1 Типы пользователей

1. Анонимный пользователь - незарегистрированный пользователь, который может только просматривать рейтинг.

2. Администратор группы - человек, вносящий данные об успеваемости студентов в систему (например, староста группы).

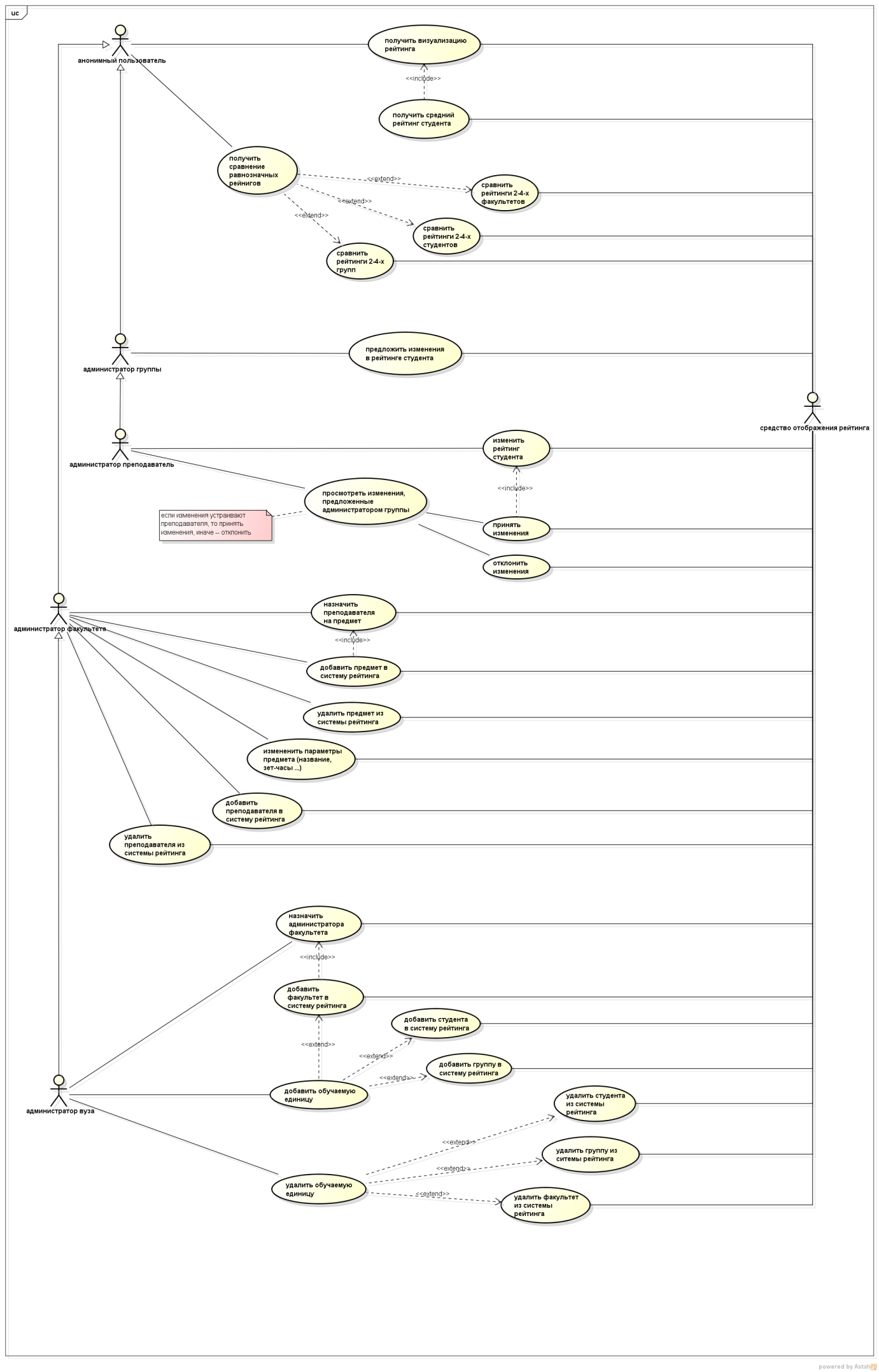
3. Администратор-преподаватель - человек, проверяющий работу администратора группы. Он может принять или отклонить введенные им данные. Также он может внести некоторые коррективы в данные, внесенные администратором группы. У каждого преподавателя по своему предмету(-ам) есть таблица норм баллов с учетом срезов, которые он может заполнять и редактировать.

4. Администратор факультета – человек, который назначает преподавателей на каждый предмет, а так же администраторов групп.

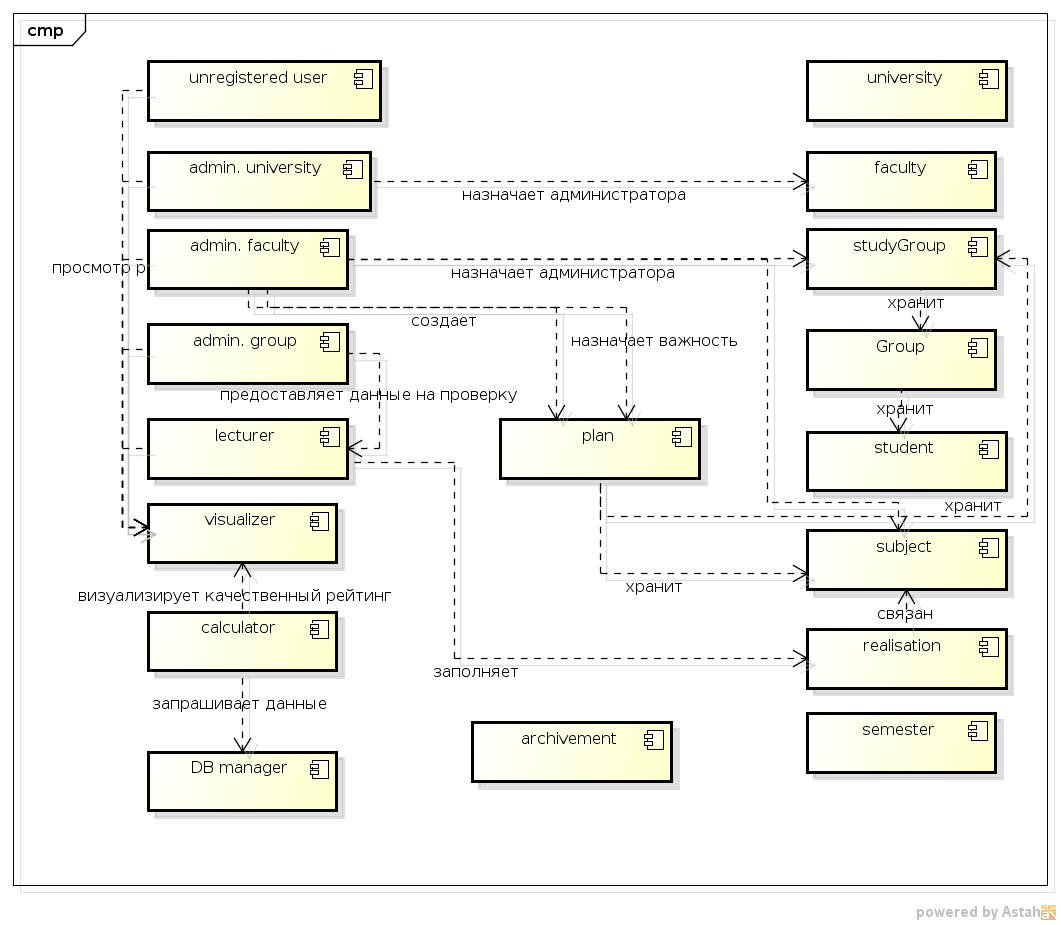
5. Администратор БД – человек, который занимается редактированием и заполнением базы данных. Он вносит новые предметы, студентов и преподавателей в базу каждый семестр. Также он назначает администраторов факультетов.

2 Основные диаграммы

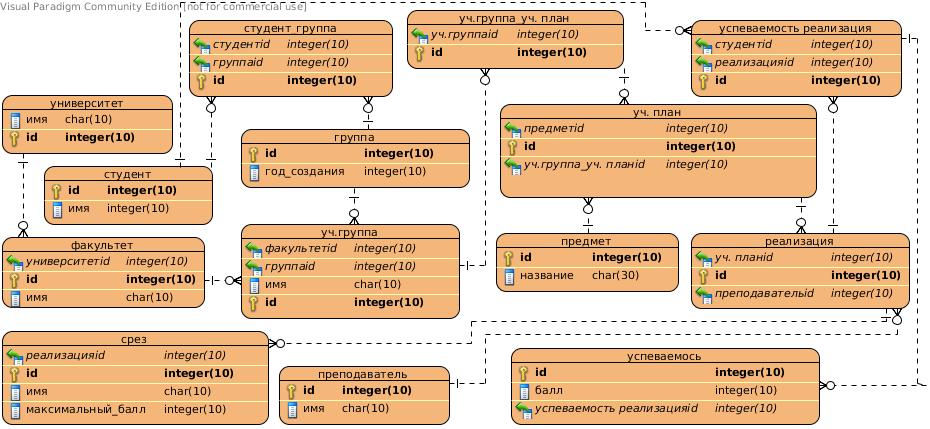
2.1 Диаграмма вариантов использования



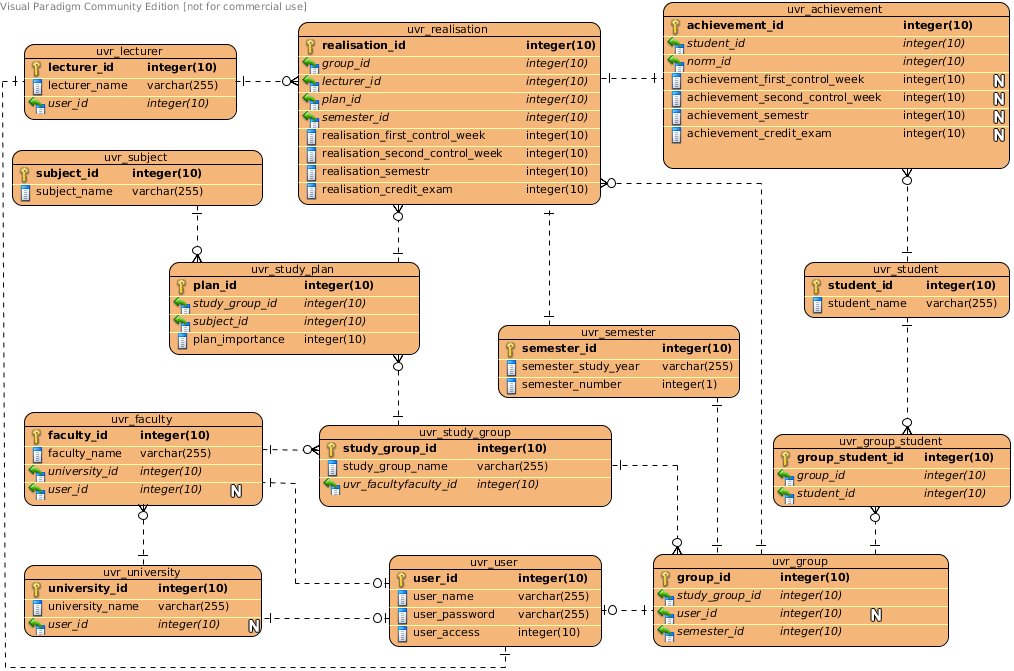
2.2 Объектная модель



2.3 Логическая структура базы данных для работы URV

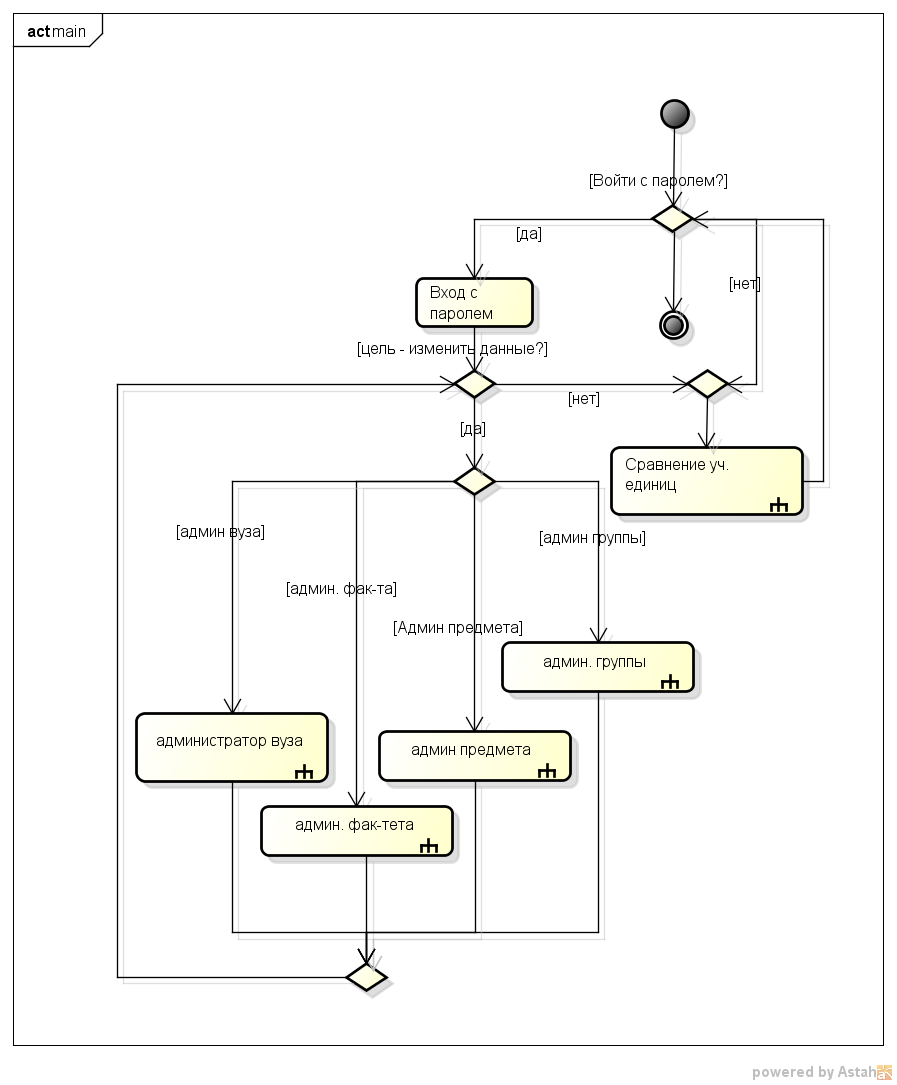


2.4 Диаграмма базы данных (физический уровень)

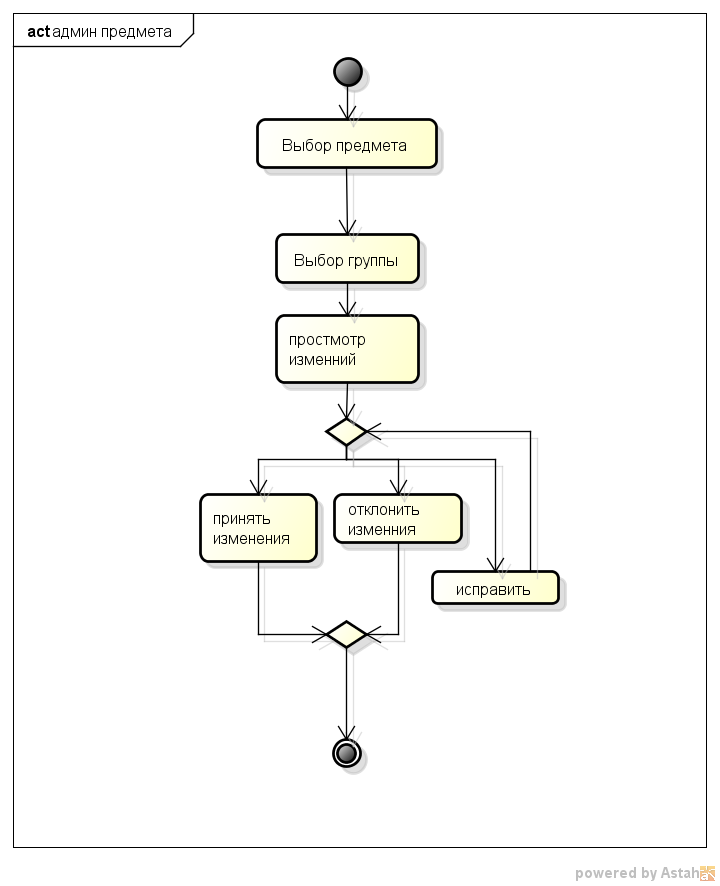


* 1. Диаграммы активности основных процессов в системе

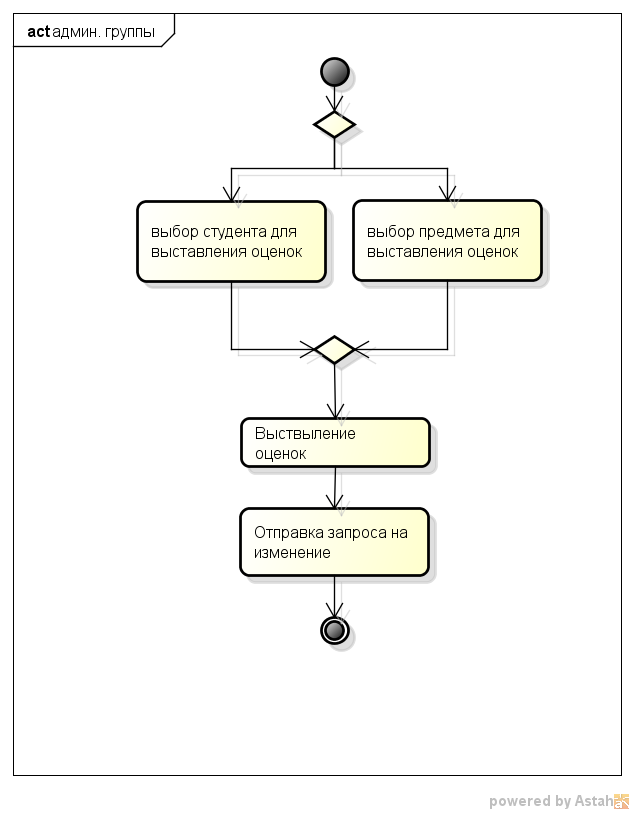
2.5.1 Главный цикл программы



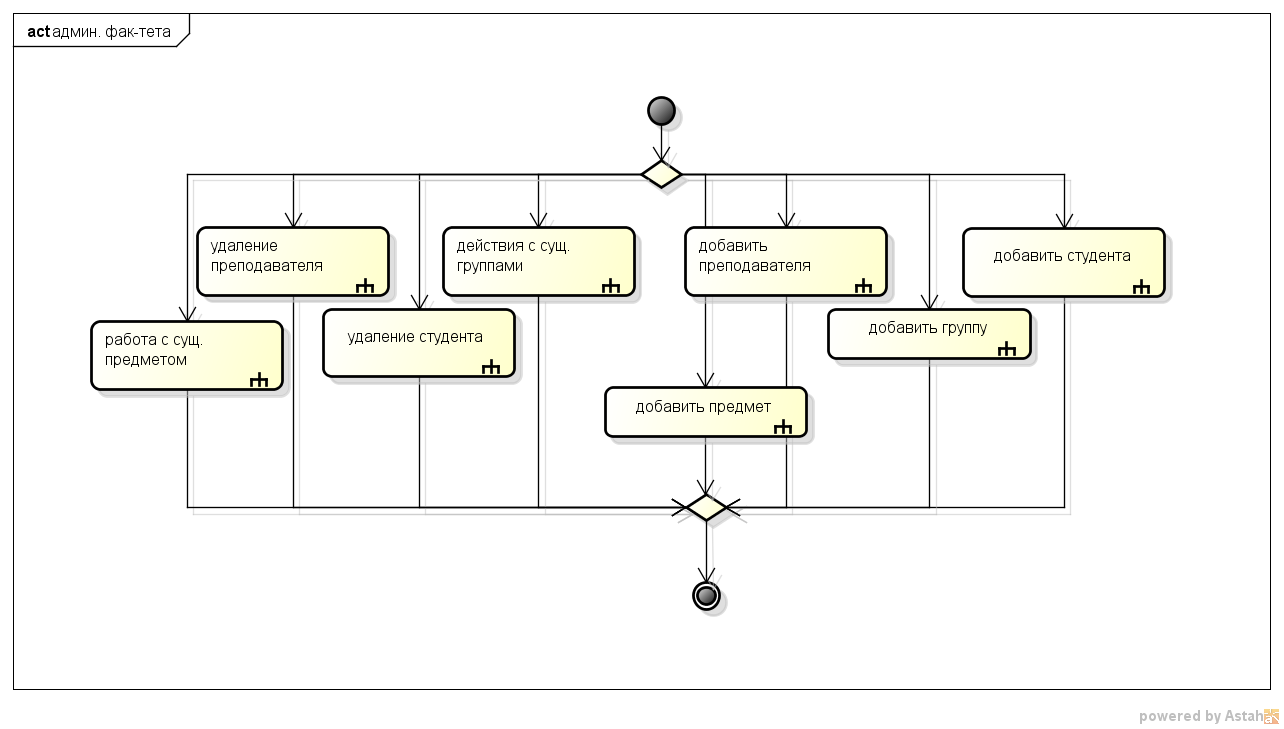
* + 1. Основные действия администратора предмета (преподавателя)



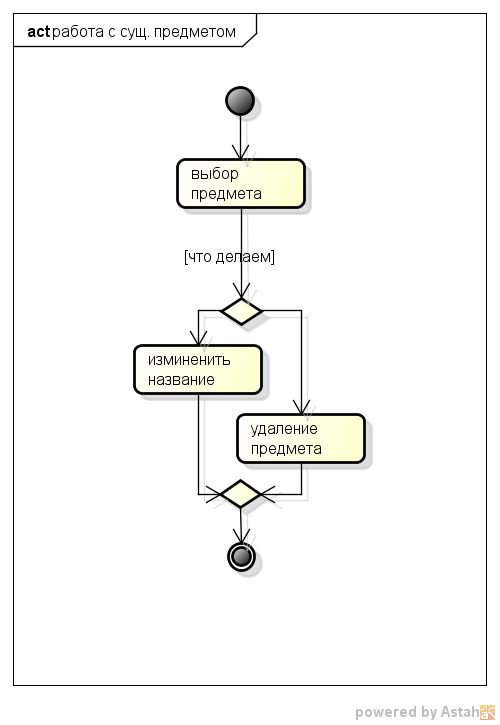
* + 1. Основные действия администратора группы



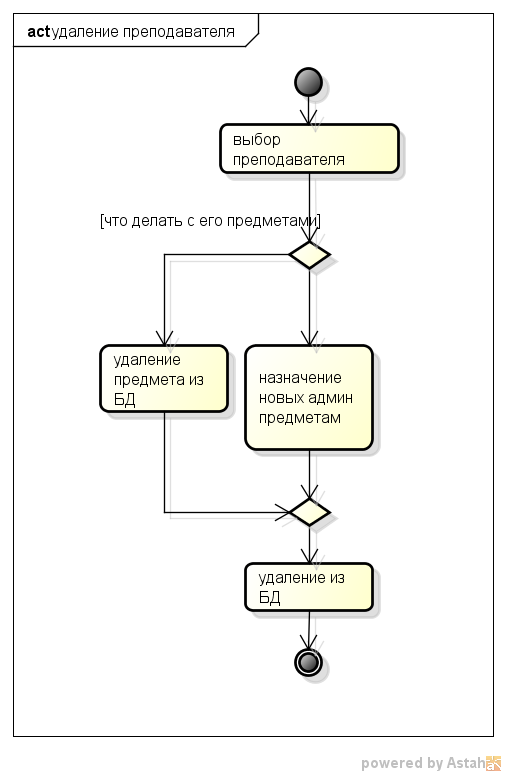
* + 1. Основные действия администратора факультета



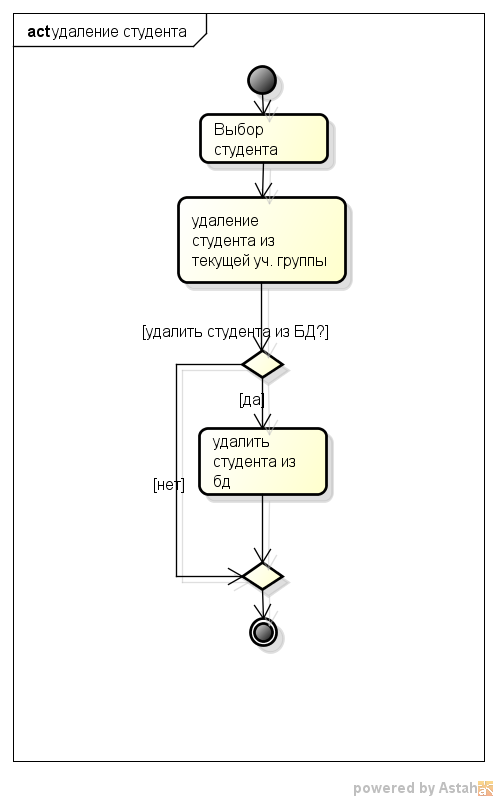
* + 1. Работа с существующим предметом



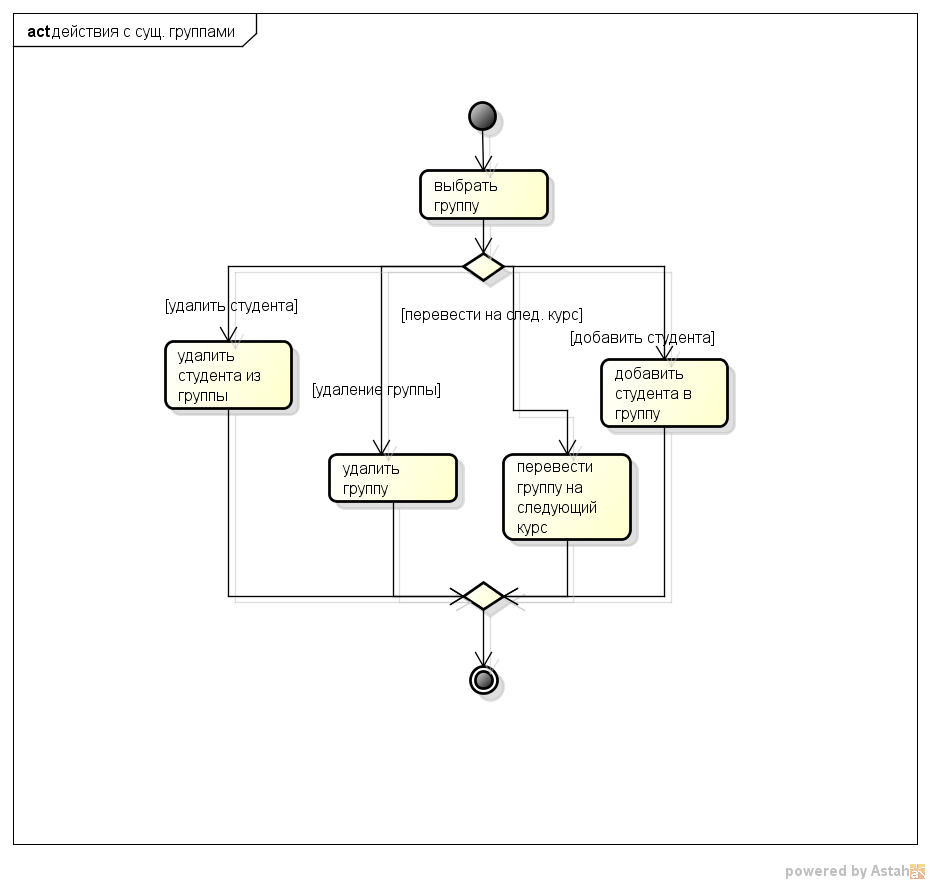
* + 1. Удаление преподавателя



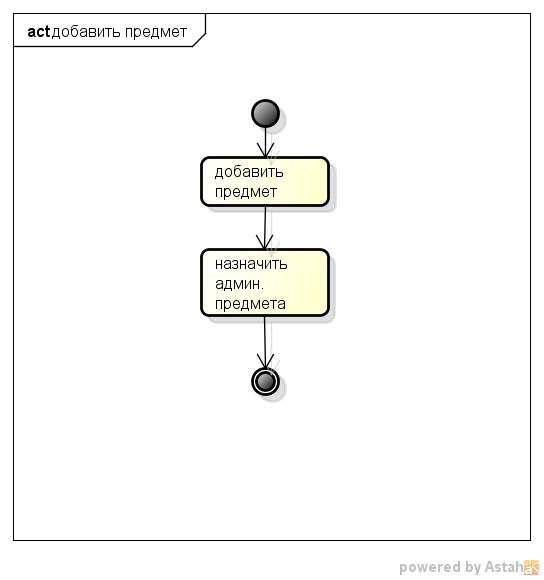
* + 1. Удаление студента



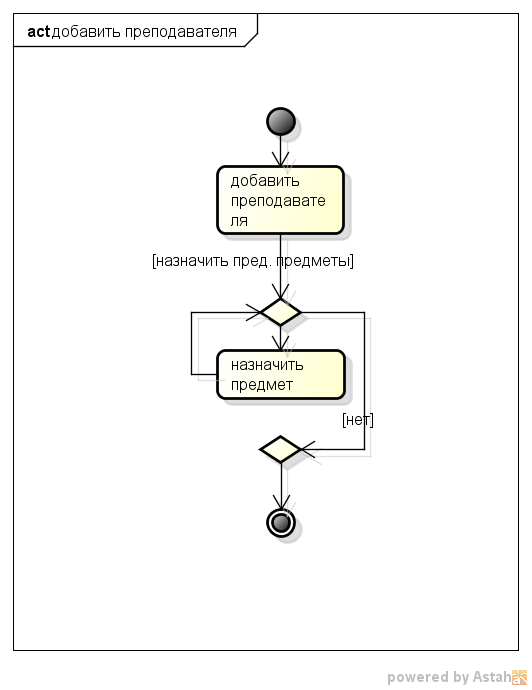
* + 1. Действия с существующими группами



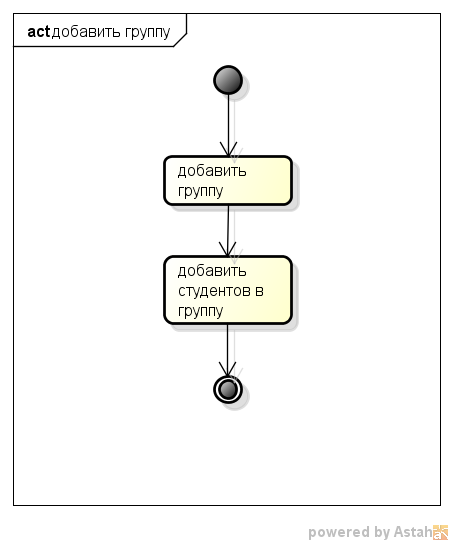
* + 1. Добавить предмет



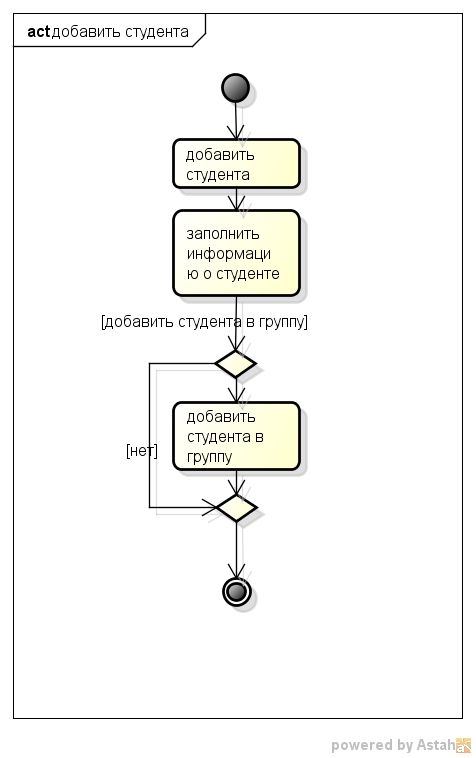
* + 1. Добавить преподавателя



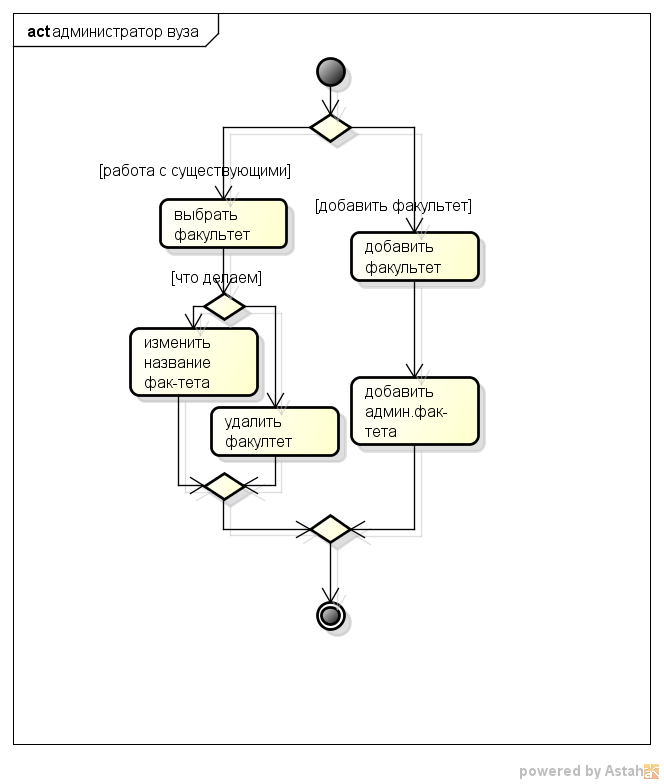
* + 1. Добавить группу



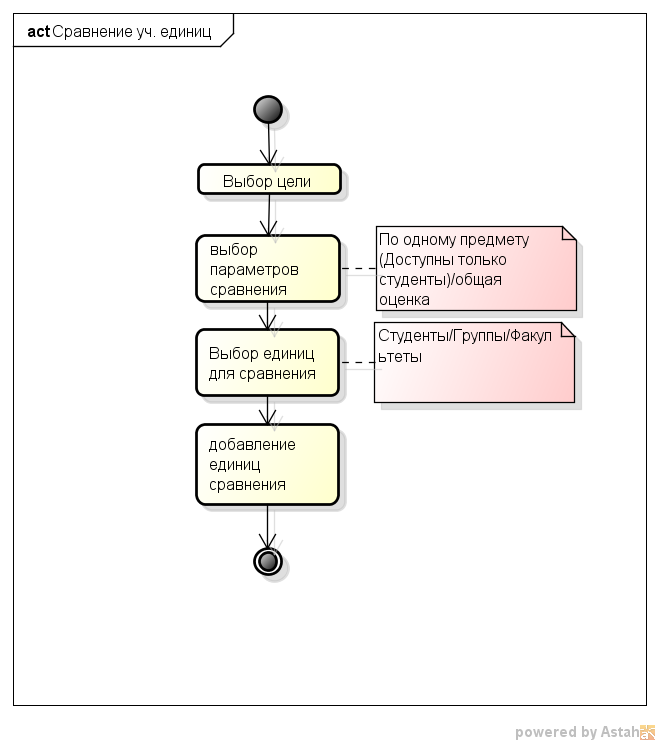
* + 1. Добавить студента



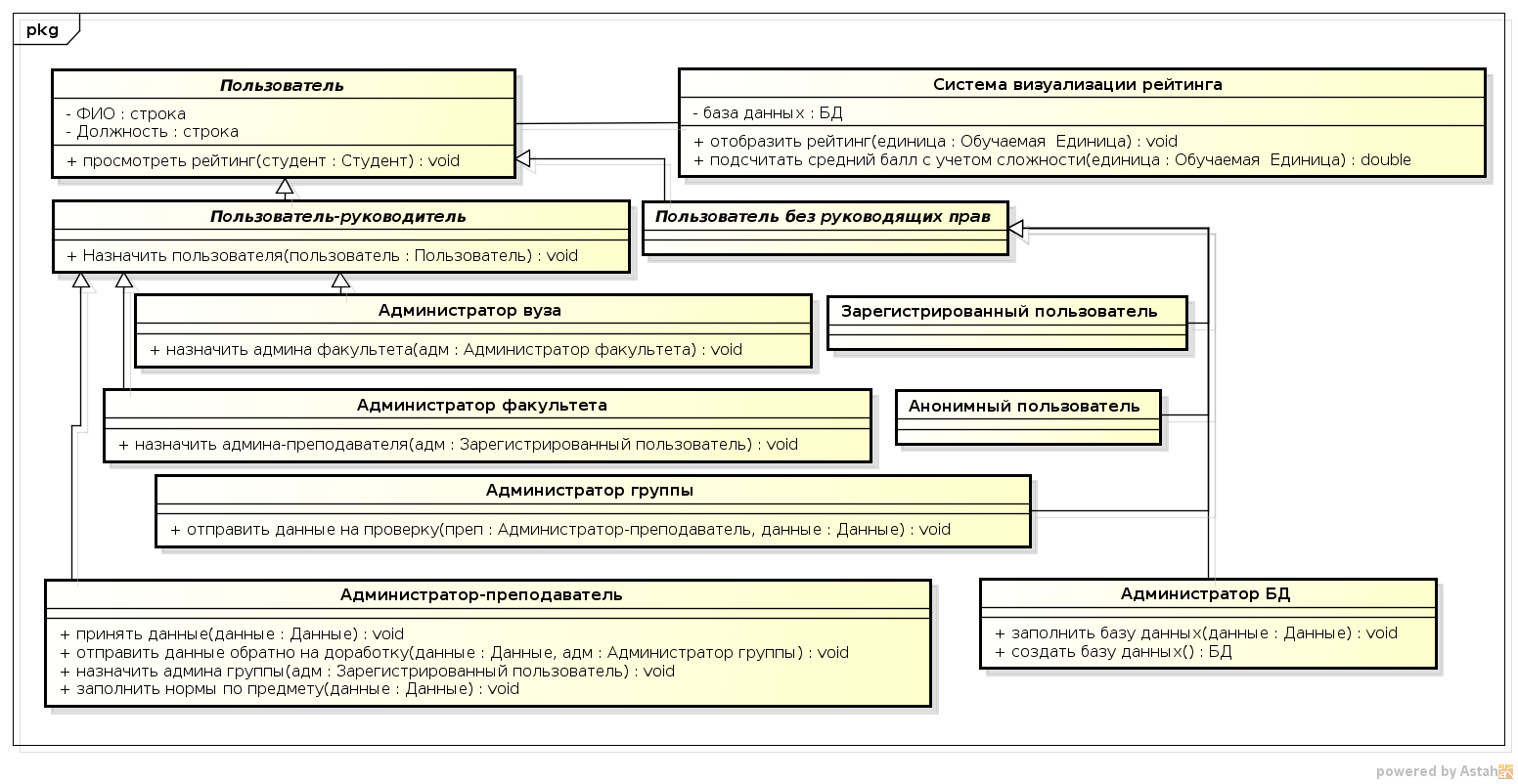
* + 1. Основные действия администратора вуза



* + 1. Сравнение визуализированных рейтингов обучающихся единиц



2.6 Диаграмма классов (концептуальный уровень)



|  |  |
| --- | --- |
| Класс | Описание полномочий класса |
| Пользователь | Содержит информацию о пользователе, а также метод просмотра визуального отображения рейтинга (Родительский класс для всех пользователей) |
| Зарегистрированный пользователь | Класс, необходимый для разграничения пользователей на зарегистрированных и незарегистрированных |
| Пользователь-руководитель | Класс, определяющий наличие руководящих полномочий у пользователя |
| Администратор факультета | Содержит методы для назначения администраторов-преподавателей и администраторов факультета |
| Администратор вуза | Содержит методы для назначения администраторов факультета |
| Администратор-преподаватель | Содержит метод для проверки полученных от администраторов групп данных, а также метод для отсылки данных на исправление. Кроме того, администратор-преподаватель заполняет нормы по предметам |
| Пользователь без руководящих прав | Класс, объединяющий пользователей, не имеющих руководящих прав |
| Администратор БД | Содержит метод, позволяющий заполнить имеющуюся БД |
| Администратор группы | Содержит метод для отправки внесенных данных администратору-преподавателю на проверку |
| Предмет | Содержит информацию по предмету |
| Система визуализации рейтинга | Содержит методы для подсчета среднего балла с учетом сложности, а также метод для визуализации полученного результата |

Таблица 1. Расшифровка классов диаграммы

2.7Диаграмма классов (физический уровень)

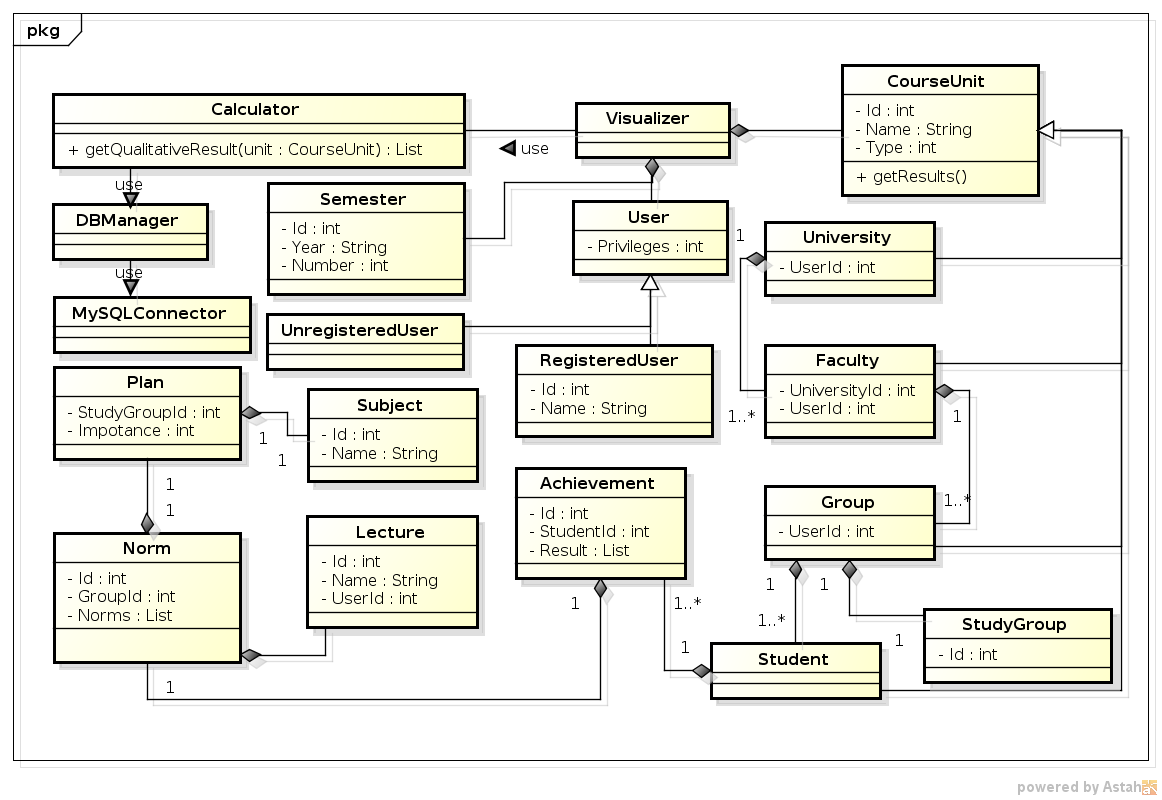


Таблица 2. Расшифровка классов диаграммы

|  |  |
| --- | --- |
| Класс | Описание полномочий класса |
| CourseInit | Отвечает за хранения результатов (вуза/факультета/группы/студента) в определенном семестре |
| University | Отвечает за хранения результатов вуза в определенном семестре |
| Faculty | Отвечает за хранения результатов факультета в определенном семестре |
| Group | Отвечает за хранения результатов группы в определенном семестре |
| Student | Отвечает за хранения результатов студента в определенном семестре |
| Achievement | Отвечает за хранения результатов и норм по учебному плану |

Продолжение таблицы 2

|  |  |
| --- | --- |
| Norm | Отвечает за хранения норм по учебному плану для определенной группы |
| Plan | Отвечает за хранения учебного плана, т.е. дисциплин и их важности |
| Subject | Отвечает за хранение дисциплины, а именно уникального номера и наименования. |
| Lecture | Отвечает за хранение данных о преподавателе, а именно уникального номера и ФИО |
| StudyGroup | Отвечает за хранение параметров для групп студентов, а именно наименование, факультет, учебный план |
| Visualizer | Отвечает за генерацию изображения по входным результатам рейтинга учебной единицы |
| Calculator | Осуществляет расчет среднего балла с учетом сложности предметов для выбранной обучающейся единицы или единиц |
| DBManager | Осуществляет работу пользователя с базой данных |
| Semester | Содержит информацию о семестре |
| User | Описание пользователя, его прав и обязанностей |

3 Режимы работы

URV поддерживает 5 режимов работы, описанных ниже.

Гостевой режим

Режим используется для просмотра визуализированного рейтинга учебной единицы.

Режим администратора группы

Режим используется для заполнения данных об успеваемости студентов определенной группы.

Режим администратора преподавателя

Режим используется для назначения норм предметам, а также для проверки данных, введенных администраторами групп, их исправления и принятия.

Режим администратора факультета

Режим используется для назначения администраторов группам и предметам, а также добавления предметов и групп.

Режим администратора вуза

Режим используется для добавления факультетов, и назначения администраторов факультетов.

3 Технологии реализации

Использование Node.js и его плагинов как основного языка разработки. Плагинная структура проектов.

4 Постанализ этапов разрабоки

Таблица 3. Разбиение задач на этапы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Этап I | | | | |
| Задача | Трудоемкость, в процентах | Исполнитель | Сколько сделал, в процентах | Статус |

Продолжение таблицы 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Анализ аналогов | 5 | Никитский Н.В. | 100% | Выполнено, без задержки |
| Макет GUI | 10 | Никитский Н.В.,  Поцелуйко А.С. | 70%  30% | Выполнено, без задержки |
| Архитектура БД | 12 | Никитский Н.В.,  Клевцов В.А. | 40%  60% | Выполнено, без задержки |
| Мат модель | 10 | Поцелуйко А. С. | 100% | Не законченно |
| Граф представление | 6 | Клевцов В.А. | 100% | Не закончено |
| Тесты | 6 | Клевцов В.А.,  Пестун М. Ю. | по 50% | Не закончено |
| Use case | 4 | Клевцов В.А.,  Пестун М. Ю. | 10%  90% | Выполнено, без задержки |
| Сценарии использования | 6 | Поцелуйко А.С. | 100% | Выполнено, без задержки |
| Сценарии по ВИ | 5 | Пестун М. Ю. | 100% | Выполнено, с незначительной задержкой |
| РП | 10 | Поцелуйко А.С.,  Клевцов В.А. | 90%  10% | Выполнено, без задержки |
| Функциональные требования | 10 | Никитский Н.В.,  Клевцов В.А. | по 50% | Выполнено, с задержкой |
| Логическая структура | 6 | Клевцов В.А. | 100% | Выполнено, с задержкой |
| Выбор технологии программирования | 10 | Клевцов В.А. | 100% | Выполнено, с задержкой |
| Этап II | | | | |
| Диаграмма ER | 30 | Клевцов В.А.,  Никитский Н.В. | 50%  50% | Выполнено, с задержкой |
| Логическая диаграмма классов | 25 | Поцелуйко А.С. | 100% | Выполнено |
| Физическая диаграмма классов | 25 | Клевцов В.А.,  Никитский Н.В. | 30%  70% | Выполнено, с задержкой |
| Объектная модель | 10 | Клевцов В.А. | 100% | Выполнено |
| CRC-карточки | 10 | Поцелуйко А.С. | 100% | Выполнено |

1. Оценка трудоемкости задач проекта

# Таблица 4. Основные функции

|  |  |
| --- | --- |
| Требование | Оценка трудоемкости (в процентах) |
| Визуализация рейтинга | 20 |
| Вычисление системного рейтинга | 30 |
| Административное ядро | 30 |
| Взаимодействие с БД | 20 |

# Таблица 5. Подфункции функции «Визуализация рейтинга»

|  |  |
| --- | --- |
| Требование | Оценка трудоемкости (в процентах) |
| Реализация различных картинок по курсам и оценкам | 40 |
| Расчет и расположение персонажей на фоновой картинке | 40 |
| Настройка рас по курсам | 15 |
| Обеспечение фонового рисунка | 5 |

# Таблица 6. Подфункции функции «Вычисление системного рейтинга»

|  |  |
| --- | --- |
| Требование | Оценка трудоемкости (в процентах) |
| Вычисление рейтинга для студента по предмету | 30 |
| Вычисление общего рейтинга студента | 30 |
| Вычисление рейтинга группы | 20 |
| Вычисление рейтинга факультета | 20 |

# 

# Таблица 7. Подфункции функции “Административное ядро”

|  |  |
| --- | --- |
| Требование | Оценка трудоемкости (в процентах) |
| Интерфейс администратора группы | 10 |
| Интерфейс администратора предмета | 20 |
| Интерфейс администратора факультета | 30 |
| Интерфейс администратора вуза | 10 |
| Интерфейс сравнения (поиск) | 10 |
| Интерфейс сравнения (сравнение) | 10 |
| Интерфейс сравнения (добавление) | 10 |

# 

# Таблица 8. Подфункции функции “Взаимодействие с БД”

|  |  |
| --- | --- |
| Требование | Оценка трудоемкости (в процентах) |
| Взаимодействие с БД(запросы) | 20 |
| Взаимодействие с БД(добавление) | 10 |
| Взаимодействие с БД(удаление) | 30 |
| Взаимодействие с БД(редактирование) | 30 |
| Создание классов-контейнеров копий объектов записей в БД | 10 |

1. CRC-карточки

Таблица 9. Карточка класса CourseUnit

|  |  |
| --- | --- |
| Название класса  CourseUnit | Суперкласс |
| Подклассы  University, Faculty, Group, Student | |
| Ответственность  Содержит (абстрактные) методы получения среднего балла с учетом сложности предметов обучающейся единицы | |
| Кооперация  Visualizer | |

Таблица 10. Карточка класса University

|  |  |
| --- | --- |
| Название класса  University | Суперкласс  CourseUnit |
| Подклассы | |
| Ответственность  Содержит метод для получения среднего балла с учетом сложности предметов по всему университету | |
| Кооперация  Faculty | |

Таблица 11. Карточка класса Faculty

|  |  |
| --- | --- |
| Название класса  Faculty | Суперкласс  CourseUnit |
| Подклассы | |
| Ответственность  Содержит метод для получения среднего балла с учетом сложности предметов по одному факультету | |
| Кооперация  Group | |

Таблица 12. Карточка класса Group

|  |  |
| --- | --- |
| Название класса  Group | Суперкласс  CourseUnit |
| Подклассы | |
| Ответственность  Содержит метод для получения среднего балла с учетом сложности предметов по одной группе | |
| Кооперация  Faculty, StudyGroup, Student | |

Таблица 13. Карточка класса Student

|  |  |
| --- | --- |
| Название класса  Student | Суперкласс  CourseUnit |
| Подклассы | |
| Ответственность  Содержит метод для получения среднего балла с учетом сложности предметов по одному студенту | |
| Кооперация  Group, Achievement | |

Таблица 14. Карточка класса Achievement

|  |  |
| --- | --- |
| Название класса  Achievement | Суперкласс |
| Подклассы | |
| Ответственность  Содержит метод для получения всех баллов студента за определенный срез | |
| Кооперация  Student, Norm | |

Таблица 15. Карточка класса Norm

|  |  |
| --- | --- |
| Название класса  Norm | Суперкласс |
| Подклассы | |
| Ответственность  Содержит нормы по всем предметам для конкретной группы | |
| Кооперация  Achievement, Plan, Lecture | |

Таблица 16. Карточка класса Plan

|  |  |
| --- | --- |
| Название класса  Plan | Суперкласс |
| Подклассы | |
| Ответственность  Получение важности предмета для определенной группы | |
| Кооперация  Norm, Subject | |

Таблица 17. Карточка класса Subject

|  |  |
| --- | --- |
| Название класса  Subject | Суперкласс |
| Подклассы | |
| Ответственность  Класс данных для работы с БД | |
| Кооперация  Plan | |

Таблица 18. Карточка класса Lecture

|  |  |
| --- | --- |
| Название класса  Lecture | Суперкласс |
| Подклассы | |
| Ответственность  Информация о преподавателе | |
| Кооперация  Norm | |

Таблица 19. Карточка класса StudyGroup

|  |  |
| --- | --- |
| Название класса  StudyGroup | Суперкласс |
| Подклассы | |
| Ответственность  Получение студентов по семестрам | |
| Кооперация  Group | |

Таблица 20. Карточка класса Visualizer

|  |  |
| --- | --- |
| Название класса  Visualizer | Суперкласс |
| Подклассы | |
| Ответственность  Осуществляет визуализацию высчитанного балла с помощью персонажа из мира | |
| Кооперация  CourseUnit, User, Semester, Calculator | |

Таблица 21. Карточка класса Calculator

|  |  |
| --- | --- |
| Название класса  Calculator | Суперкласс |
| Подклассы | |
| Ответственность  Осуществляет расчет среднего балла с учетом сложности предметов для выбранной обучающейся единицы или единиц | |
| Кооперация  Visualizer, DBManager | |

Таблица 22. Карточка класса DBManager

|  |  |
| --- | --- |
| Название класса  DBManager | Суперкласс |
| Подклассы | |
| Ответственность  Осуществляет работу пользователя с базой данных | |
| Кооперация  MySQLConector, Calculator | |

Таблица 23. Карточка класса Semester

|  |  |
| --- | --- |
| Название класса  Semester | Суперкласс |
| Подклассы | |
| Ответственность  Содержит информацию о любом семестре | |
| Кооперация  Visualizer | |

Таблица 24. Карточка класса User

|  |  |
| --- | --- |
| Название класса  User | Суперкласс |
| Подклассы | |
| Ответственность  Описание пользователя, его прав и обязанностей | |
| Кооперация  Visualizer | |

1. Паспорта классов

Таблица 25. Паспорт класса «Plan»

|  |  |
| --- | --- |
| Имя | Plan |
| Назначение | Класс, содержащий информацию о плане обучения: предмет, его нормы и группу, закрепленную за этим предметом |
| Язык | Php |
| Родитель | - |
| Свойства | StudyGroup, Subject, Norm, Importance |
| Методы | Сеттеры и геттеры на каждое свойство |
| Литература | - |

Таблица 26. Паспорт класса «Subject»

|  |  |
| --- | --- |
| Имя | Subject |
| Назначение | Класс, содержащий информацию по предмету: его название и идентификатор. |
| Язык | Php |
| Родитель | - |
| Свойства | Id, Name |
| Методы | Сеттер и геггтер для наименования предмета |
| Литература | - |

Таблица 27. Паспорт класса «Calculator»

|  |  |
| --- | --- |
| Имя | Calculator |
| Назначение | Класс для расчета среднего балла обучающейся единицы с учетом сложности предмета. |
| Язык | Php |
| Родитель | - |
| Свойства |  |
| Методы | getQualitativeResult(unit: CourseUnit) List getQualitativeResult(unit: CourseUnit[], subj Subject) List |
| Литература | - |

Таблица 28. Паспорт метода «getQualitativeResult»

|  |  |
| --- | --- |
| Имя | getQualitativeResult |
| Назначение | Получить качественную оценку среднего балла с учетом сложности предметов |
| Язык | Php |
| Принадлежность к классу | Принадлежит классу Calculator |
| Входные параметры | result : float |
| Возвращаемые параметры | String |
| Вызываемые методы | - |

Алгоритм: в зависимости от значения входного параметра result выбирается одна качественная оценка из списка.

|  |  |
| --- | --- |
| Res | Return |
| <61 | Disgusting |
| 61-66 | Bad- |
| 67-71 | Bad |
| 72-75 | Bad+ |
| 76-81 | Good- |
| 82-85 | Good |
| 86-89 | Good+ |
| 90-94 | Excellent- |
| 95-100 | Excellent |

Тесты:

1. Качественная оценка – Ужасно

Вход: res = 53

Выход: “Disgusting”

1. Качественная оценка – Плохо-

Вход: res = 63

Выход: “Bad-”

1. Качественная оценка – Плохо

Вход: res = 71

Выход: “Bad”

1. Качественная оценка – Плохо+

Вход: res = 72

Выход: “Bad+”

1. Качественная оценка – Хорошо-

Вход: res = 79

Выход: “Good-”

1. Качественная оценка – Хорошо

Вход: res = 84

Выход: “Good”

1. Качественная оценка – Хорошо+

Вход: res = 89

Выход: “Good+”

1. Качественная оценка – Отлично-

Вход: res = 90

Выход: “Excellent-”

1. Качественная оценка – Отлично+

Вход: res = 99

Выход: “Excellent”

1. Качественная оценка отрицательная

Вход: res = -10

Выход: “Incorrect”

1. Качественная оценка больше ста

Вход: res = 102

Выход: “Incorrect”

Таблица 29. Паспорт метода «GetUnityDescription»

|  |  |
| --- | --- |
| Имя | GetUnityDescription |
| Назначение | Получить описание обучающейся единицы |
| Язык | Php |
| Принадлежность к классу | Vizualizer |
| Входные параметры | Unit : CourseUnit |
| Возвращаемые параметры | Name: String |
| Вызываемые методы | getUniversityName(), getFacultyName(), GetStudentName(), getStudentGroup(), getGroupName(). |

Алгоритм: в зависимости от типа обучающейся единицы (courseUnit), строится ее описание.

|  |  |
| --- | --- |
| Тип CourseUnit | Return |
| Student | «Студент (полное имя). Группа (название группы)» |
| Group | «Группа (название группы)» |
| Faculty | «(Название факультета)» |
| University | «Университет (Название университета)» |

Тесты:

1. CourseUnit типа Student

Входные данные:

Student.Name = «Иван»

Student.Surname = «Иванов»

Student.SecondName = «Иванович»

Student.Group = «ХТ-120»

Выходные данные:

«Иванов И.И. Группа ХТ-120»

1. CourseUnit типа Group

Входные данные:

Group.Name = «ХТ-120»

Выходные данные:

«Группа ХТ-120»

1. CourseUnit типа Faculty

Входные данные:

Faculty.Name = fevt

Выходные данные:

«Факультет электроники и вычислительной техники»

1. CourseUnit типа University

Входные данные:

University.Name = «ВолгГТУ»

Выходные данные:

«Университет ВолгГТУ»

Таблица 30. Паспорт метода «GetDescription»

|  |  |
| --- | --- |
| Имя | GetDescription |
| Назначение | Получить описание семестра на основе его номера и года |
| Язык | Php |
| Принадлежность к классу | Semester |
| Входные параметры | - |
| Возвращаемые параметры | Description:String |
| Вызываемые методы | - |

Алгоритм: Составить строку вида «Семестр (номер семестра), (номер года) год».

Тесты:

1. Семестр и год заданы верно

Входные данные:

Semester.year = 2014;

Semester.number = 1;

Выходные данные: «Семестр 1, 2014 год»

1. Семестр задан числом от (3 до 12) как задают для студентов 2-6 курсов.

Входные данные:

Semester.year = 2014;

Semester.number = 7;

Выходные данные: «Семестр 1, 2014 год»

1. Семестр - отрицательное число

Входные данные:

Semester.year = 2014;

Semester.number = -10;

Выходные данные: «Incorrect semester»

1. Год отрицателен

Входные данные:

Semester.year = -2014;

Semester.number = 1;

Выходные данные: «Incorrect year»

1. Год больше текущего

Входные данные:

Semester.year = 2018;

Semester.number = 1;

Выходные данные: «Incorrect year»

1. Семестр больше 12

Входные данные:

Semester.year = 2014;

Semester.number = 15;

Выходные данные: «Incorrect semester»

Таблица 31. Паспорт класса « Визуализатор»

|  |  |
| --- | --- |
| Имя | Визуализатор |
| Назначение | Визуализация качественного рейтинга. |
| Язык | Php/Javascript |
| Родитель | --- |
| Свойства | Название(?) |
| Методы | Визуализировать, |
| Литература | ---- |

Таблица 32. Паспорт класса «Студент»

|  |  |
| --- | --- |
| Имя | Студент |
| Назначение | Сущность реализующая запись из таблицы БД. |
| Язык | Php/Javascript |
| Родитель | CourseInit |
| Свойства | Имя, Группа, id. |
| Методы | Имя, Группа, Успеваемость по предмету. |
| Литература | ---- |

Таблица 33. Паспорт класса «Реализация»

|  |  |
| --- | --- |
| Имя | Реализация |
| Назначение | Хранение норм, заданных преподавателем по определенному предмету |
| Язык | Php/Javascript |
| Родитель | --- |
| Свойства | Нормы, Группа, id. |
| Методы | Нормы, Группа, Предмет. |
| Литература | ---- |

Таблица 34. Паспорт метода «Визуализация»

|  |  |
| --- | --- |
| Имя | Визуализация |
| Назначение | По заданным параметрам получить графическое изображение являющиеся визуализации успеваемости учебной единицы. |
| Язык | Php/Javascript |
| Принадлежность | Визуализатор |
| Входные параметры | Учебная единица, по всем предметам, предмет |
| Выходные параметры | Графическое изображение студента |
| Вызываемые методы | Получение качественного рейтинга |
| Алгоритм | 1) Если входной параметр “по всем предметам” истинен то:  - запрашиваем качественный рейтинг учебной единицы по всем предметам  Иначе:  - запрашиваем качественный рейтинг учебной единицы по предмету  2) По качественному рейтингу, получаем изображение и возвращаем его. |
| Тесты | 1) все данные правильные  студент – Петров,  ложь,  Философия.  Результат  изображение соответствующее первокурснику, с оценкой 82.  2) предмета нет в БД  студент – Петров,  ложь,  гнусология.  Результат  сообщение об ошибке.  3)все данные правильные, запрос по всем предметам  студент – Петров,  истина,  null.  Результат  изображение соответствующее первокурснику, с оценкой 90. |
| Литература | --- |

Таблица 35. Паспорт метода «Запрос студента»

|  |  |
| --- | --- |
| Имя | Запрос студента |
| Назначение | Получение объекта студента из БД. |
| Язык | Php/Javascript |
| Принадлежность | Менеджер БД |
| Входные параметры | Имя, Семестр, Группа(?). |
| Выходные параметры | Студент |
| Вызываемые методы | --- |
| Алгоритм | 1) Выполняем запрос к БД по имени и семестру.  2) Возвращаем студента. |
| Тесты | 1) все данные правильные  студент – Петров,  семестр 3, 2010  Результат  объект сущность студент.  2) студента нет в БД  студент – Петровский3,  семестр 3, 2010  Результат  null.  3)семестр задан неверно  студент – Петров,  семестр 14, 2010  Результат  null. |
| Литература | --- |
| Имя | Запрос успеваемости |
| Назначение | Получение успеваемости учебной единицы из бд. |
| Язык | Php/Javascript |
| Принадлежность | Менеджер БД |
| Входные параметры | Студент, Семестр, Предмет |
| Выходные параметры | Массив объектов класса успеваемость. |
| Вызываемые методы | Запрос успеваемости, запрос предмета по студенту. |
| Алгоритм | Если входной параметр “по всем предметам” истинен то:  1) запрашиваем список предметов для студента  2) запрашиваем успеваемость для студента по каждому предмету в цикле  3) возвращаем информацию  Иначе:  1) запрашиваем успеваемость студента по предмету и возвращаем ее. |

Продолжение таблицы 35

|  |  |
| --- | --- |
| Тесты | 1) все данные правильные  студент – Петров,  семестр 3, 2010  ложь,  Физика  Результат  Массив успеваемостей, с одним элементом.  2) все данные правильные  студент – Петров,  семестр 3, 2010  истина,  null  Результат  Массив успеваемостей, с элементами по всем предметам.  3) студента нет в БД  студент – Петровский3,  семместр 3, 2010,  ложь,  Физика  Результат  null.  4)семестр задан неверно  студент – Петров,  семестр 14, 2010,  ложь,  Физика  Результат  null.  5) предмет задан неверно  студент – Петров,  семместр 3, 2010  ложь,  Физика2  Результат  null. |
| Литература | --- |

Таблица 36. Паспорт класса «Faculty»

|  |  |
| --- | --- |
| Имя | Faculty |
| Назначение | Содержит метод для получения среднего балла с учетом сложности предметов по одному факультету |
| Язык | Php |
| Родитель | CourseUnit |
| Свойства | unitid: int  unitName: string |
| Методы | getType(): String  getCommonBalls(): float  getResultList(): List |
| Литература | - |

Таблица 37. Паспорт класса « Semester »

|  |  |
| --- | --- |
| Имя | Semester |
| Назначение | Содержит информацию о любом семестре |
| Язык | Php |
| Родитель | - |
| Свойства | Year: string  Number: int |
| Методы | getDescription(): String |
| Литература | - |

Таблица 38. Паспорт метода « getCommonResult »

|  |  |
| --- | --- |
| Имя | getCommonResult(unit: courseUnit): float |
| Назначение | получения среднего балла обучающейся единицы |
| Язык | Php |
| Принадлежит классу | Calculator |
| Вход | Обучающаяся единица, средний балл которой нужно получить |
| Выход | Значение среднего балла обучающейся единицы |
| Вызываемые методы |  |
| Алгоритм метода | Для получения среднего балла обучающейся единицы сначала будет получены баллу по каждому из предметов, изучаемых данной единицей. Затем будет вычислено среднее арифметическое значение всех баллов, которое и будет являться средним рейтингом обучающейся единицы |

Продолжение таблицы 38

|  |  |
| --- | --- |
| Тесты (положительные) | - |
| Литература | - |

Таблица 39. Паспорт класса «Lecture»

|  |  |
| --- | --- |
| Имя | Lecture |
| Назначение | Сущность реализующая запись Преподаватель из БД. |
| Язык | Php |
| Родитель | - |
| Свойства | Id, Name, UserId |
| Методы | - |
| Литература | - |

Таблица 40. Паспорт класса «DBManager»

|  |  |
| --- | --- |
| Имя | DBManager |
| Назначение | Класс для работы с базай данных. |
| Язык | Php |
| Родитель | - |
| Свойства | - |
| Методы | getRecords(table : String, params : List) : List  addRecord(table : String, params : List) : boolean  deleteRecord(table : String, params : List) : boolean  updateRecord(table : String, params : List) : boolean |
| Литература | - |

Таблица 41. Паспорт класса «CourseUnit»

|  |  |
| --- | --- |
| Имя | CourseUnit |
| Назначение | Класс для хранения учебной ндиницы. |
| Язык | Php |
| Родитель | - |
| Свойства | Name, Id |
| Методы | getType() : String  getCommonMarks() : float  getResultsList() : List |
| Литература | - |

Таблица 42. Паспорт метода «getRecords»

|  |  |
| --- | --- |
| Имя | getRecords |
| Назначение | Получение записей из таблица БД. |
| Язык | Php |
| Принадлежность | BDManager |
| Входные параметры | table : String  params : List |
| Выходные параметры | records : List |
| Вызываемые методы | - |
| Алгоритм | 1. Выполняем запрос к БД по имени таблицы и по параметрам. 2. Возвращаем записи удовлетворяющие запросу. |
| Тесты | 1. Все данные правильные без параметров   Входные данные:  table = “faculty”  Выходные данные:  {((“name” : “ФЭВТ”), (“id” : “1”), (“user\_id” : “12”)),…,( (“name” : “ФАТ”), (“id” : “8”), (“user\_id” : “104”))}   1. Все данные правильные с параметрами   Входные данные:  table = “faculty”  params = {(“user\_id” : “104”)}  Выходные данные:  {( (“name” : “ФАТ”), (“id” : “8”), (“user\_id” : “104”))}   1. Не правильно указана таблица   Входные данные:  table = “facultys”  params = {(“user\_id” : “104”)}  Выходные данные:  {}   1. Не правильно заданы параметры   Входные данные:  table = “facultys”  params = {(“user” : “104”)}  Выходные данные:  {} |
| Литература | - |

Таблица 43. Паспорт метода «addRecord»

|  |  |
| --- | --- |
| Имя | addRecord |
| Назначение | Добавить объект в БД. |
| Язык | Php |
| Принадлежность | BDManager |
| Входные параметры | table : String  params : List |
| Выходные параметры | Boolean |
| Вызываемые методы | - |
| Алгоритм | 1. Выполняем запрос к БД по имени таблицы и по параметрам на добавление записи. 2. Возвращаем true если операция прошла успешно. |
| Тесты | 1. Все данные правильные без параметров   Входные данные:  table = “user”  Выходные данные:  false   1. Все данные правильные с параметрами   Входные данные:  table = “user”  params = {(“name” : “Иванов Иван Иванович”), (“password” : “qwerty”), (“access” : “1”)}  Выходные данные:  true   1. Не правильно указана таблица   Входные данные:  table = “users”  params = {(“name” : “Иванов Иван Иванович”), (“password” : “qwerty”), (“access” : “1”)}  Выходные данные:  false   1. Не правильно заданы параметры   Входные данные:  table = “user”  params = {(“name” : “Иванов Иван Иванович”), (“access” : “1”)}  Выходные данные:  false |
| Литература | - |

Таблица 44. Паспорт метода «deleteRecord»

|  |  |
| --- | --- |
| Имя | deleteRecord |
| Назначение | Удалить объект из БД. |
| Язык | Php |
| Принадлежность | BDManager |
| Входные параметры | table : String  params : List |
| Выходные параметры | Boolean |
| Вызываемые методы | - |
| Алгоритм | 1. Выполняем запрос к БД по имени таблицы и по параметрам на удаление записи. 2. Возвращаем true если операция прошла успешно. |
| Тесты | 1. Все данные правильные без параметров   Входные данные:  table = “student”  Выходные данные:  true   1. Все данные правильные с параметрами   Входные данные:  table = “student”  params = {(“id” : (1,10,103))}  Выходные данные:  true   1. Не правильно указана таблица   Входные данные:  table = “students”  Выходные данные:  false   1. Не правильно заданы параметры   Входные данные:  table = “student”  params = {(“id” : (-1,15))}  Выходные данные:  false |
| Литература | - |