LAUREL
TECHNOLOGY
SOLUTIONS

```
require 'spec_helper'
require 'rspec/rails'

require 'capybara/rspec'
require 'capybara/rails'

Capybara.javascript_driver =
Category.delete_all;
Shoulda::Matchers.configure
  config.integrate do
    with.test_framework
    with.library :rails
  end
end

# Add additional requires below this line.

# Requires supporting ruby files
# spec/support/ and its subdirectories.
# run as spec files by default.
# in _spec.rb will both be required
# run twice. It is recommended
# end with _spec.rb.
```

PYTHON ETL REPORT

Introduction

Expanding internationally is one of the most complex and chaotic tasks an organisation could take, with several systems and activities having to be disrupted and redesigned to make it possible. According to Evans (2015), The process of expanding a small firm internationally is dynamic and difficult. A crucial foundation is laid by gaining a thorough grasp of the targeted markets, the competitors, existing local market trends, and the needs to effectively launch and drive growth. With that being said, there are several potential technologies solutions that could Laurel Technology in expanding to Seoul, South Korea from the United Kingdom. This is made possible and easier by the rapid growth of technology in the 21st century with massive computing power scaled down to the smallest sizes. An example is the new iPhone with a 4nm chip.

One of these technological solutions is **Cloud Computing and services**

Large corporate firms may create, test, and deploy apps using cloud-based platforms, as well as store, backup, and recover data. A new generation of businesses may now provide services online regardless of location thanks to cloud computing. Cloud computing, in the words of McKinsey, is "accessing computer resources given over networks rather than operating software or storing data on a local computer."(Knutsson, Severin, & Söderberg, 2012). There are several ways in which cloud computing could advance the international expansion of businesses.

For one, it could “jumpstart” the process. If Laurel technology is going to set up a data centre in another country, they would generally require an on-site facility, hardware and other infrastructure that could take a large amount of time and resources to set up. However, with cloud computing, they can fast-track this process. An expansion plan of five to ten years could become two to three years using cloud computing.

Cloud computing and services also give speed and flexibility that Laurel technology would need to scale up fast. With cloud computing, the IT infrastructure would be set up rather seamlessly and this would give time to focus on other business aspects that the organization needs during the expansion.

A challenge that is normally encountered when organizations need to transmit data overseas is Data Sovereignty. Different regions have different data security requirements, with some not allowing the data gathered in their country to be taken outside. With cloud services, the data can exist in its country of origin and be assessed internationally with several compliance checks being adhered to.

Companies that want to expand abroad must have cost as one of their constraints and/or considerations. Using cloud services greatly reduces the initial cost of setting up IT infrastructure and in addition to that, the cost can be budgeted given that it's in form of a subscription cost. This provides an added advantage to planning and execution.

Another technology solution is **Communication-as-a-Service(CaaS)**

Communication is critical to the success of any business at any stage. Effective communication is essential for the daily-to-day operations, growth, and expansion of any organization. Communication can be in various means such as Text, Voice and Video calls, Emails, etc and can either be Upward, Downward, Lateral or External. CaaS is an enterprise communications service that is outsourced. These cloud service providers are in charge of managing the infrastructure and software necessary to offer their clients voice-over IP (VoIP) services, instant messaging (IM), and video conferencing capabilities. (Kulkarni, Jadhav, Bhuse, Bankar, & Sushma, 2013)

The international workforce of Laurel technology would need to be in constant communication with the team in the UK. CaaS provides the means of seamless communication and provides the same software to all teams. As language would be a barrier in communications, several CaaS providers also provide translation services to improve the quality of communication between international teams.

Another technological solution is **Artificial Intelligence (AI)**

A technological advancement that is having a significant effect on globalization is artificial intelligence (AI). Businesses are utilizing AI to automate a variety of operations and procedures, including those communications in international trade (Stephens, 2022).

Laurel technology could use Artificial Intelligence to boost productivity in areas such as business communication with the use of chatbots, automated answering machines, and others, business management by automation of business processes, forecasting, filtering of spam, automated assistance and others, e-commerce by smart searches, detection of fraud, recommendation of products and others, marketing by customer segmentation, pattern/image recognition, content recommendation, personalization of ads/ ad targeting, sentiment analysis and others. (NIBusinessInfo, 2020).

Using RDBMS to store combined data

Data points that are connected to one another are stored and accessible in a relational database, which is a form of a database. The relational model, an easy-to-understand method of expressing data in tables, is the foundation of relational databases. Each table row in a relational database is a record with a distinct ID known as the key. It is simple to determine the associations between data points since the table's columns carry the properties of the data and each record typically includes a value for each property (Oracle, 2021).

Laurel technology made use of Relational Database because of its advantage of speed, simplicity, accessibility, data integrity, collaboration and security. However, there are various drawbacks to the use of Relational Databases and should be taken into consideration by Laurel Technology. They include

Maintenance: As the amount of data inputted into the database increases, the maintenance of such database also becomes difficult. Highly skilled Data Engineers have to be present and spend a lot of time in the maintenance process

Cost: Setting up and maintaining the relational database system is expensive. Smaller firms may find the initial purchase of the software to be pretty expensive, but the situation is made worse when you take into account the need to hire a qualified technician who is also familiar with that particular type of software.

Performance: The number of tables in a relational database is always a performance factor. The response time to queries will be slower if there are more tables. More data also slows down the system and makes it harder to discover information in the long run. As a result, relational databases are notorious for being slower.

Complexity: It is challenging to depict complicated relationships between items in relational databases since they can only store data in tabular form. This is problematic since many applications need several tables to contain all the data needed by their application logic.

Scalability: The relational database's structure changes and becomes challenging to manage when used across numerous servers, especially when the amount of data is considerable. As a result, the data cannot be scaled over many physical storage servers. In the end, its performance is impacted, including load times and data availability issues. Performance difficulties like latency and availability will arise when the database grows larger or is dispersed across additional servers, which will hurt overall performance.

Longevity: Not only does the relational database depend on many tables, but it may also grow slower. The complexity of a system rises when there are several tables and data sources present. Depending on how many users are signed onto the server at once, it may cause queries to respond slowly or even fail.

As a matter of tradition, relational databases are used to manage data in several organizations. However, it has its downsides. The above limitations should be sufficient to convince Laurel technology to migrate to a cloud base platform. (Akhtar, 2021; Roor, 2021)

Big Data and Its challenges

Big data is a term used to describe datasets that are too large to be captured, stored, and analysed by conventional database software tools. This definition includes a shifting definition of how large a database must be to qualify as big data, and it is purposely ambiguous (Prabhu & Arpitha, 2014).

The biggest issue that Laurel technology would face dealing with data from non-UK branches is

Data Sovereignty

When we talk about data sovereignty, we're talking about the laws that apply to data because of the place where it is physically situated. Data transmission outside of the original nation is restricted in several countries. Additionally, several nations have privacy laws that limit the transmission of personal information to other parties. As a result, businesses will be responsible for different types of data in various places (Cloudian, 2021).

Data sovereignty could serve as a barrier to cloud adaption because Laurel technology would have to be concerned if they are legally permitted to transfer the data to the cloud. Compliance with certain region's sovereignty could lead to increased operational costs in form of providing compliance training and reduce security as there must be transparency to prove sovereignty compliance.

A solution for Laurel technology would be to Leverage the capabilities of major cloud services provider, that way each dataset can be finetuned to meet the sovereignty of the location it is being stored. Another solution would be to simplify things by adapting all dataset to the strictest data sovereignty regulation.

Data Confidentiality

Individuals have the right to privacy and control over how their personal information is used and shared. A larger definition of confidentiality would include both businesses looking to safeguard their customers and competition, as well as governments eager to maintain their dominance and society (Vadrevu, Suggala, & Varma, 2018). Example of Confidential data include Personal data, trade secrets and other restricted business data (Galicia, 2020).

Laurel technology can ensure confidentiality by restricting access to the data, encrypting the data, investing in cybersecurity, take physical measures to ensure security and other security methods. However one of the easiest method of ensuring security is to use cloud base data services.

Data Scaling

Data scaling might become an issue in the future as Laurel technology gradually gathers more data and increase their data volume overtime. Different sizes of data have different requirement and as data get bigger in size, the requirements become more. With the challenges of acquiring new services comes the challenge of increasing costs.

Insufficient Data Experts available

The truth is that big data is still a challenging area to comprehend since it requires using sophisticated tools and technology and thus, the number of professionals available that can effectively handle big data and its migration is limited. Laurel technology would have to look for competent data professionals and aside from that, also involve themselves in training employees to fill the skill gap.

Faulty Integration

Integration done incorrectly has detrimental effects. For instance, data leakage or desynchronization may happen when several departments of a business employ various software and hardware solutions. A large data system's structure ends up being needlessly complicated and expensive to maintain since not all solutions are suited for end-to-end integration. Deep automation, individual subsystem integration via an API, and the rejection of human system control are the answers to these data issues. The expenditures of this modernisation will be high, but in the long run, the chance of the issues will be reduced (Piluta, 2022).

Poor Data Quality

Data silos or poor data quality—which may be unstructured, have diverse formats, include duplicate entries, etc.—are another common issue for businesses working with big data. Because data cannot be accessible centrally, even a straightforward computation of quarterly costs may contain significant inaccuracies because the statistics from various areas of the company are not in sync. Note that the quantity and likelihood of mistakes will continuously rise as big data software becomes more complicated.

Laurel technology can resolve these by practicing data consolidation as the initial strategy. In this situation, they create a database of important information that serves as a "single source of truth." The next step is to build a data directory where all of the entries will be organized and sorted. They can get rid of duplicates in this situation. It goes without saying that this transformation of enormous volumes of data must occur slowly, thus it's critical to identify the data that is most utilized and crucial to your company (Piluta, 2022).

Big data certainly do have it's challenges especially in the case of an organization expanding internationally. Fortunately, there are several steps that could be taken to ensure smooth transition from one phase to another. If all the steps are adhered to, Laurel technology would have minimal complications that comes with the international expansion

References

1. Akhtar, Z. (2021). Relational Database Benefits and Limitations (Advantages & Disadvantages). Retrieved from https://databasetown.com/relational-database-benefits-and-limitations/
2. Cloudian. (2021). Data Sovereignty in the Cloud: Key Considerations. Retrieved from https://cloudian.com/guides/data-protection/data-sovereignty-in-the-cloud-key-considerations/
3. Evans, M. (2015). 10 Key Steps to Expanding Your Business Globally. Retrieved from https://www.forbes.com/sites/allbusiness/2015/03/04/10-key-steps-to-expanding-your-business-globally/?sh=648b3e203803
4. Galicia, G. (2020). Data Confidentiality: How Can Businesses Protect Their Data? Retrieved from https://penneo.com/blog/data-confidentiality/
5. INSIDER. (2021). 5 ways cloud technology can fast-track international expansion. Retrieved from https://www.businessinsider.com/sc/how-the-cloud-can-help-with-international-expansion-2021-12?r=US&IR=T
6. Knutsson, S., Severin, M., & Söderberg, J. (2012). Business in the Cloud. (Master Thesis). Uppsala University,
7. Kulkarni, G., Jadhav, M., Bhuse, S., Bankar, H., & Sushma, A. (2013). Communication As Service Cloud. International Journal of Computer Networking, Wireless and Mobile Communications, 3(1), 149-156.
8. NIBusinessInfo. (2020). Artificial intelligence in business. Retrieved from https://www.nibusinessinfo.co.uk/content/examples-artificial-intelligence-use-business
9. Oracle. (2021). What Is a Relational Database Retrieved from https://www.oracle.com/database/what-is-a-relational-database/
10. Piluta, R. (2022). 12 Big Data Issues Growing Companies Face. Retrieved from https://nix-united.com/blog/12-big-data-issues-growing-companies-face/
11. Prabhu, B. V. B., & Arpitha, S. (2014). Study on Causes and Issues of Big-Data. International Journal of Engineering Sciences & Research Technology 3(2), 626-629.
12. Roor, M. (2021). 6 Advantages and Disadvantages of Relational Database | Limitations & Benefits of Relational Database Retrieved from https://www.hitechwhizz.com/2021/04/6-advantages-and-disadvantages-limitations-benefits-of-relational-database.html
13. Stephens, A. (2022). How Has Communication Technology Expanded International Business?
14. Vadrevu, P. K., Suggala, R. K., & Varma, G. T. (2018). Big Data Expansion and Challenges. International Journal of Engineering Research & Technology (IJERT), 4(34).

Appendix

```
import csv
import json
import sqlite3      ##sqlite3 is part of the standard python library
from xml.etree import ElementTree ##xml is part of the standard python library
from pony.orm import *


####### To unify the data, I have to first change the headers
####### in the vehicle data to match those in the other files

# rename the headers of the vehicle data in the csvfile
with open('Data//user_data.csv', 'r', newline='') as file:
  with open('outputs//vehicle_data.csv', 'w', newline='') as newfile:
    reader = csv.reader(file)
    writer = csv.writer(newfile)
    #read the first row
    next(reader, None)
    #replace the values of the first row
    writer.writerow(['firstName', 'lastName','age','sex','vehicle_make',
'vehicle_model','vehicle_year','vehicle_type'])
    #write the remaining rows
    for row in reader:
      writer.writerow(row)

###### Next, I will convert the xml file to csv to make it easier to manipulate

#read the xml file and get the root
xml = ElementTree.parse('Data//user_data.xml')
root = xml.getroot()

#open a new csv file to write data into
statusfile = open('outputs//status_data.csv', 'w', encoding='utf-8', newline='')
statusfile_writer = csv.writer(statusfile)

#insert the column header names
statusfile_writer.writerow(['firstName', 'lastName', 'age', 'sex', 'retired',
                'dependants', 'marital_status', 'salary', 'pension',
                'company', 'commute_distance', 'address_postcode'])
```

```
#loop through the xml root and get all attributes
for child in root:
  firstName = child.attrib['firstName']
  lastName = child.attrib['lastName']
  age = child.attrib['age']
  sex = child.attrib['sex']
  retired = child.attrib['retired']
  dependants = child.attrib['dependants']
  marital_status = child.attrib['marital_status']
  salary = child.attrib['salary']
  pension = child.attrib['pension']
  company = child.attrib['company']
  commute_distance = child.attrib['commute_distance']
  address_postcode = child.attrib['address_postcode']
  #determine the content of each row
  csv_line = [firstName, lastName, age, sex,
              retired, dependants, marital_status,
              salary, pension, company, commute_distance,
              address_postcode]

  #write each row to csv file
  statusfile_writer.writerow(csv_line)
#close the file
statusfile.close()

###### Next I would join the data together using a temporary sqlite table
###### and export the data into a csv file to form the main table

#open all files
vehicle_file = csv.reader(open('outputs//vehicle_data.csv'))
status_file = csv.reader(open('outputs//status_data.csv'))
banking_file = json.load(open('Data//user_data.json'))

##preparing the json to be imported into the database
#get all the column header names
columns = [] ##empty list that would eventually contain the column header
names
column = [] ##empty list that temporarily stores the column header names
for data in banking_file:
  column = list(data.keys())
  for col in column:
     if col not in columns:
        columns.append(col)
```

```
#get all the values of the data
values = [] ##empty list that would eventually contain all the values
value = [] ##empty list that temporarily hold the values
for data in banking_file:
  for i in columns:
     value.append(str(dict(data).get(i)))
  values.append(list(value))
  value.clear()

#create a connection to a temporary database in memory
conn = sqlite3.connect(':memory:')

#create a cursor
c=conn.cursor()

#define the tables
vehicle_table = '''CREATE TABLE vehicle(
          first_name TEXT,
          last_name TEXT,
          age INTEGER,
          sex TEXT,
          vehicle_make TEXT,
          vehicle_model TEXT,
          vehicle_year INTEGER,
          vehicle_type TEXT
          );'''

status_table = """CREATE TABLE status(
          first_name TEXT,
          last_name TEXT,
          age INTEGER,
          sex TEXT,
          retired TEXT,
          dependants INTEGER,
          marital_status TEXT,
          salary INTEGER,
          pension INTEGER,
          company TEXT,
          commute_distance REAL,
          address_postcode TEXT
          );"""
```

Appendix

```
banking_table = """CREATE TABLE banking(
          first_name TEXT,
          last_name TEXT,
          age INTEGER,
          iban TEXT,
          credit_card_number TEXT,
          credit_card_security_code INTEGER,
          credit_card_start_date TEXT,
          credit_card_end_date TEXT,
          address_main TEXT,
          address_city TEXT,
          address_postcode TEXT,
          debt );"""

#create the tables
c.execute(vehicle_table)
c.execute(status_table)
c.execute(banking_table)

#define the table population
vehicle_insert_records = '''INSERT INTO vehicle(first_name, last_name,
                  age, sex, vehicle_make, vehicle_model,
                  vehicle_year, vehicle_type)
                  VALUES (?,?,?,?,?,?,?,?)'''

status_insert_records = '''INSERT INTO status(first_name, last_name, age,
                  sex, retired, dependants, marital_status, salary,
                  pension, company, commute_distance, address_postcode)
                  VALUES (?,?,?,?,?,?,?,?,?,?,?,?)'''

banking_insert_records = '''INSERT INTO banking(first_name, last_name, age,
                  iban, credit_card_number, credit_card_security_code,
credit_card_start_date, credit_card_end_date,
                  address_main, address_city, address_postcode, debt)
                  VALUES (?,?,?,?,?,?,?,?,?,?,?,?)'''

#populate the tables with values from the files opened earlier
c.executemany(vehicle_insert_records, vehicle_file)
c.executemany(status_insert_records, status_file)
c.executemany(banking_insert_records, values)
```

```
#select needed columns from joins
select_all = '''SELECT b.first_name, b.last_name, b.age, s.sex, s.marital_status,
s.retired, s.dependants,
                s.salary, s.pension, s.company, s.commute_distance,
b.address_main, b.address_city,
                b.address_postcode, b.iban, b.credit_card_number,
b.credit_card_security_code,
                b.credit_card_start_date, b.credit_card_end_date, b.debt,
v.vehicle_make, v.vehicle_model,
                v.vehicle_year, v.vehicle_type
          FROM banking AS b
          LEFT JOIN vehicle AS v
          ON b.first_name = v.first_name
          AND b.last_name = v.last_name
          AND b.age = v.age
          LEFT JOIN status AS s
          ON b.first_name = s.first_name
          AND b.last_name = s.last_name
          AND b.age = s.age'''

c.execute(select_all)
#create an csvfile of the full data
with open('outputs//full_data.csv', 'w', newline='') as outfile:
  csv_writer = csv.writer(outfile, delimiter=',')
  csv_writer.writerow([i[0] for i in c.description])
  csv_writer.writerows(c)
#close the connection
c.close()


###### Next, I would create a main database using ponyorm

#create a database
db = Database()
##In order to use PonyORM with MySQL I need to install MySQLdb or pymysql
#db.bind(provider='sqlite', filename='outputs//laureltech.db', create_db=True)
##have a system database to check if file output is okay
#db.bind(provider='mysql', host='europa.ashley.work', user='student_bi24ae',
passwd='iE93F2@8EhM@1zhD&u9M@K', db='student_bi24ae') ##bind to mysql
server
```

Appendix

```
#define the table
class Main(db.Entity):
    _table_ = 'customers'
    first_name = Required(str)
    last_name = Required(str)
    age = Required(int)
    sex = Required(str)
    marital_status = Optional(str)
    retired = Optional(str)
    dependants = Optional(str)
    salary = Optional(int)
    pension = Optional(int)
    company = Optional(str)
    commute_distance = Optional(float)
    address_main = Optional(str)
    address_city = Optional(str)
    address_postcode = Optional(str)
    iban = Required(str, unique=True)
    credit_card_number = Required(str, unique=True)
    credit_card_security_code = Required(str)
    credit_card_start_date = Required(str)
    credit_card_end_date = Required(str)
    debt = Optional(str)
    vehicle_make = Optional(str)
    vehicle_model = Optional(str)
    vehicle_year = Optional(int)
    vehicle_type = Optional(str)

#generate the table
sql_debug(True)
db.generate_mapping(create_tables=True)
```

Appendix

```
#create a session to populate the database
@db_session
def populate_database_main():
  #creates a database and populates it from given csv file
  with open('outputs//full_data.csv', 'r') as populate_file:
    populate_table_reader = csv.reader(populate_file)
    next(populate_table_reader) ##skips the first row
    for row in populate_table_reader:
      Main(first_name = row[0], last_name = row[1], age = row[2],
        sex = row[3], marital_status = row[4], retired = row[5],
        dependants = row[6], salary = row[7], pension = row[8],
        company = row[9], commute_distance = row[10], address_main =
row[11],
        address_city = row[12], address_postcode = row[13], iban = row[14],
        credit_card_number = row[15], credit_card_security_code = row[16],
        credit_card_start_date = row[17], credit_card_end_date = row[18],
        debt = row[19], vehicle_make = row[20], vehicle_model = row[21],
        vehicle_year = row[22], vehicle_type = row[23])

#run the populate function
with db_session:
    populate_database_main()
```