Контрольно-оценочные средства для проведения текущего контроля
по УП.5 Учебной практики
(2 курс, 4 семестр 2026-2027 уч. г.)

Текущий контроль №1 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)

Описательная часть: Практическая работа с применением ИКТ

Задание №1 (20 минут)

Анализ и формализация неявных требований заказчика — от "слова в слово" к функциональной спецификации:

1. Выделить явные и скрытые требования (не менее 5 шт.).
2. Классифицировать их:
    - функциональные / нефункциональные
    - обязательные / желательные
3. Сформулировать 2–3 SMART-требования (конкретных, измеримых, достижимых, релевантных, ограниченных по времени).

4. Оформить результат в виде таблицы в Markdown (GitLab Wiki или .md-файл в репозитории).

Оценка	Показатели оценки
3	Выделено ≥3 требования. Классификация частично верна. SMART-требования сформулированы, но не все параметры учтены (например, нет измеримости). Таблица оформлена, но с ошибками форматирования.
4	Выделено ≥5 требований, указаны явные и скрытые. Классификация корректна в 80%. SMART-требования соответствуют ≥4 из 5 критериев. Таблица структурирована, форматирование в Markdown верное.
5	Выделено ≥6 требований (в том числе скрытые: безопасность, масштабируемость, UX). Классификация точна. Все SMART-требования корректны. В таблице — пояснения к выбору. Работа оформлена как часть GitLab-документации (например, docs/requirements_analysis.md).

Задание №2 (25 минут)

Задание на имитацию интервью + анкетирование — сравнительный анализ:

1. На основе интервью — выявить 3 ключевые потребности и 1 риск недопонимания.
2. На основе анкеты — найти 1 статистически значимый тренд (например: «80% хотят уведомления в Telegram»).
3. Сравнить:

- Какой метод дал более точные, а какой — более полные данные?

- Какие требования совпали, какие — противоречат?

4. Сделать вывод: какой метод выбрать на старте проекта и почему?

Оценка	Показатели оценки
3	Выявлены потребности из обоих источников. Указан 1 тренд и 1 риск. Вывод общий («интервью лучше»). Отчет есть, но без аргументов.
4	Потребности сформулированы как требования. Тренд подтвержден цифрами. Риск связан с неоднозначностью формулировок. В выводе — сравнение по 2 критериям (полнота/точность/скорость). Отчет структурирован.
5	Потребности → функциональные требования (например: «Система должна отправлять уведомления в Telegram при изменении статуса задачи»). Тренд + контекст («8 из 10 — преподаватели старше 40 лет → важна простота интерфейса»). Указан метод компенсации недостатков («интервью + фокус-группа»). Отчет оформлен как Issue с label research.

Текущий контроль №2 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)

Описательная часть: Практическая работа с применением ИКТ

Задание №1 (20 минут) Создание скетчей экранов и навигации.

Условие:

Бизнес-сценарий: «Преподаватель заходит в систему, создает задачу для группы, назначает срок, прикрепляет файл, и видит список исполнителей».

Ограничения:

Всего не более 4 экранов (включая авторизацию),

Нет доступа к интернету → Figma в offline-режиме (или бумага + карандаш при необходимости),

Использовать шаблон карточек экранов (предоставлен заранее — 4 прямоугольника на А4 в .docx).

Задача:

Спроектировать логику переходов между экранами (wireflow) — указать стрелки, подписи действий.

Для каждого экрана определить:

- основной пользовательский intent (цель),
- ключевые элементы интерфейса (макс. 5 шт.),
- тип макета (одноколоночный / сетка / карточки).

Обосновать одно архитектурное решение (например: «Форма создания задачи — отдельный экран, а не модальное окно, потому что много полей»).

Оценка	Показатели оценки

3	Схема содержит ≥3 экрана, переходы обозначены. Элементы интерфейса перечислены, но без привязки к цели. Обоснование — общее («так удобнее»).
4	4 экрана, все переходы логичны, без «мертвых» путей. Для каждого экрана — указан intent и ≥4 элемента. Выбран тип макета. Обоснование — по одному принципу UX (напр., «снижение когнитивной нагрузки»).
5	Использован термин wireflow. Учтены крайние случаи (ошибка загрузки, пустой список). Элементы сгруппированы по зонам внимания (F-паттерн / Z-паттерн). Обоснование ссылается на принципы: целостность, минимализм, предсказуемость. Работа оформлена в Figma с использованием компонентов (Button, Input — даже если stub).

Задание №2 (25 минут)

В данном задании создать макет одного экрана — «Создание задачи».

Требования к макету:

Использовать только базовые фигуры: прямоугольники, линии, placeholder-текст ([название задачи], [срок]),
Соблюсти:

- Визуальную иерархию (заголовок > поля > кнопки),
- Согласованность (единые отступы, размеры кнопок),
- Доступность (контраст ≥ 4.5:1 — можно проверить по серым тонам),
- Ориентацию на действие (основная кнопка — “Создать задачу” — выделена).

Задача:

Создать макет в Figma (1 frame, 360x640 px — mobile-first).
Под макетом добавить 3 подписи-пояснения (в текстовом поле):

- Какой принцип UI/UX реализован в заголовке?
- Почему выбран именно такой порядок полей?
- Как обеспечена обратная связь при ошибке?

Оценка	Показатели оценки
3	Макет содержит все обязательные элементы. Подписи есть, но без терминов UI/UX. Иерархия слабо выражена (все одного размера).
4	Четкая визуальная иерархия (заголовок 20 px, поля 14 px, кнопка 16 px). Подписи содержат ≥1 термин («зона внимания», «прогрессивное раскрытие», «аффорданс»). Отступы единообразны (кратны 8 px).
5	Использованы: Грид, Placeholder-иконки, Состояния. Подписи ссылаются на конкретные принципы. Макет — в компонентах (даже если 1 экземпляр).

Текущий контроль №3 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)
Описательная часть: Практическая работа с применением ИКТ
Задание: Структура проектной документации по ГОСТ.

Задание №1 (15 минут)

Разделить элементы на 3 группы по фазам жизненного цикла (по ГОСТ 34.601–90):

- Предпроектная стадия
- Стадия разработки
- Ввод в действие

Выбрать 2 элемента, обязательных для проектной документации (не эксплуатационной).
Вставить выбранные в шаблон project_docs_structure.docx в раздел «Обязательные компоненты».

Оценка	Показатели оценки
3	Верно выделены ≥2 фазы, ≥4 элемента распределены правильно. Выбраны 2 элемента, но без обоснования принадлежности к проектной (а не эксплуатационной) документации.
4	Все 3 фазы названы верно. ≥6 элементов распределены корректно. Выбраны ТЭО и Требования к ИБ (или Эскизный проект) с пояснением: «предъявляются до начала разработки».
5	Все элементы распределены точно. В пояснении — ссылка на ГОСТ: «по п.5.2 ГОСТ 34.601–90, проектная документация включает ТЭО и техническое задание». Шаблон оформлен с использованием стилей Word («Заголовок 1», «Список»).

Задание №2 (15 минут)

Выделить 3 функции системы (например: «Назначение задания группе»).
Для каждой функции заполнить по шаблону:

- Наименование
- Входные данные
- Выходные данные
- Реакция системы

Использовать только термины из ГОСТ («пользователь», «система», «должна обеспечивать»).

Оценка	Показатели оценки
3	Есть 3 функции. Заполнены ≥2 поля на функцию. Использованы разговорные формулировки («чтобы было видно»).
4	Все поля заполнены. Использована формальная лексика. Есть ≥1 корректная формулировка по ГОСТ (напр., «Система должна обеспечивать формирование отчета о статусе выполнения»).
5	Все формулировки соответствуют стилю ГОСТ. Учтены: ошибочные ситуации («при отсутствии назначенной группы — вывод сообщения об ошибке»), полнота (вход/выход/реакция). Документ сохранен с метаданными (автор, дата) — свойства файла Word.

Задание №3 (15 минут)

Разделить требования по трем категориям:

- Функциональные (F)
- Нефункциональные (NF)
- Out of Scope (OOS) — «хотелки», не входящие в MVP

Для каждого NF указать тип: производительность, безопасность, юзабилити, переносимость.
Выбрать 1 F-требование и переписать его по SMART.

Оценка	Показатели оценки
3	Верно классифицировано ≥4 требования. Типы NF указаны частично. SMART — без одного параметра.
4	Все требования классифицированы верно, кроме OOS (№5 — «чат» — может быть спорным). NF типы — 3/4 верны. SMART — 4/5 параметров.
5	Четкое обоснование OOS: «чат — расширение функционала, не требуется в ТЗ». Все NF типы верны. SMART-требование — полное: «Система должна позволять преподавателю прикреплять до 5 файлов (≤10 МБ каждый) к заданию в течение 5 секунд после нажатия кнопки „Прикрепить“». Работа оформлена как Markdown-таблица.

Текущий контроль №4 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)
Описательная часть: Практическая работа с применением ИКТ
Задание №1 (45 минут)

Анализ существующего предложение оптимизации по критериям качества:
Дано:

Описание реальной (упрощенной) ИС — «Внутренний портал техникума»:

- Архитектура: PHP (устаревший код), MySQL, без кэширования
- Проблемы (из отчета пользователей):

1. Загрузка списка заданий — 8–12 сек
2. После обновления страницы — данные теряются
3. Нет фильтров — при 200+ заданиях — прокрутка вручную
4. Один и тот же SQL-запрос повторяется 5 раз на странице

Задача:

Классифицировать каждую проблему по характеристикам качества ПО (ГОСТ 28806-90):

- производительность, надежность, удобство использования, сопровождаемость и т.д.

Предложить по одному решению на проблему (техническое или архитектурное).
Выбрать одно решение и обосновать его по критериям:

- трудозатраты (низкие/средние/высокие)
- эффект (низкий/средний/высокий)

→ оформить как матрицу «затраты–эффект».

Оценка	Показатели оценки
3	Верно сопоставлены ≥2 проблемы с характеристиками качества. Предложены ≥2 решения. Матрица есть, но без обоснования.
4	Все 4 проблемы классифицированы верно. Решения конкретны. В матрице — адекватная оценка «затраты/эффект» для 1 решения.
5	Использованы термины: «время отклика», «состояние сессии», «дублирование логики». Решения включают: техническое (кеширование), архитектурное (разделение на модули), UX (добавление пагинации). В матрице — обоснование: «Пагинация — низкие затраты (фронтенд), высокий эффект (снижение нагрузки, улучшение UX)». Работа оформлена как Markdown-таблица.

Текущий контроль №5 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)

Описательная часть: Практическая работа с применением ИКТ

Задание №1 (20 минут) – 1 глубинную потребность (неочевидную, скрытую).
Выделить из анкеты — 1 статистически значимый тренд (≥70% ответов).
Сравнить методы по трем параметрам:

- глубина данных,
- скорость сбора,
- репрезентативность.

Сделать вывод: какой метод выбрать на этапе сбора требований — и почему.

Оценка	Показатели оценки
3	Выделены потребность и тренд. Сравнение по ≥1 параметру. Вывод есть, но общий.
4	Потребность — неочевидная («не просто “назначать задачи”, а “контролировать выполнение без личного общения”»). Тренд — с цифрами («8/10 — хотят уведомления в Telegram»). Сравнение по 2 параметрам. Вывод аргументирован.
5	Указаны ограничения методов («интервью — субъективно», «анкета — поверхностно»). Предложен комбинированный подход («анкета → интервью с отклонившимися»). Вывод ссылается на ОК2 (анализ информации) и ПК1 (сбор исходных данных). Работа оформлена как Markdown-таблица.

Задание №2 (25 минут) – х действие.
Описать последовательность шагов при входе пользователя (без кода!):
от ввода логина/пароля → до загрузки главной страницы с учетом роли.

Указать 2 уязвимости (например: хранение пароля в открытом виде, отсутствие лимита попыток).

Оценка	Показатели оценки
3	Таблица заполнена частично (≥4 ячейки). Последовательность — ≥2 шага. Уязвимости названы общие («пароль могут украсть»).
4	Таблица полная, с учетом «только свои» (например: «студент → редактировать — только свои»). Последовательность — 4+ шагов (ввод → хеширование → проверка → выдача токена → редирект). Уязвимости — конкретные («передача пароля в теле POST без HTTPS»).
5	В таблице — обоснование («адмн → редактировать чужие: требуется для модерации»). В последовательности — термины: хеширование (bcrypt), JWT-токен, RBAC. Уязвимости дополнены мерами защиты («лимит попыток + капча»). Работа соответствует ПК3 («разрабатывать подсистемы безопасности по ТЗ»).

Текущий контроль №6 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)
Описательная часть: Практическая работа с применением ИКТ
Задание №1 (45 минут)

Выполнить командной разработки в Git:

1. Создать ветку feature/…, внести изменения в два файла.
2. В main сделать параллельное изменение → вызвать конфликт при слиянии.
3. Разрешить конфликт вручную, сохранить обе правки.
4. Отправить ветку в локальный GitLab и создать Merge Request с описанием.

Оценка	Показатели оценки
3	етка создана, MR есть, но конфликт не разрешен или разрешен некорректно.
4	Конфликт обнаружен и разрешен вручную; MR содержит описание.
5	Все этапы выполнены корректно; в MR — структурированное описание; использованы git checkout -b, merge, push, визуальное редактирование в VS Code.

Текущий контроль №7 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)
Описательная часть: Практическая работа с применением ИКТ
Задание №1 (20 минут)

- добавить проверку isset() и !empty() для title, deadline, group_id,
- сохранять файл в /uploads/ с уникальным именем (uniqid() . '_' . basename()),
- использовать подготовленные выражения (PDO::prepare).

Добавить комментарий: «// Исправлено: валидация, безопасность, структура».

Оценка	Показатели оценки
3	Исправлено ≥1 требование (напр., только валидация). Код работает, но уязвимости остались.
4	Исправлены валидация и безопасность (PDO). Файлы сохраняются в /uploads/. Комментарий есть.
5	Все исправления + обработка ошибок (try/catch), логирование (простой error_log()), соответствие ПК4 и ПК3.

Задание №2 (25 минут)

Вызвать ее после успешного создания задачи в task_create_fixed.php.
Проверить в Postman (offline): имитация запроса → статус 200.

Оценка	Показатели оценки
3	Реализован запрос, но без обработки ошибок. Параметры жестко вписаны (не из config).
4	Используется file_get_contents() или cURL, параметры из config, JSON корректен. Тест в Postman пройден.
5	Обработка ошибок (http_response_code, лог), retry-логика (1 попытка), соответствие ПК4 и ПК3 (безопасная передача токена), вызов после коммита в БД.

Текущий контроль №8 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)
Описательная часть: Практическая работа с применением ИКТ

Задание №1 (15 минут)

Запустить тест через phpunit ValidatorTest.php (или jest для JS-версии — по выбору).
Сделать скриншот успешного прохождения (OK (3 tests)).

Оценка	Показатели оценки
3	Написан 1 тест. Синтаксис частично верен.
4	Оба теста написаны, проходят. Использованы assertTrue/assertFalse.

5	Тесты покрывают граничные случаи; есть @dataProvider (или аналог); скриншот с отчетом.

Задание №2 (15 минут)

Настроить POST-запрос в Postman (метод, URL, заголовок Content-Type: application/json, тело).

2. Выполнить → получить mock-ответ (заглушка 200 OK из файла mock/telegram_200.json).
3. Заполнить чек-лист:

- Да/Нет для каждого пункта,
- краткое пояснение для Нет.

Оценка	Показатели оценки
3	Запрос настроен, 2 пункта чек-листа заполнены.
4	Все 4 пункта, пояснения есть. Указано: «токен не должен логироваться».
5	Чек-лист в Markdown-таблице; учтены: идемпотентность, обработка 429, логирование без токена.

Задание №3 (15 минут)

Задание: Разделы «Типовые операции» (не более 150 слов):

- Как добавить задачу через админ-панель (3 шага),
- Как проверить, что задача отображается у студентов (1 шаг),
- Что делать при ошибке «Не удалось сохранить» (1 действие).

Оценка	Показатели оценки
3	Есть 2 операции, без деталей.
4	Все 3 пункта, четкие шаги («1. Открыть… 2. Ввести… 3. Нажать…»).
5	Использованы скриншоты-заглушки ([рис.1]), ссылки на лог-файлы, предупреждения (Внимание!). Соответствует ГОСТ.

Текущий контроль №9 (45 минут)

Форма контроля: Практическая работа (Информационно-аналитический)

Описательная часть: Практическая работа с применением ИКТ

Задание №1 (9 минут)

Задание: 5 пунктов (в MS Word / Markdown):

- 2 позитивных сценария (корректные данные),
- 2 негативных (пустое поле, прошедшая дата),
- 1 граничный (дата = сегодня 23:59).

Формат:

1. Ввести корректный заголовок и дату → кнопка "Создать" активна
Сдача: testing/functional_checklist.md

Оценка	Показатели оценки
3	Чек-лист содержит ≥3 пункта. Есть хотя бы 1 позитивный и 1 негативный сценарий. Формат частично соблюден (без [ ] или без →).
4	Все 5 пунктов присутствуют. Сценарии соответствуют типам: 2 позитивных, 2 негативных, 1 граничный. Есть четкое описание действия → ожидаемого результата.
5	Чек-лист оформлен по шаблону (Markdown, [ ], стрелка →). Использованы реалистичные данные (длина, формат даты). Граничный случай включает точное время (23:59) и проверку отображения. Работа сохранена в нужной папке (testing/) и названа корректно.

Задание №2 (8 минут)

В файле tests/ValidatorTest.php 2 теста:

- testValidTaskReturnsTrue()
- testPastDeadlineReturnsFalse()

Использовать assertTrue() / assertFalse().
Сдача: tests/ValidatorTest.php

Оценка	Показатели оценки
3	Написан 1 тест, синтаксически корректный (метод объявлен, вызван assertTrue/assertFalse). Тест может не проходить из-за ошибок в данных или логике.
4	Оба теста реализованы и проходят. Использованы правильные ожидаемые значения (true для валидных данных, false — для просроченной даты). Код соответствует PSR/общепринятому стилю (отступы, имена).
5	Все тесты реализованы и успешно выполняются. Данные заданы явно, есть комментарий и файл сохранен в правильной структуре проекта.

Задание №3 (9 минут)

Создать коллекцию в Postman (offline):

- 1 запрос: POST /api/create-task с телом JSON,
- 1 тест вкладке Tests.

Сдача: testing/api_test_collection.json

Оценка	Показатели оценки
3	Collection создан, есть 1 запрос. Тело запроса частично заполнено. Во вкладке Tests есть код, но он не проходит (ошибка в синтаксисе pm.test или pm.response). Файл сохранен, но не в нужной папке.

4	Collection содержит 1 корректный запрос: метод POST, URL /api/create-task, заголовок Content-Type: application/json, валидное тело (напр., {"title":"Тест","deadline":"2025-12-10"}). Во вкладке Tests — 1 рабочий тест. Файл сохранен как api_test_collection.json в папке testing/.
5	Запрос полностью соответствует заданию.

Задание №4 (9 минут)

Заполните шаблон docs/test_report_template.docx (раздел «2. Результаты модульного тестирования»):

- Название модуля,
- Количество тестов / пройдено / упали,
- Краткое описание найденного бага (можно вымышленного, напр. «не проверяется длина заголовка»).

Сдача: docs/test_report_v1.docx

Оценка	Показатели оценки
3	Раздел заполнен частично: есть название модуля и ≥1 числовое значение. Описание бага есть, но без связи с ТЗ. Оформление не по шаблону (сплошной текст).
4	Все 3 элемента присутствуют. Числа корректны (напр., 5 / 4 / 1). Баг описан конкретно («не проверяется длина заголовка — допускается 0 символов»). Использована таблица или структурированный список.
5	Данные соответствуют реалистичному сценарию (модуль validateTask, тесты 5/4/1). Описание бага включает: что, где, последствие, рекомендация («В функции validateTask() отсутствует проверка на минимальную длину заголовка → возможен ввод пустой строки → некорректное отображение в UI. Рекомендуется добавить if (strlen($title) < 3)»). Оформление — по ГОСТ (таблица, заголовки, нумерация).

Задание №5 (9 минут)

Заполните файл docs/tools_review.docx в MS Word / Markdown, в которой отразить тип тестирования, инструмент и краткое обоснование.

Оценка	Показатели оценки
3	Заполнены ≥2 строки. Инструменты частично соответствуют типу. Обоснования общие («удобный»).
4	Все 3 строки заполнены верно. Обоснования точные («Postman — для проверки эндпоинтов без UI»). Таблица структурирована.
5	Верный выбор + учтены ограничения («PHPUnit — offline, не требует браузера»), или расширение: добавлен 4-й тип. Обоснования ссылаются на ОК2 (анализ инструментов) и ПК5. Оформление — по шаблону.