

University of Bamberg



DSG-DSAM-M Lab

Session 8: Google Cloud Platform (GCP): Google App Engine (GAE) & Google Cloud SQL

Based on: <https://github.com/lion5/workshop-spring-boot>

Preparation

- Set up GCP Access:
 - Create a google account or use an existing one to log in to: <https://cloud.google.com>
 - Activate the Free Trial (see <https://cloud.google.com/free>)
- Set up gcloud CLI
 - Install it: <https://cloud.google.com/sdk/docs/install>
 - Set up credentials: <https://cloud.google.com/sdk/docs/authorizing> (with user account)

Google App Engine (GAE)



- PaaS offering (two environments available: <https://cloud.google.com/appengine/docs/the-appengine-environments>)
 - **Standard environment**
Runtimes for popular languages available: Java, Go, Python, Node.js, ...
Read and write access to the `/tmp` directory only.
 - Flexible environment
Custom Container Images, more configuration options
Ephemeral Container File System
- Infrastructure (Machine, Operating System, Networking) completely managed
- Integration with other Google services possible (Storage, Functions, ...)
- **Service:** Application that you want to run (First service is *default* service)
- Each **Service** can have multiple **Versions**
- Deployment:
 - Via Web Console
 - Via gcloud
 - Via Gradle (<https://cloud.google.com/appengine/docs/standard/java-gen2/using-gradle>)

Task 1: Deploy your application to GAE

0) Use /solutions/task3 as a template

1) Add the app engine gradle plugin (<https://github.com/GoogleCloudPlatform/appengine-plugins/tree/main/app-gradle-plugin>)

Include the `settings.gradle` file from the vc course at the root of your project and add the plugin:
`id 'com.google.cloud.tools.appengine' version '2.8.0'`

2) Configure the app engine plugin in your `build.gradle`:

```
appengine {  
    stage {  
        artifact = bootJar.archiveFile.get()  
    }  
    deploy {  
        version = "1"  
        projectId = "GLOUD_CONFIG" // read the config from gcloud cli  
        stopPreviousVersion = true // default - stop the current version  
        promote = true // default - & make this the current version  
    }  
}
```

3) At `src/main/appengine/` add a file named `app.yaml`:

```
runtime: java21  
instance_class: F1
```

25
Minutes

Task 1: Deploy your application to GAE

4) Activate an App Engine Application at: <https://console.cloud.google.com/appengine/>

5) Try to deploy the application via Gradle task → appengineDeploy

6) If an error occurs, it might be because of permissions:

The App Engine account uses Storage Buckets and an internal Registry to store artifacts and logs, therefore it needs corresponding permissions.

Look for the “App engine default service account” and add the following roles:

- Storage Admin
- Artifact Registry Create-on-Push Writer

7) If deployment succeeds, check whether you can access the application and if there is an error, check the logs ;)

25
Minutes

Task 1 Troubleshooting

- ERROR: (gcloud.app.deploy) The current Google Cloud project [...] does not contain an App Engine application. Use `gcloud app create` to initialize an App Engine application within the project.
→ Initialize an App Engine Application first
- ERROR: (gcloud.app.deploy) Permissions error fetching application [...]. Please make sure that you have permission to view applications on the project and that [...] has the App Engine Deployer (roles/appengine.deployer) role.
→ Add “App Engine Deployer” role to your service account via IAM (<https://console.cloud.google.com/iam-admin/iam>)
- ERROR: (gcloud.app.deploy) Error Response: [13] Failed to create cloud build: com.google.net.rpc3.client.RpcClientException: <eye3 title='/ArgoAdminNoCloudAudit.CreateBuild, FAILED_PRECONDITION'/> APPLICATION_ERROR;google.devtools.cloudbuild.v1/ArgoAdminNoCloudAudit.CreateBuild;invalid bucket "..."; service account ...@appspot.gserviceaccount.com does not have access to the bucket;AppErrorCode=9;StartTimeMs=1733177199405;unknown;ResFormat=uncompressed;ServerTimeSec=4.457621974;LogBytes=256;Non-FailFast;EndUserCredsRequested;EffSecLevel=privacy_and_integrity;ReqFormat=uncompressed;ReqID=976254462d2c627f;GlobalID=0;Server=[2002:aa1:2a83:0:b0:b2:b865:43a5]:4001.
→ Add “Storage Admin” role to the App Engine Service Account
- More at: <https://cloud.google.com/appengine/docs/standard/troubleshooter/deployment>



Cloud SQL



- Managed database service
- Several configuration options
 - Type of database (Postgres, MySQL, ...)
 - Resources (CPUs, RAM)
 - Availability
- Connection:
 - Directly via any suitable client
 - Using prebuilt libraries (a gradle plugin is available for example)
- Creation and Management either via Web UI or `gcloud` cli tool


Task 2: Connect to Google Cloud SQL

1) Create a database instance via Web interface: <https://console.cloud.google.com>

2) Add dependencies:

```
implementation 'com.google.cloud:spring-cloud-gcp-dependencies:4.8.4'  
implementation 'com.google.cloud:spring-cloud-gcp-starter-sql-postgresql:4.8.4'
```

2) Configure connection in `.properties` file with a profile of your choice:

A red lightning bolt icon pointing towards the configuration lines.

```
spring.cloud.gcp.sql.database-name=  
spring.cloud.gcp.sql.instance-connection-name=  
  
spring.datasource.username=postgres  
spring.datasource.password=  
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect  
spring.h2.console.enabled=false
```

3) Run your application (locally) to check if it works

10
Minutes

Task 2: Connect to Cloud SQL

4) Allow traffic from your IP address, get your public IP for example via:

```
nslookup myip.opendns.com. resolver1.opendns.com
```

5) Use pgadmin to connect to the database and verify your changes

Install locally or preferred: Run via Docker

<https://www.pgadmin.org/download/>

optional

Good to know for Cloud SQL

- To disable Cloud SQL (in a different profile) use:
`spring.cloud.gcp.sql.enabled: false`
- (Database) credentials should not be pushed to a repository
- Google Cloud SDK uses local configuration for authentication, use:
`gcloud auth login` to log in
- Some things must be explicitly enabled in the Google Cloud Console:
 - Google Cloud SQL itself
 - Google Cloud SQL Admin API
- If errors occur, check the logs ;)

Homework



Deploy your application to App Engine
configured with a connection to a Cloud SQL instance

Handle sensitive data (database credentials) carefully

Clean up (to save cost)

- Disable App Engine
- Delete buckets
- Delete database instances

Next:

- 16.12. Q&A | Preview Assignment 2
- 13.01. Cloud Firestore
- 20.01. Google Cloud Run functions

