Определение архитектуры программного обеспечения

Оглавление:

Оглавление

[1. Вступление 1](#_Toc84615790)

[1.1. Применяемые в разаработке технологии 1](#_Toc84615791)

[1.2. Ссылки 2](#_Toc84615792)

[2. Модульные структуры системы 2](#_Toc84615793)

[2.1 Общие модульные структуры 2](#_Toc84615794)

[2.2. Модульные структуры сервера 4](#_Toc84615795)

[2.3. Модульные структуры клиента 6](#_Toc84615796)

[3. Структуры «компонент» и «соединитель» системы 7](#_Toc84615797)

[3.1 Сервер 7](#_Toc84615798)

[3.2 Клиент 7](#_Toc84615799)

[4. Структура распределения системы 8](#_Toc84615800)

[5. Общая схема 9](#_Toc84615801)



Белов А.И. группа 4932

# Вступление

В качестве архитектуры п.о. выбрана трехуровневая клиент-серверная архитектура, членами которой являются: Реляционная база данных(далее – «БД»), RESTfull API сервер(далее - «сервер») и Desktop клиент(далее - «клиент»).

Данная архитектура выбрана по ряду причин:

1. Безопасность: сервер отвечает за выдачу прав пользователям и ограничивает доступ к бд.
2. Частота обновления данных: предметная область предполагает обновление данных практически ежедневно, что ограничивает возможность хранения данных на клиенте.
3. Размер клиента: основной функционал системы – информационный, из-за чего скорость и вес приложения становятся более приоритетными.
4. Мультиплатформенность: при необходимости портировать систему на другие платформы нужно будет переписывать только клиент.

В силу финансовых ограничений сервер и БД расположены на малопроизводительном бесплатном хостинге. Поэтому на данном этапе разработки было решено оставить часть бизнес-логики в клиенте.

Выбранные технологии для разработки сервера и клиента(п.п. 1.1.) мультиплатформены, что позволит без проблем развернуть систему на другом веб-сервере и других операционных системах.



## Применяемые в разработке технологии

**Для БД**:

Язык запросов: SQL

Система управления: MS SQL 2019 Express

**Для сервера:**

Язык программирования: C#

Платформа разработки: ASP.NET Core

IDE: Windows – Visual Studio, Linux – Rider

Веб-сервер: IIS

**Для клиента:**

Язык программирования: Kotlin и Java

Система сборки: Gradle

IDE: Intellij IDEA

## Ссылки

Сервер - <http://lichuhasite.somee.com>

БД - <http://universitylkdb.mssql.somee.com>

Репозиторий - <https://github.com/Lichuhah/SUAI_5sem>

# Модульные структуры системы

## 2.1 Общие модульные структуры

На сервере и клиенте представлен одинаковый набор классов-**моделей,** не реализующих никакой логики и представляющих собой исключительно набор данных, *далее для обозначения любой из данных моделей будет использоваться* ***Entity***

|  |
| --- |
| User |
| Назначение: пользователь системы |
| Структуры и типы данных модуля:  Int id, string login, string password, UserRole role, string phonenumber, string mail, Fullname fullname |
| Декомпозиция: связан с Fullname, UserRole |
| Класс или обобщение: класс |

|  |
| --- |
| Fullname |
| Назначение: информация о фио пользователя |
| Структуры и типы данных модуля:  string name, string lastname, string secondname |
| Декомпозиция: связан с User |
| Класс или обобщение: класс |

|  |
| --- |
| UserRole |
| Назначение: роли пользователей |
| Структуры и типы данных модуля:  1 – Student, 2 – Lecture, 3- Deanat |
| Декомпозиция: связан с User |
| Класс или обобщение: перечисление |

|  |
| --- |
| Student |
| Назначение: информация о студенте |
| Структуры и типы данных модуля:  Group group |
| Декомпозиция: связан с Group |
| Класс или обобщение: подкласс User |

|  |
| --- |
| Lecture |
| Назначение: информация о преподавателе |
| Структуры и типы данных модуля: |
| Декомпозиция: |
| Класс или обобщение: подкласс User |

|  |
| --- |
| Deanat |
| Назначение: информация о сотруднике деканата |
| Структуры и типы данных модуля: |
| Декомпозиция: |
| Класс или обобщение: подкласс User |

|  |
| --- |
| Discipline |
| Назначение: информация о дисциплине |
| Структуры и типы данных модуля:  Int id, string name, int hours |
| Декомпозиция: |
| Класс или обобщение: класс |

|  |
| --- |
| Group |
| Назначение: информация об учебной группе |
| Структуры и типы данных модуля:  Int id, string number, int course, Speciality speciality |
| Декомпозиция: связан с Speciality |
| Класс или обобщение: класс |

|  |
| --- |
| GroupDiscipline |
| Назначение: информация об учебной дисциплине группы |
| Структуры и типы данных модуля:  Group group, Discipline discipline, ExamType examtype, int hours, Lecture lecture |
| Декомпозиция: связан с Discipline, Group, ExamType, Lecture |
| Класс или обобщение: класс |

|  |
| --- |
| ExamType |
| Назначение: виды экзаменации |
| Структуры и типы данных модуля:  1-EXAM,2-STANDING,3-DIFFE,4-COURCE\_PROJECT,5-COURSE\_WORK |
| Декомпозиция: связан с GroupDiscipline |
| Класс или обобщение: перечисление |

|  |
| --- |
| Speciality |
| Назначение: информация о специальности |
| Структуры и типы данных модуля:  Int id, string name, string code, EducationType type |
| Декомпозиция: связан с EducationType |
| Класс или обобщение: класс |

|  |
| --- |
| EducationType |
| Назначение: ступени образования |
| Структуры и типы данных модуля:  1-BACHELOR, 2-SPECIALTY, 3-MASTER, 4-TRAINEESHIP |
| Декомпозиция: связан с Speciality |
| Класс или обобщение: перечисление |

|  |
| --- |
| Task |
| Назначение: информация о задании |
| Структуры и типы данных модуля:  Int id, date deadline, int maxmark, string name, int number, string description, GroupDiscipline group\_discipline |
| Декомпозиция: связан с GroupDiscipline |
| Класс или обобщение: класс |

|  |
| --- |
| CompleteTask |
| Назначение: информация о задании для студента |
| Структуры и типы данных модуля:  Int id, Task task, int mark, bool status |
| Декомпозиция: связан с Task |
| Класс или обобщение: класс |

|  |
| --- |
| PassTask |
| Назначение: информация о сдаче задания |
| Структуры и типы данных модуля:  Int id, CompleteTask task, string answer, date date, string description |
| Декомпозиция: связан с CompleteTask |
| Класс или обобщение: класс |

|  |
| --- |
| Material |
| Назначение: учебный материал |
| Структуры и типы данных модуля:  Int id, Group group, Lecture lecture, File file |
| Декомпозиция: связан с Group, Lecture |
| Класс или обобщение: класс |

## 2.2. Модульные структуры сервера

|  |
| --- |
| AuthRequest |
| Назначение: данные авторизации приходящие с клиента |
| Структуры и типы данных модуля:  String login, string password |
| Декомпозиция: |
| Класс или обобщение: класс |

|  |
| --- |
| AuthResponce |
| Назначение: ответ клиенту на запрос авторизации |
| Структуры и типы данных модуля:  Int idUser, UserRole role |
| Декомпозиция: связан с UserRole |
| Класс или обобщение: класс |

|  |
| --- |
| NHIbernateHelper |
| Назначение: класс для работы с бибилотекой FluentNhibernate |
| Структуры и типы данных модуля:  IsessionFactory \_sessionFactory |
| Варианты использования:  Настроить  Открытьсессию  Закрытьсессию  Получитьтекущуюсессию |
| Декомпозиция: связан с CompleteTask |
| Класс или обобщение: класс |

Дальнейшие классы повторяются для каждой общей модели(Entity) из пункта 2.1 и наследуются от базовых версий, в документе приведена информация о базовых версиях модулей, остальные наследуются от представленных и по необходимости переимплементируют их логику.

|  |
| --- |
| EntityMapping<Entity> |
| Назначение: маппинг объекта из БД в соотвутвующий класс |
| Структуры и типы данных модуля: |
| Варианты использования:  Маппинг |
| Декомпозиция: связан с Entity |
| Класс или обобщение: обобщение |

|  |
| --- |
| EntityRepository<Entity> |
| Назначение: обеспечивает логику работы с БД |
| Структуры и типы данных модуля:  ISession session (создается с помощью NhibernateHelper) |
| Варианты использования:  ПолучитьВсе<Entity>  ПолучитьОдин(int id)  СохранитьОбъект(Entity entity)  УдалитьОбъект(Entity entity) |
| Декомпозиция: связан с Entity, NhibernateHelper |
| Класс или обобщение: обобщение |

|  |
| --- |
| EntityController<Entity> |
| Назначение: обеспечивает взаимодействие с клиентом |
| Структуры и типы данных модуля: |
| Варианты использования:  Get(int id) [HttpGet]  All() [HttpGet]  Post(Entity entity) [HttpPost]  Put(Entity entity) [HttpPut]  Delete(int id) [Http Delete] |
| Декомпозиция: связан с EntityRepository<Entity> |
| Класс или обобщение: обобщение |

|  |
| --- |
| LoginController |
| Назначение: обеспечивает авторизацию пользователя |
| Структуры и типы данных модуля: |
| Варианты использования:  Авторизация(AuthRequest data) |
| Декомпозиция: связан с AuthRequest, AuthResponce и UserRepository |
| Класс или обобщение: класс |

## 2.3. Модульные структуры клиента

|  |
| --- |
| Net |
| Назначение: обеспечивает отправку и принятие запросов на сервер |
| Структуры и типы данных модуля: |
| Варианты использования:  Создатьзапрос |
| Декомпозиция: |
| Класс или обобщение: класс |

Дальнейшие классы повторяются для каждой общей модели(Entity) из пункта 2.1 и наследуются от базовых версий, в документе приведена информация о базовых версиях модулей, остальные наследуются от представленных и по необходимости переимплементируют их логику.

|  |
| --- |
| EntityRepository<Entity> |
| Назначение: обеспечивает логику работы с сервером |
| Структуры и типы данных модуля: |
| Варианты использования:  ПолучитьВсе<Entity>  ПолучитьОдин(int id)  СохранитьОбъект(Entity entity)  УдалитьОбъект(Entity entity) |
| Декомпозиция: связан с Entity, Net |
| Класс или обобщение: обобщение |

|  |
| --- |
| EntityManager<Entity> |
| Назначение: обеспечивает бизнес-логику |
| Структуры и типы данных модуля: |
| Варианты использования:  ПолучитьВсе<Entity>  ПолучитьОдин(int id)  ДобавитьОбъект(Entity entity)  ИзменитьОбъект(Entity entity)  УдалитьОбъект(Entity entity) |
| Декомпозиция: связан с EntityRepository |
| Класс или обобщение: обобщение |

|  |
| --- |
| EntitiesScreen/EntityScreen |
| Назначение: обеспечивает отрисовку пользовательского интерфейса |
| Структуры и типы данных модуля: |
| Варианты использования:  ПолучитьДанные()  СоздатьФорму()  + функции для элементов UI |
| Декомпозиция: связан с EntityManager |
| Класс или обобщение: обобщение |

# Структуры «компонент» и «соединитель» системы

## 3.1 Сервер

|  |
| --- |
| База данных – EntityRepository (Модель) |
| Назначение: получение/добавление модели из/в БД |
| Совместно используемые данные: EntityMapping |
| Процесс: Получить, Сохранить, Удалить |

|  |
| --- |
| EntityRepository – EntityController (Модель) |
| Назначение: получение/добавление данных в зависимости от запроса |
| Совместно используемые данные: Entity или List<Entity> |
| Процесс: Получить, Добавить, Изменить, Удалить |

## 3.2 Клиент

|  |
| --- |
| EntityController – EntityRepository (Запрос/Ответ) |
| Назначение: получение/добавление данных в зависимости от ответа |
| Совместно используемые данные: Json запроса/ответа |
| Процесс: Get, Post, Put, Delete запросы |
| Параллелизм: Клиент1-Сервер, Клиент2-Сервер, Клиент3-Сервер ……… |
| Клиент-сервер:  Сервер(EntityController) – Клиент (EnityRepository)  Соединитель – Http протокол. |

|  |
| --- |
| EntityRepository – EntityManager (Модель) |
| Назначение: отправка/получение данных по запросу пользователя |
| Совместно используемые данные: Entity или List<Entity> |
| Процесс: Получить, Добавить, Изменить, Удалить |

|  |
| --- |
| EntityManager – EntityScreen (Данные) |
| Назначение: отправка данных пользователем или запрос на получение данных для отрисовки |
| Совместно используемые данные: данные формы, Entity или List<Entity> |
| Процесс: работа пользователя с UI |

# Структура распределения системы

|  |
| --- |
| Размещение:  БД(развертывание) - <ftp://UniversityLkDB.backup.somee.com/UniversityLkDB_MSSql_Database_Backup>  БД(репозиторий)- <https://github.com/Lichuhah/SUAI_5sem/tree/master/Fundamentals_of_Software_Engineering/back/lkDB>  Сервер(развертывание) - <ftp://192.52.242.121/www.lichuhasite.somee.com>  Сервер(репозиторий) - <https://github.com/Lichuhah/SUAI_5sem/tree/master/Fundamentals_of_Software_Engineering/back/lkAPI>  Клиент(репозиторий) -<https://github.com/Lichuhah/SUAI_5sem/tree/master/Fundamentals_of_Software_Engineering/front> |
| Реализация:  Сервер:  Entity - \*/Models  NHibernateHelper - \*/NHibernate  EntityMapping - \*/NHibernate/Mapping  EntityRepository - \*/Repositories  EntityController - \*/Controllers  Клиент:  Entity - \*/src/main/java/org/suai/common/Models  Net - \*/src/main/java/org/suai/common/  EntityRepository - \*/src/main/java/org/suai/common/Repositories  EntityManager - \*/src/main/java/org/suai/common/Managers  EntityScreen - src/main/kotlin/org/suai/common/Screen |
| Распределение функций:  Белов А.И. (C#, SQL) – БД и сервер  Иванов Н.С. (Java) – Бэк клиента (Net, EntityRepository, EntityManager)  Коваленко С.И. (Kotlin) – Фронт клиента (EntityScreen) |

# Общая схема

