

Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Отчет по преддипломной практике

Выполнила студент гр. 151002
Тарасевич В. В.

Руководитель практики от
предприятия:
Колковский Д. Г.

Руководитель практики
от университета:
Шостак Е. В.

Минск 2016

СОДЕРЖАНИЕ

Введение	3
1 Анализ существующих решений и формирование требований к проектируемому программному средству	5
1.1 Требования к приложениям обмена файлами	5
1.2 Анализ существующих решений	6
1.3 Постановка задачи	14
2 Анализ требований к ПС	15
2.1 Выбор технологий проектирования	15
2.2 Выбор программной платформы	16
2.3 Выбор серверного языка программирования	19
2.4 Выбор системы контроля версий	23
2.5 Выбор среды разработки	26
2.6 Выбор системы управления реляционными базами данных	27
Список использованных источников	29

ВВЕДЕНИЕ

С момента создания глобальной сети интернет появилась необходимость в поиске и систематизации поступающей в сеть информации. Одним из возможных решений этой задачи являются сайты, предоставляющие доступ к определенным категориям данных – файловые хостинги. Пользователи этих сервисов могут искать необходимые данные среди уже загруженных на сервер, а также загружать свои собственные. Очевидно, что эффективность подобного рода сервисов довольно высока, так как пользователю не приходится тратить время на поиск информации среди различных источников.

Появление файлового хостинга с базой актуальных данных обуславливает приток большого количества пользователей, что, в свою очередь, служит необходимым фактором для пополнения базы новой информацией, а также актуализации устаревших данных. Таким образом, создаётся устойчивая саморегулируемая система.

Ключевыми компонентами файлового хостинга являются база данных, поисковая система и система учёта пользователей. В совокупности эти компоненты максимально упрощают задачу поиска информации для каждого пользователя сервиса.

Представленная дипломная работа посвящена задаче систематизации больших объемов данных. Решение этой задачи позволит существенно сократить время поиска информации, а также предоставит возможность обмена данными между пользователями.

Целью настоящей работы является эффективная реализация файлового хостинга с целью упрощения поиска необходимой пользователям информации, реализация возможности загрузки файлов на сервер хостинга, организация файлов по категориям, предоставление пользователям возможности конфигурировать параметры доступа к загруженным файлам.

Решаемые задачи:

- реализация сервиса для систематизации информации;
- анализ условий использования файловых хостингов пользователями;
- анализ имеющихся файловых хостингов и их возможностей;
- понижение уровня сложности доступа к данным;
- автоматизация систематизирующих алгоритмов;
- удобство обмена файлами между пользователями сервиса;
- повышение надежности хранения информации;

– разработка структуры и реализация файлового хостинга с использованием перспективных технологий обработки данных.

В дипломном проекте описаны существующие решения указанной задачи, проведен их сравнительный анализ с проектируемой системой. Приведено описание разработанной автоматизированной системы. Кроме того, дипломный проект включает в себя руководство пользователя и руководство программиста (для корректного сопровождения программного продукта), а также главы, в которых произведена оценка качества и экономическое обоснование эффективности применения системы, рассмотрены вопросы безопасности и экологичности.

1 АНАЛИЗ СУЩЕСТВУЮЩИХ РЕШЕНИЙ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ

В дипломном проекте поставлена задача создания веб-приложения для обмена файлами. Данное приложение должно обеспечивать хранение, загрузку и скачивание файлов. Данное приложение следует проектировать и разрабатывать как систему, способную выполнять следующие функции:

- обеспечивать хранение файлов;
- обеспечивать загрузку файлов;
- обеспечивать скачивание файлов.

1.1 Требования к приложениям обмена файлами

Требования, предъявляемые к приложениям для обмена файлами включают в себя:

- требования к функциональным характеристикам;
- требования к надежности;
- настраиваемость;
- условия эксплуатации;
- требования к информационной и программной совместимости;
- требования к составу и параметрам технических средств;
- требования к документации.

Требования к функциональным характеристикам. При выборе между объектными и структурными методами следует использовать принцип концептуальной общности, который предполагает следование единой философии на всех этапах жизненного цикла. Если предполагается использовать структурное программирование, то и на этапе анализа следует использовать структурный подход, а в случае использования объектно-ориентированных языков разработки - объектный анализ и объектное проектирование. При необходимости структурный и объектный подходы могут использоваться одновременно.

Требования к надежности – должны быть определены требования к обеспечению надежного функционирования: контроль входной и выходной информации, время и механизмы восстановления после программных и аппаратных отказов.

Настраиваемость – программное обеспечение должно обладать адаптивными возможностями, то есть указывается, должны быть предусмотрены различные изменения в методах управления и бизнес процессах.

Условия эксплуатации описывают необходимое обслуживание, которое требуется для работы системы, например, создание резервных копий, реиндексирование баз и т. п., а также требования к квалификации персонала (пользователей и обслуживающего персонала).

Требования к составу и параметрам технических средств определяют необходимый состав технических средств с указанием их основных технических характеристик. Могут указываться требования к помещениям, в которых будет находиться оборудование. Также указываются требования к переносимости системы.

Требования к информационной и программной совместимости определяют требования к информационным структурам на входе и выходе, методам решения, исходным кодам, языкам программирования и программным средствам, используемым программным комплексом.

1.2 Анализ существующих решений

1.2.1 DepositFiles

Сервис «DepositFiles» (рисунок 1.1) предоставляет функционал по загрузке и скачиванию файлов. Практически все элементы интерфейса собраны на главной странице. Кнопка «загрузить» позволяет пользователю с бесплатным аккаунтом загрузить до 50 файлов объемом до 10 гигабайт.

Возможности сервиса DepositFiles:

- бесплатное хранение файлов на серверах неограниченное время;
- скачивание файлов с медленных серверов на сервис и их хранение;
- максимально возможный размер хранимого файла — 10 гигабайт;
- суммарный объем хранимых файлов неограничен;
- бесплатное программное обеспечение для работы с файлами;
- установка пароля на скачивание файла;
- удаление файла в удобное время;
- скачивание файла в удобное время.

На страницах «гейт», «скачивания» и на странице «удаленные файлы» присутствует реклама. Сервис содержит полный набор атрибутов, необходимых для работы файлового хостинга и практически не содержит дополнительного функционала, не связанного с загрузкой пользовательских



Depositfiles - идеальное место для того, чтобы хранить Ваши ценные файлы в безопасности и делиться ими. Мы знаем, как сделать это быстро и легко. Все, что Вам необходимо - создать аккаунт и загрузить файлы.

Подпишитесь сейчас!

Почему стоит использовать DepositFiles?

- Неограниченное время хранения файлов и пространство
- Максимальный размер загружаемого файла - 10 ГБ
- Простое управление файлами
- Защита ссылки паролем
- Загрузка файлов с удаленных HTTP или FTP серверов
- Бесплатные приложения Depositfiles



Рисунок 1.1 – Главная страница сервиса по обмену файлами DepositFiles.

файлов. Таким образом, он представляет собой пример традиционного файлового хостинга и максимально упрощает навигацию пользователя за счёт простоты интерфейса.

1.2.2 4shared

Файловый хостинг «4shared.com» (рисунок 1.2) предназначен для обмена файлами между пользователями, быстрого поиска необходимых файлов и просмотра содержимого файлов в реальном времени. За счет упорядочения и систематизации большого объема информации, необходимой для обеспечения работоспособности ресурса, достигается высокая скорость поиска и загрузки файлов.

Традиционно файловые хостинги не реализуют функционал, не связанный с обработкой пользовательских данных, однако «4shared.com» предоставляет некоторые возможности по настройке пользовательского интерфейса (фон, индикаторы, статистика).

Характерными особенностями системы является возможность про-

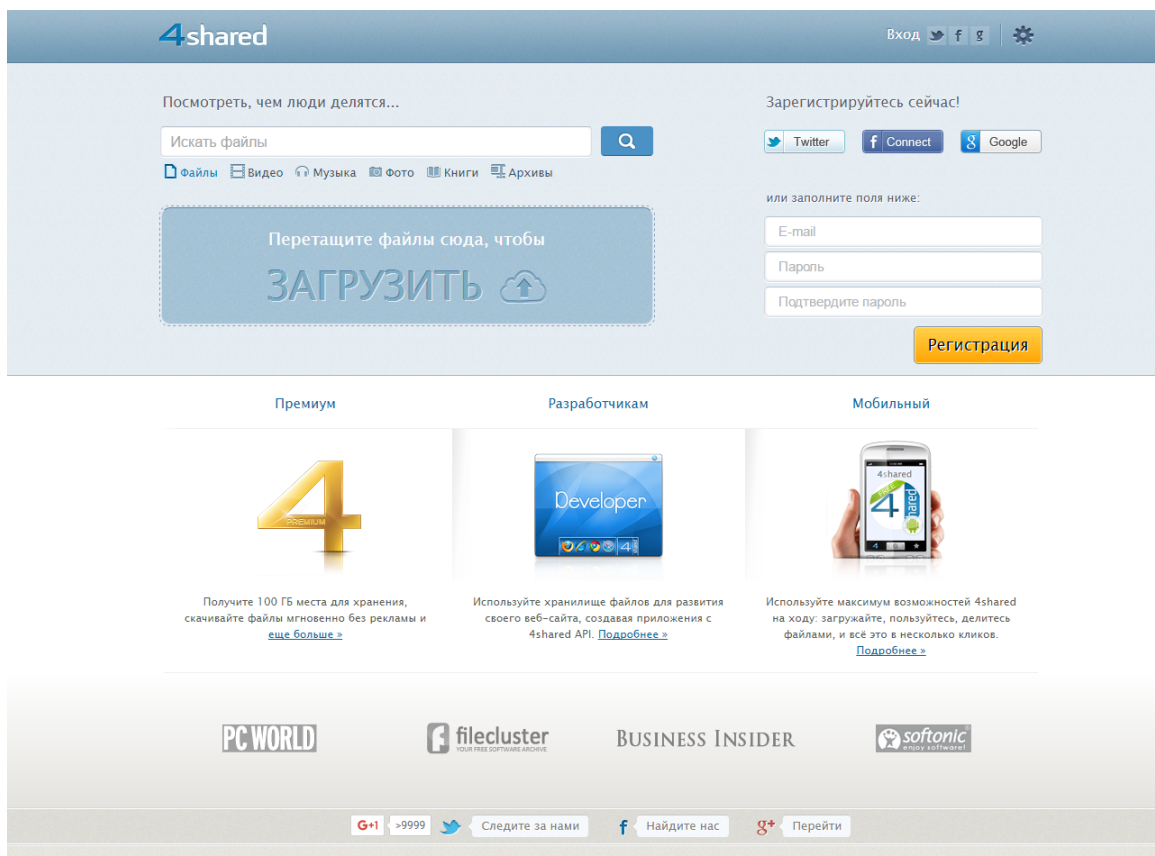


Рисунок 1.2 – Главная страница сервиса по обмену файлами 4shared.

смотря содержимого интересующих пользователя файлов, возможность установить QR-коды для каждого файла, а также автоматическая архивация файла в процессе загрузки.

Сервис «4shared.com» позволяет загружать данные любого типа размером до 100 гигабайт, обладает объёмом виртуального диска до 100 гигабайт и способен восстанавливать удалённые файлы пользователей. Большая часть дополнительного функционала доступна только владельцам премиум-аккаунтов.

Пользователю без премиум аккаунта доступен следующий функционал:

- объём виртуального диска - 15 гигабайт;
- максимальный размер загружаемого файла - 2 гигабайта;
- количество субдоменов – 100;
- лимит трафика – 0;
- функция поиска;
- управление файлами онлайн;
- windows-интерфейс;
- совместимость с операционными системами Windows, Linux и Mac;

- многоуровневая система файлов;
- восстановление файлов.

Сервис имеет простой и удобный интерфейс, позволяющий быстро найти необходимые элементы управления и получить доступ к интересующему функционалу.

1.2.3 RGhost

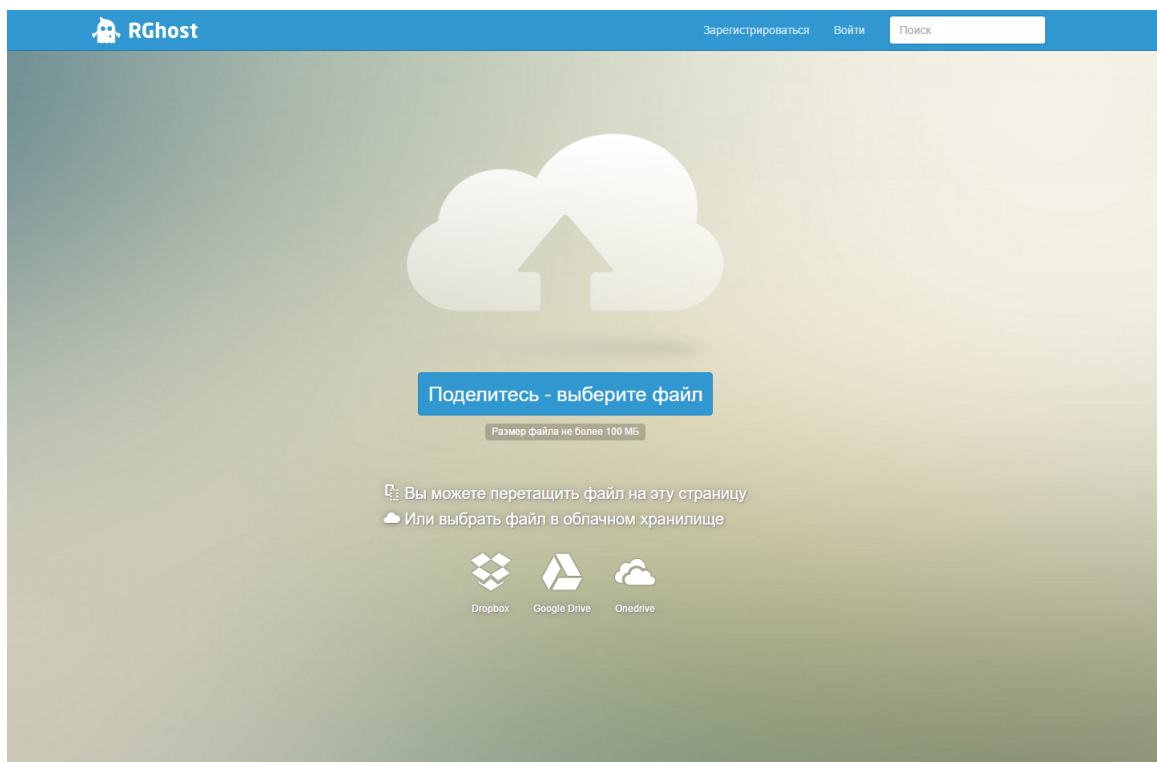


Рисунок 1.3 – Главная страница сервиса по обмену файлами RGhost.

Файловый хостинг «RGhost» (рисунок 1.6) предназначен для хранения и обмена файлами между пользователями.

Данный сервис предоставляет пользователям возможность загружать неограниченное количество файлов по суммарному объему, но размер каждого загруженного файла должен быть в пределах 50 мегабайт. Время жизни ссылок на файлы не ограничивается, но количество запросов по ссылке не должно превышать более 30 в минуту, а также данный сервис не ограничивает скорость скачивания файлов.

Пользователям доступен следующий функционал:

- неограниченный суммарный объем загруженных файлов;
- максимальный размер загружаемого файла - 50 мегабайт;
- создание бекапов меданных файла;

- установка пароля на файл;
- функция поиска;
- управление файлами онлайн;
- создание превью файла;

Сервис поддерживает большое количество вариантов оплаты премиум-аккаунта.

Счета выставляются и оплачиваются через систему OKPAY. Она поддерживает прием следующих платежных средств напрямую: OKPAY, VISA/MC RUB, Qiwi, SofortBanking, Fortumo, Money Polo, W1, OOO Pay, Alfa Click, Sberbank, PromSvyazBank, Svyaznoy, BPay.

1.2.4 My-Files.RU

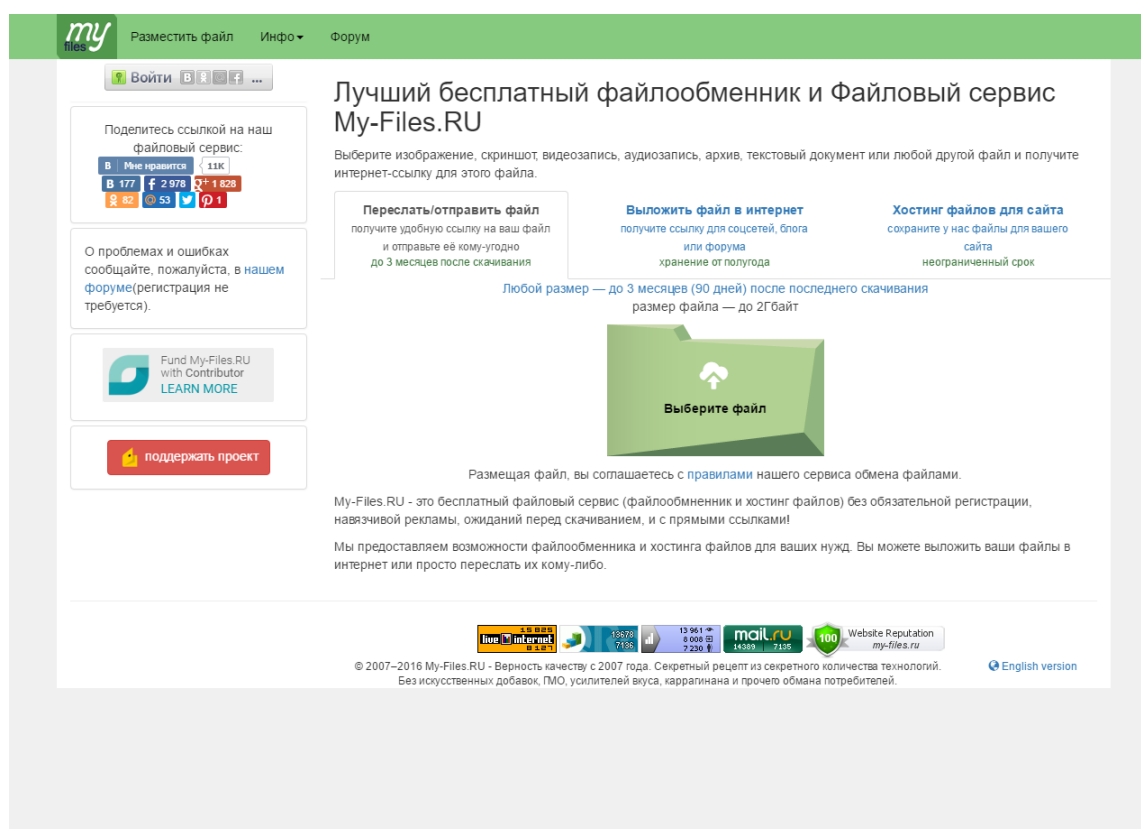


Рисунок 1.4 – Главная страница сервиса по обмену файлами My-Files.RU.

Сервис «My-Files.RU» позволяет загружать данные любого типа размером до 2 гигабайт, обладает форумом где можно отслеживать обновления и ошибки сервиса и способен восстанавливать удалённые файлы пользователей.

Данный сервис позволяет генерировать ссылки на скачивание файла в желаемом формате. Форматы представлены в виде:

- короткой ссылки на файл;
- QR-код ссылки на файл;
- кодов HTML и BBCode для установки гиперссылки на файл на веб-страницах, блогах, форумах;
- прямой ссылки на файл.

Страница загрузки файла содержит:

- информацию о файле, его размер, имя, дату размещения;
- для изображений - уменьшенное изображение по ширине страницы;
- для архивов - список содержимого архива;
- для исполняемых файлов - предупреждение о небезопасности файла.

Сервис «My-Files.RU» предоставляет возможность бесплатно размещать файлы в интернете для других пользователей.

Данный сервис подойдет для таких задач как: выложить картинки, изображения, скриншоты, фото или видео; отправить большие файлы; сохранить файлы в интернете.

Кроме того, можно получить QR-код для загруженного файла, что позволит быстро загрузить файл в мобильное устройство.

1.2.5 Файлообменник.рф

Сервис «Файлообменник.рф» - бесплатный облачный сервис для хранения файлов, куда пользователи могут загружать свои файлы, хранить их, раздавать ссылки на скачивание файлов, а также бесплатно скачивать файлы других пользователей по полученным ссылкам.

Срок хранения файлов на данном сервисе зависит от роли пользователя:

- для незарегистрированных пользователей срок хранения файлов составляет 7 дней;
- для зарегистрированных пользователей срок хранения файлов увеличивается до 45 дней с момента его последнего скачивания;
- для пользователей с Турбо доступом срок хранения составляет 90 дней с момента его последнего скачивания.

Максимальный размер загружаемого файла также зависит от роли пользователя:

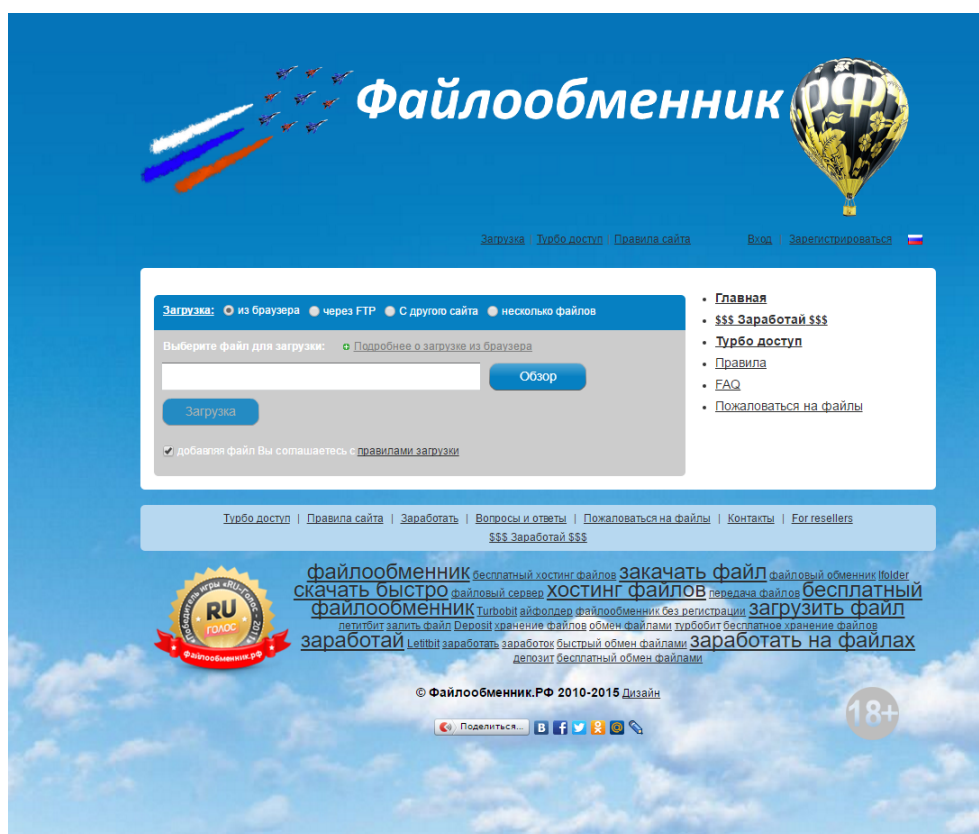


Рисунок 1.5 – Главная страница сервиса по обмену файлами
Файлообменник.рф.


- для незарегистрированных пользователей максимальный размер ограничен 200 Мб;
- для зарегистрированных пользователей максимальный размер файла 100 Гб;
- для пользователей с Турбо доступом максимальный размер файла также составляет 100 Гб.

Данный сервис также поддерживает программы для загрузки файлов такие как: Speedget, Download Accelerator Plus, GetRight, FlashGet и GoZilla, а так же google file download manager и т.д.

1.2.6 Fayloobmennik.net

При помощи сервиса «Fayloobmennik.net» можно без труда и регистрации обмениваться личными файлами, быстро загружая файлы и сохраняя на сервере.

После загрузки файла на данный сервис будет предоставлена ссылка, которую можно размещать на блогах, форумах, личных сайтах, а также отправлять пользователям сети интернет по электронной почте. Данный хо-



Загрузить свой файл

В форме приведенной ниже нажмите кнопку обзор и выберите файл который вы хотите разместить у нас. После загрузки вы получите ссылку для скачивания.

Выберите файл (MAX 2000mb): Файл не выбран

Описание:

Ваш e-mail:

Пароль к файлу:

При указании пароля файл не будет опубликован в каталоге.

☐ **Согласен с [правилами сервиса](#)**

Последние загрузки

Эп. мех. системы...rar	223.73 MB
Chrysanthemum.jpg	858.78 kB
111.rar	213.72 MB
Jellyfish.jpg	757.52 kB
Kupuna.rar	45.24 MB
Microsoft Office...exe	608.11 MB
Gothic Doom Me...ent	15.85 kB
Modern Metal...ent	33.64 kB

Новости проекта

11.10.2013 - Авария на одном из серверов

12.08.2013 - Полное снятие ограничений

06.07.2012 - DDOS атака

31.12.2011 - С новым, 2012 годом!

28.12.2011 - Игновое пополнение счета и бонусы

29.11.2011 - Изменения в правилах

15.08.2011 - Дешевле, больше, быстрее!

02.08.2011 - Технические работы

02.08.2011 - Скачать быстро без СМС

Популярные файлы

ColorOS Lenovo...zip	515.92 MB
Трофейная пьеска.exe	629.00 kB
Minecraft nupatka.exe	679.00 kB
steam_api.dll	119.12 kB
FHX-v2(A).apk	43.01 MB
PhysXLoader.dll	52.00 kB
msvcrt100.dll	808.33 kB
Foto438.jpg	326.55 kB

Популярные mp3

Bad Boy - Pul V...mp3	4.25 MB
Dilkiroi minus...mp3	4.22 MB
Bahrom Nazarov...mp3	3.24 MB
23.11.14.pamh.mp3	45.92 MB
Jivey (to'y bop...mp3	1.50 MB
ZARUBEJ - DJ MA...mp3	2.75 MB
G avrat Usmonov...mp3	3.60 MB
Sardor Mamadali...mp3	3.84 MB

Бесплатный файлообменник без регистрации, загрузить файл

При помощи сервиса fayloobmennik.net Вы без труда и регистраций сможете обмениваться личными файлами с друзьями и коллегами, быстро загружая файлы и сохраняя на сервере. После загрузки Вы получите ссылку, которую сможете размещать на блогах, форумах, личных сайтах, а также отправлять друзьям по электронной почте. Наш хостинг бесплатный, работает без каких-либо ограничений. Сразу после выбора файла начинается его загрузка, после которой Вы незамедлительно получаете рабочую ссылку. Кроме того, наш ФАЙЛООБМЕННИК уникален тем, что Вы можете в полной мере насладиться всем спектром удобных функций, не проходя при этом утомительную процедуру регистрации. Мы ломаем стереотипы - теперь для того, чтобы скачать файлы, Вам не придется пересмотреть множество рекламных баннеров, отправлять текстовые сообщения на короткий номер, или покупать «голд аккаунты» за огромные деньги. Все это в прошлом. Если контент не нарушает действующего законодательства Российской Федерации, Вы можете легко его загрузить, будь то игра, фильм, музыка, программа, или электронная книга. Если Вы устали от торрентов с их переборами, наш быстрый файлообменник станет достойной альтернативой. Загрузка на наш хостинг осуществляется напрямую через браузер и не требует установки дополнительных приложений. Все, что Вам нужно - это нажать кнопку «обзор» и выбрать интересующий файл на жестком диске и загрузить его. Если Вы хотите обратить внимание на свой материал, можете создать пост на блоге, с подробным описанием и, конечно же, фотографией. Ограничение по размеру для одного файла на нашем сервисе составляет 2000 мегабайт, однако Вы всегда можете воспользоваться архиватором (к примеру, популярным WinRar), разбить материал на тома, и загрузить его отдельными файлами. При помощи той же системы Вы можете объединить несколько небольших файлов в одном архиве для экономии времени. Если Вы цените простоту в обращении и надежность, fayloobmennik.net создан для Вас. Итак, подводя итоги Вы должны были заметить что наш сервис бесплатный, без регистрации, а самое главное это быстрый обмен файлами!

Рисунок 1.6 – Главная страница сервиса по обмену файлами Fayloobmennik.net.

стинг является бесплатным и работает без каких-либо ограничений. Сразу после выбора файла начинается его загрузка, после которой незамедлительно будет сгенерирована рабочая ссылка.

В данном сервисе есть возможность создать пост на блоге, с подробным описанием и фотографиями описывающими содержимое файла. Ограничение по размеру для одного файла на сервисе составляет 2000 мегабайт, однако можно воспользоваться архиватором (к примеру, популярным WinRar), разбить материал на тома, и загрузить его отдельными файлами. При помощи той же системы можно объединить несколько небольших файлов в одном архиве для экономии времени.

Несмотря на все преимущества рассмотренных файловых хостингов, их функциональные возможности не в полной мере соответствуют пожеланиям большинства пользователей подобных сервисов, так как требуют покупки премиум аккаунта для повышения уровня качества обслуживания, демон-

стрируют нежелательную рекламу во время загрузки файлов, тем самым загружая канал лишним трафиком, а также не имеют функционала ограничения доступа к файлу определенному кругу лиц. А также все эти сервисы не предоставляют должного уровня безопасности, который требуется при обмене файлами в корпоративной сети.

1.3 Постановка задачи

Проанализировав основные функции, а так же все достоинства и недостатки существующих веб-приложений для обмена файлами, можно выдвинуть следующие требования к разрабатываемому веб-приложению:

- регистрация и авторизация пользователя;
- загрузка и хранение файлов;
- формирование ссылок на скачивание файла;
- ограничение доступа к загруженным файлам;
- ограничение скорости на скачивание файлов;
- просмотр информации о загруженном файле
- комментирование информации о файле;
- поиск файлов;
- управление личным аккаунтом.

Входные данные : http-запрос со стартовой строкой, заголовками и телом сообщения.

Выходные данные : http-ответ с запрашиваемой информацией или содержимым файла в теле сообщения.

Средой эксплуатации должен являться веб-сервер, поддерживающий следующие технологии:

- .Net 4.5;
- MSSQL Server 2012;
- IIS 8.

2 АНАЛИЗ ТРЕБОВАНИЙ К ПС

2.1 Выбор технологий проектирования

Выбор технологий является важным предварительным этапом разработки сложных информационных систем. Платформа и язык программирования, на котором будет реализована система, заслуживает большого внимания, так как исследования показали, что выбор языка программирования влияет на производительность труда программистов и качество создаваемого ими кода [1, с. 59].

Ниже перечислены некоторые факторы, повлиявшие на выбор технологий:

- разрабатываемое ПО должно работать на операционной системе Windows 7 и более новых версиях системы;
- среди различных платформ разработки имеющийся программист лучше всего знаком с разработкой на платформе Microsoft .NET;
- имеющийся разработчик имеет опыт работы с объекто-ориентированными и с функциональными языками программирования.

Приняв во внимание необходимость обеспечения доступности дальнейшей поддержки ПО, целесообразно не использовать малоизвестные и сложные языки программирования. С учетом этого фактора выбор языков программирования сужается до четырех официально поддерживаемых Microsoft и имеющих изначальную поддержку в Visual Studio 2012: Visual C++/CLI, C#, Visual Basic .NET и F#. Необходимость использования низкоуровневых возможностей Visual C++/CLI в разрабатываемом ПО отсутствует, следовательно данный язык можно исключить из списка кандидатов. Visual Basic .NET уступает по удобству использования двум другим кандидатам из нашего списка. Использование функционального языка F# не подходит для реализации крупного веб-приложения, ввиду малой распространенности. Оставшийся язык программирования C# являются первостепенным, элегантными, мультипарадигменным языком программирования для платформы Microsoft .NET. А также для разработки клиентской части веб-приложения будет использован язык программирования JavaScript. В качестве системы контроля версий был выбрана система Git.

Таким образом, с учетом вышеперечисленных факторов, целесообразно остановить выбор на следующих технологиях:

- операционная система Windows 7;
- платформа разработки Microsoft .NET;

- языки программирования C# и JavaScript;
- система контроля версий Git

Для реализации поставленной задачи нет необходимости в использовании каких-либо прикладных библиотек для создания настольных или веб-приложений, достаточно использовать стандартные библиотеки указанных выше языков. Поддержка платформой Microsoft .NET различных языков программирования позволяет использовать язык, который наиболее просто и «красиво» позволяет решить возникающую задачу. Разрабатываемое программное обеспечение в некоторой степени использует данное преимущество платформы. Язык C# больше подходит для создания высокоуровневого дизайна приложения (иерархия классов и интерфейсов, организация пространств имен и публичного программного интерфейса), язык F# — для реализации логики приложения, функций и методов, прототипирования различных идей. Далее проводится характеристика используемых технологий и языков программирования более подробно.

2.2 Выбор программной платформы

В качестве программной платформы была выбрана программная платформа Microsoft .NET.

Программная платформа Microsoft .NET является одной из реализаций стандарта ECMA-335 [2] и является современным инструментом создания клиентских и серверных приложений для операционной системы Windows. Первая общедоступная версия .NET Framework вышла в феврале 2002 года. С тех пор платформа активно эволюционировала и на данный момент было выпущено шесть версии данного продукта. На данный момент номер последней версии .NET Framework — 4.5. Платформа Microsoft .NET была призвана решить некоторые наболевшие проблемы, скопившиеся на момент её выхода, в средствах разработки приложений под Windows. Ниже перечислены некоторые из них [3, с. XIV – XVII]:

- сложность создания надежных приложений;
- сложность развертывания и управления версиями приложений и библиотек;
- сложность создания переносимого ПО;
- отсутствие единой целевой платформы для создателей компиляторов;
- проблемы с безопасным исполнением непроверенного кода;
- великое множество различных технологий и языков программиро-

вания, которые не совместимы между собой.

Многие из этих проблем были решены. Далее более подробно рассматривается внутреннее устройство Microsoft .NET.

Основными составляющими компонентами Microsoft .NET являются общая языковая исполняющая среда (Common Language Runtime) и стандартная библиотека классов (Framework Class Library). CLR представляет из себя виртуальную машину и набор сервисов обслуживающих исполнение программ написанных для Microsoft .NET. Ниже приводится перечень задач, возлагаемых на CLR [4]:

- загрузка и исполнение управляемого кода;
- управление памятью при размещении объектов;
- изоляция памяти приложений;
- проверка безопасности кода;
- преобразование промежуточного языка в машинный код;
- доступ к расширенной информации от типов — метаданным;
- обработка исключений, включая межъязыковые исключения;
- взаимодействие между управляемым и неуправляемым кодом (в том числе и COM-объектами);
- поддержка сервисов для разработки (профилирование, отладка и т. д.).

Программы написанные для Microsoft .NET представляют из себя набор типов взаимодействующих между собой. Microsoft .NET имеет общую систему типов (Common Type System, CTS). Данная спецификация описывает определения и поведение типов создаваемых для Microsoft .NET [5]. В частности в данной спецификации описаны возможные члены типов, механизмы сокрытия реализации, правила наследования, типы-значения и ссылочные типы, особенности параметрического полиморфизма и другие возможности предоставляемые CLI. Общая языковая спецификация (Common Language Specification, CLS) — подмножество общей системы типов. Это набор конструкций и ограничений, которые являются руководством для создателей библиотек и компиляторов в среде .NET Framework. Библиотеки, построенные в соответствии с CLS, могут быть использованы из любого языка программирования, поддерживающего CLS. Языки, соответствующие CLS (к их числу относятся языки C#, Visual Basic .NET, Visual C++/CLI), могут интегрироваться друг с другом. CLS — это основа межъязыкового взаимодействия в рамках платформы Microsoft .NET [4].

Некоторые из возможностей, предоставляемых Microsoft .NET: верификация кода, расширенная информация о типах во время исполнения,

сборка мусора, безопасность типов, — невозможны без наличия подробных метаданных о типах из которых состоит исполняемая программа. Подробные метаданные о типах генерируются компиляторами и сохраняются в результирующих сборках. Сборка — это логическая группировка одного или нескольких управляемых модулей или файлов ресурсов, является минимальной единицей с точки зрения повторного использования, безопасности и управлениями версиями [5, с. 6].

Одной из особенностей Microsoft .NET, обеспечивающей переносимость программ без необходимости повторной компиляции, является представление исполняемого кода приложений на общем промежуточном языке (Common Intermediate Language, CIL). Промежуточный язык является бес типовым, стековым, объекто-ориентированным ассемблером [5, с. 16–17]. Данный язык очень удобен в качестве целевого языка для создателей компиляторов и средств автоматической проверки кода для платформы Microsoft .NET, также язык довольно удобен для чтения людьми. Наличие промежуточного языка и необходимость создания производительных программ подразумевают наличие преобразования промежуточного кода в машинный код во время исполнения программы. Одним из компонентов общей языковой исполняющей среды, выполняющим данное преобразование, является компилятор времени исполнения (Just-in-time compiler) транслирующий промежуточный язык в машинные инструкции, специфические для архитектуры компьютера на котором исполняется программа.

Ручное управление памятью всегда являлось очень кропотливой и подверженной ошибкам работой. Ошибки в управлении памятью являются одними из наиболее сложных в устранении типами программных ошибок, также эти ошибки обычно приводят к непредсказуемому поведению программы, поэтому в Microsoft .NET управление памятью происходит автоматически [5, с. 505–506]. Автоматическое управление памятью является механизмом поддержания иллюзии бесконечности памяти. Когда объект данных перестает быть нужным, занятая под него память автоматически освобождается и используется для построения новых объектов данных. Имеются различные методы реализации такого автоматического распределения памяти [6, с. 489]. В Microsoft .NET для автоматического управления памятью используется механизм сборки мусора (garbage collection). Существуют различные алгоритмы сборки мусора со своими достоинствами и недостатками. В Microsoft .NET используется алгоритм пометок (mark and sweep) в сочетании с различными оптимизациями, такими как, например, разбиение всех объектов по поколениям и использование различных куч

для больших и малых объектов.

Ниже перечислены, без приведения подробностей, некоторые важные функции исполняемые общей языковой исполняющей средой:

- обеспечение многопоточного исполнения программы;
- поддержание модели памяти, принятой в CLR;
- поддержка двоичной сериализации;
- управление вводом и выводом;
- структурная обработка исключений;
- возможность размещения исполняющей среды внутри других процессов.

Как уже упоминалось выше, большую ценностью для Microsoft .NET представляет библиотека стандартных классов — соответствующая CLS-спецификации объектно-ориентированная библиотека классов, интерфейсов и системы типов (типов-значений), которые включаются в состав платформы Microsoft .NET. Эта библиотека обеспечивает доступ к функциональным возможностям системы и предназначена служить основой при разработке .NET-приложений, компонент, элементов управления [4].

2.3 Выбор серверного языка программирования

В качестве серверного языка программирования был выбран C#.

C# — объектно-ориентированный, тип-безопасный язык программирования общего назначения. Язык создавался с целью повысить продуктивность программистов. Для достижения этой цели в языке гармонично сочетаются простота, выразительность и производительность промежуточного кода, получаемого после компиляции. Главным архитектором и идеологом языка с первой версии является Андрес Хейлсберг (создатель Turbo Pascal и архитектор Delphi). Язык C# является платформенно нейтральным, но создавался для хорошей работы с Microsoft .NET [7]. Этот язык сочетает простой синтаксис, похожий на синтаксис языков C++ и Java, и полную поддержку всех современных объектно-ориентированных концепций и подходов. В качестве ориентира при разработке языка было выбрано безопасное программирование, нацеленное на создание надежного и простого в сопровождении кода [8].

Язык имеет богатую поддержку парадигмы объекто-ориентированного программирования, включающую поддержку инкапсуляции, наследования и полиморфизма. Отличительными чертами C# с точки зрения ОО парадигмы являются:

– Унифицированная система типов. В С# сущность, содержащая данные и методы их обработки, называется типом. В С# все типы, являются ли они пользовательскими типами, или примитивами, такими как число, производны от одного базового класса.

– Классы и интерфейсы. В классической объекто-ориентированной парадигме существуют только классы. В С# дополнительно существуют и другие типы, например, интерфейсы. Интерфейс — это сущность напоминающая классы, но содержащая только определения членов. Конкретная реализация указанных членов интерфейса происходит в типах, реализующих данный интерфейс. В частности интерфейсы могут быть использованы при необходимости проведения множественного наследования (в отличие от языков С++ и Eiffel, С# не поддерживает множественное наследование классов).

– Свойства, методы и события. В чистой объекто-ориентированной парадигме все функции являются методами. В С# методы являются лишь одной из возможных разновидностей членов типа, в С# типы также могут содержать свойства, события и другие члены. Свойство — это такая разновидность функций, которая инкапсулирует часть состояния объекта. Событие — это разновидность функций, которые реагируют на изменение состояния объекта [7].

В большинстве случаев С# обеспечивает безопасность типов в том смысле, что компилятор контролирует чтобы взаимодействие с экземпляром типа происходило согласно контракту, который он определяют. Например, компилятор С# не скомпилирует код который обращается со строками, как если бы они были целыми числами. Говоря более точно, С# поддерживает статическую типизацию, в том смысле что большинство ошибок типов обнаруживаются на стадии компиляции. За соблюдение более строгих правил безопасности типов следит исполняющая среда. Статическая типизация позволяет избавиться от широкого круга ошибок, возникающих из-за ошибок типов. Она делает написание и изменение программ более предсказуемыми и надежными, кроме того, статическая типизация позволяет существовать таким средствам как автоматическое дополнение кода и его предсказуемый статический анализ. Еще одним аспектом типизации в С# является её строгость. Строгая типизация означает, что правила типизации в языке очень «сильные». Например, язык не позволяет совершать вызов метода, принимающего целые числа, передавая в него вещественное число [7]. Такие требования спасают от некоторых ошибок.

С# полагается на автоматическое управление памятью со стороны ис-

полняющей среды, предоставляя совсем немного средств для управления жизненным циклом объектов. Не смотря на это, в языке все же присутствует поддержка работы с указателями. Данная возможность предусмотрена для случаев, когда критически важна производительность приложения или необходимо обеспечить взаимодействие с неуправляемым кодом [7].

Как уже упоминалось C# не является платформенно зависимым языком. Благодаря усилиям компании Xamarin возможно писать программы на языке C# не только для операционных систем Microsoft, но и ряда других ОС. Существуют инструменты создания приложений на C# для серверных и мобильных платформ, например: iOS, Android, Linux и других.

Создатели языка C# не являются противниками привнесения в язык новых идей и возможностей, в отличии от создателей одного из конкурирующих языков. Каждая новая версия компилятора языка приносит различные полезные возможности, которые отчаются требованиям индустрии. Далее приводится краткий обзор развития языка.

Первая версия C# была похожа по своим возможностям на Java 1.4, несколько их расширяя: так, в C# имелись свойства (выглядящие в коде как поля объекта, но на деле вызывающие при обращении к ним методы класса), индексаторы (подобные свойствам, но принимающие параметр как индекс массива), события, делегаты, циклы `foreach`, структуры, передаваемые по значению, автоматическое преобразование встроенных типов в объекты при необходимости (`boxing`), атрибуты, встроенные средства взаимодействия с неуправляемым кодом (DLL, COM) и прочее [9].

Версия Microsoft .NET 2.0 привнесла много новых возможностей в сравнении с предыдущей версией, что отразилось и на языках под эту платформу. Проект спецификации C# 2.0 впервые был опубликован Microsoft в октябре 2003 года; в 2004 году выходили бета-версии (проект с кодовым названием Whidbey), C# 2.0 окончательно вышел 7 ноября 2005 года вместе с Visual Studio 2005 и Microsoft .NET 2.0. Ниже перечислены новые возможности в версии 2.0

- Частичные типы (разделение реализации класса более чем на один файл).
- Обобщённые, или параметризованные типы (`generics`). В отличие от шаблонов C++, они поддерживают некоторые дополнительные возможности и работают на уровне виртуальной машины. Вместе с тем, параметрами обобщённого типа не могут быть выражения, они не могут быть полностью или частично специализированы, не поддерживают шаблонных параметров по умолчанию, от шаблонного параметра нельзя наследоваться.

- Новая форма итератора, позволяющая создавать сопрограммы с помощью ключевого слова `yield`, подобно Python и Ruby.
- Анонимные методы, обеспечивающие функциональность замыканий.
- Оператор `??`: `return obj1 ?? obj2`; означает (в нотации C# 1.0) `return obj1 != null ? obj1 : obj2`;
- Обнуляемые (nullable) типы-значения (обозначаемые вопросительным знаком, например, `int? i = null`);, представляющие собой те же самые типы-значения, способные принимать также значение `null`. Такие типы позволяют улучшить взаимодействие с базами данных через язык SQL.
- Поддержка 64-разрядных вычислений позволяет увеличить адресное пространство и использовать 64-разрядные примитивные типы данных [9].

Третья версия языка имела одно большое нововведение — Language Integrated Query (LINQ), для реализации которого в языке дополнительно появилось множество дополнительных возможностей. Ниже приведены некоторые из них:

- Ключевые слова `select`, `from`, `where`, позволяющие делать запросы из SQL, XML, коллекций и т. п.
- Инициализацию объекта вместе с его свойствами:

```
Customer c = new Customer(); c.Name = "James"; c.Age=30;
```

можно записать как

```
Customer c = new Customer { Name = "James", Age = 30 };
```

- Лямбда-выражения:

```
listOfFoo.Where(delegate(Foo x) { return x.size > 10; });
```

теперь можно записать как

```
listOfFoo.Where(x => x.size > 10);
```

- Деревья выражений — лямбда-выражения теперь могут быть представлены в виде структуры данных, доступной для обхода во время выполнения, тем самым позволяя транслировать строго типизированные C#-выражения в другие домены (например, выражения SQL).

- Вывод типов локальной переменной: `var x = "hello"`; вместо `string x = "hello"`;
- Безымянные типы: `var x = new { Name = "James" }`;
- Методы-расширения — добавление метода в существующий класс с помощью ключевого слова `this` при первом параметре статической функции.

– Автоматические свойства: компилятор сгенерирует закрытое поле и соответствующие аксессор и мутатор для кода вида

```
public string Name { get; private set; }
```

C# 3.0 совместим с C# 2.0 по генерируемому MSIL-коду; улучшения в языке — чисто синтаксические и реализуются на этапе компиляции [9].

Visual Basic .NET 10.0 и C# 4.0 были выпущены в апреле 2010 года, одновременно с выпуском Visual Studio 2010. Новые возможности в версии 4.0:

- Возможность использования позднего связывания.
- Именованные и опциональные параметры.
- Новые возможности COM interop.
- Ковариантность и контрвариантность интерфейсов и делегатов.
- Контракты в коде (Code Contracts) [9].

В C# 5.0 было немного нововведений, но они несут большую практическую ценность. В новой версии появилась упрощенная поддержка выполнения асинхронных функций с помощью двух новых слов — `async` и `await`. Ключевым словом `async` помечаются методы и лямбда-выражения, которые внутри содержат ожидание выполнения асинхронных операций с помощью оператора `await`, который отвечает за преобразования кода метода во время компиляции.

2.4 Выбор системы контроля версий

В качестве системы контроля версий был выбрана система Git.

Система спроектирована как набор программ, специально разработанных с учётом их использования в скриптах. Это позволяет удобно создавать специализированные системы контроля версий на базе Git или пользовательские интерфейсы.

Git поддерживает быстрое разделение и слияние версий, включает инструменты для визуализации и навигации по нелинейной истории разработки.

Git предоставляет каждому разработчику локальную копию всей истории разработки, изменения копируются из одного репозитория в другой.

Ядро Git представляет собой набор утилит командной строки с параметрами. Все настройки хранятся в текстовых файлах конфигурации. Такая реализация делает Git легко портируемым на любую платформу и даёт возможность легко интегрировать Git в другие системы (в частности, создавать графические Git-клиенты с любым желаемым интерфейсом).

Репозиторий Git представляет собой каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, хранящие операции, выполняемые над репозиторием, индекс, описывающий расположение файлов и хранилище, содержащее собственно файлы. Структура хранилища файлов не отражает реальную структуру хранящегося в репозитории файлового дерева, она ориентирована на повышение скорости выполнения операций с репозиторием. Когда ядро обрабатывает команду изменения (неважно, при локальных изменениях или при получении патча от другого узла), оно создаёт в хранилище новые файлы, соответствующие новым состояниям изменённых файлов. Существенно, что никакие операции не изменяют содержимого уже существующих в хранилище файлов.

По умолчанию репозиторий хранится в подкаталоге с названием «.Git» в корневом каталоге рабочей копии дерева файлов, хранящегося в репозитории. Любое файловое дерево в системе можно превратить в репозиторий Git, отдав команду создания репозитория из корневого каталога этого дерева (или указав корневой каталог в параметрах программы). Репозиторий может быть импортирован с другого узла, доступного по сети. При импорте нового репозитория автоматически создаётся рабочая копия, соответствующая последнему зафиксированному состоянию импортируемого репозитория (то есть не копируются изменения в рабочей копии исходного узла, для которых на том узле не была выполнена команда commit).

Git использует сеть только для операций обмена с удалёнными репозиториями. Возможно использование следующих протоколов:

- Git://, открытый протокол, требующий наличие на сервере запущенного демона (поставляется вместе с Git). Протокол не имеет средств аутентификации пользователей;

- ssh:// Использует аутентификацию пользователей с помощью пар ключей, а также встроенный в UNIX-систему «основной» SSH-сервер (sshd). Со стороны сервера требуется создание аккаунтов, шеллом у которых будет некая команда Git (при разработке приложения используется именно этот протокол);

- http(s)://. Использует внутри себя инструмент curl (для Windows — поставляется вместе с Git), и его возможности HTTP-аутентификации, как и его поддержку SSL и сертификатов.

Часто называемые преимущества git перед другими DVCS:

- Высокая производительность.
- Развитые средства интеграции с другими VCS, в частности, с CVS, SVN и Mercurial. Помимо разнонаправленных конвертеров репозитория,

имеющиеся в комплекте программные средства позволяют разработчикам использовать git при размещении центрального репозитория в SVN или CVS, кроме того, git может имитировать cvs-сервер, обеспечивая работу через клиентские приложения и поддержку в средах разработки, специально не поддерживающих git.

- Продуманная система команд, позволяющая удобно встраивать git в скрипты;

- Качественный веб-интерфейс «из коробки»;

- Репозитории git могут распространяться и обновляться общесистемными файловыми утилитами архивации и обновления, такими как rsync, благодаря тому, что фиксации изменений и синхронизации не меняют существующие файлы с данными, а только добавляют новые (за исключением некоторых служебных файлов, которые могут быть автоматически обновлены с помощью имеющихся в составе системы утилит). Для раздачи репозитория по сети достаточно любого веб-сервера.

В числе недостатков git обычно называют:

- Отсутствие сквозной нумерации коммитов монотонно непрерывно возрастающими целыми числами. Во многих проектах используется автоматическое получение номера этой версии (например, командой svnversion), построение .Н файла на основе этого числа, и далее его использование при создании штампа версии исполняемого файла, некоторых вшитых в него строк и так далее;

- Отсутствие переносимой на другие операционные системы поддержки путей в кодировке Unicode в Microsoft Windows . Если путь содержит символы, отличные от ANSI, то их поддержка из командной строки требует специфических настроек, которые не гарантируют правильного отображения файловых имён при использовании тем же репозиторием из других ОС. Одним из способов решения проблемы для git 1.7 является использование специально пропатченного консольного клиента. Другой вариант — использование графических утилит, работающих напрямую через API, таких как TortoiseGit;

- Некоторое неудобство для пользователей, переходящих с других VCS. Команды git, ориентированные на наборы изменений, а не на файлы, могут вызвать недоумение у пользователей, привыкших к файл-ориентированным VCS, таким как SVN. Например, команда «add», которая в большинстве систем управления версиями производит добавление файла к проекту, в git подготавливает к фиксации сделанные в файлах изменения. При этом сохраняется не патч, описывающий изменения, а новая версия целевого фай-

ла;

- Использование для идентификации ревизий хэшей SHA1, что приводит к необходимости оперировать длинными строками вместо коротких номеров версий, как во многих других системах (хотя в командах допускается использование неполных хэш-строк);
- Большие накладные расходы при работе с проектами, в которых делаются многочисленные несвязанные между собой изменения файлов. При работе в таком режиме размеры наборов изменений становятся достаточно велики и происходит быстрый рост объёма репозитория;
- Большие затраты времени, по сравнению с файл-ориентированными системами, на формирование истории конкретного файла, истории правок конкретного пользователя, поиска изменений, относящихся к заданному месту определённого файла;
- Отсутствие отдельной команды переименования/перемещения файла, которая отображалась бы в истории как соответствующее единое действие. Существующий скрипт `gitmv` фактически выполняет переименование, копирование файла и удаление его на старом месте, что требует специального анализа для определения, что в действительности файл был просто перенесён (этот анализ выполняется автоматически командами просмотра истории). Однако, учитывая тот факт, что наличие специальной команды для переименования или перемещения файлов технически не вынуждает пользователя использовать именно её (и, как следствие, в этом случае возможны разрывы в истории), поведение `git` может считаться преимуществом;
- Система работает только с файлами и их содержимым, и не отслеживает пустые каталоги.

2.5 Выбор среды разработки

В качестве среды разработки веб-приложения был выбран программный продукт Visual Studio 2013 Ultimate Edition.

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework

и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

Visual Studio включает один или несколько компонентов из следующих:

- Visual Basic .NET, а до его появления — Visual Basic;
- Visual C++;
- Visual C#;
- Visual F#(включён начиная с Visual Studio 2010).

Visual Studio Ultimate 2013 Ultimate Edition представляет собой передовую программу, которая дает возможность любым по размеру командам осуществлять проектирование и создание привлекательных приложений. Благодаря инструментам гибкого планирования можно внедрять методы последовательной разработки и применяться гибкие методологии в темпе, удобном для пользователя.

С помощью расширенных средств моделирования, обнаружения и проектирования можно максимально полно описать систему, которая позволит наиболее удачно реализовать конкретную концепцию архитектуры.

2.6 Выбор системы управления реляционными базами данных

В качестве системы управления реляционными базами данных был выбран Microsoft SQL Server 2012.

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной исполь-

зуемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

Microsoft SQL Server является надежной базой данных для любых целей, может продолжать расширяться по мере наполнения информацией, без заметного ументшения быстродействия операций с записями в многопользовательском режиме. Пользователи могут быть добавлены путем модернизации оборудования. В последнем тесте поддерживалось до 4600 пользователей базы данных. Обеспечивается максимальная безопасность. Ваши данные защищены от несанкционированного доступа за счет интеграции сетевой безопасности с сервером безопасности. Поскольку безопасность на уровне пользователя, пользователи могут иметь ограниченный доступ к записи данных, тем самым защищая их от модификации или поиска, указав доступ на уровне пользовательских привилегией. Кроме того, с данными, хранящимися на отдельном сервере, сервер работает как шлюз, который ограничивает несанкционированный доступ.

SQL Server обрабатывает запросы от пользователей и только отправляет пользователю результаты запроса. Таким образом, минимальная информация передается по сети. Это улучшает время отклика и устраняет узкие места в сети. Это также позволяет использовать SQL Server в качестве идеальной базы данных для интернет .

Техническое обслуживание SQL Server очень простое и не требует больших знаний. Возможны изменения в структуре данных а так же резервное копирование во время работы сервера, без остановки.

Два основных языка разработки приложений используется для извлечения информации из данных SQL Server. Это C++ и Visual Basic. Эти языки являются частью Visual Studio.Net, интегрированной среды разработки Microsoft. Покупка приложений, разработанных с помощью этих продуктов гарантирует, что программное обеспечение будет модернизироваться и расширяться и развиваться в будущем.

SQL Server является приложением базы данных при работе на. Net, новейшие разработки Microsoft. Выбрав Microsoft SQL Sever в качестве базы данных информации для компании, приложение может расширяться и адаптироваться по мере изменения бизнес-климата.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Макконнелл, С. Совершенный код. Мастер-класс / Пер. с англ. / С. Макконнелл. — СПб. : Издательско-торговый дом «Русская редакция», 2005. — 896 с.
- [2] Common Language Infrastructure (CLI). Partitions I to VI. — 2012. — June. <http://www.ecma-international.org/publications/standards/Ecma-335.htm>.
- [3] Рихтер, Джеффри. CLR via C#. Программирование на платформе Microsoft .NET Framework 2.0 на языке C# / Джеффри Рихтер. — 2-е изд. — СПб. : Питер, Русская Редакция, 2007. — 656 с.
- [4] Марченко, А. Л. Основы программирования на C# 2.0 / А. Л. Марченко. — БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий — ИНТУИТ.ру, 2007. — 552 с.
- [5] Richter, Jeffrey. CLR via C# / Jeffrey Richter. Microsoft, Developer Reference. — 4-th edition. — One Microsoft Way, Redmond, Washington 98052-6399 : Microsoft Press, 2012. — 896 P.
- [6] Абельсон, Харольд. Структура и интерпретация компьютерных программ / Харольд Абельсон, Джеральд Джей Сассман, Джули Сассман. — Добросвет, 2006. — 608 с.
- [7] Albahari, Joseph. C# 5.0 in a Nutshell / Joseph Albahari, Ben Albahari. — 5-th edition. — O'Reilly Media, Inc, 2012. — June. — 1062 P.
- [8] Волосевич, А. А. Язык C# и основы платформы .NET: Учебно-метод. пособие по курсу «Избранные главы информатики» для студ. спец. I-31 03 04 «Информатика» всех форм обуч. / А. А. Волосевич. — Минск , 2006. — 60 с.
- [9] C Sharp [Электронный ресурс]. — Электронные данные. — Режим доступа: http://ru.wikipedia.org/wiki/C_Sharp. — Дата доступа: 22.03.2013.