Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

Работа выполнена в СПБ «DeCode»

СОГЛАСОВАНО

Начальник отдела ОНиПКРС

_____ Е.М. Димитриади
(подпись)

«03» 06 2024 г.

Декан _____

_____ И.А. Трещев
(подпись)

«03» 06 2024 г.

УТВЕРЖДАЮ

Проректор по научной работе

_____ А.В. Космынин
(подпись)

«04» 06 2024 г.

«Конструктор тренингов»

Комплект *конструкторской / проектной* документации

Руководитель СПБ _____ 03.06.24 *Е.Б. Абарникова*
(подпись, дата)

Руководитель проекта _____ 03.06.24 *Е.Э. Шаповалов*
(подпись, дата)

Комсомольск-на-Амуре 2024

Карточка проекта

Название	*«Конструктор тренингов»*
Тип проекта	Тип проекта: техническое творчество (инициативный)
Исполнители	Студент Т.С. Сидорин – 2ИТб-1 Студент К.М. Григорьев –2ИТб-1 Студент К.М. Черненко – 2ИТб-1 Студент И.А. Франтов – 2ИТб-1 Студент Д.А. Филиппов – 2ИТб-2
Срок реализации	Февраль 2024 – май 2024

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

ЗАДАНИЕ
на разработку

Название проекта: Разработка веб-сервиса «Конструктор тренингов»_____

_____

Назначение: Назначение проекта по созданию веб-сервиса "Конструктор тренингов" заключается в предоставлении пользователям удобной и функциональной платформы для самостоятельного создания и проведения тренингов, семинаров, мастер-классов и других образовательных мероприятий. ____

Область использования: учебные заведения, преподаватели и студенты могут использовать сервис для создания и проведения образовательных мероприятий, таких как семинары, мастер-классы, конференции_____

Функциональное описание проекта: _____

- Регистрация и авторизация пользователей: предоставление возможности зарегистрироваться и войти в свой личный кабинет на сервисе._____

- Создание и редактирование тренингов_____

Техническое описание устройства: _____

_____

_____

_____

_____

_____

Требования: _____

Функциональные требования: _____

- авторизация;_____

- регистрация;_____

- Создание и редактирование тренинга;

-Удаление тренинга и обновление информации о тренинге;

Нефункциональные требования:

- Надежность: сервис должен быть надежным и обеспечивать бесперебойную работу, минимизируя риски сбоев и потерь данных.

Безопасность: сервис должен обеспечивать безопасность данных пользователей, включая персональную информацию, контент тренингов и отзывы. Для этого необходимо использовать современные методы защиты данных, такие как шифрование, авторизация и аутентификация.

Масштабируемость: сервис должен быть масштабируемым и способным поддерживать растущее количество пользователей и тренингов, а также обеспечивать высокую производительность при высокой нагрузке;

- сервис должен быть доступен на разных устройствах (ПК, планшеты, мобильные устройства);

- сервис должен соответствовать брендбуку университета

План работ:

Наименование работ	Срок
Исследование требований	02.02-03.02
Создание прототипа	04.02-04.03
Разработка клиентской части	05.03-05.04
Разработка серверной части	06.04-06.05
Тестирование и отладка	07.05-10.05
Публикация	11.05

Комментарии:
Разработка проекта велась с использованием agile-методологий, у каждого члена команды была своя задача, благодаря этому выполнение некоторых этапов велось параллельно

Перечень графического материала:

1. Принципиальная схема;
2. Чертежи изделия (или трехмерные модели изделия);
3. Внешний вид изделия;
4. Блок-схема алгоритмов (при наличии управляющих программ);

Руководитель проекта 03.06.24 *Е.Э. Шаповалов*

(подпись, дата)

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

ПАСПОРТ

Разработка веб-сервиса «Конструктор тренингов»

Руководитель проекта 03.06.24 *Е.Э. Шаповалов*

(подпись, дата)

Комсомольск-на-Амуре 2024

Содержание

1 Общие положения

Настоящий паспорт является документом, предназначенным для ознакомления с основными техническими характеристиками, устройством, правилами установки и эксплуатации проекта «Карта выпускника» (далее «сервис»).

1.1 Наименование изделия

Полное наименование изделия – «Конструктор тренингов».

1.2 Наименования документов, на основании которых ведется проектирование изделия

Проектирвание «Конструктор тренингов» осуществляется на основании требований и положений следующих документов:

- задание на разработку.

1.3 Перечень организаций, участвующих в разработке изделия

Заказчиком проекта «Конструктор тренингов» является Федеральное государственное бюджетное образовательное учреждение высшего образования «Комсомольский-на-Амуре государственный университет» (далее заказчик), находящийся по адресу: 681013, Хабаровский край, г. Комсомольск-на-Амуре, Ленина пр-кт., д. 17.

Исполнителями проекта «Карты выпускника» являются Конструкторы студенческого конструкторского студенческого проектного бюро «DeCode» (далее СПБ «DeCode»), студенты группы Т.С. Сидорин – 2ИТб-1, Студент К.М. Григорьев –2ИТб-1, К.М. Черненко – 2ИТб-1, И.А. Франтов – 2ИТб-1 Д.А. Филиппов – 2ИТб-2.

.

1.4 Сведения об использованных при проектировании нормативно-технических документах

При проектировании использованы следующие нормативно-технические документы:

ГОСТ 2.001-2013. Единая система конструкторской документации. Общие положения.

ГОСТ 2.102-2013. Единая система конструкторской документации. Виды и комплектность конструкторских документов.

ГОСТ 2.105-95. Единая система конструкторской документации. Общие требования к текстовым документам.

ГОСТ 2.610-2006. Единая система конструкторской документации. Правила выполнения эксплуатационных документов.

ГОСТ 2.004-88. Единая система конструкторской документации. Общие требования к выполнению конструкторских технологических документов на печатающих и графических устройствах вывода ЭВМ.

ГОСТ 2.051-2006. Единая система конструкторской документации. Электронные документы. Общие положения.

ГОСТ 2.052-2006. Единая система конструкторской документации. Электронная модель изделия. Общие положения.

ГОСТ 2.601-2013. Единая система конструкторской документации. Эксплуатационные документы.

2 Техническое задание

2.1 Наименование проекта

Разработка веб-сервиса «Конструктор тренингов».

2.2 Цель проекта

Целью проекта по созданию веб-сервиса "Конструктор тренингов" является предоставление пользователям удобной и функциональной платформы для самостоятельного создания, продвижения и проведения тренингов, семинаров, мастер-классов и других образовательных мероприятий.

2.3 Требования к функциональности

Сервис должен содержать следующую информацию : фио, год окончания, какой университет, какой факультет закончил, в какой группе обучался, где работает, в каком городе работает, должность и почту о выпускниках университета, которые на нем отмечены, а также предоставлять данные при нажатии на соответсвующий маркер на карте.

Цветовая гамма и шрифт текста должны соответсвовать фирменному стилю Университета.

2.4 Требования к дизайну

Дизайн сервиса должен быть современным и привлекательным для целевой аудитории. Для этого необходимо использовать сочетание цветовых решений, текстовых блоков и графических элементов.

Также цветовая гамма должна соответствовать фирменному стилю Университета, а использование шрифтов должно соответствовать корпоративному стилю.

2.5 Технические требования

Сервис должен быть разработан с использованием современного стека технологий Django. Должна быть поддержка с другими системами. Для удобства пользователей сервис должен поддерживать отображение на мобильных устройствах и иметь адаптивный дизайн.

2.6 Срок выполнения проекта

Проект должен быть завершен 11.05.2024.

2.7 Требования к аппаратному обеспечению

Для просмотра сервиса необходимо использовать персональный компьютер или мобильное устройство с доступом в Интернет и следующими характеристиками:

- процессор Intel Core i3 или аналогичный;
- оперативная память не менее 4 Гб;
- разрешение экрана не менее 1280 x 720 пикселей.

2.8 Требования к программному обеспечению

Для просмотра сервиса необходимо использовать веб-браузер, поддерживающий HTML3 и CSS3. Рекомендуется использовать один из следующих браузеров:

- Google Chrome версии 80 или выше;
- Mozilla Firefox версии 75 или выше;
- Apple Safari версии 13 или выше;
- Microsoft Edge версии 80 или выше.

3 Руководство разработчика

В качестве инструмента разработки использовался Django, предоставляющий набор инструментов и библиотек для разработки веб-приложений, включая системы управления базами данных, маршрутизацию URL, обработку запросов и ответов HTTP, а также шаблоны HTML. Данный фреймворк основан на принципах DRY (Don't Repeat Yourself) и MVC (Model-View-Controller), что позволяет разработчикам создавать гибкие и масштабируемые веб-приложения..

Основные блоки кода показаны в листингах 3.1 – 3.3

Листинг 3.1 – Модуль отображения views.py

```
from django.shortcuts import render, get_object_or_404
from django.shortcuts import redirect
from .forms import TrainingEditForm
from django.contrib import messages
from django.urls import reverse
from django.contrib.auth import get_user_model
from django.http import JsonResponse
from .models import Coordinates
# Create your views here.
from django.http import HttpResponseRedirect, HttpResponseNot-
Found
from trainings.models import Training
from django.http import JsonResponse
import json
import  datetime
from django.views.decorators.csrf import csrf_exempt
from django.contrib.auth.models import User

def mytrainings(request):
    author = request.user.id
    trainings = Training.objects.filter(author=author)

    return render(request, "mytrainings.html", {"trainings":
trainings})


# сохранение данных в бд
def create(request):
    if request.method == "POST":
        training = Training()
        training.title = request.POST.get("title")
        training.describe = request.POST.get("describe")
```

```
        training.author = request.user.id
        training.logo = request.FILES.get("logo")
        training.save()
    return HttpResponseRedirect("/")


# изменение данных в бд
def edit(request, id):
    try:
        training = Training.objects.get(id=id)

        if request.method == "POST":
            training.title = request.POST.get("title")
            training.describe = request.POST.get("describe")
            training.author = request.POST.get("user")
            training.logo = request.FILES.get("logo")
            training.save()
            return HttpResponseRedirect("/")
        else:
            return render(request, "edit.html", {"training":
training})
    except Training.DoesNotExist:
        return HttpResponseNotFound("<h2>Person not found</h2>")


# удаление данных из бд
def delete(request, id):
    try:

        training = Training.objects.get(id=id)
        training.delete()
        return HttpResponseRedirect("/")
    except Training.DoesNotExist:
        return HttpResponseNotFound("<h2>Training not
found</h2>")


def workspace(request, id):
    try:
        training = get_object_or_404(Training, id=id)
        return render(request, "workspace.html", {"training":
training})
    except Training.DoesNotExist:
        return HttpResponseNotFound("<h2>Training not
found</h2>")


def traningedit(request, id):
    training = get_object_or_404(Training, id=id)
    if request.method == 'POST':
        form = TrainingEditForm(request.POST, instance=training)
        if form.is_valid():
```

```
            form.save()
            messages.success(request, 'Training edited success-
fully.')
            # Перенаправление на страницу complete с id трени-
ровки
            return redirect(reverse('complete', kwargs={'id':
id}))
    else:
        form = TrainingEditForm(instance=training)  # Preload
existing title
    return render(request, "traningedit.html", {"form": form,
"training": training})


def complete(request, id):
    try:
        training = get_object_or_404(Training, id=id)
        return render(request, "complete.html", {"training":
training})
    except Training.DoesNotExist:
        return HttpResponseNotFound("<h2>Training not
found</h2>")

def pool(request, user_id):
    try:
        author = request.user.id
        User = get_user_model()
        user = User.objects.get(id=user_id)
        return render(request, "pool.html", {"user": user, "au-
thor": author})
    except User.DoesNotExist:
        return HttpResponseNotFound("<h2>User not found</h2>")


def save_coordinates(request):
    if request.method == 'POST':
        try:
            data = json.loads(request.body)
            object_id = data.get('object_id')
            top = data.get('top')
            left = data.get('left')
            width = data.get('width')
            height = data.get('height')

            # Check for missing or None values in coordinates
            if None in [top, left, width, height]:
                return JsonResponse({'error': 'Missing data'},
status=400)

            if object_id:
                # Update existing coordinates
                coordinates = Coordi-
```

```
nates.objects.get(id=object_id)
                coordinates.top = top
                coordinates.left = left
                coordinates.width = width
                coordinates.height = height
                coordinates.save()
            else:
                # Create new coordinates
                coordinates = Coordi-
nates.objects.create(top=top, left=left, width=width,
height=height)

            # Return object_id in the response
            return JsonResponse({'success': 'Coordinates saved
or updated', 'object_id': coordinates.id})

        except Coordinates.DoesNotExist:
            return JsonResponse({'error': 'Object not found'},
status=404)
        except IntegrityError as e:
            return JsonResponse({'error': str(e)}, status=400)

    else:
        # Handle non-POST requests
        return JsonResponse({'error': 'Invalid request method'},
status=405)
```

Листинг 3.2 – Код файла tranings.py

```
from django.contrib.auth import authenticate, login
from django.views import View
from django.shortcuts import render, redirect

from users.forms import UserCreationForm


class Register(View):
    template_name = 'registration/register.html'

    def get(self, request):
        context = {
            'form': UserCreationForm()
        }
        return render(request, self.template_name, context)

    def post(self, request):
        form = UserCreationForm(request.POST)

        if form.is_valid():
            form.save()
            username = form.cleaned_data.get('username')
```

```
            password = form.cleaned_data.get('password1')
            user    =    authenticate(username=username,    pass-
word=password)
            login(request, user)
            return redirect('home')
        context = {
            'form': form
        }
        return render(request, self.template_name, context)
```

Листинг 3.3 – Точка входа в программу

```
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys


def main():
    """Run administrative tasks."""
    os.environ.setdefault('DJANGO_SETTINGS_MODULE',    'Construc-
torTraining.settings')
    try:
        from       django.core.management       import       exe-
cute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed
and "
            "available on your PYTHONPATH environment variable?
Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)


if __name__ == '__main__':
    main()
```

4 Руководство пользователя

Основные окна приложения показаны на рисунках 4.1 – 4.5.

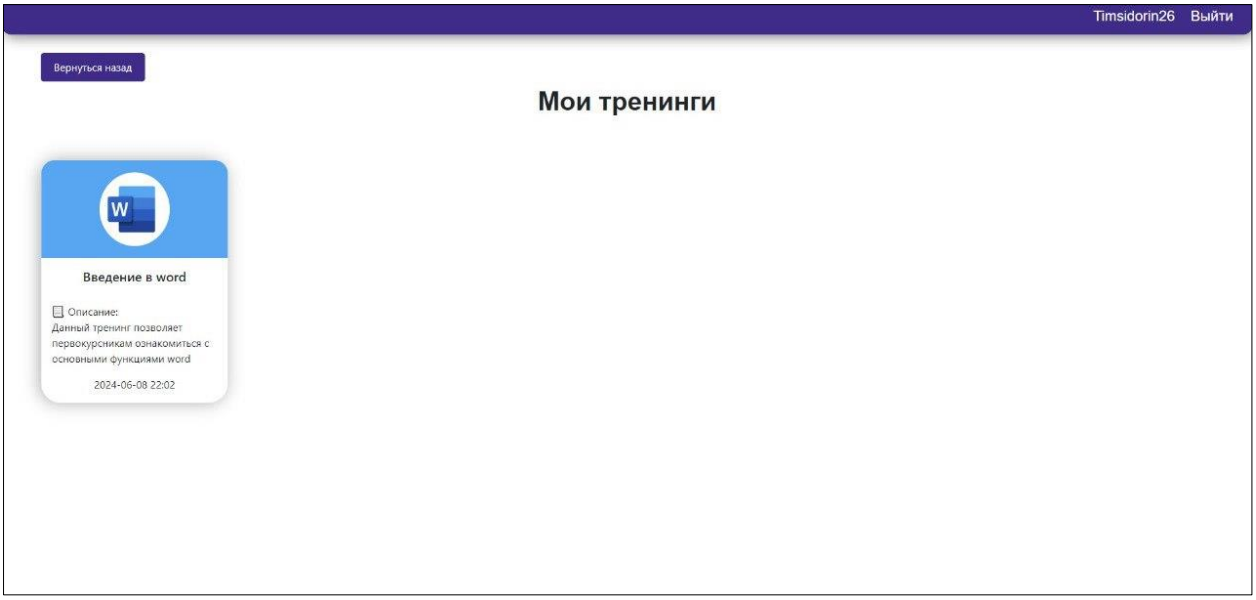

Рисунок 4.1 – Окно просмотра ранее созданных тренингов

Рисунок 4.2 – Главное окно

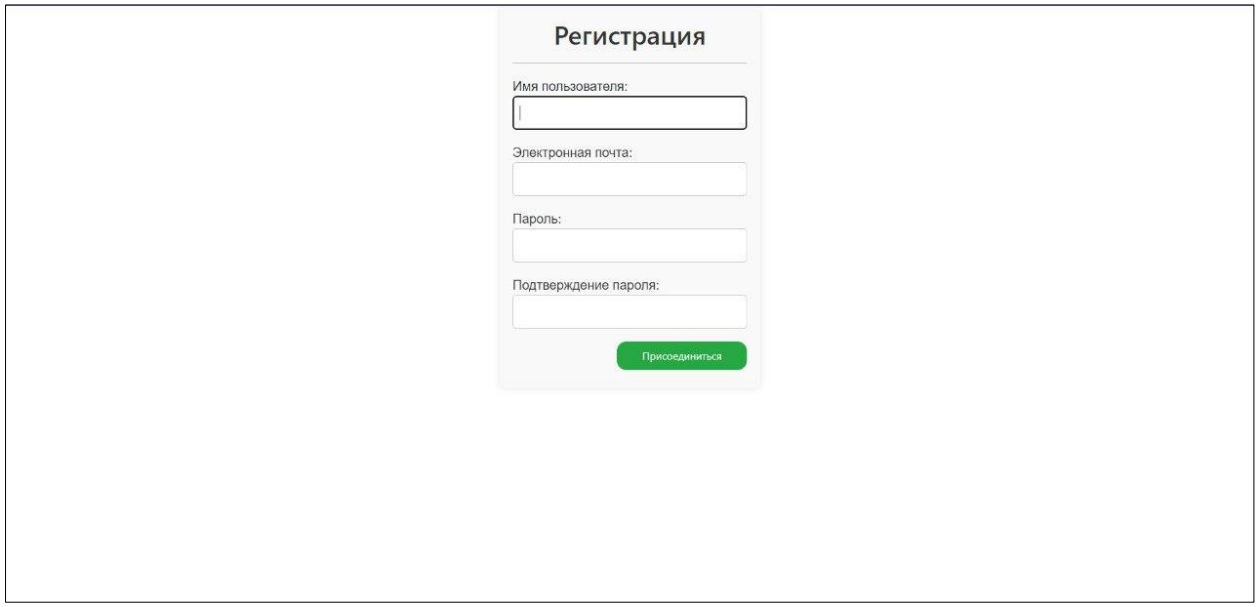

Рисунок 4.3 – Процесс регистрации на сервисе

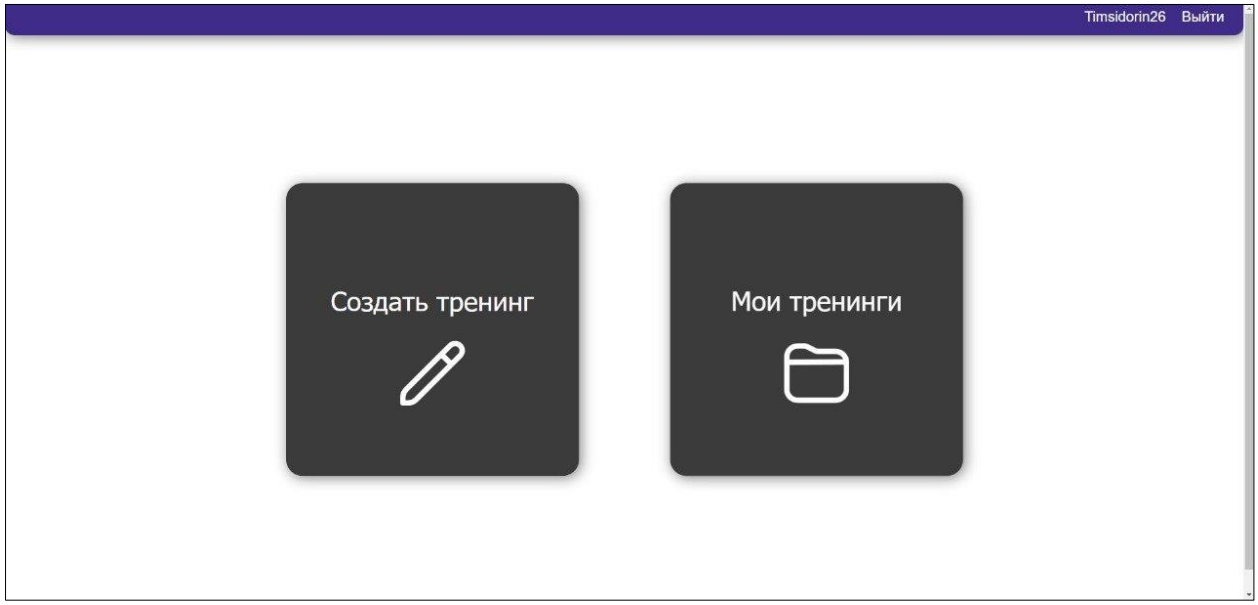

Рисунок 4.4 – Меню выбора действий

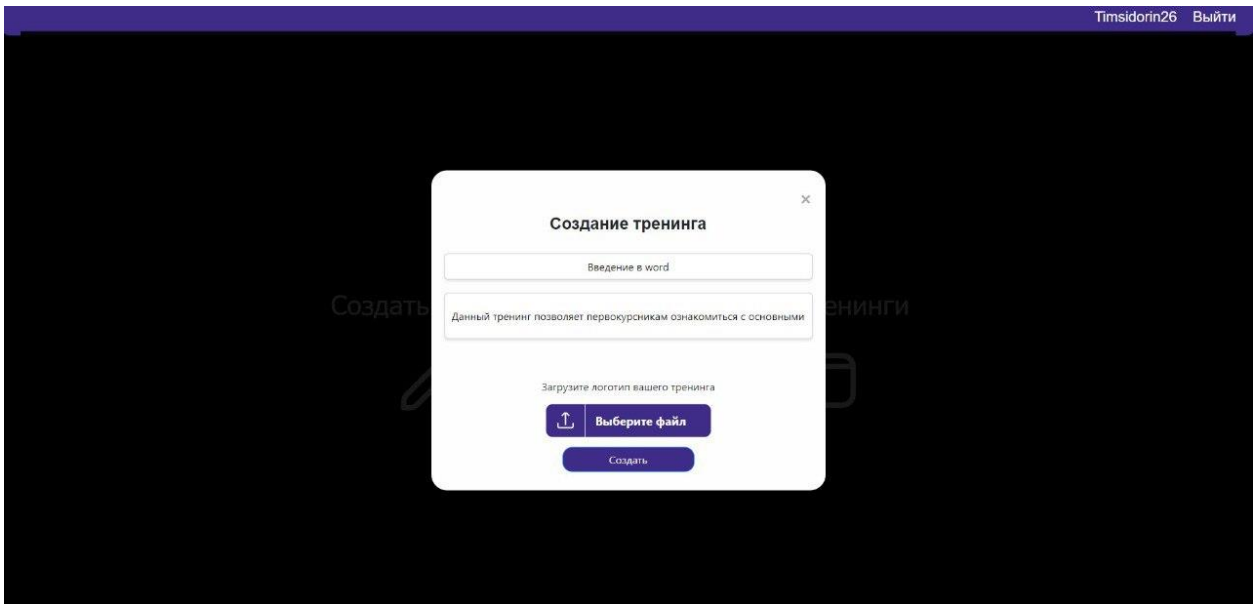

Рисунок 4.5 – Окно создания тренинга

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

СОГЛАСОВАНО

Начальник отдела ОНиПКРС
_____ Е.М. Димитриади
(подпись)
«03» 06 20__ г.

Декан _____
_____ *И.А. Трещев*
(подпись)

УТВЕРЖДАЮ

Проректор по научной работе
_____ А.В. Космынин
(подпись)
«04» 06 20 24 г.

АКТ

о приемке в эксплуатацию проекта
Разработка веб-сервиса «Конструктор тренингов»

г. Комсомольск-на-Амуре «04» 06 20 24 г.

Комиссия в составе представителей:

со стороны заказчика

- Е.Б. Абарникова – руководитель СПБ «DeCode»,
- И.А. Трещев – декана ФКТ

со стороны исполнителя

- Е.Э. Шаповалова – руководителя проекта,
- Т.С. Сидорин – 2ИТб-1
- К.М. Григорьев –2ИТб-1
- К.М. Черненко – 2ИТб-1
- И.А. Франтов – 2ИТб-1
- Д.А. Филиппов – 2ИТб-2

«Исполнитель» передает проект *Разработка веб-сервиса «Конструктор тренингов»*, в составе:

1 Руководство пользователя

2 Руководство программиста

Руководитель проекта 03.06.24 Е.Э. Шаповалов
(подпись, дата)

Исполнители проекта
(подпись, дата) Т.С. Сидорин
(подпись, дата) К.М. Григорьев
(подпись, дата) К.М. Черненко
И.А. Франтов
(подпись, дата) Д.А. Филиппов