
RestorePhotos.io

This project restores old face photos using AI. Watch the 4 minute explainer video to see how I built this or see the 15 second demo.



Restore any face photo

Side by Side ☐ Compare

Original Photo



Restored Photo



Upload New Photo

Download Restored Photo

Powered by [Next.js](#), [Vercel](#), & [Replicate](#).



How it works

It uses an ML model from the Applied Research Center called GFPGAN on Replicate to restore face photos. This application gives you the ability to upload any photo, which will send it through this ML Model using a Next.js API route, and return your restored photo.

Running Locally

Note: I just added auth so these steps are not complete as of now. You can git clone from this specific commit.

Cloning the repository the local machine.

```
1 git clone
```

Creating a account on Replicate to get an API key.

1. Go to Replicate to make an account.
2. Click on your profile picture in the top right corner, and click on “Dashboard”.
3. Click on “Account” in the navbar. And, here you can find your API token, copy it.

Storing API key in .env file.

Create a file in root directory of project with env. And store your API key in it, as shown in the .example.env file.

If you'd also like to do rate limiting, create an account on UpStash, create a Redis database, and populate the two environment variables in `.env` as well. If you don't want to do rate limiting, you don't need to make any changes.

Installing the dependencies.

```
1 npm install
```

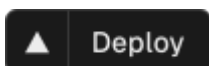
Running the application.

Then, run the application in the command line and it will be available at <http://localhost:3000>.

```
1 npm run dev
```

One-Click Deploy

Deploy the example using Vercel:



Powered by

This example is powered by the following services:

- Replicate (AI API)
- Bytescale (storage + image processing API)
- Vercel (hosting, serverless functions, analytics)
- Auth.js + Neon (auth + DB)
- Upstash Redis (rate limiting)