

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий

(наименование института полностью)

Кафедра «Прикладная математика и информатика»

(наименование)

09.04.03 Прикладная информатика

(код и наименование направления подготовки)

Информационные системы и технологии корпоративного управления

(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему База знаний системы поддержки принятия решений IT-компании ООО
«Директ Лайн»

Студент

Е.А. Дорош

(И.О. Фамилия)

(личная подпись)

Научный
руководитель

канд. пед. наук, доцент, О.М. Гущина

(ученая степень, звание, И.О. Фамилия)

Тольятти 2021

Содержание

Введение.....	3
1 Анализ предметной области	6
1.1 Базы знаний в экспертных системах	6
1.2 Характеристика компании ООО «Директ Лайн»	10
1.3 Описание процесса принятия решений	19
1.4 Моделирование процесса принятия решений.....	31
1.5 Обзор систем поддержки принятия решений	34
1.6 Требования к информационной системе	37
2. Проектирование базы знаний информационной системы поддержки принятия решений.....	40
2.2 Архитектура и структура базы знаний	44
2.3 СУБД для реализации проекта	48
2.4 Язык программирования для реализации проекта базы знаний	51
2.5 Среда разработки базы знаний	55
2.6 Проектирование функционала и компонентов базы знаний	56
2.7 Проектирование базы данных.....	69
3. Реализация базы знаний информационной системы поддержки принятия решений	72
3.1 Физическая модель базы данных	72
3.2 Реализация модуля базы знаний информационной системы поддержки принятия решений.....	76
3.3 Тестирование созданного приложения.....	83
Заключение	88
Список используемой литературы	90
Приложения А Создание таблиц	96
Приложения Б Основной модуль программы	98

Введение

На сегодняшний день для решения задач повседневной управленческой деятельности применяются максимально адаптированные системы поддержки принятия решений. Они предназначены для помощи лицам, принимающим решения.

Задача принятия решений возникает, когда присутствует несколько вариантов действий (альтернатив) для достижения, заданного или желаемого результата, при этом требуется выбрать наилучшую в определенном смысле альтернативу. Помощь лицам, осуществляющим решения в трудных обстоятельствах для беспристрастного и полного анализа предметной деятельности – цель компьютерной автоматизированной системы поддержки принятия решений.

В общем виде — это совокупность эвристических и математических моделей и методов, связанных единой методикой выработки альтернатив управленческих решений в организационных системах, поиска результатов реализации любой альтернативы и подбор обоснованного максимально подходящего управленческого решения, или комплекс процедур обработки суждений и данных, для помощи начальнику в принятии решений [6, 9]. Интерактивные автоматизированные системы, помогающие людям принимать решения, применяют модели и данные, чтобы решить неструктурированные проблемы и как информационные компьютерные системы, применяемые для поддержания всевозможных разновидностей деятельности во время принятия решений при моментах, в которых необходим контроль человека над процессом определения наилучшего решения, так определяются СППР в работах западных ученых [14, 16].

Высокий интерес, который возник к таким системам, обусловлен тем, что их использование снижает сложность в процессе принятии решений, дает точность в оценке различных альтернатив их влияние.

Системы поддержки принятия решений появились как естественное развитие и обобщение информационных систем управления и систем управления баз данных (СУБД) к их приспособленности и более подходящих для решения повседневных задач управленческой деятельности. В большинстве случаев интерактивные автоматизированные системы, помогающие пользователю применять модели и данные для поиска и решения проблем и принятия решений являются системами поддержки принятия решений. Работа системы должна осуществляться на простом для изучения языке для работы с запросами в интерактивном режиме

Основная задача при принятии решения - выбрать лучший вариант для достижения определенной цели или ранжировать набор возможных вариантов по степени их влияния на достижение этой цели. Следующие задачи по принятию решений - найти критерии для оценки альтернатив и преодоления множества критериев. Наконец, сама задача выбора и реализации решений. Помогают принимать решения экспертные системы, которые включают в свой состав базу знаний. База знаний занимает центральное место среди инструментов управления знаниями организации, поскольку позволяет сохранить весь спектр ценных знаний организации и использовать их в последующем.

Изучали вопросы строения и функционирования экспертных систем поддержки принятия решений и баз знаний множество отечественных и зарубежных ученых, среди которых: Т. К. Кравченко, И. Т. Давыденко, Н. С. Кузнецов, И. А. Идимечев.

Целью данной работы является разработка базы знаний системы поддержки принятия решений ИТ-компании ООО «Директ Лайн». В соответствии с поставленной целью были поставлены следующие задачи:

- провести анализ предметной области и проанализировать деятельность ООО «Директ Лайн»;
- разработать проект базы знаний информационной системы поддержки принятия решений;

- реализовать базу знаний для информационной системы поддержки принятия решений ООО «Директ Лайн»

- протестировать работу разработанной системы и оценить ее работоспособность для компании.

Объект работы: система поддержки принятия решений ООО «Директ Лайн».

Предмет работы: база знаний системы поддержки принятия решений ООО «Директ Лайн».

Для решения поставленных в диссертационной работе задач используются следующие методы исследования: абстрактно-логический метод; анализ научной и учебной литературы; классификация; метод обобщения; методы проектирования и разработки веб-приложений; моделирование; описательный метод; проектный метод; синтез; аналогия; дедукция; системный анализ и подход; сравнительный анализ; формализация; тестирование; эмпирический метод.

Основные положения, выносимые на защиту:

- База знаний обеспечивает пользователей (сотрудников организации) требуемыми данными о предметной области, что позволяет эффективно решать вопросы функционирования предприятия.

- Проект базы знаний системы поддержки принятия решений содержит программные компоненты, обеспечивающие управление данными предметной области, обеспечивает процесс добавления данных экспертами и процесс использования данных сотрудниками организации.

- База знаний системы поддержки принятия решений направлена на совершенствование функционирования предприятия за счет накопления знаний о предметной области.

Структура магистерской диссертации. Работа состоит из введения, 3 глав, заключения, содержит 29 рисунков, 3 таблиц, список использованной литературы, 2 приложений. Основной текст работы изложен на 105 страницах.

1 Анализ предметной области

1.1 Базы знаний в экспертных системах

Качественная информация является знанием, к знанию не относится предполагаемая информация, так как она имеет большой шанс оказаться некачественной информацией.

Существуют такие формы знаний как знания и метазнания (различная глубина сопоставлений). В результате сравнения различных знаний образуются метазнания. Метазнания отличаются глубиной сопоставления, при сопоставлении метазнаний результатом будут метазнания и т.д.

Ценность – главное свойство знания, определяющее на своей основе, важность принимаемых решений. Иерархия целей, для достижения которых ведется поиск решений, влияет на важность принятия решений. От того, какие цели имеют первоочередную значимость в данных случаях, зависит ценность знаний в различные промежутки времени.

Существует два вида ценности знаний – потенциальная и непосредственная. Применение знаний для принятия решений и создания новых знаний в будущем - потенциальная ценность. Использование знаний при принятии решений и создание знаний в настоящий момент – непосредственная ценность.

Также отметим, что «Совокупность знаний из некоторой предметной области и формально представленных таким образом, чтоб на их основе можно было осуществлять рассуждения является базой знаний. Это, особого рода база данных, содержащая информацию о человеческом опыте и знаниях в некоторой предметной области и созданная для управления этими знаниями, их сбора, хранения, поиска и выдачи. Используются базы знаний в процессах поддержки принятия решений» [52].

Базы знаний чаще всего используются в контексте экспертных систем, где с их помощью представляются навыки и опыт экспертов, занятых

практической деятельностью в соответствующей области – деятельности ИТ-компании. Под совокупностью правил ввода и фактов, образующих логический вывод и осмысленную обработку информации, подразумевают базу знаний.

Описание баз знаний, как правило, выглядит в «форме конкретных фактов и правил логического вывода над БД и процедурами обработки данных, которые предоставляют информацию и знания о людях, предметах, фактах событиях и процессах в логической форме» [52].

Для хранения данных организации: статей технического обеспечения, руководств, документации и создания экспертных систем применяются простые базы знаний. Помощь в процессах принятия решений является основной целью разработки таких баз.

Программа, обладающая способностью частично заменить специалиста-эксперта при принятии решения в проблемной ситуации, является экспертной системой

«Экспертные системы в информатике анализируются вместе с базами знаний в виде моделей действий специалистов в отдельной области знаний с применением процедур логического вывода и принятия решений, а базы знаний – совокупность правил и фактов логического вывода в выбранной предметной области деятельности» [6].

Из двух основных частей состоит архитектура экспертной системы: база знаний; программный инструмент доступа и обработки знаний, который состоит из механизмов приобретения знаний, вывода решения, объяснения получаемых результатов и интеллектуального интерфейса (рис. 1).

Характерными чертами экспертной системы являются [52]:

- алгоритм решения не описывается заранее, а строится самой экспертной системой;
- ориентация на решение неформализованных (способ формализации пока неизвестен) задач;

- отсутствие гарантии нахождения оптимального решения с возможностью учиться на ошибках.
- результат выдается в виде конкретных рекомендаций для действий в сложившейся ситуации, не уступающих решениям специалистов;
- способность объяснять ход и результат решения понятным для пользователей способом;
- способность пополнять базу знаний, возможность наращивания системы;
- способность принимать решения в условиях неопределенности;
- четкая ограниченность предметной области;
- четкое разделение декларативных и процедурных знаний (фактов и механизмов вывода).

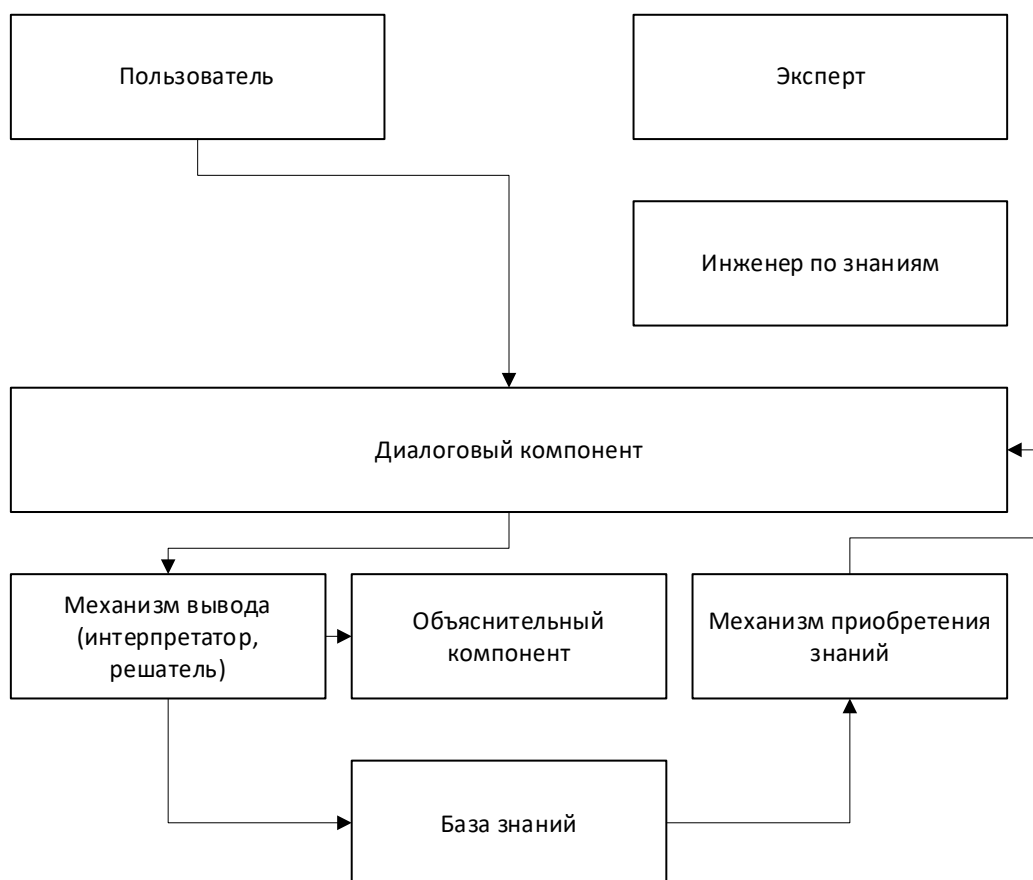


Рисунок 1 – Архитектура экспертной системы

Пользовательские правила обработки информации отдельной проблемы составляют базу знаний.

Основной целью базы знаний выступает сокращение времени и трудозатрат на решение задач, которые входят в компетенцию определенных специалистов, преимущественно выполняющих руководящую роль.

Отметим следующие определения понятия база знаний: «это особого рода база данных, содержащая информацию о человеческом опыте и знаниях в некоторой предметной области и созданная для управления этими знаниями, их сбора, хранения, поиска и выдачи».

Еще одно определение гласит, что «под базами знаний понимает совокупность фактов и правил вывода, допускающих логический вывод и осмысленную обработку информации. Базы знаний описываются в форме конкретных фактов и правил логического вывода над базами данных и процедурами обработки информации, представляющих сведения и знания о людях, предметах, фактах событиях и процессах в логической форме» [52].

По мнению Н.А. Гулякиной, «к достоинствам интеллектуальной справочной системы, разрабатываемой на основе предложенной технологии, можно отнести возможность богатого разнообразия представляемых знаний в ее базе (теоретико-множественные связи между понятиями, терминологическое описание понятий, логическая иерархия понятий, аксиоматизация предметной области, описание утверждений различного рода, а также доказательств, описание задач и способов их решений, когнитивные иллюстрации и др.), способность системы отвечать на большое число вопросов пользователя (при этом учитывается полнота ответа на вопрос), при отсутствии ответа в базе знаний система пытается ответить на него с помощью решателя задач» [53].

Понятие базы знаний тесно связано с понятием предметной области (рис. 2). Семантика базы знаний интеллектуальной системы — это соотношение между базой знаний и описываемой ею предметной областью

[3]. База знаний может быть представлена набором правил, на основе которых алгоритм логического вывода определяет выходные данные.

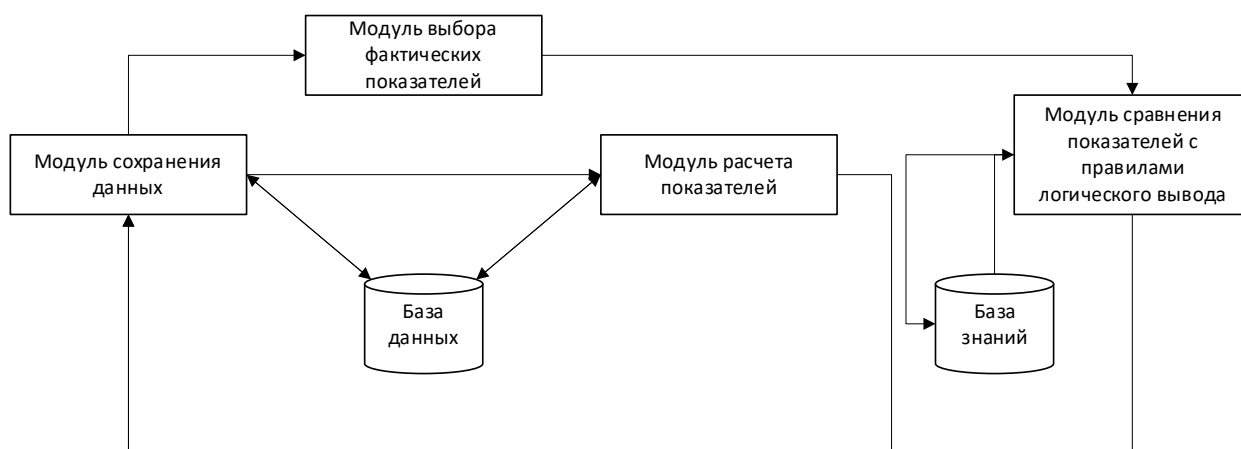


Рисунок 2 – Схема модели системы поддержки принятия решений

Также Н.А. Гулякина отмечает, что «семантическая структура базы знаний интеллектуальной системы трактуется в рамках семантической технологии проектирования баз знаний интеллектуальных систем как иерархическая система взаимосвязанных между собой предметных областей, которые представляются в базе знаний» [53].

1.2 Характеристика компании ООО «Директ Лайн»

Общество с ограниченной ответственностью «Директ Лайн» - является агентством по оказанию комплексных услуг, по продвижению бизнеса, а в частности: разработка и поддержка сайтов, реклама в интернете, внедрение Битрикс24, обучение в сферах бизнеса, предпринимательства, а также по разработке сайтов. Юридический адрес: 445044, Самарская обл., г. Тольятти, проспект Степана Разина, д.44.

Согласно характеристике предприятия ООО «Директ Лайн» - «компания является коммерческой организацией, созданной в организационно-правовой

форме общества с ограниченной ответственностью, в соответствии с действующим законодательством Российской Федерации. Компания является юридическим лицом и строит свою деятельность на основании действующего Устава и действующего законодательства Российской Федерации: Конституции Российской Федерации, Гражданского Кодекса, Трудового Кодекса и Налогового Кодекса. Основной целью организации является получение прибыли в интересах акционеров общества. Устав является учредительным документом общества с ограниченной ответственностью.

Общество вправе в установленном порядке открывать банковские счета на территории Российской Федерации и за ее пределами. Общество имеет круглую печать, содержащую его полное фирменное наименование на русском языке, а также указание на его место нахождения. Общество вправе иметь штампы и бланки со своим наименованием, собственную эмблему и другие средства визуальной идентификации.

ООО «Директ Лайн» отвечает по своим обязательствам всем принадлежащим ему на правах собственности имуществом. Участник имеет предусмотренные законом и учредительными документами Общества обязательственные права по отношению к Обществу».

Основными целями деятельности Общества являются расширение рынка товаров и услуг, а также извлечение прибыли. Для достижения целей Общество осуществляет в установленном законодательством Российской Федерации порядке следующие виды деятельности, не запрещенные законодательством Российской Федерации:

- 62.01 - Разработка компьютерного программного обеспечения (основной вид деятельности);
- 62.09 - Деятельность, связанная с использованием вычислительной техники и информационных технологий, прочая;
- 63.11 - Деятельность по обработке данных, предоставление услуг по размещению информации и связанная с этим деятельность;
- 63.91 - Деятельность информационных агентств.

Общество вправе заниматься и другими видами деятельности, не запрещенными законом. Отдельными видами деятельности, перечень которых определяется законом, общество может заниматься только на основании специального разрешения (лицензии). Право общества осуществлять деятельность, на занятие которой необходимо получение лицензии, возникает с момента получения такой лицензии или в указанный в ней срок и прекращается по истечении срока ее действия, если иное не установлено законом или иными правовыми актами.

Органами управления ООО «ДИРЕКТ ЛАЙН» являются:

- общее собрание акционеров;
- совет директоров;
- генеральный директор (единоличный исполнительный орган).

Контроль за финансово-хозяйственной деятельностью общества осуществляет ревизионная комиссия.

Высшим органом управления общества является общее собрание акционеров. К основным компетенциям общего собрания акционеров относятся следующие вопросы:

- внесение изменений и дополнений в Устав Общества или утверждение Устава Общества в новой редакции;
- образование единоличного исполнительного органа Общества, а также досрочное прекращение его полномочий;
- принятие решений об одобрении сделок в случаях, предусмотренных статьей 83 Федерального закона «Об акционерных обществах»;
- утверждение внутренних документов, регулирующих деятельность органов общества;
- утверждение годовых отчетов и годовой бухгалтерской отчетности.

Вопросы, отнесенные к компетенции общего собрания акционеров, не могут быть переданы на решение Совету директоров и исполнительному органу Общества.

Общее собрание акционеров не вправе рассматривать и принимать решения по вопросам, не отнесенным к его компетенции.

Решение общего собрания акционеров по вопросу, поставленному на голосование, принимается большинством голосов акционеров - владельцев голосующих акций Общества, если иное не установлено законодательством Российской Федерации.

Общество обязано ежегодно проводить годовое общее собрание акционеров. Годовое общее собрание акционеров проводится ежегодно не ранее чем через два месяца и не позднее чем через шесть месяцев после окончания финансового года.

Совет директоров Общества осуществляет общее руководство деятельностью Общества, за исключением решения вопросов, отнесенных к компетенции общего собрания акционеров. Совет директоров Общества состоит из 5 человек.

Руководство текущей деятельностью Общества осуществляется Генеральным директором Общества (единоличный исполнительный орган), который подотчетен Совету директоров и общему собранию акционеров Общества.

По решению общего собрания акционеров полномочия единоличного исполнительного органа Общества могут быть переданы по договору коммерческой организации (управляющей организации). Решение о передаче полномочий единоличного исполнительного органа Общества управляющей организации принимается общим собранием акционеров по предложению Совета директоров Общества.

Совет директоров вправе принять решение о приостановлении полномочий Генерального директора Общества, управляющей организации, образовав временный единоличный исполнительный орган Общества, и о проведении внеочередного общего собрания акционеров для решения вопроса о досрочном прекращении полномочий Генерального директора Общества или управляющей организации и обо образовании нового единоличного

исполнительного органа Общества или о передаче полномочий единоличного исполнительного органа Общества управляющей организации.

Временный исполнительный орган Общества осуществляет руководство текущей деятельностью Общества в пределах компетенции исполнительного органа Общества.

Генеральный директор Общества избирается общим собранием акционеров сроком на 5 лет.

Генеральный директор Общества:

- выдает доверенности от имени Общества;
- заключает договоры и совершает иные сделки, в порядке, предусмотренном Федеральным законом «Об акционерных обществах» и настоящим Уставом;
- обеспечивает организацию и планирование работы подразделений, филиалов и представительств Общества, осуществляет контроль за их деятельностью;
- определяет организационную структуру Общества, утверждает штатное расписание Общества, а также его филиалов и представительств;
- применяет к работникам меры поощрения и налагает на них взыскания в порядке и на условиях, предусмотренных действующим законодательством о труде, а также внутренними документами Общества.

Схема организационной структуры предприятия ООО «ДИРЕКТ ЛАЙН» представлена на рисунке 3. Из рисунка видно, что органами управления общества является: общее собрание, совет директоров и генеральный директор.

Генеральный директор без доверенности действует от имени Общества. Генеральный директор определяет позицию Общества (представителей Общества) по вопросам повестки дня общего собрания акционеров (участников) и заседания совета директоров дочерних и зависимых обществ за исключением случаев, когда в соответствии с уставом Общества такие полномочия отнесены к компетенции Совета директоров Общества.

Во главе каждого подразделения стоит один руководитель, в чьих руках сосредоточены все функции управления. Он осуществляет единоличное руководство над подчиненными ему сотрудниками компании. Приказы, которые он отдает, обязательны для выполнения нижестоящими звеньями. Каждый руководитель, в свою очередь, подчиняется директору.

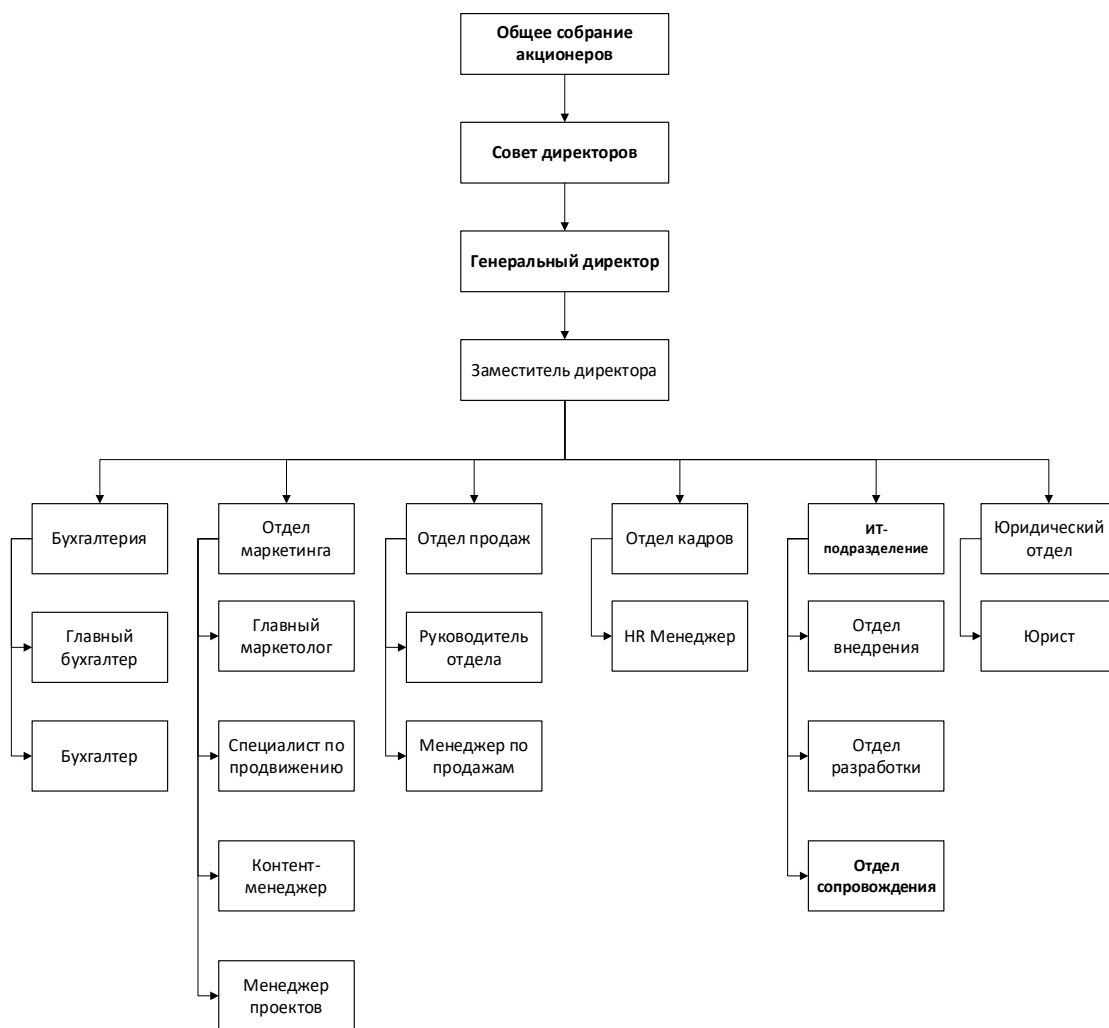


Рисунок 3 – Организационная структура предприятия ООО «Директ Лайн»

Централизующим звеном в последовательности продаж компании является её менеджер, обычно это стресс устойчивый человек с высшим образованием, который легко находит общий язык с каждым клиентом и обладающий большим творческим потенциалом, его непосредственным начальником является коммерческий директор.

Ответственным за финансовую деятельность организации выступает главный бухгалтер, его главная задача – соблюдение норм законов в рамках бухгалтерской деятельности. На должность главного бухгалтера может претендовать человек с аналитическим складом ума и высшим образованием в сфере экономики, его работа состоит в ведении бухучета имущества, обязательств и хозяйственных операций, разработки и реализации процесса соблюдения рационального использования ресурсов и финансовой дисциплины.

Основные обязанности главного бухгалтера:

- Принимает и контролирует соответствие документацией участкам бухгалтерского учета, а также готовит их к счетной обработке;
- Выводит в счета бухучета, направленные денежных средств;
- Подготавливает отчетный калькулятор, измеряющий себестоимость продукции, определяет несообразные траты и план их дальнейшего недопущения;
- Занимается введением налогового документооборота, перечисляет налогу во все инстанции, начисляет заработную плату и иные меры поддержки работникам.

Руководителем отдела могут стать лица с высшим образованием в соответствующем отделе направлении, которые хорошо идут на контакт, готовы быть ответственными за свою и чужую работу, готовые проявлять креатив, и быстро находить решению по любым задачам, для правильной организации рабочего процесса в своем отделе.

За все вопросы закупки, маркетинга и продаж, а также формирования их планов, несет ответственность коммерческий директор.

Ответственным за руководства отделом продаж является начальник отдела, обязанности которого сводятся к контролю и координированию продаж в фирме, а также за разработку должностных инструкций для всех работников своего отдела, разработку мероприятий, направленных на расширение рынка сбыта и повышению продаж. Помимо всего этого, он

занимается формированием планов и проведением обучения сотрудников отдела.

Главным для всех сотрудников профессиональным качеством является компетентность и вежливость, которыми должны обладать все сотрудники компании, потому как работа сотрудника требует хорошего понимания потребностей клиента, а также их качественное и оперативное обслуживание.

Именно такой подход приводит к повышению уровню товаров и привлечению новых клиентов, для дальнейшего сотрудничества.

При возникновении жалоб покупателей, которые связаны с неудовлетворительным качеством товара, условиями транспортировки, отгрузки и т. д., устраняются в минимальные сроки.

Системный администратор отдела технического обеспечения решает текущие и перспективные задачи организации, связанные с ИТ. Осуществляет оперативную поддержку пользователей. Обеспечивает работоспособность или быструю замену техники. Разрабатывает предложения по формированию планов перспективного развития ИТ в организации.

При организации работ по технической поддержке структурных подразделений, а также для обеспечения их эффективной работы в локальной электронной сети специалисты отдела технического обеспечения решают следующие задачи:

- выполняют работы по обслуживанию пользователей локальной электронной сети;
- контролируют работу оргтехники общего пользования (принтеры, копировальная техника) и принимает меры по ее ремонту и снабжению расходными материалами;
- выполняют работы по проведению анализа и обобщению потребностей структурных подразделений в компьютерной, копировально-множительной и другой электронной оргтехнике;

- разрабатывают предложения о качестве, количестве, составе и конфигурации техники, необходимой для автоматизации работы структурных подразделений организации, и вынесение их на рассмотрение руководства;
- обеспечивают компьютерами и копировально-множительной техникой рабочие места сотрудников в установленном порядке;
- организуют выполнение работ по техническому обслуживанию компьютерного оборудования и копировально-множительной техники в установленном порядке;
- выполняют работы по модернизации компьютерного парка;
- оказывают техническую помощь и консультации сотрудников по вопросам, связанным с использованием ИТ;
- обеспечивают подключение телефонного оборудования конечным пользователям;
- сопровождают программные продукты, используемые в организации.

Юрист компании осуществляет разработку учредительных документов, обеспечивает регистрацию юридических лиц, внесение изменений в учредительные документы, определяет правовые основы органов предприятия; разрабатывает положение о сделках, связанных с приобретением или отчуждением имущества. Обеспечивает подразделения предприятия, необходимыми для осуществления ими своих функций и обязанностей. Также юрист представляет интересы предприятия при проверках, проводимых на предприятии государственными контрольно-надзорными органами с целью правового контроля за соблюдением процессуальных действий проверяющими, обоснованностью и правильностью выводов проверяющих, оформлением результатов проверок и составлением процессуальных документов.

Служба по внедрению выполняет функции по первичной настройке нового программного обеспечения (ПО), первичному обучению

пользователей работе с новым ПО, проведению анализа данных, созданию файлов для загрузки данных в новое программное обеспечение.

Служба разработки выполняет задачи по созданию нового программного обеспечения на каждой стадии внедрения, доработки системы в условиях требований пользователей на стадии сопровождения системы, поддержке системы в рабочем состоянии, оптимизации процессов функционирования системы.

Служба сопровождения выполняет поддержку пользователей, консультирование по функциональным возможностям программы, написание технических заданий и выполнение критических операций по работе в системе.

1.3 Описание процесса принятия решений

1.3.1 Обзор и понятие системы принятия решений

Все процессы функционирования современного предприятия, от разработки продукта до его продажи, тесно взаимосвязаны и требуют четкого централизованного управления. Согласно мнению П.В Терелянского, «основные решения, принимаемые на уровне руководителя предприятия, невозможно реализовать без развитой информационной инфраструктуры. Создание информационно-управляющих систем (ИСМ) предприятия позволяет оптимизировать существующие каналы сбора информации и обеспечить более полное удовлетворение информационных потребностей менеджеров и коллектива в целом. Существующая в настоящее время системная и техническая инфраструктуры большинства предприятий в той или иной степени обеспечивают только определенные виды производственно-хозяйственной, финансово-хозяйственной деятельности и управления. ИСМ — это сложная многоуровневая информационная система, которая гарантирует автоматизированный контроль всех подсистем системы управления и деятельности предприятия» [50].

Создание и внедрение DSS в АСУ ТП компании требует поэтапной разработки и развития ряда всех вспомогательных подсистем DSS: технической, математической, программной, информационной и организационной поддержки.

Также отметим, что «процессы принятия решений имеют центральное значение для комплексного организационного управления промышленной компанией. Цикл управления организацией начинается с этапа получения задач с более высокого уровня по информационным каналам, понимания поставленных задач и оценки возможности их выполнения в соответствии с имеющимися ресурсами и условиями» [46].

Понятие «Система поддержки принятия решений характеризуется как это компьютерная автоматизированная система, цель которой - помочь людям, принимающим решения в сложных условиях, для полного и объективного анализа основной деятельности. DSS возникла в результате слияния информационных систем управления и систем управления базами данных» [42].

DSS включает в себя следующие этапы: анализ ситуаций и постановка проблем, формирование и выбор вариантов решений, организация реализации решений, контроль выполнения решений.

В зависимости от типа принимаемых решений уровни DSS подразделяются: оперативный, тактический и стратегический.

Оперативный уровень обеспечивает решение повторяющихся задач и операций на коротком временном интервале (неделя, декада, месяц и т. д.). На этом уровне велики как объем выполняемых операций, так и динамика принятия управленческих решений. Тактический уровень обеспечивает решение задач, требующих предварительного анализа информации, подготовленной на первом уровне.

Тактические решения принимаются на более длительный период (квартал, полгода и т. д.). На этом уровне количество решаемых задач уменьшается, но увеличивается их сложность. Стратегический уровень

обеспечивает разработку решений, направленных на достижение долгосрочных стратегических целей организации. Этот тип решения характеризуется длительным периодом времени (годы, несколько лет и т. д.), а сфера применения — это весь объект, управляемый как единое целое (предприятие, межорганизационный комплекс и т. д.). DSS характеризуется следующими отличительными особенностями:

Ориентация на решение плохо структурированных (формализованных) задач, характерных в основном для высоких управленческих уровней;

Умение сочетать традиционные методы доступа и обработки компьютерных данных с возможностями математических моделей и методов решения задач на их основе.

Нацелено на непрофессионального конечного пользователя компьютера, использующего интерактивный режим работы.

Термин «система поддержки принятия решений» или «система принятия решений» появился в начале 1970-х годов, однако на сегодняшний день он не нашел общепризнанного определения ни учеными, ни разработчиками» [42].

Достаточное количество работ отечественных и зарубежных специалистов в различных предметных областях посвящено использованию DSS и определению их функционального назначения.

К примеру, Д. Ж. Пауер DSS описывает «как инструмент для «расчета решений», основанный на «использовании моделей серии процедур обработки данных и суждений, которые помогают лицу, принимающему решения, принять решение» [54].

DSS также определяется как «компьютерная информационная система, «используемая для поддержки различных действий по принятию решений в ситуациях, когда невозможно или нежелательно иметь автоматические системы, которые полностью выполняют весь процесс принятия решений. DSS не заменяет лицо, принимающее решения, автоматизирует процесс принятия решений, но помогает ему в решении задачи» [41].

Система поддержки принятия решений (DSS) — это «компьютеризированная система, цель которой - помочь людям, принимающим решения в сложных условиях, провести полный и объективный анализ фоновой деятельности. DSS возникла в результате слияния информационных систем управления и систем управления базами данных».

Также Д.Ж. Пауер отмечает, что «для анализа и разработки предложений в DSS используются различные методы. Это могут быть: поиск информации, интеллектуальный анализ данных, поиск знаний в базах данных, рассуждения на основе прецедентов, имитационное моделирование, эволюционные вычисления и генетические алгоритмы, нейронные сети, анализ ситуаций, когнитивное моделирование и т. д. Некоторые из этих методов были разработаны в рамках искусственный интеллект. Если работа DSS основана на методах искусственного интеллекта, мы говорим об интеллектуальном DSS или IDSS» [53].

На основе первых определений DSS был определен круг решаемых с их помощью задач: неструктурированные и полуструктурированные. Классификация проблем, согласно которой неструктурированные задачи имеют только качественное описание, основанное на суждении лица, принимающего решения, а количественные отношения между основными характеристиками задачи не известны, оказала значительное влияние на эту ориентацию DSS. С помощью хорошо структурированных задач можно количественно оценить важные взаимосвязи. Промежуточное положение занимают полуструктурированные задачи, «в которых сочетаются количественные и качественные зависимости, при этом малоизвестные и неопределенные аспекты задачи имеют тенденцию преобладать» [11].

Ряд исследователей рассматривают DSS как средство «принятия решений» и определяют его как систему, «основанную на использовании моделей ряда вычислительных методов и суждений, чтобы помочь менеджеру в принятии решений».

Одним из наиболее важных факторов, определяющих качество принимаемых решений, является интеллект лица, принимающего решения (босса, менеджера, пользователя). Под интеллектом следует понимать весь интеллектуальный потенциал лица, принимающего решения, в целом: навыки творческого мышления, данные природой, знания, полученные в процессе обучения, практики, жизненного опыта и т. д.

В этом контексте необходимо рассматривать задачи повышения интеллектуального уровня лиц, принимающих решения, не за счет использования традиционных методов обучения, а за счет использования методов и средств (систем) искусственного интеллекта (ИИ) на основе аналитических данных. технологии обработки.

Это направление ориентировано на создание набора подходящих программных и аппаратных средств, которые лица, принимающие решения, могут использовать для решения задач интеллектуального характера, требующих семантической обработки больших объемов информации, хранящейся в базах данных.

Таким образом, совокупность современных информационных технологий позволяет говорить о разработке информационной системы (подсистемы) интеллектуальной поддержки принятия решений, основной целью которой является своевременное и качественное обеспечение всех информационных потребностей менеджеров при принятии решения. Производственный процесс. Это делает возможным:

- Автоматизировать процессы бизнес-администрирования за счет интеллектуализации бизнес-решений.
- Повысить эффективности производства компании.
- Открыть сферу производства, позволяющую говорить о разработке полноценных автоматизированных систем управления информацией (комплекс технического, специализированного математического программного обеспечения, а также информационное и языковое

обеспечение) на основе современных информационных технологий для обработки данных.

1.3.2 Схема процесса принятия решений

Каждый руководитель в процессе функционирования предприятия сталкивается с необходимостью принятия решений. При этом зачастую возникает проблема, как сделать оптимальный выбор, получить наибольшую пользу и уменьшить возможные негативные последствия от реализованного решения.

Б. Маклафлин считал, что «принятие решений в профессиональном отношении представляет собой особый вид человеческой деятельности, который состоит в обоснованном выборе наилучшего в некотором смысле варианта или нескольких предпочтительных вариантов из имеющихся возможных» [26].

Задачи принятия решений часто отождествляются с задачами выбора, являющимися одними из самых распространенных задач, с которыми человек сталкивается в своей деятельности. В сложных и редких, уникальных ситуациях человек более тщательно подходит к своему выбору. Прежде чем принять решение, он старается детально рассмотреть, оценить и сопоставить различные варианты, учесть разные точки зрения.

Еще более сложные задачи выбора возникают в профессиональной деятельности каждого сотрудника с компетенциями принятия решений. «Для сравнения различных вариантов действий приходится проводить всесторонний, иногда достаточно сложный анализ проблемной ситуации, разрабатывать для этого специальные модели, привлекать к выработке вариантов решения специалистов, экспертов, консультантов, аналитиков, использовать средства вычислительной техники, строить компьютерные системы поддержки принятия решений» [36].

Как отмечал Терлянский П.В, «в ситуациях принятия сложных решений всегда существует недостаток информации. Часть нужной информации

нередко отсутствует, а имеющаяся информация может быть противоречивой. Опытный руководитель или специалист покрывает неполноту информации своими знаниями, умением и интуицией. Принятие верных решений в сложных ситуациях является своего рода искусством, которым владеют немногие» [46].

Принятие решений, как уже отмечалось, «есть особый вид человеческой деятельности, направленный на нахождение наилучших из возможных вариантов. Конечный результат решения проблемы определяется многими участниками, имеющими различные функции. Главное место принадлежит человеку или группе людей, которые фактически осуществляют выбор предпочтительного решения» [48].

Формальные методы принятия решения могут оказаться полезными в следующих случаях:

- существует некоторая проблема или проблемная ситуация, требующая своего разрешения;
- имеется несколько вариантов решения проблемы, способов достижения цели, действий, объектов, среди которых производится выбор;
- присутствуют факторы, накладывающие определённые ограничения на возможные пути решения проблемы, достижения цели;
- имеется человек или группа лиц, которые заинтересованы в разрешении проблемы, имеют полномочия для выбора того или иного варианта решения и несут ответственность за выполнение принятого решения.

Приведем типовую схему процесса принятия решения, устанавливающую набор и последовательность этапов при принятии решения, и обозначим основных действующих лиц этого процесса и их роли. Жизненный цикл решения проблемы состоит из нескольких стадий (рис. 4) и представляет собой многоэтапную итеративную процедуру.

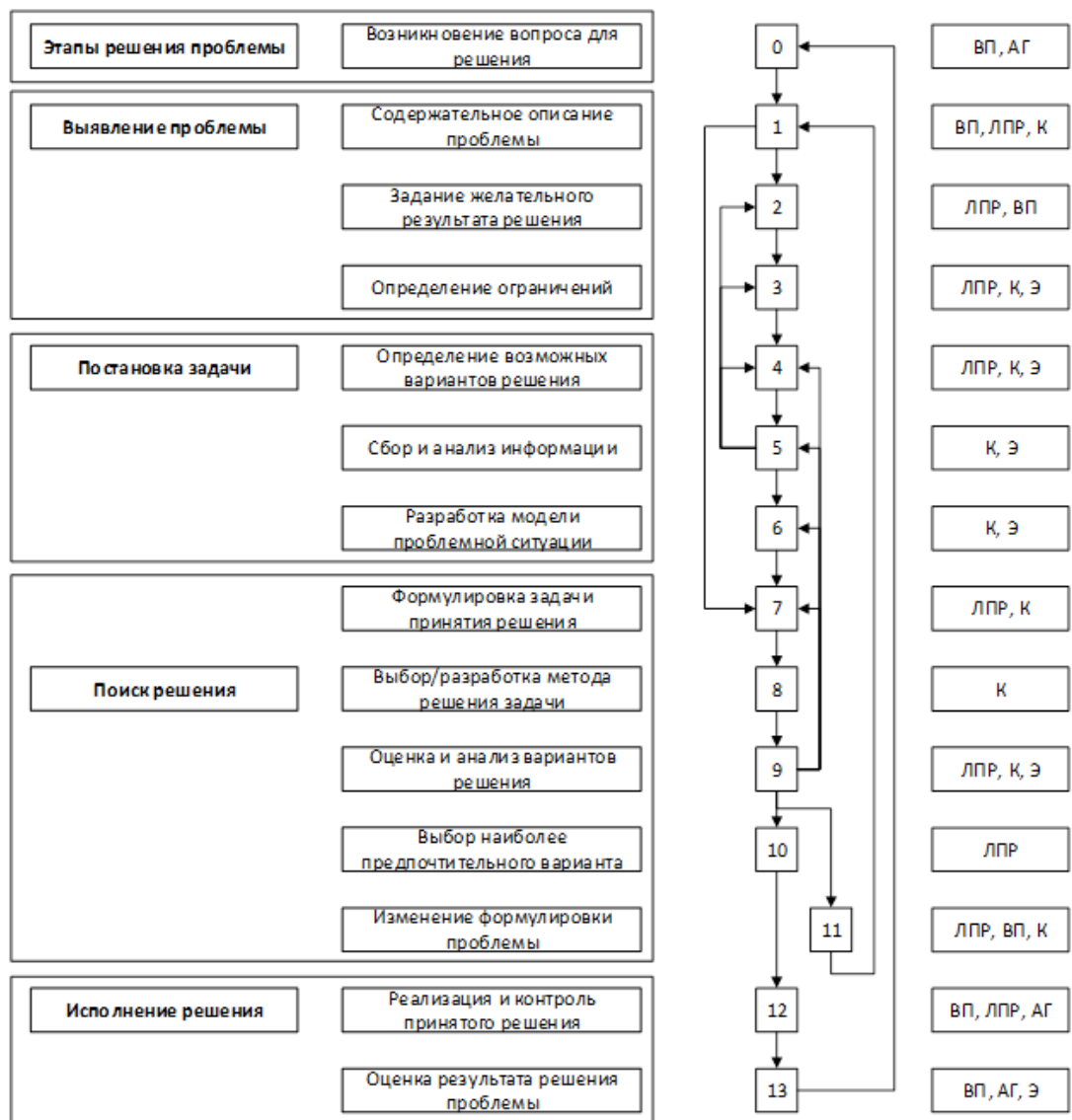


Рисунок 4 - Жизненный цикл решения проблемы

Потребность в том, что требуется принять решение, появляется в случае, когда возникает какая-либо проблема (нулевой этап). В данной ситуации выявляются проблемы (первый – третий этапы). Иначе говоря, устанавливается и охарактеризовывается суть проблемы, а также определяется какой результат ожидается после того, как она будет решена, производится оценка присутствующих ограничений.

На этапах с четвертого по седьмой устанавливаются задачи принятия решения. С этой целью необходимо установить ряд различных способов решения.

Исходя из того какая проблема, количество альтернативных решений, как правило, достигает порядка десятка, сотен и даже тысяч единиц. С точки зрения теории количество анализируемых вариантов бывает и бесконечным.

Для наиболее полного описания всех существующих способов решения, как правило, нужно производить сбор и анализ множества информации, которая имеет прямое отношение к проблеме и различным вариантам ее решения. «Отсутствие или невозможность получения нужных сведений может сделать проблему неразрешимой» [54]. В таких случаях приходится возвращаться к исходной постановке проблемы и изменять ее описание. Вторая стадия завершается формулировкой задачи принятия решения.

Так же обратим внимание на то, что наиболее подробное и полное описание решаемой проблемы, начиная с 1-ой стадии, в большинстве случаев устанавливает возможные варианты ее решения и может уже подтолкнуть к определению задачи принятия решения, обходя многие, а часто все последующие стадии.

Определив задачу принятия решений, далее идет этап, который заключается в поиске решения (этапы 8 — 10). Сюда входят следующие шаги:

- подборка распространенного способа решения задачи либо формирование нового;
- непосредственно этап процесса решения, который включает проведение оценки и анализа разных способов решения и выбор среди них более эффективного.

Среди общего числа задач достижение окончательного результата не является сложным процессом. Тем не менее, зачастую принятие решений являются непростыми процессами, которые требуют привлечения знаний и опыта большинства людей и возможностей современной вычислительной техники [10].

Также отметим, что возможно ситуация, когда нет наиболее подходящего решения, оно отсутствует. В этом случае допускается изменение формулировки исходной проблемы (одиннадцатая стадия), или возврат на

прошлые этапы и сбор соответствующей информации, внесение изменений в формальную постановку задачи или модель проблемы. Так или иначе проведенный способ поиска наилучшего варианта решения, даже если он не привел к положительному результату, не будет бесполезным.

Он может натолкнуть на новое понимание рассматриваемой проблемы, обратить внимание на какие-то новые аспекты, которые необходимо учесть, указать на иные пути решения задачи. Если приемлемый вариант найден, наступает стадия исполнения решения (этапы 12, 13), на которой происходит реализация принятого решения, осуществляется контроль над процессом реализации и оценивается результат разрешения проблемной ситуации.

Этап реализации решения не имеет прямого отношения к процедуре принятия решения, однако, интеграция ее в общую схему играет важную роль с практической точки зрения, поскольку данный этап завершает жизненный цикл процесса появления, решения и устранения проблемы.

На сегодняшний день процедура принятия решения получает поддержку автоматизированными программными системами. Понятие «системы поддержки принятия решений» (далее СППР) по своей сути компьютерные системы, которые за счет собирания и анализа общего количества сведений зачастую влияют на процесс принятия решений организационного плана в предпринимательской. Благодаря интерактивным системам руководству извлечь наиболее полезную информацию из первоисточников, провести анализ имеющихся бизнес-модели для решения тех или иных задач. Посредством СППР возможно отслеживание доступных информационных активов, выявить сравнительные значения по определенным параметрам, рассмотреть возможные альтернативные решения.

СППР заключается в решении 2-х ключевых задач [53]. Первая – выбрать наиболее эффективное решение из ряда всех возможных (оптимизация). Вторая – упорядочить различные способы решений в соответствии с их ранжированием.

Каждая из этих задач опирается на выбор совокупности критериев, согласно которым далее будет проводиться оценка и сравнение возможных способов решения. Благодаря СППР пользователь уже может выбрать окончательный вариант решения.

Сегодня составляющие СППР включают в себя: пользовательский интерфейс, который основан на знаниях подсистемы, модуль управления данными, модуль управления моделями.

Пользовательский интерфейс представляет собой компонент, обеспечивающий взаимодействие пользователя с СППР. Метод управления данными служит в качестве подсистемы компьютерной СППР и включает в себя следующие собственные подкомпоненты:

- БД интегрированной СППР, состоящей из данных, которые были получены из внутренних и внешних источников;
- система управления БД (БД может быть реляционной или многомерной);
- словарь данных, который включает в себя каталог, где содержатся все определения данных БД;
- инструменты запросов, которые предполагают присутствие языков для запросов к БД.

Модуль управления моделью включает в себя такие компоненты, как:

- модельная база, которая включает количественные модели, которые позволяющие системе проводить анализ и выявлять способы решения проблем;
- модуль управления модельной базой, предназначенная для создания новых моделей с использованием языков программирования;
- словарь моделей, содержащий определение моделей и другую информацию, связанную с ними;
- модуль создания, исполнения и интеграции моделей, который будет интерпретировать инструкции пользователя в соответствии моделями и перенесет их в систему управления этими моделями.

СППР бывает следующих типов [10]:

- Системы запроса статуса.
- Система анализа данных.
- Информационно-аналитическая система.
- Система бухгалтерского учета.
- Модель на основе системы.

Согласно исследованию О.И. Ларичева, «процесс принятия решений с помощью, компьютеризированных системам, и включает в себя следующие шаги:

Определение проблемы. Это важный этап. Он предоставляет лицам, принимающим решения, базу, на которой они могут строить предположения, собирать и анализировать данные и оценивать альтернативы. Определение проблемы начинается с признания того, что проблема существует. Проблема существует, когда:

- есть разница между ожидаемым и доставленным;
- существует отклонение от обычного;
- предпринятые действия не оправданы.

СППР определяет проблему и сложности, связанные с сопоставлением результатов [52].

Определение лица, принимающего решения. В зависимости от характера проблемы, ее отправляют нужному человеку. Плохо структурированная проблема перейдет к высшему руководству, сложная проблема — менеджерам, а повторяющиеся будут отправлены работнику на более низком иерархическом уровне.

Сбор информации. Как только проблема отправлена нужному человеку, заинтересованный человек может начать сбор данных и выявление факторов, влияющих на ситуацию. Без СППР сбор и анализ данных займет слишком много времени. СППР может обработать большое количество данных за короткое время.

Оценка альтернатив и решение. Эта стадия включает в себя анализ всех возможных направлений действий и определение наиболее подходящего из них путем оценки плюсов и минусов каждой альтернативы. DSS помогает в обосновании конкретного выбора.

Внедрение и контроль. Как только решение принято, необходимо идти дальше. Реализация требует планирования. Мониторинг также важен для определения полезности конкретного решения для достижения целей. Это может потребовать некоторых корректировок или привести к новой проблеме. В последнем случае, возможно, придется повторить весь процесс» [34].

1.4 Моделирование процесса принятия решений

Более подробно деятельность предприятия можно проанализировать посредством методологии IDEF0. Как считает М.И. Башмаков, «методология IDEF0 является эффективным средством анализа, проектирования, а также представления деловых процессов. Структурной единицей модели IDEF0 есть диаграмма, которая представляет собой графическое описание модели предметной области, а также и ее части. Основным компонентом диаграммы IDEF0 является блок. Блоки диаграммы отображают работы, процессы, функции и задачи, выполняемые в определенное время.

Модели в нотации IDEF0 необходимы с целью высокоуровневого описания бизнеса фирмы в функциональном плане. Нотация DFD дает возможность отражения последовательности работ, которые выполняются по ходу процесса, и потоков сведений, циркулирующих между данными работами. Цель методологии IDEF0 – построить функциональную схему исследуемой системы, которая описывает все процессы с точностью, которой достаточно в целях однозначного моделирования деятельности системы. В IDEF0 моделируемую систему представляют как совокупность связанных друг с другом работ. Основой методологии являются 4 ключевых понятия: функциональный блок, декомпозиция, дуга (стрелка), глоссарий» [9].

Методология IDEF0 использует метод функционального моделирования сложных систем, являющийся частью методологии структурного анализа и проектирования SADT (Structured Analysis and Design Technique). IDEF0 позволяет строить функциональные модели, которые описывают бизнес-процессы в виде иерархической системы взаимосвязанных функций.

Интересное исследование было проведено С.К. Беловой, которая отмечает, что «моделирование бизнес-процессов затрагивает многие аспекты деятельности объекта исследования:

- изменение внутренних нормативных документов и технологии проведения операций;
- изменение организационной структуры;
- новые требования к автоматизации выполняемых процессов;
- оптимизацию функций подразделений и сотрудников;
- перераспределение прав и обязанностей руководителей.

Моделирование бизнес-процессов позволяет не только определить, как система работает в целом, как взаимодействует с внешними факторами, но и как организована деятельность на каждом рабочем месте. Моделирование бизнес-процессов — это эффективное средство поиска пути оптимизации системы, средство прогнозирования и минимизации рисков» [10].

Нотация IDEF0 используется для создания верхнего уровня модели бизнес-процессов. Построение IDEF0-диаграммы верхнего уровня обеспечивает наиболее общее или абстрактное описание объекта моделирования. Подготовленную модель необходимо согласовывать с архитекторами и программистами, для подтверждения того, что структура бизнес-процессов наглядна, понятна и доступна. Модели должны быть согласованы с ответственными специалистами организации, обладающими полной информацией по конкретному бизнес-процессу.

Функциональность системы поддержки принятия решений описана с помощью методологии IDEF0, которая показана на рис. 5–6.

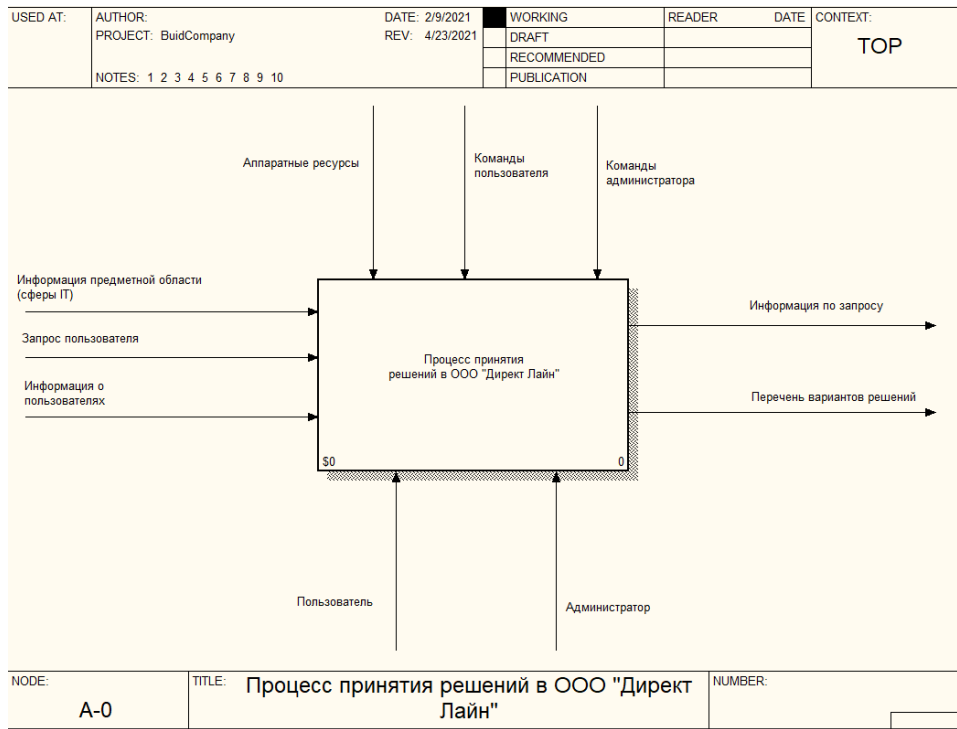


Рисунок 5 – Процесс поддержки принятия решений

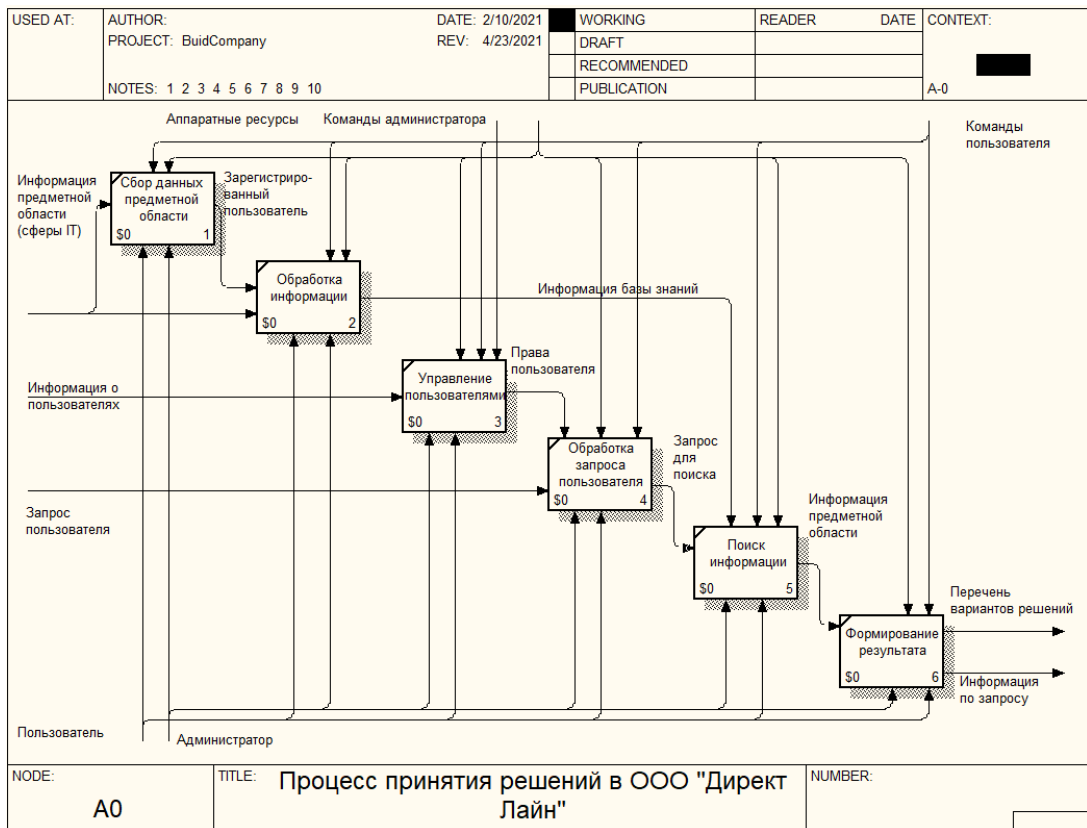


Рисунок 6 – Декомпозиция процесса поддержки принятия решений

Входящими потоками системы поддержки принятия решений являются информация предметной области, запрос пользователя, информация о пользователях. Механизмы включают пользователей и администраторов системы [53]. Управляющими потоками являются аппаратные ресурсы, команды пользователя, команды администратора. На выходе находится информация по запросу, перечень вариантов решений.

Развернутый процесс включает блоки: регистрация в системе, авторизация в системе, управление пользователями, управление проектами, сбор данных предметной области, формирование отчетов.

1.5 Обзор систем поддержки принятия решений

Существует множество программ, которые применяются в качестве систем поддержки принятия решений. Одной из таких систем является NERPA. В качестве информационной системы поддержки принятия решений в структуре программы NERPA ERP используется модуль аналитики BI. С помощью этого модуля процесс поддержка управления процессами в различных отраслях реализуется в виде анализа и разнообразных бизнес-отчётов, помогающих быстрой и результативной выработке оптимальных управленческих решений.

BI система (бизнес-аналитика, Business Intelligence) — «это система для построения отчётов различных отделов предприятия. Система отчётности входит в пакет ERP решения NERPA и используется всеми модулями программы для целей бизнес-аналитики».

Управление отчётностью необходимо практически всем сотрудникам предприятия, использующим модули NERPA для планирования различных ресурсов. BI систему используют:

При помощи автоматизированной системы отчётности, предоставляемой BI модулем, выполняются и автоматизируются следующие операции на предприятии:

– Формирование текущих и заключительных отчётов в соответствии со стандартами предприятия, нормативными актами или потребностями управляющего персонала.

– Систематизация данных ВІ отчётности для задач бизнес-аналитики, например, для поиска слабых мест в организации работ, планирования расходов и т. д.

– Групповая консолидация, бюджетирование, учёт текущих и необходимых ресурсов, создание скользящих прогнозов, упрощение принятия решений.

– Мониторинг ключевых показателей эффективности и производительности, соответствия стандартам и регламентам.

– Информирование руководящих специалистов о ходе выполнения задач, занятости и ответственности сотрудников, предоставление других важных данных для организации работы предприятия [13].

Система поддержки принятия решений (СППР) "Выбор" – «аналитическая система, основанная на методе анализа иерархий (МАИ), является простым и удобным средством, которое поможет:

- структурировать проблему;
- построить набор альтернатив;
- выделить характеризующие их факторы;
- задать значимость этих факторов и оценить альтернативы по каждому из факторов;
- найти неточности и противоречия в суждениях лица, принимающего решение;
- провести анализ решения и обосновать полученные результаты».

Система опирается на математически обоснованный метод анализа иерархий Томаса Саати. СППР «Выбор» на основе МАИ может использоваться при решении следующих типовых задач:

- оценка качества организационных, проектных и конструкторских решений;

- определение политики инвестиций в различных областях; задачи размещения (выбор места расположения вредных и опасных производств, пунктов обслуживания);
- распределение ресурсов;
- проведение анализа проблемы по методу "стоимость-эффективность";
- стратегическое планирование;
- проектирование и выбор оборудования, товаров;
- выбор профессии, места работы, подбор кадров [7].

Также поддержку принятия решений можно осуществлять при помощи программ имитационного моделирования: RAO-studio, RepastPy, Simio, BPsim.MAS.

Аксенов в своей работе обращает внимание на Программный комплекс RAO-studio, который «является основной частью интеллектуальной интегрированной среды имитационного моделирования и предназначен для разработки и отладки имитационных моделей на языке РДО. Основные цели данного комплекса - обеспечение пользователя легким в обращении, но достаточно мощным средством разработки текстов моделей на языке РДО, которое обладало бы большинством функций по работе с текстами программ, характерных для сред программирования, а также средствами проведения и обработки результатов имитационных экспериментов» [2].

Принцип функционирования языка РДО относится к поддержке событийного подхода и подхода сканирования активностей с возможностью применения продукционных правил, что дает гибкость в написании моделей для разных сфер человеческой деятельности. Интерфейс среды прост в понимании и в изучении, обладает доступным и понятным набором правил для пользователя, но не имеет графической части. И так же отсутствие работы с базой данных и со структурированным выводом результатов, особенно в виде таблицы усложняет работу в среде.

Технология Simio — это коммерческое программное обеспечение, предназначенное для моделирования систем при условиях неопределенности ситуации. Это система моделирования, построенная на интеллектуальных объектах и относится к поддержке процессно-ориентированного подхода.

Система подходит для реализации задач совершенствования производства, интуитивно понятна, само построение моделей осуществляется в большей части графически с помощью блоков, ссылок и ресурсов с созданием в теле самих элементов продукционных правил, что говорит о быстроте, удобстве построения и наглядности.

Repast – «библиотека написана на нескольких языках программирования: RepastJ и RepastS (Java), RepastPy (Python), Repast.Net (C# и .Net), и предоставляющая средства для агентно-ориентированного моделирования систем» [4]. BPsim.MAS — «система динамического моделирования ситуаций. BPsim.MAS поддерживает описание модели мультиагентных процессов преобразования ресурсов с помощью графической нотации и продукционных правил «Если ... То» [12, 14].

В данной системе доступно несколько способов создания моделей: на ресурсах – данный способ реализации применяется в случае простых моделей процессов преобразования ресурсов и является наиболее доступным для неподготовленных пользователей; на заявках (транзактах) – применяется в том случае, когда необходимо различать ресурсы (объекты) и их свойства (атрибуты); на интеллектуальных агентах – используется при необходимости описания процессов управления.

1.6 Требования к информационной системе

База знаний для ООО «Директ Лайн» будет разработана собственными силами. Преимуществами этого подхода при приобретении информационной системы состоит в полном контроле процессов проектирования и разработки, свободе выбора программного обеспечения для разработки, выбор более

актуальных технологий для разработки информационной системы, понимание процессов разработки и функционирования системы, обеспечение возможности последующего обновления информационной системы за счет разработки и подключения дополнительных функциональных модулей.

По результатам изучения предметной области определены следующие функции программного средства – базы знаний системы поддержки принятия решений:

Функции администратора системы:

- авторизация в системе;
- управление учетными записями пользователей;
- управление ролями и доступом пользователей к системе;
- управление информацией в базе знаний;
- управление разделами информации в базе знаний;
- управление списком экспертов.

Функции пользователя:

- регистрация и авторизация в системе;
- поиск информации в базе знаний;
- сохранение информации по определенному вопросу в личный кабинет;
- просмотр истории поиска;
- отправка вопросов другим пользователям (экспертам).

Системные требования для рабочего места пользователя системой: гарантированное подключение к сети Интернет от 2МБит/с, браузер – GoogleChrome.

Системные требования для рабочего места учителя: минимальные требования - ОС - Windows XP/7/8, процессор - Intel i3 4xxx, ОЗУ - 4Гб. Рекомендуемые технические параметры: ОС - Windows 7/8/10, процессор - Intel i5 4xxx, ОЗУ - 6Гб.

Системные требования для рабочего места ученика: минимальные - ОС Windows XP/7/8, процессор - 2-ядерный с частотой ядра не менее 2 ГГц, ОЗУ - 2Гб. Рекомендуемые: ОС - Windows 7/8/10, процессор - Intel i3 / i5, ОЗУ - 4Гб.

В состав СППР входят три главных компонента: информационное ядро, которое, в свою очередь, состоит из базы данных и базы знаний, далее следует база моделей, а также программная управляющая среда в виде систем управления базой данных, базой моделей и пользовательским интерфейсом [51].

В представленной СППР при запуске основного программного компонента выбираются критерии для анализа. Далее выбираются альтернативы. После этого нужно будет попарно сравнить выбранные критерии, а затем попарно сравнить альтернативы для выбранных критериев, что выполняется в соответствии со значимостью этих критериев в соответствии с оценками эксперта.

В итоге система генерирует отчет с результатом анализа, в котором представлена важность каждого критерия в процентном соотношении для выбранных критериев, общие оценки с выделенной оценкой лучшей альтернативы по итогам сравнения, относительные индексы целостности каждого сравнения критериев и информацию по сравнению альтернатив для критериев [2]. Таким образом, производится обоснованный анализ выбранных альтернатив по необходимым критериям, а также формируется общая оценка приоритетности альтернатив при решении поставленных задач.

Выводы по разделу. В 1 разделе рассмотрены теоретические вопросы: базы знаний в экспертных системах; характеристика компании ООО «Директ Лайн»; описание процесса принятия решений; моделирование процесса принятий решений; обзор систем поддержки принятия решений; требования к информационной системе. Также обоснована структура системы поддержки принятия решения в соответствии с основными требованиями к ее описанию, и определена роль базы знаний в осуществлении деятельности системы.

2. Проектирование базы знаний информационной системы поддержки принятия решений

Жизненный цикл проектируемого программного продукта включает период от появления потребности в данной информационной системе и до прекращения ее применения.

Предпроектное обследование в себя включает:

- Подготовка сведений для проектирования, с выделением формулированных требований, с исследования объекта автоматизации, предоставляются первичные выводы пред проектным вариантом информационной системы;

- исследование материалов и создание документации, предоставление технико-экономического обоснования с техническим заданием на проектирование информационной системы.

Проектирование:

- предварительное проектирование: по аспектам разработки ИС подбираются проектные решения; описания реальных компонент ИС; технический проект оформляется и утверждается;

- детальное проектирование: подбор или создание необходимых алгоритмов программ или математических методов; регулирование структур БД; разработка документации на установку и доставку программных продуктов; подбор комплекса технических средств с документацией на ее установку;

- техно-рабочий проект ИС должен быть разработан;

- для реализации функций управления с помощью ИС и описания регламента действий аппарата управления разрабатывается методология.

Разработка ИС [3]:

- программные и технические средства должны быть получены и установлены;

- программный комплекс должен быть протестирован и налажен;

- инструкций по эксплуатации программно-технических средств должны быть разработаны.

Внедрение информационной системы в работу:

- Технические средства должны быть внедрены;
- Программные средства должны быть внедрены;
- Персонал должен быть обучен и сертифицирован;
- Должен быть произведен пробный запуск;
- Система должна быть сдана и подписаны все акты приемки-сдачи работ.

Эксплуатация ИС:

- Ежедневная эксплуатация;
- Полное сопровождение проекта.

Для разработки базы знаний системы поддержки принятия решений выбран модифицированный каскадный способ. Разделение всей разработки на этапы - главная характеристика классического каскадного способа, только после завершения работы на текущем этапе происходит переход к следующему. Выпуск полного комплекта документации, по которой возможно продолжение разработки другой командой специалистов, предшествует концу каждого этапа.

Применение каскадного этапа имеет следующие положительные стороны [51]:

- Соответствующий параметрам полноты и согласованности конечный комплект проектной документации создается на каждом этапе;
- Возможность спланировать сроки окончания работ и затраты на них, дают производимые в логичном порядке этапы работ.

На практике реальный процесс разработки не может уложиться в такую жесткую схему, что является одним из недостатков классического каскадного подхода. Иногда появляется необходимость вернуться к предыдущим этапам и пересмотреть или уточнить уже принятые, при создании информационной системы, решения.

Реальный процесс разработки базы знаний системы поддержки принятия решений ИТ-компании ООО «Директ Лайн» в итоге принимает следующий вид (рис. 7).

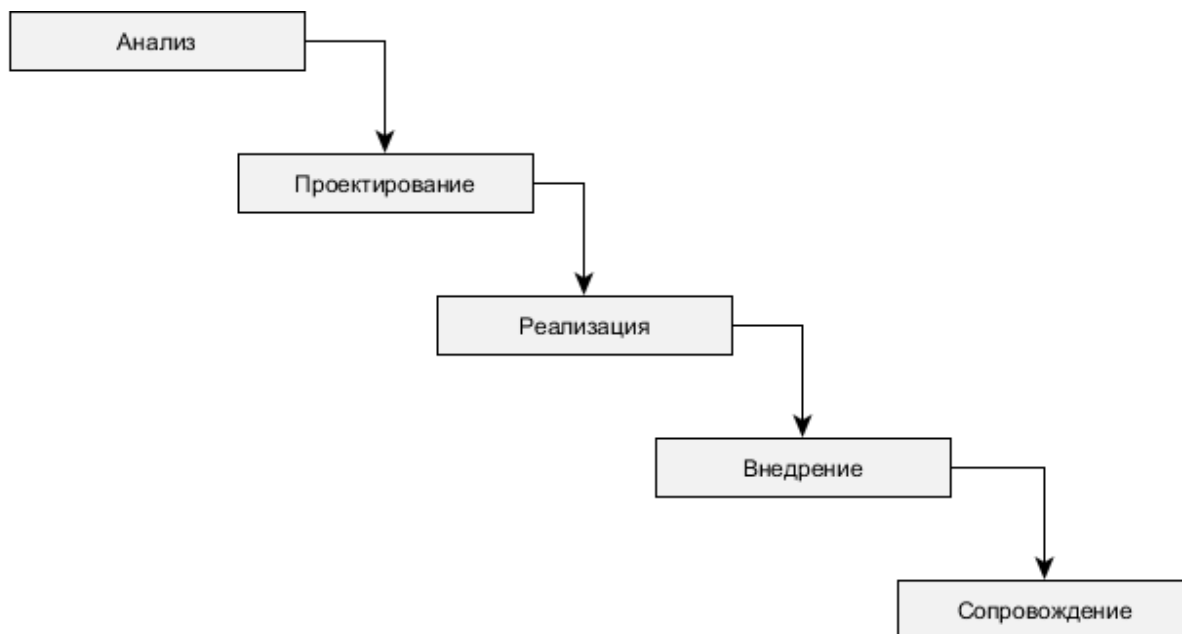


Рисунок 7 – Каскадная схема разработки АИС

В точках, намеченных по окончании каждого этапа работ, проходит согласование результатов с пользователями, на весь период создания ИС, требования в виде технического задания остаются "заморожены". Получается, что внесение поправок пользователями становится возможно только после завершения работы над системой.

Пользователи получают не соответствующую их потребностям систему, если требования будут неточно сформулированы или изменятся за продолжительное время разработки программы.

Модели автоматизируемого объекта (как информационные, так и функциональные) имеют шанс устареть синхронно с их утверждением.

В реальных условиях программный продукт – база знаний системы поддержки принятия решений развивается по каскадной модели, с контролем после каждого этапа. Первым жизненным этапом является анализ предметной

области, на этом этапе проводится исследование предметной области, далее осуществляется проектирование программного продукта и его базы данных. После этого идет этап реализации в процессе которого осуществляется разработка модулей информационной системы (рис. 8).

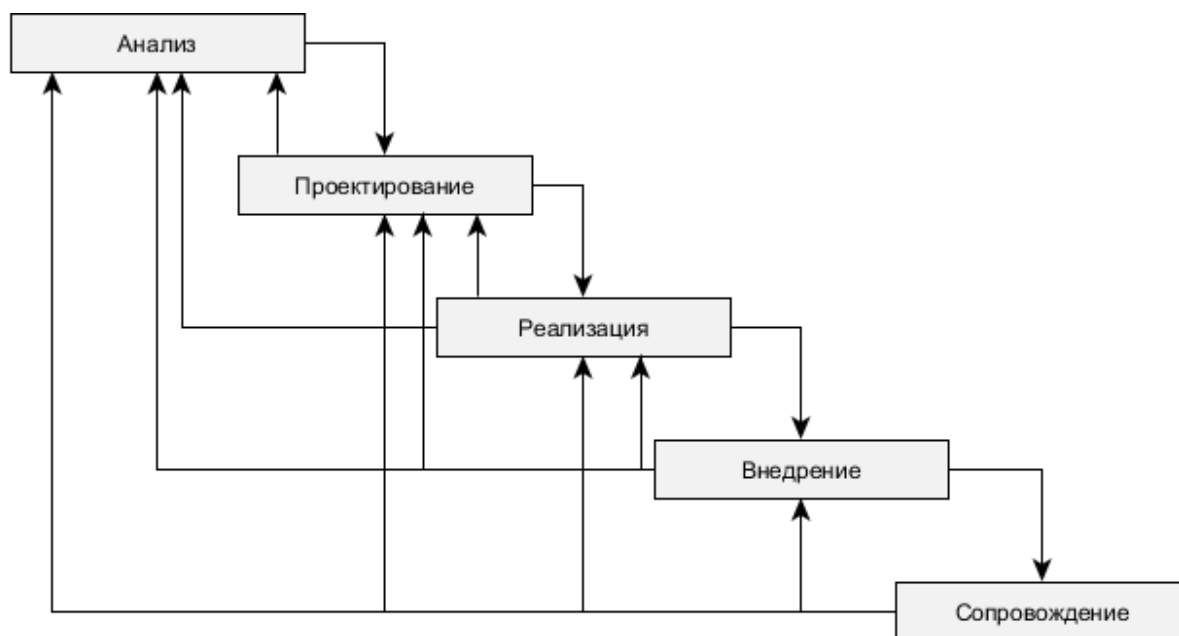


Рисунок 8 – Реальный процесс разработки программы по каскадной схеме

После создания системы следуют этапы внедрения и сопровождения. В представленной схеме жизненного цикла информационной системы прослеживается тесная взаимосвязь и взаимовлияние между этапами, поскольку в случае возникновения потребности при завершении какого-либо этапа, и выявлении в нем необходимости изменений осуществляется возврат на предыдущие этапы для внесения нужных корректировок. Таким образом, для реализации программного продукта – базы знаний системы поддержки принятия решений определена каскадная модель разработки с поэтапным контролем.

2.2 Архитектура и структура базы знаний

Структура отдельного программного приложения, заключающая в себе внешние свойства и взаимосвязи программных элементов, является архитектурой программы. Определение организационной структуры планируемой к разработке системы – главная задача, исполняемая программной архитектурой [20]. Важность программной архитектуры заключается в возможности планирования и распределения ресурсов, необходимых для конкретного проекта, с большей точностью, кроме того, программная архитектура устанавливает ограничения исполнения каждого проекта.

Проектируемая система основывается на архитектуре клиент-сервера. «Режимом работы запрос-ответ» именуют связи клиента с сервером. Он показан на рисунке 9.

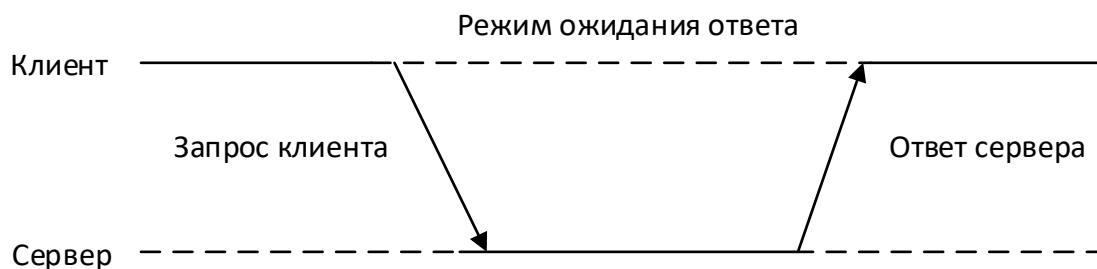


Рисунок 9 - Обобщенное взаимодействие клиента и сервера

При надежности базовой сети аналогичной локальной, взаимодействие между сервером и клиентом реализуется с помощью простого протокола, без установления соединения.

Надежный протокол (TCP/IP) с установкой соединения используется во многих клиент-сервер системах [44]. Ответное сообщение отправляется сервером с использованием того же самого соединения, после чего оно будет разорвано.

Три уровня включает в себя любое клиент-серверное приложение: хранение данных, обработка информации, пользовательский интерфейс.

Реализация уровня пользовательского интерфейса, обычно производится на клиентах. Пользователь работает с веб-приложением с помощью программ, охватываемых этим уровнем.

Основная функциональность приложения, в особенности функции передачи данных, их обработки и т. п. обеспечивается уровнем обработки.

Программные средства, обеспечивающие данные для обработки в приложении, содержит уровень данных в модели клиент-сервер. Сохранность данных – главная характеристика и назначение уровня данных. В расчете на дальнейшее использование данные обязаны сохраняться в определенном месте, пока программа не функционирует.

На стороне сервера традиционно находится уровень данных у модели клиент-сервер. Метаданные сохраняются на этом уровне, что значит поддержание целостности для баз данных. В итоге самостоятельность данных представляется главным фактором. Независимость от веб-приложения данных, сохраненных в хранилище (базе данных), обеспечивает неизменность веб-приложения при изменении организации данных и наоборот.

Трехуровневая архитектура (рисунок 10) является основой структуры разрабатываемой системы

Трехуровневая архитектура (рисунок 10) является основой структуры разрабатываемой системы. Для клиент-серверного приложения как стандартная модель используется трехуровневая архитектура.



Рисунок 10 - Трехуровневая архитектура системы

Управление прикладным выполнением и базой данных является целью прибавочного («среднего») уровня. Размещение уровней на разных компьютерах или на одном в тестовом режиме схоже с двухуровневой моделью [19]. Программные продукты, взаимодействующие через стандартный Интернет-браузер в архитектуре «клиент-сервер» строятся с помощью web-приложения. Пользователь взаимодействует с приложением через формы и страницы, из которых он получает необходимую ему информацию и отправляет какую-то информацию назад. Благодаря применению протокола НТТР (Hyper Text Transfer Protocol) осуществляется связь частей подобранной трехуровневой архитектуры. Он является протоколом прикладного уровня, используемым для передачи гипертекста. Для извлечения файлов с соответствующих ресурсов используется в сетях ТСР/ІР как протокол высокого уровня. Все транзакции рассматриваются независимо от остальных, так как НТТР не сохраняет данные о своем состоянии.

Иерархическая структура будет использована в разрабатываемой системе (рисунок 11). Для систематизированной, проработанной и организованной информации отлично применяется это представление. В направлении от общего к частному происходит перемещение по уровням такой структуры.

На рис. 11 представлена структура разрабатываемой базы знаний информационной системы поддержки принятия решений.

Структура приложения базы знаний представлена главной страницей, с которой осуществляется переход на следующие разделы:

- поиск;
- справочник;
- определения;
- тематика определений;
- эксперты;
- экспертные мнения;

- помощь;
- статьи;
- категории статей;
- история поиска;
- личный кабинет;
- панель администратора.

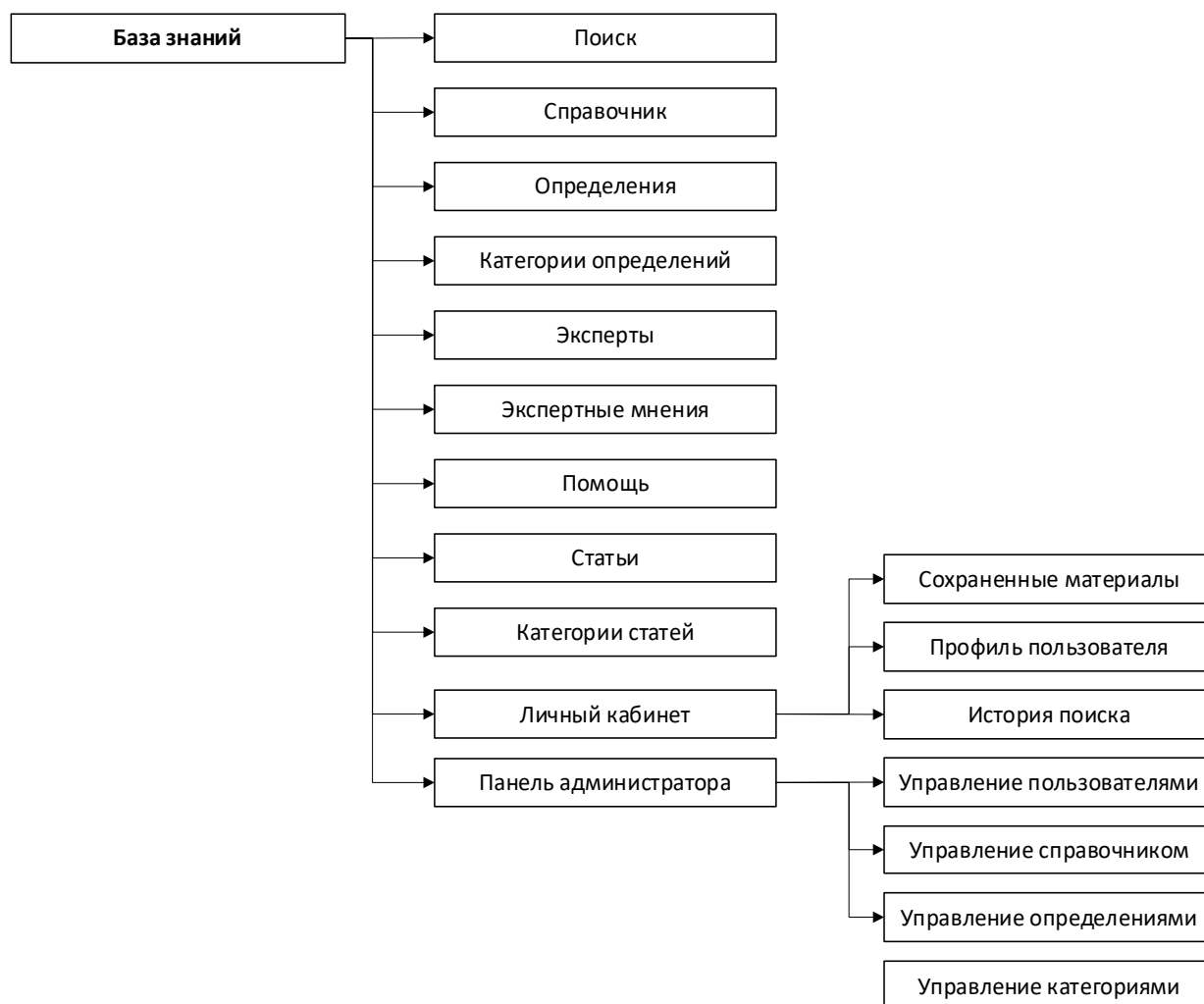


Рисунок 11 – Структура разрабатываемой базы знаний

Для разрабатываемой базы знаний подходит данная информационная архитектура, она легко позволит воспринимать посетителям приложения, информацию, которая указана на нем и включает справочники, определения, статьи по разным категориям, экспертные мнения.

2.3 СУБД для реализации проекта

MySQL выбрана для проекта также как и СУБД, что обоснованно путем анализа различных СУБД, представленных в таблице 1.

Взглянув на таблицу, можно заметить, что СУБД MySQL выделяется среди конкурентных СУБД, у нее есть огромное преимущество - поддержание языком программирования PHP любой библиотек. Отличительные качества данной программы в том, что это реляционная база данных, которая обеспечивает хранение информации с высокой степенью удобства и надежности. Само слово «relational» происходит от английского слова «Relationship». Реляционная СУБД [30, 47]. используется для работы с реляционными базами данных.

Таблица 1 – Сравнение СУБД

	Oracle	MSSQL	MySQL	FireBird
Бесплатность	нет	нет	да	Да
Unix-совместимость	да	нет	да	Да
Максимальный размер базы	неогр.	16ТБ	неогр.	131 ТБ
Максимальный размер таблицы	неогр.	532 ГБ	256 ТБ	2,5 ТБ
Защита от подбора пароля	да	нет	нет	да
Максимальное число пользователей	не ограничено	не ограничено	не ограничено	1000
Наличие встраиваемых версий СУБД	нет	нет	да	да

В этой модели данные выглядят как двумерные таблицы, каждая из них обладать следующими свойствами:

- каждый элемент таблицы — это только один элемент данных;
- каждая ячейка в столбце таблицы однородна - все ее элементы должны быть одного типа (INT, FLOAT, VARCHAR и т. Д.);
- каждый столбец имеет уникальное имя, которое идентифицирует этот столбец;
- таблица не должна содержать одинаковых строк;
- Порядок столбцов и строк произвольный.

Практически любая система управления реляционными БД дает возможность работать с языком SQL, с которыми появляется возможность определить и пройти модификацию и др. изменения структуры данных [43].

Наиболее популярны серверы реляционных БД, которые берут свои корни из клиент-серверной архитектуры [49]. Благодаря этим серверам организуется постоянная работа с базами данных большого количества клиентов.

Теперь обратим внимание на СУБД MySQL. Широко распространённое использование MySQL обуславливает то какое-либо объединение уникальных особенностей. Данные особенности представлены, надежность, скорость, расширяемость и открытость исходного кода [18].

Отметим, что существенно значимым показателем для системы управления реляционными базами данных выступает скорость, которая представляется как раз-таки временем, что затрачивается на осуществление запроса и ответ на запрос тому, кто его создал. Так или иначе MySQL является производительной системой, которая, в большинстве случаев, оказывается в разы быстрее, нежели решения-конкуренты [45]. Тесты производительности, доступные на сайте MySQL, отражают сведения о том, что MySQL идет на опережение почти каждой существующей сегодня в мире системы управления базами данных [30].

MySQL способна обрабатывать очень больших и не самых простых БД с сохранением параллельно с этим высокого уровня производительности. Таблицы, которые обладают большим объемом (в Гб), и состоят из тысячи

записей, являются широко известным явлением в ИС, которые в качестве БД используют MySQL.

Также отметим, что система MySQL и легко настраивается, и может быть оптимизирована при самых требовательных приложениях [37, 40]. Говоря о коммерческом окружении, отметим, что она полностью реализует профессионального обучения, проведение консультаций и выполняет техническое сопровождение MySQL.

MySQL также поддерживает немалое число значимых требований стандарта ANSI SQL и зачастую позволяет дополнить его различными функциями для пользователя, а также расширениями и видами данных, которые призваны совершенствовать переносимость и обеспечить для потребителей более широкий ряд функциональных возможностей. Кроме того, MySQL реализует поддержку ОС UNIX, и другие отличающиеся от нее ОС, в частности, к ним относятся Linux, FreeBSD, Solaris, OS/2, Windows 95-Vista, MacOS. У MySQL есть возможность работать на разных архитектурах, например таких, как: Intel x86, Alpha, SPARC, PowerPC и IA64 [38].

Поскольку MySQL является цельной и многофункциональной многопользовательской системой, то большинство пользователей имеют возможность получить доступ и обращаться к одной и более БД MySQL. Данный факт носит крайне важный характер, в рамках работы над созданием web-приложений, которым требуется поддержка подключения в одно и то же время множества удаленных клиентов [18].

В связи с тем, что MySQL является программным решением, которое используют на практике огромное число людей практически в каждой точке мира, она также осуществляет поддержку кодировки Юникод, включая самые значимые наборы символов (включая китайские и латинские). Наборы символов начинают принимать ко вниманию при сортировке, сравнении и сохранении данных.

Программное решение MySQL предусматривает присутствие интерфейсов к программированию приложений API на различных языках, что

позволит разрабатывать управляемых БД web-приложения на разных языках программирования. Сегодня MySQL позволяет работать с такими языками, как C, C++, Java, Eiffel, Perl, Python, PHP, Ruby и Tcl, при этом существуют коннекторы для ODBC, JDBC– и .NET–приложений.

2.4 Язык программирования для реализации проекта базы знаний

В качестве языка программирования базы знаний информационной системы поддержки принятия решений будет использован PHP. Средой разработки приложения выбрана программа PHPStorm [17]. Функции веб-сервера будут обеспечены за счет программы Open Server. Более подробно рассмотрим представленные элементы программного обеспечения [18].

В качестве языка программирования выбран PHP. Сравнительная характеристика PHP с другими языками программирования показана в таблице 2 [26].

Из таблицы можно отметить, что рассмотренные языки программирования идентичны по ряду показателей, однако PHP является преимущественным в разделе доступности средств проектирования и доступности в системе веб-сервисов. Из-за этого качество языка программирования для разработки информационной системы используется PHP.

В мире на данный момент PHP – это один из самых распространенных языков веб-программирования [1]. Существенное количество веб-сайтов и сервисов на просторах интернета пишется при помощи PHP. По статистике данный язык программирования применяем более, чем для 80 % сайтов. Это такие сервисы, как facebook.com, vk.com, baidu.com и др. Популярность PHP неувидительна, ведь доступный язык дает возможность быстро и просто разрабатывать сайты и порталы абсолютно любой сложности [30].

Таблица 2 - Сравнение языков программирования

	PHP	Ruby	Python
Предназначение	Создание динамических веб-приложений	Создание веб-приложений и компьютерных программ	Создание динамических веб-приложений и компьютерных программ
Год создания	1995	1995	1991
Сложность освоения	6	4	5
ООП	Да	Да	Да
Инструкция break	Да	Да	Да
Многомерные массивы	Да	Да	Да
Цикл foreach	Да	Да	Да
Множественное наследование	Нет	Нет	Да
Макросы	Нет	+/-	Нет
Именованные параметры	Нет	Да	Да
Наличие библиотек для работы с графикой и мультимедиа	Да	Да	Да
Работа с MySQL	Да	Да	Да
Наличие удобных и доступных средств разработки	Да (NetBeans)	Нет	Да (Visual Studio)
Поддержка популярными веб-серверами	Да (по умолчанию)	Нет (дополнительные службы)	Нет (дополнительные службы)

PHP разработан в 1994. Его создателем стал программист из Дании – Рasmus Лерддорф. Изначально PHP был простым набором скриптов на языке Perl, а позже набор скриптов переписывается в интерпретатор на языке Си. С начала существования PHP – это сокращение от PHP: Hypertext Preprocessor -

PHP: Препроцессор гипертекста) выглядел в качестве несложного набора инструментов для простого проектирования веб-сайтов и приложений [18]

К операционным системам, которые распространены практически повсеместно (Windows, MacOS, Linux) существуют собственные версии пакетов разработки на PHP. Это даёт возможность разработки и создания веб-сайтов на любой операционной системе.

PHP – это уникальный язык, который способен к работе во взаимосвязи с разнообразными веб – серверами, такими как Apache, Nginx, IIS.

Стоит отметить простоту и лёгкость освоения PHP. Если иметь базовый опыт программирования на PHP, то есть возможность создания лёгких веб – сайтов. Язык PHP схож с Си, так что, если знать Си или любой другой язык, подобный ему, то овладеть языком PHP станет просто. Язык PHP находится во взаимосвязи с большим количеством платформ баз данных (MySQL, MSSQL, Oracle, Postgre, MongoDB и другие).

Выбор сервисных услуг и оценка стоимости. Организации сервером охватывают платформы на веб-серверах Apache или Nginx. Конечно, они имеют поддержку со стороны операционной системы из Linux. Очевидно, что веб-серверы и операционные системы, которые находятся на базе Линкуса будут бесплатными, что приводит к минимуму конечную стоимость разработки на сервере [15, 21].

Язык PHP необходим для создания сайтов в интернете разной направленности. Программа несёт собой текстовый файл с известным расширением – «.php». Последовательность команд (инструкций) содержится в файле. Эти инструкции выполняет веб-сервер. Язык PHP считывает файл сверху вниз, то есть построчно, затем он выполняет записанные команды.

PHP выполняет код, который находится внутри ограничителей, например, `<?php и?>`. Наименования переменных начинаются со знака \$, указывать вид переменной не требуется. PHP самостоятельно способен к интерпретированию перехода на новую строку в роли пробела. Команды разделяются при помощи точки с запятой, (;), кроме отдельных ситуаций.

Одна из возможностей PHP – это работа с файлами (рис. 12). Возможность даёт перспективу хранения данных внутри вызовами скриптов.

После отправки формы, в скрипте, который был, вызвал появляется перспектива использования значения, которое было передано значением в HTML-коде так: `echo $_REQUEST[«name»]`.

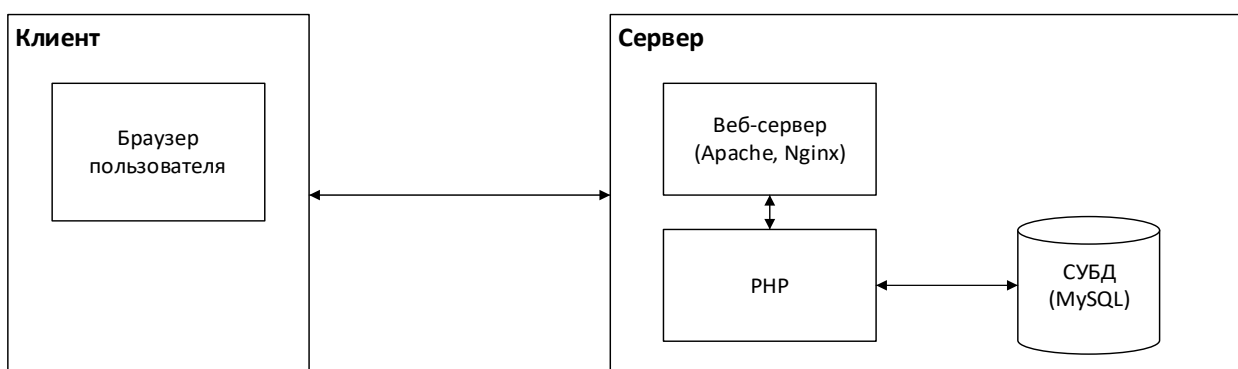


Рисунок 12 - Архитектура веб-приложений, реализуемых на языке PHP

Когда происходит понимание полей формы, есть возможность формирования условий вызова функций для обработки сведений формы и отображения.

Когда дело доходит до отправления макета, `$_REQUEST[«stage»]` при помощи команды `results` получает значение, таким образом, вызывается функция-обработчик формы. В случае, когда сведения не переданы в скрипт (первая загрузка), то получается возможность вывода формы.

Происходит взаимосвязь функции от выбора пользователя создаёт, а также выносит информационную строку.

Рабочий скрипт будет подготовлен только в том случае, если функции, которые были указаны до, будут перемещены в единый файл [8].

Существует не только суперглобальный массив `$_REQUEST` в php. Есть различные массивы ассоциаций, которые дают обработку данных, которые передаёт клиент.

Если прибегнуть к помощи этих возможностей языка программирования, то есть вариант реализации и проектирования клиент – серверного приложения. можно использовать в качестве системы поддержки принятия решений.

2.5 Среда разработки базы знаний

В качестве среды разработки для реализации проекта выбрана IDE PHPStorm. PhpStorm – это взаимосвязанная платформа для моделирования на PHP с умным корректором. Она точно понимает код, имеет возможность поддержки PHP 5.3.-7.4. для традиционных и современных моделей. Он обеспечивает – самое лучшее автоматическое дополнение кода, рефакторинг, дает возможность избежать несоответствий сразу и поддерживает сплетение языков.

Поддержка главных фреймворков. PhpStorm отлично взаимодействует с Symfony, Drupal, WordPress, Zend Framework, Laravel, Magento, Joomla! CakePHP, Yii и с иными фреймворками.

PhpStorm проводит глубокий анализ системы кода и приходит к пониманию кода. Он поддерживает вариативность возможностей PHP, как в усовершенствованных, так и в устаревших проектах. Корректировщик содействует автодополнению кода и изменение факторов [23, 35], сразу же избегает ошибки.

Данная программа поддерживает фронтенд-технологии. В ней есть возможность разработки с современными технологиями. HTML 5, CSS, Sass, Less, Stylus, CoffeeScript, TypeScript, Emmet и JavaScript. Рефакторинг в доступе, отладка и юнит-тестирование. Функция Live Edit позволяет сразу отследить все изменения в браузере в браузере.

Инструменты, которые встроены для проектировщиков. Задачи, которые являются одними и теми же можно выполнять непосредственно в программе. IDE соединяется с системами контрольного всех версий. Она

поддерживает развёртывание на расстоянии, данных и SQL, инструменты командной строки, Docker, Composer, REST-клиент и другое.

PhpStorm содержит в себе весь WebStorm. Он содержит и полное поддержание PHP, баз данных и SQL.

Он, безусловно, заботится об уровне качества отличного кода при помощи тысяч инспекций. Они осуществляют надзор кода мгновенно и переходят к аналитической работе всего проекта [28]. Он поддерживает PHPDoc, code (re)arranger, инструмент редактирования, форматирует коррективы и иные функциональные возможности могут помочь разработчику в написании качественного кода. Такой код будет нетрудно поддерживать.

Благодаря легкой навигации, PhpStorm безопасно изменяет код при помощи проверенных рефакторингов изменения, уничтожения и перемещения, извлечения методов, ввод переменных, изменения положения элементов вверх/вниз, полного изменения сигнатуры и многих других. Изменение факторов учитывает уникальность данного языка, поможет применить изменение по всей модели за несколько движений мыши, стоит учесть, что изменения отменяются.

Налаживание и проверка. Визуально-графический наладчик PhpStorm не претендует дополнительных настроек, отладчик наглядно отражает происходящее в приложении на каждом этапе налаживания. PhpStorm взаимодействует с Xdebug и Zend Debugger, может пользоваться как конкретно, также на расстоянии. IDE поддерживает модельное тестирование с PHPUnit, BDD с Behat и связывается с профилировщиком [5].

2.6 Проектирование функционала и компонентов базы знаний

При описании устройства и реакций многоуровневой системы необходимо разработать огромное число моделей. При этом модели обязательно должны быть связаны. В использовании разработки информационных систем – это необходимость универсального языка для

создания. С помощью такого «универсального» языка становится возможность описывать разные стороны устройства и поведения системы на отдельном уровне её жизненного цикла. От осмысления требований до применения. На данном этапе – это Unified Modeling Language (UML). Unified Modeling Language (UML) – это популярный язык моделирования, который применяется при разработке ПО [53].

Язык моделирования UML – визуальный, графический язык моделирования для общего назначения. Он предназначен для визуального проектирования и работы с документами [32]. Визуализации происходит при работе со всеми компонентами данного аспекта, которые создаются при создании программных систем.

Язык UML – это объектно-ориентированным язык. Область его применения – это построение моделей информационных систем требует понимания общих принципов методологии объектно-ориентированного анализа и проектирования (ООАП). В соответствии с ООАП методология моделирования сложных, в своей структуре, систем складывается из некоторых принципов:

- Принцип абстрагирования обязывает ко включению в модель тех аспектов проектируемой системы, которые относятся к исполнению системой функционала или своего задачного предписания. Все остальные детали не берутся во внимание, чтобы не делать процесс анализа новой модели слишком сложным

- Многомодельность. Значение многомодельности состоит в том, что для объективного проявления требуется рассмотрение сложной системы с разных сторон

- Иерархичность. В основе этого принципа построения макетов сложных систем находится принцип иерархического построения. Он означает, что рассматривать процессы разработки моделей необходимо на различных стадиях. Это необходимо для фиксации представлений.

Подводя итог, можно сказать, что процесс ООАП представлен в виде спуска от общих моделей и представления осмысленного уровня к локальным представлениям физической и логической ступени. Конечно, на каждом из этих уровней происходит дополнение количеством деталей ООАП данных и черты макетов [24, 33]. Всё это даёт возможность объективно рассматривать многие стороны реализации многоуровневой системы.

UML – это язык объектно-направленного моделирования. Он обладает некоторыми характерными чертами:

- Обеспечение поддержки всех уровней цикла жизни ПО;
- Язык зрительного, визуализированного моделирования. Обеспечивает создание репрезентативных моделей;
- Имеет возможности расширения и специализации пользовательских стратегий языка;
- Это одна из типичных нотаций визуальной разработки ПС и имеет поддержку большого числа продуктов CASE, которые ориентированы на объект.

Анализ ориентации на объект и моделирование структуры предугадывает то, что необходимо воспользоваться тезаурусом языка UML, который включает 3 вида строительных блоков: сущности, отношения и диаграммы.

Язык UML содержит 4 вида сущностей: структурные, поведенческие, группирующие, аннотационные, 4 вида взаимодействий: зависимость, ассоциация, обобщение, реализация и восемь типов диаграмм [22].

Диаграмма UML представляет из себя набор элементов, которые графически представлены в виде изображаемой графы с сущностями и ребрами (отношениями), используемые для реализации визуализации системы с разных точек рассмотрения.

Все диаграммы UML можно условно разбить на две группы: общие диаграммы и специальные диаграммы. Общие диаграммы практически не зависят от предмета моделирования и могут применяться в любом

программном проекте. Специальные диаграммы служат для дополнения какой-либо общей диаграммы, например, являются ее частным случаем или играют вспомогательную роль, уточняя некоторые детали.

Как правило, диаграммы раскрывают маленький показ элементов, которые являются составляющими моделирующей системы. Правда, есть и исключения, это простейшие макеты. Всего лишь один элемент системы присутствует во всех диаграммах или в нескольких.

Диаграммы UML – средство для внешнего отражения модели системы, которая находится на стадии разработки. При помощи диаграммы описывается система с абсолютно разных точек зрения. Ни одна диаграмма не будет окончательно для точной и объемной характеристики системы, так как отдельно взятая диаграмма заточена под один уровень функционирования структуры и характеризует его на конкретном шаге абстракции.

Каждая из диаграмм относится как к некоторой частной и ограниченной. Каждая диаграмма соответствует некоторой частной и исключительной точке зрения к разрабатываемой системе. Каждая диаграмма, которая взята отдельно, не является макетом системы. Набор лишь нескольких диаграмм разного варианта выступает как модель системы и вполне полно её описывает.

Диаграммы этого языка широко пользуются спросом при характеристике и описании разнообразных сторон действий и системы ИС на разнообразных этапах реализации, и, конечно, на нескольких шагах моделирования [27, 39].

В момент создания смысловой модели ИС для характеристики автоматических бизнес-процессов можно использовать прецеденты и диаграммы. Такие диаграммы, как диаграммы деятельности. Для описания бизнес – объектов – прецедент – модели бизнес – объектов и последовательные диаграммы.

Давайте более точно разберём этапы создания ИС, диаграммы языка UML, которые используются при проектировании модели системы.

Создание модели прецедентов. В создании системы отправной точкой всегда будет изучение деятельности внешнего компонента. Уровень заканчивает только по окончании реализации диаграмм деятельности для назначенных прецедентов. На следующих шагах анализа и разработки ИС выявляются некоторые параметры автоматизации деятельности объекта, поэтому созданная модель прецедентов в дальнейшем исправляется.

Создание модели бизнес-объектов. Дальнейшим уровнем реализации ИС будет создание макета бизнес – объектов. Она отражает исполнение бизнес – процессов устройства её внутренними исполнителями. Главные составляющие моделей объектов разряда бизнес – объект будут внешние и внутренние исполнители. Уровень можно считать законченным только после создания нужного количества диаграмм последовательности. Конечным результатом шага будет согласование с заказчиком и точные характеристики исполнителей организация, которые собираются внедрить, которые будут обеспечивать исполнение её функционала [38].

Разработка концептуальной (смысловой) модели данных.

3. Разработка концептуальной (смысловой) модели данных. Основываясь на информации, которая получена на предыдущих шагах появляется необходимость в выполнении создания концептуальной модели данных, использующихся в проектируемой системе. Созданные диаграммы – это лишь начало в процессах моделирования базы данных и приложений ИС. Они гарантируют согласованность и взаимосвязь действий бизнес-аналитиков и разработчиков во время продолжительной работы над системой.

Создание требований к системе.

Данный уровень формулировки определенных требований определяется областью использования системой ИС, которая разрабатывается. Создаётся картина о тех способностях системы, которые были бы желаемыми. Модель системных прецедентов – это основа разработки, которая отражает исполнение некоторых обязанностей внешними и внутренними исполнителями, при использовании ИС. Результат выполнения требований к

системе на данном уровне – это не только перечень функций, который является исчерпывающими. Безусловно, они создаются в моделируемой системе. А также описание реализаций данных функций.

Анализ требований и предварительное проектирование системы. Цель этого шага – разработка единого для всех задействованных лиц, создание первоначального проекта ИС, который отвечал бы тем требованиям, которые были озвучены выше. На основании макета системных прецедентов строятся классовые диаграммы, при этом корректируются диаграммы последовательности исполнения конкретных, локальных бизнес – процессов [31]. Результатом данного этапа станет очень точное и развёрнутое описание характеристик и возможностей, создающиеся структуры, при которых используются ИС данные.

При проектировании макетов баз данных и приложений необходимо осуществить отражение компонентов моделей классов, которые были получены ранее в базах данных и приложениях (классы отображаются в таблицы БД, атрибуты классов отображаются в столбцы, в качестве примера). Все макеты баз данных создаются примерно по одной и той же логически выстроенной модели, то происходит автообеспечение данных проектов [29]. А как результат этапа – детальная характеристика проекта базы данных и приложений структуры.

Проект физического создания системы. На этом уровне модели БД и ИС восполняют друг друга описанием размещения на средствах технической стороны, проектируемой системы. Диаграммы развёртывания дают возможность отображения на общей схеме многие составляющие системы (программные и, конечно, информационные), распределение по компонентам комплекса технических средств.

Проведя исследование, можно сделать вывод о том, что диаграммы языка UML создаются и используются как модели ИС на всех её уровнях проектирования и разработки, на всех ступенях цикла жизни системы.

2.6.1 Диаграмма вариантов использования (Use Case)

Для того, чтобы описать функциональные требования применялись функции диаграмм вариантов применения (Use Case), позволяющие показать пользователей системы (актеров) и функции, выполняющиеся каждым актером. Диаграмма вариантов использования, выступающая в качестве первоначального концептуального представления системы в рамках ее создания. Состав этой диаграммы включает в себя актеров, способов применения и связей между ними. В рамках формирования диаграммы могут применяться и общие нотационные элементы: примечания и алгоритмы расширения.

Основная идея этой диаграммы предусматривает следующее: разрабатываемая система должна представляться в форме большого количества актеров, которые взаимодействуют с системой за счет имеющихся вариантов применения [25]. Также отметим, что актером здесь может выступать. Любой вариант обуславливает систему при взаимодействии с актером. Вместе с тем на модель это никоим образом не влияет на то, как будет выполнен сбор действий. Здесь речь идет о разных вариантах взаимодействия. Основные элементы диаграммы - участник (actor) и прецедент (вариант). Участник — это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Участником может быть человек или другая система, подсистема или класс, которые представляют нечто вне сущности. Графически участник изображается “человечком”.

Диаграмма бизнес-прецедентов предназначена для графического описания взаимодействия участников в процессе реализации некоторых вариантов использования. Она помогает представить действия участников процессов и в дальнейшем определить, какие из функций участников могут быть возложены на проектируемую информационную систему. На рис. 13 показан диаграмма бизнес-прецедентов, созданная в программе Software Ideas Modeler.

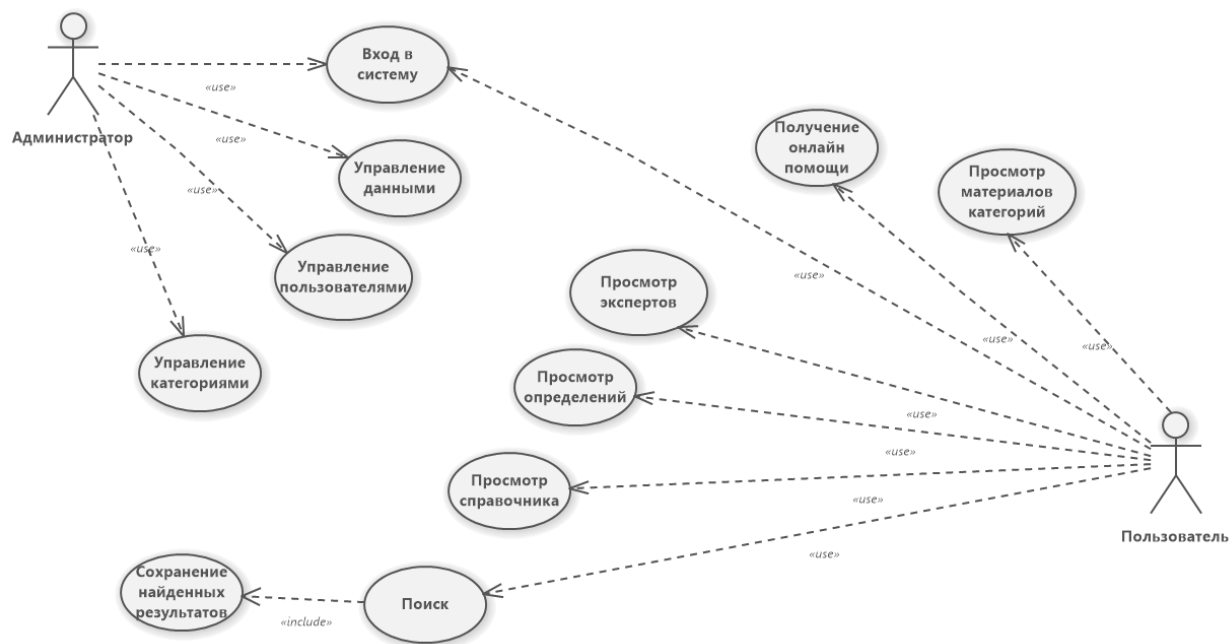


Рисунок 13 - Модель базы знаний в нотации UML

Для описания системы используется модель классов. Модель классов описывает объекты, входящие в состав системы, и отношения между ними. Диаграмма классов отображает классы системы и их взаимоотношения. Диаграммы классов — это основа объектно-ориентированного моделирования. Диаграммы классов используются для отображения того, что делает система (анализ) и как система будет построена (проектирование).

Это набор статических, декларативных элементов модели, таких как классы, интерфейсы и их отношения. Диаграммы могут организовываться в модули (packages). Модули могут включать одну или несколько диаграмм, такое деление создано для удобства доступа к частям визуального представления модели.

Понятие класса связано неразрывно с понятием объекта, т. к. класс является его описанием. Определение классов и объектов - одна из самых сложных задач объектно-ориентированного проектирования. Трудно предложить с первой итерации удачную структуру классов. Поэтому за основу

берется какая-то начальная структура классов, которая постепенно совершенствуется.

На рис. 14 представлена диаграмма классов проектируемой системы, которая включает классы: категория статьи, статья, сохранённые результаты, экспертное мнение, эксперт, пользователь, роль, справочник, категория данных справочника.

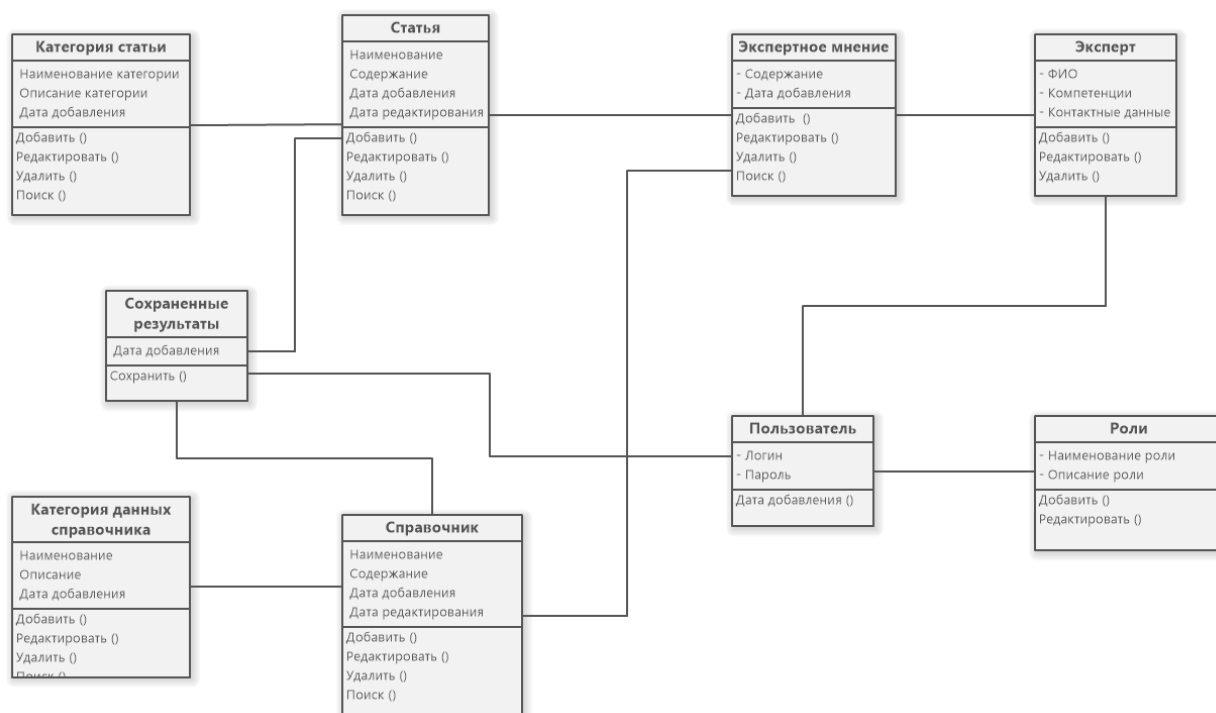


Рисунок 14 – Диаграмма классов базы знаний

Диаграмма классов является дальнейшим развитием концептуальной модели проектируемой информационной системы. Она подразумевает отражение различных взаимосвязей между отдельными сущностями предметной области, а также описание их структуры и типов отношений.

2.6.2 Диаграмма компонентов

Для отображения архитектуры приложения использованы возможности диаграммы компонентов. Диаграмма компонентов была разработана для того,

чтобы распределить классы и объекты согласно их компонентам в рамках физического выстраивания системы. Часто данный тип диаграмм называют диаграммами модулей. При проектировании больших систем может оказаться, что система должна быть разложена на несколько сотен или даже тысяч компонентов, и этот тип диаграмм позволяет не потеряться в обилии модулей и их связей.

Понятие «диаграммы компонентов» характеризуется как вид диаграмм, которые используются в процессе проектирования физических аспектов объектно-ориентированной системы. Данные диаграммы компонентов отражают структуру наборов компонентов и зависимости их друг с другом. Это статическая структурная диаграмма, показывающая разбиение системы на отдельные структурные компоненты и связи между этими компонентами. В качестве компонентов обычно выступают файлы, модули, библиотеки и другие программные компоненты.

Также использование диаграмм компонентов возможно при проектировании статической разновидности системы в контексте ее реализации. Так, оно может содержать организацию физических сущностей, применяемых в системе, в частности, выполняемых программ, библиотек, таблиц, файлов и определенных документов. В результате мы можем сказать, что диаграммы компонентов являются не просто диаграммами классов, которые сосредотачиваются на компонентах системы. Особую роль они играют в процессах визуализации, специфицирования и документирования системы, которая в качестве основы принимает компоненты, а также в процессах формирования исполняемых систем за счет прямого и обратного моделирования [6].

Программная система сервера состоит из базовых модулей, которые включают в себя методы и классы для работы с БД, модулей авторизации и регистрации пользователей в системе. Также она включает методы и классы для осуществления парсинга веб-страниц и сторонние библиотеки, обеспечивающие работу системы.

Таким образом, архитектура программного средства состоит из таких компонентов:

- сервер веб-приложения;
- сервер БД;
- веб-клиент (веб-браузер);
- модуль управления;

Визуально архитектура приложения показана с помощью диаграммы компонентов, которая изображена на рисунке 15.

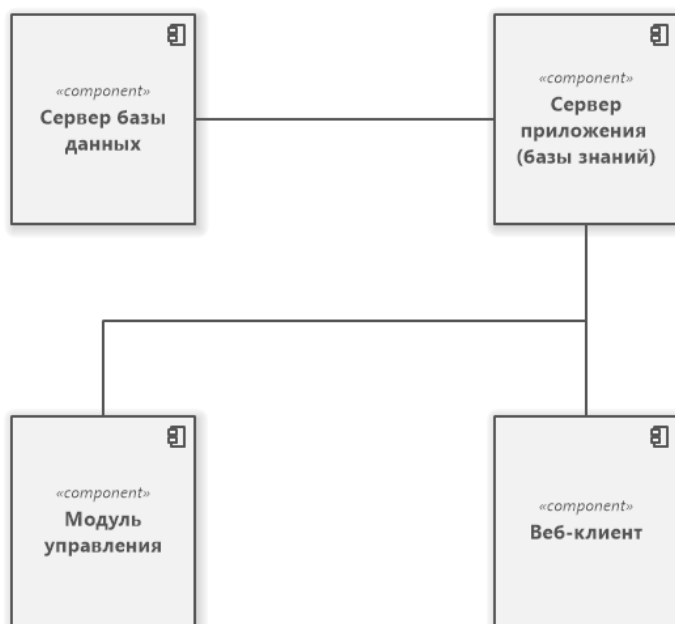


Рисунок 15 – Диаграмма компонентов

Сервер веб-приложения отвечает за обработку запросов клиента и за осуществление процессов извлечения данных со сторонних веб-ресурсов. Веб-сервер отвечает за функционал приложения, который включает авторизацию, регистрацию и другие функции приложения, которые определяют обработку запросов клиента. Взаимодействие веб-сервера с другими компонентами происходит по протоколу HTTP.

Сервер БД осуществляет управление БД, отвечает за целостность и сохранность данных, которые попадают в базу данных с веб-приложения. Веб-сервер формирует запросы к БД, после чего запрос отправляется серверу БД, где обрабатывается, осуществляются необходимые операции и возвращается результат.

Веб-клиент (веб-браузер) обеспечивает возможность взаимодействие пользователя с веб-приложением. Посредством веб-браузера пользователь управляет возможностями приложения.

2.6.3 Диаграмма развертывания

Понятие «диаграмма развертывания» определяется как диаграмма, основная цель которой заключается в представлении узлов осуществления программных компонентов реального времени и процессов с объектами. Использование диаграммы развертывания заключается в отражении общей конфигурации и топологии распределенной системы, также она данный процесс включает в себя графическое изображение размещения компонентов по каждому узлу системы в отдельности.

Основное предназначение диаграммы развертывания заключается в визуализации программных элементов и компонентов, которые присутствуют только на стадии реализации программы. Однако отметим, что здесь происходит выступают как исполнимые файлы либо библиотеки динамического типа. Касаясь компонентов, что не применяются на стадии реализации, отметим, что на диаграмме развертывания они попросту отсутствуют. Таким образом, компоненты с исходными текстами программ как правило, представляются исключительно на диаграмме компонентов, а на диаграмме развертывания их не представляют.

Диаграмма развертывания также включает в себя графические изображения процессоров, устройств, процессов и их взаимосвязей. Вместе с тем диаграмма развертывания выступает как единая диаграмма для системы в целом, так как ей требуется полностью отображать специфику ее исполнения.

Данная диаграмма, как правило, приводит к завершению процессу для определенной программной системы, и ее проектирование зачастую выступает как завершающая стадия спецификации модели.

Цели, которые установлены в рамках создания диаграммы развертывания:

- установка распределения компонентов системы в соответствии с ее физическими узлами;
- отображение физических связей между всеми узлами осуществления системы на стадии ее реализации;
- выявление узкие места системы и реконфигурация ее топологии для того, чтобы прийти к необходимой производительности.

На рисунке 16 можно рассмотреть, как выглядит диаграмма развертывания. Пользователь за счет использования браузер выполняет обращение на сервер, где находится модуль управления и модули сохранения и поиска данных.

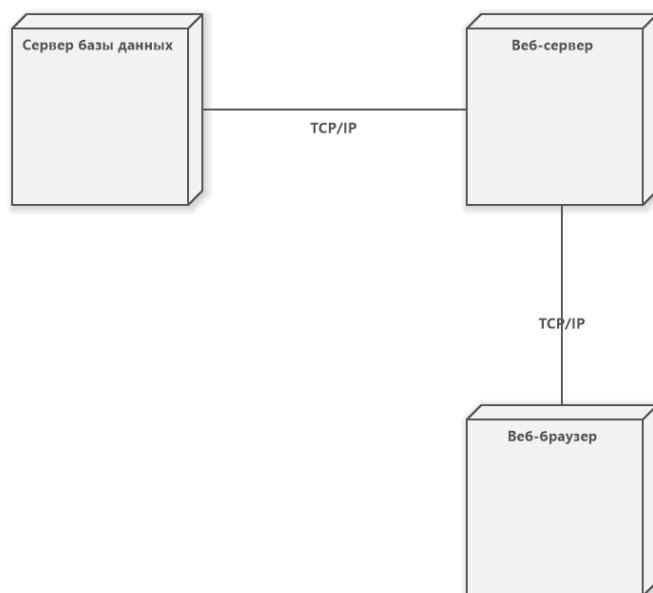


Рисунок 16 – Диаграмма развертывания

Через интерфейс БД происходит общение с сервером БД, посредством программных библиотек осуществляется обработка данных, которые сохраняются в БД. Система состоит из 3 узлов: веб-сервер, сервер БД, веб-браузер. Веб-браузер обращается к веб-серверу, который обрабатывает запросы клиента и возвращает ответ. В свою очередь веб-сервер обращается к серверу БД и серверу навигации. Взаимодействие веб-сервера с сервером БД обеспечивает сохранение и обработку данных, которые вносятся в базу знаний.

2.7 Проектирование базы данных

Под «базой данных» (БД) принято рассматривать набор объединенных данных, который представляет состояние объектов и их отношение в рассматриваемой предметной области. БД — это фундамент каждой создаваемой информационной системы.

Процесс создания базы данных включает в себя 4 стадии:

Системный анализ и словесное описание информационных объектов предметной области. На данном этапе главная задача заключается в сборе требований, которые выставлены в отношении пользователей. Объединяя представления о том, какие данные содержит в себе база данных, которые были получены в ходе опроса пользователей, и свои представления о данных, что потребуются в разрабатываемых программах, программисты формируют обобщенное неформальное описание базы данных.

Проектирование инфологической модели. Инфологическая модель представляет собой формализованное описание объектов предметной области в терминах определенной семантической модели, к примеру - модели. Этот этап заключается в формировании концептуальной модели предметной области, конкретизированной в будущем. Инфологическая модель абсолютно не зависит от физических параметров среды хранения данных.

Логическое проектирование БД. Характеризуется как представление и трактовка в терминах принятой логической модели данных. Эта стадия предусматривает выбор типа системы управления БД (СУБД), формируется схема базы данных в терминах соответствующей модели данных, формируется комплекс всевозможных типовых запросов и создается спецификация для ПО.

Физическое проектирование БД. Физическое проектирование БД предусматривает выбор варианта её размещения на внешних носителях для того, чтобы обеспечить более эффективную работу приложений информационной системы. Также здесь устанавливается конкретная СУБД и формируется уже реальная база данных.

Представить сведения, которые присутствуют в базе данных, можно при определенных ограничениях, которые обусловлены применяемой информационной системой и выбранной логической и физической структурами организации данных. Данные ограничения устанавливают все допустимые типы данных, взаимосвязи данных с операциями, выполняемых в отношении данных и связей. Кроме того, есть ограничения, которые обуславливают целостность БД.

Задачей данного пункта является проектирование логической модели. После того, как были рассмотрены специфика и стадии формирования модели БД, сформируем логическую модель БД для информационной системы.

Для логического моделирования БД была применена MySQL Workbench. С помощью этой программы можно наглядным образом представить сложно образованные структуры данных.

На рис. 17 показана логическая модель базы данных.

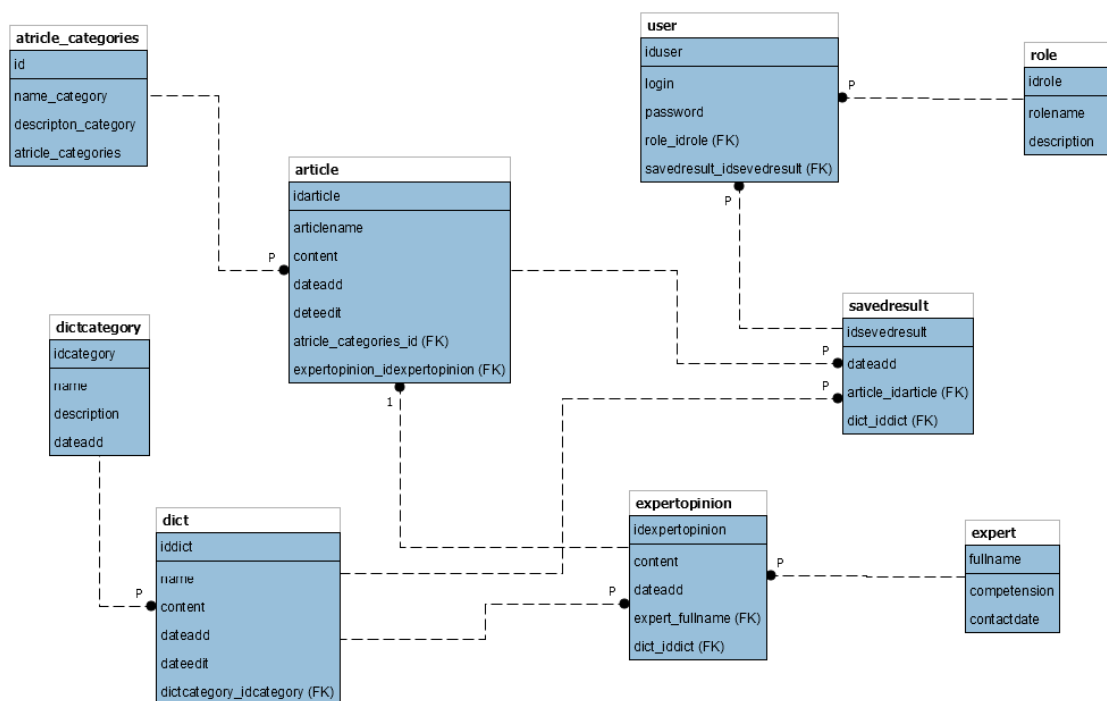


Рисунок 17 – Логическая модель базы данных

Как видно с рисунка, база данных включает следующие сущности, с помощью которых будет функционировать база знаний: категория статьи, статья, сохранённые результаты, экспертное мнение, эксперт, пользователь, роль, справочник, категория данных справочника.

Выводы по разделу 2. Во второй главе реализована архитектура и структура знаний. MySQL выбрана для проекта так же, как и СУБД, что обоснованно путем анализирования различных СУБД. В качестве языка программирования базы знаний информационной системы поддержки принятия решений использован PHP. В качестве среды разработки для реализации проекта выбрана PHPStorm. Спроектирована база данных. Определена роль базы данных в определении базы знаний, как ее структурного компонента.

3. Реализация базы знаний информационной системы поддержки принятия решений

3.1 Физическая модель базы данных

Физический уровень (представление администратора) проектирования базы данных представляется как группировка данных, индексы, методики доступа.

Необходимо, чтобы физическая модель данных в полной мере соответствовала описанию данных в БД той или иной системы управления БД, т.е. соответствовала схеме данных. Особенности определенной системы управления БД заключаются в выставлении ограничений на именование объектов БД и ограничений на те категории данных, которые поддерживаются в данном случае. Вместе с тем особенности конкретной системы управления БД при физическом моделировании (проектировании) состоит из выбора решений, взаимосвязанных с физической средой хранения данных (выбор методик управления памятью на диске, группирование базы данных согласно видам файлов и устройств, способов доступа к данным), а также из процесса по формированию индексов [5].

После проектирования базы данных необходимо создать скрипт, для создания базы данных на сервере MySQL представлен в приложении А.

После создания базы данных требуется рассмотреть вопросы целостности данных. Целостность данных представляет собой определенный свод правил, которые применяются для поддержки взаимосвязей записей друг с другом в связанных таблицах, и обеспечивающий защиту от возможного и случайного удаления либо изменения взаимосвязанных между собой данных.

В разработанной БД (рис. 18) целостность данных осуществляется посредством ограничений для полей, а также ограничений для связей. Каждое поле, которое является идентификатором, обладает типом данных int, которые в обязательном порядке нужно заполнять (NOT NULL). Связи

таблиц друг с другом в БД не дают разрешение на удаление полей со связанных главных страниц, в том случае, когда в подчиненной таблице имеется запись, кроме того, не допустимо внесение записей создание записей в подчиненной таблице, когда в поле вводится значение, отсутствующее в основной таблице.

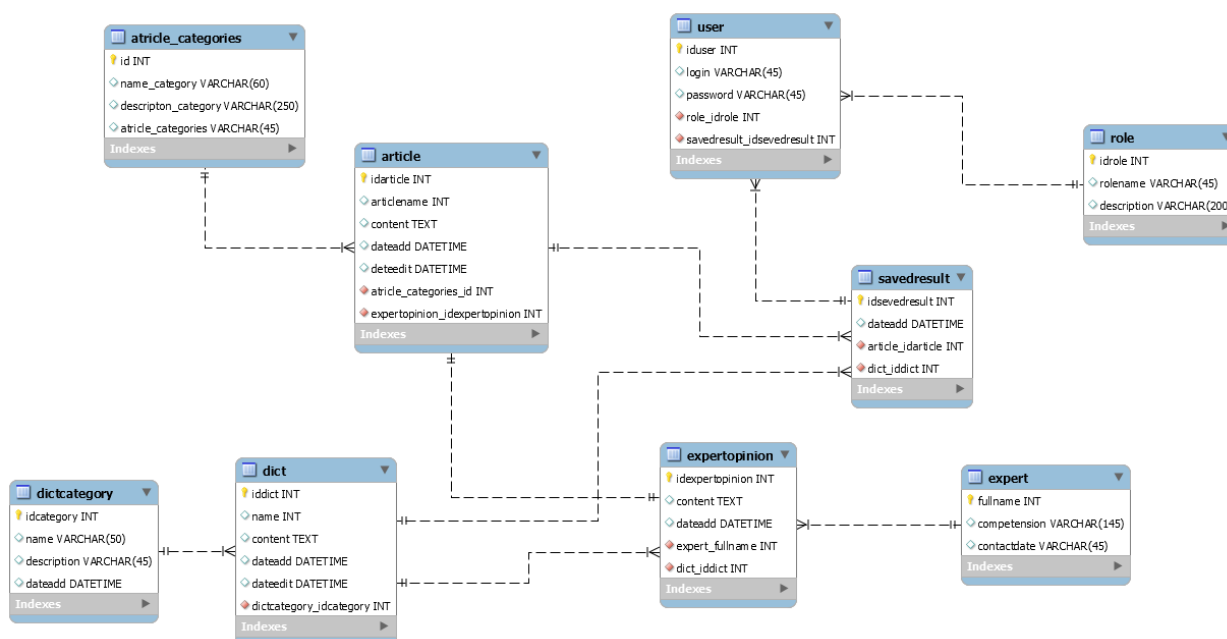


Рисунок 18 –Схема базы данных на физическом уровне

Одним из важнейших условий безопасного и продуктивного применения БД MySQL выступает верное использование системы прав доступа, которая существует в данной БД, а также грамотное применение инструментальных средств, которые направлены на управление правами доступа пользователей.

БД MySQL дает возможность крайне точно устанавливать права доступа разных пользователей, подключающихся в БД через использование клиентских программ, которые находятся в разных точках системы. Права доступа разделяются на 4 нисходящих уровня: глобальные права доступа, права доступа БД, права доступа к таблицам и права доступа к столбцам. В связи с этим с теоретической точки зрения присутствует возможность

выполнения регулирования доступа определенного пользователя для записи данных исключительно в указанные столбцы указанных таблиц указанных БД на указанном сервере MySQL. Также просто возможно представить каждому пользователю БД, который подключается в независимости от местонахождения, те же права, что и пользователю БД root (однако подобная организация защиты доступа не рекомендована).

В соответствии с требованиями защиты необходимо, чтобы каждому пользователю были предоставлены лишь минимальные права доступа, в отсутствии которых у него бы не было возможности реализовывать свой функционал. Однако здесь важно принимать во внимание обратную сторону данного процесса: насколько лучше детализирована используемая схема прав доступа, настолько спадает скорость выполнения каждого оператора INSERT, SELECT, UPDATE и DELETE. Это обуславливается тем, что при условии наличия наиболее подробно регламентированных прав доступа в БД MySQL необходимо проверять множество таблиц с данными о правах доступа. Тем не менее, на самом деле наблюдаемое здесь понижение производительности отрицательным образом сказывается на работе далеко не каждого узла, однако в случае, что подобное воздействие начинает быть более ощутимым, требуется найти какой-либо компромисс между степенью защиты и производительностью.

В качестве основы системы прав доступа MySQL выступает таблица, которая должны сопровождаться администратором БД — таблица пользовательской БД MySQL (эта БД, которая также как БД test, поставляется с каждой инсталляцией MySQL).

В случае, когда на сервер MySQL поступает запрос от пользователя, для представляющейся под значением N в поле, которое соответствует действию, попытка выполнения которого предпринята пользователем, сервер приступает к прохождению вниз по иерархии уровней прав доступа. Несмотря на то, что права доступа на уровне столбца либо таблицы могут быть представлены исключительно одному среди множества. Также в системе MySQL есть

возможность проверить указанные таблицы прав доступа для каждого пользователя. Так, именно по этой причине, представление прав доступа к столбцу либо таблице даже одному пользователю окажет заметное влияние на снижение скорости всех операторов SQL каждого пользователя.

Для того, чтобы добавить или же редактировать права доступа пользователей в БД MySQL часто используют два разных способа (при соблюдении такого условия, как модификация прав доступа выполняется пользователь БД root): исполнение операторов SQL (к примеру, ввод буквы Y ручным образом в каждое соответствующее поле каждой соответствующей таблицы прав доступа) либо применение синтаксических конструкций под названием GRANT и REVOKE. Описанный вариант наиболее прост и не так опасен, в случае, что была выполнена ошибка, так как, как правило, попытка выполнить простой ошибочный запрос завершится неудачей с сообщением об ошибке SQL, однако в системе защиты в данном случае не появится пробел.

Для введения данных о вновь зарегистрированном пользователе MySQL, как правило, применяется:

```
GRANT priv_type [(column1, column2, column3)]  
ON database.[table]  
TO user@host IDENTIFIED BY 'new_password';
```

Отметим, что для предоставления конкретному пользователю права доступа SHUTDOWN на уровне таблицы нет необходимости. Вместе с тем формируется сообщение об ошибке, где сразу предложено изучить соответствующие документы. В случае, что представлены права доступа ALL на уровне столбца, таблицы или БД, у пользователя возникает право на использование исключительно того набора прав доступа, что в полной мере соответствует указанному уровню.

Особенно важно быть осторожными, предоставляя пользователям следующих прав доступа, так как абсолютно все эти права носят опасный характер. К этим правам относятся: GRANT, ALTER, CREATE, DROP, FILE,

SHUTDOWN, PROCESS. Также отметим, что на производстве подобные права доступа обычным пользователям БД абсолютно не нужны.

Синтаксическая конструкция оператора отмены прав доступа подобна соответствующему оператору предоставления прав доступа, однако имеет наиболее упрощенное строение:

```
REVOKE priv_type [(column1, column2, column3)]  
ON database[.table]  
FROM user@host;
```

После того, как было права доступа были представлены либо отменены, каждому пользователю нужно перезагрузить БД для ввода в память новой информации о правах доступа. В данном случае нужно применить команду FLUSH PRIVILEGES. Таким же образом возможно выполнит остановку и запуск сервера, однако подобное решение в большинстве случаев на практике часто не применяется.

В БД проекта ее безопасность обеспечивается посредством ограничения прав пользователя БД для той составной части web-сайта, которая является публичной. Как правило, у пользователя для публичной части сайта есть только права, дающие разрешение на осуществление операции SELECT и INSERT для таблицы, сохраняющей данные заявок, что были созданы посредством обращения на web- сайт.

3.2 Реализация модуля базы знаний информационной системы поддержки принятия решений

Реализация модуля включает реализацию алгоритмов и создание на основании алгоритмов программного кода. Ниже приведены схемы алгоритмов и показан процесс реализации программного кода, с помощью которого создается функционал приложения базы знаний системы поддержки принятия решений.

На рисунке 19 представлен алгоритм регистрации пользователя для применения БД ИС поддержки принятия решений (рис. 19).

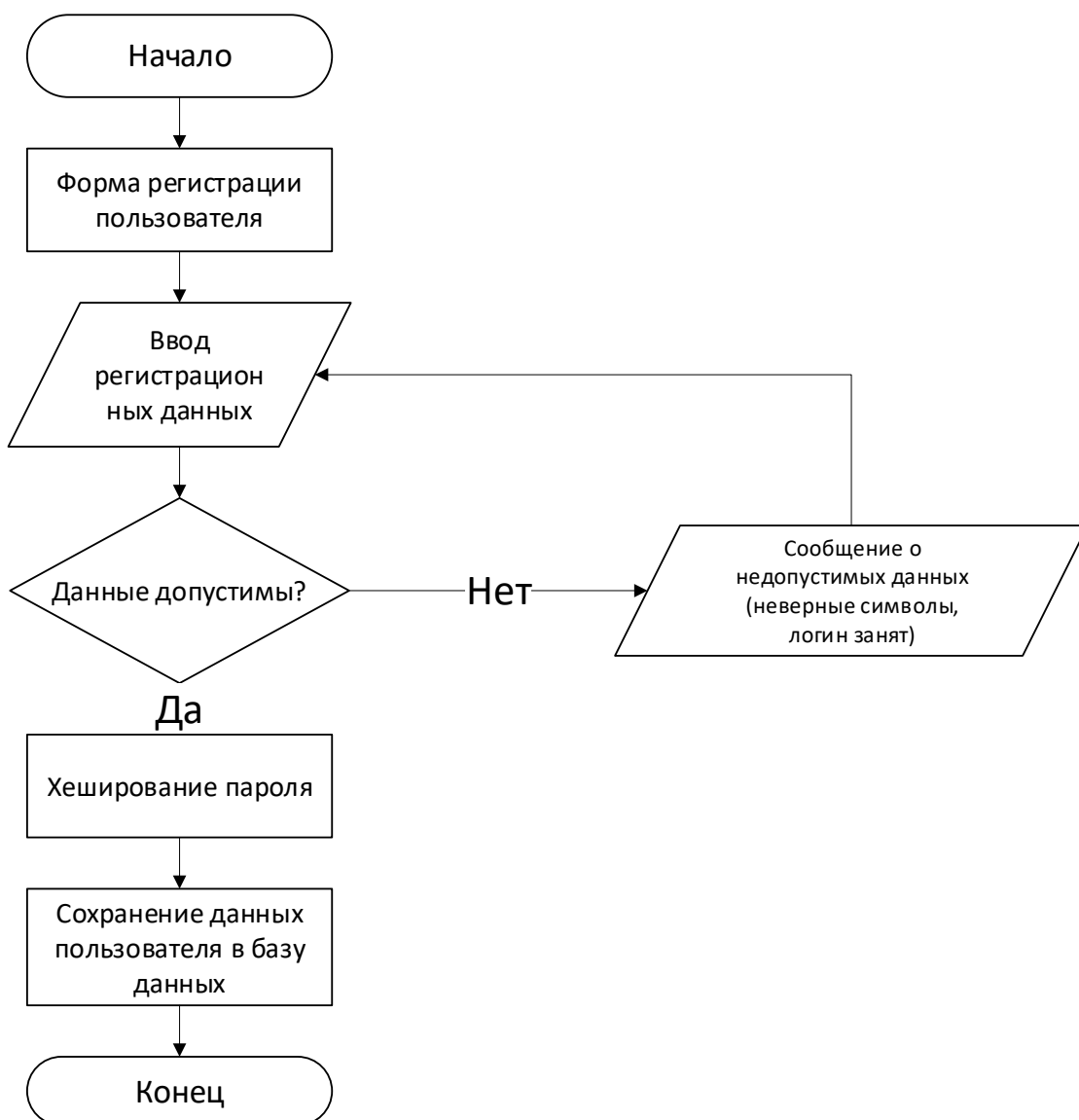


Рисунок 19 – Схема алгоритма регистрации в системе

Алгоритм пользовательской регистрации — это важнейший элемент современного web-приложения, посредством которого можно осуществлять взаимодействие с гостями web-сайта. Благодаря процессу регистрации пользователь может внести в программу свою информацию. За счет этого программа сможет его на фоне других пользователей, также программа даст доступ к тем или иным функциональным возможностям, а также предоставит

возможности для определенного пользователя во время осуществления авторизации по данным, которые были введены при регистрации. Когда пользователь открывает регистрационное окно, оно представляется перед ним в виде формы регистрации с несколькими полями. Как только пользователь внесет свои личные данные, программа начинает осуществлять проверку на правильность их написания, а также на то, существует ли в системе пользователь с такими данными или нет.

В случае, что информация внесена неправильно, на экране всплывает информационное окно о том, что данные недопустимы. Как правило, это случается в случае, что неверно заполнено то или иное поле, либо информация данного пользователя уже присутствует в системе. В случае, что информация, которую внес пользователь, допустима, программа генерирует пароль и сохраняет данные в БД.

Следующий шаг после регистрации пользователя в системе, выступает алгоритм, позволяющий применять информацию, введенную при регистрации, служит алгоритм авторизации пользователей.

На рисунке 20 представлен алгоритм авторизации и способ его реализации (рис. 20).

Выполняя вход в систему на экране, всплывает форма с полями для ввода личных данных, которые включают в себя логин и пароль. Далее необходимо ввести эти данные, после чего проверяется верно ли они введены. Если данные введены неверно или какое-либо поле не заполнено, система отправляет сообщение о необходимости заполнить все имеющиеся поля. После введения данных начинается процесс, осуществляющий поиск пользователя. В случае, что пользователь не найден, нужно идти в конец, иначе вход в систему и установление сессии невозможно.

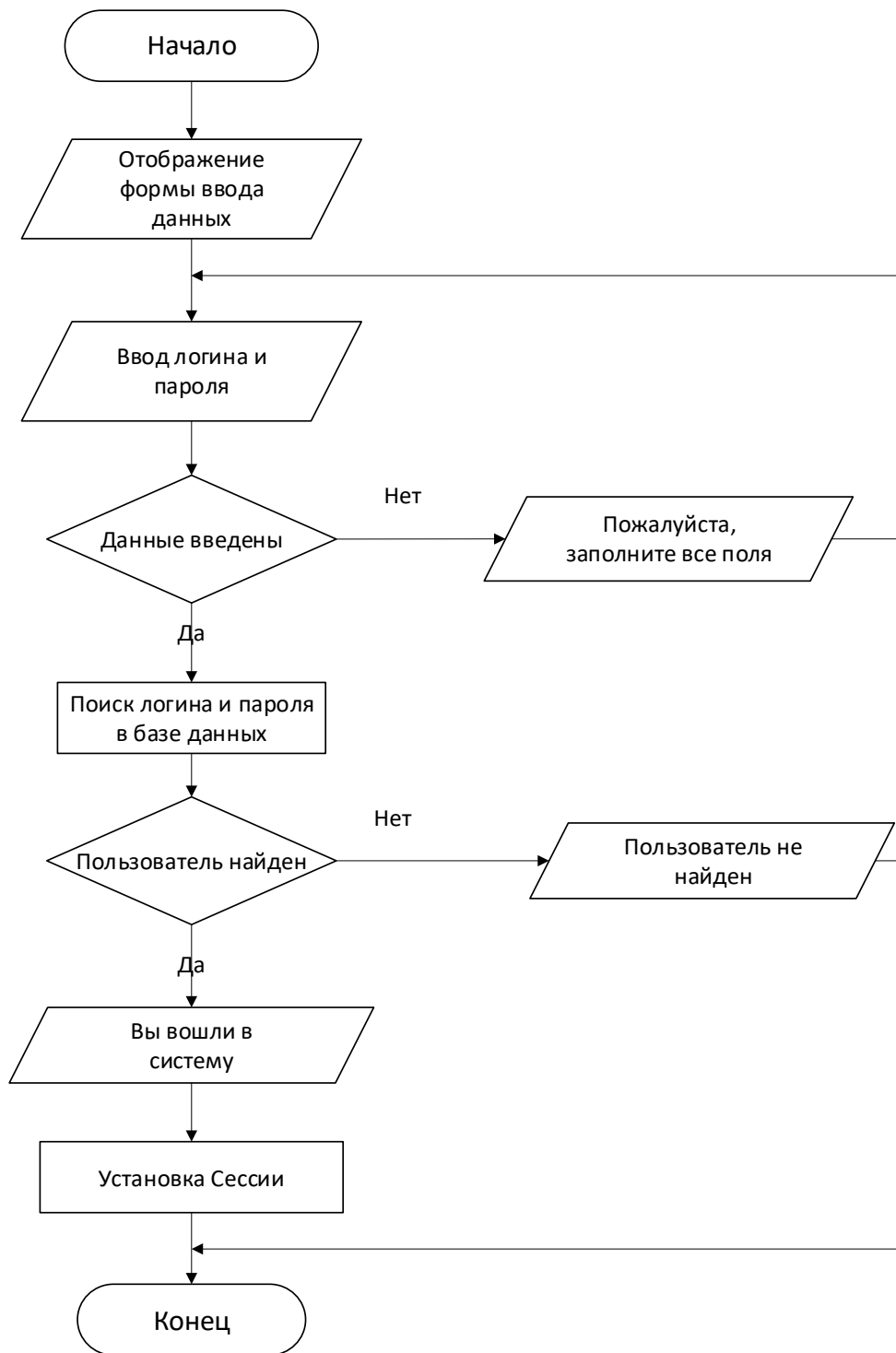


Рисунок 20 – Схема алгоритма авторизации в системе

В качестве одной из наиважнейших функций, позволяющих устанавливать в системе необходимые сведения о предметной области, выступает внесение записей в базу знаний. На рисунке 21 представлена схема алгоритма добавления записей в базу знаний.

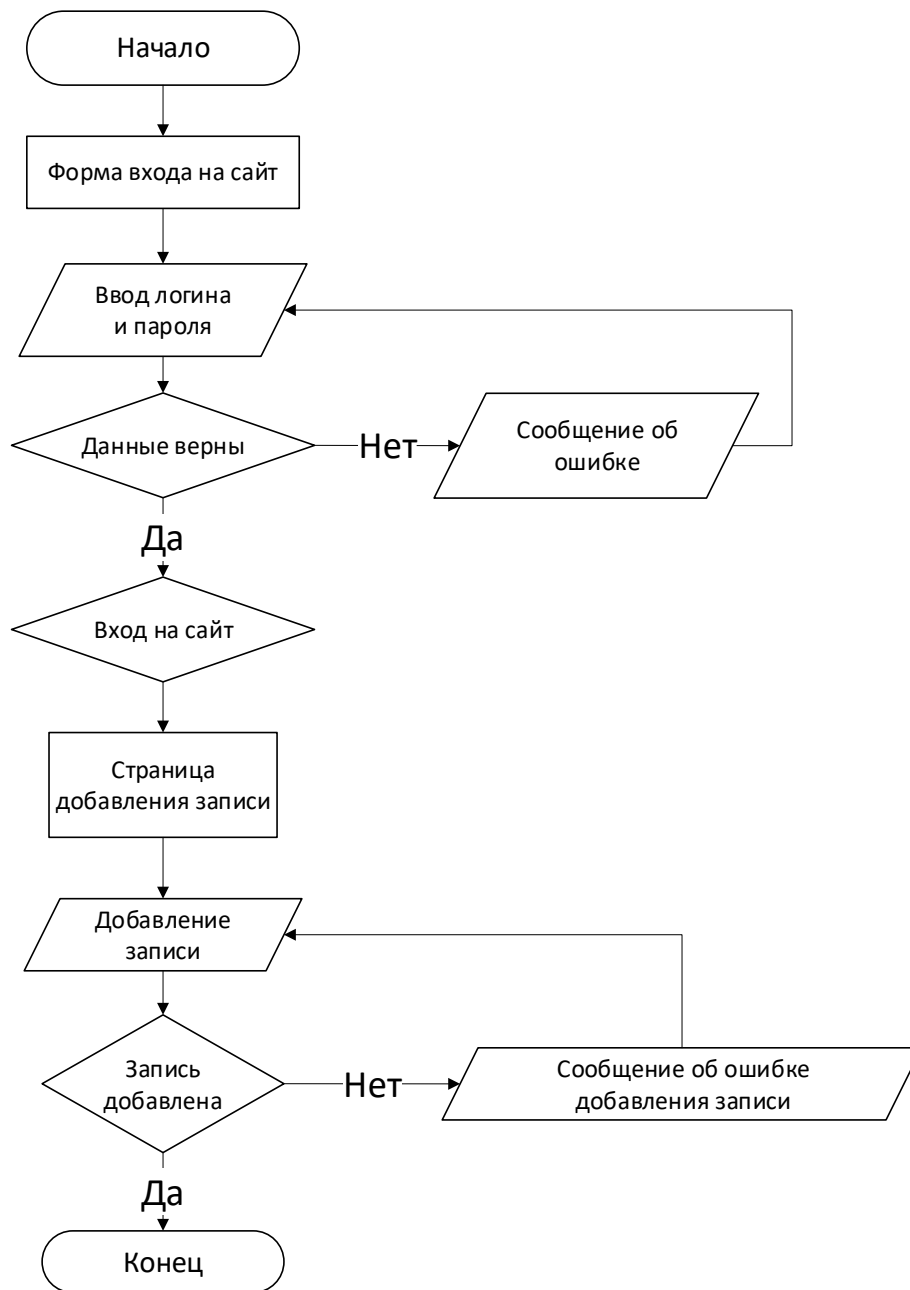


Рисунок 21 – Схема алгоритма добавления записи в базу знаний

По этой причине алгоритм добавления записей это, можно сказать, важнейший компонент web-сайта –«Добавление записей». Перед тем как разместить запись требуется авторизация в системе. На экране компьютера всплывает окно с формой входа, где необходимо ввести логин и пароль. Далее проверяется достоверность ввода данных, и в случае, что они верны, осуществляется вход в систему. Затем можно выполнить переход на страницу с добавлением записи. Далее всплывает окно, где представлена форма с

добавлением записи с соответствующими полями для ввода. После того, как осуществлен набор соответствующего текста в запись, возможно добавление записи. В процессе добавления записи проверяется ее верность написание. В случае, когда запись не была добавлена, всплывает окно с указанием ошибки добавления записи. В обратном случае - запись добавляется.

В работе системы был реализован следующий алгоритм (рисунок 22).

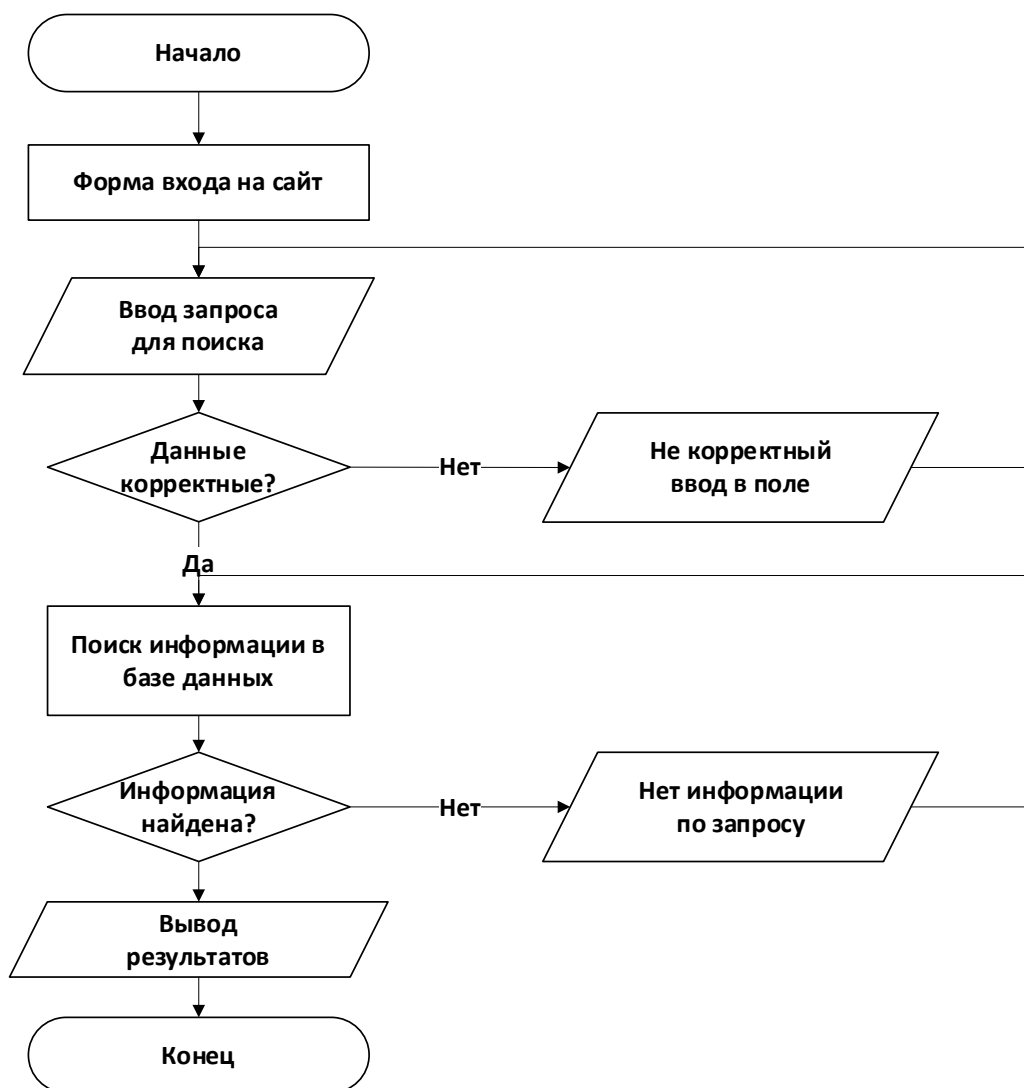
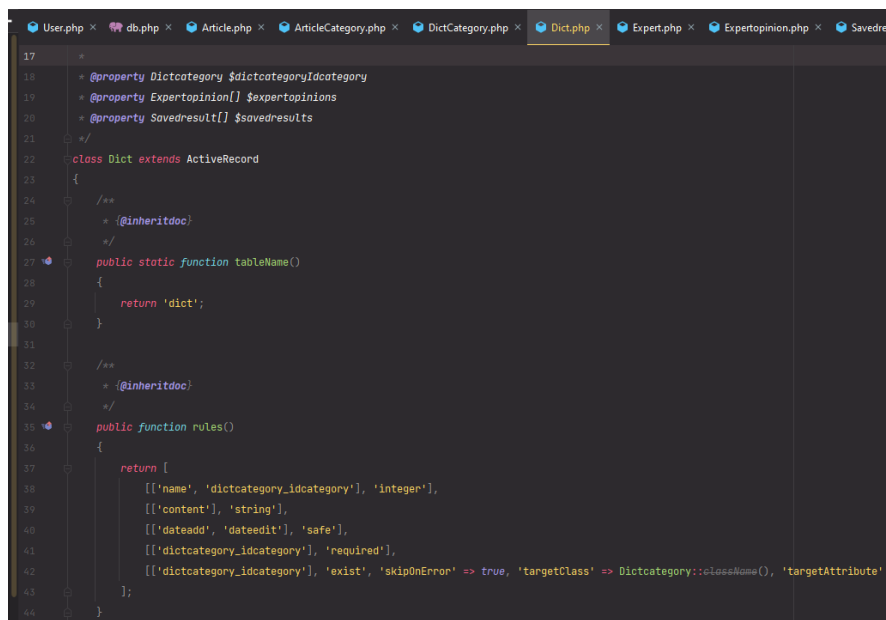


Рисунок 22 – Алгоритм поиска по базе знаний

Одной из целей программного комплекса является возможность поиска информации по определенным критериям в проектируемой базе знаний информационной системы поддержки принятия решений.

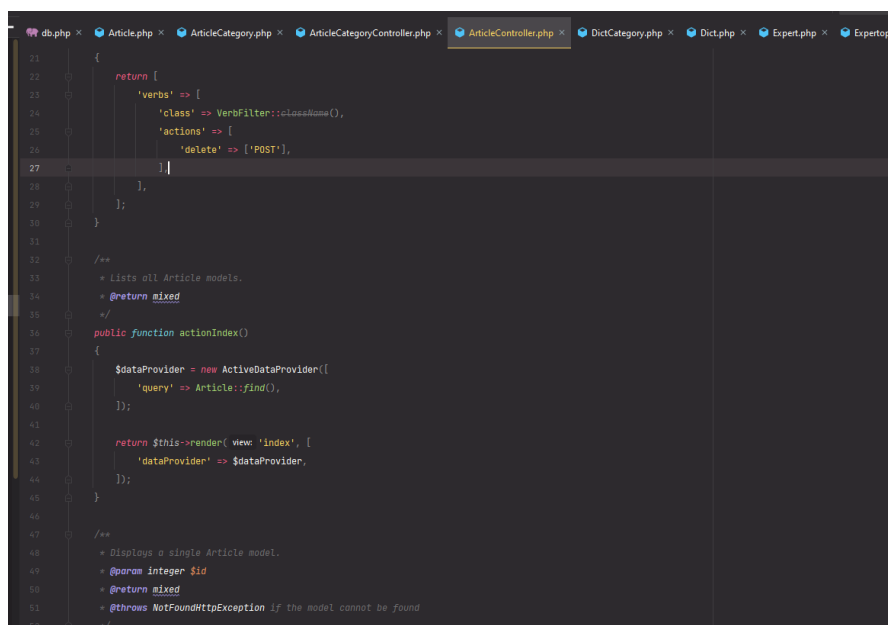
Если пользователь задает критерии поиска, то происходит фильтрация записей по выбранным критериям и вывод обнаруженных данных.

На основании представленных алгоритмов был создан программный код базы знаний. Процесс работы над исходным кодом показан на рис 23–25.



```
17 *
18 * @property Dictcategory $dictcategoryIdcategory
19 * @property Expertopinion[] $expertopinions
20 * @property Savedresult[] $savedresults
21 */
22 class Dict extends ActiveRecord
23 {
24     /**
25      * @inheritdoc
26      */
27     public static function tableName()
28     {
29         return 'dict';
30     }
31
32     /**
33      * @inheritdoc
34      */
35     public function rules()
36     {
37         return [
38             [['name', 'dictcategory_idcategory'], 'integer'],
39             [['content'], 'string'],
40             [['dateadd', 'dateedit'], 'safe'],
41             [['dictcategory_idcategory'], 'required'],
42             [['dictcategory_idcategory'], 'exist', 'skipOnError' => true, 'targetClass' => Dictcategory::className(), 'targetAttribute'
43             ];
44         ];
45     }
46 }
```

Рисунок 23 – Работа с моделью приложения



```
21 {
22     return [
23         'verbs' => [
24             'class' => VerbFilter::className(),
25             'actions' => [
26                 'delete' => ['POST'],
27             ],
28         ],
29     ];
30 }
31
32 /**
33  * Lists all Article models.
34  * @return mixed
35  */
36 public function actionIndex()
37 {
38     $dataProvider = new ActiveRecordDataProvider([
39         'query' => Article::find(),
40     ]);
41
42     return $this->render( view: 'index', [
43         'dataProvider' => $dataProvider,
44     ]);
45 }
46
47 /**
48  * Displays a single Article model.
49  * @param integer $id
50  * @return mixed
51  * @throws NotFoundHttpException if the model cannot be found
52  */
```

Рисунок 24 – Работа с контроллерам приложения

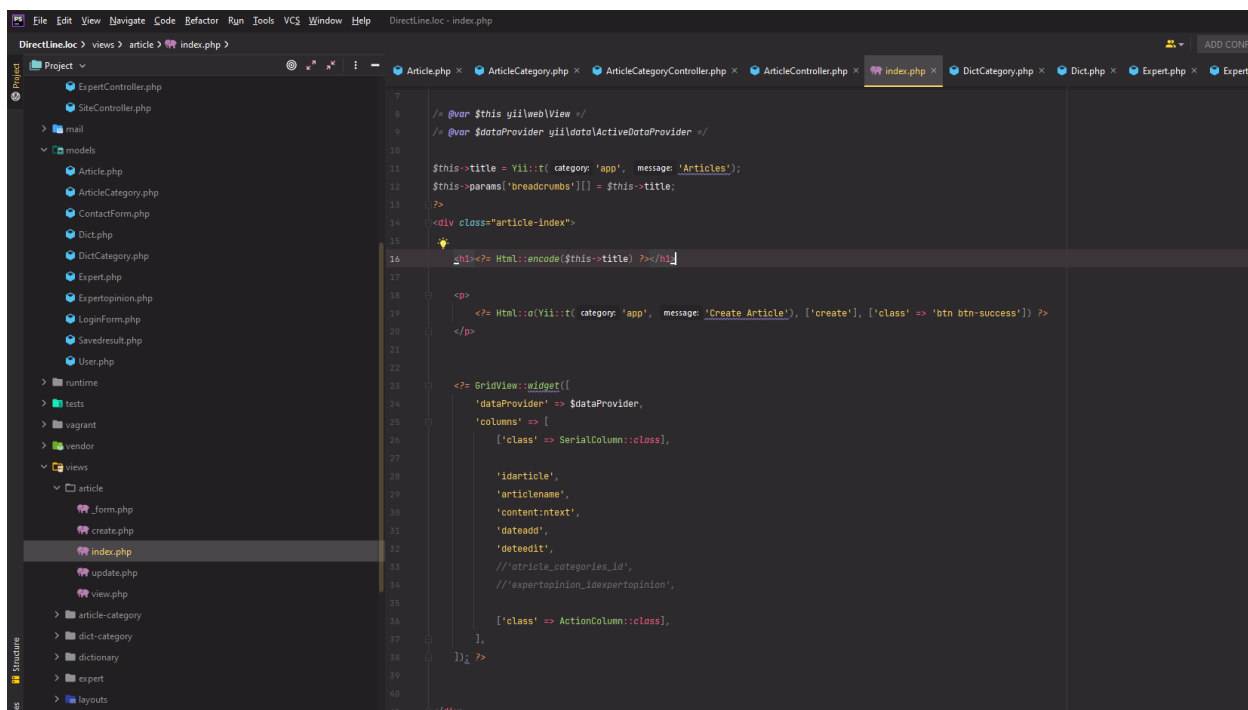


Рисунок 25 – Работа с представлением приложения

В результате реализации модулей была создана база знаний системы поддержки принятия решений.

3.3 Тестирование созданного приложения

При проведении тестирования ПО, требуется выполнить такие тесты, как:

- тесты функционального типа;
- Конфигурационные тесты;
- тест юзабилити;
- тест сборки.

Тест функционального типа заключается в проверке исполнения программой того функционала, что содержится в техзадании и в проверке уровня точности осуществления тестируемых функций.

В рамках теста конфигурационного типа осуществляется проверка того, насколько приложение способна функционировать на разных компьютерных устройствах и с разным установленным ПО.

Тест юзабилити программы может установить, насколько оно удобно в эксплуатации.

Тест сборки это по своей сути то же «дымовой» тест, реализующийся с целью подтверждения того, что после запуска программы, она в полной мере способна исполнять свои функциональные возможности. Как правило, данный тест, проводят программисты после того, как приложение уже разработано. В случае, когда приложение не проходит все этапы тестирования, тогда отпадает необходимость проведения последующего тестирования, поскольку данные тесты уже выявили недостатки программы, а точнее то, что в ней не работает должным образом.

В качестве примера недоработки программы и не прохождения ею тестирования выступает то, что она не смогла запуститься, или присутствует ошибка подключения к БД. Говоря о таком понятии, как тестовый случай, отметим, по сути, он выступает в качестве последовательных действий, ряда определенных факторов и параметров, требующихся в обязательном порядке для того, чтобы проверить созданное приложение или программу. Тестовые сценарии, предназначенные для проведения тестирования в отношении программных модулей, можно подробно рассмотреть в таблицах 5–9.

Тестирование программного обеспечения представляет собой процесс выявления наличия дефектов в программных системах.

Тестирование имеет большое значение в силу того, что этот процесс существенно содействует тому, чтобы конкретное программное приложение делало именно то, что ожидают от него проектировщики.

Для тестирования данной системы было выбрано функциональное тестирование. В результате тестирования была сформирована таблица тестовых случаев (таблица 3).

Таблица 3 - Тестирование базы знаний

№ тестового случая	Название теста	Ожидаемый результат	Полученный результат	Результат тестирования
1	Войти в административную часть (неправильные параметры входа)	Ошибка входа в административную часть сайта	Ошибка входа в административную часть сайта	Тест пройден успешно
2	Войти в административную часть (правильные параметры входа)	Вход в административную часть сайта	Успешный вход в консоль управления сайтом	Тест пройден успешно
3	Переход в модуль поиска данных	Переход в модуль поиска данных	Открыта страница базы знаний с формой поиска	Тест пройден успешно
4	Поиск информации по базе данных	Поиск информации в базе знаний	Информация по запросу найдена	Тест пройден успешно
5	Добавление категорий	Добавление категорий на страницы	Добавление категорий на сайт	Тест пройден успешно
6	Добавление пользователя в систему	Добавить пользователя в систему	Новый пользователь добавлен	Тест пройден успешно
7	Добавление эксперта	Добавление эксперта	Добавление экспертов на сайт	Тест пройден успешно
8	Просмотреть информацию в базе знаний	Отображение списка понятий	Просмотр списка понятий	Тест пройден успешно
10	Добавление записи (понятия) в базу знаний	Понятие добавлено в систему	Понятие добавлено в систему	Тест пройден успешно

Внешний вид (интерфейс пользователя) рабочих окон создаваемого приложения показан на рис. 26–28.

База знаний Директ Лайн

Войти в панель Базы знаний

Email / Username

Password

Remember me?

Рисунок 26 – Вход в систему

Категории информации в базе знаний

Добавить запись

Наименование категории	Описание категории	Дата добавления	
Информационное обеспечение	Информационное обеспечение	5/28/2021 12:00:00 AM	Редактировать Удалить
Ремонт технического оборудования	Ремонт технического оборудования	5/26/2021 12:00:00 AM	Редактировать Удалить
Продвижение сайтов	Продвижение сайтов	6/10/2021 12:00:00 AM	Редактировать Удалить
Разработка мобильных приложений	Разработка мобильных приложений	6/1/2021 12:00:00 AM	Редактировать Удалить
Закупки программных средств	Закупки программных средств	6/1/2021 12:00:00 AM	Редактировать Удалить
Проектирование информационных систем	Проектирование информационных систем	5/26/2021 12:00:00 AM	Редактировать Удалить

Рисунок 27 – Категории информации в базе знаний

База знаний

Добавить понятие в базу знаний

Наименование	Описание	Дата добавления	Дата редактирования	Категория
База данных	Совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняются в соответствии с правилами средств моделирования данных	6/9/2021 12:00:00 AM	6/9/2021 12:00:00 AM	Базы данных
СУБД	СУБД – комплекс программ, позволяющих создать базу данных (БД) и манипулировать данными (вставлять, обновлять, удалять и выбирать). Система обеспечивает безопасность, надежность хранения и целостность данных, а также предоставляет средства для администрирования БД	6/9/2021 12:00:00 AM		Базы данных
Сервер	Сервер (англ. server от англ. to serve – служить, мн. ч. серверы) – выделенный или специализированный компьютер для выполнения сервисного программного обеспечения (в том числе серверов тех или иных задач).	6/9/2021 12:00:00 AM		Аппаратное оборудование

Рисунок 28 – Список записей в базе знаний

Добавляют записи (понятия) в систему эксперты, пример добавления записи показан на рис. 29.

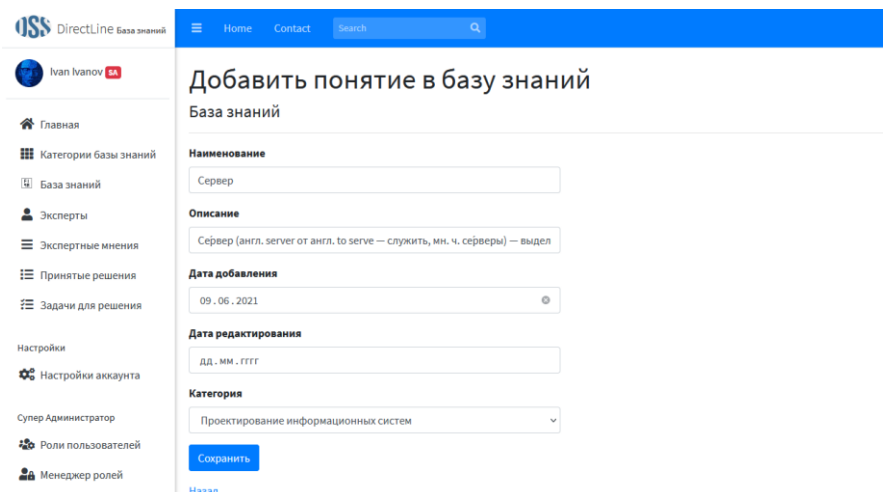


Рисунок 29 – Форма добавления понятия в базу знаний

В результате тестирования созданной базы знаний системы поддержки принятия решений определено, что все требуемые функции работоспособны и система готова к использованию в качестве модуля хранения и поиска информации предметной области.

Выводы по разделу 3. База знаний информационной системы поддержки принятия решений для ООО «Директ Лайн» реализована в виде веб-приложения в трехзвенной архитектуре с использованием языка программирования PHP и MySQL. Содержит база знаний информацию о предметной области сферы деятельности ООО «Директ Лайн» и включает вопросы, которые относятся к процессам разработки и продвижения веб-сайтов, а также других видов деятельности ООО «Директ Лайн». База знаний помогает осуществлять процесс принятия решений за счет сохранения и предоставлению пользователям информации о предметной сфере.

В результате тестирования созданной базы знаний системы поддержки принятия решений определено, что все требуемые функции работоспособны и система готова к использованию в качестве модуля хранения и поиска информации предметной области.

Заключение

В результате выполнения работы достигнута поставленная цель – разработана база знаний системы поддержки принятия решений для IT-компании ООО «Директ Лайн». При достижении цели были решены следующие задачи:

- проведен анализ предметной области и проанализирована деятельность ООО «Директ Лайн»;
- разработан проект базы знаний информационной системы поддержки принятия решений;
- реализована база знаний для информационной системы поддержки принятия решений ООО «Директ Лайн».

Сочетание современных информационных технологий позволяет проводить разработку базу знаний информационной системы (подсистемы) поддержки принятия решений, основной целью которой является своевременное и качественное обеспечение процессов поиска информации о предметной области в процессах принятия решений в условиях IT-компании ООО «Директ Лайн».

Системы поддержки принятия решений появились как естественное развитие и обобщение информационных систем управления и систем управления базами данных (СУБД) в направлении их наибольшей пригодности и приспособленности к задачам управленческой деятельности. Как правило, системы поддержки принятия решений представляют собой интерактивную автоматизированную систему, помогающую пользователю применять данные и модели для того, чтобы определить и решить проблемы, а также принять соответствующие решения. Также системы поддержки принятия решений возможно использовать, осуществляя выбор решений некоторых неструктурированных и частично структурированных проблем, включая несколько критериев. Данный процесс выполняется менеджерами организации.

База знаний представляет собой набор знаний, которые относятся к определенной предметной области и формально представлены так, чтобы, используя ее, была возможность выполнять рассуждения. Это особого рода база данных, содержащая информацию о человеческом опыте и знаниях в некоторой предметной области и созданная для управления этими знаниями, их сбора, хранения, поиска и выдачи. Используются базы знаний в процессах поддержки принятия решений.

Применение простых баз знаний, как правило, направлено на разработку экспертных систем и хранения данных о предприятии, в частности: сведения о документах, о составе руководства, о статьях техобеспечения. Основной целью разработки подобных баз служит помощь в процессах принятия решений.

База знаний представляет собой свод правил анализа данных от пользователя по какой-либо определенной проблеме. База знаний экспертной системы включает в себя факты (данные статистического характера о предметной области) и свод правил – набор инструкций, при использовании которых к известным фактам возможно получение новых фактов. Ключевой целью базы знаний выступает сокращение времени и трудозатрат на решение задач, которые входят в компетенцию определенных специалистов, преимущественно выполняющих руководящую роль.

База знаний информационной системы поддержки принятия решений для ООО «Директ Лайн» реализована в виде веб-приложения в трехзвенной архитектуре с использованием языка программирования PHP и MySQL. Содержит база знаний информацию о предметной области сферы деятельности ООО «Директ Лайн» и включает вопросы, которые относятся к процессам разработки и продвижения веб-сайтов, а также других видов деятельности ООО «Директ Лайн». База знаний помогает осуществлять процесс принятия решений за счет сохранения и предоставлению пользователям информации о предметной сфере и видах деятельности ООО «Директ Лайн».

Список используемой литературы

1. Агальцов В. П. Базы данных. В 2-х т. Т. 2. Распределенные и удаленные базы данных: Учебник / В. П. Агальцов. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 272 с.
2. Аксенов К. А. Разработка и применение системы поддержки принятия решений в управлении строительным холдингом / К. А. Аксенов, А. С. Антонова, О. П. Аксенова, А. А. Липодаева // Научно-технические ведомости СПбГПУ 4' 2011 Информатика. Телекоммуникации. Управление - С. 53–61.
3. Алиева Л. В. Моделирование научный метод познания современных реалий воспитания /Л.В. Алиева // Современные модели воспитания в условиях диверсификации образовательного пространства: Тезисы участников и программа Летней научной школ. – Тверь, 2005. – 38 с.
4. Андреев А. А. Педагогическая модель компьютерной сети /Андреев, А.В. Барабанщиков // Педагогическая информатика. – 2005.– №2. – 97 с.
5. Андреев А. А. Средства новых информационных технологий в образовании: систематизация и тенденции развития /А.А. Андреев. – М.: ВУ, 1995. – 86 с.
6. Аникин А. В. Подходы к реализации узкоспециализированных систем поддержки принятия решений с применением компьютерных языков программирования [Электронный ресурс] Режим доступа: http://elib.sfu-kras.ru/bitstream/handle/2311/6140/s2_31.pdf (Дата обращения: 15.01.2021).
7. Антошин М. К. Учимся работать на компьютере /М.К. Антошин – М.: Айрисс-пресс, 2008. – 114 с.
8. Ануфриева З. Д. Мотивационное управление педагогическим коллективом /З. Д. Ануфриева, Е.Ю. Кобаленова, Т.М. Трошкова // Управление дошкольным образовательным учреждением. – 2009. – №8. – 59 с.

9. Башмаков М. И. Информационная среда обучения /М.И.Башмаков, С. Н. Поздняков, Н. А. Резник. – СПб.: СВЕТ, 1997. – 400 с.
10. Белова С. К. Проектные основы бизнес-моделирования, классификации организации работы баров / С. К. Белова. - НАУКА – СЕРВИСУ. Материалы XXIII Международной научно-практической конференции. - 2018. – Т. 2. - С. 17–36.
11. Беспалько В. П. Образование и обучение с участием компьютеров (педагогика третьего тысячелетия) /В. П. Беспалько. – М.: НПО «Модэк», 2002. – 274 с.
12. Богомолова О. Б. Искусство презентации /О. Б. Богомолова. –М.: Педагогика, 2010. – 116 с.
13. Вольфсон Б. Гибкие методологии разработки. – М.: Эксмо, 2013. – 112 с.
14. Воронина Т. П. Философские проблемы образования в информационном обществе: автореф. дис. д-ра филос. Наук МГУ им. М. В. Ломоносова / Т. П. Воронина. – М.: Весна, 2005. – 20 с.
15. Гарнаев А. WEB-программирование на Java и JavaScript / Андрей Гарнаев , Сергей Гарнаев. - Москва: СПб. [и др.] : Питер, 2017. - 718 с.
16. Грин Д., Стеллман Э. Постигая Agile. Ценности, принципы, методологии. – Манн, Иванов и Фербер, 2018.– 448 с.
17. Гулякина Н. А. Методика проектирования семантической модели интеллектуальной справочной системы, основанная на семантических сетях / Н. А. Гулякина, И. Т. Давыденко, Д. В. Шункевич //
18. Давыдов В. В. Теория развивающего обучения /В.В. Давыдов. – М.: Рост, 1996. – 160 с.
19. Дамашке Г. PHP и MySQL; НТ Пресс - Москва, 2012. - 320 с. Дюбуа, Поль MySQL; М.: Вильямс; Издание 2-е - Москва, 2011. - 283 с. Кузнецов М., Симдянов И. MySQL на примерах; БХВ-Петербург - Москва, 2011. - 592 с.

20. Дахин А. Н. Педагогическое моделирование: сущность, эффективность и непосредственность /А.Н. Дахин // Педагогика. – 2003. – № 4. – 95 с.
21. Дорохин А.В. Интернет как инновационная технология социального управления /А.В. Дорохин // Первая международная конференция «Социология инноватики: теория и практика». – М.: РГИИС, 2006. – 86 с.
22. Дробышев Ю. А. Возможности использования новых информационных технологий при обучении младших школьников решению логических задач /Ю. А. Дробышев, С.Н. Ерлыченко // Информационные технологии в образовании. – М.: МИФИ, 2008. – 105 с.
23. Дронов В. А. PHP, MySQL, HTML5 и CSS3. Разработка современных динамических Web-сайтов / В. А. Дронов. – Спб.: BHV, 2016. – 688 с.
24. Дунаев, В. Сценарии для Web-сайта. PHP и JavaScript / В. Дунаев. - М.: БХВ-Петербург, 2017. - 576 с.
25. Дюбуа П. MySQL. Сборник рецептов / Дюбуа П. // Пер. с англ. – СПб: Символ–Плюс, 2006. – 1056 с.
26. Емельянов В. В. RAO-STUDIO для разработки имитационных моделей / В. В. Емельянов, А. В. Урусов, П. А. Захаров, А. В. Барс. Известия ТРТУ. Тематический выпуск «Интеллектуальные САПР». - С. 157
27. Емельянов В. В. RAO-STUDIO для разработки имитационных моделей / В. В. Емельянов, А. В. Урусов, П. А. Захаров, А. В. Барс. Известия ТРТУ. Тематический выпуск «Интеллектуальные САПР». - С. 157
28. Жадаев Александр PHP для начинающих; Питер - М., 2016. - 768 с.
29. Зайцева Е. Л. Формирование концепции построения имитационных моделей исполняемых бизнес-процессов// ИММОД-2009. Москва. Институт проблем управления РАН. -2009. - Т.1. С.135-139.
30. Информационные системы в экономике// Под ред. проф. В. В. Дика. - М.: Финансы и статистика, 2002.- 387 с.
31. Кеннет Рубин. Основы Scrum: Практическое руководство по гибкой разработке ПО. – М.: «Вильямс», 2016. – 544 с.

32. Колисниченко Денис PHP и MySQL. Разработка Web-приложений; БХВ-Петербург - М., 2017. - 560 с.

33. Колосов Д. М. Сравнительный анализ систем имитационного моделирования RDO и VPSim2 / Д. М. Колосов, К. А. Аксенов // Стендовые доклады. - ИММОД-2007. – С. 271–275.

34. Королева Н. Л., Абрицова М.В. Применение зарубежного опыта для использования информационных технологий в развитии детей 3-8 лет // Психолого-педагогический журнал Гаудеамус, №2 (22), 2013 URL: <https://cyberleninka.ru/article/n/primenenie-zarubezhnogo-opyta-dlya-ispolzovaniya-informatsionnyh-tehnologiy-v-razvitii-detey-3-8-let> (дата обращения: 19.03.2021).

35. Ларичев О. И. Системы поддержки принятия решений. Современное состояние и перспективы их развития [Электронный ресурс] / О. И. Ларичев, А. Б. Петровский // Итоги науки и техники. Сер. Техническая кибернетика. — Москва: ВИНТИ, 1987. — Т.21. — С. 131–164. — Режим доступа: http://www.raai.org/library/papers/Larichev/Larichev_Petrovsky_1987.pdf (дата обращения: 23.01.2021).

36. Ларичев О. И. Системы поддержки принятия решений. Современное состояние и перспективы их развития [Электронный ресурс] / О. И. Ларичев, А. Б. Петровский // Итоги науки и техники. Сер. Техническая кибернетика. — Москва: ВИНТИ, 1987. — Т.21. — С. 131–164. — Режим доступа: http://www.raai.org/library/papers/Larichev/Larichev_Petrovsky_1987.pdf (Дата обращения: 15.03.2021).

37. Маклафлин Б. PHP и MySQL. Исчерпывающее руководство / Б. Маклафлин. – СПб.: Питер, 2014. – 544 с.

38. Методология научных исследований: учеб. пособие / А. Б. Пономарев, Э. А. Пикулева. – Пермь: Изд-во Перм. нац. исслед. поли-техн. ун-та, 2014. – 186 с.

39. Методы и средства научных исследований: учеб. пособие / Ю. Н. Колмогоров [и др.]. — Екатеринбург: Изд-во Урал. ун-та, 2017. — 152 с.

40. Мустафаева Э. И. Использование СУБД MySQL для разработки информационно–справочной системы «Рекреационные ресурсы города Евпатории» [Электронный ресурс] / Э. И. Мустафаева, Ф. В. Шкарбан // Режим доступа: <http://eztuir.ztu.edu.ua/123456789/1037> (Дата обращения: 20.03.2021)
41. Прибыл Б. Oracle PL/SQL. Для профессионалов / Билл Прибыл. - М.: Питер, 2014. - 295 с.
42. Рон, Хардман Oracle Database PL/SQL. Рекомендации эксперта Хардман Рон. - М.: ЛОРИ, 2014. - 803 с.
43. Системы поддержки принятия решений [Электронный ресурс] URL:<http://xreferat.com/37/22-1-sistemy-podderzhki-i-prinyatiya-resheniiy.html> Дата обращения: 15.03.2021
44. СППР Выбор [Электронный ресурс] / Режим доступа: <http://www.ciritas.ru/product.php?id=10#39> (дата обращения: 09.02.2021).
45. Терелянский П. В. Системы поддержки принятия решений. Опыт проектирования: монография / П. В. Терелянский. — Волгоград: ВолгГТУ. 2009. — 127 с.
46. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 с.
47. Хеффельфингер, Дэвид Разработка приложений Java EE 6 в NetBeans 7 / Дэвид Хеффельфингер. - М.: ДМК Пресс, 2013. - 213 с.
48. Шабанов Р. М., Микушин Н. А. Интеллектуальная информационная система поддержки принятия решений. Молодой исследователь Дона №4(19) 2019
49. Шкрыль, А. PHP — это просто. Програмируем для Web-сайта / А. Шкрыль. - М.: БХВ-Петербург, 2016. - 368 с.
50. Aksyonov K., Vykov E., Sysoletin E., Aksyonova O., Goncharova N. Perspectives of Modeling in Metallurgical Production // International Conference on Social Science, Management and Economics (SSME 2015) Guangzhou, China, May 09-10, 2015, P. 876-880.

51. Aksyonov K.A., Bykov E.A., Smoliy E.F., Aksyonova O.P., Wang Kai
Planning and bottleneck analysis of construction enterprise project portfolio // 7th
IFAC Conference on Manufacturing Modelling, Management, and Control, MIM
2013; Saint Petersburg; Russian Federation; 19-21 June 2013. pp. 659-663.

52. Collier N. Repast for python scripting //Proceedings of the Agent 2004
Conference on Social Dynamics: Interaction, Reflexivity and Emergence, Chicago,
IL. – 2004. – P. 231-237.

53. Power D. J. «What is a DSS?» // The On-Line Executive Journal for Data-
Intensive Decision Support, 1997. — v. 1. — N3.

54. Scott Morton M. S. Management Decision Systems: Computer-based
Support for Decision Making. — Boston: Harvard University, 1971.

Приложения А

Создание таблиц

```
CREATE TABLE IF NOT EXISTS `directline`.`article_categories` (  
  `id` INT NOT NULL AUTO_INCREMENT,  
  `name_category` VARCHAR(60) NULL,  
  `descripton_category` VARCHAR(250) NULL,  
  `article_categories` VARCHAR(45) NULL,  
  PRIMARY KEY (`id`))  
ENGINE = InnoDB;  
CREATE TABLE IF NOT EXISTS `directline`.`expert` (  
  `fullname` INT NOT NULL AUTO_INCREMENT,  
  `competension` VARCHAR(145) NULL,  
  `contactdate` VARCHAR(45) NULL,  
  PRIMARY KEY (`fullname`))  
ENGINE = InnoDB;  
CREATE TABLE IF NOT EXISTS `directline`.`dictcategory` (  
  `idcategory` INT NOT NULL AUTO_INCREMENT,  
  `name` VARCHAR(50) NULL,  
  `description` VARCHAR(45) NULL,  
  `dateadd` DATETIME NULL,  
  PRIMARY KEY (`idcategory`))  
ENGINE = InnoDB;  
CREATE TABLE IF NOT EXISTS `directline`.`dict` (  
  `iddict` INT NOT NULL AUTO_INCREMENT,  
  `name` INT NULL,  
  `content` TEXT NULL,  
  `dateadd` DATETIME NULL,  
  `dateedit` DATETIME NULL,  
  `dictcategory_idcategory` INT NOT NULL,  
  PRIMARY KEY (`iddict`),  
  INDEX `fk_dict_dictcategory1_idx` (`dictcategory_idcategory` ASC) VISIBLE,  
  CONSTRAINT `fk_dict_dictcategory1`  
    FOREIGN KEY (`dictcategory_idcategory`)  
    REFERENCES `directline`.`dictcategory` (`idcategory`)  
    ON DELETE NO ACTION
```



```

        ON UPDATE NO ACTION)
ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `directline`.`expertopinion` (
  `idexpertopinion` INT NOT NULL AUTO_INCREMENT,
  `content` TEXT NULL,
  `dateadd` DATETIME NULL,
  `expert_fullname` INT NOT NULL,
  `dict_iddict` INT NOT NULL,
  PRIMARY KEY (`idexpertopinion`),
  INDEX `fk_expertopinion_expert1_idx` (`expert_fullname` ASC) VISIBLE,
  INDEX `fk_expertopinion_dict1_idx` (`dict_iddict` ASC) VISIBLE,
  CONSTRAINT `fk_expertopinion_expert1`
    FOREIGN KEY (`expert_fullname`)
    REFERENCES `directline`.`expert` (`fullname`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `directline`.`article` (
  `idarticle` INT NOT NULL AUTO_INCREMENT,
  `articlename` INT NULL,
  `content` TEXT NULL,
  `dateadd` DATETIME NULL,
  `deteedit` DATETIME NULL,
  `atricle_categories_id` INT NOT NULL,
  `expertopinion_idexpertopinion` INT NOT NULL,
  PRIMARY KEY (`idarticle`),
  INDEX `fk_article_atricle_categories1_idx` (`atricle_categories_id` ASC) VISIBLE,
  INDEX `fk_article_expertopinion1_idx` (`expertopinion_idexpertopinion` ASC)
VISIBLE,
ENGINE = InnoDB;

```

Приложения Б

Основной модуль программы

```
<?php

namespace app\models;

use Yii;

/**
 * This is the model class for table "article".
 *
 * @property int $idarticle
 * @property int|null $articlename
 * @property string|null $content
 * @property string|null $dateadd
 * @property string|null $deteedit
 * @property int $atricle_categories_id
 * @property int $expertopinion_idexpertopinion
 *
 * @property ArticleCategory $atricleCategories
 * @property Expertopinion $expertopinionIdexpertopinion
 * @property Savedresult[] $savedresults
 */
class Article extends \yii\db\ActiveRecord
{
    /**
     * {@inheritdoc}
     */
    public static function tableName()
    {
        return 'article';
    }

    /**
     * {@inheritdoc}
     */
}
```

```

*/
public function rules()
{
    return [
        [['articlename', 'atricle_categories_id', 'expertopinion_idexpertopinion'], 'integer'],
        [['content'], 'string'],
        [['dateadd', 'deteedit'], 'safe'],
        [['atricle_categories_id', 'expertopinion_idexpertopinion'], 'required'],
        [['atricle_categories_id'], 'exist', 'skipOnError' => true, 'targetClass' =>
AtricleCategories::className(), 'targetAttribute' => ['atricle_categories_id' => 'id']],
        [['expertopinion_idexpertopinion'], 'exist', 'skipOnError' => true, 'targetClass' =>
Expertopinion::className(), 'targetAttribute' => ['expertopinion_idexpertopinion' =>
'idexpertopinion']],
    ];
}

/**
 * {@inheritdoc}
 */
public function attributeLabels()
{
    return [
        'idarticle' => Yii::t('app', 'Idarticle'),
        'articlename' => Yii::t('app', 'Articlename'),
        'content' => Yii::t('app', 'Content'),
        'dateadd' => Yii::t('app', 'Dateadd'),
        'deteedit' => Yii::t('app', 'Deteedit'),
        'atricle_categories_id' => Yii::t('app', 'Atricle Categories ID'),
        'expertopinion_idexpertopinion' => Yii::t('app', 'Expertopinion Idexpertopinion'),
    ];
}

/**
 * Gets query for [[AtricleCategories]].
 *

```

```

        * @return \yii\db\ActiveQuery
        */
        public function getAtricleCategories()
        {
            return $this->hasOne(AtricleCategories::className(), ['id' =>
'atricle_categories_id']);
        }

        /**
         * Gets query for [[ExpertopinionIdexpertopinion]].
         *
         * @return \yii\db\ActiveQuery
         */
        public function getExpertopinionIdexpertopinion()
        {
            return $this->hasOne(Expertopinion::className(), ['idexpertopinion' =>
'expertopinion_idexpertopinion']);
        }

        /**
         * Gets query for [[Savedresults]].
         *
         * @return \yii\db\ActiveQuery
         */
        public function getSavedresults()
        {
            return $this->hasMany(Savedresult::className(), ['article_idarticle' => 'idarticle']);
        }
    }
}
<?php

namespace app\models;

use Yii;

```

```

/**
 * This is the model class for table "atricle_categories".
 *
 * @property int $id
 * @property string|null $name_category
 * @property string|null $descripton_category
 * @property string|null $atricle_categories
 *
 * @property Article[] $articles
 */
class ArticleCategory extends \yii\db\ActiveRecord
{
    /**
     * {@inheritdoc}
     */
    public static function tableName()
    {
        return 'atricle_categories';
    }

    /**
     * {@inheritdoc}
     */
    public function rules()
    {
        return [
            [['name_category'], 'string', 'max' => 60],
            [['descripton_category'], 'string', 'max' => 250],
            [['atricle_categories'], 'string', 'max' => 45],
        ];
    }

    /**
     * {@inheritdoc}
     */

```

```

public function attributeLabels()
{
    return [
        'id' => Yii::t('app', 'ID'),
        'name_category' => Yii::t('app', 'Name Category'),
        'descripton_category' => Yii::t('app', 'Descripton Category'),
        'atricle_categories' => Yii::t('app', 'Atricle Categories'),
    ];
}

/**
 * Gets query for [[Articles]].
 *
 * @return \yii\db\ActiveQuery
 */
public function getArticles()
{
    return $this->hasMany(Article::className(), ['atricle_categories_id' => 'id']);
}
}
<?php

```

```

namespace app\models;

```

```

use Yii;

```

```

use yii\db\ActiveRecord;

```

```

/**
 * This is the model class for table "dict".
 *
 * @property int $iddict
 * @property int|null $name
 * @property string|null $content
 * @property string|null $dateadd
 * @property string|null $dateedit

```

```

* @property int $dictcategory_idcategory
*
* @property Dictcategory $dictcategoryIdcategory
* @property Expertopinion[] $expertopinions
* @property Savedresult[] $savedresults
*/
class Dict extends ActiveRecord
{
/**
 * {@inheritdoc}
 */
public static function tableName()
{
    return 'dict';
}

/**
 * {@inheritdoc}
 */
public function rules()
{
    return [
        [['name', 'dictcategory_idcategory'], 'integer'],
        [['content'], 'string'],
        [['dateadd', 'dateedit'], 'safe'],
        [['dictcategory_idcategory'], 'required'],
        [['dictcategory_idcategory'], 'exist', 'skipOnError' => true, 'targetClass' =>
Dictcategory::className(), 'targetAttribute' => ['dictcategory_idcategory' => 'idcategory']],
    ];
}

/**
 * {@inheritdoc}
 */
public function attributeLabels()

```

```

{
    return [
        'iddict' => Yii::t('app', 'Iddict'),
        'name' => Yii::t('app', 'Name'),
        'content' => Yii::t('app', 'Content'),
        'dateadd' => Yii::t('app', 'Dateadd'),
        'dateedit' => Yii::t('app', 'Dateedit'),
        'dictcategory_idcategory' => Yii::t('app', 'Dictcategory Idcategory'),
    ];
}

/**
 * Gets query for [[DictcategoryIdcategory]].
 *
 * @return \yii\db\ActiveQuery
 */
public function getDictcategoryIdcategory()
{
    return $this->hasOne(Dictcategory::className(), ['idcategory' =>
'dictcategory_idcategory']);
}

/**
 * Gets query for [[Expertopinions]].
 *
 * @return \yii\db\ActiveQuery
 */
public function getExpertopinions()
{
    return $this->hasMany(Expertopinion::className(), ['dict_iddict' => 'iddict']);
}

/**
 * Gets query for [[Savedresults]].
 *

```



```
* @return \yii\db\ActiveQuery
*/
public function getSavedresults()
{
    return $this->hasMany(Savedresult::className(), ['dict_iddict' => 'iddict']);
}
}
<?php
```

```
use yii\db\Connection;
```

```
return [
    'class' => Connection::class,
    'dsn' => 'mysql:host=localhost;dbname=directline',
    'username' => 'root',
    'password' => 'root',
    'charset' => 'utf8',
```

```
;
```