

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет
имени К.И.Сатпаева

Институт автоматизации и информационных технологий

Кафедра «Программная инженерия»

Садыкова Алтынай Маратовна

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

Образовательная программа: 6B06102 Computer Science

Алматы 2025

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет
имени К.И.Сатпаева

Институт автоматизации и информационных технологий

Кафедра «Программная инженерия»



ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой ПИ
канд. тех. наук, ассоц. профессор
Ф.Н. Абдолдина
« 30 » 05 2025 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

На тему: «Система оценки знаний и цифровой след обучающихся»

Образовательная программа: 6B06102 Computer Science

Выполнила

Садыкова А.М.

Рецензент

PhD, ассоц. проф.-исследователь

Тойбаева Ш.Д.

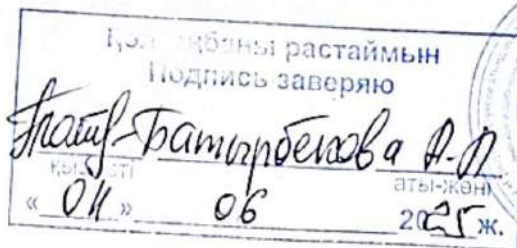
« 04 » 06 2025 г.

Научный руководитель

канд. техн. наук, профессор

Кубеков Б.С.

« 04 » 06 2025 г.



Алматы 2025

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет
имени К.И.Сатпаева

Институт автоматизации и информационных технологий

Кафедра «Программная инженерия»



УТВЕРЖДАЮ

Заведующий кафедрой ПИ
канд. тех. наук, ассоц. профессор
Ф.Н. Абдолдина

« 30 » 05 2025 г.

ЗАДАНИЕ

на выполнение дипломного проекта

Обучающемуся: *Садыкова Алтынай Маратовна*

Тема: Система оценки знаний и цифровой след обучающихся

Утверждена приказом проректора по академической работе:

№ 1804-90

	от « 25 » 11 2024 г.
Срок сдачи законченного проекта	« 05 » 06 2025 г.

Исходные данные к дипломному проекту:

А) Анализ предметной области, анализ аналогичных проектов;

Б) Разработка технического задания;

В) Разработка архитектуры ПО;

Г) Проектирование и разработка понятного и интерактивного дизайна;

Д) Разработка и тестирование ПО;

Перечень подлежащих разработке в дипломном проекте вопросов: (с точным указанием обязательных чертежей): *представлены 15 слайдах презентации.*

Рекомендуемая основная литература: *из 10 наименований.*

Алматы 2025

ГРАФИК
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю и консультантам	Примечание
1. Анализ предметной области	22.02.2025	Выполнено
2. Разработка технического задания	06.03.2025	Выполнено
3. Построение и описание задач модуля	15.03.2025	Выполнено
4. Решение задачи и проектирование архитектуры	25.04.2025	Выполнено
5. Сравнение разных подходов к архитектуре	03.05.2025	Выполнено
6. Написание пояснительной записки к дипломному проекту	04.05.2025	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект
с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты, И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись	Оценка
Программное обеспечение	Шаханов Э.Р. ст. преподаватель, магистр	29.05.25		30
Нормоконтролер	Ердалиева З.У. ст. преподаватель, магистр	29.05.25		

Научный руководитель Кубеков Б.С.

Задание принял к исполнению обучающийся Садыкова А.М.

Дата « 05 » 06 2025 г.

Алматы 2025

АҢДАТПА

Бұл бағдарламалық шешім жеке білім беру траекториялары бойынша деректерді жинауды, сақтауды және аналитикалық өңдеуді қамтамасыз ететін құзыреттілікке бағытталған бағалау тұжырымдамасын жүзеге асырады. Жүйе оқытушыға зертханалық тапсырмалар шеңберінде студенттердің қызметін құрылымдық бағалауды жүргізуге және CDIO кезеңдері арқылы үлгерімін бақылауға, әрбір қатысушының цифрлық профилін қалыптастыруға мүмкіндік береді.

Жоба барысында қолданыстағы бағалау әдістерін талдау, ақпараттық жүйеге қойылатын талаптарды ресімдеу, оңтайлы технологиялық стек таңдау, қолданбалы архитектураны модельдеу, пайдаланушы интерфейсін әзірлеу, визуализация модульдерін енгізу және PDF есеп беру міндеттері шешілді. Пайдаланылған құралдар Python тілі, Flask құрылымы, әкімші интерфейсін құруға арналған PyQt кітапханасы, SQLite деректер қорын басқару жүйесі, сонымен қатар нәтижелерді өңдеу және ұсыну үшін Pandas және Matplotlib пакеттері болды.

Дипломдық жұмыстың практикалық нәтижесі оқытушылар мен әдіскерлерге бағытталған, техникалық және инженерлік жоғары оқу орындарының оқу үдерісіне кіріктірілуі мүмкін толық функционалды веб-қосымшаны құру болды. Жүйе әртүрлі білім беру үлгілері үшін масштабтау, модификациялау және оқу орындарының жеке қажеттіліктеріне бейімделу мүмкіндігіне ие.

АННОТАЦИЯ

Разработанное программное решение реализует концепцию компетентностно-ориентированной оценки, обеспечивая сбор, хранение и аналитическую обработку данных об индивидуальных образовательных траекториях. Система позволяет преподавателю проводить структурированную оценку деятельности студентов в рамках лабораторных заданий и отслеживать прогресс по этапам CDIO, формируя цифровой профиль каждого участника.

В ходе выполнения проекта были решены задачи анализа существующих методов оценки, формализации требований к информационной системе, выбора оптимального стека технологий, моделирования архитектуры приложения, разработки пользовательского интерфейса, реализации модулей визуализации и PDF-отчётности. В качестве инструментов использованы язык Python, фреймворк Flask, библиотека PyQt для построения интерфейса администратора, система управления базами данных SQLite, а также пакеты Pandas и Matplotlib для обработки и представления результатов.

Практическим результатом дипломной работы стало создание полнофункционального веб-приложения, ориентированного на преподавателей и методистов, которое может быть интегрировано в образовательный процесс технических и инженерных вузов. Система обладает потенциалом масштабирования, модификации под различные образовательные модели и адаптации к индивидуальным потребностям учебных заведений.

ABSTRACT

The developed software solution implements the concept of competency-based assessment, ensuring the collection, storage, and analytical processing of data on individual educational trajectories. The system allows teachers to conduct structured assessments of students' performance in laboratory assignments and track progress through the CDIO stages, forming a digital profile for each participant.

During the project, the following tasks were completed: analysis of existing assessment methods, formalization of requirements for the information system, selection of the optimal technology stack, modeling of the application architecture, development of the user interface, and implementation of visualization and PDF reporting modules. The tools used were the Python language, the Flask framework, the PyQt library for building the administrator interface, the SQLite database management system, and the Pandas and Matplotlib packages for processing and presenting results.

The practical result of the thesis was the creation of a full-featured web application aimed at teachers and methodologists, which can be integrated into the educational process of technical and engineering universities. The system has the potential to be scaled, modified for different educational models, and adapted to the individual needs of educational institutions.

СОДЕРЖАНИЕ

	Введение	9
1	Исследовательско-технологическая часть	10
1.1	Анализ предметной области	10
1.2	Обзор и сравнительный анализ аналогов	12
1.3	Стандарты формирования цифрового следа обучающихся	14
1.4	Методология CDIO: концепция, проектирование, реализация	16
1.5	Матрица RACI и ее применение в учебном проекте	19
2	Проектная часть	22
2.1	Постановка задачи	22
2.2	Обоснование выбора стека технологий	24
2.3	Архитектура программной системы	26
2.4	Пользовательские сценарии и интерфейс приложения	27
2.5	Реализация системы	29
2.6	Реализация моделей CDIO и RACI в рамках системы	30
2.7	Диаграммы UML	32
3	Экспериментальная часть	35
3.1	Методика тестирования программного продукта	35
3.2	Описание тестовых сценариев и вводных данных	36
3.3	Результаты тестирования функциональных модулей	38
3.4	Анализ соответствия результатов поставленным задачам	39
	Заключение	41
	Приложение А. Техническое задание	42
	Список использованной литературы	44
	Приложение Б. Листинг (код) программы	46

ВВЕДЕНИЕ

Актуальность дипломного проекта определяется современным развитием инженерного образования, которое обуславливает необходимость пересмотра систем оценки успеваемости, с акцентом на объективную проверку и измерение профессиональных навыков. В условиях стремительного обновления требований отраслей и смещения акцента с теоретической подготовки на практико-ориентированную деятельность, особую значимость приобретает внедрение цифровых механизмов анализа проектной активности студентов. Разрабатываемое программное решение даёт возможность внедрить модели CDIO и RACI, формируя цифровой след студентов, используемый для аналитики и сопровождения их индивидуального профессионального развития.

Объектом исследования выступает система оценки учебных достижений студентов, основанная на проектной организации учебного процесса.

Предметом является проектирование и реализация веб-приложения, фиксирующего результаты деятельности студента в цифровом формате, и обеспечивающего сопоставление этих данных с установленной матрицей компетенций и ролевым распределением ответственности в проектной команде.

Цель дипломного проекта заключается в создании программной системы с графическим интерфейсом, предназначенной для регистрации проектных активностей, оценки знаний и навыков обучающихся в привязке к этапам CDIO, с возможностью визуализации индивидуального прогресса и формирования PDF-отчётности.

Для достижения поставленной цели в процессе реализации проекта необходимо решить ряд задач:

- 1) Проанализировать теоретические и методологические основы проектного обучения и соответствующие ему цифровые средства оценивания.
- 2) Сформулировать функциональные, пользовательские и технические требования к системе.
- 3) Разработать архитектуру веб-приложения и прототип интерфейса пользователя с использованием шаблонного движка Jinja2.
- 4) Реализовать серверную часть приложения на базе Flask с интеграцией ORM SQLAlchemy и базы данных SQLite.
- 5) Внедрить в систему модель компетенций и матрицу RACI.
- 6) Обеспечить возможности графической визуализации данных и генерации PDF-отчётов средствами Matplotlib и ReportLab.
- 7) Провести тест и оценку системы на её функциональную пригодность.
- 8) Подготовить комплект технической и пользовательской документации.

Ожидаемым результатом выполнения проекта предполагается создание программного комплекса, обеспечивающего прозрачную, системную и аналитически подкреплённую оценку образовательной деятельности студентов исходя из их компетенций, способствующее индивидуальному обучению и более тесной взаимосвязи между учениками и преподавателями.

1 Исследовательско-технологическая часть

1.1 Анализ предметной области

Современное инженерное образование, ориентированное на подготовку выпускников, обладающих комплексом профессиональных и надпрофессиональных компетенций, сталкивается с необходимостью пересмотра устоявшихся парадигм оценки учебных достижений. В условиях растущей значимости практико-ориентированных форм обучения, в первую очередь – проектной деятельности, оценивание студенческой активности требует выхода за рамки традиционной балльно-рейтинговой системы, направленной преимущественно на фиксацию факта усвоения теоретического материала. Это обуславливает потребность в формировании многоаспектных моделей оценивания, способных учитывать как индивидуальный вклад, так и динамику командного взаимодействия.

Цифровизация образовательной среды трансформирует традиционную роль обучающегося, превращая его из пассивного потребителя информации в активного участника, ответственного за развитие собственных профессиональных траекторий. В условиях информационной насыщенности и быстрой смены технологий востребован не абстрактный объём знаний, а способность применять их в условиях неопределённости, нестандартных задач и командного взаимодействия. Концепция деятельностного подхода приобретает методологическую значимость, так как акцентирует внимание на деятельности как источнике развития компетенций, включая критическое мышление, самоорганизацию, принятие решений и рефлексию над результатами.

Среди наиболее распространённых подходов к оценке проектной деятельности можно выделить условно три группы: **индикаторные модели, экспертные системы и гибридные методы**, сочетающие количественные и качественные параметры.

Индикаторные модели базируются на заранее определённом наборе формализованных критериев, включающих параметры качества выполнения проекта (соответствие техническому заданию, соблюдение сроков, полнота реализации), структурные характеристики (наличие проектной документации, обоснованность проектных решений) и степень достижения запланированных результатов. Как правило, такие модели реализуются через рубрикаторы, чек-листы или матрицы оценивания. Преимущество данного подхода заключается в его объективности и воспроизводимости, однако он зачастую не отражает полноту когнитивной и социальной составляющей учебной деятельности.

В рамках **экспертных систем** оценка осуществляется преподавателем, куратором или сторонними специалистами на основе наблюдения, анализа представленных артефактов (отчётов, прототипов, презентаций) и самоанализа обучающихся. Такой подход обеспечивает высокий уровень

контекстуализации, позволяет учитывать креативность, инициативность и рефлексивность студентов, однако страдает от субъективизма и высокой зависимости от квалификации эксперта. Для минимизации этих ограничений используются шкалы Блума, дескрипторы уровней освоения компетенций и методы многоуровневой экспертной верификации.

Гибридные подходы опираются на интеграцию количественных метрик и экспертных заключений, что позволяет формировать многомерную оценочную модель, учитывающую как поведенческие, так и содержательные аспекты проектной деятельности. Одной из наиболее перспективных моделей в этом контексте является **CDIO-подход (Conceive – Design – Implement – Operate)**, разработанный Массачусетским технологическим институтом. Он предполагает поэтапное оценивание студента на всех стадиях проектного цикла, фиксируя его вовлечённость, степень инициативы, умение работать в команде, а также способность к самостоятельному решению инженерных задач. CDIO сопровождается матрицей компетенций, согласованной с требованиями профессиональных стандартов, и позволяет осуществлять системное и непрерывное оценивание.

Каждый из этапов CDIO сопряжён с конкретными видами деятельности обучающегося: от анализа проблемной ситуации и проектной концептуализации до внедрения продукта и оценки его эффективности. Структурная логика CDIO выстраивает целостную педагогическую архитектуру, в которой теория не отделяется от практики, а знание формируется через решение инженерных задач.

Особую значимость в оценке проектной деятельности приобретает **модель ролевого распределения ответственности RACI (Responsible – Accountable – Consulted – Informed)**, активно применяемая в управлении проектами. Интеграция RACI в образовательный контекст даёт возможность более точно дифференцировать вклад каждого участника команды, определить зоны ответственности и уровни принятия решений. Это особенно важно в ситуациях, когда коллективный результат затрудняет идентификацию индивидуального вклада студентов.

Значительный потенциал в системе подготовки кадров принадлежит концептуализации компетентностной модели. Компетенции рассматриваются не как набор абстрактных умений, а как интегральная характеристика, отражающая способность к применению знаний в конкретных условиях, к межличностному взаимодействию и к профессиональному суждению. Применение компетентностного подхода требует наличия инструментов для объективной оценки сформированности компетенций, отслеживания их развития и визуализации вклада каждого участника проектной группы. Именно в этом контексте возникает потребность в цифровых решениях, позволяющих собирать, обрабатывать и анализировать данные, генерируемые в процессе проектной работы обучающегося.

Наряду с этим, в последние годы всё более востребованными становятся **цифровые платформы**, позволяющие не только организовать проектную

деятельность в онлайн-среде, но и собирать разносторонние данные о действиях студентов – от активности в задачах до взаимодействия с сокурсниками и кураторами. Такие данные составляют **цифровой след**, который может быть использован для построения аналитических моделей, прогнозирования успеваемости, выявления образовательных дефицитов и персонализации обучения. Примерами таких платформ выступают Learning Management Systems (LMS), обладающие функциональностью сбора логов, трекинга активности и встроенной системы оценивания (например, Moodle, Canvas, Google Classroom).

В результате современные подходы к оценке проектной деятельности представляют собой совокупность инструментов, ориентированных на комплексное фиксирование как результатов, так и процессов, ведущих к ним. Выбор той или иной модели обусловлен целями образовательной программы, уровнем подготовки обучающихся, спецификой реализуемых проектов и доступными средствами аналитики. Наиболее эффективными на текущем этапе признаны гибридные модели, интегрирующие методологию CDIO, элементы RACI и инструменты цифровой аналитики, позволяющие осуществлять объективную, прозрачную и многомерную оценку проектной активности обучающихся в логике развития их профессиональных компетенций.

1.2 Обзор и сравнительный анализ аналогов

Разработка информационных систем в сфере образования требует опоры на существующий опыт и анализ функциональных решений, представленных на рынке. В контексте создания системы оценки знаний на основе проектного обучения и модели компетенций, актуальным представляется проведение сравнительного анализа программных платформ, применяемых для сопровождения образовательного процесса, контроля успеваемости и управления проектной деятельностью обучающихся. Данный анализ позволяет обоснованно выявить функциональные пробелы, определить степень инновационности предлагаемой разработки и уточнить направления проектной реализации.

На современном этапе можно выделить три ключевых класса программных решений, отчасти пересекающихся с поставленными задачами:

- 1) Системы управления обучением (Learning Management Systems, LMS)
- 2) Платформы для управления проектами (Project Management Systems, PMS)
- 3) Специализированные цифровые среды оценки компетенций и профессионального развития (например, e-portfolio-системы).

К наиболее известным **LMS-платформам** относятся **Moodle, Blackboard, Canvas и Google Classroom**. Эти системы обеспечивают широкие возможности по организации курсов, управлению учебным контентом, коммуникации между

участниками образовательного процесса и фиксации оценочной информации. Однако их функциональность преимущественно ориентирована на линейное представление учебного материала, выполнение индивидуальных заданий и контроль текущей успеваемости. В них, как правило, отсутствует развитый инструментарий для поддержки командных проектов, проектной методологии CDIO и распределения ролей в групповой работе. Кроме того, традиционные LMS-системы не обладают встроенными средствами фиксации и визуализации цифрового следа обучающегося, основанного на анализе его компетентностного роста в ходе выполнения этапов проекта.

Второй класс решений – **платформы управления проектами**, это **Trello, Jira, Asana, ClickUp**, – предоставляют мощные средства планирования, делегирования задач, отслеживания прогресса и координации команды. Их интерфейс и логика взаимодействия ориентированы на корпоративные процессы и продуктовые команды. Несмотря на визуальную наглядность и гибкость настройки, данные системы не адаптированы под образовательный контекст, не содержат механизмов оценки компетенций, не учитывают педагогическую специфику, а также не обеспечивают связи между этапами проекта и образовательными целями, особенно в контексте CDIO.

Третий сегмент – **системы цифровых портфолио и индивидуального развития** (например, **Mahara, PebblePad, EduBrite**), – предоставляет возможность фиксировать достижения обучающегося, собирать артефакты проектной деятельности, формировать индивидуальные траектории. Однако и здесь отсутствует тесная интеграция с командной работой, проектной логикой и RACI-структурами. Большинство e-portfolio-систем сконцентрированы на сборе данных и визуализации прогресса, но не предоставляют функционала для распределения ролей, проектного управления или автоматизированной аналитики по этапам CDIO-модели.

Следующим образом, проведённый анализ свидетельствует о наличии функциональных и концептуальных ограничений в каждом из перечисленных классов решений. Ни одна из существующих систем не объединяет в себе следующие ключевые компоненты:

- поэтапную модель проектной деятельности CDIO;
- фиксацию и анализ индивидуальных компетенций обучающегося на каждом этапе выполнения проекта;
- распределение ролей и ответственности с использованием модели RACI;
- визуализацию цифрового следа и аналитические отчёты с использованием встроенных инструментов обработки данных.

Проектируемая система направлена на устранение указанных ограничений и формирование интегрированного решения, сочетающего элементы LMS, PMS и e-portfolio в рамках единой образовательной среды. Её уникальность заключается в синтезе проектного подхода, компетентностной модели и аналитического сопровождения, что обеспечивает высокий уровень адаптации под требования инженерного образования и цифровых стандартов подготовки кадров.

1.3 Стандарты формирования цифрового следа обучающихся

Понятие цифрового следа в контексте образовательной среды представляет собой совокупность зафиксированных данных о действиях, результатах и прогрессе обучающегося в цифровой образовательной среде, поддающихся аналитической обработке с целью оценки компетенций, идентификации дефицитов знаний и прогнозирования образовательной траектории. Ввиду цифровизации высшего образования и активного внедрения информационно-коммуникационных технологий, цифровой след становится неотъемлемым компонентом систем персонализированного обучения, оценки качества образования и обеспечения прозрачности образовательного процесса.

Концептуальная значимость цифрового следа заключается в его способности представлять эволюцию профессионального становления личности на основании объективно зафиксированных показателей. В отличие от классических форм контроля, основанных на выборочных проверках и итоговой аттестации, цифровой след отражает весь континуум образовательной активности, позволяя учитывать не только результат, но и процесс его достижения. Это открывает возможности для более тонкой калибровки образовательных траекторий, персонализации сопровождения и предотвращения академических рисков на ранних этапах.

Механизмы сбора цифрового следа реализуются через интеграцию обучающих платформ с системами регистрации действий пользователя, что позволяет осуществлять сбор информации о ходе проектной работы, времени, затраченном на выполнение заданий, частоте взаимодействий с интерфейсом, результативности командного участия и многом другом. Полученные данные поддаются дальнейшей интерпретации, агрегированию и визуализации, формируя целостный аналитический портрет обучающегося, доступный для преподавателя, наставника и самого участника образовательного процесса.

Принципиальным моментом является то, что цифровой след способен функционировать как основа для рефлексивного анализа со стороны самого обучающегося. Наличие доступной визуализации прогресса стимулирует развитие навыков самооценки, формирует мотивационную направленность на достижение образовательных целей и способствует осознанию личной ответственности за траекторию профессионального развития. Кроме того, цифровой след может служить компонентом портфолио выпускника, демонстрируя не декларативно заявленные, а реально проявленные компетенции, подкреплённые данными из образовательной среды.

Особую актуальность цифровой след приобретает в системах проектного обучения, основанных на модели CDIO, где фиксируются не только конечные результаты, но и все этапы деятельности, включая коллективное взаимодействие, роль в команде, соблюдение сроков, инициативность и вклад в общее дело. Такая многоуровневая аналитика позволяет точно оценить степень освоения компетенций, сформулированных на основе образовательных

стандартов, а также выявить скрытые потенциалы и области дефицита, неочевидные в рамках стандартных методов контроля.

Нарастающее внимание к цифровому следу в контексте инженерного образования обусловлено необходимостью оперативного получения достоверной информации о качестве подготовки, адаптивности обучающихся к изменяющимся условиям, способности к проектной деятельности и командному взаимодействию. В условиях высокотехнологичной образовательной среды цифровой след становится не просто технологическим элементом, а интегральной частью образовательного процесса, способной выступать в роли индикатора компетентностного роста, инструмента педагогической диагностики и средства повышения качества подготовки выпускников.

На сегодняшний день не существует единого международного стандарта, регламентирующего структуру, методы и инструменты формирования цифрового следа обучающегося, однако в образовательной практике широкое распространение получили методологические подходы и рекомендации, выработанные в рамках инициатив международных организаций, таких как IEEE, ISO/IEC, IMS Global Learning Consortium и др.

Одним из ключевых документов в данной области выступает стандарт **IEEE 1484.1 (Learning Technology Standards Committee – LTSC)**, в котором вводится концепция Learning Record Store (LRS) и технологии xAPI (Experience API). Данный подход позволяет фиксировать любые действия обучающегося в электронной образовательной среде – от прохождения тестов и взаимодействия с образовательным контентом до участия в командных проектах. Особенностью xAPI является ориентация на событийную модель, где каждое действие представляется в формате "Актор – Глагол – Объект", что обеспечивает гибкость при интерпретации образовательной активности и её последующем анализе.

Важной методологической основой также является концепция **Learning Analytics**, сформулированная в рамках ISO/IEC 20748, ориентированная на сбор, анализ и интерпретацию цифровых данных с целью поддержки принятия педагогических решений и оптимизации образовательных стратегий. Данный стандарт предлагает структуру метаданных и формализованные способы представления информации о результатах обучения, взаимодействии с контентом, социальной активности и прогрессе обучающегося.

Особую значимость в данном контексте приобретает использование CDIO-подхода (Conceive – Design – Implement – Operate), в рамках которого цифровой след должен не просто отражать факт выполнения задания, но и фиксировать качественные показатели участия студента на каждом из этапов проектного цикла. В дополнение к этому, применение модели RACI (Responsible – Accountable – Consulted – Informed) для описания ролевого распределения в проектной группе требует от системы цифрового следа способности учитывать контекст коллективной ответственности, что накладывает дополнительные требования к структуре и логике хранения

данных.

Современный стандарт цифрового следа обучающегося включает в себя не только фактологические данные об активности, но и формализованные индикаторы освоения компетенций, описания ролевых функций, параметры командного взаимодействия и индивидуальной ответственности. Такой подход требует интеграции гетерогенных источников данных (систем управления обучением, модулей оценки, взаимодействий в команде), а также разработки логической модели, обеспечивающей интероперабельность и соответствие требованиям образовательной аналитики.

Разработка программного обеспечения, способного фиксировать и интерпретировать цифровой след в соответствии с вышеуказанными принципами, требует создания унифицированной схемы данных, поддержки адаптивных форматов отчетности и обеспечения устойчивой связи между показателями цифрового следа и компетентностной моделью образовательной программы. В среде растущей сложности образовательных процессов и необходимости адаптации к индивидуальным траекториям, стандартизированный цифровой след становится необходимым элементом современной образовательной экосистемы, обеспечивающим как объективную оценку, так и прозрачность всей траектории профессионального становления обучающегося.

1.4 Методология CDIO: концепция, проектирование, реализация

CDIO-модель рассматривается как целостная методология трансформации инженерного образования, представляющая собой совокупность универсальных принципов проектирования образовательных программ, построения материально-технической инфраструктуры, а также организации системы отбора и повышения квалификации преподавательского состава. В её основе лежит ориентация на обучающегося как активного субъекта образовательного процесса, в том числе интеграция учебной деятельности с реальным производственным контекстом, что позволяет устранить разрыв между академической подготовкой и профессиональной практикой.

Формирование концепции CDIO восходит к концу 1990-х годов, когда в Соединённых Штатах Америки стали активно звучать критические оценки в адрес традиционного инженерного образования, не способного в полной мере отвечать на запросы индустрии. Ответом на это стало движение за реформу инженерной подготовки, в рамках которого представители академического и профессионального сообществ инициировали разработку нового подхода. В 2000 году официальное становление сообщества CDIO произошло благодаря совместной инициативе Массачусетского технологического института, Технологического университета Чалмерса, Линчёпингского университета и

Королевского технологического института Швеции.

Проектно-ориентированный подход, реализуемый в рамках этапов **CDIO (Conceive, Design, Implement, Operate)**, должен рассматриваться как методологический контекст образовательной среды, в которой обеспечиваются условия для целенаправленного формирования ключевых компонентов профессиональной компетентности будущего инженера. Именно в этой среде происходит выявление и удовлетворение индивидуальных образовательных потребностей обучающегося, а также выстраивается логически последовательная и адаптивная структура учебных мероприятий, соотносимых с этапами проектной деятельности и уровнем подготовки студента.

Интеграция CDIO-подхода в университетскую образовательную среду рассматривается как один из ключевых компонентов модернизации подготовки инженерных кадров. Это предполагает не только переосмысление содержания образовательных программ и их ориентацию на практическую значимость, но и создание среды, способствующей развитию личностной автономии студентов, формированию академической ответственности и системной профессионализации преподавателей. Вследствие, CDIO становится основой для перехода к личностно-ориентированной модели образования, способной обеспечить подготовку специалистов, востребованных в условиях высокотехнологичного производства.

Центральной задачей данной инициативы выступает повышение качества инженерного образования посредством приближения учебного процесса к требованиям современного рынка труда, обеспечивая тем самым соответствие выпускаемых специалистов ожиданиям индустрии.

Реализация CDIO-модели основывается на формировании у обучающихся прочных технических знаний, актуальных в условиях инженерной практики, через последовательное освоение стадий планирования, проектирования, производства и применения инженерных решений. Образование в рамках данной модели ориентировано на достижение трёх ключевых целей:

- 1) способность выпускников применять технические знания в реальных профессиональных условиях;
- 2) умение организовывать и руководить процессами проектирования, создания и эксплуатации инженерных объектов и систем;
- 3) осознание социального и этического измерения научно-технического прогресса и его воздействия на общество в целом.

Профессор Эдвард Ф. Кроули – специалист в области аэронавтики, астронавтики и инженерных систем Массачусетского технологического института, а также один из основателей инициативы CDIO – подчёркивает, что стратегической задачей современного инженерного образования является формирование в университетах образовательной среды, благоприятной для теоретического и практического усвоения знаний. Такая среда должна обеспечивать не только развитие профессиональных навыков, но и их последующее применение в условиях, приближённых к реальной инженерной деятельности.

Комплексная методология выявления образовательных потребностей и определения необходимых ресурсов составляет концептуальную основу для формулирования планируемых результатов освоения инженерных образовательных программ. Эта методология находит своё воплощение в структуре документа CDIO Syllabus, являющегося центральным компонентом модели CDIO и задающего вектор проектирования образовательного контента.

CDIO Syllabus опирается на стандарты CDIO Standards, в рамках которых результаты обучения представлены в виде формализованного перечня компетенций, отражающих профессиональные и личностные качества, необходимые современному инженеру. Формирование данного перечня осуществляется на основании системного анализа требований профессионального сообщества, а также экспертной оценки со стороны работодателей и участников инженерных образовательных программ. Итак, CDIO Syllabus представляет собой результат обобщения актуальных квалификационных ожиданий, транслируемых в структуру образовательного процесса.

CDIO Syllabus компетенции выполняют функцию опорной схемы для определения целей обучения, моделирования образовательных траекторий, проектирования содержания дисциплин и разработки механизмов оценивания учебных достижений. Они обеспечивают преемственность между образовательной деятельностью и результатами обучения, поскольку компетенции интерпретируются как инструментализированное выражение знаний и умений, закрепляющихся в деятельности обучающегося.

Модель компетенций в логике CDIO трактуется как многокомпонентная структура, включающая профессиональные, базовые и дополнительные компетенции, которые необходимо сформировать в процессе выполнения учебных проектов. Такая модель обеспечивает не только системность инженерной подготовки, но и возможность её адаптации под конкретные задачи и особенности образовательных программ.

В целях адекватного отражения профессиональной специфики образовательной программы по направлению «Программная инженерия» в модели CDIO применяется комплексное содержание, основанное на нескольких взаимодополняющих источниках:

1) Во-первых, используется классификация, предложенная в рамочной модели SWEBOK (Software Engineering Body of Knowledge), которая определяет базовый объем знаний по ключевым аспектам программной инженерии.

2) Во-вторых, учитываются этапы и стандарты CDIO, транслирующие перечень квалификационных требований к специалистам ИТ-профиля. В-третьих, привлекаются модели компетенций, адаптированные под каждый этап CDIO, и включающие как профессиональные, так и общекультурные компоненты.

Кроме того, применяется методика семантического анализа информационно-образовательных ресурсов, направленная на выявление,

формализацию и структурирование ключевых понятий, а также онтологическое моделирование связей между ними. Подобная методология позволяет конструировать содержательные рамки дисциплин на основании логико-семантических связей между знаниями и компетенциями, обеспечивая тем самым когерентность образовательного контента.

Фундаментальная идея CDIO заключается в том, что подготовка инженера должна охватывать весь спектр этапов жизненного цикла продукта, начиная с момента его концептуализации и заканчивая эксплуатацией и модернизацией. Образовательная модель, базирующаяся на принципах жизненного цикла, служит системной рамкой для формирования инженерных компетенций, способных отвечать на вызовы современной технологической среды.

Этап **Conceive** (планирование и осмысление) включает определение потребностей целевой аудитории, анализ нормативных требований и рыночных условий, а также разработку концепции, технических и бизнес-планов. **Design** (проектирование) охватывает подготовку проектной документации, выбор архитектурных решений и моделирование сценариев внедрения. На стадии **Implement** (реализация и внедрение) осуществляется практическое выполнение проекта: программирование, сборка, верификация, тестирование и оформление результатов. Заключительная стадия **Operate** (эксплуатация и сопровождение) предполагает внедрение продукта в реальную среду, его поддержку, масштабирование, обновление и, при необходимости, вывод из эксплуатации.

Отсюда следует, что CDIO не только задаёт логику проектного цикла, но и конституирует образовательную среду, в рамках которой осуществляется формирование инженерного мышления, практического мастерства и способности к профессиональному самоопределению. Философия CDIO выступает как структурно-содержательный ориентир, формирующий культурный контекст инженерного образования и обеспечивающий интеграцию академического знания с реальными практиками индустрии.

1.5 Матрица RACI: применение в учебном проекте

С целью обеспечения эффективного управления коллективной деятельностью студентов в рамках учебного проекта, выстраиваемой согласно логике методологии CDIO, предлагается использование матрицы компетенций. Ее применение предполагает соблюдение ряда ключевых организационно-методических условий, критичных для удачного достижения проектных целей.

Матрица RACI является инструментом проектного управления, предназначенный для чёткого разграничения функциональных ролей участников учебного проекта. Название данной методики образовано от начальных букв четырёх ключевых ролей: **Responsible** (исполнитель), **Accountable** (ответственное лицо), **Consulted** (консультант), **Informed**

(информируемое лицо). А в научной и деловой литературе она известна как матрица разделения ответственности (Responsibility Assignment Matrix, RAM) или же матрица проектного назначения (Project Assignment Matrix, PAM).

Применение данной модели позволяет сформировать последовательный и управляемый процесс взаимосвязи участников проекта, при котором каждый студент осознаёт границы своей ответственности, степень включённости и характер содействия с другими участниками команды. Соответственно, достигается организационная упорядоченность и также стимулируется личная обязанность по достижению учебных целей.

По своей форме матрица RACI состоит из табличной схемы, где для каждой задачи/этапа проекта указано, какие участники выполняют соответствующие функции. Этот подход особенно ценен в образовательной среде, где значительно обеспечить чёткое понимание ролей, исключить повторение усилий и рационализировать распределение нагрузки между студентами.

Рассмотрим подробнее функции, обозначенные данной матрицей. Роль, обозначенная буквой **R (Responsible)**, предполагает непосредственное исполнение задания. Исполнитель – это лицо, на которое возлагается практическая реализация поставленной задачи в рамках согласованных параметров и сроков. Допускается наличие нескольких исполнителей на одну задачу, однако при этом необходимо учитывать баланс нагрузки и детализированное распределение обязанностей по этапам реализации проекта.

Функция **A (Accountable)** относится к ответственному лицу, которое осуществляет общее руководство выполнением задачи. В отличие от роли исполнителя, данная функция не может быть делегирована, поскольку предполагает принятие решений, контроль сроков и качества выполнения, включая также обеспечение соответствия результатов установленным стандартам. В целях повышения эффективности проектной деятельности рекомендуется назначать на данную позицию одного участника, обладающего необходимыми полномочиями и уровнем компетентности.

Категория **C (Consulted)** включает лиц, обладающих экспертными знаниями и компетенциями, способными оказать методологическую или содержательную поддержку. Консультанты играют ключевую роль в обеспечении информационной насыщенности проекта, предоставляя консультации, дополнительные разъяснения и доступ к необходимым ресурсам. В зависимости от специфики задания и его сложности возможно привлечение одного или нескольких консультантов.

Наконец, функция **I (Informed)** отводится заинтересованным сторонам, которым необходимо регулярно предоставлять информацию о ходе выполнения проекта. К данной категории могут относиться, в частности, представители образовательной организации, научные руководители или внешние партнёры – например, представители IT-компаний. Информирование таких лиц обеспечивает прозрачность проектной деятельности, способствует укреплению доверия и гарантирует соответствие получаемых результатов ожиданиям всех

заинтересованных сторон.

Важным методологическим принципом является включение компетенций всех четырёх групп в модель, предназначенную для управленческих ролей, тогда как для исполнителей линейного уровня достаточным является формирование компетентностного профиля на базе первых трёх категорий. В следствии матрица компетенций будет представлять собой совокупность моделей, в которой каждая из них включает три или четыре взаимодополняющие группы компетенций, отражающие нрав и уровень вовлеченности участника в коллективную деятельность.

В чем необходимость матрицы компетенции?

Прежде всего, матрица компетенций описывается как инструмент, позволяющий системно визуализировать распределение необходимых профессиональных навыков среди участников учебного проекта. Её заполнение позволяет получить целостную картину существующих компетенций у студентов, обнаружить недостатки в их развитии, а также обусловить назначения дальнейшего профессионального развития.

Во-вторых, применение матрицы содействует формированию у студентов навыков самооценки, критической рефлексии и адекватного соотнесения своих действий с профессиональными стандартами. Анализ структуры распределения ролей и соответствия личных характеристик требованиям конкретных функций способствует выявлению точек роста, активизирует процесс самопознания и инициирует устойчивую мотивацию к личностному и профессиональному развитию.

В-третьих, функциональность матрицы проявляется и в оптимизации процесса подбора участников проектных групп. При формировании новой команды или перераспределении ролевых функций в действующей структуре, она позволяет избежать дублирования процессов описания требований и профилей. Следовательно, достигается стандартизированность отбора и возможность объективного сопоставления участников по критериям компетентностного соответствия, что особенно важно при необходимости оперативного формирования эффективной команды.

В-четвёртых, матрица компетенций создаёт основу для формирования индивидуальных образовательных траекторий, направленных на развитие студентов. На базе анализа заполненной матрицы возможно формирование персонализированных планов развития, учитывающих текущие показатели и потенциальные направления роста. В рамках реализации проекта становится возможным гибко перераспределять роли между участниками с учётом их компетенционного профиля, что содействует более эффективной командной работе, и стимулирует личностное, профессиональное развитие студентов.

И в пятую очередь, матрица компетенций выполняет функцию цифрового портфолио, отражающего индивидуальные достижения студента в ходе образовательного процесса. В динамике реализации учебного проекта формируется цифровой след, фиксирующий прогресс обучающегося по различным направлениям деятельности.

2 Проектная часть

2.1 Постановка задачи

В рамках трансформации образовательного пространства, ориентированной на формирование устойчивых профессиональных компетенций и развитие проектного мышления у обучающихся, возникает необходимость в создании интеллектуальных систем, обеспечивающих всестороннюю поддержку процессов оценки учебной деятельности. В частности, актуальной становится задача проектирования и реализации программного комплекса, способного регистрировать, структурировать и интерпретировать цифровые следы студента, отражающие как содержательную сторону освоения дисциплины, так и ролевое поведение в командной проектной деятельности.

Целью проектной работы выступает разработка веб-приложения, реализующего функциональность анализа проектной активности студентов на основе CDIO-подхода, с одновременным сопоставлением индивидуальных результатов с матрицей компетенций и ролевой моделью RACI. Таким образом, проект направлен на создание инструмента для преподавателя и студента, позволяющего фиксировать этапы вовлечённости в проект, проводить многоаспектную оценку и визуализировать траекторию развития обучающегося в цифровом формате.

Постановка задачи требует формализации структуры данных, а также выделения ключевых компонентов программного обеспечения, каждый из которых выполняет строго определённые функции в рамках общей архитектуры системы. Прежде всего, необходимо определить круг **функциональных требований**, реализация которых обеспечит достижение поставленной цели.

1) Учётная и ролевая модель пользователей. Система должна предусматривать регистрацию преподавателей (инструкторов), обладающих возможностью оценки студентов. Поскольку проект ориентирован на однопользовательский режим (на этапе прототипа), авторизация может быть временно отключена, но архитектура должна допускать её активацию при необходимости. Каждому студенту назначается уникальный идентификатор, указывается принадлежность к группе, а также формируются записи по каждому лабораторному заданию с фиксацией полученных баллов, комментариев и этапа CDIO.

2) Управление лабораторными работами и критериями оценки. Приложение должно позволять преподавателю создавать лабораторные работы, привязывая к ним набор критериев, соответствующих определённым компетенциям. Каждый критерий должен обладать уникальным кодом, кратким описанием, максимальной оценкой и быть связан с конкретным CDIO-этапом. Такая детализация необходима для последующего анализа степени освоения

компетенций и ролевого вклада в проект.

3) Модуль оценивания и комментариев. Оценка деятельности студента осуществляется по выбранной лабораторной работе и включает выставление баллов по каждому критерию, а также, при необходимости, добавление текстовой обратной связи. При этом формируется агрегированный балл за работу и запись в таблицу оценок. Вся информация сохраняется в базе данных и подлежит последующей аналитической обработке.

4) Формирование цифрового следа и визуализация результатов. Программа должна генерировать цифровой след студента, отражающий совокупность оценок по всем выполненным заданиям, включая распределение по этапам CDIO и сопоставление с ожидаемыми компетенциями. Визуализация реализуется в виде диаграмм и графиков (с использованием библиотеки Matplotlib), доступных на отдельной вкладке интерфейса. Это предоставляет пользователю возможность анализа динамики прогресса и выявления «узких мест» в образовательной траектории.

5) Генерация отчётов в формате PDF. Одной из ключевых функций системы выступает автоматизированное формирование отчётов об успеваемости конкретного студента. Отчёт должен содержать персональные данные, оценки по каждому критерию, этапы CDIO, а также диаграмму с графическим представлением накопленных баллов. Для создания отчёта используется библиотека ReportLab, а файл предоставляется пользователю для скачивания.

6) Устойчивое хранение и извлечение данных. Для хранения всей информации используется локальная реляционная база данных (SQLite), включающая таблицы студентов, лабораторных работ, критериев, оценок и преподавателей. Архитектура БД проектируется с учётом нормализации и поддержки связей «один ко многим» между сущностями. Это позволяет обеспечить целостность данных и масштабируемость проекта.

7) Интуитивно понятный интерфейс и автономность использования. Графический интерфейс приложения реализуется с использованием фреймворка PyQt, обеспечивая кроссплатформенность, визуальную согласованность и возможность локального запуска без необходимости подключения к внешним ресурсам. Интерфейс должен поддерживать сценарии ввода, редактирования, навигации по данным и экспорта отчётов в формате, пригодном для печати или хранения в цифровом портфолио.

Исходя из этого, формулируемые функциональные требования охватывают как техническую, так и методологическую стороны реализации проекта. Они позволяют не только автоматизировать рутинные процессы оценивания, но и способствуют повышению прозрачности и обоснованности результатов, ориентируя систему на компетентностную парадигму современного образования.

Наряду с функциональными характеристиками, определяющими поведение системы в терминах выполняемых операций, значительное внимание в рамках разработки уделялось **нефункциональным требованиям**,

формирующим качественные атрибуты программного продукта и определяющим его эксплуатационные свойства.

1) Надёжность. Система должна обеспечивать устойчивую работу в условиях непредсказуемого поведения пользователей и отказов файловой системы. Особое внимание уделено обработке ошибок при загрузке базы данных, формировании PDF-документов и визуализации данных.

2) Удобство использования (юзабилити). Интерфейс системы реализован с использованием библиотеки PyQt и разработан с учётом минимизации когнитивной нагрузки. Пользовательские сценарии взаимодействия интуитивны и не требуют предварительной технической подготовки. Применение кириллических шрифтов в отчётах обеспечивает читаемость информации.

3) Производительность. Программа должна обеспечивать отклик не более 300–500 мс при выборке данных и построении графиков. Система функционирует в рамках локального исполнения и не зависит от скорости интернет-соединения.

4) Безопасность и конфиденциальность. Предусмотрена архитектурная возможность ограничения доступа по ролям (при необходимости активации авторизации). Работа с пользовательскими данными осуществляется в рамках локальной базы, что минимизирует риски утечек.

5) Портруемость и совместимость. Система должна функционировать на ОС Windows в рамках локальной среды Python, используя виртуальное окружение и минимальное количество сторонних зависимостей. Поддерживаются шрифты и кодировки, совместимые с русскоязычной образовательной документацией.

6) Масштабируемость и расширяемость. Архитектура приложения построена модульно, что допускает добавление новых компонентов – в частности, возможности интеграции с LMS-системами (например, Moodle), экспорт результатов в табличные форматы (Excel/CSV) и развитие системы цифровой аналитики.

Учёт вышеуказанных нефункциональных требований обеспечил разработку не только функционально корректного, но и практически пригодного для эксплуатации программного решения, соответствующего современным ожиданиям в сфере образовательной цифровизации.

2.2 Обоснование выбора стека технологий

Разработка цифровой системы оценки проектной деятельности студентов, основанной на модели CDIO и принципах компетентностного подхода, потребовала выбора технологического стека, который обеспечивал бы одновременно устойчивость, расширяемость и гибкость решения, а также соответствие образовательной специфике и требованиям аналитической

визуализации. Особое внимание при этом было уделено веб-ориентированному характеру системы, что предполагает кроссплатформенность, отсутствие необходимости установки, а также возможность централизованного доступа через браузер.

В качестве серверной платформы был выбран микрофреймворк **Flask на языке Python**. Его легковесность, модульность и низкий порог вхождения делают Flask особенно эффективным для создания кастомизированных веб-приложений, ориентированных на быстрое прототипирование и последующую интеграцию с аналитическими инструментами. Возможность тонкой настройки архитектуры, гибкость маршрутизации и наличие большого числа расширений позволяют реализовать как административную логику, так и логику взаимодействия с базой данных и системой генерации отчётов.

Для хранения данных выбрана встроенная реляционная база данных **SQLite**. Данный выбор обусловлен ее простотой развертывания и достаточной функциональностью для задач, связанных с хранением результатов оценивания, описаний лабораторных работ, компетенций и цифрового следа студентов. SQLite не требует отдельного сервера, легко интегрируется с Flask и полностью покрывает потребности проекта в рамках локального или малонагруженного образовательного сервиса.

Интерфейсная часть приложения построена на основе **HTML-шаблонов** с использованием движка **Jinja2**, который нативно поддерживается Flask. Это обеспечивает динамическую генерацию страниц в зависимости от контекста, а также реализацию интерактивных элементов, таких как выпадающие списки, формы выбора студентов и кнопки экспорта.

Для аналитической обработки данных применяются библиотеки **Pandas** и **Matplotlib**. Первая отвечает за структурированное представление данных в виде таблиц и обеспечивает удобные средства фильтрации, агрегации и трансформации данных. Вторая позволяет формировать визуализации в виде гистограмм и графиков, которые интегрируются как в веб-интерфейс, так и в итоговый PDF-отчет, обеспечивая визуальную интерпретацию прогресса студентов по лабораторным работам.

Генерация отчётов в формате PDF реализована с использованием библиотеки **ReportLab**, которая предоставляет гибкие возможности верстки, поддержки кириллической кодировки, включения графиков и текстов, распределённых по страницам. Это позволяет экспортировать оценочные данные и цифровой след студента в автономный, стандартизированный формат, пригодный для хранения, печати и последующего анализа.

Каждый компонент выбранного стека технологий выполняет определённую функциональную роль, при этом совокупное использование указанных инструментов позволило достичь баланса между функциональностью, простотой разработки и расширяемостью системы. Такая архитектурная связка обоснована как с инженерной, так и с образовательной точки зрения, полностью удовлетворяя требованиям к цифровым системам поддержки проектного обучения и формирования цифрового следа студентов.

2.3 Архитектура программной системы

Проектируемая система представляет собой веб-приложение, реализующее функциональность оценки проектной деятельности студентов с опорой на компетентностный подход и цифровое сопровождение образовательного процесса. Архитектура решения построена в соответствии с моделью разделения ответственности MVC (Model–View–Controller), которая обеспечивает логическую декомпозицию системы на взаимонезависимые уровни, повышая тем самым сопровождаемость, тестируемость и масштабируемость приложения.

Логическая структура

1) Модель (Model). Данный уровень отвечает за представление и управление данными предметной области. В рамках проекта модели реализованы с использованием системы объектно-реляционного отображения SQLAlchemy, предоставляющей высокоуровневый интерфейс взаимодействия с базой данных SQLite. Модельный уровень включает в себя следующие сущности:

- Student – содержит сведения об учащихся, включая ФИО и учебную группу;
- Lab – описания лабораторных работ, включая наименование, CDIO-этап и содержание;
- Criterion – набор критериев оценки с указанием максимального балла;
- Evaluation – результаты оценивания конкретного студента по заданной работе;
- Instructor – информация о преподавателях, участвующих в оценке и управлении данными.

Каждая сущность моделирует соответствующий элемент образовательной деятельности и формирует логическую основу для построения цифрового следа.

2) Представление (View). Представление реализовано в виде HTML-шаблонов с использованием шаблонизатора Jinja2. На этом уровне осуществляется генерация пользовательского интерфейса в зависимости от контекста запроса, данных из модели и бизнес-логики. Интерфейс организован в виде набора форм, выпадающих списков, таблиц и кнопок, обеспечивающих интуитивно понятное взаимодействие с системой.

Особое внимание уделено реализации страниц:

- панели выбора студента и группы;
- цифрового следа, отражающего оценочную активность по CDIO-этапам;
- страницы генерации отчёта с визуализацией результатов;
- интерфейса работы с лабораторными работами и критериями.

3) Контроллер (Controller). Контроллерный слой реализован посредством маршрутов Flask, описанных в модуле routes.py. Он обеспечивает обработку входящих запросов, выборку и агрегацию данных, вызов бизнес-логики и

передачу результатов в шаблоны. На этом уровне реализованы функции фильтрации студентов по группам, генерации PDF-документов, построения графиков, а также контроль доступа к маршрутам (в будущем – через систему авторизации).

Организация данных и взаимодействие компонентов

Центральным элементом системы является база данных SQLite, организованная в соответствии с нормализованной структурой, обеспечивающей целостность и непротиворечивость хранимых данных. Взаимодействие между представлением и моделью осуществляется опосредованно через контроллер, что соответствует принципам MVC и снижает связанность компонентов.

Отдельным модулем реализована функция генерации PDF-отчётов (`generate_pdf()`), включающая механизмы верстки, форматирования и встраивания графиков. Построение визуализаций осуществляется библиотекой Matplotlib, результаты сохраняются во временный файл и встраиваются в итоговый отчёт, после чего временный файл удаляется.

Также в системе предусмотрены фильтры и декораторы для предварительной обработки данных (`from_json`, `disable_csrf_for_login`), а защита форм осуществляется за счёт подключения CSRF-механизма через Flask-WTF.

Расширяемость и модульность

Разработанная архитектура обеспечивает возможность расширения функциональности без необходимости переработки существующих компонентов. Добавление новых типов отчетов, моделей данных, интерфейсных элементов или логики оценивания может быть выполнено локально, не затрагивая остальную структуру. Это особенно важно в контексте образовательных систем, подверженных частым изменениям в методических подходах и требованиях.

Данная архитектура программного решения представляет собой рационально организованную, логически структурированную систему, обеспечивающую выполнение поставленных задач по фиксации, анализу и визуализации проектной активности студентов в цифровом формате. Она отвечает современным требованиям к образовательным веб-сервисам, демонстрируя высокий уровень адаптивности и инженерной обоснованности.

2.4 Пользовательские сценарии и интерфейс приложения

Проектирование пользовательского интерфейса (UI) и сценариев взаимодействия (UX) в образовательной информационной системе основывается на необходимости обеспечения интуитивно понятного, логически организованного и функционально насыщенного пространства для преподавателя как основного пользователя. Дизайн приложения ориентирован на последовательную реализацию сценариев, соответствующих ключевым

этапам работы с системой: выбор студента, проведение оценки проектной деятельности, анализ цифрового следа и генерация итогового отчета. Все действия пользователя укладываются в четкую структуру, исключая избыточные переходы и обеспечивающую минимизацию когнитивной нагрузки.

Сценарий №1: Оценка студента. На главной странице пользователь переходит по ссылке «Оценить студента» и попадает на интерфейс выбора группы и обучающегося. После выбора студента и лабораторной работы преподавателю предлагается ввести значения по каждому критерию оценки, отображаемому в виде формы с кодами, описаниями и возможными баллами. Для каждого критерия доступно поле ввода комментария, что обеспечивает возможность развернутой обратной связи. По завершении вводов пользователь сохраняет результат, который фиксируется в базе данных. Такой сценарий реализует непрерывную и систематизированную фиксацию оценки проектной активности обучающегося.

Сценарий №2: Анализ цифрового следа. На вкладке «Цифровой след» преподаватель может просматривать динамику активности и уровень компетентности студентов. Данный раздел предоставляет визуализированные данные, агрегированные по этапам CDIO и ролям, зафиксированным в модели RACI. В интерфейсе реализована фильтрация по группе и студенту, а также наглядная таблица, содержащая краткие итоги оценивания, даты выполнения и прогресс по ключевым компетенциям. Такой функционал позволяет выявлять пробелы в освоении компетенций и формировать траектории индивидуального сопровождения.

Сценарий №3: Формирование отчёта. Раздел «Отчет успеваемости» позволяет выбрать обучающегося и получить PDF-документ, содержащий структурированный отчет с перечнем лабораторных работ, суммарной и детализированной оценкой, а также индивидуализированными комментариями преподавателя. Помимо текстовой части, в отчет встраивается диаграмма, отражающая соотношение баллов по каждому из заданий, что способствует лучшему восприятию результатов. Подобный отчет может быть использован как для внутренних целей мониторинга, так и для предоставления студенту в рамках обратной связи.

Особенности интерфейсных решений

Интерфейс системы реализован с применением технологий HTML, CSS и Jinja2 (шаблонизатор Flask), что обеспечивает динамическую генерацию контента в зависимости от пользовательского ввода и состояния сессии. Формы ввода снабжены базовой валидацией, предотвращающей некорректные значения. Элементы управления логически сгруппированы, исключены перегрузка экрана и дублирующий функционал. Все навигационные элементы размещены в виде ссылок на основной странице, что исключает необходимость вложенной навигации.

Данное взаимодействие с системой построено по принципу минимизации действий, необходимых для достижения цели, что полностью соответствует

рекомендациям по проектированию интерфейсов в образовательной среде. Концепция UX направлена на поддержку преподавателя в аналитической и оценочной деятельности, обеспечивая удобство, прозрачность и структурированность рабочих процессов.

2.5 Реализация системы

Реализация веб-приложения для оценки проектной деятельности студентов и анализа их цифрового следа осуществлялась с использованием модульного подхода, предусматривающего логическую и функциональную декомпозицию компонентов, обеспечивающих устойчивость системы к изменениям и расширяемость. Разработка велась на языке программирования Python с применением веб-фреймворка Flask, что позволило сформировать легковесную, но в то же время структурированную серверную часть, обслуживающую маршруты, запросы к базе данных и генерацию шаблонов.

Серверная логика организована в виде отдельных маршрутов (`routes.py`), каждый из которых обслуживает конкретный функциональный сценарий: авторизацию, выбор студентов, фиксацию оценок, генерацию отчётов и визуализацию цифрового следа. Для обеспечения удобства навигации и изоляции ответственности каждый маршрут строго связан с соответствующим шаблоном отображения (HTML + Jinja2) и модулем обработки (`models.py`, `forms.py`).

Применение механизма **маршрутизации** Flask позволило реализовать REST-подобный подход, при котором клиент инициирует HTTP-запрос (GET или POST), сервер обрабатывает данные, взаимодействует с базой данных и возвращает HTML-страницу, динамически сгенерированную на основе переданных параметров. Такой подход упростил отладку и сопровождение, а также обеспечил потенциальную возможность масштабирования системы.

Интерфейс пользователя создаётся при помощи Jinja2 – шаблонизатора, интегрированного во Flask, что дало возможность формировать HTML-страницы динамически, подставляя необходимые значения в структуру документа. В рамках шаблонов реализованы условные блоки, позволяющие управлять отображением интерфейса в зависимости от состояния (например, выбран ли студент или лабораторная работа). Такой подход минимизирует дублирование кода и повышает гибкость визуальной части.

Для хранения информации о студентах, лабораторных работах, критериях, ролях и результатах оценивания использована **реляционная база данных** SQLite. Взаимодействие с базой данных реализовано через библиотеку SQLAlchemy – ORM (Object-Relational Mapping), обеспечивающую прозрачное отображение объектов Python в таблицы и обратно. Каждая сущность системы определена в виде модели (например, Student, Lab, Evaluation), что упрощает реализацию операций чтения, добавления и фильтрации данных.

Конструкции SQLAlchemy позволяют использовать декларативный стиль определения моделей, включая связь между сущностями (one-to-many, many-to-one), индексацию и валидацию полей. Такой способ работы способствует снижению количества ручных SQL-запросов и уменьшает вероятность ошибок, связанных с синтаксисом запросов.

Аналитическая часть системы реализована с помощью библиотек Pandas и Matplotlib, которые обрабатывают и визуализируют информацию об успеваемости студентов. На основе данных о выполненных лабораторных работах генерируются графики с наглядным представлением прогресса. Эти графики встраиваются как на страницу цифрового следа, так и в итоговый отчёт в формате PDF.

Генерация PDF-отчётов реализована с использованием библиотеки ReportLab. Отчёт содержит сведения о студенте, перечень оценённых работ, распределение баллов, расшифровку критериев и комментарии преподавателя. Дополнительно в отчёт добавляется диаграмма успеваемости, сформированная на лету. Для поддержки кириллической кодировки была интегрирована поддержка шрифта DejaVu Sans.

Особенностью реализации стало создание временного файла с диаграммой, который встраивается в PDF и затем удаляется во избежание утечек и накопления лишних данных. Такой подход обеспечил надёжную работу механизма отчётности без зависимости от постоянного хранения графических файлов на сервере.

Реализованы базовые механизмы обработки исключений, обеспечивающие корректное поведение системы при возникновении ошибок. Например, при некорректных значениях формы, отсутствии данных в базе или невозможности генерации графика. Пользователь в этих случаях получает понятное уведомление, а система сохраняет свою стабильность.

Программная реализация проекта основывается на современных подходах к разработке веб-приложений с чётким разделением ответственности между слоями: моделью, представлением и контроллером. Это обеспечивает структурированность архитектуры, устойчивость к ошибкам и возможность дальнейшей адаптации под различные образовательные сценарии.

2.6 Реализация моделей CDIO и RACI в рамках системы

Интеграция концептуальных моделей CDIO и RACI в разрабатываемую веб-систему осуществлена как на уровне логической структуры приложения, так и в архитектуре хранения и интерпретации цифрового следа обучающегося. Оба подхода, изначально разработанные в различных контекстах – инженерного образования и управления проектами соответственно, – в данной системе адаптированы и синтезированы для целей комплексной оценки проектной деятельности студентов.

Фиксация этапности проектной активности CDIO

В системе каждый лабораторный модуль, связанный с конкретным учебным проектом, классифицируется в соответствии с фазами CDIO. Для этого используется поле `cdio_stage`, ассоциированное с каждым заданием и фиксируемое при оценивании. Такое структурирование позволяет отслеживать, на каком этапе проектной деятельности студент проявил активность, а также выявлять зоны дефицита компетенций в рамках конкретных фаз инженерного цикла.

Технически реализация осуществляется путём записи этапа CDIO непосредственно в модель Evaluation, что позволяет не только сегментировать оценки по фазам, но и производить их агрегированный анализ. Благодаря этому становится возможным формирование более точного профиля компетентности, ориентированного не на усреднённый балл, а на многоаспектную структуру навыков, охватывающих весь жизненный цикл инженерного проекта.

RACI – ролевая спецификация в командной деятельности

Параллельно с фазовой классификацией CDIO в системе реализована ролевая декомпозиция по матрице RACI. При фиксации проектной активности обучающегося преподаватель может определить его участие в рамках командного взаимодействия как: ответственное (Responsible), контролирующее (Accountable), консультативное (Consulted) либо информационное (Informed). Для этого в интерфейсе оценивания предусмотрен механизм выбора ролевой позиции, соответствующее значение сохраняется в базе данных.

Реализация логики RACI выполняется через поле `role` модели Evaluation с ограниченным набором допустимых значений. Такая формализация позволяет не только сопоставлять оценку с видом активности, но и проводить перекрёстный анализ вклада студента в коллективный результат. Использование матрицы RACI способствует повышению прозрачности командных процессов и формирует у студентов осознанное отношение к распределению ответственности в рамках проектной среды.

Взаимосвязь моделей и формирование цифрового следа

Системное объединение CDIO и RACI в одном механизме оценивания позволяет формировать многоуровневый цифровой след, отражающий не только количественные параметры успеваемости, но и качественные характеристики участия студента в проектной деятельности. Таким образом, каждый зафиксированный результат включает не только балльную оценку, но и два независимых, но взаимосвязанных признака – этап проектного цикла и ролевую функцию.

На уровне визуализации цифрового следа информация о стадиях CDIO и ролях RACI агрегируется и представляется в виде диаграмм и таблиц, что обеспечивает наглядность анализа и предоставляет преподавателю инструмент для более точного и индивидуализированного сопровождения образовательной траектории студента. Данный подход также позволяет выявлять дисбаланс в участии студента по стадиям CDIO или в ролях RACI, что может быть использовано как основа для корректирующих образовательных интервенций.

Методологические аспекты реализации

Следует подчеркнуть, что при разработке и внедрении моделей CDIO и RACI в систему акцент был сделан на их совместимость с принципами гибкого и проектно-ориентированного обучения. Благодаря этому инструмент оценки не просто фиксирует факты выполненных заданий, но и служит индикатором уровня сформированности проектных и коммуникационных компетенций, что особенно важно в контексте подготовки специалистов, способных эффективно работать в междисциплинарных командах.

2.7 Диаграммы UML

Для формализации архитектурных решений, отражения логики взаимодействия компонентов и моделирования структуры разрабатываемой системы использовался язык UML (Unified Modeling Language), являющийся де-факто стандартом визуального представления сложных программных систем. В рамках проектирования веб-приложения были построены ключевые диаграммы, обеспечивающие целостное понимание внутреннего устройства системы и взаимосвязей между её составляющими. Выбор типов диаграмм обусловлен необходимостью представить как статические, так и динамические аспекты функционирования приложения.

Диаграмма компонентов демонстрирует модульную организацию системы и отражает разделение на логические блоки: контроллеры, модели, представления, база данных, модули маршрутизации и вспомогательные библиотеки. Центральным элементом выступает контроллер, принимающий HTTP-запросы, перенаправляющий данные между представлением и моделью, а также координирующий вызовы функций аналитики, генерации отчётов и визуализации цифрового следа. Такая структура полностью соответствует паттерну MVC (Model–View–Controller), что обеспечивает поддерживаемость и расширяемость системы.

Модуль базы данных представлен ORM-моделью SQLAlchemy, включающей сущности Student, Lab, Evaluation, Criterion, Instructor, каждая из которых описывает соответствующую таблицу и её связи. Компоненты визуализации (модули на основе Matplotlib и ReportLab) отделены в самостоятельный слой и вызываются по мере необходимости, что способствует модульности кода и снижает связность между логическими уровнями приложения.

Диаграмма классов отражает основные сущности предметной области и их взаимосвязи, реализованные средствами объектно-реляционного отображения. Классы Student, Evaluation, Lab, Criterion и Instructor связаны отношениями "один ко многим" и "многие ко многим", что позволяет обеспечить логическую целостность при работе с данными. Атрибуты классов соответствуют колонкам в таблицах базы данных и отражают как простые поля

(например, `full_name`, `group`), так и составные структуры (`criteria_json`, `feedback_json`), используемые для хранения оценок и комментариев.

Кроме атрибутов и связей, диаграмма включает методы, обеспечивающие бизнес-логику: подсчёт итоговой оценки, интерпретация цифрового следа, выборка данных по CDIO-этапу и RACI-роли. Такая детализация позволяет чётко проследить логику работы с сущностями на уровне кода и облегчает интеграцию новых функциональностей.

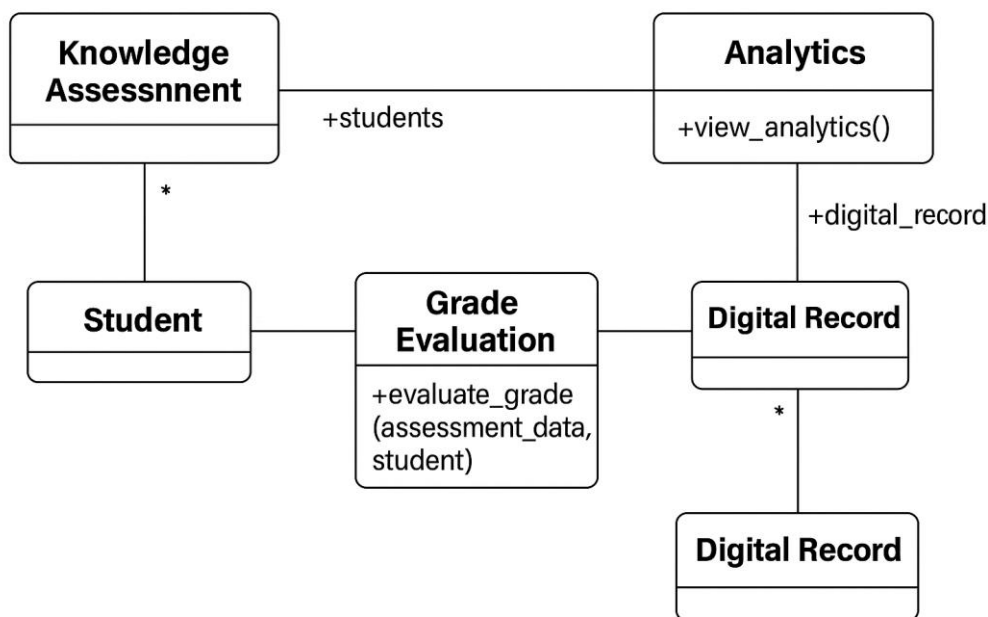


Рисунок 2.1 – Диаграмма классов системы оценки знаний

Диаграмма последовательностей моделирует взаимодействие между пользователем и системой в рамках сценария "оценивание студента". Последовательность сообщений между фронтендом, контроллером, моделью и базой данных позволяет проанализировать порядок вызовов, уточнить точки входа и выхода из операций, а также выявить потенциальные места узких горлышек. В частности, подчёркивается взаимодействие интерфейса с динамически подгружаемыми списками студентов и лабораторных работ, выбор ролевой функции и генерация PDF-отчёта, сопровождающегося построением графика.

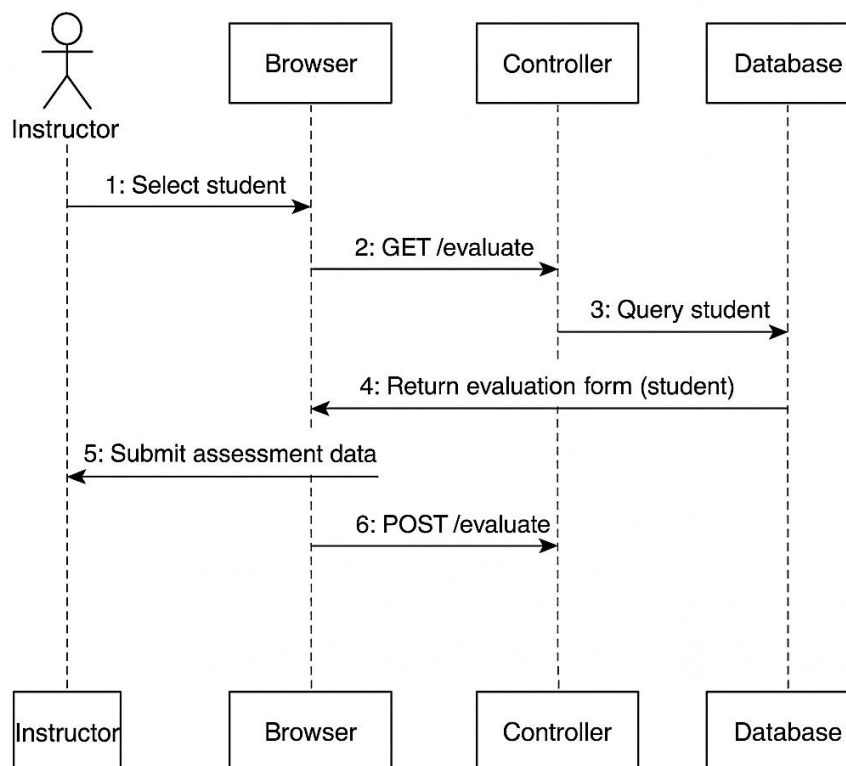


Рисунок 2.2 – Диаграмма последовательностей

Наличие данной диаграммы особенно важно при проведении тестирования и отладки, поскольку она даёт наглядное представление о логике обработки пользовательского запроса и сопоставлении его с внутренними ресурсами системы.

Методологическое значение UML-моделирования

Использование UML в процессе проектирования обеспечило формализацию требований, снижение вероятности логических ошибок и создание единого понятийного пространства для всех участников разработки. Диаграммы выступили не только в роли документации, но и как средство верификации корректности проектных решений, а также основа для автоматизированного анализа и генерации кода.

Визуальное моделирование архитектуры с использованием UML выступило необходимым этапом при разработке системы и обеспечило её структурную прозрачность, архитектурную строгость и соответствие методологическим требованиям современного проектирования.

3 Экспериментальная часть

3.1 Методика тестирования программного продукта

Тестирование программного обеспечения является неотъемлемой стадией жизненного цикла разработки и служит ключевым инструментом для верификации корректности реализованных функций, соответствия системы заданным требованиям, а также выявления возможных дефектов до её внедрения в эксплуатацию. В рамках данного проекта методика тестирования была выстроена в соответствии с принципами черного и серого ящика, с акцентом на проверку бизнес-логики, интерфейсных взаимодействий, устойчивости к некорректному вводу и целостности данных в базе.

Тестирование проводилось поэтапно, охватывая три основных направления: функциональное тестирование, тестирование пользовательского интерфейса и тестирование устойчивости к ошибкам.

1. Функциональное тестирование включало в себя проверку соответствия всех реализованных модулей заданным требованиям технического задания. Проверялись корректность регистрации и обработки цифрового следа студента, формирование отчётности, выбор и фильтрация данных, отображение графиков, а также генерация PDF-документа. Каждый функциональный блок (например, модуль оценки, модуль формирования отчёта, модуль управления лабораторными работами) подвергался отдельной проверке, в ходе которой вводились предопределённые сценарии и анализировались ответы системы.

2. Интерфейсное тестирование было направлено на проверку корректности отображения элементов управления, адаптивности макета под разные разрешения, а также логичности и интуитивной понятности пользовательского взаимодействия. Особое внимание уделялось проверке реакций формы при изменении группы и выборе студента в модуле отчётности. Кроме того, оценивалась визуальная интерпретация графиков и доступность информации, представленной в таблицах и диаграммах.

3. Тестирование устойчивости и обработки ошибок преследовало цель выявить потенциальные уязвимости, возникающие при введении некорректных данных, отсутствии обязательных параметров или повреждении содержимого базы данных. Были протестированы сценарии, включающие отсутствие подключённой базы, попытку доступа к удалённой записи, генерацию PDF-файла при пустом списке оценок и другие пограничные случаи. При этом проверялась способность системы выдавать информативные сообщения об ошибках и корректно завершать процесс без аварийного завершения сессии.

Все тестовые случаи документировались, а результаты верифицировались вручную, с фиксацией обнаруженных несоответствий. Были составлены таблицы тест-кейсов, содержащие входные данные, ожидаемый результат и фактическое поведение системы.

Следует отметить, что при тестировании учитывались также потенциальные

особенности развёртывания проекта на стороннем хостинге, включая совместимость с виртуальной средой, корректность выполнения скрипта `gulp` и доступность внешних ресурсов, таких как подключаемые шрифты и изображения.

Применённая методика тестирования обеспечила всестороннюю проверку системы в условиях, приближенных к реальной эксплуатации, и позволила выявить и устранить ряд недочётов, повысив стабильность и надёжность работы всего программного комплекса.

3.2 Описание тестовых сценариев и вводных данных

В процессе тестирования разработанной системы была применена последовательная стратегия, предусматривающая построение тестовых сценариев, охватывающих ключевые пользовательские действия и внутренние процессы обработки данных. Каждый сценарий формировался с учётом заранее определённого набора входных данных и предусматривал фиксированный ожидаемый результат, что позволило обеспечить объективную оценку функционирования отдельных компонентов системы.

Тестовые сценарии были структурированы по функциональным модулям, охватывая следующие аспекты: выбор и отображение групп и студентов, формирование отчёта с динамически загружаемыми оценками, генерация PDF-документа, построение графиков, а также обработка некорректных и пограничных данных.

Сценарий 1. Отображение списка студентов по выбранной группе

Цель: проверка правильности фильтрации студентов по выбранной группе.

Вводные данные: в базе данных содержится группа: Computer Science. В группе – семь студентов.

Шаги:

- 1) Пользователь открывает страницу "Отчёт успеваемости".
- 2) В выпадающем списке выбирает группу Computer Science.
- 3) Ожидаемый результат: в списке студентов отображаются только студенты, принадлежащие к группе Computer Science.

Сценарий 2. Отображение цифрового следа выбранного студента

Цель: проверка загрузки оценок и проектной активности по выбранному студенту.

Вводные данные: студент Садыкова Алтынай из группы Computer Science имеет три завершённые лабораторные работы с различными оценками.

Шаги:

- 1) Выбрать группу Computer Science.
- 2) Выбрать студента Садыкова Алтынай
- 3) Ожидаемый результат: на странице отображаются данные об оценках, названиях лабораторных работ, этапах CDIO и комментариях преподавателя (если имеются).

Сценарий 3. Генерация PDF-отчета

Цель: проверка корректности формирования PDF-документа по выбранному студенту.

Вводные данные: студент Садыкова Алтынай имеет две выполненные лабораторные работы с заполненными критериями и отзывами.

Шаги:

- 1) Выбрать соответствующую группу и студента.
- 2) Нажать на кнопку "Скачать PDF".
- 3) Ожидаемый результат: открывается или загружается PDF-файл, содержащий полную информацию об успеваемости студента, включая баллы, CDIO-этапы, критерии, комментарии, а также диаграмму оценок.

Сценарий 4. Обработка отсутствующих данных

Цель: проверка поведения системы при отсутствии оценок у выбранного студента.

Вводные данные: студент Садыкова Алтынай из группы Computer Science, по которому не внесены никакие данные в таблицу оценок.

Шаги:

- 1) Выбрать группу ИС-203.
- 2) Выбрать студента Садыкова Алтынай
- 3) Ожидаемый результат: на странице не отображаются оценки и графики, но не возникает ошибок или сбоев. При попытке генерации PDF выводится уведомление о невозможности построения отчёта.

Сценарий 5. Проверка визуализации диаграмм

Цель: проверка отображения всех названий лабораторных работ в графике и корректности балльной шкалы.

Вводные данные: студент имеет пять лабораторных работ с уникальными названиями длиной до 100 символов.

Шаги:

- 1) Выбрать студента.
- 2) Сформировать отчёт.
- 3) Ожидаемый результат: на диаграмме корректно отображаются все названия, подписи и значения по осям. Горизонтальная прокрутка или сокращение названий не происходит.

Сценарий 6. Работа при ошибочных параметрах запроса

Цель: оценка устойчивости системы при попытке подстановки несуществующего student_id или group.

Вводные данные: в адресной строке вручную вводится несуществующий идентификатор, например ?student_id=999.

Шаги:

- 1) Перейти по ссылке /report?student_id=999.
- 2) Ожидаемый результат: система корректно реагирует – данные не отображаются, ошибки не выбрасываются, предлагается выбрать корректного студента.

Следовательно, охват тестовыми сценариями различных ситуаций – как

штатных, так и преднамеренно ошибочных – позволил удостовериться в высокой степени отказоустойчивости, надёжности и функциональной полноте программного решения. Результаты проведённого тестирования легли в основу итоговой валидации системы и подтвердили её готовность к внедрению в учебный процесс.

3.3 Результаты тестирования функциональных модулей

В процессе верификации программного решения была проведена поэтапная проверка всех ключевых функциональных модулей системы. Целью данного этапа выступало установление степени соответствия реализованных компонентов заранее сформулированным требованиям, а также выявление потенциальных недочётов в логике интерфейса, структуре обработки данных и механизмах визуализации.

Функциональные модули тестировались как изолированно, так и в составе общего пользовательского сценария с учётом возможных пограничных состояний, что позволило повысить достоверность оценки их корректной работы.

Модуль выбора группы и студента

Проверка выпадающих списков групп и студентов продемонстрировала стабильную фильтрацию обучающихся в зависимости от выбранной группы. Навигация реализована через передачу параметров в URL (GET-запросы), что позволило обеспечить надёжную работу без необходимости обновления страницы вручную. Ошибок отображения, зависаний и ситуаций с пустыми данными при наличии заполненной базы выявлено не было.

Модуль отображения оценок и цифрового следа

Интерфейс цифрового следа корректно реагировал на выбор студента, формируя таблицу с перечислением лабораторных работ, баллов и этапов CDIO. Были протестированы случаи как с полной историей оценивания, так и при отсутствии данных – в обоих сценариях интерфейс функционировал стабильно, предоставляя пользователю соответствующую информацию или оставляя область пустой без выброса ошибок.

Модуль генерации PDF-отчёта

Механизм формирования отчета в формате PDF сработал без сбоев во всех тестовых кейсах. Проверялись случаи с длинными названиями лабораторных работ, нестандартными символами (включая кириллицу) и многострочными комментариями преподавателя. Использование шрифта DejaVu обеспечило корректное отображение всех символов, включая русскоязычный текст, а встроенная функция обёртки строк (`draw_wrapped_text`) устранила проблему выхода текста за границы страницы.

Графики в PDF отображались в соответствии с введёнными баллами, оси были подписаны корректно, а подписи лабораторных работ читались без искажений. Проблемы с временными файлами, возникавшие ранее на этапе

генерации изображений графиков, были устранены путём использования явного закрытия файла перед его удалением.

Модуль построения диаграмм в интерфейсе

Диаграмма, отображаемая на веб-странице, визуализировала данные в соответствии с ожиданиями. Названия лабораторных работ, вне зависимости от длины, выводились в полном виде, обеспечивая читаемость за счёт поворота подписей по оси X и применения уменьшенного шрифта. Масштабирование графика происходило автоматически благодаря динамическому расчёту размеров области визуализации.

Обработка исключений и устойчивость к ошибкам

В системе были преднамеренно вызваны ситуации с несуществующими идентификаторами студентов, отсутствием лабораторных работ, а также некорректными параметрами запроса. Во всех случаях приложение отработало стабильно, ошибки перехватывались корректно, страницы не "падали", и пользователь получал ожидаемую реакцию системы – либо пустую таблицу, либо предложение выбрать корректные параметры.

В совокупности все модули показали устойчивость к пользовательским ошибкам, логическую целостность и соответствие заданной функциональности. Система успешно прошла этап функционального тестирования и может быть рекомендована к эксплуатации в учебной среде с дальнейшей адаптацией под конкретные педагогические задачи.

3.4 Анализ соответствия результатов поставленным задачам

В рамках выполнения дипломного проекта была сформулирована совокупность целей и задач, направленных на создание веб-ориентированной информационной системы, обеспечивающей сбор, фиксацию, обработку и визуализацию цифрового следа обучающихся в условиях проектно-ориентированного образовательного процесса. Проведённый анализ результатов реализации системы позволяет утверждать, что основные задачи проекта были успешно решены, а целевая функциональность достигнута в полном объёме.

Так, в части **исследования подходов** к оценке проектной деятельности и **анализу образовательных цифровых платформ**, были выявлены ключевые ограничения традиционных форм фиксации успеваемости и сформулированы принципы, на которых должно строиться цифровое оценивание в рамках CDIO- и компетентностно-ориентированной модели. Это исследование легло в основу архитектуры приложения и выбора методов обработки данных.

На этапе **проектирования системы** выполнена разработка логической и физической архитектуры, включающей клиентскую и серверную части, базу данных, модули визуализации, PDF-отчётности и динамического взаимодействия с пользователем. Каждый компонент проектировался с учётом требований устойчивости, масштабируемости и расширяемости.

Особое внимание было уделено **разработке пользовательских сценариев**, полностью покрывающих предполагаемые кейсы взаимодействия преподавателя с системой: от оценки конкретных критериев до генерации аналитического отчета. Интерфейс обеспечил интуитивно понятную навигацию, а визуальные компоненты (таблицы, графики) – наглядность восприятия данных.

Функциональная реализация системы включила в себя интеграцию модели CDIO с разбивкой оценочной деятельности по этапам «Conceive», «Design», «Implement» и «Operate», а также поддержку распределения ответственности в команде по матрице RACI. Это позволило задокументировать роль каждого участника проекта в рамках командной работы и отразить эту информацию в цифровом следе обучающегося.

Тестирование всех модулей подтвердило корректность работы системы при различных сценариях. Зафиксированы стабильные показатели по всем функциональным направлениям: загрузка данных, фильтрация, визуализация, генерация отчетов и устойчивость к ошибкам пользователя. Система успешно справляется с обработкой как типовых, так и пограничных ситуаций.

Следующим образом, все поставленные задачи – от концептуализации и выбора технологий до реализации, тестирования и анализа – были последовательно выполнены. Разработанное программное средство соответствует исходным требованиям и может быть использовано в образовательных организациях, ориентированных на компетентностный подход и формирование цифрового профиля студентов.

ЗАКЛЮЧЕНИЕ

В рамках данной дипломной работы была разработана веб-система для оценки проектной деятельности студентов, основанная на анализе их цифрового следа. Основное внимание в проекте было уделено формированию объективной модели отслеживания прогресса студентов, с интеграцией подходов CDIO и RACI, что позволило отразить не только уровень освоения компетенций, но и распределение ролевой ответственности в команде.

В ходе выполнения проекта были решены следующие задачи:

- проведён анализ существующих систем оценки, применяемых в контексте проектного обучения;
- определены функциональные и нефункциональные требования к системе;
- обоснован выбор используемых технологий;
- спроектирована архитектура клиент-серверного веб-приложения;
- разработан пользовательский интерфейс и реализована база данных;
- внедрены механизмы формирования цифрового следа, привязанные к компетенциям и этапам проектной деятельности;
- реализована генерация отчётов об успеваемости с возможностью загрузки в формате PDF.

Разработанная система отвечает поставленным требованиям, успешно справляется с задачей фиксации и анализа проектной активности обучающихся, обеспечивает визуализацию результатов и удобный инструментарий для преподавателя. Таким образом, цель дипломной работы достигнута, а её результаты могут быть применены на практике в образовательных учреждениях, использующих проектно-ориентированный подход в обучении.

Приложение А

Техническое задание

Техническое задание на разработку веб-приложения «Система оценки знаний и цифровой след обучающихся»

1) Наименование проекта

Разработка веб-приложения для преподавателей, обеспечивающего автоматизированную оценку качества выполнения лабораторных работ студентами, а также формирование их цифрового следа на основе критериев освоения компетенций.

2) Основание для разработки

Данный проект реализуется в рамках выпускной квалификационной работы по направлению «Программная инженерия» с целью создания программного инструмента, способствующего совершенствованию процедур мониторинга индивидуальных образовательных достижений и оценки сформированности профессиональных компетенций студентов.

3) Назначение и область применения

Создаваемое программное обеспечение предназначено для преподавателей высших учебных заведений, осуществляющих контроль за качеством выполнения студентами лабораторных заданий, а также для формирования и анализа индивидуальных траекторий обучающихся посредством отслеживания их цифрового следа.

Система может применяться в образовательной среде для:

- ведения аналитического учета прогресса студентов;
- принятия решений по индивидуальному сопровождению обучающихся;
- повышения объективности и прозрачности процедуры оценивания.

4) Цель и задачи разработки

Цель разработки заключается в создании функционально завершённого веб-приложения, позволяющего автоматизировать процесс выставления оценок по множеству критериев и формировать цифровой профиль обучающегося на основе накопленных данных.

Основные задачи:

- формализация критериев оценивания лабораторных работ;
- разработка интерфейса для ввода и обработки экспертных оценок;
- построение и визуализация цифрового следа обучающегося;
- хранение и обработка результатов в базе данных;
- обеспечение логической целостности и доступности информации о текущем уровне компетенций.

5) Требования к функциональности

Приложение должно обеспечивать реализацию следующих функций:

- 1) Регистрация и отображение списка студентов.
- 2) Выбор лабораторной работы для оценки.
- 3) Ввод оценок преподавателем по заранее определённым критериям

(например, корректность постановки задачи, полнота UML-моделирования, архитектурные решения, прототип интерфейса, качество базы данных и др.)

- 4) Расчёт итогового балла по шкале, соответствующей 100%-ной системе.
- 5) Хранение всех оценок в единой базе данных.
- 6) Отображение индивидуального профиля обучающегося.
- 7) Формирование цифрового следа на основе накопленных компетенций.
- 8) Визуализация динамики успеваемости и профиля компетенций (графики, диаграммы).

6) Требования к реализации

Разработка программного обеспечения осуществляется в виде веб-приложения с использованием следующих технологий:

- Backend: язык программирования Python с использованием микрофреймворка Flask;
- Frontend: HTML5, CSS3, шаблонизатор Jinja2, библиотека визуализации Chart.js;
- База данных: SQLite, с возможностью миграции на PostgreSQL при расширении функционала;
- ORM: SQLAlchemy;
- Формы и валидация: Flask-WTF.

Особое внимание должно быть уделено устойчивости к пользовательским ошибкам, надёжному хранению и корректной обработке данных, а также обеспечению логической завершенности интерфейсных сценариев.

7) Требования к надёжности, эргономике и оформлению

- Веб-интерфейс должен быть интуитивно понятным, минималистичным и унифицированным во всех модулях;
- Информация об обучающемся должна быть представлена в структурированной и визуально доступной форме;
- Результаты оценивания и данные цифрового следа должны быть защищены от несанкционированного изменения.

8) Этапы разработки

Разработка программного обеспечения предполагает реализацию следующих этапов:

- 1) Анализ предметной области, формализация требований, проектирование интерфейса и архитектуры.
- 2) Разработка базы данных и моделей.
- 3) Создание модулей интерфейса (главная страница, страница оценки, страница аналитики).
- 4) Реализация логики обработки и сохранения данных.
- 5) Визуализация цифрового следа.
- 6) Тестирование и отладка системы.
- 7) Подготовка пользовательской и технической документации.

На выходе будет разработано веб-приложение, позволяющее оценивать знания студентов, учитывать их компетенции, анализировать динамику их развития и формировать отчеты по результатам обучения.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

- 1 Блум Б. Таксономия образовательных целей: Классификация целей в когнитивной области / Пер. с англ. – М.: Педагогика, 1986. – 178 с.
- 2 Kolb D. *Experiential Learning: Experience as the Source of Learning and Development*. – Englewood Cliffs: Prentice Hall, 1984. – 256 p.
- 3 González J., Wagenaar R. *Tuning Educational Structures in Europe. Final Report*. – Bilbao: University of Deusto, 2008. – URL: <https://tuningacademy.org/publications/> (дата обращения: 13.05.2025).
- 4 Bourque P., Fairley R.E. (eds.). *Guide to the Software Engineering Body of Knowledge (SWEBOOK V3.0)*. – IEEE Computer Society, 2014. – URL: <https://www.computer.org/education/bodies-of-knowledge/software-engineering> (дата обращения: 13.05.2025).
- 5 Crawley E., Malmqvist J., Östlund S., Brodeur D. *Rethinking Engineering Education: The CDIO Approach*. – 2nd ed. – New York: Springer, 2014. – 311 p.
- 6 Мурзин Д.Ю., Беляев С.А. *Компетентностный подход в инженерном образовании*. – Казань: КНИТУ-КАИ, 2021. – 204 с.
- 7 Глухов В.В., Татаринов В.А. *Управление проектами: учебник*. – М.: Юрайт, 2022. – 427 с.
- 8 Сергеев И.В., Чернышев С.А. *Управление знаниями в цифровой экономике*. – СПб.: Питер, 2020. – 304 с.
- 9 Фурсова С.Е. CDIO-подход к проектированию образовательных программ – 2020. – № 2. – С. 48–55.
- 10 Журавлева И.В., Козлова И.В. Формирование цифрового следа обучающегося // *Инф. технологии в образовании*. – 2021. – № 4 (40). – С. 12–18.
- 11 Siemens G. *Learning Analytics: The Emergence of a Discipline* // *American Behavioral Scientist*. – 2013. – Vol. 57(10). – P. 1380–1400.
- 12 Милютин А.В., Шабанова И.М. Цифровые следы обучающихся в образовательных системах // *Университетская книга*. – 2021. – № 6. – С. 36–41.
- 13 OECD. *The Future of Education and Skills: Education 2030*. – OECD Publishing, 2018. – URL: <https://www.oecd.org/education/2030-project/> (дата обращения: 15.05.2025).
- 14 Python Software Foundation. *The Python Language Reference*. – URL: <https://docs.python.org/3/reference/> (дата обращения: 10.05.2025).
- 15 SQLite Documentation. – URL: <https://sqlite.org/docs.html> (дата обращения: 10.05.2025).
- 16 PyQt Documentation. – URL: <https://www.riverbankcomputing.com/static/Docs/PyQt5/> (дата обращения: 10.05.2025).
- 17 Hunter J.D. *Matplotlib: A 2D Graphics Environment* // *Computing in Science & Engineering*. – 2007. – Vol. 9(3). – P. 90–95.
- 18 McKinney W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. – 2nd ed. – O'Reilly Media, 2017. – 544 p.

- 19 Ляпунов В.П., Козырева Т.А. Цифровые следы в электронных образовательных средах // Образовательные технологии и общество. – 2019. – Т. 22, № 3. – С. 358–370.
- 20 Репина И.В. Цифровая трансформация образования: вызовы и перспективы – 2021. – № 10. – С. 5–13.
- 21 Кудрявцев А.Ф. Информационные технологии в образовании. – М.: Форум, 2020. – 208 с.
- 22 Галицкий Б.М. Искусственный интеллект и системы поддержки принятия решений. – М.: ДМК Пресс, 2018. – 352 с.
- 23 Костина И.А. CDIO как средство формирования цифровой компетентности студентов // Современные проблемы науки и образования. – 2022. – № 2.
- 24 ISO/IEC 25010:2011. Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – System and software quality models.
- 25 Артемьев С.А. Внедрение компетентностной модели в ИС для оценки знаний // Современные информационные технологии и ИТ-образование. – 2023. – № 2. – С. 79–83.

Приложение Б

Листинг (код) программы

```
# Код run.py – точка входа (Flask app)
from app import app

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=10000)

# Код маршрутов (страниц) routes.py
from flask import render_template, request, redirect, url_for, flash, send_file
from flask_login import login_user, logout_user, current_user, login_required
from flask_wtf.csrf import generate_csrf

from app import app, db
from app.models import Student, Lab, Criterion, Evaluation, Instructor, Group
from app.forms import LoginForm, RegisterForm, ChangePasswordForm

from datetime import datetime
from io import BytesIO
import io
import json
import os
import tempfile
import base64

import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt

from reportlab.pdfgen import canvas
from reportlab.lib.pagesizes import A4
from reportlab.lib.units import cm

@app.route('/auth/login', methods=['GET', 'POST'])
def login():
    login_form = LoginForm()
    register_form = RegisterForm()
    if login_form.validate_on_submit():
        user = Instructor.query.filter_by(username=login_form.username.data).first()
        if user and user.check_password(login_form.password.data):
            login_user(user)
            flash("Добро пожаловать!", "success")
            return redirect(url_for('index'))
```

```

        flash("Неверное имя пользователя или пароль.", "danger")
    return render_template('auth.html', login_form=login_form,
register_form=register_form)

@app.route('/auth/register', methods=['GET', 'POST'])
def register():
    login_form = LoginForm()
    register_form = RegisterForm()

    if register_form.validate_on_submit():
        existing =
Instructor.query.filter_by(username=register_form.username.data).first()
        if existing:
            flash('Имя пользователя уже занято.', 'warning')
        else:
            new_user = Instructor(
                full_name=register_form.full_name.data,
                username=register_form.username.data
            )
            new_user.set_password(register_form.password.data)
            db.session.add(new_user)
            db.session.commit()
            flash("Регистрация прошла успешно! Войдите в систему.", "success")
            return redirect(url_for('login'))
    return render_template('auth.html', login_form=login_form,
register_form=register_form)

@app.route('/logout')
@login_required
def logout():
    logout_user()
    flash('Вы вышли из системы.', 'info')
    return redirect(url_for('login'))

@login_required
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/profile')
@login_required
def profile():
    instructor = current_user
    my_groups = Group.query.filter_by(instructor_id=instructor.id).all()

```

```

my_labs = Lab.query.filter_by(instructor_id=instructor.id).all()
return render_template('profile.html', instructor=instructor, my_groups=my_groups,
my_labs=my_labs)

```

```

@app.route('/profile/change_password', methods=['GET', 'POST'])
@login_required
def change_password():
    form = ChangePasswordForm()
    if form.validate_on_submit():
        if current_user.check_password(form.old_password.data):
            current_user.set_password(form.new_password.data)
            db.session.commit()
            flash("Пароль успешно изменён", "success")
            return redirect(url_for('profile'))
        else:
            flash("Старый пароль неверен", "danger")
    return render_template('change_password.html', form=form)

```

```

@app.route('/students')
def student_list():
    groups = db.session.query(Student.group).distinct().all()
    group_names = [g[0] for g in groups]

    selected_group = request.args.get('group')
    students = []

    if selected_group:
        students = Student.query.filter_by(group=selected_group).all()

    return render_template('students.html', students=students, groups=group_names,
selected_group=selected_group)

```

```

@app.route('/my-groups')
@login_required
def my_groups():
    groups = db.session.query(Student.group).distinct().all()
    group_names = [g[0] for g in groups]

    selected_group = request.args.get('group')
    students = []
    evaluated_ids = []
    student_data = []

    if selected_group:

```

```

students = Student.query.filter_by(group=selected_group).all()
evaluated_ids = [e.student_id for e in Evaluation.query.all()]
for s in students:
    evaluations = Evaluation.query.filter_by(student_id=s.id).all()
    total_score = sum(e.total_score for e in evaluations if e.total_score is not
None)
    student_data.append({
        'id': s.id,
        'full_name': s.full_name,
        'group': s.group,
        'total_score': round(total_score, 2)
    })

    return render_template('my_groups.html', students=students, groups=group_names,
selected_group=selected_group, evaluated_ids=evaluated_ids,
student_data=student_data)

@app.route('/evaluate/<int:student_id>', methods=['GET', 'POST'])
def evaluate(student_id):
    student = Student.query.get_or_404(student_id)
    labs = Lab.query.all()
    selected_lab_id = request.args.get('lab_id', type=int)

    if request.method == 'POST':
        selected_lab_id = int(request.form['lab_id'])
        criteria = Criterion.query.filter_by(lab_id=selected_lab_id).all()

        scores = {}
        feedback = {}
        total_score = 0

        for criterion in criteria:
            score = float(request.form.get(f'score_{criterion.id}', 0))
            text = request.form.get(f'feedback_{criterion.id}', "").strip()
            scores[criterion.code] = score
            if text:
                feedback[criterion.code] = text
            total_score += score

        lab = Lab.query.get(selected_lab_id)
        cdio_stage = lab.cdio_stage if lab and lab.cdio_stage else '—'

        existing = Evaluation.query.filter_by(student_id=student.id,
lab_id=selected_lab_id).first()

```

```

if existing:
    existing.criteria_json = json.dumps(scores, ensure_ascii=False)
    existing.feedback_json = json.dumps(feedback, ensure_ascii=False)
    existing.total_score = total_score
    existing.cdio_stage = cdio_stage
    existing.task_name = Lab.query.get(selected_lab_id).title
else:
    evaluation = Evaluation(
        student_id=student.id,
        lab_id=selected_lab_id,
        task_name=Lab.query.get(selected_lab_id).title,
        criteria_json=json.dumps(scores, ensure_ascii=False),
        feedback_json=json.dumps(feedback, ensure_ascii=False),
        total_score=total_score,
        cdio_stage=cdio_stage
    )
    db.session.add(evaluation)
    db.session.commit()
    flash("Оценка успешно сохранена", "success")
    return redirect(url_for('evaluate', student_id=student.id, lab_id=selected_lab_id))

criteria = []
if selected_lab_id:
    criteria = Criterion.query.filter_by(lab_id=selected_lab_id).all()

return render_template('evaluate.html', student=student, labs=labs,
                       selected_lab_id=selected_lab_id, criteria=criteria,
                       csrf_token=generate_csrf())

@app.route('/labs')
def labs():
    labs = Lab.query.all()
    return render_template('labs.html', labs=labs)

@app.route('/trace', methods=['GET'])
#@login_required
def trace():
    groups = db.session.query(Student.group).distinct().all()
    group_names = [g[0] for g in groups]

    selected_group = request.args.get('group')
    selected_student_id = request.args.get('student_id', type=int)

```

```

students = []
student = None
evaluations = []
graph_base64 = None
criteria_map = {}
criteria_cdio = {}

if selected_group:
    students = Student.query.filter_by(group=selected_group).all()

if selected_student_id:
    student = Student.query.get(selected_student_id)
    raw_evaluations =
Evaluation.query.filter_by(student_id=selected_student_id).all()

for c in Criterion.query.all():
    key = f"{c.lab_id}:{c.code}".strip()
    criteria_map[key] = c.description.strip()
    criteria_cdio[key] = c.cdio_stage.strip() if c.cdio_stage else '—'

for e in raw_evaluations:
    lab = Lab.query.get(e.lab_id)
    lab_full_title = f"{lab.title}. {lab.description}" if lab else e.task_name
    criteria = Criterion.query.filter_by(lab_id=e.lab_id).all()
    max_score = sum(c.max_score for c in criteria)
    evaluations.append({
        'task_name': lab_full_title,
        'cdio_stage': e.cdio_stage or "",
        'criteria': json.loads(e.criteria_json),
        'feedback': json.loads(e.feedback_json) if e.feedback_json else {},
        'total_score': round(float(e.total_score), 2),
        'max_score': round(float(max_score), 2),
        'updated_at': e.updated_at.strftime("%d.%m.%Y") if e.updated_at else '—',
        'lab_id': e.lab_id
    })

# график
scores = [e['total_score'] for e in evaluations]
labels = [e['task_name'] for e in evaluations]
if scores:
    fig, ax = plt.subplots(figsize=(9, 4))
    ax.plot(labels, scores, marker='o', linestyle='-', color='teal')
    for i, score in enumerate(scores):
        ax.annotate(f"{score:.2f}", (i, score), textcoords="offset points", xytext=(0,

```

```

-12), ha='center', fontsize=8)
    ax.set_title(f'Процесс студента: {student.full_name}', fontsize=12)
    ax.set_xlabel('Лабораторные работы', fontsize=10)
    ax.set_ylabel('Баллы', fontsize=10)
    ax.set_ylim(0, max(scores + [5]))

    short_labels = [label.split('.')[0] + '.' for label in labels]
    ax.set_xticks(range(len(short_labels)))
    ax.set_xticklabels(short_labels, rotation=30, ha='right', fontsize=9)

    buf = io.BytesIO()
    plt.tight_layout()
    plt.savefig(buf, format='png')
    buf.seek(0)
    graph_base64 = base64.b64encode(buf.read()).decode('utf-8')
    plt.close()

    return render_template(
        'trace.html',
        students=students,
        groups=group_names,
        selected_group=selected_group,
        selected_student_id=selected_student_id,
        student=student,
        evaluations=evaluations,
        criteria_map=criteria_map,
        criteria_cdio=criteria_cdio,
        graph_base64=graph_base64
    )

@app.route('/report', methods=['GET'])
def report():
    groups = db.session.query(Student.group).distinct().all()
    group_names = [g[0] for g in groups]

    selected_group = request.args.get('group')
    students = []
    student = None
    evaluations = []
    enriched_evaluations = []

    if selected_group:
        students = Student.query.filter_by(group=selected_group).all()

```

```

selected_student_id = request.args.get('student_id', type=int)

if selected_student_id:
    student = Student.query.get(selected_student_id)
    evaluations = Evaluation.query.filter_by(student_id=student.id).all()

    for e in evaluations:
        lab = Lab.query.filter_by(title=e.task_name).first()
        full_title = f"{e.task_name}. {lab.description}" if lab else e.task_name
        enriched_evaluations.append({
            'full_title': full_title,
            'score': round(e.total_score, 2) if e.total_score else 0,
            'cdio_stage': e.cdio_stage
        })

    total_score_sum = sum(e['score'] for e in enriched_evaluations)

    return render_template(
        'report.html',
        groups=group_names,
        students=students,
        selected_group=selected_group,
        selected_student_id=selected_student_id,
        student=student,
        enriched_evaluations=enriched_evaluations,
        total_score_sum=total_score_sum
    )

def draw_wrapped_text(pdf, text, x, y, max_width, font="DejaVu", size=10,
leading=12):
    from reportlab.pdfbase.pdfmetrics import stringWidth
    pdf.setFont(font, size)
    words = text.split()
    line = ""
    for word in words:
        test_line = f"{line} {word}".strip()
        if stringWidth(test_line, font, size) <= max_width:
            line = test_line
        else:
            pdf.drawString(x, y, line)
            y -= leading
            line = word
    if line:
        pdf.drawString(x, y, line)

```

```

    y -= leading
    return y

def generate_pdf(student, evaluations):
    from reportlab.pdfbase import pdfmetrics
    from reportlab.pdfbase.ttfonts import TTFont
    from flask_login import current_user
    from datetime import datetime

    font_path = os.path.join('app', 'static', 'fonts', 'DejaVuSans.ttf')
    pdfmetrics.registerFont(TTFont('DejaVu', font_path))

    buffer = BytesIO()
    pdf = canvas.Canvas(buffer, pagesize=A4)
    width, height = A4

    pdf.setFont("DejaVu", 12)
    pdf.drawString(2 * cm, height - 2 * cm, f"Отчёт по студенту:
{student.full_name}")
    pdf.setFont("DejaVu", 12)
    pdf.drawString(2 * cm, height - 2.7 * cm, f"Группа: {student.group}")

    y = height - 4 * cm
    for eval in evaluations:
        lab = Lab.query.filter_by(title=eval.task_name).first()
        max_score = sum(c.max_score for c in
Criterion.query.filter_by(lab_id=lab.id).all()) if lab else 0

        pdf.setFont("DejaVu", 12)
        lab = Lab.query.filter_by(title=eval.task_name).first()
        full_title = f"{eval.task_name}. {lab.description}" if lab else eval.task_name
        pdf.drawString(2 * cm, y, f"{full_title}. Оценка:
{eval.total_score:.2f}/{max_score:.2f}")
        y -= 0.6 * cm

        criteria = json.loads(eval.criteria_json)
        feedback = json.loads(eval.feedback_json) if eval.feedback_json else {}

        pdf.setFont("DejaVu", 12)
        for code, score in criteria.items():
            crit = Criterion.query.filter_by(lab_id=lab.id, code=code).first()
            if crit:
                cdio = crit.cdio_stage or "—"
                desc = crit.description

```

```

        desc_text = f"Критерий {code}. {desc} — {cdio} : {score}"
    else:
        desc_text = f"Критерий {code}. Неизвестный критерий : {score}"

    y = draw_wrapped_text(pdf, desc_text, 2.5 * cm, y, max_width=16 * cm)
    if code in feedback:
        comment = f"Комментарий: {feedback[code]}"
        y = draw_wrapped_text(pdf, comment, 3 * cm, y, max_width=16 * cm)

    pdf.line(2 * cm, y, width - 2 * cm, y)
    y -= 0.5 * cm
    if y < 5 * cm:
        pdf.showPage()
        y = height - 3 * cm

    total_score_sum = sum(e.total_score for e in evaluations if e.total_score is not
None)
    pdf.setFont("DejaVu", 12)
    pdf.drawString(2 * cm, y, f"Общий балл: {total_score_sum:.2f}")
    y -= 1 * cm

    # график
    fig, ax = plt.subplots(figsize=(6, 2.5))

    titles = []
    for e in evaluations:
        short_title = e.task_name[:4] if e.task_name.startswith("ЛП") else e.task_name
        titles.append(short_title)

    scores = [e.total_score for e in evaluations]

    ax.plot(titles, scores, marker='o', color='teal')
    for i, score in enumerate(scores):
        ax.annotate(f"{score:.2f}", (i, score), textcoords="offset points", xytext=(0, -12),
ha='center', fontsize=8)
    ax.set_title("Оценки по лабораторным", fontsize=10)
    ax.set_ylabel("Баллы", fontsize=9)
    ax.set_xlabel("Лабораторные работы", fontsize=9)
    ax.set_ylim(0, max(scores) + 1)
    ax.tick_params(axis='x', labelsize=8)
    ax.tick_params(axis='y', labelsize=8)
    plt.xticks(rotation=45, ha='right')

    fig.tight_layout(rect=(0, 0.1, 1, 1))

```

```

tmpfile = tempfile.NamedTemporaryFile(delete=False, suffix='.png')
fig.savefig(tmpfile.name, dpi=150, bbox_inches='tight')
plt.close(fig)

if y < 9 * cm:
    pdf.showPage()
    y = height - 3 * cm

pdf.drawImage(tmpfile.name, 2 * cm, y - 6 * cm, width=16 * cm, height=6 * cm)
y -= 7.5 * cm

pdf.setFont("DejaVu", 10)
pdf.drawString(2 * cm, y - 0.5 * cm, f'Оценил: {current_user.full_name}')
now = datetime.now().strftime("%d.%m.%Y")
pdf.drawString(2 * cm, y - 1.2 * cm, f'Дата формирования отчёта: {now}')

pdf.save()
tmpfile.close()
os.unlink(tmpfile.name)
buffer.seek(0)
return send_file(buffer, as_attachment=True,
download_name="student_report.pdf", mimetype='application/pdf')

@app.route('/report/pdf')
def export_pdf():
    student_id = request.args.get('student_id', type=int)
    student = Student.query.get(student_id)
    evaluations = Evaluation.query.filter_by(student_id=student_id).all()
    return generate_pdf(student, evaluations)

# Код ORM-моделей (SQLAlchemy) models.py
from app import db
from flask_login import UserMixin
from werkzeug.security import generate_password_hash, check_password_hash
from datetime import datetime

class Instructor(UserMixin, db.Model):
    id = db.Column(db.Integer, primary_key=True)
    full_name = db.Column(db.String(100), nullable=False)
    username = db.Column(db.String(50), unique=True, nullable=False)
    password_hash = db.Column(db.String(128), nullable=False)
    groups = db.relationship('Group', backref='instructor', lazy=True)

    def set_password(self, password):

```

```

        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)

class Group(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(50), nullable=False)
    instructor_id = db.Column(db.Integer, db.ForeignKey('instructor.id'),
nullable=True)

class Student(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    full_name = db.Column(db.String(100), nullable=False)
    group = db.Column(db.String(50), nullable=False)
    evaluations = db.relationship('Evaluation', backref='student', lazy=True)

class Evaluation(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    student_id = db.Column(db.Integer, db.ForeignKey('student.id'), nullable=False)
    lab_id = db.Column(db.Integer, db.ForeignKey('lab.id'), nullable=False)
    task_name = db.Column(db.String(100), nullable=False)
    criteria_json = db.Column(db.Text, nullable=False)
    feedback_json = db.Column(db.Text, nullable=True)
    total_score = db.Column(db.Float)
    submitted_at = db.Column(db.DateTime, default=datetime.utcnow)
    updated_at = db.Column(db.DateTime, default=datetime.utcnow,
onupdate=datetime.utcnow)
    cdio_stage = db.Column(db.String(50))

class Lab(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    title = db.Column(db.String(255), nullable=False)
    description = db.Column(db.Text, nullable=True)
    cdio_stage = db.Column(db.String(50), nullable=True)
    instructor_id = db.Column(db.Integer, db.ForeignKey('instructor.id'),
nullable=True)
    instructor = db.relationship('Instructor', backref='labs')

    criteria = db.relationship('Criterion', backref='lab', lazy=True)
    def __repr__(self):
        return f"<Lab {self.title}>"

class Criterion(db.Model):

```

```

id = db.Column(db.Integer, primary_key=True)
lab_id = db.Column(db.Integer, db.ForeignKey('lab.id'), nullable=False)
code = db.Column(db.String(10), nullable=False)
description = db.Column(db.String(300), nullable=False)
max_score = db.Column(db.Float, nullable=False)
def __repr__(self):
    return f"<Criterion {self.code}: {self.description[:30]}...>"
cdio_stage = db.Column(db.String(50))

# Код для форм ввода оценок (WTForms) forms.py
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, SubmitField
from wtforms.validators import DataRequired, EqualTo, Length

class LoginForm(FlaskForm):
    username = StringField('Имя пользователя', validators=[DataRequired()])
    password = PasswordField('Пароль', validators=[DataRequired()])
    submit = SubmitField('Войти')

class RegisterForm(FlaskForm):
    full_name = StringField('ФИО преподавателя', validators=[DataRequired()])
    username = StringField('Имя пользователя (логин)', validators=[DataRequired()])
    password = PasswordField('Пароль', validators=[DataRequired(), Length(min=5)])
    confirm_password = PasswordField('Повторите пароль',
validators=[DataRequired(), EqualTo('password')])
    submit = SubmitField('Зарегистрировать')

class ChangePasswordForm(FlaskForm):
    old_password = PasswordField('Старый пароль', validators=[DataRequired()])
    new_password = PasswordField('Новый пароль', validators=[DataRequired(),
Length(min=5)])
    confirm_password = PasswordField('Повторите новый пароль',
validators=[DataRequired(), EqualTo('new_password')])
    submit = SubmitField('Сменить пароль')

# Код инициализации Flask-приложения __init__.py
from flask import Flask, request
from flask_sqlalchemy import SQLAlchemy
from flask_wtf import CSRFProtect
from flask_login import LoginManager
import json

app = Flask(__name__)
app.config['SECRET_KEY'] = 'fb97c9dd9bb3e0ba1a3daa707a7488e0'

```

```

app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///../instance/database.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db = SQLAlchemy(app)

csrf = CSRFProtect()
csrf.init_app(app)

@app.template_filter('from_json')
def from_json_filter(s):
    return json.loads(s)

@app.before_request
def disable_csrf_for_login():
    if request.endpoint == 'login':
        csrf._exempt_views.add('login')

# создание LoginManager
login_manager = LoginManager()
login_manager.login_view = 'login'
login_manager.login_message = "Пожалуйста, авторизуйтесь для доступа к этой
странице."
login_manager.init_app(app)

# импорт модели
from app.models import Instructor

@login_manager.user_loader
def load_user(user_id):
    return Instructor.query.get(int(user_id))

# импорт маршрутов
from app import routes

# Код главной страницы index.html
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Главная — Система оценивания</title>
    <link rel="stylesheet" href="{ { url_for('static', filename='styles.css') } }">
</head>
<body>
    {% if current_user.is_authenticated %}

```

```

    <p>Вы вошли как: <strong>{{ current_user.full_name }}</strong> | <a
href="{{ url_for('profile') }}">Личный кабинет</a> | <a
href="{{ url_for('logout') }}">Выйти</a></p>
    {% else %}
    <p><a href="{{ url_for('login') }}">Войти</a></p>
    {% endif %}

<h1>Система оценки знаний и цифрового следа</h1>
<h3>Выберите действие:</h3>
<div class="card-container">
    <a href="{{ url_for('student_list') }}" class="card">
        
        <span>Оценить студента</span>
    </a>
    <a href="{{ url_for('trace') }}" class="card">
        
        <span>Цифровой след</span>
    </a>
    <a href="{{ url_for('report') }}" class="card">
        
        <span>Отчёт успеваемости студента</span>
    </a>
</div>
</body>
</html>

```

Код формы оценивания evaluate.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Оценка студента</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <style>
    .flash-message {
        position: fixed;
        top: 20px;
        right: 20px;
        background-color: #d4edda;
        color: #155724;
        border: 1px solid #c3e6cb;
        padding: 10px 20px;
        border-radius: 8px;
        z-index: 9999;
    }
    
```

```

    box-shadow: 0 2px 8px rgba(0,0,0,0.2);
    font-weight: bold;
    opacity: 1;
    transition: opacity 0.5s ease;
  }
</style>
</head>
<body>
  {% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
      {% for category, message in messages %}
        <div class="flash-message" id="flash">{{ message }}</div>
      {% endfor %}
    {% endif %}
  {% endwith %}
<script>
  window.onload = function () {
    const flash = document.getElementById('flash');
    if (flash) {
      setTimeout(() => {
        flash.style.opacity = '0';
        setTimeout(() => flash.remove(), 500);
      }, 3000);
    }
  };
</script>

<h1>Оценка: {{ student.full_name }}</h1>
<form method="get">
  <label for="lab_id">Выберите лабораторную работу:</label>
  <select name="lab_id" id="lab_id" onchange="this.form.submit()">
    <option value="">-- Выберите --</option>
    {% for lab in labs %}
      <option value="{{ lab.id }}" {% if lab.id == selected_lab_id %}selected{%
endif %}>
        {{ lab.title }}. {{ lab.description }}
      </option>
    {% endfor %}
  </select>
</form>

{% if criteria %}
<form method="post">
  <input type="hidden" name="csrf_token" value="{{ csrf_token }}">

```

```

<input type="hidden" name="lab_id" value="{{ selected_lab_id }}">
<table border="1" cellpadding="5">
  <tr>
    <th>№</th>
    <th>Критерии оценивания</th>
    <th>Макс. балл</th>
    <th>Оценка</th>
    <th>Комментарий</th>
  </tr>
  {% for criterion in criteria %}
  <tr>
    <td>{{ criterion.code }}</td>
    <td>{{ criterion.description }}</td>
    <td>{{ criterion.max_score }}</td>
    <td>
      <input type="number" step="0.01" max="{{ criterion.max_score }}"
name="score_{{ criterion.id }}" required>
    </td>
    <td>
      <textarea name="feedback_{{ criterion.id }}" rows="1"
cols="25"></textarea>
    </td>
  </tr>
  {% endfor %}
</table>
<button type="submit">Сохранить оценку</button>
</form>
{% endif %}

<p><a href="{{ url_for('student_list') }}">← Назад к Списку
студентов</a></p>
</body>
</html>

```

Код списка студентов students.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Список студентов</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
  <h1>Список студентов</h1>

```

```

<h3>Выберите группу:</h3>
{% for group in groups %}
    {% if group == selected_group %}
        <strong>[ {{ group }} ]</strong>
    {% else %}
        <a href="{{ url_for('student_list', group=group) }}">[ {{ group }} ]</a>
    {% endif %}
{% endfor %}
<hr>
{% if selected_group %}
    <h2>Студенты группы: {{ selected_group }}</h2>
    <p><strong>Чтобы оценить — выберите студента из списка ниже и
нажмите «Оценить».</strong></p>
{% endif %}
{% if students %}
    <table border="1">
        <tr>
            <th>ID</th>
            <th>ФИО</th>
            <th>Группа</th>
            <th>Действие</th>
        </tr>
        {% for student in students %}
            <tr>
                <td>{{ student.id }}</td>
                <td>{{ student.full_name }}</td>
                <td>{{ student.group }}</td>
                <td><a href="{{ url_for('evaluate',
student_id=student.id) }}">Оценить</a></td>
            </tr>
        {% endfor %}
    </table>
{% else %}
    <p>Нет студентов в выбранной группе.</p>
{% endif %}

<p><a href="{{ url_for('index') }}">← Назад на Главную</a></p>
</body>
</html>

```

```

# Код списка лабораторных работ labs.html
<!DOCTYPE html>
<html lang="ru">
<head>

```

```

<meta charset="UTF-8">
<title>Мои работы</title>
<link rel="stylesheet" href="{ { url_for('static', filename='styles.css') } } ">
</head>
<body>
  <h1>Лабораторные работы</h1>

  <table>
    <thead>
      <tr>
        <th>№</th>
        <th>Название работы</th>
        <th>Критерии оценивания</th>
        <th>Этап CDIO</th>
        <th>Макс. балл</th>
      </tr>
    </thead>
    <tbody>
      {% for lab in labs %}
        {% set lab_number = loop.index %}
        {% set criteria = lab.criteria %}
        {% for criterion in criteria %}
          <tr>
            {% if loop.first %}
              <td rowspan="{ { criteria|length } }">{{ lab_number }}</td>
              <td rowspan="{ { criteria|length } }">{{ lab.title }}.
            {% else %}
              <td>{{ criterion.code }}. {{ criterion.description }}</td>
            {% endif %}
            <td>{{ criterion.cdio_stage or "—" }}</td>
            <td>{{ "%.1f"|format(criterion.max_score) }}</td>
          </tr>
        {% endfor %}
      {% endfor %}
    </tbody>
  </table>

  <p><a href="{ { url_for('profile') } } ">← Назад в Личный кабинет</a></p>
</body>
</html>

```

Код страницы «Цифровой след» trace.html

```

<!DOCTYPE html>
<html lang="ru">

```

```

<head>
  <meta charset="UTF-8">
  <title>Цифровой след</title>
  <link rel="stylesheet" href="{ { url_for('static', filename='styles.css') } }">
</head>
<body>
  <h1>Цифровой след обучающегося</h1>

  <form method="get">
    <label for="group">Выберите группу:</label>
    <select name="group" id="group" onchange="this.form.submit()">
      <option value="">-- Все группы --</option>
      {% for group in groups %}
        <option value="{{ group }}" {% if group == selected_group %}selected{%
endif %}>{{ group }}</option>
      {% endfor %}
    </select>

    {% if students %}
      <label for="student_id">Студент:</label>
      <select name="student_id" id="student_id" onchange="this.form.submit()">
        <option value="">-- Выберите --</option>
        {% for s in students %}
          <option value="{{ s.id }}" {% if s.id == selected_student_id %}selected{%
endif %}>{{ s.full_name }}</option>
        {% endfor %}
      </select>
    {% endif %}
  </form>

  {% if student %}
    <h2>Студент: {{ student.full_name }}</h2>
    {% if evaluations %}
      <table>
        <thead>
          <tr>
            <th>Название работы</th>
            <th>Критерии и баллы</th>
            <th>Этап CDIO</th>
            <th>Комментарии</th>
            <th>Итоговая оценка</th>
            <th>Дата оценивания</th>
          </tr>
        </thead>

```

```

<tbody>
  {% for eval in evaluations %}
    {% set criteria_items = eval.criteria.items() %}
    {% set row_count = criteria_items|length %}
    {% for code, val in criteria_items %}
      <tr{% if loop.last %} {% endif %}>
        {% if loop.first %}
          <td rowspan="{{ row_count }}">{{ eval.task_name }}</td>
        {% endif %}
        <td>
          {% set full_key = eval.lab_id ~ ':' ~ code %}
          {% set desc = criteria_map[full_key]|default('Неизвестный
критерий') %}
          {{ desc }} : <strong>{{ val }}</strong>
        </td>
        <td>{{ criteria_cdio[full_key]|default('—') }}</td>
        <td>{{ eval.feedback[code]|default('—') }}</td>
        {% if loop.first %}
          <td
rowspan="{{ row_count }}"><strong>{{ "%.2f"|format(eval.total_score) }}</strong>
> / {{ "%.2f"|format(eval.max_score) }}</td>
          <td rowspan="{{ row_count }}">{{ eval.updated_at }}</td>
        {% endif %}
      </tr>
    {% endfor %}
  {% endfor %}
</tbody>
</table>
{% if graph_base64 %}
  <h3>График успеваемости</h3>
  
{% endif %}
{% else %}
  <p>Нет записей оценивания.</p>
{% endif %}
{% endif %}

  <p><a href="{{ url_for('index') }}">← Назад на Главную</a></p>
</body>
</html>

```

Код страницы «Отчёт успеваемости студента» report.html
 <!DOCTYPE html>

```

<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Отчёт успеваемости</title>
  <link rel="stylesheet" href="{ { url_for('static', filename='styles.css') } }">
</head>
<body>
  <h1>Отчёт успеваемости студента</h1>

  <form method="get" action="/report">
    <label for="group">Выберите группу:</label>
    <select name="group" id="group" onchange="this.form.submit()">
      <option value="">-- Все группы --</option>
      {% for group in groups %}
        <option value="{{ group }}" {% if group == selected_group %}selected{%
endif %}>{{ group }}</option>
      {% endfor %}
    </select>

    {% if students %}
      <label for="student_id">Студент:</label>
      <select name="student_id" id="student_id" onchange="this.form.submit()">
        <option value="">-- Выберите --</option>
        {% for s in students %}
          <option value="{{ s.id }}" {% if s.id == selected_student_id %}selected{%
endif %}>{{ s.full_name }}</option>
        {% endfor %}
      </select>
    {% endif %}
  </form>

  {% if student %}
    <h2>Студент: {{ student.full_name }}</h2>
    {% if enriched_evaluations %}
      <form method="get" action="{ { url_for('export_pdf') } }">
        <input type="hidden" name="student_id"
value="{ { selected_student_id } }">
        <button type="submit">Скачать PDF-файл отчёта</button>
      </form>

      <h2>Оценки по лабораторным</h2>
      <table border="1" cellpadding="6">
        <tr>
          <th>Название работы</th>

```

```

        <th>Итоговая оценка</th>
    </tr>
    {% for e in enriched_evaluations %}
    <tr>
        <td>{{ e.full_title }}</td>
        <td>{{ "%.2f"|format(e.score) }}</td>
    </tr>
    {% endfor %}
</table>
<p><strong>Общий балл:</strong>
{{ "%.2f"|format(total_score_sum) }}</p>
{% else %}
    <p>Отчёт недоступен — студент ещё не оценён.</p>
{% endif %}
{% endif %}
<br>
<p><a href="{{ url_for('index') }}">← Назад на Главную</a></p>
</body>
</html>

```

Код страницы авторизации auth.html

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Вход/Регистрация</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <style>
        .auth-box {
            max-width: 400px;
            margin: 60px auto;
            background: #fff;
            padding: 30px;
            border-radius: 8px;
            box-shadow: 0 4px 20px rgba(0,0,0,0.1);
        }

        .auth-box h2 {
            text-align: center;
            margin-bottom: 20px;
        }

        .auth-box form p {
            margin: 15px 0;
        }
    </style>

```

```

}

.auth-box input[type="submit"] {
    width: 100%;
    padding: 10px;
}

.toggle-link {
    text-align: center;
    margin-top: 15px;
}

.toggle-link a {
    color: #2980b9;
    text-decoration: none;
    font-style: italic;
}

.toggle-link a:hover {
    text-decoration: underline;
}

.hidden {
    display: none;
}
</style>
</head>
<body>
<div class="auth-box">
    {% with messages = get_flashed_messages(with_categories=True) %}
    {% for category, message in messages %}
        <p style="color: red;">{{ message }}</p>
    {% endfor %}
    {% endwith %}

    <div id="login-form">
        <h3>Войти</h3>
        <form method="POST" action="{% url_for('login') %}">
            {{ login_form.hidden_tag() }}
            <p>{{ login_form.username.label }}<br>{{ login_form.username(size=32)
    }}</p>
            <p>{{ login_form.password.label }}<br>{{ login_form.password(size=32,
id="login_password") }}
            <input type="checkbox" onclick="toggleLogin()"> Показать</p>

```

```

        <p><input type="submit" name="login_submit" value="Войти"></p>
    </form>

    <div class="toggle-link">
        <p>Вы не зарегистрированы? <a href="#"
onclick="switchForm('register')">Перейти к регистрации</a></p>
    </div>
</div>

<div id="register-form" class="hidden">
    <h3>Регистрация</h3>
    <form method="POST" action="{{ url_for('register') }}">
        {{ register_form.hidden_tag() }}
        <p>{{ register_form.full_name.label }}<br>{{ register_form.full_name(size
e=40) }}</p>
        <p>{{ register_form.username.label }}<br>{{ register_form.username(size
=30) }}</p>
        <p>{{ register_form.password.label }}<br>{{ register_form.password(size
=30, id="reg_pass") }}</p>
        <p>{{ register_form.confirm_password.label }}<br>{{ register_form.confir
m_password(size=30) }}</p>
        <p><input type="submit" name="register_submit"
value="Зарегистрироваться"></p>
    </form>
    <div class="toggle-link">
        <p>Уже есть аккаунт? <a href="#" onclick="switchForm('login')">Назад
к входу</a></p>
    </div>
</div>
</div>

<script>
    function toggleLogin() {
        const field = document.getElementById("login_password");
        field.type = field.type === "password" ? "text" : "password";
    }

    function switchForm(target) {
        document.getElementById("login-form").classList.toggle("hidden", target
=== 'register');
        document.getElementById("register-form").classList.toggle("hidden", target
=== 'login');
    }
</script>

```

```
</body>
</html>
```

Код страницы «Личный кабинет» profile.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Личный кабинет</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
  <h1>Личный кабинет преподавателя</h1>

  <h2>Профиль</h2>
  <p><strong>ФИО:</strong> {{ instructor.full_name }}</p>
  <p><strong>Логин:</strong> {{ instructor.username }}</p>
  <p><a href="{{ url_for('change_password') }}">Сменить пароль</a></p>

  <h2>Мои группы</h2>
  <p><a href="{{ url_for('my_groups') }}">Список студентов и оценки</a></p>

  <h2>Мои работы</h2>
  <p><a href="{{ url_for('labs') }}">Список лабораторных работ</a></p>
  <br>
  <p><a href="{{ url_for('index') }}">← Назад на Главную</a></p>
</body>
</html>
```

Код страницы для смены пароля в личном кабинете change_password.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Смена пароля</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
  <h2>Смена пароля</h2>
  {% with messages = get_flashed_messages(with_categories=true) %}
    {% for category, message in messages %}
      <p style="color: red;">{{ message }}</p>
    {% endfor %}
  {% endwith %}
```

```

<form method="POST">
    {{ form.hidden_tag() }}
    <p>{{ form.old_password.label }}<br>{{ form.old_password(size=32) }}</p>
    <p>{{ form.new_password.label }}<br>{{ form.new_password(size=32) }}</p>
>
    <p>{{ form.confirm_password.label }}<br>{{ form.confirm_password(size=32)
    }}</p>
    <p>{{ form.submit() }}</p>
</form>
<p><a href="{{ url_for('profile') }}">← Назад в Личный кабинет</a></p>
</body>
</html>

```

Код страницы «Мои группы» в личном кабинете

```

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Мои группы</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
</head>
<body>
    <h1>Список студентов</h1>
    <h3>Выберите группу:</h3>
    {% for group in groups %}
        {% if group == selected_group %}
            <strong>[ {{ group }} ]</strong>
        {% else %}
            <a href="{{ url_for('my_groups', group=group) }}">[ {{ group }} ]</a>
        {% endif %}
    {% endfor %}
    <hr>
    {% if selected_group %}
        <h2>Студенты группы: {{ selected_group }}</h2>
    {% endif %}
    {% if students %}
        <table border="1">
            <tr>
                <th>ID</th>
                <th>ФИО</th>
                <th>Группа</th>
                <th>Общий балл по лабораторным</th>
            </tr>

```

```

        {% for student in student_data %}
            <tr>
                <td>{{ student.id }}</td>
                <td>{{ student.full_name }}</td>
                <td>{{ student.group }}</td>
                <td>{{ student.total_score }}</td>
            </tr>
        {% endfor %}
    </table>
{% else %}
    <p>Нет студентов в выбранной группе.</p>
{% endif %}

    <p><a href="{{ url_for('profile') }}">← Назад в Личный кабинет</a></p>
</body>
</html>

```

Код импорта списка студентов из CSV-файла в базу данных

sync_and_import_students.py

```
import csv
```

```
from app import app, db
```

```
from app.models import Student
```

```
def get_students_from_csv(filename):
```

```
    students = []
```

```
    with open(filename, newline="", encoding='utf-8') as csvfile:
```

```
        reader = csv.DictReader(csvfile)
```

```
        for row in reader:
```

```
            id_ = int(row['id'].strip())
```

```
            full_name = row['full_name'].strip()
```

```
            group = row['group'].strip()
```

```
            students.append({'id': id_, 'full_name': full_name, 'group': group})
```

```
    return students
```

```
def sync_and_import(filename):
```

```
    with app.app_context():
```

```
        students_in_csv = get_students_from_csv(filename)
```

```
        # удаление тех, кого нет в CSV
```

```
        existing_students = Student.query.all()
```

```
        csv_keys = {(s['full_name'], s['group']) for s in students_in_csv}
```

```
        removed = 0
```

```
        for student in existing_students:
```

```
            key = (student.full_name.strip(), student.group.strip())
```

```

        if key not in csv_keys:
            db.session.delete(student)
            removed += 1
    db.session.commit()

    # добавление новых студентов из CSV
    added = 0
    for s in students_in_csv:
        exists = Student.query.filter_by(id=s['id']).first()
        if not exists:
            new_student = Student(id=s['id'], full_name=s['full_name'],
group=s['group'])
            db.session.add(new_student)
            added += 1
    db.session.commit()

    print(f'Синхронизация завершена. Удалено: {removed} | Добавлено:
{added}')

if __name__ == '__main__':
    sync_and_import('students.csv')

# Список студентов students.csv в виде CSV-файла для удобства импорта в
базу данных
id,full_name,group
1,Асыллова Анар,Computer Science
2,Бекзат Еркебулан,Computer Science
3,Ербаев Касым,Computer Science
4,Никитов Георгий,Computer Science
5,Садыкова Алтынай,Computer Science
6,Смоленко Елизавета,Computer Science
7,Тулегенова Айгерим,Computer Science

# Код импорта списка лабораторных работ из CSV-файла в базу данных
import_criteria.py
import csv
from app import app, db
from app.models import Lab, Criterion

with app.app_context():
    with open('criteria.csv', newline="", encoding='utf-8') as csvfile:
        reader = csv.DictReader(csvfile)
        for row in reader:
            # найти или создать лабораторную работу

```

```

lab = Lab.query.filter_by(title=row['lab_title']).first()
if not lab:
    lab = Lab(
        title=row['lab_title'],
        description=row['lab_description'],
        cdio_stage=row['cdio_stage']
    )
    db.session.add(lab)
    db.session.commit()

# проверка наличия критерия
existing = Criterion.query.filter_by(
    lab_id=lab.id,
    code=row['criterion_code']
).first()

if not existing:
    criterion = Criterion(
        lab_id=lab.id,
        code=row['criterion_code'],
        description=row['criterion_description'],
        max_score=float(row['max_score'])
    )
    db.session.add(criterion)

db.session.commit()
print("Импорт критериев завершён успешно.")

```

Список лабораторных работ criteria.csv в виде CSV-файла для удобства импорта в базу данных

lab_title,lab_description,cdio_stage,criterion_code,criterion_description,max_score
 ЛР 3,Анализ требований и описание бизнес-процесса в IDEF0,Conceive (Этап анализа),1,Спецификация контекстного и иерархических блоков,0.9
 ЛР 3,Анализ требований и описание бизнес-процесса в IDEF0,Design (Этап проектирования),2,Анализ процессов и рекомендации по ТО-БЕ модели,1.2
 ЛР 3,Анализ требований и описание бизнес-процесса в IDEF0,Operate (Этап эксплуатации),3,Качество защиты лабораторной работы,0.6
 ЛР 3,Анализ требований и описание бизнес-процесса в IDEF0,Operate (Этап эксплуатации),4,Своевременность сдачи работы,0.3

ЛР 4,Формализация сущностей и связей в нотации Р.Barkera,Conceive (Этап анализа),1,Качество модели сущностей и связей,1.2

ЛР 4,Формализация сущностей и связей в нотации Р.Barkera,Design (Этап проектирования),2,Связь моделей сущностей с функциональными моделями,0.9

ЛР 4,Формализация сущностей и связей в нотации P.Barkera,Operate (Этап эксплуатации),3,Качество защиты лабораторной работы,0.6
ЛР 4,Формализация сущностей и связей в нотации P.Barkera,Operate (Этап эксплуатации),4,Своевременность сдачи работы,0.3

ЛР 5,Диаграмма вариантов использования в UML,Conceive (Этап анализа),1,Формулировка функциональных требований,1.2
ЛР 5,Диаграммы вариантов использования в UML,Design (Этап проектирования),2,Соответствие объектов спецификациям сценариев,1.6
ЛР 5,Диаграммы вариантов использования в UML,Operate (Этап эксплуатации),3,Качество защиты лабораторной работы,0.8
ЛР 5,Диаграммы вариантов использования в UML,Operate (Этап эксплуатации),4,Своевременность сдачи работы,0.4

ЛР 6,Спецификация потоков событий абстрактных вариантов,Conceive (Этап анализа),1,Спецификация "Организация лекции",0.4
ЛР 6,Спецификация потоков событий абстрактных вариантов,Conceive (Этап анализа),2,Спецификация "Мониторинг знаний",0.4
ЛР 6,Спецификация потоков событий абстрактных вариантов,Design (Этап проектирования),3,Диаграмма навигации интерфейса,1.0
ЛР 6,Спецификация потоков событий абстрактных вариантов,Design (Этап проектирования),4,Обоснование выбора ИТ-инструментов,1.0
ЛР 6,Спецификация потоков событий абстрактных вариантов,Implement (Этап реализации),5,Описание тестов и сценариев,1.2

ЛР 7,Поведение объектов и аналитическая модель,Design (Этап проектирования),1,Диаграммы по сценарию "вопрос-ответ",2.0
ЛР 7,Поведение объектов и аналитическая модель,Design (Этап проектирования),2,Диаграммы по тестовому сценарию,2.0

ЛР 8,Диаграммы состояний управляющих объектов,Implement (Этап реализации),1,Состояния и переходы по сценарию "вопрос-ответ",2.0
ЛР 8,Диаграммы состояний управляющих объектов,Implement (Этап реализации),2,Состояния и переходы по тестовому сценарию,2.0

ЛР 9,Проектная документация и спецификации,Design (Этап проектирования),1,Разработка проектной документации (артефакты),1.6
ЛР 9,Проектная документация и спецификации,Design (Этап проектирования),2,Обоснование проектных решений (Conceive+Design),1.6
ЛР 9,Проектная документация и спецификации,Operate (Этап эксплуатации),3,Оценка и перспективы развития проекта,0.8

Файл requirements.txt – список зависимостей
blinker==1.9.0

click==8.2.1
colorama==0.4.6
contourpy==1.3.2
cyclr==0.12.1
Flask==3.1.1
Flask-SQLAlchemy==3.1.1
Flask-WTF==1.2.2
fonttools==4.58.0
greenlet==3.2.2
itsdangerous==2.2.0
Jinja2==3.1.6
kiwisolver==1.4.8
MarkupSafe==3.0.2
matplotlib==3.10.3
narwhals==1.40.0
numpy==2.2.6
packaging==25.0
pandas==2.2.3
pillow==11.2.1
plotly==6.1.1
pyparsing==3.2.3
python-dateutil==2.9.0.post0
pytz==2025.2
six==1.17.0
SQLAlchemy==2.0.41
typing_extensions==4.13.2
tzdata==2025.2
Werkzeug==3.1.3
WTForms==3.2.1