Social Media Apps Programming

Google App Engine

1071SMAP11
TLMXM1A (8550) (M2143) (Fall 2018)
(MIS MBA) (2 Credits, Elective) [Full English Course]
Thu 8,9 (10:10-12:00) B206

Min-Yuh Day, Ph.D.
Assistant Professor

Department of Information Management
Tamkang University

http://mail.tku.edu.tw/myday

2018-12-06
<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Subject/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2018/09/13</td>
<td>Course Orientation and Introduction to Social Media and Mobile Apps Programming</td>
</tr>
<tr>
<td>2</td>
<td>2018/09/20</td>
<td>Introduction to Android / iOS Apps Programming</td>
</tr>
<tr>
<td>3</td>
<td>2018/09/27</td>
<td>Developing Android Native Apps with Java (Android Studio)</td>
</tr>
<tr>
<td>4</td>
<td>2018/10/04</td>
<td>Developing iPhone / iPad Native Apps with Swift (XCode)</td>
</tr>
<tr>
<td>5</td>
<td>2018/10/11</td>
<td>Mobile Apps using HTML5/CSS3/JavaScript</td>
</tr>
<tr>
<td>6</td>
<td>2018/10/18</td>
<td>jQuery Mobile</td>
</tr>
<tr>
<td>7</td>
<td>2018/10/25</td>
<td>Create Hybrid Apps with Phonegap</td>
</tr>
<tr>
<td>8</td>
<td>2018/11/01</td>
<td>jQuery Mobile/Phonegap</td>
</tr>
<tr>
<td>9</td>
<td>2018/11/08</td>
<td>jQuery Mobile/Phonegap</td>
</tr>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>10</td>
<td>2018/11/15</td>
<td>Midterm Exam Week / Project Presentation</td>
</tr>
<tr>
<td>11</td>
<td>2018/11/22</td>
<td>Case Study on Social Media Apps Programming and Marketing in Google Play and App Store</td>
</tr>
<tr>
<td>12</td>
<td>2018/11/29</td>
<td>Google Cloud Platform</td>
</tr>
<tr>
<td>13</td>
<td>2018/12/06</td>
<td>Google App Engine</td>
</tr>
<tr>
<td>14</td>
<td>2018/12/13</td>
<td>Google Map API</td>
</tr>
<tr>
<td>15</td>
<td>2018/12/20</td>
<td>Facebook API (Facebook JavaScript SDK) (Integrate Facebook with iOS/Android Apps)</td>
</tr>
<tr>
<td>16</td>
<td>2018/12/27</td>
<td>Twitter API</td>
</tr>
<tr>
<td>17</td>
<td>2019/01/03</td>
<td>Final Project Presentation</td>
</tr>
<tr>
<td>18</td>
<td>2019/01/10</td>
<td>Final Exam Week / Final Project Presentation</td>
</tr>
</tbody>
</table>
Outline

• Google App Engine
  – Google Cloud Platform
  – Google Cloud Datastore
  – Google Firebase
App
Backend
App
Frontend

iOS

Android

Web
Mobile Apps Backend on Google Cloud

Firebase

App Backend
App Backend

Frontend

iOS

Android

Web
Google Cloud Platform

Hosting + Compute

Storage

Big Data

Services

Source: https://cloud.google.com/products/
Google Cloud Platform

**Compute**
- App Engine
- Compute Engine
- Container Engine

**Storage**
- Cloud Datastore
- Cloud SQL
- Cloud Storage

**Big Data**
- Big Query

**Services**
- Cloud Endpoints
- Translate API
- Prediction API

Source: [https://cloud.google.com/products/](https://cloud.google.com/products/)
Mobile App Backend Services

iOS
Android
Web

Cloud Endpoints → App Engine → Cloud Datastore

Services → Compute → Storage

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Mobile App Backend Services

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Firebase

Update Data

Synchronize Across Mobile Devices

iOS

Android

Web

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Firebase and Google App Engine
standard environment

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Firebase and App Engine
flexible environment

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
App Engine and Cloud Endpoints

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Compute Engine and REST or gRPC

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Storing data and Exchanging data
• JSON
  – JavaScript Object Notation.

• JSON is a syntax for storing and exchanging data.

• JSON is an easier-to-use alternative to XML.

Source: http://www.w3schools.com/js/js_json_intro.asp
json

{"employees": [
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}

Source: http://www.w3schools.com/js/js_json_intro.asp
<employees>
  <employee>
    <firstName>John</firstName> <lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName> <lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName> <lastName>Jones</lastName>
  </employee>
</employees>

Source: http://www.w3schools.com/js/js_json_intro.asp
{"employees": [
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}

<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>

Source: http://www.w3schools.com/js/js_json_intro.asp
API
Application Programming Interface
Your App

API

Your App
Your App API Request
http://graph.facebook.com/4

Your App
Your App

Request

http://graph.facebook.com/4

API

Response

{  
  "id": "4",  
  "first_name": "Mark",  
  "gender": "male",  
  "last_name": "Zuckerberg",  
  "link": "https://www.facebook.com/zuck",  
  "locale": "en_US",  
  "name": "Mark Zuckerberg",  
  "username": "zuck"  
}
http://graph.facebook.com/4

```json
{
  "id": "4",
  "name": "Mark Zuckerberg",
  "first_name": "Mark",
  "last_name": "Zuckerberg",
  "link": "http://www.facebook.com/zuck",
  "username": "zuck",
  "gender": "male",
  "locale": "en_US"
}
```
Facebook API (JSON)
http://graph.facebook.com/4

{
"id": "4",
"first_name": "Mark",
"gender": "male",
"last_name": "Zuckerberg",
"link": "https://www.facebook.com/zuck",
"locale": "en_US",
"name": "Mark Zuckerberg",
"username": "zuck"
}
http://graph.facebook.com/minyuhday

{
    "id": "684393172",
    "name": "Min-Yuh Day",
    "first_name": "Min-Yuh",
    "last_name": "Day",
    "link": "http://www.facebook.com/minyuhday",
    "username": "minyuhday",
    "gender": "male",
    "locale": "en_US"
}
JavaScript vs. JSON

• JSON
  – JavaScript Object Notation
    • Format for sharing data
    • Derived from JavaScript
    • Language independent
    • An alternative to XML

Source: https://www.youtube.com/watch?v=40aKlrL-2V8
JSON

- Advantages
  - Easy to read
  - Easy to write
  - Easy to Parse
    ```javascript
    var info = JSON.parse(data);
    info.name
    info.position
    info.courses[i]
    ```
  - Learner than XML
  - Growing support in APIs (i.e., Facebook, Twitter)
  - Natural format for JavaScript
  - Implementation in many languages

Source: https://www.youtube.com/watch?v=40aKlrL-2V8
Min-Yuh Day

Assistant Professor

Social Media Apps Programming
Social Media Marketing
Data Mining

Source: https://www.youtube.com/watch?v=40aKIrL-2V8
App Engine and Cloud Endpoints

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Mobile Apps Backend on Google App Engine

Google Cloud Endpoints Architecture

Source: https://cloud.google.com/appengine/docs/java/endpoints/
Datasotre is a database (persistent storage) for App Engine.
Mobile Solutions on the Google Cloud Platform

- Your Application Code running on Google App Engine (GAE), Google Compute Engine (GCE), and Client Devices
- Google Cloud Platform Services
- Optional components
- Capabilities Included

Source: https://cloud.google.com/developers/articles/mobile-application-solutions/
Mobile App Solutions Architecture

Source: https://cloud.google.com/developers/articles/mobile-application-solutions/
Storing data

Mobile Backend Running On Google App Engine Instances

Source: https://cloud.google.com/developers/articles/mobile-application-solutions/
Optimizing data access with Memcache

Source: https://cloud.google.com/developers/articles/mobile-application-solutions/
Google App Engine
Platform as a Service (PaaS)

build and run applications on Google’s infrastructure

Source: https://cloud.google.com/appengine/docs
Google App Engine
Platform as a Service (PaaS)

Google Compute Engine
Infrastructure as a Service (IaaS)

Source: https://cloud.google.com/appengine/docs
Mobile App Backend Services

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Mobile App Backend Services

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Google App Engine

• 1 GB of data storage and traffic for free

• can be increased by enabling paid applications

Source: https://cloud.google.com/appengine/docs/whatisgoogleappengine
Google App Engine supports apps written in a variety of programming languages: Python, Java, PHP, Go

Source: https://cloud.google.com/appengine/docs/whatisgoogleappengine
Google Cloud Datastore

Cloud Datastore

Use a managed, NoSQL, schemaless database for storing non-relational data. Cloud Datastore automatically scales as you need it and supports transactions as well as robust, SQL-like queries.

Features

- Schemaless access, with SQL-like querying
- Managed database
- Autoscale with your users

Cloud Datastore is a schemaless storage service that allows you to be agile by removing the need to think about the underlying structure of the database.

Cloud Datastore automatically handles sharding and replication in order to provide you with a highly available and consistent database.

Cloud Datastore automatically scales depending on your needs. This allows you to focus on building your application and not on worrying about provisioning and load anticipation.

https://cloud.google.com/datastore/
Google Cloud Endpoints

Create RESTful services and make them accessible to iOS, Android and Javascript clients. Automatically generate client libraries to make wiring up the frontend easy. Built-in features include denial-of-service protection, OAuth 2.0 support and client key management.

Features

- One tool, multiple clients
  Build client libraries for Android, iOS and web-based clients from one source. Cloud Endpoints wraps your code to build an API server in just a few steps. Cloud Endpoint API libraries are available in Java, Python, Go and PHP. Learn more

- Extending App Engine infrastructure
  All of the tools and libraries made available in App Engine are now available to your mobile devices. Access Datastore, Cloud Storage and Task Queues using your App Engine backend with no extra

- Low maintenance client-server
  Because Cloud Endpoints is backed by App Engine, you have no servers to maintain, no load balancing to worry about and the same quick and painless scaling. Like App Engine and our other Cloud services, you only pay for what you use.

Source: https://cloud.google.com/endpoints/
Mobile Apps Backend on Google App Engine

Google Cloud Endpoints Architecture

Source: https://cloud.google.com/appengine/docs/java/endpoints/
Mobile App, Google App Engine, Cloud Datastore

Mobile Backend Starter

Your App
Mobile Backend Starter Client Library

Cloud Endpoints

Mobile Backend Starter
Cloud Datastore

Google cloud messaging

http://www.youtube.com/watch?v=v5u_Owtbfew
Try Google Cloud Platform for free

Build on top of the infrastructure that powers Google.
Sign up for free and get $300 to spend over 60 days on all Google Cloud Platform services.

Certain terms and conditions apply. Learn more

Country
Taiwan

Account type
- Business
- Individual

Name and address
- Business name
- Name
- Street address

Get $300 to kick start your app.
Sign up for free and get $300 to spend on Google Cloud Platform over the next 60 days.

Why do you need my billing information?
We use your billing information to verify that you're a real person. Don't worry, you will not be billed for the free trial.

Do I have to pay when my free trial ends?
No. You're under no obligation to buy anything when the free trial ends. If you want to continue to use Google Cloud Platform, just upgrade before your trial runs out.

Questions?
Check out the FAQ or leave us a message.
Try Google Cloud Platform for free

Google Developers Console

Primary contact
Name
Phone number
imyday@gmail.com

What you pay with
Credit or debit card
Card number
MM / YY
Cardholder name

Billing communication language
English (United States)

I have read and agree to the Google Cloud Platform Free Trial Terms of Service.

Accept and start free trial Clear
Get started quickly

If you’re ready to get started, try these solutions for getting software up and running on our platform!

**LAMP development stack**

LAMP (an acronym for Linux, Apache, MySQL, and PHP) is the archetypal open-source web development stack for many developers, and it runs great on Compute Engine!

**Ruby development stack**

Ruby on Rails is one of the most popular frameworks for developing web applications, powering sites like Github, Basecamp and Shopify. Rapidly create new features, easily maintain code, and take advantage of the many open source contributions to Ruby on Rails, running on Compute Engine.

**Quickstart for Wordpress**

Set up a project, download a zip, change your config file, and deploy—and you’ll have a working WordPress project running on Google Cloud Platform, with App Engine as your hosting environment.

**App Engine "Hello World" starter**

Start editing a working "Hello World" app right now, in the browser. This gives you a good starting point and a feel for what it’s like editing a working App Engine application.

Documentation

Use the following section or the left-hand navigation to access the various sets of documentation that cover Google Cloud Platform. Choose from computing and hosting, storage, big data, management, services, and developer tools.

Computing and hosting
Google Cloud Platform

https://cloud.google.com/docs/

Computing and hosting

**App Engine**

App Engine is Google's Platform-as-a-Service (PaaS). Develop your application easily using built-in services that make you more productive. Deploy to a fully-managed platform and let Google carry the pager. Just download the SDK and start building immediately for free with no credit card required.

- Overview
- Get Started
- Tutorials: Java, PHP, Python, Go
- Documentation

**Compute Engine**

Compute Engine is Google's Infrastructure-as-a-Service (IaaS). Run large-scale workloads on virtual machines hosted on Google's infrastructure. Choose a VM that fits your needs and gain the performance and consistency of Google's worldwide fiber network. With per-minute billing, you pay only for what you use.

- Overview
- Get Started
- Tutorial
- Documentation

Storage

**Cloud SQL**

Store and manage data using a relational MySQL database. Google handles replication, patch management and database management to ensure availability and performance, and you can even have your instance automatically co-locate with your deployed applications.

- Overview
- Tutorial
- Documentation

**Cloud Storage**

Use a durable and highly available object storage service. With global edge-caching, your users have fast access to your app's data from any location. Google manages versioning, guarantees a strong SLA and provides a simple API that allows you to manage your data programmatically.

- Overview
- Tutorial
- Documentation

**Cloud Datastore**

Cloud Datastore provides a managed, NoSQL, schemaless database for storing non-relational data. Cloud Datastore automatically scales as you need it and supports transactions as well as robust, SQL-like queries.

- Overview
- Tutorial
- Documentation
Mobile App Backend Services

iOS

Android

Web

Cloud Endpoints

App Engine

Cloud Datastore

Services

Compute

Storage

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Google App Engine

App Engine "Hello World" starter

https://console.developers.google.com/start/appengine

Deploy your first app in five minutes

• Start editing a working "Hello World" app right now, in the browser.

• This gives you a good starting point and a feel for what it's like editing a working App Engine application.
Try Google App Engine Now

1. NAME YOUR PROJECT
2. SELECT YOUR LANGUAGE
3. EXPLORE THE STARTER CODE
4. INSTALL GOOGLE CLOUD SDK
5. RUN YOUR APP LOCALLY
6. CREATE YOUR PROJECT AND DEPLOY
Try Google App Engine Now

Creating an App Engine app is easy, and it's free to start. Upload your app and share it with users right away, at no charge and with no commitment required.

1. NAME YOUR PROJECT

You use your project to manage all of the Google Cloud Platform resources for your app, including deployment, access control, billing, and services. You can change your project name later.

   HelloWorldGoogleAppEngine

2. SELECT YOUR LANGUAGE

   | Python | Java |
Google App Engine

2. SELECT YOUR LANGUAGE

- Python
- Java
- PHP
- Go

3. EXPLORE THE STARTER CODE

Browse the starter code and see the app running below.

```
HELLO WORLD - JAVA

package myapp;

import java.io.IOException;
import javax.servlet.http.*;

public class DemoServlet extends HttpServlet {
  @Override
  public void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws IOException {
    resp.setContentType("text/plain");
  }
}
```
Google App Engine

3 EXPLORE THE STARTER CODE

Browse the starter code and see the app running below.

HELLO WORLD - JAVA

```java
package myapp;

import java.io.IOException;
import javax.servlet.http.*;

public class DemoServlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println("\"name\": \"World\" ");
    }
}
```

Hello, World
Hello, World
Google App Engine

4 INSTALL GOOGLE CLOUD SDK

1. Download and install Google Cloud SDK by running the following command in your shell or Terminal:

   curl https://sdk.cloud.google.com/ | bash

   Follow the prompts to install the Java App Engine package.

2. Restart your shell or Terminal to pick up environment changes.

3. Sign in to Google Cloud Platform using this command:

   gcloud auth login

4. Install the App Engine package for Java using this command:

   gcloud components update gae-java

5 RUN YOUR APP LOCALLY
5  RUN YOUR APP LOCALLY

Note: App Engine supports Java 7. Make sure you have the Java 7 JDK installed.

1. Download appengine.try.java.zip and unpack it. This creates your project directory, including src/ and war/subdirectories.

2. Download and install Apache Maven version 3.0 or later. The starter app includes an Apache Maven build file.

3. Build and run the sample locally using the following commands:

```bash
cd appengine.try.java
mvn package
dev_appserver.sh target/appengine.try.java-1.0
```

You can stop the server by pressing Ctrl-C in the command window.

4. Visit the locally running app in your browser: http://localhost:8080

6  CREATE YOUR PROJECT AND DEPLOY

Now that you've seen your app running on your local machine, you're ready to create and deploy your project, HelloWorldGoogleAppEngine.
Google App Engine

5. RUN YOUR APP LOCALLY

Note: App Engine supports Java 7. Make sure you have the Java 7 JDK installed.

1. Download `appengine-try-java.zip` and unpack it. This creates your project directory, including `src/` and `war/` subdirectories.

2. Download and install Apache Maven version 3.0 or later. The starter app includes an Apache Maven build file.

3. Build and run the sample locally using the following commands:

   ```
   cd appengine-try-java
   mvn package
   dev_appserver.sh target/appengine-try-java-1.0
   ```

   You can stop the server by pressing Ctrl-C in the command window.

4. Visit the locally running app in your browser: `http://localhost:8080`

6. CREATE YOUR PROJECT AND DEPLOY

Now that you've seen your app running on your local machine, you're ready to create and deploy your project, `HelloWorldGoogleAppEngine`. 
Google App Engine

1. NAME YOUR PROJECT
2. SELECT YOUR LANGUAGE
3. EXPLORE THE STARTER CODE
4. INSTALL GOOGLE CLOUD SDK
5. RUN YOUR APP LOCALLY
6. CREATE YOUR PROJECT AND DEPLOY

mvn package
dev_appserver.sh target/appengine-try-java-1.0

You can stop the server by pressing Ctrl-C in the command window.

4. Visit the locally running app in your browser: http://localhost:8080

6 CREATE YOUR PROJECT AND DEPLOY

Congratulations! Your project is ready. Your unique project ID is psychic-outcome-783.

Deploy your app from your local dev environment using this command:

appcfg.sh -A psychic-outcome-783 update target/appengine-try-java-1.0

After deploying your app, you can visit it with your browser at this URL:

psychic-outcome-783.appspot.com

That's it! You're running on Google App Engine. Go to your project dashboard to see how your app is performing.

View your project dashboard

Activities (Idle)
Create Project: HelloWorldGoogleAppEngine

See all activity
Google App Engine
Build an App Engine Application using Python

Creating a Guestbook
1. Download the App Engine SDK
2. Explaining the webapp2 Framework
3. Using the Users Service
4. Handling Forms with webapp2
5. Using the Datastore
6. Using Templates
7. Using Static Files
8. Uploading Your Application

Source: https://cloud.google.com/appengine/docs/python/gettingstartedpython27/introduction
Google App Engine

Introduction

Welcome to Google App Engine! Creating an App Engine application is easy, only takes a few minutes, and it's free to start.

Google App Engine applications can be written in the Python 2.7, Java, Go or PHP programming languages. This tutorial covers Python 2.7. If you would prefer to use Java, Go or PHP to build your applications, see the Java, Go or PHP guides.

In this tutorial, you will learn how to:

- build an App Engine application using Python
- use the webapp2 web application framework
- use the App Engine datastore with the Python modeling API
- integrate an App Engine application with Google Accounts for user authentication
- use Jinja2 templates with your app
- upload your app to App Engine

By the end of the tutorial, you will have implemented a working application, a simple guest book that lets users post messages to a public message board.

Get set up

Before we continue, you will need to download the [App Engine Python SDK](https://cloud.google.com/appengine/docs/python/gettingstartedpython27/introduction), which includes a web server application that simulates the...
Introduction

Welcome to Google App Engine! Creating an App Engine application is easy, only takes a few minutes, and it's free to start.

Google App Engine applications can be written in the Python 2.7, Java, Go or PHP programming languages. This tutorial covers Python 2.7. If you would prefer to use Java, Go or PHP to build your applications, see the Java, Go or PHP guides.

In this tutorial, you will learn how to:

- build an App Engine application using Python
- use the webapp2 web application framework
- use the App Engine datastore with the Python modeling API
- integrate an App Engine application with Google Accounts for user authentication
- use Jinja2 templates with your app
- upload your app to App Engine

By the end of the tutorial, you will have implemented a working application, a simple guest book that lets users post messages to a public message board.

Get set up

Before we continue, you will need to download the App Engine Python SDK, which includes a web server application that simulates the App Engine environment, and tools to deploy your application to the App Engine production environment. Follow the directions for your operating system, then come back here so we can get going!
Explaining the webapp2 Framework

The Web Server Gateway Interface (WSGI) standard is simple, but it would be cumbersome to write all of the code that uses it by hand. Web application frameworks handle these details for you, so you can focus your development efforts on your application’s features. Google App Engine supports any framework written in pure Python that speaks WSGI, including Django, CherryPy, Pylons, web.py, and web2py. You can bundle a framework of your choosing with your application code by copying its code into your application directory.

App Engine includes a simple web application framework, called webapp2. The webapp2 framework is already installed in the App Engine environment and in the App Engine Python SDK, so you do not need to bundle it with your application code to use it. We will use webapp2 for the rest of this tutorial.

Hello, webapp2!

A webapp2 application has two parts:

- one or more RequestHandler classes that process requests and build responses
- a WSGIApplication instance that routes incoming requests to handlers based on the URL

Let’s take a look at Hello World!:

```python
import webapp2

class MainPage(webapp2.RequestHandler):
    def get(self):
        self.response.headers['Content-Type'] = 'text/plain'
        self.response.write('Hello, World!')

app = webapp2.WSGIApplication([]
```

Source: https://cloud.google.com/appengine/docs/python/gettingstartedpython27/introduction
Using the Users Service

Google App Engine provides several useful services based on Google infrastructure, accessible by applications using libraries included with the App Engine Python SDK. One such service is the Users service, which lets your application integrate with Google user accounts. With the Users service, your users can use the Google accounts they already have to sign in to your application.

Let's use the Users service to personalize this application's greeting.

Using Users

We're going to build on the Hello, World! application. You can clone the code for this step from this branch on GitHub, or if you already have the original Hello, World! app, replace its contents with the following:

```python
from google.appengine.api import users
import webapp2

class MainPage(webapp2.RequestHandler):
    def get(self):
        # Checks for active Google account session
        user = users.get_current_user()
        if user:
            self.response.headers['Content-Type'] = 'text/html; charset=utf-8'
            self.response.write('Hello, ' + user.nickname())
```

Source: https://cloud.google.com/appengine/docs/python/gettingstartedpython27/introduction
Handling Forms with webapp2

If we want users to be able to post their own greetings, we need a way to process information submitted by the user with a web form. The webapp2 framework makes processing form data easy.

From Hello World to Guestbook

In order to prepare the Hello World app we've created thus far, please make the following changes:

- Rename the top level helloworld directory to guestbook
- Rename helloworld.py to guestbook.py
- Replace the handlers section of app.yaml with:

```
app.yaml

handlers:
- url: /.*
  script: guestbook.app
```

Restart the development server using the new guestbook directory.

Handling Web Forms With webapp2

Declare that you are using webapp2 by adding this libraries section to your app.yaml:

```
app.yaml

libraries:
  - webapp2
```

Source: https://cloud.google.com/appengine/docs/python/gettingstartedpython27/introduction
Google App Engine

Using the Datastore

Storing data in a scalable web application can be tricky. A user could be interacting with any of dozens of web servers at a given time, and the user’s next request could go to a different web server than the previous request. All web servers need to be interacting with data that is also spread out across dozens of machines, possibly in different locations around the world.

With Google App Engine, you don’t have to worry about any of that. App Engine’s infrastructure takes care of all of the distribution, replication, and load balancing of data behind a simple API—and you get a powerful query engine and transactions as well.

App Engine’s data repository, the High Replication Datastore (HRD), uses the Paxos algorithm to replicate data across multiple datacenters. Data is written to the Datastore in objects known as entities. Each entity has a key that uniquely identifies it. An entity can optionally designate another entity as its parent; the first entity is a child of the parent entity. The entities in the Datastore thus form a hierarchically-structured space similar to the directory structure of a file system. An entity’s parent, parent’s parent, and so on recursively, are its ancestors; its children, children’s children, and so on, are its descendants. An entity without a parent is a root entity.

The Datastore is extremely resilient in the face of catastrophic failure, but its consistency guarantees may differ from what you’re familiar with. Entities descended from a common ancestor are said to belong to the same entity group; the common ancestor’s key is the group’s parent key, which serves to identify the entire group. Queries over a single entity group, called ancestor queries, refer to the parent key instead of a specific entity’s key. Entity groups are a unit of both consistency and transactionality: whereas queries over multiple entity groups may return stale, eventually consistent results, those limited to a single entity group always return up-to-date, strongly consistent results.

The sample application in this guide organizes related entities into entity groups, and uses ancestor queries on those entity groups to return strongly consistent results. In the example code comments, we highlight some ways this approach might affect the design of your application. For more detailed information, see Structuring Data for Strong Consistency.

A Complete Example Using the Datastore

Here is a new version of guestbook/guestbook.py that creates a page footer that stores greetings in the Datastore. The rest of this page discusses excerpts from this larger example, organized under the topics of storing the greetings and retrieving them.
Using Templates

HTML embedded in code is messy and difficult to maintain. It's better to use a templating system, where the HTML is kept in a separate file with special syntax to indicate where the data from the application appears. There are many templating systems for Python: EZT, Cheetah, ClearSilver, Quixote, Django, and Jinja2 are just a few. You can use your template engine of choice by bundling it with your application code.

For your convenience, App Engine includes the Django and Jinja2 templating engines.

Using Jinja2 Templates

First modify the libraries section at the bottom of guestbook/app.yaml:

```
app.yaml

libraries:
  - name: webapp2
    version: latest
  - name: jinja2
    version: latest
```

This configuration makes the newest supported version of Jinja2 available to your application. To avoid possible compatibility issues, serious applications should use an actual version number rather than latest.

Now modify the statements at the top of guestbook/guestbook.py:

```
guestbook.py

import os
```
Using Static Files

Unlike a traditional web hosting environment, Google App Engine does not serve files directly out of your application's source directory unless configured to do so. We named our template file `index.html`, but this does not automatically make the file available at the URL `/index.html`.

But there are many cases where you want to serve static files directly to the web browser. Images, CSS stylesheets, JavaScript code, movies and Flash animations are all typically stored with a web application and served directly to the browser. App Engine can serve specific files directly without you having to code your own handler.

Using Static Files

Edit `guestbook/app.yaml` and replace its contents with the following:

```
app.yaml

version: 1
runtime: python27
api_version: 1
threadsafe: true

# Handlers match in order, put above the default handler.
handlers:
- url: /stylesheets
  static_dir: stylesheets
- url: /.*
  script: guestbook.app
```

Source: [https://cloud.google.com/appengine/docs/python/gettingstartedpython27/staticfiles](https://cloud.google.com/appengine/docs/python/gettingstartedpython27/staticfiles)
Uploading Your Application

To upload your application:

1. Sign in to App Engine using your Google account. If you do not have a Google account, you can create a Google account with an email address and password.

2. If you haven't already done so, create a project for your App Engine app as follows:
   a. Visit the Google Cloud Platform Console and click Create Project.
   b. Supply the desired project name in the New Project form. It doesn't have to match your app name, but using the same name as your app might make administration easier.
   c. Accept the generated project ID or supply your own ID. This project ID is used as the App Engine application ID. Note that this ID can only be used once: if you subsequently delete your project, you won't be able to re-use the ID in a new project.

   Note: You can specify that your new application should reside in the European Union, rather than the United States. Hosting applications in the European Union is especially useful if your application's users are closer to Europe than to the United States. There is less network latency and the End User Content will be stored at rest in the European Union. You must specify this location when you register the application; you cannot change it later. Click Show Advanced Options in the Create Project section to select a location option, either United States or European Union.

3. Note the application ID (project ID) you created above.
4. Upload your finished application to Google App Engine by invoking the following command. This opens a browser window for you to sign in using your Google account. You'll be providing the project ID as the argument for `-A`.

   appcfg.py -A <YOUR_PROJECT_ID> update guestbook/

5. The Datastore Indexes may take some time to generate before your application is available. You will receive a NeedIndexError when accessing your app if the indexes are still in the process of being generated. This is a transient error for the example, so try a little later if at first you receive this exception.

6. Your app is now deployed and ready for users!
Pricing overview
Run your application using the same technology and tools used at Google. Cloud Platform provides the building blocks so you can develop quickly, using the services that you need.

Pricing calculator
Want to get a sense of what running in the cloud will cost? The pricing calculator provides a quick and easy way to estimate what your usage will look like.
Get a custom quote

Pricing philosophy
We believe that pricing should be as simple as possible and the burden of getting the best deal should be on us not you. That’s why we have sustained-use discounts, pay by the minute pricing, and believe in Moore’s law in the cloud.
See philosophy

Billing questions
Sometimes you just need to get questions about your bill answered. Take a look at our frequently asked questions and if you don’t get an answer contact us.
See billing faqs
Google Cloud Platform

With Google Cloud Platform, developers can build, test and deploy applications on Google's highly-scalable and reliable infrastructure for your web, mobile and backend solutions.

Focus on writing code, not on infrastructure, and use the same infrastructure that Google uses for your application, computing and big data needs.

Get started quickly

If you're ready to get started, try these solutions for getting software up and running on our platform!

- [LAMP development stack](#)
- [Ruby development stack](#)
Try Google Cloud Platform for free

Build on top of the infrastructure that powers Google.
Sign up for free and get $300 to spend over 60 days on all Google Cloud Platform services.

Certain terms and conditions apply. Learn more

Country: Taiwan

Account type
- Business
- Individual

Name and address
- Business name
- Name
- Street address

Get $300 to kick start your app.
Sign up for free and get $300 to spend on Google Cloud Platform over the next 60 days.

Why do you need my billing information?
We use your billing information to verify that you're a real person. Don't worry, you will not be billed for the free trial.

Do I have to pay when my free trial ends?
No. You're under no obligation to buy anything when the free trial ends. If you want to continue to use Google Cloud Platform, just upgrade before your trial runs out.

Questions?
Check out the FAQ or leave us a message.
Try Google Cloud Platform for free
Get started quickly

If you're ready to get started, try these solutions for getting software up and running on our platform!

- **LAMP development stack**
  
  LAMP (an acronym for Linux, Apache, MySQL, and PHP) is the archetypal open-source web development stack for many developers, and it runs great on Compute Engine!

- **Ruby development stack**
  
  Ruby on Rails is one of the most popular frameworks for developing web applications, powering sites like Github, Basecamp and Shopify. Rapidly create new features, easily maintain code, and take advantage of the many open source contributions to Ruby on Rails, running on Compute Engine.

- **Quickstart for Wordpress**
  
  Set up a project, download a zip, change your config file, and deploy—and you’ll have a working WordPress project running on Google Cloud Platform, with App Engine as your hosting environment.

**App Engine "Hello World" starter**

Start editing a working "Hello World" app right now, in the browser. This gives you a good starting point and a feel for what it's like editing a working App Engine application.

Documentation

Use the following section or the left-hand navigation to access the various sets of documentation that cover Google Cloud Platform. Choose from computing and hosting, storage, big data, management, services, and developer tools.

Computing and hosting
Google App Engine

App Engine "Hello World" starter

https://console.developers.google.com/start/appengine

Deploy your first app in five minutes

• Start editing a working "Hello World" app right now, in the browser.
• This gives you a good starting point and a feel for what it's like editing a working App Engine application.
Try Google App Engine Now

1. NAME YOUR PROJECT
2. SELECT YOUR LANGUAGE
3. EXPLORE THE STARTER CODE
4. INSTALL GOOGLE CLOUD SDK
5. RUN YOUR APP LOCALLY
6. CREATE YOUR PROJECT AND DEPLOY
Try Google App Engine Now

Creating an App Engine app is easy, and it's free to start. Upload your app and share it with users right away, at no charge and with no commitment required.

1. NAME YOUR PROJECT

You use your project to manage all of the Google Cloud Platform resources for your app, including deployment, access control, billing, and services. You can change your project name later.

   HelloWorldGoogleAppEngine

2. SELECT YOUR LANGUAGE

Python  Java
Google App Engine

2. SELECT YOUR LANGUAGE

- Python
- Java
- PHP
- Go

3. EXPLORE THE STARTER CODE

Browse the starter code and see the app running below.

HELLO WORLD - JAVA

```java
package myapp;

import java.io.IOException;
import javax.servlet.http.*;

public class DemoServlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws IOException {
        resp.setContentType("text/plain");
    }
}
```
Google App Engine

3 EXPLORE THE STARTER CODE

Browse the starter code and see the app running below.

HELLO WORLD - JAVA

```java
package myapp;

import java.io.IOException;
import javax.servlet.http.*;

public class DemoServlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println("\"name\": \"World\" ");
    }

Hello, World
```
Google App Engine

HELLO WORLD - JAVA

```java
package myapp;
import java.io.IOException;
import javax.servlet.http.*;

public class Demoservlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws IOException {
        resp.setContentType("text/plain");
        resp.getWriter().println("\"name\": \"World\" ");
    }
}
```

Hello, World

Download this code
Google App Engine

4 INSTALL GOOGLE CLOUD SDK

1. Download and install Google Cloud SDK by running the following command in your shell or Terminal:

```
curl https://sdk.cloud.google.com/ | bash
```

Follow the prompts to install the Java App Engine package.

2. Restart your shell or Terminal to pick up environment changes.

3. Sign in to Google Cloud Platform using this command:

```
gcloud auth login
```

4. Install the App Engine package for Java using this command:

```
gcloud components update gae-java
```

5 RUN YOUR APP LOCALLY
5   RUN YOUR APP LOCALLY

Note: App Engine supports Java 7. Make sure you have the Java 7 JDK installed.

1. Download `appengine-try-java.zip` and unpack it. This creates your project directory, including `src/` and `war/` subdirectories.

2. Download and install Apache Maven version 3.0 or later. The starter app includes an Apache Maven build file.

3. Build and run the sample locally using the following commands:

   ```
   cd appengine-try-java
   mvn package
   dev_appserver.sh target/appengine-try-java-1.0
   ```

   You can stop the server by pressing Ctrl-C in the command window.

4. Visit the locally running app in your browser: `http://localhost:8080`

6    CREATE YOUR PROJECT AND DEPLOY

Now that you've seen your app running on your local machine, you're ready to create and deploy your project, `HelloWorldGoogleAppEngine`.
5 RUN YOUR APP LOCALLY

Note: App Engine supports Java 7. Make sure you have the Java 7 JDK installed.

1. Download appengine-try-java.zip and unpack it. This creates your project directory, including src/ and war/subdirectories.

2. Download and install Apache Maven version 3.0 or later. The starter app includes an Apache Maven build file.

3. Build and run the sample locally using the following commands:

   cd appengine-try-java
   mvn package
   dev_appserver.sh target/appengine-try-java-1.0

   You can stop the server by pressing Ctrl-C in the command window.

4. Visit the locally running app in your browser: http://localhost:8080

6 CREATE YOUR PROJECT AND DEPLOY

Now that you've seen your app running on your local machine, you're ready to create and deploy your project, HelloWorldGoogleAppEngine.
Google Developers Console

mvn package
dev_appserver.sh target/appengine-try-java-1.0

You can stop the server by pressing Ctrl-C in the command window.

4. Visit the locally running app in your browser: http://localhost:8080

---

6. CREATE YOUR PROJECT AND DEPLOY

Congratulations! Your project is ready. Your unique project ID is psychic-outcome-783.

Deploy your app from your local dev environment using this command:

appcfg.sh -A psychic-outcome-783 update target/appengine-try-java-1.0

After deploying your app, you can visit it with your browser at this URL:

psychic-outcome-783.appspot.com

That's it! You're running on Google App Engine. Go to your project dashboard to see how your app is performing.

View your project dashboard
Google Cloud Datastore
Mobile App Backend Services

- iOS
- Android
- Web

Services
- Cloud Endpoints
- App Engine

Compute

Storage
- Cloud Datastore

Source: [https://cloud.google.com/solutions/mobile/mobile-app-backend-services](https://cloud.google.com/solutions/mobile/mobile-app-backend-services)
Datastore Internals

• Based on Bigtable
  – high scalability
  – High availability
  • synchronous writes on multiple datacenters

Source: Datastore Introduction, [http://www.youtube.com/watch?v=fQazhzcC-rg](http://www.youtube.com/watch?v=fQazhzcC-rg)
What is Bigtable?

• Scalable, distributed, highly-available and structured storage
  – Bigtable is not database by itself (it doesn’t support query)

• Google usage
  – In production since April 2005
  – Web Search, YouTube, Earth, Analytics

Source: Datastore Introduction, http://www.youtube.com/watch?v=fQazhzC-rg
Bigtable Data Model

• A row has a Key and Columns
• Sorted by Key
  – In lexical order
  – Enables range query by application
## Google Datastore Basic Operation

Different terms for corresponding concepts

<table>
<thead>
<tr>
<th>Category of object</th>
<th>Google Datastore</th>
<th>Relational Database Management System (RDBMS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category of object</td>
<td>Kind</td>
<td>Table</td>
</tr>
<tr>
<td>One entry/object</td>
<td>Entity</td>
<td>Row</td>
</tr>
<tr>
<td>Unique identifier of data entry</td>
<td>Key</td>
<td>Primary Key (PK)</td>
</tr>
<tr>
<td>Individual data</td>
<td>Property</td>
<td>Field</td>
</tr>
</tbody>
</table>

Source: Datastore Introduction, [http://www.youtube.com/watch?v=fQazhzC-rg](http://www.youtube.com/watch?v=fQazhzC-rg)
Kind, Entity and Key

**BlogEntry**
- Key: 1234
- name: joe@ex.com
- message: xxxxx
- data: 1/1/2012 12:32

**User**
- Key: joe@ex.com
- email: joe@ex.com
- followees: [usr2@ex.com, usr3@ex.com]
- followers: []

- Key: usr2@ex.com
- email: usr2@ex.com
- followees: []
- followers: [joe@ex.com]

Source: Datastore Introduction, [http://www.youtube.com/watch?v=fQazhzcC-rg](http://www.youtube.com/watch?v=fQazhzcC-rg)
Properties and Data Types

Each entity has one or more **named properties**

- **Variety of datatypes** (int, float, boolean, String, Date, ...)
- Can be multi-valued

---

**BlogEntry**

- Key: 1234
- name: joe@ex.com
- message: xxxxx
- data: 1/1/2012 12:32

**User**

- Key: joe@ex.com
- email: joe@ex.com
- followees: 
  
  [usr2@ex.com, usr3@ex.com]
- followers: 
  
  []

- Key: usr2@ex.com
- email: usr2@ex.com
- followees: 
  
  []
- followers: 
  
  [joe@ex.com]
Creating an Entity with Java Low-level API

DatastoreService datastore = DatastoreServiceFactory.getDatastoreService();

Entity employee = new Entity(“Employee”);
employee.setProperty(“name”, “Antonio Saliery”);
employee.setProperty(“hireDate”, new Date());
employee.setProperty(“attendedHrTraining”, true);
datastore.put(employee);

Source: Datastore Introduction, http://www.youtube.com/watch?v=fQazhzcC-rg
Mobile App Backend Services

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Firebase: Backend as a Service (BaaS)

Source: https://firebase.google.com/
Mobile App Backend Services

Source: https://cloud.google.com/solutions/mobile/mobile-app-backend-services
Mobile App Backend Services

Source: [https://cloud.google.com/solutions/mobile/mobile-app-backend-services](https://cloud.google.com/solutions/mobile/mobile-app-backend-services)
App success made simple

The tools and infrastructure you need to build better apps and grow successful businesses

Move fast

Firebase is a mobile platform that helps you quickly develop high-quality apps, grow your user base, and earn more money. Firebase is made up of complementary features that you can mix-and-match to fit your needs.

ALL FEATURES

Source: https://firebase.google.com/
Firebase

Develop
- Realtime Database
- Authentication
- Cloud Messaging
- Storage
- Hosting
- Test Lab
- Crash Reporting

Grow
- Notifications
- Remote Config
- App Indexing
- Dynamic Links
- Invites
- AdWords

Earn
- AdMob

Source: https://firebase.google.com/
Work across platforms

Deliver cross-platform apps with APIs packaged into single SDKs for iOS, Android, JavaScript and C++. Expand to a different platform without modifying your infrastructure.

- Try iOS tutorial
- Try Android tutorial
- Try Web tutorial

Free to start, scale with ease

Most Firebase features are free forever, for any scale. Our four paid features have a generous free tier and two paid plans once you begin to grow.

Source: https://firebase.google.com/
Firebase is a mobile platform that helps you quickly develop high-quality apps, grow your user base, and earn more money. Firebase is made up of complementary features that you can mix-and-match to fit your needs.

Source: https://firebase.google.com/
Welcome to Firebase

Tools from Google for developing great apps, engaging with your users, and earning more through mobile ads. Learn more

CREATE NEW PROJECT
or import a Google project

Your projects using Firebase

Your next project
Your next big hit
Your next weekend hack

https://console.firebase.google.com/?pli=1
Welcome to Firebase

Tools from Google for developing great apps, engaging with your users, and earning revenue. Learn more

CREATE NEW PROJECT
or import a Google project

Create a project

Project name
My awesome project

Country/region
United States

By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at any time. Learn more

By proceeding and clicking the button below, you agree that you are using Firebase services in your app and agree to the applicable terms.

CANCEL CREATE PROJECT
Welcome to Firebase

Tools from Google for developing great apps, engaging with your users, and earning mobile ads. Learn more

Create a project

Project name

United States

Projects span apps across platforms.

A project is a container for your apps across platforms: Android, iOS, and web. It supports sharing features such as Database, User Management, and Remote Config across your apps.

By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at anytime. Learn more

By proceeding and clicking the button below, you agree that you are using Firebase services in your app and agree to the applicable terms.
Welcome to Firebase

Tools from Google for developing great mobile apps.

Create a project

- Project name
  - HelloWorldFirebase

- Country/region
  - United States

By default, your Firebase Analytics data will enhance other Firebase features and Google products. You can control how your Firebase Analytics data is shared in your settings at anytime.

By proceeding and clicking the button below, you agree that you are using Firebase services in your app and agree to the applicable terms.
Add Firebase to Your Web App

Welcome to Firebase! Get started here.

Add Firebase to your iOS app
Add Firebase to your Android app
Add Firebase to your web app
Add Firebase to Your Web App

Copy and paste the snippet below at the bottom of your HTML, before other `script` tags.

```html
<script src="https://www.gstatic.com/firebasejs/3.6.4/firebase.js"></script>

// Initialize Firebase
var config = {
  apiKey: "AIzaSyDy4OdvREiNjYJP2_YBvgZLP41HUEB_g",
  authDomain: "helloworldfirebase-d6ef2.firebaseapp.com",
  databaseURL: "https://helloworldfirebase-d6ef2.firebaseapp.com",
  StorageBucket: "helloworldfirebase-d6ef2.appspot.com",
  messagingSenderId: "77894601289"
};

firebase.initializeApp(config);
</script>

Check these resources to learn more about Firebase for web apps:

- Get Started with Firebase for Web Apps
- Firebase Web SDK API Reference
- Firebase Web Samples

Add Firebase to Your Web App

Analytics
Get detailed analytics to measure and

Authentication
Authenticate and manage users from a

Database
Store and sync data in realtime across all
Add Firebase to Your Web App

```html
<script src="https://www.gstatic.com/firebasejs/3.6.4/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "AIzaSyDu4JodvREiNjYJP2_YBVgZLpNu1HUEB_g",
    authDomain: "helloworldfirebase-d6ef2.firebaseapp.com",
    databaseURL: "https://helloworldfirebase-d6ef2.firebaseio.com",
    storageBucket: "helloworldfirebase-d6ef2.appspot.com",
    messagingSenderId: "778984601289"
  };
  firebase.initializeApp(config);
</script>
```
Add Firebase to Your Web App

```html
<script src="https://www.gstatic.com/firebasejs/3.6.1/firebase.js"></script>

// Initialize Firebase
// TODO: Replace with your project's customized code snippet
var config = {
    apiKey: "<API_KEY>",
    authDomain: "<PROJECT_ID>.firebaseapp.com",
    databaseURL: "https://<DATABASE_NAME>.firebaseio.com",
    storageBucket: "<BUCKET>.appspot.com",
    messagingSenderId: "<SENDER_ID>",
};
firebase.initializeApp(config);
</script>

**firebase-app** - The core firebase client (required).
**firebase-auth** - Firebase Authentication (optional).
**firebase-database** - The Firebase Realtime Database (optional).
Add Firebase to Your Web App

<!-- Leave out Storage -->
<!-- <script src="https://www.gstatic.com/firebasejs/3.6.1/firebase-storage.js"></script> -->

<script>
  var config = {
    // ...
  };
  firebase.initializeApp(config);
</script>
Add Firebase to Your Web App

```javascript
var firebase = require("firebase/app");
require("firebase/auth");
require("firebase/database");

// Leave out Storage
//require("firebase/storage");

var config = {
  // ...
};
firebase.initializeApp(config);
```
Add Firebase to Your Web App

- `firebase.auth()` - Authentication
- `firebase.storage()` - Storage
- `firebase.database()` - Realtime Database
/** Set the configuration for your app */
// **TODO**: Replace with your project's config object

```javascript
var config = {
  apiKey: "apiKey",
  authDomain: "projectId.firebaseapp.com",
  databaseURL: "https://databaseName.firebaseio.com",
  storageBucket: "bucket.appspot.com"
};

firebase.initializeApp(config);

// **Get a reference to the database service**
var database = firebase.database();
```
```javascript
{
    // Chats contains only meta info about each conversation
    // stored under the chats's unique ID
    "chats": {
        "one": {
            "title": "Historical Tech Pioneers",
            "lastMessage": "ghopper: Relay malfunction found. Cause: moth.",
            "timestamp": 1459361875666
        },
        "two": { ... },
        "three": { ... }
    },

    // Conversation members are easily accessible
    // and stored by chat conversation ID
    "members": {
        // we'll talk about indices like this below
        "one": {
            "ghopper": true,
            "alovelace": true,
            "eclarke": true
        },
        "two": { ... },
        "three": { ... }
    }
}
```
Firebase Database Structure Data

// Messages are separate from data we may want to iterate quickly
// but still easily paginated and queried, and organized by chat
// conversation ID
"messages": {
  "one": {
    "m1": {
      "name": "eclarke",
      "message": "The relay seems to be malfunctioning.",
      "timestamp": 1459361875337
    },
    "m2": { ... },
    "m3": { ... }
  },
  "two": { ... },
  "three": { ... }
}
}
Firebase Database Write Data

```javascript
// Get a reference to the database service
var database = firebase.database();

function writeUserData(userId, name, email, imageUrl) {
  firebase.database().ref('users/' + userId).set(
    {
      username: name,
      email: email,
      profile_picture: imageUrl
    });
}
```
Firebase Database Read Data

```javascript
// Get a reference to the database service
var database = firebase.database();

var userId = firebase.auth().currentUser.uid;
return firebase.database().ref('/users/' + userId).once('value').then(function(snapshot) {
  var username = snapshot.val().username;
  // ...
});
```
Firebase Database
Firebase Database Rules

Default security rules require users to be authenticated

```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null"
  }
}
```
Firebase Database

Realtime Database

DATA    RULES    USAGE    BACKUPS

👉 https://helloworldfirebase-d6ef2.firebaseapp.com/

⭐ Default security rules require users to be authenticated

helloworldfirebase-d6ef2: null +

Store and sync data in realtime across all connected clients

Learn more
Firebase Database

Realtime Database

DATA          RULES          USAGE          BACKUPS

https://helloworldfirebase-d6ef2.firebaseapp.com/

Default security rules require users to be authenticated

<table>
<thead>
<tr>
<th>Name</th>
<th>score</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>

email: "imyday@gmail.com"
username: "imyday"
Summary

• Google App Engine
  – Google Cloud Platform
  – Google Cloud Datastore
  – Google Firebase
References

• Google Cloud Platform, https://cloud.google.com/

• Google App Engine, https://cloud.google.com/appengine/

• Google Cloud Datastore, https://cloud.google.com/datastore/

• Google Cloud Endpoints, https://cloud.google.com/endpoints/

• Google Firebase https://firebase.google.com/