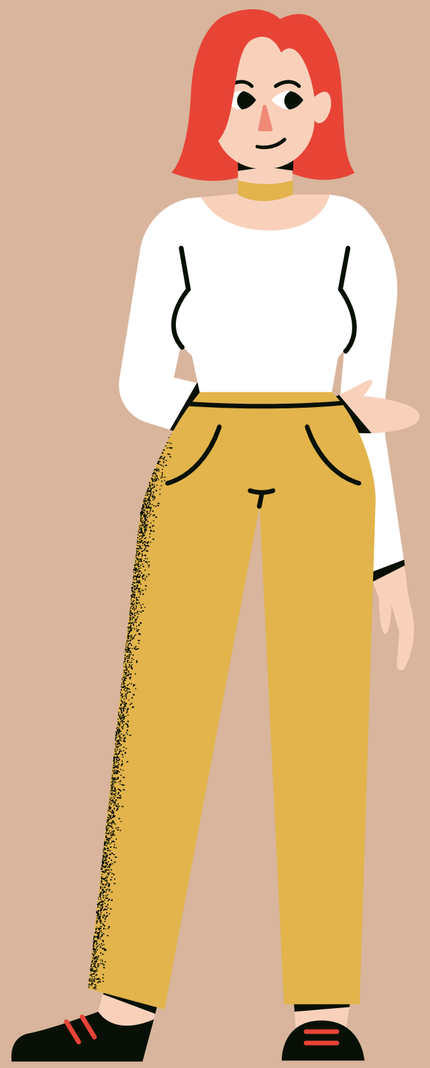
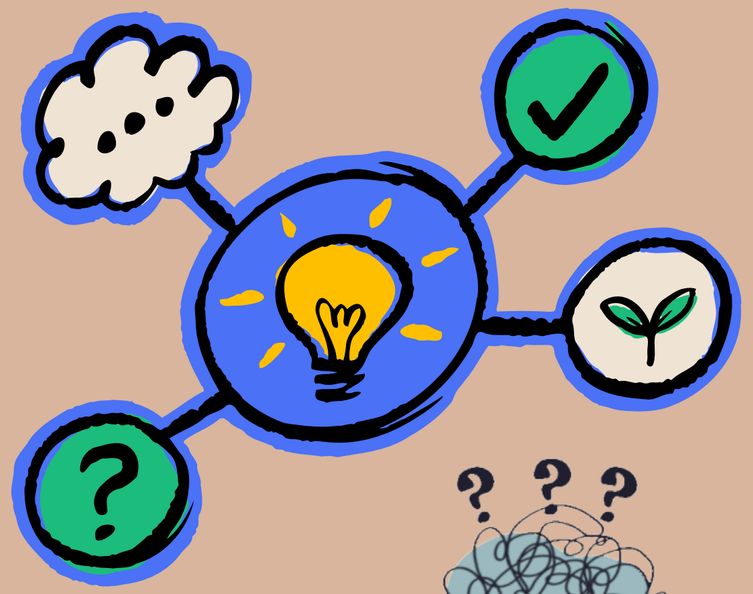
Advanced Programming Project

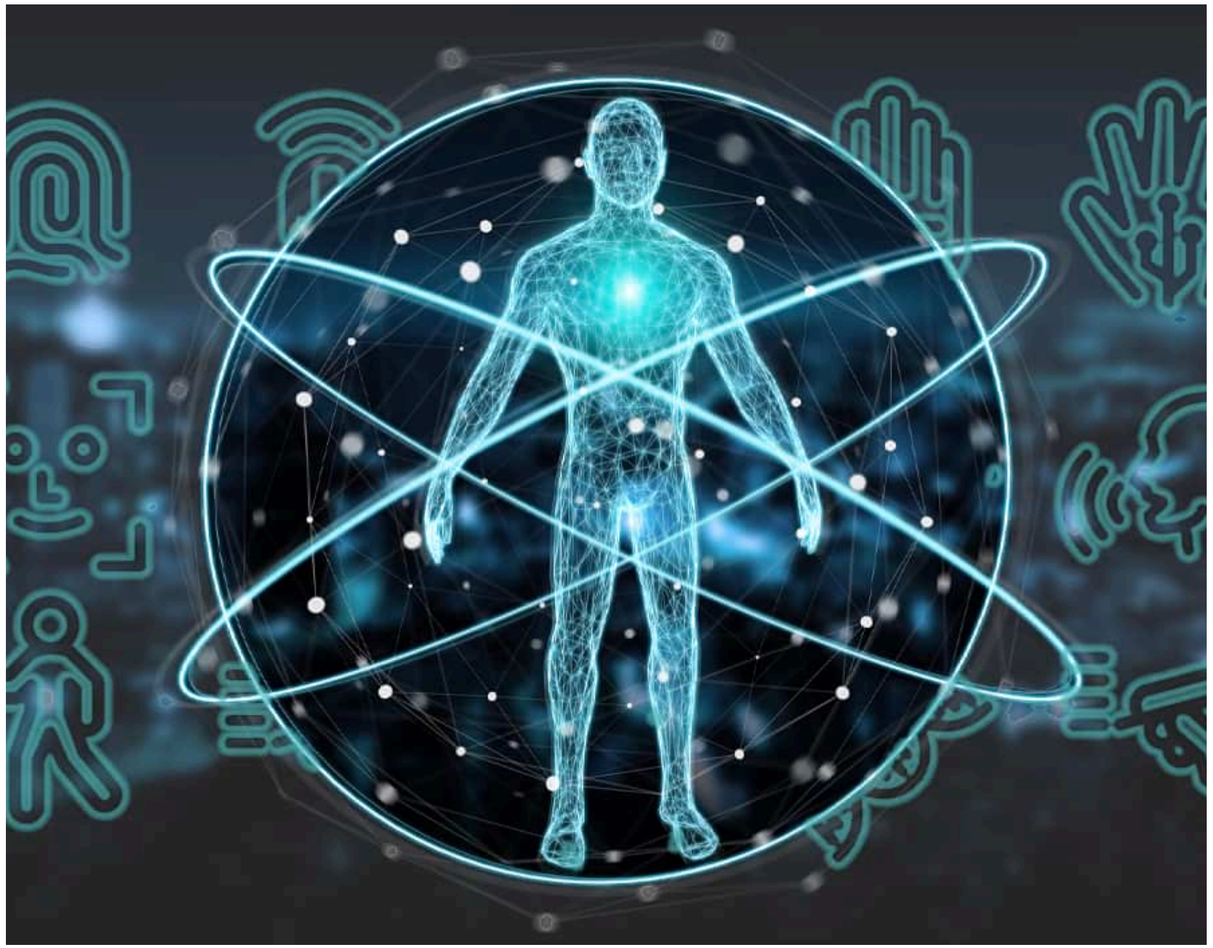
Yachay Tech University

Presented by: José Adrián Quel

Date: 3rd June 2025





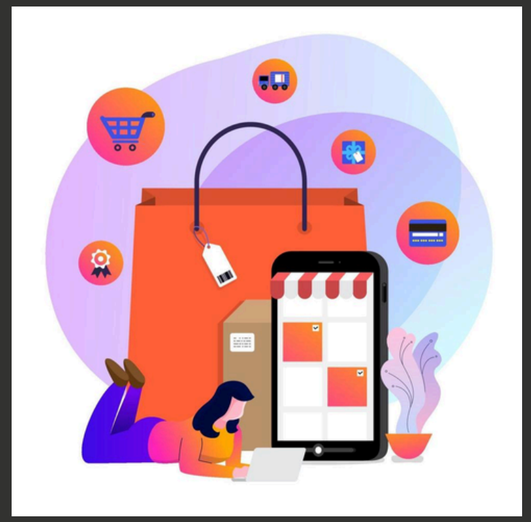


Biometric

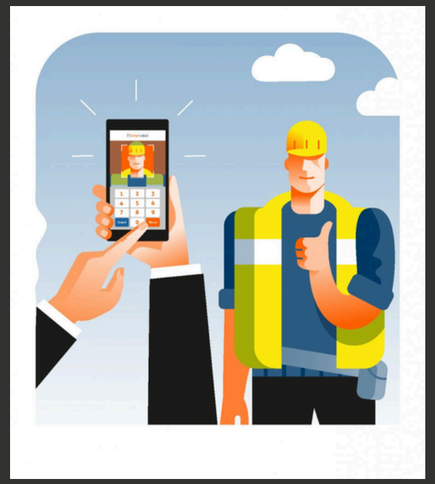
Matching System

for Everyday Applications

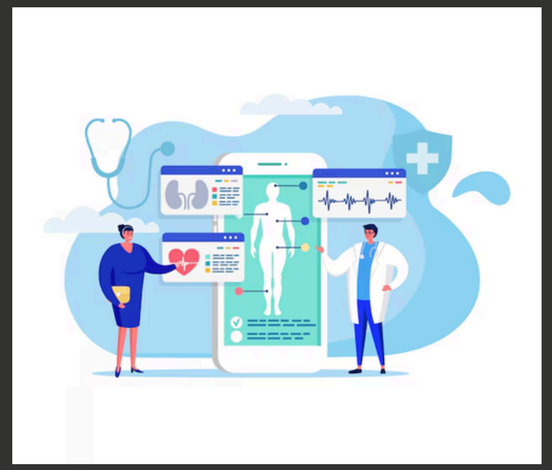
Problem and Proposed solution



Online Sales



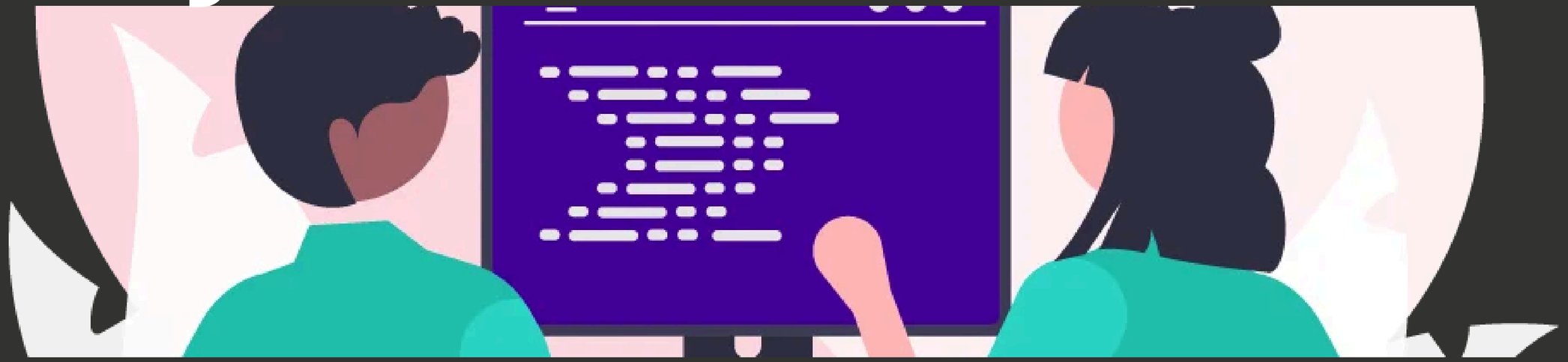
Security



Health

Program in C++

Objectives



Develop a program

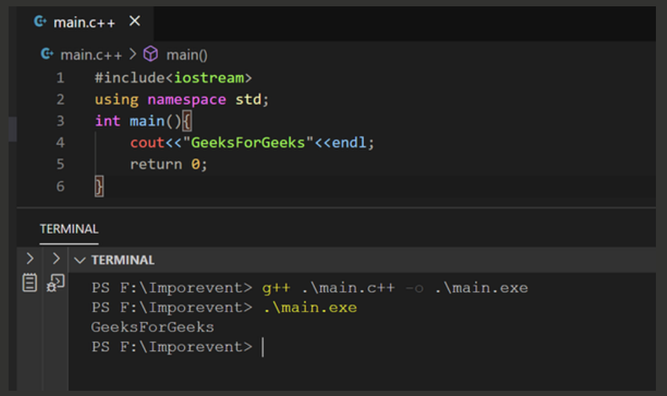
Create a standardized data structure

Implement a comparison algorithm

Ensure ease of use and maintenance

Materials

- **Programming Language**
- **Compiler**
- **Code Editor**
- **Input Files**
- **Terminal**

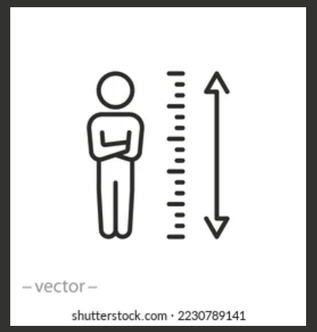


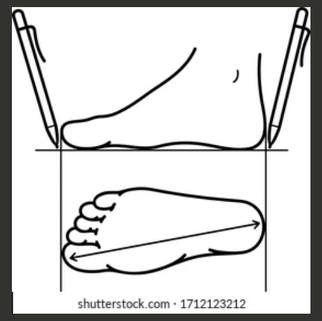
Methodos

1. Structs

```
struct BiometricData {
    std::string Bio_ID;
    double Bio_height;
    double Bio_weight;
    double Bio_handLength;
    double Bio_footSize;
    double Bio_faceRatio;

    static BiometricData fromString(const std::string& line) {
        BiometricData data;
        std::stringstream ss(line);
        std::string token;
        std::getline(ss, data.Bio_ID, ',');
        std::getline(ss, token, ','); data.Bio_height = std::stod(token);
        std::getline(ss, token, ','); data.Bio_weight = std::stod(token);
        std::getline(ss, token, ','); data.Bio_handLength = std::stod(token);
        std::getline(ss, token, ','); data.Bio_footSize = std::stod(token);
        std::getline(ss, token, ','); data.Bio_faceRatio = std::stod(token);
        return data;
    }
};
```





```
for (int i = 1; i < argc; ++i) {
    std::ifstream file(argv[i]);
    if (!file.is_open()) {
        std::cerr << "Error: Cannot open " << argv[i] << "\n";
        return 1;
    }
    std::string line;
    if (std::getline(file, line) && !line.empty()) {
        dataList.push_back(BiometricData::fromString(line));
    } else {
        std::cerr << "Error: File " << argv[i] << " is empty or invalid\n";
        return 1;
    }
    file.close();
}
```

2. Reading of .txt files

3. Comparing data files

```
bool isMatch(const BiometricData& a, const BiometricData& b) {
    double tolerance = 0.05;
    return (
        std::abs(a.Bio_height - b.Bio_height) <= tolerance &&
        std::abs(a.Bio_weight - b.Bio_weight) <= tolerance * 100 &&
        std::abs(a.Bio_handLength - b.Bio_handLength) <= tolerance * 100 &&
        std::abs(a.Bio_footSize - b.Bio_footSize) <= tolerance * 100 &&
        std::abs(a.Bio_faceRatio - b.Bio_faceRatio) <= tolerance
    );
```

4. The main function

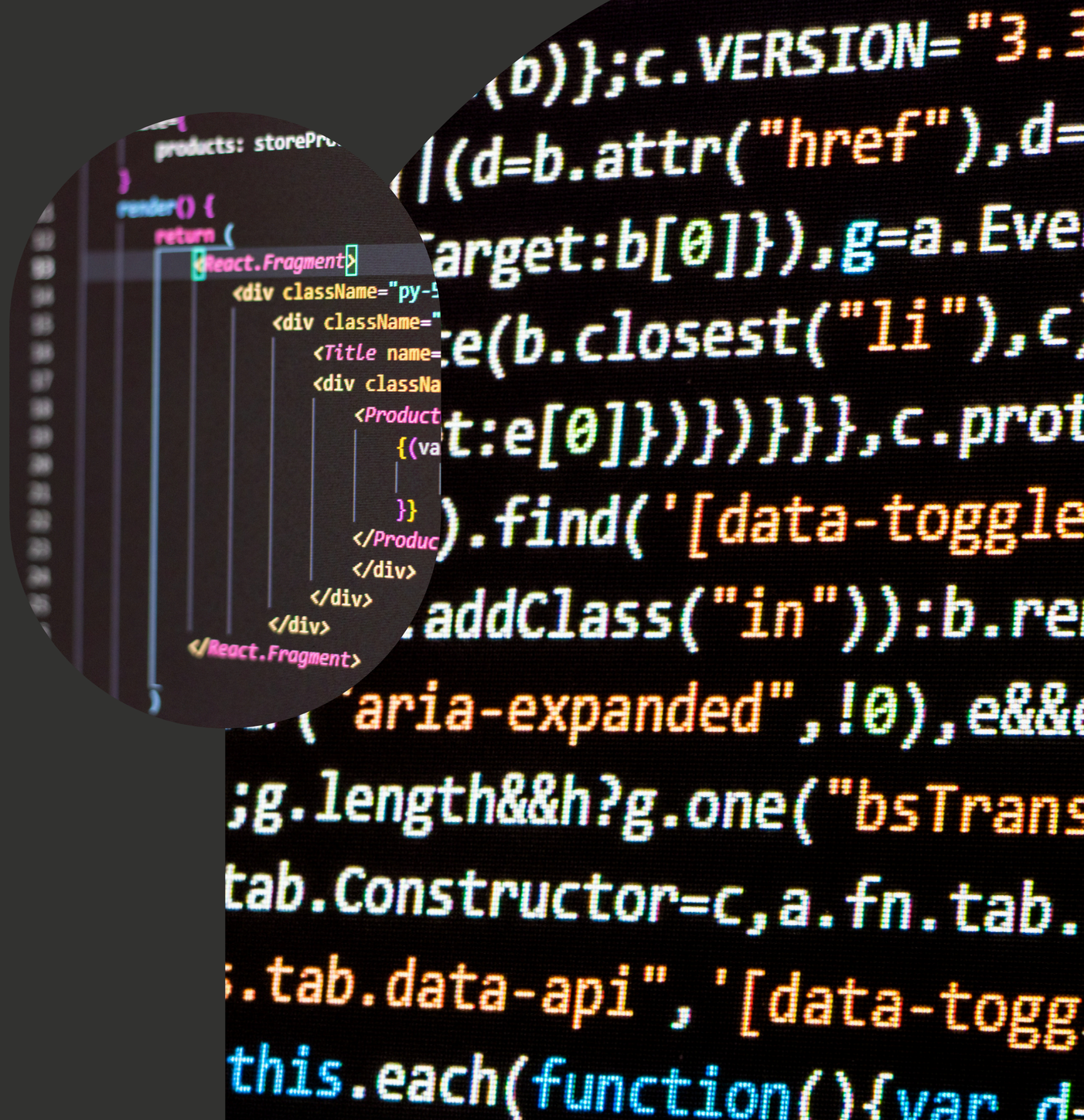
```
int main(int argc, char* argv[]) {
    if (argc < 2) {
        std::cerr << "Usage: BioData.exe <file1.txt> <file2.txt> ...\n";
        return 1;
    }

    std::vector<BiometricData> dataList;

    // Read biometric data from each file
    for (int i = 1; i < argc; ++i) {
        std::ifstream file(argv[i]);
        if (!file.is_open()) {
            std::cerr << "Error: Cannot open " << argv[i] << "\n";
            return 1;
        }
        std::string line;
        if (std::getline(file, line) && !line.empty()) {
            dataList.push_back(BiometricData::fromString(line));
        } else {
            std::cerr << "Error: File " << argv[i] << " is empty or invalid\n";
            return 1;
        }
        file.close();
    }
```

Input

Output

Match
or no
Match

Thank you for your time!