Зміст

[Перелік скорочень 3](#_Toc375657619)

[Вступ 4](#_Toc375657619)

Розділ 1

[Характеристика предметної області і постановка задачі 5](#_Toc375657620)

[1.1 Аналіз предметної області 5](#_Toc375657621)

[1.2 Аналіз інформаційного забезпечення предметної області](#_Toc375657622) 8

[1.2.1 Аналіз існуючих технологій інформаційної оптимізації інтернет-додатку](#_Toc375657629) 8

[1.2.2 Аналіз сучасних комп’ютерних платформ для реалізації програмного продукту 1](#_Toc375657629)4

[1.3 Постановка задачі](#_Toc375657623) 19

[Висновки по розділу 1 21](#_Toc375657619)

Розділ 2

[Проектування структури інформаційної системи](#_Toc375657624) 22

[2.1 Аналіз та автоматизація обробки інформаційних потоків 22](#_Toc375657621)

[2.2 Розробка структури інформаційної системи 24](#_Toc375657621)

[2.3 Вибір засобів розробки інформаційної системи 26](#_Toc375657621)

[2.3.1 Вибір засобів розробки підсистеми розробки тестових завдань](#_Toc375657629) 26

[2.3.2 Вибір СКБД для підсистеми розробки тестових завдань](#_Toc375657629) 33

[2.3.3 Принципова](#_Toc375657629) схема дисперсійного аналізу при обробці інформаційних данних у роботі бджільництва…………………………37

[2.3.4](#_Toc375657629)  Аналіз статистики роботи та візуалізація даних…………………46

[Висновки по розділу 2 49](#_Toc375657619)

Розділ 3

[Програмна реалізація](#_Toc375657626) 29

[3.1 Структура і функціональне призначення модулів, їх взаємозв’язок](#_Toc375657627) 29

[3.2 Розробка програмних модулів](#_Toc375657628) 31

[3.3 Тестування інформаційної системи](#_Toc375657635) 39

[3.4 Інструкція користувача](#_Toc375657636) 42

[3.5 Вимоги до апаратних та програмних засобів](#_Toc375657636) 47

[Висновки по розділу 3 3](#_Toc375657619)

Загальні [висновки](#_Toc375657639) 49

Перелік посилань49

[Додатки](#_Toc375657636)

# Перелік скорочень

|  |  |
| --- | --- |
| Скорочення, термін, позначення | Пояснення |
| CT | Сloud Technology |
| ОС | Операційна Система |
| SCADA | Supervisory Control And Data Acquisition |
| PHP | Personal Home Page |
| AJAX | Asynchronous JavaScript і XML |

Вступ

Із розвитком інноваційних технологій науково-технічний процес торкнувся усіх сфер людської діяльності. Масова комп’ютеризація та розвиток ефективних інформаційних технологій привели на порозі третього тисячоліття до якісної зміни інформаційної складової розвитку сфер виробництва, науки, соціального життя.Інформація стала, тісно пов’язана з управлінням та організацією і перетворилася в глобальний ресурс людства, багаторазово збільшуючи його потенційні можливості в усіх сферах життєдіяльності.

Одним із пріоритетних напрямків інформатизації суспільства стає процес впровадження програмного забезпечення у бджільництві, який передбачає використання можливостей нових інформаційних технологій, методів та засобів інформатики для реалізації ідей розвиваючого господарства, інтенсифікації усіх видів виробничих процесів у бджільництві, підвищенню ефективності і якості підготовки вуликів до комфортних умов для отримання великого взятку. Для цього на заміну стереотипним паперовим журналам-заміткам бджільника, які за звичай загублювались або з часом просто набували не читабельному стану, було використано інформаційні технології.

Стрімкий стрибок, у розвитку персональних комп’ютерів, як і завдяки використання інтернету, за останні роки зробив їх доступними, що зумовило, підкорити сферу фермерського господарства. Тому впровадження комп’ютерних технологій в робочий процес можна охарактеризувати як логічний і необхідний крок у розвитку сучасної інформаційно забезпеченості бджільництва в світі.

**Мета і задачі дослідження**. Метою роботи є дослідження технологій створення та впровадження комп’ютерних технологій в робочий процес фермерського господарства бджільництва. Для досягнення поставленої мети необхідно вирішити цілий комплекс завдань, проведення аналізу можливостей мови ASP.Net, що надаються web-розробнику.

*Об’єкт дослідження* – Об'єктом дослідження є хмарні технології, web та статистичний робочий процес фермерського господарства бджільництва.

*Предмет дослідження* – Предметом дослідження роботи є адаптація статистичних робіт на web у використані хмарних технологіях.

**Наукова новизна одержаних результатів.**

Вперше запропоновано використання хмарних технологій, в часності CT, в роботі web для статистичних робіт фермерського господарства бджільництва, що опирається на зроблені висновки дослідження в даній роботі.

**Практичне значення одержаних результатів.**

Практична реалізація, заснована на результатах проведеного аналітичного дослідження, являє собою web-додаток. Даний web-додаток дозволяє користувачу не тільки здійснити огляд найбільш цікавих функціональних можливостей ASP.Net і JavaScript, але також стати безпосереднім учасником процесу формування електроного журнала, динамічно і статистично вести роботу над web-додатком з метою візуального відображення роботи над бджільництвом.

Розділ 1

Характеристика предметної області і постановка задачі

1.1 Аналіз предметної області

Бджільництво – це спеціально обладнане місце, у якому розміщенні вулики з медоносними бджолами. Іншими словами область сільського господарства, яка призначена для утримання і розведення бджіл з метою отримання таких продуктів як: мед, пилок, прополіс, віск, перга, маточне молочко, гомогенад та інших цінних продуктів бджільництва.

В бджільництві головним обладнанням є вулик. Так як частина успіху залежить від його характеристик, наскільки сприятиме розвитку сім’ї та від зручності роботи з ним. Вони бувають різними за розміром, за типом конструкції: нерозбірні, розбірні, лінійні, рамкові, горизонтальні, вертикальні, багатокорпусні і в усіх різна кількість рамок. Але, так як гнізда у бджіл бувають досить високі, тому рамки у вуликах строго розташовані по вертикалі. Розташування вулика не менш важливе. При виборі місця для пасіки варто знати, що медозбір відбувається на території діаметром 2-3 кілометри, тому варто подбати про наявність цвіту. І необхідно пам’ятати, що місце перебування бджолиних сімей не повинно перетинатися з іншими сусідніми пасіками - це може спровокувати боротьбу і знищення бджіл, а також крадіжку меду. Для розміщення пасіки поблизу людей потрібно огороджуватись парканом заввишки не менше 2 метра. При розташуванні вуликів на території пасіки потрібно передбачити для них достатньо широкий простір. При цьому самі вулики розташовують на високих ніжках, і «лицем» направляють на південь або південний схід.

Пасіка буває не тільки стаціонарною, коли вулики стоять на землі і не змінюють своє місце розташування, а й мобільною, коли вулики знаходяться на перевізній конструкції. Така пасіка може перевозитися на віддалені території, яка дасть змогу збільшити кількості виробленого меду і його різновидність. Але тут постає проблема, що здатність догляду і спостерігання за бджолами знижується в рази.



Рисунок 1.1 – Розташування бджільництва в лісі

Як неможливе існування міста без людей, так само і бджільництво не може існувати без бджіл. Бджоли – це підвид в сімействі літаючих комах підроду стебельчатобрюхих. Бджоли наділені довгим хоботком, завдяки якому вони мають змогу висмоктувати нектар з різноманітних рослин. І двома парами крил, передніми і задніми, які надають їм велику маневреність. Бджола – це комаха, яка є прикладом відданості своїй справі. Вони здатні працювати і жертвувати собою заради розвитку і виживання своєї сім’ї. Бджола нездатна на такий вчинок як халатність, нездатна на егоїзм, від їхньої роботи залежатиме їхнє існування. Кожна бджола виконує свої обов’язки, одні захищають вулики, інші літають у поля в пошуках нектару, решта підтримують температуру вулика для розведення розплоду.

Бджола – комаха яка здатна здобувати нектар і влаштовувати свою сім’ю не знищуючи нічого і не змагаючись ні з ким. При укусі бджоли, отрута яку виділяє жало піде організму на користь, адже невелика доза бджолиної отрути принесе користь слабкому організму і допоможе вилікувати його від багатьох хвороб. Ще одне із цінних властивостей бджоли є її здатність сприянню запиленню рослин. Завдяки бджолам запиленість допомагає сільськогосподарським рослинам підвищувати врожайність.

Робота на бджільництві весною

Робота на бджільництві

літку

Робота на бджільництві осінню

Перегляд вуликів після зимування

Перегляду оцінювання кількості меду

Зменшення кількості сімей

Оцінка сім’ї після зимування

Підготовка медогонки до роботи

Підкормлювання сімей для зимівлі

Очищення вуликів

Процес викачування меду

Підготовка до зимування

Підсилення сімей

Підготовка вулика до літнього сезону

Підготовка до осіннього сезону

Робота на бджільництві

по сезонах

Рисунок 1.2 – Схема роботи пасіки у відповідності зі зміною сезонів

Робота на бджільництві обов’язково повинна простежуватись людиною. Без догляду, бджоли вам, втечуть або в найгіршому випадку просто загинуть. Тому для збереження цілісності пасіки потрібно обслуговувати бджіл, задовольняти їхні потреби. Обслуговувати пасіку мусить компетентна людина – пасічник. Бджільник – людина, яка професійно використовує свої знання в області пасіки, для утримання і розведення бджіл. Так як і самі бджоли, пасічник піклується про бджіл, так як це є його задоволення і дохід. Кількість роботи, яку виконує пасічник на бджільництві, досить значна. ЇЇ можна розділити на сезони: весняний, літній та осінній.

У весняному періоді виконується робота з оглядом вуликів, як вони перезимували, і підготовкою до літнього. В літньому періоді виконується робота зі збору бджолиної продукції. В осінньому періоді йде підготовка до зимування бджіл. Але робота, яка їх об’єднує, це огляд вуликів, відслідковування та аналізування їхньої роботи. Для такої роботи пасічник використовує щоденник. Щоденник знаходиться у кожному вулику. У ньому ведуться відмітки про огляди, а саме день огляду і дії які виконуються. В щоденник записується робота і стан вулика. Щоб побачити щоденник, який відслідковує роботу, і використати для аналізу вулика, пасічник змушений відкривати кожен вулик. Це є незручно і досить довго, що забирає багато часу для звичайного огляду.

* 1. Аналіз інформаційного забезпечення предметної області

### 1.2.1 Аналіз існуючих технологій інформаційної оптимізації інтернет-додатку

В арсеналі сучасного бджоляра є безліч способів які дають можливість організувати взаємодію з бджолами найбільш комфортно. Комп'ютерні програми не тільки спрощують ведення обліку стану бджолосімей на бджільництві, але дозволяють прогнозувати з певною часткою ймовірності подальшого розвитку подій.

Існують програми різного рівня складності та призначення, але в більшості своїй вони досить універсальні і можуть бути корисними для бджолярів, що застосовують різні технології розведення і утримання.

Інший спосіб, що дозволяє значно знизити витрати праці при обслуговуванні бджолосімей, заснований на застосуванні електромагнітних полів заданої напруженості та частоти для управління їхнім станом. Але ось доцільність використання електромагнітних дій для спрощення певних видів робіт, у значній мірі залежить від методів, на які спирається бджоляр у своїй роботі.

Тому це є однією з причин, по якій управління сім'ями за допомогою електромагнітних випромінювань не отримало широкого розповсюдження у бджільництві. Технології, які пропагувалися в літературі досі вважаються перспективними, погано поєднуються з методами електромагнітного впливу і тому не використовують для полегшення робіт з обслуговування бджолиних сімей.

І поєднання з сучасними методами роботи за пасікою, які не є трудомісткими, управління станом бджолиних сімей із застосуванням електроніки, роблять роботу на бджільництві ще простіше. І така робота підвищується в ефективності. Програми які використовуються на бджільництві переважно призначені для замінити звичайного щоденника. Такі програми як Apiary, Airbees, Бджола 1, Бджола 2 Профі.

Дана програма дозволяє здійснювати внесення даних міток, сімей і прив'язувати їх до корпусів (вуликів), а також вносити дані огляду родини. Крім цього є можливість налаштовувати програму під свої потреби. Для внесення даних є обов'язкові поля, але є можливість внесення власного поля, які необхідні конкретному пасічнику, так як скільки бджолярів, стільки й варіантів заповнення щоденника. Всі дані заносяться в локальну базу даних.

Недоліками програми зокрема є те що, при внесенні даних працювати можна тільки з клавіатурою, мишка не працює, в той же час повної підтримки клавіатури немає, після занесення даних потрібно обов'язково натиснути на кнопку зберегти або скасувати, оскільки кнопка Enter не спрацює. Крім описаних випадків є й багато інших вузьких місць, але в цілому програмою можна користуватися.

Російська версія програми для бджолярів Airbees. У даній програмі міститься довідник «Медоносні рослини», на його основі можна зробити карту свого регіону, що містить розташування виростання даних рослин, а ще побудувати графіки цвітіння їх. Виходячи з цього, можна вже створити власну базу даних розташування пасік. Ще в цій програмі є щоденник спостережень за контрольним вуликом та льотної діяльністю бджіл. Можна робити прогнози формування погоди, початку цвітіння медоносів. Крім цього, можна постійно оновлювати базу даних програми шляхом завантаження оновлень у вигляді довідників, карт тощо.

Недоліками цієї програми є те, що неможливо зробити записи на довгий час, перебування можливо визначити, але це незакономірно, так як рослини можуть зникнути, необхідно періодично робити запис і їздити, а по поверненню додому ввести дані в базу даних програми «Airbees». Програма, швидше, підходить для бджолярів-початківців, тому носить більш пізнавальний і проектувальний характер.

Програма Бджола 2. Розрахована на всі варіанти рамок і вуликів, передбачено багатокорпусна система з будь-якими рамками і кількістю. В одному корпусі від 1 до 32 рамок; передбачено двох-маточне бджільництво, постановка розділової рамки (діафрагми) після будь рамки, є можливість враховувати різні породи маток в одному двох-матковому вулику; і вставка розділових ґрат між будь-яким корпусом. У програмі враховується порода матки, вік матки, походження матки, проставляються дати оглядів, дати перестановки корпусів, дати установки корпусів. Корпуси у вулику переставляється автоматично, як в одному вулику, так і в декількох вуликах відразу, за бажанням бджоляра. Оцінка бджільницьких сезонів у порівнянні з попередніми роками.

У графі примітка можна записати будь-яку необхідну інформацію, на розсуд бджоляра. Примітки передбачені окремо по кожному корпусу, окремо з огляду, і окремо по вулику. Групуються сім'ї по меду, розплоду, спаданням показників; за кольором вулика (рій, відводок, медовик і. т.д.); за останнім оглядом, за датою оглядів, по породі матки, за віком матки. Може показати будь-який обраний вулик в розрізі. Групує сім'ї за віком матки, по породі матки. Видає розвиток бджолиної сім'ї у вигляді графіка на підставі оглядів і попереджає бджолярів про наближення підготовці сім'ї до роїння. За запитом бджоляра, програма видасть всю кількість товарного меду, зібрану бджолами пасіки за сезон і зможе порівнювати з попередніми роками. Є прогнозування витрати кормових запасів під час зимівлі, завдяки чому бджоляр з осені може знати, коли і в якій сім'ї закінчиться запаси кормів, тим самим запобігти восени загибель бджіл від голоду, або під час зими підгодувати бджіл помадкою.

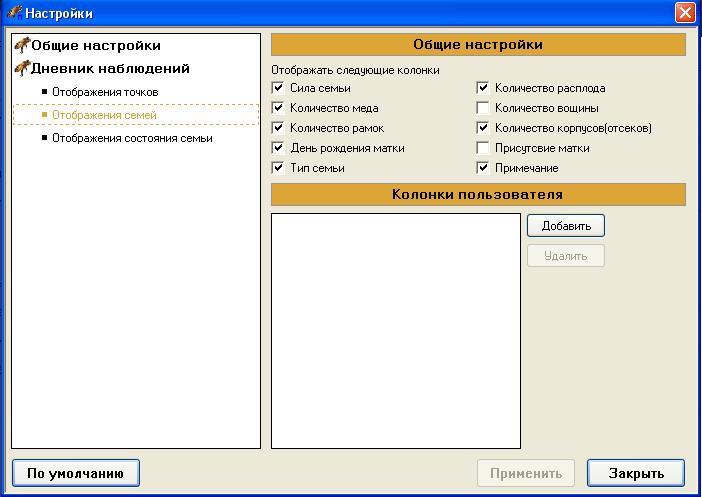


Рисунок 1.3 – Інтерфейс програми “Apiary”

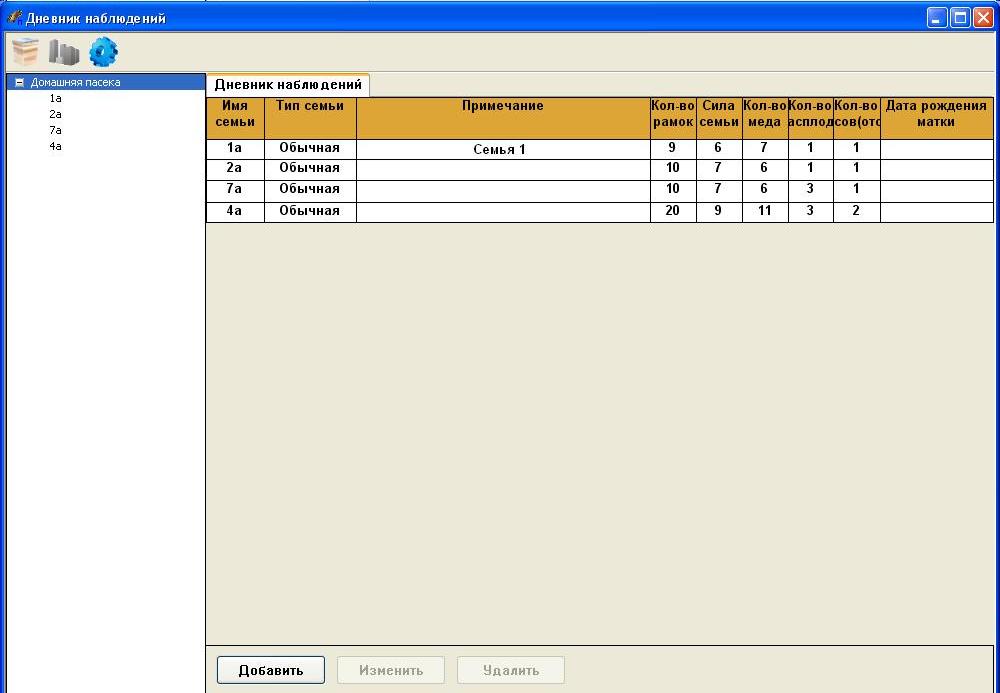


Рисунок 1.4 – Заповнений щоденник “Apiary”

Програма Airbees.

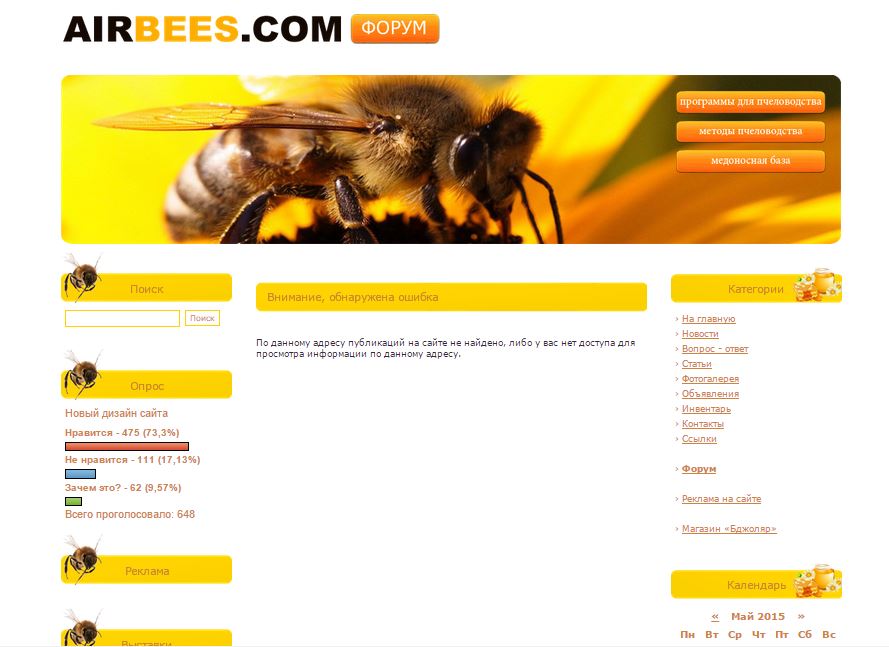


Рисунок 1.5 – Сайт програми “AirBees”

Російська версія програми для бджолярів Airbees. У даній програмі міститься довідник «Медоносні рослини», на його основі можна зробити карту свого регіону, що містить розташування виростання даних рослин, а ще побудувати графіки цвітіння їх. Виходячи з цього, можна вже створити власну базу даних розташування пасік. Ще в цій програмі є щоденник спостережень за контрольним вуликом та льотної діяльністю бджіл. Можна робити прогнози формування погоди, початку цвітіння медоносів. Крім цього, можна постійно оновлювати базу даних програми шляхом завантаження оновлень у вигляді довідників, карт тощо.

Недоліками цієї програми є те, що неможливо зробити записи на довгий час, перебування можливо визначити, але це незакономірно, так як рослини можуть зникнути, необхідно періодично робити запис і їздити, а по поверненню додому ввести дані в базу даних програми «Airbees». Програма, швидше, підходить для бджолярів-початківців, тому носить більш пізнавальний і проектувальний характер.

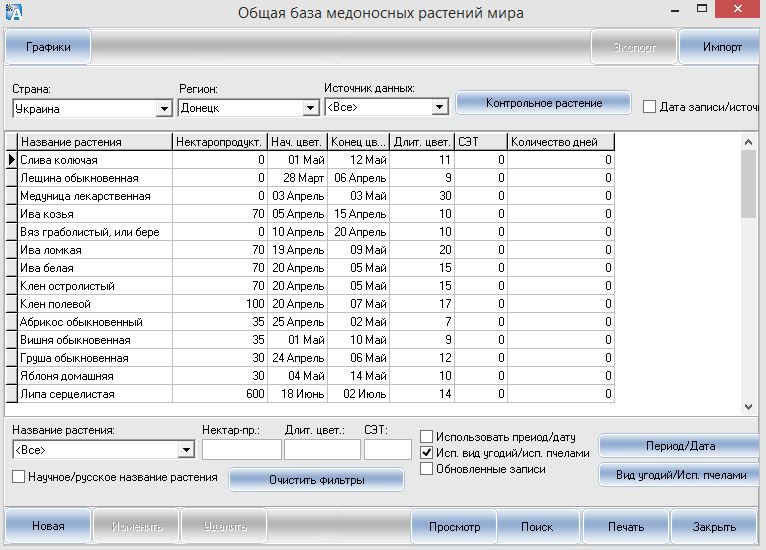


Рисунок 1.6 – Спільна база медоносних рослин

Програма Бджола 2. Розрахована на всі варіанти рамок і вуликів, передбачено багатокорпусна система з будь-якими рамками і кількістю. В одному корпусі від 1 до 32 рамок; передбачено двох-маточне бджільництво, постановка розділової рамки (діафрагми) після будь рамки, є можливість враховувати різні породи маток в одному двох-матковому вулику; і вставка розділових ґрат між будь-яким корпусом. У програмі враховується порода матки, вік матки, походження матки, проставляються дати оглядів, дати перестановки корпусів, дати установки корпусів. Корпуси у вулику переставляється автоматично, як в одному вулику, так і в декількох вуликах відразу, за бажанням бджоляра. Оцінка бджільницьких сезонів у порівнянні з попередніми роками.

У графі примітка можна записати будь-яку необхідну інформацію, на розсуд бджоляра. Примітки передбачені окремо по кожному корпусу, окремо з огляду, і окремо по вулику. Групуються сім'ї по меду, розплоду, спаданням показників; за кольором вулика (рій, відводок, медовик і. т.д.); за останнім оглядом, за датою оглядів, по породі матки, за віком матки. Може показати будь-який обраний вулик в розрізі. Групує сім'ї за віком матки, по породі матки. Видає розвиток бджолиної сім'ї у вигляді графіка на підставі оглядів і попереджає бджолярів про наближення підготовці сім'ї до роїння. За запитом бджоляра, програма видасть всю кількість товарного меду, зібрану бджолами пасіки за сезон і зможе порівнювати з попередніми роками. Є прогнозування витрати кормових запасів під час зимівлі, завдяки чому бджоляр з осені може знати, коли і в якій сім'ї закінчиться запаси кормів, тим самим запобігти восени загибель бджіл від голоду, або під час зими підгодувати бджіл помадкою.

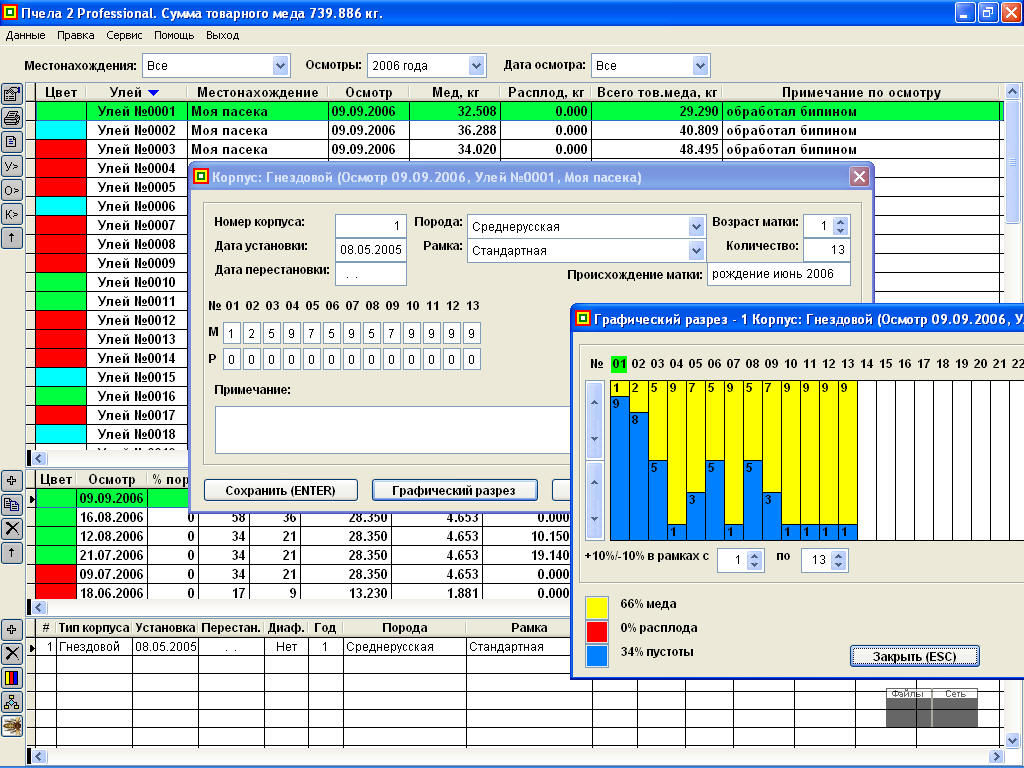


Рисунок 1.7 – Інтерфейс програми Бджола 2 Profesional

Недоліками такої програми є нездатність переносити свою базу на інші комп’ютери, прив’язаність до робочого місця за комп’ютером; за орієнтир по меду і розплоду беруться середні по весні бджолосім'ї, і саме за цими критеріями вирівнюється бджільництво, що й дає сумарне значення; використовується прогноз корму на зиму, що дає змогу передбачити, але не враховуються усі можливі параметри, і немає змоги підтримувати стеження за наявністю корму; складність у використані.

Крім перерахованих вище популярних вітчизняних програм для бджолярів, існують не менш корисні й зручні іноземні програми: BeeKeeper 3.0 (Канада, інтерфейс англійською), Apilogic (Франція), BeeAware2000 (США), VarroaPop 2.1 (США), "Янтарний стільник" (Україна), Imkerbuch für Windows (Німеччина), IMKI 1.0 (Німеччина), Bidata (Данія), BeeMine, Xen-APIARY ™, "L-Hive" та деякі інші.

Недоліком таких програм є те, що вони є надзвичайно дорогі у придбанні та експлуатації.

Таблиця 1.1 – Аналіз існуючого програмного забезпечення

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Програми** | **візуальне забезпечення** | **можливість відаленого доступу** | **зручність у використані** | **функціональна забезпеченість** |
| **Бджолий дім** | + | + | +- | + |
| **Бджола 2** | +- | - | +- | + |
| **Airbees** | +- | - | +- | +- |
| **BeeKeeper 3.0** | + | - | + | - |
| **Янтарний стільник** | +- | - | +- | +- |

Тут «+» - забезпечено в повному обсязі, «+-» - забезпечено частково, «-» - не забезпечено (не існує).

### 1.2.2 Аналіз сучасних комп’ютерних платформ для реалізації програмного продукту

На сьогоднішній день у розробників веб-сайтів є багатий вибір щодо того, яку мову (або технологію) використовувати для створення сайту. Варіантів багато, це може бути: PHP, ASP.Net, ASP.NET Core, JSP, Python, Ruby, Perl тощо.

Серверні мови веб-програмування можуть бути умовно розділені по операційній системі, на якій вони переважно працюють: Windows і \*nix. Це розділення в деякій мірі умовно, тому практично всі популярні мови і фреймворки портовані на обидві групи ОС.

Для ОС Windows безроздільно панує технологія ASP.NET, розроблена компанією Microsoft. За допомогою ASP.NET можна створювати сайти будь-якого рівня складності - від найпростіших, що складаються із декількох сторінок, до дуже складних, що обробляють мільйони запитів на день (сайти написані на ASP.NET є одними з найбільш відвідуваних в Інтернеті). Технологія ASP.NET приваблива для тих, хто непогано розбирається в ОС Windows, але незнайомий з Unix-подібними системами. Основний недолік - кількість дешевих хостингів або необхідність покупки серверної ліцензії, у випадку з виділеним хостингом. Але у порівнянні з вартістю розробки складних сайтів і вартістю трафіку, різниця витрат на Windows- і \*nix-хостинг може бути практично мала.

Найпопулярнішою мовою веб-програмування є PHP. PHP розшифровується, як препроцес гіпертексту, і є мовою, що працює на стороні сервера. Це означає, що, коли відвідувач відкриває сторінку, сервер обробляє команди PHP, а потім відправляє результати в браузер відвідувача, як і у випадку з технологією ASP. Типові файли PHP будуть складатися з команд, які будуть виконані на сервері, в додаток до звичайної суміші тексту і HTML-тегів (Hypertext Markup Language). Однак, PHP є відкритим вихідним кодом і крос-платформний. PHP працює на багатьох версіях \*nix і Windows NT і може бути використаний і у вигляді модуля Apache, і як двійковий файл, який може працювати як CGI.

Синтаксис мови PHP схожий на C і в Perl. Змінні не потрібно оголошувати, перш ніж їх використовувати, так і для створення масивів і хешів (асоційованих масивів). PHP навіть має певні об'єктно-орієнтовані можливості, забезпечуючи зручний спосіб організації і інкапсуляції коду. За його основного синтаксису, PHP також може похвалитися широким спектром інтерфейсів, що дозволяють йому спілкуватися з усім – від інших веб-сторінок, до баз даних, включаючи ODBC, та інші мови програмування, такі як Java або COM. Код PHP пишеться прямо у веб-сторінках, так що не має необхідності в спеціальному середовищі розробки або IDE. Велика перевага PHP – це написання багатьох популярних рушіїв, зокрема один із них - найпопулярніший рушій для stand-alone блогів WordPress.



Рисунок 1.8 – Приклад роботи PHP з сервером

Ruby - це ретельно збалансована мова. Її називають красивою і майстерною мовою і в той же час зручною і практичною.

 Її творець Юкіхіро Мацумото (відомий як "Matz"), об'єднав частини мов (Perl, Smalltalk, Eiffel, Ada і Lisp), щоб сформувати нову мову, в якій парадигма функціонального програмування збалансована принципами імперативного програмування.

Ruby грунтується на такому вислові: «проста на вигляд, але дуже складна всередині, подібно до людського тілу».

Індекс TIOBE, який вимірює зростання популярності мов програмування, поміщає Ruby на 9 місце серед загальновизнаних мов програмування. Зростання популярності мови багато в чому приписується популярності програмного забезпечення, написаного на Ruby, особливо – Ruby on Rails, середовищу розробки веб-додатків.

Ruby також абсолютно відкритий. Відкритий для безкоштовного використання, змін, копіювання та розповсюдження.

В Ruby все - об'єкт. Для кожної частки інформації або коду можуть бути визначені власні властивості і дії. В об'єктно-орієнтованому програмуванні властивостями називаются змінні об'єкта, а дії - методами. Найчистіше об'єктно-орієнтований підхід Ruby може бути продемонстрований парою рядків коду, в яких виконується дія над числом.

У багатьох мовах числа та інші примітивні типи даних не є об'єктами. Ruby під впливом мови Smalltalk дозволяє задати методи і змінні об'єкта всім типам даних. Це спрощує використання Ruby, так як правила застосовні до об'єктів - застосовні до всієї Ruby.

Ruby дуже гнучка мова, яка дозволяє її користувачам вільно змінювати її частини. Основні частини Ruby можуть бути вилучені або перевизначені за бажанням, а існуючі частини можна модифікувати. Ruby намагається ні в чому не обмежувати користувача.

Оператори в Ruby - синтаксичний «цукор» для методів. Програміст також може перевизначати їх.

Блоки в Ruby також є чудовим джерелом гнучкості. Програміст може додати замикання до будь-якого методу, описуючи, як цей метод повинен діяти. Замикання називається блоком і є однією з найбільш популярних конструкцій, що прийшли в Ruby з імперативних мов програмування, таких як PHP або Visual Basic.

На відміну від багатьох об'єктно-орієнтованих мов програмування, Ruby надає лише єдине наслідування, але Ruby також надає концепцію модулів.

Ruby сповнена іншими особливостями і конструкціями, і ось деякі з них.

1. В Ruby є конструкції для обробки виключень, як в Java або Python, які дозволяють простіше працювати з помилками.
2. В Ruby представлений справжній mark-and-sweep (познач-і-відчисти) збирач сміття для всіх Ruby-об'єктів. Не потрібно вручну відстежувати кількість посилань в сторонніх бібліотеках.
3. Писати розширення на C в Ruby простіше, ніж в Perl або Python, за допомогою дуже елегантного API для виклику Ruby з C. Він включає в себе виклики для вбудовування Ruby в програмне забезпечення, щоб використовувати її, як скриптову мову. Також доступний інтерфейс SWIG.
4. Ruby може довантажувати сторонні бібліотеки динамічно, якщо дозволяє операційна система.
5. В Ruby реалізовані незалежні від операційної системи потоки. Таким чином, на будь-яких платформах, де запускається Ruby, також є можливість використовувати багатопоточність, не залежно від того, чи підтримує дана система потоки чи ні.
6. Ruby відрізняється високою переносністю.

Інша популярна мова веб-програмування на платформі Unix - мова Perl. Вона має складний заплутаний синтаксис і ніколи не призначалася для веб-програмування. Не рекомендується її використовувати для створення сучасного сайта.

JSP (Java Server Pages) - це частина технології J2EE, призначена для створення сайту за допомогою мови Java. JSP має дуже багато спільного з ASP.NET, і вибір між цими двома технологіями найчастіше грунтується не на будь-яких перевагах чи недоліках цих платформ, а на суб'єктивних перевагах.

Платформа ASP.NET Core представляє технологію від компанії Microsoft, призначену для створення різного роду веб-додатків: від невеликих веб-сайтів до великих веб-порталів і веб-сервісів.

З одного боку, ASP.NET Core є продовженням розвитку платформи ASP.NET. Але з іншого боку, це не просто черговий реліз. Вихід ASP.NET Core фактично означає революцію всієї платформи, її якісна зміна.

ASP.NET Core тепер повністю є opensource-фреймворком. Всі вихідні файли фреймворку доступні на GitHub.

ASP.NET Core побудований на основі крос-платформної середовища .NET Core, яка може бути розгорнута на основних популярних операційних системах: Windows, Mac OS X, Linux. І хоча Windows як середовище для розробки і розгортання програми досі превалює, але тепер вже ми не обмежені тільки цією операційною системою. Тобто ми можемо запускати веб-додатки не тільки на ОС Windows, але і на Linux і Mac OS. А для розгортання веб-додатки можна використовувати традиційний IIS, або крос-платформний веб-сервер Kestrel.

Хоча ASP.NET Core переважно націлене на використання .NET Core, але фреймворк також може працювати і з повною версією фреймворка .NET.

Будучи повністю відкритим вихідним кодом не є легким завданням, Microsoft зробила деякі дивовижні роботи по змусити його працювати через Windows, Mac, Linux. блог ScottGu на Вводячи ASP.NET 5 відмінно читання, щоб зрозуміти його особливості.

Різниця між ASP.NET MVC і ядра ASP.NET MVC 5 в 10 точках

Різниця 1 - вирівняно веб-стека для ASP.NET MVC Core і веб-інтерфейсів.

ASP.NET MVC 5 дасть нам можливість вибору MVC або Web API або як при створенні веб-додатки. Це тому, що веб-стек для MVC 5 і Web API не було те ж саме.

Різниця 2 - Проект (Solution) Зміна структури

Якщо ви бачите ASP.NET MVC Core рішення провідника з правого боку, немає Web.config, Global.asax. Тоді як вона має справу з параметрами конфігурації, перевірки автентичності та запуску додатка, конкретне виконання коду.

Project.json, appsettings.json деякі файли, які роблять ці роботи відсутніх файлів з ASP.NET MVC 5. Є багато змін, якщо ми подивимося на папці по папках.

Різниця 3 - Про крос-платформний ! Так, тепер ми можемо розробити ASP.NET Ключових веб-додатків проти Core.NET і працювати в будь-якій ОС Windows або Linux або Mac.

Різниця 4 - ASP.NET Core Основні програми не потрібно IIS для хостингу

1.3 Постановка задачі

Завданням даної кваліфікаційної роботи бакалавра є створення системи автоматизації ведення господарства пасіки. Для виконання завдання автоматизації було визначено наступні завдання:

- аналіз предметної області та інформаційного забезпеченння в галузі сільського господарства;

- визначення переваг та недоліків програмних продуктів в галузі сільського господарства, а саме господарства ведення пасіки;

- створення методу автоматизації процесів аналізу та систематизації робіт на бджільництві для створення повноцінного програмного комплексу;

- розробка структури програмного комплексу автоматизації процесів ведення господарства пасіки на основі методу автоматизації процесів аналізу та систематизації робіт на бджільництві для створення повноцінного програмного комплексу;

- визначення базових елементів структури програмного комплексу автоматизації процесів ведення господарства пасіки із метою подальшого створення програмного продукту;

- створення програмного продукту автоматизації процесів ведення господарства пасіки;

- розробка інструкції користувача.

Необхідність створення сайту пояснюється, в першу чергу, потребами на бджільництві швидкого внесення змін в щоденник, які дають змогу переглядати і аналізувати не будучи в бджільництві. Такі записи не можливо загубити, і це друга необхідність.

Оскільки веб-сайт розроблювся для бджільництва, то у ньому потрібно передбачити можливість реєстрації візитів клієнтів, що дає змогу ширше використовувати сайт та щоденник, що дасть змогу заповнювати усі роботи виконані на бджільництві, інформацію про допомогу, яка дає змогу початківцю навчитись працювати за пасікою, також можливість збереження клієнтів у базі та їх роботи. Такий веб-сайт повинен полегшувати процес роботи у бджільництві, слідкувати за рухом і аналізувати роботу для покращення і не допускати втрати сімей.

Принцип роботи даної предметної області буде полягати в залученні нових і утриманні старих користувачів, отримання прибутку від наданих послуг. Застосування щоденника має за мету підвищення швидкості обробки інформації, cпрощення ведення роботи пасіки щодо перегляду і аналізу.

Метою створення даного дипломного проекту є легкодоступність, простота роботи, низька вартість, і головне – висока якість опрацювання оперативної інформації. Зазначений веб-сайт призначений для заміни звичайного паперового щоденника на електронний. Що дає можливість дистанційно управляти окремими процесами, підвищувати ефективність діяльності суб'єкта господарювання, знижувати трудомісткість праці, зменшувати затрати праці.

Отже, постановкою завдання буде веб-реалізація автоматизованої пасіки для швидкого і зручного використання її та для отримання максимальної користі з наступними функціями та можливостями:

- наочність та простота;

- перегляд інформаційних контекстів;

- реєстрація клієнтів;

- створення особистої пасіки;

- створення можливості редагування вуликів;

- надійність;

- створення різноманітних звітів по роботі пасіки;

- створення інтерфейсу для роботи зручного робочого місця пасічника;

- розширюваність і масштабованість системи;

- фіксування різного типу роботи у календарі;

- аналізування проведених робі у бджільництві;

- забезпечення кросбраузерності кінцевого продукту;

- створення тематичного дизайну, логотипу, назви та доменного імені сайту, що відповідає предметній області;

- можливість коригування усіх даних;

- створення інформаційної бази, для успішного використання пасіки;

- легкість подальшої модифікації.

**Висновки по розділу 1**

Ідея доступних комп’ютерних послуг в бджільництві стає реальністю.

Можливості ведення інформаційних технологій дозволяють розв’язувати завдання бізнесу та надає реальні перспективи для інформаційних технологій у бджільництві. Інтернет сервіси обробки даних отримують можливість надавати свої послуги більшій кількості користувачів. Пасічники можуть думати про нові генерації своїх продуктів.

Передбачають, що масової міграції комерційних структур у публічні використання не буде, повної відмови від власних ручних записів також не передбачається, але пасічники прийдуть до гібридної моделі, де збережуться обидва елементи.

Розділ 2

Проектування структури інформаційної системи

2.1 Аналіз та автоматизація обробки інформаційних потоків

Процес проектування системи пов’язаний з розробкою і прийняттям складних рішень. Тому при проектуванні часто виникає, потреба складання і корекції, різних ескізів системи, які допомагають осмислити її призначення, оглянути систему з різних точок зору і, в результаті, скласти модель системи.

Для того, щоб пред’явити закінчену модель замовнику, необхідно скласти її опис на деякій зрозумілій мові. Природні мови та мови програмування не дозволяють дати одночасно і точний і зрозумілий опис моделі складної системи. Для цього зручніше використовувати, візуальні мови, які представлятимуть основну інформацію про модель за допомогою графічних символів і зв’язків між ними, - мови моделювання. Однією із них є UML. Застосування візуальної мови робить, можливим відстеження нетривіальних зв’язків між компонентами системи і полегшує роботу зі створення моделі. Однак існує інформація, яку все-таки легше виразити за допомогою звичайного тексту.

Проектування, інформаційної системи – складна задача. Ідеальним був би опис усієї системи за допомогою однієї схеми, яка однозначно визначає властивості системи, і при цьому проста для розуміння і використання. Опис комплексної системи проводиться ,шляхом окремого розгляду різних її ,аспектів. Ці аспекти поділяються на три категорії: функціональні, нефункціональні й організаційні. Функціональні, (статична структура і динамічна взаємодія окремих компонентів, структури), нефункціональні ,(продуктивність, надійність, розподіл ресурсів тощо) і організаційні (розподіл роботи, перенос моделей у реальний код тощо). Таким чином, інформаційна система на етапі проектування зазвичай описується з різних поглядів або точок зору, які відповідають різним аспектам функціонування системи.

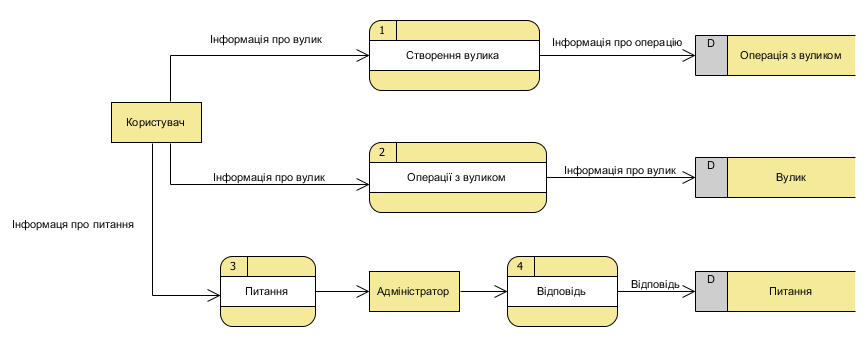


Рисунок 2.1 — Діаграма потоків даних

Розглядаючи даний програмний продукт з точки зору користувачів слід виділити наступні функції (рисунок 2.2).

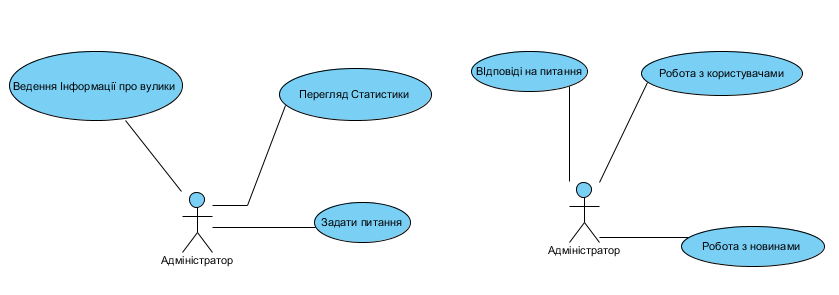


Рисунок 2.2 — Діаграма варіантів використання

Крім того веб-застосування працює за класичною клієнт-серверною технологією. У такому випадку саме веб-застосування та база даних знаходиться на CT. Доступ до застосування клієнти одержують за допомогою браузера через мережу Інтернет (рисунок 2.3).

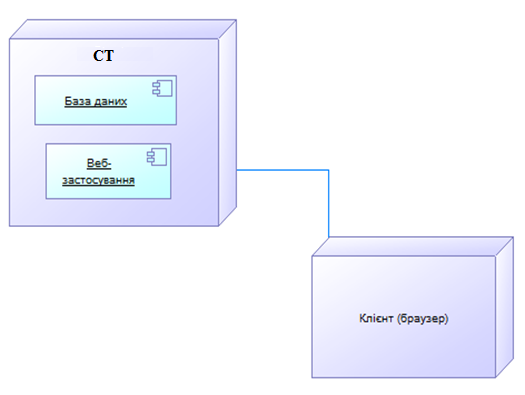
****

Рисунок 2.3 − Схема роботи системи

2.2 Розробка структури інформаційної системи

Структура розроблюваної програмної системи, подібна до структури багатьох сучасних веб-додатків, які працюють з базами даних.

Аналізуючи інформаційні потоки у системі, можна виділити дві категорії користувачів, системи — користувачі й адміністратори. Користувачі програмної системи мають доступ до усіх функціональних можливостей, системи, пов’язаних із виконанням перегляду, списків, адміністратори володіють правами на зміну пароля та інших даних користувачів, а також на зміну наповнення сайту.

Виходячи з вимог до, розроблюваного програмного комплексу структура програмного комплексу буде такою (у вигляді загального опису).

Для реалізації програмного комплексу слід обрати архітектуру клієнт-сервер. При цьому бажано використовувати тонкого клієнта, який, буде використовуватися, лише для, введення-виведення даних, а обробку даних та, збереження результатів цілком покласти на серверну частину. В якості клієнта можна використати web-браузер, організувавши обмін інформацією із сервером через HTTP-протокол. Це також, дасть можливість уникнути, необхідності в інсталяції клієнтської частини програмного комплексу на кожній робочій станції. Браузер наявний практично у кожній сучасній операційній системі.

Дані для, розрахунку та результати розрахунку можуть бути збережені, за бажанням користувача, на стороні сервера. Для цього доцільно використати, базу даних, у якій би зберігалася інформація про користувача, який є власником інформації, яку треба зберегти, та сама інформація (вхідні та вихідні дані). Структура бази даних буде уточнюватися в процесі подальшого проектування, програмного комплексу.

Інтерфейс користувача буде, генеруватися сервером автоматично у вигляді HTML-сторінки. Це забезпечить широкі можливості для маніпулювання зовнішнім виглядом інтерфейсу. З цією метою буде доцільно використовувати таблиці каскадних стилів, параметри яких будуть, зберігатися для кожного, користувача системи окремо у базі даних, конфігурації, що дасть можливість кожному користувачу створити індивідуальні настройки для інтерфейсу, змінити шрифти, кольорову гаму інтерфейсу тощо.

Програмний комплекс, повинен мати відносно простий та зручний для користувача, інсталятор, який би повідомляв користувача про те, як відбувається процес інсталяції сервера, та повідомляв які параметри конфігурації отримав сервер після інсталяції, за мовчазною згодою (default configuration).

Далі варто розглянути, деякі аспекти архітектури програмного комплексу більш детально. Зокрема, потрібно виділити особливості загальної архітектури, програмного комплексу. Загальна архітектура програмного комплексу подана на рисунку 2.4.

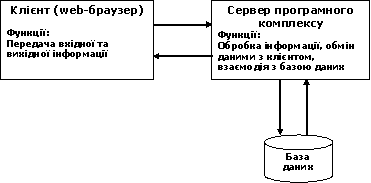


Рисунок 2.4 — Загальна архітектура програмного комплексу

Таким, чином, для реалізації програмного забезпечення інформаційної системи багатомодульна архітектура, яка забезпечує високу ефективність, процесу проектування, програмної реалізації, тестування, впровадження та, експлуатації. Система складається з таких основних блоків:

1. блок роботи з вуликами;
2. блок введення інформації – новин;
3. блок «Запитання відповіді»
4. адміністративний, інтерфейс;
5. загальнодоступний, інтерфейс для користувачів.

Адміністративний, інтерфейс доступний лише адміністратору, і повинен бути недоступним, для інших користувачів системи.

2.3 Вибір засобів розробки інформаційної системи

Розробку web-додатку планується проводити з використанням сучасних, засобів розробки програмного забезпечення для мережевих інформаційних систем. Вибрані інструментальні, засоби являють собою універсальне інструментальне програмне, забезпечення, за допомогою якого можна розробляти різні за складністю та функціональним призначенням інформаційні системи.

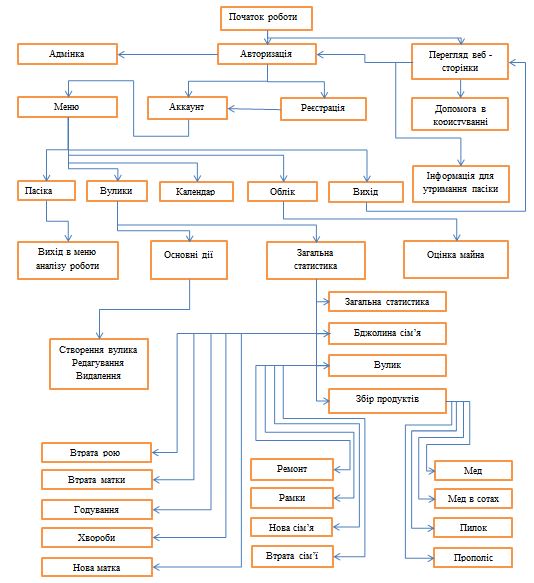


Рисунок 2.5 – Схема автоматизації процесів на бджільництві

Крім того використовуване програмне, забезпечення розповсюджується цілком безкоштовно та у вихідних текстах, що дозволяє з одного боку здійснювати гнучке ,налаштування програмного забезпечення або його модифікацію, з метою адаптації під специфічні потреби та вимоги конкретного, замовника та ,користувачів, а з іншого боку, – забезпечити вільне перенесення інструментального програмного забезпечення, на платформу практично, будь-якої ,сучасної операційної системи. Звичайно, вільно розповсюджуване, програмне ,забезпечення дозволяє ,також уникнути додаткових витрат на придбання, ліцензій на його використання та розповсюдження, а також на ліцензування розроблених на його базі програмних, рішень.

2.3.1 Вибір засобів розробки підсистеми розробки тестових завдань

Щоб система відповідала сучасним вимогам швидкодії та інформаційної надійності, вона розробляється з використанням новітніх ІТ-технологій ASP.NET та ADO.NET (технологія доступу до даних). ASP.NET це складова частина платформи Microsoft.NET і є розвитком більш старої технології Microsoft ASP. Ця технологія забезпечує програмну модель та інфраструктуру для створення масштабованих, безпечних та стійких web-програм. З її допомогою розробка стала значно легшою ніж з попередніми технологіями для програмування в мережі Internet. ASP.NET має перевагу у швидкості в порівнянні з іншими технологіями, заснованими на скриптах, тому що код на стороні web-сервера звичайно компілюється в одну або декілька DLL. А використання на сторінках сайту елементів технології AJAX (технологія асинхронних запитів на сервер) дозволяє взаємодіяти з серверною частиною додатку практично в режимі реального часу, без перезавантаження всієї web-сторінки.

2.3.2 Вибір СКБД для підсистеми розробки тестових завдань

В якості СУБД було обрано СТ, що зумовлено перевагами адміністрування, налаштування, надійності, зручності та найбільшої сумісності з технологією ADO.NET.

**Хмарні обчислення** (англ. cloud computing) — це модель забезпечення повсюдного та зручного мережевого доступу на вимогу до загального пулу конфігуруємих обчислювальних ресурсів (наприклад, мереж передачі даних, серверів, пристроїв зберігання даних, додатків і сервісів - як разом, так і окремо) , які можуть бути оперативно надані і звільнені з мінімальними експлуатаційними витратами і / або зверненнями до провайдера.

**Основні переваги**:

* не потрібні великі обчислювальні потужності ПК - по суті будь-який смартфон, планшет і т.д., при відкритті вікна браузера отримує величезний потенціал.
* відмовостійкість;
* певний рівень безпеки;
* висока швидкість обробки даних;
* економія на покупці софта - всі необхідні програми вже є в сервісі, де будуть працювати додатки;
* Ваш власний вінчестер не наповнюється - всі дані зберігаються в мережі..

**Є й ряд недоліків**:

* хмарна послуга надається завжди якоюсь компанією, відповідно, збереження даних користувача залежить від цієї компанії;
* поява хмарних монополістів;
* необхідність завжди бути в мережі для роботи;
* небезпека хакерських атак на сервер (при зберіганні даних на комп'ютері ви в будь-який час можете відключитися від мережі і очистити систему за допомогою антивірусу);
* можлива подальша монетизація ресурсу - цілком можливо, що компанії надалі вирішить брати за послуги плату з користувачів.

З точки зору постачальника, завдяки об'єднанню ресурсів і непостійному характеру споживання з боку споживачів, хмарні обчислення дозволяють економити на масштабах, використовуючи менші апаратні ресурси, ніж потрібні були б при виділених апаратних потужностях для кожного споживача, а за рахунок автоматизації процедур модифікації виділення ресурсів істотно знижуються витрати на абонентське обслуговування.

З точки зору споживача, ці характеристики дозволяють отримати послуги з високим рівнем доступності (англ. High availability) і низькими ризиками непрацездатності, забезпечити швидке масштабування обчислювальної системи завдяки еластичності без необхідності створення, обслуговування і модернізації власної апаратної інфраструктури.

Зручність і універсальність доступу забезпечується широкою доступністю послуг і підтримкою різного класу термінальних пристроїв (персональних комп'ютерів, мобільних телефонів, інтернет-планшетів).

Обов'язкові характеристики хмарних обчислень:

* Самообслуговування по вимозі (self service on demand), споживач самостійно визначає і змінює обчислювальні потреби, такі як серверний час, швидкість доступу та обробки даних, обсяг збережених даних без взаємодії з представником постачальника послуг;
* Універсальний доступ по мережі, послуги доступні споживачам по мережі передачі даних незалежно від використовуваного термінального пристрою;
* Об'єднання ресурсів (resource pooling), постачальник послуг об'єднує ресурси для обслуговування великої кількості споживачів в єдиний пул для динамічного перерозподілу потужностей між споживачами в умовах постійної зміни попиту на потужності; при цьому споживачі контролюють тільки основні параметри послуги (наприклад, обсяг даних, швидкість доступу), але фактичний розподіл ресурсів, що надаються споживачеві, здійснює постачальник (в деяких випадках споживачі все-таки можуть управляти деякими фізичними параметрами перерозподілу, наприклад, вказувати бажаний центр обробки даних з міркувань географічної близькості);
* Еластичність, послуги можуть бути надані, розширені, звужені в будь-який момент часу, без додаткових витрат на взаємодію з постачальником, як правило, в автоматичному режимі;
* Облік споживання, постачальник послуг автоматично обчислює спожиті ресурси на певному рівні абстракції (наприклад, обсяг збережених даних, пропускна спроможність, кількість користувачів, кількість транзакцій), і на основі цих даних оцінює обсяг наданих споживачам послуг.

**Приватна хмара**(private cloud) —інфраструктура, призначена для використання однією організацією, що включає кілька споживачів (наприклад, підрозділів однієї організації), можливо також клієнтами і підрядниками даної організації. Приватна хмара може перебувати у власності, управлінні та експлуатації як самої організації, так і третьої сторони (або будь-якої їх комбінації), і воно може фізично існувати як всередині, так і поза юрисдикцією власника.

**Публічна хмара**(public cloud) — інфраструктура, призначена для вільного використання широкою публікою. Публічна хмара може перебувати у власності, управлінні та експлуатації комерційних, наукових та урядових організацій (або будь-якої їх комбінації). Публічна хмара фізично існує в юрисдикції власника — постачальника послуг.

**Гібридна хмара** (hybrid cloud) — це комбінація з двох або більше різних хмарних інфраструктур (приватних, публічних або суспільних), що залишаються унікальними об'єктами, але пов'язаних між собою стандартизованими або приватними технологіями передачі даних і додатків (наприклад, короткочасне використання ресурсів публічних хмар для балансування навантаження між хмарами).

**Суспільна хмара**(англ. community cloud) — вид інфраструктури, призначений для використання конкретною спільнотою споживачів з організацій, що мають спільні завдання (наприклад, місії, вимоги безпеки, політики, та відповідності різним вимогам). Громадська хмара може перебувати в кооперативній (спільній) власності, управлінні та експлуатації однієї або більше з організацій співтовариства або третьої сторони (або будь-якої їх комбінації), і вона може фізично існувати як всередині, так і поза юрисдикцією власника.

Ефективний, простий і зручний доступ до централізованої бази даних програмного забезпечення для розрахунку припусків на механічну обробку можуть забезпечити сучасні веб-технології, оскільки переважна більшість сучасних мов програмування для веб має ефективні засоби для взаємодії із сучасними серверами баз даних та зручного представлення одержаної з бази даних інформації для кінцевого користувача. Крім того, при використанні сучасних мов веб-програмування для створення інформаційних систем, зі сторони користувача відпадає необхідність у встановленні та експлуатації спеціалізованого клієнтського програмного забезпечення для роботи з базою даних розроблюваної інформаційної системи.

Всі операції з даними відбуваються з використанням процедур та функцій, що зберігаються. Це значно підвищує швидкість роботи додатку з базою даних, а також безпеку додатку, оскільки неможливо передати підзапит у параметрах (протидія SQL-Injection). До того ж вони дозволяють відокремити логіку роботи з даними у базі даних від логіки додатку, що покращує ефективність розробки додатку та подальшої його модифікації.

jQuery − бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елементу DOM, звертатися до атрибутів і вмісту елементів DOM, маніпулювати ними. Також бібліотека jQuery надає зручний API для роботи з AJAX.

Головні можливості jQuery:

* движок кросбраузерності CSS-селекторів Sizzle, що виділився в окремий проект;
* перехід по дереву DOM, включаючи підтримку XPath як плагіна;
* події;
* візуальні ефекти;
* AJAX-додатки;
* JavaScript-плагіни.

2.3.3 Принципова схема дисперійного аналізу при обробці інформаційних данних у роботі бджільництва

Часто дослідники порівнюють середні кількох вибірок, об'єднаних у єдиний комплекс. Наприклад, вивчаючи вплив різних видів і доз добрив на урожайність сільськогосподарських культур досліди повторюють у різних варіантах. У цих випадках попарні порівняння стають громіздкими, а статистичний аналіз усього комплексу потребує застосування особливого методу. Такий метод, розроблений в математичній статистиці, дістав назву дисперсійного аналізу. Вперше його застосував англійський статистик Р. Фішер при обробці результатів агрономічних дослідів (1938 р.).

**Дисперсійний аналіз**- це метод статистичної оцінки надійності проявлення залежності результативної ознаки від одного або кількох факторів. За допомогою методу дисперсійного аналізу проводиться перевірка статистичних гіпотез відносно середніх в кількох генеральних сукупностях, які мають нормальний розподіл.

Дисперсійний аналіз є одним з основних методів статистичної оцінки результатів експерименту. Все більш широке застосування отримує він і в аналізі економічної інформації. Дисперсійний аналіз дає змогу встановити, наскільки вибіркові показники зв'язку результативного і факторних ознак достатні для поширення одержаних за вибіркою даних на генеральну сукупність. Достоїнством цього методу є те, що він дає досить надійні висновки по вибірках невеликої чисельності.

Досліджуючи варіацію результативної ознаки під впливом одного або кількох факторів за допомогою дисперсійного аналізу можна одержати крім загальних оцінок істотності залежностей, також і оцінку відмінностей у величині середніх, що формуються при різних рівнях факторів, та істотності взаємодії факторів. Дисперсійний аналіз застосовується для вивчення залежностей як кількісних, так і якісних ознак, а також при їх поєднанні.

Суть цього методу полягає в статистичному вивченні вірогідності впливу одного або кількох факторів, а також їх взаємодії на результативну ознаку. Відповідно до цього за допомогою дисперсійного аналізу вирішуються три основних завдання: 1) загальна оцінка істотності відмінностей між груповими середніми; 2) оцінка вірогідності взаємодії факторів; 3) оцінка істотності відмінностей між парами середніх. Найчастіше такі завдання доводиться вирішувати дослідникам при проведенні польових і зоотехнічних дослідів, коли вивчається вплив кількох факторів на результативну ознаку одночасно.

Принципова схема дисперсійного аналізу включає встановлення основних джерел варіювання результативної ознаки і визначення обсягів варіації (сум квадратів відхилень) за джерелами її утворення; визначення числа ступенів свободи, що відповідають компонентам загальної варіації; обчислення дисперсій як відношення відповідних обсягів варіації до їх числа ступенів свободи; аналіз співвідношень між дисперсіями; оцінка вірогідності різниці між середніми і формулювання висновків.

Зазначена схема зберігається як при простих моделях дисперсійного аналізу, коли дані групуються за однією ознакою, так і при складних моделях, коли дані групуються за двома і більшим числом ознак. Однак із збільшенням числа групувальних ознак ускладнюється процес розкладання загальної варіації за джерелами її утворення.

Відповідно до принципової схеми дисперсійний аналіз можна подати у вигляді п'яти послідовно виконуваних етапів:

1) визначення і розкладання варіації;

2) визначення числа ступенів свободи варіації;

3) обчислення дисперсій та їх співвідношень;

4) аналіз дисперсій та їх співвідношень;

5) оцінка вірогідності різниці між середніми і формулювання висновків з перевірки нульової гіпотези.

Найбільш трудомісткою частиною дисперсійного аналізу є перший етап - визначення і розкладання варіації за джерелами її утворення. Порядок розкладання загального обсягу варіації докладно розглядався в розділі 5.

В основі розв'язування задач дисперсійного аналізу лежить закон розкладання (додавання) варіації, відповідно до якого загальна варіація (коливання) результативної ознаки поділяється на дві: варіацію, зумовлену дією досліджуваного фактора (факторів), і варіацію, викликану дією випадкових причин, тобто

http://pidruchniki.com/imag/stat/marm_tst/image265.jpg

Щоб оцінити вірогідність відмінностей між груповими середніми, необхідно визначити міжгрупову та внутрішньогрупову варіації. Якщо міжгрупова (факторна) варіація значно перевищує внутрішньогрупову (залишкову) варіацію, то фактор впливав на результативну ознаку, істотно змінюючи значення групових середніх величин. Але виникає питання, яке співвідношення між міжгруповою і внутрішньогруповою варіаціями можна розглядати як достатнє для висновку про вірогідність (істотність) відмінностей між груповими середніми.

Для оцінки істотності відмінностей між середніми і формулювання висновків з перевірки нульової гіпотези (Н0 :х1 = х2 =... = хп) в дисперсійному аналізі використовується своєрідний норматив - Г-критерій, закон розподілу якого встановив Р.Фішер. Цей критерій являє собою відношення двох дисперсій: факторної, породжуваної дією досліджуваного фактора, та залишкової, зумовленої дією випадкових причин:

http://pidruchniki.com/imag/stat/marm_tst/image266.jpg

Дисперсійне відношення Г= £>і *:* £\*2 американським статистиком Снедекором запропоновано позначати літерою Г на честь винахідника дисперсійного аналізу Р.Фішера.

Дисперсії °2 іо2 є оцінками дисперсії генеральної сукупності. Якщо вибірки з дисперсіями °2 і °2 зроблені з однієї і тієї самої генеральної сукупності, де варіація величин мала випадковий характер, то розбіжність у величинах °2 і °2 також випадкова.

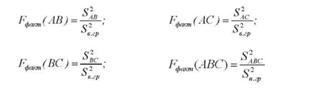
Якщо в експерименті перевіряють вплив кількох факторів (А, В, С і т.д.) на результативну ознаку одночасно, то дисперсія, що обумовлена дією кожного з них, має бути порівняна з *°е.гР* , тобто

http://pidruchniki.com/imag/stat/marm_tst/image267.jpg

Якщо значення факторної дисперсії значно більше залишкової, то фактор істотно впливав на результативну ознаку і навпаки.

В багатофакторних експериментах крім варіації, зумовленої дією кожного фактора, практично завжди є варіація, яка зумовлена взаємодією факторів ($ав: ^лс ^вс $лііс). Суть взаємодії полягає в тому, що ефект одного фактора істотно змінюється на різних рівнях другого (наприклад, ефективність якості Грунту при різних дозах добрив).

Взаємодія факторів також має бути оцінена шляхом порівняння відповідних дисперсій 3 ^в.гр :



При обчисленні фактичного значення Б-критерію в чисельнику береться більша з дисперсій, тому Б > 1. Очевидно, що чим більше критерій Б, тим значнішими є розбіжності між дисперсіями. Якщо Б = 1, то питання про оцінку істотності відмінностей дисперсій знімається.

Для визначення меж випадкових коливань відношення дисперсій Р. Фішер розробив спеціальні таблиці Б-розподілу (дод. 4 і 5). Критерій Б функціонально зв'язаний з імовірністю і залежить від числа ступенів свободи варіації *к1* і к2 двох порівнюваних дисперсій. Звичайно використовуються дві таблиці, що дозволяють робити висновки про гранично високе значення критерію Б для рівнів значущості 0,05 і 0,01. Рівень значущості 0,05 (або 5%) означає, що тільки в 5 випадках із 100 критерій Б може приймати значення, що дорівнює вказаному в таблиці або вище його. Зниження рівня значущості з 0,05 до 0,01 призводить до збільшення значення критерію Б між двома дисперсіями в силу дії тільки випадкових причин.

Значення критерію Б також залежить безпосередньо від числа ступенів свободи двох порівнюваних дисперсій. Якщо число ступенів свободи прямує до нескінченості (к-мо), то відношення Б для двох дисперсій прямує до одиниці.

Табличне значення критерію Б показує можливу випадкову величину відношення двох дисперсій при заданому рівні значущості і відповідному числі ступенів свободи для кожної з порівнюваних дисперсій. В зазначених таблицях наводиться величина Б для вибірок, зроблених з однієї і тієї самої генеральної сукупності, де причини зміни величин тільки випадкові.

Значення Г знаходять за таблицями (дод. 4 і 5) на перетині відповідного стовпця (число ступенів свободи для більшої дисперсії - к1) і рядка (число ступенів свободи для меншої дисперсії - к2). Так, якщо більшій дисперсії (чисельник Г) к1 = 4, а меншій (знаменник Г) к2 = 9, то Га при рівні значущості а = 0,05 становитиме 3,63 (дод. 4). Отже, в результаті дії випадкових причин, оскільки вибірки малочисельні, дисперсія однієї вибірки може при 5%-ному рівні значущості перевищувати дисперсію другої вибірки в 3,63 раза. При зниженні рівня значущості з 0,05 до 0,01 табличне значення критерію Г, як зазначалося вище, буде збільшуватися. Так, при тих самих ступенях свободи к1 = 4 і к2 = 9 і а = 0,01 табличне значення критерію Г становитиме 6,99 (дод. 5).

Розглянемо порядок визначення числа ступенів свободи в дисперсійному аналізі. Число ступенів свободи, що відповідає загальній сумі квадратів відхилень, розкладається на відповідні компоненти аналогічно розкладанню сум квадратів відхилень (^заг = №^гр + ]¥вгр) , тобто загальне число ступенів свободи (к") розкладається на число ступенів свободи для міжгрупової (к1) і внутрішньогрупової (к2) варіацій.

Так, якщо вибіркова сукупність, що складається з *N* спостережень, поділена на ***т***груп (число варіантів досліду) і ***п***підгруп (кількість повторностей), то число ступенів свободи к відповідно становитиме:

а) для загальної суми квадратів відхилень (й7заг)

http://pidruchniki.com/imag/stat/marm_tst/image269.jpg

б) для міжгрупової суми квадратів відхилень *^м.гР)*

http://pidruchniki.com/imag/stat/marm_tst/image270.jpg

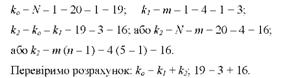
в) для внутрішньогрупової суми квадратів відхилень *уУ*в.гР)

http://pidruchniki.com/imag/stat/marm_tst/image271.jpg

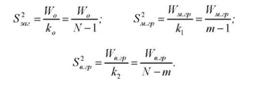
Згідно з правилом додавання варіації:

http://pidruchniki.com/imag/stat/marm_tst/image272.jpg

Наприклад, якщо в досліді сформовано чотири варіанти досліду (т = 4) у п'яти повторностях кожен (п = 5), і загальна кількість спостережень N = = *т* o п = 4 o 5 = 20, то число ступенів свободи відповідно дорівнює:



Знаючи суми квадратів відхилень і число ступенів свободи, можна визначити незміщені (скориговані) оцінки для трьох дисперсій:



Нульову гіпотезу Н0 за критерієм Б перевіряють так само, як і за і-критерієм Стьюдента. Щоб прийняти рішення з перевірки Н0, необхідно розрахувати фактичне значення критерію і порівняти його з табличним значенням Ба для прийнятого рівня значущості а і відповідного числа ступенів свободи *к1* і к2 для двох дисперсій.

Якщо Бфакг > Ба, то згідно з прийнятим рівнем значущості можна зробити висновок, що відмінності вибіркових дисперсій визначаються не лише випадковими факторами; вони істотні. Нульову гіпотезу в цьому випадку відхиляють і є підстава стверджувати, що фактор істотно впливає на результативну ознаку. Якщо ж < Ба, то нульову гіпотезу приймають і є підстава стверджувати, що відмінності між порівнюваними дисперсіями знаходяться в межах можливих випадкових коливань: дія фактора на результативну ознаку не є істотною.

Застосування тієї або іншої моделі дисперсійного аналізу залежить як від кількості досліджуваних факторів, так і від способу формування вибірок.

Залежно від кількості факторів, що визначають варіацію результативної ознаки, вибірки можуть бути сформовані за одним, двома і більшим числом факторів. Відповідно до цього дисперсійний аналіз поділяється на однофакторний і багатофакторний. Інакше його ще називають однофакторним і багатофакторним дисперсійним комплексом.

Схема розкладання загальної варіації залежить від формування груп. Воно може бути випадковим (спостереження однієї групи не пов'язані із спостереженнями другої групи) і невипадковим (спостереження двох вибірок пов'язані між собою спільністю умов експерименту). Відповідно дістають незалежні і залежні вибірки. Незалежні вибірки можуть бути сформовані як з рівною так і з нерівною чисельністю. Формування залежних вибірок передбачає їх рівну чисельність.

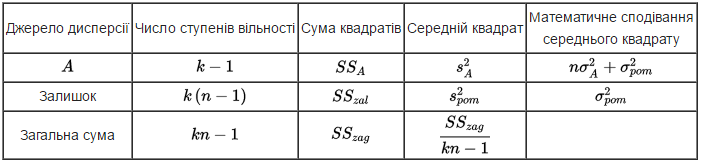
Якщо групи сформовані у невипадковому порядку, то загальний обсяг варіації результативної ознаки включає в себе поряд з факторною (міжгруповою) і залишковою варіацією варіацію повторностей, тобто

http://pidruchniki.com/imag/stat/marm_tst/image275.jpg

На практиці в більшості випадків доводиться розглядати залежні вибірки, коли умови для груп і підгруп вирівнюються. Так, у польовому досліді всю ділянку розбивають на блоки, з максимально вирівняннями умовами. При цьому кожен варіант досліду отримує рівні можливості бути представленим в усіх блоках, чим досягається вирівнювання умов для всіх перевірюваних варіантів, досліду. Такий метод побудови досліду дістав назву методу рендомізованих блоків. Аналогічно проводяться і досліди з тваринами.

При обробці методом дисперсійного аналізу соціально-економічних даних необхідно мати на увазі, що в силу багаточисельності факторів та їх взаємозв'язку важко навіть при самому ретельному вирівнюванні умов встановити ступінь об'єктивного впливу кожного окремого фактора на результативну ознаку. Тому рівень залишкової варіації зумовлюється не тільки випадковими причинами, але й суттєвими факторами, які не були враховані при побудові моделі дисперсійного аналізу. Внаслідок цього залишкова, дисперсія як база порівняння інколи стає неадекватною своєму призначенню, вона явно завищується за величиною і не може виступати як критерій істотності впливу факторів. В зв'язку з цим при побудові моделей дисперсійного аналізу стає актуальною проблема відбору найважливіших факторів і вирівнювання умов для проявлення дії кожного з них. Крім того. застосування дисперсійного аналізу передбачає нормальний або близький до нормального розподіл досліджуваних статистичних сукупностей. Якщо ця умова не витримується, то оцінки, одержані в дисперсійному аналізі, виявляться перебільшеними.

Таблиця – 1 Вихідні дані для однофакторного дисперсійного аналізу з рівним числом паралельних дослідів



Як видно з графіків, гібридні методи працюють краще за інших, тому що вони беруть кращі якості використовуваних методів, при цьому залишаючи або закриваючи їх недоліки. В даному випадку, недоліком CF-алгоритмів є відсутність аналізу зміни поведінки користувача з часом. SR-алгоритм, який сам по собі дає погані результати, покращує підсумковий результат гібридного методу в порівнянні з початковими методами.

2.3.4 Аналіз статистики роботи та візуалізація даних

За допомогою сегментів конверсії можна виокремлювати й аналізувати окремі набори шляхів до конверсії у звітах про багатоканальні послідовності. Наприклад, можна створити сегмент конверсії, який містить лише ті послідовності, у яких першою взаємодією була конверсія з певною цінністю. Переглядаючи звіти про багатоканальні послідовності, ви зможете аналізувати дані лише для цього сегмента конверсії. Можна також порівнювати дані в цьому сегменті з інформацією в інших сегментах.

Значення відносних показників для аналізу досить велике, адже за їх допомогою порівнюють характеристики окремих одиниць груп і сукупностей у цілому, вивчають структуру явищ та закономірності їх розвитку, аналізують виконання плану, вимірюють темпи розвитку і інтенсивність поширення суспільних явищ. За формою відносний показник являє собою дріб, чисельником якого є величина, котру порівнюють (в окремих випадках її називають поточною, або звітною), а знаменником - величина, з якою здійснюють порівняння. Знаменник відносної величини вважається базою порівняння. Так, питому вагу висококваліфікованих працівників підприємства розраховують діленням кількості осіб з високим рівнем кваліфікації на загальну чисельність працюючих. Базою порівняння у наведеному прикладі є загальна чисельність працюючих.

Якщо базову величину показника приймають за одиницю, формою її зображення буде коефіцієнт (кратне відношення), якщо за 100 - формою зображення відносних показників будуть проценти.

Коефіцієнт як форма виразу відносної величини показує, у скільки разів порівнювальна величина більша базової (чи яку частину від неї становить, якщо величина коефіцієнта менша за одиницю).

У статистичній практиці коефіцієнти, як правило, використовують для вираження відносних величин у випадках, коли порівнювальна величина перевищує базову більш як у 2-3 рази. Якщо таке співвідношення має менші розміри - застосовують процентні числа. У випадках, коли базову величину приймають за 1000, відносні показники виражають у проміле (%0). Наприклад, якщо питома вага осіб сільського населення району з вищою освітою становить 16 %0, це означає, що на кожну 1000 сільського населення у середньому припадає 16 чоловік з вищою освітою.

В окремих випадках відносні показники розраховують на 10000 (продециміле), 100000, 1000000 одиниць (наприклад, у статистиці охорони здоров'я розраховують кількість ліжко-місць на 10000 населення). Відносні величини, виражені на 1000, 10000, 100000 і т. д. одиниць, вживають з метою надання їм більш придатного для сприйняття вигляду, оскільки, підібравши вдало базу порівняння, можна запобігти дробовим числам.

Елементи управління ASP.NET згруповані в панелі інструментів на основі широких категорій, таких як звітність, дані, дії і декількох інших призначених для користувача категорій, що дозволяє швидко знайти будь-який з приблизно дев'яносто різних елементів управління. Використовуючи додаток ASPxGridView, як і інші з категорії даних, в тому числі WebChartControl і ASPxGaugeControl, які роблять забезпечуючи графічне відображення даних легко. Поза категорії даних, ви можете захотіти поглянути на RichEditControl (RTF редактор) і вдосконаленої бібліотеки управління печаткою, Xtra.Printing.

Таблиця 2 – Методи аналізу проваджені у роботі у інформаційній роботі

Бджільництва

|  |  |
| --- | --- |
| Назва методу аналізу | Короткий зміст і основні цілі |
| Аналіз математичних даних збору по кількості продукції по сортах | Порівняння динаміки збору різних груп товарів: різного типу; різного по часовому періоду збору; різних цінових категорій. Визначення методів роботи з різними продуктами в залежності від значимості і ролі групи в асортименті магазину. |
| Аналіз по критеріях якості роботи єдиного вулика | Оцінка тенденцій розвитку магазину. Оцінка ефективності проведених заходів щодо стимулювання збуту, по оптимізації системи мерчандайзингу тощо. |
| Аналіз загальної якості роботи вуликів | Оцінка ефективності роботи вулика на загальному відображенні усієї пасіки. Знаходження найменш прибуткового і витратного вулика. |
| Аналіз беззбитковості підприємства | Дозволяє визначити той мінімально необхідний обсяг реалізації продукції, при якому підприємство покриває свої видатки і працює беззбитково, не даючи прибуток, але і не терпить збитків. |

ASP.NET Chart Control  надає широкий спектр різних 2D типів діаграм(званих також [типів уявлень](https://documentation.devexpress.com/AspNet/CustomDocument15846.aspx) ), які дають вам можливість відображати дані, як будь-який з основних типів діаграм, або навіть як поєднання різних поглядів серії. Крім того, якщо дані, надані для конкретного типу діаграми залишається в силі для іншого типу діаграми, ви можете змінити поточний вигляд серії з одним натисненням кнопки або рядка коду. Це робить управління Chart більш гнучким і ефективним при поданні даних.

**Висновки по розділу 2**

В роботі проведено дослідження особливостей мови ASP.Net, та найбільш важливих їх нововведень:

* проаналізована абсолютно нова система СТ, що дає змогу нових можливостям, яка забезпечує повні захищеність;
* описаний елемент SCADA;
* висвітлено процеси роботи кластерного алгоритмів (методу K середніх).

У результаті роботи було досліджено методи та алгоритми, які можуть бути використані при розробці програмного продукту, елементів інформаційної технології. Була спроектована програма, аналітичного відображення показників. Вона забезпечувала прозорість та об’єктивність процесу тестування. Для реалізації системи було обрано сучасні мови програмування та СКБД, що дозволять швидко та якісно реалізувати поставлену задачу.

3 Програмна реалізація

3.1 Структура і функціональне призначення модулів системи, їх взаємозв’язок

Відповідно до розробленої структури, описаної в розділі 2.2, система ерозділяється на наступні модулі:

1. модуль реєстрації користувачів;
2. модуль для авторизованого доступу до власної інформації;
3. модуль для перегляду записів;
4. статистичний модуль для перегляду роботи пасіки і стану вуликів;
5. модуль новин;
6. модуль зворотнього зв’язку;
7. модуль «Питання/Відповіді»;
8. модуль для роботи з електронними листами.

Модуль реєстрації нового користувача – це розділ, який відповідає за реєстрацію нового користувача в базі даних, перевіряє правильність набору введеної інформації та заносить дану інформацію до БД. Новий користувач після проходження реєстрації отримує можливість змінювати дані власної анкети. Тільки адміністратори мають право на зміну іншої інформації у БД.

Модуль для авторизованого доступу до власної інформації – це розділ що відповідає за перегляд та редагування даних авторизованого користувача. В даному модулі користувач має можливість переглянути і записати дані власної пасіки. Також користувач має можливість аналізовувати дані як і старі так і нові.

Модуль для перегляду записів – це розділ, який відповідає відображення усіх подіій на бджільництві. Події відображаються за допомогою календаря на сторінці. Всі події сортуються по даті по порядку спадання.

Статистичний модуль для перегляду роботи пасіки і стану вуликів – розроблено для зручного перегляду поточного стану вуликів. Функціональність даного модуля подяляється на три вида – перегляд, і аналізування. Перегляд заключається в оцінювані стану вулика і редагування його стану. Для ефективності роботи проводиться аналіз роботи пасіки.

Модуль новин забезпечує можливість виводити новини веб-застосування для користувачів та редагувати створювати та видаляти новини адміністраторами веб-застосування.

Модуль зворотнього звязку забезпечує можливість звязку користувачів з адміністрацією веб-застосування.

Модуль «Питання/Відповіді» забезпечує тематичне спілкування між користувачами системи та адміністраторами вкб-застосування.

Модуль для роботи з електронними листами забезпечує можливість відпраки електронних листів про нові питання та нові повідомлення через форму зворотнього звязку.

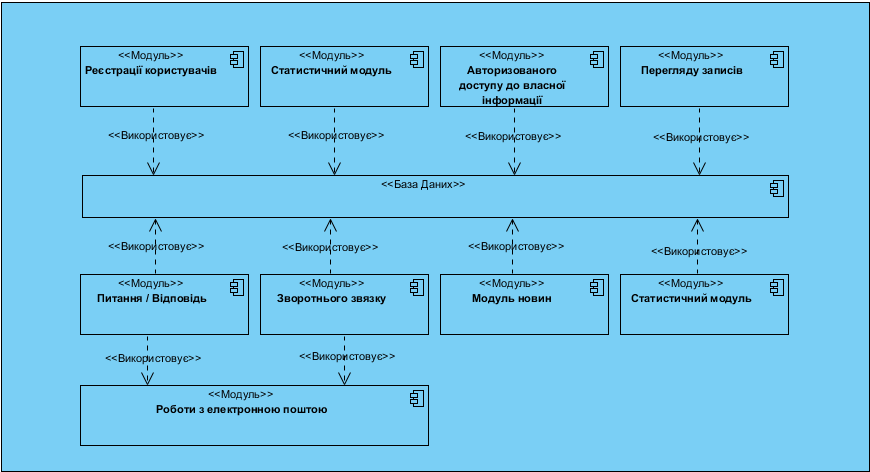


Рисунок 3.1 – Схема використання модулів

3.2 Розробка програмних модулів

Модуль реєстрації використовується під час процесу реєстрації нових користувачів. Для захисту додатку від можливих атак використано перевірку капчі. «Капча» - повністю автоматизований публічний тест Тюрінга для розрізнення комп'ютерів і людей. Тобто це вид спеціального комп'ютерний тесту, який використовується для того, щоб визначити, ким є користувач системи: людиною або комп'ютером (рисунок 3.1).

Для даної цілі використовується компонент ASPxCapcha з бібліотеки компонентів DevExpress.

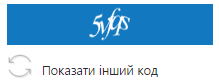


Рисунок 3.1 − Генерація капчі

protected void cbRegister\_OnCallback(object source, CallbackEventArgs e)

{

var userman = CreateManager<UsersManager>();

var email = GetValue<string>(txtEmail);

var client = userman.GetByEmail(email);

if (client != null)

throw new Exception("Нажаль користувач з таким e-mail вже зареєстрований.");

client = new Users()

{

Email = email,

FirstName = GetValue<string>(txtFirstName),

LastName = GetValue<string>(txtLastName),

RegistrationDate = DateTime.Now,

Password = GetValue<string>(txtPassword),

};

userman.Model = client;

userman.Save();

}

Модуль для авторизованого доступу до власної інформації реалізовує головний функціонал додатку. Саме цей модуль відповідає за редагування інформації про вулики таїхній стан.

#region Properties

protected override ASPxButton GetOkButton() { return btnOk; }

protected override ASPxCallback GetCallbackControl()

{

return cbSave;

}

protected override ASPxButton GetCancelButton()

{

return btnCancelEdit;

}

#endregion

protected override void DoInitialize\_2\_LoadDataInitControls()

{

base.DoInitialize\_2\_LoadDataInitControls();

InitCombo<eBeeHiveStatus>(ddlStatus, false, e0Type.Empty);

InitCombo<eBeeHiveType>(ddlType, false, e0Type.Empty);

}

protected override void InitControlsByModelData(BeeHive model)

{

if (model != null)

{

txtName.Value = model.BeeHiveName;

txtRate.Value = model.CurrentFrameAmount;

ddlStatus.Value = (int)model.BeeHiveStatus;

ddlType.Value = (int)model.BeeHiveType;

txtOnDate.Value = model.OnDate;

}

}

protected override void LoadModelDataFromControls(BeeHive model)

{

if (model.IsNew)

model.UserID = UserID;

model.BeeHiveName = GetValue<string>(txtName);

model.OnDate = GetValue<DateTime>(txtOnDate);

model.CurrentFrameAmount = GetValue<int>(txtRate);

model.BeeHiveStatus = (eBeeHiveStatus)GetValue<int>(ddlStatus);

model.BeeHiveType = (eBeeHiveType)GetValue<int>(ddlType);

}

Модуль для перегляду записів дозволяє отримати наочну інформацію про всі події що відбувалися на бджільництві у хроно логічному порядку. Для відображення цієї інформації використано компонент DevExpress ASPxSchedurel. Даний компонент дозволяє відображати події за допомогою зрозумілого користувачеві інферфейсі схожому на календар Microsoft Outlook.

<dxwschs:ASPxScheduler ID="dxShed" runat="server" GroupType="Resource" Theme="Moderno"

ActiveViewType='Month' OnDataBinding="dxShed\_OnDataBinding">

<Storage>

<Appointments>

<Mappings AppointmentId="ID" Label="ID" Description="Description"

End="CalendarEndEventDay" Start="StartEventDate" Status="ID"

Subject="EventTypeGridText"></Mappings>

</Appointments>

</Storage>

<OptionsCustomization AllowAppointmentCreate="None"

AllowAppointmentEdit="None"

AllowAppointmentDelete="None" />

<Views>

<WorkWeekView Enabled="false" />

<TimelineView Enabled="false" />

</Views>

</dxwschs:ASPxScheduler>

Статистичний модуль для перегляду роботи пасіки і стану вуликів дозволяє у наочній формі одержати інформацію про поточний стан пасіки та про збір продуктів бджільництва. Для відображення цих даних в інтуїтивно зрозумілій користувачу формі використано діаграми.

<dxchartsui:WebChartControl ID="dxHiveHoney" runat="server" BackColor="Transparent" CrosshairEnabled="True" CrosshairOptions-ValueLineStyle-DashStyle="Dash" CssClass="AlignCenter TopLargeMargin" Height="200px" Legend-BackColor="Transparent" PaletteName="CustomPalette" ToolTipEnabled="True" Width="400px" SideBySideBarDistanceFixed="6">

<legend alignmentvertical="Bottom" alignmenthorizontal="RightOutside" backcolor="Transparent" direction="TopToBottom" maxhorizontalpercentage="30">

<padding bottom="15"></padding>

<border visible="False"></border>

</legend>

<seriesserializable>

<cc1:Series Name="Series 1" ArgumentDataMember="ItemName" ValueDataMembersSerializable = "ItemValue" LegendPointOptions-PointView="Argument" LabelsVisibility="True">

<viewserializable>

<cc1:SideBySideBarSeriesView BarWidth="0.5" ColorEach="True">

</cc1:SideBySideBarSeriesView>

</viewserializable>

<labelserializable>

<cc1:SideBySideBarSeriesLabel MaxLineCount="1" ShowForZeroValues="True">

<PointOptions>

<ValueNumericOptions Precision="2" Format="Number"></ValueNumericOptions>

</PointOptions>

</cc1:SideBySideBarSeriesLabel>

</labelserializable>

<LegendPointOptionsSerializable>

<cc1:PointOptions PointView="Argument"></cc1:PointOptions>

</LegendPointOptionsSerializable>

</cc1:Series>

</seriesserializable>

<seriestemplate>

<viewserializable>

<dx:SideBySideBarSeriesView>

</dx:SideBySideBarSeriesView>

</viewserializable>

<labelserializable>

<dx:SideBySideBarSeriesLabel LineVisible="True">

<fillstyle>

<optionsserializable>

<dx:SolidFillOptions />

</optionsserializable>

</fillstyle>

<pointoptionsserializable>

<dx:PointOptions>

</dx:PointOptions>

</pointoptionsserializable>

</dx:SideBySideBarSeriesLabel>

</labelserializable>

<legendpointoptionsserializable>

<dx:PointOptions>

</dx:PointOptions>

</legendpointoptionsserializable>

</seriestemplate>

<clientsideevents objecthottracked="function(s, e) {

var hitInPie = e.hitInfo.inSeries &amp;&amp; !e.hitInfo.inLegend;

s.SetCursor(hitInPie ? 'pointer' : 'default');

}" />

<fillstyle>

<optionsserializable>

<dx:SolidFillOptions />

</optionsserializable>

</fillstyle>

<padding bottom="2" left="2" right="2" top="2" />

<borderoptions visible="False" />

<titles>

</titles>

<diagramserializable>

<dx:SimpleDiagram>

</dx:SimpleDiagram>

<cc1:XYDiagram PaneDistance="120">

<AxisX VisibleInPanesSerializable="-1" visible="False"></AxisX>

<AxisY VisibleInPanesSerializable="-1">

<scalebreakoptions sizeinpixels="2" style="Waved" />

<autoscalebreaks enabled="True" maxcount="7" />

</AxisY>

<defaultpane weight="5">

</defaultpane>

</cc1:XYDiagram>

</diagramserializable>

<tooltipoptions>

<tooltippositionserializable>

<dx:ToolTipMousePosition />

</tooltippositionserializable>

</tooltipoptions>

<palettewrappers>

<dxchartsui:PaletteWrapper Name="CustomPalette" ScaleMode="Repeat">

<palette>

<cc1:PaletteEntry Color="72, 139, 248" Color2="48, 97, 178" />

<cc1:PaletteEntry Color="10, 165, 96" Color2="4, 113, 65" />

<cc1:PaletteEntry Color="222, 73, 62" Color2="167, 51, 41" />

<cc1:PaletteEntry Color="251, 204, 68" Color2="233, 180, 26" />

<cc1:PaletteEntry Color="46, 204, 221" Color2="41, 186, 202" />

<cc1:PaletteEntry Color="255, 152, 51" Color2="231, 136, 32" />

<cc1:PaletteEntry Color="255, 53, 53" Color2="233, 26, 26" />

<cc1:PaletteEntry Color="130, 224, 81" Color2="105, 196, 52" />

<cc1:PaletteEntry Color="165, 62, 221" Color2="148, 53, 199

<cc1:PaletteEntry Color="54, 213, 143" Color2="30, 190, 120" />

<cc1:PaletteEntry Color="97, 248, 79" Color2="136, 46, 142" />

<cc1:PaletteEntry Color="72, 42, 251" Color2="55, 26, 232" />

</palette>

</dxchartsui:PaletteWrapper>

</palettewrappers>

<crosshairoptions valuelinestyle-dashstyle="Dash"></crosshairoptions>

</dxchartsui:WebChartControl>

Модуль новин використовується для показу новин на сторінці веб-додатку, а також для їх редагування, додавання та видалення адміністраторами. Для відображення новин використовується компонент DevExpress ASPxNews.

<dx:ASPxNewsControl ID="ctNews" runat="server" NavigateUrlFormatString="javascript:void('{0}');"

Width="100%" Theme="SoftOrange"

TextField="Text" DateField="OnDate" HeaderTextField="Subject"

DataMember="Text" ImageUrlField="ImagePath" NameField="UserID">

<ItemSettings ShowImageAsLink="True" MaxLength="250" TailText="Далі" ShowHeaderAsLink="True" />

<PagerSettings CurrentPageNumberFormat="{0}" EllipsisMode="OutsideNumeric" Position="Bottom" Visible="True" PageSizeItemSettings-Visible="True">

</PagerSettings>

<ItemImage Height="150px" Width="180px"></ItemImage>

<ClientSideEvents TailClick="function(s, e) { OnTailClick(e.name, e.htmlElement); }" />

</dx:ASPxNewsControl>

Під час розробки модуля зворотнього зв’язку для відображення контактних даних було використано сервіс Google Maps, що дозволяє наочно одержувати інформацію про місце знаходження необхідних обэктів. Крім того для відпралення введеної користувачем інформації адміністрації додатку використано модуль роботи з електронними листами.

protected void cbSave\_OnCallback(object source, CallbackEventArgs e)

{

SendEmailMessage(MailGunHelper.ClientAuth.EmailFrom,

string.Format("Повідомлення з форми контактів від '{0}', телефон - '{1}'", GetValue<string>(txtName),

GetValue<string>(txtPhone)), GetValue<string>(txtMessage), GetValue<string>(txtEmail));

}

Модуль «Питання/Відповіді» дозволяє задавати питання користувачам додатку. Відповіді на ці питання можуть дати тільки користувачі додатку з спеціальними правами (адміністратори). Після отримання відповіді користувач, що задавав питання одержує e-mail з відповіддю. Якщо адміністратор вважає що питання можу бути цікавим для інших користувачів системи, він може «Питання», що є доступною для перегляду усіма користувачами.

#region Properties

protected override ASPxButton GetOkButton() { return btnOk; }

protected override ASPxCallback GetCallbackControl()

{

return cbSave;

}

protected override ASPxButton GetCancelButton()

{

return btnCancelEdit;

}

#endregion

protected override void InitControlsByModelData(fForum model)

{

if (model != null)

{

txtQuestionName.Value = model.ThreadName;

txtLogin.Value = model.CreatorEmail;

txtAnswer.Value = model.ThreadAnswer;

txtName.Value = model.CreatorName;

dvAnswer.Visible = dvIsPublished.Visible = User != null && User.IsModerator;

}

else

{

dvAnswer.Visible = dvIsPublished.Visible = false;

if (User != null)

{txtLogin.Value = User.Email;

txtName.Value = User.FullName;

}

}

}

protected override void LoadModelDataFromControls(fForum model)

{

if (model.IsNew)

{

model.UserID = User != null ? (int?)User.UserID : null;

model.CreatorEmail = GetValue<string>(txtLogin);

model.CreatorName = GetValue<string>(txtName);

model.CreatedDate = DateTime.Now;

}

model.ThreadName = GetValue<string>(txtQuestionName);

if (dvAnswer.Visible)

{

model.ThreadAnswer = GetValue<string>(txtAnswer);

if (!string.IsNullOrWhiteSpace(model.ThreadAnswer))

{

model.AnswerDate = DateTime.Now;

model.IsPublished = GetValue<bool>(chbIsPublished);

}

else

model.IsPublished = false;

}

}

Модуль для роботи з електронними листами використовує потужний API-інтерфейс MailGun, який дозволяє відправляти, отримувати і відстежувати електронну пошту легко. MailGun надає можливість відсилати 10000 листів безкоштовно в місяць. Крім того існує можливість перегляду статистики відправлених листів та моніторингу їхнього стану. Це дозволяє оперативно виявляти проблеми з можливою втратою листів користувачами через портапляння в спам або невірно введену e-mail адресу. Для роботи з даним сервісом використовується RestSharp.ddl. Дана бібліотека є простим   
Rest і HTTP API клієнтом для .NET.

public class MailGunSender

{

#region -= Properties =-

public string MailGunClientBaseUrl { get { return "https://api.mailgun.net/v2"; } }

public MailGunClientAuth CurrentMailGunClientAuth { get; private set; }

#endregion

public MailGunSender(MailGunClientAuth mailGunClientAuth)

{

CurrentMailGunClientAuth = mailGunClientAuth;

}

#region -= Send Email =-

public IRestResponse SendMessage(MailMessage mail)

{

IRestResponse RestResponse = null;

if (CurrentMailGunClientAuth == null)

throw new NullReferenceException();

ExecuteMethod(delegate()

{

var client = new RestClient()

{

BaseUrl = new Uri(MailGunClientBaseUrl),

Authenticator = new HttpBasicAuthenticator("api", CurrentMailGunClientAuth.ApiKey)

};

var request = new RestRequest { Resource = "{domain}/messages" };

request.AddParameter("domain",

CurrentMailGunClientAuth.SubDomain, ParameterType.UrlSegment);

if (!string.IsNullOrEmpty(mail.From.Address))

request.AddParameter("from", mail.From);

else

request.AddParameter("from", CurrentMailGunClientAuth.EmailFrom);

request.AddParameter("subject", mail.Subject);

request.AddParameter("html", mail.Body);

foreach (var to in mail.To)

request.AddParameter("to", to.ToString());

foreach (var tocc in mail.CC)

request.AddParameter("cc", tocc.ToString());

foreach (var tobcc in mail.Bcc)

request.AddParameter("bcc", tobcc.ToString());

if (mail.Attachments.Any())

{

foreach (var attachment in mail.Attachments)

request.AddFile("attachment", attachment.ContentStream.ReadAsBytes(), attachment.Name);

}

request.Method = Method.POST;

RestResponse = client.Execute(request);

});

return RestResponse;

}

#endregion

}

3.3 Тестування інформаційної системи

Якість, програмного продукту характеризується, набором властивостей, що визначають, наскільки, продукт «хороший» з точки зору зацікавлених сторін, таких як замовник продукту, спонсор, кінцевий користувач, розробники, і тестувальники продукту, інженери підтримки, співробітники відділів [маркетингу](http://ua-referat.com/%D0%9C%D0%B0%D1%80%D0%BA%D0%B5%D1%82%D0%B8%D0%BD%D0%B3), [навчання](http://ua-referat.com/%D0%9D%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%8F) і, продажів.

З технічної точки зору тестування, полягає у виконанні програми на деякій множині, вихідних даних м звірці одержуваних результатів із, заздалегідь відомими (еталонними) з метою, [встановити](http://ua-referat.com/%D0%92%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8) відповідність різних властивостей і характеристик, програми замовленим, властивостями.

Основна ідея юніт (або модульного, як його ще називають) тестування - тестування окремих компонентів програми, тобто класів та їх методів. Розробляти код, покритий тестами, дуже корисно, тому що при їх правильному використанні практично виключається можливість регресії в історії розвитку програми - «щось нове додали, половина старого перестало працювати». Також зараз вельми модна методологія розробки "TDD" - Test Driven Development. Відповідно до неї, програміст спочатку розробляє набір тестів для майбутньої функціональності, прораховує всі варіанти виконання, і лише потім починає писати безпосередньо робочий код, що підходить під вже написані тести.

Для Visual Studio, та мови програмування С#, існують наступні найбільші популярні засоби для юніт тестування - Nunit та Unit Testing Framework.

Unit Testing Framework - це вбудована в Visual Studio система тестування, що розробляється Майкрософт, постійно розвивається, і що важливо, вона майже напевно буде існувати весь час, поки є Visual Studio. Перевагами даного методу є інтеграція з IDE і функція підрахунку відсотка покриття коду в програмі.

Таким чином для тестування роботи програми було обрано Visual Studio Unit Testing Framework.

Visual Studio Unit Testing Framework включає в себе набір засобів тестування модулів, які були інтегрувані в деяких версії Visual Studio 2005 і пізніші версії. Модульні тести, створені з рамках модульного тестування можуть бути виконані в Visual Studio або, використовуючи утиліту MSTest.exe, з командного рядка.

Приклад юніт тесту для користувача:

[TestClass]

public class TestUsers : BaseTest

{

[TestMethod]

public void AddUser()

{

var manager = GetManager<UsersManager>();

manager.Model = new Users()

{

FirstName = new Guid().ToString(),

LastName = new Guid().ToString(),

Email = new Guid().ToString(),

Password = new Guid().ToString(),

Role = eUserRole.User,

RegistrationDate = DateTime.Now,

};

manager.Save();

Assert.AreNotEqual(manager.Model.ID, 0);

}

[TestMethod]

public void AddUserFailed()

{

try

{

var manager = GetManager<UsersManager>();

manager.Model = new Users()

{

FirstName = new Guid().ToString(),

LastName = new Guid().ToString(),

Email = new Guid().ToString(),

Password = new Guid().ToString(),

};

manager.Save();

Assert.Fail();

}

catch

{

Assert.IsTrue(true);

}

}

[TestMethod]

public void GetUser()

{

var manager = GetManager<UsersManager>();

var model = manager.GetByEmail("a@a.a");

Assert.IsNotNull(model);

}

[TestMethod]

public void GetUserFailed()

{

var manager = GetManager<UsersManager>();

var model = manager.GetByEmail(new Guid().ToString());

Assert.IsNull(model);

}

}

Тестування допомагає виявити деякі помилки в програмі, нехай вони виникають від забудькуватості, ліні чи не знання, але так їх виправляти їх значно легше, коли помилки виявлені відразу після написання коду.

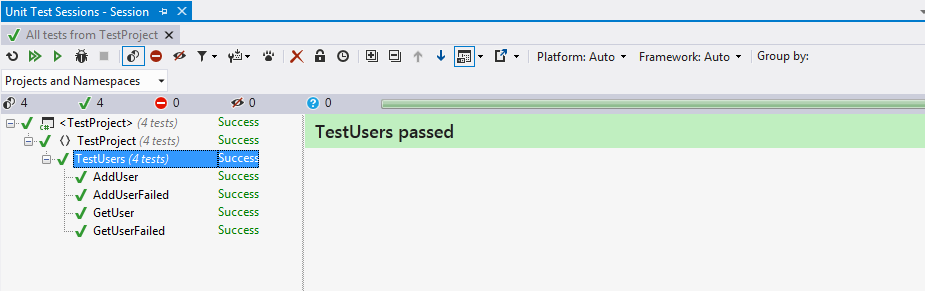


Рисунок 3.2 – Тест на виявлення помилок

3.4 Інструкція користувача

При першому відвідуванні веб-застосування з’являється стартова форма додатку. Така форма – одночасно і відправна точка, і центр дій. Типова стартова форма містить набір кнопок, напрямних в різні місця. Неавторизованому користувачеві доступні модулі «новин», «запитань», «авторизації» та «реєстрації» (рисунок 3.3).

Крім того на стартовій сторінці присутній опис основного функціоналу веб-застосування.

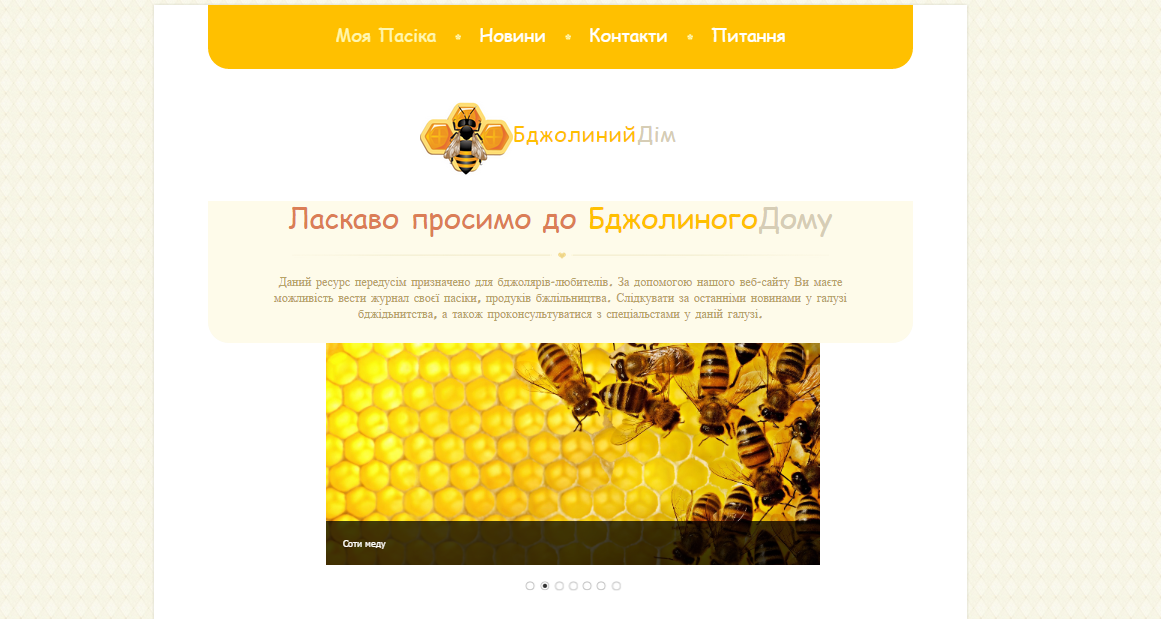


Рисунок 3.3 – Головна сторінка

Для доступу до модуля роботи з пасікою необхідна авторизація (рисунок 3.4)

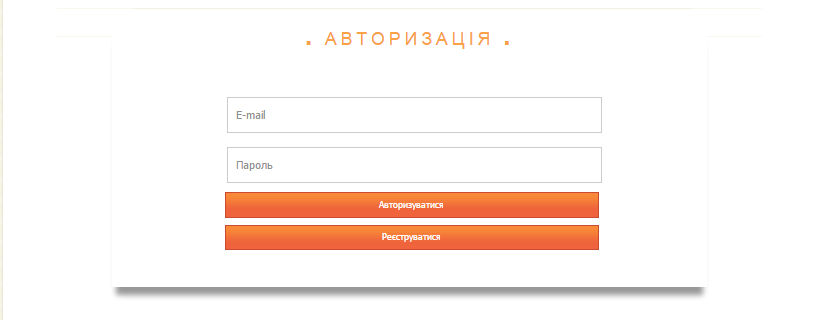


Рисунок 3.4 – Сторінка авторизації

Після авторизації завантажується форма з статистикою пасіки (рисунок 3.5).

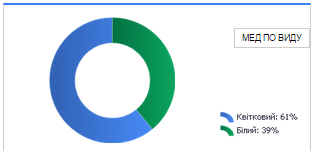


Рисунок 3.5 – Статистичний блок «Мед по виду»

Для початку роботи з вуликами необхідно обрати відповідний пункт меню та обрати існуючий або створит новий вулик (рисунок 3.6)

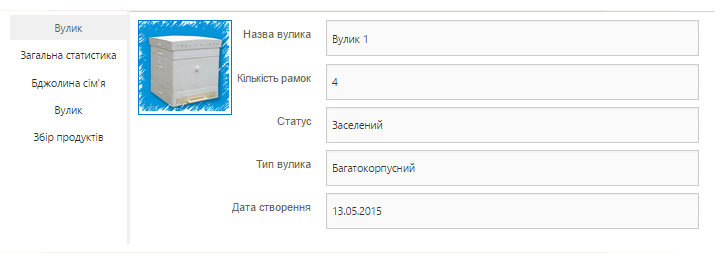


Рисунок 3.6 – Фома роботи з вуликом

Для ведення журнаду вулика необхідно обрати тип події та ввести потрібну інформацію (рисунок 3.7)

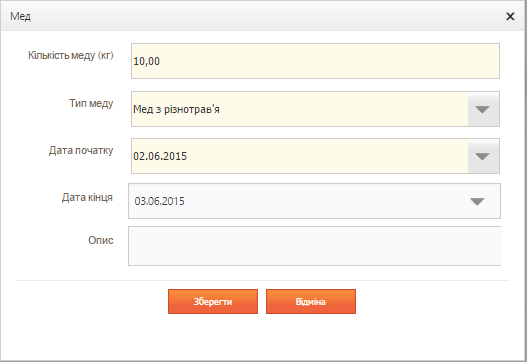


Рисунок 3.7 – Форма введення інформаці ї про збір меду

Для адміністратора веб-застосування доступні функції редагування інформації про каристувачів, новин та відповідей на поставлені користувачами питання (Рисунок 3.8)

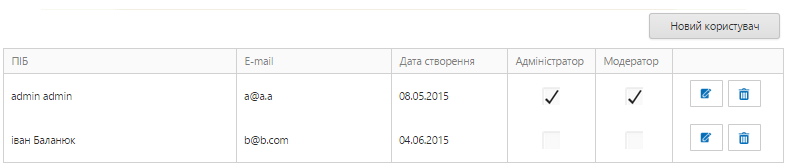


Рисунок 3.8 – Форма «Користувачі»

Перейшовши по посиланню «Новини» можна переглянути додані аднімінстарорами веб-застосування новини (рисунок 3.9)

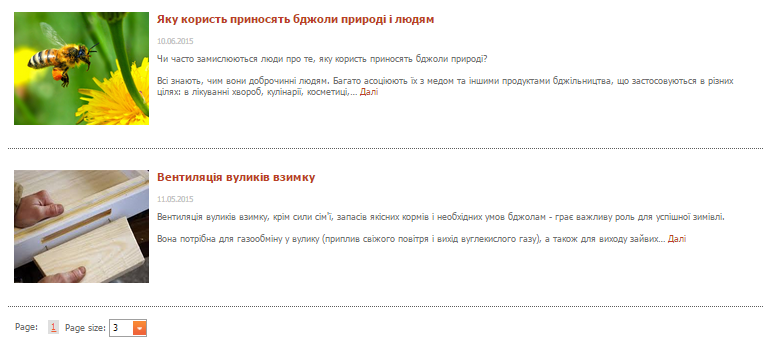


Рисунок 3.9 – Сторінка «Новини»

Для того, щоб задати питання необхідно обрати відповідний пункт меню та натиснути на кнопку «Задати питання» (рисунок 3.10).

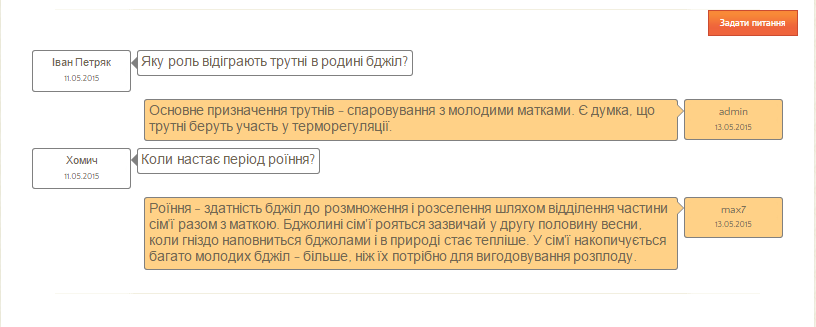


Рисунок 3.10 – Сторінка Питань

Для звязку з адміністрацією у веб-застосування реалізовано форму зворотнього звязку, що містить контактні дані. Крім того існує можливість звязатися задміністраціє залишивши повідомлення, яке буде відправлено на пошту адміністратора сайту (рисунок 3.11)

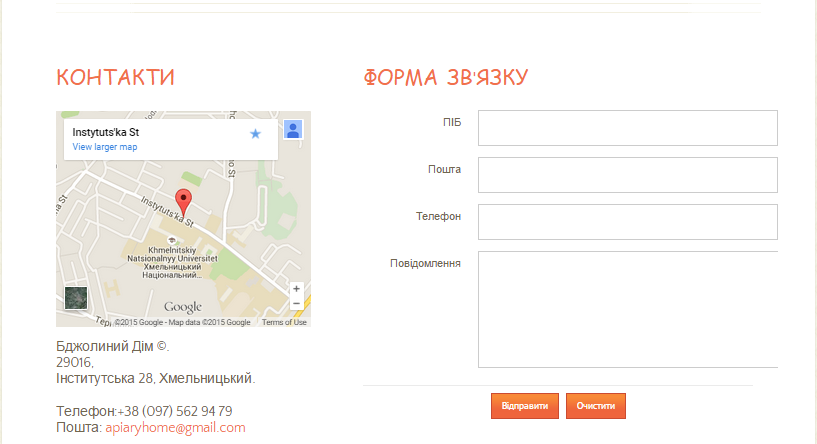


Рисунок 3.11 – Форма зворотного зв’язку

За допомогою форм робота з веб-застосування стає простим завданням. Чим кращий інтерфейс розробить розробник, тим простішою буде робота з веб-застосуванням для користувача. Потрібно також зауважити, що продумати інтерфейс та функціонал форми не є простим завданням. Враховувати потрібно й те, що форма має виконувати лише ті функції, які потрібні користувачу. Зайві функції, кнопки можуть тільки заплутати людину, що працює з програмою.

3.5 Вимоги до апаратних та програмних засобів

Для, веб-додатків, до, яких належить і розроблена у рамках дипломного проекту програмна система, вимоги, до апаратних засобів залежать переважно не від вимог розробленого програмного забезпечення, а від вимог операційної системи та, веб-сервера, на базі якого працюватиме web-додаток.

Однак рекомендованими мінімальними вимогами до апаратного забезпечення розрахункової програми можна вважати такі.

Вимоги для системи користувача:

* + - процесор Athlon 2400 або сумісний та рівно потужний з даним;
    - пам’ять комп’ютера не менше 512 МБ;
    - мережева карта 10/100 Мбіт;
    - місце на жорсткому диску не менше 100 МБ;
    - операційна система – будь-яка;
    - Internet-браузер.

Вимоги для роботи Web-сервера.

Для розміщення веб-додатку на сервері потрібно, щоб він відповідав таким технічним та програмним вимогам, а саме:

* Windows Server 2008 (2012);
* встановлений Internet Information Services;
* MS SQLServer Express 2008 R2 або MS SQLServer 2008 R2;
* процесор Pentium 4, 2,5 ГГц, або більше;
* оперативна пам'ять не менше 1024 МБ;
* вільного місця на жорсткому диску не менше 200 МБ;

Висновки по розділу 3

Провівши тестування програмного продукту та порівнявши його з існуючими рішеннями було з’ясовано, що розроблений прототип має перевагу в аналітичному відображені інформації і зручності за спостеріганням процесу користувачам над більшістю існуючих рішень. Оптимізований інструмент адміністрування значно спрощує роботу адміністратора і відповідає всім його вимогам. Розроблена система тестування є більш гнучкою за існуючі аналоги, оскільки обирає не тільки вказані чинники, а аналізуючи не достаючи значення що не завдає складність користувачеві на виході отриманих даних.

Загальні висновки

Під час роботи над дипломним проектом було проаналізовано сучасний стан інформатизації заданої предметної області – пасічного господарства та сучасних платформ розробки програмного забезпечення. Було поставлено задачу розробки системи автоматизації пасічного господарства. Для цього спроектовано структуру системи у вигляді сайту з базою даних, та обрано засоби розробки:

* C#;
* ASP.NET;
* .NET Framework 4,5;
* MS SQL Server Express 2008 R2;
* jQuery.

У результаті виконання дипломного проекту було розроблено веб-застосування, що повністю відповідає поставленому завданню, а саме:

* дозволяє вести записи та статистичні дані стосовно своєї пасіки;
* реалізовує зручний для користувача інтерфейс;
* забезпечує можливість комунікації між адміністрацією додатку та користувачами;
* забезпечує перегляд статистики своєї пасіки.

Програмний продукт протестовано засобами MS Visual Studio.

Сферою застосування даного програмного продукту визначено використання програмного продукту у якості інструменту для інформаційної та аналітичної підтримки пасічників. Розроблений веб-додаток є зручним у використанні та готовим до практичного застосування.

Даний веб-додаток може бути вдосконалено та наповнено функціями відповідно до необхідних вимог, завдяки модульній архітектурі додатку.

Перелік посилань

Контакт. Медовік [Електронний ресурс]. – Режим доступу: <http://vk.com/medovik_su>

Контакт. Пасічник [Електронний ресурс]. – Режим доступу: <http://vk.com/club20828336>

Контакт. Медова імперія [Електронний ресурс]. – Режим доступу: http://vk.com/medovaja\_imperija

Бджільництво [Електронний ресурс]. – Режим доступу: http://pasika.if.ua/index.php/bdjilnytstvo-vid-a-do-ya

Портал бджільництва [Електронний ресурс]. – Режим доступу: <http://apis.at.ua/>

1. Библиотека класов С# [Електронний ресурс] – Режим доступу: <http://msdn.microsoft.com/>
2. Интернет-Университет Информационных Технологий [Електронний ресурс]. – Режим доступу: http://www.INTUIT.ru
3. Microsoft SQL Server [Електронний ресурс]. – Режим доступу: http://ru.wikipedia.org/wiki/SQL\_Server.
4. Портал програмістів [Електронний ресурс]. – Режим доступу: http://professorweb.ru/my/ASP\_NET/base/level1/aspnet\_info.php
5. Хабрахабр. Unit-тестирование средствами .NET[Електронний ресурс]. – Режим доступу: <http://habrahabr.ru/post/98062/>
6. DevExpress [Електронний ресурс]. – Режим доступу: <https://www.devexpress.com/>
7. Портал бджільництва. Бджола - 2professional [Електронний ресурс]. – Режим доступу: <http://www.euro-honey.com/pchela/pchela2.html>
8. Портал бджільництва. Комп’ютер та бджоли [Електронний ресурс]. – Режим доступу: <http://p4elovod.com/programmy>
9. Мак-Дональд, Мэтью, Фримен, Адам, Шпушта, Марио. М15 Microsoft ASP.NET 4.0 с примерами на C# 2010 для профессионалов, 4-е изд. : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2011. — 1424 с. : ил. — Парал. тит. англ.

Додатки

Додаток А. Даталогічна модель БД