App Dev: Storing Application Data in Cloud Datastore - Python

GSP184

Overview

Google Cloud Datastore is a NoSQL document database built for automatic scaling, high performance, and ease of application development. In this lab, you use Datastore to store application data for an online Quiz application. You also configure the application to retrieve from Datastore and display the data in the quiz.
The Quiz application skeleton has already been written. You clone the 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. You then modify the code that stores data to use Cloud Datastore.

Objectives

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

- Harness Cloud Shell as your development environment
- Preview the application
- Update the application code to integrate Cloud Datastore

Qwiklabs setup

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new,

temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

What you need

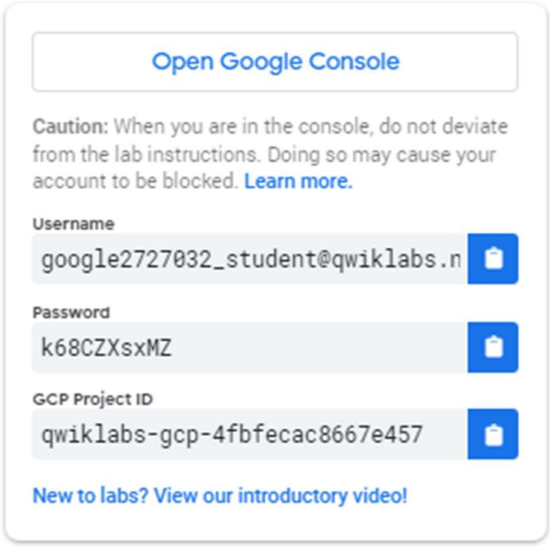
To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.
  Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

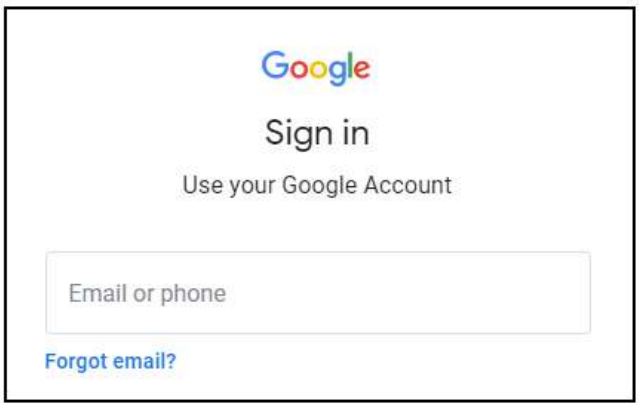
Note: If you are using a Pixelbook, open an Incognito window to run this lab.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.

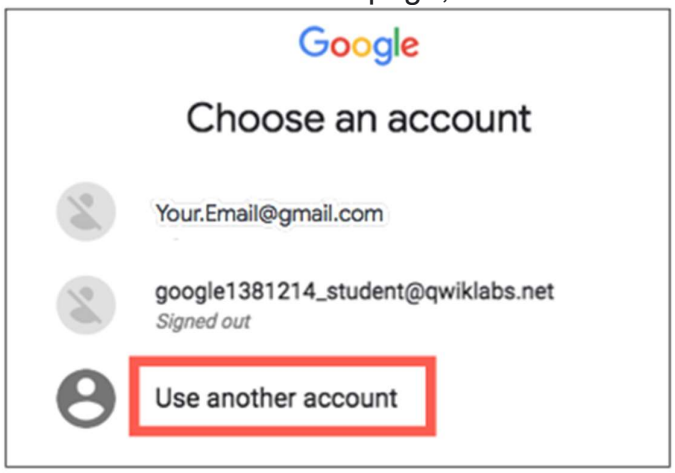


2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



Tip: Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another Account**.



3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

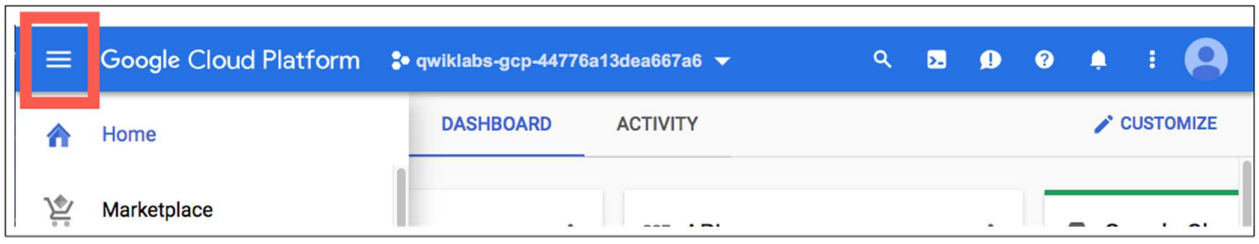
   Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

4. Click through the subsequent pages:

   - Accept the terms and conditions.
   - Do not add recovery options or two-factor authentication (because this is a temporary account).
   - Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

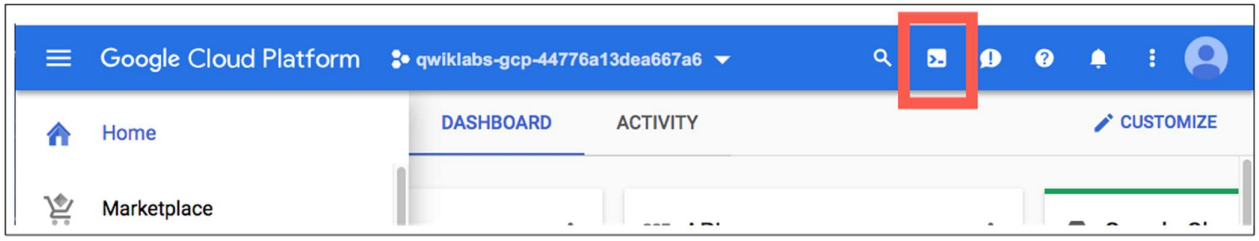
Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



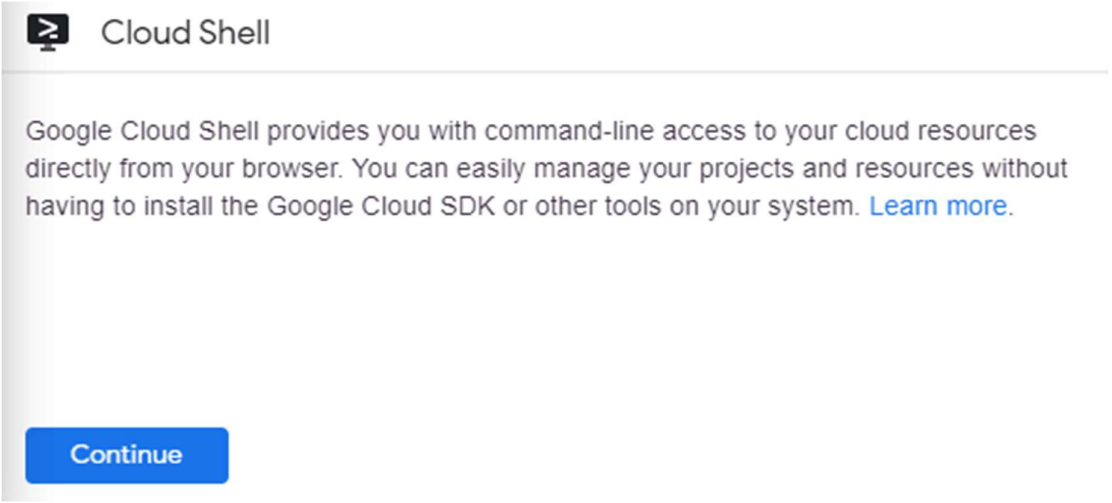
Activate Cloud Shell

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.

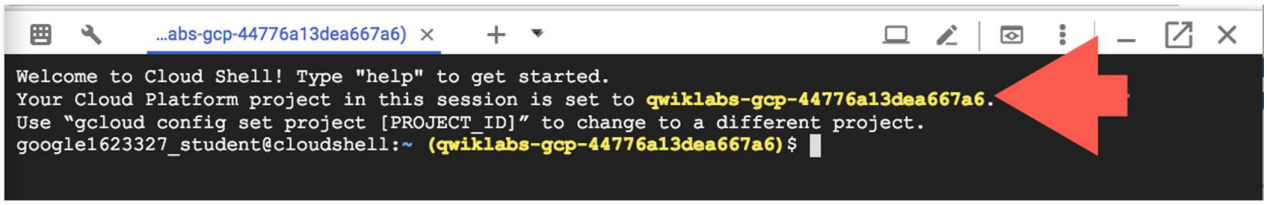
In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



Click **Continue**.



It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT_ID*. For example:



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

You can list the active account name with this command:

```
gcloud auth list
```
(Output)

```
Credentialed accounts:
 - <myaccount>@<mydomain>.com (active)
```
(Example output)

```
Credentialed accounts:
 - google1623327_student@qwiklabs.net
```
You can list the project ID with this command:

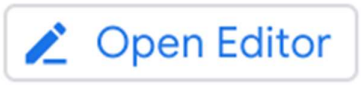
```
gcloud config list project
```
(Output)

```
[core]
project = <project_ID>
```
(Example output)

```
[core]
project = qwiklabs-gcp-44776a13dea667a6
```
For full documentation of `gcloud` see the gcloud command-line tool overview.

Launch the Cloud Shell code editor

From Cloud Shell, click **Open Editor** icon to launch the code editor. You may need to click on **Open In New Window**.



The code editor launches in a separate tab of your browser.

Create a virtual environment

Click on the **Open Terminal** icon.

Python virtual environments are used to isolate package installation from the system.

```
virtualenv -p python3 vrenv
```

Activate the virtual environment

```
source vrenv/bin/activate
```

Prepare the Quiz application

The repository containing the Quiz application is located on GitHub.com. In this section, you use Cloud Shell to enter commands that clone repository and run the application.

Clone source code in Cloud Shell

Clone the repository for the class:

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

Configure and run the Quiz application

1. Change the working directory:

```
cd ~/training-data-analyst/courses/developingapps/python/datastore/start
```

2. Export an environment variable, `GCLOUD_PROJECT` that references the Project ID:

   ```
   export GCLOUD_PROJECT=$DEVSHELL_PROJECT_ID
   ```

   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.

3. Install the application dependencies:

```
4. pip install -r requirements.txt
```

Ignore the incompatibility warnings.

5. Run the application:

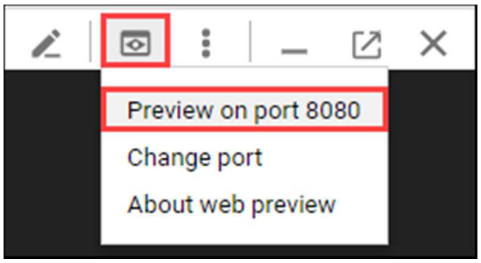
```
6. python run_server.py
```

The application is running When you see a message similar to the following:

```
* Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 179-313-240
```

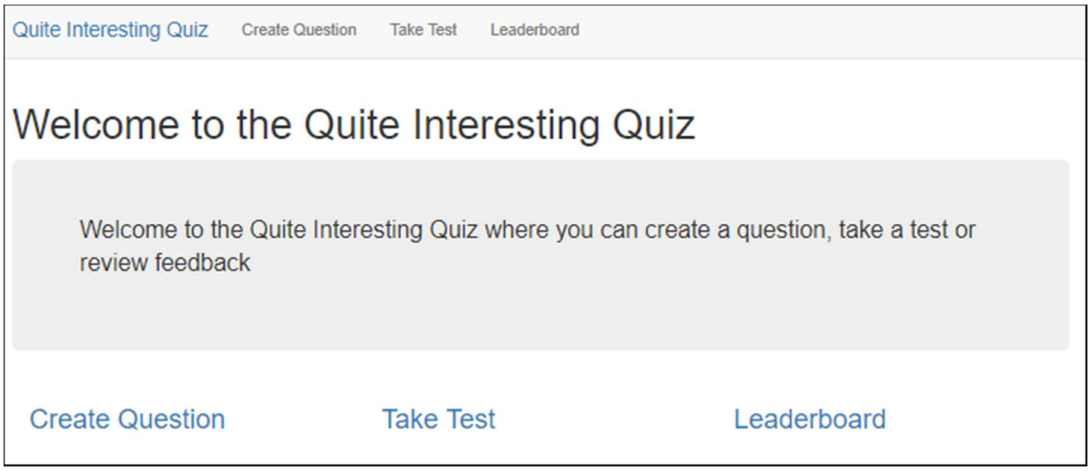
Review the Quiz application

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:

- **Create Question**
- **Take Test**
- **Leaderboard**



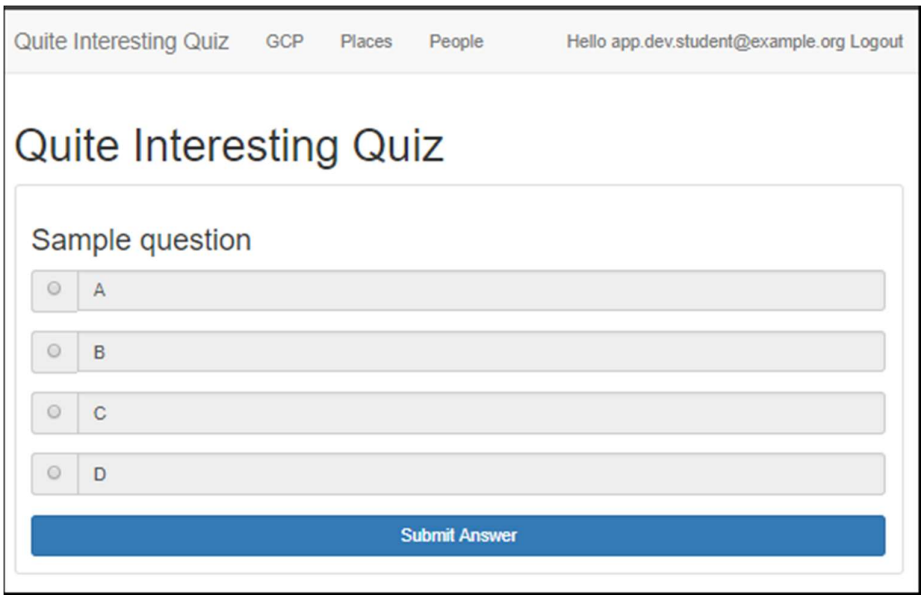
2. In the navigation bar, click **Create Question**.

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 Python web application framework Flask.

3. In the navigation bar, click **Take Test**, and then **GCP** to access the Google Cloud questions.

   You should see a sample question.



   Quiz takers answer questions in this part of the application.

   This part of the application is written as a client-side web application.

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

Examine the Quiz Application Code

In this lab you'll view and edit files. You can use the shell editors that are installed on Cloud Shell, such as `nano` or `vim` or the Cloud Shell code editor.

This lab uses the Cloud Shell code editor to review the Quiz application code.

Review the Flask Web application

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

   Note: Paths will be relative to this folder. This application is a standard Python application written using the popular Flask application framework.

2. Select the `...run-server.py` file.

   This file contains the entrypoint for the application, and runs it on port 8080.

3. Select the `...quiz/init.py file`.

   This file imports routes for the web application and REST API.

4. Select the `...quiz/webapp/questions.py` and `...quiz/webapp/routes.py` file.

   These files contain the routes that map URIs to handlers that display the form and collect form data posted by quiz authors in the web application.

5. Select the `...quiz/webapp/templates` folder.

   This folder contains templates for the web application user interface using Jinja2 templates.

6. View the `...quiz/webapp/templates/add.html` file.

   This file contains the Jinja2 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.

7. Select the `...quiz/api/api.py` file.

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

8. Select the `...quiz/gcp/datastore.py` file.

   This is the file where you 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.

Adding Entities to Cloud Datastore

In this section, you 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 for Cloud 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:

```
3. gcloud app create --region "us-central"
```

You'll see this message when the App Engine has been created:

```
Creating App Engine application in project [qwiklabs-gcp-f67238775c00cfaa] and region
[us-central]....done.
Success! The app is now created. Please use `gcloud app deploy` to deploy your first
app.
```

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.

Click **Check my progress** below to check your lab progress.

Import and use the Python Datastore module

Open the `...quiz/gcp/datastore.py` file in the Cloud Shell editor and add the following code to perform the following:

- Import the `os` module.
- Use the os module to get the `GCLOUD_PROJECT` environment variable.
- Import the `datastore` module from the `google.cloud` package.
- Declare a `datastore.Client` client object named `datastore_client`.

Updated datastore.py

```
# TODO: Import the os module

import os

# END TODO

# TODO: Get the GCLOUD_PROJECT environment variable

project_id = os.getenv('GCLOUD_PROJECT')

# END TODO

from flask import current_app

# TODO: Import the datastore module from the google.cloud package

from google.cloud import datastore

# END TODO

# TODO: Create a Cloud Datastore client object
# The datastore client object requires the Project ID.
# Pass through the Project ID you looked up from the
# environment variable earlier

datastore_client = datastore.Client(project_id)

# END TODO
```

Write code to create a Cloud Datastore entity

Still in `...quiz/gcp/datastore.py`,

Move to the `save_question()` function and remove the existing `pass` placeholder statement. Add the following code to perform the following:

- Use the Datastore client object to create a key for a Datastore entity whose kind is `'Question'`.
- Use Datastore to create a Datastore question entity with the key.
- Iterate over the items in the dictionary of values supplied from the Web application form.
- In the body of the loop, assign each key and value to the Datastore entity object.
- Use the Datastore client to save the data.

datastore.py - save_question() function

```
"""
Create and persist and entity for each question
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
"""
def save_question(question):
# TODO: Create a key for a Datastore entity
# whose kind is Question

    key = datastore_client.key('Question')

# END TODO

# TODO: Create a Datastore entity object using the key

    q_entity = datastore.Entity(key=key)

# END TODO

# TODO: Iterate over the form values supplied to the function

    for q_prop, q_val in question.items():

# END TODO

# TODO: Assign each key and value to the Datastore entity

        q_entity[q_prop] = q_val

# END TODO

# TODO: Save the entity

    datastore_client.put(q_entity)

# END TODO
```

Save `datastore.py`.

Run the application and create a Cloud Datastore entity

1. Save the `...quiz/gcp/datastore.py` file and then return to the Cloud Shell command prompt.

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

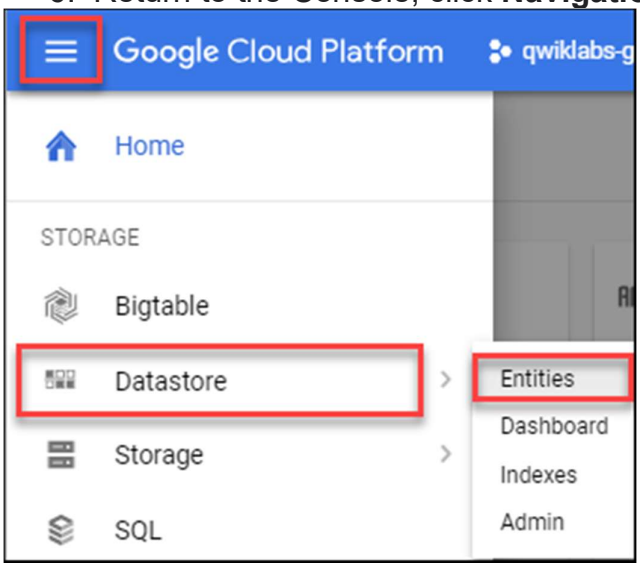
```
python run_server.py
```

3. In **Cloud Shell**, click **Web preview** > **Preview on port 8080** to preview the quiz application.
4. Click **Create Question**.
5. 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 Google Cloud?`
Answer 1	`Amazon`
Answer 2	**`Google`** (select the Answer 2 radio button!)
Answer 3	`IBM`
Answer 4	`Microsoft`

You should returned to the application home page.

6. Return to the Console, click **Navigation menu** > **Datastore** > **Entities**.



You should see your new question!

Click **Check my progress** below to check your lab progress.

Retrieve Cloud Datastore entities

In this section, you write code to retrieve entity data from Cloud Datastore to view your question in the application.

Write code to retrieve Cloud Datastore entities

In the code editor, in the `...quiz/gcp/datastore.py` file, remove the code for the `list_entities(quiz, redact)` function and replace it with a query that:

- Retrieves Question entities for a specific quiz from Cloud Datastore.
- Uses the Datastore client to fetch the query, and uses the returned data to create a list.
- Enumerate the list of items, and promote each entity's Key identifier to a top level property.
- Return the results.

Replace this code:

```
"""
Returns a list of question entities for a given quiz
- filter by quiz name, defaulting to gcp
- no paging
- add in the entity key as the id property
- if redact is true, remove the correctAnswer property from each entity
"""
def list_entities(quiz='gcp', redact=True):
    return [{'quiz':'gcp', 'title':'Sample question', 'answer1': 'A', 'answer2': 'B',
'answer3': 'C', 'answer4': 'D', 'correctAnswer': 1, 'author': 'Nigel'}]

"""
```

With this code:

```
"""
Returns a list of question entities for a given quiz
- filter by quiz name, defaulting to gcp
- no paging
- add in the entity key as the id property
- if redact is true, remove the correctAnswer property from each entity
"""
def list_entities(quiz='gcp', redact=True):
    query = datastore_client.query(kind='Question')
    query.add_filter('quiz', '=', quiz)
    results =list(query.fetch())
    for result in results:
        result['id'] = result.key.id
    if redact:
        for result in results:
            del result['correctAnswer']
    return results
"""
```

Save `datastore.py`.

Run the application and test the Cloud Datastore query

Now to test if your question is retrieved from Datastore and loaded into your Quiz application.

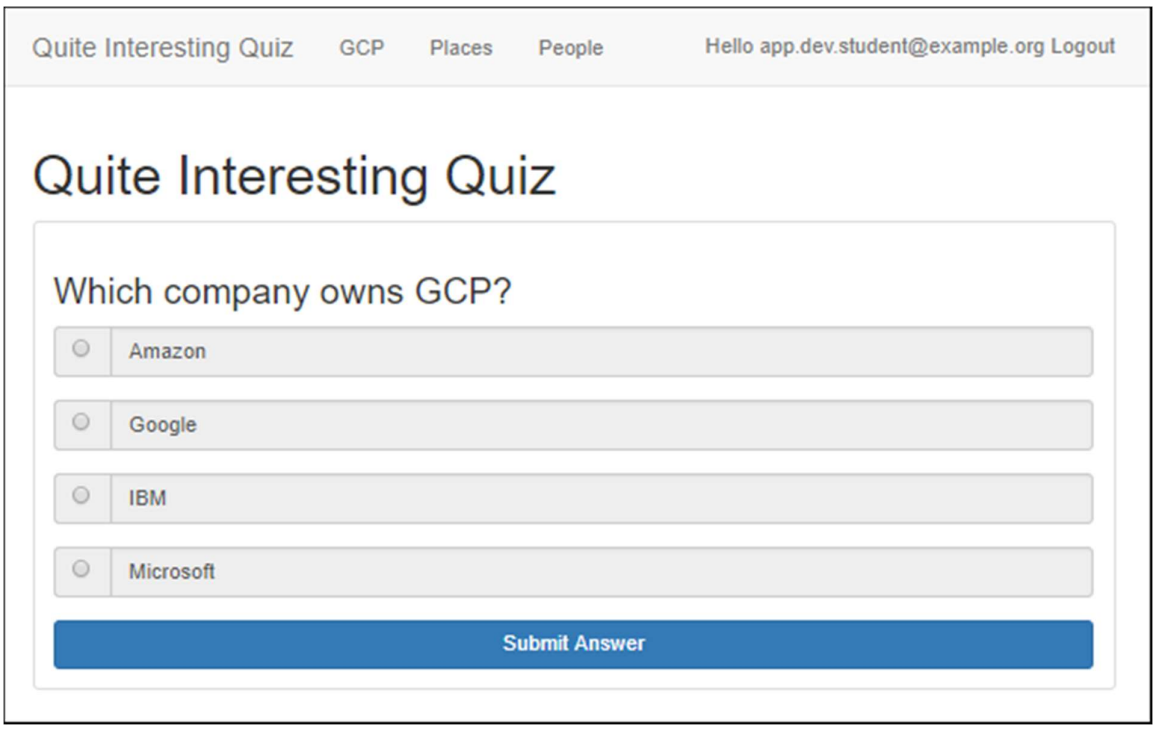
In Cloud Shell, press **Ctrl+c** to stop the application, then restart the application:

```
python run_server.py
```

Preview the quiz: If the browser running the quiz is still open, reload the browser. Otherwise, click **Web preview** > **Preview on port 8080**.

Click **Take Test** > **GCP**.

You should see the questions you created.



Congratulations!

This conclude the self-paced lab, App Dev: Storing Application Data in Cloud Datastore - Python. You used Datastore to store application data for an online Quiz application. You also configured the application to retrieve and display the data in the quiz.

Finish your Quest

This self-paced lab is part of the Application Development - Python and Cloud Development Quests. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in this Quest and get immediate completion credit if you've taken this lab. See other available Qwiklabs Quests.

Next steps / learn more

- For more information about Datastore, see Google Cloud Datastore Documentation.
- Learn more about Python on the Google Cloud.

Manual last updated December 01, 2020
Lab last tested December 01, 2020