

Національний аерокосмічний університет ім. М.Є. Жуковського
«Харківський авіаційний інститут»

Факультет радіоелектронних систем літальних апаратів

Кафедра комп'ютерних систем та мереж

КУРСОВИЙ ПРОЕКТ (РОБОТА)

з _____ Периферійні пристрої _____
(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 _____ ст. викладач Желтухін А.А. _____
(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)

Оглавление

ВВЕДЕНИЕ	3
1. ПОСТАНОВКА ЗАДАЧИ	4
1.1. Анализ предметной области.....	4
2. ПРОЕКТИРОВАНИЕ.....	9
2.1. Выбор архитектуры.....	9
2.2. Уровень web-приложения.....	10
2.3. Уровень сервиса.....	14
2.4. Уровень хранилища данных.....	14
2.5. UML –диаграмма использования.....	15
2.6. Проектирование графического интерфейса: проектирование представлений Web-приложения и интерфейса программы для устройств-серверов	21
3. РАЗРАБОТКА.....	26
3.1. Разработка диаграммы классов.....	26
3.2. Поля, методы и свойства	27
3.3. Разработка алгоритмов.....	30
4. ВЕРИФИКАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММЫ.....	33
ВЫВОДЫ	35
СПИСОК ЛИТЕРАТУРЫ.....	35
ПРИЛОЖЕНИЕ А Техническое задание	36
ПРИЛОЖЕНИЕ В Руководство оператора	41

ВВЕДЕНИЕ

Пояснительная записка курсового проекта включает в себя:

- страниц – 50;
- рисунков – 35;
- таблиц – 9;
- приложений – 4;
- источников – 6.

Целью данной курсовой работы является разработка приложения “Сервис облачного хранения данных”.

Задачами курсовой работы являются:

- 1) Реализация алгоритма загрузки и выгрузки файлов любых форматов и размеров;
- 2) Реализация облачной инфраструктуры;
- 3) Поддержание целостности данных пользователя;

1. ПОСТАНОВКА ЗАДАЧИ

1.1. Анализ предметной области

Современные компьютерные устройства и современное компьютерное оборудование с каждым днем становится все мощнее и мощнее, а также требует все больше и больше памяти, на которой будет храниться гигантское количество, по меркам 20 и начала 21 столетия, информации. Это проблема решалась и до сих пор решается разными способами: увеличение объема памяти на устройстве, использование легко переносимых флеш-накопителей, разделение данных между двумя или несколько схожими устройствами, способных передавать информацию кабельным или радиоволновым путем. Однако эту актуальную проблему призваны легко решить сервисы облачного хранения данных, без каких-либо существенных денежных затрат для конечного пользователя.

Облачное хранилище или «Облако» – это онлайн-хранилище, которое распределяет все пользовательские данные на множестве удаленных друг от друга серверов. Об организации данного сервиса никто не может знать, кроме специалистов, принимавших участие в создании комплекса ПО для реализации этого проекта, поэтому это повышает надежность всей системы. Хакерские атаки должны быть направлены на множество точек на земном шаре, чтобы нарушить работу всего «облака», и как следствие утраты пользовательских личных данных и личной информации. Однако даже это будет осуществить непросто, в особенности, если у системы есть мощные ответные механизмы защиты и высоконадежные, отказоустойчивые с резервными компонентами.

Помимо хранения информации, данная инфраструктура позволяет проводить самые разнообразные операции с данными, манипуляция которыми зачастую необходима нам в тех случаях, когда нет возможности скачать все необходимое на устройство, используемое в данный момент, или же, которое не содержит тех необходимых инструментов, которые предоставляет сервис облачного хранения данных: редактирование, конвертирование, архивирование, воспроизведение и т.д. Поэтому оно поддерживается всеми видами устройств, имеющих выход во всемирную сеть.

На сегодняшний день существует множество сервисов облачного хранилища. Все они имеют как преимущества, так и недостатки, и чтобы разобраться, каким функционалам должен обладать сервис облачного хранения данных, чтобы быть конкурентоспособным и востребованным, рассмотрим существующие аналоги:

GoogleDrive

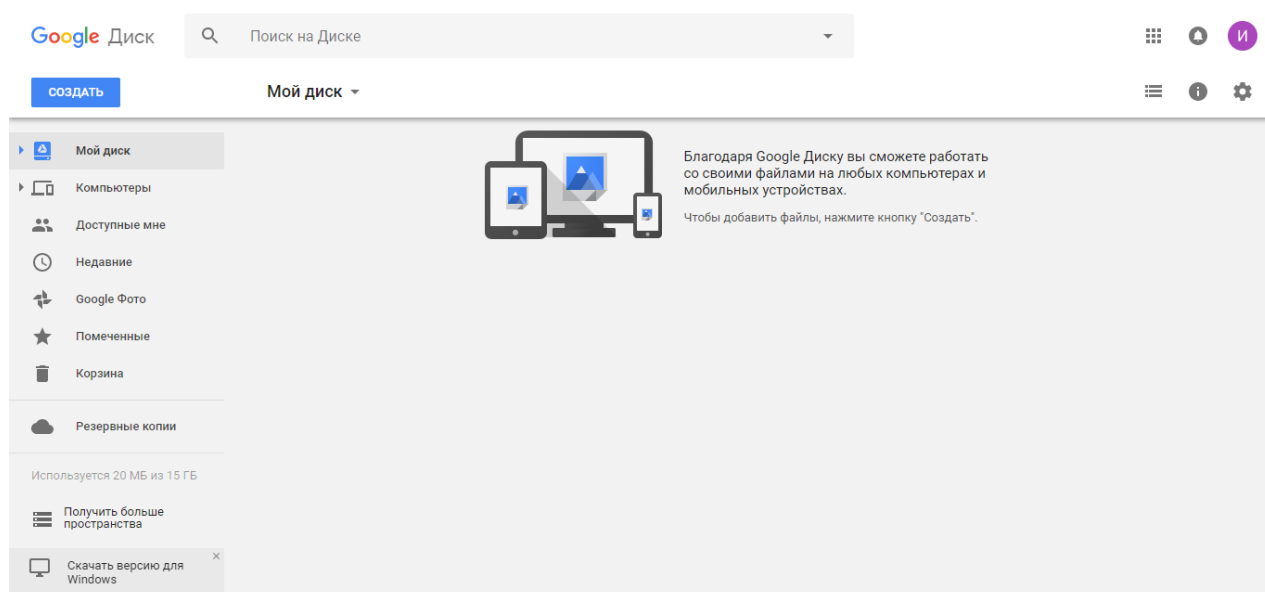
Один из самых популярных сервисов облачного хранилища, преимуществом которого является многофункциональный web-интерфейс, в котором доступны следующие операции:

- Редактирование и создание документов в режиме совместного доступа;
- Просмотр мультимедийного контента;
- Расширение почти всех существующих надстроек;
- Поиск документов;
- Управление правами доступа.

На фоне всех других подобных сервисов можно выделить следующие преимущества и недостатки:

+	–
Неограниченное пространство под хранение изображение и фотографий.	Выделенное свободное пространство будет использоваться другими смежными сервисами.
Гибкий web-интерфейс, позволяющий настраивать рабочую среду под нужды пользователя.	Десктопная версия приложения не обладает какими-либо преимуществами.
При регистрации сразу предоставляется 15 Гб бесплатного дискового пространства	

Таблица 0.1. Преимущества и недостатки сервиса GoogleDrive



Изображение 0.1. Web-интерфейс сервиса GoogleDrive

Microsoft OneDrive

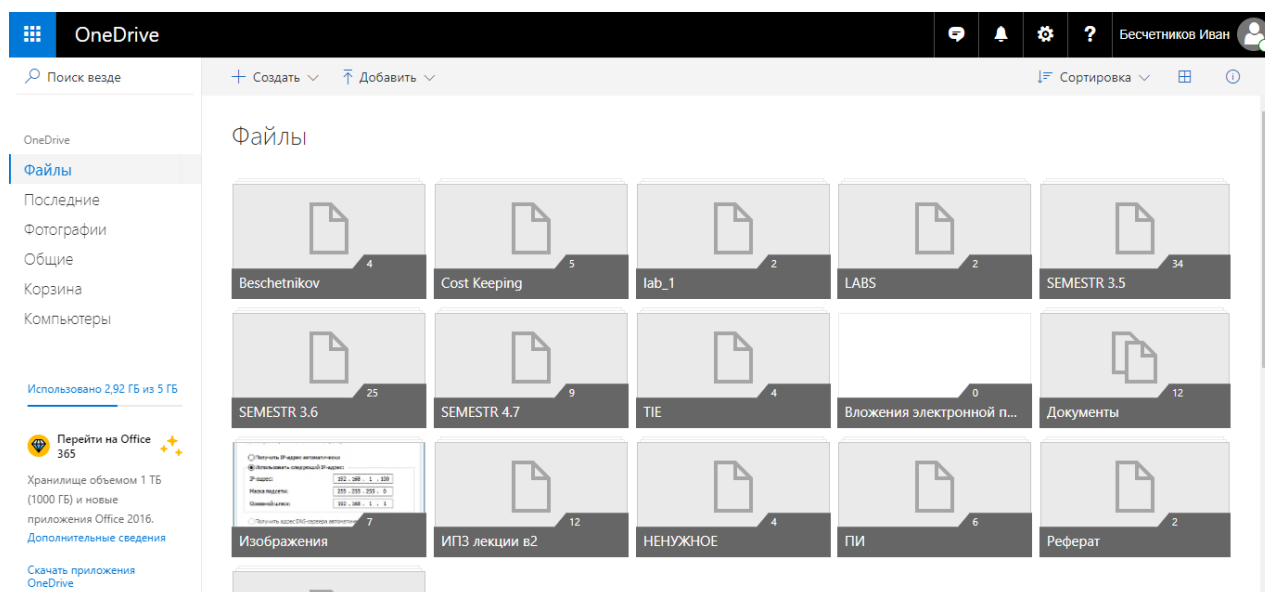
Надежное онлайн-хранилище от компании Microsoft, изначально интегрированное в ОС Windows 10, что дает возможность работать со своими данными на локальном компьютере с дальнейшей автосинхронизацией с «облаком». Отличительная особенность данного сервиса – интеграция с пакетом Office 365, дающая ряд преимуществ при работе с документами разного типа: Word, Excel, PowerPoint, Visio, Access, Publisher, OneNote и т.д. Основные характеристики сервиса:

- Бесплатное место: 5 Гб.
- Максимальный размер одного файла: 10 Гб.
- Максимальный объем пространства: 5 Тб.
- Совместимость с мобильными ОС: поддерживается на Android выше 4.0, iOS выше 9.0, Windows Phone 7 / 8, на Symbian Belle и на MeeGo 1.2.

Преимущества и недостатки сервиса:

+	–
Интеграция с пакетом Office 365.	Отсутствие дополнительных функций.
Высокая скорость и стабильная работа.	Бесплатный тариф предоставляет всего 5 Гб свободного дискового пространства.
Скидки при покупке дискового пространства с Office 365.	

Таблица 0.2. Преимущества и недостатки сервиса OneDrive



Изображение 0.2. Web-интерфейс сервиса OneDrive

Dropbox

Является одним из самых первых сервисов «облачной» архитектуры, который за все свое долгое время существования и развития успел доработать каждый узел инфраструктуры. Dropbox, как и большинство других популярных сервисов, предоставляет довольно функциональный и удобный web-интерфейс. Также он работает по модели Freemium, позволяющий при создании аккаунта выбрать объем свободного дискового пространства с дальнейшим увеличением, но при внесении оплаты.

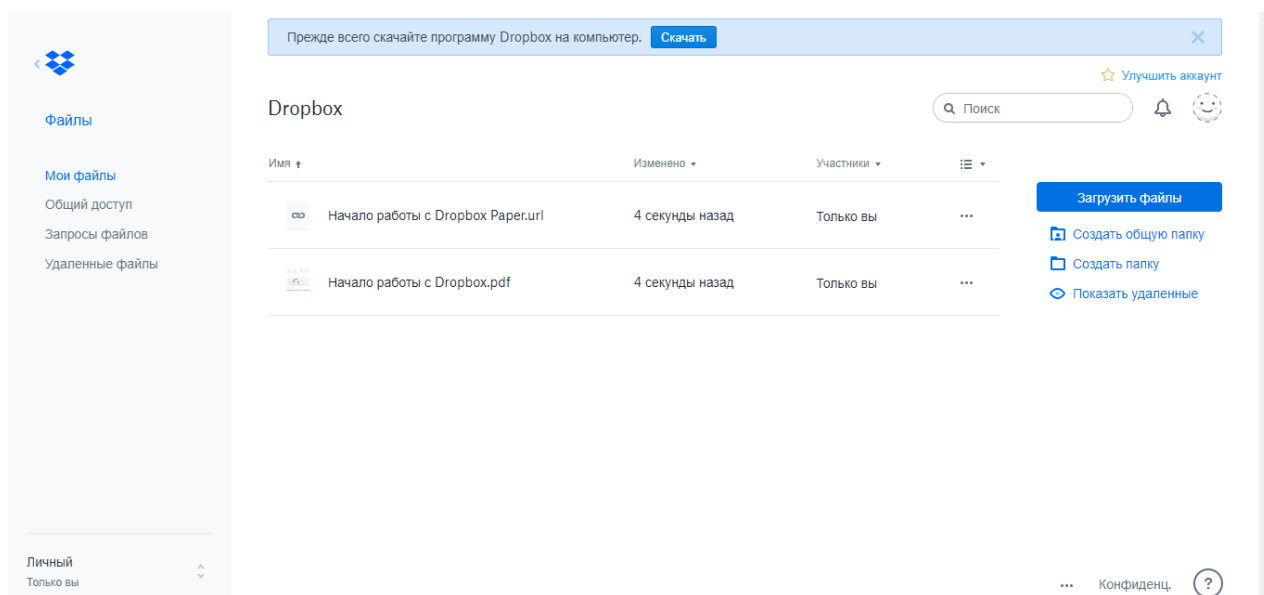
Основные характеристики сервиса:

- Бесплатное место: 2 Гб.
- Максимальный размер одного файла через web-интерфейс – 20 Гб, через десктопное приложение – без ограничений.
- Максимальный объем пространства: 1 Тб.
- Совместимость с мобильными ОС: поддерживается на Windows, macOS, Linux; на мобильных ОС Android, iOS, Windows Phone и BlackBerry.

Преимущества и недостатки сервиса:

+	–
Гибкий и функциональный web-интерфейс.	Максимально количество допустимых функций распространяется только для бизнес-клиентов.
Удобство в использовании на клиентских версиях.	Бесплатный тариф предоставляет всего 2 Гб свободного дискового пространства.
256-битное шифрование AES и шифрование SSL	
Лучшая среди аналогов технология синхронизации	
Интеграция с Microsoft Office 365	
Неограниченное восстановление файлов и журнал версий	
Ссылки доступа с паролем и сроком действия	
Возможность настраивать и управлять уровнями доступа	

Таблица 0.3. Преимущества и недостатки сервиса Dropbox



Изображение 0.3. Web-интерфейс сервиса Dropbox

Mega

Перспективный облачный файлообменник, предоставляющий максимально удобный набор инструментов для пользователей, которые не предпочитают выкладывать деньги за пользование небольшим объемом памяти под хранение личных данных. Такой сервис также предоставляет пользователям возможность делиться своими личными файлами по схеме «friend-to-friend», используя для этих целей надежный криптографический алгоритм шифрования AES, причем открытый ключ известен только тому пользователю, которому станет доступна ссылка на файл обмена.

«Mega» позиционируется как сервис, защищающий личные данные пользователей, поэтому используется сквозное шифрование.

Данный web-сервис существует в трех вариантах для использования: web-приложение, desktop-приложение и мобильное приложение.

Основные характеристики сервиса:

- Бесплатное место: 50 Гб.
- Максимальный размер одного файла через web-интерфейс – 10 Гб, через десктопное приложение – без ограничений.
- Максимальный объем пространства: 4 Тб.
- Совместимость с мобильными ОС: поддерживается на Windows, macOS, Linux; на мобильных ОС Android, iOS, Windows Phone и BlackBerry.

Преимущества и недостатки сервиса:

+	–
Плагины для браузеров.	Не высокая скорость доступа
Интегрированный чат.	Существуют лимиты на трафик, исчезающие при платной подписке.
2048-битное шифрование AES и шифрование SSL	
Предоставляется большой объем бесплатного хранилища – 50 Гб, что является рекордным показателем среди других существующих облачных сервисов.	

Таблица 0.4. Преимущества и недостатки сервиса Dropbox

2. ПРОЕКТИРОВАНИЕ

2.1. Выбор архитектуры

В ходе проектирования инфраструктуры сервиса была выбрана трехуровневая архитектура:

- Уровень web-приложения;
- Уровень сервиса;
- Уровень хранилища данных.

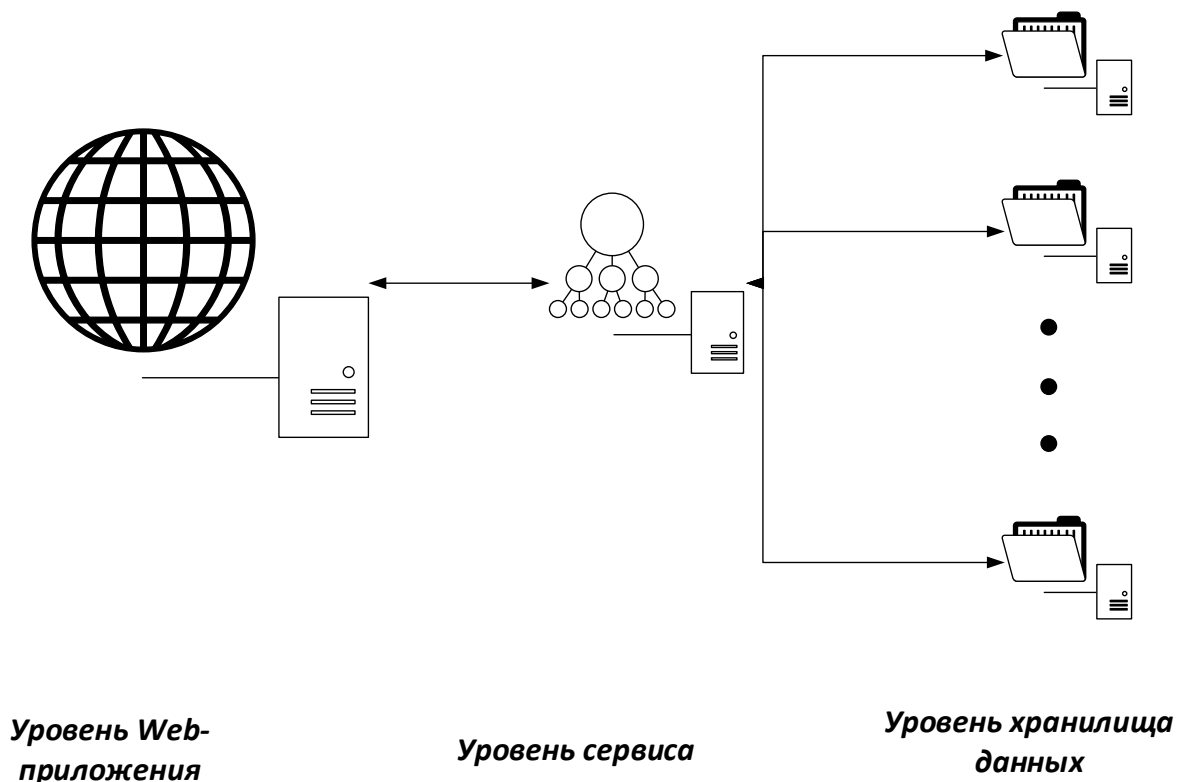


Рис. 2. 1 – трехуровневая архитектура проекта

2.2. Уровень web-приложения

Уровень web-приложения представляет из себя приложение, которое является главным во всем проекте и служит для контроля всех пользовательских действий: регистрация, авторизация, аутентификация, валидация. В случае неуспеха выполнения данных процессов, будет выброшена системой соответствующая ошибка, которую пользователь будет обязан исправить, следуя полученной инструкции от браузера или web-приложения.

Для реализации этого узла системы был использован ASP.NET MVC 5 фреймворк для создания web-приложений.

ASP.NET MVC 5 – это высоконадежная инфраструктура web-приложения, которая призвана обеспечить создание легковесной надежной архитектуры проекта, достигающаяся путем заранее реализованных полноценных отказоустойчивых и проверенных модулей. Она так же поддерживает расширяемость проектов, тестируемость, поэтому сопровождение становится легче и проще.

Эта платформа имеет в наличии отличный инструментарий, позволяющий создавать простые, а вместе с ним и гибкие модули. Встроенные вспомогательные методы HTML генерируют ясный и удобочитаемый код разметки, соответствующий всем современным стандартам описания графических web-интерфейсов. Высокопроизводительная маршрутизация дает возможность создавать удобные, читабельные URL-адреса, благодаря чему программист волен придумывать любые осмысленные имена страницам, или же имена, которые скрывают некоторую логику работы всего web-приложения.

Проекты, написанные на ASP.NET MVC 5 поддерживают расширяемость, а это значит, что каждый отдельный модуль или модуль, встраиваемый в систему можно проверить в модульном тестировании, а интеграционные тесты будут отслеживать состояние всей системы в фоновом режиме работы.

Данная платформа для разработки web-приложений использует архитектурный шаблон MVC. К преимуществам этого шаблона можно отнести следующее:

- Пользователь взаимодействует с приложением в соответствии с естественным циклом: пользователь совершает некоторое действие, в ответ на которое модель изменяет свое состояние и рендерит определенное представление, что хорошо вписывается в логику работы запросов и ответов протокола HTTP.

- Ряд web-приложений нуждаются в разделении модулей всего проекта: HTML-представления, компоненты по улучшению web-интерфейса, базы данных, бизнес-логика, сервисы и т.д. Все это отлично вписывается в концепцию MVC.

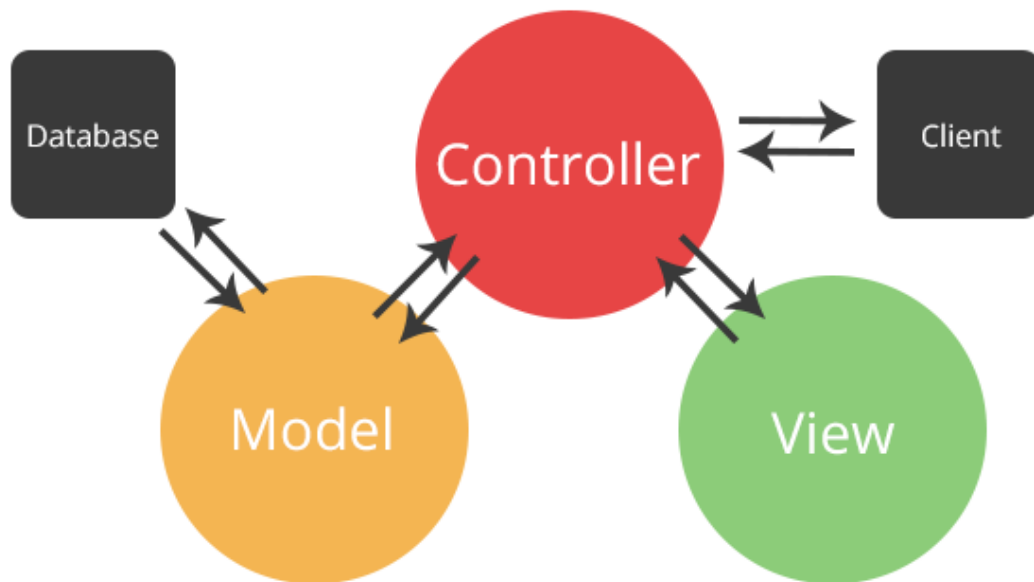


Рис. 2.1.1 – Архитектура MVC-проектов

Инфраструктура ASP.NET MVC Framework реализует этот шаблон, причем с доработанными и улучшенными некоторым сценариями использования данного паттерна.

Расширяемость

Инфраструктура MVC Framework построена таким образом, компоненты которой являются взаимно независимыми и представляются в виде абстрактных базовых классов. Поэтому существуют следующие схемы использования встроенных компонентов:

- Полное использование встроенных компонентов;
- Частичное использование встроенных компонентов с некоторой корректировкой, реализуя собственный класс с наследованием от абстрактного класса или интерфейса всех его методов и свойств;
- Полная замена существующего компонента собственнореализованным.

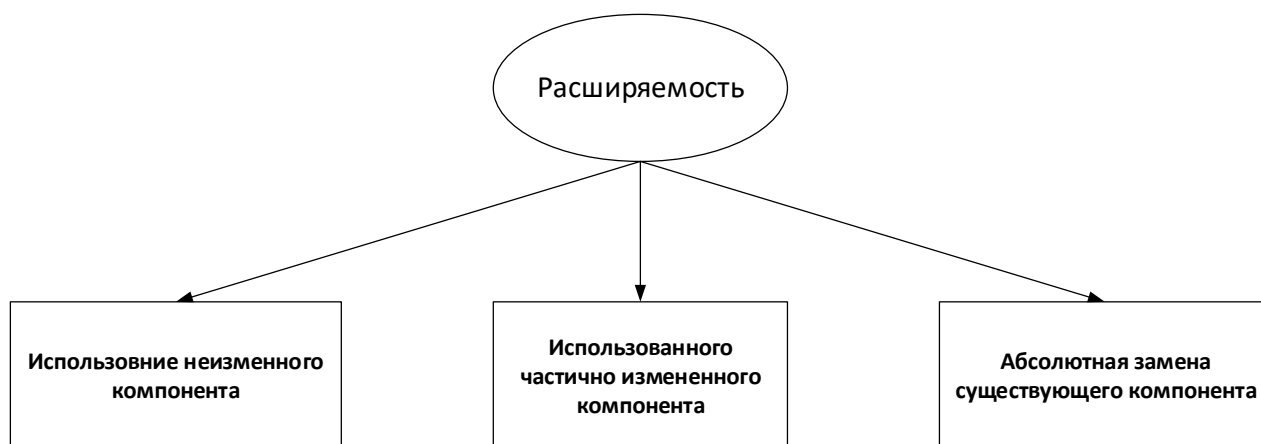


Рис. 2.1.2 – Сценарии использования встроенных компонентов ASP.NET MVC Framework

Контроль HTTP и HTML

Инфраструктура ASP.NET MVC Framework генерирует ясную и простую разметку, при использовании HTML-хелперов. Однако помимо простой генерации HTML-разметки, она способна на построение элегантных, легко обрабатываемых блоков кода-разметки, которые будут в значительно большей степени лучше отрендерены, нежели чем бесчисленное множество сложных блоков, которые так же трудно управляемы. Генерируемый код вдобавок будет оформлен красивыми и «дружелюбными» стилями CSS и плавной анимацией JavaScript.

С другой стороны, если требуется создание более сложных элементов управления, такие как DatePicker, Clock, CascadeMenu и т.д., то можно воспользоваться встроенными библиотеками, которые значительно ускоряют и упрощают процесс разработки таких элементов. Например, библиотеки jQuery и Bootstrap CSS уже в последних версиях сред разработок, таких как Visual Studio и Code Studio, имеются, и значатся, как встроенные компоненты, эффективность которых оспаривать никто не станет.

Так же, в отличие от страниц WEB.FORMS страницы ASP.NET MVC Framework значительно меньше по размеру, даже несмотря на то, что скорость соединения с web-сайтами возросла в разы, тем не менее это повышает комфорт использования для конечного пользователя и ускоряет процесс запуска самого web-приложения. Это достигается за счет отсутствия View State-данных страницы, с которыми в основном работает платформа WEB.FORMS.

ASP.NET MVC тесно связана с HTTP-протоколом. Это позволяет контролировать запросы и ответы, возникающие между сервером и клиентом. Благодаря этому, можно точно корректировать каждый момент работы двух сторон. А при использовании AJAX-запросов не будут нужны никакие автоматические отправки обратного вызова, т.к. она проста и не использует состояние клиентской стороны.

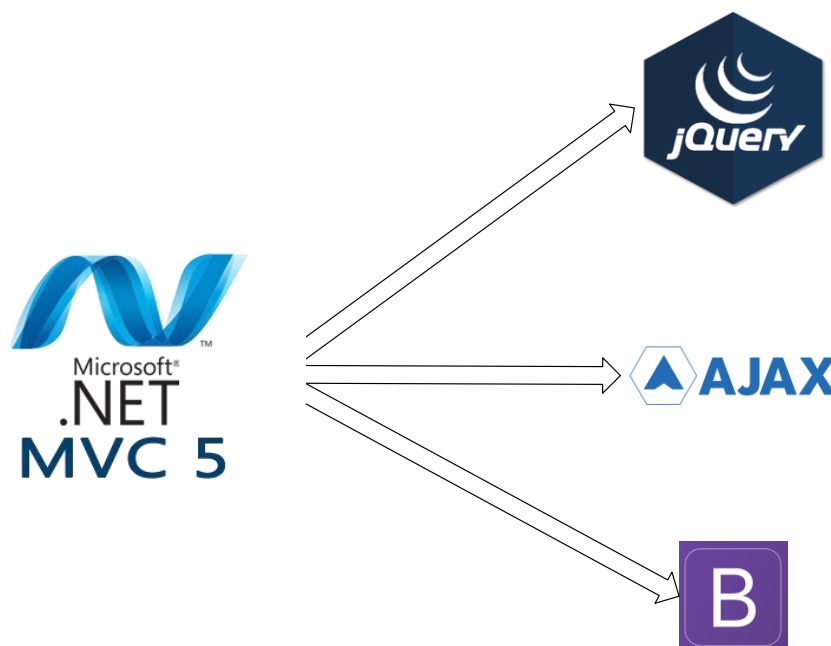


Рис. 2.0.3 – Библиотеки, встроенные по умолчанию в ASP.NET MVC 5

Тестируемость

Разделение ответственности между всеми частями web-приложения – то, что позволяет данный фреймворк, делая сопровождение и тестирование отдельных частей всего приложения гораздо проще и удобнее. Однако на этом возможности ASP.NET MVC Framework платформы не ограничены. Для каждого модуля компонентно-ориентированного проекта всей системы, она обеспечивает структурированность, которая так необходима при использовании модульного тестирования.

Помимо модульного тестирования, существуют также средства, призванные симитировать как действия пользователя, так и взаимодействие с ним в графическом интерфейсе приложения, обработка которых автоматически осуществляется всей инфраструктурой, и нет больше необходимости следить за изменениями структуры HTML-разметки, или же отслеживать генерируемые HTML-элементы и идентификаторы CSS.

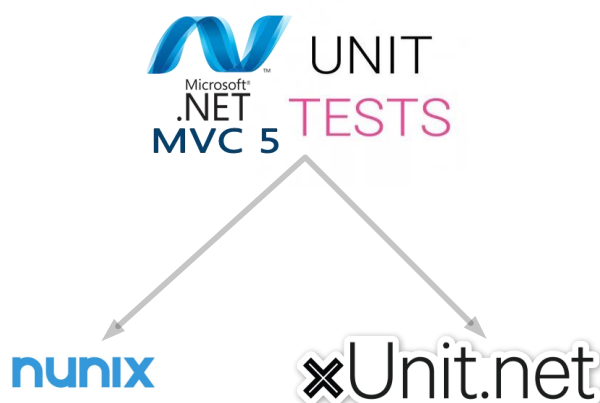


Рис. 2.0.4 – Интегрированные инструменты модульного тестирования

2.3.Уровень сервиса

Уровень сервиса. Данный уровень является интерфейсом по обмену данными между web-приложением и серверными приложениями. Его работа осуществляется независимо от работы web-приложения, однако уровень web-приложения может получать некоторую информацию о процессе транзакции по извлечению или загрузке файлов.

Уровень хранилища данных. На данном уровне оперируются один или группа файлов, которые следует сохранить или извлечь из памяти устройства. При извлечении данные передаются уровню, с которым он непосредственно связан – уровень сервиса. В случае возникновения ошибок, приложение этого уровня должно сообщить приложению сервиса об этом, а также ее название. Тогда уровень сервиса в обязательном порядке предпримет попытку по устранению ошибки, если, конечно, она логическая и не связана с аппаратной составляющей серверов-хранилищ. Примерами таких ошибок могут выступать:

- Потеря пакетов данных при передаче или извлечении данных;
- Временная недоступность сервера-хранилища;
- Нехватка свободной памяти;

2.4.Уровень хранилища данных

Уровень хранилища данных. Данный уровень представляет из себя множество приложений, установленных на устройствах, способных хранить, получать и извлекать информацию из памяти. Это может быть любой ПК, любая рабочая станция или любой гаджет, который работает под управлением четырех операционных систем: Windows 10 (любая редакция), Android (любая версия), iOS (любая версия) и Windows Phone 8.1. Такая интероперабельность достигается за счет использования Xamarin.



Рис. 2.3.1 – кроссплатформенность Xamarin

Xamarin – это инструмент разработки кроссплатформенного ПО, позволяющий создавать приложения под управлением платформы .NET, которая в свою очередь предоставляет возможность использовать такие языки программирования как C#, F# и Visual Basic; а также приложения, написанные на языке программирования C++.

Несомненно, это удобное решение для проектов, в которых реализуется кроссплатформенность за достаточно короткие сроки. К преимуществам данного выбора можно отнести следующее:

- Создание унифицированного кода для устройств, работающих под управлением разных ОС;
- Предоставляется возможность обращаться к нативным API разных платформ;
- Использование в качестве языка программирования – C#, который прост в понимании и использовании.

А к недостаткам данного выбора можно отнести следующее:

- Приложения выходят менее производительными;
- Приложения используют больше памяти, чем, если бы они были написаны на Java или C++ для отдельно взятой платформы.

В данном проекте использовались Windows 10 и Android платформы. Android с пакетами для двух версий: 4.2 и 7.1.

2.5.UML –диаграмма использования

Функции, которые реализованные в проекте, можно представить с помощью диаграммы прецедентов.

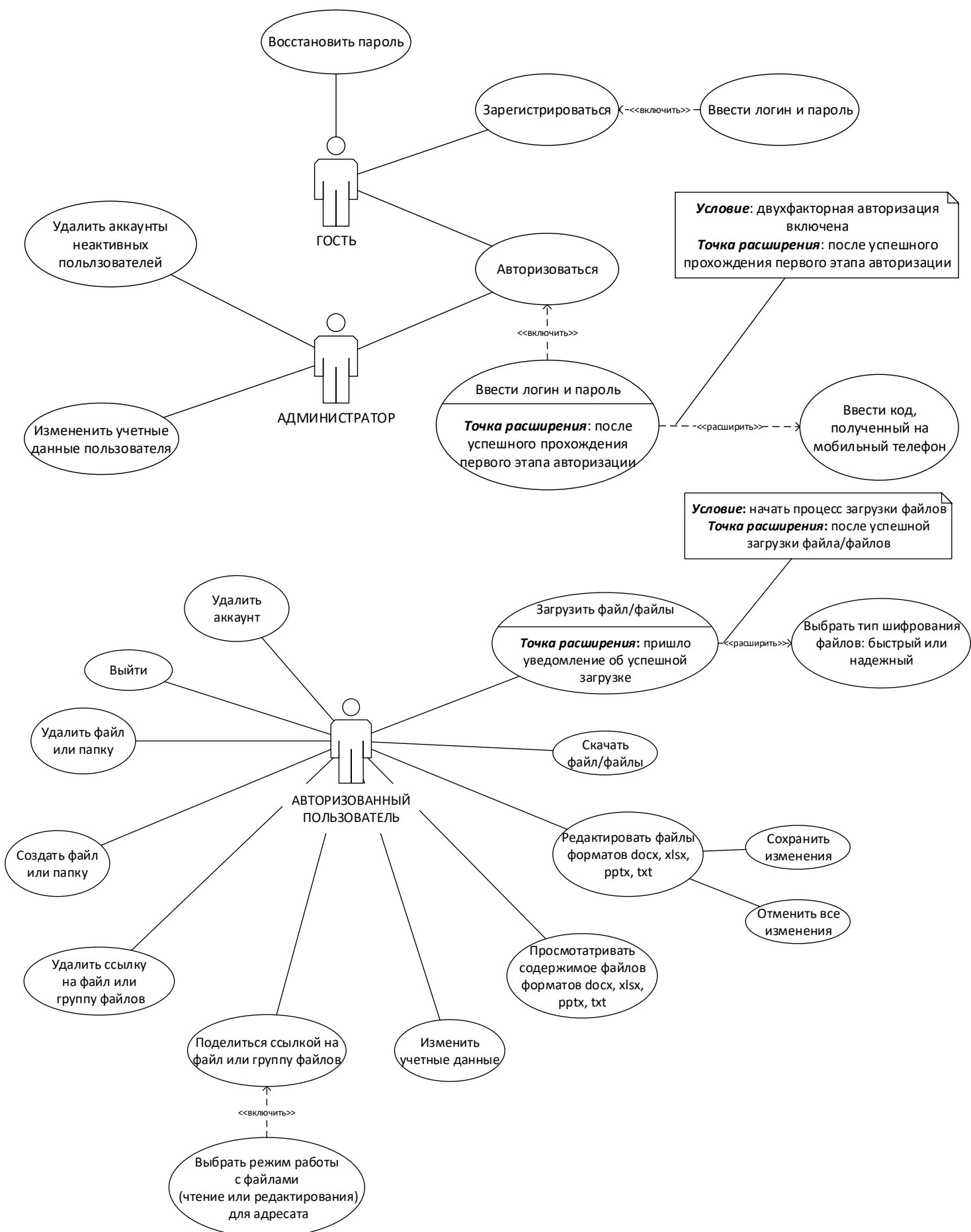


Рис. 2.4 – UML-диаграмма использования. Взаимодействие пользователя с интерфейсом программы и ее функциями.

Таблица 2.4.1 – Прецедент «Зарегистрироваться»

Название прецедента	«Зарегистрироваться»
Описание	Данный прецедент позволяет регистрироваться на данный сервис.

Таблица 2.4.2 – Прецедент «Авторизоваться»

Название прецедента	«Авторизоваться»
Описание	Данный прецедент позволяет авторизоваться.

Таблица 2.4.3 – Прецедент «Добавить запись»

Название прецедента	«Добавить запись»
Описание	Данный прецедент позволяет добавить строку-запись, в которой можно осуществить следующие операции: ввести сумму, выбрать категорию, добавить категорию, выбрать дату, добавить комментарий к записи.

Таблица 2.4.4 – Прецедент «Восстановить пароль»

Название прецедента	«Восстановить пароль»
Описание	Данный прецедент позволяет восстановить доступ к аккаунту, в случае если пользователь забыл пароль.

Таблица 2.4.5 – Прецедент «Удалить аккаунты неактивных пользователей»

Название прецедента	«Удалить аккаунты неактивных пользователей»
Описание	Данный прецедент позволяет удалять администратору аккаунты тех пользователей, а также на свое усмотрение, которые на протяжении длительного времени не пользовались данным сервисом.

Таблица 2.4.6 – Прецедент «Изменить учетные данные пользователя»

Название прецедента	«Изменить учетные данные пользователя»
Описание	Данный прецедент позволяет администратору изменять учетные данные пользователя, такие как логин, пароль, номер мобильного телефона и другое, если пользователю не удастся войти в свой аккаунт.

Таблица 2.4.7 – Прецедент «Изменить учетные данные пользователя»

Название прецедента	«Изменить учетные данные пользователя»
Описание	Данный прецедент позволяет администратору изменять учетные данные пользователя, такие как логин, пароль, номер мобильного телефона и другое, если пользователю не удастся войти в свой аккаунт.

Таблица 2.4.8 – Прецедент «Удалить аккаунт»

Название прецедента	«Удалить аккаунт»
Описание	Данный прецедент позволяют пользователю при необходимости удалить собственный аккаунт.

Таблица 2.4.9 – Прецедент «Выйти»

Название прецедента	«Выйти»
Описание	Данный прецедент позволяет выйти из учетной записи.

Таблица 2.4.10 – Прецедент «Удалить файл или папку»

Название прецедента	«Удалить файл или папку»
Описание	Данный прецедент позволяет пользователю удалять существующую папку или файл.

Таблица 2.4.11 – Прецедент «Создать файл или папку»

Название прецедента	«Создать файл или папку»
Описание	Данный прецедент позволяет пользователю создать папку или файл.

Таблица 2.4.12 – Прецедент «Удалить ссылку на файл или группу файлов»

Название прецедента	«Удалить ссылку на файл или группу файлов»
Описание	Данный прецедент позволяет удалять пользователю ранее созданную ссылку на файл или группу файлов для лиц, получивших доступ к ним.

Таблица 2.4.13 – Прецедент «Поделиться ссылкой на файл или группу файлов»

Название прецедента	«Поделиться ссылкой на файл или группу файлов»
Описание	Данный прецедент позволяет генерировать ссылку на файл или группу файлов для доверенных пользователей.

Таблица 2.4.14 – Прецедент «Изменить учетные данные»

Название прецедента	«Изменить учетные данные пользователя»
Описание	Данный прецедент позволяет пользователю изменять некоторые учетные данные: логин, пароль, ФИО, аватар, номер мобильного телефона и т.д.

Таблица 2.4.15 – Прецедент «Просматривать содержимое файлов форматов docx, xlsx, pptx, txt»

Название прецедента	«Просматривать содержимое файлов форматов docx, xlsx, pptx, txt»
Описание	Данный прецедент позволяет пользователю просматривать содержимое файлов с такими расширениями.

Таблица 2.4.16 – Прецедент «Редактировать файлы форматов docx, xlsx, pptx, txt»

Название прецедента	«Редактировать файлы форматов docx, xlsx, pptx, txt»
Описание	Данный прецедент позволяет редактировать файлы с такими расширениями.

Таблица 2.4.17 – Прецедент «Сохранить изменения»

Название прецедента	«Сохранить изменения»
Описание	Данный прецедент позволяет пользователю сохранить все внесенные изменения в файле при его редактировании.

Таблица 2.4.18 – Прецедент «Скачать файл/файлы»

Название прецедента	«Скачать файл/файлы я»
Описание	Данный прецедент позволяет скачивать файл или группу файлов из сервиса на используемое в данный момент устройство пользователем.

Таблица 2.4.19 – Прецедент «Загрузить файл/файлы»

Название прецедента	«Загрузить файл/файлы»
Описание	Данный прецедент позволяет пользователю загружать файлы или группу файл в сервис из устройства, используемое в данный момент времени.

2.6. Проектирование графического интерфейса: проектирование представлений Web-приложения и интерфейса программы для устройств-серверов

Пользовательский интерфейс был реализован с использованием HTML, CSS и движка представлений Razor. `_ViewStart` – представление, визуализирующееся при запуске приложения. Оно не содержит визуальных элементов, а устанавливает представление шаблона (`_Layout`). Представление `_Layout` определяет шаблон страниц логирования и регистрации приложения. Всего в приложении имеется порядка 30 представлений.

Страница логирования предлагает ввести существующие данные учетной записи пользователя в соответствующие поля для ввода: Email и Password.

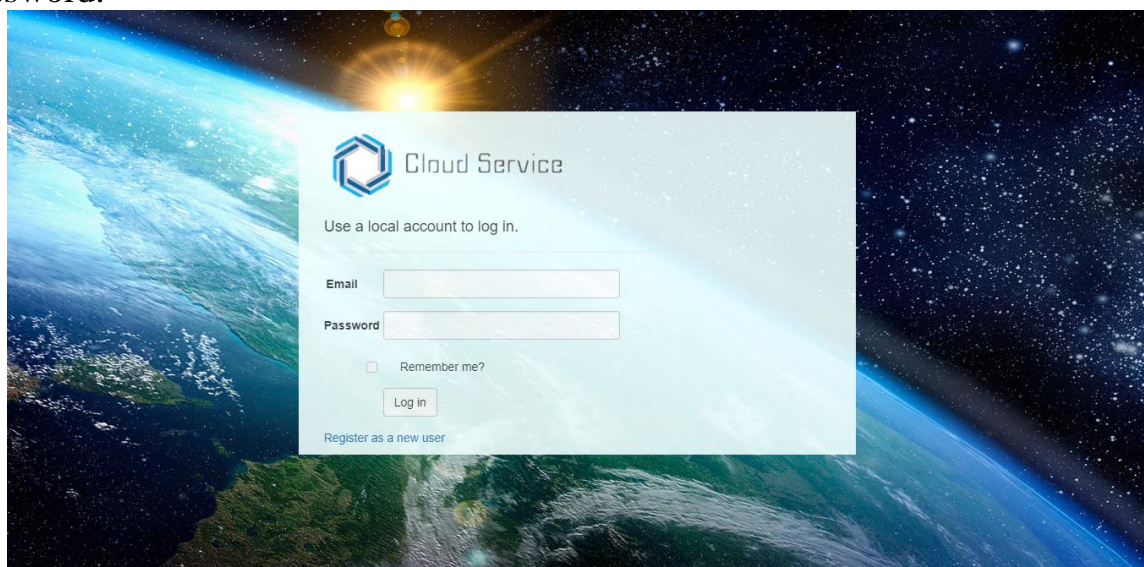


Рис. 2.6.1 – Страница авторизации.

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

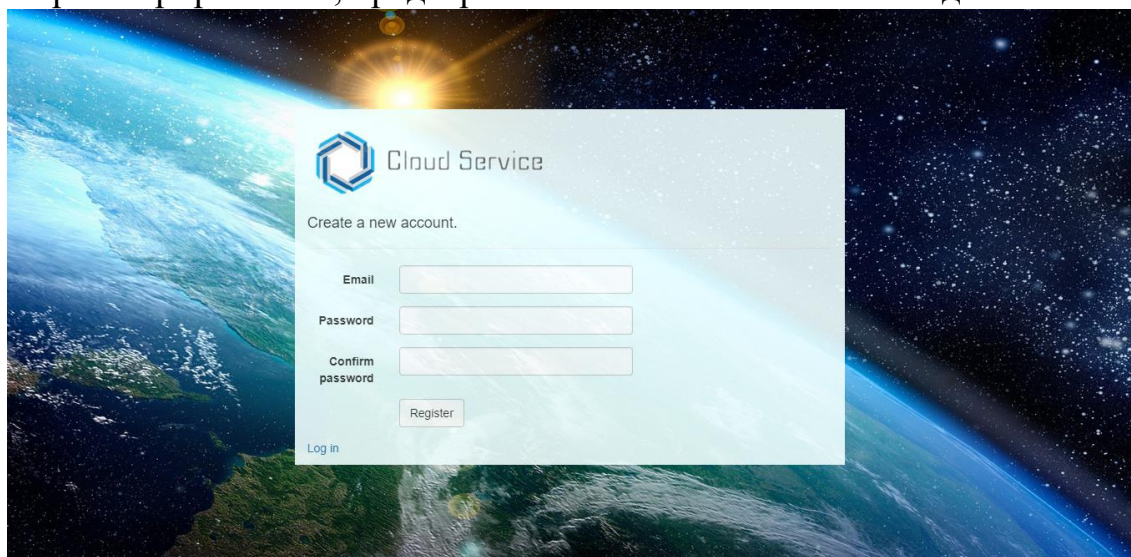


Рис. 2.6.2 – Страница регистрации.

Представление `_Layout2` определяет шаблон всех остальных страниц. В верхней части второго шаблона находится логотип, который является ссылкой на домашнюю страницу приложения, а также множество ссылок на частичные представления, необходимое по работе с полем, содержащим все ранее загруженные пользовательские файлы и папки. При необходимости, есть 2 управляющие ссылки учетной записью, которые позволят деавторизоваться и настроить параметры аккаунта



Рис. 2.6.3 – 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 «Download», при нажатии которой выводится небольшое контекстное меню, с выбором объектов, выгружаемых на сервер: файлы или папки.

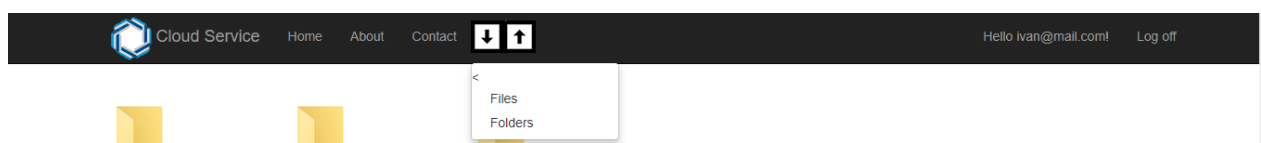


Рис. 2.6.4 – Кнопка загрузки файлов на сервер, а так же кнопка скачивания.

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

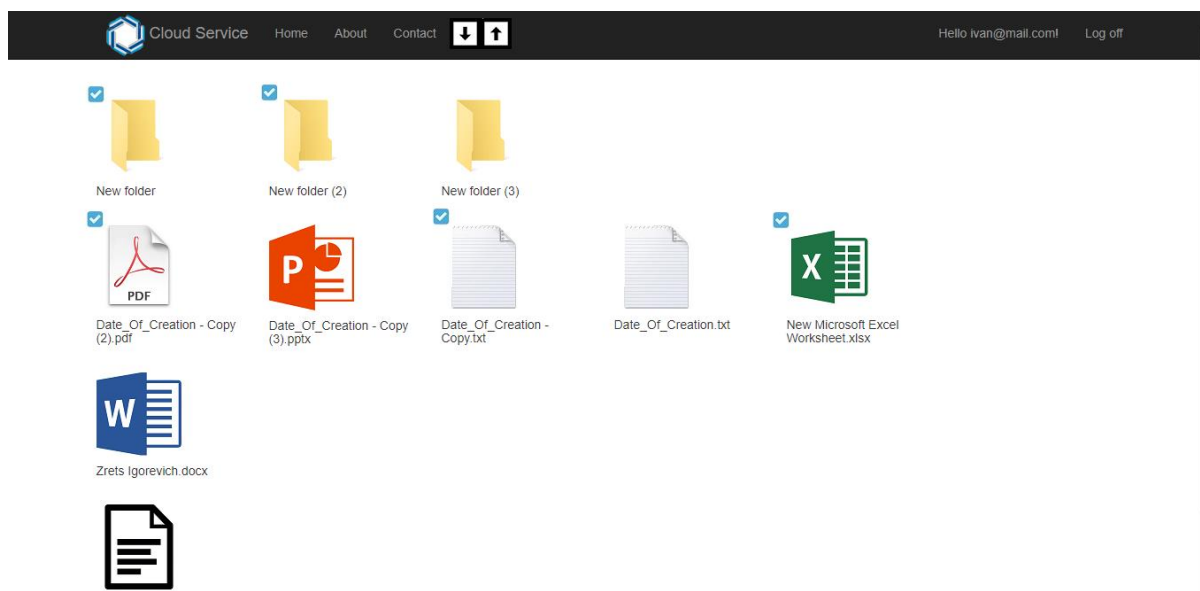


Рис. 2.6.5 – Страница со всеми личными файлами и папками.

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.

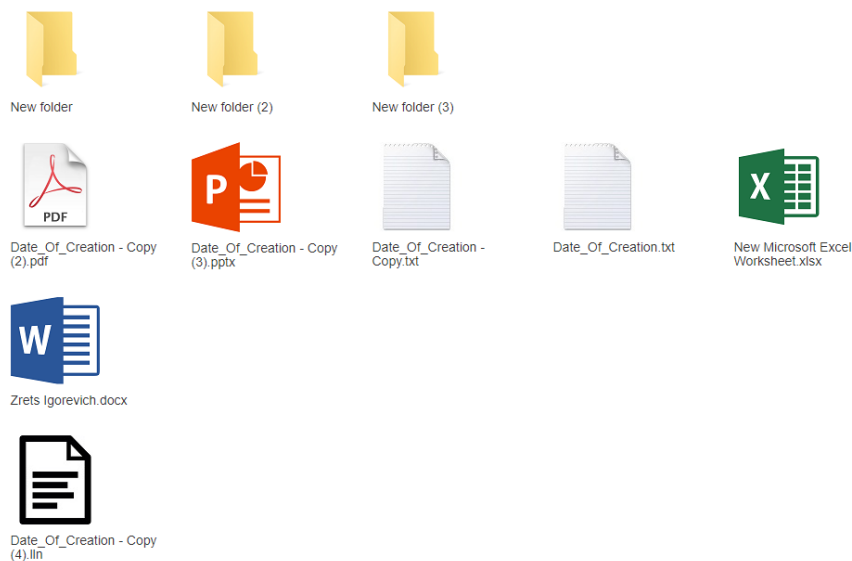


Рис. 2.6.6 – Результат работы метода DisplayFiles для контроллера FileController.

Все папки всегда располагаются в самой верхней части этой среды, а все файлы – ниже. Так же, для увеличения удобства в поиске необходимого файла/файлов, файлы, которые входят в состав пакета офисных документов Microsoft, такие как Word, Excel, PowerPoint, а также стандартные документы расширений .txt и .pdf, они будут располагаться выше, нежели файлов, нераспознанных сервисом.

Index (ManageController)

Это представление позволяет авторизованному пользователю настраивать свою учетную запись:

- Изменить логин;
- Изменить пароль;
- Добавить номер мобильного телефона;
- Включить или выключить двухфакторную авторизацию.

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

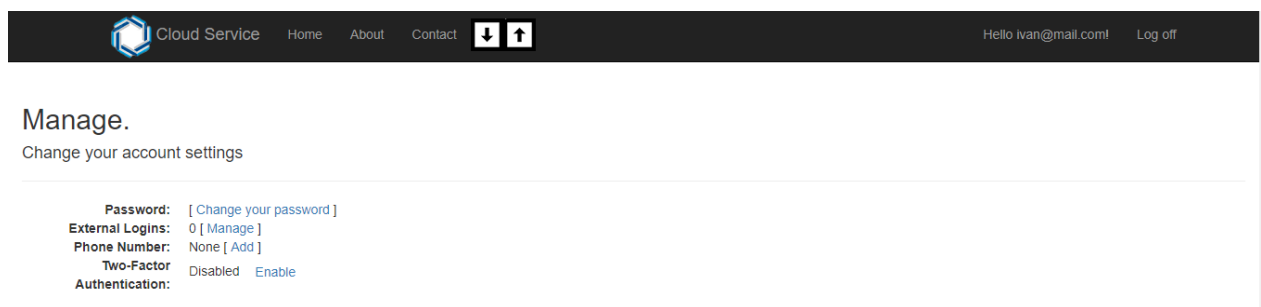


Рис. 2.6.7 – Результат работы метода Index для контроллера ManageController.

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.



Рис. 2.6.8 – 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.

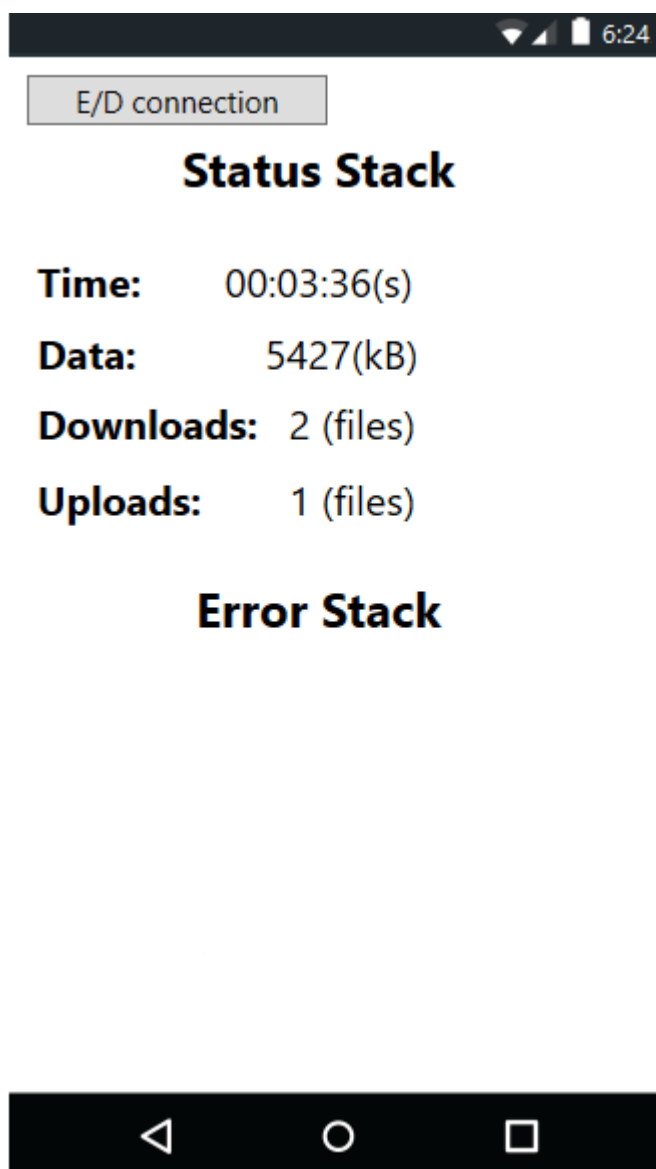


Рис. 2.6.9 – Интерфейс программы для устройства-сервера в режиме работы.

3. РАЗРАБОТКА

3.1.Разработка диаграммы классов

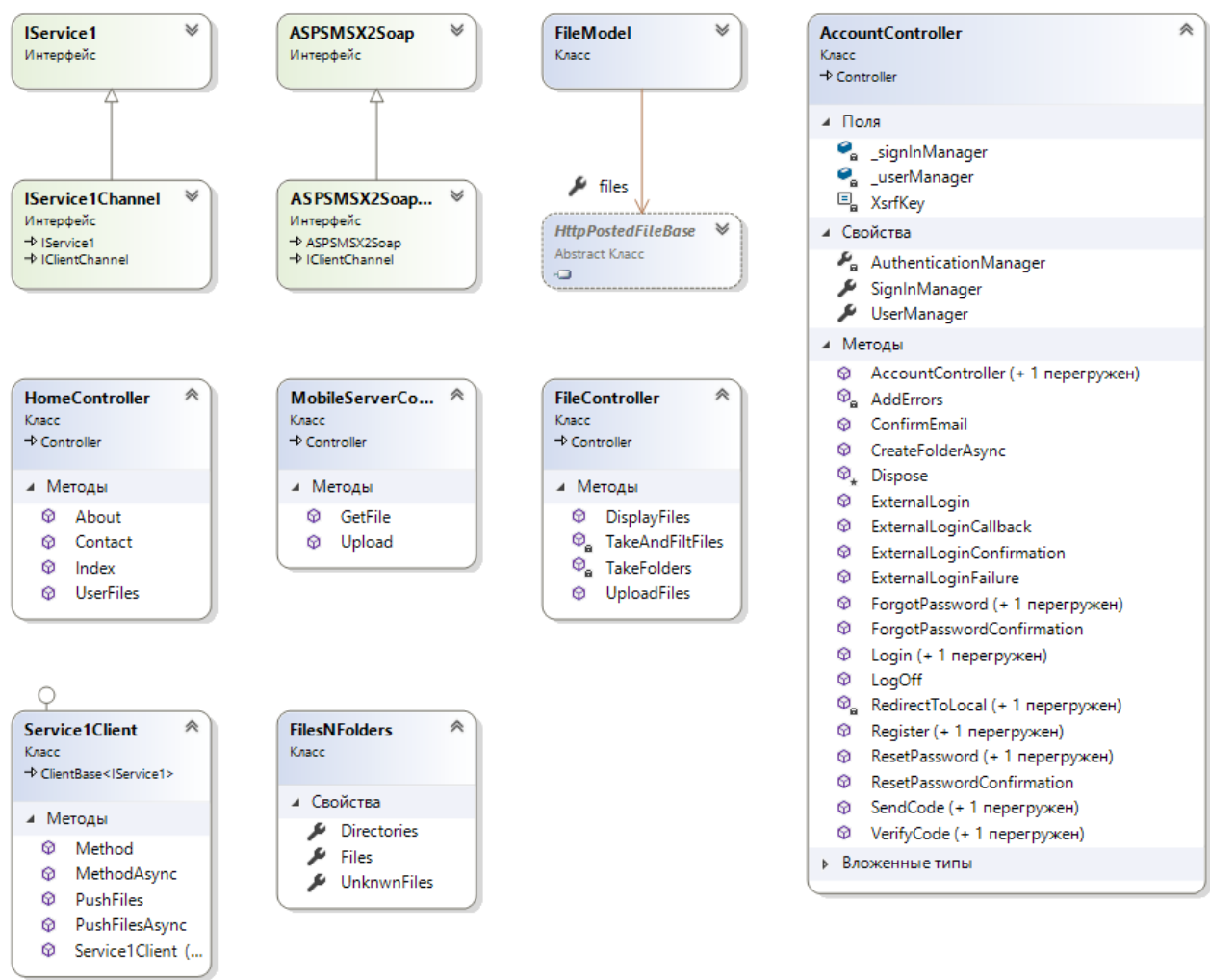


Рис. 3.1 – Классы web-приложения.

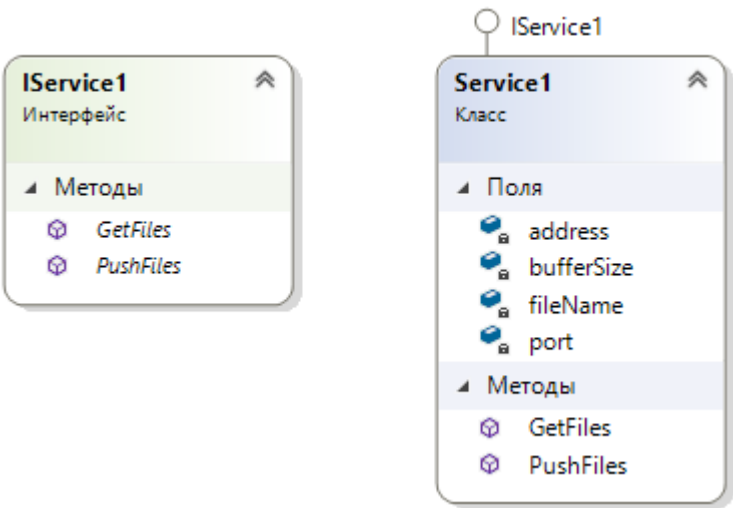


Рис. 3.2 – Классы сервиса.

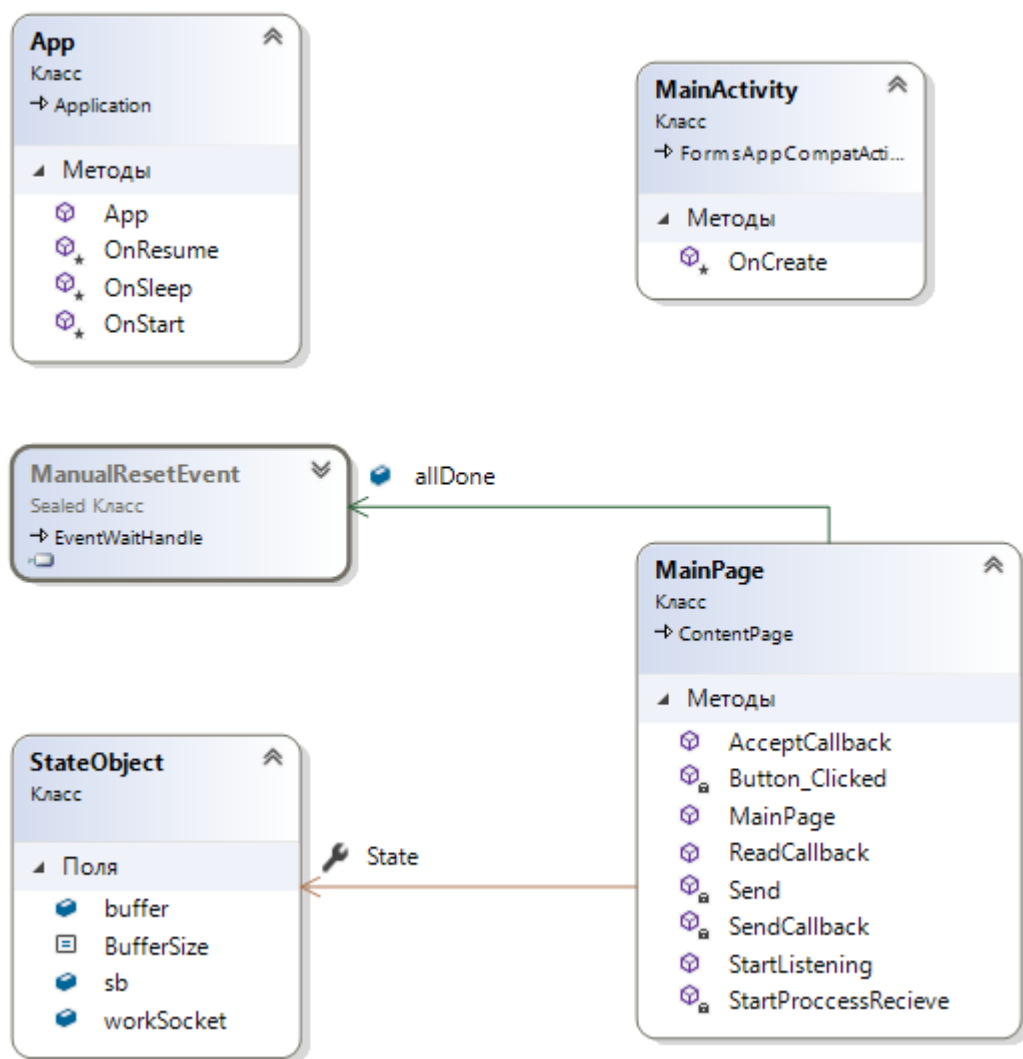


Рис. 3.3 – Диаграмма классов сервера.

3.2.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 3.2.1 – Методы HomeController

Название	Описание
Методы CostKeeping	
public ActionResult Index	Выводит страницу авторизации или регистрации
public ActionResult UserFiles	Выводит все файлы и папки пользователя

Таблица 3.2.2 – 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, вызываемый методом GET. Служит для вывода частичного представления логирования
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)	Асинхронный метод логирования, вызываемый запросом POST. Служит для получения модели данных LoginViewModel
public ActionResult Register()	Метод, вызываемый запросом GET. Служит для вывода частичного представления регистрации
public async Task<ActionResult> Register(RegisterViewModel model)	Асинхронный метод регистрации, вызываемый методом POST. Служит для получения модели данных RegisterViewModel
public Task CreateFolderAsync(string name)	Асинхронный метод для создания личной папки пользователя на сервере, вызываемый 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 3.2.3 – 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 3.2.4 – 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 3.2.5 – 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, вызываемый запросом POST. Привязывает номер мобильного телефона к аккаунту

Таблица 3.2.6 – 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

3.3.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:

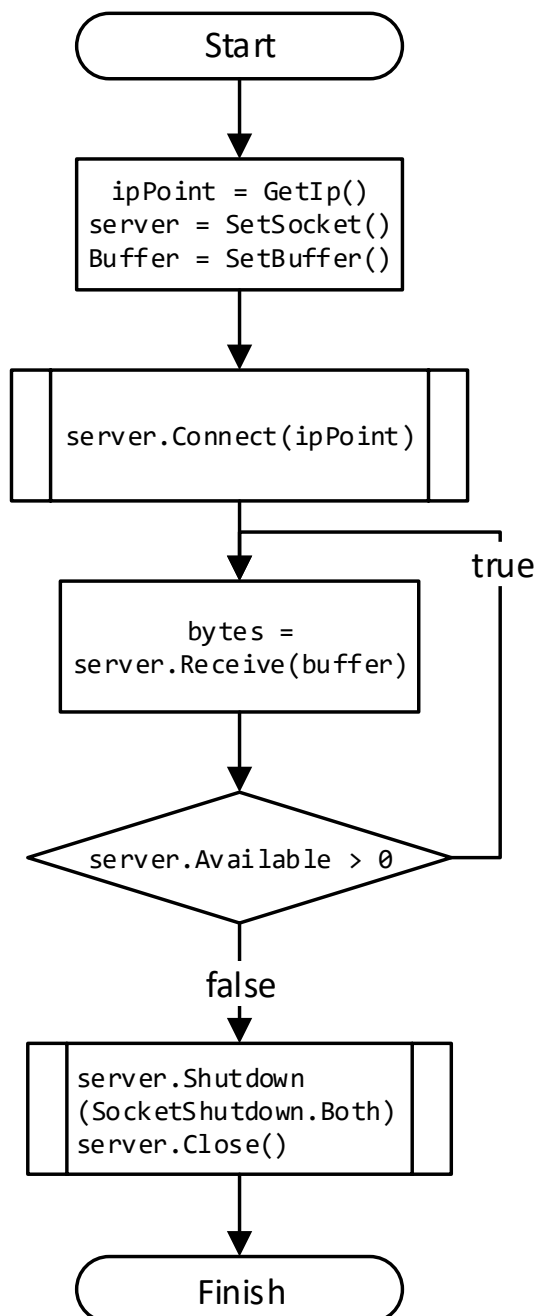


Рис. 3.3.1 – Блок-схема принципа работы метода PushFiles

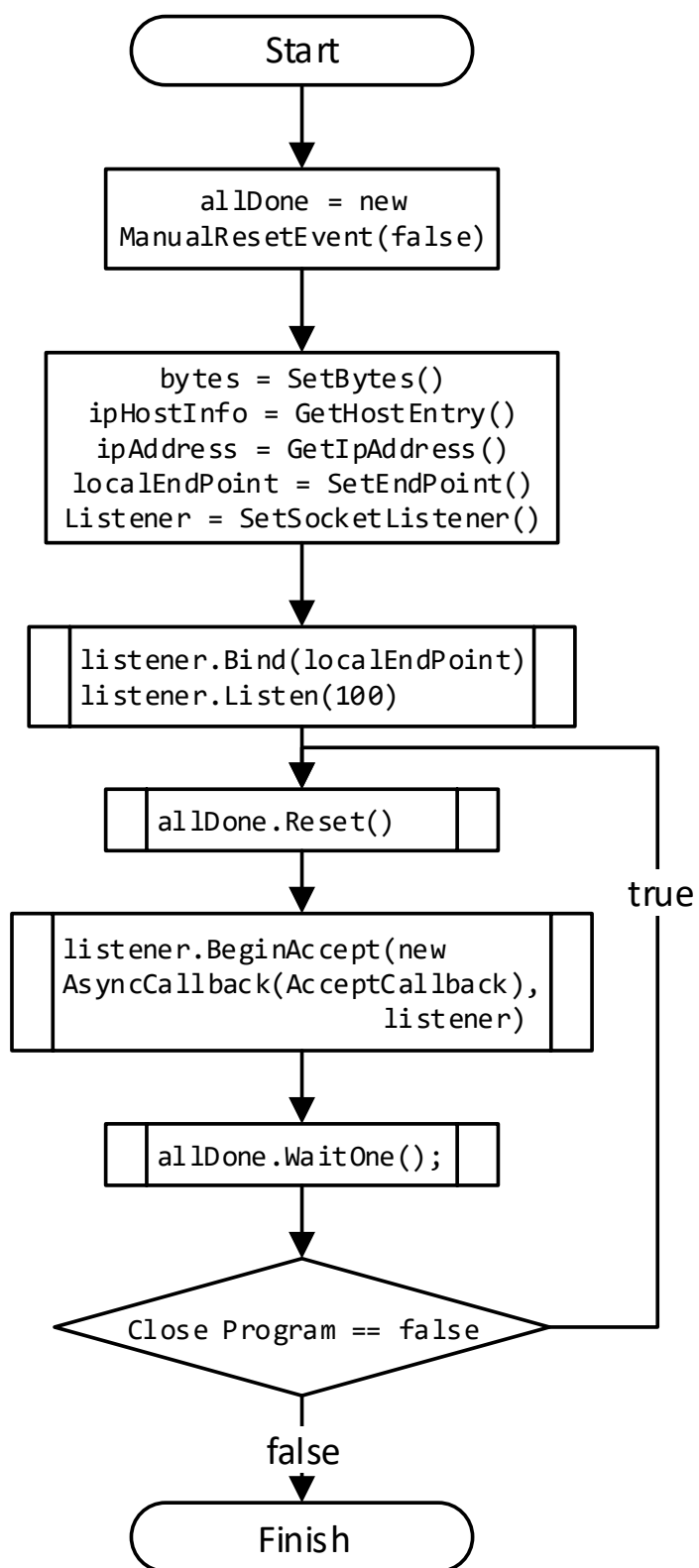


Рис. 3.3.2 – Блок-схема принципа работы метода `StartListening`.

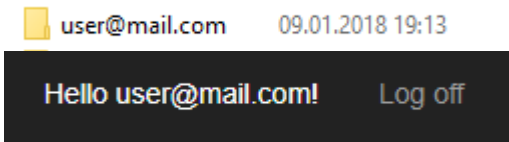
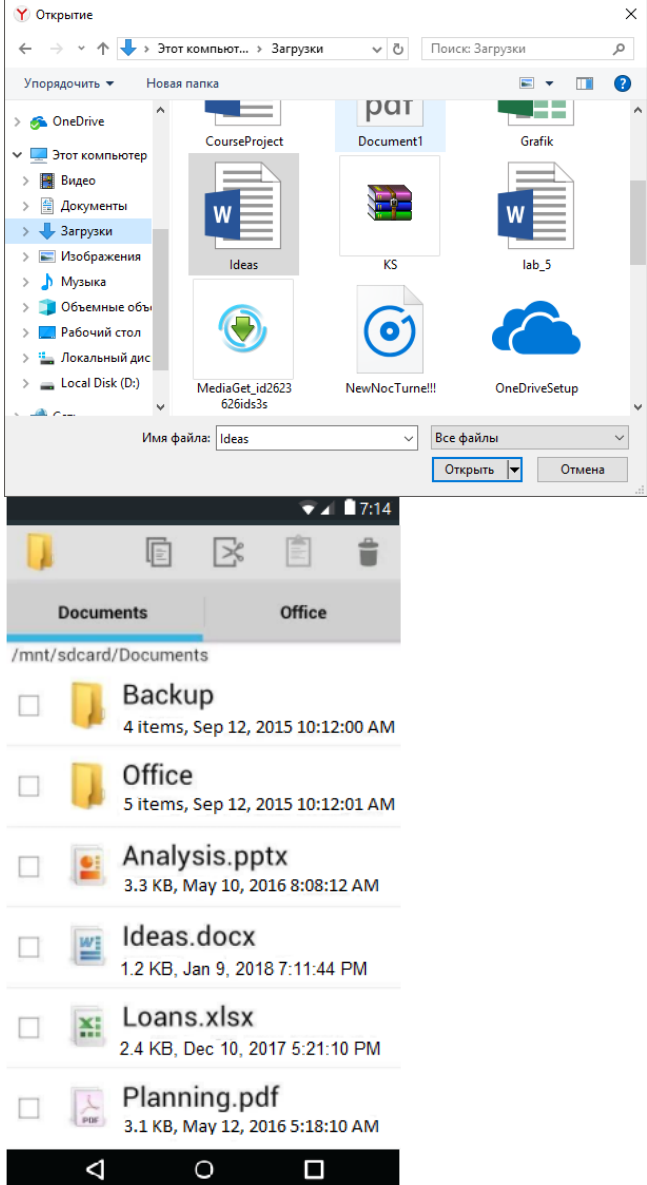
4. ВЕРИФИКАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММЫ





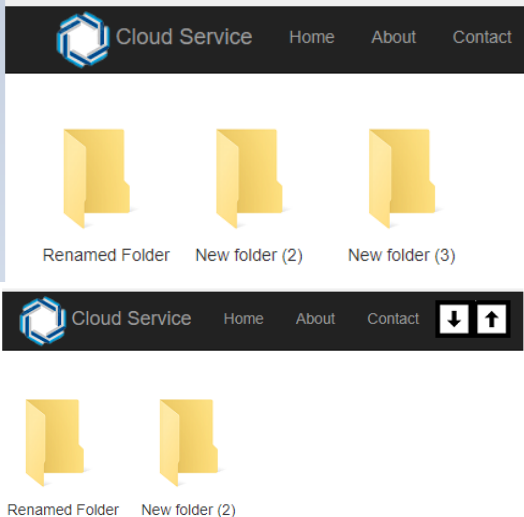
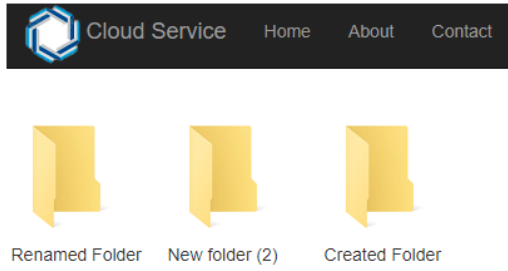
Верификация проводилась путем проверки реализации требований технического задания:

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	 New folder  Renamed Folder
Переименование файла	Файл успешно переименован	 Date_Of_Creation - Copy (2).pdf  Renamed file.pdf
Удаление папки	Папка успешно удалена	
Создание папки	Папка успешно создана	

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. Программирование на C# и .NET – METANIT.COM [Электр. ресурс] – Режим доступа: <https://metanit.com/sharp/>
2. ASP.NET MVC 5 – METANIT.COM [Электр. ресурс] – Режим доступа: <https://metanit.com/sharp/mvc5/>
3. XAMARIN – METANIT.COM [Электр. ресурс] – Режим доступа: <https://metanit.com/sharp/xamarin/>
4. MSDN [Электр. ресурс] – Режим доступа: <https://msdn.microsoft.com/ru-ru/default.aspx>
5. PROFESSORWEB [Электр. ресурс] – Режим доступа: <https://professorweb.ru/>
6. WPF 4. Подробное руководство. 4-е изд. — СПб.: Символ-Плюс, 2011. — 880 с.: ил. — (Серия «Для профессионалов»).

ПРИЛОЖЕНИЕ А Техническое задание

1 Введение

1.1 Наименование программы

Разрабатываемый комплекс программ называется «Сервис облачного хранилища файлов».

1.2 Краткая характеристика области применения

Этот сервис предназначен для хранения, редактирования файлов на удаленных серверах, а также для их обмена. Он организован по «облачному» принципу и аналогичен другим существующим на сегодняшний день сервисам, таким как “OneDrive”, “GoogleDrive”, “DropBox”, “Mega” и т. д.

2 Основание для разработки

2.1 Основание для проведения разработки

Основание для разработки Программы - задание на курсовой проект кафедры "Компьютерные системы и сети" Национального аэрокосмического университета им. Н.Е. Жуковского "ХАИ".

2.2 Наименование и условное обозначение разработки

Наименование темы разработки – «Сервис облачного хранилища файлов».

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

3 Назначение разработки

3.1 Функциональное назначение разработки

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

3.2 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 ранее созданного аккаунта. Авторизованное лицо так же может редактировать файлы форматов docx, xlsx, pptx 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.

4 Требования к программе или программному изделию

4.1 Требования к функциональным характеристикам

4.1.1 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 выше 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.

4.1.2 Требования к составу выполняемых функций

Сервис должен выполнять следующие функции:

- a) Загружать и выгружать файлы;
- b) Шифровать и дешифровать файлы в соответствии с выбранным пользователем режимом: режим повышенной надежности (алгоритм AES) и режим быстрой работы с файлами (алгоритм DES).
- c) Ограничивать размер загружаемых файлов. Если пользователь будет пытаться загрузить файл, размером свыше 5 Гб, отклонить запрос;
- d) Искать, сортировать, группировать, создавать файлы в web-приложении;
- e) Предоставлять возможность просматривать или редактировать содержимое файлов форматов docx, xlsx, pptx;
- f) Изменять параметры созданного аккаунта: почта, номер мобильного телефона, двухфакторная авторизация, имя, возраст, аватар (графическое представление пользователя в системе);
- g) Удалять учетную запись.

4.1.3 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.

4.1.4 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.

4.1.5 Требования к временным характеристикам

Требования к временным характеристикам не предъявляются.

4.1.6 Требования к безопасности

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

4.2 Требования к надежности

4.2.1 Перечень мероприятий по обеспечению надежного (устойчивого) функционирования сервиса

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

- а) организация бесперебойного питания технических средств;
- б) использование лицензионного программного обеспечения;
- в) регулярное выполнение рекомендаций Министерства труда и социального развития Украины по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств;
- г) регулярное выполнение требований ГОСТ, касающихся защиты информации путем испытания программных средств на наличие компьютерных вирусов.

4.2.2 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.

4.2.3 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 столько-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.

4.2.4 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.

4.3 Условия эксплуатации

4.3.1 Климатические условия эксплуатации

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

4.3.2 Требования к видам обслуживания

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

4.3.3 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:

- а) поддержание работоспособности технических средств;
- б) установка (инсталляция) и поддержание работоспособности системных программных средств - операционной системы;
- с) 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).

4.4 Требования к составу и параметрам технических средств

К составу технических средств должны входить:

- Сервер либо стационарный компьютер:
 - Процессор Intel Core i5 или Intel Xeon E3 1240v3 и выше;
 - Память 8 Гб DDR3 и больше;
 - Дисковая система 240 Гб SSD или HDD и больше;
 - ОС Windows 8 или Windows Server 2012 и выше;
 - IIS 8.0 и выше.
- Устройства под управлением следующих ОС на выбор: IOS, Android, Windows Phone.

4.5 Требования к информационной и программной совместимости

4.5.1 Требования к информационным структурам и методам решений

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

4.5.2 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 разработки Visual Studio 2016 и выше.

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

Для разработки серверного приложения должен использоваться должен использоваться C# 4.0 и выше, в среде разработки Xamarin Studio.

4.5.3 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.

4.5.4 Требования к защите информации и программ

Требования к защите информации и программ не предъявляются.

4.6 Специальные требования

Специальные требования не предъявляются.

5 Требования к программной документации

5.1 Предварительный состав программной документации

В результате разработки сервиса должна быть представлена следующая программная документация:

- а) техническое задание;
- б) схемы алгоритмов и данных;
- в) тексты программ;
- г) план тестирования и верификации;
- д) пояснительная записка;
- е) 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 В Руководство оператора

1. Назначение проекта

Разработанный проект может использоваться в качестве учебного примера при изучении взаимодействия ASP.NET с W3C и XAMARIN.

2. Условия работы приложений

Web-приложение:

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

Сервис:

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

Устройство сервис:

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

3. Выполнение программ

Для установки приложений выполните следующие шаги:

- a) скачать файл-архив;
- b) распаковать;
- c) убедиться в наличии платформы на конечном устройстве платформы .NET Framework с версией 4.5.2 и выше;
- d) убедиться в наличии IIS Express 10.0 и выше.

4. Возможные сообщения оператору при не правильном выполнении каких-то действий в программе

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