Week-5. Write a program to create a voting application using React JS

```
CREATE
OR REPLACE VIEW "public"."poll_results" AS
SELECT
  poll.id AS poll_id,
o.option_id,
  count(*) AS votes
FROM
  (
    (
      SELECT
vote.option_id,
option.poll_id,
        option.text
      FROM
        (
vote
          LEFT JOIN option ON ((option.id = vote.option_id))
        )
    ) o
    LEFT JOIN poll ON ((poll.id = o.poll_id))
  )
GROUP BY
  poll.question,
o.option_id,   poll.id;
CREATE
OR REPLACE VIEW "public"."online_users" AS
SELECT
  count(*) AS count
FROM
  "user"
WHERE
  (
    "user".last_seen_at > (now() - '00:00:15' :: interval)
  );
```

```
import { ApolloClient, HttpLink, InMemoryCache, split } from "@apollo/client";
import { GraphQLWsLink } from '@apollo/client/link/subscriptions'; import {
createClient } from "graphql-ws";
```

```
import { getMainDefinition } from "@apollo/client/utilities"; const
GRAPHQL_ENDPOINT = "realtime-poll-example.hasura.app";

const scheme = (proto) =>
  window.location.protocol === "https:" ? `${proto}s` : proto;

const wsURI = `${scheme("ws")}://${GRAPHQL_ENDPOINT}/v1/graphql`;
const httpURL = `${scheme("https")}://${GRAPHQL_ENDPOINT}/v1/graphql`;
const splitter = ({ query }) => {
  const { kind, operation } = getMainDefinition(query) || {};
const isSubscription =
    kind === "OperationDefinition" && operation === "subscription";
return isSubscription;
};
const cache = new InMemoryCache();
const options = { reconnect: true };

const wsLink = new GraphQLWsLink(createClient({ url: wsURI,
connectionParams: { options } })); const httpLink
= new HttpLink({ uri: httpURL }); const link =
split(splitter, wsLink, httpLink);
const client = new ApolloClient({ link, cache });
```

output: