FH JOANNEUM

Cloud Computing
Google App Engine

"Run your Apps, Host your Sites, Store your Data"

DI Manfred Pamsl
DI(FH) Mathias Knoll, MSc

FH JOANNEUM

Overview

- Platform as a Service
  - Web applications are deployed to the Google platform
  - Integration with Google accounts through APIs
- Programming Language Support
  - Java, Python, Go, PHP
- Flexible, scalable application storage
  - Relational database, cloud storage
- Developer tools
  - Eclipse plug-in, web development tools, ...

The Sandbox

- Applications run in an environment with limited access to the operating system
  - Isolates applications in an environment that is independent of the hardware, OS and physical location of the web server
  - Access to the applications only via HTTP(S)
  - Applications may only access files uploaded by the application code itself, there is no general file system access
  - Application code is only executed in response to a web request, a queued task or scheduled task - response data must be returned within 60s
- Restrictions allow Google to distribute the requests and applications across their infrastructure

Generally Available Features

- Data storage
- Communication
- Process management
- Computation
- App configuration and management

FH JOANNEUM

Data Storage

- Schema-less object datastore
- Accessible through SQL like language GQL
- Supports Binary Large Objects (BLOBs)
- Distributed in-memory data cache (“memcache”)
- Provides programmatic access to application logs and request logs

Communications

- Persistent communication channels between web clients (via Javascript library) and the application
- Send and receive email messages using Google mail accounts
- Issue HTTP(S) request using the Google networking infrastructure
- Support for Extensible Messaging and Presence Protocol (XMPP)

Process Management & Computation

- Perform tasks outside a user request
- Configuration of regularly scheduled tasks
- “Backends” are instances of an application which have access to more computing resources
- Image conversion (photographs, graphics) API

FH JOANNEUM

App Configuration & Management

- Application identity services with “Oauth” (see: http://tools.ietf.org/html/draft-ietf-oauth-v2-22)
- Provides detection of outages and scheduled maintenance
- Application access with HTTP(S) via a custom domain instead of the default “appspot.com”
- Remote API lets you transparently access App Engine services from any application
- Namespaces API to build multi-tenant applications
- Traffic Splitting to different versions of an application based on IP address or Cookies (“A/B Testing”)
- User authentication through Google accounts or OpenID

FH JOANNEUM

The Java Runtime Environment

- Applications can be developed using common Java development tools and API standards
- Interaction with the environment using Java Servlet Standard, applications may use the Java Server Pages technology
- Runtime enviroment uses the Java 7 standard
- Many App Engine services are accessed using standard Java APIs
  - E.g. Java Data Objects (JDO), Java Persistence API (JPA), Java Mail API, java.net HTTP APIs
- Additional APIs for App Engine services not covered by Java standards

Options for Storing Data

- App Engine Datastore
  - NoSQL schemaless object data store
- Google Cloud SQL
  - Relational SQL database service based on MySQL
- Google Cloud Storage
  - Storage Service for objects and files (comparable to OpenStack Swift)

FH JOANNEUM

App Engine Data Store

- Distributed NoSQL data storage service
- Features a query engine and atomic transactions
- Data objects (“entities”) are identified by keys consisting of
  - “Kind”: category of the object
  - “Identifier”: name string or automatically assigned integer value
  - “Ancestor Path”: optional, link to parent entity, locates the entity in the datastore hierachy
- Entities may have properties assigned (e.g. integer, strings, ...)
- Queries can retrieve entities of a given kind filtered and sorted by the values of properties

FH JOANNEUM

Application Development

- App Engine SDK includes a web server that emulates all of the App Engine services, includes a tool to upload the application to the App Engine
- Java SDK runs on any Java 7 platform
- Google Eclipse Plugin is available to create, test and upload App Engine Applications
- Web-based Administration Console is used to manage the applications on the App Engine

FH JOANNEUM

Exercise Steps

- Use the provided Eclipse Environment for the App Engine application development
    - Already includes the Plugin, App Engine SDK and JDK 7
- Follow the guideline to create your first sample "guestbook" application: https://developers.google.com/appengine/docs/java/gettingstarted/introduction
    - Skip the the SDK installation step, it is already included in the Eclipse bundle
    - Start with "Creating a Project"
- As your homework, create an additional "point of interest" application as defined on the next slides

Point of Interest Application

- Purpose
  - Public accessible registration of points of interests, usable by any application
- Implementation requirements
  - Implement a Java application hosted on the Google AppEngine Cloud infrastructure
  - Provide ReST Web Services, that uses data in JSON format for point of interest registration and retrieval
    - you may use the gson library: http://code.google.com/p/google-gson/ for converting Java objects to JSON format and vice versa
  - Use the App Engine schema-less object data store for data storage

FH JOANNEUM

Point of Interest Attributes

- “id”: unique identifier, long value, automatically generated when adding a point of interest
- “name”: Name of the point of interest, “String” value
- “latitude”: latitude position value, “double” value
- “longitude”: longitude position value
- “creator”: person creating the entry, “String” value
- “description”: short description of the point of interest, “String” value
- “category”: category of the point of interest, “String” value

JSON POI Example

```
{
   "id": 12346789,
   "name": "FH-Graz",
   "latitude": 47.0693127,
   "longitude": 15.4079899,
   "creator": "john",
    "description": "FH JOANNEUM Graz",
   "category":"Fachhochschule"
 }
```

FH JOANNEUM

ReST API Description

- http://<YourGoogleAppURL>/resources/poi
  - GET: retrieve all POIs as JSON array
  - POST: Add one POI in JSON format (attribute "id" shall be ignored), returns the JSON POI with the generated "id" attribute
- http://<YourGoogleAppURL>/resources/poi?<attribute>=<value>...
  - GET: returns the POIs with matching attribute values (logical "and" relation)
- http://<YourGoogleAppURL>/resources/poi/<id>
  - GET: retrieve the POI defined by ID <id>
  - POST: Update the POI defined by ID <id>
  - DELETE: Delete the POI defines by ID <id>
- When encountering an error, an appropriate HTTP error code (4xx) and error message shall be returned

Teamwork

- Build teams of three members
- Responsibilities:
  - First member: implements the ReST API Servlet
  - Second member: implements the internal database API used by the ReST API Servlet
  - Third member: client program for basic CRUD operations (any technology you like, e.g. native Java or Python client program, Google App Engine web interface, ...)

Deliverable

- Application has to be deployed on the Google cloud
- Upload of a ZIP file containing:
    - (Eclipse) projects including the source code
    - Document describing the client program usage and access to the ReST service