# Contents





Written By Ethan Krimer (V. 0 – 29.8.2023)
Validated By Arad Donenfeld (V. 1 – 6.9.2024)

## App Designation

The application will operate as a platform for uploading and reviewing CV files between the members of the Magshimim-Next community.
Only verified members will be able to use the website.
This will allow us to improve and maximize the potential of every member's CV – by getting a lot of input from various users.

## How Will It Work?

Upon entering the platform, a user will have to register/sign in with a google account.

After that, an admin must change the user type of the new user (read more about the user type in the data schemas section).

This is done because we want only Magshimim Next members to be able to join.

Once the user is signed in and with the correct permissions– he will be able to upload as well as view and comment on the various CVs of the active platform members.

### Uploading a CV

A user can choose to upload a link that withstand the following standards:

- The link must be a link to a Google Docs file.
- The file must be open to view to anyone with the link.

- Once a user submits a link, it will be added with the user's ID.
    - This means we can upload several CVs at a time.
- Upon submission – each CV should also be linked to at least one job category (read more about categories in the data schemes section).
- Each CV must also include a brief description of the CV.

### Viewing a CV

A user can view any CVs that he'd like – the CVs will be shown in a feed-like page.
The feed will have filtering options but in default settings – will show the most recently Uploaded CVs.

- There will also be an option to filter by category (read more about categories in the data schemes section), by description, and by the user uploading.

Once a user has clicked on a CV, he will be sent to a page dedicated to the CV where he can leave a detailed review of that CV and even resolve any of his reviews.

- Liking comments and commenting under other comments is also available.



Written By Ethan Krimer (V. 0 – 29.8.2023)
Validated By Arad Donenfeld (V. 1 – 6.9.2024)

## Technologies

Due to the short time schedule on this project and the project being a voluntary work – we will not be creating a server side for this platform but instead use serverless solutions such as Supabase.

- Supabase – The Free Firebase Alternative – as the DB of the project.
- Digital Ocean Kubernetes – to host the application and the development – preferably with a custom docker image.
- Typescript NextJS – To code the application.
- Github – Hosting the code, running relevant pipelines, and hosting a static page for the community using Github Pages.
- Namecheap – Registering DNS.

Although it is important to note that due to everything being voluntary work, financing support for thousands of users is not possible, so selfhosted-limitless solutions might be the way to go in the future and student credits are our way to go at the moment.

## Schemas & Collections

### Categories (Enum)

1) General
2) Medical
3) Insurance
4) Financial
5) Legal
6) Education
7) Full Stack
8) Front End
9) Back End
10) DevOps
11) Cyber Security
12) Freelance

### Work Status (Enum)

1) Nothing
2) Open to work
3) Hiring

### User Type (Enum)

1) Inactive
2) Active
3) Admin

## Users (Supabase generated)

More fields are generated, but only these are used.

- Unique ID
- Email address
- Name
- Avatar URL

## CVs

- Unique ID
- Document link
- User ID
- Upload date (long epoch time)
- Category IDs (from the categories Enum)
- Description
- Deleted
- Resolved

## Comments

- Unique ID
- Document ID
- Parent Comment ID (Null if root comment)
- User ID
- Last update (long epoch time)
- Upvotes (list of user IDs)
- Comment Data
- Resolved
- Deleted

## Profiles

- Unique ID (from the Users table)
- Full name (from the Users table)
- Avatar URL (from the Users table)
- Username (defaults to NULL, users can change theirs)
- Last update (long epoch time)
- Work status (users can change theirs according to the work status Enum)
- Work status categories (users can change theirs with data from the categories Enum)

## Profiles Perms

- Unique ID (from the Users table)
- User type (from the user type Enum)

### Admins/Whitelisted views

- User id from the perms table based on the user type (if admin, the id is in the admin, and if active or more, in whitelisted).

### Previews bucket

- A bucket of preview images from Google according to the Google Docs ID.
- Each file there is the first page of the CV with the name being the ID.

## Workflow Logic

### CV Previews

Each request to the feed will try to update the image in the bucket if that hash of the file changed.

The request will fetch the image from the bucket, save it to the cache, and render it along with the uploader's name, the relevant categories, and the upload date.

We have though of a way to improve the number of requests and conflicts that may occur. Cron jobs and other similar ideas are still though about.

### User Permissions

When a user signs in, he automatically receives the inactive type, which means he has no way of doing anything and he is always redirected to an "access denied" page.

An admin must change the type on the admin page/the Supabase console at the perms table.

The active role can do anything to themselves except for updating their role (they can only do things that are implemented in the code – if something is changeable and not referenced in any point in the code, it can't be updated).

Admins can do anything they like (change names, delete comments, delete CVs, etc). This is enforced on the DB using Row Level Security and on the backend.

## High Level Frontend Layout

A clear version of our draw.io is available at our repository.