


Deploy a Modern Web App connected to a Cloud Spanner Instance

Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

1. Click **Activate Cloud Shell**  at the top of the Google Cloud console.
2. Click through the following windows:
 - Continue through the Cloud Shell information window.
 - Authorize Cloud Shell to use your credentials to make Google Cloud API calls.

When you are connected, you are already authenticated, and the project is set to your **Project_ID**, `qwiklabs-gcp-02-b3ac78d1ff68`. The output contains a line that declares the **Project_ID** for this session:

```
Your Cloud Platform project in this session is set to quiklabs-gcp-02-b3ac78d1ff68
```

`gcloud` is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

3. (Optional) You can list the active account name with this command:
`gcloud auth list`

4. Click **Authorize**.

Output:

```
ACTIVE: *
ACCOUNT: student-03-906d853a6ad1@quiklabs.net

To set the active account, run:
$ gcloud config set account `ACCOUNT`
```

5. (Optional) You can list the project ID with this command:
`gcloud config list project`

Output:

```
[core]
project = qwiklabs-gcp-02-b3ac78d1ff68
```

Note: For full documentation of `gcloud`, in Google Cloud, refer to [the gcloud CLI overview guide](#).

Cloud Spanner instance

In order to allow you to move more quickly through this lab the Cloud Spanner instance, database, and tables required for the OmegaTrade application were automatically created for you.

Here are some details for your reference:

Item	Name	Details
Cloud Spanner Instance	omegatrade-instance	This is the project-level instance
Cloud Spanner Database	omegatrade-db	This is the instance specific database
Table	Users	Contains user accounts
Table	Companies	Contains company name and stock symbol
Table	CompanyStocks	Contains stock values
Table	Simulations	Tracks the state of each simulation

Task 1. Enable required Google Cloud APIs

First enable the Google Cloud APIs for Cloud Spanner, Container Registry, and Cloud Run.

1. In the **Cloud Shell** enter the following commands:

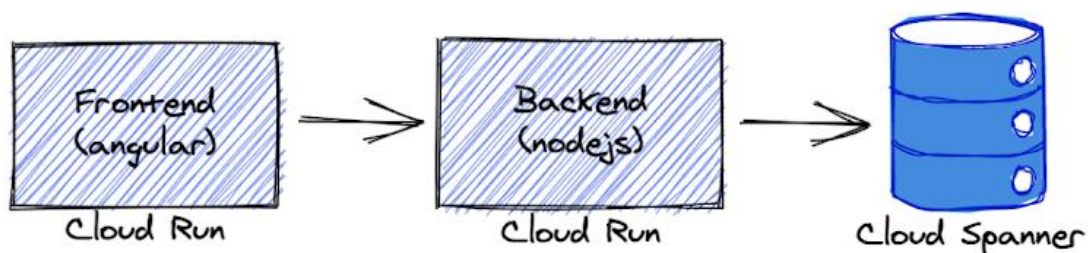
```
gcloud services enable spanner.googleapis.com
gcloud services enable artifactregistry.googleapis.com
gcloud services enable containerregistry.googleapis.com
gcloud services enable run.googleapis.com
```

Task 2. Download and inspect the application code

1. Download the code repository for use in this lab. In the **Cloud Shell** enter the following:

```
git clone https://github.com/GoogleCloudPlatform/training-data-analyst
```
2. Navigate to the folder containing the application code.

```
cd training-data-analyst/courses/cloud-spanner/omegatrade/
```
3. The code is divided primarily into two parts, backend and frontend. The application architecture is depicted in the diagram below:



4. The application relies on a deployment-specific file named **.env** to successfully communicate with the Cloud Spanner instance. You will create this file in the next task.

5. With respect to the backend, some of the bindings and interactions with the Cloud Spanner tables are managed by Node.js models - three of these **company.model.js**, **simulation.model.js**, and **user.model.js** - reside in the **models** folder. Issue the following command to navigate to the models folder:

```
cd backend/app/models
```

6. Issue the following command to inspect the **company.model.js** file. This file contains database operations to interact with the **companies** table.
- ```
more company.model.js
```

Press the spacebar to advance through the file until its end. If you wish to close the file early, type **q** to close the **more** command.

7. Some of the frontend interactions with these models and other structures in the Node.js backend occur through Angular components located in the **components** folder. Issue the following command to navigate to the components folder:
- ```
cd ../../../../frontend/src/app/components
```

For instance, the **company** component contains the base application code to manage and update company information.

8. Issue the following command to inspect the **manage-company.component.ts** TypeScript configuration file. This file contains methods for deleting or editing a company, among other actions.

```
more company/manage-company/manage-company.component.ts
```

Press the spacebar to advance through the file until its end. If you wish to close the file early, type **q** to close the **more** command.

Task 3. Build and deploy the backend component

1. Navigate to the folder containing the code required to build and deploy the backend.
- ```
cd ../../../../backend
```
2. Create the **.env** file. As mentioned earlier, this file contains project specific information so that the application's backend component can communicate with the Cloud Spanner instance.

In the cloud shell enter the following command to invoke the **Nano** text editor and create a new **.env** file.

```
nano .env
```

Paste the code block listed below.

```
PROJECTID = qwiklabs-gcp-02-b3ac78d1ff68
INSTANCE = omegatrade-instance
DATABASE = omegatrade-db
JWT_KEY = w54p3Y?4dj%8Xqa2jjVC84narhe5Pk
EXPIRE_IN = 30d
```

Press **Ctrl+X** to exit Nano, **Y** to confirm the update, and press **Enter** to save your changes.

3. Before you proceed further you must install updated components for **npm** so that the backend can be properly compiled. **npm** is a package manager for JavaScript. **npm** is the default package manager for the JavaScript runtime environment Node.js.

```
nvm install 22.6
```

```
npm install npm -g
npm install --loglevel=error
```

4. Next build the backend application using a reference dockerfile that exists in the repository folder.

```
docker build -t gcr.io/qwiklabs-gcp-02-b3ac78d1ff68/omega-
trade/backend:v1 -f dockerfile.prod .
```

**Note:** You may safely ignore any **npm notice...** messages that appear during the build process

5. Prior to pushing the new application package, run the following command to set configuration permissions in the Cloud Shell.

Enter **'y'** when prompted if you want to continue.

```
gcloud auth configure-docker
```

6. Push the newly created application package to the Container Repository for your Qwiklabs project.

```
docker push gcr.io/qwiklabs-gcp-02-b3ac78d1ff68/omega-trade/backend:v1
```

7. Finally deploy the backend application using Cloud Run. Cloud Run is a serverless deployment framework which abstracts away infrastructure management and scales up or down automatically almost instantaneously depending on traffic.

```
gcloud run deploy omegatrade-backend --platform managed --region us-
east1 --image gcr.io/qwiklabs-gcp-02-b3ac78d1ff68/omega-
trade/backend:v1 --memory 512Mi --allow-unauthenticated
```

8. Copy the URL provided at the end of the deployment. Preserve the URL in notepad, a text file, or other readily accessible location. This URL will be supplied to the frontend application to ensure the user interface can properly communicate with the Cloud Spanner database via the backend.

The backend Service URL will appear in the following format:

```
https://omegatrade-backend-zzzyyyxx1x-uw.a.run.app
```

## Task 4. Import sample stock trade data to the database

1. To import sample company and stock data, run the following command in the current (main backend) folder.

```
unset SPANNER_EMULATOR_HOST
node seed-data.js
```

2. You will receive confirmation that the tables were loaded successfully.

```
Inserting Companies...
done
Inserting Simulations...
done
Inserting Stocks...
done
Data Loaded successfully
```

## Task 5. Build and deploy the frontend component

1. Navigate to the directory containing the frontend code. Specifically navigate to the environments folder to update the configuration file to point to your backend component.

```
cd ../frontend/src/environments
```

2. In the cloud shell enter the following command to invoke the **Nano** text editor and open the **environment.ts** file.

```
nano environment.ts
```

3. Carefully delete the string **http://localhost:3000** and replace it with your backend URL.

**Note:** Be certain to retain the **/api/v1/** portion of the URL

Press **Ctrl+X** to exit Nano, **Y** to confirm the update, and press **Enter** to save your changes.

Your updated **environment.ts** file should appear like the example below.

```
export const environment = {
 production: false,
 name: "dev",
 // change baseUrl according to backend URL
 baseUrl: "https://omegatrade-backend-zzzzyyxx1x-uw.a.run.app/api/v1/",
 // change clientId to actual value you have received from OAuth
 console
 clientId: ""
};
```

4. Navigate to the main frontend folder.

```
cd ../../
```

5. Install updated components for **npm** so that the frontend can be properly compiled.

```
npm install npm -g
```

```
npm install --loglevel=error
```

**Note:** You may safely ignore any **npm WARN config...** messages that appear during the installation process

6. Next build the frontend application using a reference dockerfile that exists in the repository folder. The frontend build may take 5 to 10 minutes to complete.

```
docker build -t gcr.io/qwiklabs-gcp-02-b3ac78d1ff68/omegatrade/frontend:v1 -f dockerfile .
```

**Note:** You may safely ignore any **npm notice...** messages that appear during the build process

7. Push the newly created application package to the Container Repository for your Qwiklabs project.

```
docker push gcr.io/qwiklabs-gcp-02-b3ac78d1ff68/omegatrade/frontend:v1
```

8. Finally deploy the frontend application using Cloud Run.

```
gcloud run deploy omegatrade-frontend --platform managed --region us-east1 --image gcr.io/qwiklabs-gcp-02-b3ac78d1ff68/omegatrade/frontend:v1 --allow-unauthenticated
```

9. The frontend Service URL will appear in the following format. You may click the URL directly to open it or copy the URL and paste it into a new tab.

```
https://omegatrade-frontend-zzzzyyxx1x-uw.a.run.app
```

## Task 6. Perform operations in the OmegaTrade Application

1. On the application launch page click the **sign up** link.
2. Use the following details to create a new account for a fictitious company named **Spanner1**.

| Item                  | Value              |
|-----------------------|--------------------|
| Business email        | admin@spanner1.com |
| Full Name             | Spanner1 Admin     |
| Password              | Spanner1           |
| Confirm your Password | Spanner1           |

3. Your account will be created and you will be logged in.
4. To examine the OmegaTrade application, navigate to the **Dashboard** and choose **Foobar Inc** from the selector to see the stock performance chart for Foobar Inc. You will see a range of simulated stock prices over time for Foobar Inc.
5. Navigate to **Manage Company** and add **Spanner1** as a new company.
6. Click **Add Company** on the right side of the page. On the pop-up window, input the following values:



| Item         | Value    |
|--------------|----------|
| Company Name | Spanner1 |
| Short Code   | SPN      |

Then click **Save**.

- Spanner1** is now in the list of companies.
- Navigate to the **Dashboard** and select **Spanner1** if it is not already selected. You will see that no simulation exists for **Spanner1**. Click the link entitled **here** to generate a simulation.
- Under **Simulate Data**, provide the following details:

| Item              | Value    |
|-------------------|----------|
| Select Company    | Spanner1 |
| Select Interval   | 5        |
| Number of Records | 50       |

Then click **Simulate**.

- Navigate to the **Dashboard** which will immediately update the chart for **Spanner1** as the simulation progresses. It will take between 3 and 6 minutes for the simulation to complete.

11. The OmegaTrade application also allows you to modify existing company information. On the **Manage Company** tab click the pencil icon under **Action** for **Acme Corp**.
12. Update the company name to **Coyote Inc**. Notice that the **Short Code** cannot be updated in the user interface. Click **Update** to close and accept the change.
13. The update to the company name is immediate. Navigate to the **Dashboard** and you will see that **Acme Corp** no longer appears and **Coyote Inc** has taken its place.
14. Occasionally data changes are required that exceed the capabilities of the application code. As an empowered user you have the ability to update data used in the OmegaTrade application by making direct changes in the Cloud Spanner database.
15. In the Cloud Console, click **Navigation menu (≡) > View All Products > Databases > Spanner**.
16. Accept any acknowledgement or information window that may appear.
17. Click on the **omegatrade-instance** name and then **omegatrade-db** under **Databases**. From the list of tables on the bottom of the page, click **companies**.
18. Click **Data** on the left side pane to see the table contents.
19. You will update the name of the **Bar Industries** entity.
20. Click on the **checkbox** for the **Bar Industries** row. Then click **Edit** from the choices listed above the table rows.
21. You are taken to the **Spanner Studio**. Click on + **New tab**. Enter the following query to update the value for **companyName** from **Bar Industries** to **Consolidated Enterprises Inc**.

```
UPDATE
 companies
SET
 companyName='Consolidated Enterprises Inc'
WHERE
 companyName='Bar Industries';
```

**Note:** In the Cloud Spanner query window you have the ability to change the **Short Code** value. The application code as designed does not allow this value to be updated via the user interface.

22. Click **Run** to process the update.

23. Return to the application page, refresh your browser, and navigate to **Dashboard** tab. You will see that **Bar Industries** no longer appears and **Consolidated Enterprises Inc** has taken its place.