# ICT6 PROJECT GUIDE

# MAJOR PROJECTS IN MS ACCESS

**ICT6 Major Coursework Guide Database projects in MS Access –**

## Contents

## Copyright/Acknowledgements

This project guide is provided "copyright free" i.e. schools and colleges can distribute it to students in paper or electronic format (via a school/college intranet or website) and can make changes to the guide to meet the needs of their own students. If you do distribute the guide and/or make changes I would appreciate feedback about:

- How you used the guide
- Student perceptions
- Limitations of the guide and/or errors
- If you changed it how and why
- Suggested improvements

Some parts of the project guide contain information already published in other documents (books/websites etc.) Where possible, I have aimed to acknowledge intellectual property where it exists and I appreciate having any oversights drawn to my attention. **Thanks to D. Yates at the KJS, Essex for his valuable input and suggestions in the construction of this guide. The KJS website** www.thekjs.essex.sch.uk/yates/ **contains lots of additional coursework resources that you should refer to when planning and implementing ICT6 (and ICT3) projects. FatMax January 2002** fatmax_uk@yahoo.com

## Introduction

This project guide has been designed to give you advice about completing the documentation for AQA's ICT6 coursework module. Before starting your project read the advice below from the AQA specification.

The major project will require candidates to identify and research a realistic problem for which there must be a **real** end-user. (The candidate is not permitted to be their own end-user) The problem will be of a substantial nature and is intended to integrate the various skills and concepts developed during the course. The **emphasis** will be on the project being an open system of a cyclic nature, such as being repeated once a year or once an event. The solution is likely to involve the appropriate use of a range of advanced features and functionalities. It is possible that these may be provided by a suite of generic application software (e.g. MS Office).

To obtain high marks it is expected that the candidate's solution must accommodate the system's **information flow and data dynamics**. There is likely to be some consideration of initialising the system, clearing down data from the previous use, processing data, transferring data such as logging transactions and archiving data.

The project will involve the candidate in identifying a problem requiring information technology tools and techniques for its possible solution and then selecting the appropriate tool or tools for the solution of the problem.

The **emphasis** in the major project will be on the candidate's ability to produce a high quality analysis and design and to document the solution in a comprehensive manner.

**ICT3 vs. ICT6? A summary:**

- *The minor project (ICT3) can be a "task-driven solution" but the major project (ICT6) must be a "reusable system". Your system should be an MIS, which passes information to people who need to make decisions.*
- *In the major project, there should be more use of "advanced features" e.g. action queries, VBA, expressions, macros, and so on. It is very important, however, that you don't use advanced features for their own sake. You should use them **appropriately**. Your choice of project, therefore, is very important.*
- *The other major difference is in the quality and quantity of technical documentation.*

Thanks to D. Yates at the KJS for this information

## Project guide structure

This guide is divided into 7 sections covering the 7 criteria of the AQA ICT6 mark scheme. Before starting your project you **must** read the **whole** guide thoroughly as it does not follow a **logical order of progression**! For example, you'll need to read and understand the contents of section D (testing) before you can tackle sections B and C (Design and Implementation).

| **Section** | **Marks** |
|---|---:|
| 1. Analysis | 18 |
| 2. Design | 16 |
| 3. Implementation | 15 |
| 4. Testing | 12 |
| 5. User Guide | 8 |
| 6. Evaluation | 10 |
| 7. Report | 8 |
| **Total Mark** | **90** |

Each section of the guide has been split into 3 parts:

**1. Aim**
Some background information about the AQA specification mark scheme and what you should be trying to achieve.

**2. Preparation**
Advice about the essential work, such as research and planning, that you will need to carry out before tackling the documentation.

**3. Documentation**
Set of guidelines for completing the documentation that you will submit for marking.

## Mark Scheme

The grade boundaries for the ICT6 module in 2004 were:

| **Grade** | **Mark (out of 90)** |
|---|---|
| A | 59+ |
| B | 50 – 58 |
| C | 42 – 49 |
| D | 34 – 41 |
| E | 26 – 33 |
| N | Who cares?!? |

For a more detailed breakdown of the mark scheme by assessment objective look at:

http://www.dohacollege.com/dcweb/zips/ict6criteria.pdf

## Improvements

Your feedback is welcome! If any teachers or students using this guide would like to offer comments, criticisms or suggestions please mail me

**fatmax_uk@yahoo.com**

## Section A – Specification and Analysis (18 Marks)

### Aim

The aim of this section of your project is to investigate the client's requirements and draw up a set of working objectives. You will investigate an existing system (either computer and/or paper-based) within an organisation and determine how it could be improved by the introduction of a database system. This involves a feasibility study and analysis of existing systems to determine what user requirements (objectives) could be addressed by the introduction of a new system.

**Quote from AQA Specification:**

*It is expected that candidates will carry out the following stages in the work:*

- *Identify and research a real and realistic problem to determine the client's requirements and identify the role of the information technology in meeting these requirements.*
- *Produce an agreed requirement specification.*
- *Analyse the problem and design an appropriate solution.*

## Preparation

You need to find a **real** client (end-user) for your project as soon as possible. After finding an organisation on which to base your coursework you need to get in touch with them to discuss possible projects in more detail. A three-step approach is recommended:

1. **An informal discussion** to establish the problem to be solved. You should then complete **appendix A** and discuss it with your teacher before agreeing to tackle the problem. Remember that the project has to have enough scope to gain you good marks and that you have a limited amount of time in which to complete it. Under no circumstances take on a project that is too big or too small without consulting your teacher.

2. **A formal investigation**. This should take the form of an interview, observation and/or questionnaire. Try to get hold of any original documents (order forms, invoices, receipts etc.) used in the current system for inclusion in your appendices. If a computer based system is already in place try to get screen dumps (or photographs if this isn't possible) of the current system. These will help you to identify necessary inputs, stored data, processing and outputs.

The transcript of the actual interview (or questionnaire) will be included in your appendices but the information gathered here will provide you with the information you need to document the detailed **requirements specification**. Make sure you are fully prepared for the interview – asking carefully considered questions now will provide you with a good framework for tackling the specification and analysis.

In addition, your end-user is likely to be more receptive and helpful if it looks like you are treating the project seriously!

## Interview

The aim of your interview should be to gather information about the **Current System** and proposed improvements. It should cover the following:

- The organisation/department function
- Aims and objectives of the current system
- General procedures including time-lines/frequency of events
- Users of the current system
- Information/data gathered to be **input** – what and how?
- Documents/reports that are produced using the gathered data i.e. **outputs**

- How the data is **processed** into the output information
- Security, storage of data and backup issues
- Problems encountered during any of the above procedures including errors caused and encountered
- Hardware and software available on-site and/or planned to purchase for the new system
- Users' ICT skills – current ICT usage, training etc.
- Client's requirements regarding a new system with reference to the above

Read the rest of this section before preparing for the interview.

3. **Summarise**. On one page of A4, list as numbered points the key factors identified at your interview:

- Current system
- Problems encountered
- User requirements for the new system
- Your added suggestions for the new system based on your analysis of the current system

The client should verify the contents of this document and sign/date their agreement. Both the summary and the interview transcript will be included in your appendices and **must** be referred to in your documentation.

## Documentation

You could use the following outline plan to write up your specification. This is not prescriptive and you may choose to include more headings:

## 1. Introduction

### 1.1 Background

Describe the organisation in very **general** terms and give some brief background information about it i.e. nature of its business; is it a large/small organisation; number of transactions per day etc.

### 1.2 Overview of the problem

Give an overview of the problem you intend to solve using the example format below for guidance:

*"XYZ Ltd deals with 30 telephone orders per day. At present these orders are recorded manually on a carbon pad and then passed on to the warehouse for processing. If ordered items are in stock the warehouse will process the order and complete an invoice and delivery note using a word processing package – total values are calculated by hand. One copy is despatched to the customer and another sent to the Accounts department. The system generally works well but as this side of the business has expanded Mrs. Smith(Partner) feels that the current system has major flaws. She has three concerns:*

1. *On busy days there can be a delay in processing customer orders. The telephone operator has some idea of which items are in stock but their information is based on the previous days stock levels. If an item has run out of stock they will only find out when the warehouse has attempted to process the order. This can lead to the embarrassing situation where customers have to be phoned backed and told that their order can't be processed immediately.*
2. *The manual calculation of order values can lead to mistakes being made. This is particularly true as product prices change on a regular basis. Customers are often quoted one price and then invoiced for a different one. This has led to customer dissatisfaction and even threats of court action!*
3. *The present system seems to generate unnecessary amounts of paper! Communication between telephone operators, the warehouse and accounts department is by printed copy.*

*Mrs. Smith feels that the introduction of a computer database system could tackle some or all of these problems. She is hoping for a system that can process customer orders faster and more accurately. She also wants to reduce the amount of paper generated by the system and feels that a "centralised" system could improve communication between the 3 departments.*

*I believe that the introduction of a new computer based system could address Mrs. Smith's requirements and in addition could also:*

- *provide her with MIS information (sales, customers etc)*
- *provide a facility for backing up important documents*
- *improve data security*

In your statement avoid phrases such as "up-to-date" and "modern system" as they are meaningless. Try to identify what the client wants that databases are the most appropriate software for – storing large volumes of data, which can be accessed and processed quickly and accurately based on user-defined criteria.

## 2. Investigation

### 2.1 The current system – detailed description

This should be a detailed description of the main points of your investigation/ research regarding the current system. Describe how the current system operates with the aid of data flow diagrams and flowcharts. You need to identify tasks being performed and data flows. Refer to the questionnaire/interview transcript and any original **annotated** source documents in your appendices.

Your detailed analysis must explicitly identify **inputs**, **outputs** and **processing**:

- List all the data input (and data capture methods)
- Describe the information output (including format and content)

- Describe what data processing is carried out

### 2.2 Problems with the current system

What problems exist with the current system? You need to try and be specific here. Orders take too long to process leading to customer dissatisfaction. It is difficult to find out if an item is in stock. Invoices have to be calculated manually leading to the possibility of mistakes being made.

List and describe the problems identified in detail. The more problems or potential problems you can identify the more scope you will have to exploit the features of Access.

## 3. Requirements of the new system (Requirements Specification)

### 3.1 General objectives

Referring to 2.2 above describe the client's requirements and your aims in very general terms. This might include, for example holding details about customers, products and orders or calculating the value of invoices. The specific objectives outlined below will act as performance criteria for your project.

### 3.2 Specific objectives – quantitative

Prepare a numbered list of your quantitative objectives. These are objectives that can be measured i.e. it should be possible to find a customer's details much faster; stock levels should be automatically updated when an order is processed.

### 3.3 Specific objectives – qualitative

Prepare a numbered list of your qualitative objectives. These are objectives that can't easily be measured (e.g. the system should be user friendly) but are important for the project to be successful. On completion of the project the client will need to be involved in the testing to show that you have met these objectives.

Get the client to sign and date your objectives to show that he/she agrees with your assessment of the problem and what needs to be achieved.

This part of your paperwork is **VERY** important, as you will need to refer back to it in your design, testing and evaluation. Objectives should be numbered so that you can refer back to them e.g. "Objective 3 was met by…" Unless you can link these sections of your coursework to specific objectives you may score badly.

**Requirements Specification**

You should describe all the requirements in detail, using ICT terminology wherever possible.

- Data capture methods and procedures
- Validation and verification

- Data Processing
- Output required (including format)
- The user interface
- Navigation around the system
- MIS issues
- Security issues
- Data backup issues
- Archiving
- Training needs

## 4. Performance Criteria

With reference to the Requirements Specification describe how you will assess whether your solution has fulfilled the requirements. In table format list the performance criteria as either **qualitative** (user/process/system effectiveness) or **quantitative** (realistic timing issues in comparison to the old system). This section is very important as it will be used to carry out your evaluation.

## 5. Constraints

### 5.1 Hardware

Describe the hardware available to you at the client’s organisation, at school and at home.

### 5.2 Software

Describe the software available to you at the client’s organisation, at school and at home.

### 5.3 User’s IT skills and knowledge

Give an indication of the client’s ICT skills (novice, intermediate or expert?) and identify what effects this may have on the design of the new systems e.g. will the user interface need to be intuitive? Explain any training needs that will need to be addressed.

### 5.4 Legal Implications

If your system includes data of a personal nature it will need to comply with the Data Protection Act 1998. Outline the nature and content of the DPA and explain how you will ensure that your system does not breach this Act.

## 6. Project Plan I

Draw up a “rough” project plan/Gantt chart which covers the different stages of the system life cycle – analysis, design, implementation, testing, evaluation and documentation. This does **not** have to be a work of art! Nor does it, at this stage of the project, need to be particularly detailed. It is here to help you manage your time effectively throughout the project. A more detailed project plan will follow when you have tackled the design and have a better understanding of the tasks and sub-tasks you will need to complete.

## Section B – The Detailed Design (16 Marks)

### Aim

Really this section should be entitled "Analysis and Design" as both the **Analysis** (18 marks) and **Implementation** (15 marks) sections of the mark scheme require you to show evidence that you have carefully considered the client's needs and have thought about different ways of tackling the problem before embarking on a solution.

Detailed design work demonstrates that you have analysed the problem in detail (gaining good marks on the Analysis section) and helps you consider the "advanced" Access features you could use to implement an effective solution (gaining good marks on the Implementation section). Effectively, your design contributes towards over a third of the total project mark.

**Quote from AQA Specification:**

*The design phase includes the bringing together of the results of the analysis and the gathering and ordering of information related to the background of the problem into the generation of a range of possible solutions which meet them. This may be alternative types of package or alternative solutions within a package.*

*The solution design should be specified so that a competent person can implement it. There should be a clear specification of how each of the sub-tasks identified in the analysis is to be solved.*

In this section of your project you are trying to take the client's specification and produce a workable database solution on **paper**. A relatively competent 3rd party should be able to implement your ideas from your documentation alone. This part of your project will involve you:

- Identifying the required outputs
- Identifying input data
- Specifying the processing that needs to be carried out (i.e. how to get the outputs you want)
- Comparing alternative solutions
- Devising a test strategy and plan
- Planning a schedule of activities to ensure that the project is finished by the deadline

### Preparation

Obviously, before documenting this section of your report you'll need to do a lot of thinking, planning and drafting.

While you're working through the tasks in the preparation section of this document discuss your ideas/designs with your teacher – he/she has probably seen similar projects in the past and will be able to advise you about how you can improve your designs. THIS IS VERY IMPORTANT! Good ideas can fail because of poor design!

Make sure that your teacher agrees that your design is workable and that it contains sufficient depth to gain you a good mark before tackling the documentation. The basic preparation that you need to do should include:

## A. Database Design

Consider your research and own ideas and try to identify the entities (objects/people/things) involved in the system you are proposing and the relationships (transactions) between them. It's a good idea at this stage to research normalization to make sure you understand why it's necessary and the distinction between 1NF, 2NF and 3NF.

Draw a rough ER-D (entity relationship diagram) for each relationship you identify in the system. This can be revised with the help of your teacher and as your design progresses.

## B. Identifying Outputs

Look back at your specification and research to help identify what outputs you'll have to produce. This part of the design process is particularly important as it will determine **exactly** what input data you're going to need and what processing will have to be carried out on the data to achieve the desired results.

Draft a numbered list of the reports or other outputs (mailing labels, form letters etc.) that you intend your system to produce.

## C. Identifying Inputs and Data Stores

You should now have some kind of idea about what data you'll need to input and/or have stored in order to produce your output. Look at your E-RD and proposed outputs. What data are you going to need to accomplish your objectives? At some stage in the future you or the client is going to have to input this data.

In a two column table list the necessary inputs and stored data.

## D. Processing

As this is only your second (or even first) Access project this will probably prove to be your most difficult task - how to get from A (input) to B (output). Look at the user-requirements/objectives and your output ideas and try to identify any output requirements that cannot come from stored data **alone**. Examples: archiving data, calculations, records that meet a specific criteria etc.

As part of your preparation you should try to identify and draft a list of processing needs **in plain English** e.g. archive last weeks orders, multiply price by quantity on an order, calculate VAT on an invoice, identify overdue subscriptions etc.

Try and identify the Access tools (macros, expressions, action queries) that will be used to address each of the identified processing requirements.

## E. Menus and Navigation

Your system has to be user friendly. How are you going to achieve this? The user interface should be structured logically so that a novice has no problems “navigating” it e.g. (s)he should be presented with a main menu from which (s)he can make a simple choice. Start thinking about how the end-user will access the forms/reports/features that you are going to create and try to design a “navigation” system for your application. Look at the example on page 186 of Heathcote (2nd Edition) for some ideas.

## F. Testing Strategy/Plan

System **testing** accounts for a significant proportion of the marks in the mark scheme.

It’s important at the design stage of a project to think about how you are going to test your prototype. For your testing you’ll need to draw up a testing plan that shows that you have carefully chosen your test data and tested each module thoroughly. In order to do this you’ll need to make sure you understand:

a) What the objectives of testing are
b) Types of testing
c) The distinction between normal, extreme and erroneous test data

Read Heathcote pages 188-190 and make sure you understand these concepts. Start thinking about which elements of your solution need to be tested, why and how?

## Documentation

You could use the following outline plan to document your design. This is not prescriptive and you may choose to include more headings:

## 1. Consideration of a possible solution

### 1.1 Comparison of alternative solutions

Your project could probably be implemented using software applications other than Access (for example a spreadsheet package). One solution might even be to streamline/improve a paper-based system. You need to justify the use of a RDBMS by comparing Access with at least one other package.

Compare and contrast different applications with reference to the Requirements Specification to identify a specific package to use.

### 1.2 Justification for Chosen Solution

Based on the above explain why you have chosen Access. In addition to your reference to 1.1 try to relate your explanation to the key advantages of a RDBMS: the reduction of redundancy, inconsistency, program/data dependency. You **must** justify the chosen package with reference to the performance criteria and in terms of its usability and functionality.

### 1.3 Hardware Requirements

Describe the minimum hardware requirements for the new system (including peripherals).

## 2. Project Plan II

This is a more detailed plan than the one included in the Analysis section. Your preparation should have given you a better idea of all the tasks you will need to tackle in order to meet the deadline. Prepare a plan listing and prioritizing how you will complete the project (including start and end dates). You must describe in detail the **sub-tasks** that will need to be completed and should include a Gantt chart.

## 3. Database Design

### 3.1 Normalization

Explain the purpose of normalization. Show and describe, in detail, the stages of normalization from 0NF to 3NF with reference to **your** design. If you quote definitions of normal forms from other sources (e.g. Heathcote) these **must** be attributed.

### 3.2 Entity-Relationship Diagram

Explain briefly what an E-RD is and draw a diagram to provide evidence that you have normalized the data in your design. The diagram should be fully annotated to explain the **exact** nature of the relationships (e.g. one salesperson is allocated one company car – one-to-one relationship). Write a brief sentence to explain the purpose of each table in your design e.g. Customers – this will hold details such as the name and address of existing customers. In the case of "link" tables (many-to-many relationships) such as Orders explain why the link is necessary. Make sure that you use **Leszinsky-Reddick** naming conventions.

### 3.3 Data Flow

Draw a level one Data Flow Diagram for the whole system and a series of **referenced** and fully annotated level two DFDs showing the flow of data through each of the proposed subsystems.

### 3.4 Data Descriptions (Data Dictionary)

You will need to describe in detail the attributes of every piece of stored data in your system. For each table described above complete a table design sheet – Appendix B.

Attention to detail here is particularly important. You receive better marks for showing an appreciation of the importance of data type, field length, format, default values, validation rules and input masks. Annotate your design sheets to explain why you have made decisions e.g. why is the product description field data type memo? Why did you use a lookup field on the customer's courtesy title?

### 3.5 Query Design

For each named query complete a query design sheet – Appendix C. The design sheet should be fully annotated to describe the purpose of the query, its type, criteria used and fields to be output

### 3.6 Macro Design

For each identified process describe the purpose of the macro, how it is initiated (OnClick etc.) and the actions required. You could structure your macro designs using a three column table i.e. column 1 – Action; column 2- Arguments; column 3 – Conditions (if applicable).

## 4. Interface/Input Design

### 4.1 Forms - Background

Explain why the use of forms is important in a **database application**. Remember that you are designing a system for another user and not yourself. Your end-user may be a database novice and could easily get into trouble trying to enter, amend or delete data at table and query level. Using input forms protects her or him from the reality of the database – they don't need to know how that data is stored in different tables or how to perform operations such as searching the database. The forms you create will act as a "bridge" between a novice user and a complex system e.g. one important use of forms is the implementation of "form level validation" – the use of tools such as check boxes, radio buttons and combo/list boxes to limit data entry.

Re-read your tutorial notes and the Access help files (about forms) for helpful information about what you could include in this section.

### 4.2 Storyboard

Draw a detailed storyboard of the system from the user's viewpoint i.e. the interface forms/switchboards, dialogue boxes and outputs. You must show the links between "boards" and for each sketched board add reference numbers that can be cross-referenced in the sub-sections below.

### 4.3 Interface/ Input Forms

Interface/Input forms need to be sketched by hand (or using a DTP package). Your form designs should be annotated to describe each form element used. The designs do not need to be works of art but should contain enough detail and information to allow a 3rd party to implement the design **as you intended**

**Form design checklist**

*For each form you must include:*

- *A description of the forms purpose*
- *Sketches of designs in development – including any rejected ideas*
- *Fully annotated sketches – source or target tables/queries/fields, background or text colours and styles, additional formatting*
- *Reference numbers to pages or sections for controls that run macros or trigger events*

Try and maintain a consistent look and style for each form.

You should try in your designs to show different methods of achieving the same objectives.

Your forms may develop as your project progresses – you may find that an idea doesn't work or that there is a better solution to the one you originally designed. This isn't important! You can show, by including the original designs (annotated, in the appendices) how your project has progressed.

## 5. Output Design

### 5.1 Reports - Background

Explain why the use of reports is important in a **database application**. Re-read your tutorial notes and the Access help files (about reports) for helpful information about what you could include in this section.

### 5.2 Reports/Other Output

Reports and other outputs need to be sketched by hand (or using a DTP package). Your designs should be annotated to describe each element used. The designs do not need to be works of art but should contain enough detail and information to allow a 3rd party to implement the design **as you intended**

**Output design checklist:**

*For each output you must include:*

- *A description of the outputs purpose*
- *Sketches of designs in development – including any rejected ideas*
- *Fully annotated sketches – source tables/queries/fields, background or text colours and styles, calculations, additional formatting*

Try and maintain a consistent look and style for each output.

Your outputs may develop as your project progresses – you may find that an idea doesn't work or that there is a better solution to the one you originally designed. This isn't important! You can show, by including the original designs (annotated, in the appendices) how your project has progressed.

## 6. Other Issues

Some, if not all, of the data in your application is going to be commercially sensitive. Assigning passwords and/or security levels allows you to control who sees what in your application. Explain why security is necessary (refer to the DPA) and explain how you will protect the data in the application.

Prepare a strategy for backing up the system. This can be a simple batch file to take a copy of the database.

## 7. Testing Plan

### 7.1 Testing Strategy

Write a few short paragraphs about the nature and purpose of testing to show that you recognise why testing is so important. With reference to **your** Requirements Specification discuss the testing strategy in terms of:

- **Functional Testing** - unit testing of individual components (e.g. macros, action queries etc.) and integrity testing of combined units.
- **System Testing** – a full system test from beginning to end
- **End User Testing** – testing by the client

### 7.2 Testing Plan

Your testing plan is very important! A good design will take into account any and all possibilities of why the system might fail (the aim of testing is to provoke failure!) Using the example format in appendix XX, draw up an **exhaustive** test plan for your project.

It cannot be emphasised enough how **important** this element of your design is. Make sure that every field and control in your application is tested. Look back at your input/output/table designs and identify what you need to test. Make sure that you test relationships, queries, calculations, expressions, macros, data types, field lengths, validation rules and input masks.

Try to be systematic in drawing up your plan. Take one module (function) at a time and think about how you can test each element.

Design a questionnaire for the end-user element of the testing strategy

**Checklist for testing plan**

- *A numbered list of what elements you are going to test, when you are going to carry out the test and a reason given as to why you are testing an element*
- *If validation is applied to fields then each field will need to be tested.*
- *What values you are going to use including normal, extreme (boundary) and erroneous values.*
- *How many times are you going to try out the same test to ensure that it was not just coincidence?*
- *What results are expected for each value?*
- *An extra column for the actual value returned on testing and another for your resulting comment.*

## Section C – Implementation (15 Marks)

## Aim

Now you have designed your system you need to implement it to make sure it actually works! Remember, you can only get marks for implementation based on the written evidence you supply.

**Quote from AQA specification:**

- *The candidate has fully implemented the detailed design unaided, in an efficient manner with no obvious defects. All of the appropriate facilities of the software and hardware available were fully exploited.* ***The documentation is clear and thorough.***

**What this means:**

- *You shouldn't assume that the Examiner is an Access "guru". He/she will certainly understand the concept of databases but may be more familiar with another package. It is* ***your responsibility*** *to ensure that the Examiner understands how you have exploited the features of your chosen software.*
- *A competent 3rd party should be capable of creating your system from the documentation alone*

The implementation section of your documentation should be a step-by-step guide to building the system using detailed screenshots and annotations. The commentary should describe how you approached the task, in what order, what was done, what decisions were made and why, any problems encountered and how they were dealt with. You must document any deviations from the design and explain why.

Write the guide aimed at someone with the same level of expertise as you. This is not a beginner's guide to Access.

As you are working with a short deadline you'll need to be organised to ensure that you can implement your solution **and** document it in the time available. If you don't have access to a PC outside of school there is still a lot of work that you can be doing at home

**Time Management**

- *Read software guides and/or tutorials to help you overcome any problems you encounter*
- *Refine your designs or annotate printouts*
- *Create sets of test data*
- *Draw up test plans*
- *Plan/draft your documentation*
- *Plan how you are going to make the best use of your next practical session*

## Preparation

Good organisation is essential if you are going to score well on the implementation. Many good projects fail to impress the Examiner because students have not given enough thought to documenting their implementation. It's not enough to produce a fantastic working database! The only evidence you can submit to the Examiner is paper-based and without this paperwork you cannot be credited for what you have achieved.

1. Buy a binder or folder and keep everything related to your project carefully stored in it. This should be divided into sections so that you can quickly find what you are looking for. At this stage you may choose to keep your notes in plastic wallets, for organisational purposes, but your finished report must not be handed in inside either a ring binder or wallets.

2. Make sure you have your design documentation in front of you and **use** it. You need to ensure that you don't forget to implement all elements of your design. If your design needs to be altered for any reason do so, and make a note of why. This is inevitable, as you'll still be learning how to use Access as you create your application and will be finding new and better ways of accomplishing your objectives. Under these circumstances you will be credited rather than penalised for showing how your designs have developed.

3. Make sure that you take two backup copies of your work after every revision. Blaming Microsoft, the school network and the family dog won't bring back your project if you lose it! It's a good idea to "time stamp" backups with the date/time in the file name so that if things go terribly wrong you can systematically work backwards through your backups until you find a "good" version.

4. Create a project implementation log to record what you did, when and why. Fill this in at the end of every practical session (leave about 10 minutes at the end of the session) and note what you did, any problems you encountered and what you need to do next. This will help you complete the **Implementation Schedule** described below.

5. Take screenshots as you implement the solution to show how the design progresses to finished application. It's far easier to document the implementation as you go along rather than finish the practical work and then have to reproduce what you did just to get a screen dump. Documenting your project in this way allows you to show how your design has progressed. Do not use the built-in Access Documenter to provide the sole evidence of implementation, as this will simply produce pages of meaningless "code". If you do need to make use of the Documenter you should be very selective in its use and **must** hand annotate any output you use.

## Documentation

You could use the following outline plan to write up the implementation section of your project. This is not prescriptive and you may choose to include more headings:

### 1. Implementation Schedule

Refer back to **Project Plan II** (Section B – The Detailed Design) and describe whether the identified sub-tasks were completed **on schedule**. If you failed to meet a deadline explain why e.g. unrealistic deadline, design faults etc.

## 2. Implementation

### 2.1 Tables

Having created the database you should show each table in Design View. Provide evidence of your work by taking screenshots of the completed Design Views. Where certain fields require additional parameters (e.g. validation rules) to be set you will need to add cropped shots of that part of the screen. Annotate all of your work by hand or by using callouts in Word or Publisher.

Include copies of the tables in Datasheet View.

### 2.2 Relationships

Include a screenshot of the established relationships between tables annotated to describe any additional parameters that were set e.g. referential integrity or cascading deletes/updates.

### 2.3 Queries

Show each query in Design View (or as recommended in SQL view) annotated to explain the expected result of any criteria, actions, sorting or expressions. Where criteria are entered directly by the user (e.g. parameter query) you will need to add cropped shots of that part of the screen. Annotate all of your work by hand or by using callouts in Word or Publisher.

Include copies of the queries in Datasheet View to show that data is updatable.

### 2.4 Forms

Show the **development** of forms in Design View with annotations describing links made to fields/tables/queries etc and any associated expressions written for any element of the form. Also describe any additional and/or conditional formatting that has been carried out or properties that have been changed. Where macros, code, expressions or sub forms have been used you will need to add cropped shots of that part of the design. Don't forget to include the Switchboard in your notes.

Include copies of the forms in Form View to show the effect of changed property settings e.g. maximise, resize, borders, scrolling etc.

### 2.5 Reports

Show the development of reports in Design View with annotations describing links made to fields/tables/macros/queries etc and any associated expressions written for any element of the report. Also describe any formatting carried out.

Do the same for any other non-Access output (mailing labels, form letters etc.) you have used in your system. Print a fieldname view of the mail merge document e.g. Word, Publisher, and annotate it to describe the associations made.

Include copies of the printed reports to show that formatting (e.g. page width) is correct.

### 2.6 Macros

List the macros used including where they are used or initiated and the actions set or code written.

### 2.7 Security/Start-up

Use screenshots to show how you set the users' rights to establish the security requirements of your system. Show how you set the parameters for start-up of the database.

## Section D – Testing (12 Marks)

Testing should really be an on-going process throughout the implementation of your project. You should test each module as it is completed to ensure there are no major design flaws before moving onto the next module (see appendix F). This is particularly important in the design of a relational database where seemingly minor faults in a module design can have wider implications later on.

Testing is included as a separate section of your project documentation because many students fail to fully understand its importance and as a consequence fail to provide sufficient evidence that they have produced a workable solution to their problem.

### Preparation

Before embarking on your testing:

1. Read appendix F and make sure that you understand the different types of testing you will need to carry out and the reasons for these tests.

2. Make sure you understand the nature of "good" testing. Read or reread Heathcote pages 188-189 (2nd Edition). Pay particular attention to Heathcote's words "…it's not sufficient to think up a few tests you are fairly sure will not make your program crash or give a wrong result; you must use tests and test data which has been carefully thought out to test all parts of the program".

3. Make sure that you understand the distinction between normal, extreme and erroneous test data. Start thinking about how these concepts apply to your project and try to draw up data sets that will test how your database performs under different conditions.

4. Look carefully at the system requirements of your project and/or your designs. Try and identify the operations your database is expected to perform – calculations, opening a form or report with data already in it. How are you going to test these operations?

5. Does your project include any validation rules? If so these need to be tested. Identify wherever validation and/or input masks have been used and consider what type of type of data you could use to test the validation.

6. Consider your original qualitative objectives. One of them should have been the creation of an intuitive, user-friendly system. Only the end-user can test whether you have achieved this. Draft an interview pro forma or questionnaire for your end-user to complete when they have tested the finished system.

## Documentation

You could use the following outline plan to write up your testing section. Hard copy evidence of test runs (referred to in the last column of the table) should be included in your appendices, should be clearly cross-referenced in the test plan and should be clearly annotated to show evidence of a successful test run.

## 1. Testing

### 1.1 Module Testing

If you've followed the steps outlined above writing up this part of your project should be fairly easy, although time-consuming!

Appendix F uses the example of one module to show how you can use tables to organise module testing. Note: It does not include a **full** set of tests or test data and is for illustration only. You will need to test all modules and include more tests for each module.

Make sure that the evidence referred to in the tables is fully annotated and where possible cross-referenced to your test plan.

**Quote from Heathcote:**

*"... projects with no annotation and no cross-referencing to the test plan are virtually useless!"*

Heathcote

### 1.2 System Testing

Most, if not all, of your modules can be tested in isolation. Once all the modules are complete you need to test the entire system with a realistic set of test data. You need to provide some evidence that your modules work together.

### 1.3 End-User Testing

Include a copy of your questionnaire completed by the end-user in your appendices. Restate your qualitative objectives and any other requirements the end-user asked for in the original specification and produce a short report (no more than one side of A4) cross-referenced to the end-user's comments.

Don't be afraid to include "negative" criticism! In the real world projects like this might take months, or even years, to get right – you've had 2 months!

There are many reasons why an objective might not have been achieved:

- The end-user wasn't clear enough about her/his objectives
- There wasn't enough time to carry out all the tasks
- The objective wasn't realistic to start with
- New ideas made the objective redundant

In Section F – Evaluation you are credited for recognising the limitations of the solution you have produced.

Get the end-user to sign and date your report to show that (s)he agrees with your evaluation of the end-user testing.

## Section E – Documentation: The User Guide (8 Marks)

### Aim

A new system will typically come with two pieces of documentation:

- Technical Documentation

- A User Guide

Your aim is to provide clear and comprehensive documentation that users of your system can refer to during the introduction of the system (**training**) and use as a reference if they experience difficulties later (**troubleshooting**).

**Quote from AQA mark scheme**

*A comprehensive, well illustrated user guide is produced that deals with all aspects of the system (Installation, backup procedures, general use and trouble shooting).*

**The User Guide: An important point**

*A draft User Guide should be handed to the client* ***before*** *end-user testing. Documentation is part of the system itself and the quality of help documentation and/or support is often a major consideration for organisations when purchasing software. If the client suggests improvements you should document this in both the end-user testing (section D) and evaluation (section F).*

Thanks to D. Yates at the KJS for this suggestion

### Preparation

Read the notes below:

## 1. Technical Documentation

Technical documentation is provided for the person who will administer your system when it is in place. (S)he will be responsible for ensuring that the system is maintained and/or updated if the organisation decides to extend the system. In order to do this (S)he will need to know about the structure of your system – how are tables related, which controls are dependent on others, how are controls formatted etc.

Your paperwork in the design, implementation and testing sections covers this to a large degree and you can mostly ignore this aspect of documentation (since no specific mark is allocated!) However, in the introduction to the User Guide, it is **important** that you outline general administrative procedures – install, backup, troubleshooting etc.

## 2. User Guide/Manual

The people who will be using your system on a day-to-day basis may need to be trained in its use or may encounter occasional problems. If your system is intuitive the User Guide will be **fairly** short. However, the guide must be comprehensive.

**User Guide Checklist:**

- *A table of contents, numbered pages, consistent style*
- *Introduction – what the system is about and who it is for.*
- *Modules – how to use each part of the system e.g. what do the various buttons on a form do? Prepare the chapters about individual modules before writing the first chapter which should give an overview of the system.*
- *Special Considerations – explain any special validation rules that might apply.*
- *Error Messages – error messages that may be displayed and what to do in the event.*
- *A help number or Email address*

## Documentation

Your User Guide should be aimed at a relatively non-technical user and where possible should use sympathetic language i.e. plain English rather than geek-speak. Presentation is very important, use screenshots rather than long descriptive lists. The User Guide should be a separate document and should be bound separately from the rest of your coursework.

### 1. The Good User Guide

Explain why user documentation is important and the factors that contribute towards good documentation.

### 2. User Guide

Noting the points in "preparation", create the User Guide. In the main body of your documentation include a reference to the **separate** User Guide and explain how you have addressed issues relating to "good" documentation. This can be a simple checklist i.e. sympathetic language, consistent style/labelling/numbering, use of screenshots etc.

3. **Online User Guide (Optional)**
   If you are using Microsoft Word to create the User Guide use the **save as web page** option to create a HTML version which can either be included on an installation CD or uploaded to the client's internet/intranet presence. If you choose to include online help you should explain its benefits/drawbacks and include screenshots.

## Section F - Evaluation

### Aim

The system life cycle is a continuous process – problems are analysed, solutions are designed, implemented and tested and then the whole process starts again! Look at any popular piece of software and you'll find it's in its nth version (Paintshop Pro 7, Office 2000, Flash 5 etc.) Why? After completing projects good system designers carry out a thorough evaluation of their solutions. Features that work well in the solution will be incorporated into later versions while those features that are missing or don't perform as well as expected are either added or replaced.

**Quote from AQA mark scheme**

- *The candidate has considered clearly a full range of qualitative and quantitative criteria for evaluating the solution.*
- *The candidate has fully evaluated his/her solution intelligently against the requirements of the user(s).*
- *Evidence of end-user involvement during this stage is provided.*

It's tempting, after mastering Access and producing a workable solution, to sit back, think "I've finished!" and produce a 3 line evaluation patting yourself on the back for a job well done. **Don't!**

1. **No design is perfect.**

- Due to time limits you may not have been able to satisfy all the user requirements.

- You may have had ideas about a feature you would like to have used but at this stage, didn't know how to do it.

- You may have expected Access to perform a function beyond its capabilities.

- Your testing will have identified some limitations and/or other errors – how many calculated fields did you leave named as “expr1”or formatted as number when they should have been currency?

2. **Evaluation accounts for 10% of the total mark!**

- Three lines is hardly likely to gain 10%
- You need to evaluate your project against your original specification, identify its limitations and identify potential improvements.

## Preparation

Evaluation is all about assessing what was successful **and** how your solution could be improved:

1. Look at you original objectives. Make a note of those you achieved and those that weren’t achieved. You’ll need to look at your testing to find evidence that you actually fulfilled these objectives. Make a note of the pages that show the evidence.

2. Try and evaluate why some objectives weren’t achieved and make notes on why not. You don’t need to know the solution but try to explain how your solutions failed. Be honest!

3. Make sure you have an end-user questionnaire. See Section D. Your end-user is the only person who can truly evaluate the system. (S)he should have evaluated your system already. Try and get them to complete a summary evaluation (cross-referenced to the original Requirements Specification) printed or written on company notepaper.

## Documentation

Your evaluation must prove two things:

1. You have, as far as possible, satisfied the Requirements Specification
2. You have taken account of the limitations of your solution

You could use the following outline plan to write up your evaluation. This is not prescriptive and you may choose to include more headings:

## 1. Evaluation

### 1.1 User requirements

Restate your original Requirements Specification and/or objectives as a numbered list. This will provide a checklist for you to evaluate your project against

### 1.2 System Evaluation

Work through the numbered list systematically and evaluate how well you have met your objectives – some will have been completely met, others partially met and others not met at all. Be honest in your evaluation. You'll be given credit for recognising the limitations of your solution.

Don't make broad, sweeping statements such as "it worked well". Each of the comments in your evaluation **must** be cross-referenced to evidence – either the results of your testing or feedback provided by the client.

### 1.3 Enhancements

Even if your system met all the end-user requirements and passed rigorous testing it is likely that it could still be improved. Write a few paragraphs outlining how your system could be improved. This may include minor enhancements (such as adding or removing controls from a form) or major upgrades (adding completely new modules). Suggested enhancements **must** be cross referenced to either the Requirements Specification, testing or the end-user's comments.

## Section G – Report (8 Marks)

This is **not** a separate section of your project. These eight marks are awarded for the general quality of your documentation. Ensure that you have spell-checked your project, that you have included referenced diagrams and that you have a well-organised folder. Include a contents page and an index and number your pages.

The report should be accurate and concise and it should not exceed 4,000 words (excluding annotations, screen dumps, diagrams, etc.)

**Checklist for report:**

- *Consistent style for headings, sub-headings and body text*
- *Consistent numbering/labelling throughout*
- *All diagrams referenced*
- *Report (and User Guide) spell-checked*
- *Contents page and index included*
- *Title page includes centre and candidate number*

## Appendix A – Coursework Guidance Sheet

| |
|---|
| **Organisation:** |
| **Background** (Brief description of the organisation and problem): |

| |
|---|
| **Problem statement** (Describe in general terms the nature of the problem): |
| **Objectives** (Objectives of the new system): |
| **MS Access features to be used:**<br><br>i)<br><br>ii)<br><br>iii)<br><br>iv) |

| | | |
|---|---|---|
| **Project Approved?** | Yes ☐ | No ☐ |

| |
|---|
| **Comment:** |

## Appendix B – Table Design

| | | | | | |
|---|---|---|---|---|---|
| ***Database File*** | .mdb | ***Table Name*** | tbl | ***(Primary/Composite) Key Field*** | |

**Related to:**

| ***Table Name*** | ***Foreign Key*** | ***Table Name*** | ***Foreign Key*** |
|---|---|---|---|
| tbl | | tbl | |
| | | | |

| **General table description:** |
|---|
| |

| ***Field Name*** | ***R*** | ***I*** | ***Data Type*** | ***Length*** | ***Input Mask/Validation Rule*** | ***Default Value*** | ***Description*** | ***Typical Data*** |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Key:** | **R=Required** | Y | (Yes) | N | (No) | | |
| | **I=Indexed** | | Not Indexed | X | No Duplicates | ✓ | Duplicates OK |

## Appendix C – Query Design

| | | |
|---|---|---|
| ***Database File*** | ***.mdb*** | ***Purpose of Query*** |
| ***Query Name*** | ***qry*** | |

| | | | | | | |
|---|---|---|---|---|---|---|
| ***Tables*** | | | | | | |
| ***Field*** | | | | | | |
| ***Sort*** (tick) | Ascending Descending Not Sorted | Ascending Descending Not Sorted | Ascending Descending Not Sorted | Ascending Descending Not Sorted | Ascending Descending Not Sorted | Ascending Descending Not Sorted |
| ***Show*** (tick) | Yes No | Yes No | Yes No | Yes No | Yes No | Yes No |
| ***Criteria:*** | | | | | | |
| ***Or:*** | | | | | | |
| | | | | | | |
| | | | | | | |

## Appendix D - Testing Your System

For your coursework you are required to produce a test plan that would enable someone who knows nothing about your system to be able to test it thoroughly. This means that it must be sufficiently detailed, and not contain vague statements such as "Are all fields present?"

The easiest way to produce a test plan is to use tables, and possibly to break the plan down into sections. Depending on what your system does, you will have some or all of the following sections for each module:

- Initial control states
- Dependent control states
- Validation/Test data

In Access, the word "control" means some feature on the screen, such as a field, menu option, or button that you have created. Everything you can see on a form or report is a control.

The examples which are given below show **some** of the tests that could be carried out on one module – processing a customer order.

### 1. Initial Control States

This section would contain a list of all the features, such as fields and buttons, combos etc. that the module should contain. These could be taken directly from the system requirements and/or your designs.

In some systems, the fields may have default values, e.g. all values start at zero, or today's date, and your plan should also take account of this.

**Test No. 3.1 frmOrders – Initial Control State Tests**

| Test No. | Action/Test Data | Purpose | Expected Result | Comment/ Verified |
|---|---|---|---|---|
| 3.1.1 | Click “Order” button on main menu form | Test format of **frmOrders** | Order form opens<br><br>Screen maximised<br><br>New blank record selected<br><br>All fields visible<br><br>Date and order number automatically filled | Form opens as expected<br><br>Page 57 – screenshot |

## 2. Dependent Control States

This section of your test plan tests any parts of your system that can change another part or that depend on the action of a user. For example, on the order form, changing the quantity ordered should affect a calculated sub-total field.

**Test No. 3.2 frmOrders – Dependant Control State Tests**

| Test No. | Action/Test Data | Purpose | Expected Result | Comment/ Verified |
|---|---|---|---|---|
| 3.2.1 | Select customer "Everard" from combo box | Test function of combo box | Details for Everard – name, address etc. filled in automatically | Pass<br><br>Page 58 – screenshot |
| 3.2.2 | Enter **CustomerID** "Jones" in combo box | Test referential integrity | ID rejected – error message | Pass<br><br>Page 59 - screenshot |
| 3.2.3 | Enter **ProductID** "0001" on subform | Test referential integrity | Product fields, name, price etc. filled in automatically | Pass<br><br>Page 59 – screenshot |
| 3.2.4 | Enter **ProductID** "1000" on subform | Test referential integrity | ID rejected – error message | Pass<br><br>Page 59 – screenshot |
| 3.2.5 | Enter **Quantity** "25" on subform | Test calculated field **SubTotal** | **SubTotal** updated to £200 | Fail<br><br>Page 60 – screenshot<br><br>**SubTotal** not formatted as currency |
| 3.2.6 | Click Add Customer button | Test add customer macro | New blank record created | Pass<br><br>Verified by supervisor |

This is only a sample of what you are expected to do!

## 3. Validation

This is the section of the module testing in which you test your validation and/or input masks. You will also need to look at how each field in your system responds to normal, extreme and erroneous data.

**Test No. 3.3 frmOrders – Validation/Test Data**

| Test No. | Action/Test Data | Purpose | Expected Result | Comment/ Verified |
|---|---|---|---|---|
| 3.3.1 | Enter **Payment** "Visa" | Test validation rule on **Payment** field | Visa accepted | Pass<br><br>Page 68 - screenshot |
| 3.3.2 | Enter **Payment** "Delta" | Test validation rule on **Payment** field | Error message<br><br>Delta rejected | Pass<br><br>Page 68 - screenshot |
| 3.3.3 | Enter **DeliveryDate** "28/02/01" | Test date format – only UK formats<br><br>Normal data | Date accepted | Pass<br><br>Page 68 – screenshot |
| 3.3.4 | Enter **DeliveryDate** "02/28/10" | Test date format – only UK formats<br><br>Extreme data | Date accepted | Pass<br><br>Page 68 - screenshot |
| 3.3.5 | Enter **DeliveryDate** "02/28/01" | Test date format – only UK formats<br><br>Erroneous data | Error message<br><br>Date rejected | Pass<br><br>Page 68 - screenshot |

This is only a **very limited** sample of what you are expected to do! Your testing must be as detailed and rigorous as possible. Test anything and everything! Try and provoke failure. You should test and restest to make sure your results aren't a one-off fluke.

## Appendix XX – ICT6 Checklist

Make sure that you have addressed all of the following issues/points in your coursework. Checklist provided by D. Yates at the KJS Essex.

| **Analysis (18 Marks)** |
|---|
| Identify and delimit a problem that is both real and realistic and substantial ("delimit" means to break the problem down into sub-problems). Prioritise the tasks required. Marks will be lost if you need the help of a teacher to identify the problem. |
| Make a clear outline statement describing the content and the nature of the problem |
| Explain in detail any relationship between the system you intend to produce and the existing manual system. |
| You should have a lengthy interview with your end-user (do not just rely on a questionnaire). Spend time in the working environment and talk to everyone whose working life will be affected by your system. Gather sample documents and include them in your project. Make a full record of any visits/meetings. |

| |
|---|
| Come up with an agreed requirements specification with your intended user(s). Get the end-user to sign his/her requirements. These will be your performance criteria that will drive the whole project, from this point onwards. |
| Identify qualitative and quantitative evaluation criteria (obviously, these should be heavily influenced by the end-user's requirements specification). |
| Explain clearly how your system will improve the current system (avoid vague statements such as "to save time", "to improve efficiency", "to make system user friendly" etc.) |
| Identify those elements of your system that are intended to be "reusable" |
| What hardware and software will be used and why? *See http://www.athree.com/access_info/pros_and_cons.html for a description of the pros and cons of Access.* |
| You should show an appreciation of the full potential of the hardware and software that you intend to use. |
| Fully explain the information flow (include a Data Flow Diagram (see: http://www.studyict.com/data_flow_diagram.htm), Jackson Flow Diagram or systems flowchart of the existing system). See http://www.smartdraw.com/resources/examples/software/ssadm1.htm for examples. |
| Evaluate the user's current IT skill level and their training needs |
| Are there any legal, ethical and social issues arising from what you intend to do (e.g. issues of data protection?) |

| **Design (16 marks)** |
|---|
| Consider in detail a range of appropriate approaches to a solution. Why is an Access solution preferable to an Excel solution? Why is a computerised system preferable to a manual system? |
| Give compelling reasons for the final choice of solution. Consider the likely effectiveness of your solution |
| Follow a process of Normalization |
| Produce entity-relationship diagrams |
| Break down all envisaged tasks into sub-tasks (process decomposition). |
| Create a well defined schedule and work plan to show how each task will be carried out. |
| Specify the solution clearly enough to allow a competent third party to implement it. Include diagrams for reports, forms, tables, queries, relationships, etc. Fully annotate the diagrams to show validation, fonts, field sizes, default values, data capture methods, security features and so on. Make sure you design the *processes* as well as the inputs and the outputs. |
| Refer back to your requirements specification and ensure that your design meets your user's requirements. |
| Create a full and effective testing plan with a comprehensive selection of test data. Give reasons for your choice of this test data. Make it obvious that your testing plan is inspired by your requirements specification. Remember that you are testing the flow of data, not just whether a particular button works or not. |
| Design the user interface in relation to the experience/skill level of the end user(s) |
| Design online/offline help |
| Design a backup system (remember that you are thinking here about the backup system to be used by your end user - it should go without saying that you should backup your project). |

| |
|---|
| Design any inter-relationship between software packages |
| Include navigational diagrams |
| Consider prototyping |
| Consider the "management of change" |

| **Implementation (15 marks)** |
|---|
| All or most of the facilities of the software and the hardware must be fully exploited |
| Progression of work must be shown by printing out regularly and annotating fully |
| If wizards are used, should how they have been modified. |
| Commentary should explain what decisions you made and why you made them. |
| Print out and annotate VBA code or expressions. |
| Include a "data dictionary" |
| Organise appropriate training |

| **User Guide (8 marks)** |
|---|
| Include fully illustrated and comprehensive user guide, with examples of screen displays, explanation of validation restrictions, user interface, error messages that the user might encounter, etc. Wording should be sympathetic (avoid jargon). |
| Make sure you include sections on installation, backup procedures, general use and troubleshooting. |
| User Guide should be bound separately from the rest of the project. It should be well organised. Include contents page and index. Number the pages. |
| Hand the User Guide to your end-user *before* the Testing stage. |

| **Testing (15 marks)** |
|---|
| Clear evidence of end-user involvement in testing (not just a questionnaire) |
| Test outputs should be fully annotated and cross-referenced. |
| Test typical, extreme and erroneous data |
| Testing should show appreciation of different circumstances (e.g. the difference between a standalone computer and one on a network) |
| Explain the reason for each test |

| **Evaluation (10 marks)** |
|---|
| Consider clearly a full range of qualitative and quantitative criteria for evaluating the solution. Make it clear that these criteria relate to the requirement of the user(s). |
| What problems did you encounter and how did you overcome them? |
| How could your system be developed/extended |
| Explain any differences between your original design and the system you eventually produced. |
| Show involvement of the end-user in the evaluation stage. |
| A "letter of acceptance" from your end-user is required. |

| **Report (8 marks)** |
|---|
| This is not a separate section of your project. These eight marks are awarded for the general quality of your documentation. Ensure that you have spell-checked your project, that you have included referenced diagrams and that you have a well-organised folder. Include a contents page and an index and number your pages. |
| The report should be accurate and concise and it should not exceed 4,000 words (excluding annotations, screen dumps, diagrams, etc.) |