

The Student Attendance

Software Requirements Specification

Version 1.0

20.10.2015

1. Введение

1.1 Назначение

Данный документ описывает требования к системе автоматической проверки посещаемости *The Student Attendance*, разрабатываемой студентами в рамках курса «Анализ требований к программному обеспечению». Настоящий документ предназначен для участвующих студентов в разработке указанной системы и заказчика.

1.2 Область действия

Система проверки посещаемости *The Student Attendance* должна представлять собой распределенную систему, обеспечивающую проверку посещаемости студентов в ВУЗ-ах.

Задачами системы являются:

1. Возможность считывать барт-код с разных источников.
2. Возможность точной проверки присутствия студента на паре.
3. Поддержка уже существующих систем БД студентов, преподавателей и занятий путем предоставления API.

При этом система не должна быть сильно зависимой от определенного ВУЗ-а или страны, а также от наличия определенных устройств у студентов.

Система *The Student Attendance* будет применяться для упрощения процесса переключки в ВУЗах. Переход от ручного внесения изменений в журнал к применению разрабатываемой системы позволит ускорить процесс переключки, а также на порядок снизить количество ошибок, допускаемых кассирами. Также система позволит вручную редактировать журналы, в случае возникновения ошибок в работе системы.

1.3 Список определений, сокращений и аббревиатур

JPA – Java Persistence API

СУБД — Система управления базами данных

СУРБД — Система управления реляционными базами данных

API – интерфейс программирования приложений

MVC – набор паттернов Model-View-Controller

TSA – сокращенное название проекта «The Student Attendance»

1.4 Ссылки

1. <http://www.oracle.com/technetwork/java/javasee/tech/persistence-jsp-140049.html>
2. <https://ru.wikipedia.org/wiki/Model-View-Controller>

1.5 Структура документа

Глава 2 содержит описание системы TSA общими словами без деталей. Она предназначена для ознакомления с проектируемой системой.

В главе 3 содержится детализированное описание функциональности TSA.

2. Общее описание

2.1 Перспективы продукта

Разрабатываемый программный продукт не является частью более крупной программной системы. Система TSA должна содержать серверную часть и клиентскую часть. При этом необходимо разработать три типа клиентов – для доступа с Android-совместимого устройства, для доступа с браузера, и для доступа с других приложений(собственный API).

Среди аналогов данного решения найдутся совершенно разные системы со своими преимуществами и недостатками, но таких систем обнаруженных не было.

2.2 Функции продукта

Приведем список основных функций разрабатываемой системы.

2.2.1. Хранение информации о студентах, их занятиях, и преподавателях

Должна быть создана база данных, хранящая информацию о студентах, группах, потоках, их занятиях, и их преподавателей.

2.2.2. Редактирование информации в базе данных из решения

Администратор системы должен иметь возможность с помощью приложения редактировать имеющуюся и добавлять новую информацию в базу данных:

- добавлять новых студентов, изменять существующих;
- изменять журнал посещаемости;
- изменять содержимое групп, потоков;
-

2.2.3. Считывание барт-кода

Преподаватели должны иметь возможность считывания барт-кодов с камеры Android-совместимого устройства в реальном времени.

2.2.4. Отправка данных на обработку сервером

Клиентская программа должна уметь отправлять данные о студентах и их посещенных парах на обработку и хранение сервером, для формирования журнала в дальнейшем.

2.3 Характеристика пользователей

Предполагаются, что разрабатываемой системой будут пользоваться две категории пользователей.

- администратор – обладает опытом работы со сторонними API, и интеграцией уже существующих приложений с ними;
- преподаватель – преподаватель в ВУЗе, который будет обладать устройством для считывания барт-кодов, и будет отправлять данные а сервер, при появлении первой же возможности сделать это.

2.4 Ограничения

Ниже перечислены ограничения на технологии, используемые при разработке системы.

2.4.1. Использование языка Java

Данное ограничение вызвано требованием работоспособности сервера под управлением различных операционных систем. Также Java очень известная технология, с очень большим комьюнити, что означает наличие необходимых ресурсов для работы.

2.4.2. Параллельная обработка запросов, поступающих от клиентов, в серверном приложении

Данное ограничение вызвано необходимостью быстрой обработки запросов сервером и низкого времени отклика.

2.5 Предположения и зависимости

Предположения:

- Приложение будет востребовано преподавателями ВУЗов
- Приложение будет генерировать большой трафик
- В ВУЗах уже существуют какие-то электронные СУБД

Ограничения:

- Скорость отклика и обновления зависят от мощности и устойчивости сервера, который принимает и обрабатывает данные, а также от скорости интернет-подключения

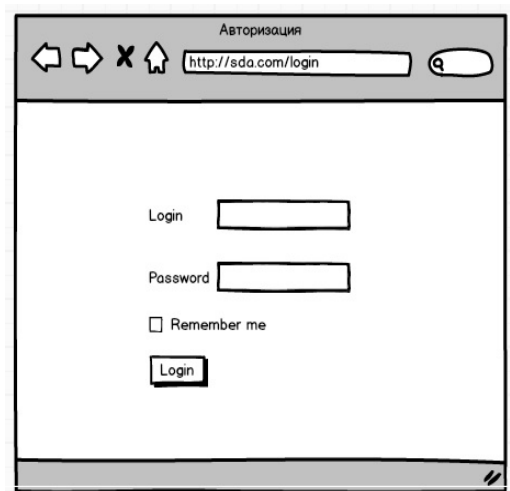
3 Специфические требования

3.1 Требования к Внешним интерфейсам

3.1.1 Пользовательские интерфейсы

3.1.1.1 Веб-часть

Окно авторизации:



Авторизация

http://sda.com/login

Login

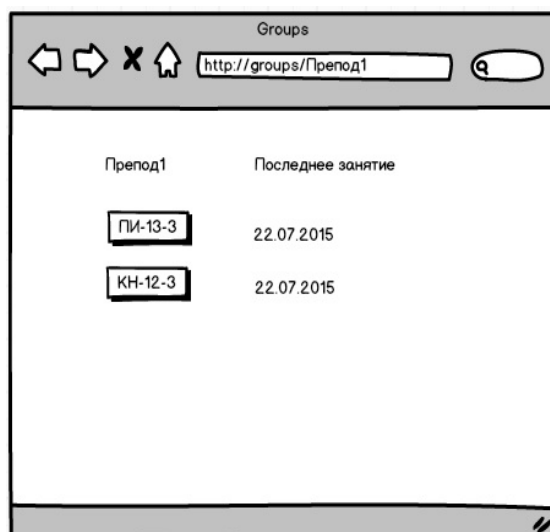
Password

☐ Remember me

Login

На него пользователь попадает сразу при обращении на веб-сайт. В данном окне находятся 2 поля для ввода логина и пароля соответственно. Нужен для входа в систему, идентификации пользователя и определения его роли в системе.

Окно списка групп преподавателя



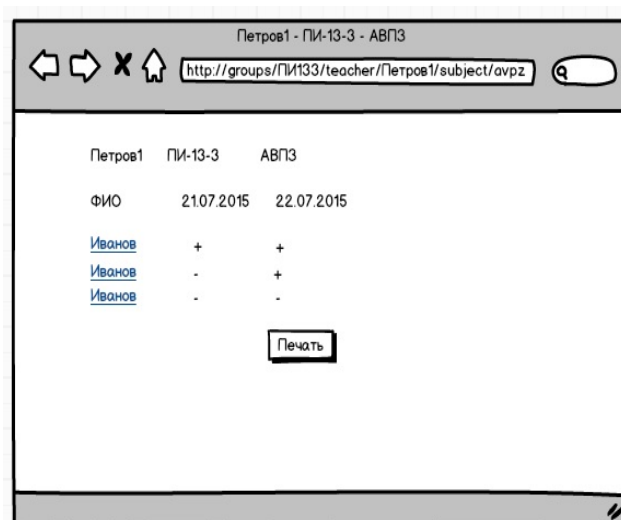
Groups

http://groups/Препо1

Препо1	Последнее занятие
ПИ-13-3	22.07.2015
КН-12-3	22.07.2015

На него пользователь попадает после того, как назовет группу в главном меню. Здесь можно посмотреть список групп, у определенного преподавателя, и дату последнего занятия. Также перейти на определенную группу.

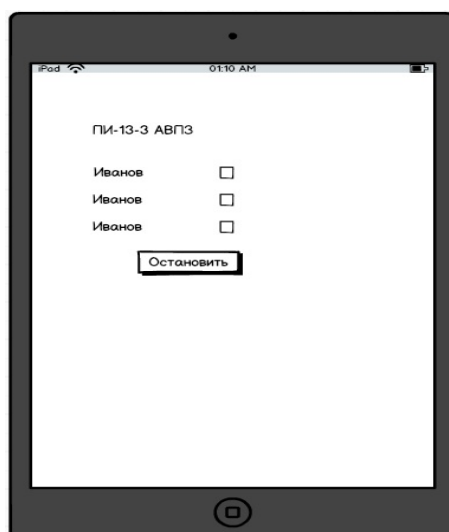
Окно группы определенного преподавателя определенной группы



Окно группы со списком студентов и их посещения в определенные дни. Нужен для компактного представления. Формируется из отправленных данных с устройства преподавателя. Также можно редактировать это, и вывести напечатать.

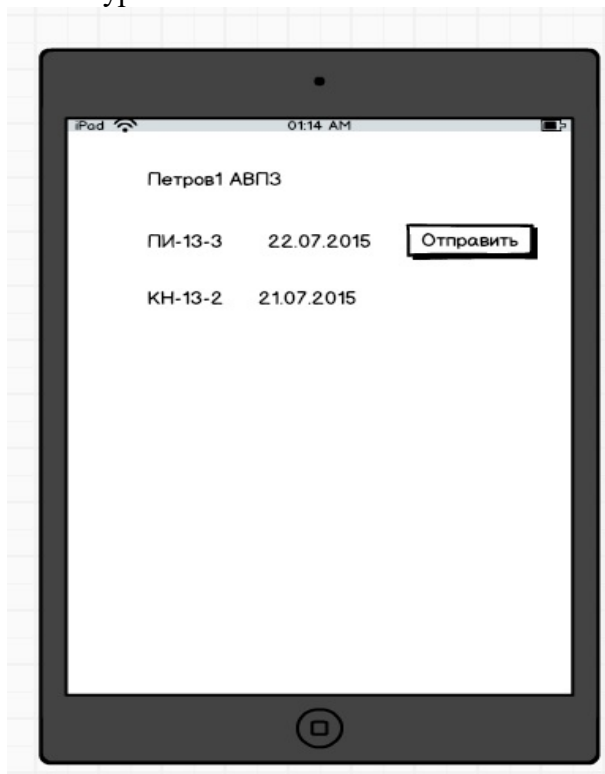
3.1.1.2 Андроид часть

Окно журнала



Окно во время считывание барт-кодов студентов во время работы приложения. Тут можно посмотреть тех студентов, которые отметились, и оостановить прием. Также, при необходимости можно убрать галочку о присутствии определенного студента.

Окно просмотра списков журналов



Здесь можно посмотреть список групп, и дат их последних занятий. Также можно отправить журналы на обработку сервером.

3.2 Функциональные требования

3.2.1 Получение и сохранение барт-кода пользователей

3.2.1.1 Приложение будет получать и сохранять барт-код пользователей, для идентификации пользователя и отправки данных на сервер

3.2.1.2 Вводными данными будут барт-коды пользователей

3.2.1.3 В качестве обработчика будет использована сторонняя библиотека

3.2.1.4 Результатом будет информация, которая запишется во встраиваемую БД, для дальнейшей работы с ней.

3.2.1.5 При подстановке непонятно чего к камере ничего не будет происходить, или если будет ошибка при считывании барт-кода, будет воспроизведен соответствующий звук.

3.2.2 Синхронизация барт-кода с системой

3.2.2.1 Отправка барт-кода на сервер, получение данных о студенте с данным барт-кодом с сервера в качестве ответа.

3.2.2.2 Считанные ранее барт-коды

3.2.2.3 Сохранение сервером информации, и идентификации барт-кода

3.2.2.4 Результатом из сервера придет список студентов, и данные поступят на обработку, и сохранятся

3.2.2.5 В качестве ошибок из сервера будут приходиться соответствующие данные, помимо правильно сохраненных данных о существующих студентах

3.2.3 Формирование журнала

3.2.3.1 Формирование журнала студентов, из имеющихся данных

3.2.3.2 Сервер получит данные из устройства телефона

3.2.3.3 Сервер начнет полученные обрабатывать данные, идентифицировать пользователя для создания журнала

3.2.3.4 В результате сервер может выдать журнал посещаемости

3.2.3.5 В случае ошибок будет выдана определенная информация

3.2.3 Возможность вносить ручные правки в журнал посещаемости

3.2.3.1 Возможность вносить ручные правки в журнал посещаемости, для получения более корректного представления о посещаемости студентов, также может понадобиться в случае возникновения ошибок

3.2.3.2 Сервер будет иметь уже существующие данные из уже сформированного журнала, и получит обновленные данные от преподавателя

3.2.3.3 Сервер просто сохранит данные в системе

3.2.3.4 Обновленные данные журнала

3.2.3.5 В принципе ошибок быть не может, но в случае возникновения ошибки пользователь будет переадресован на страницу с информацией об ошибке

3.2.4 Регистрация пользователей вручную

3.2.4.1 Позволяем регистрироваться пользователям вручную, в случае невозможности автоматической регистрации

3.2.4.2 Данные, введенные студентами

3.2.4.3 Валидация данных, и попытка сохранить данные

3.2.4.4 Новая учетная запись студента, для его идентификации

3.2.4.5 Будет показано, в каких полях возникла ошибка

3.2.5 Возможность объединять группы в потоки

3.2.5.1 Возможность объединения студентов в группы

3.2.5.2 Уже созданные группы, и их список, выбранный пользователем

3.2.5.3 Сервер попытается создать новый поток, и добавить в него группы

3.2.5.4 Новый поток, с которым можно будет работать

3.2.5.5 В случае ошибки, будет выведена соответствующая информация пользователю

3.2.6 Считывание данных с файлов определенного формата

3.2.6.1 Получение данных с текстовых файлов, например группы, студентов, потоков, предметов и преподавателей, в случае невозможности других способов

3.2.6.2 Файл пользователя в определенном формате

3.2.6.3 Система попытается обработать файлы, и сохранить всю информацию

3.2.6.4 Будут добавлены данные в систему для дальнейшей работы

3.2.6.5 В случае ошибки будет выведена соответствующая информация, и все данные, которые были правильными тоже не добавятся.

3.2.7 Вывод на печать журнала посещаемости

3.2.7.1 Возможность вывода в определенные форматы для вывода на печать

3.2.7.2 Журналы посещаемости из системы

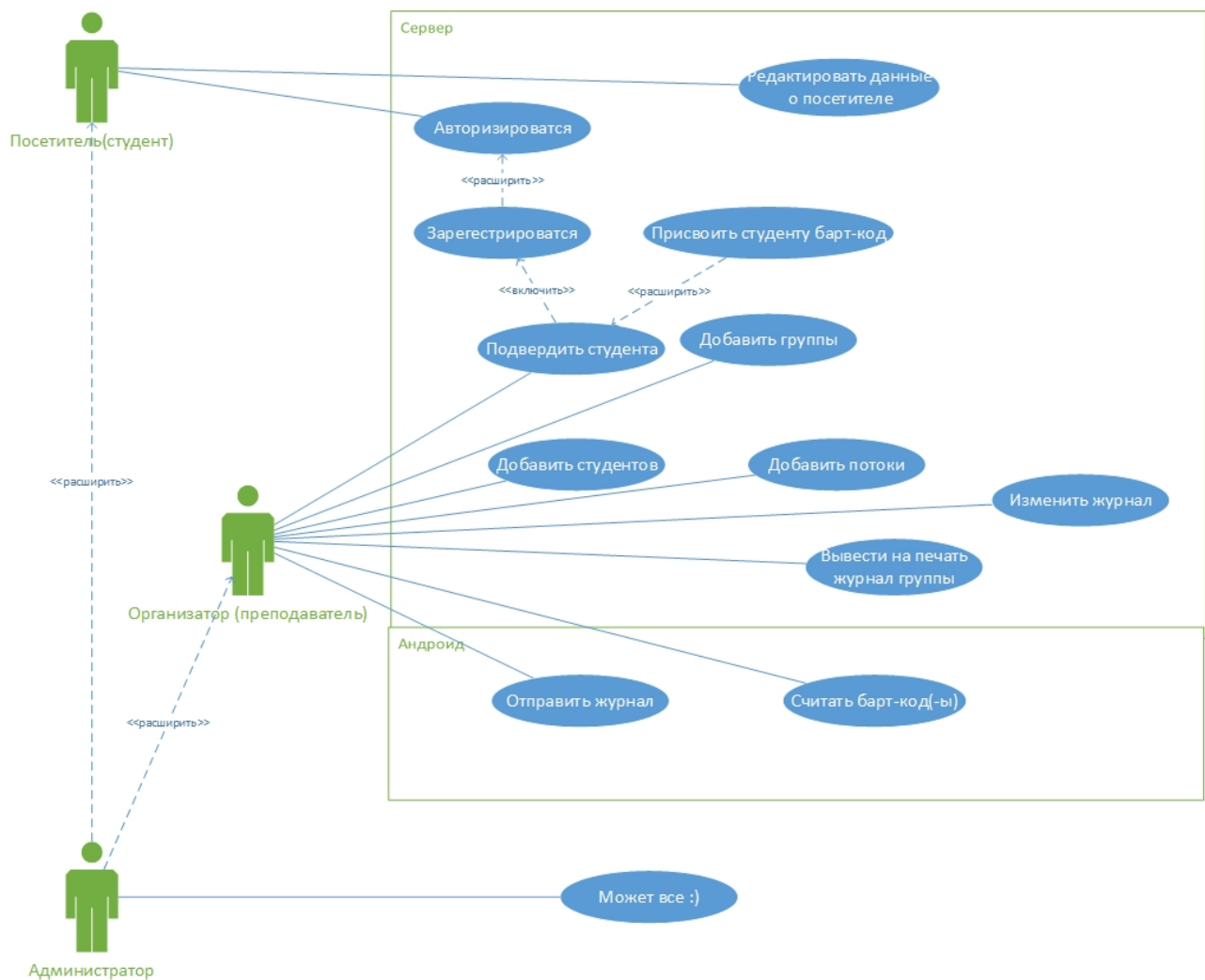
3.2.7.3 Обработать журнал, и создать определенный документ для вывода на печать

3.2.7.4 Документ для печати

3.2.7.5 Переадресация на соответствующую страницу с ошибкой

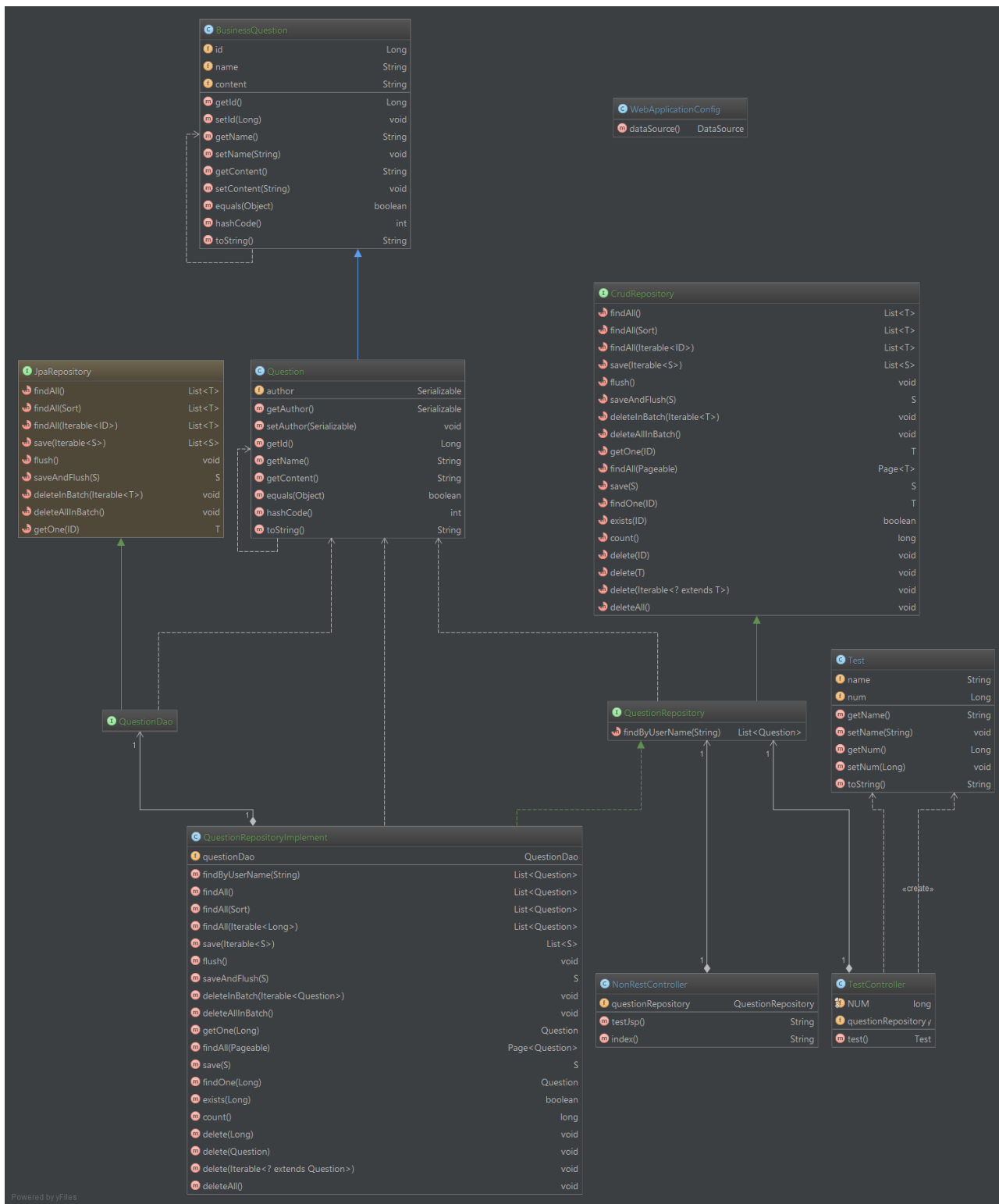
3.3 Use-case

Далее представлена диаграмма вариантов использования для TSA



3.4 Class diagrams

Далее представлена диаграмма классов проекта. На ней можно увидеть, и выделить такие классы, как BusinessQuestion, Question, WebAppConfig, QuestionDao, QuestionDaoImpl, QuestionRepository, CrudRepository, NonRestController, RestController.



3.4.1.1.2 Содержит конфигурацию приложений, в виде аннотаций

- 3.4.1.4 QuestionDao
 - 3.4.1.1.1 -
 - 3.4.1.1.2 Предоставление необходимых низкоуровневых методов для работы с данными
- 3.4.1.5 QuestionDaoImpl
 - 3.4.1.1.1 Настроенный источник данных
 - 3.4.1.1.2 Предоставление реализации необходимых низкоуровневых методов для работы с данными; реализация в ручную не пишется
- 3.4.1.6 CrudRepository
 - 3.4.1.1.1 -
 - 3.4.1.1.2 Предоставляет базу для высокоуровневых методов работы с данными
- 3.4.1.7 QuestionRepository
 - 3.4.1.1.1 -
 - 3.4.1.1.2 Предоставляет высокоуровневые операции для работы с определенными сущностями, настраивает транзакции, настраивает безопасность выполнения операций
- 3.4.1.8 QuestionRepositoryImpl
 - 3.4.1.1.1 QuestionDao
 - 3.4.1.1.2 Предоставляет реализации для высокоуровневых операций для работы с определенными сущностями
- 3.4.1.9 NonRestController
 - 3.4.1.1.1 QuestionRepository
 - 3.4.1.1.2 Управляет запросами пользователей, и предоставляет необходимые данные при наличии прав доступа; также изменяет информацию, при правильных данных
- 3.4.1.10 RestController
 - 3.4.1.1.1 QuestionRepository
 - 3.4.1.1.2 Управляет веб-запросами пользователей, и предоставляет необходимые данные при наличии прав доступа; также изменяет информацию, при правильных данных

3.5 Не функциональные требования

3.5.1 Производительность

Система должна будет работать с очень большим количеством пользователей, и даже не смотря на низкую нагрузку со стороны пользователей, нужно будет очень большая производительность, а соответственно и масштабируемость, расширяемость, настраиваемость. Система должна выдерживать минимум 10000 активных пользователей, что примерно 5000 запросов в секунду. Также на выполнение простых операций должно уходить не более 2х секунд со стороны сервера.

3.5.2 Надежность

Система должна будет иметь высокую работоспособность, и отказоустойчивость, даже при большом количестве пользователей. На начальных этапах эксплуатации системы, это может быть проблемным, но после расширений это требование должно соблюдаться, даже при очень высоких нагрузках. Java-системы должны всегда обеспечивать высокий уровень надежности.

Также, с другой стороны, должна быть выведена всегда корректная информация, и пользователь должен получать только правильные данные.

3.5.3 Доступность

Система должна быть доступна в сети интернет в любое время. Это обусловлено тем, что ВУЗы могут быть расположены в любых часовых поясах.

3.5.4 Защищённость

Система не должна позволять выполнять операции неавторизованным пользователям. Безопасность должна быть как минимум на 2х главных слоях: слое контроллеров и сервисов,

и слое репозитория.

3.5.5 Сопровождаемость

Требования к данному пункту не сильно нужны, т.к. система будет работать под нашим управлением, и это в разы уменьшает эти требования, т. к. мы сами сможем осведомлять наших специалистов о нашем решении, процессах внутри него и подобных действиях.

3.5.6 Портруемость

Система должна запускаться на операционных системах семейств Windows последних версий (выпуск 2008 и выше), а также unix-системах. Т.к. Наша система работает под управлением Java, то необходимо всего лишь наличие поддержки Java версий 8 и выше.

3.5.7 Требования СУРБД

Систему рекомендуется использовать с СУРБД типа Oracle версий 11g и выше, т.к. она обеспечивает лучшую производительность, безопасность, отказоустойчивость и подобные функции. Также она разрабатывается корпорацией Oracle, которая будет поддерживать свой продукт очень долго.

На начальных стадиях рекомендуется использовать MySQL — она обеспечит скорость анписания, и простоту. А также она поддерживается корпорацией Oracle.

Appendix A – Glossary

Сервис - означает некоторый вид контроллеров, которые предназначены для работы с другими системами, например существующими системами и т.п., сам по себе он возвращает объекты, в качестве результатов, и переводит их в запрашиваемый формат, например xml, json, etc.

Контроллер — объект, принимающий данные от пользователей из браузера, в качестве результата возвращает тоже список объектов, но дополнительно ещё и адрес представления.

Репозиторий — надстройка над слоем доступа к данным, которая выносятся для обеспечения безопасности, и транзакций, для возможности удобной их настройки.