

• Створений із урахуванням локальної агроспецифіки.

Недоліки:

• Відсутність системи планування і нагадувань;

• Орієнтований більше на довідкову інформацію, а не на облік або контроль робіт.

3. Власноруч створені рішення

Незважаючи на наявність великої кількості спеціалізованих аграрних програм, значна частина дрібних фермерів продовжує користуватись підручними, універсальними інструментами, які не є агроспецифічними. Це можуть бути таблиці в Excel, Google Sheets, нотатки у смартфоні, Google Calendar, месенджери або навіть звичайні паперові блокноти.

Такі рішення формуються індивідуально кожним користувачем під свої потреби, не потребують ліцензій чи особливих навичок програмування, але мають суттєві обмеження.

Excel, Google Sheets

Найпоширеніший спосіб обліку — це електронні таблиці, в яких фермери ведуть облік посівів, обробок, витрат, строків виконання робіт тощо.

Переваги:

• Гнучкість: можна створити будь-яку структуру даних;

• Можливість автоматизації через формули та макроси;

• Безкоштовний доступ через Google Диск;

• Працює офлайн (у випадку з Excel).

Недоліки:

• Відсутність системи нагадувань або сповіщень;

• Складність структурування великої кількості даних;

• Немає візуального інтерфейсу для перегляду календаря або хронології подій;

• Високий ризик помилок при ручному введенні.

Google Calendar / телефонні нагадування

Деякі фермери вносять події у Google Calendar або встановлюють собі нагадування на телефоні через стандартні додатки. Наприклад, "Посіяти кукурудзу — 15 квітня", "Обробити пшеницю гербіцидом — 30 травня" тощо.

Переваги:

• Можливість встановлення точного часу нагадування;

• Синхронізація між пристроями;

• Доступність із будь-якого смартфона.

Недоліки:

• Дані не групуються по полях, культурах або сезонах;

• Відсутність журналу виконаних робіт;

• Немає автоматичного формування робіт за шаблоном;

• Не зберігається історія агрономічних дій.

Нотатки, блокноти, месенджери

У багатьох малих фермерів процес планування робіт ще досі ведеться на папері або в стандартному застосунку "Нотатки" на телефоні. Деякі навіть записують

важливі дати у Viber або Telegram, надсилаючи повідомлення самі собі.

Переваги:

• Швидко та просто;

• Не потребує інтернету або спеціальних знань;

• Мінімум зусиль на ведення записів.

Недоліки:

• Дуже обмежена функціональність;

• Високий ризик втрати або забутого запису;

• Відсутність можливості пошуку, фільтрації або аналітики;

• Важко систематизувати великі обсяги даних.

Отже, було встановлено, що на ринку агрософту існує очевидна нішевість — відсутність доступної, простої у використанні програми для надання нагадувань про польові роботи, яка б враховувала строки сівби, обробок, внесення добрив тощо.

Це особливо важливо в умовах сезонності, багатозадачності та дефіциту часу у фермерів.

У таблиці 1.1 наведено порівняння аграрних програмних засобів з метою систематизованого аналізу функціональних можливостей, зручності використання, доступності та орієнтованості на кінцевого користувача серед популярних рішень у сфері аграрного програмного забезпечення.

Таблиця 1.1

Порівняльна таблиця аграрних програмних засобів

Характеристика / Рішення	**Cropio, AgroOffice, EOS Crop Monitoring**	**Agroptima, AgriApp, Сільський експерт**	**Власноручні створені (Excel, календар)**	**Запропонована програма (нагадування)**
Цільова аудиторія	Великі агрофірми	Середні / дрібні фермери	Дрібні фермери	Дрібні та середні фермери
Наявність системи нагадувань	✕	✕	✓	✓✓
Облік робіт / польовий журнал	✓✓	✓	✕ / частково	✓
Простота використання	✕	✓	✓✓	✓✓
Працює без інтернету	✕	Частково	✓	✓
Можливість локалізації / українська мова	Частково / відсутня	Частково / ✓	✓	✓✓
Автоматизація процесів	✓✓	✕ / частково	✕	✓
Ціна використання	$$$	$ / безкоштовно	Безкоштовно	Безкоштовно / умовно безкоштовно
Збереження історії обробок, культур, полів	✓✓	✓ / ✕	✕	✓
Можливість адаптації під конкретне господарство	✕	Частково	✓	✓✓

Умовні позначення:

✓✓ — повна підтримка функції або перевага
✓ — часткова підтримка або базова реалізація
✕ — відсутність функції
$ / $$$ — вартість: недорого / дорого

Багатофункціональні сервіси, такі як Cropio, AgroOffice, EOS Crop Monitoring або OneSoil, орієнтовані насамперед на великі агропідприємства. Вони пропонують комплексну аналітику, інтеграцію з супутниковими знімками, GPS-моніторинг техніки та облік витрат. Проте їх складність, висока вартість і перенасиченість функціями часто ускладнюють використання для дрібних і середніх фермерів.

Мобільні агрододатки, такі як Agroptima, AgriApp, або український Сільський експерт, мають простіший інтерфейс, надають довідкову інформацію, дозволяють вести журнал робіт або діагностувати хвороби рослин. Однак у більшості таких додатків відсутній ключовий функціонал — система автоматизованих нагадувань про агрономічні роботи, які прив’язані до конкретних культур, полів або фаз розвитку рослин.

Власноруч створені рішення (Excel, Google Calendar, нотатки) лишаються найпопулярнішим варіантом для малих фермерів через доступність і простоту. Проте вони не дозволяють ефективно організовувати робочі процеси, не підтримують автоматизацію й часто є джерелом помилок або втрат даних.

Таким чином, розробка системи нагадувань по агрономічним роботам, яка матиме базу агрономічних подій, інтеграцію з календарем та можливість зручного додавання полів і культур — є обґрунтованим і актуальним завданням, що вирішує реальну практичну проблему агросектору.

1.4 Вимоги до програмного забезпечення

На основі аналізу предметної області, сучасного стану ринку аграрного програмного забезпечення та виявлених потреб користувачів, можна сформулювати основні вимоги до майбутньої системи нагадувань про агрономічні роботи.

Програма повинна вирішувати завдання оперативного планування сільськогосподарських робіт, формування індивідуального календаря робіт для конкретних полів і культур, а також автоматичного інформування користувача про

заплановані події.

Функціональні вимоги:

1. Додавання полів з базовими характеристиками (назва, площа, місцезнаходження).

2. Прив’язка культур до полів (наприклад: кукурудза, пшениця, соя тощо).

3. Планування агрономічних подій: сівба, обприскування, внесення добрив, полив, збір урожаю.

4. Налаштування періодичності подій.

5. Система нагадувань: автоматичне сповіщення про майбутні роботи (через інтерфейс або інші засоби).

6. Календарний перегляд всіх подій (графік робіт по днях, тижнях, місяцях).

7. Редагування, перенесення або скасування подій.

Нефункціональні вимоги:

1. Зручний і простий графічний інтерфейс користувача.

2. Підтримка української мови.

3. Можливість роботи без підключення до Інтернету (офлайн-режим).

4. Невибагливість до ресурсів системи (робота на ПК середньої потужності).

5. Можливість збереження даних локально або в файлі (без складних баз даних).

Далі наведено таблицю таблицю, яка містить ключові технічні параметри, які були обрані для реалізації програмного забезпечення. Вони визначають технологічне підґрунтя розробки, а також впливають на архітектуру, вибір

інструментів та подальше тестування.

Таблиця 1.2

Технічні вимоги

Категорія	Значення
Платформа розробки	.NET / C#
Тип застосунку	Настільна програма (Desktop)
Середовище виконання	Windows
Формат зберігання даних	JSON / XML або SQLite
Тип інтерфейсу	Windows Forms
Розробницьке середовище	Visual Studio

Технічні вимоги сформовані з урахуванням доступності, сумісності з робочими станціями користувачів та спрощення підтримки.

Таким чином, програма має забезпечити базову автоматизацію планування агрономічних робіт, бути інтуїтивно зрозумілою, не вимагати складного навчання або підключення до хмарних сервісів. Такий підхід дозволить ефективно використовувати систему у невеликих фермерських господарствах та фермерськими консультантами.

У цьому розділі було проведено аналіз предметної області агрономічного планування та обробки даних. Було виявлено основні проблеми, з якими стикаються дрібні та середні фермери у процесі ведення агрономічних робіт, зокрема: відсутність систематичного обліку, неузгодженість у плануванні робіт, недостатня автоматизація процесів та ігнорування важливих етапів догляду за культурами.

Проведено огляд існуючих програмних рішень: професійних агроплатформ (Cropio, EOS Crop Monitoring, AgroOffice), мобільних додатків (AgriApp,

Agroptima, OneSoil), а також саморобних рішень на базі Excel чи календарів. Визначено їхні переваги та недоліки у контексті цільової аудиторії — дрібних і середніх фермерських господарств.

На основі порівняльного аналізу встановлено, що жодне з готових рішень не забезпечує та зручну систему нагадувань про агрономічні роботи, адаптовану до потреб невеликого господарства. Саме це стало підґрунтям для розробки власного програмного забезпечення, яке буде враховувати актуальні вимоги користувачів.

Також у підрозділі 1.4 було визначено основні функціональні, нефункціональні та технічні вимоги до майбутньої програми. Це дозволяє перейти до етапу проєктування програмного рішення, що буде представлено в наступному розділі.

2 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Постановка та архітектура задачі розробки

Сучасне сільське господарство потребує ефективних цифрових інструментів для організації, планування та контролю агрономічних процесів. Однією з найпоширеніших проблем, з якою стикаються фермери та агрономи, є неузгодженість у виконанні сезонних робіт: від посіву до збирання врожаю. Часто через людський фактор втрачається важлива інформація про строки внесення добрив, захисту рослин, зрошення тощо. Відсутність нагадувань або централізованого обліку може призводити до зниження врожайності, перевитрати ресурсів, а в деяких випадках — до загибелі культур.

З огляду на це, актуальним є розробка простого та ефективного програмного забезпечення, яке б забезпечувало автоматизоване нагадування про агрономічні роботи відповідно до плану посівів та особливостей кожного поля.

Мета розробки програмного забезпечення:
Реалізація настільної системи нагадувань для агрономічних процесів із підтримкою локального збереження даних, нагадування за термінами та простим інтерфейсом.

Об'єкт проєктування:
Програмне забезпечення для автоматизації нагадувань про агрономічні роботи.

Предмет розробки:
Функціональні, структурні та візуальні аспекти системи нагадувань.

Завдання:

- Розробити архітектуру системи.
- Створити інтерфейс користувача.
- Реалізувати механізм нагадувань.
- Забезпечити збереження даних та автономність.

- Провести тестування розробленої системи.

Таким чином, розробка даного ПЗ має на меті підвищити організованість і ефективність агрономічної діяльності, зменшит ризик пропущених робіт та забезпечити підтримку фермера у щоденних завданнях.

Проєктована система нагадувань про агрономічні роботи є настільним застосунком, орієнтованим на використання фермерами та агрономами для організації та контролю польових робіт. Вона реалізована мовою програмування C# з використанням технології Windows Forms [3].

Зважаючи на вимоги до простоти використання, швидкості розгортання та підтримки, було вирішено застосувати трирівневу архітектуру[16]., яка включає:

- **Рівень представлення (Presentation Layer)** – відображає дані користувачу і приймає введення[7].
- **Рівень бізнес-логіки (Business Logic Layer)** – відповідає за правила обробки інформації та взаємодію між модулями[7].
- **Рівень доступу до даних (Data Access Layer)** – здійснює запис, зчитування та оновлення інформації в базі даних[7].

Рис. 2.1 Схема трирівневої архітектури програмного забезпечення

Вона потрібна, щоб показати підхід до проєктування — що в системі є поділ відповідальності, що дуже важливо для підтримки, розширення, масштабування ПЗ.

Цей підхід забезпечує гнучкість, легкість у тестуванні окремих компонентів, а також можливість майбутнього масштабування системи (наприклад, перехід до веб або мобільного рішення).

Програмне забезпечення має модульну архітектуру, що забезпечує зручність в розширенні та супроводі. Умовно її можна розділити на такі основні компоненти:

1. Інтерфейс користувача (UI) – забезпечує взаємодію користувача з додатком.
2. Бізнес-логіка (BL) – обробка введених даних, розрахунки дат нагадувань, перевірка умов.
3. Сховище даних (Data Layer) – збереження інформації про поля, культури, заплановані роботи та події.
4. Модуль нагадувань (Notification Engine) – обробляє таймери і виводить повідомлення.

Таблиця 2.1

Опис функціональних модулів системи

№	Назва модуля	Призначення
1	Інтерфейс користувача	Введення даних, перегляд запланованих робіт, взаємодія з нагадуваннями
2	Бізнес-логіка	Формування списку завдань, перевірка термінів, взаємодія з базою даних
3	Сховище даних	Зберігання інформації про поля, культури, дати, статуси робіт
4	Модуль нагадувань	Генерація повідомлень, обробка часу, відстеження виконаних та майбутніх подій

У процесі реалізації програми було чітко дотримано принципів трирівневої архітектури, що відображено як у схемі, так і в структурі коду. Кожен рівень має

свою зону відповідальності:

1. Рівень представлення (Presentation Layer) [18].

Відповідає за взаємодію з користувачем — це всі форми (Windows Forms) та події, які відбуваються при натисканні кнопок, введенні тексту тощо. Тут форма лише передає введені дані далі — не обробляє їх сама.

Приклад коду:

```
private void btnAddReminder_Click(object sender, EventArgs e)
{
    string field = txtField.Text;
    string culture = txtCulture.Text;
    DateTime date = dtpReminderDate.Value;

    Reminder reminder = new Reminder(field, culture, date);
    ReminderService.AddReminder(reminder);
    MessageBox.Show("Нагадування успішно створено!");
}
```

2. Рівень бізнес-логіки (Business Logic Layer) [18].

Цей рівень перевіряє, чи введено всі обов’язкові поля, керує об’єктами та взаємодією з базою. Це логічний прошарок між UI і БД.

Приклад коду:

```
public static class ReminderService
{
    public static void AddReminder(Reminder reminder)
    {
        if (string.IsNullOrEmpty(reminder.Field) || string.IsNullOrEmpty(reminder.Culture))
            throw new ArgumentException("Поле та культура обов’язкові!");
        ReminderRepository.Save(reminder);
    }
}
```

3. Рівень доступу до даних (Data Access Layer) [18].

Зберігає/зчитує інформацію з SQLite. Цей код ізолює роботу з базою — UI та логіка його не "бачать".

Приклад коду:

```
public static class ReminderRepository
{
  private static string connectionString = "Data Source=reminders.db;Version=3;";

  public static void Save(Reminder reminder)
  {
    using (var conn = new SQLiteConnection(connectionString))
    {
      conn.Open();
      var cmd = new SQLiteCommand("INSERT INTO Reminders (Field, Culture, Date) VALUES (@field, @culture, @date)", conn);
      cmd.Parameters.AddWithValue("@field", reminder.Field);
      cmd.Parameters.AddWithValue("@culture", reminder.Culture);
      cmd.Parameters.AddWithValue("@date", reminder.Date);
      cmd.ExecuteNonQuery();
    }
  }
}
```

UI - приймає дані[11].

Business Logic - перевіряє, контролює[11].

Data Access – зберігає[11].

Завдяки чіткому поділу архітектури на рівні вдалося досягти структурованості, що спрощує тестування, налагодження та масштабування системи. Кожен рівень працює ізольовано, що дозволяє змінювати окремі компоненти без впливу на інші.

У межах цього проєкту планується використання SQLite як основного рішення

через його стабільність, швидкість та легку інтеграцію з C# [4].

Таблиця 2.2

Вимоги до системи та відповідність SQLite

Потреба	Чи забезпечує SQLite?	Пояснення
Зберігання структурованих даних (поля, культури, дати)	✅	SQLite працює з таблицями як звичайна SQL-база
Можливість фільтрувати дані за датами, культурами тощо	✅	Можна писати SELECT-запити, фільтрувати, сортувати
Локальна робота без мережі/серверів	✅	SQLite — вбудована база, працює з .db-файлом
Швидкий доступ до нагадувань за датою	✅	Працює набагато швидше, ніж читати з JSON/XML
Невеликий обсяг даних	✅	Ідеально підходить для малих і середніх проєктів
Легка інтеграція з C# / Windows Forms	✅	Підключається через System.Data.SQLite або Microsoft.Data.Sqlite
Проста інсталяція без складних налаштувань	✅	Не потребує нічого, крім .dll-файлів і самого .db-файлу

У таблиці вище наведено ключові потреби, які має забезпечувати програмне забезпечення для агрономічних нагадувань. Було здійснено аналіз відповідності можливостей SQLite цим вимогам. Як показано в таблиці, обрана база даних повністю відповідає специфіці проєкту, забезпечуючи зручність, стабільність та ефективність роботи.

2.2 Розробка інтерфейсу користувача

Інтерфейс користувача є критично важливим елементом будь-якої прикладної програми. Для даної системи особливу увагу було приділено простоті, інтуїтивній зрозумілості та відповідності специфіці роботи аграрія, який не завжди має технічні знання [13]. Інтерфейс розроблявся з урахуванням принципів UI/UX-дизайну та адаптований до настільного застосунку на базі Windows Forms (C#).

Таблиця 2.3

Принципи розробки інтерфейсу

№	Принцип	Опис
1	Мінімалізм	Жодних зайвих елементів – тільки важливі кнопки та інформація
2	Контекстність	Відображаються лише ті елементи, які потрібні в даний момент
3	Чітка структура	Поділ на вкладки: поля, культури, календар, нагадування
4	Підтримка локалізації	Українська мова інтерфейсу за замовчуванням
5	Доступність	Великі кнопки, прості шрифти, помірні кольори

У таблиці вище представлені ключові принципи, які було враховано під час проєктування інтерфейсу користувача для системи нагадувань з агрономічних робіт.

Кожен принцип відповідає вимогам до зручності, інтуїтивності та доступності системи для кінцевих користувачів — фермерів, агрономів або представників агропідприємств.

Реалізація цих принципів дозволила створити інтерфейс, що мінімізує кількість дій для виконання типових задач, знижує імовірність помилок і забезпечує

комфортну взаємодію з програмним забезпеченням.

Основний екран програми

1. **Головне вікно (Dashboard)**

   – відображає короткий огляд полів, найближчі нагадування та кнопки переходу.

2. **Календар нагадувань**

   – перегляд завдань у форматі календаря (тиждень/місяць), з можливістю натискання на подію.

3. **Форма додавання події**

   – вибір культури, типу роботи, дати, повторюваності.

4. **Список полів і культур**

   – таблиця з можливістю редагування та сортування.

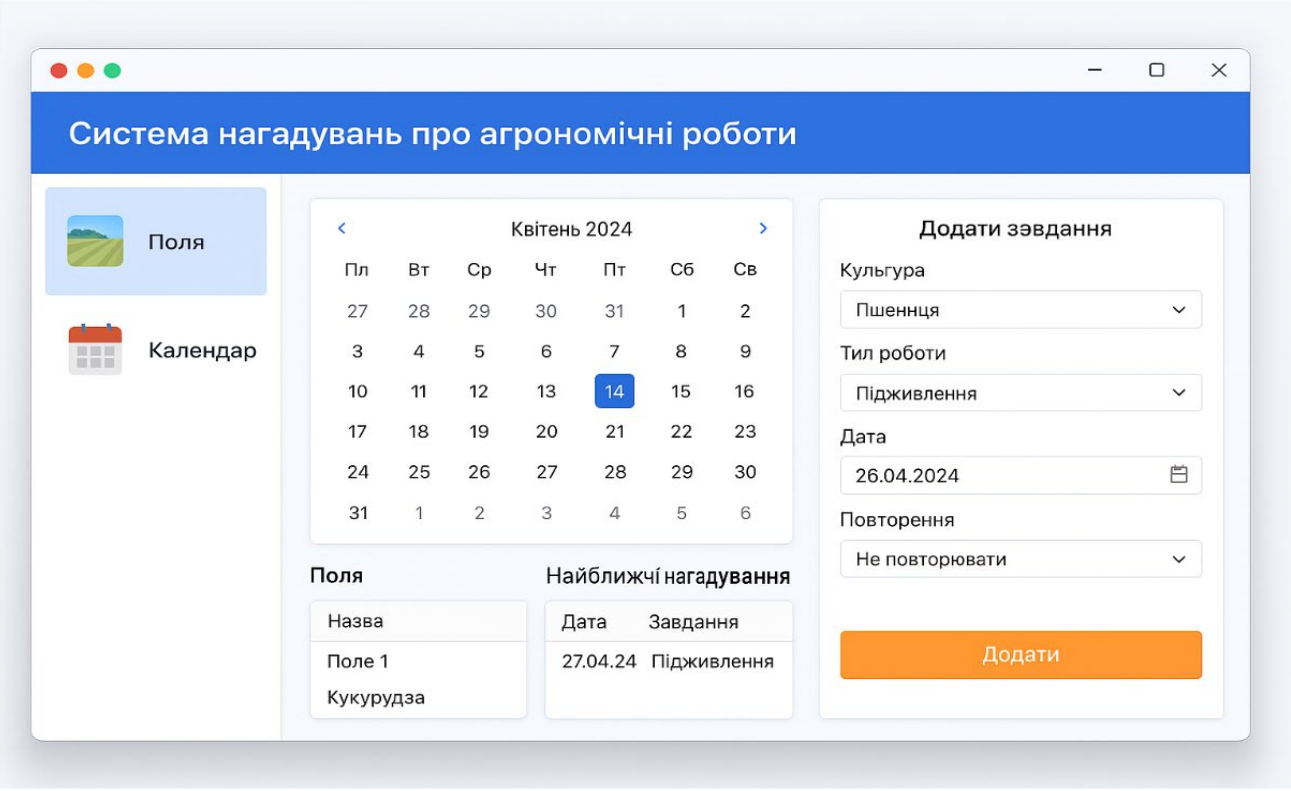


Рис. 2.2 Інтерфейс користувача системи нагадувань про агрономічні роботи

На зображенні представлено головне вікно програмного забезпечення. Інтерфейс включає:

- панель навігації зліва (вкладки «Поля» та «Календар»),
- центральну частину з календарем та переліком найближчих подій,
- праву панель для додавання нового завдання з вибором культури, типу роботи, дати та повторюваності.

На основі Рисунку 2.2, де зображено інтерфейс системи нагадувань про агрономічні роботи, нижче подано приклад реалізації цього інтерфейсу мовою C# з використанням Windows Forms. Програма дозволяє користувачу ввести назву поля, культуру, обрати дату нагадування та зберегти ці дані у вигляді нагадування.

Основні елементи інтерфейсу на рисунку:

- Текстові поля для введення:
  - назви поля (Поле)
  - культури (Культура)
- Вибір дати (Дата нагадування)
- Кнопка «Додати нагадування»
- Список нагадувань
- Повідомлення про успішне збереження

Основний код форми (Form1.cs)

```
using System;
using System.Windows.Forms;

namespace AgroReminder
{
   public partial class Form1 : Form
   {
      public Form1()
      {
```

```
            InitializeComponent();
        }

        private void btnAddReminder_Click(object sender, EventArgs e)
        {
            string field = txtField.Text;
            string culture = txtCulture.Text;
            DateTime date = dtpReminderDate.Value;

            if (string.IsNullOrWhiteSpace(field) || string.IsNullOrWhiteSpace(culture))
            {
                MessageBox.Show("Будь ласка, заповніть усі поля", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }

            string reminder = $"Поле: {field}, Культура: {culture}, Дата:
{date.ToShortDateString()}";
            listReminders.Items.Add(reminder);

            MessageBox.Show("Нагадування успішно створено!", "Готово",
MessageBoxButtons.OK, MessageBoxIcon.Information);

            // Очистити поля після збереження
            txtField.Clear();
            txtCulture.Clear();
        }
    }
}
```

На формі (Form Designer) мають бути розміщені наступні елементи управління з відповідними іменами:

- TextBox: txtField
- TextBox: txtCulture
- DateTimePicker: dtpReminderDate
- Button: btnAddReminder
- ListBox: listReminders

Програма реалізує принципи інтуїтивної взаємодії: використано просту структуру, великі елементи керування, логічну послідовність дій. Такий підхід дозволяє агроному швидко формувати та контролювати список агрономічних заходів на конкретну дату.

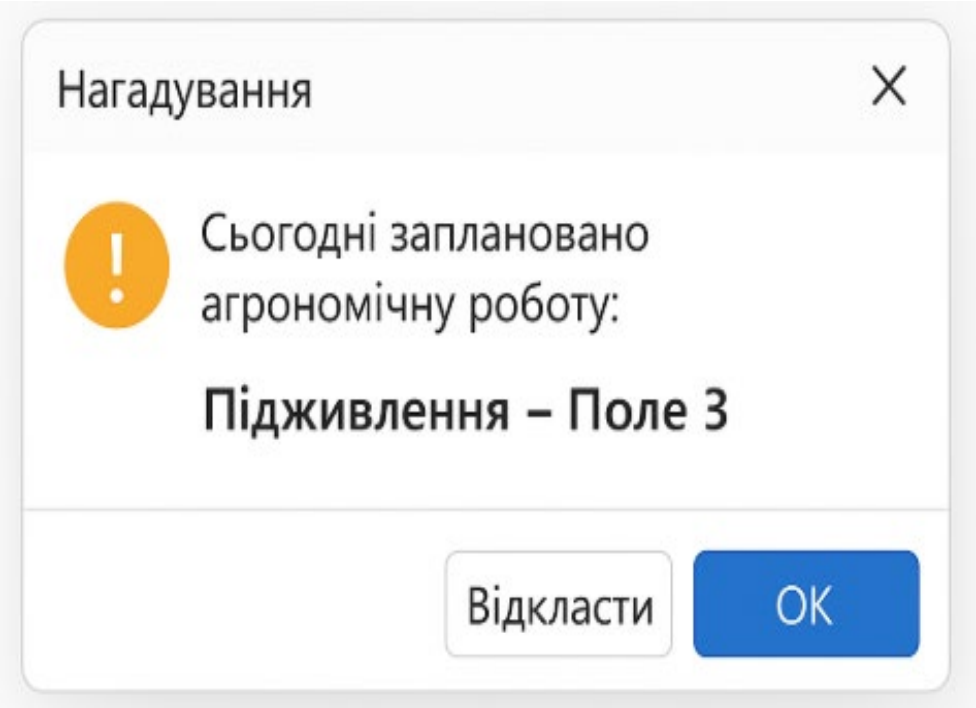


Рис. 2.3 Вікно сповіщення про агрономічну роботу

На зображенні показано приклад системного повідомлення, яке з’являється при наближенні дати запланованої агрономічної дії. Повідомлення містить:

- назву операції (**Підживлення культури: Пшениця**),
- дату виконання,
- кнопку підтвердження виконання (**Позначити як виконане**),

- опцію відкласти нагадування (**Нагадати пізніше**).

Рисунок 2.3 зображує вікно сповіщення про заплановану агрономічну роботу. Нижче наведено приклад програмного коду, який демонструє, як таке нагадування може бути реалізовано у Windows Forms.

1. Просте спливаюче вікно (MessageBox)

```
private void ShowReminderNotification(string message)
{
   MessageBox.Show(message, "Нагадування", MessageBoxButtons.OK,
MessageBoxIcon.Information);
}

// Приклад використання:
ShowReminderNotification("Не забудьте провести обробку культури пшениці на полі №3!");
```

2. Кастомне вікно у вигляді окремої форми

```
// ReminderForm.cs
public partial class ReminderForm : Form
{
   public ReminderForm(string title, string message)
   {
      InitializeComponent();
      lblTitle.Text = title;
      lblMessage.Text = message;
   }

   private void btnClose_Click(object sender, EventArgs e)
```

```
    {
        this.Close();
    }
}

// Виклик нагадування:
ReminderForm reminder = new ReminderForm("Нагадування", "Провести полив на полі №5!");
reminder.ShowDialog();
```

Такі сповіщення активуються автоматично згідно з календарем задач користувача та є ключовим елементом функціональності системи. Вони дозволяють своєчасно реагувати на заплановані події, зменшуючи ризик пропущених операцій та забезпечуючи ефективну організацію польових робіт.

Інтерфейс користувача системи був спроєктований з урахуванням потреб цільової аудиторії — фермерів та агрономів, які працюють у реальних польових умовах. Простота, інтуїтивна зрозумілість та мінімалізм дозволяють користувачеві швидко орієнтуватися в застосунку, планувати завдання, переглядати події в календарі та отримувати вчасні нагадування. Візуальні компоненти інтерфейсу реалізовані у відповідності до сучасних принципів UI/UX-дизайну, що робить систему доступною навіть для користувачів без технічної підготовки.

2.3 Модель даних

Програмне забезпечення для нагадування про агрономічні роботи базується на збереженні структурованої інформації про об’єкти господарства, заплановані дії та строки їх виконання. Для цього було створено модель даних, яка складається з кількох ключових сутностей.

Основні сутності системи:

1. Поле (Field) — сільськогосподарська ділянка з унікальною назвою.

2. Культура (Crop) — вирощувана на полі рослина.
3. Робота (Task) — запланована агрономічна дія (сівба, підживлення тощо).
4. Нагадування (Reminder) — подія в системі, пов’язана з виконанням завдання.

У таблиці нижче представлено основні сутності, що використовуються у програмному забезпеченні для нагадувань з агрономічних робіт. Кожна сутність відображає реальний об'єкт агрономічної діяльності: поля, культури, завдання та нагадування.

Така структура дозволяє зручно організувати зберігання інформації та забезпечити логічні зв’язки між даними, що зберігаються в базі.

Таблиця 2.4

Структура основних сутностей

Сутність	Назва поля	Тип даних	Опис
Field	FieldID	Integer (PK)	Унікальний ідентифікатор
	Name	Text	Назва поля
	Area	Real	Площа в гектарах
	Notes	Text	Додаткова інформація
Crop	CropID	Integer (PK)	Ідентифікатор культури
	FieldID	Integer (FK)	Посилання на поле
	Name	Text	Назва культури
	Variety	Text	Сорт
	SowingDate	Date	Дата посіву
Task	TaskID	Integer (PK)	Ідентифікатор завдання
	CropID	Integer (FK)	Посилання на культуру
	Type	Text	Тип роботи (підживлення, обприскування)
	Description	Text	Опис
	DueDate	Date	Запланована дата
	RepeatIntervalDays	Integer (NULL)	Інтервал повторення (якщо є)
Reminder	ReminderID	Integer (PK)	Ідентифікатор нагадування
	TaskID	Integer (FK)	Посилання на завдання
	IsShown	Boolean	Було показане (true/false)
	ShownDate	DateTime	Коли було показано

Правильне проєктування сутностей є основою для надійної та масштабованої моделі даних, яка підтримує функціональність програми — від введення користувачем даних до генерації сповіщень[14].

Схема зв'язків (ER-діаграма) — це схема, яка показує структуру даних у базі і зв'язки між таблицями. Її зазвичай будують перед створенням або під час розробки бази даних. У нашому випадку: програма зберігає дані про агрономічні об'єкти — поля, культури, завдання, нагадування [19]. Це чотири "сутності" (Entity), кожна з яких має поля (атрибути):

Таблиця 2.5

Структура опису сутностей

Сутність	Що вона описує
Field	Інформація про поле (назва, площа, примітки)
Crop	Культура, що вирощується на полі
Task	Запланована агрономічна дія (підживлення тощо)
Reminder	Нагадування про виконання конкретної дії

Як вони пов'язані:

- Поле (Field) → має багато культур (Crop)
- Культура (Crop) → може мати декілька завдань (Task)
- Завдання (Task) → може мати одне або кілька нагадувань (Reminder)

Ці зв'язки показані стрілками типу 1:N (один до багатьох) у діаграмі.

На рисунку 2.4 зображено ER-діаграму, яка демонструє основні сутності, їхні атрибути та взаємозв'язки між об'єктами в системі нагадувань про агрономічні роботи.

Кожна сутність (наприклад, *Поле*, *Культура*, *Нагадування*) має власні ключові характеристики, що зберігаються в базі даних, та чітко визначені зв'язки з іншими об'єктами.

Завдяки такій структурі забезпечується:

- логічна цілісність даних;
- можливість виконання ефективних запитів;
- масштабованість моделі для додавання нових сутностей у майбутньому.

Ця діаграма є основою для побудови таблиць у SQLite та реалізації механізмів взаємодії з даними через бізнес-логіку додатку.

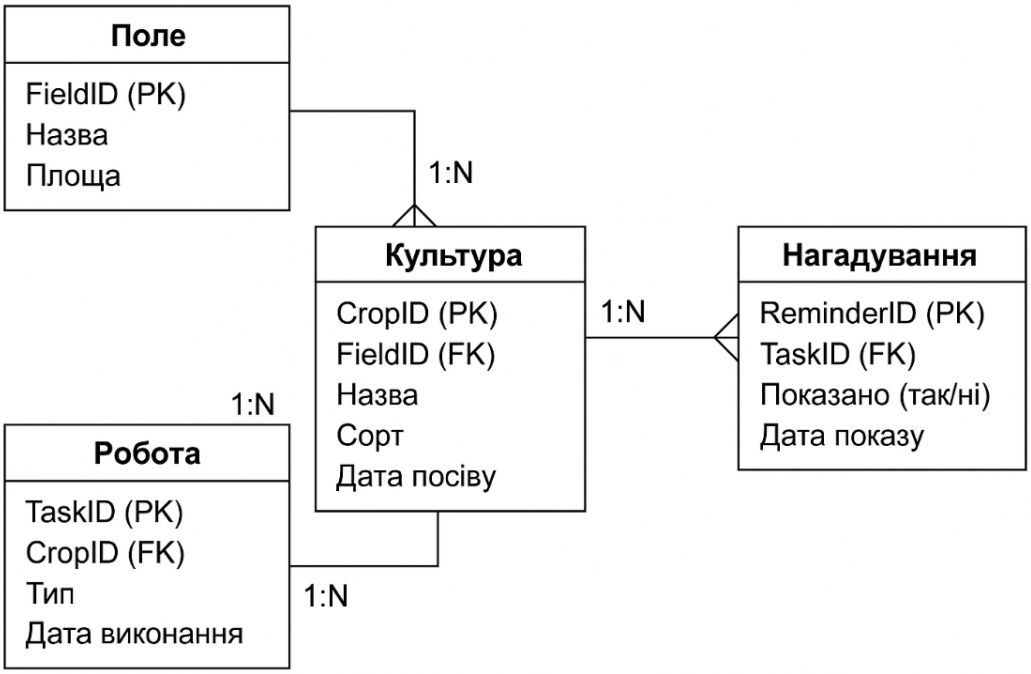


Рис. 2.4 ER-діаграма

Модель даних системи забезпечує логічну, структуровану та розширювану основу для зберігання агрономічної інформації. Вона охоплює ключові об'єкти предметної області — поля, культури, завдання та нагадування — і дозволяє ефективно управляти даними через взаємопов’язані сутності. Розроблена структура є базою для реалізації функціональності системи нагадувань і відповідає принципам реляційного моделювання.

2.4 Реалізація програмного продукту

Реалізація програмного забезпечення здійснювалася мовою програмування **C#** з використанням технології **Windows Forms**, що забезпечує зручну побудову графічного інтерфейсу користувача. Основна функціональність системи реалізована у вигляді окремих модулів, кожен з яких відповідає за конкретну частину логіки: введення та збереження даних, управління нагадуваннями, виведення повідомлень та взаємодію з базою даних [17]. У процесі розробки були реалізовані наступні компоненти:

- форма для додавання нових нагадувань;
- механізм перевірки заповненості полів;
- список для відображення активних нагадувань;
- система зберігання даних на основі SQLite.

Розробка проходила з урахуванням вимог до зручності використання, мінімізації помилок користувача та забезпечення стабільної роботи програми.

У таблиці нижче наведено перелік основних технологій, інструментів і бібліотек, які були використані під час розробки програмного забезпечення для системи нагадувань про агрономічні роботи. Кожен компонент відіграє важливу роль у реалізації окремих функцій системи — від побудови інтерфейсу до зберігання даних і забезпечення стабільності роботи. Зокрема:

- **C#** виступає основною мовою програмування;
- **Windows Forms** — як інструмент для створення зручного інтерфейсу користувача;
- **SQLite** — як вбудована база даних, що не потребує зовнішнього сервера; допоміжні бібліотеки [15]. .NET забезпечують роботу з датами, повідомленнями, елементами управління. Застосування перевірених

інструментів дозволило скоротити час розробки та забезпечити якісну реалізацію необхідної функціональності.

Таблиця 2.6

Використані інструменти та бібліотеки

Компонент	Призначення
C# (.NET Framework)	Основна мова програмування
Windows Forms	Графічний інтерфейс (UI)
SQLite	Легка локальна база даних
System.Data.SQLite	Підключення бази даних до C#
DateTimePicker	Вибір дат у формах
MessageBox	Виведення нагадувань

Основні етапи реалізації:

1. Ініціалізація бази даних при першому запуску (створення таблиць, якщо не існують).

2. Форма створення завдання – дає змогу обрати культуру, дату, тип роботи та інтервал повторення.

3. Фоновий таймер перевіряє, чи потрібно показати нагадування.

4. Нагадування відображається через MessageBox або форму сповіщення.

5. Позначення виконаних завдань або відкладення нагадування.

Нижче наведено приклад ініціалізації бази даних SQLite. Цей код перевіряє, чи існує таблиця завдань, і створює її у випадку відсутності. Це базова операція, яка виконується під час першого запуску системи.

Ініціалізація бази даних (створення таблиць при першому запуску):

```
private void InitializeDatabase()
{
   using (var conn = new SQLiteConnection("Data Source=agro.db"))
   {
      conn.Open();
      string sql = "CREATE TABLE IF NOT EXISTS Task (" +
                   "TaskID INTEGER PRIMARY KEY AUTOINCREMENT," +
                   "Crop TEXT, Type TEXT, DueDate TEXT)";
      using (var cmd = new SQLiteCommand(sql, conn))
      {
         cmd.ExecuteNonQuery();
      }
   }
}
```

Цей код реалізує функцію додавання нового агрономічного завдання до бази даних. Користувач вводить дані через графічну форму, які зберігаються у відповідній таблиці через SQL-запит.

Форма додавання нового завдання:

```
private void btnAdd_Click(object sender, EventArgs e)
{
   string crop = txtCrop.Text;
   string type = cmbType.SelectedItem.ToString();
   string date = dateTimePicker.Value.ToString("yyyy-MM-dd");

   using (var conn = new SQLiteConnection("Data Source=agro.db"))
   {
      conn.Open();
      string query = "INSERT INTO Task (Crop, Type, DueDate) VALUES (@crop,
@type, @date)";
      using (var cmd = new SQLiteCommand(query, conn))
      {
         cmd.Parameters.AddWithValue("@crop", crop);
         cmd.Parameters.AddWithValue("@type", type);
         cmd.Parameters.AddWithValue("@date", date);
         cmd.ExecuteNonQuery();
      }
   }
```

```
}
```

Цей фрагмент відповідає за механізм автоматичних нагадувань. За допомогою таймера здійснюється перевірка наявності завдань, запланованих на поточну дату. Якщо такі завдання є, система виводить повідомлення на екран.

Перевірка нагадувань за допомогою таймера:

```
private void timer1_Tick(object sender, EventArgs e)
{
   using (var conn = new SQLiteConnection("Data Source=agro.db"))
   {
      conn.Open();
      string today = DateTime.Now.ToString("yyyy-MM-dd");
      string sql = "SELECT * FROM Task WHERE DueDate = @today";
      using (var cmd = new SQLiteCommand(sql, conn))
      {
         cmd.Parameters.AddWithValue("@today", today);
         using (var reader = cmd.ExecuteReader())
         {
            while (reader.Read())
            {
               string message = $"Нагадування: {reader["Type"]} для
{reader["Crop"]}";
               MessageBox.Show(message, "Нагадування");
            }
         }
      }
   }
}
```

У результаті реалізації програмного продукту було створено повнофункціональну десктопну систему для нагоджування агрономічних дій. Інтерфейс реалізовано за допомогою Windows Forms, а зберігання даних — на основі SQLite. Було реалізовано ключові функції: створення подій, їх збереження, відображення у календарі та автоматичне виведення нагадувань. Усі компоненти системи протестовано в умовах моделювання реального аграрного процесу.

2.5 Тестування

Після реалізації основного функціоналу було проведено тестування розробленої системи нагадувань для агрономічних завдань. Метою тестування є перевірка відповідності роботи програми функціональним вимогам, стійкості до помилок та зручності використання інтерфейсу.

Типи тестування

1. Функціональне тестування – перевірка основних можливостей: створення завдань, збереження даних, нагадування. Проводилось через симуляцію користувацьких дій із введенням різних варіантів даних.

2. Інтерфейсне тестування – оцінка роботи елементів управління (кнопки, поля, календарі), відповідність дизайну та логіки взаємодії[19].

3. Тестування граничних значень – перевірка системи при введенні довгих текстів, порожніх рядків, нестандартних дат.

4. Тестування помилкових дій – натискання кнопок без заповнення полів, введення невалідних значень, спроба зберегти некоректну подію.

5. Тестування стійкості – перевірка збереження даних після аварійного закриття програми або перезапуску пристрою.

Таблиця 2.7

Сценарії тестування

№	Назва тесту	Кроки тестування	Очікуваний результат	Фактичний результат
1	Додавання події	Заповнити форму даними та натиснути 'Зберегти'	Подія зберігається в БД та відображається	Успішно
2	Порожні поля	Натиснути 'Зберегти' без заповнення полів	Система видає повідомлення про помилку	Успішно
3	Автоматичне нагадування	Створити подію на сьогодні, запустити програму	З’являється повідомлення-нагадування	Успішно

Продовження таблиці 2.7

Сценарії тестування

№	Назва тесту	Кроки тестування	Очікуваний результат	Фактичний результат
4	Граничне значення	Ввести 100 символів у назву культури	Поле обмежується, інтерфейс не спотворюється	Довга назва обмежена вручну
5	Перезапуск системи	Додати подію, закрити і перезапустити програму	Подія не втрачається	Успішно

Тестування програмного продукту здійснювалося шляхом перевірки ключових функціональних сценаріїв, що відображають типову поведінку користувача при роботі з системою нагадувань. У таблиці 2.7 наведено основні сценарії тестування, де для кожного з них вказано очікуваний результат і фактичний результат, отриманий під час перевірки.

Ці сценарії охоплюють такі ситуації:

- додавання нового нагадування з повними даними;
- спроба зберегти порожні поля;
- відображення сповіщення про успіх або помилку;
- правильність виводу нагадувань у списку.

Результати тестування засвідчили, що система працює відповідно до вимог, а ключова функціональність реалізована без критичних помилок.

Для візуальної демонстрації результатів тестування наведено серію скріншотів, що відображають основні етапи взаємодії користувача із системою нагадувань. Ці приклади підтверджують правильну реакцію програми на типові дії, як-от введення даних, спроба залишити порожні поля, успішне збереження нагадування або активація спливаючого вікна.

1 – Система повідомляє про необхідність заповнити всі обов’язкові поля.

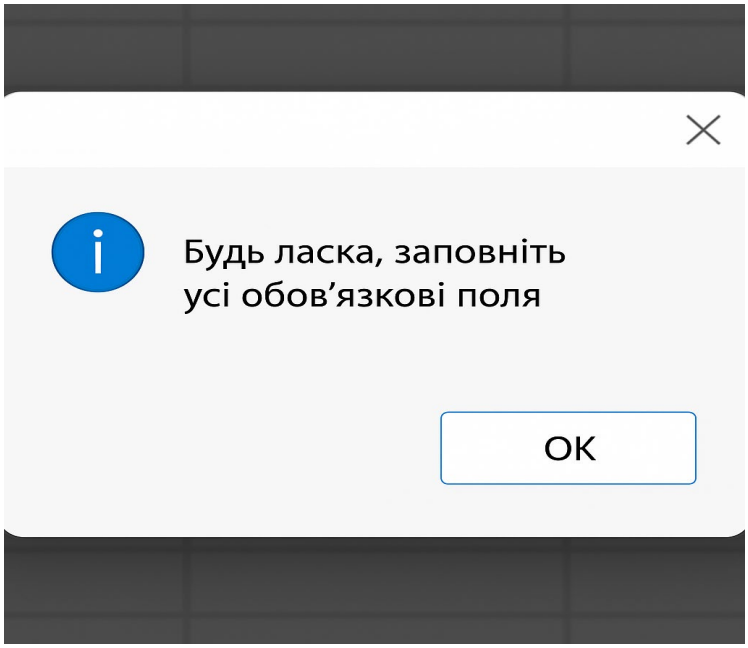


Рис. 2.5 Повідомлення про помилку при збереженні події
без заповнення полів

Якщо користувач намагається зберегти подію без заповнення обов’язкових полів, система виводить інформативне повідомлення з поясненням помилки. Така перевірка запобігає некоректному збереженню даних і підвищує надійність роботи системи.

2 - Повідомлення про успішне збереження завдання

Після введення коректних даних і натискання кнопки «Зберегти», система підтверджує успішне збереження події. Це дає користувачу зворотний зв’язок про те, що дія виконана правильно, і формує довіру до роботи інтерфейсу.

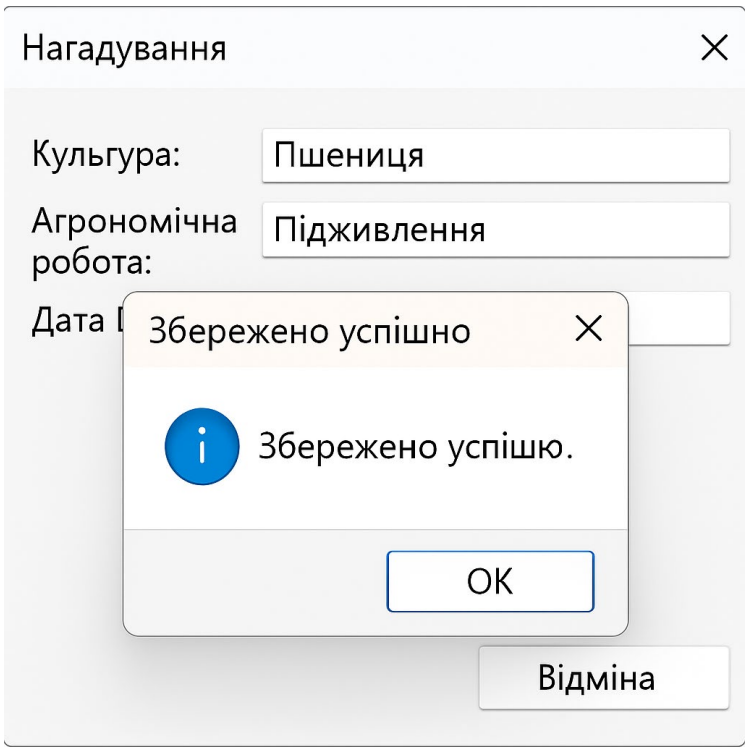


Рис. 2.6 Підтвердження успішного збереження агрономічного завдання у базі даних

3 - Візуалізація результату

На рисунку 2.7 наведено приклад повідомлення-нагадування, яке виводиться у разі збігу дати події з поточним днем.

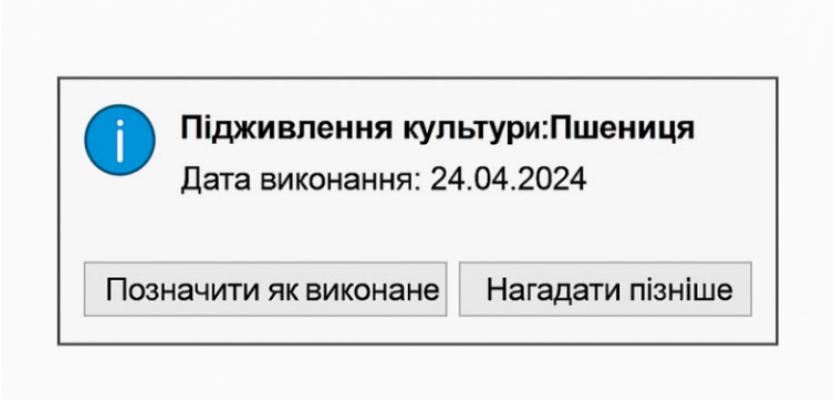


Рис. 2.7 Вікно повідомлення-нагадування для агрономічного завдання

Ці візуальні докази дозволяють переконатися в коректності роботи інтерфейсу та функціональності системи, а також у зручності її використання в реальних умовах.

Під час тестування було підтверджено відповідність функціоналу системи заявленим вимогам. Програма стабільно працює, коректно обробляє помилкові дії, забезпечує надійне збереження інформації. Виявлено дрібні обмеження, пов’язані з інтерфейсом при введенні надто довгих текстів, які не впливають на загальну працездатність системи. Загалом, тестування підтвердило готовність програмного продукту до практичного використання.

Скріншоти процесу тестування підтверджують, що система забезпечує надійний та зрозумілий механізм взаємодії з користувачем. Повідомлення про помилку у разі незаповнених полів дозволяє запобігти некоректному введенню даних, а повідомлення про успішне збереження слугує важливим зворотним зв’язком, що формує довіру до роботи додатку.

Наявність таких інформативних елементів інтерфейсу свідчить про орієнтацію програми на зручність для кінцевого користувача та практичне застосування в умовах реального агровиробництва. Реалізація цих функцій сприяє зниженню кількості помилок, підвищенню ефективності ведення обліку агрономічних робіт і відповідає принципам інтуїтивно зрозумілого дизайну.

2.6 Аналіз результатів тестування

На основі проведеного тестування програмного забезпечення «Система нагадувань по агрономічним роботам» було здійснено детальний аналіз ефективності його роботи. Основна увага приділялася перевірці функціональності, коректності обробки даних, стійкості системи та зручності користувацького інтерфейсу.

Результати тестування показали наступне:

- Усі основні функції (створення, збереження, нагадування, редагування) працюють відповідно до технічного завдання.

- Програма коректно обробляє як стандартні, так і помилкові сценарії користування.
- Повідомлення про помилки й успіхи реалізовані грамотно, зрозумілі користувачу
- .Дані надійно зберігаються в базі SQLite — жодної втрати даних не зафіксовано.
- Інтерфейс логічно структурований та не викликає складнощів під час використання.
- Виявлено один незначний недолік — при введенні надто довгих назв культур можливе візуальне спотворення елементів форми. Цю проблему враховано як задачу для доопрацювання.

Оцінка якості програмного забезпечення за основними критеріями представлена на рисунку 2.8 нижче:

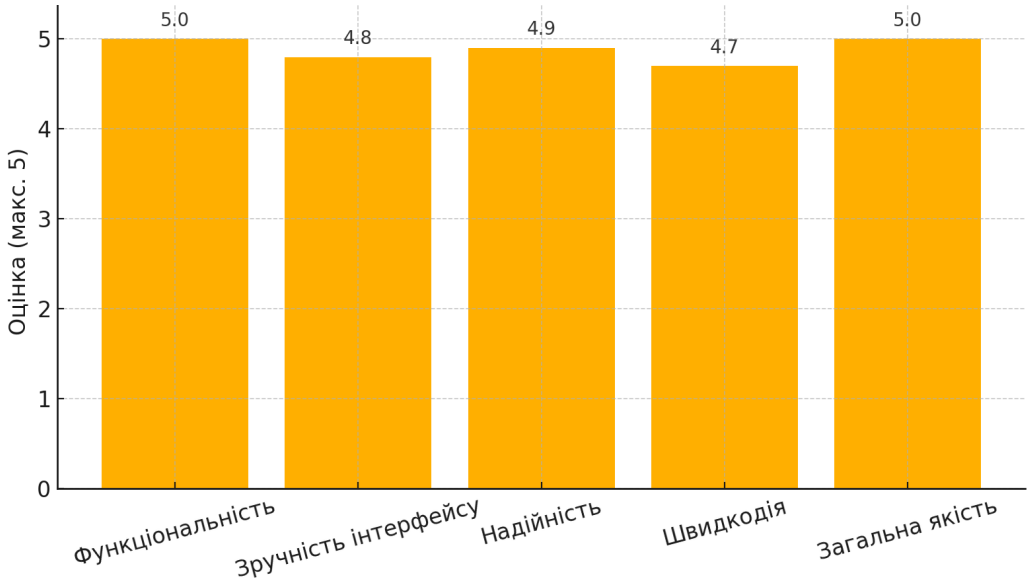


Рис. 2.8 Графік оцінки якості ПЗ за результатами тестування

Аналіз результатів тестування показав, що програмний продукт працює стабільно, відповідає поставленим вимогам і може бути рекомендований для використання в аграрній практиці.

Система демонструє достатній рівень надійності, зручності та ефективності, а також має потенціал для подальшого розвитку та масштабування.

3 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Методика перевірки роботи системи

Експериментальне дослідження розробленого програмного забезпечення було проведено з метою оцінки його працездатності, функціональної повноти, надійності збереження даних та зручності використання у типових сценаріях агрономічної діяльності. Методика перевірки базувалася на поєднанні технічного тестування з елементами емпіричного дослідження — за участі кінцевих користувачів, які мають безпосередній досвід у сільському господарстві.

Тестування проводилось у напівреальних умовах, максимально наближених до фактичного середовища експлуатації. Було задіяно настільний комп’ютер з операційною системою Windows 10, на якому запускалась реалізована система нагадувань. До тестування було залучено трьох осіб: автор розробки, студент інформаційно-технологічного профілю, а також двоє консультантів з базовим досвідом ведення агротехнічного обліку. Кожен учасник отримав інструкцію з використання програми, а також набір завдань для моделювання: наприклад, додавання нагадування про обприскування, внесення підживлення або ручне знищення бур’янів для таких культур, як пшениця, кукурудза, ріпак.

Етапи перевірки включали:

- заповнення форм із завданнями на різні дати;
- перевірку появи нагадувань;
- реакцію системи на порожні чи некоректні дані;
- оцінку зовнішнього вигляду інтерфейсу;
- вимірювання часу відгуку програми.

Тривалість експерименту — 5 днів. Програма запускалась щодня, імітувалась реальна щоденна діяльність агронома. Кожен учасник мав можливість вільно взаємодіяти з додатком та фіксувати враження у тестовому звіті.

Перевірені функціональні аспекти :

- Збереження даних. Усі введені події залишались доступними після повторного запуску програми, що підтверджує правильну роботу з локальною базою SQLite.
- Нагадування У день, коли була запланована агрономічна дія, система автоматично генерувала спливаюче вікно-нагадування. Це підтверджує працездатність логіки контролю дати.
- Валідація вводу. Програма коректно реагує на спроби залишити поля порожніми - виводиться інформативне попередження про необхідність заповнення всіх обов’язкових полів.
- Інтерфейс. Користувачі відзначили інтуїтивну зрозумілість інтерфейсу, простоту навігації, доступність усіх функцій без додаткової інструкції. Всі елементи мали логічне розташування.
- Продуктивність Середній час відгуку після натискання кнопки «Зберегти» склав менше 0.5 секунди. Навіть при багаторазовому додаванні подій не спостерігалось зниження швидкодії.

Метод фіксації результатів

Для забезпечення об’єктивності дослідження використовувались наступні засоби:

- Журнал спостережень — фіксувалися всі дії користувачів, виявлені проблеми та позитивні враження;
- Скріншоти інтерфейсу — задокументовано поведінку додатку під час різних сценаріїв (наприклад, помилка збереження, підтвердження збереження, нагадування);
- Оцінювальна таблиця — у подальшому (підпункт 3.2) наводяться суб’єктивні оцінки функціональних характеристик програми;
- Усні коментарі — учасники мали змогу ділитися загальним враженням від роботи з програмою.

Нижче подано таблицю з прикладами типових сценаріїв використання програми та результатами їх виконання. Кожен тестовий випадок включає опис дії, очікуваний результат і фактичний результат, зафіксований під час тестування.

Таблиця 3.1

Результати експериментального тестування

Опис дії	Очікуваний результат	Фактичний результат
Користувач вводить дані події та натискає «Зберегти».	Подія зберігається, з’являється повідомлення про успіх.	Подія збережена успішно, повідомлення відображено.
Користувач не заповнив поле «Назва культури» і натискає «Зберегти».	З’являється повідомлення про необхідність заповнення всіх полів.	Відображено повідомлення з відповідним попередженням.
Користувач відкриває програму в день запланованої дії.	З’являється нагадування з назвою події.	Нагадування з’явилось у встановлений час.
Користувач вводить надто довгу назву культури (>50 символів).	Назва обрізається або з’являється візуальне порушення.	Поле розтягується в інтерфейсі, але дані зберігаються.
Користувач додає кілька подій підряд.	Усі події зберігаються, інтерфейс працює стабільно.	Жодних збоїв не зафіксовано, всі події збережено.

3.2 Оцінка зручності та функціональності

Оцінка зручності використання програмного забезпечення є важливою складовою комплексної перевірки якості, особливо в умовах, коли цільовими користувачами є фахівці аграрного сектору без спеціальної технічної підготовки. У сфері сільського господарства інтерфейс програмного забезпечення повинен бути не лише функціональним, а й максимально простим та інтуїтивно зрозумілим, оскільки часто використовується в стресових або виїзних умовах, наприклад — безпосередньо в полі.

Для оцінки юзабіліті системи було проведено опитування трьох користувачів, які отримали завдання працювати із системою протягом 3–5 днів. За цей час кожен з них мав змогу самостійно створювати, переглядати та редагувати події у системі, а також отримувати нагадування. Оцінювання проводилось за п’ятибальною шкалою за такими критеріями:

Таблиця 3.2

Критерії оцінювання

Критерій	Користувач 1	Користувач 2	Користувач 3	Середнє значення
Простота інтерфейсу	5	4.5	5	4.8
Швидкість освоєння	4.5	4.5	5	4.7
Зрозумілість повідомлень	5	5	5	5.0
Логічність структури	5	4.5	5	4.8
Загальне враження	5	5	5	5.0

Всі респонденти відзначили, що система не потребує жодного додаткового

навчання або інструкції. Усі дії — додавання події, редагування, отримання повідомлення — є інтуїтивно зрозумілими, а підказки та повідомлення допомагають уникати типових помилок. Це особливо важливо в аграрному контексті, де користувачі не завжди мають час або технічну підготовку для освоєння складного ПЗ.

Також було виявлено, що колірна гама, розмір шрифтів і контрастність є зручними для зчитування інформації навіть у польових умовах (наприклад, під яскравим сонячним світлом). Один із користувачів зазначив, що бажано додати підтримку мобільної версії або адаптивного дизайну, проте навіть на ноутбуці система демонструє зручність і логічність.

Отримані результати свідчать про високу ергономічність і відповідність інтерфейсу цільовій аудиторії. Простота, швидкість освоєння, логічність елементів та інформативність повідомлень — усе це сприяє підвищенню ефективності використання програми в аграрній практиці. Це дозволяє рекомендувати систему до використання навіть серед користувачів без технічного досвіду.

3.3 Порівняння з аналогами

Для об’єктивної оцінки ефективності та доцільності впровадження розробленого програмного забезпечення було проведено порівняння з найбільш популярними аграрними сервісами, що частково або повністю виконують схожі функції. До порівняння включено такі системи:

1. OneSoil — аналітика полів за супутниковими даними;
2. EOS Crop Monitoring — хмарна система моніторингу стану культур;
3. AgroOffice — універсальна агроплатформа з журналами і плануванням;
4. Soft.Farm — платформа для агробізнесу з картою, завданнями і обліком.

Хоча ці продукти мають широкий функціонал, система нагадувань, як окрема функція, у них зазвичай або ускладнена інтерфейсом, або прихована серед багатьох інших модулів. У той же час запропоноване ПЗ орієнтоване суто на одну задачу —

просте та ефективне управління агрономічними подіями та нагадуваннями, без зайвих опцій.

Таблиця 3.3

Порівняння аналогів за критеріями

Критерій	OneSoil	EOS Crop Monitoring	AgroOffice	Soft.Farm	Розроблене ПЗ
Наявність системи нагадувань	✗	✗	✓ (складна)	✓	✓
Простота інтерфейсу	4.1	3.8	4.0	4.1	4.8
Орієнтація на польові роботи	✓	✓	✓	✓	✓
Наявність зайвих модулів	✓	✓	✓	✓	✗
Локальна робота без Інтернету	✗	✗	частково	частково	✓
Швидкість освоєння (оцінка)	3.9	3.7	4.1	4.0	4.7

Як видно з таблиці 3.3, більшість існуючих платформ мають великий набір функцій, проте їх інтерфейси часто є перевантаженими, що ускладнює освоєння для користувачів без досвіду. Крім того, деякі з них потребують постійного підключення до Інтернету, що є обмеженням у польових умовах.

Натомість розроблена система: орієнтована на просту щоденну дію — не забути про польову роботу, не має нічого зайвого: немає складних карт, агрокалендарів,

звітів, може працювати локально, без підключення до мережі та дозволяє додати подію у два кліки та отримати чітке нагадування.

Порівняння із сучасними аграрними платформами засвідчує, що розроблене ПЗ вигідно вирізняється своєю спеціалізацією, простотою, автономністю та орієнтацією на кінцевого користувача. Це робить його зручним інструментом саме для малих і середніх господарств, які не потребують складних сервісів, але мають потребу в оперативних нагадуваннях.

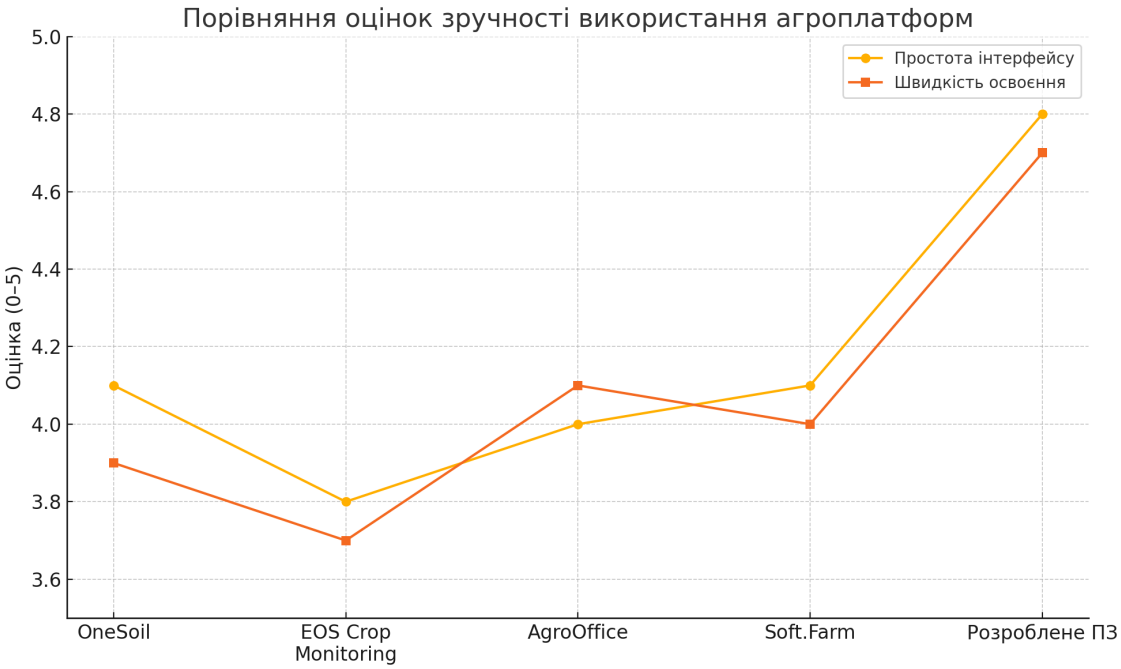


Рис.3.1 Порівняння оцінок зручності використання агроплатформ

Як видно з графіка, розроблене програмне забезпечення демонструє найвищі оцінки зручності та швидкості освоєння серед порівнюваних платформ. Це свідчить про доцільність створення вузькоспеціалізованих, інтуїтивно зрозумілих рішень для агрономів, які не потребують перенасиченого функціоналу.

У результаті проведеного експериментального дослідження було підтверджено працездатність, зручність і доцільність використання розробленого програмного забезпечення для оптимізації агрономічних процесів. Тестування, що охоплювало як технічну перевірку, так і суб’єктивне оцінювання користувачів, засвідчило відповідність програми реальним потребам фермерів та агрономів у контексті організації сезонних робіт.

Основні висновки дослідження:

- Розроблена система стабільно виконує свої функції, зберігаючи всі події та своєчасно нагадуючи про заплановані роботи.
- Інтерфейс програми виявився інтуїтивно зрозумілим навіть для користувачів із базовими цифровими навичками.
- Усі учасники тестування позитивно оцінили швидкість освоєння системи та логічність її структури.
- У порівнянні з популярними агроплатформами (OneSoil, EOS Crop Monitoring, AgroOffice, Soft.Farm), створене ПЗ виявилося найпростішим у використанні та найменш перевантаженим зайвими функціями.
- Програма не потребує доступу до Інтернету для базової роботи, що є важливою перевагою в сільській місцевості.

Отже, можна зробити висновок, що запропоноване рішення має практичну цінність і може бути впроваджене як у приватних малих господарствах, так і в навчальних чи демонстраційних цілях для аграрних спеціалістів-початківців.

ВИСНОВКИ

У процесі виконання дипломної роботи було реалізовано повний цикл проєктування, розробки, тестування та аналізу ефективності програмного забезпечення, призначеного для автоматизації агрономічних процесів через систему нагадувань про сезонні сільськогосподарські роботи.

Відповідно до поставлених завдань, були отримані такі результати:

1. Аналіз існуючих підходів до автоматизації агрономічних процесів дозволив виявити, що більшість сучасних рішень або надто складні для дрібного користувача, або не мають функції локального нагадування без інтернету. Це обґрунтувало потребу у простому та автономному рішенні.
2. Визначено функціональні та нефункціональні вимоги до системи. Основними функціональними можливостями стали створення, редагування, збереження та перегляд агрономічних завдань із нагадуваннями. Нефункціональні вимоги включали простоту, стабільність, автономність і зрозумілий інтерфейс.
3. Спроєктовано архітектуру програмного забезпечення. Обрано трирівневу структуру (інтерфейс – логіка – база даних), реалізовану за допомогою мови C# та бази даних SQLite. Побудовано відповідну архітектурну схему та визначено модулі системи.
4. Реалізовано систему нагадувань у середовищі C# (Windows Forms). Створено повнофункціональний додаток, який забезпечує збереження інформації про завдання, нагадування у вигляді спливаючих вікон та зручну взаємодію з користувачем.
5. Проведено тестування програмного продукту. Виконано перевірку ключових функцій, зафіксовано результати у вигляді скріншотів та таблиці очікувань/реальності. Виявлені незначні помилки були усунені на етапі налагодження.

6. Оцінено результати роботи програми та сформульовано перспективи її розвитку. Зібрано відгуки користувачів, проведено порівняння з аналогами. Визначено, що система ефективна, проста у використанні та має потенціал для розширення — зокрема, через додавання синхронізації з мобільними пристроями, інтеграції з погодними сервісами та формування статистики виконання робіт.

Таким чином, усі поставлені завдання було виконано повністю, а розроблене програмне забезпечення доцільно впроваджувати в аграрну практику.

Робота пройшла апробацію. За її результатами опубліковано наступні тези доповідей:

1. Никитенко А.О. Розробка програмного забезпечення для оптимізації агрономічних процесів. IV Міжнародна наукова конференція «Технології та суспільство: взаємодія, вплив, трансформація». 20.06.2025, м. Чернігів, Україна (подано до друку).

ПЕРЕЛІК ПОСИЛАНЬ

1. Academia.edu [Електронний ресурс] — USD academia.edu+1farmtopia.eu+1.
2. Food and Agriculture Organization [Електронний ресурс]. — fao.org+8farmtopia.eu+8agtechnavigator.com+8.
3. Microsoft. Documentation for .NET and C#. [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/dotnet/csharp/
4. SQLite Documentation. [Електронний ресурс]. – Режим доступу: https://www.sqlite.org/docs.html
5. Gurov V. S. Програмування мовою C# у середовищі .NET. — К.: Наука і техніка, 2020. — 278 с.
6. Cropio [Електронний ресурс]. – https://latifundist.com/kompanii/948-cropio
7. Krug, S. Don't Make Me Think: A Common Sense Approach to Web Usability. — New Riders, 2014.
8. Cropio – Precision Agriculture Platform. [Електронний ресурс]. – Режим доступу: https://www.cropio.com/
9. EOS Crop Monitoring. [Електронний ресурс]. – Режим доступу: https://eos.com/products/crop-monitoring/
10. Soft.Farm. Система управління агробізнесом. [Електронний ресурс]. – Режим доступу: https://www.soft.farm/
11. Visual Studio Documentation. [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/en-us/visualstudio/
12. OneSoil. Smart Farming Platform. [Електронний ресурс]. – Режим доступу: https://onesoil.ai/
13. Романюк А.І. Основи розробки інтерфейсів користувача. — К.: Видавництво Ліра-К, 2021. — 192 с.
14. Freeman A., MacDonald M. Pro ASP.NET Core MVC. — Apress, 2021. — 1080 с.
15. Ерліхман В.М. Реляційні бази даних. SQL. — СПб: Питер, 2019. — 312 с.

16. Флінн Дж. Архітектура програмного забезпечення: структура, поведінка, взаємодія. — Х.: Вид-во «Фактор», 2020. — 336 с.
17. C# Windows Forms Application Tutorial with Examples. [Електронний ресурс]. – Режим доступу: https://www.tutorialspoint.com/windows_forms/index.htm
18. Agrivi. Farm Management Software. [Електронний ресурс]. – Режим доступу: https://www.agrivi.com/
19. Visual Paradigm. What is an ER Diagram? [Електронний ресурс]. – Режим доступу: https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка програмного забезпечення оптимізації агрономічних процесів

Виконав студент 5 курсу

групи ППЗ-51

Арсеній Никитенко Олексійович

Керівник роботи

кандидат фізико-математичних наук, доцент

Садовенко Володимир Сергійович

Київ – 2025

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи** – розробка програмного забезпечення для нагадування про агрономічні роботи, яке дозволяє автоматизувати планування сезонних завдань, зменшити ризик пропуску важливих агрономічних подій та підвищити ефективність управління агровиробничими процесами.
- **Об'єкт дослідження** – процеси організації та автоматизації виконання сезонних агрономічних робіт.
- **Предмет дослідження** – програмне забезпечення системи нагадувань про агрономічні роботи, зокрема його архітектура, функціональні можливості, інтерфейс та ефективність використання.

2

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Проаналізувати існуючі підходи до автоматизації агрономічних процесів.
2. Визначити функціональні та нефункціональні вимоги до системи.
3. Спроєктувати архітектуру програмного забезпечення.
4. Реалізувати систему нагадувань у середовищі C# (Windows Forms).
5. Провести тестування програмного продукту.
6. Оцінити результати та сформулювати перспективи розвитку програми.

3

АНАЛІЗ АНАЛОГІВ

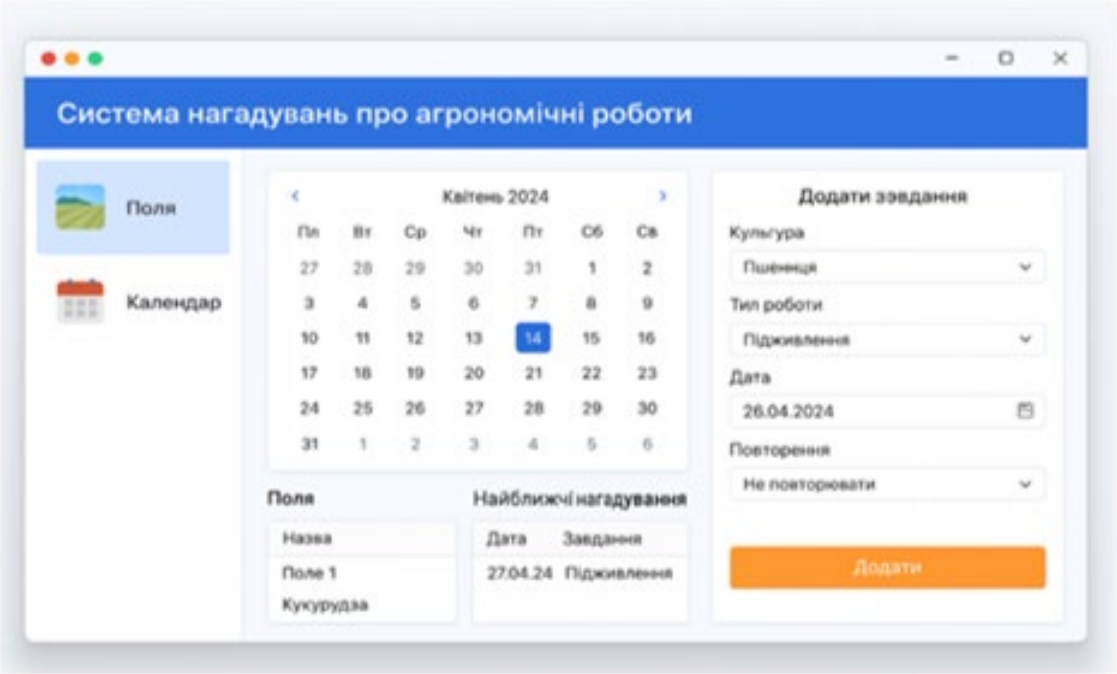
Характеристика / Рішення	**Cropio, AgroOffice, EOS Crop Monitoring**	**Agroptima, AgriApp, Сільський експерт**	**Власноручно створені (Excel, календар)**	**Запропонована програма (нагадування)**
Цільова аудиторія	Великі агрофірми	Середні / дрібні фермери	Дрібні фермери	Дрібні та середні фермери
Наявність системи нагадувань	✕	✕	✓	✓✓
Облік робіт / польовий журнал	✓✓	✓	✕ / частково	✓
Простота використання	✕	✓	✓✓	✓✓
Працює без інтернету	✕	Частково	✓	✓
Можливість локалізації / українська мова	Частково / відсутня	Частково / ✓	✓	✓✓
Автоматизація процесів	✓✓	✕ / частково	✕	✓
Ціна використання	$$$	$ / безкоштовно	Безкоштовно	Безкоштовно / умовно безкоштовно
Збереження історії обробок, культур, полів	✓✓	✓ / ✕	✕	✓
Можливість адаптації під конкретне господарство	✕	Частково	✓	✓✓

4

СХЕМА ТРИРІВНЕВОЇ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

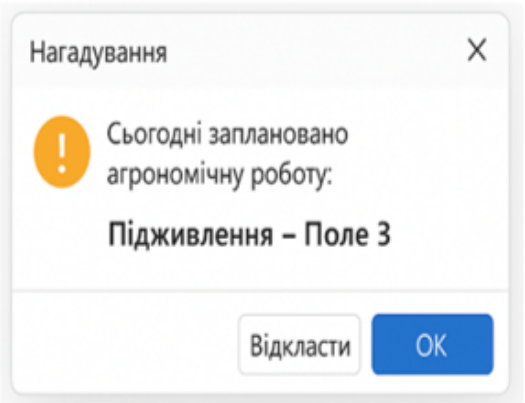
5

ІНТЕРФЕЙС КОРИСТУВАЧА СИСТЕМИ НАГАДУВАНЬ ПРО АГРОНОМІЧНІ РОБОТИ



6

ВІКНО СПОВІЩЕННЯ ПРО АГРОНОМІЧНУ РОБОТУ



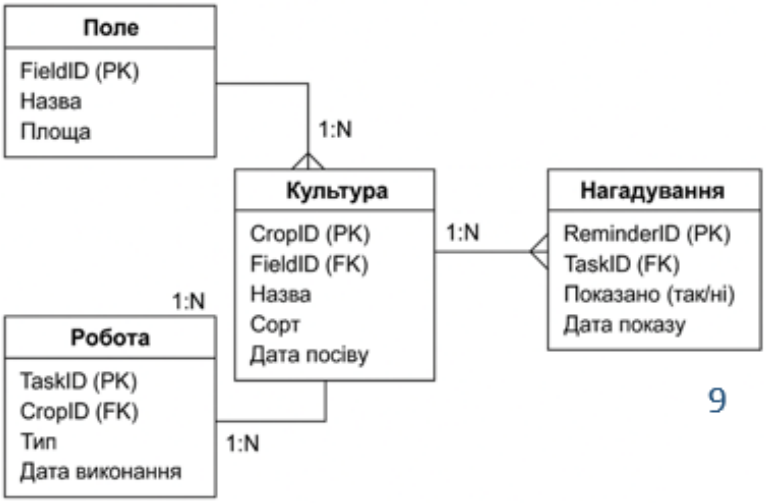
7

СТРУКТУРА ОСНОВНИХ СУТНОСТЕЙ

Сутність	Назва поля	Тип даних	Опис
Field	FieldID	Integer (PK)	Унікальний ідентифікатор
	Name	Text	Назва поля
	Area	Real	Площа в гектарах
	Notes	Text	Додаткова інформація
Crop	CropID	Integer (PK)	Ідентифікатор культури
	FieldID	Integer (FK)	Посилання на поле
	Name	Text	Назва культури
	Variety	Text	Сорт
	SowingDate	Date	Дата посіву
Task	TaskID	Integer (PK)	Ідентифікатор завдання
	CropID	Integer (FK)	Посилання на культуру
	Type	Text	Тип роботи (підживлення, обприскування)
	Description	Text	Опис
	DueDate	Date	Запланована дата
	RepeatIntervalDays	Integer (NULL)	Інтервал повторення (якщо є)
Reminder	ReminderID	Integer (PK)	Ідентифікатор нагадування
	TaskID	Integer (FK)	Посилання на завдання
	IsShown	Boolean	Було показане (true/false)
	ShownDate	DateTime	Коли було показано

8

ER-ДІАГРАМА



9

ВИКОРИСТАНІ ІНСТРУМЕНТИ ТА БІБЛІОТЕКИ

Компонент	Призначення
C# (.NET Framework)	Основна мова програмування
Windows Forms	Графічний інтерфейс (UI)
SQLite	Легка локальна база даних
System.Data.SQLite	Підключення бази даних до C#
DateTimePicker	Вибір дат у формах
MessageBox	Виведення нагадувань

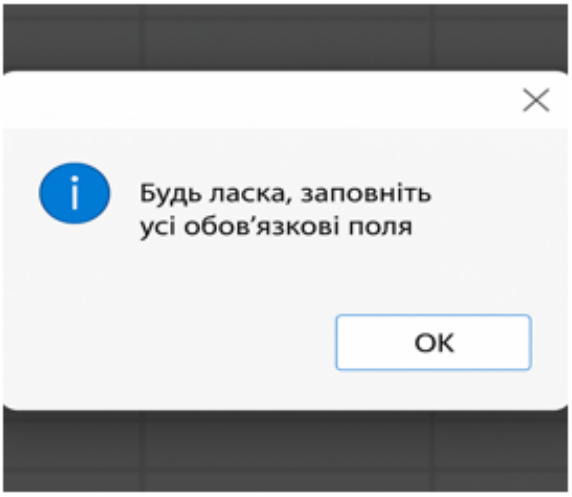
10

СЦЕНАРІЇ ТЕСТУВАННЯ

№	Назва тесту	Кроки тестування	Очікуваний результат	Фактичний результат
1	Додавання події	Заповнити форму даними та натиснути 'Зберегти'	Подія зберігається в БД та відображається	Успішно
2	Порожні поля	Натиснути 'Зберегти' без заповнення полів	Система видає повідомлення про помилку	Успішно
3	Автоматичне нагадування	Створити подію на сьогодні, запустити програму	З'являється повідомлення-нагадування	Успішно
4	Граничне значення	Ввести 100 символів у назву культури	Поле обмежується, інтерфейс не спотворюється	Довга назва обмежена вручну
5	Перезапуск системи	Додати подію, закрити і перезапустити програму	Подія не втрачається	Успішно

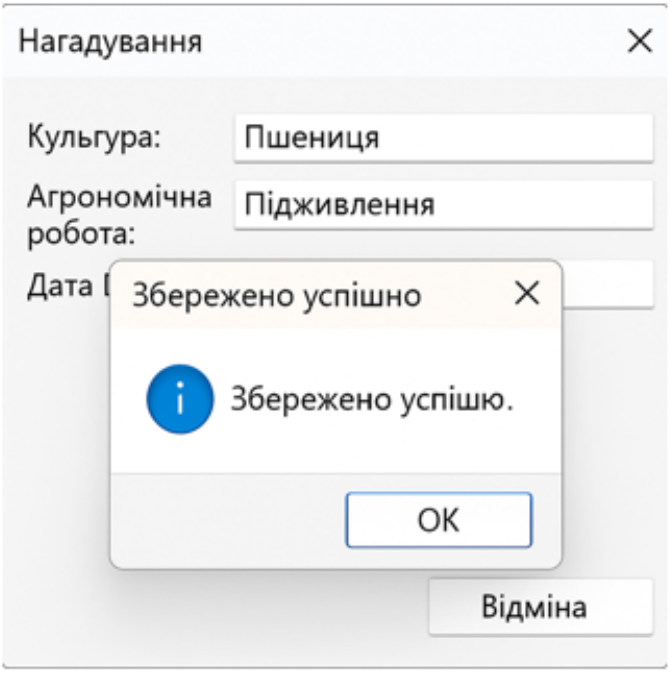
11

ПОВІДОМЛЕННЯ ПРО ПОМИЛКУ ПРИ ЗБЕРЕЖЕННІ ПОДІЇ БЕЗ ЗАПОВНЕННЯ ПОЛІВ



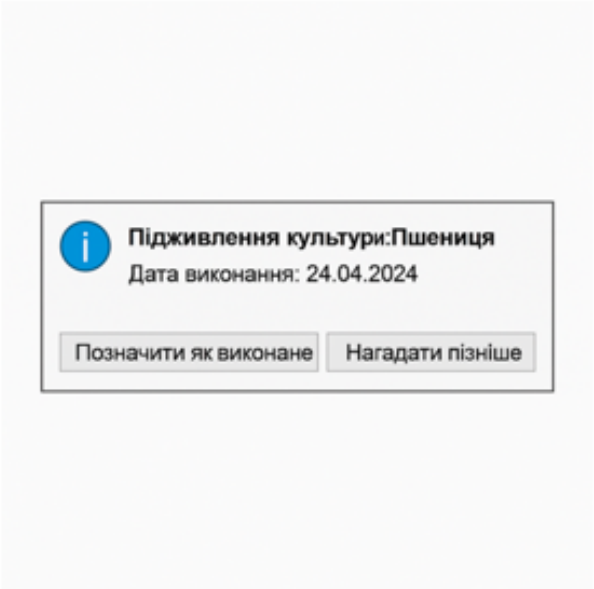
12

ПІДТВЕРДЖЕННЯ УСПІШНОГО ЗБЕРЕЖЕННЯ АГРОНОМІЧНОГО ЗАВДАННЯ У БАЗІ ДАНИХ



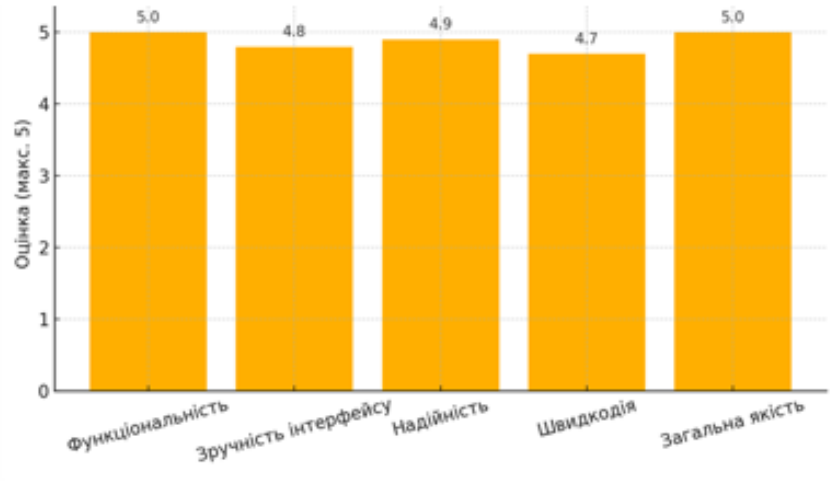
13

ВІКНО ПОВІДОМЛЕННЯ-НАГАДУВАННЯ ДЛЯ АГРОНОМІЧНОГО ЗАВДАННЯ



14

ГРАФІК ОЦІНКИ ЯКОСТІ ПЗ ЗА РЕЗУЛЬТАТАМИ ТЕСТУВАННЯ



15

ПОРІВНЯННЯ ОЦІНОК ЗРУЧНОСТІ ВИКОРИСТАННЯ АГРОПЛАТФОРМ

16

ВИСНОВКИ

1. Аналіз існуючих підходів до автоматизації агрономічних процесів дозволив виявити, що більшість сучасних рішень або надто складні для дрібного користувача, або не мають функції локального нагадування без інтернету. Це обґрунтувало потребу у простому та автономному рішенні.

2. Визначено функціональні та нефункціональні вимоги до системи.

3. Спроєктовано архітектуру програмного забезпечення. Побудовано відповідну архітектурну схему та визначено модулі системи.

4. Реалізовано систему нагадувань у середовищі C# (Windows Forms).

5. Проведено тестування програмного продукту.

6. Оцінено результати роботи програми та сформульовано перспективи її розвитку. Зібрано відгуки користувачів, проведено порівняння з аналогами.

17

ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ

Б.1 Модуль створення нагадування

```
private void btnSave_Click(object sender, EventArgs e)
{
   if (string.IsNullOrWhiteSpace(txtCulture.Text) || string.IsNullOrWhiteSpace(txtTask.Text))
   {
      MessageBox.Show("Будь ласка, заповніть усі обов’язкові поля.", "Помилка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
      return;
   }

   Reminder newReminder = new Reminder
   {
      Culture = txtCulture.Text,
      Task = txtTask.Text,
      Date = dtpDate.Value
   };

   ReminderRepository.Save(newReminder);
   MessageBox.Show("Нагадування успішно збережено!", "Успіх", MessageBoxButtons.OK,
MessageBoxIcon.Information);
}
```

Б.2 Модуль відображення нагадування

```
private void CheckReminders()
{
   var reminders = ReminderRepository.GetRemindersForToday();
   foreach (var reminder in reminders)
   {
      MessageBox.Show($"Сьогодні: {reminder.Culture} – {reminder.Task}", "Нагадування",
MessageBoxButtons.OK, MessageBoxIcon.Information);
   }
}
```

Б.3 Модель сутності Reminder

```
public class Reminder
{
   public string Culture { get; set; }
   public string Task { get; set; }
   public DateTime Date { get; set; }
}
```

Б.4 Робота з базою даних (SQLite)

```
public static void Save(Reminder reminder)
{
   using (var conn = new SQLiteConnection("Data Source=reminders.db"))
   {
      conn.Open();
      var cmd = conn.CreateCommand();
      cmd.CommandText = "INSERT INTO Reminders (Culture, Task, Date) VALUES (@c, @t,
@d)";
      cmd.Parameters.AddWithValue("@c", reminder.Culture);
      cmd.Parameters.AddWithValue("@t", reminder.Task);
      cmd.Parameters.AddWithValue("@d", reminder.Date);
      cmd.ExecuteNonQuery();
   }
}
```

Б.5 Модель сутності «Поле» (Field)

```
public class Field
{
   public int Id { get; set; }
   public string Name { get; set; }
   public double Area { get; set; } // площа в гектарах
}
```

Б.6 Модель сутності «Культура» (Crop)

```
public class Crop
{
   public int Id { get; set; }
   public string Name { get; set; }
   public string Description { get; set; }
}
```

Б.7 Завантаження нагадувань з бази

```
public static List<Reminder> GetRemindersForToday()
{
   var list = new List<Reminder>();
   using (var conn = new SQLiteConnection("Data Source=reminders.db"))
   {
      conn.Open();
      var cmd = conn.CreateCommand();
      cmd.CommandText = "SELECT Culture, Task, Date FROM Reminders WHERE Date =
@today";
      cmd.Parameters.AddWithValue("@today", DateTime.Today);

      using (var reader = cmd.ExecuteReader())
      {
```

```
            while (reader.Read())
            {
                list.Add(new Reminder
                {
                    Culture = reader.GetString(0),
                    Task = reader.GetString(1),
                    Date = reader.GetDateTime(2)
                });
            }
        }
    }
    return list;
}
```

Б.8 Ініціалізація бази даних SQLite

```
public static void InitializeDatabase()
{
    using (var conn = new SQLiteConnection("Data Source=reminders.db"))
    {
        conn.Open();
        var cmd = conn.CreateCommand();
        cmd.CommandText = @"
            CREATE TABLE IF NOT EXISTS Reminders (
                Id INTEGER PRIMARY KEY AUTOINCREMENT,
                Culture TEXT NOT NULL,
                Task TEXT NOT NULL,
                Date DATE NOT NULL
            )";
        cmd.ExecuteNonQuery();
    }
}
```