Deploy a Modern Web App connected to a Cloud Spanner Instance

Objective:

To deploy a simulated stock trading application (OmegaTrade) using Google Cloud managed services (Cloud Spanner, Cloud Run, Artifact Registry) to understand scalable, serverless application deployment, database interaction, and CI/CD pipeline concepts.

Tools & Services Used:

- **Google Cloud Shell**
- **Cloud Spanner** (managed relational DB)
- **Artifact Registry** (container image storage)
- **Cloud Run** (serverless deployment)
- **Container Registry** (for images)
- **Node.js, Angular** (for backend & frontend)
- **Docker**
- **Git**

Task 1. Enable required Google Cloud APIs

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

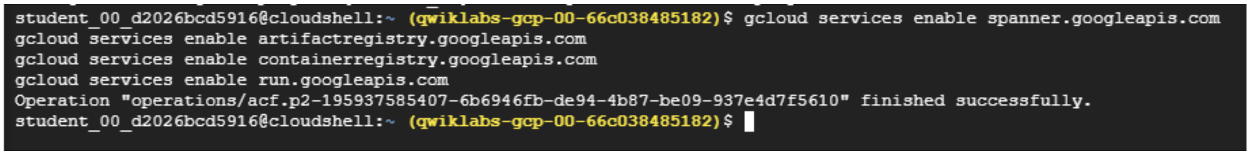
1. In the **Cloud Shell** enter the following commands:

   gcloud services enable spanner.googleapis.com

   gcloud services enable artifactregistry.googleapis.com

   gcloud services enable containerregistry.googleapis.com

   gcloud services enable run.googleapis.com

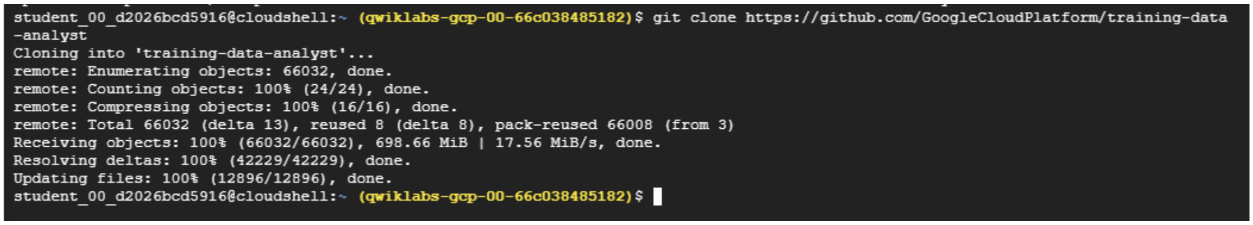


```
student_00_d2026bcd5916@cloudshell:~ (qwiklabs-gcp-00-66c038485182)$ gcloud services enable spanner.googleapis.com
gcloud services enable artifactregistry.googleapis.com
gcloud services enable containerregistry.googleapis.com
gcloud services enable run.googleapis.com
Operation "operations/acf.p2-195937585407-6b6946fb-de94-4b87-be09-937e4d7f5610" finished successfully.
student_00_d2026bcd5916@cloudshell:~ (qwiklabs-gcp-00-66c038485182)$ 
```

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:

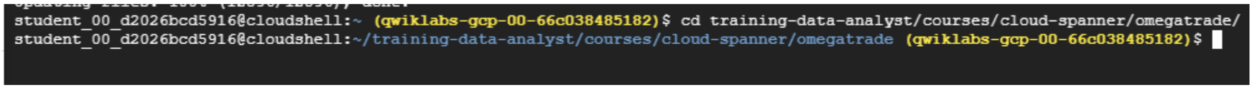
   git clone https://github.com/GoogleCloudPlatform/training-data-analyst

```
student_00_d2026bcd5916@cloudshell:~ (qwiklabs-gcp-00-66c038485182)$ git clone https://github.com/GoogleCloudPlatform/training-data
-analyst
Cloning into 'training-data-analyst'...
remote: Enumerating objects: 66032, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 66032 (delta 13), reused 8 (delta 8), pack-reused 66008 (from 3)
Receiving objects: 100% (66032/66032), 698.66 MiB | 17.56 MiB/s, done.
Resolving deltas: 100% (42229/42229), done.
Updating files: 100% (12896/12896), done.
student_00_d2026bcd5916@cloudshell:~ (qwiklabs-gcp-00-66c038485182)$
```

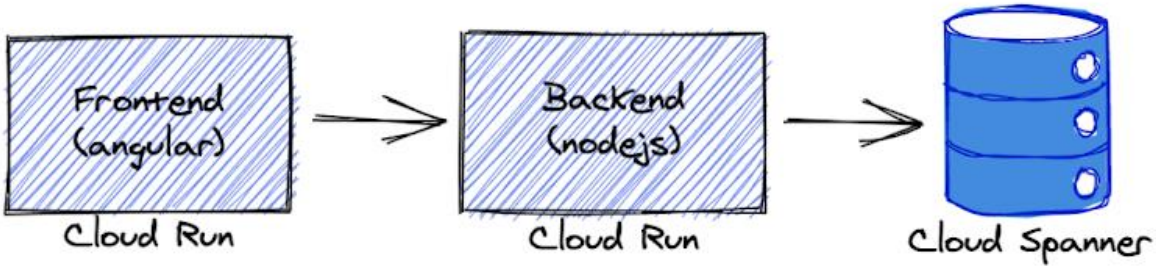
2. Navigate to the folder containing the application code.

   cd training-data-analyst/courses/cloud-spanner/omegatrade/

```
student_00_d2026bcd5916@cloudshell:~ (qwiklabs-gcp-00-66c038485182)$ cd training-data-analyst/courses/cloud-spanner/omegatrade/
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade (qwiklabs-gcp-00-66c038485182)$
```

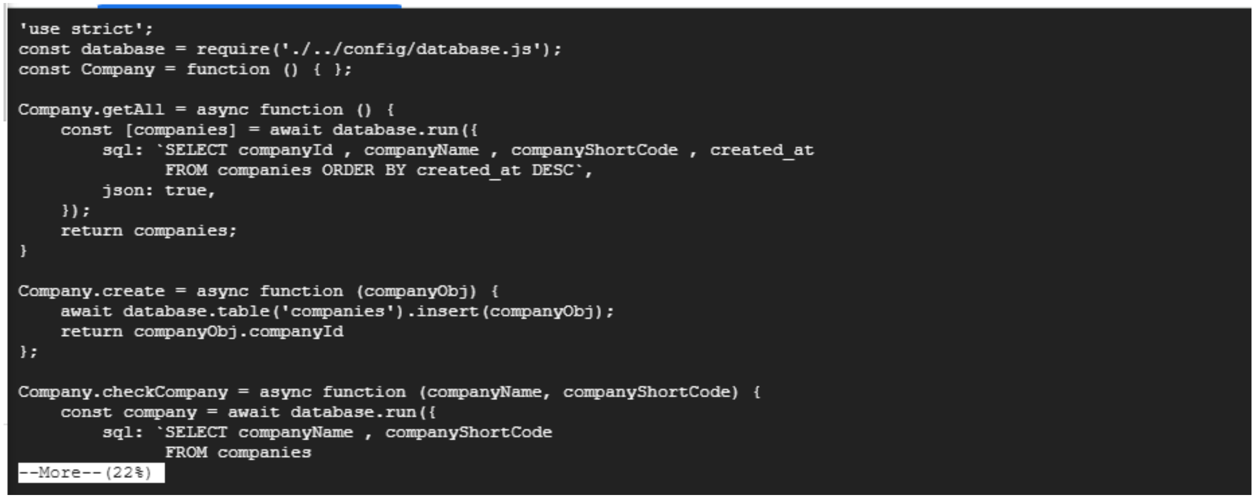
3. The code is divided primarily into two parts, backend and frontend. The application architecture is depicted in the diagram below:



4. The application relies on a deployment-specific file named **.env** to successfuly 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:

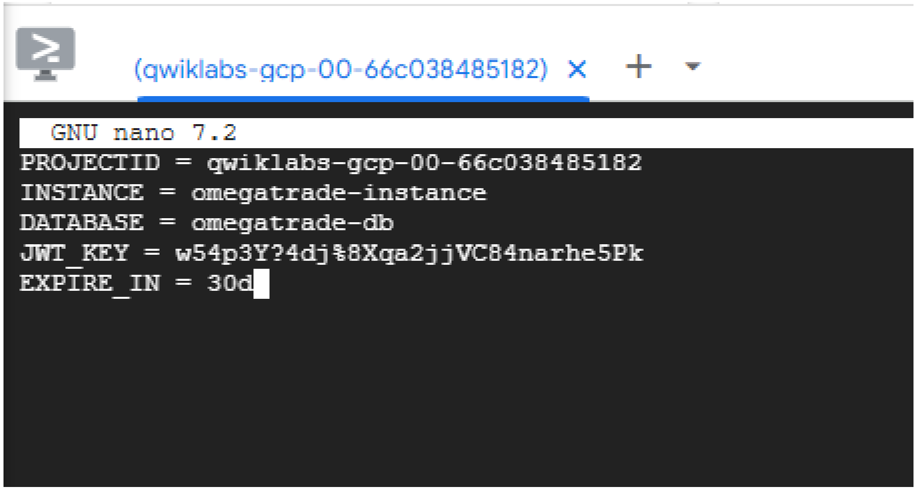
   cd backend/app/models



```
'use strict';
const database = require('./../config/database.js');
const Company = function () { };

Company.getAll = async function () {
    const [companies] = await database.run({
        sql: `SELECT companyId , companyName , companyShortCode , created_at
              FROM companies ORDER BY created_at DESC`,
        json: true,
    });
    return companies;
}

Company.create = async function (companyObj) {
    await database.table('companies').insert(companyObj);
    return companyObj.companyId
};

Company.checkCompany = async function (companyName, companyShortCode) {
    const company = await database.run({
        sql: `SELECT companyName , companyShortCode
              FROM companies
--More--(22%)
```

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



```
  GNU nano 7.2
PROJECTID = qwiklabs-gcp-00-66c038485182
INSTANCE = omegatrade-instance
DATABASE = omegatrade-db
JWT_KEY = w54p3Y?4dj%8Xqa2jjVC84narhe5Pk
EXPIRE_IN = 30d
```

   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

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 infromation 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.

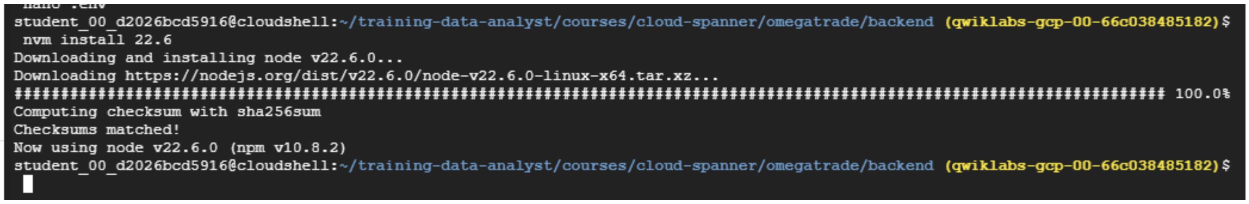
   ```
   nano .env
   ```

   ```
   PROJECTID = Project ID

   INSTANCE = omegatrade-instance

   DATABASE = omegatrade-db

   JWT_KEY = w54p3Y?4dj%8Xqa2jjVC84narhe5Pk

   EXPIRE_IN = 30d
   ```

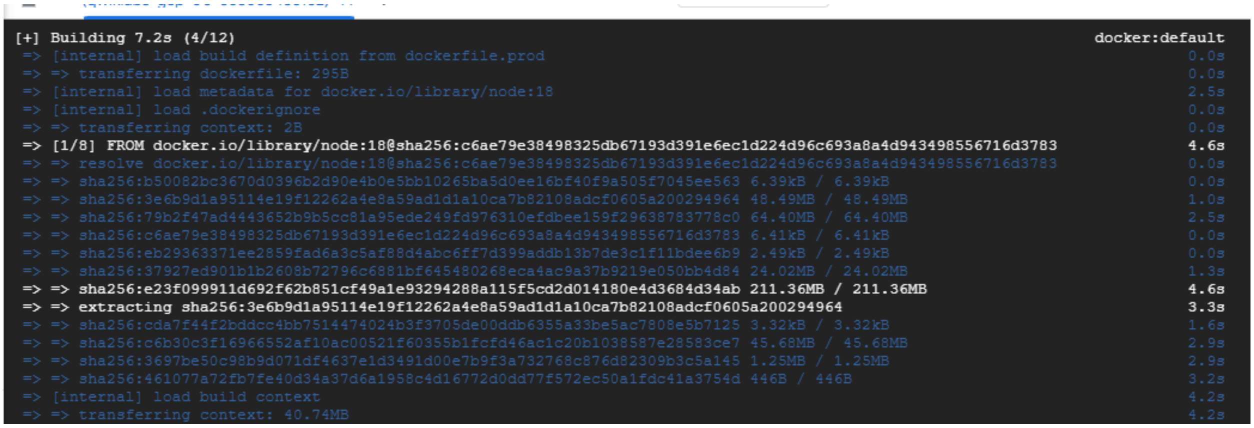
   Press **Ctrl+X** to exit Nano, **Y** to confirm the update, and press **Enter** to save your changes.

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
   ```



```
 nano .env
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/backend (qwiklabs-gcp-00-66c038485182)$
 nvm install 22.6
Downloading and installing node v22.6.0...
Downloading https://nodejs.org/dist/v22.6.0/node-v22.6.0-linux-x64.tar.xz...
############################################################################################################################## 100.0%
Computing checksum with sha256sum
Checksums matched!
Now using node v22.6.0 (npm v10.8.2)
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/backend (qwiklabs-gcp-00-66c038485182)$
```



```
[+] Building 7.2s (4/12)                                                                                      docker:default
 => [internal] load build definition from dockerfile.prod                                                              0.0s
 => => transferring dockerfile: 295B                                                                                   0.0s
 => [internal] load metadata for docker.io/library/node:18                                                             2.5s
 => [internal] load .dockerignore                                                                                      0.0s
 => => transferring context: 2B                                                                                        0.0s
 => [1/8] FROM docker.io/library/node:18@sha256:c6ae79e38498325db67193d391e6ec1d224d96c693a8a4d943498556716d3783       4.6s
 => => resolve docker.io/library/node:18@sha256:c6ae79e38498325db67193d391e6ec1d224d96c693a8a4d943498556716d3783       0.0s
 => => sha256:b50082bc3670d0396b2d90e4b0e5bb10265ba5d0ee16bf40f9a505f7045ee563 6.39kB / 6.39kB                         0.0s
 => => sha256:3e6b9d1a95114e19f12262a4e8a59ad1d1a10ca7b82108adcf0605a200294964 48.49MB / 48.49MB                       1.0s
 => => sha256:79b2f47ad4443652b9b5cc81a95ede249fd976310efdbee159f29638783778c0 64.40MB / 64.40MB                       2.5s
 => => sha256:c6ae79e38498325db67193d391e6ec1d224d96c693a8a4d943498556716d3783 6.41kB / 6.41kB                         0.0s
 => => sha256:eb29363371ee2859fad6a3c5af88d4abc6ff7d399addb13b7de3c1f11bdee6b9 2.49kB / 2.49kB                         0.0s
 => => sha256:37927ed901b1b2608b72796c6881bf645480268eca4ac9a37b9219e050bb4d84 24.02MB / 24.02MB                       1.3s
 => => sha256:e23f099911d692f62b851cf49a1e93294288a115f5cd2d014180e4d3684d34ab 211.36MB / 211.36MB                     4.6s
 => => extracting sha256:3e6b9d1a95114e19f12262a4e8a59ad1d1a10ca7b82108adcf0605a200294964                              3.3s
 => => sha256:cda7f44f2bddcc4bb7514474024b3f3705de00ddb6355a33be5ac7808e5b7125 3.32kB / 3.32kB                         1.6s
 => => sha256:c6b30c3f16966552af10ac00521f60355b1fcfd46ac1c20b1038587e28583ce7 45.68MB / 45.68MB                       2.9s
 => => sha256:3697be50c98b9d071df4637e1d3491d00e7b9f3a732768c876d82309b3c5a145 1.25MB / 1.25MB                         2.9s
 => => sha256:461077a72fb7fe40d34a37d6a1958c4d16772d0dd77f572ec50a1fdc41a3754d 446B / 446B                             3.2s
 => [internal] load build context                                                                                      4.2s
 => => transferring context: 40.74MB                                                                                   4.2s
```

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

   docker build -t gcr.io/Project ID/omega-trade/backend:v1 -f dockerfile.prod .

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.

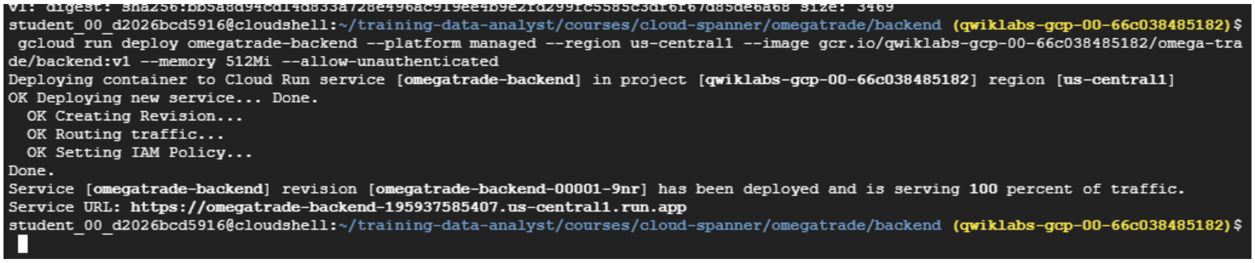
   gcloud auth configure-docker

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

   docker push gcr.io/Project ID/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 Default Region --image gcr.io/Project ID/omega-trade/backend:v1 --memory 512Mi --allow-unauthenticated



```
v1: digest: sha256:bb5a8d94cd14d833a728e496ac919ee4b9e21d299fc5585c3df6f67d85de6a68 size: 3469
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/backend (qwiklabs-gcp-00-66c038485182)$
 gcloud run deploy omegatrade-backend --platform managed --region us-central1 --image gcr.io/qwiklabs-gcp-00-66c038485182/omega-tra
de/backend:v1 --memory 512Mi --allow-unauthenticated
Deploying container to Cloud Run service [omegatrade-backend] in project [qwiklabs-gcp-00-66c038485182] region [us-central1]
OK Deploying new service... Done.
  OK Creating Revision...
  OK Routing traffic...
  OK Setting IAM Policy...
Done.
Service [omegatrade-backend] revision [omegatrade-backend-00001-9nr] has been deployed and is serving 100 percent of traffic.
Service URL: https://omegatrade-backend-195937585407.us-central1.run.app
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/backend (qwiklabs-gcp-00-66c038485182)$
```

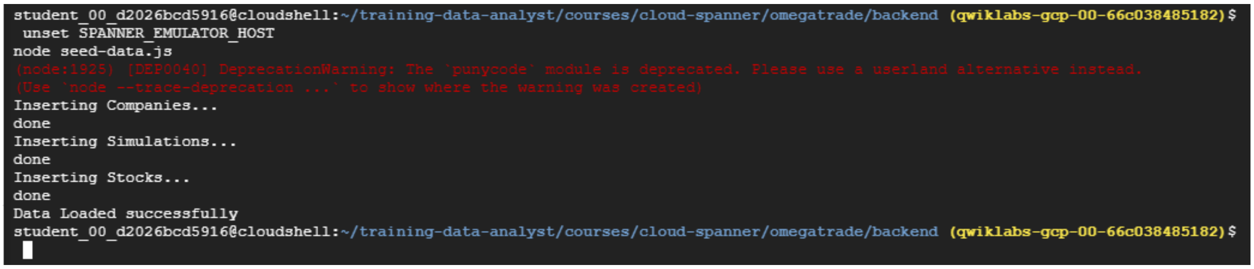
8. Copy the URL provided at the end of the deployment. Preserve the URL in notepad, a text file, or other readily accessible location. This URL will be supplied to the frontend application to ensure the user interface can properly communicate with the Cloud Spanner database via the backend.

   https://omegatrade-backend-195937585407.us-central1.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



```
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/backend (qwiklabs-gcp-00-66c038485182)$
 unset SPANNER_EMULATOR_HOST
node seed-data.js
(node:1925) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
Inserting Companies...
done
Inserting Simulations...
done
Inserting Stocks...
done
Data Loaded successfully
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/backend (qwiklabs-gcp-00-66c038485182)$
```

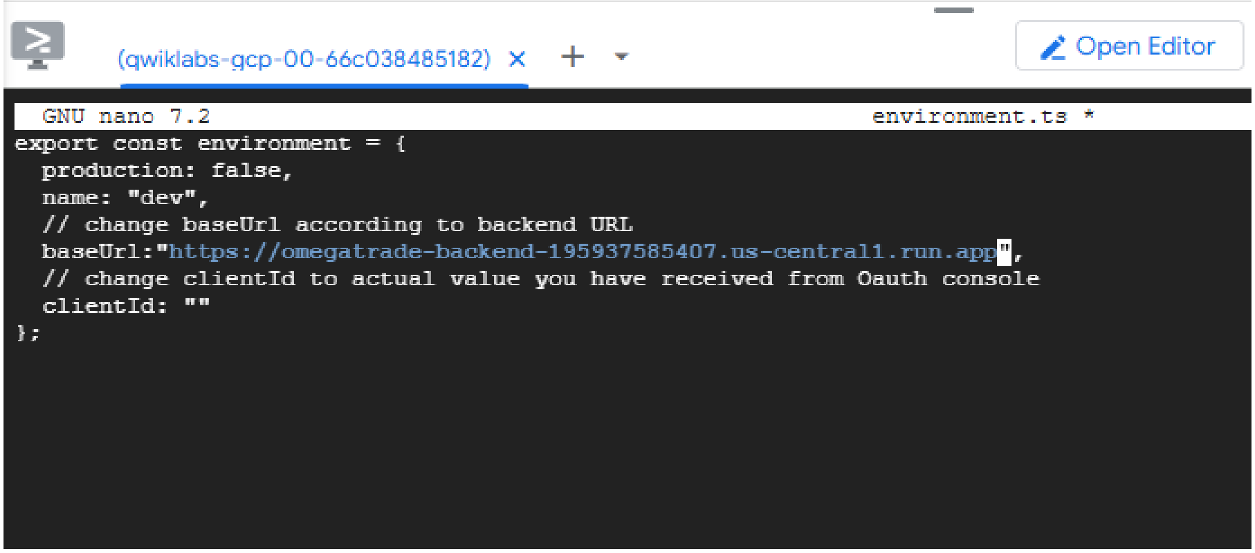
Task 5. Build and deploy the frontend component

1. Navigate to the directory containing the front-end 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

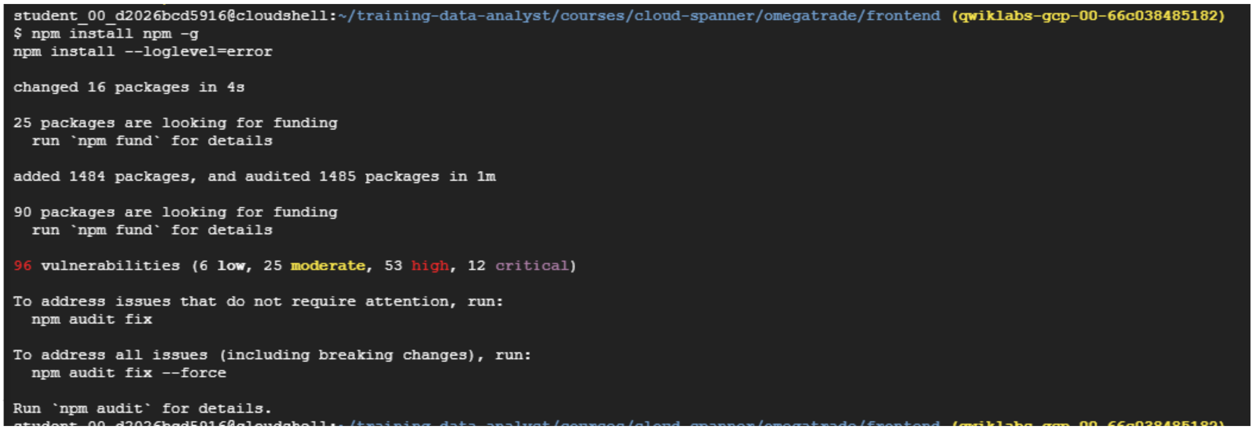


```
  GNU nano 7.2                                                environment.ts *
export const environment = {
  production: false,
  name: "dev",
  // change baseUrl according to backend URL
  baseUrl:"https://omegatrade-backend-195937585407.us-central1.run.app",
  // change clientId to actual value you have received from Oauth console
  clientId: ""
};
```

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

4. Navigate to the main frontend folder.

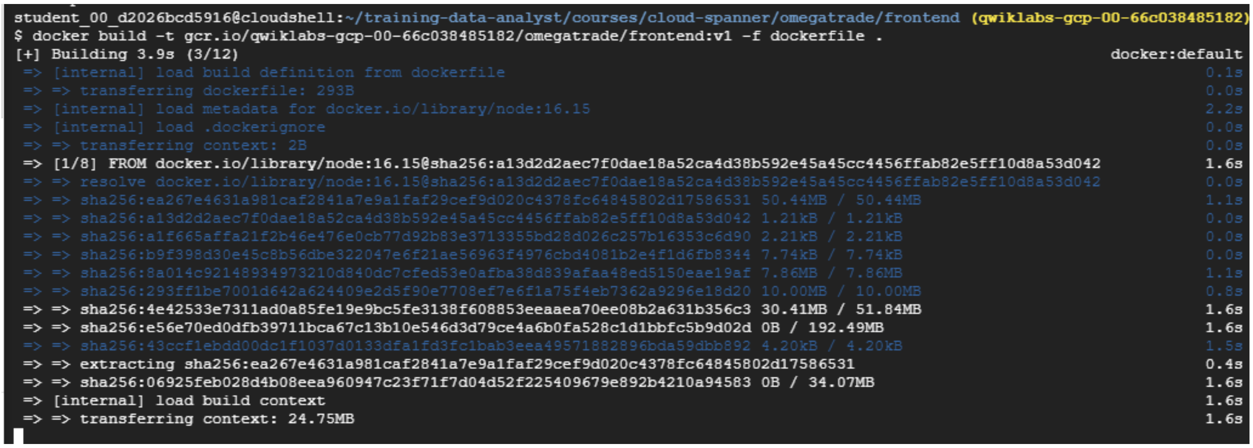
```
cd ../..
```



```
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/frontend (qwiklabs-gcp-00-66c038485182)
$ npm install npm -g
npm install --loglevel=error

changed 16 packages in 4s

25 packages are looking for funding
  run `npm fund` for details

added 1484 packages, and audited 1485 packages in 1m

90 packages are looking for funding
  run `npm fund` for details

96 vulnerabilities (6 low, 25 moderate, 53 high, 12 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/frontend (qwiklabs-gcp-00-66c038485182)
```

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

```
npm install npm -g

npm install --loglevel=error
```



```
student_00_d2026bcd5916@cloudshell:~/training-data-analyst/courses/cloud-spanner/omegatrade/frontend (qwiklabs-gcp-00-66c038485182)
$ docker build -t gcr.io/qwiklabs-gcp-00-66c038485182/omegatrade/frontend:v1 -f dockerfile .
[+] Building 3.9s (3/12)                                                                                         docker:default
 => [internal] load build definition from dockerfile                                                                       0.1s
 => => transferring dockerfile: 293B                                                                                       0.0s
 => [internal] load metadata for docker.io/library/node:16.15                                                              2.2s
 => [internal] load .dockerignore                                                                                          0.0s
 => => transferring context: 2B                                                                                            0.0s
 => [1/8] FROM docker.io/library/node:16.15@sha256:a13d2d2aec7f0dae18a52ca4d38b592e45a45cc4456ffab82e5ff10d8a53d042        1.6s
 => => resolve docker.io/library/node:16.15@sha256:a13d2d2aec7f0dae18a52ca4d38b592e45a45cc4456ffab82e5ff10d8a53d042        0.0s
 => => sha256:ea267e4631a981caf2841a7e9a1faf29cef9d020c4378fc64845802d17586531 50.44MB / 50.44MB                           1.1s
 => => sha256:a13d2d2aec7f0dae18a52ca4d38b592e45a45cc4456ffab82e5ff10d8a53d042 1.21kB / 1.21kB                             0.0s
 => => sha256:a1f665affa21f2b46e476e0cb77d92b83e3713355bd28d026c257b16353c6d90 2.21kB / 2.21kB                             0.0s
 => => sha256:b9f398d30e45c8b56dbe322047e6f21ae56963f4976cbd4081b2e4f1d6fb8344 7.74kB / 7.74kB                             0.0s
 => => sha256:8a014c92148934973210d840dc7cfed53e0afba38d839afaa48ed5150eae19af 7.86MB / 7.86MB                             1.1s
 => => sha256:293ff1be7001d642a624409e2d5f90e7708ef7e6f1a75f4eb7362a9296e18d20 10.00MB / 10.00MB                           0.8s
 => => sha256:4e42533e7311ad0a85fe19e9bc5fe3138f608853eeaaea70ee08b2a631b356c3 30.41MB / 51.84MB                           1.6s
 => => sha256:e56e70ed0dfb39711bca67c13b10e546d3d79ce4a6b0fa528c1d1bbfc5b9d02d 0B / 192.49MB                               1.6s
 => => sha256:43ccf1ebdd00dc1f1037d0133dfa1fd3fc1bab3eea49571882896bda59dbb892 4.20kB / 4.20kB                             1.5s
 => => extracting sha256:ea267e4631a981caf2841a7e9a1faf29cef9d020c4378fc64845802d17586531                                  0.4s
 => => sha256:06925feb028d4b08eea960947c23f71f7d04d52f225409679e892b4210a94583 0B / 34.07MB                                1.6s
 => [internal] load build context                                                                                          1.6s
 => => transferring context: 24.75MB                                                                                       1.6s
```

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/Project ID/omegatrade/frontend:v1 -f dockerfile .

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

   docker push gcr.io/Project ID/omegatrade/frontend:v1

8. Finally deploy the frontend application using Cloud Run.

   gcloud run deploy omegatrade-frontend --platform managed --region Default Region --image gcr.io/Project ID/omegatrade/frontend:v1 --allow-unauthenticated

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**.

7. **Spanner1** is now in the list of companies.

8. 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 simiulation.

9. Under **Simulate Data**, provide the following details:

Item	Value
Select Company	Spanner1

Select Interval	5
Number of Records	50

Then click **Simulate**.

10. 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 capabilites 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';
```

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.

    Outcome:

- Successfully deployed a full-stack application with **Cloud Spanner integration** on **Cloud Run**.
- Understood:
  - Containerization & pushing images to Artifact Registry.
  - Backend and frontend integration with Cloud Run services.
  - Cloud Spanner usage for direct SQL edits beyond app capabilities.
- Practiced **serverless scaling**, CI/CD pipeline patterns, and cloud application troubleshooting.