ЗМІСТ

СПИСОК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

СЕД	– Системи електронного документообігу
IDE	– Integrated Development Environment
JSON	– JavaScript Object Notation
EDMS	– Electronic Document Management System
ERMS	– Electronic Records Management Systems

ВСТУП

Ще декілька років тому про системи електронного документообігу говорили як про "світле майбутнє". Але життя виявилося стрімкішим за наші уявлення. Системи електронного документообігу вже активно застосовуються на великих і середніх підприємствах, у державних структурах, і, що найголовніше, інтерес до них безперервно зростає.

Якщо на підприємстві вводиться певний формалізм в управлінні і в організації всіляких ділових процесів ("бізнес-процесів"), то рано чи пізно виникає необхідність хоча б частину управлінських механізмів переводити на впорядковану документальну основу. В результаті виникає документообіг.

Електронний документообіг – це життєвий цикл електронних документів в організації, починаючи від їх отримання, проходження у підрозділах і закінчуючи списанням в архів [1]. Часто електронний документообіг позначається терміном workflow, який характеризує рух документів як потік робіт, виконуваних в рамках того або іншого бізнес-процесу. Система електронного документообігу – це програмне забезпечення, головними завданнями якого є організація та підтримка життєвого циклу електронних документів. На сьогоднішній день існує безліч як платних, так і безкоштовних opensource рішень, якими може скористатися організація. Однак, всі з них володіють тими чи іншими вадами.

Система, розроблена в даній магістерській роботі, застосована на одному з підприємств нашого міста. На підприємстві виникла необхідність перевести обіг звітів філій підприємства з звичайних поштових розсилок в систему документообігу. Це необхідно було зробити для більшої зручності з введенням, зберіганням, систематизацією та переглядом звітів.

Звітом є строга форма в документі формату Excel. Метою створення системи є спрощення роботи топ-менеджменту зі звітами. Як відомо, час управлінців вищої ланки - дуже недешевий ресурс, а впровадження системи, дозволило б його заощадити. Була потрібна система, яка могла б інтегруватися у вже існуючі обчислювальні й інформаційні потужності підприємства, працювати на застосовуваних у компанії технологіях, надавати зручний і зрозумілий графічний інтерфейс, який дозволив би користувачу без підготовки відразу почати працювати з нею.

Метою даної комплексної магістерської роботи є розробка та реалізація WEB-орієнтованої системи електронного документообігу підприємства з територіально-розподіленою структурою.

Метою першої частини комплексної магістерської роботи є розробка клієнтської частини системи електронного документообігу підприємства.

Для досягнення поставленої мети в роботі необхідно вирішити наступні завдання:

- дослідження існуючих аналогів системами електронного документообігу;
- аналіз технічного завдання;
- функціональний опис системи електронного документообігу;
- логічне проектування системи електронного документообігу;
- вибір та обґрунтування технічних вимог та інструментальних засобів для фізичного проектування системи електронного документообігу;
- розробка інтерфейсу користувача та шаблону оформлення системи електронного документообігу;
- тестування системи.

1 АНАЛІТИЧНА ЧАСТИНА

Електронний документообіг – сукупність автоматизованих процесів по роботі з документами, представленими в електронному вигляді, з реалізацією концепції «безпаперового діловодства».

Існують такі види електронного документообігу:

- виробничий документообіг;
- управлінський документообіг;
- архівна справа (сукупність процедур архівної документообігу);
- кадровий документообіг (процедури кадрового обліку);
- бухгалтерський документообіг;
- складський документообіг;
- секретне і конфіденційне діловодство;
- технічний і технологічний документообіг.

Як ми можемо бачити, що систем документообігу може бути стільки ж, скільки існує видів діяльності, як наслідок, інформаційні системи, що автоматизують приватні види документообігу, розвиваються у напрямку масовості.

На системи електронного документообігу покладено наступні завдання:

- забезпечення ефективного управління за рахунок автоматичного контролю виконання, прозорості діяльності всієї організації на всіх рівнях;
- підтримка системи контролю якості, що відповідає міжнародним нормам;
- Підтримка ефективного накопичення, управління і доступу до інформації та знань. Забезпечення кадрової гнучкості за рахунок більшої формалізації діяльності кожного співробітника і можливості зберігання всієї передісторії його діяльності;
- протоколювання діяльності підприємства в цілому (внутрішні службові розслідування, аналіз діяльності підрозділів, виявлення "гарячих точок" в діяльності);
- оптимізація бізнес-процесів і автоматизація механізму їхнього виконання і контролю;
- виняток паперових документів з внутрішнього обороту підприємства. Економія ресурсів за рахунок скорочення витрат на управління потоками документів в організації;

- виняток необхідності чи істотне спрощення і здешевлення збереження паперових документів за рахунок наявності оперативного електронного архіву.

Не можна не відзначити, що такі системи мають безліч переваг [2]. По-перше це економія часу – службовці витрачають менше часу на пошук паперових документів. Завдяки центральній базі даних, регулярно створюються резервні копії файлів, завдяки чому виключається можливість того, що документ буде безповоротно втрачено, якщо його забудуть в літаку, випадково або навмисно знищать або ж просто згине в офісному безладі. Абсолютно виключається втрата часу на пошуки файлів і документів, яких, з якоїсь причини, не виявилося на своєму місці.

По-друге – підвищення прозорості внутрішньої роботи підприємства. Можливість для керівників спостерігати за статусом документа, протягом усіх етапів його погодження та затвердження. На додаток до цього, СЕД дозволяє моментально і легко викликати не тільки запитуваний файл, але також і повний звіт про те, хто його створив, хто мав до нього доступ і хто його редагував.

По-третє це підвищення безпеки інформації та документів (центральна база даних дозволяє робити резервні копії документів, завдяки чому знижується ризик випадкової або навмисної втрати файлів.)

По-четверте – більше гнучкості щодо фізичного місцезнаходження співробітників. Завдяки можливостям електронного доступу і комунікацій, службовці отримують можливість працювати віддалено. І навіть перебуваючи в одному і тому ж географічному місці, службовцям більше не буде потрібно чекати, поки паперові копії файлів будуть пересилатися з сусіднього офісу.

З урахуванням того, що на сьогоднішній день існує безліч систем електронного документообігу, слід врахувати наступні критерії при їх виборі:

1) Вимоги за обсягом зберігання. Необхідно вибрати систему документообігу, яка підтримує ієрархічне структурне зберігання (HSM - Hierarchal Storage Management). Цей механізм зберігає найбільш активно використовуються дані на найбільш швидких, але і найбільш дорогих носіях, в той час як рідше використовується інформація автоматично переноситься на повільні і дешеві носії.
2) Наявність формалізованих процедур, що вимагають підтримки їх виконання і автоматизації контролю (підготовки документів певного типу, виконання стандартних функцій організації і т.д.).

3) Необхідність автоматизації адміністративного управління організацією. Ступінь складності організаційної структури.
4) Наявність територіально розподілених підрозділів. Цей фактор накладає певні вимоги до віддаленого доступу, до реплікації даних і т.д.
5) Наявність паперового архіву великого обсягу. Деякі системи документообігу поставляються з вже інтегрованими підсистемами масового введення документів.
6) Наявність не задовольняє поточним потребам системи документообігу.
7) Необхідність в розвиненою маршрутизації документів, в управлінні потоками робіт (workflow managing). Як продовження цієї необхідності потреба в підтримці довільних бізнес-процесів, можливо працюють спільно з прикладними системами підтримки цих процесів.
8) Вимоги щодо термінів зберігання документів. При великих термінах зберігання (десятки років) варто серйозно подумати про організацію паралельного архіву на мікрофільмах.
9) Вимоги до "відкритості", розширення системи. Можливість інтеграції з існуючими інформаційними системами і використання наявного обладнання.
10) Необхідність зберігання зображень документів. Використання в організації специфічних форматів зберігання документів. Необхідність підтримки інженерних і конструкторських завдань, інших особливостей діяльності підприємства.
11) Необхідність розвинених засобів пошуку інформації. Повна підтримка системою мов наявних в організації документів.

З урахуванням існуючих систем документообігу та обліком всіх їх недоліків і переваг, було прийнято рішення розробити власну систему, яка відповідала б усім необхідним критеріям.

1.1 Аналіз технічного завдання

Темою магістерської роботи є розробка системи для обороту звітності для підприємств з географічно-розгалуженою структурою.

Звітом є строга форма в документі формату Excel. Дана система призначена для зберігання та систематизації звітності підприємств, що мають кілька офісів розташованих географічно далеко один від одного.

Метою створення системи є спрощення збору інформації по офісах для топ-менеджменту.

Система повинна мати механізм авторизації користувачів для обмеження доступу до її даних.

В системі повинні бути реалізовані ролі, за якими буде доступний різний її функціонал. У цілому, система повинна надавати користувачеві наступні можливості:

- перегляд звітів;
- створення звітів;
- редагування звітів;
- імпорт звітів у систему з excel-файлів;
- експорт звітів з системи у excel-файли;
- створення користувачів;
- редагування користувачів.

Доступ до системи користувачем повинен проводиться через браузер Google Chrome по мережі Інтернет. Відповідно, система повинна мати веб-інтерфейс, який надаватиме відповідний функціонал для відображення даних і проведення маніпуляцій з ними і системою. Всього користувачеві будуть доступні вісім сторінок. Це:

- “Сторінка авторизації ” ;
- “Сторінка перегляду та редагування звітів”;
- “Сторінка імпорту звітів”;
- “Сторінка створення звітів”;
- “Сторінка пошуку профілів користувачів”;
- “Сторінка редагування профілів користувачів”;
- “Сторінка створення користувачів (Add New User Page);
- “Сторінка скидування паролю”.

Повинні бути реалізовані кілька спливаючих вікон, які дозволять виконувати наступні дії:

- “Спливаюче вікно відновлення паролю” (“Forgot password”);
- “Спливаюче вікно для створення нового адміністративного юниту-офісу підприємства” – “branch” (“Add New Administrative Unit”);
- “Інформаційні спливаючі вікна” (“Information Popups”).

Всі спливаючі вікна повинні мати кнопку закриття вікна (X). Повинна бути реалізована обробка невірного URL і відображення відповідної інформації на сторінці. Для цього ми введемо додаткову сторінку - “Сторінка-обробник невірної адреси (Invalid URL Page) ”.

1.2 Опис сторінок, функцій і переходів

1.2.1 Сторінка авторизації(Login Page)

На сторінці мають бути розміщені:

- поле для вводу логіну;
- поле для вводу пароля;
- кнопка “Login” для введення даних в систему (також, авторизація має відбуватися по натисненню на кнопку “Enter”);
- посилання, “Forgot password”;
- чек-бокс “Remember me”.

Після введення коректних даних в систему повинен відбуватися перехід на сторінку “ Report Page”.

При натисканні на посилання "Forgot password" повинно з'являтися спливаюче вікно, за допомогою якого користувач зможе відновити свій пароль.

Чек-бокс "Remember me" дозволить користувачеві зберігати відкриту сесію, тобто навіть після закриття сторінки з проектом, дані про авторизацію повинні зберігатися і при наступному відкритті сторінки проекту, користувач має бути авторизований.

1.2.2 Сторінка перегляду і редагування звіту (Report Page)

Основні завдання, які має виконувати сторінка це:

- пошук і відображення звітів за заданий період часу;
- редагування відображуваних звітів.

На сторінці повинні бути розміщені:

- поле для введення початкової дати пошуку;
- поле для введення кінцевої дати пошуку;
- кнопка пошуку звітів;
- таблиці для відображення знайдених звітів;

- кнопка скасування редагування "Cancel" (скасовує поточне редагування);
- кнопка “Submit”;
- список, що випадає з іменами branches.

Для редагування має бути реалізовано 2 сценарії:

1) користувач робить подвійне клацання на комірці таблиці, і ряд, якому належить комірка, стає редагованим;
2) користувач натискає кнопку "Edit", і ряд обраний ним до цього стає редагованим. У разі, якщо ряд не був обраний – редагованим стає перший ряд в таблиці.

Поля звіту «дата», «branch», і «власник репорту» не повинні бути редагованими. Поля для редагування повинні бути багаторядковими.

При натисканні на кнопку "Submit", дані повинні відправлятися на сервер і зберігатися в базу. Після завершення операції повинно з'являтися спливаюче вікно з відповідною інформацією про успішне або неуспішному завершенні операції (у разі помилок або проблем – відповідна інформація має бути виведена у спливаючому вікні).

Список, що випадає, з іменами “branches” необхідний, так як програма передбачає наявність кількох офісів і різний рівень доступу до звітності. Для користувачів з доступом до всіх звітів, у випадаючому меню повинні бути доступні всі branches, а так же пункт "All". При виборі одного з branch, користувач отримує можливість пошуку звітів тільки з цього branch. При виборі вкладки "All" пошук повинен проходити по всіх доступних звітах. Для користувачів з обмеженим доступом до звітів, меню, що випадає, має бути неактивним, а в ньому був обраний branch, до якого приписаний (в якому працює) користувач.

Сторінка повинна надавати переходи на наступні сторінки:

- “Import Page”;
- “Create Report Page”;
- “Search User Page”.

1.2.3 Сторінка імпорту звітів (Import Page)

Дана сторінка призначена для завантаження звітів з документів формату Excel в базу. Звіт має строгу форму (рис. 1.1)

Reporter:	Date			
John Smith	01/03/2013			
John Smith	06/03/2013			
Name Surname	06/03/2013			
SUMMARY				
New summary number 69				
Some summary by Alex				
My summary				
	ACTION	**DATE**	**OWNER**	**Status***
SUCCESSES:				
Description of report detail number 321	action	01/03/2013	John Smith	Open
Some success	action	05/03/2013	John Smith	Open
my success edited	my action	06/03/2013	Name Surname	Open
OPPORTUNITIES:				
Description of report detail number 322	action	01/03/2013	Smith John	Open
FAILURES:				
Description of report detail number 323	action	01/03/2013	Billy Bob	Closed
THREATS / RISKS:				
Description of report detail number 324	action	01/03/2013	Billy Bob	Open
ESCALATIONS:				
Description of report detail number 325	action	01/03/2013	Smith John	Open

Рисунок 1.1 – Приклад звіту в Excel документі

Кількість записів в секціях “Reporter”, “Summary”, “Successes”, “Opportunities”, “Failures”, “Threats\Risks”, “Escalations” може змінюватись, тому необхідно передбачити коректну обробку цих ситуацій.

Сторінка повинна давати користувачеві наступні можливості:

- завантаження декількох звітів одночасно;
- верифікація даних у звітах під час завантаження (програма має перевіряти заповнення полів і показувати відповідну помилку, у разі, якщо якихось даних не вистачає).

Сторінка повинна з'являтися в окремому вікні. На ній повинно бути розміщено 2 елементи:

- посилання “New”;
- кнопка “Cancel”.

При натисканні на кнопку "Cancel" сторінка повинна закриватися.

При натисканні на кнопку "New" повинно з'являтися стандартне вікно ОС Windows для завантаження файлів. У ньому користувач може вибрати кілька звітів, після чого завантаження повинно бути розпочато в автоматичному режимі.

Після завантаження файлу, його ім'я і розмір повинні бути відображені у вікні. Посилання "New" повинно бути замінено на "Add another one”.

1.2.4 Сторінка створення звіту (Create Report Page)

Сторінка повинна містити елементи:

- таблиці для заповнення звітів;
- кнопку “Cancel” ;
- кнопку “Submit”.

При натисканні на кнопку "Submit" програма має працювати за алгоритмом, представленому на рис. 1.2.

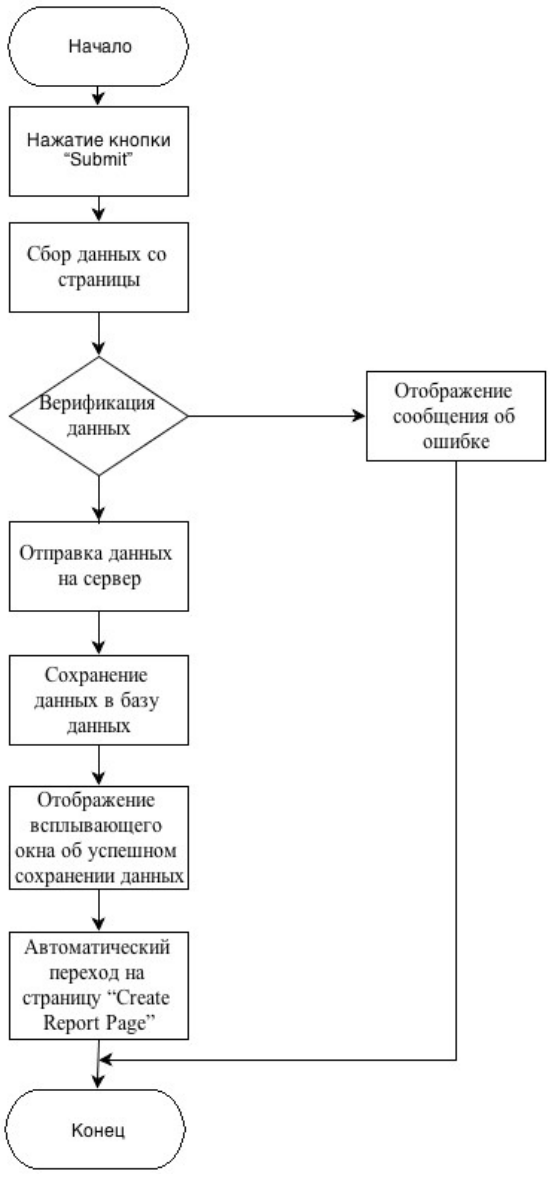

Рисунок 1.2 – Алгоритм роботи програми при натисканні кнопки “Submit”

По натисканню на кнопку "Cancel" має з'явитися спливаюче вікно, в якому користувач міг би підтвердити скасування створення звіту або ж продовжити його створення у разі, якщо кнопка була натиснута випадково. При натисканні кнопки "Ok" у спливаючому вікні, воно має бути закрите, а користувач перенаправлений на сторінку "Report Page". У разі натискання у спливаючому вікні кнопки "Cancel", воно повинно закриватися. Вся раніше введена інформація повинна бути збережена. При переході на сторінку "Create Report Page" і до введення інформації до звіту кнопка "Cancel" повинна бути неактивною.

Повинен бути передбачений функціонал, який дозволяв би додавати рядки в таблиці і видаляти непотрібні рядки у випадку, якщо якась із частин звіту ("Successes", "Opportunities", "Failures", "Threats \ Risks", "Escalations") не повинна містити жодної інформації.

Сторінка повинна надавати переходи на сторінки:

- “Report Page”;
- “Search User Page”.

1.2.5 Сторінка пошуку профілю користувача (Search User Page)

У сторінки повинен бути зручний інтерфейс для пошуку профілю користувачів в системі. На сторінці повинні бути розміщені:

- поле для введення пошукової інформації;
- таблиця для відображення результатів пошуку;
- кнопка “Edit” ;
- кнопка “Add User”.

Пошук повинен проводитися по клацанню клавіші "Enter", коли поле пошуку знаходиться у фокусі або ж по клацанню по іконці лупи. Поле має містити як мінімум одну латинську літеру, щоб користувач міг провести пошук. Система повинна шукати введені дані у всіх даних користувача, тобто в його логіні, імені, прізвища, електронній адресі. Після пошуку всі результати повинні бути виведені на сторінку.

Функціонал кнопки "Edit" наступний. Користувач повинен вибрати один з результатів, клацнувши по ньому. При цьому результат повинен підсвітити. Потім користувач повинен натиснути кнопку "Edit". При цьому повинен відбутися перехід на сторінку "Edit User Page", а всі поля сторінки повинні бути заповнені даними обраного користувача.

По натискання на кнопку "Add User" користувач повинен бути перенаправлено на сторінку "Add New User Page ”.

На сторінці повинні бути переходи на сторінки:

- “Report Page”;
- “Search User Page”.

1.2.6 Сторінка редагування профілю користувача (Edit User Page)

Сторінка повинна містити такі елементи:

- поле “User login”;
- випадаючий список “User role”;
- поле “User name”;
- поле “User surname”;
- поле “E-mail”;
- випадаючий список “Administrative unit”;
- чек-бокс “Enable user profile”;
- поле “Enter new user password”;
- поле “Confirm new user password”;
- посилання “Back to User profile search page” (перехід на “Search User Page”);
- кнопку “Add new”;
- кнопку “Add user”;
- кнопку “Save”.

На всіх полях має бути реалізована відповідна валідація. Поле "User login" повинно відповідати таким вимогам:

- містити в собі від 6 до 16 символів;
- містити тільки латинські символи.

Поле "User name" повинне відповідати таким вимогам:

- містити тільки латинські символи;
- перша літера повинна бути заголовної (якщо перша літера не була заголовною – необхідний її такою зробити).

Поле “User surname” повинне відповідати таким вимогам:

- містити тільки латинські символи;
- перша літера повинна бути заголовної (якщо перша літера не була заголовною - необхідний її такою зробити).

Інформація в полі "E-mail" повинна задовольняти наступним вимогам:

- мати вигляд “username@hostname”.

На частину "username" накладаються наступні обмеження:

- містити латинські символи;
- містити цифри символи;
- может містити лише латинські символи;
- может містити знаки ! # $ % & ' * + — / =? ^ _ ` { | } ~, а так само точку, за винятком першого і останнього знака, яка не може повторюватись.

На частину "hostname" накладаються наступні обмеження:

- складається з кількох компонентів, розділених крапкою:
  1) складаються з латинських букв, цифр і дефісів;
  2) дефіси не можуть бути на початку або в кінці компонента.
- не перевищує 63 символи;
- містить суфікси-домени першого рівня.

Поля "Enter new user password" і "Confirm new user password" повинні відповідати таким вимогам:

- містити від шести до шістнадцяти символів;
- містити тільки латинські символи, цифри, спец-символи;
- містити як мінімум одну малу літеру;
- містити як мінімум одну велику літеру;
- містити як мінімум один спец-символ;
- містити як мінімум одну цифру.

Символи, що вводяться не повинні бути видимі. Пароль і перевірочний пароль повинні бути однаковими.

“User role” – меню, що випадає, яке повинно містити всі доступні ролі. На даний момент – це "Administrator", "Executor", "Supervisor".

“Administrative unit” – це випадаючий список, котрий повинен містити в собі імена branches (офісів або міст в яких вони розташовані).

“Enable user profile” – це чекбокс, який відповідає за активацію користувача. Якщо прапорець знятий, то користувач стає неактивним, і ввійти їм в систему неможливо.

По натисканню кнопки "Add new" повинно з'являтися спливаюче вікно для додавання нового адміністративного юніта (офіс підприємства-branch).

Кнопка "Add user" повинна перенаправляти користувача на сторінку "Add New User Page".

Натискання кнопки "Save", у разі коректності введених даних, має призвести до збору даних зі сторінки і передачі їх на сервер для подальшого збереження в базу.

На сторінці повинні бути переходи на наступні сторінки:

- “Report Page”;
- “Search User Page”;
- “Add New User Page”.

1.2.7 Сторінка додавання користувача (Add New User Page)

Сторінка повинна містити такі елементи:

- поле “User login”;
- випадаючий список “User role”;
- поле “User name”;
- поле “User surname”;
- поле “E-mail”;
- випадаючий список “Administrative unit”;
- чекбокс “Enable user profile”;
- поле “Enter new user password”;
- поле “Confirm new user password”;
- посилання “Back to User profile search page” (перехід на “Search User Page”);
- кнопка “Add new”;
- кнопка “Save”.

На всіх полях має бути реалізована відповідна валідація. Поле "User login" повинно відповідати таким вимогам:

- містити в собі від 6 до 16 символів;
- містити тільки латинські символи.

Поле "User name" повинне відповідати таким вимогам:

- містити тільки латинські символи;
- перша літера повинна бути заголовної (якщо перша літера не була заголовною – необхідний її такою зробити).

Поле “User surname” повинне відповідати таким вимогам:

- містити тільки латинські символи;
- перша літера повинна бути заголовної (якщо перша літера не була заголовною - необхідний її такою зробити).

Інформація в полі "E-mail" повинна задовольняти наступним вимогам:

- мати вигляд “username@hostname”.

На частину "username" накладаються наступні обмеження:

- містити латинські символи;

- містити цифри символи;
- може містити лише латинські символи;
- може містити знаки ! # $ % & ' * + — / =? ^ _ ` { | } ~, а так само точку, за винятком першого і останнього знака, яка не може повторюватись.

На частину "hostname" накладаються наступні обмеження:

- складається з кількох компонентів, розділених крапкою:
  1) складаються з латинських букв, цифр і дефісів;
  2) дефіси не можуть бути на початку або в кінці компонента.
- не перевищує 63 символи;
- містить суфікси-домени першого рівня.

Поля "Enter new user password" і "Confirm new user password" повинні відповідати таким вимогам:

- містити від шести до шістнадцяти символів;
- містити тільки латинські символи, цифри, спец-символи;
- містити як мінімум одну малу літеру;
- містити як мінімум одну велику літеру;
- містити як мінімум один спец-символ;
- містити як мінімум одну цифру.

Символи, що вводяться не повинні бути видимі. Пароль і перевірочний пароль повинні бути однаковими.

“User role” – меню, що випадає, яке повинно містити всі доступні ролі. На даний момент-це "Administrator", "Executor", "Supervisor".

“Administrative unit”- випадаючий список, котрий повинен містити в собі імена branches (офісів або міст в яких вони розташовані).

“Enable user profile”- чекбокс, який відповідає за активацію користувача. Якщо прапорець знятий, то користувач стає неактивним, і ввійти їм в систему неможливо.

По натисканню кнопки "Add new" повинно з'являтися спливаюче вікно для додавання нового адміністративного юніта (офіс підприємства –branch).

Натискання кнопки "Save", у разі коректності введених даних, має призвести до збору даних зі сторінки і передачі їх на сервер для подальшого збереження в базу.

На сторінці повинні бути переходи на наступні сторінки:

- “Report Page”;
- “Search User Page”.

1.2.8 Сторінка скидання пароля (Reset Password Page)

Сторінка повинна містити такі елементи:

- поле “New password”;
- поле “Confirm password”;
- кнопку “Submit”;
- кнопку “Cancel”.

Пароль повинен задовольняти наступним вимогам:

- містити від шести до шістнадцяти символів;
- містити тільки латинські символи, цифри, спец символи;
- містити як мінімум одну малу літеру;
- містити як мінімум одну велику літеру;
- містити як мінімум один спец-символ;
- містити як мінімум одну цифру;

Відповідна валідація повинна бути на полях "New password", "Confirm password". Символи, які мі в них вводимо не повинні бути бачимими (повинні замінюватися колами). Пароль і перевірочний пароль повинні бути однаковими.

По натисканню кнопки "Submit", якщо паролі введені правильно, вони повинні бути збережені в базу, користувач повинен побачити повідомлення "Your password has been changed." і направлений на сторінку "Login Page”.

По натисканню на кнопку "Cancel" користувач повинен перенаправлятися на сторінку "Login Page”.

1.2.9 Сторінка-обробник невірної адреси (Invalid URL Page)

“ Invalid URL Page " – сторінка, яка повинна з'являтися у випадку, якщо користувач ввів некоректний URL.

Сторінка повинна містити:

- повідомлення “The URL " введений користувачем URL " is invalid!”;
- посилання “Back to Login Page”.

По натисненню на посилання “Back to Login Page”, користувач має опинятися на сторінці “Login Page”.

1.3 Спливаючі вікна, їх функціонал

1.3.1 Спливаюче вікно відновлення паролю (“Forgot password”)

Спливаюче вікно повинно містити в собі:

- текстову підказку;
- поле для вводу логіна;
- кнопку “Ok”;
- кнопку “Cancel”.

Якщо був введений логін, який є в системі, і натиснута кнопка "Ok" – користувачеві на пошту повинно бути вислано лист з посиланням на сторінку зміни пароля. Лист повинен мати наступний зміст, показане на малюнку нижче (рис. 1.3)

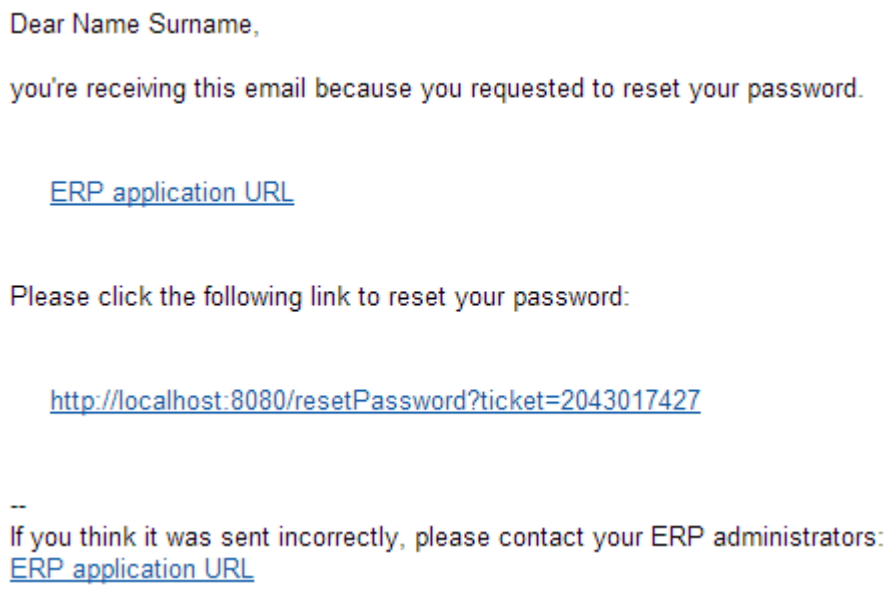

Рисунок 1.3 – Форма листа з посиланням на відновлення пароля

У разі якщо такого логіна в системі немає – має бути показано відповідне повідомлення. Кнопка "Cancel" повинна закривати вікно.

1.3.2 Додавання нового адміністративного юніт-офісу підприємства (“Add New Administrative Unit”)

Спливаюче вікно повинно містити в собі:

- поле “Enter full unit name”;
- поле “Enter short unit name”;
- кнопку “Ok”;
- кнопку “Cancel”.

До полів "Enter full unit name" і "Enter short unit name" повинна бути застосована валідація.

Поле “Enter full unit name” має вдовольняти наступним вимогам:

- поле не має бути порожнім;
- може містити тільки латинські символи і символ пропуску;
- пропуски на початку і кінці рядка повинні обрізатися;
- мінімальна довжина рядка – 2 символи;
- максимальна довжина рядка – 20 символів.

Поле "Enter short unit name" повинне відповідати таким вимогам:

- поле не має бути порожнім;
- може містити тільки латинські символи і символ пропуску;
- пропуски на початку і кінці рядка повинні обрізатися;
- мінімальна довжина рядка – 2 символи;
- максимальна довжина рядка – 7 символів;

У разі порушення валідації, поля повинні підсвічуватися червоним кольором, при наведенні на них мишкою повинно з'являтися відповідне повідомлення про помилку.

По натисканню на кнопку "Ok" дані з полів спливаючого вікна повинні збиратися, перевірятися і, у разі їх правильності, відправлятися на сервер для подальшого збереження в базу.

По натисканню на кнопку "Cancel", спливаюче вікно повинно закриватися.

1.3.3 Інформаційні спливаючі вікна (“Information Popups”)

Дані вікна призначені для відображення повідомлень про успішне або неуспішному проходженні тієї чи іншої операції.

Вікна повинні містити:

- іконку “success”\“failure\alert”;
- текст з інформацією;
- кнопку “Ok”.

По натисканню на кнопку, інформаційне вікно повинно закриватися. Необхідний передбачити гнучкий функціонал, щоб при необхідності кнопка "Ok" могла виконувати різні дії та їх комбінації:

- закриття спливаючого вікна;
- закриття спливаючого вікна і перехід на іншу сторінку;
- закриття спливаючого вікна і оновлення сторінки;

- інші опціональні дії.

2 ОБГРУНТУВАННЯ ВИБОРУ ПРОГРАМНИХ ЗАСОБІВ РОЗРОБКИ

2.1 Огляд існуючих систем

На сьогоднішній день існує безліч систем документообігу. Розглянемо деякі з них.

1) DOCS Open/Fusion

Система DOCS Open / Fusion розроблена компанією PC DOCS. Розглянемо головний екран даної системи (рис. 2.1). Вона призначена для використання на великих і середніх підприємствах для автоматизації управління технічною документацією та проектами: від введення інформації з паперових носіїв, її редагування і обробки до приміщення інформації в архів і отримання твердої копії і різних звітів по проекту. DOCS Open / Fusion дозволяє зберігати різні типи документів: креслення (векторні, растрові і гібридні); специфікації, моделі складу вироби; аудіо і відеофайли, текстові файли, електронні таблиці та ін. На базі DOCS Open / Fusion створюються архівні системи в масштабах відділу , підрозділи, підприємства. Підтримується робота з мобільними і віддаленими користувачами і групами користувачів.

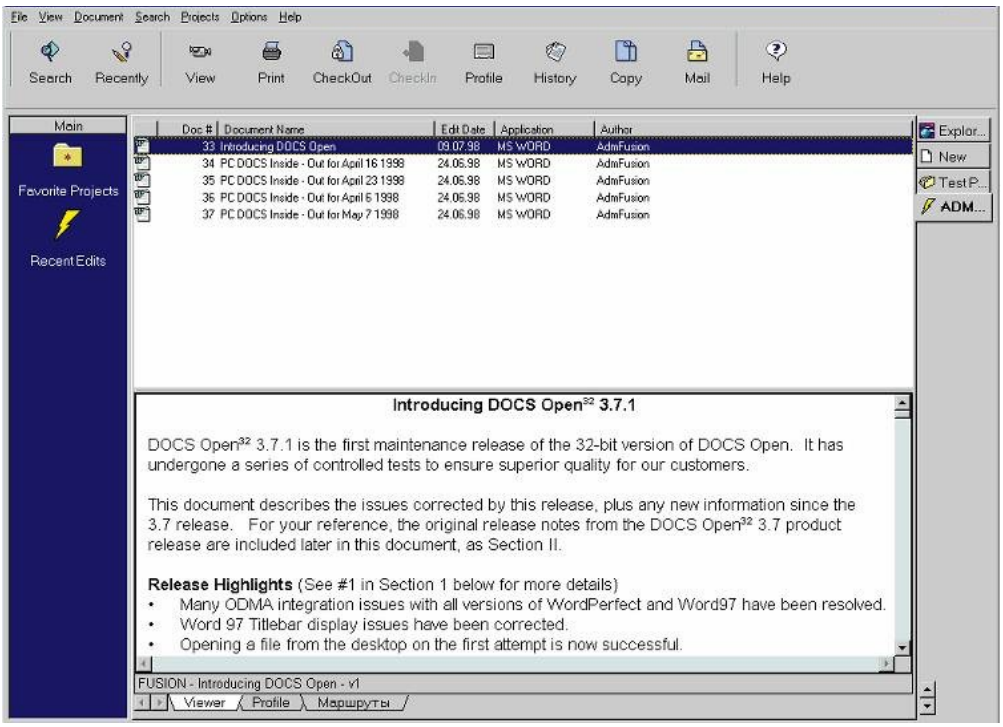

Рисунок 2.1 – Інтерфейс PowerDOCS -кліентского модуля DOCSFusion

2) CyberDocs

Клієнт CyberDocs забезпечує практично ту ж функціональність, що і PowerDocs, але через Internet-браузер. Розглянемо робочу область даної системи (рис. 2.2) В одному комплексі може бути встановлено кілька серверів DocsFusion, при цьому автоматично реалізується балансування навантаження і стійкість до збоїв (fault tolerance).

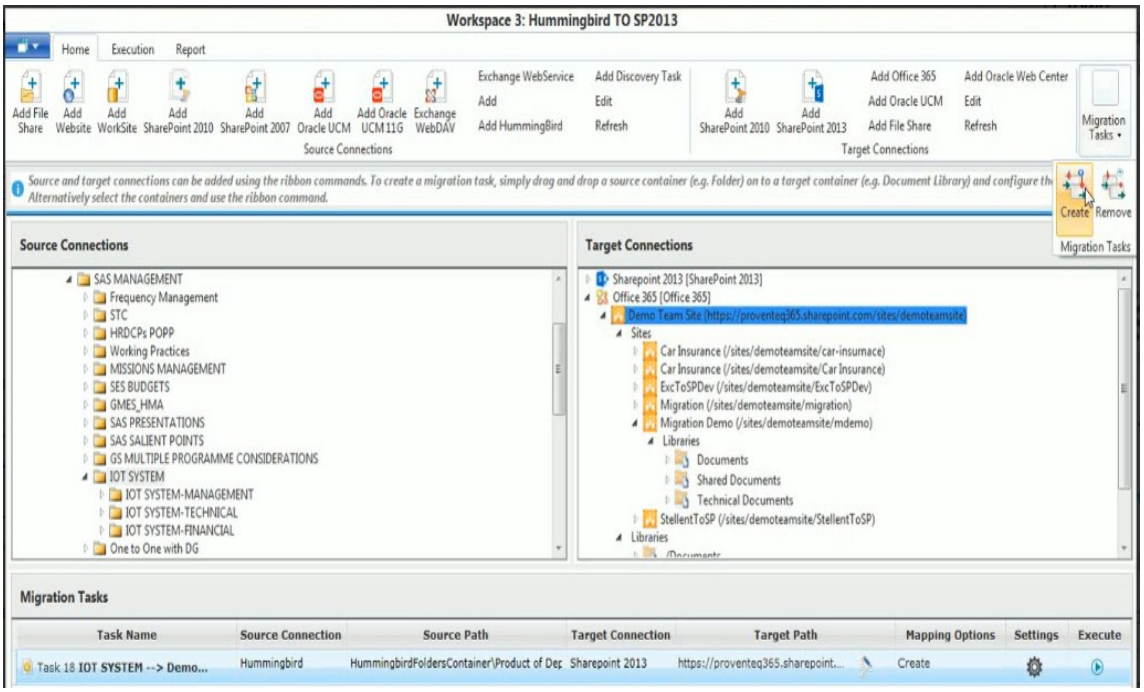

Рисунок 2.2 – Робоча область CyberDocs

Це означає, що при збої одного з серверів користувачі відчують лише деяке уповільнення роботи системи, а сама система дійсно може забезпечити одночасну роботу з нею досить великої кількості користувачів. Для зберігання даних системи необхідно використовувати Microsoft SQL Server або Oracle. Як сховище для самих документів використовується файлова система. Підтримується механізм ієрархічного зберігання даних HSM. Система дозволяє легко здійснити інтеграцію і стикування з іншими прикладними системами як на рівні клієнта PowerDocs, так і на рівні сервера. Docs – це відкрита платформа, до неї поставляються засоби розробки для створення спеціалізованих додатків або інтеграції з іншими системами.

3) Documentum EDMS

Відома в світі система Documentum EDMS розробки компанії Documentum, призначена для середніх і великих підприємств (особливо зі складним гетерогенним IT-середовищем). Головна сторінка Documentum

зображена на рис. 2.3. В основі Documentum EDMS положено об'єктно-орієнтований підхід. Система базується на концепції узагальненого документ-об'єкта (Docobject). Кожен такий об'єкт складається з 4-х частин: вмісту документа, його атрибутів, зв'язків з іншими об'єктами і операцій, що проводяться над документ-об'єктом, зокрема, маршрутів його проходження. Один документ-об'єкт може містити кілька подань (форматів) документа, система автоматично синхронізує вміст всіх уявлень при зміні одного з них.

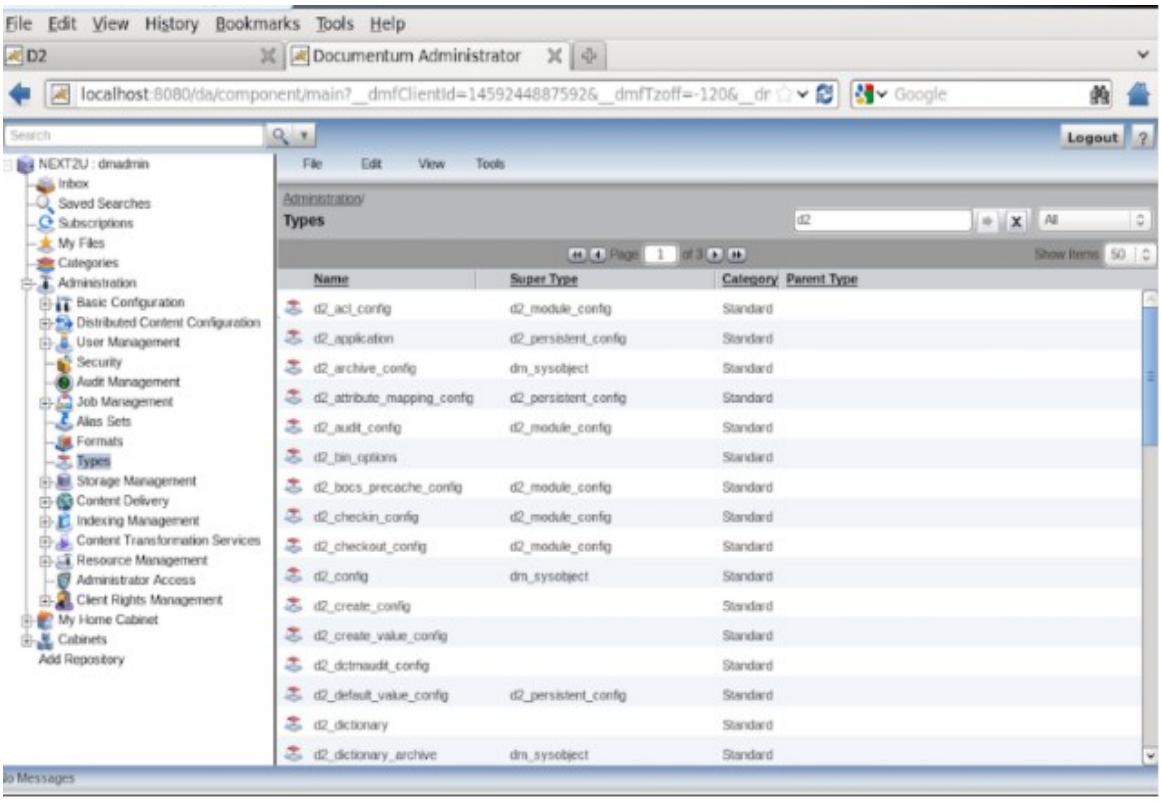

Рисунок 2.3 – Головна сторінка Documentum

4) Staffware

Система Staffware створена компанією Staffware plc і призначена для комплексного вирішення завдань управління бізнес-процедурами, діловими операціями і документообігом. Розглянемо інтерфейс розширеного пошуку (рис 2.4).

У Staffware кожна регулярно повторювана управлінська функція представляється у вигляді процедури, що складається з окремих кроків і містить алгоритм виконання кожного кроку, а також опису порядку кроків. Конкретна реалізація управлінського процесу являє собою варіант процедур, з яким і мають справу користувачі системи. Побудова процедур здійснюється

за допомогою графічного конструктора процедур. Кожному кроку процедури відповідає екранна форма, що містить деяку підмножину полів даних. Створення форм є прерогативою розробника процедур і здійснюється за допомогою графічного конструктора форм. У процесі виконання процедури Staffware накопичує завдання, відповідні окремим крокам процедури, і формує черги завдань різних типів як для кожного користувача, так і для груп користувачів. Для контролю та управління поточним станом виконання варіантів процедур у Staffware передбачені реєстраційні журнали та адміністративні звіти, що формуються автоматично.

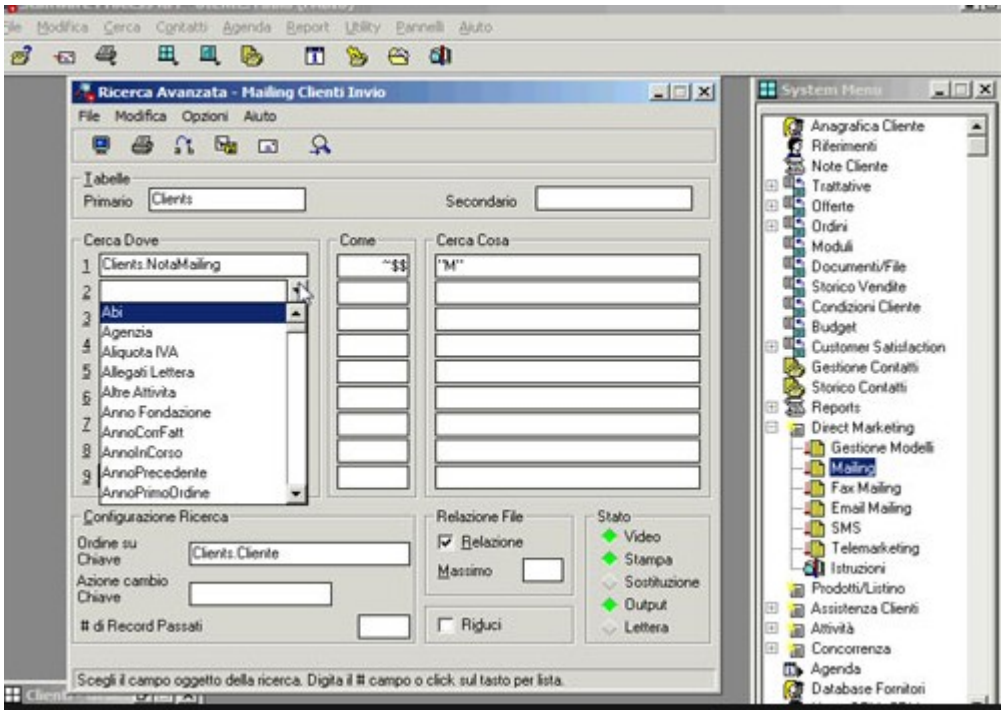

Рисунок 2.4 – Інтерфейс розширеного пошуку

5) ActionWorks Metro

Web-орієнтоване ПО організації документообігу ActionWorks Metro компанії Action Technologies. Розглянемо інтерфейс ActionWorks Metro(рис. 2.5). Це типова програма для Інтернет: сервер управління процесами розташовується позаду Web-сервера, який функціонує в якості внутрішнього шлюзу. Перевірити статус завдання користувачі можуть за допомогою Web-браузера. Для визначення документообігу вони використовують виділений клієнт або інструментальний засіб проектування, Process Builder, який звертається безпосередньо до сервера ActionWorks Metro.

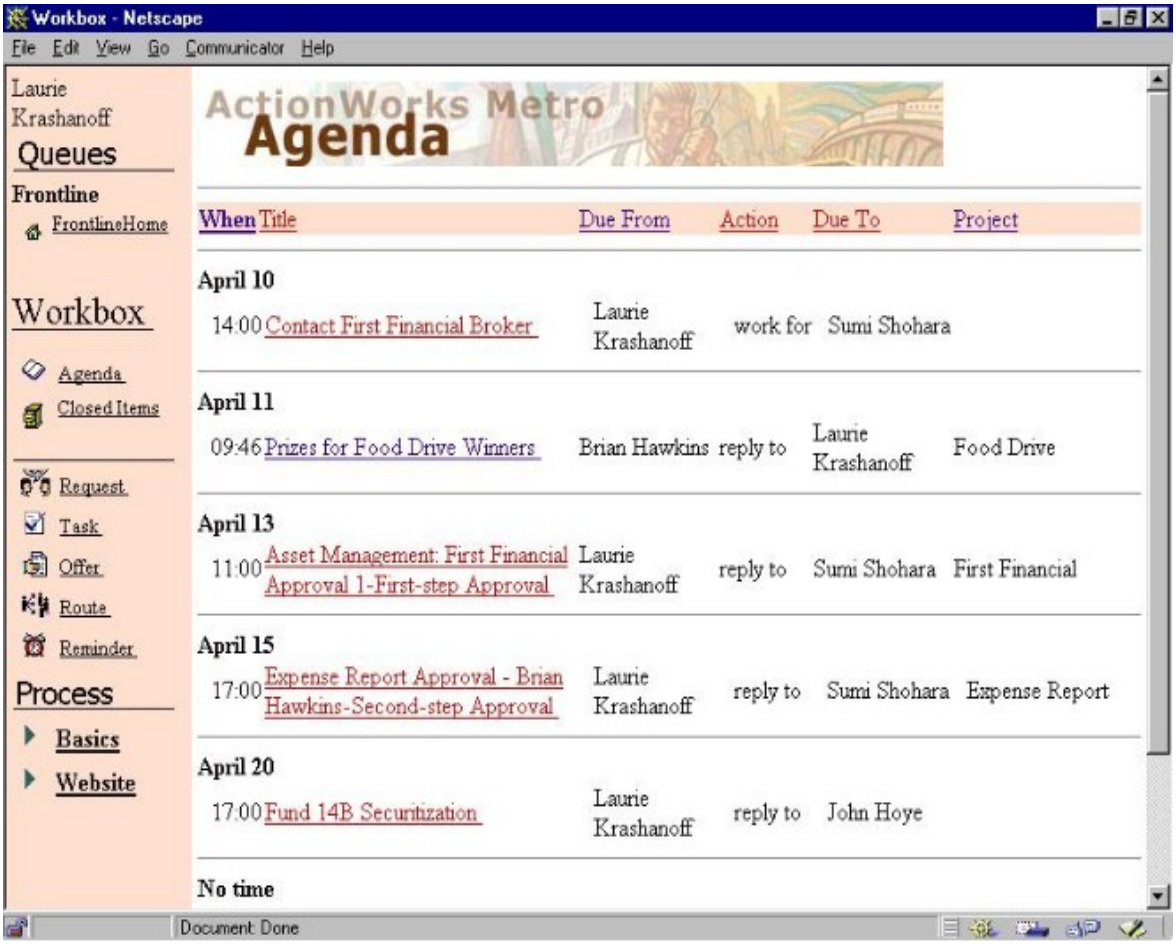

Рисунок 2.5 – Інтерфейс ActionWorks Metro

Process Builder дозволяє створювати візуальну діаграму послідовності проходження документа з використанням маркованої рамки для позначення кожного етапу і стрілок для визначення відносин між ними. За допомогою даного інструменту користувачі можуть задавати складні правила автоматизації комплексних процесів.

У свою чергу, інструментальний засіб AutoPilot робить можливим проектування із браузера (слід зазначити, що цей тонкий клієнт дозволяє описувати тільки базові процеси). Документообіг AutoPilot заснований на використанні форм. Потенційний розробник системи документообігу може вибрати з декількох створених наперед форм або створити свою власну. Користувачеві пропонується 1 або 2 варіанти вибору, а після визначення процесу AutoPilot автоматично завантажує його на сервер Metro. Після завантаження даного процесу користувачі приймають участь в документообігу за допомогою своїх Java-сумісних Web-браузерів.

6) LanDocs

Система в першу чергу орієнтована на діловодство та архівне зберігання документів. Вона складається з декількох компонентів: систем

діловодства, серверів документів (архівів), підсистем сканування та візуалізації образів, підсистем організації дистанційного доступу з використанням Інтернет-клієнта, поштового сервера. Розглянемо інтерфейс перегляду списку повідомлень (рис. 2.6).

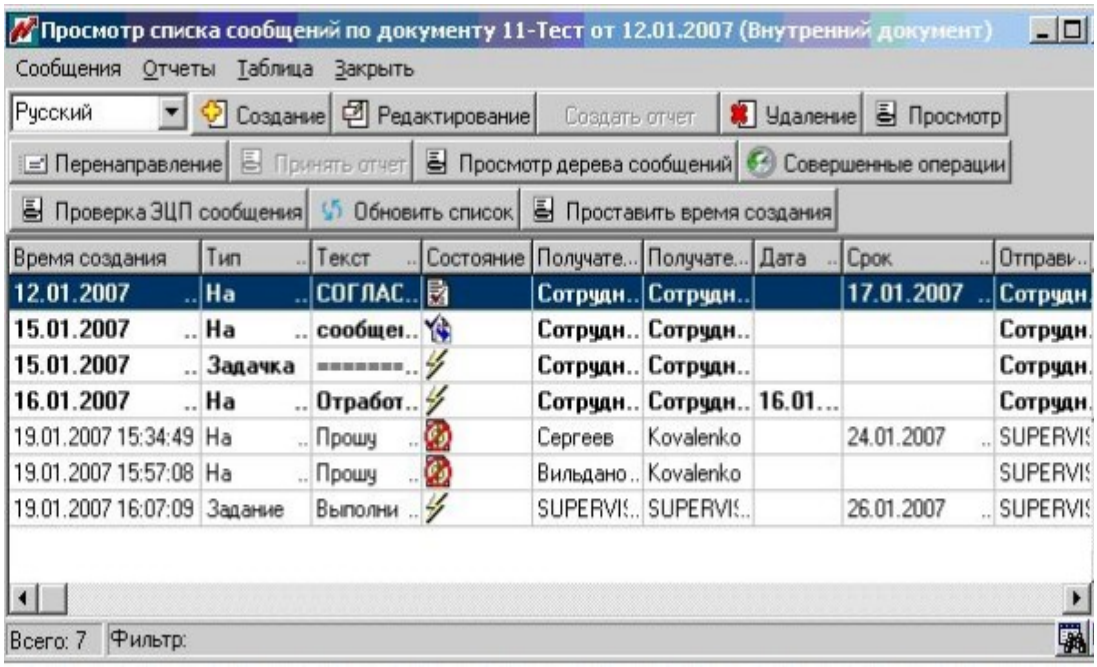

Рисунок 2.6 – Інтерфейс перегляду списку повідомлень.

Компонент діловодства реалізований в клієнт-серверній архітектурі на базі промислової СУБД: Oracle або Microsoft SQL Server. Програмне забезпечення для централізованого управління зберіганням документів в електронному архіві реалізується у вигляді окремого сервера. Як окремий варіант постачається модуль повного текстове пошуку документів з урахуванням правил російського мови. Почтовая служба LanDocs була зроблена таким чином, що співробітники, у яких встановлений спеціальний клієнтський компонент LanDocs, можуть отримувати повідомлення-задання і звітувати по ним, використовуючи стандартний поштовий ящик Microsoft Exchange або Lotus Notes. Продукт відкритий для розробників - має API для вбудування LanDocs в Windows-додатки сторонніх розробників. Компонент сканування та роботи з зображеннями має досить просунуту функціональність: він дозволяє фільтрувати зображення, виправляти перекос, виник після сканування, розпізнати текст.

Система LanDocs не орієнтована на підтримку колективної роботи та процесу створення документів.

2.2 Технічна пропозиція

У всіх перерахованих вище систем є кілька недоліків:

1) вимагають розгортки і кропіткого налаштування. В окремих випадках, необхідна доробка фахівцями;
2) пропонують зайвий функціонал, який в даному випадку не потрібний і надлишковий;
3) як правило, вимагають постійної кваліфікованої підтримки.
4) практично всі подібні системи платні і покупка такої системи може стати серйозним ударом по бюджету підприємства.

Зважаючи на ці недоліки і специфічні вимоги, було прийнято рішення розробити свою систему.

Наша програма має будуватися на основі структури MVC. Загальна схема представлена на рис. 2.7.

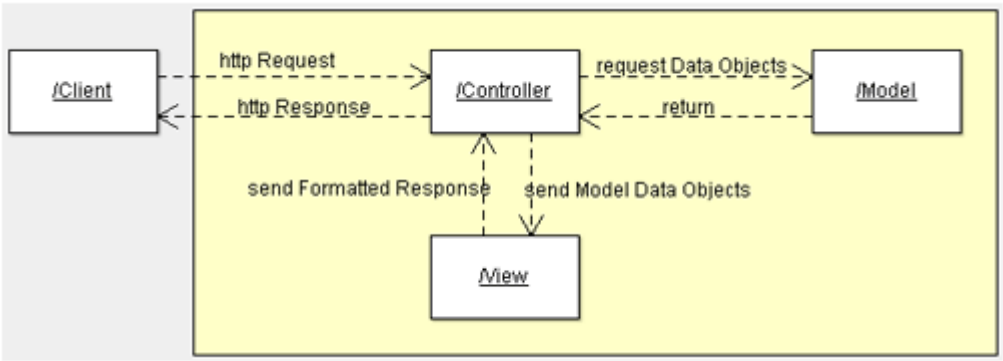

Рисунок 2.7 – Загальна схема структури MVC

Model-view-controller (Модель-представлення-контролер) – це схема використання декількох шаблонів проектування, за допомогою яких модель даних програми, інтерфейс користувача і взаємодія з користувачем розділені на три окремих компонента так, що модифікація одного з компонентів надає мінімальний вплив на інші [3]. Дана схема проектування часто використовується для побудови архітектурного каркасу, коли переходять від теорії до реалізації в конкретній предметній області.

1) Контролер

Контролер у MVC виконує роль такого собі диспетчера-регулювальника. Він не повинен змінювати або додавати дані, проводити розрахунки і т.д. Його мета – обслуговувати запити, що надходять і на кожен такий запит відповідним чином відповідати. Наприклад, користувач

звертається до головної сторінки сайту. Його запит в першу чергу отримує певний контролер (якщо розглядати на прикладі шаблону MVC). Після отримання запиту він виконує дію, визначену на даний тип запиту. Причому сам контролер ні в якому разі не повинен містити код для вибірки даних і т.д. Сама вибірка повинна бути організована в моделі.

2) Модель

У моделі описується вся бізнес – логіка програми. Вибірки, модифікація даних, розрахунки - все це повинно виконуватися в моделі, до якої і буде звертатися контролер. Причому всі згенеровані в моделі дані не повинні містити розмітку представлення. Тільки дані в сирому вигляді і нічого більше.

3) Представлення

Останній компонент архітектури MVC – представлення. Цей елемент відповідає за виведення даних в потрібному вигляді. Саме в них і повинна бути вся розмітка. Безсумнівно, в них також може міститися і програмний код, відповідальний суто за виведення даних, отриманих від контролера. Але ніякої бізнес-логіки там бути не може.

Перспективи і плюси використання MVC очевидні. Крім вирішення банального завдання на зразок відділення бізнес-логіки від інтерфейсу, ми автоматично отримуємо можливість комфортної командної розробки. Одні розробники можуть займатися вдосконаленням моделей, інші – готувати подання і т.д.

2.3 Життєвий цикл інформаційної системи

Життєвий цикл інформаційної системи є базовими елементом концепції проектного аналізу. Життєвий цикл інформаційної системи – це час від першої затрати до останньої вигоди проекту [4]. Він відображає розвиток проекту, роботи, які проводяться на різних стадіях підготовки, реалізації та експлуатації проекту. До поняття життєвого циклу інформаційної системи входить визначення різних стадій розробки й реалізації проекту.

Життєвий цикл інформаційної системи являє собою певну схему або алгоритм, за допомогою якого відбувається встановлення певної послідовності дій при розробці та впровадженні проекту .

Ступінь деталізації і термінологія опису відповідних процедур залежать від характеру проекту, предметної культури, поставлених завдань, наявних ресурсів і, можливо, уподобань та смаків проектного аналітика [5].

Головне в процесі виділення фаз, стадій та етапів проекту полягає у позначенні деяких контрольних точок, під час проходження яких використовується додаткова (зовнішня) інформація і визначаються або оцінюються можливі напрями проекту. В будь – якому разі, прийнятий поділ відображає взаємодію проекту з середовищем (діючий механізм регулювання економіки країни, політики держави, існуюче становище в економіці тощо).

Реалізація проекту вимагає виконання певної кількості різноманітних заходів і робіт, які для зручності розгляду можна поділити на дві групи: основна діяльність і діяльність із забезпечення проекту [6]. Такий поділ є поділом процесу реалізації проекту на фази і стадії, оскільки ці діяльності часто зберігаються в часі.

До основної діяльності звичайно відносять аналіз проблеми, формування цілей проекту, базове та детальне проектування, виконання будівельно-монтажних і пусконалагоджувальних робіт, здавання проекту, експлуатацію проекту, ремонт, обслуговування та демонтаж обладнання тощо.

Діяльність по забезпеченню проекту, в свою чергу, може бути поділена на організаційну, правову, кадрову, фінансову, матеріально – технічну, комерційну та інформаційну.

Чіткого й однозначного розподілу цих робіт у логічній послідовності та у часі за можливою кількістю проектів не існує (відповідно і фаз та етапів виконання проекту), оскільки визначальними є цілі й умови проекту.

У зарубіжній літературі з аналізу та управління проектами використовуються різні підходи при поділі реалізації проекту на фази. Так, у Німеччині переважає підхід, що ґрунтується на основній діяльності, аналізі проблем, розробці концепції та детальному поданні проекту, використання результатів його реалізації, ліквідації об’єктів проекту.

У публікаціях деяких російських авторів пропонується розглядати три фази проекту – концептуальну, контрактну і фазу реалізації проекту. З огляду на запропоноване розрізнення концептуальна фаза має такі стадії: розробка концепції проекту, оцінка життєдіяльності проекту, планування проекту,

розробка вимог до проекту, вибір і придбання земельної ділянки. Контрактна фаза включає вироблення кваліфікаційних вимог, підготовку попереднього завдання на проектування, заяву про наміри, добір потенційних виконавців, оформлення контракту з обраними виконавцями, вибір і затвердження остаточного варіанту проекту. Фаза реалізації проекту має дві стадії – детальне проектування та поставки; будівництво або інсталяція.

Програмою промислового розвитку ООН (UNIDO) запропоновано своє бачення проекту як циклу, що складається з трьох окремих фаз – передінвестиційної, інвестиційної та експлуатаційної.

Передінвестиційна фаза має такі стадії: визначення інвестиційних можливостей, аналіз альтернативних варіантів і попередній вибір проекту – попереднє техніко – економічне обґрунтування, висновок по проекту і рішення про інвестування. Інвестиційна фаза має такі стадії: встановлення правової, фінансової та організаційної основ для здійснення проекту, придбання і передача технологій, детальне проектне опрацювання і укладання контрактів, придбання землі, будівельні роботи і встановлення обладнання, перед виробничий маркетинг, набір і навчання персоналу, здача в експлуатацію та запуск.

Фаза експлуатації розглядається як у довгостроковому, так і в короткостроковому планах. У короткостроковому плані вивчається можливе виникнення проблем, пов’язаних із застосуванням обраної технології, функціонуванням обладнання або з кваліфікацією персоналу. У плані, довгостроковому, до розгляду береться обрана стратегія та сукупні витрати на виробництво і маркетинг, а також надходження від продажу.

Універсальним підходом до визначення робіт, які відносяться до різних фаз і стадій ЦП, є підхід Всесвітнього банку.

На рис. 2.8 показано шість стадій, які відіграють важливу роль у більшості проектів. Це ідентифікація, розробка, експертиза, переваги, реалізація та завершальна оцінка.

Ці стадії об’єднані в дві фази: фаза проектування – перші три стадії; фаза впровадження – останні три стадії. Розглянемо їх докладніше далі на рис. 2.8.

Перша стадія циклу – ідентифікація – стосується вибору або генерування таких ґрунтовних ідей, які можуть забезпечити виконання важливих завдань розвитку. На цій стадії слід скласти перелік всіх можливих ідей, придатних для досягнення цілей економічного розвитку. На подальших стадіях циклу проекту ці та інші ідеї буде уточнено і піддано дедалі

ретельнішому аналізові в міру просування по стадіях проекту з метою остаточного визначення тієї комбінації заходів, що найкращим чином забезпечить досягнення цілей проекту. Ідеї, відображені на першій стадії, повинні відповідати деяким широким критеріям здорового глузду, а саме умовам, що прибуток від реалізації проекту перевищить витрати на його здійснення.

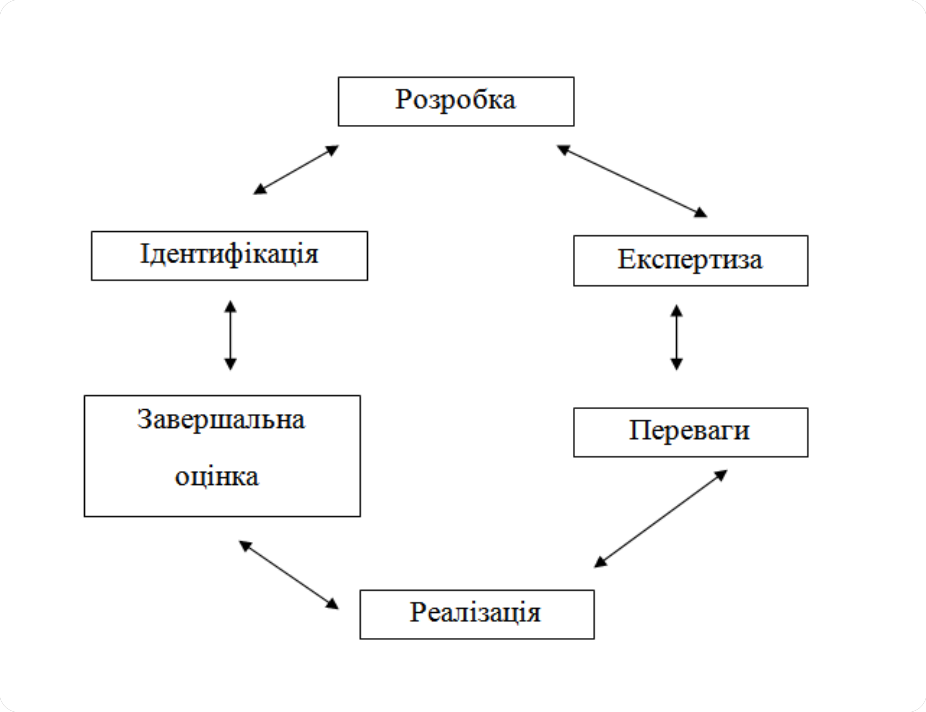

Рисунок 2.8 – Стадії життєвого циклу інформаційної системи

Таким чином, перша стадія циклу проекту виходить з чіткого формулювання цілей і тим самим утворює місток поміж аналізом здійснимості проекту. Завдання аналізу економічної політики полягає у встановленні пріоритетних цілей економічного розвитку та дослідженні тих змін у політиці й керівництві, які потрібні для виконання цих завдань. Аналіз можливості здійснення проекту передбачає оцінку цих завдань шляхом порівняння альтернативних засобів їх виконання та вибір найвигідніших варіантів.

Після того, як проект пройшов першу стадію циклу (ідентифікацію), необхідно прийняти рішення, чи варто продовжувати розгляд ідеї. Розпочинається стадія розробки. Для цього потрібне послідовне уточнення проекту за всіма його параметрами, а саме за його технічними характеристиками, врахування його впливу на довколишнє середовище, ефективності та фінансової здійснимості, прийнятності з соціальних і культурних міркувань, а також масштабності організаційних заходів.

Розробка проекту включає звуження кола запропонованих на першій стадії циклу ідей шляхом детальнішого їх вивчення [7]. Можливе проведення кількох типів досліджень, у тому числі попереднє інженерне проектування, аналіз економічної та фінансової здійснимості, розгляд систем адміністративного управління, які необхідні для успішного здійснення проекту та подальшої його експлуатації, оцінка альтернативних варіантів під поглядом захисту навколишнього середовища, оцінка впливу проекту на місцеве населення та його най вразливіші групи тощо. Чим більше ми знаємо про різні підходи до управління проектом, тим більше можливості маємо забракувати невдалі варіанти й приступити до детального вивчення обраного проекту.

Експертиза забезпечує остаточну оцінку всіх аспектів проекту перед запитом чи рішенням про його фінансування. На заключному етапі розробки проекту готується детальне обґрунтування його доцільності та здійснимості із зазначенням тих компонентів проекту, які дадуть максимальний прибуток. На стадії експертизи увага, як правило, зосереджується на оптимальному варіанті. Проводиться докладне вивчення фінансово – економічної ефективності, факторів невизначеності й ризиків, а також окремих змін у керівництві або політиці, які можуть вплинути на успіх здійснення проекту.

На стадії переговорів інвестор і замовник, який хоче одержати фінансування під проект, докладають зусиль для того, щоб дійти згоди щодо заходів, необхідних для забезпечення успіху проекту. Досягнуті домовленості потім оформлюються як документально застережені юридичні зобов’язання. Після проведення переговорів складається протокол намірів, меморандум або інші документи, що відображають досягнуті домовленості.

Під реалізацією проекту розуміють виконання необхідних робіт для досягнення його цілей. На стадії реалізації провадиться контроль і нагляд за всіма видами робіт чи діяльності в міру розвитку проекту. Порядок проведення контролю та інспекції має бути погоджено на стадії переговорів.

На стадії завершальної оцінки визначається ступінь досягнення цілей проекту, із набутого досвіду робляться висновки для його використання в подальших проектах. У перебігу цієї стадії треба порівняти фактичні результати проекту із запланованими.

2.4 Вибір архітектури

На основі аналізу аналогів, було виявлено, що для успішного функціонування системи, потрібна наявність розвиненої інформаційної системи, потрібна наявність розвиненої інформаційної системи, яка реалізує автоматизований збір, обробку і маніпулювання даними.

База даних забезпечує зберігання інформації, а також зручний і швидкий доступ до даним. Вона є сукупністю даних різного характеру, організованих по певних правилах.

На сьогодні незаперечний той факт, що сучасний рівень web – технологий дозволяє вважати їх найбільш перспективними для створення розподілених інформаційних систем. Це можуть бути не тільки відкриті інтернет – системи, але і «закриті» корпоративні автоматизовані системи управління, розподілені на великі території і відстані. Істотною особливістю таких систем є здійснення видаленого доступу до сховищ інформації – баз даних. Об'єднання інтернет – технологій і технологій СУБД як способу організації доступу до даних має ряд безумовних достоїнств і вимагає не тільки знання цих технологій, але і уміння аналізувати і вибирати оптимальну архітектуру таких інформаційних систем.

Радикальним вирішенням проблеми мережевого трафіку і інших проблем, що виникають при збільшенні об'єму даних і числа користувачів настільних СУБД, став перехід до архітектури клієнт – сервер, яка представлена на рис. 2.9.

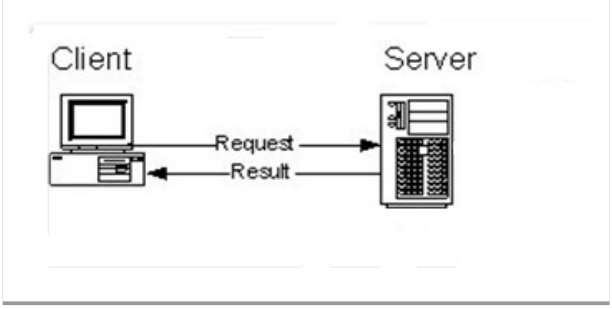

Рисунок 2.9 – Архітектура «клієнт – сервер»

Клієнт – серверна система характеризується наявністю двух взаємодіючих самостійних процесів – клієнта і сервера, які, в загальному випадку, можуть виконуватися на різних комп'ютерах, обмінюючись даними по мережі. За такою схемою можуть бути побудовані системи обробки даних на основі СУБД, поштові та інші системи. Сервер баз даних здійснює цілий комплекс дій з управління даними.

Основними його обов'язками є:

– виконання призначених для користувача запитів на вибір і модифікацію даних і метаданих;
– зберігання і резервне копіювання даних;
– підтримка цілісності даних згідно з визначеними в базі даних правилам;
– забезпечення авторизованого доступу до даних на основі перевірки і привілеїв користувачів;
– протоколювання операцій і ведення журналу транзакцій.

Як робоче місце користувача може бути використаний звичайний персональний комп'ютер, що дозволяє не відмовлятися від звичного робочого середовища [8].

Іншими словами, в простому випадку клієнт – серверна інформаційна система складається з двох основних компонентів:

– сервера баз даних, керівника даними і що виконує запити клієнтських застосувань;
– клієнтських застосувань, що надають інтерфейс користувача і що посилають запити до сервера.

Існують і складніші реалізації архітектури клієнт – сервер, наприклад багатоланкові інформаційні системи з використанням серверів додатків, що реалізовують бізнес – логіку і що здійснюють обробку даних.

Дворівнева архітектура клієнт – сервер, яка представлена на рис. 2.10 використовується в клієнт – серверних системах, де сервер відповідає на клієнтські запити безпосередньо і в повному обсязі, при цьому використовуючи лише власні ресурси. Тобто сервер не викликає сторонні мережеві додатки і не звертається до сторонніх ресурсів для виконання якої-небудь частини запиту.

Необхідність масштабованості системи стала непереборним бар'єром для традиційної дворівневої архітектури клієнт – сервер. У результаті цього на стороні клієнта чітко окреслилися проблеми, що перешкоджають досягненню справжньої масштабованості додатків: для ефективної роботи "товстого" клієнта потрібні значні обчислювальні ресурси, включаючи дисковий простір, оперативну пам'ять і потужність центрального процесора.

У 1995 році з'явився новий варіант моделі традиційної дворівневої архітектури клієнт – сервер, який був покликаний вирішити проблеми корпоративної масштабованості – (рис. 2.11).

У цій новій архітектурі пропонувалися три рівні програмного забезпечення, кожен з яких може функціонувати на різних платформах:

– рівень користувальницького інтерфейсу, який розташовується на комп'ютері кінцевого користувача (Client);
– рівень бізнес – логіки й обробки даних. Цей проміжний рівень розташовується на сервері і часто називається сервером додатків (Application server);
– СУБД, в якій зберігаються дані, необхідні для функціонування проміжного рівня. Цей рівень може виконуватися на окремому сервері бази даних (Data server).

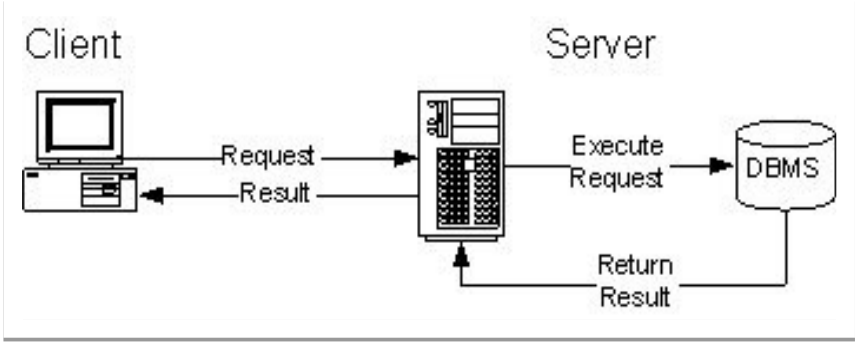

Рисунок 2.10 – Дворівнева архітектура «клієнт – сервер»

Клієнт відповідає тільки за користувальницький інтерфейс і, можливо, виконує деяку дуже просту логічну обробку даних, наприклад перевірку коректності введення даних. Обмежений подібним функціональним набором клієнт отримав назву "тонкого" клієнта. Основна бізнес – логіка додатка тепер знаходиться на власному виділеному рівні, який фізично пов'язаний з клієнтом і сервером бази даних за допомогою локальної (Local Area Network) або глобальної (Wide Area .Network) обчислювальної мережі. При цьому передбачається, що один сервер додатків може обслуговувати безліч клієнтів.

Трирівнева архітектура клієнт – сервер має багато переваг перед своїми попередниками. Нижче перераховані деякі з них: для "тонкого" клієнта потрібно менше дорогого апаратного забезпечення. Централізація бізнес – логіки для багатьох кінцевих користувачів на одному сервері програми. Завдяки цьому виключається необхідність розгортання програмного забезпечення на великій кількості комп'ютерів, що представляє собою одну з найскладніших завдань у дворівневій моделі клієнт – сервер. Додаткова модульність спрощує модифікацію або заміну програмного забезпечення кожного рівня без надання впливу на інші рівні. Відділення основної бізнес – логіки програми від функцій бази даних спрощує завдання рівномірного розподілу навантаження [9].

Додаткова перевага полягає в тому, що трирівнева архітектура досить природно відображається на середу WWW, де браузер грає роль "тонкого" клієнта, а веб – сервер – сервера додатків.

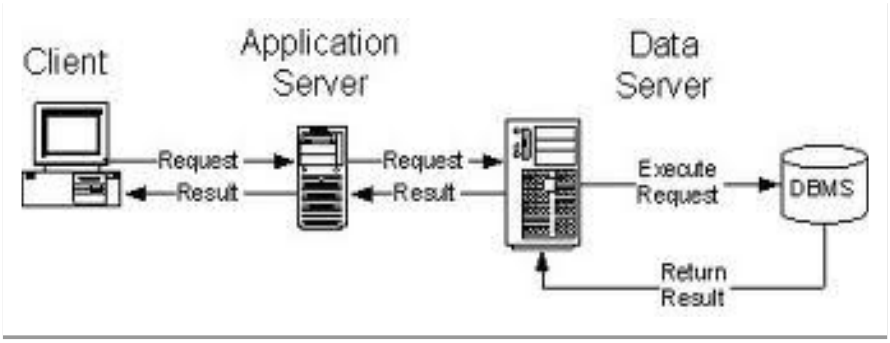

Рисунок 2.11 – Трирівнева архітектура «клієнт – сервер»

Трирівнева архітектура може бути розширена до N – рівневої архітектури. Наприклад, проміжний рівень в трирівневій архітектурі може бути розщеплений на два рівні, один з яких може виконувати завдання звичайного веб – сервера, а інший – типового сервера додатків – (рис. 2.12).

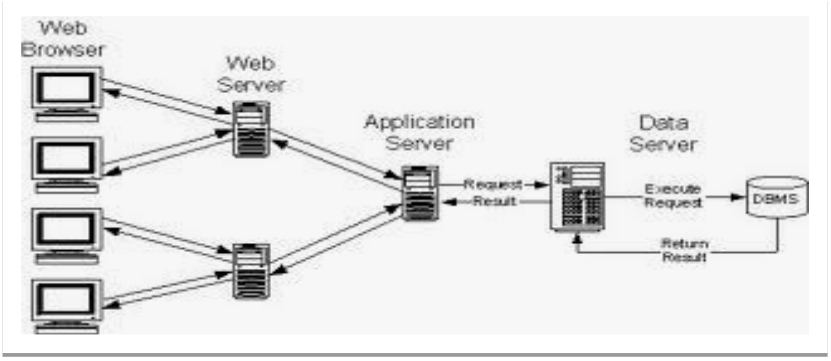

Рисунок 2.12 – N – рівнева архітектура «клієнт – сервер»

2.5 Вибір мови програмування

Для реалізації представлень, сторінок, які буде бачити користувач, будемо використовуватися HTML, CSS, Javascript.

2.5.1 Мови реалізації сценаріїв

Javascript використовується як вбудована мова для програмного доступу до об'єктів додатків [10]. Найбільш широке застосування знаходить як мова сценаріїв для додання інтерактивності web-сторінкам.

Основні архітектурні риси: динамічна типізація, автоматичне керування пам'яттю, прототипне програмування, функції як об'єкти першого класу.

За допомогою мови javascript, домоглися гарних результатів, як оформлення, так і функціональності системи. Незважаючи на те, що javascript дозволяє динамічно змінювати вигляд сторінок, він ще може допомогти і зі зміною конкретних даних на Web-вузлі.

Так як на сьогоднішній день існує безліч добрих фреймворків і бібліотек для роботи з Javascript, що надають багатий додатковий функціонал, реалізацію багатьох корисних функцій і шаблонів, використання чистого Javascript не є доцільним. У проекті передбачається великий обсяг роботи з таблицями, формами, перевіркою даних. Тому була обрана бібліотека Ext JS.

Ext JS, у свою чергу, вже має багато реалізацій цих, а так само інших корисних для нас функцій. Так само, бібліотека має функції для зручної реалізації AJAX-запитів на сервер з їх подальшою обробкою.

Ext JS — бібліотека JavaScript для розробки веб-додатків і інтерфейсів користувача, спочатку задумана як розширена версія Yahoo! UI Library, однак перетворений потім в окремий фреймворк. До версії 4.0 використовувала адаптери для доступу до бібліотек Yahoo! UI Library, jQuery або Prototype / script.aculo.us, починаючи з 4-ої версії адаптери відсутні. Підтримує технологію AJAX, анімацію, роботу з DOM, реалізацію таблиць, вкладок, обробку подій і всі інші нововведення Web 2.0.

З версії 2.1 бібліотека Ext JS передбачає подвійне ліцензування: може бути поставлена за ліцензією GPL v3, або за комерційною ліцензією компанії Sencha [11].

Починаючи з версії Ext JS 3.0 бібліотека розбивається на дві частини: Ext Core (набір JavaScript-функцій, що дозволяє створювати динамічні веб-сторінки з уніфікацією обробки в різних браузерах і поширюваний по MIT-ліцензії) та Ext JS (власне набір віджетів для створення користувацьких інтерфейсів з подвійним ліцензуванням по GPL v3 або за комерційною ліцензією).

Компанія, що розробляє і підтримує фреймворк Ext JS, спочатку носила найменування Ext, LLC. 14 червня 2010 розробники jQTouch і Raphaël приєдналися до компанії Ext LLC, і компанія змінила найменування на Sencha, Inc., А Ext JS, зберігши свою назву, ставши одним з продуктів оновленій компанії.

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

Бібліотека jQuery містить функціональність, корисну для максимально широкого кола завдань. Була реалізована архітектура компактного універсального ядра бібліотеки і плагінів [12]. Це дозволяє зібрати для ресурсу саме ту JavaScript-функціональність, яка на ньому була б затребувана.

Бібліотека jQuery володіє наступними перевагами:

– невеликий розмір файлу (близько 91 Кбайт для версії 1.7);
– надзвичайно простий синтаксис;
– можливість об'єднання послідовно викликаних методів в ланцюги;
– проста архітектура модулів, що розширюють базові можливості фреймворка;
– величезне мережеве співтовариство користувачів;
– корисні розширення, такі як jQuery UI, що надають додаткову функціональність.

2.5.2 Мова розмітки гіпертексту HTML

Також при розробці системи була використана мова розмітки гіпертексту – Hyper Text Markup Language (HTML) призначений для написання гіпертекстових документів, що публікуються в World Wide Web.

Гіпертекстовий документ – це текстовий файл, що має спеціальні мітки, звані тегами, які згодом розпізнаються браузером і використовуються ним для відображення вмісту файлу па екрані комп'ютера.

За допомогою цих міток можна виділяти заголовки документа, вимірювати колір, розмір і накреслення літер, вставляти графічні зображення і таблиці [13]. Але основною перевагою гіпертексту перед звичайним текстом є можливість додавання до вмісту документа гіперпосилань – спеціальних конструкцій мови HTML, які дозволяють клацанням миші перейти до перегляду іншого документа.

HTML-документ складається з двох частин: власне тексту, тобто даних, складових вміст документа, і тегів – спеціальні конструкції мови HTML, використані для розмітки документа і керуючих його відображенням. Теги мови HTML визначають, в якому вигляді буде представлений текст, які його

компоненти будуть виконувати роль гіпертекстових посилань, які графічні або мультимедійні об'єкти повинні бути включені в документ [14].

Графічна та звукова інформація, що включається в HTML-документ, зберігається в окремих файлах. Програми перегляду HTML-документів (браузери) інтерпретують прапори розмітки і розташовують текст і графіку на екрані відповідним чином. Для файлів, що містять HTML-документи, прийняті розширення .htm або .html.

HTML – це не мова програмування, вона служить лише для розмітки сторінок, додання певного виду того чи іншого елементу, будь то таблиця, текст або картинки.

Здійснюється це шляхом присвоєння кожному елементу своїх параметрів, які розпізнає браузер. Параметри ці можуть бути задані як для одного, так і для групи або типу елементів

2.5.3 Формальна мова розмітки CSS

Також була використана формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки – CSS (англ. Cascading Style Sheets – каскадні таблиці стилів) [15]. Переважно використовується як засіб опису, оформлення зовнішнього вигляду веб-сторінок, написаних за допомогою мов розмітки HTML і XHTML, але може також застосовуватися до будь-яких XML-документах, наприклад, до SVG або XUL.

CSS використовується для завдання кольорів, шрифтів, розташування відділених блоків та інших аспектів представлення зовнішнього вигляду веб-сторінок. Основна мета розробки CSS – це розділення опису логічної структури від опису зовнішнього вигляду Wев-сторінки [16]. Такий поділ може збільшити доступність документа, надати велику гнучкість і можливість управління його поданням, а також зменшити складність і повторюваність в структурному вмісті. Крім того, CSS дозволяє представляти один і той же документ в різних стилях або методах виведення, таких як екранне уявлення, друковане уявлення, читання голосом.

Головні переваги CSS:

– більш чистий код;
– цей код легше підтримувати;
– швидше завантажується;
– краще оптимізований для пошукових систем;

- модульний код;
- правила стилю можуть застосовуватися до безлічі сторінок;
- однаковий дизайн.

3 ЕСКІЗНИЙ ПРОЕКТ

Грунтуючись на прийнятому рішенні реалізувати проект у вигляді веб-додатку, введення\виведення даних відбуватиметься через веб-інтерфейс. За допомогою AJAX-запитів сторінки будуть обмінюватися даними з відповідними оброблювачами на сервері.

3.1 Ескізи сторінок

3.1.1 Сторінка авторизації(Login Page)

Попередній ескіз сторінки представлений на рис. 3.1

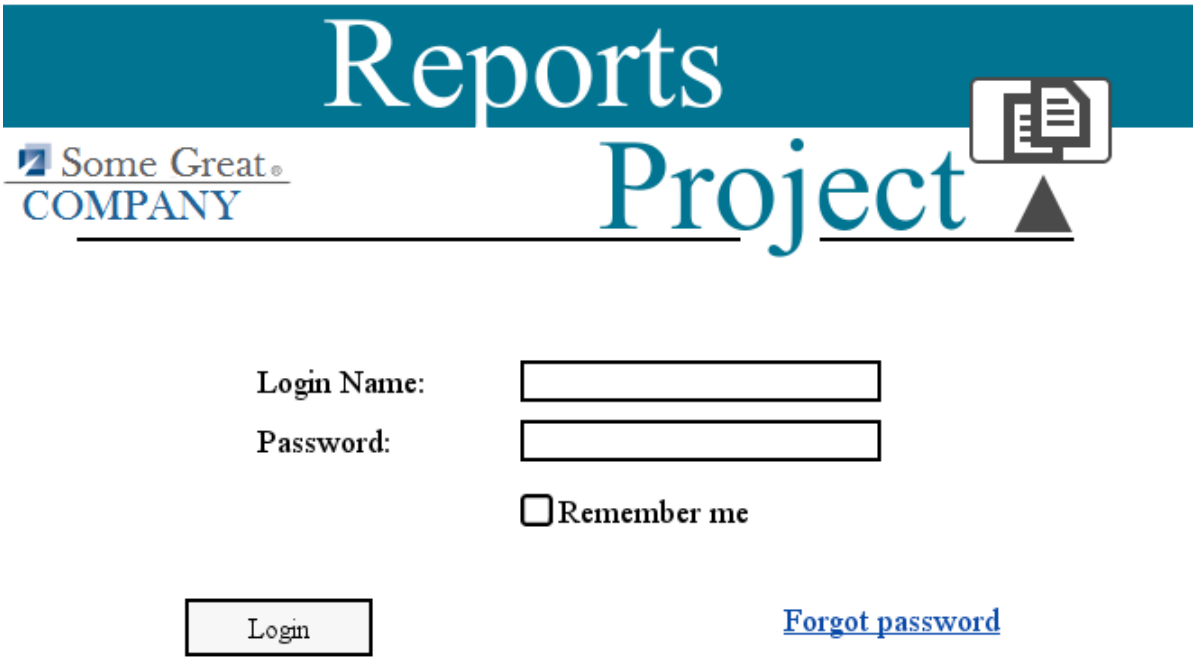

Рисунок 3.1 – Сторінка авторизації (Login Page)

На сторінці є поля для авторизації користувача. Після введення даних у поля і натиснення кнопки "Login", дані з полів за допомогою відповідного обробника збираються в єдиний об'єкт і AJAX-запитом відправляються на сервер для подальшої обробки. Дані відправляються у вигляді JSON-об'єкта.

Поля для введення необхідно реалізувати за допомогою компонентів ExtJs – inputForm. У них передбачений функціонал додавання напису (Login Name, Password).

Кнопку Login необхідно реалізувати за допомогою ExtJs button. Натискання на неї призведе до запуску механізму авторизації користувача.

Елемент "Forgot Password" – звичайне html посилання. Скориставшись ним, користувач відкриє вікно відновлення паролю.

Елемент "Remember me" – html checkbox. З його допомогою користувач може зберегти сесію.

3.1.2 Сторінка перегляду і редагування звіту (Report Page)

Сторінка перегляду звітів. На сторінці розташовані поля для введення початковою і кінцевою дати і список, що випадає з іменами branches (рис. 3.2).

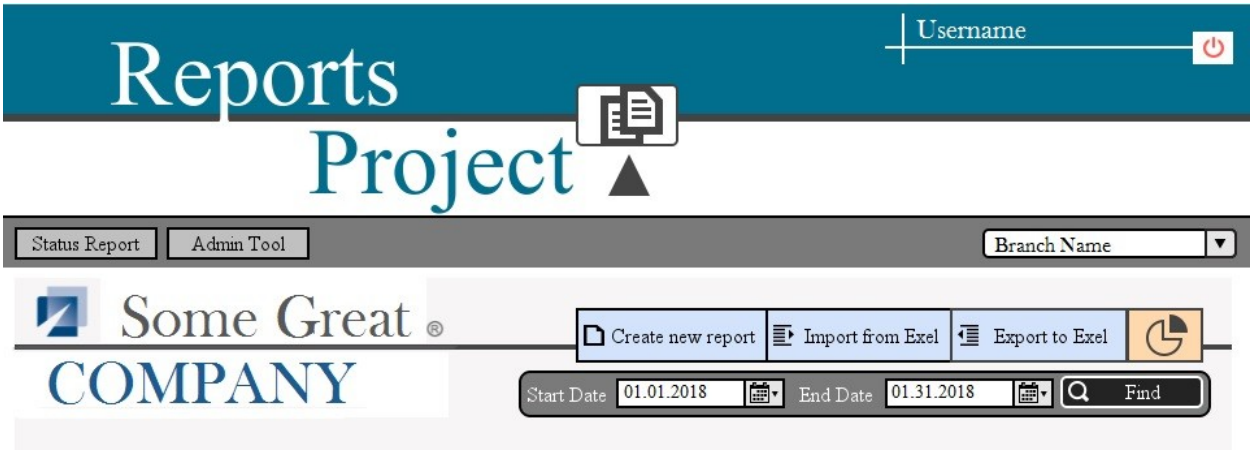

Рисунок 3.2 – Сторінка пошуку звіту за конкретним проміжком часу

У цих полях ми вводимо проміжок часу, у який було додано цікавий для нас звіт. Також, для спрощення пошуку, ми можемо вибрати з випадаючого списку цікавий для нас регіон чи офіс. Після натискання кнопки "Find", дані з цих полів збираються і AJAX-запитом відправляються на сервер для отримання звітів. Дані передаються у вигляді JSON-об'єкта. Після обробки отриманих даних, сервер повертає звіти у вигляді JSON-об'єкта, які потім обробляються вже на стороні клієнта за допомогою javascript і відображаються в таблицях. При цьому деталі звітів повинні бути розбиті за відповідними вкладках нижньої таблиці. Ескіз сторінки представлений на рис. 3.3.

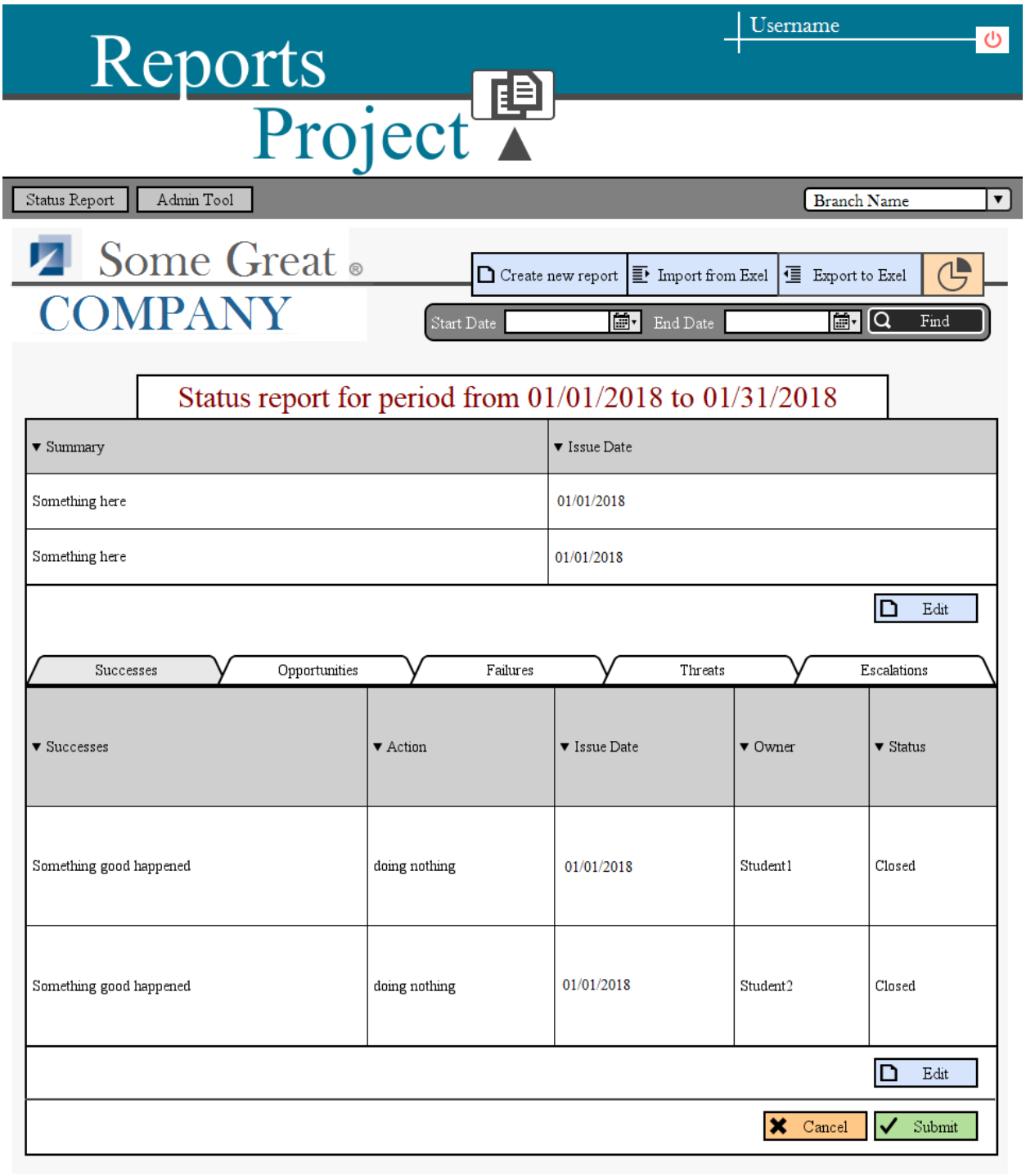

Рисунок 3.3 – Сторінка перегляду і редагування звіту (Report Page)

Якщо нам потрібно відредагувати якусь інформацію (звісно якщо ми маємо на те права), ми можемо зробити це двома способами:

1) робимо подвійне клацання на комірці таблиці, і ряд, якому належить комірка, стає редагованим;

2) натискаємо кнопку "Edit", і ряд ,обраний нами до цього, стає редагованим. У разі, якщо ряд не був обраний – редагованим стає перший ряд в таблиці.

При натисканні на кнопку "Submit", дані відправляються на сервер і зберігаються у базі. По завершенню операції – з'являється спливаюче вікно з відповідною інформацією про успішне (рис. 3.4) або неуспішне (рис. 3.5) завершенні операції (у разі помилок або проблем).

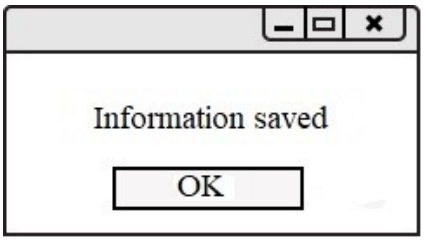

Рисунок 3.4 – Повідомлення про успішну операцію

Рисунок 3.5 – Повідомлення про помилку при збереженні інформації

3.1.3 Сторінка імпорту звітів (Import Page)

Сторінка "Import Page" призначена для завантаження нових звітів. При натисканні на посилання “Import Page”, ми бачимо стандартне вікно ОС Windows для завантаження файлів. У ньому ми можемо вибрати кілька звітів, після чого завантаження розпочинається в автоматичному режимі.

Після завантаження файлу, його ім'я і дата завантаження будуть відображені у вікні, а також з’явиться посилання "Add another one” (рис. 3.6).

Сторінка надає користувачеві наступні можливості:

- завантаження декількох звітів одночасно;
- верифікація даних у звітах під час завантаження – програма перевіряє заповнення полів і показувати відповідну помилку, у разі, якщо якихось даних не вистачає.

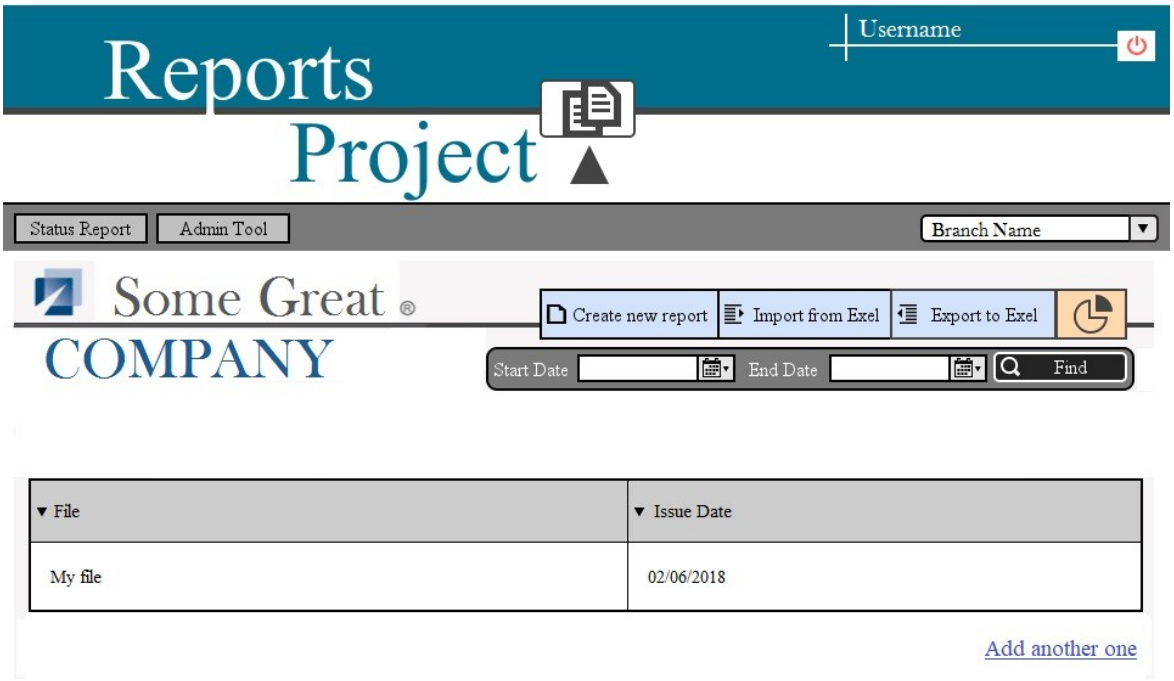

Рисунок 3.6 – Сторінка імпорту звітів

3.1.4 Сторінка створення звіту (Create Report Page)

Сторінку створення звіту можна побачити на рис. 3.7.

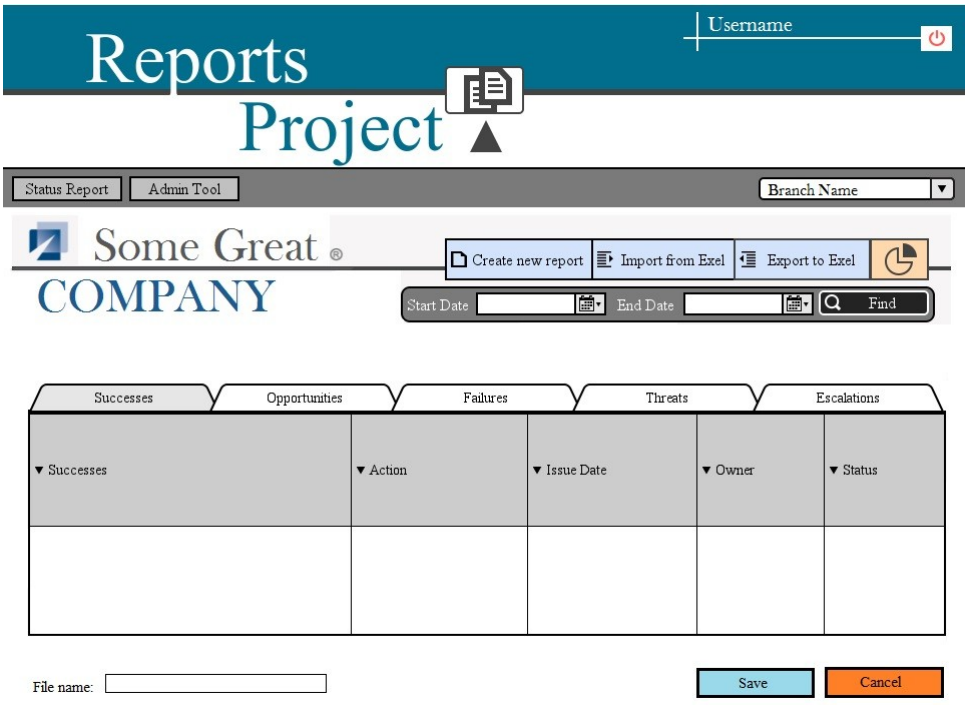

Рисунок 3.7 – Сторінка створення звіту

За допомогою "Create new report" користувач зможе перейти на сторінку створення звіту.

При цьому йому буде запропоновано:

- поле для введення імені файлу;
- стандартна пуста форма для заповнення необхідною інформацією;
- кнопка “Save” для збереження звіту;
- кнопка “Cancel” для відміни створення нового звіту.

По натисканню на кнопку "Cancel" з'явиться спливаюче вікно, в якому користувач може підтвердити скасування ствоення звіту або ж продовжити його створення у разі, якщо кнопка була натиснута випадково. При натисканні кнопки "Ok" у спливаючому вікні, воно має бути закрите, а користувач перенаправлений на сторінку "Report Page".

3.1.5 Сторінка експорту звіту (Export to Excel)

Сторінка "Export to Excel" дозволить вивантажити знайдені звіти (рис. 3.8).

Рисунок 3.8 – Сторінка експорту звіту

При натисканні на неї ми бачимо список доступних для нас файлів у порядку їх завантаження. Обравши потрібний файл натисканням кнопки миши та натиснувши на кнопку “Export file” ми експортуємо даний файл у вигляді xlsx файлу.

Панель для вибору дати – ще один Ext.Container. Написи "Start Date", "End Date" необхідно виконати звичайним html. Форми для вибору дат – Ext.form.DateField. Валідацію необхідно реалізувати спеціальними оброблювачами. Кнопка "Find" – html-посилання. За допомогою CSS необхідно надати їй вигляду як на макеті.

Кнопки "Create new report", "Import from Excel", "Export to Excel" необхідно помістити в окремий Ext.Container. Елементи в контейнері отримають спеціальну html-обгортку. Самі кнопки необхідно виконати у вигляді html-посилань. За допомогою CSS потрібно створити вигляд як на макеті.

3.1.6 Сторінка пошуку профілю користувача (Search User Page)

Ескіз сторінки пошуку профілю користувача представлений на рис. 3.9.

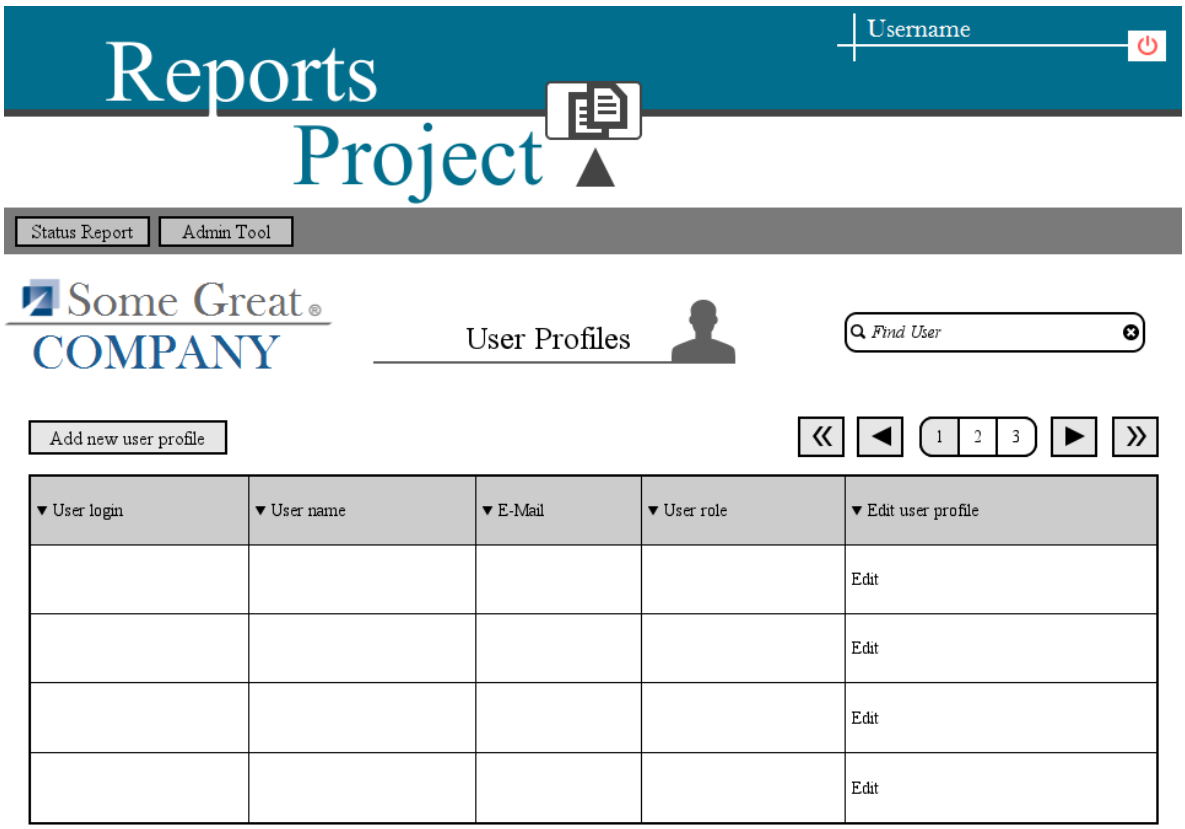

Рисунок 3.9 – Сторінка пошуку профілю користувача (Search User Page)

На сторінці пошуку профілю є поле для введення пошукової інформації. Після натискання клавіші "Enter", дані з поля повинні збиратися скриптом на сторінці і AJAX-запитом відправляються на сервер. Відповідні контролери на сервері повинні обробляти отриману інформацію і повертати на сторінку дані у вигляді JSON-об'єкта, де вони повинні бути відображені в таблиці.

Елемент "Find User" необхідно реалізувати за допомогою Ext.form.TextField. Іконки, стилі потрібно зробити за допомогою html та CSS. Ввівши дані в поле і натиснувши "Enter", користувач зможе провести пошук за даними в базі і побачити результат у таблиці нижче. Таблицю потрібно реалізувати за допомогою Ext.grid.GridPanel.

Елемент "Add new user profile" необхідно реалізувати за допомогою Ext.Button. При натисканні на кнопку, адміністратор зможе перейти на сторінку створення нового користувача. Перехід буде реалізований за допомогою відповідної функції-обробника натискання.

3.1.7 Сторінка додавання користувача (Add New User Page)

На сторінці користувачеві надані поля для заповнення і випадні списки для вибору «стандартної» інформації. Ескіз сторінки представлений на рис. 3.10.

Після роботи з полем при втраті фокуса на ньому, «вистрілюється» відповідна подія, яка обробляється скриптом на сторінці. Якщо дані введені вірно, то ніякої реакції не буде. У випадку помилки, поле буде виділено і користувач побачить відповідне повідомлення про помилку. Користувач не зможе зберегти дані до тих пір, доки всі помилки не будуть виправлені. У разі, якщо помилок не було або вони всі виправлені, то дані з полів і випадаючих меню будуть зібрані скриптом і передані на сервер AJAX-запитом у вигляді JSON-об'єкта.

Сторінка редагування профілю користувача (Edit User Page) аналогічна сторінці додавання користувача за зовнішнім виглядом і функціоналу.

Сторінка повинна містити такі елементи:

- поле “Логін користувача”;
- випадаючий список “Роль користувача”;
- поле “Им’я користувача”;
- поле “Прізвище користувача”;
- поле “E-mail”;

- випадаючий список “Административний юнит”;
- чекбокс “ Увімкнути профіль користувача ”;
- поле “ Введіть новий пароль користувача ”;
- поле “ Підтвердити новий пароль користувача ”;
- посилання “ Назад на сторінку пошуку профілю користувача ” ;
- кнопка “Додати новий”;
- кнопка “Зберігти”.

Рис. 3.10 – Сторінка додавання користувача (Add New User Page)

3.1.8 Сторінка-обробник невірної адреси (Invalid URL Page)

Сторінка-обробник невірної адреси показує користувачеві інформацію, якщо він ввів некоректний адресу. У цьому випадку ми маємо два варіанти розвитку ситуації:

1) Повернутись до сторінки авторизації, натиснувши на клавішу “Back to Login Page”;
2) Відновити пароль за допомогою клавіши “Forgot password”.

Ескіз сторінки представлений на рис. 3.11.

Рисунок 3.11 – Сторінка-обробник невірної адреси (Invalid URL Page)

3.2 Ескізи спливаючих вікон

3.2.1 Спливаюче вікно відновлення паролю (“Forgot password”)

Ця сторінка необхідна якщо користувач забув свій пароль. Спливаюче вікно відновлення паролю (“Forgot password”) очікує введення інформації у відповідне поле. Після введення ім’я користувача, по натисканню на кнопку "Ok", код на сторінці збирає дані і відправляє їх на сервер AJAX-запитом у вигляді JSON-об'єкта. На сервері перевіряється наявність такого логіна в системі і, в разі його наявності, користувачеві на пошту висилається лист з посиланням для відновлення пароля. Ескіз спливаючого вікна представлений на рис. 3.12.

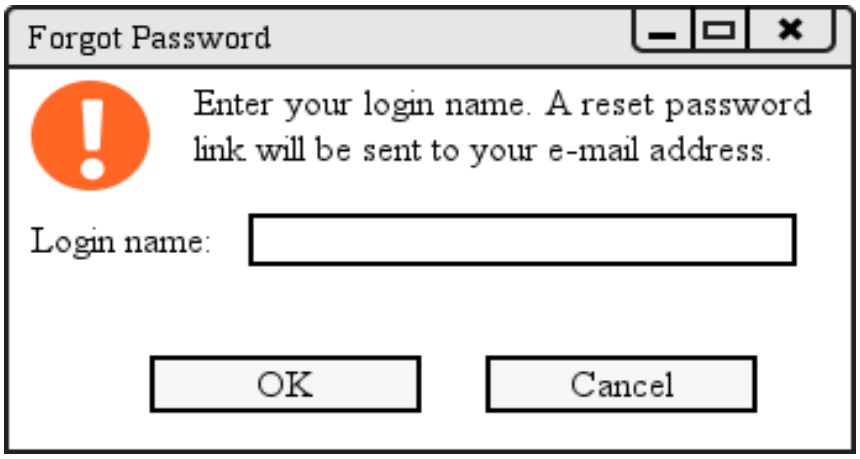

Рисунок 3.12 – Ескіз спливаючого вікна відновлення пароля (“Forgot password”)

3.2.2 Спливаюче вікно для додавання нового адміністративного юніт офісу підприємства, branch (“Add New Administrative Unit”)

Спливаюче вікно для додавання нового адміністративного юніта-офісу підприємства, branch (“Add New Administrative Unit”) очікує введення інформації у відповідне поле. По натисканню на кнопку "Ok", код на сторінці збирає дані і відправляє їх на сервер AJAX-запитом у вигляді JSON-об'єкта. На сервері дані обробляються і зберігаються в базу.

Ескіз спливаючого вікна представлений на рис. 3.13.

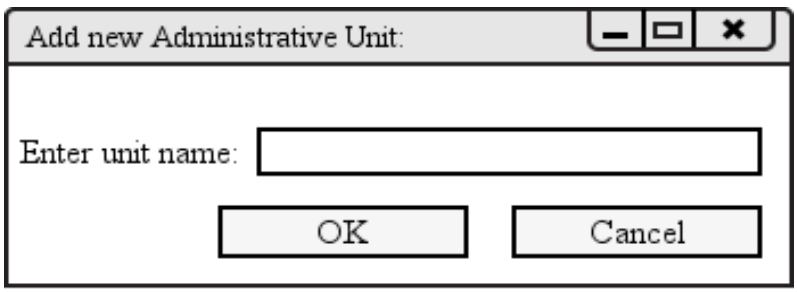

Рисунок 3.13 – Спливаюче вікно для додавання нового адміністративного юніта-офісу підприємства, branch (“Add New Administrative Unit”)

ВИСНОВКИ

Електронний документообіг – це життєвий цикл електронних документів в організації, починаючи від їх отримання, проходження у підрозділах і закінчуючи списанням в архів. Часто електронний документообіг позначається терміном workflow, який характеризує рух документів як потік робіт, виконуваних в рамках того або іншого бізнес-процесу. Система електронного документообігу – це програмне забезпечення, головними завданнями якого є організація та підтримка життєвого циклу електронних документів.

Темою магістерської роботи є розробка системи для обороту звітності для підприємств з географічно-розгалуженою структурою.

У ході роботи були досліджені існуючі рішення, було проведено їх аналіз, виявлені переваги і недоліки. На цій підставі було прийнято рішення про створення нової системи.

Для досягнення зазначеної мети було представленно ряд завдань, які повинна виконувати система:

- консолідувати існуючі види тижневих звітів філій підприємств;
- створити єдиний репозиторій звітів для контролю діяльності офісів;
- здійснювати операції експорту / імпорту із звітними формами і зняти ризик помилок (людський фактор);
- надати зручний, інтуїтивно-зрозумілий і красивий інтерфейс до системи

На основі завдань, які повинна вирішувати система, був описаний функціонал, який необхідно було реалізувати, складені макети сторінок.

Був проведений аналіз технічних засобів, якими можна реалізувати описані функції системи.

Для реалізації графічного інтерфейсу користувача були використані сучасні програмні засоби розробки, а саме:

- HTML;
- CSS;
- Ext JS (бібліотека JavaScript для розробки веб-додатків і інтерфейсів користувача);
- jQuery (бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML).

Результатом проекту стала форма звіту зі зручним інтерфейсом, що дозволяє створювати, редагувати і переглядати існуючі звіти. Програма

підтримує роботу зі звітами в MS Excel і надає багаторолевий доступ до функціональності проекту, вирішуючи поставлені цілі і завдання.

ПЕРЕЛІК ПОСИЛАНЬ

1. Богданов К.О., Порай Д.Д. Робота з документами. – СПб.: Едиторіал УРСС, 2000. – 386 с.

2. Кузнецов І. Документальне забезпечення управління. – СПб.: Юрайт, 2016. – 576 с.

3. Відеоуроки з створення Інтернет-систем [Електроний ресурс] – Режим доступу: http://www.ruseller.com/

4. Тереза Нейл, Білл Скотт. Проектування веб-інтерфейсів. – СПб.: Символ-Плюс, 2010. – 352 с.

5. Попов Є.Ю. JavaScript та JQuery. Відеоуроки [Електроний ресурс] – Режим доступу: http://www.evgeniypopov.ru/

6. Кірсанов Д. Веб-дизайн: книга Дмитра Кірсанова. – СПб.: Символ-плюс, 1999. – 376 с.

7. Нільсен, Хоа Лоранжер. Web-дизайн. Зручність використання Web-сайтів. – СПб.: Символ-плюс, 2003. – 512 с.

8. Стів Макконнелл. Досконалий код. Майстер-клас. – СПб.:, 2017. – 896 с.

9. Джеффрі Фрідл. Регулярні вирази. – СПб.: Символ-Плюс, 2008. – 608 с.

10. Фленаган Девід. JavaScript. Детальне керівництво. – СПб.: Символ-Плюс, 2013. – 1080 с.

11. Пошукова система гугл [Електронний ресурс] – Режим доступу до сайту: https://www.google.com.ua/

12. Ієгуда Кац. jQuery. Докладне керівництво по просунутому JavaScript. – СПб.: Символ-Плюс, 2011. – 624 с.

13. Головний екран Wikipedia [Електроний ресурс] − Режим досупу: http://ru.wikipedia.org/wiki/Main_Page

14. Террі Фельке-Моррис. Велика книга веб-дизайну. – СПб.: Litres, 2017. – 602 с.

15. Хеник Бен. HTML і CSS. Шлях до досконалості. – СПб.: Пітер, 2010. – 336 с.

16. Збірник статей [Електронний ресурс] – Режим доступу до сайту: http://habrahabr.ru/

Д О Д А Т К И

ДОДАТОК А ЛІСТИНГ JAVASCRIPT-КОДУ СТОРІНКИ User Search Page

```
var userName;
var span;
var userStore = new Ext.data.ArrayStore({
autoDestroy:true,
storeId:"userStore",
fields:[
{ name:"login", type:"string" },
{ name:"branch", type:"string" },
{ name:"userName", type:"string" },
{ name:"email", type:"string" },
{ name:"userRole", type:"string" } ,
{ name:"id" }
]
});
var searchPhrase = "";
var searchButtonClick = function ()
{
if (searchPhrase.search(new RegExp("[a-zA-Z]")) != -1)
Ext.Ajax.request({
url:"getUsersBySearchPhrase",
params:{ searchPhrase:searchPhrase },
method:"GET",
success:function (response, opts)
{
var users =
Ext.decode(response.responseText.replace(/"userRole":"ROLE_ADMIN"/g,
'"userRole":"Administrator"').replace(/"userRole":"ROLE_EXECUTOR"/g,
'"userRole":"Executor"').replace(/"userRole":"ROLE_SUPERVISOR"/g,
'"userRole":"Supervisor"'));
var usersArray = [];
for (var i = 0; i < users.length; i++)
{
usersArray.push([]);
usersArray[i][0] = users[i].login;
usersArray[i][1] = users[i].branch;
usersArray[i][2] = users[i].userName;
usersArray[i][3] = users[i].email;
usersArray[i][4] = users[i].userRole;
usersArray[i][5] = users[i].id;
}
userStore.loadData(usersArray);
},
failure:function (response, opts)
{
console.log('server-side failure with status code ' + response.status);
}
});
}
Ext.onReady(function () {
Ext.form.Field.prototype.msgTarget = 'side';
//--------------Head part-----------------------
//search field for users
var searchButton = new Ext.Button({
cls: 'no-style search-button'
,text: '<p class="icon-search" onclick="searchButtonClick()"></p>'
});
// reset button for search field
var resetButton = new Ext.Button({
cls:'no-style reset-button'
,text: '<p class="icon-remove-sign"></p>'
```

```
});
var searchUserField = new Ext.form.TextField({
cls:'search-field',
emptyText:'Find User',
enableKeyEvents:true,
getErrors: function(value){
if(value.search(new RegExp("[a-zA-Z]"))==-1)
return ["Search phrase should contain at least one letter."];
else return [];
},
listeners:{
change:function (field, newValue, oldValue)
{
searchPhrase = newValue;
},
keypress:function (field, e)
{
if (e.getKey() == 13)
{
searchPhrase = field.getValue();
searchButtonClick();
}
}
}
});
var headPartContainer = new Ext.Container({
id:'headPart-container',
height:100,
autoEl:'div',
items: [
{
cls:'headPart-search',
items: [searchButton,searchUserField,resetButton]
}
]
});
var manIconContainer = new Ext.Panel({
id: 'search-man-icon-container',
items:[
{
cls:'headPartImg2',
html:'<img id="formImage2" src="resources/images/userProfiles.png"/>'
}
]
});
//--------------eo Head part-----------------------
//--------------Above table part-----------------------
// Add new user profile button
var addNewUserProfileButton = new Ext.Button({
id:'addNewUserProfile-button',
cls:'no-style greenButton submitButton',
overCls:'greenButton-hover',
height: 45,
pressedCls:'greenButton-active',
text:'<p class="buttonText icon-plus"> Add user</p>',
listeners:{
click:function () {
window.location = 'addNewUser';
}
}
});
var editButton = new Ext.Button({
cls:'no-style blueButton editButton',
```

```
overCls:'blueButton-hover',
pressedCls:'blueButton-active',
height: 45,
text:'<p class="buttonText icon-edit"> Edit</p>',
listeners:{
click:function () {
if(gridSearch.getSelectionModel().getSelected()!=undefined)
window.location = window.location.href.replace("#",
"").replace("userSearchPage", "") + "editNewUserProfile?" +
gridSearch.getSelectionModel().getSelected().data.id;
}
}
})
var bottomButtonHolder = new Ext.Panel({
cls:'search-page-bottom-button-holder',
width: 865,
items: [
addNewUserProfileButton,
editButton
]
});
//--------------Table part-----------------------
var data = [
['','', '', '']
];
var gridSearch = new Ext.grid.GridPanel({
store: userStore,
autoHeight:true,
viewConfig:{forceFit:true},
selModel:new Ext.grid.RowSelectionModel({singleSelect:true}),
cm:new Ext.grid.ColumnModel({
columns:[
{ header:"User Login", dataIndex:"login" },
{ header:"Branch", dataIndex:"branch"},
{ header:"User name", dataIndex:"userName" },
{ header:"E-mail", dataIndex:"email" },
{ header:"User role", dataIndex:"userRole" }
]
}),
listeners:
{
dblclick:function(event)
{
window.location = window.location.href.replace("#",
"").replace("userSearchPage", "") + "editNewUserProfile?" +
gridSearch.getSelectionModel().getSelected().data.id;
}
}
});
var centerContainer = new Ext.Container({
autoEl:'div', cls:'center-container', border:false, items:[
{
autoEl:'div', cls:'content', border:false, items:[
headPartContainer,
{
autoEl: 'div', cls: 'inner-content', items:
[manIconContainer, gridSearch,bottomButtonHolder]
}
]
}
,southContainer
]
});
```

```
//----------------- Center part--------------------
//Main View
var mainViewport = new Ext.Viewport({
border:false
,items:[
northContainer
,centerContainer
]
});
});
function setUserName(newText) {
userName = newText;}
```