

# Тезисы доклада

## 1. Актуальность работы

1. Число студентов инженерных и технических направлений растёт, нагрузка на преподавателей в вузах достигла 13,3 студента на одного преподавателя (данные Минобрнауки, 2024).
2. Существующие системы (Moodle, Stepik) автоматизируют тесты, но не проверяют лабораторные работы с кодом и текстовыми пояснениями.
3. Современные большие языковые модели умеют анализировать сложные артефакты и давать персонализированную обратную связь, однако готовых российских решений для комплексной проверки лабораторных работ по информатике практически нет.

Вывод: тема актуальна, разработка такого модуля снизит нагрузку на преподавателей и повысит качество обучения.

## 2. Объект и предмет исследования

Объект: образовательный процесс в условиях цифровизации и внедрения ИИ.

Предмет: автоматизированная проверка лабораторных работ по информатике с использованием LLM и NLP.

## 3. Цель и задачи

Цель: разработка и программное внедрение в рамках платформы LumiQa модуля для автономной проверки лабораторных работ и контрольных тестов с применением языковых моделей и баз знаний.

Задачи:

1. Проанализировать существующие решения.
2. Изучить методы NLP и LLM, их возможности и недостатки.
3. Обосновать выбор инструментов.
4. Описать архитектуру и реализацию модуля.
5. Провести тестирование программного модуля.

## 4. Теоретическая база

- Рассмотрена первая ИОС SCHOLAR (1970) – предшественник современных систем.
- Проанализированы адаптивные платформы (Knewton, CogBooks, Plarion) – они персонализируют контент, но не проверяют лабораторные работы.
- Изучены близкие аналоги – AI-ассистент куратора проектного практикума и LLM-CoT-GDS (разработки ИТМО). Их ограничения: отсутствие проверки кода или оценки только текстовых отчётов.
- Описаны методы критериального оценивания (рубрики), динамического тестирования и статического анализа.
- Рассмотрены подходы к оценке текста: AES, семантическая близость, LLM-as-a-Judge с усилением через RAG.
- Приведены этапы NLP: токенизация, лемматизация, удаление стоп-слов, выделение сущностей, векторизация.
- Дан обзор архитектуры трансформеров и принципов работы LLM (обучение, возможности, ограничения, RAG).

## 5. Разработанный модуль (практическая часть)

- Модуль интегрирован в учебную платформу LumiQa (стек: TypeScript, Node.js, Next.js, Prisma, PostgreSQL).
- Архитектура построена как последовательный конвейер (pipeline): исполнитель оценки → реестр типов заданий → адаптеры → сервис проверки лабораторных работ.
- Сервис включает:
  - Загрузчик – безопасное скачивание файлов по URL (проверка хостов, блокировка локальных адресов, ограничение размера).
  - Парсеры – извлечение кода, текста, метрик из Markdown и Jupyter Notebook.
  - Загрузчик контекста – получение критериев и эталонов из БД (RAG).
  - Механизм принятия решения – эвристики (пересечение токенов, объём, количество кода), пороги и статусы (`auto\_final`, `ai\_provisional`, `awaiting\_teacher`).
- Гибридный подход: LLM (OpenAI API) используется для оценки текстовых ответов в тестах и может быть адаптирован для лабораторных работ.
- Safety path: при любых ошибках работа переводится на ручную проверку преподавателя (human-in-the-loop).

## 6. Пример функционирования

- Студент отправляет ссылку на GitHub MR, .md или .ipynb.
- Модуль загружает файл, парсит, извлекает контекст, вычисляет баллы по рубрике, формирует статус и комментарий ассистента.
- При низкой уверенности работа направляется преподавателю.

## 7. Результаты и новизна

- Разработан модуль, объединяющий:
  - проверку программного кода и текстовых пояснений;
  - критериальное оценивание с помощью рубрик;
  - интеграцию с полным циклом курса (теория + тесты + лабораторные);
  - гибридную схему с участием преподавателя.

Модуль протестирован на тестовых лабораторных работах, показал работоспособность и снижение рутинной нагрузки.

## 8. Практическая значимость

- Модуль внедрён в любую систему LumiQa.
- Методические рекомендации по использованию позволяют преподавателям быстро адаптировать систему под свои курсы, а студентам быстро разобраться в новой технологии.
- Экономия времени преподавателя, повышение объективности и оперативности обратной связи.

## 9. Заключение

- Поставленные задачи выполнены, цель достигнута.
- Разработанный модуль интегрирован в адаптивную обучающую систему LumiQa.
- Перспективы: доработка полного цикла LLM-оценки лабораторных работ, расширение на другие языки программирования, поддержка более сложных рубрик.