МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеративное государственное бюджетное образовательное учреждение

высшего профессионального образования

**“УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”**

Кафедра «Измерительно-вычислительные комплексы»

**Отчёт**

к лабораторной работе №4

по дисциплине “Облачные вычисления и системы”

Выполнил:

студент группы ЦИСТбв-51

Нгуен Х. А.

Принял:

преподаватель

Сазонов С. Н.

Ульяновск

УлГТУ

2023

# Цель лабораторной работы

Создать веб-сайт ASP.NET Core Blazor и развернуть его на платформе Яндекс Облако.

# Выполнение работы

Для создания веб-сайта ASP.NET нам нужно установить пакет средства разработки ПО или коротко говоря SDK .NET на машину, где будет вестись разработка. В этом случае используется домашний компьютер. SDK доступна для основных операционных систем, таких как Windows, Mac OS и Linux. Ссылка на скачивание находится по URL адресу - <https://dotnet.microsoft.com/en-us/download/dotnet/7.0>.

После установки SDK создаём проект используя консольную команду dotnet. Чтобы его создать нужно ввести следующую команду

dotnet new blazorserver --name lab4

Результат введенной команды представлена на рисунке 1.

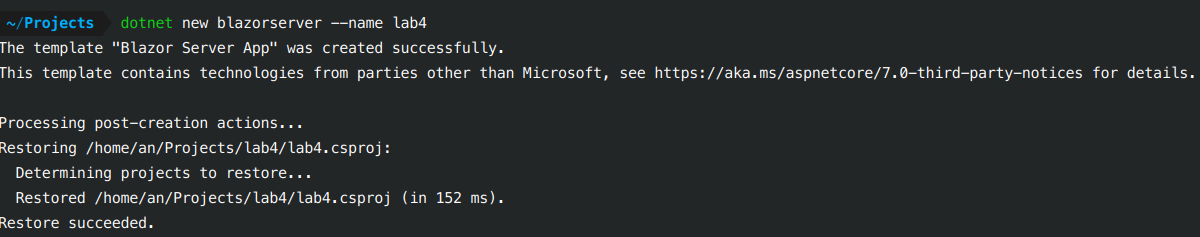


Рисунок 1

Далее открываем в среде разработки Visual Studio Code. На нём должны быть установлены расширения для работы с проектами на C#. Список расширении VS Code для работы с проектом:

* *C#* - Автор: Microsoft
* *C# Dev Kit* – Автор: Microsoft
* *Microsoft.AspNetCore.Razor.VSCode.BlazorWasmDebugging* – Автор: Microsoft

Далее нам нужно открыть файл *Pages/Index.razor* и заменить весь текст на следующий код

@page "/"

<PageTitle>Основная страница</PageTitle>

<h1>Основная страница сайта</h1>

<p>Сайт на ASP.NET Core Blazor</p>

<img src="images/something.jpeg" alt="Здесь должна быть картинка">

<p>Автор - студент ЦИСТбв-51 Нгуен Хыу Ан</p>

Потом нам нужно добавить изображение в папку *wwwroot/images*. Взятое изображение было сгенерировано на сайте <https://deepai.org/machine-learning-model/abstract-painting-generator>.

Также в *Program.cs* нам нужно добавить код, который задаёт порт прослушки через получение переменных окружения

…

var port = Environment.GetEnvironmentVariable("PORT");

app.Urls.Add($"http://\*:{port}");

…

Есть следующие способы развертки сайта на облачной платформе от Яндекса:

* на виртуальной машине без контейнеров (сервис Cloud Compute);
* на виртуальной машине с контейнерами Docker (сервис Cloud Compute);
* serverless (сервис Serverless Containers);
* serverless функции (Serverless Functions)

Для нашего веб-сайта выберем serverless решение, так как им не будут пользоваться постоянно и стоит дешевле, чем решение на базе виртуальной машины. Serverless решение имеет такую особенность, как холодный запуск. При обращении к сайту, облако начинает запускать контейнер из Docker образа, и ресурсы машины на котором работают контейнеры будут использованы только при обращении к сайту. Но в зависимости от размера приложения и мощностей хоста на котором работают контейнеры, контейнер будет загружаться не сразу. Сравнение сервисов можно посмотреть на странице документации <https://cloud.yandex.ru/docs/serverless-containers/service-comparison>. Для serverless решения нужен Docker-образ нашего приложения. Для этого нужно установить Docker и затем создать файл *Dockerfile* в корневой директории проекта. Создаём файл и добавляем следующий текст

FROM --platform=$BUILDPLATFORM mcr.microsoft.com/dotnet/sdk:7.0 AS build

WORKDIR /source

# copy csproj and restore as distinct layers

COPY \*.sln .

COPY ./\*.csproj .

RUN dotnet restore

# copy and publish app and libraries

COPY . .

RUN dotnet publish -c release -o /app --no-restore

# final stage/image

FROM mcr.microsoft.com/dotnet/aspnet:7.0

WORKDIR /app

COPY --from=build /app .

USER $APP\_UID

ENTRYPOINT ["dotnet", "lab4.dll"]

Экземпляры Dockerfile-а можно посмотреть в github репозитории <https://github.com/dotnet/dotnet-docker>, либо на странице документации Microsoft <https://learn.microsoft.com/en-us/aspnet/core/host-and-deploy/docker/building-net-docker-images?view=aspnetcore-7.0#build-and-deploy-manually>. Также нам нужно создать файл *.dockerignore*, содержащий список исключающих файлов и glob-шаблонов, который docker должен не использовать их при сборке образа. Следующий текст это файла:

# directories

\*\*/bin/

\*\*/obj/

\*\*/out/

# files

Dockerfile\*

\*\*/\*.md

Проверим правильность dockerfile-а. Для этого нам нужно зайти в директорию проекта через терминал и набрать команду для сборки образа приложения:

docker build -t blazor-server-an-nguen .

Результат работы команды представлена на рисунке 2:

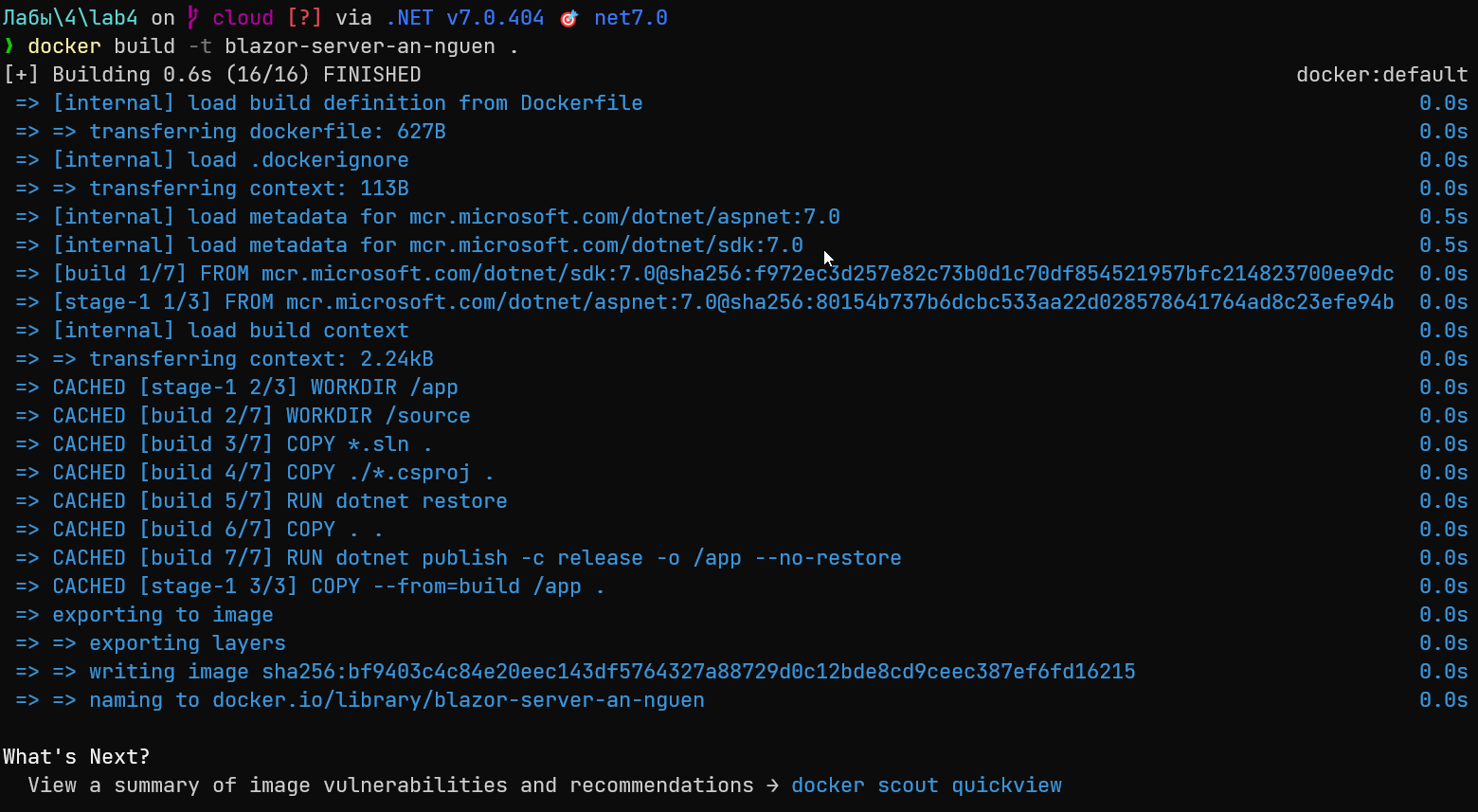


Рисунок 2

Сборка прошла успешно и Dockerfile является корректным.

В консоли облачной платформы, создаём каталог для этой лабораторной работы. Далее открывается диалоговое окно создания каталога. В поле *«Имя» вводим lab<номер лабораторной работы>-<ФИО>* (рис. 3).

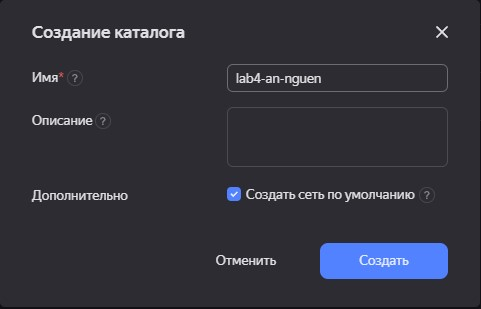


Рисунок 3 – Диалоговое окно создания каталога

Далее открываем страницу реестра контейнеров Container Registry в консоли Яндекс Облака (рис. 4).

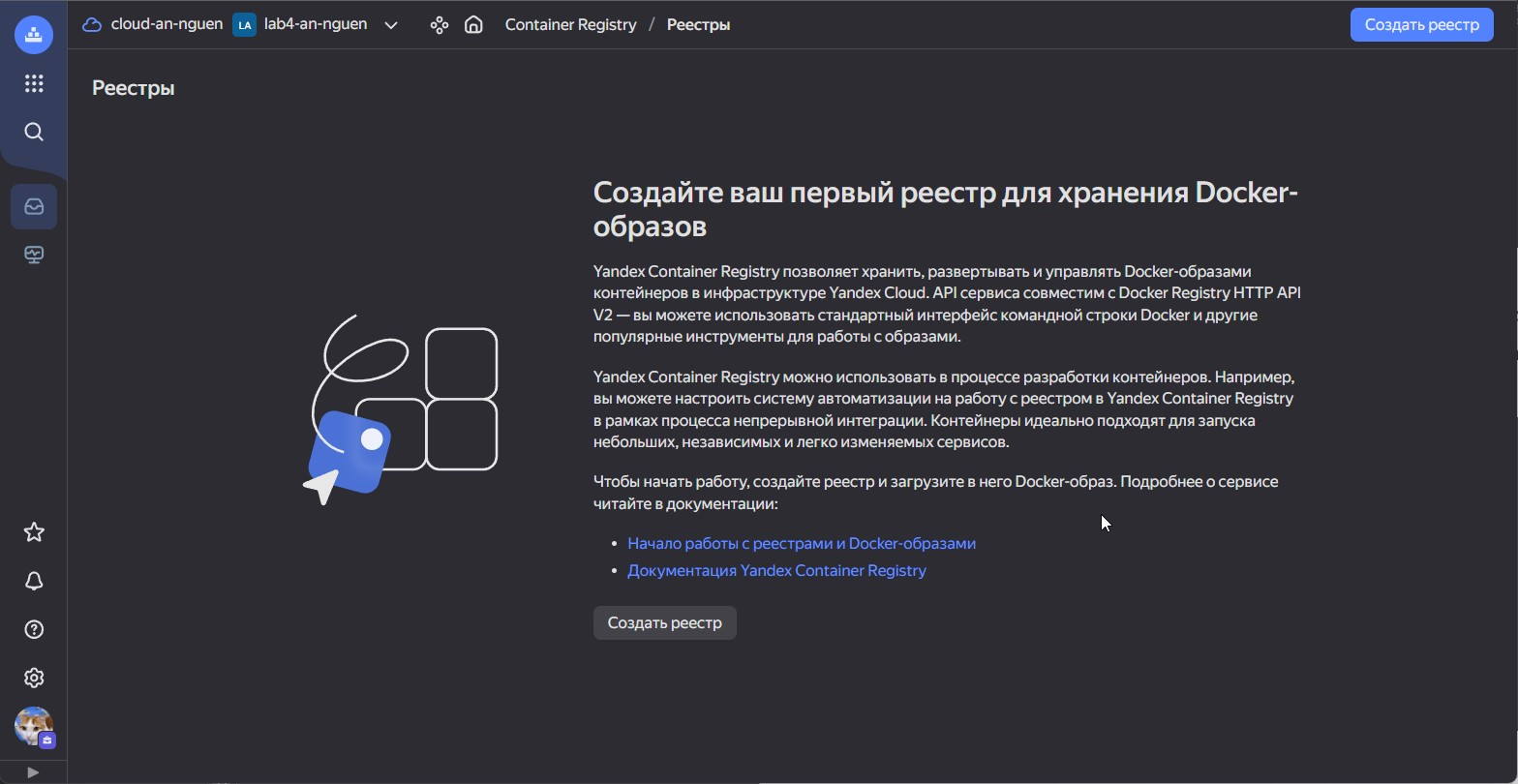


Рисунок 4

Нажимаем на кнопку «Создать реестр» и появляется окно создания реестра Вводим названия реестра и нажимаем на кнопку создания (рис. 5).

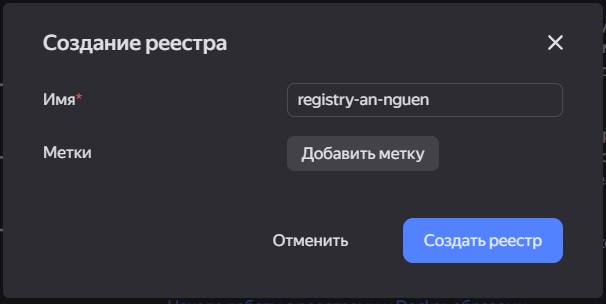


Рисунок 5

После создания на странице появится таблица с нашими реестрами (рис. 6).

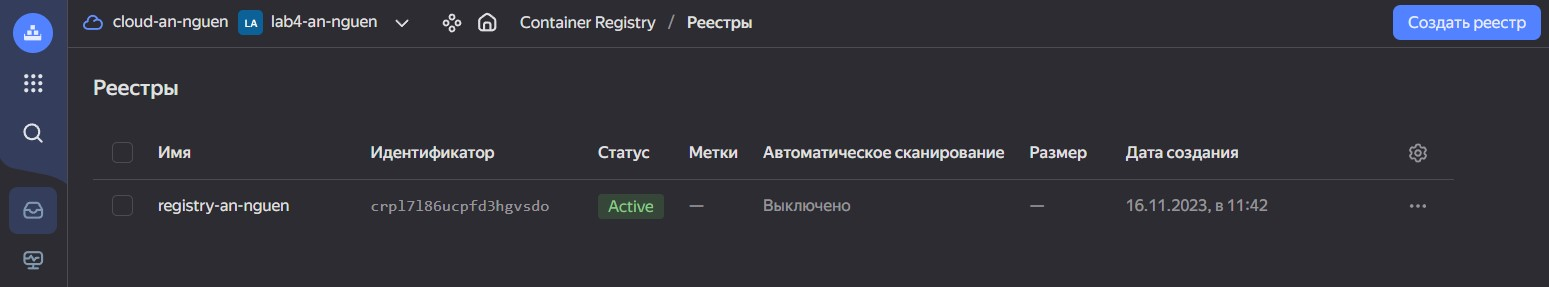


Рисунок 6

Для того чтобы мы могли загружать образы в реестр нам нужно авторизоваться. Исходя из документации <https://cloud.yandex.ru/docs/container-registry/operations/authentication> мы можем авторизоваться несколькими способами. Самый простой вариант - получение OAuth-токен. Для этого нужно перейти по адресу <https://cloud.yandex.ru/docs/container-registry/operations/authentication#user-oauth>, где откроется окно авторизации. После входа отобразится страница с нашим токеном.

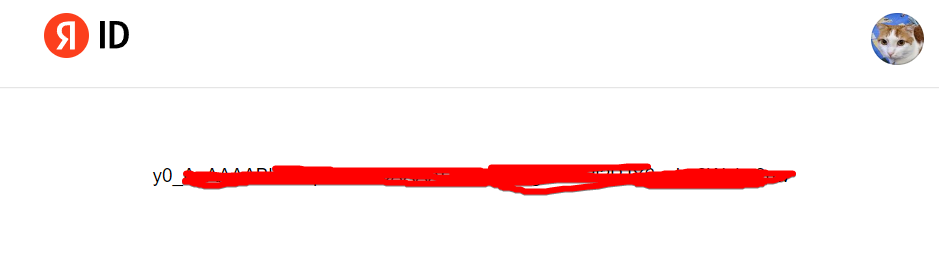


Рисунок 7

Далее нам нужно ввести следующую команду

docker login \

--username <тип\_токена> \

--password <токен> \

cr.yandex

где *<тип\_токена>* может принимать значения *oauth*, *iam* или *json\_key, <токен> -* сам токен. В нашем случае наша команда примерно будет выглядеть так

docker login --username oauth --password <мой\_токен> cr.yandex

Результат команды представлена на рисунке 8.

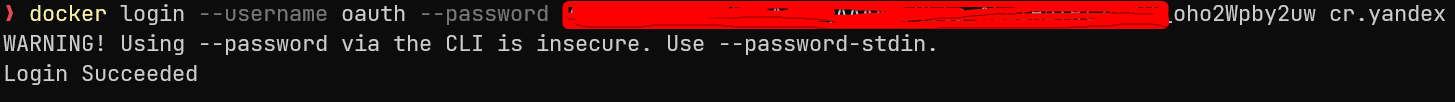


Рисунок 8

Далее собираем снова образ только с другим тегом. Команда будет выглядеть так:

docker build -t cr.yandex/<ид\_реестра>/<имя\_образа>:<тег> .

Здесь <ид\_реестра> есть идентификатор реестра контейнеров. Она у нас отображена в таблице реестра и указана красным цветом на рисунке 9.

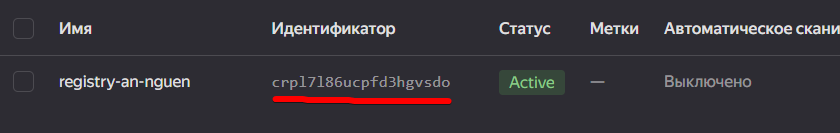


Рисунок 9

Имя образа было выбрано *blazor-server-an-nguen.* Тег можно выбрать любой.Получаем следующую команду

docker build -t cr.yandex/crpl7l86ucpfd3hgvsdo/blazor-server-an-nguen:latest .

После сборки образа, далее нам нужно загрузить образ в реестр используя следующую команду

docker push cr.yandex/crpl7l86ucpfd3hgvsdo/blazor-server-an-nguen

Результат работы команды изображена на рисунке 10.



Рисунок 10

Далее проверяем загрузился ли образ в реестр на странице реестра во вкладке «Обзор» (рис. 11).

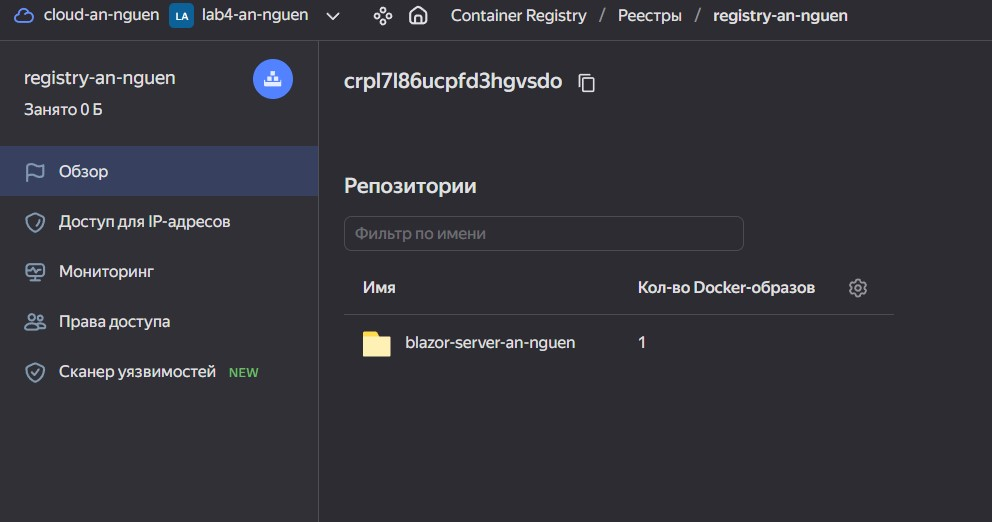


Рисунок 11

Далее заходим на страницу Serverless Containers (рис. 12).

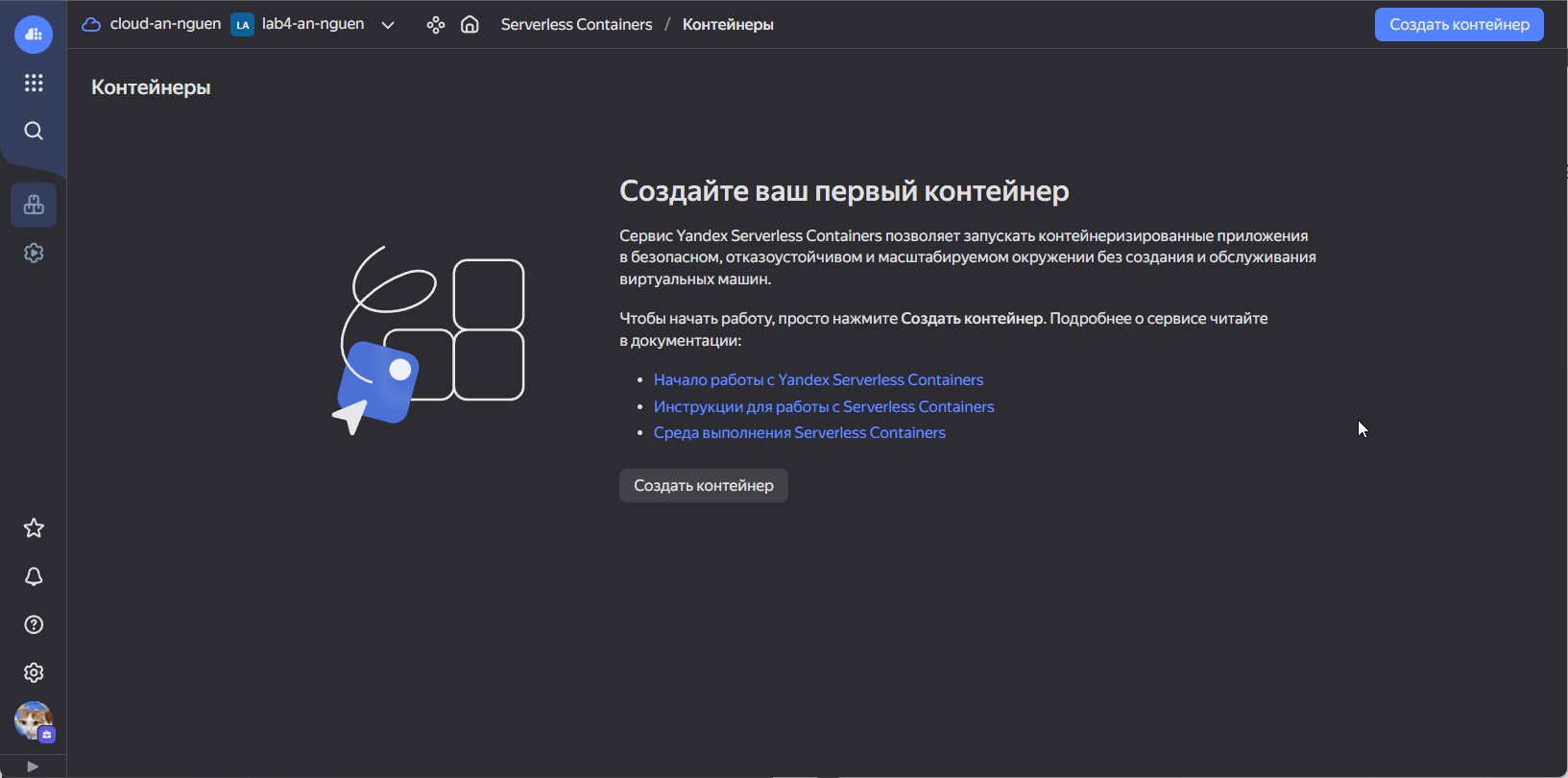


Рисунок 12

Создаём контейнер (рис. 13).

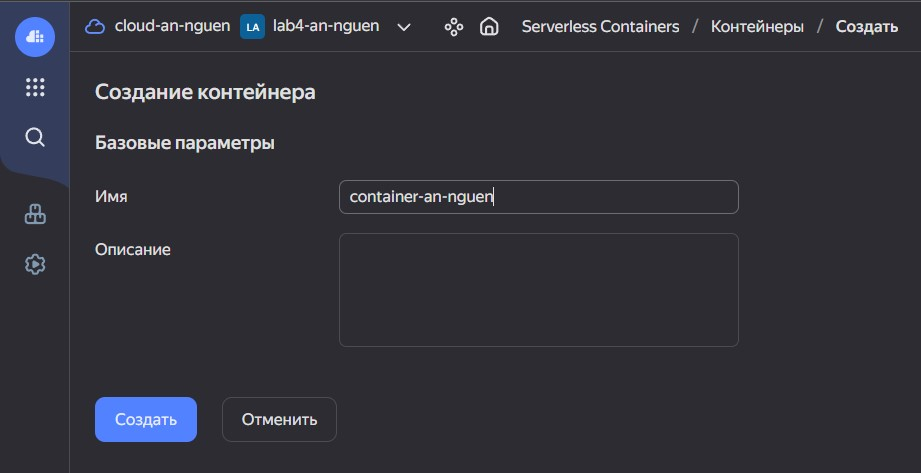


Рисунок 13

После создания откроется страница создания ревизии контейнера (рис. 14, 15).

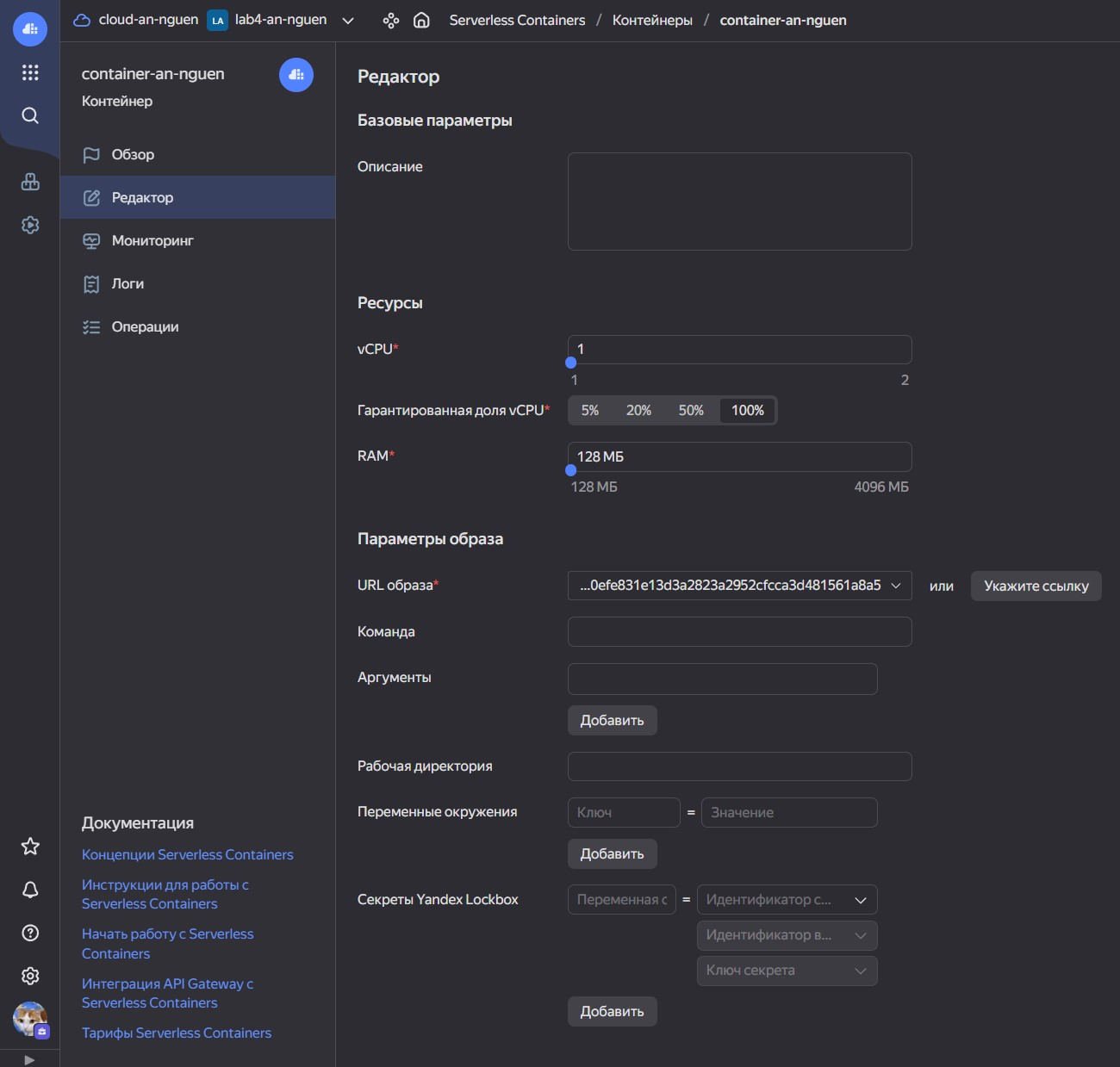


Рисунок 14

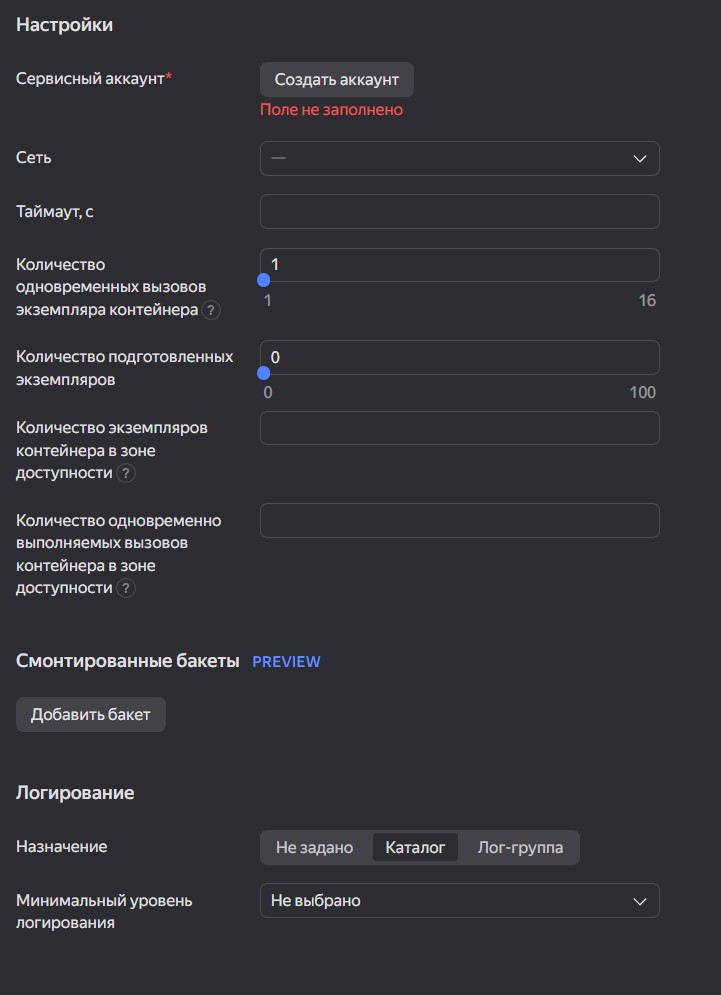


Рисунок 15

Здесь мы должны настроить ресурсы, выбрать образ и выбрать сервисный аккаунт. После настройки в верхнем правом углу нажимаем кнопку создания.

Далее нам нужно сделать доступ к сайту публичным. Для этого во вкладке обзора контейнера нужно включить пункт «Публичный контейнер» (рис. 16)

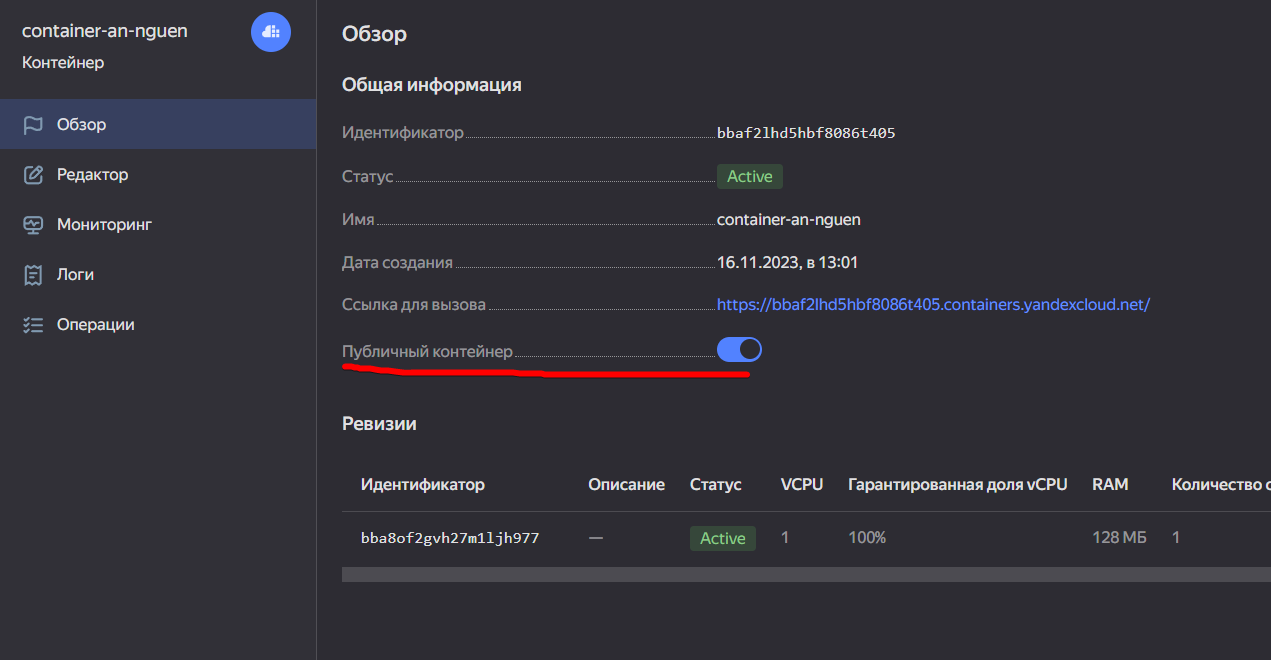


Рисунок 16

На этой же странице есть пункт «ссылка для вызова». Нажимаем на ссылку напротив этой графы, и через несколько секунд загружается наш сайт (рис. 17).

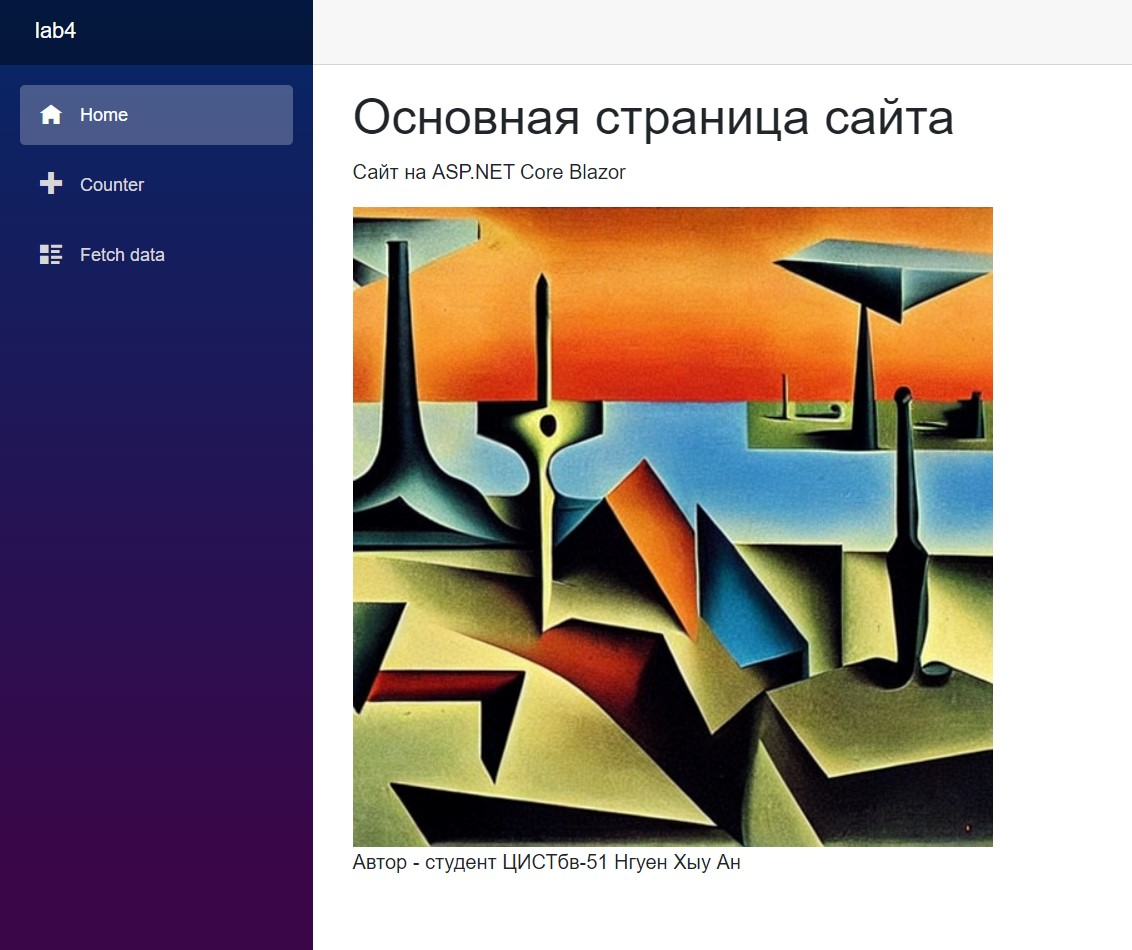


Рисунок 17

Вывод: в этой лабораторной работе мы познакомились с работой serverless решений от Яндекс Облако, разместили простенький сайт на нём и научились работать с Docker контейнерами.