1. ВИБІР МЕТОДИКИ РОЗРОБКИ ВЕБ СИСТЕМИ ЩО ДОЗВОЛЯЄ ЗБЕРІГАТИ ТА ОТРИМУВАТИ ІНФОРМАЦІЮ ПРО ВОДНІ ОБ’ЄКТИ

Сучасний стан якості та стану багатьох річок в Україні не є задовільним, у т.ч. на окремих територіях Миколаївської області. Особливо, це стосується місць, розташованих у басейні річки Південний Буг нижче за течією від Олександрівського водосховища, особливо у меженний період, та річок, на які постійно здійснюється антропогенний вплив у вигляді скиду стічних та зворотних вод, а також, природний стік із забруднених територій та сільськогосподарських полів і ділянок. Є необхідним вжиття певних заходів з регулювання цього впливу та управління ситуацією, що склалася, з метою поліпшення якості поверхневих вод та приведення їх у відповідність до нормативів. Подібні заходи та прийняття керівних рішень повинні бути науково-обґрунтованими та оптимальними у класі можливих рішень.

Для того ж, щоб забезпечити зручний доступ до даних моніторингу та даних про водні ресурси, необхідним є створення програмно-інформаційного забезпечення системи для Миколаївської області з використанням геоінформаційних технологій, які, як показав світовий досвід, є оптимальним способом формалізації та подання екологічної інформації.

Робота відповідає Закону України "Про затвердження Загальнодержавної цільової програми розвитку водного господарства та екологічного оздоровлення басейну річки Дніпро на період до 2021 року" від 24 травня 2012 р. № 4836-VI та Розпорядженню Кабінету Міністрів України від 17 жовтня 2007 р. № 880-р «Про схвалення Концепції національної екологічної політики України на період до 2020 року», а також основним завданням Водної Рамкової Директиви Європейського Союзу [1].

Основною метою роботи є підвищення обґрунтованості та ефективності рішень з інтегрованого управління водними ресурсами Миколаївської області за рахунок створення та впровадження геоінформаційної системи з основними даними про водні ресурси та даними моніторингу якості вод і водокористування Миколаївської області як в цілому по області, так і окремо для басейнів річок у межах області.

* 1. **Огляд методів розробки складних веб-систем**

При всьому різноманітті різних веб систем для зберігання і пошуку об’єктів, типовий метод складається з трьох основних компонентів:

* розробка стратегії зберігання інформації у базі даних;
* створення серверної логіки для прийняття вихідних даних клієнту для подальшої обробки;
* створення клієнтської частини з інтерфейсом що відповідає вимогам мікроергономіки

Крім цього, побудова веб-системи спирається на апріорну інформацію про предметну область (в даному випадку – водні об’єкти фонду) і коригується вихідними даними користувача, що з'являються по мірі користування веб-системою.

* + 1. **Клієнтська частина**

Клієнтська частина - це практичне застосування технологій HTML, CSS та JavaScript для створення сайту або веб-додатку, щоб користувач переглядав їх та взаємодіяв з ними напряму. Вимоги що відносяться до клієнстької розробки повязані з інструментами та техніками що використовуються для створення сайту постійно змінюються, і розробник повинен постійно слідкувати за розвитком цих інструментів та технологій

Задача в розробці сайту - впевнетись що коли користувач відкриє сайт - він побачить релевантну інформацію у зручному для читання вигляді. Це ускладнюється ще й тим, що тепер користувачі використовують велику різноманітність пристроїв з різними розмірами екрану і розширенями, таким чином змушуючи верстальника враховувати ці аспекти при розробці сайту.

Розробник повинен гарантувати, що сайт коректно відображається в різних браузерах (cross-browser), різних операційні системи (cross-platform) і на різних пристроях (cross-device), що вимагає ретельного планування на стороні розробника.

Є кілька інструментів які можуть бути використані для розробки клієнтської частини веб-сайту, розуміння, які інструменти краще всього підходить для виконання конкретних завдань відзначає різницю між оптимізованим та нераціональним сайтом

Першою з тріади технологій веб сайтів є мова розмітки HTML, це стандартна мова розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді.

HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документа та ідеологію структурної розмітки тексту. Попри те, що HTML - штучна комп'ютерна мова, вона не є мовою програмування. HTML разом із каскадними таблицями стилів та вбудованими скриптами - це три основні технології побудови веб-сторінок.

HTML впроваджує засоби для:

* створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;
* отримання інформації із Всесвітньої мережі через гіперпосилання;
* створення інтерактивних форм;
* включення зображень, звуку, відео, та інших об'єктів до тексту.

Проте для більш зручної стилізації використовуються каскадні таблиці стилів (CSS) - це спеціальна мова, що використовується для опису сторінок, написаних мовами розмітки даних. Найчастіше CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів. Специфікації CSS були створені та розвиваються Консорціумом Всесвітньої мережі.

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3. Профілі - сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів. Наприклад, існують профілі CSS для принтерів, мобільних пристроїв тощо.

CSS (каскадна або блочна верстка) прийшла на заміну табличній верстці веб-сторінок. Головна перевага блочної верстки - розділення змісту сторінки (даних) та їхньої візуальної презентації.

Однак якщо є потреба створення динамічної та інтерактивної веб системи – потрібно застосувати JavaScript, що дозволить взаємодіяти користувачу з інтерфейсом та організувати клієнт-серверну комунікацію.

JavaScript - динамічна, об'єктно-орієнтована мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується як частина браузера, що надає можливість коду на стороні клієнта (такому, що виконується на пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки. Мова JavaScript також використовується для програмування на стороні сервера (подібно до таких мов програмування, як Java і C#), розробки ігор, стаціонарних та мобільних додатків, сценаріїв в прикладному ПЗ (наприклад, в програмах зі складу Adobe Creative Suite), всередині PDF-документів тощо.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Незважаючи на схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманний «у спадок» від мови С, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme.

* + 1. **Серверна частина**

Веб-сервер - це [сервер](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80), що приймає [HTTP](https://uk.wikipedia.org/wiki/HTTP)-запити від [клієнтів](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), зазвичай [веб-браузерів](https://uk.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80), видає їм [HTTP](https://uk.wikipedia.org/wiki/HTTP)-відповіді, зазвичай разом з [HTML](https://uk.wikipedia.org/wiki/HTML)-сторінкою, зображенням, [файлом](https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB), медіа-потоком або іншими даними. Веб-сервер - основа [Всесвітньої павутини](https://uk.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D1%81%D0%B2%D1%96%D1%82%D0%BD%D1%8F_%D0%BF%D0%B0%D0%B2%D1%83%D1%82%D0%B8%D0%BD%D0%B0). Веб-сервером називають як [програмне забезпечення](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), що виконує функції веб-сервера, так і комп'ютер, на якому це програмне забезпечення працює. Клієнти дістаються веб-сервера за [URL](https://uk.wikipedia.org/wiki/URL)-адресою потрібної їм [веб-сторінки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0) або іншого ресурсу.

Додатковими функціями багатьох веб-серверів є:

- Ведення [журналу серверу](https://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3) про звернення користувачів до ресурсів

- [Автентифікація](https://uk.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F) користувачів

- Підтримка сторінок, що динамічно генеруються

- Підтримка [HTTPS](https://uk.wikipedia.org/wiki/HTTPS) для захищених з'єднань з клієнтами

Сервер - у комп'ютерній термінології термін може стосуватися окремого комп'ютера чи програми. Головною ознакою в обох випадках є здатність машини чи програми переважну кількість часу працювати автономно, без втручання людини, реагуючи на зовнішні події відповідно до встановленого програмного забезпечення. Втручання людини відбувається під час встановлення серверу і під час його сервісного обслуговування. Часто це роблять окремі [адміністратори](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%B8%D0%B9_%D0%B0%D0%B4%D0%BC%D1%96%D0%BD%D1%96%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%BE%D1%80) серверів з вищою кваліфікацією.

Сервер як комп'ютер - це [комп'ютер](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) у [локальній](https://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0) чи глобальній мережі, який надає користувачам свої обчислювальні і дискові ресурси, а також доступ до встановлених сервісів; найчастіше працює цілодобово, чи у час роботи групи його користувачів.

Сервер як програма - [програма](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B0), що надає деякі послуги іншим програмам ([клієнтам](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82)). Зв'язок між клієнтом і сервером зазвичай здійснюється за допомогою [передачі повідомлень](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BC%D1%96%D0%BD_%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D0%BE%D0%BC%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F%D0%BC%D0%B8), часто через мережу, і використовує певний [протокол](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) для [кодування](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) запитів клієнта і відповідей сервера. Серверні програми можуть бути встановлені як на серверному, так і на персональному комп'ютері, щоразу вони забезпечують виконання певних служб (наприклад, [сервер баз даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%9A%D0%91%D0%94) чи [веб-сервер](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80)).

У більшості загального користування сервер фізичного [комп'ютера](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) (система комп'ютерної техніки) призначений запустити одну або декілька послуг (як приймаюча сторона) для задоволення потреб користувачів інших комп'ютерів в мережі. В залежності від обчислювальних послуг, які вона пропонує, це може бути сервер баз даних, файловий сервер, поштовий сервер, сервер друку, веб-сервер, ігровий сервер, або якийсь інший сервер. У контексті архітектури клієнт-сервер, сервер являє собою комп'ютерну програму, яка обслуговує запити інших програм - «клієнтів».

Таким чином, сервер виконує деякі обчислювальні завдання від імені "клієнтів". Сервери часто надають основні послуги через мережу, або в приватних користувачів - всередині великої організації або громадським користувачам - через [Інтернет](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82). Мережевий сервер являє собою комп'ютер, призначений для обробки запитів і передачі даних на інші (клієнт) комп'ютери по локальній мережі або через Інтернет. Мережеві сервери зазвичай конфігуруються з додатковою пам'яттю і ємністю для обробки навантаження з обслуговування клієнтів.

Вимоги до обладнання для серверів варіюються залежно від сервера додатків. Абсолютна швидкість [процесора](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80) не настільки важлива для сервера, як для настільного комп'ютера. Обов'язки сервера надавати послуги багатьом користувачам по мережі призводять до різних вимог, таких як швидке підключенням до мережі та висока [пропускна спроможність](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BF%D1%83%D1%81%D0%BA%D0%BD%D0%B0_%D1%81%D0%BF%D1%80%D0%BE%D0%BC%D0%BE%D0%B6%D0%BD%D1%96%D1%81%D1%82%D1%8C_%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB%D1%83). Так як сервери, як правило, доступні по мережі, вони можуть працювати без монітору. Процеси, які не потрібні для функції сервера не використовуються. Багато серверів не мають графічного інтерфейсу користувача. Крім того, аудіо-та USB інтерфейси можуть бути опущені. Сервери часто працюють протягом тривалого часу без перерви, тому надійність обладнання і довговічність надзвичайно важлива.

Хоча сервери можуть бути побудовані з частин комп'ютера, критично важливі корпоративні сервери не можливі без використання спеціалізованого устаткування з низьким рівнем збою в цілях максимального часу безперебійної роботи, оскільки навіть короткострокові невдачі можуть коштувати дорожче, ніж покупка і установка системи . Наприклад, це може зайняти всього декілька хвилин часу простою на національній [фондовій біржі](https://uk.wikipedia.org/wiki/%D0%A4%D0%BE%D0%BD%D0%B4%D0%BE%D0%B2%D0%B0_%D0%B1%D1%96%D1%80%D0%B6%D0%B0), щоб виправдати рахунок повністю замінити системи з чимось більш надійним. Сервери можуть включати в себе більшу ємність [жорстких дисків](https://uk.wikipedia.org/wiki/%D0%96%D0%BE%D1%80%D1%81%D1%82%D0%BA%D0%B8%D0%B9_%D0%B4%D0%B8%D1%81%D0%BA), більше комп'ютерних вентиляторів або водяного охолодження, щоб допомогти усунути тепло, і джерела безперебійного живлення, які забезпечують роботу сервера в разі збою живлення. Ці компоненти забезпечують більш високу продуктивність і надійність за відповідно більш високою ціною.

* + 1. **Аналіз існуючих баз даних**

База даних – сукупність [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96_(%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D1%8E%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0_%D1%82%D0%B5%D1%85%D0%BD%D1%96%D0%BA%D0%B0)), організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами; ця сукупність підтримує щонайменше одну з областей застосування. В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти.

Дані у базі організовують відповідно до моделі організації [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96). Таким чином, сучасна база даних, крім саме даних, містить їх опис та може містити засоби для їх обробки.

В загальному випадку базою даних можна вважати будь-який впорядкований набір даних. Наприклад, паперову картотеку з формулярами про працівників підприємства у відділі кадрів. Але дана стаття зосереджена на використанні баз даних в [інформаційних системах](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0). На даний час [застосунки](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) для роботи з базами даних є одними з найпоширеніших прикладних програм.

У сучасних мережевих інформаційних системах для роботи із загальною базою даних використовують архітектуру [«клієнт-сервер»](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). При цьому в мережі розміщують сервер баз даних. Ним виступає [комп'ютер](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80) (або комп'ютери), який містить бази даних, [СКБД](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) та пов'язане з ними програмне забезпечення, і налаштований для надання користувачам інформаційної системи доступу до бази даних.

Клієнти, які працюють із даними (вони можуть бути розташовані на різнихкомп'ютерах мережі), надсилають відповідні запити серверу. Сервер їх отримує, опрацьовує, та надсилає відповідь клієнту.

Сучасні [СКБД](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) ([MySQL](https://uk.wikipedia.org/wiki/MySQL), [PostgreSQL](https://uk.wikipedia.org/wiki/PostgreSQL), [Microsoft SQL Server](https://uk.wikipedia.org/wiki/Microsoft_SQL_Server) та інші) працюють відповідно до цієї архітектури. Сервер баз даних, як правило, є достатньо потужною багатопроцесорною системою, яка використовує масиви дисків [RAID](https://uk.wikipedia.org/wiki/RAID) для підвищення надійності зберігання даних. Використання дискових масивів [RAID](https://uk.wikipedia.org/wiki/RAID) дозволяє відновити дані, навіть якщо один з дисків вийшов з ладу.

Первинним призначенням бази даних є зберігання масивів даних. Але їх широко використовують і для збереження адміністративної інформації та спеціалізованих даних, наприклад, для інженерних даних чи для економічних моделей. Прикладами використання баз даних можуть бути:

* автоматизовані системи обліку;
* реєстри та каталоги;
* геоінформаційні системи;

Об'єктно-орієнтована база даних - база даних, яка створюється і використовується в середовищі СУБД, яке засновано на принципах об'єктно-орієнтованого підходу і підтримує об'єктну модель даних. Об'єктна база даних являє собою сукупність взаємозв'язаних об'єктів, які відповідають певній схемі.

Використання об’єктної моделі не є доцільним для предметної області нормативних документів ВНТУ, так як переваги такої моделі будуть слабо відчуватися через малу потужність універсального відношення. Крім того виникнуть проблеми на фізичному та даталогічному рівнях проектування у зв’язку із складністю реалізації алгоритмів роботи СУБД.

Після аналізу основних моделей даних, для курсової роботи було обрано реляційну модель даних. Саме ця модель має ряд переваг, таких як: незалежність від фізичного рівня представлення, зручність і розуміння організації даних користувачами, максимальна гнучкість при обробці непередбачених запитів, можливість розширення бази приєднанням нових елементів, записів без зміни при цьому існуючих підсхем та прикладних програм.

Через такі недоліки ієрархічної моделі, як: надлишковість зберігання інформації, так як ієрархічні структури не підтримують взаємозв'язки Б:Б; строгу ієрархічну впорядкованість, яка ускладнює процедури включення та вилучення записів; вилучення вихідних вузлів призводить до вилучення відповідних їм породжених , що вимагає особливої обережності; ускладнюється доступ до даних , які лежать на більш низьких рівнях ієрархії, так як кореневий вузол завжди є головним, а доступ до любого породженого вузла може здійснюватись через вихідний, ієрархічна модель не може бути використана для даної курсової роботи.

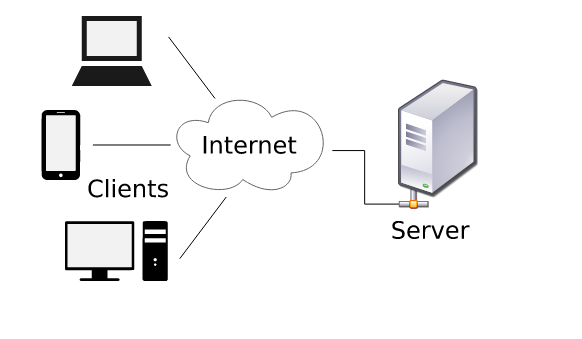
А недоліки використання мережевої моделі даних полягають в складності по відношенню до ієрархічних структур. Прикладному програмісту часто необхідно знати логічну структуру бази даних.

* + 1. **Клієнт-серверна архітектура**

Архітектура клієнт-сервер є одним із [архітектурних шаблонів програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%BD%D1%96_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) та є домінуючою концепцією у створенні розподілених мережних [застосунків](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти:

* набір [серверів](https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80), які надають інформацію або інші послуги програмам, які звертаються до них;
* набір [клієнтів](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82_(%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), які використовують сервіси, що надаються серверами;
* [мережа](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D1%83%D0%BD%D1%96%D0%BA%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0), яка забезпечує взаємодію між клієнтами та серверами.

Сервери є незалежними один від одного. Клієнти також функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів.



Дуже важливо ясно уявляти, хто або що розглядається як «клієнт». Можна говорити про клієнтський комп'ютер, з якого відбувається звернення до інших комп'ютерів. Можна говорити про клієнтське та серверне програмне забезпечення. Нарешті, можна говорити про людей, які бажають за допомогою відповідного програмного та апаратного забезпечення отримати доступ до тієї чи іншої інформації.

Модель клієнт-серверної взаємодії визначається перш за все розподілом обов'язків між клієнтом та сервером. Логічно можна відокремити три рівні операцій:

- рівень представлення даних, який по суті являє собою інтерфейс користувача і відповідає за представлення даних користувачеві і введення від нього керуючих команд;

- прикладний рівень, який реалізує основну логіку застосунку і на якому здійснюється необхідна обробка інформації;

- рівень управління даними, який забезпечує зберігання даних та доступ до них.

Дворівнева клієнт-серверна архітектура передбачає взаємодію двох програмних модулів - клієнтського та серверного. В залежності від того, як між ними розподіляються наведені вище функції, розрізняють:

- модель тонкого клієнта, в рамках якої вся логіка застосунку та управління даними зосереджена на сервері. Клієнтська програма забезпечує тільки функції рівня представлення;

- модель товстого клієнта, в якій сервер тільки керує даними, а обробка інформації та інтерфейс користувача зосереджені на стороні клієнта. Товстими клієнтами часто також називають пристрої з обмеженою потужністю: кишенькові комп'ютери, мобільні телефони та ін.

Типовим прикладом клієнт-серверної взаємодії є WWW. Існує величезна кількість веб-серверів, на яких розміщується та чи інша інформація. У найпростішому випадку ця інформація являє собою набір веб-сторінок, які можуть зберігатися на сервері у вигляді файлів, розмічених за допомогою мови розмітки HTML. Але ситуація, як правило, є складнішою; значна частина веб-ресурсів на сучасному етапі є динамічними, тобто вони не існують в заздалегідь підготовленому вигляді, а створюються безпосередньо в процесі обробки запиту від користувача.

Для того, щоб людина, яка працює в Інтернеті, могла переглянути ту чи іншу сторінку, на її комп'ютері повинно бути встановлено відповідне програмне забезпечення. Програми для перегляду веб-сторінок називаються браузерами (веб-оглядачами). Найпоширеніші браузери: Google Chrome, Internet Explorer, Firefox, Safari і Opera.

Але, крім браузерів, до серверів можуть звертатися і інші клієнти, а саме - автономні програми. Вони можуть передбачати взаємодію з людиною, а можуть працювати в цілком автоматичному режимі. Типовим класом таких програм є роботи, призначені для автоматичного перегляду веб-ресурсів. Зокрема, роботи є важливим елементом пошукових систем і використовуються ними для перегляду сторінок і збору інформації про них.

Для запиту до веб-сервера клієнтська програма повинна задати місцезнаходження комп'ютера, на якому розміщується серверна програма, назву потрібного документа і, можливо, інші дані, які специфікують запит. Мережа забезпечує знаходження сервера і передачу йому клієнтського запиту. Серверні програми обробляють цей запит, відповідь пересилається по мережі клієнтові.

1. ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

При розробці веб-системи, необхідно звернути увагу на функції, які повинна вона виконувати, а саме: зручність та швидкість пошуку.

Пошук водних об’єктів – клопіткий процес, який потребує детального аналізу. Не завжди вдається знайти той чи інший ставок, річку або водосховище у відповідних архівах. Тому використання програмного засобу для такого завдання є актуальним.

Веб-система для пошуку водних об’єктів повинен отримати вхідні критерії водного об’єкту, та виконати відповідні алгоритми на сервері для перевірки наявності таких об’єктів, знайти геодані, відповідну статистику для вказаного типу водного об’єкту (ставок, річка, водосховище). Після пошуку веб-система повинна вивести інформацію знайдені об’єкти та відобразити їх користувачу.

Це повинно полегшити процес пошуку водних об’єктів для людей, які мають справи з водними кадастрами, підприємців, особи що працюють у сфері гідрології, або просто бажаючих дізнатись більше про водні об’єкти. Веб-система є інструментом, який забезпечить зручний та ефективний пошук водних об’єктів на відміну від малофункціональних або важких для розуміння аналогів.

Проблема полягає у відсутності простого, безкоштовного, і водночас багатофункціонального та продуктивного інструменту для пошуку водних об’єктів. Дана проблема стосується осіб що працюють у сфері ботаніки, або просто бажаючих дізнатись більше інформації про рослин. Як наслідок – виникають проблеми з швидким пошуком інформації про певні водні об’єкти. Вирішення проблеми дозволить створити ефективний та зручний інструмент для пошуку водних об’єктів.

1. ВИБІР МЕТОДИКИ РОЗРОБКИ

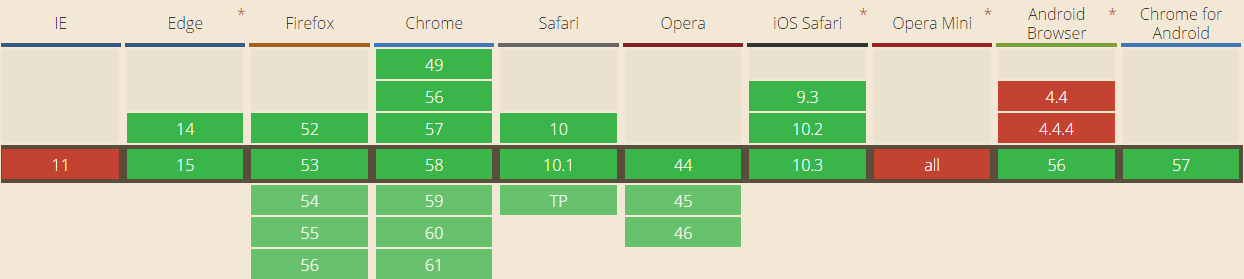
Для створення веб-системи що дозволить виконувати пошуку водних об’єктів використаємо стек MERNRW (MySQL, Express, React, NodeJS, Redux, Webpack).

Стек - набір програмних підсистем або компонентів, необхідних для створення повної платформи таким чином, що ніякого додаткового програмного забезпечення не вимагається для підтримки додатків.

Стек MERN складається з чотирьох основних технологій:

* MySQL - вільна система керування реляційними базами даних, використовується, в першу чергу, для створення динамічних веб-сторінок, оскільки має чудову підтримку з боку різноманітних мов програмування. Це компактний багатопотоковий сервер баз даних. Характеризується високою швидкістю, стійкістю і простотою використання
* Express - програмний каркас розробки веб-застосунків для Node.js, реалізований як вільне і відкрите програмне забезпечення під ліцензією MIT. Він спроектований для створення веб-застосунків і API
* React - відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків. React дозволяє розробникам створювати великі веб-застосунки, які використовують дані, котрі змінюються з часом, без перезавантаження сторінки. React обробляє тільки користувацький інтерфейс у застосунках
* Node.js - платформа з відкритим кодом для виконання високопродуктивних мережевих застосунків, написаних мовою JavaScript. Node.js характеризується такими властивостями як асинхронна однопотокова модель виконання запитів, неблокуючий ввід/вивід, система модулів CommonJS, рушій JavaScript Google V8
* Redux - JavaScript бібліотека з відкритим кодом створена для управління станом веб-додатка. В основному використовується разом з React для розробки гнучких веб інтерфейсів.
* Webpack - JavaScript модуль збірки з відкритим кодом. Webpack бере модулі веб додатку та їх залежності та генерує статичні файли що репрезентують ці модулі. Також може відобразити граф залежностей проекту, що дозволить розробнику застосувати модулярний підхід для створення веб додатку

Однією з особливостей Node.js та React є можливість використання експерементального стандарту JavaScript що має назву ECMAScript 6 або скорочено ES6. На відміну від свого попередника ECMAScript 2015 в ES6 раціонально реалізовано об’єктно-орієнтований підхід та ряд інших функцій що дозволяє розробнику створювати більш гнучкі додатки. Проте стандарт ES6 підтримується лише у 73% браузерів на всіх пристроях [15]



Однак цю ситуацію можна виправити за допомогою JavaScript компіляторів, що дозволяють виконати конвертацію коду стандарту ES6 в стандарт ECMAScript 2015, котрий підтримується набагато більшою кількістю браузерів. Для такої задачі найчастіше за все використовують Webpack. Він дозволяє налаштувати процес непреривної компіляцій вихідних файлів JavaScript ES6 у JavaScript ECMAScript 2015.

1. ВИЗНАЧЕННЯ МЕТИ, ОБ’ЄКТУ, ПРЕДМЕТУ ТА ЗАВДАНЬ ПОДАЛЬШОГО ДОСЛІДЖЕННЯ

Метою дослідження є збільшення точності розпізнавання об’єктів на зображенні.

Об’єктом даного дослідження є процес розпізнавання образів.

Предметом даного дослідження є методи штучного інтелекту, що дозволяють розпізнавати образи.

Основними завданнями подальшого дослідження є:

* Розробка структури веб-системи;
* Розробка пошуку водних об’єктів;
* Розробка інформаційно-довідкової веб-системи обліку об’єктів водного фонду Миколаївської області;
* Налагодження і тестування програми.

ПЕРЕЛІК ПОСИЛАНЬ

1. Закон України "Про затвердження Загальнодержавної цільової програми розвитку водного господарства та екологічного оздоровлення басейну річки Дніпро на період до 2021 року" від 24 травня 2012 р. № 4836-VI.
2. Комп’ютеризовані регіональні системи державного моніторингу поверхневих вод: моделі, алгоритми, програми. Монографія / Під ред. В. Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2005. - 315 с.
3. Система прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Південний Буг з використанням геоінформаційних технологій: Звіт про НДР / В.Б. Мокін, М.П. Боцула та ін. / Вінниц. нац. техн. ун-т. - 2805; № ДР 0105U006684.- Інв. № 0206U006852.- К., 2006.- 99 с.
4. Дезірон О. В., Мокін В. Б., Крижановський Є. М. Геоінформаційна система басейну річки Сіверський Донець та її роль в прийнятті управлінських рішень // Водне господарство України. – 2006.– №4.– С. 10–15.
5. Система прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Південний Буг з використанням геоінформаційних технологій: Звіт про НДР / В.Б. Мокін, М.П. Боцула та ін. / Вінниц. нац. техн. ун-т. - 2805 (№ ДР 0105U006684) - Інв. № 0206U006852.- К., 2006.- 99 с.
6. Створення системи підтримки прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Сіверський Донець з використанням геоінформаційних технологій: Звіт про НДР / В.Б. Мокін, М.П. Боцула та ін. / Вінниц. нац. техн. ун-т. - 2809; (№ ДР 0107U005997).- Інв. № 0207U006945.- К., 2007.- 58 с.
7. Розробка методів інтеграції математичних моделей природних процесів з геоінформаційними системами природних екосистем: Звіт про НДР / В.Б. Мокін, Г.В. Горячев та ін. / Вінниц. нац. техн. ун-т. - 28-Д-299; № ДР 0108U000654- Інв. № 0211U001042.- К., 2009.- 217 с.
8. Інформаційна технологія інтегрування математичних моделей у геоінформаційні системи моніторингу поверхневих вод : монографія / В. Б. Мокін, Є. М. Крижановський, М. П. Боцула. - Вінниця : ВНТУ, 2011 - 152 с.
9. Система підтримки прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Дністер з використанням геоінформаційних технологій: Методичний посібник / [Мокін В.Б., Мокін Б.І., М.Я. Бабич та ін.]; під ред. В.Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2009.- 252 с.
10. Система підтримки прийняття рішень з моніторингу та управління водними ресурсами Львівської області: Методичний посібник / [Мокін В.Б., Мокін Б.І., Сташук В.А. та ін.]; під ред. В.Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2009.- 236 с.
11. Система прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Південний Буг з використанням геоінформаційних технологій.: Методичний посібник / [Мокін В.Б., Мокін Б.І., О.В. Дезірон та ін.]; під ред. В.Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2009.- 244 с.
12. Система підтримки прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Прип’ять з використанням геоінформаційних технологій: Методичний посібник / [Мокін В.Б., Мокін Б.І., Сташук В.А. та ін.]; під ред. В.Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2009.- 236 с.
13. Геоінформаційна аналітична система моніторингу якості і використання водних ресурсів та стану водогосподарських об’єктів річки Тиса у Закарпатській області: Методичний посібник / [Мокін В.Б., Мокін Б.І., В.П. Чіпак та ін.]; під ред. В.Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2009.- 228 с.
14. Система підтримки прийняття управлінських рішень керівниками водогосподарських організацій для басейну річки Сіверський Донець з використанням геоінформаційних технологій: Методичний посібник / [Мокін В.Б., Мокін Б.І., М. Я. Бабич та ін.]; під ред. В.Б. Мокіна. - Вінниця: УНІВЕРСУМ-Вінниця, 2009.- 352 с.
15. "Can I use" provides up-to-date browser support tables for support of front-end web technologies on desktop and mobile web browsers. <http://caniuse.com/#search=es6>