

# Assessment

## Module 15) Python - Advance python programming

### Case Overview

You are working as an **Advanced Python Developer** at a software development firm. One of your clients, **TechRepair Hub**, is a large electronics repair chain that wants to digitize their operations. They've requested a **Python-based desktop application** called **RepairMate** that can streamline customer information, repair tracking, billing, and user access — all from a simple GUI.

This application will involve **file operations, exception handling, object-oriented programming, basic database connectivity**, and a **Tkinter-based GUI**.

Core functionality	
Customer & Device Management	<ul style="list-style-type: none"><li>• Create/update customer profiles using classes</li><li>• Link multiple devices to each customer</li><li>• Store data using file operations or local SQLite3</li></ul>
Repair Order Tracking	<ul style="list-style-type: none"><li>• Use GUI form to capture repair issues, assign technician</li><li>• Maintain repair history per customer/device</li></ul>
Billing & Invoicing	<ul style="list-style-type: none"><li>• Calculate cost of service + parts + tax</li><li>• Handle errors like missing fields, divide by zero etc.</li><li>• Save invoice to file (text or CSV)</li></ul>
User Roles	<ul style="list-style-type: none"><li>• Implement basic Admin vs Technician interface using Tkinter</li><li>• Use exception handling for access control or invalid operations</li></ul>
Regex Search (Reports)	<ul style="list-style-type: none"><li>• Implement search functionality for device model, status using re module</li><li>• Allow pattern matching (e.g., show all "Pending" repairs)</li></ul>

**Key Competencies Tested :**

**Python Concept:**

- File I/O (open, write, read)
- Exception Handling (try-except-finally, user-defined exceptions)
- Classes & Objects (with inheritance, overloading, overriding)

- SQLite3 / PyMySQL (for simple data persistence)
- Regular Expressions (search, match, modifiers)
- Tkinter GUI with widgets (forms, labels, buttons)

**Practical considerations:**

Clean folder/file storage for records

Error-free console + GUI experience

Use of super(), OOP hierarchy for repair module Proper

regex handling for search/filter features

**Reflective thinking:**

Can this app later connect to a cloud database?

Could the GUI support barcode-based device entries?

What changes would be needed to deploy this on multiple computers?