

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии

СОГЛАСОВАНО

УТВЕРЖДАЮ

Приглашенный преподаватель
департамента программной инженерии
факультета компьютерных наук, бакалавр

Академический руководитель
образовательной программы
«Программная инженерия»

_____ Г. М. Сосновский
«_____» _____ 2024 г.

_____ Н. А. Павловчев
«_____» _____ 2024 г.

ОБРАЗОВАТЕЛЬНАЯ ПЛАТФОРМА.
ПОДСИСТЕМА «COURSES»

Руководство программиста

Лист Утверждения

RU.17701729.09.12-01 33 01-1-ЛУ

Инв. № подл	Подл. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Исполнитель: Студент группы БПИ 217
Киселёв И. А.
«_____» _____ 2024 г.

УТВЕРЖДЁН

RU.17701729.09.12-01 33 01-1-ЛУ

ОБРАЗОВАТЕЛЬНАЯ ПЛАТФОРМА.
ПОДСИСТЕМА «COURSES»

Руководство программиста

RU.17701729.09.12-01 33 01-1

Листов 20

Инв. № подл	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Москва 2024

Аннотация

Данный документ является руководством программиста, предназначенного для ознакомления программистом, эксплуатирующим программу.

Данный документ содержит следующие разделы: Раздел «Назначение и условия применения программы» содержит назначение и функции, выполняемые программой, условия, необходимые для выполнения программы (объем оперативной памяти, требования к составу и параметрам периферийных устройств, требования к программному обеспечению и т.п.).

Раздел «Характеристика программы» содержит описание основных характеристик и особенностей программы (режим работы, средства контроля правильности выполнения и т.п.).

Раздел «Обращение к программе» содержит инструкции к использованию программы.

Раздел «Входные и выходные данные» содержит описание организации используемой входной и выходной информации.

Раздел «Сообщения программисту» содержит тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Содержание

1 Назначение программы	4
1.1 Функциональное назначение	4
1.2 Эксплуатационное назначение	4
1.3 Функции и назначение программы	5
2 Условия применения программы	6
2.1 Требование к техническим характеристикам устройств	6
3 Характеристика программы	7
3.1 Процесс развертки	7
4 Обращение к программе	8
4.1 Запуск и завершение программы	8
4.2 Файлы конфигураций	8
4.3 Описание использования API	9
5 Входные и выходные данные	10
5.1 Описание организации и использования API	10
5.2 Описание использования спецификации API	10
6 Сообщения	12
6.1 Описание работы со средствами мониторинга	12
7 Список используемой литературы	13
8 Глоссарий	19

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1. Назначение программы

1.1. Функциональное назначение

Система должна предоставить клиентам возможность просматривать курс, приобретать доступ к урокам курсов, изучать купленные курсы, прослеживать свой прогресс.

Клиент, получив доступ и роль "учитель" сможет размещать курсы. Данный пользователь, учитель, сможет создавать и редактировать свои шаблоны курсов, публиковать их (и получать часть прибыли с их продажи).

Пользователи с ролями модератора смогут просматривать курсы (и их содержимое) и обрабатывать заявки на публикацию курсов.

Подсистема отвечает за бизнес логику работы с курсами: доступы для учителей и учеников, создание изменение курсов. И не отвечает за авторизацию, аутентификацию, рассылку нотификаций и продажу продуктов (обработку денежных транзакций). Для лучшего понимания функционального назначения подсистемы ее диаграмма прецедентов представлена в Приложении 3.

1.2. Эксплуатационное назначение

Данная система будет представлено в виде нескольких подсистем. Часть подсистем будет реализовать клиентскую часть, веб-браузер с программой, обрабатывающий действия пользователя на веб-сайте. Клиентская часть будет выводить визуализировать необходимые данные пользователь и предоставлять пользовательский интерфейс для работы с информационной системой. Клиентская часть будет получать данные из серверной части.

Серверная часть представляет собой подсистемы, запущенные во внутренней сети. Данные подсистемы будут запущены в контейнерах на одном или нескольких серверах. Данные подсистемы будут принимать запросы по сети от друг друга или от клиентской части. Подсистемы и связь между ними показана в Приложении 1 и Приложении 2.

Подсистемы Courses – одна из подсистем серверной части.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1.3. Функции и назначение программы

Для использованием клиентом, через сервис «API Gateway». Программа предоставляет следующие функции

- 1) Предоставление краткой информации о курсе
- 2) Предоставление информации всех составляющих курса
- 3) Предоставление информации курсов
- 4) Предоставление информации всех доступных курсов
- 5) Проверка курса
- 6) Покупка курса
- 7) Выдача доступа на курс
- 8) Создание курса и его составляющих
- 9) Первоначальная проверка данных и составляющих курса
- 10) Изменение курса и его составляющих
- 11) Создание заявки на публикацию курса
- 12) Одобрение заявки на публикацию курса
- 13) Отклонение заявки на публикацию курса
- 14) Отметка о прохождении урока
- 15) Просмотр статистики прохождения курсов

Для использования сервисом «Finances». Программа предоставляет следующие функции

- 1) Подтверждение оплаты

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

2. Условия применения программы

2.1. Требование к техническим характеристикам устройств

Минимальные системные требования к техническим средствам при использовании разработанного продукта:

1) Сервер:

- 64-разрядный (x64) процессор с тактовой частотой 3.2 ГГц или выше;
- 8 ГБ DDR4 (для 64-разрядной системы) оперативной памяти (ОЗУ);
- 1 ТБ SATA HDD;
- CentOS 8;
- Gigabit Ethernet и совместимая с ним сетевая карта;

Или же

1) Компьютер, оснащённый:

- 64-разрядный (x64) процессор с тактовой частотой 1 гигагерц (ГГц) или выше;
- 8 ГБ (для 64-разрядной системы) оперативной памяти (ОЗУ);
- 8 ГБ (для 64-разрядной системы) свободного пространства на жестком диске;
- ОС Windows 10 или выше; Linux Bugie, Cinnamon; MacOS Lion или выше
- Интернет-браузер Safari, Google Chrome
- Сетевая карта

2) Дополнительно:

- Экран
- Видеокарта
- Мышь
- Клавиатура

Минимальные требования к техническим средствам для развертывания и обеспечения доступа к разработанному продукту:

1) Сервер:

- 64-разрядный (x64) процессор с тактовой частотой 3.2 ГГц или выше;
- 8 ГБ DDR4 (для 64-разрядной системы) оперативной памяти (ОЗУ);
- 1 ТБ SATA HDD;
- CentOS 8;
- Gigabit Ethernet и совместимая с ним сетевая карта;

Для соблюдения минимальных требований рекомендуется сервер HP ProLiant MicroServer Gen10 Plus.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

3. Характеристика программы

3.1. Процесс развертки

Программа запускается с помощью Docker compose, предварительно используя собранные Docker образы. Docker образы собираются по через Dockerfile, сборка использует multistage build принцип, чтобы в Docker образе не было лишних данных. При своем запуске программа читает файлы конфигураций.

Docker compose содержит в себе настройку запуска самой программы ее зависимостей и другой инфраструктуры (MongoDB с ее репликами, а также сервис сообщений Kafka). На примере docker compose файла в приложении 4, мы видим процесс развертки сервиса курсов и его зависимостей. Засчет средства контроля правильности сборки в виде функционала healthcheck у docker, перед программой сервиса курсов запуститься только при корректном состоянии необходимых ей зависимостей.

Multistage build Dockerfile сервиса курсов изображен на рис.1. В этапе build мы собираем нашу программу, что требует наличия всех программных зависимостей и библиотек. Затем на этапе production переносятся лишь .exe (бинарный файл) программы и только необходимые файлы конфигураций в новый легкий контейнер, который уже не содержит кода ненужных библиотек и лишних файлов.

Рис. 1 — Multistage build. Courses Dockerfile

```

courses.Dockerfile ✘
1  ► FROM golang:1.21.4 as ↵ build
2    WORKDIR /app
3
4    COPY . .
5
6    ENV CGO_ENABLED=0
7    ENV GOOS=linux
8
9    RUN go build -o courses-svc ./services/courses/cmd
10
11   FROM alpine:latest as ↵ production
12
13   COPY --from=build /app/driver-svc ./
14
15   COPY --from=build /app/.env ./
16   COPY --from=build /app/services/courses/migrations ./migrations
17   COPY --from=build /app/services/courses/config/config.docker.yml ./config/config.local.yml
18
19   CMD ["./courses-svc"]
20
21   EXPOSE 8080

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

4. Обращение к программе

4.1. Запуск и завершение программы

Запуск программы осуществляется командой "make up"

Запуск сервисов мониторинга для всех программ осуществляется командой "make up-obs" из корневой папки директории с программой. Данные сервисы для всех программ нужно запустить лишь 1 раз.

4.2. Файлы конфигураций

На рисунке 2 показан пример файла конфигураций программы сервиса курсов.

Опишем не тривиальные к изменению параметры. Параметр wait_timeout структуры graceful_shutdown отвечает за время, при котором программа пытается себя завершить без потери данных с успешным закрытием ресурсов, по истечении данного времени программа завершается принудительно. Параметр level структуры logger отвечает за уровень вывода логов, а параметр env отвечает за работу логгера в режиме разработки или реального использования.

Рис. 2 – Configuration setup. Courses configuration file

```

apiVersion: "1.0.0"
app:
  name: "courses"
  env: "local"
graceful_shutdown:
  delay: "3s"
  wait_timeout: "15s"
  callback_timeout: "5s"
http:
  host: "0.0.0.0"
  port: 8080
  read_timeout: "15s"
  write_timeout: "15s"
mongo:
  database: courses
  uri: "mongodb://mongo1:27017,mongo2:27018,mongo3:27019/courses?replicaSet=rs0&authSource=admin&readPreference=primaryPreferred"
migrations_mongo:
  uri: "mongodb://mongo1:27017,mongo2:27018,mongo3:27019/courses?replicaSet=rs0&authSource=admin&readPreference=primaryPreferred"
  path: "migrations"
  enabled: true
logger:
  env: "dev" # "prod"
  # level: "debug"
  level: "info"
  outputs:
    - "stdout"
  error_outputs:
    - "stderr"
  encoding: "json"
  sentry_level: "error"
  sentry_dsn: ""

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

4.3. Описание использования API

Другие программы не обращаются к данной напрямую, взаимодействие происходит на уровне сети посредством протокола HTTP. Более подробно про взаимодействие в пункте 5.1

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

5. Входные и выходные данные

5.1. Описание организации и использования API

5.1.1. Общее описание организации API

Данная программа принимает запросы от других подсистем посредством HTTP (2.0). Содержание запросов может быть любым, однако при несоблюдении спецификации или других нарушениях программа ответит ошибкой (http error response).

Основные виды ошибок

- 1) 400 Bad Request (Плохой запрос): Сервер не может обработать запрос из-за недопустимого синтаксиса.
- 2) 401 Unauthorized (Неавторизован): Для доступа к ресурсу требуется аутентификация пользователя, но она не была предоставлена или была предоставлена неправильно.
- 3) 403 Forbidden (Доступ запрещен): Сервер понял запрос, но отказывает в выполнении из-за ограничений на стороне пользователя. Это может быть из-за отсутствия прав доступа к ресурсу или из-за ограничений IP-адреса.
- 4) 404 Not Found (Не найдено): Ресурс, который запрашивается, не найден на сервере.

Если ошибка проходит не по вине пользователя (а по вине системы) программа возвращает 500 Internal Server Error (Внутренняя ошибка сервера).

5.2. Описание использования спецификации API

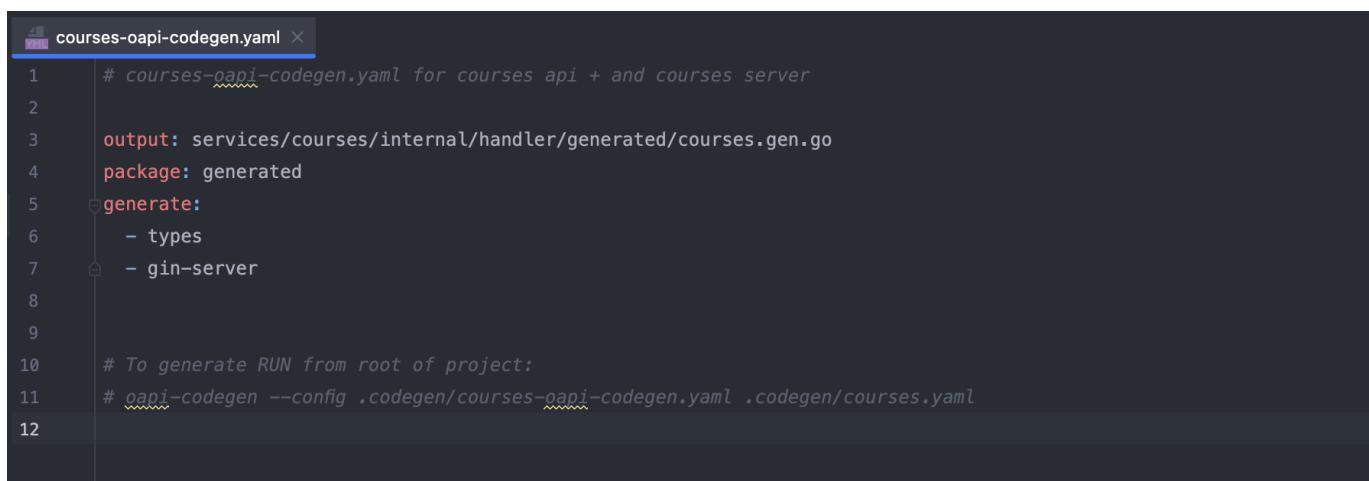
Для данного сервиса будет определена спецификации API по стандарту OpenAPI v3. Данная спецификация пишется в файле типа yaml. Текст актуальной на настоящий момент спецификации (или же ее расположение в сети Интернет) можно найти в настоящем Тексте программы данной подсистемы. Стандарт OpenAPI поддерживается многими программами и позволяет описывать JSON объекты входных данных с их типами и ограничениями (например, на размер текста), а также выходных. Спецификация также описывает версию и конечные точки API с комментариями к ними и адресами обращения. К каждому запросу в спецификации можно также зафиксировать его параметры и заголовки.

Данную спецификацию можно импортировать и использовать в Postman как Postman коллекцию (для работы с приложением).

Также для многих языков программирования существуют компиляторы, создающие код классов моделей запросов и ответов, а также классы контролера включающие в себя проверку поступающих в подсистему запросов. Пример конфигурации для генерации класса контролера (хэндлера) и моделей, используемых на входе и выходе в программу приведен на рисунке. В конце рисунка также написан код запуска компилятора (кодогенерации).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Рис. 3 — Configuration for OpenAPI codegeneration in Go. oapi-codegen configuration file



```
courses-oapi-codegen.yaml ×
1  # courses-oapi-codegen.yaml for courses api + and courses server
2
3  output: services/courses/internal/handler/generated/courses.gen.go
4  package: generated
5  generate:
6    - types
7    - gin-server
8
9
10 # To generate RUN from root of project:
11 # oapi-codegen --config .codegen/courses-oapi-codegen.yaml .codegen/courses.yaml
12
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

6. Сообщения

В разделе «Сообщения» должны быть указаны тексты сообщений, выдаваемых программисту или оператору в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

6.1. Описание работы со средствами мониторинга

Во время работы программа выдает сообщения - логи.

Тексты сообщений можно увидеть в консоли выполняемой программы, выбор консоли можно совершить в файле конфигурации. Данные сообщения содержат в себе информации о уровне лога, сервисе из которого она возникла, а также самого сообщения. При настройке (в файле конфигурации) логгера на работу в "env: production" (для использования на сервере), и уровня вывода сообщений "error" все сообщения обязательно читать специалистом. При наличие данных сообщений следует обратиться к разработчику. Данные сообщения об ошибках также поддерживают возможность рассылки на почту, однако данное решение платное и касается сервисов внешней программы (Sentry[15]).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

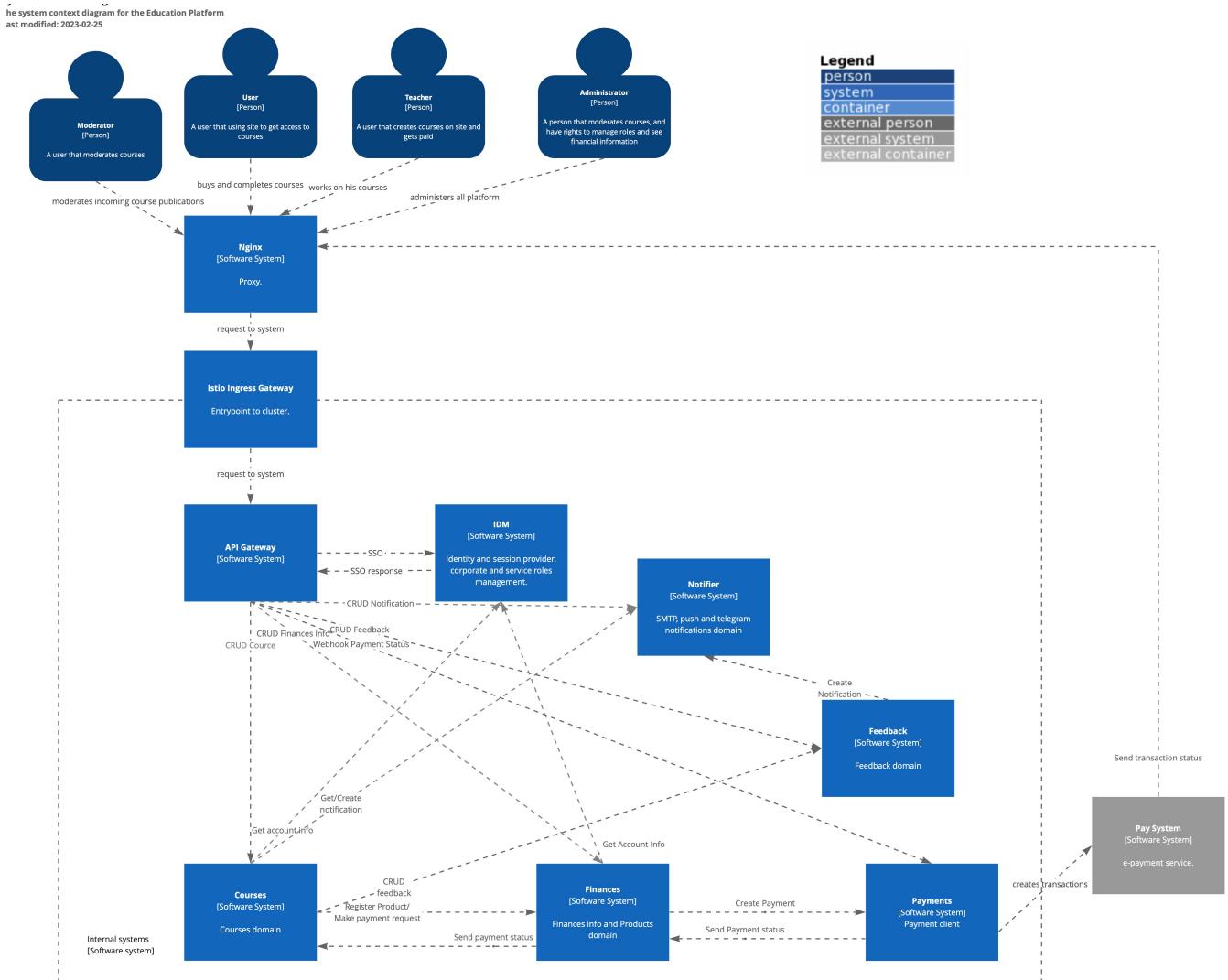
7. Список используемой литературы

- 1) ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 2) ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 3) ГОСТ 19.103-77 Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 4) ГОСТ 19.104-78 Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 5) ГОСТ 19.105-78 Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 7) ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 8) ГОСТ 19.603-78 Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 9) ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 10) ГОСТ 19.404-79 Программа и методика испытаний. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 11) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 12) ГОСТ 19.505-79 Руководство оператора. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 13) ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 14) ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 15) Sentry. Оповещения об ошибках [Электронный ресурс]. — <https://sentry.io/welcome/>

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Приложение 1. Уровень контекста системы Edu Platform

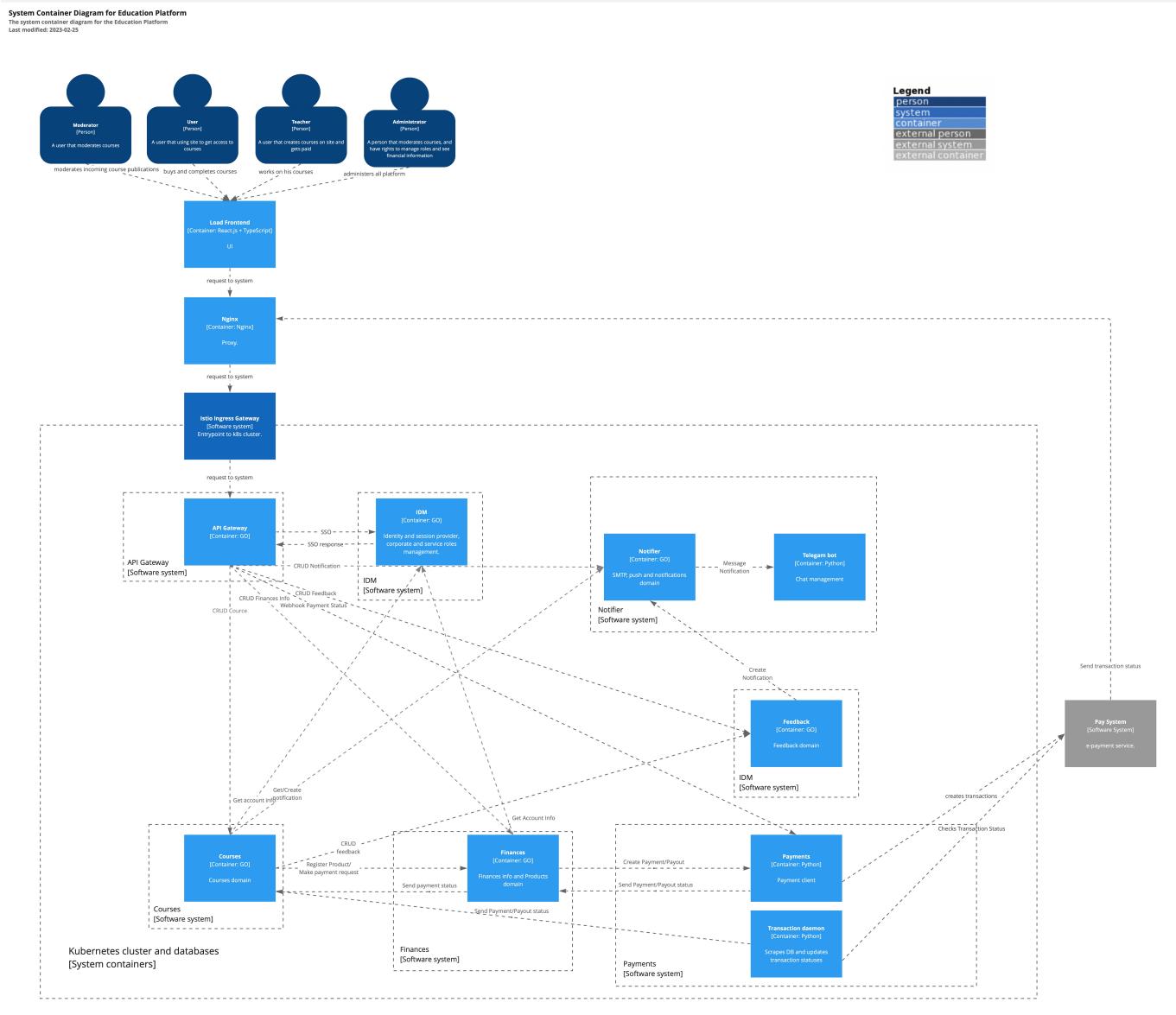
Рис. 4 – Context level diagram. Edu Platform c4 model



Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Приложение 2. Уровень контейнера системы Edu Platform

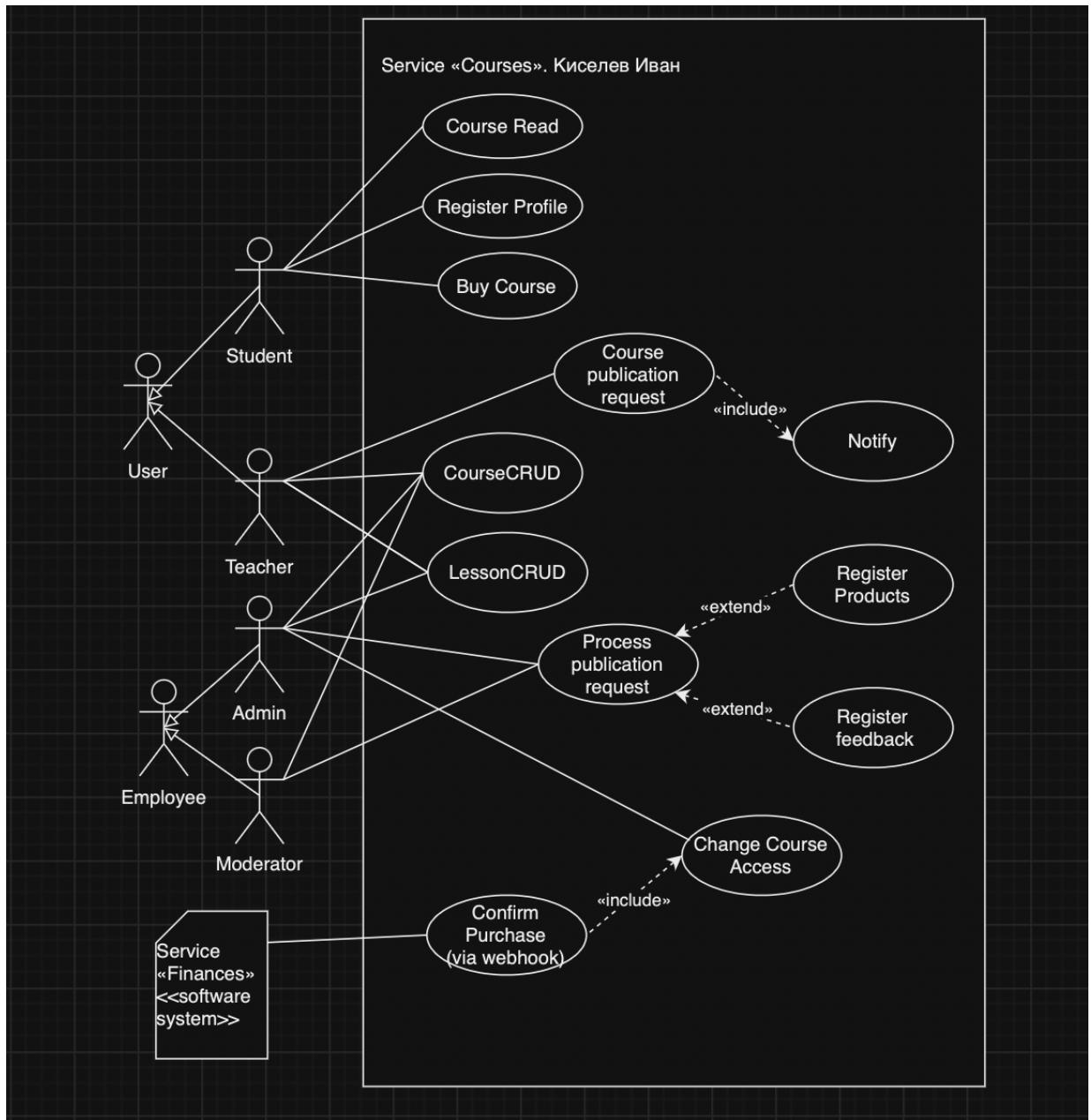
Рис. 5 — Container level diagram. Edu Platform c4 model



Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Приложение 3. Модель прецендентов подсистемы Courses

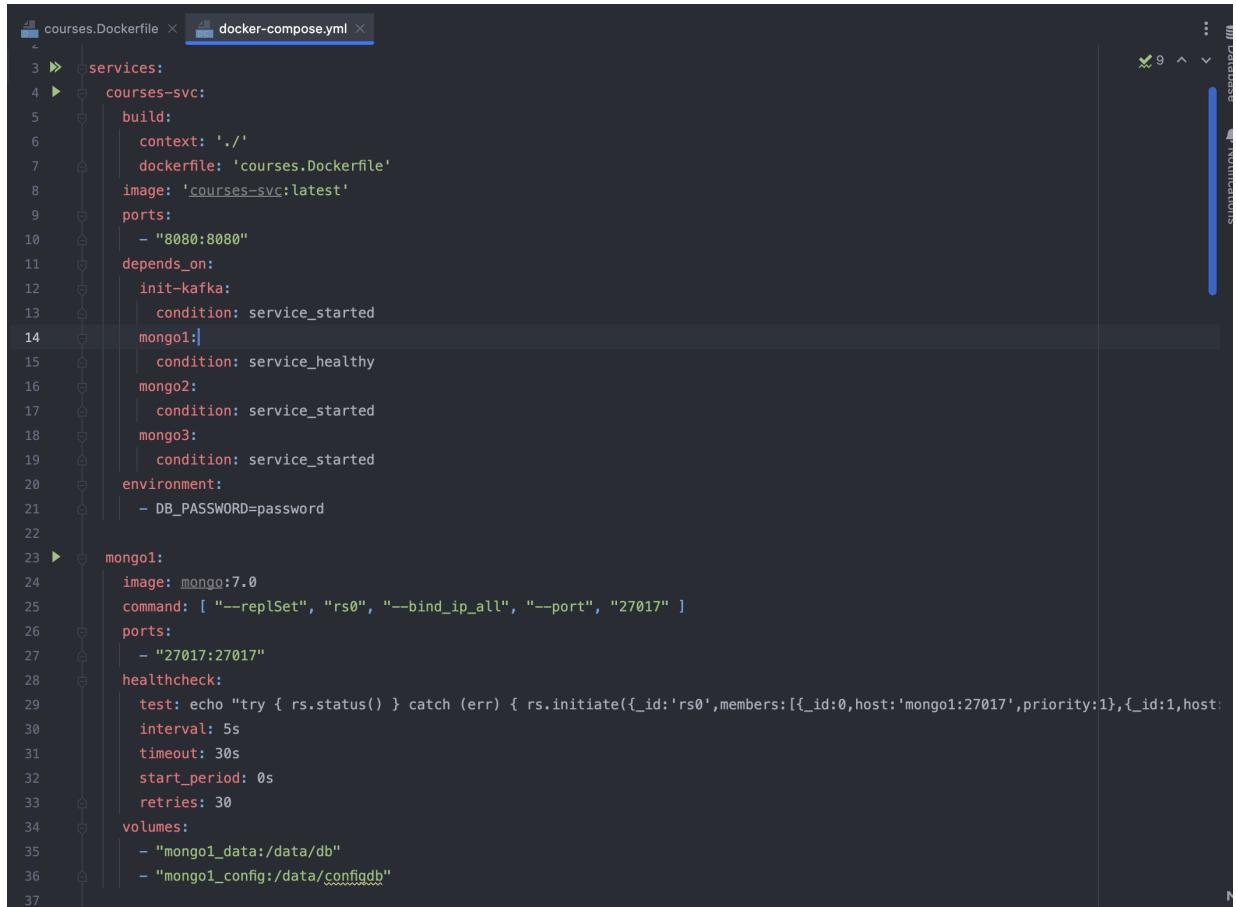
Рис. 6 — Usecase diagram. Courses software system



Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Приложение 4. Пример файла развертки сервиса «Courses»

Рис. 7 — Course microservice and MongoDB. Courses docker compose file



```
courses.Dockerfile x docker-compose.yml x
 3 ► services:
 4 ►   courses-svc:
 5   build:
 6     context: './'
 7     dockerfile: 'courses.Dockerfile'
 8     image: 'courses-svc:latest'
 9     ports:
10       - "8080:8080"
11     depends_on:
12       init-kafka:
13         condition: service_started
14       mongo1:
15         condition: service_healthy
16       mongo2:
17         condition: service_started
18       mongo3:
19         condition: service_started
20     environment:
21       - DB_PASSWORD=password
22
23 ►   mongol:
24   image: mongo:7.0
25   command: [ "--replSet", "rs0", "--bind_ip_all", "--port", "27017" ]
26   ports:
27     - "27017:27017"
28   healthcheck:
29     test: echo "try { rs.status() } catch (err) { rs.initiate({_id:'rs0',members:[{_id:0,host:'mongo1:27017',priority:1},{_id:1,host:'mongo2:27017'}, {_id:2,host:'mongo3:27017'}],arbiterCount:0}) }"
30     interval: 5s
31     timeout: 30s
32     start_period: 0s
33     retries: 30
34   volumes:
35     - "mongo1_data:/data/db"
36     - "mongo1_config:/data/configdb"
37
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Рис. 8 — MongoDB replicas. Courses docker compose file

```

37
38 ► mongo2:
39   container_name: mongo2
40   image: mongo:7.0
41   command: [ "--replSet", "rs0", "--bind_ip_all", "--port", "27018" ]
42   ports:
43     - "27018:27018"
44   volumes:
45     - "mongo2_data:/data/db"
46     - "mongo2_config:/data/configdb"
47
48 ► mongo3:
49   container_name: mongo3
50   image: mongo:7.0
51   command: [ "--replSet", "rs0", "--bind_ip_all", "--port", "27019" ]
52   ports:
53     - "27019:27019"
54   volumes:
55     - "mongo3_data:/data/db"
56     - "mongo3_config:/data/configdb"
57

```

Рис. 9 — Kafka. Courses docker compose file

```

91 ► zookeeper:
92   image: wurstmeister/zookeeper:latest
93   ports:
94     - "2181:2181"
95   environment:
96     ZOOKEEPER_CLIENT_PORT: 2181
97 ► kafka:
98   depends_on:
99     - zookeeper
100   container_name: kafka
101   image: wurstmeister/kafka:latest
102   ports:
103     - "9094:9094"
104   environment:
105     KAFKA_BROKER_ID: 1
106     KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
107     KAFKA_LISTENERS: INTERNAL://0.0.0.0:9092,OUTSIDE://0.0.0.0:9094
108     KAFKA_ADVERTISED_LISTENERS: INTERNAL://kafka:9092,OUTSIDE://localhost:9094
109     KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: INTERNAL:PLAINTEXT,OUTSIDE:PLAINTEXT
110     KAFKA_INTER_BROKER_LISTENER_NAME: INTERNAL
111 ► init-kafka:
112   image: confluentinc/cp-kafka:latest
113   depends_on:
114     - kafka
115   entrypoint: [ '/bin/sh', '-c' ]
116   command:
117     "
118       kafka-topics.sh --bootstrap-server kafka:9092 --list
119
120       echo -e 'Creating kafka topics'
121       kafka-topics.sh --bootstrap-server kafka:9092 --create --if-not-exists --topic outbound --replication-factor 1 --partitions 1
122       kafka-topics.sh --bootstrap-server kafka:9092 --create --if-not-exists --topic inbound --replication-factor 1 --partitions 1
123
124       echo -e 'Successfully created the following topics:'
125       kafka-topics.sh --bootstrap-server kafka:9092 --list
126     "

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

8. Глоссарий

- 1) HTTP - протокол для передачи данных
- 2) Микросервис - программная подсистема варианта сервис-ориентированной архитектуры программного обеспечения, направленный на взаимодействие насколько это возможно небольших, слабо связанных и легко изменяемых модулей
- 3) Docker – Платформа для контейнеризации приложений, которая позволяет упаковывать приложения и их зависимости в легко переносимые контейнеры. Docker обеспечивает изоляцию ресурсов и среды выполнения.
- 4) Docker Compose – Инструмент для определения и управления многоконтейнерными приложениями с помощью файла конфигурации в формате ".yaml". Docker Compose позволяет определить сервисы, их зависимости и настройки в одном файле, что упрощает развертывание и управление многоконтейнерными приложениями.
- 5) Multistage Build (Многоэтапная сборка) - методология сборки программного обеспечения, которая включает несколько этапов, где каждый использует определенные инструменты и зависимости для выполнения его задач. Этот подход позволяет уменьшить размер конечного образа или бинарного файла, улучшить скорость сборки и обеспечить более чистое и эффективное управление зависимостями и окружением исполняемого файла.
- 6) API (англ. Application Programming Interface — программный интерфейс приложения) — это набор способов и правил, по которым различные программы общаются между собой и обмениваются данными.
- 7) JSON — текстовый формат обмена данными.
- 8) Go — компилируемый многопоточный язык программирования.
- 9) MongoDB — документоориентированная система управления базами данных.
- 10) Postman — это HTTP-клиент для тестирования API.
- 11) C4 Model - методология архитектуры ПО, которая позволяет описывать систему на различных уровнях абстракции (контекст, контейнеры, компоненты и код)
- 12) База данных – совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.
- 13) СУБД (Система управления базами данных) - программное средство для использования баз данных.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.09.12-01 33 01-1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Лист регистрации изменений