Голуб Назарій Олексійович

Розробка єдиної системи доступу до публічної інформації з використанням ASP .Net MVC Framework та технології C#

Науковий керівник: Цуприк Галина Богданівна

ЗМІСТ

АНОТАЦІЯ

Голуб Н.О. Розробка єдиної системи доступу до публічної інформації з використанням ASP.NetMVC Framework та технології С. – Рукопис

Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедри програмної інженерії, групи СПм-61. – міста Тернополя,2021.

Пояснювальна записка до кваліфікаційної роботи на здобуття освітнього ступеню «магістр» складається з: сторінок, рисунків, таблиць,та додатків.

За мету до кваліфікаційної роботи обрано оптимізація варіанту моделі єдиної системи доступу до публічної інформації при використанні сучасних інформаційних технологій.

Практичне застосування – запропоновану єдину систему доступу до документообігу, який в сучасних умовах рекомендовано здійснювати віддалено. Рекомендовано до впровадження у всіх органах самоврядування, зокрема місцевого значення.

Технічні вимоги – методи розробки базуються на використанні архітектурного шаблону MVC, сучасних засобів програмування та керування базами даних.

Ключові слова: ОБ'ЄКТ, ПРЕДМЕТ, АНАЛІЗ, ЗАДАЧА, ВАРІАНТИ ВИКОРИСТАННЯ, БАЗА ДАНИХ, ТЕХНОЛОГІЇ, ПРОЄКТУВАННЯ, КЛАСИ, ТЕСТУВАННЯ, ОХОРОНА ПРАЦІ

ABSTRACT

Holub N.O. Development of a unified system of access to public information using ASP.NetMVC Framework and C technology. – Manuscript

Ternopil Ivan Pul'ui National Technical University, Faculty of Computer Information Systems and Software Engineering, Software Engineering Department, group SPm-61. – Ternopil, 2021.

Magisters's thesis contains ___pages, ___figures, ___tables and ___supplements.

The purpose of the qualification work is to optimization of the model of unified system of access to public information. Modern information technologies used for this purpose.

Practical application - proposed the unified system of access to document management. This process must be distance. It is caused by pandemic restrictions. Is is recomendated to use and for implementation in all self-government bodies, including local ones.

Technical requirements are development methods are based on the use of MVC architectural template, modern programming and database management tools.

Keywords: OBJECT, SUBJECT, ANALYSIS, TASK, USE CASE, DATABASE, TECHNOLOGIES, DESIGN, CLASSES, TESTING, LABOUR PROTECTION

ЗМІСТ

ВСТУП

Кожна людина, тим паче громадянині не лише нашої, але й будь-якої держави світу з верховенством права, має не лише обов'язки перед державою на території якої вона проживає, але і права, ґарантовані конституційно і ніхто позбавити її цього в жодному разі, крім законних шляхів, не може. Гарантії та порядок реалізації прав громадянина закріплено конкретним законом України «Про можливість доступу до публічної інформації». Усі суб'єкти владних органів з відповідними повноваженнями, наприклад місцеве самоврядування, територіальні управління чи ограни державної влади, одним із своїх обов'язків перед громадою мають інформування її та засобів масової інформації як про діяльність в цілому, так і про рішення, які розглядаються та приймаються чи відхиляються. З цією метою часто створюють цілі окремі інформаційні підрозділи, з призначеною відповідальною особою та штатом працівників, через яких ґарантують реалізацію права кожного громадянина до ознайомлення з актуальною публічною інформацією та вчасне оприлюднення. Крім цього в кожному державному органі чи самоврядуванні місцевого значення законом закріплена можливість звернення громадян в спеціально організоване місце для роботи безпосередньо з відвідувачами та їх запитами, з наданими документами чи їх копіями, зі зверненнями. Також надавати, при отриманні відповідного запиту виписок з документів, витягів, фото, сканів, при необхідності записувати інформацію на цифрові носії, тобто виконувати всі дії, які підпадають під закон про доступ до публічної інформації. До повноважень таких підрозділів належить також і облік документації, її ведення, та облік всіх запитів будь-якого типу, які до них надходять, чи ними видаються.

Водночас, одним із основник критеріїв оцінки якості роботи будь-якого державного органу, є достовірність оприлюдненої інформації. При чому важливо надати та оприлюднити лише ту інформацію, яка є точно

повною,точною та достовірною. Крім цього, при необхідності потрібно в будь-який момент бути готовими перевірити і контролювати за правильністю, об’єктивністю та оновленням наданої інформації.

Як варіант інформація, яка відповідає всім критерії якості переліченим вище, може оприлюднюватись шляхом друку в офіційних виданнях, публікації на офіційних сторінках (web-сайтах) у Всесвітній мережі, висвітлення з використанням єдиного державного web-порталу дані на якому є відкритими, оприлюднення з використанням інформаційних стендів чи будь-якими іншими доступними офіційними шляхами. Також публічна інформація може надаватись у вигляді відповіді на запити, при цьому такою можливістю можуть скористатись у відповідності з Законом України її громадяни, іноземці, особи, які не мають громадянства, юридичні особи з приватним правом, особи-біженці, а також громадські об’єднання, що не мають статусу юридичної особи.

При цьому, для того, щоб релевантність такого запиту була якнайвищою, важливо, щоби він містив наступні інформацію:

1 – ПІП запитувача (в окремих випадках найменування), його поштова адреса чи адреса електронної пошти, при наявності контактний номер запитувача чи його вповноваженої особи;

1 – вид чи узагальнено опис інформації, яку потрібно отримати, назва, реквізити або якщо можливо ж суть (зміст) документу, стосовно кого здійснено запит;

2 – якщо запит подано у письмовій формі, то його потрібно завізувати підписом та поставити дату.

Також варта зазначити, що сам запит може бути різних видів та типів, зокрема індивідуальний чи колективний, подаватися усно, письмово, через пошту, факс, телефон, електронну пошту), тобто будь-яким легальним способом на розсуд запитувача.

Як видно процедура є доволі не простою, і складність полягає першу чергу в коректності запиту, високій можливості впливу так званого

людського фактору, тривалості по часу процедури в цілому, затрат часу необхідного на відвідування установи, необхідності повторного звернення в разі відсутності відповіді та ще великої кількості факторів, які по одному чи в поєднанні можуть вплинути на якість кінцевого результату. Крім цього, у випадку коли подається запит у письмовому вигляді, то його форма є довільною, і це може спричинити ще один вид проблем, пов’язаних з особливостями сприйняття світу кожної людини окремо.

З метою хоча б часткового спрощення цієї доволі рутинної процедури, запитувач має можливість подати запит за допомогою спеціальних форм, заповнивши їх. Отримати ці форми можна безпосередньо за місцем реєстрації його чи на офіційному web-сайті. Що також не дуже зручно, оскільки звертаються найрізноманітніші соціальні та вікові групи, і для деяких з них це також викличе певні складнощі.

Це одним важливим моментом є і те, що не дивлячись на форму запиту (письмове звернення чи через веб-сторінку) розглядатись звернення запитувача буде щонайменше від п’яти до двадцяти робочих днів рахуючи від дня подачі його. Строк розгляду залежить в основному від об'єму запитуваної інформації чи кількості даних серед який такий пошук потрібно провести. Є лише один вийняток коли термін розгляду запиту може бути скорочено до сорока восьми годин – це коли запит стосується даних які стосуються напряму життя та свободи запитувача, також сюди можна віднести стан навколишнього середовища, якість побутових предметів, продуктів харчування, коли йдеться про катастрофи, аварії, небезпечні природні явища, різні надзвичайні ситуації, які трапляються, трапились чи можуть трапитись і несуть загрозу безпеці людей. Важливо також зазначити, що запитувач отримує інформацію стосовно свого запиту повністю безкоштовно, відшкодовуючи лише, в разі необхідності, вартість розхідного матеріалу для друку та копіювання при необхідності виготовлення копії від десяти сторінок.

Саме це, а також особистий досвід, який я отримав після того, як я стикнувся з особливостями роботи органів самоврядування місцевого рівня і спонукало мене дослідити цей напрямок та обрати означений предмет дослідження. Також мене цікавило і я дослідив питання стосовно законності, і зробив висновок, що розроблюваний програмний продукт не суперечитиме Закону України («Про доступ до публічної інформації») дасть можливість використати сучасні інформаційні технології на службу людям, а саме для простих громадян, юридичних осіб приватного права, громадських об'єднань без статусу юридичних осіб. Запитувачі матимуть легкий, інтуїтивно зрозумілий, швидкий доступ до достовірної та актуальної публічної інформації, якою володіють органи державної влади та форми власності так і органи місцевого самоврядування. І що важливо, що буде реальна можливість і інструмент для контролювання діяльності органів влади. Таким чином кожен зможе реалізувати свої конституційне право на інформацію, не зважаючи на рівень комп'ютерної грамотності, використовуючи інтуїтивно зрозумілий інтерфейс, скористатись ним і буде в курсі всього, що робиться, які рішення приймаються, і що державі відомо про кожного з нас.

1 РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ

1.1 Аналіз предмету дослідження та означення моделі об'єкту

При побудові правового, але лояльного до людей громадського суспільства, та ефективної влади важливе місце посідає громадський контроль за діяльністю органів місцевого самоврядування. Держава, яка будує свою діяльність на засадах чесності та прозорості, по відношенню до своїх громадян, повинна спрощувати та надавати механізми та інструменти з допомогою яких можливо проводити даний громадський контроль, та бути відкритою для людей. Зараз на рівні обласних та деяких районних рад механізм подібного контролю існує – це web-сайти, з допомогою них дані органи місцевого самоврядування висвітлюють свою діяльність(публікують свої рішення та розпорядження) на деяких із них є можливість подання електронного запиту на отримання доступу до інформації, яку можна оприлюднити (є публічною), що також є певним інструментом контролю влади. І зовсім новим але також ефективним інструментом народовладдя є електронні петиції. Ці три інструменти в купі є досить ефективними засобами контролю влади для активних людей, але нажаль не кожен сайт органів місцевої влади володіє цими інструментами, а на рівні сіл та селищ такого поняття як сайт ради взагалі не існує, там щоб дізнатись якусь інформацію потрібно щоразу іти і розпитувати потрібні дані. Якщо взяти до уваги що зараз країна перебуває на етапі децентралізації, відповідно фізична відстань у сільській місцевості між владою та людьми зросте і відповідно без інтернет інструментів громадськості буде важко слідкувати за діяльністю влади.

Досить нелогічним в плані ефективності використання державних коштів є те, що кожна обласна і районна ради створюють свій окремий сайт відповідно кожна використовує чималі кошти на послуги програмістів для розробки і подальшої підтримки своїх сайтів, хоча функціонал у цих усіх сайтів приблизно однаковий і набагато кращим був би варіант єдиного уніфікованого

централізованого порталу який би надавав можливість для усіх ланок органів місцевого самоврядування створювати свої інтернет-сторінки.

Використання єдиного інтернет-порталу для усіх ланок органів місцевого самоврядування має цілий ряд безсумнівних переваг, зокрема:

- держава визначає єдиний загальнонаціональний стандарт обов'язкового функціоналу для сторінок органів влади(зараз органи місцевого самоврядування на своїх офіційних сайтах надають користувачеві лише той функціонал який для влади на місцях є зручнішим);
- велика економія бюджетних кошів на послугах програмістів, уже не буде потреби для кожної із рад писати окремий сайт;
- зручність безпосередньо для громадян оскільки інформація і офіційні документи усіх обласних, усіх районних, селищних та сільських рад знаходитиметься в одному місці, на одному порталі;
- обмін документами між владою та людьми відбуватиметься через мережу інтернет, людям не потрібно буде витрачати час на дорогу до органів влади, таким чином зростає оперативність отримання даних;
- можливість запровадити електронний документообіг на усіх рівнях органів місцевого самоврядування;
- зростання рівня публічності влади усіх рівнів і відповідно зростання рівня дотримання закону, оскільки кожен посадовець знатиме що його рішення будуть викладені на загал громадськості.

1.1.1 Опис функціоналу розроблюваної системи

Для забезпечення усіх вище в підрозділі 1.1 описаних переваг розроблювана система документообігу повинна володіти наступним обов'язковим функціоналом, зокрема:

1 – надання можливості реєстрації різного рівня рад, із одночасною реалізацією механізму контролю реєстрації раді вищої інстанції(рада вищої інстанції надаватиме і додаватиме реєстраційні паролі радам нижчої інстанції і це дозволить унеможливити несанкціоновану реєстрацію);
2 – функція формування запиту на отримання доступу до інформації, яку можна оприлюднити (є публічною), якою володіє рада, користувачам сайту із контролем вчасності надання відповіді органами влади і відповідно повідомлення органів вищої інстанції в разі не надання відповіді на запит громадян в визначений законом термін;
3 – функція подачі петиції із підтримкою усього необхідного функціоналу(подача петиції, голосування за петицію, інформування влади про петиції що набрали необхідну кількість голосів, механізм надання відповіді влади);
4 – функціонал для оприлюднення через інтернет усіх офіційних документів рад із категорійним сортуванням.

1.1.2 Опис вимог до розроблюваної моделі системи

Щоб дана система була дійсно ефективною і її можна було б застосовувати на загальнодержавному рівні пріоритетом номер один повинен бути простий і інтуїтивно зрозумілий інтерфейс оскільки кінцеві користувачі(як держслужбовці з однієї сторони, так і прості громадяни з іншої) це люди із різним, не завжди високим рівнем комп’ютерної грамотності.

Незважаючи на безперечну актуальність вибраного напрямку дослідження, яка однозначно збережеться і після завершення періоду пандемії, на усі наявні достоїнства та велику економію коштів, чомусь поки що аналогів подібної системи не існує. Тож основна мета даної дипломної роботи на здобуття освітнього рівня «магістр» – показати бачення того, який вигляд повинна мати система подібного типу, та яким функціонал вона повинна підтримувати.

Саме це, а також особистий досвід, який я отримав після того, як я стикнувся з особливостями роботи органів самоврядування місцевого рівня і спонукало мене дослідити цей напрямок та обрати означений предмет дослідження. Також мене цікавило і я дослідив питання стосовно законності, і зробив висновок, що розроблюваний програмний продукт не суперечитиме Закону України («Про доступ до публічної інформації»)[1] дасть можливість використати сучасні інформаційні технології на службу людям, а саме для простих громадян, юридичних осіб приватного права, громадських об'єднань без статусу юридичних осіб. Запитувачі матимуть легкий, інтуїтивно зрозумілий, швидкий доступ до достовірної та актуальної публічної інформації, якою володіють органи державної влади та форми власності так і органи місцевого самоврядування. І що важливо, що буде реальна можливість і інструмент для контролювання діяльності органів влади. Таким чином кожен зможе реалізувати свої конституційне право на інформацію, не зважаючи на рівень комп'ютерної грамотності, використовуючи інтуїтивно зрозумілий інтерфейс, скористатись ним і буде в курсі всього, що робиться, які рішення приймаються, і що державі відомо про кожного з нас.

Це одним важливим моментом є і те, що не дивлячись на форму запиту (письмове звернення чи через веб-сторінку) розглядатись звернення запитувача буде щонайменше від п'яти до двадцяти робочих днів рахуючи від дня подачі його. Строк розгляду залежить в основному від об'єму запитуваної інформації чи кількості даних серед який такий пошук потрібно провести. Є лише один вийняток коли термін розгляду запиту може бути скорочено до сорока восьми годин – це коли запит стосується даних які стосуються напряму життя та свободи запитувача, також сюди можна віднести стан навколишнього середовища, якість побутових предметів, продуктів харчування, коли йдеться про катастрофи, аварії, небезпечні природні явища, різні надзвичайні ситуації, які трапляються, трапились чи можуть трапитись і несуть загрозу безпеці людей. Важливо також зазначити, що запитувач отримує інформацію стосовно свого запиту повністю

безкоштовно, відшкодовуючи лише, в разі необхідності, вартість розхідного матеріалу для друку та копіювання при необхідності виготовлення копії від десяти сторінок [].

В цілому хочеться сказати, про норму Закону «про обов'язкове оприлюднення інформації на web-сайтах: її сміливо можна назвати революційною, і такою, що вперше було закріплено в українському законодавстві. Саме завдяки їй в найближчому майбутньому може так статись, що ми побачимо принципово новий спосіб влади «говорити» зі своїми платниками податків.

Законом було зобов'язано усіх офіційних осіб, які наділені повноваженнями такого характеру, що передбачено статтею 13 Закону України, відповідати за вчасне надання та оприлюднення публічної інформації, яка є відображеною та задокументовану будь-яким із легальних засобів та за допомогою любих дозволених носіїв інформації, яку було отримано чи створено під час процесу виконання суб'єктами відповідних структур чи владних повноважень своїх безпосередніх обов'язків, що передбачено чинним законодавством, або ж яка є у їх розпорядження чи в межах доступу [].

Однак, важливим моментом є і те, що Закон чітко встановлює, що призначена на оприлюднення інформація є відкрита, а виконання його в сенсі доступу до неї реалізується через систематичне та оперативне її оприлюднення через офіційні друковані виданнях, офіційними web-сайтами через глобальну мережу, інформаційними стендами чи якимось з інших способів, чи надаючи безпосередньо у відповідь на запит.

Зрозуміло, що не будь-яку інформацію може бути оприлюднено повністю чи навіть частково. Якщо доступ обмежено, то тут вступає в дію Закон про таємницю і такі дані є конфіденційними, такими що представляють таємницю чи службового користування. Доступ до такої інформації регламентовано Законом в в рамках цієї роботи не розглядається [].

Для того, щоб релевантність такого запиту була якнайвищою, важливо, щоби він містив наступні інформацію:

1 – ПІП запитувача (в окремих випадках найменування), його поштова адреса чи адреса електронної пошти, при наявності контактний номер запитувача чи його вповноваженої особи;

3 – вид чи узагальнено опис інформації, яку потрібно отримати, назва, реквізити або якщо можливо ж суть (зміст) документу, стосовно кого здійснено запит;

4 – якщо запит подано у письмовій формі, то його потрібно завізувати підписом та поставити дату.

Таким чином, будь-який громадянин України має право отримати всю, дозволену законом України достовірну та якісню інформацію швидко, без шкоди здоров'ю, легко, і без зайвих зусиль та зайвих затрат часу та сил.

1.2 Аналіз вимог до програмної системи

На даний час в системі органів місцевого самоврядування в Україні не існує єдиної централізованої системи електронного документообігу між владою та громадянами. Частково оприлюднення інформації відбувається на сайтах рад, та не усі ради мають свої сайти тай функціонал на цих сайтах що є, визначає рада, тому склалась така ситуація що на одних сайтах функціонал для громадського контролю є, а на інших немає - сайт носить лише інформаційний характер із розміщенням на ньому загальної інформації про раду.

З точки зору раціональності використання бюджетних коштів шлях коли кожна із рад створює свій окремий сайт і потім витрачає кошти на його підтримку є недоцільним. Така система має цілий ряд недоліків, зокрема:

- великі фінансові витрати;
- не усі рівні органів місцевої влади можуть собі дозволити профінансувати розробку та підтримку якісної системи документообігу;
- недостатня функціональність;

– відсутність державного контролю та уніфікованої моделі функціоналу сайту;

– надзвичайно дорогий і трудомісткий процес додавання нових функцій та модернізації систем(оскільки зміни потрібно проводити на кожному сайті окремо);

– складність пошуку.

Якщо говорити про нераціональне використання бюджетних коштів слід також згадати, про великі кошти які щороку витрачаються органами місцевого самоврядування на поштові послуги та витрати що пов’язані із друком різного роду документів які потім відправляються органам місцевого самоврядування нижчого рівня.

Також великою проблемою є те, що оприлюднення рішень сільської та селищної ради взагалі не відбувається оскільки сайтів там просто-напросто не існує і про прийняття рішень наприклад про виділення та роздачу землі ніхто навіть і не здогадується, так само як і не здогадується про вміст бюджету що прийняла рада оскільки він ніколи ніде не публікується.

При проведенні адміністративно-територіальної реформи коли фізична відстань між громадянами та владою збільшиться, а влада на місцях отримає більше повноважень та більший бюджет проблема відсутності централізованої системи документообігу постане ще гостріше.

1.2.1 Формулювання та постановка задачі

Аналіз предметної області та існуючої нормативно-законодавчої бази дав інформацію про те, які задачі постають перед розроблюваним системним продуктом та яким основним функціоналом він має бути наділений, зокрема система повинна відповідати наступним вимогам:

– об’єднання сайтів усіх органів місцевого самоврядування на одному порталі;

– можливість ієрархічного створення сайтів для кожної із ланок органів місцевого самоврядування засобами порталу;

– додавання по замовчуванню єдиного уніфікованого загальнодержавного функціоналу при створенні сторінок рад, до цих функцій відноситься:

1) функція подання електронної заяви про доступ до публічної інформації із контролем вчасності надання відповіді;

2) функція подачі петиції із підтримкою усього необхідного функціоналу(подача петиції, голосування за петицію, інформування влади про петиції що набрали необхідну кількість голосів, механізм надання відповіді влади);

3) функціонал для інтернет оприлюднення усіх офіційних документів рад із категорійним сортуванням.

4) функція відправки повідомлень – таким чином буде налагодити оперативний звязок та можна обмінюватись документами між органами місцевої влади різних рівнів

– перелік обов’язкового функціоналу з часом може бути доповнений, при прийнятті відповідного рішення, тому у системі повинен бути продуманий механізм додавання обов’язкового функціоналу без необхідності ручного редагування вмісту кожного сайту рад(вміст по замовчуванню обновлятиметься автоматично на усіх сайтах одночасно);

– захист системи від несанкціонованої реєстрації та несанкціонованого доступу;

– гнучкість системи;

– зручна функція пошуку необхідної ради;

– простий інтуїтивно зрозумілий інтерфейс орієнтований та продуманий для користувачів із різним рівнем володіння комп’ютером.

Таким чином, було визначено що одними із основних пріоритетів розроблюваного програмного продукту повинна бути зручність та можливість

подальшої модернізації. Подальша модернізація є важлива оскільки шлях українських органів місцевого самоврядування в електронному аспекті лише починається і з кожним роком перелік послуг які надаватимуться сайтами рад лише збільшуватиметься, тому довговічність подібної системи буде напряму залежати від простоти і кількості ресурсів які необхідно буде затратити на подальшу модернізацію системи.

Оскільки програма буде взаємодіяти з базами даних, то також є важливим забезпечення надійного з'єднання програми з БД, та передбачення цілого ряду запитів до неї, та обробник виключних ситуацій що можуть виникнути, програма повинна запобігати спотворенню інформації що відображається.

1.2.2 Пошук актантів та варіантів використання

У результаті детального аналізу предметної області та вимог до програмного забезпечення було виявлено основних актантів , зокрема до них відноситься:

- адміністратор сайту;
- гість.

Дані актанти я актуальними при реалізації системи у теперішньому вигляді, вигляді окремих сайтів рад, але оскільки ми реалізуємо дану систему як централізовану і загальнодержавну, то це обов'язково призведе до появи цілого ряду нових завдань пов'язаних із адмініструванням порталу і замість адміністратора сайту буде головний адміністратор порталу.

На тому ж рівні що й адміністратор порталу знаходитиметься працівник контролюючих органів який контролюватиме роботу працівників ради вищих інстанцій(рівень обласних рад).

У нашій системі зникне необхідність у адміністраторі сайту ради, оскільки функціонал системи буде реалізовано максимально просто і на місцях систему, а

зокрема сайти ради обслуговуватиме звичайний працівник ради, чи то заступник чи секретар. Відповідно до функціоналу порталу працівник ради вищої інстанції також володітиме засобами для початкової реєстрації рад нижчих інстанцій, та інструментами які допомагатимуть контролювати дотримання часових рамок при відповіді на звернення громадян із запитом про доступ до публічної інформації та контроль розгляду петицій які набрали необхідну кількість голосів.

Працівник ради нижчого рівня володітиме таким самим функціоналом як і попередній актант, лише буде позбавлений контролюючих функцій оскільки його рада є радою найнижчого рівня органів місцевого самоврядування України.

Користувачі системи які ще не пройшли процедури реєстрації або не ввійшли в систему, це ще один вид користувачів – гість.

Таким чином після детальнішого аналізу предметної області було виявлено наступні види користувачів та варіанти використання якими кожен із них володітиме:

– Головний адміністратор порталу(див рис.1.1).

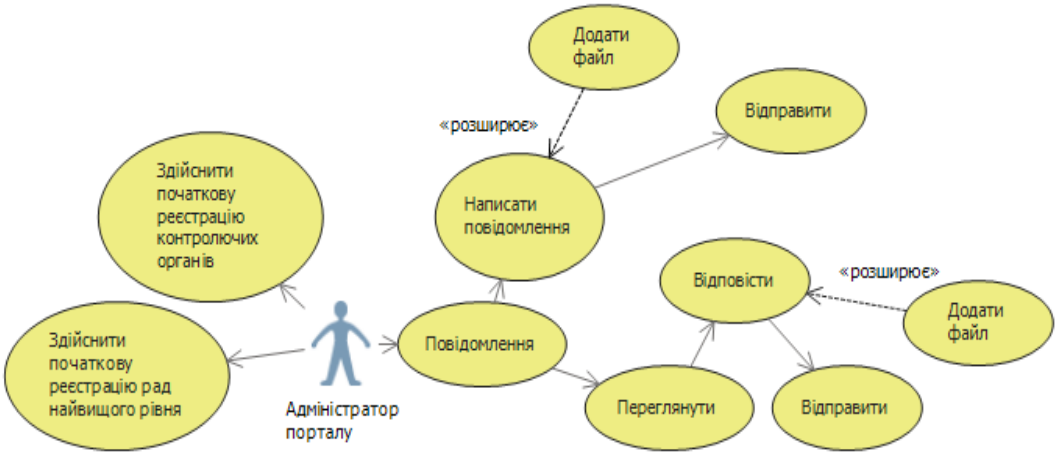

Рисунок 1.1 – UML діаграма варіантів використання
для адміністратора порталу

користувач наділений правами по початковій реєстрації обласних рад, та контролю роботи системи діаграма варіантів використання виглядатиме наступним чином;

– Працівник контролюючих органів – користувач наділений функціями контролю за органами місцевого самоврядування найвищого рівня(обласні ради) у нього також буде можливість здійснювати розсилку офіційних документів усім органам влади. (див рис.1.2);

Рисунок 1.2 – UML діаграма варіантів використання
для працівника вищих контролюючих органів

– Працівник ради вищої інстанції – користувач наділений найбільшим функціоналом з поміж усіх користувачів системи. Зокрема він володіє функціоналом для перегляду та розгляду петицій, написанню відповідей на запити про доступ до публічної інформації, здійснює розгляд скарг та пропозицій як від громади так і від органів місцевого самоврядування нижчого рівня. Здійснення завантаження розпоряджень голови та рішень ради можна проводити одночасно із їх розсилкою по радах нижчого рівня. До окремого блоку функцій відносяться контролюючі функції з допомогою яких здійснюється контроль за

діяльністю рад нижчого рівня, зокрема контроль дотримання часових рамок при відповіді на звернення громадян із запитом про доступ до публічної інформації та контроль розгляду петицій які набрали необхідну кількість голосів. Також до його обовязків відноситься здійснення початкової реєстрації рад нижчої інстанції (див рис.1.3);

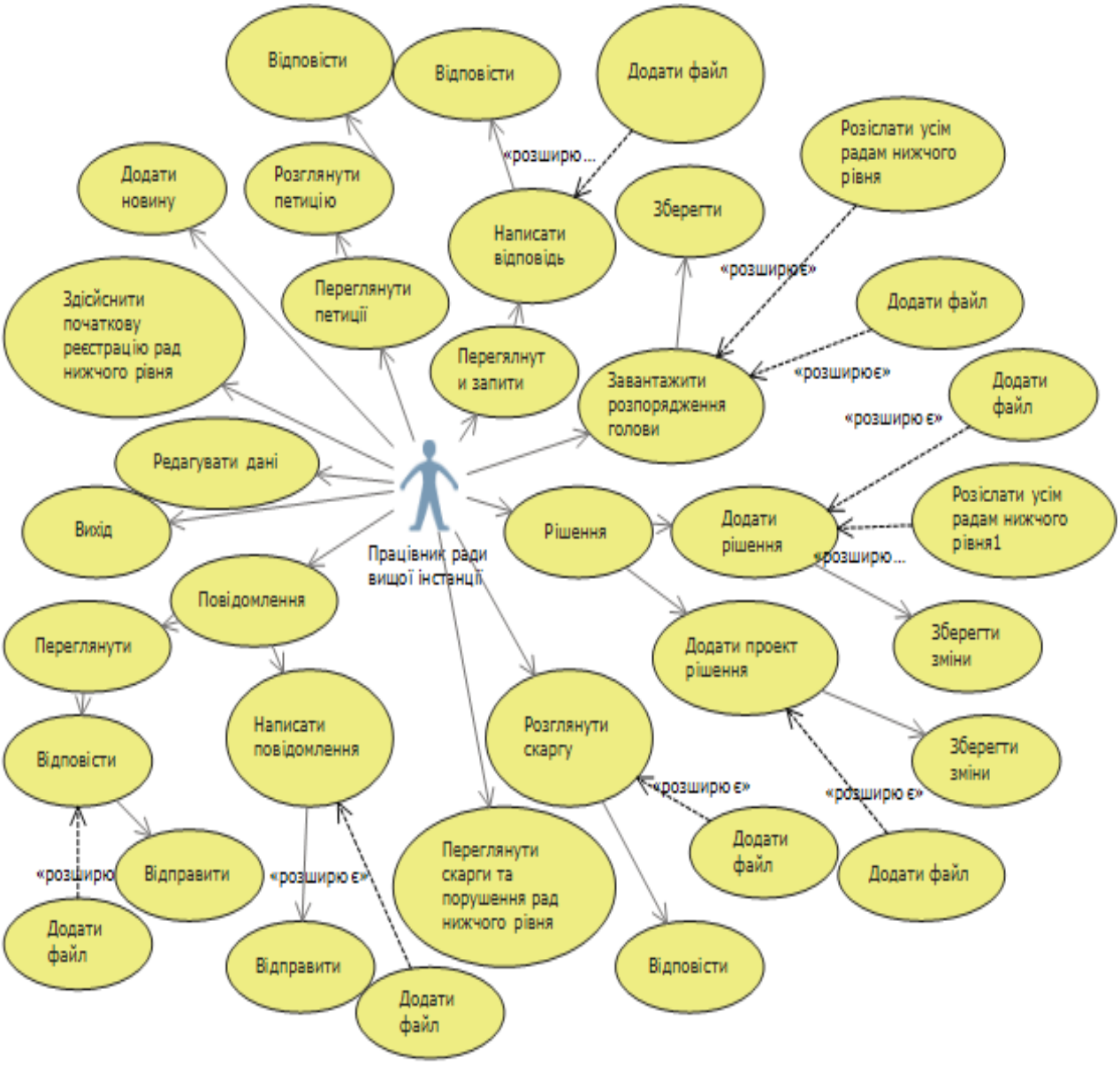

Рисунок 1.3 – UML діаграма варіантів використання
для працівника ради вищої інстанції

– Працівник ради нижчої інстанції – користувач який наділений подібним функціоналом що й працівник ради вищої інстанції, він лише позбавлений контролюючих функцій оскільки він є найнижчою ланкою в ієрархії і відповідно йому немає кого контролювати (див рис.1.4);

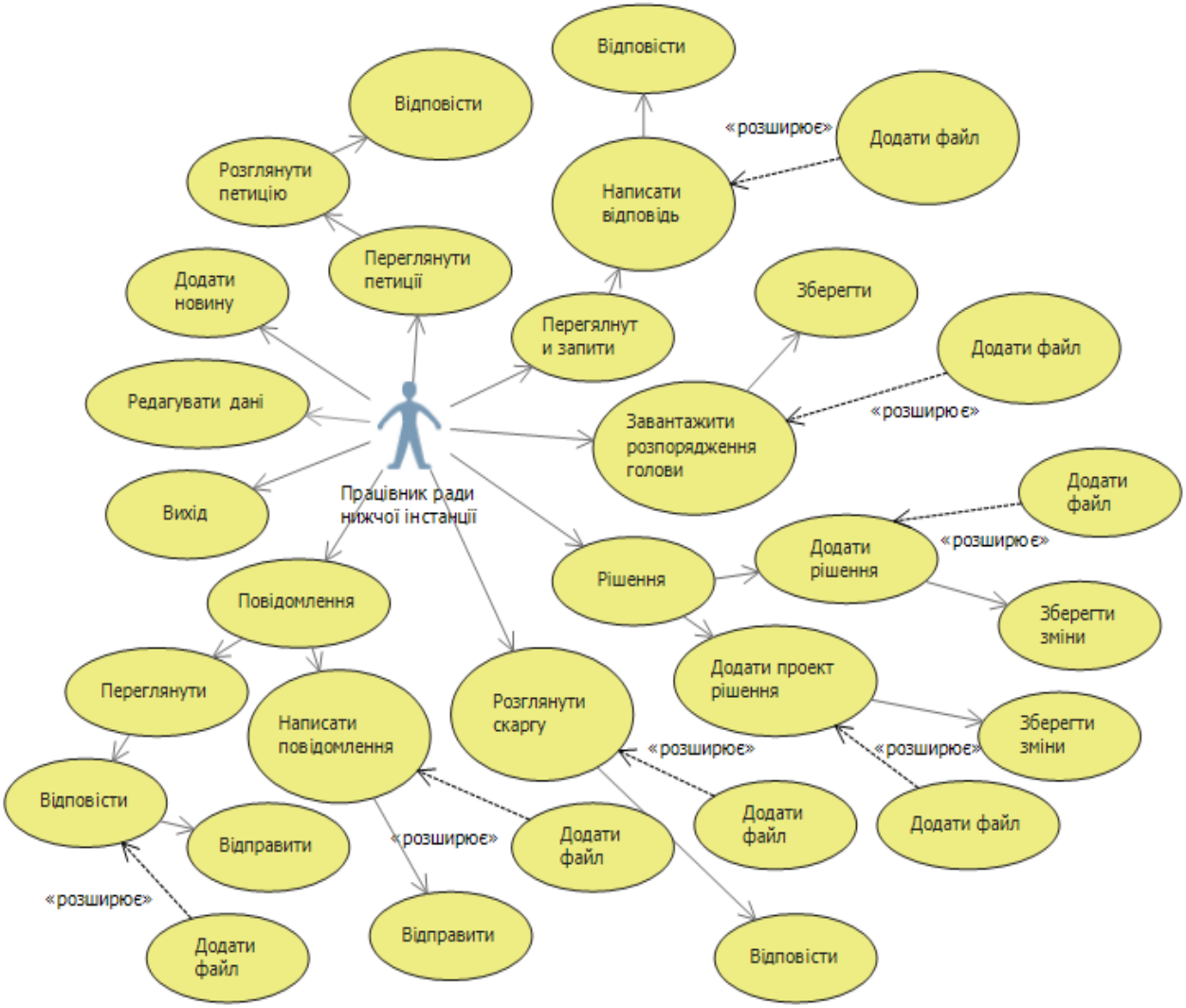

Рисунок 1.4 – UML діаграма варіантів використання
для працівника ради нижчої інстанції

- Користувач гість – актант який суміщає в собі функціонал громадського контролю за діяльністю органів місцевого самоврядування а також з допомогою системи електронних петицій представляє можливості висунення пропозицій до влади. Якщо ж влада якимось чином ігнорує проголосовані петиції чи не спішить

відповідати на запити про доступ до публічної інформації користувач на її дії може написати скаргу, яка одночасно буде надіслана й у органи вищої інстанції. Користувачі які не пройшли процедури входу чи реєстрації можуть провести вхід чи реєстрацію з допомогою функціоналу гостя. Нижче представлена діаграма варіантів використання для користувача гість(див рис.1.5);

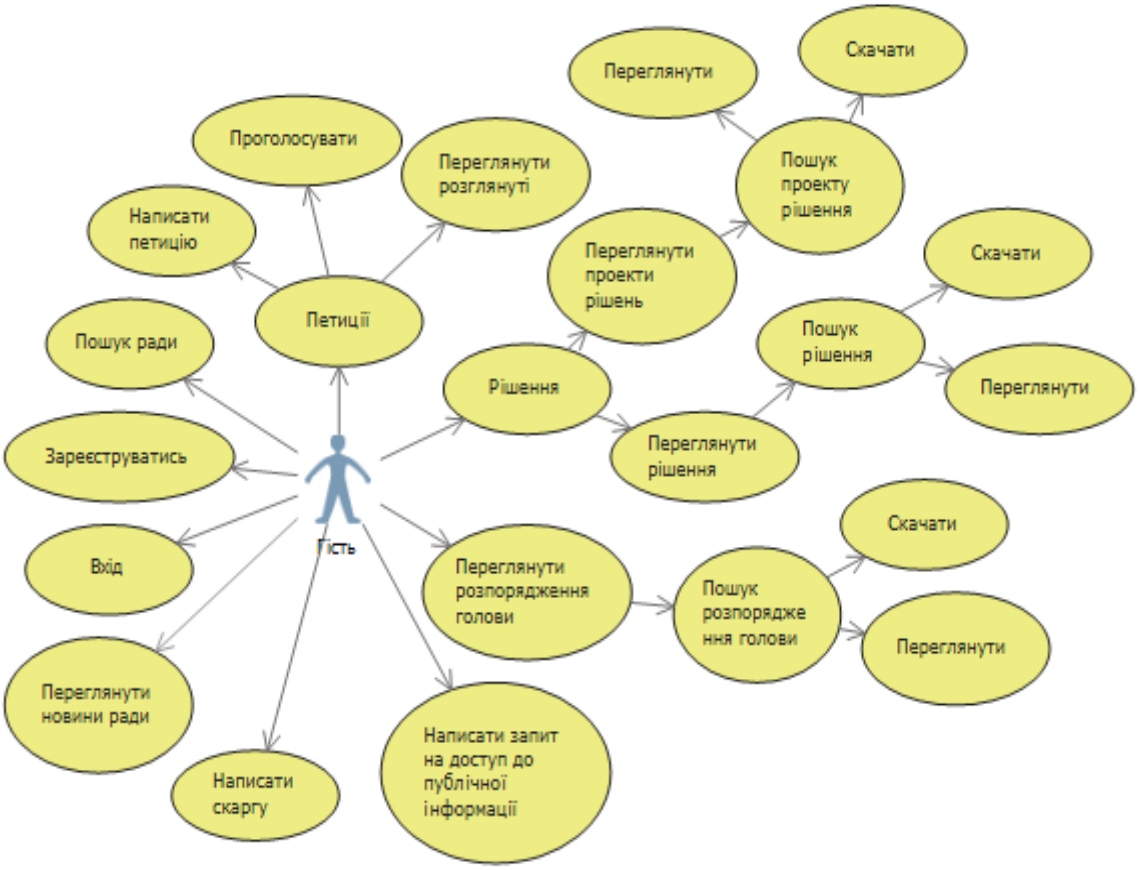

Рисунок 1.5 – UML діаграма варіантів використання
для потенційного запитувача-гостя

1.3 Обґрунтування доцільності вибору моделі бази даних

1.3.1 Вибір та обґрунтування вибору підходу організації інформаційних масивів

На сьогоднішній день відомо про два види підходів які використовуються з метою організації даних інформаційних масивів. До таких можна віднести:

1 - файлова організація даних.

Цим типом передбачається спеціалізація, а також зберігання інформації, яка є орієнтованою зазвичай, на вирішення однієї прикладної задачі. Файлову організацію забезпечує прикладний програміст. Такий тип організації дає можливість досягати високу швидкість обробки інформації, проте, для нього характерними є цілий ряд недоліків, які суттєво впливають на кінцевий результат. Характерною рисою файлового підходу організації впорядкування даних можна назвати вузьку спеціалізацію як програм, які піддаються обробці, так і файлів для зберігання даних. Саме це може слугувати причиною значної надлишковості, оскільки різними системами виконується збереження тих самих елементів. Так як керування можуть виконувати різні особи чи групи осіб, то є відсутньою можливість виявлення порушень суперечливості інформації, яка зберігається, виникає так звана неузгодженість даних. Розроблювані файли призначені для спеціалізованих програм прикладного характеру не рекомендовані до використання з метою задоволення запитів від користувачів, якими перекриваються дві чи й більше областей. Крім цього, такою формою організації даних як файлова, в силу наявних відмінностей в структурі записів та форматів передачі даних, не забезпечується реалізаціяя багатьох запитів інформаційного типу навіть у тих випадках, в який усі необхідні елементи даних знаходяться у наявних файлах.

В результаті, враховуючи вище сказане, і виникла гостра потреба у відокремленні даних від їх описів, визначення такого типу організації зберігання даних, який передбачає облік існуючих зв’язків поміж ними. Це б дало змогу

використати ці такі дані для багатьох застосувань причому одночасно. Вважаю, що саме наведені причини і послугували появі бази даних.

2 – організація даних у вигляді структур - баз даних.

Під базою даних розуміють певним способом організовану сукупність даних, які відображають стан об’єктів певної предметної області і зв’язків поміж ними. В основі такої організації лежить певна структура.

До відмінних від інших типів організації даних, до рис притаманних базам даних варта віднести принцип збереження даних, суть якого у тому, що збереження даних виконується разом з їхнім описом, при чому опис даних в програмах прикладного типу не міститься. Дані, які не залежать від програми користувача, називають метаданими.

Основні переваги при виборі типу організовування інформації в формі (структур) типу бази даних полягають у:

– використанні багато разів;

– економії витрат пов’язаних зі створенням і введенням;

– зменшенні надлишковості;

– швидкості обробки;

– простоті і зручності при внесенні змін до бази даних;

– логічній та фізичній незалежності даних від програм прикладного значення.

Зрозуміло і відомо факт того, що в базі даних містяться не лише самі дані, а й їх опис. Власне з цієї причини інформація, що стосується форми збереження даних вже не прихована в зв’язці «файл-програма». Її явним чином задекларовано в базі. В загальному ж варта зазначити, що базу даних орієнтовано на інтеґровані запити і нею можна забезпечити потреби багатьох користувачів. Це є відмінністю від файлового підходу, в якому орієнтація здійснена на одну програму.

1.3.2 Система управління базами даних для автоматизованої системи управління. Обґрунтування вибору MicrosoftSQL Server

Для системи управління базами даних автоматизованої системи керування було обрано MicrosoftSQL Server. Таке рішення вважаю обґрунтованим та його було прийнято з метою щоби кілька користувачів мали можливість взаємодіяти із базами даних.

Зв'язок кінцевого користувача (-ів) та прикладних програм з базами даних, здійснюється також саме через СКБД (рисунок.1.6), яка служить як інтерфейс між користувачами та базами даних. В якості користувачів баз такого типу можуть виступати як визначені фізичні особи так і слугувати прикладні програми.

Рисунок 1.6 – Роль системи управління БД[]

Якщо говорити про систему управління баз даних, то її можна вважати за комплекс, куди входять програмні та мовні засоби призначення і загального і спеціального напрямків, необхідних і для створення баз такого типу, і їх підтримки в реальному до актуального стані. Також не варта опускати і функцію маніпуляції даними, так само як і організації доступу до даних користувача в умовах визначеної технології для опрацювання,обробки даних.

До основних функцій СУ базами даних можна віднести:

– управління даними які зберігаються на дисках (в носіях зовнішньої пам'яті);

– керування даними в пам`яті яка являється оперативною;

– журналізація змін та відновлення баз даних після виникнення різмонамітних збоїв при роботі;

– підтримка мов під бази даних (зокрема, мови призначені для визначення даних, мови маніпулювання даними, тощо).

Для виконання означеного в дипломній роботі на здобуття освітнього ступеня «магістр» завдання було визначено, що вибір слід зупинити на системі керування БД – MicrosoftSQL Server, як – комерційної системиуправління базами даних, що розповсюджується корпорацією Microsoft. Мовою, якою користуються для формування запитів є TransactSQL, яку було створено спільно з Microsoft-том та Sybase-ою. Transact-SQL реалізацовано відповідно до стандарту ANSI/ISO, в сенсі використання мови запитів, яка є структурованою(SQL) з розширенням. Обґрунтованим таке застовування, коли мова йде як про невеликі і середні за розміром бази даних, так і коли потрібно використати бази даних великі, зокрема коли йдеться про масштаб цілого підприємства чи крупної організації [3].

На MicrosoftSQL Server я зупинив свій вибір саме через цілий цілий ряд переваг та можливостей такої СКБД, які росто відсутні в інших СКБД, які можна назвати альтернативними. До цих таких переваг можна віднести те, що:

– в SQL Server-і міститься по новому спроєктований процесор для запитів, який може забезпечити як підтримку баз даних значного за розміром об’єму, а також це стосується і обробки складних запитів;

– використовуються складні індекси, нових алґоритмів хешування і злиття, множинні триґери, а також опрацювання розподілених і паралєльних запитів;

– збільшення до 8 КБ розміру сторінок. Це дасть можливість швидко вилучати дані, дозволяє використання рядків та стовпців більшого розміру. Це дає можливість реалізувати ефективне збереження складних,докладних даних;

– автоматизація великої кількості рутинних завдань адміністрування. Алґоритм управління пам’ятю і блокування адаптується достатньо динамічно, розміри файлів автоматично збільшуються і скорочуються. Крім цього, засоби автоматичного налаштування динамічно налаштовують алґоритми по використанню ресурсів в залежності від встановленого робочого навантаження;

– підтримка лінгвістичного пошуку, що дозволяє створювати спеціальні індекси ключових слів або фраз для визначених (вказаних) стовпців або таблиць.

1.3.3 Виявлення та аналіз основних сутностей предметної області

Зазвичай база даних, містить велику кількість різних типів сутностей. Кожен із типів сутності має унікальний набір атрибутів і також кожна із сутностей – свої (власні) значення (для кожного з атрибутів). Всі ці властивості визначаються атрибутами сутності,- це властивості, якими буде володіти об’єкт який розглядається.

База даних ще володіє певним набором відношень. Відношення – це логічно побудований взаємозв’язок поміж сутностями в базі даних. Відношення задаються за допомогою ключів.

Самі ж таблиці мають певним чином організовуватись, що забезпечує найбільш оптимальне використання інформації, зокрема це стосується усунення повторень. Для таких цілей використовують нормальну форму а щоб привести таблиці до нормальної форми застосовують нормалізацію відношень.

В результаті маємо отримати такий склад атрибутів відношень в базі даних, який відповідатиме наступним вимогам, а саме: поміж атрибутів не повинно бути не бажаних залежностей функціонального типу, групування для атрибутів повинне забезпечити мінімально можливе дублювання даних, їх обробка та поновлення повинна здійснюватись без будь-яких ускладнень та аномалій.

Отже, після дослідження предметної області було виявлено наступні сутності:

– користувачі;
– ради;
– документи

Беручи до уваги на наведені вище особливості спроектованої автоматизованої системи управління навчальним процесом у загальноосвітньому навчальному закладі проведено проектування таблиць БД.

У таблиці 1.1 представлено структуру таблиці для збереження користувачів системи.

Таблиця 1.1 – Структура таблиці користувачів системи

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Ідентифікатор користувача
Role	nvarchar	128	Роль в системі
Email	nvarchar	128	Email користувача
Login	nvarchar	128	Логін користувача(унікальне ім’я для входу)
Password	nvarchar	128	Пароль

Таблиця називатиметься users, міститиме первинний унікальний ключ користувача – id, роль користувача в системі(головний адміністратор порталу, працівник контролюючих органів, працівник ради вищої інстанції, працівник ради нижчої інстанції) e-mail адреса користувача, логін користувача(повинен бути унікальним, оскільки він використовуватиметься для ідентифікації користувача при вході) і пароль.

В таблиці 1.2 наведено структуру таблиці БД в якій зберігатимуться дані про орган самоврядування, на прикладі ради.

Таблиця 1.2 – Структура таблиці з інформацією про раду

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Збереження первинного ключа
RegistrationCode	nvarchar	128	Ключ реєстрації
UserId	int	11	Зовнішній ключ, id із таблиці користувачів
NameSenate	nvarchar	128	Назва ради
Region	nvarchar	128	Область
District	nvarchar	128	Район
NameCityOrVillage	nvarchar	128	Назву населеного пункту

Таблиця матиме назву Senates і міститиме унікальний ідентифікатор, реєстраційний ключ, зовнішній ключ користувача, назву ради, область в якій вона розміщується, район, та назву населеного пункту.

В таблиці 1.3 наведено структуру таблиці БД в якій зберігатимуться дані про документ.

Таблиця 1.3 – Структура таблиці з інформацією про документ

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Ідентифікатор документу
SenateId	int	11	Зовнішній ключ, id ради
SesionId	int	11	Зовнішній ключ, id сесії
NameDocument	nvarchar	128	Назва документа
TypeOfDocument	nvarchar	128	Тип документу
Way	nvarchar	128	Розміщення документу
DateForReview	datetime	7	Дата завантаження документу для ознайомлення
DateEnactment	datetime	7	Дата рішення

Таблиця матиме назву Documents і міститиме унікальний ідентифікатор, зовнішній ключ ради, зовнішній ключ сесії, назву документа, тип документа(розпорядження, закон, проект рішення, рішення або інший), шлях розміщення, дата завантаження документу для ознайомлення, та дата прийняття рішення.

В таблиці 1.4 наведено структуру таблиці БД в якій зберігатимуться дані про петиції.

Таблиця 1.4 – Структура таблиці з інформацією про петиції

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Ідентифікатор петиції
SenateId	int	11	Зовнішній ключ, id ради
DateOfCreation	datetime	7	Дата створення
Name	nvarchar	128	Назва петиції
TextPetition	nvarchar	max	Текст петиції
Status	nvarchar	128	Статус петиції
Answer	nvarchar	max	Відповідь

Таблиця матиме назву Petitions, зберігатиме в собі унікальний ідентифікатор, зовнішній ключ ради якій адресується петиція, дата створення петиції, назва петиції, її текст, статус(триває збір підписів, на розгляді, з відповіддю), та текст самої відповіді.

В таблиці 1.5 наведено структуру таблиці БД в якій зберігатимуться дані про підписників петицій.

Таблиця 1.5 – Структура таблиці з інформацією про підписників

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Ідентифікатор підписника
PetitionId	int	11	Ідентифікатор петиції
Email	nvarchar	128	Емейл
Date	datetime	7	Дата
Name	nvarchar	128	Ім’я
LastName	nvarchar	128	Прізвище

Таблиця матиме назву – Subscriptions, в ній зберігатиметься основна інформація про підписників на петиції, зокрема id підпису, ідентифікатор петиції, email підписника, дата підпису, імя та прізвище.

В таблиці 1.6 наведено структуру таблиці БД в якій зберігатимуться дані про підписників новини рад.

Таблиця 1.6 – Структура таблиці новин

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Ідентифікатор новини
SenateId	int	11	Зовнішній ключ, id ради
Date	datetime	7	Дата
Name	nvarchar	128	Назва новини
TextNews	nvarchar	max	Текст новини

Таблиця називатиметься – News, в ній зберігатиметься інформація про новини рад, зокрема таблиця міститиме такі дані – id новини, ідентифікатор ради до якої відноситься новина, дата оприлюднення новини, назва новини та сам текст новини.

В таблиці 1.7 наведено структуру таблиці БД в якій зберігатимуться дані про підписників новини рад.

Таблиця 1.7 – структура таблиці сесій

Назва поля	Тип поля	Довжина	Призначення
Id	int	11	Ідентифікатор сесії
SenateId	int	11	Зовнішній ключ, id ради
Date	datetime	7	Дата
Name	nvarchar	128	Назва сесії

Таблиця називатиметься – Sesions, в ній зберігатиметься інформація про сесії різного рівня рад, зокрема таблиця міститиме такі дані – id сесії, ідентифікатор ради до якої відноситься сесія, дата коли відбудеться сесія, та назва сесії.

1.3.4 Представлення схеми реляційної бази даних, модель побудови

Реляційною базою даних (далі базою даних), ввадають таку модель впорядкування інформації, яка базується на структурі типу реляційної моделі структуризації даних.

Модель впорядкування даних такого типу базується на застосуванні математичних принципів, в основі яких лежить теорія множин і логіка предикатів. Якщо мову вести про принципи формування системи баз даних за реляційним типом, то їх пропоную сформулювати в такому вигляді:

– якщо говорити про концептуальний рівень даних, то їх представляють у формі впорядкованої структури, яка визначаться як стрічки і стовбці, які поєднуються між собою відношеннями;

– якщо говорити про значення, то вони є скалярного типу, тобто будь-яка стрічка чи стовбець для будь якого відношення має лише одне і лише одне значення;

– якщо говорити про виконувані операції, то їх виконання здійснюється над цілими відношеннями, а результатом їх виконання також є ціле відношення. Такий принцип називається замиканням [4].

Можна сказати і по іншому, тоді реляційною базою даних називають таку базу даних такого типу, яку користувач сприймає у формі (вигляді) набору нормалізованих відношень, при чому різного ступеня. Якщо ж говорити про мету нормалізації, то маємо справу з усуненням недоліків усієї структури баз даних. В результаті застосування такого підхіду можна втратити оптимальності у вигляді надмірності в даних. Це в свою чергу може потенційно призвести до різноманітних аномальностей та порушень в цілісності даних. Теоретиками баз даних (реляційного типу) під час процесу розвитку цієї теорії було виявлено та описано типові приклади таких порушень у вигляді надмірностей та перелічено способи їх усунення через оптимізацію.

У розділі 11111 було визначено та описано сутності та відношення між ними. Це дає мені право і змогу на цьому етапі визначитись з ER моделлю бази даних та сформувати її. Моделлю сутність+зв'язок або так званою ER-моделлю є модель даних, яка дає змогу описати концептуальні схеми використовуючи для цього узагальнені конструкції блоків. ER-модель є мета-моделлю даних, засобом опису моделі (-ей) даних. Існує цілий ряд моделей призначених для представлення, проте одним з найзручніших інструментів представлення даних уніфікованого типу, який є незалежним від програмного забезпечення яке його реалізує, є модель типу сутність та зв'язок, яку представлено на рисунку 1.7.

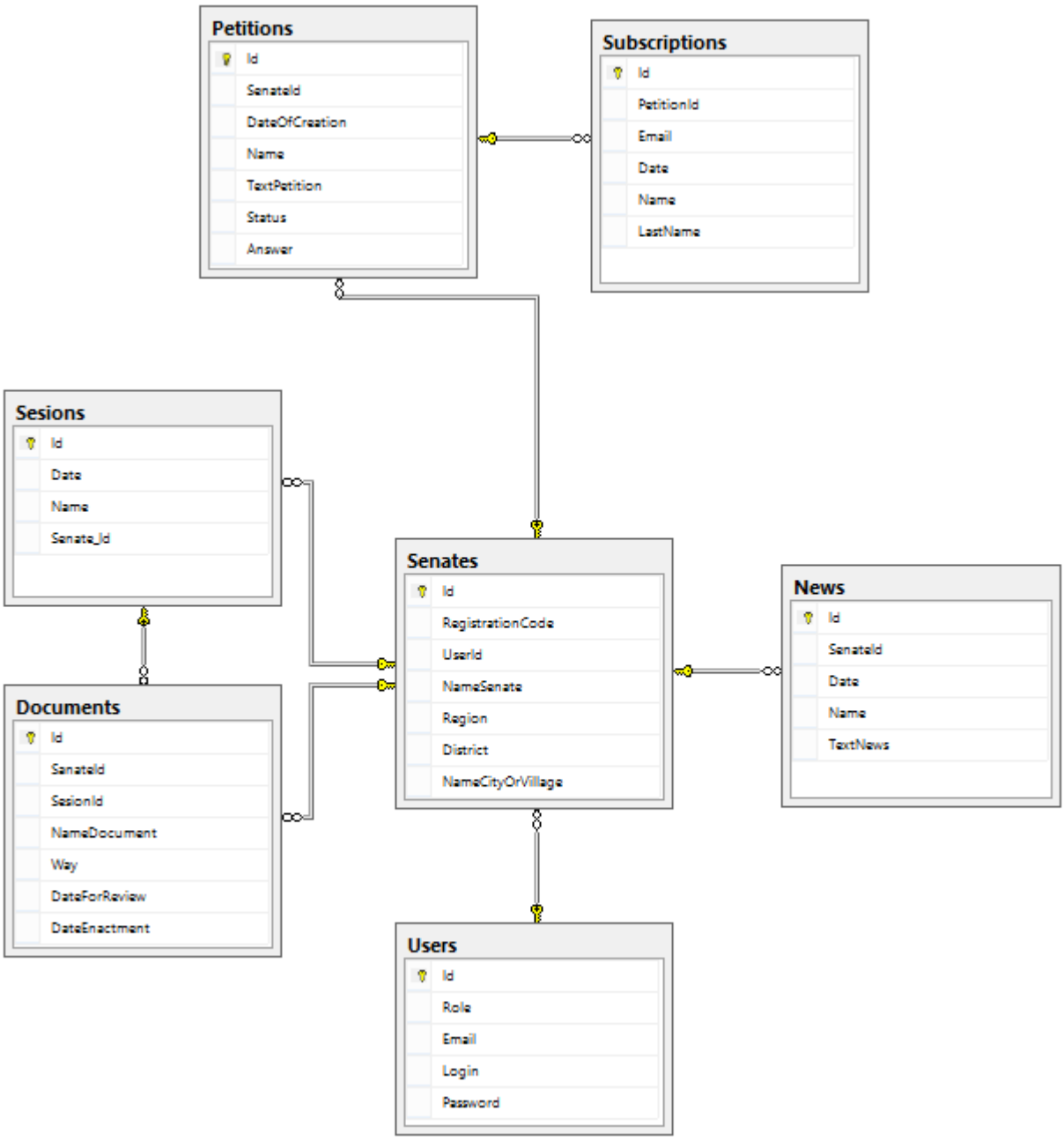

Рисунок 1.7 – ER модель бази даних

1.3.5 Проєктування бази даних централізованої системи електронного документообігу для органів місцевого самоврядування на її фізичному рівні

Оскільки уже виявлено основні сутності та побудовано реляційну модель даних, наступним етапом проектування буде створення БД безпосередньо у самій СКБД(рисунок 1.8).

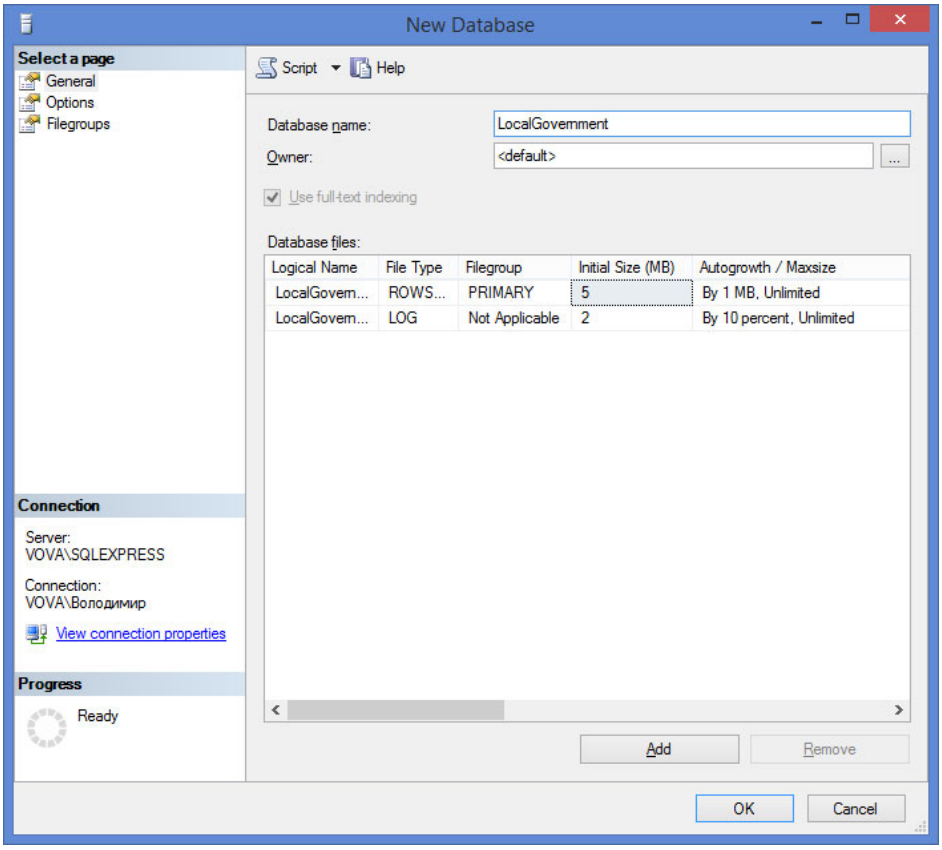

Рисунок 1.8 – Створення бази даних

1.3.6 Вибір підходу роботи з даними в Entity Framework

Існує три підходи роботи з даними в Entity Framework : Database First, Model First, Code First(рисунок 1.9).

– Database First – в випадку існуючої бази даних Entity Framework може автоматично створювати модель даних, що складається із класів і властивостей, що відповідають об’єктам бази даних(такими, як таблиці і стовбці). Інформація про структуру бази(store schema), модель даних(conceptual model) і мапінг їх один на одного що міститься в XML в .edmx. Visual Studio представляє графічний дизайнер Entity Framework, з допомогою якого можна переглядати і редагувати .edmx.

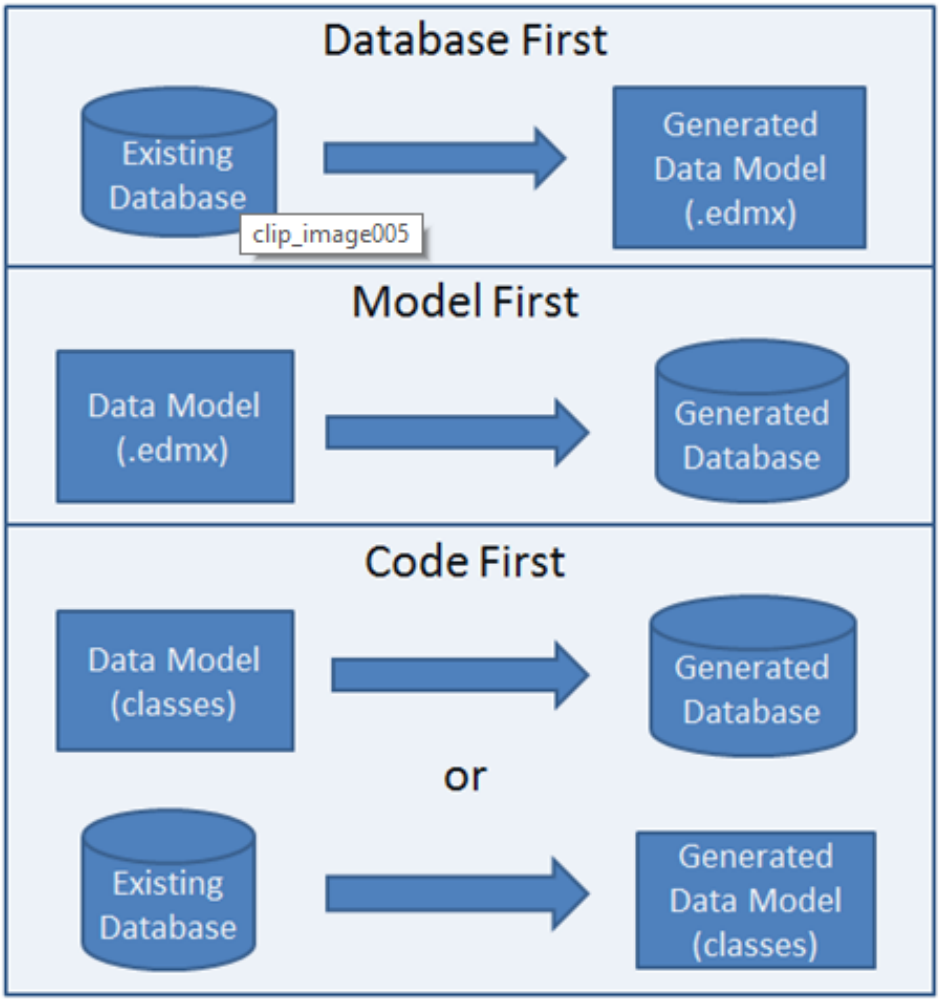

Рисунок 1.9 – Підходи роботи з даними в Entitty Framework []

– Model First – якщо бази даних немає, можна почати із створення моделі даних, використовуючи дизайнер Entity Framework Visual Studio. Після закінчення робіт над моделлю дизайнер згенерує DDL(data definition language) – код для

створення бази даних. В цьому підході для збереження моделі і мапінгу також використовується .edmx.

– Code First в незалежності від існування бази можна вручну написати код класів і властивостей, що відповідають сутностям в базі і використовувати цей код для Entity Framework без використання файлу .edmx. Якщо бази ще немає, Entity Framework може створювати, видаляти або перестворювати її в випадку зміни моделі.

Через цілий ряд достоїнств було обрано Code First для роботи з даними в Entity Framework, оскільки при ще відсутній базі даних це є кращим підходом ніж його аналоги.

Робота із Code First відбувається в наступній послідовності:

– створення класів що визначають модель;

– запуск застосунку;

– CodeFirst API створює нову базу даних або проектує існуючі таблиці з класами;

– заповнює базу даних тестовою інформацією;

– старт додатку.

Даний підхід значно спрощує процес написання додатку який взаємодіятиме із БД, оскільки майже зникає необхідність працювати на пряму із запитами до БД.

1.4 Огляд та обґрунтування технологій, які запропоновано до застосування

ASP.NET MVC це фреймворк від Microsoft, призначений для розробки. Він цікавий поєднанням у собі ефективності та і акуратності архітектури типу MVC, найсучасніших ідей та методів ґнучкої розробки з кращими властивостями для існуючої платформи ASP.NET. Він є альтернативою для традиційних ASP.NET

WebForms, що дає змогу домогтися істотної переваги для усіх, окрім проєктів web-розробки, які є найпростішими та тривіальними.

ASP.NET MVC Framework реалізує MVC патерн і, тим самим, забезпечує значно поліпшений поділ концепцій[3].

MVC є стандартним шаблоном для розробки. Перелічу компоненти, які є складовими платформи MVC:

– моделі, об’єкти яких є частиною програм, за допомогою яких реалізується логіку для домену цих таких програми. Часто стан моделі у базі даних отримується та зберігається за допомогою об’єктів моделей. Для прикладу можу назвати об'єкт Product, за допомогою якого можна отримати інформацію безпосередньоз бази даних, а також взаємодіяти з нею, після чого записати оновлені дані в таблицю ProductsбазиSQL Server. Для малих додатків модель такого типу несе в собі більше концептуальність, чим суть фізичного поділу. Для прикладу, додатком лише зчитується набір даних і здійснюється його відправка в представлення, при цьому характерна відсутність фізичного шару моделі та пов’язаних з ним класами. В такому випадку набор даних переймає на себе роль об’єкту для моделі;

– представлення, які слугують для здійснення відображення інтерфейса додатка. Інтерфейс зі сторони користувача як правило створюють на базі моделі. За приклад можна взяти представлення призначене для здійсненні редагування таблиць Products, в якому містяться текстові поля, списки, які розкриваються та прапорці, значення для яких зумовлено поточним станом об’єкту Product;

– контролери, за допомогою яких здійснюється взаємодія з користувачом, взаємодія з моделлю, так само як і обрання подання, за допомогою якого відображається користувальницький інтерфейс. Додатком MVCпредставлення лише відображаються дані, а контролером обробляються дані, які вводять і відповідають на дії користувача (-ів). Для прикладу, контролером можуть опрацьовуватись строкові значення запитів та здійснюватись їх передача в модель, якаю ці значення можуть використовуватись з метою відправки запитів до бази даних.

ASP.NET MVC визнає важливість отримання чистої, відповідної стандартам розмітки. Його вбудовані методи HTML помічника надають відповідні стандартам вихідні дані[4].

JavaScript є назвою реалізації для стандарту мови програмування ECMA Script від Netscape. Її робота ґрунтується на основі принципів прототипного програмування. Найпоширенішим і найвідомішим застосуванням мови є написання сценаріїв для web-сторінок, проте її також використовують і для впровадження сценаріїв для управління об'єктами, які вбудовано до інших програм [5].

CSS (англійською Cascading Style Sheets, українською каскадні таблиці стилів) є набіром параметрів для форматування, який застосовують до елементів документу для того, щоби внести зміни до їхнього зовнішнього вигляду. Можливість працювати зі стилями здавна включена в розвинених видавничих системах та текстових редакторах. Тим самим це дає змогу одним натиском кнопки зробити текст заданого, заздалегідь встановленого вигляду. Сьогодні це є доступним і для творців сайтів, у випадках коли кольори, розмір тексту та інші параметри зберігають в певному визначеному місці і їх достатньо нескладно «прикрутити» до будь-якого теґу. Ще до однієї з переваг стилів є те, що ними пропонуються значно більше можливостей для форматування, аніж при використанні звичайного HTML[6].

Hypertext Markup Language (скорочено англійською HTML) є простою системою призначеною для створення гіпертекстових документів, які достатньо просто перенести з однієї платформи на іншу. По своїй суті, HTML документами є SGML документи які володіють узагальненою семантикою, яка є придатною для представлення інформації для великої кількості додатків[7]. HTML можна використати для представлень наступного змісту:

- ґіпертекстові новини, пошта, документація та ґіпермедіа;
- меню опцій;
- результати запитів до баз даних;
- прості документи з вбудованими графічними образами;

– гіпертекстовий перегляд існуючих масивів інформації.

SQL (англійською Structured Query Language, українською структурована мова запитів) є мовою для керування базами даних, призначена для використання з роботою баз даних реляційного типу. Сама по собі SQL не є тюринг-повною мовою програмування, але стандарт що в ній використано, дає змогу створювати для процедурні розширення, за допомогою яких можна розширити її функціональність, і тоді вже можна говорити про повноцінну мову програмування.

Мову була створено у 1970-х роках під робочою назвою SEQUEL та призначено для роботи з системою керівання базами даних (СКБД) System R. Через деякий час її було перейменовано в "SQL" з метою уникнення конфліктів між торговими марками. У 1979 році SQL було вперше опубліковано як комерційний продукт Oracle V2[8].

1.5 Проєктування структури головних каталогів та файлової архітектури автоматизованої системи

Відповідно до аналізу структури інформаційних файлів можна попередньо спроектувати структуру та склад каталогів в яких розміщуватимуться файли централізованої системи. Оскільки дана централізована система буде досить складним продуктом із великою кількістю різноманітних файлів доцільним буде розділити усе рішення яке називатиметься LocalGovernment на кілька менших проєктів(рисунок 1.10), зокрема рішення нашої централізованої системи скрадатиметься із трьох основних і двох додаткових проектів, а саме:

– BusinessLogic – один із трьох основних проектів, типу – ClassLibrary(бібліотека класів) в ньому міститиметься уся бізнес логіка рішення;

– Domain– наступний основний проєкт нашого рішення також типу ClassLibrary, в ньому буде визначена модель рішення, зокрема розміщуватимуться бізнес сутності з якими працюватиме рішення;

– LocalGovernmentMvc – останній основний проект нашого рішення має тип ASP.NETMVC 4 WebApplication;

– IntegrationTests – один із двох додаткових проектів, має тип ClassLibrary і служить місцем збереження інтеграційних тестів;

– UnitTests – додатковий проєкт типу ClassLibrary що зберігатиме файли Unit-тестування.

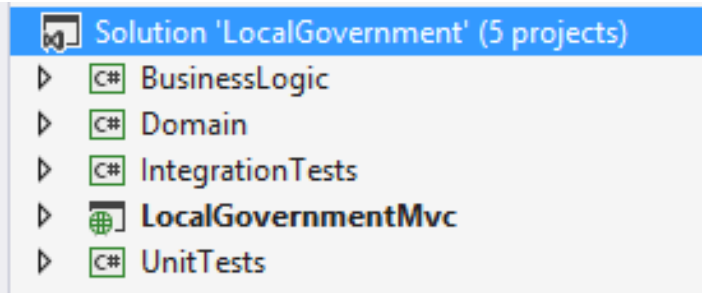

Рисунок 1.10 – Розподіл рішення на проєкти

Далі розглянемо структуру каталогів кожного із основних проєктів окремо. На рисунку 1.11 показано структуру каталогів проєкту BusinessLogic

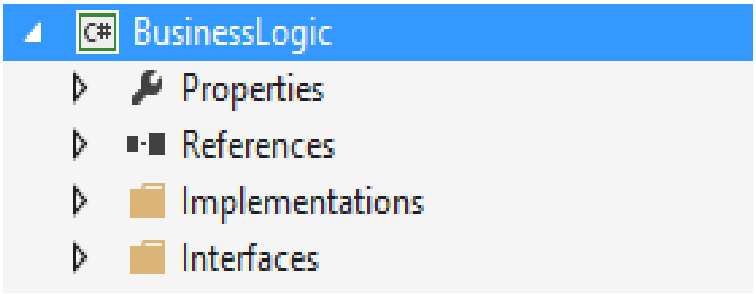

Рисунок 1.11 – Структура каталогів проєкту BusinessLogic

Структура каталогів проєкту Domain (рисунок1.12) окрім каталогів створених середовищем розробки також містить каталог Entities, що міститиме бізнес сутності рішення.

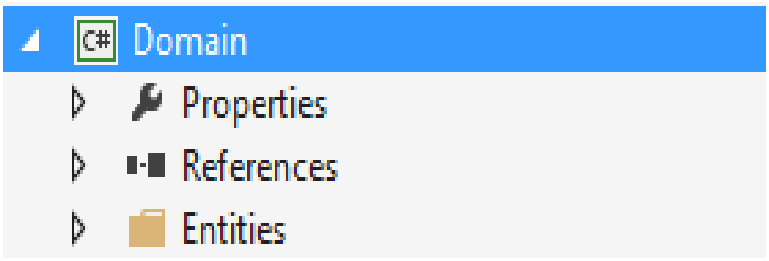

Рисунок 1.12 – Структура каталогів проєкту Domain

Проєкт LocalGovernmentMvc можна вважати одним із основних проєктів нашого рішення, оскільки саме він безпосередньо містить представлення і відповідні контролери. Структура каталогів проєкту LocalGovernmentMvc (рисунок 1.13).

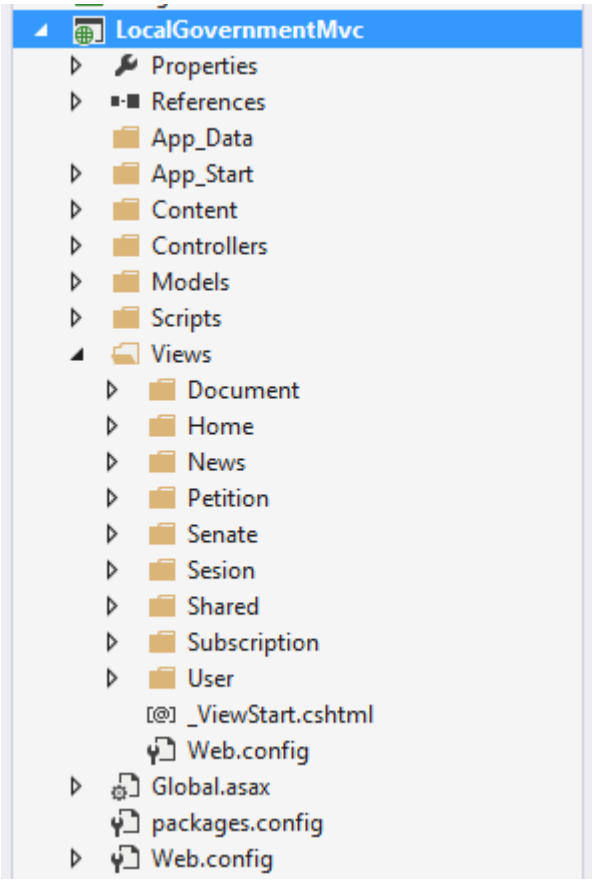

Рисунок 1.13 – Структура папок проєкту LocalGovernmentMvc

Даний проєкт містить наступні каталоги:

– Properties – каталог в якому міститься файл інформації про поточне рішення чи проєкт;

– References – каталог містить посилання на служби та пакети що є задіяні в даному проєкті;

– Add_Data – папка для збереження даних що представляють нашу модель;

– Content – служить для збереження усього статичного контенту нашого проєкту, наприклад файли каскадних таблиць стилів – CSS, теми, відео чи картинки,;

– Controllers – каталог збереження усіх класів контролерів нашого проєкту;

– Images – папка в якій містяться елементи графічного оформлення нашого сайту;

– Models – каталог в якому зберігаються класи моделі проєкту;

– Scripts – служить для збереження JavaScript файлів нашого проєкту;

– View – каталог в якому зберігаються усі представлення проєкту. Оскільки наш проєкт є досить великий папка View розділена на папки представлення відповідних контролерів нашого сайту, зокрема:

1. Document – каталог збереження представлень контролера по роботі із документами;
2. Home – каталог представлення головної сторінки порталу;
3. News – каталог в якому містяться представлення для роботи із новинами органів самоврядування;
4. Petition – каталог збереження представлень які відповідають за роботу із петиціями;
5. Senate – основний каталог в якому знаходяться представлення для реєстрації і редагування даних рад;
6. Sesion – служить для зберігання представлень, що відносяться до сесій;
7. Shared – служить для зберігання представлень майстер сторінок сайту;
8. User – каталог для зберігання представлень які відносять до обновлення даних користувачів порталу.

2 ТЕСТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

2.1 Основні причини виникнення помилок та обгрунтування необхідності проведення тестування

Тестування програмного забезпечення, є логічним і навіть природнім невід'ємним процесом створення якісного програмного продукту. Відому істину про те, що не помиляється лише той, хто нічого не робить, можна поширити і на програмування. Тож допускаю можливість виявлення певної кількості помилок, які можуть виникати на різних етапах створення програмного продукту. З практики ж можу сказати, що джерелом помилки дуже часто можуть бути безпосередньо самі їх створювачі програмісти. Існує такий собі загальний закон для практичного програмування, в якому йдеться про те, що ні одна програма не дасть очікуваного результату при першому спробі трансляції і виконання. Якусь думку стосовно справді реальних причин виникнення помилок, які перешкоджають якісній роботі програми, можна скласти маючи відсоткове співвідношення джерел можливих збоїв, наведені таблично в табл. 2.1.

Таблиця 2.1 – Причини виникнення помилок

Причина помилки	%
Вхідні дані	1
Помилки користувача	5
Апаратура	1
Системне програмне забезпечення	3
Розробка системи	15
Програмування	75

Варта сказати, що робота програміста полягає не просто в написанні ефективного, оптимального програмного продукту, а й в виявленні та

знешкодженні помилок, які в кінцевому випадку негативно впливають на його якість в цілому. Для сучасної практики вивчення програмування в загальному притаманне виконання програмістом лишепершої половини роботи, а вже ту іншу сторону він повинен проштудіювати та освоїти фактично самостійно.

В цілому можна говорити про помилки двох типів, зокрема:

– помилки синтаксичного характеру. Вони можуть виникати в результаті порушень правил використання тієї чи іншої мови програмування. Найчастіше ці помилки виявляють під час компіляції. Їх можна виключити достатньо просто. Навіть у випадку, коли виконувати перегляд тексту програми можна не сумніватись, що компілятор на етапі трансляції виявить помилки і сигналізує засобом відповідних попереджень. Таким чином виявлення помилок здійснюється компілятором, а їх коригування програмістом;

– помилки семантично-логічного типу. Ними можуть провокуватись різноманітні некоректності в обчисленнях та виникати помилки під час виконання так званого «run-timeerror». Як правило, помилки такого типу можна усунути використовуючи програми в яких відповідально підібрані перевірочні дані, для яких правильна відповідь є наперед відомою.

Тестування програмного забезпечення можна назвати технікою, інструментом для контролю за якістю, за допомогою якого здійснюється перевірка відповідності між реальною і очікуваною поведінка ми програмного продукту завдячуючи прикінцевому наборові тестів, які визначаються та обирають за певним чином [9].

Також справедливо буде зауважити і те, що для більшість помилок все таки виникають абсолютно не з безпосередньої вини програмістів. З досвіду свого та й даних оприлюднених великими компаніями випливає, що головною причиною виникнення помилок в програмному забезпеченні є помилки, виявлені в специфікаціях і це показано на діаграмі в частковому співвідношенні як показано на рисунку 2.1.

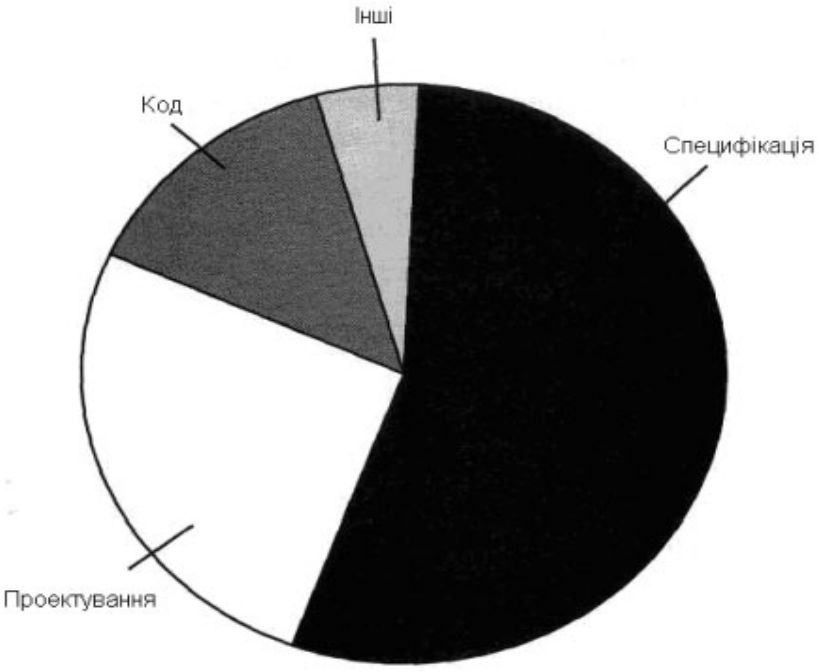

Рисунок 2.1 – Діаграма часткового співвідношення виникнення помилок

Насправді можна говорити про декілька причин реальності справдження представленої на рисунку 2.1 діаграми. В деяких випадках самої специфікації просто не має, деколи влне не є повною, або ж надто зачасто видозмінюється.В процесі розробки програмного забезпечення досить часто програмістами забувається про те, що специфікація являється одною з основних частин всього цикла розробки програмного продукту. Таким чином, у випадку якщо вказаний етап пропущено, або його проведено з допущенням певних некоректностей, то помилки обов’язково з’являтимуться.

Ще одним суттєвим джерелом виникнення помилок можна назвати етап проєктування. Некоректно спроєктована архітектура програмного забезпечення через момент поспіху, відсутності досвіду, якоїсь може і необдуманості призведе до достатньо великих наслідків.

Якщо говорити про код, то тут помилки можуть виникати в наслідок великої складності програмного забезпечення, недостатність документації, високої щільності в графіку по виконанню завдання чи виникати просто рандомно. Важливо, на мою думку, також наголосити про те, що для більшості

випадків так звані поверхневі помилки, можуть вказувати на проблеми пов’язані із специфікаціями та проєктуванням.

І ще одним типом помилок можна назвати випадки, коли виявленні помилки ними фактично не являються, тобто виявлено не існуючі помилки. Також може помилки можуть копіюватись і утворювати так звані повтори помилок, які виникають при копіюванні з одного і того ж джерела. За допомогою деяких помилок можна робити висновки про неправильно проведене тестування. В загальному підсумку цей вид помилок складають достатньо низький відсоток, і суттєво не впливають на якість кінцевого продукту.

2.2 Оцінка надійності програмного продукту

Надійність програмного продукту, її оцінка – це питання достатньо складне, а дослідження в цьому напрямку хоча і проводяться, проте перебувають на початковому етапі. Обґрунтованим є виділення двох сторін такого не простого питання як надійність програмного забезпечення визначеного об’єкту, зокрема сюди відноситься програмна надійність об’єкта, яка є властивістю об’єкта реалізовувати заплановані функції, які обумовлюються якістю створюваного проґрамного продукту. Надійність програмного забезпечення є властивістю його реалізовувати заплановані та описані програмні вимоги. Якщо говорити про програмну надійність продукту, то її визначають спільна роботі як апаратури так і безпосередньо програми. Вона характеризується здатністю виробу реалізовувати функції, які задаються, з дотриманням умови, що програма перебуватиме у тому або ж видозміненому іншому стані.

Беззаперечним є і факт того, що якщо ми говоримо про якісний програмний продукт, то паралельно безсумнівною має бти і його надійність, яка і є характеристикою якісного стану програми в цілому.

Варта пам’ятати, що програмною надійністю об’єкту найбільше цікавиться кінцевий споживач чи замовник. Для того, щоби її забезпечити належним чином потрібно, щоби програма відповідала критеріям «правильна», «коректна» та «надійна», тобто щоби в ній не містилось помилок, або їх наявність не впливала на якісну роботу кінцевого продукту.

Програмна надійність є особливо актуальним питанням у випадку, якщо програма являється самостійною як продукт. В такому випадку її створюють, перевіряють та піддають випробуванню (тестують) фактично так само як і звичайний об’єкт.

2.3 Характеристика програмних і апаратурних відмов

Якщо говорити про як програмну так і апаратну відмови то можна говорити про те що в цих звавалося б різних питаннях є багато подібного та спільного, проте є й істотні відмінності. До спільного можна віднести:

– задані функції об’єктом не виконуються;

– випадковість часу, який фіксується до відмови та часу, який затрачається на усунення відмови;

– обробка статистичних даних стосовно відмов виконується однаковими методами їх обробки;

До істотних відмінностей мід програмними та апаратними типами відмов можна віднести:

– залежність від часу: для апаратної відмови характерна залежність від чогось одного від часу, чи від обсягів виконаних робіт, для програмної ж характерною є залежність від функцій, які виконуються під керуванням програми (точніше сказати, залежить від ймовірності виходу програми на ділянку, на якій помилка виникне);

– те що апаратну відмову виявлено та усунуто зовсім не значить того, що така ж чи подібна відмова не буде повторюватись при продовженні роботи виробу чи пристрою, проте у випадку з програмними відмовами, їх виявлення та усунення (корегування, правка програми) означає, що подібна помилка в подальшій роботі повторюватись більше не буде;

– прогноз виникнення відмови, у випадку їх апаратного типу здійснюється достатньо просто, якщо говорити про програмний тип відмов – тут ситуація значно складніша в плані того, що тут прогноз в окремих випадках здійснити достатньо складно, а частіше й взагалі неможливо. Передбачення часу коли програмні відмови стають діючими, а коли ні також здійснити достатньо важко;

– у випадку з апаратними відмовами, то оптимально їх поділяти на два види: раптові та поступові. Тобто такі відмови є різними за своєю фізичною суттю, законом розподілу часу до настання такої відмови, методами оптимізації за рахунок зниження імовірності їх настання.

Відсутній будь-який сенс застосовувати поділ на на раптові та поступові у випадку програмних відмов, оскільки їх виникнення є рандомниь вдразу, в той самий момент, коли програмою здійснюється перехід на ділянку, на якій допущено помилку. Проте, в той самий час вони за своєю суттю не збігаються з неочікуваними (раптовими) відмовами апаратного типу. Ймовірність їх настання ніяким чином не пов’язується із часом роботи пристрою, а замовлюється умовною імовірністю того, що в програмі міститься помилка в цій частині програми, а також імовірністю того, що пристрій працюватиме під управлінням цієї частини програми.

Існує також ще і питання дослідження надійності програмного забезпечення в основні задачі якого можна віднести:

– розробку методів для оцінювання та прогнозу у сенсі надійності програмного забезпечення в цілому, яку здійснюють ґрунтуючись на кількісних показниках та характеристиках (їх сукупності), які є ідентичними до показників по апаратній надійності;

– фактори та їх визначення, які можуть повпливати на досягнення певного заданого рівня надійності програмного забезпечення;

– методи та їх розробка за допомогою яких забезпечується досягнення певного заданого рівня надійності програмного забезпечення;

– методи підвищення надійності програмного забезпечення та їх вдосконалення при проєктуванні та й експлуатації програмного забезпечення;

Ефективним способом для підвищення надійності програмного забезпечення можна назвати застосування методу (-ів) структурного проєктування програм, оскільки в залежності від структури програмного забезпечення окремі помилки та їхні наслідки можна або легко виявити, локалізувати та виправити на деяких невеликих ділянках програми або ж вони будуть поширюватись і на інших рівнях та модулях програмного забезпечення.

Саме тестування програм та програмних продуктів на надійність їх програмного забезпечення є одним із обов'язкових етапів при перевірці на надійність системи. Перевірка на надійність програм реалізується при використанні спеціальних специфічних програм (так званих тестувальних) та з застосуванням спеціалізованих імітаційних стендів. Тоді здійснюється перевірка степіні відпрацьованості програми та її відповідності до заданих вимог.

Випробовування на перевірку на надійність пристроїв робота який, їх функціонування в цілому здійснюється безпосередньо при програмному управлінні, здійснюються спільною роботою як програми так і приствою. В цьому випадку здійснюється перевірка степені відпрацьованості програми відповідно заданим вимогам, коректності до заданих вимог, погодженості взаємодії як самої програми так і апаратури.

Складність методу перевірки залежить від точності, якої необхідно досягти в її процесі. Перевірка ж степені відпрацьованості програми може здійснюватися з використанням різних методів.

2.4 Прийоми та техніка тестування програмного забезпечення

Станом на сьогодні відомо достатньо багато як прийомів так і технік по тестуванню. Одні з них, природно, можна вважати кращими за інші, ще інші, при необхідності досягнення покращених результатів рекомендовані до використання сукупно. В цьому підрозділі, я виріших навести перелік найбільш поширених прийомів для тестування:

– прийом ручного тестування, це коли тести виконують люди , а тестові дані, для кожного з випробувань, є складеними попередньо, чи визначеними також попередньо;

– прийом автоматизованого тестування, це коли тести виконують при застосуванні спеціальних інструментів або як самостійні процеси Їх можна повторювати достатньо багато разів. При застосуванні цього виду тестування, тестувальні дані вводяться чи генеруються наперед;

– прийом регресивного тестування, це коли тести, як правило є автоматизованими, а метою процесу є виявлення негативного впливу на функції зміни в програмі, якими пройдено попередню перевірку;

– прийом димового тестування, це коли метою тестів є швидка перевірка базової функціональності. На меті виявити чи варта тестувати новий білд (нову програмну версію або версію певного її модулю) [10];

– прийом дослідницького тестування, це коли тести виконують при відсутній специфікації. Тестером розробляється власна система тестування, яка ґрунтується на набутому досвіді самого тестеру. Такий вид тестування може проводитися лише тестером з достатньо великим досвідом роботи, оскільки тут передбачено наявність достатньо глибоких знань про тестування;

– прийом так званого «мавпячого» тестування, це коли немає певної системи тестування, тестувальник проводитьє “швидку атаку” програми;

– прийом так званого стрес-тестування, це коли тести призначено для перевірки програми на стійкість при надмірному навантаженні при недостатньості ресурсів;

– прийом здійснення тестування через навантаження. Даним таким видом тестування передбачено його здійснення на різних рівнях навантаження. Тут за мету є перевірка поведінки програми, а також виявлення максимально можливого рівня навантаження, перевірка чи такий рівень навантаження відповідатиме вимогам поставленим до системи;

– прийом тестування на продуктивність, це коли тести виконують з метою порівняння поточної та розрахункової продуктивностей;

– прийом тестування інсталяції, це коли тестування відбувається при встановленні на передбачених специфікацією платформах, їх комбінації, а також перевірки на те чи всі файли переписані при цьому не зазнаючи жодних змін, чи робота програми в цілому є коректною.

– прийом тестування через довге користуванням. У цьому випадку тести проводяться протягом тривалого часу, а на меті ставиться виявлення помилок такого роду, коли виявлення є неможливим при здійсненні короткого використання [11].

2.5 Тестування на перевірку ергономічності

Юзабіліті (Usability) дослівно означає ступінь зручності даного предмета при його використанні. Застосовуючи до комп'ютерних технікам цим терміном можна назвати концепцію такої розробки інтерфейсів для користувачів, які насамперед орієнтовані на саме максимальне естетичне і психологічне зручність для користувача. Це концепції побудови таких веб-інтерфейсів, які будуть максимально сприяти досягненням цілей які поставлені перед сайтом, це може бути такі цілі як замовлення, дзвінок менеджеру, купівля товарів та інші.

Поліпшення юзабіліті сайтів значно збільшує конвертацію відвідувачів у замовлення, контакти і покупки. Воно має прямий вплив на ставлення користувача сайту до бренду і образу торгової марки представленому на ньому,

тому що зручність використання даного сайту, формує лояльність відвідувача до бренду і посилення його позицій.

2.6 Тестування розроблюваної системи. Тестування системи на кросбраузерність

Так як я планую, що систему я буду запускати в одному з встановлених браузерів занально доступних та поширених, а вони самі, тобто браузера, на комп'ютерах можуть бути різними, вважаю, що однією із важливих характеристик для розроблюваної системи буде саме кросбраузерність.

Кросбраузерність або по іншому браузерна незалежність – є властивістю сайту до ідентичної роботи та відображення в усіх браузерах (як правило, враховуються найпоширеніші). Під ідентичністю мається на увазі відсутність зміщень для основних блоків сайта, а також здатність до відображення матеріалу з одинаковою степінню читабельності.

У випадку, якщо не приділяти увагу до цього моменту (кросбраузерності), є високою ймовірність багатьох відвідувачів просто втратити. Причиною цього може бути те, що першою думкою при оцінці сайту користувачем, зазвичай залежить від його зовнішнього вигляду, зрозумілості, фону, впорядкованості та логічності розташування елементів. Некоректність відображення деяких елементів дизайна, розбіжності стиків, та подібне може не сподобатись користувачеві та відразу викликати негативну його реакцію та зіпсувати перше враження, а в зв'язку з тим що вже виникне в певній мірі упередженість в ставленні та оцінці, це може бути причиною того, що користувач покине сайт навіть не оцінивши всіх його переваг та можливостей. Пам'таючи попередній досвід, якщо цей сайт йому зустрінеться в процесі подального пошуку, він (користувач) його швидше за все проігнорує.

При тестуванні моєї роботи, процес включав в себе запуск сторінок автоматизованої системи на різних браузерах. Також було оцінено візуально на

предмет правильності та коректності відображення представлених на ній даних. Перша сторінка порталу в браузері представлена на рисунку 2.2 нижче.

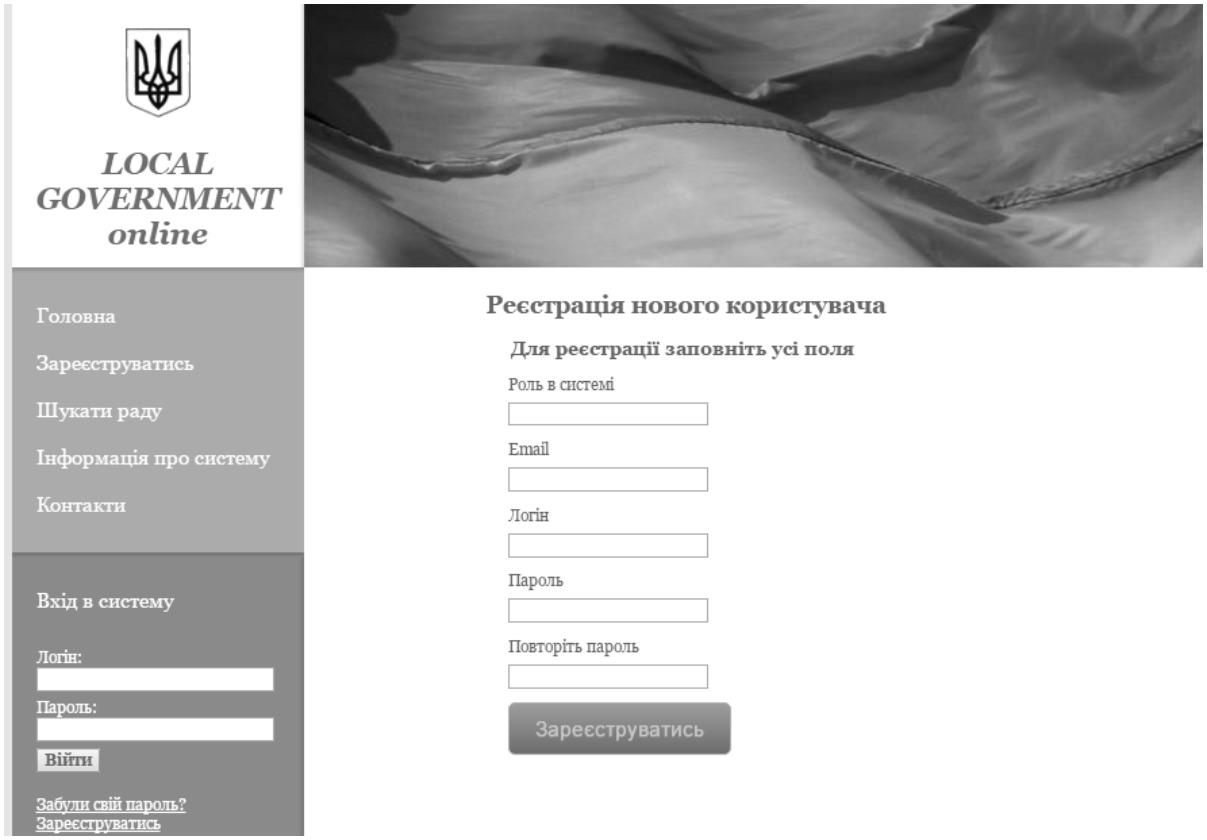

Рисунок 2.2 – Вигляд сторінки порталу в браузері Internet Explorer

ВИСНОВКИ

Розробка єдиної системи доступу до публічної інформації з використанням ASP .Net MVC Framework та технології C виконана в рамках кваліфікаційної роботи на здобуття освітнього ступеня «магістр» за спеціальністю 121 – Інженерія програмного забезпечення.

Використання єдиного системи інтернет-порталу призначеної для всіх ланок органів місцевого самоврядування має цілий ряд безсумнівних практичних переваг, які роблять її перспективною і затребуваною і, водночас, доводять правильність обраного напрямку дослідження, та предмет дослідження, зокрема:

1 - державою визначено єдиний загальнонаціональний стандарт обов'язкового функціоналу для web-сторінок органів влади (сьогодні не логічним і не зрозумілим є підхід, коли органи місцевого самоврядування на своїх офіційних сайтах надають користувачеві лише той функціонал, який для влади, а не для користувача на місцях, є збільш зручним);

2 – суттєвою є економія бюджетних кошів, зокрема на послугах спеціалістів IT галузі, коли зникне необхідність писати окремий сайт для кожного органу самоврядування;

3 - зручність безпосередньо для громадян оскільки інформація і офіційні документи усіх обласних, усіх районних, селищних та сільських рад буде уніфікованою та її можна знайти подібним шляхом, в одному місці, на одному порталі;

4 – дотримання умов безпеки, вкрай актуально сьогодні в силу пандемічних обмежень. Віддалений доступ до бази-системи уможливить обмін документами між владою та людьми через глобальну мережу інтернет, і громадянам не потрібно буде ризикувати здоров'ям, витрачати час на дорогу до органів влади, тощо В результаті зростає ефективність роботи установ та оперативність отримання даних;

5 – уніфікація та стандартизація, як можливість запровадити електронний документообіг на усіх рівнях органів місцевого самоврядування;

6 - зростання рівня відкритості, лояльності та публічності влади усіх рівнів і відповідно зростання рівня дотримання закону, оскільки кожен посадовець знатиме про те, що його рішення будуть викладені на розсуд громадськості.

Програмний продукт повинен якісно та коректно виконувати свої функції в різних ОС (операційних системах, зокрема Linux-i, Windows-i, MacOS-i), на яких можливе встановлення JRE версії 1.6та вищі. При розробці програмного забезпечення було використано ASP.NET MVC Framework та технологічні можливості мови програмування C#, за середовище було обрано середовище програмування VisualStudio.

Практичне застосування зумовлене актуальністю теми та полягає в тому, що розроблено систему запропоновано розглядати як уніфіковану, яка працює в якості єдиної систему доступу до документообігу, який в сучасних умовах повсемісних пандемічних обмежень, рекомендовано здійснювати в дистанційно, віддалено та в силу інтуїтивної зрозумілості, зручності та з метою економії часу та ресурсів, можна пропонувати до впровадження у всіх органах самоврядування, зокрема, для початку, апробувати на установах місцевого значення.

Роботу апробовано в рамках студентської конференції (IX науково-технічної конференції «Інформаційні моделі, системи та технології») м.Тернопіль, 2021 р.