

Building Cloud Native Applications with Choreo

Training Objective:

In this training, you will learn to deploy a full stack app in Choreo.

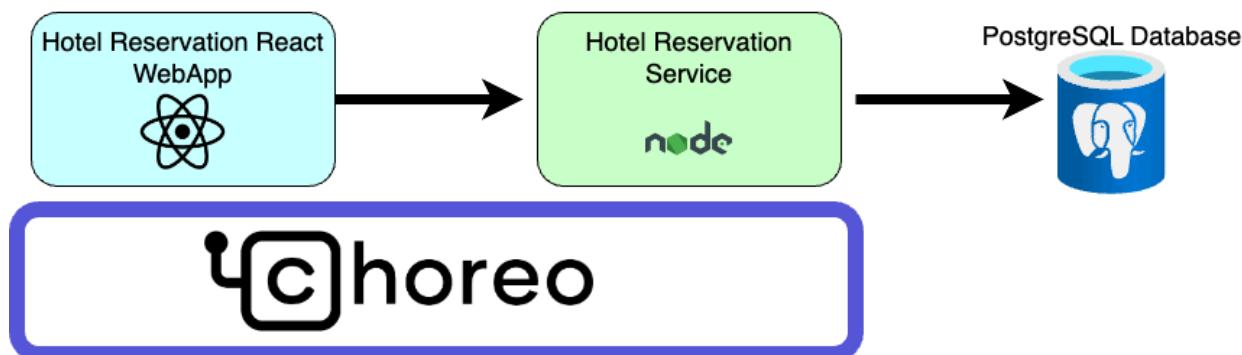
1. Deploy a backend service with a database
2. Deploy a frontend web app
3. Securing the frontend with Choreo Managed Authentication
4. Connecting the backend and frontend
5. Monitoring and troubleshooting

Prerequisites:

1. A GitHub Account
2. Git installed in your workstation
3. A recent version of Google Chrome, Mozilla Firefox
4. Microsoft Visual Studio (VSCode)
5. [Choreo](#) Account
6. [NodeJS](#) installed (above v20.11.0)
7. [DBeaver](#) or any other postgresql client

Business Scenario:

A luxury hotel has a reservation system that needs to be deployed to the cloud. The hotel reservation system will include a Hotel Reservation Service developed using NodeJS, and a Hotel Reservation Web Application developed using ReactJS. The data will be persisted in a PostgreSQL database.



The web application offers the following features:

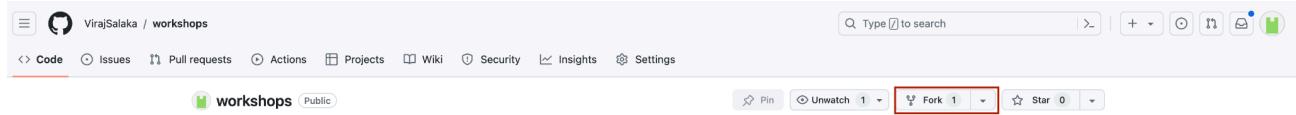
- **Room Search:** Users can search for rooms by specifying check-in and check-out dates, with an option to filter by number of guests. Search results will showcase a list of room types (eg: single, double etc). Each room listing will feature a "Reserve" button for easy booking.
- **Room Reservation:** To reserve a room, users need to input personal information: full name, contact number, and email. Upon reservation, a unique reference number is displayed for the user to copy.
- **List Reservations:** Users can look up their reservations once they are logged in. Each entry in the list will provide options to either update the reservation details or cancel the reservation.
- **Update Reservations:** Users have the flexibility to modify any part of their reservation.
- **Cancel Reservations:** Users can cancel their reservations at any time, with a straightforward option to cancel their booking.

Detailed Steps:

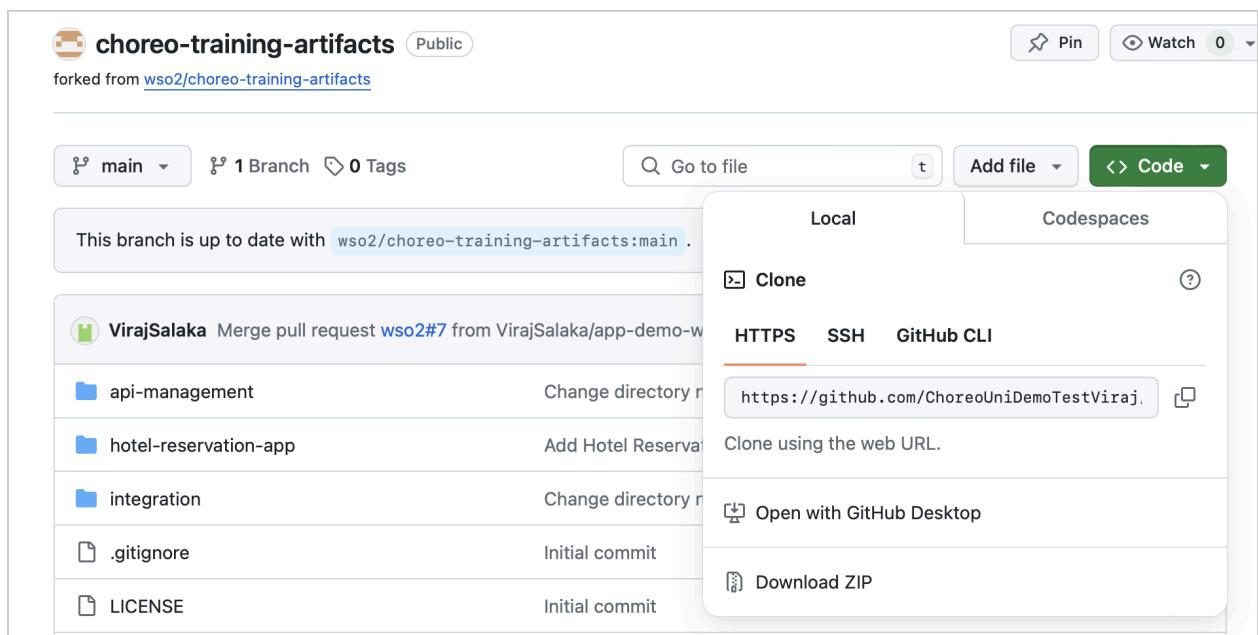
Follow the steps below to deploy the Hotel Reservation System.

Step 1: Fork and Clone the repository

1. Go to <https://github.com/wso2/choreo-training-artifacts> and click **Fork** repository.



2. To clone the repository, click the **Code** dropdown and copy the HTTPS URL that appears in the text box.



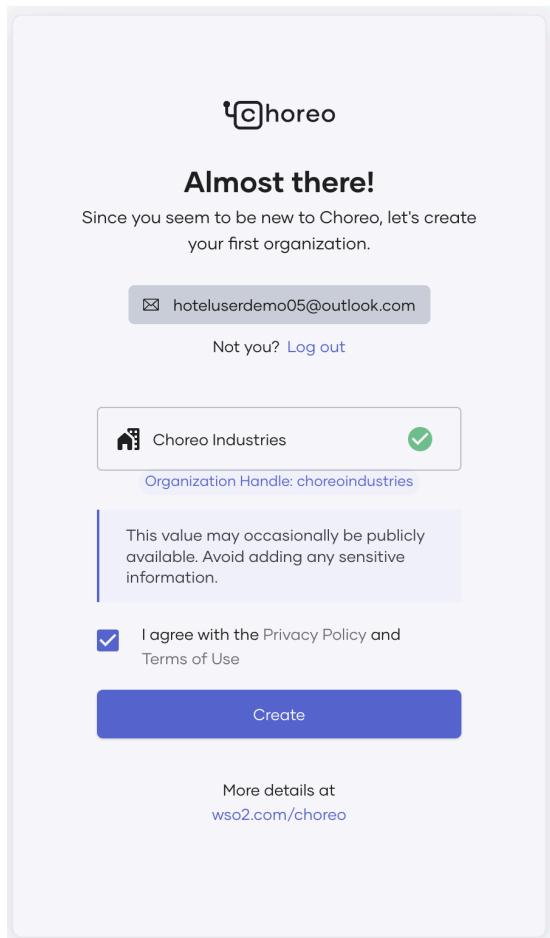
3. Run the following command in your terminal by adding the URL you copied in the previous step.

For Windows users: Open git bash and run the command below.

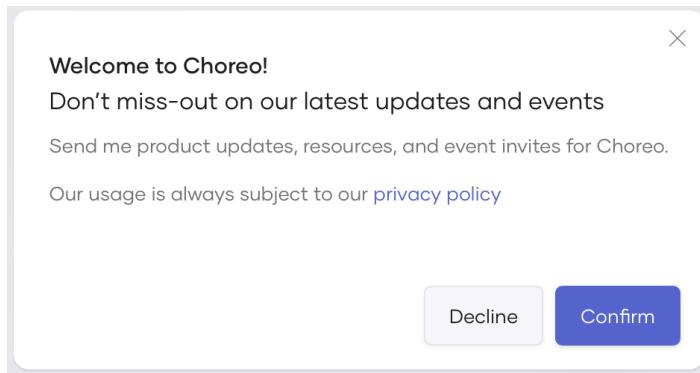
```
Unset
git clone <your_repository_url>
```

Step 2: Sign Up to Choreo

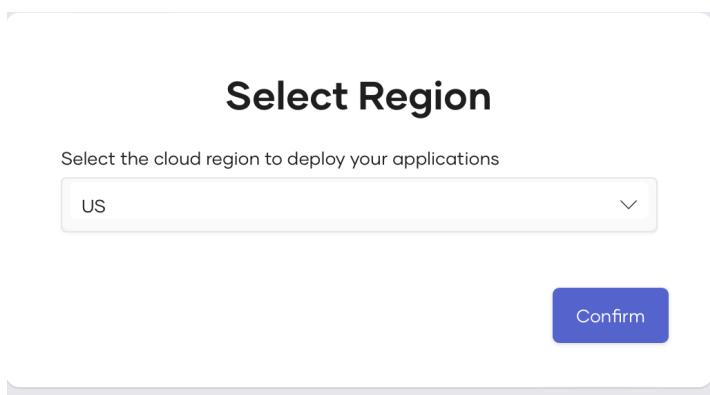
1. Sign Up to [Choreo](#).
2. Once you log in to Choreo for the first time, you will be asked to create an [organization](#). Provide a name for your organization and click **Create**. (*The organization name must be unique*)



3. If you are willing to subscribe for new updates regarding choreo, click **Confirm**.



4. In the region selector window, keep the default selected option as US. Then click **Confirm**.



Step 3: Create a database

- You are currently in the Organization view. In the left navigation menu, click **Dependencies > Databases**.

- In the Databases page, click the **Create** button.

- Add the details shown below and click **Next**. (*This is a reference for the database and not the database name*)

| Field Name | Field Value |
|------------|------------------------------|
| Name | hotel-reservation-service-db |

← Back to database server list

STEP 1
Select Database Type

STEP 2
Select service plan

Create Database Server

Select Storage

- PostgreSQL**
A highly performant, fully-managed object-relational database management system
- MySQL**
A fully-managed offering of the world's most popular relational database management system
- Choreo Cache**
A managed in-memory NoSQL database compatible with legacy Redis® OSS

Service Name
hotel-reservation-db

Back Next

4. The free tier only provides Digital Ocean as the available cloud provider. Leave the default values and click on the Hobbyist card under the **Select Service Plan**. Then, click **Create**.

5. The created database will be displayed (The database creation process may take a few minutes).

Step 4: Populate the database

1. Open the [Choreo Console](#) in a new tab.

- Click the **Organization** card in the top menu. In the left navigation menu, click **Dependencies -> Databases**. You can see your database and view database related information. Make sure that the database is in **active** state to perform the following operations. Click on the row to see database information.

The screenshot shows the WSO2 Choreo interface with the 'Databases' section selected. A single database entry is listed:

- Name:** hotel-reservation-service-db (PostgreSQL)
- Status:** Active
- Cloud/Region:** Digital Ocean Europe
- Service Plan:** Hobbyist
- Nodes:** 1
- CPU:** 1
- RAM:** 1GB
- Storage:** 8 GB
- Created:** 3 hours ago
- Actions:** (trash icon)

- We will use the database information to make a connection from our PostgreSQL database client.

The screenshot shows the detailed configuration for the 'hotel-reservation-service-db' database:

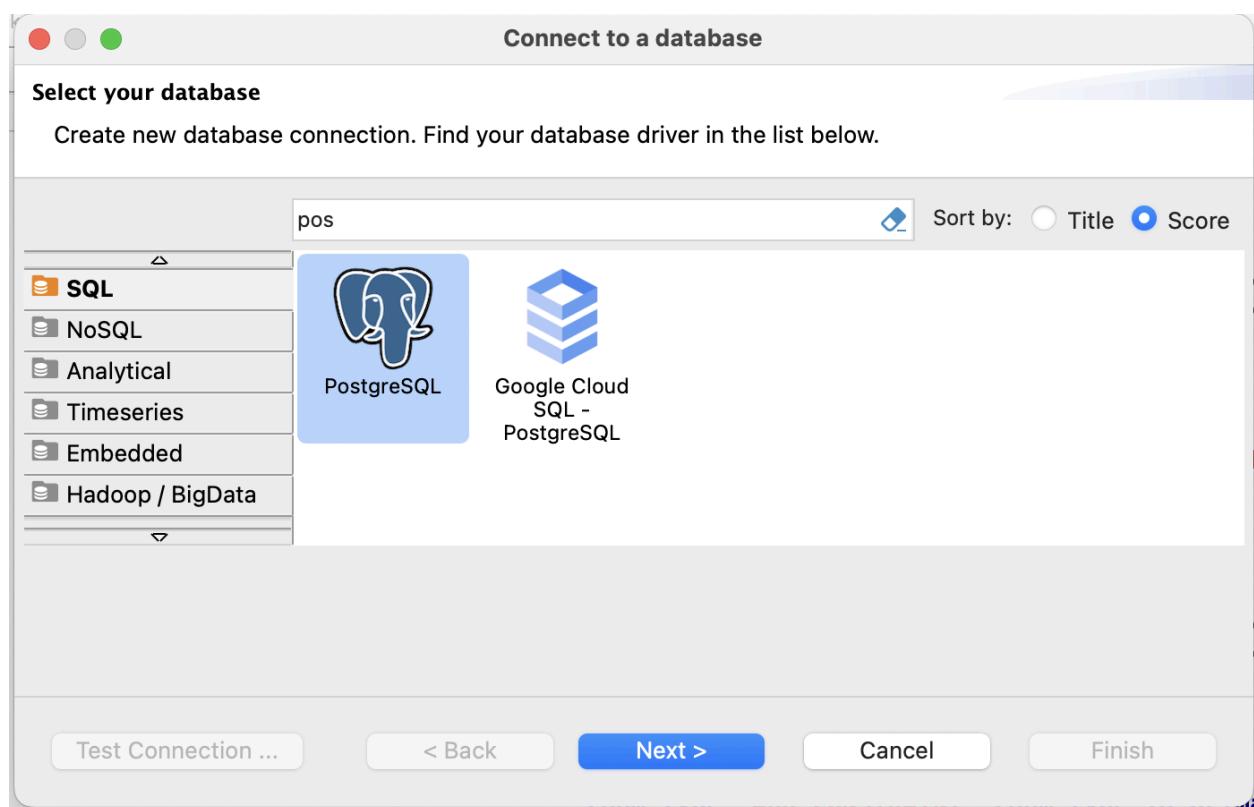
- Status:** Active
- Host:** pg-1d50754b-fd7f-4214-bd24-49815e18022a-hotelre1141684737-choreo.livencloud.com
- Port:** 14838
- Default User:** avnadmin
- Default Database:** defaultdb
- Password:** (View Password)
- CA Certificate:** (Download CA Certificate)

- Open DBeaver (or any other PostgreSQL client you use. The following steps are for DBeaver).
- From DBeaver, create a database connection.

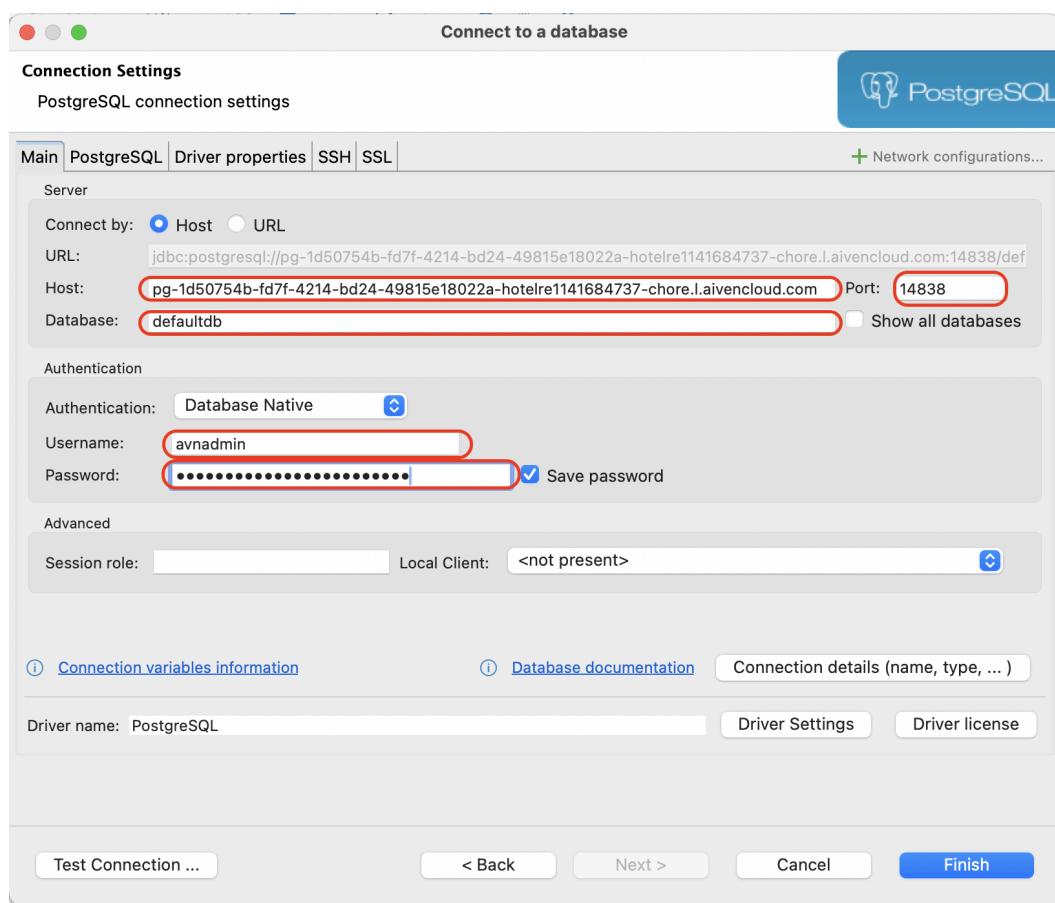
The screenshot shows the DBeaver application window with the following interface elements:

- Toolbar:** Includes File, Edit, Navigate, Search, SQL Editor, Database, and a Wrench icon.
- Tool Buttons:** Includes icons for New Connection, Open Connection, Refresh, and others.
- Bottom Bar:** Includes Database Navigator, Projects, and a search bar.
- Red Circle:** A red circle highlights the 'New Connection' button (a plus sign inside a blue circle) in the bottom bar.

- Select the driver as **PostgreSQL** and click **Next**.

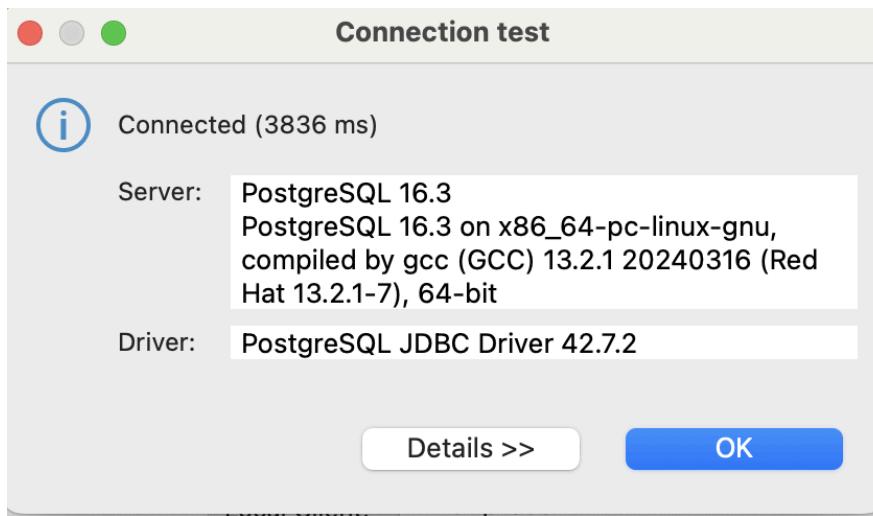


7. Update the Host, Port, Database, Username and Password with the database details obtained in step 3.

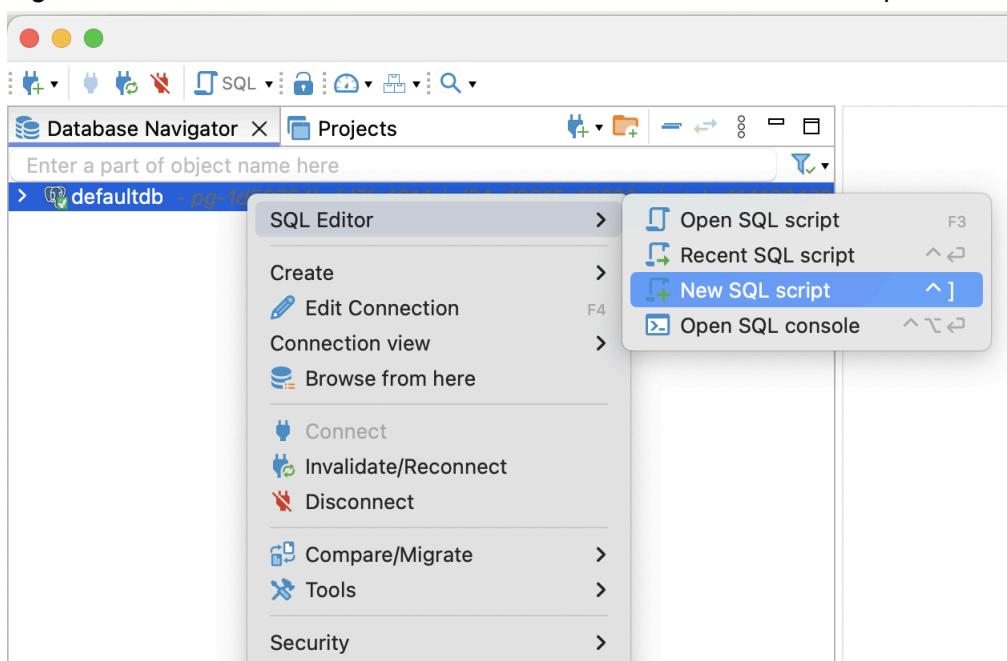


8. Click **Test Connection** to verify the configurations. If the connection is successful you will see the following message.

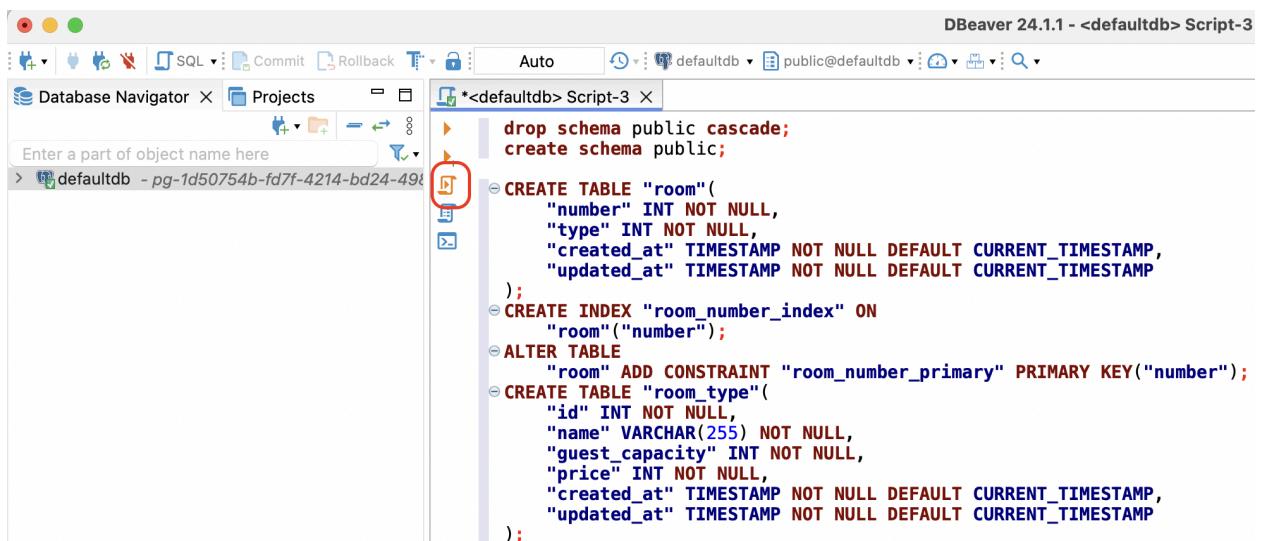
Note: If your connection test fails, verify your configurations and whether the database is in active state.



9. Finally click **Finish** to confirm the connection.
10. Right click on the database and select **SQL Editor -> New SQL script** and click.



11. Copy the content you find in this [schema.sql file](#) to the create script. And then click on the Execute Script icon. Wait till all the lines are executed.



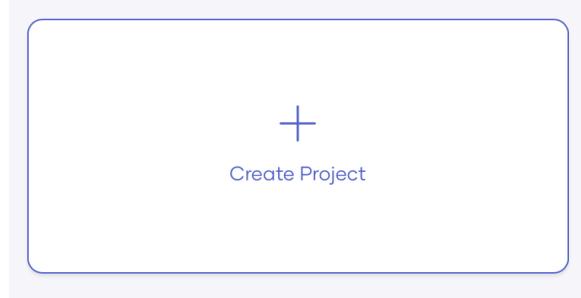
12. Open another script as in step 10. And Copy the content from the [seed.sql file](#) and execute the complete script. The database is now populated.

Step 5: Create a new Project

1. Click the Organization card in the top menu.



2. Click **Create Project** to create a new project.



3. Add the details shown below and click **Create**.

| Field Name | Field Value |
|------------|---------------|
| Name | Luxury Hotels |

The screenshot shows a modal dialog titled 'Create New Project'. It contains the following fields:

- Project Display Name:** A text input field containing 'Luxury Hotels'.
- Name:** A text input field containing 'luxury-hotels' with a help icon (?) and a copy icon (c).
- Project Description (Optional):** A text area containing 'Hotel Reservation system for Luxury Hotels'.

At the bottom of the dialog are two buttons: 'Back' (gray) and 'Create' (blue).

4. Once the project is successfully created, you will be directed to the overview of the project.

Step 6: Create and Deploy the Hotel Reservation Service

1. To deploy the backend NodeJS service, we will create a service component. On the component page, select **Service**.

The screenshot shows the WSO2 Choreo web interface. At the top, there are dropdown menus for 'Organization' (Choreo Industries) and 'Project' (Luxury Hotels). On the right, there are buttons for 'copilot', 'Upgrade', and a user profile. The main area has a sidebar with various icons. A central panel titled 'Explore Choreo in our Demo Organization' encourages users to 'Join Now'. Below this is a section titled 'Let's Start Building...' with a brief description and five steps: 'Step 1 Develop in VSCode', 'Step 2 Build and Deploy', 'Step 3 Manage', 'Step 4 Promote', and 'Step 5 Observe'. A large red box highlights the 'Service' component under 'Create a Component', which includes icons for REST, GraphQL, and GRPC. Other components shown are Web Application, API Proxy, Webhook, Scheduled Task, Manual Task, Event Handler, and Test Runner.

- Provide a name for the service component.

| Field Name | Field Value |
|------------------------|---------------------------|
| Component Display Name | Hotel Reservation Service |

- Click on the **Authorize With GitHub** button and click **Authorize Choreo.dev**.

Note: If you prefer not to authorize choreo.dev for your GitHub account at the moment, you can provide the GitHub repository URL in the input field provided and proceed.

- Expand the Select your organization section and click **+Add**.

The screenshot shows a modal dialog box for selecting an organization. It has tabs for 'GitHub', 'Bitbucket', and 'Container Registry', with 'GitHub' selected. Below the tabs is a search bar labeled 'Organization' with the placeholder 'Select your organization'. At the bottom is a button labeled '+ Add'.

- Choose the **Only select repositories** radio button (You can select the **all repositories** option is preferred). Then select the **choreo-training-artifacts** repository from the repository list and finally click **Install & Authorize**.



Install & Authorize Choreo.dev

Install & Authorize on your personal account
choreodemoviraj002

for these repositories:

All repositories
This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

Only select repositories
Select at least one repository.
Also includes public repositories (read-only).

Select repositories ▾

Selected 1 repository.

choreodemoviraj002/choreo-training-artifacts ×

with these permissions:

- Read access to issues and metadata
- Read and write access to code, pull requests, and repository hooks

Install & Authorize **Cancel**

Next: you'll be redirected to <https://console.choreo.dev/gapp>

- Click the refresh icon next to Organization.

GitHub Bitbucket Container Registry

Organization  Repository Public

Select your organization  Search for a repository 

- Select the relevant organization and select the **choreo-training-artifacts** repository. The branch will be automatically selected as main.

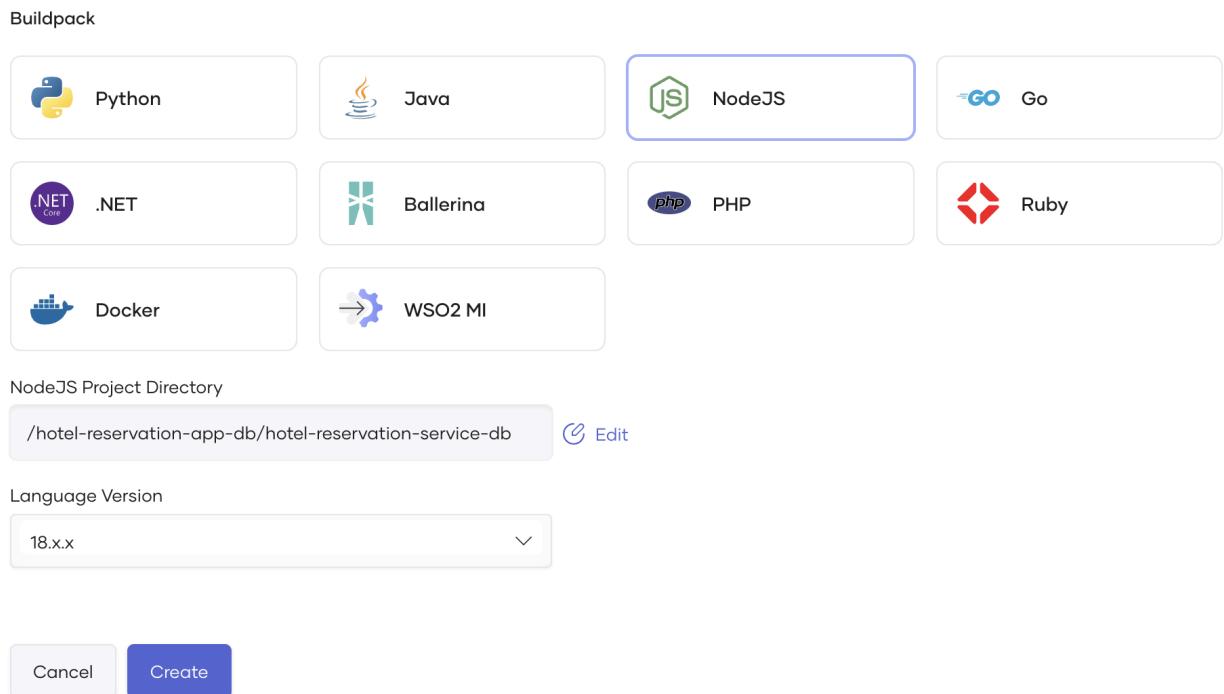
Connect Your Repository

Organization Repository Public Branch

choreodemoviraj002 choreo-training-artifacts main

- Select NodeJS as the Buildpack and set the Project Directory as below. Then click **Create**.

| Field Name | Field Value |
|--------------------------|---|
| NodeJS Project Directory | hotel-reservation-app-db/hotel-reservation-service-db |
| Language Version | 18.x.x |



9. Your component will now appear in the component listing. Click on the component to proceed.

Choreo Organization: Choreo Industries Project: Luxury Hotels

Explore Choreo in our Demo Organization
Join the Demo Organization and explore Choreo's features and capabilities with a predefined project and components.
[Join Now](#)

Non-Production Environments Development Production

| | | | |
|------------------|-------------------|------------------|-------------------|
| 8 Total Releases | 1 Weekly Releases | 1 Total Releases | 0 Weekly Releases |
|------------------|-------------------|------------------|-------------------|

Data for the past 3 months

Production Environments Development Non-Production Environments

| | | | |
|------------------|-------------------|------------------|-------------------|
| 1 Total Releases | 0 Weekly Releases | 8 Total Releases | 1 Weekly Releases |
|------------------|-------------------|------------------|-------------------|

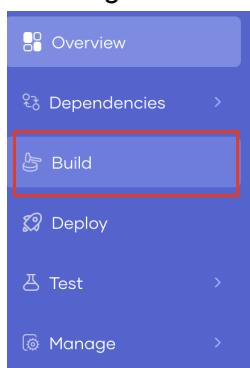
Data for the past 3 months

Component Listing [Create](#)

| Name | Description | Type | Last Updated |
|---------------------------|---|---------|---------------|
| Hotel Reservation Service | Hotel Reservation Backend Service for ... | Service | 2 minutes ago |

Create from a sample

10. Once you click the component from the listing as mentioned above, you are on the component's page. Next step is to build the code. Navigate to the **Build** page from the left navigation menu.





11. Click **Build Latest** to build the latest commit of the code. This process will take a few minutes.

The screenshot shows the 'Builds' section of the WSO2 UI. At the top right, there is a toggle switch for 'Auto Build on Commit' and a blue button labeled 'Build Latest'. Below the header, there is a table with columns: BUILD ID, COMMIT, STATUS, TIME, and ACTION. A message 'No Builds Available' is displayed in the center of the table area.

Once the build is completed, the status will show as **Success**.

The screenshot shows the same 'Builds' section after a build has been completed. The table now lists one build entry: BUILD ID 9806922982, COMMIT 127b8a18a (Manual), STATUS Success, and TIME 14 seconds ago. The 'View Details' link is visible on the right.

12. Until the build is complete, we can move to populating the database with necessary tables and data in **Step 3**.

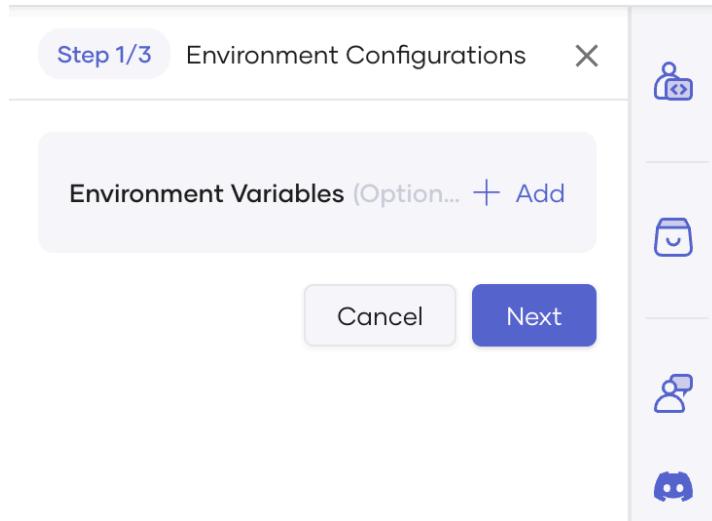
13. After a successful build the next step is to deploy the service. Click **Deploy** on the left navigation menu.

The screenshot shows the left navigation menu of the WSO2 UI. The options are: Overview, Dependencies, Build, Deploy, and Test. The 'Deploy' option is highlighted with a red rectangle.

14. In the Deploy page, click **Configure & Deploy**.

The screenshot shows the 'Deployment Track' page in the WSO2 UI. The top navigation bar shows 'Organization: Choreo Industries', 'Project: Luxury Hotels', and 'Component: Hotel Reservation Ser...'. The main area is titled 'Deployment Track' and shows three stages: Set up, Development, and Production. The 'Set up' stage is active, displaying a 'Build ID' of 9879309055 (Latest) and a 'Commit Details' section. The 'Configure & Deploy' button at the bottom of the 'Set up' section is highlighted with a red rectangle.

15. In the right side panel, you can add environment variables by clicking **+Add**.

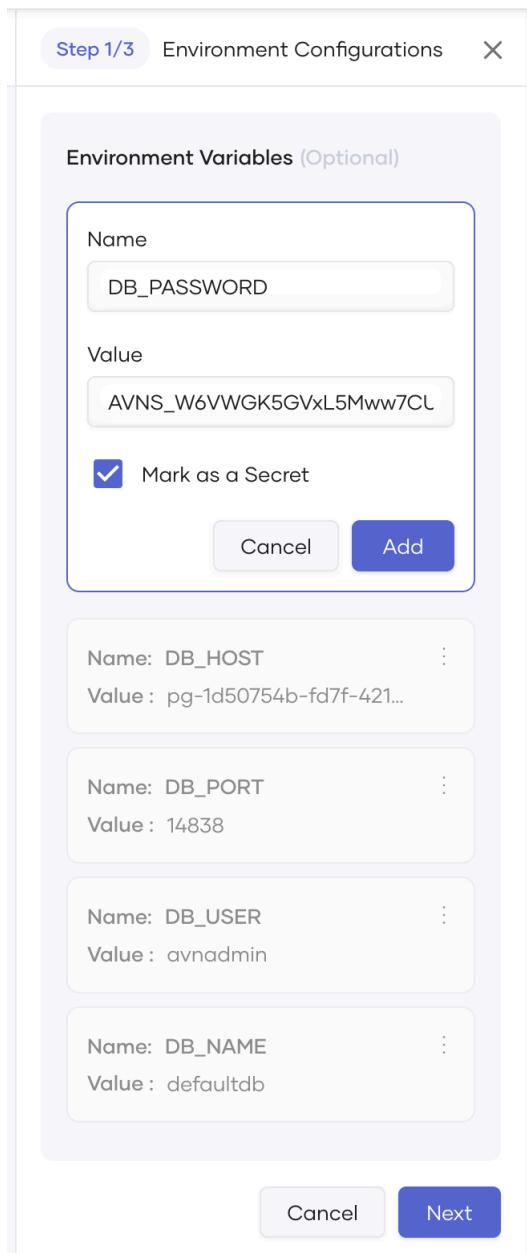


16. Add the following key value pairs as environment variables from the database connection details from Step 3. (You can refer to step 4.3 on how to obtain these values.

| Name | Value |
|---------|-------------------------------|
| DB_HOST | <i>Your database host</i> |
| DB_PORT | <i>Your database port</i> |
| DB_USER | <i>Your database username</i> |
| DB_NAME | <i>Your Database name</i> |

17. Let's add the database password as a secret. Get the password from database connection details as in above. Click **+Add** and provide the value, **tick the check box** to make it a secret. After providing the key value pair click **Add**.

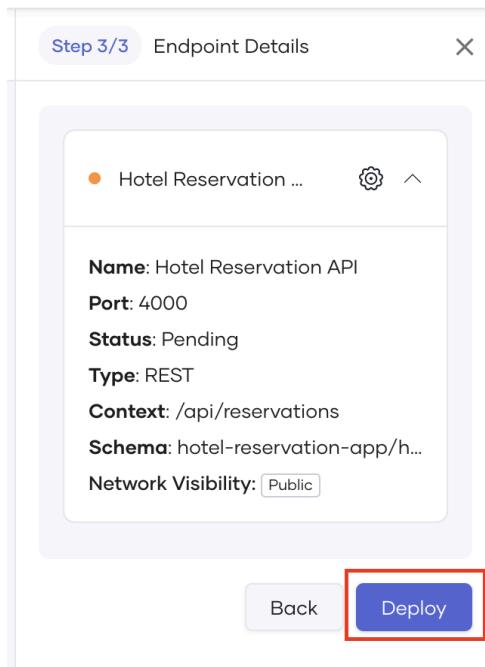
| Name | Value |
|-------------|-------------------------------|
| DB_PASSWORD | <i>Your database password</i> |



18. After all the environment variables are added, click **Next**.

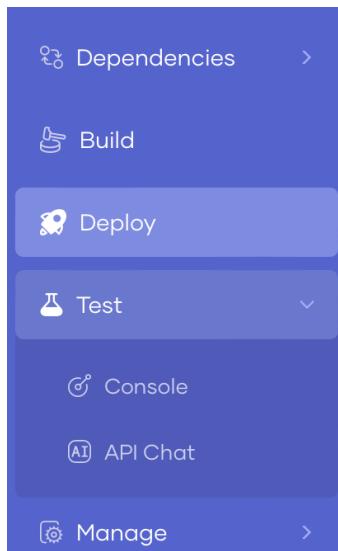
19. As for file mounts, click the **Next** button in the right side panel, as we don't have any configurations to provide.

20. Click **Deploy** in the right side panel to deploy the service.

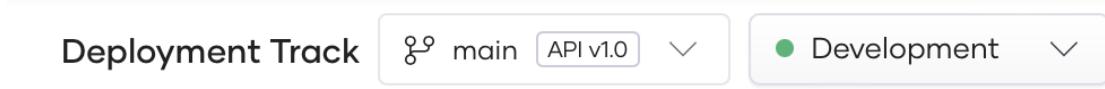




21. Navigate to **Test** in the left navigation menu and expand it. Click **Console** to access the Open API Console to test the service.



22. To test the service deployed in the development environment, select **Development** from the drop down.



23. Expand the **POST/** resource and click **Try it out**.

The screenshot shows the WSO2 API Management console interface. At the top, there's a header with 'Deployment Track' set to 'main API v1.0' and 'Development'. Below the header, there's a form for an endpoint named 'Hotel Reservation API' with the URL 'https://a71d4da0-1ffa-459b-ad60-b236bd45afee-dev1-us-east-azure.choreoapis.dev/luxury-hc'. Underneath the URL, there's a 'Security Header' field and a 'Get Test Key' button. The main area is titled 'POST /'. It has sections for 'Parameters' (which is empty) and 'Request body'. A 'Try it out' button is located in the top right corner of the request body section, and it is highlighted with a red box. There are also buttons for 'Example Value' and 'Schema' at the bottom of the request body section.

24. Provide the following details to the Request body section:

| Field Name | Field Value |
|--------------|---|
| Request body | { "checkinDate": "2024-02-19T14:00:00Z", "checkoutDate": "2024-02-20T10:00:00Z", "rate": 120, "user": { "id": "123", "name": "testuser", "email": "testuser@someemail.com", "mobileNumber": "911234567821" }, "roomType": "Family" } |

25. Click **Execute**.

26. Upon a successful POST request, you will receive a response as shown below.

```

{
    "id": "e386e7be-075c-4922-93ea-fbf603ab21ee",
    "user": {
        "id": "123",
        "name": "testuser",
        "email": "testusersome@email.com",
        "mobileNumber": "911234567821"
    },
    "room": 403,
    "checkinDate": "2024-02-19T14:00:00Z",
    "checkoutDate": "2024-02-20T10:00:00Z"
}
  
```

Download

27. Expand the **GET /roomTypes** resource and click **Try it out**.

28. Add the following details for each parameter and click **Execute**.

| Field Name | Field Value |
|---------------|----------------------|
| checkinDate | 2024-02-19T14:00:00Z |
| checkoutDate | 2024-02-20T14:00:00Z |
| guestCapacity | 2 |

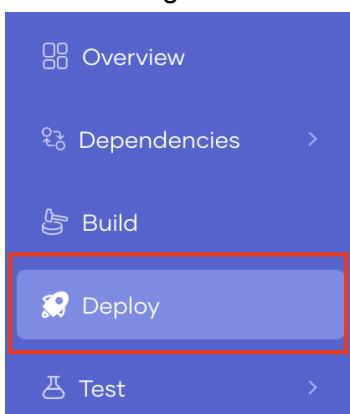
29. Upon a successful GET request, you will receive a response as shown below.

```

[
    {
        "id": 1,
        "name": "Double",
        "guestCapacity": 2,
        "price": 120
    },
    {
        "id": 3,
        "name": "Suite",
        "guestCapacity": 4,
        "price": 300
    }
]
  
```

Download

30. The next step is to deploy the service to the production environment. Click **Deploy** on the left navigation menu.



31. Click **Promote** to promote the build to the Production environment.

The screenshot shows the WSO2 Deployment Track interface. On the left is a vertical sidebar with various icons. The main area is divided into three panels: 'Set up' (containing build ID, commit details, and deployment status), 'Development' (showing a deployed build, endpoints, and scaling options), and 'Production' (showing a 'Not yet Deployed' status). A blue 'Promote' button in the 'Development' panel is highlighted with a red box.

32. In the right side panel, choose **Use Development Configurations** and click **Next**. In all the next configurations in the right side panel click **Next** and finally **Promote**.

The screenshot shows a modal dialog titled 'Step 1/4 Configuration Types'. It contains two radio button options: 'Define new configuration values' and 'Use Development configurations'. The second option is selected. At the bottom are 'Cancel' and 'Next' buttons. The 'Next' button is highlighted with a red box.

Step 7: Create and Deploy the Hotel Reservation Web Application

- From the top main menu, select the **Project** tab.

The screenshot shows the top navigation bar of the application. It includes the 'choreo' logo, 'Organization' dropdown (Choreo Industries), 'Project' dropdown (Luxury Hotels, highlighted with a red box), and 'Component' dropdown (Hotel Reservation Ser...).

- Click **Create** to create a new component.

The screenshot shows the 'Component Listing' page. It displays a table with columns: Name, Description, Type, and Last Updated. One row is shown: 'Hotel Reservation Service' (Description: 'Hotel Reservation Backend Service for Luxury Hotels', Type: 'Service', Last Updated: '11 minutes ago'). At the top right of the table is a search bar and a blue '+ Create' button, which is highlighted with a red box.

- To deploy the React web application, we will create a Web Application component. On the component page, select **Web Application**.

[← Back to Project Home](#)

Create a New Component

Select a Type [Try a Sample](#)

| | | | |
|---------------------|----------------------|--|--|
| Service | Web Application | API Proxy | Webhook |
| [REST] [RPC] [GRPC] | HTML [Node.js] [PHP] | [REST] | [GitHub] [Android] [iOS] [Google Sheets] |
| Scheduled Task | Manual Task | Event Handler | Test Runner |
| [Database] [File] | [Database] | [Apache Beam] [Apache Flink] [Apache Nifi] | [Jest] [Mocha] [Karma] [Gherkin] |

4. Provide a name for the Web Application.

| Field Name | Field Value |
|------------|----------------------------|
| Name | Hotel Reservation Frontend |

5. Click on **Authorize With GitHub**.

6. Select the **choreo-training-artifacts** repository and the **main** branch.

GitHub Bitbucket Container Registry

Connect Your Repository

Organization: ChereoUniDemoTestViraj Repository: choreo-training-artifacts Branch: main

7. Select the **React** Buildpack and set the following details. Then click **Create**.

| Field Name | Field Value |
|-------------------|--|
| BuildPack | React |
| Project Directory | /hotel-reservation-app-db/hotel-reservation-frontend |
| Build Command | npm run build |
| Build Path | /build |
| Node Version | 20 |



Buildpack

| | | | |
|--------|--------|----------------|--------|
| NodeJS | React | Angular | .NET |
| Vue.js | Python | Spring Boot | PHP |
| Go | Ruby | Static Website | Docker |

Project Directory

/hotel-reservation-app-db/hotel-reservation-frontend [Edit](#)

Build Command

npm run build

Build Path

/build

Node Version

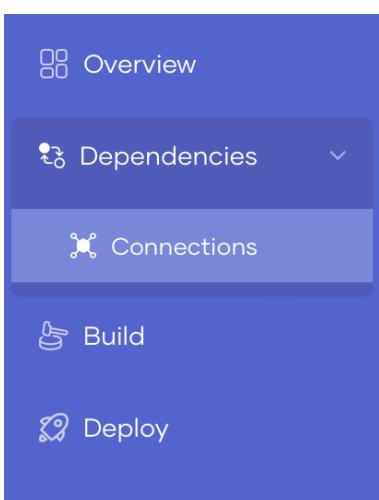
20

[Cancel](#) [Create](#)

8. Click **Build** on the left navigation menu. Click **Build Latest**.

| Builds C | | | | |
|--------------------------|--|-----------------------------|----------------|------------------------------|
| BUILD ID | COMMIT | STATUS | TIME | ACTION |
| 9807347044 | e379f9b1d Manual | In Progress | 17 seconds ago | View Details |

9. In order to connect the frontend and the backend we need to create a connection in Choreo. Click and expand **Dependencies** on the left navigation menu and click the **Connections** tab.



10. Click **Create**.

11. Select the **Hotel Reservation Service** that was created in step 4.

[← Back to Connection Listing](#)

Select a Service

The screenshot shows the 'Select a Service' search results. On the left, there are filters for Type (Internal, Third Party), Network Visibility (Organization, Public), and Categories. On the right, a service is listed with the following details:

- H** (Service icon)
- Hotel Reservation Service**
- Luxury Hotels
- REST
- Version: v1 Status: Published
- Hotel Reservation Backend Service for Luxury Hotels
- Public (checkbox checked)
- 2 days ago (last updated)

12. Provide a name for the connection and click **Create**.

| Field Name | Field Value |
|------------|------------------------------|
| Name | Hotel Reservation Connection |

13. Copy and save the Service URL as we need to provide it as a configuration when deploying the web application.

[← Back to Connection Listing](#)

Hotel Reservation Connection

[Developer Guide](#)

Connecting to Hotel Reservation Service [🔗](#)
 Connection Schema Default OAuth Connection - Public
 Service URL </choreo-apis/luxury-hotels/hotel-reservation-serv> [🔗](#)

14. Click **Deploy** on the left navigation menu and click **Configure & Deploy**.

In the right side panel, provide config.json as the Configuration File Name and provide the following content:

```
Unset
window.configs = {
    apiUrl: '<Service URL>',
};
```

Replace <Service URL> with the value that you copied after creating a connection to the Service in Step 13.

Ex:

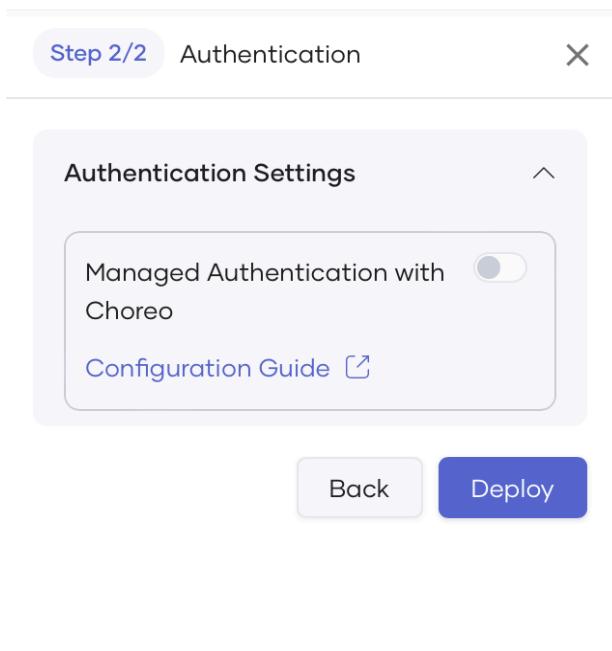
Unset

```
window.configs = {
    apiUrl: '/choreo-apis/luxury-hotels/hotel-reservation-service/v1',
};
```



15. Click Next.

16. Disable Managed Authentication with Choreo (Toggle the button). In our source code, we do not have the necessary implementation to make it work as of now. We will revisit this later.



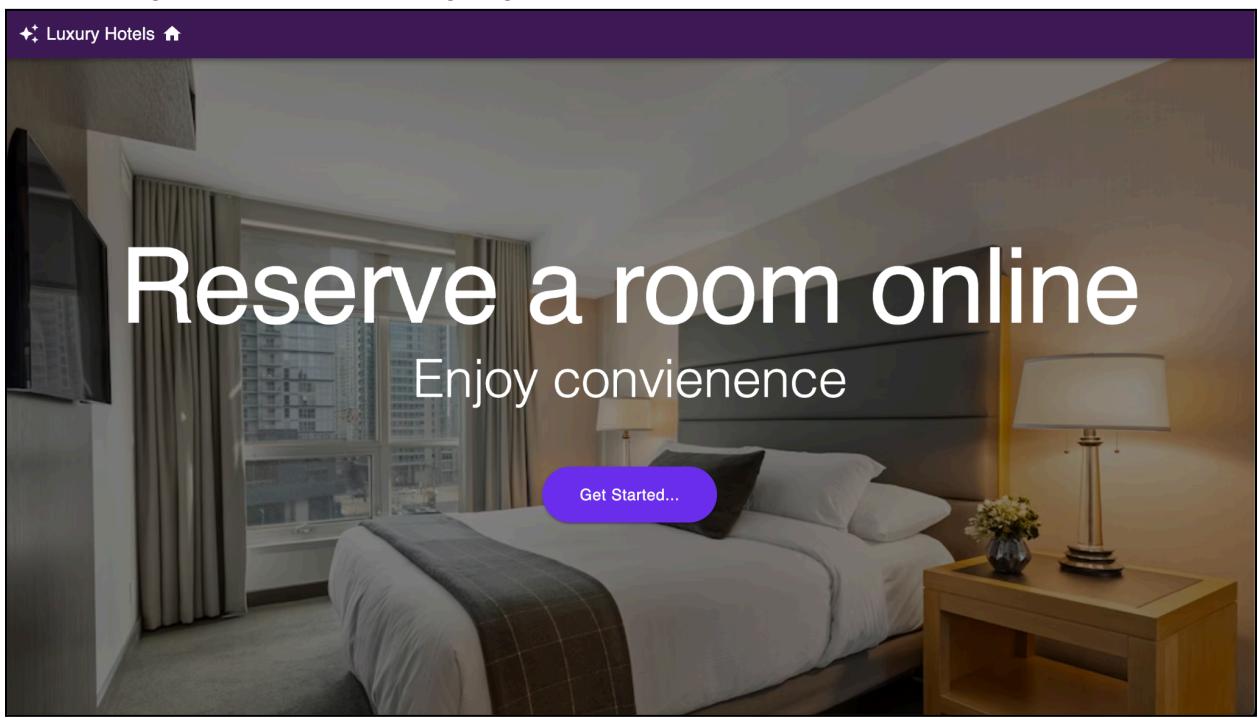
17. Click Deploy in the right side panel to deploy the web application.

18. Once the web application is deployed successfully, click on the Web App URL to access the web app.

The screenshot shows the WSO2 Choreo interface with the following details:

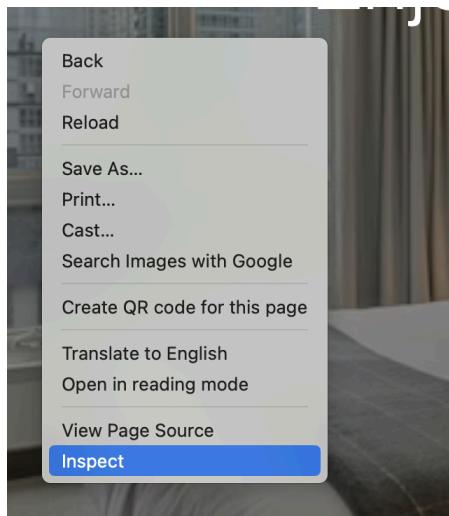
- Organization:** Choreo Industries
- Project:** Luxury Hotels
- Component:** Hotel Reservation Fro...
- Deployment Track:** main
- Development Environment:**
 - Build ID: 9877602034 (Latest)
 - Commit Details: Add Hotel Reservation Demo : wit...
 - by: VirajSalaka
 - Auto Deploy on Build: Enabled
 - Deployment Status: Active
 - Web App URL: <https://36b76148-0f38-486d-933...>
 - Scale to Zero: Enabled
 - Authentication: Managed Authentication: Disabled
 - Configurations: Manage Configs and Secrets
- Production Environment:**
 - Deployment Status: Not yet Deployed

19. You will be greeted with the landing page shown below:

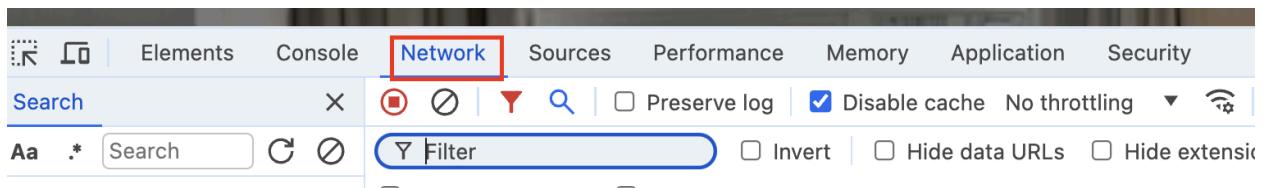


20. Click the **Get Started...** button.

21. Right click and click **inspect**.



22. Click on the **network** tab in the inspect view. Here we will monitor the network behavior.



23. Click **Get Started** first.

24. Set a check in date, check out date and select the no of guests. Then click **Search**.

| Name | Status | Type | Initiator | Size | Time |
|---|--------|------|-------------|-------|---------|
| roomTypes?checkinDate=2024-07-03T11:01:45.316Z&checkoutDate=2024-07-04... | 401 | xhr | retry.ts:7 | 432 B | 1.07 ms |
| refresh | 405 | xhr | retry.ts:13 | 701 B | 316 ms |

The reservation failed because our backend is secured. Our web application requires a valid token for successful invocation.

Step 8 - Add Managed Authentication for the Hotel Reservation Web App

1. Open **Visual Studio Code** and open the cloned repository.
2. Open the Terminal and run the following commands to navigate inside the web application source.

```
Unset  
cd hotel-reservation-app-db
```

```
Unset  
cd hotel-reservation-frontend
```

3. Run the following command to install node modules.

```
Unset  
npm install
```

4. We need to secure our web application. Hence we need to perform login functionality right after the user clicks on **Get Started**. Therefore, we need to do the following change in **"hotel-reservation-app-db/hotel-reservation-frontend/src/pages/landing_page/index.tsx"** file so that it directs to the login page rather than the *rooms* page.

Replace the following code segment:

```
Unset  
<Button  
    onClick={() => {  
        window.location.href = "/rooms";  
    }}  
    variant="contained"  
    color="secondary"  
    style={{  
        borderRadius: 32,  
        textTransform: "none",  
        height: 64,  
        width: 200,  
        fontSize: 18,  
    }}  
>  
    Get Started...  
</Button>
```

With the following:

```
Unset  
<Button  
    onClick={() => {  
        window.location.href = "/auth/login";  
    }}  
    variant="contained"  
    color="secondary"  
    style={{  
        borderRadius: 32,  
        textTransform: "none",  
        height: 64,  
        width: 200,  
        fontSize: 18,  
    }}  
>  
    Get Started...  
</Button>
```

5. Add the import statement and the code segment highlighted in green to the "**“hotel-reservation-app-db/hotel-reservation-frontend/src/App.tsx”** file in the exact order.

Unset

```
import Cookies from "js-cookie";
```

```
JavaScript
export default function App() {
  const [signedIn, setSignedIn] = useState(false);
  const [user, setUser] = useState<User>({
    email: "",
    id: "",
    name: "",
    mobileNumber: ""
  });
  const [isAuthLoading, setIsAuthLoading] = useState(false);

  function getMappedUser(userInfo: any): User {
    return {
      email: userInfo?.email || "",
      id: userInfo?.sub || "",
      name: userInfo?.first_name + " " + userInfo?.last_name,
      mobileNumber: userInfo?.mobile_number || ""
    };
  }

  useEffect(() => {
    setIsAuthLoading(true);
    if (Cookies.get("userinfo")) {
      // We are here after a login
      const userInfoCookie = Cookies.get("userinfo");
      sessionStorage.setItem("userInfo", userInfoCookie || "");
      Cookies.remove("userinfo");
      var userInfo = userInfoCookie ? JSON.parse(atob(userInfoCookie)) : {};
      setSignedIn(true);
      setUser(getMappedUser(userInfo));
    } else if (sessionStorage.getItem("userInfo")) {
      // We have already logged in
      var userInfo = JSON.parse(atob(sessionStorage.getItem("userInfo")!));
      setSignedIn(true);
      setUser(getMappedUser(userInfo));
    } else {
      console.log("User is not signed in");
      if (
        window.location.pathname !== "/auth/login" &&
        window.location.pathname !== "/"
      ) {
        window.location.pathname = "/auth/login";
      }
    }
    setIsAuthLoading(false);
  }, []);
}
```

```
if (isAuthLoading) {  
    return <div>User authenticating...</div>;  
}
```

6. Let's now implement the Logout functionality. Go to "**hotel-reservation-app-db/hotel-reservation-frontend/src/layout/AppBar.tsx**" and add the following import statement and the code segment highlighted in green to add the **Logout** functionality.

```
Unset  
import Cookies from "js-cookie";
```

```
Unset  
<MenuItem  
    onClick={() => {  
        sessionStorage.removeItem("userInfo");  
        window.location.href =  
`/auth/logout?session_hint=${Cookies.get('session_hint')}`;  
    }}  
    >  
    <Button style={{ transform: "none" }}>  
        <Typography textAlign="center">Logout</Typography>  
    </Button>  
</MenuItem>
```

7. We now have all the required code level changes. Next we need to push these changes to the github repository. Run the following set of commands.

```
Unset  
git add -A
```

```
Unset  
git commit -m "Implement authentication functionality for the web application"
```

```
Unset  
git push origin main
```

Optional:

In case you run into issues with pushing the changes to the remote repository, Follow the github docs to [use a personal access token](#). And when the terminal prompts you asking the password, provide the github personal access token.

8. Visit the github repository and verify that the changes are there.

9. Visit the [Choreo console](#). Click **Hotel Reservation Frontend** from the component listing.

| Name | Description | Type | Last Updated |
|------------------------------|---|-----------------|----------------|
| H Hotel Reservation Frontend | | Web Application | 59 minutes ago |
| H Hotel Reservation Service | Hotel Reservation Backend Service for Luxury Hotels | Service | 1 hour ago |

10. In the left navigation menu, click **Build Latest**.

11. While the build is queued, make sure that the latest **commit ID** you have in your github repository's main branch matches with the **commit** listed in the processing record .

In choreo:

| BUILD ID | COMMIT | STATUS | TIME | ACTION |
|------------|-----------------|--------|----------------|------------------------------|
| 9779190308 | 3b5d713d Manual | Queued | 21 seconds ago | View Details |

In github commits,

Commits

main

Commits on Jul 3, 2024

- Add managed auth back

VirajSalaka committed 18 hours ago

3b5d711
- fix

e8ba47a

12. Once the build is successful, click **Deploy** on the left navigation menu.

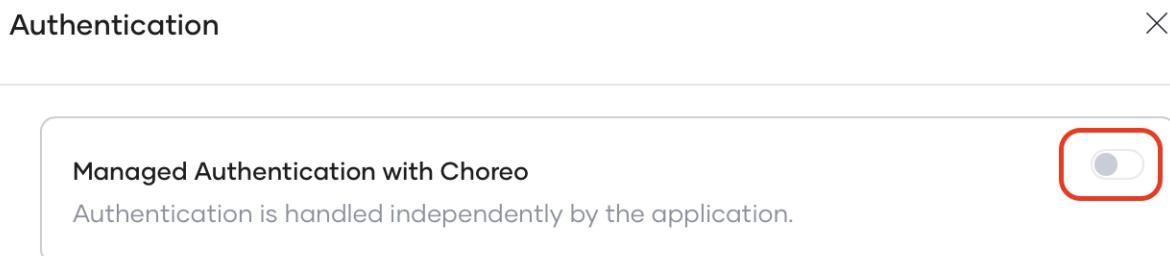
- Dependencies
- Connections
- Build
- Deploy
- Test
- Manage

13. In the **Set up** card in the Deploy page , click **Authentication Settings**.

Deployment Track main

| Set up | Development | Production | | | | | | | | |
|--|---|----------------------|----------|------------|----------------|-------------------------------------|----|-----------|----|-------------|
| Build ID 9808095616 <small>Latest</small> <small>1 minute ago</small> Commit Details Implement authentication functio... -O 51bcd977f by VirajSalaka | Deployed <small>1 hour ago</small> Deployment Status Active | Not yet Deployed | | | | | | | | |
| Auto Deploy on Build <input checked="" type="checkbox"/> Authentication Settings <input type="button" value="Deploy"/> | | | | | | | | | | |
| Image Deployment History <table border="1"> <tr> <td>Build ID</td> <td>9807347044</td> </tr> <tr> <td>Commit Details</td> <td>Add Hotel Reservation Demo : wit...</td> </tr> <tr> <td>-O</td> <td>e379f9b1d</td> </tr> <tr> <td>by</td> <td>VirajSalaka</td> </tr> </table> Web App URL https://b72526a8-3be3-4e09-8d... Scale to Zero Enabled | | | Build ID | 9807347044 | Commit Details | Add Hotel Reservation Demo : wit... | -O | e379f9b1d | by | VirajSalaka |
| Build ID | 9807347044 | | | | | | | | | |
| Commit Details | Add Hotel Reservation Demo : wit... | | | | | | | | | |
| -O | e379f9b1d | | | | | | | | | |
| by | VirajSalaka | | | | | | | | | |
| Authentication | | | | | | | | | | |

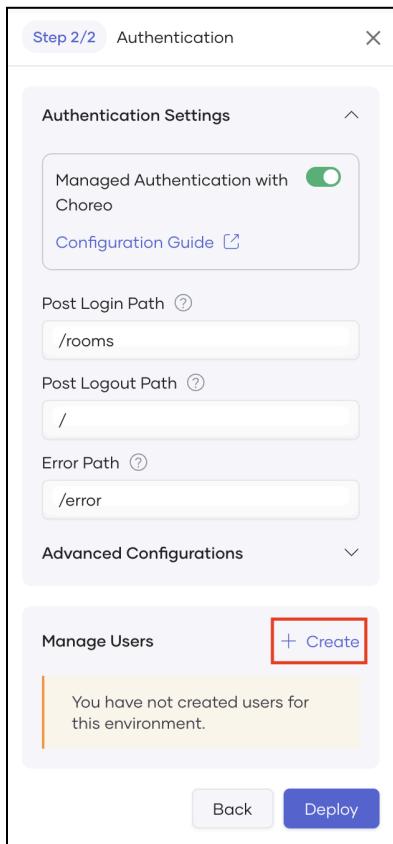
14. Enable Managed Authentication by clicking on the **Manage Authentication with Choreo** toggle button in the right side menu.



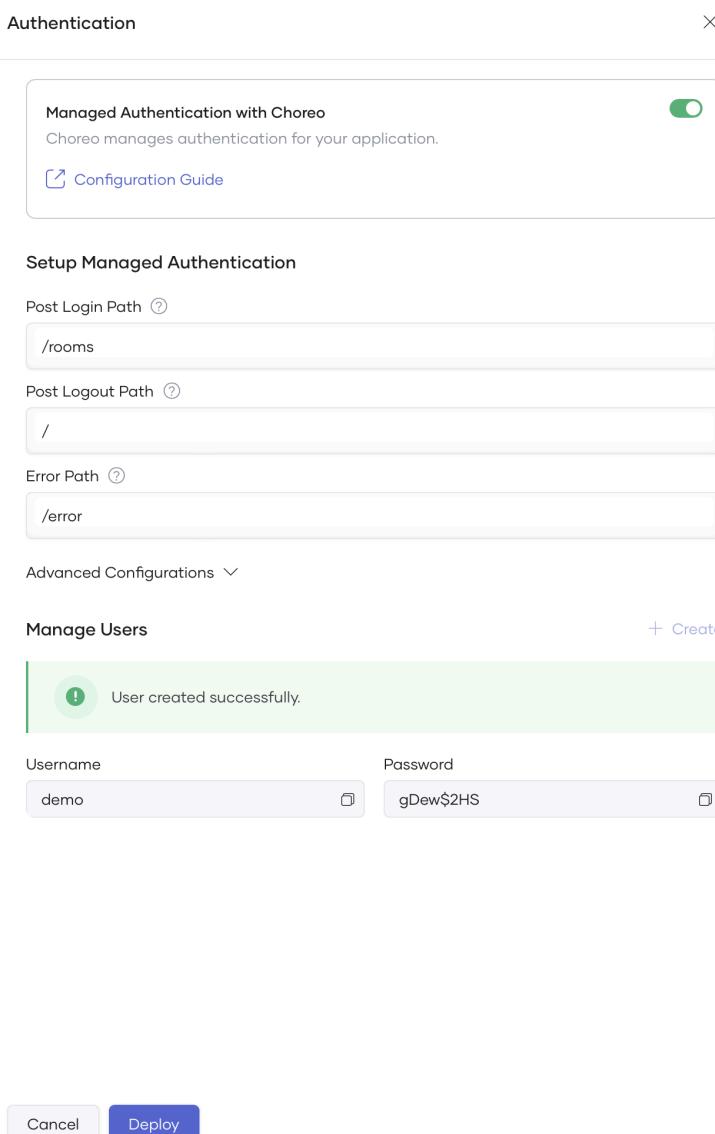
15. Provide following values for each parameter in the Authentication Settings window.

| Field Name | Field Value |
|------------------|-------------|
| Post Login Path | rooms |
| Post Logout Path | / |
| Error Path | /error |

16. Click **Create** to create a demo user.



17. Once the user is created, save the username and the password for later use.

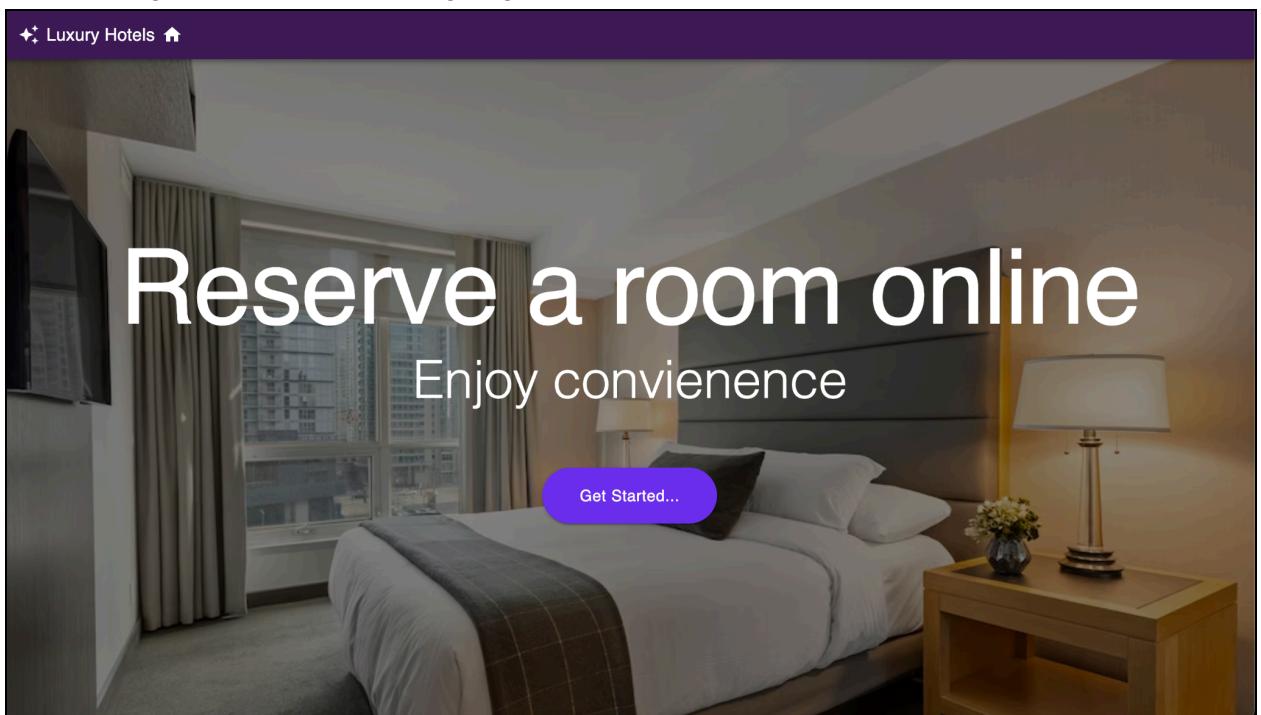


18. Click **Deploy** in the right side panel to deploy the web application.

19. Once the web application is deployed successfully, click on the Web App URL to access the web app.

The screenshot shows the WSO2 Choreo Deployment Track interface. At the top, it displays the organization 'Choreo Industries', project 'Luxury Hotels', and component 'Hotel Reservation F...'. The main area is divided into 'Development' and 'Production' sections. In the Development section, a deployment for Build ID 9877602034 (Latest) is shown as 'Deployed 19 seconds ago' with an 'Active' status. The 'Web App URL' field contains the value <https://36b76148-0f38-486d-933...>, which is also highlighted with a red box. The Production section shows a status of 'Not yet Deployed' with a bell icon.

20. You will be greeted with the landing page shown below:



21. Click the **Get Started...** button.

22. This will direct you to the login page. Provide the demo username and password (created in the Step 17) and click **Sign In**.

Note

If you forgot to copy the password, you can still retrieve it. Follow the instructions in [this document](#) to navigate to the location where you can view the configured user and copy the username and password.

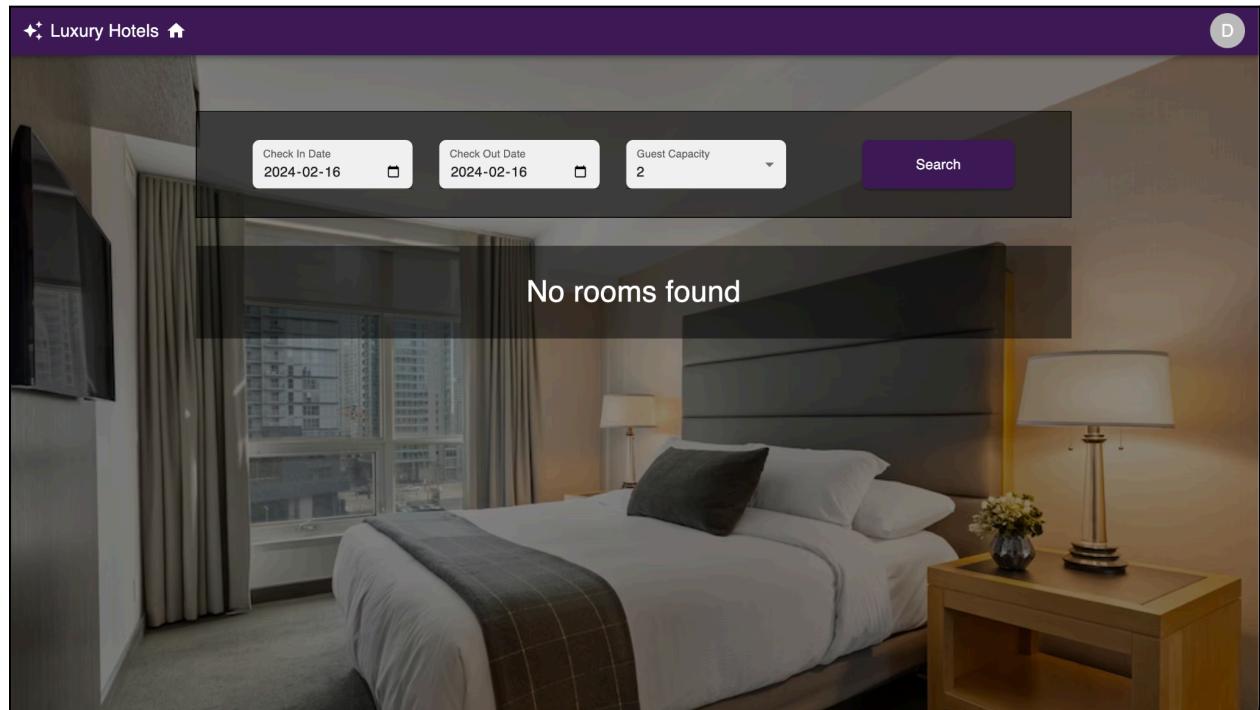
Sign In

Username

Password

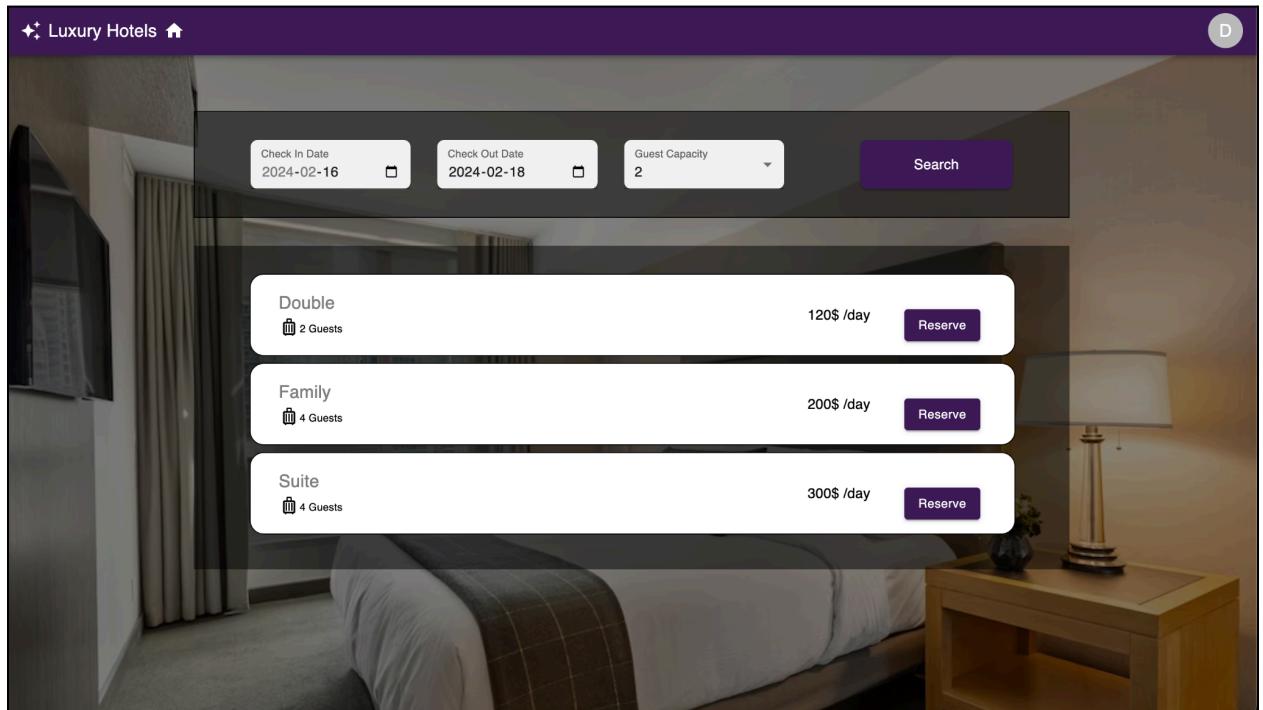
Sign In

23. You will be logged into the Hotel Reservation web application.



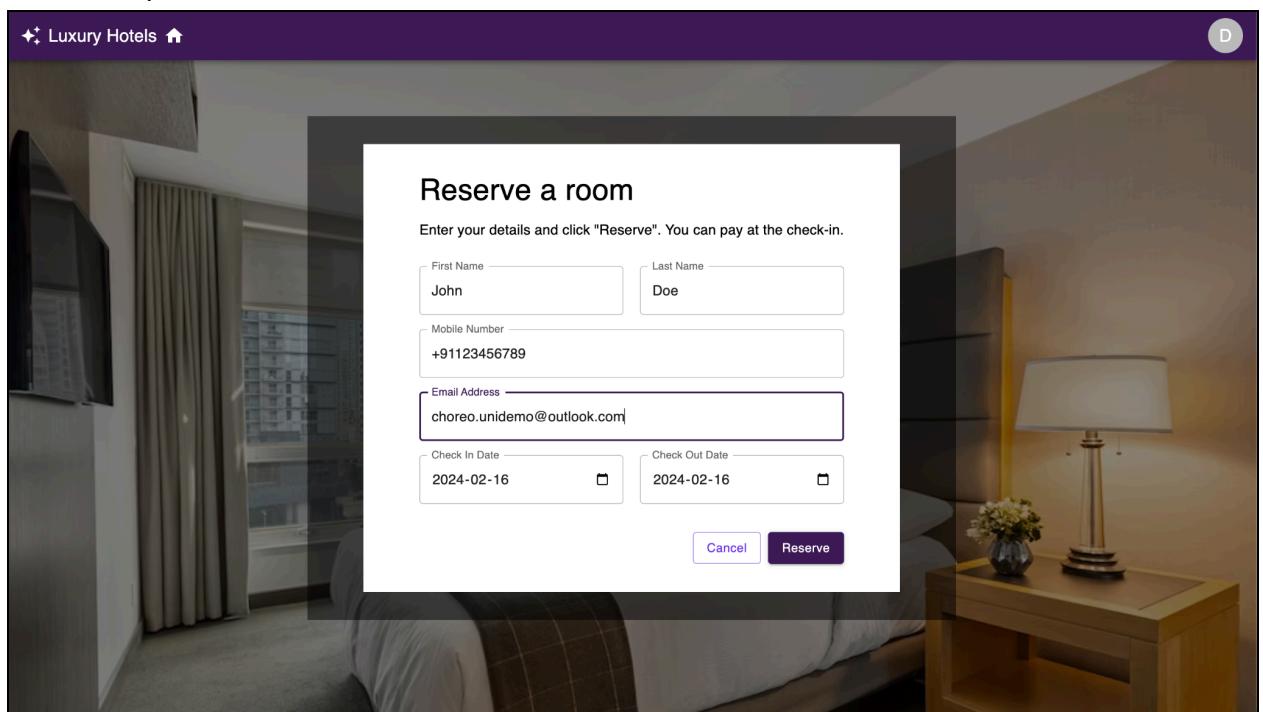
24. Set a check in date, check out date and select the no of guests. Then click **Search**.

25. You will be able to get the available rooms for the selected date range and guest capacity.



26. Click **Reserve**.

27. Provide required details as below and click **Reserve**.



28. Your reservation will be created.