1. Start the container
2. Create .env file

```
PORT=5001
DB_USER="postgres"
DB_HOST="localhost"
DB_DATABASE="express-crud"
DB_PORT=5432
DB_PASSWORD="postgres@123"
```

3. Install the following packages

```
  "cors": "^2.8.5",
    "dotenv": "^16.4.7",
    "express": "^4.21.2",
    "nodemon": "^3.1.9",
    "pg": "^8.13.3"
```

4. Create db.js as the config file

```
import pg from "pg";
const { Pool } = pg;
dotenv.config();

const pool = new Pool({
  user: process.env.DB_USER,
  host: process.env.DB_HOST,
  database: process.env.DB_DATABASE,
  password: process.env.DB_PASSWORD,
  port: process.env.DB_PORT,
});

pool.on("connect", () => {
  console.log("Pool connection established with
Database");
});

export default pool;
```

5. Create index.js

```
dotenv.config();
const app = express();
```

```
const port = process.env.PORT || 3001;

app.use(express.json());
app.use(cors());

//Routes
app.use("/api", userRoutes);

//Create table
createUserTable();

//listen
app.listen(port, () => {
  console.log(`Server is running on ${port}`);
});
```

6. createUserTable

```
export const createUserTable = async () => {
  const query = `CREATE TABLE IF NOT EXISTS users(
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  email VARCHAR(255) NOT NULL UNIQUE,
  created_at TIMESTAMP DEFAULT NOW()
)
  `;
  try {
    await pool.query(query);
    console.log("User Table created if not exists");
  } catch (error) {
    console.log("Error creating Table");
  }
};
```

7. Create userRoutes

```
import express from "express";
const router = express.Router();

router.post("/user", createUser);
```

```
router.get("/user", getAllUsers);
router.get("/user/:id", getUserById);
router.put("/user/:id", updateUser);
router.delete("/user/:id", deleteUser);

export default router;
```

7.create controllers userController.js

```
export const createUser = async (req, res, next) => {
  const { name, email } = req.body;
  try {
    const newUser = await createUserService(name, email);
    handleResponse(res, 201, "User created Successfully",
newUser);
  } catch (err) {
    next(err);
  }
};
```

8 . In this controller instead of using service use the functions

```
 const users = await pool.query("SELECT * FROM users");
```

```
 const user = await pool.query("SELECT * FROM users WHERE
id=$1", [id]);
```

```
const newUser = await pool.query(
    "INSERT INTO users (name,email) VALUES($1,$2)
RETURNING *"
  );
```

```
const updateUser = await pool.query(
    "UPDATE users set name=$2, email=$3 WHERE id=$1
RETURNING *",
    [id, name, email]
  );
```

```
 const deleteUser = await pool.query(
    "DELETE from users WHERE id=$1 RETURNING *",
```

```
    [id]
);
```