Software Documentation

Popescu Victoria (popescu2014.pv@gmail.com)

This document describes the user guidelines for using (https://js-bootcamp21.www), the installation instructions and other source code specifications and explanations that are a must in developing a high quality software.

1. User Guidelines

1.1. Navbar and header

- Event name and main image;
- The route links to views of the website (Home and About us);
- The route links to the related social media accounts for more information;
- Real time data of the total number of the registered volunteers.

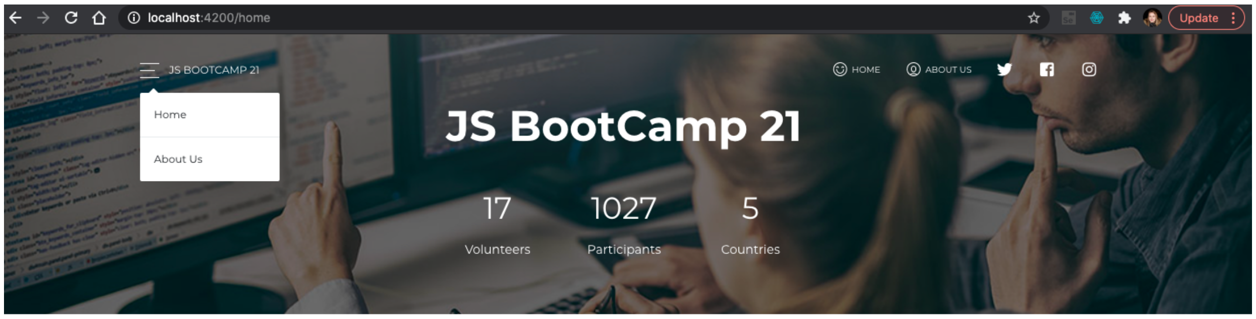
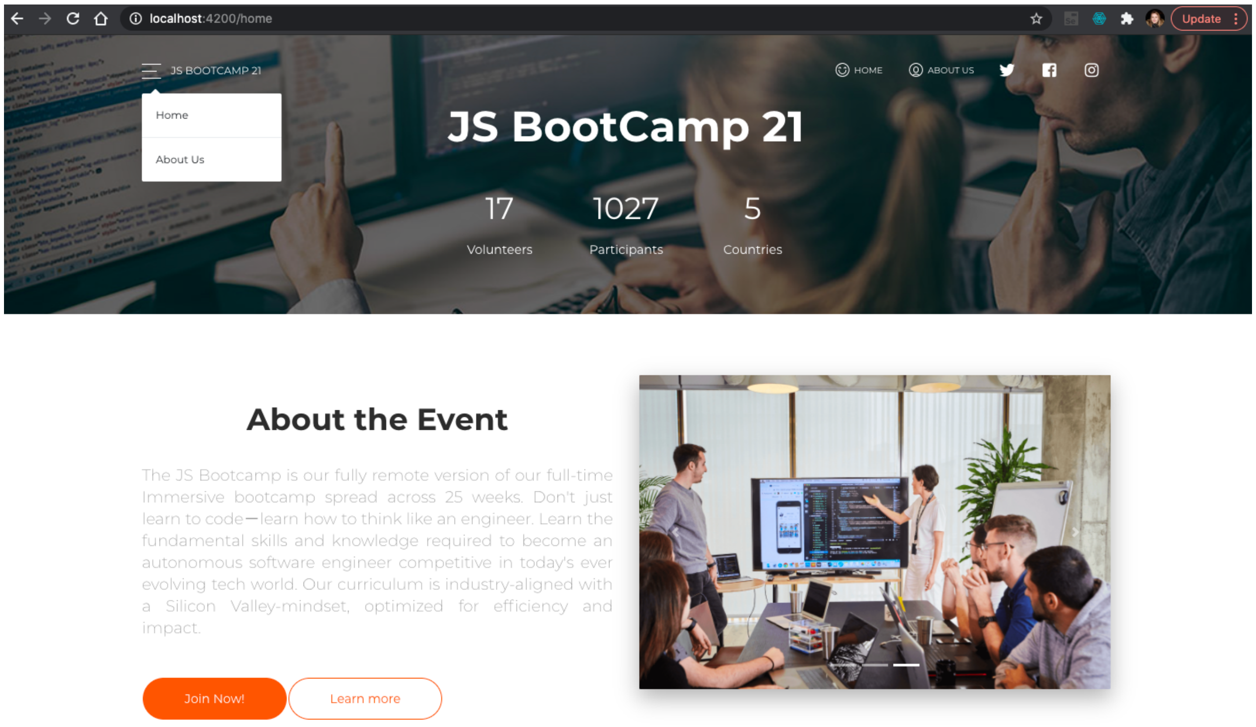


Figure 1 - Navbar and Header

1.2. Home page

The home page displays the main information regarding the event "JS Bootcamp 21", it offers a short description of the event in the "About the Event " section having on the right scrolling carousel of images taken at the previous edition. On the bottom of the "About the Event " section there are two buttons displayed, one of which opens the "Volunteering Form " named "Join us Now!" and the second button "Learn More" redirects to another webpage offering more information regarding the concept of a bootcamp. (*see figure.2)*



The JS Bootcamp is our fully remote version of our full-time Immersive bootcamp spread across 25 weeks. Don't just learn to code—learn how to think like an engineer. Learn the fundamental skills and knowledge required to become an autonomous software engineer competitive in today's ever evolving tech world. Our curriculum is industry-aligned with a Silicon Valley-mindset, optimized for efficiency and impact.

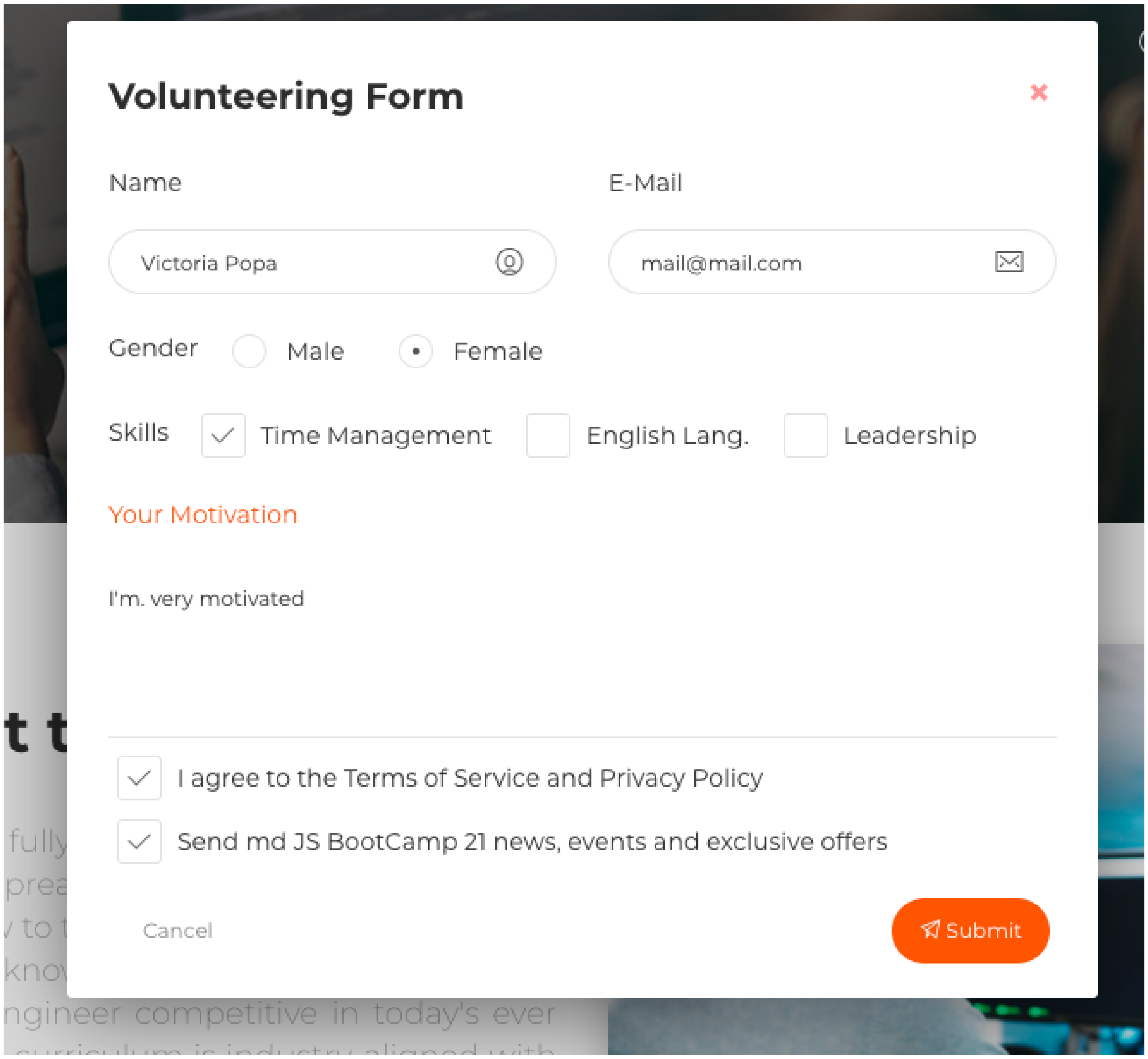
Figure 2 - Home page

1.3. Volunteering Form - modal

The registration form requires the user to insert their own details such as: name, email address, gender, skills and motivation to participate. The fields “name”, “email” and “agree to the Terms of service and Privacy Policy” are mandatory and apply validation constraints. In such cases where the inserted fields are not valid - a detailed warning is displayed (*see figure.3 and figure.4).*

After submitting the form details the user receives a confirmation to the email address that was inserted into the form. (*see figure.5)*

This section is followed by a “Contact us” view, where all the information regarding the event location and organisers’ contacts is accessible. (*see figure.6)*



Volunteering Form

Name: Victoria Popa

E-Mail: mail@mail.com

Gender: Male [✓ Female]

Skills: [✓ Time Management] English Lang. Leadership

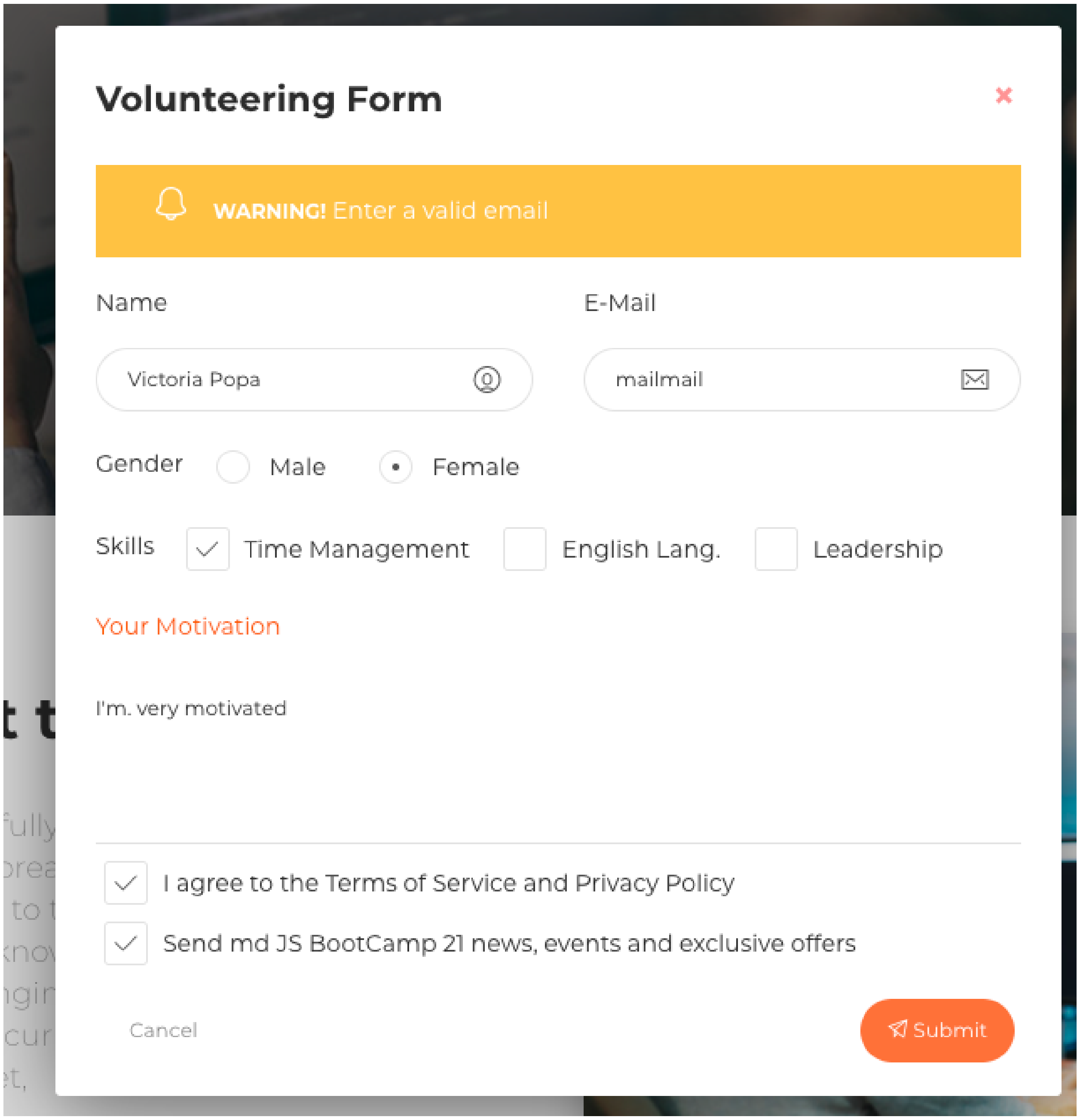
Your Motivation

I'm. very motivated

✓ I agree to the Terms of Service and Privacy Policy

✓ Send md JS BootCamp 21 news, events and exclusive offers

Figure 3 - Volunteering Form Modal



Volunteering Form

WARNING! Enter a valid email

Name	E-Mail
Victoria Popa	mailmail

Gender Male [✓ Female]

Skills [✓ Time Management] English Lang. Leadership

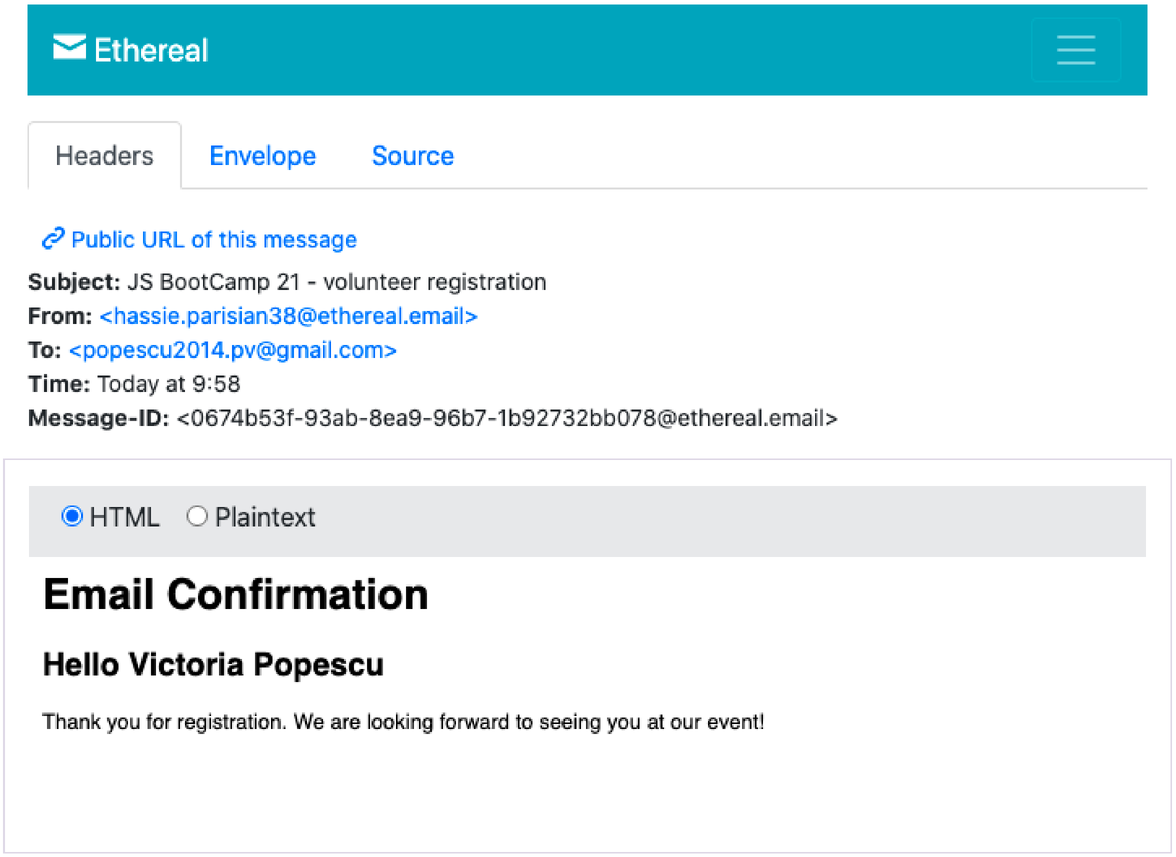
Your Motivation

I'm. very motivated

[✓] I agree to the Terms of Service and Privacy Policy

[✓] Send md JS BootCamp 21 news, events and exclusive offers

Figure 4 - Validation



Ethereal

Headers | Envelope | Source

Public URL of this message

Subject: JS BootCamp 21 - volunteer registration
From: <hassie.parisian38@ethereal.email>
To: <popescu2014.pv@gmail.com>
Time: Today at 9:58
Message-ID: <0674b53f-93ab-8ea9-96b7-1b92732bb078@ethereal.email>

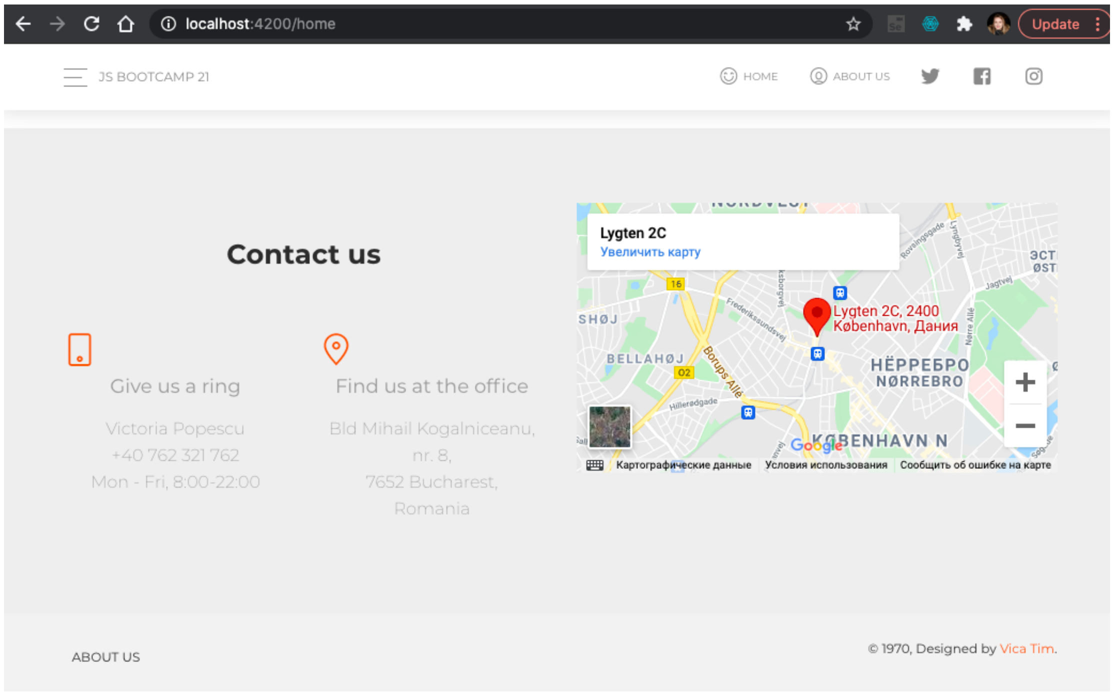
[✓ HTML] Plaintext

Email Confirmation

Hello Victoria Popescu

Thank you for registration. We are looking forward to seeing you at our event!

Figure 5 - Confirmation E-Mail



Contact us

Give us a ring

Victoria Popescu
+40 762 321 762
Mon - Fri, 8:00-22:00

Find us at the office

Bld Mihail Kogalniceanu, nr. 8,
7652 Bucharest,
Romania

ABOUT US

© 1970, Designed by Vica Tim.

Figure 6 - “Contact us” view

1.4. About us

In this section, there is a short static video that reflects the spirit of “JS BootCamp 21”. (*see figure.7)*

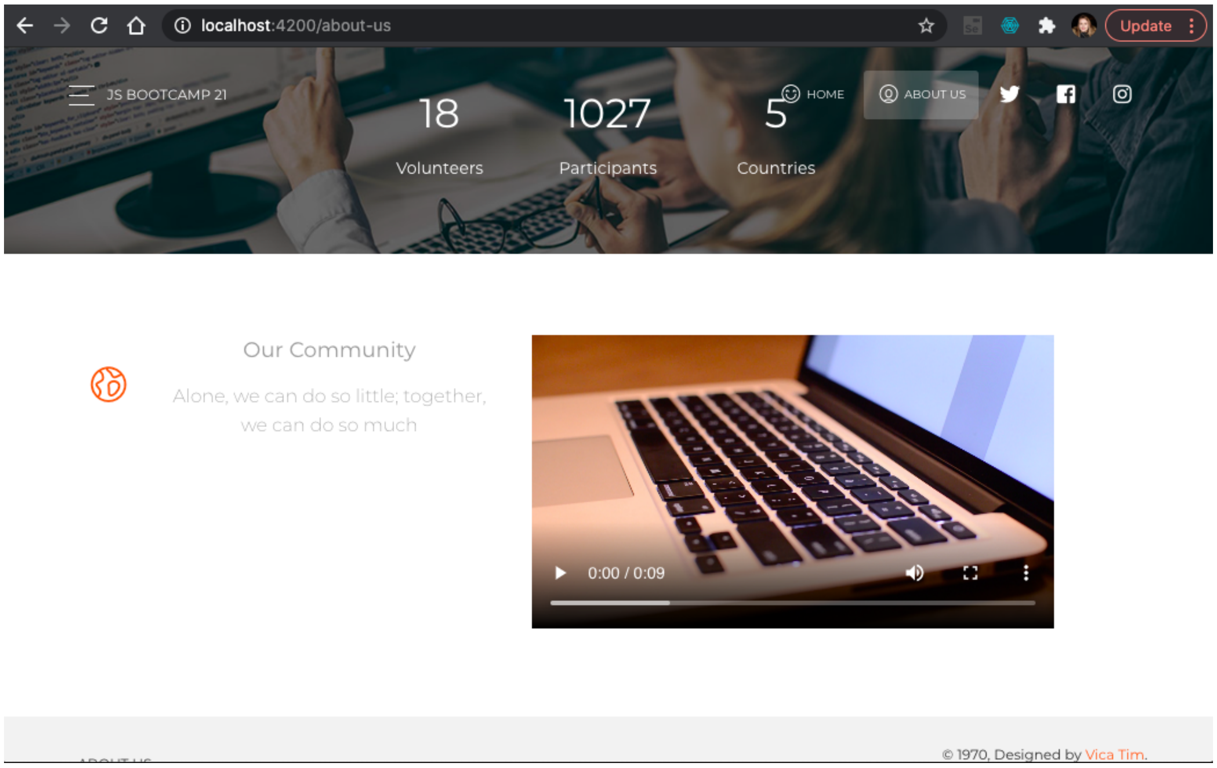


Figure 7 - “About us” view

2. Technical details

The application consists of two parts (client and server). The client side represents the GUI that the user interacts with and the BE that is mainly the Server, particularly the application that processes the requests coming from the client.

Framework	version	Used modules/packages
Angular (client side)	11.0.4	LayoutModule, NgbModule, FormsModule, RouterModule, AppRoutingModule, I18nModule,

		ReactiveFormsModule, AgmCoreModule.forRoot({ apiKey: 'AIzaSyDEAOXuWJY0GSPVdsfjXzLUwLo-3erB0kU' }), CommonModule
NodeJS (server side)	12.16.3	body-parser express nodemailer
Postgres	12	n/a

3. Installation
   3.1. STEP 1 - create the directory
   3.2. STEP 2 - clone the code source

FE: `git clone https://github.com/Vittorria/bootcamp-frontend.git`

BE: `git clone https://github.com/Vittorria/bootcamp_backend`

   3.3. STEP 3 - run npm install in both projects
   3.4. STEP 4 - create the config.json file in backend and add the following cofings:

```
{"NODE_EN":"DEVELOPMENT",
 "HOST":"127.0.0.1",
 "PORT":3333,
 "SSL_PORT":3443,
 "SSL_KEY":"/etc/nginx/ssl/server.key",
 "SSL_CRT":"/etc/nginx/ssl/server.crt",
 "DB_HOST":"localhost",
 "DB_PORT":5432,
 "DB_USER":"postgres",
 "DB_PASSWORD":"123123",
 "DB_NAME":"wiredelta",
 "API": "/api/v1",
 "SECRET": "bezkoder-secret-key",
 "USER": "hassie.parisian38@ethereal.email",
 "PASS": "mAvzyrsf1VzsHdc1kM"
}
```

3.5. STEP 5 - set the configs properties according to your needs;
3.6. STEP 6 - configure pg database, add its credential to config.json
3.7. STEP 7 - run the following queries to you db schema “public”

4.

```
CREATE DATABASE wiredelta
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    LC_COLLATE = 'C'
    LC_CTYPE = 'C'
    TABLESPACE = pg_default
    CONNECTION LIMIT = -1;
```

```
-- Table: public.volunteers

-- DROP TABLE public.volunteers;

CREATE TABLE public.volunteers
(
    name "char",
    id integer NOT NULL,
    gender "char",
    email "char",
    bl_time_mng boolean,
    bl_english boolean,
    bl_leadership boolean,
    comment "char",
    CONSTRAINT volunteers_pkey PRIMARY KEY (id)
)

TABLESPACE pg_default;

ALTER TABLE public.volunteers
    OWNER to postgres;
```

```
-- FUNCTION: public.f_add_new_volunteer(character varying, character varying,
character varying, boolean, boolean, boolean, character varying)

-- DROP FUNCTION public.f_add_new_volunteer(character varying, character varying,
character varying, boolean, boolean, boolean, character varying);

CREATE OR REPLACE FUNCTION public.f_add_new_volunteer(

       name character varying,

       email character varying,

       gender character varying,

       bl_time_mng boolean,
       bl_english boolean,

       bl_leadership boolean,

       comment character varying)

    RETURNS text

    LANGUAGE 'plpgsql'



    COST 100

    VOLATILE



AS $BODY$

DECLARE

message text;

begin

       message      := 'FAILED';



       IF COALESCE(name, '') = '' THEN

              message      := 'Data is empty';

              RETURN message;

       END IF;



              WITH t1 AS (SELECT MAX("id") cid

                            FROM public.volunteers)

              INSERT INTO public.volunteers (name, id, gender, email, bl_time_mng,
bl_english, bl_leadership, comment)

              SELECT name, (t1.cid + 1), gender, email, bl_time_mng, bl_english,
bl_leadership, comment

                     FROM t1;



              message := 'Added';

              Return message;
```

```
END



$BODY$;



ALTER FUNCTION public.f_add_new_volunteer(character varying, character varying,
character varying, boolean, boolean, boolean, character varying)

    OWNER TO postgres;
```

4.1. Run the projects