Google App Engine (GAE)

Google App Engine (GAE)

- is a **Platform as a Service (PaaS)** cloud computing solution provided by Google.
- It allows developers to **build, deploy, and manage applications** without worrying about the underlying infrastructure, such as servers and databases.

Example
Imagine you create a web app for online voting.

- With **GAE**, you upload the app code, and Google automatically hosts it, ensuring it works smoothly, even if thousands of users vote at the same time.
- You **don’t** need to manage servers, install software, or worry about scaling—Google does it for you!

- **Amazon's cloud (IaaS - Infrastructure as a Service)** lets users set up and manage their own virtual machines.
- They have to install software, configure settings, and maintain everything themselves.
- **Google App Engine (PaaS - Platform as a Service),** on the other hand, hides all the technical setup.
- It gives developers a ready-to-use platform where they just upload their app, and Google handles everything else.
- PaaS automatically adjusts resources based on traffic. If more people use the app, it scales up; if usage is low, it scales down—saving costs and improving efficiency.

Example:

Suppose you create an online **food ordering app** using Google App Engine.

- You upload your code, and Google **automatically hosts** it.
- If **100 users** visit, GAE uses minimal resources.
- If **1 million users** visit, GAE automatically scales up to handle the traffic **without crashing**.
- Thus, **GAE helps developers focus on building apps** rather than managing infrastructure!

How GAE is Different from Amazon EC2?

- In **Amazon EC2 (IaaS)**, users must manually set up and manage virtual machines.
- In **GAE (PaaS)**, users just upload their code, and Google takes care of running it efficiently.

Key Features of GAE:

1. **Fully Managed Platform** – No need to manage servers, as Google handles everything.
2. **Automatic Scaling** – GAE adjusts resources based on user demand, ensuring smooth performance.
3. **Supports Multiple Programming Languages** – It supports **Python, Java, Go, PHP, and Node.js**.
4. **Built-in Security & Monitoring** – Google provides security updates and real-time monitoring tools.
5. **Easy Deployment** – Developers upload their code, and GAE automatically makes it available online.

Figure depicts a user view of Google App Engine.

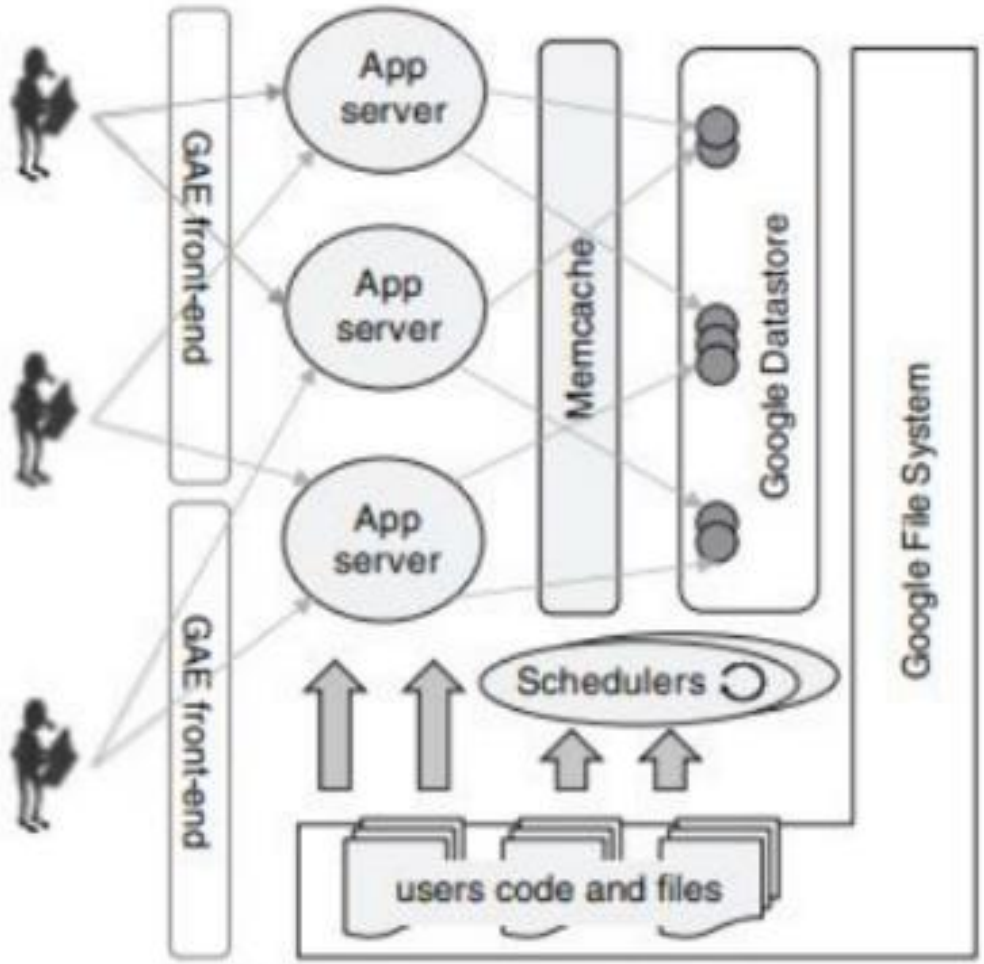


FIGURE 5.2. Google App Engine

Users Upload Code

- Developers upload their application code in Java or Python, along with necessary files.
- These files are stored in Google File System (GFS), which is fault-tolerant and redundant (meaning it ensures data is safe even if a server fails).

GAE Frontend & App Servers

When a user accesses an application, the GAE frontend handles the request and routes it to an appropriate application server to process it.

Caching & Databases

- To make the app faster, Memcache temporarily stores frequently accessed data.
- For permanent data storage, Google Datastore is used.

Schedulers & Processing

- If the app requires background tasks (e.g., sending emails or processing data), schedulers manage these tasks efficiently.

Instant Availability & Metered Usage

- As soon as an application is uploaded, it becomes live on the internet—no need to set up virtual machines.
- Billing is based on actual usage, meaning charges apply only for the web requests served and the computing power used, making it cost-effective.

1. Pay-as-You-Go Model

1. In **Platform as a Service (PaaS)** like **Google App Engine (GAE)**, applications can be available **24/7** worldwide.
2. However, **you are charged only when users access the application** or when background tasks (batch jobs) run.

2. Cost Difference Between PaaS & IaaS

1. In **Infrastructure as a Service (IaaS)** (like Amazon EC2), you **pay for running servers all the time**, even if no one is using your application.
2. In **PaaS (GAE)**, **if no one is using your app, you don't pay for computing resources**—only storage costs apply.

3. Free Deployment

1. **Google App Engine allows free deployment within usage limits** (e.g., limited CPU, bandwidth, and storage).
2. This is possible because **GAE does not use dedicated virtual machines (VMs) for each application**.
3. If no one accesses your app, it **only takes up storage space and does not use computing power** (so no extra cost).

- 🔷 **Key Benefit:** PaaS is more **cost-effective** for apps with fluctuating traffic, as it **scales automatically** and charges based on actual usage rather than fixed server costs.

Global Distribution

- GAE applications run on many web servers in Google's data centers worldwide.
- When a user makes a request, the nearest available web server handles it for faster response times.

Code Execution

- The web servers load application code from the Google File System (GFS) into memory to process requests.
- This ensures that applications scale automatically as demand increases.

Handling Requests

- Any web server can serve any request, meaning the same user's requests may go to different servers each time.

Memcache for Session Data

- Because requests are handled by different servers, storing temporary data (like login sessions) in-memory is difficult.
- Memcache, a distributed caching system, helps store session data across servers.
- This ensures that even if a user's requests go to different servers, their session data is still available most of the time.

Google Datastore

1. Non-Relational Database 

1. Google Datastore is a **NoSQL database**, similar to Amazon SimpleDB.
2. Unlike traditional databases (like MySQL), it doesn't use **tables and rows** but instead stores **structured data** in a flexible format.

2. Data Structure 

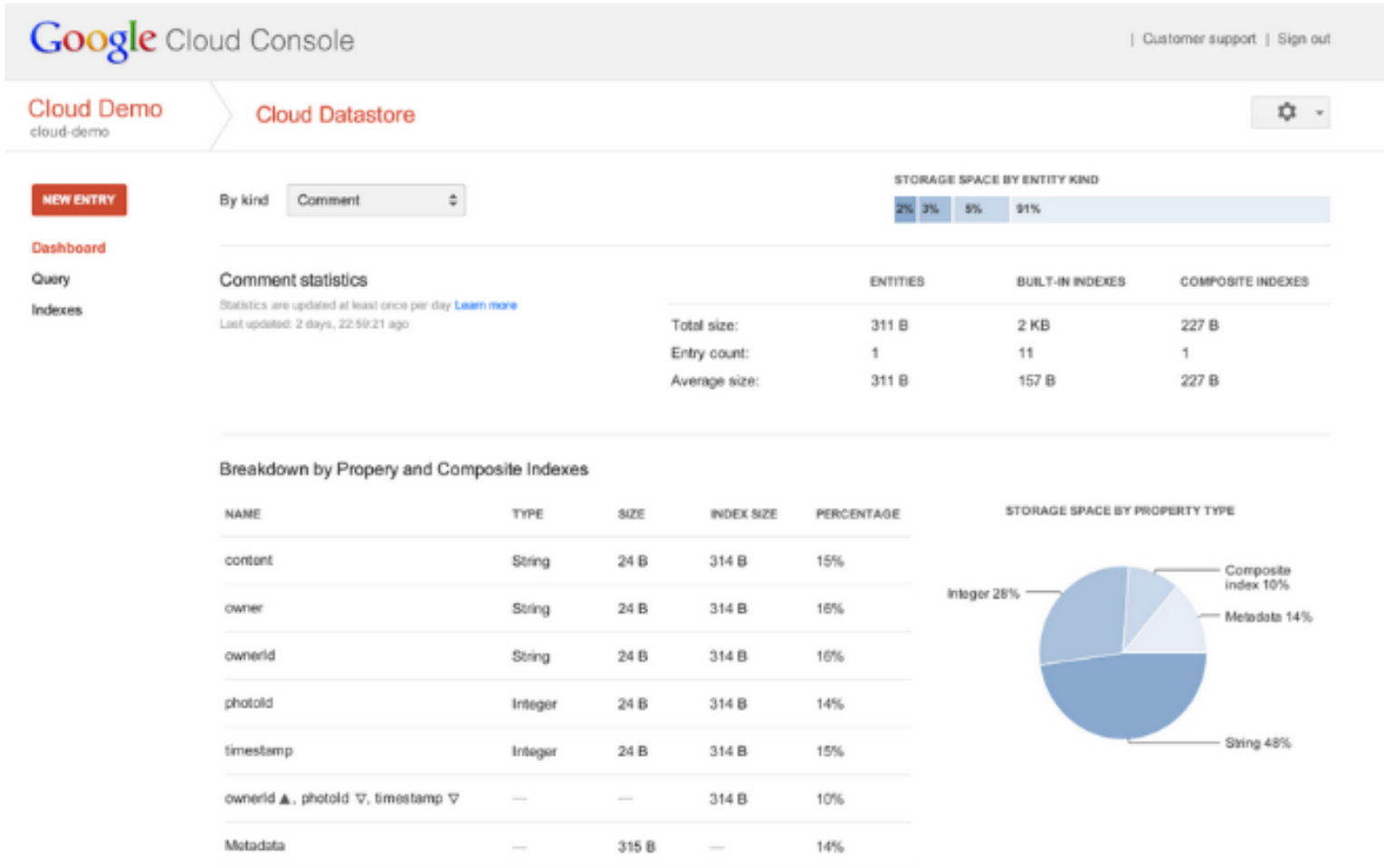
1. Data is stored in **'kinds'** (similar to tables) and **'entities'** (similar to records).
2. Unlike relational databases where all rows must follow a fixed format, Datastore **allows different entities to have different properties**.
3. Example:
    1. A "Products" database can have different properties for shoes (size, color) and books (author, genre) **without needing extra tables**.

Flexible Queries

- Queries use Google Query Language (GQL), similar to SQL.
- However, GQL does not support JOINs, meaning complex relationships must be handled differently.

Efficient Data Retrieval

- To ensure fast access and atomic updates (so that data changes remain consistent), related data is stored close together in the system.
- Even though multiple users share the Datastore, Google ensures data is secure and optimized for performance.



More details on creation : https://cloudplatform.googleblog.com/2013/05/get-started-with-google-cloud-datastore-nosql-database.html

Google Datastore

- Schema-less Database
- does not require a fixed structure like traditional databases.
- can easily modify your data structure as your application evolves without worrying about predefined tables and columns.
- Powerful Query Engine
- Allows searching across multiple properties and sorting results efficiently.
- Unlike SQL databases, it does not support complex joins, but it is optimized for scalability.
- Automatic Scaling & High Availability
- Google Datastore automatically scales to handle high loads.
- It manages data replication, ensuring high availability and durability even under heavy traffic.
- Difference Between Datastore & Database
- A datastore is any system that stores data persistently, including databases, files, and emails.
- A database is a structured collection of data managed by a Database Management System.

Feature	Amazon SimpleDB	Google Datastore
Type	Non-relational, schema-less database	Non-relational, schema-less database
Terminology	Domains → Kinds Items → Entities	Kinds → Table-like types Entities → Records with properties
Schema Flexibility	Schema-less: Items can have different sets of attributes	Schema-less: Entities can have different properties
Support for Joins	❌ Not supported	❌ Not supported
Object Relationships	❌ Not supported	✅ Supported (object-oriented style relationships)
Transactions	❌ Not supported	✅ Supported (within entity groups)
Query Language	Simple Select-like language	GQL (Google Query Language, similar to SQL)
Storage System	Data is automatically replicated for durability	Data is stored on Google File System (GFS), replicated
Scalability & Availability	✅ Automatically scalable and available	✅ Automatically scalable and available
Ideal For	Basic key-value use cases with moderate complexity	More complex app models needing structured queries and relations