UNLIMITED PRODUCTION

eXpress Communication System

Administrator’s Guide

Chatbot/SmartApp Deployment

07/02/2025

Mailing address:	127030, Moscow, 24/1 Novoslobodskaya Street
Phone:	+7 (499) 288-01-22
E-mail:	sales@express.ms
Web:	https://express.ms/

TABLE OF CONTENTS

INTRODUCTION **4**

DEPLOYMENT PROCESS **5**

Prerequisites **5**

System requirements 5

Checking Docker Operation 5

Registering a Chatbot in the Administrator Panel **5**

Preparing for Launch **7**

Downloading the Image 7

Creating a Shared Storage 8

Directory for Chat-Bot/Smartapp 8

Starting/Stopping the Bot **9**

Checking Chatbot Functionality **9**

UPDATING APP IMAGE **10**

Upgrading to a Specific Version 10

Upgrading to the Latest Version 10

INTRODUCTION

This guide is intended for administrators of eXpress Communication System, DevOps engineers and system programmers. It contains the information required to deploy a chatbot of a SmartApp.

Product Support Service You can contact the product support service by e-mail support@express.ms. The page of the product support service on the Unlimited Production website is available at https://express.ms/faq/.

Website. Information on the product by Unlimited Production can be found on the website https://express.ms/.

DEPLOYMENT PROCESS

PREREQUISITES

Chatbots and SmartApps require a separate server. There must be network communication set up between the bot server and the CTS in both directions.

SYSTEM REQUIREMENTS

The server on which the deployment is performed shall meet the following requirements:

- Linux OS;
- installed software:
  - Docker;
  - Docker Compose;
  - PostgreSQL v15 (Docker image or server);
  - Redis v7 (Docker image or server);
- an open free port in the range of 1024 to 65535.

CHECKING DOCKER OPERATION

To check Docker operation:

1. Check Docker status:

```
systemctl status docker
```

2. If the status is "inactive", start Docker with the following command:

```
sudo systemctl enable --now docker
```

The "enable" command ensures that Docker will start automatically after the system is turned on. Containers with the "restart: always" parameter will also be started.

Note. If the chatbot/SmartApp must interact with the customer's information system, contact the developer to clarify any questions related to additional requirements for network interactions.

REGISTERING A CHATBOT IN THE ADMINISTRATOR PANEL

Before launching, register the chatbot/SmartApp in the CTS Administrator Panel.

Note. To gain access to the CTS Administrator Panel, please contact the developers of eXpress CS.

To register a chatbot in the Administrator Panel:

1. Go to the "Bots" section using the side menu (see Figure 1).
2. Click the "Create Bot" button in the upper right corner.

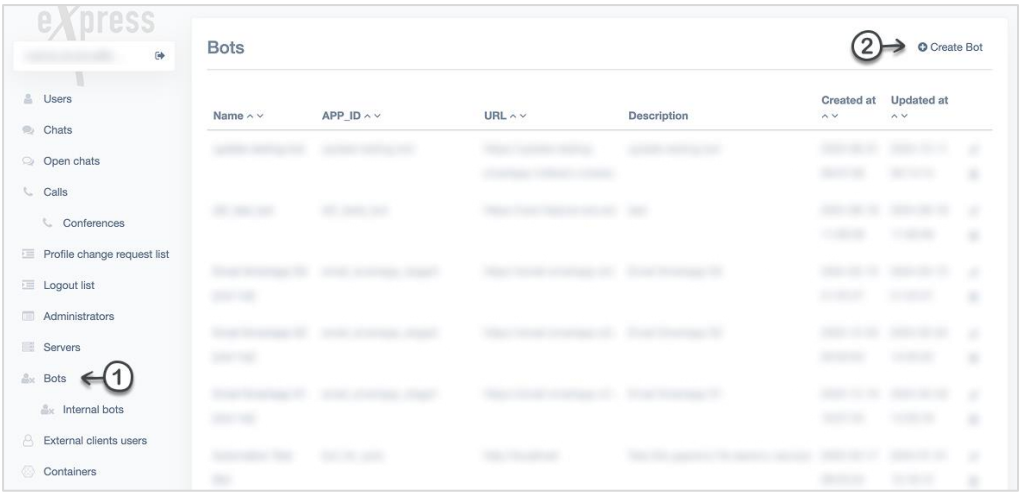

Figure 1

3. On the chatbot creation page, fill in all the fields and save the changes (see Figure 2).

   The chatbot will appear in the general list.

Note. When filling in the "URL" field, make sure the link is accessible from outside.

URL example: https://<bot server address>:8000.

If there are multiple chatbots installed on the server, the port specified in the docker-compose.yml file of a specific chatbot is used.

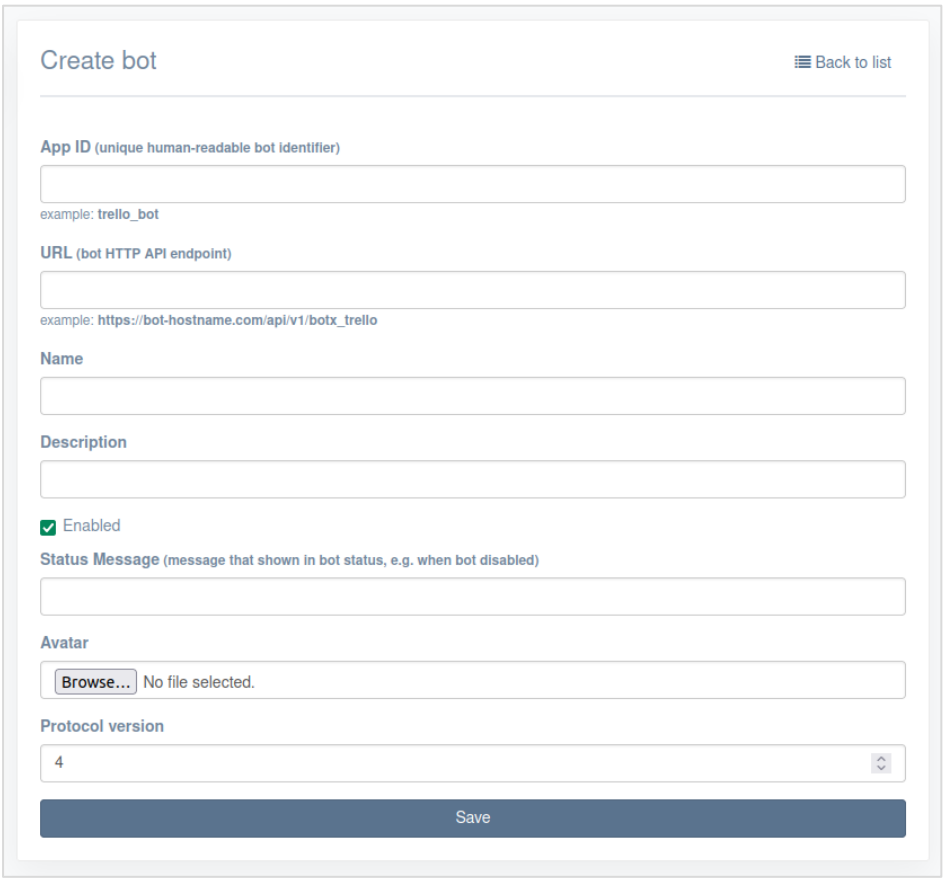

Figure 2

4. Click on the icon opposite the chatbot name (see Figure 3).

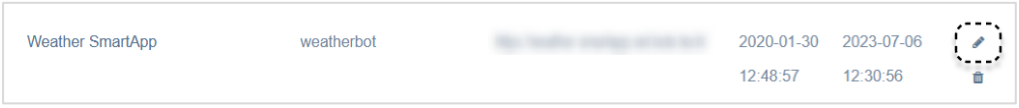

Figure 3

The chatbot editing page will open.

5. Note down the values of the "ID" and "Secret key" fields. They will be needed when creating a directory for the chatbot/SmartApp.
6. For SmartApp, fill in the fields in the corresponding block (see Figure 4) and save the changes.

Figure 4

PREPARING FOR LAUNCH

Place the following files, which were received from the developer, on the server:

- docker-compose.storages.yml
- docker-compose.yml
- example.env or .env

Further preparation includes the following steps:

1. Downloading the Image.
2. Creating a Shared Storage.
3. Creating Chatbot/SmartApp Directory.

DOWNLOADING THE IMAGE

Docker images of chatbots and SmartApps are available in the public Docker Registry at the address registry.public.express. The app image is specified in the docker-compose.yml file.

Note. The Login and Password valuesobtained from the developers are used as login and password.

To download the image:

1. Log in to Docker Registry with the following command:

```
docker login -u <Login> -p <Password> registry.public.express
```

2. Run the following command:

```
docker compose up -d
```

The app image will be downloaded automatically.

CREATING A SHARED STORAGE

Note. If the shared storage was created earlier (when deploying other chatbots), skip this step.

Create a separate docker network so that each chatbot/SmartApp does not create a new instance of PostgreSQL and Redis. In doing so, each app will use its own database.

To create a shared storage:

1. Create a directory for PostgreSQL and Redis:

```
mkdir -p /opt/express/bots/storages
```

2. Copy the attached docker-compose.storages.yml file to the directory that was created.
3. In the same directory, create an .env file with the following contents:

```
POSTGRES_USER="postgres" # General PostgreSQL user, the bot
will have its own user

POSTGRES_PASSWORD="<GENERATE>"
```

Note. To generate a password, use the “openssl rand -hex 32” command .

4. Run the containers with the following command:

```
docker compose -f docker-compose.storages.yml up -d
```

5. Check the status of containers with the following command:

```
docker compose -f docker-compose.storages.yml ps
```

The status must be set to “up”.

6. Check that the entries in the storage log files do not contain information about errors:

```
docker compose -f docker-compose.storages.yml logs
```

DIRECTORY FOR CHAT-BOT/SMARTAPP

To create a directory:

1. Create a directory with the following command:

```
mkdir -p /opt/express/bots/your_bot
```

2. Copy the attached docker-compose.yml file to the directory that was created. Edit it according to the comments in the file.
3. Create a database and user for the chatbot:

```
docker exec storages-postgres-1 psql -U postgres -c "create
user '<your_bot_user>'"

docker exec storages-postgres-1 psql -U postgres -c "alter user
'<your_bot_user>' with password '<GENERATE>'"

docker exec storages-postgres-1 psql -U postgres -c "create
database 'your_bot_db' with owner '<your_bot_user>'"
```

Note. To generate a password, use the “openssl rand -hex 32” command .

4. Copy the attached .env file to the directory that was created and set the values for the environment variables.

   Note. If the file name is example.env, rename it to .env.

   Please note:

   - the secret_key and bot_id from the BOT_CREDENTIALS variable are generated when registering a chatbot/SmartApp in the CTS Administrator Panel;
   - the host value matches the CTS Administrator Panel host.

STARTING/STOPPING THE BOT

To start the bot, run the following command:

```
docker compose up -d
```

To stop containers, run the following command**:**

```
docker compose down
```

CHECKING CHATBOT FUNCTIONALITY

After launching the chatbot container, check for errors with the following command:

```
docker compose logs
```

Possible causes of errors in Postgres or Redis:

- one of the previous steps was skipped;
- software versions installed do not comply with recommendations;
- services are not running.

If you experience any chatbot errors, please send diagnostic information to the developer.

To generate a log file with diagnostic information, run the following command:

```
docker logs <bot-container> >& <file_name>.log
```

Note. The bot-container value can be obtained from the NAMES column using the “docker ps” command (see Figure 5).

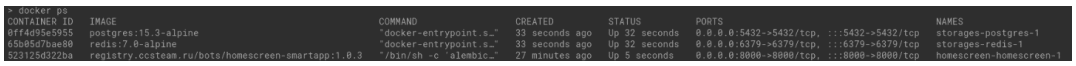

Figure 5

UPDATING APP IMAGE

Each app image is hosted in the public Docker Registry repository under a tag that corresponds to the app version. The latest available versions of images are additionally marked with the "latest" tag.

UPGRADING TO A SPECIFIC VERSION

To download a specific version of the app, specify the corresponding tag in the docker-compose.yml file. If the specified image version does not exist on your server, Docker will automatically download it when you start app containers.

To download the image manually, run the following command:

```
docker pull <image:tag>
```

UPGRADING TO THE LATEST VERSION

To update an image with the "latest" tag, first delete the image that is located on the server.

Note. Before deleting the app image, make sure that the containers with the chatbot/SmartApp are stopped.

To update to the latest version of the app, delete the existing image using the following command:

```
docker rmi <image:latest>
```

Docker will automatically download the new image when you start app containers.

To download the image manually, run the following command:

```
docker pull <image:latest>
```