Форма № Н-6.01

Національний аерокосмічний університет ім. М.Є. Жуковського

«Харківський авіаційний інститут»

Факультет радіотехнічних систем літальних апаратів

Кафедра комп'ютерних систем, мереж та кібербезпеки

**КУРСОВИЙ ПРОЕКТ**

**(РОБОТА)**

з Проектування МПС

(назва дисципліни)

на тему: Сервіс хмарного зберігання файлів

Студента (ки) 4 курсу 545 групи

напряму підготовки комп’ютерна інженерія

Бесчетнікова І. О.

(прізвище та ініціали)

Керівник доц. Галькевич О. О.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала

Кількість балів: Оцінка: ECTS

Члени комісії

(підпис) (прізвище та ініціали)

(підпис) (прізвище та ініціали)

(підпис) (прізвище та ініціали)

м. Харків – 2018 рік

Содержание

[**ВВЕДЕНИЕ** 3](#_Toc503907473)

[1. ПОСТАНОВКА ЗАДАЧИ 4](#_Toc503907474)

[1.1. Анализ предметной области 4](#_Toc503907475)

[2. ПРОЕКТИРОВАНИЕ 11](#_Toc503907476)

[2.1. Выбор архитектуры 11](#_Toc503907477)

[2.2. Уровень web-приложения 11](#_Toc503907478)

[2.2.1. Расширяемость 13](#_Toc503907479)

[2.2.2. HTTP и HTML 14](#_Toc503907480)

[2.3. Уровень сервиса 16](#_Toc503907481)

[2.4. Уровень хранилища данных 16](#_Toc503907482)

[2.5. UML –диаграмма использования 18](#_Toc503907483)

[2.6. Проектирование графического интерфейса: проектирование представлений Web-приложения и интерфейса программы для устройств-серверов 25](#_Toc503907484)

[3. РАЗРАБОТКА 31](#_Toc503907485)

[3.1. Разработка диаграммы классов 31](#_Toc503907486)

[3.2. Поля, методы и свойства 34](#_Toc503907487)

[3.3. Разработка алгоритмов 39](#_Toc503907488)

[4. ВЕРИФИКАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММЫ 43](#_Toc503907489)

[ВЫВОДЫ 46](#_Toc503907490)

[СПИСОК ЛИТЕРАТУРЫ 47](#_Toc503907491)

[**ПРИЛОЖЕНИЕ А Техническое задание** 48](#_Toc503907497)

[**ПРИЛОЖЕНИЕ Б Руководство клиента** 54](#_Toc503907498)

[**ПРИЛОЖЕНИЕ В Руководство администратора** 56](#_Toc503907499)

# **ВСТУП**

Пояснювальна записка курсового проекту включає в себе:

– сторінок – 60;

– малюнків – 26;

– таблиць – 31;

– додатків – 3;

– джерел – 5.

Метою даної курсової роботи є розробка комплексу програм “Сервіс хмарного зберігання файлів”.

Завданнями курсової роботи є:

1. реалізація алгоритму завантаження та вивантаження файлів будь-якого формату і розміру;
2. реалізація хмарної інфраструктури;
3. захист та шифрування даних користувачів;
4. ПОСТАНУВАННЯ ЗАВДАННЯ
   1. Аналіз предметної області

Згідно з законом Муру, кількість транзисторів на кристалі процесору кожні 2 роки росте, і крива, характеризуюча дане поведінку, має експоненціальний характер[[1]](#_6._WIKIPEDIA_[Электр.).

Комп’ютерні пристрої и комп’ютерне обладнання з кожним днем стає продуктивніше, а також потребує більше об’єму пам’яті, на котрій буде зберігатися гігантська кількість за мірками кінця ХХ і початку ХХІ століття інформації. Ця проблема вирішувалась та досі вирішується різними способами: збільшення об’єму пам’яті на пристрої, використання легко транспортуємих флеш-накопичувачів, поділення даних між двома або декількома схожими пристроями, здатних передавати інформацію кабельним або радіохвильовим шляхом. Однак цю актуальну проблему призвані легко розрішити сервіси хмарного зберігання даних, без будь-яких істотних фінансових затрат для кінцевого користувача.

Хмарне сховище або «Хмара» – це онлайн-сховище, яке розподіляє усі дані користувача на безлічі віддалених один від одного серверів. Про організацію даного сервісу ніхто не може знати, зокрема спеціалістів, приймавших участь у створенні комплексу ПО для реалізації цього проекту, отже це підвищує надійність усієї системи. Хакерські атаки повинні бути спрямовані на безліч об’єктів, зазвичай розташованих в різних регіонах планети, щоби порушити роботу всього комплексу серверів, після чого втрачаються або потрапляють у руки зловмисників усі особисті дані користувачів. Однак навіть це буде непросто здійснити, особисто, якщо у системі є потужні відповідні механізми захисту та високонадійні, відмовостійкі компоненти.  
 Крім зберігання інформації, дана інфраструктура дозволяє проводити найрізноманітніші операції з даними, маніпуляція якими найчастіше необхідна нам у тих випадках, коли нема можливості завантажити усе необхідне на пристрій, який використовується в даний момент, або яке не має інструментарію, який надає сервіс хмарного зберігання даних: редагування, конвертування, архівування, шифрування і т.д. Тому «хмара» концептуально підтримується усіма видами пристроїв, маючих доступ до всесвітньої мережі.

На сьогодні існує безліч сервісів хмарного зберігання даних. Усі вони мають як переваги, так і недоліки, котрі далі будуть розглядані.

**GoogleDrive**

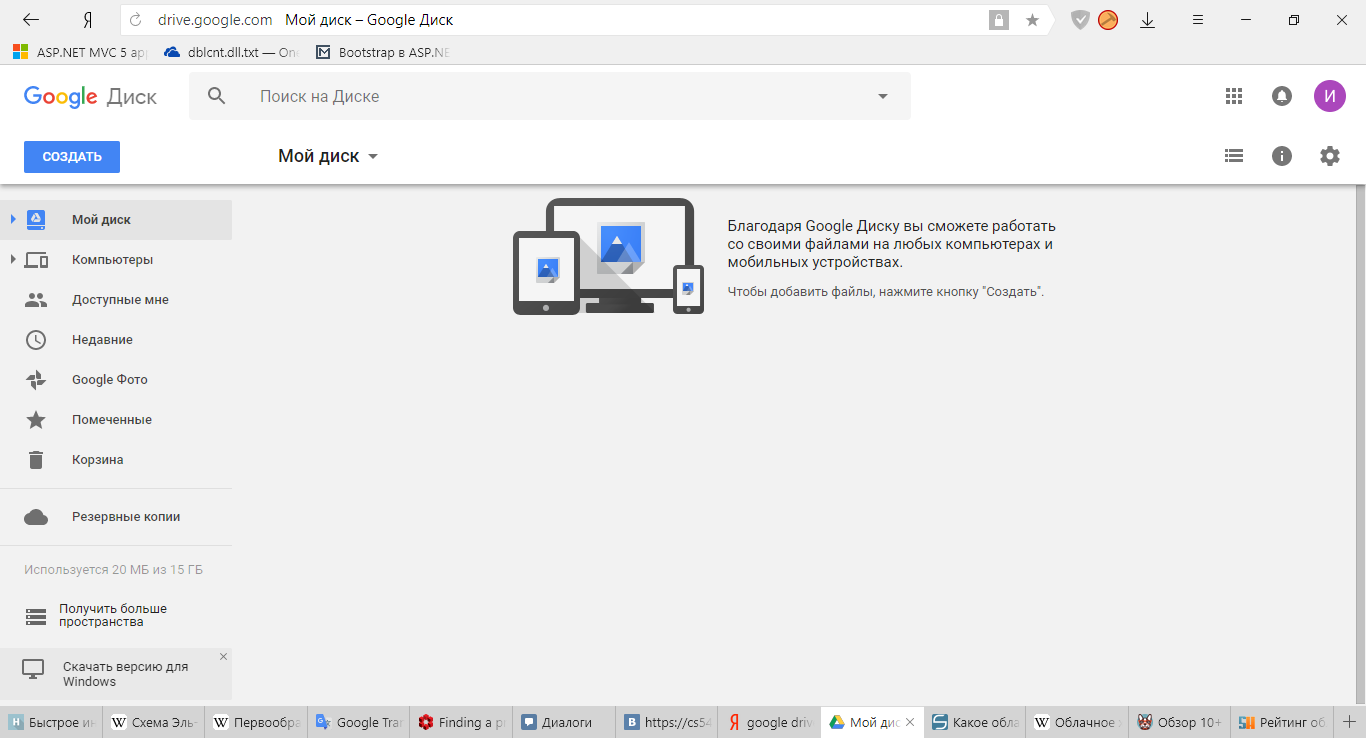
Один з найпопулярніших сервісів хмарного сховища, перевагою якого є багатофункціональний web-інтерфейс, а також в якому доступні такі операції:

* редагування та створення документів в режимі спільного доступу;
* перегляд мультимедійного контенту;
* розширення майже всіх існуючих надбудов;
* пошук документів;
* управління правами доступу.

На тлі всіх інших подібних сервісів можна виділити наступні переваги і недоліки:

Таблиця 1.1 – Переваги та недоліки сервісу GoogleDrive

|  |  |
| --- | --- |
| **+** | **–** |
| Необмежений простір під зберігання зображення і фотографій. | Виділений вільний простір буде використовуватися іншими суміжними сервісами. |
| Гнучкий web-інтерфейс, що дозволяє налаштовувати робоче середовище під потреби користувача. | Windows Desktop версія програми не має будь-яких переваг. |
| При реєстрації відразу надається 15 ГБ безкоштовного дискового простору. |  |



Малюнок 1.1 – Web-інтерфейс сервісу GoogleDrive

**Microsoft OneDrive**

Надійне онлайн-сховище від компанії Microsoft, яке за замовчуванням інтегроване в ОС Windows 10, що дає можливість працювати зі своїми даними на локальному комп'ютері з подальшою автосінхронізацією з «хмарою». Відмінна особливість даного сервісу - інтеграція з пакетом Office 365, що надає ряд переваг при роботі з документами різного типу: Word, Excel, PowerPoint, Visio, Access, Publisher, OneNote і т.д.

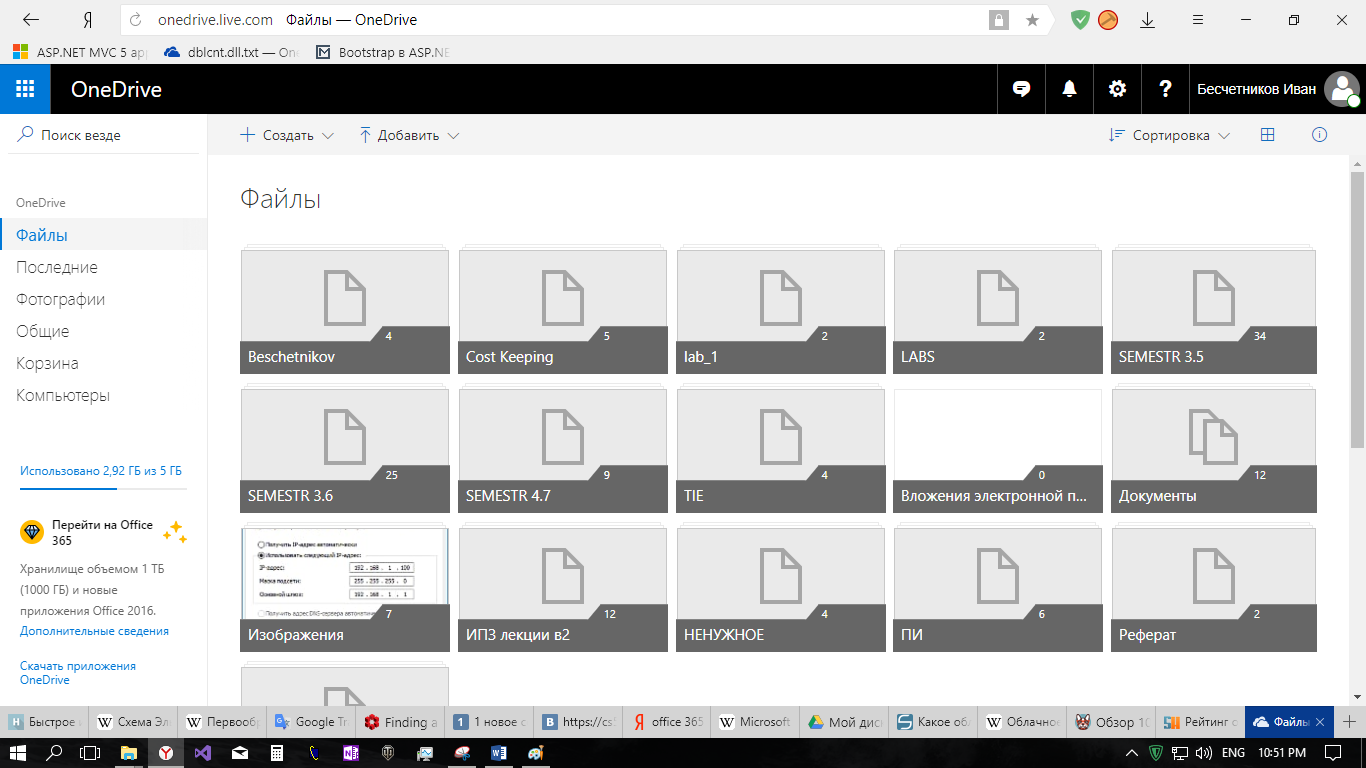
Основні характеристики сервісу:

* безкоштовне місце: 5 ГБ;
* максимальний розмір одного файлу: 10 ГБ;
* максимальний обсяг простору: 5 ТБ;
* сумісність з мобільними ОС: підтримується на Android вище 4.0, iOS вище 9.0, Windows Phone 7/8, на Symbian Belle і на MeeGo 1.2.

Переваги та недоліки сервісу:

Таблиця 1.2 – Переваги та недоліки сервісу OneDrive

|  |  |
| --- | --- |
| **+** | **–** |
| Інтеграція з пакетом Office 365. | Відсутність додаткових функцій. |
| Висока швидкість та стабільна робота. | Безкоштовний тариф надає усього 5 ГБ вільного дискового простору. |
| Знижки при покупці дискового простору з Office 365. |  |



Малюнок 1.2 – Web-інтерфейс сервису OneDrive

**Dropbox**

Є одним з найперших сервісів «хмарної» архітектури, який за весь свій довгий час існування і розвитку встиг доробити кожен вузол інфраструктури. Dropbox, як і більшість інших популярних сервісів, надає досить функціональний і зручний web-інтерфейс. Також він працює по моделі Freemium, що дозволяє при створенні аккаунта вибрати обсяг вільного дискового простору з подальшим збільшенням, але тільки при внесенні оплати.

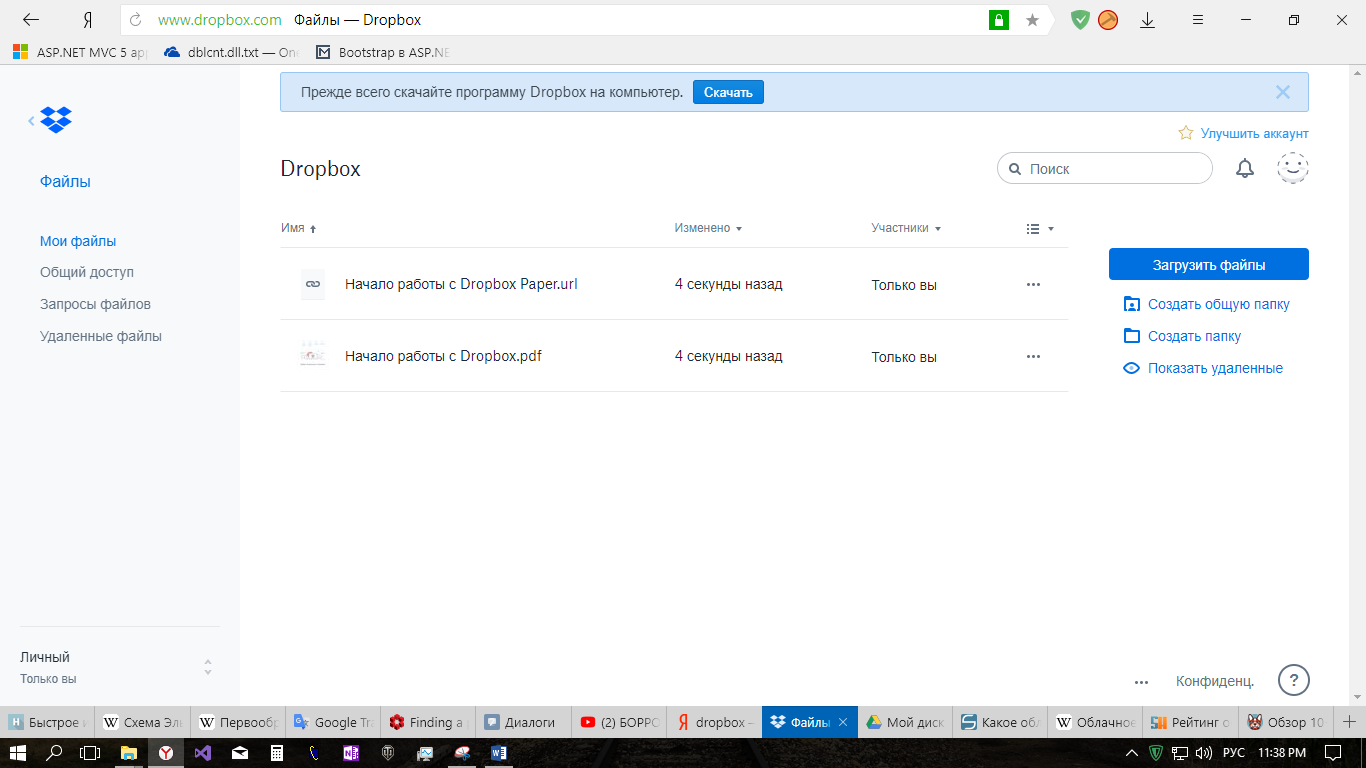
Основні характеристики сервісу

* безкоштовне місце: 2 ГБ;
* максимальний розмір одного файлу через web-інтерфейс - 20 ГБ, через Windows Desktop - без обмежень;
* максимальний обсяг простору: 1 ТБ;
* сумісність з мобільними ОС: підтримується на Windows, macOS, Linux; на мобільних ОС Android, iOS, Windows Phone і BlackBerry

Переваги та недоліки сервісу:

Таблиця 1.3 – Переваги та недоліки сервісу Dropbox

|  |  |
| --- | --- |
| **+** | **–** |
| Гнучкий та функціональний web-інтерфейс. | Максимальна кількість допустимих функцій поширюється тільки для бізнес-клієнтів. |
| Зручність у використанні на клієнтських версіях. | Безкоштовний тариф надає всього 2 ГБ вільного дискового простору. |
| 265-бітне шифрування AES та шифрування SSL. |  |
| Найкраща серед аналогів технологія синхронізації. |  |
| Інтеграція з Microsoft Office 365. |  |
| Необмежене відновлення файлів та журналу версій. |  |
| Посилання доступу з паролем та терміном дії. |  |
| Можливість налаштовувати і управляти рівнями доступу. |  |



Малюнок 1.3 – Web-інтерфейс сервису Dropbox

**Mega**

Перспективний хмарний файлообмінник, що надає максимально зручний набір інструментів для користувачів, які не люблять викладати гроші за користування невеликого обсягу пам'яті під зберігання особистих даних. Такий сервіс також надає користувачам можливість ділитися своїми особистими файлами за схемою «friend-to-friend», використовуючи для цих цілей надійний криптографічний алгоритм шифрування AES, причому відкритий ключ відомий тільки тому користувачеві, якому стане доступна посилання на файл обміну.

«Mega» позиціонується як сервіс, що захищає особисті дані користувачів, тому використовується наскрізне шифрування.

Даний web-сервіс існує в трьох варіантах для використання: web-додаток, Windows Desktop додаток і мобільний додаток.

Основні характеристики сервісу:

* безкоштовне місце: 50 ГБ;
* максимальний розмір одного файлу через web-інтерфейс - 10 ГБ, через десктопних програм - без обмежень;
* максимальний обсяг простору: 4 ТБ;
* сумісність з мобільними ОС: підтримується на Windows, macOS, Linux; на мобільних ОС Android, iOS, Windows Phone і BlackBerry.

Переваги та недоліки сервісу:

Таблиця 1.4 – Переваги та недоліки сервісу Dropbox

|  |  |
| --- | --- |
| **+** | **–** |
| Плагіни для браузерів. | Не висока швидкість доступу. |
| Інтегрований чат. | Існують ліміти на трафік. |
| 2048-бітне шифрування AES і шифрування SSL. |  |
| Надається великий обсяг безкоштовного сховища – 50 ГБ. |  |

Порівняльна оцінка деяких параметрів, що існують популярних хмарних сервісів:

Таблиця 1.5 – Зведена таблиця для популярних хмарних сервісів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерії оцінки** | **Google Drive** | **Microsoft OneDrive** | **Mega** | **Dropbox** |
| Максимально допустимий обсяг виділеної пам'яті (ТБ) | 30 | 5 | 4 | 1 |
| Максимальний розмір одного файлу (ГБ) | 5000 | 10 | 10 | 20 |
| Шифрування даних | – | – | + | + |
| Інтеграція з офісним пакетом документів від Microsoft | + | + | – | – |
| Сумісність з мобільними ОС | + | + | + | + |
| Синхронізація даних на ПК | + | + | – | + |
| Редагування та створення документів в режимі спільного доступу | + | + | – | + |
| Перегляд мультимедійного контенту | + | – | – | – |
| Велика розмаїтість додаткових функцій | + | – | + | – |

1. ПРОЕКТУВАННЯ
   1. Вибір архітектури

В ході проектування інфраструктури сервісу була обрана трирівнева архітектура:

* рівень web-додатку;
* рівень сервісу;
* рівень сховища даних.



Малюнок 2.1 – Трирівнева архітектура проекту

* 1. Рівень web-додатку

Рівень web-додатки являє собою додаток, який є головним у всьому проекті і служить для контролю всіх призначених для користувача дій: реєстрація, авторизація, автентифікація, валідація. У разі неуспіху виконання даних процедур, буде викинута системою відповідна помилка, яку користувач буде зобов'язаний виправити, слідуючи отриманої інструкції від браузера або web-додатка.

Для реалізації цього вузла системи був використаний ASP.NET MVC 5 фреймворк для створення web-додатків.[[2]](#_5._PROFESSORWEB_[Электр.)

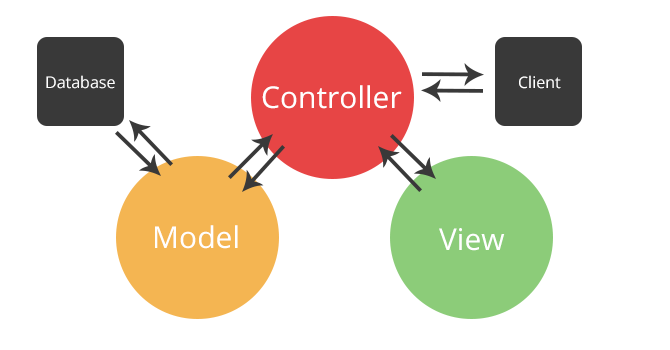
ASP.NET MVC 5 - це високонадійна інфраструктура web-додатка, яка покликана забезпечити створення легковагій надійної архітектури проекту, досягається шляхом заздалегідь реалізованих повноцінних відмовостійких і протестованих модулів. Вона також підтримує розширюваність, тестування проектів, тому супровід стає легшим і простішим.

Ця платформа має в наявності відмінний інструментарій, що дозволяє створювати прості, а разом з ним і гнучкі модулі. Вбудовані допоміжні методи HTML генерують ясний та чіткий код розмітки, який відповідає всім сучасним стандартам опису графічних web-інтерфейсів. Високопродуктивна маршрутизація дає можливість створювати зручні, читабельні URL-адреси, завдяки чому програміст вільний придумувати будь-які осмислені імена сторінок, або ж імена, які приховують деяку логіку роботу усього web-додатка.

Проекти, написані на ASP.NET MVC 5 підтримують розширюваність, а це значить, що кожен окремий модуль або модуль, вбудований в систему можна перевірити в модульному тестуванні, а інтеграційні тести будуть відслідковувати стан всієї системи в фоновому режимі роботи.

Дана платформа для розробки web-додатків використовує архітектурний шаблон MVC. До переваг цього шаблону можна віднести наступне:

* користувач взаємодіє з додатком відповідно до природного циклу: користувач робить деяку дію, у відповідь на яку модель змінює свій стан і рендерить певне уявлення, що добре вписується в логіку роботи запитів і відповідей протоколу HTTP.
* Ряд web-додатків потребують поділу модулів всього проекту: HTML-сторінки, компоненти щодо поліпшення web-інтерфейсу, бази даних, бізнес-логіка, сервіси і т.д. Все це відмінно вписується в концепцію MVC.[[3]](#_6._WIKIPEDIA_[Электр.)



Малюнок 2.2 – Архітектура MVC-проектів

Інфраструктура ASP.NET MVC Framework реалізує цей шаблон, причому з доробленими і поліпшеними деякими сценаріями використання даного патерну.

* + 1. Розширюваність

Інфраструктура MVC Framework побудована таким чином, компоненти якої є взаємно незалежними і представляються у вигляді абстрактних базових класів. Тому існують такі схеми використання вбудованих компонентів:

* повне використання вбудованих компонентів;
* часткове використання вбудованих компонентів з деяким коректуванням, реалізуючи власний клас з успадкуванням від абстрактного класу або інтерфейсу всіх його методів і властивостей;
* повна заміна існуючого компонента власним.



Малюнок 2.3 – Сценарії використання вбудованих компонентів ASP.NET MVC Framework

* + 1. HTTP и HTML

Інфраструктура ASP.NET MVC Framework генерує ясну і просту розмітку, при використанні HTML-хелперів. Однак крім простої генерації HTML-розмітки, вона здатна на побудову елегантних, легко оброблюваних блоків коду-розмітки, які будуть в значно більшій ступені краще рендерити, аніж незліченна безліч складних блоків, якими важко керувати. Генерований код до того ж буде оформлений красивими і «доброзичливими» стилями CSS і плавною анімацією JavaScript.

З іншого боку, якщо виникає потреба у створенні більш складних елементів управління, таких як DatePicker, Clock, CascadeMenu і т.д., то можна скористатися вбудованими бібліотеками, які значно прискорюють і спрощують процес розробки цих елементів. Наприклад, бібліотеки jQuery і BootStrap CSS вже в останніх версіях середовищ розробок, таких як Visual Studio і Code Studio, є, і вважаються вбудованими компонентами, ефективність яких оскаржувати ніхто не стане.

Так само, на відміну від сторінок WEB.FORMS сторінки ASP.NET MVC Framework значно менше за розміром, навіть незважаючи на те, що швидкість з'єднання з web-сайтами зросла в рази, проте це підвищує комфорт використання для кінцевого користувача і прискорює процес запуску самого web-додатки. Це досягається за рахунок відсутності View State-даних сторінки, з якими в основному працює платформа WEB.FORMS.

ASP.NET MVC тісно пов'язана з HTTP-протоколом.[[4]](#_1._Программирование_на) Це дозволяє контролювати запити і відповіді, що виникають між сервером та клієнтом. Завдяки цьому, можна точно коригувати кожен момент роботи двох сторін. А при використанні AJAX-запитом не будуть потрібні ніякі автоматичні відправки зворотного виклику, тому що вона проста і не використовує стан клієнтської сторони.



Малюнок 2.4 – Бібліотеки, вбудовані за замовчуванням в ASP.NET MVC 5

**Здатність до тестування**

Поділ відповідальності між усіма частинами web-додатки – те, що дозволяє даний фреймворк, роблячи супровід і тестування окремих частин всієї програми набагато простіше і зручніше. Однак на цьому можливості ASP.NET MVC Framework платформи не обмежені. Для кожного модуля компонентно-орієнтованого проекту всієї системи, вона забезпечує структурованість, яка так необхідна при використанні модульного тестування.

Крім модульного тестування, існують також засоби, покликані зімітувати як дії користувача, так і взаємодію з ним в графічному інтерфейсі додатка, обробка яких автоматично здійснюється всією інфраструктурою, і немає більше необхідності стежити за змінами структури HTML-розмітки, або ж відстежувати генеруються HTML-елементи і ідентифікатори CSS.



Малюнок 2.5 – Інтегровані інструменти модульного тестування

* 1. Рівень сервісу

Даний рівень є інтерфейсом з обміну даними між web-додатком і серверними додатками. Його робота здійснюється незалежно від роботи web-додатка, проте рівень web-додатка може отримувати деяку інформацію про процес транзакції по вилученню або завантаженню файлів, яку сервіс «витягне» або обробить.

На цьому рівні також оперуються один або група файлів, які слід зберегти або витягти з пам'яті пристрою. Під час вилучення дані передаються рівню, з яким він безпосередньо пов'язаний - рівень сервісу. У разі виникнення помилок, додаток цього рівня має повідомити додатком сервісу про це, а також її назву. Тоді рівень сервісу в обов'язковому порядку зробить спробу щодо усунення помилки, якщо, звичайно, вона не потребує безпосереднього втручання спеціалістів. Прикладами таких помилок можуть виступати:

* втрата пакетів даних при передачі або отриманні даних;
* тимчасова недоступність сервера-сховища;
* відсутність вільної пам'яті;

Додатковими обов’язками цього рівня – шифрування та дешифрування бажаних користувачем даних, що забезпечує зменшення навантаження на систему загалом та її збільшення продуктивності.

* 1. Рівень сховища даних

Даний рівень представляє з себе безліч ftp-серверів, встановлених на пристроях, здатних зберігати, видаляти, отримувати та передавати файл або файлі. Це може бути будь-який ПК, будь-яка робоча станція або гаджет, який працює під управлінням операційної системи Android з 17 рівнем API.

На додаток до програми, що забезпечує обмін даними по ftp-протоколу, на пристрій встановлюється додаток, що відстежує зміну файлової системи, тому що ftp-сервер не має додаткових інструментів, здатних відповідати на запит від сервісу про кількість вільної пам'яті, в іншому випадку система викине відповідну помилку, і програма екстрено завершить свою роботу

* 1. UML – діаграма використання

Функції, які реалізовані в проекті, можна уявити за допомогою діаграми використання.



Малюнок 2.7 – UML-діаграма використання. Взаємодія користувача з інтерфейсом програми і її функціями.

Дійова особа "Гість": даний актор являє собою користувача, який не авторизований або не має зареєстрованого облікового запису в цьому сервісі.

Дійова особа "Авторизований користувач»: даний актор являє собою користувача, який увійшов в систему, пройшовши етап авторизації або реєстрації.

Дійова особа "Адміністратор»: даний актор являє собою користувача, уповноваженого адмініструванням облікових записів інших користувачів

Таблиця 2.1 – Прецедент «Зареєструватися»

|  |  |
| --- | --- |
| Назва прецеденту | «Зареєструватися» |
| Опис | Даний прецедент дозволяє реєструватися на даний сервіс. |

Таблиця 2.2 – Прецедент «Авторизуватись»

|  |  |
| --- | --- |
| Назва прецеденту | «Авторизуватись» |
| Опис | Даний прецедент дозволяє авторизуватись. |

Таблиця 2.3 – Прецедент «Відновити пароль»

|  |  |
| --- | --- |
| Назва прецеденту | «Відновити пароль» |
| Опис | Даний прецедент дозволяє відновити доступ до аккаунту, в разі якщо користувач забув пароль. |

Таблиця 2.4 – Прецедент «Видалити акаунти користувачів»

|  |  |
| --- | --- |
| Назва прецеденту | «Видалити акаунти користувачів» |
| Опис | Даний прецедент дозволяє на свій розсуд видаляти адміністратору акаунти користувачів. |

Таблиця 2.5 – Прецедент «Змінити облікові дані користувача»

|  |  |
| --- | --- |
| Назва прецеденту | «Змінити облікові дані користувача» |
| Опис | Даний прецедент дозволяє адміністратору змінювати облікові дані користувача, такі як логін, пароль, номер мобільного телефону та інше, якщо користувачеві не вдається увійти в свій аккаунт. |

Таблиця 2.6 – Прецедент «Видалити акаунт»

|  |  |
| --- | --- |
| Назва прецеденту | «Видалити акаунт» |
| Опис | Даний прецедент дозволяють користувачеві при необхідності видалити власний аккаунт. |

Таблиця 2.7 – Прецедент «Вийти»

|  |  |
| --- | --- |
| Назва прецеденту | «Вийти» |
| Опис | Даний прецедент дозволяє вийти з облікового запису. |

Таблиця 2.8 – Прецедент «Видалити файл або папку»

|  |  |
| --- | --- |
| Назва прецеденту | «Видалити файл або папку» |
| Опис | Даний прецедент дозволяє користувачеві видаляти існуючу папку або файл. |

Таблиця 2.9 – Прецедент «Створити файл або папку»

|  |  |
| --- | --- |
| Назва прецеденту | «Створити файл або папку» |
| Опис | Даний прецедент дозволяє користувачеві створити папку або файл. |

Таблиця 2.10 – Прецедент «Змінити облікові дані»

|  |  |
| --- | --- |
| Назва прецеденту | «Змінити облікові дані» |
| Опис | Даний прецедент дозволяє користувачеві змінювати деякі облікові дані: логін, пароль, ПІБ, аватар, номер мобільного телефону і т.д. |

Таблиця 2.11 – Прецедент «Переглядати вміст файлів форматів docx, xlsx, pptx, txt»

|  |  |
| --- | --- |
| Назва прецеденту | «Переглядати вміст файлів форматів docx, xlsx, pptx, txt» |
| Опис | Даний прецедент дозволяє користувачеві переглядати вміст файлів з такими розширеннями. |

Таблиця 2.12 – Прецедент «Редагувати файли форматів docx, xlsx, pptx, txt»

|  |  |
| --- | --- |
| Назва прецеденту | «Редагувати файли форматів docx, xlsx, pptx, txt» |
| Опис | Даний прецедент дозволяє редагувати файли з такими розширеннями. |

Таблиця 2.13 – Прецедент Ззберегти зміни»

|  |  |
| --- | --- |
| Назва прецеденту | «Зберегти зміни» |
| Опис | Даний прецедент дозволяє користувачеві зберегти всі внесені зміни у файлі при його редагуванні. |

Таблиця 2.14 – Прецедент «Завантажити файл/файли»

|  |  |
| --- | --- |
| Назва прецеденту | «Завантажити файл/файли» |
| Опис | Даний прецедент дозволяє користувачеві завантажувати файли або групу файл на сервіс з пристрою, що використовується в даний момент часу. |

Таблиця 2.15 – Прецедент «Завантажити файл/файли з сервера»

|  |  |
| --- | --- |
| Назва прецеденту | «Завантажити файл/файли з сервера» |
| Опис | Даний прецедент дозволяє скачувати файл або групу файлів з сервісу на пристрій, який використовується в даний момент. |

* 1. Проектування графічного інтерфейсу: проектування сторінок Web-додатка

Пользовательский интерфейс был реализован с использованием HTML, CSS и движка представлений Razor. Большая часть CSS-стилей, которые повсеместно использовались во всех представлениях в проекте, – относятся к CSS-фреймворку Bootstrap 4.

Данная библиотека стилей имеет в наличии большое количество HTML- CSS-шаблонов, благодаря которым процесс создания web-интерфейса проходит проще и быстрее. Также она предоставляет ряд готовых, протестированных скриптов, которые делают сайт интуитивно понятным и удобным.

\_ViewStart – представление, визуализирующееся при запуске приложения. Оно не содержит визуальных элементов, а устанавливает представление шаблона (\_Layout). Представление \_Layout определяет шаблон страницы авторизации, регистрации, управления учетной записью, рабочее пространство с индивидуальными файлами и папками пользователя и т.д.

Всего в приложении имеется порядка 30 представлений, 3 из которых ялвяются шаблонами: \_LayoutEntrance, \_LayoutMaster и \_LayoutAdmin.

\_LayoutEntrance – шаблон, определяющий основной вид HTML-страниц при регистрации и авторизации.

\_LayoutMaster – шаблон, определяющий основй вид для большей части HTML-страниц, отображающих и позволяющих настраивать учетную запись пользователя.

\_LayoutAdmin – шаблон, определяющий основй вид для HTML-страниц, которые только администратор будет видеть; позволяющие администрировать учетные записи всех пользователей.

Страница авторизации предлагает ввести существующие данные учетной записи пользователя в соответствующие поля для ввода: Email и Password; или же воспользоваться авторизацией через одну из раннее созданных учетных записей в популярном сервисе, таком как Google или Facebook.

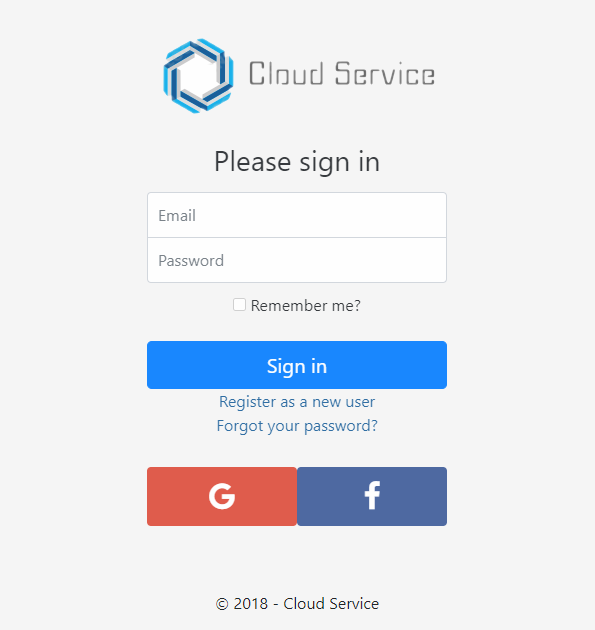


Рисунок 2.8 – Страница авторизации.

Если же пользователь является Гостем в системе, то он может перейти на страницу регистрации по соответствующей ссылке ниже, а затем зарегистрироваться, предварительно заполнив все необходимое поля.

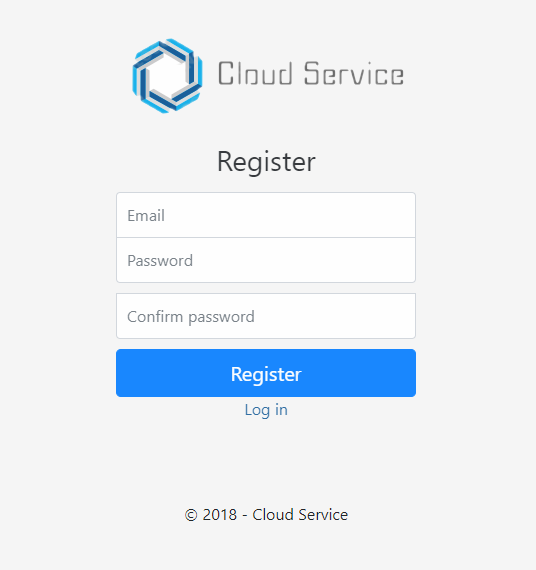


Рисунок 2.9 – Страница регистрации.

Представление \_LayoutMaster определяет шаблон всех страниц, связанных с работой авторизованного пользователя. В верхней части второго шаблона находится логотип данного сервиса, который является ссылкой на домашнюю страницу приложения. Правее от логотипа находятся 2 кнопки, которые запускают процесс загрузки файла/файлов на сервер и скачивают оттуда соответственно. В случае, если пользователю надо произвести настройку учетной записи, ему следует перейти на страницу настроек, нажав на ссылку, представляющую из себя установленное пользовательское имя, а правее от нее находится кнопка выхода из системы.



Рисунок 2.10 – Панель управления для авторизованного пользователя.

Используя эту навигационную панель, можно загружать и выгружать файлы или папки в облачном хранилище. Для того, чтобы загрузить файл или группу файлов, а также папку или группу папок в хранилище сервера, предназначена кнопка «Upload» в виде «облака» со стрелкой, направленной вверх, при нажатии которой начинается процесс загрузки предварительно выбранных файлов, из установленной пользователем директории на устройстве, на сервер.



Рисунок 2.11 – Кнопка загрузки файлов на сервер.

Для того, чтобы скачать файлы с сервера, предназначена кнопка «Download», которая расположена левее, чем кнопка «Upload». Она сработает только в том случае, если пользователь выбрал необходимые файлы, отметив их флажками.

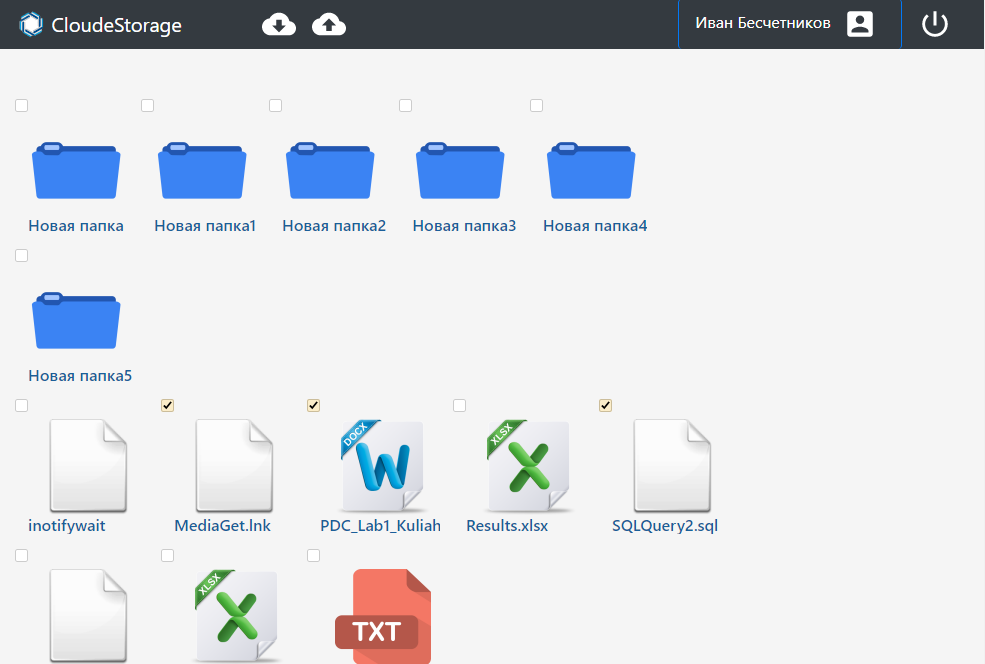


Рисунок 2.12 – Страница со всеми личными файлами и папками.

Затем сервис автоматически их соберет в архив, который и будет скачан в конечном итоге пользователем на ПК или гаджет.

Т.к. в проекте не менее 30 как частичных, так и простых страниц, то далее будут рассмотрены основные, которые демонстрируют выполнение основного функционала приложением.

**DisplayFiles (FileController)**

Частичное представление, в котором все файлы и папки отображаются в отсортированном виде, поэтому пользователю будет легче ориентироваться в данном информационном пространстве.

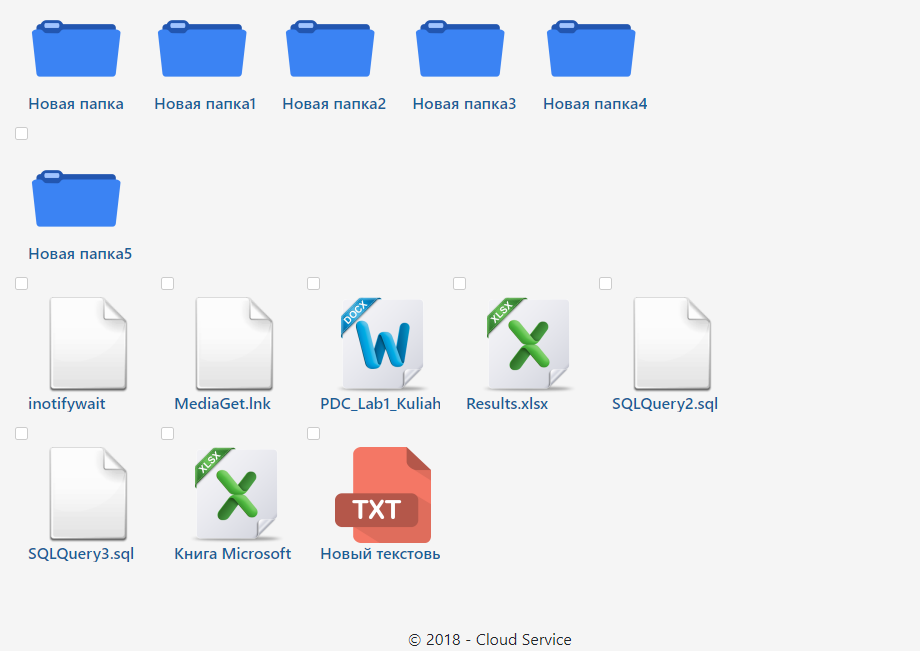


Рисунок 2.13 – Результат работы метода DisplayFiles для контроллера FileController.

Все папки всегда располагаются в самой верхней части этой среды, а все файлы – ниже.

**Index (ManageController)**

Это представление позволяет авторизованному пользователю настраивать свою учетную запись:

* изменить логин;
* изменить пароль;
* добавить номер мобильного телефона;
* включить или выключить двухфакторную авторизацию.

Чтобы перейти на эту страницу, следует нажать на ссылку в правом верхнем углу собственного логина.

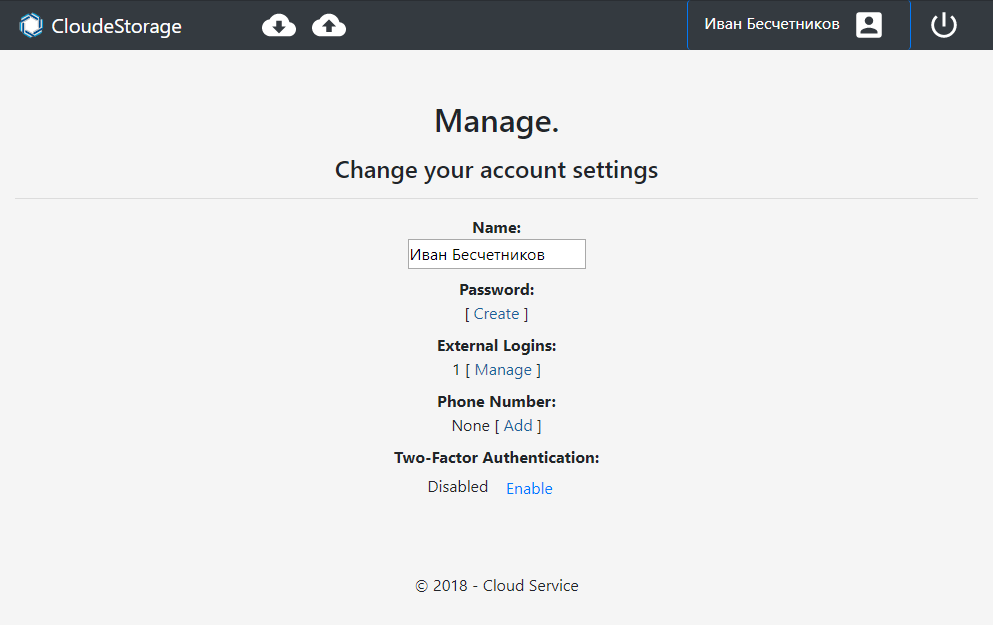


Рисунок 2.14 – Результат работы метода Index для контроллера ManageController.

Разрабатываемый графический интерфейс для устройств-серверов представляет из себя единственное окно приложения минималистичного вида, которое состоит всего из трех частей: кнопка активации и деактивации соединения с сервисом по обмену данными, стек статуса работы всего приложения и стек ошибок.

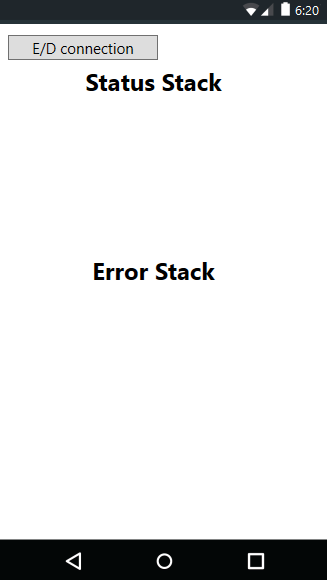


Рисунок 2.15 – Интерфейс программы для серверного устройства.

Кнопка активации и деактивации соединения служит для того, чтобы уведомить сервис о том, что устройство готово к обмену данными, как, например, при запуске приложения; или же для того, чтобы прервать соединение, в случае некоторых технических неполадок.

Стек статуса выводит краткую информацию о времени работы всего приложения, количество использованного трафика в процессе обмена данными, количество полученных файлов и количество переданных сервису файлов.

Стек ошибок выводит информацию об ошибках, возникающих в процессе работы приложения, поэтому в случае обнаружения оператором конфликтов между web-приложением и устройствами-серверами, он может получить необходимую информацию об ошибке здесь. Следует заметить, что ошибки (кроме технических) не прерывают работу всего приложения.

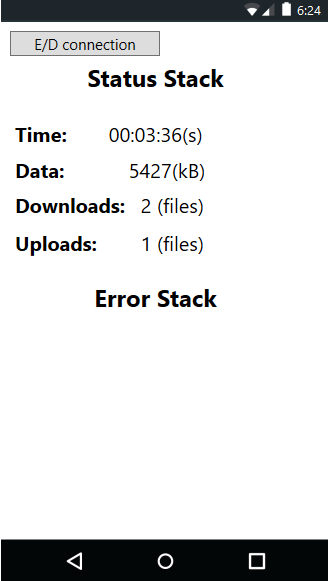
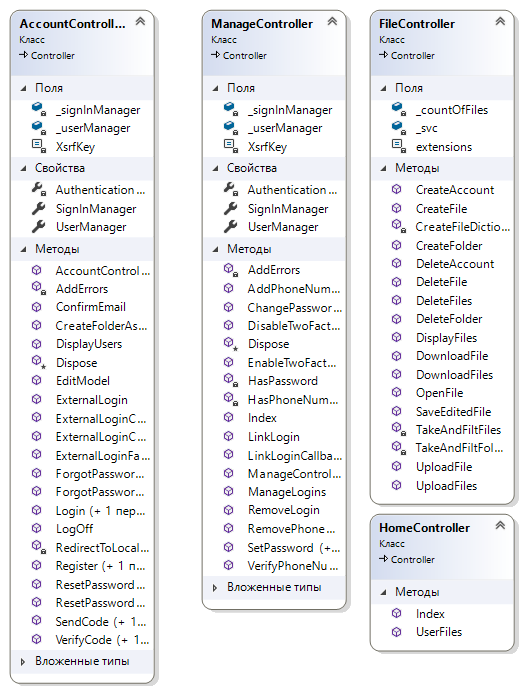


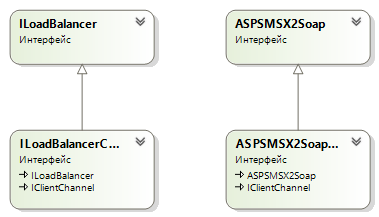
Рисунок 2.16 – Интерфейс программы для устройства-сервера в режиме работы.

1. РОЗРОБКА
   1. Розробка діаграми класів

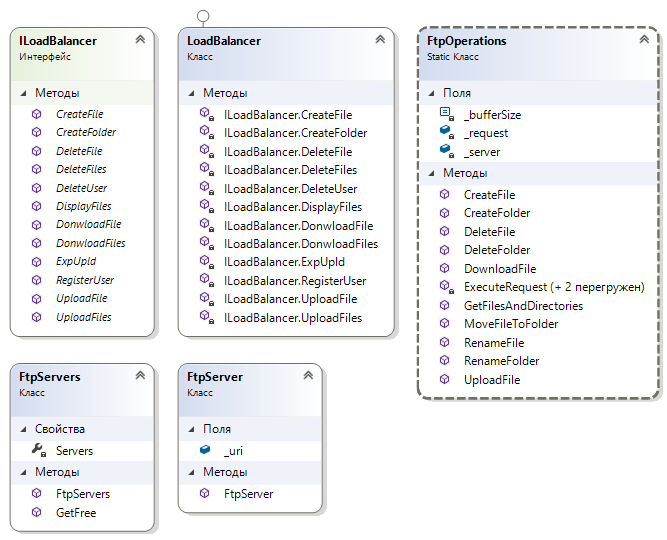
При аналізі поставленого завдання були виділені три основні сутності, які представлені трьома проектами, а також в яких були створені класи і інтерфейси для їх взаємодії: UI (HomeController, AccountController, ManageController, FileController, MainPage), BusinessLogic (FtpOperations, LoadBalancer, FileNFolders,), DAL ( ApplicationDbContext, ApplicationUser, ApplicationUserManager, FileToSave, FileToDelte, FileToDisplay, FileToDownload, FileToUpload, Registration, DeleteAccount).



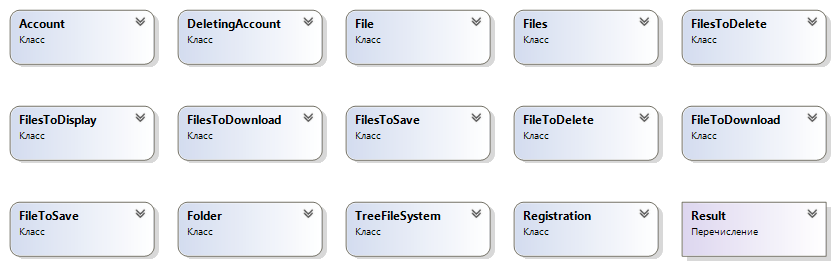
Малюнок 3.1 – Діаграма класів-контролерів web-додатка.



Малюнок 3.2 – Діаграма інтерфейсів служб: служба взаємодії з серверами і служба двухфакторной авторизації.



Малюнок 3.3 – Діаграма класів севріса. Класи взаімодейтсвіе web-додатка і серверів.



Малюнок 3.4 – Діаграма класів севріса. Класи, що містять методи, націлені на виконання конкретного завдання або завдань над одним єдиним об'єктом, і які відповідають SOLID-принципам.

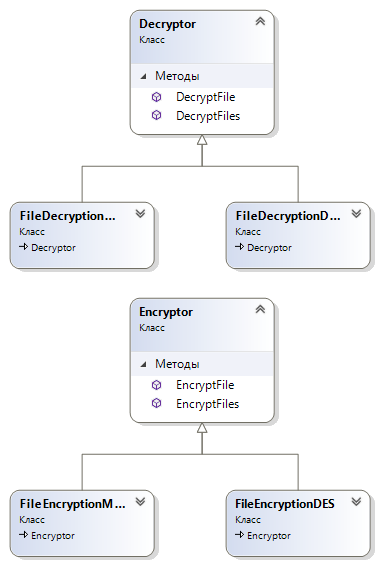


Рисунок 3.5 – Діаграма класів сервісу. Клас шифрування і дешифрування даних.

Клас HomeController - контролер, який відповідає за відображення «домашньої» сторінки, тобто сторінки з одними файлами і папками.

Клас AccountConrtoller - контролер, який відповідає за створення і управління профілем користувача: реєстрація, авторизація, видалення і створення акаунтів.

Клас ManageConrtoller - контролер, який відповідає за зміну всіх налаштувань облікового запису, а також покликаний взаємодіяти з базами даних.

Клас FileController - контролер, який відповідає за завантаження, скачування, видалення, створення, редагування, зберегти і сортування по типу файлів, папок на стороні сервера.

Клас LoadBalancer - клас, завдяки якому web-додаток взаємодіє з сервісом.

Клас FtpOperations - клас, що містить усі методи, що виробляють відповідні операції на стороні сервера/серверів.

Класи FtpServer і FtpServers - об'єкти прив'язки сервера і серверів відповідно.

Класи FileToSave, FileToDelte, FileToDisplay, FileToDownload, FileToUpload, Registration, DeleteAccount і т.д. - класи, що містять методи, які робляють елементарні операції на стороні сервера.

Класи Encryptor і Decryptor - об'єкти для шифрування і дешифрування даних.

* 1. Поля, методы и свойства

**Методы и поля, использованные в проекте Cloud Service, WebService и Mobile Server**

В таблицах приведено описание(назначение) методов и полей следующих классов:

Таблица 3.1 – Методы HomeController

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **Методы HomeController** | |
| public ActionResult Index | Запрос GET на рендеринг главной web-страницы и возврат ее в виде ответа пользователю |

Таблица 3.2 – Поля, методы и свойства AccountController

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **Поля AccountController** | |
| private ApplicationSignInManager \_signInManager | Объект класса авторизованных пользователей |
| private ApplicationUserManager \_userManager | Объекты класса зарегистрированных пользователей |
| **Методы AccountController** | |
| public AccountController(ApplicationUserManager userManager, ApplicationSignInManager signInManager ) | Конструктор, инициализирующий объекты классов ApplicationUserManager и ApplicationSignInManager |
| public ActionResult Login(string returnUrl) | Метод, вызывающийся методом GET. Служит для вывода частичного представления логировния |
| public async Task<ActionResult> Login(LoginViewModel model, string returnUrl) | Асинхронный метод логирования, вызывающийся запросом POST. Служит для получения модели данных LoginViewModel |
| public ActionResult Register() | Метод, вызывающийся запросом GET. Служит для вывода частичного представления регистрации |
| public async Task<ActionResult> Register(RegisterViewModel model) | Асинхронный метод регистрации, вызывающийся методом POST. Служит для получения модели данных RegisterViewModel |
| public Task CreateFolderAsync(string name) | Асинхронный метод для создания личной папки пользователя на сервере, вызывающийся после успешного выполнения метода Register |
| **Свойства AccountController** | |
| public ApplicationSignInManager SignInManager | Свойство, получающее или возвращающее объект класса ApplicationSignInManager |
| public ApplicationUserManager UserManager | Свойство, получающее или возвращающее объект класса ApplicationUserManager |

Таблица 3.3 – Методы FileController

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **Методы FileController** | |
| public ViewResult DisplayFiles(string userName) | Метод, выводящий частичное представление отображения файлов и папок |
| public ActionResult UploadFiles(HttpPostedFileBase[] files) | Метод, вызывающийся запросом POST, получающий загружаемые пользователем файлы и папки на сервер |
| private List<FileInfo[]> TakeAndFiltFiles(List<FileInfo> files, int itemPerRow | Метод, сортирующий файлы при выводе частичного представления |
| private List<DirectoryInfo[]> TakeFolders(List<DirectoryInfo> files, int itemPerRow) | Метод, сортирующий папки при выводе частичного представления |

Таблица 3.4 – свойства FilesNFolders

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **Своства FilesNFolders** | |
| public List<FileInfo[]> Files | Содержит все пользовательские файлы |
| public List<FileInfo[]> UnknwnFiles | Содержит все пользовательские файлы с неизвестным расширением |
| public List<DirectoryInfo[]> Directories | Содержит все пользовательские папки |

Таблица 3.5 – Методы ManageController

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **Методы ManageController** | |
| public async Task<ActionResult> Index(ManageMessageId? message) | Асинхронный метод, вызывающийся запросом GET. Изменяет все пользовательские учетные данные и уведомляет об этом пользователя |
| public async Task<ActionResult> RemoveLogin(string loginProvider, string providerKey) | Асинхронный метод, вызывающийся запросом POST. Удаляет старый логин и устанавливает новый |
| public ActionResult AddPhoneNumber() | Метод, вызывающийся запросом GET. Выводит частичное представление для привязки мобильного номера к аккаунта |
| public async Task<ActionResult> AddPhoneNumber(AddPhoneNumberViewModel model) | Асинхронный метод, вызывающийся запросом POST. Привязывает номер мобильного телефона к аккауниу |

Таблица 3.6 – Поля и методы MobileServerController

|  |  |
| --- | --- |
| **Название** | **Описание** |
| **Методы MobileServerController** | |
| public string Upload(IEnumerable<HttpPostedFileBase> uploads) | Передает загруженные пользователем файлы сервису |
| public string Download(IEnumerable<HttpPostedFileBase> uploads) | Получает пользовательские файлы от сервиса |

* 1. Разработка алгоритмов

В соответствии с разработанной диаграммой классов необходимо спроектировать такие классы и методы:

1. HomeController;
2. AccountController;
3. ManageController;
4. FileController;
5. MobileController;
6. PushFiles;
7. GetFiles;
8. StartListening;
9. И др.

Ниже приведены несколько алгоритмов работы некоторых методов из списка:



Рисунок 3.4 – Схема принципа работы метода PushFiles, производящий передачу файлов службе, до тех пор, пока есть данные и сервис работоспособен.



Рисунок 3.5 – Схема принципа работы метода StartListening на серверной стороне, который «прослушивает» входящие подключения от службы.



Рисунок 3.6 – Схема принципа работы метода Index для контроллера HomeController, проверяющий, авторизован ли пользователь и, в соответствии с полученными результатами, перенаправляет пользователя на определенную страницу.

1. ВЕРИФИКАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММЫ

Верификация проводилась путем проверки реализации требований технического задания:

1. Регистрация, авторизация пользователя.

2. Загрузка файла на сервер.

3. Создание, удаление, переименование файла.

Результаты проделанной работы, которые подтверждают, что программа правильно выполняет задачи приведены ниже:

Таблица 4.1 – Набор тестов

|  |  |  |
| --- | --- | --- |
| Название теста | Результат теста | Скриншот с результатом |
| Регистрация пользователя Гостя | Пользователь успешно зарегистрирован |  |
| Загрузка файла на сервер | Файл успешно загружен |  |
| Переименование папки | Папка успешно переименована |  |
| Переименование файла | Файл успешно переименован |  |
| Удаление папки | Папка успешно удалена |  |
| Создание папки | Папка успешно создана |  |

Таблица 4.2 – Набор Unit-тестов

|  |  |  |
| --- | --- | --- |
| Тестируемый метод | Описание метода | Скриншот с результатом |
| CreatingUser StorageFolder\_ CreatedUser StorageFolder | Создание папки под хранение личных файлов для только что зарегистрировавшегося пользователя. |  |
| DownloadigFiles\_DownloadedFiles | Загрузка файлов на сервер. |
| UploadingFiles\_ UploadedFiles | Скачивание файлов из сервера. |
| CreatingFolder\_ CreatedFolder | Создание простой папки. |
| RomovingFile\_ RemovedFile | Удаление файла. |
| IsEqualHashesOfLoginAndPassword\_CheckHashesOfLoginAndPassword | Одностороннее шифрование пользовательских учетных данных (логин и пароль) hash-функцией. |  |
| IsEncryptedDataOfFile\_CheckDataOfFile | Шифрование содержимого файла алгоритмом RSA. |

ВЫВОДЫ

При выполнении данной курсовой работы был реализован проект, позволяющий загружать, выгружать файлы, папки, также организовывать рабочее пространство с данными. Разработанная программа верифицирована и протестирована.

Данное приложение выполняет:

1. регистрацию, авторизацию пользователей;
2. создание, удаление, изменение файлов, папок;
3. загрузку файлов, папок в сервера;
4. выгрузку файлов, папок из сервера;
5. удаление аккаунта;
6. настраивание учетных данных;

Для дальнейшей модернизации проекта можно:

1. улучшить оптимизацию приложения;
2. добавить учетную запись для администратора;
3. добавить функцию «поделиться» для файлов и папок;
4. добавить чат для двух или группы пользователей;
5. добавить шифрование файлов;

СПИСОК ЛИТЕРАТУРЫ

1. WIKIPEDIA [Электр. ресурс] – Режим доступа:  
   <https://ru.wikipedia.org/wiki/Закон_Мура>

# ASP.NET MVC FRAMEWORK – WIKIPEDIA [Электр. ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/ASP.NET_MVC_Framework>

# ASP.NET MVC 5 – METANIT.COM [Электр. ресурс] – Режим доступа: <https://metanit.com/sharp/mvc5/>

# XAMARIN [Электр. ресурс] – Режим доступа: <https://www.xamarin.com/platform>

# PROFESSORWEB [Электр. ресурс] – Режим доступа: <https://professorweb.ru/>

# WIKIPEDIA [Электр. ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Controller>

# **ПРИЛОЖЕНИЕ А Техническое задание**

1. **Введение** 
   1. **Наименование программы**

Разрабатываемый комплекс программ называется «Сервис облачного хранилища файлов».

* 1. **Краткая характеристика области применения**

Этот сервис предназначен для хранения, редактирования файлов на удаленных серверах, а также для их обмена. Он организован по «облачному» принципу.

1. **Основание для разработки**
   1. **Основание для проведения разработки**

Основание для разработки Программы - задание на курсовой проект кафедры "Компьютерных систем, сетей и кибербезопасности" Национального аэрокосмического университета им. Н.Е. Жуковского "ХАИ" по дисциплине «Проектирование МПС».

* 1. **Наименование и условное обозначение разработки**

Наименование темы разработки – «Сервис облачного хранения файлов».

Условное обозначение темы разработки (шифр темы) – «СОХФ-01».

1. **Назначение разработки**
   1. **Функциональное назначение разработки**

Функциональным назначением сохранение, загрузка, выгрузка файла/файлов разных форматов, а также предоставление возможности редактировать их (файлы форматов docx, xlsx, pptx, txt).

* 1. **Эксплуатационное назначение**

Сервис предназначен для хранения файлов на удаленных серверах, которые неизвестны пользователю и представляются единым сервером, с которым и осуществляется работа. Доступ к данным происходит за счет авторизации пользователя посредством раннее созданного аккаунта. Авторизованное лицо так же может редактировать файлы форматов docx, xlsx, pptx, txt и создавать ссылку на один или группу файлов, с выбранным режимом доступа, и делиться ею для с доверенными лицами. Безопасность личных данных учетной записи возложена на сервис, который будет шифровать их и при извлечении дешифровать.

1. **Требования к программе или программному изделию**
   1. **Требования к функциональным характеристикам**
      1. **Общие сведения, назначения и состав сервиса СОХФ-01**

СОХФ-01 – это программный комплекс средств, предназначенных для организации облачного хранилища файлов всех форматов и размерами, не свыше 5 Гб. Данный сервис будет доступен только авторизованным пользователям, у которых есть своя учетная запись с личными файлами. Не авторизованным пользователям следует пройти регистрацию, чтобы получить доступ к хранилищу, выделенного сервисом. Он позволяет хранить абсолютно любые файлы, редактировать файлы форматов docx, xlsx, pptx; делиться файлом или группой файлов при помощи генерируемой ссылкой, которая неявно включает параметры режима доступа к ним (режима чтения и режим редактирования).

СОХФ-01 состоит из главного сервера, сервиса и множества серверов-хранилищ. В обязанности главного сервера входит:

* Регистрация пользователей;
* Авторизация пользователей;
* Аутентификация пользователей;
* Верификация получаемых данных;
* Вывод графического интерфейса web-приложения;
* Получение данных от клиентской стороны;
* Передача данных сервису;
* Получение данных от сервиса.

Назначением сервиса является передача и получение данных как от web-приложения, так и от устройств-серверов. Причина, по которой этот компонент отделен от других в том, что это повышает надежность и отказоустойчивость, а также защищенность всей системы в целом.

Т.к. концепция «облака» подразумевает использование множество удаленных друг от друга серверов, то в данном случае используются сервера, на которых будут храниться все пользовательские данные, предварительно зашифрованные web-приложением.

* + 1. **Требования к составу выполняемых функций**

Сервис должен выполнять следующие функции:

1. Регистрация, авторизация, аутентификация пользователей в системе;
2. Деавторизация пользователей;
3. Валидация данных, получаемых от пользователей;
4. Наделять пользователей при регистрации в системе ролью «user»;
5. Удалять аккаунты пользователей, наделенных ролью «admin», пользователем с ролью «senioradmin», а также аккаунты, имеющих роль «user»;
6. Удалять аккаунты пользователей, наделенных ролью «user», пользователем с ролью «admin»;
7. Загружать и выгружать файлы;
8. Сохранять пользовательские файлы на работоспособных и в то же время «активных» устройствах-серверах;
9. Шифровать и дешифровать файлы в соответствии с выбранным пользователем режимом: режим повышенной надежности (алгоритм RSA) и режим быстрой работы с файлами (алгоритм DES).
10. Ограничивать размер загружаемых файлов. Если пользователь будет пытаться загрузить файл, размером свыше 5 ГБ, отклонить запрос;
11. устанавливать или разрывать соединение с сервисом из приложения на устройстве-сервере;
12. Отображать краткую статистику по обмену данными на устройстве-сервере;
13. В случае возникновения программной ошибки на устройстве-сервере, продолжить функционирование приложения и отобразить сведения о возникшей ошибке;
14. Искать, сортировать, группировать, создавать файлы в web-приложении;
15. Предоставлять возможность просматривать или редактировать содержимое файлов форматов docx, xlsx, pptx, txt, pdf;
16. Изменять параметры созданного аккаунта: почта, номер мобильного телефона, двухфакторная авторизация, имя, возраст, аватар (графическое представление пользователя в системе);
17. Создавать учетную запись после успешной регистрации пользователя;
18. Удалять учетную запись.
    * 1. **Требования к организации входных данных**

Входные данные при регистрации: электронная почта и номер мобильного телефона должны быть существующими.

Входные данные при авторизации: учетные данные зарегистрированного пользователя.

Входные данные при загрузке файлов: размер файла должен быть не свыше 5 ГБ.

Входные данные при редактировании содержимого файла: файл должен представляться форматом или docx, или xlsx, или pptx, или txt.

Входные данные при просмотре содержимого файла: файл должен представляться форматом или docx, или xlsx, или pptx, или txt, или pdf.

* + 1. **Требования к организации выходных данных**

Выходные данные при регистрации: оповещение об успешной или неуспешной регистрации.

Выходные данные при авторизации: переход на страницу личного кабинета, либо оповещение о возникшей ошибке.

Выходные данные при загрузке файлов: оповещение об успешно проведенной операции либо о возникшей ошибке.

Выходные данные для отредактированного (но изменения которого не были зафиксированы пользователем) файла до наступления момента автосохранения: неизменившееся содержимое файла.

Выходные данные для отредактированного (но изменения которого не были зафиксированы пользователем) файла после наступления момента автосохранения: сохранный файл, с изменившимся или неизменившимся содержимым.

Выходные данные для отредактированного (изменения которого были зафиксированы пользователем) файла до наступления момента автосохранения: сохранный файл, с изменившимся или неизменившимся содержимым.

Выходные данные для отредактированного (изменения которого были зафиксированы пользователем) файла после наступления момента автосохранения: сохранный файл, с изменившимся или неизменившимся содержимым.

* + 1. **Требования к временны характеристикам**

Максимально допустимое время загрузки файла/файлов на сервер за один сеанс: 7 часов.

Максимально допустимое время скачивания файла/файлов за один сеанс: 7 часов.

* + 1. **Требования к безопасности**

Сервис должен гарантировать безопасность учетных данных пользователя (логин, пароль, адрес электронной почты, номер мобильного телефона), а также личных данных.

* 1. **Требования к надежности**
     1. **Перечень мероприятий по обеспечению надежного (устойчивого) функционирования сервиса**

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий:

1. организация бесперебойного питания технических средств;
2. использование лицензионного программного обеспечения;
3. допустимый диапазон температуры помещения, в котором эксплуатируются устройства-сервера: от +15 до +35 °С;
4. допустимый диапазон температуры помещения, в котором хранятся устройства-сервера: от +10 до +45 °С;
5. ограничение доступа к помещению, в котором находятся устройства-сервера;
6. адекватное, рациональное и исключительное наделение ролью «admin» учетных записей сотрудников.
   * 1. **Требования к обеспечению надежного функционирования сервиса**

Во процессе работы Программы возможны отказы по ее вине, других программных средств или аппаратного обеспечения.

Возможные отказы в работе Программы по ее вине не должны приводить к потере данных в файлах, используемых программой или пользователем в момент отказа.

Безопасность данных в случае аварийных ситуаций, которые не вызваны Программой, должна обеспечиваться средствами ОС.

Не допускается требование перезагрузки ОС после отказа Программы.

* + 1. **Время восстановления после отказа**

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), нефатальным сбоем (не крахом) операционной системы, не должно превышать стольких-то минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

* + 1. **Отказы из-за некорректных действий оператора**

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий.

* 1. **Условия эксплуатации**
     1. **Климатические условия эксплуатации**

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

* + 1. **Требования к видам обслуживания**

См. Требования к обеспечению надежного (устойчивого) функционирования программы.

* + 1. **Требования к численности и квалификации персонала**

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее двух штатных единиц - системный администратор и конечный пользователь программы (оператор).

Системный администратор должен иметь высшее профильное образование и сертификаты компании-производителя операционной системы. Задачи, выполняемые системным администратором:

1. поддержание работоспособности технических средств;
2. установка (инсталляция) и поддержание работоспособности системных программных средств - операционной системы;
3. установка (инсталляция) программы.

Конечный пользователь программы (оператор) должен обладать практическими навыками работы с графическим пользовательским интерфейсом операционной системы.

Персонал должен быть аттестован на II квалификационную группу по электробезопасности (для работы с конторским оборудованием).

* 1. **Требования к составу и параметрам технических средств**

К составу технических средств должны входить:

* Сервер либо стационарный компьютер:
  1. Процессор Intel Core i5 или Intel Xeon E3 1240v3 и выше;
  2. Память 8 Гб DDR3 и больше;
  3. Дисковая система 240 Гб SSD или HDD и больше;
  4. ОС Windows 8 или Windows Server 2012 и выше;
  5. IIS 8.0 и выше.
* Устройства под управлением следующих ОС на выбор: IOS, Android, Windows Phone.
  1. **Требования к информационной и программной совместимости**
     1. **Требования к информационным структурам и методам решений**

Требования к информационным структурам на входе и выходе, а также к методам решения не предъявляются.

* + 1. **Требования к исходным кодам и языкам программирования**

Для разработки web-приложения должны использоваться следующие языки программирования: C# (7.0 и выше) – разработка инфраструктуры серверной части и JavaScript – разработка «дружелюбного» графического интерфейса приложения, в среде разработке Visual Studio 2016 и выше.

Для разработки сервиса также должен использоваться С# 7.0 и выше, в среде разработке Visual Studio 2016 и выше.

Для разработки серверного приложения должен использоваться должен использоваться С# версии не ниже 4.0, в среде разработке Xamarin Studio.

* + 1. **Требования к программным средствам, используемым сервисом**

Web-приложение должно работать под управлением ОС Windows 8 и выше или Windows Server 2012 и выше, а также допускаются соответствующие пакты обновлений.

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы. Допускается применение соответствующего пакета обновления.

Для запуска web-приложения должен быть предустановлен веб-сервер IIS Express или IIS 5.0, один из которых следует корректно сконфигурирован в соответствии с расположением папки проекта web-приложения и ОС.

Параметры ОС настроены таким образом, чтобы исполнение программы было возможным. Для этого как минимум должны быть правильно настроены переменные окружения среды выполнения web-приложения и корректно установлены права пользователя.

* 1. **Специальные требования**

Специальные требования не предъявляются.

1. **Требования к программной документации**
   1. **Предварительный состав программной документации**

В результате разработки сервиса должна быть представлена следующая программная документация:

1. техническое задание;
2. схемы алгоритмов;
3. тексты программ;
4. пояснительная записка;
5. руководство оператора;

Кроме программного обеспечения на диске обязательно должна находиться пояснительная записка в электронном виде, содержащая весь комплект документации, предусмотренный в п. 5.1 настоящего технического задания.

# **ПРИЛОЖЕНИЕ Б Руководство клиента**

1. **Условия работы приложений Web-приложение**

В состав технических средств должно входить любое дэсктопное или мобильное устройство, с наличием любого браузера и доступом в Интернет.

1. **Использование**

Чтобы иметь возможность пользоваться сервисом, необходимо пройти этап регистрации или авторизации. Если пользователь не имеет ранее созданную учетную запись, ему следует зарегистрироваться, перейдя на соответствующую страницу регистрации, кликнув по ссылке, скриншот которой приведен в П.Б.1.

Для регистрации следует заполнить 3 поля: Email, Password и Confirm Password.

Email – поле, в которое надо ввести существующий адрес электронной почти.

Password – поле, в которое надо ввести придуманный пароль, необходимый также и при дальнейших авторизациях.

Confirm Password – поле, в которое надо ввести пароль, объявленный в «Password». Это необходимо, чтобы убедиться в том, что случайно допущенная опечатка пользователем в пароле не будет сохранена системой и избавит от перерегистрации или прохождения процесса восстановления пароля (ссылка на эту функцию показана на скриншоте П.Б.2. Появляется после неудачно пройденного этапа авторизации).

Для пользователей, имеющих раннее созданную учетную запись, следует заполнить поля Email и Password. В П.Б.3 приведен скриншот со страницей авторизации.

На «домашней» странице пользователю представляется весь основной функционал по манипуляции личными файлами и папками (П.Б.4). Чтобы загрузить файл или группу файлов, следует нажать на кнопку «Upload», которая выглядит, как на скриншоте в П.Б.5. Если же надо скачать файлы, необходимо выбрать целевые, т.е. отметить их флажками, как показано на скриншоте П.Б.6, а затем нажать на кнопку «Download» (П.Б.7).

Чтобы создать, удалить или переименовать файл/папку, необходимо кликнуть правой кнопкой мыши по выбранному элементу, после чего появится небольшое контекстное меню с перечнем этих функций. Скриншот контекстного меню показан в П.Б.8.

Перейти на страницу управления личным аккаунтом можно кликнув по ссылке, которая является логином (пример такой ссылки показан на скриншоте П.Б.9).

На странице управления личным аккаунтом отображены 4 ссылки, которые активируют ту или иную функцию, в зависимости от того, чем пользователь хочет воспользоваться: изменить пароль, установить другой способ авторизации (через внешние сервисы), привязать номер мобильного телефона или же включить двухфакторную авторизацию. Скриншот с функциями показан в П.Б.10.  
 Для того, чтобы деавторизоваться, предназначена ссылка «Log off» (П.Б.11).

# **ПРИЛОЖЕНИЕ В Руководство администратора**

1. **Сервер под web-приложение:**

В состав технических средств должно входить любое дэсктопное или лэптопное устройство с минимальными техническими характеристиками (CPU – 2.4Ghz, RAM – 4 GB, HDD – 1 GB).

1. **Серверные устройства под хранение файлов:**

В состав технических средств должно входить любое мобильное устройство с минимальными техническими характеристиками (CPU – 1.4Ghz, RAM – 512 MB, SSD – 1 GB).

1. **Установка web-приложения**

Для установки приложений выполните следующие шаги:

1. скачать файл-архив;
2. распаковать в любой удобной директории;
3. убедиться в наличии на конечном устройстве платформы .NET Framework с версией 4.5.2 и выше;
4. убедиться в наличии IIS Express 10.0 и выше;
5. В корневой директории IIS Express, в папке config, открыть файл applicationhost.config любым инструментом по редактированию текстовых файлов. В начало секции <sites> добавить следующий код:  
   <site name="Cloud Storage" id="1">

<application path="/" applicationPool="Clr4IntegratedAppPool">

<virtualDirectory path="/" physicalPath="..." />

</application>

<bindings>

<binding protocol="http" bindingInformation=":57460:localhost"/>

</bindings>

</site>

Вместо троеточия в свойстве physicalPath прописать путь к разархивированному приложению;

1. Запустить IIS Express.
2. **Установка сервиса**
3. скачать файл-архив;
4. распаковать в любой удобной директории;
5. убедиться в наличии платформы на конечном устройстве платформы .NET Framework с версией 4.5.2 и выше;
6. убедиться в наличии IIS Express 10.0 и выше;
7. убедиться в наличии Visual Studio не ниже 15;
8. запустить visual studio;
9. собрать решение и проект;
10. запустить проект «Service1»
11. **Установка приложения для устройств-серверов:**
12. скачать файл-архив;
13. распаковать в любой удобной директории;
14. убедиться в наличии платформы на конечном устройстве платформы .NET Framework с версией 4.5.2 и выше;
15. убедиться в наличии IIS Express 10.0 и выше;
16. убедиться в наличии Visual Studio 17 с предустановленным набором инструментов Xamarin;
17. запустить visual studio;
18. собрать решение и проект;
19. подсоединить мобильное устройство к ПК по USB-кабелю.
20. На мобильном устройстве с ОС Android установить в настройках режим «Отладка по USB»
21. Выбрать проект для мобильного устройтва: в случае ОС Android – MobileServer.Android, в случае ОС WindowsPhone 8.1 – MobileServer.WP, в случае ОС IOS – MobileServer.IOS.
22. Для целевого проекта убедиться в наличии строго необходимых библиотек под определенную версию ОС мобильного устройства, а также убедиться в том, что в настройках запускаемого проекта выбраны эти библиотеки.
23. Запустить проект.
24. **Возможные сообщения администратору при не правильном выполнении каких-то действий в программе**

Если в процессе работы программы возникают некоторые ошибки, сбои или приложение просто не может запуститься, и, если вам не удается разрешить программные конфликты самостоятельно, то опишите подробно проблему в сообщении, а затем отправьте на следующий адрес электронной почты: ivanbesch@outlook.com.

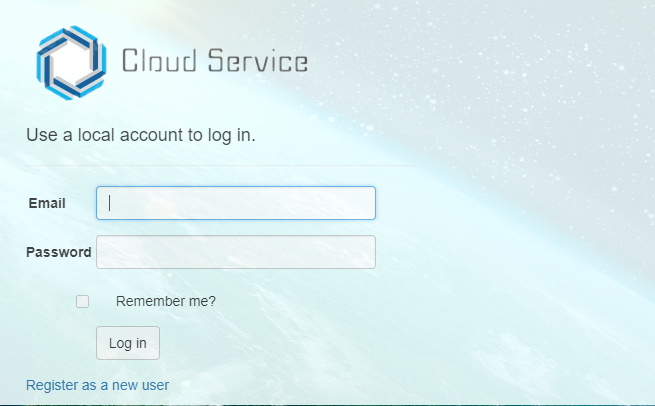
**П.Б.1. Ссылка на страницу регистрации.**



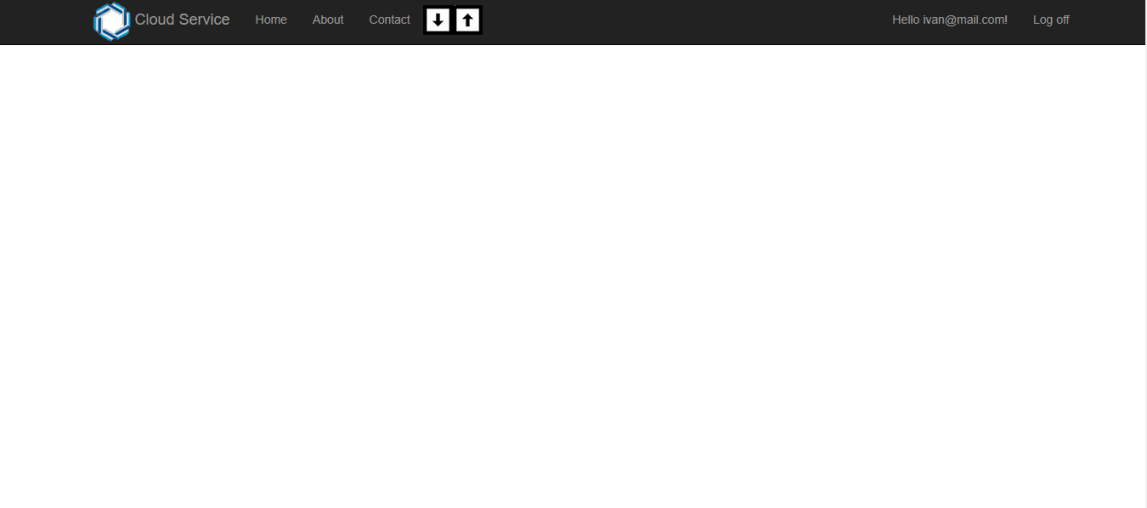
**П.Б.2. Ссылка на восстановление пароля.**



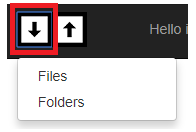
**П.Б.3. Страница авторизации.**



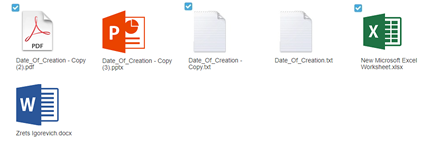
**П.Б.4. Панель функций и содержимое хранилища пользователя.**



**П.Б.5. Кнопка загрузки на сервер файлов/папок.**



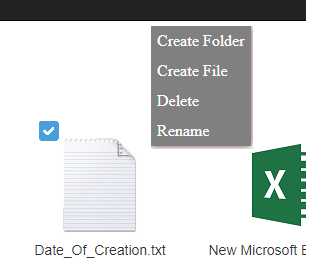
**П.Б.6. Файлы, отмеченные флажками.**



**П.Б.7. Кнопка скачивания.**



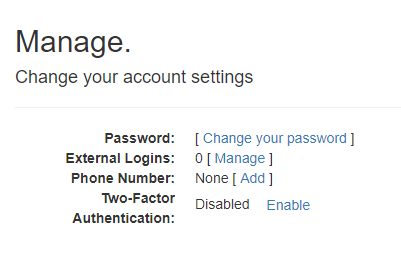
**П.Б.8. Контекстное меню при нажатии правой кнопки мыши.**

****

**П.Б.9. Ссылка-логин на учетные данные.**

****

**П.Б.10. Страница управления учетными данными.**



**П.Б.11. Ссылка на деавторизацию.**

