App Dev: Storing Application Data in Cloud Datastore v1.1

<https://googlepluralsight.qwiklabs.com/focuses/10986421?parent=lti_session>

Session Recorded: chrome-extension://cpaoncflippinhakkpipomcollmebloa/preview.html

Overview

In this lab, you will review the case study application, an online Quiz. You will store application data for the Quiz application in Cloud Datastore.

The Quiz application skeleton has already been written for you. You will clone a repository that contains the skeleton using Google Cloud Shell, review the code using the Cloud Shell editor, and view it using the Cloud Shell web preview feature.

Then you will modify the code that stores data to use Cloud Datastore.

Objectives

In this lab, you will learn how to perform the following tasks:

* Harness Cloud Shell as your development environment.
* Integrate Cloud Datastore into a NodeJS application.

A screenshot of a social media post

Description automatically generated

Previewing the Case Study Application

In this section, you will access Cloud Shell, clone the git repository containing the Quiz application, and run the application.

### **Clone source code in Cloud Shell**

1. On the **Cloud Platform Console**, click **Activate Google Cloud Shell**.
2. If a dialog box appears, click **Start Cloud Shell**.
3. To clone the repository for the class, execute the following command:

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

### **Configure and run the case study application**

1. To change the working directory, execute the following command:

cd ~/training-data-analyst/courses/developingapps/nodejs/datastore/start

1. To export an environment variable, GCLOUD\_PROJECT that references the GCP Project ID, execute the following command:

export GCLOUD\_PROJECT=$DEVSHELL\_PROJECT\_ID

**GCP Project ID in Cloud Shell**

While working in Cloud Shell, you will have access to the Project ID in the $DEVSHELL\_PROJECT\_ID environment variable.

1. To install the application dependencies, execute the following command:

npm install

The installation may take a couple of minutes.

1. To run the application, execute the following command:

npm start

### **Review the case study application**

1. In **Cloud Shell**, click **Web preview > Preview on port 8080** to preview the quiz application.

You should see the user interface for the web application. The three main parts to the application are:

1. Create Question
2. Take Test
3. Leaderboard
4. In the navigation bar, click **Create Question**.

You should see a simple form that contains textboxes for the question and answers with radio buttons to select the correct answer.

Quiz authors can add questions in this part of the application.

This part of the application is written as a server-side web application using the popular Node.js web application framework Express.

1. In the navigation bar, click **Take Test**.

You will be taken to a client application.

1. In the navigation bar, click **GCP**.

You should see a sample question.

Quiz takers can answer questions in this part of the application.

This part of the application is written as a client-side web application using the popular JavaScript framework AngularJS.

It receives JSON data from the server and uses JavaScript in the browser to display questions and collect answers.

1. To return to the server-side application, click on the **Quite Interesting Quiz** link in the navigation bar.

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

## Examining the Case Study Application Code

In this section, you will use the Cloud Shell text editor to review the case study application code.

### **Employ the Cloud Shell Editor**

1. From **Cloud Shell**, click **Launch code editor**.

The Orion integrated code editor will launch in a separate tab of your browser, along with Cloud Shell.

1. Navigate to the **/training-data-analyst/courses/developingapps/nodejs/datastore/start/server** folder using the file browser panel on the left side of the editor.

The application is a standard Node.js application written using the popular Express application framework.

### **Review the Express Web application**

1. Select the **.../datastore/start/server/app.js** file.

This file contains the entrypoint for the Express application and registers the main application routes for creating questions and delivering quizzes to users.

1. Select the **.../datastore/start/server/web-app** folder.

This folder contains the Node.js modules for the server-side web application.

1. Select the **.../datastore/start/server/web-app/questions.js** file.

This file contains the handlers that display the form and collect form data posted by quiz authors in the web application.

1. In the **questions.js** file, find the handler that responds to HTTP POST requests for the /questions/add route.

Review the source code comments for a detailed description of this handler's functionality.

1. Select the **.../datastore/start/server/web-app/views** folder.

This folder contains templates for the web application user interface using the pug (formerly jade) layout engine.

1. View the **.../datastore/start/server/web-app/views/questions/add.pug** file.

This file contains the pug template for the Create Question form.

Notice how there is a select list to pick a quiz, textboxes where an author can enter the question and answers, and radio buttons to select the correct answer.

1. Select the **.../datastore/start/server/api/index.js** file.

This file contains the handler that sends JSON data to students taking a test.

Notice that the handlers also make use of the model object.

1. Select the **.../datastore/start/server/gcp/datastore.js** file.

This is the file where you will write Datastore code to save and load quiz questions to and from Cloud Datastore.

This module will be imported into the web application and API as the model object for the web application and API

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a social media post

Description automatically generated

## Adding Entities to Cloud Datastore

In this section, you will write code to save form data in Cloud Datastore.

**Important**: Update code within the sections marked as follows:

// TODO

// END TODO

To maximize your learning, try to write the code without reference to the completed code block at the end of the section. In addition, review the code, inline comments, and related API documentation.

You can view the API documentation for Cloud Datastore at:

<https://googlecloudplatform.github.io/google-cloud-node/#/docs/datastore>

### **Create an App Engine application to provision Cloud Datastore**

1. Return to **Cloud Shell** and stop the application by pressing **Ctrl+C**.
2. To create an App Engine application in your project, execute the following command in **Cloud Shell**:

gcloud app create --region "us-central"

Note: You aren't using App Engine for your web application yet.

However, Cloud Datastore requires you to create an App Engine application in your project.

### **Import and use the NodeJS Datastore module**

1. Open the **...gcp/datastore.js** file in the Cloud Shell editor.
2. Load the config module from the parent folder.
3. Load the @google-cloud/datastore module.
4. Declare a Datastore client object named ds.

#### **datastore.js**

// TODO: Load the ../config module

const config = require('../config');

// END TODO

// TODO: Load the @google-cloud/datastore module

const Datastore = require('@google-cloud/datastore');

// END TODO

// TODO: Create a Datastore client object, ds

// The Datastore(...) factory function accepts an options // object which is used to specify which project's

// Datastore should be used via the projectId property.

// The projectId is retrieved from the config module. This // module retrieves the project ID from the GCLOUD\_PROJECT // environment variable.

const ds = Datastore({

projectId: config.get('GCLOUD\_PROJECT')

});

// END TODO

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

### **Write code to create a Cloud Datastore entity**

1. Declare a constant named kind, initialized with the value 'Question'.

#### **datastore.js**

// TODO: Declare a constant named kind

//The Datastore key is the equivalent of a primary key in a // relational database.

// There are two main ways of writing a key:

// 1. Specify the kind, and let Datastore generate a unique // numeric id

// 2. Specify the kind and a unique string id

const kind = 'Question';

// END TODO

1. In the create(...) function, remove the existing Promise.resolve({}) placeholder statement from the create(...) function.
2. Declare a constant called key to store the key for this entity.
3. Declare a constant named entity and initialize it with the key and the quiz question properties extracted from the form data.
4. Use the Datastore client object (ds) to save the entity by calling the save(entity) method.

#### **datastore.js**

// The create({quiz, author, title, answer1, answer2,

// answer3, answer4, correctAnswer}) function uses an

// ECMAScript 2015 destructuring assignment to extract

// properties from the form data passed to the function

function create({ quiz, author, title, answer1, answer2,

answer3, answer4, correctAnswer }) {

// TODO: Declare the entity key,

// with a Datastore generated id

const key = ds.key(kind);

// END TODO

// TODO: Declare the entity object, with the key and data

const entity = {

key,

// The entity's members are represented in a data property.

// This is an array where each element represents one

// member in the entity. Each element is an object with a // name and a value

data: [

{ name: 'quiz', value: quiz },

{ name: 'author', value: author },

{ name: 'title', value: title },

{ name: 'answer1', value: answer1 },

{ name: 'answer2', value: answer2 },

{ name: 'answer3', value: answer3 },

{ name: 'answer4', value: answer4 },

{ name: 'correctAnswer', value: correctAnswer },

]

};

// END TODO

// TODO: Save the entity, return a promise

// The ds.save(...) method returns a Promise to the

// caller, as it runs asynchronously.

return ds.save(entity);

// END TODO

}

A screenshot of a cell phone

Description automatically generated

### **Run the application and create a Cloud Datastore entity**

1. Save the **...gcp/datastore.js** file and then return to the Cloud Shell command prompt.
2. To start the application, execute the following command:

npm start

1. In **Cloud Shell**, click **Web preview** > **Preview on port 8080** to preview the quiz application.
2. Click **Create Question**.
3. Complete the form with the following values, and then click **Save**.

|  |  |
| --- | --- |
| **Form Field** | **Value** |
| Author | Your Name |
| Quiz | Google Cloud Platform |
| Title | Which company owns GCP? |
| Answer 1 | Amazon |
| Answer 2 | **Google** (select the Answer 2 radio button!) |
| Answer 3 | IBM |
| Answer 4 | Microsoft |

You should be returned to the home page for the application.

1. Return to the **Cloud Platform Console** and, on the **Navigation menu**, click **Datastore**.
2. On the **Datastore** page, click **Entities**.

You should see your new question!

**Review**

Choose the two correct options to create a key in Cloud Datastore.

1. You can specify the kind and let Cloud Datastore generate a unique numeric ID.
2. You can specify the kind and a unique string ID.
3. You can ask Cloud Datastore to leave the key as a null value.
4. You can create an entity without a key.

Which method enables you to store entities in Cloud Datastore?

1. persist
2. save

Click Check my progress to verify the objective.

Create an App Engine application and add entities to Cloud Datastore

Check my progress

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

## Bonus: Querying Cloud Datastore

In this section, you will write code to retrieve entity data from Cloud Datastore.

### **Write code to retrieve Cloud Datastore entities**

1. Return to the code editor.
2. In the **...gcp/datastore.js** file, remove the code from the list(quiz)method.
3. In the list(...) method, a query that retrieves Question entities for a specific quiz from Cloud Datastore.
4. Use the Datastore client to run the query, and assign the returned promise object to a constant.
5. Write a statement to return the promise.
6. Chain a then(...) method to the promise.
7. Write an arrow function in the then(...) method to retrieve the response from Cloud Datastore.
8. In the arrow function extract the results from the response.
9. Reshape the data by adding each entity ID and removing the correct answer from the data returned from Cloud Datastore.
10. Complete the code in the arrow function body to return a page of entities or an object that indicates that this is the last page of results.

### **Run the application and test the Cloud Datastore query**

1. Save the **...gcp/datastore.js** file, and then return to the Cloud Shell command.
2. Stop the application by pressing **Ctrl+C**.
3. Start the application.
4. In **Cloud Shell**, click **Web preview > Preview on port 8080** to preview the quiz application.
5. Replace the querystring at the end of the application's URL with /api/quizzes/gcp.

The URL will be in the form: https://8080-dot-####-dot-devshell.appspot.com/api/quizzes/gcp

You should see that JSON data has been returned to the client corresponding to the question you added in the web application!

1. Return to the application home page, and click **Take Test**.
2. Click **GCP**.

You should see that the quiz question has been formatted inside the client-side web application!

You can find the solution to the bonus in the lab's **bonus** folder.

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated