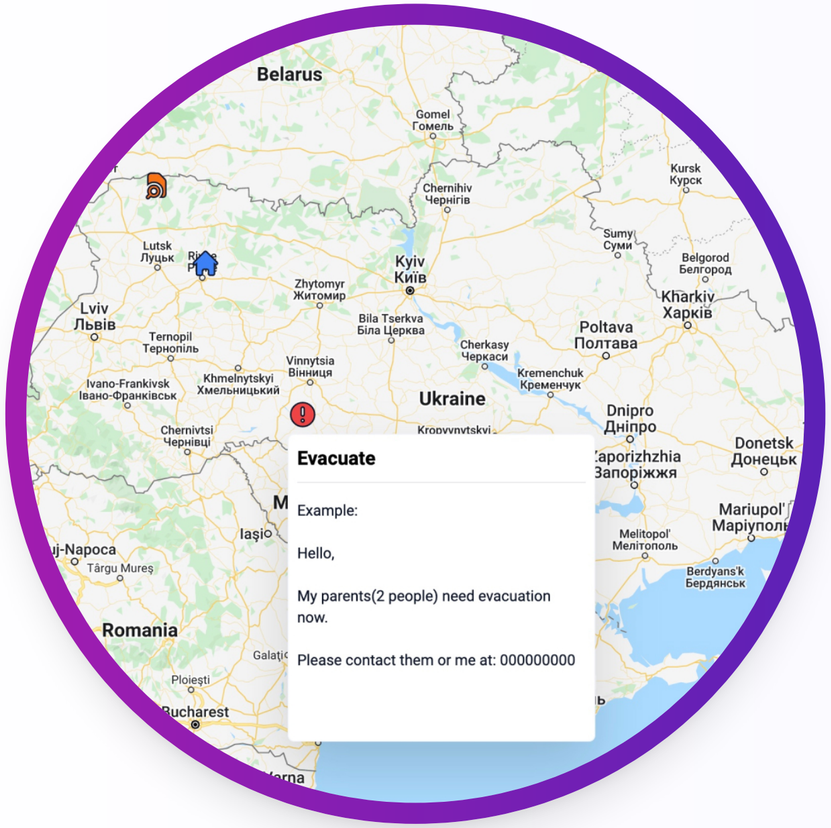


How I built with-humanity

Andrei Zgirvaci

with-humanity.vercel.app

@andrei_zgirvaci

Tech Stack

- Front-End:
  - Next.js (TypeScript)
  - Tailwind CSS, Headless UI, Heroicons
  - google-map-react
  - ga-4-react

Tech Stack

- Front-End:
  - Next.js (TypeScript)
  - Tailwind CSS, Headless UI, Heroicons
  - google-map-react
  - ga-4-react

- Back-End:
  - Firebase (Firestore)

Create Next.js project

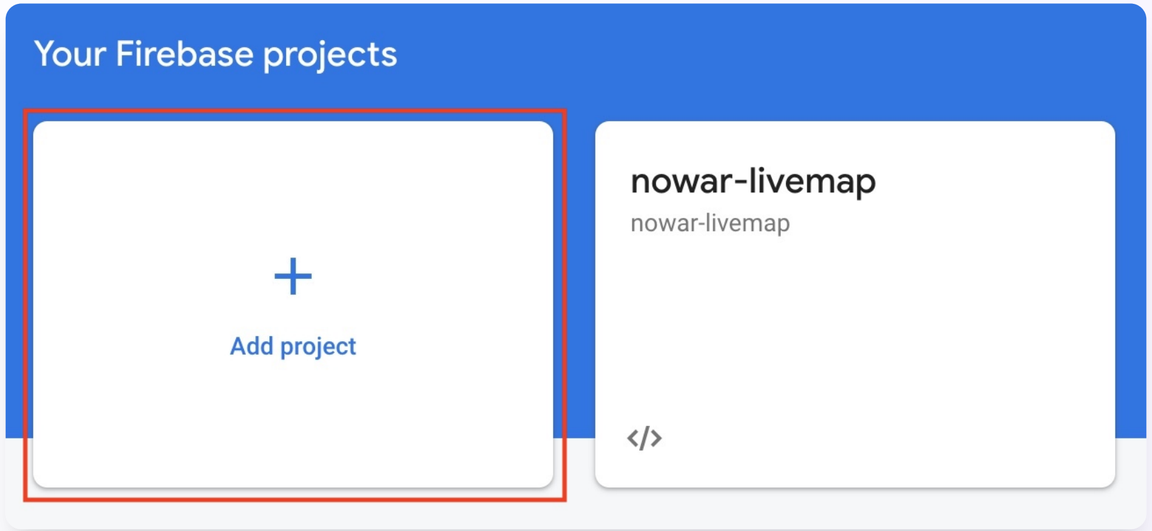
1. Create a new project in Next.js:

```
npx create-next-app@latest --ts
```

2. Add google-map-react & firebase:

```
npm add firebase google-map-react
```

3. Create Firebase Project:



4. Enable Google Maps APIs:

To use Google Maps Platform, you must enable the APIs or SDKs you plan to use with your project.

Note that some integrations require you to enable multiple APIs/SDKs. If you are not sure which APIs or SDKs to enable, try using the API Picker, or consult the documentation for the API/SDK you want to use.

To enable one or more APIs or SDKs:

Console | Cloud SDK

1. See the Google Maps Platform APIs and SDKs that you can enable by going to the Maps API Library page in the Cloud Console:

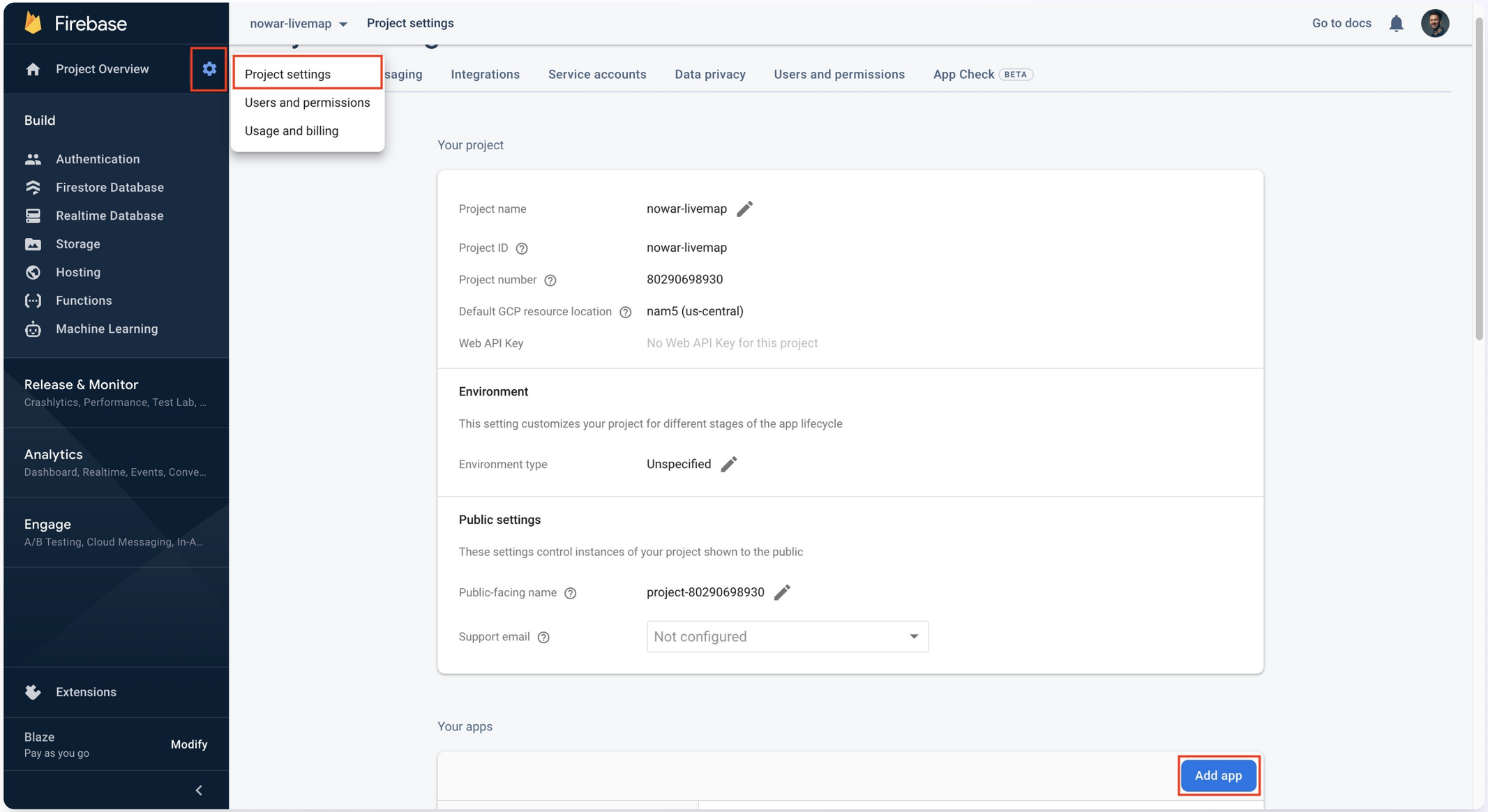
   Go to the **Maps API Library page**

2. Click the API or SDK you want to enable.
   - If the button says **ENABLE**, click the button to enable the API or SDK.
   - If the button says **MANAGE**, the API or SDK is already enabled and you don't need to do anything further.
   - Clicking either button will display the dashboard for the API or SDK. (Click the **DISABLE** button to remove the API or SDK from this project.)

* Make sure that billing is enabled for your Cloud project.

* *Google Cloud offers a $0.00 charge trial. The trial expires at either end of 90 days or after the account has accrued $300 worth of charges, whichever comes first.*

5. Create Firebase Web App:



6. Copy App Credentials & Initialize Firebase:

```
// pages/_app.tsx
import { initializeApp } from 'firebase/app';

const firebaseConfig = {
  apiKey: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
  authDomain: 'nowar-livemap.firebaseapp.com',
  projectId: 'nowar-livemap',
  storageBucket: 'nowar-livemap.appspot.com',
  messagingSenderId: '80290698930',
  appId: '1:80290698930:web:60de91103d9e5e44061bab',
  measurementId: 'G-E1R6X235XR',
};

initializeApp(firebaseConfig);

/* ... */
```

7. Copy `apiKey` and add it as an environment variable:

```
// .env.local
NEXT_PUBLIC_FIREBASE_API_KEY=****************************************
```

8. Add Google Map component:

```
// pages/index.tsx
const Home = () => {
  const defaultProps = {
    center: {
      lat: 50.46372175476692, // kiev
      lng: 30.529234424870925, // kiev
    },
    zoom: 6,
  };

  return (
    <main className="flex flex-col h-full">
      <section className="flex flex-1">
        <GoogleMapReact
          bootstrapURLKeys={{
            key: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
          }}
          defaultCenter={defaultProps.center}
          defaultZoom={defaultProps.zoom}
        />
      </section>
    </main>
  );
}
```

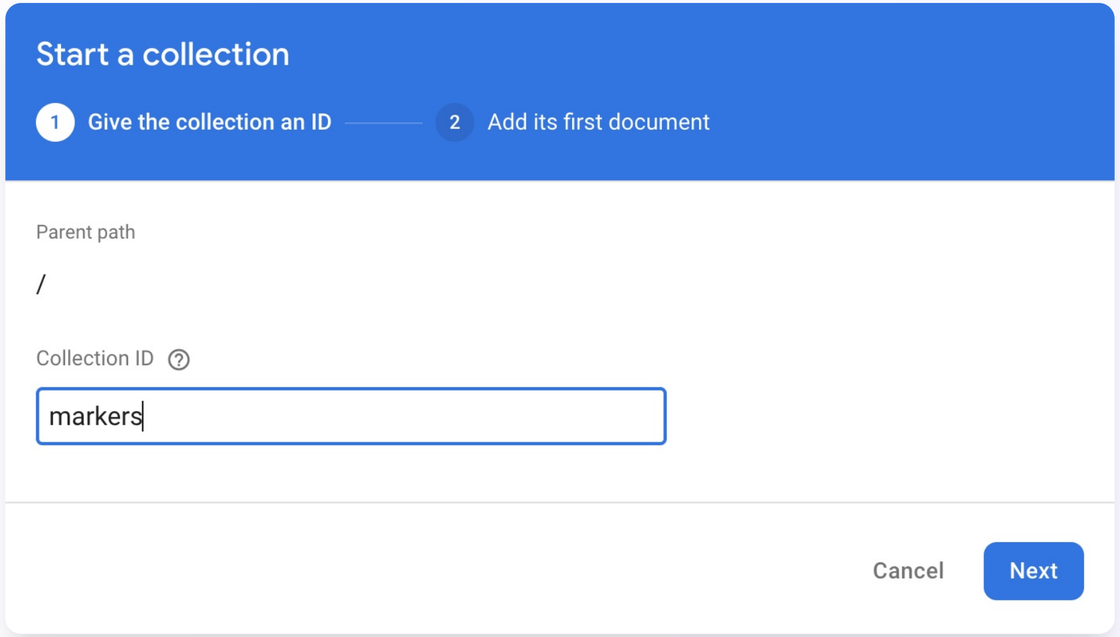
8. Add Google Map component:

```
// pages/index.tsx
const Home = () => {
  const defaultProps = {
    center: {
      lat: 50.46372175476692, // kiev
      lng: 30.529234424870925, // kiev
    },
    zoom: 6,
  };

  return (
    <main className="flex flex-col h-full">
      <section className="flex flex-1">
        <GoogleMapReact
          bootstrapURLKeys={{
            key: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
          }}
          defaultCenter={defaultProps.center}
          defaultZoom={defaultProps.zoom}
        />
      </section>
    </main>
  );
}
```

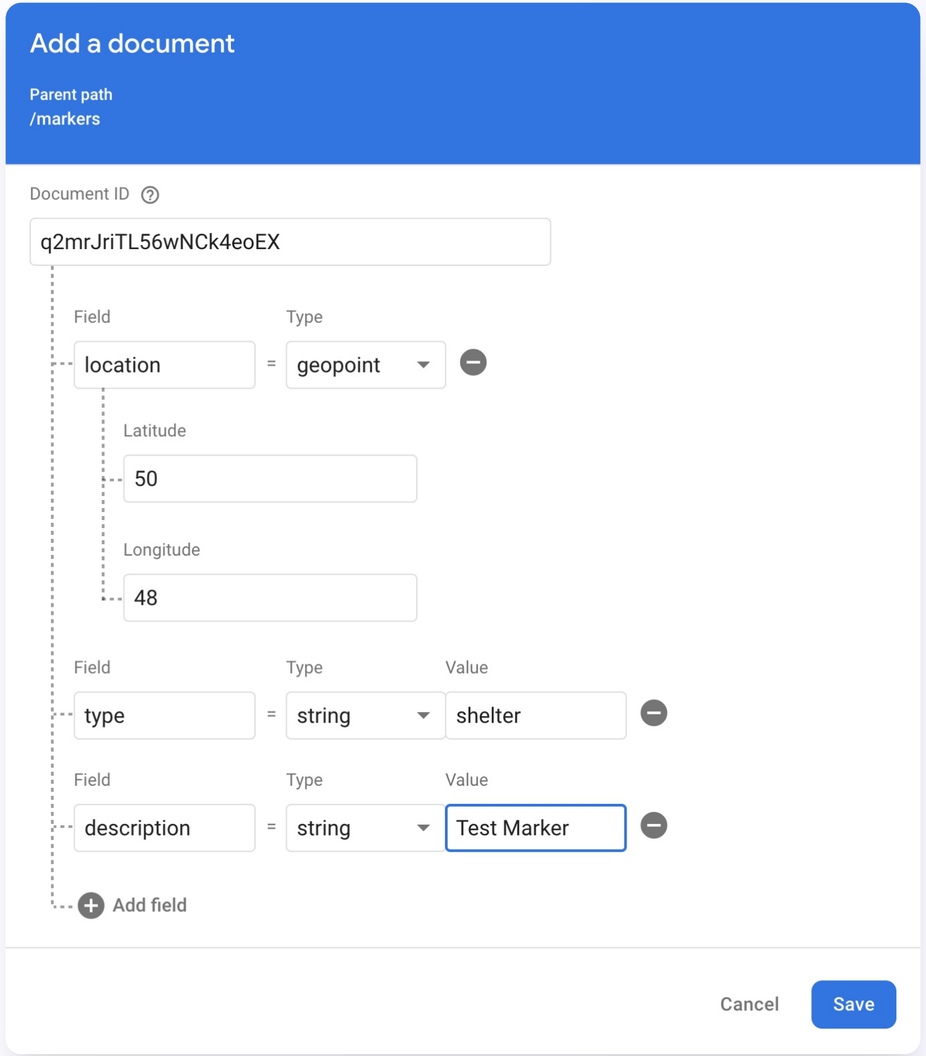
8. Add Google Map component:

```
// pages/index.tsx
const Home = () => {
  const defaultProps = {
    center: {
      lat: 50.46372175476692, // kiev
      lng: 30.529234424870925, // kiev
    },
    zoom: 6,
  };

  return (
    <main className="flex flex-col h-full">
      <section className="flex flex-1">
        <GoogleMapReact
          bootstrapURLKeys={{
            key: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
          }}
          defaultCenter={defaultProps.center}
          defaultZoom={defaultProps.zoom}
        />
      </section>
    </main>
  );
}
```

9. Create a new collection in Firestore:



10. Create a test document in `markers` collection:



11. Get Markers from Firestore:

```
// lib/markers.ts
import { collection, getDocs, getFirestore } from 'firebase/firestore';

const db = getFirestore();

export default async function getMarkers() {
  const querySnapshot = await getDocs(collection(db, 'markers'));

  const markers = [];

  querySnapshot.forEach((doc) => {
    const data = doc.data();

    markers.push({
      id: doc.id,
      lat: data.location._lat,
      lng: data.location._long,
      type: data.type,
      description: data.description,
    });
  });

  return markers;
}
```

11. Get Markers from Firestore:

```
// lib/markers.ts
import { collection, getDocs, getFirestore } from 'firebase/firestore';

const db = getFirestore();

export default async function getMarkers() {
  const querySnapshot = await getDocs(collection(db, 'markers'));

  const markers = [];

  querySnapshot.forEach((doc) => {
    const data = doc.data();

    markers.push({
      id: doc.id,
      lat: data.location._lat,
      lng: data.location._long,
      type: data.type,
      description: data.description,
    });
  });

  return markers;
}
```

11. Get Markers from Firestore:

```
// lib/markers.ts
import { collection, getDocs, getFirestore } from 'firebase/firestore';

const db = getFirestore();

export default async function getMarkers() {
  const querySnapshot = await getDocs(collection(db, 'markers'));

  const markers = [];

  querySnapshot.forEach((doc) => {
    const data = doc.data();

    markers.push({
      id: doc.id,
      lat: data.location._lat,
      lng: data.location._long,
      type: data.type,
      description: data.description,
    });
  });

  return markers;
}
```

11. Get Markers from Firestore:

```
// lib/markers.ts
import { collection, getDocs, getFirestore } from 'firebase/firestore';

const db = getFirestore();

export default async function getMarkers() {
  const querySnapshot = await getDocs(collection(db, 'markers'));

  const markers = [];

  querySnapshot.forEach((doc) => {
    const data = doc.data();

    markers.push({
      id: doc.id,
      lat: data.location._lat,
      lng: data.location._long,
      type: data.type,
      description: data.description,
    });
  });

  return markers;
}
```

11. Get Markers from Firestore:

```
// lib/markers.ts
import { collection, getDocs, getFirestore } from 'firebase/firestore';

const db = getFirestore();

export default async function getMarkers() {
  const querySnapshot = await getDocs(collection(db, 'markers'));

  const markers = [];

  querySnapshot.forEach((doc) => {
    const data = doc.data();

    markers.push({
      id: doc.id,
      lat: data.location._lat,
      lng: data.location._long,
      type: data.type,
      description: data.description,
    });
  });

  return markers;
}
```

12. Render Markers in Maps:

```
// pages/index.tsx
import getMarkers from 'lib/markers';
import Marker from 'components/Marker';

export async function getServerSideProps() {
  const markers = await getMarkers();

  return {
    props: { markers },
  };
}


...
<GoogleMapReact ...>
  {markers.map((marker) => (
    <Marker key={marker.id} {...marker} />
  ))}
<GoogleMapReact/>
...
```

12. Render Markers in Maps:

```
// pages/index.tsx
import getMarkers from 'lib/markers';
import Marker from 'components/Marker';

export async function getServerSideProps() {
  const markers = await getMarkers();

  return {
    props: { markers },
  };
}


...
<GoogleMapReact ...>
  {markers.map((marker) => (
    <Marker key={marker.id} {...marker} />
  ))}
<GoogleMapReact/>
...
```

13. Create `Marker` Component:

```
// components/Marker.tsx
import PopoverComponent from 'components/Popover';

export default function Marker({ $hover, description }) {
  return (
    <Popover>
      <Icon className={`h-7 w-7 text-blue-500`} />

      {$hover && (
        <Popover.Panel static>
          <div className="w-72 h-72 bg-white p-2">
            <PopoverComponent description={description} />
          </div>
        </Popover.Panel>
      )}
    </Popover>
  );
}
```

13. Create `Marker` Component:

```
// components/Marker.tsx
import PopoverComponent from 'components/Popover';

export default function Marker({ $hover, description }) {
  return (
    <Popover>
      <Icon className={`h-7 w-7 text-blue-500`} />

      {$hover && (
        <Popover.Panel static>
          <div className="w-72 h-72 bg-white p-2">
            <PopoverComponent description={description} />
          </div>
        </Popover.Panel>
      )}
    </Popover>
  );
}
```

13. Create `Marker` Component:

```
// components/Marker.tsx
import PopoverComponent from 'components/Popover';

export default function Marker({ $hover, description }) {
  return (
    <Popover>
      <Icon className={`h-7 w-7 text-blue-500`} />

      {$hover && (
        <Popover.Panel static>
          <div className="w-72 h-72 bg-white p-2">
            <PopoverComponent description={description} />
          </div>
        </Popover.Panel>
      )}
    </Popover>
  );
}
```

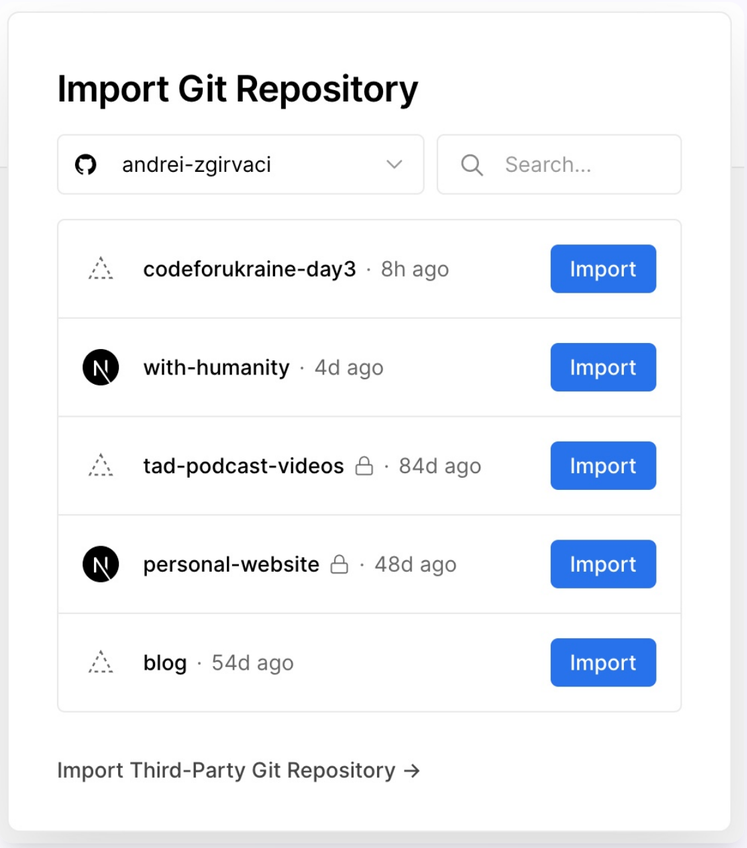
13. Create `Marker` Component:

```tsx
// components/Marker.tsx
import PopoverComponent from 'components/Popover';

export default function Marker({ $hover, description }) {
  return (
    <Popover>
      <Icon className={`h-7 w-7 text-blue-500`} />

      {$hover && (
        <Popover.Panel static>
          <div className="w-72 h-72 bg-white p-2">
            <PopoverComponent description={description} />
          </div>
        </Popover.Panel>
      )}
    </Popover>
  );
}
```

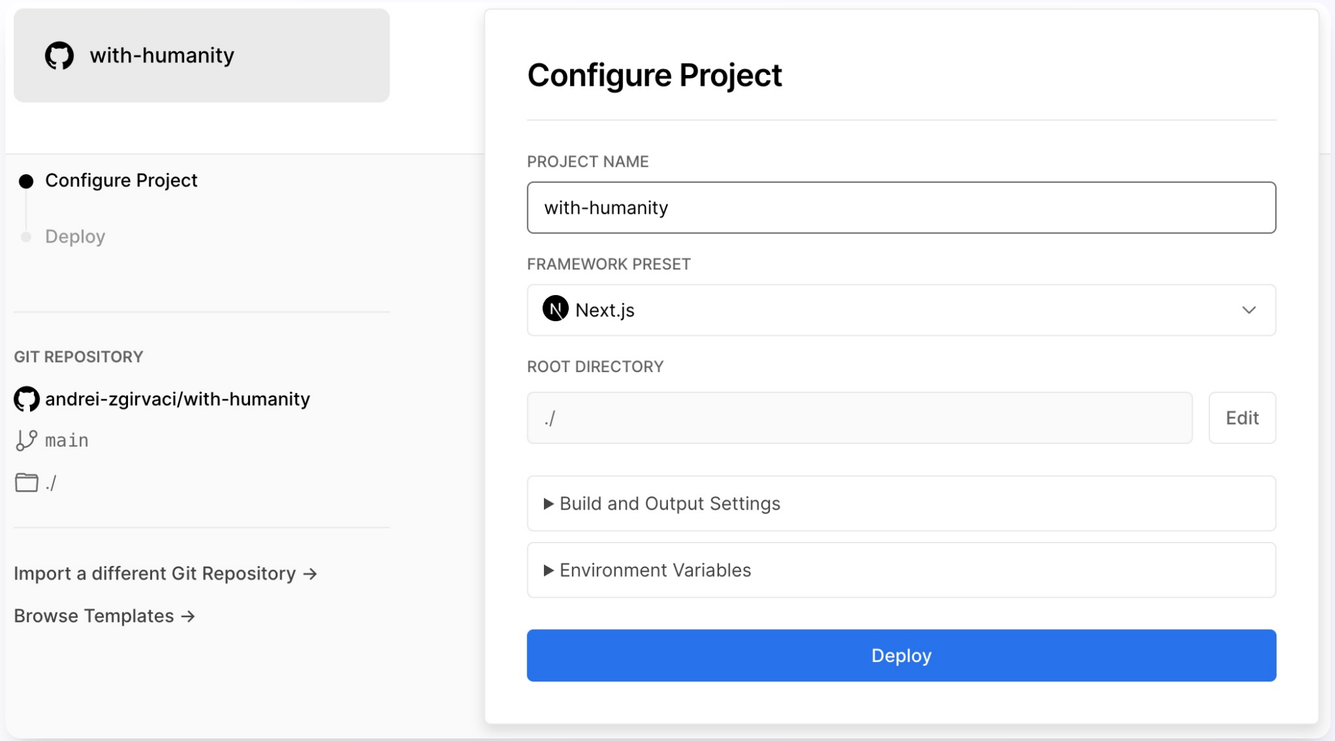
14. Create `Popover` Component:

```
// components/Popover.tsx
export default function Popover({ type, description }) {
  return (
    <>
      <p className="mb-2 text-lg font-bold">{type.toUpperCase()}</p>

      <hr className="mb-4" />

      <div className="text-sm text-slate-900">
        <p>{description}</p>
      </div>
    </>
  );
}
```

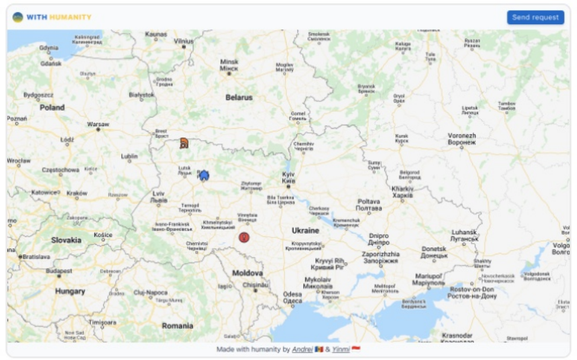
15. Deploy your application to Vercel (1)



16. Deploy your application to Vercel (2):



17. Deploy your application to Vercel (3):



STATUS
Ready

ENVIRONMENT
Production

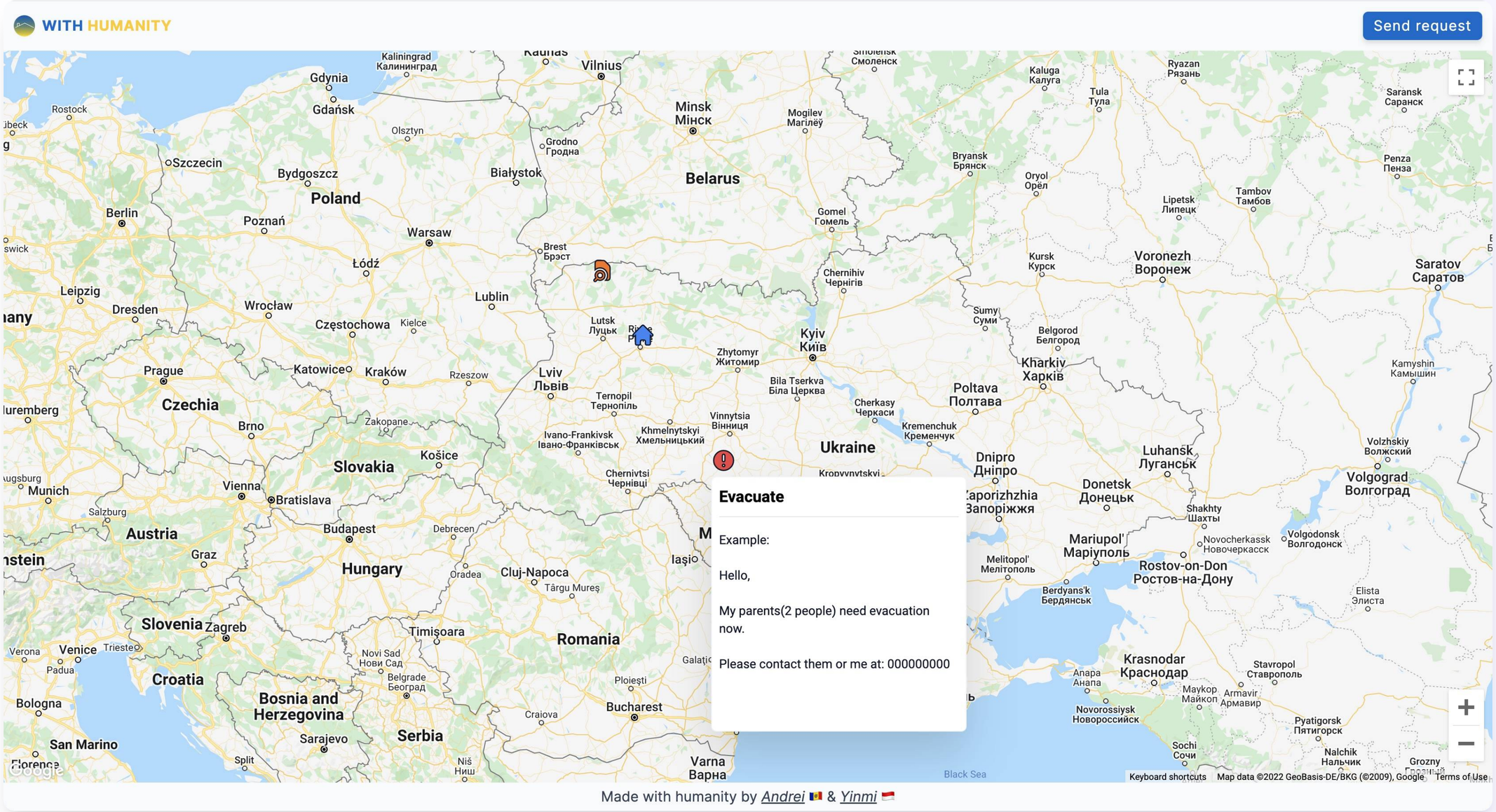
DURATION
24s (2d ago)

Visit

DOMAINS
with-humanity.vercel.app +3

BRANCH
main
fa79fd1 — refactor

Demo


WITH HUMANITY
Send request
Evacuate
Example:
Hello,
My parents(2 people) need evacuation now.
Please contact them or me at: 000000000
Made with humanity by Andrei 🇲🇩 & Yinmi 🇮🇩

That’s it! Thank you!

Q&A

website: https://with-humanity.vercel.app
github: https://github.com/andrei-zgirvaci/with-humanity