# Application Program to extract data from ESSLX990

**1. System Access**

Determine how data is accessed:

* 🟩 Via **API** → Use requests, httpx, or zeep (for SOAP)
* 🟨 Via **database** → Use pyodbc, psycopg2, or sqlalchemy
* 🟥 Via **screen scraping (terminal-based)** → Use py3270, pyte, or automate with pexpect

**2. UI Framework (Python Desktop GUI)**

Use one of:

* Tkinter – built-in and lightweight
* PyQt5/PyQt6 or PySide6 – professional-grade GUI
* customtkinter – modern look with less effort

**3. Features of the App**

* UI to accept query parameters (like customer ID, date range, etc.)
* Button to **connect** and extract data
* Show results in a **table view**
* Option to **export** to Excel or CSV

**🛠️ Example App using Tkinter + Mock Data**

Here’s a simple UI-based app that simulates fetching data:

python

CopyEdit

import tkinter as tk

from tkinter import ttk, messagebox, filedialog

import csv

def fetch\_data():

# Replace this part with actual data extraction from esslx990

data = [

{"ID": "1001", "Name": "Alice", "Amount": 5000},

{"ID": "1002", "Name": "Bob", "Amount": 6200},

{"ID": "1003", "Name": "Charlie", "Amount": 4700}

]

for row in tree.get\_children():

tree.delete(row)

for item in data:

tree.insert("", tk.END, values=(item["ID"], item["Name"], item["Amount"]))

def export\_data():

file\_path = filedialog.asksaveasfilename(defaultextension=".csv",

filetypes=[("CSV files", "\*.csv")])

if not file\_path:

return

with open(file\_path, mode='w', newline='') as file:

writer = csv.writer(file)

writer.writerow(["ID", "Name", "Amount"])

for row\_id in tree.get\_children():

writer.writerow(tree.item(row\_id)['values'])

messagebox.showinfo("Export", "Data exported successfully!")

# UI setup

root = tk.Tk()

root.title("ESSLX990 Data Extractor")

root.geometry("500x400")

frame = ttk.Frame(root)

frame.pack(pady=20)

ttk.Button(frame, text="Fetch Data", command=fetch\_data).pack(side=tk.LEFT, padx=10)

ttk.Button(frame, text="Export CSV", command=export\_data).pack(side=tk.LEFT, padx=10)

columns = ("ID", "Name", "Amount")

tree = ttk.Treeview(root, columns=columns, show='headings')

for col in columns:

tree.heading(col, text=col)

tree.pack(fill=tk.BOTH, expand=True)

root.mainloop()

**🔌 Integration with ESSLX990**

You’ll need to replace the mock fetch\_data() with actual logic:

* If **API**, use requests.get(url, params)
* If **ODBC/DB**, use pyodbc or sqlalchemy
* If **Mainframe terminal**, use py3270, Expect, or automation

**📦 Optional Features**

* Login screen (username/password)
* Export to Excel (openpyxl)
* Pagination or filtering in UI
* Scheduling automatic extraction

**📁 Would You Like?**

I can generate:

* A full working ZIP with source code
* Setup instructions
* UI sketch
* Deployment instructions (.exe or PyInstaller packaging)

Please confirm:

1. What exactly is **ESSLX990**?
2. How do you currently access data from it?
3. Do you want **just a basic app** or a **production-level app with packaging**?

Once you confirm, I’ll tailor the code precisely.

**✅ Goal**

**Build a Python GUI app** that:

* Connects to the ESSL X990 biometric device
* Extracts attendance logs
* Displays them in a table
* Allows export to CSV/Excel

**🧠 How ESSL X990 Works**

The **X990 device** typically communicates over the **TCP/IP** network using **ZKAccess/ZKSoftware protocol**.

To extract data:

* You connect using the **ZKTeco SDK**
* OR use third-party Python libraries that support the protocol
* The device generally listens on port 4370

**🛠️ Step-by-Step Implementation Plan**

**🔌 1. Communication with ESSL X990**

Use the zk Python library: [zk](https://pypi.org/project/zk/)

Install it:

bash

CopyEdit

pip install zk

Basic code to connect:

python

CopyEdit

from zk import ZK, const

conn = ZK('192.168.1.201', port=4370, timeout=5)

zk = conn.connect()

zk.disable\_device()

attendances = zk.get\_attendance()

for att in attendances:

print(att)

zk.enable\_device()

zk.disconnect()

**🎨 2. Create a Simple UI with Tkinter**

We’ll combine this with a UI that:

* Lets you **input IP address**
* Connects and **downloads data**
* Displays in a **table**
* **Exports to CSV**

**✅ Complete Working Python App (Basic UI + ESSL X990)**

python

CopyEdit

import tkinter as tk

from tkinter import ttk, messagebox, filedialog

from zk import ZK, const

import csv

def fetch\_data():

ip = ip\_entry.get()

if not ip:

messagebox.showerror("Error", "Please enter device IP")

return

try:

conn = ZK(ip, port=4370, timeout=5)

zk = conn.connect()

zk.disable\_device()

attendances = zk.get\_attendance()

# Clear old data

for row in tree.get\_children():

tree.delete(row)

for att in attendances:

tree.insert("", tk.END, values=(att.user\_id, att.timestamp, att.status))

zk.enable\_device()

zk.disconnect()

except Exception as e:

messagebox.showerror("Connection Error", str(e))

def export\_data():

file\_path = filedialog.asksaveasfilename(defaultextension=".csv",

filetypes=[("CSV files", "\*.csv")])

if not file\_path:

return

with open(file\_path, mode='w', newline='') as file:

writer = csv.writer(file)

writer.writerow(["User ID", "Timestamp", "Status"])

for row\_id in tree.get\_children():

writer.writerow(tree.item(row\_id)['values'])

messagebox.showinfo("Export", "Data exported successfully!")

# --- UI SETUP ---

root = tk.Tk()

root.title("ESSL X990 Attendance Extractor")

root.geometry("600x450")

# IP Input

frame\_top = ttk.Frame(root)

frame\_top.pack(pady=10)

ttk.Label(frame\_top, text="Device IP:").pack(side=tk.LEFT, padx=5)

ip\_entry = ttk.Entry(frame\_top)

ip\_entry.pack(side=tk.LEFT, padx=5)

ip\_entry.insert(0, "192.168.1.201") # default IP

ttk.Button(frame\_top, text="Fetch Attendance", command=fetch\_data).pack(side=tk.LEFT, padx=5)

ttk.Button(frame\_top, text="Export CSV", command=export\_data).pack(side=tk.LEFT, padx=5)

# Table

columns = ("User ID", "Timestamp", "Status")

tree = ttk.Treeview(root, columns=columns, show='headings')

for col in columns:

tree.heading(col, text=col)

tree.pack(fill=tk.BOTH, expand=True, padx=10, pady=10)

root.mainloop()

**📦 How to Run**

1. Make sure your PC is on the same LAN as the device
2. Set IP (e.g. 192.168.1.201) from device settings
3. Install required Python packages:

bash

CopyEdit

pip install zk

1. Run the script:

bash

CopyEdit

python essl\_gui\_app.py

**📁 Want a ZIP?**

Would you like me to:

* Bundle this into a full working project (code + requirements.txt)?
* Package it as a .exe using PyInstaller for Windows?

**🔒 Future Enhancements (Optional)**

* **Login authentication**
* **Database integration** (e.g., store logs in SQLite/PostgreSQL)
* **Filter by date/user**
* **Scheduled auto-fetch and sync**
* Export to **Excel** (openpyxl)