**🛠️ 1. Compilation Engine (Pebble → C Translator)**

**🔹 Purpose:**

Pebble ka high-level code samjhta hai aur usko equivalent **C code** me convert karta hai.  
Yeh step manual hai (custom C code likha gaya hai), jo input le kar ek output.c file me likhta hai.

**🔹 Tool: Custom C Code (pebble\_compiler.c ya similar)**

**🔹 Example:**

**Input (Pebble Code):**

let a = 5;

let b = 10;

display(a + b);

**Generated Output (C Code):**

#include <stdio.h>

int main() {

int a = 5;

int b = 10;

printf("%d\n", a + b);

return 0;

}

**🔹 Notes:**

* let → int
* display() → printf()
* Variables directly mapped
* Strings are handled with "%s" format

**🔹 Output File:**

This code is written to a file named like output.c

**⚙️ 2. GCC Compiler Phase**

**🔹 Purpose:**

Generated C code (output.c) ko **machine-level executable** me convert karta hai using **GCC** compiler.

**🔹 Tool: GCC (GNU Compiler Collection)**

**🔹 Command:**

gcc output.c -o run

* output.c → C code file
* -o run → Executable filename (run ya run.exe)

**🔹 Possible Outputs:**

* If no error: Creates executable run
* If error: Shows **compiler error**

**🔹 Common Compiler Errors:**

| **⚠ Error** | **💬 Explanation** |
| --- | --- |
| undeclared variable | Pebble variable not defined properly |
| expected ';' | Missing semicolon in converted C code |
| printf format mismatch | String/number mismatch in display() |

**✅ Example Error**

**Pebble Code:**

let a = "hello";

display(a + 1);

**Translated C Code:**

char\* a = "hello";

printf("%d\n", a + 1); // ❌ ERROR: invalid arithmetic on strings

**GCC Output:**

vbnet

CopyEdit

error: invalid operands to binary + (have ‘char \*’ and ‘int’)

**🚨 3. Error Handler (Your Custom Checker)**

**🔹 Purpose:**

Yeh phase Pebble code check karta hai for:

* Syntax errors (missing brackets, semicolons)
* Semantic errors (undeclared vars, invalid operations)
* Friendly error messages print karta hai

**🔹 Tool: Custom C code (in parser or semantic phase)**

**🔹 Example Messages:**

* ✅ Error: Variable 'x' not declared before use.
* ✅ Error: Expected ';' after statement.
* ✅ Error: Invalid operation — cannot add string and number.

**🔹 Usage:**

* In Lex/Yacc: add error productions using error token.
* In custom C code: use fprintf(stderr, "...");

**✅ 4. Output Phase**

**🔹 Purpose:**

Final executable (run) ko execute karna aur output terminal pe dikhana.

**🔹 Command:**

./run

**🔹 Output Example:**

**Pebble Input:**

let x = 10;

let y = 20;

display(x + y);

**Terminal Output:**

30

**📊 Summary Table for Report**

| **Phase** | **Tool/Language** | **Input** | **Output** | **Error Handling?** |
| --- | --- | --- | --- | --- |
| Compilation Engine | C (Translator) | .peb code | output.c | Semantic errors |
| GCC Compiler | GCC | output.c | Executable | Syntax/type errors |
| Execution | Terminal | Executable | Final Output | Runtime errors (div by 0 etc.) |

**📦 Folder Structure Suggestion:**

lua

CopyEdit

Pebble\_Project/

├── input.peb

├── pebble.l

├── pebble.y

├── pebble\_compiler.c

├── output.c

├── run (Executable)

└── output.txt (optional)