Использование интернет-технологий для дистанционного обучения и вебинаров позволяет студентам получить доступ к контенту знаний с любой точки мира, а также делает его более персонифицированным. Успешность применения вебинаров обусловлена невысокими инвестициями, требующими для их организации. В данной работе реализована функциональная структура вебинара, позволяющая проводить занятия в режимах online и offline. Разработаны следующие функции вебинара: Video, Audio, кроссворды, презентация слайдов, проведение тренингов. Информационное обеспечение создано на основе реляционной БД и СУБД MySQL-5.5. Прикладное программное обеспечение реализовано на языках web-программирования HTML5, PHP5.3, JavaScript.

Алпысбаев Чингис Муратович, бакалавр «Техники и Технологии» специальности 5В070400 КазНТУ им. К.И.Сатпаева.Волобуева Ольга Петровна, профессор КазНТУ им. К.И.Сатпаева, академик Международной Академии Информатизации (МАИН)

978-3-659-72007-9

Чингис Алпысбаев
Ольга Волобуева

Создание приложения вебинар для изучения ПОИЯ (английский)

LAP LAMBERT Academic Publishing

Чингис Алпысбаев
Ольга Волобуева

Создание приложения вебинар для изучения ПОИЯ (английский)

Чингис Алпысбаев
Ольга Волобуева

Создание приложения вебинар для изучения ПОИЯ (английский)

LAP LAMBERT Academic Publishing

Impressum / Выходные данные
Bibliografische Information der Deutschen Nationalbibliothek: Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Библиографическая информация, изданная Немецкой Национальной Библиотекой. Немецкая Национальная Библиотека включает данную публикацию в Немецкий Книжный Каталог; с подробными библиографическими данными можно ознакомиться в Интернете по адресу http://dnb.d-nb.de.

Coverbild / Изображение на обложке предоставлено: www.ingimage.com

Verlag / Издатель:
LAP LAMBERT Academic Publishing
ist ein Imprint der / является торговой маркой
OmniScriptum GmbH & Co. KG
Heinrich-Böcking-Str. 6-8, 66121 Saarbrücken, Deutschland / Германия
Email / электронная почта: info@lap-publishing.com

Herstellung: siehe letzte Seite /
Напечатано: см. последнюю страницу
ISBN: 978-3-659-72007-9

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Обучение сегодня проводится на множественных уровнях, которые включают в себя и персональных преподавателей, и передовые интернет-технологии. Использование современных информационных технологий сделало обучение более доступным, в том числе увеличив спрос на дистанционное обучение и вебинары. Анализ рынка показывает, что многие высшие учебные заведения стремятся удовлетворить этот спрос. Обзор использования дистанционного обучения показал, что число зачисленных в высшие учебные заведения студентов вырастает в течение года на 1 000 000 человек, и что около 30% студентов колледжей и университетов проходят обучение с использованием хотя бы одного дистанционного курса. Использование технологий дистанционного обучения упрощает студентам доступ к обучению, а также делает его более персонифицированным, что объясняет рост популярности дистанционного обучения среди студентов и профессионалов, заинтересованных в дальнейшем обучении. Феноменальный рост использования технологий дистанционного обучения дает высшим учебным заведениям не только преимущества, но и порождает набор новых проблем, вызванных тем, что они привыкли проводить обучение в более привычных формах. Кроме того, появление технологий дистанционного обучения привело к появлению новых запросов и ожиданий у всех участников учебного процесса, включая студентов, преподавателей, и т.п. Учитывая эти новые вызовы, существует ряд факторов, которые должны быть учтены высшими учебными заведениями, которые планируют внедрить или улучшить проводимое ими дистанционное обучение. За последние несколько лет вебинары (webinars) вошли в число средств дистанционного обучения, пользующихся наибольшей популярностью. Вебинары используются и для проведения повышения квалификации, и для маркетинговых целей, и для обучения в рамках среднего и высшего образования. Сегодня трудно найти организацию, широко применяющую технологии дистанционного обучения и не использующую

вебинары. Такую популярность вебинары получили вследствие относительно не высокой стоимости организации и крайне высокой эффективности обучения.

В данной работе в первой главе представлен обзор функций вебинаров. Вторая глава посвящена разработке информационного обеспечения и базы данных (БД), выбору СУБД MySQL. В третьей главе представлено разработанное программное обеспечение вебинара с использованием языков PHP, HTML, JavaScript. Четвертая глава посвящена демонстрационной версии разработанного вебинара; реализованы следующие функции вебинара: слайдовая презентация, Video, VoIP (Audio), синтезатор голоса носителя языка, обсуждение презентаций, проведения тренинга, организация коллективной работы.

1 ОБЗОР СУЩЕСТВУЮЩИХ ВЕБИНАРОВ

В первые годы после появления Интернета термином «веб-конференция» часто называли ветку форума или доски объявлений. Позже термин получил значение общения именно в режиме реального времени. В настоящее время вебинар используется в рамках системы дистанционного обучения.

1.1 Представление предметной области

Online-семинар — разновидность веб-конференции, проведение online-встреч или презентаций через Интернет. Во время веб-конференции каждый из участников находится у своего компьютера, а связь между ними поддерживается через Интернет посредством загружаемого приложения, установленного на компьютере каждого участника, или через веб-приложение. В последнем случае, чтобы присоединиться к конференции, нужно просто ввести URL (адрес сайта) в окне браузера.

1.1.1 Вебинары и их функции

В некотором смысле начало истории вебинаров можно отсчитывать с момента создания первых систем текстового общения в реальном времени. Появились подобные системы в конце 1970-х годов. В середине 1990-х появились более совершенные системы общения: чаты и системы обмена мгновенными сообщениями. К концу 1990-х появились первые полноценные системы организации конференц-связи. К настоящему времени существует большое количество различных систем, позволяющих организовать общение в режиме реального времени.

Вебинары могут быть совместными и включать в себя сеансы голосований и опросов, что обеспечивает полное взаимодействие между

аудиторией и ведущим. В некоторых случаях ведущий может говорить через телефон, комментируя информацию, отображаемую на экране, а слушатели могут ему отвечать, предпочтительно по телефону с громкоговорителем. На рынке также присутствуют технологии, в которых реализована поддержка VoIP-аудио технологий, обеспечивающих полноценную аудио связь через сеть. Вебинары (в зависимости от провайдера) могут обладать функцией анонимности или «невидимости» пользователей, благодаря чему участники одной и той же конференции могут не знать о присутствии друг друга.

Среди прочих типичных функций конференц-связи [1]:

- слайдовые презентации;
- видео в режиме реального времени;
- VoIP (аудиосвязь через компьютер в режиме реального времени с использованием наушников или колонок);
- whiteboard (электронная доска для комментариев, на которой ведущий и слушатели могут оставлять пометки или комментировать пункты слайдовой презентации);
- текстовый чат — для сеансов вопросов и ответов в режиме реального времени, проводимых только для участников конференции, в чате возможно как групповое (сообщения видны всем участникам) так и приватное общение (разговор между двумя участниками);
- голосования и опросы (позволяют ведущему опрашивать аудиторию, предоставляя на выбор несколько вариантов ответов);
- удалённый рабочий стол, совместное использование приложений (когда участники могут просматривать всё, что уже было отображено на их мониторе ведущим веб-конференции; некоторые приложения совместного использования имеют функции удаленного рабочего стола, что позволяет участникам частично управлять компьютером (экраном) ведущего;
- веб-туры — когда адреса страниц, данные форм, cookies, скрипты и другая информация о сеансе может быть передана другим участникам с целью использования её для наглядного обучения с элементами входа в систему,

кликами, переходами между экранами, данный тип функций используется для демонстрации сайта или приложений при непосредственном участии пользователей;

- трансляция записи (размещается по уникальному веб-адресу, для последующего просмотра и прослушивания любым пользователем).
- проведения виртуальных лекций с возможностью взаимодействия всех участников дистанционного обучения;
- проведения кратковременных семинаров;
- выступления с докладами и защиты выполненных работ;
- презентации продуктов и услуг коммерческими компаниями;
- проведения тренингов;
- организации коллективной работы.

Наибольшее распространение вебинары получили в организациях, использующих их для организации дистанционного обучения в маркетинговых целях. Многие коммерческие компании проводят вебинары с целью демонстрации своих продуктов и услуг. Особенно большое распространение получили вебинары среди компаний, производящих высокотехнологичные продукты и услуги.

Услуга конференц-связи через сеть зачастую представляет собой сервис, расположенный на веб-сервере компании-поставщика. У каждого поставщика свои условия, однако большинство из них используют модель поминутного расчёта стоимости на пользователя или фиксированную месячную плату. Некоторые поставщики также предлагают серверные решения, которые позволяют заказчику размещать сервис конференц-связи на своём сервере. Важной функцией программ для организации конференц-связи через сеть является совместное использование приложений. Это значит, что один участник веб-конференции может передать контроль над приложением любому другому участнику [2].

1.1.2 Механизм проведения вебинара

Вебинары появились не так давно, но их использование настолько интенсивно, что уже сейчас накоплено большое количество инструментов и средств, которые могут быть использованы при проведении дистанционного обучения с использованием вебинара [1][3].

Аудио. Во время проведения вебинара основное общение между преподавателем и слушателями осуществляется посредством аудиосвязи в режиме реального времени. Также слушатели и преподаватель могут размещать аудиозаписи, предоставив к ним доступ остальным участникам обучения.

Видео. Во время обучения слушателям могут демонстрироваться видеоролики. Также существует возможность показывать видео в режиме реального времени, используя веб-камеру или цифровую камеру. Видео может транслироваться, как от преподавателя к слушателям, так и в обратную сторону. Очень часто во время вебинара все слушатели видят преподавателя.

Презентации. При проведении вебинара преподаватель может демонстрировать слушателям на их персональных компьютерах слайды презентации, осуществляя управление презентацией в режиме реального времени.

Демонстрация документов. При проведении вебинара преподаватель может демонстрировать слушателям на их персональных компьютерах различные документы, выделяя в них области которым необходимо уделить особенное внимание.

Обмен файлами. Во время обучения преподаватель и слушатели могут обмениваться или предоставлять доступ к своим файлам.

Электронная доска. Виртуальной аналог учебной доски в классе. Как и в классе на доске можно рисовать, стирать и т.д. Оставлять записи на доске могут все участники обучения в соответствии с существующими у них правами.

Демонстрация рабочего стола. Во время вебинара можно демонстрировать всем участникам свой рабочий стол Windows, показывая действия, которые ты совершаешь.

Чат. Эффективным средством организации взаимодействия слушателей вебинара является чат, с помощью которого они могут обмениваться мгновенными сообщениями в режиме реального времени. Сообщения могут быть доступны всем слушателям дистанционного обучения, а могут быть доступны только определенному кругу лиц. Границы видимости сообщений определяются преподавателем и слушателями вебинара.

Голосования и опросы. Крайне эффективным средством проведения вебинара являются голосования и опросы, позволяющие в реальном времени собрать информацию от слушателей по тому или иному вопросу.

Удаленный рабочий стол. Многие программные продукты, использующиеся при проведении вебинара, предоставляют участникам возможность манипулировать объектами на компьютере другого пользователя. Это может быть полезным, когда необходимо что-то показать слушателю вебинара. Такой возможностью обладают и слушатели, и преподаватели.

Совместное использование приложений. Как и в случае с удаленным рабочим столом цель данного средства предоставить слушателю вебинара и преподавателю возможность манипуляции программным обеспечением, запущенным на другом компьютере. Отличием от удаленного рабочего стола является то, что в первом случае пользователь получает полный контроль над компьютером другого пользователя, а во втором только над программой, к которой ему предоставлен доступ.

Поддержка мобильных устройств. Многие из программных продуктов и сервисов, предназначенных для проведения вебинаров, поддерживают большинство существующих на сегодня мобильных устройств.

Запись вебинаров. Пользователям вебинаров предоставляется возможность записи вебинаров в которых они участвуют, чтобы в последствии они могли их повторно просмотреть.

Интеграция с другими информационными системами. Программное обеспечение и сервисы, предназначенные для проведения вебинаров, часто предоставляют пользователям возможность интеграции вебинаров в сайт или интранет. Это позволяет организовать вебинар таким образом, чтобы пользователи оставались в привычном им пространстве сайта или портала.

Технологии конференц-связи через Интернет не стандартизированы, что отрицательно сказывается на функциональной совместимости, зависимости от платформы, вопросах безопасности, цене и сегментации на рынке. В 2003 году IETF учредила рабочую группу под названием «Centralized Conferencing» (XCON) для установления стандартов конференц-связи. Среди запланированных целей XCON значатся [3]:

- базовый протокол «floor control» — Binary Floor Control Protocol (BFCP), сформулированный в RFC 4542;
- механизм контроля членства и авторизации;
- механизм управления совмещением различных типов медиафайлов (аудио, видео, текстовых) и его описание;
- механизм извещения об относящихся к конференции событиях/изменениях (например смена протокола).

1.1.2 Организация вебинара

Вебинар довольно сложное с организационной точки зрения мероприятие. Его проведение требует хорошей подготовки и досконально отработанного проведения. Возникновение сбоев во время проведения вебинара приведет к снижению качества обучения и отрицательному впечатлению слушателей. Обучение, проводимое с использованием вебинаров, имеет свои характерные особенности, которые обязательно необходимо учитывать при проведении обучения. К таким характерным особенностям следует отнести [5]:

– сложность контроля поведения слушателя дистанционного обучения, вплоть до определения его присутствия на вебинаре (может включить компьютер и уйти);

– отсутствие непосредственного визуального контакта между преподавателем и слушателями;

– недостаточная подготовка слушателей для пользования программным обеспечением с помощью которого проводится вебинар;

– высокие требования к технической инфраструктуре (в первую очередь к пропускной способности каналов передачи данных);

– большее время, требующееся на взаимодействие по сравнению с традиционным очным обучением.

Вебинар требует довольно сложной технической подготовки. Необходимо развернуть и подготовить необходимое программное и аппаратное обеспечение, а также обеспечить наличие необходимых каналов связи. Чем сложнее средства обучения используются при проведении вебинара, тем выше должна быть пропускная способностью канала связи. При выборе технологической платформы или сервиса следует особое внимание уделить следующим аспектам [5]:

– каково качество звука и изображения при проведении вебинара на базе выбранной платформы или сервиса;

– какие мобильные устройства поддерживает;

– насколько эргономичен интерфейс;

– не возникает ли проблем при значительном увеличении количества участников вебинара.

Также особого внимания требует разработка учебного контента, который специфичен. Особенно в случае если преподаватель не имеет большого опыта их проведения. Крайне полезно перед принятием решения о дате проведения вебинара провести опрос участников с целью определения оптимальной даты и времени проведения вебинара. При рассылке уведомлений о проведении вебинара целесообразно использовать механизмы, позволяющие получателям

легко перенести информацию о вебинаре в свой электронный календарь. В обязательном порядке в информационное письмо должна быть включена следующая информация [5]:

- цели и задачи вебинара;
- программа вебинара;
- какую пользу принесет вебинар слушателям;
- имена преподавателей, модераторов и т.п., которые будут участвовать в вебинаре;
- место, время, продолжительность вебинара.

Необходимо также уделить максимальное внимание тому, в каком состоянии находится рабочее место. Стоит все привести в порядок, чтобы картинка, которую увидят слушатели вебинара не вызывала у них раздражения. Микрофон следует закрепить таким образом, чтобы руки оставались свободными, так как во время проведения вебинара может возникнуть необходимость выполнения каких-либо действий за компьютером.

Непосредственно проведение вебинара также требует большого внимания. Следует придерживаться следующих основных принципов [6]:

- оптимальным временем проведения вебинара является 3 - 1 часов утра;
- преподаватель должен быть доступен в рамках вебинара не менее чем за 3 минут до его начала. Его присутствие должно быть очевидным для подключающихся пользователей;
- необходимо проверить все средства проведения вебинара. В том числе, как слушатели слышат, отражается ли у них презентация, могут ли они пользоваться чатом и т.д.;
- во время проведения вебинара преподаватель должен иметь одного или нескольких помощников, решающих технические и организационные вопросы;

– выступление преподавателя не должно быть монотонным, оно должно прерываться обсуждениями в которых должны участвовать слушатели вебинара;

– повторите всем участникам вебинара правила участия в вебинаре и его основные возможности;

– ознакомьте всех с планом вебинара.

Обязательно нужно иметь несколько сценариев проведения вебинара с тем, чтобы при возникновении непредвиденных ситуаций оперативно среагировать на них и продолжить обучение.

После завершения вебинара обязательно необходимо разослать его участникам информационные письма, в которых поблагодарить их за участие, напомнить о ключевых пунктах презентации (не более трех). Обязательно включить в текст письма гиперссылки, которые использовались во время вебинара. Также целесообразно разместить в сети Интернет все материалы вебинара и предоставить им доступ к этим материалам.

Полезно после завершения вебинара просмотреть его запись с целью выявления проблем и коррекции регламентов проведения следующих вебинаров. Если проводиться большое количество вебинаров следует проводить их выборочный мониторинг.

1.1.3 Преимущества использования вебинаров при проведении обучения

Использование при проведении обучения вебинаров позволяет *организаторам* обучения получить следующие преимущества [5]:

– минимальные затраты на подготовку мероприятия;

– низкая себестоимость по сравнению с другими формами обучения;

– высокое качество обучения;

– возможность обучения большого количества слушателей.

Использование при проведении обучения вебинаров позволяет *слушателям* получить следующие преимущества:

– стоимость обучения на вебинарах гораздо дешевле по сравнению с другими формами обучения;

– вебинары позволяют слушателям сэкономить время, которые они затрачивают на обучение;

– слушатель легко может получить доступ к качественному обучению. Стать слушателем вебинара сегодня можно практически из любой точки мира.

1.1.4 Проблемы, возникающие при проведении вебинаров

Эффективность проведения вебинаров во многом зависит от качества подготовки к ним. Необходимо быть готовым и принимать превентивным меры по отношению к наиболее распространенным проблемам, возникающим при проведении вебинаров. Список наиболее часто возникающих проблем представлен ниже [8].

Технические неполадки. Проведение вебинара является довольно сложным технологическим процессом. Сбои могут произойти на сервере, на рабочем месте пользователя, у пользователя. Могут возникнуть проблемы с каналами передачи данных. Вследствие этого необходимо очень внимательно подходить к качеству сервиса, обеспечивающего проведение вебинара.

Потеря контакта со слушателями. Преподаватель не видит реакции слушателей вебинара, из-за чего может быть потерян контакт с аудиторией. Целесообразно иметь помощников, которые подскажут, что что-то идет не так.

Неготовность слушателей. Слушатели не имеют достаточных навыков для участия в вебинаре. Поэтому выполнение многих действий может занимать большое количество времени, что не приемлемо, особенно в случае, если в вебинаре принимает участие большое количество человек. Данная проблема решается двумя путями:

1. наличием у преподавателя помощников, которые помогут пользователям;

2. обеспечением хорошей подготовки участников к проведению вебинаров (наличие хорошей инструкции, видео уроков по участию в вебинаре и т.д.).

1.2 Технология масштабируемого видеокодирования

Одним из главных движителей рынка видеоконференцсвязи (ВКС) сегодня является снижение стоимости подключения одной точки. Во многом это связано с переходом к алгоритмам масштабируемого кодирования (SVC), которые позволяют избавить серверы от ресурсоемкой процедуры перекодирования потоков. SVC (Scalable Video Coding) – это технология масштабируемого видеокодирования, позволяющая передавать в одном потоке несколько подпотоков видео различного качества. Обычно это два подпотока - базовый и вспомогательный [6]. Базовый подпоток передается в стандартном качестве, а вспомогательный – в улучшенном, например, с большей частотой кадров или с большим разрешением видео.

Технология SVC позволяет серверу видеоконференций подстраивать видеопоток под изменяющиеся характеристики терминалов участников, такие, как процессорные ресурсы и ширина канала связи. Сервер назначает устройствам, какой из потоков декодировать: пользователи с большой шириной канала связи будут декодировать полный поток, а слабым каналам или устройствам (мобильные телефоны, планшеты) достанется только базовый поток с меньшей скоростью передачи данных. Таким образом, устраняется недостаток влияния слабого участника конференции.

Без использования SVC все участники групповой видеоконференции получали бы видео такого качества, которое удовлетворяло терминал с самыми слабыми характеристиками. Теперь же пользователи многоточечной видеоконференцсвязи будут видеть картинку в том качестве, в котором позволяет их оборудование и каналы связи.

Суть SVC сводится к послойному кодированию видео на терминале, как показано на рисунке 1 – каждый слой повышает разрешение картинки, поэтому при передаче видео на другой терминал серверу не надо ничего перекодировать – достаточно выбрать столько слоев (потоков), чтобы разрешение соответствовало характеристиками этого терминала и канала связи с ним. При такой схеме сервер MCU (Multipoint Control Unit - аппаратно-программное устройство, предназначенное для объединения аудио- и видеоконференции в многоточечный режим) выполняет функции некоего коммутатора видеопотоков (он становится не транскодирующим, а переключающим MCU), ресурсов на такие операции требуется значительно меньше, следовательно, сервер просто реализовать на стандартных процессорах и стоить он будет значительно меньше [6].

SD (Standard Definition, иногда также расшифровывается как Standard Digital) — разновидность телевизионных вещательных стандартов, параметры которых выбраны, исходя из расстояния наблюдения, равного шести высотам наблюдаемого изображения

HD (High Definition) — система телевидения, с разрешающей способностью по вертикали и горизонтали, увеличенной примерно вдвое по сравнению со стандартной

CIF (Common Intermediate Format) — это формат, используемый для стандартизации вертикального и горизонтального разрешения в пикселях - последовательностях в видеосигнале.

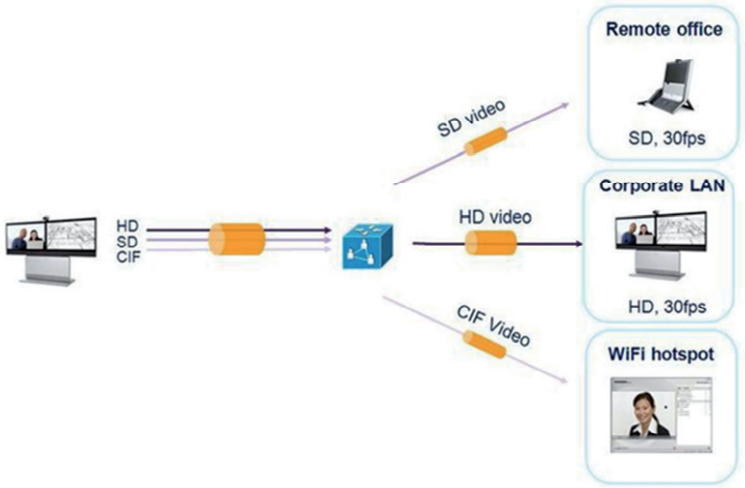

Рисунок 1 – Послойное видеокодирование

Различают три основных типа масштабирования видеопотоков – временное, пространственное и качественное.

Временное масштабирование (измененная частота кадров) — передается вспомогательный поток с большей частотой кадров. Этот тип позволяет декодерам пропускать часть или все кадры из вспомогательного потока.

Пространственное масштабирование (измененный размер изображения) – передается вспомогательный поток с большим разрешением видео. Декодер может переключаться между различными разрешениями.

Качественное масштабирование (измененное качество изображения) – вспомогательный поток закодирован с большим качеством, при отсутствии вспомогательного потока, декодер использует только базовый. Кроме этих типов масштабирования существует комбинированный тип, использующий комбинацию из перечисленных выше, что дает еще большую возможность серверу варьировать качеством видео и скоростью передаваемого потока.

Поддержка технологии масштабируемого видеокодирования дает пользователю массу преимуществ, ведь с использованием SVC участники групповой видеоконференции будут получать изображение именно в том качестве, котором позволяет их оборудование и каналы связи. При использовании SVC сохраняется мультиплатформенность, что позволяет работать на любой операционной системы и а всех устройствах, что показано на рисунке 2. Тогда как без использования данной технологии всем пользователям пришлось бы довольствоваться тем полученным видео, которое удовлетворяло бы терминальное устройство с самыми слабыми характеристиками [6].

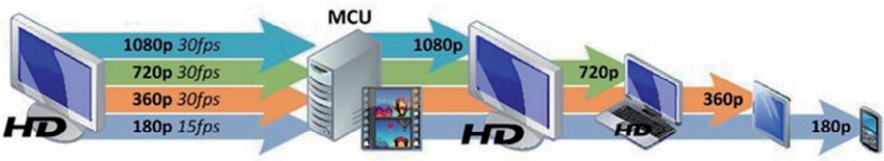

Рисунок 2 – Совместимость с устройствами и операционными системами

1.3 Предметная область изучения языка

В данной работе показана разработка приложения вебинар для изучения профессионально-ориентированного иностранного языка английский. Профессиональный английский язык является *необходимым, но не достаточным для формирования комплекса знаний, умений и навыков*, позволяющих использовать методы управления проектом при исследовании и проектировании информационных систем в различных предметных областях *венчурных предприятий*. Тем не менее, хорошие знания разговорного и письменного английского языка позволят студентам по окончании

университета работать в международных совместных предприятиях. Это важно особенно в нынешних условиях, когда обучение студентов будет проводиться в условиях передачи части государственных университетов в доверительное управление консорциумов иностранных и отечественных инвесторов.

Приведем некоторые примеры проектов – как деловых, так и личных: строительство моста; начало новой рекламной компании; переезд в новый дом; перенос данных на новый сервер; разработка брошюры о новой услуге; поиск пособия на обучения ребенка в университете; организация семинара; разработка нового оборудования; разработка или внедрение программных средств; реорганизация компании и т.д. Все эти виды деятельности имеют между собой целый ряд *общих признаков*, делающих их проектами и отличающих от других видов деятельности [9]:

– направлены на достижение конкретных целей;

– включают в себя координированное выполнение взаимосвязанных действий;

– имеют ограниченную протяженность во времени, с определенным началом и концом;

– все они в определенной степени неповторимы и уникальны.

Направленность на достижение целей. Проекты нацелены на получение определенных результатов, т.е. они направлены на достижение целей. Эти цели являются движущей силой проекта, и все усилия по его планированию и реализации предпринимаются для того, чтобы эти цели были достигнуты. Проект обычно предполагает целый комплекс взаимосвязанных целей. Важной чертой управления проектами является точное определение и формулирование целей, начиная с высшего уровня, а затем постепенно опускаясь до детализированния целей и задач.

Координированное выполнение взаимосвязанных действий. Проекты включают в себя выполнение многочисленных взаимосвязанных действий. В отдельных случаях эти взаимосвязи достаточно очевидны (например, технологические зависимости), в других случаях они имеют более тонкую

природу. Некоторые промежуточные задания не могут быть реализованы, пока не завершены другие задания; некоторые задания могут осуществляться только параллельно, и так далее. Если нарушается синхронизация выполнения разных заданий, весь проект может быть поставлен под угрозу. Проект – это система, то есть целое, складывающееся из взаимосвязанных частей, причем *система динамическая*, и, следовательно, требующая особых подходов к управлению.

Ограниченная протяженность во времени. Проекты выполняются в течение конечного периода времени. Они временны; у них есть четко выраженные начало и конец. Проект заканчивается, когда достигнуты его основные цели. Значительная часть усилий при работе с проектом направлена именно на обеспечение того, чтобы проект был завершен в намеченное время. Для этого готовятся графики, показывающие время начала и окончания заданий, входящих в проект. Проект как система деятельности существует ровно столько времени, сколько его требуется для получения конечного результата.

Уникальность. Проекты – мероприятия в известной степени неповторимые и однократные. Вместе с тем, степень уникальности может сильно отличаться от одного проекта к другому.

Проект – это ограниченное по времени целенаправленное изменение отдельной системы с установленными требованиями к качеству результатов, возможными рамками расхода средств и ресурсов и специфической организацией [7].

Включение в определение «отдельной системы» указывает не только на целостность проекта и его разграниченность с другими предприятиями, но и подчеркивает единственность проекта (в отличие от серийного производства), а значит – его неповторимость и признаки новизны.

Многообразие проектов, с которыми приходится сталкиваться в реальной жизни, чрезвычайно велико. Они могут сильно отличаться по сфере приложения, составу предметной области, масштабам, длительности, составу участников, степени сложности, влиянию результатов и т.п.

Под *управлением проектом* подразумевается деятельность, направленная на реализацию проекта с максимально возможной эффективностью при заданных ограничениях по времени, денежным средствам (и ресурсам), а также качеству конечных результатов проекта (документированных, например, в техническом задании).

В конце 50-х годов XX века в числе первых методов управления проектами были разработаны *методы сетевого планирования и управления [9]*:

– CPM (Critical Path Method – метод определения критического пути) – был разработан фирмой "Дюпон" для использования в крупных промышленных невоенных проектах;

– PERT (Program Evaluation and Review Technique – техника оценки и обзора проектов) – впервые использовалась в проекте "Поларис" фирмами "Локхид" и "Буз, Аллен и Гамильтон";

– диаграмма Ганта (Gantt chart – разделение всего проекта на определенную последовательность составных частей) – широко используется в современных пакетах прикладных программ по управлению проектами.

В 60-е годы начался поиск новых методов управления и организационных структур проектов, способных быстро приспосабливаться к изменяющим условиям. В 20-е годы широкое внедрение компьютерных систем обработки информации, растущие масштабы и сложность деятельности предприятий в условиях жесткой конкуренции способствовало тому, что все большее число компаний стало развивать и использовать методы управления проектами. Для того, чтобы справиться с ограничениями по времени используются методы построения и контроля календарных графиков работ. Для управления денежными ограничениями используются методы формирования финансового плана (бюджета) проекта и, по мере выполнения работ, соблюдение бюджета отслеживается, с тем, чтобы не дать затратам выйти из-под контроля. Для выполнения работ требуется их ресурсное обеспечение и существуют специальные методы управления человеческими и материальными ресурсами (например, матрица ответственности, диаграммы загрузки ресурсов). Из трех

основных ограничений труднее всего контролировать ограничения по заданным результатам проекта. Проблема заключается в том, что задания часто трудно и формулировать, и контролировать. Для решения данных проблем используются, в частности, методы управления качеством работ. Итак, руководители проектов отвечают за три аспекта реализации проекта: сроки, расходы и качество результата. В соответствии с общепринятым принципом управления проектами, считается, что эффективное управление сроками работ является ключом к успеху по всем трем показателям. Временные ограничения проекта часто являются наиболее критичными. Там, где сроки выполнения проекта серьезно затягиваются, весьма вероятными последствиями являются перерасход средств и недостаточно высокое качество работ. Поэтому, в большинстве методов управления проектами основной акцент делается на календарном планировании работ и контроле за соблюдением календарного графика.

Практически *применение методов и средств управления проектами* помогает решить следующие *основные задачи [7]*:

- обосновать целесообразность инвестиций;
- разработать оптимальную схему финансирования работ, поставок материалов и оборудования;
- составить план работ, включающий сроки исполнения работ, потребление ресурсов, необходимые затраты;
- проанализировать проектные риски;
- обеспечить эффективное взаимодействие участников проекта;
- эффективно контролировать исполнение составленного плана;
- анализировать отклонения фактического хода выполнения работ от запланированного и своевременно и обоснованно корректировать плановые показатели;
- моделировать управленческие воздействия на информационных моделях проектов и принимать обоснованные управленческие решения;

– вести архивы проектов и анализировать опыт их реализации, который может быть использован в других проектах, и многое другое.

Термин "менеджмент" применяется лишь к управлению социально-экономическими процессами на уровне организации (фирмы) в рыночных условиях. Менеджмент имеет свой собственный экономичный механизм, направленный на решение конкретных проблем взаимодействия в реализации социально-экономических, технологических, социально-психологических задач, возникающих в процессе хозяйственной деятельности. Экономичный механизм менеджмента состоит из трех блоков: внутрифирменное управление; управление производством; управление персоналом. Успехи и неудачи фирмы – это успехи и неудачи менеджмента. К числу основных принципов управления могут быть отнесены: научность; системность и комплексность; единоначалие и коллегиальность; демократический централизм; сочетание отраслевого и территориального подхода в управлении.

Теория и практика менеджмента – это систематизированный набор положений о наиболее эффективном управлении фирмой, носящих обобщающий, эмпирический и интуитивный характер. *Общий менеджмент охватывает планирование, организацию, обеспечение персоналом, исполнение и управление операционной деятельностью работающего предприятия*. В него входят [7]:

– управление финансами и бухгалтерский учет;
– закупки и снабжение;
– продажи и маркетинг;
– контракты и торговое право;
– производство и дистрибуция;
– логистика и логистическая цепочка;
– стратегическое, тактическое и оперативное планирование;
– организационные структуры, управление персоналом и карьерным ростом;

– здравоохранение и техника безопасности;

– информационные технологии.

Общий менеджмент обеспечивает основу для наработки навыков по управлению проектами и часто является необходимым для менеджера проекта. В любом проекте могут потребоваться навыки в любой из областей общего менеджмента.

Управление проектом – это интегративное действие. Интеграция управления проектом требует: все процессы проектов и продуктов должны быть выстроены и связаны с другими процессами таким образом, чтобы их можно было бы координировать и *оптимально (рационально)* ими управлять.

Управление проектами заключается в осуществлении и доведении проекта до логического завершения путем организации и управления людьми, временем, издержками и ресурсами. Как уже было отмечено, каждый проект содержит в себе несколько отличительных черт, делающих его уникальным. Отчасти из-за своей уникальности проект – изначально рискованное мероприятие.

Управление проектом – это такой подход к реализации проекта, при котором полномочия и ответственность за реализацию проекта передаются проект-менеджеру. При этом он, как правило, обладает полномочиями нанимать специалистов со стороны или привлекать к работе над проектом других членов организации, независимо от их должностного положения.

Управление проектом – это применение знаний, способностей, инструментов и технологий к широкому диапазону различных действий для того, чтобы выполнить цели проекта.

Американская общественная организация «Институт управления проектами» (Project Management Institute (PMI®)) приводит следующее определение управления проектами [7]: «Управление проектом - это искусство руководства и координации людских и материальных ресурсов на протяжении жизненного цикла проекта путем применения современных методов и техники управления для достижения определенных в проекте результатов по составу и

объему работ, стоимости, времени, качеству и удовлетворению участников проекта».

Управление проектом - интегрированный процесс [9]. Действия (или их отсутствие) в одном направлении обычно влияют и на остальные направления. Такая взаимосвязь заставляет балансировать между задачами проекта - часто улучшение в одной области может быть достигнуто лишь за счет ухудшения в другой. Для лучшего понимания интегрированной природы управления проектом опишем его через процессы, из которых оно состоит и их взаимосвязи.

Полная совокупность различных стадий развития проекта образуют жизненный цикл проекта. Начало жизненного цикла проекта совпадает по времени с началом проекта, а его окончание – с завершением проекта.

Любой проект проходит через определенные фазы в своем развитии. Фазы жизненного цикла проекта могут различаться в зависимости от сферы деятельности и принятой системы организации работ. Однако, у каждого проекта можно выделить начальную (пред инвестиционную) стадию, стадию реализации проекта и стадию завершения работ по проекту. Понятие жизненного цикла проекта является одним из важнейших для менеджера, поскольку именно текущая стадия определяет задачи и виды деятельности менеджера, используемые методики и инструментальные средства.

Концепция проекта. Разработка концепции проекта, по существу подразумевает функцию выбора проекта. Проекты инициируются в силу возникновения потребностей, которые нужно удовлетворить. Однако, в условиях дефицита ресурсов невозможно удовлетворить все потребности без исключения. Решения принимаются исходя из наличия ресурсов, и в первую очередь - финансовых возможностей, сравнительной важности удовлетворения одних потребностей и игнорирования других, сравнительной эффективности проектов. Решения по отбору проектов к реализации тем важнее, чем масштабнее предполагается проект, поскольку крупные проекты определяют

направление деятельности на будущее (иногда на годы) и связывают имеющиеся финансовые и трудовые ресурсы.

Определяющим показателем здесь является альтернативная стоимость инвестиций [10]. Иными словами, выбирая проект "А", а не проект "В", организация отказывается от тех выгод, которые мог бы принести проект "В".

Для сравнительного анализа проектов на данном этапе применяются методы проектного анализа, включающие в себя финансовый, экономический, коммерческий, организационный, экологический, анализ рисков и другие виды анализа проекта.

1.4 Задачи исследования

Быстрое развитие технологий дистанционного обучения привело к развитию нового направления обучения интернет-обучению, понятие которого близко по смыслу к понятиям: дистанционное обучение, e-learning, электронное обучение. Интернет-обучение расширяет сферу применения и в отличие от дистанционного обучения предполагает: доставку учебного контента обучаемым посредством сети интернет; возможность прохождения обучения всеми желающими.

Наибольшее распространение в интернет обучении получило использование средств, предполагающих общение в online режиме. Очень большое распространение получило использование вебинаров, которые проводят коммерческие компании для своих клиентов. Популярность вебинаров обусловлена невысокими инвестициями, требующимися для их организации и проведения.

Целью работы является построение вебинара для изучения профессионально-ориентированного иностранного языка (английский), апробирование и исследование его функций в учебном процессе.

Для достижения поставленной цели необходимо разработать:

– функциональную структуру вебинара, которая бы позволяла проведение виртуальных занятий, организацию коллективной работы;

– алгоритм функционирования информационной системы в режиме online и offline;

– реляционную модель базы данных (БД) для хранения контента знаний при изучении профессионально-ориентированного иностранного языка;

– информационное обеспечение, включающее реляционную БД и СУБД MySQL;

– прикладное программное обеспечение для разработанного вебинара с использованием языков PHP, HTML, JavaScript.

При этом необходимо учесть следующие ограничения:

– использовать клиент-серверную архитектуру;

– обеспечить техническую инфраструктуру, а именно минимально необходимый комплекс технических средств.

2 РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ

Базы данных (БД) являются важнейшей темой при изучении информационных систем. В последние годы всплеск популярности Интернета и бурное развитие новых технологий для интернета сделали знание технологии БД для многих одним из актуальнейших путей карьеры. Технологии БД увели интернет - приложения далеко от простых брошюрных публикаций, которые характеризовали ранние приложения. В то же время интернет-технология обеспечивает пользователям стандартизированные и доступные средства публикации содержимого БД. Правда, ни одна из этих новых разработок не отменяет необходимости в классических приложениях БД, которые появились еще до развития Интернета для нужд бизнеса. Это только расширяет важность знания БД. Проектирование и разработка базы данных требуют и искусства, и умения. Понимание пользовательских требований и перевод их в эффективный проект базы данных можно назвать творческим процессом. Преобразование этих проектов в физические базы данных с помощью функционально полных и высокопроизводительных приложений — инженерный процесс.

2.3 Выбор модели БД; выбор СУБД

В широком смысле слова база данных (БД) – это совокупность сведений о конкретных объектах реального мира в какой-либо предметной области. Для удобной работы с данными их необходимо структурировать, т.е. ввести определенные соглашения о способах их представления.

База данных (в узком смысле слова) — поименованная совокупность структурированных данных, относящихся к некоторой предметной области. В реальной деятельности в основном используют системы БД.

Система баз данных (СБД) – это компьютеризированная система хранения структурированных данных, основная цель которой – хранить информацию и предоставлять ее по требованию. Системы БД существуют и на малых, менее мощных компьютерах, и на больших, более мощных; на больших применяют в основном многопользовательские системы, на малых – однопользовательские [11]:

– однопользовательская система (single-user system) – это система, в которой в одно и то же время к БД может получить доступ не более одного пользователя;

– многопользовательская система (multi-user system) - это система, в которой в одно и то же время к БД может получить доступ несколько пользователей.

Основная задача большинства многопользовательских систем – позволить каждому отдельному пользователю работать с системой как с однопользовательской. Различия однопользовательской и многопользовательской систем содержатся в их внутренней структуре, конечному пользователю они практически не видны.

СБД содержит четыре основных элемента: *данные, аппаратное обеспечение, программное обеспечение* и *пользователи*.

Данные в БД являются *интегрированными* и *общими*[11].

Интегрированные – значит, данные можно представить как объединение нескольких, возможно перекрывающихся, отдельных файлов данных. (Например, имеется файл, содержащий данные о студентах – фамилию, имя, отчество, дату рождения, адрес и т.д., а другой – о спортивной секции. Необходимые данные о студентах, посещающих секцию, можно получить путем обращения к первому файлу.)

Общие – значит, отдельные области данных могут использовать различные пользователи, т.е. каждый из этих пользователей может иметь доступ к одной и той же области данных, даже одновременно. (Например, одни

и те же данные БД о студентах может одновременно использовать студенческий отдел кадров и деканат.)

К аппаратному обеспечению относятся:

– накопители для хранения информации вместе с подсоединенными устройствами ввода-вывода, каналами ввода-вывода и т.д.

– процессор (или процессоры) вместе с основной памятью, которая используется для поддержки работы программного обеспечения системы.

Между собственно данными и пользователями располагается уровень программного обеспечения. Ядром его является система управления базами данных (database management system –DBMS), или диспетчер БД (database manager).

Система управления базами данных (СУБД) - это комплекс программных и языковых средств, необходимых для создания БД, поддержания их в актуальном состоянии и организации поиска в них необходимой информации.

Основная функция СУБД – это предоставление пользователю БД возможности работы с ней, не вникая в детали на уровне аппаратного обеспечения. Все запросы пользователя к БД, добавление и удаление данных, выборки, обновление данных – все это обеспечивает СУБД. Иными словами, СУБД поддерживает пользовательские операции высокого уровня. Сюда включены и операции, которые можно выполнить с помощью языка SQL.

SQL - это специальный язык БД. Сейчас он поддерживается большинством СУБД является официальным стандартом языка для работы с реляционными системами. Название SQL вначале было аббревиатурой от Structured Query Language (язык структурированных запросов), сейчас название языка уже не считается аббревиатурой, так как функции его расширились и не ограничиваются только созданием запросов.

СУБД – это не единственный программный компонент системы, хотя и наиболее важный. Среди других – утилиты, средства разработки приложений, средства проектирования, генераторы отчетов и т.д.

Пользователей СУБД можно разделить на три группы.

Прикладные программисты. Отвечают за написание прикладных программ, использующих БД. Для этих целей применимы различные языки программирования. Прикладные программы выполняют над данными стандартные операции – выборку, вставку, удаление, обновление – через соответствующий запрос к СУБД. Такие программы бывают простыми – *пакетной обработки*, или *оперативными приложениями* – для поддержки работы конечного пользователя.

Конечные пользователи. Работают с системами БД непосредственно через рабочую станцию или терминал. Конечный пользователь может получить доступ к БД, используя оперативное приложение или интегрированный интерфейс самой СУБД (такой интерфейс тоже является оперативным приложением, но встроенным). В большинстве систем есть хотя бы одно такое встроенное приложение – *процессор языка запросов* (или *командный интерфейс*). Язык SQL – пример языка запросов для БД. Кроме языка запросов в современных СУБД, как правило, есть *интерфейсы, основанные на меню и формах* – для непрофессиональных пользователей. Понятно, что командный интерфейс более гибок, содержит больше возможностей.

Администраторы БД. Отвечают за создание БД, технический контроль, обеспечение быстродействия системы, ее техническое обслуживание.

СУБД имеют свою архитектуру. В процессе разработки и совершенствования СУБД предлагались различные архитектуры, но самой удачной оказалась трехуровневая архитектура, предложенная исследовательской группой ANSI/SPARC американского комитета по стандартизации ANSI (American National Standards Institute) [12].

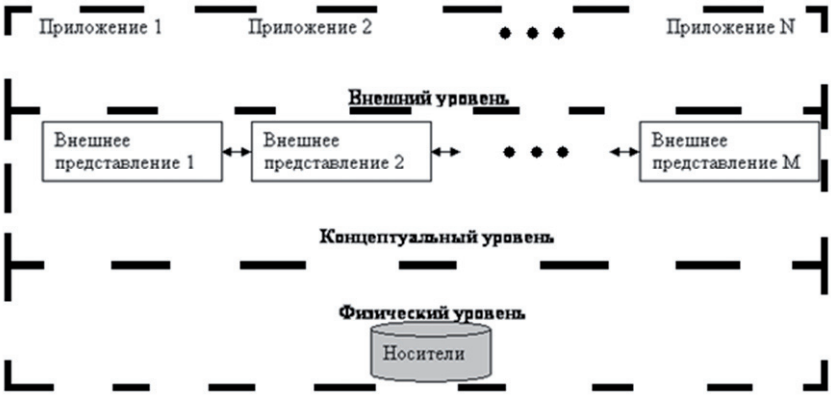

Рисунок 3 – Упрощенная схема архитектуры СУБД

Внешний уровень – это уровень пользователя. По сути, это совокупность внешних представлений данных, которые обрабатывают приложения и какими их видит пользователь на экране. Это может быть таблица с отсортированными данными, с примененным фильтром, форма, отчет, результат запроса. Внешние представления взаимосвязаны, т.е. из одного внешнего представления можно получить другое, как показано на рисунке 3, здесь M≠N, в частном случае M=N.

Концептуальный уровень – центральный. Здесь БД представлена в наиболее общем виде, который объединяет данные, используемые всеми приложениями. Т.е. это обобщенная *модель* предметной области, для которой созданы БД. Можно сказать, что концептуальный уровень формируется при создании таблиц (определение их полей, типов, свойств), связей, а так же при заполнении таблиц, в соответствии с рисунком 3, здесь M≠N, в частном случае M=N.

Физический уровень – собственно данные, расположенные на внешних носителях, что можно увидеть на рисунке 3.

2.3.1 Классификация моделей данных

Ядром любой БД является модель данных. Модель данных – это совокупность структур данных и операций их обработки. Так как СУБД имеет 3-х уровневую архитектуру, то понятие модели данных связано с каждым уровнем. *Физическая* модель данных связана с организацией внешней памяти и структур хранения, используемых в данной операционной среде. На *концептуальном* уровне модели данных важны для разработчиков БД, т.к. именно ими определяется тип СУБД. Для *внешнего* уровня отдельных моделей данных нет, они лишь являются подсхемами концептуальных моделей данных.

Кроме моделей данных, соответствующих трем уровням архитектуры СУБД, существуют *предшествующие* им, не связанные с компьютерной реализацией. Они служат переходным звеном от реального мира к БД. Это класс инфологических (семантических) моделей.

Модели данных [11]:

1 Инфологические (семантические) модели.
- 1.1 Диаграмма Бахмана.
- 1.2 Модель сущность-связь (ER-модель).

2 Физические модели.
- 2.1 Основанные на файловых структурах.
- 2.2 Основанные на странично-сегментарной организации.

3 Даталогические модели.
- 3.1 Документальные модели.
  - 3.1.1 Ориентированные на формат документа.
  - 3.1.2 Дескрипторные модели.
  - 3.1.3 Тезаурусные модели.
- 3.2 Фактографические модели.
  - 3.2.1 Объектно-ориентированные.
  - 3.2.2 Теоретико-графовые.
    - 3.2.2.1 Иерархическая.

3.2.2.2 Сетевая.

3.2.3 Теоретико-множественные

3.2.3.1 Реляционная.

3.2.3.2 Бинарных ассоциаций (инвертированных списков).

Инфологические (семантические) модели данных используются на ранних стадиях проектирования БД.

Даталогические модели данных уже поддерживаются конкретной СУБД.

Физические модели данных связаны с организацией данных на носителях.

Документальные модели данных соответствуют слабоструктурированной информации, ориентированной на свободные форматы документов на естественном языке.

Модели данных, *ориентированные на формат документа*, связаны со стандартным общим языком разметки SGML (Standart Generaliset Markup Language), а также HTML, предназначенным для управления процессом вывода содержимого документа на экран.

Дескрипторные модели данных – самые простые, широко использовались раньше. В них каждому документу соответствует дескриптор – описатель, который имеет жёсткую структуру и описывает документ в соответствии с заранее определенными характеристиками.

Тезаурусные модели данных основаны на принципе организации словарей. Содержат языковые конструкции и принципы их взаимодействия в заданной грамматике. Эти модели используются, например, в системах-переводчиках.

Объектно-ориентированная модель перекликается с семантическими моделями данных. Принципы похожи на принципы объектно-ориентированных языков программирования. Структура таких моделей графически представима в виде дерева, узлами которого являются объекты. Свойства объектов описываются типом.

Объекты *иерархической* модели данных связанны иерархическими отношениями и образуют ориентированный граф. Основные понятия иерархических структур: уровень, узел (совокупность свойств данных, описывающих объект), связь.

В *сетевой* модели данных при тех же основных понятиях (уровень, узел, связь) каждый элемент может быть связан с любым другим элементом.

В *реляционной* модели данных данные представлены только в виде таблиц.

Исходя из выше изложенного мною выбрана реляционная модель БД.

Создавая базу данных, требуется упорядочить информацию по различным признакам для того, чтобы потом извлекать из нее необходимые нам данные в любом сочетании. Сделать это возможно, только если данные структурированы. Существует огромное количество разновидностей баз данных, отличающихся по различным критериям и имеющие свои плюсы и минусы.

Поэтому для создания базы данных в данной работе я выбрал клиент-серверную архитектуру.

Логическим развитием архитектуры информационных систем, причем развитием на качественно новом уровне, стала архитектура “*клиент-сервер*”. Все современные системы управления базами данных, и продукция фирмы Informix Software, в частности, выполнены по архитектуре “клиент-сервер”. Архитектура “клиент-сервер” относится к описанию взаимодействия программ, причем как находящихся на одном компьютере, так и на разных. Данная архитектура получила широкое распространение благодаря широкому внедрению вычислительных сетей. Рассмотрим основные идеи этой архитектуры [13].

Если мы вернемся к архитектуре информационных систем на основе идеи файл-сервера, то принципиальным недостатком такой архитектуры являлось разнесение программ, обрабатывающих данные, и самих данных. Основным принципиальным недостатком терминальной архитектуры являлось слишком

сильное разнесение пользователя и программы обработки. Архитектура “клиент-сервер” устраняет эти недостатки.

Архитектура “клиент-сервер” подразумевает наличие двух типов программ - программы-клиента и программы-сервера. Программа-клиент является “активной” программой, то есть в ее задачи входит генерация некоторых обращений за услугами к программе серверу. Программа сервер является пассивной программой, то есть в ее функции входит ожидание запроса от программы-клиента. Когда такой запрос поступает, программа-сервер отрабатывает его и, при необходимости, возвращает программе-клиенту некоторые результаты, согласно рисунку 4.

Рисунок 4 – Многопользовательская информационная система на основе архитектуры “клиент сервер”

Естественно, взаимодействие между программой-клиентом и программой-сервером должно происходить по четко определенному интерфейсу. В противном случае, они рискуют не понять друг друга.

Как правило компьютеры и программы, входящие в состав информационной системы, не являются равноправными. Некоторые из них владеют ресурсами (файловая система, процессор, принтер, база данных и т.д.), другие имеют возможность обращаться к этим ресурсам. Компьютер (или программу), управляющий ресурсом, называют сервером этого ресурса (файл-сервер, сервер базы данных, вычислительный сервер...). Клиент и сервер какого-либо ресурса могут находится как на одном компьютере, так и на различных компьютерах, связанных сетью.

В рамках многоуровневого представления вычислительных систем можно выделить три группы функций, ориентированных на решение различных подзадач:

– функции ввода и отображения данных (обеспечивают взаимодействие с пользователем);

– прикладные функции, характерные для данной предметной области;

– функции управления ресурсами (файловой системой, базой данных и т.д.)

Рисунок 5 – Компоненты сетевого приложения

Выполнение этих функций в основном обеспечивается программными средствами, которые можно представить в виде взаимосвязанных компонентов, как показано на рис 5, где:

– компонент представления отвечает за пользовательский интерфейс;

– прикладной компонент реализует алгоритм решения конкретной задачи;

– компонент управления ресурсом обеспечивает доступ к необходимым ресурсам.

Автономная система (компьютер, не подключенный к сети) представляет все эти компоненты как на различных уровнях (ОС, служебное ПО и утилиты,

прикладное ПО), так и на уровне приложений (не характерно для современных программ). Так же и сеть — она представляет все эти компоненты, но, в общем случае, распределенные между узлами. Задача сводится к обеспечению сетевого взаимодействия между этими компонентами.

Архитектура «клиент-сервер» определяет общие принципы организации взаимодействия в сети, где имеются серверы, узлы-поставщики некоторых специфичных функций (сервисов) и клиенты, потребители этих функций.

Практические реализации такой архитектуры являются клиент-серверными технологиями. Каждая технология определяет собственные или использует имеющиеся правила взаимодействия между клиентом и сервером, которые называются протоколом обмена (протоколом взаимодействия).

Двухзвенная архитектура

В любой сети (даже одноранговой), построенной на современных сетевых технологиях, присутствуют элементы клиент-серверного взаимодействия, чаще всего на основе двухзвенной архитектуры. Двухзвенной (two-tier, 2-tier) она называется из-за необходимости распределения трех базовых компонентов между двумя узлами (клиентом и сервером).

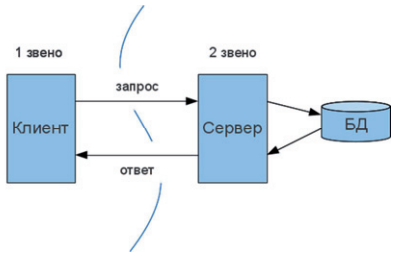

Рисунок 6 – Двухзвенная клиент-серверная архитектура

Двухзвенная архитектура используется в клиент-серверных системах, где сервер отвечает на клиентские запросы напрямую и в полном объеме, при этом используя только собственные ресурсы. Сервер не вызывает сторонние сетевые

приложения и не обращается к сторонним ресурсам для выполнения какой-либо части запроса, как показано на рис 6.

Расположение компонентов на стороне клиента или сервера определяет следующие основные модели их взаимодействия в рамках двухзвенной архитектуры [13]:

– сервер терминалов — распределенное представление данных;

– файл-сервер — доступ к удаленной базе данных и файловым ресурсам;

– сервер БД — удаленное представление данных;

– сервер приложений — удаленное приложение.

Перечисленные модели с вариациями представлены на рисунке 7.

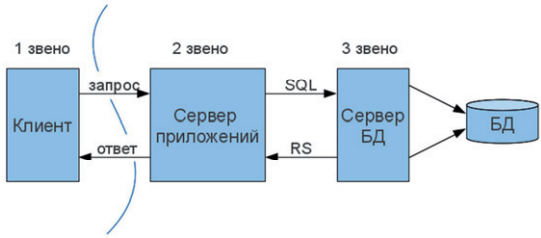

Рисунок 7 – Модели клиент-серверного взаимодействия

Исторически первой появилась модель распределенного представления данных (модель сервер терминалов). Она реализовывалась на универсальной ЭВМ (мэйнфрейме), выступавшей в роли сервера, с подключенными к ней алфавитно-цифровыми терминалами. Пользователи выполняли ввод данных с клавиатуры терминала, которые затем передавались на мэйнфрейм и там выполнялась их обработка, включая формирование «картинки» с результатами. Эта «картинка» и возвращалась пользователю на экран терминала.

С появлением персональных компьютеров и локальных сетей была реализована модель файлового сервера, представлявшего доступ файловым ресурсам и к удаленной базе данных. В этом случае выделенный узел сети

является файловым сервером, на котором размещены файлы базы данных. На клиентах выполняются приложения, в которых совмещены компонент представления и прикладной компонент (СУБД и прикладная программа), использующие подключенную удаленную базу как локальный файл. Протоколы обмена при этом представляют набор низкоуровневых вызовов операций файловой системы.

Такая модель показала свою неэффективность ввиду того, что при активной работе с таблицами БД возникает большая нагрузка на сеть. Частичным решением является поддержка тиражирования (репликации) таблиц и запросов. В этом случае, например при изменении данных, обновляется не вся таблица, а только модифицированная ее часть.

С появлением специализированных СУБД появилась возможность реализации другой модели доступа к удаленной базе данных — модели сервера баз данных. В этом случае ядро СУБД функционирует на сервере, прикладная программа на компьютере клиенте, а протокол обмена обеспечивается с помощью языка SQL. Такой подход по сравнению с файловым сервером ведет к уменьшению загрузки сети и унификации интерфейса «клиент-сервер». Однако, сетевой трафик остается достаточно высоким, кроме того, по прежнему невозможно удовлетворительное администрирование приложений, поскольку в одной программе совмещаются различные функции.

С разработкой и внедрением на уровне серверов баз данных механизма хранимых процедур появилась концепция активного сервера БД. В этом случае часть функций прикладного компонента реализованы в виде хранимых процедур, выполняемых на стороне сервера. Остальная прикладная логика выполняется на клиентской стороне. Протокол взаимодействия — соответствующий диалект языка SQL.

Преимущества такого подхода очевидны:

– возможно централизованное администрирование прикладных функций;

– снижение стоимости владения системой (TOC, total cost of ownership) за счет аренды сервера, а не его покупки;

– значительное снижение сетевого трафика (т.к. передаются не SQL-запросы, а вызовы хранимых процедур).

Основной недостаток — ограниченность средств разработки хранимых процедур по сравнению с языками высокого уровня.

Реализация прикладного компонента на стороне сервера представляет следующую модель — сервер приложений. Перенос функций прикладного компонента на сервер снижает требования к конфигурации клиентов и упрощает администрирование, но представляет повышенные требования к производительности, безопасности и надежности сервера.

В настоящее время намечается тенденция возврата к тому, с чего начиналась клиент-серверная архитектура — к централизации вычислений на основе модели терминал-сервера. В современной реинкарнации терминалы отличаются от своих алфавитно-цифровых предков тем, что имея минимум программных и аппаратных средств, представляют мультимедийные возможности. Работу терминалов обеспечивает высокопроизводительный сервер, куда вынесено все, вплоть до виртуальных драйверов устройств, включая драйверы видеоподсистемы [13].

2.4 Разработка БД для вебинара

При выполнении работы создана БД, которая содержит информацию о студентах, группах студентов на данной дисциплине. В данной БД содержится информация о студентах: id, логин, пароль, код активации. Подробную информацию о студентах: id, имя, Email, статус, уровень группы, номер группы. Информацию о группе студентов: имя группы, название группы, номер группы, размер группы. Для разработанного web-сайта была создана таблица 1 user, которая хранит информацию о студентах; таблица 2 userdata, которая

содержит подробную информацию о пользователях; таблица 3 usergroup , которая содержит информацию о группах пользователей. На рисунке 6 изображена схема данных в БД.

Таблица 1 - Структура таблицы user

Название поля	Тип данных	Назначение	Свойства
id	Int	Номер пользователя в таблице	Размер символов-11
Login	Char	Логин пользователя	Размер символов-50
Pass	Char	Пароль пользователя	Размер символов-50
Hash	Char	Код активации	Размер символов-50

Таблица 2 - Структура таблицы userdata

Название поля	Тип данных	Назначение	Свойства
id	Int	Номер пользователя в таблице	Размер символов-11
Email	Char	Email пользователя	Размер символов-50
Status	Bit	Статус аккаунта	Размер символов-1
Name	Char	Имя пользователя	Размер символов-50
Level	Char	Уровень знания языка пользователем	Размер символов-50
Idgroup	Int	Номер группы пользователя	Размер символов-11

Таблица 2. Структура таблицы usergroup

Название поля	Тип данных	Назначение	Свойства
idgroup	Int	Номер группы в таблице	Размер символов-11
Namegroup	Char	Название пользователя	Размер символов-50
Level	Char	Уровень группы	Размер символов-50
Groupsize	Int	Размер группы	Размер символов-11

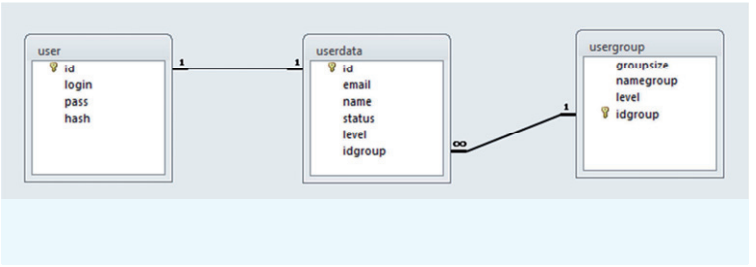

Рисунок 6 – Схема данных в БД

2.5 Создание карты сайта, макета сайта

Карта сайта – это отдельная страница сайта, на которой собраны ссылки на все внутренние страницы сайта. Обычно карта сайта создается для удобства посетителей. По ней легко найти нужный материал, просмотрев всю структуру сайта. Это касается HTML - карты. А вот для поисковиков создается несколько иной формат карты сайта, так называемый sitemap XML. Он не читабелен для посетителей, так как представляет собой файл XML, в котором содержится

список всех URL сайта. Создается sitemap XML с одной целью - ускорить индексацию вашего сайта роботами поисковых систем. Роботам будет предоставлен список всех URL на сайте, даже тех, которые трудно обнаружить в ходе стандартного сканирования.

Для сайтов с небольшим количеством страниц, можно создать карту сайта самостоятельно. Для этого создается на сайте еще одна страница с названием ее "Карта сайта". Затем в виде перечня ссылок указываются все страницы сайта, начиная с главной страницы, как показано на рисунке 9.

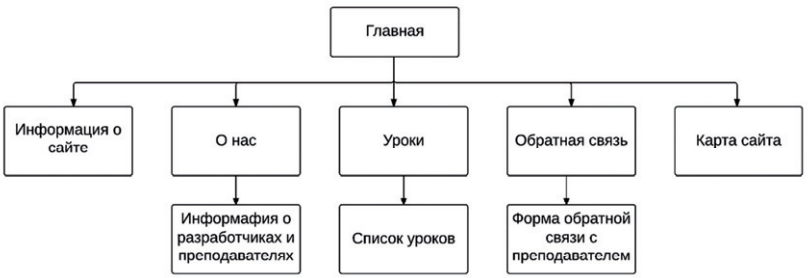

Рисунок 9 – Карта сайта

При планировании структуры сайта необходимо продумать несколько основных вещей: структуру каталогов, структуру навигации, необходимость заставки сайта. Для этого следует определить структуру вашего будущего сайта. Существует несколько основных структур [14].

Линейная - страницы располагаются в определенном порядке. Переход с одной страницы на другую строго определен. Такая структура обоснована, например, при обучении. Общая схема представлена на рисунке 10.

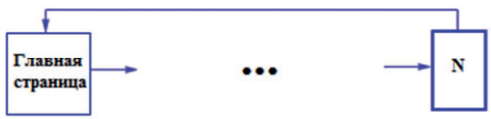

Рисунок 10 – Линейная структура сайта

Иерархическая - страницы разбиты по категориям и подкатегориям. Такая структура наиболее удобна. Структура показана на рисунке 11.

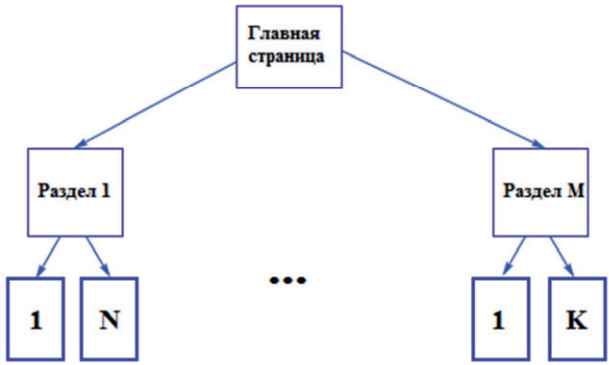

Рисунок 11 – Иерархическая структура сайта

Произвольная - страницы расположены в свободном порядке. Такая структура оправдана только для небольших сайтов. Структура показана на рисунке 12.

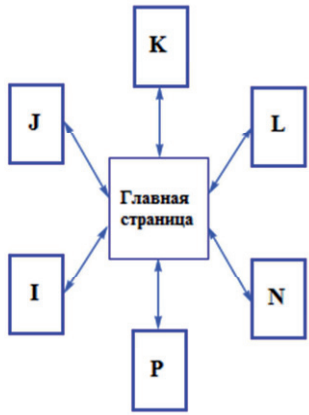

Рисунок 12 – Произвольная структура сайта

Мною была выбрана иерархическая структура сайта. Данная структура позволяет более удобно и доступно разместить весь необходимый контент. Разработанная структура сайта представлена на рисунке 13.

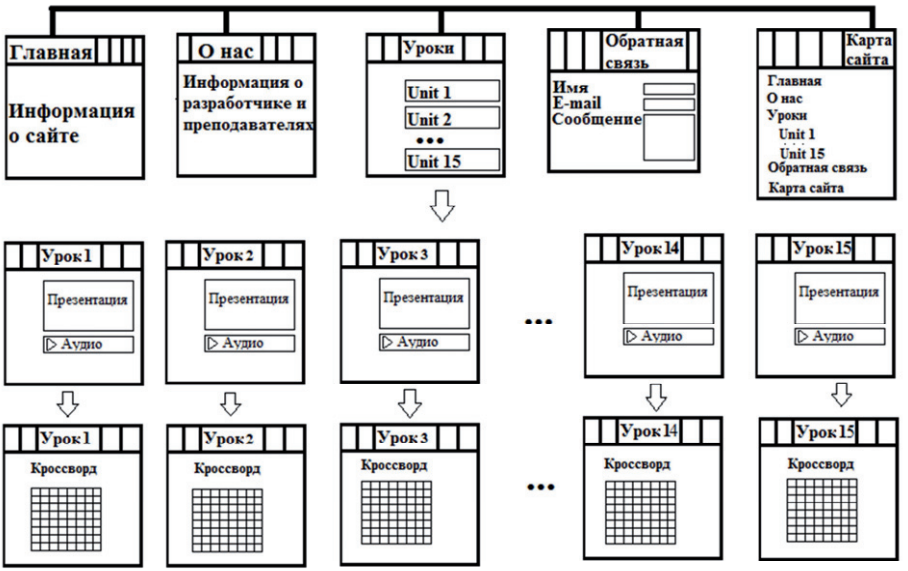

Рисунок 13 – Разработанная структура сайта для вебинара

Структура разработанного web-сайта позволяет наиболее выгодно и просто разместить контент для изучения профессионально-ориентированного иностранного языка. Сайт начинается с главной страницы откуда можно попасть на страницы: О нас, Уроки, Обратная связь, Карта сайта. На странице Уроки модно увидеть список занятий. Каждое занятие является отдельной страницей где содержится контент знаний.

3 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ВЕБИНАРА

Программное обеспечение для проведения вебинаров, как правило, позволяет:

- демонстрировать документы в наиболее распространённых форматах;
- передавать речь и видеоизображение ведущего и нескольких участников;
- общаться в чате и приватном чате;
- демонстрировать видеоролики;
- рисовать графические объекты и текст на белой доске;
- осуществлять перехват экрана компьютера;
- размещать файлы для обмена;
- проводить опросы слушателей.

3.3 Разработка структуры программного обеспечения

В настоящее время наряду с понятием программа используется понятие приложение. Между ними нет принципиальной разницы. Есть мнение, что программа – это одна единица, а приложение – это совокупность программ, решающих совместно одну или несколько близких задач. Однако данное деление может быть достаточно условным в связи с тем, что большинство даже очень простых программ обычно включают различные библиотеки и модули сторонних разработчиков. С другой стороны, вычленить из приложения какую-либо программу так, чтобы она работала самостоятельно, может быть невозможно [15]. Совокупность программ, предназначенная для решения задач на компьютере, называется программным обеспечением (ПО). Это могут быть внутренние команды, управляющие оборудованием или программа,

выполняющая какие либо действия в ответ на вводимые с клавиатуры команды. Состав программного обеспечения компьютера называют программной конфигурацией. Оно является логическим продолжением технических средств.

Программное обеспечение, можно условно разделить на три категории, как показано на рисунке 14:

– системное ПО, выполняющие различные вспомогательные функции, например создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.д.;

– инструментальное ПО, обеспечивающее разработку новых программ для компьютера на языке программирования;

– прикладное ПО, обеспечивающее выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, обработка информационных массивов и т.д.

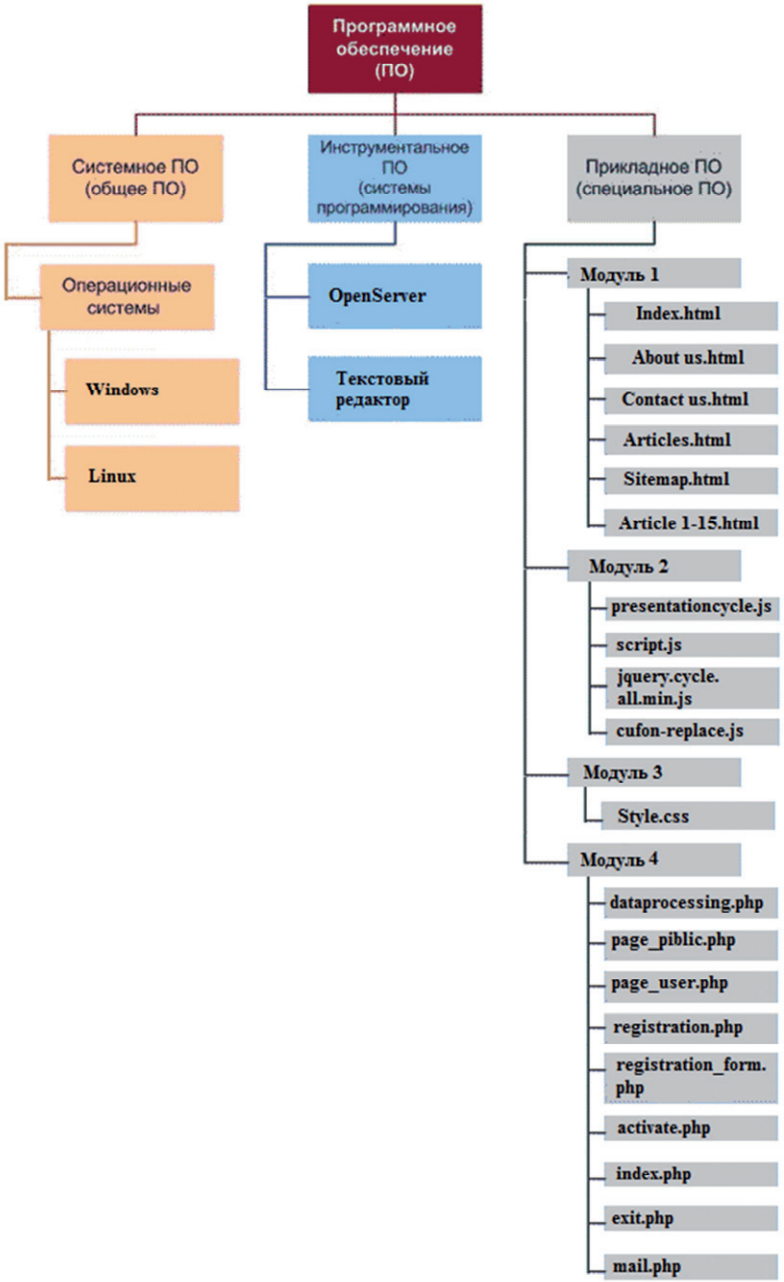

Рисунок 14 – Структура программного обеспечения

Системное ПО – это программы общего пользования не связаны с конкретным применением ПК и выполняют традиционные функции: планирование и управление задачами, управления вводом-выводом и т.д. Другими словами, системные программы выполняют различные вспомогательные функции, например, создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и другие функции, как представлено на рисунке 14. К системному ПО относятся [15]:

- операционные системы (эта программа загружается в ОЗУ при включении компьютера);
- программы – оболочки (обеспечивают более удобный и наглядный способ общения с компьютером, чем с помощью командной строки DOS);
- операционные оболочки – интерфейсные системы, которые используются для создания графических интерфейсов, мультипрограммирования и прочие;
- драйверы (программы, предназначенные для управления портами периферийных устройств, обычно загружаются в оперативную память при запуске компьютера);
- утилиты (вспомогательные или служебные программы, которые представляют пользователю ряд дополнительных услуг).

К утилитам относятся:

- диспетчеры файлов или файловые менеджеры;
- средства динамического сжатия данных (позволяют увеличить количество информации на диске за счет ее динамического сжатия);
- средства просмотра и воспроизведения;
- средства диагностики; средства контроля позволяют проверить конфигурацию компьютера и проверить работоспособность устройств компьютера, прежде всего жестких дисков;

– средства коммуникаций (коммуникационные программы) предназначены для организации обмена информацией между компьютерами ;

– средства обеспечения компьютерной безопасности (резервное копирование, антивирусное ПО).

Необходимо отметить, что часть утилит входит в состав операционной системы, а другая часть функционирует автономно. Большая часть системного ПО входит в состав операционной системы. Часть общего ПО входит в состав самого компьютера (часть программ операционной системы и контролирующих тестов записана в ПЗУ или ППЗУ, установленных на системной плате). Часть общего ПО относится к автономными программам и поставляется отдельно.

Инструментальное ПО или системы программирования - это системы для автоматизации разработки новых программ на языке программирования. В самом общем случае для создания программы на выбранном языке программирования (языке системного программирования) нужно иметь следующие компоненты, указанные в рисунке 14 [15]:

– текстовый редактор для создания файла с исходным текстом программы. Мной был выбран текстовый редактор *Notepad++*. Выбранный текстовый редактор является бесплатным приложением для редактирования исходных текстовых программы;

– Open Server — это портативная серверная платформа и программная среда, созданная специально для веб-разработчиков с учётом их рекомендаций и пожеланий. Программный комплекс имеет богатый набор серверного программного обеспечения, удобный, многофункциональный продуманный интерфейс, обладает мощными возможностями по администрированию и настройке компонентов. Платформа широко используется с целью разработки, отладки и тестирования веб-проектов, а так же для предоставления веб-сервисов в локальных сетях. Хотя изначально программные продукты, входящие в состав комплекса, не разрабатывались специально для работы друг с другом, такая связка стала весьма популярной

среди пользователей Windows, в первую очередь из-за того, что они получали бесплатный комплекс программ с надежностью на уровне Linux серверов.

Прикладное ПО могут использоваться автономно или в составе программных комплексов или пакетов. Прикладное ПО – программы, непосредственно обеспечивающие выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, создание электронных таблиц и т.д. Пакеты прикладных программ – это система программ, которые по сфере применения делятся на проблемно – ориентированные, пакеты общего назначения и интегрированные пакеты. Современные интегрированные пакеты содержат до пяти функциональных компонентов: тестовый и табличный процессор, СУБД, графический редактор, телекоммуникационные средства. Разработанное прикладное ПО отображено на рисунке 14.

К прикладному ПО, например, относятся [15]:

- комплект офисных приложений MS OFFICE;
- бухгалтерские системы;
- финансовые аналитические системы;
- интегрированные пакеты делопроизводства;
- CAD – системы (системы автоматизированного проектирования);
- редакторы HTML или Web – редакторы;
- браузеры – средства просмотра Web – страниц;
- графические редакторы;
- экспертные системы.

3.4 Описание прикладных программ

Программный проект приложения вебинар состоит из одного style.css, одной БД и следующих html, php, js страниц:

- index.html;

- article.html;
- unit1.html;
- unit2.html;
- unit3.html;
- …
- unit15.html;
- about us.html;
- contact.html;
- sitemap.html;
- active.php;
- registration.php;
- registration_form.php;
- dataprocessing.php;
- page_public.php;
- page_user.php;
- exit.php;
- mail.php;
- jquery.cycle.all.min.js;
- presentationcycle.js;
- cufon-replace.js.

3.4.1 Функциональное назначение

Описание функционального назначения модулей разработанного веб сайта:

- style.css – файл содержащий в себе разметку и стиль для html страниц;
- index.html – главная страница сайта;
- article.html – страница со списком занятий по дисциплине;

– unit1.html - unit15.html – страницы с занятиями по дисциплине;

– about us.html – страница с информацией о разработчике и преподавателе;

– contact.html – страница обратной связи студентов с преподавателем;

– sitemap.html – карта сайта.

– mail.php – файл позволяющий отправлять письма с формы обратной связи;

– registration_form.php – форма регистрации;

– registration.php – обработчик формы регистрации;

– activate.php – обработчик кода активации, полученного от пользователя через почту;

– index.php — главная страница пользователя в его личном кабинете;

– exit.php — выход из личного кабинета пользователя;

– page_public.php — библиотека классов для публичной части сайта;

– page_user.php — библиотека классов для личного кабинета пользователя;

– dataprocessing.php — библиотека классов для обработки данных;

– jquery.cycle.all.min.js – сценарий для плавного перехода по сраницам;

– presentationcycle.js – сценарий для прокрутки презентаций;

– cufon-replace.js – сценарий для добавления эффектов на странице.

3.5 Описание логической структуры модулей

Тексты всех описанных модулей программного пакета приведены в Приложении А.

Логическая структура модуля style.css с привязкой к строкам текста имеет следующий вид:

1-6 – группа тегов описанных в стиле;

7-13 – настройки для основного тела страницы;

14-19 – масштабирование для страниц;

20-26 – блок с правилами для тега container;

27-35 – правила оформления для заголовка страницы;

36-63 – настройки для тегов связанных с отображением текста;

64-83 – оформление для текста;

84-111 – оформление для фона блока;

112-125 – оформление для страниц с уроками;

126-136 – оформление для страницы с картой сайта;

137-159 – атрибут для списка вариантов;

160-172 – стиль для иконок уроков;

173-179 – правила для отступа;

180-203 – правила для страницы с обратной связью;

204-225 – добавление стиля для заголовка;

226-279 – разметка для текста;

280-346 – задний фон для заголовка страницы;

347-365 – правила для блока с текстом;

367-396 – правила для блоков на странице с обратной связью;

397-439 – правила оформления для нижнего колонтитула;

447-532 – стиль и ограничения для презентаций.

Логическая структура модуля index.html с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с формой авторизации и меню;

81-92 – контейнер с информацией о сайте;

94-104 – нижний колонтитул.

Логическая структура модуля article.html с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с формой авторизации и меню;

81-124 – ссылки на уроки;

124-134 – нижний колонтитул.

Логическая структура модуля unit1.html - unit15.html с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с формой авторизации и меню;

81-144 –контент знаний отображенный на сайте;

145-254 – скрипт кроссворда;

255-265 – нижний колонтитул.

Логическая структура модуля about us.html с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с формой авторизации и меню;

81-104 – контейнер с информацией о разработчиках;

105-114 – нижний колонтитул.

Логическая структура модуля contact.html с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с формой авторизации и меню;

81-104 – форма обратной связи с преподователем;

105-114 – нижний колонтитул.

Логическая структура модуля sitemap.html с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с формой авторизации и меню;

81-104 – карта сайта;

105-114 – нижний колонтитул.

Логическая структура модуля mail.php с привязкой к строкам текста имеет следующий вид:

1-24 – проверка заполнения полей;

25-55 – формирование и отправка сообщения.

Логическая структура модуля registration_form.php с привязкой к строкам текста имеет следующий вид:

1-8 – подключение класса и определение функции;

9-17 – определение таблицы с полями и заголовком;

18-52 – активная форма для заполнения данных.

Логическая структура модуля registration.php с привязкой к строкам текста имеет следующий вид:

1-8 – подключение класса и определение функции;

9-32 – проверка заполнения полей;

33-42 – формирование и отправка письма с ссылкой для подтверждения регистрации;

43-59 – проверка отправки письма, если не отправилось выводим сообщение о некорректном вводе email, иначе добавляем запись в БД;

60-79 – вывод результатов в зависимости от проверки заполнения полей.

Логическая структура модуля activate.php с привязкой к строкам текста имеет следующий вид:

1-8 – подключение класса и определение функции;

9-19 – проверка кода для регистрации;

20-25 – если код активации найден в БД активируем аккаунт;

26-37 – вывод результатов в зависимости от проверки кода активации.

Логическая структура модуля index.php с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с личными данными пользователя и меню;

81-92 – контейнер с информацией о сайте;

94-104 – нижний колонтитул.

Логическая структура модуля exit.php с привязкой к строкам текста имеет следующий вид:

1 – начало страницы;

2 – начало заголовка;

3-16 – подключение JavaScript;

17-34 – контейнер с ссылками в заголовке;

35-80 – контейнер с меню;

81-92 – контейнер с информацией выходе из личного аккаунта;

94-104 – нижний колонтитул.

Логическая структура модуля page_public.php с привязкой к строкам текста имеет следующий вид:

1-8 – подключение класса и определение функции;

9-29 – вывод содержания страницы;

30-41 – вывод результатов в зависимости от проверки регистрационных данных.

Логическая структура модуля page_user.php с привязкой к строкам текста имеет следующий вид:

1-8 – подключение класса и определение функции;

9-34 – вывод содержания страницы;

35-115 – проверка данных при аутентификации.

Логическая структура модуля dataprocessing.php с привязкой к строкам текста имеет следующий вид:

1-8 – подключение класса и определение функции;

9-12 – начало сессии;

13-25 – подключение к БД;

26-52 – проверка введенных данных;

53-83 – поиск введенных данных в БД и вывод результатов поиска.

3.6 Выбор языка программирования

Язык веб-программирования – это совокупность операторов, с помощью которых создаются коды веб-программ, или их еще называют скриптами, сценариями. Язык программирования передает понятные компьютеру инструкции для выполнения определенных операций. Так, с помощью языков программирования человек «разговаривает» с машиной. Обычно коды, написанные на веб-языках, читаются браузерами. Среди самых распространенных языков веб-программирования можно отметить: HTML, CSS, PHP, JavaScript, Perl, jQuery [16]:

HTML (HyperText Markup Language - "язык разметки гипертекста") – самый известный для веб-разработчиков язык программирования, хотя по своей функциональности он скорее всего относится к языкам разметки. Язык применяется для распределения объектов и текста на веб-страницах. Для лучшего понимания сущности языка HTML можно косвенно сравнить с программой Office Word. Язык оснащен тегами, которые и являются, по сути, инструкциями компьютеру. Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи. Например, тег <TABLE> предназначен для создания в документах таблиц, но часто используется и для оформления размещения элементов на странице. С течением времени основная идея платформонезависимости языка HTML была принесена в жертву современным потребностям в мультимедийном и графическом оформлении.

PHP (Hypertext Preprocessor– “процессор гипертекста”) [17] – является СИ-подобным скриптовым языком. Самая первая версия PHP была разработана еще в 1994, но к 1998 году появилась основная версия PHP – 5.4. Язык PHP широко используется программистами для написания сценариев, которые выполняются на серверах при каждом обновлении страницы сайта. PHP действительно похож на язык СИ, и многое он позаимствовал из языка JAVA и технологии JSP. Сегодня PHP используется многими программистами, потому ядром огромного количества сайтов является php-код. Преимущества PHP:

- весь код обрабатывается и исполняется на стороне сервера;
- поддерживает работу с множеством СУБД (MySQL, Oracle, PostgreSQL и т.д.);

- является программным обеспечением с открытым исходным кодом;
- работает на разных платформах (Windows, Linux, Unix подобных)
- PHP очень прост для освоения;
- много возможностей по расширению возможностей языка;
- поддержка различных веб-серверов.

JavaScript [18]– язык программирования, созданный для «оживления и придания динамичности» веб-сайтам. Развитие языка началось с 1996 года. Программы, написанные на языке JavaScript, называются скриптами, которые выполняются совместно с HTML-документами. С помощью JavaScript программисты создают некоторые функции, как например открытие нового окошка с выводом в нем сообщения об ошибке после некоторого действия пользователя. Язык JavaScript способен выполнять свои скрипты спустя заданные интервалы времени. В общем, JavaScript это и самостоятельный язык, но также его можно назвать вспомогательным для остальных, так как с помощью него легко сделать сайт более функциональным и интересным для пользователя. JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам — функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания — что придаёт языку дополнительную гибкость.

Несмотря на схожий с Си синтаксис, JavaScript по сравнению с языком Си имеет коренные отличия [18]:

- объекты, с возможностью интроспекции;
- функции как объекты первого класса;
- автоматическое приведение типов;
- автоматическая сборка мусора;
- анонимные функции.

В языке отсутствуют такие полезные вещи, как:

– модульная система: JavaScript не предоставляет возможности управлять зависимостями и изоляцией областей видимости;

– стандартная библиотека: в частности, отсутствует интерфейс программирования приложений по работе с файловой системой, управлению потоками ввода-вывода, базовых типов для бинарных данных;

– стандартные интерфейсы к веб-серверам и базам данных;

– система управления пакетами, которая бы отслеживала зависимости и автоматически устанавливала их.

Синтаксис языка JavaScript во многом напоминает синтаксис Си и Java, семантически же язык гораздо ближе к Self, Smalltalk или даже Лиспу.

В JavaScript [18]:

– все идентификаторы регистрозависимы,

– в названиях переменных можно использовать буквы, подчёркивание, символ доллара, арабские цифры,

– названия переменных не могут начинаться с цифры,

– для оформления однострочных комментариев используются //, многострочные и внутристрочные комментарии начинаются с /* и заканчиваются */.

Структурно JavaScript можно представить в виде объединения трёх чётко различимых друг от друга частей:

– ядро (ECMAScript),

– объектная модель браузера (Browser Object Model или BOM (en)),

– объектная модель документа (Document Object Model или DOM).

Если рассматривать JavaScript в отличных от браузера окружениях, то объектная модель браузера и объектная модель документа могут не поддерживаться.

Объектную модель документа иногда рассматривают как отдельную от JavaScript сущность, что согласуется с определением DOM как независимого от

языка интерфейса документа. В противоположность этому ряд авторов находят BOM и DOM тесно взаимосвязанными.

ECMAScript не является браузерным языком и в нём не определяются методы ввода и вывода информации. Это, скорее, основа для построения скриптовых языков. Спецификация ECMAScript описывает типы данных, инструкции, ключевые и зарезервированные слова, операторы, объекты, регулярные выражения, не ограничивая авторов производных языков в расширении их новыми составляющими.

Объектная модель браузера — браузер-специфичная часть языка, являющаяся прослойкой между ядром и объектной моделью документа. Основное предназначение объектной модели браузера — управление окнами браузера и обеспечение их взаимодействия. Каждое из окон браузера представляется объектом window, центральным объектом DOM. Объектная модель браузера на данный момент не стандартизирована, однако спецификация находится в разработке WHATWG и W3C.

Помимо управления окнами, в рамках объектной модели браузера, браузерами обычно обеспечивается поддержка следующих сущностей:

- управление фреймами;
- поддержка задержки в исполнении кода и зацикливания с задержкой;
- системные диалоги;
- управление адресом открытой страницы;
- управление информацией о браузере;
- управление информацией о параметрах монитора;
- ограниченное управление историей просмотра страниц;
- поддержка работы с HTTP cookie.

Объектная модель документа — интерфейс программирования приложений для HTML и XML-документов. Согласно DOM, документ (например, веб-страница) может быть представлен в виде дерева объектов,

обладающих рядом свойств, которые позволяют производить с ним различные манипуляции [18]:

- генерация и добавление узлов;
- получение узлов;
- изменение узлов;
- изменение связей между узлами;
- удаление узлов.

Для добавления JavaScript-кода на страницу, можно использовать теги <script></script>, которые рекомендуется, но не обязательно, помещать внутри контейнера <head>. Контейнеров <script> в одном документе может быть сколько угодно. Атрибут «type='text/javascript'» указывать необязательно, так как по умолчанию стоит javascript.

Скрипт, выводящий модальное окно с классической надписью «Hello, World!» внутри браузера:

```
<script type="text/javascript">
  alert('Hello, World!');
</script>
```

Спецификация HTML описывает набор атрибутов, используемых для задания обработчиков событий. Пример использования:

```
<a href="delete.php" onclick="return confirm('Вы уверены?');">
  Удалить
</a>
```

В приведённом примере при нажатии на ссылку функция confirm ('Вы уверены?'); вызывает модальное окно с надписью «Вы уверены?», a return false; блокирует переход по ссылке. Разумеется, этот код будет работать только если в браузере есть и включена поддержка JavaScript, иначе переход по ссылке произойдёт без предупреждения.

Использование кода JavaScript в контексте разметки страницы расценивается в рамках ненавязчивого JavaScript как плохая практика. Аналогом (при условии снабжения ссылки идентификатором alertLink)

```
<a href="delete.php" id="alertLink">
  Удалить
</a>
```

приведённого примера может являться, например, следующий фрагмент JavaScript:

```
window.onload = function() {
  var linkWithAlert = document.getElementById("alertLink");
  linkWithAlert.onclick = function() {
    return confirm('Вы уверены?');
  };
};
```

Есть и третья возможность подключения JavaScript — написать скрипт в отдельном файле, а потом подключить его с помощью конструкции

```
<head>
  <script type="text/javascript" src="http://Путь_к_файлу_со_скриптом">
  </script>
</head>
```

Элемент script, широко используемый для подключения к странице JavaScript, имеет несколько атрибутов [18]:

– обязательный атрибут type для указания MIME-типа содержимого;

– необязательный атрибут src, принимающий в качестве значения адрес к файлу со скриптом;

– необязательный атрибут charset, используемый вместе с src для указания используемой кодировки внешнего файла;

– необязательный атрибут defer указывает, что получение скрипта происходит асинхронно, но выполнение следует отложить до тех пор, пока страница не будет загружена целиком;

– необязательный атрибут async указывает, что получение скрипта происходит асинхронно, а выполнение будет произведено сразу по завершению скачивания. Очередность выполнения скриптов не гарантируется.

jQuery [19]– это библиотека многократно используемых объектов и функции JavaScript, созданная Джоном Резигом и представленная в 2006 году. Обычно jQuery является отдельным JavaScript-файлом. jQuery можно назвать фреймворком (framework), т.е. набором операции и инструкции для решения однотипных задач. Библиотека позволяет вам работать и управлять различными объектами на веб-страницах.

Возможности:

– движок кроссбраузерных CSS-селекторов Sizzle, выделившийся в отдельный проект;

– переход по дереву DOM, включая поддержку XPath как плагина;

– события;

– визуальные эффекты;

– AJAX-дополнения;

– JavaScript-плагины.

Точно так же, как CSS отделяет визуализацию от структуры HTML, JQuery отделяет поведение от структуры HTML. Например, вместо прямого указания на обработчик события нажатия кнопки управление передаётся JQuery, которая идентифицирует кнопки и затем преобразует его в обработчик события клика. Такое разделение поведения и структуры также называется принципом ненавязчивого JavaScript.

Библиотека jQuery содержит функциональность, полезную для максимально широкого круга задач. Тем не менее, разработчиками библиотеки не ставилась задача совмещения в jQuery функций, которые подошли бы всюду,

поскольку это привело бы к большому коду, бóльшая часть которого не востребована. Поэтому была реализована архитектура компактного универсального ядра библиотеки и плагинов. Это позволяет собрать для ресурса именно ту JavaScript-функциональность, которая на нём была бы востребована.

jQuery, как правило, включается в веб-страницу как один внешний JavaScript-файл [19]:

```
<head>
  <script src="jquery-2.1.1.min.js">
  </script>
</head>
```

Вся работа с jQuery ведётся с помощью функции $. Если на сайте применяются другие JavaScript библиотеки, где $ может использоваться для своих нужд, то можно использовать её синоним — jQuery. Второй способ считается более правильным, а чтобы код не получался слишком громоздким, можно писать его следующим образом:

```
jQuery(function($) {
  // здесь код скрипта, где в $ будет находиться объект, предоставляющий доступ к функциям jQuery
})
```

Работу с jQuery можно разделить на 2 типа:

Получение jQuery-объекта с помощью функции $(). Например, передав в неё CSS-селектор, можно получить jQuery-объект всех элементов HTML, попадающих под критерий и далее работать с ними с помощью различных методов jQuery-объекта. В случае, если метод не должен возвращать какого-либо значения, он возвращает ссылку на jQuery объект, что позволяет вести цепочку вызовов методов согласно концепции текучего интерфейса.

Типичный пример манипуляции сразу несколькими узлами DOM заключается в вызове $ функции со строкой селектора CSS, что возвращает

объект jQuery, содержащий некоторое количество элементов HTML-страницы. Эти элементы затем обрабатываются методами jQuery. Например,

$("div.test").add("p.quote").addClass("blue").slideDown("slow");

находит все элементы div с классом test, а также все элементы p с классом quote, и затем добавляет им всем класс blue и визуально плавно спускает вниз. Здесь методы add, addClass и slideDown возвращают ссылку на исходный объект $("div.test"), поэтому возможно вести такую цепочку.

CSS (Cascading Style Sheets- “каскадные таблицы стилей”) [16] – язык программирования, который скорее также относится к языкам разметки и форматирования. CSS стал разрабатываться в 1994 году Хокон Виум Ли и Бертом Босом. Основной задачей было создания языка, который бы форматировал HTML-объекты и текст: работал с шрифтами, цветами, стилями. В общих чертах, CSS работает с внешним видом сайтов. Язык CSS используется с целью «украсить» веб-страницы.

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля.

Правила CSS пишутся на формальном языке CSS и располагаются в таблицах стилей, то есть таблицы стилей содержат в себе правила CSS. Эти

таблицы стилей могут располагаться как в самом веб-документе, внешний вид которого они описывают, так и в отдельных файлах, имеющих формат CSS. (По сути, формат CSS — это обычный текстовый файл. В файле .css не содержится ничего, кроме перечня правил CSS и комментариев к ним.) То есть, эти таблицы стилей могут быть подключены, внедрены в описываемый ими веб-документ четырьмя различными способами [17]:

– когда таблица стилей находится в отдельном файле, она может быть подключена к веб-документу посредством тега <link>, располагающегося в этом документе между тегами <head> и </head>. (Тег <link> будет иметь атрибут href, имеющий значением адрес этой таблицы стилей). Все правила этой таблицы действуют на протяжении всего документа;

```
<!DOCTYPE html>
<html>
  <head>
    .....
    <link rel="stylesheet" href="Style.css">
  </head>
  <body>
    .....
  </body>
</html>
```

– когда таблица стилей находится в отдельном файле, она может быть подключена к веб-документу посредством директивы @import, располагающейся в этом документе между тегами <style> и </style> (которые, в свою очередь, располагаются в этом документе между тегами <head> и </head>) сразу после тега <style>, которая также указывает (в своих скобках, после слова url) на адрес этой таблицы стилей. Все правила этой таблицы действуют на протяжении всего документа [16];

```
<!DOCTYPE html>
<html>
```

```
<head>
  .....
  <style media="all">
    @import url(style.css);
  </style>
</head>
</html>
```

– когда таблица стилей описана в самом документе, она может располагаться в нём между тегами <style> и </style> (которые, в свою очередь, располагаются в этом документе между тегами <head> и </head>). Все правила этой таблицы действуют на протяжении всего документа;

```
<!DOCTYPE html>
<html>
  <head>
    .....
    <style>
      body {
        color: red;
      }
    </style>
  </head>
  <body>
    .....
  </body>
</html>
```

– когда таблица стилей описана в самом документе, она может располагаться в нём в теле какого-то отдельного тега (посредством его атрибута style) этого документа. Все правила этой таблицы действуют только на содержимое этого тега.

```
<!DOCTYPE>
```

```
<html>
  <head>
    .....
  </head>
  <body>
    <p style="font-size: 20px; color: green; font-family: arial, helvetica, sans-serif">
      .....
    </p>
  </body>
</html>
```

В первых двух случаях говорят, что к документу применены внешние таблицы стилей, а во вторых двух случаях — внутренние таблицы стилей.

Для добавления CSS к XML-документу, последний должен содержать специальную ссылку на таблицу стилей. Например [17]:

```
<?xml-stylesheet type="text/css" href="style.css"?>
```

Как известно, HTML-документы строятся на основании иерархии элементов, которая может быть наглядно представлена в древовидной форме. Элементы HTML друг для друга могут быть родительскими, дочерними, элементами-предками, элементами-потомками, сестринскими.

Элемент является родителем другого элемента, если в иерархической структуре документа он находится сразу, непосредственно над этим элементом. Элемент является предком другого элемента, если в иерархической структуре документа он находится где-то выше этого элемента.

В первых трёх случаях подключения таблицы CSS к документу (см. выше) каждое правило CSS из таблицы стилей имеет две основные части — селектор и блок объявлений. Селектор, расположенный в левой части правила, определяет, на какие части документа распространяется правило. Блок объявлений располагается в правой части правила. Он помещается в фигурные скобки, и, в свою очередь, состоит из одного или более объявлений,

разделённых знаком «;». Каждое объявление представляет собой сочетание свойства CSS и значения, разделённых знаком ": ". Селекторы могут группироваться в одной строке через запятую. В таком случае свойство применяется к каждому из них.

```
селектор, селектор {
  свойство: значение;
  свойство: значение;
  свойство: значение;
}
```

В четвёртом случае подключения таблицы CSS к документу (см. список) правило CSS (являющееся значением атрибута style тега, на который оно действует) представляет собой перечень объявлений («свойство CSS : значение»), разделённых знаком «;».

Основное отличие между классами элементов и идентификаторами элементов в том, что в документе какой-нибудь класс может быть присвоен сразу нескольким элементам, а идентификатор — только одному. Также отличие в том, что могут существовать множественные классы (когда класс элемента состоит из нескольких слов, разделённых пробелами). Для идентификаторов такое невозможно.

Важно отметить следующее отличие идентификатора от класса: идентификаторы широко используются в JavaScript для нахождения уникального элемента в документе [17].

Имена классов и идентификаторов, в отличие от названий тегов и их атрибутов, чувствительны к регистру ввода букв.

Свойства классов и идентификаторов задаются с помощью соответствующих селекторов. Причём может быть задано как свойство класса в целом (в таком случае селектор начинается с «.»), или свойство идентификатора самого по себе (в таком случае селектор начинается с «#»), так и свойство какого-нибудь элемента этого класса или с этим идентификатором.

В CSS помимо классов, задаваемых автором страницы, существует также ограниченный набор так называемых псевдоклассов, описывающих вид гиперссылок с определённым состоянием в документе, вид элемента, на котором находится фокус ввода, а также вид элементов, являющихся первыми дочерними элементами других элементов. Также в CSS существует четыре так называемых псевдоэлемента: первая буква, первая строка, применение специальных стилей до и после элемента.

Применение CSS к документам HTML основано на принципах наследования и каскадирования. Принцип наследования заключается в том, что свойства CSS, объявленные для элементов-предков, наследуются элементами потомками. Но, естественно, не все свойства CSS наследуются — например, если для тега параграфа p средствами CSS задана рамка, то она не будет наследоваться ни одним тегом, содержащимся в данном теге <p>. Так сделано в предположении, что обрамление всех-всех вложений в тег - менее тривиальная задача, чем задание одиночной рамки. А вот если для параграфа p средствами CSS задан цвет шрифта, то это свойство будет унаследовано каждым элементом-тегом, находящимся в параграфе, до тех пор, пока этому тегу не будет назначен свой цвет шрифта. Который, в свою очередь, будет теперь наследоваться всеми вложенными в него подэлементами, не распространяясь на элементы-соседи тега.

Принцип каскадирования применяется в случае, когда какому-то элементу HTML одновременно поставлено в соответствие более одного правила CSS, то есть, когда происходит конфликт значений этих правил. Чтобы разрешить такие конфликты, вводятся правила приоритета [16].

Наиболее низким приоритетом обладает стиль браузера.

Следующим по значимости является стиль, заданный пользователем браузера в его настройках.

И наиболее высоким приоритетом обладает стиль, заданный непосредственно автором страницы. И далее, уже в этом авторском стиле приоритеты расставляются следующим образом:

Самым низким приоритетом обладают стили, наследуемые в документе элементом от своих предков.

– Более высоким приоритетом обладают стили, заданные во внешних таблицах стилей, подключённых к документу.

– Ещё более высоким приоритетом обладают стили, заданные непосредственно селекторами всех десяти видов (см. подраздел «виды селекторов»), содержащимися в контейнерах style данного документа. Нередки случаи, когда к какому-нибудь элементу имеют отношение, задают его вид, несколько таких селекторов. Такие конфликты между ними разрешаются с помощью расчёта специфичности каждого такого селектора и применения этих селекторов к данному элементу в порядке убывания их специфичностей.

– Ещё более высоким приоритетом обладают стили, объявленные непосредственно в теге данного элемента посредством атрибута style этого тега.

– И наконец самым высоким приоритетом обладают стили, объявленные автором страницы или пользователем, с помощью сопроводительного слова !important. Если таких свойств несколько, то предпочтение отдаётся в первую очередь стилям, заданным пользователем, а для остальных свойств (которые будут являться задаваемыми автором страницы) потребуется определить их специфичности по принципам, описанным выше, и применять эти свойства в порядке убывания этих их специфичностей.

До появления CSS оформление веб-страниц осуществлялось исключительно средствами HTML, непосредственно внутри содержимого документа. Однако с появлением CSS стало возможным принципиальное разделение содержания и представления документа. За счёт этого нововведения стало возможным лёгкое применение единого стиля оформления для массы схожих документов, а также быстрое изменение этого оформления.

Преимущества [16]:

– несколько дизайнов страницы для разных устройств просмотра. Например, на экране дизайн будет рассчитан на большую ширину, во время печати меню не будет выводиться, а на КПК и сотовом телефоне меню будет следовать за содержимым;

– уменьшение времени загрузки страниц сайта за счет переноса правил представления данных в отдельный CSS-файл. В этом случае браузер загружает только структуру документа и данные, хранимые на странице, а представление этих данных загружается браузером только один раз и может быть закэшировано;

– простота последующего изменения дизайна. Не нужно править каждую страницу, а лишь изменить CSS-файл;

– дополнительные возможности оформления. Например, с помощью CSS-вёрстки можно сделать блок текста, который остальной текст будет обтекать (например для меню) или сделать так, чтобы меню было всегда видно при прокрутке страницы.

Недостатки [17]:

– различное отображение вёрстки в различных браузерах (особенно устаревших), которые по-разному интерпретируют одни и те же данные CSS;

– часто встречающаяся необходимость на практике исправлять не только один CSS-файл, но и теги HTML, которые сложным и ненаглядным способом связаны с селекторами CSS, что иногда сводит на нет простоту применения единых файлов стилей и значительно удлиняет время редактирования и тестирования;

– наиболее полно поддерживающими стандарт CSS являются браузеры, работающие на движках Gecko (Mozilla Firefox и др.), WebKit (Safari, Arora, Google Chrome) и Presto (Opera);

– бывший когда-то самым распространённым браузером Internet Explorer 6 поддерживает CSS далеко не полностью.

– вышедший спустя 7 лет после своего предшественника Internet Explorer 7 хотя и значительно улучшил уровень поддержки CSS, но всё ещё содержит значительное количество ошибок.

– в Internet Explorer 8 используется новый движок, который полностью поддерживает CSS 2.1 и частично — CSS 3.

– для проверки поддержки браузером веб-стандартов (в том числе и различных частей стандарта CSS) был разработан тест Acid. Его вторая версия называется Acid2, а третья, соответственно, Acid3.

Многие веб-мастера для кросс-браузерности стилей не используют нововведения в CSS3, заменяя их изображениями. Например, вместо закругления углов используют фоновое изображение, на котором изображен этот блок без содержания (текста) с закругленными углами.

Различия в реализации CSS различными браузерами заставляют веб-разработчиков искать решения, как заставить все браузеры отображать страницу одинаково. CSS-фильтры позволяют выборочно применять (или не применять) стили к различным элементам. Например, известно, что Internet Explorer 6 применяет правила, использующие селекторы вида * html селектор (фильтр, известный как «star html bug»). Тогда, чтобы заставить и браузеры, использующие блоковую модель W3C и IE, работающего в Quirks mode со своей блоковой моделью, отображать блок #someblock шириной в 100 пикселей и внутренними отступами в 10 пикселей, можно написать такой код:

```
/* Модель W3C - 80px ширина содержимого и 10px отступы с каждой стороны */
#someblock { width: 80px; padding: 10px; }
/* Следующее правило применит только IE6. */
* html #someblock { width: 100px; padding: 10px; }
```

Ещё одним способом выборочного применения правил для Internet Explorer являются условные комментарии[16] [17].

4 РЕАЛИЗАЦИЯ ВЕБИНАРА В ВИДЕ WEB-САЙТА

Web-сайт является наиболее легким и доступным способом для получения необходимой информации. Поэтому для создания вебинара был выбран web-сайт. Разработанный сайт представляет ученикам и преподавателям необходимый доступ к контенту знаний и всех необходимых материалов для изучения дисциплины профессионально-ориентированного иностранного языка (английский).

4.1 Пользование сайтом

Для запуска сайта использовалась портативная серверная платформа Open Server. После запуска сервера переходим на сайт, как представлено на рисунке 15. При переходе на сайт открывается главная страница.

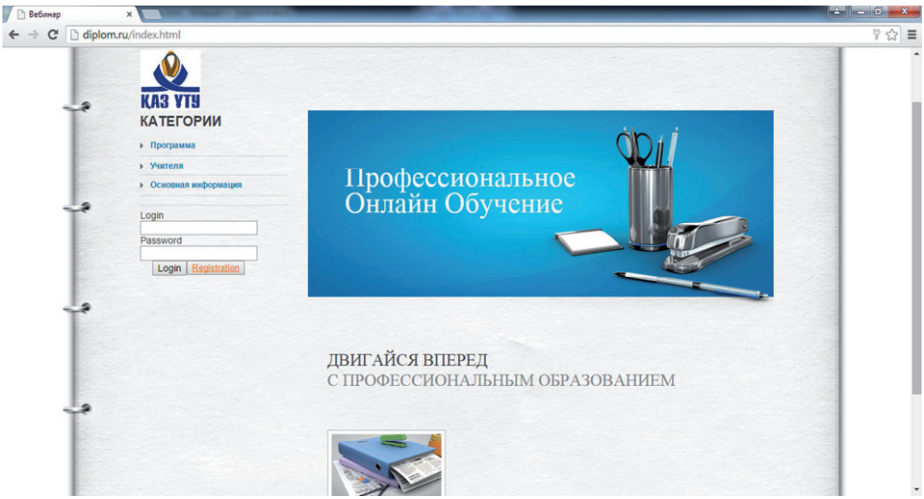

Рисунок 15 – Главная страница сайта

На главной странице сайта можно увидеть цель создания вебинара и основную информацию о находящимся контенте на сайте, как показано на рисунке 15. Также в левом блоке расположена форма авторизации пользователя. В данном блоке содержится ссылка на регистрационную форму. Регистрационная форма позволяет зарегистрироваться в системе новым пользователям.

На странице «О нас» можно увидеть информацию о разработчике, а так же преподавателе данной дисциплины, что можно увидеть на рисунке 16. В данных о преподавателе указана ученая степень, фотография и ФИО. Данные о разработчике показывают ФИО, фотография и название группы в которой он учится.

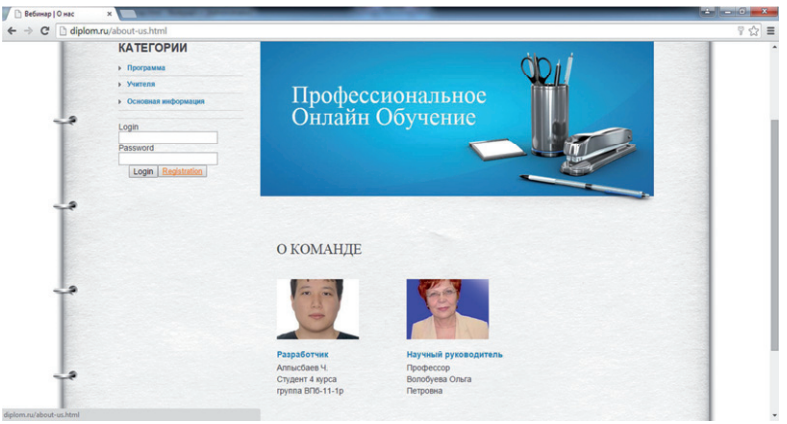

Рисунок 16 – Страница о разработчике и преподавателе.

Страница «Уроки» содержит в себе информацию об уроках за весь курс по дисциплине профессионально-ориентированного иностранного языка (английский), представленного на рисунке 17. Каждый урок расположен на отдельной странице, пример пользования уроком, как продемонстрировано на рисунке 20.

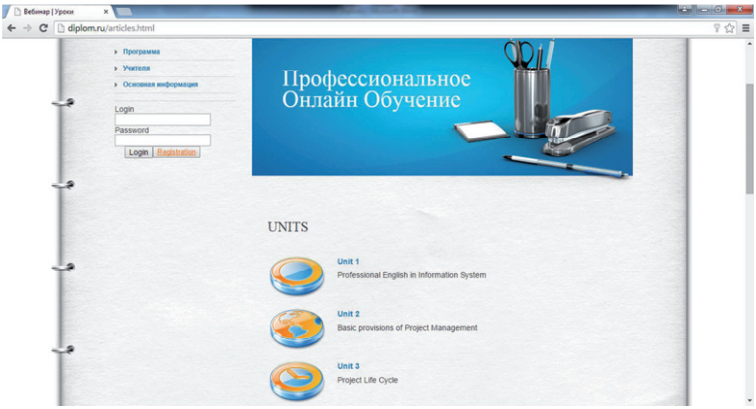

Рисунок 17 – Страница со списком занятий

Страница «Обратной связь» позволяет связаться студентам с преподавателем, либо разработчиком, что показано на рисунке 16. Необходимо заполнить поля с email адресом, именем и написать текстовое сообщение. Также на этой странице указана контактная информация о кафедре и университете.

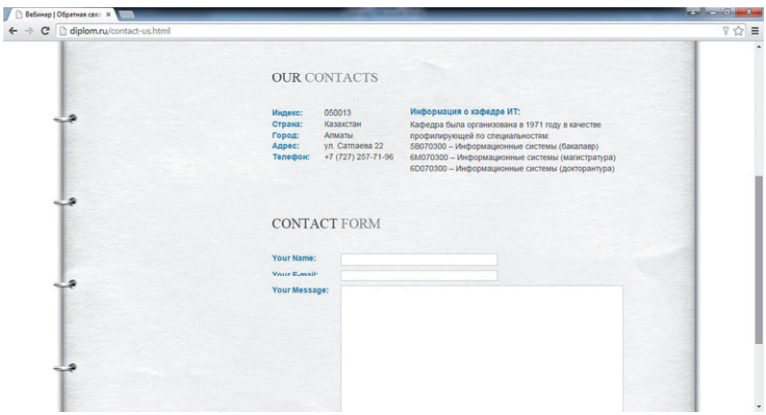

Рисунок 18 – Страница контактной формы

На данной странице находится карта сайта, откуда можно попасть на другие страницы сайта, что видно на рисунке 19.

Рисунок 19 – Карта сайта

4.2 Пользование уроками

Перейдя по одной из ссылок, мы попадаем на страницу одного из занятий, где нам предоставляется контент знаний, необходимый для изучения дисциплины, показано на рисунке 20. На данной странице предоставлены презентация, как показано на рисунке 21, видео, что можно наблюдать на рисунке 22 и аудио, которое представлено на рисунке 23, которое облегчит усвоение материала по данной теме. Также для проверки усвоенного материала в конце странице предложен кроссворд, который представлен на рисунках 24, 25.

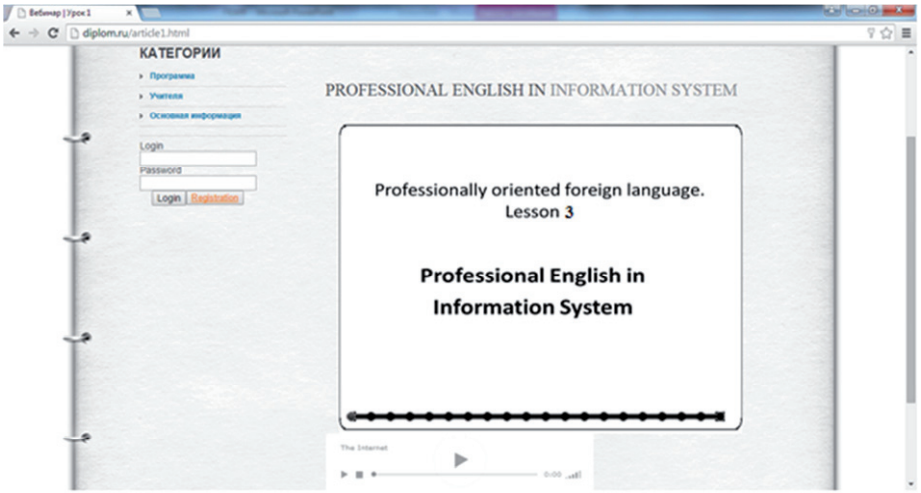

Рисунок 18 – Начало презентации с темы занятия

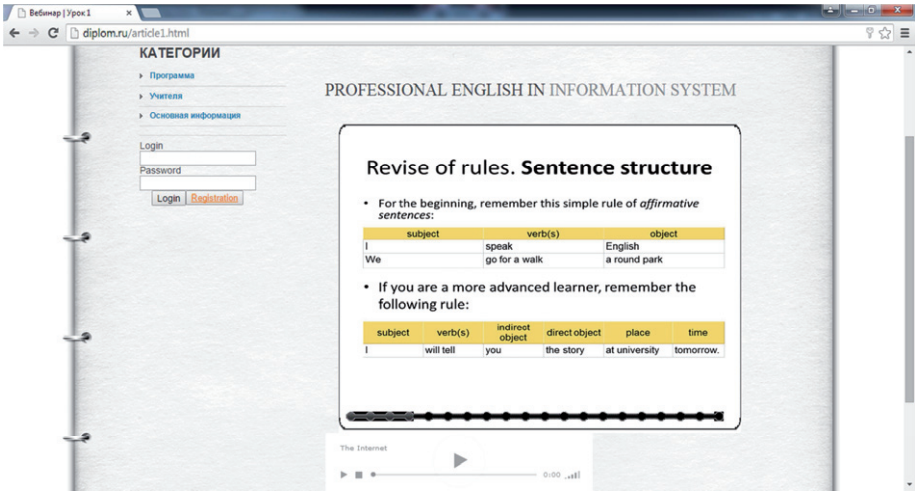

Рисунок 19 – Повторение грамматики английского языка

Рисунок 20 – Использование Video в учебном процессе

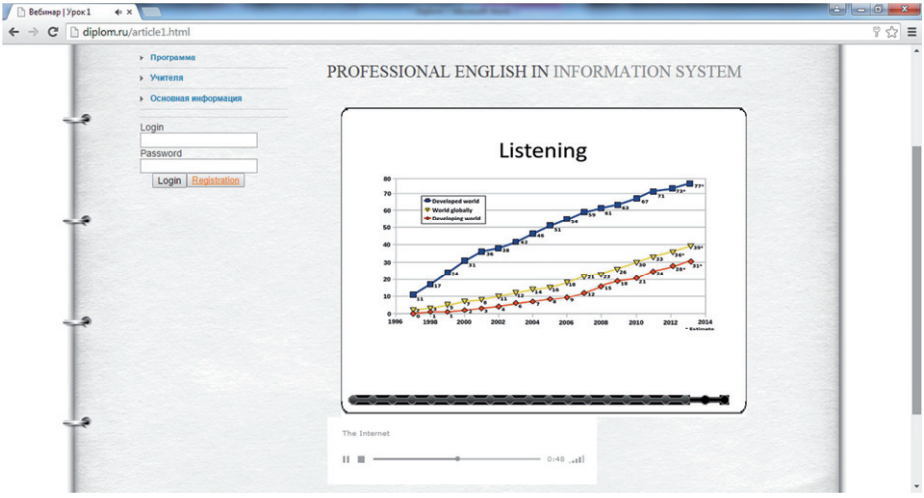

Рисунок 21 – Использование Audio в учебном процессе

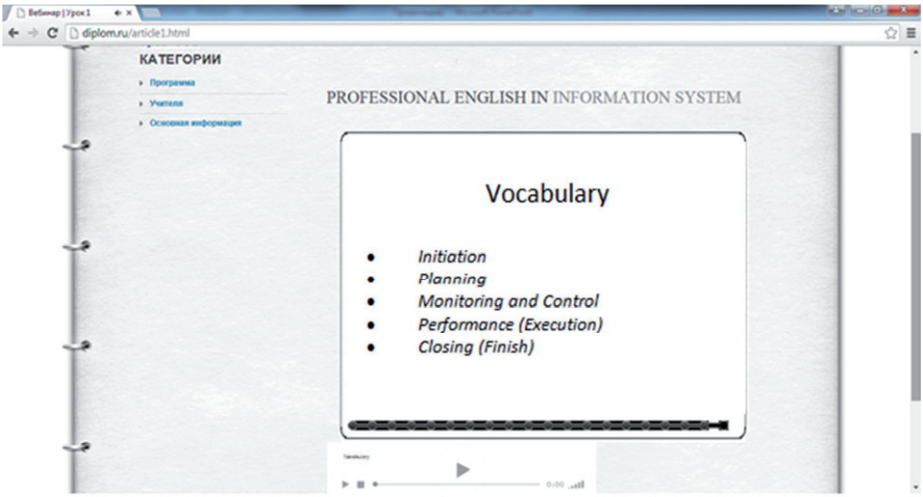

Рисунок 22 – Список слов для запоминания в конце урока

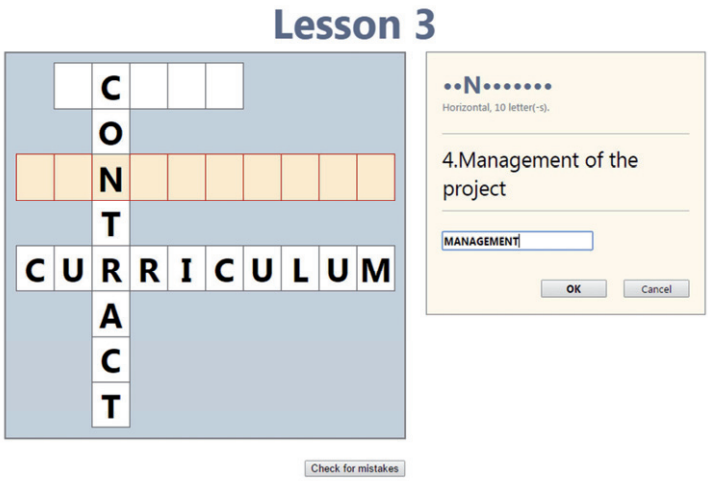

Рисунок 22 – Использование кроссвордов в учебном процессе; заполнение недостающих слов

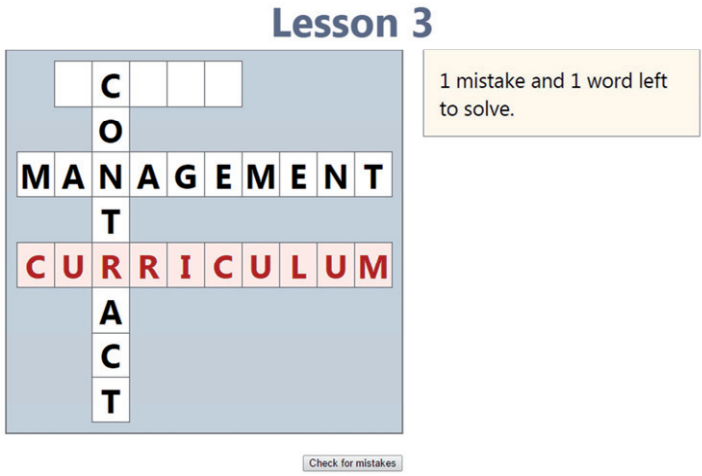

Рисунок 23 – Проверка ошибок

ЗАКЛЮЧЕНИЕ

Сегодня дистанционное обучение переживает период стремительного развития. Все большее количество учебных заведений, компаний, государственных организаций внедряют в учебный процесс технологии дистанционного обучения. Использование вебинаров расширяется. Однако необходимо отметить, что целесообразно использовать вебинары совместно с другими средствами дистанционного обучения. Совместное использование вебинаров с другими средствами дистанционного обучения позволит значительно повысить эффективность дистанционного обучения.

В первой главе приведены функции вебинаров, достоинства и недостатки которые следует учесть при создании вебинара. Все это позволило сформулировать содержательную постановку задачи исследования.

Во второй главе описана основная идея архитектуры «клиент-сервер», которая состоит в разделении сетевого приложения на несколько компонентов, каждый из которых реализует специфический набор сервисов. Компоненты такого приложения могут выполняться на разных компьютерах, выполняя серверные и/или клиентские функции. Это позволяет повысить надежность, безопасность и производительность сетевых приложений и сети в целом. Поэтому в данной работе выбрана клиент-серверная архитектура с реляционной БД, разработанной и представленной в данной главе. Для управления БД использована СУБД MySQL.

В третьей главе представлены языки web-программирования: HTML, PHP, CSS, JavaScript, Jquery, которые использованы для разработки программного обеспечения вебинара. Выбранные языки позволили полностью реализовать необходимые функции для создания вебинара: Video, Audio, кроссворды, презентация слайдов, проведение тренингов. Также описаны разработанные прикладные программы содержащий в себе все функции web-сайта.

В четвертой главе приведена подробная информация для грамотного использования web-сайта. Продемонстрировано пользование уроком, в котором показаны все реализованные функции вебинара.

Разработанный web-сайт позволяет эффективно использовать вебинар для обучения профессионально-ориентированного иностранного языка, в частности английского. Использование вебинара позволит легко осваивать необходимые навыки для дальнейшего саморазвития.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Трайнёв В.А., Трайнёв И.В. Информационные коммуникационные педагогические технологии. – М.: Дашков и К°, 2005. - 240 с.
2. Осин А.В. Мультимедиа в образовании: контекст информатизации. - М.: Издательский сервис, 2004 . - 320 с.
3. Алпысбаев Ч.М., Волобуева О.П. Вебинар для изучения профессионально-ориентированного иностранного языка // Труды «Роль и место молодых ученых в реализации новой экономической политики Казахстана» Международные Сатпаевские чтения. Том IV. – Алматы: КазНТУ, 2015. – 36-40 с.
4. Полат Е.С. Теория и практика дистанционного обучения. - М.: Академия,2004. - 416с.
5. Морозов И.О., Логинова А.Ю. Оценка эффективности обучения в организации. – М.: Академия АйТи, 2006. - 226 с.
6. Александр Барсков AVC, SVC, S-SVC… Что дальше? http://www.osp.ru/bvideo/2013/06/13035947.html
7. Ричард Ньютон. Управление проектами от А до Я. - М.: Альпина Паблишер, 2013. - 140с.
8. Михайлычев Е.А. Дидактическая тестология. - М.: Народное образование, 2001. - 432 с.
9. Светлов Н.М., Светлова Г.Н. Информационные технологии управления проектами. - М.: ФГОУ ВПО РГАУ, 2002. — 144 с.
10. Гаевская А. Преимущество процессного подхода: направленность на результат и применение оптимальных способов его достижения http://www.cfin.ru/itm/project/pmi.shtml
11. Кузнецов С.Д. Основы баз данных.- М.: БИНОМ. Лаборатория знаний, 2002. - 444 с.
12. Дейт К.Дж., Дарвен Хью Основы будущих систем баз данных. М.: Янус-К, 2004. - 656 с.

13. Анатольев А.Г. Компоненты сетевого приложения. Клиент-серверное взаимодействие и роли серверов. http://www.4stud.info/networking/lecture5.html

14. Joost de Valk Структура сайта и SEO. https://yoast.com/site-structure-seo/

15. Батоврин В. К. Толковый словарь по системной и программной инженерии. - М.: ДМК Пресс, 2012. - 280 с.

16. Фримен Эрик, Фримен Элизабет Изучаем HTML, XHTML и CSS = Head First HTML with CSS & XHTML. - 1-е изд. - М.: «Питер», 2010. – 656 с.

17. Стивен Шафер. HTML, XHTML и CSS. Библия пользователя, 5-е издание = HTML, XHTML, and CSS Bible, 5th Edition. - М.:Диалектика, 2010. - 656 с.

18. Рейсиг Д. Инструменты отладки и тестирования // JavaScript. Профессиональные приёмы программирования = Pro JavaScript™ Techniques / Перевод Н. Вильчинский. — СПб.: Питер, 2008. - 706 с.

19. Бер Бибо, Иегуда Кац. Расширение jQuery // jQuery. Подробное руководство по продвинутому JavaScript = jQuery in Action. - Спб.: Символ-Плюс, 2009. - 384 с.