ЗМІСТ

[ВСТУП 6](#_Toc485119634)

[1. Задача розробки web-додатку внесення первинних даних для екологічного моніторингу стану підземних вод 7](#_Toc485119635)

[1.1 Функціональні вимоги до web-додатку 8](#_Toc485119636)

[1.2 Класифікація критеріїв оцінки якості води 9](#_Toc485119637)

[1.3 Норми показників якості води 22](#_Toc485119638)

[2. Огляд існуючих програмних рішень для збору первинної інформації стану підземних вод 26](#_Toc485119639)

[3. Засоби розробки програмного продукту 28](#_Toc485119640)

[3.1 Веб-сервер Windows Server (IIS) 28](#_Toc485119641)

[3.2 Microsoft Visual Studio 2017 29](#_Toc485119642)

[3.3 Microsoft .NET Framework 30](#_Toc485119643)

[4. Програмна реалізація web-додатку збору даних стану вод 32](#_Toc485119644)

[4.1 Обґрунтування архітектури програмного засобу 32](#_Toc485119645)

[4.2 Технології реалізації web-додатку збору даних стану підземних вод 36](#_Toc485119646)

[4.2.1 Entity Framework 37](#_Toc485119647)

[4.2.2 Web-модель ASP.NET MVC 5 та її переваги 39](#_Toc485119648)

[4.2.3 Шаблони програмування 41](#_Toc485119649)

[4.3 Концептуальна модель бази даних 42](#_Toc485119650)

[4.4 Програмні модулі web-додатку 46](#_Toc485119651)

[5. Методика роботи користувача з програмною системою 49](#_Toc485119652)

[5.1 Системні вимоги до використання 49](#_Toc485119653)

[5.2 Опис графічного інтерфейсу системи 49](#_Toc485119654)

[ВИСНОВКИ 58](#_Toc485119655)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 59](#_Toc485119656)

[Додаток 1 61](#_Toc485119657)

[Додаток 2 64](#_Toc485119658)

[Додаток 3 74](#_Toc485119659)

[Додаток 4 84](#_Toc485119660)

[Додаток 5 87](#_Toc485119661)

[Додаток 6 91](#_Toc485119662)

[Додаток 7 94](#_Toc485119663)

# ВСТУП

Моніторинг навколишнього середовища – основа для нашого майбутнього успішного існування. Саме він є першим і основним інструментом для попередження, уникнення та усунення усіх можливих шкод для навколишнього середовища. Під моніторингом розуміються дії, які направленні на збір, обробку, аналіз та застосування екологічної інформації про стан природнього об’єкту.

Проблема екології в нашій країні постає дуже гостро. В першу чергу це стосується водних об’єктів. Особливо тих об’єктів, які є першочерговими для успішного і здорового існування людини. До таких об’єктів відносяться підземні води. Саме вони є основним джерелом питної води, особливо у сільських місцевостях, де очищення води не проводиться через брак воодоочисних споруд. Забруднення підземних вод може спричинити масове отруєння людей та гибель живих культур. Тому моніторинг стану підземних вод є необхідною та важливою ланкою у мережі моніторингу природнього середовища в Україні.

# 1. Задача розробки web-додатку внесення первинних даних для екологічного моніторингу стану підземних вод

Веб-додаток розробляється як базовий програмний засіб для збору та зберігання первинних даних про стан підземних вод в Україні. Першочергова мета такої роботи – заняття гідної ланки в складній системі державного моніторингу навколишнього середовища, зокрема водних ресурсів. Такий засіб є необхідним, тому що він може стати механізмом отримання інформації як для подальшого дослідження і виявлення причин забруднення підземних водних ресурсів так і попередження таких. На разі за внесення повної інформації про водні об’єкти, він може стати інструментом для дослідників водних об’єктів, працівників різних державних служб та природоохороних комплексів та міністерств. Веб-додаток дозволяє вчасно та ефективно вносити заміри та отримувати попередньо оброблену інформацію за лічені секунди.

Така система має тенденції до розвитку, адже вона може стати джерелом збору інформації про інші природні об’єкти, які потребують контролю та моніторингу. Наприклад, земля, ліс, повітря тощо.

Також веб-додаток буде актуальним і для виконавчого апарату країни. За залученням додаткових опції ця інформація може бути використана для складання документів податкової та обрахункової палати, для чиновників – для вчасного реагування, попередження та виділення коштів для заподіяння непоправної шкоди природним комплексам.

Підприємства також мають бути зацікавлені у розвитку такої бази, адже це дозволить спростити і зробити прозорими процедури, пов’язані з моніторингом довкілля на території.

## 1.1 Функціональні вимоги до web-додатку

Функціональними вимогами до web-додатку є (рисунок 1.1.1):

– введення даних показників стану об’єктів моніторингу за шкалами і перевірка їх за нормами (для створення нового забору води обов`язково має бути введеною дата забору, інформація про особу, хто вводить дані (така інформація може бути доступною після авторизації), місце забору (ідентифікатор місця або інше) і температура. Усі інші показники доступні для подальшого редагування запису);

– перегляд інформації по забору води;

– перегляд основної інформації по лабораторіях, станціях, працівниках;

* редагування основної інформації по заборах води, лабораторіях, станціях, працівниках;

– редагування показників забору води (окрім температури – вона вводиться лише раз, при створенні нового забору води);

– відображення маркерами на карті, пунктів забору води (різними за кольором маркерами за класами, клас вираховується програмно в залежності від результатів останнього дослідження даного пункту);

– відображення інформації про пункт забору води на карті за допомогою додаткових вікон тощо;

– можливість перегляду динаміки показників в джерелі води за певний період;

– дата забору генерується програмою і не доступна для подальшого редагування, дата здачі зразка до лабораторії може генеруватися програмою, але доступна для редагування. Без введення лабораторної дати не змінюється статус досліджуваного зразка.

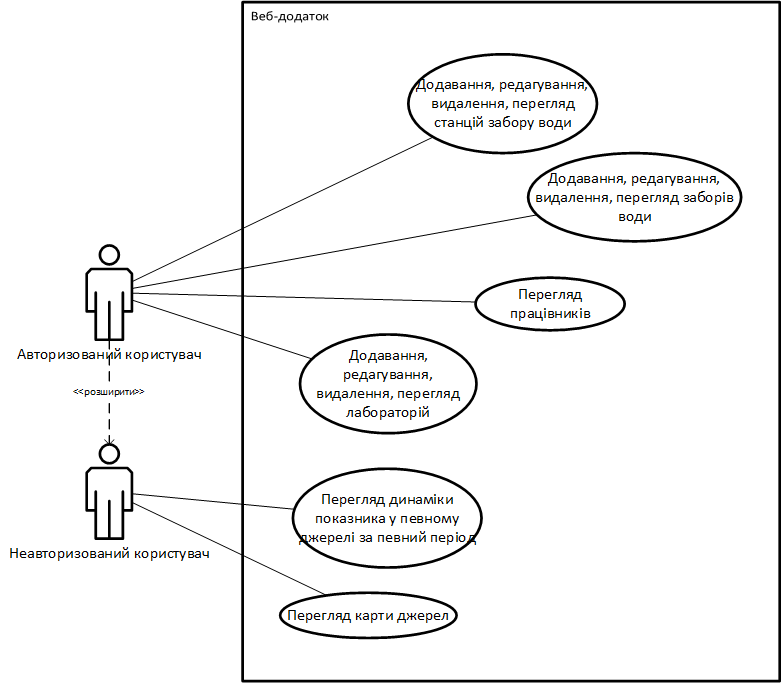


Рисунок 1.1.1 – Діаграма прецедентів для опису сценарію роботи web-додатку

## 1.2 Класифікація критеріїв оцінки якості води

Показники якості води поділяються на 2 групи: органолептичні та хімічні [1] (рисунок 1.2.1).

Перша група показників відповідає за індикатори, які можуть бути відзначені людиною без будь-яких маніпуляцій зі зразком або із незначними первинними операціями. Частіше їх можна виміряти з використанням підручних засобів.

Друга група показників – індикатори, які потребують лабораторних дослідів та спеціальних реагентів, спеціальних знань, а отже можуть бути виконані лише у спеціальних приміщеннях спеціальним персоналом – лаборантами. Для таких досліджень зразки потрібно здавати до спеціальної установи – лабораторії, яка вповноважена проводити такі досліди та має спеціальні дозволи.

На забруднення води може впливати фізичні та хімічні фактори забруднення.

Фізичне забруднення води відбувається внаслідок накопичення в ній нерозчинних домішок - піску, глини, мулу в результаті змивання дощовими водами з розорених ділянок (полів), надходження суспензій з підприємств гірничодобувної промисловості, потрапляння пилу, що переноситься вітром в суху погоду тощо.

Хімічне забруднення води відбувається через надходження у водойми зі стічними водами різних шкідливих домішок неорганічного (кислоти, луги, мінеральні солі, мінеральні добрива) та органічного (нафта й нафтопродукти, миючі засоби, органічні добрива тощо) складу. Шкідлива дія токсичних речовин, що потрапляють у водойми, по­силюється за рахунок так званого кумулятивного ефекту (прогресуюче збільшення вмісту шкідливих сполук у кожній наступній ланці трофічного ланцюга).

Органолептичні показники

Температура води – показник, який часто впливає на всі процеси, від яких залежить склад і властивості води.

Кольоровість — природна властивість води, зумовлена гуміновими речовинами, які вимиваються з ґрунту під час формування поверхневих та підземних водойм і надають воді жовто-коричневого забарвлення. Кольоровість вимірюють у градусах за допомогою спектрофотометрів та фотоколориметрів шляхом порівняння із забарвленням розчинів хромово-кобальтової чи платиново-кобальтової шкали, які імітують кольоровість природної води. Для доброї води колір допускають не більше 20-300.

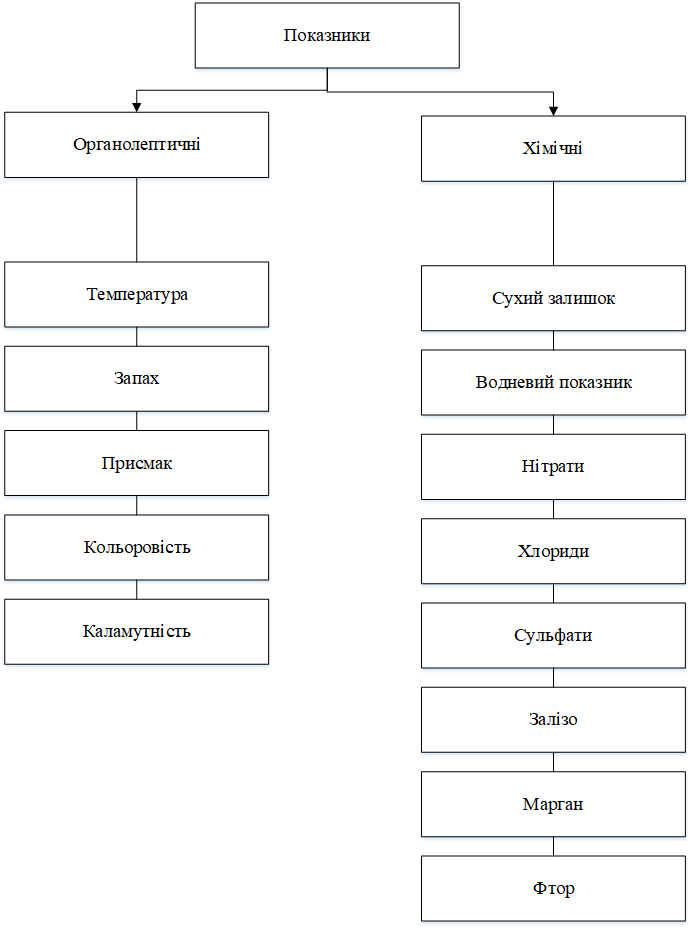


Рисунок 1.2.1 Показники якості води

Забруднена вода може мати неприродний колір, зумовлений барвниками, які можуть потрапляти у водойми зі стічними водами підприємств легкої промисловості, деякими неорганічними сполуками як природного, так і техногенного походження. Так, залізо і марганець можуть спричиняти забарвлення води від червоного до чорного, мідь – від блідо-блакитного до синьо-зеленого. Цей показник зветься забарвленням води.

Каламутність — природна властивість води, зумовлена вмістом завислих речовин органічного і неорганічного походження (глини, мулу, органічних колоїдів, планктону і т. ін.). Каламутність вимірюють нефелометрами, спектрофотометрами та фотоколориметрами за імітуючою каоліновою шкалою, яка являє собою набір суспензій білої глини каоліну у дистильованій воді. Каламутність води вимірюють в мг/л шляхом порівняння її оптичної щільності зі щільністю стандартних суспензій каоліну, згідно ДСанПіН 383 – в нефелометричних одиницях каламутності (НОК). Каламутність води визначається кількістю органічних і мінеральних речовин, які перебувають у воді в завислому і колоїдному стані, в результаті чого вона може бути каламутною. Крім того, в воді може зустрічатися певна кількість розчинених у ній двовуглекислих солей закису заліза, які під час відстоювання води випадають у вигляді гідрату оксиду феруму, в результаті чого з’являється каламуть. Тому ступінь прозорості води має бути не менше як 30 см (це висота стовпа води в циліндрі).

Запах води – дана властивість створюється специфічними речовинами, які надходять у воду в результаті життєдіяльності гідробіонтів, біохімічного розкладання органічних речовин, хімічної взаємодії компонентів, що містяться у воді, а також надходження з промисловими. господарсько-побутовими стічними водами тощо. Запах води залежить від складу речовин, які містяться в ній, від рН водного середовища, температури, рівня забруднення води і гідрологічних умов [10, 15].

Смак води – він буває гіркий, кислий, солоний. Усі інші смакові відчуття кваліфікуються як присмак. Смак може бути солоний, солодкий, гіркий, а також з іншими присмаками. Інтенсивність запахів і присмаків за балами поділяють так: дуже слабкий – 0, слабкий – 1, помітний – 2, виразний – 3, дуже виразний – 4. Наявність у воді хлориду натрію, хлориду калію понад 500 мг/л надає воді солоний смак, а кількість солей магнію у воді понад 1000 мг/л спричиняє у ній гіркий смак води. Вміст у воді двовуглекислих солей закису заліза, сульфату міді, солі кальцію призводять до чорнильного і в’яжучого смаку. Продукти гниття органних речовин надають воді неприємного затхлого, гнилісного, сірководневого смаку, а гумунові кислоти – болотистого смаку.

Одиниці виміру показників стану води та умови отримання значень показників:

– температура (в Цельсіях) – під час забору;

– запах (в балах) – при температурі 20 С;

– запах (в балах) – при температурі 60 С;

– присмак (в балах);

– каламутність;

– кольоровість (в градусах).

Хімічні показники

До хімічних показників води належать лужна реакція води, сухий залишок, твердість води, кількість у воді нітритів і нітратів, хлоридів, сульфатів, феруму тощо.

Сухий залишок (мінералізація загальна) — це кількість розчинених речовин, переважно (90 %) мінеральних солей, в 1 л води. Воду з сухим залишком до 1000 мг/л називають прісною, від 1000 до 3000 мг/л – солонуватою, понад 3000 мг/л ­– солоною. Оптимальною вважається мінералізація на рівні 300—500 мг/л. Вода із сухим залишком 100—300 мг/л вважається задовільно мінералізованою, 500—1000 мг/л — підвищено, але допустимо мінералізованою. Мікроелементами називають речовини, які знаходяться у воді в малих кількостях (менше 1 мг/л). Багато з них необхідні для життєдіяльності організмів, але підвищена концентрація деяких з них у воді може перетворити воду на отруту. До мікроелементів відносяться: бром В, йод І, фтор F, літій Li, барій Ba та інші. Дослідженнями доведено, що концентрація фтору 1,5-2 мг/кг у воді призводить до захворювання флюорозу у людини і тварини. Якщо низький вміст фтору у питній воді (менше 0,4 мг/кг), то це призводить до карієсу зубів у дітей. При нестачі йоду в питній воді у тварин можу виникнути хвороба у тварин, яка називається ензоотичним зобом.

Солонувата і солона вода неприємна на смак. Вживання такої води супроводжується підвищенням гідрофільності тканин, затримкою води в організмі, зменшенням на 30—60 % діурезу, внаслідок чого підвищується навантаження на серцево-судинну систему, стає тяжчим перебіг ішемічної хвороби серця, міокардіодистрофії, гіпертонічної хвороби, підвищується ризик їх загострення; може спричинити диспепсичні розлади у осіб, котрі змінили місце проживання, внаслідок зміни секреторної і моторної функцій шлунку, подразнення слизових оболонок тонкої і товстої кишок і посилення їхньої перистальтики; сприяє розвитку і тяжкості перебігу сечокам’яної і жовчнокам’яної хвороб.

Систематичне вживання маломінералізованої води призводить до порушення водно-електролітного гомеостазу, яке ґрунтується на реакції осморецептивного поля печінки, що зумовлює підвищений викид натрію в кров і супроводжується перерозподілом води між позаклітинною та внутрішньоклітинною рідиною.

Водневий показник (рН). Визначення рН водного середовища є показником концентрації у воді іонів водню. Існує шкала значень рН від 0 до 14, згідно з якою водний розчин з рН – 7 нейтральний, розчин з рН менше 7 кислий, а з рН більше 7 лужний. Для більшості мешканців водних екосистем оптимальне значення рН не повинне значно відхилятися в бік кислотності чи лужності. Значення рН водойми залежить від багатьох факторів, насамперед від вмісту у воді органічних речовин, тобто від того, наскільки водойма евтрофічна чи оліготрофічна. Окиснення органічних речовин редуцентами супроводжується виділенням у воду вуглекислого газу, який підвищує кислотність води, а отже, знижує рН. Саме з цієї причини екосистеми оліготрофічних і евтрофічних водойм відрізняються своїм складом [10]. Якщо рН досліджуваної водойми менше ніж 7 або більше за 8,5 і ніяких природних причин цього явища не встановлено, то джерело забруднення водойми буде штучного походження. До промислових підприємств, що найсильніше впливають на рН водних джерел, належать хімічні заводи, заводи штучних добрив, деревообробні й паперові комбінати, молочні комбінати й металургійні підприємства. рН водних джерел значною мірою знижують кислотні дощі, головною причиною яких є теплові електростанції і металургійні комбінати.

Особливо токсичними вважаються нітрити (вони є нестабільними сполуками і у воді швидко переходить у нітрати). Аміак нейтралізується хлором і тому зникає в процесі хлорування питної води. Вміст нітратів у питній воді не повинен перевищувати 45 мг/л.. До шкідливих ефектів, які спричиняються високим вмістом у водоймах азотних сполук, можна віднести зниження кількості розчиненого у воді кисню, появу неприємного запаху й смаку, засмічування фільтрів водоочисних споруд мікро водоростями, зниження природних властивостей водойм.

Жорсткість загальна — природна властивість води, зумовлена наявністю так званих солей жорсткості, а саме: кальцію і магнію (сульфатів, хлоридів, карбонатів, гідрокарбонатів та ін.). Розрізняють загальну, усунену, постійну й карбонатну жорсткість.

Вміст солей жорсткості понад 7 мг-екв/л надає воді гіркого смаку. Різкий перехід від м’якої води до жорсткої може призвести до диспепсії. У районах зі спекотливим кліматом користування водою з високою жорсткістю призводить до погіршення перебігу сечокам’яної хвороби. Солі жорсткості погіршують всмоктування жирів внаслідок їхнього омилення і утворення в кишечнику нерозчинних кальцієво-магнезіальних мил. При цьому обмежується надходження в організм ПНЖК, жиророзчинних вітамінів, деяких мікроелементів (вода з жорсткістю понад 10 мг-екв/л підвищує ризик захворювання на ендемічний зоб). Висока жорсткість сприяє виникненню дерматитів внаслідок подразливої дії кальцієво-магнезіальних мил, котрі утворюються при омиленні шкірного сала. З підвищенням жорсткості води ускладнюється кулінарна обробка харчових продуктів (гірше розварюється м’ясо і бобові, погано заварюється чай, утворюється накип на стінках посуду), підвищуються витрати мила, волосся після миття стає жорстким, шкіра грубішає, тканини жовтіють, втрачають м’якість, гнучкість, вентилюючи здатність за рахунок імпрегнації кальцієво-магнезіальних мил.

Тривале користування м’якою водою, збідненою на кальцій, може призвести до його дефіциту в організмі (у дітей, які мешкають у районах з м’якою водою, на зубній емалі утворюються лілові плями, які є наслідком декальцинації дентину; урівська хвороба (хвороба Кашина–Бека), яка є ендемічним полігіпермікроелементозом стронцію, заліза, марганцю, цинку, фтору, виникає в місцевостях з низьким вмістом кальцію в питній воді). Вода з низьким вмістом електролітів, які зумовлюють жорсткість, сприяє розвитку серцево-судинних захворювань.

Нітрати і нітрити. Наявність у воді альбуміноїдного аміаку, а також солей аміаку і нітратної кислоти свідчить про забруднення її органічними речовинами тваринного походження (гній, сеча), в результаті чого ця вода є шкідливою за санітарним станом. Якщо концентрація нітритів у питній воді перевищує 30 мг/кг і цю воду споживають, то виникає хвороба метгемаглобінемія у людському організмі. За нормативами у доброякісній питній воді аміаку і нітратів не повинно бути взагалі. Але якщо у воді є кількість нітратної кислоти і не має аміаку та солей нітратної кислоти, то це свідчить про те, що процес окислення (мінералізації) завершився і така вода є не шкідливою. Якщо водночас у воді з солями нітратної кислоти є кількість солей нітритної кислоти і аміаку, то дане джерело водопостачання являється забрудненим.

Хлориди. Наявність хлоридів у воді може бути в разі забруднення її стічними водами, сечею, гноївкою води, джерело якої протікає по солончакових ґрунтах. багатих на хлористі сполуки. Якщо по солончакових ґрунтах протікає джерело води з вмістом 300-500 мг/л хлоридів, то така вода називається солончаковою, є нешкідливою і придатною для господарського-питного використання. Якщо у воді присутня кількість аміаку, солей нітритної кислоти і певна кількість хлоридів у воді не пов’язана з солончаковим ґрунтом, то така вода є непридатною для напування тварин і недоброякісною. Вміст хлоридів в воді встановлюють залежно від походження хлориду – тваринного або мінерального.

Залізо. Підвищена кількість заліза у воді впливає негативно на органолептичні показники води, в результаті чого вода стає мутною, має бурий колір, болотистий присмак. У молочному виробництві вода з великим вмістом заліза є непридатною для використання, оскільки вона надає молоку і молочним продуктам поганого присмаку, а у виробах утворюються іржаві плями.

Марганець. В концентраціях, що перевищують 0,15 мг/л, марганець зумовлює забарвлення води в рожевий колір, надає їй неприємного присмаку, зафарбовує при пранні білизну, утворює накип на посуді. Якщо сполуки марганцю (ІІ) у воді піддаються окисленню, то негативний вплив на органолептичні властивості посилюється (при аерації води, яка містить марганцю більше за 0,1 мг/л, буде утворюватися темно-бурий осад MnО2, при озонуванні з метою знезараження за рахунок утворення солей Mn7+ (перманганатів) може виникнути рожеве забарвлення).

Сульфати. Вміст сульфатів або солей сульфатної кислоти є у волі, коли проходять процеси окислення білкових речовин, які мають сульфур. Сульфати також зустрічаються у воді, яка містить гіпс. Вміст сульфатів у воді залежить від біологічних умов окремих регіонів. Вміст у воді сульфатів натрію і магнію призводить до гіркого смаку, має послаблювальну дію, що в подальшому спричиняє розлад шлунково-кишкового тракту у тварин. Кількість сульфатів у волі деяких районів може досягти до 2000-3000 мг/л.

Отже, екологічний стан водойм потрібно оцінювати особливо за санітарно-токсикологічними показниками, які в подальшому можуть свідчити про безпечність і шкідливість питної води.

Одиниці виміру показників стану води:

– сухий залишок, мг/дм3;

– водневий показник (pH);

– жорсткість, мг-екв/дм3;

– нітрати, мг/дм3;

– хлориди, мг/дм3;

– сульфати, мг/дм3;

– залізо загальне, мг/дм3;

– марган, мг/дм3;

– фтор, мг/дм3.

Класи забрудненності підземних вод.

В системі екологічного моніторингу підземних джерела води поділяються на класи в залежності від концентрації певних речовин у воді.

Джерело води відноситься до 1-го класу, якщо значення усіх показників останнього забору води не перевищують значення відповідних показників відповідного класу небезпеки в ГОСТ – 2761 – 84.

Джерело води відноситься до 2-го класу, якщо значення усіх показників останнього забору води не перевищують значення відповідних показників відповідного класу небезпеки в ГОСТ – 2761 – 84.

Джерело води відноситься до 3-го класу, якщо значення усіх показників останнього забору води не перевищують значення відповідних показників відповідного класу небезпеки в ГОСТ – 2761 – 84.

Джерело води вважається непридатним до вживанням, якщо значення хоча б одного з показників останнього забору води перевищує значення відповідного показника 3-го класу небезпеки в ГОСТ – 2761 – 84.

Джерело води вважається непридатним до вживанням, якщо значення хоча б одного з показників забору води не є у інтервалі допустимих значень відповідного показника небезпеки в ГОСТ – 2761 – 84.

## 1.3 Норми показників якості води

Залежно від якості води, методів та засобів, необхідних для отримання доброякісної питної води, підземні та поверхневі джерела поділені на три класи (таблиця 1.3.1) [2].

Таблиця 1.3.1. Норми показників

| Показники якості води | Вид джерела води | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| підземні води | | | поверхневі води | | |
| Клас | | | | | |
| Ι | ΙΙ | ΙΙΙ | Ι | ΙΙ | ΙΙΙ |
| Органолептичні: | | | | | | |
| Запах при 20 С і 60 С, бали | 2 | 2 | 2 | 2 | 3 | 4 |
| Присмаки, бали | 2 | 2 | 2 | 2 | 3 | 4 |
| Колірність, градуси | 20 | 20 | 50 | 35 | 120 | 200 |
| Температура,°С | 8-12 | 8-12 | 8-12 | 8-25 | 8-25 | 8-25 |
| Показники природного хімічного складу (вибірково): | | | | | | |
| Сухий залишок, мг/дм3 | 1000-1500 |  |  | 1000-1500 |  |  |
| Водневий показник рН | 6-9 |  |  | 6,5-8,5 |  |  |
| Жорсткість, мг-екв/дм3 | 7-10 |  |  | 7-10 |  |  |
| Хлориди, мг/дм3 | 350 |  |  | 350 |  |  |
| Сульфати, мг/дм3 | 500 |  |  | 500 |  |  |
| Залізо, мг/дм3 | 0,3 | 10 | 20 | 1 | 3 | 5 |
| Марган, мг/дм3 | 0,1 | 1,0 | 2,0 | 0,1 | 1,0 | 2,0 |
| Фтор, мг/дм3 | 1.5 | 1.5 | 5,0 |  | 0,1-0,5 |  |
| Нітрати, мг/дм3 | 45 |  |  | 45 |  |  |

Для деяких органолептичних показників існують спеціальні шкали, наприклад, для запаху і присмаку води [4] (таблиця 1.3.2), для водневого показника (рисунок 1.3.3).

Таблиця 1.3.2 Шкала інтенсивності запаху і присмаку

|  |  |
| --- | --- |
| Інтенсивність, Бал | Запах і присмак |
| 0 | не виявляються |
| 1 | виявляються лише досвідченим дегустатором |
| 2 | виявляються споживачем |
| 3 | виявляються легко і є причиною скарг |
| 4 | сильний запах і присмак, що роблять воду непридатною для пиття |
| 5 | мають таку інтенсивність, при якій вода непридатна для пиття |

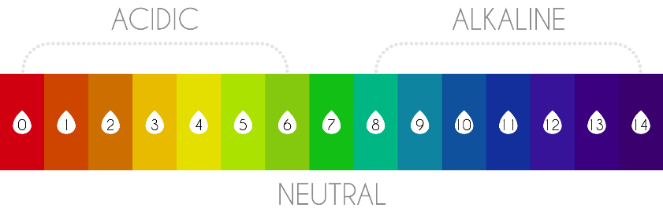


Рисунок 1.3.3 Шкала інтенсивності водневого показника

# 2. Огляд існуючих програмних рішень для збору первинної інформації стану підземних вод

В результаті проведеного в дипломному проектуванні пошуку та аналізу існуючих програмних рішень поставленої задачі виявлено, що засоби, які б були загальновідомими та розповсюдженими для використання у вітчизняних санітарних та екологічних установах держави, відсутні. Тому, щоб системно займатися проблемою моніторингу стану підземних вод, потрібно організувати та налагодити складний процес, що включає в себе наступні кроки, а саме:

– По-перше, деякі стандарти та накопичені статистичні дані щодо питання стану вод можна отримати, якщо проаналізувати інформацію з офіційних регіональних сайтів водоканалів або вповноважених державних структур, інформаційних довідників з інформацією про лабораторії та закони України, спеціальних актів щодо кількості та місцезнаходження станцій для забору води.

– По-друге, виконавши такий досить складний пошук та збір даних, потрібно вручну здійснити обчислення щодо отримання висновків про стан вод для місцевостей чи регіонів, в яких проводиться моніторинг.

– Наступним кроком є залучання геоінформаційних систем та електронних мап-ресурсів для побудування мап з результатами збору та обчислення даних для використання цих мап в моніторингу.

Якщо ці незалежні дії та засоби об’єднати у спільну систему (рисунок 2.1), можливо отримати стандартизовану, централізовану, досить зручну у використанні єдину систему по збору та обробці первинної інформації, яка б дозволяла спростити підготовку всіх даних, необхідних для моніторингу стану ґрунтових вод, на рівні як окремого регіону чи місцевості, так і держави в цілому.

Для того, щоб розроблювальна програмна система була корисною та ефективною, розрахунки в розробленій системі виконуються з залученням актуальної інформації про стан та норми підземних вод. Тому важливо не тільки розробити та відлагодити саму систему обробки інформації, але й розробити зручні та легкі в використанні модулі та пристрої, що дозволять на місцевості забирати дані проб підземних вод, виконуючи швидко їх лабораторний аналіз, та заносити в базу даних центральної системи. Лише при виконанні таких умов програмне забезпечення виправдає і себе, і витрати, пов’язані з її розробкою та впровадженням.

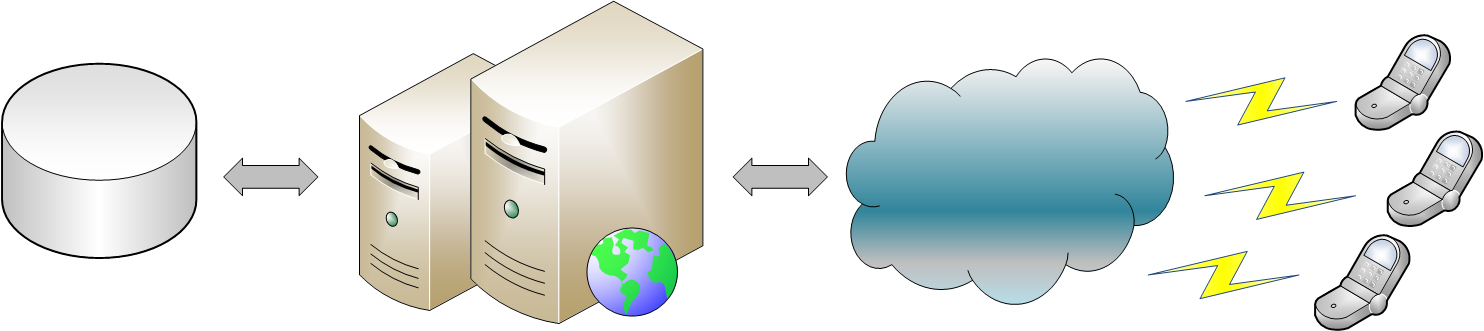


Рисунок 2.1 – Структура веб-системи збору первинної інформації стану підземних вод

# 3. Засоби розробки програмного продукту

Для розробки даного програмного продукту необхідні наступні апаратні та програмні складові:

– веб-технологія для створення графічної оболонки web-додатку;

– сервер для розгортання веб-сервісу;

– програмний засіб для реалізації.

## 3.1 Веб-сервер Windows Server (IIS)

Роль веб-сервера (IIS) - забезпечити безпечну, легко керовану, модульну та розширювану платформу для надійного розміщення веб-вузлів, служб і додатків. Використання веб-сервера Служби IIS 8 забезпечує доступ до інформації користувачам в Інтернеті, інтрамережі і екстранет. Автоматизація призначеного для користувача інтерфейсу також дозволяє скриптам автоматичних тестів взаємодіяти з UI.

Переваги використання Служби IIS 8 для розгортання та хостынгурозроблюваного web-додатку:

– максимізація рівня веб-безпеки завдяки скороченню обсягу сервера і автоматичної ізоляції додатків;

– просте розгортання та запуск веб-додатків ASP.NET, Classic ASP і PHP на одному сервері;

– ізоляція додатків шляхом унікальної ідентифікації робочих процесів і їх запуску в ізольованому середовищі за замовчуванням, що скорочує ризики небезпеки;

– просте додавання, видалення і заміна вбудованих компонентів IIS з налаштованими модулями, що відповідають потребам користувача;

– підвищення швидкості роботи веб-сайту за допомогою вбудованого динамічного кешування і розширеного стиснення.

## 3.2 Microsoft Visual Studio 2017

Середовище Microsoft Visual Studio – інтегроване середовище розробки програмного забезпечення у поєднанні з іншими інструментальними засобами. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом (Windows Forms та WPF), а також веб-сайти, веб-додатки, веб-служби в керованому коді для всіх платформ, що підтримують операційні системи Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight.

Інтегроване середовище розробки програмного забезпечення Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Решта вбудованих інструментів включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server).

## 3.3 Microsoft .NET Framework

.NET Framework – програмна платформа, випущена компанією Microsoft. Основою платформи є загальномовне середовище виконання Common Language Runtime (CLR), яке використовується для різних мов програмування. Функціональні можливості CLR доступні в будь-яких мовах програмування, що використовують це середовище.

.NET є патентованою технологією корпорації Microsoft і офіційно розрахована на роботу під операційними системами сімейства Microsoft Windows, але існують незалежні проекти (перш за все це Mono і Portable.NET), що дозволяють запускати програми .NET на деяких інших операційних системах.

Основною ідеєю при розробці .NET Framework було забезпечення потреб розробника за рахунок надання йому можливості створювати додатки різних типів, які здатні виконуватися на різних типах пристроїв і в різних середовищах.

Другим принципом стала орієнтація на системи, що працюють під управлінням сімейства операційних систем Microsoft Windows [26].

Архітектура .NET Framework описана і опублікована в специфікації Common Language Infrastructure (CLI), розробленої Microsoft і затвердженої ISO і ECMA. У CLI описані типи даних .NET, формат метаданих про структуру програми, система виконання байт-коду і багато іншого.

Об'єктні класи .NET, доступні для всіх підтримуваних мов програмування, містяться в бібліотеці Framework Class Library (FCL). У FCL входять класи Windows Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation та інші. Ядро FCL називається Base Class Library (BCL).

.NET підтримують наступні середовища розробки:

– Microsoft Visual Studio (C#, Visual Basic .NET, Managed C++, F#);

– SharpDevelop;

– MonoDevelop;

– Embarcadero RAD Studio (Delphi for .NET); ранее Borland Developer Studio (Delphi for .NET, C#);

– Zonnon;

– PascalABC.NET;

– JetBrains Rider.

Для розробленої системи необхідно мати встановлений .NET Framework версії 4.0 або вище.

# 4. Програмна реалізація web-додатку збору даних стану вод

Для успішної реалізації та подальшого розвитку рекомендується будувати програмне забезпечення, використовуючи певний архітектурний шаблон, що дозволить виділити та при необхідності відокремити структурні частини програмної системи, розроблюючи їх незалежно, із застосуванням різних технологій та достатньо швидко заміняючи один варіант їх реалізації іншим.

## 4.1 Обґрунтування архітектури програмного засобу

В контексті даної задачі найбільш доцільною є багаторівнева архітектура.

Вона здатна забезпечити поступову розробку кожного компонента без внесення принципових змін у інші, забезпечуючи для web-додатку логічний розподіл функцій і задач.

На разі існує та має певні переваги ціла низка видів і типів архітектур, які успішно застосовуються при розробці сучасного програмного забезпечення. Однієї з найбільш використовуваних є класична трирівнева система, яка передбачає поділ додатка на три рівні.

В багаторівневій архітектурі розрізняють два окремих поняття: n-layer і n-tier. І layer, і tier, як правило, позначаються словом "рівень", іноді по відношенню до "layer" ще вживається слово "шар". Однак в обох випадках рівні будуть різного порядку.

Шар Tier представляє фізичний рівень. Тобто якщо йде мова про трирівневу архітектуру, то n-tier додаток може бути розділеним на такі рівні: сервер бази даних, веб-додаток на веб-сервері і браузер користувача. І кожен рівень представляв би особливий окремий фізичний процес, навіть якщо б і сервер баз даних, і веб-сервер, і браузер користувача знаходилися б на одному комп'ютері. Якби в якості клієнта альтернативно використовувалося мобільний додаток, то це був би ще один фізичний рівень.

Шар Layer представляє логічний рівень. Тобто в системі може бути рівень доступу до даних, рівень бізнес-логіки, рівень уявлення, рівень сервісів тощо. При цьому логічні рівні не збігаються з фізичними. Так, звичайно рівень надання в додатку ASP.NET містить і контролери, які обробляють введення, і уявлення, які відображаються в веб-браузері, тобто розділяється на два фізичних рівня.

В даному випадку при побудові архітектури будуть застосовуватись поняття логічного рівня, тобто web-додаток розроблюється згідно n-layer архітектурі.

Класична трирівнева система складається з наступних рівнів на рисунок 4.1.1.

Рівень Presentation layer (рівень представлення): це рівень, з яким безпосередньо взаємодіє користувач. Цей рівень включає компоненти для користувача інтерфейсу, механізм отримання введення від користувача. Стосовно до asp.net mvc на даному рівні розташовані представлення (Views) і всі супутні компоненти, які складають призначений для користувача інтерфейс (стилі, статичні сторінки html, javascript), а також моделі уявлень, контролери, об'єкти контексту запиту.

Рівень Business layer (рівень бізнес-логіки): містить набір компонентів, які відповідають за обробку отриманих від рівня уявлень даних, реалізує всю необхідну логіку додатка, необхідні для домену обчислення, взаємодіє (не напряму) з базою даних і передає рівню подання результат обробки для відображення користувачеві додатку.

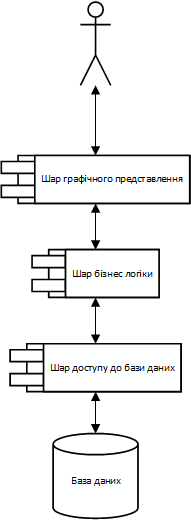


Рисунок 4.1.1 – N-Layer Architecture in ASP.NET

Рівень Data Access layer (рівень доступу до даних): зберігає моделі, що описують використовувані суті, також містить специфічні класи для роботи з різними технологіями доступу до даних, наприклад, клас контексту даних Entity Framework. Тут також зберігаються репозиторії, через які рівень бізнес-логіки взаємодіє з базою даних.

При цьому треба зазначити, що крайні рівні не можуть взаємодіяти між собою, тобто рівень представлення (стосовно ASP.NET MVC, контролери) не можуть безпосередньо звертатися до бази даних і відповідно до рівня доступу до даних, а тільки через рівень бізнес-логіки.

Рівень доступу до даних не залежить від інших рівнів, рівень бізнес-логіки залежить від рівня доступу до даних, а рівень представлення - від рівня бізнес-логіки.

Компоненти, як правило, повинні бути слабо зв'язані (loose coupling), тому невід'ємною ланкою багаторівневих додатків є впровадження DI-залежностей.

Щодо технології розробки web-додатків .Net ASP.NET MVC – вона використовується для розробки саме web-частини додатку, а саме рівня представлення, інші ж рівні можуть бути реалізовані незалежно і можуть використовуватися в додатках на інших технологіях, як Windows Forms, WPF і т.д. І, як правило, весь додаток в цілому буде представляти рішення (solution) в Visual Studio, а окремі рівні - проекти. При цьому необов’язково одному рівню повинен відповідати окремий проект. При складності рівня можливо роздрібнювати один рівень на кілька проектів, головне, щоб його функціонал представляв єдину логічну ланку.

## 4.2 Технології реалізації web-додатку збору даних стану підземних вод

Для додатку було використано технологію Entity Framework, застосовано модель ASP.NET MVC 5 та використані шаблони програмування.

### 4.2.1 Entity Framework

Технологія Entity Framework являє спеціальну об'єктно-орієнтовану технологію на базі фреймворка .NET для роботи з даними. Якщо традиційні засоби ADO.NET дозволяють створювати підключення, команди та інші об'єкти для взаємодії з базами даних, то Entity Framework являє собою більш високий рівень абстракції, який дозволяє абстрагуватися від самої бази даних і працювати з даними незалежно від типу сховища. Якщо на фізичному рівні присутнє оперування таблицями, індексами, первинними і зовнішніми ключами, але на концептуальному рівні, який пропонує Entity Framework, виконується робота з об'єктами ООП.

Центральної концепцією Entity Framework є поняття сутності або entity. Сутність представляє набір даних, асоційованих з певним об'єктом. Тому дана технологія передбачає роботу не з таблицями, а з об'єктами і їх наборами.

Будь-яка сутність, як і будь-який об'єкт з реального світу, має низку властивостей. Наприклад, якщо сутність описує людини, то можливо виділити такі властивості, як ім'я, прізвище, зріст, вік, вага. Властивості необов'язково представляють прості дані типу int, а й можуть представляти більш комплексні структури даних. І у кожної сутності може бути одна або кілька властивостей, які будуть відрізняти цю сутність від інших і будуть унікально визначати цю сутність. Подібні властивості називають ключами.

При цьому суті можуть бути пов'язані асоціативним зв'язком «один-до-багатьох», «один-до-одного» і «багато-до-багатьох», подібно до того, як в реальній базі даних відбувається зв'язок через зовнішні ключі.

Відмінною рисою Entity Framework є використання запитів LINQ для вибірки даних з БД. За допомогою LINQ можливо не тільки отримувати певні рядки, що зберігають об'єкти, з БД, а й отримувати об'єкти, пов'язані різними асоціативними зв'язками.

Іншим ключовим поняттям є Entity Data Model. Ця модель зіставляє класи сутностей з реальними таблицями в БД.

Модель Entity Data Model складається з трьох рівнів: концептуального рівня, рівня сховища і рівня зіставлення (mapping).

На концептуальному рівні відбувається визначення класів сутностей, які використовуються в додатку.

Рівень сховища визначає таблиці, стовпці, відносини між таблицями і типи даних, з якими порівнюється використовувана база даних.

Рівень відображення (mapping) слугує посередником між попередніми двома, визначаючи зіставлення між властивостями класу суті і стовпцями таблиць.

Таким чином, можливо через класи, визначені у додатку, взаємодіяти з таблицями з бази даних.

Технологія Entity Framework передбачає три можливі способи взаємодії з базою даних:

– database first: Entity Framework створює набір класів, які відображають модель конкретної бази даних;

– мodel first: спочатку розробник створює модель бази даних, по якій потім Entity Framework створює реальну базу даних на сервері;

– сode first: розробник створює клас моделі даних, які будуть зберігатися в бд, а потім Entity Framework за цією моделлю генерує базу даних і її таблиці.

### 4.2.2 Web-модель ASP.NET MVC 5 та її переваги

Платформа ASP.NET MVC являє собою фреймворк для створення сайтів і веб-додатків за допомогою реалізації паттерна MVC.

Концепція паттерна (шаблону) MVC (model – view – controller) передбачає поділ додатка на три компоненти:

Контролер (controller) представляє клас, що забезпечує зв'язок між користувачем і системою, поданням і сховищем даних. Він отримує введені користувачем дані і обробляє їх. І в залежності від результатів обробки відправляє користувачеві певний результат, наприклад, у вигляді представлення (view).

Представлення (view) - це власне візуальна частина або призначений для користувача інтерфейс програми. Як правило, html-сторінка, яку користувач бачить, зайшовши на сайт.

Модель (model) представляє клас, що описує логіку використовуваних даних.

Загальну схему взаємодії цих компонентів можна представити таким чином (рисунок 4.2.2.1):

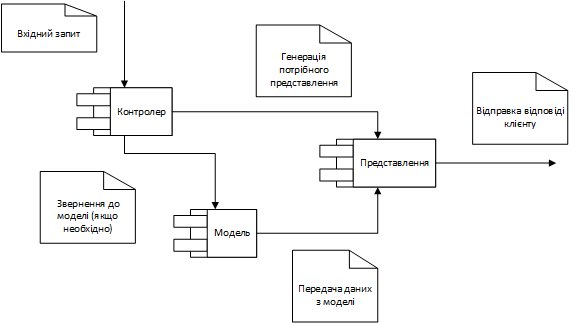


Рисунок 4.2.2.1 – Загальна схема взаємодії компонентів в ASP.NET MVC

У цій схемі модель є незалежним компонентом – будь-які зміни контролера або представлення не впливають на модель. Контролер і представлення є відносно незалежними компонентами, і нерідко їх можна змінювати окремо один від одного.

Завдяки цьому принципу побудови коду реалізується концепція розподілу відповідальності, в зв'язку з чим зручно побудувати роботу над окремими компонентами. Крім того, додаток має кращу можливість для написання та виконання модульних та інших тестів, використовуючи різні технології та інструменти. Наприклад, інтерфейс (frontend) тестується незалежно від контролера за допомогою засобів розробки frontend. Серверна частина покривається тестами, написаними на мові backend C# засобами NUnit, XUnit тощо.

Конкретні реалізації та визначення даного шаблону можуть відрізнятися, але в силу своєї гнучкості і простоти він став дуже популярним останнім часом, особливо в сфері веб-розробки.

Свою реалізацію шаблону представляє платформа ASP.NET MVC 5

Компоненти та функціональні можливості ASP.NET MVC 5, що застосовані в розробці програмного забезпечення дипломної роботи:

– технологія аутентифікації і авторизації. В ПЗ впроваджена система ASP.NET Identity, яка використовує компоненти OWIN і Katana;

– css-фреймворк Bootstrap використовується для створення адаптивного і розширюваного інтерфейсу веб-додатку.

### 4.2.3 Шаблони програмування

Шаблони програмування – сформовані загальноприйняті рішення для схожих між собою або типових завдань. Вони є необхідним інструментом для спрощення процесу програмування та проектування будь-якого програмного продукту.

Шаблон Unit of Work дозволяє спростити роботу з різними репозиторіями і дає впевненість, що все репозиторії використовуватимуть один і той же контекст даних.

При використанні шаблон Репозиторій для роботи з базою даних визначається інтерфейс сховища та створюються його окремі реалізації.

Клас UnitOfWork надає доступ до репозиторіїв через окремі властивості і визначає загальний контекст для обох репозиторіїв.

Крім того, даний клас містить додаткові методи Save() і Dispose(), які дозволяють забезпечити транзакційність операціям з БД, що виконуються через репозиторії, та своєчасно звільняти ресурси. Але так як цей функціонал буде загальним для обох репозиторіїв, то його краще винести в клас UnitOfWork.

Одним з найбільш часто використовуваних шаблонів при роботі з даними є патерн «Репозиторий». Репозиторій дозволяє абстрагуватися від конкретних підключень до джерел даних, з якими працює програма, і є проміжною ланкою між класами, що безпосередньо взаємодіють з даними, і решті програмою.

Наприклад, в системі будується сховище даних, що використовує СУБД MS SQL Server. Однак, якщо з’явиться потреба перемістити джерело даних з MS SQL на іншу СУБД – наприклад, MySQL або MongoDB, то буде достатньо легко виконати таке заміщення. Це стає можливим завдяки тому, що код верхніх ріднів/шарів програмної системи не опікується способом роботи з даними, які є специфічними для кожного типу сховища даних, а оперує об’єктами класів – абстрактними об’єктно-орієнтованими представниками сховищ даних Таким чином, репозиторій як такий представник додає програмі гнучкість при роботі з різними типами джерел даних.

## 4.3 Концептуальна модель бази даних

Реляційна база даних СУБД MS SQL програмного продукту складається з 9 відношень (таблиць), 3 з них – довідкові (рисунок 4.3.1). Вони включають в себе інформацію про значення показників відповідно до норм показників, перелічених у довідникових таблицях, а також шкалу значень для показників: запах при 20 і 60 градусів, присмак.

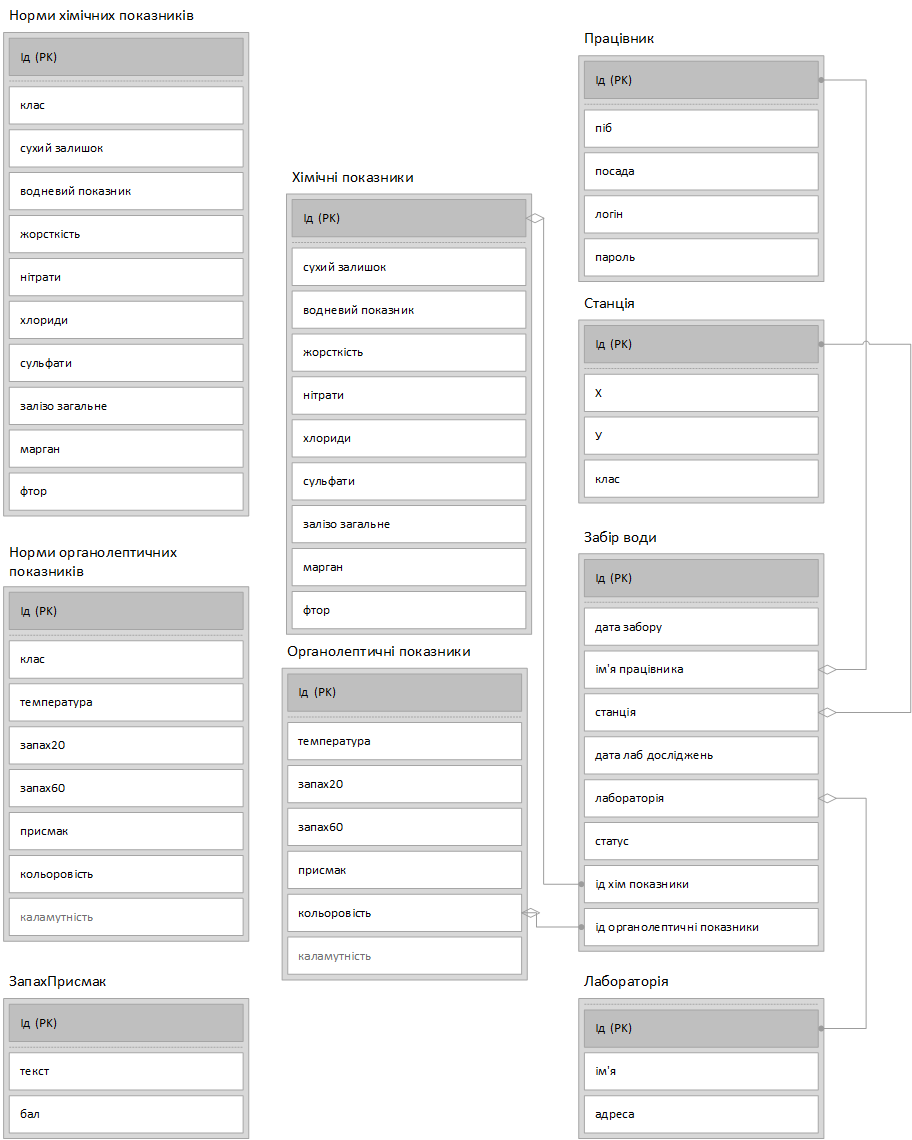


Рисунок 4.3.1 – Концептуальна схема бази даних

Відношення “Забір води” зберігає у собі загальну інформацію про кожний окремий забір води: дата забору, пункт забору води, працівника, що зразок отримав, дату лабораторного обстеження та лабораторію, де досліджувався зразок, статус зразка та поля, що повязують забір води з значеннями показників.

Відношення “Станція” зберігає інформацію про пункт забору води: координати та клас небезпеки.

Відношення “Працівник” зберігає загальну інформацію про працівника так як піб та посаду. А також зберігає дані, які використовуються в процесах аутентифікації та верифікації.

Відношення “Лабораторії” зберігає інформацію про назву та місцезнаходження лабораторії за адресою.

Відношення з показниками “Органолептичні показники”, “Хімічні показники” зберігають значення показників забору води: температура, вміст заліза, фтору, нітратів, сульфатів, запах, присмак тощо.

Відношення “ЗапахПрисмак” є довідковою таблицею. Відношення відображає шкалу запаху при різних значеннях температури і присмаку, зберігає бали та їхню розшифровку. Ця таблиця використовується таблицями “Органолептичні показники”, “Хімічні показники”, “Норми органолептичних показників”, ”Норми хімічних показників”.

Відношення “Норми органолептичних показників”, “Норми хімічних показників” – довідкові таблиці зі значеннями показників відповідно до класу небезпеки.

Зв’язки, що використовуються у відношеннях – «один-до-багатьох» та «один-до-одного».

## 4.4 Програмні модулі web-додатку

Для виконання перелічених функціональних вимог web-додатку було формалізовано предметну галузь, до якої відноситься розроблювальне програмне забезпечення, а саме визначено показники, за якими належить проводити вимірювання та оцінювання якості води. На основі отриманого формалізованого опису розроблено схему бази даних web-додатку.

Шар доступу до даних.

Шар роботи з даними, що отримуються з БД додатку, побудовано у вигляді динамічної бібліотеки. Вихідний код бібліотеки містить опис репозиторіїв, що інкапсулюють CRUD-операції з відповідними наборами сутностей Entity Framework:

DbSet<Chemical> Chemicals

DbSet<Laboratory> Laboratories

DbSet<OrganolepticNorm> OrganolepticNorms

DbSet<ChemicalNorm> ChemicalNorms

DbSet<Organoleptic> Organoleptics

DbSet<ScentFlavor> ScentFlavors

DbSet<Station> Stations

DbSet<WaterIntake> WaterIntakes

Репозиторії поєднуються об’єктом класу-контексту GroundwaterContext та відповідним йому UnitOfWork, щоб представити функціонал роботи з БД Groundwater.mdf.

Шар бізнес-логіки.

Шар, що описує процеси обробки даних домену, реалізований у вигляді динамічної бібліотеки. Класи та методи шару забезпечують:

– звертання до репозиторіїв шару доступу до даних з метою отримання та зміни даних, що знаходяться в БД. Ці дані необхідні для відображення користувачу web-додатку, введення нових даних забору та результатів лабораторних досліджень;

– обчислення критеріїв якості води за розробленим алгоритмом. Ці дані необхідні для відображення стану підземних вод у вигляді діаграм, графіків та мап;

– вибірка накопичених та обчислених даних за заданими критеріями та умовами з метою передачі їх шару представлення для відображення у графічному вигляді.

Шар представлення реалізовано у вигляді додатку asp.net MVC5, що є web-додатком, з яким безпосередньо взаємодіє користувач.

Додаток містить три контролери, які групують операції з користувачами та ролями та операції введення, обробки та представлення значень.

Операції з користувачами виконуються відповідно до технології аутентифікації та авторизації Identity і є стандартними.

Операції, що стосуються предметної галузі, можливо поділити на:

– Робота зі станціями забору води, лабораторіями аналізу проб - облік введених та створення нових пунктів;

– побудова діаграм статистичних розрахунків;

– побудова мап з занесенням на них інформації про стан вод на станціях забору.

Для побудови графіків на рівні шару представлення застосовувались сторонні js-бібліотеки, наприклад,

<script type="text/javascript"src="https://www.gstatic.com/charts/loader.js"></script>

Для швидкої побудови вмісту сторінок використовувалась бібліотека jQuery з функцією виконання json-запитів до методів контролеру $.getJSON.

Для побудови мап використовувався сервіс Google

<script type="text/javascript"src="https://www.google.com/jsapi"></script>

Для вбудови маркерів станцій забору було отримано ключ авторизації

<script async defer src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDcmm rxQrZRTzVcCU6y3JCto0f6C33vTM&callback=initMap"></script>

json-запитами отримано дані заборів води по станціях регіону, обраховано класи придатності води обраних станцій та відображено результати обчислень на Google -мапах. Маркери станцій розташовані на мапах з використанням техніки кластерізації. Для побудови кластерів на мапах використано бібліотеку:

<script src="https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/markerclusterer.js"></script>

# 5. Методика роботи користувача з програмною системою

Для забезпечення безвідмовної роботи програмної системи зі збору та обробки первинних даних стану підземних вод треба дотримуватися основних вимог рекомендацій щодо її використання.

## 5.1 Системні вимоги до використання

Для використання розробленої програмної системи персональний комп’ютер має мати встановлений веб-браузер та мати доступ до мережі інтернет. Особливого значення, який браузер буде встановлено у користувача, не має, але рекомендований – Mozilla Firefox версії не менш ніж 44.

## 5.2 Опис графічного інтерфейсу системи

Після переходу за посиланням користувач бачить головну сторінку (рисунок 5.2.1) на якій розміщене меню, коротка інформація про сайт, кнопки “Register” та “Log in”. При натисканні кнопки “Log in” виконується перехід на форму авторизації для підтвердження прав на виконання певних дій у системі (рисунок 5.2.2).

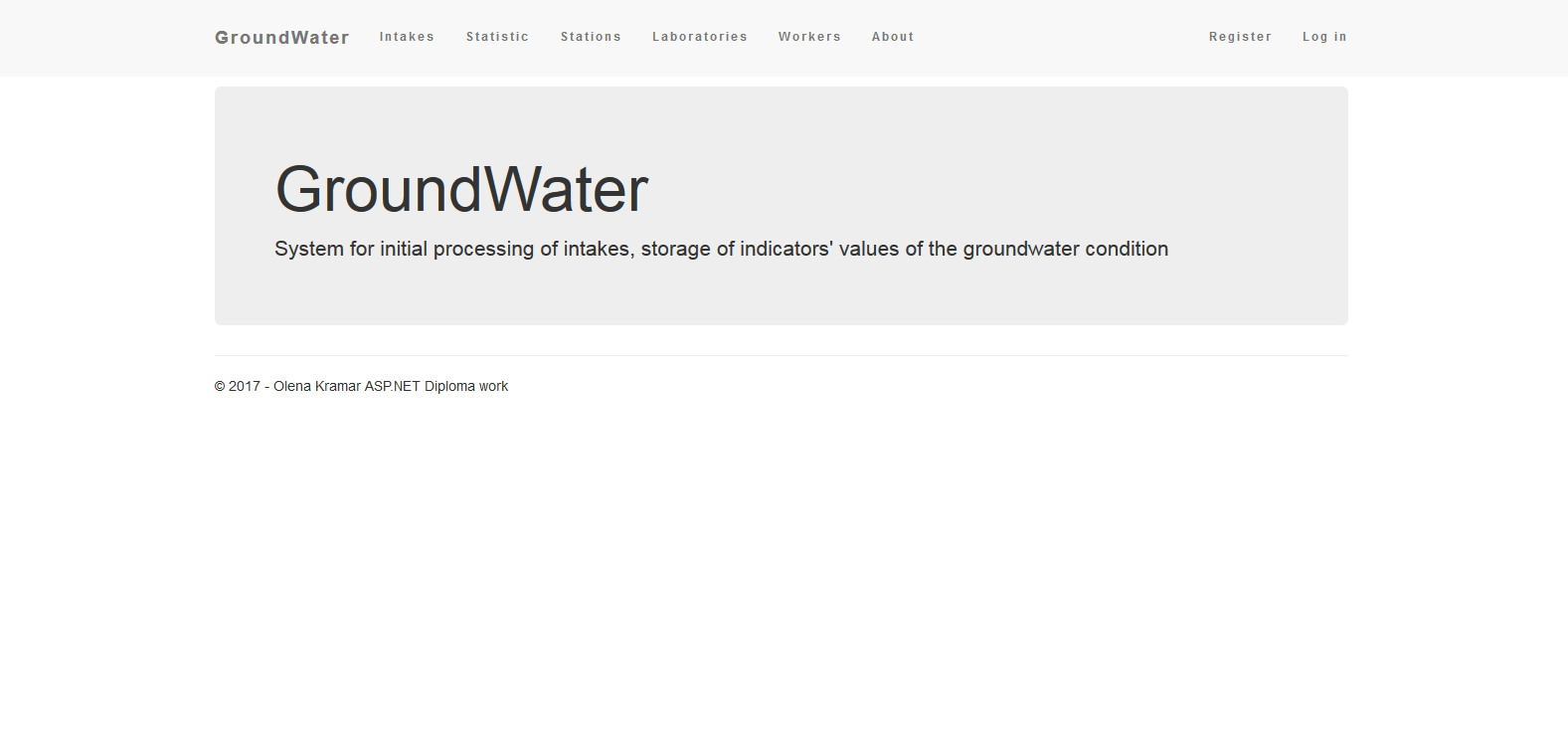


Рисунок 5.2.1 – Головна сторінка веб-додатку.

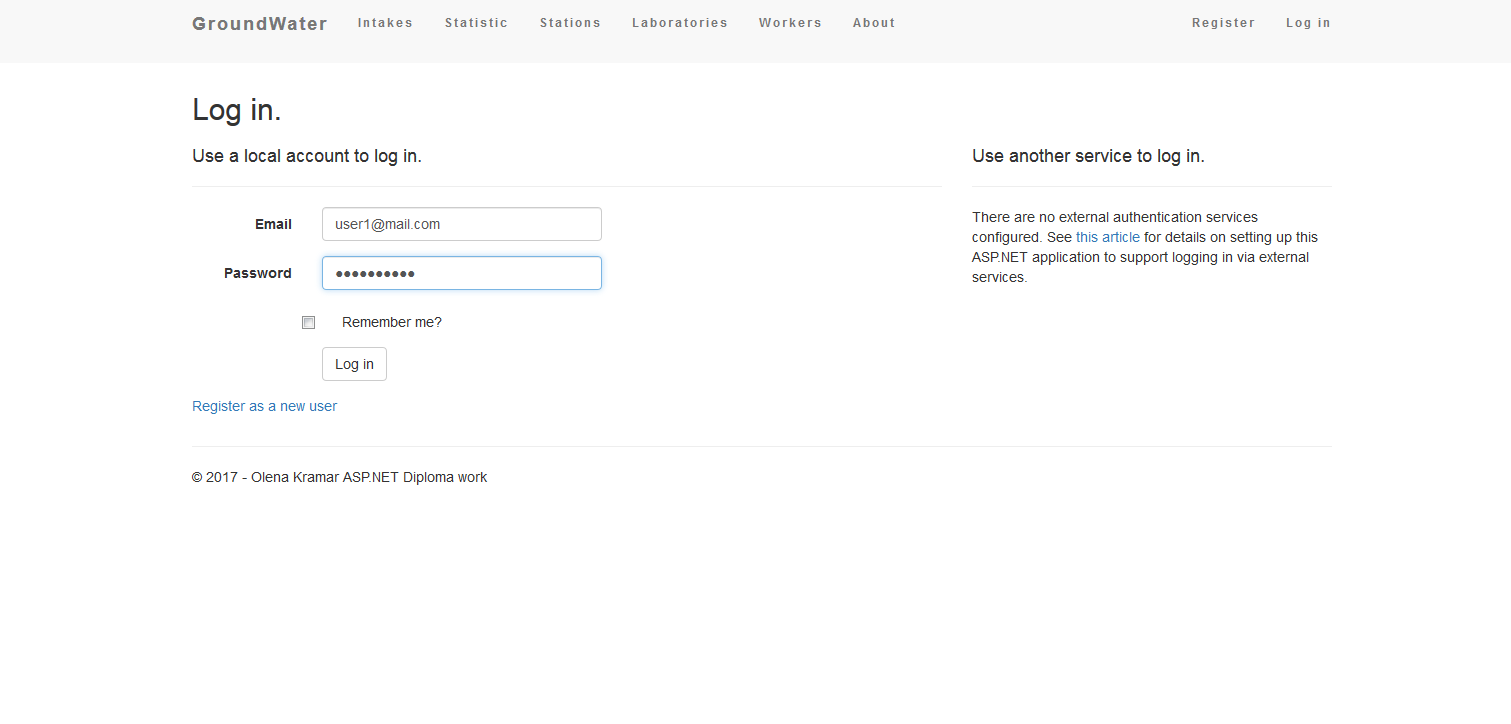


Рисунок 5.2.2 – Сторінка авторизації.

Після натискання на кнопку головного меню користувач бачить список усіх заборів води по усіх станціях та деяку інформацію про них (рисунок 5.2.3).

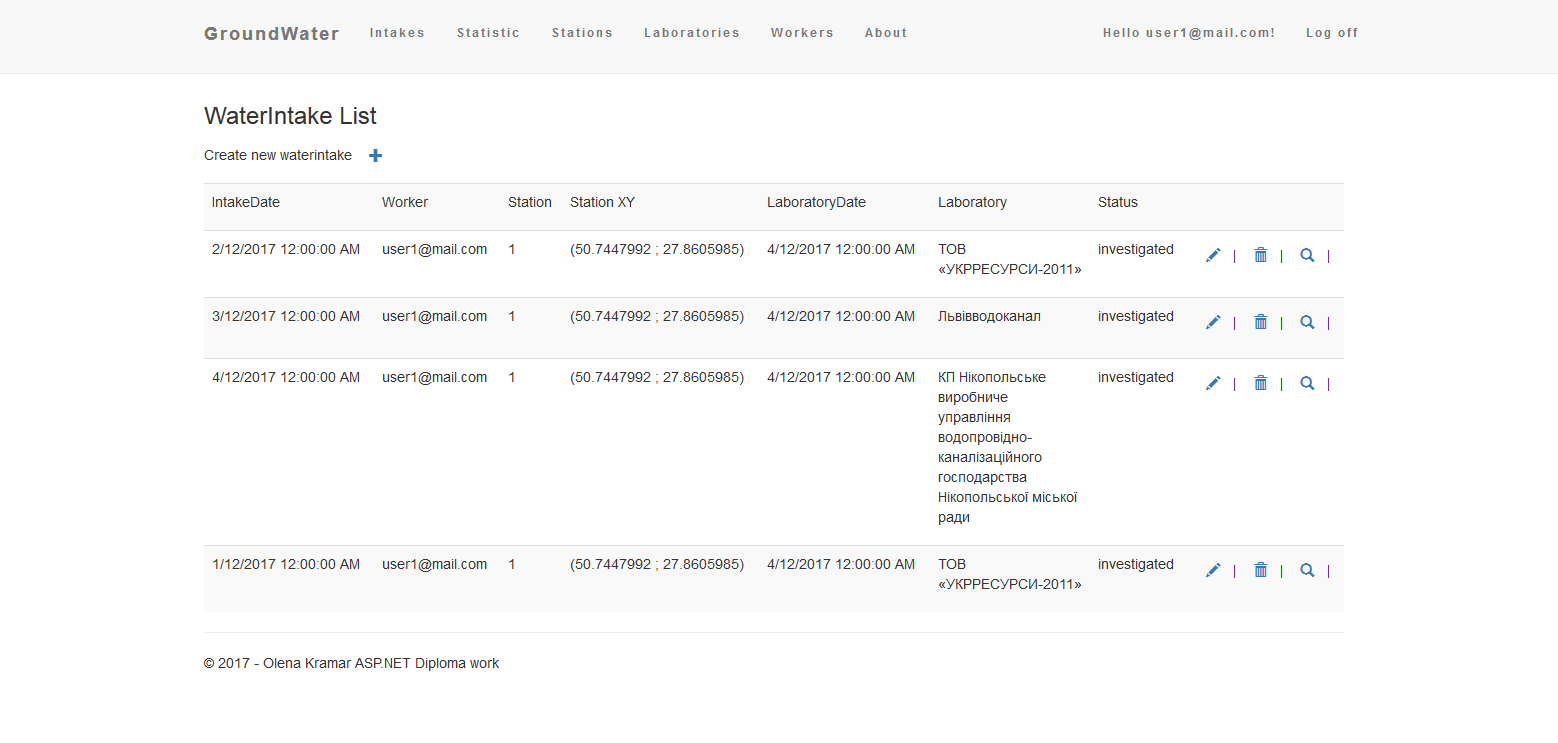


Рисунок 5.2.3 – Сторінка заборів води.

Якщо користувач хоче створити новий забір води, він має заповнити форму (рисунок 5.2.4). Щоб зменшити вірогідність введення некоректних даних або помилки користувача, система налаштована так, що:

- дата забору задається автоматично, її значення – значення дати в системі пристрою (але її можна корегувати);

- працівник, що забирав пробу зі станції, вводиться автоматично з профайлу користувача, і відповідне поле недоступне для редагування;

- поля для введення інформації про станцію та лабораторію –список з усіма існуючими записами з бази даних;

- статус зразка автоматично встановлюється на «забраний».

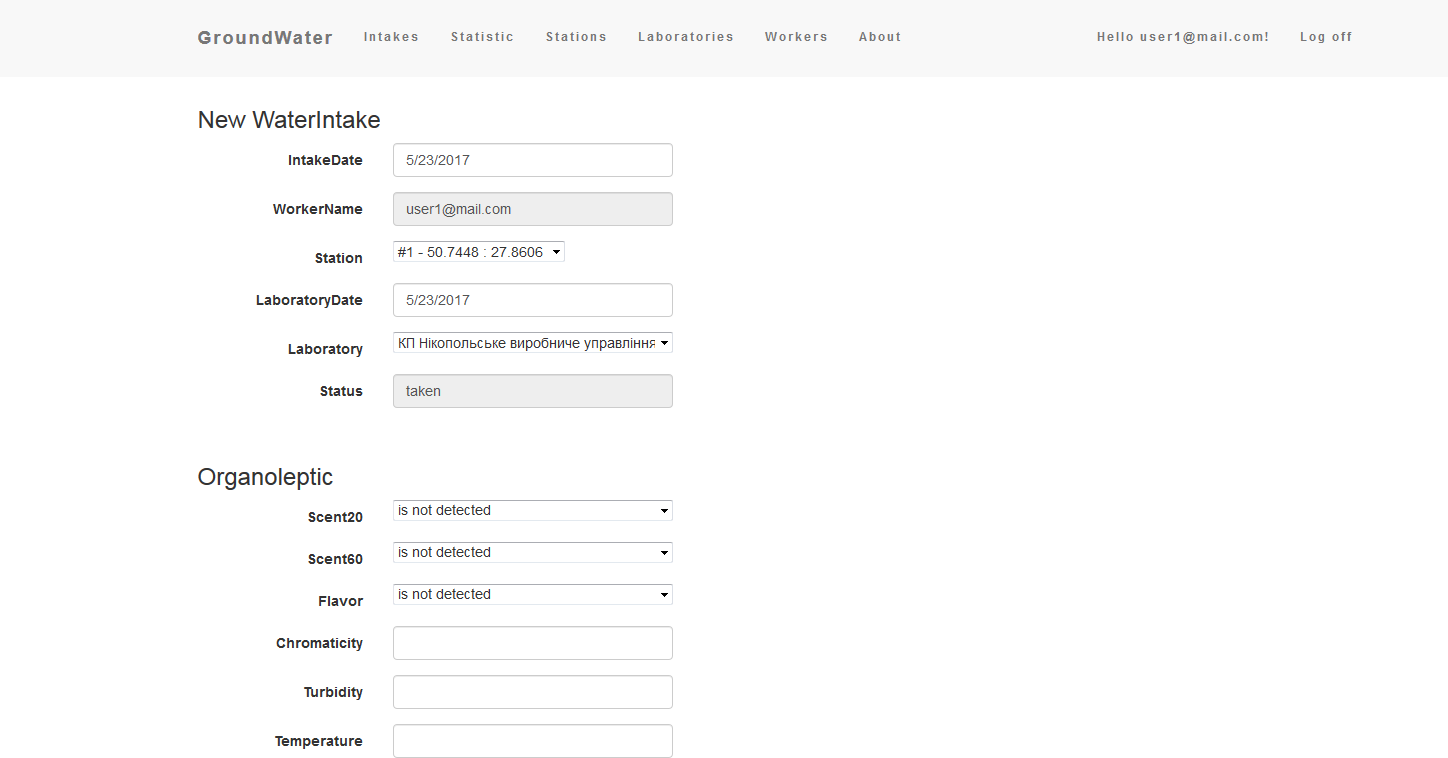


Рисунок 5.2.4 – Форма для додавання нового забору води.

У випадку успішної операції додавання відображується оновлений список заборів води.

Форма для редагування даних для забору води має подібний інтерфейс.

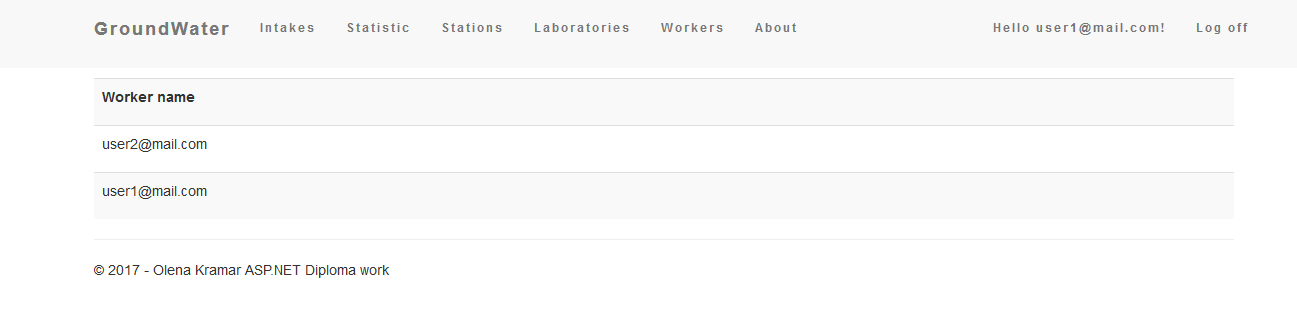
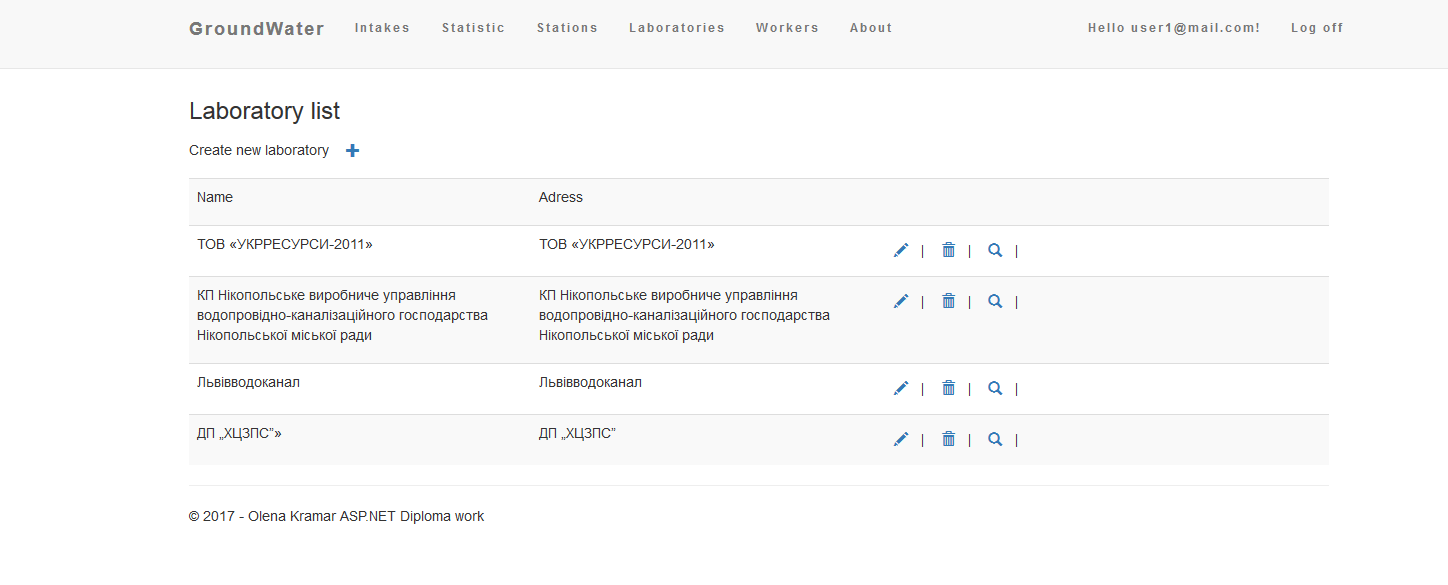
Таким чином можливо переглянути кількість і коротку інформацію про інші таблиці: працівників (рисунок 5.2.5), лабораторії (рисунок 5.2.6), станції (рисунок 5.2.7).

Рисунок 5.2.5 – Сторінка працівників.



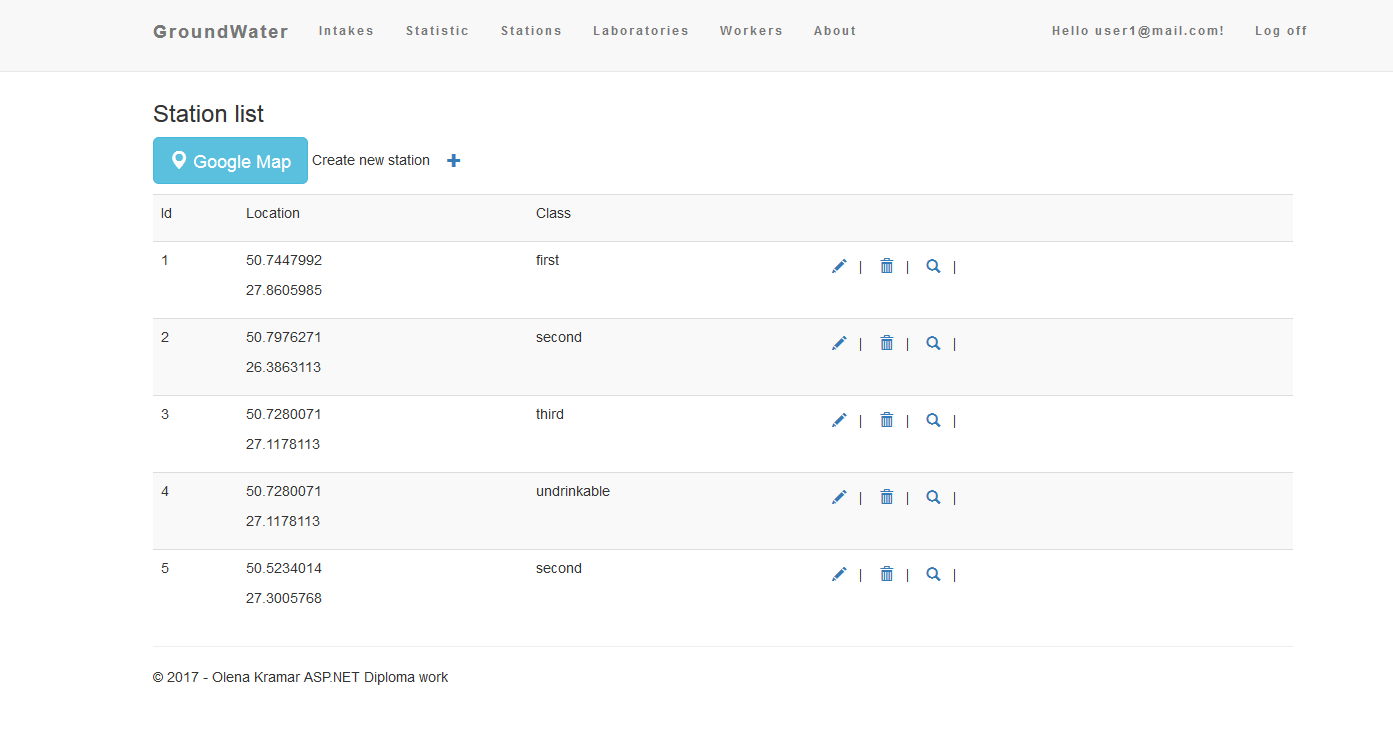
Рисунок 5.2.6 – Сторінка лабораторій для дослідження води.

Рисунок 5.2.7 – Сторінка станцій з забору води.

Також є можливість переглянути детальну інформацію про будь-який запис з таблиць у окремому вікні. Детальна інформація по запису станції забору води зображена на рисунку 5.2.8

Дана опція має схожий вигляд і для інших таблиць системи.

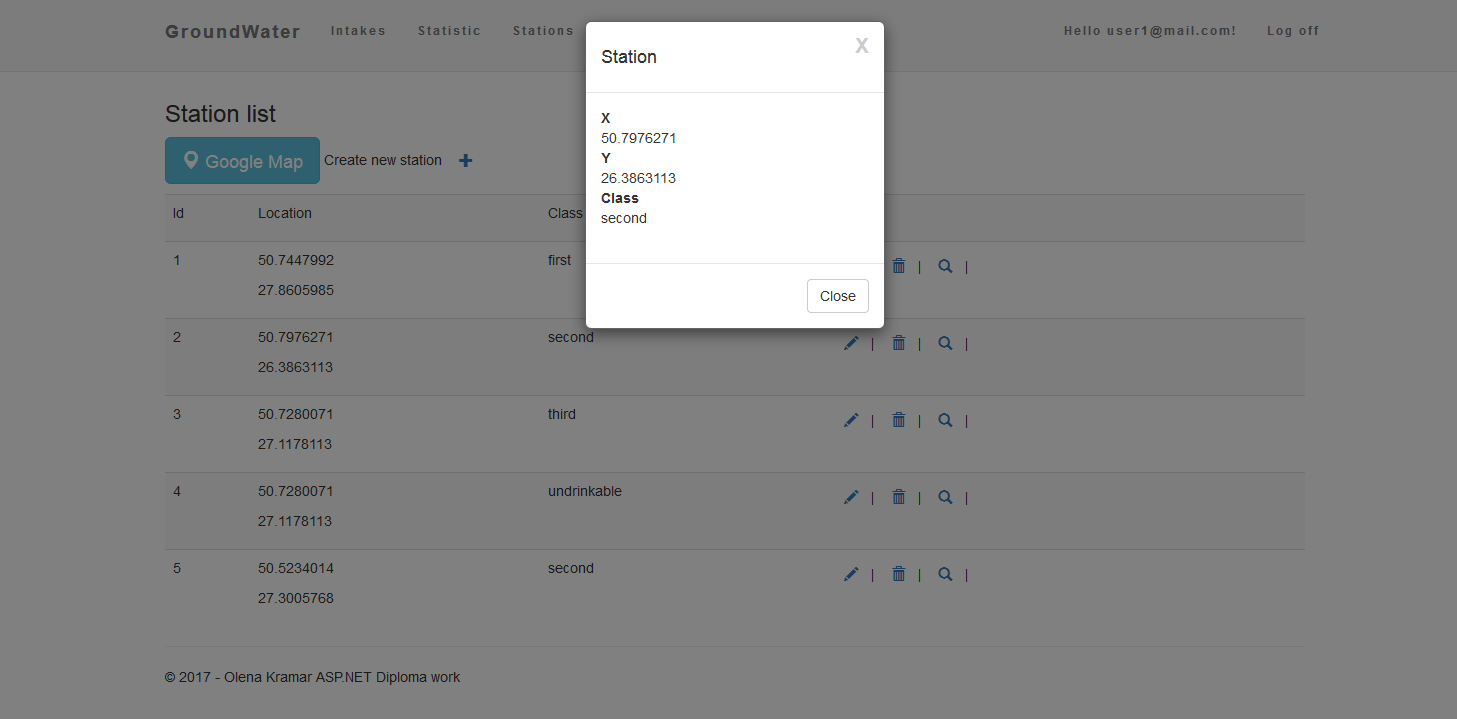


Рисунок 5.2.8–Вікно детальної інформації по запису станції забору води.

Якщо користувач не авторизується в системі, усі вище зазначені дії для нього недоступні. Такий користувач має можливість переглянути лише сторінку статистики (рисунок 5.2.9), маючи можливість дізнатись динаміку забрудненості конкретного джерела (рисунок 5.2.10) за певний період (рисунок 5.2.11) по будь-якому індикатору (рисунок 5.2.12). Результат відображається у вигляді ламаної на графіку (рисунок 5.2.13).

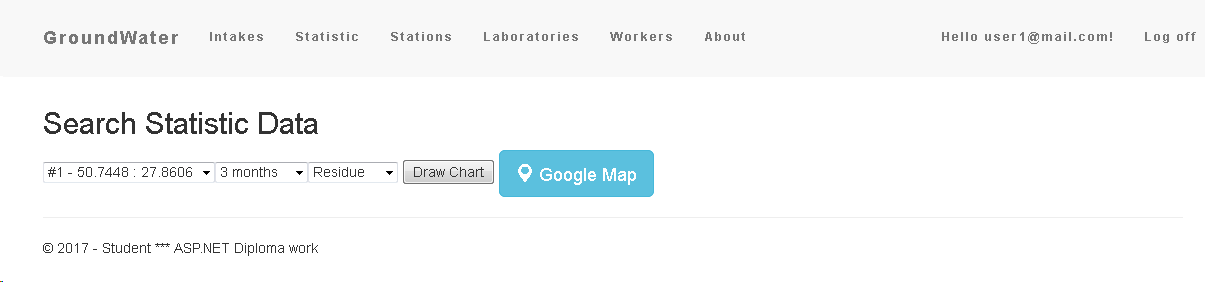


Рисунок 5.2.9 – Сторінка статистики.

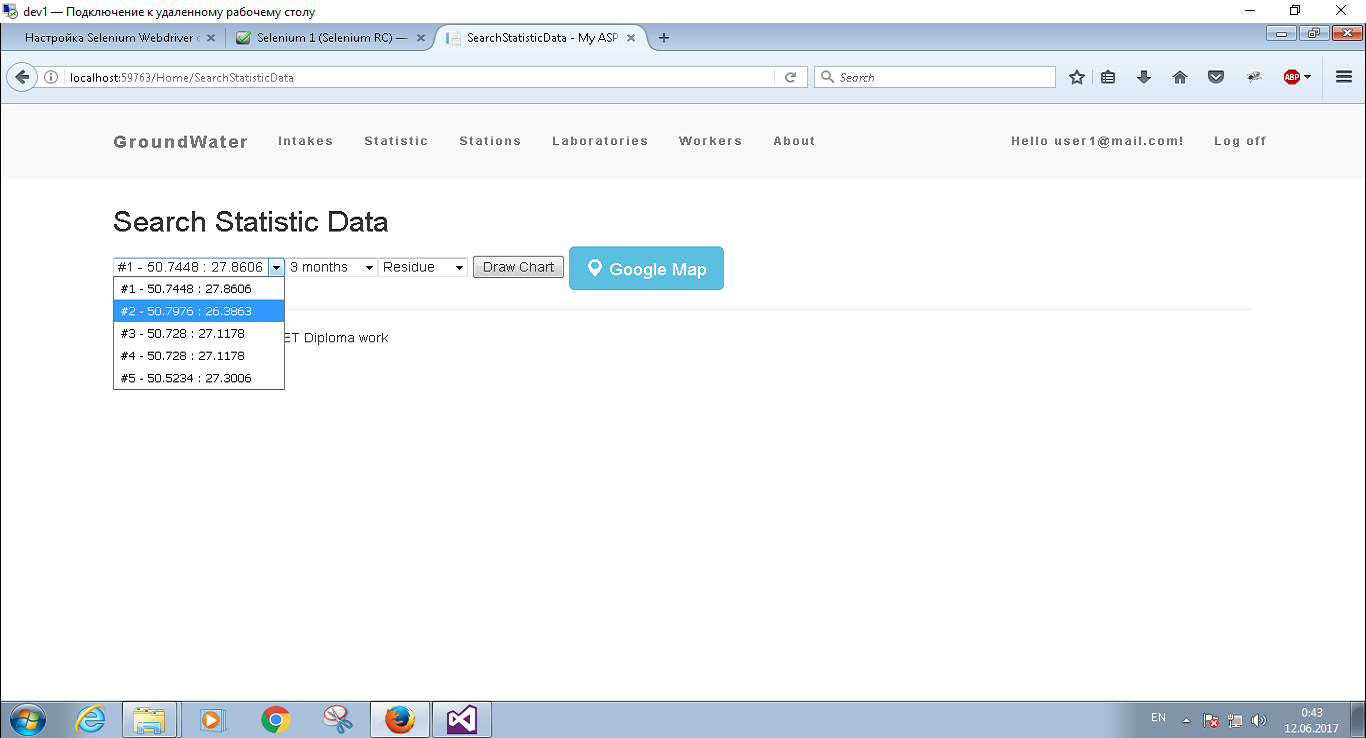


Рисунок 5.2.10 – Вибір станції забору води.

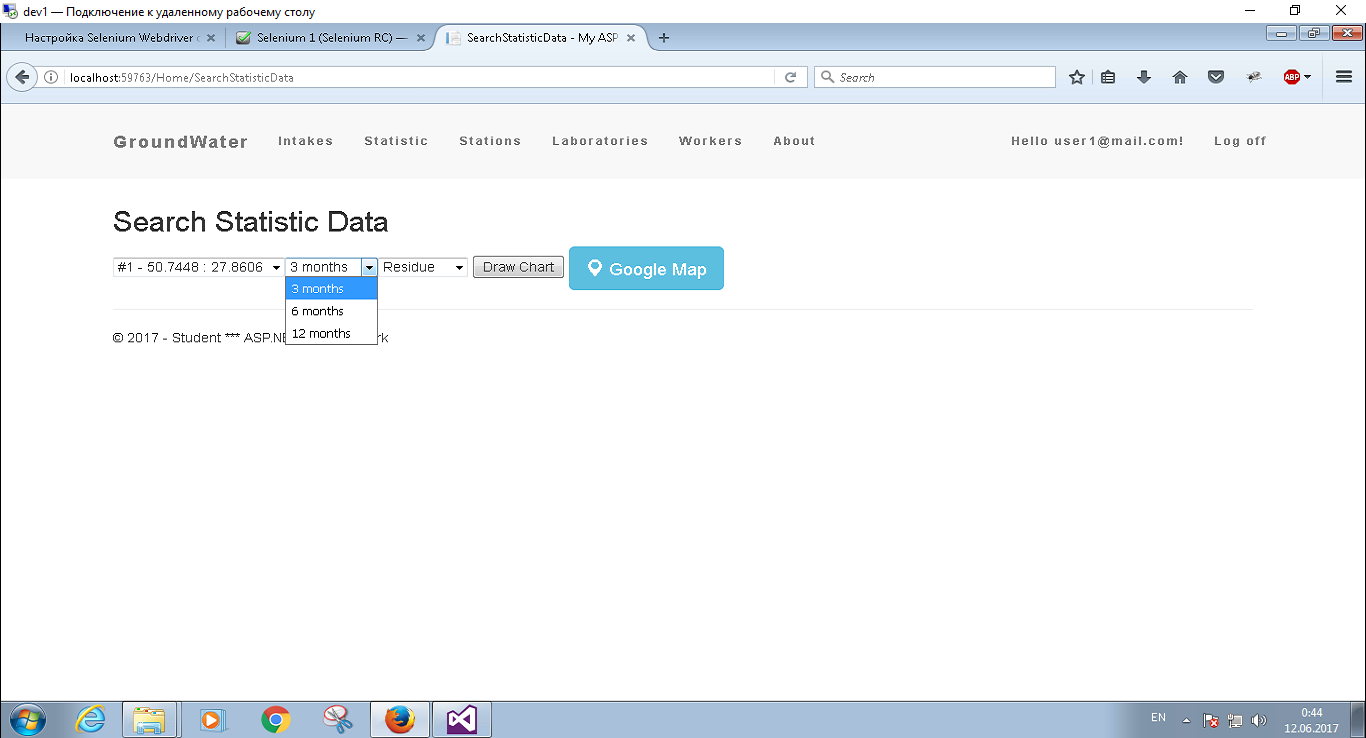


Рисунок 5.2.11 – Вибір періоду часу для відображення.

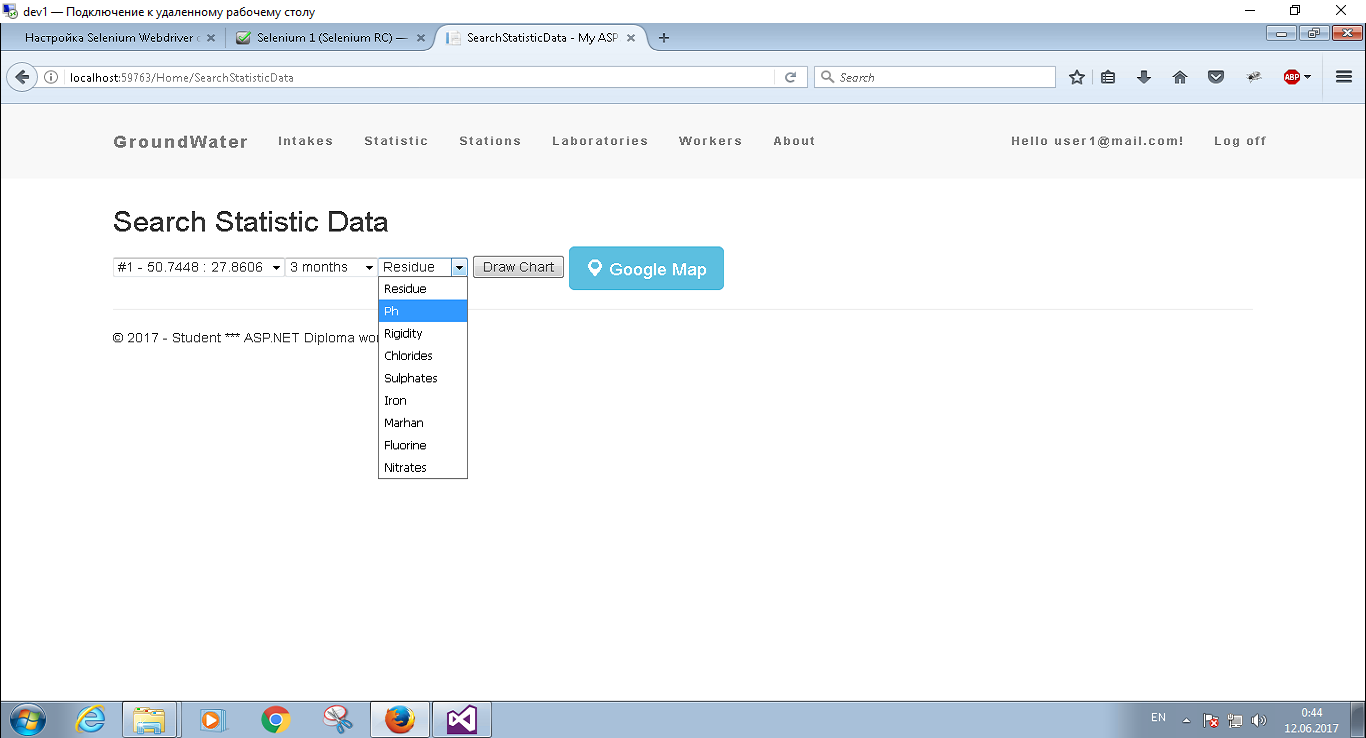


Рисунок 5.2.12 – Вибір індикатора для відображення.

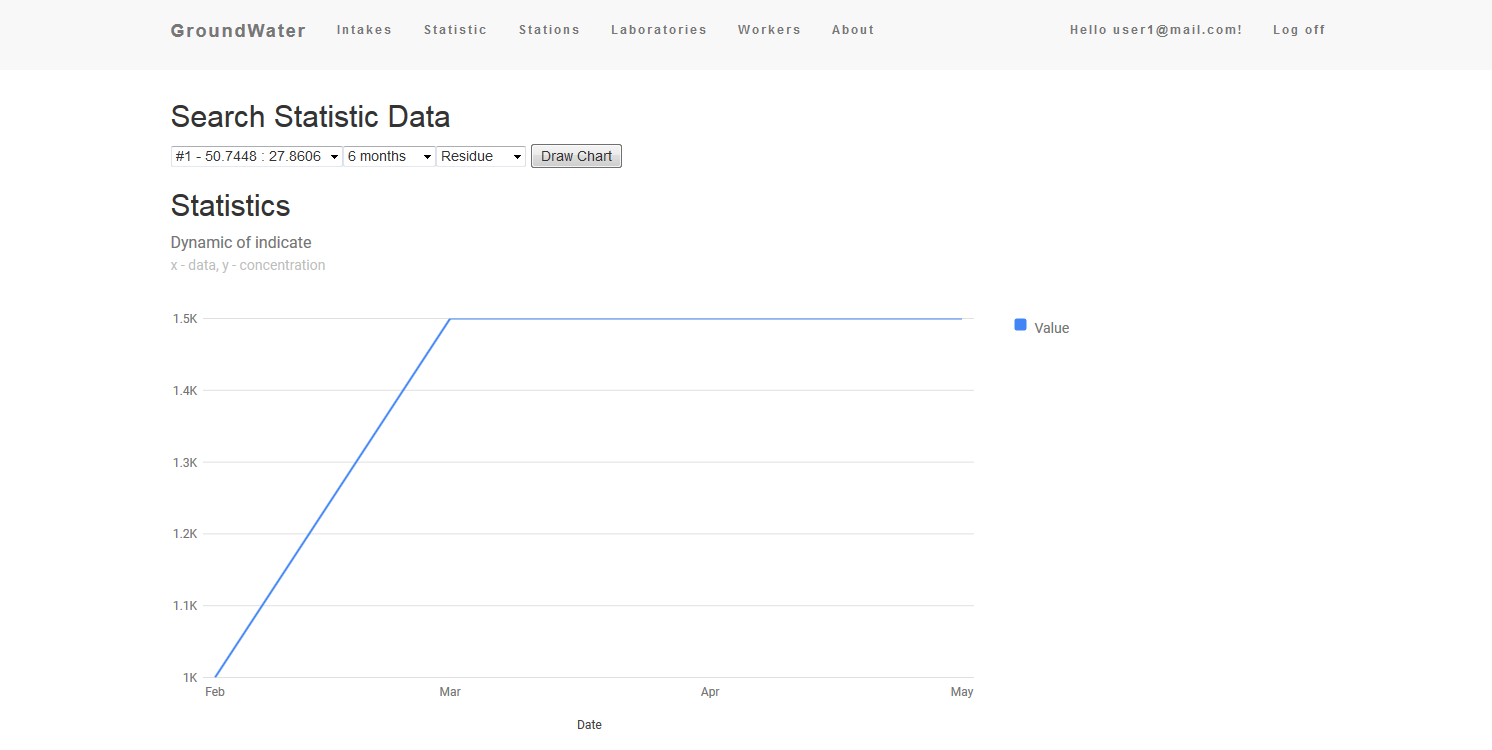


Рисунок 5.2.13 – Вигляд результатів обробки запиту.

Також користувач може переглянути карту (рисунок 5.2.14) з зображеними на ній у вигляді маркерів станцій забору води. Кожен маркер при натисканні на нього коливається, також над ним з'являється вікно з інформацією про дане джерело.

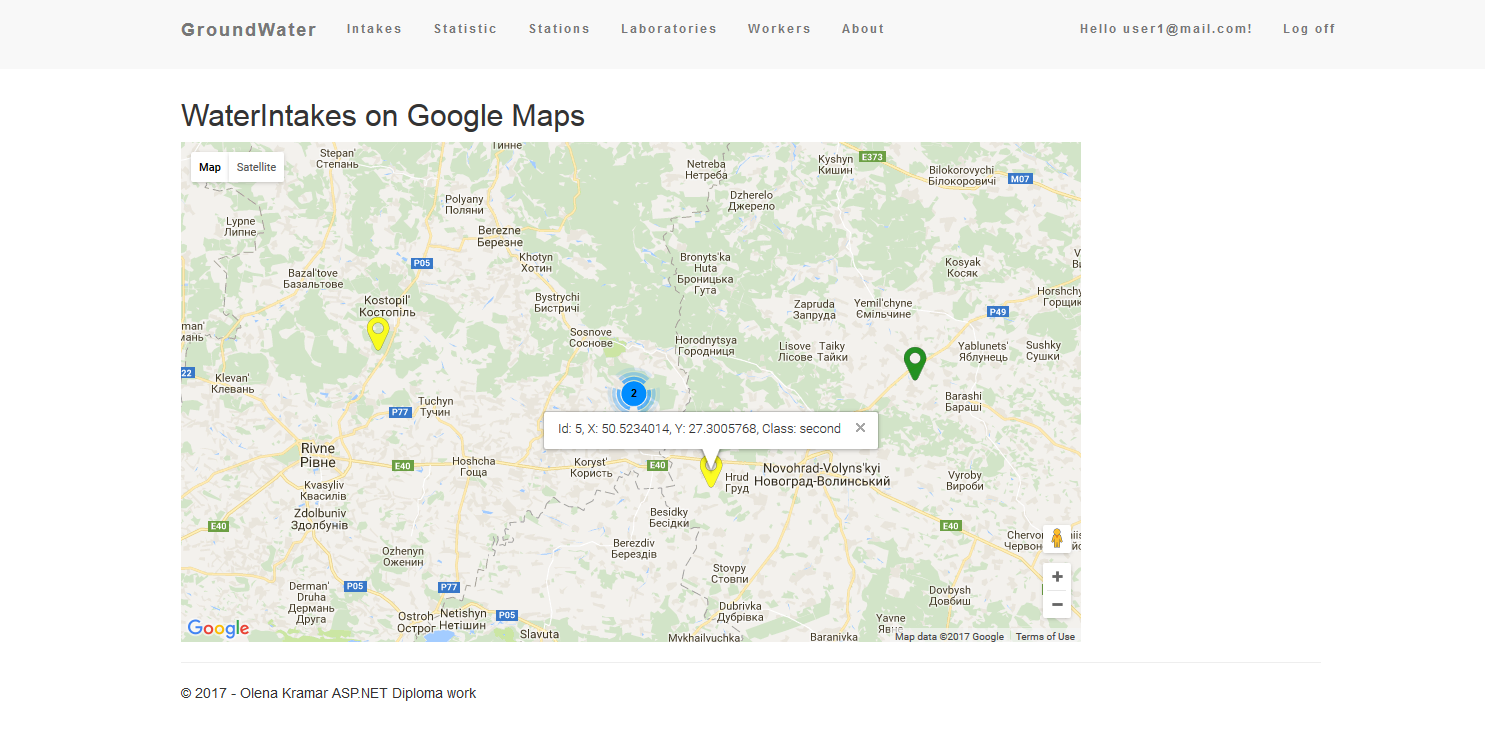
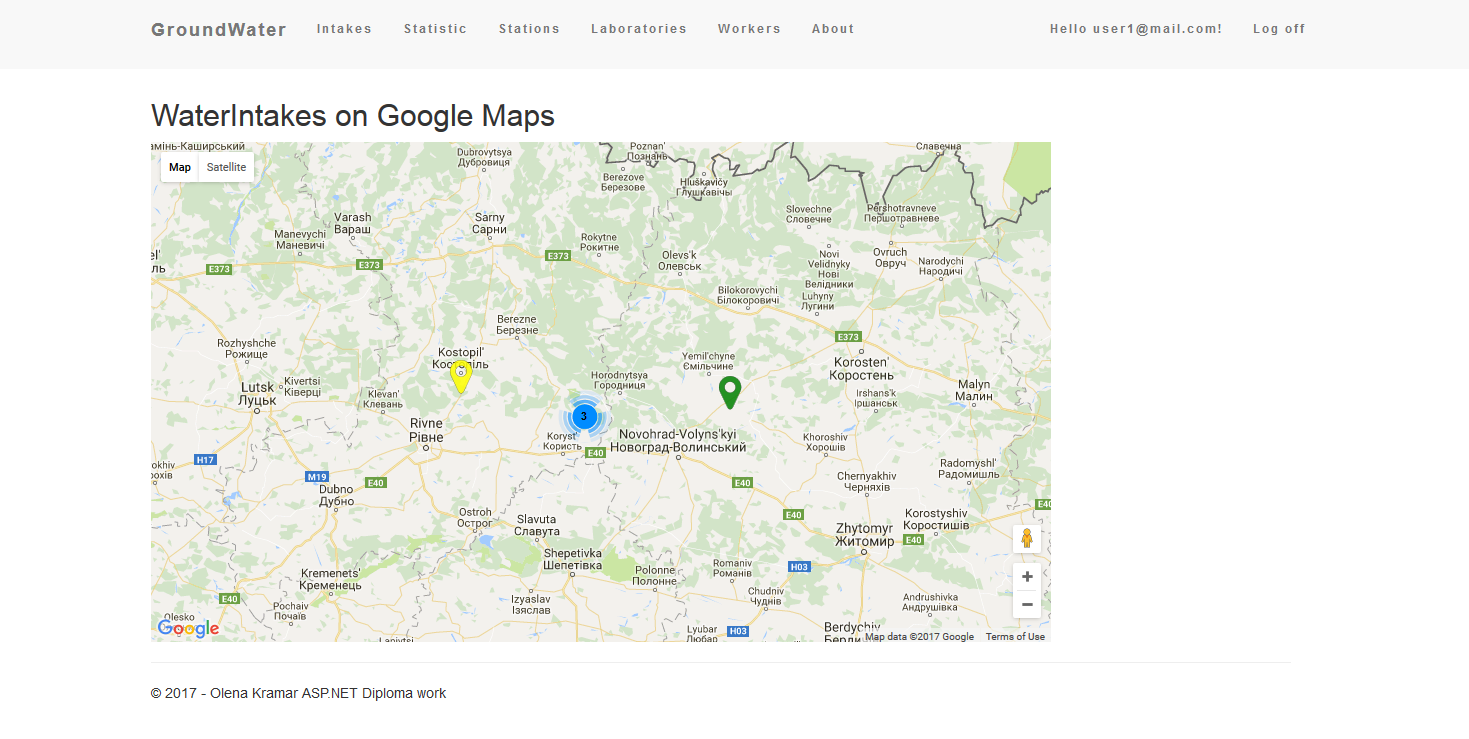


Рисунок 5.2.14 – Відображення маркерів на карті та вікна інформації.

При повторному натисканні анімація припиняється, вікно з інформацією зникає. Кожен маркер має певний колір. Він залежить від того, якого класу небезпеки це джерело. Якщо маркер зелений, то це джерело відноситься до першого класу, жовтий - до другого, червоний – до третього, чорний – джерело непридатне для використання.

В залежності від масштабу карти може з'являтися ще один вид елемента на карті – кластер. Кластер – об'єднання декількох маркерів в один елемент в даному випадку за відносною відстанню один від одного (рисунок 5.2.15).

Рисунок 5.2.15 – Кластеризація маркерів.

# ВИСНОВКИ

Проблема якості питної води, яка погіршується з року в рік через загострення екологічної обстановки, є на сьогоднішній день однією з головних. Вода є одним з найцінніших природних скарбів, адже без неї неможливе органічне життя на землі, їй належить одна з головних ролей у життєдіяльності людини. Напруженість сучасної екологічної обстановки вимагає невідкладного прийняття заходів в бік покращення якості питних вод.

Одним з першочергових кроків може стати впровадження загально державної системи збору первинних даних показників стану підземних вод в системі моніторингу водних ресурсів. Сучасна система, до якої централізовано вносяться дані, стане важливою базою для попередження, усунення екологічних проблем, дозволить контролювати викиди і виявити неоподатковані викиди. Система має інтуїтивний інтерфейс, тому не потребує особливих навичок від користувача під час роботи. Система не вимагає високих технічних характеристик пристроїв, тому може бути використана без перепон навіть на мобільних пристроях. Завдяки своїй архітектурі система дозволяє швидко і оперативно обробити велику кількість запитів від різних користувачів одночасно та не потребує особливих умов розташування – технічних станцій.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гігієна води та водопостачання населених пунктів. Санітарна охорона водних об’єктів. Харчування в профілактичній медицині. Гігієнічна оцінка харчового статусу. [Електронний ресурс] . — Режим доступу: http://intranet.tdmu.edu.ua/data/kafedra/internal/hihiena/classes\_stud/uk/med/lik/ptn.htm

2. ГОСТ 2761-84. [Електронний ресурс] . — Режим доступу: http://docs.cntd.ru/document/gost-2761-84

3. Моніторинг поверхневих вод та ґрунтів [Електронний ресурс] . — Режим доступу: http://vodaif.gov.ua/activity/monitirung.html

4. ГОСТ 3351-74 [Електронний ресурс] . — Режим доступу: http://standartgost.ru/g/pkey- 14294850608/%D0%93%D0%9E%D0%A1%D0%A2\_3351-74

5. Система стандартів з якості об'єктів природного середовища. [Електронний ресурс]. — Режим доступу: http://p-for.com/book\_416\_glava\_46\_2.5.3.\_Sistema\_standart%D1%96v\_z\_j.html

6. Методична вказівка [Електронний ресурс] . — Режим доступу: http://intranet.tdmu.edu.ua/data/kafedra/internal/hihiena/metod\_rozrobky/uk/med/lik/ntn.htm

7. Закон України "Про екологічну експертизу" // Відомості Верховної Ради України. – 1995. – № 8.

8. Закон України "Про пестициди і агрохімікати" // Постанова ВР України від 2.03.1995 р. № 87/95-ВР. – 1995.

9. Запольський, А.К. Основи екології / А.К. Запольський, А.І. Салюк – К.: Вища школа – 2004.

10. Зеркалов, В. Екологічна безпека: управління, моніторинг, контроль Посібник. – К.: КНТ, Дакор, Основа, 2007.

11. Методичні рекомендації з встановлення допустимих концентрацій шкідливих речовин в агрохімікатах / [За ред. д. с./г. Наук Н.А. Макаренко] –К., 2007.

12. Набивач, В.М. Основи екологічного нормування і промислової токсикології. / В.М. Набивач, М.П. Сухий. – Д.: УДУХТу,2002.

13. Сафранов, Т.А. Екологічні основи природокористування : навч. посібник. / Т.А. Сафранов. – Львів: Новий світ – 2003.

14. Джигирей, B.C. Основи екології та охорона навколишнього природного середовища / В.С. Джигирей, В.М. Сторожук, Р.А. Яшок – Львів: Афіша, 2004.

15. Алексеев, В.В. Методы и средства контроля загрязнений окружающей среды. / В.В. Алексеев, Б.Г. Комаров, П.Г. Королев. – Сан.-Пет., 2001.– 56 с.

16. Величко, О.А. Контроль забруднення довкілля / О.А. Величко, Д.В. Зеркалов. – К.: Основи, 2002.

17. Шевчук, В.Я. Екологічне управління. / В.Я. Шевчук. – К.: Либідь, 2004.

18. Шелудченко, Б.А. Нормування якості навколишнього середовища. / Б.А. Шелудченко. – Камянець-Под., 2007.

Додаток 1

Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

Специфікація

УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 2

2017

|  |  |  |
| --- | --- | --- |
| Позначення | Найменування | Примітки |
| **Документація** | | |
| УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1 | Записка.docx | Пояснювальна записка |
| **Компоненти** | | |
| УКР.НТУУ“КПI”. ТМ31128\_17Б 12-2 | DAL.dll | Модуль доступу до бази даних |
| УКР.НТУУ“КПI”. ТМ31128\_17Б 12-3 | BLL.dll | Модуль бізнес логіки |
| УКР.НТУУ“КПI”. ТМ31128\_17Б 12-4 | WebApp.csproj | Модуль графічного інтерфейсу |

Додаток 2

Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

Код програмних модулів

УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 8

2017

GroundwaterContext.cs

public class GroundwaterContext : DbContext

{

public DbSet<Chemical> Chemicals { get; set; }

public DbSet<Laboratory> Laboratories { get; set; }

public DbSet<OrganolepticNorm> OrganolepticNorms { get; set; }

public DbSet<ChemicalNorm> ChemicalNorms { get; set; }

public DbSet<Organoleptic> Organoleptics { get; set; }

public DbSet<ScentFlavor> ScentFlavors { get; set; }

public DbSet<Station> Stations { get; set; }

public DbSet<WaterIntake> WaterIntakes { get; set; }

public DbSet<Worker> Workers { get; set; }

static GroundwaterContext()

{

Database.SetInitializer<GroundwaterContext>(new GroundwaterInitializer());

}

public GroundwaterContext(string connectionString) : base(connectionString)

{ }

}

UnitOfWork.cs

public class UnitOfWork : IUnitOfWork

{

GroundwaterContext db;

ChemicalRepository chemicalrepository;

ChemicalNormRepository chemicalnormrepository;

LaboratoryRepository laboratoryrepository;

OrganolepticNormRepository organolepticnormrepository;

OrganolepticRepository organolepticrepository;

ScentFlavorRepository scentflavorrepository;

StationRepository stationrepository;

WaterIntakeRepository waterintakerepository;

WorkerRepository workerrepository;

public UnitOfWork(string connectionString)

{

Database.SetInitializer(new GroundwaterInitializer());

db = new GroundwaterContext(connectionString);

}

public IRepository<Chemical> Chemicals

{

get

{

if (chemicalrepository == null)

chemicalrepository = new ChemicalRepository(db);

return chemicalrepository;

}

}

public IRepository<Laboratory> Laboratories

{

get

{

if (laboratoryrepository == null)

laboratoryrepository = new LaboratoryRepository(db);

return laboratoryrepository;

}

}

public IRepository<OrganolepticNorm> OrganolepticNorms

{

get

{

if (organolepticnormrepository == null)

organolepticnormrepository = new OrganolepticNormRepository(db);

return organolepticnormrepository;

}

}

public IRepository<ChemicalNorm> ChemicalNorms

{

get

{

if (chemicalnormrepository == null)

chemicalnormrepository = new ChemicalNormRepository(db);

return chemicalnormrepository;

}

}

public IRepository<Organoleptic> Organoleptics

{

get

{

if (organolepticrepository == null)

organolepticrepository = new OrganolepticRepository(db);

return organolepticrepository;

}

}

public IRepository<ScentFlavor> ScentFlavors

{

get

{

if (scentflavorrepository == null)

scentflavorrepository = new ScentFlavorRepository(db);

return scentflavorrepository;

}

}

public IRepository<Station> Stations

{

get

{

if (stationrepository == null)

stationrepository = new StationRepository(db);

return stationrepository;

}

}

public IRepository<WaterIntake> WaterIntakes

{

get

{

if (waterintakerepository == null)

waterintakerepository = new WaterIntakeRepository(db);

return waterintakerepository;

}

}

public IRepository<Worker> Workers

{

get

{

if (workerrepository == null)

workerrepository = new WorkerRepository(db);

return workerrepository;

}

}

public void Save()

{

db.SaveChanges();

}

private bool disposed = false;

public virtual void Dispose(bool disposing)

{

if (!this.disposed)

{

if (disposing)

{

db.Dispose();

}

}

this.disposed = true;

}

public void Dispose()

{

Dispose(true);

GC.SuppressFinalize(this);

}

}

GroundWaterService.cs

public class GroundwaterService

{

private IUnitOfWork uow { get; set; }

public GroundwaterService(string connectionString)

{

try

{

uow = new UnitOfWork(connectionString);

}

catch (Exception e)

{

Console.WriteLine((e.Message));

}

AutoMapperConfig.RegisterMappings();

}

public int CreateWorker(WorkerDTO workerDTO)

{

Worker ac;

ac = Mapper.Map<WorkerDTO, Worker>(workerDTO);

uow.Workers.Create(ac);

uow.Save();

int id = uow.Workers.Find(obj => ((workerDTO.Name == obj.Name) && (workerDTO.Position == obj.Position))).First().Id;

return id;

}

public IEnumerable<WorkerDTO> GetWorkers()

{

return Mapper.Map<IEnumerable<Worker>, IEnumerable<WorkerDTO>>(uow.Workers.GetAll());

}

public IEnumerable<LaboratoryDTO> GetLaboratories()

{

return Mapper.Map<IEnumerable<Laboratory>, IEnumerable<LaboratoryDTO>>(uow.Laboratories.GetAll());

}

}

HomeController.cs

private class MapMarkerItem

{

public double X { get; set; }

public double Y { get; set; }

public string Color { get; set; }

public string Text { get; set; }

}

public JsonResult GetMap()

{

var allStations = db.Stations;

var result = new List<MapMarkerItem>();

foreach (var st in allStations)

{

string color = "black";

if (st.Class == DAL.Class.first)

color = "green";

else

if (st.Class == DAL.Class.second)

color = "yellow";

else

if (st.Class == DAL.Class.third)

color = "red";

result.Add(new MapMarkerItem { X = st.X, Y = st.Y, Color = color, Text = string.Format("Id: {0}, X: {1}, Y: {2}, Class: {3}", st.Id, st.X, st.Y, st.Class) });

}

return Json(new { Values = result }, JsonRequestBehavior.AllowGet);

}

#region WaterIntakes

[Authorize]

public ActionResult WaterIntakes()

{

ViewBag.Title = "WaterIntakes";

return View(db.WaterIntakes.Include(p => p.Laboratory).Include(r => r.Station));

}

[HttpGet]

[Authorize]

public ActionResult CreateWaterIntake()

{

WaterIntakeViewBag();

ViewBag.IntakeDate = DateTime.Now;

ViewBag.LaboratoryDate = DateTime.Now;

return View();

}

private void WaterIntakeViewBag()

{

ViewBag.WorkerName = User.Identity.Name;

var laboratoriesList = db.Laboratories.Select(l => new { LaboratoryId = l.Id, Name = l.Name }).ToList();

SelectList laboratories = new SelectList(laboratoriesList, "LaboratoryId", "Name");

ViewBag.Laboratories = laboratories;

var stationsList = db.Stations.Select(l => new { StationId = l.Id, XY = "#" + l.Id + " - " + l.X + " : " + l.Y }).ToList(); //String.Format("({0}, {1})", l.X, l.Y)

SelectList stations = new SelectList(stationsList, "StationId", "XY");

ViewBag.Stations = stations;

var scentFlavorList = db.ScentFlavors.Select(l => new { Point = l.Point, Text = l.Text }).ToList();

SelectList scentFlavors = new SelectList(scentFlavorList, "Point", "Text");

ViewBag.ScentFlavors = scentFlavors;

ViewBag.Status = DAL.Status.taken;

}

[HttpPost]

public ActionResult CreateWaterIntake(WaterIntakeModel wtmodel)

{

Laboratory b = db.Laboratories.Find(wtmodel.LaboratoryId);

Station s = db.Stations.Find(wtmodel.StationId);

if (b == null)

{

throw new OperationCanceledException("Laboratory not found");

}

if (s == null)

{

throw new OperationCanceledException("Station not found");

}

WaterIntake wt = new WaterIntake()

{

Id = 100,

LaboratoryId = wtmodel.LaboratoryId,

Laboratory = b,

IntakeDate = wtmodel.IntakeDate,

LaboratoryDate = wtmodel.LaboratoryDate,

WorkerName = wtmodel.WorkerName,

StationId = wtmodel.StationId,

Station = s,

Status = DAL.Status.taken

};

Chemical ch = new Chemical()

{

Id = 100,

Residue = wtmodel.Residue,

Ph = wtmodel.Ph,

Rigidity = wtmodel.Rigidity,

Chlorides = wtmodel.Chlorides,

Sulphates = wtmodel.Sulphates,

Iron = wtmodel.Iron,

Marhan = wtmodel.Marhan,

Fluorine = wtmodel.Fluorine,

Nitrates = wtmodel.Nitrates

};

Organoleptic or = new Organoleptic()

{

Id = 100,

Scent20 = wtmodel.Scent20,

Scent60 = wtmodel.Scent60,

Flavor = wtmodel.Flavor,

Chromaticity = wtmodel.Chromaticity,

Turbidity = wtmodel.Turbidity,

Temperature = wtmodel.Temperature

};

if (ModelState.IsValid)

{

if (wtmodel.LaboratoryDate == null)

{

wt.Status = DAL.Status.taken;

}

else

{

wt.Status = DAL.Status.investigated;

}

s.Class = CheckingClass(ch, or);

db.WaterIntakes.Add(wt);

db.Chemicals.Add(ch);

db.Organoleptics.Add(or);

db.SaveChanges();

return RedirectToAction("WaterIntakes");

}

else

{

WaterIntakeViewBag();

if (wtmodel.IntakeDate == default(DateTime))

ViewBag.IntakeDate = DateTime.Now;

else

ViewBag.IntakeDate = wtmodel.IntakeDate;

if (wtmodel.LaboratoryDate == default(DateTime))

ViewBag.LaboratoryDate = DateTime.Now;

else

ViewBag.LaboratoryDate = wtmodel.LaboratoryDate;

return View(wtmodel);

}

}

Statistic.cshtml

@using WebApp.Models

@model StatisticModel

@\*<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>\*@

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<script type="text/javascript" src="https://www.google.com/jsapi"></script>

<h2>Statistics</h2>

@\*<p>@Model.Id</p>

<p>@Model.Indicator</p>

<p>@Model.MounthAmount</p>\*@

<div id="linechart" style="width: 900px; height: 500px"></div>

<script>

google.charts.load('current', { 'packages': ['line'] });

google.charts.setOnLoadCallback(drawChart);

// X - date, Y - integer values. To get mvc data use controller method GetLineChartStatistic

function drawChart() {

//EX: You called the draw() method with the wrong type of data rather than a DataTable or DataView

//mvc data

var data = new google.visualization.DataTable();

data.addColumn('datetime', 'Date');

data.addColumn('number', 'Value');

var options = {

chart: {

title: 'Dynamic of indicate',

subtitle: 'x - data, y - concentration'

},

width: 900,

height: 500

};

var path = '@Html.Raw(Url.Action("GetLineChartStatistic", "Home", new { station = Model.Id, mounthAmount = Model.MounthAmount, indicator = Model.Indicator }))';

$.getJSON(path, function (result) {

if (result.Values.length) {

$.each(result.Values, function (index, c) {

data.addRow(

[new Date(c.Year, c.Month), c.Value]

);

});

};

});

var chart = new google.charts.Line(document.getElementById('linechart'));

// pass dataLocal or data to get local data source or mvc data source

chart.draw(data, google.charts.Line.convertOptions(options));

}

//});

</script>

Додаток 3

Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

Опис програмних модулів

УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 2

2017

**АНОТАЦІЯ**

Розроблені програмні модулі призначений для доступу до бази даних, обробки запитів, необхідних користувачу, та візуалізації даних.

Підключення до бази даних здійснюється під час обробки запиту за допомогою технології Entity Framework.

Мова програмної реалізації — C#. Середовище розробки — Microsoft Visual Studio 2017.

**ANNOTATION**

The software modules were designed to access a database, processing requests that users require, and data visualization.

Connection to the database is performed during processing user`s request using technology Entity Framework.

Language software implementation - C #. Development environment - Microsoft Visual Studio 2017.

**ЗМІСТ**

1. ЗАГАЛЬНІ ВІДОМОСТІ 4

2. ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ 6

3. ОПИС ЛОГІЧНОЇ СТРУКТУРИ 7

4. ВХІДНІ ДАНІ ТА РЕЗУЛЬТАТ 8

**1. ЗАГАЛЬНІ ВІДОМОСТІ**

Програмний продукт складається з трьох модулів:

— DAL.dll;

— BLL.dll;

— WebApp.caproj.

**1.1 Програмний модуль DAL.dll**

Найменування програмного модуля ― “DAL.dll”. Мова програмної реалізації — C#. Середовище розробки – Microsoft Visual Studio 2017. Використана технологія Entity Framework.

Модуль являє собою шар доступу до бази даних.

Даний модуль являє собою бібліотеку, необхідну для виконання таких функцій, як:

— підключення до бази даних;

— доступ до даних;

— запис, редагування, видалення даних.

**1.2 Програмний модуль BLL.dll**

Найменування програмного модуля ― “BLL.dll”. Мова програмної реалізації — C#. Середовище розробки – Microsoft Visual Studio 2017. Використана технологія AutoMapper.

Модуль являє собою шар бізнес логіки.

Даний модуль являє собою бібліотеку, необхідну для виконання таких функцій, як:

— виконання усіх функціональних вимог системи;

— надання коректних даних до шару доступу до бази даних

**1.3 Програмний модуль WebApp.caproj**

Найменування програмного модуля ― “WebApp.caproj”. Мова програмної реалізації — C#, JavaScript, HTML, CSS. Середовище розробки – Microsoft Visual Studio 2017. Використана технологія Google Maps API.

Модуль являє собою шар графічного представлення даних.

Даний модуль являє собою веб-проект, необхідний для виконання таких функцій, як:

—взаємодія з користувачем;

— візуалізація даних;

**2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ**

Розроблені програмні модулі забезпечують користувачу виконання усіх функцій системи, а саме доступ да даних, редагування даних, виконання запитів користувача та відображення статистичної інформації.

В модулі DAL.dll виконуються наступні функції:

— public void Save();

— public virtual void Dispose(bool disposing) ;

— public void Dispose();

— public IEnumerable<Station> Find(Func<Station, Boolean> predicate) ;

— public void Create(Station item) ;

— public void Update(Station item) ;

— public void Delete(int id);

Тощо.

В модулі BLL.dll виконуються наступні функції:

— public int CreateWorker(WorkerDTO workerDTO) ;

— public bool DeleteWorker(WorkerDTO workerDTO) ;

— public IEnumerable<WorkerDTO> GetWorkers();

— public bool DeleteLaboratory (LaboratoryDTO laboratoryDTO) ;

— public IEnumerable<LaboratoryDTO> GetLaboratories();

Тощо.

В модулі WebApp.caproj виконуються наступні функції:

— public ActionResult SearchStatisticData();

— public ActionResult DrawChart(StatisticModel model) ;

— public ActionResult Maps();

— public JsonResult GetLineChartStatistic(int station, int mounthAmount, int indicator);

Тощо.

**3. ОПИС ЛОГІЧНОЇ СТРУКТУРИ**

Модуль WebApp.caproj відповідає за візуалізацію даних та коректну відповідь користувачу.

В методах цього модуля викликаються методи модуля BLL.dll, які відповідають за функціональну обробку даних і ці методи в свою чергу можуть викликати методи модуля DAL.dll, які взаємодіють з даними в базі даних..

**4. ВХІДНІ ДАНІ ТА РЕЗУЛЬТАТ**

Модуль WebApp.caproj. викликається як реакція на дії користувача з веб-сторінкою. В якості вхідних даних використовуються дані, що були попередньо введені до системи та ті, що були надані користувачем в процесі роботи.

Результатом використання модуля є виведення графіків динаміки індикаторів забруднення води. А також карта з позначенням на ній місцеположення пунктів забору води маркерами в залежності від класу їх небезпеки. Користувач може вільно працювати з отриманими результатами.

Додаток 4

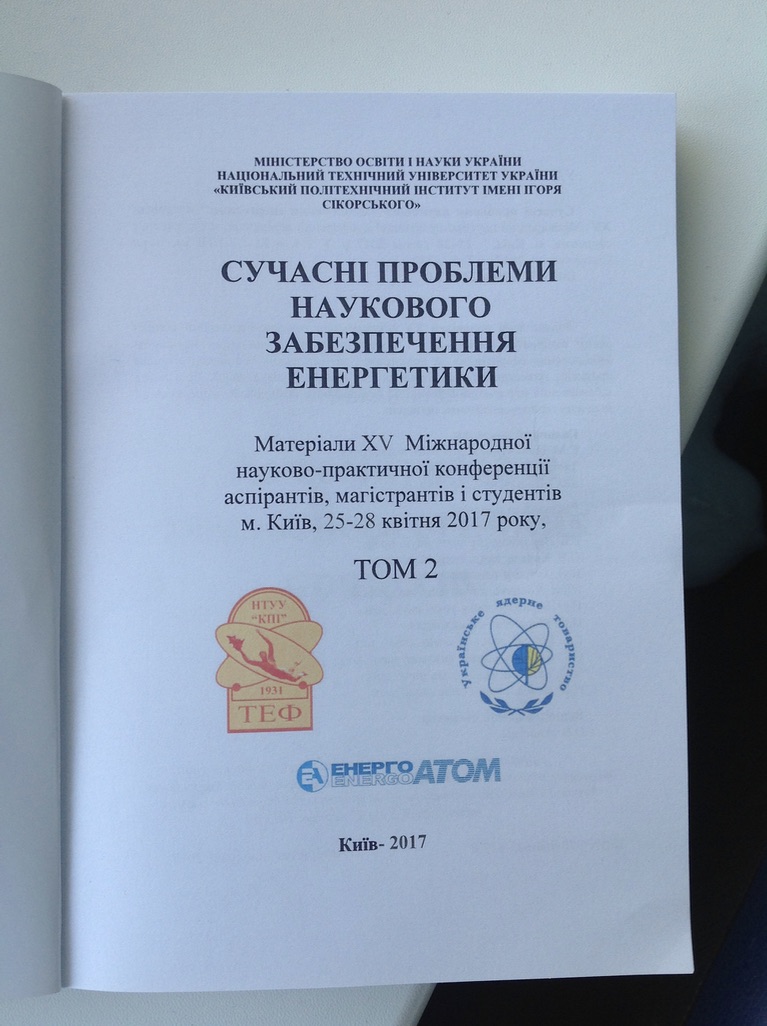
Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

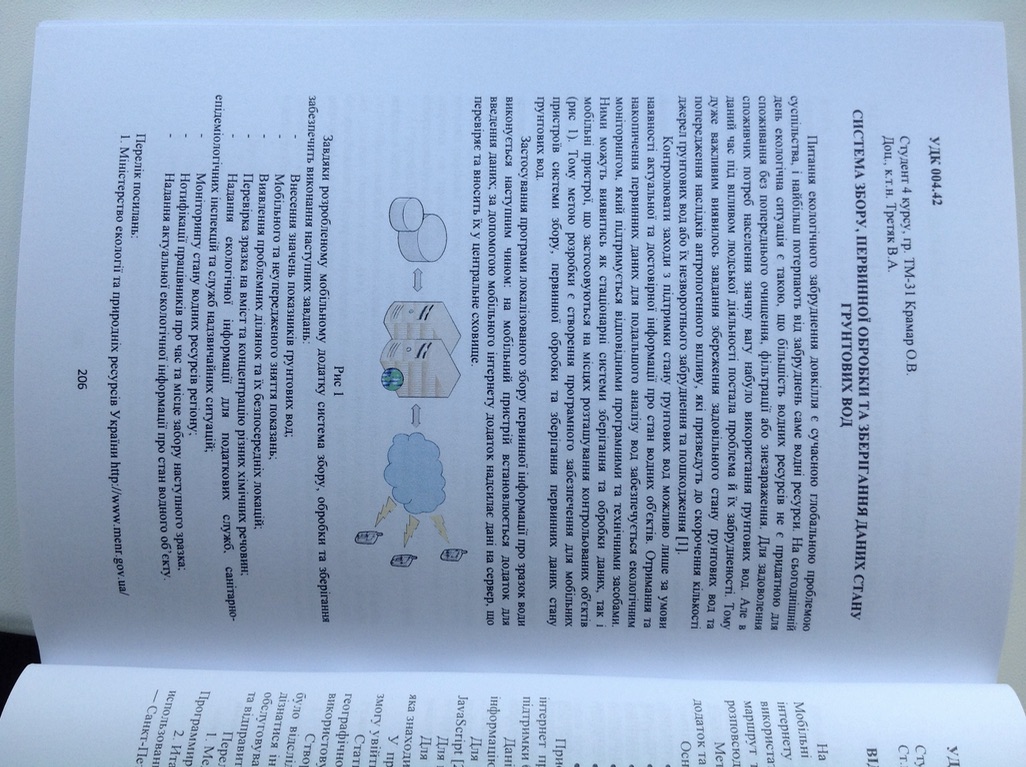
Публікації

УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 2

2017





Додаток 5

Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

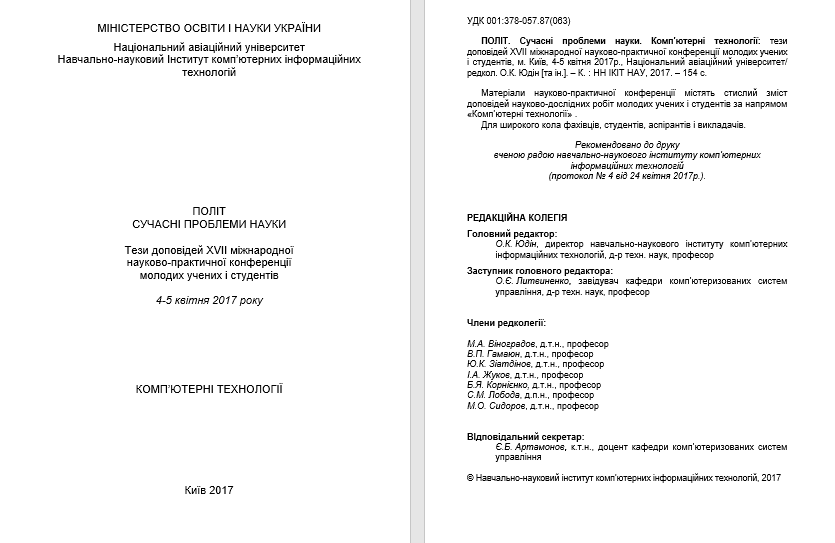
Публікації

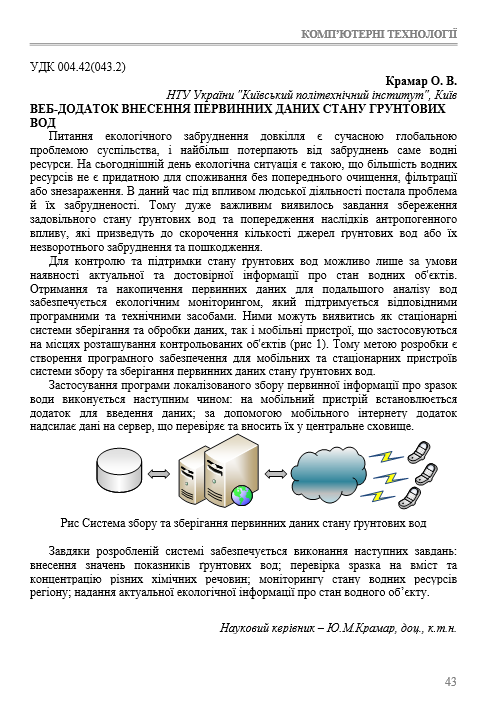
УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 3

2017







Додаток 6

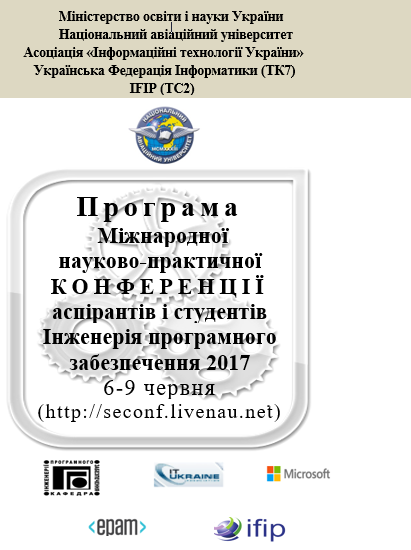
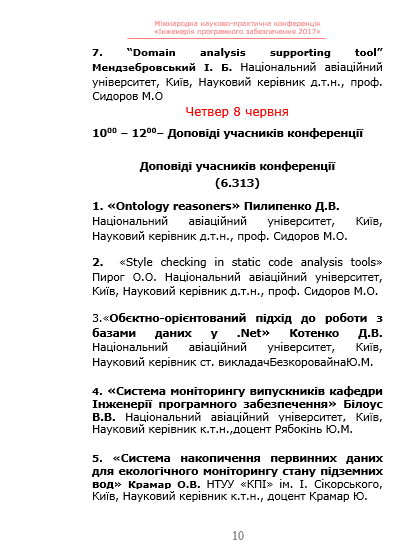
Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

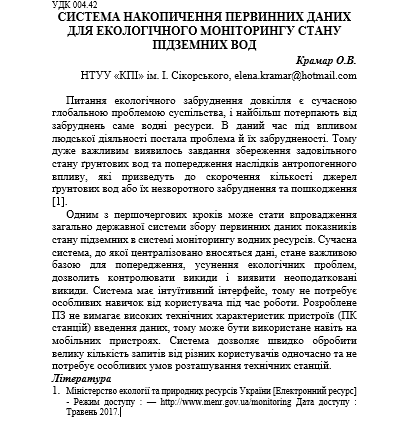
Публікації

УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 2

2017



Додаток 7

Web-додаток внесення первинних даних для екологічного моніторингу стану підземних вод

Копія акту впровадження

УКР.НТУУ “КПІ ім. І. Сікорського”. ТМ31128\_17Б 13-1

Аркушів 2

2017