КГКП «Сарыкольский колледж агробизнеса и права»
Управления образования акимата Костанайской области

РАСШИРЕНИЕ BRISKTEACHING КАК ИНСТРУМЕНТ ЦИФРОВОЙ ТРАНСФОРМАЦИИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА.

(методические рекомендации)

Сарыколь, 2025

Автор: ***Тулемисова Меруерт Конасбаевна**, преподаватель специальных дисциплин КГКП «Сарыкольский колледж агробизнеса и права» УОАКО.*

Методические рекомендации предназначены для администрации колледжа, методистов, заведующих отделениями, кураторов учебных групп и преподавателей. Пособие призвано содействовать внедрению в образовательный процесс расширения BriskTeaching как современного цифрового ресурса, основанного на технологиях искусственного интеллекта, повышению качества обучения и развитию цифровых компетенций педагогических кадров.

СОДЕРЖАНИЕ

Пояснительная записка

Расширение BriskTeaching как инструмент цифровой трансформации образовательного процесса

В условиях цифровой трансформации образования особое значение приобретает внедрение инновационных технологий, направленных на повышение эффективности обучения и развитие цифровых компетенций педагогов. Быстрое развитие информационных технологий, внедрение искусственного интеллекта и необходимость подготовки конкурентоспособных специалистов диктуют новые требования к образовательным организациям.

В этом контексте цифровые инструменты становятся неотъемлемой частью современной педагогической практики. Одним из таких решений является расширение **BriskTeaching**, которое автоматизирует создание учебных материалов, упрощает планирование и оценивание, повышает вовлечённость студентов. BriskTeaching выступает цифровым ассистентом педагога, интегрируется с привычными сервисами (Google Docs, Slides, YouTube и др.) и предлагает широкий спектр шаблонов для организации образовательной деятельности.

Актуальность методических рекомендаций обусловлена потребностью в современных и доступных инструментах, способствующих:
• экономии времени педагогов за счёт автоматизации рутинных задач;
• повышению прозрачности и объективности оценивания;
• развитию критического мышления и самостоятельности студентов;
• внедрению элементов искусственного интеллекта для улучшения качества обучения.

Настоящее пособие разработано с целью оказания практической помощи педагогам и администрации колледжа в освоении и эффективном применении BriskTeaching в учебно-воспитательном процессе. Документ включает описание возможностей инструмента, примеры использования шаблонов и советы по адаптации сгенерированных материалов под требования образовательных стандартов.

Рекомендации охватывают ключевые аспекты работы с BriskTeaching — от установки и базовых функций до применения интерактивных инструментов для активизации познавательной деятельности студентов. Особое внимание уделено вопросам практического применения: созданию поурочных планов, тестов, презентаций, контрольно-измерительных материалов и критериев оценивания.

Документ предназначен для администрации колледжа, методистов, заведующих отделениями, кураторов учебных групп и преподавателей, заинтересованных во внедрении современных цифровых инструментов, повышении качества обучения и развитии цифровых компетенций педагогических кадров.

Введение

1.1. Что такое BriskTeaching?

По мере того, как сфера образования продолжает развиваться, интеграция инструментов искусственного интеллекта (ИИ) становится всё более распространённой. У педагогов возникает потребность в ресурсах, которые обеспечивают широкий спектр функциональных возможностей на базе нейросетей: от создания презентаций и планов уроков до получения обратной связи и составления оценочных материалов.

BriskTeaching — это универсальное расширение для браузера Chrome, включающее в себя возможности искусственного интеллекта для оказания помощи как преподавателям, так и обучающимся в выполнении педагогических и проектных задач. Оно позволяет создавать документы на основе веб-ресурсов, формировать отзывы о работах студентов, разрабатывать проекты и готовить интерактивные тесты для проверки знаний. Кроме того, расширение автоматизирует создание планов занятий, презентаций, лабораторных и исследовательских заданий, что значительно экономит время педагога.

Таким образом, BriskTeaching выступает современным инструментом цифровой трансформации образовательного процесса, упрощая подготовку к занятиям и повышая эффективность обучения.

1.2. Особенности и преимущества расширения

BriskTeaching выступает как комплексное решение для оптимизации учебного процесса, позволяя педагогам создавать персонализированные, качественные и многоязычные образовательные ресурсы с минимальными затратами времени.

Ключевые преимущества:

- Экономия времени за счёт автоматизированного создания планов уроков, рабочих листов, презентаций, словарей и тестов на основе веб-страниц, документов и мультимедийных источников.
- Адаптивность материалов под различные уровни подготовки обучающихся, включая возможность изменения сложности текста и перевода на 48 языков.
- Интеграция с цифровыми сервисами (Google Docs, Google Slides, Google Classroom, Google Drive, YouTube), что обеспечивает удобное хранение, синхронизацию и совместное использование ресурсов.
- Интерактивность благодаря инструментам для создания увлекательных заданий и тестов, повышающих вовлечённость обучающихся.
- Аналитика и обратная связь: проверка письменных работ на орфографию, стиль, структуру и плагиат с возможностью комментирования прямо в Google Docs.
- Универсальность: поддержка текстов на русском языке и обработка материалов из различных источников (PDF, Word, PowerPoint, открытые веб-страницы).

- Доступность: простая установка как расширения Chrome и наличие бесплатного базового функционала с возможностью подключения расширенных премиум-инструментов для образовательных организаций.

1.3. Сравнение с другими сервисами на основе ИИ.

Среди современных AI-инструментов для образования можно выделить ряд сервисов, схожих по функциональности с BriskTeaching:

- *Synthesis Tutor* — облегчает изучение математики, делая сложные темы доступными и понятными.
- *SchooLinks* — платформа для отслеживания учебного прогресса и планирования профессионального развития.
- *StoryShots* — микролернинг-сервис, предоставляющий ключевые идеи из нон-фикшн литературы в различных форматах.
- *Eduaide* — AI-платформа для создания занятий, учебных материалов и персонализированной обратной связи.
- *Cognii* — инструмент диалогового обучения с использованием искусственного интеллекта.
- *Classkick* — сервис для разработки и выполнения интерактивных заданий.
- *Edulastic* — система для оценки знаний с интеграцией в популярные образовательные платформы.

В отличие от перечисленных решений, BriskTeaching имеет следующие преимущества:

- глубокая интеграция с Google Workspace (Docs, Slides, Forms, Classroom, Drive), позволяющая автоматически создавать и хранить материалы;
- возможность персонализации и адаптации заданий под уровень подготовки студента;
- предоставление обратной связи в реальном времени по письменным работам с анализом орфографии, структуры текста, стиля и уникальности;
- широкий спектр инструментов: от планов занятий и тестов до словарей, презентаций и интерактивных упражнений.

Если многие альтернативы решают отдельные задачи (создание заданий, контроль знаний или сопровождение учебного прогресса), то BriskTeaching выступает как комплексное AI-решение, оптимизированное для эффективной работы со студентами и организации занятий в цифровой экосистеме Google.

2. Установка и интеграция BriskTeaching.

2.1. Порядок установки из Chrome Web Store. Интеграция.

Для установки расширения необходимо выполнить следующие действия:

1. Перейдите в магазин расширений Google Chrome или на сайт https://www.briskteaching.com/ru

2. Найдите и установите "Brisk Teaching"
3. Согласитесь с условиями использования
4. Создайте аккаунт с помощью своей учетной записи Google
После установки и регистрации расширение становится доступным в панели браузера Chrome.

Расширение обрабатывает и генерирует материалы на основе следующих источников: YouTube-видео, Google Docs, Google Slides, Word Doc, PDF-файлы, веб-страницы, изображения.
Поддерживается генерация и перевод учебных материалов на 64 языка.

2.2. Интерфейс расширения и основные элементы.

После установки значок BriskTeaching отображается в панели браузера и интегрируется в используемые сервисы. На нижней панели YouTube, Google Docs и других ресурсов появляется кнопка с доступом к функциям.
Кликнув по ней, открывается контекстное окно (Рис. 2. Основные функции) с предложением выполнить различные действия.

- Create — создать материал по шаблону (тест, план, ресурс и т.д.)
- Give Feedback — добавить комментарии или обратную связь к тексту
- Inspect Writing — проверить текст на признаки использования искусственного интеллекта
- Change Level — изменить уровень сложности материала
- Boost Student Activity — активизировать участие студентов с помощью специальных инструментов (тьютер, чат-персонажи, дебаты и др.)

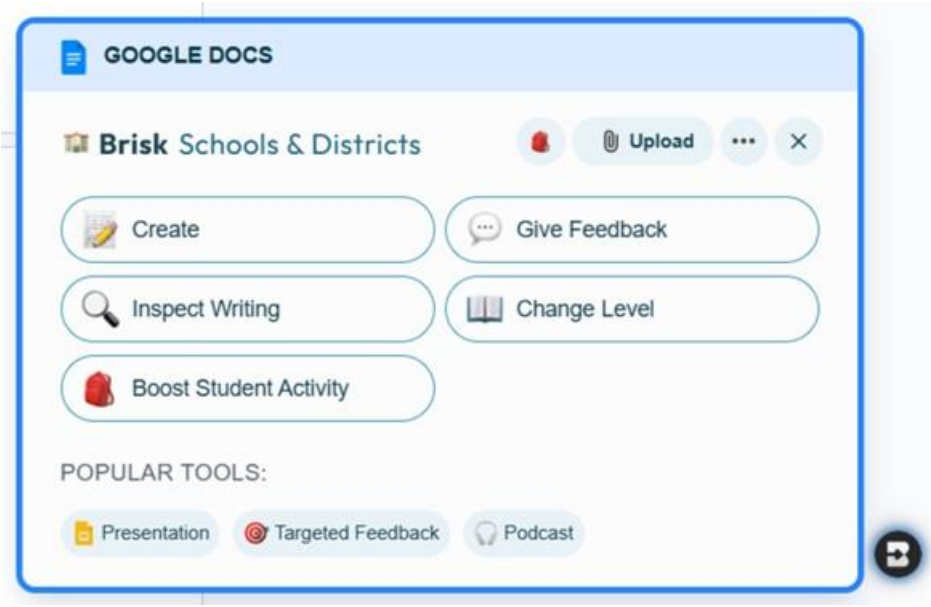

Рис. 2. Основные функции

3. Вкладка Create

После того как вы заходите во вкладку Create (Рис. 3. Вкладка Create), перед вами открывается окно с шаблонами. Эта вкладка предназначена для

упрощения работы преподавателя: здесь собраны готовые формы для разных задач. По сути, Create – это библиотека шаблонов, которые помогают быстро создать нужный документ, будь то для учебного процесса, административной деятельности или индивидуальных планов поддержки студентов.

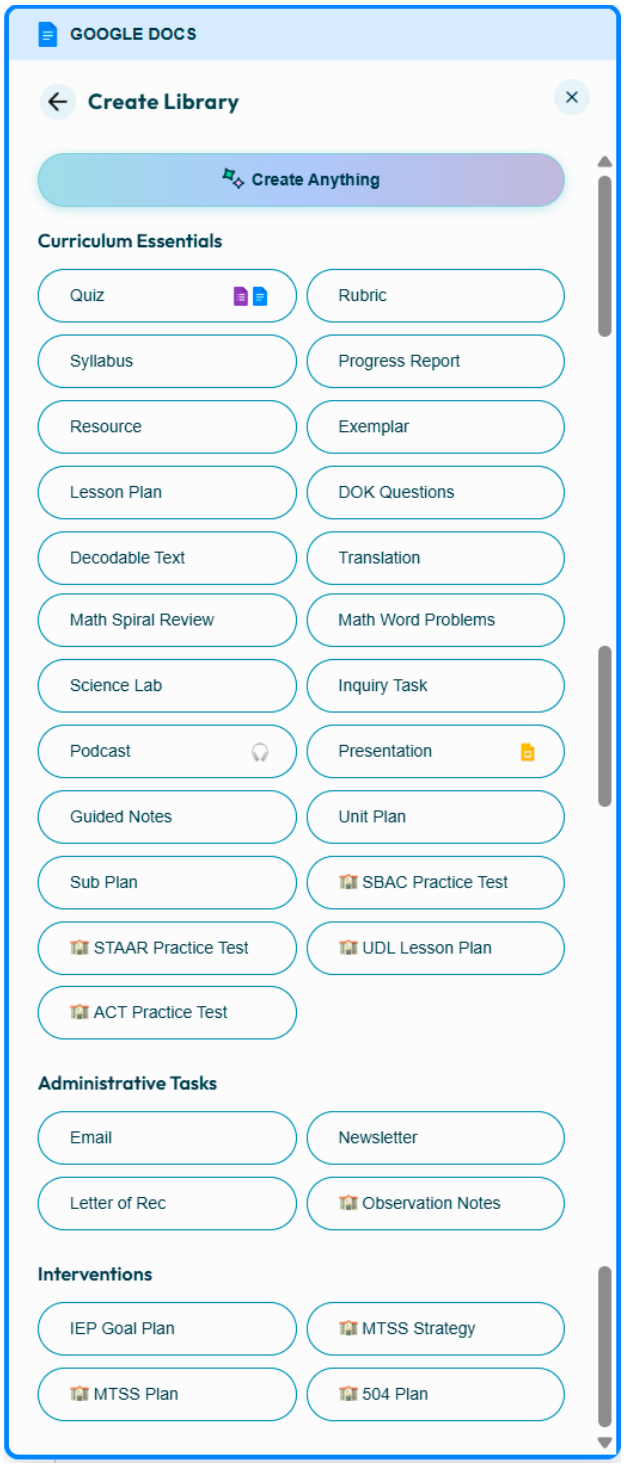

Рис. 3. Вкладка Create

Первая область называется Curriculum Essentials – основа учебного процесса. В этом блоке можно выбрать шаблоны для составления плана урока,

создания тестов, рубрик, программы курса, отчётов об успеваемости, а также дополнительных учебных материалов. Среди них:

- Quiz — тест (краткая проверка знаний).
- Rubric — рубрика для оценки (критерии и шкала оценки).
- Syllabus — программа курса (план тем и требований).
- Progress Report — отчёт об успеваемости.
- Resource — учебный ресурс (листовки, материалы).
- Exemplar — образец работы (пример выполнения задания).
- Lesson Plan — план урока.
- DOK Questions — вопросы по уровням сложности (Depth of Knowledge).
- Decodable Text — читаемый по уровням текст (для обучения чтению).
- Translation — перевод (шаблон/вариант перевода материалов).
- Math Spiral Review — спиральное повторение математических тем.
- Math Word Problems — текстовые (словесные) задачи по математике.
- Science Lab — лабораторная работа (план/карточка для эксперимента).
- Inquiry Task — исследовательская/проективная задача.
- Podcast — подкаст (сценарий или структура эпизода).
- Presentation — презентация (шаблон слайдов/плана выступления).
- Guided Notes — направленные конспекты (структурированные заметки для учащихся).
- Unit Plan — план учебного блока/модуля.
- Sub Plan — план урока для замены (для substitute teacher).
- SBAC Practice Test — практический тест формата SBAC.
- STAAR Practice Test — практический тест формата STAAR.
- UDL Lesson Plan — урок по принципам UDL (универсальный дизайн обучения).
- Standards Unpacker — разбор учебных стандартов (детализация целей/компетенций).
- SAT Practice Test — практический тест формата SAT.
- ACT Practice Test — практический тест формата ACT.

Всё это помогает педагогу организовать и структурировать образовательный процесс.

Вторая область – Administrative Tasks (административные задачи). Здесь собраны шаблоны для административной работы:

- Email – письмо.
- Newsletter – информационная рассылка.
- Letter of Rec – рекомендательное письмо.
- Observation Notes – заметки наблюдения.

А третья область – Interventions (поддержка обучающихся с особыми образовательными потребностями). В этом разделе можно составить:

- IEP Goal Plan – индивидуальный образовательный план для обучающегося с особыми образовательными потребностями.
- 504 Plan – план обеспечения равных условий для студента (адаптация среды, материалов и пр.).

- MTSS Plan – многоуровневая система поддержки обучения студента.
- MTSS Strategy – стратегия поддержки обучающегося в рамках MTSS.

Эти шаблоны помогают адаптировать обучение под конкретные нужды учащихся.

4. Вкладка Inspect Writing.

Вкладка Inspect Writing расширения *BriskTeaching* доступна пользователям по подписке. Она предназначена для анализа процесса написания письменных заданий студентами и позволяет в формате видео-повтора просматривать весь путь работы над текстом — от первого черновика до финальной версии.

Основные возможности:

- Отслеживание процесса написания: фиксация времени выполнения, последовательности действий и всех внесённых изменений.
- Выявление заимствований: определение скопированных или вставленных фрагментов текста, что способствует поддержанию академической честности.
- Анализ использования AI-инструментов: возможность установить, создан ли текст самостоятельно или с применением искусственного интеллекта.
- Формирование адресной обратной связи: преподаватель получает данные о том, как именно студент работал с текстом, что помогает корректнее выстраивать рекомендации и поддержку.
- Экономия времени педагога: визуализация процесса и автоматический отчёт упрощают проверку и оценивание письменных работ.

Использование вкладки Inspect Writing обеспечивает более глубокое понимание того, как студенты создают тексты, повышает объективность оценивания, способствует академической честности и помогает в развитии навыков письма у обучающихся.

5. Вкладка Feedback Styles

Вкладка **Feedback Styles** предоставляет преподавателям широкий спектр стилей обратной связи для оценки и комментирования письменных работ обучающихся. Она позволяет гибко организовывать процесс рецензирования за счёт настройки критериев и целей оценивания, что делает отзывы персонализированными и релевантными. Интеграция с Google Docs обеспечивает удобное добавление комментариев и аналитических таблиц непосредственно в документ обучающегося, а оформление отзывов в виде таблиц или других структурированных форм облегчает их восприятие и систематизацию.

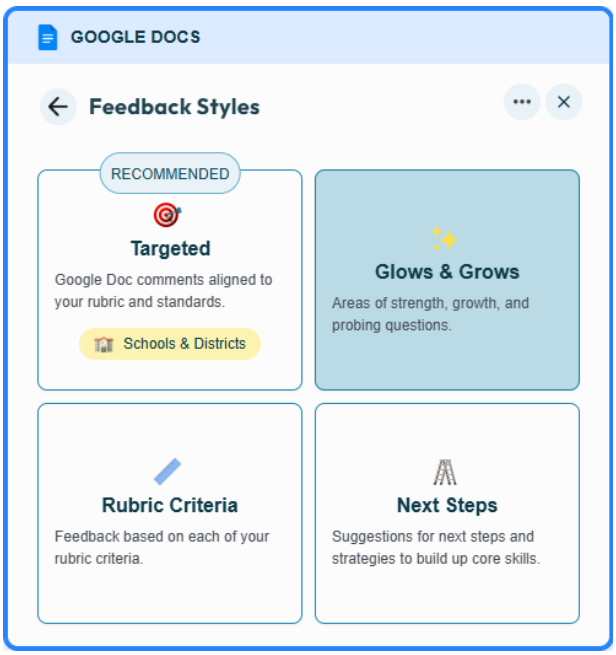

Рис. 5 Вкладка Feedback Styles

Выбор стиля обратной связи:

- Targeted — комментарии к отдельным фрагментам текста в соответствии с критериями (доступна по подписке).
- Glow and Grow — выделение сильных сторон работы и зон для улучшения с уточняющими вопросами.
- Rubric Criteria — оценка по заданным критериям рубрики (язык, стиль, структура, критическое мышление и др.).
- Next Steps — конкретные рекомендации и шаги для дальнейшего улучшения.

Такой подход помогает экономить время педагога, повышать качество коммуникации и эффективно поддерживать развитие навыков письма у обучающихся.

6. Вкладка Change Level.

Вкладка **«Change Level»** обеспечивает адаптацию учебных материалов к уровню подготовки обучающихся. Она автоматически определяет сложность текста и позволяет скорректировать её в пределах от дошкольного до высшего уровня образования (grade level). С помощью встроенных инструментов искусственного интеллекта материалы могут быть упрощены либо усложнены в зависимости от образовательных задач.

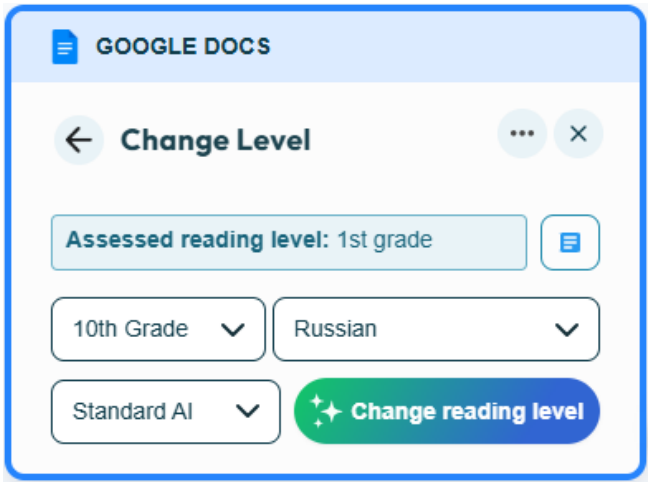

Рис.6 Вкладка Change Level.

7. Вкладка Boost Student Activity.

Boost Student Activity – это инструменты, направленные на активное вовлечение студентов в учебный процесс, помогая преподавателям превращать любой образовательный контент в интерактивные и персонализированные учебные задания с поддержкой искусственного интеллекта. Вкладка подразделяется на три секции:

- Engage помогает вовлечь студентов в учебный процесс через пошаговое сопровождение, интерактивное объяснение и оценку понимания материала в ходе занятия. Основные возможности:
  - Tutor – Тьютер (помогает разобраться в теме, задавая наводящие вопросы)
  - Hook – Зацепка / Яркое вступление (мотивация, интерес к теме)
  - Character Chat – Чат с персонажем (например, Шекспир, Наполеон)
  - Debate – Дебаты (создание задания или примера для дискуссии)
  - Socratic Questions – Сократические вопросы (поощрение критического мышления)
  - Inquiry – Исследование (вопросы, гипотезы, анализ информации)
  - Act it Out – Ролевая игра / Инсценировка
  - Brainstorm – Мозговой штурм
  - Reflect – Рефлексия (что я понял, что вызвало затруднение)
  - Exit Ticket – Рефлексия в конце урока (маленькое задание на выход)
- Assess позволяет определить, насколько студент владеет материалом на начальном этапе, а также как он усвоил новый материал.
- Write содержит инструменты, которые помогают студентам при написании текстов — будь то эссе, аргументированные ответы или пояснительные записки.

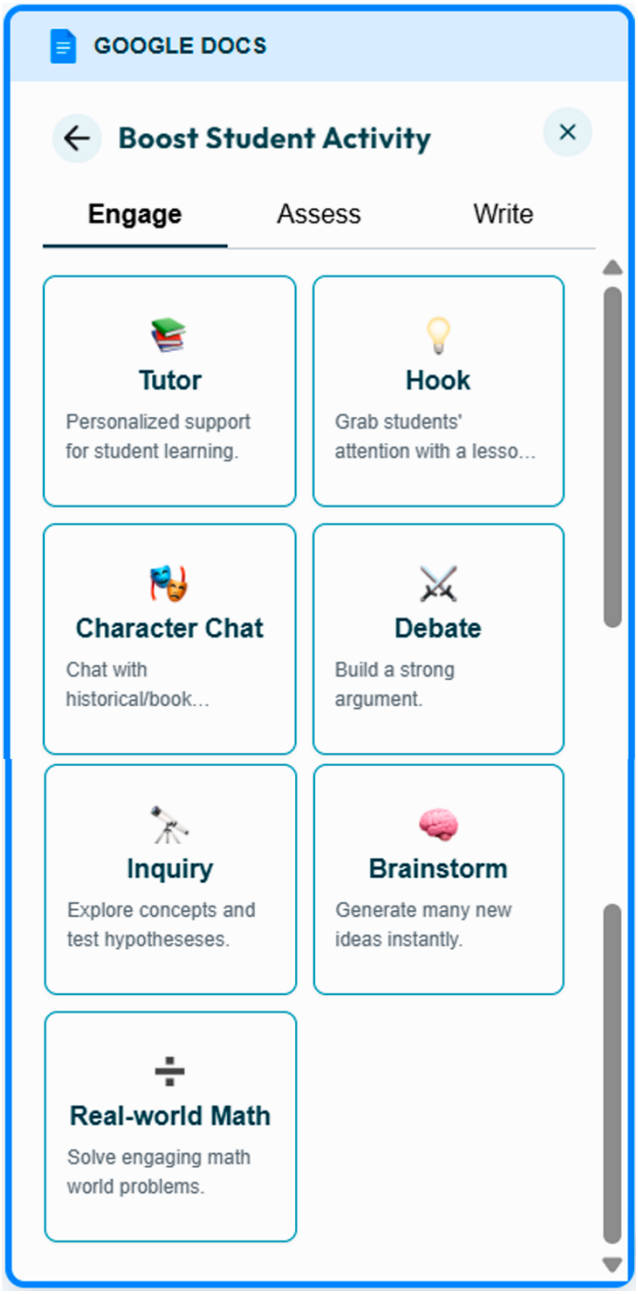

Рис.7 Вкладка Boost Student Activity.

Список используемых интернет-ресурсов.

1. https://www.briskteaching.com/?utm_source=chatgpt.com
2. http://didaktor.ru/brisk-teaching-unikalnaya-nejroset-budushhego/
3. https://www.youtube.com/watch?v=xXMl98UMfuM&t=5s
4. https://www.briskteaching.com/ru/how-brisk-integrates?utm_source=perplexity
5. https://www.ebool.com/alternatives/brisk-teaching?utm_source=perplexity
6. https://habr.com/ru/news/894644/
7. https://method.gsom.spbu.ru/tpost/erxf069uf1-brisk-teaching?utm_source=perplexity

Приложение 1.

Шаблон: Lesson Plan

Методологии разработки программного обеспечения

Цели урока
- Изучить основные методологии разработки программного обеспечения
- Понять особенности различных подходов к разработке ПО
- Развить критическое мышление в выборе методологии для конкретного проекта

Образовательные стандарты
- Профессиональный стандарт "Программист"
- ФГОС ВО по направлению подготовки "Информатика и вычислительная техника"

Структура урока (60 минут)

Введение (10 минут)
- Актуализация знаний о жизненном цикле разработки ПО
- Обсуждение: Почему важен правильный выбор методологии?
- Ключевой вопрос: Как методология влияет на успех проекта?

Основная часть (35 минут)

Теоретический блок (20 минут)
1. Waterfall Model
   - Характеристики
   - Преимущества и ограничения
   - Области применения

2. Agile Model
   - Принципы гибкой разработки
   - Основные подходы (Scrum, Kanban)
   - Преимущества для современных проектов

3. Сравнительный анализ методологий
   - Incremental Model
   - Iterative Model
   - Spiral Model
   - V-Model
   - RAD Model

Практическая работа (15 минут)
- Кейс-стади: Выбор методологии для конкретного проекта
- Групповое упражнение:
  1. Команда получает описание проекта
  2. Анализ требований
  3. Выбор оптимальной методологии
  4. Презентация обоснования

Заключение (15 минут)
- Дискуссия: Плюсы и минусы каждой методологии

- Рефлексия: Что нового узнали?
- Домашнее задание: Подготовить сравнительную таблицу методологий

Методические материалы
- Презентация
- Раздаточные материалы с описанием методологий
- Кейсы для группового анализа

Критерии оценивания
- Понимание теоретического материала
- Качество анализа в групповой работе
- Аргументированность выбора методологии
- Активность участия в дискуссии

Технические средства обучения
- Компьютер
- Проектор
- Доска
- Раздаточные материалы

Рекомендации для преподавателя
- Поощрять критическое мышление
- Стимулировать дискуссию
- Делать акцент на практическом применении знаний
- Адаптировать сложность материала под уровень группы

Дополнительные источники
- Книга "Agile. Разработка программного обеспечения"
- Статьи профессиональных IT-изданий
- Видеокурсы по методологиям разработки ПО

Возможные проблемы и пути их решения

- Сложность восприятия теоретического материала
  Использовать больше практических примеров
- Недостаток времени на обсуждение
  Сократить теоретическую часть
- Низкая активность студентов
  Применять интерактивные методы обучения

Приложение 2.

Шаблон: UDL Lesson Plan

План урока по методологиям разработки программного обеспечения

Тема: Ключевые методологии разработки программного обеспечения

Длительность: 80 минут

Цели урока:
1. Познакомить учащихся с основными методологиями разработки программного обеспечения

2. Развить понимание жизненного цикла разработки ПО
3. Научить сравнивать и выбирать подходящие методологии для разных проектов

Введение (10 минут)

Множественные способы представления:
- Показать короткое видео о разработке ПО (2-3 минуты)
- Представить визуальную схему жизненного цикла ПО на доске

Множественные способы действия и выражения:
- Попросить учеников назвать известные им этапы разработки ПО
- Предложить записать идеи на стикерах и прикрепить к доске

Множественные способы вовлечения:
- Обсудить, почему важно иметь структурированный подход к разработке ПО
- Привести примеры из реальной жизни, где используется ПО

Основная часть (55 минут)

1. Обзор основных методологий (25 минут)

Множественные способы представления:
- Презентация с кратким описанием каждой методологии
- Использование метафор для объяснения (например, Waterfall как строительство дома)

Множественные способы действия и выражения:
- Работа в малых группах: каждая группа изучает одну методологию и представляет ее классу
- Создание интеллект-карт по каждой методологии

Множественные способы вовлечения:
- Обсуждение преимуществ и недостатков каждой методологии
- Предложить ученикам подумать, какую методологию они бы выбрали для своего проекта

2. Сравнение методологий (15 минут)

Множественные способы представления:
- Использование сравнительной таблицы на доске
- Демонстрация графиков, показывающих различия в подходах

Множественные способы действия и выражения:
- Ролевая игра: ученики представляют разные роли в команде разработчиков
- Дебаты: аргументы за и против разных методологий

Множественные способы вовлечения:
- Анализ кейсов: предоставить примеры проектов и обсудить, какая методология подойдет лучше
- Мозговой штурм: как можно улучшить существующие методологии

3. Практическое задание (15 минут)

Множественные способы представления:

- Предоставить письменное описание проекта
- Показать видео-пример похожего проекта

Множественные способы действия и выражения:
- Работа в парах: выбрать методологию для заданного проекта и обосновать выбор
- Создание мини-презентации о выбранной методологии

Множественные способы вовлечения:
- Предложить ученикам придумать свой собственный проект и выбрать для него методологию
- Обсудить, как выбор методологии может повлиять на успех проекта

Заключение (15 минут)

Множественные способы представления:
- Краткое повторение ключевых моментов урока
- Визуальное представление связей между разными методологиями

Множественные способы действия и выражения:
- Мини-тест с выбором ответов по изученному материалу
- Предложить ученикам задать вопросы по теме

Множественные способы вовлечения:
- Обсуждение, как полученные знания могут пригодиться в будущем
- Предложить темы для дальнейшего самостоятельного изучения

Домашнее задание:
Выбрать одну методологию и подготовить презентацию о ее применении в реальном проекте.

Использование вспомогательных технологий:
- Предоставить материалы урока в электронном виде для учеников с особыми потребностями
- Использовать программы для чтения с экрана для учеников с нарушениями зрения
- Разрешить использование планшетов или ноутбуков для заметок и выполнения заданий

Адаптации:
- Предоставить упрощенные определения для сложных терминов
- Разрешить ученикам с трудностями письма записывать ответы на диктофон
- Предложить дополнительное время для выполнения заданий ученикам, которым это необходимо

Приложение 3.

Шаблон: Rubric

Рубрика оценки ресурса по методологиям разработки программного обеспечения

Критерий	**4 балла (отлично)**	**3 балла (хорошо)**	**2 балла (удовлетворительно)**	**1 балл (неудовлетворительно)**
Полнота раскрытия методологий	Глубокое и исчерпывающее описание всех основных методологий разработки ПО с подробным анализом их особенностей и различий [1][2]	Описание большинства методологий с умеренным уровнем детализации	Поверхностное описание 2-3 методологий без глубокого анализа	Минимальное или неточное описание методологий
Практическая применимость	Четкие рекомендации по выбору методологии для различных типов проектов с конкретными примерами и кейсами [3][5]	Общие рекомендации по применению методологий с частичными примерами	Общие утверждения без конкретных практических рекомендаций	Отсутствие практических рекомендаций по применению методологий
Структура и логика изложения	Идеально структурированный материал с логической последовательностью, clear flow между разделами, использование визуальных элементов [4]	Хорошая структура с незначительными нарушениями логики изложения	Частично структурированный материал с логическими разрывами	Хаотичное и бессистемное изложение материала
Актуальность и современность	Включение новейших тенденций в методологиях разработки ПО, ссылки на современные источники и практики [6]	Умеренное освещение современных подходов с некоторыми устаревшими элементами	Минимальное отражение современных тенденций	Устаревшая информация, не отражающая текущее состояние методологий

Приложение 4.

Шаблон: Resource

Методологии разработки программного обеспечения

Введение

Разработка программного обеспечения (ПО) - это сложный процесс, требующий структурированного подхода. В этом ресурсе мы рассмотрим основные методологии разработки ПО, их особенности и применение.

Основные методологии разработки ПО

1. Водопадная модель (Waterfall)

Водопадная модель - это классический последовательный подход к разработке ПО [1].

Особенности:
- Строгая последовательность этапов: сбор требований, проектирование, реализация, тестирование, интеграция, поддержка
- Подходит для проектов с четкими и стабильными требованиями
- Изменения на поздних этапах затруднены

Пример: Разработка системы управления банковскими счетами, где требования строго регламентированы и редко меняются.

2. Гибкая разработка (Agile)

Agile - это семейство методологий, основанных на итеративном и инкрементальном подходе [1][3].

Особенности:
- Быстрая адаптация к изменениям
- Тесное сотрудничество с заказчиком
- Приоритет работающего продукта над документацией

Примеры Agile-методологий:
- Scrum
- Kanban
- Экстремальное программирование (XP)

Пример применения: Разработка мобильного приложения, где требования пользователей могут часто меняться.

3. Инкрементная модель

Инкрементная модель предполагает разделение проекта на части (инкременты) [3].

Особенности:
- Каждый инкремент добавляет новую функциональность
- Позволяет быстро выпускать прототипы
- Снижает риски проекта

Пример: Разработка веб-сайта, где сначала создается базовая функциональность, а затем постепенно добавляются новые разделы и возможности.

4. Итеративная модель

Итеративная модель основана на повторяющихся циклах разработки [2][4].

Особенности:
- Постепенное улучшение продукта
- Возможность учета обратной связи после каждой итерации
- Гибкость в отношении изменений требований

Пример: Разработка игрового движка, где каждая итерация улучшает производительность и добавляет новые функции.

5. V-образная модель

V-образная модель делает акцент на тестировании на каждом этапе разработки [2][4].

Особенности:
- Тесная связь между этапами разработки и тестирования
- Повышенное внимание к качеству продукта
- Подходит для проектов с высокими требованиями к надежности

Пример: Разработка программного обеспечения для медицинского оборудования, где критически важна безошибочность работы.

Выбор методологии

Выбор методологии зависит от многих факторов [1][3][4]:

- Характер проекта и его требования
- Степень определенности и стабильности требований
- Необходимость в гибкости и быстрой обратной связи
- Уровень рисков проекта
- Требования к качеству и тестированию

Заключение

Понимание различных методологий разработки ПО позволяет выбрать наиболее подходящий подход для конкретного проекта. Это способствует более эффективной работе команды разработчиков и повышает шансы на успешное завершение проекта.

Sources
[1] https://wezom.com.ua/blog/metodologija-razrabotki-programmnogo-obespechenija?
[2] https://sky.pro/media/kakie-sushhestvuyut-metodologii-razrabotki-programmnogo-obespecheniya/
[3] https://gb.ru/blog/modeli-i-metodologii-razrabotki-po/
[4] https://wezom.com.ua/blog/metodologija-razrabotki-programmnogo-obespechenija
[5] https://leantech.ai/metodologii-razrabotki-programmnogo-obespecheniya

Приложение 5.

Шаблон: Quiz

Тест по методологиям разработки программного обеспечения

1. Какая методология разработки ПО наиболее эффективна при работе над проектом с постоянно меняющимися требованиями?
A) Waterfall Model
B) V-Model
C) Agile Model
D) RAD Model

2. Что является ключевым преимуществом итеративной модели разработки?
A) Полная документация проекта
B) Возможность улучшения продукта на основе обратной связи
C) Жесткое следование изначальному плану
D) Минимизация затрат на начальном этапе

3. Почему спиральная модель (Spiral Model) особенно подходит для сложных проектов?
A) Она дешевле других методологий
B) Фокусируется на постоянном анализе и управлении рисками
C) Требует минимального участия заказчика
D) Гарантирует моментальный успех проекта

4. Какой этап разработки ПО считается критическим для понимания потребностей будущего продукта?
A) Разработка и программирование
B) Тестирование
C) Анализ требований
D) Документация

5. Что отличает гибкую методологию Agile от классических подходов к разработке?
A) Полное отсутствие планирования
B) Жесткое следование первоначальному техническому заданию
C) Разделение проекта на короткие циклы с постоянной коммуникацией
D) Минимальное взаимодействие между членами команды

Ключ ответов:
1. C
2. B
3. B
4. C
5. C

Приложение 6.

Контрольная работа: Методологии разработки программного обеспечения

Инструкция: Внимательно прочитайте каждый вопрос и дайте развернутый ответ. Постарайтесь продемонстрировать глубокое понимание материала.

1. Сравните методологии Waterfall и Agile. Какие принципиальные различия существуют между ними, и в каких проектных ситуациях каждая из них будет наиболее эффективна?

Ваш ответ:

_____

_____

2. Объясните, почему анализ рисков является критически важным этапом при выборе методологии разработки программного обеспечения. Приведите конкретный пример, как риски могут повлиять на выбор модели разработки.

Ваш ответ:

_____

_____

3. Как этап проектирования влияет на качество и успешность конечного программного продукта? Раскройте взаимосвязь между проектированием и последующими стадиями разработки.

Ваш ответ:

_____

_____

4. Почему в современной разработке ПО так важна гибкость методологий? Опишите преимущества и потенциальные недостатки итеративных подходов.

Ваш ответ:

_____

_____

5. Критически оцените утверждение: "Документация — второстепенный элемент в процессе разработки программного обеспечения". Аргументируйте свою позицию, опираясь на материалы текста.

Ваш ответ:

_____

_____

Критерии оценивания:
- Полнота ответа
- Логичность аргументации
- Понимание материала
- Использование специальной терминологии
- Самостоятельность мышления

Приложение 7.

Шаблон: DOK Questions

Вопросы по глубине знаний о методологиях разработки программного обеспечения

DOK 1 Воспроизведение и recall
1. Перечислите основные методологии разработки ПО (Waterfall, Agile, Scrum, Kanban, Extreme Programming, RAD, Spiral).
2. Назовите классические стадии разработки ПО: анализ требований, проектирование, разработка, тестирование, внедрение, поддержка.

3. Дайте определение методологии разработки ПО как систематического подхода к созданию программных продуктов.
4. Объясните разницу между моделью каскадной разработки и итеративной моделью.
5. Перечислите основные роли в команде разработчиков ПО.

💡 Активность: Создайте интерактивный глоссарий технических терминов с визуальными карточками

DOK 2 Навыки и концепции
1. Сравните методологии Waterfall и Agile по следующим критериям:
   - Гибкость изменений
   - Скорость разработки
   - Взаимодействие с заказчиком
2. Объясните, как специфика проекта влияет на выбор методологии разработки.
3. Проанализируйте преимущества и недостатки гибких методологий разработки.
4. Опишите процесс планирования спринта в методологии Scrum.
5. Раскройте принципы методологии Kanban в управлении разработкой.

💡 Активность: Создайте сравнительную матрицу методологий разработки ПО

DOK 3 Стратегическое мышление
1. Проанализируйте влияние выбора методологии на:
   - Качество конечного продукта
   - Эффективность команды разработчиков
   - Удовлетворенность заказчика
2. Разработайте стратегию минимизации рисков при выборе методологии разработки.
3. Оцените экономическую эффективность различных методологий для разных типов проектов.
4. Исследуйте взаимосвязь между корпоративной культурой и методологией разработки.
5. Предложите механизмы адаптации методологии под конкретные бизнес-требования.

💡 Активность: Смоделируйте кейс выбора оптимальной методологии для стартап-проекта

DOK 4 Критическое мышление
1. Проследите эволюцию методологий разработки ПО от классических моделей до современных подходов.
2. Спрогнозируйте влияние искусственного интеллекта на будущее методологий разработки.
3. Разработайте концепцию новой гибридной методологии, учитывающей:
   - Удаленную работу
   - Постоянно меняющиеся технологические ландшафты
   - Необходимость быстрой адаптации
4. Исследуйте потенциал интеграции методологий разработки с облачными технологиями.
5. Создайте теоретическую модель универсальной методологии разработки ПО.

💡 Активность: Подготовьте футуристическую презентацию о трансформации методологий разработки ПО в следующие 10 лет

Приложение 8.

Шаблон: Science Lab

Лабораторная работа: Методологии разработки программного обеспечения
Цель
Изучить и сравнить различные методологии разработки программного обеспечения, используемые в современной IT-индустрии.
Введение
Методологии разработки программного обеспечения играют ключевую роль в создании качественных цифровых продуктов. В этой лабораторной работе мы рассмотрим несколько основных методологий и проанализируем их особенности.
Материалы
- Компьютер с доступом в интернет
- Бумага и ручка для записей
Процедура
Часть 1: Изучение методологий

1. Прочитайте описание следующих методологий разработки ПО:
   - Waterfall Model
   - Incremental Model
   - Iterative Model
   - Agile Model
   - Spiral Model
   - V-Model
   - RAD Model

2. Для каждой методологии запишите:
   - Краткое описание
   - Основные этапы
   - Преимущества
   - Недостатки

Часть 2: Сравнительный анализ

Заполните таблицу, оценивая каждую методологию по следующим критериям:

Методология	**Гибкость**	**Скорость разработки**	**Управление рисками**	**Вовлечение клиента**
Waterfall				
Incremental				
Iterative				
Agile				

Spiral				
V-Model				
RAD				

Используйте оценки от 1 до 5, где 1 - низкий уровень, 5 - высокий уровень.
Часть 3: Практическое применение

1. Придумайте идею простого программного продукта (например, приложение для составления списка покупок).

2. Выберите две методологии из изученных.

3. Опишите, как бы вы применили каждую из этих методологий для разработки вашего продукта. Укажите:
- Основные этапы разработки
- Примерные сроки каждого этапа
- Потенциальные преимущества и недостатки выбранной методологии для вашего проекта

Вопросы для обсуждения

1. Какая методология, на ваш взгляд, наиболее универсальна? Почему?
2. В каких случаях Waterfall модель может быть предпочтительнее Agile?
3. Как выбор методологии может повлиять на качество конечного продукта?
4. Почему важно понимать различные методологии разработки ПО?

Заключение

Подведите итоги вашего исследования. Какие выводы вы сделали о различных методологиях разработки ПО? Как полученные знания могут пригодиться вам в будущем?

Приложение 9.

Шаблон: Inquiry Worksheet

Рабочий лист: Методологии разработки программного обеспечения
Часть 1: Основные понятия

1. Дайте определение термину "методология разработки программного обеспечения".

2. Перечислите три основные задачи, которые решает выбор методологии разработки ПО.

3. Назовите семь базовых методологий разработки ПО, упомянутых в тексте.
Часть 2: Сравнение методологий

4. Заполните таблицу, используя информацию из текста:

Методология	Для чего подходит	Основная особенность
Waterfall		
Agile		
Spiral		

Часть 3: Этапы разработки

5. Расположите следующие этапы разработки ПО в правильном порядке:
- Тестирование
- Анализ требований
- Внедрение и сопровождение
- Проектирование
- Разработка и программирование
- Документация

6. Выберите один из этапов разработки и опишите его более подробно (3-4 предложения).

Часть 4: Критическое мышление

7. Почему важно выбирать подходящую методологию разработки ПО для каждого конкретного проекта? Приведите два аргумента.

8. Представьте, что вы разрабатываете мобильное приложение для заказа еды. Какую методологию разработки вы бы выбрали и почему?

Часть 5: Практическое применение

9. Создайте краткий план разработки простого веб-сайта, используя одну из методологий, описанных в тексте. Укажите основные этапы и их особенности.

10. Как вы думаете, какие навыки необходимы для успешной работы в команде разработчиков ПО? Назовите три ключевых навыка и объясните их важность.

Приложение 10.

Шаблон: Guided Notes

Методологии разработки программного обеспечения

Введение

Разработка качественного продукта начинается с определения его _____ цикла. Это четкий план действий, позволяющий понять:

- Что должно получиться у разработчиков
- Как достичь _____
- Какие _____ для этого использовать

Что такое модель разработки продукта?

Каждый проект имеет собственный _____ цикл, который состоит из следующих этапов:

1. Создание _____ и принятие решений
2. Подготовка и _____
3. Поиск концепции и пути создания _____

Выбор модели разработки ПО позволяет:

- Определить порядок выполнения и _____ задач
- Разработать систему _____ и оценки разработки
- Сформировать _____ создания продукта
- Определить _____

Основные методологии разработки ПО

1. _____ Model (Водопадная модель)
   - Подходит для: _____

2. _____ Model (Инкрементная модель)
   - Подходит для: _____

3. _____ Model (Итеративная модель)
   - Подходит для: _____

4. _____ Model (Гибкая методология)
   - Подходит для: _____

5. _____ Model (Спиральная модель)
   - Подходит для: _____

6. _____ Model (V-образная модель)
   - Подходит для: _____

7. _____ Model (Модель быстрой разработки приложений)
   - Подходит для: _____

Стадии разработки ПО

1. _____ требований
2. _____
3. Разработка и _____
4. _____
5. _____
6. Внедрение и _____

Заключение

Выбор методологии разработки ПО зависит от:

- Размера и _____ проекта
- _____ и четкости требований к ПО
- Структуры и квалификации _____
- Доступности _____, технологий и ресурсов

Правильный выбор методологии помогает создать _____, функциональный, надежный и удобный продукт.