TypeBot docker

Automação chatbot e whatsapp

- Instalação e configurações TypeBot
  - Instalação TypeBot docker
  - Dicas problemas de configurações
  - Configurations self-hosting TypeBot

Instalação e configurações TypeBot

Procedimentos de instalação e configuração TypeBot

Instalação TypeBot docker

Link: https://docs.typebot.io/self-hosting/deploy/docker em 06/06/2025

The easiest way to get started with Typebot is with the official managed service in the Cloud. You'll have high availability, backups, security, and maintenance all managed for you by me, Baptiste, Typebot's founder. The cloud version can save a substantial amount of developer time and resources. For most sites this ends up being the best value option and the revenue goes to funding the maintenance and further development of Typebot. So you'll be supporting fair source software and getting a great service!

Requirements

You need a server with Docker installed. If your server doesn't come with Docker pre-installed, you can follow their docs to install it.

Installation

1. Download the compose file

On your server, download the latest `docker-compose.yml` and the starter `.env` file:

Copy

```
wget https://raw.githubusercontent.com/baptisteArno/typebot.io/latest/docker-compose.yml
wget https://raw.githubusercontent.com/baptisteArno/typebot.io/latest/.env.example -O .env
```

2. Add the required configuration

1. You'll first need a random 32-character secret key which will be used to encrypt sensitive data. Here is a simple way to generate one:
   Copy
   ```
   openssl rand -base64 24 | tr -d '\n' ; echo
   ```
2. Fill the `.env` file with your values.
3. Configure at least one authentication provider (Email, Google, GitHub, Facebook or GitLab). More info here: Configuration.

By default the compose file will pull the latest stable Typebot images: `baptistearno/typebot-builder:latest` and `baptistearno/typebot-viewer:latest`. You can decide to replace `latest` with a specific version. You can find all the existing tags here

3. Start the server

Once you've added your configuration to the compose file, you're ready to start up the server:

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

When you run this command, by default, it does the following:

- Create a database
- Run the migrations
- Start the builder on port 8080
- Start the viewer on port 8081
- All Typebot's data is stored in the `.typebot` folder in the current directory

You can now navigate to `http://typebot.domain.com:8080` and see the login screen. Login with the admin email to have access to a Team plan workspace automatically.

Typebot server itself does not perform SSL termination. It only runs on unencrypted HTTP. If you want to run on HTTPS you also need to set up a reverse proxy in front of the server. See below instructions.

Update Typebot

Typebot is updated regularly, but it is up to you to apply these updates on your server. By virtue of using Docker, these updates are safe and easy to apply.

1. Pull the new images:
   Copy
   ```
   docker-compose pull typebot-builder
   docker-compose pull typebot-viewer
   ```
   Alternatively, you can pull specific versions:
   Copy
   ```
   docker-compose pull typebot-builder:3.4.2
   docker-compose pull typebot-viewer:3.4.2
   ```
2. Stop the server:
   Copy
   ```
   docker-compose down
   ```
3. Start the server (with the new images):
   Copy
   ```
   docker-compose up -d
   ```

The self-hosted version is somewhat of a LTS, only getting the changes (~ once per month) after they have been battle tested on the cloud version. If you want features as soon as they are available, consider becoming a cloud user.

Optional extras

Reverse proxy

By default, Typebot runs on unencrypted HTTP on ports 8080 for the builder and 8081 for the viewer. We recommend running it on HTTPS behind a reverse proxy of some sort. You may or may not already be running a reverse proxy on your host, let’s look at both options:

No existing reverse proxy

If your DNS is managed by a service that offers a proxy option with automatic SSL management, feel free to use that. For example, you could use Cloudflare as a reverse proxy in front of Typebot.

Alternatively, you can run your Caddy server as a reverse proxy. This way your SSL certificate will be stored on the host machine and managed by Let’s Encrypt. The Caddy server will expose port 443, terminate SSL traffic and proxy the requests to your Typebot server.

Here is an example of a docker-compose file using Caddy as a reverse proxy:

Copy

```
services:
  caddy-gen:
    container_name: caddy-gen
    image: 'wemakeservices/caddy-gen:latest'
    restart: always
    volumes:
      - /var/run/docker.sock:/tmp/docker.sock:ro
      - {$PWD}/.typebot/caddy-certificates:/data/caddy
    ports:
      - '80:80'
      - '443:443'
    depends_on:
      - typebot-builder
      - typebot-viewer

  typebot-builder:
    labels:
      virtual.host: 'typebot.domain.com' # change to your domain name
      virtual.port: '3000'
      virtual.tls-email: 'admin@example.com' # change to your email
```

```
  typebot-viewer:
    labels:
      virtual.host: 'bot.domain.com' # change to your domain name
      virtual.port: '3000'
      virtual.tls-email: 'admin@example.com' # change to your email
      # Necessary to enable message streaming
      virtual.proxy.directives: |
        flush_interval -1
```

This config requires you to add the following DNS entry:

Copy

```
typebot IN A <server_ip>
bot IN A <server_ip>
```

You can merge this compose file with the first one. Make sure that `NEXTAUTH_URL` is set to `https://typebot.domain.com` and `NEXT_PUBLIC_VIEWER_URL` is set to `https://bot.domain.com`.

When running the compose file, it should automatically enable SSL on your server and you should be able to navigate to:

- `https://typebot.domain.com` for the builder
- `https://bot.domain.com` for the viewer

Existing reverse proxy

If you're already running a reverse proxy, the most important things to note are:

1. Configure the virtual hosts to match the `NEXTAUTH_URL` and `NEXT_PUBLIC_VIEWER_URL` in your `docker-compose` configuration.
2. Proxy the traffic to `127.0.0.1:8080` or `{ip-address}:8080` and to `127.0.0.1:8081` or `{ip-address}:8081` if running on a remote machine

SMTP

I highly recommend using an external SMTP service. There are tons of options out there, including SendInBlue, Mailgun and SendGrid. It will avoid severe headaches □□. Then, you will only need to add the required

SMTP configuration variables.

If, however, you don’t want to, you can instantiate an SMTP server in the docker-compose file.

Copy

```
services:
  mail:
    image: bytemark/smtp
    restart: always
```

And add the following variables to your `.env` file:

Copy

```
SMTP_HOST=mail
NEXT_PUBLIC_SMTP_FROM=notifications@typebot.domain.com
```

You will probably need to make sure that `typebot.domain.com` has a valid SPF record and that your server IP has a rDNS set up.

You can merge this compose file with the main one.

S3 storage

If you don’t already have an S3 storage available, you could include it in your docker-compose file:

Copy

```
services:
  minio:
    image: minio/minio
    command: server /data
    ports:
      - '9000:9000'
    environment:
      MINIO_ROOT_USER: minio
      MINIO_ROOT_PASSWORD: minio123
    volumes:
```

```
      - s3-data:/data
  # This service just makes sure a bucket with the right policies is created
  createbuckets:
    image: minio/mc
    depends_on:
      - minio
    entrypoint: >
      /bin/sh -c "
      sleep 10;
      /usr/bin/mc config host add minio http://minio:9000 minio minio123;
      /usr/bin/mc mb minio/typebot;
      /usr/bin/mc anonymous set public minio/typebot/public;
      exit 0;
      "

volumes:
  s3-data:
```

And add the following variables to your `.env` file:

Copy

```
S3_ACCESS_KEY=minio
S3_SECRET_KEY=minio123
S3_BUCKET=typebot
S3_ENDPOINT=storage.domain.com
```

This config requires you to add the following DNS entry:

Copy

```
storage IN A <server_ip>
```

You can merge this compose file with the main one.

Config example with all the extras

Here is a config example that spins up Typebot with HTTPS, SMTP and S3 storage.

Copy

```
services:
  caddy-gen:
    image: 'wemakeservices/caddy-gen:latest'
    restart: always
    volumes:
      - /var/run/docker.sock:/tmp/docker.sock:ro
      - {$PWD}/.typebot/caddy-certificates:/data/caddy
    ports:
      - '80:80'
      - '443:443'
    depends_on:
      - typebot-builder
      - typebot-viewer
  typebot-db:
    image: postgres:16
    restart: always
    volumes:
      - db-data:/var/lib/postgresql/data
    environment:
      - POSTGRES_DB=typebot
      - POSTGRES_PASSWORD=typebot
    healthcheck:
        test: ["CMD-SHELL", "pg_isready -U postgres"]
        interval: 5s
        timeout: 5s
        retries: 5
  typebot-builder:
    labels:
      virtual.host: 'typebot.domain.com' # change to your domain
      virtual.port: '3000'
      virtual.tls-email: 'admin@example.com' # change to your email
    image: baptistearno/typebot-builder:latest
    depends_on:
      typebot-db:
        condition: service_healthy
    restart: always
    extra_hosts:
      - 'host.docker.internal:host-gateway'
```

```
  # See https://docs.typebot.io/self-hosting/configuration for more configuration options
  env_file:
    - .env
typebot-viewer:
  labels:
    virtual.host: 'bot.domain.com' # change to your domain
    virtual.port: '3000'
    virtual.tls-email: 'admin@example.com' # change to your email
    # Necessary to enable message streaming
    virtual.proxy.directives: |
      flush_interval -1
  image: baptistearno/typebot-viewer:latest
  depends_on:
    typebot-db:
      condition: service_healthy
  restart: always
  # See https://docs.typebot.io/self-hosting/configuration for more configuration options
  env_file:
    - .env
mail:
  image: bytemark/smtp
  restart: always
minio:
  labels:
    virtual.host: 'storage.domain.com' # change to your domain
    virtual.port: '9000'
    virtual.tls-email: 'admin@example.com' # change to your email
  image: minio/minio
  command: server /data
  ports:
    - '9000:9000'
  environment:
    MINIO_ROOT_USER: minio
    MINIO_ROOT_PASSWORD: minio123
  volumes:
    - s3-data:/data
# This service just make sure a bucket with the right policies is created
createbuckets:
  image: minio/mc
  depends_on:
    - minio
```

```
    entrypoint: >
      /bin/sh -c "
      sleep 10;
      /usr/bin/mc config host add minio http://minio:9000 minio minio123;
      /usr/bin/mc mb minio/typebot;
      /usr/bin/mc anonymous set public minio/typebot/public;
      exit 0;
      "

volumes:
  db-data:
  s3-data:
```

Build your own images

To build your own builder Docker image

Copy

```
docker build -t typebot-builder --build-arg SCOPE=builder .
```

To build your own viewer Docker image

Copy

```
docker build -t typebot-viewer --build-arg SCOPE=viewer .
```

Troubleshooting

Avoiding network issues when migrating in Portainer

If you migrate a Typebot stack between Portainer instances, hostname resolution may fail, causing `typebot-viewer` to be unable to connect to `typebot-db`.

Before deploying the stack in Portainer, ensure the network is **explicitly defined as attachable** in `docker-compose.yml`:

Copy

```
networks:
  typebot_network:
    driver: bridge
    attachable: true

services:
  typebot-db:
    ...
    networks:
      - typebot_network
  typebot-builder:
    ...
    networks:
      - typebot_network
  typebot-viewer:
    ...
    networks:
      - typebot_network
```

When deploying the stack, Portainer will automatically create the network with the correct settings. This prevents hostname resolution issues after migration.

>

If you're self-hosting Typebot, sponsoring me is a great way to give back to the community and to contribute to the long-term sustainability of the project. It also comes with some perks like priority support and private workshops. ❤

>

This doc has been inspired by Plausible docs. They have a similar self-hosting solutions, and their documentation is □□.

Dicas problemas de configurações

LInk: https://github.com/baptisteArno/typebot.io/issues/942

Procedimentos extraídos de listas do typebot sobre dificuldades de instalações.

Cannot get SMTP config to send email #942

baptisteArno commented on Oct 17, 2023 •
Probably an issue with the single quote on `NEXT_PUBLIC_SMTP_FROM`
My config with Sendinblue:

```
SMTP_USERNAME=baptiste@typebot.io
SMTP_PASSWORD=...
SMTP_HOST=smtp-relay.sendinblue.com
SMTP_PORT=587
NEXT_PUBLIC_SMTP_FROM="'Typebot Notifications' <notifications@typebot.io>"
```

MAS-CreativeLabs commented on Nov 12, 2023

I apologize for the delayed reply, I was on a trip. her's my configuration: I'm hosting typebot on a Synology NAS with a reverse proxy (the same exact test machine I use for all my docker in my testing environment. I used portainer for the deployment of typebot and I followed the instructions from the official documentation.

Docker compose file :

`version: '3.3'

volumes:
db-data:

services:
typebot-db:
image: postgres:14-alpine

restart: always
volumes:
- db-data:/var/lib/postgresql/data

```
environment:
  - POSTGRES_DB=typebot
  - POSTGRES_PASSWORD=typebot
```

typebot-builder:
image: baptistearno/typebot-builder:latest
restart: always
depends_on:
- typebot-db
ports:
- '8080:3000'
extra_hosts:
- 'host.docker.internal:host-gateway'
env_file: stack.env

typebot-viewer:
image: baptistearno/typebot-viewer:latest
restart: always
ports:
- '8081:3000'
env_file: stack.env`

.env file:

`ENCRYPTION_SECRET=XXXXXXXXXXXXXXXXXXXXXXXX

DATABASE_URL=postgresql://postgres:typebot@typebot-db:5432/typebot

NEXTAUTH_URL=https://typebot.mydonmain.com

NEXT_PUBLIC_VIEWER_URL=https://bot.mydonmain.com

ADMIN_EMAIL=XXXXX@gmail.com

NEXTAUTH_URL_INTERNAL=http://localhost:3000

DEFAULT_WORKSPACE_PLAN=UNLIMITED

DISABLE_SIGNUP=false

NEXT_PUBLIC_BOT_FILE_UPLOAD_MAX_SIZE=250

SMTP_USERNAME=XXXXX@XXXXX.com

SMTP_PASSWORD=XXXXXXXXXXXXXXXXXXX

SMTP_HOST=ssl0.ovh.net

SMTP_PORT=465

SMTP_SECURE=true

NEXT_PUBLIC_SMTP_FROM="'Typebot Notifications' XXXXX@XXXXX.com"

GOOGLE_CLIENT_ID=XXXXXXXXXX-
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.apps.googleusercontent.com

GOOGLE_CLIENT_SECRET=XXXXX-XXXXXXXXXX_XXXXX

GITHUB_CLIENT_ID=XXXXXXXXXXXXXXXXXXXXXXXX

GITHUB_CLIENT_SECRET=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

FACEBOOK_CLIENT_ID=XXXXXXXXXXXXXXXXXX

FACEBOOK_CLIENT_SECRET=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

NEXT_PUBLIC_GIPHY_API_KEY=XXXXXXXXXXXXXXXXXXXXXXXX

NEXT_PUBLIC_UNSPLASH_APP_NAME=Typebot

NEXT_PUBLIC_UNSPLASH_ACCESS_KEY=XXXXXXXXXXXXXXXXXXXXXXXX

Obs: Para funcionar no laboratório nestas configurações foi apontado o NEXTAUTH_URL e NEXT_PUBLIC_VIEWER_URL para o mesmo endereço de viewer.

Configurations self-hosting TypeBot

Link: https://docs.typebot.io/self-hosting/configuration

Configuration

If you're self-hosting Typebot, sponsoring me is a great way to give back to the community and to contribute to the long-term sustainability of the project. It also comes with some perks like priority support and private workshops. ❤

Parameters marked with * are required.

General

Parameter	Default	Description
DATABASE_URL *		The database URL
ENCRYPTION_SECRET *		A 256-bit key used to encrypt sensitive data. It is strongly recommended to generate a new one. The secret should be the same between builder and viewer.
NEXTAUTH_URL *		The builder base URL. Should be the publicly accessible URL (i.e. `https://typebot.domain.com`)
NEXT_PUBLIC_VIEWER_URL *		The viewer base URL. Should be the publicly accessible URL (i.e. `https://bot.domain.com`)
ADMIN_EMAIL		The email that will get an `UNLIMITED` plan on user creation. The associated user will be able to bypass database rules. You can provide multiple emails separated by a comma without spaces.
NEXTAUTH_URL_INTERNAL		The internal builder base URL. You have to set it only when `NEXTAUTH_URL` can't be reached by your builder container / server. For a docker deployment, you should set it to `http://localhost:3000`.
DEFAULT_WORKSPACE_PLAN	FREE	Default workspace plan on user creation or when a user creates a new workspace. Possible values are `FREE`, `STARTER`, `PRO`, `LIFETIME`, `UNLIMITED`. The default plan for admin user is `UNLIMITED`
DISABLE_SIGNUP	false	Disable new user sign ups. Invited users are still able to sign up.

Parameter	Default	Description
NEXT_PUBLIC_ONBOARDING_TYPEBOT_ID		Typebot ID used for the onboarding. Onboarding page is skipped if not provided.
DEBUG	false	If enabled, the server will print valuable logs to debug config issues.
NEXT_PUBLIC_BOT_FILE_UPLOAD_MAX_SIZE		Limits the size of each file that can be uploaded in the bots (i.e. Set `10` to limit the file upload to 10MB)

Email (Auth, notifications)

Used for sending email notifications and authentication

Parameter	Default	Description
SMTP_USERNAME		SMTP username
SMTP_PASSWORD		SMTP password
SMTP_HOST		SMTP host. (i.e. `smtp.host.com`)
SMTP_PORT	25	SMTP port
NEXT_PUBLIC_SMTP_FROM		From name and email (i.e. `'Typebot Notifications' <notifications@host.com>`)
SMTP_SECURE	false	If true the connection will use TLS when connecting to server. If false (the default) then TLS is used if server supports the STARTTLS extension. In most cases set this value to true if you are connecting to port 465. For port 587 or 25 keep it false
SMTP_AUTH_DISABLED	false	To disable the authentication by email but still use the provided config for notifications

Google Auth

Parameter	Default	Description
GOOGLE_CLIENT_ID		The Client ID from the Google API Console
GOOGLE_CLIENT_SECRET		The Client secret from the Google API Console

Parameter	Default	Description
NEXT_PUBLIC_GOOGLE_API_KEY		The API Key from the Google API Console

Google Sheets

Parameter	Default	Description
GOOGLE_CLIENT_ID		The Client ID from the Google API Console
GOOGLE_CLIENT_SECRET		The Client secret from the Google API Console
NEXT_PUBLIC_GOOGLE_API_KEY		The API Key from the Google API Console

Google Fonts

Used authentication in the builder and for the Google Sheets integration step.

Parameter	Default	Description
NEXT_PUBLIC_GOOGLE_API_KEY		The API Key from the Google API Console

GitHub (Auth)

Used for authenticating with GitHub. By default, it uses the credentials of a Typebot-dev app.

You can create your own GitHub OAuth app here. The Authorization callback URL should be `$NEXTAUTH_URL/api/auth/callback/github`

Parameter	Default	Description
GITHUB_CLIENT_ID		Application client ID. Also used to check if it is enabled in the front-end
GITHUB_CLIENT_SECRET		Application secret

GitLab (Auth)

Used for authenticating with GitLab. Follow the official GitLab guide for creating OAuth2 applications here. The Authorization callback URL should be `$NEXTAUTH_URL/api/auth/callback/gitlab`

Parameter	Default	Description
GITLAB_CLIENT_ID		Application client ID. Also used to check if it is enabled in the front-end
GITLAB_CLIENT_SECRET		Application secret
GITLAB_BASE_URL	https://gitlab.com	Base URL of the GitLab instance
GITLAB_REQUIRED_GROUPS		Comma-separated list of groups the user has to be a direct member of, e.g. `foo,bar`
GITLAB_NAME	GitLab	Name of the GitLab instance, used for the SSO Login Button

Facebook (Auth)

You can create your own Facebook OAuth app here. The Authorization callback URL should be `$NEXTAUTH_URL/api/auth/callback/facebook`

Parameter	Default	Description
FACEBOOK_CLIENT_ID		Application client ID. Also used to check if it is enabled in the front-end
FACEBOOK_CLIENT_SECRET		Application secret

Azure AD (Auth)

If you are using Azure Active Directory for the authentication you can set the following environment variables. The Authorization callback URL should be `$NEXTAUTH_URL/api/auth/callback/azure-ad`

Parameter	Default	Description
AZURE_AD_CLIENT_ID		Application client ID
AZURE_AD_CLIENT_SECRET		Application client secret. Can be obtained from Azure Portal.
AZURE_AD_TENANT_ID		Azure Tenant ID

Keycloak (Auth)

Used for authenticating with Keycloak. Follow the official Keycloak guide for creating OAuth2 applications here.

Parameter	Default	Description
KEYCLOAK_CLIENT_ID		Application client ID.
KEYCLOAK_CLIENT_SECRET		Application secret
KEYCLOAK_REALM		Your Keycloak Realm
KEYCLOAK_BASE_URL		Base URL of the Keycloak instance

Custom OAuth Provider (Auth)

Parameter	Default	Description
CUSTOM_OAUTH_NAME	Custom OAuth	Provider name. Will be displayed in the sign in form.
CUSTOM_OAUTH_CLIENT_ID		OAuth client ID.
CUSTOM_OAUTH_CLIENT_SECRET		OAuth client secret.
CUSTOM_OAUTH_WELL_KNOWN_URL		OAuth .well-known URL (i.e. `https://auth.domain.com/.well-known/openid-configuration`)
CUSTOM_OAUTH_USER_ID_PATH	id	Used to map the id from the user info object
CUSTOM_OAUTH_USER_NAME_PATH	name	Used to map the name from the user info object

Parameter	Default	Description
CUSTOM_OAUTH_USER_EMAIL_PATH	email	Used to map the email from the user info object
CUSTOM_OAUTH_USER_IMAGE_PATH	image	Used to map the image from the user info object
CUSTOM_OAUTH_SCOPE	openid profile email	OAuth scope

For `*_PATH` parameters, you can use dot notation to access nested properties (i.e. `account.name`).

The Authorization callback URL should be: `$NEXTAUTH_URL/api/auth/callback/custom-oauth`

S3 Storage (Media uploads)

Used for uploading images, videos, etc... It can be any S3 compatible object storage service (Minio, Digital Oceans Space, AWS S3...)

Parameter	Default	Description
S3_ACCESS_KEY		S3 access key. Also used to check if upload feature is enabled
S3_SECRET_KEY		S3 secret key.
S3_BUCKET	typebot	Name of the bucket where assets will be uploaded in.
S3_PORT		S3 Host port number
S3_ENDPOINT		S3 endpoint (i.e. `s3.domain.com`).
S3_SSL	true	Use SSL when establishing the connection.
S3_REGION		S3 region.
S3_PUBLIC_CUSTOM_DOMAIN		If the final URL that is used to read public files is different from `S3_ENDPOINT`

Note that for AWS S3, your endpoint is usually: `s3.<S3_REGION>.amazonaws.com`

In order to function properly, your S3 bucket must be configured. Make sure to read through the S3 configuration doc.

Giphy (GIF picker)

Used to search for GIF. You can create a Giphy app here

Parameter	Default	Description
NEXT_PUBLIC_GIPHY_API_KEY		Giphy API key

Unsplash (image picker)

Used to search for images. You can create an Unsplash app here

Parameter	Default	Description
NEXT_PUBLIC_UNSPLASH_APP_NAME		Unsplash App name
NEXT_PUBLIC_UNSPLASH_ACCESS_KEY		Unsplash API key

Pexels (video picker)

Used to search for videos. You can create a Pexels app here

Parameter	Default	Description
NEXT_PUBLIC_PEXELS_API_KEY		Pexels API key

Tolgee (i18n contribution dev tool)

If you'd like to join contribute to Typebot's translation join the Discord server and ask for an access to Tolgee in the #contributors channel.

Set up these environment variables to enable Tolgee dev tool.

Parameter	Default	Description
NEXT_PUBLIC_TOLGEE_API_KEY		Your Tolgee API key
NEXT_PUBLIC_TOLGEE_API_URL	https://tolgee.server.baptistearno.com	The Tolgee API base URL

WhatsApp (Preview)

In order to be able to test your bot on WhatsApp from the Preview drawer, you need to set up a WhatsApp business app.

Parameter	Default	Description
META_SYSTEM_USER_TOKEN		The system user token used to send WhatsApp messages
WHATSAPP_PREVIEW_FROM_PHONE_NUMBER_ID		The phone number ID from which the message will be sent
WHATSAPP_PREVIEW_TEMPLATE_NAME		The preview start template message name
WHATSAPP_PREVIEW_TEMPLATE_LANG	en_US	The preview start template message name
WHATSAPP_CLOUD_API_URL	https://graph.facebook.com	The WhatsApp Cloud API base URL
WHATSAPP_INTERACTIVE_GROUP_SIZE	3	The array size of items to send to API on choice input. You can't choose a number higher than 3 if you are using the official cloud API URL.

Redis

In Typebot, Redis is optional and is used to:

- Rate limit the sign in requests based on user IP
- Enable multiple media upload on WhatsApp

Parameter	Default	Description
REDIS_URL		The database URL. i.e. `redis://<username>:<password>@<host>:<port>`

Others

The official Typebot managed service uses other services such as Stripe for processing payments, Sentry for tracking bugs and Sleekplan for user feedbacks.

The related environment variables are listed here but you are probably not interested in these if you self-host Typebot.

Vercel (custom domains)

Parameter	Default	Description
VERCEL_TOKEN		Vercel API token
NEXT_PUBLIC_VERCEL_VIEWER_PROJECT_NAME		The name of the viewer project in Vercel
VERCEL_TEAM_ID		Vercel team ID that contains the viewer project

Telemetry

Parameter	Default	Description
MESSAGE_WEBHOOK_URL		Webhook URL called to receive important system messages
USER_CREATED_WEBHOOK_URL		Webhook URL called whenever a new user is created

PostHog

Parameter	Default	Description
NEXT_PUBLIC_POSTHOG_KEY		PostHog API Key
NEXT_PUBLIC_POSTHOG_HOST	https://app.posthog.com	PostHog API Host

System labels

Parameter	Default	Description
NEXT_PUBLIC_VIEWER_404_TITLE	404	
NEXT_PUBLIC_VIEWER_404_SUBTITLE	The bot you’re looking for doesn’t exist	