Лабораторна робота № 3. Реєстрація та налаштування сервісу хмарних обчислень

Частина 1. Підготовка середовища розробки Java або Python для розробки додатків у хмарному сервісі. Консоль Адміністрування.

Мета роботи: ознайомитися на практиці з засобами реєстрації та налаштування сервісу хмарних обчислень Google Application Engine. Підготувати середовище розробки для виконання наступних лабораторних робіт. Набути практичні навички роботи з консоллю адміністрування.

Задача: вивчити механізм реєстрації сервісу хмарних обчислень, провести необхідні налаштування хмарного сервісу для виконання лабораторних робіт.

Короткі теоретичні відомості

1. Реєстрація та налаштування сервісу хмарних обчислень

Точка доступу до ресурсів проекту Google App Engine: https://cloud.google.com/appengine/

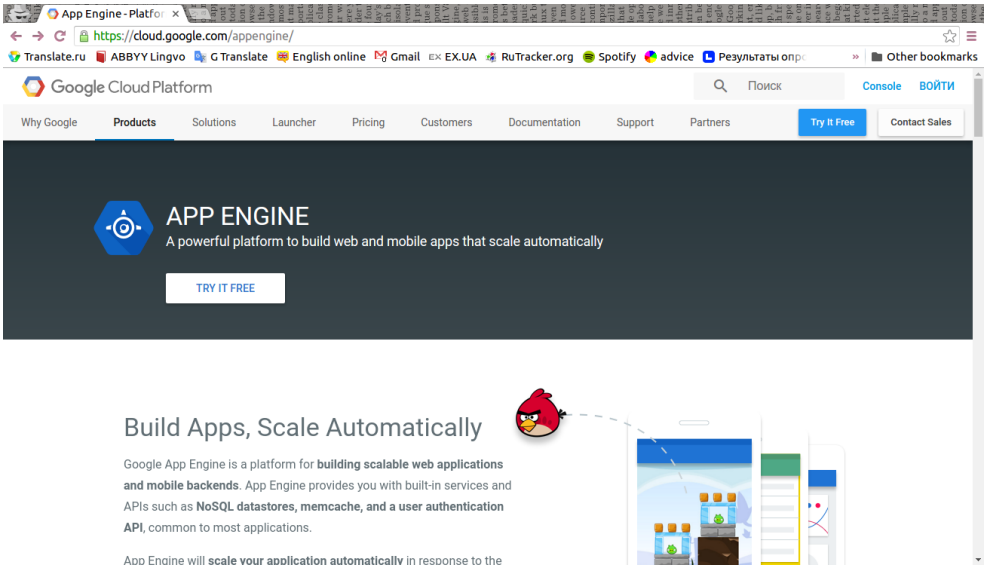

Рис. 1.1. Точка доступу до ресурсів проекту GAE

В разі необхідності можна звернутися до документації доступної за посиланням https://cloud.google.com/appengine/docs.

Доступ до ресурсу за будь-яким обліковим записом Google (в т.ч. – електронної пошти Gmail). За відсутності такого запису слід зареєструвати новий.

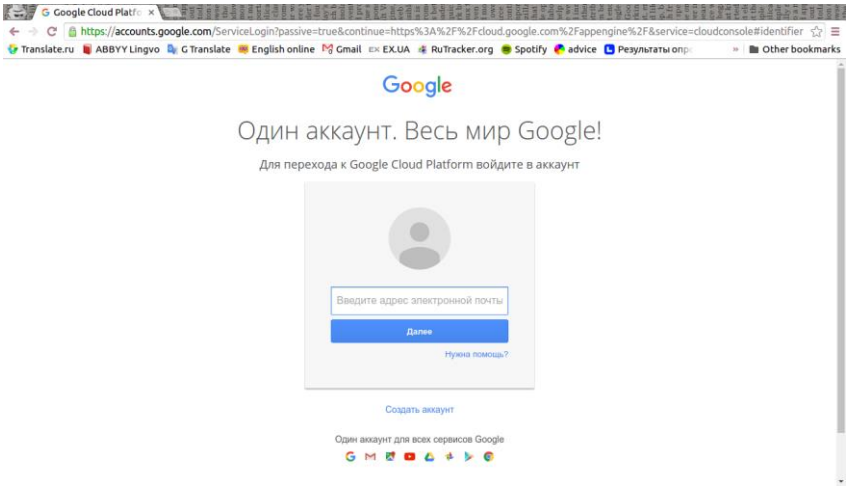

Рис. 1.2. Доступ до акаунту GAE

Після входу – повертаємось на початкову сторінку проекту Google App Engine. Обираємо “Try it Free”. Тим самим ми переходимо в меню розробника, де нам пропонується ввести данні кредитної картки. Це не є необхідною умовою користування сервісом, цей етап можна пропустити. Для створення нового проекту можна вибрати “Відкрити проект”. Або перейти в бокове меню, там обрати “Головну сторінку”, або обрати ресурс “App Engine”.

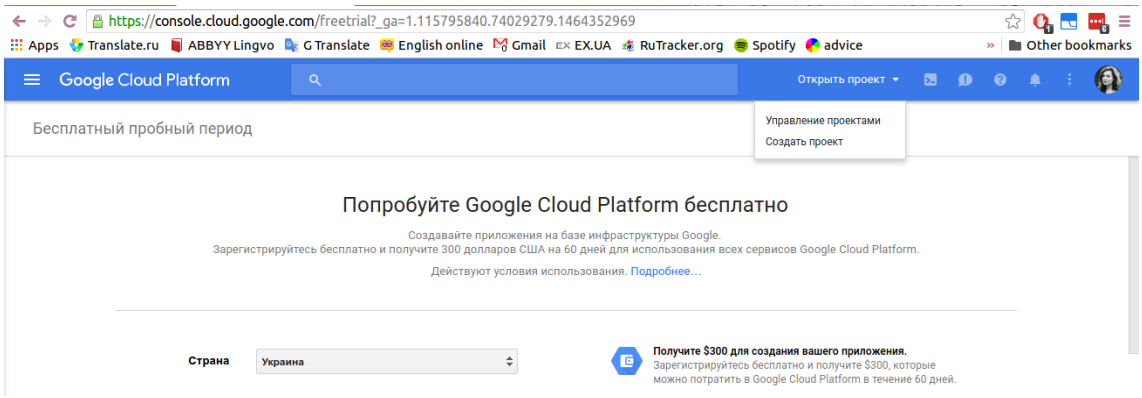

Рис. 1.3. Консоль Google Cloud Platform

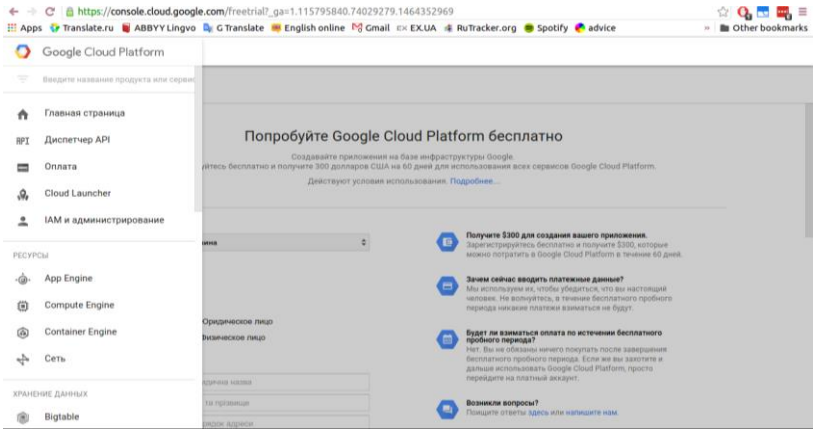

Рис. 1.4. Консоль адміністрування

Відкривається сторінка створення проекту.

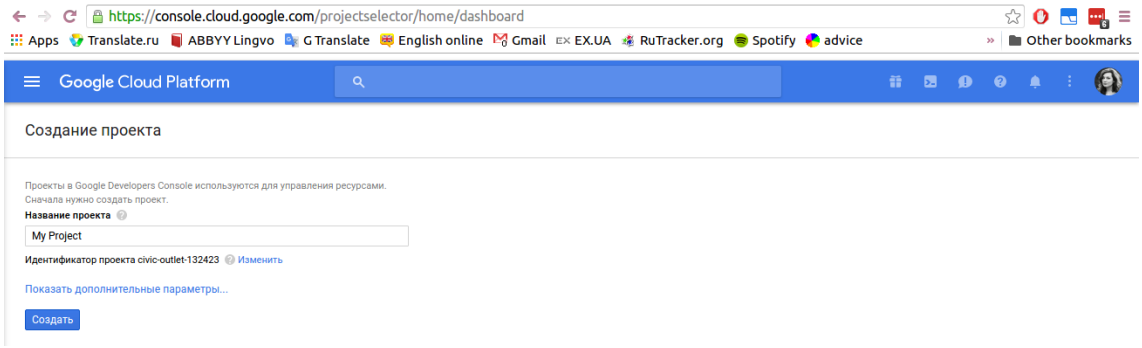

Рис. 1.5. Створення проекту

Кожен додаток (з 15 можливих на 1 обліковий запис) має глобальний ідентифікатор. За умовчуванням вашому проекту надається стандартний ідентифікатор, при створенні проекту його можна редагувати. Ідентифікатор має бути унікальним, якщо це не так, система не дасть можливості створити проект. Перевірити унікальність обраного ідентифікатора можна за допомогою відповідної кнопки поруч. Для продовження слід також вказати назву проекту. В додаткових параметрах можна вибрати локалізацію серверів.

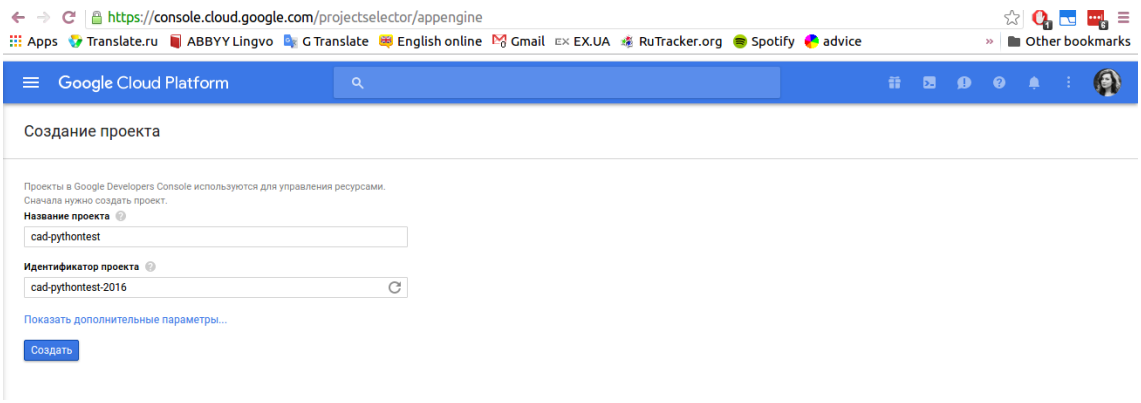

Рис. 1.6. Параметри створення проекту

Після вказаних дій буде створено (але не розгорнуто) перший хмарний додаток. Далі необхідно активізувати App Engine. Використовуючи бокове меню переходимо на ресурс App Engine.

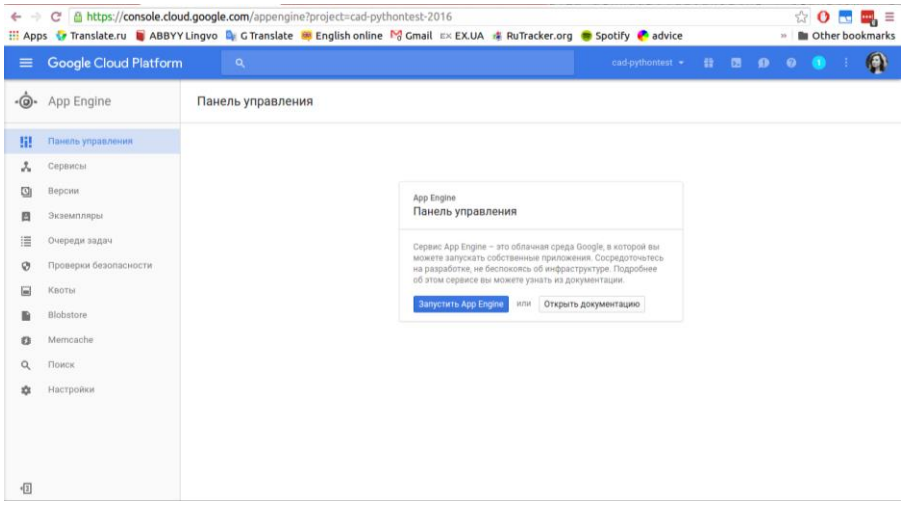

Рис. 1.7. Активація Google App Engine

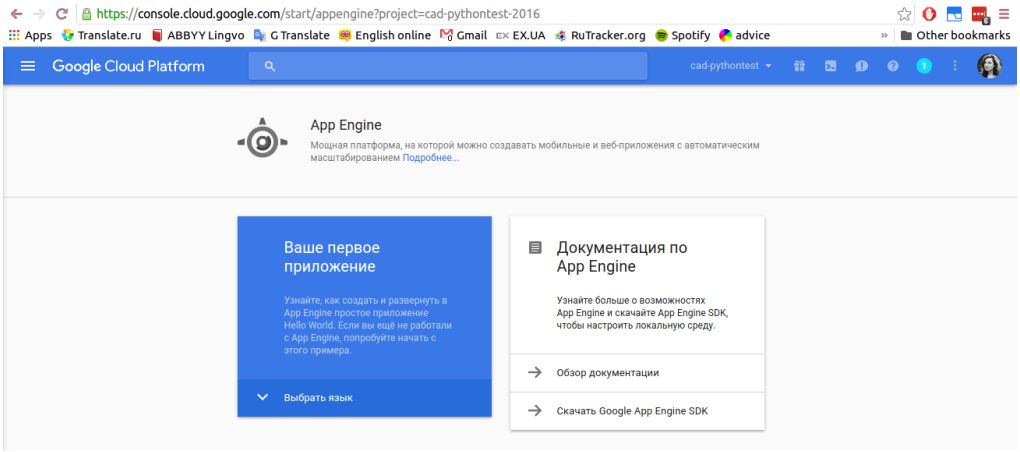

Рис. 1.8. Створення додатку в Google App Engine

Таким чином можна, слідуючи інструкціям початкового туторіалу, навчитись основам створення і користування Google App Engine.

2. Підготовка середовища розробки Java

2.1. Підготовка середовища розробки Java для розробки додатків у хмарному сервісі.

Вимоги до системи:

Необхідна умова: Java SE Development Kit (5 або, бажано, 6 версії)

Перевірити версію: **java -version**

Завантажити:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

Опція: середовище розробки Eclipse (для використання плагіну, що полегшує розробку)

Завантажити: http://www.eclipse.org/

Встановлення App Engine Java SDK

Завантажити: http://code.google.com/appengine/downloads.html

Розпакувати у каталог, на який далі будемо посилатися як: **appengine-java-sdk**

Опція: плагін для IDE Eclipse

Інструкції зі встановлення: http://dl.google.com/eclipse/plugin/3.6

Перевірка працездатності середовища розробки – локальній запуск демонстраційної програми.

Вважатимемо, що робочий каталог – **appengine-java-sdk**.

Запуск (через консоль) демонстраційної програми.

- Windows: **bin\dev_appserver.cmd demos\guestbook\war**
- Linux: **./bin/dev_appserver.sh ./demos/guestbook/war**

При вдалому старті веб-серверу із демо-додатком у консолі буде виведено, серед іншого:

```
The server is running at http://localhost:8080/
```

Перегляд запущеного додатку через браузер:

Рис. 1.9. Запуск додатку на локальному комп'ютері

2.2. Підготовка середовища розробки Python для розробки додатків у хмарному сервісі

Вимоги до системи: Необхідна умова: Python 2.7 (із версією 3 є сумісність лише в новому середовищі, що є на етапі бета-тестування).

Перевірити версію: **python -version**

Завантажити: http://www.python.org/download/ (або встановити з репозиторіїв).

Установка App Engine SDK

Завантажити:

https://cloud.google.com/appengine/downloads#Google_App_Engine_SDK_for_Python

Unix

Розпакувати у каталог, на який далі будемо посилатися як: **google_appengine** та додати директорію до PATH:

```
>>unzip google_appengine_1.9.38.zip
```

>>export PATH=$PATH:/path/to/google_appengine/

Windows

Подвійним кліком запустити встановлення скачаного **GoogleAppEngine-1.9.38.msi**

Перевірка працездатності середовища розробки – локальній запуск демонстраційної програми. Вважатимемо, що робочий каталог – **google_appengine**.

Запуск (через консоль) демонстраційної програми:

- Windows: **dev_appserver.py demos\guestbook**
- Linux: **./dev_appserver.py ./demos/python/guestbook/**

з'явиться питання "Allow dev_appserver to check for updates on startup? (Y/n):" – треба погодитись.

При вдалому старті веб-серверу із демо-додатком у консолі буде виведено, серед іншого:

```
Running application guestbook on port 8080:
http://localhost:8080
```

Перегляд запущеного додатку через браузер:

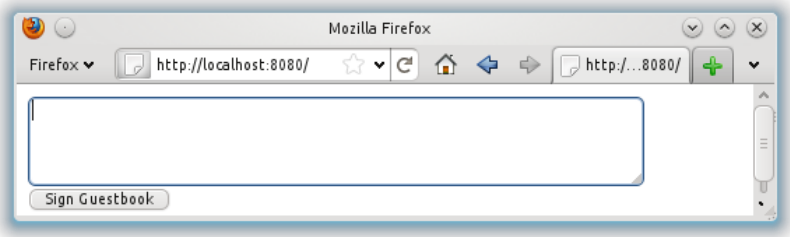

Рис. 1.10. Перегляд додатку через браузер

3. Консоль адміністрування

На сторінці https://console.cloud.google.com/appengine знаходиться консоль адміністрування розгорнутим додатком.

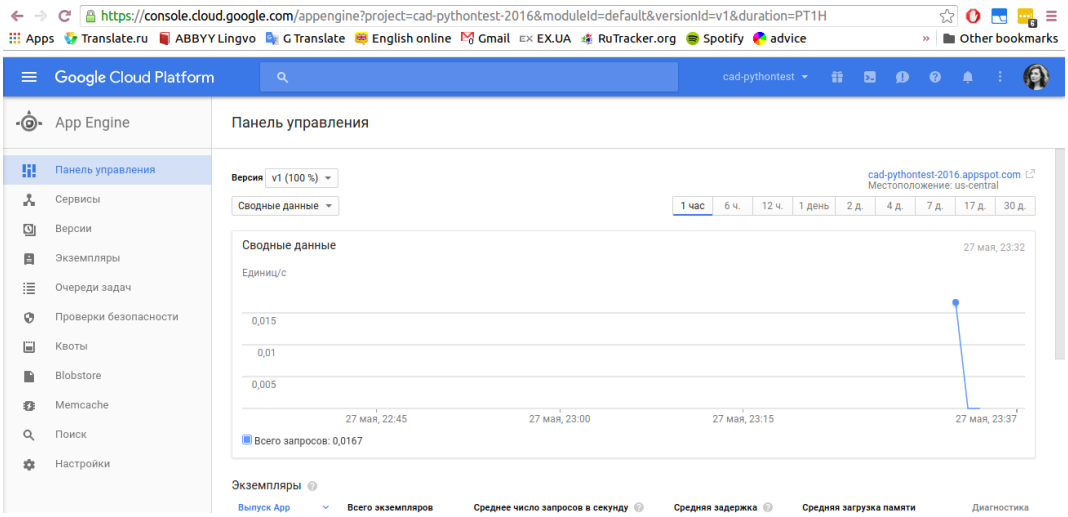

Рис. 1.11. Консоль адміністрування додатку

Консоль адміністрування надає доступ до детальної статистики та налаштувань додатку через такі пункти навігації:

- Панель керування – загальна інформація:
  Графічне представлення даних про додаток
  Дані про екземпляри
  Статус оплати
  Поточне навантаження
  Помилки додатку
  Помилки сервера
  Помилки клієнта
- Сервіси – сервіси, що виконуються під проектом
- Версії – можливість запуску/затримання/оновлення/перенесення трафіку версій
- Екземпляри – графічне представлення даних роботи екземплярів
- Черги задач – Черги push-запитів, Черги pull-запитів, Завдяння cron (запуск задач за разкладом)

- Перевірка безпеки – при активації Cloud Security Scanner він допомагає виявити проблеми безпеки в додатку і запобігти діяльність потенційних зловмисників
- Квоти – відомості про квоту для цього додатка
- Blobstore – інформація про blob об’єкти
- Memcache – дані про кешовану інформацію додатка
- Пошук – пошук по индексам
- Налаштування

Завдання

1. Описати роботу з механізмом реєстрації та налаштуванням хмарного середовища.
2. Описати процес встановлення App Engine SDK.
3. Ознайомитися з роботою консолі адміністрування хмарного додатку.

Частина 2. Розробка коду програми для роботи у хмарному середовищі

Мета роботи: набути практичних навичок розробки архітектури та коду програми в хмарному середовищі.

Задача: розробти код додатку згідно індивідуального завдання та розмістити його у хмарному середовищі, виконати необхідний обсяг тестування. Використовувати метод CRUD (Create, Read, Update, Delete) для роботи з даними хмарного додатку.

Короткі теоретичні відомості

Приклад додатку на Java

Приклад надано для мови програмування Java, наявність IDE Eclipse та плагіну Google не обов’язкова. Команди та шляхи до файлів вказані для Linux-систем.

1. Загальна структура файлів деякого проекту Project:

Project/

src/	вихідні коди на Java
war/	файли веб-додатку: JSP, зображення, дані тощо
WEB-INF/	конфігурація веб-додатку
lib/	JAR-файли бібліотек
classes/	скомпільовані файли

Приклад структури ресурсів демо-програми **guestbook**:

appengine-java-sdk\demos\guestbook

Підготуємо наступну структуру каталогів для проекту Example:

```
Example
   /src
   /war
      /WEB-INF
         /classes
            /Example
```

2. Вихідний файл – код простого сервлету Example.java

(Example/src/Example.java)

```
package Example;

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

public class ExampleServlet extends HttpServlet {
    public void doGet(HttpServletRequest req,
HttpServletResponse resp)
                throws IOException {

     resp.setContentType("text/plain");
```

```
        resp.getWriter().println("Hello! It works!");
    }
}
```

3. Файл конфігурації сервлету – web.xml (Example/war/WEB-INF/web.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE web-app PUBLIC
    "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app xmlns="http://java.sun.com/xml/ns/javaee"
version="2.5">
    <servlet>
        <servlet-name>Example</servlet-name>
        <servlet-class>Example.ExampleServlet</servlet-
class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Example</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

4. Додатковий файл конфігурації для App Engine – appengine-web.xml (Example/war/WEB-INF/appengine-web.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app
xmlns="http://appengine.google.com/ns/1.0">
    <application>appID</application>
    <version>1</version>
</appengine-web-app>
```

Значення елемента application повинно містити унікальний ідентифікатор додатку, створеного на http://appengine.google.com. Позначимо його тут і далі як **appID**.

5. Компіляція

Робочий каталог: **Example**

Примітка: для зборки необхідні бібліотеки AppEngine Java SDK

Шлях до каталогу із AppEngine Java SDK **позначено** як **appengine-java-sdk**

Тоді:

```
javac -cp appengine-java-sdk/lib/shared/servlet-api.jar
./src/ExampleServlet.java

mv ./src/ExampleServlet.class ./war/WEB-
INF/classes/Example/ExampleServlet.class
```

6. Перевірка

```
appengine-java-sdk/bin/dev_appserver.sh ./war
```

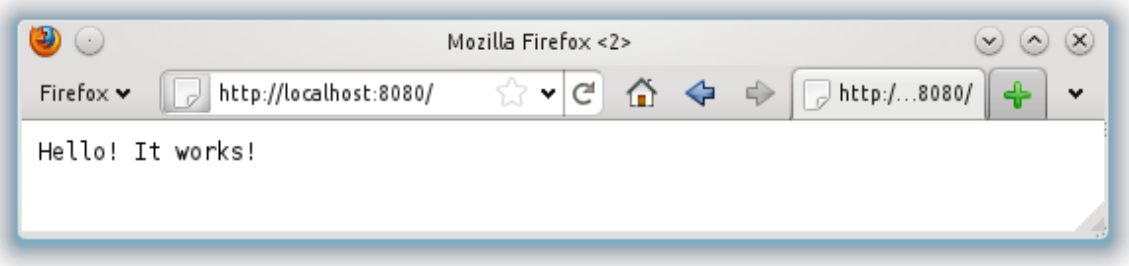

Рис. 2.1. Перевірка роботи додатку на локальному комп'ютері

7. Розгортання у хмарі

```
appengine-java-sdk/bin/appcfg.sh update ./war
```

Перед вивантаженням у хмару ця утиліта просить ввести логін та пароль:

```
Reading application configuration data...
Aug 30, 2011 1:16:14 AM
com.google.apphosting.utils.config.AppEngineWebXmlReader
readAppEngineWebXml
```

```
INFO: Successfully processed ./war/WEB-INF/appengine-web.xml
2011-08-30 01:16:14.316:INFO::Logging to STDERR via
org.mortbay.log.StdErrLog
Aug 30, 2011 1:16:14 AM
com.google.apphosting.utils.config.AbstractConfigXmlReader
readConfigXml
INFO: Successfully processed ./war/WEB-INF/web.xml
Beginning server interaction for appID...
0% Created staging directory at: '/tmp/appcfg4072153328197594448.tmp'
5% Scanning for jsp files.
20% Scanning files on local disk.
25% Initiating update.
Email:
Password for:
28% Cloning 4 application files.
40% Uploading 4 files.
52% Uploaded 1 files.
61% Uploaded 2 files.
68% Uploaded 3 files.
73% Uploaded 4 files.
77% Initializing precompilation...
90% Deploying new version.
95% Will check again in 1 seconds.
98% Will check again in 2 seconds.
99% Closing update: new version is ready to start serving.
99% Uploading index definitions.
Update completed successfully.
Success.
Cleaning up temporary files...
```

Перевірка:

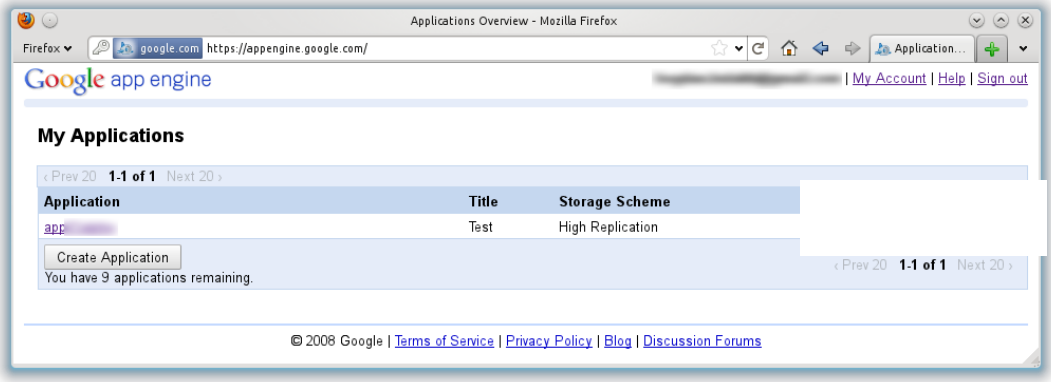

Рис. 2.2. Список додатків в акаунті GAE

Додаток доступний за адресою **appID**.appspot.com, де **appID** – унікальний ідентифікатор додатку.

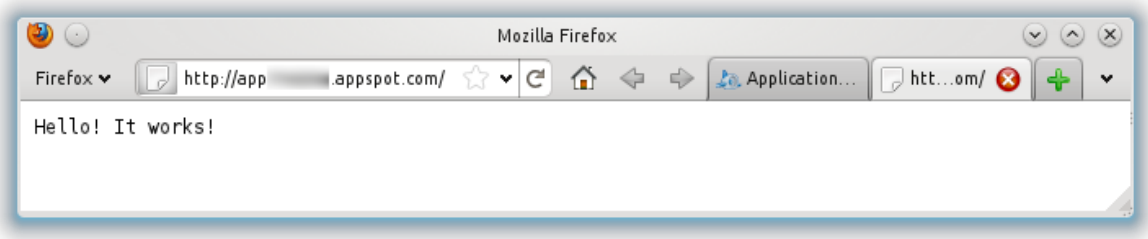

Рис. 2.3. Запуск додатку у хмарі.

Приклад додатку на Python

Приклад надано для мови програмування Python 2.7, наявність IDE не обов'язкова. Команди та шляхи до файлів вказані для Linux-систем.

1. Загальна структура файлів деякого проекту Project:

Project/

main.py Основний файл коду, який виконує всю роботу додатка.

app.yaml Конфігураційний файл використовується для вказання ідентифікатору проекту, версію програми конфігурації додатку.

Структуру файла можна скопіювати використовуючи шаблон проекту: google_appengine/new_project_template

2. Вихідний файл – main.py

```
import webapp2

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

Для побудови додатку використовується Web Server Gateway Interface (WSGI). App Engine включає в себе просту структуру веб-додатків, називається webapp2. Структура webapp2 вже встановлена в середовищі App Engine і в Python SDK App Engine, так що вам не потрібно пов'язувати його з кодом програми, щоб використовувати його.

Додаток webapp2 складається з двох частин:

- один або кілька класів RequestHandler, які обробляють запити і будують відповіді;
- примірника WSGIApplication, який направляє вхідні запити обробникам, засновуючись на URL.

Даний код визначає один обробник запитів, MainPage, відображений в кореневій URL (/). Коли webapp2 отримує запит GET HTTP до URL '/', він створює екземпляр класу MainPage і викликає його метод `get`. Усередині методу, інформація про запит доступна при використанні self.request. Як правило, метод задає властивості self.response для підготовки відповіді, а потім завершує свою роботу. Webapp2 відправляє відповідь, ґрунтуючись на кінцевому стані примірника MainPage.

3. Файл конфігурації сервлету – app.yaml

Файл app.yaml повинен знаходитися в кореневому каталозі програми.

В файлі налаштовуються параметри застосування App Engine. Цей файл визначає, відповідності URL запитів їх обробникам і статичним файлам. Цей конфігураційний файл також містить інформацію про код програми, такі як ідентифікатор додатка і останні ідентифікатор версії.

```
application: new-project-template
version: 1
runtime: python27
api_version: 1
threadsafe: yes
handlers:
- url: .*
  script: main.app
libraries:
- name: webapp2
  version: "2.5.2"
```

4. Перевірка

```
>>python dev_appserver.py папка_проекту/
```

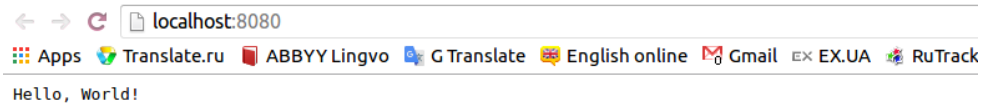

Рис. 2.4. Запуск тестового додатку у хмарі

5. Розгортання у хмарі

Зі сторінки адміністрування Google App Engine беремо id створеного раніше проекту:

```
>>python appcfg.py -A <YOUR_PROJECT_ID> -V v1 update
папка_проекту/
```

Перед вивантаженням у хмару ця утиліта запрошує доступ до акаунту через браузер:

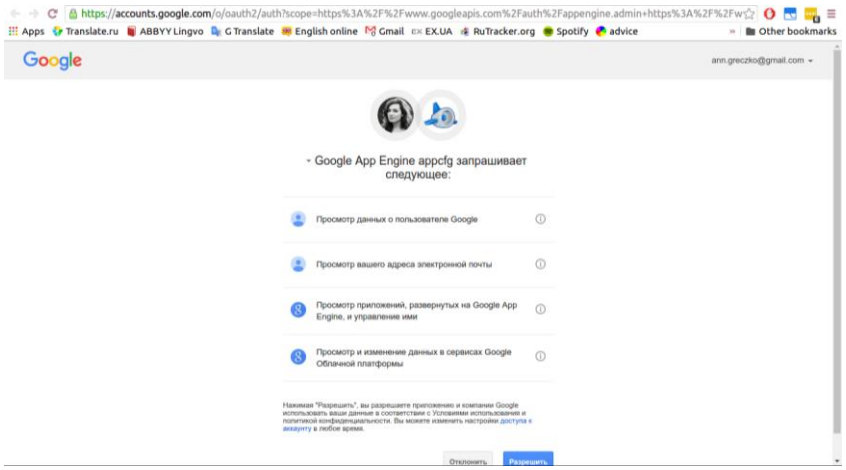

Рис. 2.4. Вивантаження додатку у хмару

6. Перевірка

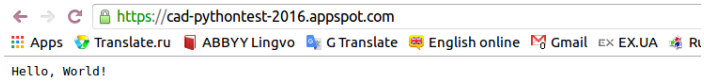

Рис. 2.5. Перевірка роботи додатку у хмарі

Додаток доступний за адресою appID.appspot.com, де appID – унікальний ідентифікатор додатку.

Приклад додатку на PHP

Приклад простого додатку на PHP для GAE можна знайти за посиланням https://github.com/GoogleCloudPlatform/getting-started-php

Завдання

1. Описати роботу з кодом додатку хмарного середовища.
2. Описати процес встановлення функцій App Engine SDK.
4. Розмістити додаток у хмарному середовищі.

Зміст звіту

1. Мета роботи.
2. Завдання роботи.

3. Оформлення результатів роботи.

4. Опис коду програми хмарного додатку.

5. Опис процедури деплоймента додатку у хмару.

6. Опис процеса тестування хмарного додатку.

7. Висновки.

Контрольні питання

1. Якими вимогами має керуватися розробник хмарних додатів ?

2. Які функції SDK хмарного сервісу використовувалися у процесі роботи?

3. Як відбувається процес деплоймента у хмарне середовище?