Activity: Design Patterns I

CMPE 131-S04: Software Engineering I
San Jose Staté University
Tuesday, November 25, 2025

Name: Jan David Ella

Student ID: 017704908

Grade

Mastery: complete a majority of questions

Objectives

- Designing Factory

Instructions

- Go to: https://gist.github.com/ProfRojas
- Open file named: serializer_demo.py

Q: What does this code do? Draw a diagram of the major components

```
import json
import xml.etree.ElementTree as et

class Song:
    def __init__(self, song_id, title, artist):
        self.song_id = song_id
        self.title = title
        self.artist = artist


class SongSerializer:
    def serialize(self, song, format):
        if format == 'JSON':
            song_info = {
                'id': song.song_id,
                'title': song.title,
                'artist': song.artist
            }
            return json.dumps(song_info)
        elif format == 'XML':
            song_info = et.Element('song', attrib={'id': song.song_id})
            title = et.SubElement(song_info, 'title')
            title.text = song.title
            artist = et.SubElement(song_info, 'artist')
            artist.text = song.artist
            return et.tostring(song_info, encoding='unicode')
        else:
            raise ValueError(format)
```

This code serializes a song so that it can be stored in storage when, for example, a computer is shut down. The class Song shows the contents of the song, which includes its ID, title, and artist. The class Song Serializer takes an instance of that class along with the selected format and converts the song into the selected format.

Q2: Write out what components of the code can be simplified so that it is easier to expand the code.

The two if statements in serialize() can be separated so that the definition of the serialize() function can be easier to read, as well as easier to look for any issues or improvements and modify.

Q3: Write code using factory method

```
import json
import xml.etree.ElementTree as et
class Song:
    def __init__(self, song_id, title, artist):
        self.song_id = song_id
        self.title = title
        self.artist = artist
class SongSerializer:
    # New Method

    def _serialize_to_JSON(self, song):
        song_info = {
            'id': song.song_id,
            'title': song.title,
            'artist': song.artist
        }
        return json.dumps(song_info)

    def _serialize_to_XML(self, song):
        song_info = et.Element('song', attrib={'id': song.song_id})
```

```
        title = et.SubElement(song_info, 'title')
        title.text = song.title
        artist = et.SubElement(song_info, 'artist')
        artist.text = song.artist
        return et.tostring(song_info, encoding='unicode')

    def serialize(self, song, format):
        if format == 'JSON':
            return self._serialize_to_JSON(self, song)
        elif format == 'XML':
            return self._serialize_to_XML(self, song)
        else:
            raise ValueError(format)
```