Министерство образования Иркутской области
Государственное бюджетное профессиональное
образовательное учреждение Иркутской области
«Иркутский авиационный техникум»

Методические указания по выполнению самостоятельной работы по междисциплинарному курсу

МДК.02.02 Осуществление интеграции программных модулей

специальности
09.02.11 Разработка и управление программным обеспечением

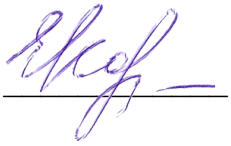
Иркутск, 2026

РАССМОТРЕНЫ

Председатель ЦК

_____ / /

УТВЕРЖДАЮ
Зам. директора


_____ Е.А. Коробкова

№	Разработчик ФИО
1	Степанов Станислав Евгеньевич

Пояснительная записка

МДК.02.02 Осуществление интеграции программных модулей относится к ПМ.02 разработка и интеграция модулей программного обеспечения. Самостоятельная работа является одним из видов учебно работы обучающегося без взаимодействия с преподавателем.

Основные цели самостоятельной работы:

- систематизация и закрепление теоретических знаний и практических умений обучающихся;
- углубление и расширение теоретических знаний, формирование умений использовать справочную документацию и дополнительную литературу;
- развитие познавательных способностей и активности обучающихся, творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельного мышления;
- развитие исследовательских умений.

Рекомендации для обучающихся по выработке навыков самостоятельной работы:

Внимательно читать план выполнения работы. Учиться кратко и емко излагать свои мысли. Обращать внимание на рекомендуемые источники.

Тематический план

Раздел Тема	Тема занятия	Название работы	Количество часов
Раздел 6. Комплексное проектирование. Тема 8. Применять кэширование интеграционных запросов.	Кэширование: Redis для интеграционных запросов.	Кэширование: Redis для интеграционных запросов.	2

Самостоятельная работа №1

Название работы: Кэширование: Redis для интеграционных запросов..
Цель работы: Систематизация знаний и умений.
Уровень СРС: воспроизводящая.
Форма контроля: Практическая работа с применением ИКТ.
Количество часов на выполнение: 2 часа.
Задание:
Разработайте небольшой программный модуль, который получает данные из внешнего API и кэширует результат запроса в Redis.
В качестве внешнего сервиса можно использовать открытый API, например:

- получение данных о пользователях;
- получение списка товаров;
- получение публикаций;
- получение информации о погоде;
- получение курса валют.

Требования к выполнению

1. Создайте запрос к выбранному внешнему API.
2. Подключите Redis к разрабатываемому приложению.
3. Перед отправкой запроса к внешнему API выполните проверку наличия данных в Redis.
4. Если данные присутствуют в кэше:
    - получите их из Redis;
    - не выполняйте повторный запрос к внешнему API;
    - выведите сообщение: «Данные получены из кэша».
5. Если данные в кэше отсутствуют:
    - выполните запрос к внешнему API;
    - сохраните полученный результат в Redis;
    - выведите сообщение: «Данные получены из внешнего API».
6. Для кэшируемых данных задайте время жизни — 60 секунд.
7. После истечения времени жизни повторный запрос должен снова обращаться к внешнему API и сохранять обновлённые данные в Redis.
8. Обработайте возможные ошибки:
    - внешний API недоступен;
    - Redis недоступен;
    - внешний сервис вернул некорректный ответ.
9. Продемонстрируйте не менее трёх последовательных запусков:
    - первый запрос — получение данных из внешнего API;
    - второй запрос — получение данных из Redis;
    - третий запрос после истечения времени жизни — повторное

получение данных из внешнего API.

Рекомендуемая структура программы

Программа должна содержать:

- настройку подключения к Redis;
- функцию выполнения запроса к внешнему API;
- функцию получения данных из кэша;
- функцию сохранения данных в кэш;
- обработку ошибок;
- вывод источника полученных данных;
- вывод времени выполнения запроса.

Результат выполнения

Необходимо представить:

1. Исходный код программы.
2. Скриншот или журнал первого запроса к внешнему API.
3. Скриншот или журнал повторного получения данных из Redis.
4. Скриншот или журнал запроса после истечения времени жизни кэша.
5. Краткий вывод объёмом 3–5 предложений, содержащий:
    - назначение Redis;
    - влияние кэширования на скорость работы приложения;
    - ситуации, в которых кэширование интеграционных запросов наиболее эффективно.

Критерии оценки:

оценка «5» -

- полностью и корректно реализован механизм кэширования интеграционных запросов;
- используются обоснованные и уникальные ключи Redis;
- корректно реализована проверка кэша до обращения к внешнему API;
- время жизни данных составляет 60 секунд и фактически проверено;
- программа демонстрирует получение данных из API, Redis и повторное обновление после истечения TTL;
- обработаны ошибки внешнего API, подключения к Redis и некорректного ответа;
- выводится время выполнения запросов, позволяющее сравнить скорость получения данных;
- код структурирован по функциям или отдельным компонентам;
- отсутствует необоснованное дублирование кода;
- результаты работы оформлены полностью, вывод содержит

обоснованное объяснение преимуществ и ограничений кэширования.

оценка «4» -
- корректно реализован запрос к внешнему API;
- перед выполнением запроса проверяется наличие данных в Redis;
- данные корректно сохраняются и извлекаются из кэша;
- установлено время жизни кэша 60 секунд;
- выводится источник данных: Redis или внешний API;
- продемонстрирована работа до и после истечения времени жизни кэша;
- реализована обработка основных ошибок;
- код имеет понятную структуру и выполняется без существенных ошибок;
- представлен краткий вывод по результатам работы.

оценка «3» -
- реализован запрос к внешнему API;
- выполнено подключение к Redis;
- полученные данные сохраняются в кэш;
- программа может получить данные из кэша при повторном запросе;
- задано время жизни кэша;
- допускаются отдельные ошибки в структуре кода или выводе сообщений;
- обработка ошибок реализована частично;
- работоспособность программы продемонстрирована с помощью преподавателя.