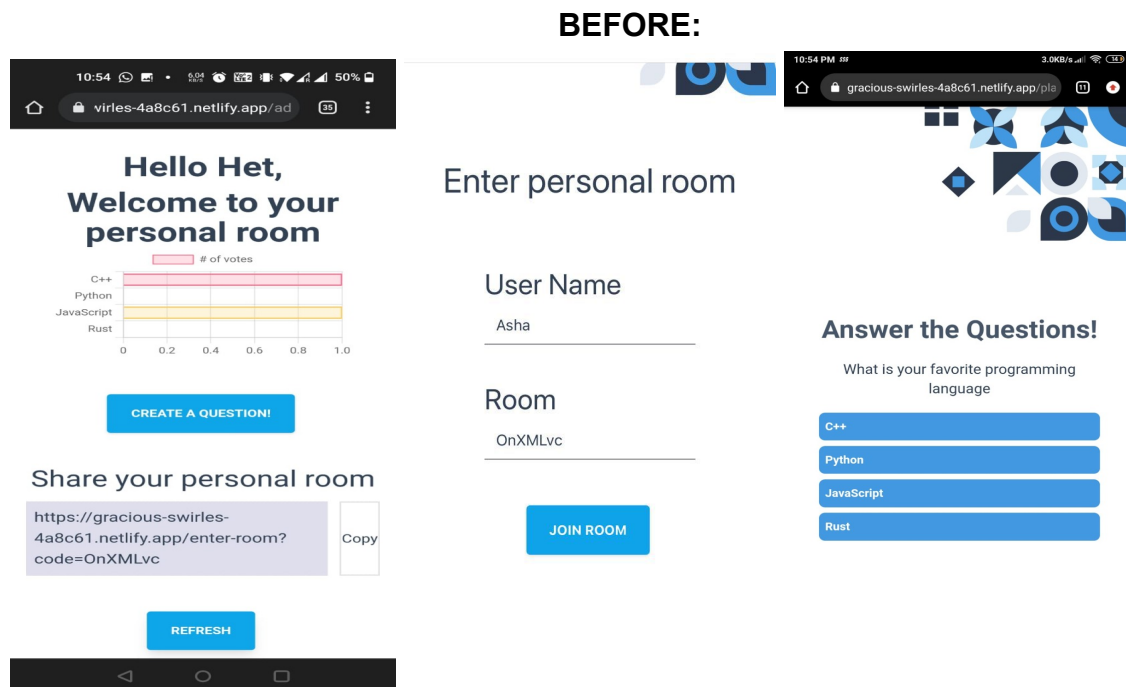


As part of our project-2 work, we have improved this project significantly from where it was left by the previous team. We have made multiple changes to the repo by changing the whole UI and improvements in architecture. Below are some of the major enhancements achieved by our team.

UI Enhancements

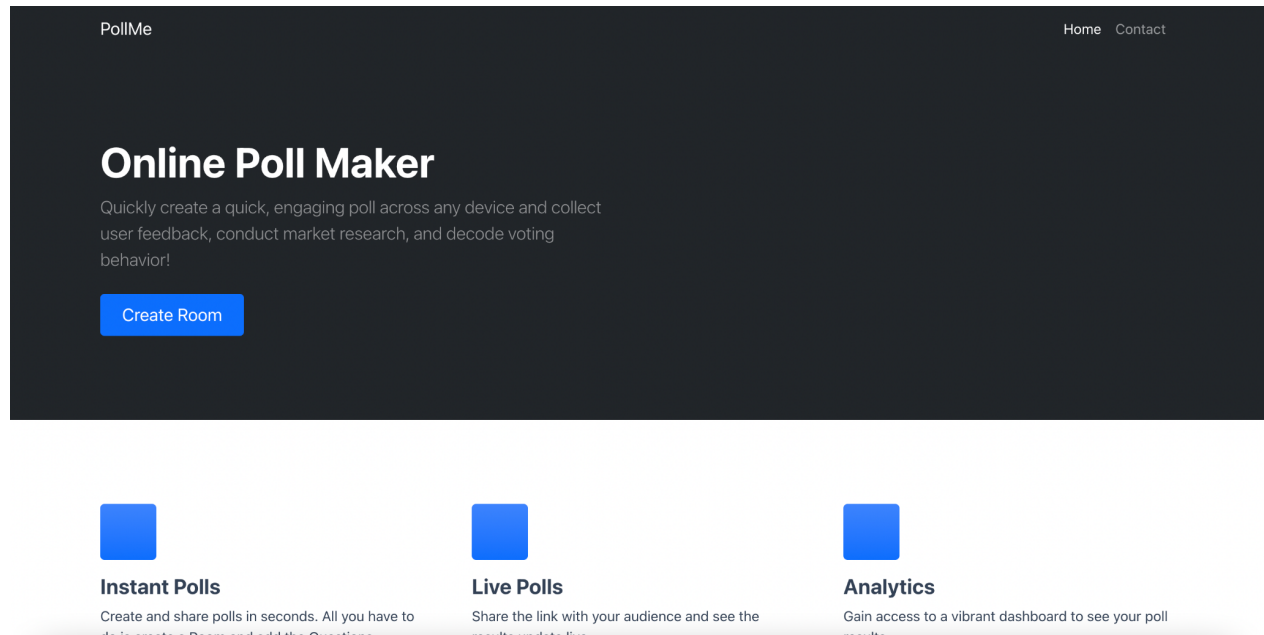
1. **Home Page** : As a part of the enhancement, we added a homepage for our website. Earlier the poll me home page was missing for the webapp. We added a new homepage with a lot of functionalities such as contact details, enter room button and with basic promotional information about polly.
2. **Create Room Page**: We have modified the create room page to look more integrated and user-friendly. The UI changes are seen by these snapshots.
3. **Analytics**: We show Analytics inside the room to make it more convenient for the user to see the data while being inside the room.
4. **Survey page**: The layout of this page has become responsive rather than static as designed earlier.

All UI changes have been highlighted via the following comparison:

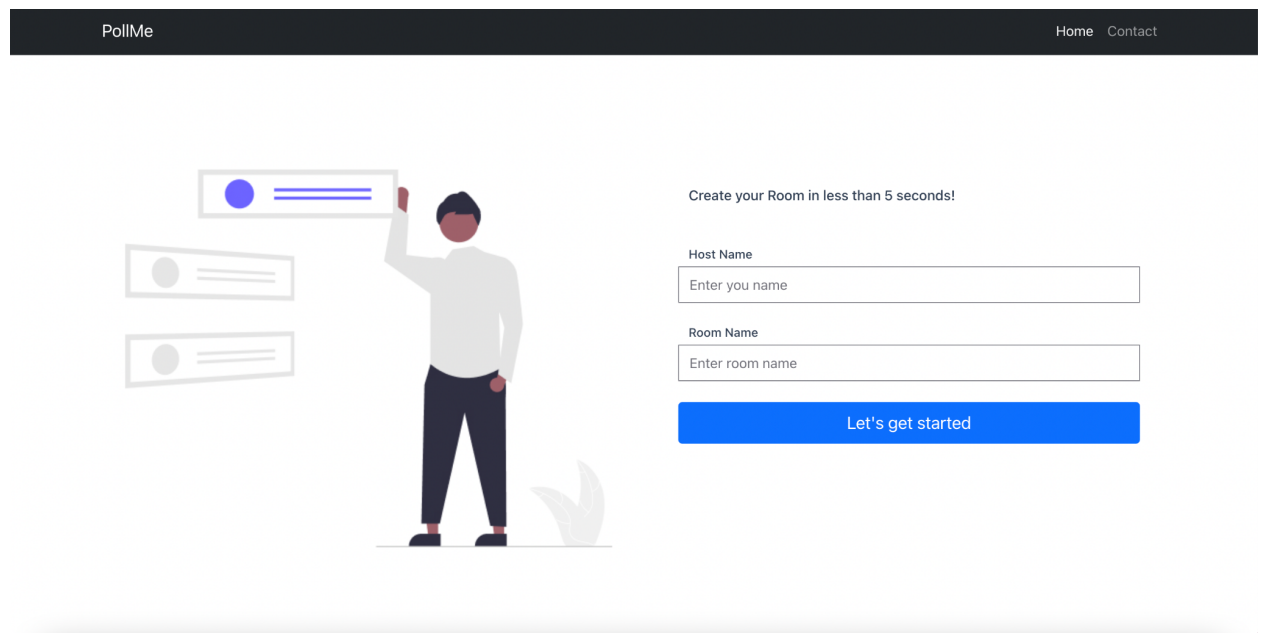


AFTER:

1. Home Page:



2. Create Room:



3. Room Page:

PolliMe

HomeContact

Click on the button below to add another question to your room

Create a question!

Share your personal room

http://localhost:3001/enter-room?code=0nY7ZIM

Copy

Refresh

Analytics

of votes

Bridgerton	1.0
Squid Game	0.0
Peaky Blinders	0.0
The Office	0.0

4. Enter Room for audiences:

PolliMe

HomeContact

Join your Room to take the survey Poll!


Player Name

Player

Room Key

0nY7ZIM

Join Room



Architecture Enhancements:

We made the program quite modular and changed the way the webpage was functioning. The changes are highlighted below:

Use of BootStrap in frontend:

To make the UI better in terms of user experience and extendibility, we used bootstrap in our frontend stack. There are many advantages of using bootstrap which includes,

- Seeing fewer cross browser bugs. Compatible with most browsers.
- A consistent framework that supports most major of all browsers and CSS compatibility fixes.
- Bootstrap is lightweight and customizable.
- It provides responsive structures and styles.
- Using bootstrap came with the advantage of several JavaScript plugins.
- Also, bootstrap is progressive framework with good documentation and community support.

```
48
49 @charset "UTF-8";
50
51 /*!
52 * Bootstrap v5.1.3 (https://getbootstrap.com/)
53 * Licensed under MIT (https://github.com/twbs/bootstrap/blob/main/LICENSE)
54 */
55 :root {
56   --bs-blue: #0d6efd;
57   --bs-indigo: #6610f2;
58   --bs-purple: #6f42c1;
59   --bs-pink: #d63384;
60   --bs-red: #dc3545;
61   --bs-orange: #fd7e14;
62   --bs-yellow: #ffc107;
63   --bs-green: #198754;
64   --bs-teal: #20c997;
65   --bs-cyan: #0dcaf0;
66   --bs-white: #fff;
67   --bs-gray: #6c757d;
68   --bs-gray-dark: #343a40;
69   --bs-gray-100: #f8f9fa;
70   --bs-gray-200: #e9ecef;
71   --bs-gray-300: #dee2e6;
72   --bs-gray-400: #ced4da;
```

Centralized Database:

We changed the whole database structure of the project to easily maintain the queries in a centralized structure. Before, Each serverless function instance that runs required it's own database connection. This can cause issues at scale as database connections are often quite limited. There were several methods which we tried to implement. Some of them are:

- Each serverless function would push their intent into a queue when counting votes. A separate consumer lambda could then aggregate these votes and count them together
- Another solution could be to use database connection pools to manage and preserve connections

We made the database access centralized so that all the rooms/player will write on one single data instance.

Correct Answer Support:

Now for quiz typed surveys, users can select the correct choice for the question, And when users answer the question, it shows if their selected answer is correct or incorrect. If highlighted in dark, the answer is correct, otherwise not.

When do US celebrate it's Independence day ?

Aug 3rd

July 4th

Never

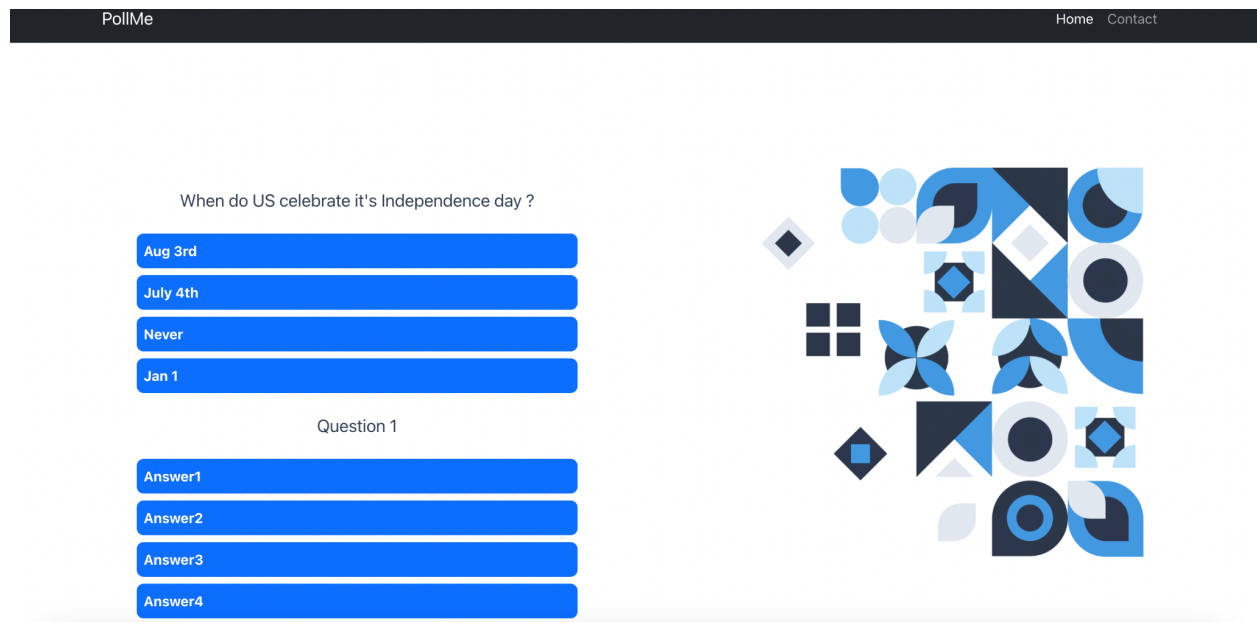
Jan 1

Reduced API calls to backend:

We modified the logic of our program code to make less calls to the backend API. Which means the program makes much fewer calls while handling the html queries. Which in turn reduces the system load. This enhancement has significantly improved the response time of the webpage actions and restricted unnecessary resource usage.

Dynamic loading of the questions in a room:

Users can see the survey questions dynamically as soon as they are uploaded by the room admin. Once the room admin chooses to publish the question, the player/users sees those questions on their screen immediately. Then they can answer all the questions simultaneously.




Deployment Enhancement

We made the tedious process of deploying the program dependencies, connecting to the databases etc very simple. By using dockerization, we made it possible to deploy the whole software in a single command.

Before:

Users had to download all kinds of software dependencies manually via different commands. It is shown below:

 **Local development setup**

Backend

- Switch to backend directory

```
cd backend
```
- Install dependencies

```
npm i
```
- Create and populate `.env` file from the `env.example` template
- Generate prisma client

```
npx prisma generate
```
- Start local serverless API

```
npx serverless offline
```

Performing database changes(Refer [prisma docs](#))

- Make your schema changes in the `prisma/schema.prisma`
- To create a migration, use the command

```
npx prisma migrate dev --name <migration_name>
```

Frontend

- Switch to frontend directory

```
cd frontend
```
- Install dependencies

```
npm install
```
- Create and populate `.env` file from the `env.example` template
- Start local serverless API

```
npm run start
```

AFTER:

By using docker we could deploy the whole system in one single command :

“Docker compose up”

The above single command set up the whole environment and downloads the dependencies to make it run seamlessly on any platform.

The advantages of using dockerization is:

- Portability: Can be deployed on any kind of platform seamlessly.
- Performance: faster to create, quicker to start
- Isolation: It maintains the software requirements throughout the development stages.
- Scalability
You can quickly create new containers if demand for your applications requires them.