

IMPLEMENTASI KONSEP PEMROGRAMAN BERORIENTASI OBJEK DALAM PADA PENGGUNAAN SISTEM INVENTARIS BARANG DENGAN C++

Fadilla¹

Email: 25dilla@gmail.com

Marissa Dealova Marbun²

Email: marissadealovamarbun@gmail.com

Rifki Ferdiansyah³

Email: ikyferdi49@gmail.com

Patra Adypratama⁴

Email: patra191219@gmail.com

^{1,2,3,4}Universitas Negeri Medan

ABSTRAK

Pemrograman Berorientasi Objek (PBO) adalah paradigma pemrograman yang menggunakan konsep objek untuk menyusun kode secara modular, meningkatkan fleksibilitas, dan mempercepat pengembangan perangkat lunak. Dalam penelitian ini, sistem inventaris barang dikembangkan menggunakan bahasa pemrograman C++ dengan konsep PBO yang mencakup enkapsulasi, pewarisan, dan polimorfisme. Sistem ini dirancang untuk membantu pengelolaan stok barang dalam suatu gudang dengan fitur pencatatan, pencarian, dan pelaporan data barang secara efisien. Hasil pengujian menunjukkan bahwa sistem yang dibangun dapat bekerja dengan baik dalam mengelola data inventaris secara otomatis dan meminimalkan kesalahan manusia dalam pencatatan barang.

Kata Kunci: Pemrograman Berorientasi Objek, C++, Sistem Inventaris, Perangkat Lunak.

ABSTRACT

Object-Oriented Programming (OOP) is a programming paradigm that utilizes object concepts to structure code modularly, enhance flexibility, and accelerate software development. In this study, an inventory system was developed using the C++ programming language with OOP concepts, including encapsulation, inheritance, and

polymorphism. The system is designed to assist in stock management within a warehouse by providing features for recording, searching, and reporting inventory data efficiently. The test results show that the developed system functions effectively in managing inventory data automatically and minimizing human errors in record-keeping.

Keywords: *Object-Oriented Programming, C++, Inventory System, Software.*

1. PENDAHULUAN

Dalam era digital saat ini, sistem pengelolaan data menjadi semakin penting dalam berbagai aspek kehidupan, termasuk dalam dunia bisnis dan industri. Salah satu sistem yang krusial adalah sistem inventaris barang, yang digunakan untuk mencatat, mengelola, dan melacak stok barang dalam suatu gudang atau perusahaan. Penggunaan metode manual dalam pengelolaan inventaris sering kali menyebabkan berbagai kendala, seperti human error dalam pencatatan data, kehilangan informasi, serta kesulitan dalam pencarian barang tertentu. Oleh karena itu, diperlukan sistem inventaris yang dapat bekerja secara otomatis untuk meningkatkan efisiensi serta akurasi dalam manajemen stok barang.

Pemrograman Berorientasi Objek (PBO) menawarkan pendekatan yang modular dan terstruktur dalam pengembangan perangkat lunak, yang sangat cocok untuk membangun sistem inventaris barang. Dengan menggunakan bahasa pemrograman C++, konsep PBO seperti enkapsulasi, pewarisan, dan polimorfisme dapat diterapkan untuk menciptakan sistem yang lebih fleksibel dan mudah dikembangkan. Sistem ini memungkinkan pencatatan barang secara terstruktur, pencarian data yang cepat, serta pembuatan laporan yang lebih akurat. Penelitian ini bertujuan untuk mengembangkan dan menguji sistem inventaris berbasis PBO dengan C++ guna memastikan kehandalan dan efektivitasnya dalam pengelolaan inventaris.

Selain itu, implementasi sistem berbasis PBO dalam C++ juga memungkinkan peningkatan skalabilitas dan keamanan dalam pengelolaan data inventaris. Dengan adanya fitur enkapsulasi, akses ke data barang dapat dikontrol dengan lebih baik sehingga mengurangi risiko manipulasi yang tidak sah. Sistem ini juga dapat dikembangkan lebih lanjut dengan integrasi ke dalam basis data atau layanan cloud, sehingga memungkinkan pemantauan stok barang secara real-time dan akses yang lebih fleksibel dari berbagai lokasi. Dengan demikian, sistem ini diharapkan dapat menjadi solusi yang efektif dalam

manajemen inventaris bagi berbagai jenis usaha dan organisasi.

Sistem inventaris berbasis komputer memiliki keunggulan dibandingkan pencatatan manual dalam aspek kecepatan dan ketepatan. Dengan bantuan teknologi, perusahaan dapat mengurangi kesalahan pencatatan yang sering terjadi pada sistem manual. Selain itu, sistem ini dapat memudahkan analisis data dengan menyediakan laporan yang dapat diakses dengan cepat dan akurat. Implementasi teknologi dalam sistem inventaris juga dapat meningkatkan efisiensi operasional serta mempercepat proses pengambilan keputusan.

C++ dipilih sebagai bahasa pemrograman dalam penelitian ini karena memiliki keunggulan dalam hal performa dan fleksibilitas. Sebagai bahasa yang mendukung pemrograman berorientasi objek, C++ memungkinkan pengembang untuk membuat sistem yang modular dan mudah untuk diperluas. Selain itu, bahasa ini memiliki pustaka standar yang kaya serta dukungan komunitas yang luas, yang dapat membantu dalam pengembangan sistem inventaris yang lebih canggih dan efisien.

Penerapan konsep pewarisan dalam sistem ini memungkinkan pengelolaan berbagai jenis barang dengan lebih terstruktur. Misalnya, dalam sistem inventaris, barang dapat dikategorikan menjadi barang elektronik dan non-elektronik. Dengan pewarisan, kelas dasar dapat menyimpan atribut umum, sedangkan subclass dapat menambahkan atribut spesifik sesuai dengan jenis barang. Hal ini memudahkan pengelolaan data serta memungkinkan sistem untuk dikembangkan lebih lanjut dengan fitur tambahan di masa mendatang.

Penelitian ini akan menguji efektivitas sistem inventaris berbasis PBO dengan berbagai skenario pengujian. Pengujian akan mencakup pencatatan barang, pencarian barang, serta pembaruan stok dalam sistem. Hasil pengujian diharapkan dapat menunjukkan bahwa sistem yang dikembangkan dapat memberikan kemudahan dalam pengelolaan inventaris dan meningkatkan efisiensi dalam pencatatan serta pelaporan data barang.

2. METODE PENELITIAN

Penelitian ini menggunakan metode rekayasa perangkat lunak yang terdiri dari beberapa tahapan utama. Tahap pertama adalah analisis kebutuhan, di mana dilakukan identifikasi terhadap kebutuhan sistem inventaris, termasuk fitur pencatatan, pencarian, dan pelaporan barang. Setelah itu, tahap perancangan sistem dilakukan dengan

menggunakan diagram UML seperti Use Case Diagram dan Class Diagram untuk memodelkan struktur dan fungsionalitas sistem secara lebih jelas.

Setelah tahap perancangan selesai, sistem diimplementasikan menggunakan bahasa pemrograman C++ dengan menerapkan prinsip PBO, termasuk enkapsulasi, pewarisan, dan polimorfisme untuk meningkatkan modularitas dan fleksibilitas sistem. Pengujian sistem dilakukan menggunakan metode black-box testing guna memastikan bahwa setiap fitur sistem bekerja sesuai dengan spesifikasi yang telah dirancang. Evaluasi hasil pengujian digunakan untuk mengukur efektivitas dan efisiensi sistem dalam mengelola inventaris barang serta untuk mengidentifikasi kemungkinan perbaikan atau pengembangan lebih lanjut.

3. HASIL DAN PEMBAHASAN

Hasil implementasi sistem inventaris berbasis PBO menunjukkan bahwa sistem ini mampu mengelola stok barang secara efektif dan efisien. Dengan menerapkan konsep enkapsulasi, pewarisan, dan polimorfisme dalam pengembangannya, sistem ini dapat menyimpan data barang dalam struktur yang terorganisir, memungkinkan pencarian barang secara cepat, serta memberikan kemudahan dalam pembaruan data inventaris. Uji coba menunjukkan bahwa sistem dapat mencatat barang yang masuk dan keluar dengan akurat serta menghasilkan laporan inventaris yang dapat digunakan untuk analisis lebih lanjut.

Selain itu, pengujian sistem dengan metode black-box testing menunjukkan bahwa semua fitur yang dikembangkan berfungsi sesuai dengan spesifikasi yang telah dirancang. Pengguna dapat dengan mudah menambahkan, mengedit, mencari, dan menghapus data barang sesuai kebutuhan. Sistem juga menunjukkan stabilitas dalam menangani jumlah data yang besar tanpa mengalami penurunan performa yang signifikan. Dengan demikian, sistem ini dinilai layak untuk digunakan dalam manajemen inventaris pada berbagai sektor industri.

a. Use Case Diagram

Use Case Diagram merupakan salah satu jenis diagram UML yang digunakan untuk menggambarkan interaksi antara pengguna dengan sistem. Dalam sistem inventaris barang berbasis PBO ini, terdapat beberapa aktor utama, yaitu administrator dan pengguna. Administrator memiliki hak akses penuh untuk menambahkan, menghapus, memperbarui, dan melihat data barang, sedangkan pengguna hanya dapat melakukan

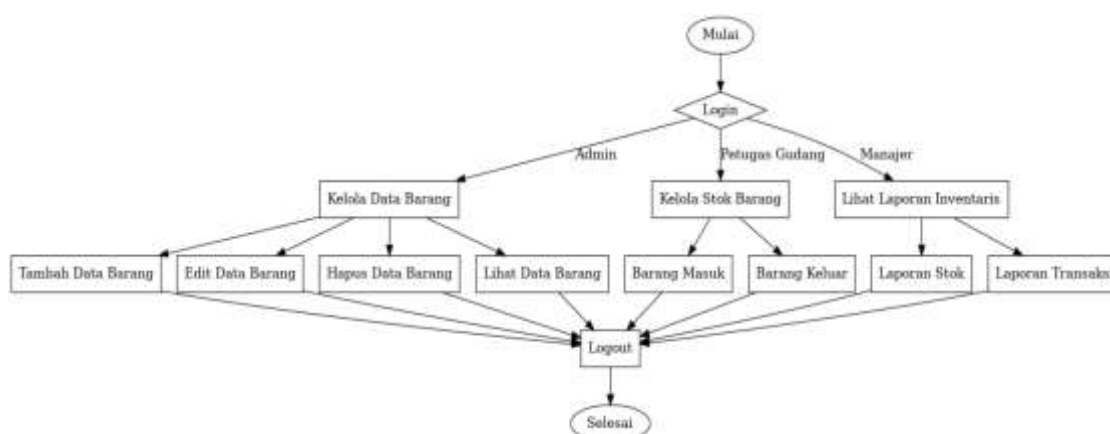
pencarian barang dan melihat laporan inventaris. Diagram ini juga mencakup berbagai use case utama seperti manajemen data barang, pencarian stok, serta pembuatan laporan. Dengan adanya use case diagram, pengembang dapat memahami hubungan antar-entitas dalam sistem dan memastikan bahwa semua kebutuhan pengguna telah diakomodasi dengan baik.



Gambar 1. Use Case Diagram Hasil Inventaris Barang

b. Activity Diagram

Activity Diagram merupakan salah satu jenis diagram UML yang digunakan untuk memodelkan alur kerja dalam sistem. Pada sistem inventaris barang berbasis PBO ini, activity diagram menggambarkan urutan aktivitas mulai dari pengguna masuk ke dalam sistem, melakukan pencatatan barang, pencarian stok, hingga pembuatan laporan inventaris. Diagram ini membantu menggambarkan bagaimana setiap proses berjalan dan bagaimana pengguna berinteraksi dengan sistem secara keseluruhan. Dengan adanya activity diagram, pengembang dapat memahami bagaimana aliran data dan proses terjadi dalam sistem, sehingga memudahkan identifikasi potensi peningkatan efisiensi serta pengembangan fitur tambahan.

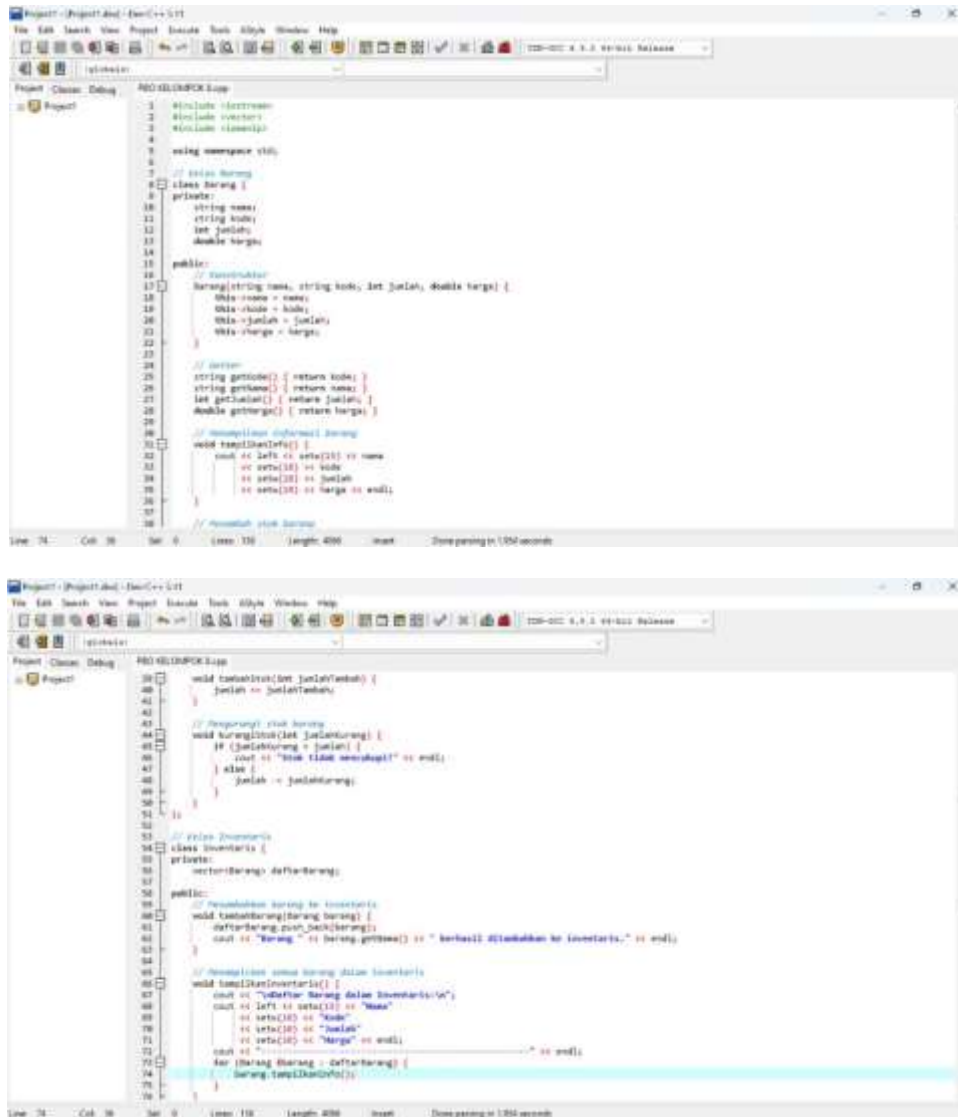


Gambar 2. Activity Diagram Hasil Inventaris Barang

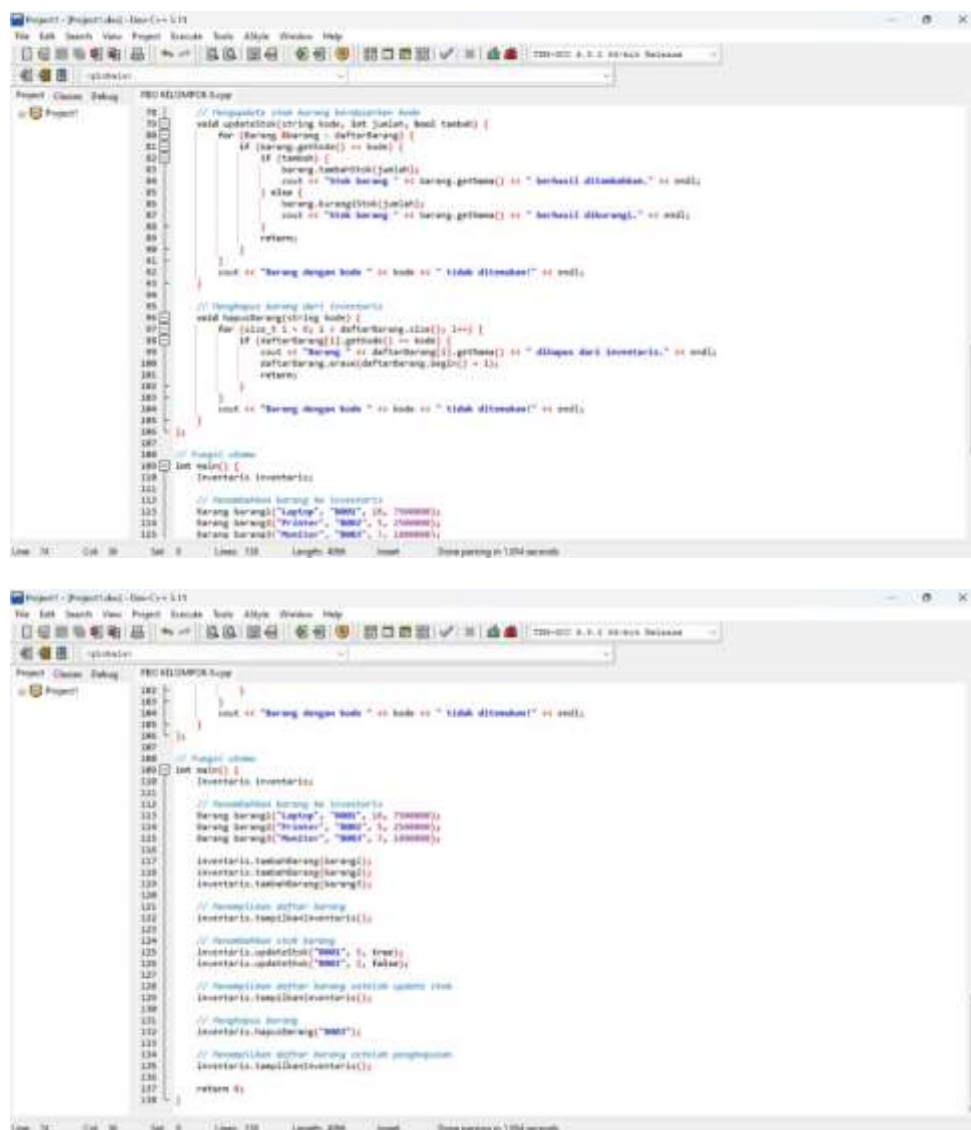
c. Jalannya program dan kode codingan pemrograman C++

Sistem inventaris barang berbasis PBO ini dirancang dengan antarmuka berbasis

teks yang memungkinkan pengguna untuk menambahkan, menghapus, mencari, dan menampilkan data barang dengan mudah. Setelah pengguna masuk ke dalam sistem, mereka dapat memilih opsi dari menu utama untuk mengelola inventaris. Berikut adalah contoh kode implementasi sistem dalam C++:



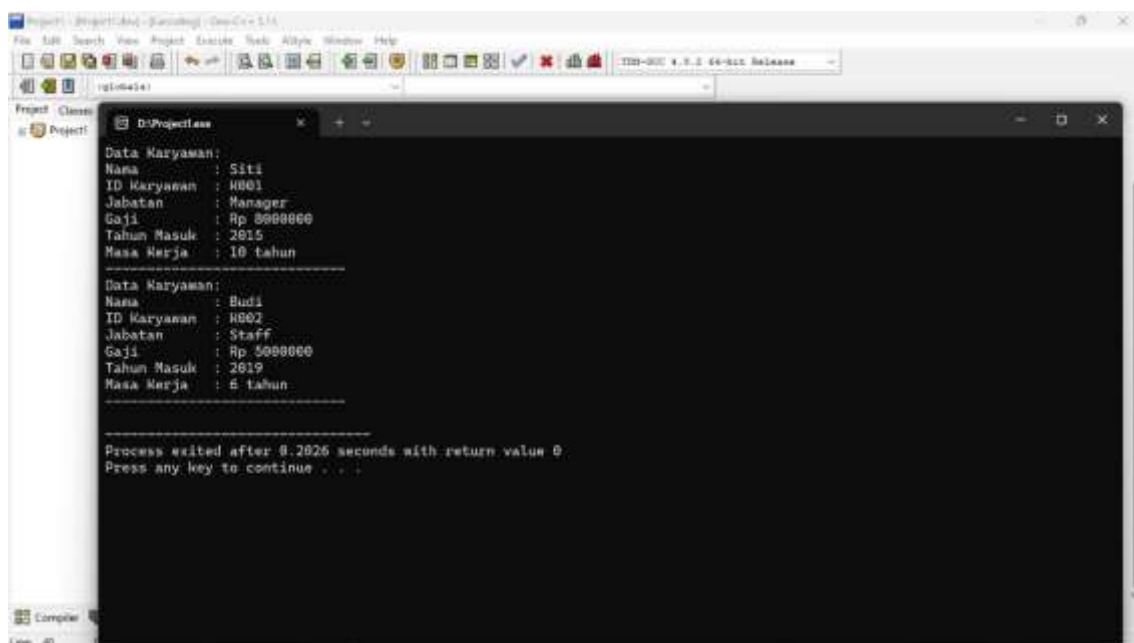
```
1 #include <iostream>
2 #include <vector>
3 #include <string>
4
5 using namespace std;
6
7 // class barang
8 class barang {
9 private:
10     string nama;
11     string kode;
12     int jumlah;
13     double harga;
14
15 public:
16     // constructor
17     barang(string nama, string kode, int jumlah, double harga) {
18         this->nama = nama;
19         this->kode = kode;
20         this->jumlah = jumlah;
21         this->harga = harga;
22     }
23
24     // getter
25     string getnama() { return nama; }
26     string getkode() { return kode; }
27     int getjumlah() { return jumlah; }
28     double getharga() { return harga; }
29
30     // menampilkan data barang
31     void tampilInfo() {
32         cout << left << setw(25) << nama
33             << endl;
34         cout << left << setw(25) << kode
35             << endl;
36         cout << left << setw(25) << jumlah
37             << endl;
38         cout << left << setw(25) << harga << endl;
39     }
40
41     // destructor
42     ~barang() {}
43 };
44
45 // class inventaris
46 class inventaris {
47 private:
48     vector<barang> daftarBarang;
49
50 public:
51     // menambahkan barang ke inventaris
52     void tambahBarang(barang barang) {
53         daftarBarang.push_back(barang);
54         cout << "Barang " << barang.getnama() << " berhasil ditambahkan ke inventaris." << endl;
55     }
56
57     // menampilkan semua barang dalam inventaris
58     void tampilInfoInventaris() {
59         cout << "Daftar Barang dalam Inventaris:" << endl;
60         cout << left << setw(25) << "Nama"
61             << endl;
62         cout << left << setw(25) << "Kode"
63             << endl;
64         cout << left << setw(25) << "Jumlah"
65             << endl;
66         cout << left << setw(25) << "Harga" << endl;
67         cout << "-----" << endl;
68         for (barang barang : daftarBarang) {
69             barang.tampilInfo();
70         }
71     }
72 };
73
74 // main
75 int main() {
76     inventaris inv;
77     barang b1("Beras", "B1", 10, 15000);
78     barang b2("Gandum", "G2", 5, 20000);
79     inv.tambahBarang(b1);
80     inv.tambahBarang(b2);
81     inv.tampilInfoInventaris();
82     return 0;
83 }
```



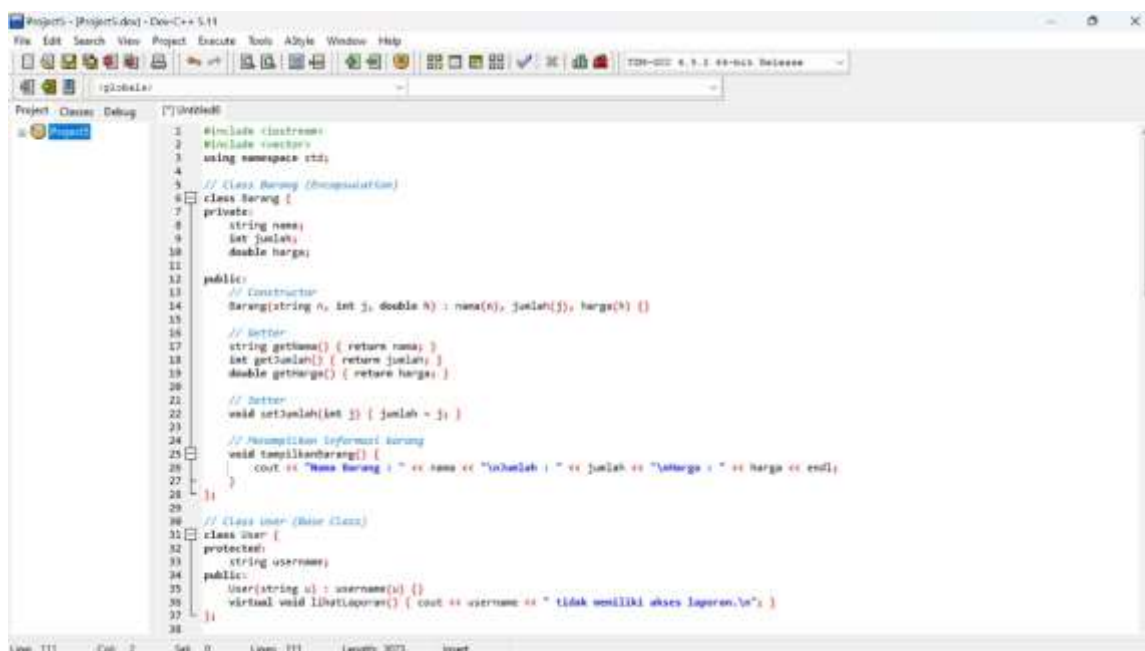
Gambar 3. Kode codingan

Lalu selanjutnya yaitu melihat hasil dari kode codingan inventaris barang dengan bahasa pemrograman C++ ini.

Pada tampilan ini kami menginput data dari hasil kode codingan tersebut tentang inventaris barang, data ini berupa Data karyawan, Nama, ID karyawan, Jabatan, Gaji, Tahun masuk, Masa kerja. Lalu Langkah selanjutnya yaitu menerima inputan data dengan cara melihat hasil kode codingan tersebut.

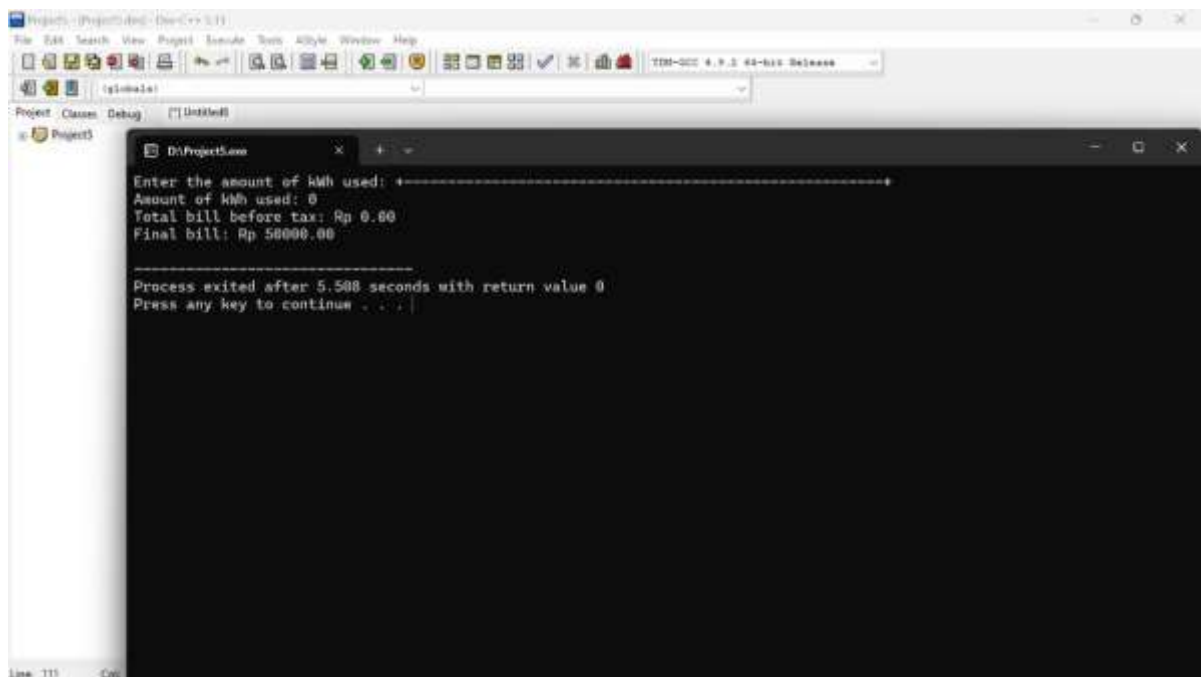


Gambar 4. Hasil Input Kode Data



Gambar 5. Kode kodingan laporan keuangan barang inventaris

Dalam menggunakan bahasa pemrograman C++ ini sangatlah moda dalam mengganti/menghapus hasil data yang tidak kita inginkan atau tidak di perlukan, bahasa pemrograman ini juga untuk bisa mempermudah karyawan dalam hasil laporan barang-barang, dan juga bahasa pemrograman ini sangatlah mengembangkan kemajuan data hasil yang ada, jika ingin menghapus data yang tidak di inginkan contoh hasil penghapusan di data hasil penginputan kode kodingan yang ada di C++



Gambar 6. Hasil Input Data Laporan Keuangan

d. Kode Kodingan Hasil Penggunaan Sistem Inventaris Barang Dengan C++

1. `#include <iostream>`
2. `#include <vector>`
3. `#include <iomanip>`
- 4.
5. `using namespace std;`
- 6.
7. `// Kelas Barang`
8. `class Barang {`
9. `private:`
10. `string nama;`
11. `string kode;`
12. `int jumlah;`
13. `double harga;`
- 14.
15. `public:`
16. `// Konstruktor`
17. `Barang(string nama, string kode, int jumlah, double harga) {`
18. `this->nama = nama;`

```
19.     this->kode = kode;
20.     this->jumlah = jumlah;
21.     this->harga = harga;
22. }
23.
24. // Getter
25. string getKode() { return kode; }
26. string getNama() { return nama; }
27. int getJumlah() { return jumlah; }
28. double getHarga() { return harga; }
29.
30. // Menampilkan informasi barang
31. void tampilkanInfo() {
32.     cout << left << setw(15) << nama
33.         << setw(10) << kode
34.         << setw(10) << jumlah
35.         << setw(10) << harga << endl;
36. }
37.
38. // Menambah stok barang
39. void tambahStok(int jumlahTambah) {
40.     jumlah += jumlahTambah;
41. }
42.
43. // Mengurangi stok barang
44. void kurangiStok(int jumlahKurang) {
45.     if (jumlahKurang > jumlah) {
46.         cout << "Stok tidak mencukupi!" << endl;
47.     } else {
48.         jumlah -= jumlahKurang;
49.     }
50. }
```

```
51. };
52.
53. // Kelas Inventaris
54. class Inventaris {
55. private:
56.     vector<Barang> daftarBarang;
57.
58. public:
59.     // Menambahkan barang ke inventaris
60.     void tambahBarang(Barang barang) {
61.         daftarBarang.push_back(barang);
62.         cout << "Barang " << barang.getNama() << " berhasil ditambahkan ke
            inventaris." << endl;
63.     }
64.
65.     // Menampilkan semua barang dalam inventaris
66.     void tampilkanInventaris() {
67.         cout << "\nDaftar Barang dalam Inventaris:\n";
68.         cout << left << setw(15) << "Nama"
69.             << setw(10) << "Kode"
70.             << setw(10) << "Jumlah"
71.             << setw(10) << "Harga" << endl;
72.         cout << "-----" << endl;
73.         for (Barang &barang : daftarBarang) {
74.             barang.tampilkanInfo();
75.         }
76.     }
77.
78.     // Mengupdate stok barang berdasarkan kode
79.     void updateStok(string kode, int jumlah, bool tambah) {
80.         for (Barang &barang : daftarBarang) {
81.             if (barang.getKode() == kode) {
```

```
82.         if (tambah) {
83.             barang.tambahStok(jumlah);
84.             cout << "Stok barang " << barang.getNama() << " berhasil
                ditambahkan." << endl;
85.         } else {
86.             barang.kurangiStok(jumlah);
87.             cout << "Stok barang " << barang.getNama() << " berhasil
                dikurangi." << endl;
88.         }
89.         return;
90.     }
91. }
92.     cout << "Barang dengan kode " << kode << " tidak ditemukan!" << endl;
93. }
94.
95. // Menghapus barang dari inventaris
96. void hapusBarang(string kode) {
97.     for (size_t i = 0; i < daftarBarang.size(); i++) {
98.         if (daftarBarang[i].getKode() == kode) {
99.             cout << "Barang " << daftarBarang[i].getNama() << " dihapus dari
                inventaris." << endl;
100.                daftarBarang.erase(daftarBarang.begin() + i);
101.                return;
102.            }
103.        }
104.        cout << "Barang dengan kode " << kode << " tidak ditemukan!" <<
            endl;
105.    }
106. };
107.
108. // Fungsi utama
109. int main() {
```

```
110.      Inventaris inventaris;
111.
112.      // Menambahkan barang ke inventaris
113.      Barang barang1("Laptop", "B001", 10, 7500000);
114.      Barang barang2("Printer", "B002", 5, 2500000);
115.      Barang barang3("Monitor", "B003", 7, 1800000);
116.
117.      inventaris.tambahBarang(barang1);
118.      inventaris.tambahBarang(barang2);
119.      inventaris.tambahBarang(barang3);
120.
121.      // Menampilkan daftar barang
122.      inventaris.tampilkanInventaris();
123.
124.      // Menambahkan stok barang
125.      inventaris.updateStok("B001", 5, true);
126.      inventaris.updateStok("B002", 2, false);
127.
128.      // Menampilkan daftar barang setelah update stok
129.      inventaris.tampilkanInventaris();
130.
131.      // Menghapus barang
132.      inventaris.hapusBarang("B003");
133.
134.      // Menampilkan daftar barang setelah penghapusan
135.      inventaris.tampilkanInventaris();
136.
137.      return 0;
138.  }
```

e. Kode Kodingan Laporan Keuangan Barang

1. #include <iostream>
2. #include <vector>

```
3. using namespace std;
4.
5. // Class Barang (Encapsulation)
6. class Barang {
7. private:
8.     string nama;
9.     int jumlah;
10.    double harga;
11.
12. public:
13.    // Constructor
14.    Barang(string n, int j, double h) : nama(n), jumlah(j), harga(h) {}
15.
16.    // Getter
17.    string getNama() { return nama; }
18.    int getJumlah() { return jumlah; }
19.    double getHarga() { return harga; }
20.
21.    // Setter
22.    void setJumlah(int j) { jumlah = j; }
23.
24.    // Menampilkan informasi barang
25.    void tampilkanBarang() {
26.        cout << "Nama Barang : " << nama << "\nJumlah : " << jumlah << "\nHarga
        : " << harga << endl;
27.    }
28. };
29.
30. // Class User (Base Class)
31. class User {
32. protected:
33.    string username;
```

```
34. public:
35.     User(string u) : username(u) {}
36.     virtual void lihatLaporan() { cout << username << " tidak memiliki akses
        laporan.\n"; }
37. };
38.
39. // Class Admin (Turunan dari User)
40. class Admin : public User {
41. public:
42.     Admin(string u) : User(u) {}
43.
44.     void tambahBarang(vector<Barang> &inventory, Barang b) {
45.         inventory.push_back(b);
46.         cout << "Barang berhasil ditambahkan oleh " << username << endl;
47.     }
48.
49.     void hapusBarang(vector<Barang> &inventory, string nama) {
50.         for (size_t i = 0; i < inventory.size(); i++) {
51.             if (inventory[i].getNama() == nama) {
52.                 inventory.erase(inventory.begin() + i);
53.                 cout << "Barang " << nama << " berhasil dihapus oleh " << username
                    << endl;
54.                 return;
55.             }
56.         }
57.         cout << "Barang tidak ditemukan.\n";
58.     }
59. };
60.
61. // Class PetugasGudang (Turunan dari User)
62. class PetugasGudang : public User {
63. public:
```

```
64.  PetugasGudang(string u) : User(u) {}
65.
66.  void updateStok(vector<Barang> &inventory, string nama, int jumlah) {
67.      for (auto &b : inventory) {
68.          if (b.getNama() == nama) {
69.              b.setJumlah(jumlah);
70.              cout << "Stok barang " << nama << " diperbarui oleh " << username <<
endl;
71.              return;
72.          }
73.      }
74.      cout << "Barang tidak ditemukan.\n";
75.  }
76. };
77.
78. // Class Manajer (Turunan dari User) - Override lihatLaporan()
79. class Manajer : public User {
80. public:
81.     Manajer(string u) : User(u) {}
82.
83.     void lihatLaporan(vector<Barang> &inventory) override {
84.         cout << "==== Laporan Inventaris =====\n";
85.         for (auto &b : inventory) {
86.             b.tampilkanBarang();
87.             cout << "-----\n";
88.         }
89.     }
90. };
91.
92. // Main Program
93. int main() {
94.     vector<Barang> inventory; // Menyimpan daftar barang
```



```
95.  
96.   Admin admin("Admin1");  
97.   PetugasGudang petugas("Petugas1");  
98.   Manajer manajer("Manajer1");  
99.  
100.      // Admin menambahkan barang  
101.      admin.tambahBarang(inventory, Barang("Laptop", 10, 15000000));  
102.      admin.tambahBarang(inventory, Barang("Mouse", 50, 150000));  
103.  
104.      // Petugas Gudang memperbarui stok  
105.      petugas.updateStok(inventory, "Laptop", 8);  
106.  
107.      // Manajer melihat laporan  
108.      manajer.lihatLaporan(inventory);  
109.  
110.      return 0;  
111.  }
```

4. KESIMPULAN DAN SARAN

Kesimpulan

Implementasi PBO dalam pengembangan sistem inventaris barang menggunakan C++ memberikan keuntungan dalam hal modularitas, pemeliharaan kode, dan pengembangan fitur tambahan. Dengan penerapan konsep enkapsulasi, pewarisan, dan polimorfisme, sistem ini dapat mengelola data barang secara lebih efisien dan fleksibel. Penelitian selanjutnya dapat mengintegrasikan sistem ini dengan basis data untuk meningkatkan skalabilitas dan aksesibilitas. Sistem inventaris berbasis PBO dengan C++ yang dikembangkan dalam penelitian ini telah terbukti efektif dalam mengelola stok barang dengan lebih efisien dan akurat. Dengan menerapkan prinsip enkapsulasi, pewarisan, dan polimorfisme, sistem ini memberikan fleksibilitas tinggi serta kemudahan dalam pengembangan lebih lanjut. Implementasi sistem berbasis komputer ini mampu meminimalkan kesalahan pencatatan manual serta meningkatkan efisiensi operasional dalam manajemen inventaris.

Saran

Untuk pengembangan lebih lanjut, sistem inventaris ini dapat ditingkatkan dengan integrasi ke dalam basis data untuk penyimpanan data yang lebih aman dan efisien. Selain itu, antarmuka grafis yang lebih user-friendly dapat ditambahkan agar mempermudah interaksi pengguna. Penggunaan teknologi berbasis cloud juga dapat menjadi opsi untuk meningkatkan aksesibilitas dan fleksibilitas sistem, memungkinkan pengelolaan stok barang dari berbagai lokasi secara real-time. Dengan pengembangan ini, sistem diharapkan dapat lebih optimal dalam memenuhi kebutuhan industri yang lebih luas.

DAFTAR PUSTAKA

- Stroustrup, B. (2013). *The C++ Programming Language*. Addison-Wesley.
- Pressman, R. S. (2015). *Software Engineering: A Practitioner's Approach*. McGraw-Hill.
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- Sommerville, I. (2011). *Software Engineering* (9th ed.). Pearson.
- Booch, G. (2007). *Object-Oriented Analysis and Design with Applications* (3rd ed.). Addison-Wesley