

Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации
(Минцифры РФ)

Федеральное государственное бюджетное образовательное учреждение
высшего образования «Сибирский государственный университет
телекоммуникаций и информатики»
(СибГУТИ)



МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению контрольной работы по дисциплине
«Технологии разработки программного обеспечения»
для студентов направления подготовки
09.03.01 «Информатика и вычислительная техника»

Новосибирск,
2024

УДК 629.113(071):004.01:004.4

Методические указания к выполнению контрольной работы по дисциплине «Технологии разработки программного обеспечения» для студентов направления подготовки 09.03.01 «Информатика и вычислительная техника» / Составитель: доц. каф. ММиЦРБС Полетайкин А.Н., – Новосибирск: СибГУТИ, 2024. – 38 с.

Методические указания содержат задание и методические рекомендации к выполнению контрольной работы по дисциплине «Технологии разработки программного обеспечения», а также требования к содержанию и оформлению отчета к контрольной работе.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
1. ЦЕЛЬ И ЗАДАЧИ ВЫПОЛНЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ.....	5
2. СОДЕРЖАНИЕ И ЭТАПЫ ВЫПОЛНЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ.....	7
3. ТЕМАТИКА ПРОЕКТИРОВАНИЯ	10
4. ТРЕБОВАНИЯ К СТРУКТУРЕ КОНТРОЛЬНОЙ РАБОТЫ	13
5. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА.....	20
6. ПОРЯДОК СДАЧИ НА ПРОВЕРКУ И ЗАЩИТЫ КОНТРОЛЬНОЙ РАБОТЫ	25
СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ	28
ПРИЛОЖЕНИЕ А Образец титульного листа	31
ПРИЛОЖЕНИЕ Б Образец бланка задания на КР	32
ПРИЛОЖЕНИЕ В Пример реализации простейшего web-приложения с микросервисной архитектурой.....	33

ВВЕДЕНИЕ

Контрольная работа (КР) по дисциплине "Технологии разработки программного обеспечения" выполняется студентами направления 09.03.01 «Информатика и вычислительная техника» в процессе изучения одноименной учебной дисциплины и посвящается разработке программной системы (ПС) с микросервисной архитектурой с использованием современных технологий проектирования и разработки ПС.

Для выполнения КР должна быть выбрана индивидуальная тема, связанная с разработкой программной системы. Примерная тематика рассматривается в разделе 3 данных методических указаний. Выбранную тему целесообразно предварительно согласовать с преподавателем.

Контрольная работа оформляется в виде отчета, сдается на проверку в последние две недели семестра и защищается студентом в соответствии с графиком, утвержденным руководителем.

Руководитель КР осуществляет следующие функции:

- согласование темы КР и выдача студенту задания на КР;
- предоставления помощи студенту относительно конкретизации содержания КР и разработки календарного плана её выполнения;
- консультации и научно-методическое руководство работой студента в процессе выполнения КР;
- систематический контроль за ходом выполнения КР;
- предоставления помощи студенту в сборе основного и дополнительного материала для выполнения КР и оформления отчета;
- проверка завершенной КР и отчета;
- подготовка студента к защите КР перед комиссией.

Работа с настоящими методическими указаниями в идеале предполагает последовательное изучение методического материала «от корки до корки», с дальнейшим обращением к соответствующим подразделам раздела 4, а также к разделу 5, в процессе проектирования ПС, сверяясь при этом с графиком в разделе 2.

Для студентов, по каким-либо причинам не имеющих достаточного времени на добросовестное изучение методических указаний, рекомендуется в обязательном порядке ознакомиться с разделами 2, 4 и 6, чтобы правильно оценить масштаб деятельности. Дальнейшая работа над проектом с целью минимизации риска несоответствия работы установленным требованиям должна выверяться по содержанию разделов согласно требованиям к содержанию отчетов о выполнении практических работ по дисциплине.

1. ЦЕЛЬ И ЗАДАЧИ ВЫПОЛНЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ

Контрольная работа по дисциплине "Технологии разработки программного обеспечения" представляет собой завершённую проектную разработку в профессиональной сфере для решения актуальной задачи из области прикладной информатики, в которой:

- сформулирована актуальность и место решаемой задачи информатизации в определенной предметной области;
- анализируется литература и информация, полученная с помощью глобальных сетей, по функционированию подобных систем и технологий в данной области или в смежных предметных областях;
- исследуются подходы, математические методы, математические модели, алгоритмы, программы, технические и инструментальные средства разработки приложений с современной архитектурой применительно к созданию информационного и программного обеспечения ПС;
- определяются и конкретно описываются выбранные студентом современные технологии для решения задачи;
- разрабатывается база данных для информационного обеспечения решения задачи при помощи современной СУБД;
- разрабатывается программное приложение на одном или нескольких языках программирования при помощи современных средств разработки и отладки ПО;
- осуществляется версионный контроль разрабатываемого приложения ПС при помощи современных распределенных средств управления версиями;
- осуществляется тестирование выполненных разработок с применением специальных методов и современных средств тестирования и технологий CI/CD.

Целью выполнения КР по дисциплине "Технологии разработки программного обеспечения" является практическое закрепление теоретической части этого курса, а именно, углубление знаний по программированию и коллективной разработке приложений с применением современных технологий разработки ПО, а также формирование умений и навыков разработки ПО и приложений на его основе.

В результате выполнения контрольной работы студенты должны приобрести такие практические навыки:

- организации программного процесса коллективной разработки ПС;
- выбора современных технологий для коллективной разработки ПС;

- управления требованиями к ПС, постановки задачи на создание ПС с микросервисной архитектурой;
- постановки задачи на создание ПС;
- развертывание рабочей среды для коллективной разработки ПС;
- разработки и тестирования ПС с использованием современных технологий CI/CD;
- сопровождения ПС в условиях реализации версионного контроля;
- оформления технической документации к ПС и программному приложению.

2. СОДЕРЖАНИЕ И ЭТАПЫ ВЫПОЛНЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ

Контрольная работа по дисциплине "Технологии разработки программного обеспечения" предполагает решение конкретной практической задачи информатизации и содержит разработку рабочих проектных решений, связанных с созданием программного приложения с микросервисной архитектурой.

Задание на КР:

В соответствии с согласованной индивидуальной темой реализовать ПС в виде микросервисной архитектуры – логически разделить ПС на выполняемые задачи и для каждой задачи реализовать отдельный контейнер (или группу контейнеров) как законченное приложение. Обмен данными между микросервисами реализовать через WEB-API (запрос/ответ).

В качестве средства контейнеризации использовать Docker.

Рекомендуемая платформа web-разработки: Django.

Рекомендуемая СУБД: PostgreSQL.

Содержание отчета к контрольной работе фактически представляет собой рабочий проект приложения на индивидуально выбранную тему. Наиболее приемлемым вариантом будет реализация программного приложения в контексте выполнения магистерской диссертации. Также возможно выполнение КР по новой теме предложенной студентом и согласованной с руководителем. По согласованию с руководителем возможна разработка и написание комплексных контрольных работ командой из нескольких студентов.

Типовая тематика контрольного проектирования по дисциплине "Технологии разработки программного обеспечения" напрямую связана с областью научных исследований кафедры ММиЦРБС и представлена в [разделе 3](#).

В процессе выполнения КР студент обязан разработать программное приложение в соответствии с индивидуально поставленной задачей, для чего должен:

- сформулировать функциональные требования к ПС и представить их в формальном виде;
- осуществить обоснованный выбор средств разработки приложения: СУБД, платформы для web-разработки, языковых средств, средств управления программным процессом, в т.ч. конфигурированием и тестированием, и пр.
- построить функциональные модели для решения задач информатизации и реализации отдельных микросервисов;

- разработать диаграммы реализации (deployment diagrams UML) программного приложения;
- выполнить установку и настройку ОС Linux для дальнейшего развертывания рабочей среды;
- выбрать систему контроля версий и на ее базе создать репозиторий для проекта;
- развернуть и настроить среду разработки программного приложения коллективным способом;
- развернуть Docker Debian и сформировать контейнер для разработки приложения с микросервисной архитектурой;
- разработать программные модули и интерфейсную часть приложения;
- выполнить тестирование приложения и описать полученные результаты;
- представить решения по развертыванию и дальнейшему сопровождению приложения;
- оформить отчет и защитить КР в соответствии с действующими стандартами.

Выполнение контрольной работы по дисциплине "Технологии разработки программного обеспечения" должно соответствовать утвержденному календарному графику и включает следующие этапы ([таблица 2.1](#)).

Таблица 2.1. Календарный график выполнения контрольной работы

Номер этапа	Наименование этапа	Неделя выполнения
1.	Выбор темы и согласование задания на контрольную работу	1 – 3
2.	Изучение информационных процессов в данной предметной области: описание существующей схемы управления объектом и ее недостатков, составление схемы инфопотоков, анализ существующих компьютерных разработок по выбранной теме	3 – 4
3.	Постановка задачи на создание приложения: формализация требований ко всем частям приложения и системе в целом, планирование ресурсов и итераций командного проекта	5 – 6
4.	Изучение функциональной части и построение функциональных моделей, диаграмм поведения и реализации UML, иллюстрирующих решение задачи информатизации и реализацию автоматизированных функций приложения	6 – 7
5.	Создание репозитория для командного проекта, организация	7 – 8

Номер этапа	Наименование этапа	Неделя выполнения
	версионного контроля компонентов приложения. Выбор средств разработки программного приложения коллективным способом	
6.	Программная реализация приложения. Описание микросервисов приложения и их взаимодействия между собой посредством API	9 – 10
7.	Тестирование приложения Unit-тестами, системное и исследовательское тестирование	11 – 12
8.	Разработка решений по развертыванию и дальнейшему сопровождению отдельных микросервисов и приложения в целом	13 – 14
9.	Оформление отчета к контрольной работе	14 – 15
10.	Сдача на проверку и защита контрольной работы	16 – 17

Более детальное описание структуры отчета к контрольной работе представлено в [разделе 4](#).

3. ТЕМАТИКА ПРОЕКТИРОВАНИЯ

В соответствии с квалификационной характеристикой направления подготовки 09.03.01 "Информатика и вычислительная техника" возможны следующие основные направления тематики контрольных работ по дисциплине "Технологии разработки программного обеспечения" (см. табл. 3.1)

Таблица 3.1. Тематика разработки ПС

№	Тематика
1.	Разработка программных систем, подсистем и модулей, обеспечивающих обработку информации по комплексу задач бизнес-процессов и функций управления ресурсами в различных сферах деятельности
2.	Разработка информационных управляющих систем (подсистем, модулей) для управления различными экономическими объектами
3.	Разработка систем (подсистем) компьютерного мониторинга, контроля и анализа состояния различных организационных объектов и их среды
4.	Разработка компьютерных подсистем управления технологическими процессами с непрерывным характером производства в разных областях промышленности: металлургическая, химическая, энергетическая и др.
5.	Разработка компьютерных подсистем визуализации движения подвижных объектов на транспорте
6.	Разработка компьютерных подсистем управления производством на предприятиях с дискретным и дискретно-непрерывным характером производства в разных областях промышленности: машиностроение, приборостроение, угольная и др.
7.	Разработка компьютерных экспертных систем для управления сложными объектами в разных сферах людской деятельности: производство, логистика, экономика, экология и др.
8.	Разработка компьютерных систем для сбора и обработки статистической информации в разных областях: транспорт, экономика, медицина, промышленность, сельское хозяйство и др.
9.	Разработка компьютерных учебных систем и систем для тестирования и проверки знаний, умений с использованием средств искусственного интеллекта
10.	Разработка компьютерных подсистем для мониторинга, контроля и анализа стана

№	Тематика
	окружающей среды, промышленных объектов, транспортных средств и систем и др.
11.	Разработка компьютерных подсистем для анализа и прогнозирования финансово-экономических показателей работы предприятий, потребление ресурсов разной природы с использованием средств искусственного интеллекта
12.	Составление тестовых последовательностей для нахождения дефектов цифровых устройств и систем с использованием современных информационных технологий
13.	Разработка компьютерных подсистем для анализа слабо структурированных проблемных ситуаций в организационных системах с использованием средств искусственного интеллекта
14.	Разработка компьютерных подсистем для анализа эффективности алгоритмов разного назначения
15.	<p>Разработка компьютерных подсистем оптимизации процессов разной природы с использованием эволюционного поиска и других методов:</p> <ul style="list-style-type: none"> - пассажирских и грузовых перевозок; - распределения грузов в кузове транспортного средства; - нормализации трафика в электрических, компьютерных, улично-дорожных и магистральных сетях; - генерации тестовых последовательностей для диагностики сложных цифровых устройств и систем; - построения сложных структур алгоритмов разного назначения; - раскроя разных материалов (дерево, стекло, металл и др.); - составление расписаний для упорядочения разных процессов: следование пассажирского транспорта, проведение учебных занятий, работы промышленного оборудования и т.п.
16.	Разработка компьютерных подсистем для контроля доступа к объектам, материальным и информационным ресурсам с использованием разных средств аутентификации
17.	<p>Разработка систем поддержки принятия решений (СППР) в разных сферах управления (производство, экономика, финансы и др.) с использованием:</p> <ul style="list-style-type: none"> - хранилищ данных; - средств аналитической обработки данных; - средств искусственного интеллекта

№	Тематика
18.	Разработка специализированных компьютерных систем для планирования эксперимента
19.	Разработка параллельных сред для моделирования сложных процессов и систем
20.	Разработка моделей и программных средств для анализа и определения параметров корпоративных информационных систем
21.	Разработка и исследования методов и структур аппаратной генерации тестов и анализа тестовых реакций при осуществлении диагностики вычислительной техники
22.	Разработка методов и средств выявления и устранения сбойных тестовых векторов при диагностировании цифровых устройств
23.	Разработка облачных технологий и сервисов для распределенной обработки данных или распределенного управления организационными объектами
24.	Разработка цифровых сервисов для цифровизации отдельных процессов и задач в разных областях деятельности
25.	Разработка систем (подсистем) поддержки принятия решения для менеджеров различного уровня
26.	Проектирование информатизации предприятий разных сфер на основе существующих систем электронного бизнеса (ERP, CRM, ECM, HRM, SCM)
27.	Разработка учебных информационных систем для тестирования и проверки знаний, умений с использованием средств искусственного интеллекта
28.	Исследование и адаптация различных объектов и процессов с применением статистического имитационного моделирования
29.	Создание и исследование информационных технологий управления в организационных системах
30.	Разработка системы риск-менеджмента в технике, социальных и экономических средах

4. ТРЕБОВАНИЯ К СТРУКТУРЕ КОНТРОЛЬНОЙ РАБОТЫ

Результаты выполнения контрольной работы оформляются каждым студентом в виде отчета. Отчет содержит весь проектный материал по созданию ПС, представленный в виде текста, расчетных формул, таблиц, рисунков и др. Типовая структура отчета следующая:

- титульный лист типовой формы (см. [приложение А](#));
- задание на контрольное проектирование на типовом бланке (см. [приложение Б](#));
- реферат;
- содержание;
- введение;
- основная часть, состоящая из нескольких разделов;
- заключение;
- перечень ссылок;
- приложения.

Задание на КР размещается сразу после титульного листа и является документом, который определяет объем и порядок выполнения работы, регламентирует исходные данные, функциональные требования к ПС и основные результаты проектирования. Задание утверждается руководителем КР.

Реферат предназначен для ознакомления с проектом. Он должен быть коротким, информативным и содержать сведения, которые позволяют представить сущность КР. Реферат должен содержать:

- библиографическое описание КР;
- текст реферата;
- перечень ключевых слов.

Библиографическое описание содержит сведения об объеме отчета, количество иллюстраций, таблиц, приложений, а также количество источников в перечне ссылок. Это описание выполняется согласно требованиям действующих стандартов по библиотечному и издательскому делу и имеет такой вид:

Контрольная работа: ____ стр., ____ рис., ____ табл., ____ прил., ____ ист.

При этом сведения приводятся с учетом всех приложений.

Текст реферата должен отображать информацию, представленную в отчете и, как правило, в определенной последовательности:

- объект разработки или исследования;

- цель работы;
- методы и средства разработки;
- результаты работы;
- основные технико-эксплуатационные, конструктивные и технологические характеристики;
- значимость работы и выводы.

Части реферата, сведения о которых отсутствуют, не приводят. Объем реферата не более 850 знаков. Реферат располагается на одной странице формата А4.

Ключевые слова, существенные для раскрытия сути КР, размещают после текста реферата. Перечень ключевых слов содержит от 5 до 15 слов (словосочетаний), напечатанных прописными буквами в именительном падеже в строку через запятые.

Лист реферата выполняется с основной надписью.

Содержание располагают после реферата, начиная с новой страницы.

Содержание включает:

- введение;
- последовательно перечисленные названия всех разделов, подразделов;
- заключение;
- перечень использованных источников;
- последовательно перечисленные приложения с заголовками.

В содержании указываются номера страниц, на которых расположено начало соответствующего материала.

Нумерация страниц в отчете сквозная, начинается с титульного листа, но номера страниц проставляются, начиная с содержания.

Перечень условных сокращений и аббревиатур располагают на отдельной странице. В перечне в виде списка пересчитываются обозначения и сокращения, использованные в тексте ПО, а через короткое тире по правую сторону приводится их расшифровка.

Введение отчета должно содержать оценку современного состояния и актуальности решаемой задачи, краткое описание используемых методов и средств решения задачи, цель работы и область ее применения.

Введение располагается с новой страницы.

Основная часть отчета должна включать следующие разделы:

1. Характеристика объекта информатизации.
2. Техническое задание на создание приложения.
3. Функциональная структура приложения.

4. Формирование среды разработки приложения.
5. Программная реализация приложения.
6. Тестирование приложения.
7. Развертывание приложения.

Примерный объем отчета 30 – 50 страниц без учета приложений.

В разделе 1 приводится краткая характеристика предметной области. Приводится подробная информация о структуре объекта информатизации, выполняемых функциях и решаемых задачах. Особо выделяется бизнес-процесс, подлежащий информатизации (процесс информатизации), дается его определение, формулируется цель (процесса, а не информатизации), приводятся правила организации и возможные ограничения. Описание должно давать представление о масштабе деятельности объекта, численности персонала, занятого в бизнес-процессе, характере взаимодействия с другими подразделениями и задачами.

В данном разделе необходимо разместить иллюстрацию, характеризующую объект и процесс информатизации. Это может быть какая-либо модель, формально представляющая объект, функционально-структурная схема, организационная диаграмма и т.п.

В разделе 2 приводится описывается назначение разрабатываемого продукта (ПС) и цели его создания. При описании назначения указывают вид деятельности, которая автоматизируется (расчет, управление, диагностика, планирование, прогнозирование, проектирование и т.п.) и перечень объектов информатизации, на которых предполагается ее использовать. При описании цели создания продукта приводят наименование и необходимые значения технических, технологических, производственно-экономических или других показателей объекта информатизации, которые должны быть достигнуты в результате создания ПС, и указывают критерии оценки достижения указанных целей.

Указывается перечень задач и автоматизируемых функций ПС, их назначение, основные характеристики. Приводятся требования к режимам функционирования ПС в целом и по отдельным задачам (функциям), разбивается на несколько пунктов в зависимости от количества задач (функций), например:

- 1 Требования к задаче (или функции) “...”
- 2 Требования к задаче (или функции) “...”
- 3 Требования к задаче (или функции) “...”

В каждом пункте указывается полное название задачи, функции или отдельной задачи. Надо помнить, что задача ПС – это функция или часть функции ПС, являющаяся формализованной совокупностью действий, выполнение которых приводит к результату

заданного вида. Поэтому в качестве задач надо выбирать такие, для которых можно четко сформулировать результат. Такими задачами, например, могут быть задача введения входных данных, задача формирования статистической отчетности, задача диагностики состояния объекта, задача анализа данных и принятия решений и др.

Кроме того, указывают нефункциональные требования к ПС (не менее пяти), отражающие её качественные характеристики согласно ГОСТ Р ИСО/МЭК 9126-93.

Также предъявляются требования к обеспечивающим подсистемам ПС:

- требования к организации данных, которые должны храниться в ПС, включая основные сущности БД, уместно представить диаграмму классов UML или логическую модель данных;
- требования к составу, области применения (ограничение) и средств использования в системе математических методов и моделей, типичных алгоритмов и алгоритмов, которые подлежат разработке;
- требования к программному обеспечению (ПО), в том числе к операционной среде, к инструментальным средствам разработки ПС и управления её жизненным циклом, к составу и функциям специального ПО, подлежащему разработке;
- требования к техническому обеспечению, а именно к конструктивным и эксплуатационным характеристикам средств ТО ПС, к конфигурациям компьютеров серверов и клиентских устройств.

При формулировании требований к обеспечивающим подсистемам необходимо учитывать, что эти требования обусловленные спецификой задач, которые решаются в ПС, и требованиями к другим частям ПС.

В разделе 3 приводятся проектные решения по функциональной структуре ПС согласно требованиям к задачам (функциям) ПС, приведенным в разделе 2. Приводится краткое (не более 100 слов) описание процесса функционирования объекта информатизации в условиях функционирования ПС. Составляется описание процессов выполнения каждой задачи (функции). Выделяются существенные структурные элементы и действия, определяются связи между ними. Уместно построить функционально-структурную схему ПС в виде диаграмм поведения UML. Представить структурную схему микросервисной архитектуры ПС по принципу: «одна задача – один сервис».

В разделе 4 отражается обоснованный выбор программного средства для разработки специального ПО. Это должен быть какой-либо фреймворк для создания web-приложения или WEB-сервер с PHP. Отражается установка выбранных СУБД и фреймворка для создания сервисов web-приложения, установка и настройка Docker Debian.

Также отражается создание репозитория проекта ПС для осуществления версионного контроля проекта (может быть использована любая система контроля версий: Microsoft Visual SourceSafe, IBM ClearCase, CVS, Subversion, Perforce, Git, Mercurial, Bazaar, Darcs и др.; рекомендуемая система управления версиями – Git), создание инициализирующего коммита, размещение в репозитории созданного проекта приложения.

Все шаги документируются иллюстрациями и краткими их комментариями.

В разделах 5 и 6 рассмотреть первую половину этапов цикла DevOps – CD: «Написание кода», «Сборка», «Тестирование».

В разделе 5 отражается процесс рабочего проектирования (реализации) ПС: создание БД, образа для реализации приложения, разработка и контейнеризация web-приложения посредством Docker с обеспечением логирования http-подключений (время подключения, IP-адрес клиента), размещение текущей версии проекта в удаленном репозитории Git. Примерная структура раздела:

- основные файлы приложения (при использовании Django – urls.py, models.py, settings.py);
- код, реализующий бизнес-логику приложения;
- докер-файл;
- средства web-интерфейса (функция index, шаблон старницы);
- файл docker-compose.yaml.

Код, реализующий бизнес-логику приложения, если он достаточно объемный, может быть вынесен в приложение.

Пример реализации простейшего web-приложения с микросервисной архитектурой и его краткого описания представлен в [приложении В](#).

Все приведенные компоненты описываются. Целесообразно привести структуру приложения в виде диаграммы компонентов UML, пример которой показан на рис. В.6.

Таким образом реализуется первый этап цикла DevOps части CD: «Написание кода».

В отдельном подразделе следует представить

В разделе 6 реализуется второй и третий этап DevOps части CD: «Сборка» и «Тестирование». Необходимо отразить настройку и работу автоматической сборки приложения на базе выбранной системы контроля версий и последующее тестирование проекта. Описать настроенные триггеры для активации сборки. Представить листинги unit-тестов и результаты их выполнения, текстовые описания и результаты прогона других тестов в соответствии с возможностями используемых средств автоматизированного тестирования.

Также необходимо доказательно и лаконично изложить доводы по поводу того, что ПС соответствует функциональным требованиям, которые сформулированы в ТЗ (раздел 2).

В разделе 7 необходимо представить и описать процедуру доставки приложения ПС до целевой системы и тестирование его работоспособности, используя CI/CD Pipeline. Рассмотреть вторую половину этапов цикла DevOps – CD: «Развертывание», «Поддержка и мониторинг», «Планирование» в условиях потенциального или реального использования ПС конечными пользователями. Для этапа «Планирование» представить проект нового функционала и план доработок для захода на следующий цикл DevOps CI/CD.

В целом текст основной части отчета должен давать полное представление о всех этапах жизненного цикла ПС, а разделы 5–7 должны полностью отражать все этапы цикла DevOps CI/CD. При этом множественные рисунки и копии экрана, не несущие основной информационной нагрузки, целесообразно располагать в приложениях к отчету.

Заключение (краткие выводы студента по контрольной работе) должно содержать краткое описание сущности решенной задачи, содержать оценку работоспособности разработанной ПС и правильности полученных результатов, а также перспективы развития ПС.

Текст заключения может разделяться на пункты.

Перечень использованных источников должен содержать список использованной литературы, оформленный согласно требованиям ГОСТ.

Приложения размещают после основной части отчета. В них подаются разные материалы (таблицы, рисунки, схемы, формы документов, листинги программ и др.), которые являются необходимым дополнением основного материала, но не могут быть последовательно размещенные в основной части отчета из-за большого объема или способа изложения. Приложения должны содержать: заполненные макеты входных документов, структуру БД ПС, листинги исходных текстов программного приложения ПС, распечатки сформированных отчетов, материалы, полученные при тестировании, а также по данным контрольного примера. Приложения обозначаются буквами, начиная с А, Б и т.д.

Текст отчета выполняется в редакторе MS Word 2003 и выше в печатном виде на стандартных листах формата А4 (210*290 мм) с соблюдением полей: верхнее и нижнее по 20 мм, левое – 25 мм, правое – 15 мм. Шрифт для набора текста – Times New Roman, размер – 14 пт, интервал – полуторный. Страницы отчета должны быть заполнены полностью. Исключение составляет последняя страница раздела. В конце остальных страниц допускается не более 2–3 пустых интервалов, свободное место необходимо дополнять текстом, масштабировать рисунки и таблицы, и т.д.

Графический материал: схема данных БД, диаграммы UML и BPWin, структура программного приложения, экранные формы, укрупненные блок-схемы алгоритмов, диаграммы для демонстрации результатов тестирования помещается по тексту в виде рисунков MS Word.

Более детальные требования к оформлению отчета представлены в [разделе 5](#).

5. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА

Отчет должна быть выполнен печатным способом с использованием компьютера и принтера на одной стороне листа белой бумаги формата А4 через полтора интервала. Для создания текста отчета рекомендуется использовать текстовый редактор MS Word. Шрифт на протяжении всего документа должен быть одинаковый: Times New Roman 14-го размера, за исключением оформления иллюстраций, таблиц и формул, в которых допускается использовать шрифт меньшего размера.

Текст отчета следует печатать, соблюдая следующие размеры полей: слева – 25 мм, справа – 15 мм, сверху и снизу – 20 мм.

При оформлении текста работы следует использовать абзацный отступ, который должен составлять 15-17 мм от левого поля документа

Вне зависимости от способа выполнения отчета качество напечатанного текста и оформления иллюстраций, таблиц, распечаток с ПЭВМ должно удовлетворять требованию их четкого воспроизведения.

Наименования структурных элементов отчета “Реферат”, “Содержание”, “Обозначения и сокращения”, “Введение”, “Заключение”, “Список использованных источников” служат заголовками структурных элементов отчета.

Основную часть отчета следует делить на разделы, подразделы и пункты. Разделы, подразделы и пункты следует нумеровать арабскими цифрами и записывать с абзацного отступа. Разделы должны иметь порядковую нумерацию в пределах всего текста, за исключением приложений. Каждый раздел работы должен начинаться с новой страницы.

Разделы, подразделы должны иметь заголовки. Пункты, как правило, заголовков не имеют. Заголовки должны четко и кратко отражать содержание разделов, подразделов. Заголовки разделов, подразделов и пунктов следует печатать с абзацного отступа с прописной буквы без точки в конце, не подчеркивая. Если заголовок состоит из двух предложений, их разделяют точкой.

Нумерация страниц отчета. Страницы отчета следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию по всему тексту отчета. Номер страницы проставляют в центре нижней части листа без точки. Титульный лист включают в общую нумерацию страниц. Номер страницы на титульном листе не проставляют. Иллюстрации и таблицы, расположенные на отдельных листах, включают в общую нумерацию страниц отчета.

Нумерация разделов, подразделов, пунктов, подпунктов отчета. Разделы отчета должны иметь порядковые номера в пределах всего документа, обозначенные арабскими цифрами без точки и записанные с абзацного отступа. Подразделы должны иметь нумерацию в пределах каждого раздела. Номер подраздела состоит из номеров раздела и подраздела, разделенных точкой. В конце номера подраздела точка не ставится. Разделы, как и подразделы, могут состоять из одного или нескольких пунктов. Если документ не имеет подразделов, то нумерация пунктов в нем должна быть в пределах каждого раздела, и номер пункта должен состоять из номеров раздела и пункта, разделенных точкой. В конце номера пункта точка не ставится.

Иллюстрации. Иллюстрации (чертежи, графики, схемы, компьютерные распечатки, диаграммы, фотоснимки) следует располагать в отчете непосредственно после текста, в котором они упоминаются впервые, или на следующей странице. Иллюстрации могут быть в компьютерном исполнении, в том числе и цветные. На все иллюстрации должны быть даны ссылки в отчете.

Иллюстрации, за исключением иллюстрации приложений, следует нумеровать арабскими цифрами сквозной нумерацией. Если рисунок один, то он обозначается “Рисунок 1”. Слово “Рисунок” и его наименование располагают посередине строки. Допускается нумеровать иллюстрации в пределах раздела. В этом случае номер иллюстрации состоит из номера раздела и порядкового номера иллюстрации, разделенных точкой. Например, Рисунок 1.1. Иллюстрации, при необходимости, могут иметь наименование и пояснительные данные (подрисуночный текст). Слово “Рисунок” и наименование помещают после пояснительных данных и располагают следующим образом: Рисунок 1.1 — Функциональная схема. Иллюстрации каждого приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения. Например, Рисунок А.3. При ссылках на иллюстрации следует писать “... в соответствии с рисунком 2” при сквозной нумерации и “... в соответствии с рисунком 1.2” при нумерации в пределах раздела.

Таблицы применяют для лучшей наглядности и удобства сравнения показателей. Название таблицы, при его наличии, должно отражать ее содержание, быть точным, кратким. Название таблицы следует помещать над таблицей слева, без абзацного отступа в одну строку с ее номером через тире.

При переносе части таблицы название помещают только над первой частью таблицы, нижнюю горизонтальную черту, ограничивающую таблицу, не проводят.

Таблицу следует располагать в отчете непосредственно после текста, в котором она упоминается впервые, или на следующей странице. На все таблицы должны быть ссылки в отчете. При ссылке следует писать слово “таблица” с указанием ее номера.

Таблицу с большим количеством строк допускается переносить на другой лист (страницу). При переносе части таблицы на другой лист (страницу) слово “Таблица” и номер ее указывают один раз справа над первой частью таблицы, над другими частями пишут слово “Продолжение” и указывают номер таблицы, например: “Продолжение таблицы 1”. При переносе таблицы на другой лист (страницу) заголовок помещают только над ее первой частью.

Таблицу с большим количеством граф допускается делить на части и помещать одну часть под другой в пределах одной страницы. Если строки и графы таблицы выходят за формат страницы, то в первом случае в каждой части таблицы повторяется головка, во втором случае — боковик.

Если повторяющийся в разных строках графы таблицы текст состоит из одного слова, то его после первого написания допускается заменять кавычками; если из двух и более слов, то при первом повторении его заменяют словами “То же”, а далее — кавычками. Ставить кавычки вместо повторяющихся цифр, марок, знаков, математических и химических символов не допускается. Если цифровые или иные данные в какой-либо строке таблицы не приводят, то в ней ставят прочерк.

Таблицы, за исключением таблиц приложений, следует нумеровать арабскими цифрами сквозной нумерацией. Допускается нумеровать таблицы в пределах раздела. В этом случае номер таблицы состоит из номера раздела и порядкового номера таблицы, разделенных точкой. Таблицы каждого приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения. Если в документе одна таблица, то она должна быть обозначена “Таблица 1” или “Таблица В.1”, если она приведена в приложении В.

Заголовки граф и строк таблицы следует писать с прописной буквы в единственном числе, а подзаголовки граф — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблиц точки не ставят. Таблицы слева, справа и снизу, как правило, ограничивают линиями. Допускается применять размер шрифта в таблице меньший, чем в тексте. Горизонтальные и вертикальные линии, разграничивающие строки таблицы, допускается не проводить, если их отсутствие не затрудняет пользование таблицей. Заголовки граф, как правило, записывают параллельно строкам таблицы. При

необходимости допускается перпендикулярное расположение заголовков граф. Головка таблицы должна быть отделена линией от остальной части таблицы.

Формулы и уравнения. Уравнения и формулы следует выделять из текста в отдельную строку. Выше и ниже каждой формулы или уравнения должно быть оставлено не менее одной свободной строки. Если уравнение не уместится в одну строку, то оно должно быть перенесено после знака равенства (=) или после знаков плюс (+), минус (-), умножения (х), деления (:), или других математических знаков, причем знак в начале следующей строки повторяют. При переносе формулы на знаке, символизирующем операцию умножения, применяют знак “Х”.

Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, в которой они даны в формуле.

Формулы в отчете следует нумеровать порядковой нумерацией в пределах всего отчета арабскими цифрами в круглых скобках в крайнем правом положении на строке. Одну формулу обозначают – (1). Формулы, помещаемые в приложениях, должны нумероваться отдельной нумерацией арабскими цифрами в пределах каждого приложения с добавлением перед каждой цифрой обозначения приложения, например формула (В.1). Ссылки в тексте на порядковые номера формул дают в скобках. Пример – ... в формуле (1). Допускается нумерация формул в пределах раздела. В этом случае номер формулы состоит из номера раздела и порядкового номера формулы, разделенных точкой, например (3.1).

Ссылки. В отчете допускаются ссылки на данный документ, стандарты, технические условия и другие документы при условии, что они полностью и однозначно определяют соответствующие требования и не вызывают затруднений в пользовании документом.

Ссылаться следует на документ в целом или его разделы и приложения. Ссылки на подразделы, пункты, таблицы и иллюстрации не допускаются, за исключением подразделов, пунктов, таблиц и иллюстраций данного документа.

При ссылках на стандарты и технические условия указывают только их обозначение, при этом допускается не указывать год их утверждения при условии полного описания стандарта в списке использованных источников в соответствии с ГОСТ 7.1. Ссылки на использованные источники следует приводить в квадратных скобках.

Перечень обозначений и сокращений. Перечень должен располагаться столбцом. Слева в алфавитном порядке приводят сокращения, условные обозначения, символы, единицы физических величин и термины, справа — их детальную расшифровку.

Список использованных источников. Сведения об источниках следует располагать в порядке появления ссылок на источники в тексте отчета и нумеровать арабскими цифрами без точки и печатать с абзацного отступа.

Приложения оформляют как продолжение данного документа на последующих его листах или выпускают в виде самостоятельного документа. В тексте документа на все приложения должны быть даны ссылки. Приложения располагают в порядке ссылок на них в тексте документа, за исключением справочного приложения “Библиография”, которое располагают последним.

Каждое приложение следует начинать с новой страницы с указанием наверху посередине страницы слова “Приложение”, его обозначения и степени. Приложение должно иметь заголовок, который записывают симметрично относительно текста с прописной буквы отдельной строкой. Приложения обозначают заглавными буквами русского алфавита, начиная с А, за исключением букв Ё, З, Й, О, Ч, Ъ, Ы, Ъ. После слова “Приложение” следует буква, обозначающая его последовательность. Допускается обозначение приложений буквами латинского алфавита, за исключением букв I и O. В случае полного использования букв русского и латинского алфавитов допускается обозначать приложения арабскими цифрами. Если в документе одно приложение, оно обозначается "Приложение А".

Текст каждого приложения, при необходимости, может быть разделен на разделы, подразделы, пункты, подпункты, которые нумеруют в пределах каждого приложения. Перед номером ставится обозначение этого приложения. Приложения должны иметь общую с остальной частью документа сквозную нумерацию страниц. При необходимости такое приложение может иметь "Содержание". Приложениям или частям, выпущенным в виде самостоятельного документа, обозначение присваивают как части документа с указанием в коде документа ее порядкового номера.

6. ПОРЯДОК СДАЧИ НА ПРОВЕРКУ И ЗАЩИТЫ КОНТРОЛЬНОЙ РАБОТЫ

Защита контрольной работы проводится с целью:

- определения уровня владения студентом универсальных, общепрофессиональных и профессиональных компетенций в соответствии с требованиями ФГОС 3++ по направлению подготовки 09.03.01 «Информатика и вычислительная техника»;
- предоставления студенту возможности тренировки в публичном выступлении и демонстрации полученных результатов;
- получения объективной оценки работы.

Защиты КР проводятся на кафедре ММиЦРБС в соответствии с календарным графиком (см. [табл. 2.1](#)) в последние 2 недели учебного семестра. Очередность прохождения защиты КР студентами и конкретные даты защит устанавливается руководителем, исходя из предпочтений студентов, и оформляется в виде графика защиты КР, который подписывает руководитель КР и заведующий кафедрой ММиЦРБС. График защиты доводится до сведения студентов не позже, чем за 20 дней до конца зачетной недели

Защита проводится комиссией, в состав которой, кроме руководителя, могут входить другие члены профессорско-преподавательского состава кафедры ММиЦРБС.

Не менее чем за 3 дня до даты защиты, определяемой согласно графику защиты, студент должен сдать полностью подготовленный отчет на проверку. Отчета представляется в электронном виде и загружается через сайт. С порядком отправки выполненной работы можно ознакомиться в инструкции по отправке, доступной по адресу: <http://ndo.sibsutis.ru/dist03/newssystem.htm>.

В процессе защиты студент должен в течение 5 – 7 минут кратко раскрыть актуальность, цель и задачи контрольной работы, содержание работы, используемый модельно-методический аппарат и т.п., изложить полученные результаты и выводы, после чего ответить на вопросы членов комиссии.

Представляется целесообразным подготовка конспекта доклада, однако студент должен свободно излагать содержание работы, не пользуясь постоянно подготовленным докладом. Выступление должно быть иллюстрировано таблицами, графиками, схемами и т.п. Демонстрационный материал должен быть представлен методом проекции. При ответах на вопросы студент имеет право пользоваться отчетом.

По результатам защиты комиссия принимает решение об оценке контрольной работы. Оценка КР при ее защите основывается на следующих критериях:

- соответствие работы утвержденной теме;
- степень решения поставленных задач, обоснованность, значимость и полнота сделанных выводов и предложений;
- качество выступления и ответов на поставленные вопросы при защите КР.

Оценка по результатам защиты выставляется по четырехбалльной шкале: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка **«отлично»** выставляется, если:

- работа выполнена в соответствии с заданием, содержание и оформление работы в полной мере соответствует заявленной теме и требованиям методических указаний;
- выступление на защите структурировано, раскрыты актуальность темы, цель и задачи работы, структура объекта информатизации, логика представленных выводов;
- длительность выступления соответствует регламенту;
- ответы на вопросы членов комиссии логичны, раскрывают сущность вопроса, показывают самостоятельность и глубину изучения проблемы;
- применение информационных технологий, как в самой работе, так и во время выступления широко;
- эффективная работа студента в соответствии с календарным графиком (см. [табл. 2.1](#)) с положительной динамикой в течение семестра.

Оценка **«хорошо»** выставляется, если:

- работа выполнена в соответствии с заданием, содержание и оформление работы в полной мере соответствует заявленной теме и требованиям методических указаний;
- выступление на защите структурировано, допускаются одна-две неточности при раскрытии актуальности темы, целей и задач работы, объекта информатизации, допускается погрешность в логике вывода одного из наиболее значимых выводов, которая устраняется в ходе дополнительных уточняющих вопросов;
- длительность выступления в основном соответствует регламенту;
- в ответах на вопросы членов комиссии допущено нарушение логики, но, в целом, раскрыта сущность вопроса, показывающая самостоятельность и глубину изучения проблемы;
- применение информационных технологий, как в самой работе, так и во время выступления ограничено;
- работа студента не в полной мере соответствует календарному графику (см. [табл. 2.1](#)).

Оценка **«удовлетворительно»** выставляется, если:

- работа выполнена в соответствии с заданием, но содержание и оформление работы не в полной мере соответствует теме и требованиям методических указаний;
- выступление на защите структурировано, допускаются неточности при раскрытии актуальности темы, целей и задач работы, объекта информатизации, допущена грубая погрешность в логике вывода одного из наиболее значимых выводов, которая при указании на нее устраняется с трудом;
- длительность выступления не соответствует регламенту;
- ответы на вопросы членов комиссии не раскрывают до конца сущности вопросов, показывают недостаточную самостоятельность и глубину изучения проблемы;
- применение информационных технологий, как в самой работе, так и во время защиты недостаточно;
- работа студента в основном не соответствует календарному графику (см. [табл. 2.1](#)), большая часть работы выполнена в последние 2 недели;
- в ходе защиты студент демонстрирует понимание содержания допущенных им ошибок.

Оценка **«неудовлетворительно»** выставляется, если:

- работа выполнена с нарушением задания, содержание и оформление отчета не отвечает требованиям методических указаний;
- выступление на защите не структурировано, недостаточно раскрываются актуальность темы, цели и задачи работы, объект информатизации, допускаются грубые погрешности в логике вывода нескольких из наиболее значимых выводов, которые при указании на них, не устраняются;
- длительность выступления не соответствует регламенту;
- ответы на вопросы членов комиссии не раскрывают сущности вопроса, показывают отсутствие самостоятельности и глубины изучения проблемы студентом;
- информационные технологии не применялись ни в работе, ни при выступлении;
- работа студента полностью не соответствует календарному графику (см. [табл. 2.1](#)), материал отчета руководителю представлен впервые на защите;
- в ходе процедуры защиты студент демонстрирует непонимание содержания допущенных им ошибок.

В случае неявки студента на защиту работы по уважительной причине студенту назначается защита в другое время, но не позже даты окончания зачетной недели.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Список основной литературы (в соответствии с ГОСТ Р 7.1.-2003)

1. Митра Р., Надареишвили И. Микросервисы. От архитектуры до релиза. — СПб.: Питер, 2023. — 336 с.: ил. — (Серия «Бестселлеры O'Reilly»).
2. Доррер, Г. А. Методология программной инженерии : учебное пособие / Г. А. Доррер. — Красноярск : Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева, 2021. — 190 с.
3. Меле, А. Django 2 в примерах / А. Меле ; перевод Д. В. Плотникова. — Москва : ДМК Пресс, 2019. — 408 с. — ISBN 978-5-97060-746-6. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/126199.html>. — Режим доступа: для авторизир. пользователей
4. Меле А. Django 4 в примерах / пер. с англ. А. В. Логунова. — М.: ДМК Пресс, 2023. — 800 с.
5. Шениг, Г. -Ю. PostgreSQL 11. Мастерство разработки / Г. -Ю. Шениг ; перевод А. А. Слинкин. — Москва : ДМК Пресс, 2019. — 352 с. — ISBN 978-5-97060-671-1. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/125100.html>. — Режим доступа: для авторизир. пользователей.
6. Ригс, С. Администрирование PostgreSQL 9. Книга рецептов / С. Ригс, Х. Кросинг ; перевод Е. В. Самохвалов. — Москва : ДМК Пресс, 2018. — 364 с. — ISBN 978-5-97060-609-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/124979.html>. — Режим доступа: для авторизир. пользователей.
7. Златопольский, Д. М. Основы программирования на языке Python / Д. М. Златопольский. — 2-е изд. — Москва : ДМК Пресс, 2018. — 396 с. — ISBN 978-5-97060-641-4. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/124998.html> (дата обращения: 18.10.2022). — Режим доступа: для авторизир. пользователей
8. Карякин, М. И. Технологии программирования и компьютерный практикум на языке Python : учебное пособие / М. И. Карякин, К. А. Ватульян, Р. М. Мнухин. — Ростов-на-Дону, Таганрог : Издательство ЮФУ, 2022. — 241 с. — ISBN

978 - 5 - 9275 - 4108 - 9. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/125718.html> (дата обращения: 09.11.2022). — Режим доступа: для авторизир. пользователей.

9. Ленц, М. Python: непрерывная интеграция и доставка / М. Ленц ; перевод А. Е. Мамонов, Д. А. Беликов. — Москва : ДМК Пресс, 2020. — 168 с. — ISBN 978-5-97060-797-8. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: <https://www.iprbookshop.ru/126206.html> (дата обращения: 30.11.2022). — Режим доступа: для авторизир. пользователей.

Список дополнительной литературы (в соответствии с ГОСТ Р 7.1.-2003)

1. Методы программирования : учебно-методическое пособие / авторы В. В. Подколзин, А. Н. Полетайкин, Е. П. Лукашик [и др.] ; Министерство науки и высшего образования Российской Федерации, Кубанский государственный университет. - Краснодар : Кубанский государственный университет, 2020. - 174 с.
2. Гагарина, Л. Г. Разработка и эксплуатация автоматизированных информационных систем [Электронный ресурс] : учебное пособие для СПО / Л. Г. Гагарина. - Москва : ИД "ФОРУМ" : ИНФРА-М, 2018. - 384 с. - <http://znanium.com/catalog.php?bookinfo=942717>. - ЭБС «ZnaniUM.COM».
3. Полетайкин, А. Н. Социальные и экономические информационные системы. Законы функционирования и принципы построения [Электронный ресурс] : учебное пособие / А. Н. Полетайкин. — Электрон. текстовые данные. — Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2016. — 241 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/54800.html>.
4. Дерябкин, В. П. Проектирование информационных систем по методологии UML с использованием Qt-технологии программирования [Электронный ресурс] : учебное пособие / В. П. Дерябкин, В. В. Козлов. — Электрон. текстовые данные. — Самара : Самарский государственный технический университет, ЭБС АСВ, 2017. — 156 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/83601.html>.
5. Носова, Л. С. Case-технологии и язык UML [Электронный ресурс] : учебно-методическое пособие / Л. С. Носова. — 2-е изд. — Электрон. текстовые данные. — Челябинск, Саратов : Южно-Уральский институт управления и экономики, Ай Пи Эр Медиа, 2019. — 67 с. — 978-5-4486-0670-0. — Режим доступа: <http://www.iprbookshop.ru/81479.html>.

6. Мостовой Я.А. Управление программными проектами [Электронный ресурс]: учебное пособие/ Мостовой Я.А.— Электрон. текстовые данные.— Самара: Поволжский государственный университет телекоммуникаций и информатики, 2016.— 103 с.— Режим доступа: <http://www.iprbookshop.ru/71894.html>.— ЭБС «IPRbooks».
7. Влацкая И.В. Проектирование и реализация прикладного программного обеспечения [Электронный ресурс]: учебное пособие/ Влацкая И.В., Заельская Н.А., Надточий Н.С.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2015.— 119 с.— Режим доступа: <http://www.iprbookshop.ru/54145.html>.— ЭБС «IPRbooks».
8. Джефф Форсье. Django - Разработка веб-приложений на Python. 2009.
9. Адриан Головатый. Django - Подробное руководство, 2-е издание. 2010.
10. <https://www.djangoproject.com>
11. <https://tproger.ru/articles/pochemu-vam-stoit-vybrat-frejmwork-django-dlja-svoego-sledujushhego-proekta/>
12. <https://medium.com/nuances-of-programming/python-и-веб-разработка-краткое-руководство-858bf8987691>

ПРИЛОЖЕНИЕ А

Образец титульного листа

Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации
(Минцифры РФ)

Федеральное государственное бюджетное образовательное учреждение
высшего образования «Сибирский государственный университет
телекоммуникаций и информатики»
(СибГУТИ)

Институт заочного образования

ОТЧЁТ

о выполнении контрольной работы по дисциплине “Технологии разработки
программного обеспечения”

на тему: _____

Исполнитель

студент гр. _____
(группа) (подпись) (ФИО)

Руководитель

(подпись) (должность, ФИО)

Оценка _____ Дата защиты _____

Новосибирск,
20__

ПРИЛОЖЕНИЕ Б

Образец бланка задания на КР

Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации
(Минцифры РФ)

Федеральное государственное бюджетное образовательное учреждение
высшего образования «Сибирский государственный университет
телекоммуникаций и информатики»
(СибГУТИ)

Институт заочного образования

ЗАДАНИЕ НА КОНТРОЛЬНУЮ РАБОТУ

по дисциплине “Технологии разработки программного обеспечения”
студенту _____ группы _____

1. Тема контрольной работы: _____

2. Перечень исходных материалов: _____

3. Основные функции приложения: _____

4. Используемые инструментальные средства: _____

5. Основные результаты работы приложения: _____

6. Функционал приложения, обеспечиваемого контейнерами Docker:

1. _____

2. _____

3. _____

7. Срок сдачи проекта на проверку и защиты: с _____.____.20__ по _____.____.20__ г.

Руководитель _____
(подпись)

(должность, ФИО)

ПРИЛОЖЕНИЕ В

Пример реализации простейшего web-приложения с микросервисной архитектурой

Имеется сайт агентства по поиску вакансий, в функционал которого входит вывод данных об имеющихся вакансиях и средней предлагаемой зарплате по всем вакансиям (рис. В.1).

This is a system of vacancy!

Middle check in base: 55400.0

Vacansies in base:

Name	Pay	Phone
Программист	125000	79132098743
Системный администратор	79000	79132098743
Оператор ПК	29000	79612380965
Настройщик рояля	19000	79099831881
Уборщик	25000	79132404033

Рис. В.1 – Внешний вид исправного сайта агентства по поиску вакансий

Предполагается, что за функционал сайта отвечает отдельное приложение, а функционал, обеспечивающий ведение информации о вакансиях, также реализуется отдельным приложением. Взаимодействие приложения сайта с приложением вакансий обеспечивается путем WEB-запросов.

В случае остановки приложения, обеспечивающего взаимодействие с вакансиями, сайт станет неисправен. Однако он сохранит работоспособность, поскольку основное приложение сайта продолжит работать, но информация о вакансиях будет недоступна (см. рис. В.2).

This is a system of vacancy!

Middle check in base: Service is unavailable

Vacansies in base:

Service is unavailable!

Рис. В.2 – Работа неисправного сайта в условиях остановки приложения, обеспечивающего взаимодействие с вакансиями

Приведем пример реализации приложения вакансий. Создадим джанго-приложение (см. практическую работу 3)

Пример файла **urls.py**:

```
from django.contrib import admin
from django.urls import path, include, re_path
from . import views

urlpatterns = [
    path(r'req', views.req, name='req'),
    path(r'req1', views.req1, name='req1'),
]
```

В данном файле имеются две функции обработки req и req1, вызывающиеся по url /req или req1. Приведем эти функции.

```
def req(request):
    a={}
    b=vacancy.objects.all()
    j=0
    k=0
    for i in b:
        j=j+i.zarpl
        k+=1
    a['sz']=j/k
    otv=json.dumps(a)
    return HttpResponse(otv)

def req1(request):
    a={}
    a['vac']=[]
    b=vacancy.objects.all()
```

```

for i in b:
    el={}
    el['id'], el['name'], el['zarpl'], el['tel'] = i.pk,
i.name, i.zarpl, i.tel
    a['vac'].append(el)
otv=json.dumps(a)
return HttpResponse(otv)

```

Как можно видеть, данные функции выполняют чтение вакансий из БД, упаковывают результат в словарь, переводят в формат json и возвращают HTTP-ответ в виде json-строки.

База данных состоит из одной таблицы, описание которой приведено в models.py:

```

class vacancy(models.Model):
    name = models.CharField(max_length=200)
    zarpl = models.IntegerField()
    tel = models.CharField(max_length=20)

```

Далее папку с приложением и проектом необходимо скопировать в образ контейнера докер. Пример докер-файла:

```

# Забираем официальный базовый образ Python
FROM python:3.10-alpine
# Установка необходимых библиотек, перечисленных в файле
requirements.txt
RUN pip install --upgrade pip
COPY ./requirements.txt .
RUN pip install -r requirements.txt
# Копирование папок проекта и приложения в корень контейнера
(докер файл в этой же директории), а так же задание переменных
окружения
COPY . .
ENV PYTHONDONTWRITEBYTECODE=1
ENV PYTHONUNBUFFERED=1
#Команда запуска
CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]

```

Файл requirements.txt

```

Django
psycopg2-binary
requests

```

Далее покажем пример приложения самого сайта:

```

Urls.py
from django.contrib import admin
from django.urls import path, include, re_path
from . import views

urlpatterns = [
    path(r'', views.index, name='index'),
]

```

Функция обработки **index**:

```
def index(request):
    try:
        res = requests.get('http://vacan:8000/req')
        load=json.loads(str(res.text))
    except:
        load={}
        load['sz']='Service is unavailable'

    try:
        res1 = requests.get('http://vacan:8000/req1')
        load1=json.loads(str(res1.text))
    except:
        load1={}
        load1['vac']='NONE'

    l1={}
    l1['param1']=load
    l1['param2']=load1

    return render(request, 'index.html',l1)
```

Данная функция делает запрос к приложению вакансий, получает json, парсит его и отправляет в шаблон.

ВНИМАНИЕ! Полу жирным выделено название контейнера **vacan** с приложением вакансий. Это должно быть ИМЯ КОНТЕЙНЕРА ДОКЕР данного приложения, так как внутри докер-сети возможно взаимодействие именно по именам контейнера и можно абстрагироваться от ip-адресов.

Шаблон:

```
<center><H1> This is a system of vacancy! </H1></center></br>
<center><H1>      Middle      check      in      base:      {{param1.sz}}
</H1></center></br>
<center><H1> Vacansies in base: </br> </H1></center></br>
{% if param2.vac == 'NONE' %}
  <center><H3> Service is unavailable! </br> </H3></center></br>
{% else %}
<center>
<table border="1">
  <tr><td>Name</td><td>Pay</td><td>Phone</td></tr>
  {% for i in param2.vac %}
    <tr><td>{{ i.name }}</td><td>{{ i.zarpl }}</td><td>{{ i.tel
  }}</td></tr>
  {% endfor %}
</table>
</center>
{% endif %}
```

Создание образа докер и файл докер в данном случае не отличается от приложения вакансий. Приведем пример файла docker-compose:

```

version: '3.7'

services:
#Контейнер вакансий
  vacan:
#указывает , что все данные для построения образа находится в
#поддиректории imgvacan/
    build: imgvacan/
# задает имя контейнера
    container_name: vacan
    restart: always
#Аналогично собирается образ сайта
  main:
    build: imgmain/
    container_name: main
    restart: always
  ports:
    - 8001:8001

```

Обратите внимание на то, что для контейнера **vacan** порт не пробрасывается на хост-систему потому, что взаимодействие осуществляется внутри сети докер. Для контейнера с сайтом порт пробрасывается на хост-систему.

Для запуска сборки образа и старта контейнера необходимо находясь в директории с файлом **docker-compose.yml** запустить команду:

```
docker-compose up -d
```

Начнется сборка, процесс которой отобразится так, как показано на рис. В.3.

```

root@debian12uvs1n:/dima# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@debian12uvs1n:/dima# docker-compose up -d
[+] Building 2.2s (4/5)                                docker:default
=> [vacan internal] load .dockerignore                  0.2s
=> => transferring context: 2B                          0.0s
=> [main internal] load .dockerignore                  0.2s
=> => transferring context: 2B                          0.0s
=> [vacan internal] load build definition from Dockerfile 0.2s
=> => transferring dockerfile: 339B                     0.0s
=> [main internal] load build definition from Dockerfile 0.2s
=> => transferring dockerfile: 339B                     0.0s
=> [vacan internal] load metadata for docker.io/library/python:3.10-alpine 2.0s

```

Рис. В.3 – Процесс сборки приложения с МСА

В случае успешной сборки запустятся контейнеры (см. рис. В.4).

```

=> => exporting layers
=> => writing image sha256:54cd62fb53a223fe0e6a7f6ca75eaf6080c85309bfa569
=> => naming to docker.io/library/dima-vacan
[+] Running 3/3
 ✓ Network dima_default   Created
 ✓ Container vacan        Started
 ✓ Container main         Started
root@debian12uvs1n:/dima#

```

Рис. В.4 – Результат успешной сборки и запуск контейнеров

Командой `docker ps` можно посмотреть запущенные контейнеры (рис. В.5).

```
root@debian12:~# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
813379cd7b4a   dima-main  "python manage.py ru..." 50 seconds ago  up 48 seconds  0.0.0.0:8001->8001/tcp, :::8001->8001/tcp  main
1c6ca4d11cc8   dima-vacan "python manage.py ru..." 50 seconds ago  up 48 seconds  0.0.0.0:8001->8001/tcp, :::8001->8001/tcp  vacan
```

Рис. В.5 – Результат выполнения команды `docker ps`

Проверка работы приложения показана на рис. В.1.

Данное приложение имеется на гитлаб: <http://pa.k-lab.su/dad/msa.git>

В данном разделе представлен простейший пример микросервисной архитектуры, физическая реализация которой показана на рис. В.6. Приложение **Main** на основе данных, полученных через собственный web-интерфейс посредством функции **Requests** инициирует запросы к приложению **Vacan** через Web API. Последний рассылает сообщения функциям **Req** и **Req1**, реализованные на базе приложения **Vacan**, которые, обращаясь к БД **Details**, вычисляют требуемые результаты и возвращают их обратно в **Main**. Приложение **Main** обладает собственной БД **Mainbase**, в которую может записать, например, параметры пользовательского запроса и полученный от **Vacan** результат.

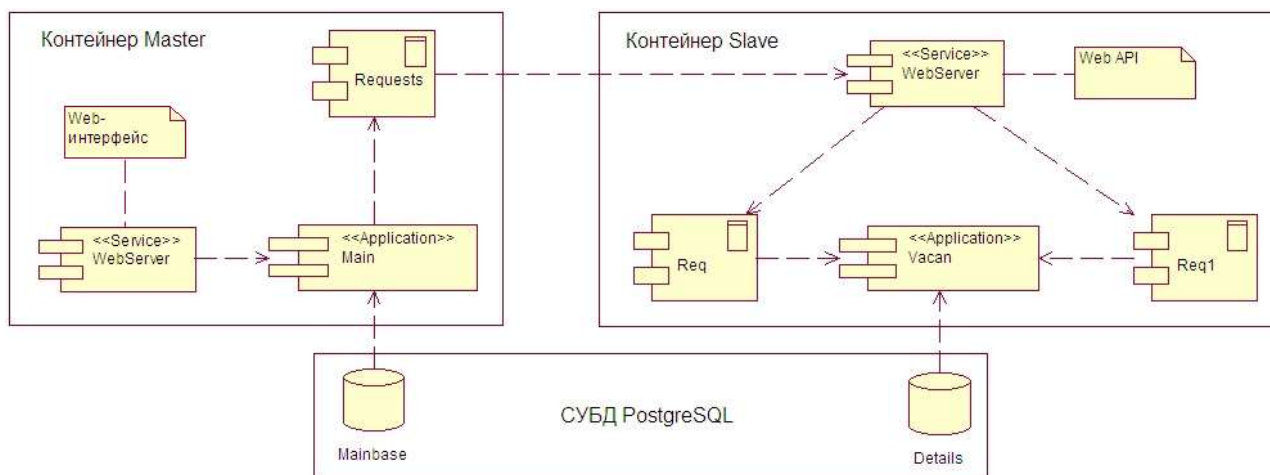


Рис. В.6 – Диаграмма компонентов UML MCA-приложения

В более сложных системах строятся кластеры хостов, устанавливаются системы определения сервисов и т.д. Далее можно вынести в отдельное приложение (сервис), например, работу с базой соискателей, систему генерации отчетов и пр. Web-запросы можно реализовать более сложными, передавая, например, параметры фильтрации и т.д. Несмотря на то, что в данном примере оба приложения (сервиса) реализованы с помощью Django, однако это совсем не обязательно. Сервисы могут быть реализованы с помощью любых инструментов, главное согласовать форматы WEB-запросов и ответов на запросы.