SUSS
SINGAPORE UNIVERSITY
OF SOCIAL SCIENCES

Aug 2024

Introduction to Data Programming Seminar 1

ICT233 Data Programming

Veronica Hu **huhe001@suss.edu.sg**

COURSE OVERVIEW

- Seminar 1 - Introduction to Data Programming
- Seminar 2 - Data Management
- Seminar 3 - Data Types and Structures
- Seminar 4 - Data Manipulation
- Seminar 5 - Data Mungling
- Seminar 6 - Data Scraping

COURSE OVERVIEW

Knowledge & Understanding (Theory Component)

- Develop analytic mindset to understand and interpret datasets
- Analyze HTTP and design parsing methods for information retrieval
- Apply Object-Relational Mapping (ORM) to manage information between objects and databases.
- Compose and utilize query languages for information retrieval from databases.

COURSE OVERVIEW

Key Skills (Practical Component)

- Perform ETL (Extraction, Transformation, Loading) and calculations using Python and Pandas.
- Develop programs to perform CRUD operations on database information.
- Formulate communication methods for exchanging information over the WWW.
- Conduct effective data visualization.

COURSE OVERVIEW

Learning Material (Optional)

- Dale, K. (2022). *Data Visualization with Python and JavaScript , 2nd Edition 2023*. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc.
- Severance, C. R. (n.d.). *Python for everybody: Exploring data in Python 3*. Copyright ~2009-Charles Severance.

Website(s)

- http://www.dr-chuck.com/

COURSE OVERVIEW

Learning Mode

- Self-study guided by the study guide units. Independent study will require at least **3 hours per week**.
- Working on **individual** assignments
- Seminar sessions (3 hours each session, 6 sessions in total)
- Online Office hours (2 hours each session, 6 sessions in total)

iStudyGuide

- Under L01 group -> [iBookstore] module

COURSE OVERVIEW

Interaction with Instructor and Fellow Students

- Collaborate and Share using Discussion Forum
- Online Office hours

Academic Integrity

COURSE OVERVIEW

Assessment Overview

Assessment	Description	Weight Allocation
Assignment 1	Online Quiz	6%
Assignment 2	Tutor Marked Assignment	24%
Examination	ECA	70%
TOTAL		100%

Chapter 1: ETL Process

Extract

- Extract raw data from various source systems
- e.g. databases, APIs, flat files, or other data repositories.

Transform

1) **Data Cleaning**: Removing errors, duplicates, and inconsistencies.
    - misspellings, incorrect values, missing data, or non-standard formats
    - e.g. inconsistent date formats "2024-08-14" vs. "14/08/2024".

2) **Data Integration**: Combining data from different sources into a unified format.

Chapter 1: ETL Process

Transform

3) **Data Transformation**: Sorting, filtering, aggregating, or enriching the data.
4) **Data Normalization**: Standardizing data to ensure consistency.
    - organizing data into tables to reduce redundancy
    - ensure that similar data is stored in a uniform manner

Load

- Load transformed data into the target system
    - e.g. data warehouse or database
- Facilitate easy querying and analysis.

Chapter 1: ETL Process

Seminar 1

Extract

- One or more source systems containing customer, financial, or product data (CRM, Accounting system, Warehouse, MES)
- Files types - Flat files, XML, Oracle, IBM DB2, SQL Server,, IBM Websphere MQ, ODBC, JDBC, Hadoop Distributed File System (HDFS), Hive/HCatalog, JSON, Mainframe (IBM z/OS), Salesforce.com, SAP/R3

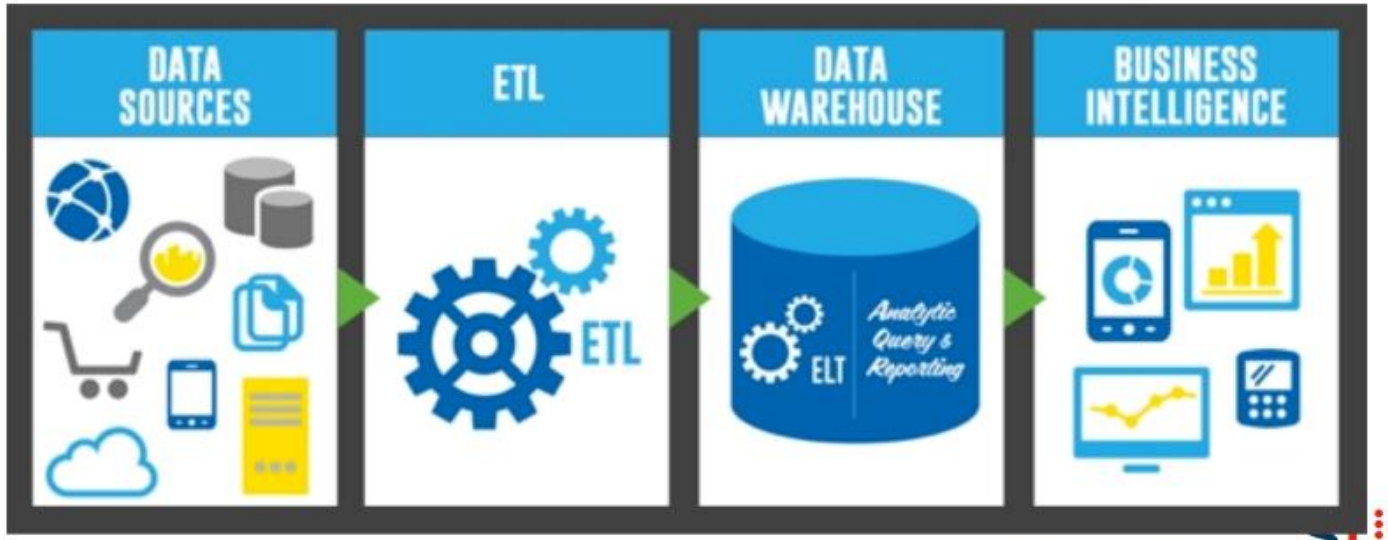
Transform

- Applying business rules, cleansing, and validating the data.
- Aggregation, Copy, Join, Sort, Merge, Partition, Filter, Reformat, Lookup
- Mathematical: +, -, x, /, Abs, IsValidNumber, Mod, Pow, Rand, Round, Sqrt, ToNumber, Truncate, Average, Min, Max
- Logical: And, Or, Not, IfThenElse, RegEx, Variables
- Text: Concatenate, CharacterLengthOf, LengthOf, Pad, Replace, ToLower, ToText, ToUpper, Translate, Trim, Hash
- Date: DateAdd, DateDiff, DateLastDay, DatePart, IsValidDate
- Format: ASCII, EBCDIC, Unicode

Load

Load the results into one or more target systems such as a data warehouse, datamart, or business intelligence reporting system.

Output: Flat files, XML, Oracle, IBM DB2, SQL Server, Teradata, Sybase, Vertica, Netezza, Greenplum, ODBC, JDBC, Hadoop Distributed File System (HDFS), Hive/HCatalog, Mainframe (IBM z/OS), Salesforce.com, Tableau, QlikView



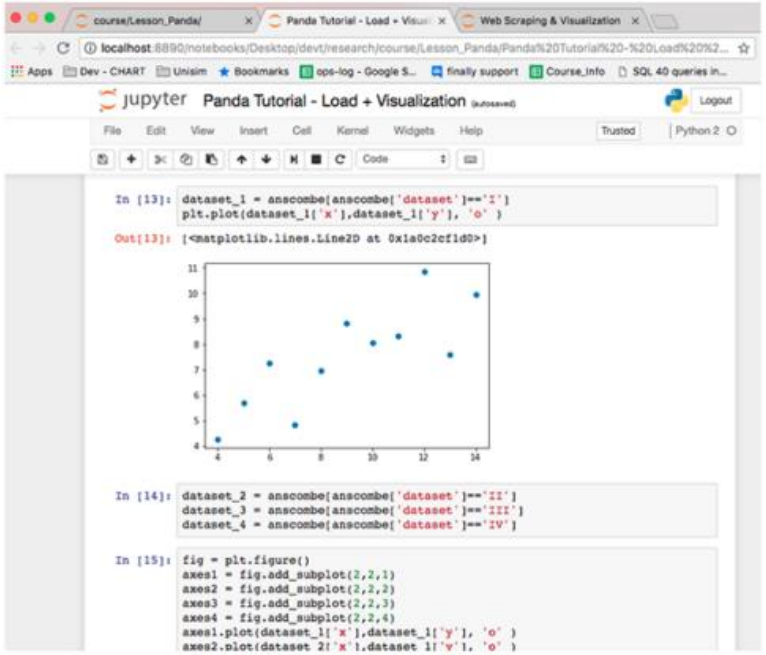
Chapter 2: Using Python for Data Processing

2.1 Setting Up Environment

1) Python
2) Anaconda
3) Jupyter Notebook

Python Primer Exercises – Pandas for Everyone Appendix

1) Installation
2) Command Line
3) Project Template
4) Using Python
5) Working Directory
6) Environment
7) Install Package



Chapter 2: Using Python for Data Processing

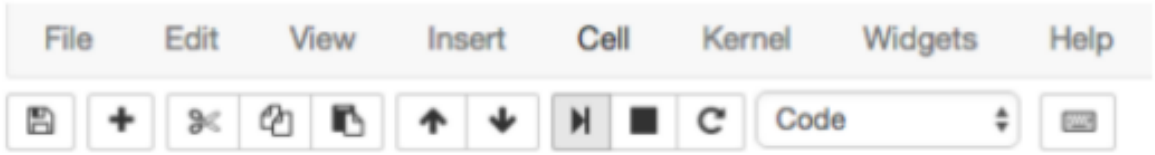


Figure 1.3 Jupyter Notebook Menu, Running Codes
(Source: Created by developer)

```
In [2]: print ('Hello World')
        Hello World
```

```
my_project/
    |
    |- data/
    |    |
    |    + data.csv
    |
    |- *src/
    |    |
    |    + script.py
    |
    +- output/
```

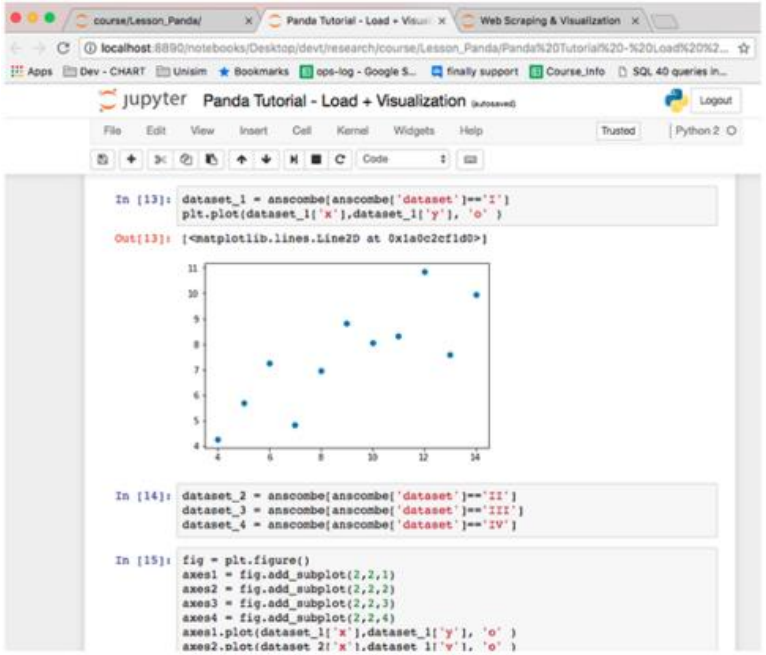
Chapter 2: Using Python for Data Processing

2. 2 Using Python to Read and Write Data

1) Getting Data
2) Dictionary and JSON format
3) CSV, TSV, Other formats

Python Primer Exercises – Pandas for Everyone Appendix

8) Import libraries
9) List
10) Tuples
11) Dictionaries
12) Slicing Values
13) Loops
14) Comprehensions
15) Functions



Chapter 2: Using Python for Data Processing

2.2 Using Python to Read and Write Data

- Getting Data
- Dictionary and JSON format

```
data = ['John', 'Tom', 'Mary', 'Jane']
class_list = data
```

```
with open('classdata.txt') as file:
    class_list = file.readlines()
for name in class_list:
    print(name)
```

```
class_list = [
    {'name': 'John', 'email': 'john@gmail.com', 'id': 1},
    {'name': 'Mary', 'email': 'mary@gmail.com', 'id': 2},
    {'name': 'Peter', 'email': 'peter@gmail.com', 'id': 3}
]
```

Chapter 2: Using Python for Data Processing

2.2 Using Python to Read and Write Data

- CSV, TSV, Other formats - Input

start	end	west	east	central	south	north
2023-02-04T06:00:00+08:00	2023-02-04T12:00:00+08:00	Partly Cloudy (Day)	Partly Cloudy (Day)	Partly Cloudy (Day)	Partly Cloudy (Day)	Partly Cloudy (Day)
2023-02-04T12:00:00+08:00	2023-02-04T18:00:00+08:00	Thundery Showers	Thundery Showers	Thundery Showers	Thundery Showers	Thundery Showers
2023-02-04T18:00:00+08:00	2023-02-05T06:00:00+08:00	Cloudy	Cloudy	Cloudy	Cloudy	Cloudy

Figure 2.5 CSV Format (Source: Created by developer)

```
import csv

with open('data.csv') as f:
    reader = csv.reader(f)
    for row in reader:
      print (row)
```

Output:

```
['car_park_no', 'address', 'x_coord', 'y_coord', 'car_park_type',
'type_of_parking_system', 'short_term_parking', 'free_parking', 'night_parking'] ['ACB',
'BLK 270/271 ALBERT CENTRE BASEMENT CAR PARK', '30314.7936', '31490.4942', 'BASEMENT CAR
PARK', 'ELECTRONIC PARKING', 'WHOLE DAY', 'NO', 'YES']
['ACM', 'BLK 98A ALJUNIED CRESCENT', '33758.4143', '33695.5198', 'MULTI-STOREY CAR PARK',
'ELECTRONIC PARKING', 'WHOLE DAY', 'SUN & PH FR 7AM-10.30PM', 'YES']
```

Chapter 2: Using Python for Data Processing

2.2 Using Python to Read and Write Data

- Output CSV file as Dictionary

```
import csv
with open('forecast.csv') as f:
    # read in all the rows from csv
    reader = csv.DictReader(f)
```

- Output Python Dictionary as CSV file

```
# create a python dictionary
my_dict = [{"fruit":'apple', "color":'red'}, {"fruit": 'blue berry', "color": "blue"}]

# prepare the header (obtain the keys from dictionary)
cols = my_dict[0].keys()
print('cols:',cols)

with open('newdata.csv', 'w') as f:
    # write the header in
    print("***header***", ','.join(cols))
    f.write(','.join(cols)+'\n')

    # write rest of the data into csv
    for o in my_dict:
        row = [str(o[col]) for col in cols]
        print("***row***",','.join(row))
        f.write(','.join(row) +'\n')
```

Chapter 2: Using Python for Data Processing

2.3 Regular Expression

1) String Operations
2) String & Number Formatting
3) Regular Expressions – Special Characters
4) Escape Characters

```
txt = '0123456789'
print(txt[:5])
print(txt[3:])
```

01234
3456789

```
mystr = 'Extract the number after colon :0.8475'
number = mystr[mystr.find(':')+1:]
print(float(number))
```

0.8475

```
a = 'Value of variable A'
b = 123
print('VAR a is {} and VAR B is {}'.format(a, b))
```

VAR a is Value of variable A and VAR B is 123

```
print('This is a large number {}'.format(2719249123))
print('This is a large number {:,}'.format(2719249123))
```

This is a large number 2719249123
This is a large number 2,719,249,123

Chapter 2: Using Python for Data Processing

2.3 Regular Expression

- https://docs.python.org/3.8/library/re.html

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
Return-Path: <postmaster@collab.sakaiproject.org>
Received: from murder (mail.umich.edu [141.211.14.90])
          by frankenstein.mail.umich.edu (Cyrus v2.3.8) with LMTPA;
          Sat, 05 Jan 2008 09:14:16 -0500
X-Sieve: CMU Sieve 2.3
Received: from murder ([unix socket])
          by mail.umich.edu (Cyrus v2.2.12) with LMTPA;
          Sat, 05 Jan 2008 09:14:16 -0500
Received: from holes.mr.itd.umich.edu (holes.mr.itd.umich.edu [141.211.14.79])
         by flawless.mail.umich.edu () with ESMTP id m05EEFR1013674;
         Sat, 5 Jan 2008 09:14:15 -0500
Received: FROM paploo.uhi.ac.uk (app1.prod.collab.uhi.ac.uk [194.35.219.184])
         BY holes.mr.itd.umich.edu ID 477F90B0.2DB2F.12494 ;
          5 Jan 2008 09:14:10 -0500
Received: from paploo.uhi.ac.uk (localhost [127.0.0.1])
         by paploo.uhi.ac.uk (Postfix) with ESMTP id 5F919BC2F2;
         Sat,  5 Jan 2008 14:10:05 +0000 (GMT)
Message-ID: <200801051412.m05ECIaH010327@nakamura.uits.iupui.edu>
Mime-Version:
```

Chapter 2: Using Python for Data Processing

2.3 Regular Expression

- http://www.rexegg.com/regex-quickstart.html
- Regular Expressions – Special Characters
- Escape Characters

S/N	Character	Legend	Pattern	Sample Match
1	\d	one digit from 0 to 9	\d\d	25
2	\w	ASCII letter, digit or underscore	\w\w\w	B_1
3	\s	space, tab, newline, carriage return, vertical tab	a\sb\sc	a b c
4	\S	one character that is not whitespace	\S\S\S\S	a3d_
5	.	any character	x.y	xry or xly
6	\	Escapes a special character	\.\{\}	.{}
7	+	One or more	\w--\w+	A--b1_1
8	{3}	Exactly three times	\w{3}	Aa_
9	*	Zero or more times	A*B*C*	AAACCC
10	?	Once or none	plurals?	plural

Chapter 2: Using Python for Data Processing

2.3 Regular Expression

```
import re
hand = open('mbox-short.txt')
for line in hand:
    line = line.rstrip()
    x = re.findall('\S+@\S+', line)
    if len(x) > 0:
        print(x)
```

[a-zA-Z0-9]\S*@\S*[a-zA-Z0-9]

The code with the revised regular expression should produce output as follows:

```
['stephen.marquard@uct.ac.za']
['postmaster@collab.sakaiproject.org']
['200801051412.m05ECIaH010327@nakamura.uits.iupui.edu']
['source@collab.sakaiproject.org']
```

```
import re
hand = open('mbox-short.txt')
for line in hand:
    line = line.rstrip()
    x = re.findall('[a-zA-Z0-9]\S*@(\S*[a-zA-Z0-9])', line)
    if len(x) > 0:
        print(x)
```

The output will be the list of domain names extracted from the emails.

Chapter 3: Data Processing via Internet

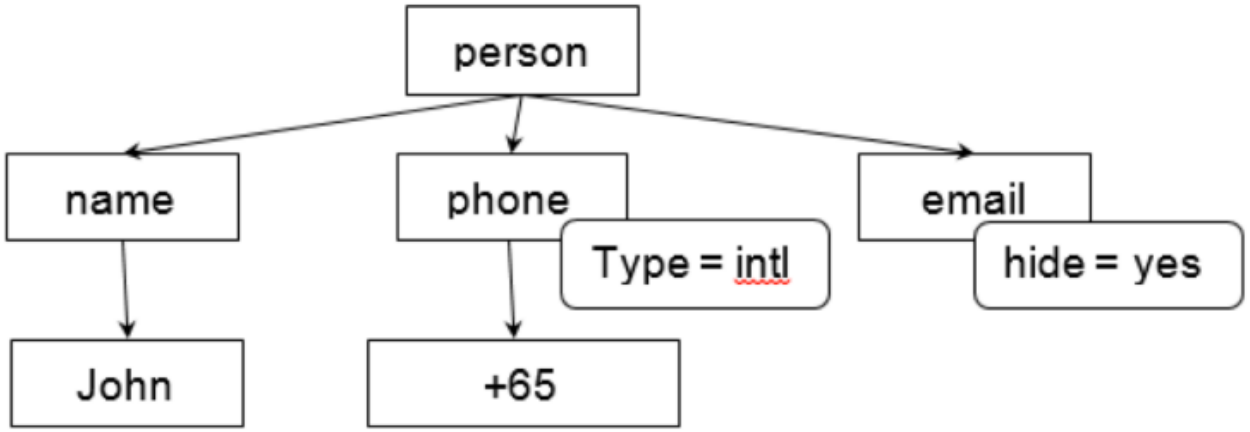
3.1 Reading Web Page Using HTTP

- HyperText Transport Protocol – HTTP
- Getting web data with Sockets
- Using urllib
  - Text
  - Image
- Using Regular Expressions

Chapter 3: Data Processing via Internet

3.2 Using Web Services

- Parsing XML



Chapter 3: Data Processing via Internet

3.2 Using Web Services

- Parsing JSON

```
class_list = '''
[
    {"id": 1, "name": "John", "email": "john@gmail.com"},
    {"id": 2, "name": "Mary", "email": "mary@gmail.com"},
    {"id": 3, "name": "Peter", "email": "peter@gmail.com"}
]
'''

import json

info = json.loads(class_list)
print(len(info))
for item in info:
    print(item['id'], item['name'], item['email'])
```

Chapter 3: Data Processing via Internet

3.2 Using Web Services

- Parsing Data from Web Services

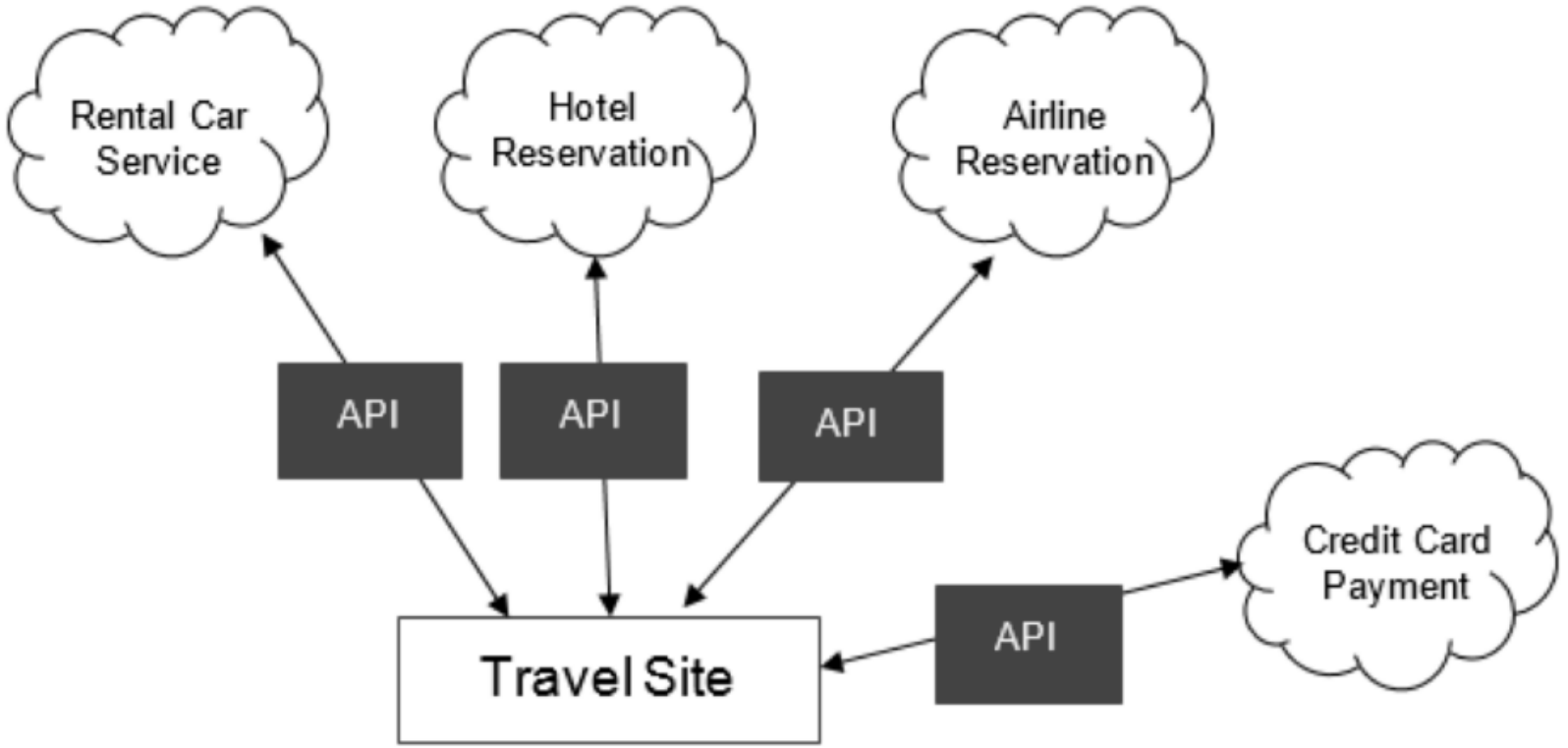
```
import urllib.request, json

# The resources gives information on graduates' salary by different universities.
# The data provided by the web resources are all in JSON format.

url = '''
 https://data.gov.sg/api/action/datastore_search?resource_id=3a60220a-80ae-4a63-afde-413f05328914
  '''

req = urllib.request.Request(url, headers={'User-Agent':'Mozilla/5.0'})
html = urllib.request.urlopen(req).read()

# The result is a dictionary file with nested arrays inside
data = json.loads(html)

print (len(data['result']['records'])) # Number of records
print (data['result']['records'][0].keys()) # which are the column names of each record from this resource
```

```
100
dict_keys(['school', 'degree', 'university', 'gross_monthly_median', 'gross_mthly_25_percentile', 'basic_monthly_median', 'empl
oyment_rate_ft_perm', 'gross_mthly_75_percentile', 'gross_monthly_mean', 'basic_monthly_mean', 'year', '_id', 'employment_rate_
overall'])
```

Chapter 3: Data Processing via Internet

3.3 Application Programming Interfaces

- Service-Oriented Architecture (SOA)



Chapter 3: Data Processing via Internet

3.3 Application Programming Interfaces

- Representational State Transfer Architecture (REST)

  - performing CRUD operations over web resources

HTTP request	Response
GET /students	Return list of all students
GET /student/<id>	Return detail of student of <id>
POST /student	Add a new student resource
PUT /student/<id>	Update/Change details of student of <id>
DELETE /student/<id>	Delete record of student of <id>

Chapter 3: Data Processing via Internet

3.3 Application Programming Interfaces

- Representational State Transfer Architecture (REST)

```
import requests
URI = 'https://jsonplaceholder.typicode.com/posts/'
# return list of all students from the url (JSON list of all the posts in the database)
r = requests.get(URI)
print (r.text)
```


```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
     "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto"
  },
  {
    "userId": 1,
    "id": 2,
    "title": "qui est esse",
     "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla"
  },
…
]
```

SEMINAR LEARNING OUTCOME

INTRODUCTION TO DATA PROGRAMMING – LEARNING OBJECTIVES

1) Recap using Python programming language to retrieve, create, and update data via a variety of data sources and file types.
2) Reflect on the various techniques of processing and parsing information, e.g., using regular expression libraries to recognize and process text patterns.
3) Understand how information is presented, delivered, and consumed on the internet via HTTP (Hypertext Transport Protocol).
4) Understand the concept of web services as a mean to provide and exchange information between consumers and providers of information.
5) Appreciate the architectural philosophies behind web services.
6) Know the popular data formats (XML, JSON) and their relevant usage patterns in information retrieval and exchanges.

THANK YOU