Новосибирский Государственный
Университет Факультет Информационных
Технологий

Техническое описание проекта по курсу ООАД

NSU Timetable

Студенты ФИТ НГУ
Кузнецов Илья Максимович
Мурашев Артемий Дмитриевич
группа 23204

Версия 1.1.0

Содержание

Техническое описание проекта по курсу ООАД

1. Введение

1.1. Цель

Данный документ представляет собой техническое описание проекта NSU Timetable и содержит основные требования к разрабатываемой в рамках проекта программной системе и описание архитектуры программного решения.

1.2. Область действия

Документ разработан в рамках проекта *NSU Timetable* на основе стандартного шаблона и предназначен для использования студентами ФИТ и преподавателями дисциплины ООАД.

1.3. Определения и сокращения

Таблица 1: Определения и сокращения

Термин	Описание
Telegram бот	Telegram бот - это роботизированный аккаунт в мессенджере Telegram, который запрограммирован на автоматическое совершение действий

1.4. Краткое описание

Содержание данного документа построено таким образом, чтобы дать ответ на следующие вопросы:

- Какие проблемы предметной области должен решать будущий программный продукт
- Посредством какой функциональности системы будут достигнуто решение проблем предметной области
- Какова архитектура программного решения

Описание предметной области и проблем, для решения которых предназначен будущий программный продукт, приведены в разделе 2.

Раздел 3 содержит описание требований к программному решению, раздел 4 – описание архитектуры выбранного решения.

2. Предметная область проекта

В учебном процессе студентов и преподавателей НГУ ежедневно возникает необходимость работать с большим количеством информации: расписанием занятий, домашними заданиями, дедлайнами. Использование официальных сайтов или мобильного приложения НГУ часто оказывается неудобным: требуется ручной поиск расписания, отсутствует система напоминаний и учёта заданий. В условиях высокой учебной нагрузки студентам и преподавателям нужен инструмент, обеспечивающий быстрый и удобный доступ к этой информации.

2.1. Существующие проблемы

Студенты НГУ сталкиваются с рядом трудностей. Поиск актуального расписания требует дополнительных действий и времени. Отсутствует интегрированный инструмент для ведения и отслеживания домашних заданий. Нет автоматических напоминаний о начале и окончании пар, а также о дедлайнах. Сторонние приложения не всегда позволяют быстро поделиться информацией с одногруппниками и могут быть избыточно сложными. Эти проблемы затрудняют организацию учебного процесса и повышают риск пропустить занятие или просрочить выполнение задания.

2.2. Предполагаемое решение

Мы предлагаем создать Telegram-бота, который решает эти проблемы. Бот позволит быстро получать расписание по группе, преподавателю или аудитории. Будет реализована возможность привязки к своей группе для автоматического доступа к занятиям. Пользователи смогут добавлять домашние задания с дедлайнами и приоритетами сложности. Также бот будет отправлять уведомления о начале и окончании пар и напоминания о приближении дедлайнов. Использование Telegram как платформы делает бота доступным на любом устройстве без необходимости установки дополнительных приложений. Пользователи смогут оперативно получать информацию и управлять заданиями в удобном чате. Такой инструмент станет помощником в организации учебного процесса студентов и преподавателей НГУ.

3. Требования к программному решению

Данный раздел описывает требования к программной системе, разрабатываемой в рамках проекта NSU Timetable.

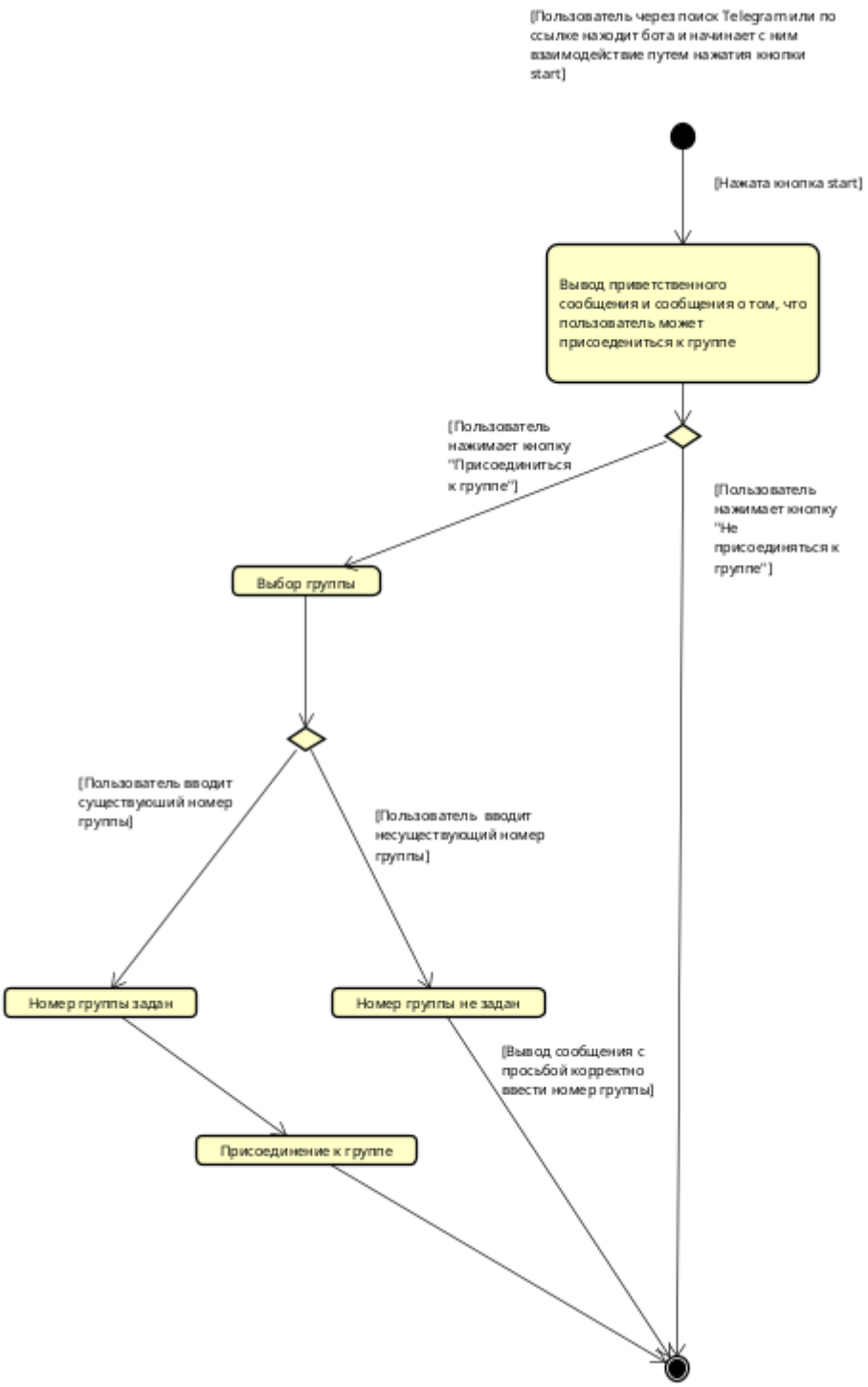
3.1. Роли

Роль - это что-то (например: другая система) или кто-то (например: человек) вне системы, которые взаимодействуют с ней. В предлагаемой к разработке системе идентифицированы следующие роли:

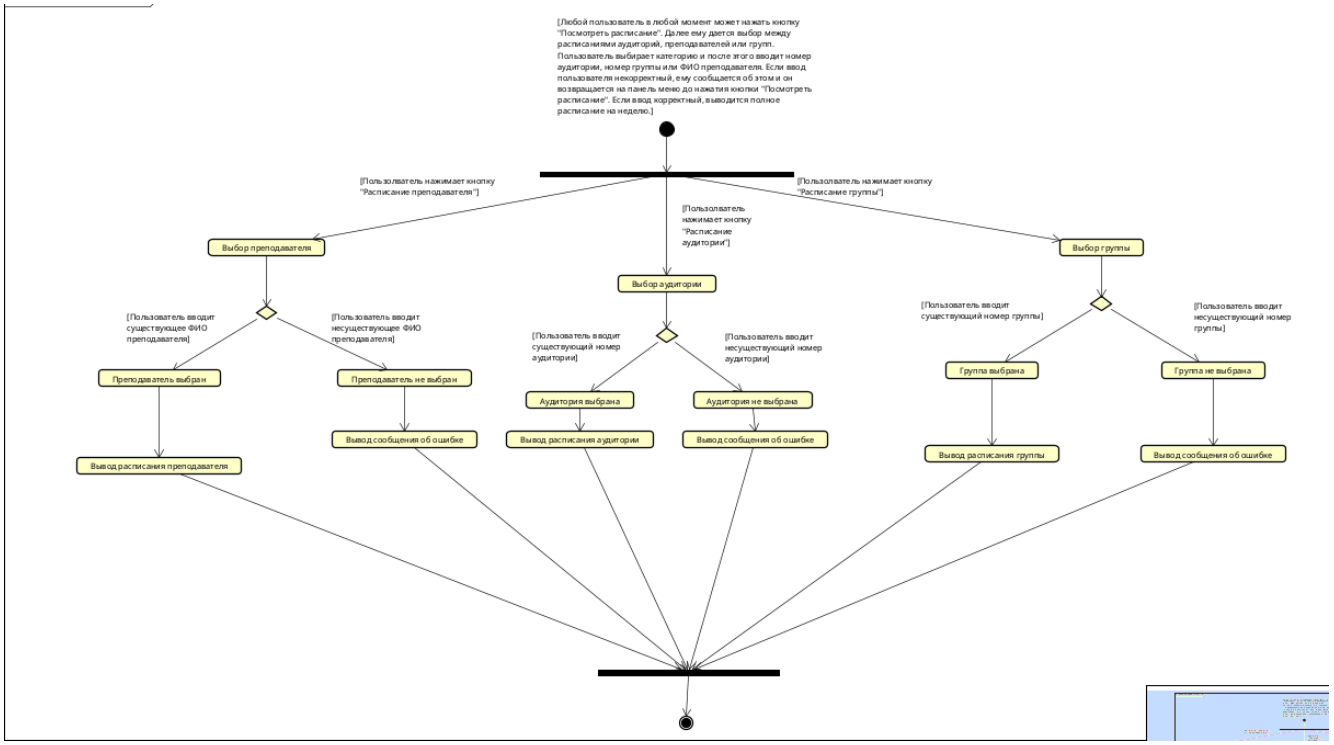
1. *Неавторизованный пользователь* - пользователь, который еще не осуществил аутентификацию в боте. Имеет возможность просматривать расписание групп, аудиторий, преподавателей.
2. *Авторизованный пользователь* - пользователь, имеющий доступ ко всем предоставляемым услугам бота с возможностью прикрепления домашнего задания, редактирования домашнего задания, удаления домашнего задания, установки отметки о выполнении домашнего задания

3.2. Функциональные требования для роли Неавторизованный пользователь

3.2.1. Авторизация

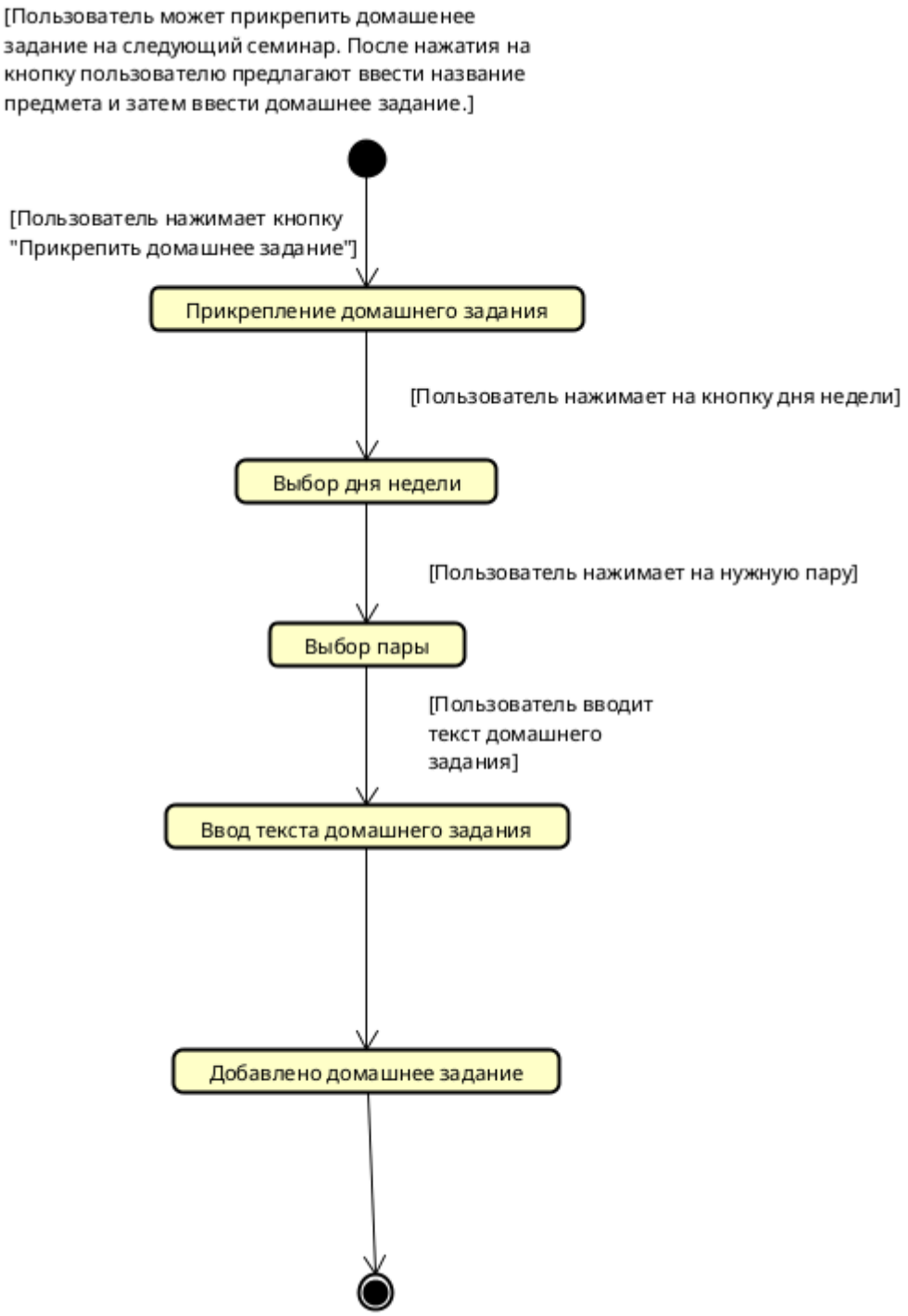


3.2.2. Просмотр расписания

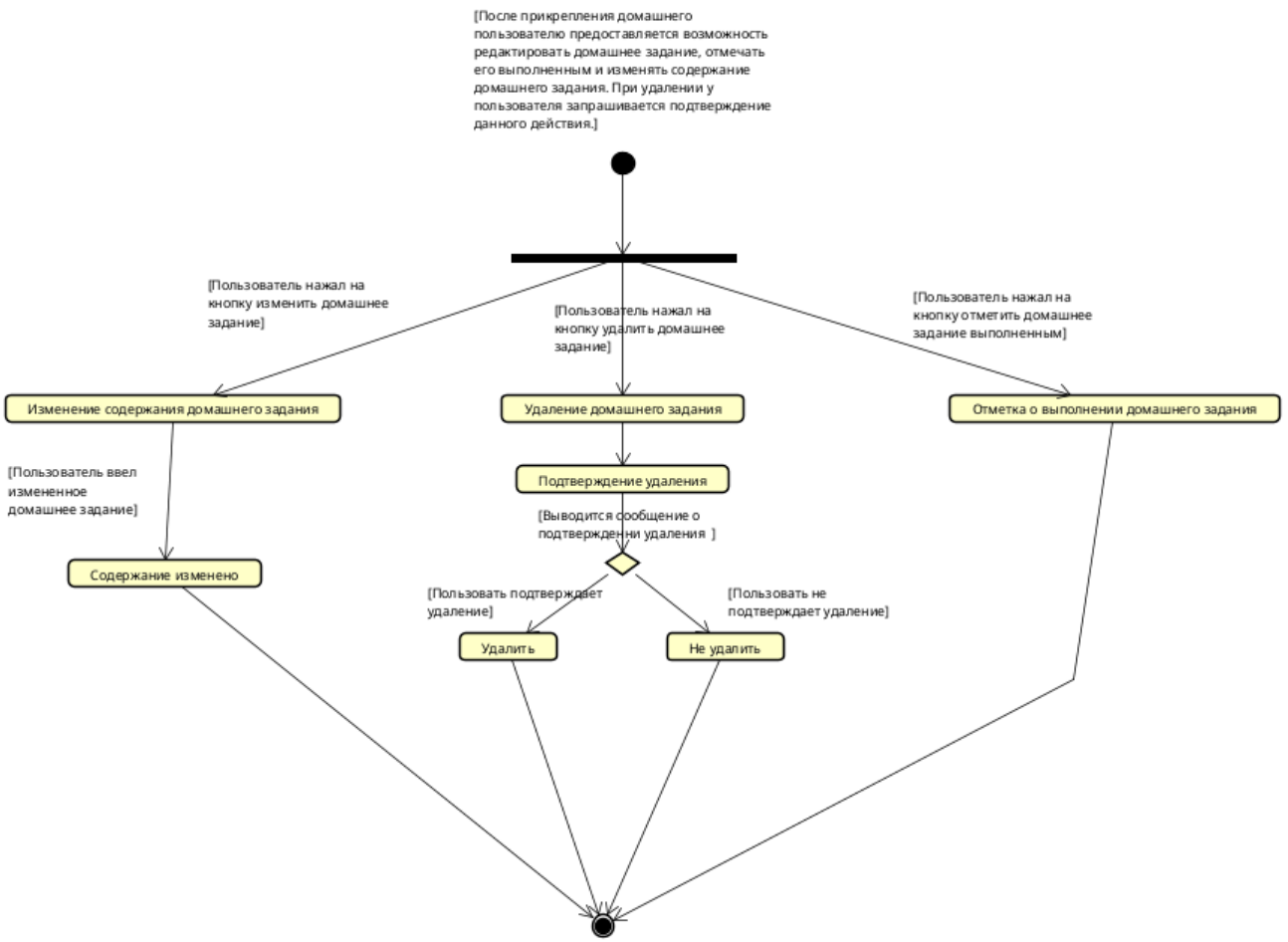


3.3. Функциональные требования для роли Авторизованный пользователь

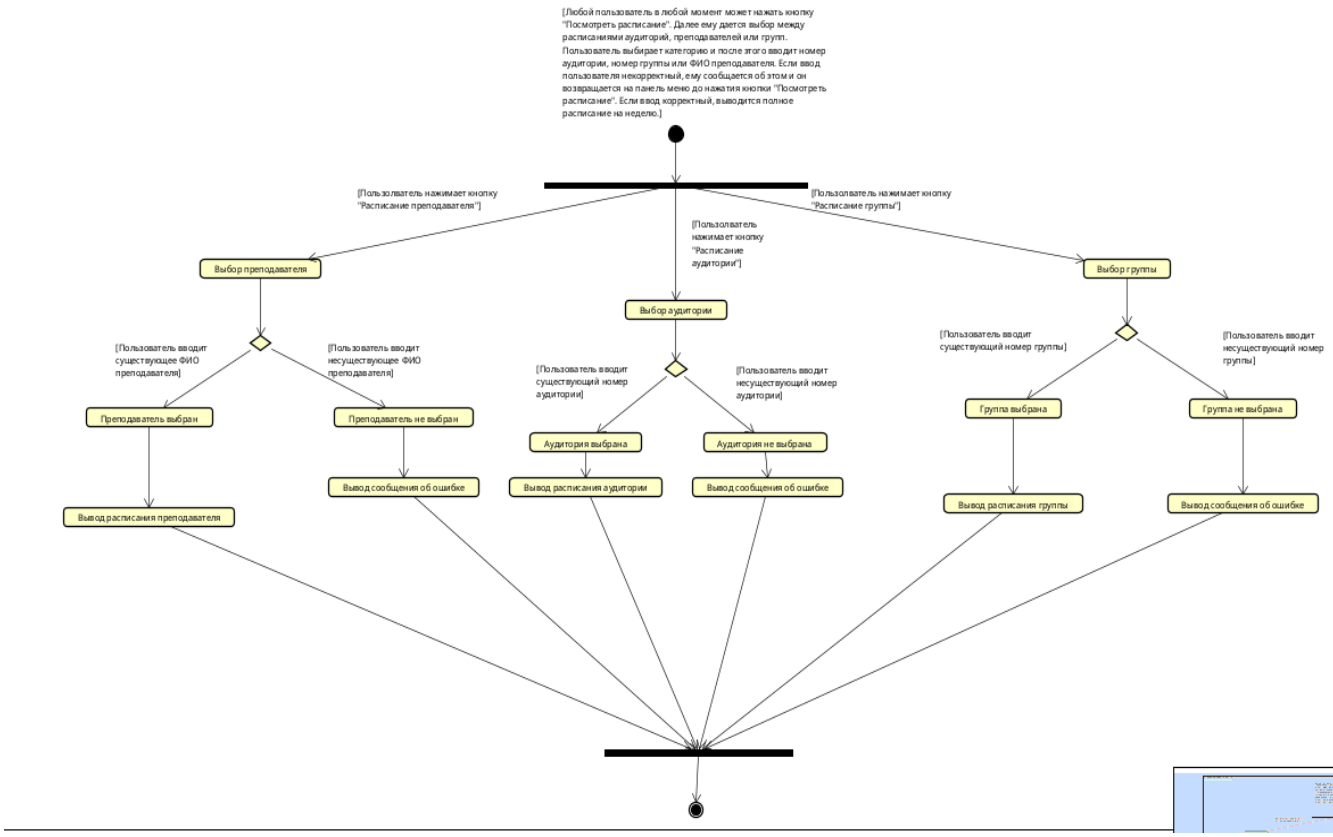
3.3.1. Прикрепление домашнего задания



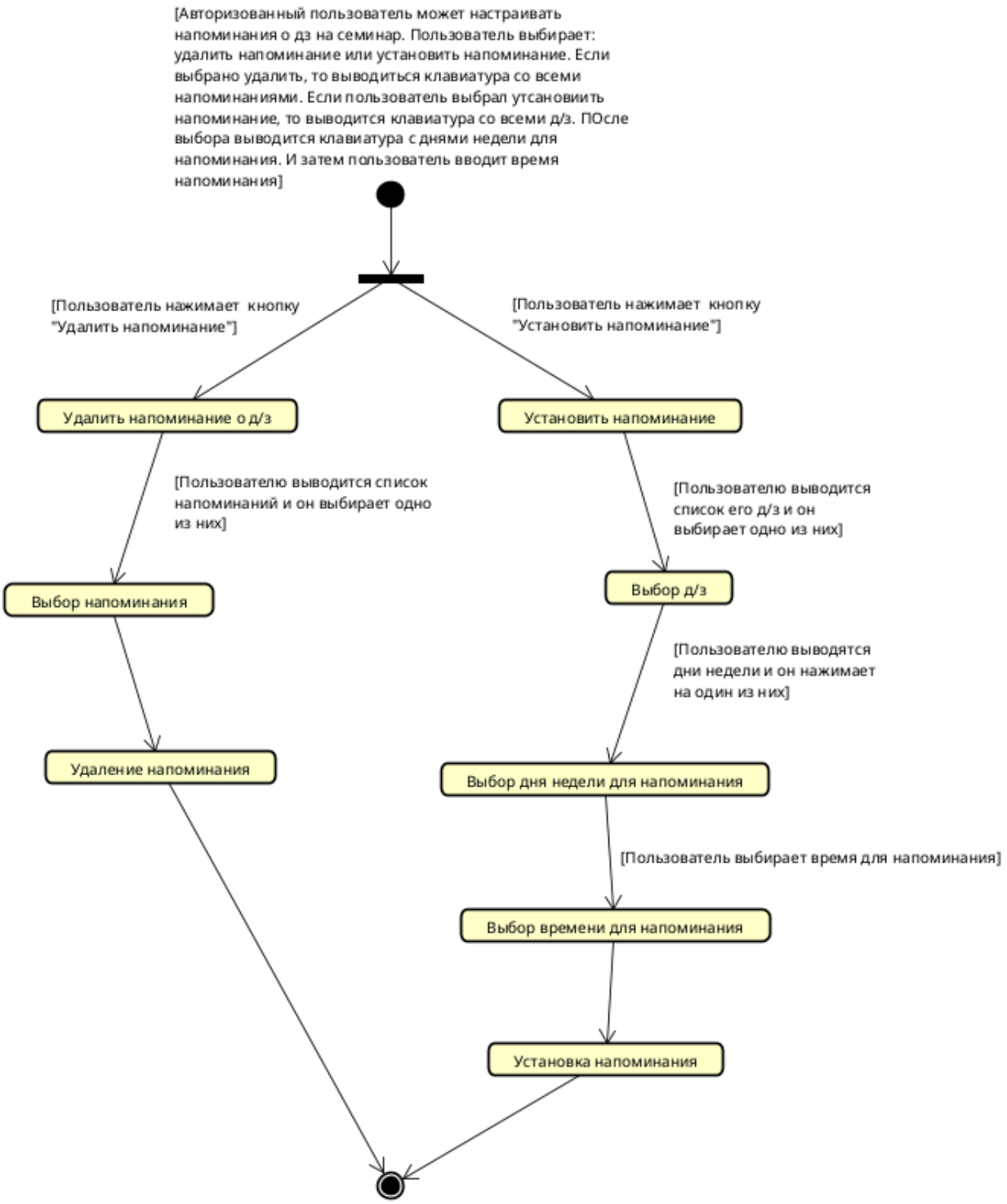
3.3.2. Редактирование домашнего задания



3.3.3. Просмотр расписания

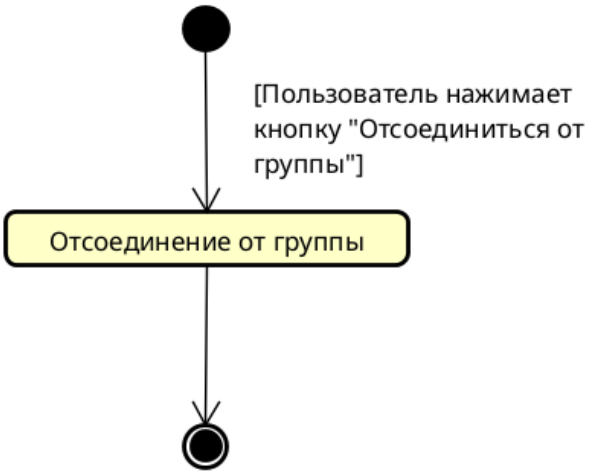


3.3.4. Настройка напоминаний о домашнем задании



3.3.5. Отсоединиться от группы

[Авторизованный пользователь
может в любой момент нажать
кнопку "Отсоединиться от группы"]



3.4. Нефункциональные требования

3.4.1. Производительность

- Бот должен иметь быстрое время ответа, не превышающее 3 секунд, чтобы обеспечить хороший пользовательский опыт.
- Система должна поддерживать обработку как минимум 20 запросов в секунду.

3.4.2. Надежность

- Бот должен быть доступен более 97% времени в сутки

3.4.3. Безопасность

- Система должна быть защищенной от атак, таких как SQL-инъекции.

3.4.4. Масштабируемость

- Система должна быть модульной

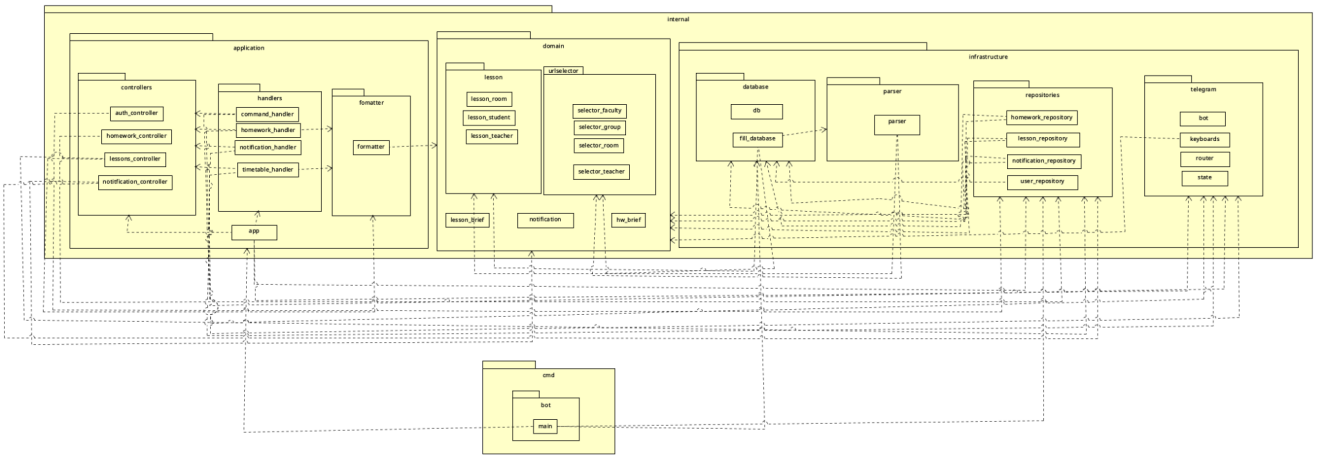
3.4.5. Документация

- Код должен быть снабжен комментариями

4. Обзор архитектуры

Этот раздел описывает архитектуру системы.

4.1.1 Компонентная модель системы



4.1.1.1 cmd

cmd – пакет, содержащий исполняемые файлы приложения.

4.1.1.2 internal

internal – пакет, содержащая внутренние пакеты приложения.

4.1.1.3 application

application - пакет, содержащий бизнес логику приложения

- controllers - пакет содержащий прослойку между handlers и инфраструктурными компонентами
- handlers - пакет содержит классы, отвечающие за связь с Telegram API и обработку сообщений.
- formatter - пакет содержит компоненты отвечающие за обработку выходных данных

4.1.1.4 domain

domain - пакет содержит модели приложения

- lessons - пакет содержит все модели отвечающие за различные виды пар в университете
- urlselector - пакет содержит модели необходимые для парсинга ссылок из расписания

4.1.1.5 infrastructure

infrastructure - пакет, в котором содержатся инфраструктурные компоненты приложения

- database - пакет содержит классы-драйверы баз данных
- parser - пакет содержит http парсер для сайта с расписанием
- repositories - пакет содержит структуры отвечающие за работу с базой данный
- telegram - пакет содержит компоненты для взаимодействия с telegram api

4.1.2 Компоненты сторонних производителей

Библиотеки:

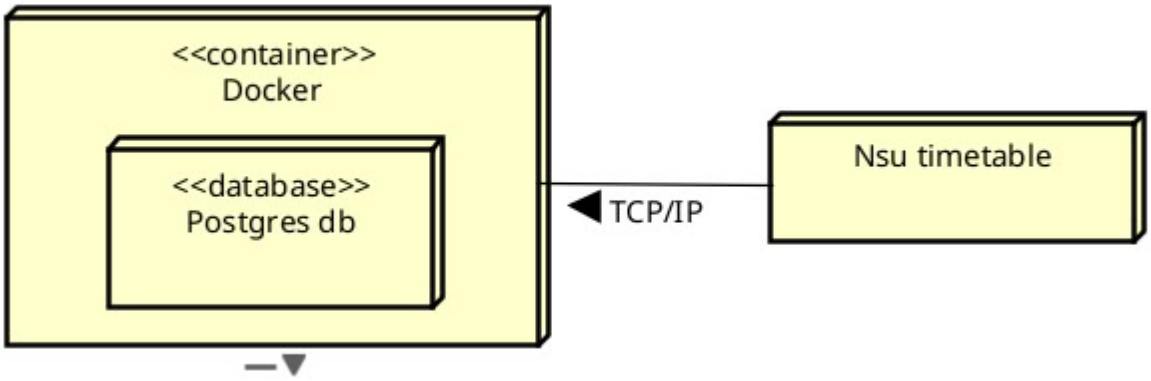
- *github.com/go-telegram-bot-api/telegram-bot-api/v5 - библиотека для упрощенного взаимодействием с Telegram API*
- *github.com/PuerkitoBio/goquery - библиотека для удобного парсинга HTML*
- *github.com/jackc/pgx/v5/stdlib - это драйвер, который даёт возможность использовать высокопроизводительный PostgreSQL-клиент pgx через стандартный интерфейс database/sql*

Программное обеспечение:

- *PostgreSQL - база данных*
- *Docker - система контейнеризации*

4.1.3 Схема развертывания приложения

Для развертывания приложения используются Docker контейнеры. Для запуска контейнеров используется Docker-compose.



Приложение состоит из двух нод : PosgreSQL Database и NSU Timetable bot. Ноды взаимодействуют по протоколу TCP/IP.
Для развертывания приложения необходим сервер/пк с минимальными требованиями:

- Жесткий диск 20 гб
- Оперативная память 4гб
- Процессор с частотой 2.4 ггц
- Установленное программное обеспечение docker, docker-compose
- Подключение к интернету

5. Допущения и ограничения

Ограничения:

- На разработку диаграмм (use-case, sequence, классов, пакетов) было применено временное ограничение в 1 месяц
- На разработку приложение было применено временное ограничение в 1.5 месяца

Допущения

- При разработке проекта принято допущение, что число транзакций в единицу времени значительно (более чем в 10 раз) снижается в ночное время, что позволяет в период с 01:00 до 6:00 производить автоматическое обновление программного обеспечения системы, требующее полной перезагрузки и остановки сервиса на период до 5 минут.

6. Известные проблемы

Ниже приводятся известные на данный момент проблемы и недоработки выработанного программного решения, а также возможные пути их устранения в последующих итерациях проекта.

6.1. Невысокая производительность приложения

Проблема	Производительность приложения экспоненциально деградирует при общем числе пользователей выше 10000 и числе одновременных сессий выше 100.
Ранг	10 (высокий)
Влияние на проект	Невозможность использования системы при числе пользователей более 10000.
Пути решения	Кластеризация веб-сервера и сервера базы данных, а также применение load balancer в точке маршрутизации запроса к веб-серверу.

6.2. Ресурсоемкая адаптация к изменению расписания

Проблема	При изменении расписания на сайте nsu.ru , приходится перезапускать бота и ждать, пока все расписание снова не загрузится в БД
Ранг	10 (высокий)
Влияние на проект	Невозможность быстрого обновления при изменениях, то есть пользователь не получает сразу актуальное расписание
Пути решения	Чтобы пользователи всегда получали самое свежее расписание без простоев, необходимо реализовать фоновую синхронизацию с сайтом nsu.ru. Бот будет периодически проверять наличие обновлений и применять их "на лету", загружая лишь те изменения, которые действительно произошли. Это полностью устранит необходимость в его перезапуске и длительном ожидании.