МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ 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 кафедры)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Прикладная информатика в социальной сфере
(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

На тему Разработка системы управления подписками на основе REST-сервисов

Студент А.А. Романов _____
(И.О. Фамилия) (личная подпись)

Руководитель О.В. Аникина _____
(И.О. Фамилия) (личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____
(ученая степень, звание,И.О.Фамилия) (личнаяподпись)

«_____»_____20_____г.

Тольятти 2018

АННОТАЦИЯ

Темой данной выпускной квалификационной работы является "Разработка системы управления подписками на основе REST-сервисов".

Работа выполнена студентом Тольяттинского Государственного Университета, института математики, физики и информационных технологий, группы ПИБЗ-1331 Романовым Александром Александровичем.

Актуальность темы "Разработка системы управления подписками на основе REST-сервисов" обуславливается необходимостью разработки и внедрения системы, связывающей между собой все информационные ресурсы компании, которая позволит осуществлять поиск и управление подписками из единого пользовательского интерфейса.

Объект исследования: сторонние информационные системы, используемые сотрудниками компании, система менеджмента задач, а также, система для работы с документацией продукта и проектов.

Предмет исследования: возможность осуществления взаимосвязи сторонних систем для создания единого интерфейса поиска и просмотра информации, хранящейся в них, а также, подписки на уведомления о событиях, касающихся определенного рода информации.

Во введении описывается актуальность проводимого исследования, выделяются проблемы исследования, формируется цель и ставятся задачи.

В первой главе проведена технико-экономическая характеристика предметной области, смоделированы бизнес-процессы для постановки задач нового варианта, разработаны и проанализированы бизнес-модели «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

Во второй главе описаны функциональные и технические архитектуры информационной системы, спроектированы концептуальная и логическая модели баз данных, спроектирован дизайн-макет web-интерфейса пользователя.

В третьей главе построена физическая модель баз данных, спроектирован web-интерфейс пользователя, разработано руководство пользователя, проведено тестирование.

В заключении приводятся основные выводы, достигнутые в ходе выполнения работы.

В результате была описана, разработана и протестирована система управлениями подписками на интересующие пользователей публикации на основе REST-сервисов.

Выпускная квалификационная работа содержит пояснительную записку объемом 55 страниц, включая 30 иллюстраций, 6 таблиц, список литературы из 20 наименований, приложение.

ANNOTATION

The theme of graduation work is "Development of a subscription management system based on REST-services".

The relevance of the topic "Development of a subscription management system based on REST-services" is conditioned by the need to develop and implement a system linking all information resources of the company, which will allow searching and managing subscriptions from a single user interface.

Object of the graduation work: third-party information systems used by company employees, task management system, and also a system for working with product documentation and projects.

The subject of the graduation work: the possibility of implementing the interconnection of third-party systems to create a single interface for searching and viewing information stored in them, as well as subscribing to notifications of events relating to certain types of information.

The introduction describes the relevance of the research, highlight research problems, the goal is formed and tasks are set.

In the first chapter, the technical and economic characteristics of the subject area were carried out, business processes were modeled for setting the tasks of the new option, business models "AS IS" and "HOW TO BE" were developed and analyzed.

The second chapter describes the functional and technical architectures of the information system, the conceptual and logical models of databases have been designed, and the design layout of the web user interface has been designed.

In the third chapter, a physical database model is constructed, a web-based user interface is designed, a user's guide is developed, and testing is conducted.

In conclusion, the main conclusions reached in the course of the work are given.

As a result, the system has described, developed and tested the system by managing subscriptions to publications based on REST services.

The graduation work contains an explanatory note in the volume of 55 pages, including 30 illustrations, 6 tables, a list of literature from 20 titles, appendix.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В большинстве компаний в информационной сфере используется множество программных продуктов, информационных ресурсов и других систем, которые рассчитаны на решение различного рода задач. Однако, чем больше таких ресурсов, тем сложнее становиться процесс получения из них необходимой конкретной информации. **Актуальность** работы обусловлена необходимостью взаимосвязи систем, использующихся в компании, посредством внедрения новой информационной системы (ИС) для предоставления актуальной информации сразу из нескольких ресурсов компании, а также предоставления возможности подписки на получение уведомлений о интересующих в них событиях.

В компании Netcracker используется несколько информационных систем, которые доступны всем сотрудникам компании: система менеджмента задач и ошибок проектов, различные информационные системы и система учета документации. Информация в этих системах имеет множество общих тем и разделов, но на данный момент для ее поиска и осуществления подписки на интересующую тему необходимо просматривать эти системы раздельно. Внедрение ИС, способной группировать информацию с возможностью оповещения о новых или измененных элементов систем (статьях, документах, задачах) с единого места значительно сократит время на поиск нужной информации и добавит новый удобный интерфейс для ее просмотра.

Объект исследования: сторонние информационные системы, используемые сотрудниками компании, система менеджмента задач, а также, система для работы с документацией продукта и проектов.

Предмет исследования: возможность осуществления взаимосвязи сторонних систем для создания единого интерфейса поиска и просмотра информации, хранящейся в них, а также, подписки на уведомления о событиях, касающихся определенного рода информации.

Данная работа выполнялась по заказу организации, и **целью** работы является проектирование и разработка системы управления подписками на основе REST-сервисов.

Для достижения поставленной цели необходимо решить следующие **задачи**:

- описать существующую архитектуру, в которую будет внедрена система;
- разработать модель процесса управления подписками;
- разработать проект решения задачи, привести сравнение аналогов архитектуры, обосновать выбор архитектуры;
- разработать эскизный проект, построить концептуальную и логическую модели данных.

В первой главе проведена технико-экономическая характеристика предметной области, смоделированы бизнес-процессы для постановки задач нового варианта, разработаны и проанализированы бизнес-модели «КАК ЕСТЬ» и «КАК ДОЛЖНО БЫТЬ».

Во второй главе описаны функциональные и технические архитектуры информационной системы, спроектированы концептуальная и логическая модели баз данных, спроектирован дизайн-макет web-интерфейса пользователя.

В третьей главе построена физическая модель баз данных, спроектирован web-интерфейс пользователя, показан процесс разработки системы с описанием используемых технологий, разработано руководство пользователя, проведено тестирование.

В заключении приводятся основные выводы, достигнутые в ходе выполнения работы.

ГЛАВА 1 ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Технико-экономическая характеристика деятельности ООО «Неткрэкер»

Составление технико-экономической характеристики является важным этапом, поскольку дает представление о предметной области, компании, ее организационной структуре и сложившейся технической архитектуре. Ниже приведено краткое описание компании, для которой будет проектироваться информационная система для управления подписками на получение уведомлений об интересующей информации.

Наименование организации: ООО «Неткрэкер».

Юридический адрес: г. Тольятти, ул. Фрунзе 14, Б, (ДЦ "Квадрат"), офис 313.

Netcracker Technology — это мировой лидер в разработке и внедрении программного обеспечения для операторов связи. На сегодняшний день компания имеет более 20 лет отраслевого опыта, 8000 специалистов, тысячи успешных внедрений и высочайший уровень стандартизации в индустрии. Успехи компании неоднократно отмечены самыми престижными отраслевыми наградами и признаны ведущими аналитиками телеком-отрасли, включая Gartner, Analysys Mason и Stratecast. В сотрудничестве со своей материнской компанией — японской корпорацией NEC — Netcracker уже несколько лет занимает ведущие позиции на рынке инновационных технологий SDN/NFV. Центры разработки находятся в России, Украине, Беларуси, Великобритании, Индии, США и Израиле.

Компания занимается разработкой, внедрением и поддержкой собственного продукта «Netcracker» – системы, предоставляющей OSS/BSS решений для различных компаний-провайдеров в сфере телекоммуникаций.

Компания Netcracker имеет сложную структуру, в которую включены множество департаментов:

- Отдел тестирования;

- Отдел разработки;
- Отдел бизнес-аналитиков;
- Отдел кадров;
- Администрация;
- Отдел продаж.

Рисунок 1.1 – Организационная схема компании ООО «НетКрэкер»

В свою очередь все эти элементы так же подразделяются на более мелкие объекты: практики, отделы, группы.

В состав ИТ - инфраструктуры службы входит следующее оборудование: двухранговая вычислительная сеть, состоящая из серверов телематических служб, серверов баз данных, коммутационного оборудования: коммутаторов второго и третьего уровня, маршрутизаторов и персональных компьютеров с операционными системами Windows 7 или Windows 10.

1.2 Концептуальное моделирование системы управления подписками

1.2.1 Выбор технологии концептуального моделирования системы управления подписками

Моделирование происходящих в организации бизнес-процессов, реализующих ее цели и задачи, является одним из основных этапов разработки информационной системы. Существует несколько методик процесса построения бизнес-модели, которые можно разделить на два вида: объектно-ориентированные и структурно-функциональные.

Для объектно-ориентированной методики характерно, что моделируемая организация рассматривается как набор объектов, взаимодействующих между собой. Явление, предмет или некая реальная сущность в целом, характеризующийся набором атрибутов и имеющий определенное поведение является в данном случае объектом. Такая методика применяется для выделения конкретных объектов и распределения между ними ответственностей за выполняемые действия.

Для структурно-функциональной методики характерно, что моделируемая организация рассматривается как набор функций, направленных на преобразование входного потока информации в выходной поток. Данная методика отличается от объектной в основном четким отделениям функций (методов обработки данных) от самих данных.

Для проектирования системы был выбран структурный анализ, т.к. он имеет несколько преимуществ, по сравнению с объектно-ориентированным подходом. Основным достоинством является реализация подхода к проектированию системы по иерархическому принципу, когда каждый функциональный блок может быть декомпозирован на множество связанных подфункций. Также, в структурно-функциональной методике соблюдается строгость декомпозиции системы и наглядность представления.

Различные CASE-средства помогают облегчить процесс анализа и проектирования ИС, а также, подготовить проектную документацию, обеспечивают качество принимаемых технических решений. CASE-технологии представляют собой совокупность методологий проектирования и сопровождения ИС на всем жизненном цикле. Среди средств моделирования предметной области можно выделить графические средства, они наглядно позволяют разработчикам изучать ИС, перестраивать ее в зависимости от имеющихся ограничений и поставленных целей.

Все CASE-средства классифицируют по нескольким типам:

− средства анализа для построения моделей предметной области;

− средства анализа и проектирования для создания проектных спецификаций;

− средства проектирования баз данных, обеспечивающие моделирование данных и генерацию схем баз данных;

− средства разработки приложений;

− средства реинжиниринга.

Для построения моделей бизнес-процесса в данной работе используется средство BPWin 4.0, у которого имеется пробная бесплатная версия.

1.2.2 Моделирование бизнес-процесса подписки на рассылку информации от сторонних систем для постановки задачи нового варианта решения

Как уже было отмечено ранее, в компании используются различные системы хранения информации, которые распределены по разным тематикам и назначению. Эти системы перечислены ниже и рассматриваются более подробно каждая в отдельности.

Первая из них это информационная система с удобным Web-интерфейсом, предоставляющая информационные пространства (разделы) для продукта и проектов компании, в котором информация храниться в виде статей. В них может быть приведено описание продукта, проектов и их компонент, опыт внедрения различных решений известных проектных задач список часто задаваемых вопросов и многое другое.

Также есть система отслеживания задач и ошибок, которая предоставляет для продукта и каждого проекта возможность управления процессом разработки, хранящая информацию в виде определенных сущностей – задач. Ответственные за разработку проекта люди могут создавать списки задач и назначать эти задачи на участников команды разработки проекта, а также отслеживать их статус выполнения, тестирования на всех этапах разработки.

Третья рассматриваемая система предназначена для работы с документацией продукта и проектов. Ее элементами являются конечные документы, а также находящиеся в стадии разработки.

Фактически, любой сотрудник компании имеет доступ к этим ресурсам и публикуемым на них материалам в целях получения новостей компании, информации по интересующим темам, обмена знаниями и опытом. На каждом из ресурсов информация структурирована по категориям и\или проектам, пользователь может подписаться на получение уведомлений при добавлении или обновлении материалов в интересующих его категориях и\или проектах. Уведомления приходят на электронную почту, и пользователь имеет возможность в любое время отписаться от любых уведомлений.

1.2.3 Разработка и анализ модели бизнес-процесса подписки на рассылку информации от сторонних систем «КАК ЕСТЬ»

Функциональная модель «как есть» является объектом и отчасти инструментом анализа и позволяет выявить причины нерационального функционирования исследуемого бизнес-процесса. Сверху диаграммы показаны стрелками управляющие воздействия, слева отображены входные данные, справа отображены выходные данные, снизу показаны исполнители.

Исходя из описанного ранее предмета исследования, можно выделить основные компоненты бизнес-процесса осуществления подписки на получение уведомлений от ИС компании:

- запрос на подписку;
- уведомление об успешной подписке;
- пользователи (сотрудники компании);
- информационные системы.

На рисунках 1.2, 1.3, 1.4 представлена функциональная модель бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании.

Рисунок 1.2 – Контекстная IDEF0-диаграмма бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании КАК ЕСТЬ (0-й уровень)

Входными данными данной диаграммы является запрос на оформление подписки. Выходными данными представлены уведомление о том, что подписка на интересующую тему на прошла успешно. Правила пользования ресурсом, руководство пользователя «Корпоративная сеть и Интернет» и политика безопасности компании являются управляющими воздействиями. Исполнителем на данной диаграмме является пользователь – сотрудник компании.

Как видно из декомпозиции бизнес-процесса, показанной на рисунке 1.3, процесс подписки на получение уведомлений от информационных ресурсов компании подразумевает собой что, подписку на интересующую информацию во всех рассматриваемых в данной работе информационных ресурсах нужно проходить в каждой по отдельности.

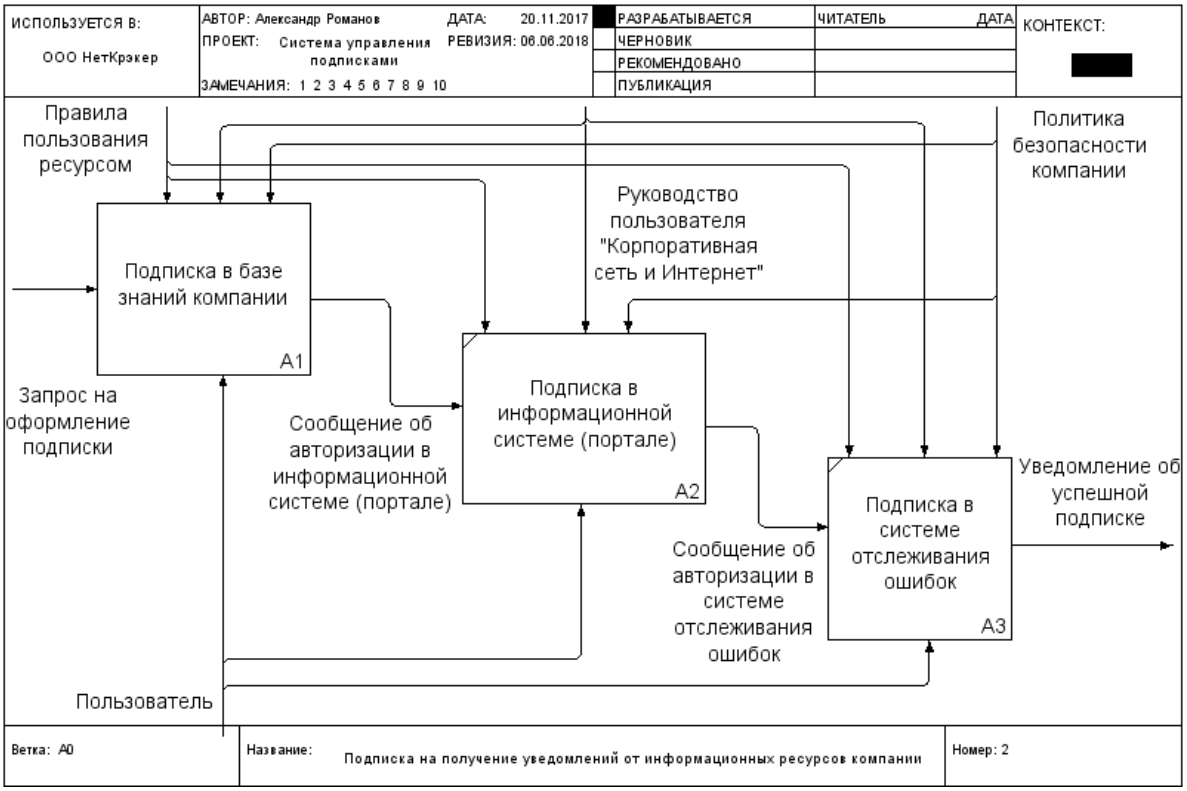

Рисунок 1.3 – IDEF0-декомпозиция бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании КАК ЕСТЬ (1-й уровень)

Входными данными данной диаграммы является запрос на оформление подписки. Выходными данными представлены уведомление о том, что подписка на интересующую тему на прошла успешно. Правила пользования ресурсом, руководство пользователя «Корпоративная сеть и Интернет» и политика безопасности компании являются управляющими воздействиями. Исполнителем на данной диаграмме является пользователь – сотрудник компании.

Более детально процесс подписки на рассылку уведомлений об интересующей информации показан ниже на рисунке 1.4.

По сути, для каждой из систем нужно выполнить ряд схожих действий, для достижения конечного результата. Для начала нужно пройти авторизацию в конкретной системе, затем осуществить поиск нужной информации, и затем уже необходимо послать запрос на подписку на рассылку уведомлений о новой или измененной информации.

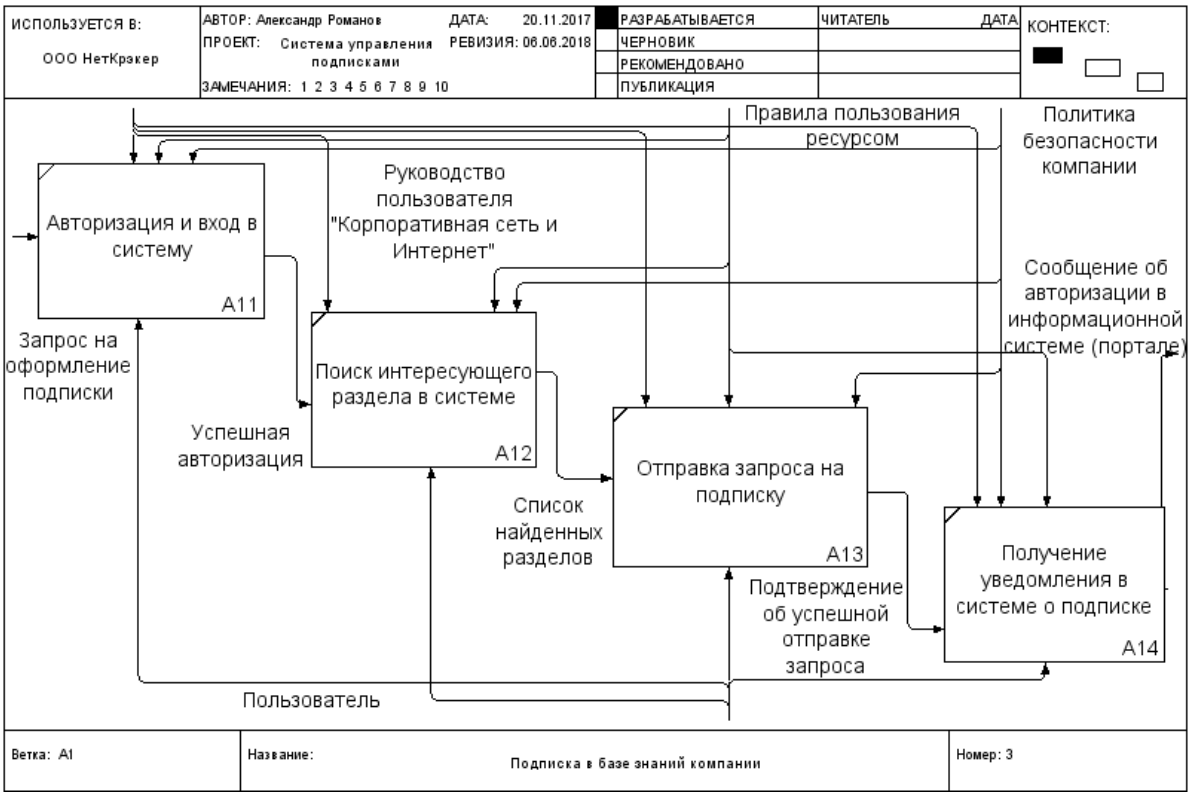

Рисунок 1.4 – IDEF0-декомпозиция бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании КАК ЕСТЬ (2-й уровень)

Входными данными данной диаграммы является запрос на оформление подписки. Выходными данными представлены уведомление о том, что подписка на интересующую тему на прошла успешно. Правила пользования ресурсом, руководство пользователя «Корпоративная сеть и Интернет» и политика безопасности компании являются управляющими воздействиями. Исполнителем на данной диаграмме является пользователь – сотрудник компании.

1.2.4 Обоснование необходимости нового варианта решения и формирование требований к новой технологии

Исходя из построенной модели «КАК ЕСТЬ» можно отметить, что существующая модель имеет ряд недостатков.

Одним из важных недостатков является трудоемкость поиска интересующей информации, ввиду того, что для осуществления такового

необходимо производить его во всех трех системах. Так, если в одном проекте используется готовое решение, предоставленное другим проектом (например, используется разработанная в компании библиотека компонентов для проектов, использующих фреймворк Angular), то при поиске в рассматриваемых системах компании можно обнаружить и документацию по этой библиотеке, и отдельный раздел в ИС, хранящей описание в более простой форме, дальнейшие планы разработки и список часто задаваемых вопросов. Также в системе менеджмента задач можно найти много информации, касающейся обнаружению и исправлению ошибок в компонентах библиотеки.

Для осуществления подписки на нужную информацию нужно также просмотреть все три рассматриваемые системы в отдельности, и в каждой отправлять соответствующие запросы на подписку уведомлений о ней.

После успешной подписки на нужную информацию, рассматриваемые системы будут отправлять уведомления на электронную почту не согласованно – каждая отдельным письмом и в разное время, что также является не очень удобным, пользователь может пропустить какое-либо из уведомлений из большого потока ему электронных писем в компании.

Из вышеперечисленного можно выделить следующий список недостатков:

- отсутствие единой точки сбора информации по интересующим темам;
- пользователю нужно посещать все информационные ресурсы в поисках обновлений;
- пользователю нужно оформлять подписки на всех ресурсах в отдельности;
- уведомления приходят на почту разными письмами в разное время, что делает вероятным упущение из виду какого-либо уведомления в большом потоке писем пользователю;
- нет единого места управления подписками.

Таким образом, для устранения вышеперечисленных недостатков нужно разработать и внедрить новую информационную систему, которая смогла бы обеспечить быстрый поиск нужной информации и удобный интерфейс для управления подписками. Следующим этапом будет постановка задачи на разработку проекта создания ИС.

1.3 Постановка задачи на разработку проекта создания системы управления подписками

Цель разработки решения, как правило, должна сводиться к устранению выявленных в ходе анализа недостатков. Рассматривая отмеченные выше недостатки, можно выделить следующие цели:

- сокращение времени на поиск нужной информации;
- сокращение времени на процесс подписки на уведомления об интересующей информации;
- предоставление возможности управления подписками.

Назначением решения задачи являются:

- автоматизация сбора новой информации в системах;
- автоматизация формирования и отправки уведомлений на электронную почту пользователям;
- формирование списка существующих у пользователя подписок.

Источниками информации являются три рассмотренные выше информационные системы, информация в которых храниться в разном виде, в зависимости от того, что это за система: это могут быть статьи, задачи и документы. В дальнейшем можно объеденить эти сущности под одним названием – публикации. С учетом вышеизложенного, поставлена задача разработки системы управления подписками на информационных ресурсах компании. Для достижения этого необходимо произвести разработку проекта решения задачи, в котором нужно:

- смоделировать бизнес-процесс КАК ДОЛЖНО БЫТЬ;
- сформировать требования к ИС;
- создать вариант технического задания на разработку ИС;

- выбрать архитектуру ИС;
- выбрать средства разработки ИС;
- определить состав и содержание работ.

В разрабатываемой системе должны быть такие элементы, как: список новых публикаций, список публикаций, на которые пользователь имеет подписку на получение уведомлений, форма поиска, настройки получения уведомлений, меню для переключения между элементами системы.

После проведения поиска информации по нужным критериям, система должна выдать список подходящих публикаций, каждая из которых должна содержать необходимую основную информацию: дату публикации или ее последнего изменения, название, автора, краткую информацию, ссылку на основной источник, а также список разделов (тегов), к которым эта информация относиться.

Рассматриваемые ИС, с которыми нужно будет осуществить интеграцию разрабатываемой системы – это известные продукты линейки Atlassian, которые имеют свой набор инструментов Atlassian SDK для написания плагинов к основным продуктам Atlassian. Для разработки плагинов используются такие технологии, как: Java как язык программирования для написания приложения, Maven как средство их сборки, OSGi как контейнер для приложений, AX-RS для REST и другие. На странице администрирования системы есть специальная форма, куда можно загрузить jar-файл, который затем добавляется в OSGi-контейнер, происходит процесс развертывания плагина, и он начинает работать.

Формирование запросов к системе будет осуществляться в соответствии с технологией REST.

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, который, как правило, используется для построения веб-служб. В общем случае REST является очень простым интерфейсом управления информацией, где каждая единица информации однозначно определяется глобальным идентификатором, таким

как URL. Каждая URL в свою очередь имеет строго заданный формат и по сути является первичным ключом для единицы данных. Архитектура REST очень проста в плане использования и главное достоинство состоит в том, что с ней работать может какая угодно система, что необходимо для обеспечения взаимосвязи нескольких систем.

1.4 Разработка модели бизнес-процесса «КАК ДОЛЖНО БЫТЬ»

Функциональная модель «как должно быть» является основой технического задания на создание или модернизацию системы и представляет собой концептуальную модель оптимизированного бизнес-процесса. Разработанная модель бизнес-процесса КАК ДОЛЖНО БЫТЬ показана на рисунке 1.5.

Рисунок 1.5 – Контекстная IDEF0-диаграмма бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании КАК ДОЛЖНО БЫТЬ (0-й уровень)

Входными данными данной диаграммы является запрос на оформление подписки. Выходными данными представлены уведомление о том, что подписка на интересующую тему на прошла успешно. Правила пользования ресурсом, руководство пользователя «Корпоративная сеть и Интернет» и политика безопасности компании являются управляющими воздействиями. Исполнителем на данной диаграмме является пользователь – сотрудник компании.

На рисунке 1.6 представлена декомпозиция бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании.

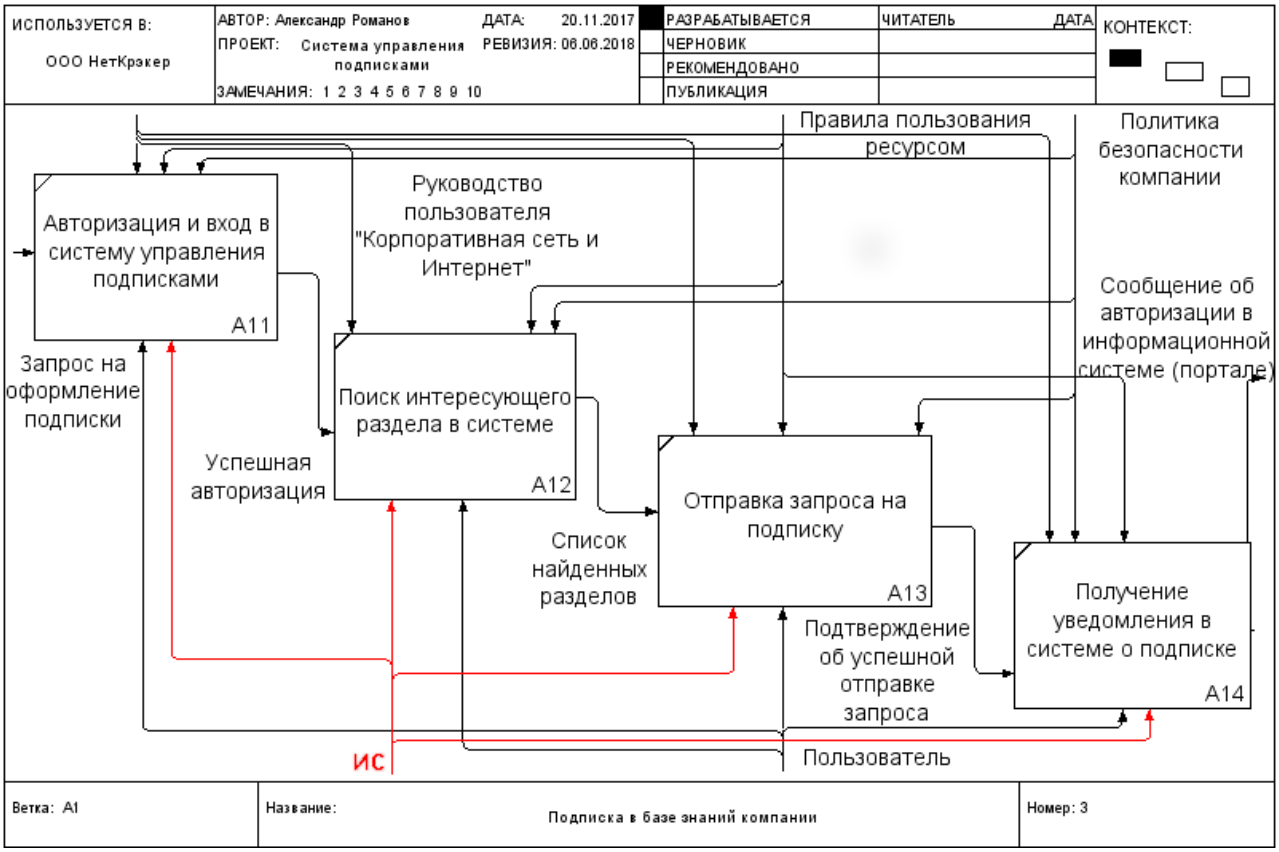

Рисунок 1.6 – IDEF0-декомпозиция бизнес-процесса подписки на получение уведомлений от информационных ресурсов компании КАК ДОЛЖНО БЫТЬ (1-й уровень)

Входными данными данной диаграммы является запрос на оформление подписки. Выходными данными представлены уведомление о том, что

подписка на интересующую тему на прошла успешно. Правила пользования ресурсом, руководство пользователя «Корпоративная сеть и Интернет» и политика безопасности компании являются управляющими воздействиями. Исполнителем на данной диаграмме является пользователь – сотрудник компании.

В новом решении введен автоматический сбор информации (изменениях, добавлениях статей, событий) и публикации их в новой системе, введен Web-интерфейс для просмотра всей доступной интересующей информации в удобном виде, что позволит отслеживать все обновления на интересующие темы, введена возможность управления имеющимися подписками. Таким образом, усовершенствование исследуемого бизнес-процесса достигается путем разработки и внедрения системы управления подписками, отвечающей требованиям заказчика.

1.4.1 Формирование требований к системе управления подписками

1.4.1.1 Выбор архитектуры системы

Как правило, компоненты любой информационной системы, взаимодействующие между собой, не являются равноправными. Одни из компонентов могут иметь доступ к ресурсам других. Компонент, владеющий каким-либо ресурсом, называется сервером. Компонент, имеющий доступ к этому ресурсу – клиентом. В настоящее время клиент-серверная архитектура является распространенным способом организации сетевого взаимодействия. Архитектура «клиент-сервер» может быть представлена в следующих вариантах.

Двухзвенная архитектура. В ней компоненты системы распределяются между двумя узлами – клиентом и сервером. Сервер, не вызывая сторонние сетевые приложения и не обращаясь к сторонним сетевым ресурсам, напрямую и в полном объеме отвечает на клиентские запросы.

В роли первого звена могут выступать клиентские компьютеры с прикладными программами, с помощью которых происходит обращение пользователей к базе данных по сети.

В роли второго звена выступает сервер базы данных, участвующий в обработке данных.

Трехзвенная архитектура. Как и в двухзвенной архитектуре, в трехзвенной присутствуют те же компоненты, и добавляется третье звено – сервер приложений. В данной архитектуре компоненты распределяются иначе, чем в двухзвенной. На стороне клиента выполняется представление данных, прикладной компонент выполняется на сервере приложений, а управление ресурсами происходит на сервере базы данных, который предоставляет запрашиваемые данные.

Схема трехзвенной архитектуры клиент-сервер представлена на рисунке ниже.

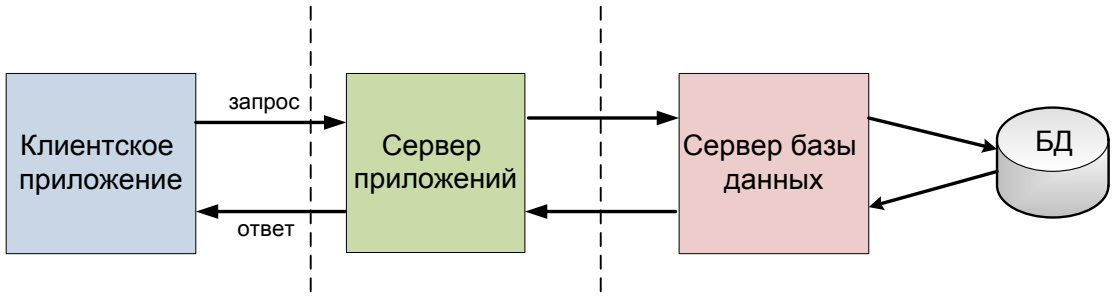

Рисунок 1.8 – Схема трехзвенной архитектуры клиент-сервер

В силу того, что в двухзвенной архитектуре одним сервером обрабатываются все запросы от клиентов, она является более простой в реализации. Однако она является менее надежной и более требовательной к производительности сервера. Трехзвенная структура сложнее в реализации и развертывании, но она более надежна, имеет высокую степень масштабируемости и гибкости, так же предоставляет более высокую производительность, т.к. все задачи распределены между разными серверами.

Выбор архитектуры клиент-сервер можно осуществить, проведя сравнительный анализ положительных сторон каждой из архитектур.

Результаты сравнительного анализа архитектур построения АИС приведены в таблице 1.1.

Таблица 1.1 – Результаты сравнительного анализа архитектур клиент-сервер

Архитектура / Требование	Клиент-сервер, двухзвенная	Клиент-сервер, трехзвенная
Высокая надежность	-	+
Высокая степень масштабируемости	-	+
Простая в реализации и развертывании	+	-
Пониженные требования к производительности сервера	-	+
Итого:	1	3

По итогам сравнительного анализа, представленных в таблице 1.1, принято решение о построении системы управления подписками на основе трехзвенной архитектуры клиент-сервер.

1.4.2 Выбор средств разработки системы управления подписками

Язык программирования, выбранный для разработки системы управления подписками, может в значительной степени повлиять на процесс разработки системы, так и на конечный результат. Поддержка языком различных парадигм программирования может повлиять на структуру программы. Для разработки системы управления подписками был выбран язык Java. Java — это объектно-ориентированный язык программирования, который был разработан компанией Sun Microsystems и официально был выпущен 23 мая 1995 года. Он имеет ряд положительных особенностей, которые необходимы при разработки ИС в рамках данной работы:

− большинство дополнительных библиотек доступно под свободными лицензиями с открытым исходным кодом;

− Java является объектно-ориентированным языком, и как следствие можно создавать модульные программы, исходный код которых может использоваться многократно;

− Java обладает большой библиотекой программ для передачи данных на основе таких протоколов TCP/IP, как HTTP или FTP;

− Java является надежным языком , основное внимание в языке Java уделяется раннему обнаружению возможных ошибок, динамической проверке (во время выполнения программы), а также исключению ситуаций, подверженных ошибкам;

− язык Java позволяет создавать системы, защищенные от вирусов и постороннего вмешательства.

Таким образом, в качестве языка программирования система управлениями подписками (СУП) для данной работе выбран язык JAVA.

База данных (БД) – структурированный организованный набор данных, описывающих характеристики каких-либо физических или виртуальных систем. Система управления базами данных – это наиболее распространенное и эффективное средство, предназначенное для организации и ведения логически взаимосвязанных данных на машинном носителе, а также обеспечивающее доступ к данным.

В качестве СУБД была выбрана Oracle, т.к. она предлагает множество опций и функций уровня базы данных для повышения производительности, поддерживает горизонтальное и вертикальное масштабирование, предоставив при этом пользователям широкий набор функций для обеспечения безопасности данных: проверка подлинности, авторизация, проверка прав доступа, шифрование данных и другие.

1.4.3 Состав и содержание работ

Разработка системы управления подписками должна проходить в три этапа: разработка эскизного проекта, разработка технического проекта и разработка технической документации на систему и ее части.

На этапе разработки эскизного проекта должен быть определен состав сущностей области постоянного хранения, построена концептуальная модель данных, построена логическая модель данных. Провести эскизное проектирование интерфейсов пользователя.

Более подробное описание наименований и результатов по окончанию работ на этапе разработки эскизного проекта приведено в таблице 1.4.

Таблица 1.4 – Этапы разработки эскизного проекта

Наименование работ	Результат
Определение общей функциональной и технической архитектур.	Общее описание функциональной и технической архитектур. Разработана концептуальная модель данных.
Разработка логической модели данных.	- Логическая модель данных. - Определен состав сущностей области постоянного хранения.
Эскизное проектирование интерфейсов пользователя.	Общее описание интерфейсов ввода и предоставления данных.

На этапе разработки технического проекта необходимо привести описание функциональной архитектуры, построить физическую модель данных, провести проектирование интерфейсов пользователя.

Более подробное описание наименований и результатов по окончанию работ на этапе разработки технического проекта приведено в таблице 1.5.

Таблица 1.5 – Этапы разработки технического проекта

Наименование работ	Результат
Определение функциональной и технической архитектур.	− Описание функциональной архитектуры (включая описание каждой функции, задачи, методов реализации). − Описание технической архитектуры.

Построение физической модели данных.	Построена физическая модель данных.
Проектирование интерфейсов пользователя.	Описание интерфейсов ввода и предоставления данных.

На этапе разработки технической документации на систему и ее части должны быть описаны общее описание системы, руководство пользователя.

Вывод к главе 1

1. Netcracker Technology — это мировой лидер в разработке и внедрении программного обеспечения для операторов связи. Netcracker имеет сложную организационную и IT-структуры.
2. Обеспечения удобного интерфейса управления подписками может обеспечить более высокую производительность труда, сокращая при этом затраты на время сбора и поиска нужной и актуальной информации.
3. Моделирование происходящих в организации бизнес-процессов, реализующих ее цели и задачи, является одним из основных этапов разработки информационной системы. Правильное составление и анализ бизнес-процессов обеспечивает и облегчает дальнейшую разработку информационной системы.
4. Разработка проекта решения задачи - сложный процесс, включающий в себя множество этапов, таких как: моделирование бизнес-процессов, формирование требований к системе, выбор архитектуры и средств разработки ИС, а также определение состава и содержание работ.
5. Исходя из анализа архитектур, языков программирования и СУБД для разработки ИС была выбрана трехзвенная клиент-серверная архитектура, язык программирования Java и СУБД Oracle Database.

Глава 2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ СИСТЕМЫ УПРАВЛЕНИЯ ПОДПИСКАМИ

2.1 Проектирование эскизного проекта

2.1.1 Общее описание функциональной и технической архитектур информационной системы

Функциональная архитектура ИС состоит из следующих элементов:

- Web-браузер, установленный на ПЭВМ пользователя;
- сервер приложений;
- разрабатываемое приложение;
- система управления базами данных.

Web-браузер является инструментом, необходимым пользователю разрабатываемой информационной системы для обработки, вывода и просмотра Web-страниц приложения, сгенерированных системой в ответ на запрос пользователя.

Web-браузер должен выполнять такие функции, как:

- формирование запроса на основании действий пользователя и отправка запроса на сервер;
- обработка и отправка ответа пользователю в виде Web-страниц, содержащих запрашиваемую им информацию.

Разрабатываемое приложение должно выполняться на сервере приложений, поэтому сервер приложений является необходимым элементом архитектуры.

Сервер приложений может выполнять такие функции как:

- прием и передача запросов пользователей и пакетов данных;
- регулировка доступа запросов от клиента к серверу базы данных, обеспечивая баланс нагрузки сервера БД.

СУБД – система управления базами данных, представляет собой программный комплекс поддержки совокупности данных. Такая система

предназначена для формирования, ведения и использования базы данных многими прикладными программами и пользователями.

Система управления базами данных обеспечивает:

- защиту логической целостности базы данных;
- управление данными в памяти, как во внешней, так и в и оперативной;
- управление транзакциями, происходящих после определенных действий пользователей;
- обработку и синхронизацию работы нескольких прикладных систем, использующих БД или пользователей;
- поддержку языков БД.

Разрабатываемому приложению необходимо содержать логику бизнес-процессов, связанных с предметной областью, а также обеспечивать взаимосвязь остальных компонентов информационной системы.

Разрабатываемое приложение должно выполнять следующие функции:

- обработка запросов, полученных совершения определенных действий пользователя в Web-интерфейсе системы;
- обеспечение и контроль доступа к базе данных, формирование и отправка к ней запросов;
- получение и обработка результатов, полученных после запроса к базе данных;
- динамическое формирование контента Web-страницы и отправка пользователю.

Техническую архитектуру системы можно разделить на три уровня:

- уровень ядра;
- уровень распределения;
- уровень доступа.

На уровне ядра находятся такие элементы системы, как сервер базы данных и Web-сервер.

На уровне распределения находятся такие элементы системы, как маршрутизаторы и коммутаторы. На уровне доступа находятся такие устройства, как ПЭВМ пользователей системы, а также коммутаторы.

Так же в техническую архитектуру сети входят элементы пассивного сетевого оборудования, такие как: провода (UTP 5 cat., оптоволокно), патч-панели, монтажные шкафы и стойки, телекоммуникационные шкафы.

2.1.2 Построение концептуальной и логической моделей данных

Описать главные (основные) сущности и отношения между ними возможно с помощью построения концептуальной модели базы данных. Концептуальная модель включает в себя сами сущности, отношения, показанных соединяющими их линиями, а также элементов из которых они состоят. На рисунке 2.1 представлена концептуальная модель ИС.

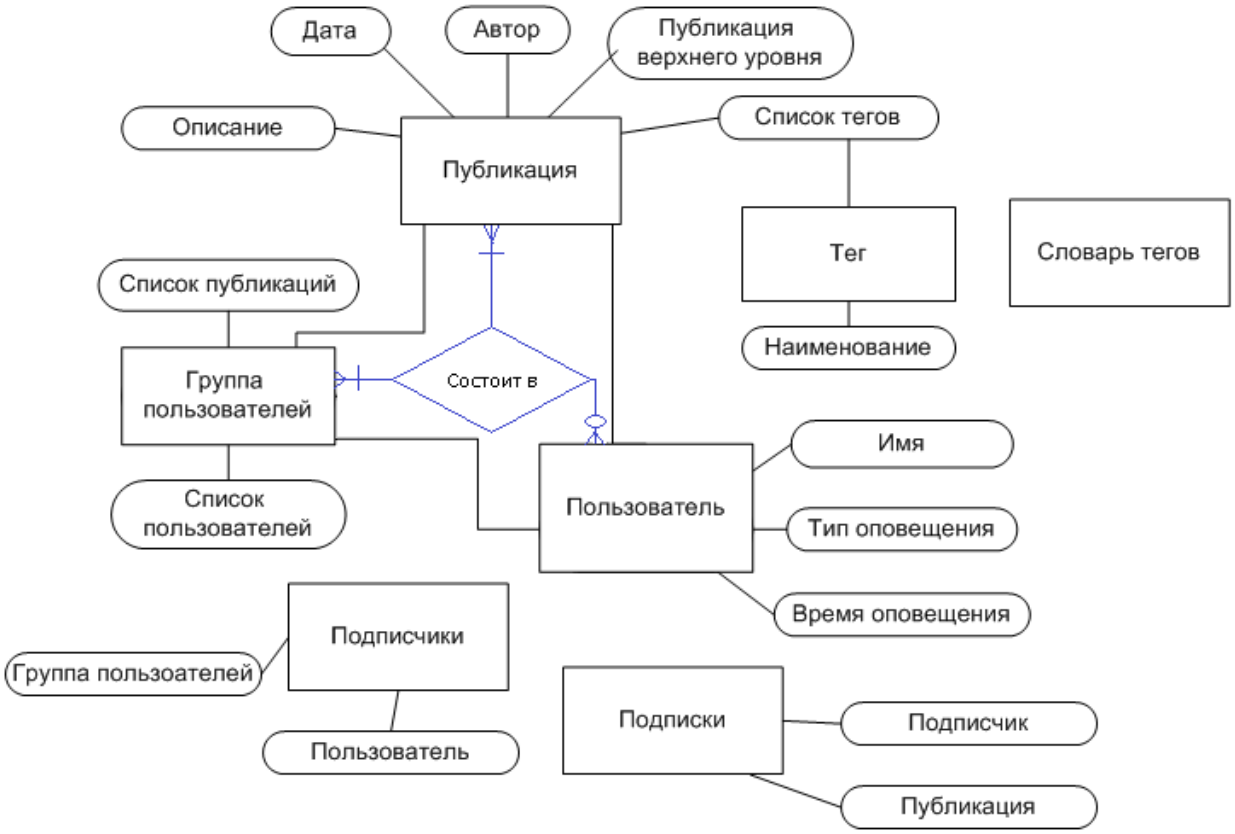

Рисунок 2.1 – Схема концептуальной модели системы

На представленной выше модели отображены все сущности, необходимые для разработки ИС. С помощью построенной выше модели можно составить логическую модель данных разрабатываемой системы, в которой для всех сущностей ИС определяются атрибуты, описания и ограничения, уточняется состав сущностей и взаимосвязи между ними.

Разработка логической модели данных производиться по методу IDEF1X, и весь процесс можно разбить на такие этапы, как: определение сущностей с помощью сопоставления с инфологической моделью, выявление зависимостей между сущностями ИС (ассоциации и агрегации преобразуются в не идентифицирующие отношения), задание первичных и альтернативных ключей, определение не ключевых атрибутов сущностей ИС, построение IDEF1X –диаграммы, которое осуществляется с помощью доступных CASE-средств.

Одним из таких CASE- средств является dbForge, имеющий бесплатную пробную версию. Разработанная в dbForge Studio логическая модель данных ИС имеет вид, представленный на рисунке 2.2.

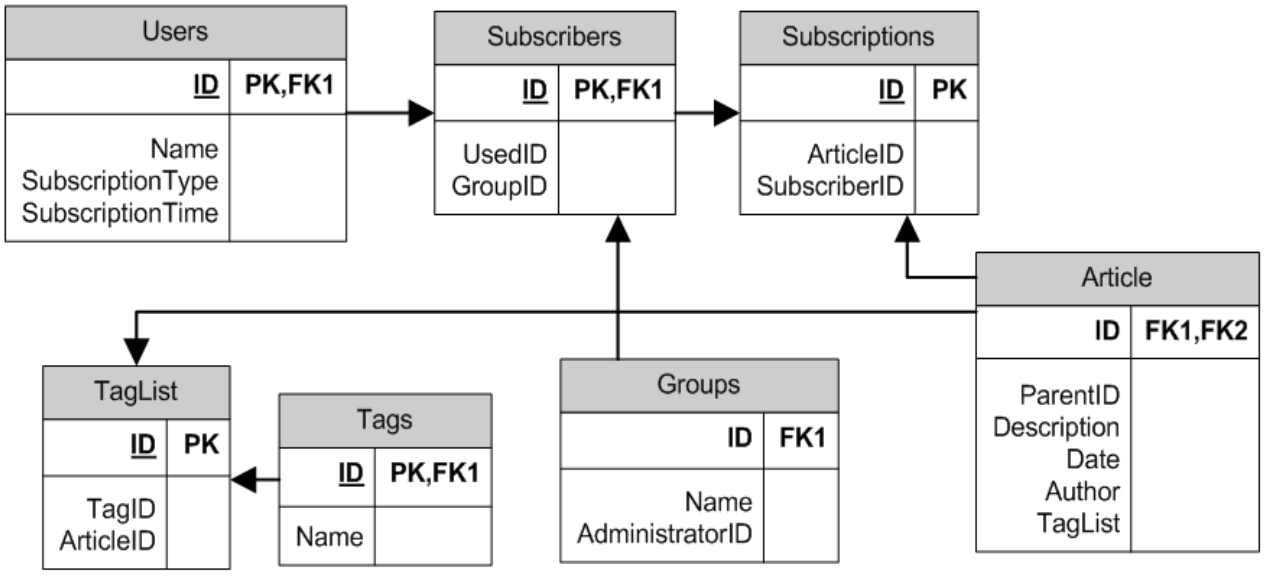

Рисунок 2.2 – Логическая модель данных системы

Данная модель включает в себя названия сущностей и их элементов на английском языке, что позволяет использовать эти наименования в дальнейшем при составлении физической модели данных ИС.

2.1.3 Проектирование дизайн-макета web-интерфейса пользователя

Подключение к системе должно происходить с использованием Web-браузера и Web-интерфейса. Авторизация проходит автоматически по средством внутренней программы авторизации сотрудников в корпоративной системе компании. Стартовой страницей при подключении должна являться страница с категориями публикаций. В шапке web-страницы должно быть меню навигации, по которому пользователь может перемещаться на страницы с такими разделами, такими как: «Categories», «Last Publications», «My Subscriptions», «Settings». Ниже меню навигации должно быть поле поиска.

Основной контент страницы должен быть поделен на две части. Слева должна быть представлена область, где будет отражена запрашиваемая пользователем информация, в правой части страницы должен быть блок популярных тегов в виде ссылок для более быстрого перехода к наиболее часто посещаемым разделам информационных ресурсов.

На рисунке 2.3 показан ориентировочный дизайн-макет Web-страницы с открытым разделом «Categories».

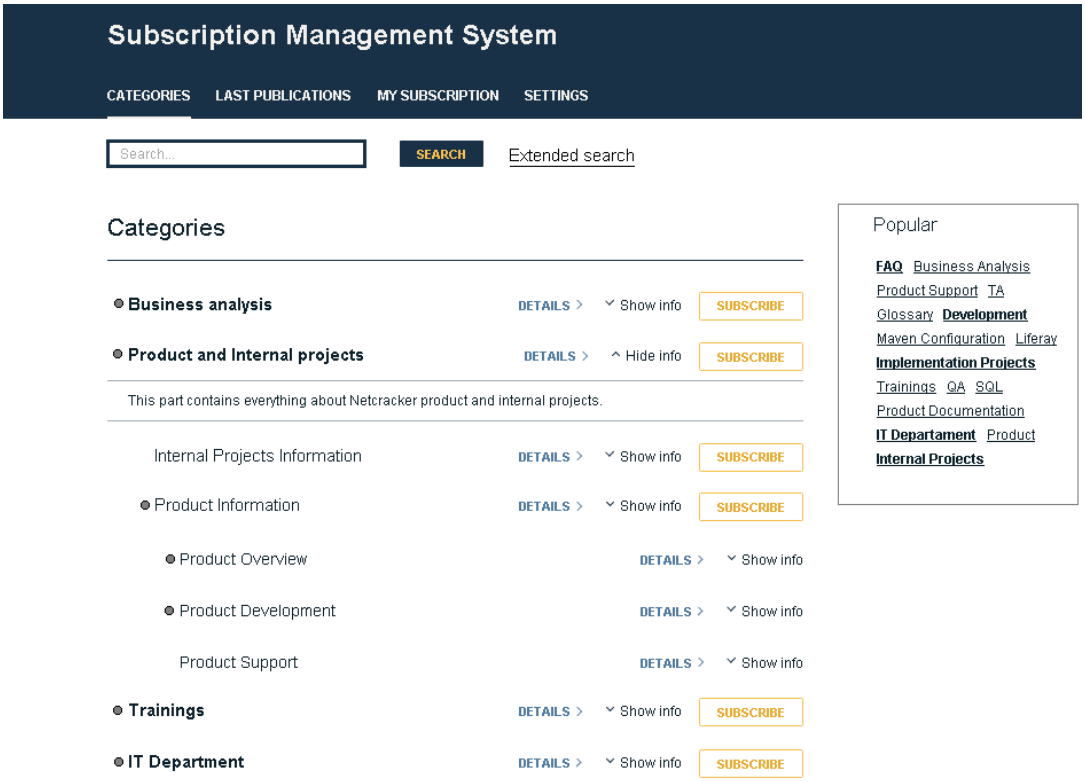

Рисунок 2.3 – Дизайн-макет Web-страницы

На представленном выше макете отображены основные элементы: строка с пунктами навигации, по которым пользователь может посмотреть содержимое других разделов, место для основного контента системы, название раздела, который открыт в данный момент и список популярных тегов.

На макете, представленном на рисунке 2.5, отображен список публикаций, на которые подписан пользователь. Для каждой публикации отображается необходимая информация: название, дата публикации, автор, список тегов, краткая информация о данной публикации, кнопка, позволяющая пользователю отписаться от рассылки по конкретной публикации.

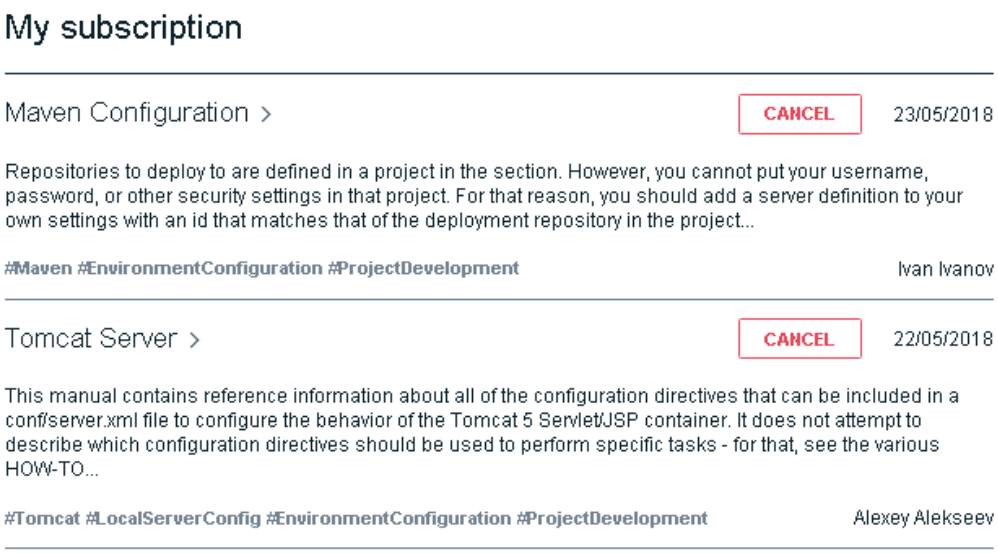

Рисунок 2.5 – Макет таблицы с выведенным списком публикаций, на которые подписан пользователь

Рассмотренные выше макеты отвечают всем требованиям их можно рассматривать в качестве опорных вариантов дизайна при разработке информационной системы управления подписками.

Выводы к главе 2

Функциональная архитектура ИС состоит из множества элементов: Web-браузер, сервер приложений, разрабатываемое приложение, система управления базами данных.

Описать главные (основные) сущности и отношения между ними возможно с помощью построения концептуальной, логической моделей базы данных, на основе которых, в дальнейшем будет разрабатываться физическая модель данных.

Проектирование пользовательского интерфейса является одной из важных частей разработки информационной системы, он должен отвечать всем требованиям заказчика и включать в себя определенный на этапе анализа функционал.

Глава 3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ СИСТЕМЫ УПРАВЛЕНИЯ ПОДПИСКАМИ

3.1 Проектирование технического проекта

3.1.1 Построение физической модели данных

Физическая модель данных зависит от конкретной СУБД, и, по сути, является отображением системного каталога.

В физической модели должна содержатся вся информация обо всех объектах базы данных. Так как физическая модель зависит от конкретной реализации СУБД, одной логической модели вполне могут соответствовать несколько различных физических моделей. В физической модели важно описать всю информацию о конкретных физических объектах — таблицах, колонках, индексах, процедурах и так далее. Также, в отличие от логической модели, в физической важно описать типы атрибутов для каждой из таблиц.

При помощи индексов, декларативных ограничений целостности, триггеров, хранимых процедур и т.д. реализуются ограничения, имеющиеся в логической модели данных.

Отношения, установленные при построении логической модели данных, преобразуются в таблицы базы данных, атрибуты становятся столбцами таблиц, для ключевых атрибутов создаются уникальные индексы, домены преображаются в типы данных, принятые в конкретной СУБД.

На рисунке 3.1 представлена физическая модель данных ИС, построенная для СУБД Oracle в дизайнере dbForge Studio, в которой перечислены необходимые для разрабатываемой ИС сущности:

- Users;
- Subscribers;
- Subscriptions;
- Article;
- Tag List;
- Tags;

– Groups.

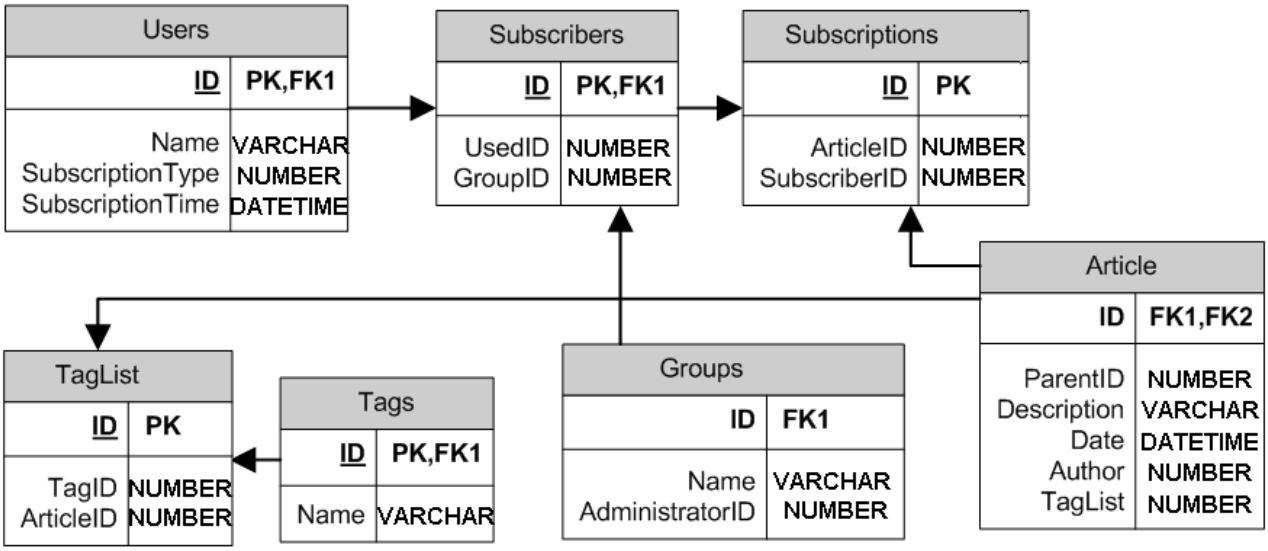

Рисунок 3.1 – Физическая модель данных ИС

Данная физическая модель данных отображает все необходимые поля, а также показывает первичные и вторичные ключи таблиц.

3.1.2 Проектирование интерфейса пользователя

В соответствии с разработанными дизайн-макетами Web-интерфейса системы, а также в соответствии с техническим заданием спроектирован интерфейс пользователя системы. Интерфейс спроектирован в строгих сине-серых тонах, с использованием цветовой гаммы, утвержденной компанией.

Спроектированный интерфейс пользователя при входе в систему представлен на рисунке 3.2.

На главной странице, которой является раздел «Categories», отображено название раздела и основной контент, который включает в себя список публикаций верхнего (базового) уровня, содержащий также подсписки публикаций, размещающихся в них.

Публикация, содержащая в себе элементы может быть раскрыта вниз, показывая тем самым список этих элементов. Рядом с каждой публикацией есть кнопка «Subscribe», нажатие которой дает возможность подписаться на рассылку этой публикации и всех ее подкатегорий.

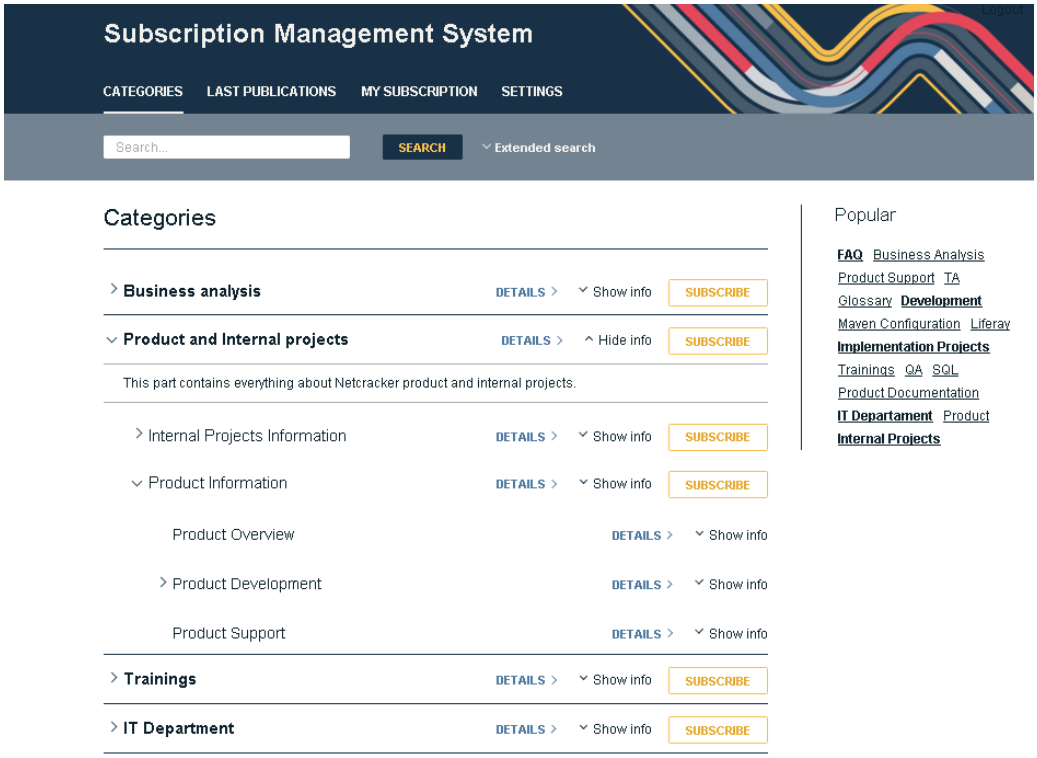

Рисунок 3.2 – Спроектированный интерфейс пользователя ИС, раздел «Categories»

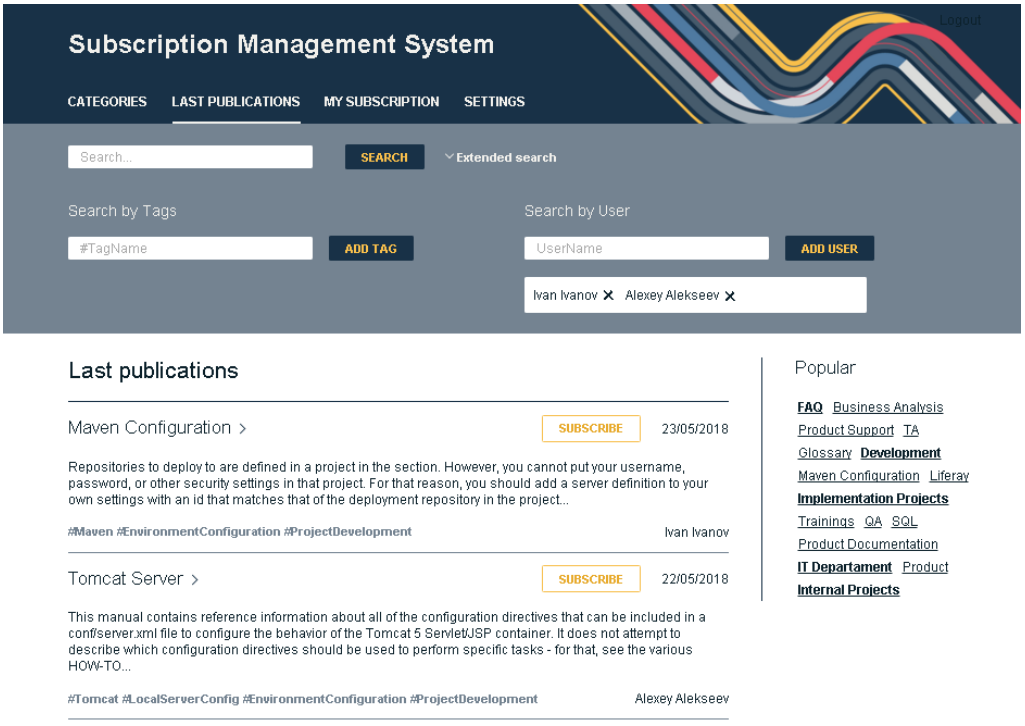

Рисунок 3.3 – Спроектированный интерфейс пользователя ИС, раздел «Last publications»

Интерфейс пользователя со списком публикаций, на рассылку которых подписан пользователь представлен на рисунке 3.4.

Данный раздел системы представляет все публикации, на рассылку которых подписан пользователь. Здесь же он имеет возможность отписаться от рассылки на не интересующую более публикацию путем нажатия на кнопку «Cancel».

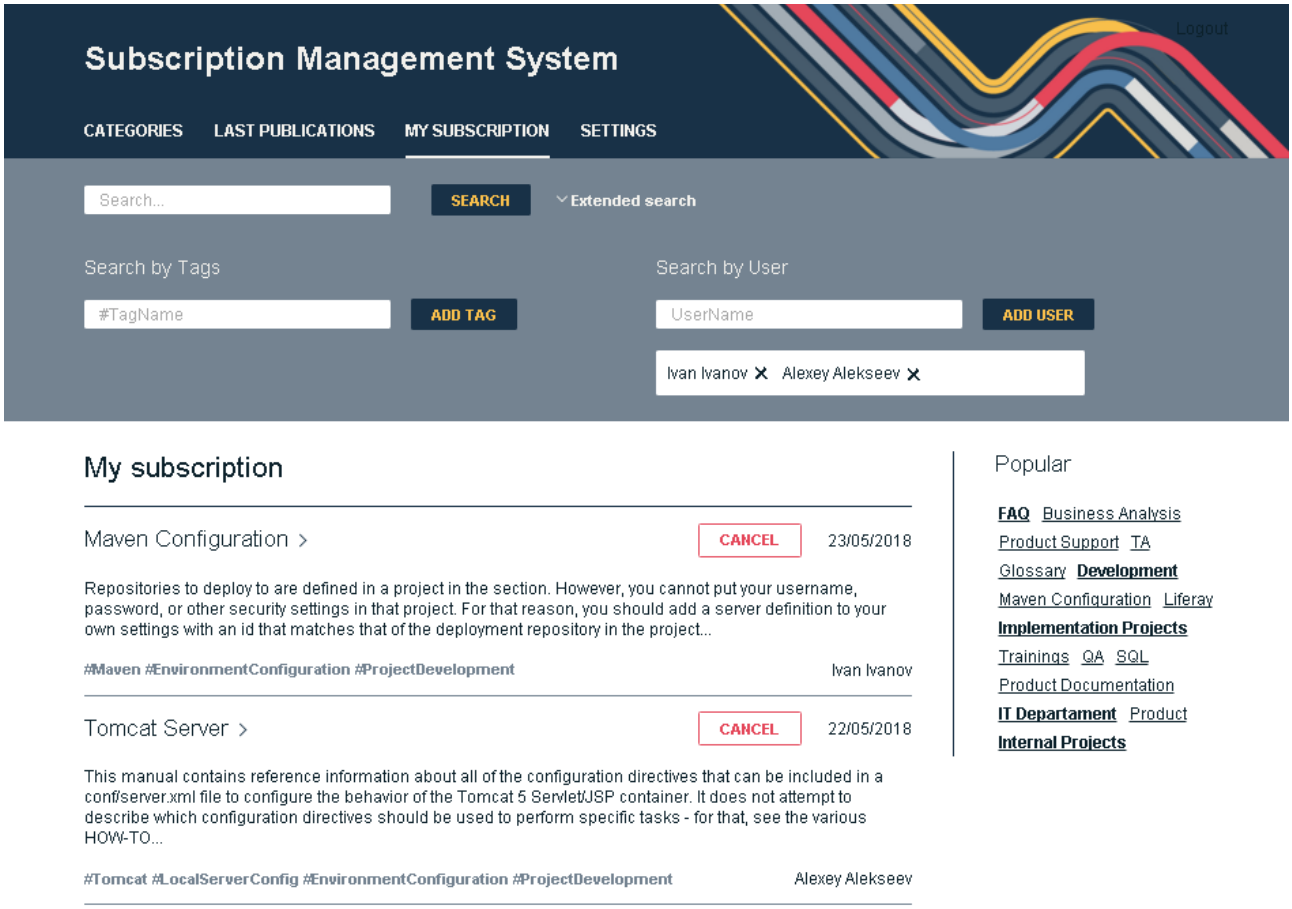

Рисунок 3.4 – Спроектированный интерфейс пользователя ИС, раздел «My subscription»

Спроектированный интерфейс пользователя отображает весь описанный ранее необходимый функционал системы и полностью соответствует представленным ранее дизайн-макетам.

3.2 Разработка системы управления подписками

Полный код разработанной информационной системы не может быть представлен, так как является собственностью ООО НетКрэкер.

Сборка проекта будет осуществляться с помощью инструмента Maven. Maven – это инструмент, осуществляющий сборку Java-проекта, с его помощью производиться компиляция проекта, создание исполняемых файлов jar или war, создание дистрибутивов программ, генерация документации. Для сборки jar файла можно, конечно, запустить команду и из командной строки в консоли, но для больших проектов эта команда может оказаться очень громоздкой, из-за множества входящих в нее параметров. Инструменты сборки помогают собрать jar файлы намного быстрее и проще, т.к. вся конфигурация находиться в конфигурационных файлах, которые просто редактировать, добавлять необходимые зависимости, подключать нужные модули и прочее. Есть несколько инструментов сборки. Кроме Maven одним из популярных на сегодняшний день является инструмент Ant. Он был выпущен намного раньше, чем Maven, он имеет огромную поддержку, он очень гибкий, он не навязывает определенную структуру проекта, соглашения по написанию кода. Однако, Maven по сравнению с ним имеет ряд преимуществ:

- Независимость от операционной системы - один и тот же проект может быть собран в любой ОС.
- Управление зависимостями – большие проекты, как правило, пишутся с использованием различных библиотек, которые могут быть разных версий и тоже могут содержать в себе зависимости на другие библиотеки. Maven может управлять этими сложными зависимостями и разрешает конфликты версий.
- Проект можно собрать из командной строки – это бывает необходимо для сборки на сервере.
- Интеграция со средами разработки. Все популярные среды разработки могут открыть проекты, собирающиеся с помощью maven.
- Декларативное описание проекта.

Поэтому для сборки был выбран именно этот инструмент. Конфигурационным файлом является pom.xml. Именно в нем настраиваются все зависимости. Для разработки ИС понадобиться зависимость для подключения Spring Framework и Spring MVC. Это популярный на сегодняшний день фрэймворк для разработки Java проектов, который представляет собой сборник решений многих задач, которые могут встретиться при разработки больших enterprise-приложений, а также представляет собой альтернативу модели Enterprise JavaBeans.

Подключение в pom.xml показано на рисунке 3.6.

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>4.3.1.RELEASE</version>
</dependency>


 <dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.8.3</version>
</dependency>
```

Рисунок 3.6 – Подключение Spring Framework и Spring MVC в pom.xml

Spring Framework содержит в себе множество различного функционала и состоит из большого числа модулей:

- модуль аспектно-ориентированного ;
- модуль доступа к данным;
- модуль управления транзакциями;
- модуль MVC;
- модуль удалённого доступа;
- модуль аутентификации и авторизации;
- модуль удалённого управления;

- модуль работы с сообщениями;
- модуль для тестирования.

Для разработки ИС сейчас наибольший интерес представляет модуль Spring MVC. Spring MVC обеспечивает построение приложения, согласно архитектуре известного паттерна Model — View — Controller (Модель — Отображение — Контроллер) при помощи практически не связанных между собой готовых компонентов.

Паттерн MVC разделяет приложение на три основных компоненты: логику приложения, логику по работе с данными и логику отображения. Такое разделение архитектуры приложения обеспечивает огромную гибкость, возможности для масштабирования системы, расширение функционала или бизнес-логики и простоту поддержки ИС в целом. Функции, выполняемые отдельными компонентами перечислены ниже.

Модель содержит в себе функционал по работе с данными, она отвечает за выполнение таких операций, как:

- сохранение данных;
- создание данных;
- добавление данных;
- удаление данных.

Этот функционал, как правило располагается в классах-обработчиках данных, написанных для каждой сущности отдельно. Как правило, классы самих сущностей представлены в виде POJO (Plain Old Java Object) - Java классов, содержащих в себе поля для данных и методы получения и установки этих полей. В процессе работы разрабатываемая информационная система будет создавать для каждой записи в базе данных один экземпляр класса соответствующей сущности.

Для отображения всех имеющихся у пользователя подписок, а также для отображения последних публикаций данные будут передаваться в виде списка экземпляров этих классов.

Контроллер обеспечивает обработку запросов, пришедших от пользователей, создание моделей и отправку полученных данных от модели в компонент View для отображения результатов пользователю.

Отображение отвечает за отображение данных Модели посредством генерации кода в формате HTML, который рисуется браузером в виде страниц. Общая схема паттерна MVC представлена на рисунке 3.7.

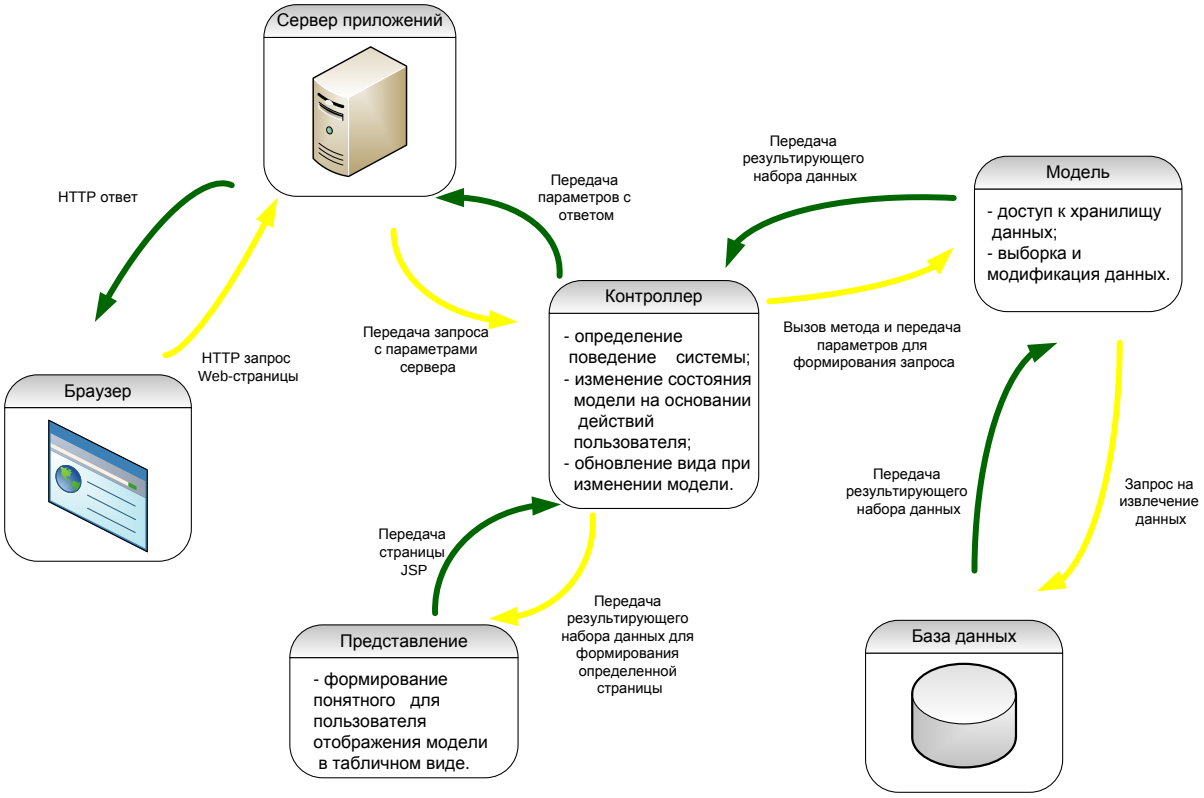

Рисунок 3.7 – Общая схема архитектуры приложения в соответствии с паттерном проектирования MVC (Model – View - Controller)

Фрагмент класса-контроллера, принимающего запросы от пользователей представлен на рисунке 3.8.

Данный класс содержит в себе несколько аннотаций, таких как: @Controllerannotation, которая показывает, что этот класс является контроллером, аннотация @RequestMapping, которая используется для связывания с URL для всего класса или для метода обработчика.

В соответствии с архитектурой REST каждое действие над данными должно задаваться с помощью методов:

- GET (получить);
- PUT (добавить);
- POST (изменить);
- DELETE (удалить).

Перечисленные методы описываются в контроллере в качества параметра аннотации @RequestMapping.

```
@Controller
@RequestMapping("/hello")
public class HelloController {

    @RequestMapping(method = RequestMethod.GET)
    public String getUsers(ModelMap model) {
        model.addAttribute("usersList", UserModel.getUserList());
        return model;
    }

    @RequestMapping(method = RequestMethod.DELETE)
    public String getUsers( @PathVariable("id") String id) {
        UserModel.deleteUser(id);
        model.addAttribute("usersList", UserModel.getUserList());
        return model;
    }
```

Рисунок 3.8 – Фрагмент класса - контроллера

Для каждой сущности в соответствии с ранее разработанной физической моделью данных необходимо создать POJO-класс с описанием этой сущности. POJO классы для сущностей состоят из полей и методов для получения и установки этих полей. Кроме этих методов, как правило, никакая логика в них не храниться.

Для каждой единицы информации определяются следующие действия (на примере подписки):

– GET /subscription – получает список всех подписок;

– GET /subscription/{id} – получает полную информацию о объекте;

– PUT /subscription – создает новый объект;

– POST /subscription/{id} – изменяет данные с идентификатором {id}, возможно заменяет их.

– DELETE / subscription /{id} – удаляет все хранящиеся в базе данных данные с идентификатором {id}.

Код класса, определяющего сущность User приведен на картинке 3.9.

```
public class User {

    private int id;
    private String name;
    private String subscriptionType;
    private String subscriptionTime;
    public User() {
    }

    public User(int id, String name, String subscriptionTime,
                String subscriptionType) {
        this.id = id;
        this.subscriptionTime = subscriptionTime;
        this.subscriptionType = subscriptionType;
        this.name = name;
    }

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = ud;
    }

    public String getName() {
        return name;
    }
    public void setName(String newName) {
        this.name = newName;
    }
```

Рисунок 3.9 – Фрагмент POJO-класса для сущности User

Аналогичным образом создаются классы для других сущностей.

Для возвращаемых данных пользователю будет использоваться формат JSON, который будет осуществляться посредством инструмента Jackson (jackson-databind), который может конвертировать данные из класса в этот формат и обратно. Для его использования необходимо подключить его как зависимость в файле pom.xml, подключение показано на рисунке 3.10.

```
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.8.3</version>
</dependency>
```

Рисунок 3.10 – Подключение зависимости на инструмент Jackson

Схема преобразования данных из POJO-класса показана на рисунке 3.11.

Рисунок 3.11 – Схема преобразования данных из POJO-класса в формат JSON.

В качестве View компонента будет использован шаблонизатор Closure Templates - это система клиентских и серверных шаблонов, которая помогает динамически создавать элементы HTML и UI многократного использования. Они имеют простой синтаксис, и их можно настроить их в соответствии с

потребностями приложения. В отличие от традиционных систем шаблонов, в которых обычно создается один монолитный шаблон на странице, Closure Templates можно разбить на множество небольших компонентов, которые нужны для создания пользовательского интерфейса.

Шаблоны Closure реализованы как для JavaScript, так и для Java, поэтому, можно использовать те же шаблоны как на стороне сервера, так и на стороне клиента. Они используют модель данных и синтаксис выражений, которые работают для любого языка.

Для клиентской стороны шаблоны Closure предварительно скомпилированы в эффективный JavaScript. У Closure Templates есть ряд преимуществ, исходя из которых и был выбран именно этот шаблонизатор:

- Шаблоны Closure обеспечивают простой способ создания фрагментов HTML для пользовательского интерфейса приложения и помогают отделить логику приложения от отображения;
- Шаблоны Closure работают с JavaScript или Java;
- Скорость на стороне клиента. Шаблоны Closure скомпилированы для эффективных функций в JavaScript для максимальной производительности;
- Легко читается;
- Предназначен для программистов. Шаблоны - это просто функции, которые могут вызывать друг друга. Синтаксис включает в себя конструкции, знакомые программистам;
- Хорошо работает в любой среде веб-приложений в сочетании с любыми библиотеками, фреймворками или другими инструментами.

Код шаблона представляет собой привычный фрагмент HTML, с определением внутри него переменных, при формировании ответа сервер подставляет в качестве переменных полученные от модели данные и генерирует тем самым итоговый HTML.

В выбранном шаблонизаторе присутствуют такие операторы, как:

- циклы;

- простые условия (if);
- условия по входному параметру (switch);
- переменные и многое др., что упрощает разработку шаблона.

Фрагмент шаблона для списка подписок, имеющихся у пользователя представлен на рисунке 3.12.

```
<main class="content">
    <div class="content__row">
        <div class="content__col _main">
            <div class="title">
                <div class="title__link">My subscription</div>
            </div>
            <foreach item from $subscriptions>
                <div class="article">
                    <div class="article__header">
                        <div class="article__title">
                            {$item.title}
                        </div>
                        <div class="article__control">
                            <input class="button _tertiary _big"
                                   type="button" value="cancel">
                        </div>
                        <div class="article__date">{$item.date}</div>
                    </div>
                    <div class="article__content">
                        {item.description}
                    </div>
                    <div class="article__bottom">
                        <div class="article__tags">
                            <foreach tag in $item.tags>
                                {$tag}
                            </foreach>
                        </div>
                        <div class="article__author">
                            {$item.author}
                        </div>
                    </div>
                </div>
            </foreach>
```

Рисунок 3.12 – Фрагмент шаблона для отображения списка подписок пользователя.

Аналогичным образом оформляются и другие шаблоны.

Благодаря четко разбитой архитектуры приложения на три компонента, весь процесс был поделен на множество этапов:

- конфигурация Maven;
- подключения Spring Framework и Spring MVC;
- конфигурация Jackson для конвертации классов POJO в формат JSON;
- написание класса компонента контроллера;
- написание POJO классов для модели;
- написание классов, содержащих логику работы с данными для компонента Model;
- написание шаблонов в качестве View компонента.

В целом, с помощью использования описанных выше инструментов разработка системы оказалась не слишком трудоемкой, система получилось открытой для дальнейшего масштабирования, если таковое понадобиться в будущем.

3.3 Тестирование системы и разработка рабочей документации на систему и ее части

Одним из важных документов на ИС является руководство пользователя. Назначение данного документа состоит в том, чтобы предоставить пользователям необходимую информацию о том, как правильно пользоваться системой, о ее компонентах, а также обеспечить пользователям возможность самостоятельно решать основные задачи при работе с ИС.

Для данной информационной системы управления подписками, руководство пользователя состоит из следующих разделов:

- назначение системы;
- вход в систему;
- разделы системы: «Categories», «Last publications», My subscription», Search», «Settings».

Полное руководство пользователя системы приведено в приложении А.

Для контроля качества разрабатываемой информационной системы, а также для заблаговременного обнаружения ошибок в работе ИС или в отображении пользовательского интерфейса ИС необходимо тестирование системы.

Оно помогает заранее обнаружить какие-либо ошибки или недочеты в работе ИС, облегчает понимание удобства или его недостатка при использовании спроектированного интерфейса пользователя, что позволяет на раннем этапе разработки вносить поправки и может существенно повысить качество разрабатываемого продукта.

Для осуществление грамотного тестирования необходимо описать все возможные задачи или «Test Cases» (TC), которые могут быть решены с помощью разрабатываемой ИС. Ниже описаны возможные TC для разрабатываемой ИС.

Настройка и сохранение параметров оповещений

Для того, чтобы настроить тип нотификации, время нотификации и настроить фильтрацию по тегам необходимо следующие действия.

1. Открыть приложение «Subscription Managment System» в web-браузере.

Результат: пользователь видит страницу web-приложения с активной вкладкой «Categories».

2. Нажать на вкладку «Settings» в меню навигации

Результат: отобразиться информация с настройками для пользователя.

3. Выбрать тип уведомления о подписке с помощью Notification Type

4. Выбрать время уведомлений об изменений публикаций, добавления новых публикаций в категории с помощью Notification Time и нажать «Change»

Результат: после нажатия на «Change» появляется всплывающее окно с информацией о том, что время уведомления успешно изменено.

5. В меню Filtration в поле «Enter #TagName» ввести тег, который необходимо исключить из отображения публикаций в приложении, и нажать кнопку «Add Tag».

Результат: в поле «Ignored categories» появиться тег, который добавили в исключение.

Итоговый результат: настройки должны быть сохранены и при уходе\возвращении на страницу должны отображаться в том же состоянии, что и до ухода со страницы.

Подписка на интересующую категорию\публикацию

Для того чтобы подписаться на интересующею категорию необходимо выполнить следующие действия:

1. Открыть приложение «Subscription Managment System» в web-браузере.

Результат: пользователь видит страницу web-приложения с активной вкладкой «Categories».

2. Выбрать интересующею категорию и нажать кнопку «Subscribe».

Итоговый результат: после нажатия на «Subscribe» появляется всплывающее окно с информацией о том, что запрос на подписку успешно отправлен.

Отмена подписки на публикацию

Для того чтобы просмотреть активные подписки, а также отписаться от публикаций и категорий необходимо выполнить следующие действия:

1. Открыть приложение «Subscription Managment System» в web-браузере

Результат: пользователь видит страницу web-приложения с активной вкладкой «Categories»

2. Нажать на вкладку «My Subscriptions» в меню навигации

Результат: отобразится информация с активными подписками на публикации и категории.

3. Выбрать нужную публикацию, от которой необходимо отписаться и нажать кнопку «Cancel».

Итоговый результат: после нажатия на "Cancel" появляется всплывающее окно с информацией о том, что запрос на отписку успешно отпрален.

После того как все ТС описаны можно приступить к тестированию информационной системы и замерить время, затраченное на их прохождение. Также, для сравнения необходимо замерить время, затраченное на аналогичные операции, приводящие к тем же результатам до внедрения ИС для сравнения. Результат сравнения представлен в таблице 3.1.

Таблица 3.1 – Показатели тестирования системы

	Первоначальные затраты	Затраты после внедрения ИС	Абсолютное изменения затрат	Коэффициент изменения затрат
Среднее время процесса подписки на публикацию	3мин	0.5мин	2.5мин	83.3%
Среднее время отмены подписки на публикацию	2мин	0.5мин	1.5мин	75%

Таким образом, после прохождения основных ТС как в разработанной ИС, так и без ее использования, получаем для сравнения и анализа показатели тестирования. Можно отметить, что временные затраты после внедрения ИС значительно меньше, чем до ее внедрения.

Вывод к главе 3

Описать главные (основные) сущности и отношения между ними возможно с помощью построения физической модели базы данных.

Разработка дизайн-макета необходима для определения положений элементов страниц системы, и является начальным этапом перед проектированием пользовательского интерфейса.

Назначение такого документа, как руководство пользователя состоит в том, чтобы предоставить пользователям необходимую информацию о том, как правильно пользоваться системой, о ее компонентах, а также обеспечить пользователям возможность самостоятельно решать основные задачи при работе с ИС.

Исходя из полученных результатов тестирования после внедрения системы время, затраченное на совершения операций в ИС значительно меньше, чем результаты, полученные в результате прохождения аналогичных процессов без внедрения информационной системы.

ЗАКЛЮЧЕНИЕ

В данной работе выполнено проектирование и разработка системы управления подписками на основе REST-сервисов.

В такой компании как ООО «НетКрэкер» сбор, хранение и быстрый поиск актуальной и необходимой информации по всем категориям рабочих вопросов является немаловажным процессом.

Различные системы хранения информации, использующиеся в компании имеют материалы на похожие тематики. Однако, отсутствие взаимосвязи между ними способствует более долгому процессу поиска нужной информации в них, и означает, что в каждой из систем рассматриваемый процесс подписки на уведомления об интересующей информации должен происходить по отдельности, а не с единого интерфейса.

Анализ построенной модели бизнес-процесса подписки на уведомления от информационных ресурсов показал, что она имеет ряд недостатков, которые можно устранить, разработав и внедрив информационную систему, направленную на устранение ряда выявленных недостатков.

Разработка ИС с помощью современных технологий и с правильно выбранной архитектуры обеспечивает в дальнейшем возможности для простого расширения функционала систем, а также предоставляет возможность подключения к разработанной системе других информационных ресурсов компании.

Наличие удобного интерфейса управления подписками обеспечивает более высокую производительность труда, сокращая при этом затраты на время сбора и поиска нужной и актуальной информации из использующихся в компании систем.

Новая разработанная ИС выполняет все возложенные на нее задачи, отвечает всем поставленным к ней требованиям, что позволит значительно ускорить и упростить процесс подписки на интересующую информацию для каждого сотрудника компании.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Научная и методическая литература

1. Гери Д., Хорстманн К. JavaServer Faces. Библиотека профессионала, 3-е изд.: Пер. с англ. – М.: ООО «И. Д. Вильямс», 2011. – 544 с.: ил. – Парал. тит.англ.

2. Емельянова, Н.З. Проектирование информационных систем: учебное пособие / Н.З. Емельянова, Т.Л. Партыка, И.И. Попов. - М.: Форум, 2014. - 432 с.

3. Заботина, Н. Н. Проектирование информационных систем – М.: ДРОФА, 2013. – 336 с

4. Машнин Т. С. Технология Web-сервисов платформы Java. — БХВ-Петербург, 2012. — С. 115. — 560 с.:ил.

5. Монахов В. В. Язык программирования Java и среда NetBeans. – 3-тье изд., перераб. и доп. – СПб.: БХВ-Петербург, 2012. – 704 с.: ил.

6. Трофимов В.В. Информационные системы и технологии в экономике и управлении, 4-е изд. - М.:Юрайт-Издат,2013.-521 с.

7. Фримен Э., Фримен Э., Сьерра К., Бейтс Б.. Паттерны проектирования. – СПб.: Питер, 2011. – 656 с.: ил.

8. Чистов, Д. В. Проектирование информационных систем. Учебник и практикум / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук. – М.: Юрайт, 2016. – 260 с.

9. Эккель Б. Философия Java. Библиотека программиста.4-е изд. – СПб.: Питер, 2015. – 640 с.: ил.

Электронные ресурсы

10. Сайт Apache Maven Project. [Электронный ресурс]. – Режим доступа: http://maven.apache.org.

11. Сайт BPWin. [Электронный ресурс]. – Режим доступа: www.bpwin.ru.

12. Сайт dbForge Studio for Oracle. [Электронный ресурс]. – Режим доступа: http://www.devart.com/ru/dbforge/oracle/studio/download.html.

13. Сайт GlassFish server. [Электронный ресурс]. – Режим доступа: https://glassfish.java.net.

14. Сайт Java. [Электронный ресурс]. – Режим доступа: https://www.java.com/ru.

15. Сайт Netbeans IDE – Java EE and Web Application Development. [Электронный ресурс]. – Режим доступа: https://netbeans.org/features/java-on-server/index.html.

16. Сайт Oracle. [Электронный ресурс]. – Режим доступа: http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index.html.

Литература на иностранном языке

17. Roy Thomas Fielding. Architectural Styles and the Design of Network-based Software Architectures. – Dissertation,2000. – 180 p.

18. Joshua J. Bloch. Effective Java Programming Language Guide. - Addison-Wesley,2014. – 272p.

19. Savas Parastatidis, Jim Webber, Ian Robinson. REST in Practice.Hypermedia and Systems Architecture. - O'Reilly Media,2010. – 448 p.

20. Leonard Richardson , Mike Amundsen , Sam Ruby. RESTful Web APIs: Services for a Changing World. - O'Reilly Media,2013. – 406 p.

ПРИЛОЖЕНИЕ А

Руководство пользователя

Назначение системы

Данная информационная система предназначена для управления подписками пользователей по публикуемым материалам на внутренних информационных ресурсах компании.

ИС предоставляет пользователям возможность установки желаемого времени выполнения рассылок для каждого из пользователей, возможность установки фильтрации интересующего контента каждым пользователем, а также возможность выбора способа оповещения о новых или измененных материалах.

Вход в систему

Для начала работы с системой необходимо запустить Web-браузер и набрать в адресной строке следующий URL:

https://subscription-manager.netcracker.com

После чего осуществиться переход на стандартную страницу авторизации для подключения к системе, которая является одинаковой для всех ресурсов компании.

Categories

После входа в систему пользователю отобразиться страница с разделом категорий публикаций, которая представлена на рисунке А.1.

В данном разделе отображен список публикаций, который может быть раскрыт при нажатии на иконку стрелки слева от наименования публикации.

Для просмотра\скрытия краткой информации о публикации необходимо нажать кнопку «Show more», которая покажет\скроет панель с информацией, находящуюся ниже наименования публикации.

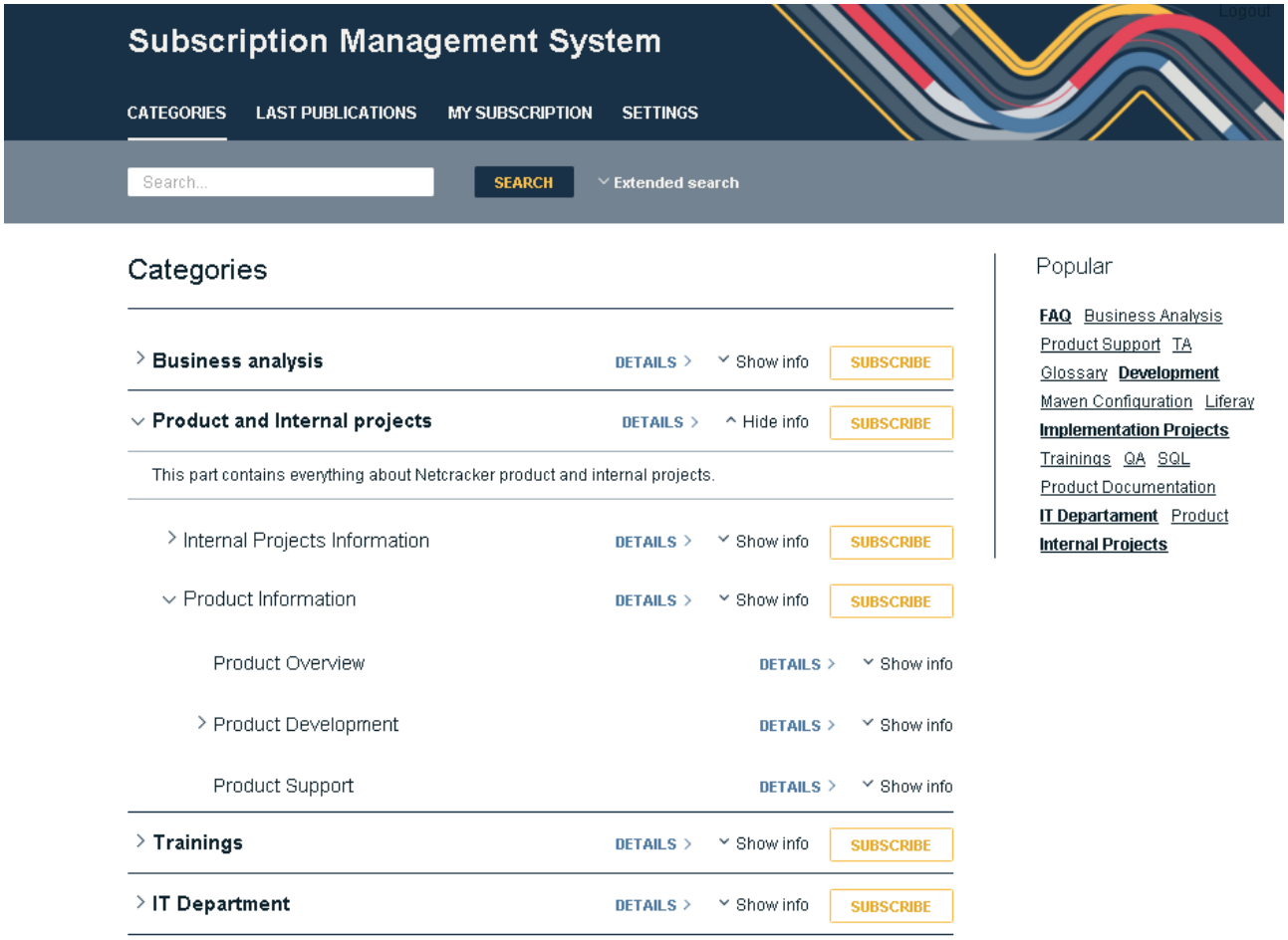

Рисунок А.1 – Раздел «Categories»

Для осуществления подписки на какую-либо публикацию необходимо нажать на кнопку «Subscribe».

Search

Форма поиска изначально имеет уменьшенный (нераскрытый) вид, и может быть раскрыта вниз при нажатии на кнопку «Extended search». В этом случае панель раскрывается и предоставляется возможность поиска по тегам и по авторам публикаций. Панель поиска представлена на рисунке А.2.

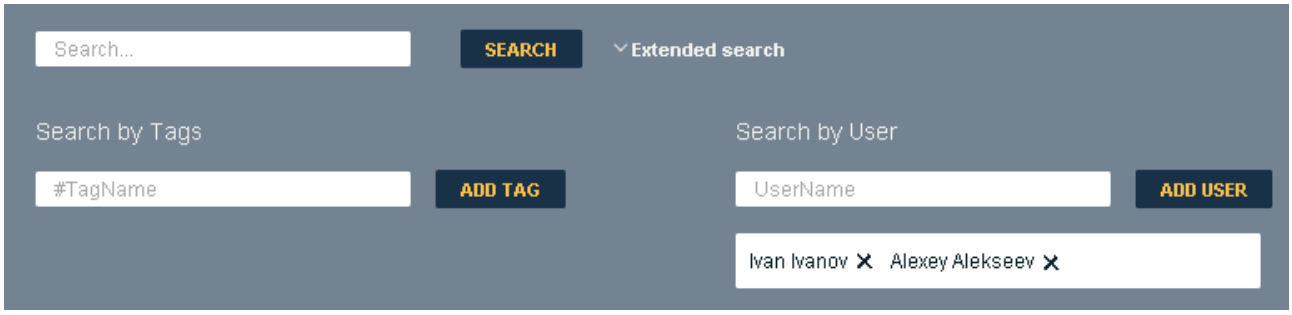

Рисунок А.2 – Панель поиска

Для поискам по тегам необходимо в соответствующее поля для поиска ввести название тега (система предоставит выпадающий список подходящих тегов) и, затем нажать на кнопку «Add tag». В появившейся ниже области отобразятся добавленные теги.

Для отправки запроса поиска на сервер необходимо нажать кнопку «Search».

Для удаления выбранного тега необходимо нажать на иконку крестика рядом с тегом в области выбранных тегов.

Поиск по авторам публикаций аналогичен поиску по тегам.

Last publications

В данном разделе информационной системы представлен список последних пятнадцати публикаций, размещенных на внутренних ресурсах компании.

Список последних публикаций представлен отдельными блоками, содержащими информацию по каждой публикации:

- дата размещения;
- автор;
- название;
- описание;
- список тегов;
- кнопку «Subscribe» для осуществление подписки на этот раздел.

Интерфейс списка публикаций представлен на рисунке А.3.

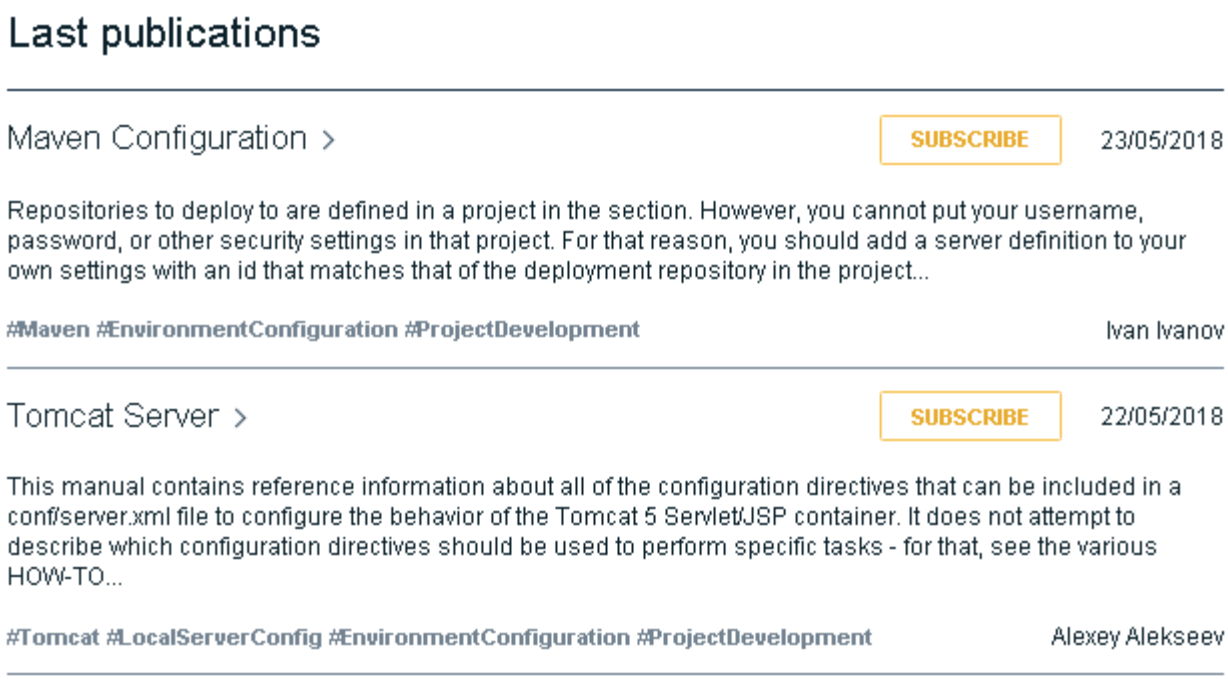

Рисунок А.3 – Раздел «Last publications»

Для осуществления подписки на рассылку уведомлений о какой-либо публикации из данного раздела необходимо нажать на кнопку «Subscribe».

My subscription

Данный раздел системы представляет все публикации на рассылку которых подписан пользователь. Окно системы при открытом данном разделе показано на рисунке А.4.

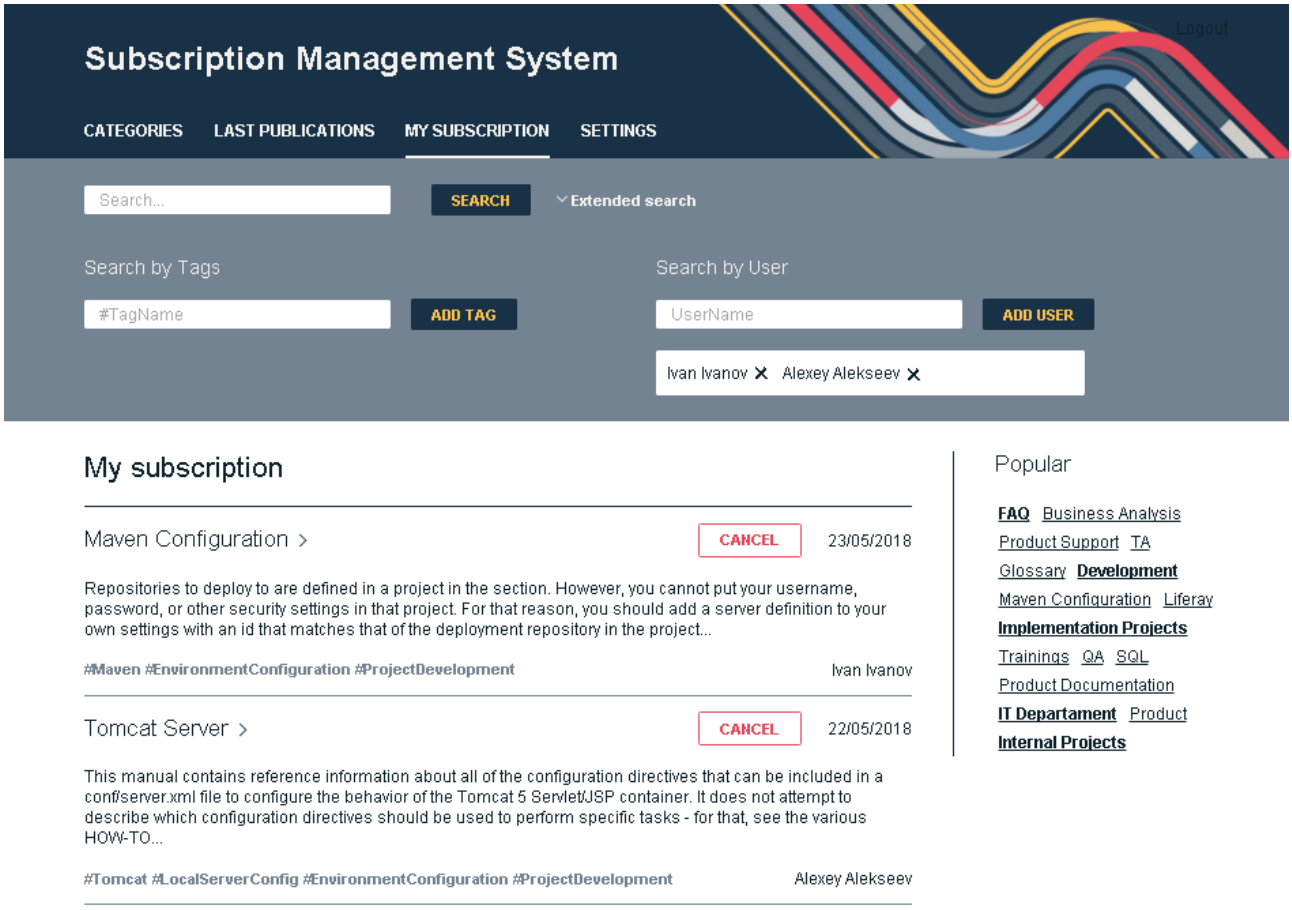

Рисунок А.4 – Раздел «My subscription»

Для осуществления отписки от рассылки на не интересующую более публикацию необходимо нажать на кнопку «Cancel» в строке с наименованием нужной рассылки.

Settings

Интерфейс пользователя с отображенной панелью настроек представлен на рисунке А.5.

Доступные настройки в данном разделе:

- тип оповещения;
- время оповещения;
- фильтрация публикаций по тегам.

Каждый пользователь информационной системы имеет возможность выбрать подходящий ему тип оповещения: отправку списка публикаций, на которые он подписан, на электронную почту пользователя, либо «push»-

оповещения, которые будут появляться в качестве всплывающего окна в браузере пользователя.

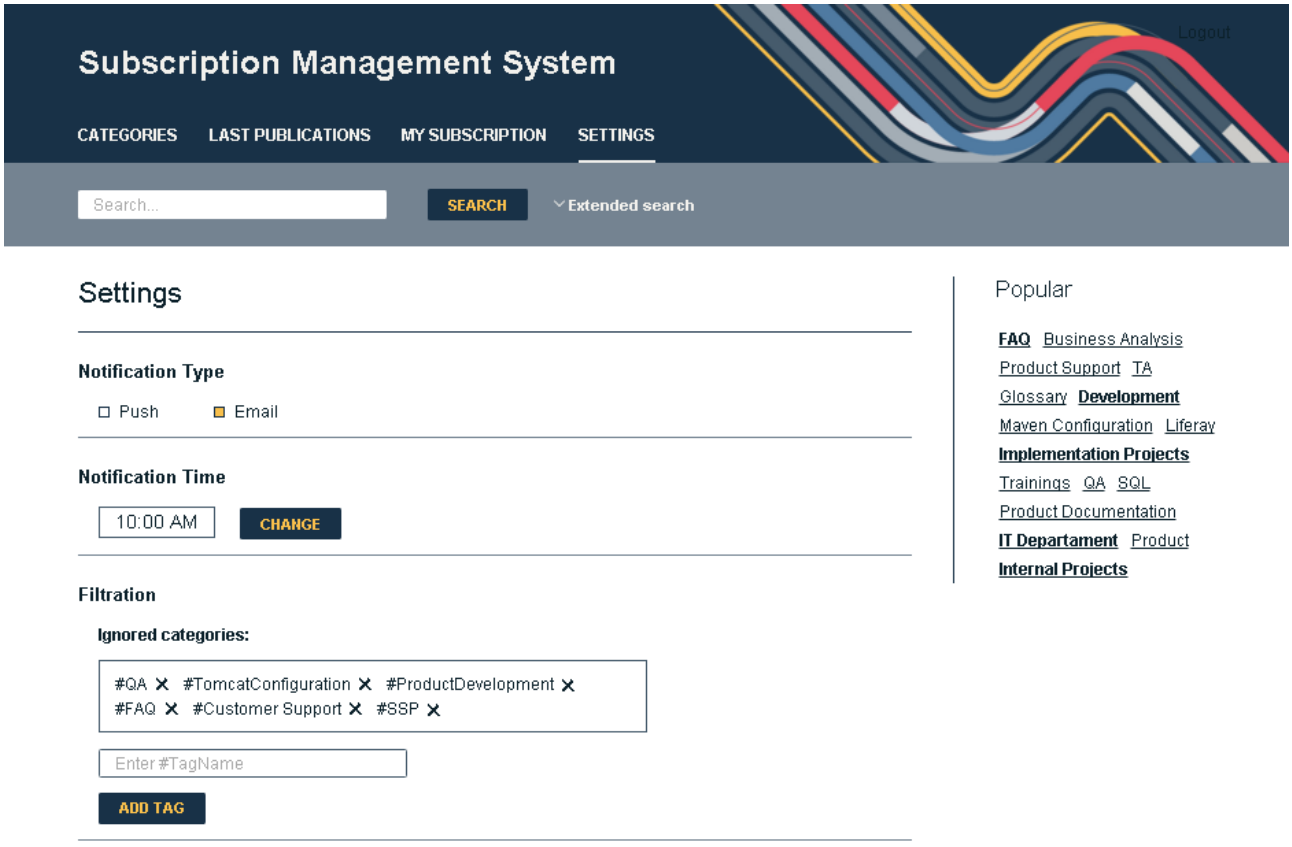

Рисунок А.5 – Раздел «Settings»

Для установки желаемого времени оповещений необходимо нажать на кнопку «Change», в результате чего появиться всплывающее окно с панелью выбора времени.

Ограничить отображение не интересующих категорий публикаций в разделе последних публикаций можно с помощью выбора определенных тегов в разделе «Filtration». Для удаления элемента из списка необходимо нажать на иконку крестика возле нужной категории.