Task 1B – Algorithm and Data Design Template

(Universal Structure for Any Scenario)

1. Systems Overview

What to include:

- A clear description of the proposed digital system
- The intended users
- The core functionality
- How the system addresses the client's objectives

Writing framework:

The proposed digital system is designed to ________________________________.

The primary users of the system are ________________________________.

The system will allow users to ________________________________.

This solution supports the client objective of ________________________________ by ________________________________.

Ensure this section:

- Explains the purpose of the system
- Identifies who interacts with it
- Links directly to the client brief

2. Inputs Identified

Structure:

Divide inputs into logical categories where appropriate:

- User inputs
- System-generated inputs
- Staff/admin inputs

Table format:

| Input Name | Data Type | Entered By | Purpose |

Writing guidance:

For each input, clearly state:

- What data is being captured
- Why it is required
- Where it is used in the system

Avoid listing inputs without explaining their role.

3. Processes Identified

What to include:

Break the system into logical processes such as:

- Authentication
- Data validation
- Record creation
- Record updates
- Data retrieval

Writing framework:

The system first processes ______________________________.

This ensures that ______________________________.

Once validated, the system proceeds to ______________________________.

The data is then stored/retrieved/updated in ______________________________.

Each process must clearly:

- Use defined inputs
- Perform a logical action
- Produce a defined output

4. Outputs Identified

Table format:

| Output | Triggered By | Displayed To | Purpose |

Writing guidance:

Explain:

- What the user or staff member sees
- When the output is generated
- How it supports usability or decision making

Ensure outputs directly relate to earlier inputs and processes.

5. Main Flowchart

Requirements:

The main flowchart must:

- Represent the primary control logic of the system
- Include input, process, decision and output steps
- Call subprograms where appropriate
- Follow standard conventions

Ensure:

- Decisions are clearly labelled
- Subprograms are named consistently
- The flow is efficient and logical

6. Pseudocode Section

You can use the following reference to help structure the pseudocode design for your subprograms: https://tools.withcode.uk/ks4pseudo/media/edexcel_pseudocode.pdf

Each subprogram must follow a consistent format:

```
FUNCTION SubProgramName(parameters)
BEGIN FUNCTION
DECLARE variables
Process logic
RETURN value (if applicable)
END FUNCTION
```

Requirements:

- Use consistent naming conventions
- Use standard pseudocode keywords
- Include appropriate data types in the variable list
- Ensure logic is precise and produces correct outcomes

Avoid mixing styles or inconsistent structures.

7. Variable List

Table format:

| Variable Name | Data Type | Description | Used In |

Requirements:

Include:

- Input variables
- Processing variables
- Output variables
- Loop control variables
- Boolean validation variables

Ensure:

- Data types are appropriate
- Naming is consistent
- Every major variable in pseudocode is listed

8. Error Handling Plan

Table format:

| Error Type | Where It Occurs | System Action | User Feedback |

Must cover:

- Missing required input
- Invalid data format
- Logical validation failure
- Database constraint failure
- System/network failure

This section must demonstrate:

- Prevention of invalid data entry
- Protection of database integrity
- Clear feedback to the user

9. Normalisation

Use the following structured headings:

Unnormalised Form (UNF)

- Identify repeating groups
- Identify non-atomic fields

First Normal Form (1NF)

- Ensure atomic values
- Remove repeating groups
- Identify primary key

Second Normal Form (2NF)

- Remove partial dependencies
- Separate data into appropriate tables

Third Normal Form (3NF)

- Remove transitive dependencies
- Ensure each table stores data about one entity only
- Introduce foreign keys

Each stage must explain:

- What changed
- Why it changed
- How it improves data integrity

10. Entity Relationship Diagram (ERD)

The ERD must:

- Clearly identify primary keys
- Clearly identify foreign keys
- Show 1:M relationships
- Show resolution of M:M relationships

Include a short explanation describing:

- How the entities relate
- How the design supports the system

11. Data Dictionary

Each table must include:

| Field Name | Data Type | PK/FK | Null Allowed | Description | Validation |

Ensure:

- Data types match SQL implementation
- Validation rules reflect system constraints
- Naming conventions are consistent

12. SQL Implementation

For each table include:

- CREATE TABLE statement
- INSERT examples
- SELECT query
- At least one UPDATE example where relevant

Explain:

- How SQL supports system functionality
- How data is stored and retrieved
- How referential integrity is maintained

13. Data Types and Structures Summary

Include a short paragraph explaining:

- Why specific data types were chosen
- Why relational tables are appropriate
- How link tables manage complex relationships
- How loops or control structures manage repeated data

Final Section Conclusion

Conclude by explaining:

- How the algorithm supports system functionality
- How the database structure ensures accuracy and integrity
- How the design meets the client’s objectives
- Why the solution is scalable and maintainable

Submission Checklist for Full Marks

Students must demonstrate:

- Clear decomposition into inputs, processes and outputs
- Precise and efficient logical structure
- Consistent conventions throughout
- Fully appropriate variables and data types
- Appropriate data structures
- Thorough error handling
- Accurate normalisation to 3NF
- Correct ERD relationships
- Working SQL aligned to the design

If all sections are completed using this framework, the submission aligns securely with Band 3 descriptors.